Manual do usuário

AWS Modernização do mainframe

AWS Modernização do mainframe: Manual do usuário

Table of Contents

O que é modernização AWS do mainframe?

AWS A modernização do mainframe ajuda você a modernizar seus aplicativos de mainframe para AWS ambientes de tempo de execução gerenciados. Ele fornece ferramentas e recursos para ajudar você a planejar e implementar a migração e a modernização. Você pode analisar suas aplicações de mainframe existentes, desenvolvê-los ou atualizá-los usando COBOL ou PL/I e implementar um pipeline automatizado para integração contínua e entrega contínua (CI/CD) das aplicações. Você pode escolher entre padrões automatizados de refatoração e redefinição da plataforma, dependendo das necessidades de seus clientes. Se você é um consultor ajudando um cliente a migrar suas cargas de trabalho de mainframe, você pode usar as ferramentas de modernização de AWS mainframe em todas as fases da jornada de migração e modernização, desde o planejamento inicial até as operações de nuvem pós-migração.

Você pode usar a modernização do AWS mainframe para ajudá-lo a criar e gerenciar com eficiência o ambiente de execução de seus aplicativos de mainframe, bem como para gerenciar e monitorar seus aplicativos modernizados. AWS

Tópicos

> **Note**
>
> Você se envolveu com parceiros de competência em migração de AWS mainframe ou serviços AWS profissionais para seu projeto de modernização de mainframe? Caso contrário, é altamente recomendável que você contrate especialistas para o seu projeto.
>
> - AWS Parceiros de competência em modernização de mainframe
> - AWS Professional Services

Os recursos e os casos de uso da modernização do AWS mainframe oferecem suporte a uma abordagem de modernização evolutiva, que oferece ganhos de curto prazo ao melhorar a agilidade e muitas oportunidades para otimizar e inovar posteriormente. Para ter mais informações, consulte Abordagem de modernização.

Características da modernização do AWS mainframe

AWS Os recursos de modernização do mainframe oferecem suporte aos seguintes casos de uso:

- Avalie: o recurso de avaliação da modernização do AWS mainframe pode ajudá-lo a avaliar, definir o escopo e planejar um projeto de migração e modernização.
- Refatoração: com tecnologia AWS Blu Age, você pode usar a refatoração para converter linguagens de programação de aplicativos legadas, criar macrosserviços ou microsserviços e modernizar interfaces de usuário (UIs) e pilhas de software de aplicativos.

  AWS O Blu Insights agora está disponível a AWS Management Console partir do login único. Você não precisa mais gerenciar credenciais separadas do AWS Blu Insights. Você pode acessar os recursos do AWS AWS Blu Age Codebase e do Transformation Center diretamente do. AWS Management Console
- Redefinição de plataforma: com a solução Micro Focus Enterprise, você pode portar a aplicação para o qual grande parte do código-fonte da aplicação é recompilada sem alterações.
- IDE do desenvolvedor: a modernização do AWS mainframe oferece um ambiente de desenvolvimento integrado (IDE) sob demanda para que os desenvolvedores possam escrever código mais rapidamente com edição e depuração inteligentes, compilação instantânea de código e testes unitários.
- Tempo de execução gerenciado: o ambiente de tempo de execução gerenciado da modernização do AWS mainframe monitora continuamente seus clusters para manter as cargas de trabalho corporativas funcionando com computação autorrecuperável e escalabilidade automatizada.
- Integração e entrega contínuas (CI/CD): o recurso CI/CD da AWS Mainframe Modernization ajuda as equipes de desenvolvimento de aplicativos a realizar alterações de código com mais frequência e confiabilidade, o que acelera a velocidade de migração, aumenta a qualidade e ajuda a reduzir o lançamento de novas funções de negócios. time-to-market
- Integrações com outros AWS serviços: a modernização do AWS mainframe oferece suporte AWS CloudFormation AWS PrivateLink, e AWS Key Management Service para implantação repetível e maior segurança e conformidade.

- Disponibilidade ampliada: a modernização do AWS mainframe agora está disponível no Leste dos EUA (Ohio), Oeste dos EUA (Norte da Califórnia), Ásia-Pacífico (Mumbai), Ásia-Pacífico (Seul), Ásia-Pacífico (Cingapura), Ásia-Pacífico (Tóquio), Europa (Londres) e Europa (Paris).

Para obter mais informações sobre os recursos de modernização do AWS mainframe, consulte. https://aws.amazon.com/mainframe-modernization/features/

Padrões

O padrão Automated Refactoring, desenvolvido pela AWS Blu Age, está focado em acelerar a modernização, convertendo a pilha completa de aplicativos legados e sua camada de dados em um aplicativo moderno baseado em Java, preservando a equivalência funcional. Durante essa transformação automatizada, ele cria uma aplicação de várias camadas com um front-end baseado em Angular, um back-end Java habilitado para API e uma camada de dados que acessa armazenamentos de dados modernos. O processo de refatoração fornece funcionalidade equivalente à pilha antiga para aumentar a automação do projeto, resultando em velocidade, qualidade e menor custo para obter benefícios comerciais mais rapidamente. Para obter mais informações, consulte AWS Mainframe Modernization Automated Refactor.

O padrão de redefinição da plataforma, desenvolvido pela suíte Micro Focus Enterprise, tem como foco preservar a linguagem, o código e os artefatos da aplicação a fim de minimizar o impacto nos ativos e nas equipes da aplicação. Ele ajuda os clientes a manter o conhecimento e as habilidades da aplicação. Embora as mudanças nas aplicações sejam limitadas, esse padrão também facilita a modernização da infraestrutura e dos processos. A infraestrutura é alterada para um serviço gerenciado moderno baseado em nuvem, enquanto os processos são alterados para seguir as melhores práticas de desenvolvimento de aplicações e operações de TI. Para obter mais informações, consulte AWS Mainframe Modernization Replatform.

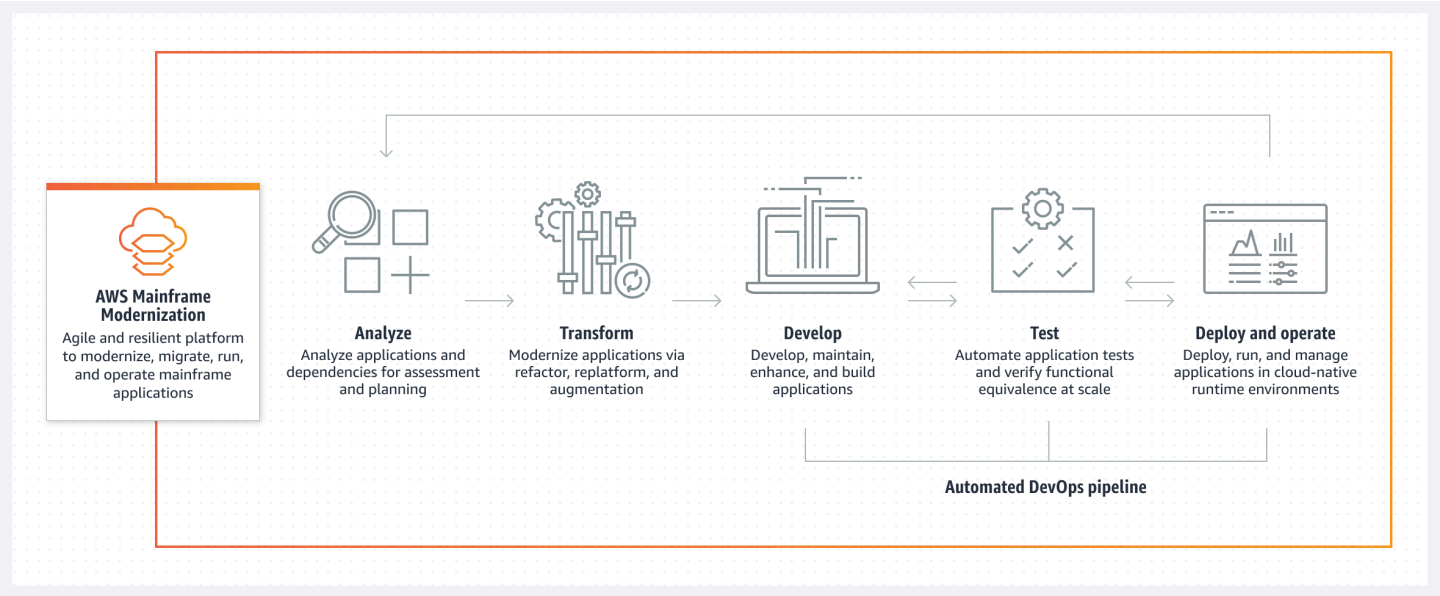
Como começar a modernizar o AWS mainframe

Experimente! Oferecemos tutoriais e exemplos de aplicativos para ajudar você a ter uma ideia do que a modernização do AWS mainframe oferece. Escolha o Tutorial: Configurar o tempo de execução gerenciado para o AWS Blu Age ou o Tutorial: Configurar o tempo de execução gerenciado para a Micro Focus para obter um step-by-step tutorial completo.

Se você estiver interessado em refatoração automatizada, confira as ferramentas do AWS Blu Age em. BluInsights Você também pode configurar o AppStream 2.0 para acessar o AWS Blu

Age Developer IDE ou as ferramentas Micro Focus Enterprise Analyzer e Micro Focus Enterprise Developer.

Os tutoriais e os aplicativos de amostra dão apenas uma ideia do que a modernização do AWS mainframe oferece. Quando você estiver pronto para iniciar um projeto de modernização, consulte Abordagem de modernização para obter detalhes sobre os estágios e tarefas de um projeto de modernização.



Serviços relacionados

Além do Blu Insights para refatoração automatizada, você pode usar os seguintes AWS serviços com a modernização do mainframe. AWS

- Amazon RDS para hospedar seus bancos de dados migrados.
- Amazon S3 para armazenar binários de aplicações e arquivos de definição.
- Amazon FSx ou Amazon EFS para armazenar dados de aplicações.
- Amazon AppStream para acesso às ferramentas Micro Focus Enterprise Analyzer e Micro Focus Enterprise Developer.
- AWS CloudFormation para o DevOps pipeline automatizado que você pode usar para configurar o CI/CD para seus aplicativos migrados.
- AWS Migration Hub
- AWS DMS para migrar seus bancos de dados.

Acessando a AWS modernização do mainframe

Atualmente, você pode acessar a Modernização do AWS Mainframe por meio do console em https://console.aws.amazon.com/m2/. Para obter uma lista das regiões em que a modernização de AWS mainframe está disponível, consulte endpoints e cotas de modernização de AWS mainframe no. Referência geral da Amazon Web Services

Você é um usuário iniciante de modernização de AWS mainframe?

Se você é um usuário iniciante da modernização de AWS mainframe, recomendamos que comece lendo as seguintes seções:

- Conceitos Básicos
- Configuração

Definição de preço

AWS A modernização do mainframe cobra pelo uso de instâncias que dão suporte aos ambientes de tempo de execução gerenciados. Além disso, a modernização do AWS mainframe oferece algumas ferramentas sem custos adicionais. Você é responsável pelas taxas incorridas por outros AWS serviços que você usa em conexão com a modernização do AWS mainframe. AWS fornecerá um aviso prévio de 30 dias antes que qualquer alteração de preço entre em vigor para o uso da modernização do AWS mainframe. Para obter mais informações, consulte Modernização de mainframe com. AWS

Com o AWS Blu Insights, você paga pelo uso do Transformation Center. Para obter mais informações, consulte Preço do AWS Mainframe Modernization.

Configurar a modernização AWS do mainframe

Antes de começar a usar a Modernização do AWS Mainframe, você ou seu administrador precisam concluir algumas etapas.

Tópicos

Inscreva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra https://portal.aws.amazon.com/billing/signup.
2. Siga as instruções online.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica e inserir um código de verificação no teclado do telefone.

   Quando você se inscreve em um Conta da AWS, um Usuário raiz da conta da AWSé criado. O usuário raiz tem acesso a todos os Serviços da AWS e atributos na conta. Como prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que exigem acesso de usuário-raiz.

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, é possível visualizar as atividades da conta atual e gerenciar sua conta acessando https://aws.amazon.com/ e selecionando Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em um Conta da AWS, proteja seu Usuário raiz da conta da AWS AWS IAM Identity Center, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

Proteja seu Usuário raiz da conta da AWS

1. Faça login AWS Management Consolecomo proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, insira sua senha.

   Para obter ajuda ao fazer login usando o usuário raiz, consulte Fazer login como usuário raiz no Guia do usuário do Início de Sessão da AWS .
2. Habilite a autenticação multifator (MFA) para o usuário raiz.

   Para obter instruções, consulte Habilitar um dispositivo de MFA virtual para seu usuário Conta da AWS raiz (console) no Guia do usuário do IAM.

Criar um usuário com acesso administrativo

1. Habilitar o IAM Identity Center.

   Para obter instruções, consulte Habilitar AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center .
2. No Centro de Identidade do IAM, conceda o acesso administrativo para um usuário.

   Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte Configurar o acesso do usuário com o padrão Diretório do Centro de Identidade do IAM no Guia AWS IAM Identity Center do usuário.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com seu usuário do Centro de Identidade do IAM, use o URL de login que foi enviado ao seu endereço de e-mail quando você criou o usuário do Centro do Usuário do IAM.

  Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte Como fazer login no portal de AWS acesso no Guia Início de Sessão da AWS do usuário.

Atribuir acesso a usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

Para obter instruções, consulte Create a permission set no Guia do usuário do AWS IAM Identity Center .

2. Atribua usuários a um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

   Para obter instruções, consulte Add groups no Guia do usuário do AWS IAM Identity Center .

Começando com a modernização AWS do mainframe

Você pode começar com a modernização do AWS mainframe seguindo os tutoriais que apresentam o serviço e cada mecanismo de tempo de execução.

Tópicos

Para continuar aprendendo, consulte os tutoriais a seguir.

Tutorial: Configurar o tempo de execução gerenciado para o AWS Blu Age

Você pode implantar um aplicativo modernizado AWS Blu Age em um ambiente de tempo de execução de modernização de AWS mainframe com um aplicativo de demonstração especificado neste tutorial.

Tópicos

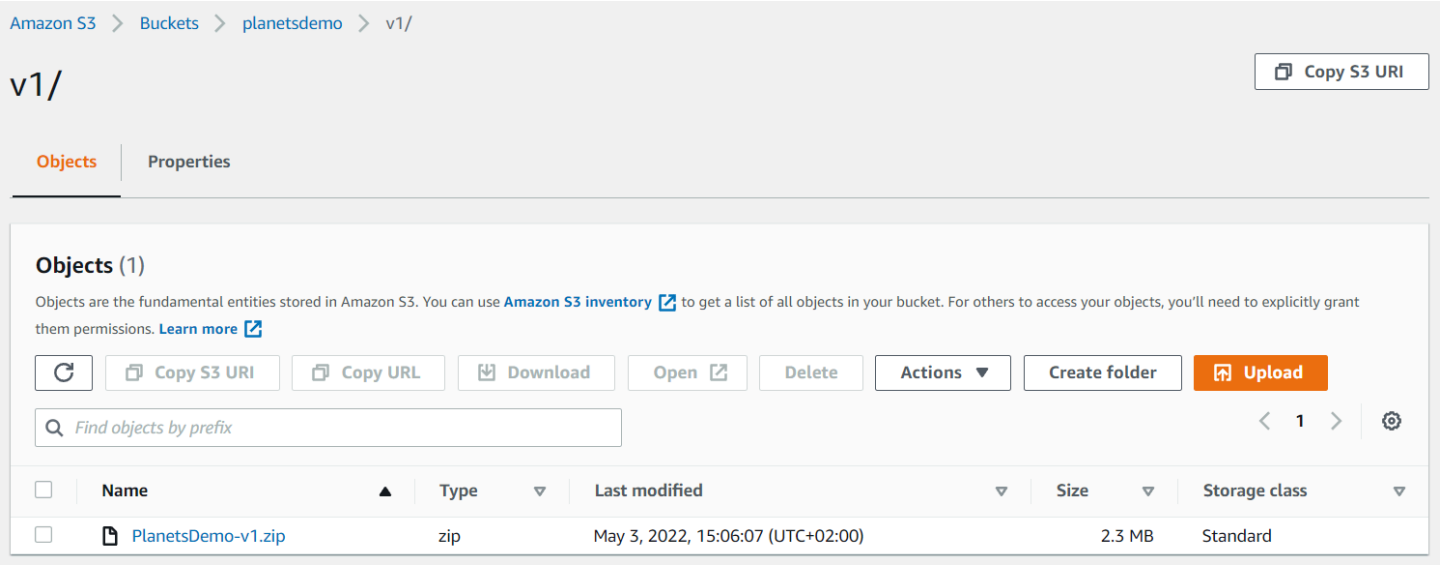
Pré-requisitos

Para concluir este tutorial, baixe o arquivo de aplicativos de demonstração PlanetsDemo-v1.zip.

A aplicação de demonstração em execução requer um navegador moderno para acesso. Se você executa esse navegador a partir do seu desktop ou de uma instância do Amazon Elastic Compute Cloud, por exemplo, dentro da VPC, determina suas configurações de segurança.

Etapa 1: fazer o upload da aplicação de demonstração

Faça upload da aplicação de demonstração em um bucket do Amazon S3. Certifique-se de que esse bucket esteja na mesma Região da AWS em que você implantará a aplicação. O exemplo a seguir mostra um bucket chamado planetsdemo, com um prefixo de chave, ou pasta, chamado v1 e um arquivo chamado `planetsdemo-v1.zip`.



> **Note**
>
> A pasta no bucket é obrigatória.

Etapa 2: criar a aplicação

Para implantar um aplicativo no tempo de execução gerenciado, você precisa de uma definição de aplicativo de modernização de AWS mainframe. Essa definição é um arquivo JSON que descreve a localização e as configurações da aplicação. O exemplo a seguir é uma definição de aplicação desse tipo para a aplicação de demonstração:

```
{
    "template-version": "2.0",
    "source-locations": [{
        "source-id": "s3-source",
        "source-type": "s3",
        "properties": {
            "s3-bucket": "planetsdemo",
            "s3-key-prefix": "v1"
        }
    }],
    "definition": {
        "listeners": [{
            "port": 8196,
            "type": "http"
        }],
        "ba-application": {
            "app-location": "${s3-source}/PlanetsDemo-v1.zip"
        }
    }
}
```

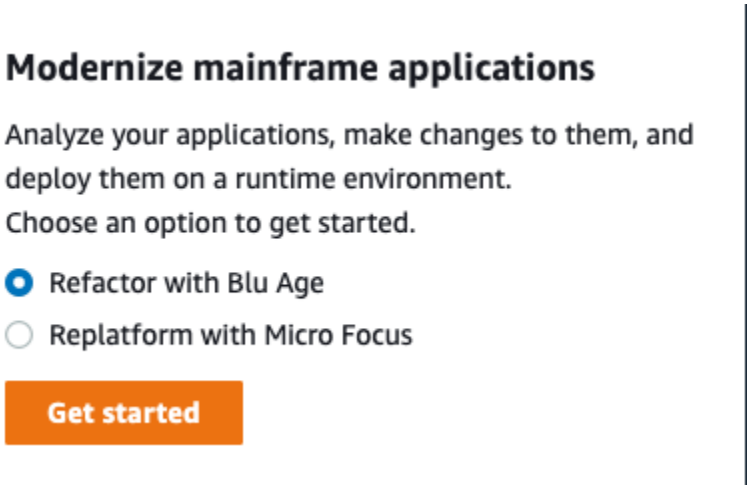
Altere a entrada `s3-bucket` para o nome do bucket em que você armazenou o arquivo zip da aplicação de amostra.

Para obter mais informações sobre a definição da aplicação, consulte AWS Exemplo de definição do aplicativo Blu Age.

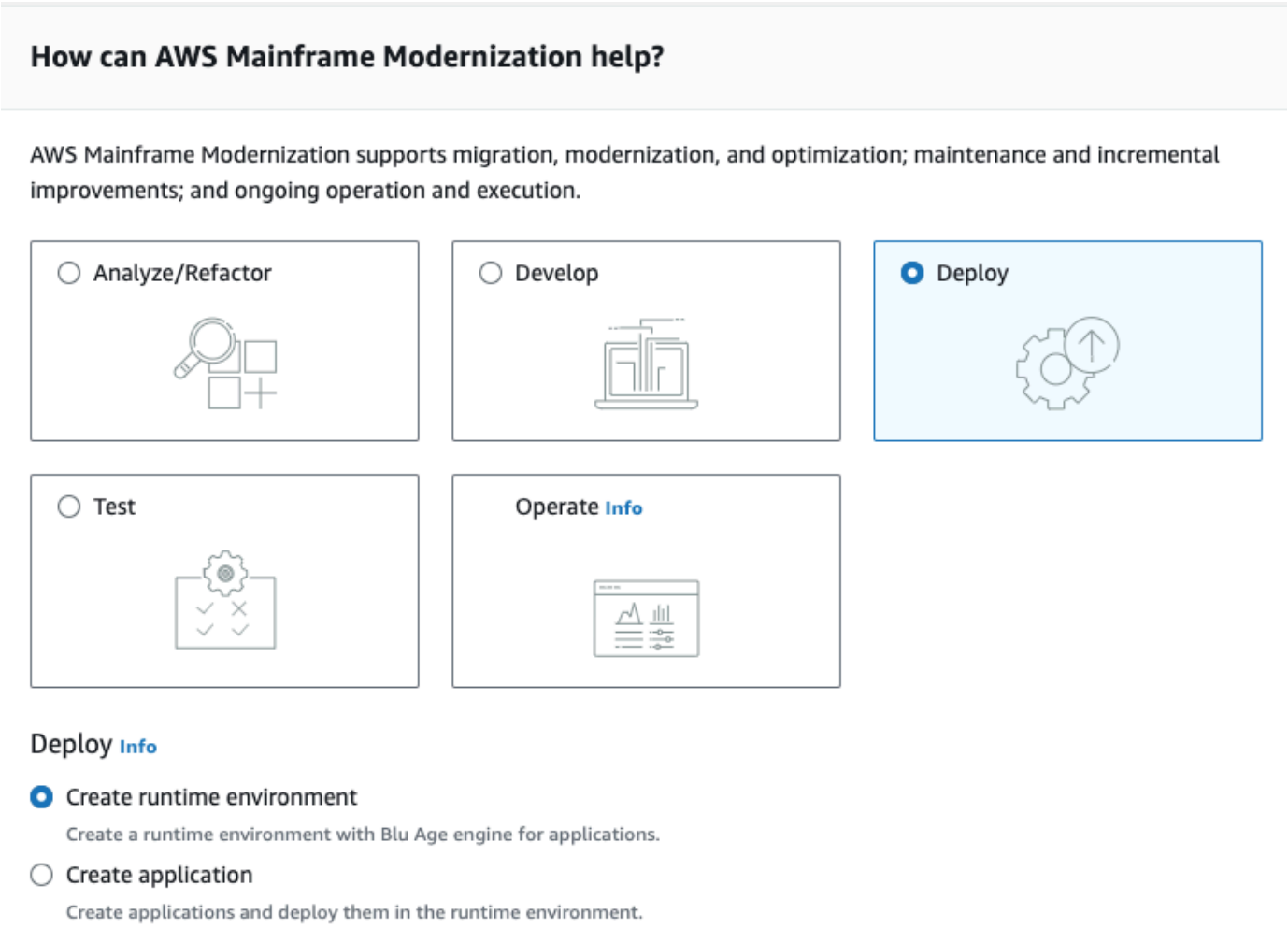
Etapa 3: criar um ambiente de runtime

Para criar o ambiente de execução da modernização do AWS mainframe, execute as seguintes etapas:

1. Abra o console do AWS Mainframe Modernization.
2. No Região da AWS seletor, escolha a região em que você deseja criar o ambiente. Isso Região da AWS deve corresponder à região em que você criou o bucket do S3. Etapa 1: fazer o upload da aplicação de demonstração
3. Em Modernizar aplicações de mainframe, escolha Refatorar com Blu Age e, em seguida, escolha Começar.

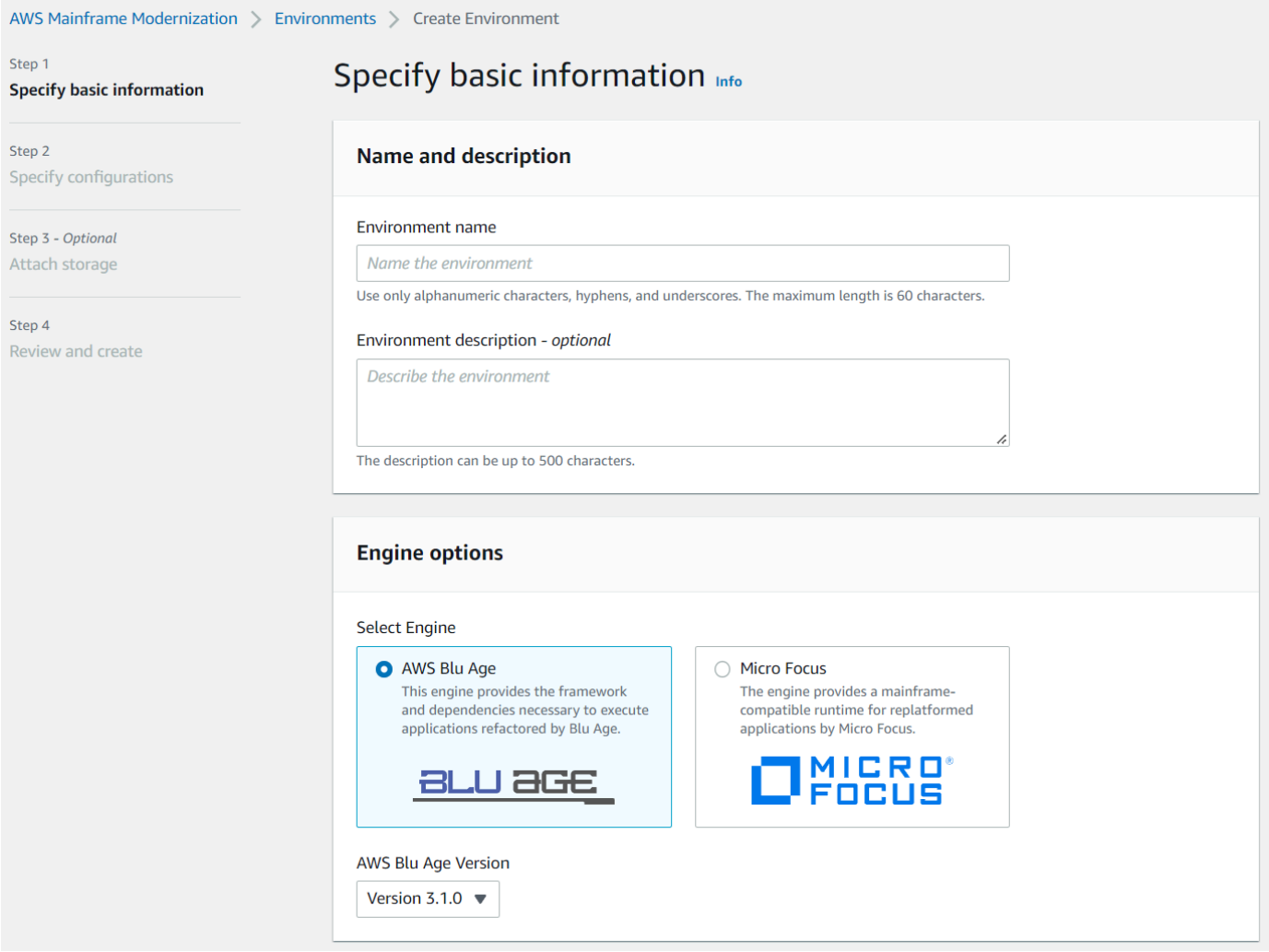


4. Em Como o AWS Mainframe Modernization pode ajudar, escolha Implantar e Criar ambiente de runtime.

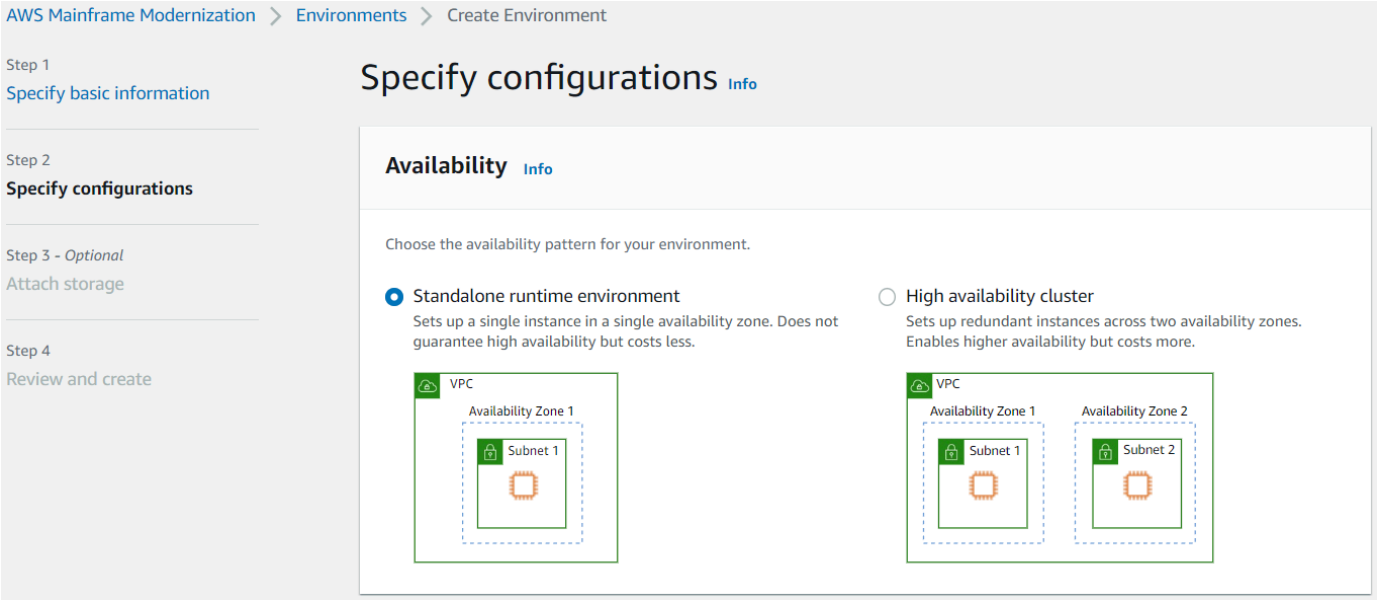


5. No painel de navegação, escolha Ambientes de computação, Criar ambiente. Na página Especificar informações básicas, insira um nome e uma descrição para seu ambiente e, em

seguida, certifique-se de que o mecanismo AWS Blu Age esteja selecionado. Opcionalmente, você pode adicionar etiquetas ao recurso criado. Em seguida, escolha Próximo.

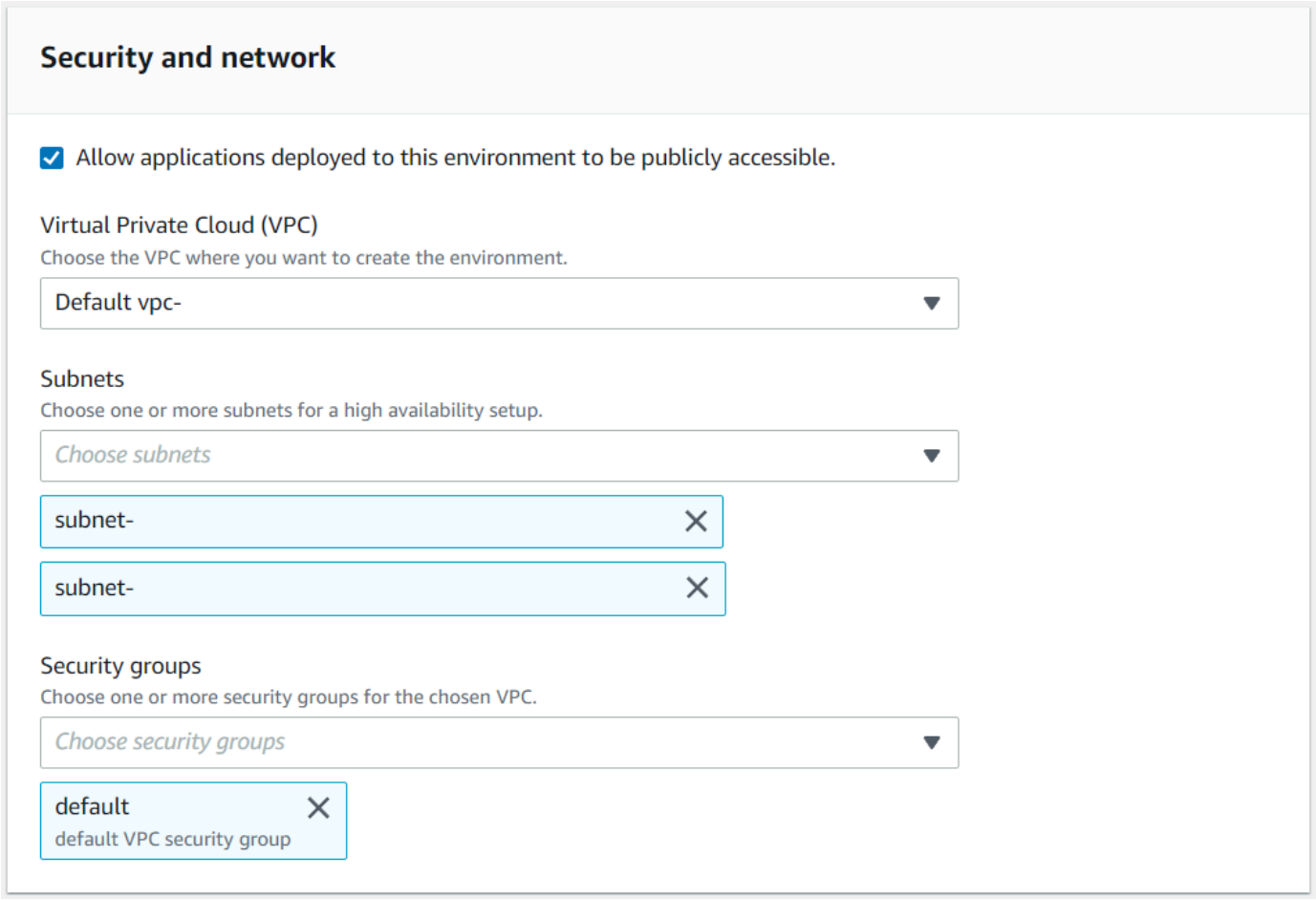


6. Na página Especificar configurações, escolha Ambiente de runtime autônomo.



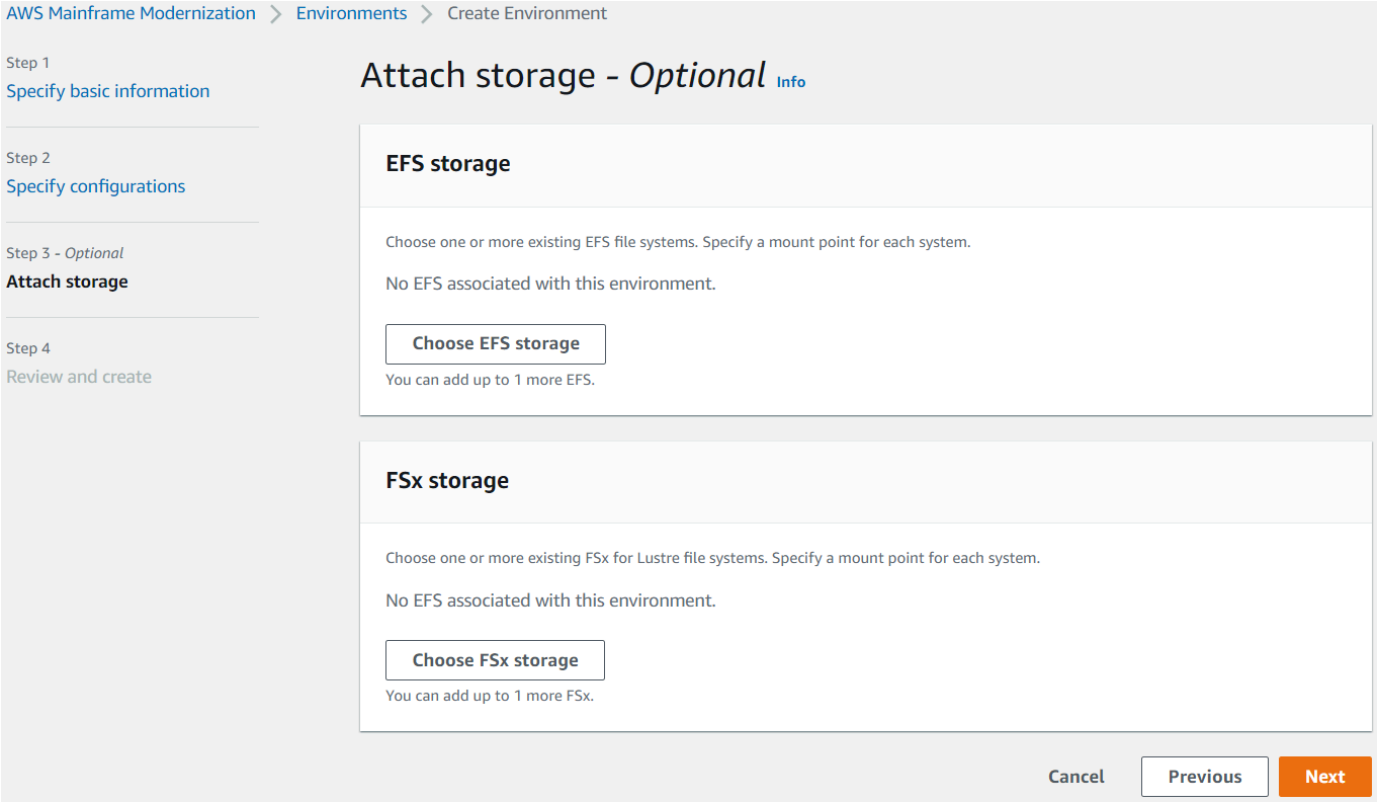
7. Em Segurança e rede, faça as seguintes alterações:

   - Escolha Permitir que as aplicações implantadas nesse ambiente sejam acessíveis ao público. Essa opção atribui um endereço IP público à aplicação para que você possa acessá-la do seu desktop.
   - Escolha uma VPC. Você pode usar o Padrão.
   - Escolha duas sub-redes. Certifique-se de que as sub-redes permitam a atribuição de endereços IP públicos.
   - Escolha um grupo de segurança. Você pode usar o Padrão. Certifique-se de que o grupo de segurança escolhido permita o acesso do endereço IP do navegador à porta especificada na `listener` propriedade da definição da aplicação. Para ter mais informações, consulte Etapa 2: criar a aplicação.



Se você quiser acessar a aplicação de fora da VPC escolhida, certifique-se de que as regras de entrada dessa VPC estejam configuradas corretamente. Para ter mais informações, consulte Não consigo acessar o URL de um aplicativo.

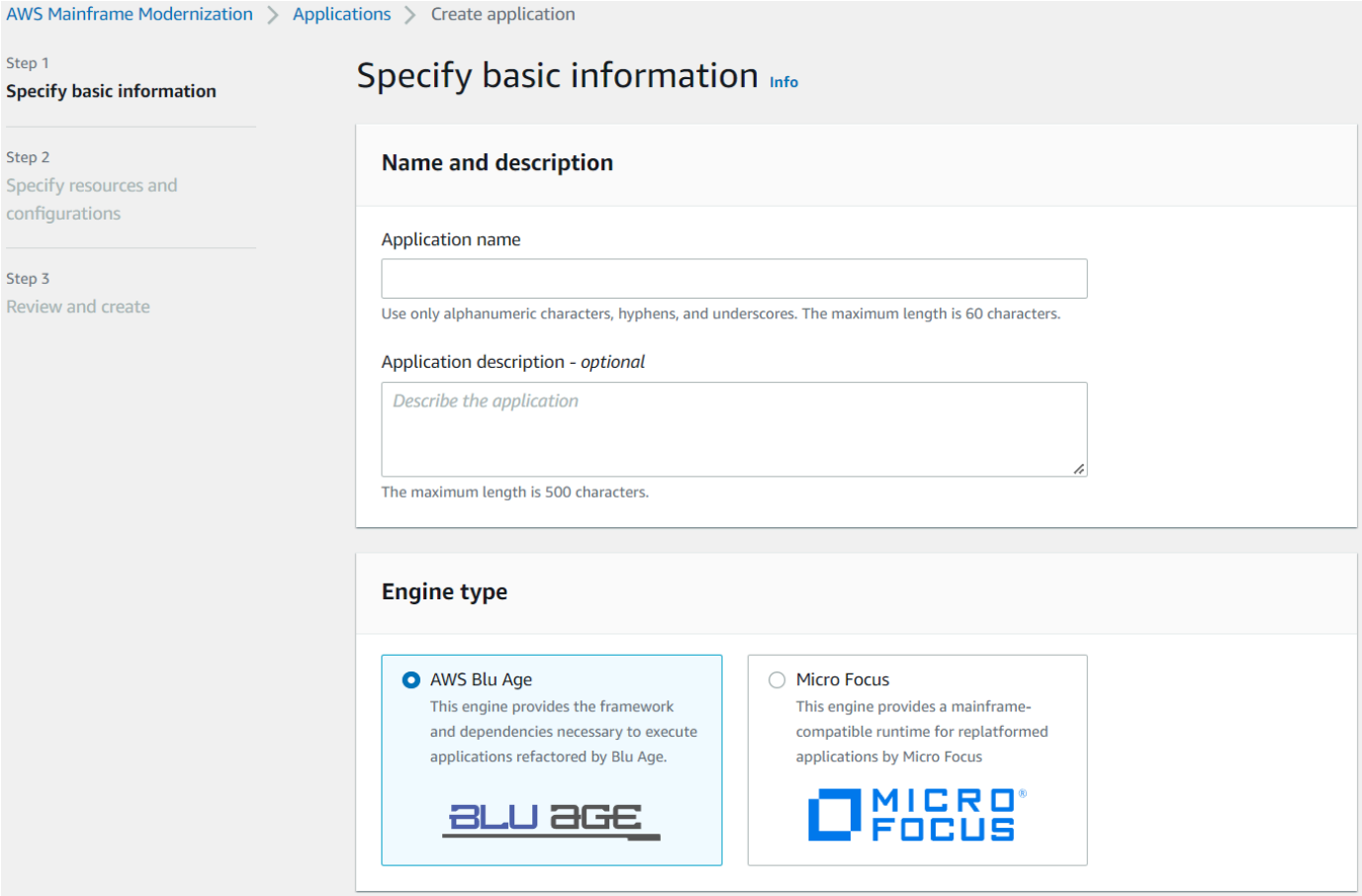
8. Escolha Próximo.
9. Em Anexar armazenamento - Opcional, deixe as seleções padrão e escolha Próximo.



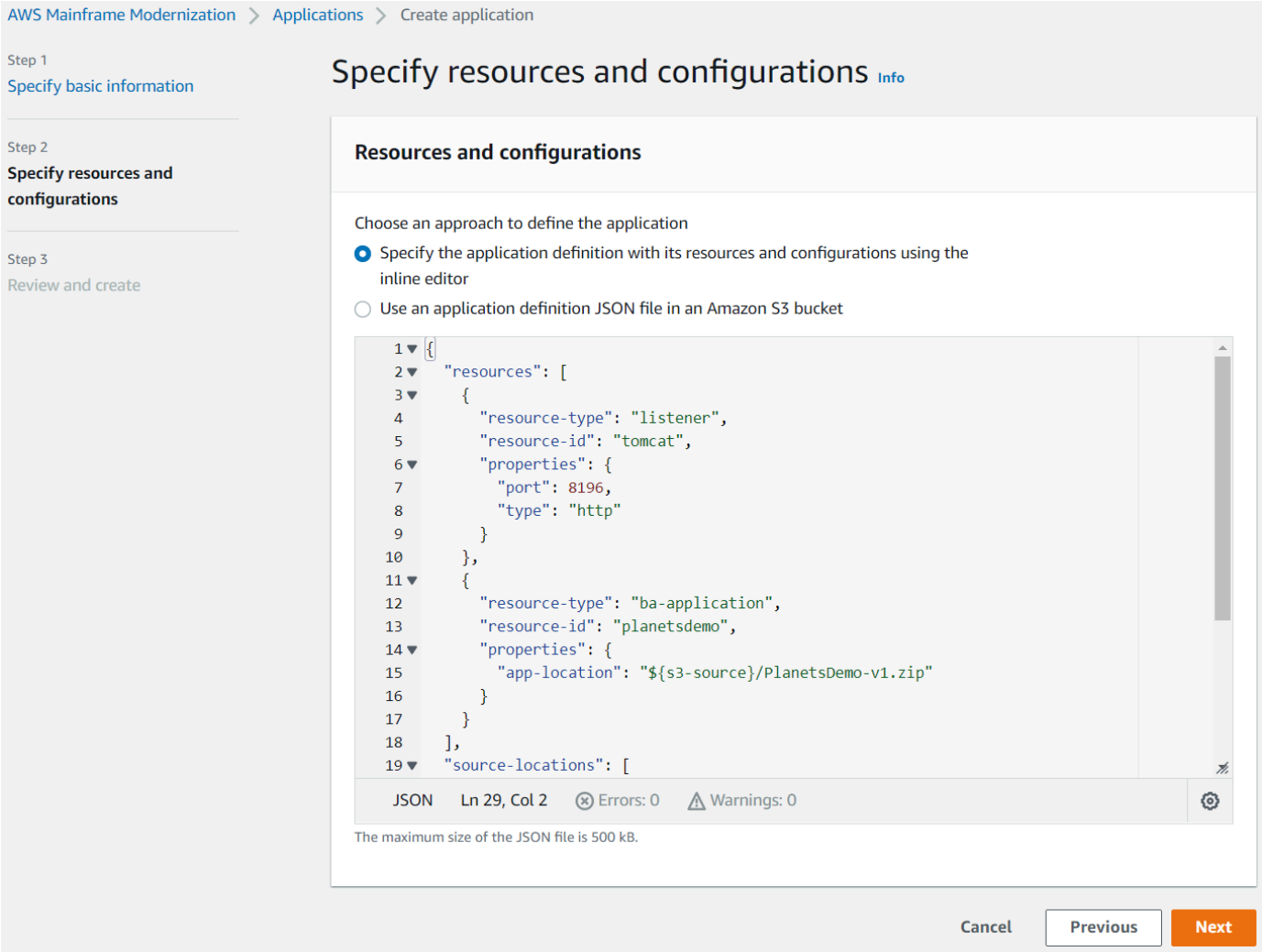
10. Em Agendar manutenção, escolha Sem preferência e, em seguida, escolha Próximo.
11. Em Revisar e criar, revise as informações e escolha Criar ambiente.

Etapa 4: criar uma aplicação

1. Navegue até o AWS Mainframe Modernization no AWS Management Console.
2. No painel de navegação, escolha Aplicações e Criar aplicação. Na página Especificar informações básicas, insira um nome e uma descrição para a aplicação e certifique-se de que o mecanismo AWS Blu Age esteja selecionado. Em seguida, escolha Próximo.



3. Na página Especificar recursos e configurações, copie e cole o JSON de definição da aplicação atualizado que você criou no the section called “Etapa 2: criar a aplicação”.

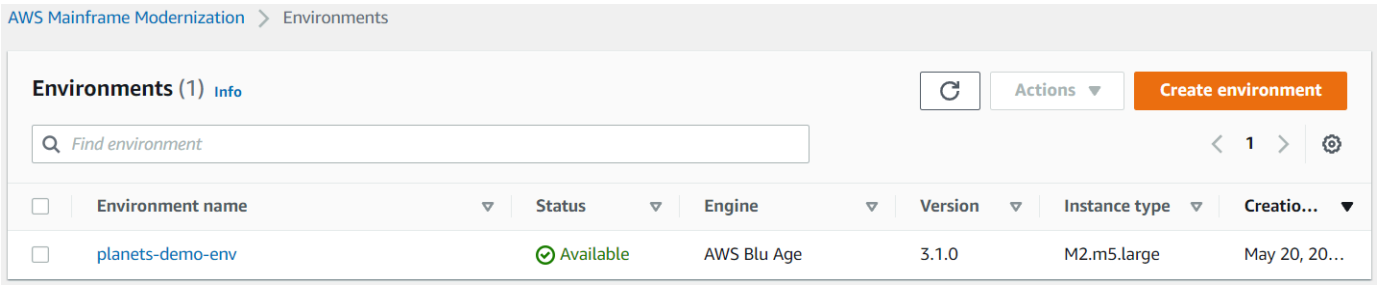


4. Na página Revisar e criar, revise suas escolhas e, em seguida, selecione Criar aplicação.

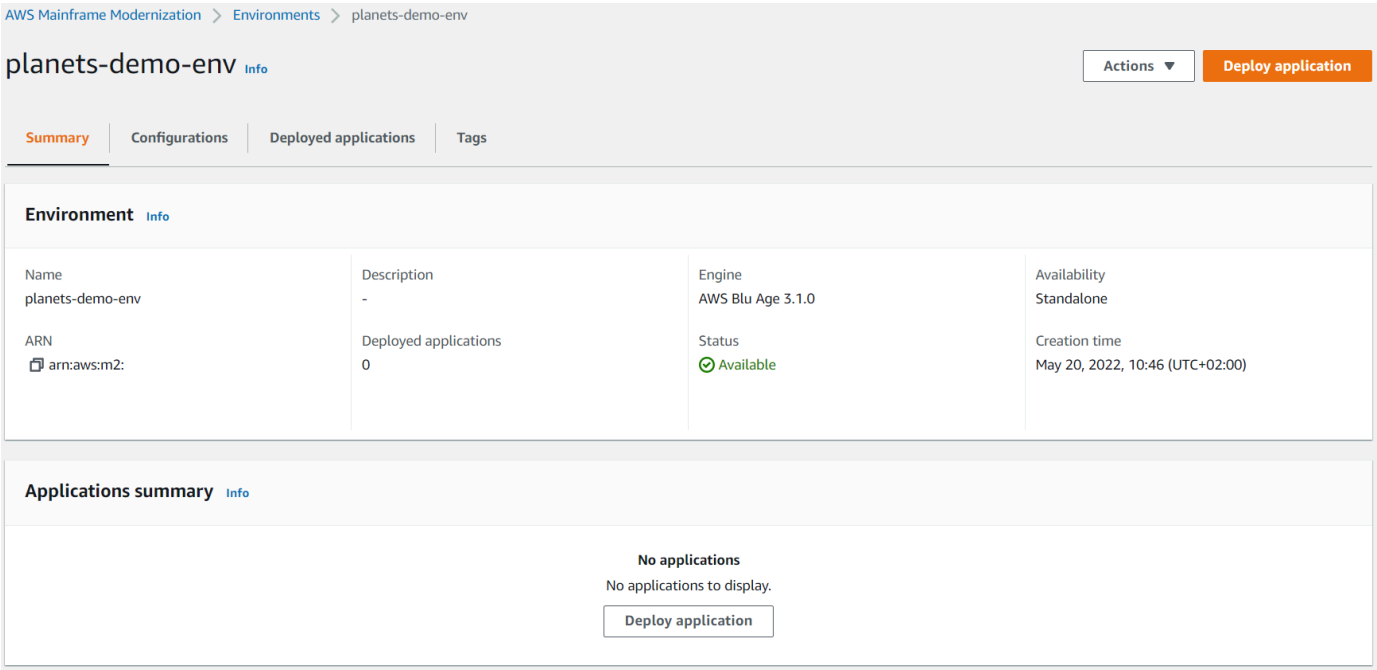
Etapa 5: implantar uma aplicação

Depois de criar com sucesso o ambiente de execução e o aplicativo de modernização de AWS mainframe, e ambos estarem no estado Disponível, você poderá implantar o aplicativo no ambiente de tempo de execução. Para fazer isso, conclua as seguintes etapas:

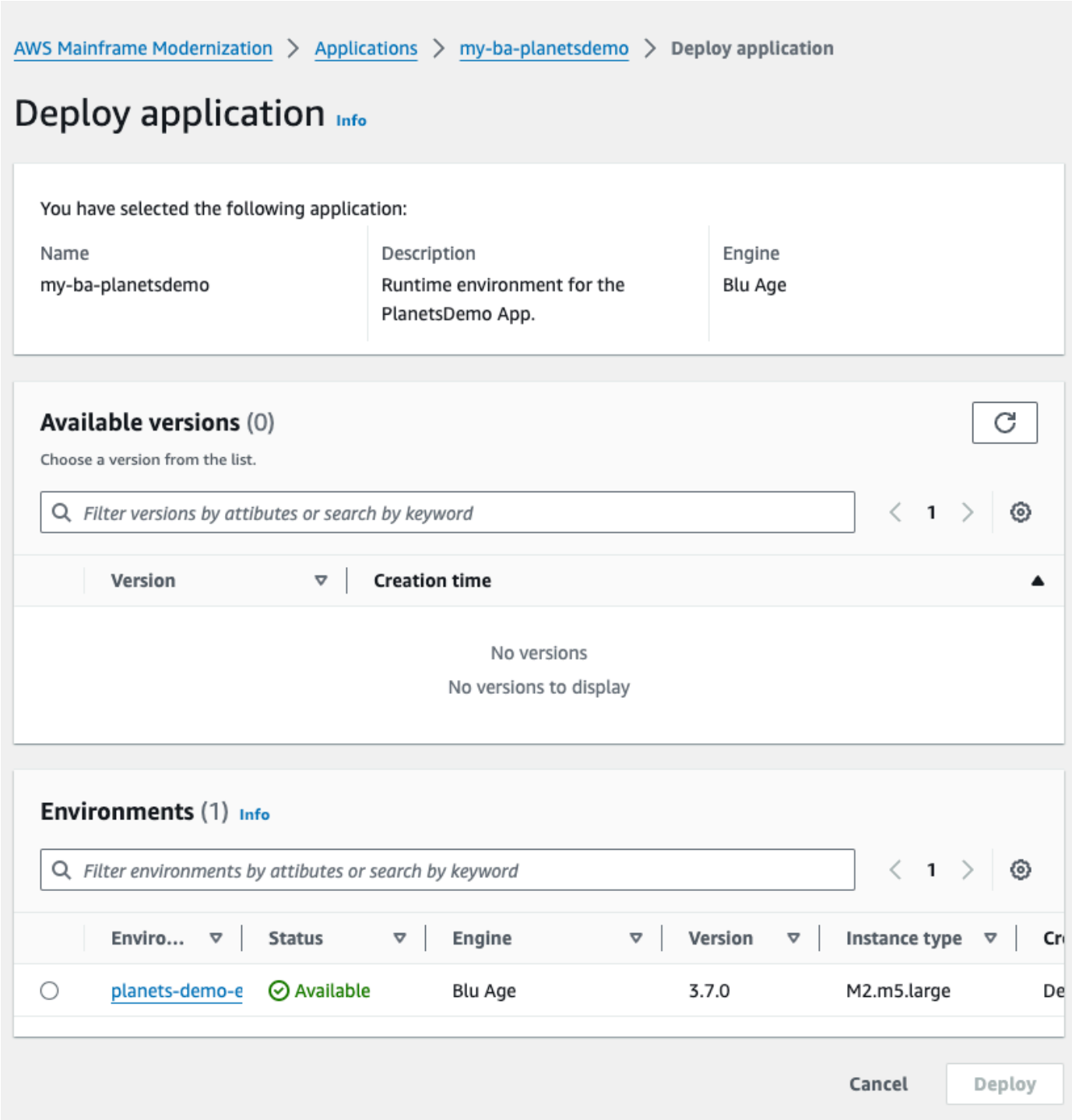
1. Navegue até a modernização do AWS Mainframe no console de AWS gerenciamento. No painel de navegação, escolha Ambientes. A página da lista de ambientes é exibida.



2. Escolha o ambiente de runtime criado anteriormente. A página do ambiente é exibida.
3. Escolha Implantar aplicação.



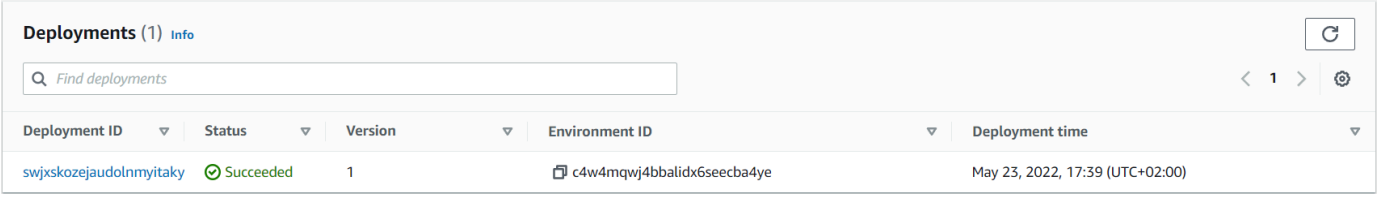
4. Escolha a aplicação criada anteriormente e, em seguida, escolha a versão na qual você deseja implantar a aplicação. Selecione Deploy (Implantar).



5. Espere até que a aplicação conclua a implantação. Você verá um banner com a mensagem A aplicação foi implantada com sucesso.

Etapa 6: iniciar uma aplicação

1. Navegue até AWS Mainframe Modernization em AWS Management Console e escolha Applications.
2. Vá até a página da aplicação e escolha Implantar. O status da aplicação deve ser Sucedido.



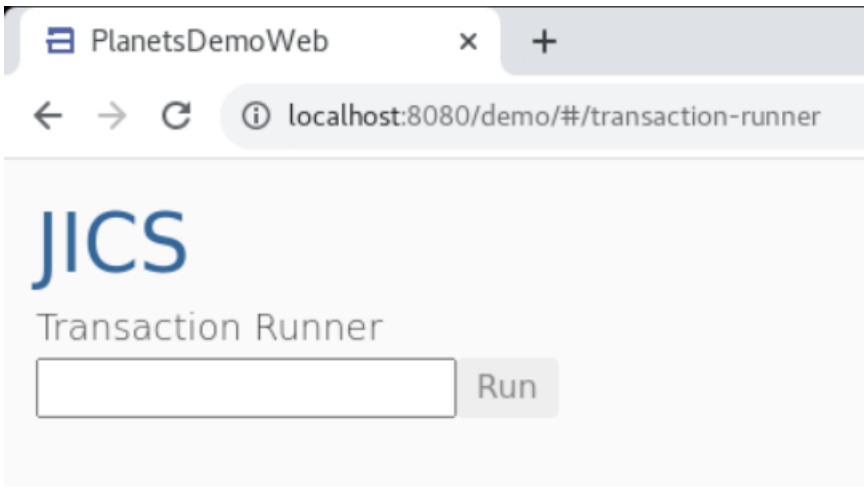
3. Escolha Ações > Iniciar aplicação.

Etapa 7: acessar a aplicação

1. Espere até que a aplicação esteja no estado Executando. Você verá um banner com a mensagem A aplicação foi iniciado com sucesso.
2. Copie o nome de host DNS da aplicação. Você pode encontrar esse nome de host na seção Informações da aplicação.
3. Em um navegador, acesse `http://{hostname}:{portname}/PlanetsDemo-web-1.0.0/`, em que:

   - `hostname` é o nome do host DNS copiado anteriormente.
   - `portname` é a porta Tomcat definida na definição da aplicação que você criou em Etapa 2: criar a aplicação.

   A tela JICS é exibida.



Se você não conseguir acessar a aplicação, consulte Não consigo acessar o URL de um aplicativo.

> **Note**
>
> Se a aplicação não estiver acessível e a regra de entrada no grupo de segurança tiver “Meu IP” selecionado na porta 8196, especifique a regra para permitir o tráfego de LB i/p na porta 8196.

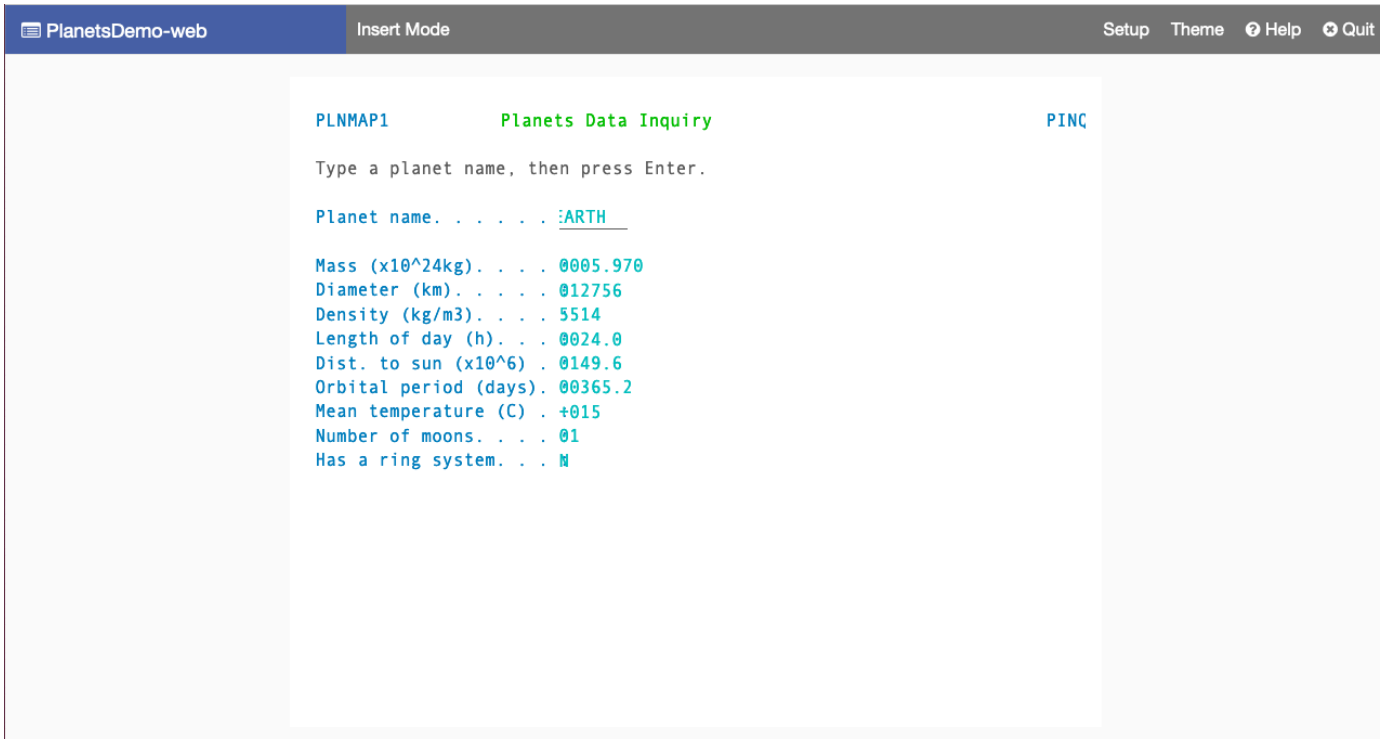
Etapa 8: Testar o aplicativo

Nesta etapa, você executa uma transação na aplicação migrada.

1. Na tela JICS, insira `PINQ` no campo de entrada e escolha Executar (ou pressione Enter) para iniciar a transação da aplicação.

   A tela da aplicação de demonstração deve aparecer.

2. Digite o nome do planeta no campo correspondente e pressione Enter.



Você deve ver detalhes sobre o planeta.

Limpar os recursos

Se você não precisar mais dos recursos que criou para este tutorial, exclua-os para evitar cobranças adicionais. Para fazer isso, realize as etapas a seguir:

- Se o aplicativo de modernização do AWS mainframe ainda estiver em execução, pare-o.
- Exclua o aplicativo do . Para ter mais informações, consulte Excluir um aplicativo de modernização AWS de mainframe.
- Exclua o ambiente de runtime. Para ter mais informações, consulte Excluir um ambiente de AWS execução de modernização de mainframe.

Tutorial: Configurar o tempo de execução gerenciado para a Micro Focus

Você pode implantar e executar um aplicativo no ambiente de tempo de execução gerenciado de modernização de AWS mainframe com o mecanismo de tempo de execução da Micro Focus. Este tutorial mostra como implantar e executar o aplicativo de CardDemo amostra em um ambiente de tempo de execução gerenciado de modernização de AWS mainframe com o mecanismo de tempo de execução da Micro Focus. O aplicativo CardDemo de amostra é um aplicativo simplificado de cartão de crédito desenvolvido para testar e mostrar a tecnologia de uma parceria para casos AWS de uso de modernização de mainframe.

No tutorial, você cria recursos em outros Serviços da AWS. Isso inclui o Amazon Simple Storage Service, o Amazon Relational Database Service AWS Key Management Service, AWS Secrets Manager e.

Tópicos

Pré-requisitos

- Verifique se você tem acesso a um emulador 3270 para usar a conexão CICS. Os emuladores 3270 gratuitos e de teste estão disponíveis em sites de terceiros. Como alternativa, você pode iniciar uma instância Micro Focus de Modernização de AWS Mainframe AppStream 2.0 e usar o emulador Rumba 3270 (não disponível gratuitamente).

  Para obter informações sobre AppStream 2.0, consultethe section called “Tutorial: Configuração AppStream 2.0 para Enterprise Analyzer e Enterprise Developer”.

  > **Note**
  >
  > Ao criar a pilha, escolha a opção Enterprise Developer (ED) e não Enterprise Analyzer (EA).

- Baixe o aplicativo CardDemo de amostra e descompacte o arquivo baixado em qualquer diretório local. Esse diretório conterá um subdiretório intituladoCardDemo.
- Identifique uma VPC em sua conta na qual você possa definir os recursos criados neste tutorial. A VPC precisará de sub-redes em pelo menos duas zonas de disponibilidade. Para obter mais informações sobre a Amazon VPC, consulte Como a Amazon VPC funciona.

Etapa 1: criar e carregar um bucket do Amazon S3

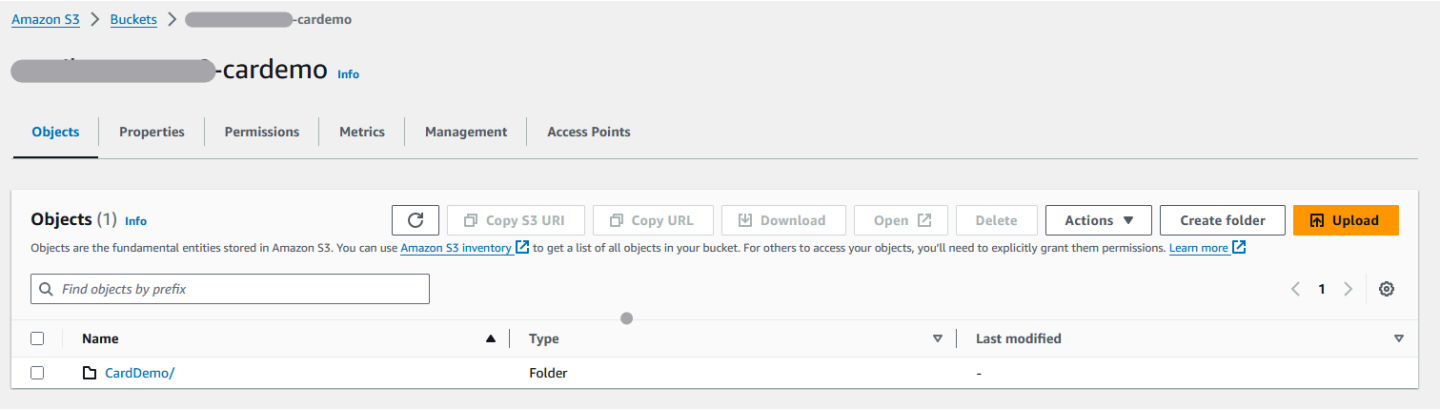
Nesta etapa, você cria um bucket do Amazon S3 e carrega CardDemo arquivos para esse bucket. Posteriormente neste tutorial, você usará esses arquivos para implantar e executar o aplicativo de CardDemo amostra em um ambiente Micro Focus Managed Runtime de modernização de AWS mainframe.

> **Note**
>
> Você não precisa criar um novo bucket do S3, mas o bucket escolhido deve estar na mesma região dos outros recursos usados neste tutorial.

Como criar um bucket do Amazon S3

1. Abra o console do Amazon S3 e escolha Create bucket.
2. Em Configuração geral, escolha a região da AWS em que você deseja criar o Micro Focus Managed Runtime de modernização de AWS mainframe.
3. Insira um nome de bucket, por exemplo,`yourname-aws-region-carddemo`. Mantenha as configurações padrão e escolha Create bucket. Como alternativa, você também pode copiar as configurações de um bucket Amazon S3 existente e, em seguida, escolher Create bucket.
4. Escolha o bucket que você acabou de criar e, em seguida, escolha Upload.
5. Na seção Carregar, escolha Adicionar pasta e, em seguida, navegue até o `CardDemo` diretório a partir do seu computador local.
6. Escolha Upload para iniciar o processo de upload. Os tempos de upload variam de acordo com a velocidade da sua conexão.
7. Quando o upload for concluído, confirme se todos os arquivos foram carregados com sucesso e escolha Fechar.

Seu bucket do Amazon S3 agora contém a `CardDemo` pasta.



Para obter informações sobre buckets do S3, consulte Criação, configuração e trabalho com buckets do Amazon S3.

Etapa 2: criar e configurar um banco de dados

Nesta etapa, você cria um banco de dados PostgreSQL no Amazon Relational Database Service (Amazon RDS). Para o tutorial, esse banco de dados contém os conjuntos de dados que o aplicativo de CardDemo amostra usa para tarefas do cliente relacionadas a transações com cartão de crédito.

Para criar um banco de dados no Amazon RDS

1. Abra o console do Amazon RDS.
2. Escolha a região da AWS na qual você deseja criar a instância do banco de dados.
3. Escolha Databases (Bancos de dados) no painel de navegação.
4. Escolha Criar banco de dados e, em seguida, escolha Criação padrão.
5. Em Tipo de mecanismo, escolha PostgreSQL.
6. Escolha uma versão do Engine de 15 ou superior.

> **Note**
>
> Salve a versão do mecanismo porque você precisará dela posteriormente neste tutorial.

7. Na seção Modelos, escolha Nível gratuito.
8. Altere o identificador da instância de banco de dados para algo significativo, por exemplo,`MicroFocus-Tutorial`.
9. Evite gerenciar credenciais mestras em. AWS Secrets Manager Em vez disso, insira uma senha mestra e confirme-a.

> **Note**
>
> Salve o nome de usuário e a senha que você usa para o banco de dados. Você os armazenará com segurança nas próximas etapas deste tutorial.

10. Em Conectividade, escolha a VPC em que você deseja criar o ambiente de tempo de execução gerenciado da modernização do AWS mainframe.
11. Selecione Criar banco de dados.

Para criar um grupo de parâmetros personalizado no Amazon RDS

1. No painel de navegação do console Amazon RDS, escolha Parameter groups e, em seguida, escolha Create parameter group.
2. Na janela Criar grupo de parâmetros, em Família de grupos de parâmetros, selecione a opção Postgres que corresponda à versão do seu banco de dados.

   > **Note**
   >
   > Algumas versões do Postgres exigem um Tipo. Selecione DB Parameter Group, se necessário. Insira um nome de grupo e uma descrição para o grupo de parâmetros.

3. Escolha Criar.

Para configurar o grupo de parâmetros personalizado

1. Escolha o grupo de parâmetros recém-criado.
2. Escolha Ações e, em seguida, escolha Editar.
3. Filtre `max_prepared_transactions` e altere o valor do parâmetro para 100.
4. Escolha Salvar alterações.

Para associar o grupo de parâmetros personalizados ao banco de dados

1. No painel de navegação do console Amazon RDS, escolha Bancos de dados e, em seguida, escolha a instância do banco de dados que você deseja modificar.
2. Selecione Modify. A página Modify DB instance (Modificar instância de banco de dados) será exibida.

   > **Note**
   >
   > A opção Modificar não está disponível até que o banco de dados termine de criar e fazer backup, o que pode levar alguns minutos.

3. Na página Modificar instância de banco de dados, navegue até Configuração adicional e altere o grupo de parâmetros do banco de dados para seu grupo de parâmetros. Se seu grupo de parâmetros não estiver disponível na lista, verifique se ele foi criado com a versão correta do banco de dados.

4. Escolha Continuar e verifique o resumo das modificações.
5. Escolha Aplicar imediatamente para aplicar as alterações instantaneamente.
6. Selecione Modificar instância de banco de dados para salvar as alterações.

Para obter mais informações sobre grupos de parâmetros, consulte Trabalho com grupos de parâmetros.

> **Note**
>
> Você também pode usar um banco de dados Amazon Aurora PostgreSQL com a modernização do AWS mainframe, mas não há opção de nível gratuito. Para obter mais informações, consulte Trabalhando com o Amazon Aurora PostgreSQL.

Etapa 3: criar e configurar um AWS KMS key

Para armazenar credenciais com segurança para a instância do Amazon RDS, primeiro crie um. AWS KMS key

Para criar um AWS KMS key

1. Abra o console do Key Management Service.
2. Escolha Create Key (Criar chave).
3. Deixe os padrões de Symmetric para o tipo de chave e Criptografar e descriptografar para o uso da chave.
4. Escolha Próximo.
5. Dê à chave um alias, como por exemplo, `MicroFocus-Tutorial-RDS-Key` e uma descrição opcional.
6. Escolha Próximo.
7. Atribua um administrador de chaves marcando a caixa ao lado do seu usuário ou função.
8. Escolha Avançar e, em seguida, escolha Avançar novamente.
9. Na tela de revisão, edite a política de chaves e insira o seguinte:

```
{
     "Sid" : "Allow access for Mainframe Modernization Service",
     "Effect" : "Allow",
```

```
        "Principal" : {
           "Service" : "m2.amazonaws.com"
                 },
      "Action" : "kms:Decrypt",
      "Resource" : "*"
},
```

Essa política concede permissões de decodificação da Modernização do AWS Mainframe usando essa política de chaves específica.

10. Escolha Concluir para criar a chave.

Para obter mais informações, consulte Criação de chaves no Guia do AWS Key Management Service desenvolvedor.

Etapa 4: criar e configurar um segredo de AWS Secrets Manager banco de dados

Agora, armazene as credenciais do banco de dados com segurança usando e. AWS Secrets Manager AWS KMS key

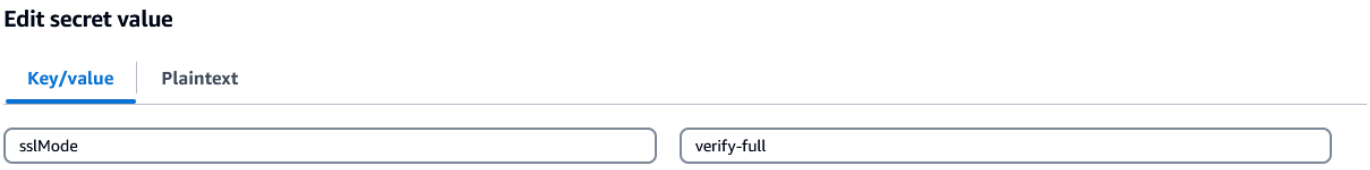
Para criar e configurar um segredo de AWS Secrets Manager banco de dados

1. Abra o console do Secrets Manager.
2. No painel de navegação, escolha Secrets (Segredos).
3. Em Segredos, escolha Armazenar um novo segredo.
4. Defina o tipo de segredo como Credenciais para o banco de dados Amazon RDS.
5. Insira as credenciais que você especificou ao criar o banco de dados.
6. Em Chave de criptografia, selecione a chave que você criou na etapa 3.
7. Na seção Banco de dados, selecione o banco de dados que você criou para este tutorial e escolha Avançar.
8. Em Nome secreto, insira um nome como `MicroFocus-Tutorial-RDS-Secret` e uma descrição opcional.
9. Na seção Permissões de recursos, escolha Editar permissões e substitua o conteúdo pela seguinte política:

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect" : "Allow",
            "Principal" : {
                "Service" : "m2.amazonaws.com"
            },
            "Action" : "secretsmanager:GetSecretValue",
            "Resource" : "*"
        }
    ]
}
```

10. Escolha Salvar.
11. Escolha Avançar para as telas subsequentes e, em seguida, escolha Armazenar. Atualize a lista de segredos para ver o novo segredo.
12. Escolha o segredo recém-criado e anote-o `Secret` ARN porque você precisará dele posteriormente no tutorial.
13. Na guia Visão geral do segredo, escolha Recuperar valor secreto.
14. Escolha Editar e, em seguida, escolha Adicionar linha.
15. Adicione uma chave para `sslMode` com um valor de`verify-full`:

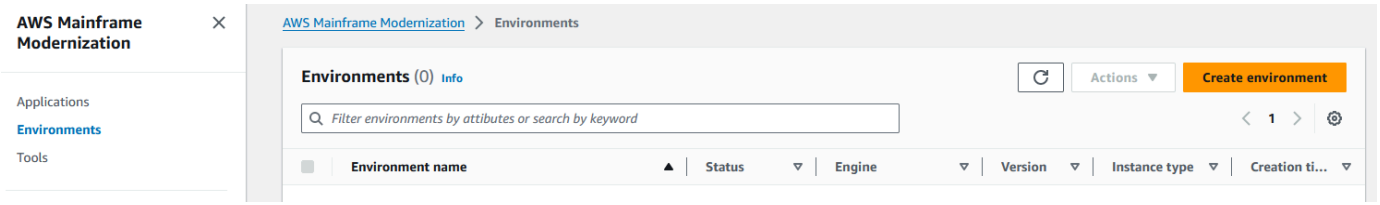


16. Escolha Salvar.

Etapa 5: criar um ambiente de tempo de execução

Para criar um ambiente de runtime

1. Abra o console do AWS Mainframe Modernization.
2. No painel de navegação, escolha Ambientes. Em seguida, escolha Criar ambiente.



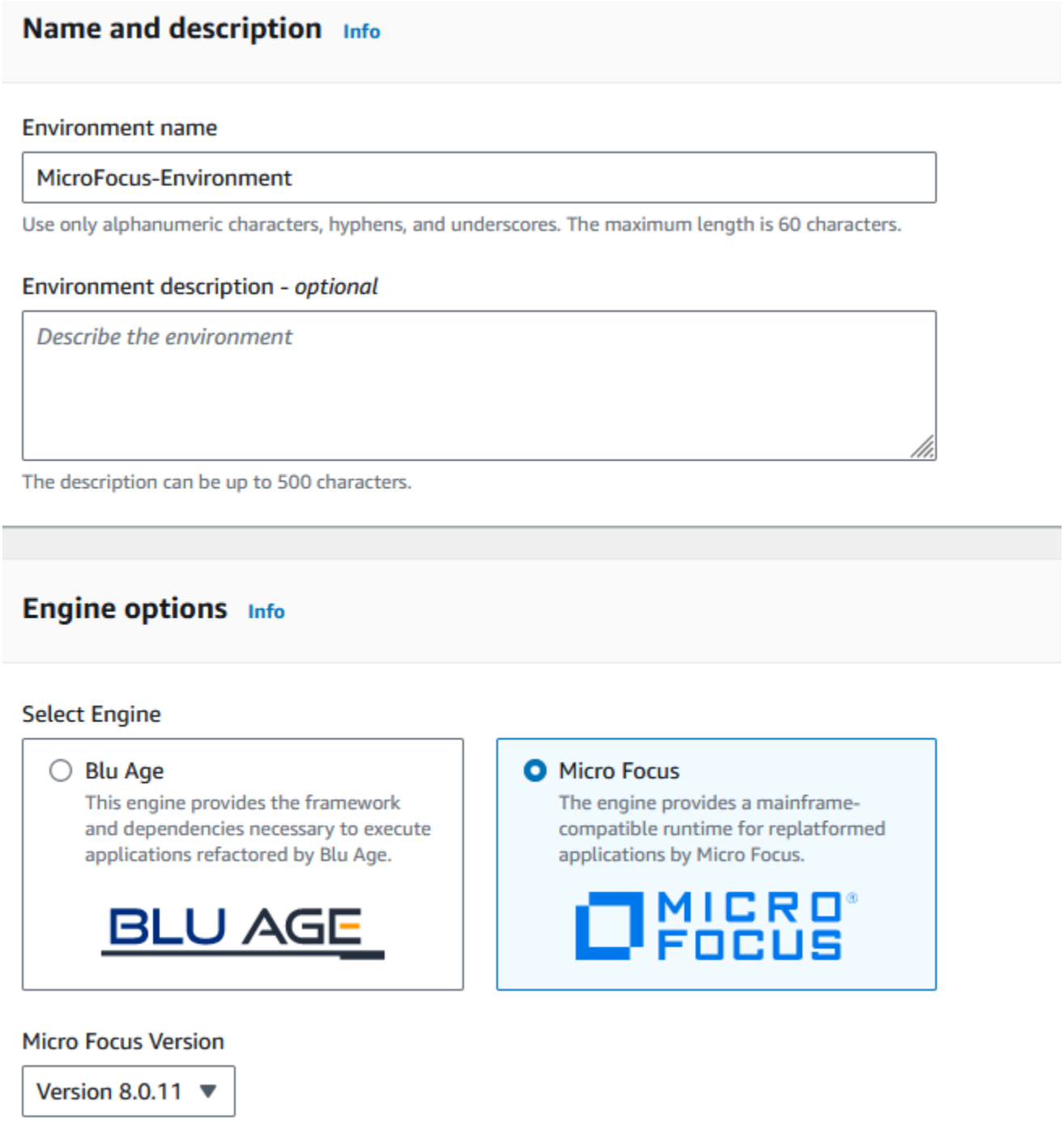
3. Em Especificar informações básicas,

    a. Insira `MicroFocus-Environment` o nome do ambiente.

    b. Nas opções do motor, verifique se a opção Micro Focus está selecionada.

    c. Escolha a versão mais recente do Micro Focus.

    d. Escolha Próximo.



4. Configure o ambiente.

   a. Em Disponibilidade, escolha Cluster de alta disponibilidade.

   b. Em Recursos, escolha um `M2.c5.large` ou `M2.m5.large` para o tipo de instância e o número de instâncias que você deseja. Especifique até duas instâncias.

c. Em Segurança e rede, escolha Permitir que aplicativos implantados nesse ambiente sejam acessíveis publicamente e escolha pelo menos duas sub-redes públicas.

d. Escolha Próximo.

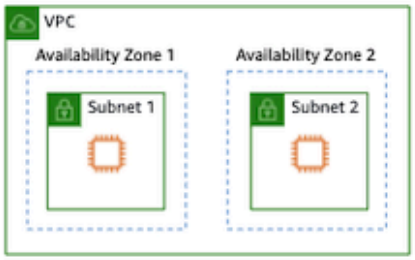
Specify configurations Info

Availability Info

Choose the availability pattern for your environment.

- Standalone runtime environment
  Sets up a single instance in a single availability zone. Does not guarantee high availability but costs less.

- High availability cluster
  Sets up redundant instances across two availability zones. Enables higher availability but costs more.



Resources

Instance type
Choose the instance type for your high availability cluster.

M2.m5.large

Desired capacity
Specify the desired number of instances.

2

Security and network

Allow applications deployed to this environment to be publicly accessible.

Virtual Private Cloud (VPC)
Choose the VPC where you want to create the environment.

Default vpc-15

Subnets
Choose one or more subnets for a high availability setup.

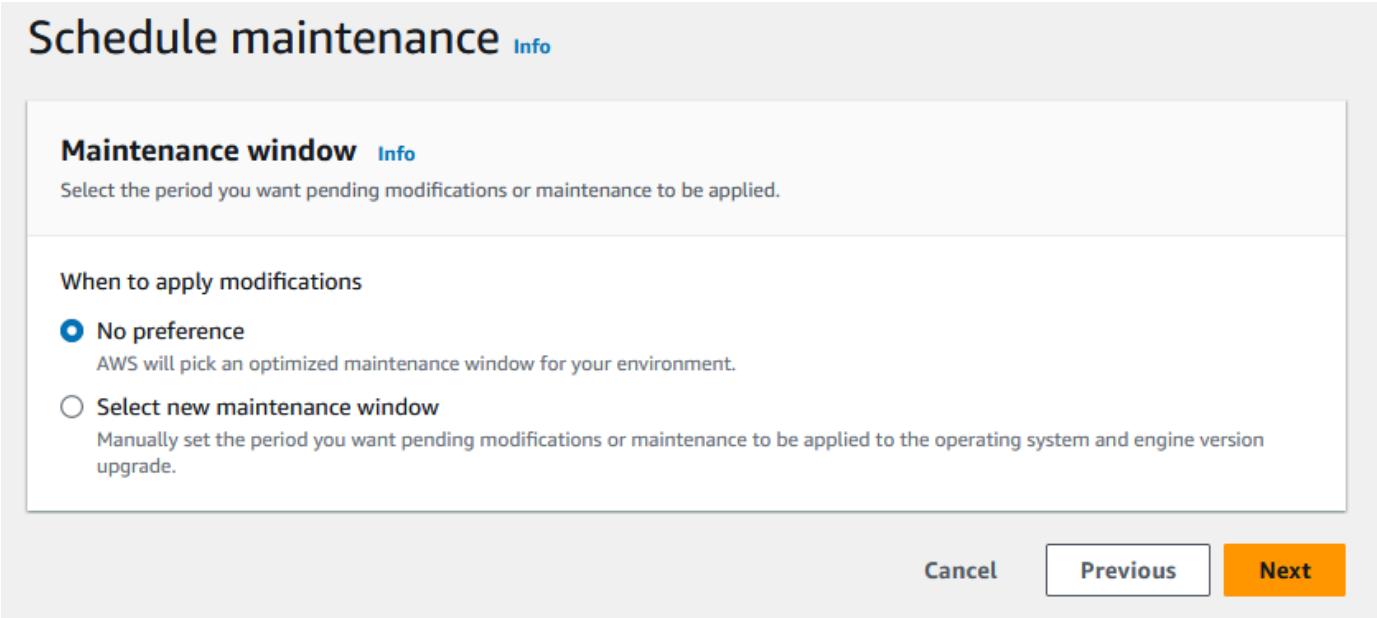
Choose subnets

subnet-56f1a | us-west-2a

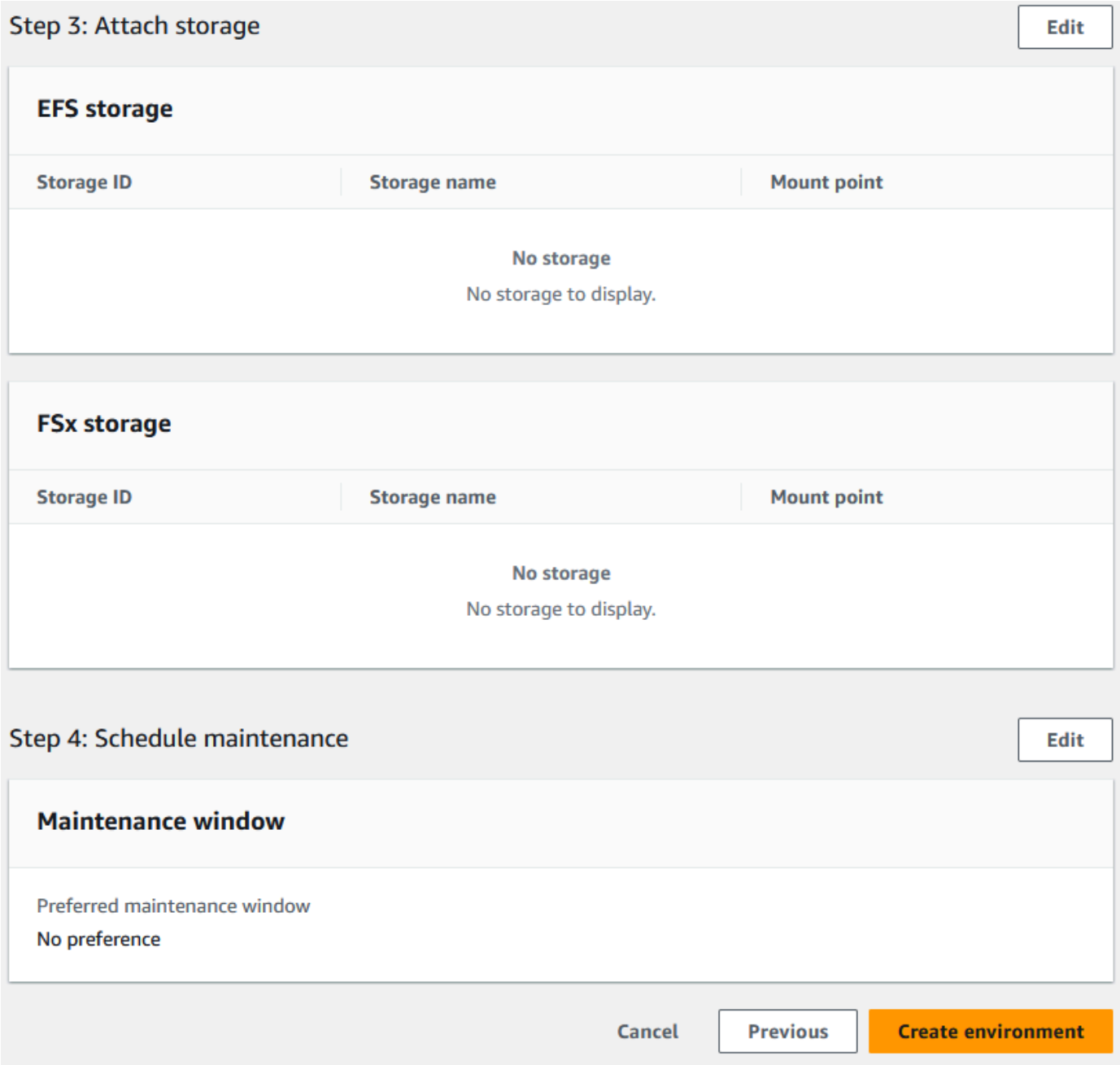
subnet-6685 | us-west-2b

Security groups
Choose one or more security groups for the chosen VPC.

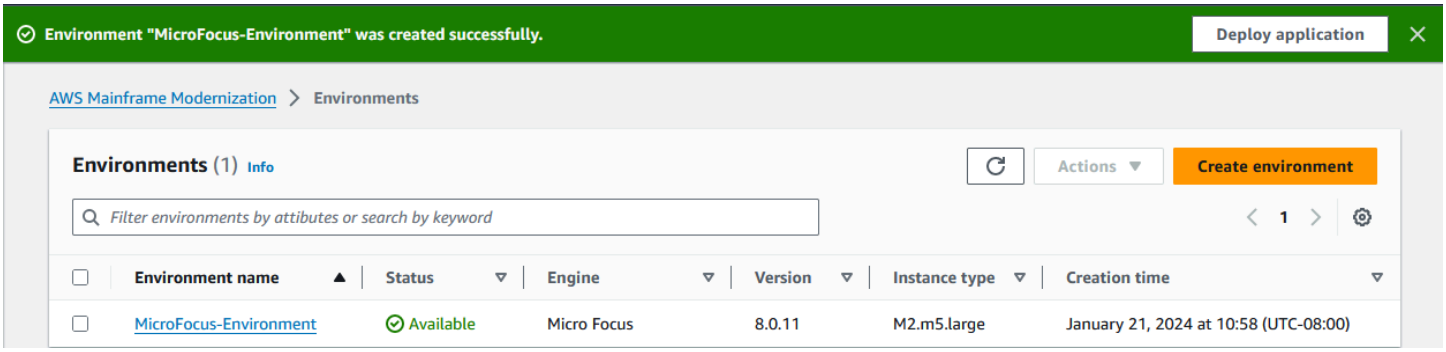
5. Na página Anexar armazenamento, escolha Próximo.
6. Na página Programar manutenção, escolha Sem preferência e, em seguida, escolha Avançar.



7. Na página Revisar e criar, revise todas as configurações fornecidas para o ambiente de tempo de execução e escolha Criar ambiente.



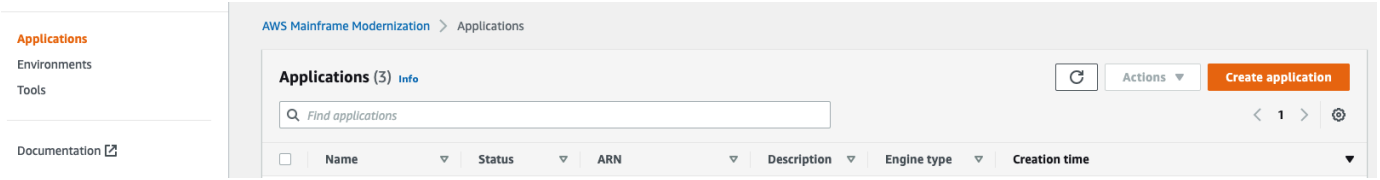
Quando você cria seu ambiente, aparece um banner que diz `Environment` *`name`* `was created successfully`, e o campo Status muda para Disponível. O processo de criação do ambiente leva alguns minutos, mas você pode continuar com as próximas etapas enquanto ele é executado.



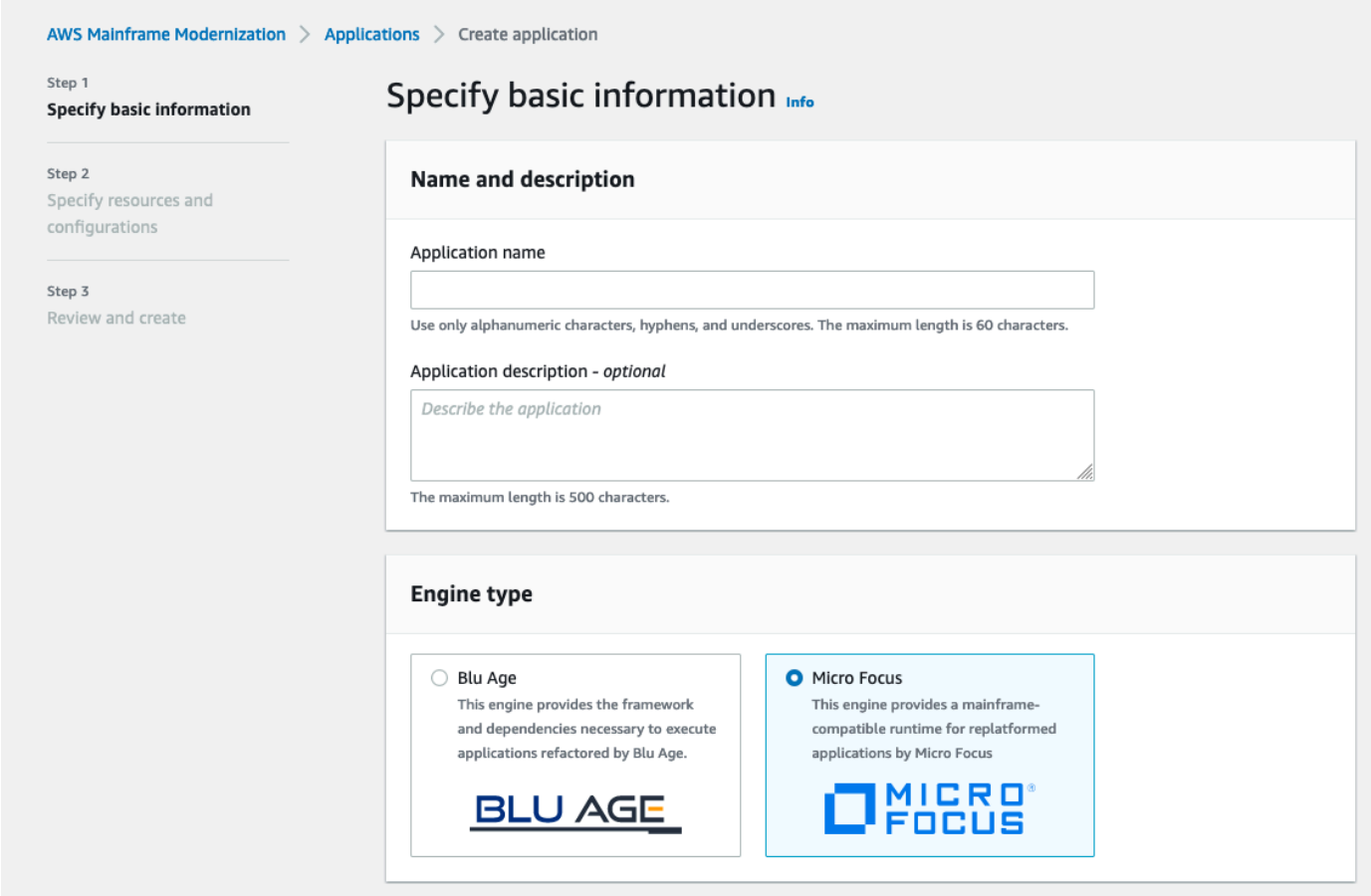
Etapa 6: criar um aplicativo

Para criar um aplicativo.

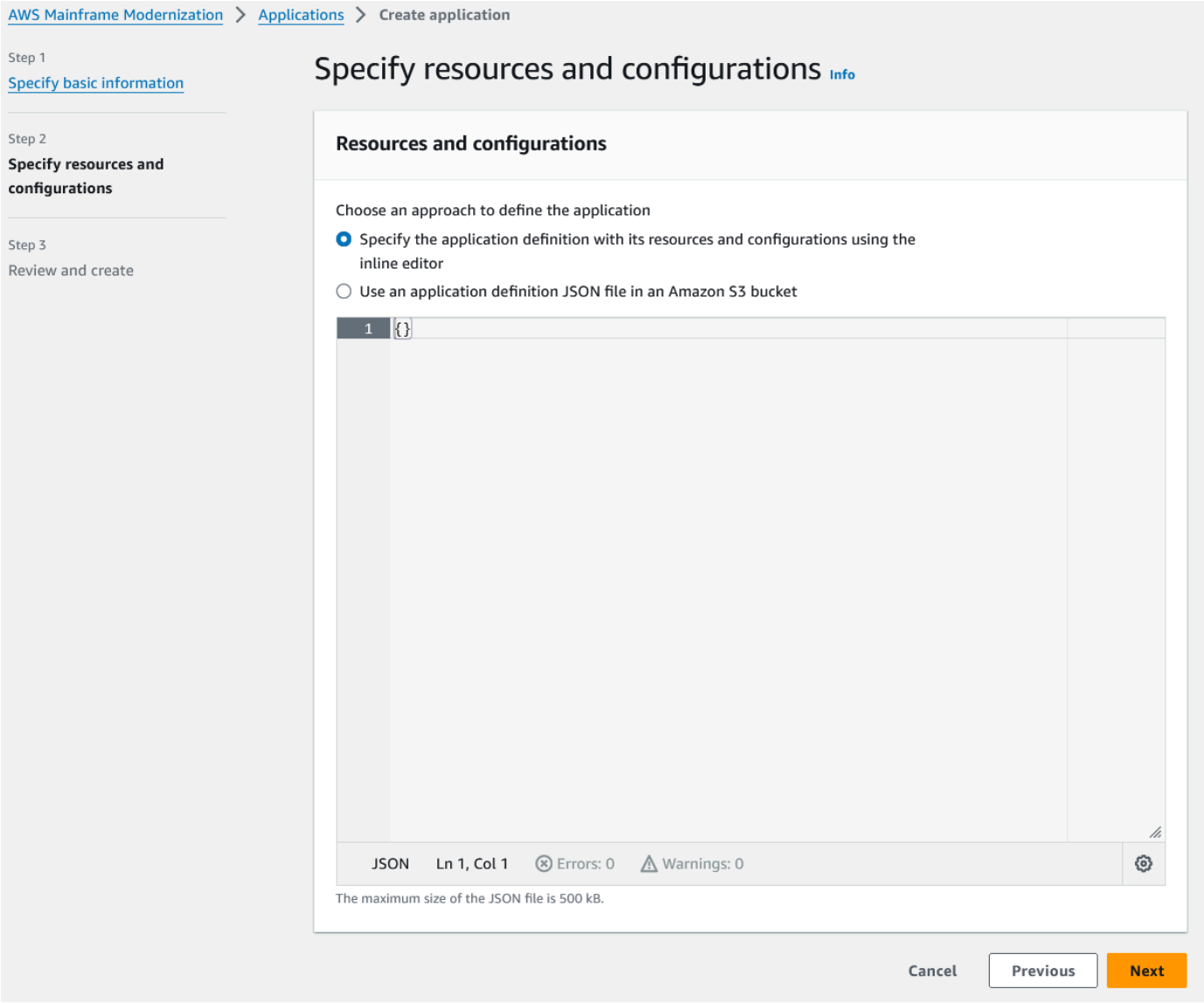
1. No painel de navegação, escolha Aplicativos. Escolha Criar aplicação.



2. Na página Criar aplicativo, em Especificar informações básicas, insira `MicroFocus-CardDemo` o nome do aplicativo e, em Tipo de mecanismo, verifique se a opção Micro Focus está selecionada. Em seguida, escolha Próximo.



3. Em Especificar recursos e configurações, escolha a opção para especificar a definição do aplicativo com seus recursos e configurações usando o editor embutido.



Insira a seguinte definição de aplicativo no editor:

```
{
  "template-version": "2.0",
  "source-locations": [
    {
      "source-id": "s3-source",
      "source-type": "s3",
      "properties": {
        "s3-bucket": "yourname-aws-region-carddemo",
        "s3-key-prefix": "CardDemo"
      }
    }
```

```
  ],
  "definition": {
    "listeners": [
      {
        "port": 6000,
        "type": "tn3270"
      }
    ],
    "dataset-location": {
      "db-locations": [
        {
          "name": "Database1",
          "secret-manager-arn":
 "arn:aws:secretsmanager:Region:123456789012:secret:MicroFocus-Tutorial-RDS-Secret-
xxxxxx"
        }
      ]
    },
    "batch-settings": {
      "initiators": [
        {
          "classes": [
               "A",
               "B"
            ],
          "description": "initiator_AB...."
        },
        {
          "classes": [
                "C",
                "D"
             ],
          "description": "initiator_CD...."
        }
      ],
      "jcl-file-location": "${s3-source}/catalog/jcl"
    },
    "cics-settings": {
      "binary-file-location": "${s3-source}/loadlib",
      "csd-file-location": "${s3-source}/rdef",
      "system-initialization-table": "CARDSIT"
    },
    "xa-resources": [
      {
```

```
        "name": "XASQL",
        "secret-manager-arn":
 "arn:aws:secretsmanager:Region:123456789012:secret:MicroFocus-Tutorial-RDS-Secret-
xxxxxx",
        "module": "${s3-source}/xa/ESPGSQLXA64.so"
      }
    ]
  }
}
```

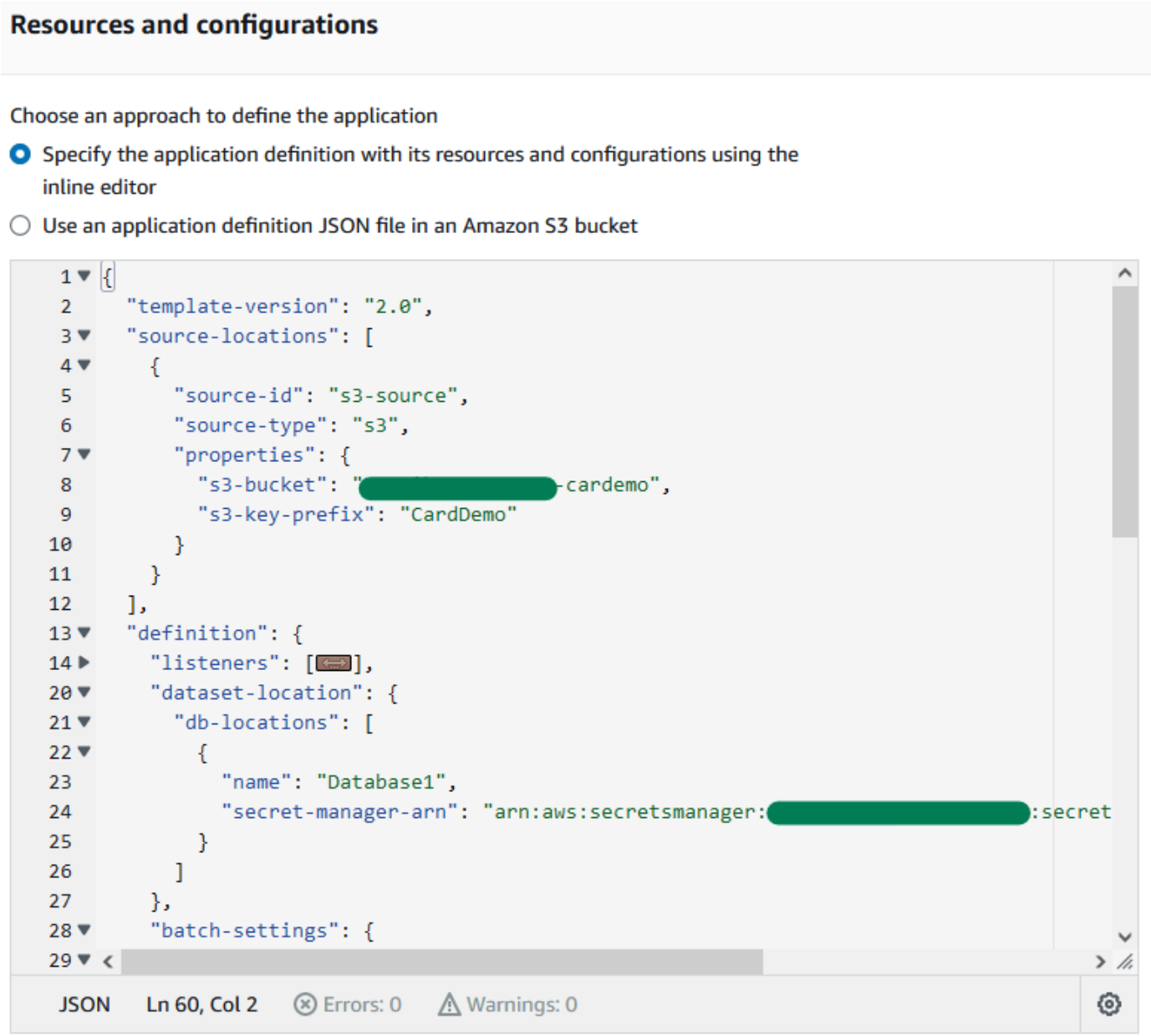
> **Note**
>
> Esse arquivo está sujeito a alterações.

4. Edite o aplicativo JSON no objeto de propriedades dos locais de origem da seguinte forma:

   a. Substitua o valor `s3_bucket` de pelo nome do bucket do Amazon S3 que você criou na Etapa 1.

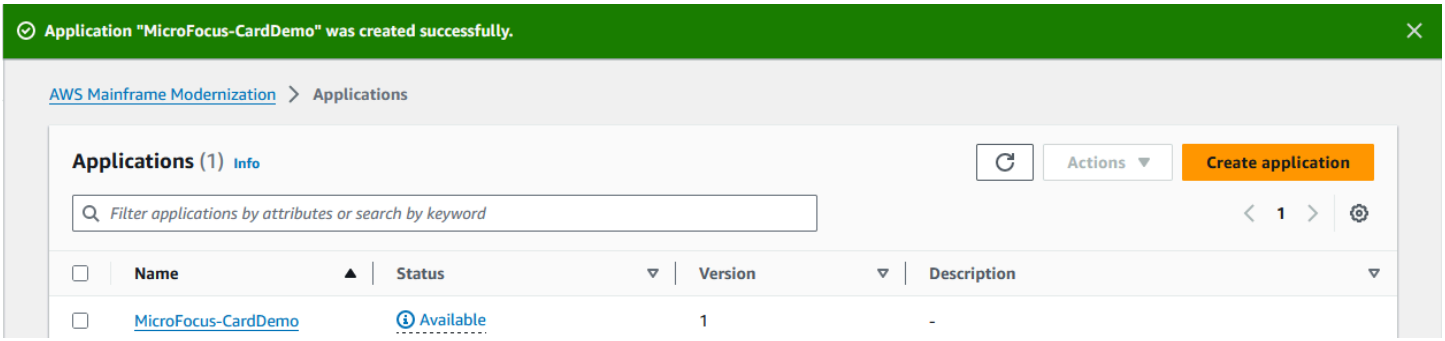
   b. Substitua o valor `s3-key-prefix` de pela pasta (prefixo de chave) na qual você fez o upload dos arquivos de CardDemo amostra. Se você fez o upload do `CardDemo` diretório diretamente para um bucket do Amazon S3, `s3-key-prefix` ele não precisa ser alterado.

   c. Substitua `secret-manager-arn` os dois valores pelo ARN do segredo do banco de dados que você criou na Etapa 4.



Para obter mais informações sobre a definição da aplicação, consulte Definição de aplicação da Micro Focus.

5. Escolha Próximo para continuar.
6. Na página Revisar e criar, revise as informações fornecidas e escolha Criar aplicativo.

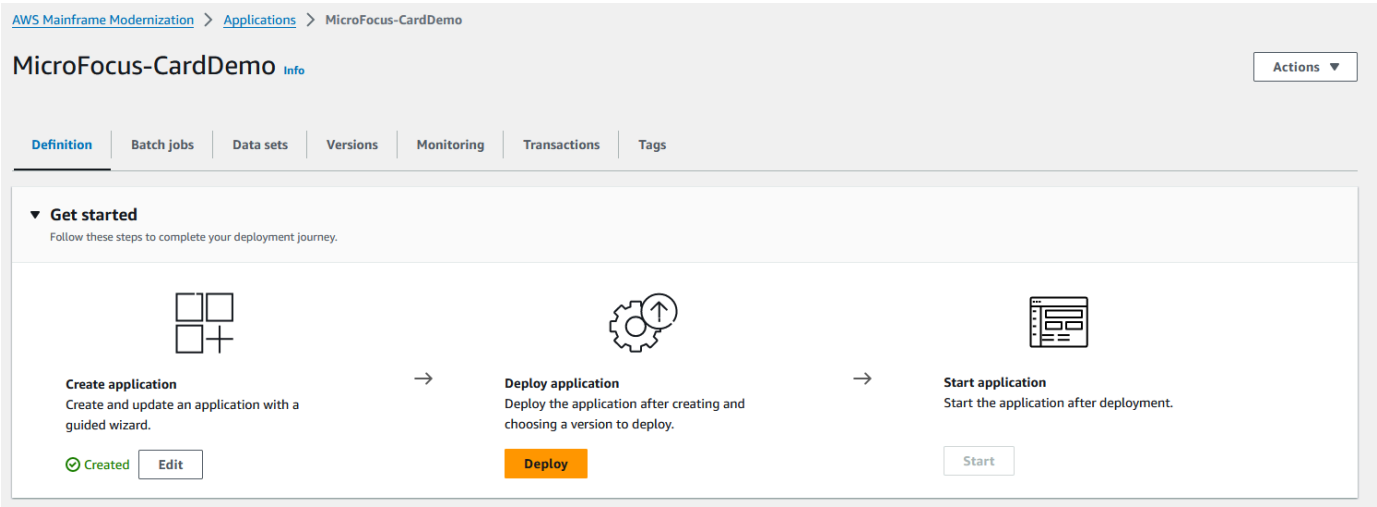


Quando você cria seu aplicativo, aparece um banner que diz`Application` *`name`* `was created successfully`. E o campo Status muda para Disponível.

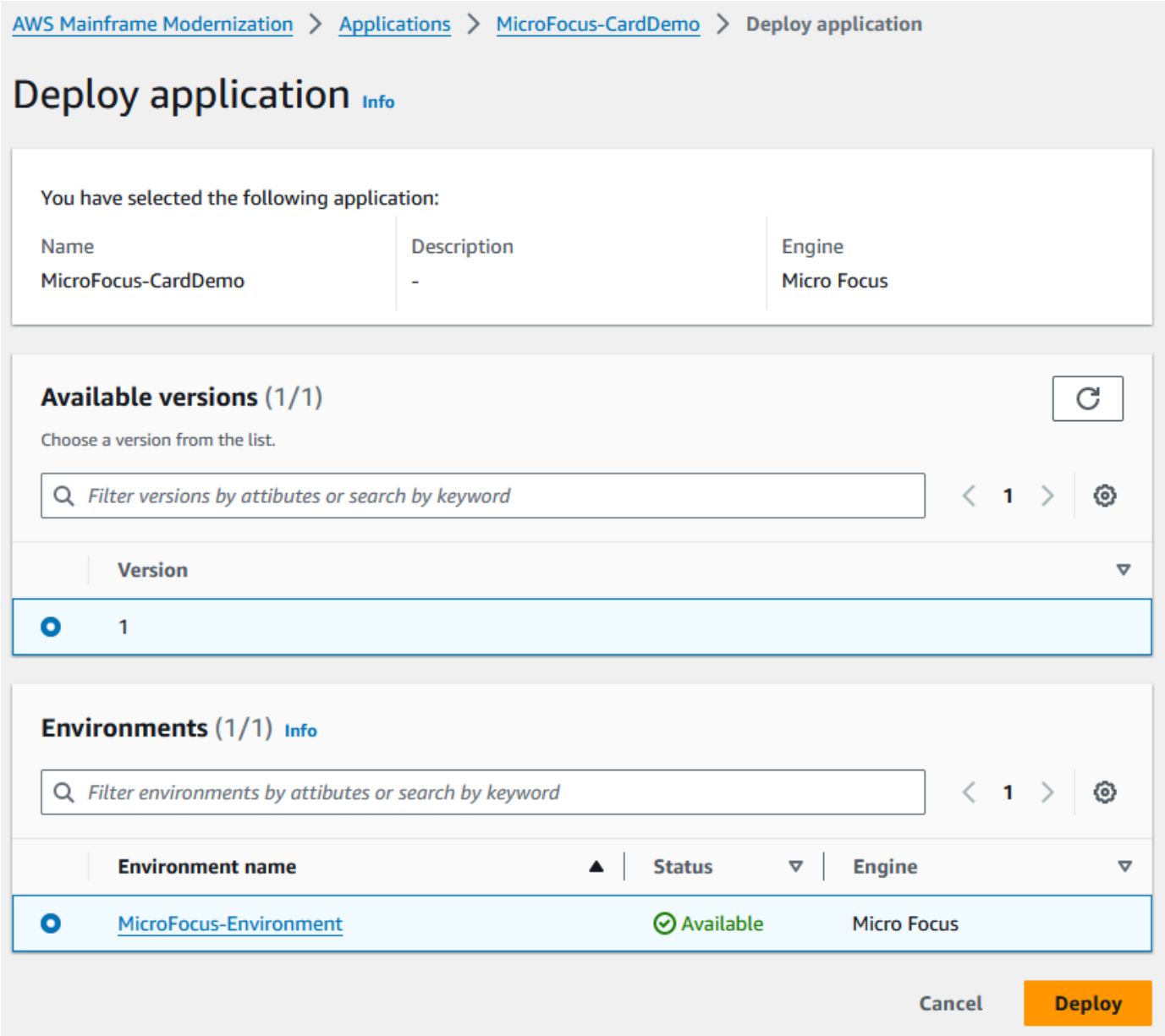
Etapa 7: implantar um aplicativo

Para implantar uma aplicação

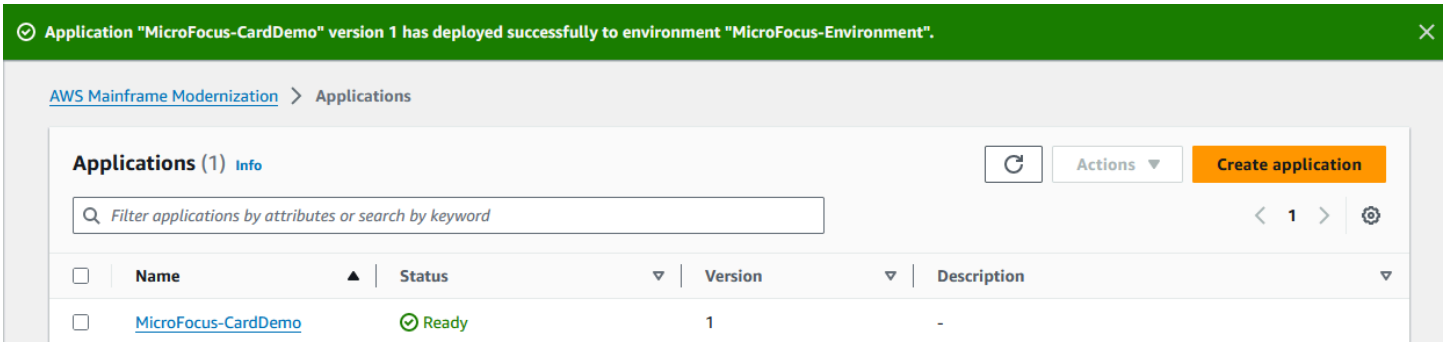
1. No painel de navegação, escolha Aplicativos e, em seguida, escolha`MicroFocus-CardDemo`.
2. Em Implantar aplicativo, escolha Implantar.



3. Escolha a versão mais recente do aplicativo e do ambiente que você criou anteriormente e, em seguida, escolha Implantar.



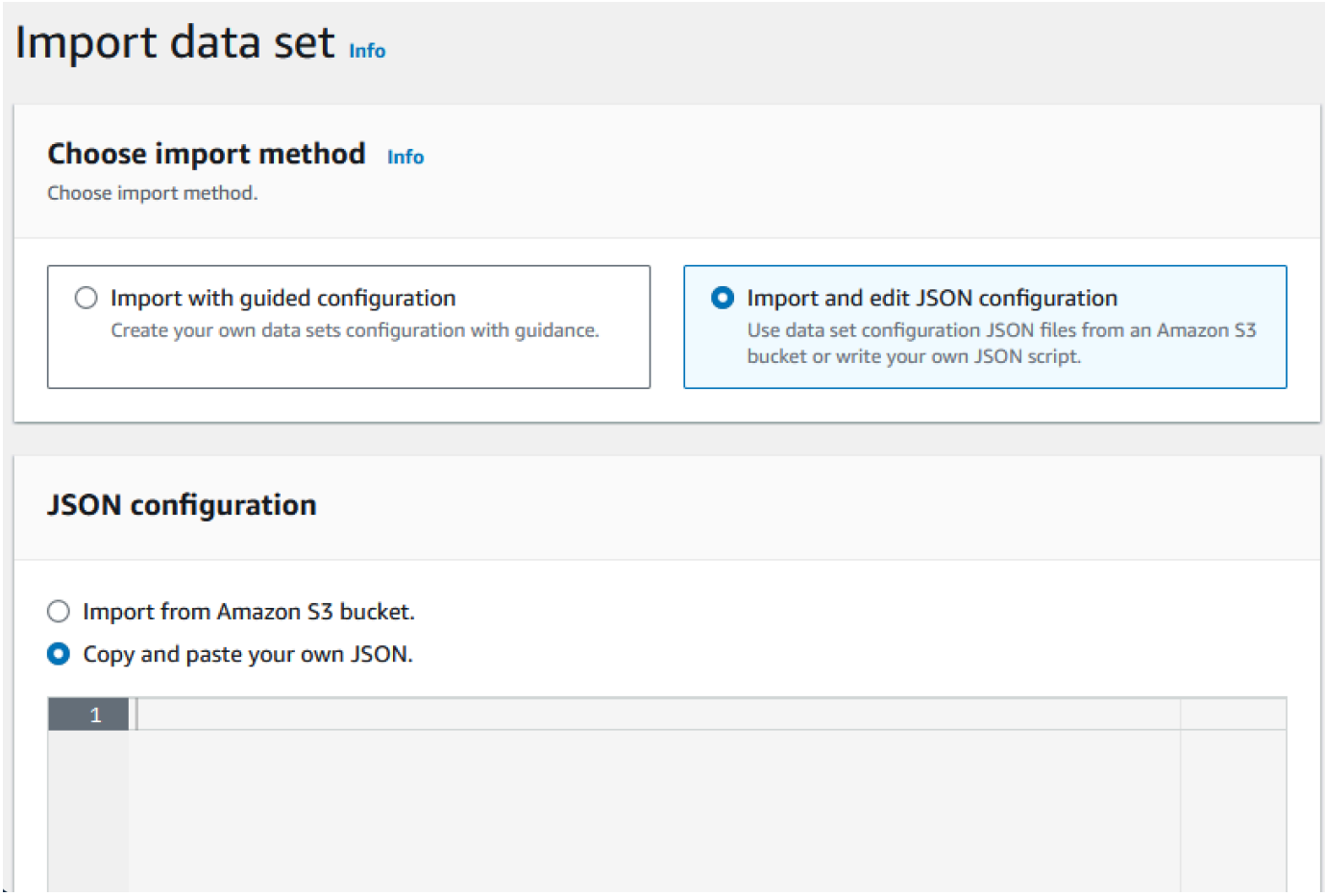
Quando o CardDemo aplicativo é implantado com êxito, o status muda para Pronto.



Etapa 8: importar conjuntos de dados

Para importar conjuntos de dados

1. No painel de navegação, escolha Aplicativos e, em seguida, escolha o aplicativo.
2. Escolha a guia Conjuntos de dados. Selecione Import (Importar).
3. Escolha Importar e editar configuração JSON e, em seguida, escolha a opção Copiar e colar seu próprio JSON.



4. Copie e cole o seguinte JSON, mas ainda não escolha “Enviar”. Esse JSON contém todos os conjuntos de dados necessários para o aplicativo de demonstração, mas precisa dos detalhes do bucket do Amazon S3.

```
{
    "dataSets": [
        {
            "dataSet": {
                "storageType": "Database",
```

```
                "datasetName": "AWS.M2.CARDDEMO.ACCTDATA.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 11,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 300,
                    "max": 300
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.ACCTDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDDATA.VSAM.AIX.PATH",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 11,
                            "offset": 16
                        }
                    }
                },
                "recordLength": {
                    "min": 150,
                    "max": 150
                }
            },
            "externalLocation": {
```

```
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.CARDDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDDATA.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 16,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 150,
                    "max": 150
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.CARDDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDXREF.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 16,
                            "offset": 0
                        }
                    }
                },
```

```
                "recordLength": {
                    "min": 50,
                    "max": 50
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.CARDXREF.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CUSTDATA.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 9,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 500,
                    "max": 500
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.CUSTDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDXREF.VSAM.AIX.PATH",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
```

```
                        "primaryKey": {
                            "length": 11,
                            "offset": 25
                        }
                    }
                },
                "recordLength": {
                    "min": 50,
                    "max": 50
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.CARDXREF.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 16,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 350,
                    "max": 350
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
```

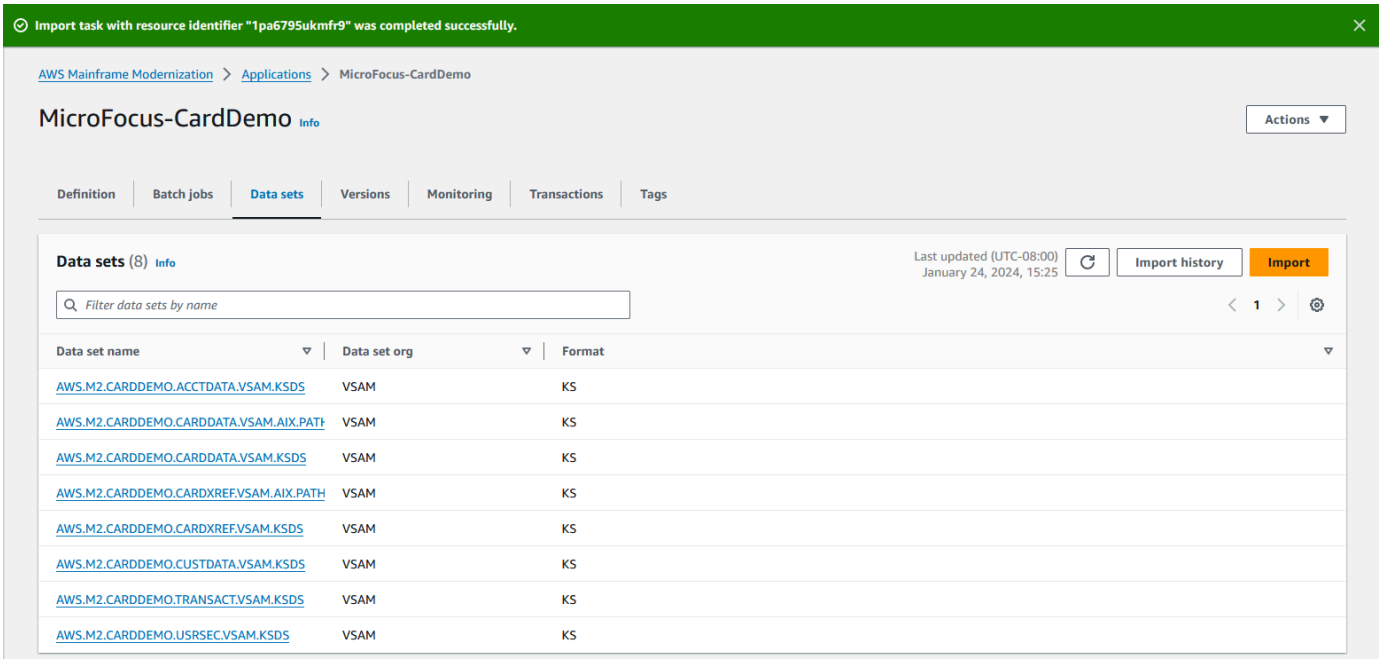
```
                "datasetName": "AWS.M2.CARDDEMO.USRSEC.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 8,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 80,
                    "max": 80
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo/catalog/data/
AWS.M2.CARDDEMO.USRSEC.VSAM.KSDS.DAT"
            }
        }
    ]
}
```

5. Substitua cada ocorrência de `<s3-bucket-name>` (há oito) pelo nome do bucket do Amazon S3 que contém a CardDemo pasta, por exemplo,. `your-name-aws-region-carddemo`

> **Note**
>
> Para copiar o URI do Amazon S3 para a pasta no Amazon S3, selecione a pasta e escolha Copiar URI do Amazon S3.

6. Selecione Enviar.

   Quando a importação é concluída, um banner aparece com a seguinte mensagem: `Import task with resource identifier` *`name`* `was completed successfully.` Uma lista dos conjuntos de dados importados é exibida.

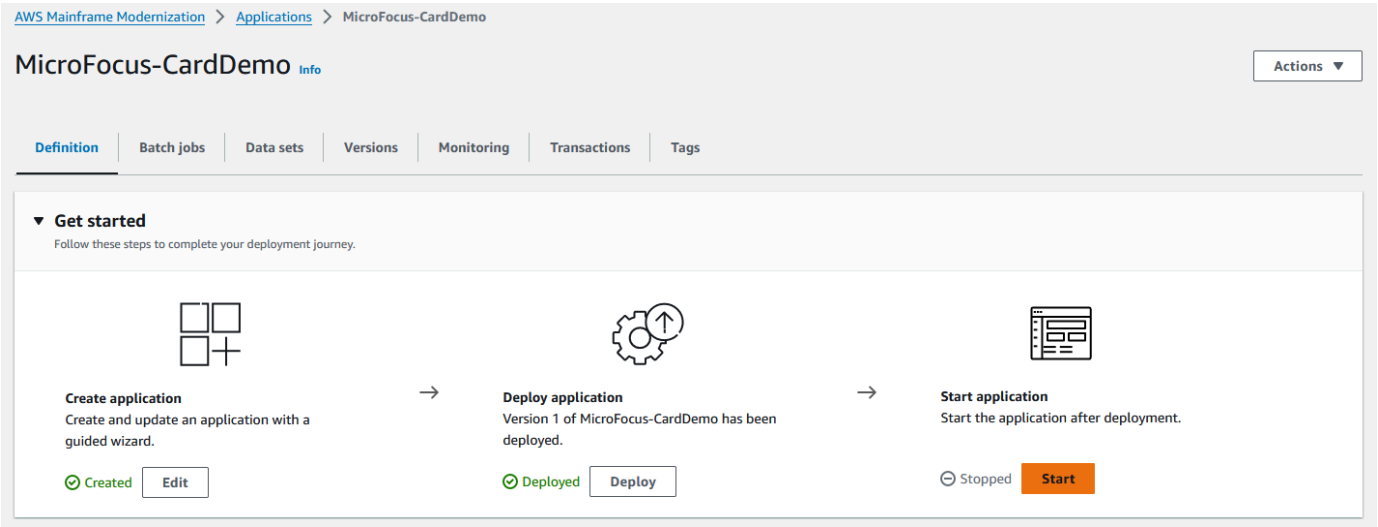


Você também pode visualizar o status de todas as importações de conjuntos de dados escolhendo Histórico de importação na guia Conjuntos de dados.

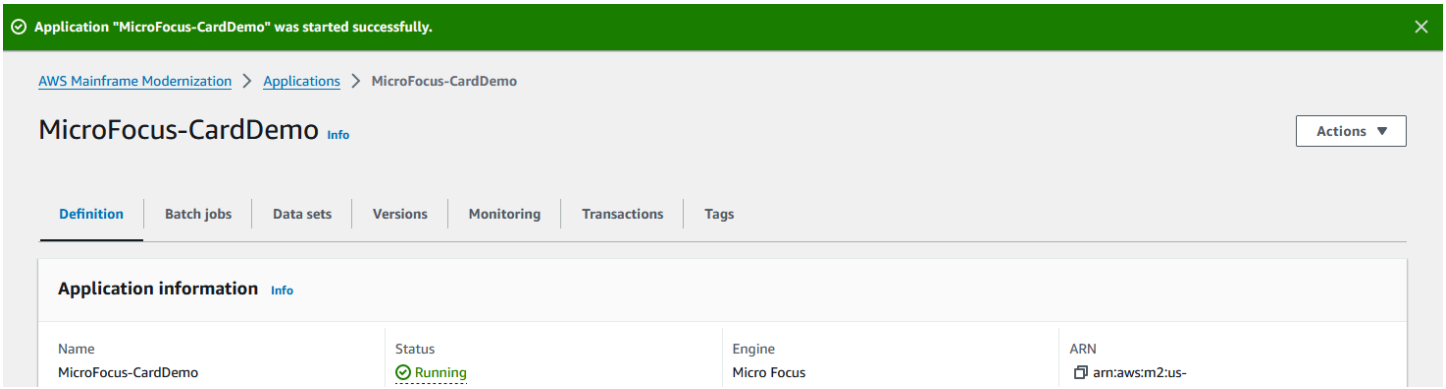
Etapa 9: iniciar um aplicativo

Para iniciar um aplicativo

1. No painel de navegação, escolha Aplicativos e, em seguida, escolha o aplicativo.
2. Escolha Iniciar aplicativo.



Quando o CardDemo aplicativo começa a ser executado com sucesso, um banner aparece com a seguinte mensagem:`Application` *`name`* `was started successfully`. O campo Status muda para Em execução.



Etapa 10: Conecte-se ao aplicativo CardDemo CICS

Antes de se conectar, certifique-se de que a VPC e o grupo de segurança que você especificou para o aplicativo sejam os mesmos que você aplicou para a interface de rede a partir da qual você se conectará.

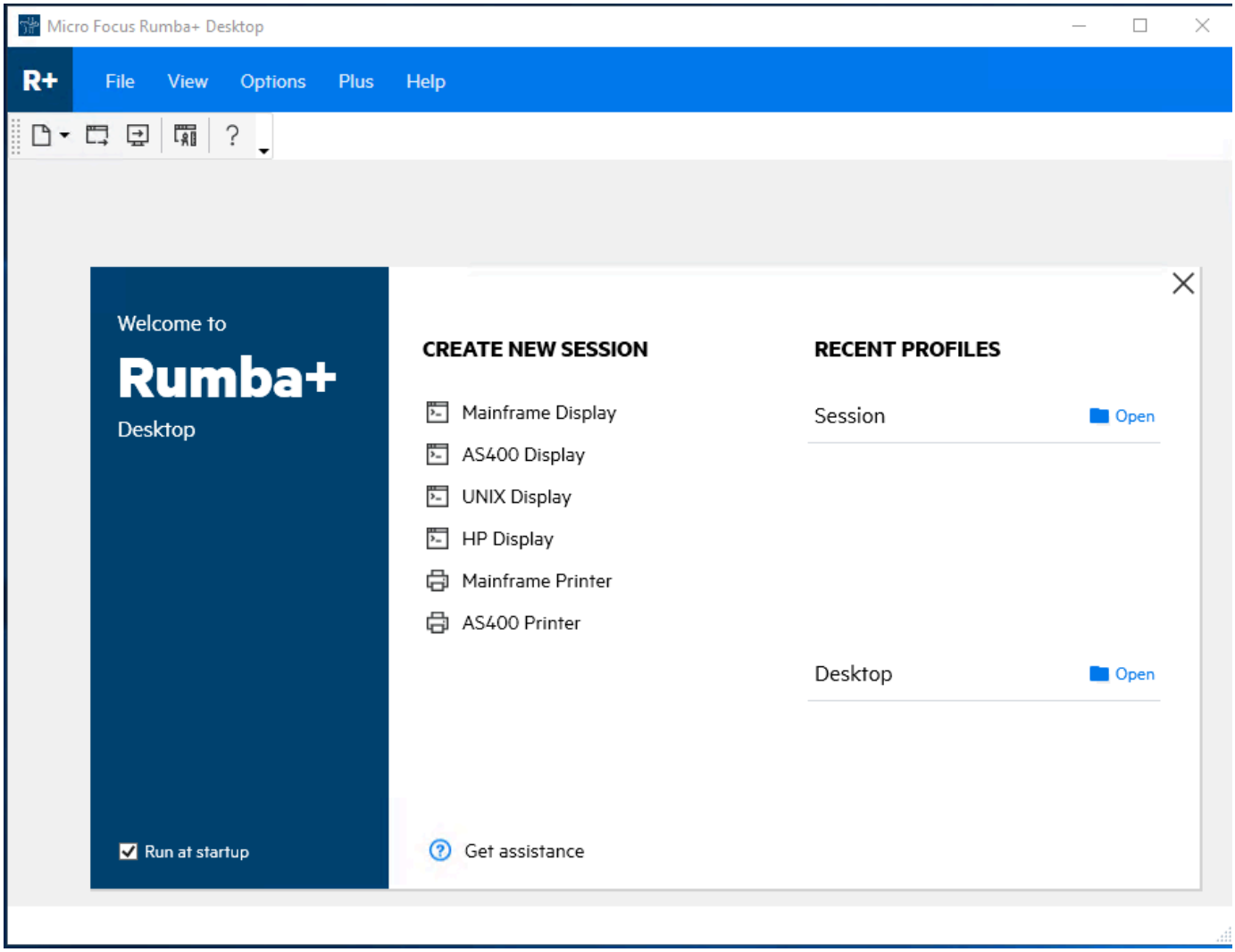
Para configurar a conexão TN3270, você também precisa do nome do host DNS e da porta do aplicativo.

Para configurar e conectar um aplicativo ao mainframe usando o emulador de terminal

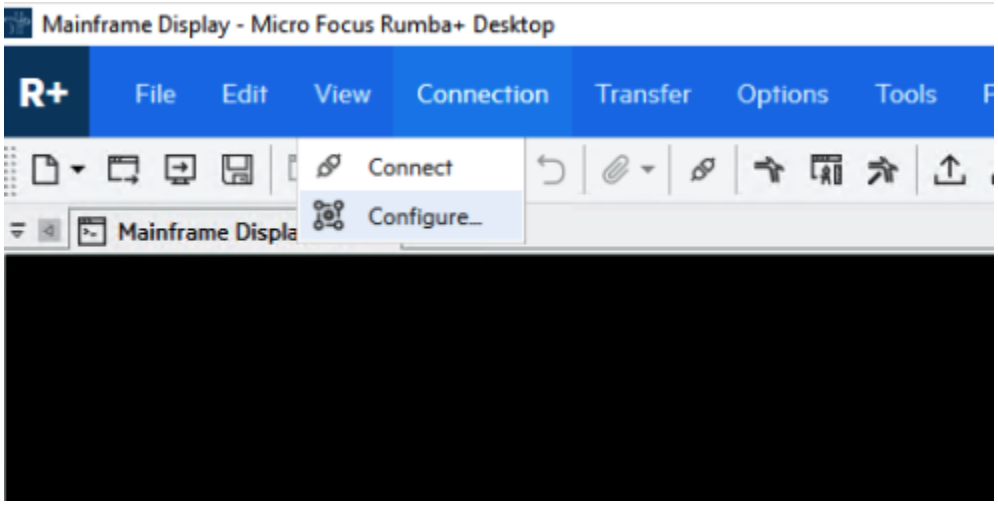
1. Abra o console de modernização do AWS mainframe, escolha Aplicativos e, em seguida, escolha. `MicroFocus-CardDemo`
2. Escolha o ícone de cópia para copiar o nome do host DNS. Além disso, certifique-se de anotar o número das portas.
3. Inicie um emulador de terminal. Este tutorial usa o Micro Focus Rumba+.

   > **Note**
   >
   > As etapas de configuração variam de acordo com o emulador.

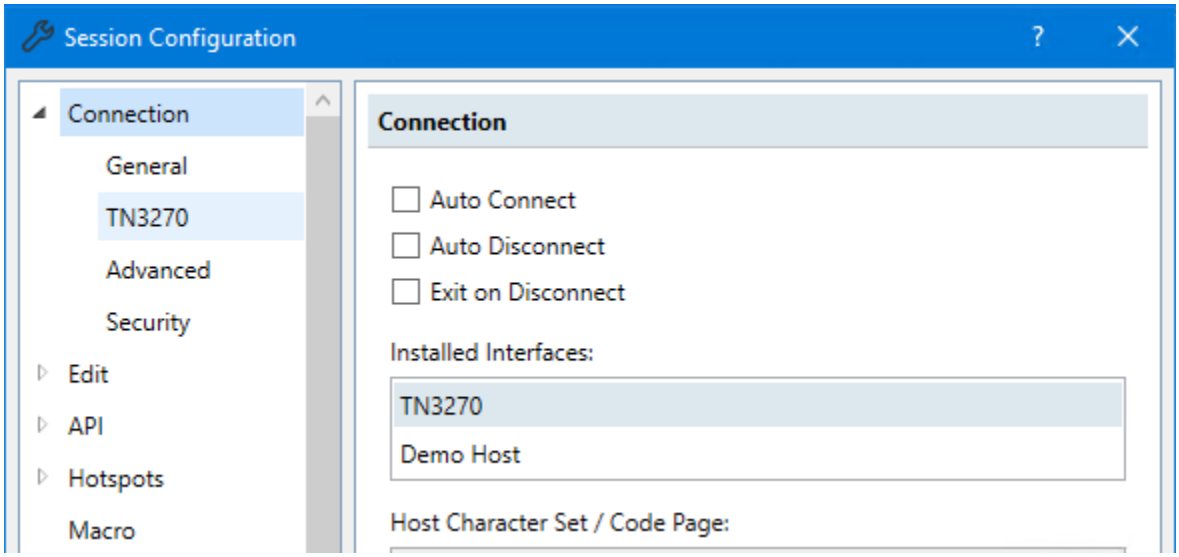
4. Escolha Mainframe Display.



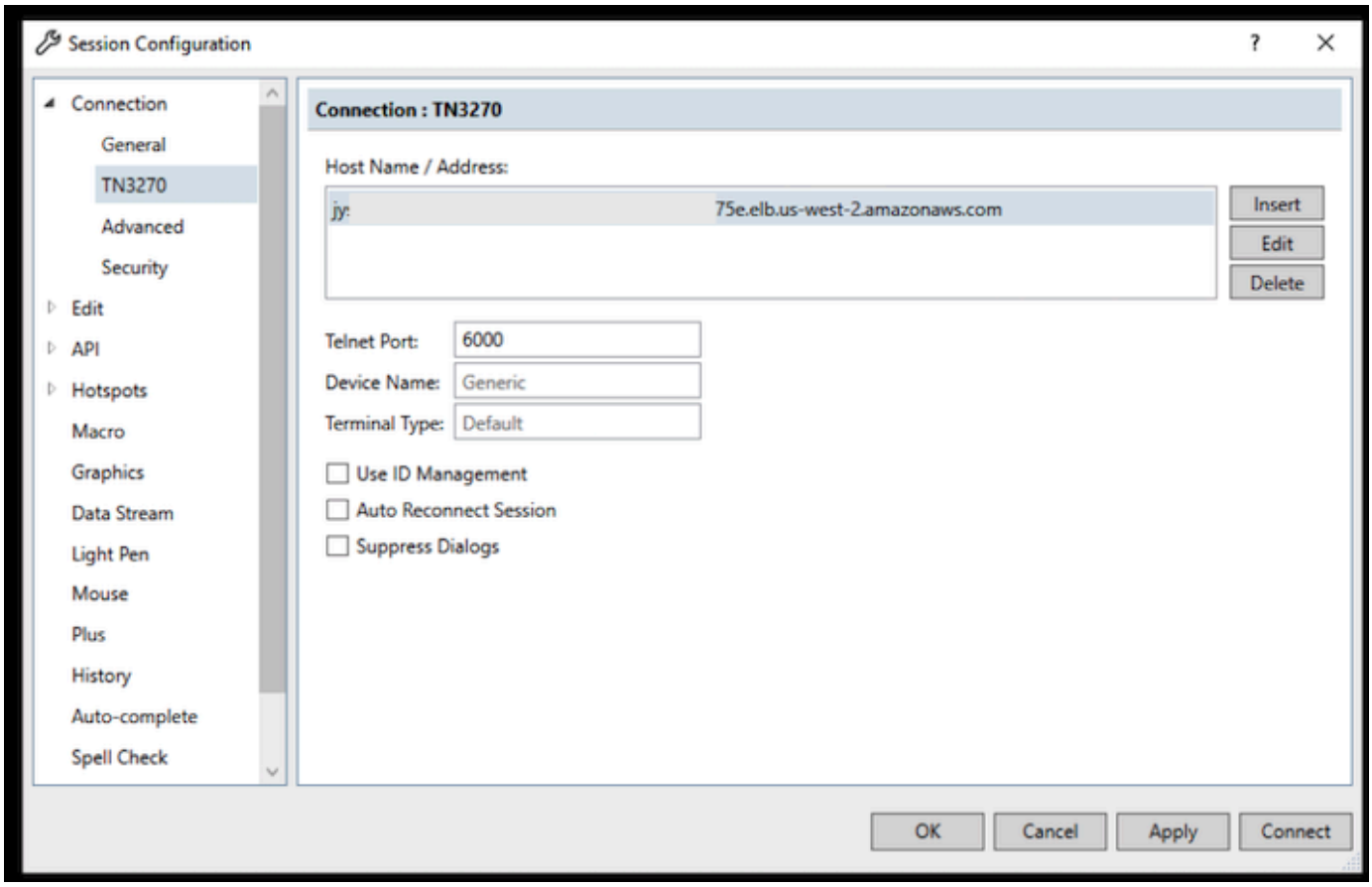
5. Escolha Conexão e, em seguida, escolha Configurar.



6. Em Interfaces instaladas, escolha eTN3270, em seguida, escolha TN3270 novamente no menu Conexão.



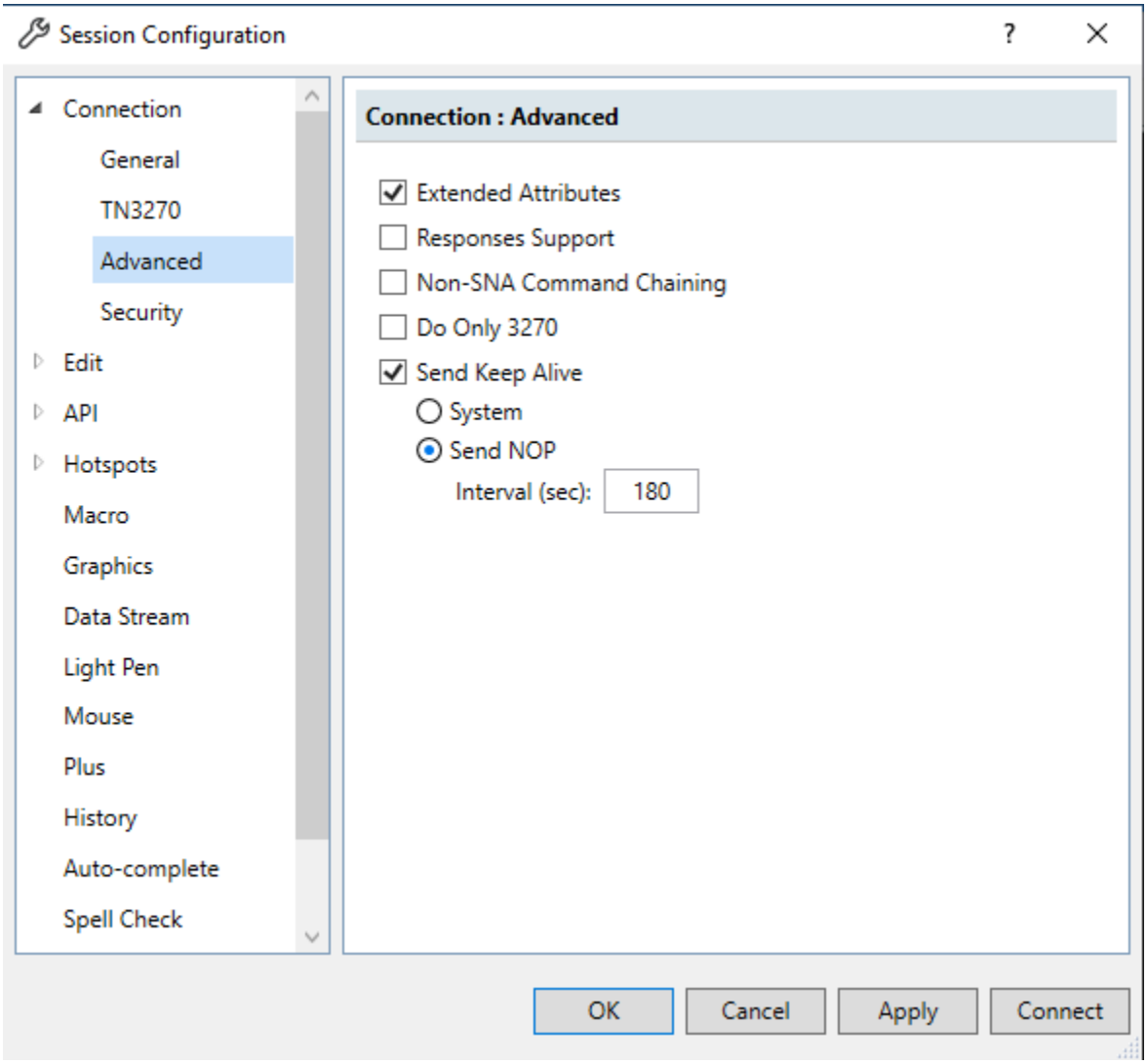
7. Escolha Inserir e cole o DNS `Hostname` para o aplicativo. Especifique `6000` para a porta Telnet.



> **Note**
>
> Se você estiver usando o AWS AppStream 2.0 em um navegador e tiver dificuldades em colar valores, consulte Solução de problemas do usuário AppStream 2.0.

8. Em Conexão, escolha Avançado e, em seguida, escolha Send Keep Alive e Send NOP e insira 180 para o Intervalo.

> **Note**
>
> Configurar a configuração keep alive em seu terminal TN3270 para pelo menos 180 segundos ajuda a garantir que o Network Load Balancer não interrompa sua conexão.



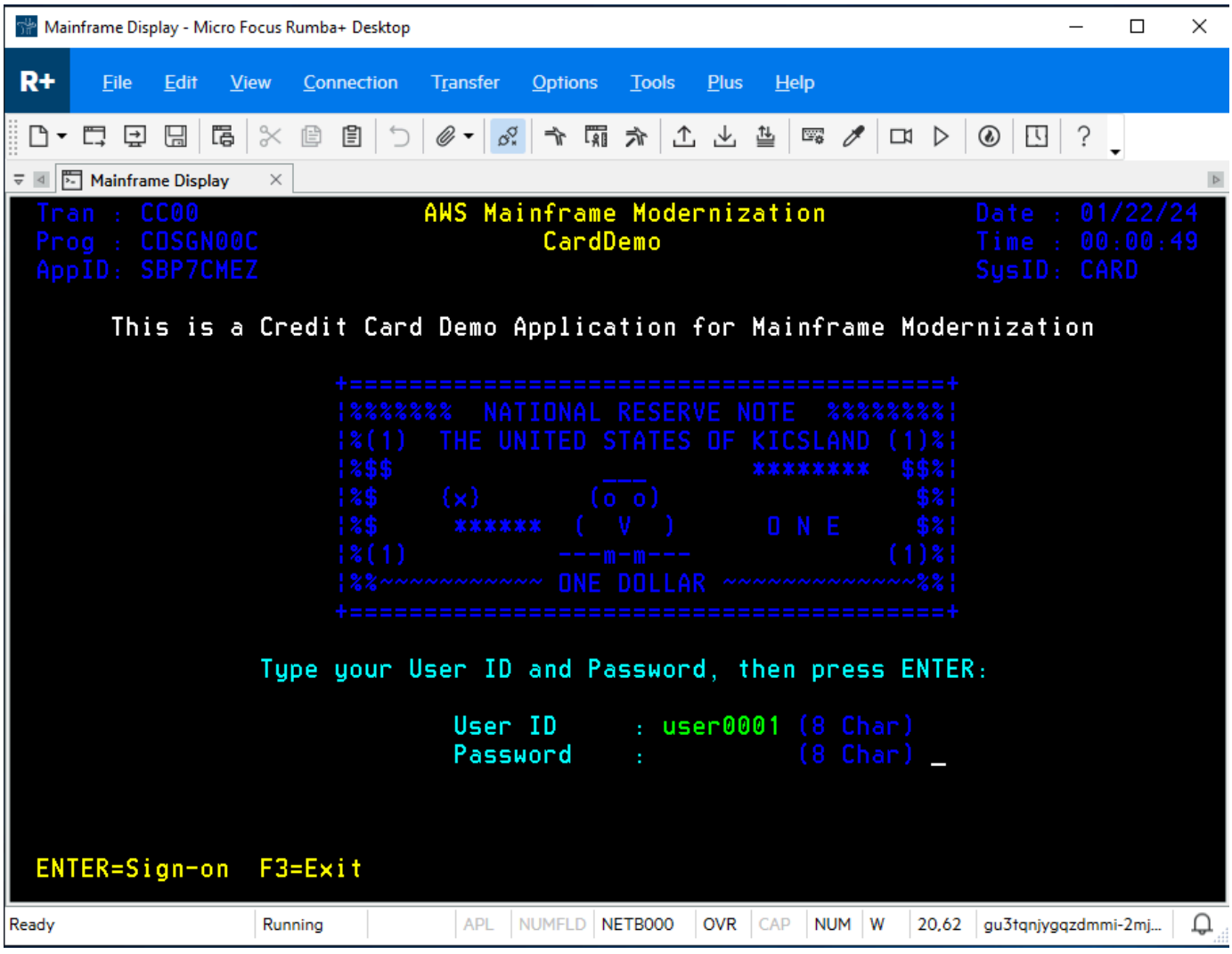
9. Selecione Conectar.

> **Note**
>
> Se a conexão falhar:
>
> - Se você estiver usando AppStream 2.0, confirme se a VPC e o grupo de segurança especificados para o ambiente do aplicativo são os mesmos da frota 2.0. AppStream
> - Use o VPC Reachability Analyzer para analisar a conexão. Você pode acessar o Reachability Analyzer por meio do console.
> - Como etapa de diagnóstico, tente adicionar ou alterar as regras de entrada do Grupo de Segurança do aplicativo para permitir o tráfego para a porta 6000 de qualquer

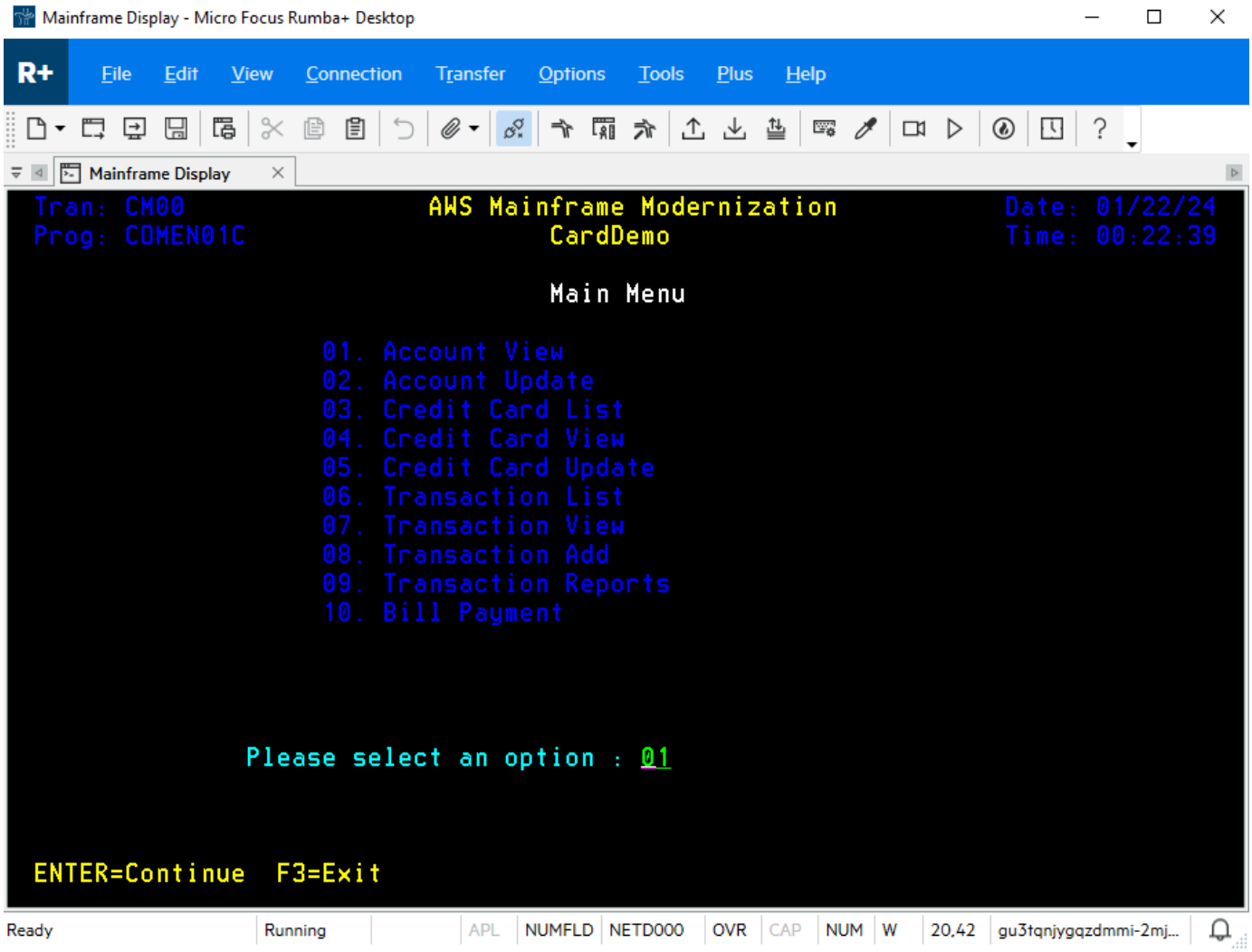
lugar (ou seja, Bloco CIDR 0.0.0.0/0). Se você se conectar com sucesso, saberá que o grupo de segurança estava bloqueando seu tráfego. Altere a fonte do grupo de segurança para algo mais específico. Para obter mais informações sobre grupos de segurança, consulte Noções básicas sobre grupos de segurança.

10. Digite `USER0001` o nome de usuário e `password` a senha.

> **Note**
>
> No Rumba, o padrão para Limpar é ctrl-shift-z, e o padrão para Redefinir é ctrl-r.

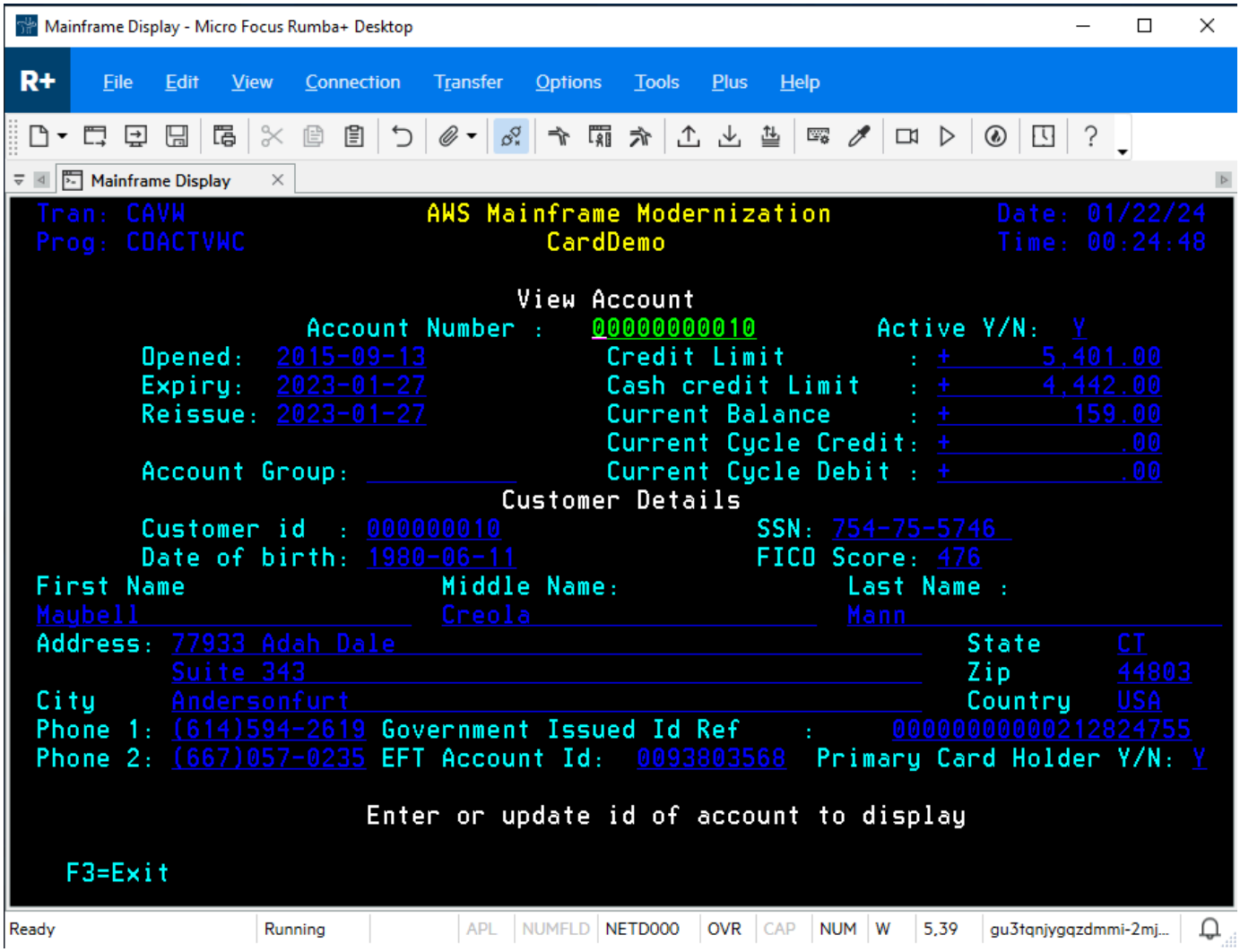


11. Depois de fazer login com sucesso, você pode navegar pelo CardDemo aplicativo.
12. Entre `01` para visualizar a conta.



13. Digite `00000000010` o número da conta e pressione Enter no teclado.

> **Note**
>
> Outras contas válidas são `0000000011` `00000000020` e.



14. Pressione F3 para sair do menu e F3 para sair da transação.

Limpar os recursos

Se você não precisar mais dos recursos que criou para este tutorial, exclua-os para evitar cobranças adicionais. Para fazer isso, realize as etapas a seguir:

- Se necessário, interrompa o aplicativo.
- Exclua o aplicativo do . Para ter mais informações, consulte Excluir um aplicativo de modernização AWS de mainframe.
- Exclua o ambiente de runtime. Para ter mais informações, consulte Excluir um ambiente de AWS execução de modernização de mainframe.

- Exclua os buckets do Amazon S3 que você criou para este tutorial. Para obter mais informações, consulte Exclusão de um bucket no Guia do usuário do Amazon S3.
- Exclua o AWS Secrets Manager segredo que você criou para este tutorial. Para obter mais informações, consulte Excluir um segredo.
- Exclua a chave KMS que você criou para este tutorial. Para obter mais informações, consulte Excluir chaves do AWS KMS.
- Exclua o banco de dados do Amazon RDS que você criou para este tutorial. Para obter mais informações, consulte Excluir a instância do EC2 e a instância de banco de dados no Guia do usuário do Amazon RDS.
- Se você adicionou uma regra de grupo de segurança para a porta 6000, exclua a regra.

Próximas etapas

Para saber como configurar um ambiente de desenvolvimento para seus aplicativos modernizados, consulte o Tutorial: Configurar AppStream 2.0 para uso com o Micro Focus Enterprise Analyzer e o Micro Focus Enterprise Developer.

Abordagem de modernização

A migração é complexa e tem muitas variáveis. AWS A modernização do mainframe oferece uma abordagem evolutiva que proporciona alguns ganhos de curto prazo ao melhorar a agilidade com muitas oportunidades de otimizar e inovar posteriormente. Além disso, a modernização do AWS mainframe ajuda a simplificar a jornada e ainda respeita as particularidades da empresa e dos negócios de seu cliente. As duas principais abordagens suportadas pela modernização do AWS mainframe são a refatoração automatizada ou a reformulação de plataformas. A escolha depende da situação do seu cliente.

A refatoração automatizada usa ferramentas AWS Blu Age para converter automaticamente código, dados e dependências em linguagem, armazenamento de dados e estruturas modernas, ao mesmo tempo em que garante a equivalência funcional com as mesmas funções de negócios.

A redefinição de plataforma usa as ferramentas da Micro Focus para transformar workloads de mainframe em serviços ágeis na AWS.

Você pode pensar na jornada de modernização em etapas. A primeira etapa inclui três fases: avaliar, mobilizar, migrar e modernizar. A próxima etapa inclui a fase de operação e otimização, na qual você pode identificar mais oportunidades de inovação.

Tópicos

Fase de avaliação

No nível mais alto, a fase Avaliação analisa se você está pronto para migrar. Você define um caso de negócios e, em seguida, educa sua equipe com workshops e um dia de imersão (demonstrações e laboratórios) oferecidos pela. AWS Workshops e dias de imersão abordam temas diferentes. Essas tarefas são conduzidas fora da modernização do AWS mainframe.

Fase de mobilização

Na fase de Mobilização, você inicia seu projeto com um pontapé inicial e, em seguida, executa um processo de descoberta que extrai dados de suas aplicações de mainframe e os ingere em uma ferramenta de migração. Você identifica as aplicações que deseja migrar e seleciona algumas para testar. Você refina seu caso de negócios, elabora seu plano de migração e decide como deseja lidar com segurança e conformidade, governança de contas e seu modelo operacional. Você configura um centro de excelência em nuvem com as pessoas certas da sua equipe. Você dirige os pilotos e documenta o que aprendeu. Você refina seu plano de migração e seu caso de negócios. Muitas dessas tarefas são conduzidas fora da modernização do AWS mainframe.

Fase de migração e modernização

A fase de migração e modernização se aplica a cada aplicativo e consiste em várias tarefas, incluindo designar pessoas, executar descobertas aprofundadas, descobrir a arquitetura correta do aplicativo, configurar ambientes de tempo de execução de aplicativos AWS, reformular a plataforma ou refatorar seu código, integrá-lo a outros sistemas e, é claro, testar. No final da fase, você implanta as aplicações refatoradas ou com plataformas redefinidas na produção e transfere para o novo sistema na AWS. A maioria ou todas essas tarefas são conduzidas na Modernização do AWS Mainframe, em outro AWS serviço ou em uma ferramenta à qual a Modernização do AWS Mainframe fornece acesso.

Se você quiser usar a refatoração automatizada, consulte Blu Insights. AWS O Blu Insights agora está disponível a AWS Management Console partir do login único. Você não precisa mais gerenciar credenciais separadas do AWS Blu Insights. Você pode acessar os recursos do AWS AWS Blu Age Codebase e do Transformation Center diretamente do. AWS Management Console

Para migrar dados do mainframe para AWS, recomendamos o. AWS SCT e o. AWS Database Migration Service Para obter mais informações, consulte O que é o AWS Schema Conversion Tool? no Guia do usuário do AWS Schema Conversion Tool e O que é o AWS Database Migration Service? no Guia do usuário do AWS Database Migration Service .

Opere e otimize a fase

Na fase Operar e Otimizar, você se concentra em monitorar suas aplicações implantadas, gerenciar recursos e garantir que a segurança e a conformidade estejam atualizadas. Você também avalia as oportunidades de otimizar os workloads migrado.

Conceitos

AWS A modernização do mainframe fornece ferramentas e recursos para ajudá-lo a migrar, modernizar e executar cargas de trabalho do mainframe. AWS

Tópicos

Aplicativo

Uma carga de trabalho de mainframe em execução na modernização do AWS mainframe. Um conjunto de trabalhos em lotes, transações interativas (CICS ou IMS) ou outros componentes compõem uma aplicação. Você define o escopo. Você deve definir e especificar quaisquer componentes ou recursos que o workload precise, como transações do CICS ou trabalhos em lotes.

Definição da aplicação

A definição ou especificação dos componentes e recursos necessários para um aplicativo (carga de trabalho do mainframe) executado na modernização do AWS mainframe. Separar a definição da aplicação em si é importante porque é possível reutilizar a mesma definição para vários estágios (pré-produção, produção), representados por diferentes ambientes de runtime.

Trabalho em lote

Um programa agendado que é configurado para ser executado sem exigir interação do usuário. No AWS Mainframe Modernization, você precisará armazenar os arquivos JCL do trabalho em lote e os binários do trabalho em lote em um bucket do Amazon S3 e fornecer o local de ambos no arquivo de definição da aplicação. Quando você executa um trabalho em lotes, a Modernização do AWS Mainframe relata os seguintes valores de status:

Enviando

A tarefa em lote está em processo de envio.

Em espera

O trabalho em lotes está suspenso.

Expedição

A tarefa em lote está em processo de envio.

Running

O trabalho em lote está em execução no momento.

Cancelando

A tarefa em lote está em processo de cancelamento.

Cancelado

O trabalho em lotes foi cancelado.

Bem-sucedida

A execução do trabalho em lote foi concluída com êxito.

Failed (Falha)

O trabalho em lotes falhou.

Foi bem-sucedido com o aviso

A execução do trabalho em lote foi concluída com êxito, com um pequeno erro relatado. O código de condição do trabalho retornado como parte da GetBatchJobExecution resposta indica a causa do erro.

Configuração

As características de um ambiente ou aplicação. As configurações do ambiente consistem em tipo de mecanismo, versão do mecanismo, padrões de disponibilidade, configurações opcionais do sistema de arquivos e muito mais.

As configurações da aplicação podem ser estáticas ou dinâmicas. As configurações estáticas mudam somente quando você atualiza uma aplicação implantando uma nova versão. As configurações dinâmicas, que geralmente são uma atividade operacional, como ativar ou desativar o rastreamento, mudam assim que você as atualiza.

Conjunto de dados

Um arquivo contendo dados para uso por aplicações.

Ambiente

Uma combinação nomeada de recursos AWS computacionais, um mecanismo de tempo de execução e detalhes de configuração criados para hospedar um ou mais aplicativos.

Mainframe Modernization

O processo de migração de aplicativos de um ambiente de mainframe legado para o. AWS

Jornada de migração

O end-to-end processo de migração e modernização de aplicativos legados, normalmente composto pelas seguintes fases: avaliar, mobilizar, migrar e modernizar e operar e otimizar.

Ponto de montagem

Um diretório em um sistema de arquivos que fornece acesso aos arquivos armazenados nesse sistema.

Refatoração automatizada

O processo de modernização de artefatos de aplicações ultrapassadas para execução em um ambiente de nuvem moderno. Ele pode incluir conversão de código e dados. Para obter mais informações, consulte AWS Mainframe Modernization Automated Refactor.

Redefinir plataformas

O processo de mover uma aplicação e seus artefatos de uma plataforma de computação para outra. Para obter mais informações, consulte AWS Mainframe Modernization Replatform

Recurso

Um componente físico ou virtual em um sistema de computador.

Mecanismo de execução

Software que facilita a execução de uma aplicação.

Refatorando aplicativos automaticamente com o Blu Age AWS

A refatoração automatizada com o AWS Blu Age fornece uma end-to-end solução para migrar e modernizar seus aplicativos de mainframe. As etapas do processo de refatoração são as seguintes:

- Analise o inventário
- Analise dependências
- Transformar código automaticamente
- Capture e gerencie cenários de teste

Você pode concluir as etapas anteriores na ferramenta Blu Insights, disponível por meio de login único no console de modernização do AWS Mainframe. Para obter mais informações sobre o Blu Insights, consulte a documentação do Blu Insights.

Quando estiver satisfeito com o código-fonte transformado, é hora de mudar para AWS, onde você concluirá as seguintes etapas:

- Crie e implante a aplicação refatorada.
- Implante e monitore seu aplicativo na modernização AWS do mainframe.

AWS O Blu Age Runtime (não gerenciado) é uma das ofertas do serviço de modernização de AWS mainframe junto com o Blu Age gerenciado. AWS Com o AWS Blu Age gerenciado, você pode implantar seu aplicativo modernizado em um ambiente AWS gerenciado que simplifica sua experiência, para que você não precise gerenciar a infraestrutura subjacente que executa seu aplicativo modernizado. Por outro lado, com o AWS Blu Age Runtime (não gerenciado), você pode implantar seu aplicativo modernizado em sua própria AWS conta, para poder gerenciar sua própria infraestrutura. Com o AWS Blu Age Runtime (não gerenciado), você tem a flexibilidade de operar todos os componentes técnicos necessários para executar seu aplicativo modernizado da maneira que desejar.

AWS O Blu Age Runtime (não gerenciado) está disponível para implantação em:

- Amazon EC2
- Amazon ECS no Amazon EC2

- Amazon EKS no Amazon EC2
- Amazon ECS gerenciado por AWS Fargate

A implantação no Amazon EC2 (as três primeiras opções na lista anterior) pode ser feita diretamente na instância ou por meio de um aplicativo Docker em contêineres, que é a forma preferida ao usar o Amazon ECS ou o Amazon EKS.

Tópicos

AWS Notas de lançamento do Blu Age

Esta seção contém as notas de lançamento do AWS Blu Age Runtime and Modernization Tools da versão 3.5.0 em diante, a mais recente, organizada por número de versão.

> **Note**
>
> Para notas de lançamento anteriores a este documento, entre em contato com os serviços de entrega da AWS Blu Age. Para obter informações sobre os recursos mais recentes do Blu Insights, consulte Blu Insights Releases.

Tópicos

Notas de lançamento 4.1.0

Esta versão do AWS Blu Age Runtime and Modernization Tools tem como foco desempenho e segurança. Alguns dos principais recursos e mudanças nesta versão são:

- Transformação e desempenho: para permitir que projetos com uma grande base de código (mais de 50 milhões de linhas de código) se transformem com sucesso, otimizamos o desempenho e o espaço de memória de todo o mecanismo de transformação.
- BAC/JAC: a segurança AWS é a maior prioridade. Os aplicativos modernizados com o AWS Blu Age devem estar em conformidade com os padrões de segurança. Fizemos algumas atualizações importantes no BluSam Administration Console (BAC) e no JICS Administration Console (JAC) para torná-los mais seguros:
  - Atualizou o aplicativo para Angular v17.

  - Além do suporte nativo para o AWS Cognito, adicionamos suporte genérico para OAuth, que permitirá mais flexibilidade para permitir que os clientes usem o provedor de identidade de sua escolha.
  - Configurou e ampliou os recursos de segurança usando cabeçalhos apropriados.
- AS400 - Suporte de vários nós para mecanismo de bloqueio de banco de dados. Ofereceu a possibilidade de conectar um servidor de cache externo e compartilhado (Redis) para executar um aplicativo em lote em várias instâncias, como a modernização gerenciada do AWS mainframe.

Essa versão do tempo de execução do Blu Age foi testada com a seguinte pilha. Outras versões também podem ser compatíveis.

Componente	Versão testada
Java	Java 17
Camada de apresentação	Modo JS 18.18, Npm 9.8, Angular 16.1
Camada de serviço	Spring Boot 3.2.5, Spring Core 6.1.5, Spring Statemachine 4.0.0
Camada de persistência	PostgreSQL 14, Oracle 21c
Servidor de aplicativos	Apache Tomcat 10.1.17

Para obter mais informações sobre as alterações incluídas nesta versão, consulte as seguintes seções:

Runtime versão 4.1.0

zOS

Novos atributos

- Configuração adicionada para tratamento dinâmico do provedor OAuth2. Introduziu SECRET_OAUTH2_PROVIDER_NAME_KEY para especificar o provedor. Método de recuperação secreta atualizado para lidar com vários provedores. Os segredos garantidos sejam recuperados com segurança de. AWS Secrets Manager

- Foi adicionado suporte às propriedades SSL do DB2 AWS Secrets Manager para possibilitar a definição de um certificado SSL (sslTrustStoreLocal) e uma sslTrustStore senha (Senha) para desbloquear o arquivo de armazenamento de chaves.
- Foi adicionado suporte para fontes externas de dados comerciais.
- JCL - Foi adicionado suporte ao mecanismo de ponto de verificação para reinicialização em lote.
- JCL - Adicionado suporte para tamanho de registro de parâmetros DCB e RDW.
- JCL - Adicionada configuração dinâmica de nome de pasta para arquivos temporários gerados.
- REDIS - Configuração de pool adicionada na configuração do Redis para JICS.
- REDIS - Índice de banco de dados adicionado na configuração do Redis para Catalog e JICS.
- BatchScript - Foi adicionada a propagação do nome da etapa para executar execuções de programas.
- CICS - Foi adicionado suporte para o comando ADDRESS SET.
- CICS - Adicionado suporte para PURGE MESSAGE e JUSTIFY.

Melhorias

- JCL - INFUTILB - Suporte aprimorado para desativar o indicador nulo com base na propriedade YML.
- JCL - INFUTILB - Suporte aprimorado para o tipo de dados CHAR/BPCHAR.
- JCL - ICEGENER - Adicionado suporte para copiar fluxos de entrada de várias linhas em arquivos.
- JCL - IEBGENER - Suporte aprimorado para lidar com a conversão de arquivos de bloco variável para bloco fixo.
- JCL - DFSORT - Suporte aprimorado para parâmetros de vários dígitos na operação DATE.
- JCL - DFSORT - Foi adicionado suporte para a cláusula INCLUDE=ALL.
- JCL - Suporte aprimorado para o utilitário SORT para lidar com o campo BDW na saída.
- JCL - Suporte aprimorado para concatenação de DD.
- JCL - Suporte aprimorado para fluxo de entrada.
- JCL - DSNUTILB - Suporte aprimorado para a instrução NULLIF ().
- JCL - INFUTILB - Adicionado suporte para descarregar dados com a opção NOPAD.
- JCL - INFUTILB - Suporte aprimorado para a data atual no INFUTILB.
- JCL - Foram adicionadas verificações de existência e tamanho do arquivo antes de usar um arquivo.

- JCL - GDG - Melhorou o tratamento de subdiretórios para o GDG.
- MQ - Abertura de conexão aprimorada na implementação do JMS.
- MQ - Configuração aprimorada do comprimento de dados da mensagem GET para a fonte de dados XA.
- MQ - caderno padrão CMQV decomposto para evitar erros de compilação e usos de refatoração.
- BluSam - Suporte aprimorado para solicitações de exclusão de conjuntos de dados inexistentes.
- Suporte aprimorado para a instrução ALLOCATE.
- Robustez aprimorada da nomenclatura TS-QUEUE.
- BatchScript - Preservação aprimorada do código de retorno da etapa anterior na reexecução do trabalho.
- Conjunto de dados - Melhorou a verificação da existência do arquivo quando um arquivo existe e é temporário.
- Conjunto de dados - Melhorou a simultaneidade ao localizar arquivos GDG para excluir.
- Conjunto de dados - Foi adicionado suporte para obter o tamanho do registro do conjunto de dados GDG.
- CICS - Suporte aprimorado para a opção SUSPENDED no comando INQUIRE TASK LIST.
- CICS - Suporte aprimorado para LOAD SET usando a instrução ADDRESS OF.
- CICS - Argumentos CICS não manipulados aprimorados REMOTESYSTEM quando o CICS INQUIRE.
- CICS - Suporte aprimorado para o comando GETMAIN para lidar com a opção SET com um ponteiro definido com a palavra-chave OF.
- JICS - Robustez aprimorada do método jicsXapRepare () adicionando a verificação do estado da transação.
- JICS XA - Foi adicionada uma verificação do estado da transação e o encerramento aprimorado do encadeamento da transação.
- BAC - Autenticação aprimorada baseada em funções no lado do cliente e refatorada/centralizada em todas as chamadas de API.
- BAC - Implementou um recurso para bloquear o acesso público ao BAC e ao JAC com base na configuração
- BAC - Atualização das dependências: Angular 17.
- BAC - Integração de segurança aprimorada com OAuth2 - /FIDIS. StateFarm

- BAC - DDL aprimorado gerado por hibernação.
- BAC - Mecanismo aprimorado do conjunto de dados de exportação.
- JAC - Atualizado para o Angular 17 e relatando todos os detalhes do trabalho do BAC (ROLE, sadmin conf, XSRF, logout).
- COBOL - Foi adicionado suporte para as funções CHAR e ORD-MIN.
- Aprimorado FileFactory para manter o tamanho do registro do catálogo na disposição MOD.
- Registro habilitado usando MDC para transações JICS.
- SQLCA > SQLSTATE aprimorado produzido para procedimentos armazenados que geram conjuntos de resultados ad-hoc.
- Suporte aprimorado para agendamento de tarefas relacionadas à última atualização do Spring.

AS400

Novos atributos

- Foi adicionado suporte a vários nós para bloqueios de registros de banco de dados usando o Redis.
- Foi adicionado suporte para BINARY CHARACTER para o tipo DDS.
- CL - Adicionado suporte para geração de arquivos de relatórios personalizados.
- RPG - Foi adicionado suporte para a palavra-chave RENAME em arquivos primários/secundários.

Melhorias

- Suporte aprimorado ao banco de dados para lidar com a coluna CTID com uma cláusula JOIN.
- Posição aprimorada do cursor para vários DSPATR (PC).
- Registro aprimorado na exceção de leitura.
- Registro de tarefas do Quartz aprimorado para incluir propriedades de trabalho no MDC.
- Suporte aprimorado para a tela de ajuda do AS400.
- CL - Suporte aprimorado para o comando RMVJOBSCDE para aceitar números de entrada com espaços à direita.
- CL - Suporte aprimorado para o comando RMVJOBSCDE para remover um agendamento de trabalho usando um nome de trabalho genérico.
- CL - Suporte aprimorado para o comando SAVOBJ para ordenar registros por chave de tabela.

- CL - Suporte aprimorado para o comando CPYF para estabelecer uma nova conexão para consultas de banco de dados.
- CL - Inserção aprimorada de mensagens de consulta na fila de mensagens com SNDPGMMSG.
- CL - Configuração aprimorada da fila de trabalhos para especificar a fila de trabalhos padrão.
- CL - Melhorou o comando CRTPF para suportar a biblioteca QTEMP e o parâmetro RCDLEN.
- CL - Suporte aprimorado para o comando CHKOBJ - Verifique a partição com a biblioteca.
- CL - RTVMGS aprimorado para enviar CPF2407 e CPF2419 quando o arquivo/ID não foi encontrado.
- CL - Interpretação aprimorada de CPYTOIMPF e CPYFRMIMPF dos parâmetros de formatação antigos.
- CL - Adicionado suporte para o parâmetro OVRPRTF USRDTA.
- CL - Melhorou o comando CPYTOIMPF CL para estabelecer uma nova conexão e evitar o fechamento de conjuntos de resultados existentes.
- CL - CHGDTAARA aprimorado para que ele não modifique mais o comprimento da área de dados ao atualizar o conteúdo.
- CL - Melhor gerenciamento ClCommand de conexão de banco de dados.
- Interação otimizada entre o front-end e o back-end.
- COBOL - Transformação atualizada para lidar com FILLER em cadernos.
- Exibição aprimorada de informações adicionais de mensagens para mensagens personalizadas enviadas ao front-end.
- Atualizou o valor padrão do seletor em app.component.ts.
- Divisão de texto aprimorada na split-dynamic-field exibição.
- Melhorou a exibição da mensagem de erro com várias gravações seguidas por uma leitura.

Recursos transversais

Novos atributos

Foi adicionado suporte para a configuração dinâmica do segredo do provedor OAuth2.

Melhorias

- Impressão - Suporte aprimorado aos parâmetros QCMDEXC para lidar com aspas e melhor formação de nomes de relatórios

- Suporte aprimorado para sintaxe delimitada ativada. RecordAdaptable
- Registro InspectBuilder de erros aprimorado para adicionar contexto sobre a string de origem.
- DataSimplifier - maior robustez para ByteArray afetação.
- Registro aprimorado do MDC com novos atributos de tempo de execução.

Ferramentas de modernização versão 4.1.0

zOS

Novos atributos

- Suporte adicionado para várias transformações de arquivos CSD
- COBOL - Foi adicionado suporte para a instrução CICS ALLOCATE.
- COBOL - Foi adicionado suporte para ON SIZE ERROR na instrução ADD CORRESPONDENING.
- COBOL - Foi adicionado suporte para EXIT PARAGRAPH.

Melhorias

- COBOL - Suporte aprimorado para o caderno -INC.
- COBOL - Suporte aprimorado para inicialização do FILLER.
- COBOL - Suporte aprimorado para comparação de valores figurativos.
- COBOL - Suporte aprimorado para WHEN ANY em cláusulas WHEN consecutivas sem blocos de código intermediários.
- COBOL - Suporte aprimorado para constante figurativa.
- COBOL - Suporte aprimorado para cálculo de tamanho de tipo compactado.
- COBOL - Melhorou o argumento CICS não tratado KEEP para SPOOLCLOSE.
- COBOL - Geração aprimorada para a função TEST-NUMVAL.
- COBOL - Argumentos de geração Java aprimorados no suporte à estrutura INSPECT.
- CICS - Suporte aprimorado para definir DFHCOMMAREA.

AS400

Novos atributos

- RPG - Foi adicionado um mecanismo de captura de erros para gerar o DDS (incompleto) para que ele não bloqueie a geração do programa.
- Foi adicionado suporte para a palavra-chave de especificação de descrição de arquivo INCLUDE.

Melhorias

- RPG - Análise totalmente gratuita aprimorada.
- RPG - Maior robustez com detecção de erros.
- RPG - Inicialização aprimorada do Field/DS com a palavra-chave export.
- RPG - Operação DAO aprimorada para lidar com indicadores.
- RPG - Manipulou o valor padrão de PERRCD com CTDATA.
- RPG - Atualizou o analisador de RPG gratuito para registrar um erro exclusivo por regra de análise.
- PRTF - Colisão de nomes manipulada entre PRTF e JRXML.
- COBOL - Suporte aprimorado da palavra-chave LIKE.

Recursos transversais

Melhorias

- Robustez adicional para a API ErrorID
- Otimização de desempenho para grandes transformações de projetos. Por exemplo: tempo limite para ignorar arquivos bloqueados, reutilização da classificação do Blu Insights e melhores alocações de memória.
- Otimizou o espaço de memória durante a transformação COBOL/PL1.
- CVE fixo em terceiros (jQuery e bootstrap).
- Opções gerenciadas de TimeoutParser no TC.
- Melhorou a reescrita de vários espaços em consultas SQL.
- Cursor somente de leitura aprimorado com atributo de sensibilidade.

Notas de lançamento 4.0.0

Para obter instruções sobre como migrar do AWS Blu Age Runtime 3.10.0 para 4.0.0, consulte. the section called “Migrando da versão 3.10.0 para a 4.0.0”

Esta versão do AWS Blu Age Runtime and Modernization Tools está focada na atualização de dependências críticas e tecnologias suportadas, ao mesmo tempo em que aumenta o desempenho em várias funcionalidades. Alguns dos principais recursos e mudanças nesta versão são:

- • Atualize do Spring Boot 2.7 para 3.2.4, do Spring Core 5.3 para 6.1.5 e do Tomcat 9.0 para 10.1.17 para fornecer segurança, desempenho e capacidade de manutenção aprimorados usando versões que estão sendo ativamente corrigidas e mantidas.
- Carregamento lento no aplicativo front-end para criar grandes projetos mais rápidos com mais de 2.000 telas e reduzir a inicialização da exibição de 10 s para 300 ms.
- Support para exibição de DBCS em aplicativos front-end. Aprimoramento do suporte a caracteres de byte duplo para fornecer uma nova fonte que manipule caracteres de byte duplo e byte único, evite a entrada de byte único em um campo de byte duplo e manipule campos com caracteres mistos de byte duplo e byte único.
- Recurso de monitoramento de threads para o aplicativo AS400 Online para executar o aplicativo AS400 com paralelização.
- Melhor desempenho no contexto e na RunUnit inicialização com a adição de um mecanismo configurável para pré-inicializar o contexto do programa, reduzindo o impacto do carregamento de estruturas complexas inerentes à complexidade herdada.

Essa versão do AWS Blu Age Runtime foi testada com a seguinte pilha. Outras versões também podem ser compatíveis.

Componente	Versão testada
Java	Java 17
Camada de apresentação	Node JS 18.18
	Npm 9,8
	Angular 16.1
Camada de serviço	Bota Spring 3.2.4
	Spring Core 6.1.5
	Máquina de estado Spring 4.0.0

Camada de persistência	Motor PostgreSQL 14
	Oráculo 21
Servidor de aplicativos	Apache Tomcat 10.1.17

Para obter mais informações sobre as alterações incluídas nesta versão, consulte as seguintes seções:

Runtime versão 4.0.0

zOS

Novos atributos

- Foi adicionado suporte para a declaração de inclusão '-INC CPYNAME'.
- CICS - Foi adicionado suporte para a instrução PUSH/POP HANDLE.
- COBOL - Foi adicionado suporte para “ASSIGN TO DYNAMIC”.
- Foi adicionado suporte para DB2 UNLOAD usando INFUTILB.
- Foi adicionado suporte para a palavra-chave SEQNUM em uma declaração OVERLAY of INREC.

Melhorias

- SORT - Foi adicionado suporte para caracteres especiais (parênteses e asteriscos) em literais de string de classificação C'... '.
- SORT - Suporte aprimorado para o argumento OUTFIL NOMATCH- (..).
- SORT - Foi adicionado suporte para a definição de dados SYMNAMES.
- SORT - Melhor manipulação dos argumentos TO= e LENGTH=.
- SORT - Melhor manuseio na disposição do MOD.
- SORT - Adicionado suporte para o argumento HIT=NEXT.
- ICEGENER aprimorado para adicionar suporte para codificação específica de arquivos de saída.
- INFUTILB - Suporte aprimorado para a cláusula WITH UR.
- INFUTILB - Suporte aprimorado para descarga quando é falso. writeNullIndicator

- DSNUTILB - Robustez aprimorada na etapa de carregamento quando a palavra-chave NULLIF está depois de uma palavra-chave SQL opcional.
- DSNUTILB - Suporte aprimorado para nome de coluna isolado.
- DSNUTILB - Foi adicionado suporte para carregar um arquivo vazio em uma tabela.
- DNSUTILB - Foi adicionado suporte para disposição de MOD para o arquivo DNSUTILB SYSDISC.
- IDCAMS - Suporte aprimorado para comentários.
- JCL- Adicionado suporte para coluna com aspas duplas LoadTask.
- JCL- Tratamento aprimorado de consultas SQL UNLOAD em relação à remoção de espaços brancos.
- JCL- Resposta aprimorada do script Groovy quando ocorre uma exceção no processamento para garantir um formato JSON.
- JCL- Melhor disposição do arquivo de verificação no caso de DISP=NEW e DISP=OLD.
- JCL - Suporte aprimorado para lidar com várias referências de geração de GDG com caracteres especiais no nome base do GDG.
- JCL- Suporte aprimorado para carregar um arquivo fictício.
- JCL - Suporte aprimorado para o parâmetro tempFilesDirectory YML.
- JCL - Retorno JSON aprimorado quando é necessário escapar de aspas duplas em um elemento de string.
- JCL - Aprimorado FileUtils para suportar o nome base GDG.
- JCL - Programa DSNTEP aprimorado para execução de várias consultas do DB2.
- Foi adicionado suporte para feijão Spring.
- SQLConverter aprimorado para evitar a retificação de datas erradas.
- Melhor JicsTimeBuilder manuseio do YYYYDDD.
- Permitiu que frascos personalizados fossem acessíveis a partir do groovy.
- IMS - Navegação aprimorada entre registros na implementação do banco de dados IMS.
- IMS - CBLTDLI aprimorado para poder iniciar a limpeza de uso do programa.
- IMS - DFSRRC00 capaz de passar os parâmetros do programa groovy para o back-end.
- Foi adicionado suporte para o comando JICS que não foi chamado por meio de um TransactionRunner.
- JICS - Desempenho aprimorado usando cache configurável.

- BluSam - Adicione suporte para desativar o aquecimento BluSam ao abrir para melhorar o desempenho de grandes conjuntos de dados.
- BluSam- Melhor comportamento de exclusão/renomeação em conjuntos de dados regulares. BluSam
- BluSam - Desempenho aprimorado em operações de gravação.
- Simplificador de dados aprimorado para os métodos que determinam se uma string tem um valor baixo.
- Suporte aprimorado para problemas de decimal compactado e ordem de classificação.
- Configuração aprimorada do DB2 como fonte de dados primária com os segredos da AWS.
- FileSystem API aprimorada para expor o status do arquivo.
- Entrada de fluxo de DynamicFileBuilder leitura aprimorada com LineSeparator.
- Simplificador de dados aprimorado para os métodos que determinam se uma string tem um valor baixo quando lida com o conjunto de caracteres CUSTOM930.
- SQL - Processamento aprimorado de saída de procedimentos armazenados em SQL.
- SQL - Mapeamento lambda aprimorado para várias tabelas com aliases.
- COBOL - Suporte aprimorado para a declaração LENGTH OF.
- COBOL - Foi adicionado suporte para a instrução TRANSFORM.
- COBOL - Adicionado suporte para 9 novas funções matemáticas.
- COBOL - Suporte aprimorado para a FUNÇÃO INTEGER-OF-DAY.
- COBOL - Suporte aprimorado para 88 níveis envolvendo valores figurativos.
- COBOL - Transformação aprimorada para a instrução SET ADDRESS.

AS400

Novos atributos

- Foram removidas entidades indicadoras duplicadas.
- Foi adicionado suporte para caracteres DBCS.
- Introduziu o tratamento da palavra-chave HELP para controle de registros de subarquivos.
- Parâmetro de configuração adicionado para alternar a capitalização do nome da coluna e dividir o conteúdo da coluna de comentários no pipe char.
- Foi adicionado suporte para usar 0x0c como última opção para campos do tipo Packed.

- RPG - Protótipos manipulados declarados com ExtProc ('sistema').
- CL - O parâmetro 'CLEAR' manipulado do comando cl RMVMSG + introduz filas de mensagens não programadas na memória.
- CL - Instruções genéricas tratadas sendo passadas para chamadas SBMJOB CMD ().
- CL - Adicionado o comando STRCMTCTL e ENDCMTCTL. Mecanismo de bloqueio modificado e limpeza de transações e bloqueios.
- CL - Adicionado suporte para o parâmetro RCDDLM para o comando CPYTOIMPF.
- CL - Foi adicionado o tratamento de zeros de preenchimento no comando SAVOBJ.
- CL - Foi adicionada a manipulação de bibliotecas incluídas no nome qualificado do parâmetro OBJ para RTVOBJD.
- CL - Adicionado suporte para os parâmetros de comando CPYTOIMPF STRDLM, STRESCCHR e RMVBLANK.
- CL - RTVMGS aprimorado para enviar CPF2407 e CPF2419 quando o arquivo/id não foi encontrado.
- CL - Comando RCVF aprimorado para receber registros de qualquer biblioteca fornecida no parâmetro DEV.

Melhorias

- Valores padrão alterados para o executor de tarefas Blu4iv para permitir um melhor dimensionamento por padrão.
- Parameterhelper modificado para converter a lista de strings e em String. ElementaryRangeReference
- CTID aprimorado para lidar com colunas não existentes no POSTGRE.
- Robustez adicionada para suportar a API de espaço do usuário “QUSPTRUS”.
- Foi adicionado suporte para as APIs de espaços de usuário QUSRUSAT e QUSCUSAT.
- Suporte aprimorado para a API de espaço do usuário (QUSPTRUS) sem código de erro.
- Foi adicionado suporte para CRON Job Scheduling usando Quartz.
- Suporte aprimorado do ciclo do programa de RPG.
- Gerenciamento de transações Blu4iv aprimorado.
- O bloqueio de registros de arquivos sob controle de compromisso na mesma transação foi aprimorado.

- Tratamento aprimorado da inicialização de subarquivos.
- Exibição aprimorada de indicadores de rolagem para linhas de mensagem.
- Evitou zeros à direita em números enviados por meio da fila de dados.
- Tela de informações adicionais de mensagens aprimorada.
- Operações de gravação JPA aprimoradas para considerar a biblioteca atual.
- Comportamento aprimorado ProgramJobExecutor ao executar programas sem parâmetros.
- Funcionalidade adicionada para passar argumentos diretamente dos links de front-end para scripts de back-end.
- Tratamento aprimorado de transações para metadados de trabalhos.
- CL - Adicionado suporte para o parâmetro SECLVL no RTVMSG.
- CL - Adicionada implementação vazia para CLRLIB.
- CL - Suporte aprimorado do CPYFRMIMPF para copiar do banco de dados e do CSV.
- CL - Implementação aprimorada do CPYFRMIMPF para ignorar colunas extras.
- CL - Interpretação aprimorada de CPYTOIMPF e CPYFRMIMPF dos parâmetros de formatação antigos.
- CL - Parâmetro adicionado removeDecimalPoint para formatar valores numéricos no SAVOBJ.
- CL - Comando RCVF aprimorado para lidar adequadamente com a condição EOF.
- CL - RTVSYSVAL - Implementação SYSVAL = QDATETIME.
- CL - Comando OVRDBF modificado para obter FieID como nome de tabela padrão.
- CL - RTVJOBA Valor indisponível para o parâmetro: USRLIBL.
- CL - Foram tratadas as barras iniciais no parâmetro SNDPGMMSG MSGF.
- CL - Suporte aprimorado para curingas no arquivo de origem no comando DSPFFD.
- CL - Melhor manipulação do parâmetro PGMQ em RCVMSG e SNDPGMMSG.
- CL - Tornou a MSG do parâmetro RTVMSG opcional para alinhar com documentos legados.

Recursos transversais

Novos atributos

- Capacidade aprimorada ao passar o parâmetro na cláusula USING do cursor OPEN.
- Desempenho: pré-inicialização aprimorada do contexto e ajuste RunUnit de desempenho.

Melhorias

- Melhorou o mecanismo para despejar valores baixos do comando UNLOAD do programa utilitário INFUTILB.
- Foi adicionado suporte à opção de esquema atual no gerenciador secreto de fontes de dados.
- Tempo de execução aprimorado para não considerar os parâmetros passados no cursor aberto quando não são necessários.
- Validação aprimorada do formato numérico para campos numéricos.
- Diagnóstico SQL aprimorado em ambiente de execução altamente paralela.
- Introduziu o unicode para sequência de bytes da página de código (FE FD).
- DataSimplifier otimização de desempenho - instruções de atribuição aprimoradas.
- DataSimplifier otimização de desempenho - Melhore o valor padrão para inicialização do tipo numérico para evitar o uso inútil. BigDecimal

Ferramentas de modernização versão 4.0.0

zOS

Novos atributos

- Foi adicionado suporte para lidar com o Abend PROGRAM.
- Suporte aprimorado para gerar conjunto de dados AIX.
- COBOL - Foi adicionado suporte para a cláusula JUSTIFIED nos campos ALFANUMÉRICO/ALFABÉTICO/GRÁFICO.

Melhorias

- Tratamento aprimorado do atributo PURGETHRESH para definições de recursos TRANSCLASS.
- Suporte aprimorado para definição de dados e declaração MOVE.
- CICS - Suporte aprimorado para o comando DELAY na opção MILLISECS.
- Mapeamento lambda SQL aprimorado para várias tabelas com aliases.
- Suporte aprimorado para localização do campo principal.
- Conjunto SQLCA sqlstate aprimorado para operação COMMIT e ROLLBACK.

- COBOL - Melhore a análise comentando parágrafos obsoletos
- COBOL - Suporte aprimorado para a cláusula REPLACING.
- COBOL - Adicionado suporte para funções matemáticas ASIN ACOS LOG TAN.
- COBOL - Foi adicionado suporte para várias instruções AFTER em PERFORM VARYING.
- COBOL - Suporte aprimorado para campos RENAMES (nível 66).
- COBOL - Método LENGTH OF aprimorado para obter o comprimento em um índice específico em um campo de matriz.
- COBOL - Foi adicionado suporte para várias cláusulas AFTER nas instruções PERFORM VARYING.
- COBOL - Suporte aprimorado para a cláusula RENAMES.
- COBOL - Suporte aprimorado da palavra-chave PICTURE.
- COBOL - Suporte aprimorado para análise de campo de nível 88.
- COBOL - Goto aprimorado, dependendo da condição, com itens de dados da tabela.

AS400

Novos atributos

- Funcionalidade adicionada para passar argumentos para chamadas java diretas de front-end.
- CL - Geração aprimorada de %SST, incluindo suporte para *LDA com CL→Java.
- RPG - Adicionado suporte ao registro descrito pelo programa para arquivos DISK.

Melhorias

- Arquivo de exibição aprimorado, resolva os campos referenciados com a palavra-chave “REFFLD”.
- Suporte aprimorado da palavra-chave do arquivo de exibição SETOF-CSRLOC.
- Arquivos removidos do controle de compromisso após o fechamento.
- Comportamento consistente garantido para operações simultâneas de leitura e gravação em uma tabela quando executadas pelo mesmo programa.
- Atribuição manipulada à substring de. SizePrefixedAlphanumericType
- Lidou com a passagem da estrutura de dados para o procedimento com um parâmetro de string de comprimento variável.

- Retenção aprimorada de valores numéricos inválidos no evento OnBlur e criação de ouvintes de eventos somente para campos válidos.
- Mensagens de erro aprimoradas nas telas e destaque de campos com entrada inválida.
- Melhor manuseio dos campos da tela condicionados aos indicadores.
- Rolagem ativada com a roda do mouse.
- Foi adicionado suporte para teclas de função na tela de Ajuda.
- Suporte aprimorado para texto longo no split-dynamic-field componente.
- Manipulação aprimorada de arquivos LF com vários registros ao renomear registros.
- CL - Comando RTVJOBD aprimorado para lidar com arquivos LF (visualizações).
- CL - Comando OVRDBF aprimorado quando usado em um LF de vários registros.
- RPG - Cenário tratado em que o procedimento define uma variável com o mesmo nome do parâmetro renomeado.
- RPG - Melhor manuseio de*ZEROS ao inicializar o BinaryInteger assinado.
- RPG - Melhor manipulação de ponteiros para variáveis não locais (de referência).
- RPG - Melhor tratamento das declarações do ELSEIF após as declarações do iFxx.
- RPG - Foi adicionado suporte para campos definidos com LIKE no protótipo.
- RPG - Melhorou o suporte para a palavra-chave LIKE de um campo criado pelo LIKEREC.
- RPG - Geração aprimorada de operadores com figuras.
- RPG - Análise aprimorada da expressão de matriz xxx (\ *) e suporte para ela em %lookup.
- RPG - Código de LookUp operação aprimorado com indicadores altos e iguais (ou baixos e iguais).
- RPG - Análise aprimorada de formulários livres.
- RPG - Análise aprimorada de constantes nomeadas pelo i-Card que seguem os formatos de registro do i-Card.
- RPG - Suporte aprimorado para o tipo INTEGER e UNSIGNED.
- COBOL - Foi adicionada a cláusula INDIC de suporte do formato DSPF na instrução COPY DDS.
- COBOL - Gramática aprimorada para instruções DISPLAY e ACCEPT para desbloquear a transformação e a geração.
- COBOL - Adicionado suporte para arquivos DISK.
- COBOL - Programas aprimorados de suporte a arquivos de exibição do DDS.
- COBOL - Adicionado suporte para a cláusula LIKE.

- COBOL - Adicionado suporte para o arquivo DISK descrito pelo programa.
- COBOL - Adicionado suporte para nome de arquivo com sufixo.

Recursos transversais

Novos atributos

- Lidou com o carregamento lento dos componentes do mapa de projetos da web.

Melhorias

- Geração java aprimorada de parâmetros de indicadores SQL.
- Capacidade aprimorada para lidar com variáveis envolvidas na instrução SET DB2.
- Aumento aprimorado do erro no final do cursor buscado quando a saída é uma matriz de entidade única.
- Caminho gerenciado no Linux.
- O Data Migrator gerencia vulnerabilidades e remove dependências não utilizadas.

Notas de lançamento 3.10.0

Esta versão do AWS Blu Age Runtime and Modernization Tools se concentra nas principais atualizações e melhorias básicas em todo o produto, buscando aumentar o desempenho e a robustez em todas as etapas de transformação e execução. Alguns dos principais recursos e mudanças nesta versão são:

- Atualização da versão do Java 8 para o Java 17, aumentando a segurança e o desempenho e permitindo que os clientes implantem e executem aplicativos implementados em uma linguagem mais moderna e usem versões recentes da estrutura de terceiros.
- Suporte adicional para gerenciar grandes espaços de memória compartilhada entre usuários ou trabalhos, armazenando dados reutilizáveis após a reinicialização do aplicativo ou da instância.
- Acesso mais rápido a grandes conjuntos de dados no Blusam usando um mecanismo de paginação que possibilita a recuperação incremental de um subconjunto de registros.

Para obter mais informações sobre as alterações incluídas nesta versão, consulte as seguintes seções:

Runtime versão 3.10.0

Esse tempo de execução é baseado em Java17, Spring2.7 e Angular16.

zOS

Novos atributos

- Blusam - Adicionado suporte para grandes conjuntos de dados por meio de um mecanismo paginado em que os índices são armazenados e carregados usando páginas

Melhorias

- DataUtils.compare aprimorado para lidar com a conversão de precedência mais baixa de string para número
- Foi adicionado suporte para verificar se não ByteRange é criado com valores impróprios por meio da propriedade YML DataSimplifier. byteRangeBoundsVerifique
- RemoveSosi () aprimorado para suportar a inicialização de um com um GraphicAlphanumericType caractere vazio
- Robustez adicional para operação de trabalho e leitura segura do estado do GDG
- Blusam - Foi adicionado suporte para limpar o Ehcache dos conjuntos de dados Blusam por meio de um novo método chamado .removeCache () CoreBluesamManager
- Blusam - Melhor comportamento de exclusão/renomeação para conjuntos de dados regulares do Blusam
- Redis - Suporte aprimorado para desbloquear conjuntos de dados e limpar o bloqueio de registros
- JICS - Melhorou a mensagem de erro para solicitações com falha
- JCL - Adicionado suporte para concatenação de variáveis ControlM com base no caractere de ponto
- JCL - Adicionado suporte para Write ADVANCING (ADV) para arquivos GDG
- JCL - Suporte aprimorado para o número da geração atual após excluir todos os arquivos GDG
- JCL - Suporte aprimorado para leitura de RDW/RecordSize do catálogo na criação do conjunto de dados
- JCL - Adicionado suporte para atualizar o objeto de recurso (de AbstractSequentialFile) ao abrir o arquivo com o tamanho do registro de saída de dados
- JCL - Melhor desempenho do IDCAMS

- JCL - Suporte aprimorado para PRINT STATEMENT adicionando “CHAR” como alias de “CHARACTER”
- SORT - Suporte aprimorado para operação de cópia de um conjunto de dados de tamanho fixo Blusam para um conjunto de dados com comprimento variável
- SORT - Gramática de classificação aprimorada para lidar com algumas declarações específicas

AS400

Novos atributos

- Foi adicionado suporte para espaços de usuário e suas APIs relacionadas
- Foi adicionado suporte para o parâmetro TOMSGQ do SNDPGMMSG e implementou filas de mensagens
- CL - Adicionado suporte para os parâmetros FILE e SPLFNAME para o comando OVRPRTF
- CL - Adicionado suporte para lidar com bibliotecas para a tabela de partições correspondente com o comando CPYF
- CL - Adicionado suporte para lidar com o comando CHGCURLIB e considerar a biblioteca atual ao criar consultas
- CL - Adicionado suporte para lidar com o comando cl como parte da chamada stacktrace

Melhorias

- Aprimorado MessageHandlingBuilder para melhor lidar com a entrada de rastreamento da pilha de chamadas
- Execução paralela aprimorada do recurso ContextPreconstruct
- Atributos de exibição aprimorados quando um registro é criado pelo SFLINZ
- SAVOBJ aprimorado para permitir o manuseio de vários arquivos de saída
- Manipulação aprimorada de programas groovy ao adicioná-los programCallStack quando são chamados a partir de um programa Java
- Detecção aprimorada do posicionamento superior do modal de ajuda
- Funcionalidade TopGMQ aprimorada quando o parâmetro TomSGQ é fornecido para SNDPGMMSG
- Busca aprimorada de mensagens predefinidas e funcionalidade do carregador de mensagens
- Manipulação aprimorada de CPYTOIMPF de caracteres delimitadores no conteúdo

- Bloqueio de liberação aprimorado no registro READ

Recursos transversais

Novos atributos

- Foi adicionada uma tradução para mensagens do sistema no Front-End
- Foi adicionado um novo método ExecutionContext para retornar a pilha de chamadas do programa
- Defina um separador de linha (para simplificador de dados), independentemente do ambiente real
- Foi adicionada a possibilidade de configurar o caminho JSON do modelo SQL

Melhorias

- Melhorou o método de comparação DataUtils. compareAlphInt() quando o preenchimento está envolvido
- Criação de um sinalizador para permitir comportamento personalizado em caso de exceção nas consultas do cursor
- Conversão gráfica aprimorada de LOWVALUES db

Terceiro

- Atualização para mitigar CVE-2024-21634, CVE-2023-34055, CVE-2023-34462, IN1-JAVA-ORG/SPRINGFRAMEWORKSECURITY-5905484, CVE-2023-46120, CVE-2023-6481, CVE-2023-6378, CVE-2023-5072)

Ferramentas de modernização versão 3.10.0

zOS

Melhorias

- COBOL - Adicionado suporte para a função ABS
- JCL - Escopo variável aprimorado: anexado ao STEP em vez de JOB
- Injeção aprimorada de parâmetros do cursor para valor baixo/alto
- Análise aprimorada de CSD, principalmente para TRANSAÇÕES remotas

AS400

Melhorias

- Verificação em branco removida para o indicador de nível de controle
- Adicionado suporte para nome externo para palavras-chave de IMPORTAÇÃO/EXPORTAÇÃO
- Foi adicionado suporte para %LEN em campos
- CL - Adicionado suporte para novos operadores para a linguagem CLLE
- CL - Adicionado suporte para IF aninhado
- COBOL - Melhor manipulação do comando START quando usado com várias teclas
- DSPF - Melhor manipulação da posição do cursor com número de registro
- DSPF - Melhorou a formatação para campos numéricos assinados, somente numéricos e campos em grande escala
- DSPF - Melhorou a determinação do título para o Screen General Help
- DSPF - Suporte aprimorado das especificações de entrada/saída
- DSPF - Melhor manuseio de separadores de agrupamento durante a validação do campo numérico
- Saída de mapeamento/registros DDS aprimorados
- Capacidade aprimorada da palavra-chave REFFLT do arquivo de impressora para resolver campos referenciados
- RPG - Suporte aprimorado para declarações “TODAS gratuitas”
- RPG - Análise de condições aprimorada e suporte adicionado para lidar com CABXX sem TAG de resultado
- RPG - Manipulação aprimorada da especificação de entrada de campos numéricos
- RPG - Melhor tratamento de chamadas de procedimentos dentro das condições IF/ELSEIF/WHEN
- RPG - Melhor manipulação do comando READ quando chamado em um arquivo dspf
- RPG - Melhore o suporte para arquivos referentes a um DDS inexistente
- Melhore o manuseio do REFFLD ao receber um nome de formato de registro físico
- Suporte adicionado para usar 'return' como nome de coluna db

Recursos transversais

Novos atributos

- Oracle - Tornou possível definir usuários além de SYS para armazenar funções integradas

Melhorias

- Versão Java atualizada de v8 para v17
- Condição SQL aprimorada com o nome da coluna Cluster
- Adicionado suporte para cláusulas ORDER BY a partir da visualização

Notas de lançamento 3.9.0

Esta versão do AWS Blu Age Runtime and Modernization Tools está focada em vários aprimoramentos transversais em todo o produto, buscando aumentar o desempenho em arquiteturas de alta disponibilidade, além de novos recursos para elevar a execução de tarefas a um novo patamar. Alguns dos principais recursos e mudanças nesta versão são:

- Atualização da versão do Angular 13 para o Angular 16, aumentando a segurança e dando acesso a novos recursos que melhoram o desempenho nas aplicações on-line do cliente.
- Adicione suporte aos recursos de trabalhos cruzados no AS400, com o principal destaque de que os trabalhos podem enviar mensagens de consulta de forma síncrona entre eles, permitindo a dissociação em trabalhos modernizados.
- Melhorias de desempenho no uso do Redis, incluindo otimização do pool de conexões, alta segurança na conexão e mecanismo de bloqueio de conjunto de dados atualizado.

Para obter mais informações sobre as alterações incluídas nesta versão, consulte as seguintes seções:

Runtime versão 3.9.0

zOS

Novos atributos

- Programa de classificação: entradas VSAM atualizadas com tamanho fixo
- JHDB DB: tempo limite configurável adicionado

Melhorias

- Suporte aprimorado para o separador de linha transmitir se usado na concatenação de arquivos
- Suporte aprimorado para abrir arquivos sequenciais concatenados. Inicializar DataSetIndex após a abertura do arquivo
- Suporte aprimorado para separador decimal virtual quando a NumericEditedType é afetado por um valor numérico
- Suporte aprimorado para NumericEditedType valores não negativos
- IDCAMS: Os cartões SYSIN agora são lidos usando a propriedade “encoding” definida em .yml application-utility-pgm
- IDCAMS: gramática atualizada para a compatibilidade com o argumento FILE (..) na instrução DEFINE CLUSTER
- INFUTILB: adicionado suporte ao argumento DFSIGDCB para substituir os parâmetros DCB de DD SYSREC
- INFUTIL: suporte aprimorado para o parâmetro “DFSIGDCB YES”
- SPLICE aprimorado para lidar com um grande arquivo de entrada
- DFSORT: melhor tratamento dos campos de observação
- DFSORT: adicionado suporte para o formato numérico de formato livre (assinado/não assinado) (SFF/UFF)
- SORT: adicionado suporte de análise para as instruções OPTION PRINT e OPTION ROUTE
- SORT/ICEMAN: adicionado suporte a operações de divisão incluídas (campo com operador DIV)
- Suporte aprimorado para CICS READ usando uma chave genérica
- Função StringUtils .chargraphic corrigida para remover SOSI de um tipo gráfico
- Melhore o desempenho ativado DataUtils. isDoubleByteCodificação
- JCL: suporte aprimorado para o modo de disposição KEEP para um conjunto de dados temporário. O sistema muda a disposição para PASS
- JCL: manipula parâmetros DCB dinamicamente
- JCL: saídas aprimoradas de SUM FIELDS para valores incorretos
- JCL: CommonDDUtils::getContent agora procura o recordSize no catálogo
- JCL: leia os atributos rdw/recordSize do catálogo na criação do conjunto de dados
- JCL: adicionado suporte a DCB=.MYDD para copiar parâmetros DCB de um DD para outro na mesma etapa do trabalho
- JCL: aprimorado o sistema de herança de tamanho de registro

- JCL: Adicionado bloqueio de conjunto de dados exclusivo (Redis)
- Redis: adicionado suporte a SSL para o modo autônomo
- Redis: adicionada a contagem sincronizada de bloqueios do Redis com bloqueio
- Redis: parâmetros de pool compatíveis para o bloqueio do Redis
- Redis: atualização otimizada de metadados com o Redis
- Redis: suporte aprimorado ao cluster do redis
- Melhoria nos bloqueios abertos com o modo do IO
- Desempenho aprimorado dos bloqueios de conjuntos de dados e eliminação de bloqueios não utilizados
- Caminho aprimorado do conjunto de dados durante o cancelamento do registro do arquivo
- Invalidação aprimorada do cache da janela de pré-busca
- Adicionado suporte para o uso do provedor de fonte de dados do utilitário de thread seguro
- Verificação aprimorada de nulidade do datasetState
- Suporte aprimorado para não reabrir conjuntos de dados já abertos
- Maior robustez para a operação final do trabalho
- Suporte aprimorado para a ordem dos índices para as chaves, permitindo duplicidades
- Suporte aprimorado para ignorar a ordem de serialização da lista
- Adicionado suporte ao recurso de depuração e despejo para ajudar a diagnosticar problemas de ordem dos índices
- Suporte aprimorado para a atualização de metadados
- Suporte aprimorado para leitura em massa Blusam

AS400

Novos atributos

- Cria um registro de contexto da aplicação
- Suporte à palavra-chave DSPF CLRL(NO) Suporte ao monitoramento de bloqueios de registros
- Support para keyed DataQueue
- Suporte a mensagens INQUIRY para trabalhos em lote
- Adicionado suporte ao arquivo de impressora descrito pelo programa para AS400 COBOL

- Manipula o comando RMVJOBSCDE cl
- Melhoria para RUNSQL/DLYJOB
- CHKOBJ: aumento do código de erro herdado para o parâmetro LIB
- SNDPGMMSG: suporte a parâmetros de string
- RTVDTAARA: Substring aprimorada no LDA
- DSPFD: suporte adicionado ao parâmetro FILE para o nome de arquivo específico
- RUNQRY: suporte ao arquivo sql no QRY PARAM
- CRTDUPOB: suporte à cópia de dados entre áreas de dados
- SBMJOB: converte instruções para uso JobQueueManager
- OPNQRYF: suporte adicionado para a biblioteca Qtemp
- CRTDUPOBJ: Lógica aprimorada para copiar o conteúdo da partição
- CRTDUPOBJ: suporte adicionado a Qtemp para visualizações
- RTVSYSVAL: suporte ao valor SYSVAL, QDATFMT no comando CL
- CHKOBJ: suporte adicionado a OUTQ
- RTVJOBA: suporte ao parâmetro SWS
- SNDPGMMSG e RCVMSG: parâmetros adicionais com suporte a MSGF, MSGFLIB, MSGDTA, MSGTYPE, KEYVAR, MSGKEY, MSGID

Melhorias

- Suporte aprimorado para placas de E/S da ESTAÇÃO DE TRABALHO
- Tratamento aprimorado da mensagem definida sobrepondo a mensagem anterior
- Suporte a informações adicionais de mensagens na array-messageline
- Acesso aprimorado ao wrapper da matriz autônoma dentro de EVAL, SortA e figurativos
- Melhorar a limpeza de DAOs quando a aplicação on-line for encerrada
- Adicionado suporte a formatos de data adicionais e melhora no tratamento de entradas de string
- Manipulação aprimorada do CVTDAT do SYSVAL adicionando parâmetros de decodificação e construção da classe auxiliar de valor do sistema a partir do comando CL SbmJob
- O pacote com.netfective.bluage.gapwalk.rt.blu4iv foi removido da verificação de componentes gapwalk-cl-command
- Aprimorado o suporte de mensagens predefinidas à API de fila de mensagens

- Melhorou o suporte retrieveSubfileRecord para registro escrito em outro programa
- Aprimorado o suporte de mensagens imediatas à API de fila de mensagens
- Tratamento aprimorado da área de dados locais ao enviar um trabalho
- Inicia JobQueues automaticamente quando o servidor é iniciado
- Usa a configuração applicationContext para decodificar parâmetros para SBMJOB
- Melhoria nas mensagens de erro fornecidas pelo sistema
- Permite que RTVMSG pesquise arquivos .properties em subdiretórios aninhados
- Lida com a redefinição de entidades vinculadas a ponteiros inválidos
- Melhorado MessageHandlingBuilder para exibir msgID e MsgFile nome como strings para RCVMSG
- Método de withMsgFile nome aprimorado da API de enfileiramento de mensagens
- Aprimorado o mecanismo de bloqueio da área de dados
- RTVMBRD: suporte a letras maiúsculas e minúsculas para o parâmetro FILE
- CRTDUPOBJ: melhoria no tratamento das visualizações
- CPYTOSTMF: melhoria no tratamento da conexão
- CPYF: melhoria no tratamento do nome do diretório ao copiar de um arquivo simples
- RCVF: lida adequadamente com os parâmetros DEV/RCDFMT e com a transformação de RCDFMT para groovy e java
- RCVF: lida com chamadas subsequentes e evita redefinir o cursor
- CPYF: adicionado suporte à gravação a partir de arquivos simples
- CRTDUPOBJ: adicionado o tratamento de novos obj com a biblioteca Qtemp
- CHGDTAARA: aumento do tamanho máximo da área de dados de 256 para 2.000
- SAVOBJ: certifique-se de que os registros salvos estejam na ordem de inserção
- RTVDTAARA: valores recuperados (não devem ser cortados)
- CHKOBJ: retorna as mensagens corretas do monitor quando o membro não existe
- RTVDTAARA: adicionado suporte à substring LDA
- RTVDTAARA: retorna espaços em branco até o tamanho da variável especificada no parâmetro RTNVAR
- RTVDTAARA: suporte a parâmetros inteiros para início e tamanho e suporte ao formato de transformação mais recente

- CHGDTAARA: adicionado suporte a parâmetros que incluem limites inferiores e superiores
- CHKOBJ: lida com o valor VIEW para o tipo de objeto do parâmetro
- CHKOBJ: resultado definido como verdadeiro, independentemente do membro se a visualização existir

Recursos transversais

Novos atributos

- Lida com a geração de relatórios para arquivos .txt
- Adicionada a propriedade da fonte de dados currentSchema XA ao gerenciador de segredos
- Adicione a propriedade YAML database.cursor.raise.already.opened.error para permitir que a estrutura gere o erro SQLCODE 502 quando o cursor já aberto estiver sendo aberto

Melhorias

- Adicionamos gapwalk poms ao AWS Blu Age na embalagem do Amazon EC2
- Usa o novo paradigma do manipulador de sinais por padrão
- Adicionar suporte ao bloqueio quando a disposição for MOD ou OLD
- Cache adicionado para armazenar padrões de data e hora do banco de dados
- Função de verificação aprimorada do PackedType
- Melhore as DataUtils funções.setTo para registros com VariableSizeArray
- Lida com a opção MQ SYNCPOINT em relação à unidade de execução
- Framework habilitada para definir SQLCODE na transação de reversão
- Adicionado nome automático da classe do driver de acordo com o segredo da chave do mecanismo
- Tempo limite do programa/da transação
- Restaurar a posição do cursor após a reversão ao acessar o cursor

Terceiro

- Atualize o SnakeyAML, o Redisson e o Amazon SDK, YamlBeans remova (reduza CVE-2022-25857, CVE-2023-24621, CVE-2023-42809, CVE-2023-44487)

Ferramentas de modernização versão 3.9.0

zOS

Melhorias

- Suporte aprimorado para XML-TEXT como origem para destino do tipo String
- Fluxo de trabalho aprimorado de STM para UML para oferecer suporte ao padrão de divisão X/(Y/Z)
- JHDB DB: aceita a chamada ROLLBACK antes de qualquer atualização do banco de dados
- JHDB DB: aceita ROLLBACK mesmo se a transação for encerrada (NOP)
- JCL: aprimorada a função de validação de etapas
- SORT: manipula a função SUM com valores decimais negativos de zona
- COBOL: adiciona suporte a escape de aspas simples/duplas em literais de string

AS400

Melhorias

- Função integrada %editc aprimorada para lidar com o código de edição X adicionando zeros à esquerda
- Aprimorada a manipulação do valor inicial dos campos somente de entrada
- Adicionadas teclas de ação para ajudar os diálogos
- Registro de rodapé da tabela dinâmica que aparece na parte inferior
- Comando START manipulado sem KEY PHASE para arquivos que especificam uma RECORD-KEY real
- Valor padrão adicionado para os tipos float e NumberUtils: :pow
- Adicionado suporte à definição de uma variável usando LIKE(IN)
- Manipulação de loop FOR atualizada para oferecer suporte à omissão de elementos opcionais
- Atualizada a análise de RPG para associar registros ao nome da matriz CTDATA
- Tratamento aprimorado de indicadores para instruções CABxx
- Suporte a parâmetros opcionais na palavra-chave COMMIT
- Suporte aprimorado para palavras-chave FORMAT no LF

- Código de operação LOOKUP gerenciado com indicadores altos e iguais (ou baixos e iguais)
- Manipulação no nome da chave PF declarado entre aspas duplas
- Tratamento aprimorado de EDTCDE X para não suprimir os zeros iniciais
- Suporte aprimorado para MSGCON no arquivo da impressora que não gera etiquetas sem nome
- O campo CONTEÚDO é compartilhado por várias estruturas de dados
- Parâmetro ERRSFL tratado em combinação com SFLMSG/SFLMSGID
- Código principal aprimorado antes do escopo da declaração de proc do rpg gratuito completo
- Adicionada a especificação de controle condicionado de análise
- Suporte aprimorado para o método setErrSfl () no dataholdermapper
- Aprimorada a resolução de tipo para variáveis criadas internamente
- Suporte aprimorado para o código de operação Z-ADD
- Tratamento aprimorado do campo constante com valor de DFT
- Melhorar o suporte ao campo inteiro dentro do status ds do programa
- Atribuição de indicadores tratada nos parâmetros ENTRY
- Melhoria no filtro de palavras-chave propagadas por meio da palavra-chave ref/reffield
- Estrutura de DataArea dados sem nome suportada
- Tratamento aprimorado do tipo de dados do ponteiro
- Tratados os elementos da matriz usados para definir variáveis com acesso à matriz de suporte à palavra-chave LIKE, no campo de saída
- Suporte aprimorado para números assinados, exibindo somente dígitos
- Suporte para a relação lógica na placa O
- Caso de teste para %CHAR em alfanumérico
- Suporte à palavra-chave principal da especificação de controle
- EDTCDE com dois parâmetros no arquivo da impressora
- Análise aprimorada FullFree de RPG
- Aprimorada a tabela dinâmica para garantir que o rodapé seja posicionado corretamente
- Adicionado suporte para inicializar tipos numéricos com a constante figurativa TODOS
- Tratamento aprimorado de vários arquivos lógicos do RPG referenciando o mesmo arquivo físico
- Melhorar a detecção de campos modificados em uma tela moderna
- Sincronização de modal com campos dinâmicos

- Tratamento aprimorado do campo numérico assinado somente de saída
- Melhorar as placas de E/S da ESTAÇÃO DE TRABALHO

Recursos transversais

Novos atributos

- Ferramenta de migração de dados: propriedade ebcdicFilesWith VarcharIn VB adicionada para permitir levar em consideração o comprimento de 2 bytes do VARCHAR ao ler bytes
- Implementado uma API comum para registrar erros
- Implementação BluAgeErrorDictionaryUtils e uso de API comum para registrar erros e/ou informações em Cobol2Model, CycleBuilder RPG, Definitions2Model e FieldsProcessor
- Aprimorada a gramática SQL para oferecer suporte a diferentes definições de cláusulas de isolamento

Melhorias

- Atualizada a versão do Angular para v16
- Angular: atualizada a versão ajv de 6 para 8.9

Terceiro

- Atualizada a versão do Groovy para 2.4.15

Notas de lançamento 3.8.0

Esta versão do AWS Blu Age Runtime and Modernization Tools está focada em vários aprimoramentos transversais em todo o produto para melhorar sua qualidade e segurança, além de melhorias no desempenho do armazenamento em cache e na unificação dos suportes de comandos em uma única distribuição. Alguns dos principais recursos e mudanças nesta versão são:

- Atualização da versão do Spring 2.5 para o Spring 2.7, aumentando o suporte de manutenção, o desempenho e a segurança da plataforma.
- Unificação do suporte de mais de 82 comandos CL como parte da over-the-counter distribuição para facilitar o uso e a implantação de aplicativos modernizados que antes usavam scripts CL.

- Novas APIs disponíveis para operar e interagir melhor com os conjuntos de dados BluSam, como importação integrada para o serviço gerenciado e a capacidade de listar informações de metadados do conjunto de dados.
- Melhorias de desempenho e extensão do uso do Redis, incluindo disponibilidade no modo de cluster, recuperação de dados de alta disponibilidade e padronização do uso de segredos.

Para obter mais informações sobre as alterações incluídas nesta versão, consulte as seguintes seções:

Runtime versão 3.8.0

zOS

Novos atributos

- Manipulando a definição da chave como uma string para DynamicFileBuilder
- DFSORT: Adicionado suporte para vários itens na inicialização gramatical OUTFIL TRAILER1 + DFSORT
- Ferramenta CommonDDUtils: tratamento do tamanho do registro em dados in-stream
- Arquivo indexado: manipulando a opção GENKEY

Melhorias

- Serviços de carregamento BluSAM externalizados em uma jarra separada
- Adicionado suporte à configuração do local para armazenar arquivos temporários
- Mecanismos aprimorados de cache compartilhado para casos de vários nós
- Uso de cache compartilhado: IDCAMS verifica a otimização
- Melhorar a injeção de ROWID para seleção incorporada
- JCL: cada procedimento de trabalho in-stream agora é gerado em um arquivo groovy distinto
- Garanta card-demo-v 2 coberturas nos cartões IDCAMS JCL
- BluSAM: evite o warmUp duplicado ao usar várias instâncias
- Diminuição do consumo de memória na hidratação do cache
- Suporte de configuração do pool Jedis
- Separador de linha adicionado para transmitir se usado na concatenação de arquivos

- Suporte para cartões EBCDIC + blocos de comentários (/*.../) no utilitário IDCAMS
- Consulta de suporte ao banco de dados: suporte para cadeias de bytes duplos na conversão do level49 em SQL
- Gramática DFSORT: implementa 17 declarações de controle + integração de 2 delas (OMIT/INCLUDE)
- Melhorar as colunas GRÁFICAS fetch INFUTILB
- Suporte para leitura de arquivo com tabela de tamanho variável
- Support for ZonedType with nibble signed, onde o primeiro bit do último byte é 'E'
- DFSORT/ICETOOL adiciona suporte ao argumento NOMATCH =(..) se um registro não corresponder a nenhuma das constantes de busca CHANGE
- Compatibilidade com o Redis Cluster
- Tratamento do Status do Job (Falha) com base no código de saída do Groovy
- Suporte aprimorado ao CICS SYNCPOINT ROLLBACK.
- Janela de pré-busca para otimizar o uso do cache do Redis
- JCL/GROOVY: herda a propriedade isRDW do conjunto de dados da etapa anterior quando DISP=(, PASS)
- Manipulação de cópia parcial de dados com matriz de tamanho variável

AS400

Novos atributos

- Suporte para placas de E/S para arquivos de exibição
- Suporte para informações adicionais de mensagens para as palavras-chave DSPF ERRMSGID e CHKMSGID
- Suporte para várias mensagens de erro na tela de front-end
- Suporte adicionado ou aprimorado de 82 comandos CL no gapwalk-cl-command aplicativo

Melhorias

- Suporte aprimorado para DELETE e READ sob controle de compromisso
- ConvertDate dentro do %dec embutido
- Cabeçalhos de segurança XSS aplicados

- Maior robustez e consistência da geração de STM (melhor manuseio de: linha de continuação em RPG de formato livre, vírgulas para parte decimal, blocos de formato livre na definição/declaração)
- DataHolderMapper Geração aprimorada
- Robustez adicionada e mudança de escopo em DataAreaFactory
- Melhorou a mudança de foco na tecla tab
- Melhor desempenho na geração de relatórios do Jasper
- Tela decimal aprimorada com preenchimento 0s
- Suporte aprimorado para o campo ROW/COL no INFDS
- Melhorar o suporte para campos modificados na tela
- Foram adicionados getters para nome e caminho do relatório gerado
- Melhorado no comprimento da fila de dados
- Configuração automática aprimorada de Job Queues para atender aos novos padrões no Spring Boot 2.7
- Atualizações aprimoradas da estação de trabalho para várias sessões simultâneas

Recursos transversais

Novos atributos

- Suporte para nenhuma tolerância de dados inválida para pacotes
- Paginação/filtragem adicionada para listar os endpoints do conjunto de dados

Melhorias

- Estratégia aprimorada de transformação de consultas ORACLE na comparação de colunas com uma string vazia
- Manipulando BLOB DB2 com programas utilitários DSNTEP e INFUTILB. O BLOB DB2 agora está modernizado para postgres do tipo BYTEA.
- Melhoria da exclusão do último item do cursor
- Suporte aprimorado para excluir arquivos RRDS
- Melhor desempenho secreto do AWS Blusam
- Manipulação aprimorada de conexões de banco de dados na estrutura SQL
- Chaves padronizadas do gerenciador AWS secreto de várias fontes de dados

- Correções de regressão de desempenho
- Função de verificação aprimorada para PackedType
- Melhor manuseio de LOW-VALUE para PackedType
- Pacote de segurança Spring atualizado para conexão cognito
- Não aplicar codificação e decodificação de codeshiftpoint em bancos de dados de destino do DB2

Terceiro

- Atualização do Spring Boot de 2.5 para 2.7

Ferramentas de modernização versão 3.8.0

zOS

Novos atributos

- JCL: Manipulação de fluxo com retorno de carro “\ r”

Melhorias

- Registro aprimorado para evitar a divisão por zero ao modernizar uma cláusula DIVIDE com ON SIZE ERROR
- JCL: suporte aprimorado para chamar um procedimento em um procedimento
- Suporte para a palavra-chave OF no comando FORMATTIME CICS quando há campos ambíguos
- JCL: suporte para o caractere Â⌘ em variáveis
- JCL: computação RC com base nas etapas anteriores
- Comparando bytes em vez de strings quando PL1 SUBSTR é usado
- Melhoria da inicialização de matrizes multidimensionais a partir de uma única fonte
- Análise aprimorada do COBOL quando envolve uma única consulta SQL em um bloco IF

AS400

Novos atributos

- Suporte para instrução IF aninhada em CL

- Suporte aprimorado para a declaração ENDDO em formato livre de RPG

Melhorias

- Suporte aprimorado para nível de controle de condicionamento
- Retorno aprimorado do protótipo com LIKE
- Suporte aprimorado para lidar com funções %months, %year, %days
- Suporte for help feature para toda a tela
- Manipulação de espaços em branco figurativos transmitidos como parâmetro
- Melhoria na expressão EVAL com o operador “”
- Manipulando o comando START sem KEY PHASE
- Melhoria no manuseio da palavra-chave LIKEREC
- Melhoria em subcampos sem nome
- Melhoria no procedimento de devolução de um tipo não assinado
- Suporte aprimorado para a operação RESET (RPG gratuito), integrações de %CHAR e %DEC
- Melhoria na função integrada %LOOKUPXX
- Suporte aprimorado para a palavra-chave LIKEDS no procedimento sem protótipo
- Manipulando o tipo de matriz de palavras-chave Dim (VAR, AUTO)
- Suporte aprimorado para o XFOOT
- COBOL: suporte aprimorado para campos RENAMES
- CL: suporte enquanto condição (verdadeira)
- Melhorou o tratamento de matrizes autônomas com a palavra-chave LIKE
- Melhoria da função incorporada %INT
- Análise de RPG totalmente gratuita aprimorada
- Suporte aprimorado para matriz na ligação
- CL2GROOVY: Declaração de seleção de suporte
- Melhoria na palavra-chave DSPF “ERRMSGID”
- Melhorou o tratamento da inicialização de bytes com zeros à esquerda
- Melhoria nos authorizedValues para campos numéricos
- Manipulando o extensor H para declaração EVAL de formato livre

- CL para Groovy: suporte a substring de LDA
- Suporte aprimorado para RESET em um registro
- Melhorou o tratamento de EDTCDE e EDTWRD com referências
- Mapeamento aprimorado do campo de entrada com campos DDS
- Suporte aprimorado para o caractere MOVEA para a matriz IN
- Melhoria no protótipo com a palavra-chave LIKEDS
- Suporte aprimorado para a palavra-chave DSPF de DSPATR
- Análise aprimorada do cartão D com +/-
- Maior robustez nas chamadas de programas
- Maior robustez no processo de resolução de campo

Recursos transversais

Melhorias

- FrontEnd: Simule o evento de colagem para entrada IME

Terceiro

- Atualização do Spring Boot de 2.5 para 2.7

Notas de lançamento 3.7.0

Esta versão do AWS Blu Age Runtime and Modernization Tools inclui principalmente aprimoramentos para oferecer melhor suporte a comandos e utilitários, recursos de integração com o AWS Secrets Manager e novos recursos de monitoramento. Algumas das principais alterações desta versão são:

- Agora, vários componentes de tempo de execução podem usar o AWS Secrets Manager para aumentar a configuração de segurança de aplicativos modernizados, principalmente relacionados a fontes de dados de serviços públicos, Redis para filas TS, BluSam cache e bloqueios.
- Endpoint de monitoramento que permite recuperar métricas de transação, lote e JVM para otimização do uso de recursos e gerenciamento operacional, como status, duração, volume e outros.

- Novos recursos para permitir chamadas do IBM MQ em RPG e maior cobertura de transformação do JCL SORT e IDCAMS.

Para obter mais informações sobre as alterações incluídas nesta versão, consulte as seguintes seções:

Runtime versão 3.7.0

Tópicos

zOS

Novos atributos

- Melhorar as consultas de análise envolvidas na aplicação utilitário do programa usando SQL como gramática. (V7-9401)
- Manipule a matriz de tamanho variável indexada quando deslocada (V7-9904)
- Suporte a coluna INSERT SQL TIME no DB2 com formato de 24:00:00 horas (V7-10023)
- Suporte a consulta INSERT SQL de matrizes com as opções FOR ROWS e ATOMIC (V7-10105)
- JCL SORT - aprimorado TranscodeTool para suportar OUTREC com IFTHEN (V7-10124)
- JCL SORT: adicione suporte para a palavra-chave DATE no comando OUTREC (V7-10125)
- JCL: adicione suporte aos procedimentos In-Stream (V7-10223)

Melhorias

- Um conjunto de dados marcado com a disposição “PASS” deve estar disponível em todas as etapas do trabalho (V7-9504)
- Suporte JCL atributo SCHENV (V7-9570)
- Suporte SEND com opção CTLCHAR (V7-9714)
- COBOL: manipule diferentes conjuntos de caracteres separadores de linha em declarações ACCEPT (V7-9875)

- Evite reversões múltiplas (V7-9958)
- Permitir o uso da disposição MOD para anexar no final dos arquivos GDG (V7-10031)
- Otimização: refatoração putAll (V7-10063)
- PutAll refatoração: adição de paginação (V7-10063)
- Torne o tempo limite de leitura do cliente Jedis configurável (V7-10063)
- UseSsl suporte para o modo autônomo (V7-10114)
- Suporte EIBDS após abrir o arquivo com sucesso (V7-10147)
- Suporte EIBDS após uma solicitação de controle de arquivos (V7-10147)
- Melhorar o suporte ao CICS SYNCPOINT (V7-10187)
- BluesamRedisSerializer: problema com a persistência de metadados (V7-10202)
- Suporte Redis AWS Secrets Manager para filas TS (V7-10204)
- Suporte JCLBCICS na personalização do tamanho do nome DD (V7-10224)
- Adiciona suporte para caminho absoluto na instrução IDCAMS DELETE (V7-10308)

AS400

Novos atributos

- Implementação do recurso de ajuda para telas AS400 (V7-9673)

Melhorias

- Número de registros no INFDS (V7-9377)

Recursos transversais

Novos atributos

- Support for Runtime no EC2 para enviar registros para a Amazon CloudWatch (D87990246)
- Novo endpoint adicionado para recuperar métricas sobre lotes, transações e JVM (D88393832)

Melhorias

- Suporte: fontes de dados do AWS Secrets Manager para utilitários pgm (V7-9570)

- Foi adicionado suporte ao Db2 para DSNUTILB DISCARD (V7-9798)
- Suporte para gravação no registrador em vez do fluxo de saída padrão do sistema nos arquivos SYSPRINT e SYSPUNCH padrão (V7-10098)
- Support BluSam Redis cache e bloqueia propriedades de conexão no AWS Secrets Manager (V7-10238)
- Suporte para conexão SSL no Db2 XA AWS secret (V7-10258)
- Metadados atualizados para IDCAMS REPRO e VERIFY (V7-10281)
- Gerenciamento aprimorado do código de retorno IDCAMS Abend (V7-10307)

Ferramentas de modernização versão 3.7.0

Tópicos

zOS

Novos atributos

- PLI: atribuição aprimorada para seção transversal de matrizes e matrizes bidimensionais (V7-9830)

AS400

Novos atributos

- Manipulação de indicadores de nível de controle (V7-9227)
- Suporte para o parâmetro EXTNAME *INPUT (V7-9897)
- Reescrita aprimorada do Goto: Suporte para tags localizadas em instruções SELECT OTHER (V7-9973)
- Suporte a palavra-chave REFSHIT DSPF (V7-10049)

Melhorias

- Melhoria no tratamento da palavra-chave de descrição de arquivo EXTIND (*INUx) (V7-7404)
- Transformação aprimorada de arquivos SQLDDS (V7-7687)
- Objetos de arquivo não são mais gerados para arquivos AS400 (V7-9062)
- Tratamento aprimorado da palavra-chave de descrição de arquivo EXTDESC (V7-9268)
- Manipulação aprimorada do %CHAR embutido (V7-9311)
- Suporte aprimorado para pagedown no último registro sem SFLEND (V7-9322)
- Suporte aprimorado para estruturas de dados prefixadas (V7-9436)
- Suporte para dimensão definida com %SIZE (V7-9472)
- Suporte para lidar com o nome do campo PF declarado entre aspas duplas (V7-9557)
- Operação de arquivo aprimorada: não diferencia maiúsculas de minúsculas (V7-9785)
- Suporte para campo inicializado para *USER (V7-9806)
- Suporte para o tipo COMP no AS400 (V7-9840)
- Análise aprimorada do COBOL400 em (Não) (V7-9922) InvalidKey
- Tratamento aprimorado da operação SCAN (V7-9971)
- Suporte aprimorado do código de operação GOTO (V7-9973)
- Manipulação aprimorada da operação EXCEPT (V7-9977)
- Suporte aprimorado a prefixos (V7-10000)
- Suporte para chamadas MQ em RPG (V7-10007)
- %LOOKUP integrado aprimorado (estrutura de dados de matriz com chave) (V7-10022)
- Suporte para operação Close *All (V7-10036)
- Suporte para a instrução SQLDDS UPDATE AS ROW CHANGE (V7-10051)
- Melhoria para lidar com o tipo de valor literal Long (V7-10073)
- Gramática RPG aprimorada (o uso da palavra-chave INZ como nome da sub-rotina) (V7-10074)
- Gramática RPG aprimorada para suportar valores numéricos com parte fracionária vazia (V7-10077)
- Suporte aprimorado para campos compartilhados entre CL e arquivo externo (V7-10081)
- Suporte aprimorado para indicadores condicionais do DDS (V7-10084)
- Suporte para o tipo binário DDS com programas COBOL (V7-10100)
- Melhor colisão de nomes com ligação (V7-10109)
- Suporte para misturar procedimentos principais e de exportação (V7-10112)
- Suporte aprimorado para DataStructure em um subprocedimento (V7-10113)

- Suporte aprimorado do CLEAR (V7-10126)
- Suporte aprimorado do loop DO (V7-10134)
- Suporte SQLTYPE em RPG totalmente gratuito (V7-10151)
- Análise aprimorada das condições na palavra-chave DDS (V7-10155)
- Geração DSL aprimorada (V7-10163)
- Melhoria para processIndicators quando a condição é uma expressão binária. (V7-10164)
- GOTOs aprimorado com condição Else (V7-10168)
- Suporte para o tipo Time and Timestamp no DSPF (V7-10173)
- Análise aprimorada da linha de continuação para DDS (V7-10183)
- Suporte COBOL para RENAMES FLD OF RECORD (V7-10195)
- Análise aprimorada de indicadores condicionais em campos DSPF (V7-10221)
- Suporte a análise da palavra-chave DDS NOALTSEQ (V7-10288)
- Menu Support Help e campos ocultos (V7-10314)
- Verificação aprimorada da integridade das palavras-chave de ajuda do DSPF (V7-10328)
- Não está mais propagando todas as palavras-chave no campo Ref (V7-10347)

Recursos transversais

Novos atributos

- Migrador de dados — Tratamento de dados CLOB (V7-9665)

Melhorias

- Propagando a propriedade JCL SCHENV da definição JOB para PROC GROOVY por meio de (V7-10225) JobContext
- FrontEnd - Ajustar o tamanho da janela em caso de ausência de borda (V7-10358)

Notas de lançamento 3.6.0

Esta versão do AWS Blu Age Runtime and Modernization Tools fornece novos recursos para migrações legadas do zOS e do AS400, principalmente orientadas para expandir os mecanismos de suporte do CICS, complementar os recursos da JCL, otimizar o desempenho em recursos

simultâneos e de alto volume e adicionar recursos. multi-data-source Algumas das principais alterações desta versão são:

- Aprimoramento do tratamento dinâmico de arquivos da JCL, expansão das instruções atuais e gerenciamento de conjuntos de dados concatenados, execução de várias instruções em um único bloco e transferência de dados de lotes para programas.
- Suporte aprimorado de vários comandos do CICS, incluindo a consulta de vários tipos de recursos do CICS.
- A capacidade de ter bancos de dados diferentes ao usar o Blu Age Runtime Utilities, mais adequado para cenários em que os dados comerciais são distribuídos em várias fontes.

Para obter mais informações sobre as alterações incluídas nesta versão, consulte as seguintes seções:

Runtime versão 3.6.0

Tópicos

zOS

Novos atributos

- JCL - DynamicFileBuilder - Gerenciamento aprimorado de manipuladores de arquivos (V7-9408)
- Conversão de formato aprimorada em algumas funções integradas do SQL DB2 ao chamar o utilitário INFUTILB UNLOAD (V7-9554)
- Atribuições de matriz multidimensional PLI aprimoradas (V7-9592)
- Tratamento do redirecionamento do sysout para o arquivo (V7-9992)

Melhorias

- Adicionar acionamento de procedimentos armazenados para DB2 RDBMS (V7-9155)
- SORT manipula a conversão para o formato PDF (V7-9286)

- JCL/GROOVY: melhorar a instrução REPRO para suportar conjuntos de dados DUMMY (V7-9424)
- Melhorar o suporte ao CICS UNLOCK (V7-9606)
- Manipule o tamanho do valor padrão para Union (V7-9648)
- O JCL/GROOVY manipula diferentes terminações/disposições em conjuntos de dados concatenados (V7-9653)
- Torne o PageSize configurável para conjuntos de dados Blusam (V7-9680)
- DSNUTIL: permite o carregamento de 24:00:00 como HORA válida no DB2LUW (V7-9697)
- Support a comparação de HIGH-VALUES (0xff) em NumberUtils .ne () NumberUtils /.eq () (V7-9731)
- JCL/GROOVY: suporte DO... Palavras-chave THEN nas cláusulas IF-THEN-ELSE do IDCAMS para executar várias instruções em um único bloco (V7-9750)
- JHDB inválido chamado programa fora do JHDB (V7-9782BatchRunner)
- Suporte a caracteres de espaço em branco no cartão de controle SORT OUTFIL (V7-9808)
- Melhorar o suporte do CICS READ PREV (V7-9845)
- Melhorar o acesso simultâneo aos índices do conjunto de dados (V7-9864)
- Melhorar o suporte ao CICS REWRITE (V7-9873)
- COBOL - suporte para SYSIN de várias linhas em instruções ACCEPT para passar dados do lote (JCL) para um programa (COBOL) (V7-9875)
- Groovy - Melhor manuseio da etapa de criação ConcatenatedFileConfiguration de arquivos (V7-9876)
- IDCAMS UTILITY: tratamento da instrução DEFINE PATH (V7-9878)
- SORT BUILD: ajuste a opção TRAN e manipule espaços em branco implícitos (V7-9925)
- Melhorar o CICS DELETE com suporte à opção GENERIC (V7-9939)
- Melhorar o suporte ao CICS STARTBR e ENDBR (V7-9952)
- Melhorar o desempenho próximo no acesso simultâneo (V7-9953)
- Melhorar o tratamento do status do arquivo na inicialização (V7-9991)
- Groovy - Permitir a chamada de getDisposition ()/()/getNormalTermination() em (getAbnormalTerminationV7-10012) ConcatenatedFileConfiguration

AS400

Novos atributos

- Suporte a indicadores externos em palavras-chave COMMIT (V7-6035)
- Redefinir o loop ReadC após a gravação SFLCTL (V7-8061)
- Suporte ao indicador LR em CALL (V7-9250)
- Adicione um novo tipo de campo dinâmico (dividido) para lidar com o campo de entrada em várias linhas (V7-9370)
- Suporte ao arquivo primário/secundário (V7-9390)
- As áreas de dados locais agora são passadas para o trabalho chamado ao enviar um trabalho (V7-9775)
- Suporte do QTEMP para área de dados e suporte à criação de valor da área de dados. (V7-9916)
- Controle de compromisso: suporte para ativar/desativar o controle de compromisso (V7-9956)
- Suporte indicadores externos em palavras-chave COMMIT

Melhorias

- Melhorar a exibição do valor 0 e o EDTWRD (V7-8933)
- Suporte da palavra-chave DSPF “CHKMSGID” (V7-9125)
- Transação de confirmação de SQL após o encerramento do lote (V7-9232)
- Melhorar o suporte das palavras-chave EXPORT e IMPORT para campo e estrutura de dados (V7-9265)
- Support em letras minúsculas DateHelper (V7-9461)
- Suporte a conversão de*CYMD para *ISO (numérico) (V7-9488)
- Melhorar o identificador do %len embutido para um campo variável (lado esquerdo e direito de uma expressão) (V7-9733)
- Melhorar o suporte para funções integradas '%LOOKUPXX' XX (“LE”, "LT”, “GE”, “GT”) (V7-10064)

Recursos transversais

Novos atributos

- CICS: melhorar a transação do Inquire para o status da opção (V7-9712)
- JCL: melhorar a carga do sysprint com o arquivo de saída do sistema (V7-9797)
- CICS: melhorar o INQUIRE TSQUEUE (V7-9823)

- CICS: melhorar o terminal Inquire para a opção ID de usuário (V7-9906)

Melhorias

- Melhorar o controle da comparação com o espaço em branco (V7-8047)
- Melhore o registro para Jics e Blusam (V7-8847)
- Suporte a atributos estendidos BMS SOSI e símbolo programado F8 para campos dinâmicos (V7-8857)
- Lidar com estouro de buffer no parâmetro do programa (V7-9138)
- Melhorar a simultaneidade de gravação de threads para o registro de bloqueios Blusam (V7-9505)
- Suporte a configuração de várias fontes de dados para Utility-PGM (V7-9570)
- Modo somente de bloqueio de nível de registro Blusam (V7-9626)
- Garanta que a persistência dos metadados resista à reinicialização do servidor (V7-9748)
- Melhorar a limpeza do DAO em caso de exceção (fechamento do navegador) (V7-9790)
- Support DummyFile para INFUTILB SYSPUNCH (V7-9799)
- Aprimorar o suporte para valores negativos em NumericEditedType (V7-9935)

Ferramentas de modernização versão 3.6.0

Tópicos

zOS

Novos atributos

- JC: melhorar o registro para o final do procedimento (V7-8509)
- PL1 - Melhore a geração de bolsas para o tipo de dados PakedLong (V7-8917)
- JCL: melhorar o registro para o final do procedimento quando o arquivo contém o marcador “final”// (V7-9509)
- PL1: melhorar o suporte para GET EDIT com fluxo de ponto fixo e SYSIN (V7-9593)

- DB2: melhorar o suporte para o tipo VARGRAPHIC DB2 (V7-9809)
- CICS: melhorar o comando QUERY SECURITY para a opção LOGMESSAGE (V7-9969)
- PL1: melhorar a geração de bolsas para carga/gráfico embutido (V7-9989)

Melhorias

- PL1: melhorar o suporte para a palavra-chave INCLUDEX (V7-9588)
- PL/I: trate a palavra-chave CHARGRAPHIC como um parâmetro válido de qualquer chamada de método (V7-9589)
- Melhorando a resolução da variável host PL1 quando nomeada com caracteres específicos @ # $ §. (V7-9654)
- COBOL: suporte das palavras-chave C01... C12 e S01... S05 como parâmetro da instrução WRITE ADVANCING na etapa de análise (V7-9669)

AS400

Novos atributos

- Suporte a transformação SQL-DDS no Analyzer (V7-7687)
- Automatize a detecção de arquivos SQL-DDS (V7-7687)
- Implementação do pré-processamento SQL-DDS (V7-7687)
- Suporte a palavra-chave ALIGN (V7-9254)
- Support ExtName para DSPF e matriz multi-dim (V7-9663)
- InvalidKey Declarações de suporte sobre COBOL WRITE (V7-9793)

Melhorias

- Melhoria no opcode TESTB (V7-8865)
- Melhorar o suporte do DECFMT em foco (V7-8933)
- Manipulação do indicador resultante no MOVE (V7-9224)
- Melhorar o suporte da palavra-chave TEMPLATE para campo e estrutura de dados (V7-9278)
- Melhoria do LIKEDS (DS definido usando LIKEDS é automaticamente qualificado) (V7-9302)
- COBOL: melhorar a geração da estrutura de indicadores (V7-9423)
- O parâmetro const no protótipo não é somente para leitura (V7-9437)

- Melhorar a palavra-chave EDTCDE com o código de edição “Y” (V7-9443)
- Suporte a geração do campo*ROUTINE em PSDS e INFDS (V7-9487)
- Melhorar o campo de regravação XXX para autônomo (o valor padrão é perdido durante a regravação) (V7-9522)
- Melhorar o suporte de palavras-chave DSPF (V7-9658)
- Manipulando o valor padrão de ZEROES no binário (V7-9666)
- Suporte: ponteiro implícito (V7-9719)
- Melhorar o tratamento da chamada embutida %size com um parâmetro (V7-9730)
- Melhorar o tratamento de referências de estrutura de dados em chamadas integradas (%ELEM) (V7-9736)
- Melhorar o tratamento do comprimento do sinal para o campo com a referência LIKE na especificação de definição (V7-9738)
- Melhoria no REWRITE (V7-9791)
- Melhoria da geração de índices a partir de arquivos DDS (V7-9803)
- Melhorar a robustez dos mapeadores com valor numérico inválido (V7-9813)
- Melhorar a geração de arquivos SQLModel e allIndexes (V7-9818)
- Melhorar o suporte qualificado do DS (V7-9863)
- Melhorar o suporte do LOOKUP (com um campo autônomo COMO um DS no parâmetro) (V7-9961)
- Melhorar o LIKE no indicador (V7-9985)
- Manipulando o indicador resultante no MVR (V7-9995)
- Suporte ao caractere N com tilde (V7-10021)
- Melhorar a geração moderna de arquivos DDL a partir de arquivos antigos SQLDDS (V7-10067)

Recursos transversais

Novos atributos

- Personalize a localização do recurso com uma propriedade yml (D88816105)
- COBOL: suporte da instrução EXIT PERFORM para sair de um PERFORM embutido sem usar um GO TO/PERFORM... THROUGH (V7-9582)
- Especificar a codificação antiga padrão a ser considerada nos metadados globais. (V7-9883)

Melhorias

- Melhorar a geração de máscaras (V7-9602)
- Melhorar o aquecimento do contexto (V7-9621)
- Torne o thread Charset CUSTOM930 seguro. (V7-9674)
- Melhoria no MOVEA (V7-9773)

Notas de lançamento 3.5.0

Esta versão do AWS Blu Age Runtime and Modernization Tools fornece novos recursos para migrações legadas do zOS e do AS400, principalmente orientadas à otimização de conjuntos de dados e mensagens, bem como recursos Java estendidos como um ativo resultante do processo de transformação. Algumas das principais alterações desta versão são:

- Capacidade de migrar programas de CL para Java, além do recurso preexistente de scripts groovy, para facilitar sua integração com outros programas modernizados e para simplificar a curva de aprendizado do cliente unificando a linguagem de programação resultante.
- Redução do tempo e otimização do desempenho das cargas de conjuntos de dados no Redis com o novo recurso de massa de dados.
- Capacidade de operar e transmitir conjuntos de dados dentro das etapas do trabalho para modernizar os comportamentos tradicionais dos conjuntos de dados.
- Extensão da migração de SQL para suportar arquivos de entrada VB e migração simplificada do Java 11.
- Vários novos mecanismos para uma integração mais rápida com o IBM MQ, incluindo cabeçalhos adicionais, suporte estendido a GET/PUT e recuperação automática de metadados da fila.
- Endpoint REST para metadados de conjuntos de dados e importar conjuntos de dados de buckets do S3.

Para obter mais informações sobre as alterações incluídas nesta versão, consulte as seguintes seções:

Runtime versão 3.5.0

Tópicos

- zOS

- AS400
- Recursos transversais

zOS

Novos atributos

- JCL SORT: lidar com a nova sobreposição de palavras-chave (V7-9409)
- ZOS COBOL: melhorar o suporte de caracteres flutuantes (V7-9404)
- Porta do RedisJics TSQueue para RedisTemplate & ListOperations (V7-9212)
- ZOS JCL - aprimora o caminho do diretório temporário com o diretório de arquivos, se definido por meio de UserDefinedParameters (V7-9012)
- Manipule a FUNÇÃO ORD-MAX com ALL (todos os itens da matriz) (V7-9366)
- Chaves prefixadas e legíveis por humanos agora são usadas ao armazenar filas TS no Redis (V7-9212)
- Adicione o endpoint get dataset para a API Blusam
- JCL: suporte ADD para trabalho em lote com nome envolvendo caracteres especiais como # (V7-9136)
- A busca do TSModel agora é executada de forma robusta sob demanda (V7-9212)

Melhorias

- Suporte INCLUDE não versionado em arquivos LNK (V7-6022)
- MQ: suporte aprimorado de codificação (V7-9652)
- Melhorando o suporte para bytes duplos ou conjuntos de caracteres mistos para vários tipos de caracteres (V7-9596)
- JCL: suporte de filesDirectory configurado em IDCAMS delete NONVSAM (V7-9609)
- Suporte do modo em massa para carregamento de conjuntos de dados ESDS e RRDS de arquivos (V7-8639)
- Manipule a abertura de ESDS vazios no modo de entrada. (V7-9287)
- Melhorar a instrução DEFINE CLUSTER com suporte à abreviatura ORD/UNORD (V7-9451)
- Melhorias no desempenho do bloqueio Blusam Redis (V7-8639)

- Melhorar a instrução DEFINE CLUSTER para suportar RECORDSIZE fornecida no escopo do argumento DATA () (V7-9337)
- Adiciona suporte aos atributos BUFFERSPACE/UNIQUE nas instruções DEFINE CLUSTER (V7-9419)
- Melhore a operação de leitura Blusam para um conjunto de dados de registro de comprimento variável. (V7-9391)
- O ENDEREÇO CICS representa corretamente o CWA ausente como nulo (V7-9491)
- Remova a gravação desnecessária nos bloqueios finais (V7-8639)
- Manipular a injeção de modelo de cache Redis no cache (V7-9510)
- Decodifique corretamente o parâmetro BPXWDYN (V7-9417)
- Melhoria no consumo de exportação do LISTCAT (V7-9201)
- Suporte a caracteres não imprimíveis no nome Blusam TS Queues (V7-9212)
- Manipule a criação de mapa de recebimento para campo com mapset null (V7-9486)
- Melhore a operação de BluesamRelativeFile exclusão e regravação para o modo de acesso dinâmico. (V7-8989)

AS400

Novos atributos

- Adicione um recurso para gerar arquivos CL como programas Java por meio do pivô DS/STM padrão (V7-9427)
- Suporte a Input File com o modo ADD (V7-9378)
- Melhorou a ordem de classificação e o gerenciamento de recuperação para suportar o comando cl OPNQRYF (Open Query File) e adicionou suporte ao parâmetro SHARE em. OverrideItem (V7-9364)

Melhorias

- Support SFLNXTCHG em (V7-8061) UpdateSubfile
- Modifique o escopo do contexto CL ao executar o comando CL (V7-9624)
- Manipule o código de retorno do programa BPXWDYN (V7-9417)
- Limpe os monitores locais. (V7-9624)

- Suporte da palavra-chave DSPF RTNCSRLOC (V7-9389)
- setOnGreaterOrEqual() não configurando Igual a 1 (V7-9342)
- Atualizar o cache de campos ativado UpdateSubfileRecord (V7-9376)
- Melhorar o suporte SFLNXTCHG (V7-8061)

Recursos transversais

Novos atributos

- Ignore o prefixo G na sequência gráfica literal. (V7-9420)
- ZOS COBOL: melhorar o suporte de Fiedl.initialize () para algumas estruturas especiais (V7-9485)
- Permitir a inicialização do contexto de forma assíncrona para melhorar o desempenho da inicialização do programa (V7-9446)
- SQL Release explicitamente a instrução de preparação aberta e. ResulSet (V7-9422)
- Melhorar o JMS MQ: suporte MQRFH2 para MQ PUT/V7-7085 — suporte ao gerenciador de filas padrão (V7-9400)
- Gerenciamento de SQL: habilite conversões do Lambda em parâmetros para comandos SET (V7-9492)
- ZOS MQ JMS: adicione suporte ao MQCOMIT e ao MQBACK (V7-9399)
- ZOS IBMMQ: melhorar o suporte ao MQINQ (V7-9544)
- Manipule a operação CONCAT com byte em vez de string ao usar a codificação de byte duplo. (V7-8932)
- ZOS IBMMQ: melhorar o suporte ao comando PUT com as opções SET_ALL_CONTEXT (V7-9544)

Melhorias

- Manipule nomes de arquivos gdg com o caractere $ (V7-9066)
- O Diagnóstico SQL retorna 1 como cláusula NUMBER quando a instrução SQL anterior é bem-sucedida. (V7-9410)
- Esboço para campo com comprimento não nulo (V7-7536)
- Suporte a função PL1 GRAPHIC integrada (V7-9245)
- MQ: adicionar suporte da versão para configuração de campos MQGMO (V7-9500)

- JMS MQ GET: melhoria do dataLength da mensagem retornada (V7-9502)
- Defina sqlerrd (3) com o número de itens buscados no contexto ROWSET. (V7-9371)

Ferramentas de modernização versão 3.5.0

Tópicos

zOS

Novos atributos

- ZOS PLI: suporte asterisk index na atribuição com expressão binária (V7-9178)
- JCL para BatchScript - Um “//” marca o fim da execução do trabalho (V7-9304)
- ZOS PLI: suporte aprimorado para caracteres flutuantes e login em tipo numérico editado (V7-8982)
- COBO: suporte da função SUM integrada (V7-9367)
- JCL: opcionalmente, comente o código morto após a declaração nula (//) (V7-9202)
- JCL: suporte de operador '|' na declaração de condição (V7-9499)
- PL/I: comentário das diretivas de pré-compilação na etapa de pré-processamento para evitar exceções de análise (V7-9507)

Melhorias

- Manipule a definição de fluxo com delimitador (V7-9615)
- Melhorando o tratamento das exportações do LISTCAT. (V7-9201)
- PL/I: aprimoramento para suportar argumentos 'nulos' implícitos (V7-9204)

AS400

Novos atributos

- Suporte da palavra-chave DDS CONCAT (V7-9439)
- Refatore o código java gerado para palavras-chave DSPF. (V7-7700)
- Suporte à palavra-chave Varying em campos dentro de uma definição de estrutura de dados (V7-9029)

Melhorias

- Melhorar a análise do relacionamento lógico E/OU (V7-9352)
- COBOL Melhorar o mapeamento entre vo e dsEntity (V7-9449)
- Exibir valor vazio se a entrada numérica estiver focada (V7-9374)
- Variável local no SQL Declare Cursor (V7-9456)
- Problema de escopo com DS vazio (V7-9466)
- Truncar linhas após a coluna 80 antes da análise (V7-9632)
- Melhorar o tratamento de referências de campo e chamadas integradas em palavras-chave (DIM, LIKE,...) na especificação de definição (V7-9358)
- Suporte a comentários SQL (--) (V7-9632)
- FullFree análise, digite Data/Hora/Timestamp (V7-9542)
- Incluir SQLCA da FullFree análise (V7-9333)
- Melhorar o Support of Control Level. (V7-9610)
- Lide com a comparação de DS com *BLANKS (V7-9668)
- Melhorar o suporte de vários indicadores no DDS (V7-9318)
- Melhoria do suporte de vários programas DSPF (V7-9657)
- Melhorar a manipulação do campo com LIKE (caso de estrutura de dados curtida e caso de estrutura de dados curtida em uma matriz) (V7-9213)
- RPG grátis, continuação de Handle no literal (V7-9686)
- Melhorar o suporte de registros de fim de programa (V7-9452)
- Suporte da frase LINKAGE na declaração CALL. (V7-9685)
- Código de operação CASXX (CASBB sem grupo CASXX) (V7-9357)
- Melhore a análise FullFree de RPG (V7-9457)
- %LEN integrado não suporta DS como argumento (V7-9267)
- Melhorias do MOVEA quando o fator 2 é *ALL'X... ' (V7-9228)

- Atribuição de suporte com campo RENAME (V7-9385)

Recursos transversais

Novos atributos

- Ferramenta SQL Migrator: adicione a opção OID para tamanho de registro variável na etapa de carregamento ebcdic. (V7-9380)
- Ferramenta SQL Migrator: suporte para Java 11 na opção OID (V7-9599)

Melhorias

- Melhorar o suporte para matrizes aninhadas (V7-9595)
- Substitua o caractere Â¬ por! no caso de Â¬ é suportado pela codificação original. (V7-9465)
- JCL: suporte de PASS normal termination para compartilhar conjuntos de dados entre as etapas do trabalho (V7-9504)
- Aplique ON NULL à definição de coluna no ORACLE ao lidar com VARCHAR e tipo de coluna db anulável. (V7-9681)
- Melhorar a conformidade com a injeção de molas (V7-9635)

Instruções de atualização para o AWS Blu Age

Esta página contém instruções para atualizar a versão AWS Blu Age.

Migrando da versão 3.10.0 para a 4.0.0

A principal mudança na versão 4.0.0 é a migração do Spring Boot 2.7 para o Spring Boot 3.2 e do Tomcat 9 para o Tomcat 10.

Alterações no código

Esta seção lista as alterações necessárias para tornar o código modernizado compatível com o AWS Blu Age Runtime 4.0.0. Você pode pular esta seção se decidir lançar uma nova geração usando a versão 4.0.0 no Blu Insights (Centro de Transformação).

Mudanças no POM

Grupo	ArtifactId	Alteração
org.slf4j	slf4j-api	Remover (é uma dependência transitiva)
org.yaml	coelho de cobra	Remover (é uma dependência transitiva)
org.springframework.boot	spring-boot-starter-web	- Atualize spring.boot.versio n para 3.2.4 - Remova a exclusão do log4 4j j-to-slf
org.springframework.boot	spring-boot-starter-jta-atômico	Mude para com.atomikos: 3-starter:6.0.0 transactions-sprin g-boot
org.apache.commons	commons-dbcp2	Atualize para 2.10.0
org.postgresql	postgreql	Atualize para 42.7.2
com.microsoft.sqlserver	mssql-jdbc	Atualize para 12.4.2.jre11
com.oracle.database.jdbc	ojdbc8	Mude para ojdbc11 versão 23.3.0.23.09

Migre de Javax para Jacarta

A atualização do tomcat vem com uma migração do pacote Java Javax para Jacarta. Certifique-se de atualizar suas importações adequadamente de javax.* para jakarta. *.

Quase todas as classes antigas referenciadas no pacote Javax podem ser encontradas em Jacarta. As exceções conhecidas a isso são os `javax.xml` pacotes `javax.sql` e, que ainda não foram alterados.

Mudança de Atomikos

Devido à alteração de dependência mencionada acima, as referências a `org.springframework.boot.jta.atomikos.AtomikosDataSourceBean` devem ser alteradas para. `com.atomikos.spring.AtomikosDataSourceBean`

Remoção do dialeto do PostgreSQL

A classe personalizada `PostgreSQLDialect.java` é removida. As referências a ele no lançador principal também devem ser removidas.

Implantação (AWS Blu Age Runtime (não gerenciado))

Tomcat

Esta versão é compatível com o Tomcat`10.1.17`. É necessário atualizar o servidor Tomcat para essa versão para executar o Blu Age Runtime. `4.0.0` Certifique-se de portar as alterações de configuração antigas (principalmente as propriedades do Catalina).

Dependências compartilhadas

A pasta compartilhada em tempo de execução contém as up-to-date dependências.

Dependências extras

Se você usou dependências extras (não incluídas no tempo de execução), talvez seja necessário atualizá-las. O arquivo readme na pasta extra lista as versões suportadas.

AWS Conceitos de tempo de execução do Blu Age

Compreender os conceitos básicos do AWS Blu Age Runtime pode ajudá-lo a entender como seus aplicativos são modernizados com a refatoração automatizada.

Tópicos

- AWS Arquitetura de alto nível do Blu Age Runtime
- AWS Estrutura Blu Age de um aplicativo modernizado
- Simplificador de dados

AWS Arquitetura de alto nível do Blu Age Runtime

Como parte da solução AWS Blu Age para modernizar programas legados para Java, o AWS Blu Age Runtime fornece um ponto de entrada unificado baseado em REST para aplicativos modernizados e uma estrutura de execução para esses aplicativos, por meio de bibliotecas que fornecem construções legadas e uma padronização da organização do código dos programas.

Esses aplicativos modernizados são o resultado do processo AWS Blu Age Automated Refactor para modernizar programas de mainframe e midrange (referidos no documento a seguir como “legados”) para uma arquitetura baseada na web.

As metas do AWS Blu Age Runtime são a reprodução do comportamento de programas legados (isofuncionalidade), desempenhos (com relação ao tempo de execução dos programas e ao consumo de recursos) e a facilidade de manutenção de programas modernizados por desenvolvedores Java, por meio do uso de ambientes e expressões idiomáticas familiares, como tomcat, Spring, getters/setters e APIs fluentes.

Tópicos

AWS Componentes de tempo de execução do Blu Age

O ambiente AWS Blu Age Runtime é composto por dois tipos de componentes:

- Um conjunto de bibliotecas java (arquivos jar) geralmente referenciado como “a pasta compartilhada” e que fornece estruturas e declarações antigas.
- Um conjunto de aplicações web (arquivos war) contendo aplicações web baseadas em Spring que fornecem um conjunto comum de estruturas e serviços para programas modernizados.

As seções a seguir detalham a função de ambos os componentes.

AWS Bibliotecas Blu Age

As bibliotecas do AWS Blu Age são um conjunto de arquivos jar armazenados em uma `shared/` subpasta adicionada ao classpath padrão do tomcat, a fim de disponibilizá-los para todos os programas Java modernizados. O objetivo deles é fornecer recursos que não estejam nem de forma nativa nem facilmente disponíveis no ambiente de programação Java, mas que sejam típicos de ambientes de desenvolvimento antigos. Esses recursos são expostos de uma forma que seja o mais familiar possível para os desenvolvedores Java (getters/setters, APIs fluentes e baseadas em classes). Um exemplo importante é a biblioteca Data Simplifier, que fornece estruturas antigas de layout e manipulação de memória (encontradas nas linguagens COBOL, PL1 ou RPG) para

programas Java. Esses jars são uma dependência central do código Java modernizado gerado a partir de programas antigos. Para obter mais informações sobre o Data Simplifier, consulte Simplificador de dados.

Aplicativo Web

Os Arquivos de Aplicações Web (WARs) são uma forma padrão de implantar código e aplicações no servidor de aplicações tomcat. Os fornecidos como parte do tempo de execução do AWS Blu Age visam fornecer um conjunto de estruturas de execução que reproduzem ambientes legados e monitores de transações (lotes JCL, CICS, IMS...) e os serviços necessários associados.

O mais importante é `gapwalk-application` (geralmente abreviado como “Gapwalk”), que fornece um conjunto unificado de pontos de entrada baseados em REST para acionar e controlar transações, programas e execução de lotes. Para ter mais informações, consulte AWS APIs de tempo de execução do Blu Age.

Essa aplicação web aloca threads e recursos de execução do Java para executar programas modernizados no contexto para o qual foram projetados. Exemplos desses ambientes reproduzidos são detalhados na seção a seguir.

Outras aplicações da Web adicionam ao ambiente de execução (mais precisamente, ao “Registro de Programas” descrito abaixo) programas que emulam aqueles disponíveis e que podem ser chamados pelos programas antigos. Duas categorias importantes são:

- Emulação de programas fornecidos pelo sistema operacional: os lotes controlados pela JCL esperam poder chamar uma variedade de programas de manipulação de arquivos e bancos de dados como parte de seu ambiente padrão. Exemplos incluem `SORT`/`DFSORT` ou `IDCAMS`. Para isso, são fornecidos programas Java que reproduzem esse comportamento e podem ser chamados usando as mesmas convenções dos antigos.
- “Drivers”, que são programas especializados fornecidos pela estrutura de execução ou pelo middleware como pontos de entrada. Um exemplo é `CBLTDLI` de quais programas COBOL executados no ambiente IMS dependem para acessar os serviços relacionados ao IMS (IMS DB, diálogo do usuário por meio do MFS etc.).

Registro de programas

Para participar e tirar proveito dessas construções, estruturas e serviços, os programas Java modernizados a partir dos antigos aderem a uma estrutura específica documentada em AWS Estrutura Blu Age de um aplicativo modernizado. Na inicialização, o AWS Blu Age Runtime coletará

todos esses programas em um “Registro de Programas” comum para que possam ser invocados (e chamados uns aos outros) posteriormente. O Registro de Programas oferece acoplamento fraco e possibilidades de decomposição (já que os programas que se chamam não precisam ser modernizados simultaneamente).

Ambientes de execução

Ambientes e coreografias antigos frequentemente encontrados estão disponíveis:

- Os lotes controlados pela JCL, uma vez modernizados para programas Java e scripts Groovy, podem ser iniciados de forma síncrona (bloqueio) ou assíncrona (desanexada). No último caso, sua execução pode ser monitorada por meio de endpoints REST.
- Um subsistema AWS Blu Age fornece um ambiente de execução semelhante ao CICS por meio de:
  - um ponto de entrada usado para iniciar uma transação do CICS e executar programas associados, respeitando a coreografia dos “níveis de execução” do CICS,
  - um armazenamento externo para definições de recursos,
  - um conjunto homogêneo de APIs fluentes em Java que reproduzem declarações, `EXEC CICS`
  - um conjunto de classes conectáveis que reproduzem serviços do CICS, como filas de armazenamento temporário, filas de dados temporários ou acesso a arquivos (várias implementações geralmente estão disponíveis, como Amazon Managed Service for Apache Flink, Amazon Simple Queue Service ou RabbitMQ para filas TD),
  - para aplicações voltadas para o usuário, o formato de descrição de tela do BMS é modernizado para uma aplicação web Angular e a caixa de diálogo “pseudo-conversacional” correspondente é suportada.
- Da mesma forma, outro subsistema fornece coreografia baseada em mensagens IMS e oferece suporte à modernização de telas de interface do usuário no formato MFS.
- Além disso, um terceiro subsistema permite a execução de programas em um ambiente semelhante ao iSeries, incluindo a modernização de telas especificadas pelo DSPF (Display File).

Todos esses ambientes se baseiam em serviços comuns em nível de sistema operacional, como:

- a emulação da alocação e layout de memória antigos (Simplificador de dados),
- Reprodução baseada em threads o Java da execução de “unidades de execução” do COBOL e do mecanismo de passagem de parâmetros (instrução `CALL`).

- emulação de organizações planas e concatenadas de VSAM (por meio do conjunto de bibliotecas Blusam) e GDG Data Set,
- acesso a armazenamentos de dados, como RDBMS (`EXEC SQL`declarações).

Ausência de estado e gerenciamento de sessões

Um recurso importante do AWS Blu Age Runtime é permitir cenários de alta disponibilidade (HA) e escalabilidade horizontal ao executar programas modernizados.

A base para isso é a apátrida, um exemplo importante disso é o tratamento de sessões HTTP.

Manuseio de sessões

Sendo o Tomcat baseado na web, um mecanismo importante para isso é o tratamento da sessão HTTP (conforme fornecido pelo tomcat e pelo Spring) e o design sem estado. Como tal, o design de ausência de estado é baseado no seguinte:

- os usuários se conectam por meio de HTTPS,
- os servidores de aplicações são implantados por trás de um balanceador de carga,
- quando um usuário se conecta pela primeira vez à aplicação, ele será autenticado e o servidor da aplicação criará um identificador (normalmente dentro de um cookie)
- esse identificador será usado como uma chave para salvar e recuperar o contexto do usuário de/para um cache externo (armazenamento de dados).

O gerenciamento de cookies é feito automaticamente pela estrutura AWS Blu Age e pelo servidor tomcat subjacente, isso é transparente para o usuário. O navegador da Internet do usuário gerenciará isso automaticamente.

A aplicação web Gapwalk pode armazenar o estado da sessão (o contexto) em vários armazenamentos de dados:

- Amazon ElastiCache para Redis
- Cluster do Redis
- no mapa de memória (somente para ambientes autônomos e de desenvolvimento, não adequado para HA).

Alta disponibilidade e apátrida

De forma mais geral, um princípio de design da estrutura AWS Blu Age é a apatridia: a maioria dos estados não transitórios necessários para reproduzir o comportamento de programas legados não são armazenados nos servidores de aplicativos, mas compartilhados por meio de uma “fonte única de verdade” externa comum.

Exemplos desses estados são as filas de armazenamento temporário ou as definições de recursos do CICS, e os armazenamentos externos típicos deles são servidores ou bancos de dados relacionais compatíveis com Redis.

Esse design, combinado com balanceamento de carga e sessões compartilhadas, faz com que a maioria dos diálogos voltados para o usuário (OLTP, “Processamento Transacional Online”) seja distribuída entre vários “nós” (aqui, instâncias do Tomcat).

Na verdade, um usuário pode executar uma transação em qualquer servidor sem se importar se a próxima chamada de transação será realizada em um servidor diferente. Então, quando um novo servidor é gerado (devido ao ajuste de escala automático ou para substituir um servidor não íntegro), podemos garantir que qualquer servidor acessível e íntegro possa executar a transação conforme o esperado com os resultados adequados (valor retornado esperado, alteração esperada de dados no banco de dados, etc.).

AWS Estrutura Blu Age de um aplicativo modernizado

Este documento fornece detalhes sobre a estrutura de aplicativos modernizados (usando ferramentas de refatoração de modernização de AWS mainframe), para que os desenvolvedores possam realizar várias tarefas, como:

- navegando pelas aplicações sem problemas.
- desenvolvendo programas personalizados que podem ser chamados a partir das aplicações modernizadas.
- refatorando com segurança aplicações modernizadas.

Presumimos que você já tenha conhecimentos básicos sobre o seguinte:

- conceitos de codificação comuns herdados, como registros, conjuntos de dados e seus modos de acesso aos registros — indexados, sequenciais —, VSAM, unidades de execução, scripts jcl, conceitos do CICS e assim por diante.
- codificação java usando o framework Spring.

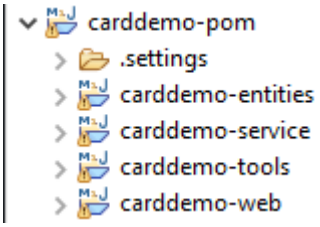
- Em todo o documento, usamos `short class names` para facilitar a leitura. Para obter mais informações, consulte AWS Mapeamentos de nomes totalmente qualificados da Blu Age para recuperar os nomes totalmente qualificados correspondentes para os elementos de tempo de execução do AWS Blu Age e Mapeamentos de nomes totalmente qualificados de terceiros para recuperar os nomes totalmente qualificados correspondentes para elementos de terceiros.
- Todos os artefatos e amostras são retirados das saídas do processo de modernização do aplicativo COBOL/CICS de amostra. CardDemo

Tópicos

Organização de artefatos

AWS Os aplicativos modernizados do Blu Age são empacotados como aplicativos web java (.war), que você pode implantar em um servidor JEE. Normalmente, o servidor é uma instância do Tomcat que incorpora o AWS Blu Age Runtime, que atualmente é construído com base nas estruturas Springboot e Angular (para a parte da interface do usuário).

A guerra agrega vários artefatos de componentes (.jar). Cada jar é o resultado da compilação (usando a ferramenta maven) de um projeto java dedicado cujos elementos são o resultado do processo de modernização.



A organização básica se baseia na seguinte estrutura:

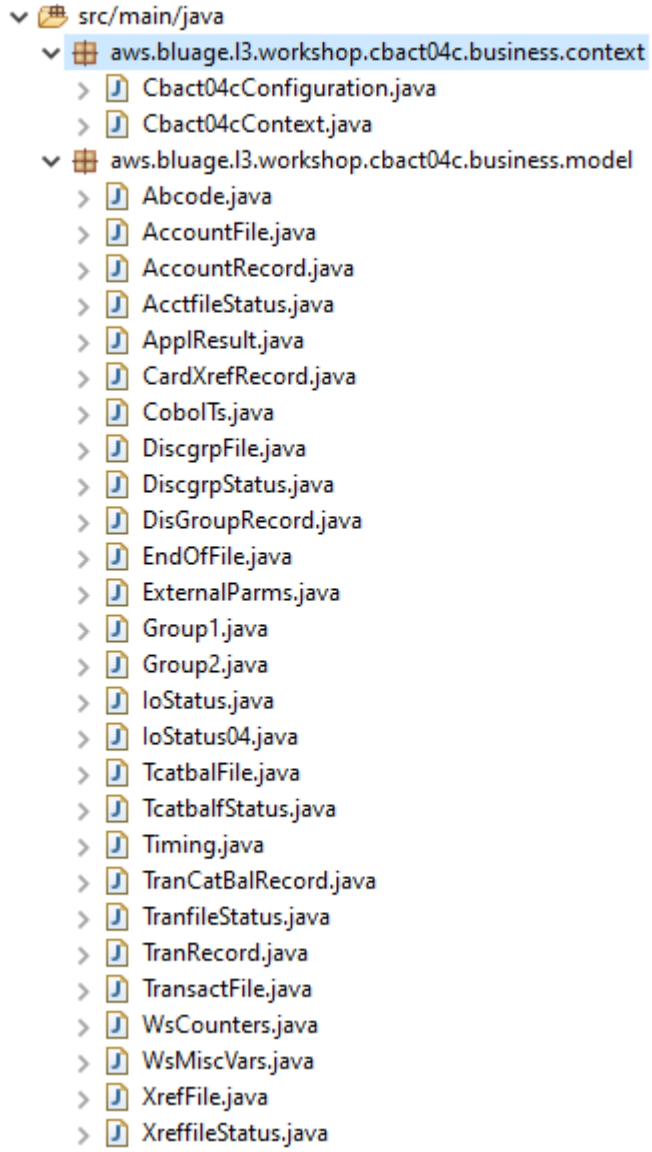
- Projeto de entidades: contém elementos do modelo de negócios e do contexto. O nome do projeto geralmente termina com “-entities”. Normalmente, para um determinado programa COBOL antigo, isso corresponde à modernização da seção de E/S (conjuntos de dados) e da divisão de dados. É possível ter um projeto de mais de uma entidade.

- Projeto de serviço: contém elementos antigos de modernização da lógica de negócios. Normalmente, a divisão de procedimentos de um programa COBOL. Você pode ter mais de um projeto de serviço.
- Projeto utilitário: contém ferramentas e utilitários comuns compartilhados, usados por outros projetos.
- Projeto Web: contém a modernização de elementos relacionados à interface do usuário, quando aplicável. Não é usado para projetos de modernização somente em lote. Esses elementos da interface do usuário podem vir dos mapas do CICS BMS, dos componentes do IMS MFS e de outras fontes de interface do usuário do mainframe. Você pode ter mais de um projeto da Web.

Conteúdo do projeto Entidades

> **Note**
>
> As descrições a seguir se aplicam somente às saídas de modernização COBOL e PL/I. As saídas de modernização do RPG são baseadas em um layout diferente.

Antes de qualquer refatoração, a organização dos pacotes no projeto da entidade está vinculada aos programas modernizados. É possível fazer isso de duas maneiras diferentes. A forma preferida é usar a caixa de ferramentas de refatoração, que opera antes de você acionar o mecanismo de geração de código. Esta é uma operação avançada, que é explicada nos BluAge treinamentos. Para obter mais informações, consulte Workshop de refatoração. Essa abordagem permite que você preserve a capacidade de regenerar o código java posteriormente, para se beneficiar de novas melhorias no futuro, por exemplo). A outra maneira é fazer a refatoração regular de java, diretamente no código-fonte gerado, usando qualquer abordagem de refatoração de java que você queira aplicar -- por sua conta e risco.



Aulas relacionadas ao programa

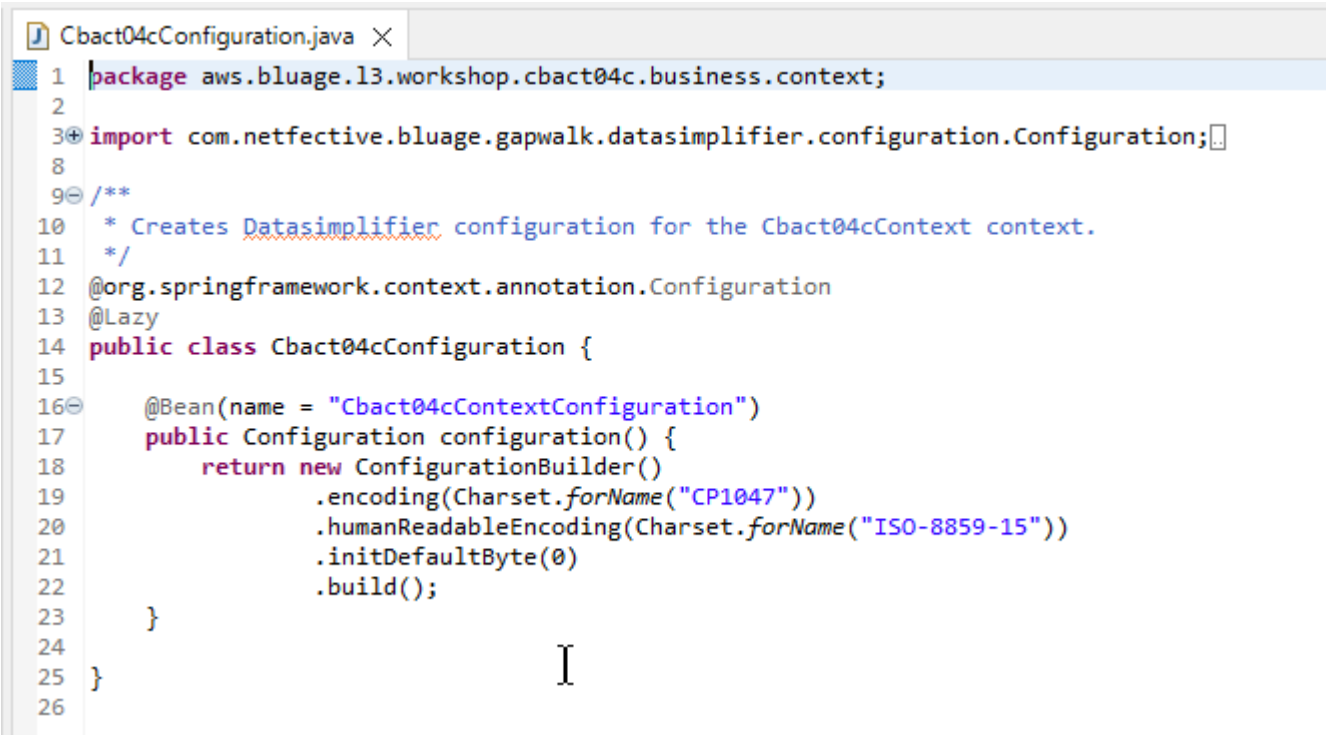
Cada programa modernizado está relacionado a dois pacotes, um pacote business.context e um pacote business.model.

- *`base package`*`.`*`program`*`.business.context`

  O subpacote business.context contém duas classes, uma classe de configuração e uma classe de contexto.

  - Uma classe de configuração para o programa, que contém detalhes de configuração específicos para um determinado programa, como o conjunto de caracteres a ser usado para representar elementos de dados baseados em caracteres, o valor de byte padrão para preencher elementos

da estrutura de dados e assim por diante. O nome da classe termina com “Configuração”. Ele é marcado com a `@org.springframework.context.annotation.Configuration` anotação e contém um único método que deve retornar um objeto `Configuration` configurado corretamente.



- Uma classe de contexto, que serve como uma ponte entre as classes de serviço do programa (veja abaixo) e as estruturas de dados (`Record`) e os conjuntos de dados (`File`) do subpacote do modelo (veja abaixo). O nome da classe termina com “Contexto” e é uma subclasse da `RuntimeContext` classe.

Cbact04cContext.java

```
@Component("aws.bluage.l3.workshop.cbact04c.business.context.Cbact04cContext")
@Import({
    aws.bluage.l3.workshop.cbact04c.business.model.TcatbalFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.XrefFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.DiscgrpFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.AccountFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.TransactFile.class
})
@Lazy
@Scope("prototype")
public class Cbact04cContext extends JicsRuntimeContext {

    @Autowired
    private TcatbalFile tcatbalFile;

    @Autowired
    private XrefFile xrefFile;

    @Autowired
    private DiscgrpFile discgrpFile;

    @Autowired
    private AccountFile accountFile;

    @Autowired
    private TransactFile transactFile;

    private IndexedFile tcatbalFileFile;

    private IndexedFile xrefFileFile;

    private IndexedFile discgrpFileFile;

    private IndexedFile accountFileFile;

    private SequentialFile transactFileFile;

    private TranCatBalRecord tranCatBalRecord;
    private TcatbalfStatus tcatbalfStatus;
    private CardXrefRecord cardXrefRecord;
```

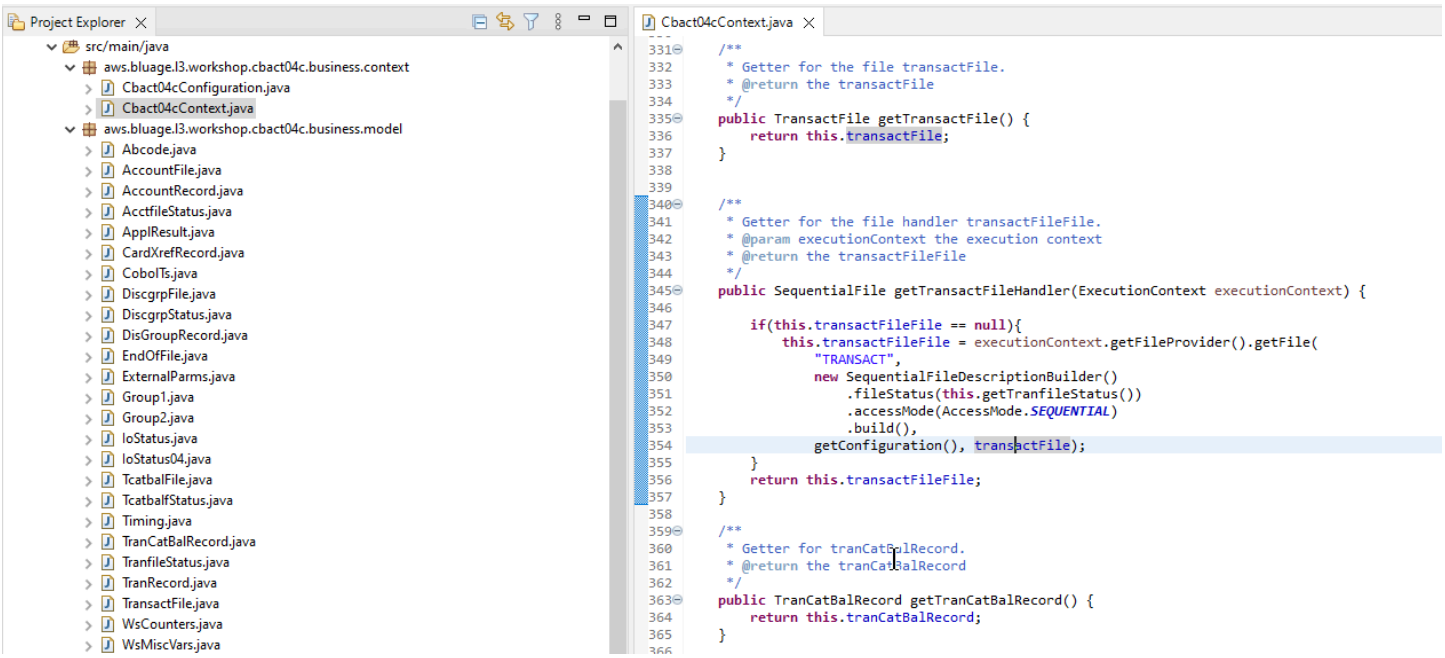
- *`base package`*`.`*`program`*`.business.model`

  O subpacote do modelo contém todas as estruturas de dados que o programa fornecido pode usar. Por exemplo, qualquer estrutura de dados COBOL de nível 01 corresponde a uma classe no subpacote do modelo (estruturas de dados de nível inferior são propriedades de sua própria estrutura de nível 01). Para obter mais informações sobre como modernizamos 01 estruturas de dados, consulte Simplificador de dados.

DiscgrpFile.java

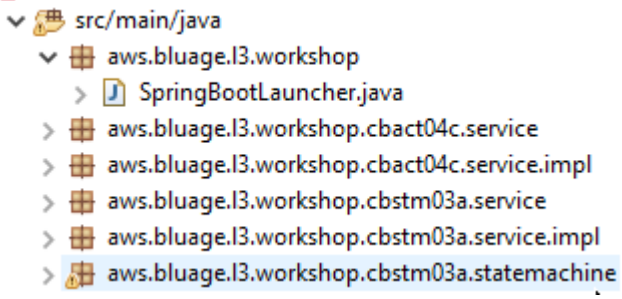
```java
package aws.bluage.l3.workshop.cbact04c.business.model;

import com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration;
import com.netfective.bluage.gapwalk.datasimplifier.data.structure.Elementary;
import com.netfective.bluage.gapwalk.datasimplifier.data.structure.Group;
import com.netfective.bluage.gapwalk.datasimplifier.entity.ElementaryRangeReference;
import com.netfective.bluage.gapwalk.datasimplifier.entity.RangeReference;
import com.netfective.bluage.gapwalk.datasimplifier.entity.RecordEntity;
import com.netfective.bluage.gapwalk.datasimplifier.metadata.type.AlphanumericType;
import com.netfective.bluage.gapwalk.datasimplifier.metadata.type.ZonedType;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.context.annotation.Lazy;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Component;

/**
 * Data simplifier file DiscgrpFile.
 *
 * <p>About 'fdDiscgrpRec' field, <br>uml entity: aws.bluage.l3.workshop.cbact04c.business.model.FdDiscgrpRec
 * <br></p>
 *
 */
@Component("aws.bluage.l3.workshop.cbact04c.business.model.DiscgrpFile")
@Lazy
@Scope("prototype")
public class DiscgrpFile extends RecordEntity {

    private final Group root = new Group(getData());
    private final Group fdDiscgrpRec = new Group(root);
    private final Group fdDiscgrpKey = new Group(fdDiscgrpRec);
    private final Elementary fdDisAcctGroupId = new Elementary(fdDiscgrpKey,new AlphanumericType(10));
    private final Elementary fdDisTranTypeCd = new Elementary(fdDiscgrpKey,new AlphanumericType(2));
    private final Elementary fdDisTranCatCd = new Elementary(fdDiscgrpKey,new ZonedType(4, 0, false));
    private final Elementary fdDiscgrpData = new Elementary(fdDiscgrpRec,new AlphanumericType(34));
```

Todas as classes estendem a `RecordEntity` classe, que representa o acesso a uma representação de registros comerciais. Alguns dos registros têm um propósito especial, pois estão vinculados a um `File`. A vinculação entre a `Record` e a `File` é feita nos FileHandler métodos * correspondentes encontrados na classe de contexto ao criar o objeto de arquivo. Por exemplo, a lista a seguir mostra como o TransactfileFile `File` está vinculado ao TransactFile `Record` (do subpacote do modelo).



Conteúdo do projeto de serviço

Cada projeto de serviço vem com uma aplicação Springboot dedicada, que é usado como a espinha dorsal da arquitetura. Isso é materializado por meio da classe chamada `SpringBootLauncher`, localizada no pacote base das fontes java do serviço:



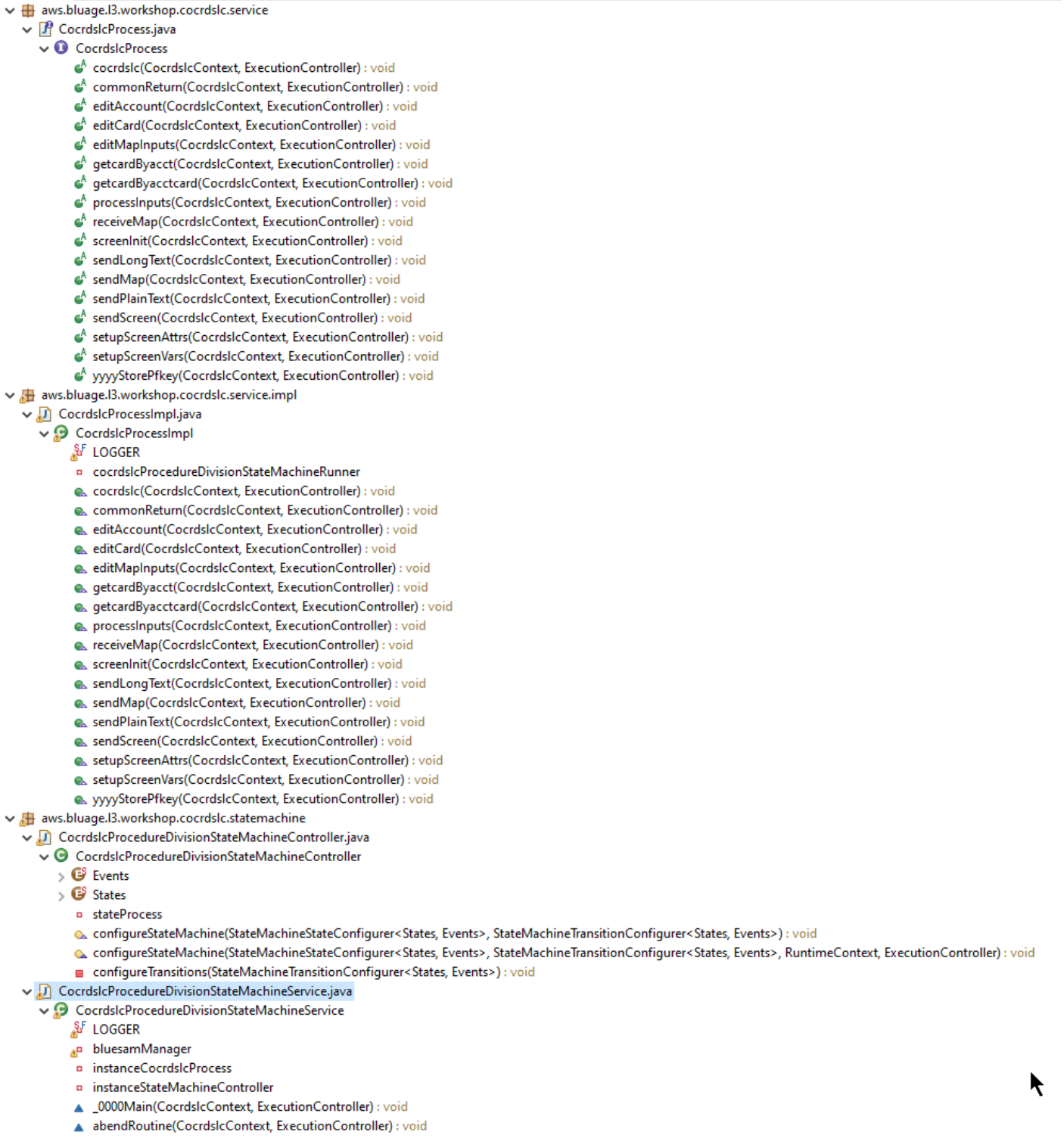
Essa classe é especialmente responsável por:

- criando a cola entre as classes do programa e os recursos gerenciados (fontes de dados/ gerenciadores de transações/mapeamentos de conjuntos de dados/ etc.).
- fornecendo dois `ConfigurableApplicationContext` programas.
- descobrindo todas as classes marcadas como componentes de primavera (`@Component`).
- garantindo que os programas sejam registrados corretamente no `ProgramRegistry` -- veja o método de inicialização responsável por esse registro.

```
/**
 * Initialization method called when the spring application is ready.
 * Register all programs and services to the gapwalk shared context.
 * @param event the application ready event
 */
@EventListener
public void initialize(ApplicationReadyEvent event) {
    Map<String, ProgramContainer> programContainers = event.getApplicationContext().getBeansOfType(ProgramContainer.class);
    programContainers.values().forEach(ProgramRegistry::registerProgram);
    Map<String, ServiceContainer> serviceContainers = event.getApplicationContext().getBeansOfType(ServiceContainer.class);
    serviceContainers.values().forEach(ServiceRegistry::registerService);
}
```

Artefatos relacionados ao programa

Sem refatoração prévia, os resultados da modernização da lógica de negócios são organizados em dois ou três pacotes por programa antigo:



O estojo mais exaustivo terá três pacotes:

- *base package*`.`*program*`.service`: contém uma interface chamada ProgramProcess, que tem métodos de negócios para lidar com a lógica de negócios, preservando o fluxo de controle de execução antigo.
- *base package*`.`*program*`.service.impl`: contém uma classe chamada ProgramaProcessImpl, que é a implementação da interface de processo descrita anteriormente. É aqui que as declarações legadas são “traduzidas” para instruções java, com base na estrutura AWS Blu Age:

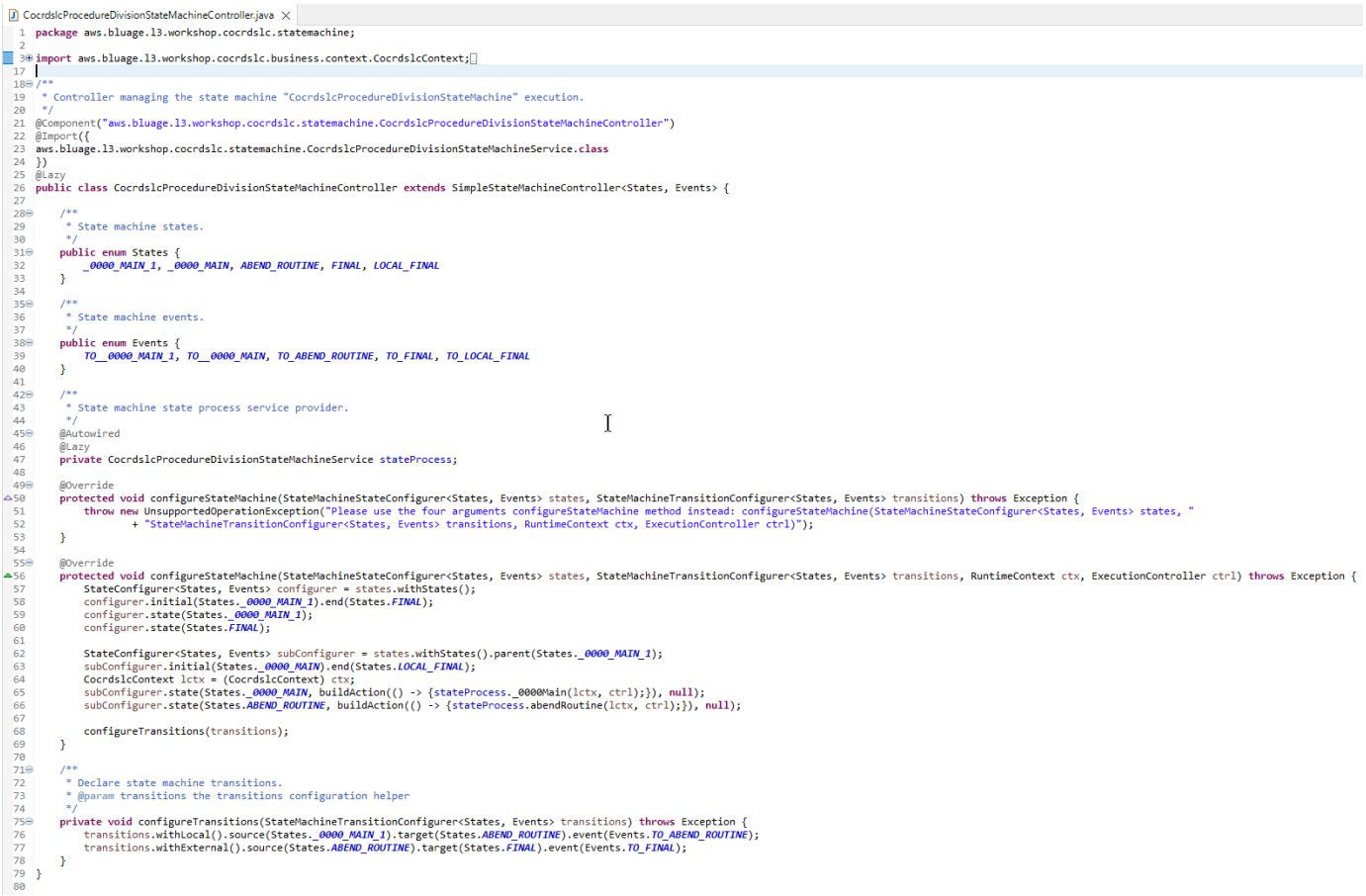
CocrdslcProcessImpl.java

```java
    /**
     * Process operation sendScreen.
     *
     * @param ctx
     * @param ctrl
     */
    @Override
    public void sendScreen(final CocrdslcContext ctx, final ExecutionController ctrl) {
        ctx.getCcWorkAreas().setCcardNextMapset(ctx.getWsLiterals().getLitThismapset());
        ctx.getCcWorkAreas().setCcardNextMap(ctx.getWsLiterals().getLitThismap());
        ctx.getCarddemoCommarea().setCdemoPgmReenter(true);
        SendMapBuilder.newInstance(ctx.getDfheiblk(), ctx)
        .withMap(ctx.getCcWorkAreas().getCcardNextMap())
        .withMapset(ctx.getCcWorkAreas().getCcardNextMapset())
        .withData(ctx.getGroup1().getCcrdslaoReference())
        .withCursor()
        .withErase()
        .withFreeKB()
        .execute();
        ctx.getWsMiscStorage().setWsRespCd(ctx.getDfheiblk().getEibresp());
    }

    /**
     * Process operation processInputs.
     *
     * @param ctx
     * @param ctrl
     */
    @Override
    public void processInputs(final CocrdslcContext ctx, final ExecutionController ctrl) {
        receiveMap(ctx, ctrl);
        editMapInputs(ctx, ctrl);
        ctx.getCcWorkAreas().setCcardErrorMsg(ctx.getWsMiscStorage().getWsReturnMsg());
        ctx.getCcWorkAreas().setCcardNextProg(ctx.getWsLiterals().getLitThispgm());
        ctx.getCcWorkAreas().setCcardNextMapset(ctx.getWsLiterals().getLitThismapset());
        ctx.getCcWorkAreas().setCcardNextMap(ctx.getWsLiterals().getLitThismap());
    }

```

- *base package*`.`*program*`.statemachine`: esse pacote pode nem sempre estar presente. É necessário quando a modernização do fluxo de controle legado precisa usar uma abordagem de máquina de estado (ou seja, usar a StateMachine estrutura Spring) para cobrir adequadamente o fluxo de execução legado.

  Nesse caso, o subpacote statemachine contém duas classes:

- *Program*`ProcedureDivisionStateMachineController`: uma classe que estende uma classe que implementa as interfaces `StateMachineController` (define as operações necessárias para controlar a execução de uma máquina de estado) e `StateMachineRunner` (define as operações necessárias para executar uma máquina de estado), usadas para conduzir a mecânica da máquina de estado Spring; por exemplo, `SimpleStateMachineController` como no caso de exemplo.



```
CocrdslcProcedureDivisionStateMachineController.java
package aws.bluage.l3.workshop.cocrdslc.statemachine;

import aws.bluage.l3.workshop.cocrdslc.business.context.CocrdslcContext;

/**
 * Controller managing the state machine "CocrdslcProcedureDivisionStateMachine" execution.
 */
@Component("aws.bluage.l3.workshop.cocrdslc.statemachine.CocrdslcProcedureDivisionStateMachineController")
@Import({
aws.bluage.l3.workshop.cocrdslc.statemachine.CocrdslcProcedureDivisionStateMachineService.class
})
@Lazy
public class CocrdslcProcedureDivisionStateMachineController extends SimpleStateMachineController<States, Events> {

    /**
     * State machine states.
     */
    public enum States {
        _0000_MAIN_1, _0000_MAIN, ABEND_ROUTINE, FINAL, LOCAL_FINAL
    }

    /**
     * State machine events.
     */
    public enum Events {
        TO__0000_MAIN_1, TO__0000_MAIN, TO_ABEND_ROUTINE, TO_FINAL, TO_LOCAL_FINAL
    }

    /**
     * State machine state process service provider.
     */
    @Autowired
    @Lazy
    private CocrdslcProcedureDivisionStateMachineService stateProcess;

    @Override
    protected void configureStateMachine(StateMachineStateConfigurer<States, Events> states, StateMachineTransitionConfigurer<States, Events> transitions) throws Exception {
        throw new UnsupportedOperationException("Please use the four arguments configureStateMachine method instead: configureStateMachine(StateMachineStateConfigurer<States, Events> states, "
                + "StateMachineTransitionConfigurer<States, Events> transitions, RuntimeContext ctx, ExecutionController ctrl)");
    }

    @Override
    protected void configureStateMachine(StateMachineStateConfigurer<States, Events> states, StateMachineTransitionConfigurer<States, Events> transitions, RuntimeContext ctx, ExecutionController ctrl) throws Exception {
        StateConfigurer<States, Events> configurer = states.withStates();
        configurer.initial(States._0000_MAIN_1).end(States.FINAL);
        configurer.state(States._0000_MAIN_1);
        configurer.state(States.FINAL);

        StateConfigurer<States, Events> subConfigurer = states.withStates().parent(States._0000_MAIN_1);
        subConfigurer.initial(States._0000_MAIN).end(States.LOCAL_FINAL);
        CocrdslcContext lctx = (CocrdslcContext) ctx;
        subConfigurer.state(States._0000_MAIN, buildAction(() -> {stateProcess._0000Main(lctx, ctrl);}), null);
        subConfigurer.state(States.ABEND_ROUTINE, buildAction(() -> {stateProcess.abendRoutine(lctx, ctrl);}), null);

        configureTransitions(transitions);
    }

    /**
     * Declare state machine transitions.
     * @param transitions the transitions configuration helper
     */
    private void configureTransitions(StateMachineTransitionConfigurer<States, Events> transitions) throws Exception {
        transitions.withLocal().source(States._0000_MAIN_1).target(States.ABEND_ROUTINE).event(Events.TO_ABEND_ROUTINE);
        transitions.withExternal().source(States.ABEND_ROUTINE).target(States.FINAL).event(Events.TO_FINAL);
    }
}
```

O controlador da máquina de estado define os diferentes estados possíveis e as transições entre eles, que reproduzem o fluxo de controle de execução antigo para o programa em questão.

Ao criar a máquina de estado, o controlador se refere aos métodos definidos na classe de serviço associada localizada no pacote da máquina de estado e descrita abaixo:

```
subConfigurer.state(States._0000_MAIN, buildAction(() ->
 {stateProcess._0000Main(lctx, ctrl);}), null);
subConfigurer.state(States.ABEND_ROUTINE, buildAction(() ->
 {stateProcess.abendRoutine(lctx, ctrl);}), null);
```

- *Program*ProcedureDivisionStateMachineService: essa classe de serviço representa alguma lógica de negócios que precisa ser vinculada à máquina de estado criada pelo controlador da máquina de estado, conforme descrito anteriormente.

  O código nos métodos dessa classe usa os eventos definidos no controlador da máquina de estado:

CocrdslcProcedureDivisionStateMachineService.java

```
59    /**
60     * State process operation _0000Main.
61     *
62     * @param ctx
63     * @param ctrl
64     */
65    void _0000Main(CocrdslcContext ctx, ExecutionController ctrl) {
66        ctx.getDfheiblk().bind(ArgUtils.get(ctx, 0));
67        ctx.getDfhcommarea().bind(ArgUtils.get(ctx, 1));
68
69        /*
70        ****************************************************************
71        Program:     COCRDSLC.CBL                                      *
72        Layer:       Business logic                                    *
73        Function:    Accept and process credit card detail request     *
74        ****************************************************************
75        Copyright Amazon.com, Inc. or its affiliates.
76        All Rights Reserved.
77        Licensed under the Apache License, Version 2.0 (the \"License\").
78        You may not use this file except in compliance with the License.
79        You may obtain a copy of the License at
80        http://www.apache.org/licenses/LICENSE-2.0
81        Unless required by applicable law or agreed to in writing,
82        software distributed under the License is distributed on an
83        \"AS IS\" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND,
84        either express or implied. See the License for the specific
85        language governing permissions and limitations under the License
86        ****************************************************************
87        Ver: CardDemo_v1.0-15-g27d6c6f-68 Date: 2022-07-19 23:16:00 CDT */
88        instanceStateMachineController.registerSignalHandler(Events.TO_ABEND_ROUTINE, "!ABEND");
89        HandleAbendBuilder.newInstance(ctx.getDfheiblk(), ctx).execute().handleException();
90        ctx.getCcWorkAreas().getCcWorkAreaReference().getField().initialize();
91        ctx.getWsMiscStorage().getField().initialize();
92        DataUtils.initialize(ctx.getWsCommarea().getWsCommareaReference());
93
```

CocrdslcProcedureDivisionStateMachineService.java

```
     *
     * @param ctx
     * @param ctrl
     */
    void abendRoutine(CocrdslcContext ctx, ExecutionController ctrl) {
        if (DataUtils.isLowValue(ctx.getAbendData().getAbendMsgReference())) {
            ctx.getAbendData().setAbendMsg("UNEXPECTED ABEND OCCURRED.");
        }
        ctx.getAbendData().setAbendCulprit(ctx.getWsLiterals().getLitThispgm());
        SendTextBuilder.newInstance(ctx.getDfheiblk(), ctx)
        .withData(ctx.getAbendData())
        .withLength(134)
        .execute();
        HandleAbendBuilder.newInstance(ctx.getDfheiblk(), ctx).cancel().execute().handleException();
        AbendBuilder.newInstance(ctx.getDfheiblk(), ctx).withAbendCode("9999").execute().handleException();

        /*
        Ver: CardDemo_v1.0-15-g27d6c6f-68 Date: 2022-07-19 23:12:33 CDT */
        instanceStateMachineController.sendEvent(Events.TO_FINAL);

    }
}
```

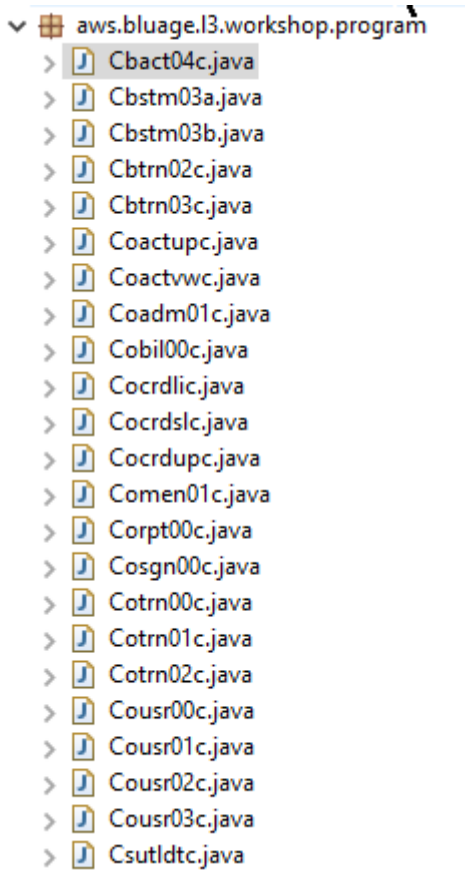
O serviço statemachine também faz chamadas para a implantação do serviço de processo descrita anteriormente:

CocrdslcProcedureDivisionStateMachineService.java

```
        /*
        ************************************************************
        COMING FROM CREDIT CARD LIST SCREEN
        SELECTION CRITERIA ALREADY VALIDATED
        ************************************************************ */
    } else if (ctx.getCarddemoCommarea().isCdemoPgmEnter() && DataUtils.compare(ctx.getCarddemoCommarea().getCdemoFromProgramReference(), ctx.getWsLiterals().getLitCclistpgmReference()) == 0) {
        ctx.getWsMiscStorage().setInputOk(true);
        ctx.getCcWorkAreas().setCcAcctIdN(ctx.getCarddemoCommarea().getCdemoAcctId());
        ctx.getCcWorkAreas().setCcCardNumN(ctx.getCarddemoCommarea().getCdemoCardNum());
        instanceCocrdslcProcess.getcardByacctcard(ctx, ctrl);
        instanceCocrdslcProcess.sendMap(ctx, ctrl);
        instanceCocrdslcProcess.commonReturn(ctx, ctrl);
    } else if (ctx.getCarddemoCommarea().isCdemoPgmEnter()) {

        /*
        ************************************************************
        COMING FROM SOME OTHER CONTEXT
        SELECTION CRITERIA TO BE GATHERED
        ************************************************************ */
        instanceCocrdslcProcess.sendMap(ctx, ctrl);
        instanceCocrdslcProcess.commonReturn(ctx, ctrl);
    } else if (ctx.getCarddemoCommarea().isCdemoPgmReenter()) {
        instanceCocrdslcProcess.processInputs(ctx, ctrl);
        if (ctx.getWsMiscStorage().isInputError()) {
            instanceCocrdslcProcess.sendMap(ctx, ctrl);
            instanceCocrdslcProcess.commonReturn(ctx, ctrl);
        } else {
            instanceCocrdslcProcess.getcardByacctcard(ctx, ctrl);
            instanceCocrdslcProcess.sendMap(ctx, ctrl);
            instanceCocrdslcProcess.commonReturn(ctx, ctrl);
        }
    } else {
        ctx.getAbendData().setAbendCulprit(ctx.getWsLiterals().getLitThispgm());
        ctx.getAbendData().setAbendCode("0001");
        DataUtils.setToBlank(ctx.getAbendData().getAbendReasonReference());
        ctx.getWsMiscStorage().setWsReturnMsg("UNEXPECTED DATA SCENARIO");
        instanceCocrdslcProcess.sendPlainText(ctx, ctrl);
    }

    /*
    If we had an error setup error message that slipped through
    Display and return */
    if (ctx.getWsMiscStorage().isInputError()) {
        ctx.getCcWorkAreas().setCcardErrorMsg(ctx.getWsMiscStorage().getWsReturnMsg());
        instanceCocrdslcProcess.sendMap(ctx, ctrl);
        instanceCocrdslcProcess.commonReturn(ctx, ctrl);
    }
    instanceCocrdslcProcess.commonReturn(ctx, ctrl);
```

Além disso, um pacote chamado *`base package`*`.program` desempenha um papel importante, pois reúne uma classe por programa, que servirá como ponto de entrada do programa (mais detalhes

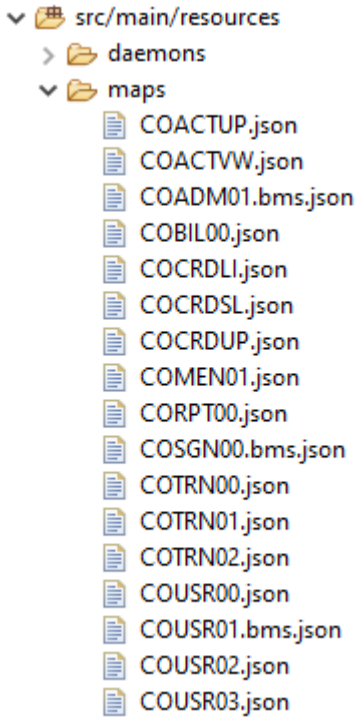
sobre isso posteriormente). Cada classe implementa a interface `Program`, marcador para um ponto de entrada do programa.



Outros artefatos

- Companheiros do BMS MAPs

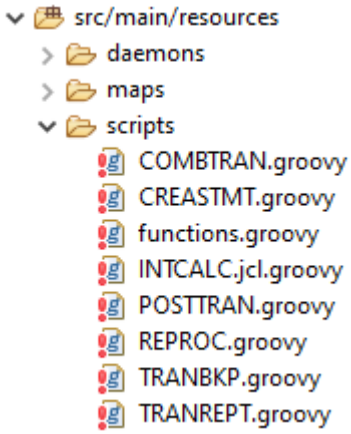
  Além dos artefatos relacionados ao programa, o projeto de serviço pode conter outros artefatos para várias finalidades. No caso da modernização de uma aplicação online do CICS, o processo de modernização produz um arquivo json e coloca na pasta map da pasta /src/main/resources:



O Blu Age Runtime consome esses arquivos json para vincular os registros usados pela instrução SEND MAP aos campos da tela.

- Scripts Groovy

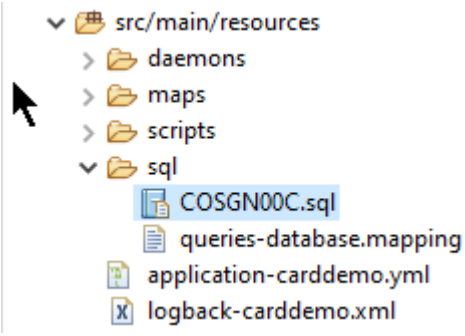
  Se a aplicação antiga tinha scripts JCL, eles foram modernizados como scripts groovy, armazenados na pasta /src/main/resources/scripts (mais sobre esse local específico posteriormente):



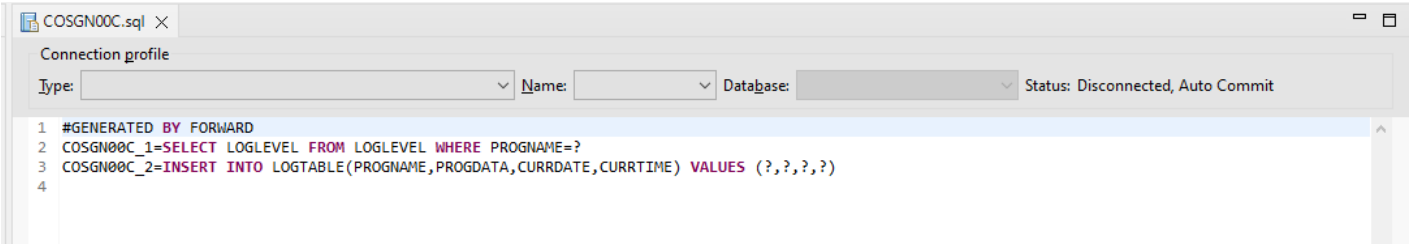
  Esses scripts são usados para iniciar trabalhos em lotes (workloads de processamento de dados dedicadas, não interativas e com uso intenso de CPU).

- Arquivos SQL

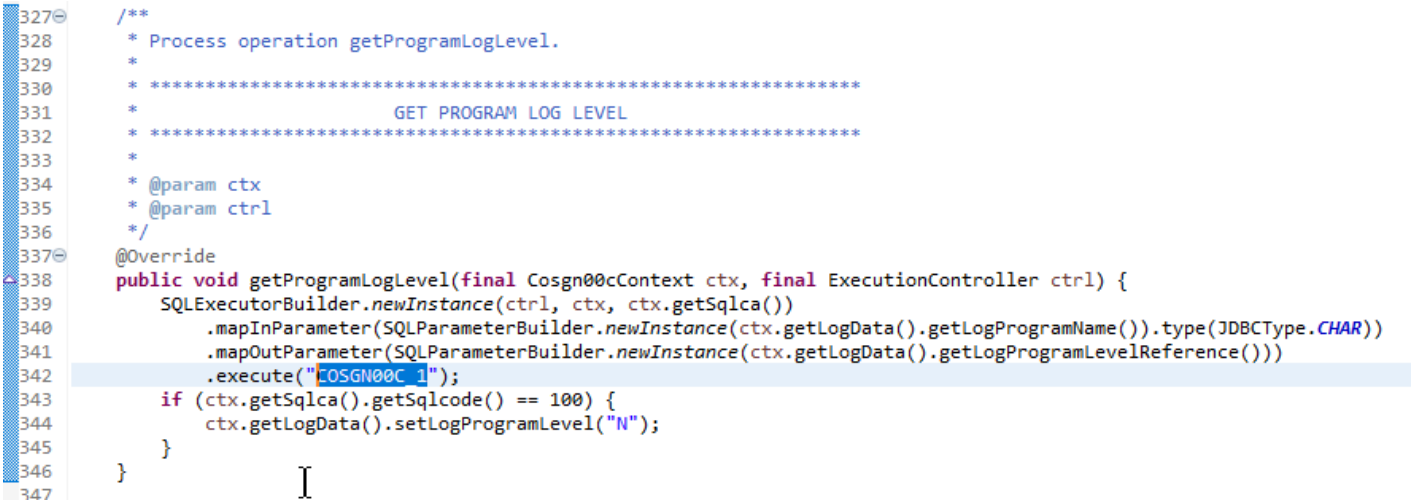
Se a aplicação antiga estava usando consultas SQL, as consultas SQL modernizadas correspondentes foram reunidas em arquivos de propriedades dedicados, com o padrão de nomenclatura program.sql, em que program é o nome do programa que usa essas consultas.



O conteúdo desses arquivos sql é uma coleção de entradas (key=query), em que cada consulta é associada a uma chave exclusiva, que o programa modernizado usa para executar a consulta fornecida:

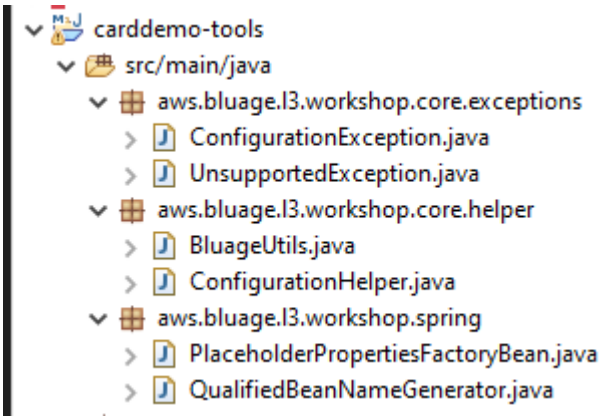


Por exemplo, o programa COSGN00C está executando a consulta com a chave “COSGN00C_1" (a primeira entrada no arquivo sql):



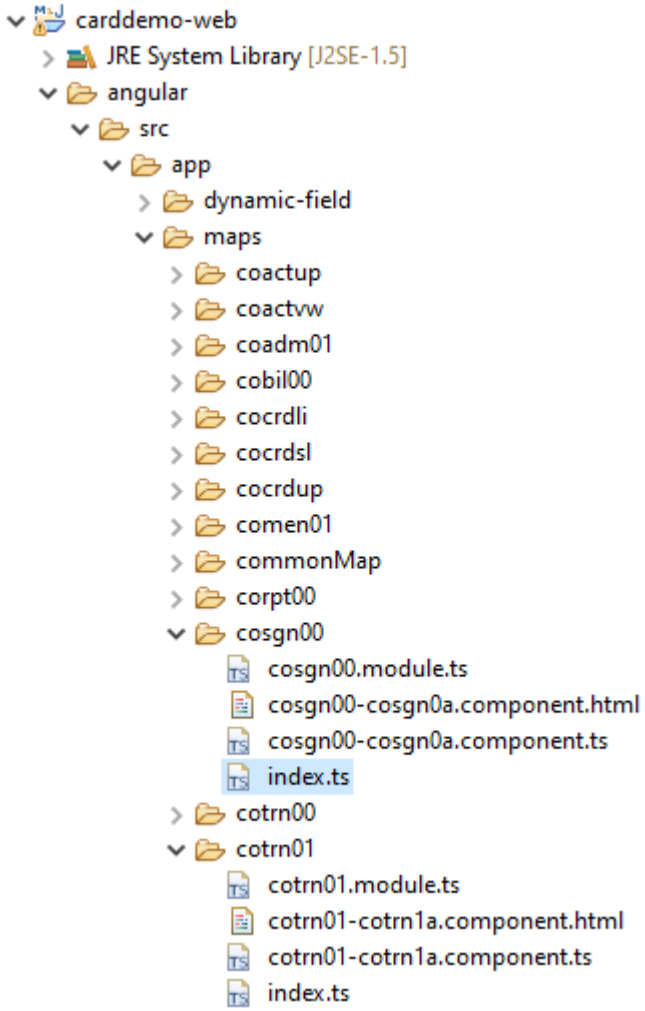
Conteúdo do projeto de utilitários

O projeto de utilitários, cujo nome termina com “-tools”, contém um conjunto de utilitários técnicos, que podem ser usados por todos os outros projetos.



Conteúdo dos projetos web

O projeto web só está presente ao modernizar elementos antigos da interface do usuário. Os elementos modernos da interface de usuário usados para criar o front-end da aplicação modernizada são baseados no Angular. A aplicação de exemplo usado para mostrar os artefatos de modernização é um aplicação COBOL/CICS, executada em um mainframe. O sistema CICS usa MAPs para representar as telas da interface do usuário. Os elementos modernos correspondentes serão, para cada mapa, um arquivo html acompanhado por arquivos Typescript:



O projeto web cuida apenas do aspecto de front-end do aplicativo. O projeto de serviço, que depende dos projetos de utilitários e entidades, fornece os serviços de back-end. O link entre o front-end e o back-end é feito por meio do aplicativo web chamado Gapwalk-Application, que faz parte da distribuição de tempo de execução padrão AWS do Blu Age.

Executando e chamando programas

Em sistemas herdados, os programas são compilados como executáveis autônomos que podem se chamar por meio de um mecanismo CALL, como a instrução COBOL CALL, transmitindo argumentos quando necessário. As aplicações modernizadas oferecem a mesma capacidade, mas usam uma abordagem diferente, porque a natureza dos artefatos envolvidos difere dos antigos.

No lado modernizado, os pontos de entrada do programa são classes específicas que implementam a interface `Program`, são componentes do Spring (@Component) e estão localizados em projetos de serviço, em um pacote chamado *`base package`*`.program`.

Registro de programas

Cada vez que o servidor Tomcat que hospeda aplicações modernizadas é iniciado, a aplicação Springboot do serviço também é iniciado, o que aciona o registro do programa. Um registro dedicado chamado `ProgramRegistry` é preenchido com entradas de programa, cada programa sendo registrado usando seus identificadores, uma entrada por identificador de programa conhecido, o que significa que, se um programa for conhecido por vários identificadores diferentes, o registro conterá tantas entradas quanto identificadores.

O registro de um determinado programa depende da coleção de identificadores retornados pelo método getProgramIdentifiers ():

Cbact04c.java

```java
package aws.bluage.l3.workshop.program;

import aws.bluage.l3.workshop.SpringBootLauncher;

/**
 * Reference the spring application of program CBACT04C.
 *  Provides an access to the contained program for the run unit.
 */
@Component
@Import({
    aws.bluage.l3.workshop.cbact04c.business.context.Cbact04cConfiguration.class,
    aws.bluage.l3.workshop.cbact04c.business.context.Cbact04cContext.class,
    aws.bluage.l3.workshop.cbact04c.service.impl.Cbact04cProcessImpl.class
})
public class Cbact04c implements Program {
    /**
     * Unique identifiers for the contained program.
     */
    private static final Set<String> programIdentifiers = Collections.unmodifiableSet(Stream.of("CBACT04C").collect(Collectors.toSet()));

    /**
     * Main program identifier for the contained program.
     */
    private static final String programIdentifier = "CBACT04C";
    @Autowired
    PlatformTransactionManager transactionManager;

    @Autowired
    Map<String, DataSource> datasources;
    @Autowired
    BeanFactory beanFactory;
    /**
     * {@inheritDoc}
     */
    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        return SpringBootLauncher.getCac();
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public void updateExecutionContext(ExecutionContext executionContext) {
        executionContext.setDatasources(datasources);
        executionContext.setDatabaseSupport(ExecutionContext.DatabaseSupport.POSTGRE);
        executionContext.setSqlcaVersion(ExecutionContext.SqlcaVersion.getEnum("ansi-comp5"));
        executionContext.setTransactionManager(transactionManager);
        executionContext.setUseSQLDateNewParadigm(true);
        executionContext.setUseSQLTrimStringType(false);
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public Set<String> getProgramIdentifiers() {
        return programIdentifiers;
    }
```

Neste exemplo, o programa é registrado uma vez, sob o nome 'CBACT04C' (veja o conteúdo da coleção programIdentifiers). Os logs do tomcat mostram cada registro do programa. O registro do programa depende apenas dos identificadores declarados do programa e não do nome da classe do programa em si (embora normalmente os identificadores do programa e os nomes das classes do programa estejam alinhados).

O mesmo mecanismo de registro se aplica aos programas utilitários trazidos pelos vários aplicativos web utilitários AWS Blu Age, que fazem parte da distribuição de tempo de execução do AWS Blu Age. Por exemplo, a aplicação web Gapwalk-Utility-Pgm fornece os equivalentes funcionais dos

utilitários do sistema z/OS (IDCAMS, ICEGENER, SORT etc.) e pode ser chamado por programas ou scripts modernizados. Todos os programas utilitários disponíveis registrados na inicialização do Tomcat são registrados nos logs do Tomcat.

Registro de scripts e daemons

Um processo de registro semelhante, no momento da inicialização do Tomcat, ocorre para scripts groovy que estão localizados na hierarquia de pastas /src/main/resources/scripts. A hierarquia da pasta de scripts é percorrida e todos os scripts groovy descobertos (exceto o script reservado de funções especiais.groovy) são registrados no `ScriptRegistry`, usando seu nome curto (a parte do nome do arquivo de script localizada antes do primeiro caractere de ponto) como chave para recuperação.

> **Note**
>
> - Se vários scripts tiverem nomes de arquivo que resultarão na produção da mesma chave de registro, somente o mais recente será registrado, substituindo qualquer registro encontrado anteriormente para essa chave específica.
> - Considerando a observação acima, preste atenção ao usar subpastas, pois o mecanismo de registro nivela a hierarquia e pode levar a substituições inesperadas. A hierarquia não conta no processo de registro: normalmente, /scripts/a/myscript.groovy e /scripts/b/myscript.groovy farão com que /scripts/b/myscript.groovy sobrescreva /scripts/a/myscript.groovy.

Os scripts groovy na pasta /src/main/resources/daemons são tratados de forma um pouco diferente. Eles ainda estão registrados como scripts regulares, mas, além disso, são lançados uma vez, diretamente no momento da inicialização do Tomcat, de forma assíncrona.

Depois que os scripts são registrados no `ScriptRegistry`, uma chamada REST pode iniciá-los, usando os endpoints dedicados que a aplicação Gapwalk expõe. Para obter mais informações, consulte a documentação correspondente.

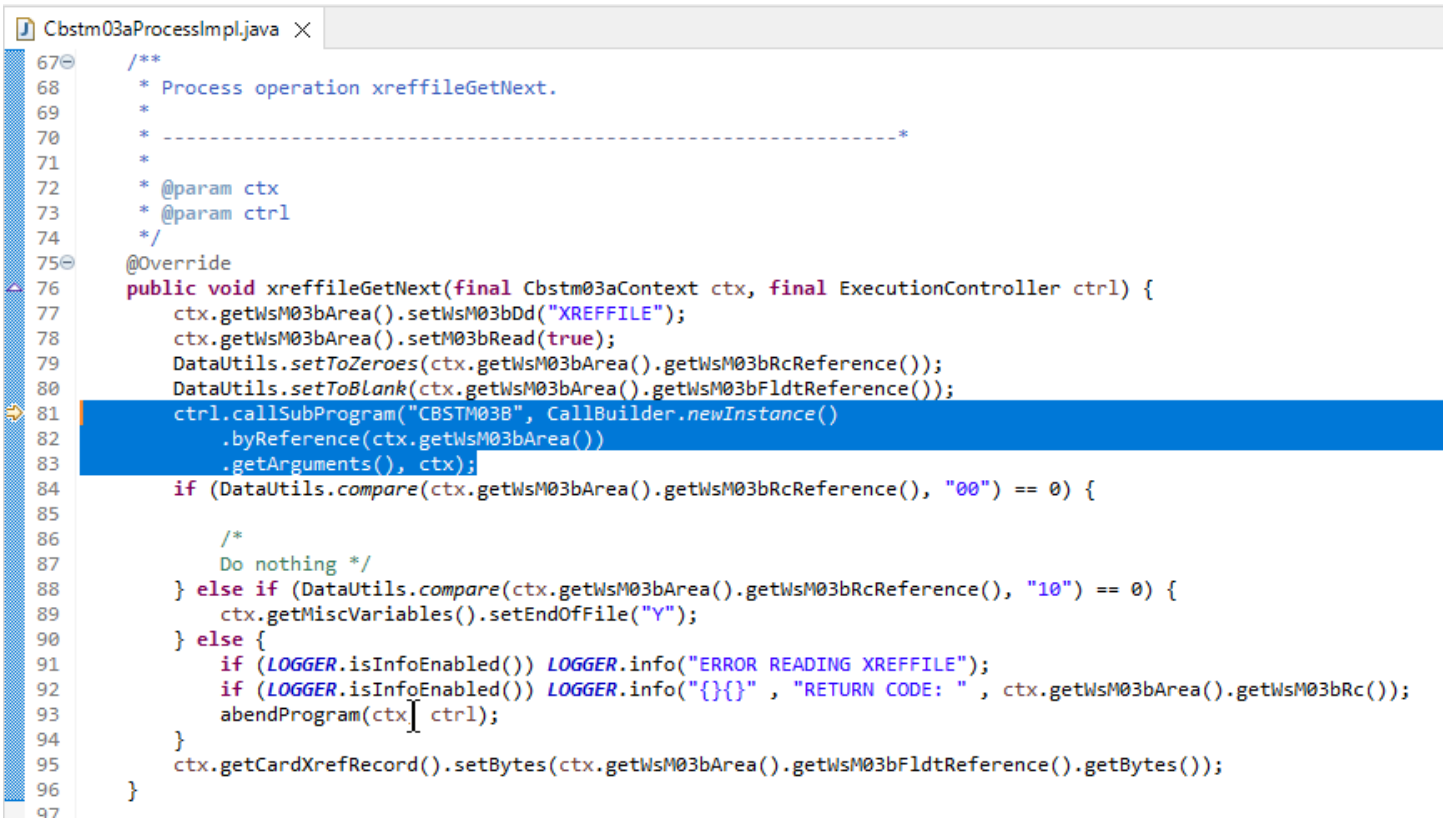
Programas de chamada de programas

Cada programa pode chamar outro programa como subprograma, passando parâmetros para ele. Os programas usam uma implantação da interface `ExecutionController` para fazer isso (na maioria das vezes, isso será uma instância `ExecutionControllerImpl`), junto com um

mecanismo de API fluente chamado de `CallBuilder` para criar os argumentos de chamada do programa.

Todos os métodos de programas usam tanto a `RuntimeContext` quanto an `ExecutionController` como argumentos de método, portanto, um `ExecutionController` está sempre disponível para chamar outros programas.

Veja, por exemplo, o diagrama a seguir, que mostra como o programa CBST03A chama o programa CBST03B como um subprograma, passando parâmetros para ele:



- O primeiro argumento do `ExecutionController.callSubProgram` é um identificador do programa a ser chamado (ou seja, um dos identificadores usados para o registro do programa -- veja os parágrafos acima).
- O segundo argumento, que é o resultado da estrutura no `CallBuilder`, é uma matriz de `Record`, correspondente aos dados passados do chamador para o chamador.
- O terceiro e último argumento é a instância `RuntimeContext` do chamador.

Todos os três argumentos são obrigatórios e não podem ser nulos, mas o segundo argumento pode ser uma matriz vazia.

O chamador só poderá lidar com os parâmetros passados se tiver sido originalmente projetado para isso. Para um programa COBOL antigo, isso significa ter uma seção LINKAGE e uma cláusula USING para que a divisão de procedimentos faça uso dos elementos LINKAGE.

Por exemplo, consulte o arquivo fonte COBOL CBSTM03B.CBL correspondente:

github.com/aws-samples/aws-mainframe-modernization-carddemo/blob/main/app/cbl/CBSTM03B.CBL

```
 98
 99          LINKAGE SECTION.
100          01  LK-M03B-AREA.
101              05  LK-M03B-DD          PIC X(08).
102              05  LK-M03B-OPER        PIC X(01).
103                88  M03B-OPEN       VALUE 'O'.
104                88  M03B-CLOSE      VALUE 'C'.
105                88  M03B-READ       VALUE 'R'.
106                88  M03B-READ-K     VALUE 'K'.
107                88  M03B-WRITE      VALUE 'W'.
108                88  M03B-REWRITE    VALUE 'Z'.
109              05  LK-M03B-RC          PIC X(02).
110              05  LK-M03B-KEY         PIC X(25).
111              05  LK-M03B-KEY-LN      PIC S9(4).
112              05  LK-M03B-FLDT        PIC X(1000).
113
114          PROCEDURE DIVISION USING LK-M03B-AREA.
115
```

Portanto, o programa CBSTM03B usa um único `Record` como parâmetro (uma matriz de tamanho 1). Isso é o que `CallBuilder` está construindo, usando o encadeamento de métodos byReference () e getArguments ().

A classe de API `CallBuilder` fluente tem vários métodos disponíveis para preencher a matriz de argumentos a serem passados para um chamador:

- asPointer (RecordAdaptable): adicione um argumento do tipo ponteiro, por referência. O ponteiro representa o endereço de uma estrutura de dados de destino.
- byReference (RecordAdaptable): adicione um argumento por referência. O chamador verá as modificações que o chamador executa.
- byReference (RecordAdaptable): variante varargs do método anterior.
- byValue (Object): adicione um argumento, transformado em um `Record`, por valor. O chamador não verá as modificações que o chamador executa.

- byValue (RecordAdaptable): igual ao método anterior, mas o argumento está diretamente disponível como a. `RecordAdaptable`
- byValueWithLimites (Object, int, int): adicione um argumento, transformado em a`Record`, extraindo a parte da matriz de bytes definida pelos limites fornecidos, por valor.

Finalmente, o método getArguments coletará todos os argumentos adicionados e os retornará como uma matriz de `Record`.

> **Note**
>
> É responsabilidade do chamador garantir que a matriz de argumentos tenha o tamanho necessário, que os itens estejam devidamente ordenados e compatíveis, em termos de layout de memória com os layouts esperados para os elementos de ligação.

Scripts chamando programas

Chamar programas registrados a partir de scripts groovy requer o uso de uma instância de classe implementando a interface `MainProgramRunner`. Normalmente, a obtenção dessa instância é obtida por meio do ApplicationContext uso do Spring:

REPROC.groovy

```groovy
// Import
import com.netfective.bluage.gapwalk.rt.provider.ScriptRegistry
import com.netfective.bluage.gapwalk.rt.call.MainProgramRunner
import com.netfective.bluage.gapwalk.io.support.FileConfigurationUtils
import com.netfective.bluage.gapwalk.rt.job.support.DefaultJobContext
import com.netfective.bluage.gapwalk.rt.utils.GroovyUtils
import com.netfective.bluage.gapwalk.rt.io.support.FileConfiguration
import com.netfective.bluage.gapwalk.rt.shared.AbendException
import com.netfective.bluage.gapwalk.rt.call.exception.GroovyExecutionException
// Variables
mpr = applicationContext.getBean("com.netfective.bluage.gapwalk.rt.call.ExecutionController", MainProgramRunner.class)
TreeMap mapTransfo = [:]
```

Depois que uma interface `MainProgramRunner` estiver disponível, use o método runProgram para chamar um programa e passar o identificador do programa de destino como parâmetro:

REPROC.groovy

```
50 //*******************************************************************
51 //*                              STEPS                              *
52 //*******************************************************************
53 // STEP PRC001 - PGM - IDCAMS*****************************************************
54 def stepPRC001(Object shell, Map params, Map programResults){
55     shell.with {
56         if (checkValidProgramResults(programResults)) {
57             return execStep("PRC001", "IDCAMS", programResults, {
58                 mpr
59                     .withFileConfigurations(new FileConfigurationUtils()
60                         .systemOut("SYSPRINT")
61                         .output("*")
62                         .build()
63                         .bluesam("FILEIN")
64                         .dataset("NULLFILE")
65                         .disposition("SHR")
66                         .build()
67                         .bluesam("FILEOUT")
68                         .dataset("NULLFILE")
69                         .disposition("SHR")
70                         .build()
71                         .fileSystem("SYSIN")
72                         .path("&CNTLLIB(REPROCT)")
73                         .disposition("SHR")
74                         .build()
75                         .getFileConfigurations(fcmap))
76                     .withParameters(params)
77                     .runProgram("IDCAMS")
78                 })
79         }
80     }
81 }
```

No exemplo anterior, uma etapa de trabalho chama o IDCAMS (programa utilitário de manipulação de arquivos), fornecendo um mapeamento entre as definições reais do conjunto de dados e seus identificadores lógicos.

Ao lidar com conjuntos de dados, os programas antigos geralmente usam nomes lógicos para identificar conjuntos de dados. Quando o programa é chamado a partir de um script, o script deve mapear nomes lógicos com conjuntos de dados físicos reais. Esses conjuntos de dados podem estar no sistema de arquivos, em um armazenamento Blusam ou até mesmo definidos por um fluxo embutido, pela concatenação de vários conjuntos de dados ou pela geração de um GDG.

Use o withFileConfiguration método para criar um mapa lógico para físico dos conjuntos de dados e disponibilizá-lo para o programa chamado.

Escreva seu próprio programa

Escrever seu próprio programa para que scripts ou outros programas modernizados possam ser chamados é uma tarefa comum. Normalmente, em projetos de modernização, você escreve seus próprios programas quando um programa antigo executável é escrito em uma linguagem que o processo de modernização não suporta, ou as fontes foram perdidas (sim, isso pode acontecer) ou o programa é um utilitário cujas fontes não estão disponíveis.

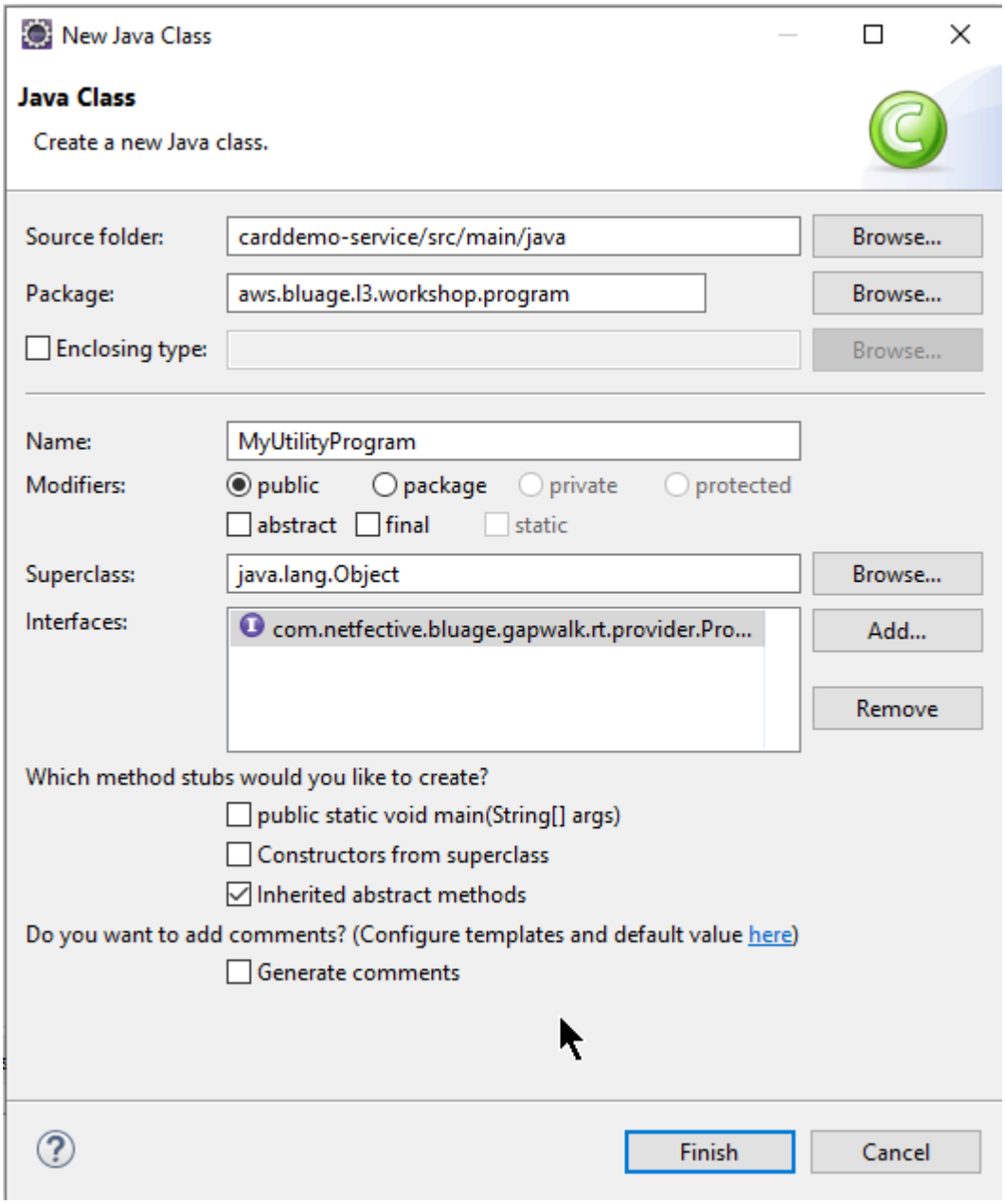
Nesse caso, talvez você precise escrever o programa ausente, em java, sozinho (supondo que você tenha conhecimento suficiente sobre qual deve ser o comportamento esperado do programa, o layout de memória dos argumentos do programa, se houver, e assim por diante). Seu programa java deve estar em conformidade com a mecânica do programa descrita neste documento, para que outros programas e scripts possam executá-lo.

Para garantir que o programa seja utilizável, você deve concluir duas etapas obrigatórias:

- Escreva uma classe que implemente a interface `Program` corretamente, para que ela possa ser registrada e chamada.
- Certifique-se de que seu programa esteja registrado corretamente, para que fique visível em outros programas/scripts.

Escrevendo a implantação do programa

Use seu IDE para criar uma nova classe java que implemente a interface `Program`:



A imagem a seguir mostra o Eclipse IDE, que se encarrega de criar todos os métodos obrigatórios a serem implementados:

MyUtilityProgram.java

```java
package aws.bluage.l3.workshop.program;

import java.util.Set;

public class MyUtilityProgram implements Program {

    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public Set<String> getProgramIdentifiers() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public Context getContext() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public void run(ExecutionController ctrl) {
        // TODO Auto-generated method stub

    }

}
```

Integração com o Spring

Primeiro, a classe deve ser declarada como um componente Spring. Anote a classe com a anotação `@Component`:

```java
import org.springframework.context.ConfigurableApplicationContext;
import org.springframework.stereotype.Component;

import com.netfective.bluage.gapwalk.rt.call.ExecutionController;
import com.netfective.bluage.gapwalk.rt.context.Context;
import com.netfective.bluage.gapwalk.rt.provider.Program;

import aws.bluage.l3.workshop.SpringBootLauncher;

@Component
public class MyUtilityProgram implements Program {
```

Em seguida, implemente os métodos necessários corretamente. No contexto desse exemplo, adicionamos o `MyUtilityProgram` ao pacote que já contém todos os programas modernizados.

Esse posicionamento permite que o programa use o aplicativo Springboot existente para fornecer o necessário `ConfigurableApplicationContext` para a implementação do método: getSpringApplication

```
public class MyUtilityProgram implements Program {

    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        return SpringBootLauncher.getCac();
    }
```

Você pode escolher um local diferente para seu próprio programa. Por exemplo, você pode localizar o determinado programa em outro projeto de serviço dedicado. Certifique-se de que o projeto de serviço fornecido tenha seu próprio aplicativo Springboot, o que possibilita recuperar o ApplicationContext (que deveria ser um). `ConfigurableApplicationContext`

Dando uma identidade ao programa

Para ser chamado por outros programas e scripts, o programa deve receber pelo menos um identificador, que não deve colidir com nenhum outro programa registrado existente no sistema. A escolha do identificador pode ser motivada pela necessidade de cobrir a substituição de um programa antigo existente; nesse caso, você precisará usar o identificador esperado, conforme encontrado nas ocorrências de CALL encontradas em todos os programas legados. A maioria dos identificadores de programas tem 8 caracteres em sistemas herdados.

Criar um conjunto não modificável de identificadores no programa é uma forma de fazer isso. O exemplo a seguir mostra a escolha de “MYUTILPG” como identificador único:

```
@Component
public class MyUtilityProgram implements Program {

    /**
     * Unique identifiers for the contained program.
     */
    private static final Set<String> programIdentifiers = Collections.unmodifiableSet(Stream.of("MYUTILPG").collect(Collectors.toSet()));


    public ConfigurableApplicationContext getSpringApplication() {

    @Override
    public Set<String> getProgramIdentifiers() {
        return programIdentifiers;
    }
```

Associar o programa a um contexto

O programa precisa de uma instância `RuntimeContext` complementar. Para programas modernizados, o AWS Blu Age gera automaticamente o contexto complementar, usando as estruturas de dados que fazem parte do programa legado.

Se você estiver escrevendo seu próprio programa, também deverá escrever o contexto complementar.

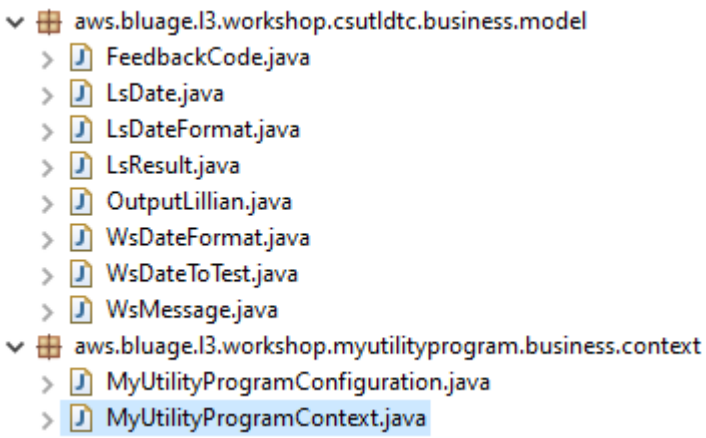
Referindo-se a Aulas relacionadas ao programa, você pode ver que um programa requer pelo menos duas classes complementares:

- uma classe de configuração.
- uma classe de contexto que usa a configuração.

Se o programa utilitário usa alguma estrutura de dados extra, ela também deve ser escrita e usada pelo contexto.

Essas classes devem estar em um pacote que faça parte de uma hierarquia de pacotes que será verificada na inicialização da aplicação, para garantir que o componente de contexto e a configuração sejam tratados pela estrutura Spring.

Vamos escrever uma configuração e um contexto mínimos, no pacote *`base package`*`.myutilityprogram.business.context`, recém-criado no projeto de entidades:



Aqui está o conteúdo da configuração. Ele está usando uma configuração similar a outros programas -- modernizados -- nas proximidades. Você provavelmente precisará personalizar isso de acordo com suas necessidades específicas.

MyUtilityProgramConfiguration.java

```java
package aws.bluage.l3.workshop.myutilityprogram.business.context;

import java.nio.charset.Charset;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Lazy;

import com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration;
import com.netfective.bluage.gapwalk.datasimplifier.configuration.ConfigurationBuilder;

/**
 * Creates Datasimplifier configuration for the MyUtilityProgram context.
 */
@org.springframework.context.annotation.Configuration
@Lazy
public class MyUtilityProgramConfiguration {

    @Bean(name = "MyUtilityProgramContextConfiguration")
    public Configuration configuration() {
        return new ConfigurationBuilder()
                .encoding(Charset.forName("CP1047"))
                .humanReadableEncoding(Charset.forName("ISO-8859-15"))
                .initDefaultByte(0)
                .build();
    }
}

```

Observações:

- A convenção geral de nomenclatura é ProgramNameConfiguração.
- Ele deve usar as anotações @org.springframework.context.annotation.Configuration e @Lazy.
- O nome do bean geralmente segue a ProgramNameContextConfiguration convenção, mas isso não é obrigatório. Certifique-se de evitar colisões de nomes de beans em todo o projeto.
- O único método a ser implementado deve retornar um objeto `Configuration`. Use a API `ConfigurationBuilder` fluente para ajudar a criar uma.

E o contexto associado:

MyUtilityProgramContext.java

```java

import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.context.annotation.Lazy;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Component;

import com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration;
import com.netfective.bluage.gapwalk.rt.context.RuntimeContext;

@Component("aws.bluage.l3.workshop.mytuilityprogram.business.context.MyUtilityProgramContext")
@Lazy
@Scope("prototype")
public class MyUtilityProgramContext extends RuntimeContext{

    protected MyUtilityProgramContext(@Qualifier("MyUtilityProgramContextConfiguration") Configuration configuration) {
        super(configuration);
    }

    @Override
    public void cleanUp() {
        // TODO implement clean-up of associated data structures if any
    }

    @Override
    protected void doReset() {
        // TODO implement reset of associated data structures if any
    }

}

```

Observações

- A classe de contexto deve estender uma implantação de interface `Context` existente (`RuntimeContext` ou `JicsRuntimeContext`, que é `RuntimeContext` aprimorado com itens específicos do JICS).
- A convenção geral de nomenclatura é ProgramNameContext.
- Você deve declará-lo como um componente de protótipo e usar a anotação @Lazy.
- O construtor se refere à configuração associada, usando a anotação @Qualifier para direcionar a classe de configuração adequada.
- Se o programa utilitário usar algumas estruturas de dados extras, elas devem ser:
  - Escrito e adicionado ao pacote *`base package`*`.business.model`.
  - referenciado no contexto. Dê uma olhada em outras classes de contexto existentes para ver como referenciar classes de estruturas de dados e adaptar os métodos de contexto (constructor/clean-up/reset) conforme necessário.

Agora que um contexto dedicado está disponível, deixe o novo programa usá-lo:

MyUtilityProgram.java

```java
import aws.bluage.l3.workshop.SpringBootLauncher;

@Component
@Import({
    aws.bluage.l3.workshop.myutilityprogram.business.context.MyUtilityProgramConfiguration.class,
    aws.bluage.l3.workshop.myutilityprogram.business.context.MyUtilityProgramContext.class
})
public class MyUtilityProgram implements Program {

    @Autowired
    BeanFactory beanFactory;

    /**
     * Unique identifiers for the contained program.
     */
    private static final Set<String> programIdentifiers = Collections.unmodifiableSet(Stream.of("MYUTILPG").collect(Collectors.toSet()));

    private static final String programIdentifier = "MYUTILPG";

    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        return SpringBootLauncher.getCac();
    }

    @Override
    public Set<String> getProgramIdentifiers() {
        return programIdentifiers;
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public String getProgramMainIdentifier() {
        return programIdentifier;
    }


    @Override
    public Context getContext() {
        return ProgramContextStore.getOrCreate(
        getProgramMainIdentifier(),
        aws.bluage.l3.workshop.myutilityprogram.business.context.MyUtilityProgramContext.class,
        beanFactory);
    }

```

Observações:

- O método getContext deve ser implementado estritamente conforme mostrado, usando uma delegação ao getOrCreate método da `ProgramContextStore` classe e ao Spring conectado automaticamente. `BeanFactory` Um único identificador de programa é usado para armazenar o contexto do programa no `ProgramContextStore`; esse identificador é referenciado como sendo o “identificador principal do programa”.
- A configuração complementar e as classes de contexto devem ser referenciadas usando a anotação spring `@Import`.

Implementando a lógica de negócios

Quando o esqueleto do programa estiver concluído, implemente a lógica de negócios para o novo programa utilitário.

Faça isso no método `run` do programa. Esse método será executado sempre que o programa for chamado, seja por outro programa ou por um script.

Feliz codificação!

Gerenciando o registro do programa

Por fim, verifique se o novo programa está registrado corretamente no `ProgramRegistry`. Se você adicionou o novo programa ao pacote que já contém outros programas, não há mais nada a ser feito. O novo programa é selecionado e registrado em todos os programas vizinhos na inicialização da aplicação.

Se você escolheu outro local para o programa, verifique se o programa está registrado corretamente na inicialização do Tomcat. Para se inspirar sobre como fazer isso, veja o método de inicialização das SpringbootLauncher classes geradas no (s) projeto (s) de serviço (consulteConteúdo do projeto de serviço).

Verifique os logs de inicialização do Tomcat. Cada registro do programa é registrado. Se o seu programa for registrado com sucesso, você encontrará a entrada de log correspondente.

Quando tiver certeza de que seu programa está registrado corretamente, você pode começar a iterar na codificação da lógica de negócios.

Mapeamentos de nomes totalmente qualificados

Esta seção contém listas do AWS Blu Age e de mapeamentos de nomes totalmente qualificados de terceiros para uso em seus aplicativos modernizados.

AWS Mapeamentos de nomes totalmente qualificados da Blu Age

Nome curto	Nome totalmente qualificado
`CallBuilder`	`com.netfective.bluage.gapwalk.runtime.statements.CallBuilder`
`Configuration`	`com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration`

Nome curto	Nome totalmente qualificado
ConfigurationBuilder	com.netfective.bluage.gapwalk.datasimplifier.configuration.ConfigurationBuilder
ExecutionController	com.netfective.bluage.gapwalk.rt.call.ExecutionController
ExecutionControllerImpl	com.netfective.bluage.gapwalk.rt.call.internal.ExecutionControllerImpl
File	com.netfective.bluage.gapwalk.rt.io.File
MainProgramRunner	com.netfective.bluage.gapwalk.rt.call.MainProgramRunner
Program	com.netfective.bluage.gapwalk.rt.provider.Program
ProgramContextStore	com.netfective.bluage.gapwalk.rt.context.ProgramContextStore
ProgramRegistry	com.netfective.bluage.gapwalk.rt.provider.ProgramRegistry
Record	com.netfective.bluage.gapwalk.datasimplifier.data.Record
RecordEntity	com.netfective.bluage.gapwalk.datasimplifier.entity.RecordEntity
RuntimeContext	com.netfective.bluage.gapwalk.rt.context.RuntimeContext

Nome curto	Nome totalmente qualificado
SimpleStateMachineController	com.netfective.bluage.gapwalk.rt.statemachine.SimpleStateMachineController
StateMachineController	com.netfective.bluage.gapwalk.rt.statemachine.StateMachineController
StateMachineRunner	com.netfective.bluage.gapwalk.rt.statemachine.StateMachineRunner

Mapeamentos de nomes totalmente qualificados de terceiros

Nome curto	Nome totalmente qualificado
@Autowired	org.springframework.beans.factory.annotation.Autowired
@Bean	org.springframework.context.annotation.Bean
BeanFactory	org.springframework.beans.factory.BeanFactory
@Component	org.springframework.stereotype.Component
ConfigurableApplicationContext	org.springframework.context.ConfigurableApplicationContext
@Import	org.springframework.context.annotation.Import
@Lazy	org.springframework.context.annotation.Lazy

Simplificador de dados

Em sistemas mainframe e midrange (referidos no tópico a seguir como sistemas “antigos”), as linguagens de programação usadas com frequência, como COBOL, PL/I ou RPG, fornecem acesso de baixo nível à memória. Esse acesso se concentra no layout de memória acessado por meio de tipos nativos, como zoneados, compactados ou alfanuméricos, possivelmente também agregados por meio de grupos ou matrizes.

Uma mistura de acessos a uma determinada parte da memória, por meio de campos digitados e como acesso direto a bytes (memória bruta), coexiste em um determinado programa. Por exemplo, os programas COBOL passarão argumentos aos chamadores como conjuntos contíguos de bytes (LINKAGE) ou lerão e gravarão dados de arquivos da mesma maneira (registros), enquanto interpretam esses intervalos de memória com campos digitados organizados em cadernos.

Essas combinações de acesso bruto e estruturado à memória, a dependência de um layout de memória preciso em nível de byte e tipos antigos, como zoneados ou compactados, são recursos que não estão disponíveis de forma nativa nem facilmente no ambiente de programação Java.

Como parte da solução AWS Blu Age para modernizar programas legados para Java, a biblioteca Data Simplifier fornece essas construções para programas Java modernizados e as expõe de uma forma que seja o mais familiar possível para os desenvolvedores Java (getters/setters, matrizes de bytes, baseadas em classes). É uma dependência central do código Java modernizado gerado a partir desses programas.

Para simplificar, a maioria das explicações a seguir é baseada em estruturas COBOL, mas você pode usar a mesma API para modernização do layout de dados PL1 e RPG, já que a maioria dos conceitos é semelhante.

Tópicos

Classes principais

Para facilitar a leitura, este documento usa os nomes abreviados Java das interfaces e classes da API AWS Blu Age. Para ter mais informações, consulte FQN dos tipos de Java discutidos.

Representação de baixo nível

No nível mais baixo, a memória (uma faixa contígua de bytes acessível de forma rápida e aleatória) é representada pela interface `Record`. Essa interface é essencialmente uma abstração de uma matriz de bytes de tamanho fixo. Dessa forma, ele fornece setters e getters capazes de acessar ou modificar os bytes subjacentes.

Representação de dados estruturados

Para representar dados estruturados, como “01 itens de dados” ou “01 cadernos”, conforme encontrado em COBOL DATA DIVISION, são usadas subclasses da `RecordEntity` classe. Normalmente, eles não são escritos à mão, mas gerados pelas ferramentas de modernização do AWS Blu Age a partir das construções legadas correspondentes. Ainda é útil conhecer sua estrutura principal e sua API, para que você possa entender como o código em um programa modernizado as usa. No caso do COBOL, esse código é gerado em Java a partir de sua DIVISÃO DE PROCEDIMENTOS.

O código gerado representa cada “01 item de dados” com uma `RecordEntity` subclasse; cada campo elementar ou agregado que o compõe é representado como um campo Java privado, organizado como uma árvore (cada item tem um pai, exceto o raiz).

Para fins de ilustração, aqui está um exemplo de item de dados COBOL, seguido pelo código correspondente gerado pelo AWS Blu Age que o moderniza:

```
01 TST2.
 02 FILLER PIC X(4).
 02 F1     PIC 9(2) VALUE 42.
 02 FILLER PIC X.
 02        PIC 9(3) VALUE 123.
 02 F2     PIC X VALUE 'A'.
```

```
public class Tst2 extends RecordEntity {

    private final Group root = new Group(getData()).named("TST2");
    private final Filler filler = new Filler(root,new AlphanumericType(4));
    private final Elementary f1 = new Elementary(root,new ZonedType(2, 0, false),new
 BigDecimal("42")).named("F1");
    private final Filler filler1 = new Filler(root,new AlphanumericType(1));
    private final Filler filler2 = new Filler(root,new ZonedType(3, 0, false),new
 BigDecimal("123"));
```

```
    private final Elementary f2 = new Elementary(root,new
 AlphanumericType(1),"A").named("F2");


    /**
     * Instantiate a new Tst2 with a default record.
     * @param configuration the configuration
     */
    public Tst2(Configuration configuration) {
        super(configuration);
        setupRoot(root);
    }
    /**
     * Instantiate a new Tst2 bound to the provided record.
     * @param configuration the configuration
     * @param record the existing record to bind
     */
    public Tst2(Configuration configuration, RecordAdaptable record) {
        super(configuration);
        setupRoot(root, record);
    }

    /**
     * Gets the reference for attribute f1.
     * @return the f1 attribute reference
     */
    public ElementaryRangeReference getF1Reference() {
        return f1.getReference();
    }

    /* *
     * Getter for f1 attribute.
     * @return f1 attribute
     */
    public int getF1() {
        return f1.getValue();
    }

    /**
     * Setter for f1 attribute.
     * @param f1 the new value of f1
     */
    public void setF1(int f1) {
        this.f1.setValue(f1);
```

```
    }
    /**
     * Gets the reference for attribute f2.
     * @return the f2 attribute reference
     */
    public ElementaryRangeReference getF2Reference() {
        return f2.getReference();
    }

    /**
     * Getter for f2 attribute.
     * @return f2 attribute
     */
    public String getF2() {
        return f2.getValue();
    }

    /**
     * Setter for f2 attribute.
     * @param f2 the new value of f2
     */
    public void setF2(String f2) {
        this.f2.setValue(f2);
    }
}
```

Campos elementares

Os campos de classe `Elementary` (ou `Filler`, quando não nomeados) representam uma “folha” da estrutura de dados antiga. Eles estão associados a uma extensão contígua de bytes subjacentes (“intervalo”) e geralmente têm um tipo (possivelmente parametrizado) que expressa como interpretar e modificar esses bytes (“decodificando” e “codificando”, respectivamente, um valor de/para uma matriz de bytes).

Todos os tipos elementares são subclasses de `RangeType`. Os tipos comuns são:

Tipo COBOL	Tipo de simplificador de dados
`PIC X(n)`	`AlphanumericType`
`PIC 9(n)`	`ZonedType`

Tipo COBOL	Tipo de simplificador de dados
`PIC 9(n) COMP-3`	`PackedType`
`PIC 9(n) COMP-5`	`BinaryType`

Agregar os campos

Os campos agregados organizam o layout de memória de seus conteúdos (outros agregados ou campos elementares). Eles próprios não têm um tipo elementar.

Campos de `Group` representam campos contíguos na memória. Cada um dos campos contidos é disposto na mesma ordem na memória, o primeiro campo está em deslocamento em `0` relação à posição do campo do grupo na memória, o segundo campo em deslocamento`0 + (size in bytes of first field)`, etc. Eles são usados para representar sequências de campos COBOL sob o mesmo campo contendo.

Campos de `Union` representam vários campos acessando a mesma memória. Cada um dos campos contidos é disposto em deslocamento em `0` relação à posição do campo de união na memória. Eles são usados, por exemplo, para representar a estrutura COBOL “REDEFINES” (os primeiros filhos da União sendo o item de dados redefinido, os segundos filhos sendo sua primeira redefinição, etc.).

Os campos de matriz (subclasses de `Repetition`) representam a repetição, na memória, do layout de seu campo filho (seja um agregado em si ou um item elementar). Eles apresentam um determinado número desses layouts infantis na memória, cada um em offset `index * (size in bytes of child)`. Eles são usados para representar estruturas COBOL “OCCURS”.

Primitivos

Em alguns casos de modernização, “Primitivos” também podem ser usados para apresentar itens de dados “raiz” independentes. Eles são muito semelhantes em uso`RecordEntity`, mas não vêm dele, nem são baseados no código gerado. Em vez disso, eles são fornecidos diretamente pelo tempo de execução do AWS Blu Age como subclasses da `Primitive` interface. Exemplos dessas aulas fornecidas são `Alphanumeric` ou `ZonedDecimal`.

Vinculação e acesso a dados

A associação entre dados estruturados e dados subjacentes pode ser feita de várias maneiras.

Uma interface importante para esse propósito é `RecordAdaptable` a que é usada para obter `Record` uma “visão gravável” dos dados `RecordAdaptable` subjacentes. Como veremos abaixo, várias classes são implementadas`RecordAdaptable`. Reciprocamente, as APIs e o código do AWS Blu Age que manipulam memória de baixo nível (como argumentos de programas, registros de E/S de arquivos, área de comunicação do CICS, memória alocada...) geralmente esperam um como um identificador para essa memória. `RecordAdaptable`

No caso de modernização do COBOL, a maioria dos itens de dados está associada à memória, que será corrigida durante a vida útil da execução do programa correspondente. Para isso, `RecordEntity` as subclasses são instanciadas uma vez em um objeto pai gerado (o contexto do programa) e se encarregarão de instanciar seu subjacente`Record`, com base no tamanho do byte. `RecordEntity`

Em outros casos de COBOL, como associar elementos LINKAGE a argumentos do programa ou modernizar a estrutura SET ADDRESS OF, uma instância `RecordEntity` deve ser associada a um `RecordAdaptable` fornecido. Para isso, existem dois mecanismos:

- se a instância `RecordEntity` já existir, o método `RecordEntity.bind(RecordAdaptable)` (herdado de `Bindable`) pode ser usado para fazer essa instância “apontar” para `RecordAdaptable`. Qualquer getter ou setter chamado no `RecordEntity` será então apoiado (leitura ou gravação de bytes) pelos bytes `RecordAdaptable` subjacentes.
- se `RecordEntity` for para ser instanciado, um construtor gerado aceitando a estará disponível `RecordAdaptable`.

Por outro lado, os dados `Record` atualmente vinculados aos dados estruturados podem ser acessados. Para isso, `RecordEntity` implementos`RecordAdaptable`, portanto, `getRecord()` podem ser chamados em qualquer instância desse tipo.

Finalmente, muitos verbos COBOL ou CICS exigem acesso a um único campo, para fins de leitura ou escrita. A `RangeReference` classe é usada para representar esse acesso. Suas instâncias podem ser obtidas a partir de métodos `getXXXReference()` gerados de `RecordEntity` (XXX sendo o campo acessado) e passadas para métodos de runtime. `RangeReference` normalmente é usado para acessar todo o `RecordEntity` ou `Group`, enquanto sua subclasse `ElementaryRangeReference` representa acessos aos campos `Elementary`.

Observe que a maioria das observações acima se aplica às `Primitive` subclasses, pois elas se esforçam para implementar um comportamento semelhante ao fornecido pelo `RecordEntity` tempo de execução do AWS Blu Age (em vez do código gerado). Para este propósito, todas as subclasses

de `Primitive` implementam as interfaces `RecordAdaptable`, `ElementaryRangeReference` e `Bindable` de forma a serem utilizáveis no lugar das subclasses `RecordEntity` e dos campos elementares.

FQN dos tipos de Java discutidos

A tabela a seguir mostra os nomes totalmente qualificados dos tipos Java discutidos nesta seção.

Nome curto	Nome totalmente qualificado
`Alphanumeric`	`com.netfective.bluage.gapwalk.datasimplifier.elementary.Alphanumeric`
`AlphanumericType`	`com.netfective.bluage.gapwalk.datasimplifier.metadata.type.AlphanumericType`
`BinaryType`	`com.netfective.bluage.gapwalk.datasimplifier.metadata.type.BinaryType`
`Bindable`	`com.netfective.bluage.gapwalk.datasimplifier.data.Bindable`
`Elementary`	`com.netfective.bluage.gapwalk.datasimplifier.data.structure.Elementary`
`ElementaryRangeReference`	`com.netfective.bluage.gapwalk.datasimplifier.entity.ElementaryRangeReference`
`Filler`	`com.netfective.bluage.gapwalk.datasimplifier.data.structure.Filler`

Nome curto	Nome totalmente qualificado
Group	com.netfective.bluage.gapwalk.datasimplifier.data.structure.Group
PackedType	com.netfective.bluage.gapwalk.datasimplifier.metadata.type.PackedType
Primitive	com.netfective.bluage.gapwalk.datasimplifier.elementary.Primitive
RangeReference	com.netfective.bluage.gapwalk.datasimplifier.entity.RangeReference
RangeType	com.netfective.bluage.gapwalk.datasimplifier.metadata.type.RangeType
Record	com.netfective.bluage.gapwalk.datasimplifier.data.Record
RecordAdaptable	com.netfective.bluage.gapwalk.datasimplifier.data.RecordAdaptable
RecordEntity	com.netfective.bluage.gapwalk.datasimplifier.entity.RecordEntity
Repetition	com.netfective.bluage.gapwalk.datasimplifier.data.structure.Repetition

Nome curto	Nome totalmente qualificado
`Union`	`com.netfective.bluage.gapwalk.datasimplifier.data.structure.Union`
`ZonedDecimal`	`com.netfective.bluage.gapwalk.datasimplifier.elementary.ZonedDecimal`
`ZonedType`	`com.netfective.bluage.gapwalk.datasimplifier.metadata.type.ZonedType`

AWS Arquivos de configuração e configuração do Blu Age Runtime

O AWS Blu Age Runtime e o código do cliente são aplicativos da web que usam a estrutura Spring Boot. Ele aproveita os recursos do Spring para fornecer configuração, com vários locais possíveis e regras de precedência. Também existem regras de precedência semelhantes para fornecer muitos outros arquivos, como scripts groovy, sql etc.

O AWS Blu Age Runtime também contém aplicativos web opcionais adicionais, que podem ser ativados se necessário.

Tópicos

- Noções básicas de configuração de aplicações
- Precedência da aplicação
- JNDI para bancos de dados
- Usando AWS segredos
- Outros arquivos (groovy, sql, etc.)
- Aplicação web adicional
- Habilitando propriedades
- Configurando a segurança para aplicativos Gapwalk

Noções básicas de configuração de aplicações

A forma padrão de lidar com a configuração do aplicativo é por meio do uso de arquivos YAML dedicados a serem fornecidos na `config` pasta do servidor do aplicativo. Há dois arquivos principais de configuração do YAML:

- `application-main.yaml`
- `application-`*`profile`*`.yaml` (em que o valor *`profile`* é configurado durante a geração da aplicação).

O primeiro arquivo configura a estrutura, ou seja `Gapwalk-application.war`, enquanto o segundo é para opções adicionais específicas para a aplicação cliente. Isso funciona com o uso de perfis de primavera: a aplicação Gapwalk usa o perfil `main`, enquanto a aplicação cliente usa o perfil *`profile`*.

O exemplo a seguir mostra um arquivo YAML principal típico.

```
########################################
##### JICS datasource configuration #####
########################################
datasource:
  jicsDs:
    driver-class-name : org.postgresql.Driver
    url: jdbc:postgresql://localhost/jics
    username: jics
    password: jics
    type : org.postgresql.ds.PGSimpleDataSource

#####################################################
##### Embedded Bluesam datasource configuration #####
#####################################################
  bluesamDs :
    driver-class-name : org.postgresql.Driver
    url : jdbc:postgresql://localhost/bluesam
    username : bluesam
    password : bluesam
    type : org.postgresql.ds.PGSimpleDataSource

########################################
##### Embedded Bluesam configuration #####
########################################
bluesam :
  remote : false
  cache : ehcache
  persistence : pgsql #pgsql, mssql, xodus...
  ehcache:
    resource-pool:
      size: 4GB
  write-behind:
```

O exemplo a seguir mostra um arquivo YAML típico do cliente.

```
# Logback context logger integration.
logging.config : classpath:logback-[redacted].xml
# Limits Spring logger output.
logging.level.org.springframework.beans.factory.support.DefaultListableBeanFactory : WARN
logging.level.org.springframework.statemachine : WARN
# If the datasource support mode is not static-xa, spring JTA transactions autoconfiguration must me disabled
spring.jta.enabled : false

spring:
  aws:
    client:
      datasources:
        names: primary
        primary:
          secret: arn:aws:secretsmanager:[redacted]

spring.jta.atomikos.datasource.primary.unique-resource-name: primary
spring.jta.atomikos.datasource.primary.xa-data-source-class-name: org.postgresql.xa.PGXADataSource
spring.jta.atomikos.datasource.primary.maxPoolSize: 20
spring.jta.atomikos.datasource.primary.autoCommit: false
```

Para obter informações sobre o conteúdo dos arquivos YAML, consulte Habilitando propriedades.

Precedência da aplicação

Para esses arquivos de configuração, as regras de precedência do Spring se aplicam. Notavelmente:

- O arquivo `application-main` YAML aparece no arquivo war principal do Gapwalk com valores padrão, e o da `config` pasta o substitui.
- O mesmo deve ser feito para a configuração da aplicação cliente.
- Parâmetros adicionais podem ser transmitidos na linha de comando no momento da inicialização do servidor. Eles substituiriam os do YAML.

Para obter mais informações, consulte a documentação oficial do Spring Boot.

JNDI para bancos de dados

A configuração do banco de dados pode ser fornecida com JNDI no arquivo context.xml no Tomcat. Qualquer configuração desse tipo substituiria a YAML. Mas preste atenção, pois usar isso não permitirá agrupar suas credenciais em um gerenciador secreto (veja abaixo).

O exemplo a seguir mostra exemplos de configurações para JICS e BluSam bancos de dados.

```
<Resource auth="Container" driverClassName="org.postgresql.Driver" initialSize="0"
 maxIdle="5"
```

```
   maxOpenPreparedStatements="-1" maxTotal="10" maxWaitMillis="-1" name="jdbc/jics"
   poolPreparedStatements="true" testOnBorrow="false" type="javax.sql.DataSource"
   url="jdbc:postgresql://XXXX.rds.amazonaws.com:5432/XXXX" username="XXXX"
password="XXXX" />
```

jdbc/jics

Seria `jdbc/jics` para o banco de dados JICS e `jdbc/bluesam` (preste atenção ao 'e') para o banco de dados blusam.

url="jdbc:postgresql://XXXX.rds.amazonaws.com:5432/XXXX" username="XXXX" password="XXXX"

O URL, nome de usuário e senha do banco de dados.

Usando AWS segredos

Algumas das configurações de recursos que contêm credenciais podem ser ainda mais protegidas usando segredos da AWS . A ideia é armazenar dados críticos em um AWS segredo e ter uma referência ao segredo na configuração do YAML para que o conteúdo secreto seja escolhido rapidamente na inicialização do tomcat.

Segredos para Aurora

A configuração do banco de dados Aurora (para jics, blusam, banco de dados do cliente etc.) usará o segredo do banco de dados integrado, que preencherá automaticamente todos os campos relevantes do banco de dados correspondente.

> **Note**
>
> A chave `dbname` é opcional, dependendo da configuração do seu banco de dados, ela entrará no segredo ou não. Você pode adicioná-lo manualmente ou fornecendo o nome ao arquivo YAML.

Outros segredos

Outros segredos são para recursos que têm uma única senha (principalmente caches redis protegidos por senha). Nesse caso, o outro tipo de segredo deve ser usado, com uma única chave `password`.

Referências YAML a segredos

Eles `application-main.yaml` podem referenciar o ARN secreto para vários recursos. Os mais importantes são:

- Credenciais do banco de dados JICS com `spring.aws.jics.db.secret`
- Credenciais do JICS TS Queues Redis com `spring.aws.client.jics.queues.ts.redis.secret`
- Credenciais do banco de dados Blusam com `spring.aws.client.bluesam.db.secret`
- Senha de cache Blusam com `spring.aws.client.bluesam.redis.secret`
- Blusam bloqueia a senha do cache com `spring.aws.client.bluesam.locks.redis.secret`

O exemplo a seguir mostra como declarar esses segredos em um arquivo YAML.

```
spring:
  aws:
    client:
      bluesam:
        locks:
          redis:
            secret: arn:aws:secretsmanager:XXXX
        db:
          dbname: bluesam
          secret: arn:aws:secretsmanager:XXXX
        redis:
          secret: arn:aws:secretsmanager:XXXX
      jics:
         queues:
          ts:
           redis:
            secret: arn:aws:secretsmanager:XXXX
    jics:
      db:
        secret: arn:aws:secretsmanager:XXXX
```

dbname: bluesam

Neste exemplo, o nome do banco de dados não está no segredo e, em vez disso, é fornecido aqui.

O cliente `application-profile.yaml` pode referenciar o ARN secreto do banco de dados do cliente. Isso requer uma propriedade adicional para listar as fontes de dados, mostradas no exemplo abaixo:

```
spring:
  aws:
    client:
      datasources:
        names: primary,host
        primary:
          secret: arn:aws:secretsmanager:XXXX
        host:

          secret: arn:aws:secretsmanager:XXXX
```

nomes: primário, host

Um exemplo com duas fontes de dados de clientes chamadas primary e host, cada uma com seu banco de dados e credenciais.

nome do banco de dados: mydb

Neste exemplo, o nome do banco de dados “host” não está no segredo e é fornecido aqui, enquanto que para o banco de dados “primário” ele está no segredo.

Você também pode configurar `application-main.yaml` para recuperar o segredo do cliente OAuth2 AWS Secrets Manager especificando o provedor e o ARN. O valor padrão para a propriedade do provedor é Amazon Cognito. Veja a seguir um exemplo de configuração para o provedor OAuth2 Keycloak:

```
spring:
  aws:
    client:
      provider: keycloak
      keycloak:
        secret: arn:aws:secretsmanager:XXXX
```

Neste exemplo, o segredo do cliente para o provedor OAuth2 Keycloak é recuperado do ARN especificado em. AWS Secrets Manager Essa configuração oferece suporte a vários provedores resolvendo dinamicamente o nome do provedor e o ARN secreto correspondente.

utilitário de aplicativo

Eles `application-utility-pgm.yml` podem referenciar o ARN secreto para vários recursos.

- `spring.aws.client.datasources.primary`
  - `secret`ARN secreto para o banco de dados do aplicativo.

Tipo: string

- `type`

Nome totalmente qualificado da implementação do pool de conexões a ser usada.

Tipo: string

Padrão: `com.zaxxer.hikari.HikariDataSource`

- `spring.aws.client.utility.pgm.datasources`
  - `names`

Lista de nomes de fontes de dados.

Tipo: string

- `dsname`
  - `dbname`

Nome do host.

Tipo: string

- `secret`

ARN secreto do banco de dados do host.

Tipo: string

- `type`

Nome totalmente qualificado da implementação do pool de conexões a ser usada.

Tipo: string

Padrão: `com.zaxxer.hikari.HikariDataSource`

Para um segredo de várias fontes de dados:

```
spring:
  aws:
    client:
      primary:
        secret: arn:aws:secretsmanager:XXXX
        type: dataSourceType
      utility:
        pgm:
          datasources:
            names: dsname1,dsname2,dsname3
              dsname1:
                dbname: dbname1
                secret: arn:aws:secretsmanager:XXXX
                type: dataSourceType
              dsname2:
                dbname: dbname2
                secret: arn:aws:secretsmanager:XXXX
                type: dataSourceType
              dsname3:
                dbname: dbname3
                secret: arn:aws:secretsmanager:XXXX
                type: dataSourceType
```

Nenhuma chave secreta compatível com XA

- motor (postgres/oracle/db2/mssql)
- porta
- dbname

- Esquema atual
- username
- password
- url
- Conexão SSL
- sslTrustStoreLocalização
- sslTrustStoreSenha

Pois `postgres` somente a chave `sslMode` secreta valued (`disable/allow/prefer/require/verify-ca/verify-full`), além da propriedade `spring.aws.rds.ssl.cert-path` YAML, permite a conexão com SSL.

Chaves secretas compatíveis com XA

Se o banco de dados do cliente estiver usando XA, as subxa-propriedades são suportadas por meio de valores secretos.

- host
- porta
- dbname
- Esquema atual
- username
- password
- url
- Conexão SSL (verdadeiro/falso)
- sslTrustStoreLocalização
- sslTrustStoreSenha

No entanto, para outras propriedades x (por exemplo, `maxPoolSize` ou`driverType`), a chave YAML normal ainda `spring.jta.atomikos.datasource.XXXX.unique-resource-name` deve ser fornecida.

O valor secreto substitui as propriedades YAML.

Outros arquivos (groovy, sql, etc.)

Os outros arquivos usados pelo projeto do cliente usam regras de precedência semelhantes às da configuração do Spring. Exemplos:

- Os scripts do Groovy são arquivos `.groovy` na pasta ou subpastas `scripts`.
- Os scripts SQL são arquivos `.sql` na pasta ou subpastas `sql`.
- Os scripts Daemon são arquivos `.groovy` na pasta ou subpastas `daemons`.
- Consultas O arquivo de mapeamento do banco de dados são arquivos nomeados `queries-database.mapping` nas subpastas da pasta `sql`.
- Os modelos do Jasper são arquivos `.jrxml` na pasta ou subpastas `templates`.
- Os catálogos de conjuntos de dados são arquivos `.json` na pasta `catalog`.
- Os arquivos Lnk são arquivos `.json` na pasta `lnk`.

Todos esses locais podem ser substituídos por meio de uma propriedade do sistema ou de uma propriedade YAML do cliente.

- Para scripts do Groovy: `configuration.scripts`
- Para scripts SQL: `configuration.sql`
- Para scripts Daemon: `configuration.daemons`
- Para o arquivo de mapeamento do banco de dados de consultas: `configuration.databaseMapping`
- Para modelos Jasper: `configuration.templates`
- Para catálogos de conjuntos de dados: `configuration.catalog`
- Para arquivos Lnk: `configuration.lnk`

Se a propriedade não for encontrada, os arquivos serão retirados do local padrão mencionado acima. A pesquisa será feita primeiro com o diretório de trabalho do tomcat como raiz e, por último, no arquivo war do aplicativo.

Aplicação web adicional

O AWS Blu Age Runtime contém aplicativos web adicionais em sua `webapps-extra` pasta. Essas aplicações não são atendidas por padrão pelo servidor tomcat.

A adesão a essas aplicações web depende do projeto de modernização e é feita movendo o arquivo war desejado da pasta `webapps-extra` para a pasta `webapps`. Depois disso, a guerra será atendida pelo servidor tomcat na próxima inicialização.

Algumas configurações adicionais específicas do projeto também podem ser adicionadas em um arquivo de configuração YAML para cada guerra adicional, conforme feito no `application-main.yml` arquivo e explicado acima. As guerras adicionais são:

- `gapwalk-utility-pgm.war`: contém suporte para programas utilitários do ZOS e usa `application-utility-pgm.yaml` como configuração.
- `gapwalk-cl-command.war`: contém suporte para programas utilitários AS/400 e usa `application-cl-command.yaml` como configuração.
- `gapwalk-hierarchical-support.war`: contém suporte a transações IMS/MFS e usa como configuração `application-jhdb.yaml`

Habilitando propriedades

Nos aplicativos Spring Boot, `application-main.yml` é o arquivo de configuração no qual definimos diferentes tipos de propriedades, como porta de escuta, conectividade do banco de dados e muito mais.

Tópicos

- Notação YML
- Início rápido / Casos de uso
- Propriedades disponíveis para a aplicação principal
- Propriedades disponíveis para aplicações web opcionais

Notação YML

Na documentação a seguir, uma propriedade como a `parent.child1.child2=true` é escrita da seguinte forma no formato YAML.

```
parent:
 child1:
   child2: true
```

Início rápido / Casos de uso

Os casos de uso a seguir mostram exemplos das chaves e valores aplicáveis.

- application-main.yml padrão

```
----
#### DEFAULT APPLICATION-MAIN.YML FILE      #####
#### SHOWING USEFUL CONFIGURATION ELEMENTS #####
#### SHOULD BE OVERRIDDEN AND EXTERNALIZED   #####

###################################
##### Logging configuration #####
###################################
logging:
  config: classpath:logback-main.xml
  level.org.springframework.beans.factory.support.DefaultListableBeanFactory : WARN

###################################
##### Spring configuration #####
###################################
spring:
  quartz:
    auto-startup: false
    scheduler-name: Default
    properties:
      org.quartz.threadPool.threadCount: 1
  jta:
    enabled: false
    atomikos.properties.maxTimeout : 600000
    atomikos.properties.default-jta-timeout : 100000
  jpa:
# DISABLE OpenEntityManagerInViewInterceptor
    open-in-view: false
    # Fix Postgres JPA Error:
    # Method org.postgresql.jdbc.PgConnection.createClob() is not yet implemented.
    properties.hibernate.temp.use_jdbc_metadata_defaults : false
#####################################
##### Jics tables configuration #####
#####################################

    # The dialect should match the jics datasource choice
```

```
    database-platform : org.hibernate.dialect.PostgreSQLDialect #
 org.hibernate.dialect.PostgreSQLDialect, org.hibernate.dialect.SQLServerDialect

    # those properties can be used to create and initialize jics tables
 automatically.
#    properties:
#      hibernate:
#      globally_quoted_identifiers: true
#        hbm2ddl:
#           import_files_sql_extractor :
 org.hibernate.tool.hbm2ddl.MultipleLinesSqlCommandExtractor
#           import_files : file:./setup/initJics.sql
#           auto : create

##########################
###### Level 2 cache #####
##########################
#        cache:
#          use_second_level_cache: true
#          use_query_cache: true
#          region:
#            factory_class: org.hibernate.cache.ehcache.EhCacheRegionFactory
#      javax:
#        persistence:
#          sharedCache:
#            mode: ENABLE_SELECTIVE
##########################
###### Redis settings #####
##########################
  session:
    store-type: none #redis

########################################
##### JICS datasource configuration #####
########################################
datasource:
  jicsDs:
    driver-class-name : org.postgresql.Driver # org.postgresql.Driver,
 com.microsoft.sqlserver.jdbc.SQLServerDriver
    url: jdbc:postgresql://localhost/jics # jdbc:postgresql://localhost:5433/jics,
 jdbc:sqlserver://localhost\SQLEXPRESS:1434;databasename=jics;
    username: jics
    password: jics
```

```
    type : org.postgresql.ds.PGSimpleDataSource #
 org.postgresql.ds.PGSimpleDataSource,
 com.microsoft.sqlserver.jdbc.SQLServerDataSource

#######################################################
##### Embedded Bluesam datasource configuration #####
#######################################################
  bluesamDs :
    driver-class-name : org.postgresql.Driver # org.postgresql.Driver,
 com.microsoft.sqlserver.jdbc.SQLServerDriver
    url : jdbc:postgresql://localhost/bluesam # jdbc:postgresql://localhost:5433/
jics, jdbc:sqlserver://localhost\SQLEXPRESS:1434;databasename=jics;
    username : bluesam
    password : bluesam
    type : org.postgresql.ds.PGSimpleDataSource #
 org.postgresql.ds.PGSimpleDataSource,
 com.microsoft.sqlserver.jdbc.SQLServerDataSource

########################################
##### Embedded Bluesam configuration #####
########################################
bluesam :
  remote : false
  cache : ehcache
  persistence : pgsql #pgsql, mssql, xodus...
  ehcache:
    resource-pool:
      size: 4GB
  write-behind:
    enabled: true
  pgsql :
    dataSource : bluesamDs

########################
##### Jics settings #####
########################
rabbitmq.host: localhost
jics:
  cache: false #redis
  resource-definitions.store-type: jpa # default value: jpa, other possible value:
 redis
  redis.hostname: 127.0.0.1 # Redis server host.
  redis.password: redis # Login password of the redis server.
  redis.port: 6379 # Redis server port.
```

```
  redis.username: # Redis username
  redis.mode: standalone # Redis mode. Possible values: standalone, cluster
jics.disableSyncpoint : false
#jics.initList:
#jics.parameters.datform: DDMMYY
#jics.parameters.applid: VELOCITY
#jics.parameters.sysid: CICS
#jics.parameters.eibtrmid: TERM
#jics.parameters.userid: MYUSERID
#jics.parameters.username: MYUSERNAME
#jics.parameters.opid: XXX
#jics.parameters.cwa.length: 0
#jics.parameters.netname: MYNETNAME
#jics.parameters.jobname: MJOBNAME
#jics.parameters.sysname: SYSNAME

################################################
##### Jics RunUnitLauncher pool settings #####
################################################
#jics.runUnitLauncherPool.enable: false
#jics.runUnitLauncherPool.size: 20
#jics.runUnitLauncherPool.validationInterval: 1000

#########################
##### Jhdb settings #####
#########################
#jhdb.lterm: LTERMVAL
#jhdb.identificationCardData: SomeIDData

###################################
##### DateHelper configuration ####
###################################
#forcedDate: "2013-08-26T12:59:58+01:57"

#############################
##### Sort configuration ####
#############################
#externalSort.threshold: 256MB

###################################
##### Server timeout (10 min)  ####
###################################
spring.mvc.async.request-timeout: 600000
```

```
################################
##### DATABASE STATISTICS  ####
################################
databaseStatistics : false

######################
##### CALLS GRAPH ####
######################
callGraph : false

################################
##### SQL SHIFT CODE POINT ####
################################
# Code point 384 match unicode character \u0180
sqlCodePointShift : 384

################################
##### LOCK TIMEOUT RECORD  ####
################################
# Blu4IV record lock timeout
lockTimeout : 100

################################
##### REPORTS OUTPUT PATH ####
################################
reportOutputPath: reports

################################
##### TASK EXECUTOR       ####
################################
taskExecutor:
  corePoolSize: 5
  maxPoolSize: 10
  queueCapacity: 50
  allowCoreThreadTimeOut: false

################################
##### PROGRAM NOT FOUND   ####
################################
stopExecutionWhenProgNotFound: false

#########################################################
#####  DISP DEFAULT VALUE (to be removed one day) ####
#########################################################
```

```
defaultKeepExistingFiles: true

####################################
#####  JOBQUEUE CONFIGURATION    ####
####################################
jobqueue:
  api.enabled: false
  impl: none # possible values: quartz, none
  schedulers:  # list of schedulers
    -
      name: queue1
      threadCount: 5
    -
      name: queue2
      threadCount: 5

#############################################################################
##### QUERY BUILDING                                                     ##
# useConcatCondition : false by default
# if true, in the query, the where condition is build with key concatenation  ##
#############################################################################
# query.useConcatCondition: true
----
```

- Use arquivos de comprimento variável com os comandos LISTCAT

```
[**/*.*]
encoding=IBM930
reencoding=false

[global]
listcat.variablelengthpreprocessor.enabled=true
listcat.variablelengthpreprocessor.type=rdw
# use "rdw" if your .listcat file contains a set of records (RDW)
# use "bdw" if your .listcat file contains a set of blocks (bdw)
```

- Forneça o valor do indicador de bytes nulos no utilitário LOAD/UNLOAD

```
# Unload properties
```

```
# For date/time: if use database configuration is enabled, formats are ignored
# For nbi; use hexadecimal syntax to specify the byte value
# - When the value is null in database : the value dumped to the file is filled by
 low value characters and the NBI is
# equal to the byte 6F (the ? character)
# - When the value is not null in database and the column is nullable: the NBI is
 equal to the byte 00 (low value) and NOT
# equal to the byte 40 (space)
unload:
  sqlCodePointShift: 0
  nbi:
    whenNull: "6F"
    whenNotNull: "00"
  useDatabaseConfiguration: false
  format:
    date: MM/dd/yyyy
    time: HH.mm.ss
    timestamp: yyyy-MM-dd-HH.mm.ss.SSSSSS
```

Propriedades disponíveis para a aplicação principal

Esta tabela fornece uma visão exaustiva dos parâmetros de chave/valores.

Chave	Tipo	Valor padrão	Descrição
`logging.config`	Path	caminho de classe: logback-main.xml	Chave padrão para a referência ao arquivo de configura ção de logback. Outras chaves de log padrão também estão disponíveis.
`spring.jt a.enabled`	boolean	false	Chave Padrão. Se o modo de suporte da fonte de dados não for static-xa , a configuração automática das

Chave	Tipo	Valor padrão	Descrição
			transações do Spring JTA deverá ser desativada.
`datasource.jicsDs + -driver-class-name + -url + -username + -password + -type`	Fonte de dados Spring padrão com subchaves		Contém as informações de conexão do banco de dados do Jics. Como alternativa, o uso de segredos da AWS é fortemente encorajado, conforme explicado em xref:../configuration/configuration.adoc[Configuration].
`datasource.bluesamDs + -driver-class-name + -url + -username + -password + -type`	Fonte de dados Spring padrão com subchaves		Contém as informações de conexão do banco de dados Blusam. Como alternativa, o uso de segredos da AWS é fortemente encorajado, conforme explicado em xref:../configuration/configuration.adoc[Configuration].
`bluesam.disabled`	boolean	false	Se deve desativar completamente o Blusam.

Chave	Tipo	Valor padrão	Descrição
`bluesam.cache`	string		Se não estiver definido, o cache do Blusam não será usado. Os valores possíveis (implementações de cache) são cache e redis.
`forcedDate`	string		Força a data até a data fornecida, se houver uma.
`frozenDate`	booliano	true	Especifica se a data deve ser congelada. Aplica-se somente se `forcedDate` também estiver definido.
`externalSort.threshold`	tamanho de dados (exemplo: 12 MB)		O limite de classificação: quando alternar para a classificação externa (mesclagem).
`jics.parameters.datform`	string	MMDDAA	O formato de data.

Chave	Tipo	Valor padrão	Descrição
`jics.initList`	string		A lista de inicialização do JICS, separada por vírgulas. Se presente, ele define nomes de listas separados por vírgula a serem ativados na inicialização do Apache Tomcat entre as listas do CICS. Valor de exemplo: `$UUU,DFH $IVPL,PEZ1` . Isso será transmitido em cascata para os grupos contidos nessas listas e suas definições de recursos subjacentes, que serão então visíveis para o runtime. Vazio por padrão.
`jics.parameters.applid`	string	VELOCITY	O aplicado para identificar o aplicativo no JICS (pelo menos 4 caracteres, sem tamanho máximo).
`jics.parameters.sysid`	string	CICS	A identificação do sistema (SYSID).

Chave	Tipo	Valor padrão	Descrição
`jics.parameters.eibtrmid`	string	PRAZO	O identificador do terminal (máximo de 4 caracteres, mínimo 1).
`jics.parameters.userid`	string		O ID do usuário (máximo de 8 caracteres, sem mínimo). Quando nenhum valor é fornecido (em branco por padrão), o ID da sessão HTTP é usado como o ID do usuário.
`jics.parameters.username`	string	MYUSERNAME	O nome de usuário (máximo de 10 caracteres, mínimo 1).
`jics.parameters.netname`	string	MYNETNAME	O nome da rede (máximo de 8 caracteres, mínimo 1).
`jics.parameters.opid`	string	XXX	A identificação do operador de 3 caracteres.
`jics.parameters.jobname`	string	MJOBNAME	O nome do trabalho.
`jics.parameters.sysname`	string	SYSNAME	O nome do sistema AS400 (sysname).
`jics.parameters.cwa.length`	número	0	O comprimento da área de trabalho comum (CWA).

Chave	Tipo	Valor padrão	Descrição
`jics.parameters.charset`	string	CP037	O conjunto de caracteres usado globalmente pelo JICS.
`jics.parameters.tsqimpl`	string	bluesam	Implementação da fila de armazenamento temporário (TSQ) do JICS (os valores permitidos são `bluesam` / `memory` / `redis`)
`jics.queues.ts.redis.hostname`	string	127.0.0.1	O nome do host do servidor jics cache do Redis.
`jics.queues.ts.redis.port`	número	6379	A porta do servidor redis de cache jics.
`jics.queues.ts.redis.password`	string	redis	A senha do servidor redis do cache jics.
`jics.queues.ts.redis.username`	string		O nome de usuário do servidor redis de cache jics. O padrão é em branco (sem nome de usuário).

Chave	Tipo	Valor padrão	Descrição
`jics.queues.ts.redis.mode`	string	standalone	O modo de cache do jics. Os valores possíveis são: `standalone` ou `cluster`. O padrão é `standalone` .
`lockTimeout`	número	500	O tempo limite do bloqueio, em milissegundos.
`sqlCodePointShift`	número		Opcional. A mudança de ponto do código sql. Muda o ponto de código dos caracteres de controle que podemos encontrar ao migrar dados antigos do RDBMS para um RDBMS moderno. Por exemplo, você pode especificar `384` para corresponder ao `\u0180` caractere Unicode.
`sqlIntegerOverflowAllowed`	boolean	false	Especifica se é permitido o estouro de números inteiros SQL, ou seja, se é permitido colocar valores maiores na variável do host.

Chave	Tipo	Valor padrão	Descrição
`database.cursor.overflow.allowed`	booliano	true	Especifica se é permitido que o cursor transborde. Defina como `true` para realizar uma próxima chamada no cursor, seja qual for sua posição. Defina como `false` para verificar se o cursor está na última posição antes de realizar uma próxima chamada no cursor. Ative somente se o cursor for SCROLLABLE (SENSITIVE ou INSENSITIVE).
`reportOutputPath`	string	`/reports`	O caminho de saída do relatório.
`spring.session.store-type`	string	nenhuma	O cache da sessão para ambientes de alta disponibilidade. Os valores possíveis são: `none` ou `redis`. O padrão é `none`.

Chave	Tipo	Valor padrão	Descrição
`stopExecutionWhenProgNotFound`	booliano	true	Especifica se a execução deve ser interrompida se um programa não for encontrado. Se definido como`true`, interrompe a execução se um programa não for encontrado.
`forceHR`	boolean	false	Especifica se o SYSPRINT legível por humanos deve ser usado no console ou na saída do arquivo.
`rollbackOnRTE`	boolean	false	Especifica se a transação de unidade de execução implícita deve ser revertida em exceções de runtime.
`sctThreadLimit`	longo	5	O limite de threads para acionar scripts.

Chave	Tipo	Valor padrão	Descrição
`dataSimplifier.onInvalidNumericData`	string	reject	Como reagir ao decodificar dados numéricos inválidos. Os valores permitidos são: `reject`, `tolerates paces` , `tolerates paceslowvalues` e `toleratemost` . O padrão é`reject`.
`filesDirectory`	string		O diretório para arquivos de entrada/saída de lotes.
`ims.messages.extendedSize`	boolean	false	Especifica se o tamanho estendido das mensagens IMS deve ser definido.
`defaultKeepExistingFiles`	boolean	false	Especifica se o valor anterior padrão do conjunto de dados deve ser definido.
`jics.db.ddlScriptLocation`	string		A localização do script Jics DDL. Permite que você inicie o esquema do banco de dados Jics usando um script.sql. Em branco por padrão. Por exemplo, `./jics/sql/jics.sql` .

Chave	Tipo	Valor padrão	Descrição
`jics.db.schemaTestQueryLocation`	string		Localização do arquivo sql que deve conter uma consulta exclusiva que retorna o número de objetos no esquema jics (se houver).
`jics.db.dataScriptLocation`	string		Localização do script initJics.sql, preparado pelo Analyzer a partir da análise das exportações de CSD do mainframe.
`jics.db.dataTestQueryLocation`	string		Localização de um script sql contendo uma única consulta sql que deve retornar uma contagem de objetos (por exemplo: contagem do número de registros na tabela do programa jics). Se a contagem for igual a 0, o banco de dados será carregado usando o script`jics.db.dataScriptLocation`, caso contrário, o carregamento do banco de dados será ignorado.

Chave	Tipo	Valor padrão	Descrição
`jics.data.dataJsonInitLocation`	string		
`jics.xa.agent.timeout`	número		
`query.useConcatCondition`	boolean	false	Especifica se a condição da chave é criada por concatenação de chaves ou não.
`system.qdecfmt`	string		
`disposition.checkexistence`	boolean	false	Especifica se deve liberar uma verificação da existência do arquivo para o conjunto de dados com DISP SHR ou OLD.
`useControlMVariable`	boolean	false	Especifica se a especificação Control-m deve ser usada para substituição de variáveis.
`card.encoding`	string	CP1145	Codificação do cartão: para ser usado com `useControlMVariable` .

Chave	Tipo	Valor padrão	Descrição
`mapTransfo.prefixes`	string	&,@,%%	Lista de prefixos a serem usados ao transformar variáveis controlM. Cada um separado por vírgula.
`checkinputfilesize`	boolean	false	Especifica se um cheque deve ser liberado se o tamanho do arquivo for múltiplo do tamanho do registro.
`stepFailWhenAbend`	booliano	true	Especifica se a suspensão deve ser levantada se uma etapa falhar ou concluir a execução.
`bluesam.fileLoading.commitInterval`	número	100000	O intervalo de confirmação do bluesam.
`uppercaseUserInput`	booliano	true	Especifica se a entrada do usuário deve estar em maiúsculas.

Chave	Tipo	Valor padrão	Descrição
`jhdb.lterm`	string		Permite que você force um ID de terminal lógico comum no caso de uma emulação de IMS. Se não for definido, o sessionId será usado.
`jhdb.identificationCardData`	string		Usado para codificar alguns “dados do cartão de identificação do operador” no campo MID designado pelo parâmetro CARD. Em branco por padrão, sem restrição de entrada.
`encoding`	string	ASCII	A codificação usada em projetos (não em arquivos groovy). Espera uma codificação válida `CP1047`, `IBM930`, `ASCII`, `UTF-8`...
`cl.configuration.context.encoding`	string	CP297	A codificação dos arquivos CL. Espera uma codificação válida`CP1047`,`IBM930`,`ASCII`, O valor padrão é`CP297`

Chave	Tipo	Valor padrão	Descrição
`cl.zonedMode`	string	EBCDIC_STRICT	O modo para codificar ou decodificar comandos da linguagem de controle (CL). Os valores permitidos são: `EBCDIC_STRICT` , `EBCDIC_MODIFIED` e `AS400`.
`ims.programs`	string		Lista de programas IMS a serem usados. Separe cada parâmetro com um ponto e vírgula (`;`) e cada transação com uma vírgula (`,`). Por exemplo: `PCP008,PCT008;PCP054,PCT054;PCP066,PCT066;PCP068,PCT068;` .
`jhdb.configuration.context.encoding`	string	CP297	A codificação JHDB (Java Hierarchical Database). Espera uma string de codificação válida `CP1047`, `IBM930`, `ASCII`, `UTF-8`...

Chave	Tipo	Valor padrão	Descrição
`jhdb.metadata.extrapath`	string	file:./setup/	Um parâmetro de configuração que especifica uma pasta raiz extra específica do runtime para as pastas psbs e dbds.

Chave	Tipo	Valor padrão	Descrição
`jhdb.checkpointPersistence`	string	nenhuma	O modo de persistência do ponto de verificação. Os valores permitidos são: `none`, `add` e `end`. Use `add` para manter os pontos de verificação quando um novo for criado e adicionado ao registro. Use `end` para manter o ponto de verificação no desligamento do servidor. Quaisquer outros valores desativam a persistência. Observe que sempre que um novo ponto de verificação for adicionado ao registro, todos os pontos de verificação existentes serão serializados e o arquivo será apagado. Não é um acréscimo aos dados existentes no arquivo. Portanto, dependendo do número de pontos de verificação, isso pode ter alguns efeitos no desempenho.

Chave	Tipo	Valor padrão	Descrição
`jhdb.checkpointPath`	string	file:./setup/	Se `jhdb.checkpointPersistence` não for `none`, esse parâmetro permitirá que você configure o caminho de persistência do ponto de verificação (local de armazenamento do arquivo checkpoint.dat); todos os dados de pontos de verificação contidos no registro são serializados e armazenados em um arquivo (checkpoint.dat) localizado na pasta fornecida. Observe que somente os dados do ponto de verificação (scriptId, stepId, posição do banco de dados e área do ponto de verificação) são afetados por esse backup.
`jhdb.navigation.cachenexts`	número	5000	A duração do cache (em milissegundos) usada na navegação hierárquica para um RDBMS.

Chave	Tipo	Valor padrão	Descrição
`jhdb.use-db-prefix`	booliano	true	Especifica se um prefixo de banco de dados deve ser ativado na navegação hierárquica para um RDBMS.
`jhdb.query.limitJoinUsage`	booliano	true	Especifica se o parâmetro limite de uso de junção deve ser usado em gráficos RDBMS.
`taskExecutor.corePoolSize`	número	5	Quando uma transação em um terminal é iniciada por meio de um script groovy, um thread é criado. Use esse parâmetro para configurar o tamanho do pool principal.
`taskExecutor.maxPoolSize`	número	10	Quando uma transação em um terminal é iniciada por meio de um script groovy, um thread é criado. Use esse parâmetro para configurar o tamanho máximo do grupo (número máximo de threads paralelos).

Chave	Tipo	Valor padrão	Descrição
`taskExecutor.queueCapacity`	número	50	Quando uma transação em um terminal é iniciada por meio de um script groovy, um thread é criado. Use esse parâmetro para configurar o tamanho da fila. (= número máximo de transações pendentes quando `taskExecutor.maxPoolSize` atingido)
`taskExecutor.allowCoreThreadTimeOut`	boolean	false	Especifica se os threads principais devem atingir o tempo limite no JCIS. Isso permite o crescimento e a redução dinâmicos, mesmo em combinação com uma fila diferente de zero (já que o tamanho máximo do pool só aumentará quando a fila estiver cheia).
`jics.runUnitLauncherPool.enable`	boolean	false	Especifica se o pool do lançador de unidades de execução deve ser ativado no JICS.

Chave	Tipo	Valor padrão	Descrição
`jics.runUnitLauncherPool.size`	número	20	O tamanho do pool do lançador da unidade de execução no JICS.
`jics.runUnitLauncherPool.validationInterval`	número	1000	O intervalo entre cada execução da tarefa que ajusta o tamanho do pool.
`jics.runUnitLauncherPool.parallelism`	número	2	O número de segmentos usados para produzir as instâncias ausentes na fila quando a tarefa de ajuste é executada .
`context.preconstruct.enable`	boolean	false	Especifica se a pré-construção do contexto do programa deve ser ativada.
`context.preconstruct.frequencyInMillis`	número	100	O intervalo entre cada execução da tarefa que ajusta o tamanho do pool.
`context.preconstruct.parallelism`	número	5	O número de segmentos usados para produzir as instâncias ausentes na fila quando a tarefa de ajuste é executada .

Chave	Tipo	Valor padrão	Descrição
`context.preconstruct.minInstances`	número	2	O número de instâncias que serão criadas na primeira vez em que um contexto for necessário.
`spring.aws.application.credentials`	string	nulo	Carregue as credenciais da AWS do arquivo de perfis de credenciais no JICS.
`jics.queues.sqs.region`	string	eu-west-1	A região da AWS para o AWS Simple Queue Service, usada no JICS.
`mq.queues.sqs.region`	string	eu-west-3	A região da AWS para o serviço AWS SQS MQ.
`quartz.scheduler.stand-by-if-error`	boolean	false	Especifica se a execução do trabalho deve ser acionado se o agendador de trabalhos estiver no modo de espera. Se verdadeiro, quando ativada, a execução do trabalho não é acionada.

Chave	Tipo	Valor padrão	Descrição
`databaseStatistics`	boolean	false	Especifica se devem permitir que os construtores de SQL coletem e exibam informações estatísticas.
`dbDateFormat`	string	aaaa-MM-dd	O formato da data alvo do banco de dados.
`dbTimeFormat`	string	HH:mm:ss	O formato de hora alvo do banco de dados.
`dbTimestampFormat`	string	aaaa-MM-dd HH:mm:ss.SSSSSS	O formato de carimbo de data/hora de destino do banco de dados.
`dateTimeFormat`	string	ISO	dateTimeFormat Descreve como inserir o tipo de data e hora do banco de dados em entidades simplificadoras de dados. Os valores permitidos são: `ISO`, `EUR`, `EUR`, `USA` e `LOCAL`.
`localDateFormat`	string		Lista de formatos de data locais. Separe cada formato com \.

Chave	Tipo	Valor padrão	Descrição
`localTimeFormat`	string		Lista de formatos de horário local. Separe cada formato com \
`localTimestampFormat`	string		Lista de formatos de carimbo de data/hora locais. Separe cada formato com \.
`pgmDateFormat`	string	aaaa-MM-dd	O formato de data e hora.
`pgmTimeFormat`	string	HH.mm.ss	O formato de hora usado para execução de pgm (programas).
`pgmTimestampFormat`	string	aaaa-MM-dd-HH.mm.ss.SSSSSS	O formato do carimbo de data e hora.
`cacheMetadata`	booliano	true	Especifica se os metadados do banco de dados devem ser armazenados em cache.
`forceDisableSQLTrimStringType`	boolean	false	Especifica se o corte de todos os parâmetros da string sql deve ser desativado.

Chave	Tipo	Valor padrão	Descrição
`fetchSize`	número		O valor fetchSize para cursores. Use ao buscar dados usando fragmentos por meio de utilitári os de carregamento/ descarregamento.
`check-groovy-file`	booliano	true	Especifica se o conteúdo dos arquivos groovy deve ser verificado antes do registro.
`qtemp.uuid.length`	número	9	O comprimento de identificação exclusivo do QTEMP.
`qtemp.dblog`	boolean	false	Se deve habilitar o log do banco de dados QTEMP.
`qtemp.cleanup.threshold.hours`	número	0	Para especificar quando `qtemp.dblog` está ativado. A vida útil da partição db (em horas).
`sort.function`	string		O nome da função de classificação para o banco de dados blu4iv.

Chave	Tipo	Valor padrão	Descrição
`invalidDataTolerence`	booliano	true	Especifica se dados inválidos são tolerados para o tipo empacotado.
`program.timeout`	número	-1	Especifica um tempo limite para a execução de qualquer programa/transação em segundos. Após esse período, o sistema tentará interromper o programa.
`gapwalk.line.separator`	string	nulo	Especifica o tipo de separador de linha no gapwalk. Os valores permitidos são WIN (CRLF)/UNIX (LF)/LINUX (LF). Outros valores são ignorados e a propriedade System line.sepatator é usada.

Chave	Tipo	Valor padrão	Descrição
`enableActivePgmIdCache`	boolean	false	Especifica se o cache local do ID do programa ativo deve ser ativado. Use esse recurso com cuidado porque os recursos do JICS podem ser compartilhados entre programas e usuários. Esses recursos podem ser alterados externamente por qualquer administrador e o cache local instalado pode ser invalidado.

Chave	Tipo	Valor padrão	Descrição
`mq.queues.default.syncpoint`	boolean	false	Especifica o comportamento padrão dos comandos MQ PUT quando nem MQPMO_SYNCPOINT nem MQPMO_NO_SYNCPOINT estão definidos. Quando definido como verdadeiro, ele age como se `MQPMO_SYNCPOINT` as mensagens NÃO fossem confirmadas diretamente durante o comando PUT. Quando definido como falso, ele age como `MQPMO_NO_SYNCPOINT` e as mensagens são confirmadas diretamente durante o comando PUT.
`dataSimplifier.byteRangeBoundsCheck`	boolean	false	Quando definido como verdadeiro, ele garante que não ByteRange seja criado com valores impróprios. O padrão é falso.

Chave	Tipo	Valor padrão	Descrição
`file.stdoutIntoLogger`	boolean	false	Especifica se a gravação no registrador deve ser ativada em vez do fluxo de saída padrão do sistema nos arquivos padrão `SYSPRINT` e. `SYSPUNCH`
`tempFilesDirectory`	string	nulo	Especifica o nome da localização da pasta dos arquivos temporários que são gerados.

Chave	Tipo	Valor padrão	Descrição
`tempFolderPattern`	string	nulo	Especifica um padrão que será usado para criar dinamicamente o nome da pasta temporária com base nas seguintes informações predefinidas e personalizáveis. HOST: o nome do host. JOBID: o ID do trabalho. HASHCODE: o código hash do contexto do trabalho. TIMESTAMP: o padrão a ser usado ao obter o timestamp. O nome de destino da pasta temporária é TMP_DIR_ {}. tempFolderPattern Por exemplo, no caso do padrão a seguir, o nome começará com o ID do trabalho e terminará com o “timestamp”: tempFolderPattern: JOBID, HOST=XXXX, HASHCODE, timestamp

Chave	Tipo	Valor padrão	Descrição
			=YYYYMMDD HHMMSS. Se a propriedade não `tempFolderPattern` for adicionada ao arquivo YAML ou estiver vazia, o nome da pasta temporária será “TMP_DIR_” + this.hashCode () (). DefaultJobContext
`database.cursor.raise.already.opened.error`	boolean	false	Especifica se é necessário ativar o aumento do erro SQLCODE 502 quando um cursor já aberto está sendo aberto.
`jics.spool.smtp.hostname`	string	nulo	Especifica o host do servidor SMTP. Exemplo: `smtp.xxx.com`
`jics.spool.smtp.port`	string	nulo	Especifica a porta do servidor SMTP. Exemplo: 25
`jics.spool.smtp.password`	string	nulo	Especifica a senha de login do servidor SMTP.

Chave	Tipo	Valor padrão	Descrição
`jics.spool.smtp.username`	string	nulo	Especifica o nome de usuário do servidor SMTP.
`jics.spool.smtp.debug`	boolean	false	Especifica o modo de depuração para o servidor SMTP.
`gapwalk-application.security`	string	`disabled`	Alterne a configuração de segurança global (autenticação XSS, CORS, CSRF, OAUTH...). Os valores permitidos são `disabled` e `enabled`.
`gapwalk-application.identity`	string	nulo	Método de autenticação global. O valor recomendado é`oauth`. Os valores permitidos são `json` e `oauth`. Essa opção é necessária quando `gapwalk-application.security` é`enabled`.

Chave	Tipo	Valor padrão	Descrição
`gapwalk-application.security.issuerUri`	string	nulo	O URI do emissor do provedor de identidade (IdP). Essa opção é necessária quando `gapwalk-application.identity` é `oauth`.
`gapwalk-application.security.allowedOrigins`	sequência de caracteres []	`null`	A lista de origens a serem permitidas. Essa opção `gapwalk-application.identity` precisa ser definida como `oauth`.
`gapwalk-application.security.claimGroupName`	string	`cognito:groups`	O atributo de declaração que contém a lista de todos os grupos aos quais um usuário pertence. Use `cognito:groups` para o Amazon Cognito ou qualquer outra string para um IdP externo.

Chave	Tipo	Valor padrão	Descrição
`gapwalk-application.security.userAttributeName`	string	`username`	O nome do atributo de declaração usado para identificar uma solicitação do usuário. Use `username` para Amazon Cognito, Keycloak ou qualquer outra string `preferred_username` para um IdP externo.
`gapwalk-application.security.localhostWhitelistingEnabled`	booliano	true	Especifica se a autenticação deve ser ativada a partir de qualquer `localhost` solicitação.
`gapwalk-application.defaultSuperAdminUserName`	string	`sadmin`	Quando `gapwalk-application.security` está desativado, especifica o nome de superusuário local padrão.
`gapwalk-application.defaultSuperAdminUserPwd`	string	`sadmin`	Quando `gapwalk-application.security` está desativado, especifica a senha padrão do superusuário local.

Chave	Tipo	Valor padrão	Descrição
`gapwalk-application.security.filterURIs`	string	`disabled`	Alterne a configuração de URIs de filtragem. Os valores permitidos são `disabled` e `enabled`.
`gapwalk-application.security.blockedURIs`	sequência de caracteres []	`null`	A lista de URIs a serem bloqueados. Essa opção é necessária quando `gapwalk-applicatio n.securit y.filterURIs éenabled`.
`jics.redis.database`	número	`0`	Especifica o índice do banco de dados para a fábrica de conexões do servidor Redis, variando de 0 a 15. O padrão é 0.
`jics.redis.maxTotal`	número	`32`	Número máximo de conexões ativas do pool Redis.
`jics.redis.maxIdle`	número	`32`	Número máximo de conexões ociosas do pool Redis.
`jics.redis.minIdle`	número	`8`	Número mínimo de conexões ociosas do pool Redis.

Propriedades disponíveis para aplicações web opcionais

Dependendo da sua aplicação modernizada, talvez seja necessário configurar um ou mais aplicações web opcionais que representem suporte para dependências como z/OS, AS/400 ou IMS/MFS. As tabelas a seguir contêm listas dos parâmetros de chave/valor disponíveis para configurar cada aplicação web opcional.

gapwalk-utility-pgm.guerra

Essa aplicação web opcional contém suporte para programas utilitários do Z/OS.

Esta tabela fornece uma visão exaustiva dos parâmetros-chave/valores dessa aplicação.

Chave	Tipo	Valor padrão	Descrição
`logging.config`	Path	classpath:logback-utility.xml	Chave padrão para a referência ao arquivo de configuração de logback. Outras chaves de log padrão também estão disponíveis.
`spring.jta.enabled`	boolean	false	Chave Padrão. Se o modo de suporte da fonte de dados não for static-xa, a configuração automática das transações do Spring JTA deverá ser desativada.
`spring.datasource.primary.jndi-name`	string	jdbc/primary	O nome JNDI (Java Naming And Directory Interface) da fonte de dados primária, se estiver usando JNDI.

Chave	Tipo	Valor padrão	Descrição
`primary.datasource -driver-class-name -url -username -password`	Fonte de dados Spring padrão com subchaves		Contém as informações de conexão do banco de dados da aplicação, se não estiver usando o JNDI. Deve ter a mesma configuração do arquivo YAML do aplicativo modernizado. Como alternativa, o uso de segredos da AWS é fortemente encorajado, conforme explicado em xref:../configuration/configuration.adoc[Configuration].
`encoding`	string	ASCII	A codificação usada em programas utilitários. Espera uma codificação válida `CP1047`, `IBM930`, `ASCII`, `UTF-8`...
`sysPunchEncoding`	string	ASCII	O conjunto de caracteres de codificação syspunch. Espera uma codificação válida `CP1047`, `IBM930`, `ASCII`, `UTF-8`...

Chave	Tipo	Valor padrão	Descrição
`zonedMode`	string	EBCDIC_STRICT	O modo para codificar ou decodificar tipos de dados zoneados. Os valores permitidos são: `EBCDIC_STRICT` , `EBCDIC_MODIFIED` e `AS400`.
`unload.chunkSize`	número	0	Tamanho do pedaço usado para o utilitário de descarga.
`unload.sqlCodePointShift`	número	0	O utilitário de mudança de pontos do código SQL para descarga. Executa o processo de mudança de caracteres. Obrigatório quando seu banco de dados de destino do DB2 é o Postgresql.
`unload.columnFiller`	string	space	O preenchedor de colunas do utilitário de descarga.

Chave	Tipo	Valor padrão	Descrição
`unload.varCharIsNull`	boolean	false	Use esse parâmetro no programa INFTILB, se definido como`true`, todos os campos não anuláveis com valores em branco (espaço) retornarão uma string vazia.
`unload.useDatabaseConfiguration`	boolean	false	Especifica se a configuração de data ou hora do application-main.yml deve ser usada no utilitário de descarregamento.
`unload.format.date`	string	MM/dd/aaaa	Se `unload.useDatabaseConfiguration` estiver ativado, o formato de data a ser usado no utilitário de descarga.
`unload.format.time`	string	HH.mm.ss	Se `unload.useDatabaseConfiguration` estiver ativado, o formato de hora a ser usado no utilitário de descarga.

Chave	Tipo	Valor padrão	Descrição
`unload.format.timestamp`	string	aaaa-MM-dd-HH.mm.ss.SSSSSS	Se `unload.useDatabaseConfiguration` estiver ativado, o formato de carimbo de data/hora a ser usado no utilitário de descarga.
`unload.nbi.whenNull`	hexadecimais	6F	O valor do Null Byte Indicator (NBI) a ser adicionado quando o valor do banco de dados for nulo.
`unload.nbi.whenNotNull`	hexadecimais	00	O valor do Null Byte Indicator (NBI) a ser adicionado quando o valor do banco de dados não for nulo.
`unload.nbi.writeNullIndicator`	boolean	false	Especifica se o indicador nulo deve ser gravado no arquivo de saída de descarga.
`unload.fetchSize`	número	0	Permite ajustar o tamanho da busca ao manipular cursores no utilitário de descarga.

Chave	Tipo	Valor padrão	Descrição
`treatLargeNumberAsInteger`	boolean	false	Especifica se os números grandes devem ser tratados como `Integer`. Eles são tratados como `BigDecimal` padrão.
`load.batchSize`	número	0	O tamanho do lote do utilitário de carga.
`load.format.localDate`	string	dd.MM.aaaa\dd/MM/aaaa\aaaa-MM-dd	O formato de data local do utilitário de carregamento a ser usado.
`load.format.localTime`	string	HH:mm:ss\HH.mm.ss	O formato de hora local do utilitário de carregamento a ser usado.
`load.format.dbDate`	string	aaaa-MM-dd	O formato do banco de dados do utilitário de carga a ser usado.
`load.format.dbTime`	string	HH:mm:ss	O tempo de uso do banco de dados do utilitário de carregamento.

Chave	Tipo	Valor padrão	Descrição
`load.sqlCodePointShift`	número	0s	A mudança de pontos do código SQL para o utilitário de carregamento. Executa o processo de mudança de caracteres. Obrigatório quando seu banco de dados de destino do DB2 é o Postgresql.
`forcedDate`	string		Força a data até a data fornecida, se houver uma.
`frozenDate`	booliano	true	Especifica se a data deve ser congelada. Aplica-se somente se `forcedDate` também estiver definido.
`jcl.type`	string	mvs	Tipo de arquivo.jcl. Os valores permitidos são: `jcl` e `vse`. Os comandos PRINT/REPRO do utilitário IDCAMS retornam 4 se o arquivo estiver vazio para um jcl que não seja vse.

Chave	Tipo	Valor padrão	Descrição
`hasGraphic`	boolean	false	Se o utilitário INFUTILB precisa lidar com colunas GRAPHIC DB2.

gapwalk-cl-command.guerra

Essa aplicação web opcional contém suporte para programas utilitários AS/400.

Esta tabela fornece uma visão exaustiva dos parâmetros-chave/valores dessa aplicação.

Chave	Tipo	Valor padrão	Descrição
`logging.config`	Path	classpath:logback-utility.xml	Chave padrão para a referência ao arquivo de configuração de logback. Outras chaves de log padrão também estão disponíveis.
`spring.jta.enabled`	boolean	false	Chave Padrão. Se o modo de suporte da fonte de dados não for static-xa, a configuração automática das transações do Spring JTA deverá ser desativada.
`spring.datasource.primary.jndi-name`	string	jdbc/primary	O nome JNDI (Java Naming And Directory Interface) da fonte de

Chave	Tipo	Valor padrão	Descrição
			dados primária, se estiver usando JNDI.
`primary.datasource + -driver-class-name + -url + -username + -password`	Fonte de dados Spring padrão com subchaves		Contém as informações de conexão do banco de dados da aplicação, se não estiver usando o JNDI. Deve ter a mesma configuração do arquivo yml da aplicação modernizada. Como alternativa, o uso de segredos da AWS é fortemente encorajado, conforme explicado em xref:../configuration/configuration.adoc[Configuration].
`encoding`	string	ASCII	A codificação usada em programas utilitários. Espera uma codificação válida `CP1047`, `IBM930`, `ASCII`, `UTF-8`...

Chave	Tipo	Valor padrão	Descrição
`zonedMode`	string	EBCDIC_STRICT	O modo para codificar ou decodificar tipos de dados zoneados. Os valores permitidos são: `EBCDIC_STRICT` , `EBCDIC_MODIFIED` e `AS400`.
`commands-off`	string		Lista de comandos a serem desativados, separados por vírgula. Os valores permitidos são: `PGM_BASIC` ,`RCVMSG`,`SNDRCVF`,`CHGVAR`,`QRCVDTAQ` e `SNDDTAQ`. Útil quando você deseja desativar ou substituir um programa existente. `PGM_BASIC` é um programa específico do AWS Blu Age Runtime projetado para fins de depuração.
`forcedDate`	string		Força a data até a data fornecida, se houver uma.

gapwalk-hierarchical-support.guerra

Essa aplicação web opcional contém suporte a transações IMS/MFS.

Esta tabela fornece uma visão exaustiva dos parâmetros-chave/valores dessa aplicação.

Chave	Tipo	Valor padrão	Descrição
`logging.config`	Path	classpath:logback-utility.xml	Chave padrão para a referência ao arquivo de configuração de logback. Outras chaves de log padrão também estão disponíveis.
`spring.jta.enabled`	boolean	false	Chave Padrão. Se o modo de suporte da fonte de dados não for static-xa, a configuração automática das transações do Spring JTA deverá ser desativada.
`jhdb.configuration.context.encoding`	string		A codificação JHDB (Java Hierarchical Database). Espera uma string de codificação válida `CP1047`, `IBM930`, `ASCII`, `UTF-8`...
`jhdb.checkpointPersistence`	string	nenhuma	O modo de persistência do ponto de verificação. Os valores permitidos são: `none`, `add` e `end`. Use `add` para manter os pontos de verificação quando um novo for criado

Chave	Tipo	Valor padrão	Descrição
			e adicionado ao registro. Use end para manter o ponto de verificação no desligamento do servidor. Quaisquer outros valores desativam a persistência. Observe que sempre que um novo ponto de verificação for adicionado ao registro, todos os pontos de verificação existentes serão serializados e o arquivo será apagado. Não é um acréscimo aos dados existentes no arquivo. Portanto, dependendo do número de pontos de verificação, isso pode ter alguns efeitos no desempenho.

Configurando a segurança para aplicativos Gapwalk

Os tópicos a seguir descrevem como proteger os aplicativos Gapwalk.

É sua responsabilidade fornecer a configuração correta para garantir que o uso da estrutura AWS Blu Age seja seguro.

Todos os recursos relacionados à segurança estão desativados por padrão. Para ativar a autenticação (e CSRF, XSS, CSP etc.), `gapwalk-application.security` defina como e para. `enabled` `gapwalk-application.security.identity` `oauth`

Tópicos

- Configurando a acessibilidade do URI para aplicativos Gapwalk
- Configurar autenticação para aplicações do Gapwalk

Configurando a acessibilidade do URI para aplicativos Gapwalk

Este tópico descreve como configurar a filtragem de URIs para aplicativos Gapwalk. Não requer um provedor de identidade (IdP).

Para bloquear uma lista de URIs, adicione as duas linhas a seguir à `application-main.yml` do seu aplicativo modernizado, substituindo *`URI-1`*, *`URI-2`* e assim por diante pelos URIs que você deseja bloquear.

```
gapwalk-application.security.filterURIs: enabled
gapwalk-application.security.blockedURIs: URI-1, URI-2, URI-3
```

Configurar autenticação para aplicações do Gapwalk

Os tópicos a seguir descrevem como configurar a autenticação OAuth2 para aplicativos Gapwalk usando um provedor de identidade (IdP), como Cognito ou Keycloak.

Tópicos

- Autenticação Gapwalk OAuth2 com o Amazon Cognito
- Autenticação Gapwalk OAuth2 com Keycloak

Autenticação Gapwalk OAuth2 com o Amazon Cognito

Este tópico descreve como configurar a autenticação OAuth2 para aplicativos Gapwalk usando o Amazon Cognito como provedor de identidade (IdP).

Pré-requisitos

Neste tutorial, usaremos o Amazon Cognito como IdP e PlanetDemo como projeto modernizado.

Você pode usar qualquer outro provedor de identidade externo. As ClientRegistration informações devem ser obtidas do seu IdP e são necessárias para a autenticação do Gapwalk. Para obter mais informações, consulte o Guia do desenvolvedor do Amazon Cognito.

As ClientRegistration informações:

client-id

O ID da ClientRegistration. Em nosso exemplo, será PlanetsDemo.

client-secret

O segredo do seu cliente.

endpoint de autorização

O URI do endpoint de autorização para o servidor de autorização.

endpoint de token

O URI do endpoint do token para o servidor de autorização.

endpoint jwks

O URI usado para obter a chave web JSON (JWK) que contém as chaves para validar a assinatura web JSON emitida pelo servidor de autorização.

URI de redirecionamento

O URI para o qual o servidor de autorização redireciona o usuário final se o acesso for concedido.

Configuração do Amazon Cognito

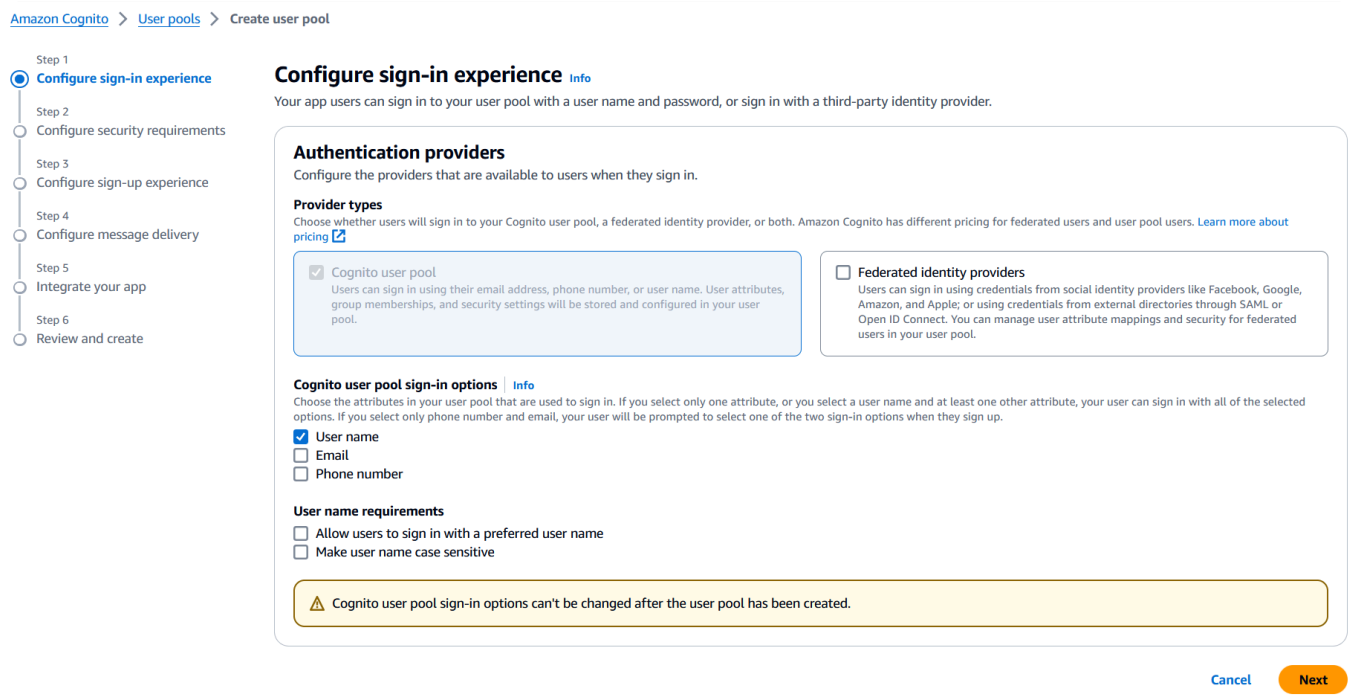
Primeiro, criaremos e configuraremos um grupo de usuários e usuários do Amazon Cognito que usaremos com nosso aplicativo Gapwalk implantado para fins de teste.

> **Note**
>
> Se você estiver usando outro IdP, você pode pular esta etapa.

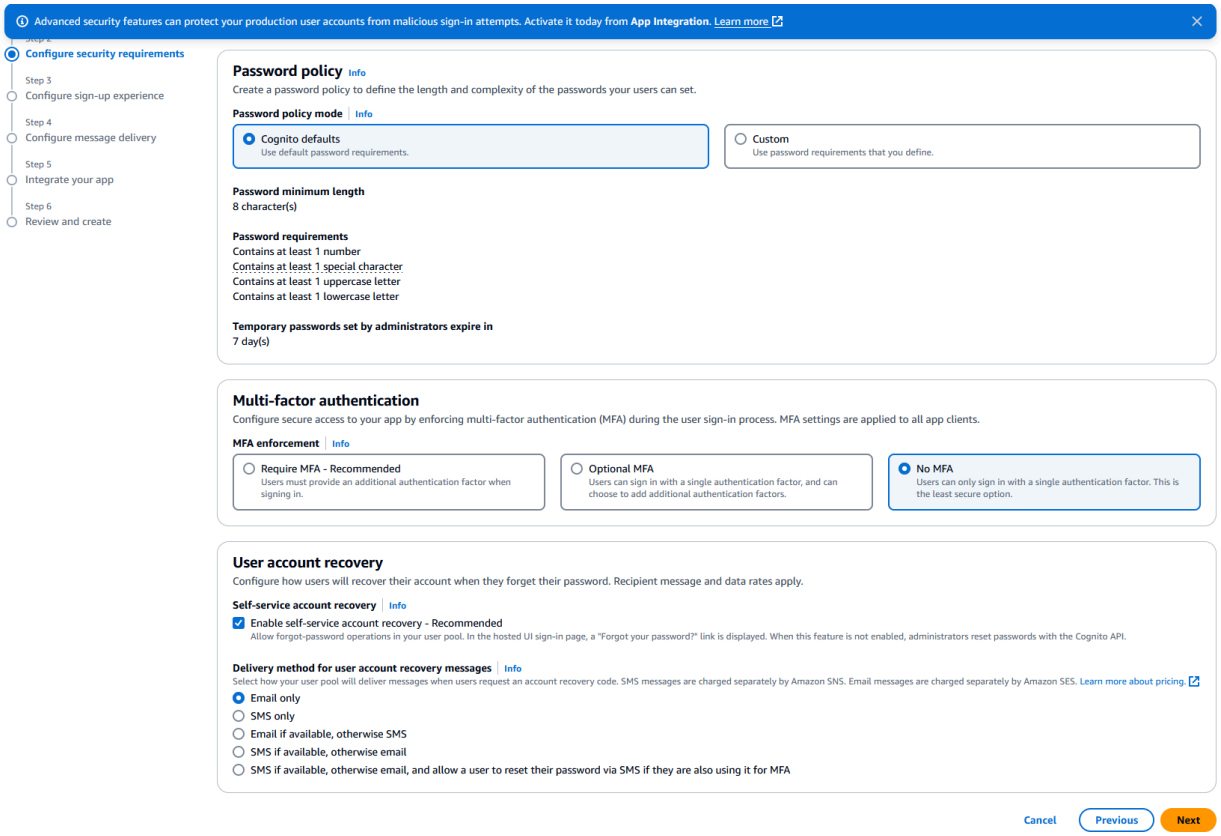
Criar grupo de usuários

1. Acesse o Amazon Cognito em AWS Management Console e autentique-se usando suas credenciais. AWS

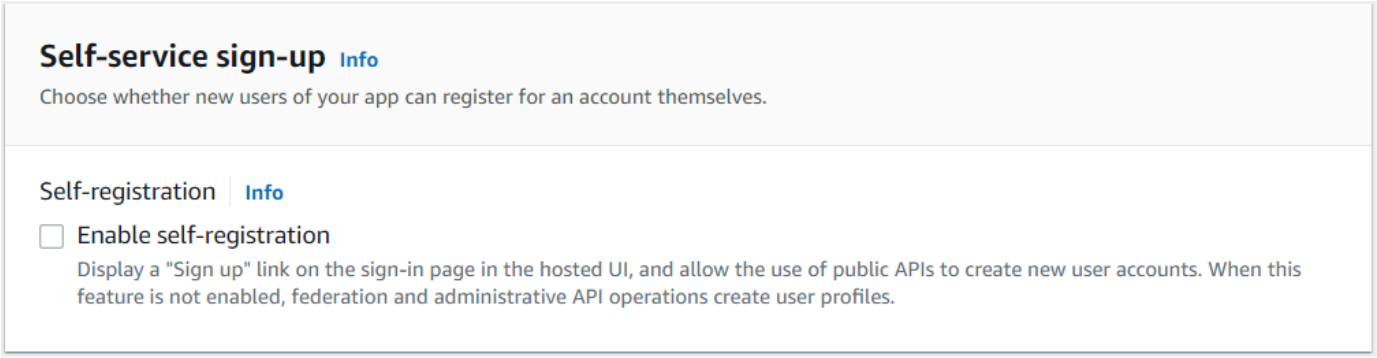
2. Escolha Grupos de usuários.
3. Selecione Criar um grupo de usuários.
4. Em Configurar a experiência de login, mantenha o tipo de provedor padrão do grupo de usuários do Cognito. Você pode escolher uma ou várias opções de login do grupo de usuários do Cognito; por enquanto, escolha Nome do usuário e escolha Próximo.



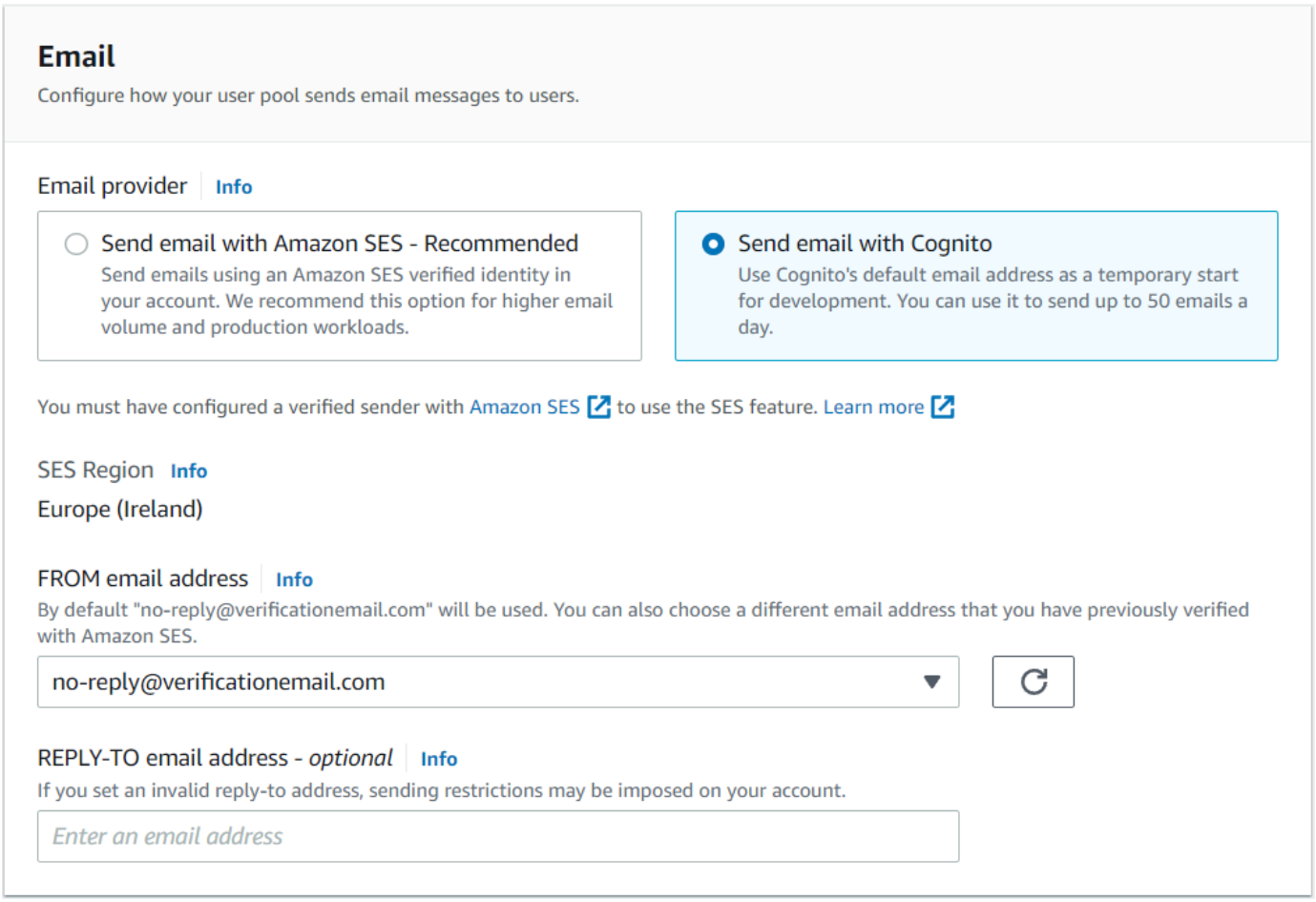
5. Em Configurar requisitos de segurança, mantenha os padrões e desative a autenticação multifator escolhendo Sem MFA e, em seguida, escolha Avançar.



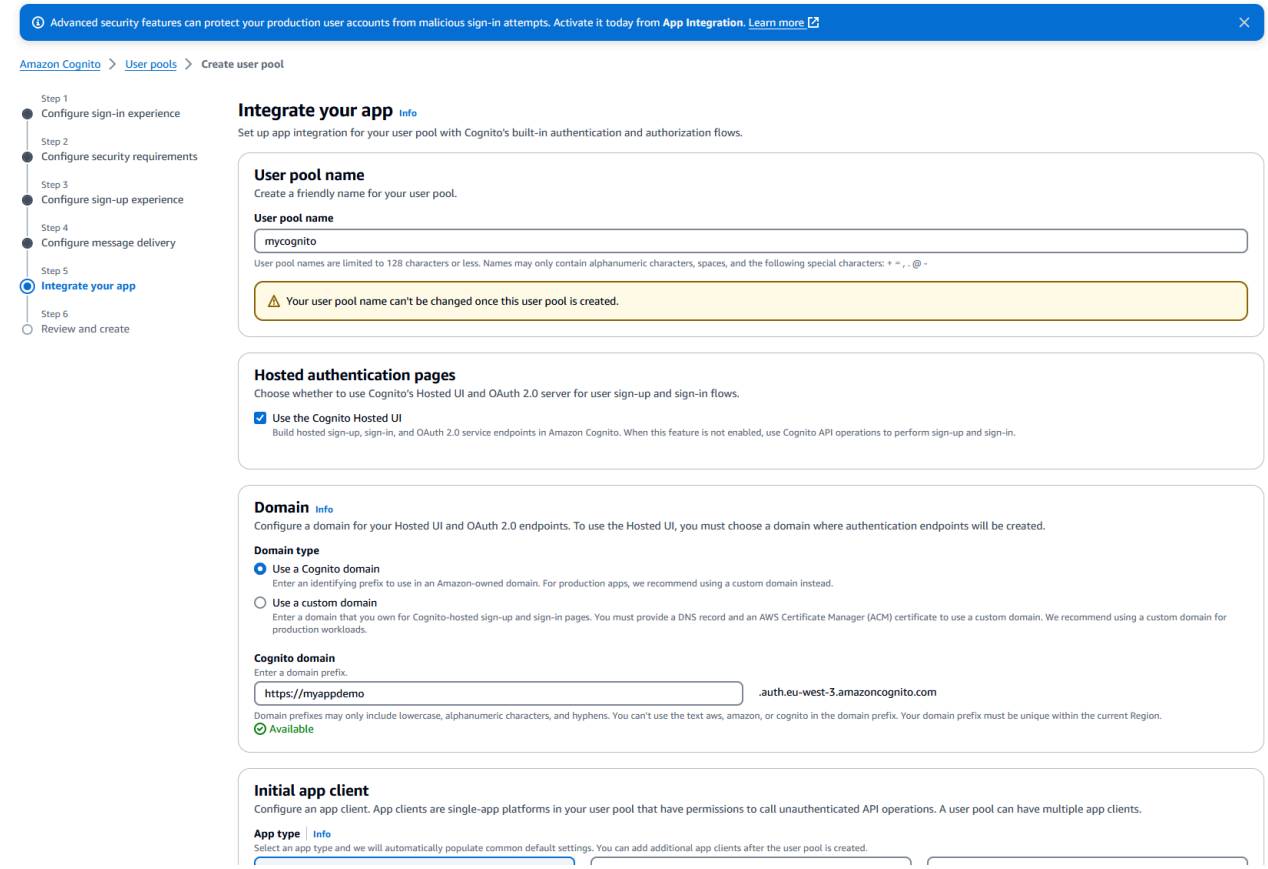
6. Como medida de segurança, desative Habilitar autorregistro e escolha Avançar.



7. Escolha Enviar e-mail com o Cognito e, em seguida, escolha Avançar.



8. Em Integrar seu aplicativo, especifique um nome para seu grupo de usuários. Nas páginas de autenticação hospedada, escolha Usar a interface do usuário hospedada do Cognito.

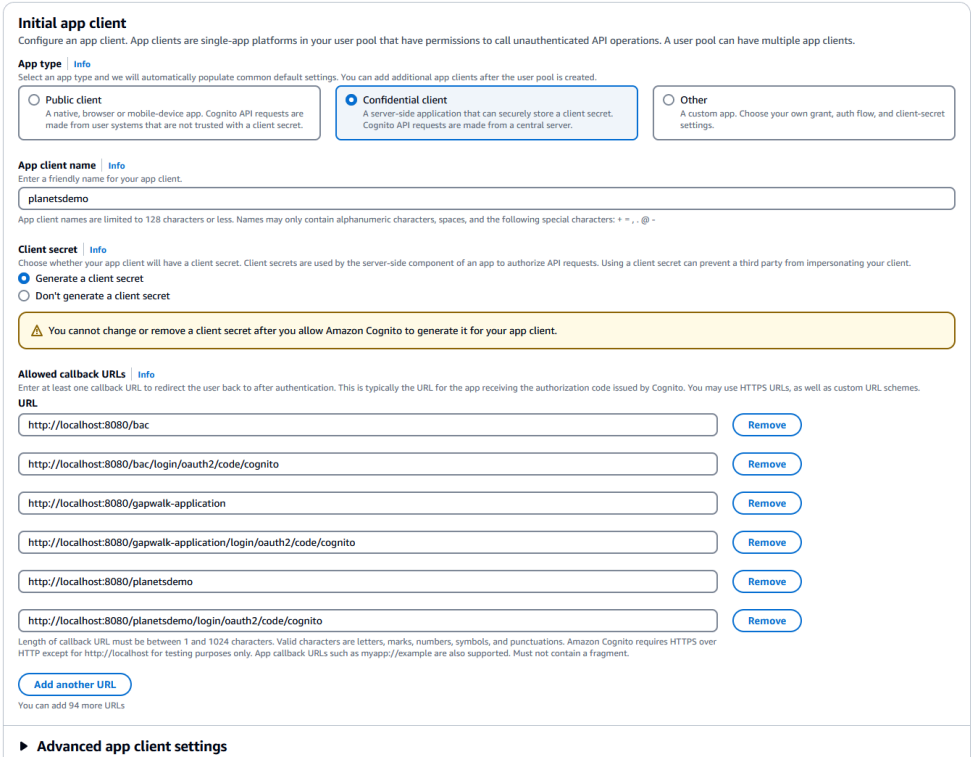


9. Para simplificar, em Domínio, escolha Usar um domínio do Cognito e insira um prefixo de domínio; por exemplo,. `https://planetsdemo` A aplicação de demonstração deve ser adicionada como cliente.

    1. Em Cliente inicial da aplicação, escolha Cliente confidencial. Insira um nome de cliente do aplicativo, como**`planetsdemo`**, e escolha Gerar uma chave secreta do cliente.
    2. Em URL de retorno de chamada permitida, insira a URL para a qual redirecionar o usuário após a autenticação. O URL deve terminar com`/login/oauth2/code/cognito`. Por exemplo, para nossos aplicativos e aplicativos de back-end Gapwalk e BAC:

    ```
    http://localhost:8080/bac
          http://localhost:8080/bac/login/oauth2/code/cognito
          http://localhost:8080/gapwalk-application
          http://localhost:8080/gapwalk-application/login/oauth2/code/cognito
          http://localhost:8080/planetsdemo
          http://localhost:8080/planetsdemo/login/oauth2/code/cognito
    ```

    Você pode editar o URL posteriormente.



3. Em URLs de saída permitidos, insira o URL da página de saída para a qual você deseja que o Amazon Cognito redirecione quando seu aplicativo desconectar os usuários. Por exemplo, para aplicativos de back-end Gapwalk e BAC:

```
http://localhost:8080/bac/logout
http://localhost:8080/gapwalk-application/logout
http://localhost:8080/planetsdemo/logout
```

   Você pode editar o URL posteriormente.

4. Mantenha os valores padrão nas configurações avançadas do cliente de aplicativo e nas seções Permissões de leitura e gravação de atributos.
5. Escolha Próximo.

10. Em Revisar e criar, verifique suas opções e escolha Criar grupo de usuários.

Para obter mais informações, consulte Criar um grupo de usuários.

Criação de usuários

Como o autorregistro está desativado, crie um usuário do Amazon Cognito. Navegue até o Amazon Cognito no AWS Management Console. Escolha o grupo de usuários que você criou e, em Usuários, escolha Criar usuário.

Em Informações do usuário, escolha Enviar um convite por e-mail, insira um nome de usuário e um endereço de e-mail e escolha Gerar uma senha. Selecione Criar usuário.

Criação de funções

Na guia Grupos, crie 3 grupos (SUPER_ADMIN, ADMIN e USER) e associe seu usuário a um ou mais desses grupos. Posteriormente, essas funções são mapeadas para ROLE_SUPER_ADMIN, ROLE_ADMIN e ROLE_USER pelo aplicativo Gapwalk para possibilitar o acesso a algumas chamadas restritas de API REST.

Integrando o Amazon Cognito ao aplicativo Gapwalk

Agora que seu grupo de usuários e usuários do Amazon Cognito estão prontos, acesse o `application-main.yml` arquivo do seu aplicativo modernizado e adicione o seguinte código:

```
gapwalk-application.security: enabled
gapwalk-application.security.identity: oauth
gapwalk-application.security.issuerUri: https://cognito-idp.<region-id>.amazonaws.com/
<pool-id>
gapwalk-application.security.domainName: <your-cognito-domain>
gapwalk-application.security.localhostWhitelistingEnabled: false

spring:
  security:
    oauth2:
      client:
        registration:
          cognito:
            client-id: <client-id>
            client-name: <client-name>
            client-secret: <client-secret>
            provider: cognito
            authorization-grant-type: authorization_code
            scope: openid
            redirect-uri: "<redirect-uri>"
        provider:
          cognito:
            issuer-uri: ${gapwalk-application.security.issuerUri}
```

```
            authorization-uri: ${gapwalk-application.security.domainName}/oauth2/
authorize
            jwk-set-uri: ${gapwalk-application.security.issuerUri}/.well-known/
jwks.json
            token-uri: ${gapwalk-application.security.domainName}/oauth2/token
            user-name-attribute: username
      resourceserver:
        jwt:
          jwk-set-uri: ${gapwalk-application.security.issuerUri}/.well-known/jwks.json
```

Substitua os seguintes espaços reservados conforme descrito:

1. Acesse o Amazon Cognito em AWS Management Console e autentique-se usando suas credenciais. AWS
2. Escolha Grupos de usuários e escolha o grupo de usuários que você criou. Você pode encontrar seu *`pool-id`* em ID do grupo de usuários.
3. Escolha Integração de aplicativos, onde você pode encontrar sua *`your-cognito-domain`*, acesse Clientes e análises de aplicativos e escolha seu aplicativo.
4. Em Cliente do aplicativo: YourApp, você pode encontrar o ***`nome do cliente, o ID do cliente e o segredo do cliente`*** (Mostrar segredo ***`do cliente`***).
5. *`region-id corresponde ao ID`* da AWS região em que você criou seu grupo de usuários e usuários do Amazon Cognito. Exemplo: `eu-west-3`.
6. Para *`redirect-uri`*, insira o URI que você especificou para URL de retorno de chamada permitido. Em nosso exemplo, é`http://localhost:8080/planetsdemo/login/oauth2/code/cognito`.

Agora você pode implantar seu aplicativo Gapwalk e usar o usuário criado anteriormente para entrar no seu aplicativo.

Autenticação Gapwalk OAuth2 com Keycloak

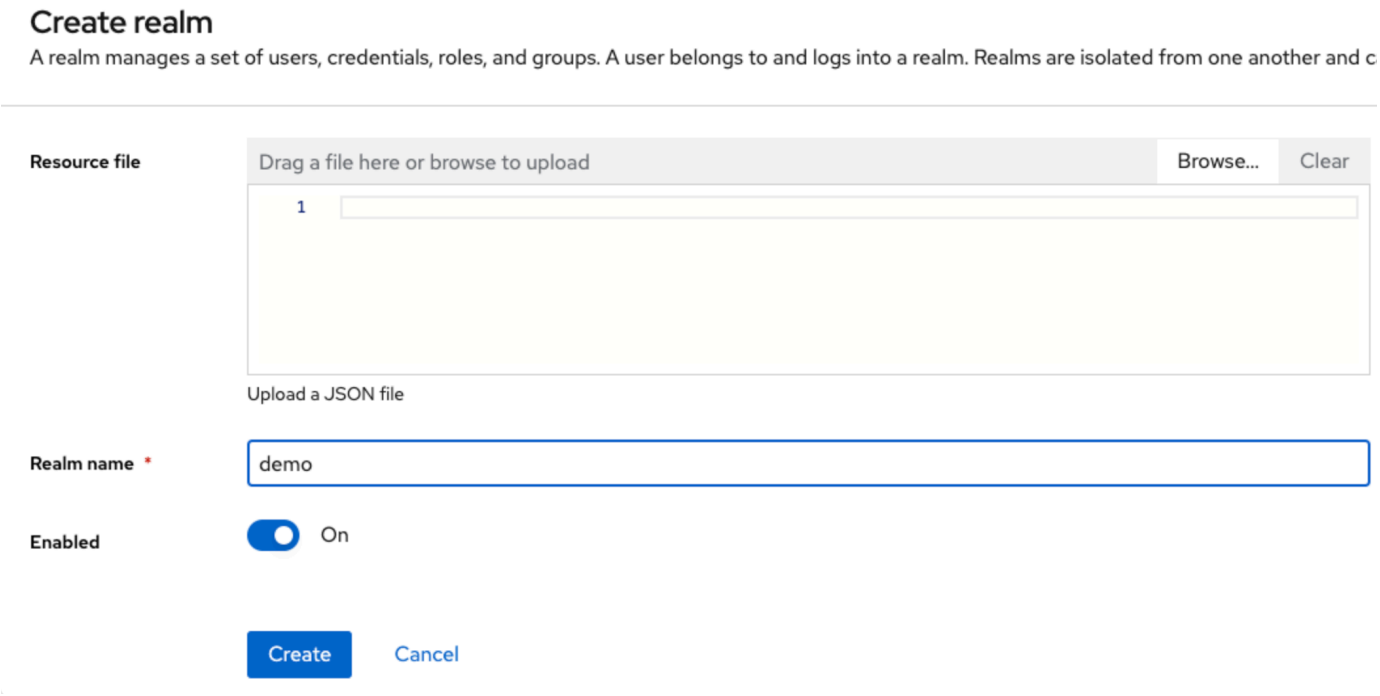
Este tópico descreve como configurar a autenticação OAuth2 para aplicativos Gapwalk usando o Keycloak como provedor de identidade (IdP). Neste tutorial, usamos o Keycloak 24.0.0.

Pré-requisitos

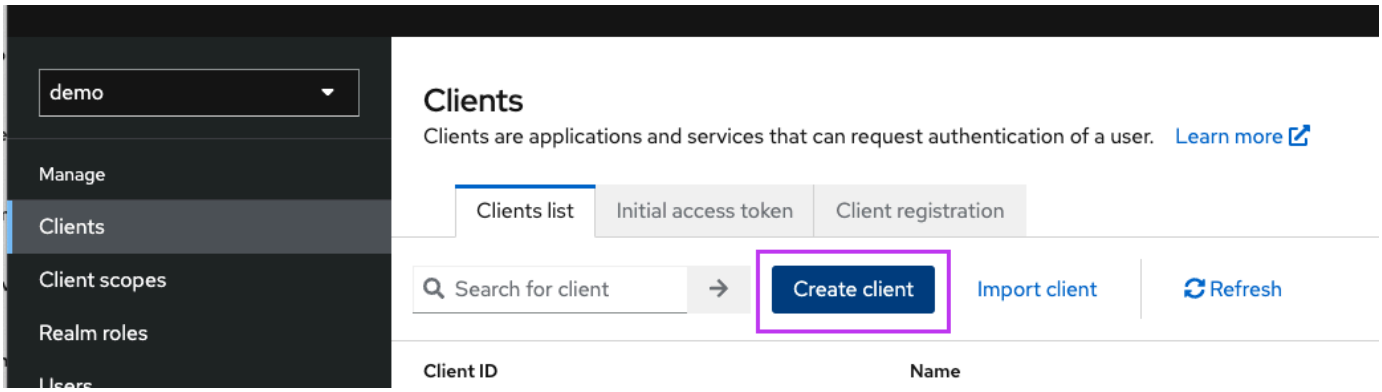
- Capa de chave
- Aplicação Gapwalk

Configuração do Keycloak

1. Acesse o painel do Keycloak em seu navegador da web. As credenciais padrão são admin/admin. Vá para a barra de navegação superior esquerda e crie um território com o nome**`demo`**, conforme mostrado na imagem a seguir.



2. Crie um cliente com o nome**`app-demo`**.



`localhost:8080`Substitua pelo endereço do seu aplicativo Gapwalk

General settings

Client ID * app-demo

Name

Description

Always display in UI Off

Access settings

Root URL http://localhost:8080

Home URL

Valid redirect URIs
- http://localhost:8080/*
- https://localhost:8080/*

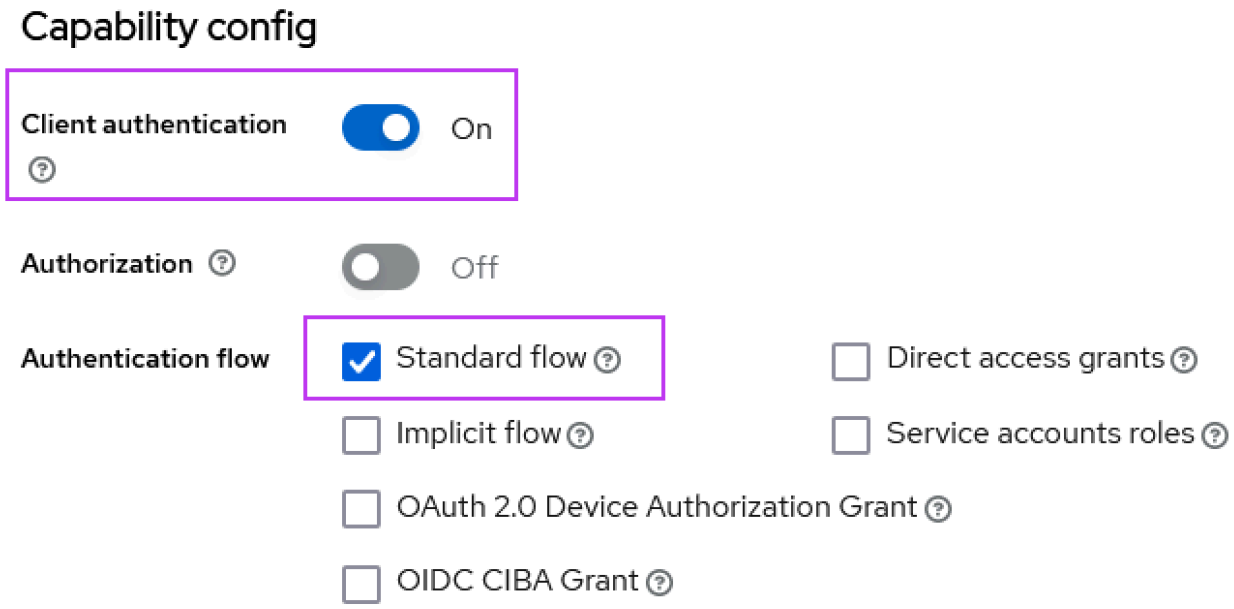
Add valid redirect URIs

Valid post logout redirect URIs
- http://localhost:8080/*
- https://localhost:8080/*

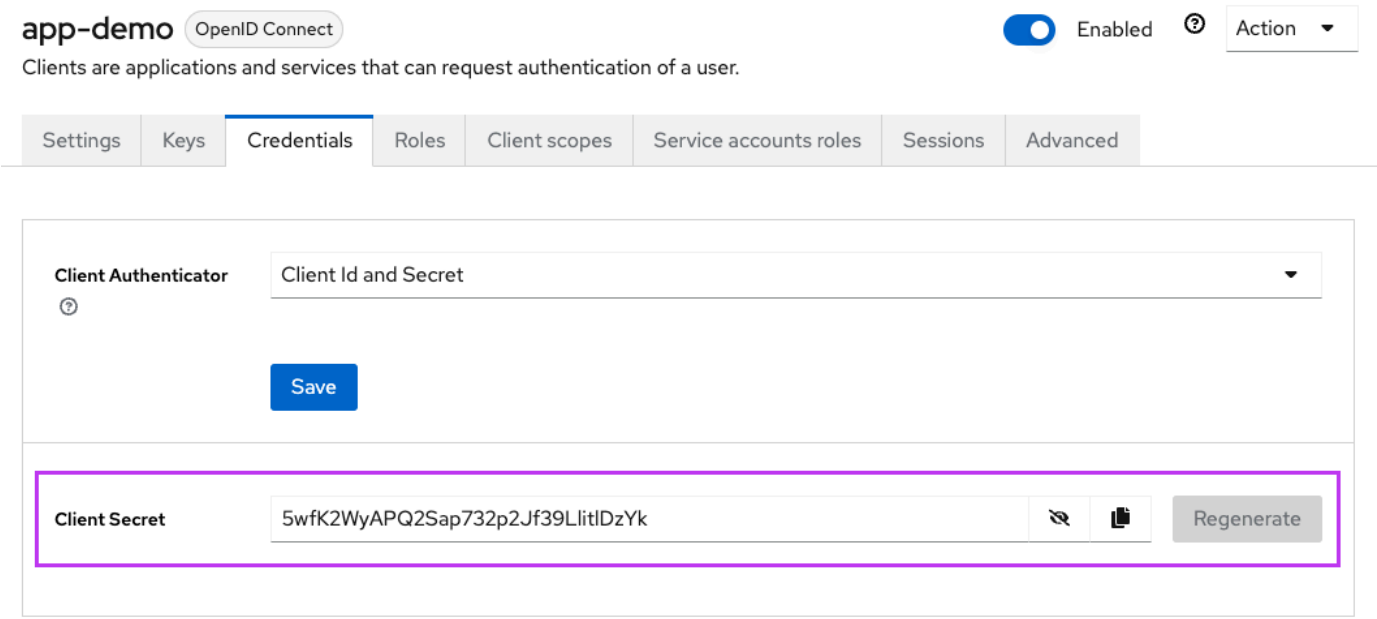
Add valid post logout redirect URIs

Web origins
- +

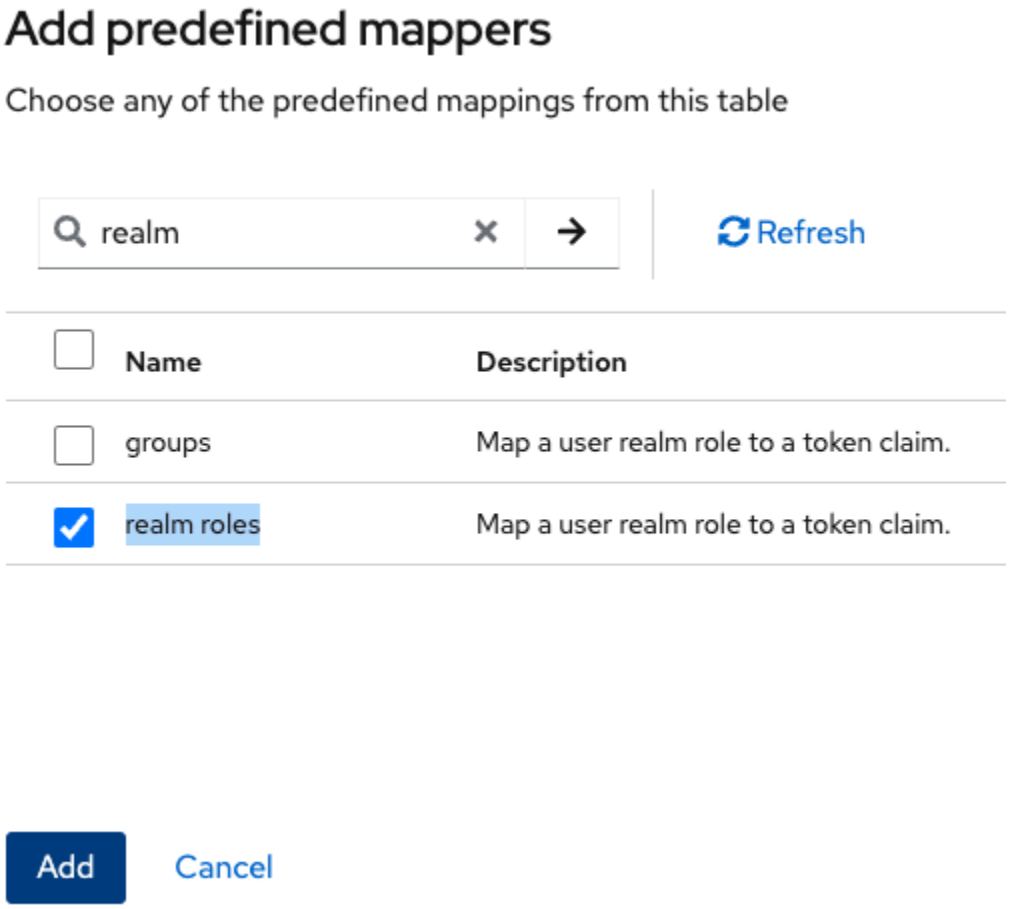
Add web origins



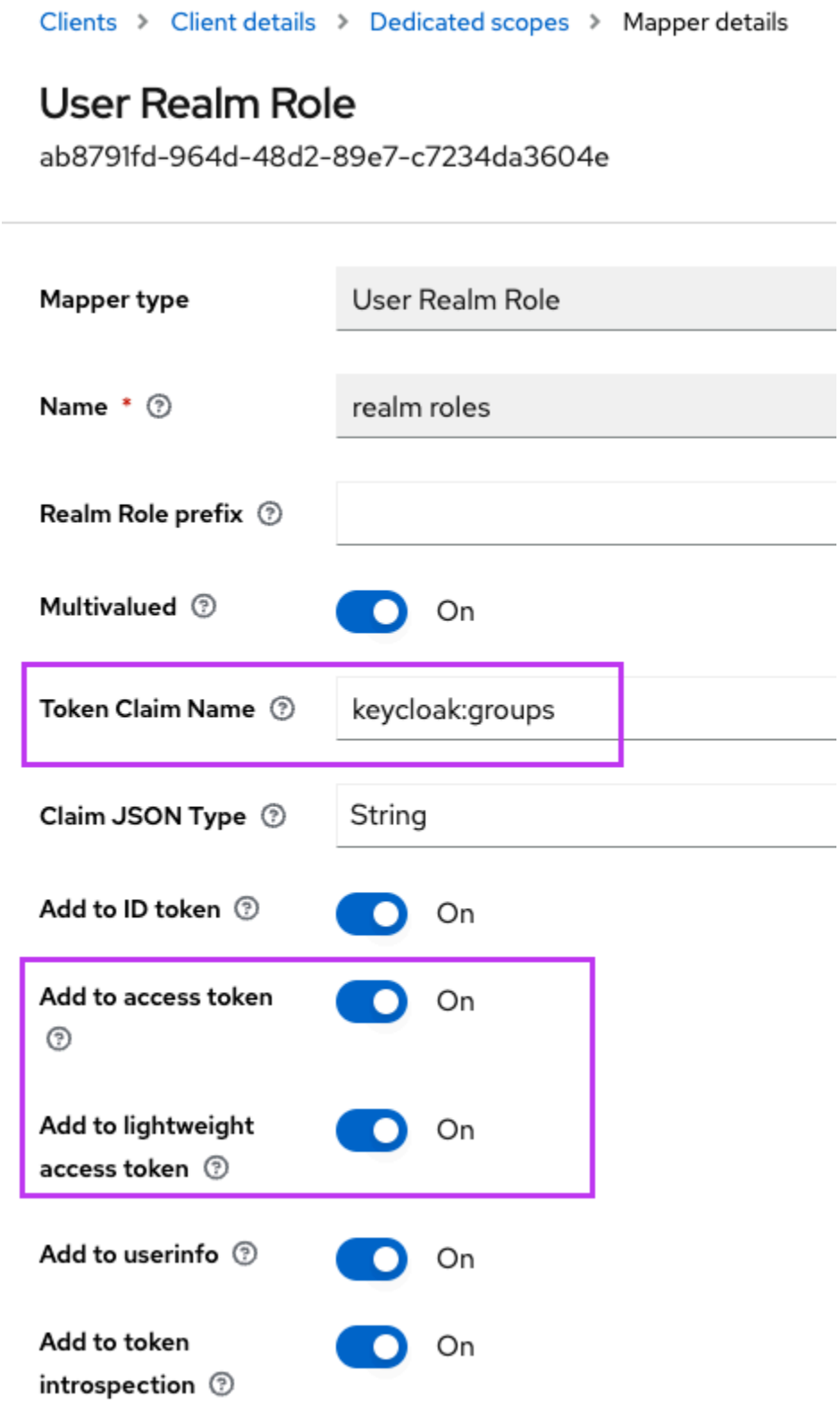
3. Para obter o segredo do seu cliente, escolha Clientes, depois app-demo e depois Credenciais.



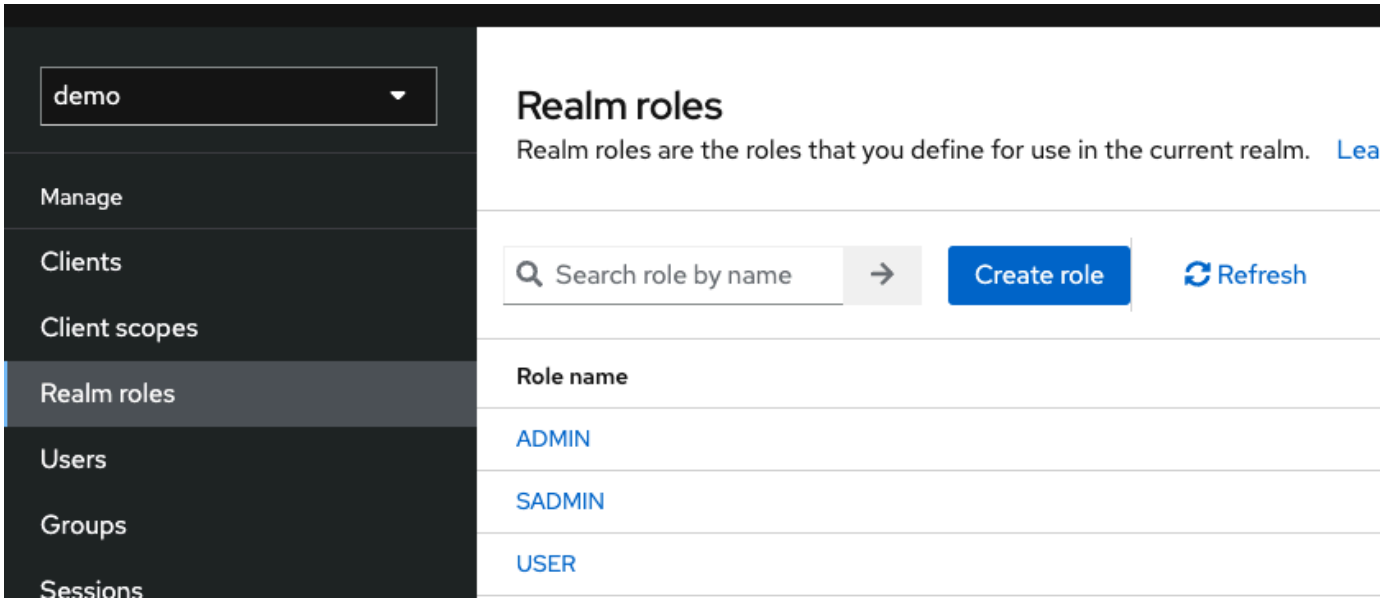
4. Escolha Clientes, depois Escopos do cliente e, em seguida, Adicionar mapeador predefinido. Escolha funções no reino.



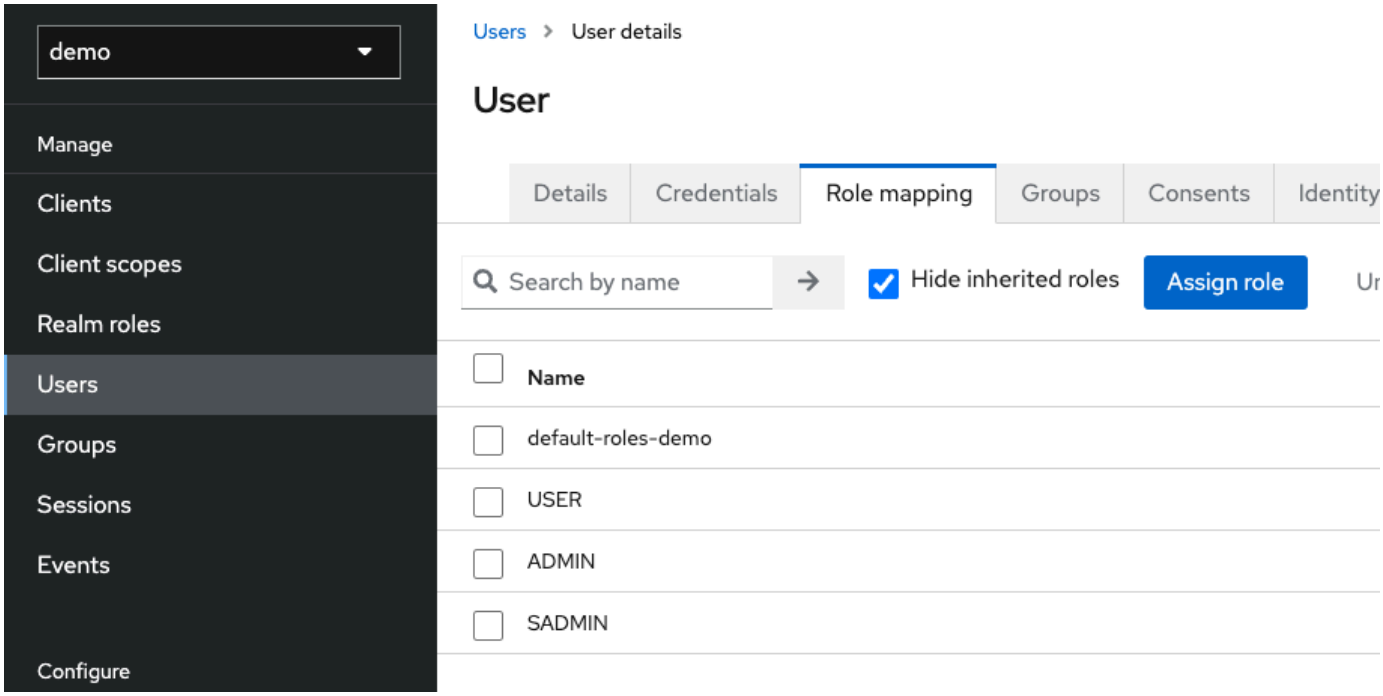
5. Edite sua função de região com a configuração mostrada na imagem a seguir.


Clients > Client details > Dedicated scopes > Mapper details
User Realm Role
ab8791fd-964d-48d2-89e7-c7234da3604e
Mapper type
User Realm Role
Name *
realm roles
Realm Role prefix
Multivalued
On
Token Claim Name
keycloak:groups
Claim JSON Type
String
Add to ID token
On
Add to access token
On
Add to lightweight access token
On
Add to userinfo
On
Add to token introspection
On

6. Lembre-se do nome de reivindicação de token definido. Você precisará desse valor na definição de configurações do Gapwalk para a `gapwalk-application.security.claimGroupName` propriedade.



7. Escolha as funções do Realms e crie 3 funções: **`SUPER_ADMINADMIN`**, e. **`USER`** Essas funções são posteriormente mapeadas para `ROLE_SUPER_ADMINROLE_ADMIN`, e `ROLE_USER` pelo aplicativo Gapwalk para poder acessar algumas chamadas restritas de API REST.



Integrando o Keycloak ao aplicativo Gapwalk

Edite o seu da `application-main.yml` seguinte forma:

```
gapwalk-application.security: enabled
gapwalk-application.security.identity: oauth
gapwalk-application.security.issuerUri: http://<KEYCLOAK_SERVER_HOSTNAME>/realms/
<YOUR_REALM_NAME>
gapwalk-application.security.claimGroupName: "keycloak:groups"

gapwalk-application.security.userAttributeName: "preferred_username"
# Use "username" for cognito,
#     "preferred_username" for keycloak
#      or any other string
gapwalk-application.security.localhostWhitelistingEnabled: false

spring:
  security:
    oauth2:
      client:
        registration:
          demo:
            client-id: <YOUR_CLIENT_ID>
            client-name: Demo App
            client-secret: <YOUR_CLIENT_SECRET>
            provider: keycloak
            authorization-grant-type: authorization_code
            scope: openid
            redirect-uri: "{baseUrl}/login/oauth2/code/{registrationId}"
        provider:
          keycloak:
            issuer-uri: ${gapwalk-application.security.issuerUri}
            authorization-uri: ${gapwalk-application.security.issuerUri}/protocol/
openid-connect/auth
            jwk-set-uri: ${gapwalk-application.security.issuerUri}/protocol/openid-
connect/certs
            token-uri: ${gapwalk-application.security.issuerUri}/protocol/openid-
connect/token
            user-name-attribute: ${gapwalk-application.security.userAttributeName}
      resourceserver:
        jwt:
          jwk-set-uri: ${gapwalk-application.security.issuerUri}/protocol/openid-
connect/certs
```

Substitua <KEYCLOAK_SERVER_HOSTNAME><YOUR_REALM_NAME>,<YOUR_CLIENT_ID>, e por <YOUR_CLIENT_SECRET>seu nome de host do servidor Keycloak, seu nome de domínio, seu ID de cliente e seu segredo de cliente.

AWS APIs de tempo de execução do Blu Age

O AWS Blu Age Runtime usa vários aplicativos da web para expor endpoints REST, fornecendo maneiras de interagir com os aplicativos modernizados usando clientes REST (por exemplo, chamando trabalhos usando um agendador).

O objetivo deste documento é listar os endpoints REST disponíveis, fornecendo detalhes sobre:

- O papel deles
- A maneira de usá-los adequadamente

A lista de endpoints é organizada em categorias, dependendo da natureza do serviço fornecido e da aplicação web que expõe os endpoints.

Presumimos que você já tenha um conhecimento básico do uso de endpoints REST usando ferramentas dedicadas, como POSTMAN, Thunder Client, CURL, navegadores da web, etc.) ou escrever seu próprio trecho de código para fazer uma chamada de API.

Tópicos

Criação de URLs

Cada aplicação da web abaixo está definindo um caminho raiz, compartilhado por todos os endpoints. Em seguida, cada endpoint adiciona seu próprio caminho dedicado. O URL resultante a ser usado é o resultado da concatenação dos caminhos. Por exemplo, considerando o primeiro endpoint a aplicação Gapwalk, temos:

- `/gapwalk-application` para o caminho raiz da aplicação web.

- `/scripts` para o caminho de endpoint dedicado.

O URL resultante a ser usado será `http://server:port/gapwalk-application/scripts`

servidor

aponta para o nome do servidor (aquele que hospeda a determinada aplicação web).

porta

a porta exposta pelo servidor.

Aplicação Gapwalk

Os endpoints da aplicação Web Gapwalk usam o caminho raiz `/gapwalk-application`.

Tópicos

- Endpoints relacionados a trabalhos em lote (JCLs modernizados e similares)
- Métricas para endpoints do
- Outros endpoints
- Endpoints relacionados ao Job Queues

Endpoints relacionados a trabalhos em lote (JCLs modernizados e similares)

Os trabalhos em lote podem ser executados de forma síncrona ou assíncrona (veja os detalhes abaixo). Os trabalhos em lote estão sendo executados usando scripts groovy que são o resultado da modernização dos scripts antigos (JCL).

Tópicos

- Listar scripts implantados
- Inicie um script de forma síncrona
- Inicie um script de forma assíncrona
- Listando scripts acionados
- Recuperando detalhes da execução do trabalho
- Listando scripts lançados de forma assíncrona que podem ser eliminados
- Listando scripts lançados de forma síncrona que podem ser eliminados

- Matando a execução de um determinado trabalho
- Listando os pontos de verificação existentes para capacidade de reinicialização
- Reiniciando um trabalho (de forma síncrona)
- Reiniciando um trabalho (de forma assíncrona)
- Definir o limite de threads para execuções de trabalhos assíncronas

Listar scripts implantados

- Método compatível: GET
- Caminho: `/scripts`
- Argumentos: nenhum
- Esse endpoint retorna a lista de scripts groovy implantados no servidor, como uma string. Esse endpoint deve ser usado principalmente em um navegador da Web, já que a String resultante é uma página HTML, com links ativos (um link por script iniciável — veja o exemplo abaixo).

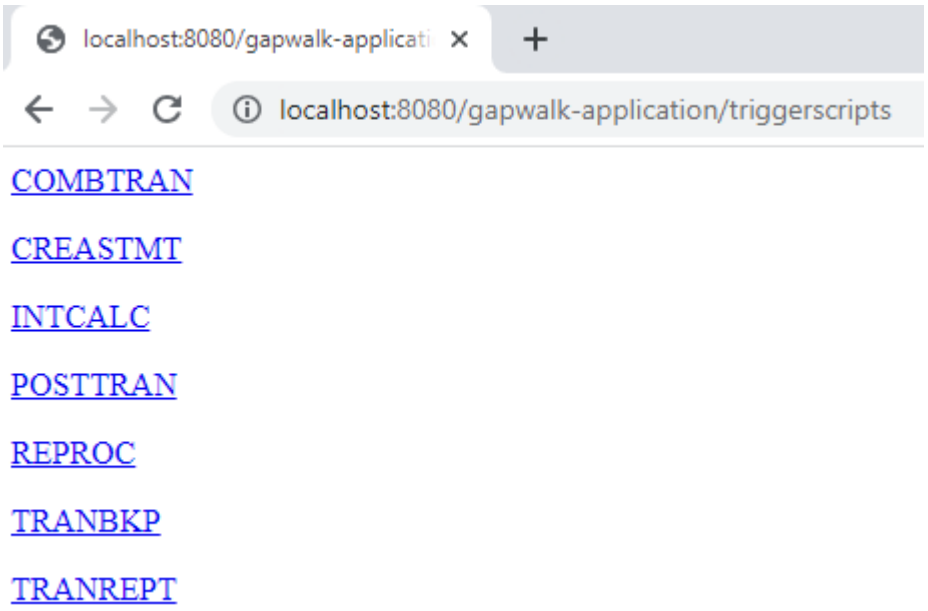
Resposta de exemplo:

```
<p><a href=./script/COMBTRAN>COMBTRAN</a></p><p><a href=./script/CREASTMT>CREASTMT</
a></p><p><a href=./script/INTCALC>INTCALC</a></p><p><a href=./script/
POSTTRAN>POSTTRAN</a></p><p><a href=./script/REPROC>REPROC</a></p><p><a href=./script/
TRANBKP>TRANBKP</a></p><p><a href=./script/TRANREPT>TRANREPT</a></p><p><a href=./
script/functions>functions</a></p>
```

> **Note**
>
> Os links representam o URL a ser usado para iniciar cada script listado de forma síncrona.

- Método compatível: GET
- Caminho: `/triggerscripts`
- Argumentos: nenhum
- Esse endpoint retorna a lista de scripts groovy implantados no servidor, como uma string. Esse endpoint deve ser usado principalmente em um navegador da Web, já que a String resultante é uma página HTML, com links ativos (um link por script iniciável — veja o exemplo abaixo).

Ao contrário da resposta anterior do endpoint, os links representam o URL a ser usado para iniciar cada script listado de forma assíncrona.



Inicie um script de forma síncrona

Esse endpoint tem duas variantes com caminhos dedicados para uso de GET e POST (veja abaixo).

- Método compatível: GET
- Caminho: `/script/{scriptId:.+}`
- Método compatível: POST
- Caminho: `/post/script/{scriptId:.+}`
- Argumentos:
  - identificador do script a ser lançado
  - opcionalmente: parâmetros a serem passados para o script, usando parâmetros de solicitação (vistos como `Map<String,String>`). Os parâmetros fornecidos serão adicionados automaticamente às ligações do script groovy invocado.
- A chamada iniciará o script com o identificador fornecido, usando parâmetros extras, se fornecidos, e aguardará a conclusão da execução do script antes de retornar uma mensagem (`String`) que será:
  - “Concluído.” (se a execução do trabalho ocorreu sem problemas).

- Uma mensagem de erro JSON com detalhes sobre o que deu errado durante a execução do trabalho. Mais detalhes podem ser recuperados dos logs do servidor para entender o que deu errado com a execução do trabalho.

```
{
    "exitCode": -1,
    "stepName": "STEP15",
    "program": "CBACT04C",
    "status": "Error"
}
```

Analisando os logs do servidor, podemos descobrir que esse é um problema de implantação (o programa esperado não foi implantado corretamente e, portanto, não pode ser encontrado, fazendo com que a execução do trabalho falhe):

```
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.s.BatchWebController          - --> executing script INTCALC
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.s.BatchWebController          - Bound jobContext 419695287 - GDGEventsQueueHandler :907380469
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.s.ScriptControlTower          - Added jobExecutor [a65c2791-864f-43c9-972a-b5f2353389e6] to Sync Script Control Tower.
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.j.s.JobExecutor              - a65c2791-864f-43c9-972a-b5f2353389e6 - worker :Thread-26 [1547512424]
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.j.s.JobExecutor              - Triggered script: INTCALC - [a65c2791-864f-43c9-972a-b5f2353389e6] - jobContext [419695287]
2023-06-09_10-27-29-613 | [JOB] INTCALC - Started
2023-06-09_10-27-29-651 | [STEP] STEP15 - Started
2023-06-09 10:27:29 default        ERROR - c.n.b.g.r.c.i.ExecutionControllerImpl   - Could not find program "CBACT04C" in the program registry.
2023-06-09 10:27:29 default        ERROR - c.n.b.g.r.c.i.ExecutionControllerImpl   - Could not find program "CBACT04C" in the program registry.
2023-06-09_10-27-29-760 | Program not found => not executed !
2023-06-09_10-27-29-761 | [STEP] STEP15 - Ended
2023-06-09_10-27-29-772 | [JOB] INTCALC - Ended
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.j.s.DefaultJobContext         - Job [419695287] - starting final operation
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.j.s.DefaultJobContext         - End of job [419695287]
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.s.ScriptControlTower          - Removed jobExecutor [a65c2791-864f-43c9-972a-b5f2353389e6] from Script Control Tower.
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.s.ScriptControlTower          - Remaining jobExecutors:0
```

> **Note**
>
> As chamadas síncronas devem ser reservadas para trabalhos de curta duração. Trabalhos de longa duração devem ser iniciados de forma assíncrona (consulte o endpoint dedicado abaixo).

Inicie um script de forma assíncrona

- Métodos compatíveis: GET/POST
- Caminho: `/triggerscript/{scriptId:.+}`
- Argumentos:
  - identificador do script a ser lançado
  - opcionalmente: parâmetros a serem passados para o script, usando parâmetros de solicitação (vistos como `Map<String,String>`). Os parâmetros fornecidos serão

adicionados automaticamente ao https://docs.groovy-lang.org/latest/html/api/groovy/lang/Binding.html[bindings] do script groovy invocado.

- Ao contrário do modo síncrono acima, o endpoint não espera a conclusão da execução do trabalho para enviar uma resposta. A execução do trabalho é iniciada de uma só vez, se for possível encontrar um thread disponível para fazer isso, e uma resposta é enviada imediatamente ao chamador, com o id de execução do trabalho, um identificador exclusivo que representa a execução do trabalho, que pode ser usado para consultar o status da execução do trabalho ou forçar o encerramento de uma execução de trabalho que deveria estar com defeito. O formato da resposta é:

```
Triggered script <script identifier> [unique job execution id] @ <date and time>
```

- Como a execução assíncrona do trabalho depende de um número fixo limitado de threads, a execução do trabalho pode não ser iniciada se nenhum thread disponível for encontrado. Nesse caso, a mensagem retornada se parecerá com:

```
Script [<script identifier>] NOT triggered - Thread limit reached (<actual thread
 limit>) - Please retry later or increase thread limit.
```

  Consulte o endpoint `settriggerthreadlimit` abaixo para saber como aumentar o limite de threads.

Resposta de exemplo:

```
Triggered script INTCALC [d43cbf46-4255-4ce2-aac2-79137573a8b4] @ 06-12-2023 16:26:15
```

O identificador exclusivo de execução de tarefas permite recuperar rapidamente as entradas de log relacionadas nos logs do servidor, se necessário. Ele também é usado por vários outros endpoints detalhados abaixo.

Listando scripts acionados

- Métodos compatíveis: GET
- Caminhos: `/triggeredscripts/{status:.+}`, `/triggeredscripts/{status:.+}/{namefilter}`
- Argumentos:

- Status (obrigatório): o status dos scripts acionados a serem recuperados. Os valores possíveis são:
  - `all`: mostre todos os detalhes da execução do trabalho, independentemente de os trabalhos ainda estarem em execução ou não.
  - `running`: mostre somente os detalhes dos trabalhos que estão sendo executados no momento.
  - `done`: mostre somente os detalhes dos trabalhos cuja execução acabou.
  - `killed`: mostre somente os detalhes dos trabalhos cuja execução foi encerrada à força usando o endpoint dedicado (veja abaixo).
  - `triggered`: mostre somente os detalhes dos trabalhos que foram acionados, mas ainda não foram lançados.
  - `failed`: mostre somente os detalhes dos trabalhos cuja execução foi marcada como falhada.
  - _namefilter (opcional) _: recupera somente execuções para o identificador de script fornecido.
- Retorna uma coleção de detalhes de execuções de trabalhos como JSON. Para ter mais informações, consulte Estrutura de mensagens do Job Execution Details.

Resposta de exemplo:

```
[
    {
      "scriptId": "INTCALC",
      "caller": "127.0.0.1",
      "identifier": "d43cbf46-4255-4ce2-aac2-79137573a8b4",
      "startTime": "06-12-2023 16:26:15",
      "endTime": "06-12-2023 16:26:15",
      "status": "DONE",
      "executionResult": "{ \"exitCode\": -1, \"stepName\": \"STEP15\", \"program\":
 \"CBACT04C\", \"status\": \"Error\" }",
      "executionMode": "ASYNCHRONOUS"
    }
  ]
```

Recuperando detalhes da execução do trabalho

- Método compatível: GET
- Caminho: `/getjobexecutioninfo/{jobexecutionid:.+}`

- Argumentos:
  - jobexecutionid (obrigatório): o identificador exclusivo de execução do trabalho para recuperar os detalhes correspondentes da execução do trabalho.
- Retorna: uma string JSON representando um único detalhe da execução do trabalho (consulte Estrutura de mensagens do Job Execution Details) ou uma resposta vazia se nenhum detalhe da execução do trabalho puder ser encontrado para o identificador fornecido.

Listando scripts lançados de forma assíncrona que podem ser eliminados

- Método compatível: GET
- Caminho: `/killablescripts`
- Retorna uma coleção de identificadores de execução de trabalhos que foram iniciados de forma assíncrona e que ainda estão em execução e podem ser eliminados à força (consulte o endpoint `/kill` abaixo).

Listando scripts lançados de forma síncrona que podem ser eliminados

- Método compatível: GET
- Caminho: `/killablesyncscripts`
- Retorna uma coleção de identificadores de execução de trabalhos que foram iniciados de forma síncrona, ainda estão em execução e podem ser eliminados à força (consulte o endpoint `/kill` abaixo).

Matando a execução de um determinado trabalho

- Método compatível: GET
- Caminho: `/kill/{identifier:.+}`
- argumento: identificador de execução do trabalho (obrigatório): o identificador exclusivo de execução do trabalho que aponta para que a execução do trabalho seja eliminada à força.
- Retorna: uma mensagem de texto detalhando o resultado da tentativa de eliminação da execução da tarefa; a mensagem conterá o identificador do script, o identificador exclusivo da execução da tarefa e a data e hora em que a eliminação da execução ocorreu. Se nenhuma execução de trabalho em execução for encontrada para o identificador fornecido, uma mensagem de erro será retornada em vez disso.

⚠ Warning

- O runtime se esforça ao máximo para eliminar bem a execução do trabalho de destino. Portanto, a resposta do endpoint /kill pode levar um pouco de tempo para chegar ao chamador, pois o tempo de execução do AWS Blu Age tentará minimizar o impacto comercial da eliminação do trabalho.
- A eliminação forçada da execução de um trabalho não deve ser feita de ânimo leve, pois pode ter consequências comerciais diretas, incluindo possível perda ou corrupção de dados. Ele deve ser reservado para casos em que a execução de um determinado trabalho tenha ocorrido mal e os meios de remediação de dados estejam claramente identificados.
- A eliminação de um emprego deve levar a investigações adicionais (análise post-mortem) para descobrir o que deu errado e tomar as medidas corretivas adequadas.
- De qualquer forma, a tentativa de eliminar um trabalho em execução será registrada nos logs do servidor com mensagens de nível de aviso.

Listando os pontos de verificação existentes para capacidade de reinicialização

A reinicialização do trabalho depende da capacidade dos scripts de registrar pontos de verificação no `CheckpointRegistry` para rastrear o progresso da execução do trabalho. Se a execução de um trabalho não terminar corretamente e os pontos de verificação de reinicialização tiverem sido registrados, basta reiniciar a execução do trabalho a partir do último ponto de verificação registrado conhecido (sem precisar executar as etapas acima do ponto de verificação).

- Método compatível: GET
- Caminho: `/restarts`
- Retorna a lista de pontos de reinicialização existentes, que podem ser usados para reiniciar um trabalho cuja execução não chegou e terminou corretamente, como uma página html. Se nenhum ponto de verificação foi registrado por nenhum script, o conteúdo da página será “Nenhum ponto de verificação registrado”.

Reiniciando um trabalho (de forma síncrona)

- Método compatível: GET
- Caminho: `/restart/{hashcode}`

- Argumentos: hashcode (inteiro — obrigatório): reinicie uma execução de trabalho abortada anteriormente, usando o hashcode fornecido como valor do ponto de verificação (consulte o `/restarts` endpoint acima para saber como recuperar um valor de ponto de verificação válido).
- Devoluções: veja a descrição da `script` devolução acima.

Reiniciando um trabalho (de forma assíncrona)

- Método compatível: GET
- Caminho: `/triggerrestart/{hashcode}`
- Argumentos: hashcode (inteiro — obrigatório): reinicie uma execução de trabalho abortada anteriormente, usando o hashcode fornecido como valor do ponto de verificação (consulte o `/restarts` endpoint acima para saber como recuperar um valor de ponto de verificação válido).
- Devoluções: veja a descrição da `triggerscript` devolução acima.

Definir o limite de threads para execuções de trabalhos assíncronas

A execução de trabalhos assíncronas depende de um grupo dedicado de threads na JVM. Esse grupo tem um limite fixo em relação ao número de threads disponíveis. O usuário tem a capacidade de ajustar o limite de acordo com as capacidades do host (número de CPUs, memória disponível, etc.). Por padrão, o limite de threads é definido como cinco threads.

- Método compatível: GET
- Caminho: `/settriggerthreadlimit/{threadlimit:.+}`
- Argumento (inteiro): o novo limite de threads a ser aplicado. Deve ser um número inteiro estritamente positivo.
- Retorna uma mensagem (`String`) fornecendo o novo limite de threads e o anterior, ou uma mensagem de erro se o valor limite de threads fornecido não for válido (não é um número inteiro estritamente positivo).

Resposta de exemplo:

```
Set thread limit for Script Tower Control to 10 (previous value was 5)
```

Contando as execuções de trabalhos acionadas atualmente em execução

- Método compatível: GET

- Caminho: `/countrunningtriggeredscripts`
- Retorna uma mensagem indicando o número de trabalhos em execução iniciados de forma assíncrona e o limite de threads (ou seja, o número máximo de trabalhos acionados que podem ser executados simultaneamente).

Resposta de exemplo:

```
0 triggered script(s) running (limit =10)
```

> **Note**
>
> Isso pode ser usado para verificar, antes de iniciar um trabalho, se o limite de threads não foi atingido (o que impediria que o trabalho fosse iniciado).

Limpar informações sobre execuções de trabalhos

As informações de execução do trabalho permanecem na memória do servidor enquanto o servidor estiver ativo. Pode ser conveniente limpar as informações mais antigas da memória, pois elas não são mais relevantes; esse é o propósito desse endpoint.

- Método compatível: GET
- Caminho: `/purgejobinformation/{age:.+}`
- Argumentos: um valor inteiro estritamente positivo que representa a idade em horas das informações a serem eliminadas.
- Retorna uma mensagem com as seguintes informações:
  - Nome do arquivo de limpeza em que as informações de execução do trabalho eliminado estão sendo armazenadas para fins de arquivamento.
  - Número de informações de execução do trabalho eliminadas.
  - Número de informações restantes sobre a execução do trabalho no memorando

Métricas para endpoints do

JVM

Esse endpoint retorna as métricas disponíveis relacionadas à JVM.

- Método compatível: GET
- Caminho: `/metrics/jvm`
- Argumentos: nenhum
- Retorna uma mensagem com as seguintes informações:
  - threadActiveCount: Número de segmentos ativos.
  - jvmMemoryUsed: Memória usada ativamente pela Java Virtual Machine.
  - jvmMemoryMax: Memória máxima permitida para a Java Virtual Machine.
  - jvmMemoryFree: Memória disponível que não está sendo usada atualmente pela Java Virtual Machine.

Sessão

Esse endpoint retorna métricas relacionadas às sessões HTTP abertas atualmente.

- Método compatível: GET
- Caminho: `/metrics/session`
- Argumentos: nenhum
- Retorna uma mensagem com as seguintes informações:
  - sessionCount: Número de sessões de usuário ativas atualmente mantidas pelo servidor.

Lote

- Método compatível: GET
- Caminho: `/metrics/batch`
- Argumentos:
  - startTimestamp (opcional, número): carimbo de data/hora inicial para filtragem de dados.
  - endTimestamp (opcional, número): carimbo de data/hora final para filtragem de dados.
  - page (opcional, número): Número da página para paginação.
  - pageSize (opcional, número): Número de itens por página na paginação.
- Retorna uma mensagem com as seguintes informações:
  - content: lista de métricas de execução em lote.
  - pageNumber: número da página atual na paginação.
  - pageSize: Número de itens exibidos por página.

- totalPages: Número total de páginas disponíveis.
- numberOfElements: Contagem de itens na página atual.
- last: bandeira booleana para a última página.
- first: bandeira booleana para a primeira página.

TRANSACTION

- Método compatível: GET
- Caminho: `/metrics/transaction`
- Argumentos:
  - startTimestamp (opcional, número): carimbo de data/hora inicial para filtragem de dados.
  - endTimestamp (opcional, número): carimbo de data/hora final para filtragem de dados.
  - page (opcional, número): Número da página para paginação.
  - pageSize (opcional, número): Número de itens por página na paginação.
- Retorna uma mensagem com as seguintes informações:
  - content: lista de métricas de execução de transações.
  - pageNumber: número da página atual na paginação.
  - pageSize: Número de itens exibidos por página.
  - totalPages: Número total de páginas disponíveis.
  - numberOfElements: Contagem de itens na página atual.
  - last: bandeira booleana para a última página.
  - first: bandeira booleana para a primeira página.

Outros endpoints

Use esses endpoints para listar programas ou serviços registrados, descobrir o status de saúde e gerenciar transações do JICS.

Tópicos

- Lista de transações JICS disponíveis
- Inicie uma transação JICS
- Iniciar uma transação JICS (alternativa)

Lista de programas registrados

- Método compatível: GET
- Caminho: `/programs`
- Retorna a lista de programas registrados, como uma página html. Cada programa é designado pelo identificador principal do programa. Tanto os programas antigos modernizados quanto os programas utilitários (IDCAMS, IEBGENER, etc.) estão sendo retornados na lista. Observe que os programas utilitários disponíveis dependerão das aplicações Web de utilitários que foram implantados no servidor tomcat. Por exemplo, os programas de suporte do utilitário z/OS podem não estar disponíveis para ativos modernizados do iSeries, pois não são relevantes.

Listando serviços registrados

- Método compatível: GET
- Caminho: `/services`
- Retorna a lista de serviços de runtime registrados, como uma página html. Os serviços fornecidos são fornecidos pelo tempo de execução do AWS Blu Age como utilitários, que podem ser usados, por exemplo, em scripts interessantes. Os serviços de carregamento Blusam (para criar conjuntos de dados Blusam a partir de conjuntos de dados antigos) se enquadram nessa categoria.

Resposta de exemplo:

```
<p>BluesamESDSFileLoader</p><p>BluesamKSDSFileLoader</p><p>BluesamRRDSFileLoader</p>
```

Status de integridade

- Método compatível: GET
- Caminho: `/`
- Retorna uma mensagem simples, indicando que a aplicação gapwalk está funcionando (`Jics application is running.`)

Lista de transações JICS disponíveis

- Método compatível: GET
- Caminho: `/transactions`
- Retorna uma página html listando todas as transações JICS disponíveis. Isso só faz sentido para ambientes com elementos JICS (modernização de elementos antigos do CICS).

Resposta de exemplo:

```
<p>INQ1</p><p>MENU</p><p>MNT2</p><p>ORD1</p><p>PRNT</p>
```

Inicie uma transação JICS

- Métodos compatíveis: GET, POST
- Caminho: `/jicstransrunner/{jtrans:.+}`
- Argumentos:
  - Identificador de transação JICS (string, obrigatório): identificador da transação JICS a ser iniciada (8 caracteres no máximo)
  - obrigatório: dados de entrada adicionais para passar para a transação, como um Map<String,Object>. O conteúdo desse mapa será usado para alimentar a COMMAREA que será consumida pela transação do JICS. O mapa pode ficar vazio se nenhum dado for necessário para executar a transação.
  - opcional: entradas de cabeçalhos HTTP, para personalizar o ambiente de runtime da transação em questão. As seguintes chaves de cabeçalho estão sendo suportadas:
    - `jics-channel`: o nome do JICS CHANNEL a ser usado pelo programa que será lançado com o lançamento dessa transação.
    - `jics-container`: o nome do JICS CONTAINER a ser usado para o lançamento dessa transação JICS.
    - `jics-startcode`: o STARTCODE (string, até 2 caracteres) a ser usado no início da transação JICS. Consulte STARTCODE para valores possíveis (navegue pela página).
    - `jicxa-xid`: o XID (estrutura XID do identificador de transação X/Open) de uma “transação global” (XA), iniciada pelo chamador, da qual participará o lançamento atual da transação JICS.

- Retorna: uma serialização `com.netfective.bluage.gapwalk.rt.shared.web.TransactionResultBean` JSON, representando o resultado do lançamento da transação JICS.

Para obter mais informações sobre os detalhes da estrutura, consulte Estrutura de resultados do lançamento da transação.

Iniciar uma transação JICS (alternativa)

- métodos compatíveis: GET, POST
- path: `/jicstransaction/{jtrans:.+}`
- Argumentos:

  Identificador de transação JICS (string, obrigatório)

  identificador da transação JICS a ser iniciada (8 caracteres no máximo)

  obrigatório: dados de entrada adicionais para passar para a transação, como um Map<String,Object>

  O conteúdo desse mapa será usado para alimentar a COMMAREA que será consumida pela transação do JICS. O mapa pode ficar vazio se nenhum dado for necessário para executar a transação.

  opcional: entradas de cabeçalhos HTTP, para personalizar o ambiente de runtime da transação em questão.

  As seguintes chaves de cabeçalho estão sendo suportadas:

  - `jics-channel`: o nome do JICS CHANNEL a ser usado pelo programa que será lançado com o lançamento dessa transação.
  - `jics-container`: o nome do JICS CONTAINER a ser usado para o lançamento dessa transação JICS.
  - `jics-startcode`: o STARTCODE (string, até 2 caracteres) a ser usado no início da transação JICS. Para valores possíveis, consulte STARTCODE (navegue pela página).
  - `jicxa-xid`: o XID (estrutura XID do identificador de transação X/Open) de uma “transação global” (XA), iniciada pelo chamador, da qual participará o lançamento atual da transação JICS.
- retorna: uma serialização `com.netfective.bluage.gapwalk.rt.shared.web.RecordHolderBean` JSON,

representando o resultado do lançamento da transação JICS. Os detalhes da estrutura podem ser encontrados emEstrutura de resultados do registro de lançamento da transação.

Endpoints relacionados ao Job Queues

As Filas de Tarefas são o suporte do AWS Blu Age para o mecanismo de envio de trabalhos do AS400. As filas de trabalhos são usadas no AS400 para executar trabalhos em grupos de threads específicos. Uma fila de trabalhos é definida por um nome e um número máximo de threads que corresponde ao número máximo de programas que podem ser executados simultaneamente nessa fila. Se mais trabalhos forem enviados na fila do que o número máximo de threads, os trabalhos aguardarão até que um tópico esteja disponível.

Para obter uma lista completa do status de um trabalho em uma fila, consulte. Possível status de trabalho em uma fila

As operações nas filas de trabalhos são realizadas por meio dos seguintes endpoints dedicados. Você pode invocar essas operações a partir do URL do aplicativo Gapwalk com o seguinte URL raiz:. `http://server:port/gapwalk-application/jobqueue`

Tópicos

- Listar filas disponíveis
- Iniciar ou reiniciar uma fila de trabalhos
- Envie um trabalho para lançamento
- Listar todos os trabalhos enviados
- Libere todos os trabalhos que estão “em espera”
- Libere todos os trabalhos que estão “em espera” para um determinado nome de trabalho
- Libere um determinado trabalho para um número de trabalho
- Envie um trabalho em um cronograma repetido
- Listar todos os trabalhos repetidos enviados
- Cancelar o agendamento de um trabalho repetido

Listar filas disponíveis

- Método compatível: GET
- Caminho: `list-queues`

- Retorna a lista de filas disponíveis junto com seu status, como uma lista JSON de valores-chave.

Resposta de exemplo:

```
{"Default":"STAND_BY","queue1":"STARTED","queue2":"STARTED"}
```

Os possíveis status de uma fila de trabalhos são:

EM ESPERA

a fila de trabalhos está esperando para ser iniciada.

STARTED

a fila de trabalhos está ativa e funcionando.

UNKNOWN

o status da fila de trabalhos não pode ser determinado.

Iniciar ou reiniciar uma fila de trabalhos

- Método compatível: POST
- Caminho: `/restart/{name}`
- Argumento: o nome da fila a ser iniciada/reiniciada, como uma string — obrigatório.
- O endpoint não retorna nada, mas depende do status http para indicar o resultado da operação de início/reinício:

  HTTP 200

  a operação de início/reinício correu bem: a fila de trabalhos fornecida agora está INICIADA.

  HTTP 404

  a fila de trabalhos não existe.

  HTTP 503

  ocorreu uma exceção durante a tentativa de iniciar/reiniciar (os logs do servidor devem ser inspecionados para descobrir o que deu errado).

Envie um trabalho para lançamento

- Método compatível: POST
- Caminho: `/submit`
- Argumento: obrigatório como corpo da solicitação, uma serialização JSON de um objeto `com.netfective.bluage.gapwalk.rt.jobqueue.SubmitJobMessage`. Para ter mais informações, consulte Envie uma tarefa e agende a entrada da tarefa.
- Retornos: um JSON contendo o original `SubmitJobMessage` e um log indicando se o trabalho foi enviado ou não.

Listar todos os trabalhos enviados

- Método compatível: GET
- Caminho: `/list-jobs?status={status}&size={size}&page={page}&sort={sort}`
- Argumentos:
  - página: Número da página a ser recuperada (padrão = 1)
  - tamanho: Tamanho da página (padrão = 50, máximo = 300)
  - ordenar: A ordem dos trabalhos. (padrão = “ExecutionID”). “ExecutionID” é atualmente o único valor suportado
  - status: (opcional) Se presente, ele filtrará o status.
- Retorna: uma lista de todos os trabalhos agendados, como uma string JSON. Para obter um exemplo de resposta, consulte Lista de respostas de trabalhos agendados.

Libere todos os trabalhos que estão “em espera”

- Método compatível: POST
- Caminho: `/release-all`
- Retorna: uma mensagem indicando o resultado da operação de tentativa de liberação. Dois casos possíveis aqui:
  - HTTP 200 e uma mensagem “Todos os trabalhos foram lançados com sucesso!” se todos os trabalhos foram liberados com sucesso.
  - HTTP 503 e uma mensagem “Trabalhos não lançados. Ocorreu um erro desconhecido. Consulte o log para obter mais detalhes” se algo deu errado com a tentativa de lançamento.

Libere todos os trabalhos que estão “em espera” para um determinado nome de trabalho

<job name, job number>Para um determinado nome de trabalho, vários trabalhos podem ser enviados, com números de trabalho diferentes (a unicidade de um trabalho executado é garantida por um casal). O endpoint tentará liberar todos os envios de trabalhos com o nome de trabalho fornecido, que estão “em espera”.

- Método compatível: POST
- Caminho: `/release/{name}`
- Argumentos: o nome do trabalho a ser procurado, como uma string. Obrigatório.
- Retorna: uma mensagem indicando o resultado da operação de tentativa de liberação. Dois casos possíveis aqui:
  - HTTP 200 e uma mensagem “Jobs in group <name>(<number of released jobs>) lançados com sucesso!” os trabalhos foram liberados com sucesso.
  - HTTP 503 e uma mensagem “Trabalhos no grupo <name>não lançados. Ocorreu um erro desconhecido. Consulte o log para obter mais detalhes” se algo deu errado com a tentativa de lançamento.

Libere um determinado trabalho para um número de trabalho

<job name, job number>O endpoint tentará liberar o envio exclusivo do trabalho, que está “suspenso”, para o casal em questão.

- Método compatível: POST
- Caminho: `/release/{name}/{number}`
- Argumentos:

  name

  o nome do trabalho a ser procurado, como uma string. Obrigatório.

  número

  o número do trabalho a ser procurado, como um número inteiro. Obrigatório.

  retorna

  uma mensagem indicando o resultado da operação de tentativa de liberação. Dois casos possíveis aqui:

- HTTP 200 e uma mensagem “Job <name/number> released with success!” se o trabalho foi lançado com sucesso.
- HTTP 503 e uma mensagem “Job <name/number> not released. Ocorreu um erro desconhecido. Consulte o log para obter mais detalhes” se algo deu errado com a tentativa de lançamento.

Envie um trabalho em um cronograma repetido

Agende um trabalho que será executado com um cronograma repetido.

- Método compatível: POST
- Caminho: `/schedule`
- Argumento: o corpo da solicitação deve conter uma serialização JSON de um `com.netfective.bluage.gapwalk.rt.jobqueue.SubmitJobMessage` objeto.

Listar todos os trabalhos repetidos enviados

- Método compatível: GET
- Caminho: `/schedule/list?status={status}&size={size}&page={page}&sort={sort}`
- Argumentos:
  1. página: Número da página a ser recuperada (padrão = 1)
  2. tamanho: Tamanho da página (padrão = 50, máximo = 300)
  3. ordenar: A ordem dos trabalhos. (padrão = “id”). “id” é o único valor suportado no momento.
  4. status: (opcional) Se presente, ele filtrará o status. Os valores possíveis são os mencionados na seção 1.
  5. status: (opcional) Se presente, ele filtrará o status. Os valores possíveis são os mencionados na seção 1.
  6. Retorna: uma lista de todos os trabalhos agendados, como uma string JSON.

Cancelar o agendamento de um trabalho repetido

Remove um trabalho que foi criado em um cronograma repetido. O status do agendamento do trabalho está definido como INATIVO.

- Método compatível: GET
- Caminho: `/schedule/remove/{schedule_id}`
- Argumento:`schedule_id`, o identificador do trabalho agendado a ser removido.

Endpoints REST do Blusam Application Console

O Blusam Application Console é uma API projetada para simplificar o gerenciamento de conjuntos de dados VSAM modernizados. Os endpoints da aplicação web Blusam usam o caminho raiz `/bac`.

Tópicos

Conjuntos de dados relacionados a endpoints

Use os seguintes endpoints para criar ou gerenciar um conjunto de dados específico.

Tópicos

Criar um conjunto de dados

Você pode usar esse endpoint para criar uma definição de conjunto de dados.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/createDataSet`
- Argumentos:

  name

  (obrigatório, string): o nome do conjunto de dados.

  tipo

  (obrigatório, string): o tipo de conjunto de dados. Os valores possíveis são: `ESDS`, `KSDS` `RRDS`.

  recordSize

  (opcional, string): tamanho máximo de cada registro do conjunto de dados.

  fixedLength

  (opcional, booleano): indica se o tamanho dos registros é fixo.

  compression

  (opcional, booleano): indica se o conjunto de dados está compactado.

  cacheEnable

  (opcional, booleano): indica se o armazenamento em cache está ativado para o conjunto de dados.

  alternativeKeys

  (opcional, lista de chaves):

  - deslocamento (obrigatório, número)
  - comprimento (obrigatório, número)
  - nome (obrigatório, número)
- Retorna um arquivo json representando o conjunto de dados recém-criado.

Exemplo de solicitação:

```
POST /api/services/rest/bluesamservice/createDataSet
{
  "name": "DATASET",
  "checked": false,
  "records": [],
  "primaryKey": {
    "name": "PK"
  },
  "alternativeKeys": [
    {
      "offset": 10,
      "length": 10,
      "name": "ALTK_0"
    }
  ],
  "type": "ESDS",
  "recordSize": 10,
  "compression": true,
  "cacheEnable": true
}
```

Resposta de exemplo:

```
{
    "dataSet": {
      "name": "DATASET",
      "checked": false,
      "nbRecords": 0,
      "keyLength": -1,
      "recordSize": 10,
      "compression": false,
      "fixLength": true,
      "type": "ESDS",
      "cacheEnable": false,
      "cacheWarmup": false,
      "cacheEviction": "100ms",
      "creationDate": 1686744961234,
      "modificationDate": 1686744961234,
      "records": [],
      "primaryKey": {
        "name": "PK",
        "offset": null,
        "length": null,
```

```
        "columns": null,
        "unique": true
      },
      "alternativeKeys": [
        {
          "offset": 10,
          "length": 10,
          "name": "ALTK_0"
        }
      ],
      "readLimit": 0,
      "readEncoding": null,
      "initCharacter": null,
      "defaultCharacter": null,
      "blankCharacter": null,
      "strictZoned": null,
      "decimalSeparator": null,
      "currencySign": null,
      "pictureCurrencySign": null
    },
    "message": null,
    "result": true
  }
```

Carregar um arquivo

Você pode usar esse endpoint para fazer upload de arquivos para o servidor. O arquivo é armazenado em uma pasta temporária que corresponde a cada usuário específico. Use esse endpoint sempre que precisar fazer upload de um arquivo.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/upload`
- Argumentos:

  file

  (obrigatório, multipartes/dados de formulário): o arquivo a ser carregado.
- Retorna um booleano que reflete o status do carregamento

Carregar um conjunto de dados (POST)

Depois de usar `createDataSet` para criar a definição do conjunto de dados, você pode carregar registros associados ao arquivo carregado em um conjunto de dados específico.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/loadDataSet`
- Argumentos:

  name

  (obrigatório, string): o nome do conjunto de dados.
- Retorna o status da solicitação e do conjunto de dados carregado.

Carregar um conjunto de dados (GET)

- Métodos compatíveis: GET
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/loadDataSet`
- Argumentos:

  name

  (obrigatório, string): o nome do conjunto de dados.

  arquivo de conjunto de dados

  (obrigatório, string): o nome do arquivo do conjunto de dados.
- Retorna o status da solicitação e do conjunto de dados carregado.

Carregar um conjunto de dados de um bucket do Amazon S3

Carrega um conjunto de dados usando um arquivo listcat de um bucket do Amazon S3.

- Métodos compatíveis: GET
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/loadDataSetFromS3`
- Argumentos:

listcatFileS3Location

(obrigatório, string): a localização do arquivo listcat no Amazon S3.

datasetFileS3Location

(obrigatório, string): a localização do Amazon S3 do arquivo do conjunto de dados.

região

(obrigatório, string): o Amazon S3 Região da AWS onde os arquivos são armazenados.

- Retorna o conjunto de dados recém-criado

Exemplo de solicitação:

```
/BAC/api/services/rest/bluesamservice/loadDataSetFromS3?region=us-
east-1&listcatFileS3Location=s3://bucket-name/listcat.json&datasetFileS3Location=s3://
bucket-name/dataset.DAT
```

Exportar um conjunto de dados para um bucket do Amazon S3

Exporta um conjunto de dados para o bucket do Amazon S3 especificado.

- Métodos compatíveis: GET
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/exportDataSetToS3`
- Argumentos:

s3Location

(obrigatório, string): o local do Amazon S3 para o qual exportar o conjunto de dados.

datasetName

(obrigatório, string): o nome do conjunto de dados a ser exportado.

região

(obrigatório, string): o Região da AWS do bucket do Amazon S3.

- Retorna o conjunto de dados exportado

Exemplo de solicitação:

```
/BAC/api/services/rest/bluesamservice/exportDataSetToS3?region=eu-
west-1&s3Location=s3://bucket-name/dump&datasetName=dataset
```

Limpar um conjunto de dados

Limpa todos os registros de um conjunto de dados.

- Métodos suportados: POST, GET
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/clearDataSet`
- Argumentos:

  name

  (obrigatório, string): o nome do conjunto de dados a ser limpo.
- Retorna o status da solicitação.

Excluir um conjunto de dados

Exclui a definição e os registros do conjunto de dados.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/deleteDataSet`
- Argumentos:

  name

  (obrigatório, string): o nome do conjunto de dados a ser excluído.
- Retorna o status da solicitação e do conjunto de dados excluído.

Contar registros do conjunto de dados

Esse endpoint retorna o número de registros associados a um conjunto de dados.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/bluesamservice/countRecords`

- Argumentos:

  name

  (obrigatório, string): o nome do conjunto de dados.
- Retornos: o número de registros

Conjuntos de dados em massa relacionados a endpoints

Use os seguintes endpoints para criar ou gerenciar vários conjuntos de dados ao mesmo tempo.

Tópicos

Exportar conjuntos de dados (GET)

- Métodos compatíveis: GET
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/bluesamservice/exportDataSet`
- Argumentos:

  datasetName

  (obrigatório, string): o nome do conjunto de dados a ser exportado.

datasetOutputFile

(obrigatório, string): o caminho da pasta em que você deseja armazenar o conjunto de dados exportado no servidor.

vermelho

(obrigatório, booleano): se você deseja que a palavra descritora de registro (RDW) faça parte dos registros exportados. Se o conjunto de dados tiver registros de tamanho fixo, o valor desse parâmetro será ignorado.

- Retorna o status da solicitação e o caminho para o arquivo que contém o conjunto de dados exportado (se houver). Se o conjunto de dados for nulo na resposta, isso significa que o sistema não conseguiu localizar um conjunto de dados com o nome fornecido.

Exportar conjuntos de dados (POST)

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/bluesamservice/exportDataSet`
- Argumentos:

  Parâmetros de despejo

  (obrigatório, BACReadParameters): parâmetros de leitura do Bluesam.
- Retorna o status do conjunto de dados exportado.

Criar vários conjuntos de dados

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/createAllDataSets`
- Argumentos:
  - Lista de conjuntos de dados

    name

    (obrigatório, string): o nome do conjunto de dados.

tipo

(obrigatório, string): o tipo de conjunto de dados. Os valores possíveis são: `ESDS`, `KSDS` `RRDS`.

recordSize

(opcional, string): tamanho máximo de cada registro do conjunto de dados.

fixedLength

(opcional, booleano): indica se o tamanho dos registros é fixo.

compression

(opcional, booleano): indica se o conjunto de dados está compactado.

cacheEnable

(opcional, booleano): indica se o armazenamento em cache está ativado para o conjunto de dados.

- Retornos: o status da solicitação e o conjunto de dados recém-criado.

Listar todos os conjuntos de dados

- Métodos compatíveis: GET
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/bluesamservice/listDataSet`
- Argumentos: nenhum
- Retornos: o status da solicitação e a lista dos conjuntos de dados.

Listar diretamente todos os conjuntos de dados

- Métodos compatíveis: GET
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/bluesamservice/directListDataSet`
- Argumentos: nenhum
- Retornos: o status da solicitação e a lista dos conjuntos de dados.

Listar diretamente todos os conjuntos de dados por página

- Métodos compatíveis: GET
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/bluesamservice/directListDataSetByPage`
- Argumentos:

  datasetName

  (obrigatório, string): o nome do conjunto de dados.

  Número da página

  (obrigatório, int): o número da página.

  pageSize

  (obrigatório, int): o tamanho da página.
- Retornos: o status da solicitação e a lista dos conjuntos de dados.

Conjunto de dados de streaming

- Métodos compatíveis: GET
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/streamDataset`
- Argumentos:

  datasetName

  (obrigatório, string): o nome do conjunto de dados.
- Retornos: um fluxo dos conjuntos de dados solicitados.

Excluir todos os conjuntos de dados

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/removeAll`
- Argumentos: nenhum
- Retorna: um booleano representando o status da solicitação.

Obtenha definições do conjunto de dados do arquivo listcat

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/getDataSetsDefinitionFromListcat`
- Argumentos:

  paramFilePath

  (obrigatório, string): o caminho para o arquivo listcat.
- Retornos: uma lista de conjuntos de dados

Obtenha as definições do conjunto de dados do arquivo cat da lista carregado

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/getDataSetsDefinitionFromUploadedListcat`
- Argumentos: nenhum
- Retornos: uma lista de conjuntos de dados

Obtenha um conjunto de dados

- Métodos compatíveis: GET
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/bluesamservice/getDataSet`
- Argumentos:

  name

  (obrigatório, string): o nome do conjunto de dados.
- Retorna o conjunto de dados solicitado.

Carregar listcat do arquivo JSON

- Métodos compatíveis: GET

- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/loadListcatFromJsonFile`
- Argumentos:

  filePath

  (obrigatório, string): o caminho para o arquivo listcat.
- Retornos: uma lista de conjuntos de dados

Registros

Use os seguintes endpoints para criar ou gerenciar registros em um conjunto de dados.

Tópicos

Criar um registro

Você pode usar esse endpoint para criar um novo registro.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/crud/createRecord`
- Argumentos:

  conjunto de dados

  (obrigatório, DataSet): o objeto do conjunto de dados

  máscara

  (obrigatório, máscara): o objeto da máscara.

- Retorna o status da solicitação e o registro criado.

Leia um conjunto de dados

Você pode usar esse endpoint para ler um conjunto de dados.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/crud/readDataSet`
- Argumentos:

  conjunto de dados

  (obrigatório, DataSet): o objeto do conjunto de dados.
- Retorna o status da solicitação e o conjunto de dados com os registros.

Excluir um registro

Você pode usar esse endpoint para excluir um registro de um conjunto de dados.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/crud/deleteRecord`
- Argumentos:

  conjunto de dados

  (obrigatório, DataSet): o objeto do conjunto de dados

  registro

  (obrigatório, Registro): o registro a ser excluído
- Retorna o status da exclusão.

Atualizar um registro

Você pode usar esse endpoint para atualizar um registro associado a um conjunto de dados.

- Métodos compatíveis: POST

- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/crud/updateRecord`
- Argumentos:

  conjunto de dados

  (obrigatório, DataSet): o objeto do conjunto de dados

  registro

  (obrigatório, Registro): o registro a ser atualizado
- Retorna o status da solicitação e o conjunto de dados com os registros.

Salvar um registro

Você pode usar esse endpoint para salvar um registro em um conjunto de dados usando uma máscara.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/crud/saveRecord`
- Argumentos:

  conjunto de dados

  (obrigatório, DataSet): o objeto do conjunto de dados

  registro

  (obrigatório, Registro): o registro a ser salvo
- Retorna o status da solicitação e o conjunto de dados com os registros.

Validar um registro

Use esse endpoint para validar um registro.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/crud/validateRecord`
- Argumentos:

conjunto de dados

(obrigatório, DataSet): o objeto do conjunto de dados

- Retorna o status da solicitação e o conjunto de dados com os registros.

Obtenha uma árvore de registros

Use esse endpoint para obter a árvore hierárquica de um registro.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/crud/getRecordTree`
- Argumentos:

  conjunto de dados

  (obrigatório, DataSet): o objeto do conjunto de dados

  registro

  (obrigatório, Registro): o registro a ser buscado
- Retorna o status da solicitação e a árvore hierárquica do registro solicitado.

Máscaras

Use os seguintes endpoints para carregar ou aplicar máscaras a um conjunto de dados.

Tópicos

- Carregue máscaras
- Aplicar máscara
- Aplicar filtro de máscara

Carregue máscaras

Você pode usar esse endpoint para recuperar todas as máscaras associadas a um conjunto de dados específico.

- Métodos compatíveis: POST

- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/crud/loadMasks`
- Argumentos:

  conjunto de dados

  (obrigatório, DataSet): o objeto do conjunto de dados
- Retorna o status da solicitação e a lista das máscaras.

Aplicar máscara

Você pode usar esse endpoint para aplicar uma máscara a um conjunto de dados específico.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/crud/applyMask`
- Argumentos:

  conjunto de dados

  (obrigatório, DataSet): o objeto do conjunto de dados

  máscara

  (obrigatório, Máscara): o objeto do conjunto de dados
- Retorna o status da solicitação e do conjunto de dados com a máscara aplicada.

Aplicar filtro de máscara

Você pode usar esse endpoint para aplicar uma máscara e um filtro a um conjunto de dados específico.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/crud/applyMaskFilter`
- Argumentos:

  conjunto de dados

  (obrigatório, DataSet): o objeto do conjunto de dados

máscara

(obrigatório, Máscara): o objeto do conjunto de dados

- Retorna o status da solicitação e do conjunto de dados com a máscara e o filtro aplicados.

Outros

Use os seguintes endpoints para gerenciar o cache de um conjunto de dados ou verificar as características do conjunto de dados:

Tópicos

Verificar o cache de aquecimento

Verifica se o cache de aquecimento está ativado para um conjunto de dados específico.

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/warmupCache`
- Argumentos:

  name

  (obrigatório, string): o nome do conjunto de dados.
- Retorna: verdadeiro se o cache de aquecimento estiver ativado e falso caso contrário.

Verifique o cache ativado

Verifica se o cache está habilitado para um conjunto de dados específico.

- Métodos compatíveis: GET
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/bluesamservice/isEnableCache`
- Argumentos: nenhum
- Retorna verdadeiro se o armazenamento em cache estiver ativado.

Habilitar cache

- Métodos compatíveis: GET
- Requer autenticação e as funções ROLE_ADMIN e ROLE_SUPER_ADMIN.
- Caminho: `/api/services/rest/bluesamservice/enableDisableCache/{enable}`
- Argumentos:

  enable

  (obrigatório, booleano): se definido como verdadeiro, ele habilitará o armazenamento em cache.
- Retornos: nenhum

Verifique o cache de RAM alocado

Você pode usar esse endpoint para recuperar a memória cache RAM alocada.

- Métodos compatíveis: GET
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/bluesamservice/allocatedRamCache`
- Argumentos: nenhum
- Retorna: o tamanho da memória como uma string

Verificar a persistência

- Métodos compatíveis: GET
- Requer autenticação e a função ROLE_USER.
- Caminho: `/api/services/rest/bluesamservice/persistence`

- Argumentos: nenhum
- Retorna: a persistência usada como uma string

Verifique os tipos de conjuntos de dados compatíveis

- Métodos compatíveis: GET
- Caminho: `/api/services/rest/bluesamservice/getDataSetTypes`
- Requer autenticação e a função ROLE_USER.
- Argumentos: nenhum
- Retorna: a lista de tipos de conjuntos de dados compatíveis como uma lista de cadeias de caracteres.

Verificar a integridade do servidor

- Métodos compatíveis: GET
- Caminho: `/api/services/rest/bluesamserver/serverIsUp`
- Argumentos: nenhum
- Devoluções: Nenhuma. O código de status de resposta HTTP 200 indica que o servidor está ativo e funcionando.

Usuários

Use os seguintes endpoints para gerenciar as interações do usuário.

Tópicos

Login

- Métodos compatíveis: POST
- Caminho: `/api/services/security/servicelogin/login`
- Argumentos:

  username

  (obrigatório, string)

  password

  (obrigatório, string)
- Retorna o nome de usuário e as funções do usuário conectado

Exemplo de resposta

```
{"login":"some-user","roles":[{"id":0,"roleName":"ROLE_ADMIN"}]}
```

Verificar conta de usuário

- Métodos compatíveis: POST
- Caminho: `/api/services/security/servicelogin/hasAccount`
- Argumentos: nenhum
- Retorna: verdadeiro se o usuário já estiver logado

Fazer logon

- Métodos compatíveis: POST
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/security/servicelogin/recorduser`
- Argumentos: nenhum
- Retorna: verdadeiro se o usuário já estiver logado

Obter informações do usuário

- Métodos compatíveis: GET

- Caminho: `/api/services/security/servicelogin/userInfo`
- Argumentos: nenhum
- Retorna: o nome de usuário e a função do usuário atualmente conectado

Listar todos os usuários

- Métodos compatíveis: GET
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/security/servicelogin/listusers`
- Argumentos: nenhum
- Retorna: a lista de todos os usuários

Delete user (Excluir usuário)

- Métodos compatíveis: POST
- Caminho: `/api/services/security/servicelogin/deleteuser`
- Argumentos:

  login

  (obrigatório, string)
- Retorna: verdadeiro se o usuário tiver sido removido com sucesso

logout

- Métodos compatíveis: POST
- Caminho: `/api/services/security/servicelogout/logout`
- Argumentos: nenhum
- Retorna: verdadeiro se o usuário tiver sido desconectado com sucesso.

Console de aplicações JICS

O componente do JICS é o suporte do AWS Blu Age para a modernização dos recursos antigos do CICS. O aplicativo web do JICS Application Console é dedicado a administrar os recursos do JICS. Os seguintes endpoints permitem realizar as tarefas de administração sem precisar interagir com a

interface de usuário do JAC. Sempre que um endpoint exigir autenticação, a solicitação deverá incluir detalhes de autenticação (nome de usuário/senha normalmente, conforme exigido pela Autenticação Básica). Os endpoints do aplicativo web do JICS Application Console usam o caminho raiz. `/jac/`

Tópicos

Gerenciamento de recursos do JICS

Todos os endpoints a seguir estão relacionados ao gerenciamento de recursos do JICS, permitindo que os administradores do JICS lidem com os recursos diariamente.

Tópicos

- Crie um ID de transação.
- Como criar um programa
- Crie um ARQUIVO
- Crie um TDQUEUE
- Cria um modelo.
- Crie elementos
- Atualizar um link
- Atualiza um grupo.
- Considerações sobre a atualização de RECURSOS COMUNS
- Atualizar uma TRANSAÇÃO
- Atualizar um PROGRAMA
- Atualizar um ARQUIVO
- Atualizar um TDQUEUE
- Atualiza um modelo.
- Atualizar elementos
- Elementos upsert
- Recuperar elementos
- Operação JICS CRUD

Listar LISTAS E GRUPOS DO JICS

A LISTA e os GRUPOS são os principais recursos de contêiner proprietários dentro do componente JICS. Todos os recursos do JICS devem pertencer a um GRUPO. Os grupos podem pertencer às LISTAS, mas isso não é obrigatório. As LISTAS podem até não existir em um determinado ambiente JICS, mas, na maioria das vezes, as LISTAS existem para fornecer uma camada extra de organização dos recursos. Para obter mais informações sobre a organização de recursos do CICS, consulte os recursos do CICS.

- Método compatível: GET
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/listJicsListsAndGroups`
- Argumentos: nenhum

- Retorna: uma lista de JicsContainer objetos serializados, tanto LISTS quanto GROUPS, como JSON.

Resposta de exemplo:

```
[
    {
      "name": "Resources",
      "children": [
        {
          "jacType": "JACList",
          "name": "MURACHS",
          "isActive": true,
          "children": [
            {
              "jacType": "JACGroup",
              "name": "MURACHS",
              "isActive": true,
              "children": []
            }
          ]
        },
        {
          "jacType": "JACGroup",
          "name": "TEST",
          "isActive": true,
          "children": []
        }
      ],
      "isExpanded": true
    }
  ]
```

Recupere recursos do JICS

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/retrieveJicsResources`

- Argumentos: uma carga JSON que representa os recursos do JICS que você deseja recuperar. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.request.RetrieveOperationRequest`.
- Retornos: uma lista de JicsResource objetos serializados. Os objetos são retornados em nenhuma ordem específica e são de tipos diferentes, como PROGRAM, TRANSACTION, FILE e assim por diante.

Listar GRUPOS JICS

- Método compatível: GET
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/listJicsGroups`
- Retorna: uma lista de JicsContainer objetos serializados (GRUPOS) como JSON. Os GRUPOS estão sendo retornados sem suas próprias informações de LISTA.

Resposta de exemplo:

```
[
    {
      "jacType": "JACGroup",
      "name": "MURACHS",
      "isActive": true,
      "children": []
    },
    {
      "jacType": "JACGroup",
      "name": "TEST",
      "isActive": true,
      "children": []
    }
  ]
```

Listar GRUPOS JICS para uma determinada LISTA

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER

- Caminho: `/api/services/rest/jicsservice/listGroupsForList`
- Argumentos: uma carga JSON, representando a LISTA JICS cujos GRUPOS estamos procurando. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACList`.

  Exemplo de solicitação:

```
{
    "jacType":"JACList",
    "name":"MURACHS",
    "isActive":true
  }
```

- Retorna: uma lista de JicsContainer objetos serializados (GROUPS) como JSON, anexados à LIST fornecida. Os GRUPOS estão sendo retornados sem suas próprias informações de LISTA.

  Resposta de exemplo:

```
[
    {
      "jacType": "JACGroup",
      "name": "MURACHS",
      "isActive": true,
      "children": []
    }
  ]
```

LISTAR recursos JICS para um determinado GRUPO

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/listResourcesForGroup`
- Argumentos: uma carga JSON, representando o GRUPO JICS cujos recursos estamos procurando. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACGroup`. Você não precisa especificar todos os campos para o GRUPO, mas o nome é obrigatório.

  Exemplo de solicitação:

```
{
    "jacType":"JACGroup",
    "name":"MURACHS",
    "isActive":true
  }
```

- Retorna: uma lista de JicsResource objetos serializados, de propriedade do DETERMINADO GRUPO. Os objetos estão sendo retornados em nenhuma ordem específica e são de tipos diferentes (PROGRAMA, TRANSAÇÃO, ARQUIVO, etc.).

LISTAR recursos JICS para um determinado GRUPO (alternativa usando um nome)

- Método compatível: POST
- Requer autenticação
- Caminho: `/api/services/rest/jicsservice/listResourcesForGroupName`
- Argumentos: o nome do GRUPO que possui os recursos que estamos procurando.
- Retorna: uma lista de JicsResource objetos serializados, de propriedade do DETERMINADO GRUPO. Os objetos estão sendo retornados em nenhuma ordem específica e são de tipos diferentes (PROGRAMA, TRANSAÇÃO, ARQUIVO, etc.)

Editando os GRUPOS próprios de várias LISTAS

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/editGroupsList`
- Argumentos: uma representação JSON de uma coleção de LISTAS com grupos infantis;

  Exemplo de solicitação:

```
[
  {
        "jacType": "JACList",
        "name": "MURACHS",
        "isActive": true,
        "children": [
```

```
          {
            "jacType": "JACGroup",
            "name": "MURACHS",
            "isActive": true,
            "children": []
          },
          {
            "jacType": "JACGroup",
            "name": "TEST",
            "isActive": true,
            "children": []
          }
        ]
  }
]
```

Antes dessa edição, somente o grupo chamado “MURACHS” pertencia à LISTA chamada “MURACHS”. Com essa edição, “adicionamos” o grupo chamado “TEST” à LISTA chamada “MURACHS”.

- Retorna um valor Booleano. Se o valor for 'true', as modificações do LISTS persistiram adequadamente no armazenamento JICS subjacente.

Excluir um link

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/deleteList`
- Argumentos: uma carga JSON, representando a LISTA JICS a ser excluída. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACList`.
- Retorna um valor Booleano. Se o valor for 'true', a exclusão de LIST foi operada adequadamente no armazenamento JICS subjacente.

Excluir um grupo

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER

- Caminho: `/api/services/rest/jicsservice/deleteGroup`
- Argumentos: uma carga JSON, representando o GRUPO JICS a ser excluído. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACGroup`.
- Retorna um valor Booleano. Se o valor for 'true', a exclusão do GRUPO foi operada adequadamente no armazenamento JICS subjacente.

Excluir uma TRANSAÇÃO

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/deleteTransaction`
- Argumentos: uma carga JSON, representando a transação JICS a ser excluída. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACTransaction`.
- Retorna um valor Booleano. Se o valor for 'true', a exclusão da TRANSAÇÃO foi operada adequadamente no armazenamento JICS subjacente.

Como excluir um programa

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/deleteProgram`
- Argumentos: uma carga JSON, representando o programa JICS a ser excluído. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACProgram`.
- Retorna um valor Booleano. Se o valor for 'true', a exclusão do PROGRAMA foi operada adequadamente no armazenamento JICS subjacente.

Excluir um arquivo

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER

- Caminho: `/api/services/rest/jicsservice/deleteFile`
- Argumentos: uma carga JSON, representando o arquivo JICS a ser excluído. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACFile`.
- Retorna um valor Booleano. Se o valor for 'true', a exclusão do ARQUIVO foi operada adequadamente no armazenamento JICS subjacente.

Excluir um TDQUEUE

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/deleteTDQueue`
- Argumentos: uma carga JSON, representando o JICS TDQUEUE a ser excluído. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACTDQueue`.
- Retorna um valor Booleano. Se o valor for 'true', a exclusão do TDQUEUE foi operada adequadamente no armazenamento JICS subjacente.

Exclui um TSMODEL

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/deleteTSModel`
- Argumentos: uma carga JSON, representando o JICS TSMODEL a ser excluído. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACTSModel``.
- Retorna um valor Booleano. Se o valor for 'true', a exclusão do TSMODEL foi operada adequadamente no armazenamento JICS subjacente.

Excluir elementos

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/deleteElements`

- Argumentos: uma carga JSON que representa os elementos JICS a serem excluídos.
- Retorna um valor booleano que `true` indica que a exclusão foi operada com sucesso no armazenamento JICS subjacente.

Criar uma LISTA

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/createList`
- Argumentos: uma carga JSON, representando a LISTA JICS a ser criada. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACList`.
- Retorna um valor Booleano. Se o valor for 'true', a LIST foi criada corretamente no armazenamento JICS subjacente.

> **Note**
>
> A LISTA sempre será criada vazia. Anexar GRUPOS à LISTA exigirá outra operação.

Criar um GRUPO

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/createGroup`
- Argumentos: uma carga JSON, representando o GRUPO JICS a ser criado. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACGroup`.
- Retorna um valor Booleano. Se o valor for 'true', o GRUPO foi criado corretamente no armazenamento JICS subjacente.

> **Note**
>
> O GRUPO sempre será criado vazio. Anexar RECURSOS ao GRUPO exigirá operações adicionais (a criação de recursos os anexará automaticamente a um determinado GRUPO).

Considerações comuns sobre a criação de RECURSOS

Todos os endpoints a seguir estão relacionados à criação de JICS RESOURCES e compartilham algumas restrições comuns: na carga útil da solicitação a ser enviada ao endpoint, o campo `groupName` precisa ser valorizado.

Restrição de propriedade do GRUPO:

Nenhum recurso pode ser criado sem ser anexado a um grupo existente, e o endpoint usa o groupName para recuperar o grupo ao qual esse recurso será anexado. O `groupName` deve ser igual ao nome de um GRUPO existente. Uma mensagem de erro com HTTP STATUS 400 será enviada se não `groupName` estiver apontando para um grupo existente no armazenamento subjacente do JICS.

Restrição de unicidade dentro de um GRUPO:

Um recurso especificado com um nome especificado deve ser exclusivo dentro de um grupo especificado. A verificação da unicidade será realizada por cada endpoint de criação de recursos. Se a carga útil fornecida não respeitar a restrição de unicidade, o endpoint enviará uma resposta com HTTP STATUS 400 (BAD REQUEST) — veja o exemplo de resposta abaixo.

Exemplo de carga útil: tentamos criar a transação 'ARIT' no grupo 'TEST', mas uma transação com esse nome já existe nesse GRUPO.

```
{
    "jacType":"JACTransaction",
    "name":"ARIT",
    "groupName":"TEST",
    "isActive":true
  }
```

Recebemos a seguinte resposta de erro:

```
{
```

```
    "timestamp": 1686759054510,
    "status": 400,
    "error": "Bad Request",
    "path": "/jac/api/services/rest/jicsservice/createTransaction"
  }
```

A inspeção dos logs dos servidores confirmará a origem do problema:

```
2023-06-14 18:10:54 default         TRACE - o.s.w.m.HandlerMethod
        - Arguments: [java.lang.IllegalArgumentException: Transaction already
 present in the group, org.springframework.security.web.header.HeaderWriterFilter
$HeaderWriterResponse@e34f6b8]
2023-06-14 18:10:54 default         ERROR - c.n.b.j.a.WebConfig                    -
 400
java.lang.IllegalArgumentException: Transaction already present in the group
 at
 com.netfective.bluage.jac.server.services.rest.impl.JicsServiceImpl.createElement(JicsServiceI
```

Crie um ID de transação.

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/createTransaction`
- Argumentos: uma carga JSON, representando a TRANSAÇÃO JICS a ser criada. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACTransaction`.
- Retorna um valor Booleano. Se o valor for 'true', a TRANSAÇÃO foi criada corretamente no armazenamento JICS subjacente.

Como criar um programa

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER

- Caminho: `/api/services/rest/jicsservice/createProgram`
- Argumentos: uma carga JSON, representando o PROGRAMA JICS a ser criado. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACProgram`.
- Retorna um valor Booleano. Se o valor for 'true', o PROGRAMA foi criado corretamente no armazenamento JICS subjacente.

Crie um ARQUIVO

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/createFile`
- Argumentos: uma carga JSON, representando o ARQUIVO JICS a ser criado. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACFile`.
- Retorna um valor Booleano. Se o valor for 'true', o ARQUIVO foi criado corretamente no armazenamento JICS subjacente.

Crie um TDQUEUE

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/createTDQueue`
- Argumentos: uma carga JSON, representando o JICS TDQUEUE a ser criado. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACTDQueue`.
- Retorna um valor Booleano. Se o valor for 'true', o TDQUEUE foi criado corretamente no armazenamento JICS subjacente.

Cria um modelo.

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/createTSModel`

- Argumentos: uma carga JSON, representando o JICS TSMODEL a ser criado. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACTSModel`.
- Retorna um valor booleano que `true` indica que a criação de elementos foi operada com sucesso no armazenamento JICS subjacente.

Crie elementos

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/createElements`
- Argumentos: uma carga JSON que representa os elementos JICS a serem criados.
- Retorna um valor Booleano. Se o valor for 'true', o TSMODEL foi criado corretamente no armazenamento JICS subjacente.

Atualizar um link

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/updateList`
- Argumentos: uma carga JSON, representando a LISTA JICS a ser atualizada. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACList`. Não há necessidade de fornecer os filhos do LIST, o mecanismo de atualização do LIST não levará isso em consideração.
- Retorna um valor Booleano. Se o valor for 'true', a LIST foi atualizada corretamente no armazenamento JICS subjacente.

A atualização do sinalizador LIST 'isActive' será propagada para todos os elementos de propriedade da LIST, ou seja, todos os GRUPOS pertencentes à LIST e todos os RECURSOS pertencentes a esses GRUPOS. Essa é uma maneira conveniente de desativar muitos recursos com uma única operação, em vários GRUPOS.

Atualiza um grupo.

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/updateGroup`
- Argumentos: uma carga JSON, representando o JICS GRUPO a ser atualizado. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACGroup`. Não há necessidade de fornecer os filhos do GRUPO, o mecanismo de atualização do GRUPO não levará isso em consideração.
- Retorna um valor Booleano. Se o valor for 'true', o GRUPO foi atualizado corretamente no armazenamento JICS subjacente.

> **Note**
>
> A atualização do sinalizador GRUPO 'isActive' se propagará para todos os elementos de propriedade do GRUPO, ou seja, todos os RECURSOS de propriedade do GRUPO. Essa é uma maneira conveniente de desativar muitos recursos com uma única operação em um determinado GRUPO.

Considerações sobre a atualização de RECURSOS COMUNS

Todos os endpoints a seguir tratam da atualização do JICS RESOURCES. Usando o `groupName` campo, você pode alterar o GRUPO proprietário de qualquer RECURSO JICS, desde que o valor do campo aponte para um GRUPO existente no armazenamento JICS subjacente (caso contrário, você receberá uma resposta BAD REQUEST (HTTP STATUS 400) do endpoint).

Atualizar uma TRANSAÇÃO

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/updateTransaction`

- Argumentos: uma carga JSON, representando a TRANSAÇÃO JICS a ser atualizada. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACTransaction`.
- Retorna um valor Booleano. Se o valor for 'true', a TRANSAÇÃO foi atualizada corretamente no armazenamento JICS subjacente.

Atualizar um PROGRAMA

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/updateProgram`
- Argumentos: uma carga JSON, representando o PROGRAMA JICS a ser atualizado. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACProgram`.
- Retorna um valor Booleano. Se o valor for 'true', o PROGRAMA foi atualizado corretamente no armazenamento JICS subjacente.

Atualizar um ARQUIVO

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/updateFile`
- Argumentos: uma carga JSON, representando o ARQUIVO JICS a ser atualizado. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACFile`.
- Retorna um valor Booleano. Se o valor for 'true', o ARQUIVO foi atualizado corretamente no armazenamento JICS subjacente.

Atualizar um TDQUEUE

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/updateTDQueue`

- Argumentos: uma carga JSON, representando o JICS TDQUEUE a ser atualizado. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACTDQueue`.
- Retorna um valor Booleano. Se o valor for 'true', o TDQueue foi atualizado corretamente no armazenamento JICS subjacente.

Atualiza um modelo.

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/updateTSModel`
- Argumentos: uma carga JSON, representando o JICS TSMODEL a ser atualizado. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.JACTSModel`.
- Retorna um valor Booleano. Se o valor for 'true', o TSMODEL foi atualizado corretamente no armazenamento JICS subjacente.

Atualizar elementos

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/updateElements`
- Argumentos: uma carga JSON que representa os elementos a serem atualizados.
- Retorna um valor booleano que `true` indica que a atualização dos elementos foi operada com êxito no armazenamento JICS subjacente.

Elementos upsert

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/upsertElements`
- Argumentos: uma carga JSON que representa os elementos a serem alterados.

- Retorna um valor booleano que `true` indica que o upsert dos elementos foi operado com sucesso no armazenamento JICS subjacente.

Recuperar elementos

- Método compatível: GET
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/retrieveElements`
- Argumentos: nenhum
- Retorna uma lista de todos os recursos serializados do JICS.

Operação JICS CRUD

- Método compatível: POST
- Requer autenticação e as seguintes funções: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER
- Caminho: `/api/services/rest/jicsservice/jicsCrudOperation`
- Argumentos: uma carga JSON que representa os recursos do JICS que estamos procurando. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.request.JicsCrudOperationRequest`.
- Retorna uma carga JSON que representa a resposta. Essa é a serialização JSON de um objeto `com.netfective.bluage.jac.entities.request.JicsCrudOperationResponse`.

Outros

Tópicos

- Status de integridade do servidor JICS

Status de integridade do servidor JICS

- Método compatível: GET
- Caminho: `/api/services/rest/jicsserver/serverIsUp`
- Argumentos: nenhum

- Devoluções: Nenhuma. Uma resposta HTTP STATUS 200 indica que o servidor está funcionando.

Endpoints de gerenciamento de usuários do JAC

Use os seguintes endpoints para gerenciar as interações do usuário.

Tópicos

Fazer login como usuário

- Método compatível: POST
- Caminho: `/api/services/security/servicelogin/login`
- Argumentos: nenhum
- Retorna a serialização JSON de um objeto `com.netfective.bluage.jac.entities.SignOn`, representando o usuário cujas credenciais são fornecidas na solicitação atual. A senha está oculta da exibição no objeto retornado. As funções atribuídas ao usuário estão sendo listadas.

Resposta de exemplo:

```
{
    "login": "some-admin",
    "password": null,
    "roles": [
      {
        "id": 0,
        "roleName": "ROLE_ADMIN"
      }
```

```
    ]
  }
```

Testando se existe pelo menos um usuário no sistema

- Método compatível: GET
- Caminho: `/api/services/security/servicelogin/hasAccount`
- Argumentos: nenhum
- Retorna o valor booleano `true` se pelo menos um usuário diferente do usuário superadministrador padrão tiver sido criado. `false`Caso contrário, retorna.

Gravação de um novo usuário

- Método compatível: POST
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/security/servicelogin/recorduser`
- Argumentos: a serialização JSON de um objeto `com.netfective.bluage.jac.entities.SignOn`, representando o usuário a ser adicionado ao armazenamento. As funções do usuário devem ser definidas, caso contrário, o usuário talvez não consiga usar o recurso e os endpoints do JAC.
- Retorna o valor booleano `true` se o usuário tiver sido criado com sucesso. `false`Caso contrário, retorna.

Exemplo de solicitação:

```
{
    "login": "simpleuser",
    "password": "simplepassword",
    "roles": [
      {
        "id": 2,
        "roleName": "ROLE_USER"
      }
    ]
  }
```

Somente as seguintes funções podem ser usadas ao gravar um novo usuário:

- ROLE_ADMIN: pode gerenciar recursos e usuários do JICS.
- ROLE_USER: pode gerenciar recursos do JICS, mas não usuários.

Informações do usuário

- Método compatível: GET
- Requer autenticação e direitos de administrador
- Caminho: `/api/services/security/servicelogin/userInfo`
- Retorna o nome de usuário e as funções do usuário atualmente conectado.

Listar usuários

- Método compatível: GET
- Requer autenticação e a função ROLE_ADMIN.
- Caminho: `/api/services/security/servicelogin/listusers`
- Argumentos: nenhum
- Retorna uma lista de`com.netfective.bluage.jac.entities.SignOn`, serializada como JSON.

Excluir um usuário

- Método compatível: POST
- Caminho: `/api/services/security/servicelogin/deleteuser`
- Argumentos: a serialização JSON de um `com.netfective.bluage.jac.entities.SignOn` objeto que representa o usuário a ser removido do armazenamento.
- Retorna o valor booleano `true` se o usuário tiver sido removido com sucesso.

> **Important**
>
> Esta ação não pode ser desfeita. O usuário excluído não conseguirá se conectar ao aplicativo JAC novamente.

Fazer logout do usuário atual

- Método compatível: GET
- Caminho: `/api/services/security/servicelogout/logout`
- Argumentos: nenhum
- Retorna a mensagem JSON `{"success":true}` se o usuário atual tiver sido desconectado com sucesso. A sessão HTTP relacionada será invalidada.

Estruturas de dados

Esta seção descreve os detalhes das várias estruturas de dados.

Tópicos

Estrutura de mensagens do Job Execution Details

Cada detalhe da execução do trabalho terá os seguintes campos:

scriptId

: o identificador do script chamado.

chamador

: Endereço IP do chamador.

Identifier

: identificador exclusivo de execução do trabalho.

startTime

data e hora em que a execução de trabalho foi iniciada.

endTime

data e hora em que a execução de trabalho foi encerrada.

status

um status para a execução do trabalho. Um valor possível entre:

- `DONE`: a execução do trabalho terminou normalmente.
- `TRIGGERED`: a execução do trabalho foi acionada, mas ainda não foi lançada.
- `RUNNING`: a execução do trabalho está em execução.
- `KILLED`: a execução do trabalho foi interrompida.
- `FAILED`: a execução do trabalho falhou.

executionResult

uma mensagem para resumir o resultado da execução do trabalho. Essa mensagem pode ser uma mensagem simples se a execução do trabalho ainda não tiver sido concluída ou uma estrutura JSON com os seguintes campos:

- exitCode: código de saída numérico; valores negativos indicam situações de falha.
- program: último programa lançado pelo cargo.
- status: um valor possível entre:
  - `Error`: quando exitCode = -1; isso corresponde a um erro (técnico) que ocorre durante a execução do trabalho.
  - `Failed`: quando exitCode = -2; Isso corresponde a uma falha que ocorre durante a execução de um programa de serviço (como uma situação ABEND).
  - `Succeeded`: quando exitCode >= 0;
- stepName: nome da última etapa executada no trabalho.

executionMode

síncrona ou assíncrona, dependendo da forma como a tarefa foi iniciada.

Exemplo de resultado:

```
{
```

```
    "scriptId": "INTCALC",
    "caller": "127.0.0.1",
    "identifier": "97d410be-efa7-4bd3-b7b9-d080e5769771",
    "startTime": "06-09-2023 11:42:41",
    "endTime": "06-09-2023 11:42:42",
    "status": "DONE",
    "executionResult": "{ \"exitCode\": -1, \"stepName\": \"STEP15\", \"program\":
\"CBACT04C\", \"status\": \"Error\" }",
    "executionMode": "ASYNCHRONOUS"
 }
```

Estrutura de resultados do lançamento da transação

A estrutura pode conter os seguintes campos:

outCome

uma string representando o resultado da execução da transação. Os valores possíveis são:

- `Success`: a execução da transação foi até o final corretamente.
- `Failure`: a execução da transação falhou ao terminar corretamente, alguns problemas foram encontrados.

commarea

uma string representando o valor final COMMAREA, como uma matriz de bytes codificada em byte64. Pode ser uma string vazia.

containerRecord

(opcional) uma string representando o conteúdo do registro do CONTAINER como uma matriz de bytes codificada em byte64.

serverDescription

Pode conter informações sobre o servidor que atendeu à solicitação (para fins de depuração). Pode ser uma string vazia.

abendCode

(opcional) se o programa referenciado pela transação lançada for alterado, o valor do código de abend será retornado como uma string nesse campo.

Respostas de exemplo:

Bem-sucedida

```
{
    "outCome": "Success",
    "commarea": "",
    "serverDescription": ""
  }
```

Falha

```
{
    "outCome": "Failure",
    "commarea": "",
    "serverDescription": "",
    "abendCode": "AEIA"
  }
```

Estrutura de resultados do registro de lançamento da transação

A estrutura pode conter os seguintes campos:

recordContent

uma string representando o conteúdo do registro do COMMAREA como uma matriz de bytes codificada em byte64.

containerRecord

uma string representando o conteúdo do registro do CONTAINER como uma matriz de bytes codificada em byte64.

serverDescription

Pode conter informações sobre o servidor que atendeu à solicitação (para fins de depuração). Pode ser uma string vazia.

Respostas de exemplo:

Bem-sucedida

```
{
    "recordContent": "",
```

```
    "serverDescription": ""
}
```

Possível status de trabalho em uma fila

Em uma fila, os trabalhos podem ter o seguinte status:

ACTIVE

O trabalho está sendo executado atualmente na fila.

EXECUTION_WAIT

O trabalho está aguardando a disponibilidade de um thread.

SCHEDULED

Os trabalhos são programados para execução em uma data e hora específicas.

HOLD

Job está esperando para ser lançado antes de ser executado.

CONCLUÍDO

Job foi executado com sucesso.

COM FALHA

Houve falha na execução de trabalho.

UNKNOWN

O status é desconhecido.

Envie uma tarefa e agende a entrada da tarefa

A entrada de envio de tarefas e agendamento de tarefas é a serialização JSON de um `com.netfective.bluage.gapwalk.rt.jobqueue.SubmitJobMessage` objeto. O exemplo de entrada abaixo exibe todos os campos desse tipo de feijão.

Exemplo de entrada para enviar um trabalho:

```
{
    "messageQueueName":null,
    "scheduleDate":null,
```

```
    "scheduleTime":null,
    "programName":"PTA0044",
    "programParams":
     {"wmind":"B"},
    "localDataAreaValue":"",
    "userName":"USER1",
    "jobName":"PTA0044",
    "jobNumber":9,
    "jobPriority":5,
    "executionDate":"20181231",
    "jobQueue":"queue1",
    "jobOnHold":false
}
```

Exemplo de entrada para agendar um trabalho:

```
{
     "scheduleCron": "*/2 * * * * ?",
     "programName":"LOGPGM",
     "programParams": {
         "cl_sbmjob_param_json": "[\"./output/schedule-job-log.txt\", \"Every 2
 seconds!\"]"
     },
     "localDataAreaValue":"",
     "userName":"PVO",
     "jobName":"LOGGERJOB",
     "jobPriority":5,
     "jobQueue":"queue1",
     "scheduleMisfirePolicy": 4,
     "startTime": "2003/05/04 07:00:00.000 GMT-06:00",
     "endTime": "2003/05/04 07:00:07.000 GMT-06:00"
 }
```

jobNumber

se o número do trabalho for 0, o número do trabalho será gerado automaticamente usando o próximo número na sequência numérica do trabalho. Esse valor deve ser definido como 0 (exceto para fins de teste).

jobPriority

A prioridade de trabalho padrão no AS400 é 5. O intervalo válido é de 0 a 9, sendo 0 a prioridade mais alta.

jobOnHold

Se um trabalho for enviado em espera, ele não será executado imediatamente, mas somente quando alguém o “liberar”. Um trabalho pode ser lançado usando a API REST (/release ou / release-all).

scheduleDate e scheduleTime

Se esses valores não forem nulos, o trabalho será executado na data e hora especificadas.

Data

Pode ser fornecido com o formato MMDdyy ou ddmMyYYY (o tamanho da entrada determinará qual formato será usado)

Tempo

Pode ser fornecido com o formato HHmm ou HHMMss (o tamanho da entrada determinará qual formato será usado)

programParams

Será passado para o programa como um mapa.

scheduleMisfirePolicy

Define a estratégia usada quando um gatilho falha no disparo. Os valores possíveis são os seguintes:

1. Solte a primeira falha de ignição e descarte as outras falhas de ignição.
2. Envie um trabalho suspenso para a primeira falha de ignição e descarte as outras falhas de ignição.
3. Descarte a falha de ignição.
4. Libere todas as falhas de ignição. A fila de trabalhos executará todos os trabalhos.

Lista de respostas de trabalhos agendados

Essa é a estrutura do endpoint da fila de trabalhos list-jobs. A mensagem de envio do trabalho que foi usada para enviar esse trabalho faz parte da resposta. Isso pode ser usado para fins de rastreamento, teste/reenvio. Quando um trabalho for concluído, a data de início e a data de término também serão preenchidas.

```
[
  {
```

```
    "jobName": "PTA0044",
    "userName": "USER1",
    "jobNumber": 9,
    "jobPriority": 5,
    "status": "HOLD",
    "jobDelay": 0,
    "startDate": null,
    "endDate": null,
    "jobQueue": "queue1",
    "message": {
      "messageQueueName": null,
      "scheduleDate": null,
      "scheduleTime": null,
      "programName": "PTA0044",
      "programParams": {"wmind": "B"},
      "localDataAreaValue": "",
      "userName": "USER1",
      "jobName": "PTA0044",
      "jobNumber": 9,
      "jobPriority": 5,
      "executionDate": "20181231",
      "jobQueue": "queue1",
      "jobOnHold": true,
      "scheduleCron": null,
      "save": false,
      "scheduleMisfirePolicy": 4,
      "omitdates": null
    },
    "executionId": 1,
    "jobScheduledId": 0,
    "jobScheduledAt": null
  },
  {
    "jobName": "PTA0044",
    "userName": "USER1",
    "jobNumber": 9,
    "jobPriority": 5,
    "status": "COMPLETED",
    "jobDelay": 0,
    "startDate": "2022-10-13T22:48:34.025+00:00",
    "endDate": "2022-10-13T22:52:54.475+00:00",
    "jobQueue": "queue1",
    "message": {
      "messageQueueName": null,
```

```
      "scheduleDate": null,
      "scheduleTime": null,
      "programName": "PTA0044",
      "programParams": {"wmind": "B"},
      "localDataAreaValue": "",
      "userName": "USER1",
      "jobName": "PTA0044",
      "jobNumber": 9,
      "jobPriority": 5,
      "executionDate": "20181231",
      "jobQueue": "queue1",
      "jobOnHold": true,
      "scheduleCron": "*/20 * * * * ?",
      "save": false,
      "scheduleMisfirePolicy": 4,
      "omitdates": null
    },
    "executionId": 2,
    "jobScheduledId": 0,
    "jobScheduledAt": null
  }
]
```

Lista de respostas de trabalhos repetidos

Essa é a estrutura do endpoint da fila de trabalhos /schedule/list.

```
[
  {
    "id": 1,
    "status": "ACTIVE",
    "jobNumber": 1,
    "userName": "PVO",
    "msg": {
      "messageQueueName": null,
      "scheduleDate": null,
      "scheduleTime": null,
      "startTime": "2024/03/07 21:12:00.000 UTC",
      "endTime": "2024/03/07 21:13:59.000 UTC",
      "programName": "LOGPGM",
      "programParams": {"cl_sbmjob_param_json": "[\"./output/schedule-job-log.txt\",
 \"Every 20 seconds!\"]"},
      "localDataAreaValue": "",
```

```
      "userName": "PVO",
      "jobName": "LOGGERJOB",
      "jobNumber": 1,
      "jobScheduleId": 1,
      "jobPriority": 5,
      "executionDate": null,
      "jobQueue": "queue1",
      "jobOnHold": false,
      "scheduleCron": "*/20 * * * * ?",
      "save": false,
      "scheduleMisfirePolicy": 4,
      "omitdates": null
    },
    "lastUpdatedAt": "2024-03-07T21:11:13.282+00:00",
    "lastUpdatedBy": ""
  }
]
```

AWS Configuração do Blu Age Runtime (não gerenciada)

Esta seção explica as etapas para configurar o AWS Blu Age Runtime (não gerenciado) em sua AWS infraestrutura.

Tópicos

- AWS Pré-requisitos do Blu Age Runtime
- AWS Integração do Blu Age Runtime
- Requisitos de configuração de infraestrutura para o AWS Blu Age Runtime (não gerenciado)
- AWS Implantação do Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate
- AWS Implantação do Blu Age Runtime no Amazon EC2
- Teste o PlanetsDemo aplicativo

AWS Pré-requisitos do Blu Age Runtime

AWS O Blu Age Runtime (não gerenciado) está disponível em várias versões de lançamento. Se você tiver projetos de modernização em andamento, talvez precise de versões incrementais do tempo de execução para fins de implementação e teste. Para definir suas necessidades, entre em contato com seu gerente de entrega da AWS Blu Age.

Antes de iniciar o processo de integração do AWS Blu Age Runtime (não gerenciado), faça o seguinte:

- Verifique se você tem uma AWS conta.
- Certifique-se de ter um aplicativo modernizado refatorado com o Blu Age. AWS
- Escolha uma AWS região e uma das opções de computação compatíveis com o AWS Blu Age Runtime (não gerenciado).
- Escolha a versão do AWS Blu Age Runtime que você deseja usar.
- Analise the section called “Requisitos de configuração de infraestrutura” e valide os componentes adicionais necessários para executar o AWS Blu Age Runtime (não gerenciado).

> **Note**
>
> Se quiser testar os recursos do AWS Blu Age Runtime (não gerenciado), você pode usar o aplicativo de demonstraçãoPlanets Demo, que pode ser baixado do PlanetsDemo -v1.zip.

AWS Integração do Blu Age Runtime

Para começar, crie um AWS Support caso para solicitar a integração para acessar o AWS Blu Age Runtime. Inclua em sua solicitação seu Conta da AWS ID, a AWS região que você deseja usar e uma opção de computação e uma versão de tempo de execução. Se você não tiver certeza de qual versão precisa, entre em contato com seu gerente de entrega da AWS Blu Age.

> **Note**
>
> O AWS Blu Age Runtime está disponível em duas variedades principais: pré-lançamentos alfa e lançamentos oficiais. Para determinar qual versão usar, consulte Get Started no site Blu Insights ou entre em contato com seu gerente de entrega da AWS Blu Age.

AWS Blu Age Runtime (não gerenciado) no Amazon EC2

Armazenamos os artefatos do AWS Blu Age Runtime (não gerenciados) em diferentes buckets do Amazon S3 por região e por opção de computação. Para acessar o bucket do seu Região da AWS for AWS Blu Age Runtime (não gerenciado) no Amazon EC2, use o nome listado na tabela a seguir.

> **Note**
>
> Essa tabela se aplica ao Amazon EC2, bem como às instâncias do Amazon EC2 usadas no Amazon ECS e no Amazon EKS.

Região da AWS	Balde de liberação	Caçamba de pré-lançamento
Leste dos EUA (Ohio)	aws-bluage-runtime-artifacts-055777665268-us-east-2	aws-bluage-runtime-artifacts-dev-055777665268-us-east-2
Leste dos EUA (Norte da Virgínia)	aws-bluage-runtime-artifacts-139023371234-us-east-1	aws-bluage-runtime-artifacts-dev-139023371234-us-east-1
Oeste dos EUA (N. da Califórnia)	aws-bluage-runtime-artifacts-788454048782-us-west-1	aws-bluage-runtime-artifacts-dev-788454048782-us-west-1
Oeste dos EUA (Oregon)	aws-bluage-runtime-artifacts-836771190483-us-west-2	aws-bluage-runtime-artifacts-dev-836771190483-us-west-2
Europa (Irlanda)	aws-bluage-runtime-artifacts-925278190477-eu-west-1	aws-bluage-runtime-artifacts-dev-925278190477-eu-west-1
Europa (Paris)	aws-bluage-runtime-artifacts-673009995881-eu-west-3	aws-bluage-runtime-artifacts-dev-673009995881-eu-west-3
Europa (Frankfurt)	aws-bluage-runtime-artifacts-485196800481-eu-central-1	aws-bluage-runtime-artifacts-dev-485196800481-eu-central-1
América do Sul (São Paulo)	aws-bluage-runtime-artifacts-737536804457-sa-lest-1	aws-bluage-runtime-artifacts-dev-737536804457-sa-east-1
Ásia-Pacífico (Tóquio)	aws-bluage-runtime-artifacts-445578176276-ap-nordeste-1	aws-bluage-runtime-artifacts-dev-445578176276-ap-nordeste-1

Região da AWS	Balde de liberação	Caçamba de pré-lançamento
Ásia-Pacífico (Sydney)	aws-bluage-runtime-artifacts-726160321909-ap-sudeste-2	aws-bluage-runtime-artifacts-dev-726160321909-ap-southeast-2

AWS Blu Age Runtime (não gerenciado) no Amazon ECS gerenciado pela Fargate

Armazenamos os artefatos do AWS Blu Age Runtime (não gerenciados) em diferentes buckets do Amazon S3 por região e por opção de computação. Para acessar o bucket do seu Região da AWS for AWS Blu Age Runtime (não gerenciado) no Amazon ECS gerenciado pela Fargate, use o nome listado na tabela a seguir.

Região da AWS	Balde de liberação	Caçamba de pré-lançamento
Leste dos EUA (Ohio)	aws-bluage-runtime-fargate-rel-483416914331-us-east-2	aws-bluage-runtime-fargate-dev-483416914331-us-east-2
Leste dos EUA (Norte da Virgínia)	aws-bluage-runtime-fargate-rel-308472162679-us-east-1	aws-bluage-runtime-fargate-dev-308472162679-us-east-1
Oeste dos EUA (N. da Califórnia)	aws-bluage-runtime-fargate-rel-343763094578-us-west-1	aws-bluage-runtime-fargate-dev-343763094578-us-west-1
Oeste dos EUA (Oregon)	aws-bluage-runtime-fargate-rel-688933007849-us-west-2	aws-bluage-runtime-fargate-dev-688933007849-us-west-2
Europa (Irlanda)	aws-bluage-runtime-fargate-rel-140138033705-eu-west-1	aws-bluage-runtime-fargate-dev-140138033705-eu-west-1
Europa (Paris)	aws-bluage-runtime-fargate-rel-339712948211-eu-west-3	aws-bluage-runtime-fargate-dev-339712948211-eu-west-3
Europa (Frankfurt)	aws-bluage-runtime-fargate-rel-339712918892-eu-central-1	aws-bluage-runtime-fargate-dev-339712918892-eu-central-1

Região da AWS	Balde de liberação	Caçamba de pré-lançamento
América do Sul (São Paulo)	aws-bluage-runtime-fargate-rel-767397998881-sa-lest-1	aws-bluage-runtime-fargate-dev-767397998881-sa-east-1
Ásia-Pacífico (Tóquio)	aws-bluage-runtime-fargate-rel-891377400849-ap-nordeste-1	aws-bluage-runtime-fargate-dev-891377400849-ap-nordeste-1
Ásia-Pacífico (Sydney)	aws-bluage-runtime-fargate-rel-533267435478-ap-sudeste-2	aws-bluage-runtime-fargate-dev-533267435478-ap-southeast-2

Usando o AWS CLI para listar o conteúdo do bucket

Depois de ser integrado, você pode listar o conteúdo do bucket executando o AWS CLI comando a seguir em um terminal.

```
aws s3 ls bucket-name
```

`bucket-name`Substitua pelo nome do seu bucket Região da AWS da tabela anterior.

Esse comando retorna uma lista de pastas que correspondem a diferentes versões do tempo de execução do AWS Blu Age Runtime (não gerenciado), como a seguinte para um bucket de lançamento:

```
PRE 3.10.0/
PRE 4.0.0/
```

Ou o seguinte para um repositório de compilação:

```
PRE 4.1.0-alpha.8/
PRE 4.1.0-alpha.9/
```

É recomendável usar a versão mais recente disponível. Se isso não for possível, use a versão de tempo de execução que foi validada durante a fase de refatoração do aplicativo. Para listar as estruturas disponíveis para uma versão específica, execute o seguinte comando:

```
aws s3 ls s3://bucket-name/version/Framework/
```

`bucket-name`Substitua pelo nome do bucket para você Região da AWS e `version` pela versão desejada. A seguir estão dois exemplos.

Para um balde de liberação:

```
aws s3 ls s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/
Framework/
```

O comando retorna uma lista de estruturas, como:

```
2024-04-08 16:11:19  152040176 aws-bluage-runtime-4.0.0.tar.gz
2024-04-08 16:11:50         45 aws-bluage-runtime-4.0.0.tar.gz.checksumSHA256
2024-04-08 16:11:52  176518889 aws-bluage-webapps-4.0.0.tar.gz
2024-04-08 16:12:28         45 aws-bluage-webapps-4.0.0.tar.gz.checksumSHA256
```

Para um repositório de compilação:

```
aws s3 ls s3://aws-bluage-runtime-artifacts-dev-139023371234-us-
east-1/4.1.0-alpha.9/Framework/
```

O comando retorna uma lista de estruturas, como:

```
2024-04-09 20:23:34  152304534 aws-bluage-runtime-4.1.0-alpha.9.tar.gz
2024-04-09 20:24:05         45 aws-bluage-runtime-4.1.0-alpha.9.tar.gz.checksumSHA256
2024-04-09 20:24:07  176262381 aws-bluage-webapps-4.1.0-alpha.9.tar.gz
2024-04-09 20:24:42         45 aws-bluage-webapps-4.1.0-alpha.9.tar.gz.checksumSHA256
```

Baixe o framework

Você pode baixar a estrutura, por exemplo, para atualizar a versão AWS Blu Age Runtime em uma instância existente do Amazon EC2.

```
aws s3 cp s3://bucket-name/version/Framework/ folder-of-your-choice --
recursive
```

Em que:

folder-of-your-choice

caminho da pasta onde você gostaria de baixar a estrutura.

Por exemplo: `aws s3 cp s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/Framework/ . --recursive`

Esse comando produzirá a saída a seguir:

```
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/
Framework/aws-bluage-runtime-4.0.0.tar.gz.checksumSHA256 to ./aws-bluage-
runtime-4.0.0.tar.gz.checksumSHA256
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/
Framework/aws-bluage-webapps-4.0.0.tar.gz.checksumSHA256 to ./aws-bluage-
webapps-4.0.0.tar.gz.checksumSHA256
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/Framework/aws-
bluage-webapps-4.0.0.tar.gz to ./aws-bluage-webapps-4.0.0.tar.gz
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/Framework/aws-
bluage-runtime-4.0.0.tar.gz to ./aws-bluage-runtime-4.0.0.tar.gz
```

Você pode listar os arquivos da estrutura da seguinte forma:

`ls -l`

Esse comando produzirá a saída a seguir:

```
total 230928
-rw-rw-r-- 1 cloudshell-user cloudshell-user 152040176 Apr  8 16:11 aws-bluage-
runtime-4.0.0.tar.gz
-rw-rw-r-- 1 cloudshell-user cloudshell-user        45 Apr  8 16:11 aws-bluage-
runtime-4.0.0.tar.gz.checksumSHA256
-rw-rw-r-- 1 cloudshell-user cloudshell-user 176518889 Apr  8 16:11 aws-bluage-
webapps-4.0.0.tar.gz
-rw-rw-r-- 1 cloudshell-user cloudshell-user        45 Apr  8 16:12 aws-bluage-
webapps-4.0.0.tar.gz.checksumSHA256
```

Requisitos de configuração de infraestrutura para o AWS Blu Age Runtime (não gerenciado)

Este tópico descreve a configuração mínima de infraestrutura necessária para executar o AWS Blu Age Runtime (não gerenciado). Os procedimentos a seguir descrevem como configurar o AWS Blu Age Runtime (não gerenciado) na computação de sua escolha para implantar um aplicativo modernizado no Blu Age Runtime. AWS Os recursos que você cria devem estar em uma Amazon VPC que tenha uma sub-rede dedicada ao domínio do seu aplicativo.

Tópicos

Requisitos de infraestrutura

Criar um grupo de segurança

Se você planeja trabalhar em instâncias do Amazon EC2 no Amazon EKS, ignore esse procedimento porque o processo de criação do cluster do Amazon EKS cria um grupo de segurança em seu nome. Use esse grupo de segurança nos procedimentos a seguir em vez de criar um novo.

1. Abra o console do Amazon VPC em https://console.aws.amazon.com/vpc/.
2. No painel de navegação esquerdo, em Segurança, escolha Grupos de segurança.
3. No painel central, escolha Criar grupo de segurança.
4. No campo Nome do grupo de segurança, insira **`M2BluagePrivateLink-SG`**.
5. Na seção Regras de entrada, escolha Adicionar regra.
6. Para Tipo, escolha HTTPS.
7. Em Origem, insira seu CIDR da VPC.
8. Na seção Regras de saída, escolha Adicionar regra.
9. Para Tipo, escolha HTTPS.
10. Em Destination, insira **`0.0.0.0/0`**.
11. Escolha Create security group (Criar grupo de segurança).

Criar um endpoint da Amazon VPC

1. Abra o console do Amazon VPC em https://console.aws.amazon.com/vpc/.
2. No menu à de navegação esquerdo, em Nuvem privada virtual, escolha Endpoints.
3. No painel central, escolha Criar endpoint.

4. Na seção Serviços, insira **SQS** o campo de pesquisa e selecione o serviço Amazon SQS que corresponde à sua região.
5. Em VPC, selecione a Amazon VPC criada na etapa anterior.
6. Na seção Sub-redes, selecione a sub-rede que você criou para o domínio da aplicação.
7. Na seção Grupos de segurança, selecione o grupo de segurança do procedimento anterior.
8. Escolha Criar endpoint.

Criar uma política do IAM

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação esquerdo, em Gerenciamento de acesso, escolha Políticas.
3. No painel central, escolha Criar política.
4. Na seção Editor de políticas, escolha JSON.
5. Substitua todo o JSON que você vê no editor pelo seguinte JSON:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "sqs:GetQueueUrl",
                "sqs:ReceiveMessage",
                "sqs:SendMessage"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Se precisar de mais detalhes para personalizar sua política, entre em contato com seu gerente de entrega ou gerente de contas da AWS Blu Age.

6. Escolha Próximo.

7. Insira um nome para a política e escolha Criar política.

Criar um perfil do IAM

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação esquerdo, em Gerenciamento de acesso, escolha Perfis.
3. No painel central, escolha Criar perfil.
4. Na seção Caso de uso, dependendo da sua escolha de computação, escolha uma das seguintes opções:
   - EC2 (para Amazon EC2 e Amazon EKS no Amazon EC2)
   - Elastic Container Service e, em seguida, EC2 Role for Elastic Container Service (para Amazon ECS no Amazon EC2)
   - Elastic Container Service e depois Elastic Container Service Task (para Amazon ECS gerenciado pela Fargate)
5. Escolha Próximo.
6. No campo de pesquisa, insira o nome da política que você criou anteriormente.
7. Marque a caixa de seleção à esquerda da política.

> **Note**
>
> Se você não conseguir adicionar uma política, conclua a criação da função e atualize a função para adicionar a política.

8. Escolha Próximo.
9. Insira um nome para o perfil e escolha Criar perfil.

Tipos de instância do Amazon EC2 para AWS Blu Age Runtime (no Amazon EC2)

A seguir está uma lista dos tipos de instância do Amazon EC2 que você pode usar para o AWS Blu Age Runtime (no Amazon EC2) ao criar instâncias do Amazon EC2 ou ao definir nós de trabalho do Amazon EKS.

```
t3.xlarge
t3.small
t3.large
```

```
t2.small
t2.large
r7a.medium
r7a.large
r7a.xlarge
r7a.2xlarge
r7a.4xlarge
r7a.8xlarge
r7a.12xlarge
r7a.16xlarge
r7a.24xlarge
r7a.32xlarge
r7a.48xlarge
r7a.metal-48xl
r7i.large
r7i.xlarge
r7i.2xlarge
r7i.4xlarge
r7i.8xlarge
r7i.12xlarge
r7i.16xlarge
r7i.24xlarge
r7i.48xlarge
r7i.metal-24xl
r7i.metal-48xl
r6i.xlarge
r6i.large
r6i.4xlarge
r6i.2xlarge
r5b.xlarge
r5b.large
r5b.2xlarge
r3.xlarge
m6i.xlarge
m6i.large
m6i.8xlarge
m6i.4xlarge
m6i.2xlarge
m6i.16xlarge
m5zn.xlarge
m5zn.large
m5zn.3xlarge
m5zn.2xlarge
m5.xlarge
```

```
m5.large
m5.8xlarge
m5.4xlarge
m5.2xlarge
m5.16xlarge
m5.12xlarge
c6i.xlarge
c6i.large
c6i.8xlarge
c6i.4xlarge
c6i.2xlarge
c6i.16xlarge
c5.xlarge
c5.large
c5.9xlarge
c5.4xlarge
c5.2xlarge
c5.18xlarge
c5.12xlarge
```

Executando o AWS Blu Age Runtime no Amazon EC2

Para criar uma instância do Amazon EC2, use as etapas a seguir.

Criar uma instância do Amazon EC2

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Escolha Iniciar instância.
3. Em Tipo de instância, escolha um dos tipos listados emthe section called “Tipos de instância do Amazon EC2 para AWS Blu Age Runtime (no Amazon EC2)”.
4. Na seção Par de chaves, selecione um par de chaves existente ou crie um.
5. Na seção Configurações de rede, escolha Selecionar grupo de segurança existente.
6. Para Grupos de segurança comuns, escolha M2 BluagePrivateLink -SG.
7. Expanda a seção Detalhes avançados.
8. Em Perfil de instância do IAM, selecione o perfil do IAM que você criou anteriormente.
9. Escolha Iniciar instância.

Instale o aplicativo na instância do Amazon EC2

1. Quando o estado da instância do Amazon EC2 mudar para Running, conecte-se à instância.
2. Instale os seguintes componentes de software na instância:

   - Ambiente de execução Java (JRE) 17.
   - Apache Tomcat 10.
   - AWS Blu Age Runtime (no Amazon EC2). Instale o runtime do AWS Blu Age na raiz da pasta de instalação do Apache Tomcat (alguns arquivos serão adicionados e outros serão sobrescritos).

   Para instalar os aplicativos web adicionais fornecidos junto com o arquivo AWS Blu Age Runtime, configure uma instância secundária do servidor Apache Tomcat e descompacte o arquivo de aplicativos web nesse local.

Executando o AWS Blu Age Runtime no Amazon ECS no Amazon EC2

1. Crie um cluster do Amazon ECS, com instâncias do Amazon EC2 como infraestrutura subjacente. Consulte Introdução ao Windows no Amazon EC2 no Amazon Elastic Container Service Developer Guide.
2. Especifique a função do IAM que você criou nas etapas anteriores.
3. Escolha um dos tipos de instância listados emthe section called “Tipos de instância do Amazon EC2 para AWS Blu Age Runtime (no Amazon EC2)”.
4. Em Configurações de rede para instâncias do Amazon EC2, escolha o grupo de segurança que você criou nas etapas anteriores.

Executando o AWS Blu Age Runtime no Amazon EKS no Amazon EC2

1. Crie um cluster do Amazon EKS. Consulte Criação de um cluster do Amazon EKS no Guia do usuário do Amazon EKS.
2. Conforme mencionado anteriormente, um grupo de segurança é criado em seu nome. Você pode usar esse grupo de segurança ao criar o endpoint da Amazon VPC.
3. Crie um grupo de nós. Especifique a função do IAM que você criou nas etapas anteriores.
4. Escolha um dos tipos de instância listados emthe section called “Tipos de instância do Amazon EC2 para AWS Blu Age Runtime (no Amazon EC2)”.

5. O Amazon EKS atribuirá automaticamente o grupo de segurança às instâncias geradas do Amazon EC2.

Executando o AWS Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate

Crie um cluster do Amazon ECS com o AWS Fargate (sem servidor) como uma infraestrutura subjacente. Consulte Introdução aos contêineres Linux no AWS Fargate no Guia do desenvolvedor do Amazon Elastic Container Service.

AWS Implantação do Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate

Os tópicos desta seção descrevem como configurar o AWS Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate, como atualizar a versão do tempo de execução, como monitorar sua implantação usando CloudWatch alarmes da Amazon e como adicionar dependências licenciadas.

Tópicos

- Configurando o AWS Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate
- Atualizando o AWS Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate
- Amazon CloudWatch Alarms for AWS Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate
- Configurando dependências licenciadas no AWS Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate

Configurando o AWS Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate

Este tópico explica como configurar e implantar o aplicativo de PlanetsDemo amostra usando o AWS Blu Age Runtime no Amazon ECS gerenciado por. AWS Fargate

AWS O Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate está disponível para Linux/X86.

Tópicos

- Pré-requisitos
- Configuração
- Teste o aplicativo implantado

Pré-requisitos

Antes de começar, certifique-se de que você concluiu os seguintes pré-requisitos.

- Configure o AWS CLI seguindo as etapas em Configuração da AWS CLI.
- Preencha the section called “AWS Pré-requisitos do Blu Age Runtime”, e the section called “AWS Integração do Blu Age Runtime”.
- Baixe o AWS Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate binários. Para obter instruções, consulte the section called “AWS Integração do Blu Age Runtime”.
- Baixe os binários do Apache Tomcat 10.
- Baixe o arquivo do PlanetsDemo aplicativo.
- Crie um banco de dados Amazon Aurora PostgreSQL para JICS e execute a consulta nele. `PlanetsDemo-v1/jics/sql/initJics.sql` Para obter informações sobre como criar um banco de dados Amazon Aurora PostgreSQL, consulte Criação e conexão com um cluster de banco de dados Aurora PostgreSQL.

Configuração

Para configurar o aplicativo de PlanetsDemo amostra, conclua as etapas a seguir.

1. Depois de baixar os binários do Apache Tomcat, extraia o conteúdo e vá para a pasta. `conf` Abra o `catalina.properties` arquivo para edição e substitua a linha que começa `common.loader` com a seguinte linha.

   ```
   common.loader="${catalina.base}/lib","${catalina.base}/lib/
   *.jar","${catalina.home}/lib","${catalina.home}/lib/*.jar","${catalina.home}/
   shared","${catalina.home}/shared/*.jar","${catalina.home}/extra","${catalina.home}/
   extra/*.jar"
   ```

2. Comprima a pasta Apache Tomcat usando o comando tar para criar um arquivo `tar.gz`.
3. Prepare um Dockerfile para criar sua imagem personalizada com base nos binários de tempo de execução fornecidos e nos binários do servidor Apache Tomcat. Veja o exemplo a seguir do Dockerfile. O objetivo é instalar o Apache Tomcat 10, seguido pelo AWS Blu Age Runtime (para Amazon ECS gerenciado por AWS Fargate) extraído na raiz do diretório de instalação do Apache Tomcat 10 e, em seguida, instalar o exemplo de aplicativo modernizado chamado. PlanetsDemo

> **Note**
>
> O conteúdo dos scripts install-gapwalk.sh e install-app.sh, usados neste exemplo do Dockerfile, está listado após o Dockerfile.

```
FROM --platform=linux/x86_64 amazonlinux:2

RUN mkdir -p /workdir/apps
WORKDIR /workdir
COPY install-gapwalk.sh .
COPY install-app.sh .
RUN chmod +x install-gapwalk.sh
RUN chmod +x install-app.sh

# Install Java and AWS CLI v2-y
RUN yum install sudo java-17-amazon-corretto unzip tar -y
RUN sudo yum remove awscli -y
RUN curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
 "awscliv2.zip"
RUN sudo unzip awscliv2.zip
RUN sudo ./aws/install

#·Installation dir
RUN mkdir -p /usr/local/velocity/installation/gapwalk
# Copy PlanetsDemo archive to a dedicated apps dir
COPY PlanetsDemo-v1.zip /workdir/apps/

# Copy resources (tomcat, blu age runtime) to installation dir
COPY tomcat.tar.gz /usr/local/velocity/installation/tomcat.tar.gz
COPY aws-bluage-on-fargate-runtime-4.x.x.tar.gz /usr/local/velocity/installation/
gapwalk/gapwalk-bluage-on-fargate.tar.gz

# run relevant installation scripts
RUN ./install-gapwalk.sh
RUN ./install-app.sh

EXPOSE 8080
EXPOSE 8081
# ...
```

```
# Run Command to start Tomcat server
CMD ["sh", "-c", "sudo /bluage-on-fargate/tomcat.gapwalk/velocity/startup.sh
 $ECS_CONTAINER_METADATA_URI_V4 $AWS_CONTAINER_CREDENTIALS_RELATIVE_URI"]
```

A seguir está o conteúdo de install-gapwalk.sh.

```
#!/bin/sh

# Vars
TEMP_DIR=/bluage-on-fargate/tomcat.gapwalk/temp

# Install
echo "Installing Gapwalk and Tomcat"
sudo rm -rf /bluage-on-fargate
mkdir -p ${TEMP_DIR}
# Copy Blu Age runtime and tomcat archives to temporary extraction dir
sudo cp /usr/local/velocity/installation/gapwalk/gapwalk-bluage-on-fargate.tar.gz
 ${TEMP_DIR}
sudo cp /usr/local/velocity/installation/tomcat.tar.gz ${TEMP_DIR}
#·Create velocity dir
mkdir -p /bluage-on-fargate/tomcat.gapwalk/velocity
#·Extract tomcat files
tar -xvf ${TEMP_DIR}/tomcat.tar.gz -C ${TEMP_DIR}
# Copy all tomcat files to velocity dir
cp -fr ${TEMP_DIR}/apache-tomcat-10.x.x/* /bluage-on-fargate/tomcat.gapwalk/
velocity
# Remove default webapps of Tomcat
rm -f /bluage-on-fargate/tomcat.gapwalk/velocity/webapps/*
# Extract Blu Age runtime at velocity dir
tar -xvf ${TEMP_DIR}/gapwalk-bluage-on-fargate.tar.gz -C /bluage-on-fargate/
tomcat.gapwalk
# Remove temporary extraction dir
sudo rm -rf ${TEMP_DIR}
```

A seguir está o conteúdo de install-app.sh.

```
#!/bin/sh

APP_DIR=/workdir/apps
TOMCAT_GAPWALK_DIR=/bluage-on-fargate/tomcat.gapwalk

unzip ${APP_DIR}/PlanetsDemo-v1.zip -d ${APP_DIR}
```

```
cp -r ${APP_DIR}/webapps/* ${TOMCAT_GAPWALK_DIR}/velocity/webapps/
cp -r ${APP_DIR}/config/* ${TOMCAT_GAPWALK_DIR}/velocity/config/
```

4. Forneça as informações de conexão do banco de dados que você criou como parte dos pré-requisitos no trecho a seguir no `application-main.yml` arquivo, que está localizado na pasta. `{TOMCAT_GAPWALK_DIR}/config` Para obter mais informações, consulte Criação e conexão a um cluster Aurora PostgreSQL DB.

```
datasource:
   jicsDs:
     driver-class-name :
     url:
     username:
     password:
     type :
```

5. Crie e envie a imagem para o seu repositório Amazon ECR. Para obter instruções, consulte Como enviar uma imagem do Docker no Guia do usuário do Amazon Elastic Container Registry.
6. Abra o console em https://console.aws.amazon.com/ecs/v2.
7. No painel de navegação, escolha Definições de tarefa.
8. Em Tipo de lançamento, escolha AWS Fargate.
9. Selecione a função da tarefa que você criou como parte dathe section called “Requisitos de configuração de infraestrutura”.
10. Anexe sua imagem ao contêiner.
11. Conclua o preenchimento do formulário e escolha Criar.
12. No painel de navegação esquerdo, escolha Clusters e, em seguida, escolha seu cluster na lista.
13. Na página de detalhes do seu cluster, na guia Serviços, escolha Criar.
14. Selecione a definição da tarefa.
15. Expanda a seção Rede e configure a VPC, as sub-redes e o grupo de segurança que você criou como parte. the section called “Requisitos de configuração de infraestrutura”
16. Implante seu serviço Amazon ECS.

Se a implantação falhar, verifique os registros. Para encontrá-los, acesse a página de tarefas no Amazon ECS gerenciada por e AWS Fargate, em seguida, escolha a guia Logs. Se você encontrar códigos de erro que começam com C seguido por um número, como CXXXX, anote as mensagens de erro. Por exemplo, o código de erro C5102 é um erro comum que indica uma configuração

incorreta da infraestrutura. Você também pode navegar dentro da sua tarefa em execução e executar alguns comandos, semelhantes ao AWS Blu Age Runtime (no Amazon EC2). Para obter mais informações, consulte Como usar o Amazon ECS Exec para depuração no Amazon Elastic Container Service Developer Guide.

Para abrir um shell interativo, execute o comando a seguir em sua máquina local.

```
aws ecs execute-command --cluster your_cluster_name --container your_container_name --
task task_id --interactive --command /bin/sh
```

Teste o aplicativo implantado

Para obter um exemplo de como testar o PlanetsDemo aplicativo, consultethe section called “Teste o PlanetsDemo aplicativo”.

Atualizando o AWS Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate

Este guia descreve como atualizar o AWS Blu Age Runtime no Amazon ECS gerenciado por. AWS Fargate

Tópicos

- Pré-requisitos
- Atualize o AWS Blu Age Runtime

Pré-requisitos

Antes de começar, verifique se você atende aos seguintes pré-requisitos.

- Preencha the section called “AWS Pré-requisitos do Blu Age Runtime”, e the section called “AWS Integração do Blu Age Runtime”.
- Baixe a versão do AWS Blu Age Runtime para a qual você deseja atualizar. Para ter mais informações, consulte the section called “AWS Integração do Blu Age Runtime”. A estrutura consiste em dois arquivos binários: `aws-bluage-runtime-x.x.x.x.tar.gz` `aws-bluage-webapps-x.x.x.x.tar.gz` e.

Atualize o AWS Blu Age Runtime

Conclua as etapas a seguir para atualizar o AWS Blu Age Runtime.

1. Reconstrua sua imagem do Docker com a versão desejada do AWS Blu Age Runtime. Para obter instruções, consulte the section called “Configurando o AWS Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate”.
2. Envie sua imagem do Docker para o seu repositório Amazon ECR.
3. Pare e reinicie seu serviço Amazon ECS.
4. Verificar os logs.

O AWS Blu Age Runtime foi atualizado com sucesso.

Amazon CloudWatch Alarms for AWS Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate

Para ter notificações mais visíveis sempre que seus aplicativos implantados encontrarem exceções, configure CloudWatch para receber o registro do aplicativo e adicione um alarme para avisá-lo sobre possíveis erros.

Configuração de alarme

Com CloudWatch os registros, você pode configurar qualquer número de métricas e alarmes, dependendo do seu aplicativo e das suas necessidades.

Especificamente, você pode configurar alarmes proativos para alertas de uso diretamente durante a criação do cluster do Amazon ECS, para que você seja notificado quando ocorrerem erros. Para destacar erros na conexão com o sistema de controle AWS Blu Age, adicione uma métrica referente à string “Erro C” nos registros. Depois, você poderá definir um alarme que reaja a essa métrica.

Configurando dependências licenciadas no AWS Blu Age Runtime no Amazon ECS gerenciado por AWS Fargate

Este tópico descreve como configurar dependências licenciadas adicionais que você pode usar com o AWS Blu Age Runtime no Amazon ECS gerenciado pelo. AWS Fargate

Tópicos

- Pré-requisitos
- Visão geral

Pré-requisitos

Antes de começar, certifique-se de que você concluiu os seguintes pré-requisitos.

- Preencha the section called “AWS Pré-requisitos do Blu Age Runtime”, e the section called “AWS Integração do Blu Age Runtime”.
- Obtenha as seguintes dependências de sua fonte.

Banco de dados Oracle

Forneça um driver de banco de dados Oracle. Por exemplo, ojdbc11-23.3.0.23.09.jar.

Conexão IBM MQ

Forneça um cliente IBM MQ. Por exemplo, com.ibm.mq.jakarta.client-9.3.4.1.jar.

Com essa versão de dependência, forneça também as seguintes dependências transitivas:

- bcprov-jdk15to18-1.76.jar
- bcpkix-jdk15to18-1.76.jar
- bcutil-jdk15to18-1.76.jar

Arquivos de impressora DDS

Forneça a biblioteca de relatórios do Jasper. Por exemplo, jasperreports-6.16.0.jar, mas uma versão mais recente pode ser compatível.

Com essa versão de dependência, forneça também as seguintes dependências transitivas:

- castor-core-1.4.1.jar
- castor-xml-1.4.1.jar
- commons-digester-2.1.jar
- ecj-3.21.0.jar
- itext-2.1.7.js8.jar
- javax.inject-1.jar
- jcommon-1.0.23.jar
- jfreechart-1.0.19.jar

- commons-beanutils-1.9.4.jar
- commons-collections-3.2.2.jar

Visão geral

Para instalar as dependências, conclua as etapas a seguir.

1. Copie qualquer uma das dependências acima, conforme necessário, para sua pasta de criação de imagens do Docker.
2. Se seu banco de dados JICS ou Blusam estiver hospedado no Oracle, forneça o driver do banco de dados Oracle. *your-tomcat-path*/extra
3. Em seu Dockerfile, copie essas dependências para. *your-tomcat-path*/extra
4. Crie sua imagem do Docker e, em seguida, envie-a para o Amazon ECR.
5. Pare e reinicie seu serviço Amazon ECS.
6. Verificar os logs.

AWS Implantação do Blu Age Runtime no Amazon EC2

Os tópicos desta seção descrevem como configurar o AWS Blu Age Runtime (não gerenciado) no Amazon EC2, como atualizar a versão do tempo de execução, como monitorar sua implantação usando alarmes da CloudWatch Amazon e como adicionar dependências licenciadas. Essas instruções são aplicáveis quando você cria instâncias do Amazon EC2, bem como ao usar o Amazon ECS no Amazon EC2 ou o Amazon EKS no Amazon EC2.

Tópicos

Configurando o AWS Blu Age Runtime (não gerenciado) no Amazon EC2

Este tópico explica como configurar e implantar o aplicativo de PlanetsDemo amostra usando o AWS Blu Age Runtime (não gerenciado) no Amazon EC2.

Tópicos

Pré-requisitos

Antes de começar, certifique-se de que você concluiu os seguintes pré-requisitos.

- Configure o AWS CLI seguindo as etapas em Configuração da AWS CLI.
- Preencha the section called “AWS Pré-requisitos do Blu Age Runtime”, e the section called “AWS Integração do Blu Age Runtime”.
- Crie uma instância do Amazon EC2 usando um dos tipos de instância compatíveis. Para obter mais informações, consulte Introdução às instâncias do Amazon EC2 Linux.
- Certifique-se de que você possa se conectar à instância do Amazon EC2 com sucesso, por exemplo, usando SSM.
- Baixe e extraia o AWS Blu Age Runtime (no Amazon EC2) em. *`your-tomcat-path`*`/*` Certifique-se de colocar o `bluage.bin` arquivo exatamente no local especificado pela variável de `CATALINA_HOME` ambiente descrita em CATALINA_HOME e CATALINA_BASE na documentação do Apache Tomcat. Para obter instruções sobre como recuperar o AWS Blu Age Runtime, consulte. the section called “AWS Integração do Blu Age Runtime”
- Baixe o arquivo do PlanetsDemo aplicativo.
- Descompacte o arquivo e faça o upload da aplicação para um bucket do Amazon S3 de sua escolha.
- Crie um banco de dados Amazon Aurora PostgreSQL para JICS e execute a consulta nele. `PlanetsDemo-v1/jics/sql/initJics.sql` Para obter informações sobre como criar um banco de dados Amazon Aurora PostgreSQL, consulte Criação e conexão com um cluster de banco de dados Aurora PostgreSQL.

Configuração

Para configurar o aplicativo de PlanetsDemo amostra, conclua as etapas a seguir.

1. Conecte-se à sua instância do Amazon EC2 e vá até a `conf` pasta abaixo da pasta de instalação do Apache Tomcat 10. Abra o `catalina.properties` arquivo para edição e substitua a linha que começa `common.loader` com a seguinte linha.

```
common.loader="${catalina.base}/lib","${catalina.base}/lib/
*.jar","${catalina.home}/lib","${catalina.home}/lib/*.jar","${catalina.home}/
shared","${catalina.home}/shared/*.jar","${catalina.home}/extra","${catalina.home}/
extra/*.jar"
```

2. Navegue para a pasta *<your-tomcat-path>*`/webapps`.
3. Copie os PlanetsDemo binários disponíveis na PlanetsDemo pasta -v1/webapps/ do bucket do Amazon S3 usando o comando a seguir.

```
aws s3 cp s3://path-to-demo-app-webapps/ . --recursive
```

> **Note**
>
> `path-to-demo-app-webapps`Substitua pelo URI correto do Amazon S3 para o bucket em que você descompactou o arquivo anteriormente. PlanetsDemo

4. Copie o conteúdo da pasta `PlanetsDemo-v1/config/` para *<your-tomcat-path>*`/config/`.
5. Forneça as informações de conexão do banco de dados que você criou como parte dos pré-requisitos no trecho a seguir no arquivo. `application-main.yml` Para obter mais informações, consulte Criação e conexão a um cluster Aurora PostgreSQL DB.

```
datasource:
   jicsDs:
     driver-class-name :
     url:
     username:
     password:
     type :
```

6. Inicie seu servidor Apache Tomcat e verifique os registros.

```
your-tomcat-path/startup.sh

tail -f your-tomcat-path/logs/catalina.log
```

Se você encontrar códigos de erro que começam com C seguido por um número, como CXXXX, anote as mensagens de erro. Por exemplo, o código de erro C5102 é um erro comum que indica uma configuração incorreta da infraestrutura.

Teste o aplicativo implantado

Para obter um exemplo de como testar o PlanetsDemo aplicativo, consultethe section called “Teste o PlanetsDemo aplicativo”.

Usando contêineres no Amazon EC2 para Amazon ECS e Amazon EKS

Este tópico explica como configurar e implantar o aplicativo de PlanetsDemo amostra usando o AWS Blu Age Runtime (não gerenciado) no Amazon EC2 como um contêiner.

Tópicos

- Pré-requisitos
- Configuração
- Teste o aplicativo implantado

Pré-requisitos

Antes de começar, certifique-se de que você concluiu os seguintes pré-requisitos.

- Configure o AWS CLI seguindo as etapas em Configuração da AWS CLI.
- Preencha the section called “AWS Pré-requisitos do Blu Age Runtime”, e the section called “AWS Integração do Blu Age Runtime”.
- Baixe o AWS Blu Age Runtime (no Amazon EC2). Para obter instruções sobre como recuperar o tempo de execução, consultethe section called “AWS Integração do Blu Age Runtime”.
- Baixe o arquivo do PlanetsDemo aplicativo.
- Crie um banco de dados Amazon Aurora PostgreSQL para JICS e execute a consulta nele. `PlanetsDemo-v1/jics/sql/initJics.sql` Para obter informações sobre como criar um banco de dados Amazon Aurora PostgreSQL, consulte Criação e conexão com um cluster de banco de dados Aurora PostgreSQL.

Configuração

Para configurar o aplicativo de PlanetsDemo amostra, conclua as etapas a seguir.

1. Prepare um Dockerfile para criar sua imagem personalizada com base nos binários de tempo de execução fornecidos e nos binários do servidor Apache Tomcat. Veja o exemplo a seguir do Dockerfile. O objetivo é instalar o Apache Tomcat 10, seguido pelo AWS Blu Age Runtime (no Amazon EC2) extraído na raiz do diretório de instalação do Apache Tomcat 10 e, em seguida, instalar o exemplo de aplicativo modernizado chamado. PlanetsDemo Os `install-app.sh` scripts `install-gapwalk.sh` e usados neste exemplo do Dockerfile são listados após o Dockerfile.

```
FROM --platform=linux/x86_64 amazonlinux:2

RUN mkdir -p /workdir/apps
WORKDIR /workdir
COPY install-gapwalk.sh .
COPY install-app.sh .
RUN chmod +x install-gapwalk.sh
RUN chmod +x install-app.sh

# Install Java and AWS CLI v2-y
RUN yum install sudo java-17-amazon-corretto unzip tar -y
RUN sudo yum remove awscli -y
RUN curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
 "awscliv2.zip"
RUN sudo unzip awscliv2.zip
RUN sudo ./aws/install

#·Installation dir
RUN mkdir -p /usr/local/velocity/installation/gapwalk

# Copy PlanetsDemo archive to a dedicated apps dir
COPY PlanetsDemo-v1.zip /workdir/apps/

# Copy resources (tomcat, blu age runtime) to installation dir
COPY tomcat.tar.gz /usr/local/velocity/installation/tomcat.tar.gz
COPY aws-bluage-runtime-4.x.x.tar.gz /usr/local/velocity/installation/gapwalk/
gapwalk.tar.gz

# run relevant installation scripts
RUN ./install-gapwalk.sh
```

```
RUN ./install-app.sh

EXPOSE 8080
EXPOSE 8081
# ...

WORKDIR /bluage/tomcat.gapwalk/velocity
# Run Command to start Tomcat server
CMD ["sh", "-c", "sudo bin/catalina.sh run"]
```

A seguir está o conteúdo de`install-gapwalk.sh`.

```
#!/bin/sh

# Vars
TEMP_DIR=/bluage/tomcat.gapwalk/temp

# Install
echo "Installing Gapwalk and Tomcat"
sudo rm -rf /bluage
mkdir -p ${TEMP_DIR}
# Copy Blu Age runtime and tomcat archives to temporary extraction dir
sudo cp /usr/local/velocity/installation/gapwalk/gapwalk-bluage.tar.gz ${TEMP_DIR}
sudo cp /usr/local/velocity/installation/tomcat.tar.gz ${TEMP_DIR}
# Create velocity dir
mkdir -p /bluage/tomcat.gapwalk/velocity
# Extract tomcat files
tar -xvf ${TEMP_DIR}/tomcat.tar.gz -C ${TEMP_DIR}
# Copy all tomcat files to velocity dir
cp -fr ${TEMP_DIR}/apache-tomcat-10.x.x/* /bluage/tomcat.gapwalk/velocity
# Remove default webapps of Tomcat
rm -f /bluage/tomcat.gapwalk/velocity/webapps/*
# Extract Blu Age runtime at velocity dir
tar -xvf ${TEMP_DIR}/gapwalk-bluage.tar.gz -C /bluage/tomcat.gapwalk
# Remove temporary extraction dir
sudo rm -rf ${TEMP_DIR}
```

A seguir está o conteúdo de`install-app.sh`.

```
#!/bin/sh

APP_DIR=/workdir/apps
```

```
TOMCAT_GAPWALK_DIR=/bluage/tomcat.gapwalk

unzip ${APP_DIR}/PlanetsDemo-v1.zip -d ${APP_DIR}
cp -r ${APP_DIR}/webapps/* ${TOMCAT_GAPWALK_DIR}/velocity/webapps/
cp -r ${APP_DIR}/config/* ${TOMCAT_GAPWALK_DIR}/velocity/config/
```

2. Forneça as informações de conexão do banco de dados que você criou como parte dos pré-requisitos no trecho a seguir no `application-main.yml` arquivo, localizado na pasta. `{TOMCAT_GAPWALK_DIR}/config` Para obter mais informações, consulte Criação e conexão a um cluster Aurora PostgreSQL DB.

```
datasource:
   jicsDs:
     driver-class-name :
     url:
     username:
     password:
     type :
```

3. Crie e envie a imagem para o seu repositório Amazon ECR. Para obter instruções, consulte Como enviar uma imagem do Docker no Guia do usuário do Amazon Elastic Container Registry. Então, dependendo da sua situação, crie um pod do Amazon EKS ou uma definição de tarefa do Amazon ECS usando sua imagem do Amazon ECR e implante-a em seu cluster. Por exemplo, consulte Criação de uma definição de tarefa usando o console no Guia do desenvolvedor do Amazon Elastic Container Service e Implantar um aplicativo de amostra no Guia do usuário do Amazon EKS.

Teste o aplicativo implantado

Para obter um exemplo de como testar o PlanetsDemo aplicativo, consultethe section called “Teste o PlanetsDemo aplicativo”.

Atualizando o AWS Blu Age Runtime no Amazon EC2

Este guia descreve como atualizar o AWS Blu Age Runtime no Amazon EC2.

Tópicos

- Pré-requisitos
- Atualize o AWS Blu Age Runtime na instância do Amazon EC2
- Atualize o AWS Blu Age Runtime em um contêiner

Pré-requisitos

Antes de começar, verifique se você atende aos seguintes pré-requisitos.

- Para verificar se há instruções específicas para sua versão, consultethe section called “Atualizando o AWS Blu Age”.
- Preencha the section called “AWS Pré-requisitos do Blu Age Runtime”, e the section called “AWS Integração do Blu Age Runtime”.
- Certifique-se de ter uma instância do Amazon EC2 que contenha a versão mais recente do AWS Blu Age Runtime. Para obter mais informações, consulte Introdução às instâncias do Amazon EC2 Linux.
- Certifique-se de que você possa se conectar à instância do Amazon EC2 com sucesso, por exemplo, usando SSM.
- Baixe a versão do AWS Blu Age Runtime para a qual você deseja atualizar. Para obter mais informações, consulte the section called “AWS Configuração do Blu Age Runtime (não gerenciada)” A estrutura consiste em dois arquivos binários: `aws-bluage-runtime-x.x.x.x.tar.gz` `aws-bluage-webapps-x.x.x.x.tar.gz` e.

Atualize o AWS Blu Age Runtime na instância do Amazon EC2

Conclua as etapas a seguir para atualizar o AWS Blu Age Runtime.

1. Conecte-se à sua instância do Amazon EC2 e altere o usuário para su executando o comando a seguir.

   ```
   sudo su
   ```

   Você precisa do privilégio de superusuário para executar comandos neste tutorial.
2. Crie duas pastas, uma para cada arquivo binário.
3. Nomeie cada pasta usando o mesmo nome que o arquivo binário.
4. Copie cada arquivo binário para a pasta correspondente.

> **Warning**
>
> A extração de cada binário produz uma pasta com o mesmo nome. Portanto, se você extrair os dois arquivos binários no mesmo local, um após o outro, substituirá o conteúdo.

5. Para extrair os binários, use os seguintes comandos. Execute os comandos em cada pasta.

```
tar xvf aws-bluage-runtime-x.x.x.x.tar.gz
tar xvf aws-bluage-webapps-x.x.x.x.tar.gz
```

6. Pare os serviços do Apache Tomcat usando os seguintes comandos.

```
systemctl stop tomcat.service
systemctl stop tomcat-webapps.service
```

7. Substitua o conteúdo de `<your-tomcat-path>/shared/` pelo conteúdo de `aws-bluage-runtime-x.x.x.x/velocity/shared/`.
8. Substitua `<your-tomcat-path>/webapps/gapwalk-application.war` pelo `aws-bluage-runtime-x.x.x.x/velocity/webapps/gapwalk-application.war`.
9. Substitua os arquivos war em `<your-tomcat-path>/webapps/`, ou seja `jac.war` e `bac.war`, pelos mesmos arquivos de `aws-bluage-webapps-x.x.x.x/velocity/webapps/`.
10. Inicie os serviços do Apache Tomcat executando os seguintes comandos.

```
systemctl start tomcat.service
systemctl start tomcat-webapps.service
```

11. Verificar os logs.

Para verificar o status da aplicação implantada, execute os seguintes comandos.

```
curl http://localhost:8080/gapwalk-application/
```

A seguinte mensagem deve ser exibida.

```
Jics application is running
```

```
curl http://localhost:8181/jac/api/services/rest/jicsservice/
```

A seguinte mensagem deve ser exibida.

```
Jics application is running
```

```
curl http://localhost:8181/bac/api/services/rest/bluesamserver/serverIsUp
```

A resposta deve estar vazia.

O tempo de execução do AWS Blu Age foi atualizado com sucesso.

Atualize o AWS Blu Age Runtime em um contêiner

Conclua as etapas a seguir para atualizar o AWS Blu Age Runtime.

1. Reconstrua sua imagem do Docker com a versão desejada do AWS Blu Age Runtime. Para obter instruções, consulte the section called “Configurando o AWS Blu Age Runtime (não gerenciado) no Amazon EC2”.
2. Envie sua imagem do Docker para o seu repositório Amazon ECR.
3. Pare e reinicie seu serviço Amazon ECS ou Amazon EKS.
4. Verificar os logs.

O AWS Blu Age Runtime foi atualizado com sucesso.

AWS Blu Age Runtime (no Amazon EC2) Amazon Alarms CloudWatch

Para que você tenha notificações mais visíveis quando seus aplicativos implantados encontrarem exceções que colocarão seu aplicativo em um período de carência, você pode configurar CloudWatch o recebimento do registro do aplicativo e adicionar um alarme para avisá-lo sobre possíveis erros.

Implantação do CloudWatch registro

Por padrão, o arquivo `application-main.yml` inclui uma referência a outro arquivo de configuração de log chamado `logback-cloudwatch.yml`.

```
logging:
```

```
  config: classpath:logback-cloudwatch.xml
```

Ambos os arquivos estão na pasta config e é assim que o CloudWatch registro é configurado, conforme explicado nas seções a seguir.

Configuração do CloudWatch registro

O arquivo `logback-cloudwatch.xml` tem o seguinte conteúdo.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration>
<configuration>

    <appender name="console" class="ch.qos.logback.core.ConsoleAppender">
        <encoder>
            <pattern>%date{yyyy-MM-dd HH:mm:ss.SSS,UTC}  %level --- [%thread{15}]
 %logger{40} : %msg%n%xThrowable</pattern>
        </encoder>
    </appender>

    <appender name="cloudwatch"
 class="com.netfective.bluage.runtime.cloudwatchlogger.CloudWatchAppender">
        <logGroup>BluAgeRuntimeOnEC2-Logs</logGroup>
        <logStream>%date{yyyy-MM-dd,UTC}.%instanceId.%uuid</logStream>
        <layout>
            <pattern>%date{yyyy-MM-dd HH:mm:ss.SSS,UTC}  %level --- [%thread{15}]
 %logger{40} : %msg%n%xThrowable</pattern>
        </layout>
        <appender-ref ref="console" />
    </appender>

    <root level="INFO">
        <appender-ref ref="cloudwatch" />
    </root>
</configuration>
```

Tudo fora do <appender name="cloudwatch"/> elemento é uma configuração padrão de logback. Há dois anexadores neste arquivo: um anexador de console para enviar registros para o console e um CloudWatch anexador para o qual enviar registros. CloudWatch

O atributo `level` no elemento `root` especifica o nível de log de toda a aplicação.

Os valores necessários dentro da tag <appender name="cloudwatch"/> são:

- <logGroup/>:Define o nome do grupo de registros em CloudWatch. Se o valor não for especificado, o padrão será `BluAgeRuntimeOnEC2-Logs`. Se o grupo de logs não existir, ele será criado automaticamente. Esse comportamento pode ser alterado por meio da configuração, que será abordada a seguir.
- <logStream/>: define o nome do LogStream (dentro do grupo de registros) em. CloudWatch

Valores opcionais:

- <region/>: Substitui a região na qual o fluxo de logs será gravado. Por padrão, os logs vão para a mesma região que a instância do EC2.
- <layout/>: o padrão que as mensagens de log usarão.
- <maxbatchsize/>: o número máximo de mensagens de registro a serem enviadas CloudWatch por operação.
- <maxbatchtimemillis/>: o tempo em milissegundos para permitir que CloudWatch os registros sejam gravados.
- <maxqueuewaittimemillis/>: o tempo em milissegundos para tentar inserir solicitações na fila de logs interna.
- <internalqueuesize/>: o tamanho máximo da fila interna.
- <createlogdests/>: crie um grupo de logs e um fluxo de logs, se eles não existirem.
- <initialwaittimemillis/>: a quantidade de tempo em que você deseja que o thread permaneça suspenso na inicialização. Essa espera inicial permite um acúmulo inicial de logs.
- <maxeventmessagesize/>: o tamanho máximo de um evento de logs. Os logs que excederem esse tamanho não serão enviados.
- <truncateeventmessages/>: trunque as mensagens que são muito longas.
- <printrejectedevents/>: Ative o anexador de emergência.

CloudWatch configuração

Para que a configuração acima envie os registros corretamente para CloudWatch, atualize sua função de perfil de instância IAM do Amazon EC2 para conceder permissões adicionais para o grupo de logs `BluAgeRuntimeOnEC2-Logs` e seus fluxos de log:

- `logs:CreateLogStream`
- `logs:DescribeLogStreams`

- `logs:CreateLogGroup`
- `logs:PutLogEvents`
- `logs:DescribeLogGroups`

Configuração de alarme

Graças aos CloudWatch registros, você pode configurar diferentes métricas e alarmes, dependendo do seu aplicativo e de suas necessidades. Especificamente, você pode configurar alarmes proativos para alertas de uso, para ser avisado no caso de erros que possam colocar a aplicação em um período de carência (e, no final, impedir que ele funcione). Para fazer isso, você pode adicionar uma métrica relacionada à string “Erro C5001" nos registros, que destaca os erros na conexão com o sistema de controle AWS Blu Age. Depois, você poderá definir um alarme que reaja a essa métrica.

Configurando dependências licenciadas no AWS Blu Age Runtime no Amazon EC2

Este guia descreve como configurar dependências licenciadas adicionais que você pode usar com o AWS Blu Age Runtime no Amazon EC2.

Tópicos

Pré-requisitos

Antes de começar, certifique-se de que você concluiu os seguintes pré-requisitos.

- Preencha the section called “AWS Pré-requisitos do Blu Age Runtime”, e the section called “AWS Integração do Blu Age Runtime”.
- Certifique-se de ter uma instância do Amazon EC2 contendo a versão mais recente do AWS Blu Age Runtime (no Amazon EC2). Para obter mais informações, consulte Introdução às instâncias do Amazon EC2 Linux.
- Certifique-se de que você possa se conectar à instância do Amazon EC2 com sucesso, por exemplo, usando SSM.
- Obtenha as seguintes dependências de suas fontes.

Banco de dados Oracle

Forneça um driver de banco de dados Oracle. Testamos a funcionalidade do AWS Blu Age Runtime (no Amazon EC2) com a versão ojdbc11-23.3.0.23.09.jar, mas uma versão mais recente pode ser compatível.

Conexão IBM MQ

Forneça um cliente IBM MQ. Testamos a funcionalidade do AWS Blu Age Runtime (no Amazon EC2) com a versão com.ibm.mq.jakarta.client-9.3.4.1.jar, mas uma versão mais recente pode ser compatível.

Com essa versão de dependência, forneça também as seguintes dependências transitivas:

- bcprov-jdk15to18-1.76.jar
- bcpkix-jdk15to18-1.76.jar
- bcutil-jdk15to18-1.76.jar

Arquivos de impressora DDS

Forneça a biblioteca de relatórios do Jasper. Testamos a funcionalidade do AWS Blu Age Runtime (no Amazon EC2) com jasperreports-6.16.0.jar, mas uma versão mais recente pode ser compatível.

Com essa versão de dependência, forneça também as seguintes dependências transitivas:

- castor-core-1.4.1.jar
- castor-xml-1.4.1.jar
- commons-digester-2.1.jar
- ecj-3.21.0.jar
- itext-2.1.7.js8.jar
- javax.inject-1.jar
- jcommon-1.0.23.jar
- jfreechart-1.0.19.jar
- commons-beanutils-1.9.4.jar
- commons-collections-3.2.2.jar

Visão geral

Para instalar as dependências, conclua as etapas a seguir.

1. Conecte-se à sua instância do Amazon EC2 e altere o usuário para su executando o comando a seguir.

   ```
   sudo su
   ```

   Você precisa do privilégio de superusuário para executar comandos neste tutorial.
2. Navegue para a pasta <your-tomcat-path>/extra/.

   ```
   cd <your-tomcat-path>/extra/
   ```
3. Copie qualquer uma das dependências acima conforme necessário nesta pasta.
4. Pare e inicie o tomcat.service executando os seguintes comandos.

   ```
   systemctl stop tomcat.service
   ```

   ```
   systemctl start tomcat.service
   ```
5. Verifique o status do serviço para se certificar de que ele está sendo executado.

   ```
   systemctl status tomcat.service
   ```
6. Verificar os logs.

Configurar as dependências para aplicações web JAC e BAC

1. Se o seu banco de dados JICS ou Blusam estiver hospedado no Oracle, será necessário fornecer o driver do banco de dados Oracle em <your-tomcat-path>/extra.
2. Crie a pasta se ela ainda não estiver presente.
3. Pare e reinicie seu servidor Apache Tomcat.
4. Verificar os logs.

Teste o PlanetsDemo aplicativo

Para verificar o status do PlanetsDemo aplicativo implantado, execute os comandos a seguir depois de substituir `load-balancer-DNS-namelistener-port`, e `web-binary-name` com os valores corretos para sua configuração.

```
curl http://load-balancer-DNS-name:listener-port/gapwalk-application/
```

Se o aplicativo estiver em execução, você verá a seguinte mensagem de saída:`Jics application is running`.

Em seguida, execute o comando a seguir.

```
curl http://load-balancer-DNS-name:listener-port/jac/api/services/rest/jicsservice/
```

Se o aplicativo estiver em execução, você verá a seguinte mensagem de saída:`Jics application is running`.

```
Jics application is running
```

Se você configurou o Blusam, pode esperar uma resposta vazia ao executar o comando a seguir.

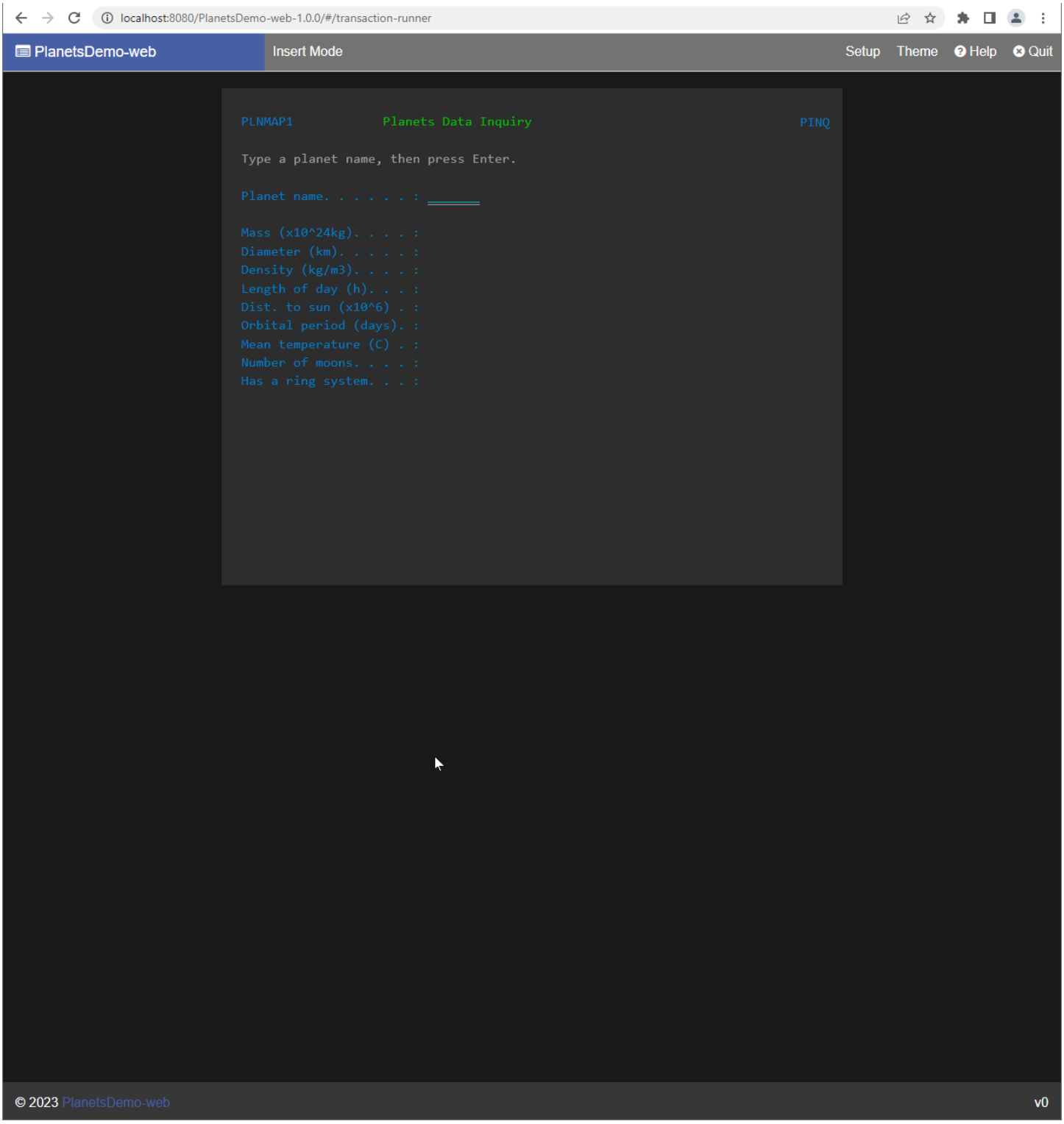
```
curl http://load-balancer-DNS-name:listener-port/bac/api/services/rest/bluesamserver/
serverIsUp
```

Observe o nome do binário da web (PlanetsDemo-web-1.0.0, se inalterado). Para acessar o PlanetsDemo aplicativo, use um URL com o seguinte formato.

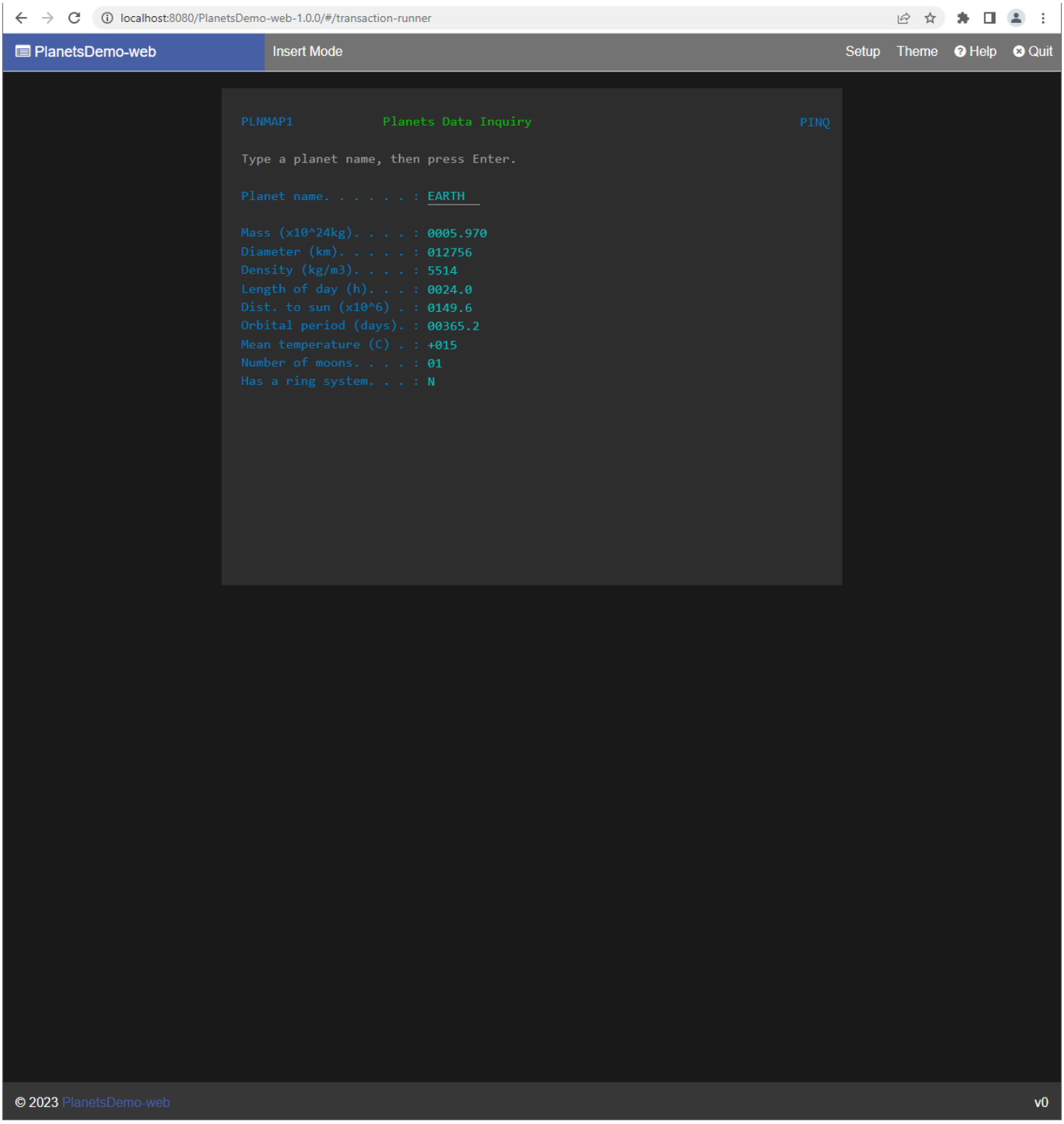
```
https://load-balancer-DNS-name:listener-port/web-binary-name
```

Depois que o PlanetsDemo aplicativo for iniciado, a página inicial será exibida.

Digite PINQ na caixa de texto e pressione Enter. A página de consulta de dados é exibida.



Por exemplo, insira EARTH no campo do PlanetsDemo nome e pressione Enter. A página do planeta que você inseriu é exibida.



Modifique o código-fonte com o Blu Age Developer IDE

Se você estiver usando o mecanismo de tempo AWS de execução AWS Blu Age gerenciado, poderá usar o Blu Age Developer para modificar o código-fonte gerado. Talvez você queira fazer

isso se precisar atualizar o código modernizado por algum motivo ou se uma parte do código-fonte antigo não puder ser modernizada. Você acessa o Blu Age Developer por meio da Amazon AppStream 2.0. Esta seção descreve como configurar o Blu Age Developer na AppStream versão 2.0. Também explica como usar o Blu Age Developer para atualizar o código-fonte usando o aplicativo PlanetsDemo de amostra.

Tópicos

Tutorial: Configurar AppStream 2.0 para AWS Blu Age Developer IDE

AWS A modernização do mainframe fornece várias ferramentas por meio do Amazon AppStream 2.0. AppStream 2.0 é um serviço de streaming de aplicativos totalmente gerenciado e seguro que permite transmitir aplicativos de desktop para usuários sem reescrever aplicativos. AppStream O 2.0 fornece aos usuários acesso instantâneo aos aplicativos de que precisam, com uma experiência de usuário responsiva e fluida no dispositivo de sua escolha. O uso da AppStream versão 2.0 para hospedar ferramentas específicas do Runtime Engine oferece às equipes de aplicativos do cliente a capacidade de usar as ferramentas diretamente de seus navegadores da web, interagindo com arquivos de aplicativos armazenados em buckets ou repositórios do Amazon S3. CodeCommit

Para obter informações sobre o suporte a navegadores na AppStream versão 2.0, consulte System Requirements and Feature Support (Web Browser) no Amazon AppStream 2.0 Administration Guide. Se você tiver problemas ao usar a AppStream versão 2.0, consulte Solução de problemas do usuário AppStream 2.0 no Guia de administração da Amazon AppStream 2.0.

Este documento descreve como configurar o AWS Blu Age Developer IDE em uma frota AppStream 2.0.

Tópicos

- Etapa 6: acessar uma instância
- Limpar os recursos

Pré-requisito

Baixe o arquivo que contém os artefatos necessários para configurar o AWS Blu Age Developer IDE em AppStream 2.0.

> **Note**
>
> Esse é um arquivo grande. Se você tiver problemas com o tempo limite da operação, recomendamos usar uma instância do Amazon EC2 para melhorar o desempenho de upload e download.

Etapa 1: Crie um bucket do Amazon S3

Crie um bucket do Amazon S3 da Região da AWS mesma forma que a frota AppStream 2.0 que você criará. Esse bucket conterá os artefatos necessários para você concluir este tutorial.

Etapa 2: anexar uma política ao bucket S3

Anexe a política a seguir ao bucket que você criou para este tutorial. Certifique-se de `MYBUCKET` substituir pelo nome real do bucket que você criou.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "AllowAppStream2.0ToRetrieveObjects",
        "Effect": "Allow",
        "Principal": {
            "Service": "appstream.amazonaws.com"
        },
        "Action": "s3:GetObject",
        "Resource": "arn:aws:s3:::MYBUCKET/*"
    }]
}
```

Etapa 3: fazer upload de arquivos para o bucket do Amazon S3

Descompacte os arquivos que você baixou no Pré-requisito e faça o upload da pasta `appstream` no seu bucket. O upload dessa pasta cria a estrutura correta no seu bucket. Para obter mais informações, consulte Upload de objetos no Guia do usuário do Amazon S3.

Etapa 4: baixar AWS CloudFormation modelos

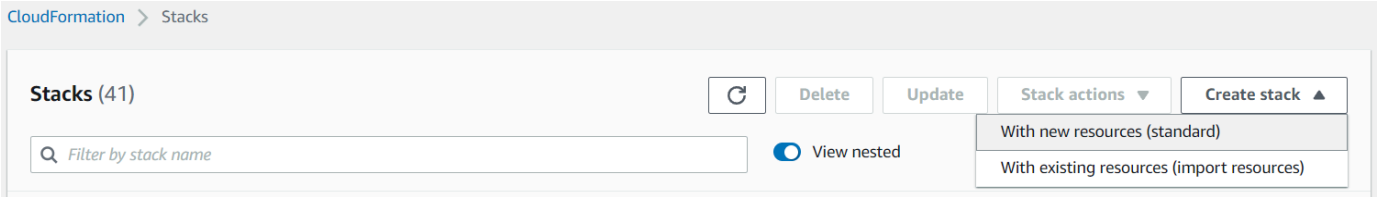
Baixe os AWS CloudFormation modelos a seguir. Você precisa desses modelos para criar e preencher a frota AppStream 2.0.

- cfn-m2- .yaml appstream-elastic-fleet-linux
- cfn-m2- -linux.yaml appstream-bluage-dev-tools
- cfn-m2- .yaml appstream-bluage-shared-linux
- cfn-m2- .yaml appstream-chrome-linux
- cfn-m2- .yaml appstream-eclipse-jee-linux
- cfn-m2- .yaml appstream-pgadmin-linux

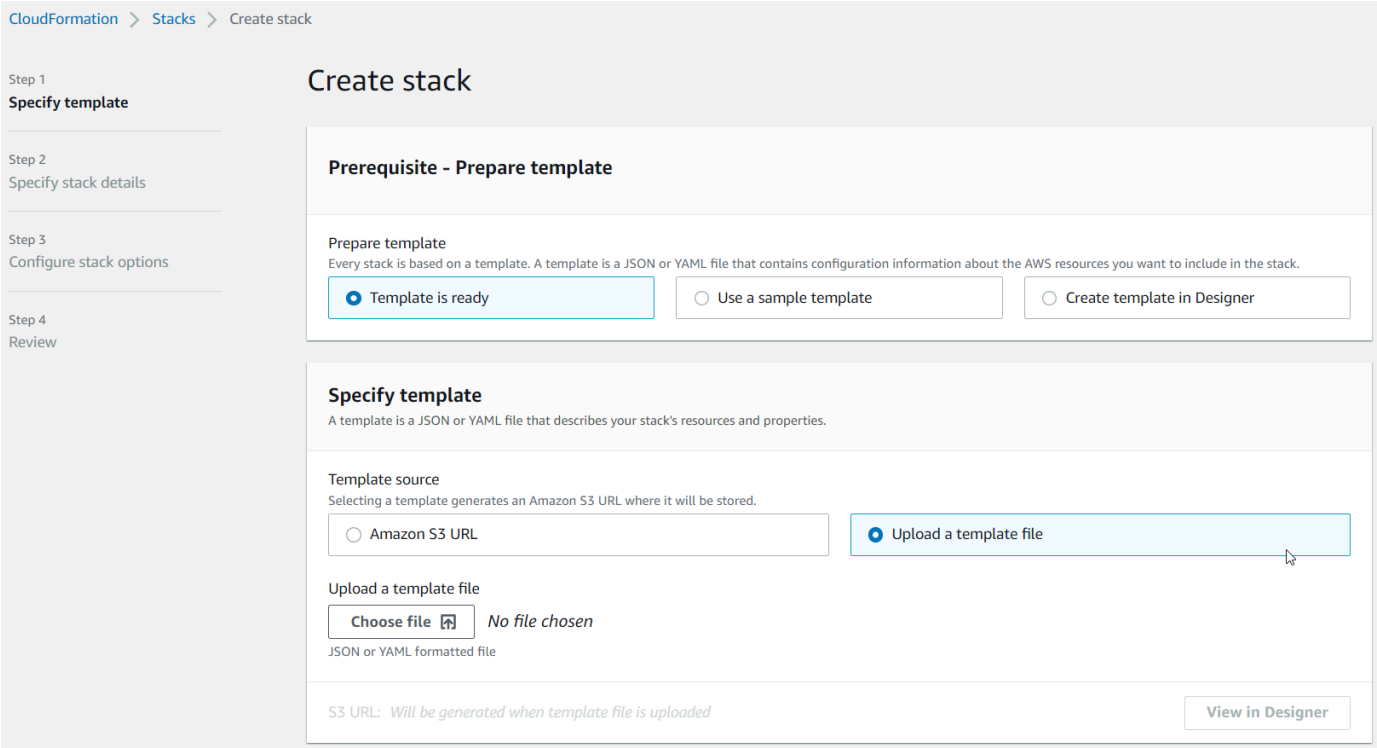
Etapa 5: Crie a frota com AWS CloudFormation

Nesta etapa, você usa o `cfn-m2-appstream-elastic-fleet-linux.yaml` AWS CloudFormation modelo para criar uma frota AppStream 2.0 e uma pilha para hospedar o AWS Blu Age Developer IDE. Depois de criar a frota e a pilha, você executará os outros AWS CloudFormation modelos baixados na etapa anterior para instalar o Developer IDE e outras ferramentas necessárias.

1. Navegue até AWS CloudFormation o console AWS de gerenciamento e escolha Pilhas.
2. Em Pilhas, selecione Criar pilha e Com novos recursos (padrão):



3. Em Criar pilha, escolha O modelo está pronto e fazer o upload de um arquivo de modelo:



4. Escolha Escolher arquivo e navegue até o arquivo `cfn-m2-appstream-elastic-fleet-linux.yaml`. Escolha Próximo.
5. Na página Especificar detalhes da pilha, forneça as seguintes informações.

   - Um nome para a pilha.
   - Seu grupo de segurança padrão e duas sub-redes desse grupo de segurança.

   > **Note**
   >
   > As duas sub-redes do grupo de segurança devem estar em zonas de disponibilidade diferentes.

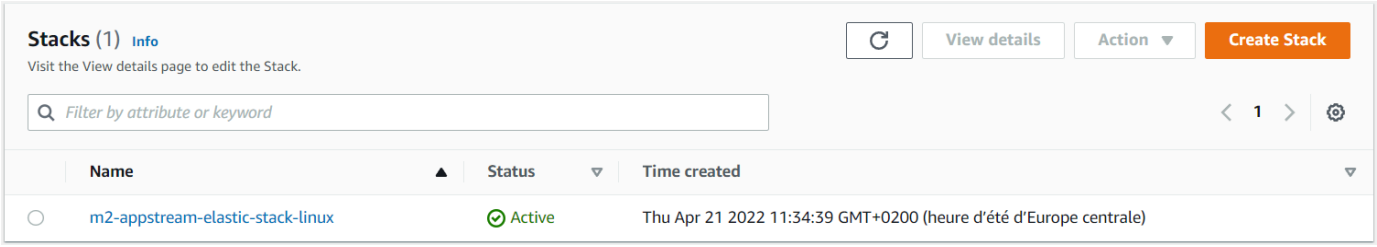
6. Selecione Próximo e Próximo novamente.
7. Escolha Eu reconheço que AWS CloudFormation pode criar recursos do IAM com nomes personalizados. e, em seguida, escolha Enviar.
8. Depois de criar a frota, crie CloudFormation pilhas com os outros modelos baixados para concluir a configuração dos aplicativos. Certifique-se de atualizar BucketNamesempre para apontar para o bucket correto do S3. Você pode editar o BucketNameno CloudFormation console. Como alternativa, você pode editar os arquivos do modelo diretamente e atualizar a `S3Bucket` propriedade.

> **Note**
>
> Os modelos baixados esperam encontrar ativos em um bucket do S3 com uma estrutura de pastas chamada `appstream/bluage/developer-ide/`. O bucket deve estar na Região da AWS mesma frota que você criou.

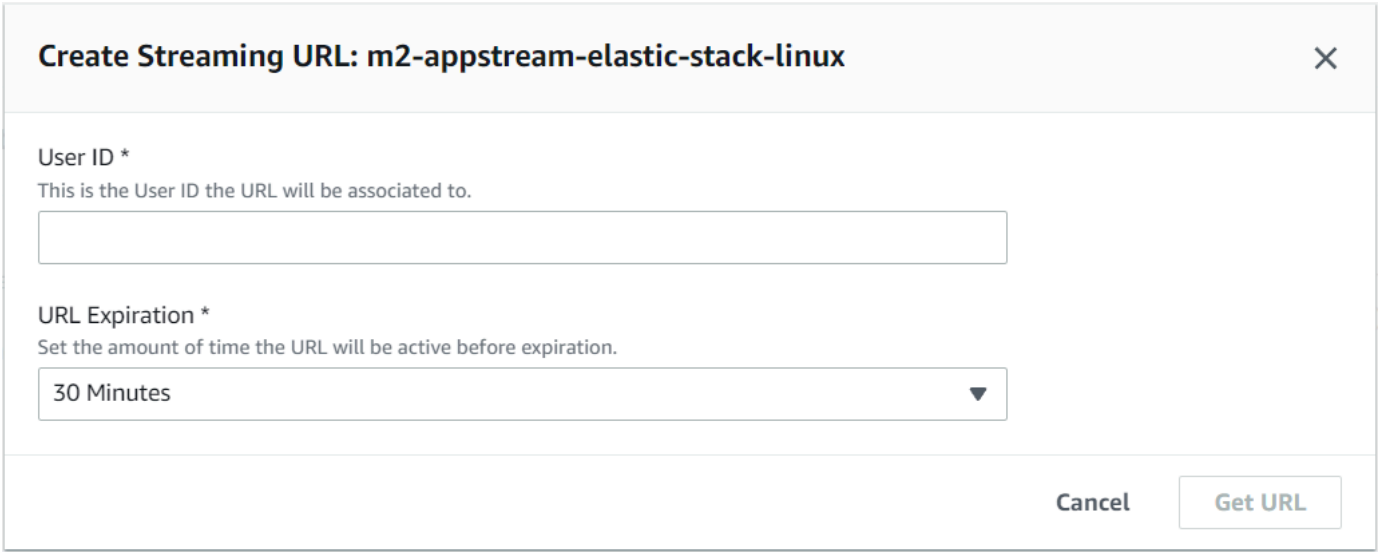
Etapa 6: acessar uma instância

Depois de criar e iniciar a frota, você pode criar um link temporário para acessar a frota por meio do cliente nativo.

1. Navegue até AppStream 2.0 no AWS Management Console e escolha a pilha criada anteriormente:



2. Na página de detalhes da pilha, escolha Ação e, em seguida, escolha Criar URL de streaming:



3. Em Criar URL de streaming, insira um ID de usuário arbitrário e um prazo de validade do URL e escolha Obter URL. Você obtém uma URL que pode ser usada para transmitir para um navegador ou para o cliente nativo. Recomendamos que você transmita para o cliente nativo.

Limpar os recursos

Para o procedimento de limpeza da pilha e das frotas criadas, consulte Criar uma frota e uma pilha AppStream 2.0.

Depois de excluir os objetos AppStream 2.0, você ou o administrador da conta também podem limpar os buckets do S3 para configurações do aplicativo e pastas pessoais.

> **Note**
>
> A pasta inicial de um determinado usuário é exclusiva em todas as frotas, portanto, talvez seja necessário mantê-la se outras pilhas AppStream 2.0 estiverem ativas na mesma conta.

Você não pode usar o console AppStream 2.0 para excluir usuários. Em vez disso, você deve usar a API do serviço com AWS CLI o. Para obter mais informações, consulte Administração de grupos de usuários no Guia de administração da Amazon AppStream 2.0.

Tutorial: Use o AWS Blu Age Developer na versão 2.0 AppStream

Este tutorial mostra como acessar o AWS Blu Age Developer na AppStream versão 2.0 e usá-lo com um aplicativo de amostra para que você possa experimentar os recursos. Ao concluir este tutorial, você poderá usar as mesmas etapas em suas próprias aplicações.

Tópicos

Etapa 1: criar um banco de dados

Nesta etapa, você usa o Amazon RDS para criar um banco de dados PostgreSQL gerenciado que a aplicação de demonstração usa para armazenar informações de configuração.

1. Abra o console do Amazon RDS.
2. Escolha Bancos de dados > Criar banco de dados.
3. Escolha Standard create > PostgreSQL, deixe a versão padrão e escolha Free tier.
4. Escolha um identificador de instância de BD.
5. Em Configurações de credenciais, escolha Gerenciar credenciais mestre no AWS Secrets Manager. Para obter mais informações, consulte Gerenciamento de senhas com Amazon RDS e o AWS Secrets Manager no Guia do usuário do Amazon RDS.
6. Certifique-se de que a VPC seja a mesma que você usa para a instância AppStream 2.0. Você pode solicitar esse valor ao administrador.
7. Para o grupo de segurança VPC, escolha Criar novo.
8. Defina Acesso público como Sim.
9. Deixe todos os outros valores padrão. Analise esses valores.
10. Selecione Criar banco de dados.

Para tornar o servidor de banco de dados acessível a partir da sua instância, selecione o servidor de banco de dados no Amazon RDS. Em Conectividade e segurança, escolha o grupo de segurança VPC para o servidor de banco de dados. Esse grupo de segurança foi criado anteriormente para você e deve ter uma descrição semelhante à do console de gerenciamento Criado pelo RDS. Escolha Ação > Editar regras de entrada, escolha Adicionar regra e crie uma regra do tipo PostgreSQL. Para fonte de regra, use o grupo de segurança padrão. Você pode começar a digitar o nome da fonte no campo Fonte e, em seguida, aceitar a ID sugerida. Por fim, escolha Salvar regras.

Etapa 2: acessar o ambiente

Nesta etapa, você acessa o ambiente de desenvolvimento AWS Blu Age na AppStream versão 2.0.

1. Entre em contato com seu administrador para saber a forma correta de acessar sua instância AppStream 2.0. Para obter informações gerais sobre possíveis clientes e configurações, consulte AppStream 2.0 Access Methods and Clients no Amazon AppStream 2.0 Administration Guide. Considere usar o cliente nativo para obter a melhor experiência.
2. Na AppStream versão 2.0, escolha Desktop.

Etapa 3: configurar o runtime

Nesta etapa, você configura o tempo de execução do AWS Blu Age. Você deve configurar o runtime na primeira inicialização e novamente se for notificado sobre uma atualização do runtime. Essa etapa preenche sua pasta `.m2`.

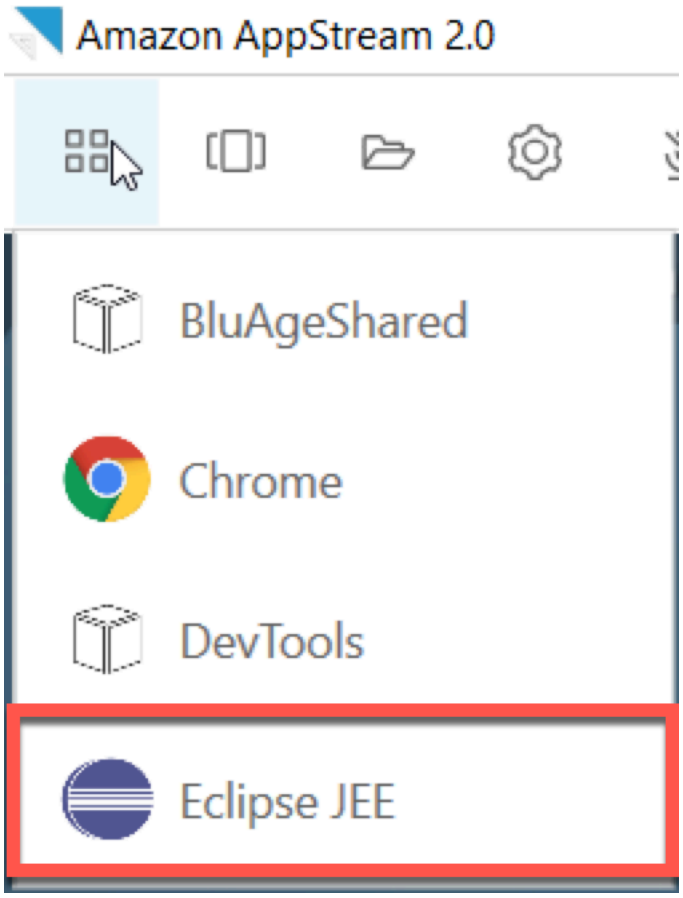
1. Escolha Aplicações, na barra de menu, e escolha Terminal.
2. Digite o comando :

```
~/_install-velocity-runtime.sh
```

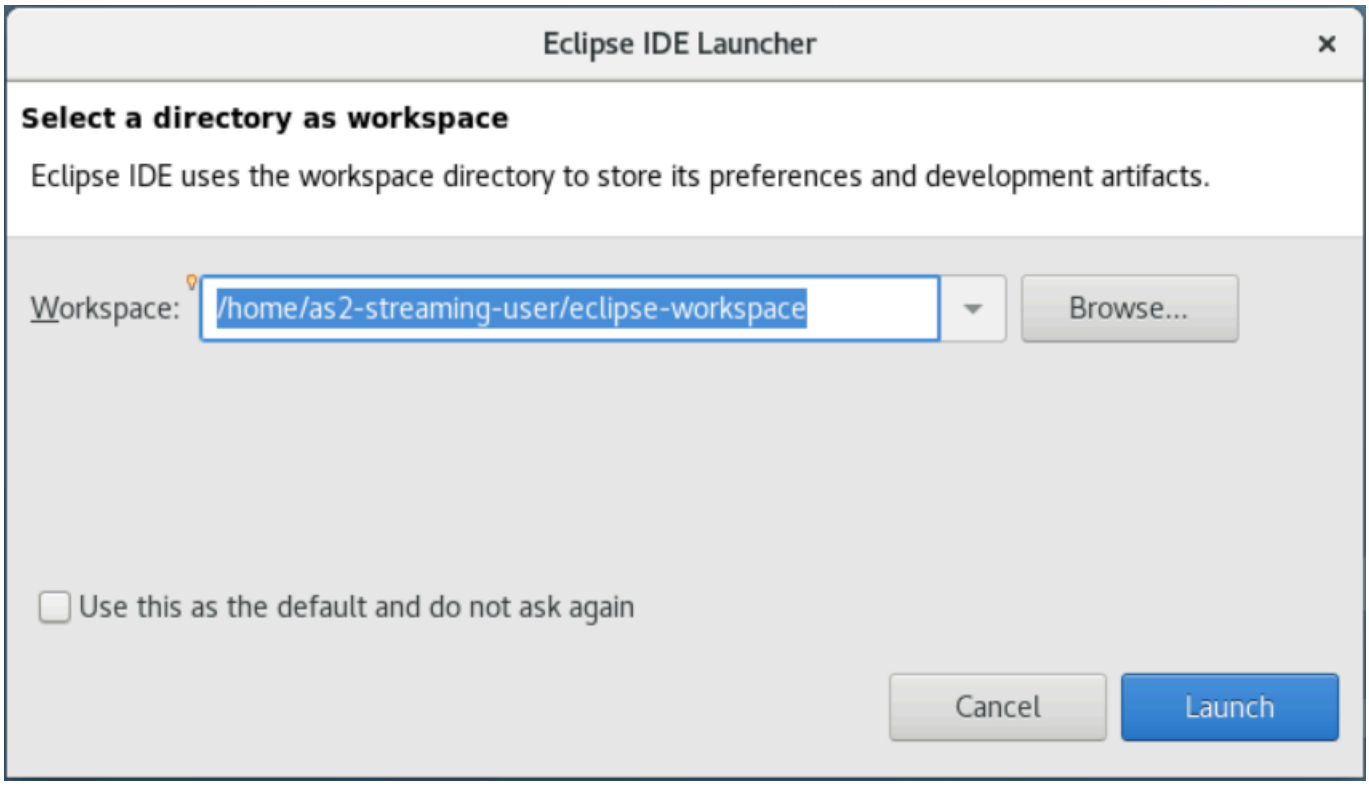
Etapa 4: iniciar o Eclipse IDE

Nesta etapa, você inicia o Eclipse IDE e escolhe um local onde deseja criar um espaço de trabalho.

1. Na AppStream versão 2.0, escolha o ícone Launch Application na barra de ferramentas e, em seguida, escolha Eclipse JEE.



2. Quando o inicializador abrir, insira o local em que você deseja criar seu espaço de trabalho e escolha Iniciar.



Opcionalmente, você pode iniciar o Eclipse a partir da linha de comando, da seguinte forma:

```
~/eclipse &
```

Etapa 5: configurar um projeto Maven

Nesta etapa, você importará um projeto em Maven para a aplicação de demonstração Planets.

1. Faça PlanetsDemoo upload de -pom.zip para sua pasta inicial. Você pode usar o recurso “Meus arquivos” do cliente nativo para fazer isso.
2. Use a ferramenta de linha de `unzip` comando para extrair os arquivos.
3. Navegue dentro da pasta descompactada e abra a raiz `pom.xml` do seu projeto em um editor de texto.
4. Edite a propriedade `gapwalk.version` para que ela corresponda ao AWS Blu Age Runtime instalado.

   Se não tiver certeza da versão instalada, emita o seguinte comando em um terminal:

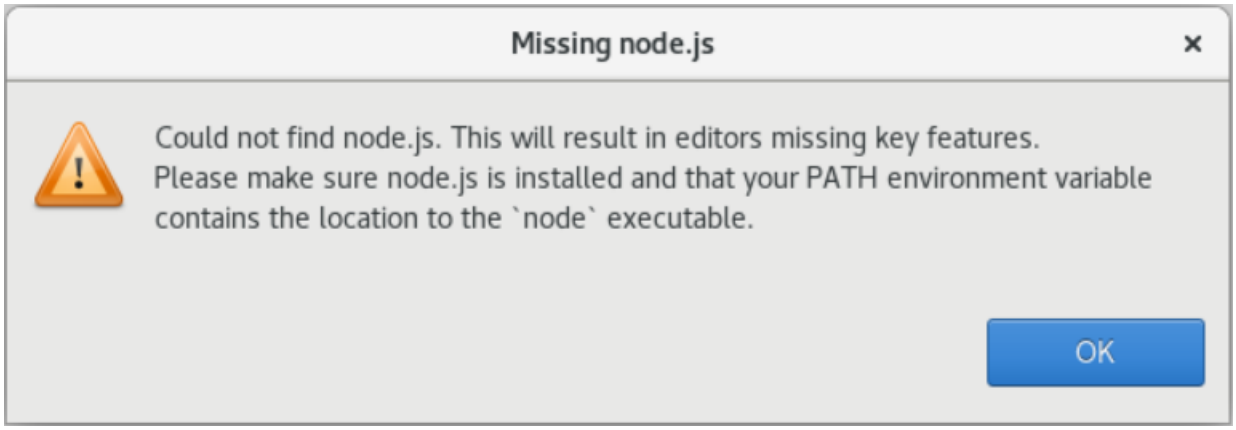
```
cat ~/runtime-version.txt
```

Esse comando imprime a versão de runtime atualmente disponível, por exemplo, `3.1.0-b3257-dev`.

> **Note**
>
> Não inclua o sufixo `-dev` em `gapwalk.version`. Por exemplo, um valor válido seria `<gapwalk.version>3.1.0-b3257</gapwalk.version>`.

5. Em Eclipse, escolha Arquivo e Importar. Na janela de diálogo Importar, expanda Maven e escolha Projetos existentes do Maven. Escolha Próximo.
6. Em Importar projetos do Maven, forneça a localização dos arquivos extraídos e escolha Concluir.

   Você pode ignorar isso com segurança. O Maven baixa uma cópia local do `node.js` para criar a parte Angular (*-web) do projeto:

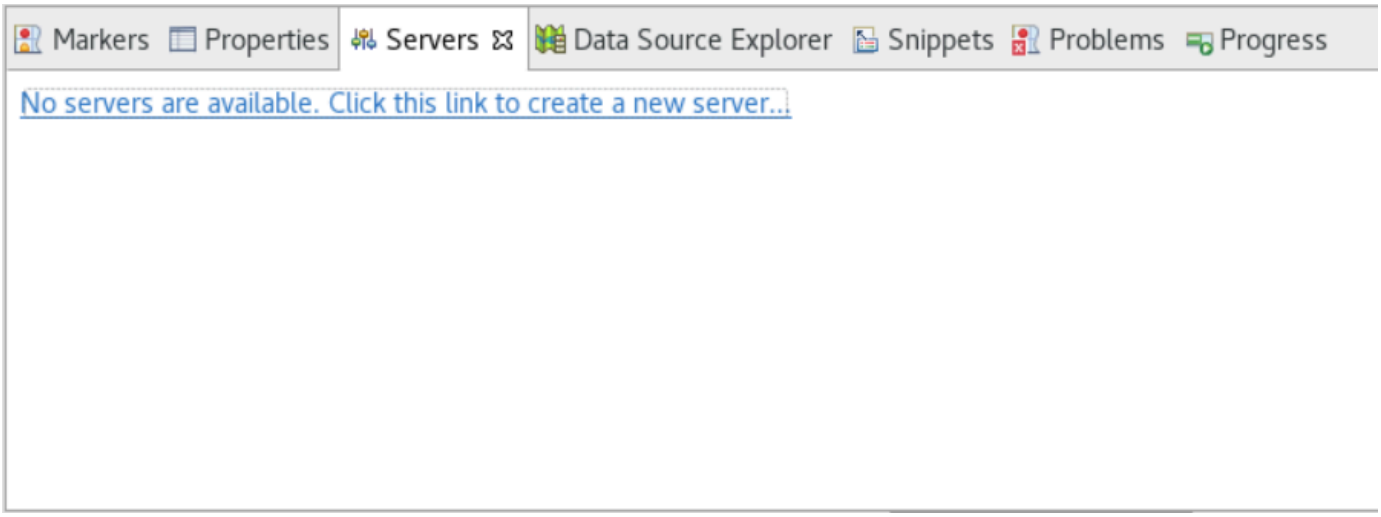
   

   Espere até o final da compilação. É possível acompanhar a compilação na visualização Progresso.
7. No Eclipse, selecione o projeto e escolha Executar como. Em seguida, escolha a instalação do Maven. Depois que a instalação do Maven for bem-sucedida, ele criará o arquivo `war` em `PlanetsDemoPom/PlanetsDemo-web/target/PlanetsDemo-web-1.0.0.war`.

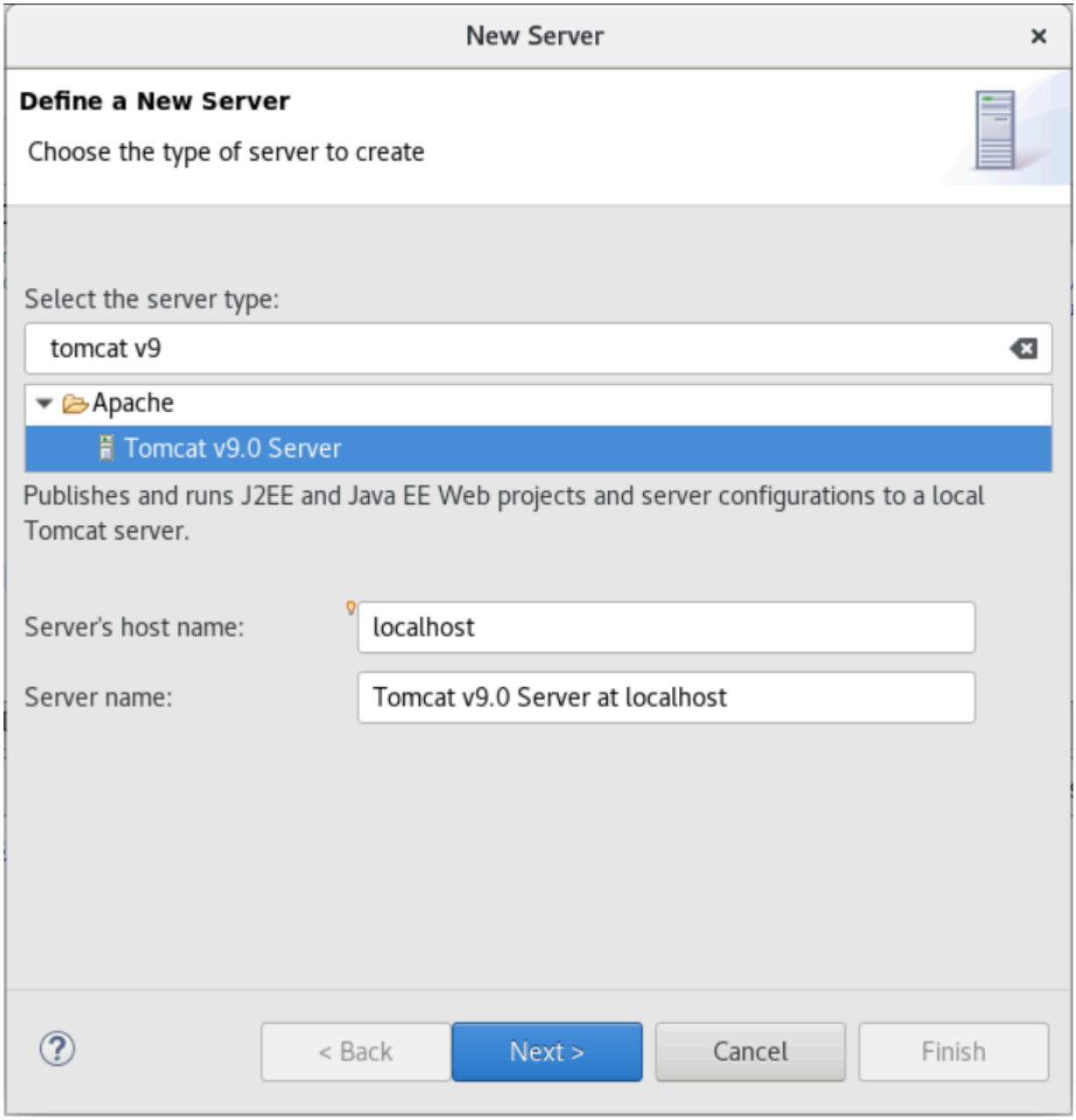
Etapa 6: configurar um servidor Tomcat

Nesta etapa, você configura um servidor Tomcat em que você implanta e inicia sua aplicação compilada.

1. No Eclipse, escolha Janela > Mostrar visualização > Servidores para mostrar a visualização Servidores:

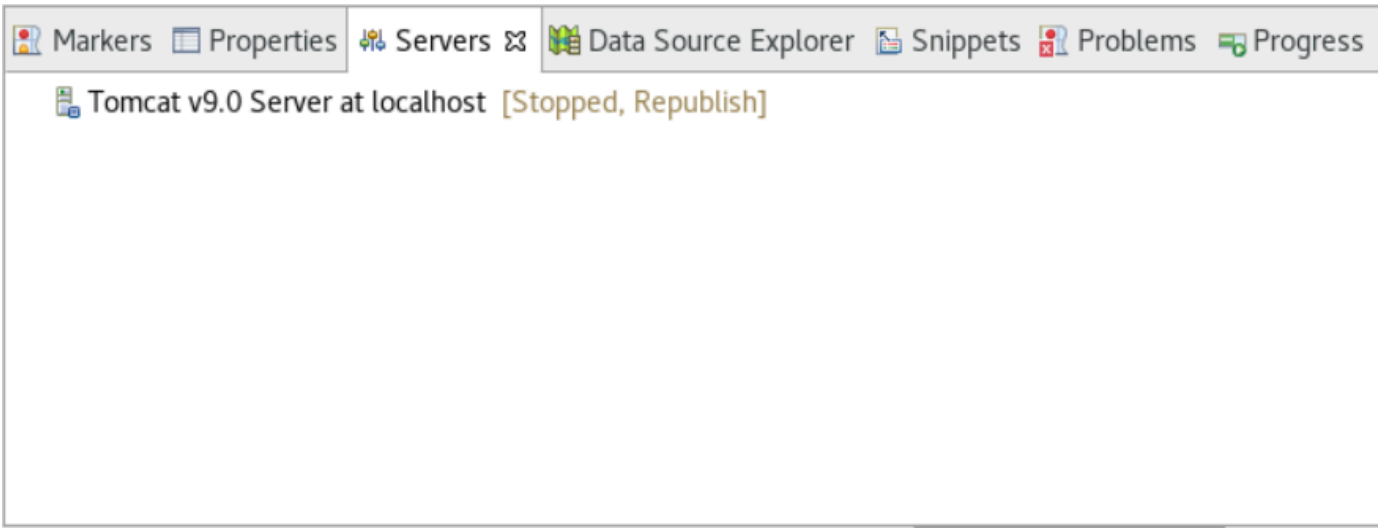


2. Escolha Nenhum servidor está disponível. Clique neste link para criar um novo servidor... . O assistente de Novo servidor é exibido. No campo Selecionar o tipo de servidor do assistente, digite tomcat v9 e escolha Servidor Tomcat v9.0. Em seguida, escolha Próximo.



3. Escolha Procurar e escolha a pasta tomcat na raiz da pasta inicial. Deixe o JRE em seu valor padrão e escolha Concluir.

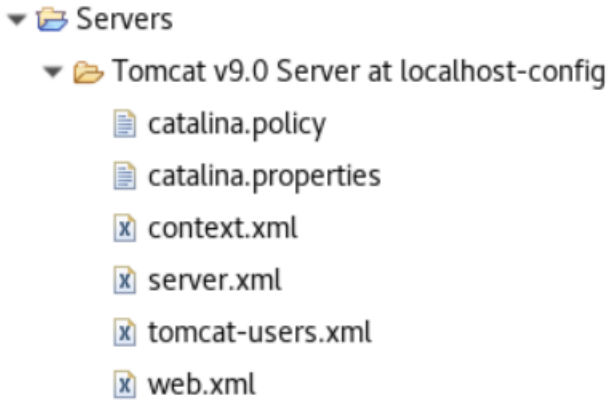
   Um projeto de servidores é criado no espaço de trabalho, e um servidor Tomcat v9.0 agora está disponível na visualização de servidores. É aqui que a aplicação compilada será implantada e iniciada:



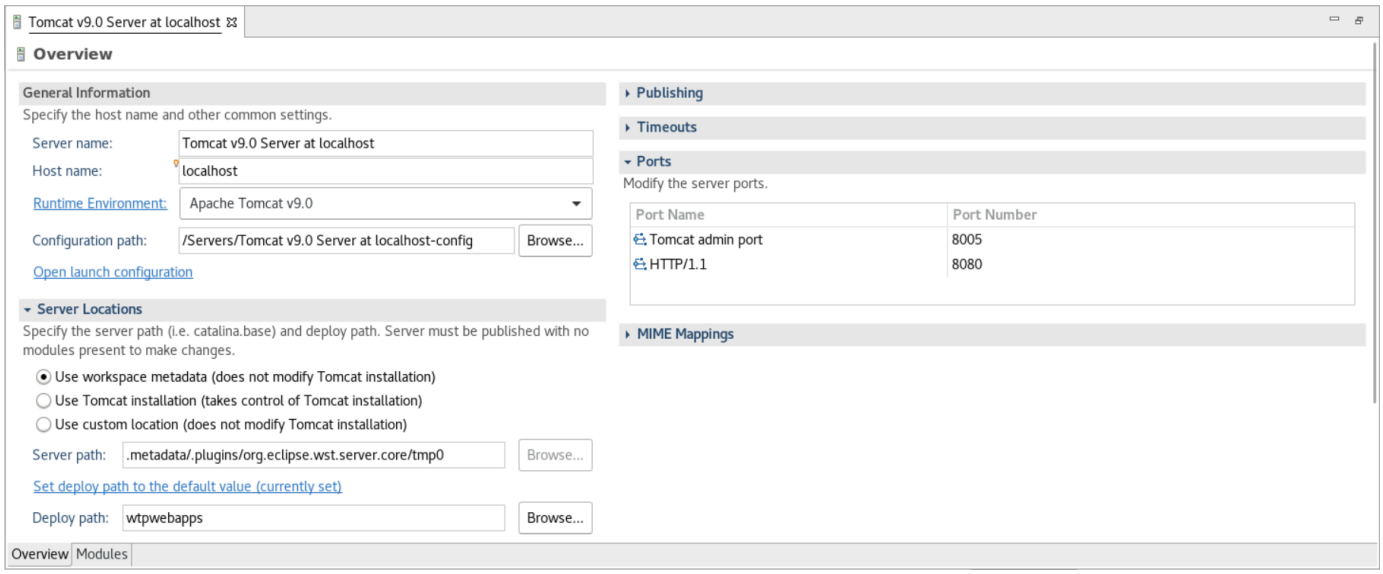
Etapa 7: implantar no Tomcat

Nesta etapa, você implantará a aplicação de demonstração Planets no servidor Tomcat para poder executar a aplicação.

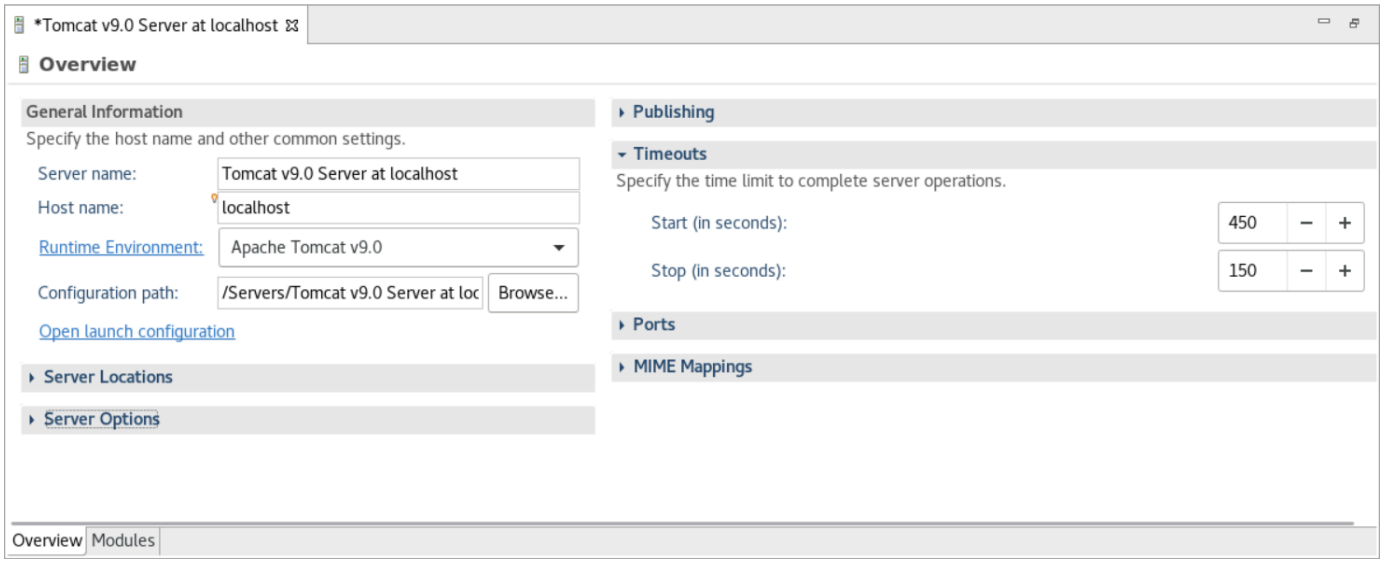
1. Selecione o arquivo `PlanetsDemo-web` e escolha Executar como > Instalação do Maven. Selecione `PlanetsDemo-web` novamente e escolha Atualizar para garantir que o frontend compilado com npm seja compilado corretamente em um .war e seja notado pelo Eclipse.
2. Faça upload do PlanetsDemo-runtime.zip na instância e descompacte o arquivo em um local acessível. Isso garante que a aplicação de demonstração possa acessar as pastas e arquivos de configuração necessários.
3. Copie o conteúdo de `PlanetsDemo-runtime/tomcat-config` na subpasta `Servers/Tomcat v9.0...` que você criou para o seu servidor Tomcat:



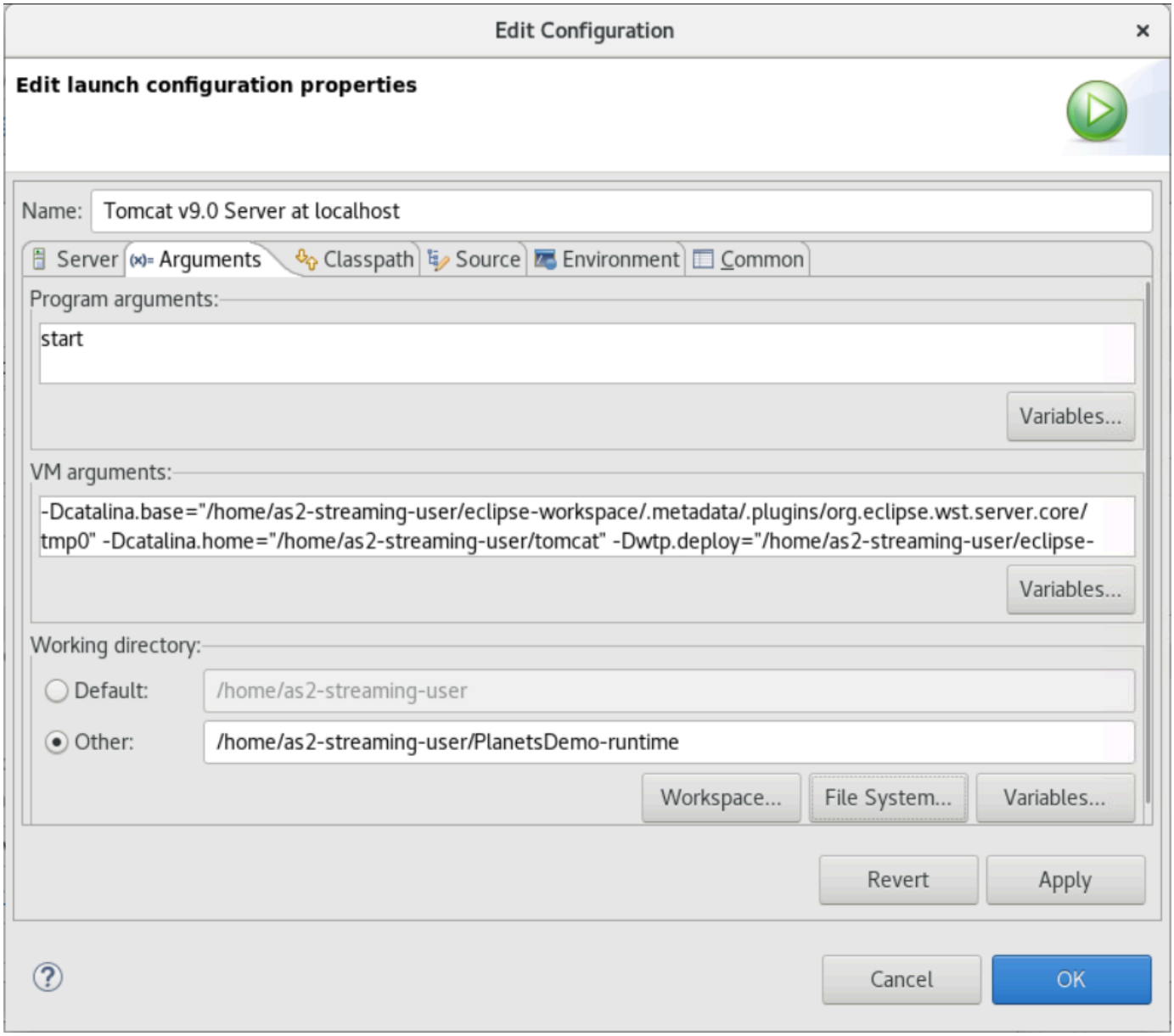
4. Abra a entrada do servidor `tomcat v9.0` na visualização Servidores. O editor de propriedades do servidor aparece:



5. Na guia Visão geral, aumente os valores de tempo limite para 450 segundos para Iniciar e 150 segundos para Parar, conforme mostrado aqui:



6. Escolha Abrir configuração de lançamento. É exibido um assistente. No assistente, navegue até a pasta Argumentos e, em Diretório de trabalho, escolha Outro. Escolha Sistema de arquivos e navegue até a `PlanetsDemo-runtime` pasta descompactada anteriormente. Essa pasta deve conter uma subpasta direta chamada config.



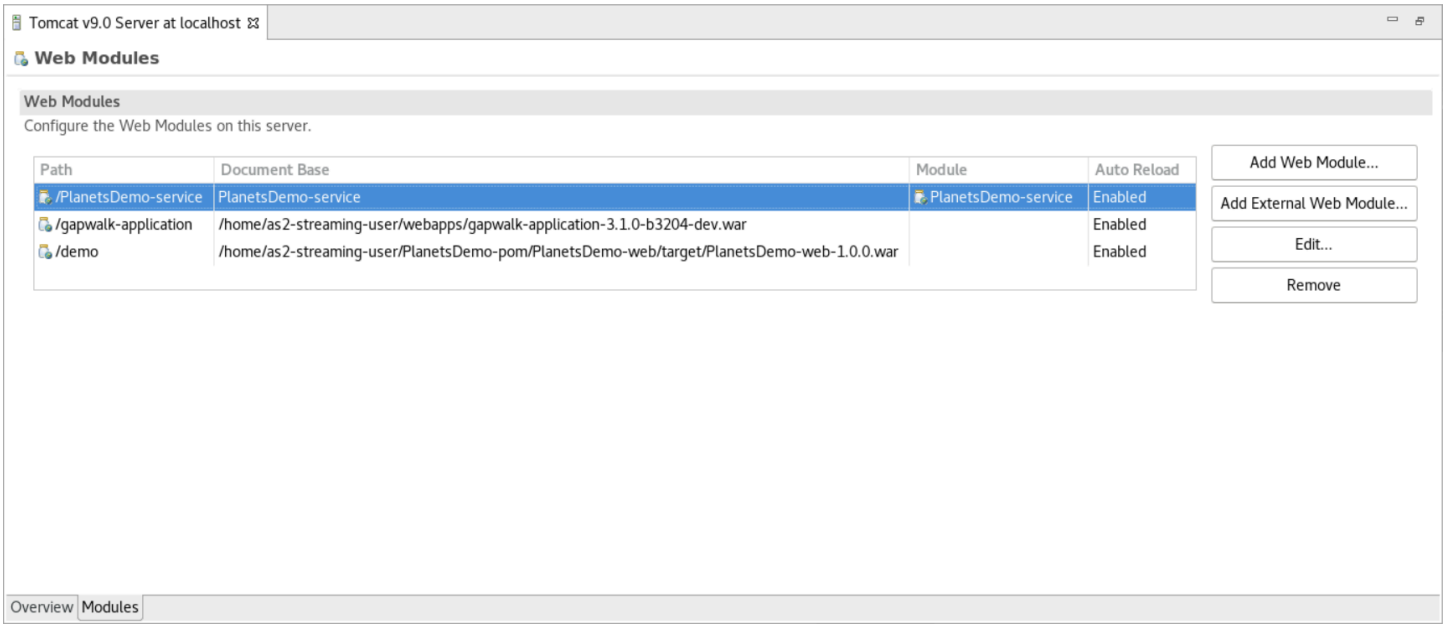
7. Escolha a guia Módulos do editor de propriedades do servidor e faça as seguintes alterações:

   - Escolha Adicionar módulo Web e adicione `PlanetsDemo-service`.
   - Escolha Adicionar módulo Web externo. A janela de diálogo Adicionar módulo Web é exibida. Faça as seguintes alterações em:
     - Na base de documentos, escolha Procurar e navegue até `~/webapps/gapwalk-application...war`
     - Em Caminho, insira`/gapwalk-application`.
   - Escolha OK.
   - Escolha Adicionar módulo Web externo novamente e faça as seguintes alterações:

- Para Document base, insira o caminho para o frontend .war (in) `PlanetsDemo-web/target`
- Em Path, insira `/demo`

- Escolha OK
- Salve as modificações do editor (Ctrl + S).

O conteúdo do editor agora deve ser semelhante ao seguinte.



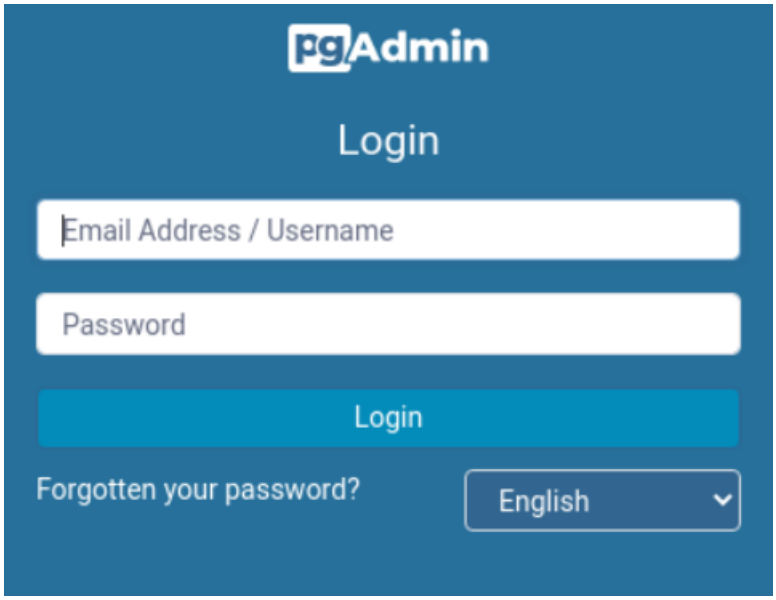
Etapa 8: criar o banco de dados do JICS

Nesta etapa, você se conecta ao banco de dados do criado em Etapa 1: criar um banco de dados.

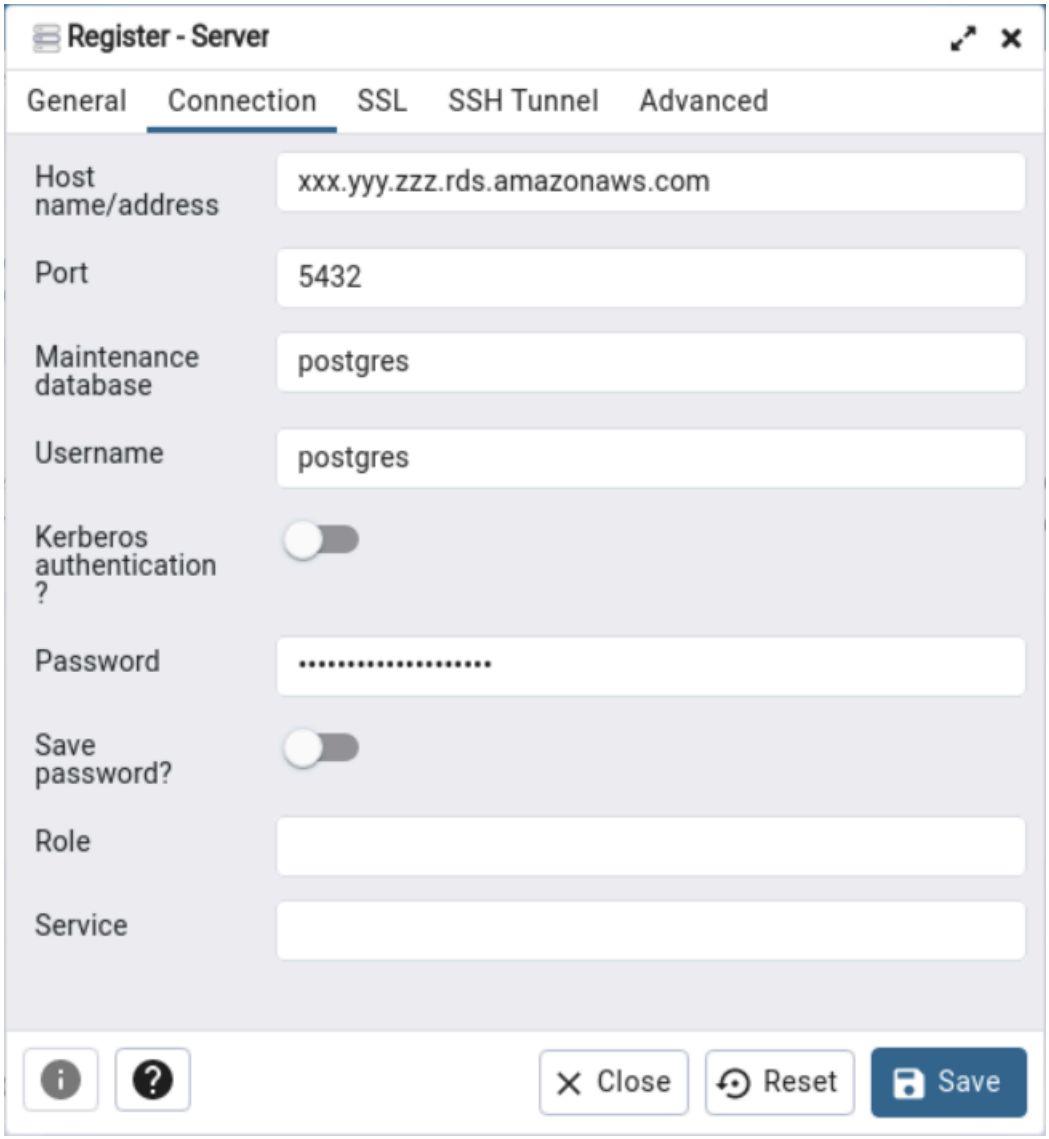
1. Na instância AppStream 2.0, execute o seguinte comando em um terminal para iniciar`pgAdmin`:

```
./pgadmin-start.sh
```

2. Escolha um endereço de e-mail e uma senha como identificadores de login. Anote o URL fornecido (normalmente http://127.0.0.1:5050). Inicie o Google Chrome na instância, copie e cole o URL no navegador e faça login com seus identificadores.



3. Depois de fazer login, escolha Adicionar novo servidor e insira as informações de conexão no banco de dados criado anteriormente da seguinte maneira.

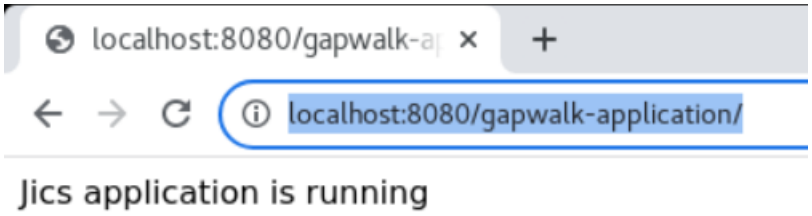


4. Ao se conectar ao servidor do banco de dados, use Objeto > Criar > Banco de dados e crie um novo banco de dados chamado jics.
5. Edite as informações de conexão do banco de dados que a aplicação de demonstração usou. Essas informações são definidas em `PlanetsDemo-runtime/config/application-main.yml`. Pesquise a entrada `jicsDs`. Para recuperar os valores `username` e `password`, no console do Amazon RDS, navegue até o banco de dados. Na guia Configuration (Configuração), em Master Credentials ARN (ARN das credenciais principais), escolha Manage in Secrets Manager (Gerenciar no Secrets Manager). Em seguida, no console do Secrets Manager, no segredo, escolha Recuperar valor secreto.

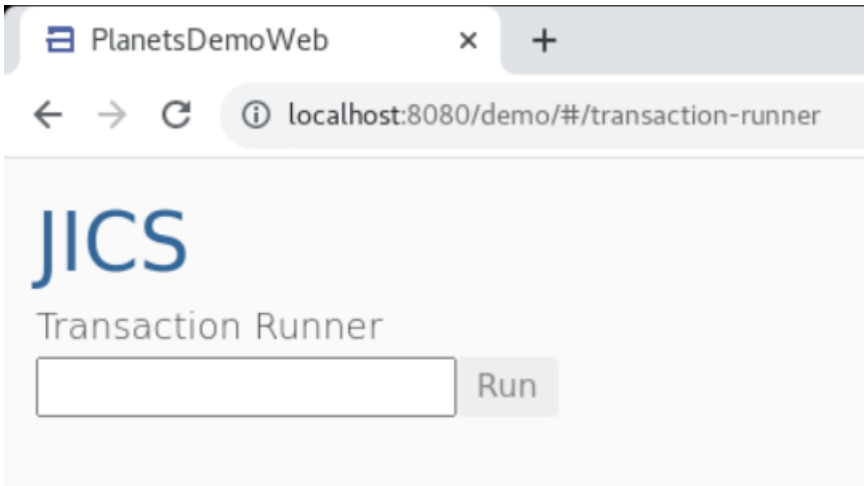
Etapa 9: iniciar e testar a aplicação

Nesta etapa, você inicia o servidor Tomcat e a aplicação de demonstração para poder testá-la.

1. Para iniciar o servidor Tomcat e as aplicações implantadas anteriormente, selecione a entrada do servidor na visualização Servidores e escolha Iniciar. É exibido um console que exibe os logs de inicialização.
2. Verifique o status do servidor na visualização Servidores ou aguarde a inicialização do servidor em [xxx] milissegundos na mensagem no console. Depois que o servidor for iniciado, verifique se a aplicação gapwalk-application está implantada corretamente. Para fazer isso, acesse a URL http://localhost:8080/gapwalk-application em um navegador Google Chrome. Você deverá ver o seguinte.

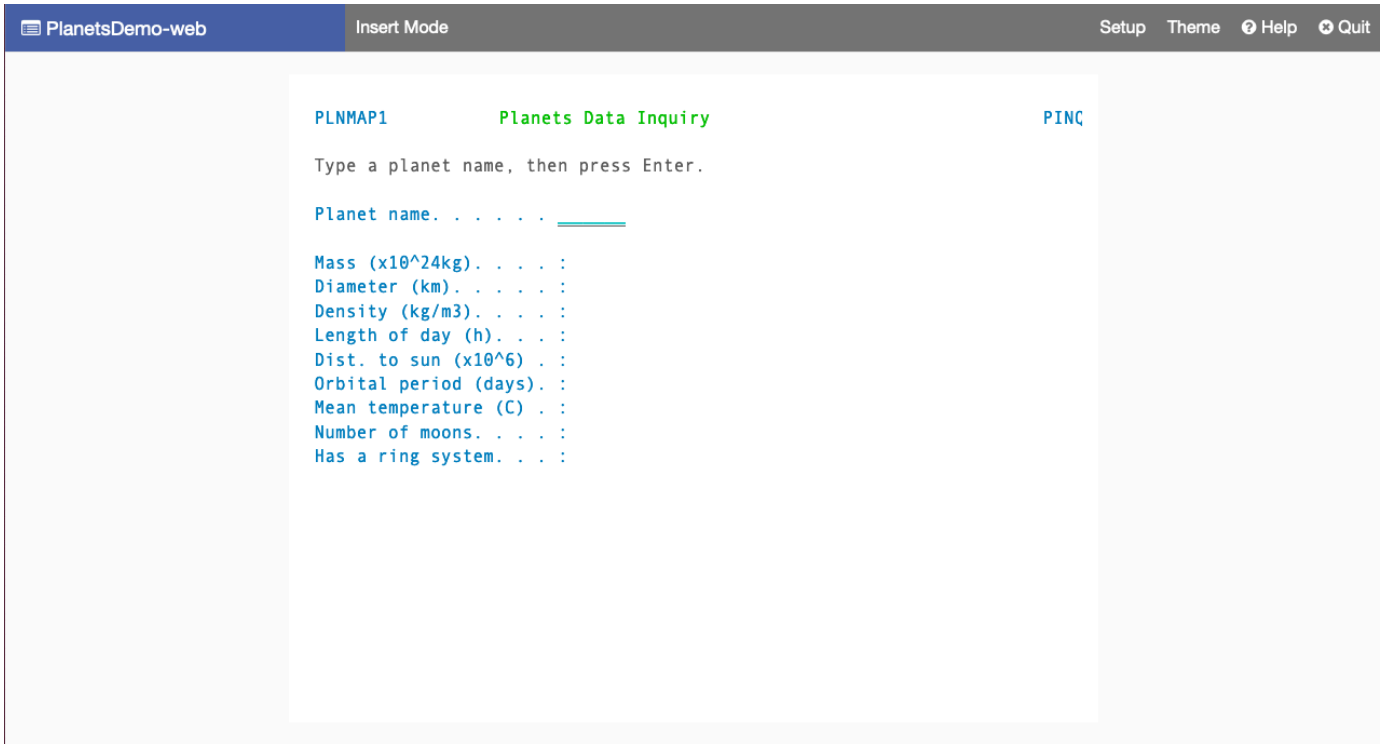


3. Acesse o front-end da aplicação implantada no Google Chrome em http://localhost:8080/demo. A seguinte página do Transaction Launcher deve aparecer.

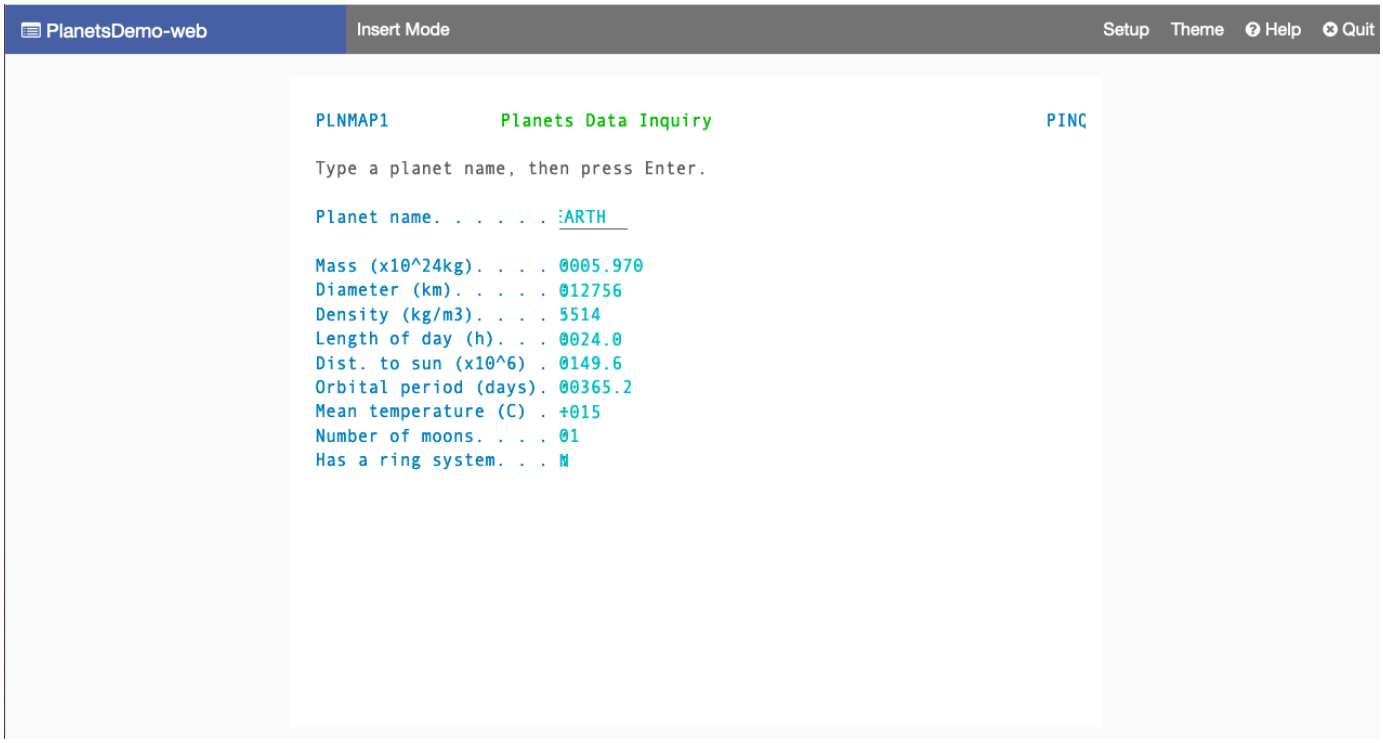


4. Para iniciar a transação da aplicação, insira `PINQ` no campo de entrada e escolha Executar (ou pressione Enter).

   A tela da aplicação de demonstração deve aparecer.



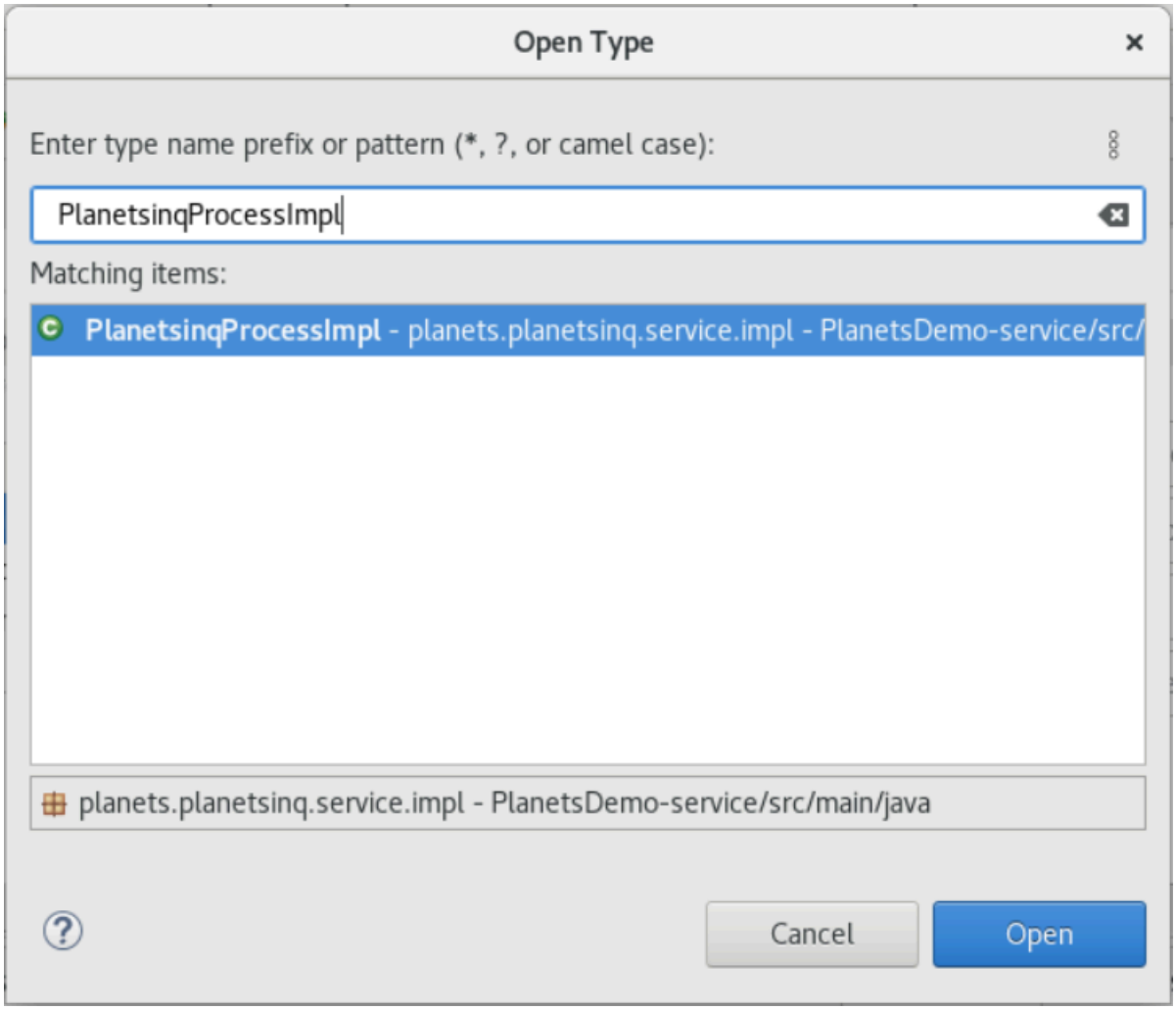
5. Digite o nome do planeta no campo correspondente e pressione Enter.



Etapa 10: depurar a aplicação

Nesta etapa, você testará usando os recursos de depuração padrão do Eclipse. Esses recursos estão disponíveis quando você trabalha em uma aplicação modernizada.

1. Para abrir a classe de serviço principal, pressione Ctrl + Shift + T. Em seguida, insira `PlanetsinqProcessImpl`.



2. Navegue até o método `searchPlanet` e coloque um ponto de interrupção nele.
3. Selecione o nome do servidor e selecione Reiniciar na depuração.
4. Repita as etapas anteriores. Ou seja, acesse a aplicação, insira o nome do planeta e pressione Enter.

   O Eclipse interromperá a aplicação no método `searchPlanet`. Agora você pode examiná-la.

Limpar os recursos

Se os recursos criados durante este tutorial não forem mais necessários, exclua-os para não incorrer em cobranças adicionais. Execute as etapas a seguir:

- Se a aplicação Planets ainda estiver em execução, pare-o.
- Exclua o banco de dados que você criou em Etapa 1: criar um banco de dados. Para ter mais informações, consulte Excluir uma instância de banco de dados.

Redefinir a plataforma de aplicações com a Micro Focus

Esta seção descreve cada etapa do processo de redefinição de plataforma. Ele descreve todas as tarefas e inclui informações sobre como configurar e operar o tempo de execução da modernização de AWS mainframe no Amazon EC2.

Tópicos

Configuração do Micro Focus Runtime (no Amazon EC2)

AWS A modernização do mainframe fornece várias imagens de máquina da Amazon (AMIs) que incluem produtos licenciados pela Micro Focus. Essas AMIs permitem que você provisione rapidamente instâncias do Amazon Elastic Compute Cloud (Amazon EC2) para oferecer suporte a ambientes da Micro Focus que você controla e gerencia. Este tópico fornece as etapas necessárias para acessar e iniciar essas AMIs. O uso dessas AMIs é totalmente opcional e elas não são obrigatórias para concluir os tutoriais deste guia do usuário.

Tópicos

- Criando a AWS Identity and Access Management função
- Conceda ao License Manager as permissões necessárias
- Inscreva-se para receber as imagens de máquina da Amazon
- Inicie uma instância Micro Focus de modernização de AWS mainframe
- Sub-rede ou VPC sem acesso à Internet
- Solução de problemas de licença

Pré-requisitos

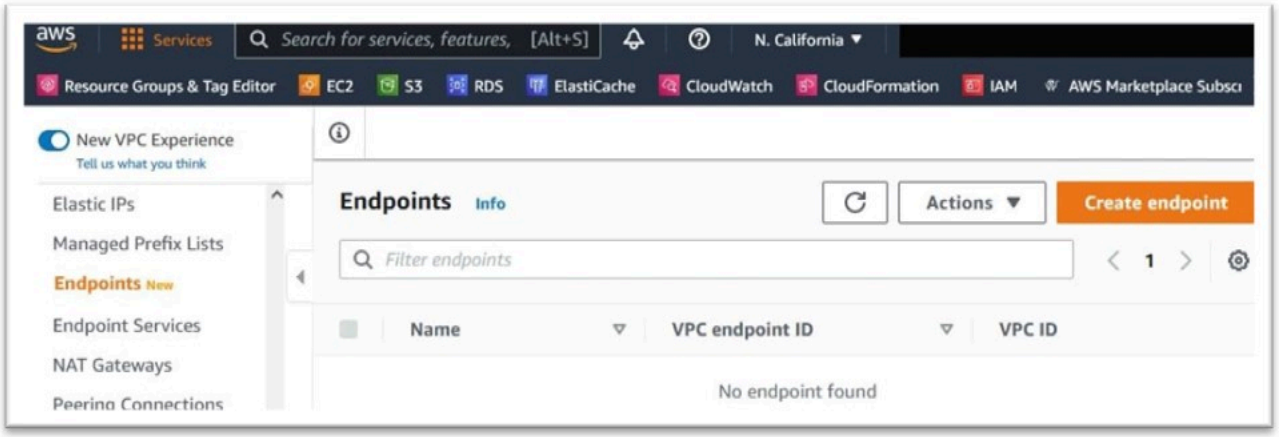
Certifique-se de que você atende aos seguintes pré-requisitos.

- Acesso do administrador à conta na qual as instâncias do Amazon EC2 serão criadas.
- Identifique Região da AWS onde as instâncias do Amazon EC2 serão criadas e verifique se o serviço de modernização do AWS mainframe está disponível. Consulte Serviços por região AWS. Selecione uma região na qual o serviço esteja disponível.
- Identifique a Amazon Virtual Private Cloud (Amazon VPC) na qual as instâncias do Amazon EC2 serão criadas.

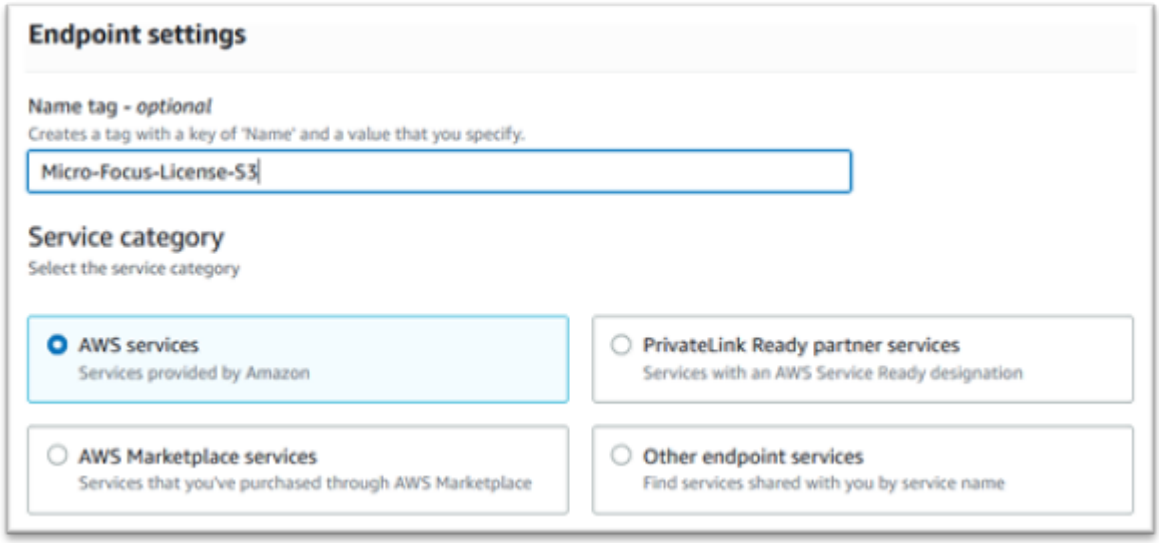
Criar o endpoint da Amazon VPC para o Amazon S3

Nesta seção, você cria um endpoint da Amazon VPC para o Amazon S3 usar.

1. Navegue até a Amazon VPC no AWS Management Console.
2. No painel de navegação, escolha Endpoints.
3. Escolha Criar endpoint.

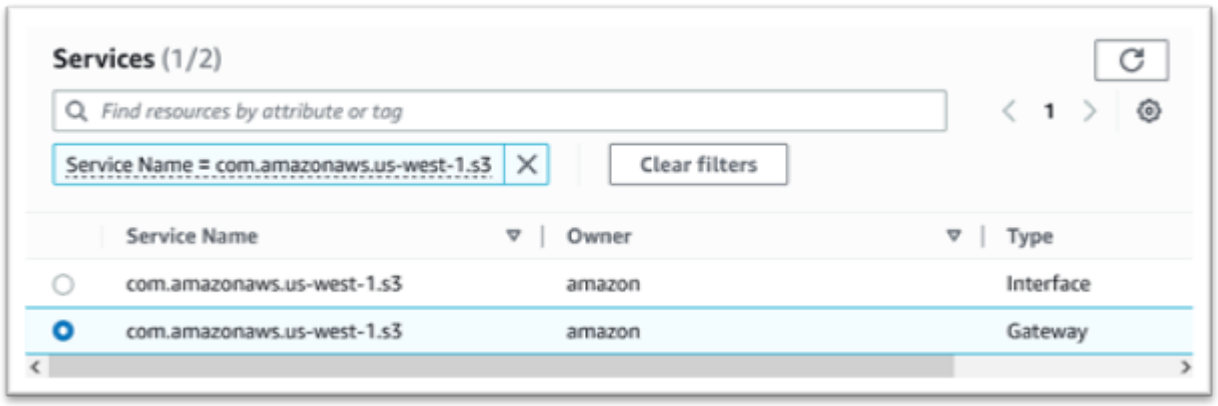


4. Insira uma etiqueta de nome significativa, por exemplo: “Micro-Focus-License-S3”.
5. Escolha AWS Services como a categoria de serviço.

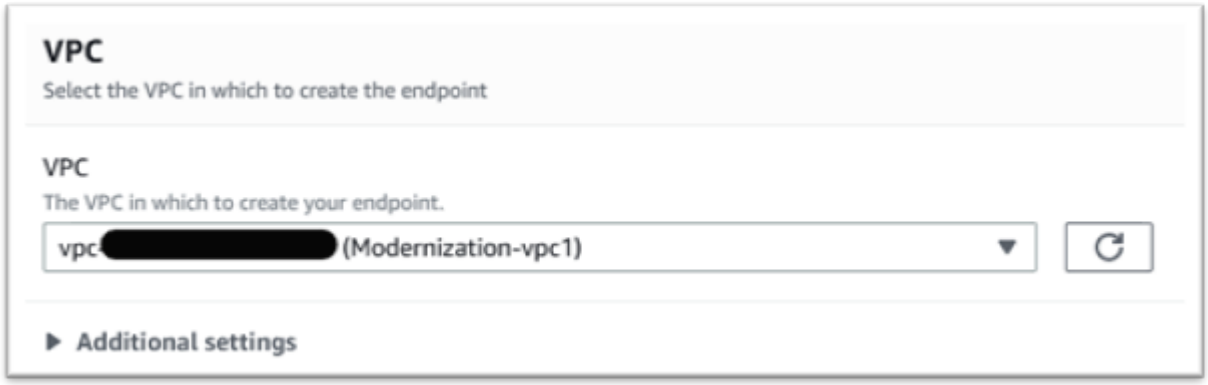


6. Em Serviços, pesquise o serviço Amazon S3 Gateway: com.amazonaws.[region].s3.

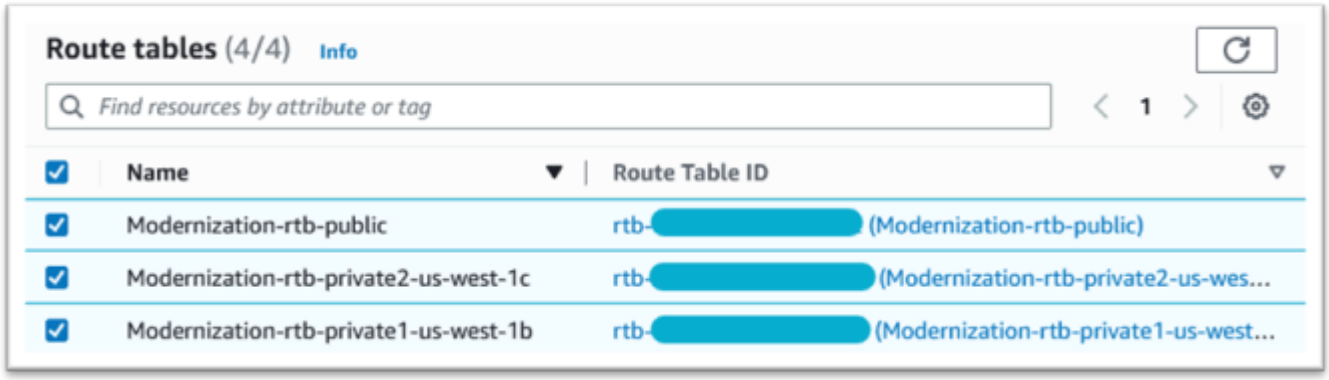
   Para `us-west-1`, isso seria: `com.amazonaws.us-west-1.s3`.
7. Escolha o serviço Gateway.



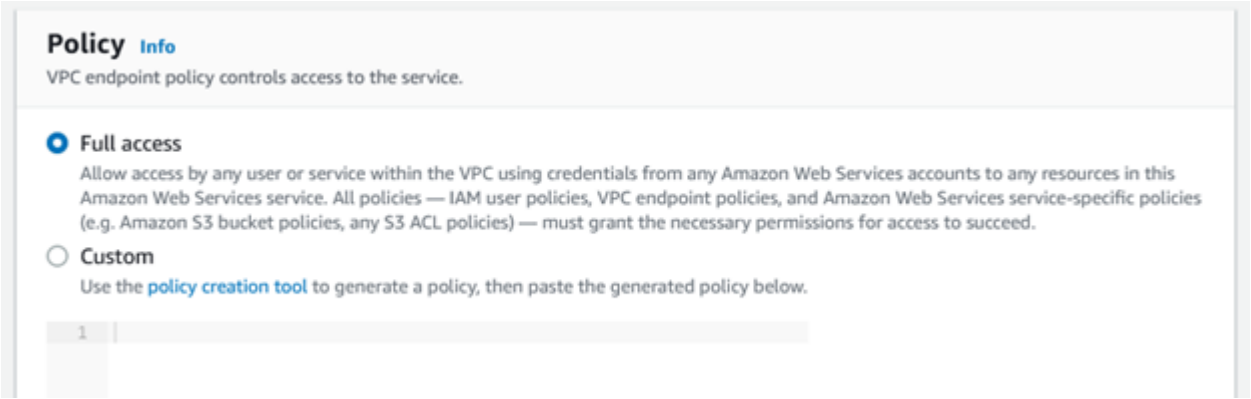
8. Para VPC, escolha a VPC que você usará.



9. Escolha todas as tabelas de rotas para a VPC.



10. Em Política, escolha Acesso total.



11. Escolha Criar Endpoint.

Solicite a atualização da lista de permissões para a conta

Trabalhe com seu AWS representante para que sua conta seja incluída na lista de permissões para as AMIs de modernização de AWS mainframe. Forneça as seguintes informações:

- O Conta da AWS ID.

- O Região da AWS local onde o endpoint da Amazon VPC foi criado.
- O ID do endpoint da Amazon VPC Amazon S3 criado em Criar o endpoint da Amazon VPC para o Amazon S3. Esse é o id `vpce-xxxxxxxxxxxxxxxxx` do endpoint com.amazonaws.[region].s3 Gateway.
- O número de licenças necessárias em todas as instâncias do Micro Focus Enterprise Suite AMI Amazon EC2.

  É necessária uma licença por núcleo de CPU (por 2 vCPUs para a maioria das instâncias do Amazon EC2).

  Para obter mais informações, consulte Otimizar as opções da CPU.

  O número solicitado pode ser ajustado futuramente por AWS.

> **Note**
>
> O AWS representante deve abrir o ticket de suporte para a solicitação da Lista de Permissões. Ela não pode ser solicitada diretamente e a solicitação pode levar vários dias para ser concluída.

Criando a AWS Identity and Access Management função

Crie uma AWS Identity and Access Management política e uma função para serem usadas pelas instâncias de modernização de AWS mainframe do Amazon EC2. Criar a função por meio do console do IAM criará um perfil da instância associado com o mesmo nome. A atribuição desse perfil da instância às instâncias do Amazon EC2 permite que as licenças da Micro Focus sejam atribuídas. Para obter mais informações sobre perfis de instância, consulte Uso de um perfil do IAM para conceder permissões a aplicações executadas em instâncias do Amazon EC2.

Criar uma política do IAM

Uma política de IAM é criada primeiro e depois anexada à função.

1. Navegue até AWS Identity and Access Management no AWS Management Console.
2. Escolha Políticas e depois Criar política.

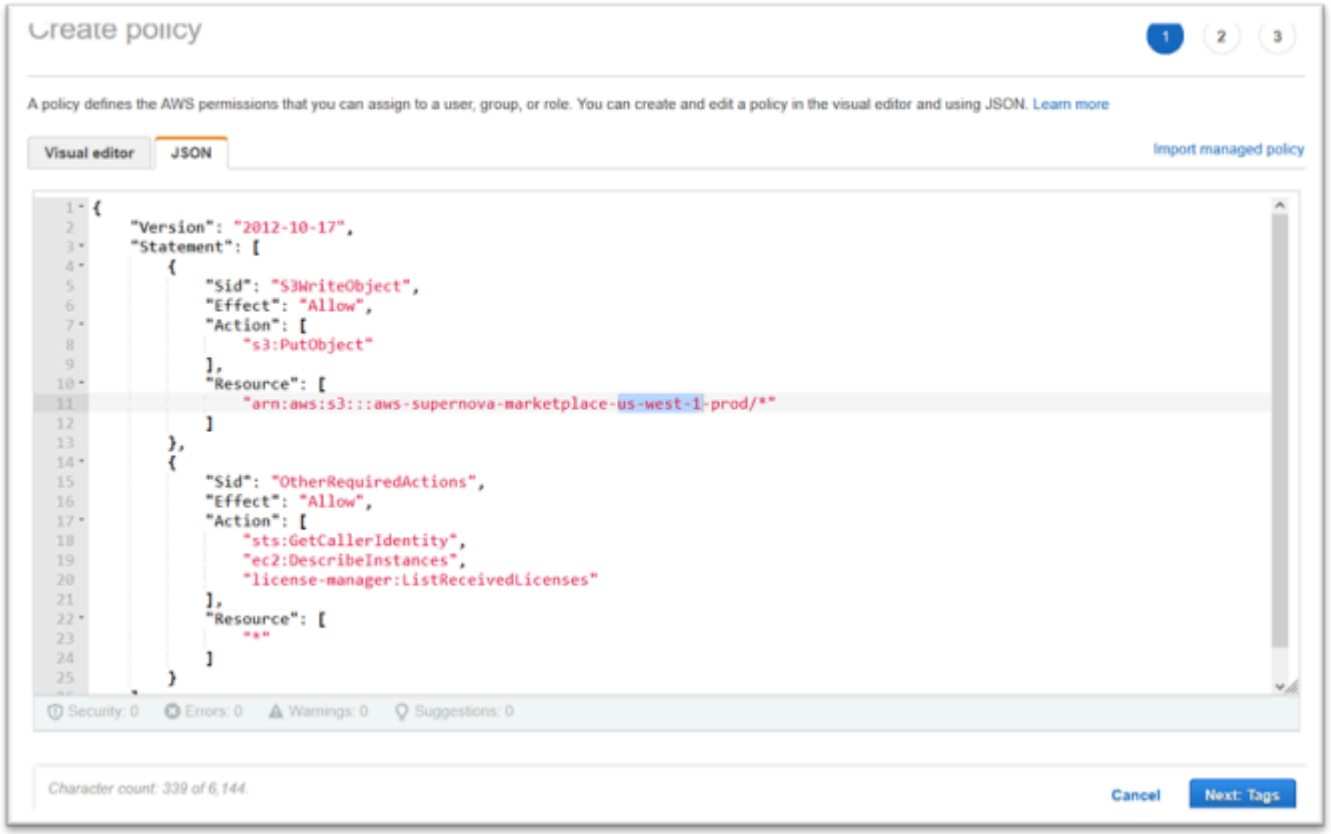
3. Selecione a guia JSON.

4. Substitua o seguinte JSON pelo local `us-west-1` em Região da AWS que o endpoint do Amazon S3 foi definido e, em seguida, copie e cole o JSON no editor de políticas.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3WriteObject",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::aws-supernova-marketplace-us-west-1-prod/*"
            ]
        },
        {
            "Sid": "OtherRequiredActions",
```

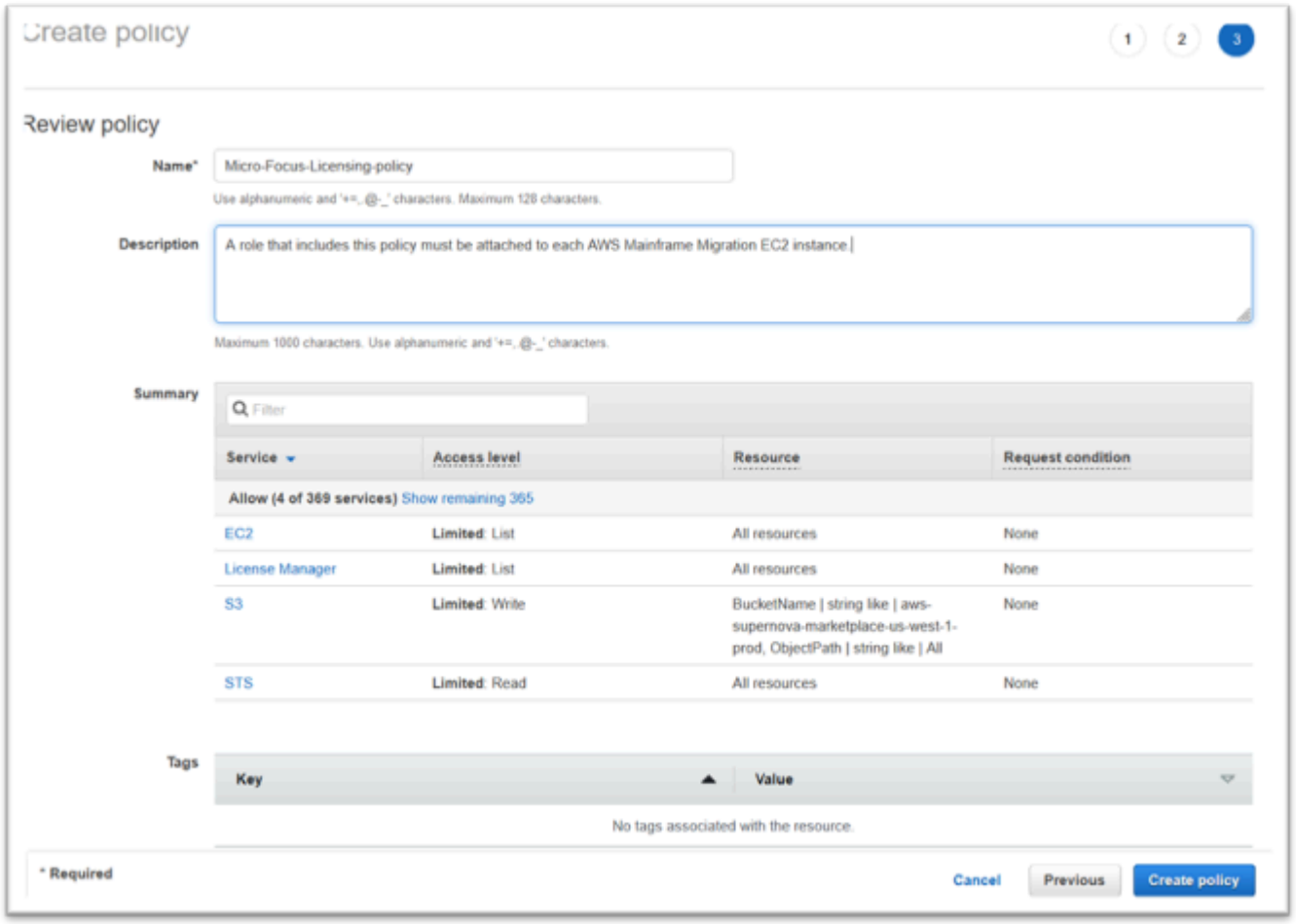
```
            "Effect": "Allow",
            "Action": [
                "sts:GetCallerIdentity",
                "ec2:DescribeInstances",
                "license-manager:ListReceivedLicenses"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

> **Note**
>
> As ações do Sid `OtherRequiredActions` não oferecem suporte a permissões de nível de recurso e precisam ser especificadas `*` no elemento de recurso.



5. Escolha Próximo: etiquetas.

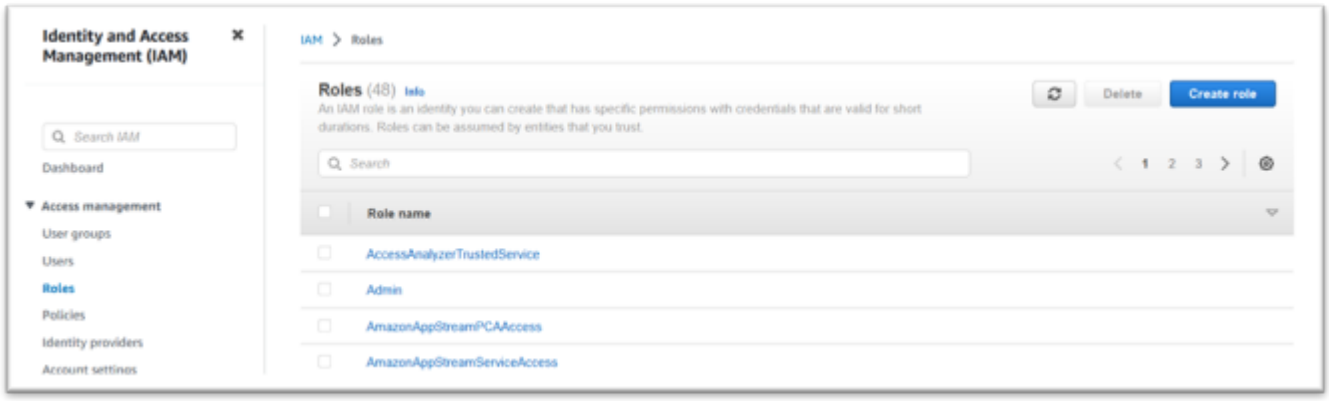
6. Opcionalmente, insira qualquer tag e escolha Próximo: Revisar.
7. Insira um nome para a política, por exemplo, “Política de licenciamento da Micro-Focus”. Opcionalmente, insira uma descrição, por exemplo, “Uma função que inclua essa política deve ser anexada a cada instância de modernização de AWS mainframe do Amazon EC2”.



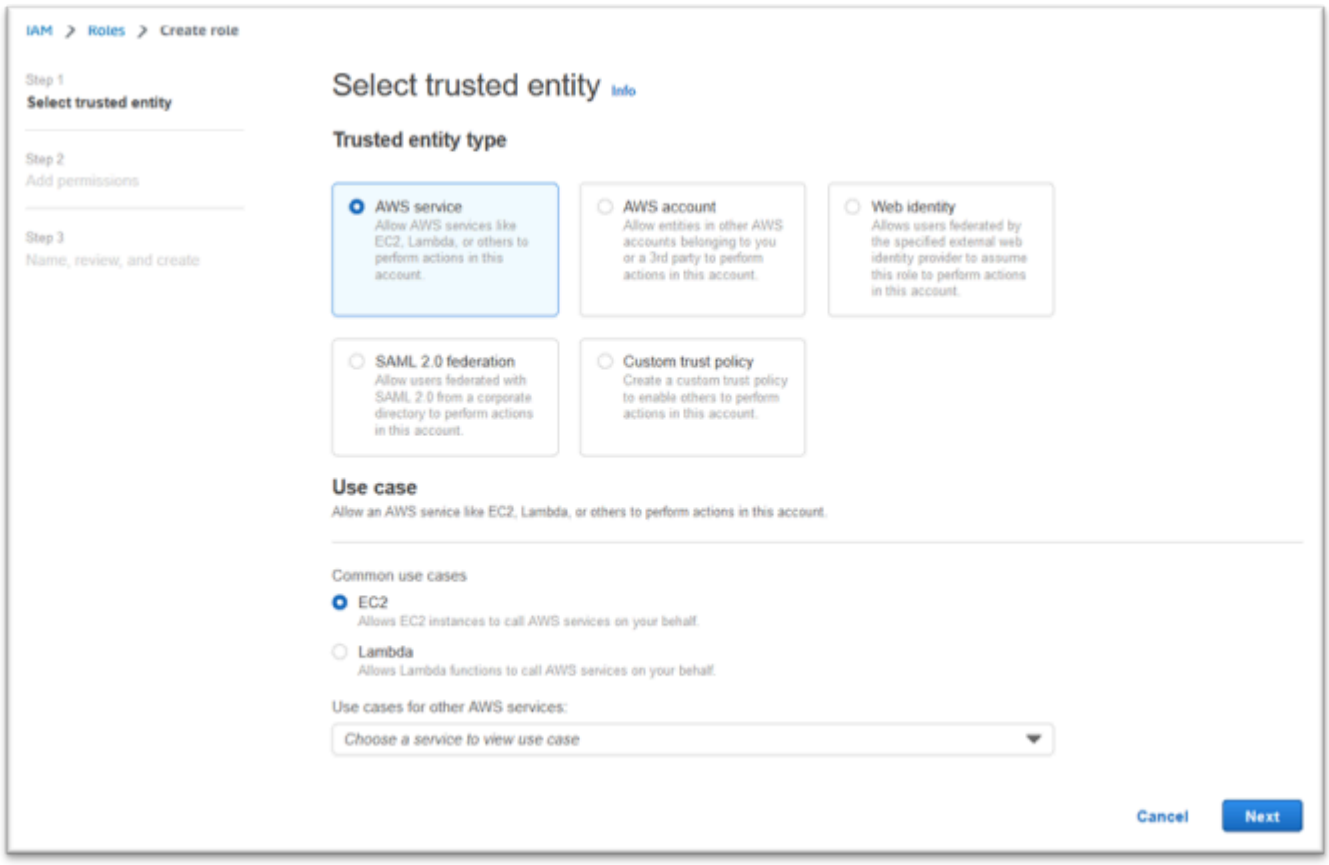
8. Escolha Create Policy.

Criar o perfil do IAM

1. Navegue até IAM no AWS Management Console.
2. Escolha Perfis > Criar perfil.

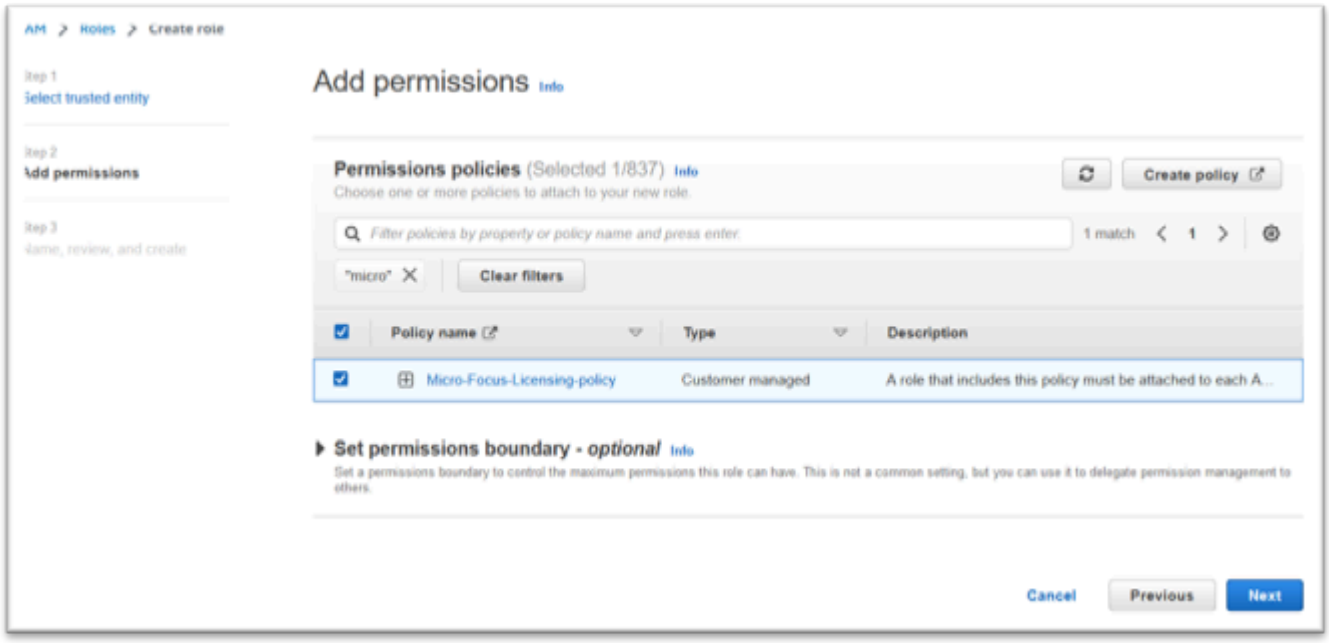


3. Deixe o tipo de entidade confiável como serviço AWS e escolha o caso de uso comum do EC2.

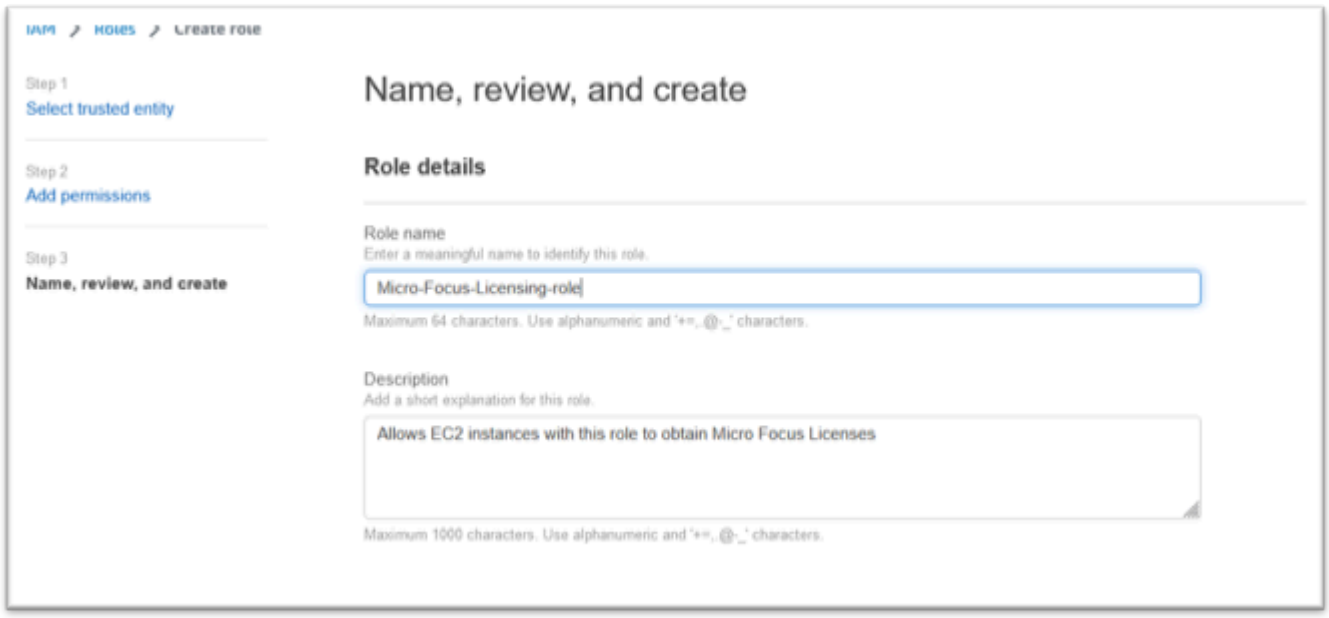


4. Escolha Próximo.
5. Digite “Micro” no filtro e pressione enter para aplicar o filtro.

6. Escolha a política que acabou de ser criada, por exemplo, a “Política de licenciamento da Micro-Focus”.
7. Escolha Próximo.



8. Insira o nome da função, por exemplo, “Micro-Focus-Licensing-Role”.
9. Substitua a descrição por uma de sua preferência, por exemplo, “Permite que instâncias do Amazon EC2 com essa função obtenham licenças da Micro Focus”.

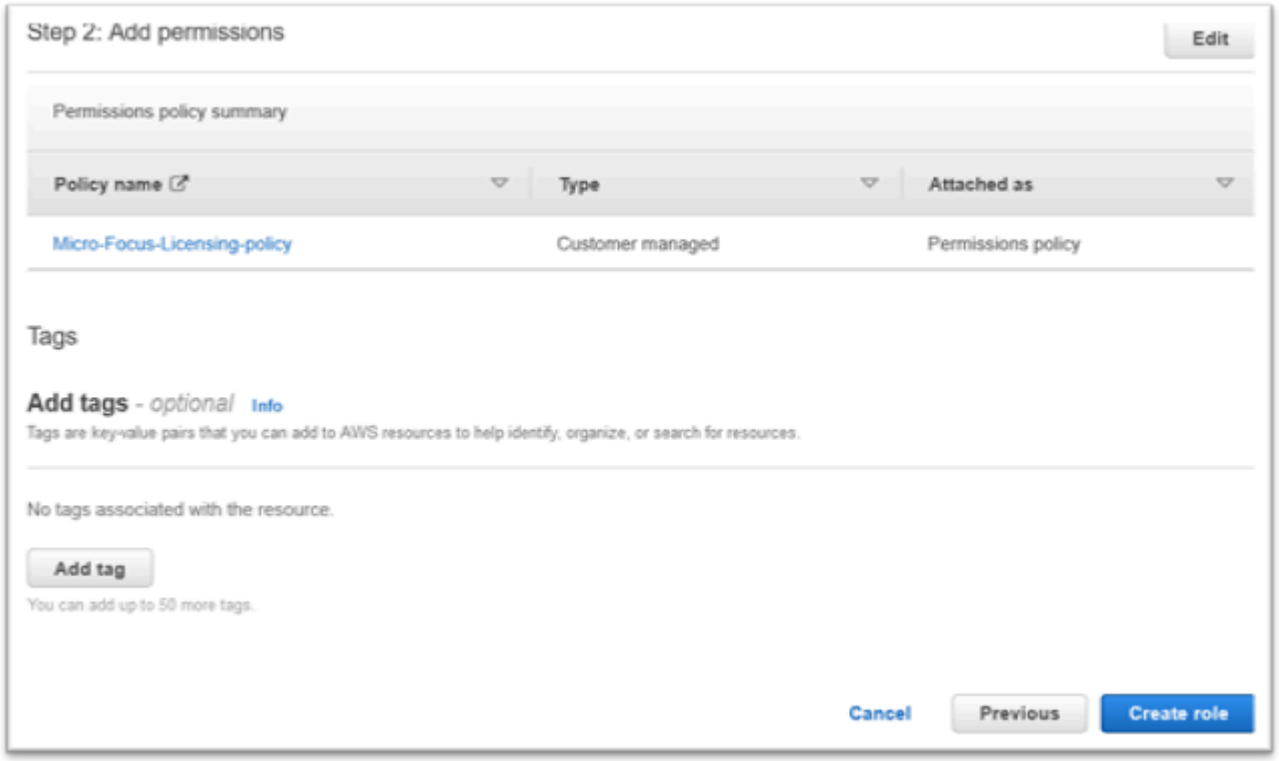


10. Em Etapa 1: selecionar entidades confiáveis, revise o JSON e confirme se ele tem os seguintes valores:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
            ],
            "Principal": {
                "Service": [
                    "ec2.amazonaws.com"
                ]
            }
        }
    ]
}
```

> **Note**
>
> A ordem do Efeito, da Ação e do Principal não é significativa.

11. Confirme se a Etapa 2: adicionar permissões mostra sua política de licenciamento.

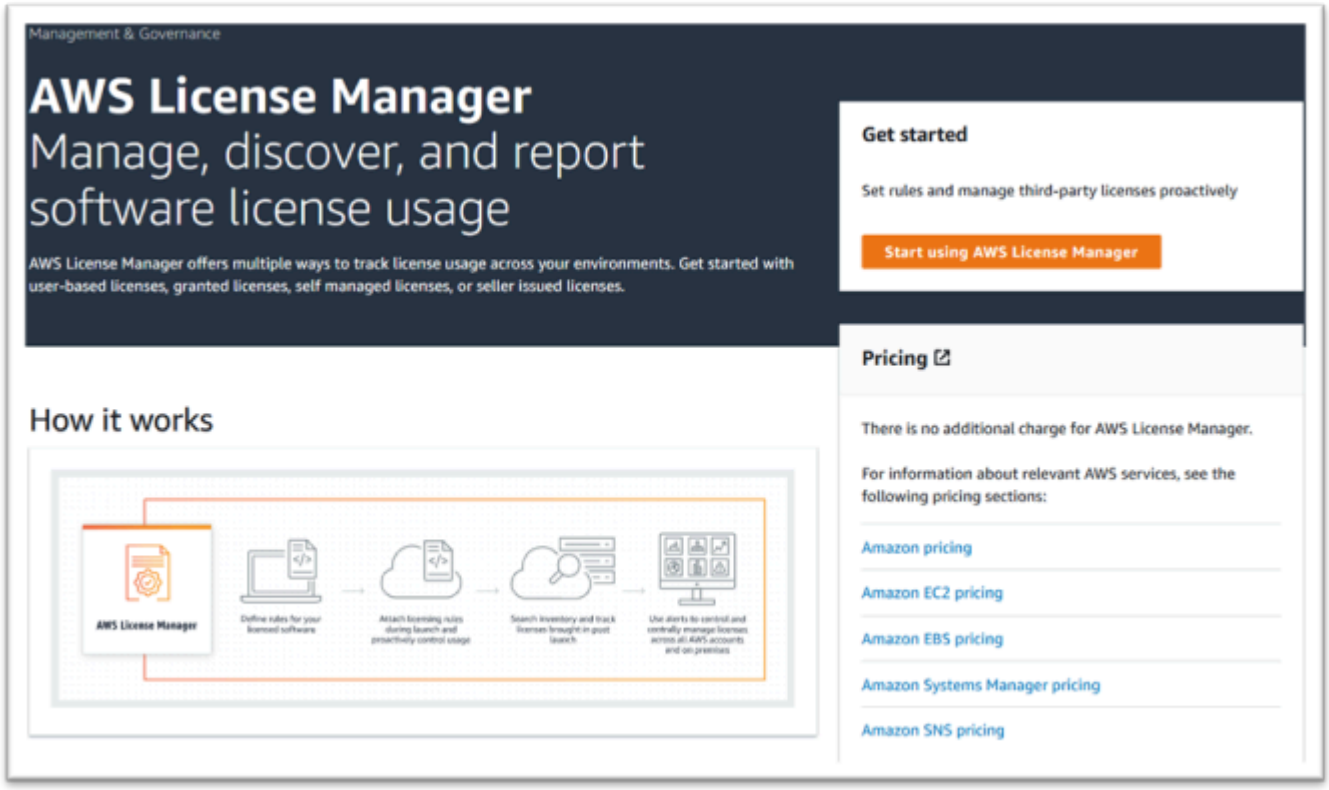


12. Selecione Criar função.

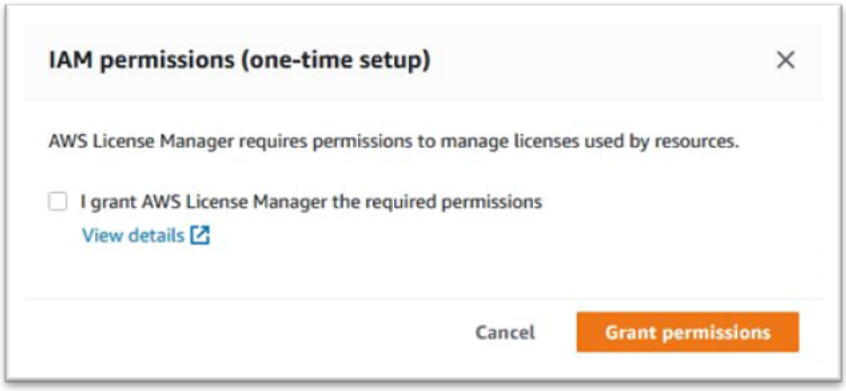
Depois que a solicitação da lista de permissões for concluída, continue com as etapas a seguir.

Conceda ao License Manager as permissões necessárias

1. Navegue até AWS License Manager no AWS Management Console.



2. Escolha Começar a usar o AWS License Manager.
3. Se você ver o pop-up a seguir, veja os detalhes, escolha a caixa de seleção e pressione Conceder Permissões.

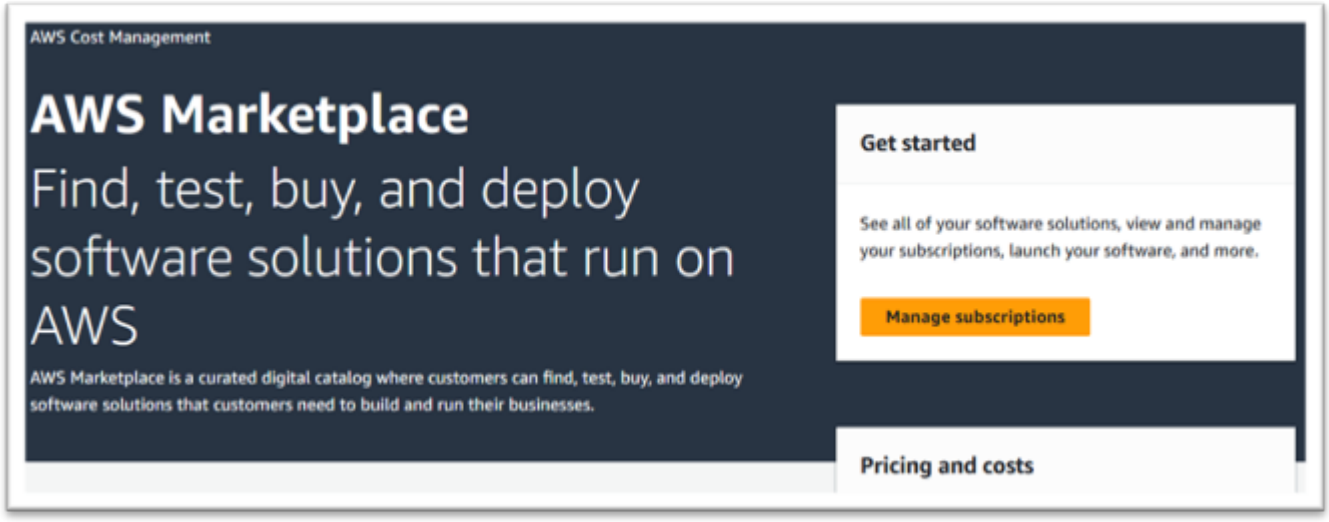


Inscreva-se para receber as imagens de máquina da Amazon

Depois de assinar um AWS Marketplace produto, você pode iniciar uma instância a partir da AMI do produto.

1. Navegue até AWS Marketplace Assinaturas no. AWS Management Console

2. Escolha Manage subscriptions (Gerenciar assinaturas).



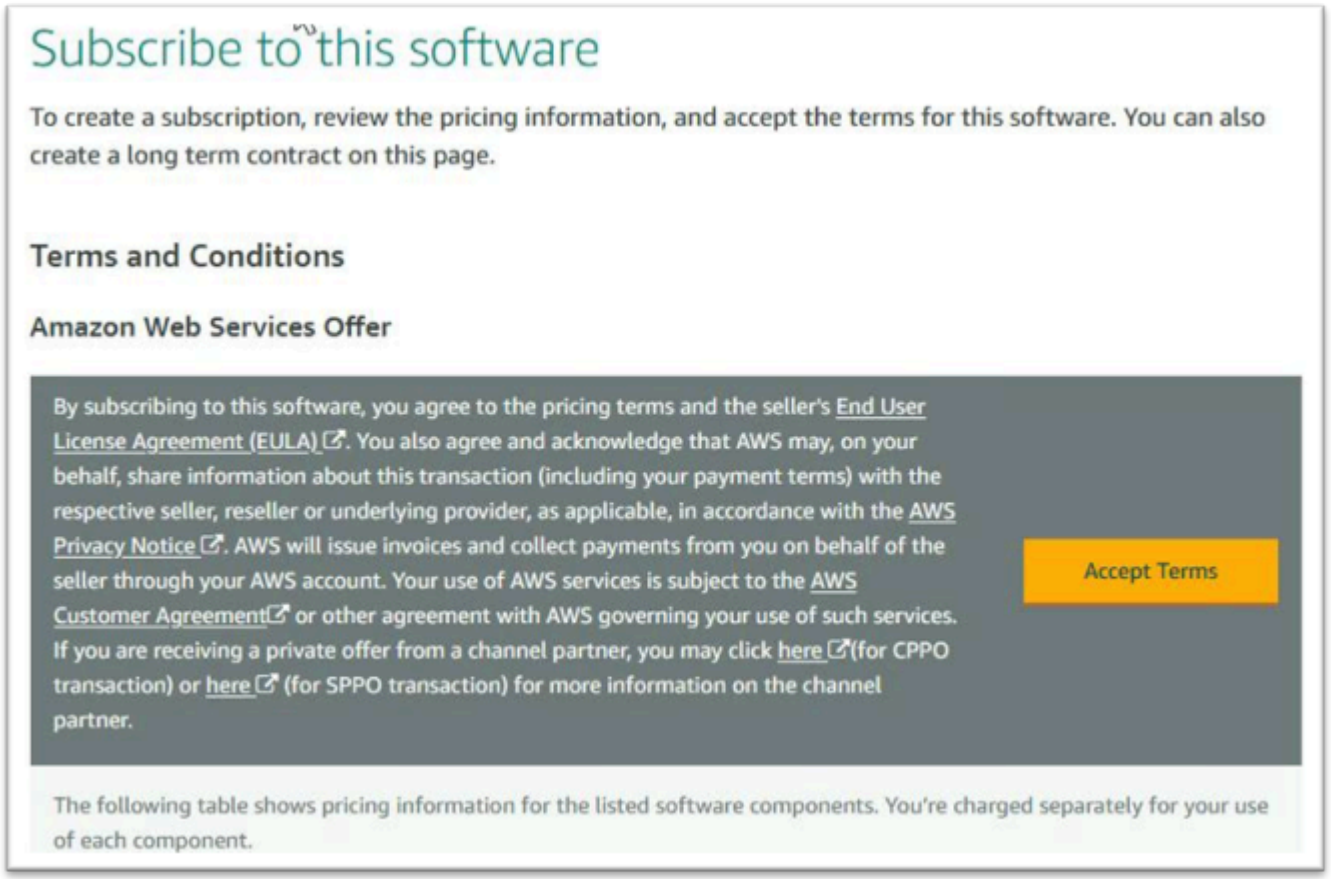
3. Copie e cole um dos links a seguir na barra de endereço do navegador.

> **Note**
>
> Escolha apenas um link para um dos produtos que você foi autorizado a usar.

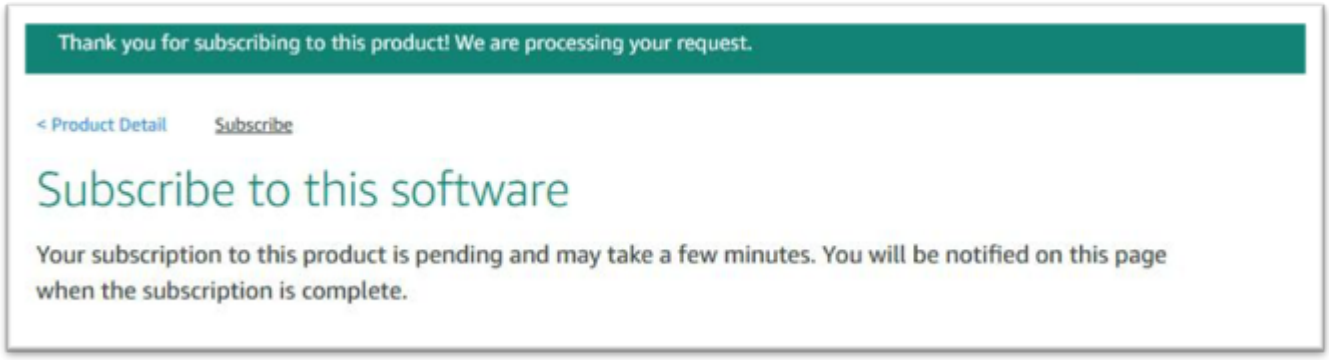
- Servidor corporativo: https://aws.amazon.com/marketplace/pp/prodview-g5emev63l7blc
- Servidor corporativo para Windows: https://aws.amazon.com/marketplace/pp/prodview-lwybsiyikbhc2
- Desenvolvedor corporativo: https://aws.amazon.com/marketplace/pp/prodview-77qmpr42yzxwk
- Desenvolvedor corporativo com Visual Studio 2022: https://aws.amazon.com/marketplace/pp/prodview-m4l3lqiszo6cm
- Analisador corporativo: https://aws.amazon.com/marketplace/pp/prodview-tttheylcmcihm
- Ferramentas de compilação empresarial para Windows: https://aws.amazon.com/marketplace/pp/prodview-2rw35bbt6uozi
- Procedimentos armazenados corporativos: https://aws.amazon.com/marketplace/pp/prodview-zoeyqnsdsj6ha
- Procedimentos armazenados corporativos com o SQL Server 2019: https://aws.amazon.com/marketplace/pp/prodview-ynfklquwubnz4

4. Escolha Continue to Subscribe (Continuar para assinar).

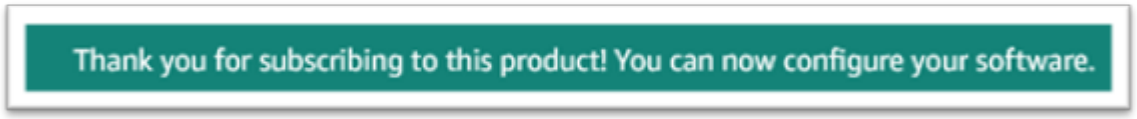
5. Se os Termos e Condições forem aceitáveis, escolha Aceitar Termos.



6. A assinatura pode levar alguns minutos para ser processada.

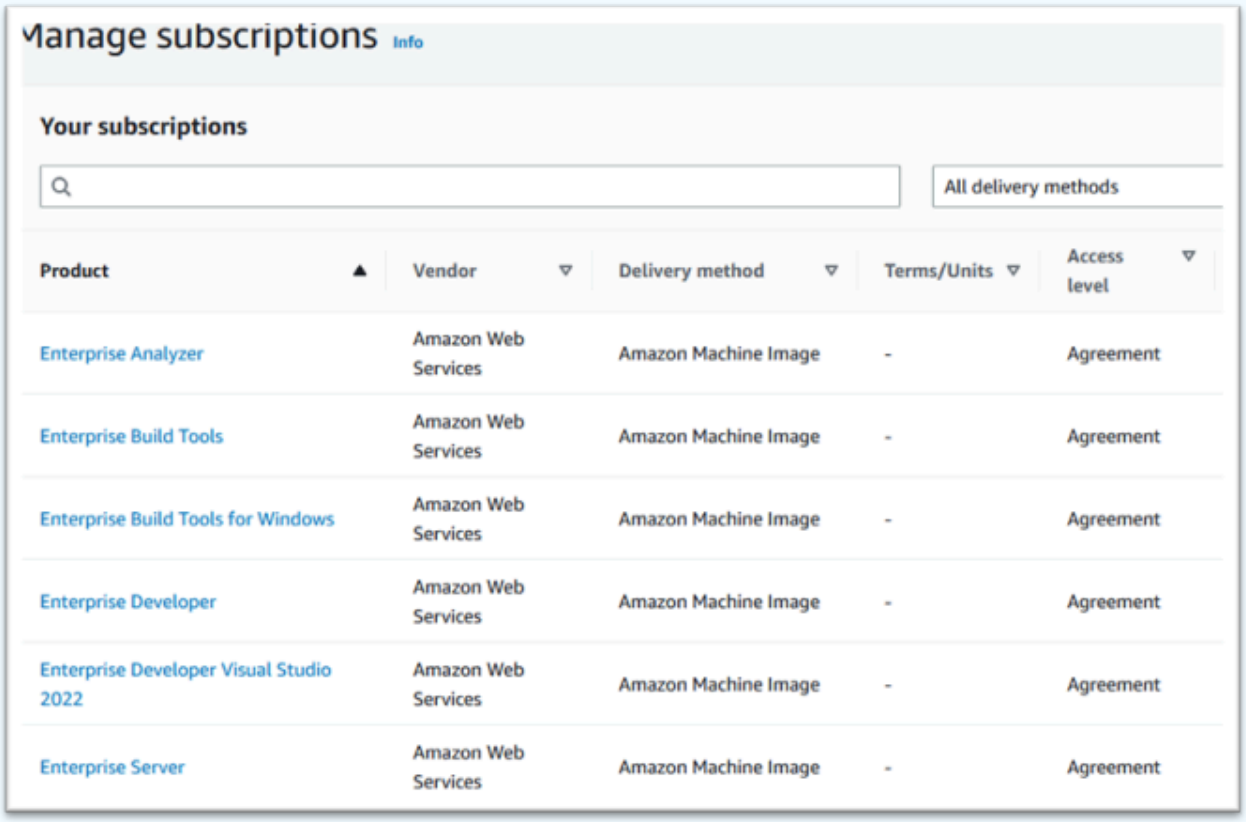


7. Depois que a mensagem de agradecimento for exibida, copie e cole o próximo link da etapa 3 para continuar adicionando assinaturas.



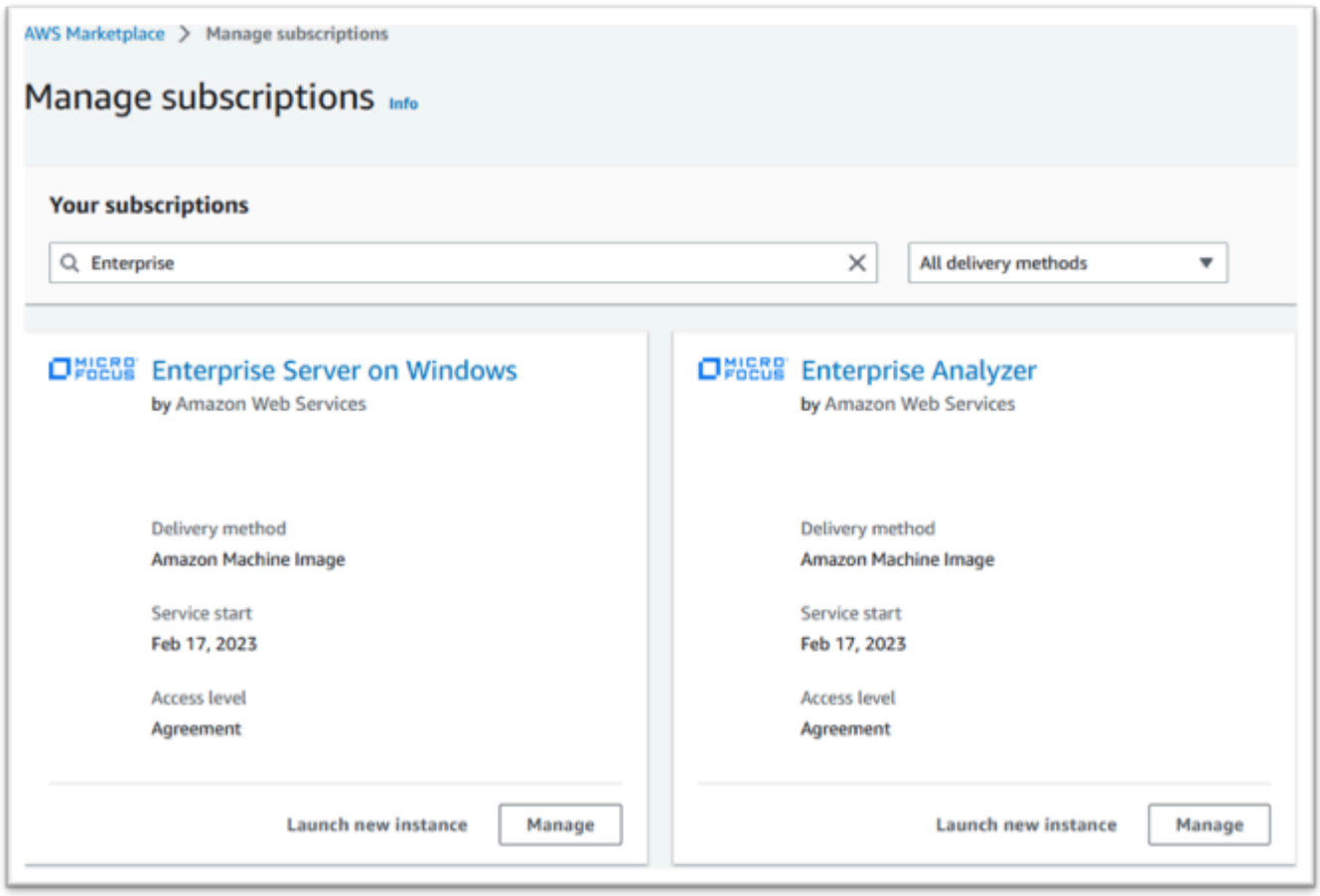
8. Pare quando Gerenciar assinaturas mostrar todas as suas AMIs inscritas.

> **Note**
>
> As preferências do painel (ícone de engrenagem) estão configuradas para mostrar a exibição como uma tabela.



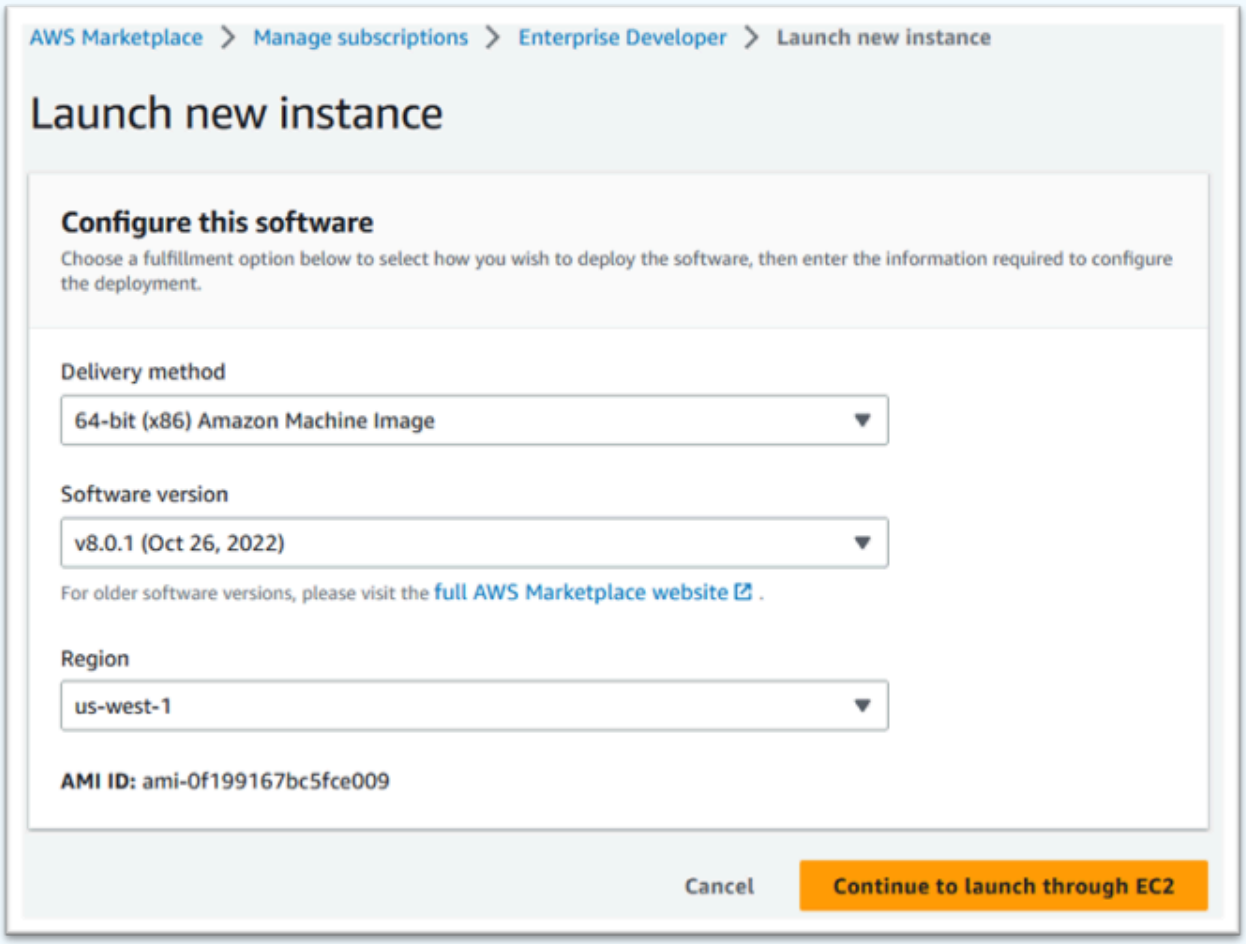
Inicie uma instância Micro Focus de modernização de AWS mainframe

1. Navegue até AWS Marketplace Assinaturas no. AWS Management Console
2. Localize a AMI a ser executada e escolha Iniciar nova instância.



3. Na caixa de diálogo Iniciar nova instância, verifique se a região da lista de permissões está selecionada.
4. Pressione Continuar para iniciar por meio do EC2.

> **Note**
>
> O exemplo a seguir mostra o lançamento de uma AMI para desenvolvedores corporativos, mas o processo é o mesmo para todas as AMIs de modernização de AWS mainframe.

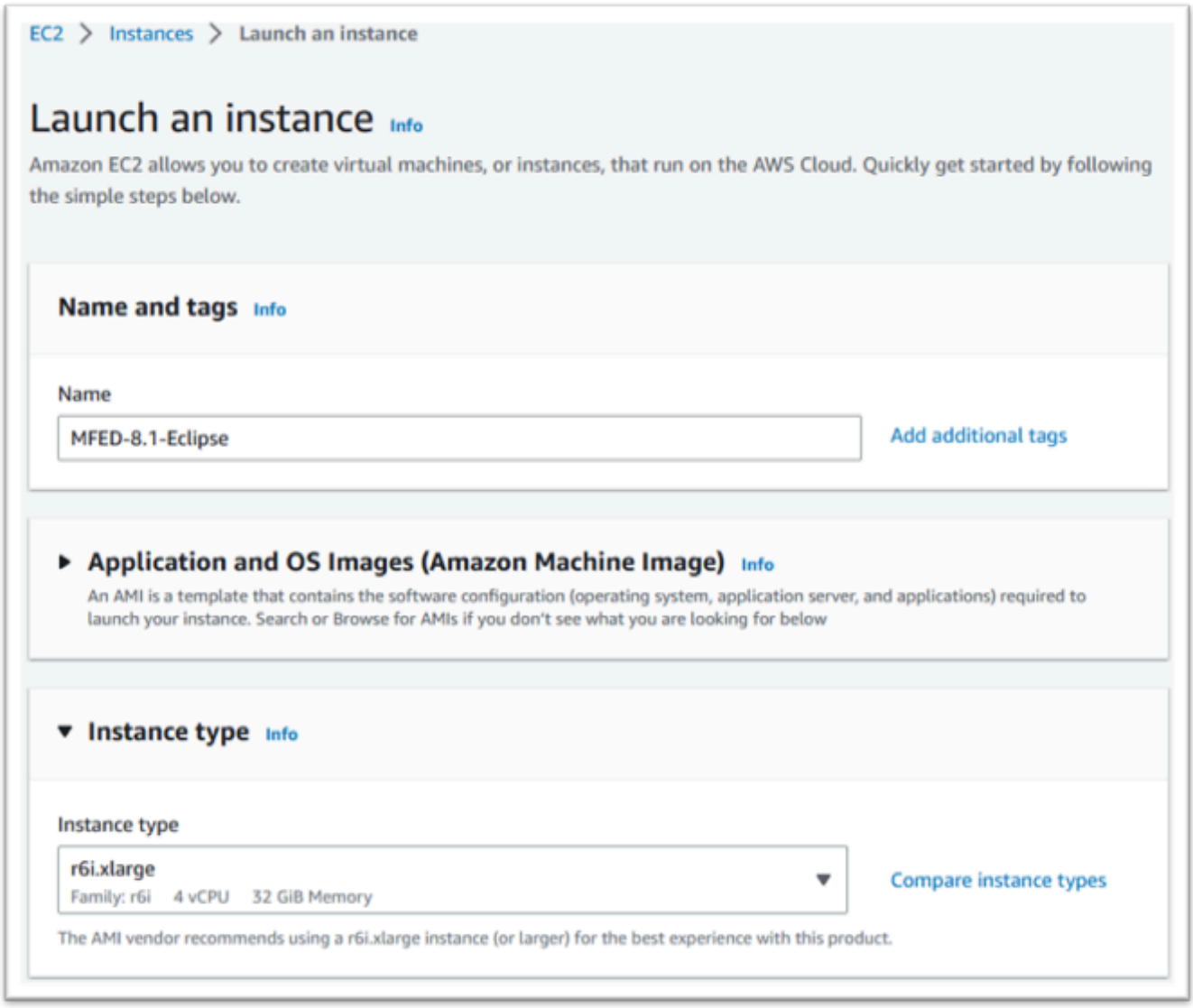


5. Digite um nome para o servidor.
6. Escolha um tipo de instância.

   O tipo de instância selecionado deve ser determinado pelo desempenho do projeto e pelos requisitos de custo. A seguir estão os pontos de partida sugeridos:

   - Para o Enterprise Analyzer, um r6i.xlarge
   - Para desenvolvedores corporativos, um r6i.large
   - Para uma instância autônoma do Enterprise Server, um r6i.xlarge
   - Para o Micro Focus Performance Availability Cluster (PAC) com escalabilidade horizontal, um r6i.large

> **Note**
>
> A seção Imagens da aplicação e do sistema operacional foi reduzida para a captura de tela.



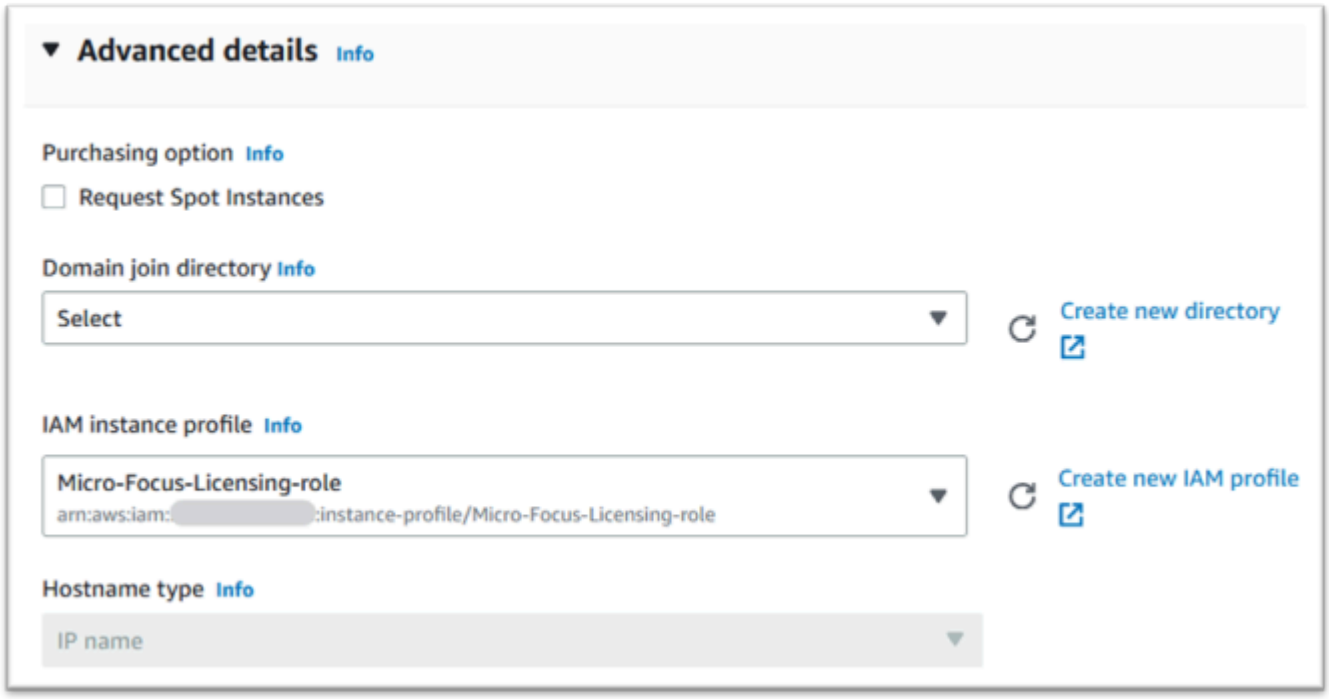
7. Escolha ou crie (e salve) um par de chaves (não exibido).

   Para obter mais informações sobre pares de chaves para instâncias do Linux, consulte Pares de chaves do Amazon EC2 e instâncias do Linux.

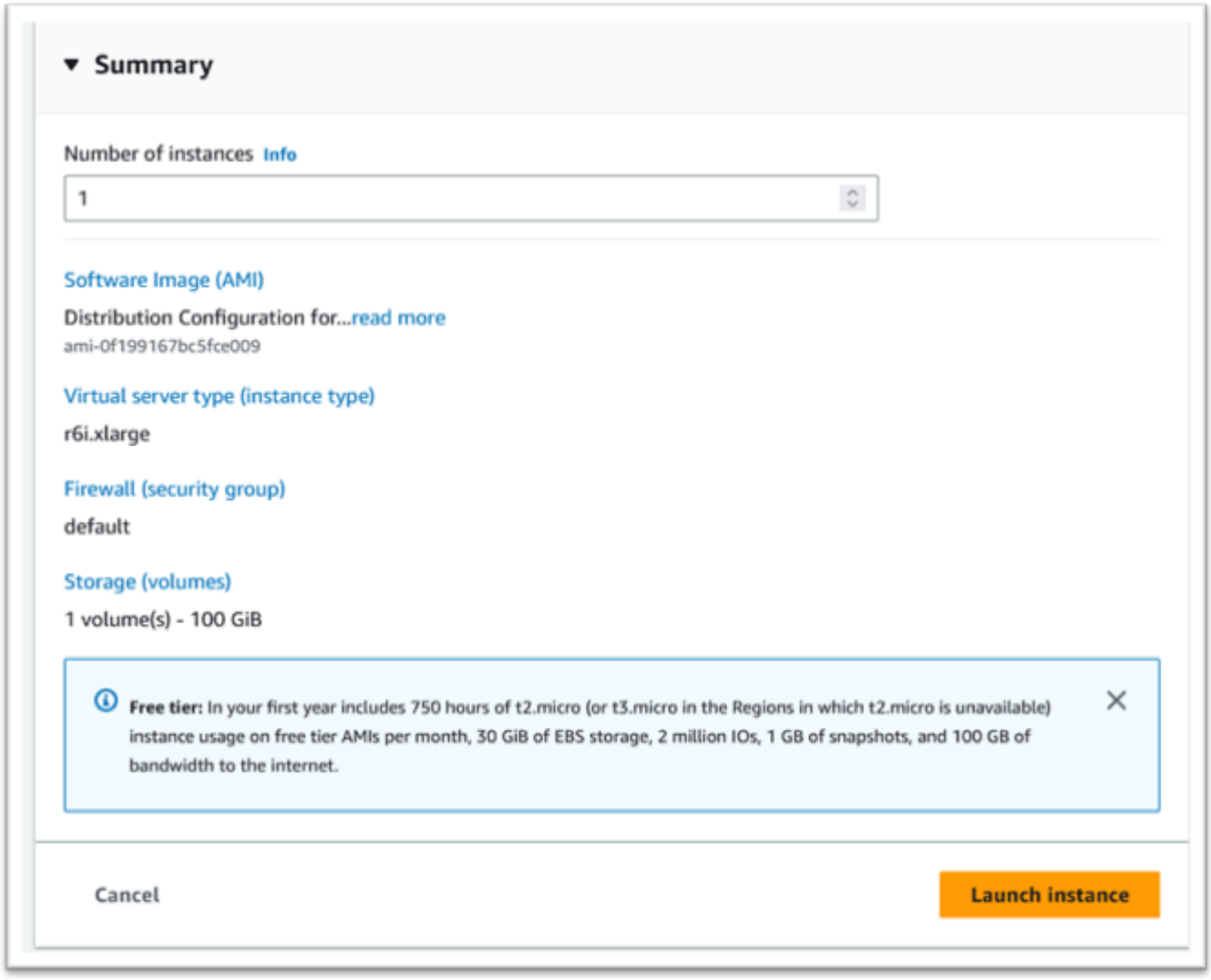
Para obter mais informações sobre pares de chaves para instâncias do Windows, consulte Pares de chaves do Amazon EC2 e instâncias do Windows.

8. Edite as configurações de rede e escolha a VPC permitida e a sub-rede apropriada.
9. Escolha Criar um novo grupo de segurança. Se for uma instância do Enterprise Server EC2, é normal permitir o tráfego TCP para as portas 86 e 10086 para administrar a configuração da Micro Focus.
10. Opcionalmente, configure o armazenamento para a instância do Amazon EC2.
11. Importante: expanda os detalhes avançados e, em Perfil da instância do IAM, escolha a função de licenciamento criada anteriormente, por exemplo, “Micro-Focus-Licensing-Role”.

> **Note**
>
> Se essa etapa for perdida, depois que a instância for criada, você poderá modificar o perfil do IAM na opção Segurança do menu Ação da instância EC2.

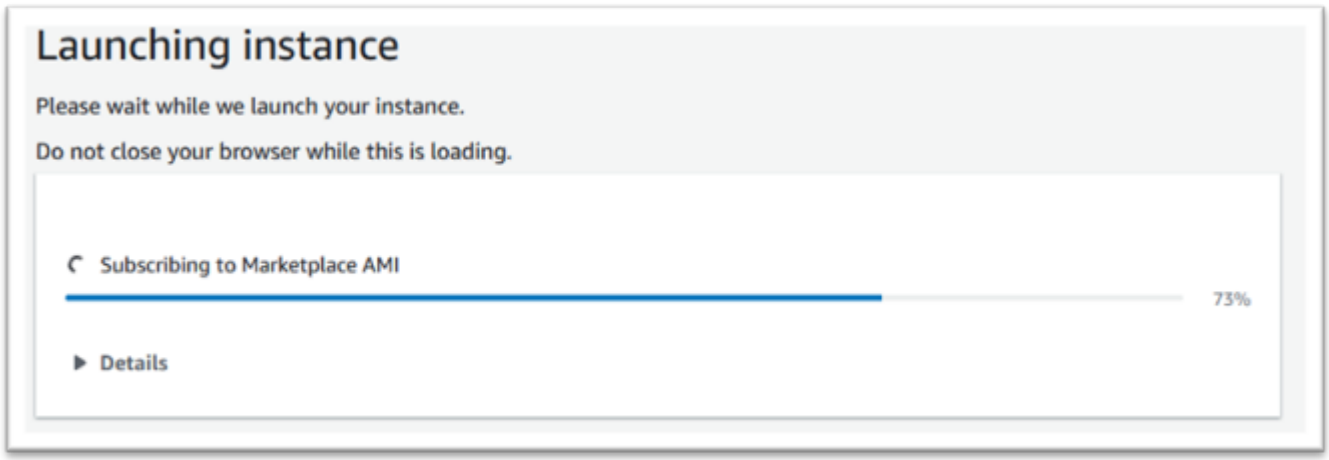


12. Revise o resumo e envie a Iniciar instância.

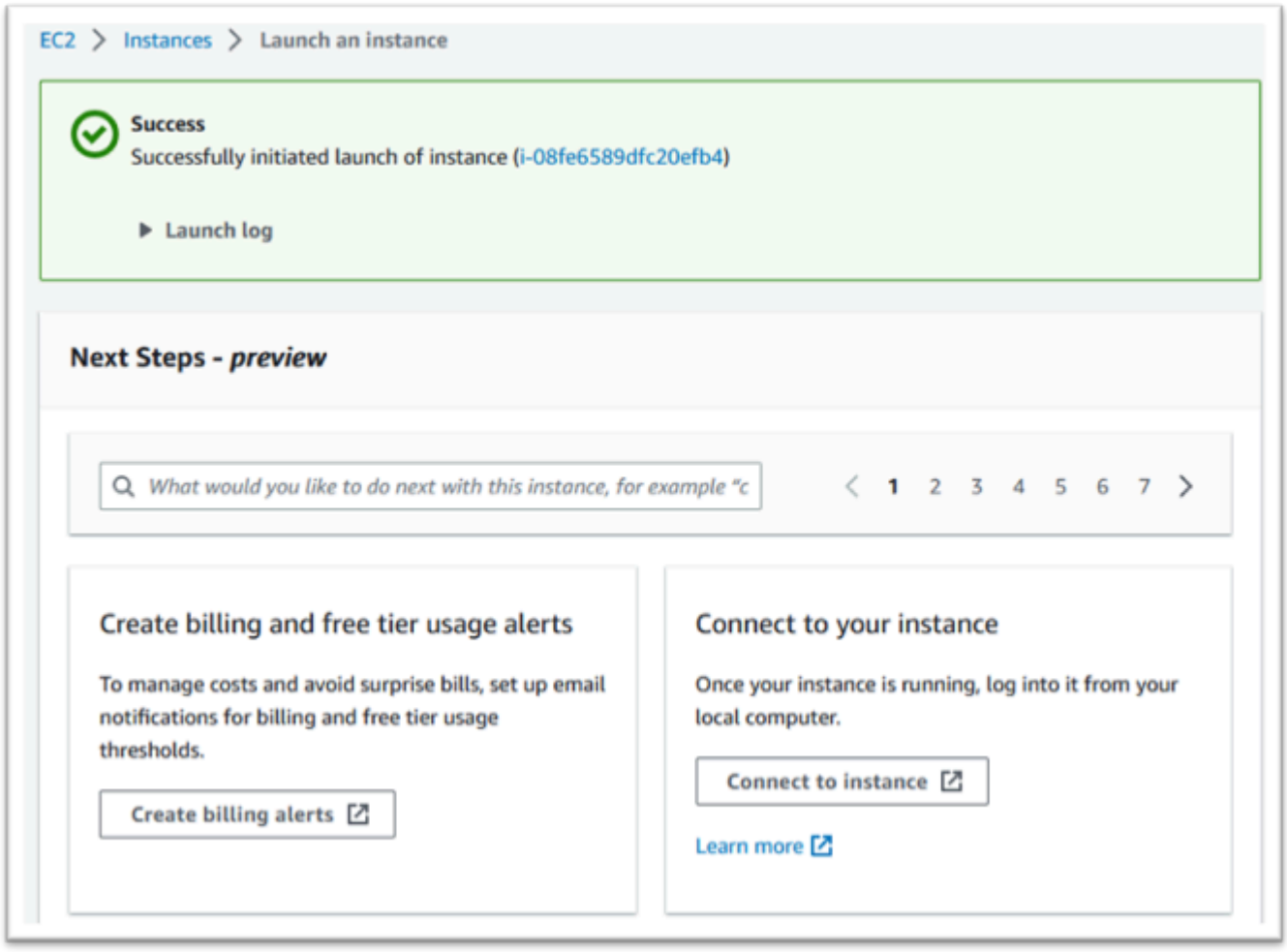


13. A inicialização da instância falhará se um tipo de servidor virtual inválido for escolhido.

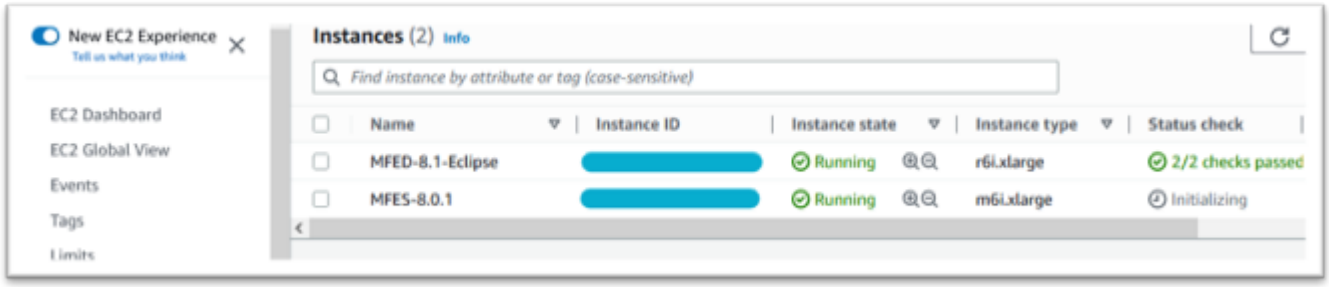
    Se isso acontecer, escolha Editar configuração da instância e altere o tipo de instância.



14. Depois que a mensagem “Sucesso” for exibida, escolha Conectar-se à instância para obter os detalhes da conexão.



15. Como alternativa, navegue até EC2 no AWS Management Console.
16. Escolha Instâncias para ver o status da nova instância.



Sub-rede ou VPC sem acesso à Internet

Faça essas alterações adicionais se a sub-rede ou a VPC não tiver acesso de saída à Internet.

O gerenciador de licenças exige acesso aos seguintes serviços da AWS:

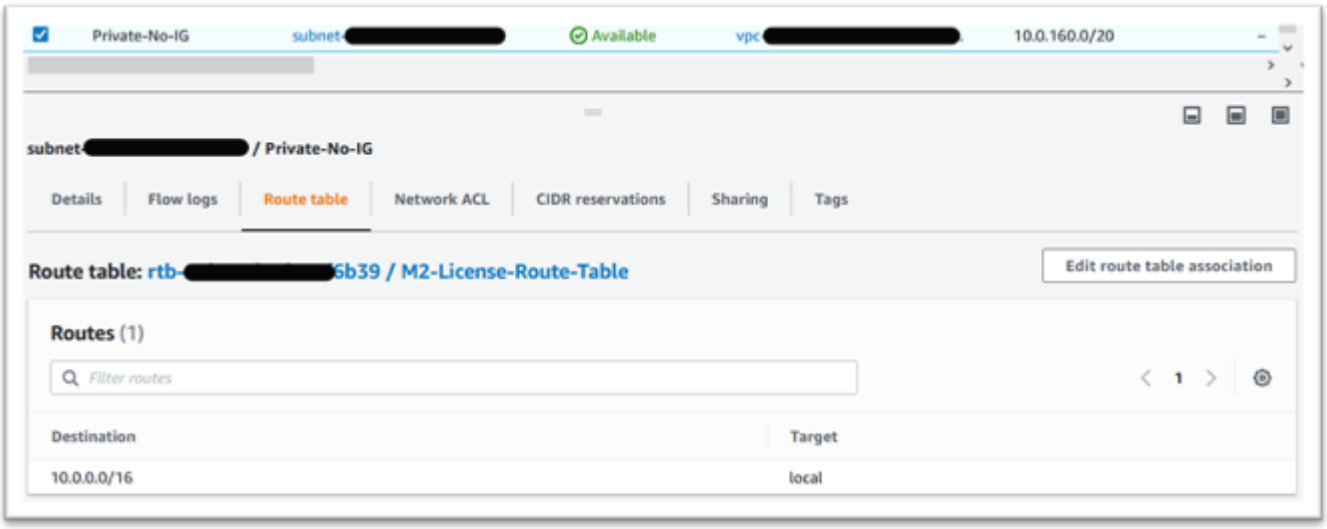
- com.amazonaws.*region*.s3
- com.amazonaws.*region*.ec2
- com.amazonaws.*region*.license-manager
- com.amazonaws.*region*.sts

As etapas anteriores definiram o serviço com.amazonaws.*region*.s3 como um endpoint de gateway. Esse endpoint precisa de uma entrada na tabela de rotas para qualquer sub-rede sem acesso à Internet.

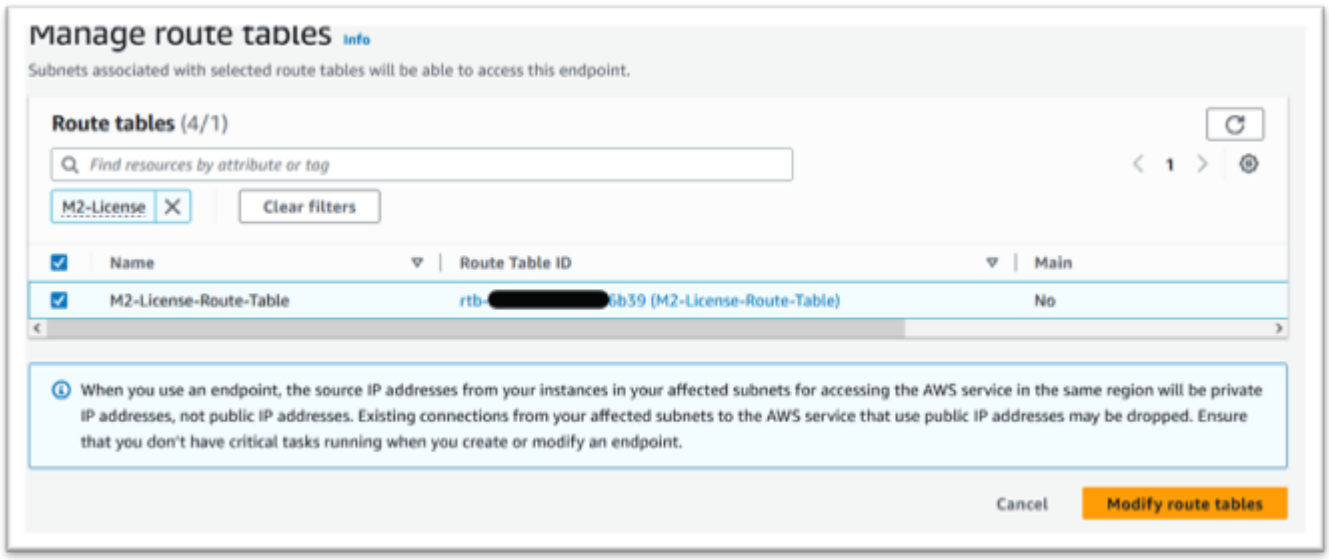
Os três serviços adicionais serão definidos como endpoints de interface.

Adicione a entrada da tabela de rotas para o endpoint do Amazon S3

1. Navegue até VPC em AWS Management Console e escolha Sub-redes.
2. Selecione a sub-rede na qual as instâncias do Amazon EC2 serão criadas e selecione a guia Tabela de rotas.
3. Observe alguns dígitos finais do ID da tabela de rotas. Por exemplo, o 6b39 na imagem abaixo.



4. No painel de navegação, escolha Endpoints.
5. Escolha o endpoint criado anteriormente e, em seguida, Gerenciar tabelas de rotas, na guia Tabelas de rotas do endpoint ou no menu suspenso Ações.
6. Escolha a tabela de rotas usando os dígitos identificados anteriormente e pressione Modificar tabelas de rotas.

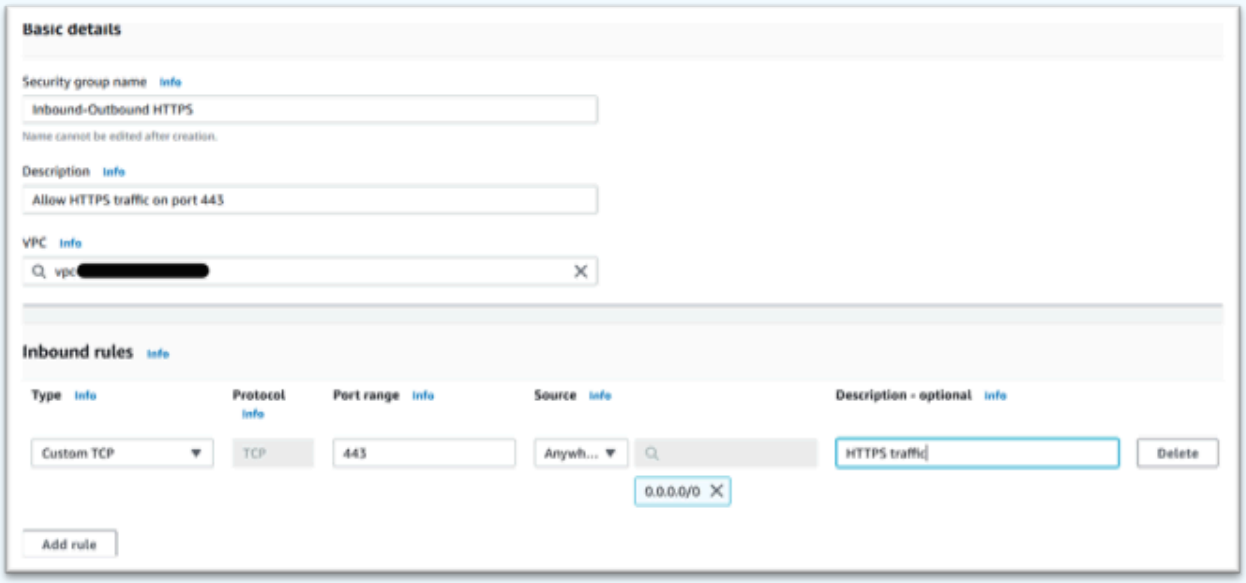


Defina o grupo de segurança necessário

Os serviços Amazon EC2 e License Manager se comunicam via HTTPS pela porta 443. AWS STS Essa comunicação é bidirecional e requer regras de entrada e saída para permitir que a instância se comunique com os serviços.

1. Navegue até a Amazon VPC no AWS Management Console.
2. Localize Grupos de segurança na barra de navegação e selecione Criar grupo de segurança.
3. Insira o nome e a descrição do grupo de segurança, por exemplo, “HTTPS de entrada e saída”.
4. Pressione o X na área de seleção da VPC para remover a VPC padrão e escolha a VPC que contém o endpoint do S3.
5. Adicione uma regra de entrada que permita o tráfego TCP na porta 443 de qualquer lugar.

> **Note**
>
> As regras de entrada (e saída) podem ser restringidas ainda mais limitando a Fonte. Para obter mais informações, consulte Controle o tráfego para seus AWS recursos usando grupos de segurança no Guia do usuário da Amazon VPC.

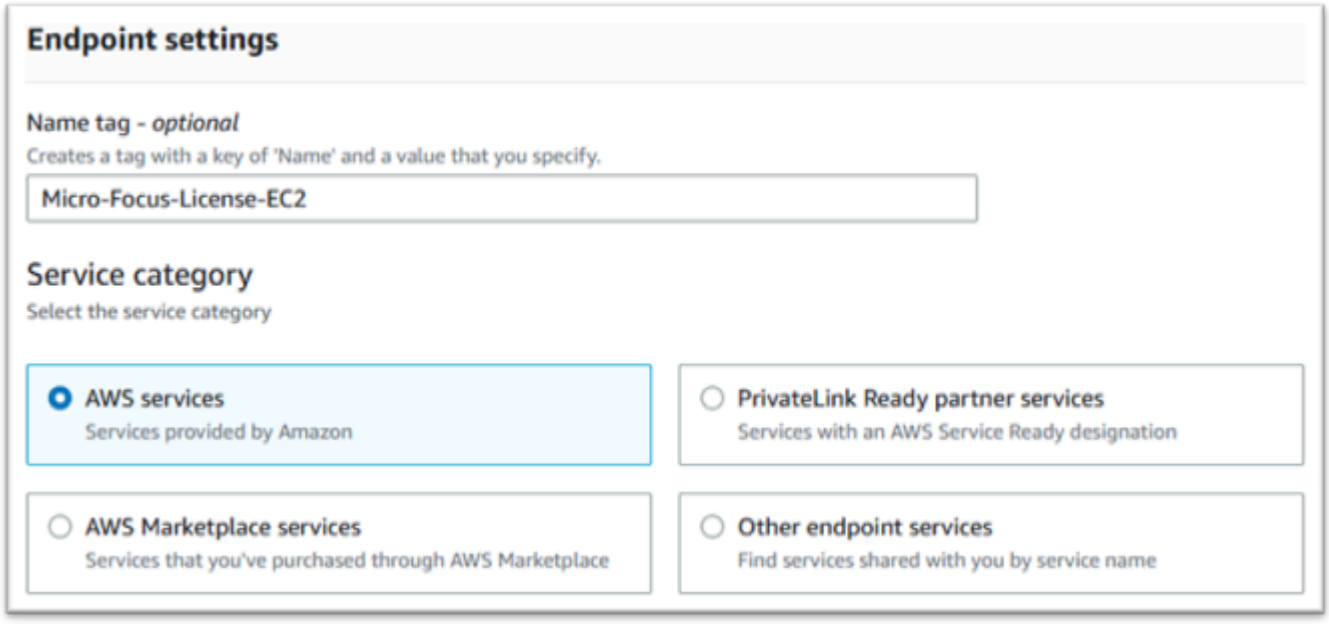


6. Pressione Criar grupo de segurança.

Criar os endpoints de serviço

Repita esse processo três vezes — uma vez para cada serviço.

1. Navegue até Amazon VPC em AWS Management Console e escolha Endpoints.
2. Pressione Criar endpoint.
3. Insira um nome, por exemplo, “Micro-Focus-License-EC2”, “Micro-Focus-License-STS” ou “Micro-Focus-License-Manager”.
4. Escolha a categoria de serviço AWS Services.



5. Em Serviços, procure o serviço de interface correspondente, que é um dos seguintes:

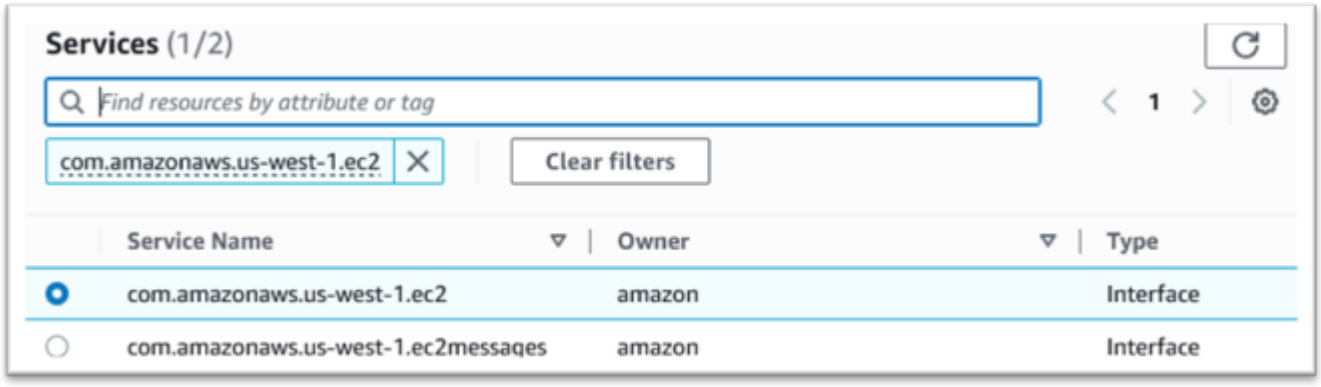
   - “com.amazonaws.*region*.ec2”
   - “com.amazonaws.*region*.sts”
   - “com.amazonaws.*region*.license-manager”

   Por exemplo: .

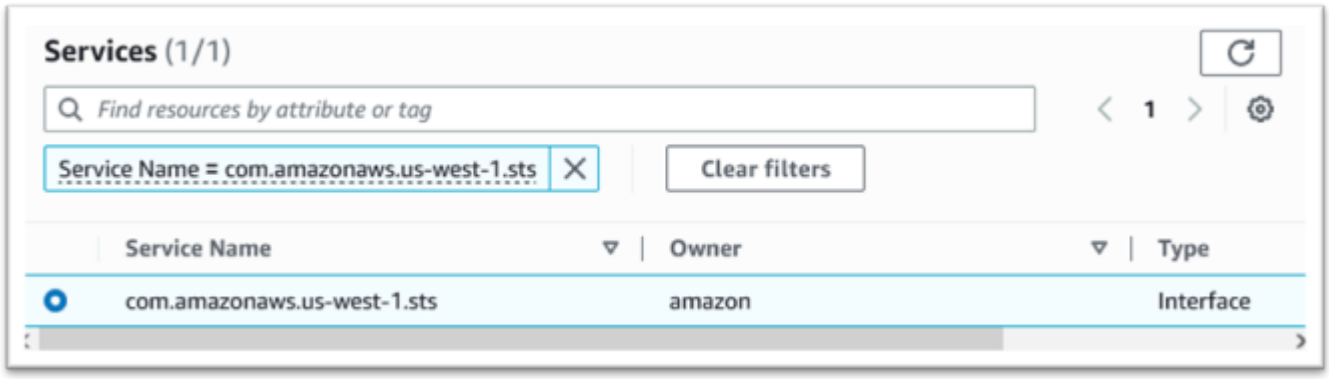
   - “com.amazonaws.us-west-1.ec2”
   - “com.amazonaws.us-west-1.sts”
   - “com.amazonaws.us-west-1.license-manager”

6. Escolha o serviço de interface correspondente.

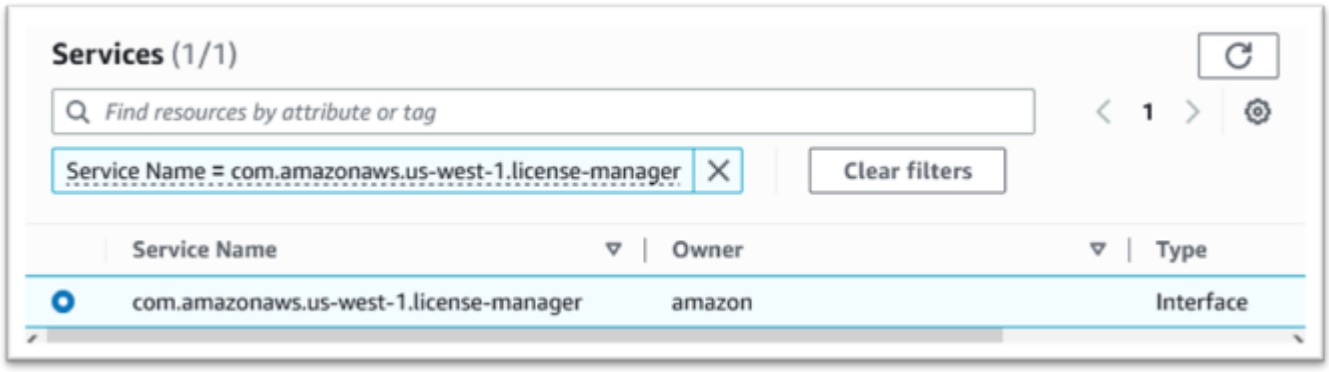
   com.amazonaws.***region***.ec2:



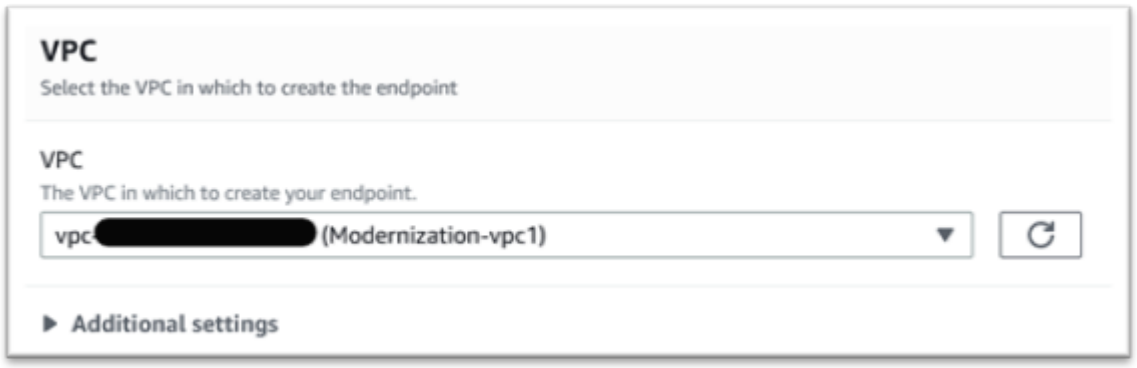
com.amazonaws.***region***.sts:



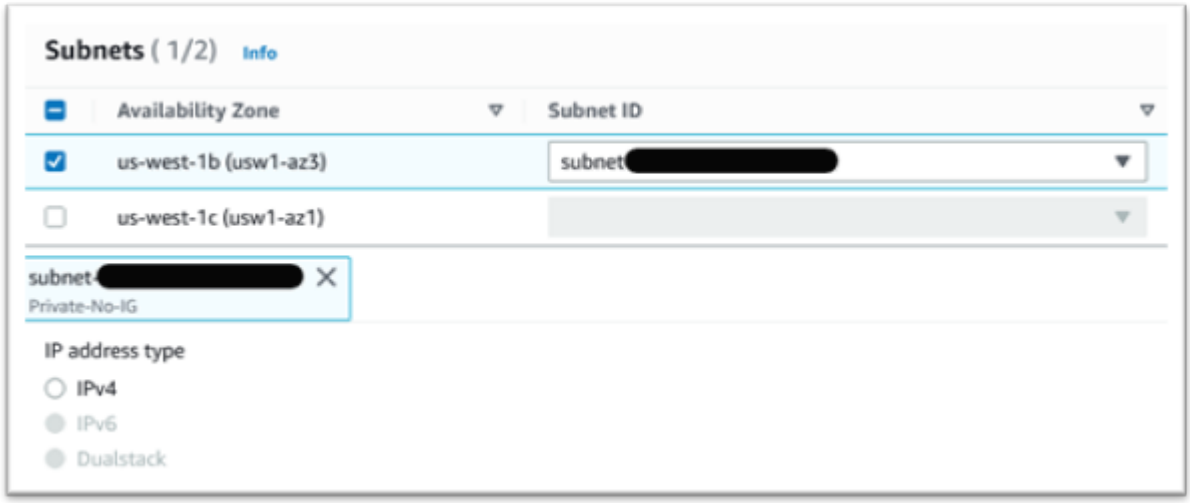
com.amazonaws.***region***.license-manager:



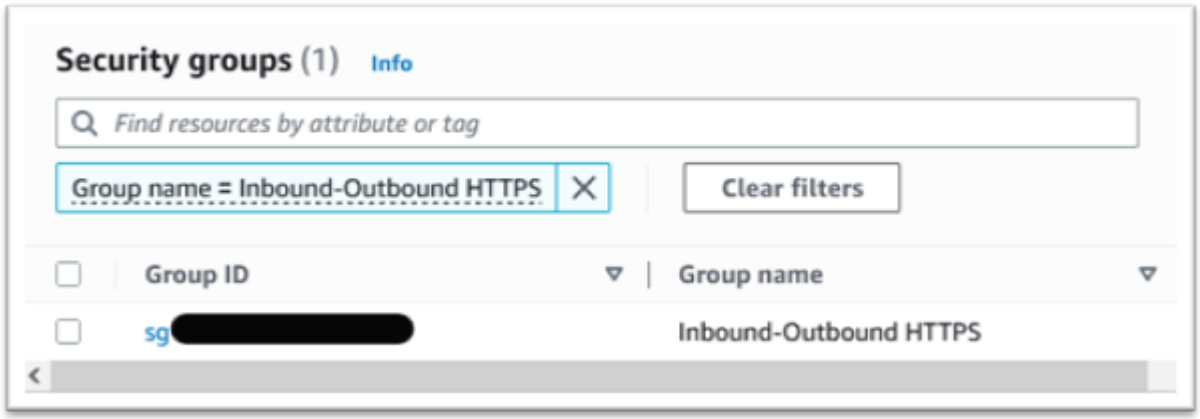
7. Para VPC, escolha a VPC para a instância.



8. Escolha a Zona de disponibilidade e as Sub-redes para a VPC.



9. Escolha Criar para criar o grupo de segurança.



10. Em Política, escolha Acesso total.

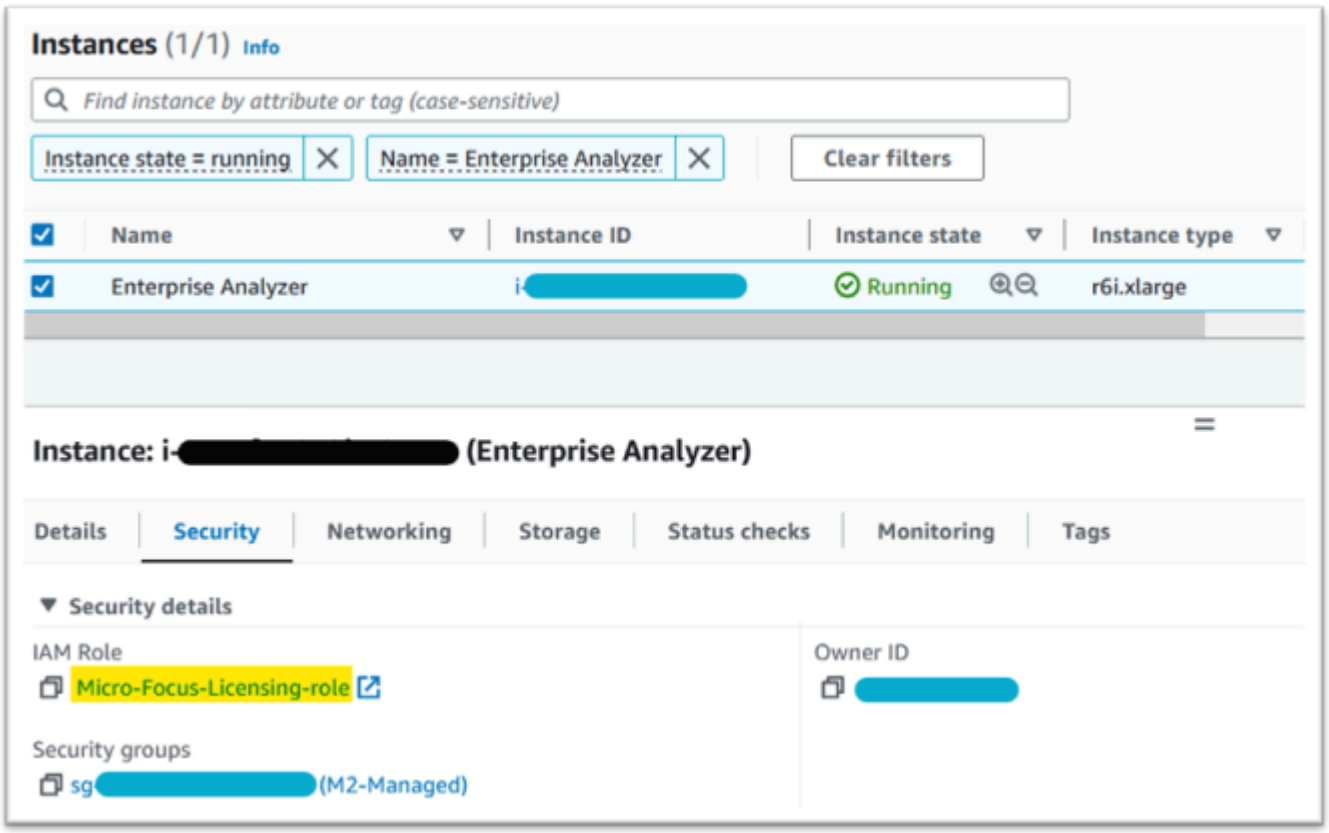
11. Escolha Criar Endpoint.
12. Repita este processo para as interfaces restantes.

Solução de problemas de licença

Se você tiver problemas para acessar ou usar as AMIs, as informações a seguir podem ajudá-lo.

Verifique se a instância do Amazon EC2 tem a função de licenciamento do IAM

Isso pode ser verificado na guia Segurança dos detalhes da instância do Amazon EC2. Isso pode ser alterado usando a Opção de Segurança do menu suspenso Ações.



Use o Reachability Analyzer

Encontre o Reachability Analyzer na página do console. AWS Network Manager

Crie e analise um caminho entre a instância do Amazon EC2 criada a partir da AMI e o endpoint da Amazon S3 VPC.

Se a instância do Amazon EC2 não tiver acesso à Internet, repita a análise do caminho para todos os 4 endpoints.

Para obter mais informações sobre o Reachability Analyzer, consulte Introdução ao Reachability Analyzer no guia do Reachability Analyzer.

Execute o daemon de licença

No Windows Enterprise Developer, use o seguinte comando em um prompt de comando:

```
“C:\Program Files (x86)\Micro Focus\Enterprise Developer\AdoptOpenJDK\bin\java” -jar
 "C:\Program Files (x86)\Micro Focus\Licensing\aws-license-daemon.jar"
```

e examine a saída. Ignore as mensagens do SLF4J e procure a primeira exceção.

No Enterprise Analyzer, use o seguinte comando em um prompt de comando:

```
"C:\Program Files (x86)\Micro Focus\AdoptOpenJDK\bin\java" -jar "C:\Program Files
 (x86)\Micro Focus\Licensing\aws-license-daemon.jar"
```

e examine a saída. Ignore as mensagens do SLF4J e procure a primeira exceção.

No Linux, execute:

```
java -jar /var/microfocuslicensing/bin/aws-license-daemon.jar
```

Ignore as mensagens do SLF4J e procure a primeira exceção.

Por exemplo, se o recurso Amazon S3 não estiver disponível, a exceção será a seguinte:

```
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.

Exception in thread "main" software.amazon.awssdk.services.s3.model.S3Exception: Access
 Denied (Service: S3, Status Code: 403, Request ID: P6
```

A mensagem de exceção indica qual recurso não está disponível. Compare os valores de configuração com os mostrados neste tópico.

Tutorial: Configure a AppStream versão 2.0 para uso com o Micro Focus Enterprise Analyzer e o Micro Focus Enterprise Developer

AWS A modernização do mainframe fornece várias ferramentas por meio do Amazon AppStream 2.0. AppStream 2.0 é um serviço de streaming de aplicativos totalmente gerenciado e seguro que permite transmitir aplicativos de desktop para usuários sem reescrever aplicativos. AppStream O 2.0 fornece aos usuários acesso instantâneo aos aplicativos de que precisam, com uma experiência de usuário responsiva e fluida no dispositivo de sua escolha. O uso da AppStream versão 2.0 para hospedar ferramentas específicas do Runtime Engine oferece às equipes de aplicativos do cliente a capacidade de usar as ferramentas diretamente de seus navegadores da web, interagindo com arquivos de aplicativos armazenados em buckets ou repositórios do Amazon S3. CodeCommit

Para obter informações sobre o suporte a navegadores na AppStream versão 2.0, consulte System Requirements and Feature Support (Web Browser) no Amazon AppStream 2.0 Administration Guide. Se você tiver problemas ao usar a AppStream versão 2.0, consulte Solução de problemas do usuário AppStream 2.0 no Guia de administração da Amazon AppStream 2.0.

Este documento é destinado aos membros da equipe de operações do cliente. Ele descreve como configurar frotas e pilhas do Amazon AppStream 2.0 para hospedar as ferramentas Micro Focus Enterprise Analyzer e Micro Focus Enterprise Developer usadas com AWS a modernização do mainframe. O Micro Focus Enterprise Analyzer geralmente é usado durante a fase de avaliação e o Micro Focus Enterprise Developer geralmente é usado durante a fase de migração e modernização da abordagem de modernização do AWS mainframe. Se você planeja usar o Enterprise Analyzer e o Enterprise Developer, deve criar frotas e pilhas separadas para cada ferramenta. Cada ferramenta requer sua própria frota e pilha porque seus termos de licenciamento são diferentes.

> **Important**
>
> As etapas deste tutorial são baseadas no AWS CloudFormation modelo disponível para download cfn-m2- .yml. appstream-fleet-ea-ed

Tópicos

Pré-requisitos

- Faça o download do modelo: cfn-m2- appstream-fleet-ea-ed .yml.
- Obtenha o ID da sua VPC e do grupo de segurança padrão. Para obter mais informações sobre o VPC padrão, consulte VPCs padrão no Guia do usuário da Amazon VPC. Para obter mais

informações sobre o grupo de segurança padrão, consulte Grupos de segurança padrão e personalizados no Guia do usuário do Amazon EC2.

- Certifique-se de que você tenha as seguintes permissões:
  - crie pilhas, frotas e usuários na AppStream versão 2.0.
  - crie pilhas AWS CloudFormation usando um modelo.
  - crie buckets e faça upload de arquivos para buckets no Amazon S3.
  - baixe as credenciais (`access_key_id` e `secret_access_key`) do IAM.

Etapa 1: Obtenha as imagens AppStream 2.0

Nesta etapa, você compartilha as imagens AppStream 2.0 do Enterprise Analyzer e do Enterprise Developer com sua AWS conta.

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. Na navegação à esquerda, selecione Ferramentas.
3. Em Análise, desenvolvimento e criação de ativos, escolha Compartilhar ativos com minha AWS conta.

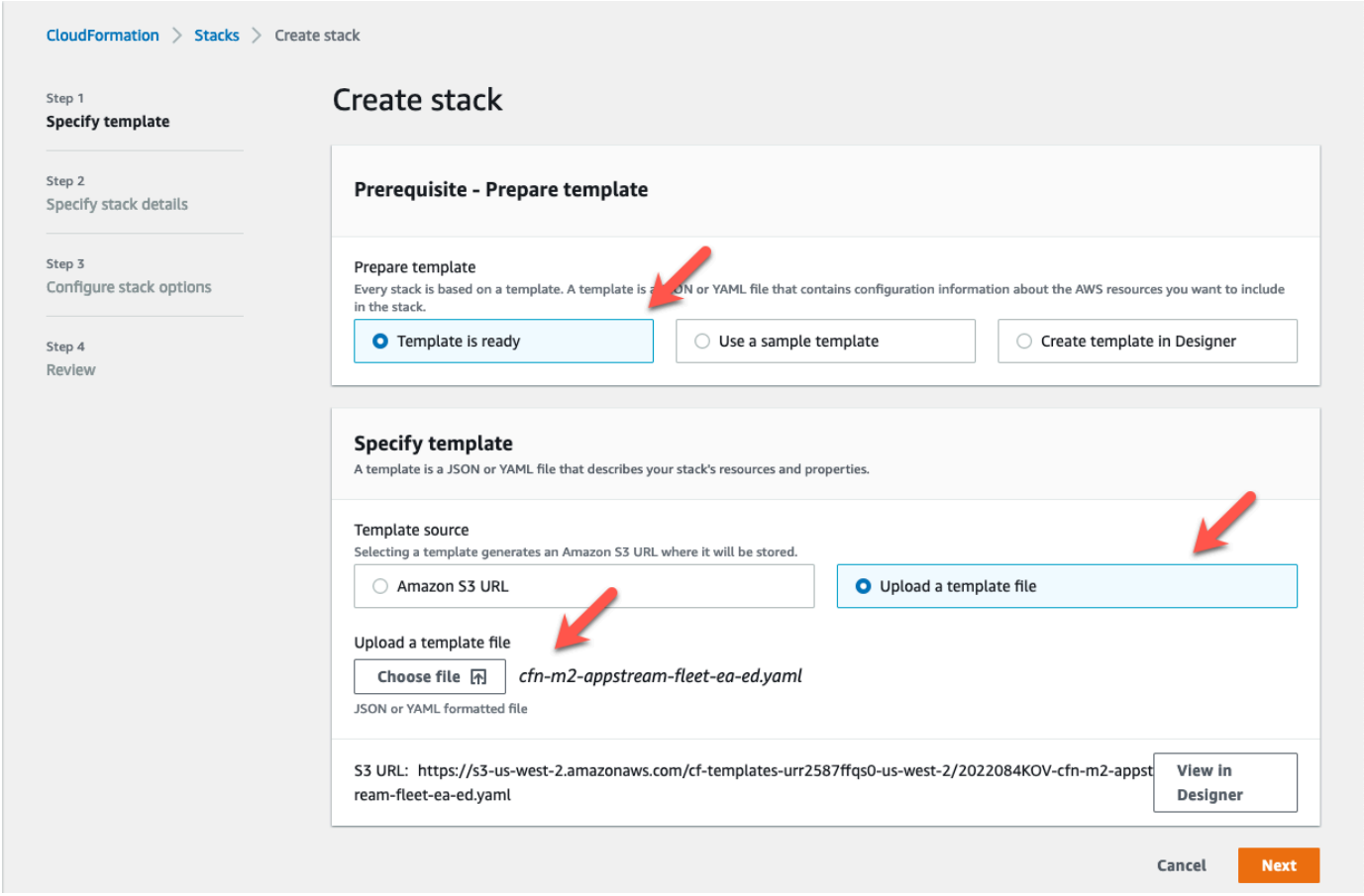
Etapa 2: criar a pilha usando o modelo AWS CloudFormation

Nesta etapa, você usa o AWS CloudFormation modelo baixado para criar uma pilha AppStream 2.0 e uma frota para executar o Micro Focus Enterprise Analyzer. Você pode repetir essa etapa posteriormente para criar outra pilha e frota AppStream 2.0 para executar o Micro Focus Enterprise Developer, já que cada ferramenta requer sua própria frota e pilha na AppStream versão 2.0. Para obter mais informações sobre AWS CloudFormation pilhas, consulte Como trabalhar com pilhas no Guia do AWS CloudFormation usuário.

> **Note**
>
> AWS A modernização do mainframe adiciona uma taxa adicional ao preço padrão AppStream 2.0 para o uso do Enterprise Analyzer e do Enterprise Developer. Para obter mais informações, consulte Preço do AWS Mainframe Modernization.

1. Faça o download do modelo cfn-m2- appstream-fleet-ea-ed .yml, se necessário.

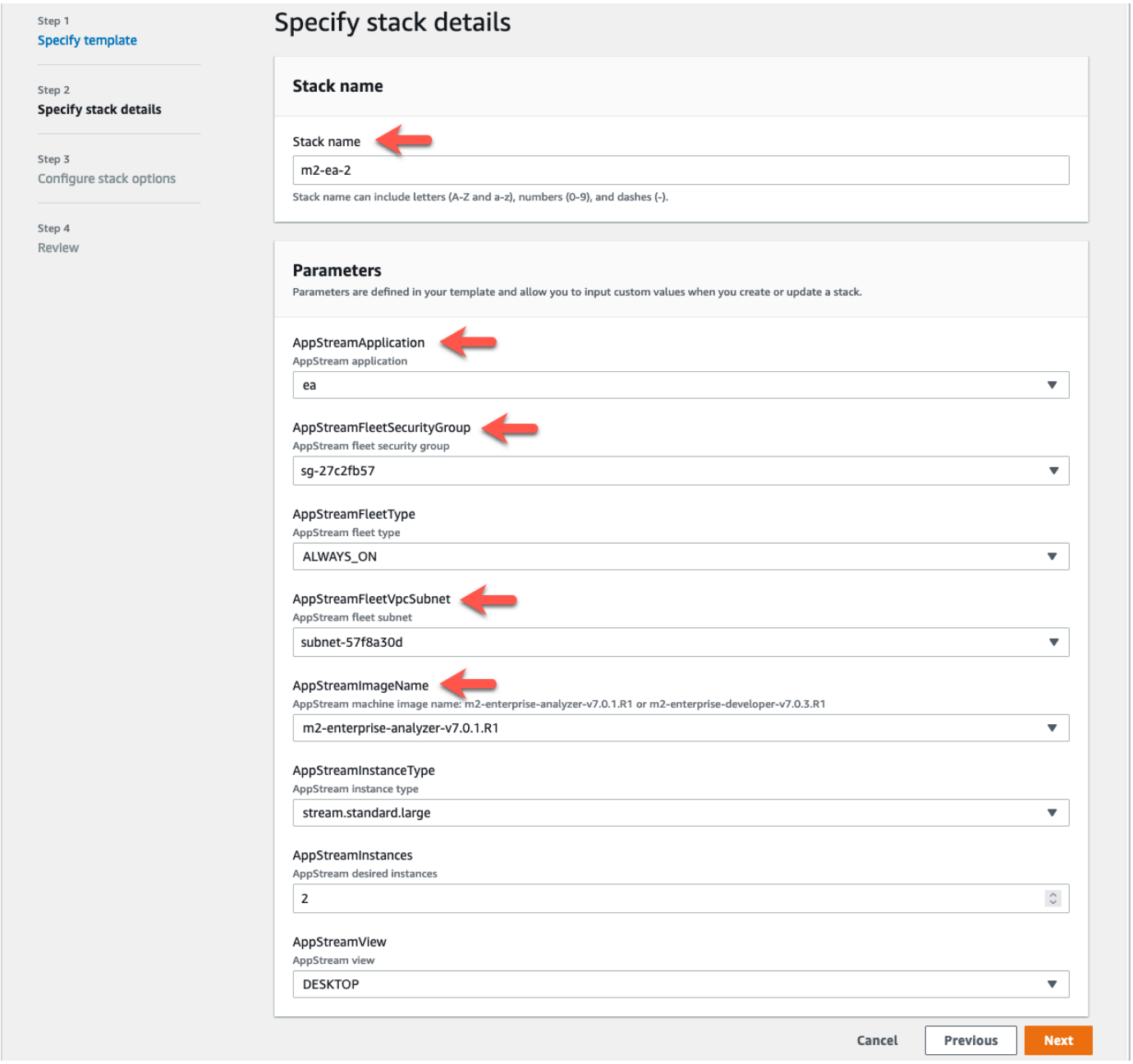
2. Abra o AWS CloudFormation console e escolha Criar pilha e com novos recursos (padrão).
3. Em Pré-requisito — Preparar modelo, selecione O modelo está pronto.
4. Em Especificar modelo, selecione Upload de um arquivo de modelo.
5. Em Carregar um arquivo de modelo, escolha Escolher arquivo e carregue o modelo [cfn-m2-appstream-fleet-ea-ed .yml.]
6. Escolha Próximo.



7. Em Especificar detalhes da pilha, insira as seguintes informações:

   - Em Nome da pilha, insira um nome de sua escolha. Por exemplo, **`m2-ea`**.
   - Em AppStreamApplication, escolha chá.
   - Em AppStreamFleetSecurityGroup, escolha o grupo de segurança padrão da sua VPC padrão.
   - Em AppStreamFleetVpcSubnet, escolha uma sub-rede em sua VPC padrão.
   - Em AppStreamImageName, escolha a imagem começando com`m2-enterprise-analyzer`. Essa imagem contém a versão atualmente suportada da ferramenta Micro Focus Enterprise Analyzer.

- Aceite os padrões para os outros campos e selecione Próximo.



8. Aceite todos os padrões e selecione Próximo novamente.
9. Na revisão, certifique-se de que todos os parâmetros sejam o que você pretende.
10. Role até o final, escolha Eu reconheço que a AWS CloudFormation pode criar recursos do IAM com nomes personalizados e escolha Create Stack.

A pilha e a frota demoram entre 20 e 30 minutos para serem criadas. Você pode escolher Atualizar para ver os AWS CloudFormation eventos à medida que eles ocorrem.

Etapa 3: criar um usuário na AppStream versão 2.0

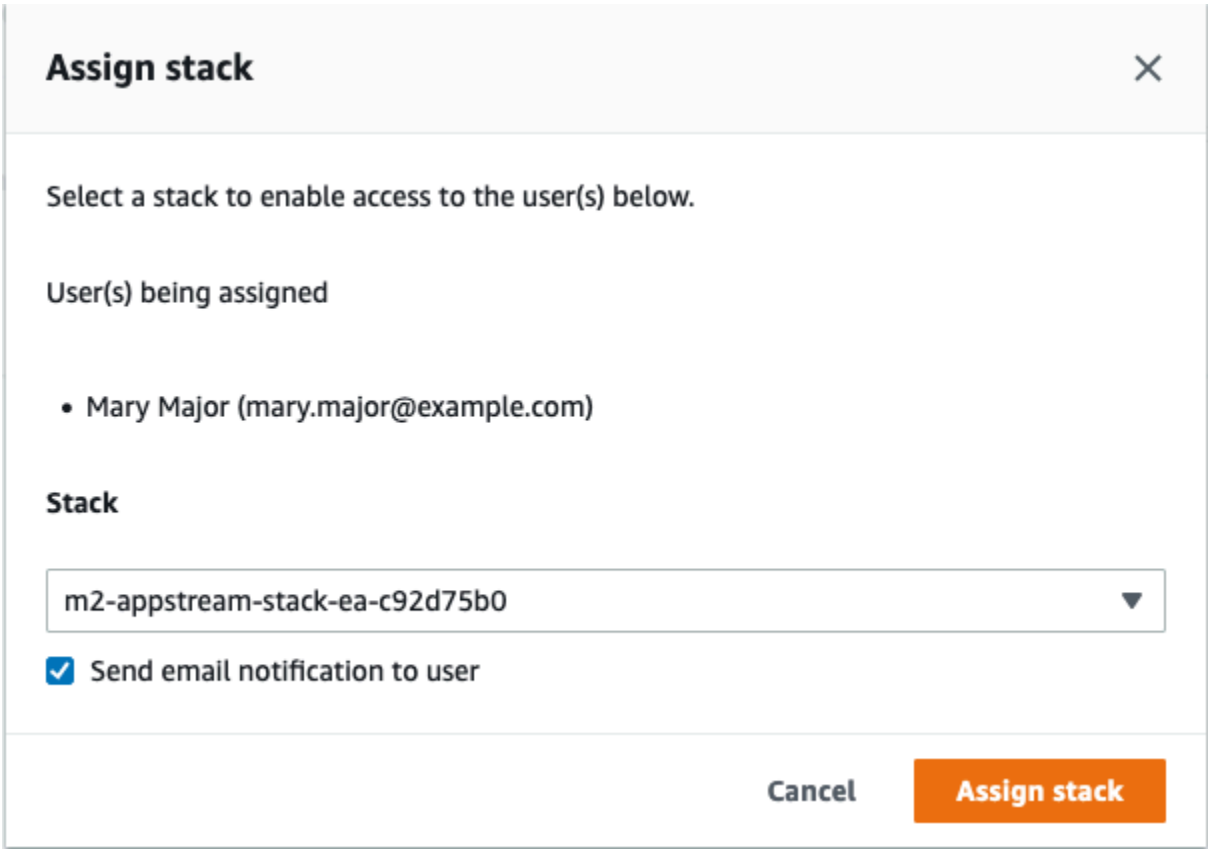
Enquanto espera AWS CloudFormation a conclusão da criação da pilha, você pode criar um ou mais usuários na AppStream versão 2.0. Esses usuários são aqueles que usarão o Enterprise Analyzer na AppStream versão 2.0. Você precisará especificar um endereço de e-mail para cada usuário e garantir que cada usuário tenha permissões suficientes para criar buckets no Amazon S3, fazer upload de arquivos em um bucket e vincular a um bucket para mapear seu conteúdo.

1. Abra o console AppStream 2.0.
2. Na navegação à esquerda, selecione Grupo de usuários.
3. Selecione Criar usuário.
4. Forneça um endereço de e-mail em que o usuário possa receber um convite por e-mail para usar AppStream 2.0, um nome e sobrenome e escolha Criar usuário.
5. Repita, se necessário, para criar mais usuários. O endereço de e-mail de cada usuário deve ser exclusivo.

Para obter mais informações sobre a criação de usuários AppStream 2.0, consulte AppStream Grupos de usuários 2.0 no Amazon AppStream 2.0 Administration Guide.

Ao AWS CloudFormation terminar de criar a pilha, você pode atribuir o usuário que você criou à pilha, da seguinte forma:

1. Abra o console AppStream 2.0.
2. Clique no nome de usuário.
3. Escolha Ação e, em seguida, Atribuir pilha.
4. Em Atribuir pilha, escolha a pilha que começa com `m2-appstream-stack-ea`.
5. Escolha Assign stack.



Atribuir um usuário a uma pilha faz com que AppStream 2.0 envie um e-mail para o usuário no endereço que você forneceu. Este e-mail contém um link para a página de login AppStream 2.0.

Etapa 4: Faça login no AppStream 2.0

Nesta etapa, você faz login no AppStream 2.0 usando o link no e-mail enviado pelo AppStream 2.0 para o usuário em que você criouEtapa 3: criar um usuário na AppStream versão 2.0.

1. Faça login no AppStream 2.0 usando o link fornecido no e-mail enviado pelo AppStream 2.0.
2. Altere sua senha, se solicitado. A tela AppStream 2.0 que você vê é semelhante à seguinte:

3. Escolha Desktop.
4. Na barra de tarefas, escolha Pesquisar e digite **`D:`** para navegar até a Pasta Pessoal.

> **Note**
>
> Se você pular essa etapa, poderá receber uma `Device not ready` mensagem de erro ao tentar acessar a Pasta Pessoal.

A qualquer momento, se você tiver problemas para fazer login no AppStream 2.0, reinicie sua frota AppStream 2.0 e tente fazer login novamente, seguindo as etapas a seguir.

1. Abra o console AppStream 2.0.
2. No painel de navegação à esquerda, escolha Frotas.
3. Escolha a frota que você está tentando usar.
4. Escolha Ação e, em seguida, escolha Parar.
5. Espere a frota parar.
6. Escolha Ação e, em seguida, escolha Iniciar.

Esse processo pode levar cerca de 10 minutos.

Etapa 5: verificar os buckets no Amazon S3 (opcional)

Uma das tarefas concluídas pelo AWS CloudFormation modelo que você usou para criar a pilha foi criar dois buckets no Amazon S3, que são necessários para salvar e restaurar os dados do usuário e as configurações do aplicativo em todas as sessões de trabalho. Esses buckets são os seguintes:

- O nome começa com `appstream2-`. Esse bucket mapeia dados para sua pasta inicial em AppStream 2.0 (`D:\PhotonUser\My Files\Home Folder`).

  > **Note**
  >
  > A Pasta Inicial é exclusiva para um determinado endereço de e-mail e é compartilhada entre todas as frotas e pilhas em uma determinada conta AWS . O nome da Pasta Pessoal é um hash SHA256 do endereço de e-mail do usuário e é armazenado em um caminho baseado nesse hash.

- O nome começa com `appstream-app-settings-`. Esse bucket contém informações da sessão do usuário para AppStream 2.0 e inclui configurações como favoritos do navegador, perfis de conexão do IDE e do aplicativo e personalizações da interface do usuário. Para obter mais informações, consulte Como funciona a persistência das configurações do aplicativo no Guia de administração da Amazon AppStream 2.0.

Para verificar se os buckets foram criados, siga estas etapas:

1. Abra o console Amazon S3.
2. Na navegação à esquerda, escolha Buckets.
3. Em Localizar compartimentos por nome, insira **`appstream`** para filtrar a lista.

Não é necessário realizar nenhuma ação adicional se você ver os compartimentos. Esteja ciente de que os buckets existem. Se você não vê os buckets, significa que o AWS CloudFormation modelo não terminou de ser executado ou ocorreu um erro. Acesse o AWS CloudFormation console e revise as mensagens de criação da pilha.

Próximas etapas

Agora que a infraestrutura AppStream 2.0 está configurada, você pode configurar e começar a usar o Enterprise Analyzer. Para ter mais informações, consulte Tutorial: Configurar o Enterprise Analyzer

na versão 2.0 AppStream . Também é possível configurar o Enterprise Developer. Para ter mais informações, consulte Tutorial: Configurar o Micro Focus Enterprise Developer na AppStream versão 2.0.

Limpar os recursos

O procedimento para limpar a pilha e as frotas criadas é descrito em Criar uma frota e uma pilha AppStream 2.0.

Quando os objetos AppStream 2.0 tiverem sido excluídos, o administrador da conta também poderá, se apropriado, limpar os buckets do Amazon S3 para as configurações do aplicativo e as pastas iniciais.

> **Note**
>
> A pasta inicial de um determinado usuário é exclusiva em todas as frotas, portanto, talvez seja necessário mantê-la se outras pilhas AppStream 2.0 estiverem ativas na mesma conta.

Por fim, o AppStream 2.0 atualmente não permite que você exclua usuários usando o console. Em vez disso, você deve usar a API de serviço com a CLI. Para obter mais informações, consulte Administração de grupos de usuários no Guia de administração da Amazon AppStream 2.0.

Tutorial: Configurar o Enterprise Analyzer na versão 2.0 AppStream

Este tutorial descreve como configurar o Micro Focus Enterprise Analyzer para analisar uma ou mais aplicações de mainframe. A ferramenta Enterprise Analyzer fornece vários relatórios com base em sua análise do código-fonte da aplicação e das definições do sistema.

Essa configuração foi projetada para promover a colaboração em equipe. A instalação usa um bucket do Amazon S3 para compartilhar o código-fonte com discos virtuais. Isso faz uso do Rclone) na máquina Windows. Com uma instância comum do Amazon RDS executando o PostgreSQL, qualquer membro da equipe pode acessar todos os relatórios solicitados.

Os membros da equipe também podem montar o disco virtual com suporte do Amazon S3 em suas máquinas pessoais e atualizar o bucket de origem a partir de suas estações de trabalho. Eles podem potencialmente usar scripts ou qualquer outra forma de automação em suas máquinas se estiverem conectados a outros sistemas internos locais.

A configuração é baseada nas imagens AppStream 2.0 do Windows que a modernização do AWS mainframe compartilha com o cliente. A configuração também é baseada na criação de frotas e pilhas AppStream 2.0, conforme descrito em. Tutorial: Configure a AppStream versão 2.0 para uso com o Micro Focus Enterprise Analyzer e o Micro Focus Enterprise Developer

> **Important**
>
> As etapas deste tutorial pressupõem que você configurou a AppStream versão 2.0 com o AWS CloudFormation modelo disponível para download cfn-m2-appstream-fleet-ea-ed.yml. Para ter mais informações, consulte Tutorial: Configure a AppStream versão 2.0 para uso com o Micro Focus Enterprise Analyzer e o Micro Focus Enterprise Developer.
> Para executar as etapas neste tutorial, você deve ter configurado a frota e a pilha do Enterprise Analyzer e elas devem estar em execução.

Para obter uma descrição completa dos recursos e resultados do Enterprise Analyzer, consulte a documentação do Enterprise Analyzer no site da Micro Focus.

Conteúdo da imagem

Além da própria aplicação Enterprise Analyzer, a imagem contém as seguintes ferramentas e bibliotecas.

Ferramentas de terceiros

- Python
- Rclone
- pgAdmin
- git-scm
- PostgreSQL ODBC driver

Bibliotecas em `C:\Users\Public`

- BankDemo código-fonte e definição do projeto para Enterprise Developer:`m2-bankdemo-template.zip`.
- Pacote de instalação do MFA para o mainframe: `mfa.zip`. Para obter mais informações, consulte Visão geral do acesso ao mainframe na documentação do Micro Focus Enterprise Developer.

- Arquivos de comando e configuração do Rclone (instruções para seu uso nos tutoriais): `m2-rclone.cmd` e `m2-rclone.conf`.

Tópicos

Pré-requisitos

- Faça upload do código-fonte e das definições do sistema para a aplicação do cliente que você deseja analisar em um bucket do S3. As definições do sistema incluem CICS CSD, definições de objetos do DB2 e assim por diante. Você pode criar uma estrutura de pastas dentro do bucket que faça sentido para a forma como você deseja organizar os artefatos da aplicação. Por exemplo, quando você descompacta a BankDemo amostra, ela tem a seguinte estrutura:

```
demo
     |--> jcl
     |--> RDEF
     |--> transaction
     |--> xa
```

- Crie e inicie uma instância do Amazon RDS executando PostgreSQL. Essa instância armazenará os dados e os resultados produzidos pelo Enterprise Analyzer. Você pode compartilhar essa instância com todos os membros da equipe da aplicação. Além disso, crie um esquema vazio chamado `m2_ea` (ou qualquer outro nome adequado) no banco de dados. Defina credenciais para usuários autorizados que lhes permitam criar, inserir, atualizar e excluir itens nesse esquema. Você pode obter o nome do banco de dados, o URL do endpoint do servidor e a porta TCP no console do Amazon RDS ou no administrador da conta.
- Certifique-se de ter configurado o acesso programático ao seu Conta da AWS. Para obter mais informações, consulte Acesso programático no Referência geral da Amazon Web Services.

Etapa 1: configuração

1. Inicie uma sessão com AppStream 2.0 com a URL que você recebeu na mensagem de e-mail de boas-vindas da AppStream versão 2.0.
2. Use seu e-mail como ID de usuário e defina sua senha permanente.
3. Selecione a pilha do Enterprise Analyzer.
4. Na página do menu AppStream 2.0, escolha Desktop para acessar a área de trabalho do Windows que a frota está transmitindo.

Etapa 2: criar a pasta virtual baseada no Amazon S3 no Windows

> **Note**
>
> Se você já usou o Rclone durante a pré-visualização da modernização do AWS mainframe, você deve atualizar `m2-rclone.cmd` para a versão mais recente localizada em. `C:\Users\Public`

1. Copie os arquivos `m2-rclone.conf` e `m2-rclone.cmd` fornecidos em `C:\Users\Public` para sua pasta pessoal `C:\Users\PhotonUser\My Files\Home Folder` usando o Explorador de Arquivos.
2. Atualize os parâmetros de `m2-rclone.conf` configuração com sua chave de AWS acesso e o segredo correspondente, bem como seu Região da AWS.

```
[m2-s3]
type = s3
provider = AWS
access_key_id = YOUR-ACCESS-KEY
secret_access_key = YOUR-SECRET-KEY
region = YOUR-REGION
acl = private
server_side_encryption = AES256
```

3. Mo `m2-rclone.cmd`, faça as seguintes alterações:
   - Altere `your-s3-bucket` para o nome do seu bucket do Amazon S3. Por exemplo, `m2-s3-mybucket`.

- Altere `your-s3-folder-key` para sua chave de bucket do Amazon S3. Por exemplo, `myProject`.
- Altere `your-local-folder-path` para o caminho do diretório em que você deseja que os arquivos da aplicação sejam sincronizados a partir do bucket do Amazon S3 que os contém. Por exemplo, `D:\PhotonUser\My Files\Home Folder\m2-new`. Esse diretório sincronizado deve ser um subdiretório da Pasta Inicial para que o AppStream 2.0 faça backup e restaure adequadamente no início e no final da sessão.

```
:loop
timeout /T 10
"C:\Program Files\rclone\rclone.exe" sync m2-s3:your-s3-bucket/your-s3-folder-
key "D:\PhotonUser\My Files\Home Folder\your-local-folder-path" --config "D:
\PhotonUser\My Files\Home Folder\m2-rclone.conf"
goto :loop
```

4. Abra um prompt de comando do Windows, toque em CD, `C:\Users\PhotonUser\My Files\Home Folder` se necessário, e execute `m2-rclone.cmd`. Esse script de comando executa um loop contínuo, sincronizando o bucket e a chave do Amazon S3 com a pasta local a cada 10 segundos. Você pode ajustar o tempo limite conforme necessário. Você deve ver o código-fonte da aplicação localizado no bucket do Amazon S3 no Windows File Explorer.

Para adicionar novos arquivos ao conjunto em que você está trabalhando ou para atualizar os existentes, faça o upload dos arquivos para o bucket do Amazon S3 e eles serão sincronizados com seu diretório na próxima iteração definida em `m2-rclone.cmd`. Da mesma forma, se quiser excluir alguns arquivos, exclua-os do bucket do Amazon S3. A próxima operação de sincronização os excluirá do seu diretório local.

Etapa 3: criar uma fonte de ODBC para a instância do Amazon RDS

1. Para iniciar a ferramenta EA_Admin, navegue até o menu seletor de aplicações no canto superior esquerdo da janela do navegador e escolha MF EA_Admin.
2. No menu Administrar, escolha Fontes de dados ODBC e escolha Adicionar na guia DSN do usuário.
3. Na caixa de diálogo Criar nova fonte de dados, escolha o driver PostgreSQL Unicode e, em seguida, selecione Concluir.

4. Na caixa de diálogo Configuração do driver ODBC PostgreSQL Unicode (psqlODBC), defina e anote o nome da fonte de dados que você deseja. Preencha os seguintes parâmetros com os valores da instância do RDS que você criou anteriormente:

   Descrição

   Descrição opcional para ajudá-lo a identificar essa conexão de banco de dados rapidamente.

   Banco de dados

   O banco de dados do Amazon RDS que você criou anteriormente.

   Servidor

   O endpoint do Amazon RDS.

   Porta

   A porta do Amazon RDS.

   Nome de usuário

   Conforme definido na instância do Amazon RDS.

   Senha

   Conforme definido na instância do Amazon RDS.

5. Escolha Testar para validar se a conexão com o Amazon RDS foi bem-sucedida e, em seguida, escolha Salvar para salvar seu novo DSN de usuário.
6. Espere até ver a mensagem que confirma a criação do espaço de trabalho adequado e, em seguida, escolha OK para finalizar com as fontes de dados ODBC e fechar a ferramenta EA_Admin.
7. Navegue novamente até o menu do seletor de aplicações e escolha Enterprise Analyzer para iniciar a ferramenta. Selecione Criar novo.
8. Na janela de configuração do espaço de trabalho, insira o nome do espaço de trabalho e defina sua localização. O espaço de trabalho pode ser o disco baseado no Amazon S3, se você trabalhar com essa configuração, ou sua pasta inicial, se preferir.
9. Escolha Escolher outro banco de dados para se conectar à sua instância do Amazon RDS.
10. Escolha o ícone do Postgre nas opções e, depois, escolha OK.

11. Para as configurações do Windows, em Opções — Definir parâmetros de conexão, insira o nome da fonte de dados que você criou. Insira também o nome do banco de dados, o nome do esquema, o nome do usuário e a senha. Escolha OK.
12. Aguarde até que o Enterprise Analyzer crie todas as tabelas, índices, etc. necessários para armazenar os resultados. Essa etapa pode levar alguns minutos. O Enterprise Analyzer confirma quando o banco de dados e o espaço de trabalho estão prontos para uso.
13. Navegue novamente até o menu do seletor de aplicações e escolha Enterprise Analyzer para iniciar a ferramenta.
14. A janela de inicialização do Enterprise Analyzer aparece no novo local selecionado do espaço de trabalho. Escolha OK.
15. Navegue até seu repositório no painel esquerdo, selecione o nome do repositório e escolha Adicionar arquivos/pastas ao seu espaço de trabalho. Selecione a pasta em que o código da aplicação está armazenado para adicioná-lo ao espaço de trabalho. Você pode usar o código de BankDemo exemplo anterior, se quiser. Quando o Enterprise Analyzer solicitar que você verifique esses arquivos, escolha Verificar para iniciar o relatório de verificação inicial do Enterprise Analyzer. A conclusão pode levar alguns minutos, dependendo do tamanho da aplicação.
16. Expanda seu espaço de trabalho para ver os arquivos e pastas que você adicionou ao espaço de trabalho. Os tipos de objetos e os relatórios de complexidade ciclomática também são visíveis no quadrante superior do painel Visualizador de gráficos.

Agora você pode usar o Enterprise Analyzer para todas as tarefas necessárias.

Sessões subsequentes

1. Inicie uma sessão com AppStream 2.0 com a URL que você recebeu na mensagem de e-mail de boas-vindas da AppStream versão 2.0.
2. Faça login com seu e-mail e senha permanente.
3. Selecione a pilha do Enterprise Analyzer.
4. Inicie `Rclone` para se conectar ao disco suportado pelo Amazon S3 se você usar essa opção para compartilhar os arquivos do espaço de trabalho.
5. Inicie o Enterprise Analyzer para fazer suas tarefas.

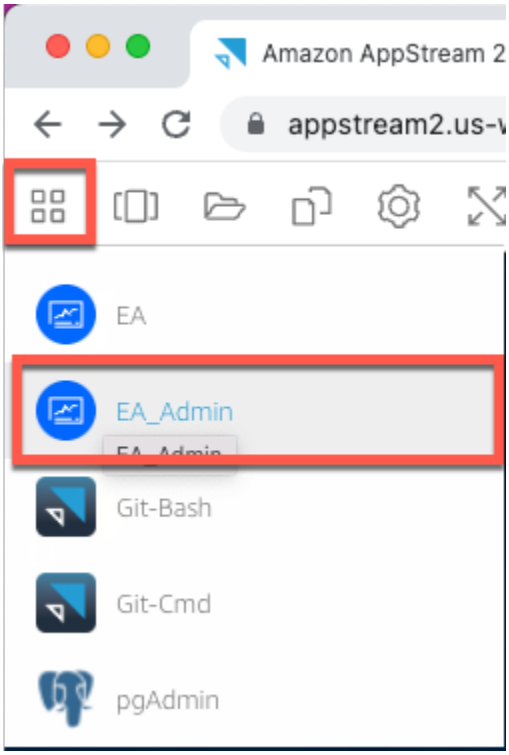
Solução de problemas de conexão do espaço de trabalho

Ao tentar se reconectar ao seu espaço de trabalho do Enterprise Analyzer, você pode ver um erro como este:

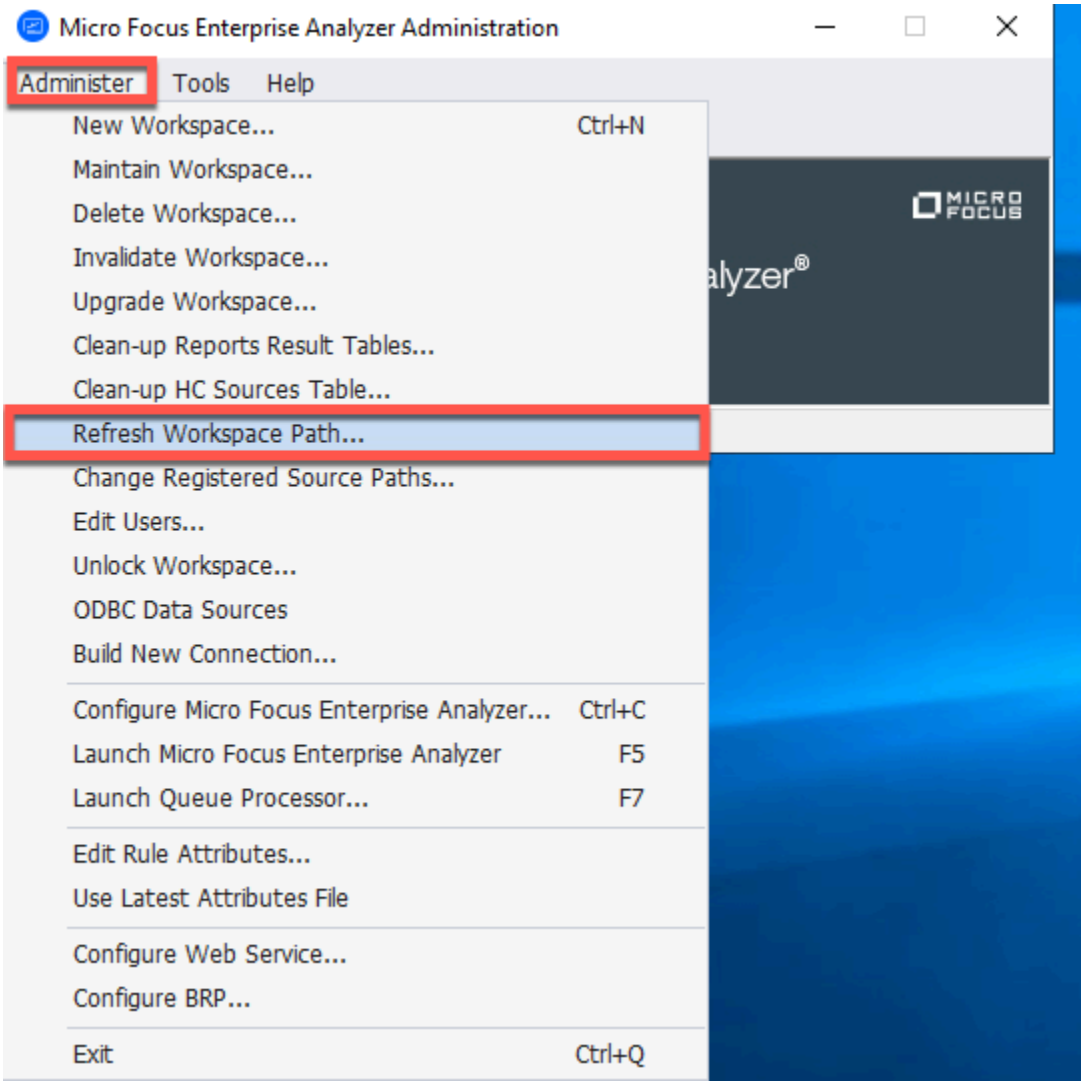
```
Cannot access the workspace directory D:\PhotonUser\My Files\Home Folder\EA_BankDemo.
 The workspace has been created on a non-shared disk of the EC2AMAZ-E6LC33H computer.
 Would you like to correct the workspace directory location?
```

Para resolver esse problema, escolha OK para limpar a mensagem e conclua as etapas a seguir.

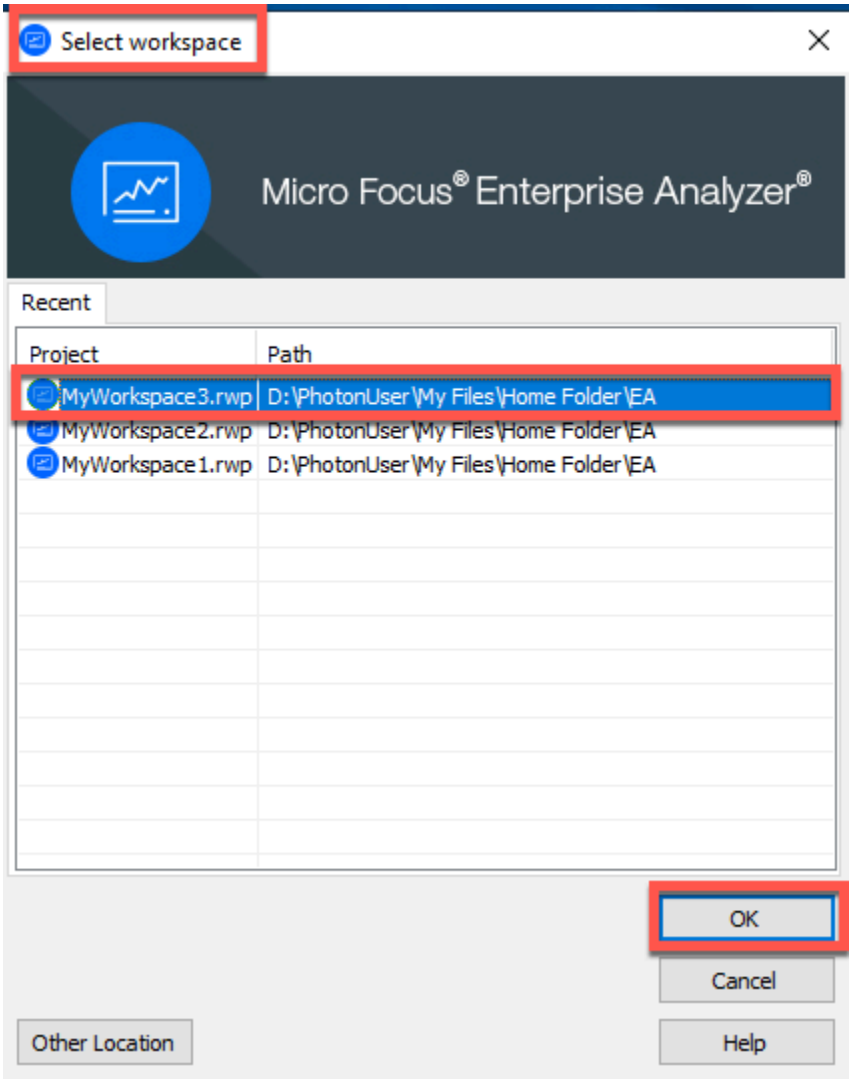
1. Na AppStream versão 2.0, escolha o ícone Iniciar aplicativo na barra de ferramentas e, em seguida, escolha EA_Admin para iniciar a ferramenta de administração do Micro Focus Enterprise Analyzer.



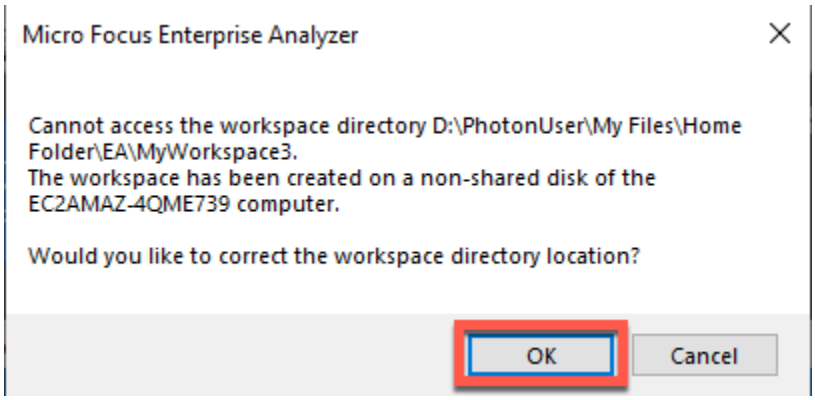
2. No menu Administrar, escolha Atualizar caminho do espaço de trabalho….



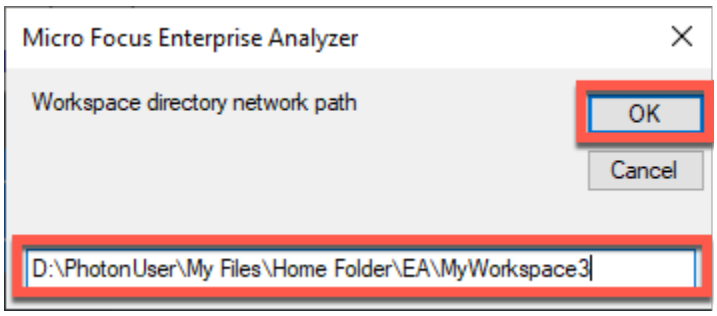
3. Em Selecionar espaço de trabalho, escolha o espaço de trabalho desejado e, em seguida, escolha OK.



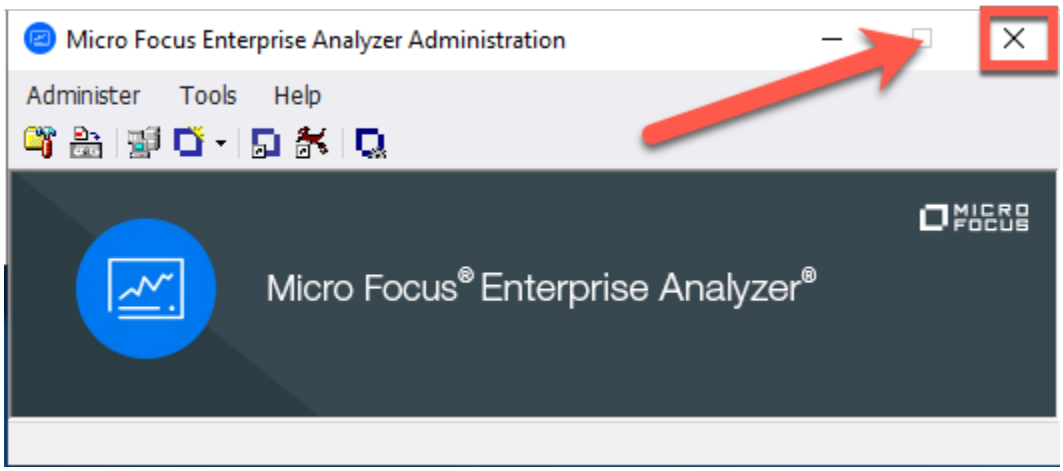
4. Escolha OK para confirmar a mensagem de erro.



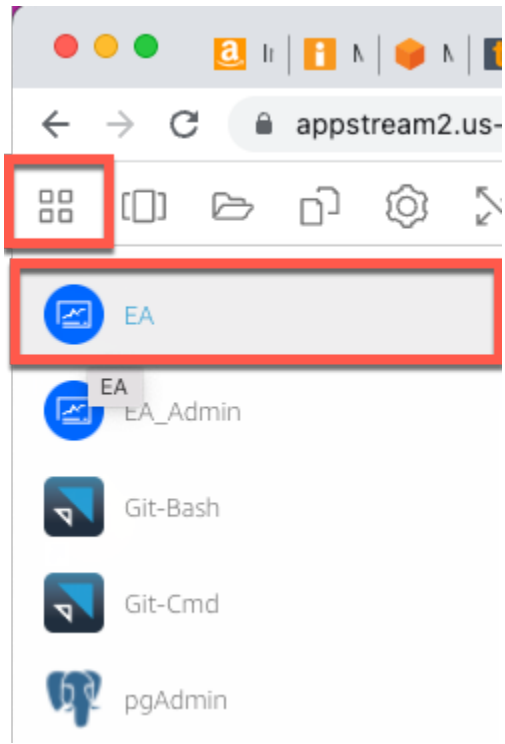
5. Em Caminho de rede do diretório Workspace, insira o caminho correto para seu espaço de trabalho, por exemplo, `D:\PhotonUser\My Files\Home Folder\EA\MyWorkspace3`.



6. Feche a ferramenta de administração do Micro Focus Enterprise Analyzer.



7. Na AppStream versão 2.0, escolha o ícone Iniciar aplicativo na barra de ferramentas e, em seguida, escolha EA para iniciar o Micro Focus Enterprise Analyzer.



8. Repita as etapas de 3 a 5.

O Micro Focus Enterprise Analyzer agora deve abrir com o espaço de trabalho existente.

Limpar recursos

Se os recursos criados durante este tutorial não forem mais necessários, exclua-os para não incorrer em cobranças adicionais. Execute as etapas a seguir:

- Use a ferramenta EA_Admin para excluir o espaço de trabalho.
- Exclua os buckets do S3 que você criou para este tutorial. Para obter mais informações, consulte Exclusão de um bucket no Guia do usuário do Amazon S3.
- Exclua o banco de dados que você criou para este tutorial. Para ter mais informações, consulte Excluir uma instância de banco de dados.

Tutorial: Configurar o Micro Focus Enterprise Developer na AppStream versão 2.0

Este tutorial descreve como configurar o Micro Focus Enterprise Developer para uma ou mais aplicações de mainframe para mantê-los, compilá-los e testá-los usando os recursos do Enterprise Developer. A configuração é baseada nas imagens AppStream 2.0 do Windows que a modernização do AWS mainframe compartilha com o cliente e na criação de frotas e pilhas AppStream 2.0, conforme descrito em. Tutorial: Configure a AppStream versão 2.0 para uso com o Micro Focus Enterprise Analyzer e o Micro Focus Enterprise Developer

> **Important**
>
> As etapas deste tutorial pressupõem que você configure a AppStream versão 2.0 usando o AWS CloudFormation modelo disponível para download cfn-m2-appstream-fleet-ea-ed.yaml. Para ter mais informações, consulte Tutorial: Configure a AppStream versão 2.0 para uso com o Micro Focus Enterprise Analyzer e o Micro Focus Enterprise Developer.
> Você deve executar as etapas dessa configuração quando a frota e a pilha do Enterprise Developer estiverem em funcionamento.

Para obter uma descrição completa dos recursos e resultados do Enterprise Developer v7, confira sua documentação up-to-date on-line (v7.0) no site da Micro Focus.

Conteúdo da imagem

Além do próprio Enterprise Developer, a imagem contém a imagem que contém Rumba (um emulador TN3270). Ele também contém as seguintes ferramentas e bibliotecas.

Ferramentas de terceiros

- Python
- Rclone
- pgAdmin
- git-scm
- PostgreSQL ODBC driver

Bibliotecas em `C:\Users\Public`

- BankDemo código-fonte e definição do projeto para Enterprise Developer:`m2-bankdemo-template.zip`.
- Pacote de instalação do MFA para o mainframe: `mfa.zip`. Para obter mais informações, consulte Visão geral do acesso ao mainframe na documentação do Micro Focus Enterprise Developer.
- Arquivos de comando e configuração do Rclone (instruções para seu uso nos tutoriais): `m2-rclone.cmd` e `m2-rclone.conf`.

Se você precisar acessar o código-fonte que ainda não foi carregado nos CodeCommit repositórios, mas que está disponível em um bucket do Amazon S3, por exemplo, para realizar o carregamento inicial do código-fonte no git, siga o procedimento para criar um disco virtual do Windows conforme descrito em. Tutorial: Configurar o Enterprise Analyzer na versão 2.0 AppStream

Tópicos

- Pré-requisitos
- Etapa 1: configuração por usuários individuais do Enterprise Developer
- Etapa 2: criar a pasta virtual baseada no Amazon S3 no Windows (opcional)
- Etapa 3: clonar o repositório
- Sessões subsequentes

- Limpar recursos

Pré-requisitos

- Um ou mais CodeCommit repositórios carregados com o código-fonte do aplicativo a ser mantido. A configuração do repositório deve corresponder aos requisitos do pipeline de CI/CD acima para criar sinergias por meio da combinação das duas ferramentas.
- Cada usuário deve ter credenciais para o CodeCommit repositório ou repositórios definidos pelo administrador da conta de acordo com as informações em Autenticação e controle de acesso da AWS. CodeCommit A estrutura dessas credenciais é revisada em Autenticação e controle de acesso para AWS CodeCommit e a referência completa para autorizações do IAM CodeCommit está na referência de CodeCommit permissões: o administrador pode definir políticas distintas do IAM para funções distintas, tendo credenciais específicas para a função de cada repositório e limitando suas autorizações do usuário ao conjunto específico de tarefas que ele precisa realizar em um determinado repositório. Portanto, para cada mantenedor do CodeCommit repositório, o administrador da conta gerará um usuário primário e concederá a esse usuário permissões para acessar o repositório ou repositórios necessários selecionando a política ou as políticas adequadas do IAM para acesso. CodeCommit

Etapa 1: configuração por usuários individuais do Enterprise Developer

1. Obtenha suas credenciais do IAM:
    1. Conecte-se ao AWS console em https://console.aws.amazon.com/iam/.
    2. Siga o procedimento descrito na etapa 3 de Configuração para usuários HTTPS usando credenciais do Git no Guia do usuário AWS CodeCommit .
    3. Copie as credenciais de CodeCommit login específicas que o IAM gerou para você, mostrando, copiando e colando essas informações em um arquivo seguro em seu computador local ou escolhendo Baixar credenciais para baixar essas informações como um arquivo.CSV. Você precisa dessas informações para se conectar CodeCommit a.
2. Inicie uma sessão com AppStream 2.0 com base na URL recebida no e-mail de boas-vindas. Use seu e-mail como nome de usuário e crie sua senha.
3. Selecione sua pilha de desenvolvedores corporativos.
4. Na página do menu, escolha Desktop para acessar a área de trabalho do Windows transmitida pela frota.

Etapa 2: criar a pasta virtual baseada no Amazon S3 no Windows (opcional)

Se houver necessidade do Rclone (veja acima), crie a pasta virtual baseada no Amazon S3 no Windows: (opcional se todos os artefatos do aplicativo vierem exclusivamente do acesso). CodeCommit

> **Note**
>
> Se você já usou o Rclone durante a pré-visualização da modernização do AWS mainframe, você deve atualizar `m2-rclone.cmd` para a versão mais recente localizada em. `C:\Users\Public`

1. Copie os arquivos `m2-rclone.conf` e `m2-rclone.cmd` fornecidos em `C:\Users\Public` para sua pasta pessoal `C:\Users\PhotonUser\My Files\Home Folder` usando o Explorador de Arquivos.
2. Atualize os parâmetros de `m2-rclone.conf` configuração com sua chave de AWS acesso e o segredo correspondente, bem como seu Região da AWS.

```
[m2-s3]
type = s3
provider = AWS
access_key_id = YOUR-ACCESS-KEY
secret_access_key = YOUR-SECRET-KEY
region = YOUR-REGION
acl = private
server_side_encryption = AES256
```

3. Mo `m2-rclone.cmd`, faça as seguintes alterações:

   - Altere `your-s3-bucket` para o nome do seu bucket do Amazon S3. Por exemplo, `m2-s3-mybucket`.
   - Altere `your-s3-folder-key` para sua chave de bucket do Amazon S3. Por exemplo, `myProject`.
   - Altere `your-local-folder-path` para o caminho do diretório em que você deseja que os arquivos da aplicação sejam sincronizados a partir do bucket do Amazon S3 que os contém. Por exemplo, `D:\PhotonUser\My Files\Home Folder\m2-new`. Esse diretório

sincronizado deve ser um subdiretório da Pasta Inicial para que o AppStream 2.0 faça backup e restaure adequadamente no início e no final da sessão.

```
:loop
timeout /T 10
"C:\Program Files\rclone\rclone.exe" sync m2-s3:your-s3-bucket/your-s3-folder-
key "D:\PhotonUser\My Files\Home Folder\your-local-folder-path" --config "D:
\PhotonUser\My Files\Home Folder\m2-rclone.conf"
goto :loop
```

4. Abra um prompt de comando do Windows, toque em CD, `C:\Users\PhotonUser\My Files \Home Folder` se necessário, e execute`m2-rclone.cmd`. Esse script de comando executa um loop contínuo, sincronizando o bucket e a chave do Amazon S3 com a pasta local a cada 10 segundos. Você pode ajustar o tempo limite conforme necessário. Você deve ver o código-fonte da aplicação localizado no bucket do Amazon S3 no Windows File Explorer.

Para adicionar novos arquivos ao conjunto em que você está trabalhando ou para atualizar os existentes, faça o upload dos arquivos para o bucket do Amazon S3 e eles serão sincronizados com seu diretório na próxima iteração definida em `m2-rclone.cmd`. Da mesma forma, se quiser excluir alguns arquivos, exclua-os do bucket do Amazon S3. A próxima operação de sincronização os excluirá do seu diretório local.

Etapa 3: clonar o repositório

1. Navegue até o menu seletor de aplicações no canto superior esquerdo da janela do navegador e selecione Enterprise Developer.
2. Conclua a criação do espaço de trabalho exigido pelo Enterprise Developer em sua pasta inicial escolhendo `C:\Users\PhotonUser\My Files\Home Folder` (aka `D: \PhotonUser\My Files\Home Folder`) como local para o espaço de trabalho.
3. No Enterprise Developer, clone seu CodeCommit repositório acessando o Project Explorer, clique com o botão direito do mouse e escolha Importar, Importar..., Git, Projetos do Git Clone URI. Em seguida, insira suas credenciais CodeCommit de login específicas e preencha a caixa de diálogo do Eclipse para importar o código.

O repositório CodeCommit git agora está clonado em seu espaço de trabalho local.

Seu espaço de trabalho do Enterprise Developer agora está pronto para iniciar o trabalho de manutenção em sua aplicação. Em particular, você pode usar a instância local do Microfocus Enterprise Server (ES) integrada ao Enterprise Developer para depurar e executar interativamente sua aplicação para validar suas alterações localmente.

> **Note**
>
> O ambiente local do Enterprise Developer, incluindo a instância local do Enterprise Server, é executado no Windows, enquanto a modernização do AWS mainframe é executada no Linux. Recomendamos que você execute testes complementares no ambiente Linux fornecido pela AWS Mainframe Modernization depois de confirmar o novo aplicativo CodeCommit e reconstruí-lo para esse destino e antes de implantar o novo aplicativo em produção.

Sessões subsequentes

Ao selecionar uma pasta que está sob gerenciamento AppStream 2.0, como a pasta inicial, para a clonagem do seu CodeCommit repositório, ela será salva e restaurada de forma transparente em todas as sessões. Conclua as seguintes etapas na próxima vez que precisar trabalhar com a aplicação:

1. Inicie uma sessão com AppStream 2.0 com base na URL recebida no e-mail de boas-vindas.
2. Faça login com seu e-mail e senha permanente.
3. Selecione a pilha Enterprise Developer.
4. Inicie `Rclone` para se conectar (veja acima) ao disco baseado no Amazon S3 quando essa opção for usada para compartilhar os arquivos do espaço de trabalho.
5. Inicie o Enterprise Developer para fazer seu trabalho.

Limpar recursos

Se os recursos criados durante este tutorial não forem mais necessários, exclua-os para que você não continue sendo cobrado por eles. Execute as etapas a seguir:

- Exclua o CodeCommit repositório que você criou para este tutorial. Para obter mais informações, consulte Excluir um CodeCommit repositório no Guia do AWS CodeCommit usuário.
- Exclua o banco de dados que você criou para este tutorial. Para ter mais informações, consulte Excluir uma instância de banco de dados.

Configure a automação para sessões de streaming do Micro Focus Enterprise Analyzer e do Micro Focus Enterprise Developer

Você pode executar automaticamente um script no início e no final da sessão para permitir uma automação específica para o contexto do seu cliente. Para obter mais informações sobre esse recurso AppStream 2.0, consulte Use scripts de sessão para gerenciar a experiência de streaming de seus usuários AppStream 2.0 no Guia de administração da Amazon AppStream 2.0.

Esse recurso exige que você tenha pelo menos as seguintes versões das imagens do Enterprise Analyzer e do Enterprise Developer:

- `m2-enterprise-analyzer-v8.0.4.R1`
- `m2-enterprise-developer-v8.0.4.R1`

Tópicos

Configure a automação no início da sessão

Se você quiser executar um script de automação quando os usuários se conectarem à AppStream versão 2.0, crie seu script e dê um nome a elem`2-user-setup.cmd`. Armazene o script na pasta inicial AppStream 2.0 para o usuário. As imagens AppStream 2.0 fornecidas pela Modernização do AWS Mainframe procuram um script com esse nome naquele local e o executam, se ele existir.

> **Note**
>
> A duração do script não pode exceder o limite definido em AppStream 2,0, que atualmente é de 60 segundos. Para obter mais informações, consulte Executar scripts antes do início das sessões de streaming no Guia de administração da Amazon AppStream 2.0.

Configure a automação no final da sessão

Se você quiser executar um script de automação quando os usuários se desconectarem da AppStream versão 2.0, crie seu script e dê um nome a elem`2-user-teardown.cmd`. Armazene

o script na pasta inicial AppStream 2.0 para o usuário. As imagens AppStream 2.0 fornecidas pela Modernização do AWS Mainframe procuram um script com esse nome naquele local e o executam, se ele existir.

> **Note**
>
> A duração do script não pode exceder o limite definido em AppStream 2,0, que atualmente é de 60 segundos. Para obter mais informações, consulte Executar scripts após o término das sessões de streaming no Guia de administração da Amazon AppStream 2.0.

Exibir conjuntos de dados como tabelas e colunas no Enterprise Developer

Você pode acessar conjuntos de dados de mainframe que são implantados na modernização de AWS mainframe usando o tempo de execução da Micro Focus. Você pode visualizar os conjuntos de dados migrados como tabelas e colunas de uma instância do Micro Focus Enterprise Developer. A visualização de conjuntos de dados dessa forma possibilita que você:

- Execute operações `SQL SELECT` nos arquivos de dados migrados.
- Exponha dados fora da aplicação de mainframe migrado sem alterar a aplicação.
- Filtre dados com facilidade e salve como CSV ou outros formatos de arquivo.

> **Note**
>
> As etapas 1 e 2 são atividades únicas. Repita as etapas 3 e 4 para cada conjunto de dados para criar as visualizações de banco de dados.

Tópicos

- Etapa 4: criar a visualização de banco de dados usando o arquivo de estrutura (STR)
- Etapa 5: exibir conjuntos de dados da Micro Focus como tabelas e colunas

Pré-requisitos

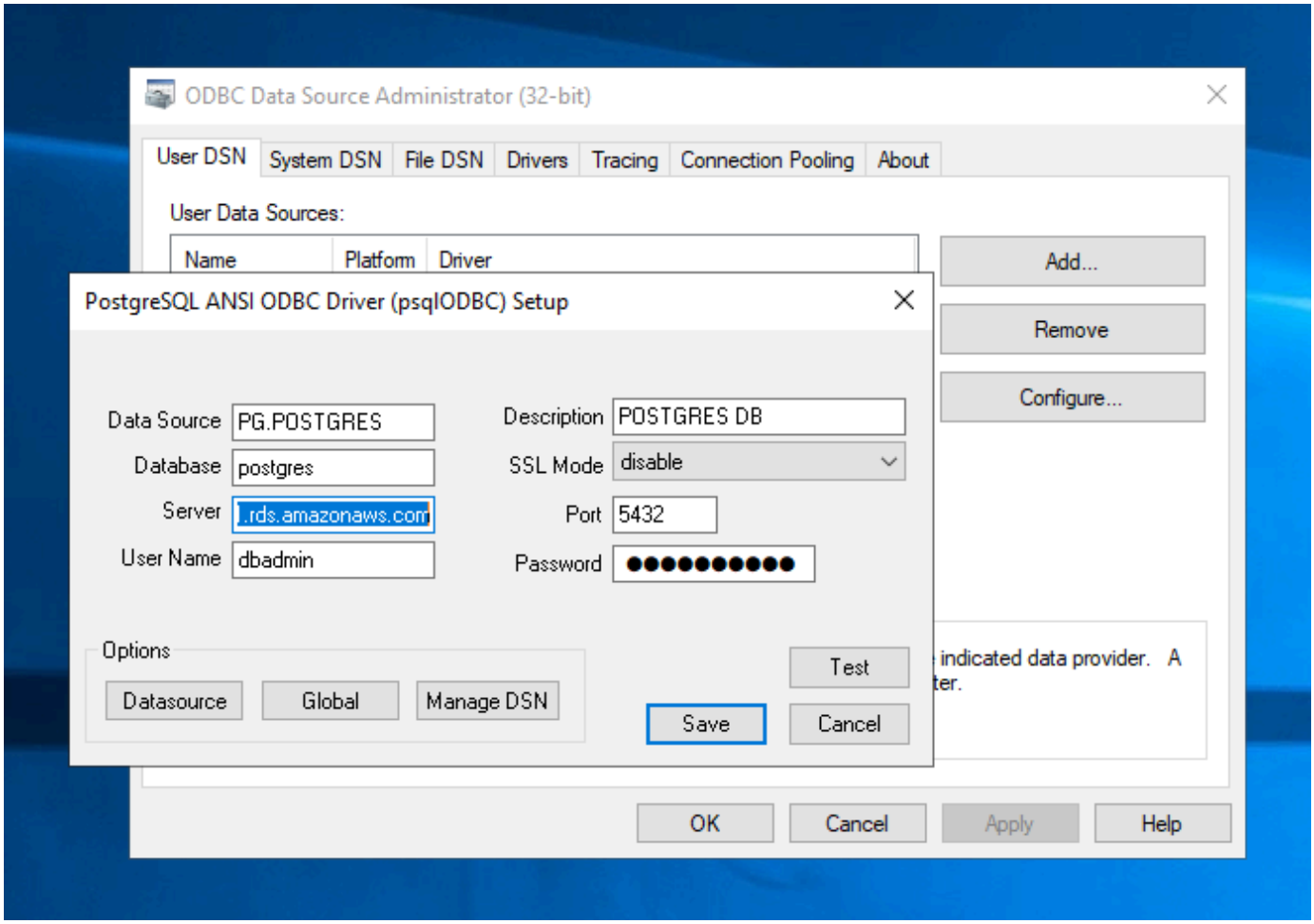
- Você deve ter acesso ao Micro Focus Enterprise Developer Desktop via AppStream 2.0.
- Você deve ter um aplicativo implantado e executado sob a modernização do AWS mainframe usando o mecanismo de tempo de execução da Micro Focus.
- Você está armazenando dados de aplicações na edição compatível com o Aurora PostgreSQL.

Etapa 1: configurar a conexão ODBC com o datastore Micro Focus (banco de dados Amazon RDS)

Nesta etapa, você configura uma conexão ODBC com o banco de dados que contém os dados que você deseja visualizar como tabelas e colunas. Esta é uma etapa única.

1. Faça login no Micro Focus Enterprise Developer Desktop usando o URL de streaming AppStream 2.0.
2. Abra o ODBC Data Source Administrator, escolha DSN do usuário e, em seguida, escolha Adicionar.
3. Em Criar nova fonte de dados, escolha PostgreSQL ANSI e, em seguida, escolha Concluir.
4. Crie uma fonte de dados `PG.POSTGRES` fornecendo as informações necessárias do banco de dados, da seguinte forma:

```
Data Source : PG.POSTGRES
Database    : postgres
Server      : rds_endpoint.rds.amazonaws.com
Port        : 5432
User Name   : user_name
Password    : user_password
```



5. Escolha Testar para garantir que a conexão funcione. Você deverá ver a mensagem `Connection successful` se o teste for bem-sucedido.

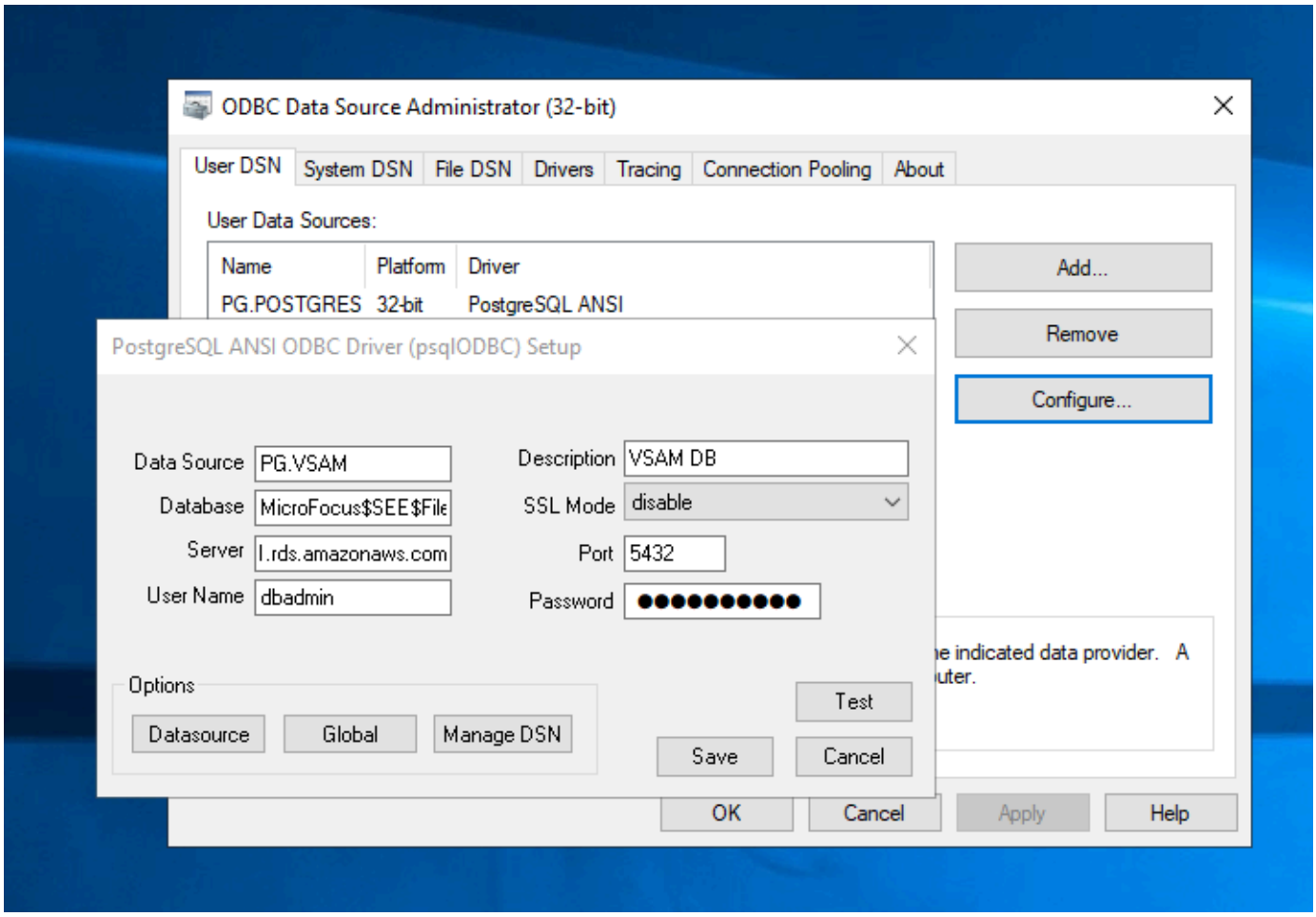
   Se o teste não for bem-sucedido, analise as informações a seguir.

   - Solução de problemas para o Amazon RDS
   - Como resolvo problemas ao me conectar à minha instância de banco de dados Amazon RDS?

6. Salve a fonte de dados.
7. Crie uma fonte de dados para `PG.VSAM`, teste a conexão e salve a fonte de dados. Forneça as seguintes informações para seu banco de dados:

```
Data Source : PG.VSAM
Database    : MicroFocus$SEE$Files$VSAM
Server      : rds_endpoint.rds.amazonaws.com
Port        : 5432
User Name   : user_name
```

```
Password    : user_password
```



Etapa 2: criar o arquivo MFDBFH.cfg

Nesta etapa, crie um arquivo de configuração que descreve o armazenamento de dados da Micro Focus. Esta é uma etapa única de configuração.

1. Na sua pasta pessoal, por exemplo, em `D:\PhotonUser\My Files\Home Folder\MFED \cfg\MFDBFH.cfg`, crie o arquivo MFDBFH.cfg com o conteúdo a seguir.

```
<datastores>
       <server name="ESPACDatabase" type="postgresql" access="odbc">
        <dsn name="PG.POSTGRES" type="database" dbname="postgres"/>
        <dsn name="PG.VSAM" type="datastore" dsname="VSAM"/>
       </server>
      </datastores>
```

2. Verifique a configuração do MFDBFH executando os seguintes comandos para consultar o datastore da Micro Focus:

```
*###*
*## Test the connection by running the following commands*
*###*

set MFDBFH_CONFIG="D:\PhotonUser\My Files\Home Folder\MFED\cfg\MFDBFH.cfg"

dbfhdeploy list sql://ESPACDatabase/VSAM?folder=/DATA
```

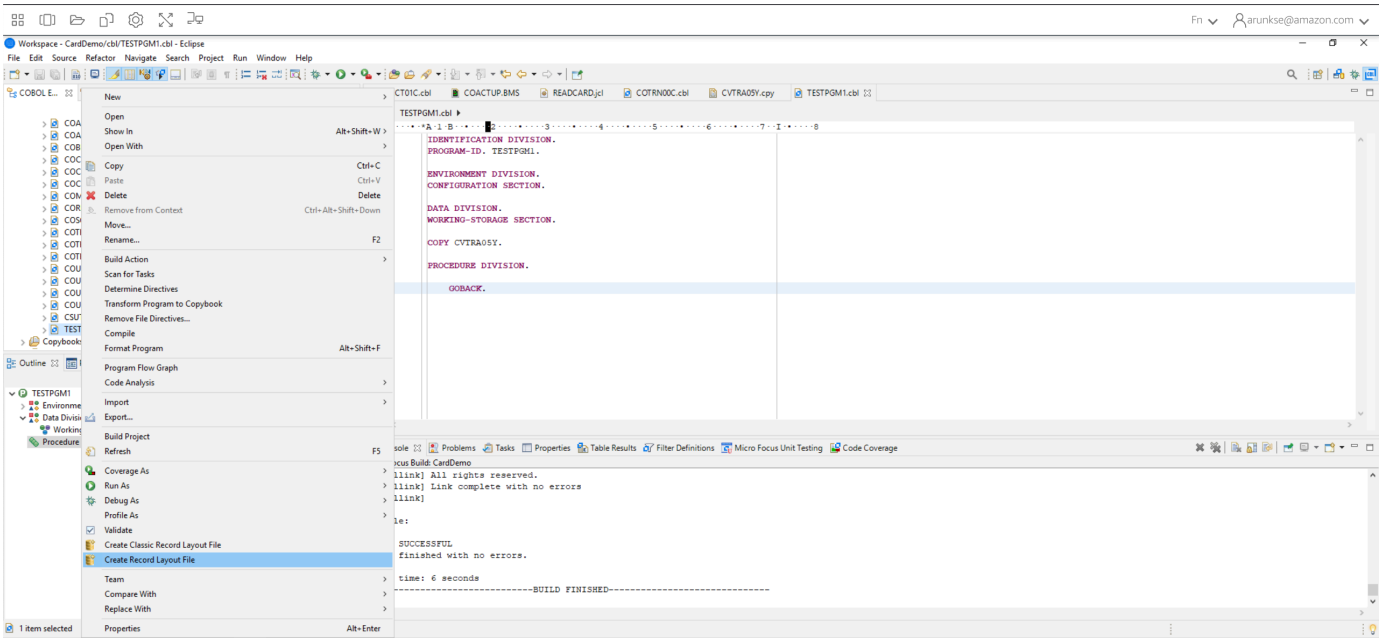
Etapa 3: criar um arquivo de estrutura (STR) para o layout do seu caderno

Nesta etapa, você cria um arquivo de estrutura para o layout do seu caderno para poder usá-lo posteriormente para criar visualizações de banco de dados a partir dos conjuntos de dados.

1. Compile o programa associado ao seu caderno. Se nenhum programa estiver usando o caderno, crie e compile um programa simples como o seguinte com uma instrução COPY para seu caderno.

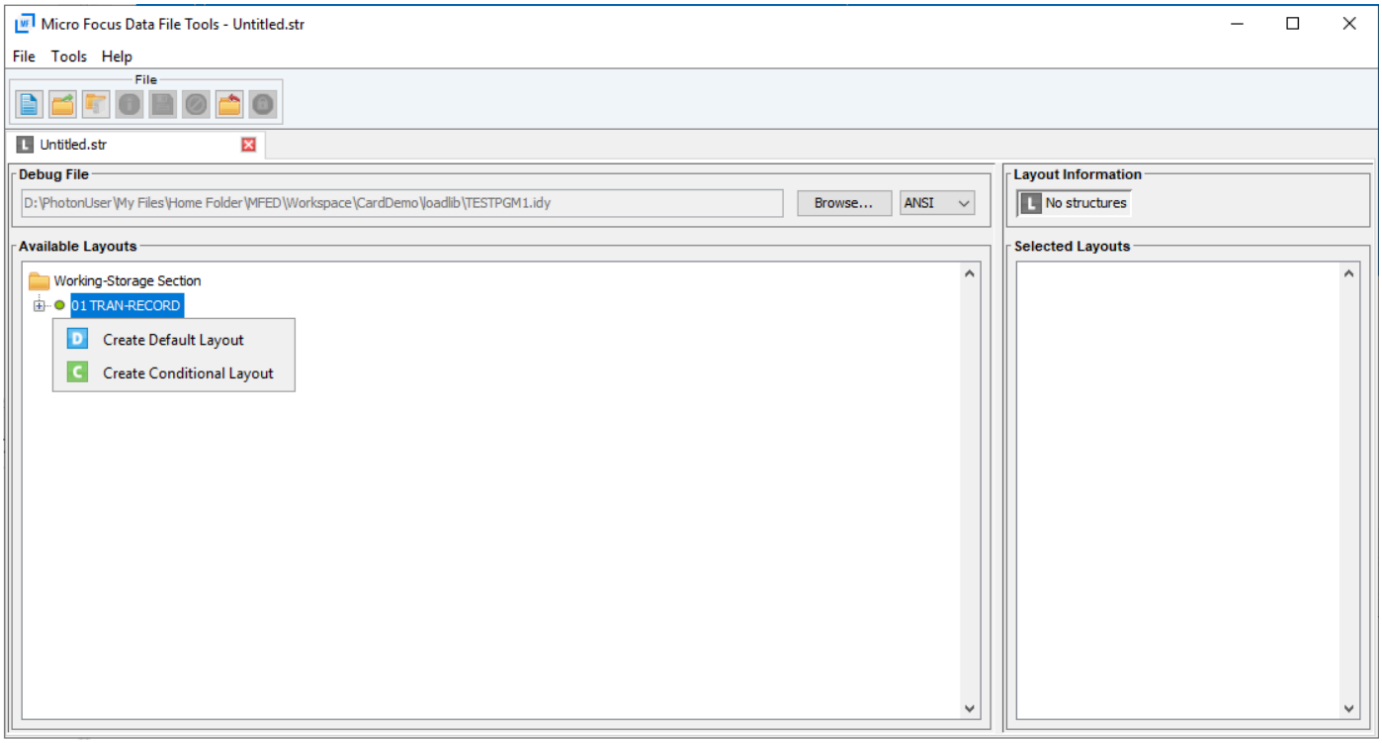
```
IDENTIFICATION DIVISION.
      PROGRAM-ID. TESTPGM1.

      ENVIRONMENT DIVISION.
      CONFIGURATION SECTION.

      DATA DIVISION.
      WORKING-STORAGE SECTION.

      COPY CVTRA05Y.

      PROCEDURE DIVISION.

      GOBACK.
```

2. Após a compilação bem-sucedida, clique com o botão direito do mouse no programa e escolha Criar arquivo de layout de registro. Isso abrirá as Ferramentas de Arquivo de Dados da Micro Focus usando o arquivo.idy gerado durante a compilação.

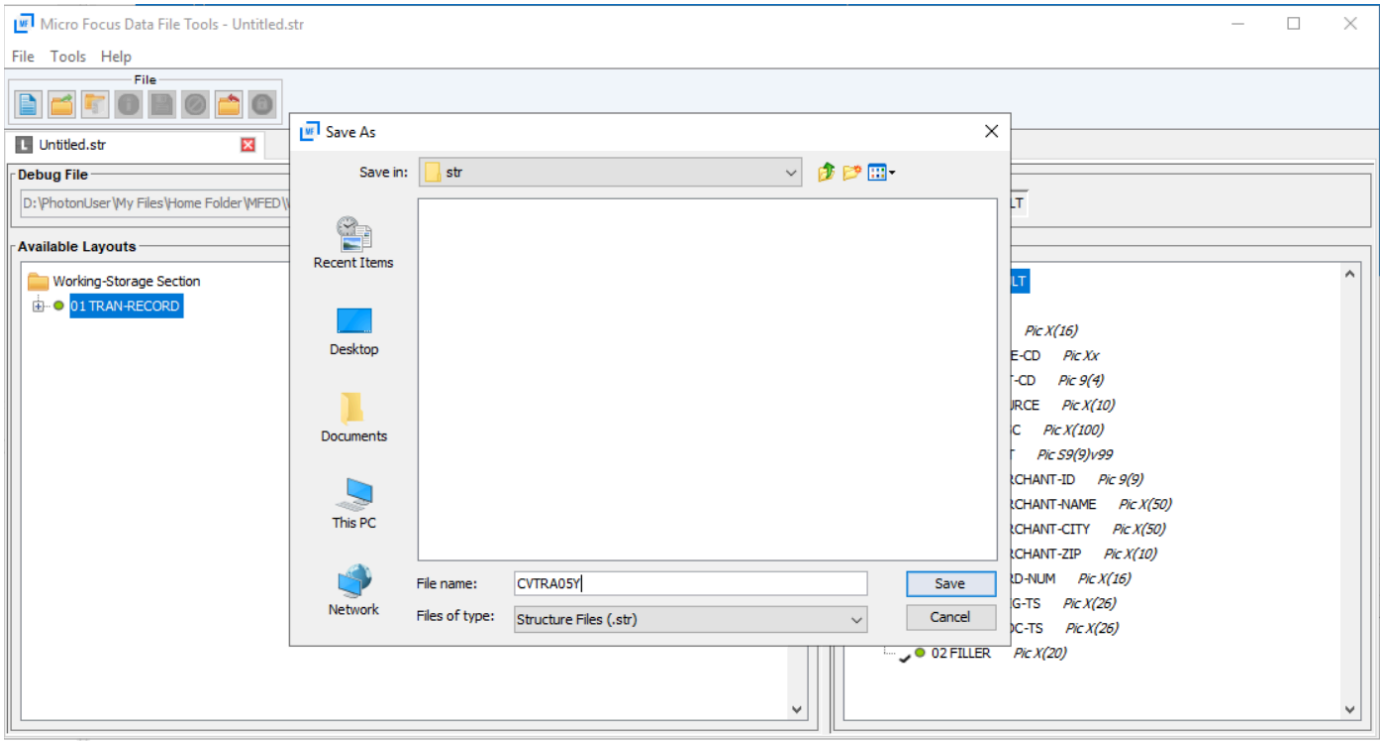


3. Clique com o botão direito na estrutura de registro e escolha Criar layout padrão (estrutura única) ou Criar layout condicional (estrutura múltipla), dependendo do layout.

   Para obter mais informações, consulte Criação de arquivos e layouts de estrutura na documentação da Micro Focus.



4. Depois de criar o layout, escolha Arquivo no menu e escolha Salvar como. Navegue e salve o arquivo em sua pasta pessoal com o mesmo nome de arquivo do seu caderno. Você pode escolher criar uma pasta chamada `str` e salvar todos os seus arquivos de estrutura lá.



Etapa 4: criar a visualização de banco de dados usando o arquivo de estrutura (STR)

Nesta etapa, você usa o arquivo de estrutura criado anteriormente para criar uma visualização do banco de dados para um conjunto de dados.

- Use o comando `dbfhview` para criar uma visualização do banco de dados para um conjunto de dados que já está no datastore da Micro Focus, conforme mostrado no exemplo a seguir.

```
##
      ## The below command creates database view for VSAM file
 AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS
      ## using the STR file CVTRA05Y.str
      ##

      dbfhview -create -struct:"D:\PhotonUser\My Files\Home Folder\MFED\str
\CVTRA05Y.str" -name:V_AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT -file:sql://
ESPACDatabase/VSAM/AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT?folder=/DATA
```

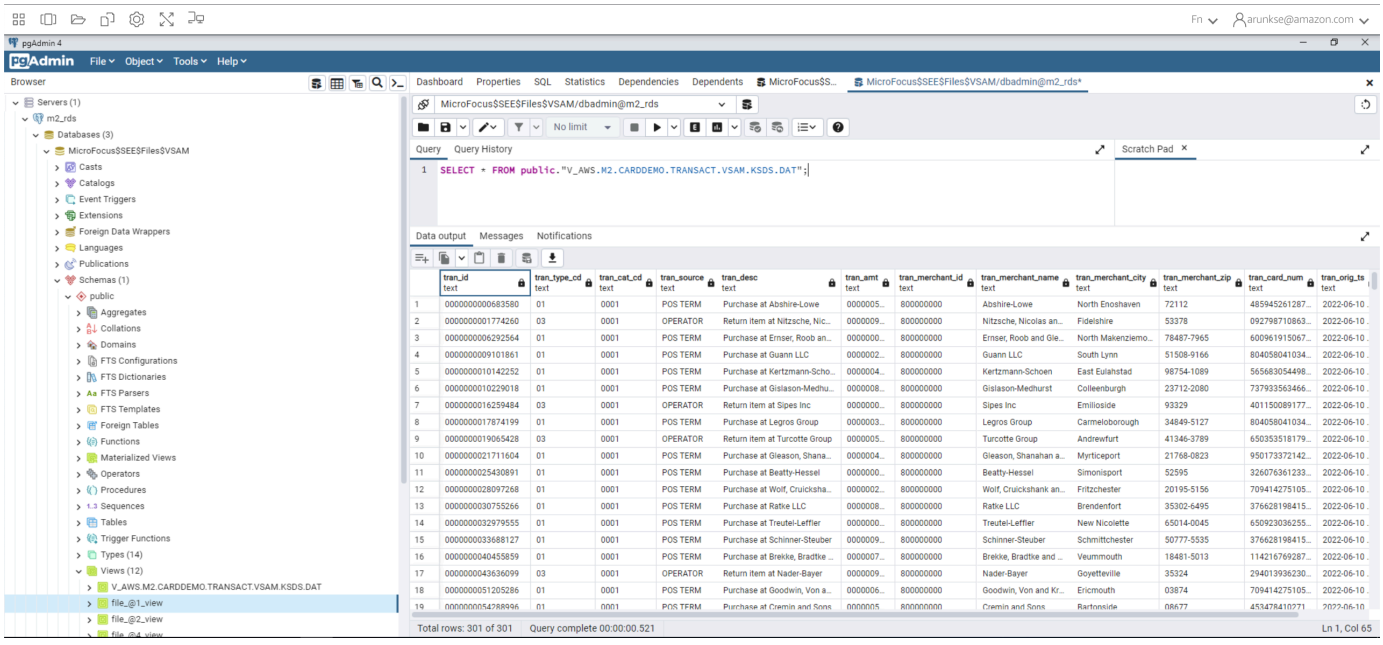
```
      ##
      ## Output:
      ##

      Micro Focus Database File Handler - View Generation Tool Version 8.0.00
      Copyright (C) 1984-2022 Micro Focus. All rights reserved.

      VGN0017I Using structure definition 'TRAN-RECORD-DEFAULT'
      VGN0022I View 'V_AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT' installed in
datastore 'sql://espacdatabase/VSAM'
      VGN0002I The operation completed successfully
```

Etapa 5: exibir conjuntos de dados da Micro Focus como tabelas e colunas

Nesta etapa, conecte-se ao banco de dados usando pgAdmin para que você possa executar consultas para visualizar os conjuntos de dados, como tabelas e colunas.

- Conecte-se ao banco de dados `MicroFocus$SEE$Files$VSAM` usando o pgAdmin e consulte a visualização do banco de dados que você criou na etapa 4.

```
SELECT * FROM public."V_AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT";
```



Tutorial: Use modelos com o Micro Focus Enterprise Developer

Este tutorial descreve como usar modelos e projetos predefinidos com o Micro Focus Enterprise Developer. Ele abrange três casos de uso. Todos os casos de uso usam o código de amostra fornecido na BankDemo amostra. Para baixar a amostra, escolha `bankdemo.zip`.

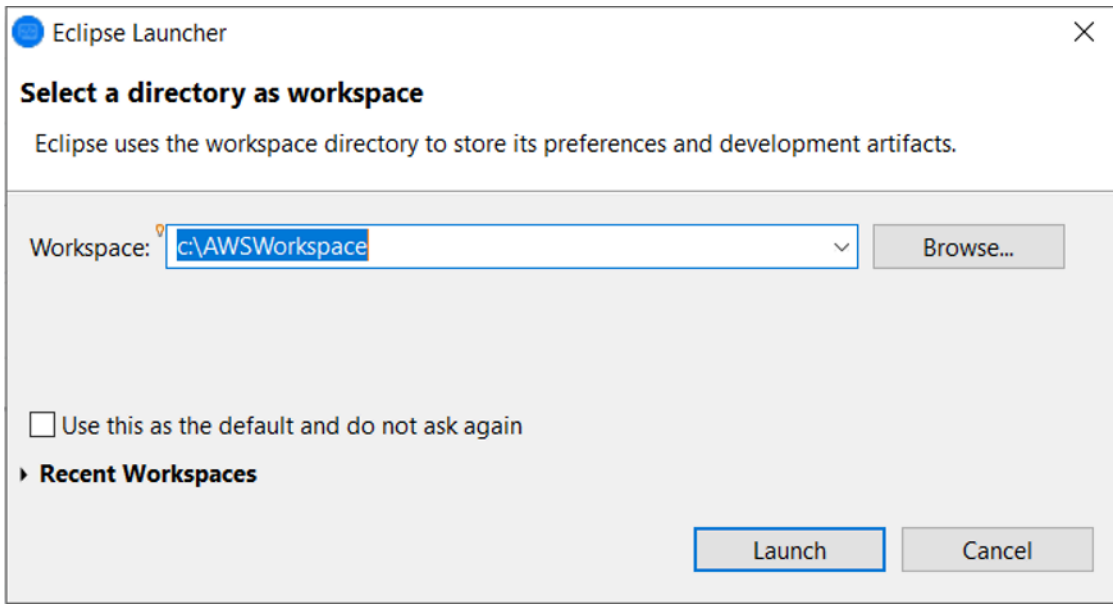
> **Important**
>
> Se você usar a versão do Enterprise Developer para Windows, os binários gerados pelo compilador poderão ser executados somente no Enterprise Server fornecido com o Enterprise Developer. Você não pode executá-los no tempo de execução da Modernização do AWS Mainframe, que é baseado no Linux.

Tópicos

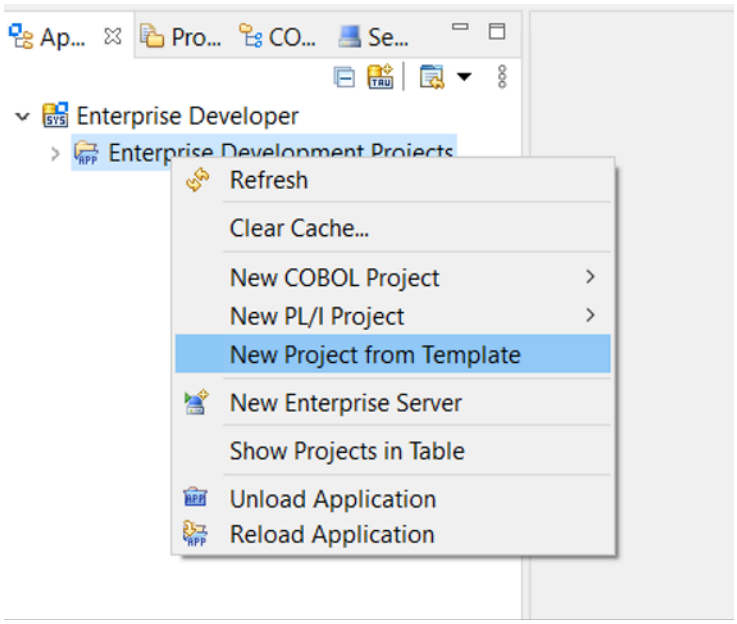
Caso de uso 1 - Usando o modelo de projeto COBOL contendo componentes de origem

Esse caso de uso exige que você copie os componentes de origem na estrutura de diretórios do modelo como parte das etapas de pré-configuração da demonstração. No, `bankdemo.zip`isso foi alterado em relação à `AWSTemplates.zip` entrega original para evitar duas cópias da fonte.

1. Inicie o Enterprise Developer e especifique o espaço de trabalho escolhido.

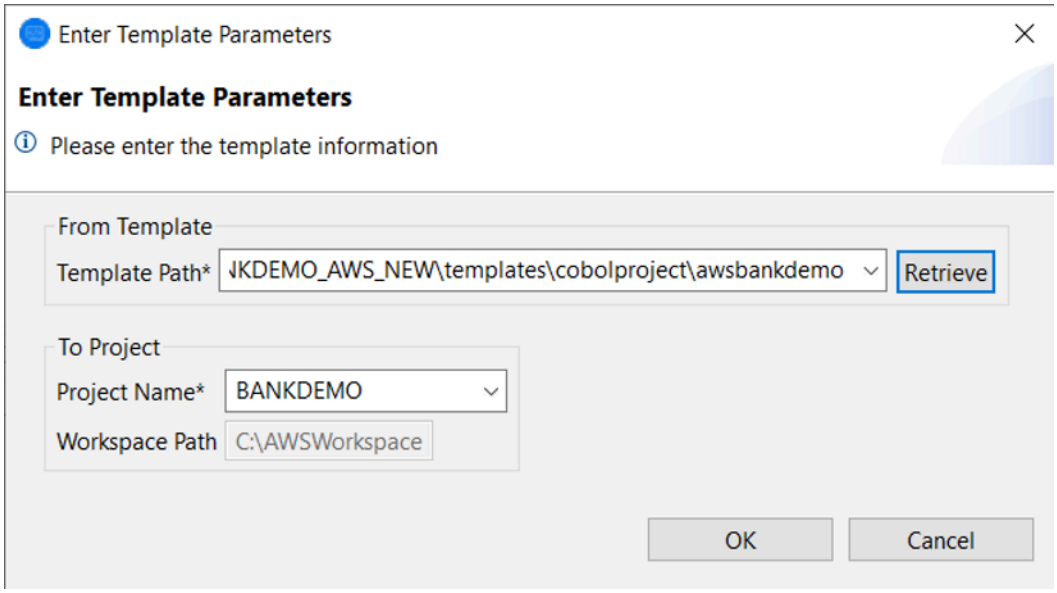


2. Na visualização do Explorador de Aplicações, no item de visualização em árvore do Enterprise Development Project, escolha Novo projeto a partir do modelo no menu de contexto.

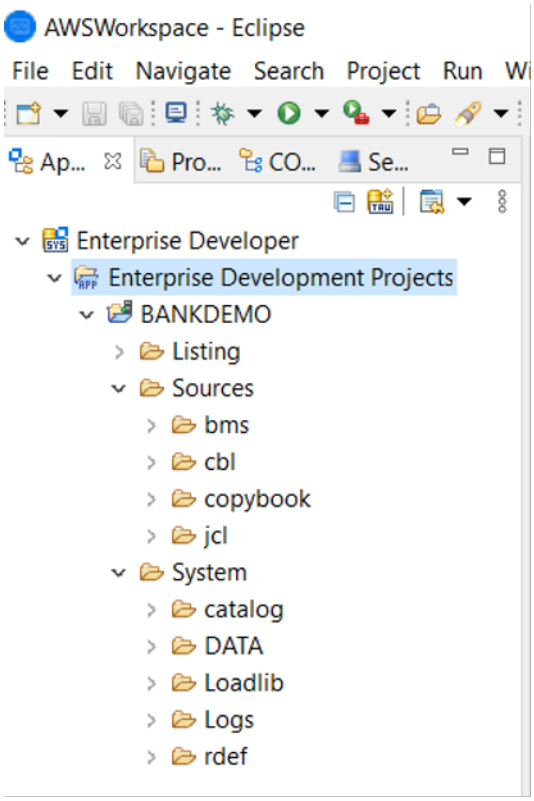


3. Insira os parâmetros do modelo conforme indicado.

> **Note**
>
> O caminho do modelo se referirá ao local onde o ZIP foi extraído.



4. Escolher OK criará um projeto Eclipse de desenvolvimento local com base no modelo fornecido, com uma estrutura completa do ambiente de origem e execução.



A estrutura `System` contém um arquivo completo de definição de recursos com as entradas necessárias para o BANKDEMO, o catálogo necessário com entradas adicionadas e os arquivos de dados ASCII correspondentes.

Como a estrutura do modelo de origem contém todos os itens de origem, esses arquivos são copiados para o projeto local e, portanto, são criados automaticamente no Enterprise Developer.

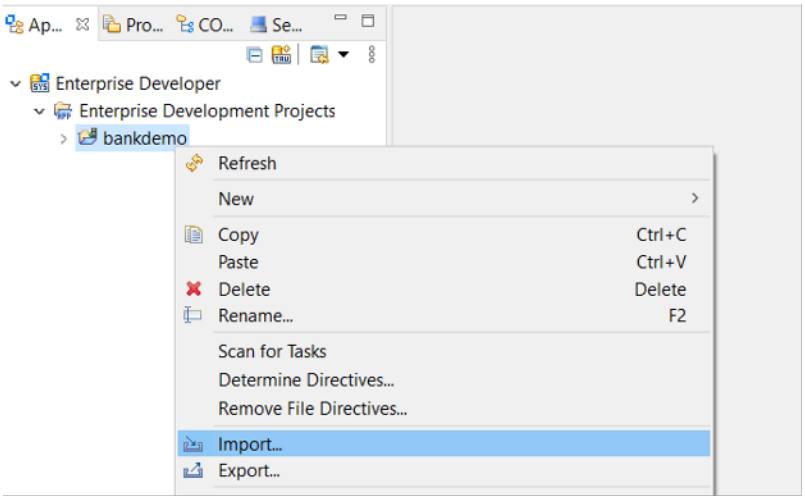
Caso de uso 2 - Usando o modelo de projeto COBOL sem componentes de origem

As etapas 1 a 3 são idênticas Caso de uso 1 - Usando o modelo de projeto COBOL contendo componentes de origem a.

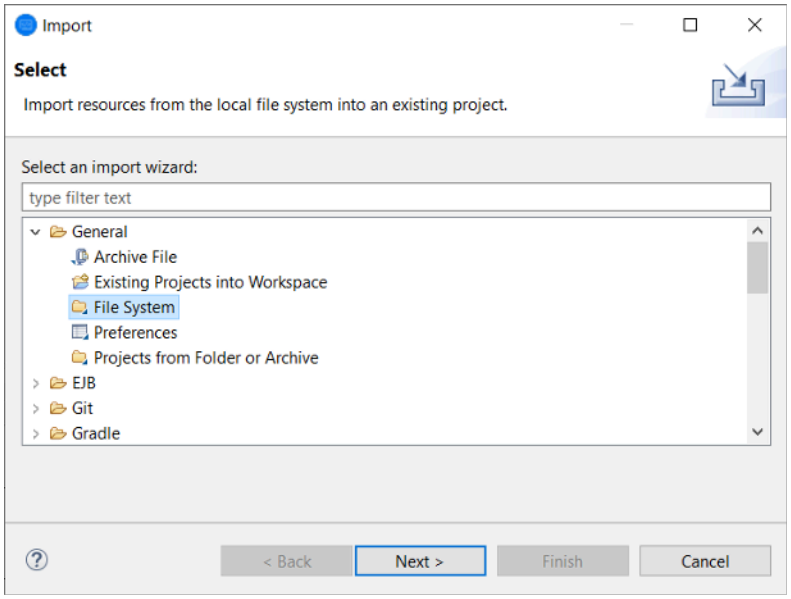
A `System` estrutura nesse caso de uso também contém um arquivo completo de definição de recursos com as entradas necessárias para BankDemo, o catálogo necessário com entradas adicionadas e os arquivos de dados ASCII correspondentes.

No entanto, a estrutura de origem do modelo não contém nenhum componente. Você deve importá-los para o projeto a partir de qualquer repositório de origem que estiver usando.

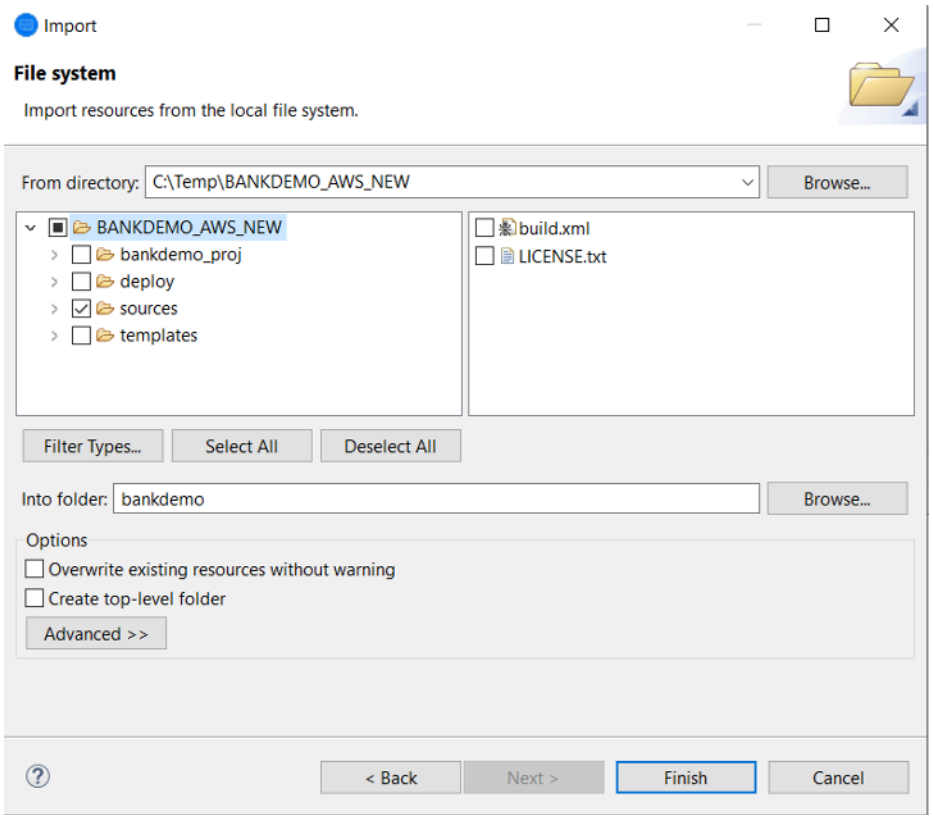
1. Escolha o nome do projeto. No menu de contexto relacionado, escolha Importar.



2. Na caixa de diálogo resultante, na seção Geral, escolha Sistema de arquivos e, em seguida, escolha Próximo.



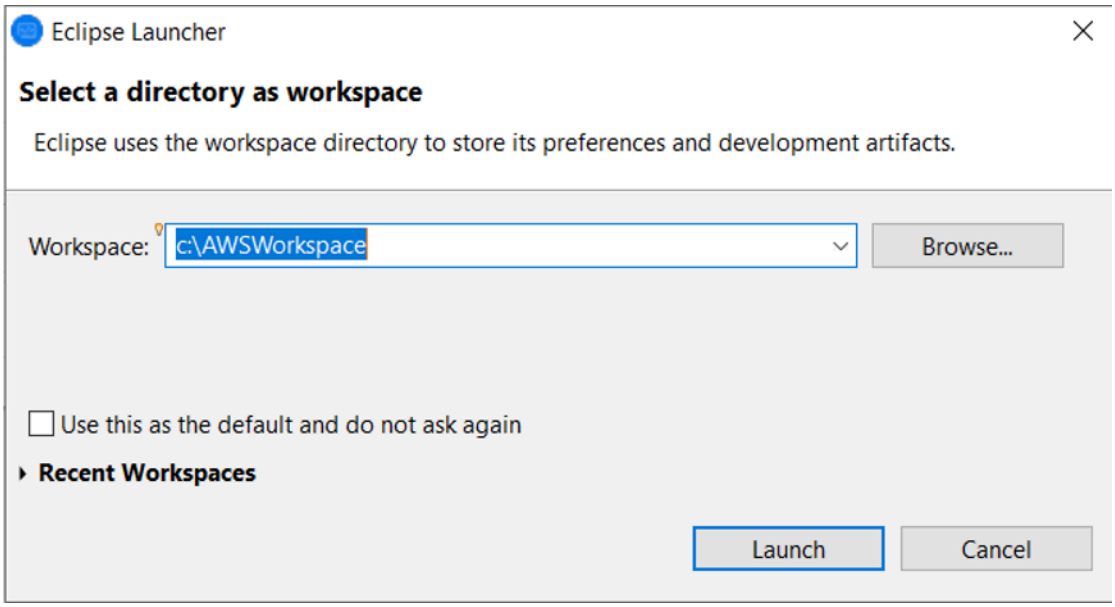
3. Preencha o campo Do diretório navegando no sistema de arquivos para apontar para a pasta do repositório. Escolha todas as pastas que você deseja importar, como `sources`. O `Into folder` campo será preenchido automaticamente. Escolha Concluir.



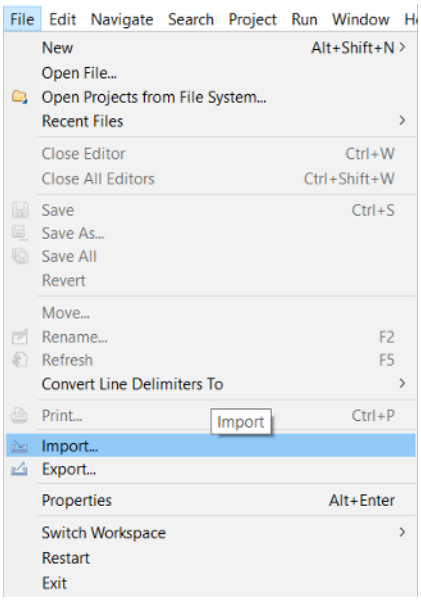
Depois que a estrutura do modelo de origem contém todos os itens de origem, eles são criados automaticamente no Enterprise Developer.

Caso de uso 3 - Usando o projeto COBOL predefinido vinculado às pastas de origem

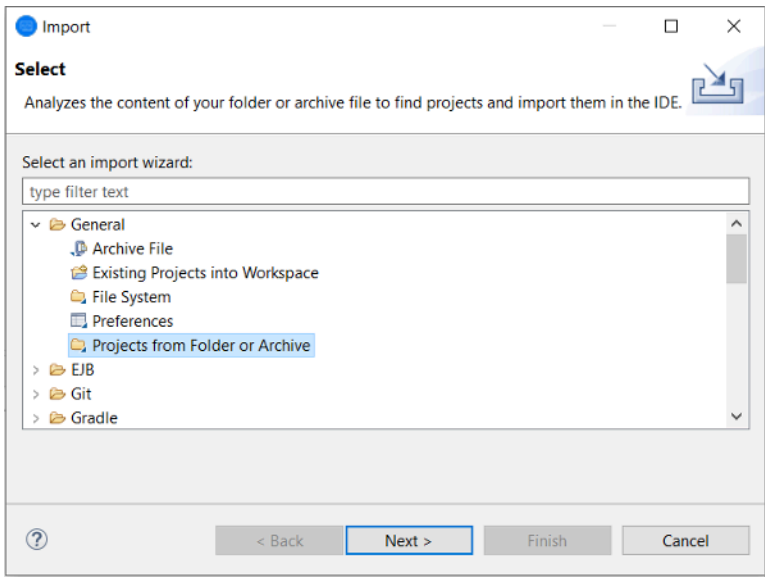
1. Inicie o Enterprise Developer e especifique o espaço de trabalho escolhido.



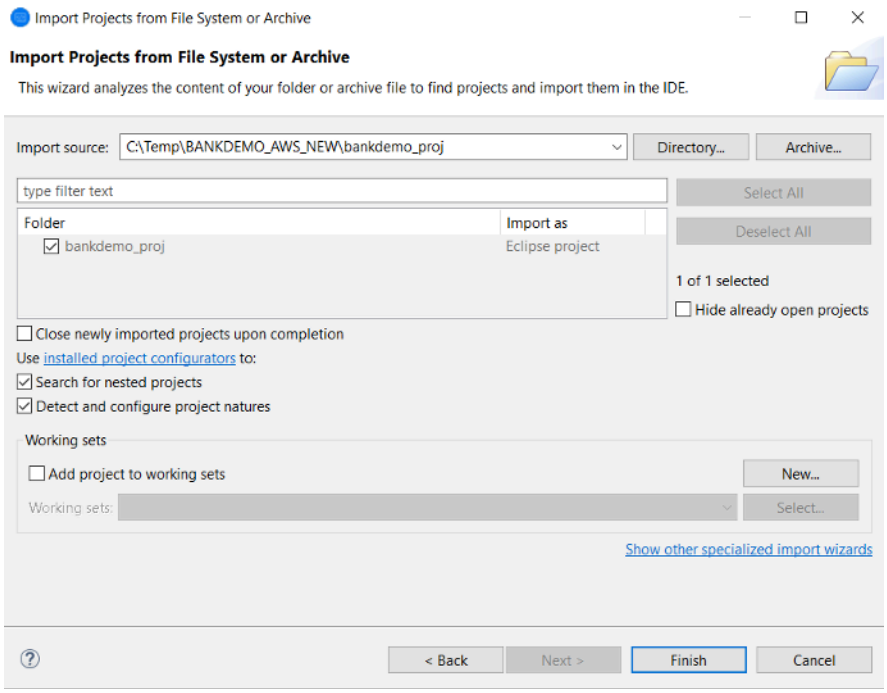
2. No menu File (Arquivo), escolha Import (Importar).



3. Na caixa de diálogo resultante, em Geral, escolha Projetos da pasta ou do arquivo e escolha Próximo.



4. Preencha a fonte de importação, escolha o diretório e navegue pelo sistema de arquivos para selecionar a pasta predefinida do projeto. O projeto contido nele tem links para as pastas de origem no mesmo repositório.

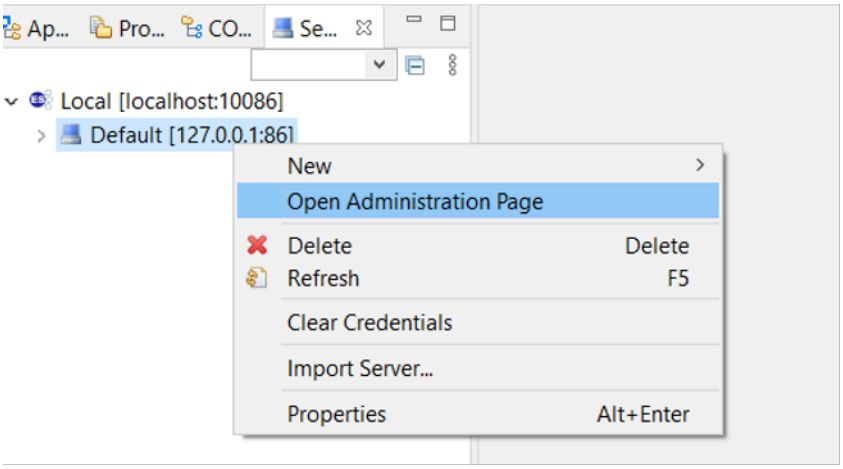


Escolha Terminar.

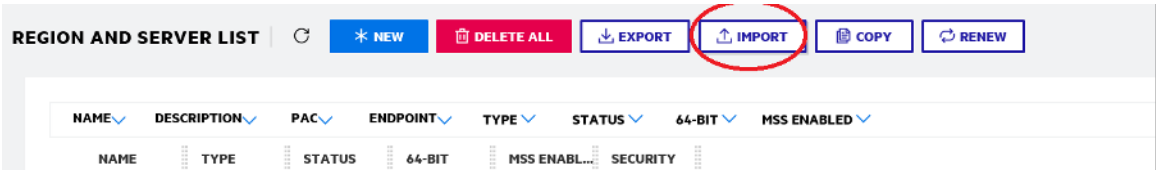
Como o projeto é preenchido pelos links para a pasta de origem, o código é criado automaticamente.

Usando o modelo JSON de definição de região

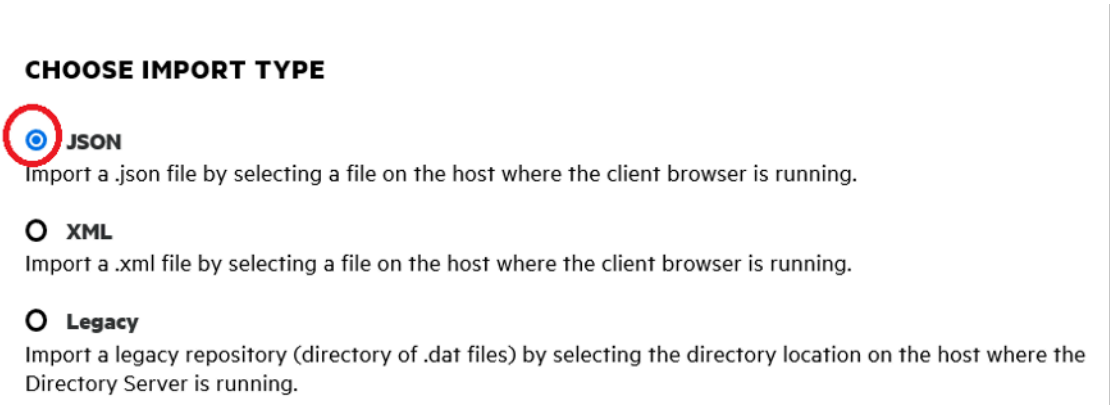
1. Alterne para a visualização Server Explorer. No menu de contexto relacionado, escolha Abrir página de administração, que inicia o navegador padrão.



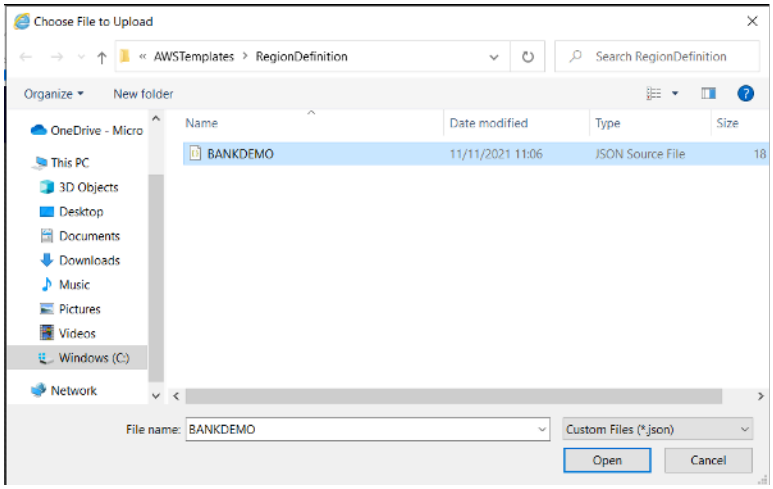
2. Na tela resultante do Enterprise Server Common Web Administration (ESCWA), escolha Importar.



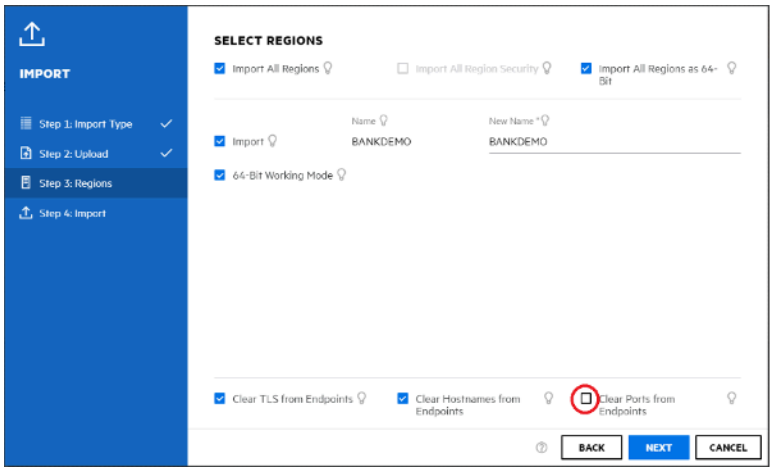
3. Escolha o tipo de importação JSON e escolha Avançar.



4. Faça o upload do arquivo `BANKDEMO.JSON` fornecido.

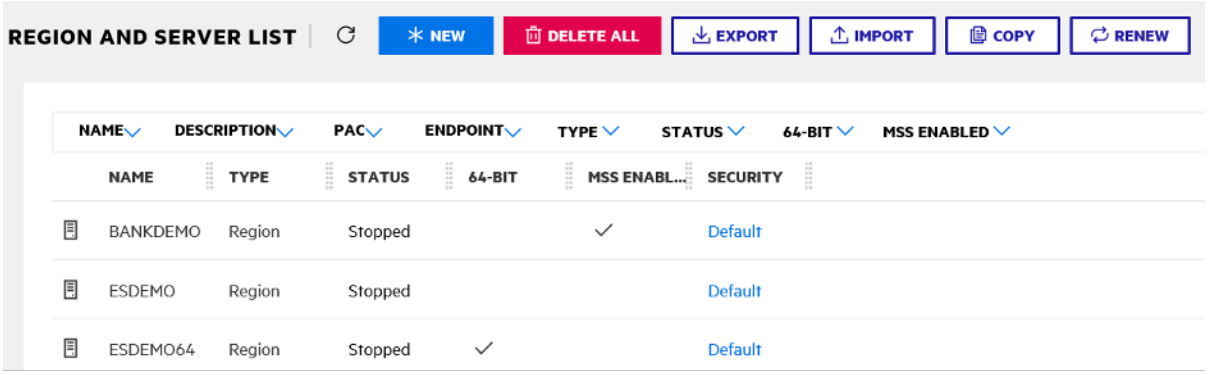


Depois de selecionado, escolha Avançar.

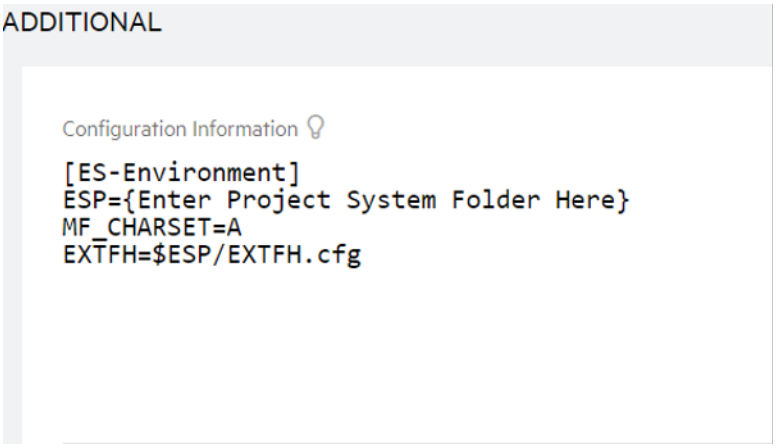


No painel Selecionar regiões, certifique-se de que a opção Limpar portas dos endpoints não esteja selecionada e continue escolhendo Avançar nos painéis até que o painel Executar importação seja exibido. Selecione Import (Importar).

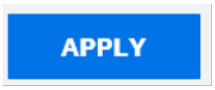
Por fim, clique em Concluir. A região BANKDEMO será então adicionada à lista de servidores.



5. Navegue até as Propriedades Gerais da região BANKDEMO.
6. Role até a seção Configuration (Configuração).
7. A variável de ambiente ESP precisa ser definida na `System` pasta relevante para o Projeto Eclipse criado nas etapas anteriores. Deve ser `workspacefolder/projectname/System`.

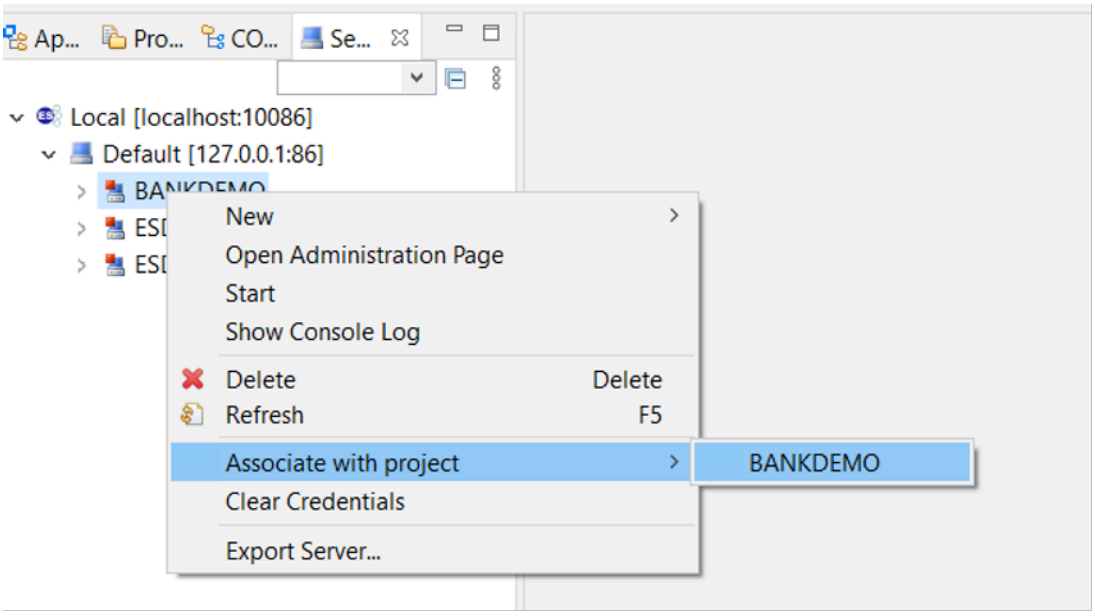


8. Clique em Aplicar.



A região agora está totalmente configurada para ser executada em conjunto com o projeto Eclipse COBOL.

9. Por fim, de volta ao Enterprise Developer, associe a região importada ao projeto.



O ambiente Enterprise Developer agora está pronto para uso, com uma versão funcional completa do BankDemo. Você pode editar, compilar e depurar o código na região.

> **Important**
>
> Se você usar a versão do Enterprise Developer para Windows, os binários gerados pelo compilador poderão ser executados somente no Enterprise Server fornecido com o Enterprise Developer. Você não pode executá-los no tempo de execução da Modernização do AWS Mainframe, que é baseado no Linux.

Tutorial: Configurando a compilação da Micro Focus para o aplicativo BankDemo de amostra

AWS A modernização do mainframe oferece a capacidade de configurar compilações e pipelines de integração contínua/entrega contínua (CI/CD) para seus aplicativos migrados. Essas compilações e pipelines usam AWS CodeBuild AWS CodeCommit, e AWS CodePipeline para fornecer esses recursos. CodeBuild é um serviço de compilação totalmente gerenciado que compila seu código-fonte, executa testes de unidade e produz artefatos prontos para implantação. CodeCommit é um serviço de controle de versão que permite armazenar e gerenciar de forma privada repositórios Git na nuvem. AWS CodePipeline é um serviço de entrega contínua que permite modelar, visualizar e automatizar as etapas necessárias para lançar seu software.

Este tutorial demonstra como usá-lo para AWS CodeBuild compilar o código-fonte do aplicativo de BankDemo amostra do Amazon S3 e depois exportar o código compilado de volta para o Amazon S3.

AWS CodeBuild é um serviço de integração contínua totalmente gerenciado que compila o código-fonte, executa testes e produz pacotes de software prontos para implantação. Com CodeBuild, você pode usar ambientes de compilação predefinidos ou criar ambientes de compilação personalizados que usam suas próprias ferramentas de compilação. Esse cenário de demonstração usa a segunda opção. Ele consiste em um ambiente de CodeBuild construção que usa uma imagem Docker pré-empacotada.

> **Important**
>
> Antes de iniciar seu projeto de modernização de mainframe, recomendamos que você conheça o Programa de Aceleração da Migração (MAP) da AWS para mainframe ou entre em contato com os especialistas em mainframe da AWS para saber mais sobre as etapas necessárias para modernizar uma aplicação de mainframe.

Tópicos

Pré-requisitos

Antes de iniciar este tutorial, conclua os seguintes pré-requisitos.

- Baixe o aplicativo BankDemo de amostra e descompacte-o em uma pasta. A pasta de origem contém programas, cadernos e definições em COBOL. Ele também contém uma pasta JCL para referência, embora você não precise criar o JCL. A pasta também contém os meta-arquivos necessários para a compilação.
- No console de modernização do AWS mainframe, escolha Ferramentas. Em Análise, desenvolvimento e criação de ativos, selecione Compartilhar ativos com minha conta da AWS.

Etapa 1: compartilhar os ativos de construção com a AWS conta

Nesta etapa, você garante compartilhar os ativos de construção com sua AWS conta, especialmente na região em que os ativos estão sendo usados.

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. Na navegação à esquerda, selecione Ferramentas.
3. Em Análise, desenvolvimento e criação de ativos, escolha Compartilhar ativos com minha AWS conta.

> **Important**
>
> Você precisa executar essa etapa uma vez em cada AWS região em que pretende fazer construções.

Etapa 2: criar buckets do Amazon S3

Nesta etapa, você cria dois buckets do Amazon S3. O primeiro é um bucket de entrada para armazenar o código-fonte e o outro é um bucket de saída para armazenar a saída da compilação. Para obter mais informações, consulte Criação, configuração e trabalho com buckets do Amazon S3 no Guia do usuário do Amazon S3.

1. Para criar o bucket de entrada, faça login no console do Amazon S3 e escolha Criar bucket.
2. Em Configuração geral, forneça um nome para o bucket e especifique Região da AWS onde você deseja criar o bucket. Um exemplo de nome é `codebuild-regionId-accountId-`

`input-bucket`, where `regionId` is the Região da AWS do bucket e `accountId` é seu Conta da AWS ID.

> **Note**
>
> Se você estiver criando o bucket em um local diferente Região da AWS do Leste dos EUA (Norte da Virgínia), especifique o `LocationConstraint` parâmetro. Para obter mais informações, consulte Criar Bucket na Referência da API do Amazon Simple Storage Service.

3. Mantenha todas as outras configurações e escolha Criar bucket.
4. Repita as etapas de 1 a 3 para criar o bucket de saída. Um exemplo de nome é `codebuild-regionId-accountId-output-bucket`, where `regionId` is the Região da AWS do bucket e `accountId` é seu Conta da AWS ID.

   Independentemente dos nomes que você escolher para esses buckets, não deixe de usá-los ao longo deste tutorial.

Etapa 3: criar o arquivo de especificação de construção

Nesta etapa, você cria um arquivo de especificações de compilação. Esse arquivo fornece comandos de compilação e configurações relacionadas, no formato YAML, CodeBuild para executar a compilação. Para obter mais informações, consulte a referência da especificação de compilação CodeBuild no Guia AWS CodeBuild do usuário.

1. Crie um arquivo chamado `buildspec.yml` no diretório que você descompactou como pré-requisito.
2. Adicione o seguinte conteúdo ao arquivo e salve. Nenhuma alteração é necessária para este arquivo.

```
version: 0.2
env:
  exported-variables:
    - CODEBUILD_BUILD_ID
    - CODEBUILD_BUILD_ARN
phases:
  install:
    runtime-versions:
```

```
      python: 3.7
  pre_build:
    commands:
      - echo Installing source dependencies...
      - ls -lR $CODEBUILD_SRC_DIR/source
  build:
    commands:
      - echo Build started on `date`
      - /start-build.sh -Dbasedir=$CODEBUILD_SRC_DIR/source -Dloaddir=
$CODEBUILD_SRC_DIR/target
  post_build:
    commands:
      - ls -lR $CODEBUILD_SRC_DIR/target
      - echo Build completed on `date`
artifacts:
  files:
    - $CODEBUILD_SRC_DIR/target/**
```

Aqui`CODEBUILD_BUILD_ID`,`CODEBUILD_BUILD_ARN`,`$CODEBUILD_SRC_DIR/source`, e `$CODEBUILD_SRC_DIR/target` estão as variáveis de ambiente disponíveis em CodeBuild. Para obter mais informações, consulte Variáveis de ambiente em ambientes de compilação.

Nesse ponto, seu diretório deve ter a seguinte aparência.

```
(root directory name)
    |-- build.xml
    |-- buildspec.yml
    |-- LICENSE.txt
    |-- source
         |... etc.
```

3. Compacte o conteúdo da pasta em um arquivo chamado `BankDemo.zip`. Para este tutorial, não é possível compactar a pasta. Em vez disso, compacte o conteúdo da pasta no arquivo `BankDemo.zip`.

Etapa 4: Carregar os arquivos de origem

Nesta etapa, você carrega o código-fonte do aplicativo de BankDemo amostra no seu bucket de entrada do Amazon S3.

1. Faça login no console do Amazon S3 e escolha Buckets no painel de navegação esquerdo. Em seguida, escolha o bucket de entrada que você criou anteriormente.
2. Em Objetos, escolha Carregar.
3. Na seção Arquivos e pastas, escolha Adicionar arquivos.
4. Navegue e escolha seu arquivo `BankDemo.zip`.
5. Escolha Carregar.

Etapa 5: criar políticas do IAM

Nesta etapa, você cria duas políticas do IAM. Uma política concede permissões para a modernização do AWS mainframe acessar e usar a imagem do Docker que contém as ferramentas de criação da Micro Focus. Essa política não é personalizada para clientes. A outra política concede permissões para que a modernização do AWS mainframe interaja com os buckets de entrada e saída e com os CloudWatch registros da Amazon que são gerados. CodeBuild

Para saber mais sobre como criar uma política de IAM, consulte Edição de políticas de IAM no Guia do usuário do IAM.

Para criar uma política para acessar imagens do Docker

1. No console do IAM, copie o seguinte documento de política e cole-o no editor de políticas.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage"
            ],
```

```
            "Resource": "arn:aws:ecr:*:673918848628:repository/m2-enterprise-build-
tools"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::aws-m2-repo-*-<region>-prod"
        }
    ]
}
```

2. Forneça um nome para a política, por exemplo, `m2CodeBuildPolicy`.

Para criar uma política que permita que a modernização do AWS mainframe interaja com buckets e registros

1. No console do IAM, copie o seguinte documento de política e cole-o no editor de políticas. Certifique-se de atualizar `regionId` para o Região da AWS, e `accountId` para o seu Conta da AWS.

```
{
      "Version": "2012-10-17",
      "Statement": [
          {
              "Action": [
                  "logs:CreateLogGroup",
                  "logs:CreateLogStream",
                  "logs:PutLogEvents"
              ],
              "Resource": [
                  "arn:aws:logs:regionId:accountId:log-group:/aws/codebuild/
codebuild-bankdemo-project",
                  "arn:aws:logs:regionId:accountId:log-group:/aws/codebuild/
codebuild-bankdemo-project:*"
              ],
              "Effect": "Allow"
          },
          {
              "Action": [
                  "s3:PutObject",
```

```
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::codebuild-regionId-accountId-input-bucket",
                "arn:aws:s3:::codebuild-regionId-accountId-input-bucket/*",
                "arn:aws:s3:::codebuild-regionId-accountId-output-bucket",
                "arn:aws:s3:::codebuild-regionId-accountId-output-bucket/*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

2. Forneça um nome para a política, por exemplo, `BankdemoCodeBuildRolePolicy`.

Etapa 6: criar uma função do IAM

Nesta etapa, você cria uma nova função do IAM que permite CodeBuild interagir com AWS os recursos para você, depois de associar as políticas do IAM que você criou anteriormente a essa nova função do IAM.

Para obter informações sobre a criação de uma função de serviço, consulte Criação de uma função para delegar permissões a um AWS serviço no Guia do usuário do IAM,.

1. Faça login no console do IAM e escolha Perfis no painel de navegação esquerdo.
2. Selecione Criar função.
3. Em Tipo de entidade confiável, escolha Serviço AWS.
4. Em Casos de uso para outros serviços da AWS, escolha e CodeBuild, em seguida, escolha CodeBuildnovamente.
5. Escolha Próximo.
6. Na página Adicionar permissões, escolha Próximo. Você atribui uma política à função posteriormente.
7. Em Detalhes da função, forneça um nome para a função, por exemplo,`BankdemoCodeBuildServiceRole`.

8. Em Selecionar entidades confiáveis, verifique se o documento de política tem a seguinte aparência:

```
{
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": "codebuild.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        }
```

9. Selecione Criar função.

Etapa 7: anexar as políticas do IAM à função do IAM

Nesta etapa, você anexa as duas políticas do IAM que criou anteriormente ao perfil `BankdemoCodeBuildServiceRole` do IAM.

1. Faça login no console do IAM e escolha Perfis no painel de navegação esquerdo.
2. Em Perfil do IAM, escolha a função que você criou anteriormente, por exemplo, `BankdemoCodeBuildServiceRole`.
3. Em Políticas de permissões, escolha Adicionar permissões e, em seguida, Anexar políticas.
4. Em Outras políticas de permissões, escolha as políticas que você criou anteriormente, por exemplo, `m2CodeBuildPolicy` e `BankdemoCodeBuildRolePolicy`.
5. Escolha Anexar políticas.

Etapa 8: criar o CodeBuild projeto

Nesta etapa, você cria o CodeBuild projeto.

1. Faça login no CodeBuild console e escolha Criar projeto de compilação.

2. Na seção Configuração do projeto, forneça um nome para o projeto, por exemplo, `codebuild-bankdemo-project`.
3. Na seção Fonte, em Provedor de origem, escolha Amazon S3 e, em seguida, escolha o bucket de entrada que você criou anteriormente, por exemplo, `codebuild-regionId-accountId-input-bucket`.
4. No campo Chave do objeto do S3 ou pasta do S3, insira o nome do arquivo zip que você carregou no bucket do S3. Nesse caso, o nome do arquivo é `bankdemo.zip`.
5. Na seção Ambiente, escolha Imagem personalizada.
6. No campo Tipo de ambiente, escolha Linux.
7. Em Registro de imagens, escolha Outro registro.
8. No campo URL do registro externo,
   - Para Micro Focus v9: Enter`673918848628.dkr.ecr.us-west-1.amazonaws.com/m2-enterprise-build-tools:9.0.7.R1`. Se você estiver usando uma AWS região diferente com o Micro Focus v9, você também pode especificar `673918848628.dkr.ecr.<m2-region>.amazonaws.com/m2-enterprise-build-tools:9.0.7.R1` onde <m2-region>está uma AWS região na qual o serviço de modernização de AWS mainframe está disponível (por exemplo,). `eu-west-3`
   - Para Micro Focus v8: Enter `673918848628.dkr.ecr.us-west-2.amazonaws.com/m2-enterprise-build-tools:8.0.9.R1`
   - Para Micro Focus v7: Enter `673918848628.dkr.ecr.us-west-2.amazonaws.com/m2-enterprise-build-tools:7.0.R10`
9. Em Perfil de serviço, escolha Perfil de serviço existente e, no campo ARN o perfil, escolha o perfil de serviço que você criou anteriormente; por exemplo, `BankdemoCodeBuildServiceRole`.
10. Na seção Buildspec, escolha Usar um arquivo buildspec.
11. Na seção Artefatos, em Tipo, escolha Amazon S3 e, em seguida, escolha seu bucket de saída, por exemplo, `codebuild-regionId-accountId-output-bucket`.
12. No campo Nome, insira o nome de uma pasta no bucket que você deseja que contenha os artefatos de saída da compilação, por exemplo, `bankdemo-output.zip`.
13. Em Embalagem de artefatos, escolha Zip.
14. Selecione Create build project (Criar projeto de compilação).

Etapa 9: iniciar a compilação

Nesta etapa, você inicia a compilação.

1. Faça login no CodeBuild console.
2. No painel de navegação esquerdo, selecione Criar projetos.
3. Escolha o projeto de compilação que você criou anteriormente, por exemplo, `codebuild-bankdemo-project`.
4. Selecione Start build.

Esse comando inicia a compilação. A compilação é executada de forma assíncrona. A saída do comando é um JSON que inclui o id do atributo. Esse id de atributo é uma referência ao id de CodeBuild compilação da compilação que você acabou de iniciar. Você pode ver o status da compilação no CodeBuild console. Você também pode ver logs detalhados sobre a execução da compilação no console. Para obter mais informações, consulte Exibir informações detalhadas da compilação no Guia do usuário do AWS CodeBuild .

Quando a fase atual for CONCLUÍDA, significa que sua compilação foi concluída com sucesso e que seus artefatos compilados estão prontos no Amazon S3.

Etapa 10: Baixar artefatos de saída

Nesta etapa, você baixa os artefatos de saída do Amazon S3. A ferramenta de compilação da Micro Focus pode criar vários tipos diferentes de executáveis. Neste tutorial, ele gera objetos compartilhados.

1. Faça login no console do Amazon S3.
2. Na seção Buckets role="bold">, escolha o nome do seu bucket de saída, por exemplo,. `codebuild-regionId-accountId-output-bucket`
3. Escolha Baixar role="bold">.
4. Não descompacte o arquivo obtido por download. Navegue até a pasta de destino para ver os artefatos de compilação. Isso inclui os objetos compartilhados do `.so` Linux.

Limpar os recursos

Se você não precisar mais dos recursos que criou para este tutorial, exclua-os para evitar cobranças adicionais. Para fazer isso, realize as etapas a seguir:

- Exclua os buckets do S3 que você criou para este tutorial. Para obter mais informações, consulte Excluir um bucket no Guia do usuário do Amazon Simple Storage Service.
- Exclua as políticas de IAM que você criou para este tutorial. Para obter mais informações, consulte Exclusão de políticas do IAM no Guia do usuário do IAM.
- Exclua o perfil do IAM que você criou para este tutorial. Para obter mais informações sobre como excluir uma função, consulte Excluir funções ou perfis de instância no Manual do usuário do IAM.
- Exclua o CodeBuild projeto que você criou para este tutorial. Para obter mais informações, consulte Excluir um projeto de compilação CodeBuild no Guia AWS CodeBuild do usuário.

Tutorial: Configurando um pipeline de CI/CD para uso com o Micro Focus Enterprise Developer

Este tutorial mostra como importar, editar, compilar e executar o aplicativo de BankDemo amostra no Micro Focus Enterprise Developer e, em seguida, confirmar suas alterações para acionar um pipeline de CI/CD.

Sumário

Pré-requisitos

Baixe os arquivos a seguir.

- `basic-infra.yaml`
  - Baixe da região Europa (Frankfurt).
  - Baixe da região Leste dos EUA (Norte da Virgínia).
- `pipeline.yaml`
  - Baixe da região Europa (Frankfurt).
  - Baixe da região Leste dos EUA (Norte da Virgínia).
- `m2-code-sync-function.zip`
  - Baixe da região Europa (Frankfurt).
  - Baixe da região Leste dos EUA (Norte da Virgínia).
- `config_git.yml`
  - Baixe da região Europa (Frankfurt).
  - Baixe da região Leste dos EUA (Norte da Virgínia).
- `BANKDEMO-source.zip`
  - Baixe da região Europa (Frankfurt).
  - Baixe da região Leste dos EUA (Norte da Virgínia).
- `BANKDEMO-exercise.zip`
  - Baixe da região Europa (Frankfurt).

- Baixe da região Leste dos EUA (Norte da Virgínia).

A finalidade de cada arquivo é a seguinte:

`basic-infra.yaml`

Esse AWS CloudFormation modelo cria a infraestrutura básica necessária para o pipeline de CI/CD: VPC, buckets Amazon S3 e assim por diante.

`pipeline.yaml`

Esse AWS CloudFormation modelo é usado por uma função Lambda para iniciar a pilha do pipeline. Certifique-se de que esse modelo esteja localizado em um bucket do Amazon S3 acessível publicamente. Adicione o link para esse bucket como o valor padrão para o `PipelineTemplateURL` parâmetro no modelo `basic-infra.yaml`.

`m2-code-sync-function.zip`

Essa função Lambda cria o CodeCommit repositório, a estrutura de diretórios com base no`config_git.yaml`, e inicia a pilha de pipeline usando. `pipeline.yaml` Certifique-se de que esse arquivo zip esteja disponível em um bucket do Amazon S3 acessível ao público em todos os Regiões da AWS lugares onde a modernização de AWS mainframe é suportada. Recomendamos que você armazene o arquivo em um bucket em um Região da AWS e o replique em buckets em todos. Regiões da AWS Use uma convenção de nomenclatura para o intervalo com um sufixo que identifique o específico Região da AWS (por exemplo,`m2-cicd-deployment-source-eu-west-1`) e adicione o prefixo `m2-cicd-deployment-source` como valor padrão para o parâmetro `DeploymentSourceBucket` e forme o intervalo completo usando a função de AWS CloudFormation substituição `!Sub {DeploymentSourceBucket}-${AWS::Region}` ao se referir a esse intervalo no modelo de recurso. `basic-infra.yaml` `SourceSyncLambdaFunction`

`config_git.yml`

CodeCommit definição da estrutura de diretórios. Para ter mais informações, consulte Exemplo de arquivo acionador YAML config_git.yml.

`BANKDEMO-source.zip`.

BankDemo código-fonte e arquivo de configuração criados a partir do CodeCommit repositório.

`BANKDEMO-exercise.zip`.

BankDemo fonte para exercícios tutoriais criados a partir do CodeCommit repositório.

Crie uma infraestrutura básica de pipeline de CI/CD

Use o AWS CloudFormation modelo `basic-infra.yaml` para criar a pilha de infraestrutura básica do pipeline de CI/CD por meio do console. AWS CloudFormation Essa pilha cria buckets do Amazon S3 nos quais você carrega o código e os dados do seu aplicativo e uma função de AWS Lambda suporte para criar outros recursos necessários, como AWS CodeCommit um repositório e um pipeline. AWS CodePipeline

> **Note**
>
> Para iniciar essa pilha, você precisa de permissões para administrar o IAM, o Amazon S3, o Lambda AWS CloudFormation e as permissões de uso. AWS KMS

1. Faça login no AWS Management Console e abra o AWS CloudFormation console em https://console.aws.amazon.com/cloudformation.
2. Crie uma nova pilha usando uma das seguintes opções:
   - Selecione Create Stack (Criar pilha). Esta será a única opção se houver uma pilha em execução no momento.
   - Na página Pilhas, selecione Criar pilha. Essa opção aparecerá somente se não houver pilhas em execução.
3. Na página Especificar modelo:
   - Em Preparar modelo, selecione O modelo está pronto.
   - Em Especificar modelo, escolha o URL do Amazon S3 como fonte do modelo e insira um dos seguintes URLs, dependendo da sua Região da AWS.
     - `https://m2-us-east-1.s3.us-east-1.amazonaws.com/cicd/mf/basic-infra.yaml`
     - `https://m2-eu-central-1.s3.eu-central-1.amazonaws.com/cicd/mf/basic-infra.yaml`
   - Para aceitar suas configurações, selecione Próximo.

   A página Criar pilha é aberta.

Specify stack details

Stack name

Stack name

m2-cicd-pipeline-foundation

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

Networking Configuration

Do you want to use an existing VPC in your account?

If you select 'Yes', then you must provide the VPC ID and the Subnet IDs.

No (Create one)

Which VPC ID should be used?

If you selected 'Yes' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

Which private subnet ID should be used?

If you selected 'Yes' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

Which private subnet ID in a different AZ should be used for HA?

If you selected 'Yes' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

Enter the CIDR block that should be used for the new VPC

If you selected 'No (Create one)' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

11.11.0.0/16

CIDR bits for creating subnets. Choose 5 for /27, 6 for /26, 7 for /25, 8 for /24 range

If you selected 'No (Create one)' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

5

Deployment Configuration

Name of the S3 bucket which contains the source files for this stack deployment

Don't change unless you know what you are doing.

m2

Name of the source package file for the infrastructure Lambda function

Don't change unless you know what you are doing.

cicd/mf/m2-code-sync-function.zip

Full URL of the pipeline CloudFormation template file

Don't change unless you know what you are doing.

https://m2-us-east-1.s3.us-east-1.amazonaws.com/cicd/mf/pipeline.yaml

What name prefix to use for the new S3 buckets?

A name prefix for the S3 buckets that will be created by this stack.

mf

Cancel Previous Next

Faça as seguintes alterações em:

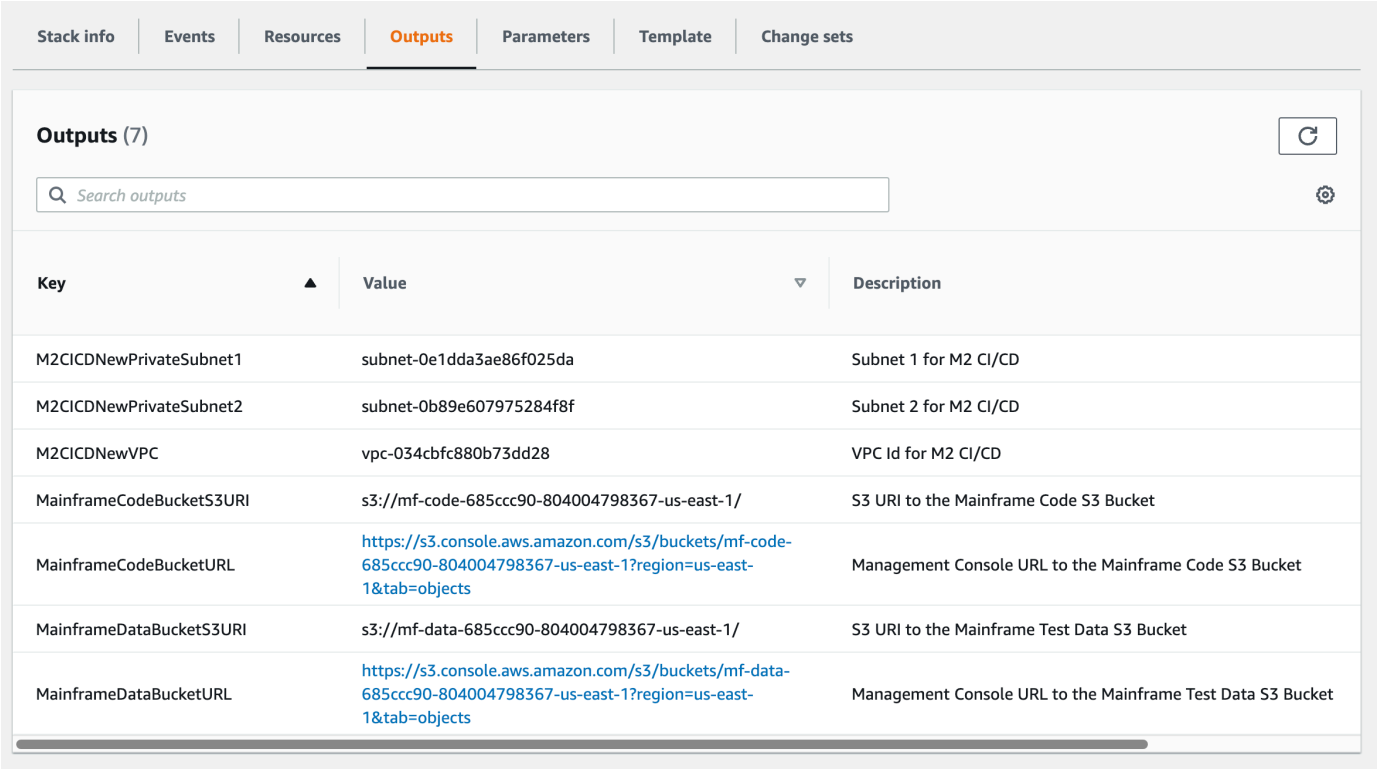
- Forneça os valores apropriados para o Nome da pilha e os parâmetros para a Configuração de rede.
- A maioria dos parâmetros nas Configurações de implantação são pré-preenchidos adequadamente para que você não precise modificá-los. Dependendo do seu Região da AWS, altere o AWS CloudFormation modelo do pipeline para um dos seguintes URLs do Amazon S3.
  - `https://m2-us-east-1.s3.amazonaws.com/cicd/mf/pipeline.yaml`
  - `https://m2-eu-central-1.s3.eu-central-1.amazonaws.com/cicd/mf/pipeline.yaml`
- Selecione Next (Próximo).

> **Note**
>
> Não altere os valores dos parâmetros padrão, a menos que você mesmo tenha modificado o AWS CloudFormation modelo.

4. Em Configurar opções de pilha, selecione Próximo.
5. Em Capacidades, escolha Eu reconheço que AWS CloudFormation posso criar recursos do IAM para permitir AWS CloudFormation a criação da função do IAM em seu nome. Selecione Criar pilha.

> **Note**
>
> Pode levar de 3 a 5 minutos para que essa pilha seja provisionada.

6. Depois que a pilha for criada com sucesso, navegue até a seção Saídas da pilha recém-provisionada. Lá você encontrará o bucket do Amazon S3 onde você precisa fazer o upload do código do mainframe e dos arquivos dependentes.



Crie AWS CodeCommit repositório e pipeline de CI/CD

Nesta etapa, você cria um CodeCommit repositório e provisiona uma pilha de pipeline de CI/CD chamando uma função Lambda que chama AWS CloudFormation para criar a pilha de pipeline.

1. Baixe o aplicativo BankDemo de amostra em sua máquina local.
2. Faça o upload de `bankdemo.zip` da sua máquina local para o bucket Amazon S3 criado em Crie uma infraestrutura básica de pipeline de CI/CD.
3. Baixar `config_git.yml`.
4. Modifique o `config_git.yml`, se necessário, da seguinte maneira:

   - Adicione seu próprio nome de repositório de destino, ramificação de destino e mensagem de confirmação.

   ```
   repository-config:
     target-repository: bankdemo-repo
     target-branch: main
     commit-message: Initial commit for bankdemo-repo main branch
   ```

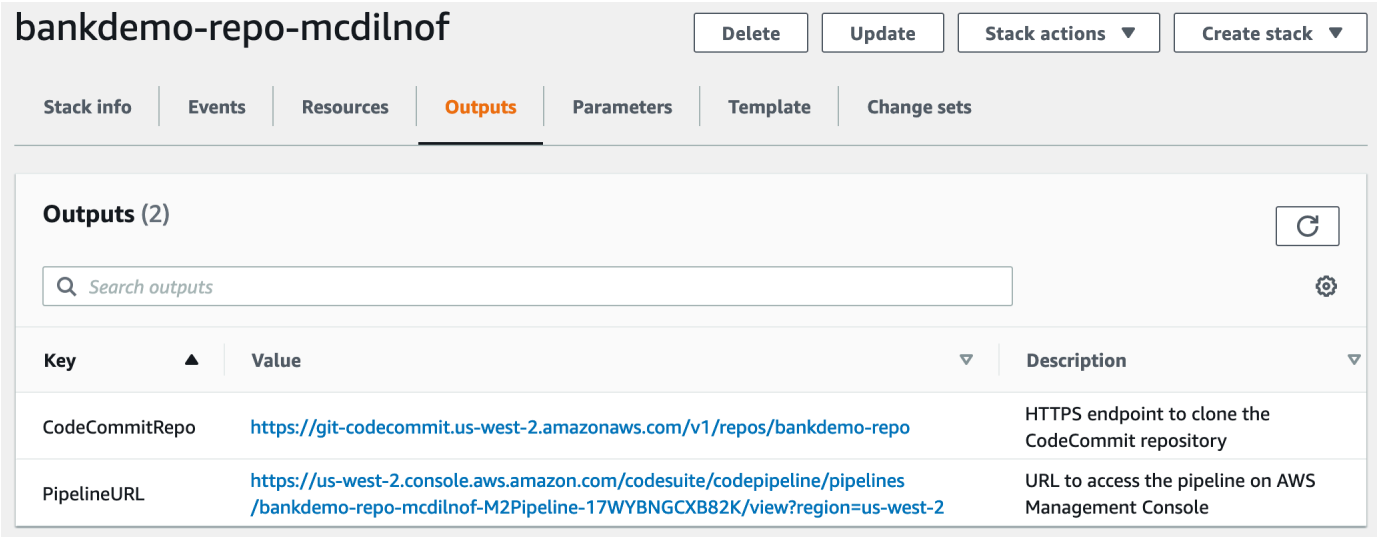
   - Adicione o endereço de e-mail que você deseja receber notificações.

```
pipeline-config:
  # Send pipeline failure notifications to these email addresses
  alert-notifications:
    - myname@mycompany.com
  # Send notifications for manual approval before production deployment to these
 email addresses
  approval-notifications:
    - myname@mycompany.com
```

5. Faça o upload do `config_git.yml` arquivo contendo a definição da estrutura da pasta do CodeCommit repositório para o bucket do Amazon S3 criado em. Crie uma infraestrutura básica de pipeline de CI/CD Isso invocará a função do Lambda que provisionará automaticamente o repositório e o pipeline.

   Isso criará um CodeCommit repositório com o nome fornecido no `target-repository` definido no `config_git.yml` arquivo; por exemplo,`bankdemo-repo`.

   A função Lambda também criará a pilha de pipeline de CI/CD. AWS CloudFormation A pilha AWS CloudFormation terá o mesmo prefixo que o nome `target-repository` fornecido, seguido de uma cadeia aleatória (por exemplo, `bankdemo-repo-`*`01234567`*. Você pode encontrar a URL do CodeCommit repositório e a URL para acessar o pipeline criado no AWS Management Console.



6. Se a criação do CodeCommit repositório for concluída, o pipeline de CI/CD será acionado imediatamente para executar um CI/CD completo.

7. Depois que o arquivo for enviado, ele acionará automaticamente o pipeline, que será construído, implantado em teste, executará alguns testes e aguardará a aprovação manual antes de implantá-lo no ambiente de produção.

Exemplo de arquivo acionador YAML config_git.yml

```
repository-config:
  target-repository: bankdemo-repo
  target-branch: main
  commit-message: Initial commit for bankdemo-repo main branch
  directory-structure:
    - '/':
        files:
        - build.xml
        - '*.yaml'
        - '*.yml'
        - '*.xml'
        - 'LICENSE.txt'
        readme: |
          # Root Folder
          - 'build.xml' : Build configuration for the application
    - tests:
        files:
        - '*.py'
        readme: |
          # Test Folder
          - '*.py' : Test scripts
    - config:
        files:
        - 'BANKDEMO.csd'
        - 'BANKDEMO.json'
        - 'BANKDEMO_ED.json'
        - 'dfhdrdat'
        - 'ESPGSQLXA.dll'
        - 'ESPGSQLXA64.so'
        - 'ESPGSQLXA64_S.so'
        - 'EXTFH.cfg'
        - 'm2-2021-04-28.normal.json'
        - 'MFDBFH.cfg'
        - 'application-definition-template-config.json'
        readme: |
          # Config Folder
```

```
          This folder contains the application configuration files.
          - 'BANKDEMO.csd'     : CICS Resource definitions export file
          - 'BANKDEMO.json'    : Enterprise Server configuration
          - 'BANKDEMO_ED.json' : Enterprise Server configuration for ED
          - 'dfhdrdat'         : CICS resource definition file
          - 'ESPGSQLXA.dll'    : XA switch module Windows
          - 'ESPGSQLXA64.so'   : XA switch module Linux
          - 'ESPGSQLXA64_S.so' : XA switch module Linux
          - 'EXTFH.cfg'        : Micro Focus File Handler configuration
          - 'm2-2021-04-28.normal.json' : M2 request document
          - 'MFDBFH.cfg'       : Micro Focus Database File Handler
          - 'application-definition-template-config.json' : Application definition for
 M2
    - source:
        subdirs:
        - .settings:
            files:
              - '.bms.mfdirset'
              - '.cbl.mfdirset'
        - copybook:
            files:
              - '*.cpy'
              - '*.inc'
            readme: |
              # Copy folder
              This folder contains the source for COBOL copy books, PLI includes, ...
              - .cpy COBOL copybooks
              - .inc PLI includes
#        - ctlcards:
#            files:
#              - '*.ctl'
#              - 'KBNKSRT1.txt'
#            readme: |
#              # Control Card folder
#              This folder contains the source for Batch Control Cards
#              - .ctl Control Cards
        - ims:
            files:
              - '*.dbd'
              - '*.psb'
            readme: |
              # ims folder
              This folder contains the IMS DB source files with the extensions
              - .dbd for IMS DBD source
```

```
              - .psb for IMS PSB source
        - jcl:
            files:
              - '*.jcl'
              - '*.ctl'
              - 'KBNKSRT1.txt'
              - '*.prc'
            readme: |
              # jcl folder
              This folder contains the JCL source files with the extensions
              - .jcl
#        - proclib:
#            files:
#              - '*.prc'
#            readme: |
#              # proclib folder
#              This folder contains the JCL procedures referenced via PROCLIB
 statements in the JCL with extensions
#              - .prc
        - rdbms:
            files:
              - '*.sql'
            readme: |
              # rdbms folder
              This folder contains any DB2 related source files with extensions
              - .sql for any kind of SQL source
        - screens:
            files:
            - '*.bms'
            - '*.mfs'
            readme: |
              # screens folder
              This folder contains the screens source files with the extensions
              - .bms for CICS BMS screens
              - .mfs for IMS MFS screens
            subdirs:
            - .settings:
                files:
                    - '*.bms.mfdirset'
        - cobol:
            files:
              - '*.cbl'
              - '*.pli'
            readme: |
```

```
              # source folder
              This folder contains the program source files with the extensions
              - .cbl for COBOL source
              - .pli for PLI source
            subdirs:
            - .settings:
                files:
                   - '*.cbl.mfdirset'
    - tests:
        files:
        - 'test_script.py'
        readme: |
          # tests Folder
          This folder contains the application test scripts
pipeline-config:
  alert-notifications:
    - myname@mycompany.com
  approval-notifications:
    - myname@mycompany.com
```

Criação do Enterprise Developer AppStream 2.0

Para configurar o Micro Focus Enterprise Developer na AppStream versão 2.0, consulteTutorial: Configurar o Micro Focus Enterprise Developer na AppStream versão 2.0.

Para conectar o CodeCommit repositório ao Enterprise Developer, use o nome especificado `target-repository` emExemplo de arquivo acionador YAML config_git.yml.

Configuração e teste para desenvolvedores corporativos

Tópicos

- Clone o BankDemo CodeCommit repositório no Enterprise Developer
- Crie um projeto COBOL de BankDemo mainframe e crie um aplicativo
- Crie um BankDemo CICS local e um ambiente de lote para testes
- Inicie o servidor BANKDEMO do Enterprise Developer
- Inicie o terminal Rumba 3270
- Executar uma BankDemo transação
- Interrompa o servidor BANKDEMO no Enterprise Developer

Conecte-se à instância do Enterprise Developer AppStream 2.0 em que você criouCriação do Enterprise Developer AppStream 2.0.

1. Inicie o Enterprise Developer a partir do Windows Start. Escolha Micro Focus Enterprise Developer e, em seguida, escolha Enterprise Developer for Eclipse. Se você está começando pela primeira vez, pode levar algum tempo.
2. No Eclipse Launcher, no Workspace: entre e `C:\Users\<username>\workspace` escolha Launch.

   > **Note**
   >
   > Certifique-se de escolher o mesmo local depois de se reconectar à instância AppStream 2.0. A seleção do espaço de trabalho não é persistente.

3. Em Bem-vindo, escolha Open COBOL Perspective. Isso só será exibido na primeira vez em um novo espaço de trabalho.

Clone o BankDemo CodeCommit repositório no Enterprise Developer

1. Escolha Janela / Perspectiva / Abrir perspectiva / Outro ... / Git.
2. Escolha Clonar um repositório Git.
3. Em Clonar repositório Git, insira as seguintes informações:

   - Em URI de localização, insira a URL HTTPS do CodeCommit repositório.

     > **Note**
     >
     > Copie o URL do clone HTTPS para o CodeCommit repositório no AWS Management Console e cole-o aqui. O URI será dividido nos caminhos Host e Repositório.

   - As credenciais do CodeCommit repositório do usuário em Usuário e Senha de Autenticação e escolha Armazenar no Armazenamento Seguro.
4. Em Seleção de ramificação, escolha Ramificação principal e, em seguida, escolha Próximo.
5. Em Destino local, em Diretório, insira `C:\Users\<username>\workspace` e escolha Concluir.

   O processo de clonagem é concluído quando `BANKDEMO [main]` exibido na visualização Repositórios Git.

Crie um projeto COBOL de BankDemo mainframe e crie um aplicativo

1. Altere para a perspectiva COBOL.
2. Em Projeto, desative Criar automaticamente.
3. Em Arquivo, escolha Novo e, em seguida, Projeto COBOL Mainframe.
4. Em Novo projeto COBOL mainframe, insira as seguintes informações:
   - Em Nome do projeto, digite `BankDemo`.
   - Escolha o modelo Micro Focus [64 bits].
   - Escolha Terminar.
5. No COBOL Explorer, expanda o novo BankDemo projeto.

   > **Note**
   >
   > `[BANKDEMO main]`entre colchetes indica que o projeto está conectado ao BankDemo CodeCommit repositório local.

6. Se a visualização em árvore não mostrar entradas para programas COBOL, cadernos, código-fonte BMS e arquivos JCL, escolha Atualizar no menu de contexto do projeto. BankDemo
7. BankDemo No menu de contexto, escolha Propriedades/Micro Focus/Configurações do projeto/COBOL:
   - Escolha Conjunto de caracteres: ASCII.
   - Escolha Aplicar e, em seguida, Fechar.
8. Se a compilação da fonte BMS e COBOL não for iniciada imediatamente, verifique no menu Projeto se a opção Criar Automaticamente está ativada.

   A saída do Build será exibida na visualização do Console e deverá ser concluída após alguns minutos com mensagens `BUILD SUCCESSFUL` e `Build finished with no errors`.

   O BankDemo aplicativo agora deve estar compilado e pronto para execução local.

Crie um BankDemo CICS local e um ambiente de lote para testes

1. No COBOL Explorer, expanda`BANKDEMO / config`.
2. No editor, abra `BANKDEMO_ED.json`.

3. Localize a string `ED_Home=` e altere o caminho para apontar para o projeto Enterprise Developer, da seguinte forma: `D:\\<username>\\workspace\\BANKDEMO`. Observe o uso de barras duplas (`\\`) na definição do caminho.
4. Salve e feche o arquivo.
5. Escolha Server Explorer.
6. No menu de contexto Padrão, escolha Abrir página de administração. A página de administração do Micro Focus Enterprise Server é aberta no navegador padrão.
7. Somente para sessões AppStream 2.0, faça as seguintes alterações para que você possa preservar sua região local do Enterprise Server para testes locais:
   - Em Servidor de Diretórios / Padrão, escolha PROPRIEDADES / Configuração.
   - Substitua a localização do repositório por`D:\<username>\My Files\Home Folder\MFDS`.

   > **Note**
   >
   > Você deve concluir as etapas 5 a 8 após cada nova conexão com uma instância AppStream 2.0.

8. Em Servidor de Diretórios / Padrão, escolha Importar e conclua as seguintes etapas:
   - Na Etapa 1: tipo de importação, escolha JSON e escolha Próximo.
   - Na Etapa 2: carregar, clique para fazer upload do arquivo no quadrado azul.
   - Em Escolher arquivo para carregar, digite:
     - Nome do arquivo: `D:\<username>\workspace\BANKDEMO\config\BANKDEMO_ED.json`.
     - Escolha Open (Abrir).
   - Escolha Next (Próximo).
   - Na Etapa 3: Regiões limpam as portas limpas dos endpoints.
   - Selecione Next (Próximo).
   - Na Etapa 4: importar, escolha Importar.
   - Escolha Terminar.

   A lista agora mostrará um novo nome de servidor `BANKDEMO`.

Inicie o servidor BANKDEMO do Enterprise Developer

1. Escolha Enterprise Developer.
2. No Explorador do servidor, escolha Padrão e, em seguida, escolha Atualizar no menu de contexto.

   A lista de servidores agora também deve mostrar BANKDEMO.
3. Escolha BANKDEMO.
4. No menu de contexto, escolha Associar ao projeto e escolha BANKDEMO.
5. No menu contextual, escolha Iniciar.

   A visualização do console deve exibir o log da inicialização do servidor.

   Se a mensagem `BANKDEMO CASSI5030I PLTPI Phase 2 List(PI) Processing Completed` for exibida, o servidor está pronto para testar a aplicação CICS BANKDEMO.

Inicie o terminal Rumba 3270

1. No Windows Start, inicie o Micro Focus Rumba+ Desktop /Rumba+ Desktop.
2. Em Bem-vindo, escolha CREATE NEW SESSION/Mainframe Display.
3. No Mainframe Display, escolha Conexão / Configurar.
4. Em Configuração da sessão, escolha Connection/TN3270.
5. Em Nome do host/Endereço, escolha Inserir e insira o endereço IP `127.0.0.1`.
6. Em Porta Telnet, insira porta `6000`.
7. Selecione Apply (Aplicar).
8. Selecione Conectar.

   A tela de boas-vindas do CICS exibe a tela com a mensagem da linha 1: `This is the Micro Focus MFE CICS region BANKDEMO`.
9. Pressione Ctrl+Shift+z para limpar a tela.

Executar uma BankDemo transação

1. Em uma tela vazia, digite `BANK`.
2. Na tela BANK10, no campo de entrada para ID do usuário...:, digite `guest` e pressione Enter.

3. Na tela BANK20, no campo de entrada antes de Calcular o custo de um empréstimo, digite `/` (barra) e pressione Enter.
4. Na tela BANK70:

   - Em O valor que você gostaria de emprestar...:, digite `10000`.
   - Em A uma taxa de juros de...:, digite `5.0`.
   - Em Por quantos meses...:, digite `10`.
   - Pressione Enter.

   O seguinte resultado deve ser exibido:

   ```
   Resulting monthly payment.............:   $1023.06
   ```

   Isso conclui a configuração da aplicação BANKDEMO no Enterprise Developer.

Interrompa o servidor BANKDEMO no Enterprise Developer

1. No Explorador do servidor, escolha Padrão e, em seguida, escolha Atualizar no menu de contexto.
2. Escolha BANKDEMO.
3. No menu de contexto, selecione Interromper.

   A visualização do Console deve exibir o log da parada do servidor.

   Se a mensagem `Server: BANKDEMO stopped successfully` for exibida, o servidor foi desligado com sucesso.

Exercício 1: Aprimorar o cálculo do empréstimo na aplicação BANKDEMO

Tópicos

- Etapa 4: confirmar as alterações e executar o pipeline de CI/CD

Nesse cenário, você percorre o processo de fazer uma alteração de amostra no código, implantá-lo e testá-lo.

O departamento de empréstimos deseja um novo campo na tela de cálculo do empréstimo BANK70 para mostrar o valor total do empréstimo. Isso requer uma alteração na tela BMS MBANK70.CBL, adicionando um novo campo e o programa de tratamento de tela correspondente SBANK70P.CBL com cadernos relacionados. Além disso, a rotina de cálculo do empréstimo no BBANK70P.CBL precisa ser estendida com a fórmula adicional.

Para concluir este exercício, preencha os pré-requisitos a seguir.

- Faça o download de BANKDEMO-exercise.zip em `D:\PhotonUser\My Files\Home Folder`.
- Extraia o arquivo zip para `D:\PhotonUser\My Files\Home Folder\BANKDEMO-exercise`.
- Crie a pasta `D:\PhotonUser\My Files\Home Folder\AnalysisRules`.
- Copie o arquivo `Loan+Calculation+Update.General-1.xml` de regras da `BANKDEMO-exercise` pasta para `D:\PhotonUser\My Files\Home Folder\AnalysisRules`.

> **Note**
>
> As alterações de código em *.CBL e *.CPY são marcadas com EXER01 nas colunas 1 a 6 deste exercício.

Adicionar regra de análise de empréstimos à Enterprise Developer Code Analysis

As regras de análise definidas no Micro Focus Enterprise Analyzer podem ser exportadas do Enterprise Analyzer e importadas para o Enterprise Developer para executar as mesmas regras de análise nas fontes do projeto Enterprise Developer.

1. Abra o `Window/Preferences/Micro Focus/COBOL/Code Analysis/Rules`.
2. Escolha Editar... e insira o nome da pasta `D:\PhotonUser\My Files\Home Folder\AnalysisRules` que contém o arquivo de regras `Loan+Calculation +Update.General-1.xml`.
3. Escolha Terminar.

4. Escolha Aplicar e depois Fechar.
5. No menu de contexto do projeto BANKDEMO, escolha Análise de código.

   Você deve ver uma entrada para Atualização do cálculo do empréstimo.

Etapa 1: realizar a análise do código para o cálculo do empréstimo

Com a nova regra de análise, queremos identificar os programas COBOL e as linhas de código que correspondem aos padrões de pesquisa `*PAYMENT*`, `*LOAN*` e `*RATE*` em expressões, instruções e variáveis. Isso ajudará a navegar pelo código e identificar as alterações de código necessárias.

1. No menu de contexto do projeto BANKDEMO, selecione Code Analysis/Loan Calculation Update.

   Isso executará a regra de pesquisa e listará os resultados em uma nova guia chamada Análise de código. A execução da análise é concluída quando a barra de progresso verde no canto inferior direito desaparece.

   A guia Análise de código deve exibir uma lista expandida de `BBANK20P.CBL`, `BBANK70P.CBL` e `SBANK70P.CBL`, e cada uma listando as declarações, expressões e variáveis que correspondem aos padrões de pesquisa.

   Observando o resultado, `BBANK20P.CBL` há apenas literais movidos que correspondem ao padrão de pesquisa. Portanto, esse programa pode ser ignorado.
2. Na barra de menu da guia, escolha - Ícone para recolher tudo.
3. Expanda `SBANK70P.CBL` e selecione qualquer linha em qualquer ordem com um clique duplo para ver como isso abrirá a fonte e destacará a linha selecionada no código-fonte. Você também reconhecerá que todas as linhas de origem identificadas estão marcadas.

Etapa 2: modificar o mapa do CICS BMS e o programa e teste COBOL

Primeiro, alteraremos o mapa `MBANK70.BMS` do BMS, o programa de manipulação de tela `SBANK70P.CBL` e o caderno `CBANKDAT.CPY` para exibir o novo campo. Para evitar codificação desnecessária neste exercício, os módulos de origem modificados estão disponíveis na pasta `D:\PhotonUser\My Files\Home Folder\BANKDEMO-exercise\Exercise01`. Normalmente, um desenvolvedor usaria os resultados da Análise de Código para navegar e modificar as fontes.

Se você tiver tempo e quiser fazer as alterações manuais, faça-o com as informações fornecidas em *Alteração manual no MBANK70.BMS e no SBANK70P.CBL (opcional) *.

Para alterações rápidas, copie os seguintes arquivos:

1. `..\BANKDEMO-exercise\Exercis01\screens\MBANK70.BMS` para `D:\PhotonUser\workspace\bankdemo\source\screens`.
2. `.\BANKDEMO-exercise\Exercis01\cobol\SBANK70P.CBL` para `D:\PhotonUser\workspace\bankdemo\source\cobol`.
3. `..\BANKDEMO-exercise\Exercis01\copybook\CBANKDAT.CPY` para `D:\PhotonUser\workspace\bankdemo\source\copybook`.
4. Para garantir que todos os programas afetados pelas alterações sejam compilados, escolha Projetar/Limpar... /Limpe todo o projeto.

Para alterações manuais em `MBANK70.BMS` e `SBANK70P.CBL`, conclua as seguintes etapas:

- Para alteração manual na `MBANK70.BMS` fonte BMS, adicione após o campo `PAYMENT`:
  - TXT09 com os mesmos atributos do TXT08 e valor INICIAL “Valor total do empréstimo”
  - TOTAL com os mesmos atributos de PAGAMENTO

Alterações no teste

Para testar as alterações, repita as etapas nas seguintes seções:

1. Inicie o servidor BANKDEMO do Enterprise Developer
2. Inicie o terminal Rumba 3270
3. Executar uma BankDemo transação

   Além disso, agora você também deve ver o texto `Total Loan Amount.....:`.
4. Interrompa o servidor BANKDEMO no Enterprise Developer

Etapa 3: adicionar cálculo do valor total no programa COBOL

Na segunda etapa, alteraremos o `BBANK70P.CBL` e adicionaremos o cálculo do valor total do empréstimo. A fonte preparada com as alterações necessárias está disponível na pasta `D:`

`\PhotonUser\My Files\Home Folder\BANKDEMO-exercise\Exercise01`. Se você tiver tempo e quiser fazer as alterações manuais, faça-o com as informações fornecidas em *Alteração manual no BBANK70P.CBL (opcional) *.

Para uma mudança rápida, copie o seguinte arquivo:

- `..\BANKDEMO-exercise\Exercis01\source\cobol\BBANK70P.CBL` para `D:\PhotonUser\workspace\bankdemo\source\cobol`.

Para fazer uma alteração manual no `BBANK70P.CBL`, conclua as seguintes etapas:

- Use o resultado da Análise de Código para identificar as alterações necessárias.

Alterações no teste

Para testar as alterações, repita as etapas nas seguintes seções:

1. Inicie o servidor BANKDEMO do Enterprise Developer
2. Inicie o terminal Rumba 3270
3. Executar uma BankDemo transação

   Além disso, agora você também deve ver o texto `Total Loan Amount.....................: $10230.60`.
4. Interrompa o servidor BANKDEMO no Enterprise Developer

Etapa 4: confirmar as alterações e executar o pipeline de CI/CD

Confirme as alterações no CodeCommit repositório central e acione o pipeline de CI/CD para criar, testar e implantar as alterações.

1. No projeto BANKDEMO, no menu de contexto, selecione Team/Commit.
2. Na guia Git Staging, digite a seguinte mensagem de confirmação: `Added Total Amount Calculation`.
3. Escolha Confirmar e enviar….
4. Abra o CodePipeline console e verifique o status da execução do pipeline.

> **Note**
>
> Caso você enfrente algum problema com a função Commit ou Push do Enterprise Developer ou do Teams, use a interface de linha de comando do Git Bash.

Exercício 2: Extrair cálculo do empréstimo no BankDemo aplicativo

Tópicos

No próximo exercício, você trabalha com outra solicitação de alteração de amostra. Nesse cenário, o departamento de empréstimos deseja reutilizar a rotina de cálculo do empréstimo de forma autônoma WebService. A rotina deve permanecer em COBOL e também deve ser chamada a partir do programa CICS COBOL existente. `BBANK70P.CBL`

Etapa 1: refatorar a rotina de cálculo do empréstimo em uma seção COBOL

Na primeira etapa, extraímos a rotina de cálculo do empréstimo em uma seção COBOL. Essa etapa é necessária para extrair o código em um programa COBOL independente na próxima etapa.

1. Abra o `BBANK70P.CBL` no editor COBOL.
2. No editor, selecione no menu de contexto Code Analysis/Loan Calculation Update. Isso examinará apenas a fonte atual em busca de padrões definidos na regra de análise.
3. No resultado na guia Análise de código, encontre a primeira declaração aritmética `DIVIDE WS-LOAN-INTEREST BY 12`.
4. Clique duas vezes na declaração para navegar até a linha de origem no Editor. Essa é a primeira declaração da rotina de cálculo do empréstimo.
5. Marque o seguinte bloco de código para que a rotina de cálculo do empréstimo seja extraída em uma seção.

```
DIVIDE WS-LOAN-INTEREST BY 12
            GIVING WS-LOAN-INTEREST ROUNDED.
```

```
           COMPUTE WS-LOAN-MONTHLY-PAYMENT ROUNDED =
             ((WS-LOAN-INTEREST * ((1 + WS-LOAN-INTEREST)
                 ** WS-LOAN-TERM)) /
             (((1 + WS-LOAN-INTEREST) * WS-LOAN-TERM) - 1 ))
                 * WS-LOAN-PRINCIPAL.
EXER01     COMPUTE WS-LOAN-TOTAL-PAYMENT =
EXER01               (WS-LOAN-MONTHLY-PAYMENT * WS-LOAN-TERM).
```

6. No menu de contexto do editor, escolha Refatorar/Extrair para seção....
7. Insira o nome da nova seção: CÁLCULO DO EMPRÉSTIMO.
8. Escolha OK.

   O bloco de código marcado agora foi extraído para a nova `LOAN-CALCULATION` seção e o bloco de código foi substituído pela instrução `PERFROM LOAN-CALCULATION`.

Alterações no teste

Para testar as alterações, repita as etapas descritas nas seções a seguir.

1. Inicie o servidor BANKDEMO do Enterprise Developer
2. Inicie o terminal Rumba 3270
3. Executar uma BankDemo transação

   Além disso, agora você também deve ver o texto `Total Loan Amount.....: $10230.60`.
4. Interrompa o servidor BANKDEMO no Enterprise Developer

> **Note**
>
> Se você quiser evitar as etapas acima para extrair o bloco de código para uma seção, você pode copiar a fonte modificada para a Etapa 1 de `..\BANKDEMO-exercise\Exercis02\Step1\cobol\BBANK70P.CBL` para `D:\PhotonUser\workspace\bankdemo\source\cobol`.

Etapa 2: extrair a rotina de cálculo do empréstimo para um programa COBOL independente

Na Etapa 2, o bloco de código na seção `LOAN-CALCULATION` será extraído para um programa independente e o código original será substituído pelo código para chamar o novo subprograma.

1. Abra `BBANK70P.CBL` no editor e encontre a nova declaração `PERFORM LOAN-CALCULATION` criada na Etapa 1.
2. Coloque o cursor dentro do nome da seção. Estará marcado em cinza.
3. No menu de contexto, selecione Refatorar->Extrair seção/parágrafo para o programa…….
4. Em Extrair seção/parágrafo para o programa, insira o Nome do novo arquivo: LOANCALC.CBL.
5. Escolha OK.

   O novo `LOANCALC.CBL` programa será aberto no editor.
6. Role para baixo e revise o código que está sendo extraído e gerado para a interface de chamada.
7. Selecione o editor com `BBANK70P.CBL` e vá para `LOAN-CALCULATION SECTION`. Revise o código que está sendo gerado para chamar o novo subprograma `LOANCALC.CBL`.

> **Note**
>
> A `CALL` instrução está usando `DFHEIBLK` e `DFHCOMMAREA` para chamar `LOANCALC` com blocos de controle do CICS. Como queremos chamar o novo `LOANCALC.CBL` subprograma de programa não CICS, precisamos remover `DFHEIBLK` e sair `DFHCOMMAREA` da chamada comentando ou excluindo.

Alterações no teste

Para testar as alterações, repita as etapas descritas nas seções a seguir.

1. Inicie o servidor BANKDEMO do Enterprise Developer
2. Inicie o terminal Rumba 3270
3. Executar uma BankDemo transação

   Além disso, agora você também deve ver o texto `Total Loan Amount.....................: $10230.60`.

4. Interrompa o servidor BANKDEMO no Enterprise Developer

> **Note**
>
> Se quiser evitar as etapas acima para extrair o bloco de código em uma seção, você pode copiar a fonte modificada para a Etapa 1 de `..\BANKDEMO-exercise\Exercis02\Step2\cobol\BBANK70P.CBL` e `LOANCALC.CBL` para `D:\PhotonUser\workspace\bankdemo\source\cobol`.

Etapa 3: confirmar as alterações e executar o pipeline de CI/CD

Confirme as alterações no CodeCommit repositório central e acione o pipeline de CI/CD para criar, testar e implantar as mudanças.

1. No projeto BANKDEMO, no menu de contexto, selecione Team/Commit.
2. Na guia Git Staging
   - Adicione os estágios não preparados LOANCALC.CBL e LOANCALC.CBL.mfdirset.
   - Insira uma mensagem de confirmação: `Added Total Amount Calculation`.
3. Escolha Confirmar e enviar….
4. Abra o CodePipeline console e verifique o status da execução do pipeline.

> **Note**
>
> Caso você enfrente algum problema com a função Commit ou Push do Enterprise Developer ou do Teams, use a interface de linha de comando do Git Bash.

Limpar recursos

Se os recursos criados durante este tutorial não forem mais necessários, exclua-os para que você não continue sendo cobrado por eles. Execute as etapas a seguir:

- Exclua o CodePipeline pipeline. Para obter mais informações, consulte Excluir um pipeline CodePipeline no Guia AWS CodePipeline do usuário.

- Exclua o CodeCommit repositório. Para obter mais informações, consulte Excluir um CodeCommit repositório no Guia do AWS CodeCommit usuário.
- Exclua o bucket do S3. Para obter mais informações, consulte Excluir um bucket no Guia do usuário do Amazon Simple Storage Service.
- Exclua a AWS CloudFormation pilha. Para obter mais informações, consulte Excluindo uma pilha no AWS CloudFormation console no Guia do AWS CloudFormation usuário.

Utilitários Batch na modernização do AWS mainframe

As aplicações de mainframe geralmente usam programas utilitários em lote para executar funções específicas, como classificar dados, transferir arquivos usando FTP, carregar dados em bancos de dados como DB2, descarregar dados de bancos de dados e assim por diante.

Ao migrar seus aplicativos para a modernização do AWS mainframe, você precisa de utilitários de substituição funcionalmente equivalentes que possam realizar as mesmas tarefas que os usados no mainframe. Alguns desses utilitários podem já estar disponíveis como parte dos mecanismos de tempo de execução da modernização do AWS mainframe, mas estamos fornecendo os seguintes utilitários substitutos:

- M2SFTP: permite a transferência segura de arquivos usando o protocolo SFTP.
- M2WAIT: espera por um período de tempo especificado antes de continuar com a próxima etapa em um trabalho em lotes.
- TXT2PDF: converte arquivos de texto em formato PDF.
- M2DFUTIL: que fornece funções de backup, restauração, exclusão e cópia em conjuntos de dados, semelhantes ao suporte fornecido pelo utilitário ADRDSSU de mainframe.
- M2RUNCMD: permite executar comandos, scripts e chamadas de sistema do Micro Focus diretamente da JCL.

Desenvolvemos esses utilitários em lote com base no feedback dos clientes e os projetamos para fornecer a mesma funcionalidade dos utilitários de mainframe. O objetivo é fazer com que sua transição do mainframe para a modernização do AWS mainframe seja a mais tranquila possível.

Tópicos

- Utilitário em lote M2WAIT
- Utilitário em lote TXT2PDF
- Utilitário em lote M2DFUTIL
- Utilitário em lote M2RUNCMD

Localização binário

Esses utilitários estão pré-instalados nos produtos Micro Focus Enterprise Developer (ED) e Micro Focus Enterprise Server (ES). Você pode encontrá-los no seguinte local para todas as variantes de ED e ES:

- Linux: `/opt/aws/m2/microfocus/utilities/64bit`
- Windows (32 bits): `C:\AWS\M2\MicroFocus\Utilities\32bit`
- Windows (64 bits): `C:\AWS\M2\MicroFocus\Utilities\64bit`

Utilitário em lote M2SFTP

O M2SFTP é um programa utilitário JCL projetado para realizar transferências seguras de arquivos entre sistemas usando o Secure File Transfer Protocol (SFTP). O programa usa o cliente Putty SFTP, `psftp`, para realizar as transferências reais de arquivos. O programa funciona de forma semelhante a um programa utilitário de FTP de mainframe e usa autenticação de usuário e senha.

> **Note**
>
> A autenticação de chave pública não é compatível.

Para converter seus JCLs FTP de mainframe para usar SFTP, mude `PGM=FTP` para `PGM=M2SFTP`.

Tópicos

- Plataformas compatíveis
- Instalar as dependências
- Configurar o M2SFTP para AWS a modernização gerenciada do mainframe
- Configurar o M2SFTP para o tempo de execução da modernização do AWS mainframe no Amazon EC2 (incluindo 2.0) AppStream

Plataformas compatíveis

Você pode usar o M2SFTP em qualquer uma das seguintes plataformas:

- AWS Modernização de mainframe gerenciada pela Micro Focus
- Micro Focus Runtime (no Amazon EC2)
- Todas as variantes dos produtos Micro Focus Enterprise Developer (ED) e Micro Focus Enterprise Server (ES).

Instalar as dependências

Para instalar o cliente Putty SFTP no Windows

- Faça o download do cliente PuTTY SFTP e instale-o.

Para instalar o cliente Putty SFTP no Linux:

- Execute o seguinte comando para instalar o cliente SFTP do Putty:

```
sudo yum -y install putty
```

Configurar o M2SFTP para AWS a modernização gerenciada do mainframe

Se seus aplicativos migrados estiverem sendo executados no AWS Mainframe Modernization Managed, você precisará configurar o M2SFTP da seguinte forma.

- Defina as variáveis de ambiente apropriadas do Micro Focus Enterprise Server para MFFTP. Aqui estão alguns exemplos:
  - `MFFTP_TEMP_DIR`
  - `MFFTP_SENDEOL`

- `MFFTP_TIME`
- `MFFTP_ABEND`

Você pode definir quantas ou quantas dessas variáveis quiser. Você pode configurá-los em seu JCL usando a `ENVAR DD` instrução. Para obter mais informações sobre essas variáveis, consulte Variáveis de controle do MFFTP na documentação da Micro Focus.

Para testar sua configuração, consulte Exemplo de JCLs.

Configurar o M2SFTP para o tempo de execução da modernização do AWS mainframe no Amazon EC2 (incluindo 2.0) AppStream

Se seus aplicativos migrados estiverem sendo executados em tempo de execução de modernização de AWS mainframe no Amazon EC2, configure o M2SFTP da seguinte forma.

1. Altere o caminho do programa Micro Focus JES para incluir a localização binária dos utilitários em lote. Se você precisar especificar vários caminhos, use dois pontos (`:`) para separar caminhos no Linux e ponto e vírgula (`;`) no Windows.
   - Linux: `/opt/aws/m2/microfocus/utilities/64bit`
   - Windows (32 bits): `C:\AWS\M2\MicroFocus\Utilities\32bit`
   - Windows (64 bits): `C:\AWS\M2\MicroFocus\Utilities\64bit`
2. Defina as variáveis de ambiente apropriadas do Micro Focus Enterprise Server para MFFTP. Aqui estão alguns exemplos:
   - `MFFTP_TEMP_DIR`
   - `MFFTP_SENDEOL`
   - `MFFTP_TIME`
   - MFFTP_ABEND

   Você pode definir quantas ou quantas dessas variáveis quiser. Você pode configurá-los em seu JCL usando a `ENVAR DD` instrução. Para obter mais informações sobre essas variáveis, consulte Variáveis de controle do MFFTP na documentação da Micro Focus.

Para testar sua configuração, consulte Exemplo de JCLs.

Exemplo de JCLs

Para testar a instalação, você pode usar um dos seguintes arquivos JCL de exemplo.

M2SFTP1.jcl

Este JCL mostra como chamar o M2SFTP para enviar um arquivo para um servidor SFTP remoto. Observe as variáveis de ambiente definidas na `ENVVAR DD` instrução.

```
//M2SFTP1 JOB 'M2SFTP1',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*--------------------------------------------------------------------**
//* Sample SFTP JCL step to send a file to SFTP server*
//*--------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=M2SFTP,
//            PARM='127.0.0.1 (EXIT=99 TIMEOUT 300'
//*
//SYSFTPD  DD  *
RECFM FB
LRECL 80
SBSENDEOL CRLF
MBSENDEOL CRLF
TRAILINGBLANKS FALSE
/*
//NETRC    DD  *
machine 127.0.0.1 login sftpuser password sftppass
/*
//SYSPRINT DD  SYSOUT=*
//OUTPUT   DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//INPUT    DD  *
type a
locsite notrailingblanks
cd files
put 'AWS.M2.TXT2PDF1.PDF' AWS.M2.TXT2PDF1.pdf
put 'AWS.M2.CARDDEMO.CARDDATA.PS' AWS.M2.CARDDEMO.CARDDATA.PS1.txt
quit
/*
//ENVVAR   DD *
```

```
MFFTP_VERBOSE_OUTPUT=ON
MFFTP_KEEP=N
/*
//*
//
```

M2SFTP2.jcl

Este JCL mostra como chamar o M2SFTP para receber um arquivo de um servidor SFTP remoto. Observe as variáveis de ambiente definidas na `ENVVAR DD` declaração.

```
//M2SFTP2 JOB 'M2SFTP2',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*-------------------------------------------------------------------**
//* Sample SFTP JCL step to receive a file from SFTP server*
//*-------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=M2SFTP
//*
//SYSPRINT DD  SYSOUT=*
//OUTPUT   DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//INPUT    DD  *
open 127.0.0.1
sftpuser
sftppass
cd files
locsite recfm=fb lrecl=150
get AWS.M2.CARDDEMO.CARDDATA.PS.txt +
'AWS.M2.CARDDEMO.CARDDATA.PS2' (replace
quit
/*
//ENVVAR   DD *
MFFTP_VERBOSE_OUTPUT=ON
MFFTP_KEEP=N
/*
//*
//
```

> **Note**
>
> É altamente recomendável armazenar as credenciais de FTP em um arquivo NETRC e restringir o acesso somente a usuários autorizados.

Referência de comando do cliente Putty SFTP (PSFTP)

O cliente PSFTP não suporta todos os comandos de FTP. A lista a seguir mostra todos os comandos que o PSFTP suporta.

Command	Descrição
!	Executar um comando local
tchau	Conclua sua sessão de SFTP
cd ~	Altere seu diretório de trabalho remoto
chmod	Alterar permissões e modos de arquivo
feche	Conclua sua sessão de SFTP, mas não saia do PSFTP
del	Excluir arquivos no servidor remoto
dir	Listar arquivos remotos
exit	Conclua sua sessão de SFTP
get	Baixe um arquivo do servidor para sua máquina local
ajuda	Dê ajuda
lcd	Alterar diretório de trabalho local
lpwd	Imprimir diretório de trabalho local
ls	Listar arquivos remotos

Command	Descrição
ímã	Baixe vários arquivos de uma só vez
mkdir	Crie diretórios no servidor remoto
entrada	Faça upload de vários arquivos de uma só vez
mv	Mover ou renomear arquivo (s) no servidor remoto
aberto	Conecte-se a um host
put	Upload um arquivo da sua máquina local para o servidor
PWD	Imprima seu diretório de trabalho remoto
sair	Conclua sua sessão de SFTP
arrepender	Continue baixando arquivos
arrancar	Mover ou renomear arquivo (s) no servidor remoto
reputação	Continuar carregando arquivos
/rm	Excluir arquivos no servidor remoto
rmdir	Remover diretórios no servidor remoto

Próximas etapas

Para carregar e baixar arquivos no Amazon Simple Storage Service usando SFTP, você pode usar o M2SFTP em conjunto com o AWS Transfer Family, conforme descrito nas postagens do blog a seguir.

- Usando diretórios lógicos AWS SFTP para criar um serviço simples de distribuição de dados
- Ativar a autenticação por senha para AWS Transfer for SFTP usar AWS Secrets Manager

Utilitário em lote M2WAIT

O M2WAIT é um programa utilitário de mainframe que permite introduzir um período de espera em seus scripts JCL especificando uma duração de tempo em segundos, minutos ou horas. Você pode chamar o M2WAIT diretamente do JCL passando o tempo que deseja esperar como parâmetro de entrada. Internamente, o programa M2WAIT chama o módulo fornecido pela Micro Focus `C$SLEEP` para aguardar um tempo especificado.

> **Note**
>
> Você pode usar aliases da Micro Focus para substituir o que você tem em seus scripts JCL. Para obter mais informações, consulte JES Alias na documentação da Micro Focus.

Tópicos

Plataformas compatíveis

Você pode usar o M2WAIT em qualquer uma das seguintes plataformas:

- AWS Modernização de mainframe gerenciada pela Micro Focus
- Micro Focus Runtime (no Amazon EC2)
- Todas as variantes dos produtos Micro Focus Enterprise Developer (ED) e Micro Focus Enterprise Server (ES).

Configurar o M2WAIT para a modernização gerenciada do AWS mainframe

Se seus aplicativos migrados estiverem sendo executados no AWS Mainframe Modernization Managed, você precisará configurar o M2WAIT da seguinte forma.

- Use o programa M2WAIT em seu JCL passando o parâmetro de entrada conforme mostrado em. Amostra de JCL

Configure o M2WAIT para o tempo de AWS execução da modernização do mainframe no Amazon EC2 (incluindo 2.0) AppStream

Se seus aplicativos migrados estiverem sendo executados em tempo de execução de modernização de AWS mainframe no Amazon EC2, configure o M2WAIT da seguinte forma.

1. Altere o caminho do programa Micro Focus JES para incluir a localização binária dos utilitários em lote. Se você precisar especificar vários caminhos, use dois pontos (`:`) para separar caminhos no Linux e ponto e vírgula (`;`) no Windows.
   - Linux: `/opt/aws/m2/microfocus/utilities/64bit`
   - Windows (32 bits): `C:\AWS\M2\MicroFocus\Utilities\32bit`
   - Windows (64 bits): `C:\AWS\M2\MicroFocus\Utilities\64bit`
2. Use o programa M2WAIT em seu JCL passando o parâmetro de entrada conforme mostrado em. Amostra de JCL

Amostra de JCL

Para testar a instalação, você pode usar o `M2WAIT1.jcl` programa.

Este exemplo de JCL mostra como chamar o M2WAIT e passá-lo por várias durações diferentes.

```
//M2WAIT1 JOB 'M2WAIT',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*-------------------------------------------------------------------**
//* Wait for 12 Seconds*
//*-------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=M2WAIT,PARM='S012'
//SYSOUT DD SYSOUT=*
//*
//*-------------------------------------------------------------------**
//* Wait for 0 Seconds (defaulted to 10 Seconds)*
//*-------------------------------------------------------------------**
//*
//STEP02 EXEC PGM=M2WAIT,PARM='S000'
//SYSOUT DD SYSOUT=*
```

```
//*
//*-------------------------------------------------------------------**
//* Wait for 1 Minute*
//*-------------------------------------------------------------------**
//*
//STEP03 EXEC PGM=M2WAIT,PARM='M001'
//SYSOUT DD SYSOUT=*
//*
//
```

Utilitário em lote TXT2PDF

O TXT2PDF é um programa utilitário de mainframe comumente usado para converter um arquivo de texto em um arquivo PDF. Esse utilitário usa o mesmo código-fonte do TXT2PDF (z/OS freeware). Nós o modificamos para ser executado no ambiente de execução de modernização de AWS mainframe da Micro Focus.

Tópicos

Plataformas compatíveis

Você pode usar o TXT2PDF em qualquer uma das seguintes plataformas:

- AWS Modernização de mainframe gerenciada pela Micro Focus
- Micro Focus Runtime (no Amazon EC2)
- Todas as variantes dos produtos Micro Focus Enterprise Developer (ED) e Micro Focus Enterprise Server (ES).

Configurar o TXT2PDF para a modernização gerenciada do mainframe AWS

Se seus aplicativos migrados estiverem sendo executados no AWS Mainframe Modernization Managed, configure o TXT2PDF da seguinte forma.

- Crie uma biblioteca REXX EXEC chamada `AWS.M2.REXX.EXEC`. Baixe esses módulos REXX e copie-os para a biblioteca.
  - `TXT2PDF.rex`: TXT2PDF z/OS freeware (modificado)
  - `TXT2PDFD.rex`: TXT2PDF z/OS freeware (não modificado)
  - `TXT2PDFX.rex`: TXT2PDF z/OS freeware (modificado)
  - `M2GETOS.rex`: para verificar o tipo de sistema operacional (Windows ou Linux)

Para testar sua configuração, consulte Amostra de JCL.

Configurar o TXT2PDF para o tempo de execução da modernização do AWS mainframe no Amazon EC2 (incluindo 2.0) AppStream

Se seus aplicativos migrados estiverem sendo executados em tempo de execução de modernização de AWS mainframe no Amazon EC2, configure o TXT2PDF da seguinte forma.

1. Defina a variável `MFREXX_CHARSET` de ambiente Micro Focus com o valor apropriado, como "`A`" para dados ASCII.

   > **Important**
   >
   > Inserir o valor errado pode causar problemas de conversão de dados (de EBCDIC para ASCII), tornando o PDF resultante ilegível ou inoperável. Recomendamos que `MFREXX_CHARSET` a configuração corresponda`MF_CHARSET`.

2. Altere o caminho do programa Micro Focus JES para incluir a localização binária dos utilitários em lote. Se você precisar especificar vários caminhos, use dois pontos (`:`) para separar caminhos no Linux e ponto e vírgula (`;`) no Windows.
   - Linux: `/opt/aws/m2/microfocus/utilities/64bit`
   - Windows (32 bits): `C:\AWS\M2\MicroFocus\Utilities\32bit`
   - Windows (64 bits): `C:\AWS\M2\MicroFocus\Utilities\64bit`

3. Crie uma biblioteca REXX EXEC chamada `AWS.M2.REXX.EXEC`. Baixe esses módulos REXX e copie-os para a biblioteca.

   - `TXT2PDF.rex`: TXT2PDF z/OS freeware (modificado)
   - `TXT2PDFD.rex`: TXT2PDF z/OS freeware (não modificado)
   - `TXT2PDFX.rex`: TXT2PDF z/OS freeware (modificado)
   - `M2GETOS.rex`: para verificar o tipo de sistema operacional (Windows ou Linux)

Para testar sua configuração, consulte Amostra de JCL.

Amostra de JCL

Para testar a instalação, você pode usar um dos seguintes arquivos JCL de exemplo.

TXT2PDF1.jcl

Esse arquivo JCL de amostra usa um nome DD para a conversão TXT2PDF.

```
//TXT2PDF1 JOB 'TXT2PDF1',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*--------------------------------------------------------------------**
//* PRE DELETE*
//*--------------------------------------------------------------------**
//*
//PREDEL  EXEC PGM=IEFBR14
//*
//DD01     DD DSN=AWS.M2.TXT2PDF1.PDF.VB,
//            DISP=(MOD,DELETE,DELETE)
//*
//DD02     DD DSN=AWS.M2.TXT2PDF1.PDF,
//            DISP=(MOD,DELETE,DELETE)
//*
//*--------------------------------------------------------------------**
//* CALL TXT2PDF TO CONVERT FROM TEXT TO PDF (VB)*
//*--------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=IKJEFT1B
//*
//SYSEXEC  DD DISP=SHR,DSN=AWS.M2.REXX.EXEC
```

```
//*
//INDD     DD *
1THIS IS THE FIRST LINE ON THE PAGE 1
0THIS IS THE THIRD LINE ON THE PAGE 1
-THIS IS THE   6TH LINE ON THE PAGE 1
THIS IS THE   7TH LINE ON THE PAGE 1
+_____________________________________ - OVERSTRIKE 7TH LINE
1THIS IS THE FIRST LINE ON THE PAGE 2
0THIS IS THE THIRD LINE ON THE PAGE 2
-THIS IS THE   6TH LINE ON THE PAGE 2
THIS IS THE   7TH LINE ON THE PAGE 2
+_____________________________________ - OVERSTRIKE 7TH LINE
/*
//*
//OUTDD    DD DSN=AWS.M2.TXT2PDF1.PDF.VB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(LRECL=256,DSORG=PS,RECFM=VB,BLKSIZE=0)
//*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD DDNAME=SYSIN
//*
//SYSIN    DD *
%TXT2PDF BROWSE Y IN DD:INDD +
OUT DD:OUTDD +
CC YES
/*
//*
//*--------------------------------------------------------------------**
//* CONVERT PDF (VB) TO PDF (LSEQ - BYTE STREAM)*
//*--------------------------------------------------------------------**
//*
//STEP02 EXEC PGM=VB2LSEQ
//*
//INFILE   DD DSN=AWS.M2.TXT2PDF1.PDF.VB,DISP=SHR
//*
//OUTFILE  DD DSN=AWS.M2.TXT2PDF1.PDF,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(LRECL=256,DSORG=PS,RECFM=LSEQ,BLKSIZE=0)
//*
//SYSOUT   DD SYSOUT=*
//*
//
```

TXT2PDF2.jcl

Esse exemplo de JCL usa um nome DSN para a conversão TXT2PDF.

```
//TXT2PDF2 JOB 'TXT2PDF2',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*--------------------------------------------------------------------**
//* PRE DELETE*
//*--------------------------------------------------------------------**
//*
//PREDEL  EXEC PGM=IEFBR14
//*
//DD01     DD DSN=AWS.M2.TXT2PDF2.PDF.VB,
//            DISP=(MOD,DELETE,DELETE)
//*
//DD02     DD DSN=AWS.M2.TXT2PDF2.PDF,
//            DISP=(MOD,DELETE,DELETE)
//*
//*--------------------------------------------------------------------**
//* CALL TXT2PDF TO CONVERT FROM TEXT TO PDF (VB)*
//*--------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=IKJEFT1B
//*
//SYSEXEC  DD DISP=SHR,DSN=AWS.M2.REXX.EXEC
//*
//INDD     DD *
1THIS IS THE FIRST LINE ON THE PAGE 1
0THIS IS THE THIRD LINE ON THE PAGE 1
-THIS IS THE   6TH LINE ON THE PAGE 1
THIS IS THE   7TH LINE ON THE PAGE 1
+_____________________________________ - OVERSTRIKE 7TH LINE
1THIS IS THE FIRST LINE ON THE PAGE 2
0THIS IS THE THIRD LINE ON THE PAGE 2
-THIS IS THE   6TH LINE ON THE PAGE 2
THIS IS THE   7TH LINE ON THE PAGE 2
+_____________________________________ - OVERSTRIKE 7TH LINE
/*
//*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD DDNAME=SYSIN
```

```
//*
//SYSIN    DD *
%TXT2PDF BROWSE Y IN DD:INDD +
OUT 'AWS.M2.TXT2PDF2.PDF.VB' +
CC YES
/*
//*
//*-------------------------------------------------------------------**
//* CONVERT PDF (VB) TO PDF (LSEQ - BYTE STREAM)*
//*-------------------------------------------------------------------**
//*
//STEP02 EXEC PGM=VB2LSEQ
//*
//INFILE   DD DSN=AWS.M2.TXT2PDF2.PDF.VB,DISP=SHR
//*
//OUTFILE  DD DSN=AWS.M2.TXT2PDF2.PDF,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(LRECL=256,DSORG=PS,RECFM=LSEQ,BLKSIZE=0)
//*
//SYSOUT   DD SYSOUT=*
//*
//
```

Modificações

Para que o programa TXT2PDF seja executado no ambiente de execução de modernização de AWS mainframe da Micro Focus, fizemos as seguintes alterações:

- Alterações no código-fonte para garantir a compatibilidade com o runtime do Micro Focus REXX
- Alterações para garantir que o programa possa ser executado nos sistemas operacionais Windows e Linux
- Modificações para suportar o runtime EBCDIC e ASCII

Referências

Referências e código-fonte do TXT2PDF:

- Conversor de texto para PDF
- Ferramentas gratuitas de TCP/IP e e-mail do z/OS
- Guia de referência do usuário do TXT2PDF

Utilitário em lote M2DFUTIL

O M2DFUTIL é um programa utilitário em JCL que fornece funções de backup, restauração, exclusão e cópia em conjuntos de dados, semelhantes ao suporte fornecido pelo utilitário ADRDSSU de mainframe. Esse programa retém muitos dos parâmetros SYSIN do ADRDSSU, o que simplifica o processo de migração para esse novo utilitário.

Tópicos

Plataformas compatíveis

Você pode usar o M2DFUTIL em qualquer uma das seguintes plataformas:

- Micro Focus ES no Windows (64 bits e 32 bits)
- Micro Focus ES no Linux (64 bits)

Requisitos da plataforma

O M2DFUTIL depende da chamada de um script para realizar um teste de expressão regular. No Windows, você deve instalar o Windows Services for Linux (WSL) para que esse script seja executado.

Suporte futuro planejado

Os recursos que atualmente não estão disponíveis no utilitário ADRDSSU de mainframe, mas que estão no escopo futuro, incluem:

- M2 gerenciado
- VSAM

- Suporte a COPY para a renomeação de nomes de arquivos
- Suporte a RENAME para RESTORE
- INCLUDE e EXCLUDE vários
- Cláusula BY para subseleção por DSORG, CREDT, EXPDT
- Cláusula MWAIT para tentar novamente falhas de enfileiramento
- Suporte ao armazenamento do S3 para DUMP/RESTORE

Locais dos ativos

O módulo de carregamento desse utilitário é chamado `M2DFUTIL.so` no Linux e `M2DFUTIL.dll` no Windows. Esse módulo de carregamento pode ser encontrado em um dos seguintes locais:

- Linux: `/opt/aws/m2/microfocus/utilities/64bit`
- Windows (32 bits): `C:\AWS\M2\MicroFocus\Utilities\32bit`
- Windows (64 bits): `C:\AWS\M2\MicroFocus\Utilities\64bit`

O script usado para testes de expressão regular é chamado `compare.sh`. O script pode ser encontrado em um dos seguintes locais:

- Linux: `/opt/aws/m2/microfocus/utilities/scripts`
- Windows (32 bits): `C:\AWS\M2\MicroFocus\Utilities\scripts`

Configure o tempo de execução do M2DFUTIL ou da modernização do AWS mainframe no Amazon EC2 (incluindo 2.0) AppStream

Configure sua região do Enterprise Server com o seguinte:

- Adicione as seguintes variáveis em [ES-Environment]:
  - `M2DFUTILS_BASE_LOC`: o local padrão para a saída DUMP
  - `M2DFUTILS_SCRIPTPATH`: o local do script `compare.sh` documentado em Locais dos ativos
  - `M2DFUTILS_VERBOSE`: [VERBOSO ou NORMAL]. Isso controla o nível de detalhe na saída `SYSPRINT`
- Verificar se o caminho do módulo de carregamento foi adicionado à configuração `JES \Configuration\JES Program Path`

- Verifique se os scripts no diretório de utilitários têm permissões de execução. É possível adicionar uma permissão de execução usando o comando `chmod + x <script name>`, no ambiente Linux

Sintaxe geral

DUMP

Fornece a possibilidade de copiar arquivos do local catalogado atual para um local de backup. No momento, esse local deve ser um sistema de arquivos.

Processar

DUMP executará o seguinte:

1. Crie o diretório do local de destino.
2. Catalogue o diretório do local de destino como membro do PDS.
3. Determine os arquivos a serem incluídos processando o parâmetro INCLUDE.
4. Desmarque os arquivos incluídos processando o parâmetro EXCLUDE.
5. Determine se os arquivos que estão sendo despejados devem ser EXCLUÍDOS.
6. Coloque em fila os arquivos a serem processados.
7. Copie os arquivos.
8. Exporte as informações do DCB catalogadas dos arquivos copiados para um arquivo secundário no local de destino para auxiliar nas futuras operações de RESTORE.

Sintaxe

```
DUMP
TARGET ( TARGET LOCATION  )    -
INCLUDE ( DSN. )
[ EXCLUDE ( DSN ) ]
[ CANCEL | IGNORE ]
[ DELETE ]
```

Parâmetros necessários

Veja abaixo os parâmetros necessários para DUMP:

- `SYSPRINT DD NAME`: para conter as informações adicionais de registro
- `TARGET`: local de destino Pode ser um ou outro.
  - Caminho completo do local de despejo
  - Nome do subdiretório criado no local definido na variável M2DFUTILS_BASE_LOC
- `INCLUDE`: um único DSNAME nomeado ou uma string de pesquisa de DSN do mainframe válida
- `EXCLUDE`: um único DSNAME nomeado ou uma string de pesquisa de DSN do mainframe válida

Parâmetros opcionais

- CANCEL: cancele se ocorrer algum erro. Os arquivos que foram processados serão retidos
- (Padrão) IGNORE: ignore qualquer erro e processo até o final
- DELETE: se nenhum erro ENQ ocorrer, o arquivo será excluído e não será catalogado

DELETE

Oferece a possibilidade de excluir e não catalogar arquivos em massa. Os arquivos não têm backup.

Processar

DELETE executará o seguinte:

1. Determine os arquivos a serem incluídos processando o parâmetro INCLUDE.
2. Desmarque os arquivos incluídos processando o parâmetro EXCLUDE.
3. Coloque em fila os arquivos a serem processados. Definindo a disposição como OLD, DELETE, KEEP.

Sintaxe

```
DELETE
INCLUDE ( DSN )
[ EXCLUDE ( DSN ) ]
[ CANCEL | IGNORE ]
[ DELETE ]
```

Parâmetros necessários

Veja abaixo os parâmetros necessários para DELETE:

- `SYSPRINT DD NAME`: para conter as informações adicionais de registro
- `INCLUDE`: um único DSNAME nomeado ou uma string de pesquisa de DSN do mainframe válida
- `EXCLUDE`: um único DSNAME nomeado ou uma string de pesquisa de DSN do mainframe válida

Parâmetros opcionais

- CANCEL: cancele se ocorrer algum erro. Os arquivos que foram processados serão retidos
- (Padrão) IGNORE: ignore qualquer erro e processo até o final

RESTORE

Fornece a possibilidade de restaurar arquivos que tiveram backup previamente DUMP. Os arquivos são restaurados no local catalogado original, a menos que RENAME seja usado para alterar o DSNAME restaurado.

Processar

RESTORE executará o seguinte:

1. Valide o diretório do local de origem.
2. Determine os arquivos a serem incluídos processando o arquivo de exportação do catálogo.
3. Desmarque os arquivos incluídos processando o parâmetro EXCLUDE.
4. Coloque em fila os arquivos a serem processados.
5. Arquivos de catálogo que não são catalogados com base em suas informações de exportação.
6. Se um arquivo já estiver catalogado e as informações do catálogo de exportação forem as mesmas, RESTORE substituirá o conjunto de dados catalogado se a opção REPLACE estiver definida.

Sintaxe

```
RESTORE
SOURCE ( TARGET LOCATION )
INCLUDE ( DSN )
[ EXCLUDE ( DSN ) ]
[ CANCEL | IGNORE ]
[ REPLACE]
```

Parâmetros necessários

Veja abaixo os parâmetros necessários para RESTORE:

- `SYSPRINT DD NAME`: para conter as informações adicionais de registro
- `SOURCE`: local de origem. Pode ser um ou outro.
  - Caminho completo do local de despejo
  - Nome do subdiretório criado no local definido na variável M2DFUTILS_BASE_LOC
- `INCLUDE`: um único DSNAME nomeado ou uma string de pesquisa de DSN do mainframe válida
- `EXCLUDE`: um único DSNAME nomeado ou uma string de pesquisa de DSN do mainframe válida

Parâmetros opcionais

- CANCEL: cancele se houver algum erro. Arquivos processados retidos
- (Padrão) IGNORE: ignore qualquer erro e processo até o final
- REPLACE: se o arquivo que está sendo restaurado já estiver catalogado e os registros do catálogo forem os mesmos, substitua o arquivo catalogado

Exemplo de JCLs

Trabalho de DUMP

Esse trabalho criará um subdiretório chamado `TESTDUMP`. Esse é o local de backup padrão especificado pela variável M2DFUTILS_BASE_LOC. Ele criará uma biblioteca PDS para esse backup chamada `M2DFUTILS.TESTDUMP`. Os dados do catálogo exportado são armazenados em um arquivo sequencial de linhas no diretório de backup chamado `CATDUMP.DAT`. Todos os arquivos selecionados serão copiados para esse diretório de backup.

```
//M2DFDMP JOB 'M2DFDMP',CLASS=A,MSGCLASS=X
//STEP001  EXEC PGM=M2DFUTIL
//SYSPRINT DD DSN=TESTDUMP.SYSPRINT,
//        DISP=(NEW,CATLG,DELETE),
//        DCB=(RECFM=LSEQ,LRECL=256)
//SYSIN    DD *
DUMP TARGET(TESTDUMP)               -
     INCLUDE(TEST.FB.FILE*.ABC)     -
 CANCEL
```

```
/*
//
```

Trabalho DELETE

Esse trabalho excluirá todos os arquivos do catálogo que correspondam ao parâmetro INCLUDE.

```
/M2DFDEL JOB 'M2DFDEL',CLASS=A,MSGCLASS=X
//STEP001  EXEC PGM=M2DFUTIL
//SYSPRINT DD DSN=TESTDEL.SYSPRINT,
//        DISP=(NEW,CATLG,DELETE),
//        DCB=(RECFM=LSEQ,LRECL=256)
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
  DELETE                              -
     INCLUDE(TEST.FB.FILE*.ABC)       -
 CANCEL
/*
 //
```

Trabalho RESTORE

Esse trabalho restaurará os arquivos que correspondem ao parâmetro INCLUDE do local de backup de TESTDUMP. Os arquivos catalogados serão substituídos se o arquivo catalogado for o mesmo da exportação de CATDUMP e a opção REPLACE for especificada.

```
//M2DFREST JOB 'M2DFREST',CLASS=A,MSGCLASS=X
//STEP001  EXEC PGM=M2DFUTIL
////SYSPRINT DD DSN=TESTREST.SYSPRINT,
//        DISP=(NEW,CATLG,DELETE),
//        DCB=(RECFM=LSEQ,LRECL=256)
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
RESTORE SOURCE(TESTDUMP)              -
     INCLUDE(TEST.FB.FILE*.ABC)       -
 IGNORE
 REPLACE
/*
//
```

Utilitário em lote M2RUNCMD

Você pode usar o M2RUNCMD, um programa utilitário em lote, para executar comandos, scripts e chamadas de sistema do Micro Focus diretamente da JCL em vez de executá-los em um terminal ou prompt de comando. A saída dos comandos é registrada no log de spool do trabalho em lote.

Tópicos

Plataformas compatíveis

É possível usar o M2RUNCMD nas seguintes plataformas:

- Micro Focus Runtime (no Amazon EC2)
- Todas as variantes dos produtos Micro Focus Enterprise Developer (ED) e Micro Focus Enterprise Server (ES).

Configure o M2RUNCMD para o tempo de execução de modernização de AWS mainframe no Amazon EC2 (incluindo 2.0) AppStream

Se seus aplicativos migrados estiverem sendo executados em tempo de execução de modernização de AWS mainframe no Amazon EC2, configure o M2RUNCMD da seguinte forma.

- Altere o caminho do programa Micro Focus JES para incluir a localização binária dos utilitários em lote. Se precisar especificar vários caminhos, use dois-pontos (:) para separar caminhos no Linux e ponto e vírgula (;) no Windows.
  - Linux: `/opt/aws/m2/microfocus/utilities/64bit`
  - Windows (32 bits): `C:\AWS\M2\MicroFocus\Utilities\32bit`
  - Windows (64 bits): `C:\AWS\M2\MicroFocus\Utilities\64bit`

Exemplo de JCLs

Para testar a instalação, use um dos seguintes arquivos JCL de exemplo:

RUNSCRL1.jcl

Esse exemplo de JCL cria um script e o executa. A primeira etapa cria um script chamado `/tmp/TEST_SCRIPT.sh` e com conteúdo de dados no stream de `SYSUT1`. A segunda etapa define a permissão de execução e executa o script criado na primeira etapa. Também é possível optar por realizar somente a segunda etapa para executar comandos já existentes do Micro Focus e do sistema.

```
//RUNSCRL1 JOB 'RUN SCRIPT',CLASS=A,MSGCLASS=X,TIME=1440
//*
//*
//*--------------------------------------------------------------------*
//*  CREATE SCRIPT (LINUX)
//*--------------------------------------------------------------------*
//*
//STEP0010 EXEC PGM=IEBGENER
//*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//*
//SYSUT1   DD *
#!/bin/bash

set -x

## ECHO PATH ENVIRONMNET VARIABLE
echo $PATH

## CLOSE/DISABLE VSAM FILE
casfile -r$ES_SERVER -oc  -ed -dACCTFIL

## OPEN/ENABLE VSAM FILE
casfile -r$ES_SERVER -ooi -ee -dACCTFIL

exit $?
/*
//SYSUT2   DD DSN=&&TEMP,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=LSEQ,LRECL=300,DSORG=PS,BLKSIZE=0)
//*MFE: %PCDSN='/tmp/TEST_SCRIPT.sh'
//*
//*--------------------------------------------------------------------*
//*   RUN SCRIPT (LINUX)                                               *
```

```
//*-------------------------------------------------------------------*
//*
//STEP0020 EXEC PGM=RUNCMD
//*
//SYSOUT  DD  SYSOUT=*
//*
//SYSIN   DD *
*RUN SCRIPT
 sh /tmp/TEST_SCRIPT.sh
/*
//
```

SYSOUT

A saída do comando ou script executado é gravada no log de SYS0UT. Para cada comando executado, ele exibe o comando, a saída e o código de retorno.

```
************ CMD Start ************

CMD_STR: sh /tmp/TEST_SCRIPT.sh

CMD_OUT:


+ echo /opt/microfocus/EnterpriseServer/bin:/sbin:/bin:/usr/sbin:/usr/bin
/opt/microfocus/EnterpriseServer/bin:/sbin:/bin:/usr/sbin:/usr/bin
+ casfile -rMYDEV -oc -ed -dACCTFIL

-Return Code:   0

Highest return code:    0

+ casfile -rMYDEV -ooi -ee -dACCTFIL

-Return Code:   8

Highest return code:    8

+ exit 8


CMD_RC=8
```

```
************  CMD End  ************
```

RUNCMDL1.jcl

Este exemplo de JCL usa RUNCMD para executar vários comandos.

```
//RUNCMDL1 JOB 'RUN CMD',CLASS=A,MSGCLASS=X,TIME=1440
//*
//*
//*--------------------------------------------------------------------*
//*   RUN SYSTEM COMMANDS                                              *
//*--------------------------------------------------------------------*
//*
//STEP0001 EXEC PGM=RUNCMD
//*
//SYSOUT  DD  SYSOUT=*
//*
//SYSIN   DD *
*LIST DIRECTORY
 ls
*ECHO PATH ENVIRONMNET VARIABLE
 echo $PATH
/*
//
```

AWS Modernização do mainframe e replicação de dados com precisão

AWS A modernização do mainframe oferece uma variedade de Amazon Machine Images (AMIs). Essas AMIs facilitam o provisionamento rápido de instâncias do Amazon EC2, criando um ambiente personalizado para replicação de dados de sistemas de mainframe para o uso do Precisely. AWS Este guia fornece as etapas necessárias para acessar e usar essas AMIs.

Pré-requisitos

- Certifique-se de ter acesso de administrador a uma AWS conta na qual você possa criar instâncias do Amazon EC2.
- Verifique se o serviço de modernização do AWS mainframe está disponível na região em que você planeja criar as instâncias do Amazon EC2. Veja a lista de serviços da AWS disponíveis por região.
- Identifique a Amazon Virtual Private Cloud (Amazon VPC) na qual as instâncias do Amazon EC2 serão criadas.
- Ao criar instâncias do Amazon EC2 em uma Amazon VPC, certifique-se de que a tabela de rotas associada tenha um gateway da Internet ou um gateway NAT.

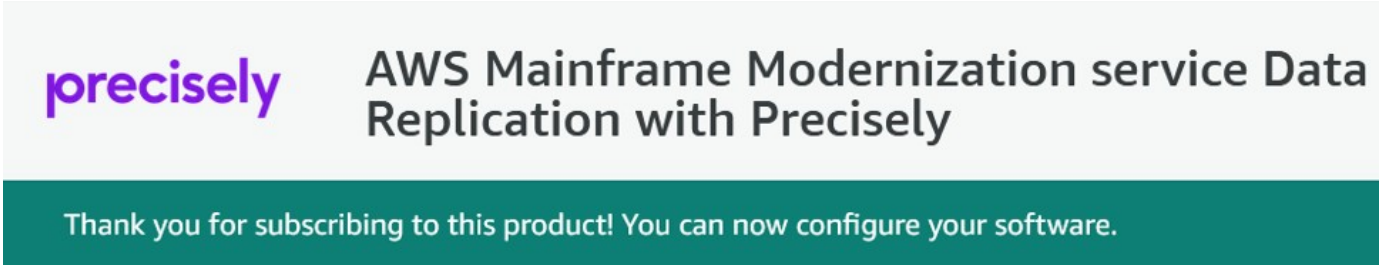
> **Note**
>
> A replicação de dados com êxito exige que a instância do EC2 da AWS tenha acesso de comunicação ao AWS Marketplace. Se houver um problema de conectividade com o AWS Marketplace, haverá falha no processo de replicação.

Inscrever-se para receber a imagem de máquina da Amazon

Após a inscrição em um produto do AWS Marketplace, você pode executar uma instância usando a AMI do produto.

1. Faça login no AWS Management Console e abra o AWS Marketplace console em https://console.aws.amazon.com/marketplace.

2. Escolha Manage subscriptions (Gerenciar assinaturas).
3. Copie e cole o seguinte link na barra de endereço do navegador: https://aws.amazon.com/marketplace/pp/prodview-en3xrbgzbs3dk
4. Escolha Continue to Subscribe (Continuar para assinar).
5. Se os Termos e Condições forem aceitáveis, escolha Aceitar Termos. A assinatura pode levar alguns minutos para ser processada.
6. Aguarde até que a mensagem de agradecimento seja exibida, conforme mostrado abaixo. Essa mensagem confirma que você se inscreveu com êxito no produto.



7. No painel de navegação esquerdo, escolha Gerenciar assinaturas. Essa visualização mostra todas as assinaturas nas quais você se inscreveu.

Inicie a replicação de dados AWS da modernização do mainframe com o Precisely

1. Abra o AWS Marketplace console em https://console.aws.amazon.com/marketplace.
2. No painel de navegação esquerdo, escolha Gerenciar assinaturas.
3. Encontre a AMI que você deseja iniciar e escolha Executar nova instância.
4. Em Região, selecione a região listada como permitida.
5. Escolha Continuar para executar por meio do EC2. Essa ação direcionará você para o console do Amazon EC2.
6. Insira um nome para o servidor.
7. Selecione um tipo de instância que corresponda aos requisitos de desempenho e custo do seu projeto. O ponto de partida sugerido para o tamanho da instância é `c5.2xLarge`.
8. Escolha um par de chaves existente ou crie um e salve-o. Para obter informações sobre pares de chaves, consulte os pares de chaves do Amazon EC2 e as instâncias Linux no Guia do usuário do Amazon EC2.
9. Edite as configurações de rede e escolha a VPC na lista de permissões e a sub-rede apropriada.

10. Escolha um grupo de segurança existente ou crie um. Além de permitir o acesso SSH (por padrão na porta 22), para a replicação de dados com uma instância EC2 do servidor da Precisely, é comum permitir o tráfego TCP para sua porta padrão 2626.
11. Configure o armazenamento para a instância do Amazon EC2.
12. Revise o resumo e envie a Executar instância. Para que a execução tenha êxito, o tipo de instância deve ser válido. Se houver falha na execução, escolha Editar configuração da instância e escolha um tipo de instância diferente.
13. Depois de ver a mensagem de êxito, escolha Conectar-se à Instância.
14. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
15. No painel de navegação esquerdo, no menu Instâncias, escolha Instâncias.
16. No painel principal, verifique o status da sua instância.

Criar uma política do IAM

Para operar com sucesso as instâncias EC2 de modernização de AWS mainframe implantadas por meio de nossa AWS Marketplace listagem, você deve configurar uma função e uma política do IAM. Essa configuração específica do IAM não é opcional; ela autoriza suas instâncias do Amazon EC2 a interagir com o serviço. AWS Marketplace A função e a política do IAM permitem que a modernização do AWS mainframe registre com precisão os dados de uso, o que é essencial para um faturamento preciso. A não implementação dessa configuração pode levar a falha nas tentativas de replicação de dados e a interrupções operacionais.

1. Abra o console IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Policies (Políticas).
3. Se essa for a primeira vez que você escolhe Políticas, a página Bem-vindo às políticas gerenciadas será exibida. Escolha Começar.
4. Na parte superior da página, escolha Criar política.
5. Na seção Editor de políticas, escolha a opção JSON.
6. Insira a seguinte política JSON:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": ["aws-marketplace:MeterUsage"],
```

```
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Criar um perfil do IAM

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Roles e Create role.
3. Na seção Tipo de entidade confiável, escolha Serviço da AWS .
4. Na seção Caso de uso, em Serviço ou caso de uso, escolha Amazon EC2.
5. Selecione Next (Próximo).
6. Na lista de políticas, selecione Gerenciada pelo cliente no menu suspenso Filtrar por tipo e insira o nome da política que você criou. Marque a caixa de seleção ao lado do nome da política.
7. Selecione Next (Próximo).
8. Insira um nome e, opcionalmente, uma descrição para o perfil.
9. Revise a política de confiança e as permissões e escolha Criar perfil.

Anexar o perfil do IAM à instância do Amazon EC2

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. No painel de navegação, escolha Instances (Instâncias).
3. Selecione sua instância Amazon EC2.
4. No menu Ações, escolha Segurança e, depois, escolha Modificar o perfil do IAM.
5. Selecione o perfil do IAM a ser anexado à instância e escolha Atualizar perfil do IAM.

Integração do Charon

Introdução ao Charon-SSP

Em 1987, a Sun Microsystems lançou o processador SPARC V7, um processador RISC de 32 bits. O SPARC V8 foi lançado em 1990, uma revisão do SPARC V7 original, com a inclusão mais notável de instruções de divisão e multiplicação de hardware. Os processadores SPARC V8 formaram a base para vários servidores e estações de trabalho, como o SPARCstation 5, 10 e 20. Em 1993, depois do SPARC V8, foi lançado o processador SPARC V9 de 64 bits. Isso também se tornou a base para vários servidores e estações de trabalho, como o Enterprise 250 e 450.

Devido à obsolescência do hardware e à falta de peças de reposição ou de peças recondicionadas, o software e os sistemas desenvolvidos para essas estações de trabalho e servidores mais antigos baseados no SPARC ficaram mais difíceis de manter. Para preencher a necessidade contínua de determinados sistemas end-of-life baseados em SPARC, a Stromasys S.A. desenvolveu a linha Charon-SSP de produtos emuladores SPARC. Os produtos a seguir são substitutos de máquinas virtuais baseados em software para os sistemas SPARC de hardware nativo especificados. A seguir encontra-se uma visão geral das famílias de hardware emulado.

O Charon-SSP/4M emula o seguinte hardware SPARC:

- Família Sun-4m (representada pelo Sun SPARCstation 20): originalmente, uma variante do multiprocessador Sun-4, baseada no barramento do módulo do processador MBus apresentado na série SPARCServer 600MP. Posteriormente, a arquitetura Sun-4m também incluiu sistemas uniprocessadores que não sejam MBus, como o SPAR Cstation 5, utilizando processadores da arquitetura SPARC V8. Compatível a partir do SunOS 4.1.2 e do Solaris 2.1 até o Solaris 9. A compatibilidade com o SPARCServer 600MP foi descontinuada após o Solaris 2.5.1.

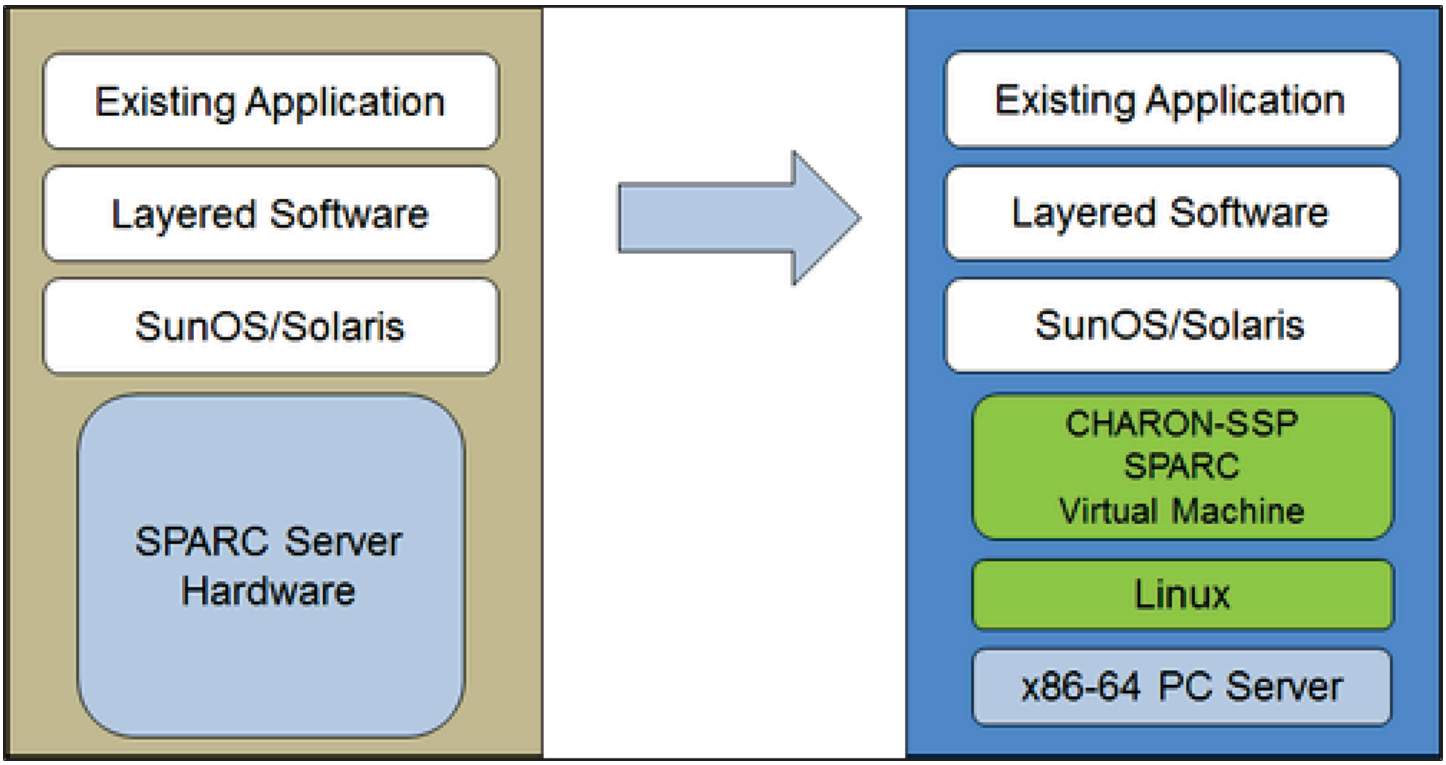
O Charon-SSP/4U(+) emula o seguinte hardware SPARC:

- Família Sun-4u (representada pelo Sun Enterprise 450): (U para UltraSPARC): essa variante apresentou a arquitetura do processador SPARC V9 de 64 bits e a interconexão do processador UPA usada pela primeira vez na série Sun Ultra. Compatível com versões de 32 bits do Solaris a partir da versão 2.5.1. A primeira versão do Solaris de 64 bits para o Sun-4u foi o Solaris 7. A compatibilidade com o UltraSPARC I foi descontinuada após o Solaris 9. O Solaris 10 é compatível com implementações do Sun-4u do UltraSP ARC II ao UltraSPARC IV.

O Charon-SSP/4V(+) emula o seguinte hardware SPARC:

- Família Sun-4v (representada pelos SPARC T2 e T4): essa variação adicionou a virtualização do processador hipervisor ao Sun-4u, apresentada no processador multicore Ultra SPARC T1. O hardware selecionado era compatível com a versão 10 do Solaris a partir da versão 3/05 HW2 (a maioria dos modelos, incluindo o hardware emulado pelo Charon-SSP, exige versões mais recentes do Solaris 10). Várias versões do Solaris 11 também são compatíveis.

A imagem a seguir mostra o conceito básico da migração de hardware físico para um emulador.



As máquinas virtuais Charon-SSP permitem que os usuários de computadores baseados em Sun e Oracle SPARC substituam seu hardware nativo de uma forma que exija pouca ou nenhuma alteração na configuração original do sistema. Isso significa que você pode continuar executando suas aplicações e dados sem a necessidade de alternar ou migrar para outra plataforma. O software Charon-SSP funciona em sistemas Intel de 64 bits básicos, garantindo a proteção contínua do seu investimento.

O Charon-SSP/4U+ é compatível com as mesmas plataformas SPARC virtuais que o Charon-SSP/4U, e o Charon-SSP/4V+ é o mesmo que o Charon-SSP/4V. No entanto, as versões 4U+ e 4V + aproveitam as vantagens da tecnologia de virtualização assistida por hardware VTx/EPT da Intel e AMD-v/NPT da AMD em CPUs modernas para oferecer melhor desempenho da CPU virtual. O

Charon-SSP/4U+ e o Charon-SSP/4V+ exigem CPUs compatíveis com VT-x/EPT ou AMD-v/NPT e devem ser instalados em um sistema host dedicado. A execução dessas variantes de produto em uma VM (por exemplo, no VMware) não é compatível.

> **Note**
>
> Se você planeja executar o Charon-SSP/4U+ ou o 4V+ em um ambiente de nuvem, entre em contato com a Stromasys ou com um VAR da Stromasys para conversar sobre seus requisitos.

Sistemas operacionais convidados compatíveis

As máquinas virtuais Charon-SSP/4M são compatíveis com as seguintes versões do sistema operacional convidado:

- SunOS 4.1.3 a 4.1.4
- Solaris 2.3 a Solaris 9

As máquinas virtuais Charon-SSP/4U(+) são compatíveis com as seguintes versões do sistema operacional convidado:

- Solaris 2.5.1 a Solaris 10

As máquinas virtuais Charon-SSP/4V(+) são compatíveis com as seguintes versões do sistema operacional convidado:

- Solaris 10 (começando com a atualização 4, 08/07) e Solaris 11.1 a Solaris 11.4

Para Charon-SSP/4V(+), observe o seguinte:

- Para o SPARC T4 emulado, as versões compatíveis do Solaris 10 são: Oracle Solaris 10 1/13, Oracle Solaris 10 8/11 e Solaris 10 9/10, ou Solaris 10 10/09 com o conjunto de patches Oracle Solaris 10 8/11.
- O modelo SPARC T4 emulado é um pré-requisito para executar o Solaris 11.4 no emulador.
- Não há compatibilidade com zonas de kernel do Solaris.

Pré-requisitos da instância de nuvem do Charon-SSP

Ao selecionar um tipo ou formato de instância, selecione o hardware virtual que será usado para a instância do host do Charon-SSP na nuvem. Portanto, a seleção de um tipo ou formato de instância determina as características do hardware do host virtual do Charon-SSP (por exemplo, quantos núcleos de CPU e quanta memória seu sistema host virtual do Charon terá).

> **Note**
>
> Se você usar uma imagem do marketplace do Charon-SSP para executar a instância, todos os requisitos do sistema operacional do host do Linux serão atendidos.

Os requisitos mínimos de hardware estão descritos abaixo.

Pontos importantes em relação às diretrizes de dimensionamento:

- As diretrizes de dimensionamento abaixo, especialmente em relação ao número de núcleos de CPU e memória do host, mostram os requisitos mínimos. Cada situação de implantação deve ser revisada e o dimensionamento real do host deve ser adaptado conforme necessário. Por exemplo, o número de núcleos de CPU disponíveis para E/S deve ser aumentado se as aplicações convidadas produzirem uma alta carga de E/S. Além disso, um sistema com muitas CPUs emuladas normalmente pode criar uma carga de E/S maior e, portanto, talvez seja necessário aumentar o número de núcleos de CPU disponíveis para E/S. Em um ambiente hyper-threading, para obter o melhor desempenho, o número de núcleos de CPU (ou seja, CPUs real/físicas) deve ser suficiente para atender aos requisitos de CPU dos emuladores ativos, evitando assim que threads de alta workload compartilhem um núcleo físico de CPU.
- A alocação de núcleos de CPU para CPUs emuladas e de núcleos de CPU para processamento de E/S é determinada pela configuração. Consulte Configuração da CPU no Guia geral do usuário do Charon-SSP para obter mais informações sobre isso e a alocação padrão de núcleos de CPU para processamento de E/S.

> **Informações gerais importantes**
>
> - Para facilitar a transferência rápida dos dados do emulador de uma instância de nuvem para outra, é altamente recomendável armazenar todos os dados relevantes do emulador

em um volume de disco separado que possa ser facilmente desanexado da instância antiga e anexado a uma nova instância.

- Certifique-se de dimensionar a instância corretamente desde o início (verifique os requisitos mínimos abaixo). A licença do Charon-SSP para o Charon-SSP AL é criada quando a instância é executada pela primeira vez. Alterar posteriormente para outro tamanho/tipo de instância e, assim, alterar o número de núcleos de CPU invalidará a licença e, assim, impedirá que as instâncias do Charon sejam executadas (será necessária uma nova instância). Se estiver planejando usar a instância do Charon-SSP AL no modo AutoVE, certifique-se de incluir as informações do servidor AutoVE antes da primeira execução, caso contrário, os servidores de licenças públicas serão usados. A licença do Charon-SSP VE é criada com base na impressão digital obtida no servidor de licenças. Se o servidor de licenças for executado diretamente no host do emulador e o host do emulador exigir posteriormente, por exemplo, uma alteração no número de núcleos de CPU, a licença será invalidada (será necessária uma nova licença e, possivelmente, uma nova instância).

Pré-requisitos da instância

Requisitos gerais de CPU: o Charon-SSP é compatível com processadores modernos de arquitetura x86-64 baseados em instâncias do Amazon EC2.

Requisitos mínimos para Charon-SSP:

- Número mínimo de núcleos de CPU do sistema host:
  - Pelo menos um núcleo de CPU para o sistema operacional host, além de:
  - Para cada sistema SPARC emulado:
    - Um núcleo de CPU para cada CPU emulada da instância, além de:
    - Pelo menos um núcleo de CPU adicional para processamento de E/S (pelo menos dois, se a otimização JIT do servidor for usada). Consulte a seção Configuração da CPU mencionada acima para ver as opções de configuração. Por padrão, o Charon atribuirá 1/3 (mínimo 1; arredondado para baixo) do número de CPUs visíveis para o host do Charon ao processamento de E/S.
- Requisitos mínimos de memória:
  - 4 GB ou mais de RAM para o sistema operacional host Linux. Os requisitos reais podem ser maiores e dependerão dos requisitos dos serviços não emuladores executados no host Linux.

A recomendação anterior de pelo menos 2 GB de RAM para o host Linux ainda será válida para muitos sistemas, mas os crescentes requisitos do sistema operacional e das aplicações do Linux levaram à recomendação atualizada para novas instalações. Além de:

  - Para cada sistema SPARC emulado:
    - A memória configurada da instância emulada, além de:
    - 2 GB de RAM (6 GB de RAM se o servidor JIT for usado) para permitir a otimização de DIT, requisitos do emulador, buffers de tempo de execução, SMP e emulação gráfica.
- Se o hyper-threading estiver habilitado em CPUs x86-64 modernas, dois threads poderão ser executados em um núcleo físico de CPU, fornecendo duas CPUs lógicas para o sistema operacional host. Se possível, desative o hyper-threading no host Charon-SSP. No entanto, isso geralmente não é possível em ambientes VMware e de nuvem, ou não está claro se o hyper-threading é usado ou não. A opção de hyper-threading do Charon-SSP permite que o Charon-SSP se adapte a esses ambientes. Consulte a seção Configuração da CPU no Guia geral do usuário do Charon-SSP mencionado acima para obter informações detalhadas da configuração. Observação: para obter o melhor desempenho, os threads Charon-SSP não devem compartilhar um núcleo físico de CPU. Núcleos físicos suficientes devem estar disponíveis no sistema host para satisfazer os requisitos do(s) emulador(es) configurado(s).
- Uma ou mais interfaces de rede, dependendo dos requisitos do cliente.
- O Charon-SSP/4U+ e o Charon-SSP/4V+ devem ser executados no hardware físico compatível com Intel VT-x/EPT ou AMD-v/NPT (instâncias baremetal) e, portanto, não podem ser executados em todos os ambientes de nuvem. Verifique a documentação do seu provedor de nuvem para saber sobre a disponibilidade desse hardware. Além disso, observe os seguintes pontos:
  - O Charon-SSP/4U+ e o Charon-SSP/4V+ só estão disponíveis ao usar um kernel do Linux compatível com a Stromasys.
  - Se você precisar desse tipo de hardware SPARC emulado, entre em contato com a Stromasys ou com seu VAR da Stromasys para discutir seus requisitos em detalhes.

Criação e configuração de uma instância de AWS nuvem para Charon (nova GUI)

Esta seção reflete o AWS Management Console na primavera de 2022. Se você ainda usa o console antigo, consulte o Apêndice do guia de introdução do AWS Charon-SSP.

Pré-requisitos gerais

Essa descrição mostra a configuração básica de uma instância do Linux na AWS. Ela não lista os pré-requisitos específicos. No entanto, dependendo do seu caso de uso, considere os seguintes pré-requisitos:

- Conta e AWS Marketplace assinaturas da Amazon
  - Para configurar uma instância Linux no AWS, você precisa de uma AWS conta com acesso de administrador.
  - Identifique a AWS região na qual você planeja executar sua instância. Certifique-se de que os serviços da AWS que você planeja usar estejam disponíveis nessa região. Consulte Serviços da AWS por região.
  - Identifique a VPC e a sub-rede na qual você planeja executar a instância.
  - Se a instância exigir acesso à Internet, certifique-se de que a tabela de rotas associada à sua VPC tenha um gateway da Internet. Se a instância exigir acesso da VPN à sua rede local, verifique se há um gateway de VPN disponível. A configuração exata da sua VPC e de suas sub-redes dependerá do design da rede e dos requisitos da aplicação.
  - Para assinar um AWS Marketplace serviço específico, escolha Assinaturas do AWS Marketplace no AWS Management Console e, em seguida, escolha Gerenciar assinaturas.
  - Procure o serviço que você planeja usar e inscreva-se nele. Depois de inscrever-se com êxito, você encontrará a assinatura na seção Gerenciar assinaturas. A partir daí, você poderá executar diretamente uma nova instância.
- Os pré-requisitos de hardware e de software da instância serão diferentes dependendo do uso planejado da instância:
  - Opção 1: a instância deve ser usada como um sistema host do emulador Charon:
    - Consulte as seções de pré-requisitos de hardware e de software do Guia do usuário e/ou do Guia de conceitos básicos do seu produto Charon para determinar os pré-requisitos exatos de hardware e de software que devem ser atendidos pela instância do Linux. A imagem que você usa para executar a instância e o tipo de instância escolhido determinam o software e o hardware da sua instância de nuvem.
    - É necessária uma licença de produto Charon para executar sistemas herdados emulados. Consulte as informações de licenciamento na documentação do seu produto Charon ou entre em contato com seu representante da Stromasys ou com o VAR da Stromasys para obter informações adicionais.
  - Opção 2: a instância deve ser usada como um servidor de licenças VE dedicado:

    - Consulte o Guia do Servidor de Licenças VE para obter os pré-requisitos detalhados.
- Alguns sistemas operacionais antigos que podem ser executados nos sistemas emulados fornecidos pelos produtos emuladores Charon exigem uma licença do fornecedor original do sistema operacional. O usuário é responsável por quaisquer obrigações de licenciamento relacionadas ao sistema operacional antigo e deve fornecer as licenças apropriadas.

Usando o AWS Management Console para iniciar uma nova instância

Como criar uma instância

1. Faça login no AWS Management Console e abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Escolha Iniciar instância.
3. Insira um nome para a instância.
4. Selecione uma AMI. Uma AMI é uma imagem pré-empacotada usada para executar instâncias de nuvem. Ela inclui o sistema operacional e o software aplicável à aplicação. A escolha da AMI depende de como você planeja usar a instância:

    - Se a instância for usada como um sistema host do emulador Charon, várias opções de AMI serão possíveis:
        - Instalar o sistema host Charon a partir de uma imagem pré-empacotada do marketplace do Charon: elas contêm o sistema operacional subjacente e o software Charon pré-instalado.
            - Verifique com seu representante da Stromasys quais opções estão atualmente disponíveis no marketplace de seus provedores de nuvem.
            - Dependendo do provedor de nuvem e dos planos de lançamento do produto da Stromasys, pode haver duas variantes:
                - Licenciamento automático (AL) para uso com um servidor de licenças público operado pela Stromasys ou com um servidor de licenças AutoVE privado operado pelo cliente
                - Ambiente virtual (VE) para uso com um servidor de licenças VE privado operado pelo cliente
        - Instalar o sistema host Charon usando uma instalação convencional do emulador Charon com os pacotes RPM de instalação do emulador Charon para Linux:

    - Escolha uma AMI do Linux de uma distribuição compatível com o produto e a versão selecionados do Charon. Consulte o guia do usuário do seu produto no site de documentação da Stromasys.
- Se a instância for usada como um servidor de licenças VE dedicado, consulte o Guia do servidor de licenças VE na documentação de licenciamento para ver os requisitos da instância do Linux.

Depois de decidir qual AMI é necessária, selecione uma AMI de produto Linux ou Charon correspondente. Caso não veja a AMI de que precisa, escolha Pesquisar mais AMIs. Escolha a AMI do Linux que corresponda à forma como você planeja usar a instância. Uma das seguintes opções é possível:

- Uma imagem pré-embalada do marketplace Charon VE. O nome da AMI incluirá a string “ve”.
- Uma imagem pré-embalada do marketplace Charon AL para licenciamento automático ou AutoVE.
- Uma versão do Linux compatível com a instalação de um produto RPM.
- Uma versão do Linux compatível com o servidor de licenças VE.

5. Selecione um tipo de instância. O Amazon EC2 oferece tipos de instância com combinações variadas de CPU, memória, armazenamento e capacidade de rede. Selecione um tipo de instância que corresponda aos requisitos do produto Charon que você deseja usar. Algumas imagens do marketplace têm uma seleção restrita de tipos de instância.
6. Selecione um par de chaves existente ou crie um e salve-o. Se você selecionar um par de chaves existente, verifique se você tem a chave privada correspondente. Caso contrário, você não poderá se conectar à instância.

> **Note**
>
> Se o seu sistema de gerenciamento for compatível com RHEL 9.x, Rocky Linux 9.x e Oracle Linux 9.x, use o tipo de chave SSH ECDSA ou ED25519. Esses tipos permitem que você se conecte a esses sistemas Linux do host Charon usando um túnel SSH sem precisar alterar as configurações padrão da política de criptografia no host Charon para configurações menos seguras. Por exemplo, isso é importante para o Charon-SSP Manager. Consulte Uso de políticas criptográficas em todo o sistema na documentação da Red Hat.

7. Na seção Configurações de rede, escolha Editar. Escolha as configurações que correspondam ao seu ambiente.

   - Especifique uma VPC.
   - Especifique uma sub-rede existente ou crie uma.
   - Habilite ou desabilite a atribuição automática de um endereço IP público à interface principal. A atribuição automática só é possível se a instância tiver uma única interface de rede.
   - Atribua um grupo de segurança personalizado novo ou existente. O grupo de segurança deve permitir que pelo menos o SSH acesse a instância. Todas as portas exigidas pelas aplicações que você planeja executar na instância também devem ser permitidas. É possível modificar o grupo de segurança a qualquer momento depois que a instância for criada.

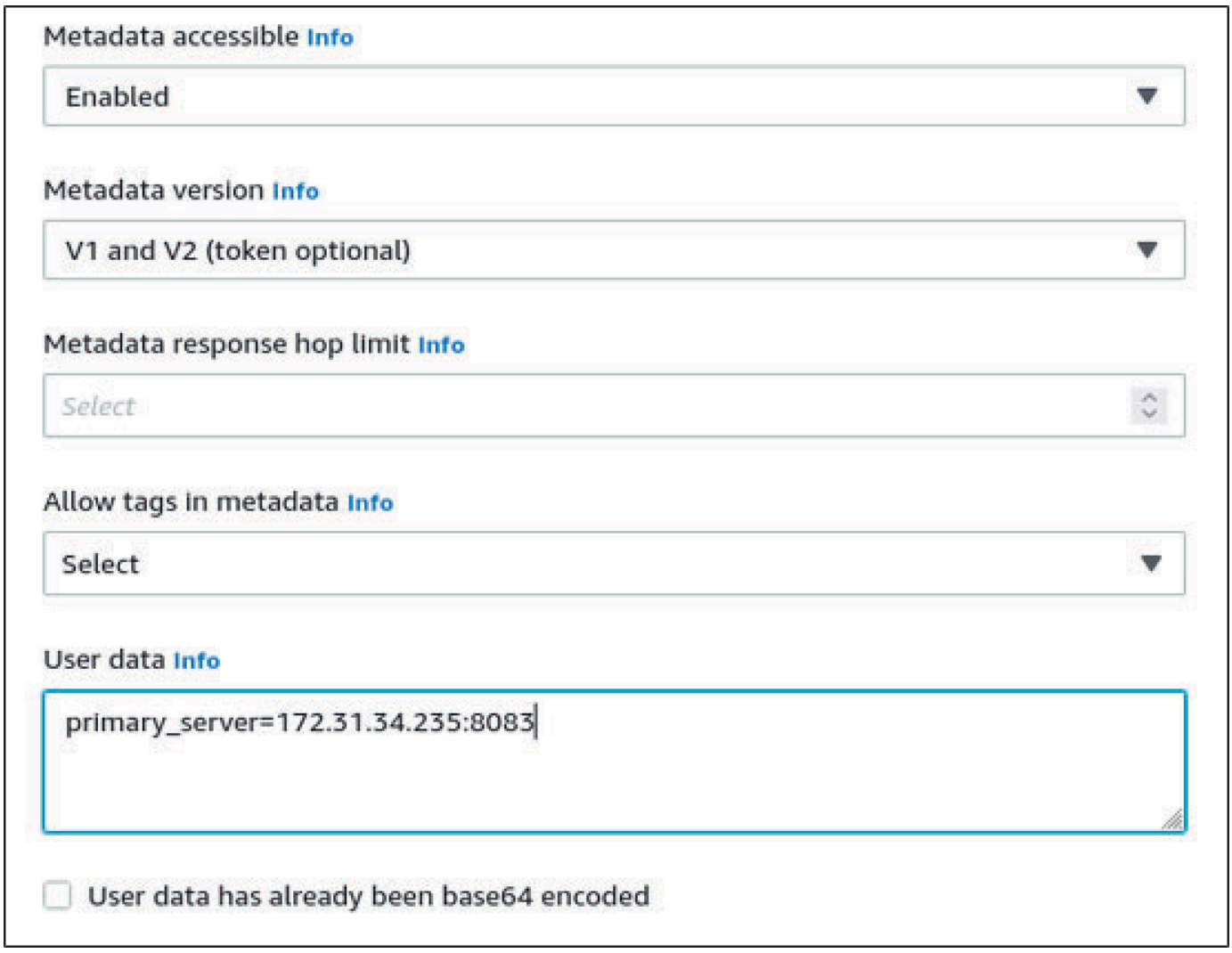
8. Na seção Armazenamento, para o volume raiz (o disco do sistema), escolha um tamanho adequado ao seu ambiente. O tamanho mínimo de disco do sistema recomendado para o sistema Linux é de 30 GiB. Para fornecer espaço para contêineres de disco virtual e outros requisitos de armazenamento, é possível adicionar mais armazenamento agora ou depois de executar a instância. Mas o tamanho do disco do sistema deve cobrir os requisitos do sistema Linux, incluindo todas as aplicações e utilitários que você planeja instalar.

   > **Note**
   >
   > Recomendamos que você crie volumes de armazenamento separados para os dados da aplicação Charon (por exemplo, imagens de disco). Se necessário, você poderá migrar esses volumes para outra instância posteriormente.

9. Expanda Detalhes avançados e marque a caixa de seleção Especificar opções de CPU. Três que têm maior probabilidade de serem úteis para um ambiente de emulador Charon são mostradas na imagem a seguir como exemplos.

10. Para um sistema de servidor de licenças VE com uma versão anterior à 1.1.23, você deverá atribuir um perfil do IAM necessário à instância. Ele deve ser um perfil que permita a ação `ListUsers`. Para atribuir um perfil, na seção expandida Detalhes avançados, selecione um perfil em Perfil da instância do IAM ou escolha Criar um perfil do IAM. Para obter mais informações, consulte Funções do IAM para Amazon EC2.
11. Se sua instância for baseada em uma AWS Marketplace imagem Charon AL e você planeja usar os servidores de licenças públicas operados pela Stromasys, você deve adicionar as informações correspondentes à configuração da instância antes de executar a instância.

    Insira as informações do servidor de licenças AutoVE, conforme mostrado na imagem a seguir.



As seguintes opções de configuração de dados do usuários são válidas:

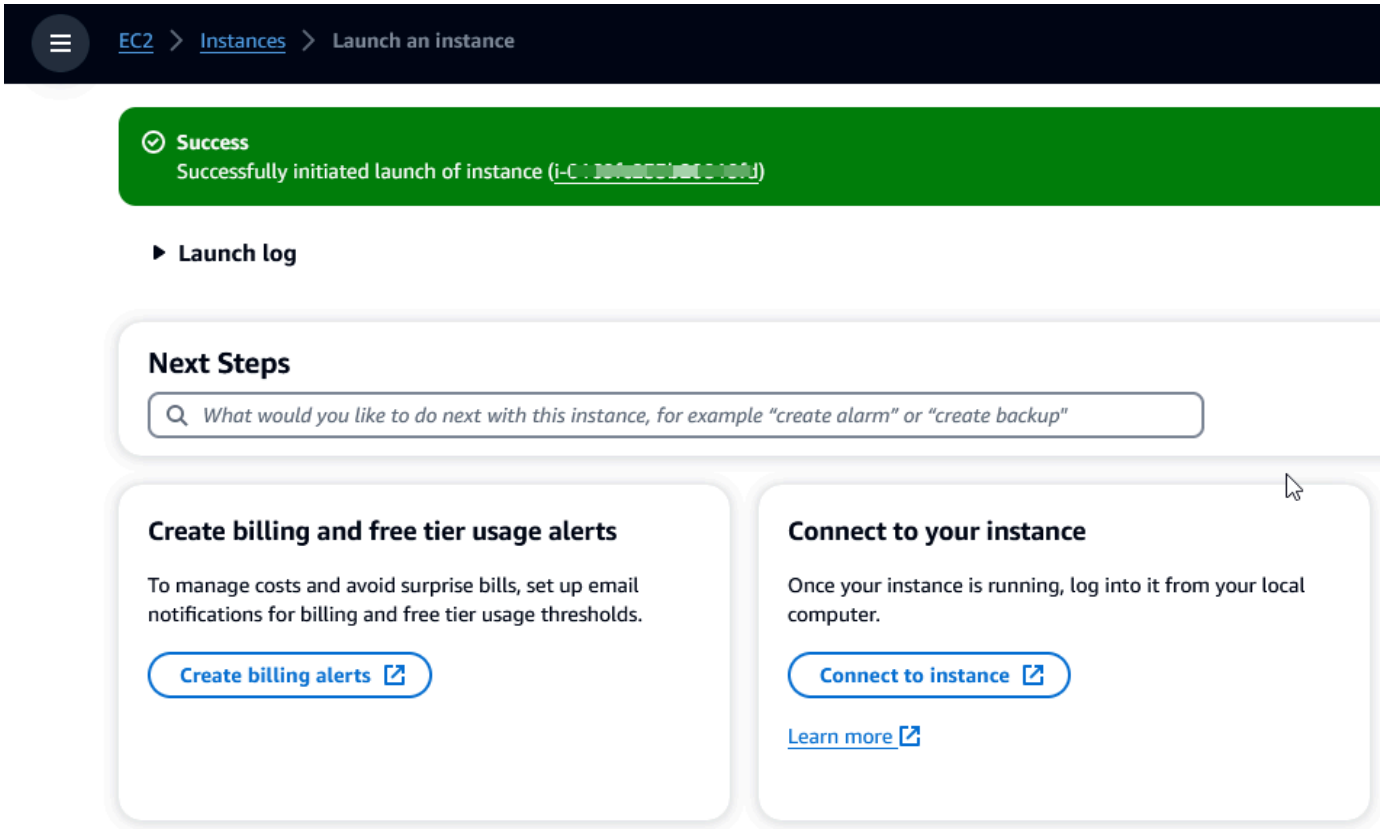
- **`primary_server=`***`<ip-address>`*`[:`*`<port>`*`]`
- **`backup_server=`***`<ip-address>`*`[:`*`<port>`*`]`

Em que

- <ip-address> significa o endereço IP do servidor primário e do servidor de backup, conforme aplicável.
- <port> significa uma porta TCP não padrão usada para se comunicar com o servidor de licenças (padrão: TCP/8083).

> **Note**
>
> Pelo menos um servidor de licenças deve ser configurado na execução inicial para ativar o modo AutoVE. Caso contrário, a instância será vinculada a um dos servidores de licenças públicas operados pela Stromasys.

12. No painel Resumol, escolha Executar instância. Depois de um tempo, você receberá a seguinte mensagem de êxito:



13. No canto inferior direito da tela, escolha Visualizar todas as instâncias.
14. Para ver os detalhes da instância, marque a caixa de seleção à esquerda da linha que representa a instância na tabela Instâncias. Os detalhes da instância serão exibidos na metade inferior da tela. Para obter informações sobre como se conectar à sua instância, consulte Connect no Guia do usuário do Amazon EC2.

AWS Modernização do mainframe e reformulação da plataforma com a NTT DATA

AWS A modernização do mainframe oferece uma variedade de Amazon Machine Images (AMIs). Essas AMIs facilitam o rápido provisionamento de instâncias do Amazon EC2, criando um ambiente personalizado para rehospedar e reformular aplicativos de mainframe usando o NTT Data. AWS Este guia fornece as etapas necessárias para acessar e usar essas AMIs.

Pré-requisitos

- Certifique-se de ter acesso de administrador a uma AWS conta na qual você possa criar instâncias do Amazon EC2.
- Verifique se o serviço de modernização do AWS mainframe está disponível na região em que você planeja criar as instâncias do Amazon EC2. Veja a lista de serviços da AWS disponíveis por região.
- Identifique a Amazon VPC em que você deseja criar as instâncias do Amazon EC2.

Inscrever-se para receber a imagem de máquina da Amazon

Após a inscrição em um produto do AWS Marketplace, você pode executar uma instância usando a AMI do produto.

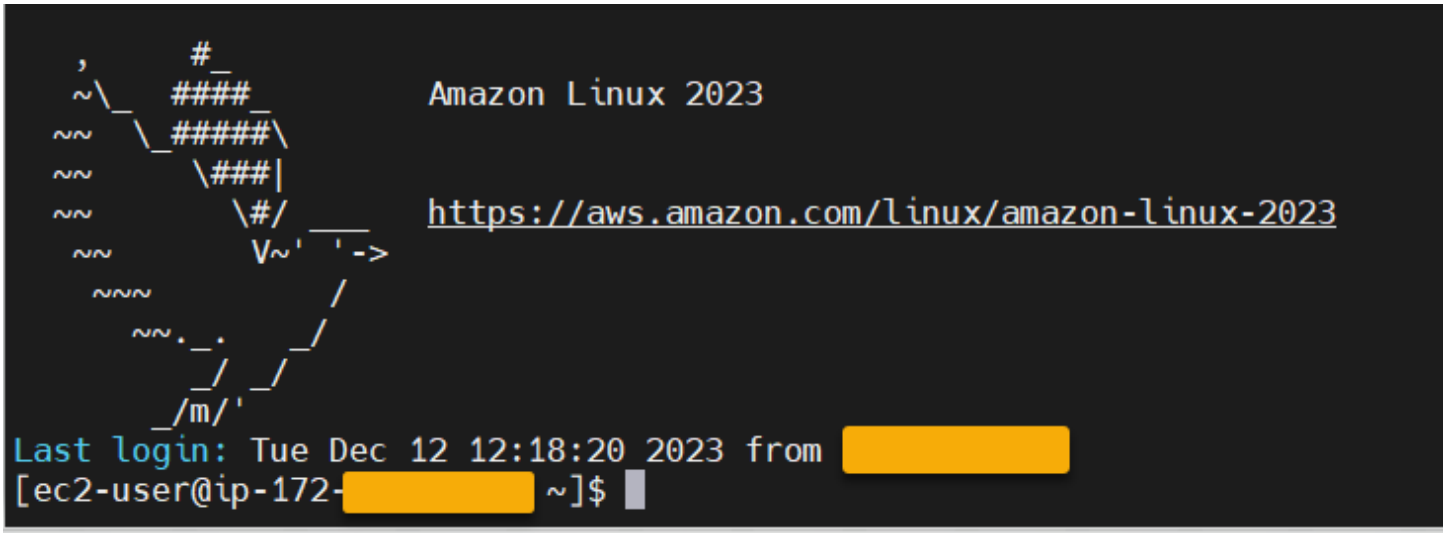
1. Faça login no AWS Management Console e abra o AWS Marketplace console em https://console.aws.amazon.com/marketplace.
2. Escolha Manage subscriptions (Gerenciar assinaturas).
3. Copie e cole o seguinte link na barra de endereço do navegador: https://aws.amazon.com/marketplace/pp/prodview-eg227ymldsnx2
4. Escolha Continue to Subscribe (Continuar para assinar).
5. Se os Termos e Condições forem aceitáveis, escolha Aceitar Termos. A assinatura pode levar alguns minutos para ser processada.
6. Aguarde até que uma mensagem de agradecimento seja exibida. Essa mensagem confirma que você se inscreveu com êxito no produto.
7. No painel de navegação esquerdo, escolha Gerenciar assinaturas. Essa visualização mostra todas as suas assinaturas.

Inicie a replataforma de modernização de AWS mainframe com a instância NTT DATA

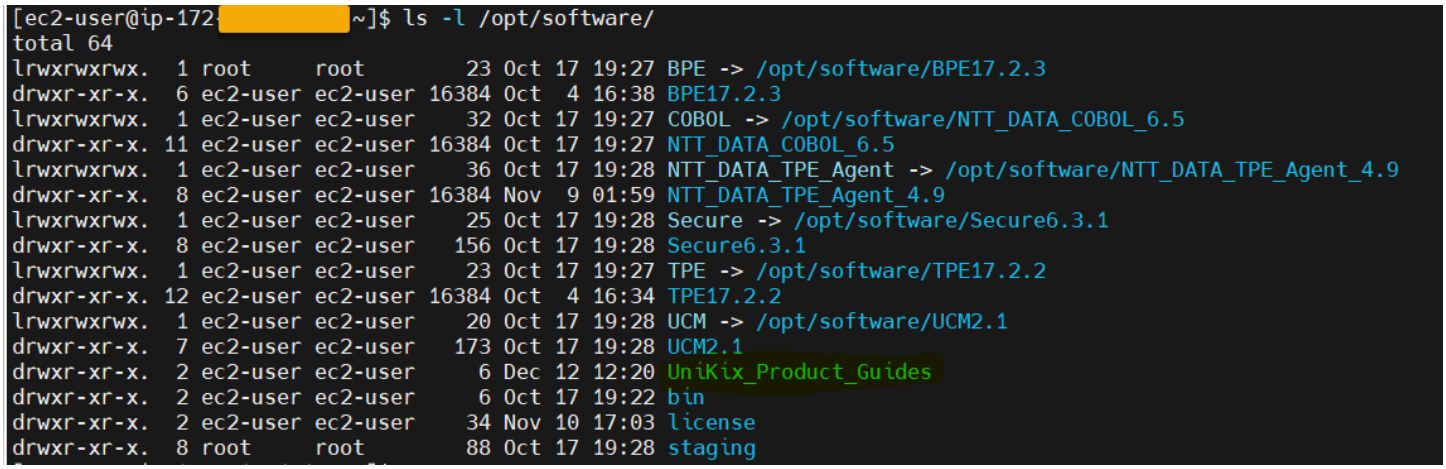
1. Abra o AWS Marketplace console em https://console.aws.amazon.com/marketplace.
2. No painel de navegação esquerdo, escolha Gerenciar assinaturas.
3. Encontre a AMI que você deseja iniciar e escolha Executar nova instância.
4. Em Região, selecione a região listada como permitida.
5. Escolha Continuar para executar por meio do EC2. Essa ação direcionará você para o console do Amazon EC2.
6. Insira um nome para o servidor.
7. Selecione um tipo de instância que corresponda aos requisitos de desempenho e custo do seu projeto. O ponto de partida sugerido para o tamanho da instância é `c5.2xLarge`.
8. Escolha um par de chaves existente ou crie um e salve-o. Para obter informações sobre pares de chaves, consulte os pares de chaves do Amazon EC2 e as instâncias Linux no Guia do usuário do Amazon EC2.
9. Edite as configurações de rede e escolha a VPC na lista de permissões e a sub-rede apropriada.
10. Escolha um grupo de segurança existente ou crie um. Se for uma instância do Amazon EC2 do Enterprise Server, é normal permitir o tráfego TCP para as portas 86 e 10086 para administrar a configuração do Micro Focus.
11. Configure o armazenamento para a instância do Amazon EC2.
12. Revise o resumo e envie a Executar instância. Para que a execução tenha êxito, o tipo de instância deve ser válido. Se houver falha na execução, escolha Editar configuração da instância e escolha um tipo de instância diferente.
13. Depois de ver a mensagem de êxito, escolha Conectar-se à Instância.
14. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
15. No painel de navegação esquerdo, no menu Instâncias, escolha Instâncias.
16. No painel principal, verifique o status da sua instância.

Conceitos básicos da NTT Data

Depois de provisionar a instância do Amazon EC2, faça o SSH nela com o nome de usuário `ec2-user`. A tela será exibida como a imagem a seguir.



Abaixo da pasta `/opt/software/`, há uma pasta chamada `UniKix_Product_Guides`, conforme mostrado na imagem a seguir.



A pasta `UniKix_Product_Guides` inclui a documentação dos seguintes componentes que estão instalados nessa instância do Amazon EC2:

- NTT DATA TPE
- NTT DATA BPE
- NTT DATA Enterprise COBOL
- NTT DATA Secure UniKix
- Gerente UniKix central da NTT DATA

A pasta `software` que aparece na imagem anterior tem os binários dos componentes listados acima.

Depois de validar com sucesso a instância do Amazon EC2, comece a AWS usar a Mainframe Modernization Replatform com a NTT DATA seguindo a documentação da NTT Data.

Aplicações na modernização AWS do mainframe

Se você é iniciante na modernização de AWS mainframe, consulte os seguintes tópicos para começar:

- O que é modernização AWS do mainframe?
- Configurar a modernização AWS do mainframe
- Tutorial: Configurar o tempo de execução gerenciado para o AWS Blu Age
- Tutorial: Configurar o tempo de execução gerenciado para a Micro Focus

Um aplicativo na modernização do AWS mainframe contém uma carga de trabalho migrada do mainframe. A aplicação é análoga a uma workload no mainframe e está associada a um ambiente de runtime. Você pode adicionar arquivos em lotes e conjuntos de dados às aplicações e monitorá-las à medida que são executadas. Você cria aplicativos de modernização de AWS mainframe para cada carga de trabalho migrada. Ao criar um aplicativo de modernização de AWS mainframe, você especifica o mecanismo no qual o aplicativo é executado ao criá-lo. Escolha AWS Blu Age se estiver usando o padrão de refatoração automatizado e escolha Micro Focus se estiver usando o padrão de replataforma.

Tópicos

- Crie um aplicativo de modernização AWS de mainframe
- Implemente um aplicativo de modernização AWS de mainframe
- Atualizar um aplicativo de modernização AWS de mainframe
- Excluir um aplicativo de modernização de AWS mainframe de um ambiente
- Excluir um aplicativo de modernização AWS de mainframe
- Envie ou cancele trabalhos em lote para aplicativos de modernização de AWS mainframe
- Importe conjuntos de dados para aplicativos de modernização AWS de mainframe
- Gerencie transações para aplicativos de modernização AWS de mainframe
- Crie AWS recursos para um aplicativo migrado
- Configurar a aplicação gerenciada
- AWS Referência de definição de aplicativo de modernização de mainframe
- AWS Referência de definição do conjunto de dados de modernização de mainframe

Crie um aplicativo de modernização AWS de mainframe

Use o console de modernização de AWS mainframe para criar um aplicativo de modernização de AWS mainframe.

Estas instruções supõem que você tenha concluído as etapas em Configurar a modernização AWS do mainframe.

Cria uma aplicação

Para criar um aplicativo.

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que você deseja criar o aplicativo.
3. Na página Aplicativos, escolha Criar aplicativo.
4. Na página Especificar informações básicas, na seção Nome e descrição, insira um nome para a aplicação.
5. (Opcional) No campo Descrição, digite uma descrição breve para a aplicação. Essa descrição pode ajudar você e outros usuários a identificar a finalidade da aplicação.
6. Na seção Tipo de mecanismo, escolha Blu Age para refatoração automatizada ou Micro Focus para redefinição da plataforma.
7. Na seção Chave KMS, escolha Personalizar configurações de criptografia se quiser usar uma AWS KMS chave gerenciada pelo cliente. Para ter mais informações, consulte Criptografia de dados em repouso para o serviço de modernização AWS de mainframe.

   > **Note**
   >
   > Por padrão, a Modernização do AWS Mainframe criptografa seus dados com uma AWS KMS chave que a Modernização do AWS Mainframe possui e gerencia para você. No entanto, você pode optar por usar uma AWS KMS chave gerenciada pelo cliente.

8. (Opcional) Escolha uma AWS KMS chave por nome ou nome de recurso da Amazon (ARN) ou escolha Criar uma AWS KMS chave para acessar o AWS KMS console e criar uma nova AWS KMS chave.
9. (Opcional) Na seção Tags, escolha Adicionar nova tag para adicionar uma ou mais tags de aplicação à sua aplicação. Uma tag de aplicativo é um rótulo de atributo personalizado que ajuda você a organizar e gerenciar seus AWS recursos).

10. Escolha Próximo.
11. Na seção Recursos e configurações, use o editor embutido para inserir a definição da aplicação. Como alternativa, escolha Usar um arquivo JSON de definição de aplicação em um bucket do Amazon S3 e forneça a localização da definição da aplicação que você deseja usar. Para obter mais informações, consulte AWS Exemplo de definição do aplicativo Blu Age ou Definição de aplicação da Micro Focus.
12. Escolha Próximo.
13. Na página Revisar e criar, analise as informações fornecidas e selecione Criar fonte de dados.

Implemente um aplicativo de modernização AWS de mainframe

Use o console de modernização de AWS mainframe para implantar um aplicativo de modernização de AWS mainframe.

Estas instruções supõem que você tenha concluído as etapas em Configurar a modernização AWS do mainframe.

Implantar uma aplicação

Para executar um aplicativo de modernização de AWS mainframe, você deve primeiro implantá-lo em um ambiente de tempo de execução. Uma aplicação pode ter mais de uma versão. Cada versão de uma aplicação tem sua própria definição de aplicação. Para implantar uma aplicação, você deve especificar a versão que deseja implantar.

Você pode implantar somente uma versão de um determinada aplicação por vez. Se você implantar uma versão de uma aplicação e decidir implantar uma versão diferente, primeiro interrompa a aplicação se ela estiver em execução.

Para implantar uma aplicação

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que você deseja criar o aplicativo.
3. Na página Aplicações, escolha a aplicação que você deseja implementar.
4. Escolha Implantar aplicação.
5. Na seção Versões disponíveis, escolha a versão que você deseja implantar.
6. Na seção Ambientes, escolha um ambiente de runtime no qual você deseja que a aplicação seja executada.

7. Escolha Implantar.

Para implantar uma versão diferente de uma aplicação implantada

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que você deseja criar o aplicativo.
3. Na página Aplicações, escolha a aplicação que você deseja implementar.
4. No menu Ações, escolha Interromper aplicação.
5. Depois que a aplicação for interrompida, escolha Implantar aplicação.
6. Na seção Versões disponíveis, escolha a versão que você deseja implantar. Na seção Ambientes, o ambiente no qual a aplicação já está implantada é pré-selecionado.
7. Escolha Implantar.

Atualizar um aplicativo de modernização AWS de mainframe

Use o console de modernização de AWS mainframe para atualizar um aplicativo de modernização de AWS mainframe.

Estas instruções supõem que você tenha concluído as etapas em Configurar a modernização AWS do mainframe.

Atualizar uma aplicação

Um aplicativo de modernização de AWS mainframe pode ter várias versões, cada uma com sua própria definição de aplicativo. Para atualizar uma aplicação, forneça uma nova definição de aplicação. Isso cria uma nova versão da aplicação.

Para atualizar uma aplicação

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que o aplicativo que você deseja atualizar foi criado.
3. Na página Aplicações, escolha a aplicação que deseja atualizar.
4. Na página de detalhes da aplicação, na seção Definição atual, escolha Editar para atualizar a definição atual da aplicação.

5. Na página Atualizar a aplicação, use o editor em linha para atualizar a definição atual da aplicação.

   Como alternativa, escolha Usar um arquivo JSON de definição de aplicação em um bucket do Amazon S3 e forneça a localização da definição da aplicação que você deseja usar. Para obter mais informações, consulte AWS Exemplo de definição do aplicativo Blu Age ou Definição de aplicação da Micro Focus.
6. Quando terminar de atualizar a definição da aplicação, escolha Atualizar.

> **Note**
>
> Depois de atualizar a aplicação, você deve implantá-la novamente. Para ter mais informações, consulte Implemente um aplicativo de modernização AWS de mainframe.

Excluir um aplicativo de modernização de AWS mainframe de um ambiente

Você pode excluir um aplicativo de modernização de AWS mainframe de um ambiente usando o console de modernização de AWS mainframe.

Estas instruções supõem que você tenha concluído as etapas em Configurar a modernização AWS do mainframe.

Excluir uma aplicação de um ambiente

Se você precisar excluir um aplicativo de modernização de AWS mainframe e ele estiver em execução, certifique-se de interrompê-lo primeiro. Você pode ver o status da aplicação na página Aplicações.

Para excluir uma aplicação de um ambiente

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que o aplicativo que você deseja excluir do ambiente foi criado.
3. Na página Aplicações, escolha a aplicação que você deseja excluir do ambiente e selecione Ações.

4. (Opcional) Se o status da aplicação for `Running`, escolha Interromper aplicação.
5. Escolha Excluir do ambiente.

   O processo de exclusão iniciará imediatamente.

Excluir um aplicativo de modernização AWS de mainframe

Use o console de modernização de AWS mainframe para excluir um aplicativo de modernização de AWS mainframe.

Estas instruções supõem que você tenha concluído as etapas em Configurar a modernização AWS do mainframe.

Deleta o aplicativo

Se você precisar excluir um aplicativo de modernização de AWS mainframe e ele estiver em execução, certifique-se de interrompê-lo primeiro. Você pode ver o status da aplicação na página Aplicações.

Como excluir uma aplicação

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que o aplicativo que você deseja excluir foi criado.
3. Na página Aplicações, escolha a aplicação que você deseja excluir e selecione Ações.
4. (Opcional) Se o status da aplicação for `Running`, escolha Interromper aplicação.
5. Selecione Excluir aplicativo.
6. Na janela Excluir aplicação, insira `delete` para confirmar que você deseja excluir a aplicação e escolha Excluir.

Envie ou cancele trabalhos em lote para aplicativos de modernização de AWS mainframe

Na Modernização do AWS Mainframe, você pode enviar trabalhos em lotes para seus aplicativos. Você pode enviar ou cancelar trabalhos em lotes e revisar detalhes sobre execuções de trabalhos em lotes. Cada vez que você envia um trabalho em lotes, a Modernização do AWS Mainframe cria

uma execução de trabalho em lotes separada. Você pode monitorar a execução dessa tarefa. Você pode pesquisar trabalhos em lote por nome e fornecer arquivos JCL ou de script para trabalhos em lote.

> **Important**
>
> Se você cancelar um trabalho em lote, isso não excluirá o trabalho. Ele cancela uma execução específica do trabalho em lotes. Os registros do trabalho em lotes permanecem disponíveis para você visualizar os detalhes da execução do trabalho em lotes.

Se seu trabalho em lotes exigir acesso a um ou mais conjuntos de dados, use o console de modernização do AWS mainframe ou o AWS Command Line Interface (AWS CLI) para importar os conjuntos de dados. Para ter mais informações, consulte Importe conjuntos de dados para aplicativos de modernização AWS de mainframe.

Estas instruções pressupõem que você tenha concluído as etapas em Configurar a modernização AWS do mainframe e em Crie um aplicativo de modernização AWS de mainframe.

Tópicos

Enviar um trabalho em lote

Para enviar um trabalho em lote

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região para a qual o aplicativo para o qual você deseja enviar um trabalho em lotes foi criado.
3. Na página Aplicações, escolha a aplicação para o qual você deseja enviar um trabalho em lotes.

> **Note**
>
> Antes de enviar uma tarefa em lotes para uma aplicação, é necessário implantar a aplicação com sucesso.

4. Na página de detalhes da aplicação, selecione Trabalhos em lote.
5. Escolha Enviar trabalho.
6. Na seção Selecionar um script, escolha um script. É possível procurar o script que deseja pelo nome do.
7. Escolha Enviar trabalho.

Reiniciar um trabalho em lotes

Para reiniciar um trabalho em lotes

> **Important**
>
> A reinicialização do trabalho em lote está disponível somente nas versões 8.0.6 ou superiores do Micro Focus Environment Engine. Você também precisa ter um sistema de arquivos EFS ou FSx conectado ao seu ambiente.

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que o aplicativo e seu trabalho em lotes foram criados.
3. Na página Aplicativos, escolha o aplicativo em que você deseja reiniciar um trabalho em lotes.
4. Na página de detalhes da aplicação, selecione Trabalhos em lote.
5. Selecione o trabalho em lotes que você deseja reiniciar na lista gerada. Navegue até o menu Ações e escolha Reiniciar tarefa.
6. Especifique como você deseja reiniciar o trabalho em lotes. Você pode optar por reiniciar do início ou reiniciar usando etapas ou procsteps.
   - A opção Reiniciar do início permite que você reinicie todas as etapas de um trabalho em lotes desde o início.
   - Com a opção Reiniciar usando etapas ou procsteps, você pode escolher uma etapa específica ou procstep (etapa do procedimento) que deseja reiniciar e, opcionalmente, uma etapa ou procstep após a qual terminar.

> **Note**
>
> A etapa final ou procstep deve ser maior ou igual ao número da etapa inicial ou do procstep.

7. Escolha Enviar trabalho.

Cancelar um trabalho em lotes

Quando você cancela um trabalho em lotes, ele não exclui um trabalho em lotes, mas a execução de tarefas para esse trabalho em lotes. Você ainda pode ver os detalhes do seu trabalho em lotes.

Para cancelar um trabalho em lotes

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a Região com o aplicativo para seus trabalhos em lotes.
3. Na lista de trabalhos em lotes, localize e selecione o trabalho em lotes que você deseja cancelar.
4. Escolha Ações e escolha Cancelar trabalho.
5. Escolha Cancelar trabalho em lotes.

Isso cancelará todas as tarefas de trabalho em lote que você tenha agendado para execução.

Importe conjuntos de dados para aplicativos de modernização AWS de mainframe

Com a modernização do AWS mainframe, você pode importar conjuntos de dados para usar com seus aplicativos. É possível especificar os conjuntos de dados em um arquivo JSON armazenado em um bucket do Amazon S3, ou os valores de configuração do conjunto de dados separadamente. Depois de importar os conjuntos de dados, você pode revisar os detalhes da tarefa de importação para confirmar se os conjuntos de dados desejados foram importados. Todos os conjuntos de dados catalogados de uma aplicação são informados juntos no console.

Use o console de modernização de AWS mainframe para importar conjuntos de dados para um aplicativo de modernização de AWS mainframe.

Estas instruções pressupõem que você tenha concluído as etapas em Configurar a modernização AWS do mainframe e em Crie um aplicativo de modernização AWS de mainframe.

Importar um conjunto de dados

Para importar um conjunto de dados

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que o aplicativo para o qual você deseja importar conjuntos de dados foi criado.
3. Na página Aplicações, escolha a aplicação para o qual você deseja importar conjuntos de dados.
4. Na página de detalhes da aplicação, selecione Conjuntos de dados.
5. Escolha Importar.
6. Execute um destes procedimentos:
   - Escolha Usar arquivo JSON de configuração do conjunto de dados em um bucket do Amazon S3 e forneça a localização da configuração do conjunto de dados.
   - Escolha Especificar os valores de configuração do conjunto de dados separadamente com a configuração guiada. Consulte the section called “Referência de definição do conjunto de dados” para obter detalhes específicos da definição.

     Insira o nome, a organização do conjunto de dados (VSAM, GDG, PO, PS), a localização e a localização externa do Amazon S3 e as configurações de parâmetros para cada valor de configuração do conjunto de dados. Na configuração guiada, você também pode escolher Gerar JSON para revisar a configuração JSON a partir de sua entrada.
7. Selecione Enviar.

Gerencie transações para aplicativos de modernização AWS de mainframe

Com a modernização do AWS mainframe, você pode executar um aplicativo, por solicitação, ao mesmo tempo que muitos outros usuários que enviam solicitações para executar o mesmo aplicativo usando os mesmos arquivos e programas. Uma única transação consiste em um ou mais programas de aplicações que realizam o processamento necessário.

Estas instruções pressupõem que você tenha concluído as etapas em Configurar a modernização AWS do mainframe e em Crie um aplicativo de modernização AWS de mainframe.

Gerencie transações para aplicações

Para gerenciar transações para aplicações

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que o aplicativo que você deseja executar foi criado.
3. Na página Aplicativos, escolha o aplicativo em que você deseja gerenciar as transações.
4. Na guia Transações, em Recursos da transação, escolha como você deseja que seus recursos sejam exibidos na lista suspensa. Você pode exibir recursos de acordo com os recursos da transação, grupos, listas ou SITs.

   - Os recursos de transação permitem que você escolha o tipo de recurso de acordo com as definições de arquivo, de transação, de programa ou de fila de dados transitórios.

     > **Note**
     >
     > O serviço de modernização de AWS mainframe oferece suporte a tipos de recursos adicionais para gerenciar transações para aplicativos e pode ser acessado no console.

   - Os grupos são uma coleção de recursos de transações. Você pode escolher grupos que deseja associar ao seu recurso de transação.
   - As listas são coleções ordenadas de grupos. Você pode ver todos os seus recursos e grupos de transações em uma exibição de lista. A lista de inicialização determina quais recursos são carregados quando o servidor é inicializado.
     - Com o mecanismo de refatoração AWS Blu Age, você especifica as listas a serem incluídas na inicialização. Não há limite para o número de listas.
     - Com o mecanismo de redefinição da plataforma da Micro Focus, você pode especificar até quatro listas em um SIT.
   - A SIT (Tabela de Inicialização do Sistema) exibe todas as configurações de transação disponíveis. Você pode encontrar SITs de acordo com as propriedades (nome, descrição e listas de inicialização). Você também pode escolher listas para associar ao SIT escolhido.

> **Note**
>
> Os SITs são aplicáveis somente ao mecanismo de redefinição da plataforma da Micro Focus.

5. Escolha um recurso de transação para exibir todas as informações do recurso. Você também pode visualizar todos os atributos associados ao seu recurso de transação.

Crie AWS recursos para um aplicativo migrado

Para executar seu aplicativo migrado em AWS, você deve criar alguns AWS recursos com outros Serviços da AWS. Os recursos que você precisa criar incluem o seguinte:

- Um bucket do S3 para armazenar o código da aplicação, a configuração, os arquivos de dados e outros artefatos necessários.
- Um banco de dados Amazon RDS ou Amazon Aurora para armazenar os dados que a aplicação exige.
- Um AWS KMS key, que é exigido AWS Secrets Manager para criar e armazenar segredos.
- Um segredo do Secrets Manager para manter as credenciais do banco de dados.

> **Note**
>
> Cada aplicação migrada exige seu próprio conjunto desses recursos. Esse é um conjunto mínimo. Sua aplicação também pode exigir recursos adicionais, como segredos do Amazon Cognito ou filas do MQ.

Permissões obrigatórias

Verifique se você tenha as seguintes permissões:

- `s3:CreateBucket`, `s3:PutObject`
- `rds:CreateDBInstance`
- `kms:CreateKey`
- `secretsmanager:CreateSecret`

Bucket do Amazon S3

Tanto as aplicações refatoradas quanto as com redefinição da plataforma exigem um bucket do S3 que você configura da seguinte forma:

```
bucket-name/root-folder-name/application-name
```

nome-do-seu-bucket

Qualquer nome dentro das restrições de nomenclatura do Amazon S3. Recomendamos que você inclua o nome da Região da AWS como parte do nome do bucket. Crie o bucket na mesma região em que você planeja implantar a aplicação migrada.

root-folder-name

Nome necessário para satisfazer as restrições na definição do aplicativo, que você cria como parte do aplicativo de modernização do AWS mainframe. Você pode usar o `root-folder-name` para distinguir entre diferentes versões de uma aplicação, por exemplo, V1 e V2.

application-name

O nome do seu aplicativo migrado, por exemplo, PlanetsDemo ou BankDemo.

Banco de dados

Tanta as aplicações refatoradas quanto as com redefinição de plataforma podem exigir um banco de dados. Você deve criar, configurar e gerenciar o banco de dados de acordo com os requisitos específicos de cada mecanismo de tempo de execução. AWS A modernização do mainframe oferece suporte à criptografia em trânsito nesse banco de dados. Se você habilitar o SSL em seu banco de dados, certifique-se de especificar `sslMode` o segredo do banco de dados junto com os detalhes da conexão do banco de dados. Para ter mais informações, consulte AWS Secrets Manager segredo.

Se você usa o padrão de refatoração AWS Blu Age e precisa de um BluSam banco de dados, o mecanismo de execução do AWS Blu Age espera um banco de dados Amazon Aurora PostgreSQL, que você deve criar, configurar e gerenciar. O BluSam banco de dados é opcional. Crie esse banco de dados somente se a sua aplicação exigir isso. Para criar o banco de dados, siga as etapas em Criação de um cluster de banco de dados Amazon Aurora no Guia do usuário do Amazon Aurora.

Se você estiver usando o padrão de redefinição da plataforma da Micro Focus, poderá criar um banco de dados do Amazon RDS ou do Amazon Aurora PostgreSQL. Para criar o banco de dados,

siga as etapas em Criar uma instância de banco de dados Amazon RDS no Guia do usuário do Amazon RDS ou em Criar um cluster de banco de dados Amazon Aurora no Guia do usuário do Amazon Aurora.

Para ambos os mecanismos de tempo de execução, você deve armazenar as credenciais do banco de dados AWS Secrets Manager usando um AWS KMS key para criptografá-las.

AWS Key Management Service chave

Você deve armazenar as credenciais do banco de dados da aplicação com segurança no AWS Secrets Manager. Para criar um segredo no Secrets Manager, é necessário criar uma AWS KMS key. Para criar uma chave KMS, siga as etapas em Criar chaves no Guia do desenvolvedor do AWS Key Management Service .

Depois de criar a chave, você deve atualizar a política de chaves para conceder permissões de descriptografia para a modernização do AWS mainframe. Adicione as declarações de política a seguir:

```
{
   "Effect" : "Allow",
   "Principal" : {
   "Service" : "m2.amazonaws.com"
   },
   "Action" : "kms:Decrypt",
   "Resource" : "*"
   }
```

AWS Secrets Manager segredo

Você deve armazenar as credenciais do banco de dados da aplicação com segurança no AWS Secrets Manager. Para criar um segredo, siga as etapas em Criar um segredo de banco de dados no Guia do usuário do AWS Secrets Manager .

AWS A modernização do mainframe oferece suporte à criptografia em trânsito nesse banco de dados. Se você habilitar o SSL em seu banco de dados, certifique-se de especificar `sslMode` o segredo do banco de dados junto com os detalhes da conexão do banco de dados. Você pode especificar um dos seguintes valores para `sslMode`: `verify-full`, `verify-ca`, ou `disable`.

Durante o processo de criação da chave, escolha Permissões de recursos - opcional e, em seguida, escolha Editar permissões. No editor de políticas, adicione uma política baseada em recursos, como a seguinte, para recuperar o conteúdo dos campos criptografados.

```
{
   "Effect" : "Allow",
   "Principal" : {
   "Service" : "m2.amazonaws.com"
   },
   "Action" : "secretsmanager:GetSecretValue",
   "Resource" : "*"
   }
```

Configurar a aplicação gerenciada

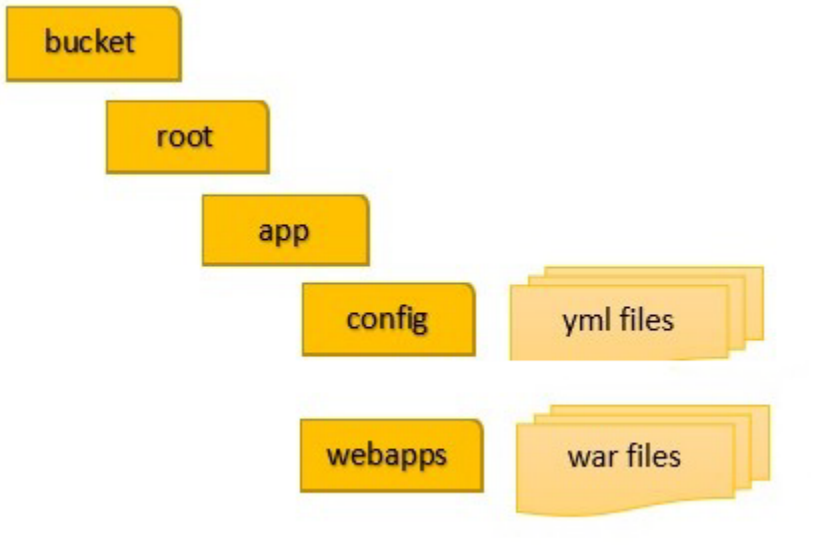
Você pode configurar a aplicação para incluir acesso a utilitários antigos. Você também pode personalizar propriedades adicionais. Para entender o que você pode configurar e onde, é útil entender a estrutura geral de um aplicativo modernizado do AWS Blu Age.

Tópicos

- Estrutura dos aplicativos gerenciados da AWS Blu Age
- Configurando o acesso a utilitários para aplicações gerenciadas
- Adicionar propriedades de configuração para o mecanismo AWS Blu Age

Estrutura dos aplicativos gerenciados da AWS Blu Age

Se você usa o padrão de refatoração do AWS Blu Age, o mecanismo de tempo de execução do AWS Blu Age espera a seguinte estrutura dentro da `application-name` pasta em seu bucket do S3:



config

Contém os arquivos YAML do seu projeto. Esses são os arquivos YAML específicos do seu aplicativo, normalmente chamados de algo parecido `application-planetsdemo.yaml` e não o `application-main.yaml` arquivo que a Modernização do AWS Mainframe fornece e configura automaticamente para você.

webapps

Contém os arquivos `war` da sua aplicação. Esses arquivos são uma saída do processo de modernização.

Uma aplicação também pode ter as seguintes pastas opcionais:

jics/sql

Contém o `initJics.sql` script que inicializa o banco de dados JICS para sua aplicação.

scripts

Contém scripts de aplicações, que você também pode fornecer diretamente nos arquivos `war`.

sql

Contém arquivos SQL da aplicação, que você também pode fornecer diretamente dentro dos arquivos `war`.

.lnk

Contém arquivos LNK da aplicação, que você também pode fornecer diretamente dentro dos arquivos `war`.

extra

Contém frascos que podem fornecer recursos adicionais para o aplicativo modernizado.

Gerenciando o consumo de memória Java de uma aplicação

Para gerenciar o consumo de memória Java para a aplicação, adicione um arquivo de propriedades chamado `tomcat.properties` à pasta `application-name`. Esse arquivo pode ter duas propriedades: `xms`, que especifica o consumo mínimo de memória Java e `xmx`, que especifica o consumo máximo de memória Java. A seguir, um exemplo dos conteúdos em um arquivo `tomcat.properties`:

```
xms=512M
xmx=1G
```

Os valores que você especifica para essas duas propriedades podem estar em qualquer uma das seguintes unidades:

- Bytes: não especifique uma unidade.
- Kilobytes: acrescente um K ao valor.
- Megabytes: acrescente um M ao valor.
- Gigabytes: acrescente um G ao valor.

Configurando o acesso a utilitários para aplicações gerenciadas

Ao refatorar um aplicativo de mainframe com o AWS Blu Age, talvez seja necessário fornecer suporte para vários programas utilitários de plataforma legados, como IDCAMS, INFUTILB, SORT e assim por diante, se seu aplicativo depender deles. AWS A refatoração do Blu Age fornece esse acesso com um aplicativo web dedicado que é implantado junto com aplicativos modernizados. Essa aplicação web requer um arquivo de configuração `application-utility-pgm.yml`, que você deve fornecer. Se você não fornecer esse arquivo de configuração, o aplicação web não poderá ser implantado junto com sua aplicação e não estará disponível.

Tópicos

Este tópico descreve todas as propriedades possíveis que você pode especificar no arquivo `application-utility-pgm.yml` de configuração, junto com seus padrões. O tópico descreve as propriedades obrigatórias e opcionais. O exemplo a seguir é um arquivo de configuração completo. Ele lista as propriedades na ordem que recomendamos. Em seguida, é possível usar esse exemplo como ponto de partida para seu próprio arquivo de configuração.

```
# If the datasource support mode is not static-xa, spring JTA transactions
 autoconfiguration must be disabled
 spring.jta.enabled: false
 logging.config: 'classpath:logback-utility.xml'

 # Encoding
 encoding: cp1047

 # Encoding to be used by INFUTILB and DSNUTILB to generate and read SYSPUNCH files
 sysPunchEncoding: cp1047

 # Utility database access
 spring.aws.client.datasources.primary.secret: `arn:aws:secretsmanager:us-
west-2:111122223333:secret:business-FfmXLG`

 treatLargeNumberAsInteger: false

 # Zoned mode : valid values = EBCDIC_STRICT, EBCDIC_MODIFIED, AS400
 zonedMode: EBCDIC_STRICT

 jcl.type: mvs

 # Unload properties
 # For date/time: if use database configuration is enabled, formats are ignored
 # For nbi; use hexadecimal syntaxe to specify the byte value
 unload:
  sqlCodePointShift: 384
  nbi:
    whenNull: "6F"
    whenNotNull: "00"
  useDatabaseConfiguration: false
  format:
    date: MM/dd/yyyy
```

```
    time: HH.mm.ss
    timestamp: yyyy-MM-dd-HH.mm.ss.SSSSSS
  chunkSize:500
  fetchSize: 500
  varCharIsNull: false
  columnFiller: space

# Load properties
# Batch size for DSNUTILB Load Task
load:
  sqlCodePointShift: 384
  batchSize: 500
  format:
    localDate: dd.MM.yyyy|dd/MM/yyyy|yyyy-MM-dd
    dbDate: yyyy-MM-dd
    localTime: 'HH:mm:ss|HH.mm.ss'
    dbTime: 'HH:mm:ss'

table-mappings:
  TABLE_1_NAME : LEGACY_TABLE_1_NAME
  TABLE_2_NAME : LEGACY_TABLE_2_NAME
```

Propriedades de configuração

Em seu arquivo de configuração, é possível especificar as propriedades a seguir.

spring.jta.enabled

Opcional. Controla se o suporte ao JTA está habilitado. Para utilitários, recomendamos que defina esse valor como `false`.

```
spring.jta.enabled : false
```

logging.config

Obrigatório. Especifica o caminho para o arquivo de configuração do registrador dedicado. Recomendamos que você use o nome `logback-utility.xml` e forneça esse arquivo como parte da aplicação modernizada. A maneira comum de organizar esses arquivos é colocar todos os arquivos de configuração do registrador no mesmo local, geralmente na subpasta `/config/logback` onde `/config` está a pasta que contém os arquivos de configuração YAML. Para obter mais informações, consulte o Capítulo 3: Configuração do Logback na documentação do Logback.

```
logging.config : classpath:logback-utility.xml
```

encoding

Obrigatório. Especifica o conjunto de caracteres que o programa utilitário usa. Na maioria dos casos, quando você migra das plataformas z/OS, esse conjunto de caracteres é uma variante do EBCDIC e deve corresponder ao conjunto de caracteres configurado para as aplicações modernizadas. Se não estiver definido, o padrão será `ASCII`.

```
encoding : cp1047
```

sysPunchEncoding

Opcional. Especifica o conjunto de caracteres que INFUTILB e DSNUTILB usam para gerar e ler arquivos SYSPUNCH. Se você usar os arquivos SYSPUNCH da plataforma antiga como estão, esse valor deve ser uma variante do EBCDIC. Se não estiver definido, o padrão será `ASCII`.

```
sysPunchEncoding : cp1047
```

Configuração da fonte de dados primária

Alguns utilitários relacionados ao banco de dados, como LOAD e UNLOAD, exigem acesso a um banco de dados de destino por meio de uma fonte de dados. Como outras definições de fonte de dados na modernização do AWS mainframe, esse acesso exige que você use. AWS Secrets Manager As propriedades que apontam para os segredos adequados no Secrets Manager são as seguintes:

spring.aws.client.datasources.primary.secret

Opcional. Especifica o segredo no Secrets Manager que contém as propriedades da fonte de dados.

```
spring.aws.client.datasources.primary.secret: datasource-secret-ARN
```

spring.aws.client.datasources.primary.dbname

Opcional. Especifica o nome do banco de dados de destino se o nome do banco de dados não for fornecido diretamente no segredo do banco de dados, com a `dbname` propriedade.

```
spring.aws.client.datasources.primary.dbname: target-database-name
```

treatLargeNumberAsInteger

Opcional. Relacionado às especificidades do mecanismo de banco de dados Oracle e ao uso dos utilitários DSNTEP2/DSNTEP4. Se você definir esse sinalizador como verdadeiro, números grandes provenientes do banco de dados Oracle (NUMBER (38,0)) serão tratados como números inteiros. Padrão: `false`

```
treatLargeNumberAsInteger : false
```

zonedMode

Opcional. Define o modo zoneado para codificar ou decodificar tipos de dados zoneados. Essa configuração influencia a forma como os dígitos do sinal são representados. Os valores a seguir são válidos:

- EBCDIC_STRICT: padrão. Use uma definição estrita para o manuseio de sinais. Dependendo se o conjunto de caracteres é EBCDIC ou ASCII, a representação do dígito de sinal usa os seguintes caracteres:
  - Caracteres EBCDIC que correspondem a bytes (`Cn+Dn`) para representar intervalos de dígitos positivos e negativos (`+0` para `+9` para, `-0` para `-9`). Os caracteres são exibidos como `{`, `A` para `I`, `}`, `J` para `R`
  - Caracteres ASCII que correspondem a bytes (`3n+7n`) para representar intervalos de dígitos positivos e negativos (`+0` para `+9` para, `-0` para `-9`). Os caracteres são exibidos como `0` para 9, p para y
- EBCDIC_MODIFIED: use uma definição modificada para o tratamento de sinais. Tanto para EBDIC quanto para ASCII, a mesma lista de caracteres representa os dígitos do sinal, ou seja, mapeados `+0` para `+9` mapeados para `{` + `A` para `I` e `-0` para `-9` mapeados para `}` + `J` para `R`. \
- AS400: use para ativos antigos modernizados provenientes de plataformas iSeries (AS400).

```
zonedMode:EBCDIC_STRICT
```

jcl.type

Opcional. Indica o tipo antigo de scripts JCL modernizados. O utilitário IDCAMS usa essa configuração para personalizar o código de retorno se o JCL de invocação for do tipo `vse`. Os valores válidos são os seguintes:

- `mvs` (padrão)
- `vse`

```
jcl.type : mvs
```

Propriedades relacionadas aos utilitários de descarga de banco de dados

Use essas propriedades para configurar utilitários que descarregam tabelas de banco de dados em conjuntos de dados. Todas as propriedades a seguir são opcionais.

Este exemplo mostra todas as propriedades de descarga possíveis.

```
# Unload properties
 # For date/time: if use database configuration is enabled, formats are ignored
 # For nbi; use hexadecimal syntaxe to specify the byte value
 unload:
 sqlCodePointShift: 0
 nbi:
 whenNull: "6F"
 whenNotNull: "00"
 useDatabaseConfiguration: false
 format:
 date: MM/dd/yyyy
 time: HH.mm.ss
 timestamp: yyyy-MM-dd-HH.mm.ss.SSSSSS
 chunkSize: 0
 fetchSize: 0
 varCharIsNull: false
 columnFiller: space
```

sqlCodePointTurno

Opcional. Especifica um valor inteiro que representa a mudança de ponto do código SQL usada nos dados. O padrão é 0. Isso significa que nenhuma mudança de ponto de código é feita. Alinhe essa configuração com o parâmetro de mudança de ponto do código SQL usado para aplicações

modernizadas. Quando a mudança de ponto de código está em uso, o valor mais comum para esse parâmetro é 384.

```
unload.sqlCodePointShift: 0
```

nbi

Opcional. Especifica um byte indicador nulo. Esse é um valor hexadecimal (como uma string) adicionado à direita do valor dos dados. Os dois valores possíveis são os seguintes:

- whenNull: adicione o valor hexadecimal quando o valor dos dados for nulo. O padrão é 6`. Às vezes, o valor alto FF é usado em vez disso.

  ```
  unload.nbi.whenNull: "6F"
  ```

- whenNotNull: adicione o valor hexadecimal quando o valor dos dados não for nulo, mas a coluna for anulável. O padrão é `00` (valor baixo).

  ```
  unload.nbi.whenNotNull: "00"
  ```

useDatabaseConfiguration

Opcional. Especifica as propriedades de formatação de data e hora. Isso é usado para lidar com objetos de data/hora em consultas UNLOAD. O padrão é `false`.

- Se definido como `true`, usa as propriedades `pgmDateFormat`, `pgmTimeFormat` e `pgmTimestampFormat` do arquivo de configuração principal (`application-main.yml`).
- Se definido como `false`, usa as seguintes propriedades de formatação de data e hora:
  - `unload.format.date`: especifica um padrão de formatação de data. O padrão é `MM/dd/yyyy`.
  - `unload.format.time`: especifica um padrão de formatação de hora. O padrão é `HH.mm.ss`.
  - `unload.format.timestamp`: especifica um padrão de formatação de carimbo de data/hora. O padrão é `yyyy-MM-dd-HH.mm.ss.SSSSSS`.

chunkSize

Opcional. Especifica o tamanho dos blocos de dados usados para criar conjuntos de dados SYSREC. Esses conjuntos de dados são o alvo da operação de descarregamento do conjunto de dados, com operações paralelas. O padrão é `0` (sem pedaços).

```
unload.chunkSize:0
```

fetchSize

Opcional. Especifica o tamanho da busca de dados. O valor é o número de registros a serem buscados ao mesmo tempo quando uma estratégia de fragmentos de dados é usada. Padrão: `0`.

```
unload.fetchSize:0
```

varCharIsNulo

Opcional. Especifica como lidar com uma coluna varchar não anulável com conteúdo em branco. O padrão é `false`.

Se você definir esse valor como `true`, o conteúdo da coluna será tratado como uma string vazia para fins de descarga, em vez de uma única string de espaço. Defina esse sinalizador somente `true` para o caso do mecanismo de banco de dados Oracle.

```
unload.varCharIsNull: false
```

columnFiller

Opcional. Especifica o valor a ser usado para preencher colunas descarregadas em colunas varchar. Os valores possíveis são espaço ou valores baixos. O padrão é espaço.

```
unload.columnFiller: space
```

Propriedades relacionadas ao carregamento do banco de dados

Use essas propriedades para configurar utilitários que carregam registros do conjunto de dados em um banco de dados de destino, por exemplo, DSNUTILB. Todas as propriedades a seguir são opcionais.

Este exemplo mostra todas as propriedades de carga possíveis.

```
# Load properties
 # Batch size for DSNUTILB Load Task
 load:
 sqlCodePointShift: 384
 batchSize: 500
```

```
format:
localDate: dd.MM.yyyy|dd/MM/yyyy|yyyy-MM-dd
dbDate: yyyy-MM-dd
localTime: HH:mm:ss|HH.mm.ss
dbTime: HH:mm:ss

table-mappings:
TABLE_1_NAME : LEGACY_TABLE_1_NAME
TABLE_2_NAME : LEGACY_TABLE_2_NAME
```

sqlCodePointTurno

Opcional. Especifica um valor inteiro que representa a mudança de ponto do código SQL usada nos dados. O padrão é 0, o que significa que as aplicações não alteram o ponto de código. Alinhe essa configuração com o parâmetro de mudança de ponto do código SQL usado para aplicações modernizadas. Quando você usa mudanças de ponto de código, o valor mais comum para esse parâmetro é 384.

```
load.sqlCodePointShift : 384
```

batchSize

Opcional. Especifica um valor inteiro que representa o número de registros a serem tratados antes de você enviar uma declaração de lote real para o banco de dados. O padrão é 0.

```
load.batchSize: 500
```

format

Opcional. Especifica os padrões de formatação de data e hora a serem usados para conversões de data/hora durante as operações de carregamento do banco de dados.

- `load.format.localDate`: Padrão de formatação de data local. Isso é padronizado como `dd.MM.yyyy|dd/MM/yyyy|yyyy-MM-dd`.
- `load.format.dbDate`: Padrão de formatação de data do banco de dados. Isso é padronizado como `yyyy-MM-dd`.
- `load.format.localTime`: Padrão de formatação da hora local. Isso é padronizado como `HH:mm:ss|HH.mm.ss`.
- `load.format.dbTime`: Padrão de formatação de hora do banco de dados. Isso é padronizado como `HH:mm:ss`.

mapeamentos de tabela

Opcional. Especifica uma coleção de mapeamentos fornecidos pelo cliente entre nomes de tabelas antigas e modernas. O programa utilitário DSNUTILB consome esses mapeamentos.

Especifique os valores no seguinte formato: MODERN_TABLE_NAME: LEGACY_TABLE_NAME

Exemplo:

```
table-mappings:
 TABLE_1_NAME : LEGACY_TABLE_1_NAME
 TABLE_2_NAME : LEGACY_TABLE_2_NAME
 ...
 TABLE_*N*_NAME : LEGACY_TABLE_*N*_NAME
```

> **Note**
>
> Quando a aplicação utilitário é iniciada, ele registra explicitamente todos os mapeamentos fornecidos.

Adicionar propriedades de configuração para o mecanismo AWS Blu Age

Você pode adicionar um arquivo na `config` pasta do seu aplicativo refatorado que lhe dará acesso aos novos recursos no mecanismo de tempo de execução do AWS Blu Age. Você deve nomear esse arquivo `user-properties.yml`. Esse arquivo não substitui a definição da aplicação, mas a estende. Este tópico descreve as propriedades que você pode incluir no arquivo `user-properties.yml`.

> **Note**
>
> Você não pode alterar alguns parâmetros porque eles são controlados pela modernização do AWS mainframe ou pela definição do aplicativo. Todos os parâmetros definidos na definição da aplicação têm prioridade sobre os parâmetros especificados em `user-properties.yml`.

Para obter mais informações sobre a estrutura de aplicações refatoradas, consulte Estrutura dos aplicativos gerenciados da AWS Blu Age.

O diagrama a seguir mostra onde localizar o `user-properties.yml` arquivo dentro da estrutura do aplicativo de amostra AWS Blu Age, PlanetsDemo.

```
PlanetsDemo-v1/
   ## config/
   #  ## application-PlanetsDemo.yml
   #  ## user-properties.yml
   ## jics/
   ## webapps/
```

Referência de propriedades de configuração

Esta é a lista de propriedades disponíveis. Todos os parâmetros são opcionais.

Tópicos

Propriedades da aplicação Gapwalk

bluesam.fileLoading.commitInterval

Opcional. O intervalo de confirmação do BlueSam.

Tipo: número

Padrão: 100000

card.encoding

Opcional. Codificação do cartão: para ser usado com `useControlMVariable`.

Tipo: string

Padrão: CP1145

checkinputfilesize

Opcional. Especifica se um cheque deve ser liberado se o tamanho do arquivo for múltiplo do tamanho do registro.

Tipo: booliano

Padrão: False

database.cursor.overflow.allowed

Opcional. Especifica se é permitido que o cursor transborde. Defina como `true` para realizar uma próxima chamada no cursor, seja qual for sua posição. Defina como `false` para verificar se o cursor está na última posição antes de realizar uma próxima chamada no cursor. Ative somente se o cursor for SCROLLABLE (SENSITIVE ou INSENSITIVE)

Tipo: booliano

Padrão: True

Simplificador de dados. onInvalidNumericDados

Opcional. Como reagir ao decodificar dados numéricos inválidos. Os valores permitidos são: `reject`, `toleratespaces`, `toleratespaceslowvalues` e `toleratemost`.

Tipo: string

Padrão: rejeitar

defaultKeepExistingArquivos

Opcional. Especifica se o valor anterior padrão do conjunto de dados deve ser definido.

Tipo: booliano

Padrão: False

disposition.checkexistence

Opcional. Especifica se deve liberar uma verificação da existência do arquivo para o conjunto de dados com DISP SHR ou OLD.

Tipo: booliano

Padrão: False

externalSort.threshold

Opcional. O limite de classificação: quando alternar para a classificação externa (mesclagem).

Tipo: string

Padrão: nulo

```
externalSort.threshold: 12MB
```

forceHR

Opcional. Especifica se o SYSPRINT legível por humanos deve ser usado no console ou na saída do arquivo.

Tipo: booliano

Padrão: False

forcedDate

Opcional. Força uma data e hora específicas no banco de dados. Use somente durante o desenvolvimento e o teste.

Padrão: nulo

```
forcedDate: 2022-08-26T12:59:58.123456+01:57
```

frozenDate

Opcional. Congela a data e a hora no banco de dados. Use somente durante o desenvolvimento e o teste.

Padrão: False

```
frozenDate: false
```

ims.messages.extendedSize

Opcional. Especifica se o extendedSize deve ser definido nas mensagens ims.

Tipo: booliano

Padrão: False

lockTimeout

Opcional. O tempo limite em milissegundos de uma transação quando não é possível adquirir um bloqueio dentro de um período de tempo especificado.

Tipo: número

Padrão: 500

mapTransfo.prefixes

Opcional. Lista de prefixos a serem usados ao transformar variáveis controlM. Cada um separado por vírgula.

Tipo: string

Padrão: &,@,%%

consulta. useConcatCondition

Opcional. Especifica se a condição da chave é criada por concatenação de chaves ou não.

Tipo: booliano

Padrão: False

rollbackOnRTE

Opcional. Especifica se a transação de unidade de execução implícita deve ser revertida em exceções de runtime.

Tipo: booliano

Padrão: False

sctThreadLimit

Opcional. O limite de threads para acionar scripts.

Tipo: número

Padrão: 5

sqlCodePointTurno

Opcional. A mudança de ponto do código sql. Muda o ponto de código dos caracteres de controle que podemos encontrar ao migrar dados rdbms antigos para um rdbms moderno. Por exemplo, você pode especificar `384` para corresponder ao `\u0180` caractere Unicode.

Tipo: número

Padrão: 0

sqlIntegerOverflowPermitido

Opcional. Especifica se é permitido o estouro de números inteiros SQL, ou seja, se é permitido colocar valores maiores na variável do host.

Tipo: booliano

Padrão: False

stepFailWhenAbend

Opcional. Especifica se a suspensão deve ser levantada se uma etapa falhar ou concluir a execução.

Tipo: booliano

Padrão: True

stopExecutionWhenProgNotFound

Opcional. Especifica se a execução deve ser interrompida se um programa não for encontrado. Se definido como `true`, interrompe a execução se um programa não for encontrado.

Tipo: booliano

Padrão: True

uppercaseUserInput

Opcional. Especifica se a entrada do usuário deve estar em maiúsculas.

Tipo: booliano

Padrão: True

useControlMVariable

Opcional. Especifica se a especificação Control-m deve ser usada para substituição de variáveis.

Tipo: booliano

Padrão: False

Propriedades do batchscript do Gapwalk

encoding

Opcional. A codificação usada em projetos de batchscript (não com groovy). Espera uma codificação válida `CP1047`, `IBM930`, `ASCII`, `UTF-8`...

Tipo: string

Padrão: ASCII

Propriedades Gapwalk Blugen

managers.trancode

Opcional. O gerenciador de diálogos trancode mapeamento. Permite mapear um código de transação JICS para um gerenciador de diálogos. O formato esperado é `trancode1:dialogManager1;trancode2:dialogManager2;`.

Tipo: string

Padrão: nulo

```
managers.trancode: OR12:MYDIALOG1
```

Propriedades do comando Gapwalk CL

commands-off

Opcional. Lista de comandos a serem desativados, separados por vírgula. Os valores permitidos são: `PGM_BASIC`, `RCVMSG`, `SNDRCVF`, `CHGVAR`, `QCLRDTAQ`, `RTVJOBA`, `ADDLFM`, `ADDPFM`, `RCVF`, `OVRDBF`, `DLTOVR`, `CPYF` e `SNDDTAQ`. Útil quando você deseja desativar ou substituir um programa existente. `PGM_BASIC`é um programa específico do AWS Blu Age Runtime projetado para fins de depuração.

Tipo: string

Padrão: nulo

spring.datasource.primary.jndi-name

Opcional. A principal fonte de dados Java Naming And Directory Interface (JNDI).

Tipo: string

Padrão: jdbc/primary

zonedMode

Opcional. O modo para codificar ou decodificar tipos de dados zoneados. Os valores permitidos são: `EBCDIC_STRICT`, `EBCDIC_MODIFIED` e `AS400`.

Tipo: string

Padrão: EBCDIC_STRICT

Propriedades do corredor Gapwalk CL

cl.configuration.context.encoding

Opcional. A codificação dos arquivos CL. Espera uma codificação válida `CP1047`, `IBM930`, `ASCII`, `UTF-8`...

Tipo: string

Padrão: CP297

cl.zonedMode

Opcional. O modo para codificar ou decodificar comandos da linguagem de controle (CL). Os valores permitidos são: `EBCDIC_STRICT`, `EBCDIC_MODIFIED` e `AS400`.

Tipo: string

Padrão: EBCDIC_STRICT

Propriedades Gapwalk JHDB

ims.programs

Opcional. Lista de programas IMS a serem usados. Separe cada parâmetro com um ponto e vírgula (`;`) e cada transação com uma vírgula (`,`). Por exemplo: `ims.programs: PCP008,PCT008;PCP054,PCT054;PCP066,PCT066;PCP068,PCT068;`

Tipo: string

Padrão: nulo

jhdb.checkpointPath

Opcional. Se `jhdb.checkpointPersistence` não for `none`, esse parâmetro permitirá que você configure o caminho de persistência do ponto de verificação (local de armazenamento do arquivo checkpoint.dat); todos os dados de pontos de verificação contidos no registro são serializados e armazenados em um arquivo (checkpoint.dat) localizado na pasta fornecida. Observe que somente os dados do ponto de verificação (ScriptID, StepID, posição do banco de dados e área do ponto de verificação) são afetados por esse backup.

Tipo: string

Padrão: file:./setup/

jhdb.checkpointPersistence

Opcional. O modo de persistência do ponto de verificação. Os valores permitidos são: `none`, `add` e `end`. Use `add` para manter os pontos de verificação quando um novo for criado e adicionado ao registro. Use `end` para manter o ponto de verificação no desligamento do servidor. Quaisquer outros valores desativam a persistência. Observe que sempre que um novo ponto de verificação for adicionado ao registro, todos os pontos de verificação existentes serão serializados e o arquivo será apagado. Não é um acréscimo aos dados existentes no arquivo. Portanto, dependendo do número de pontos de verificação, isso pode ter algum efeito no desempenho.

Tipo: string

Padrão: nenhum

jhdb.configuration.context.encoding

Opcional. A codificação JHDB (Java Hierarchical Database). Espera uma string de codificação válida `CP1047`, `IBM930`, `ASCII`, `UTF-8`...

Tipo: string

Padrão: CP297

jhdb. identificationCardData

Opcional. Usado para codificar alguns “dados do cartão de identificação do operador” no campo MID designado pelo parâmetro CARD.

Tipo: string

Padrão: ""

jhdb.lterm

Opcional. Permite que você force um ID de terminal lógico comum no caso de uma emulação de IMS. Se não for definido, o sessionId será usado.

Tipo: string

Padrão: nulo

jhdb.metadata.extrapath

Um parâmetro de configuração que especifica uma pasta raiz extra específica do runtime para as pastas psbs e dbds.

Tipo: string

Padrão: file:./setup/

> **Note**
>
> No momento, devido a restrições de implantação, você deve copiar seus diretórios dbds e psbs no diretório config da sua aplicação ou em um subdiretório do diretório config: por exemplo, config/setup
>
> ```
> config
> |- setup
>    |- dbds
>    |- psbs
> ```
>
> e definir em application-jhdb.yml
> `jhdb.metadata.extrapath: file: ./config/setup/`

jhdb.navigation.cachenexts

Opcional. A duração do cache (em milissegundos) usada na navegação hierárquica para um RDBMS.

Tipo: número

Padrão: 5000

jhdb.query. limitJoinUsage

Opcional. Especifica se o parâmetro limite de uso de junção deve ser usado em gráficos RDBMS.

Tipo: booliano

Padrão: True

jhdb. use-db-prefix

Opcional. Especifica se um prefixo de banco de dados deve ser ativado na navegação hierárquica para um RDBMS.

Tipo: booliano

Padrão: True

Propriedades do Gapwalk JICS

jics.data. dataJsonInitLocalização

Opcional. Localização do arquivo json preparado pelo Analyzer a partir da análise do CSD e usado para inicializar o banco de dados jics,

Tipo: string

Padrão: ""

jics.db. dataScriptLocation

Opcional. Localização do script initJics.sql, preparado pelo Analyzer a partir da análise das exportações de CSD do mainframe.

Tipo: string

Padrão: ""

jics.db. dataTestQueryLocalização

Opcional. Localização de um script sql contendo uma única consulta sql que deve retornar uma contagem de objetos (por exemplo: contagem do número de registros na tabela do programa jics). Se a contagem for igual a 0, o banco de dados será carregado usando o script `jics.db.dataScriptLocation`, caso contrário, o carregamento do banco de dados será ignorado.

Tipo: string

Padrão: ""

jics.db. ddlScriptLocation

Opcional. A localização do script Jics ddl. Permite que você inicie o esquema do banco de dados jics usando um script.sql.

Tipo: string

Padrão: ""

```
jics.db.ddlScriptLocation: ./jics/sql/jics.sql
```

jics.db. schemaTestQueryLocalização

Opcional. Localização do arquivo sql que deve conter uma consulta exclusiva que retorna o número de objetos no esquema jics (se houver).

Tipo: string

Padrão: ""

sucos. runUnitLauncherPool. Habilitar

Opcional. Especifica se o pool do lançador de unidades de execução deve ser ativado no JICS.

Tipo: booliano

Padrão: False

sucos. runUnitLauncherTamanho da piscina

Opcional. O tamanho do pool do lançador da unidade de execução no JICS.

Tipo: número

Padrão: 20

sucos. runUnitLauncherPool. Intervalo de validação

Opcional: O intervalo de validação do pool do lançador de unidades de execução no JICS, expresso em milissegundos.

Tipo: número

Padrão: 1000

jics.queues.sqs.region

Opcional. O Região da AWS para Amazon SQS, usado no JICS. É recomendável definir a mesma região da aplicação implantada para fins de desempenho, mas isso não é obrigatório.

Tipo: string

Padrão: eu-west-1

jics.xa.agent.timeout

Opcional. Define a duração máxima para que o agente xa responsável pelo gerenciamento de transações distribuídas conclua suas operações.

Tipo: número

Padrão: nulo

mq.queues.sqs.region

Opcional. O Região da AWS para o serviço Amazon SQS MQ.

Tipo: string

Padrão: eu-west-3

Executor de tarefas. allowCoreThreadTimeOut

Opcional. Especifica se os threads principais devem atingir o tempo limite no JCIS. Isso permite o crescimento e a redução dinâmicos, mesmo em combinação com uma fila diferente de zero (já que o tamanho máximo do pool só aumentará quando a fila estiver cheia).

Tipo: booliano

Padrão: False

Executor de tarefas. corePoolSize

Opcional. Quando uma transação em um terminal é iniciada por meio de um script groovy, um thread é criado. Use esse parâmetro para configurar o tamanho do pool principal.

Tipo: número

Padrão: 5

Executor de tarefas. maxPoolSize

Opcional. Quando uma transação em um terminal é iniciada por meio de um script groovy, um novo tópico é criado. Use esse parâmetro para configurar o tamanho máximo do grupo (número máximo de threads paralelos).

Tipo: número

Padrão: 10

taskExecutor.queueCapacity

Opcional. Quando uma transação em um terminal é iniciada por meio de um script groovy, um novo tópico é criado. Use esse parâmetro para configurar o tamanho da fila. (= número máximo de transações pendentes quando `taskExecutor.maxPoolSize` atingido)

Tipo: número

Padrão: 50

Propriedades de runtime do Gapwalk

cacheMetadata

Opcional. Especifica se os metadados do banco de dados devem ser armazenados em cache.

Tipo: booliano

Padrão: True

check-groovy-file

Opcional. Especifica se o conteúdo dos arquivos groovy deve ser verificado antes do registro.

Tipo: booliano

Padrão: True

databaseStatistics

Opcional. Especifica se devem permitir que os construtores de SQL coletem e exibam informações estatísticas.

Tipo: booliano

Padrão: False

dateTimeFormat

Opcional. dateTimeFormat Descreve como inserir o tipo de data e hora do banco de dados em entidades simplificadoras de dados. Os valores permitidos são: `ISO`, `EUR`, `USA` e `LOCAL`

Tipo: string

Padrão: ISO

dbDateFormat

Opcional. O formato da data alvo do banco de dados.

Tipo: string

Padrão: aaaa-MM-dd

dbTimeFormat

Opcional. O formato da hora alvo do banco de dados.

Tipo: string

Padrão: HH:mm:ss

dbTimestampFormat

Opcional. O formato do timestamp de destino do banco de dados.

Tipo: string

Padrão: aaaa-MM-dd HH:mm:ss.SSSSSS

fetchSize

Opcional. O valor fetchSize para cursores. Use ao buscar dados usando fragmentos por meio de utilitários de carregamento/descarregamento.

Tipo: número

Padrão: 10

Forçar a desativação do SQL TrimStringType

Opcional. Especifica se o corte de todos os parâmetros da string sql deve ser desativado.

Tipo: booliano

Padrão: False

localDateFormat

Opcional. Lista de formatos de data locais. Separe cada formato com |.

Tipo: string

localTimeFormat

Opcional. Lista de formatos de horário local. Separe cada formato com |.

Tipo: string

localTimestampFormat

Opcional. Lista de formatos de carimbo de data/hora locais. Separe cada formato com |.

Tipo: string

Padrão:

pgmDateFormat

Opcional. O formato de data e hora usado nos programas.

Tipo: string

Padrão: aaaa-MM-dd

pgmTimeFormat

Opcional. O formato de hora usado para execução de pgm (programas).

Tipo: string

Padrão: HH.mm.ss

pgmTimestampFormat

Opcional. O formato do carimbo de data e hora.

Tipo: string

Padrão: aaaa-MM-dd-HH.mm.ss.SSSSSS

Propriedades do programa utilitário Gapwalk

jcl.type

Opcional. Tipo de arquivo `.jcl`. Os valores permitidos são: `jcl` e `vse`. Os comandos PRINT/REPRO do utilitário IDCAMS retornam 4 se o arquivo estiver vazio para um jcl que não seja vse.

Tipo: string

Padrão: mvs

listcat.variablelengthpreprocessor.enabled

Opcional. Especifica se deseja ativar o pré-processador de comprimento variável para o comando LISTCAT.

Tipo: booliano

Padrão: False

listcat.variablelengthpreprocessor.type

Opcional. O tipo de objetos contidos no arquivo listcat, se você habilitar `listcat.variablelengthpreprocessor.enabled`. Os valores permitidos são: `rdw` e `bdw`.

Tipo: string

Padrão: rdw

load.batchSize

Opcional. O tamanho do lote do utilitário de carga.

Tipo: número

Padrão: 0

load.format.dbDate

Opcional. O formato do banco de dados do utilitário de carga a ser usado.

Tipo: string

Padrão: aaaa-MM-dd

load.format.dbTime

Opcional. O tempo de uso do banco de dados do utilitário de carregamento.

Tipo: string

Padrão: HH:mm:ss

load.format.localDate

Opcional. O formato de data local do utilitário de carregamento a ser usado.

Tipo: string

Padrão: dd.MM.aaaa|dd/MM/aaaa|aaaa-MM-dd

load.format.localTime

Opcional. O formato de hora local do utilitário de carregamento a ser usado.

Tipo: string

Padrão: HH:mm:ss|HH.mm.ss

carregar. sqlCodePointTurno

Opcional. A mudança de pontos do código SQL para o utilitário de carregamento. Executa o processo de mudança de caracteres. Obrigatório quando seu banco de dados de destino do DB2 é o Postgresql.

Tipo: número

Padrão: 0

sysPunchEncoding

Opcional. O conjunto de caracteres de codificação syspunch. Os valores suportados são `Cp1047` / `ASCII`.

Tipo: string

Padrão: ASCII

treatLargeNumberAsInteger

Opcional. Especifica se os números grandes devem ser tratados como `Integer`. Eles são tratados como `BigDecimal` padrão.

Tipo: booliano

Padrão: False

unload.chunkSize

Opcional. Tamanho do pedaço usado para o utilitário de descarga.

Tipo: número

Padrão: 0

unload.columnFiller

Opcional. O preenchedor de colunas do utilitário de descarga.

Tipo: string

Padrão: espaço

unload.fetchSize

Opcional. Permite ajustar o tamanho da busca ao manipular cursores no utilitário de descarga.

Tipo: número

Padrão: 0

unload.format.date

Opcional. Se `unload.useDatabaseConfiguration` estiver ativado, o formato de data a ser usado no utilitário de descarga. Para obter mais informações, consulte unload.format.date.

Tipo: string

Padrão: MM/dd/aaaa

unload.format.time

Opcional. Se `unload.useDatabaseConfiguration` estiver ativado, o formato de hora a ser usado no utilitário de descarga.

Tipo: string

Padrão: HH.mm.ss

unload.format.timestamp

Opcional. Se `unload.useDatabaseConfiguration` estiver ativado, o formato de carimbo de data/hora a ser usado no utilitário de descarga.

Tipo: string

Padrão: aaaa-MM-dd-HH.mm.ss.SSSSSS

descarregue.nbi. whenNotNull

Opcional. O valor do Indicador de Byte Nulo (nbi) a ser adicionado quando o valor do banco de dados não for nulo.

Tipo: hexadecimal

Padrão: 00

unload.nbi.whenNull

Opcional. O valor do Indicador de Byte Nulo (nbi) a ser adicionado quando o valor do banco de dados for nulo.

Tipo: hexadecimal

Padrão: 6F

descarregue.nbi. writeNullIndicator

Opcional. Especifica se o indicador nulo deve ser gravado no arquivo de saída de descarga.

Tipo: booliano

Padrão: False

descarregar. sqlCodePointTurno

Opcional. O utilitário de mudança de pontos do código SQL para descarga. Executa o processo de mudança de caracteres. Obrigatório quando seu banco de dados de destino do DB2 é o Postgresql.

Tipo: número

Padrão: 0

descarregar. useDatabaseConfiguration

Opcional. Especifica se a configuração de data ou hora do application-main.yml deve ser usada no utilitário de descarregamento.

Tipo: booliano

Padrão: False

descarregar. varCharIsNulo

Opcional. Use esse parâmetro no programa INFTILB, se definido como `true`, todos os campos não anuláveis com valores em branco (espaço) retornarão uma string vazia.

Tipo: booliano

Padrão: False

Outras propriedades

qtemp.cleanup.threshold.hours

Opcional. Para especificar quando `qtemp.dblog` está ativado. A vida útil da partição db (em horas).

Tipo: número

Padrão: 0

qtemp.dblog

Opcional. Se deve habilitar o log do banco de dados QTEMP.

Tipo: booliano

Padrão: False

qtemp.uuid.length

Opcional. O comprimento de identificação exclusivo do QTEMP.

Tipo: número

Padrão: 9

quartz.scheduler. stand-by-if-error

Opcional. Especifica se a execução do trabalho deve ser acionado se o agendador de trabalhos estiver no modo de espera. Se verdadeiro, quando ativada, a execução do trabalho não é acionada.

Tipo: booliano

Padrão: False

warmUpCache

Opcional. Especifica se todos os dados da tabela de comunicação de dados devem ser carregados em um cache de aquecimento na inicialização do servidor.

Tipo: booliano

Padrão: False

AWS Referência de definição de aplicativo de modernização de mainframe

Na Modernização do AWS Mainframe, você configura os aplicativos de mainframe migrados em um arquivo JSON de definição de aplicativo, que é específico para o mecanismo de tempo de execução escolhido. Uma definição de aplicação contém informações gerais e informações específicas do mecanismo. Este tópico descreve as definições dos aplicativos AWS Blu Age e Micro Focus e identifica todos os elementos obrigatórios e opcionais.

Sumário

- Configurações de lote: obrigatórias
- Configurações do CICS: obrigatórias
- Recursos XA: obrigatórios
- Configurações de tempo de execução - opcional

Seção de cabeçalho geral

Cada definição de aplicação começa com informações gerais sobre a versão do modelo e os locais de origem. A versão atual da definição da aplicação é 2.0.

Use a estrutura a seguir para especificar a versão do modelo e os locais de origem.

```
"template-version": "2.0",
    "source-locations": [
        {
            "source-id": "s3-source",
            "source-type": "s3",
            "properties": {
                "s3-bucket": "mainframe-deployment-bucket",
                "s3-key-prefix": "v1"
            }
        }
    ]
```

Note

É possível usar a seguinte sintaxe se você quiser inserir o ARN do S3 como s3-bucket:

```
"template-version": "2.0",
  "source-locations": [
      {
          "source-id": "s3-source",
          "source-type": "s3",
          "properties": {
              "s3-bucket": "arn:aws:s3:::mainframe-deployment-bucket",
              "s3-key-prefix": "v1"
          }
      }
  ]
```

Versão do modelo

Obrigatório. Especifica a versão do arquivo de definição da aplicação. Não mude esse valor. Atualmente, o único valor permitido é 2,0. Especifique `template-version` com uma string.

locais de origem

Especifica os locais dos arquivos e outros recursos que a aplicação exige durante o runtime.

ID de origem

Especifica um nome para o local. Esse nome é usado para referenciar o local de origem conforme necessário na definição do aplicativo JSON.

tipo de fonte

Especifica o tipo da fonte. Atualmente, o único valor permitido é s3.

propriedades

Fornece os detalhes do local de origem. Cada propriedade é especificada com uma string.

- `s3-bucket`: obrigatório. Especifica o nome do bucket do Amazon S3 onde os arquivos estão armazenados.
- `s3-key-prefix`: obrigatório. Especifica o nome da pasta no bucket do Amazon S3 em que os arquivos são armazenados.

Visão geral da seção de definição

Especifica as definições de recursos dos serviços, configurações, dados e outros recursos típicos de que a aplicação precisa para ser executada. Quando você atualiza uma definição de aplicação, o AWS Mainframe Modernization detecta alterações comparando as listas `source-locations` e `definition` das versões anteriores e atuais do arquivo JSON de definição de aplicação.

A seção de definição é específica do mecanismo e está sujeita a alterações. As seções a seguir mostram exemplos de definições de aplicações específicas do mecanismo para ambos os mecanismos.

AWS Exemplo de definição do aplicativo Blu Age

```
{
 "template-version": "2.0",
```

```
 "source-locations": [
        {
            "source-id": "s3-source",
            "source-type": "s3",
            "properties": {
                "s3-bucket": "mainframe-deployment-bucket-aaa",
                "s3-key-prefix": "v1"
            }
        }
    ],
    "definition" : {
        "listeners": [{
            "port": 8194,
            "type": "http"
        }],
        "ba-application": {
            "app-location": "${s3-source}/murachs-v6/"
        },
        "blusam": {
            "db": {
                "nb-threads": 8,
                "batch-size": 10000,
                "name": "blusam",
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:blusam-FfmXLG"
            },
            "redis": {
                "hostname": "blusam.c3geul.ng.0001.usw2.cache.amazonaws.com",
                "port": 6379,
                "useSsl": true,
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:bluesamredis-nioefm"
            }
        }
    }
}
```

AWS Detalhes da definição de Blu Age

Receptores: obrigatório

Especifique a porta que você usará para acessar o aplicativo por meio do Elastic Load Balancing criado pela AWS Mainframe Modernization. Use a seguinte estrutura:

```
"listeners": [{
            "port": 8194,
            "type": "http"
}],
```

porta

Obrigatório. Você pode usar qualquer porta disponível, exceto as conhecidas portas de 0 a 1023. Recomendamos usar o intervalo de 8192 a 8199. Verifique se não há outros receptores ou aplicações operando nessa porta.

tipo

Obrigatório. No momento, só há compatibilidade com `http`.

AWS Aplicativo Blu Age - obrigatório

Especifique o local em que o mecanismo coleta o arquivo de imagem da aplicação usando a estrutura a seguir.

```
"ba-application": {
            "app-location": "${s3-source}/murachs-v6/",
            "files-directory": "/m2/mount/myfolder",
            "enable-jics": <true|false>,
            "shared-app-location": "${s3-source}/shared/"
},
```

app-location

O local específico no Amazon S3 em que o arquivo de imagem da aplicação é armazenado.

files-directory

Opcional. A localização dos arquivos de entrada/saída para lotes. Deve ser uma subpasta da configuração do ponto de montagem do Amazon EFS ou do Amazon FSx no nível do ambiente. A subpasta deve pertencer a um usuário adequado para ser usada pelo aplicativo Blu Age executado na modernização do AWS mainframe. Para conseguir isso, ao conectar a unidade a uma instância Linux do Amazon EC2, um grupo com `101` ID e um usuário com `3001` ID devem ser criados, e a pasta desejada deve pertencer a esse usuário. Por exemplo, dessa forma, a *`testclient`* pasta pode ser usada pelo Blu Age AWS Mainframe Modernization Managed.

```
groupadd -g 101 mygroup
useradd -M -g mygroup -p mypassword -u 3001 myuser
mkdir testclient
chown myuser:mygroup testclient
```

enable-jics

Opcional. Especifica se o JICS deve ser ativado. O valor padrão é verdadeiro. Definir isso como false impede que o banco de dados JICS seja gerado.

shared-app-location

Opcional. Localização adicional no Amazon S3 em que os elementos da aplicação compartilhados são armazenados. Ele pode conter o mesmo tipo de estrutura de aplicação que app-location.

BluSAM: opcional

Especifique o banco de dados BluSAM e o cache do Redis usando a estrutura a seguir.

```
"blusam": {
            "db": {
                "nb-threads": 8,
                "batch-size": 10000,
                "name": "blusam",
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:blusam-FfmXLG"
            },
            "redis": {
                "hostname": "blusam.c3geul.ng.0001.usw2.cache.amazonaws.com",
                "port": 6379,
                "useSsl": true,
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:bluesamredis-nioefm"
            }
}
```

db

Especifica as propriedades do banco de dados usado com a aplicação. O banco de dados deve ser um banco de dados do Aurora PostgreSQL. Você pode especificar as seguintes propriedades:

- `nb-threads`: opcional. Especifica quantos encadeamentos dedicados são usados para o mecanismo de gravação no qual o mecanismo BluSam depende. O padrão é 8.
- `batch-size`: opcional. Especifica o limite que o mecanismo write-behind usa para iniciar as operações de armazenamento em lote. O limite representa o número de registros modificados que iniciarão uma operação de armazenamento em lote para garantir que os registros modificados persistam. O gatilho em si é baseado em uma combinação do tamanho do lote e do tempo decorrido de um segundo, o que for atingido primeiro. O padrão é 10000.
- `name`: opcional. Especifica o nome do banco de dados.
- `secret-manager-arn`: especifica o nome do recurso da Amazon (ARN) do segredo do que contém as credenciais do banco de dados. Para ter mais informações, consulte Etapa 4: criar e configurar um segredo de AWS Secrets Manager banco de dados.

Redis

Especifica as propriedades do cache do Redis que a aplicação usa para armazenar os dados temporários necessários em um local central para melhorar o desempenho. Recomendamos que você criptografe e proteja com senha o cache do Redis.

- `hostname`: especifica a localização do cache do Redis.
- `port`: especifica a porta, normalmente 6379, pela qual o cache do Redis envia e recebe comunicação.
- `useSsl`: especifica se o cache do Redis está criptografado. Se o cache não estiver criptografado, defina `useSsl` como false.
- `secret-manager-arn`: especifica o nome do recurso da Amazon (ARN) do segredo do que contém a senha de cache do Redis. Se o cache do Redis não estiver protegido por senha, não especifique `secret-manager-arn`. Para ter mais informações, consulte Etapa 4: criar e configurar um segredo de AWS Secrets Manager banco de dados.

AWS Filas de mensagens do Blu Age - opcionais

Especifique os detalhes da conexão JMS-MQ para AWS o aplicativo Blu Age.

```
"message-queues": [
  {
      "product-type": "JMS-MQ",
      "queue-manager": "QMgr1",
      "channel": "mqChannel1",
      "hostname": "mqserver-host1",
```

```
      "port": 1414,
      "user-id": "app-user1",
      "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:sample/mq/test-279PTa"
  },
  {
      "product-type": "JMS-MQ",
      "queue-manager": "QMgr2",
      "channel": "mqChannel2",
      "hostname": "mqserver-host2",
      "port": 1412,
      "user-id": "app-user2",
      "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:sample/mq/test-279PTa"
  }
]
```

product-type

Obrigatório. Especifica o tipo de produto. Atualmente, isso só pode ser “JMS-MQ” para AWS aplicativos Blu Age.

queue-manager

Obrigatório. Especifica o nome do gerenciador de filas.

channel

Obrigatório. Especifica o nome do canal de conexão do servidor.

hostname

Obrigatório. Especifica o nome do host do servidor da fila de mensagens.

porta

Obrigatório. Especifica o número da porta do receptor em que o servidor está escutando.

user-id

Opcional. Especifica a ID da conta de usuário permitida para realizar operações de fila de mensagens no canal especificado.

secret-manager-arn

Opcional. Especifica o nome do recurso da Amazon (ARN) do Secrets Manager que fornece a senha do usuário especificado.

AWS Configuração EFS de armazenamento de aplicativos Blu Age - opcional

Especifique os detalhes do ponto de acesso EFS do armazenamento de aplicativos usando a estrutura a seguir.

```
"ba-application": {
            "file-permission-mask": "UMASK002"
},
"efs-configs": [
           {
            "file-system-id": "fs-01376dfsvfvrsvsr",
            "mount-point": "/m2/mount/efs-ap2",
            "access-point-id": fsap-0eaesefvrefrewgv8"
           }
]
```

file-system-id

Obrigatório. A ID do sistema de arquivos EFS ao qual o ponto de acesso se aplica. Padrão: “fs-([0-9a-f] {8,40}) {1,128} $”

ponto de montagem

Obrigatório. O ponto de montagem do sistema de arquivos no nível do aplicativo. Isso deve ser diferente do ponto de montagem do armazenamento no nível do ambiente.

access-point-id

Obrigatório. O ID do ponto de acesso, atribuído pelo Amazon EFS. Padrão: “^fsap- ([0-9a-f] {8,40}) {1,128} $”

file-permission-mask

Opcional. Define a máscara de criação de arquivos para arquivos criados pelo processo do aplicativo. Por exemplo, quando o valor for definido como`UMASK006`, todos os arquivos terão a permissão 660. Isso significa que somente o proprietário do arquivo e o grupo de arquivos terão acesso de leitura e gravação, enquanto outros usuários não terão nenhuma permissão.

> **Note**
>
> O valor definido para esse campo só é considerado ao usar o armazenamento EFS no nível do aplicativo.

> **Note**
>
> Quando a configuração efs é fornecida, o diretório de arquivos deve ser especificado na seção de definição do aplicativo. Ela deve ser uma subpasta do ponto de montagem do Amazon EFS configurado no nível do aplicativo.

Definição de aplicação da Micro Focus

A seção de definição de exemplo a seguir é para o mecanismo de runtime da Micro Focus e contém elementos obrigatórios e opcionais.

```
{
 "template-version": "2.0",
 "source-locations": [
        {
            "source-id": "s3-source",
            "source-type": "s3",
            "properties": {
                "s3-bucket": "mainframe-deployment-bucket-aaa",
                "s3-key-prefix": "v1"
            }
        }
    ],
    "definition" : {
        "listeners": [{
            "port": 5101,
            "type": "tn3270"
        }],
        "dataset-location": {
            "db-locations": [{
                "name": "Database1",
                "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456"
            }]
        },
        "cognito-auth-handler": {
            "user-pool-id": "cognito-idp.us-west-2.amazonaws.com/us-west-2_rvYFnQIxL",
            "client-id": "58k05jb8grukjjsudm5hhn1v87",
            "identity-pool-id": "us-west-2:64464b12-0bfb-4dea-ab35-5c22c6c245f6"
        },
        "ldap-ad-auth-handler": {
            "ldap-ad-connection-secrets": [LIST OF AD-SECRETS]
```

```
        },
        "batch-settings": {
            "initiators": [{
                "classes": ["A", "B"],
                "description": "initiator...."
            }],
            "jcl-file-location": "${s3-source}/batch/jcl",
            "program-path": "/m2/mount/libs/loadlib:$EFS_MOUNT/emergency/loadlib",
            "system-procedure-libraries":"SYS1.PROCLIB;SYS2.PROCLIB",
            "aliases": [
               {"alias": "FDSSORT", "program": "SORT"},
               {"alias": "MFADRDSU", "program": "ADRDSSU"}
            ]
        },
        "cics-settings": {
            "binary-file-location": "${s3-source}/cics/binaries",
            "csd-file-location": "${s3-source}/cics/def",
            "system-initialization-table": "BNKCICV"
        },
        "xa-resources" : [{
            "name": "XASQL",
            "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456",
            "module": "${s3-source}/xa/ESPGSQLXA64.so"
        }],
        "runtime-settings": {
          "environment-variables": {
            "ES_JES_RESTART": "N",
            "EFS_MOUNT": "/m2/mount/efs"
          }
        }
    }
}
```

Detalhes da definição da Micro Focus

O conteúdo na seção de definição do arquivo de definição de aplicação da Micro Focus varia, dependendo dos recursos que sua aplicação de mainframe migrada exige em runtime.

Receptor(es): obrigatório

Especifique um receptor usando a seguinte estrutura:

```
"listeners": [{
```

```
    "port": 5101,
    "type": "tn3270"
}],
```

porta

Para tn3270, o padrão é 5101. Para outros tipos de receptores de serviço, a porta varia. Você pode usar qualquer porta disponível, exceto as conhecidas portas de 0 a 1023. Cada receptor deve ter uma porta distinta. Os receptores não devem compartilhar portas. Para obter mais informações, consulte Listener Control na documentação do Micro Focus Enterprise Server.

tipo

Especifica o tipo de receptor de serviço. Para obter mais informações, consulte Listeners na documentação do Micro Focus Enterprise Server.

Localizações do conjunto de dados: obrigatório

Especifique a localização do conjunto de dados usando a estrutura a seguir.

```
"dataset-location": {
            "db-locations": [{
                "name": "Database1",
                "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456"
            }],
 }
```

db-locations

Especifica a localização dos conjuntos de dados que a aplicação migrada cria. Atualmente, a modernização do AWS mainframe suporta somente conjuntos de dados de um único banco de dados VSAM.

- `name`: especifica o nome da instância do banco de dados que contém os conjuntos de dados criados pela aplicação migrada.
- `secret-manager-arn`: especifica o nome do recurso da Amazon (ARN) do segredo do que contém as credenciais do banco de dados.

Manipulador de autenticação e autorização do Amazon Cognito: opcional

AWS A modernização do mainframe usa o Amazon Cognito para autenticação e autorização de aplicativos migrados. Especifique o manipulador de autenticação do Amazon Cognito usando a estrutura a seguir.

```
"cognito-auth-handler": {
            "user-pool-id": "cognito-idp.Region.amazonaws.com/Region_rvYFnQIxL",
            "client-id": "58k05jb8grukjjsudm5hhn1v87",
            "identity-pool-id": "Region:64464b12-0bfb-4dea-ab35-5c22c6c245f6"
}
```

user-pool-id

Especifica o grupo de usuários do Amazon Cognito AWS que o Mainframe Modernization usa para autenticar os usuários do aplicativo migrado. O Região da AWS para o grupo de usuários deve corresponder ao do Região da AWS aplicativo de modernização do AWS mainframe.

client-id

Especifica a aplicação migrada que o usuário autenticado pode acessar.

identity-pool-id

Especifica o pool de identidade do Amazon Cognito em que o usuário autenticado troca um token do grupo de usuários por credenciais que permitem que o usuário acesse a modernização do mainframe. AWS O Região da AWS for do pool de identidades deve corresponder ao do Região da AWS aplicativo de modernização do AWS mainframe.

Manipulador do LDAP e do Active Directory: opcional

É possível integrar sua aplicação ao Active Directory (AD) ou a qualquer tipo de servidor LDAP para possibilitar que os usuários da aplicação usem suas credenciais do LDAP/AD para autorização e autenticação.

Como integrar a aplicação ao AD

1. Siga as etapas descritas em Configurar o Active Directory para a segurança do Enterprise Server na documentação do Micro Focus Enterprise Server.
2. Crie um AWS Secrets Manager segredo com os detalhes do AD/LDAP para cada servidor AD/LDAP que você deseja usar com seu aplicativo. Para obter informações sobre como criar um

segredo, consulte Criar um segredo do AWS Secrets Manager no Guia AWS Secrets Manager do usuário. Para o tipo de segredo, escolha Outro tipo de segredo e inclua os seguintes pares de chave/valor.

```
{
    "connectionPath"       : "<HOST-ADDRESS>:<PORT>",
    "authorizedId"         : "<USER-FULL-DN>",
    "password"             : "<PASSWORD>",
    "baseDn"               : "<BASE-FULL-DN>",
    "userClassDn"          : "<USER-TYPE>",
    "userContainerDn"      : "<USER-CONTAINER-DN>",
    "groupContainerDn"     : "<GROUP-CONTAINER-DN>",
    "resourceContainerDn"  : "<RESOURCE-CONTAINER-DN>"
}
```

⚠ Recomendações de segurança

- Pois`connectionPath`, a modernização do AWS mainframe oferece suporte aos protocolos LDAP e LDAP sobre SSL (LDAPS). Recomendamos usar o LDAPS porque é mais seguro e evita que as credenciais apareçam nas transmissões da rede.
- Para `authorizedId` e `password`, recomendamos que você especifique as credenciais de um usuário sem mais permissões do que as permissões mais restritivas de somente leitura e verificação necessárias para que a aplicação seja executada.
- Recomendamos fazer a alternância das credenciais do AD/LDAP regularmente.
- Não crie usuários do AD com o nome de usuário `awsuser` ou `mfuser`. Esses dois nomes de usuário são reservados para uso da AWS .

Veja um exemplo a seguir.

```
{
    "connectionPath" : "ldaps://msad4.m2.example.people.aws.dev:636",
    "authorizedId" :
 "CN=LDAPUser,OU=Users,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev",
    "password" : "ADPassword",
    "userContainerDn" : "CN=Enterprise Server Users,CN=Micro Focus,CN=Program
 Data,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev",
```

```
    "groupContainerDn" : "CN=Enterprise Server Groups,CN=Micro Focus,CN=Program
 Data,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev",
    "resourceContainerDn" : "CN=Enterprise Server Resources,CN=Micro
 Focus,CN=Program Data,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev"
}
```

Crie o segredo com uma chave do KMS gerenciada pelo cliente. Você deve conceder à Modernização do AWS Mainframe `DescribeSecret` as permissões `GetSecretValue` e sobre o segredo `Decrypt` e `DescribeKey` as permissões sobre a chave KMS. Para obter mais informações, consulte Permissões para a chave KMS no Guia do AWS Secrets Manager usuário.

3. Adicione o seguinte à definição da aplicação:

```
"ldap-ad-auth-handler": {
    "ldap-ad-connection-secrets": [LIST OF AD/LDAP SECRETS]
}
```

Veja um exemplo a seguir.

```
"ldap-ad-auth-handler": {
    "ldap-ad-connection-secrets": ["arn:aws:secrets:1234:us-east-1:secret:123456"]
}
```

O manipulador de autenticação do LDAP/AD está disponível para o Micro Focus 8.0.11 e versões posteriores.

Configurações de lote: obrigatórias

Especifique os detalhes exigidos pelos trabalhos em lotes que são executados como parte da aplicação usando a estrutura a seguir.

```
"batch-settings": {
            "initiators": [{
                "classes": ["A", "B"],
                "description": "initiator...."
            }],
            "jcl-file-location": "${s3-source}/batch/jcl",
            "program-path": "/m2/mount/libs/loadlib:$EFS_MOUNT/emergency/loadlib",
            "system-procedure-libraries":"SYS1.PROCLIB;SYS2.PROCLIB",
```

```
            "aliases": [
               {"alias": "FDSSORT", "program": "SORT"},
               {"alias": "MFADRDSU", "program": "ADRDSSU"}
    ]
}
```

initiators

Especifica um iniciador de lote que é iniciado quando a aplicação migrada é iniciada com êxito e continua em execução até que seja interrompida. Você pode definir uma ou várias classes por iniciador. Você também pode definir vários iniciadores. Por exemplo: .

```
"batch-settings": {
            "initiators": [
                {
                  "classes": ["A", "B"],
                  "description": "initiator...."
                },
                {
                  "classes": ["C", "D"],
                  "description": "initiator...."
                }

        ],
 }
```

Para obter mais informações, consulte Para definir um iniciador de lote ou uma impressora SEP na documentação do Micro Focus Enterprise Server.

- `classes`: especifica as classes de trabalho que o iniciador pode executar. Você pode usar até 36 caracteres. Você pode usar os seguintes caracteres: A–Z ou 0–9.
- `description`: descreve para que serve o iniciador.

jcl-file-location

Especifica a localização dos arquivos JCL (Job Control Language) que são exigidos pelos trabalhos em lotes que o aplicativo migrado executa.

caminho do programa

Especifica o caminho necessário para executar trabalhos em lotes quando um programa em uma JCL não está no local padrão. Os diferentes nomes dos caminhos são separados por dois pontos (:).

> **Note**
>
> O caminho do programa só pode ser um caminho do EFS.

system-procedure-libraries

Especifica os conjuntos de dados particionados padrão que serão pesquisados em busca de procedimentos JCL. No entanto, o procedimento não é encontrado no JCL ou por meio das instruções JCLLIB. Esses conjuntos de dados devem ser catalogados e o nome do catálogo deve ser usado. E as entradas são separadas por ponto e vírgula (;).

aliases

Define um mapeamento dos nomes de utilitários e programas usados no JCL para o nome de implementação do utilitário. AWS e utilitários de lote de terceiros (por exemplo, M2SFTP, M2WAIT, Syncsort etc.) podem opcionalmente ter aliases para eliminar a necessidade de alterar o JCL. Por exemplo: .

- Alias FDSSORT FDSSORT para SORT e Alias FDSICET para ICETOOL
- Apelido ADRDSSU MFADRDSU para ADRDSSU
- Alias de sincronização DMXMFSRT para SORT

Configurações do CICS: obrigatórias

Especifique os detalhes necessários para as transações do CICS que são executadas como parte da aplicação usando a estrutura a seguir.

```
"cics-settings": {
            "binary-file-location": "${s3-source}/cics/binaries",
            "csd-file-location": "${s3-source}/cics/def",
            "system-initialization-table": "BNKCICV"
}
```

binary-file-location

Especifica o local dos arquivos do programa de transação do CICS.

csd-file-location

Especifica a localização do arquivo de definição de recursos (CSD) do CICS para essa aplicação. Para obter mais informações, consulte Definições de recursos do CICS na documentação do Micro Focus Enterprise Server.

system-initialization-table

Especifica a tabela de inicialização do sistema (SIT) que a aplicação migrada usa. O nome da tabela SIT pode ter até 8 caracteres. Você pode usar A–Z, 0–9, $, @ e #. Para obter mais informações, consulte Definições de recursos do CICS na documentação do Micro Focus Enterprise Server.

Recursos XA: obrigatórios

Especifique os detalhes necessários para os recursos XA que a aplicação exige usando a estrutura a seguir.

```
"xa-resources" : [{
            "name": "XASQL",
            "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456",
            "module": "${s3-source}/xa/ESPGSQLXA64.so"
}]
```

name

Obrigatório. Especifica o nome do recurso XA.

secret-manager-arn

Especifica o nome do recurso da Amazon (ARN) para o segredo do que contém as credenciais para se conectar ao banco de dados.

module

Especifica a localização do arquivo executável do módulo de switch RM. Para obter mais informações, consulte Planejando e projetando XARs na documentação do Micro Focus Enterprise Server.

Configurações de tempo de execução - opcional

Especifique os detalhes necessários para que as configurações de tempo de execução gerenciem as variáveis de ambiente permitidas usando a estrutura a seguir.

```
"runtime-settings": {
          "environment-variables": {
            "ES_JES_RESTART": "N",
            "EFS_MOUNT": "/m2/mount/efs"
    }
}
```

variáveis de ambiente

Especifica as variáveis de ambiente suportadas pela Micro Focus que são aplicadas ao tempo de execução desse aplicativo.

- `ES_JES_RESTART`é uma variável de ambiente da Micro Focus que permite que o JCL reinicie o processamento. Opcionalmente, você também pode usar `ES_ALLOC_OVERRIDE` como variável de ambiente da Micro Focus.
- `EFS_MOUNT`é uma variável de ambiente personalizada que seu aplicativo pode usar para identificar onde a montagem EFS do ambiente está localizada.

Você pode acessar todas as variáveis de ambiente da Micro Focus no guia Micro Focus Enterprise Server for UNIX.

AWS Referência de definição do conjunto de dados de modernização de mainframe

Se seu aplicativo exigir mais do que alguns conjuntos de dados para processamento, inseri-los um por um no console de modernização do AWS mainframe é ineficiente. Em vez disso, recomendamos que crie um arquivo JSON para especificar cada conjunto de dados. Diferentes tipos de conjuntos de dados são especificados de forma diferente no JSON, embora muitos parâmetros sejam comuns. Este documento descreve os detalhes do JSON necessários para importar diferentes tipos de conjuntos de dados.

> **Note**
>
> Antes de importar qualquer conjunto de dados, você deve transferir os conjuntos de dados do mainframe para o. AWS Em seguida, você deve garantir que os conjuntos de dados sejam convertidos do formato de mainframe para um formato que AWS possa ser usado. Se necessário, transforme os dados conforme necessário e armazene os conjuntos de dados transformados no Amazon S3. Especifique o nome do bucket e da pasta no arquivo JSON de definição do conjunto de dados.
>
> Se você estiver usando o mecanismo de runtime da Micro Focus, poderá usar o utilitário `DFCONV` para converter os conjuntos de dados. Incluímos esse utilitário em nossas imagens do Micro Focus Enterprise Developer e Enterprise Server. Para obter mais informações, consulte DFCONV Batch File Conversion na documentação do Micro Focus Enterprise Developer.

Tópicos

Propriedades gerais

Vários parâmetros são comuns a todos os conjuntos de dados. Esses parâmetros abrangem as seguintes áreas:

- Informações sobre o conjunto de dados (`datasetName`,`datasetOrg`,`recordLength`,`encoding`)
- Informações sobre o local de onde você está importando, ou seja, o local de origem do conjunto de dados. Esse não é o local no mainframe. É o caminho para o local do Amazon S3 no qual você fez o upload do conjunto de dados (`externalLocation`).
- Informações sobre o local para o qual você está importando, ou seja, o local de destino do conjunto de dados. Esse local é um banco de dados ou um sistema de arquivos, dependendo do seu mecanismo de runtime. (`storageType` e `relativePath`).

- Informações sobre o tipo de conjunto de dados (tipo específico de conjunto de dados, formato, codificação etc.).

Cada definição de conjunto de dados tem a mesma estrutura JSON. O exemplo de JSON a seguir mostra todos esses parâmetros comuns.

```
{
    "dataSet": {
        "storageType": "Database",
        "datasetName": "MFI01V.MFIDEMO.BNKACC",
        "relativePath": "DATA",
        "datasetOrg": {
            "type": {
                type-specific properties
                    ...
            },
        },
    },
}
```

As propriedades a seguir são comuns a todos os conjuntos de dados.

storageType

Obrigatório. Aplica-se ao local de destino. Especifica se o conjunto de dados é armazenado em um banco de dados ou em um sistema de arquivos. Os valores possíveis são `Database` ou `FileSystem`.

- AWS Mecanismo de tempo de execução Blu Age: sistemas de arquivos não são suportados. Você deve usar um banco de dados.
- Mecanismo de runtime da Micro Focus: bancos de dados e sistemas de arquivos são suportados. Você pode usar o Amazon Relational Database Service ou o Amazon Aurora para bancos de dados, e o Amazon Elastic File System ou o Amazon FSx para Lustre para sistemas de arquivos.

datasetName

Obrigatório. Especifica o nome totalmente qualificado do conjunto de dados conforme ele aparece no mainframe.

relativePath

Obrigatório. Aplica-se ao local de destino. Especifica a localização relativa do conjunto de dados no banco de dados ou no sistema de arquivos.

datasetOrg

Obrigatório. Especifica o tipo de conjunto de dados. Os valores possíveis são `vsam`, `gdg`, `ps`, `po` ou `unknown`.

- AWS Mecanismo de tempo de execução Blu Age: somente conjuntos de dados do tipo VSAM são suportados.
- Mecanismo de runtime da Micro Focus: conjuntos de dados do tipo VSAM, GDG, PS, PO ou Unknown são compatíveis.

> **Note**
>
> Se a sua aplicação exigir arquivos que não sejam arquivos de dados COBOL, mas sejam PDF ou outros arquivos binários, você poderá especificá-los da seguinte forma:
>
> ```
> "datasetOrg": {
>             "type": PS {
>                 "format": U
>             },
> ```

Formato de solicitação de conjunto de dados de amostra para VSAM

- AWS Mecanismo de tempo de execução Blu Age: suportado.
- Mecanismo de runtime Micro Focus: compatível.

Se você estiver importando conjuntos de dados do VSAM, especifique `vsam` como o `datasetOrg`. Seu JSON deve ser semelhante ao exemplo a seguir:

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.VSAM.KSDS",
    "relativePath": "DATA",
```

```
    "datasetOrg": {
        "vsam": {

            "encoding": "A",
            "format": "KS",
            "primaryKey": {
                "length": 11,
                "offset": 0
            }
        }
    },
    "recordLength": {
        "min": 300,
        "max": 300
    }
},
"externalLocation": {
    "s3Location": "s3://$M2_DATA_STORE/catalog/data/AWS.M2.VSAM.KSDS.DAT"
}
```

As propriedades a seguir são compatíveis com conjuntos de dados VSAM.

encoding

Obrigatório. Especifica a codificação do conjunto de caracteres do conjunto de dados. Os valores possíveis são ASCII (`A`), EBCDIC (`E`) e Unknown (`?`).

format

Obrigatório. Especifica o tipo do conjunto de dados VSAM e o formato do registro.

- AWS Mecanismo de tempo de execução Blu Age: os valores possíveis são ESDS (`ES`), KSDS (`KS`) e RRDS (). `RR` O formato do registro pode ser fixo ou variável.
- Mecanismo de runtime da Micro Focus: os valores possíveis são ESDS (`ES`), KSDS (`KS`) e RRDS (`RR`). A definição do VSAM inclui o formato do registro, portanto, você não precisa especificá-lo separadamente.

primaryKey

Aplica-se somente aos conjuntos de dados VSAM KSDS. Especifica a chave primária. Consiste no nome da chave primária, no deslocamento da chave e no comprimento da chave. O `name` é opcional; `offset` e `length` são obrigatórios.

recordLength

Obrigatório. Especifica o tamanho de um registro. Para formatos de registro de tamanho fixo, esses valores devem corresponder.

- AWS O mecanismo de tempo de execução Blu Age: para VSAM ESDS, KSDS e RRDS, `min` é opcional e obrigatório. `max`
- Mecanismo de runtime da Micro Focus: `min` e `max` são obrigatórios.

externalLocation

Obrigatório. Especifica o local de origem: ou seja, o bucket do Amazon S3 em que você fez o upload do conjunto de dados.

Propriedades específicas do mecanismo do Blu Age

O mecanismo de tempo de execução do AWS Blu Age suporta compactação para conjuntos de dados VSAM. O exemplo a seguir mostra como você pode especificar essa propriedade no JSON.

```
{
    common properties
        ...
        "datasetOrg": {
            "vsam": {
                common properties
                ...
                "compressed": boolean,
                common properties
                ...
            }
        }
}
```

Especifique a propriedade de compactação da seguinte forma:

compression

Opcional. Especifica se os índices desse conjunto de dados são armazenados como valores compactados. Se você tiver um grande conjunto de dados (normalmente > 100 Mb), considere definir esse sinalizador como`true`.

Formato de solicitação de conjunto de dados de amostra para o GDG Base

- AWS Mecanismo de tempo de execução Blu Age: não suportado.
- Mecanismo de runtime Micro Focus: compatível.

Se você estiver importando conjuntos de dados de base do GDG, especifique `gdg` como o `datasetOrg`. Seu JSON deve ser semelhante ao exemplo a seguir:

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.GDG",
    "relativePath": "DATA",
    "datasetOrg": {
        "gdg": {
            "limit": "3",
            "rollDisposition": "Scratch and No Empty"
        }
    }
}
```

As propriedades a seguir são compatíveis com conjuntos de dados básicos do GDG.

limite

Obrigatório. Especifica o número de gerações ativas ou vieses. Para um cluster base do GDG, o máximo é 255.

rollDisposition

Opcional. Especifica como lidar com conjuntos de dados de geração quando o máximo é atingido ou excedido. Os valores possíveis são `No Scratch and No Empty`, `Scratch and No Empty`, `Scratch and Empty`, ou `No Scratch and Empty`. O padrão é `Scratch and No Empty`.

Formato de solicitação de conjunto de dados de amostra para as gerações PS ou GDG

- AWS Mecanismo de tempo de execução Blu Age: não suportado.
- Mecanismo de runtime Micro Focus: compatível.

Se você estiver importando conjuntos de dados das gerações PS ou GDG, especifique `ps` como o `datasetOrg`. Seu JSON deve ser semelhante ao exemplo a seguir:

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.PS.FB",
    "relativePath": "DATA",
    "datasetOrg": {
        "ps": {
            "format": "FB",
            "encoding": "A"
        }
    },
    "recordLength": {
        "min": 300,
        "max": 300
    }
},
"externalLocation": {
    "s3Location": "s3://$M2_DATA_STORE/catalog/data/AWS.M2.PS.LSEQ"
}
}
```

As propriedades a seguir são compatíveis com conjuntos de dados das gerações PS ou GDG.

format

Obrigatório. Especifica o formato dos registros do conjunto de dados. Os valores possíveis são `F`, `FA`, `FB`, `FBA`, `FBM`, `FBS`, `FM`, `FS`, `LSEQ`, `U`, `V`, `VA`, `VB`, `VBA`, `VBM`, `VBS`, `VM` e `VS`.

encoding

Obrigatório. Especifica a codificação do conjunto de caracteres do conjunto de dados. Os valores possíveis são ASCII (`A`), EBCDIC (`E`) e Unknown (`?`)

recordLength

Obrigatório. Especifica o tamanho de um registro. Você deve especificar o tamanho mínimo (`min`) e máximo (`max`) do registro. Para formatos de registro de tamanho fixo, esses valores devem corresponder.

externalLocation

Obrigatório. Especifica o local de origem: ou seja, o bucket do Amazon S3 em que você fez o upload do conjunto de dados.

Formato de solicitação de conjunto de dados de amostra para PO

Se você estiver importando conjuntos de dados PO, especifique `po` como o `datasetOrg`. Seu JSON deve ser semelhante ao exemplo a seguir:

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.PO.PROC",
    "relativePath": "DATA",
    "datasetOrg": {
        "po": {
            "format": "LSEQ",
            "encoding": "A",
            "memberFileExtensions": ["PRC"]
        }
    },
    "recordLength": {
        "min": 80,
        "max": 80
    }
},
"externalLocation": {
    "s3Location": "s3://$M2_DATA_STORE/source/proc/"
}
}
```

As propriedades a seguir são suportadas para conjuntos de dados PO.

format

Obrigatório. Especifica o formato dos registros do conjunto de dados. Os valores possíveis são `F`, `FA`, `FB`, `FBA`, `FBM`, `FBS`, `FM`, `FS`, `LSEQ`, `U`, `V`, `VA`, `VB`, `VBA`, `VBM`, `VBS`, `VM` e `VS`.

encoding

Obrigatório. Especifica a codificação do conjunto de caracteres do conjunto de dados. Os valores possíveis são ASCII (`A`), EBCDIC (`E`) e Unknown (`?`).

memberFileExtensions

Obrigatório. Especifica uma matriz contendo uma ou mais extensões de nome de arquivo, permitindo que você especifique quais arquivos serão incluídos como membro do PDS.

recordLength

Opcional. Especifica o tamanho de um registro. Tanto o tamanho mínimo (`min`) quanto o máximo (`max`) do registro são opcionais. Para formatos de registro de tamanho fixo, esses valores devem corresponder.

externalLocation

Obrigatório. Especifica o local de origem: ou seja, o bucket do Amazon S3 em que você fez o upload do conjunto de dados.

> **Note**
>
> A implantação atual do mecanismo de runtime da Micro Focus adiciona entradas PDS como conjuntos de dados dinâmicos.

Ambientes de tempo de execução gerenciados na AWS modernização do mainframe

Se você é iniciante na modernização de AWS mainframe, consulte os seguintes tópicos para começar:

- O que é modernização AWS do mainframe?
- Configurar a modernização AWS do mainframe
- Começando com a modernização AWS do mainframe
- Tutorial: Configurar o tempo de execução gerenciado para o AWS Blu Age
- Tutorial: Configurar o tempo de execução gerenciado para a Micro Focus

Um ambiente de tempo de execução na modernização de AWS mainframe é uma combinação nomeada de recursos AWS computacionais, um mecanismo de tempo de execução e os detalhes de configuração que você especifica. O ambiente de runtime hospeda uma ou mais aplicações. Os aplicativos na modernização AWS do mainframe contêm cargas de trabalho migradas do mainframe. Você pode escolher o mecanismo de runtime para os ambientes que você cria. Escolha AWS Blu Age se estiver usando o padrão de refatoração automatizado e Micro Focus se estiver usando o padrão de replataforma. Você também pode escolher a quantidade de recursos computacionais adequados para seu aplicativo e, opcionalmente, associar armazenamento a ambientes de tempo de execução. AWS A modernização do mainframe permite que você monitore seu ambiente de tempo de execução com CloudWatch métricas e registros da Amazon.

Tópicos

- Crie um ambiente de tempo de execução de modernização de AWS mainframe
- Atualize um ambiente de AWS tempo de execução de modernização de mainframe
- Interrompa um ambiente de execução de modernização de AWS mainframe
- Reinicie um ambiente de AWS tempo de execução de modernização de mainframe
- Excluir um ambiente de AWS execução de modernização de mainframe

Crie um ambiente de tempo de execução de modernização de AWS mainframe

Use o console de modernização de AWS mainframe para criar um ambiente de modernização de AWS mainframe.

Estas instruções pressupõem que você tenha concluído as etapas em Configurar a modernização AWS do mainframe.

Criar um ambiente de runtime

Para criar um ambiente de runtime

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que você deseja criar o ambiente.
3. Na guia Ambientes escolha Criar ambiente.
4. Na página Especificações da inferência, forneça as seguintes informações:
    a. Na seção Nome e descrição, insira um nome para o ambiente.
    b. (Opcional) No campo Descrição do ambiente, insira uma descrição para o ambiente. Essa descrição pode ajudar você e outros usuários a identificar a finalidade do ambiente de runtime.
    c. Na seção Opções do mecanismo, escolha Blu Age para refatoração automatizada ou Micro Focus para redefinição da plataforma.
    d. Escolha uma versão para o mecanismo que você selecionou.
    e. (Opcional) Na seção Tags, escolha Adicionar nova tag para adicionar uma ou mais tags de ambiente ao seu ambiente. Uma tag de ambiente é um rótulo de atributo personalizado que ajuda a organizar e gerenciar seus AWS recursos.
    f. Escolha Próximo.
5. Na página Especificar configurações, forneça as seguintes informações:
    a. Na seção Disponibilidade, escolha Ambiente de runtime autônomo ou Cluster de alta disponibilidade.

O padrão de disponibilidade determina o quão disponível sua aplicação estará quando for executada. Autônomo é bom para fins de desenvolvimento. A alta disponibilidade é para aplicações que devem estar sempre disponíveis.

b. Em Recursos, escolha um tipo de instância e a capacidade desejada.

Esses recursos são as instâncias do Amazon EC2 gerenciadas pela modernização de AWS mainframe que hospedarão seu ambiente de execução. Ambientes de runtime autônomos oferecem duas opções para o tipo de instância e permitem apenas uma instância. Ambientes de runtime de alta disponibilidade oferecem duas opções para o tipo de instância e permitem até duas instâncias.

Para obter mais informações, consulte Tipos de instância do Amazon EC2 e entre em contato com um especialista em AWS mainframe para obter orientação.

6. Na seção Configurações da rede, faça o seguinte:

a. Se você quiser que as aplicações sejam acessíveis ao público, escolha Permitir que as aplicações implantadas nesse ambiente sejam acessíveis ao público.

b. Escolha uma nuvem privada virtual (VPC).

c. Se você estiver usando o padrão de alta disponibilidade, escolha duas ou mais sub-redes. Se você estiver usando o padrão autônomo com o mecanismo AWS Blu Age, escolha duas ou mais sub-redes. Se você estiver usando o padrão autônomo com o mecanismo da Micro Focus, pode especificar uma sub-rede.

d. Escolha um grupo de segurança para a VPC que você selecionou.

> **Note**
>
> AWS A modernização do mainframe cria um Network Load Balancer para você distribuir conexões ao seu ambiente de tempo de execução. Verifique se as regras de entrada do grupo de segurança permitem o acesso de um endereço IP à porta especificada na `listener` propriedade da definição da aplicação. Para obter mais informações, consulte Alvos de registro no Guia do usuário do Network Load Balancers.

e. No campo Chave KMS, escolha Personalizar configurações de criptografia se quiser usar um cliente gerenciado AWS KMS key. Para ter mais informações, consulte Criptografia de dados em repouso para o serviço de modernização AWS de mainframe.

> **Note**
>
> Por padrão, a Modernização de AWS Mainframe criptografa seus dados com dados AWS KMS key que a Modernização de AWS Mainframe possui e gerencia para você. No entanto, é possível optar por usar um sistema de AWS KMS key gerenciada pelo cliente.

f. (Opcional) Escolha um AWS KMS key por nome ou nome de recurso da Amazon (ARN). Como alternativa, escolha Criar um AWS KMS key para acessar o AWS KMS console e criar um novo. AWS KMS key

g. Selecione Next (Próximo).

7. (Opcional) Na página Anexar armazenamento, escolha um ou mais sistemas de arquivos Amazon EFS ou Amazon FSx e, em seguida, escolha Próximo.
8. Na seção Janela de manutenção, escolha quando você deseja aplicar as alterações pendentes no ambiente.

   - Se você escolher Sem preferência, a Modernização do AWS Mainframe escolherá uma janela de manutenção otimizada para você.
   - Se você quiser especificar uma janela de manutenção específica, escolha Selecionar nova janela de manutenção. Em seguida, escolha um dia da semana, uma hora de início e uma duração para a janela de manutenção.

   Para obter mais informações sobre a janela de manutenção, consulte AWS Janela de manutenção da modernização do mainframe.

   Selecione Next (Próximo).
9. Na página Revisar e criar, analise as informações fornecidas e selecione Criar fonte de dados.

Atualize um ambiente de AWS tempo de execução de modernização de mainframe

Use o console de modernização de AWS mainframe para atualizar um ambiente de tempo de execução de modernização de AWS mainframe. Você pode atualizar a versão secundária do mecanismo de runtime ou o tipo de instância que hospeda o ambiente de runtime. Você pode

escolher se deseja aplicar as atualizações imediatamente ou durante a janela de manutenção preferida.

Estas instruções supõem que você tenha concluído as etapas em Configurar a modernização AWS do mainframe.

Atualizar um ambiente de runtime

Para atualizar um ambiente de runtime

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que o ambiente que você deseja atualizar foi criado.
3. Na página Ambientes, escolha o ambiente a ser atualizado.
4. Na página de visão geral do ambiente, escolha Ações e depois Trocar URLs do ambiente.
5. Faça uma ou todas as alterações a seguir:
   - Na seção Opções do mecanismo, escolha a versão do mecanismo que você deseja.
   - Na seção Recursos, escolha o tipo de instância desejado.
   - Na seção Janela de manutenção, escolha o dia, a hora e a duração desejados.

   > **Note**
   >
   > As únicas alterações que você pode optar por aplicar durante a janela de manutenção são as alterações na versão do mecanismo. Você deve aplicar todas as outras alterações imediatamente.

6. Selecione Next (Próximo).
7. Em Quando deseja aplicar essas alterações, selecione Imediatamente ou Durante a próxima janela de manutenção. Escolha Atualizar ambiente.

Se você escolher Imediatamente, verá uma mensagem quando a atualização do ambiente for concluída.

AWS Janela de manutenção da modernização do mainframe

Cada ambiente de execução tem uma janela de manutenção semanal de duas horas. Todas as alterações do sistema são aplicadas durante esse período. A janela de manutenção é sua chance de controlar quando ocorrem modificações e correções de software e segurança. Se um evento de manutenção estiver programado para uma determinada semana, ele começará durante essa janela de manutenção de duas horas. A maioria dos eventos de manutenção também é concluída durante a janela de manutenção de duas horas, embora eventos de manutenção maiores possam levar mais do que algumas horas para serem concluídos.

A janela de manutenção de duas horas é selecionada aleatoriamente a partir de um bloco de 8 horas por região. Se você não especificar uma janela de manutenção ao criar um ambiente de tempo de execução, a Modernização do AWS Mainframe atribuirá uma janela de manutenção de 2 horas em um dia da semana selecionado aleatoriamente.

AWS A modernização do mainframe consome alguns dos recursos da instância do seu ambiente enquanto a manutenção está sendo aplicada. Você poderá observar um impacto mínimo no desempenho ou algumas interrupções nas aplicações durante a manutenção.

A tabela a seguir lista os blocos de tempo de cada região dos quais as janelas de manutenção padrão são atribuídas.

Nome da região	Região	Bloco de hora
Leste dos EUA (Norte da Virgínia)	us-east-1	Das 3h às 11h (UTC)
Oeste dos EUA (Oregon)	us-west-2	Das 6h às 14h (UTC)
Asia Pacific (Mumbai)	ap-south-1	Das 6h às 14h (UTC)
Ásia-Pacífico (Singapura)	ap-southeast-1	De 14:00 a 22:00 UTC
Asia Pacific (Sydney)	ap-southeast-2	De 12:00 a 20:00 UTC
Asia Pacific (Tokyo)	ap-northeast-1	De 13:00 a 21:00 UTC
Canada (Central)	ca-central-1	Das 3h às 11h (UTC)

Nome da região	Região	Bloco de hora
Europa (Frankfurt)	eu-central-1	De 21:00 a 05:00 UTC
Europe (Ireland)	eu-west-1	De 22:00 a 06:00 UTC
Europe (London)	eu-west-2	De 22:00 a 06:00 UTC
Europe (Paris)	eu-west-3	De 23:59 a 07:29 UTC
América do Sul (São Paulo)	sa-east-1	De 00:00 a 08:00 UTC

Interrompa um ambiente de execução de modernização de AWS mainframe

Use o console de modernização de AWS mainframe para interromper um ambiente de tempo de execução de modernização de AWS mainframe. Quando você interrompe um ambiente, as implantações atuais da aplicação são mantidas e você não será cobrado pelo ambiente até que o ambiente seja reiniciado.

Estas instruções supõem que você tenha concluído as etapas em Configurar a modernização AWS do mainframe.

Interromper um ambiente de runtime

Se você precisar interromper um ambiente de execução de modernização de AWS mainframe, siga etapas semelhantes às da seção do ambiente de atualização.

Use o console de modernização de AWS mainframe para interromper um ambiente de tempo de execução de modernização de AWS mainframe. Quando você interrompe um ambiente, as implantações atuais da aplicação são mantidas e você não será cobrado pelo ambiente até que o ambiente seja reiniciado.

Interromper um ambiente de runtime

Para interromper um ambiente de execução de modernização de AWS mainframe, siga etapas semelhantes às da seção de atualização do ambiente.

> **Note**
>
> Você deve interromper todas as aplicações antes de interromper o ambiente.

Para interromper um ambiente de runtime

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que o ambiente que você deseja interromper foi criado.
3. Na página Ambientes, escolha o ambiente a ser interrompido.
4. Na página de visão geral do ambiente, escolha Ações e depois Trocar URLs do ambiente.
5. Na página Editar ambiente, encontre a seção Recursos e atualize a capacidade desejada para zero.

   > **Note**
   >
   > Para interromper um ambiente, você só pode optar por parar imediatamente.

6. Selecione Next (Próximo).
7. Em Quando aplicar essas alterações, escolha Imediatamente. Escolha Atualizar ambiente.

Você vê uma mensagem quando a capacidade do ambiente é atualizada.

Reinicie um ambiente de AWS tempo de execução de modernização de mainframe

Use o console de modernização de AWS mainframe para reiniciar um ambiente de tempo de execução de modernização de AWS mainframe. Quando você reinicia um ambiente de runtime, a cobrança do ambiente é retomada.

Reiniciar um ambiente de runtime

Para reiniciar um ambiente de execução de modernização de AWS mainframe, siga etapas semelhantes às da seção de interrupção do ambiente.

Para reiniciar um ambiente de runtime

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que o ambiente que você deseja reiniciar foi criado.
3. Na página Ambientes, escolha o ambiente a ser reiniciado.
4. Na página de visão geral do ambiente, escolha Ações e depois Trocar URLs do ambiente.

> **Note**
>
> A capacidade desejada para um ambiente autônomo só pode ser atualizada para 1. Para reiniciar um ambiente de runtime, você só pode optar por reiniciar imediatamente.

5. Na página Editar ambiente, encontre a seção Recursos e atualize a capacidade desejada de zero para a capacidade necessária.
6. Selecione Next (Próximo).
7. Em Quando aplicar essas alterações, escolha Imediatamente. Escolha Atualizar ambiente.

Você vê uma mensagem quando a capacidade do ambiente é atualizada e o ambiente é reiniciado.

Excluir um ambiente de AWS execução de modernização de mainframe

Use o console de modernização de AWS mainframe para excluir um ambiente de tempo de execução de modernização de AWS mainframe.

Estas instruções supõem que você tenha concluído as etapas em Configurar a modernização AWS do mainframe.

Excluir um ambiente de runtime

Se você precisar excluir um ambiente de tempo de execução de modernização de AWS mainframe, certifique-se de excluir primeiro todos os aplicativos implantados do ambiente. Você não pode excluir um ambiente de runtime em que as aplicações são implantadas.

Para excluir um ambiente

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região em que o ambiente que você deseja excluir foi criado.
3. Na página Ambientes, escolha o ambiente que você deseja excluir e, em seguida, escolha Ações e Excluir ambiente.
4. Na janela Excluir ambiente, insira `delete` para confirmar que você deseja excluir o ambiente de runtime e escolha Excluir.

Teste de aplicativos na modernização AWS do mainframe

AWS O Mainframe Modernization Application Testing fornece testes automatizados de equivalência funcional para seus projetos de migração.

Tópicos

O que é teste de aplicativos de modernização de AWS mainframe?

Os testes afetam significativamente os projetos de migração. Ele pode consumir até 70% do tempo e do esforço do seu projeto de migração, modernização ou aumento. AWS O teste de aplicativos, um recurso da modernização do AWS mainframe, fornece testes automatizados de equivalência funcional para seus aplicativos migrados. O teste de equivalência funcional ajuda você a validar se seus aplicativos no Nuvem AWS são equivalentes aos aplicativos em seu mainframe. AWS O teste de aplicativos compara automaticamente as alterações em conjuntos de dados, registros de banco de dados e telas 3270 on-line entre seu mainframe e. AWS Além disso, o Application Testing permite testes repetíveis, para que você possa executar seus cenários de teste várias vezes à medida que atualiza a arquitetura de destino, resolve problemas e avança em direção a uma aplicação totalmente migrada. Após a migração, você pode continuar usando o Application Testing para testes de regressão com o objetivo de garantir que as atualizações nos mecanismos de runtime ou em outros componentes não causem regressões. O teste de aplicativos é econômico: os ambientes de teste de destino são criados usando os CloudFormation modelos fornecidos pelo usuário, aproveitando os conceitos de infraestrutura como código (IaC). O Application Testing acelera os projetos de migração

usando a elasticidade da nuvem. Você pode executar suítes de testes independentes em quantos ambientes paralelos forem necessários, reduzindo os cronogramas de teste.

Tópicos

Você é um usuário iniciante do Application Testing?

Se estiver usando o Application Testing pela primeira vez, recomendamos que você leia as seguintes seções para começar:

- Conceitos do Application Testing
- Tutorial: Configurar CardDemo
- the section called “Tutorial: Reproduza e compare no AWS Blu Age usando CardDemo”

Benefícios do Application Testing

O Application Testing oferece vários benefícios para ajudar você no processo de migração:

- Testando aceleração, agilidade e flexibilidade.
- Conceitos de teste “Grave uma vez no mainframe, repita várias vezes na AWS”.
- Criação IaC de ambientes de destino por meio de modelos fornecidos pelo usuário CloudFormation .
- Altos graus de repetibilidade de testes.
- Criado para a nuvem, com escalabilidade e elasticidade em mente.
- Testes em grande escala com alto grau de automação.
- Eficiência de custos.

Integração com AWS CloudFormation

O teste de aplicativos usa a infraestrutura como código com AWS CloudFormation. Essa opção de design simplifica e melhora sua experiência de teste. AWS CloudFormation oferece autonomia e independência para definir a melhor infraestrutura para suas necessidades. Você pode selecionar ou definir vários parâmetros (tamanho da instância, instância do RDS, grupo de segurança ideal) de forma independente. É possível adicionar recursos, como uma fila do Amazon SQS necessária para que sua aplicação funcione adequadamente em condições de teste.

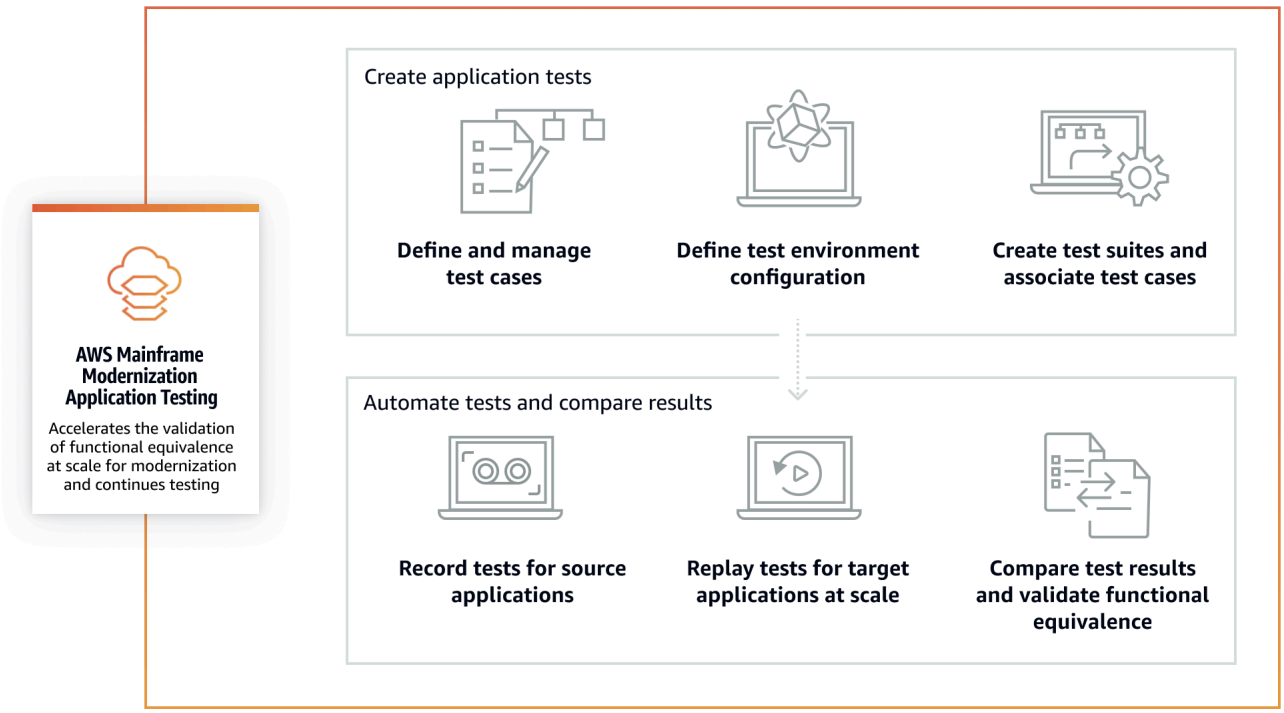
Nos AWS CloudFormation modelos fornecidos para download, você notará alguns recursos comuns:

- O teste de aplicativos cria uma pilha totalmente isolada, incluindo um ambiente de execução e um aplicativo de modernização de AWS mainframe, com suas próprias definições de rede e segurança. Essa pilha isolada fornece resiliência, porque outros atores da mesma Conta da AWS não podem interferir na atividade de teste. Ela também evita situações em que os operadores do sistema modifiquem a VPC ou o grupo de segurança padrão, o que pode causar falhas nas atividades de teste.
- O grupo de segurança também permite que você controle o acesso externo aos recursos usados nos testes. Por exemplo, um banco de dados pode conter dados confidenciais.
- O isolamento total impede que outros atores que compartilham a VPC espionem o tráfego.
- Ele aprimora o desempenho. Por exemplo, a comunicação entre o aplicativo de modernização de AWS mainframe que o modelo cria e seu banco de dados Amazon RDS ocorre em uma rede separada (uma VPC privada), o que evita que outros atores diminuam a velocidade do tráfego.

Recomendamos que você implemente esses recursos também nos AWS CloudFormation modelos criados.

Como o Application Testing funciona

A figura a seguir é uma visão geral de como o teste de aplicativos funciona.



- Você pode transferir dados de entrada da fonte para o AWS uso Transferência de arquivos ou de suas ferramentas preferidas para transferência de dados de mainframe.
- Você executa a mesma lógica de negócios na origem e no destino.
- O teste de aplicativos compara automaticamente os dados de saída (conjuntos de dados, alterações no banco de dados relacional, telas 3270 on-line e interações do usuário) da origem e do destino. Depois de executar seu cenário de teste no mainframe, você captura os dados de saída e os transfere e, em seguida AWS, reproduz o cenário de teste no destino. O teste de aplicativo compara automaticamente os dados de saída do teste executado AWS com os dados de saída da fonte. É possível ver rapidamente quais registros são idênticos, equivalentes, diferentes ou estão ausentes. Além disso, opcionalmente, é possível definir regras de equivalência, para que os registros que não sejam idênticos continuem tendo o mesmo significado comercial e sejam marcados como equivalentes.

O fluxo de trabalho a ser seguido no Application Testing consiste nas seguintes etapas:

1. Crie casos de teste: os casos de teste são a menor unidade de ações de teste. Ao criar um caso de teste, você também identifica os tipos de dados a serem comparados que melhor representam a equivalência funcional entre a origem e o destino.

2. Defina a configuração do ambiente de teste: especifique a configuração do seu ambiente especificando o AWS CloudFormation modelo e os atributos adicionais.
3. Crie suítes de teste: as suítes de teste são uma coleção de casos de teste.
4. Carregue conjuntos de dados na origem e reproduza no destino: capture os conjuntos de dados de entrada e saída no mainframe e faça o upload deles para. AWS Depois, reproduza o cenário de teste na AWS.
5. Compare conjuntos de dados de origem e destino: o teste de aplicativos compara automaticamente os conjuntos de dados de saída da origem e do destino, para que você possa ver rapidamente o que está correto e o que não está.

Tanto a ação final de um cenário de teste quanto a meta de todo o processo é identificar discrepâncias entre as execuções de teste da origem e do destino. O Application Testing compara a versão de origem e a versão de destino dos dados capturados em todos os canais de interação durante a execução do teste. Ele também compara os estados finais dos dados relevantes (conforme definido nos casos de teste).

Serviços relacionados

O teste de aplicativos é um recurso da modernização do AWS mainframe. Ele também usa a infraestrutura como código AWS CloudFormation para garantir a repetibilidade, a automação e a eficiência de custos dos testes. Para obter mais informações, consulte:

- AWS Modernização do mainframe
- AWS CloudFormation

Acessar o Application Testing

Você pode acessar o console de teste de aplicativos no console de modernização de AWS mainframe https://console.aws.amazon.com/apptest/ ou a partir dele escolhendo Teste de aplicativos no painel de navegação esquerdo.

Definição de preços do Application Testing

A definição de preços para o Application Testing pode ser encontrada em AWS Mainframe Modernization Pricing.

AWS Conceitos de teste de aplicativos de modernização de mainframe

AWS O teste de aplicativos usa termos que outros serviços de teste ou pacotes de software podem usar com um significado ligeiramente diferente. As seções a seguir explicam como o AWS Mainframe Modernization Application Testing usa essa terminologia.

Tópicos

Caso de teste

Um caso de teste é a unidade de ação individual mais atômica em seu fluxo de trabalho de teste. Normalmente, um caso de teste é usado para representar uma unidade independente da lógica de negócios que modifica os dados. As comparações serão feitas para cada caso de teste. Os casos de teste são adicionados a uma suíte de testes. Os casos de teste contêm metadados sobre os artefatos de dados (conjuntos de dados, bancos de dados) que o caso de teste modifica e sobre as funções de negócios que são acionadas durante a execução do caso de teste: trabalhos em lote, diálogos interativos 3270 e outros. Por exemplo, os nomes e as páginas de código dos conjuntos de dados.

Dados de entrada → Caso de teste → Dados de saída

Os casos de teste podem ser on-line ou do tipo em lote:

- Os casos de teste de tela 3270 on-line são casos de teste em que o usuário executa diálogos de tela interativos (3270) para ler, modificar ou produzir novos dados comerciais (registros de banco de dados e/ou conjuntos de dados).
- Casos de este em lote são casos de teste que exigem o envio de um lote para ler, processar e modificar ou produzir novos dados comerciais (conjuntos de dados e/ou registros de banco de dados).

Suíte de testes

As suítes de teste têm uma coleção de casos de teste que são executados em ordem sequencial, um por um. A repetição é feita em um nível de suíte de testes. Todos os casos de teste na suíte de testes são executados no ambiente de teste de destino quando uma suíte de testes é repetida. Se houver diferenças após comparar artefatos de teste de referência e de reprodução, as diferenças serão mostradas no nível do caso de teste.

Por exemplo, Test Suite A:

Caso de teste 1, caso de teste 2, caso de teste 3 e assim por diante.

Configuração do ambiente de teste

A configuração do ambiente de teste permite que você configure o conjunto inicial de dados e parâmetros de configuração (ou recursos) com CloudFormation os quais você precisa para tornar a execução do teste repetível.

Carregar

Os uploads são feitos em um nível de suíte de testes. Durante o upload, você deve fornecer um local do Amazon S3 que contenha os artefatos, conjuntos de dados e diários CDC do banco de dados relacional do mainframe de origem com os quais serão comparados. Eles serão considerados como dados de referência do mainframe de origem. Durante a repetição, os dados de repetição gerados serão comparados com os dados de referência enviados para garantir a equivalência do aplicativo.

Reproduzir

Os replays são feitos em um nível de suíte de testes. Durante a reprodução, o AWS Mainframe Modernization Application Testing usa o CloudFormation script para criar o ambiente de teste de destino e executar o aplicativo. Os conjuntos de dados e registros do banco de dados que são modificados durante a repetição são capturados e comparados com os dados de referência do mainframe. Normalmente, você fará o upload para o mainframe uma vez e depois reproduzirá várias vezes, até que a equivalência funcional seja alcançada.

Compare

As comparações são feitas automaticamente após o término com êxito de uma reprodução. Durante as comparações, os dados referenciados que você carregou e capturou durante a fase de upload são comparados com os dados de repetição gerados durante a fase de repetição. As comparações acontecem em um nível de caso de teste individual para conjuntos de dados, registros de banco de dados e telas on-line separadamente.

Comparações de bancos de dados

O Application Testing emprega uma funcionalidade de correspondência de progresso de estado ao comparar alterações nas gravações do banco de dados entre as aplicações de origem e de destino. A correspondência do progresso do estado compara as diferenças em cada instrução INSERT, UPDATE e DELETE de execução individual, diferentemente da comparação das linhas da tabela no final do processo. A correspondência do progresso do estado é mais eficiente do que as alternativas, fornecendo comparações mais rápidas e precisas ao comparar somente os dados alterados e ao detectar erros de autocorreção no fluxo da transação. Usando a tecnologia captura de dados de alteração (CDC), o Application Testing pode detectar alterações individuais no banco de dados relacional e compará-las entre a origem e o destino.

As alterações do banco de dados relacional são geradas na origem e no destino pelo código da aplicação testada usando instruções de linguagem de modificação de dados (DML), como SQL

INSERT, UPDATE ou DELETE, mas também indiretamente quando a aplicação está usando procedimentos armazenados, quando os acionadores do banco de dados são definidos em algumas tabelas ou quando CASCADE DELETE é usada para garantir a integridade referencial, acionando automaticamente exclusões adicionais.

Comparações de conjuntos de dados

O teste de aplicativos compara automaticamente os conjuntos de dados de referência e repetição produzidos nos sistemas de origem (gravação) e de destino (repetição).

Para comparar conjuntos de dados:

1. Comece com os mesmos dados de entrada (conjuntos de dados, banco de dados) na origem e no destino.
2. Execute seus casos de teste no sistema de origem (mainframe).
3. Capture os conjuntos de dados produzidos e carregue-os em um bucket do Amazon S3. Você pode transferir conjuntos de dados de entrada da fonte para AWS usar diários, telas e conjuntos de dados do CDC.
4. Especifique a localização do bucket do Amazon S3 em que os conjuntos de dados do mainframe foram carregados quando você fez o upload do caso de teste.

Após a conclusão da reprodução, o Teste de Aplicação compara automaticamente a referência de saída e os conjuntos de dados de destino, mostrando se os registros são idênticos, equivalentes, diferentes ou ausentes. Por exemplo, campos de data relativos ao momento da execução da workload (dia + 1, final do mês atual etc.) são automaticamente considerados equivalentes. Além disso, opcionalmente, é possível definir regras de equivalência, para que os registros que não sejam idênticos continuem tendo o mesmo significado comercial e sejam marcados como equivalentes.

Status da comparação

O Application Testing usa os seguintes status de comparação: IDÊNTICO, EQUIVALENTE e DIFERENTE.

IDÊNTICO

Os dados de origem e de destino são exatamente os mesmos.

EQUIVALENTE

Os dados de origem e de destino contêm falsas diferenças consideradas equivalências, como datas ou timestamps que não afetam a equivalência funcional quando são relativos ao momento da execução da workload. É possível definir regras de equivalência para identificar quais são essas diferenças. Quando todas as suítes de teste repetidas em comparação com suas suítes de teste de referência mostram o status IDÊNTICO ou EQUIVALENTE, sua suíte de testes não mostra diferenças.

DIFERENTE

Os dados de origem e de destino contêm diferenças, como um número diferente de registros em um conjunto de dados ou valores diferentes no mesmo registro.

Regras de equivalência

Um conjunto de regras para identificar falsas diferenças que podem ser consideradas resultados equivalentes. O teste de equivalência funcional offline (OFET) inevitavelmente causa diferenças em alguns resultados entre os sistemas de origem e de destino. Por exemplo, os timestamps de atualização são diferentes por design. As regras de equivalência explicam como se ajustar a essas diferenças e evitar falsos positivos no momento da comparação. Por exemplo, se uma data for tempo de execução de + 2 dias em uma coluna de dados específica, a regra de equivalência a descreve e aceita uma hora no sistema de destino que seja tempo de execução na meta de + 2 dias, em vez de um valor que seja estritamente igual à mesma coluna no upload da referência.

Comparação do conjunto de dados do estado final

O estado final dos conjuntos de dados que foram criados ou modificados, incluindo todas as alterações ou atualizações feitas nos conjuntos de dados a partir do estado inicial. Para conjuntos de dados, o Application Testing analisa os registros desses conjuntos de dados no final da execução de um caso de teste e compara os resultados.

Comparações de bancos de dados de progresso de estado

Comparações das alterações feitas nos registros do banco de dados como uma sequência de instruções DML individuais (Delete, Update, Insert). O Application Testing compara alterações individuais (inserir, atualizar ou excluir a linha de uma tabela) do banco de dados de origem com o banco de dados de destino e identifica as diferenças para cada alteração individual. Por exemplo,

uma instrução INSERT individual pode ser usada para inserir em uma tabela uma linha com valores diferentes no banco de dados de origem em comparação com o banco de dados de destino.

Equivalência funcional (FE)

Dois sistemas são considerados funcionalmente equivalentes se produzirem os mesmos resultados em todas as operações observáveis, dados os mesmos dados de entrada. Por exemplo, duas aplicações são consideradas funcionalmente equivalentes se os mesmos dados de entrada produzirem dados de saída idênticos (por meio de telas, alterações no conjunto de dados ou alterações no banco de dados).

Comparações online de telas 3270

Compara a saída das telas do mainframe 3270 com a saída das telas web de aplicativos modernizados quando o sistema de destino está sendo executado sob o tempo de execução do AWS Blu Age no. Nuvem AWS E compara a saída das telas 3270 do mainframe com as telas 3270 da aplicação com hospedagem redefinida quando o sistema de destino está sendo executado no runtime do Micro Focus na Nuvem AWS.

Reproduzir dados

Os dados de repetição são usados para descrever os dados gerados pela repetição de uma suíte de testes no ambiente de teste de destino. Por exemplo, os dados de repetição são gerados quando uma suíte de testes está sendo executada em um aplicativo de serviço de modernização de AWS mainframe. Os dados da reprodução são então comparados com os dados de referência capturados da origem. Toda vez que uma workload é reproduzida no ambiente de destino, uma nova geração de dados de reprodução é feita.

Dados de referência

Os dados de referência são usados para descrever os dados capturados no mainframe de origem. É a referência com a qual os dados gerados pela reprodução (destino) serão comparados. Normalmente, para cada gravação no mainframe que cria dados de referência, haverá muitas reproduções. Isso ocorre porque os usuários normalmente capturam o estado correto da aplicação no mainframe e reproduzem os casos de teste na aplicação modernizada de destino para validar a equivalência. Se forem encontrados bugs, eles serão corrigidos e os casos de teste serão reproduzidos novamente. Frequentemente, vários ciclos de reprodução, correção de bugs e nova reprodução para validar a ocorrência. Isso é chamado de paradigma de teste de captura única e várias reproduções.

Carregue, reproduza e compare

O Application Testing opera em três etapas:

- Carregar: captura os dados referenciados criados no mainframe para cada caso de teste de um cenário de teste. Isso pode incluir 3270 telas on-line, conjuntos de dados e registros de banco de dados.
  - Para telas 3270 on-line, você deve usar o emulador de terminal do Blu Insights para capturar sua workload de origem. Para obter mais informações, consulte a documentação do Blu Insights.
  - Para conjuntos de dados, você precisará capturar os conjuntos de dados produzidos por cada caso de teste no mainframe usando ferramentas comuns, como FTP ou o serviço de transferência de conjuntos de dados que faz parte da Modernização do AWS Mainframe.
  - Para alterações no banco de dados, use a documentação Replicação de dados do AWS Mainframe Modernization com a Precisely para capturar e gerar diários da CDC contendo as alterações.
- Repetição: A suíte de testes é repetida no ambiente de destino. Todos os casos de teste especificados na execução da suíte de testes. Os tipos de dados especificados criados pelos casos de teste individuais, como conjuntos de dados, alterações no banco de dados relacional ou 3270 telas, serão capturados com automação. Esses dados são conhecidos como dados de repetição e serão comparados com os dados de referência capturados durante a fase de upload.

  > **Note**
  >
  > As alterações no banco de dados relacional exigirão opções de configuração específicas do DMS em seu modelo de condição inicial. CloudFormation

- Comparar: os dados de referência do teste de origem e os dados de reprodução de destino serão comparados e os resultados serão exibidos para você como dados idênticos, diferentes, equivalentes ou ausentes.

Diferenças

Indica que foram detectadas diferenças entre os conjuntos de dados de referência e de repetição por comparação de dados. Por exemplo, um campo em uma tela 3270 on-line que mostre valores diferentes do ponto de vista da lógica de negócios entre o mainframe de origem e a aplicação

modernizada de destino será considerado uma diferença. Outro exemplo é um upload em um conjunto de dados que não é idêntico entre os aplicativos de origem e de destino.

Equivalências

Registros equivalentes são registros que são diferentes entre os conjuntos de dados de referência e de repetição, mas não devem ser tratados como diferentes do ponto de vista da lógica de negócios. Por exemplo, um registro contendo o timestamp de quando o conjunto de dados foi produzido (tempo de execução da workload). Usando regras de equivalência personalizáveis, é possível instruir o Application Testing a tratar essa diferença de falso positivo como uma equivalência, mesmo que ela mostre valores diferentes entre os dados de referência e de reprodução.

Aplicação de origem

O aplicação de origem do mainframe com a qual fazer a comparação.

Aplicação de destino

A aplicação nova ou modificada na qual o teste é feito e que será comparada à aplicação de origem para detectar quaisquer defeitos e obter equivalência funcional entre as aplicações de origem e de destino. O aplicativo de destino normalmente está sendo executado na AWS nuvem.

AWS Pré-requisitos para testes de aplicativos de modernização de mainframe

AWS O recurso de teste de aplicativos de modernização de mainframe na modernização de AWS mainframe permite que você realize testes automatizados de equivalência funcional para seus projetos de migração. Para se preparar para usar o teste de aplicativos no console de modernização do AWS mainframe, faça o seguinte:

1. Defina casos de teste: defina as unidades básicas de teste que você deseja executar e reproduzir em uma ordem específica para seu aplicativo de destino. Para obter informações adicionais sobre como criar casos de teste, consultethe section called “Casos de teste”.
2. Prepare o CloudFormation modelo e os dados de entrada: crie um CloudFormation modelo que será usado para provisionar o ambiente de teste de destino. As variáveis desse modelo serão usadas para adicionar dados de entrada e nomes de variáveis de saída em seu aplicativo de modernização de AWS mainframe. Para obter informações adicionais, consulte Trabalhando com o AWS CloudFormation modelo no Guia AWS CloudFormation do usuário.

3. Garanta o acesso ao mainframe e a captura de dados: verifique se você tem acesso ao mainframe de origem. Isso também garantirá que você possa capturar e carregar os dados de origem gerados pelos aplicativos em execução no mainframe.

Fluxos de trabalho do console

AWS O console de teste de aplicativos de modernização de mainframe ajuda você a criar casos de teste, suítes de testes e configurações de ambiente de teste.

Tópicos

Casos de teste

Um caso de teste é uma unidade atômica que representa uma determinada ação em seu fluxo de trabalho. Para obter informações adicionais sobre vários conceitos, consulte???.

> **Important**
>
> Você precisa criar pelo menos uma configuração de ambiente de teste antes de executar os casos de teste. Para criar sua primeira configuração de ambiente, consultethe section called “Configurações do ambiente de teste”.

Tópicos

Crie um caso de teste Batch

Os casos de teste em lote permitem que você envie um lote para ler, processar e modificar ou produzir novos dados comerciais (registros de banco de dados e/ou conjunto de dados).

Para criar um caso de teste Batch

1. Abra o console de teste de aplicativos de modernização de AWS mainframe em. https://console.aws.amazon.com/apptest/
2. No Região da AWS seletor, escolha a região em que o teste de aplicativos está disponível.

> **Note**
>
> Atualmente, o teste de aplicativos está disponível somente nas regiões Leste dos EUA (Norte da Virgínia), Ásia-Pacífico (Sydney), Europa (Frankfurt) e América do Sul (São Paulo).

3. No painel de navegação esquerdo, selecione Casos de teste.
4. Em Definir caso de teste, insira o nome do caso de teste e a descrição opcional. Escolha Batch em Tipo de caso de teste.
5. Escolha Próximo.
6. (Opcional) Na página Especificar parâmetros JCL em lote, adicione o nome JCL (linguagem de controle de tarefas) e seus parâmetros de tarefa (nomes e valores).
7. Escolha Próximo.
8. Na página Fonte de dados para capturar, você pode escolher entre alterações no banco de dados relacional, conjuntos de dados ou ambos.

   - Escolha Alterações no banco de dados relacional quando quiser que o caso de teste modifique os registros do banco de dados.
   - Escolha Conjuntos de dados quando quiser que o caso de teste modifique os conjuntos de dados. Em Conjuntos de dados de saída, adicione o nome do seu conjunto de dados de saída.

   > **Note**
   >
   > Você pode adicionar vários conjuntos de dados.

9. Escolha Próximo.
10. Na página Revisar e criar, revise todas as informações e escolha Criar caso de teste.

Crie um caso de teste de tela on-line 3270

Os casos de teste de tela 3270 on-line permitem que você execute diálogos de tela interativos (3270) para ler, modificar ou produzir novos dados comerciais (registros de banco de dados e/ou conjunto de dados).

Para criar um caso de teste de tela Online 3270

1. Abra o console de teste de aplicativos de modernização de AWS mainframe em. https://console.aws.amazon.com/apptest/
2. No Região da AWS seletor, escolha a região em que o teste de aplicativos está disponível.

   > **Note**
   >
   > Atualmente, o teste de aplicativos está disponível somente nas regiões Leste dos EUA (Norte da Virgínia), Ásia-Pacífico (Sydney), Europa (Frankfurt) e América do Sul (São Paulo).

3. No painel de navegação esquerdo, selecione Casos de teste.
4. Em Definir caso de teste, insira o nome do caso de teste e a descrição opcional. Escolha telas Online 3270 em Tipo de caso de teste.
5. Escolha Próximo.

   > **Note**
   >
   > A tela 3270 online não precisa que você especifique os parâmetros JCL.

6. Escolha Próximo.
7. Na página Fonte de dados para capturar, a seleção padrão é Telas Online 3270. Além disso, você pode escolher alterações no banco de dados relacional e conjuntos de dados.

   - Escolha Alterações no banco de dados relacional quando quiser que o caso de teste modifique os registros do banco de dados.
   - Escolha Conjuntos de dados quando quiser que o caso de teste modifique os conjuntos de dados. Em Conjuntos de dados de saída, adicione o nome do seu conjunto de dados de saída.

> **Note**
>
> Você pode adicionar vários conjuntos de dados.

8. Escolha Próximo.
9. Na página Revisar e criar, revise todas as informações e escolha Criar caso de teste.

Suítes de teste

As suítes de teste são séries de casos de teste que são executados em uma ordem sequencial. As suítes de teste são importantes para reproduzir casos de teste.

> **Important**
>
> Antes de criar suítes de teste, você precisa ter pelo menos um caso de teste. Você pode criar seu primeiro caso de teste usando,the section called “Casos de teste”.

Para obter informações adicionais sobre vários conceitos, consultethe section called “Conceitos do Application Testing”.

Tópicos

- Crie uma suíte de testes
- Carregar dados de referência
- Reproduza e compare

Crie uma suíte de testes

As suítes de teste permitem que você execute diferentes casos de teste, reproduza-os e compare-os posteriormente.

Para criar uma suíte de testes

1. Abra o console de teste de aplicativos de modernização de AWS mainframe em. https://console.aws.amazon.com/apptest/
2. No Região da AWS seletor, escolha a região em que o teste de aplicativos está disponível.

> **Note**
>
> Atualmente, o teste de aplicativos está disponível somente nas regiões Leste dos EUA (Norte da Virgínia), Ásia-Pacífico (Sydney), Europa (Frankfurt) e América do Sul (São Paulo).

3. No painel de navegação esquerdo, selecione Casos de teste.
4. Escolha Criar suítes de teste.
5. Na seção Criar suítes de teste, encontre casos de teste na biblioteca de casos de teste e escolha Adicionar casos de teste selecionados.

> **Note**
>
> Você pode adicionar até 20 casos de teste em uma suíte de testes.

6. No painel Suíte de testes, insira o nome da suíte de testes e a descrição opcional. Além disso, selecione entre o tempo de execução gerenciado ou o tempo de execução não gerenciado, que definirá como a suíte de testes configura e desconfigura um AWS aplicativo de modernização de mainframe. Opcionalmente, adicione o URI JSON AWS S3 do conjunto de dados de importação da modernização do mainframe.
7. Na seção Casos de teste adicionados, empilhe seus casos de teste na ordem em que você deseja carregá-los e reproduzi-los.
8. Escolha Criar suíte de testes.

Carregar dados de referência

Faça upload dos dados de referência do mainframe para o AWS Application Testing. Você só precisa salvar os dados de referência enviados na primeira vez. O serviço de teste pode reutilizar os resultados enviados da fonte e compará-los consecutivamente com os resultados repetidos no destino.

Para fazer upload de dados de referência

1. Na seção Suítes de teste, escolha a suíte de testes para carregar os dados de referência.
2. Escolha Carregar.

3. Na página Carregar dados de referência, selecione os casos de teste que você deseja reproduzir. Preencha os campos para Data de captura de dados, localização do diário de alterações do banco de dados no S3, Localização dos conjuntos de dados no S3 e Escolha Carregar.

Reproduza e compare

O processo de repetição e comparação associa seu caso de teste ao ambiente de teste de destino e executa o aplicativo. Você precisa fazer o upload dos dados antes de executar o processo de repetição.

Para reproduzir e comparar

1. Na seção Suítes de teste, escolha a suíte de testes a ser reproduzida.
2. Escolha Repetir e compare.
3. Na página Visão geral de Repetir e comparar, selecione a configuração do ambiente de teste e revise as informações. A função Editar permite que você edite qualquer campo de configuração do ambiente de teste. Você também pode encontrar AWS CloudFormation parâmetros.
4. Na seção Casos de teste a serem repetidos, escolha Casos de teste e coloque-os na ordem em que você deseja reproduzi-los.
5. Escolha Repetir e compare.

Configurações do ambiente de teste

As configurações do ambiente de teste permitem que você configure o conjunto inicial de dados e parâmetros de configuração (ou recursos) com AWS CloudFormation os quais você precisa para tornar a execução do teste repetível.

Para obter informações adicionais sobre vários conceitos, consultethe section called “Conceitos do Application Testing”.

Criar uma configuração de ambiente de teste

Configure seu ambiente de teste para reproduzir e comparar casos de teste no Application Testing.

Definir configurações do ambiente de teste

1. Abra o console de teste de aplicativos de modernização de AWS mainframe em. https://console.aws.amazon.com/apptest/
2. No Região da AWS seletor, escolha a região em que o teste de aplicativos está disponível.

> **Note**
>
> Atualmente, o teste de aplicativos está disponível somente nas regiões Leste dos EUA (Norte da Virgínia), Ásia-Pacífico (Sydney), Europa (Frankfurt) e América do Sul (São Paulo).

3. No painel de navegação esquerdo, selecione Configurações do ambiente de teste.
4. Escolha Criar configuração do ambiente de teste.
5. No painel Criar configuração do ambiente de teste, insira o nome e a descrição. Adicione também seu bucket do Amazon S3 que contém o CloudFormation modelo para testes de aplicativos. Além disso, você pode adicionar os parâmetros CloudFormation de entrada que serão usados durante a criação da CloudFormation pilha.
6. Especifique seu aplicativo de modernização do AWS Mainframe que será afetado por essa configuração de teste. Adicione o nome da variável de saída para o ID do aplicativo AWS Mainframe Modernization e o mecanismo de tempo de execução (AWS Blu Age não gerenciado ou gerenciado pela Micro Focus).

> **Note**
>
> O nome da variável de saída para o ID do aplicativo AWS Mainframe Modernization deve corresponder ao nome da variável de saída do CloudFormation modelo para criação da pilha.

> **Important**
>
> O tempo de execução não gerenciado do AWS Blu Age também exige que você especifique o nome da variável de saída para o ID do serviço do VPC endpoint, o nome da variável de saída para a porta do ouvinte e o nome da variável de saída para o nome.

WebApp Esses nomes devem corresponder aos nomes das variáveis de saída do CloudFormation modelo.

7. (Opcional) Atributos adicionais, como nome da variável de saída, podem ser definidos para a tarefa Amazon Resource Name (ARN) do Database Migration Service (DMS), que é usada para capturar alterações no banco de dados relacional. Outro atributo é o URI do banco de dados de origem DDL S3.

> **Important**
>
> O nome da variável de saída deve corresponder ao nome da variável do CloudFormation modelo.

8. (Opcional) Personalize sua chave do Serviço de Gerenciamento de Chaves (KMS). Para obter mais informações, consulte Managing access to customer managed keys (Administrando o acesso a chaves gerenciadas pelo cliente) no Guia do desenvolvedor do AWS Key Management Service .
9. Escolha Criar configuração do ambiente de teste.

Tutorial: configurar o aplicativo CardDemo de amostra

Para este tutorial, você cria uma AWS CloudFormation pilha que ajuda a configurar o aplicativo de CardDemo amostra para reformulação de plataformas com o serviço gerenciado Micro Focus on AWS Mainframe Modernization e recursos, incluindo AWS o Mainframe Modernization Application Testing. O tutorial descreve um AWS CloudFormation modelo de amostra que você pode usar para criar a pilha. Também fornecemos um arquivo compactado com os artefatos necessários da aplicação. O modelo de exemplo fornece um banco de dados, um ambiente de runtime, uma aplicação e um ambiente de rede totalmente isolado.

Esse modelo cria vários AWS recursos. Você receberá cobrança por eles se criar uma pilha com base nesse modelo.

Pré-requisitos

- Baixe e descompacte o `IC3-card-demo-zip` e o `datasets_Mainframe_ebcdic.zip`. Esses arquivos contêm a CardDemo amostra e os conjuntos de dados de amostra para uso com o Teste de AWS Aplicativos.

- Crie um bucket do Amazon S3 para armazenar os CardDemo arquivos e outros artefatos. Por exemplo, `my-carddemo-bucket`.

Etapa 1: Prepare-se para configurar CardDemo

Faça upload dos arquivos de CardDemo amostra e edite o AWS CloudFormation modelo que criará o CardDemo aplicativo.

1. Faça upload das pastas `IC3-card-demo` e `datasets_Mainframe_ebcdic` que você descompactou anteriormente para o bucket.
2. Baixe o `aws-m2-math-mf-carddemo.yaml` AWS CloudFormation modelo do seu bucket. Ele está na pasta `IC3-card-demo`.
3. Edite o `aws-m2-math-mf-carddemo.yaml` AWS CloudFormation modelo da seguinte forma:
   - Altere o parâmetro `BucketName` para o nome do bucket que você definiu anteriormente, como `my-carddemo-bucket`.
   - Altere o `ImportJsonPath` para o local do arquivo `mf-carddemo-datasets-import.json` no bucket. Por exemplo, `s3://my-carddemo-bucket/IC3-card-demo/mf-carddemo-datasets-import.json`. Ao atualizar esse valor, garanta que a saída `M2ImportJson` tenha o valor correto.
   - (Opcional) Adapte os parâmetros `EngineVersion` e `InstanceType` para que correspondam aos seus padrões.

> **Note**
>
> Não modifique as saídas `M2EnvironmentId` e `M2ApplicationId`. O Application Testing usa esses valores para localizar os recursos com os quais ele vai interagir.

Etapa 2: Criar todos os recursos necessários

Execute seu AWS CloudFormation modelo personalizado para criar todos os recursos necessários para concluir este tutorial com êxito. Esse modelo CardDemo configura o aplicativo para que você possa usá-lo em testes.

1. Faça login no AWS CloudFormation console e escolha Criar pilha e, em seguida, escolha Com novos recursos (padrão).
2. Em Pré-requisito: preparar modelo, selecione O modelo está pronto.
3. Em Especificar modelo, escolha Fazer upload de um arquivo de modelo e selecione Escolher arquivo.
4. Navegue até onde baixou `aws-m2-math-mf-carddemo.yaml` e escolha esse arquivo. Depois, selecione Próximo.
5. Em Especificar detalhes da pilha, forneça um nome para a pilha para que você possa encontrá-la facilmente em uma lista e escolha Próximo.
6. Em Configurar opções da pilha mantenha os valores padrão e escolha Próximo.
7. Em Revisão, verifique o que AWS CloudFormation está criando para você e escolha Enviar.

Demora cerca de 10 a 15 minutos para criar AWS CloudFormation a pilha.

> **Note**
>
> O modelo é configurado para acrescentar um sufixo exclusivo aos nomes dos recursos que ele cria. Isso significa que você pode criar várias instâncias desse modelo de pilha em paralelo, um recurso essencial para testes de aplicativos que permite executar várias suítes de testes ao mesmo tempo.

Etapa 3: Implantar e iniciar a aplicação

Implante o CardDemo aplicativo AWS CloudFormation criado para você e verifique se ele está em execução.

1. Abra o console de modernização do AWS mainframe e escolha Aplicativos no painel de navegação à esquerda.
2. Escolha o CardDemo aplicativo, que tem um nome parecido`aws-m2-math-mf-carddemo-abc1d2e3`.
3. Escolha Ações e, depois, escolha Implantar aplicação.
4. Em Ambientes, escolha o ambiente de runtime que corresponde à aplicação. Ele terá o mesmo identificador exclusivo anexado ao final do nome. Por exemplo, `aws-m2-math-mf-carddemo-abc1d2e3`.

5. Escolha Implantar. Espere até que a aplicação seja implantada com êxito e esteja no estado Pronta.
6. Escolha a aplicação e, depois, escolha Ações e Iniciar aplicação. Espere até que a aplicação esteja no estado Em execução.
7. Na página de detalhes da aplicação, copie a porta e o nome do host DNS necessários para se conectar à aplicação em execução.

Etapa 4: Importar os dados iniciais

Para usar o aplicativo de CardDemo amostra, você deve importar um conjunto inicial de dados. Execute as etapas a seguir.

1. Faça download do arquivo `mf-carddemo-datasets-import.json`.
2. Edite o arquivo usando o editor de texto de sua preferência.
3. Localize o parâmetro `s3Location` e atualize o valor para apontar para o bucket do Amazon S3 que você criou.
4. Faça essa mesma alteração para todas as ocorrências de `s3Location` e salve o arquivo.
5. Faça login no console do Amazon S3 e navegue até o bucket que você criou anteriormente.
6. Faça upload do arquivo `mf-carddemo-datasets-import.json` personalizado.
7. Abra o console de modernização do AWS mainframe e escolha Aplicativos no painel de navegação à esquerda.
8. Escolha o CardDemo aplicativo.
9. Escolha Conjunto de dados e selecione Importar.
10. Navegue até o local no Amazon S3 em que você fez upload do arquivo JSON personalizado e escolha Enviar.

Esse trabalho importa 23 conjuntos de dados. Para monitorar o resultado do trabalho de importação, verifique o console. Quando todos os conjuntos de dados forem importados com êxito, conecte-se à aplicação.

> **Note**
>
> Quando você usa esse modelo no Application Testing, a saída `M2ImportJson` lida automaticamente com o processo de importação.

Etapa 5: Conecte-se ao CardDemo aplicativo

Conecte-se ao aplicativo CardDemo de amostra usando o emulador 3270 de sua escolha.

- Quando a aplicação estiver em execução, use o emulador 3270 para se conectar à aplicação, especificando o nome do host DNS e o nome da porta, se necessário.

  Por exemplo, se você estiver usando o emulador c3270 de código aberto, o comando será semelhante a:

  ```
  c3270 -port port-number DNS-hostname
  ```

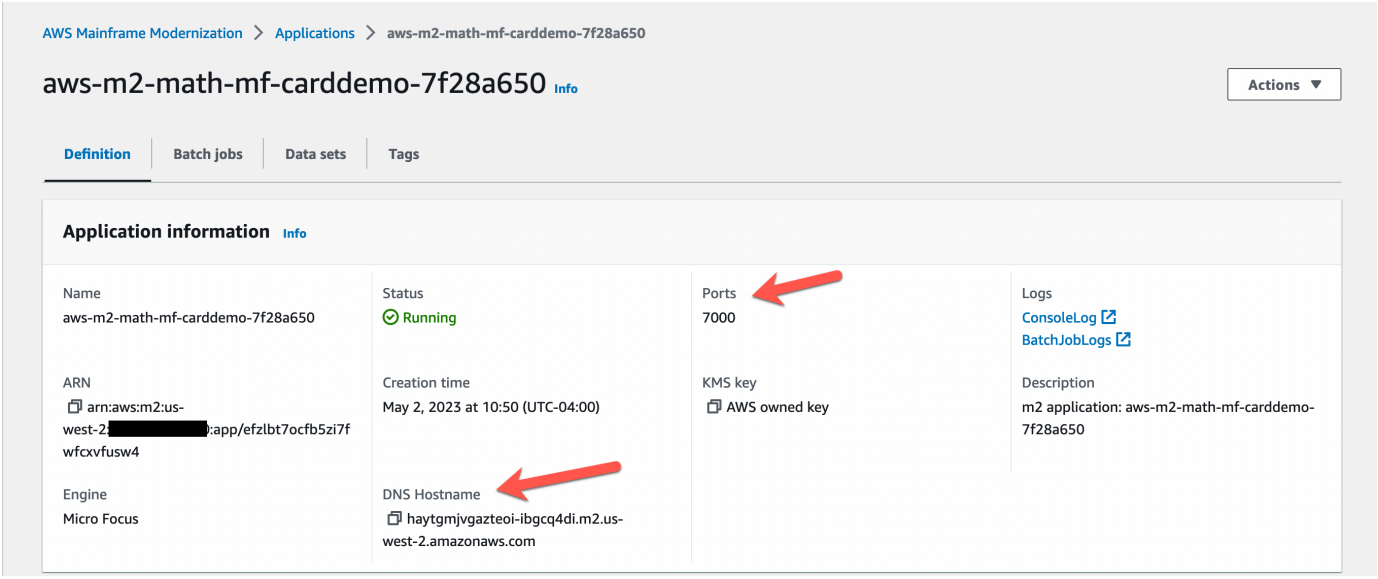
  porta

  A porta especificada na página de detalhes da aplicação. Por exemplo, 6000.

  Hostname

  O nome do host DNS especificado na página de detalhes da aplicação.

  A figura a seguir mostra onde encontrar a porta e o nome do host DSN.



Tutorial: Teste de aplicativos de modernização de AWS mainframe, reproduza e compare o uso do AWS Blu Age CardDemo implantado no Amazon EC2

Neste tutorial, você concluirá as etapas necessárias para reproduzir e comparar cargas de trabalho de teste com o CardDemo aplicativo executado no AWS Blu Age implantado no Amazon EC2.

Etapa 1: Obter a imagem de máquina da Amazon (AMI) do Amazon EC2 AWS Blu Age

Siga as instruções no tutorial de configuração do AWS Blu Age Runtime (no Amazon EC2) para ver as etapas de integração necessárias para obter acesso AWS ao Blu Age na AMI do Amazon EC2.

Etapa 2: iniciar uma instância do Amazon EC2 usando a AMI AWS Blu Age

1. Configure suas AWS credenciais.
2. Identifique a localização do arquivo binário 3.5.0 do Amazon EC2 AMI (somente CLI/versão Blu AWS Age) do bucket do Amazon S3:

```
aws s3 ls s3://aws-bluage-runtime-artifacts-xxxxxxx-eu-west-1/
aws s3 ls s3://aws-bluage-runtime-artifacts-xxxxxxx-eu-west-1/3.5.0/AMI/
```

> **Note**
>
> O recurso Application Testing está disponível para uso somente em quatro regiões em produção (us-east-1, sa-east-1, eu-central-1 e ap-southeast-2).

3. Restaure a AMI na sua conta com o seguinte comando:

```
aws ec2 create-restore-image-task --object-key 3.5.0/AMI/ami-0182ffe3b9d63925b.bin
 --bucket aws-bluage-runtime-artifacts-xxxxxxx-eu-west-1 --region eu-west-1 --name
 "AWS BLUAGE RUNTIME AMI"
```

> **Note**
>
> Substitua o nome do arquivo bin da AMI e a região onde você deseja criar a AMI.

4. Depois de criar uma instância do Amazon EC2, você poderá encontrar o ID da AMI correta que foi restaurada no bucket do Amazon S3 no catálogo de imagens do Amazon EC2.

> **Note**
>
> Neste tutorial, o ID da AMI é ami-0d0fafcc636fd1e6d, e você deve alterar esse ID nos diferentes arquivos de configuração para aquele fornecido a você.

1. Se o aws ec2 create-restore-image-task falhar, verifique sua versão do Python e da CLI usando o seguinte comando:

```
aws --version
```

> **Note**
>
> A versão do Python deve ser >= 3 e a versão da CLI deve ser >= 2.

2. Se essas versões estiverem obsoletas, a CLI deverá ser atualizada. Como atualizar a CLI:
   a. Siga as instruções em Instalar ou atualize para versão mais recente da AWS CLI.
   b. Remova a CLI v1 com o seguinte comando:

```
sudo yum remove awscli
```

   c. Instale a CLI v2 com o seguinte comando:

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
 "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

   d. Por fim, verifique a versão do Python e da CLI com o seguinte comando:

```
aws --version
```

3. Você pode então refazer o aws ec2 create-restore-image-task.

Etapa 3: Carregar arquivos CardDemo dependentes para o S3

Copie o conteúdo das pastas, bancos de dados, sistema de arquivos e dados do usuário. Baixe e descompacte os CardDemo aplicativos. Essas três pastas devem ser copiadas em um dos seus buckets chamado your-s3-bucket nesta documentação.

Etapa 4: Carregar bancos de dados e inicializar o aplicativo CardDemo

Crie uma instância temporária do Amazon EC2 que você usará como recurso computacional para gerar os snapshots de banco de dados necessários para o aplicativo. CardDemo Essa instância do EC2 não executará o CardDemo aplicativo em si, mas gerará os instantâneos do banco de dados que serão usados posteriormente.

Comece editando o CloudFormation modelo fornecido chamado 'load-and-create-ba-snapshots.yml.' Esse é o CloudFormation modelo usado para criar a instância do Amazon EC2 usada para gerar os instantâneos do banco de dados.

1. Gere e forneça seu par de chaves do EC2 que será usado para a instância do EC2. Para obter mais informações, consulte Criar pares de chaves.

   Exemplo:

   ```
   Ec2KeyPair:
       Description: 'ec2 key pair'
       Default: 'm2-tests-us-west-2'
       Type: String
   ```

2. Especifique o caminho do Amazon S3 da pasta em que você colocou a pasta banco de dados da etapa anterior:

   ```
   S3DBScriptsPath:
       Description: 'S3 DB scripts folder path'
       Type: String
       Default: 's3://your-s3-bucket/databases'
   ```

3. Especifique o caminho do Amazon S3 da pasta em que você colocou a pasta sistema de arquivos da etapa anterior:

   ```
   S3ApplicationFilesPath:
       Description: 'S3 application files folder path'
       Type: String
   ```

```
    Default: 's3://your-s3-bucket/file-system'
```

4. Especifique o caminho do Amazon S3 da pasta em que você colocou a pasta dados do usuário da etapa anterior:

```
S3UserDataPath:
    Description: 'S3 userdata folder path'
    Type: String
    Default: 's3://your-s3-bucket/userdata'
```

5. Especifique também um caminho do Amazon S3 em que você salvará os arquivos de resultados a serem usados na próxima etapa.

```
S3SaveProducedFilesPath:
    Description: 'S3 path folder to save produced files'
    Type: String
    Default: 's3://your-s3-bucket/post-produced-files'
```

6. Altere o ID da AMI pelo correto, obtido anteriormente neste tutorial, usando o modelo abaixo:

```
  BaaAmiId:
    Description: 'ami id (AL2) for ba anywhere'
    Default: 'ami-0bd41245734fd20d9'
    Type: String
```

   - Opcionalmente, você pode alterar o nome dos três instantâneos que serão criados pela execução dos bancos de dados de carga. withCloudFormation Eles ficarão visíveis na CloudFormation pilha à medida que ela for criada e serão usados posteriormente neste tutorial. Lembre-se de anotar os nomes usados para os snapshots do banco de dados.

```
SnapshotPrimary:
    Description: 'Snapshot Name DB BA Primary'
    Type: String
    Default: 'snapshot-primary'

  SnapshotBluesam:
    Description: 'Snapshot Name DB BA Bluesam'
    Type: String
    Default: 'snapshot-bluesam'
```

```
  SnapshotJics:
    Description: 'Snapshot Name DB BA Jics'
    Type: String
    Default: 'snapshot-jics'
```

> **Note**
>
> Neste documento, presumimos que o nome dos instantâneos permaneça consistente.

7. Execute o CloudFormation com CLI ou AWS console usando o botão Create Stack e o assistente. Ao final do processo, você deverá ver três snapshots no console do RDS com o nome que você escolheu seguido por um ID exclusivo. Você precisará desses nomes na próxima etapa.

   > **Note**
   >
   > O RDS adicionará postfixes aos nomes dos instantâneos definidos no modelo. AWS CloudFormation Certifique-se de obter o nome do snapshot completo do RDS antes de passar para a próxima etapa.

   Exemplo de comando da CLI

   ```
   aws cloudformation create-stack --stack-name load-and-create-ba-snapshots --
   template-url https://your-apptest-bucket.s3.us-west-2.amazonaws.com/load-and-
   create-ba-snapshots.yml --capabilities CAPABILITY_NAMED_IAM
   ```

   Você também pode verificar no caminho do Amazon S3 fornecido para o S3 se SaveProducedFilesPath os conjuntos de dados foram criados corretamente.

Etapa 5: iniciar o tempo de execução do AWS Blu Age CloudFormation

Use CloudFormation para executar a instância do Amazon EC2 com o aplicativo CardDemo AWS Blu Age. Você deve substituir algumas variáveis no CloudFormation nome `m2-with-ba-using-snapshots-https-authentication.yml` editando o arquivo YAML ou modificando os valores no console durante a inicialização do CFN.

1. Modifique o `AllowedVpcEndpointPrincipals` para especificar qual conta alcançará o VPC endpoint para acessar o tempo de execução do AWS Blu Age, usando os seguintes comandos:

```
AllowedVpcEndpointPrincipals:
    Description: 'comma-separated list of IAM users, IAM roles, or AWS accounts'
    Default: 'apptest.amazonaws.com'
    Type: String
```

2. Altere o valor das variáveis SnapshotPrimaryDb SnapshotBlusamDb, e SnapshotJicsDb para o nome dos instantâneos. Obtenha também os nomes dos snapshots do RDS depois que foram criados na etapa anterior.

```
SnapshotPrimary:
    Description: 'Snapshot DB cluster for DB Primary'
    Type: String
    Default: 'snapshot-primary87d067b0'

  SnapshotBluesam:
    Description: 'Snapshot DB cluster for DB Bluesam'
    Type: String
    Default: 'snapshot-bluesam87d067b0'

  SnapshotJics:
    Description: 'Snapshot DB cluster for DB Jics'
    Type: String
    Default: 'snapshot-jics87d067b0'
```

> **Note**
>
> O RDS adicionará seu próprio postfix aos nomes dos snapshots.

3. Forneça seu par de chaves do Amazon EC2 para a instância do EC2 usando este comando:

```
Ec2KeyPair:
    Description: 'ec2 key pair'
    Default: 'm2-tests-us-west-2'
    Type: String
```

4. Forneça a ID da AMI que você obteve durante o processo de registro da AMI para a variável BaaAmiId, usando:

```
BaaAmiId:
    Description: 'ami id (AL2) for ba anywhere'
    Default: 'ami-0d0fafcc636fd1e6d'
    Type: String
```

5. Forneça o caminho da pasta do Amazon S3 que você usou na etapa anterior para salvar os arquivos produzidos, usando o seguinte comando:

```
S3ApplicationFilesPath:
    Description: 'bucket name'
    Type: String
    Default: 's3://your-s3-bucket/post-produced-files'
```

6. Por fim, forneça o caminho da pasta s3-: userdata-folder-path

```
S3UserDataPath:
    Description: 'S3 userdata folder path'
    Type: String
    Default: 's3://your-s3-bucket/userdata'
```

- (Opcional) É possível habilitar o modo HTTPS e a autenticação HTTP básica para o tomcat. Embora as configurações padrão também funcionem.

> **Note**
>
> Por padrão, o modo HTTPS está desativado e definido para o modo HTTP no parâmetro BacHttpsMode:

Por exemplo: .

```
BacHttpsMode:
    Description: 'http or https for Blue Age Runtime connection mode '
    Default: 'http'
    Type: String
    AllowedValues: [http, https]
```

- (Opcional) Para ativar o modo HTTPS, você deve alterar o valor para HTTPS e fornecer o ARN do certificado ACM alterando o valor da variável ACM: CertArn

```
ACMCertArn:
    Type: String
    Description: 'ACM certificate ARN'
    Default: 'your arn certificate'
```

- (Opcional) A autenticação básica é desativada por padrão com o parâmetro WithBacBasicAuthenticationdefinido como false. Você poderá habilitá-lo definindo o valor como verdadeiro.

```
WithBacBasicAuthentication:
    Description: 'false or true for Blue Age Runtime Basic Authentication '
    Default: false
    Type: String
    AllowedValues: [true, false]
```

7. Depois de concluir a configuração, você pode criar a pilha usando o CloudFormation modelo editado.

Etapa 6: Testando a instância AWS Blu Age do Amazon EC2

Execute manualmente o CloudFormation modelo para criar a instância AWS Blu Age do Amazon EC2 para CardDemo o aplicativo, a fim de garantir que ele inicie sem erros. Isso é feito para verificar se o CloudFormation modelo e todos os pré-requisitos são válidos, antes de usar o CloudFormation modelo com o recurso de teste de aplicativos. Em seguida, você pode usar o Application Testing para criar automaticamente a instância AWS Blu Age Amazon EC2 de destino durante a reprodução e a comparação.

1. Execute o comando CloudFormation create stack para criar a instância AWS Blu Age do Amazon EC2, fornecendo o modelo m2 with-ba-using-snapshots - CloudFormation -https-authentication.yml que você editou na etapa anterior:

```
aws cloudformation create-stack --stack-name load-and-create-ba-snapshots —-
template-url https://apptest-ba-demo.s3.us-west-2.amazonaws.com/m2-with-ba-using-
snapshots-https-authentication.yml --capabilities CAPABILITY_NAMED_IAM --region us-
west-2
```

> **Note**
>
> Lembre-se de especificar a região correta em que a AMI AWS Blu Age foi restaurada.

2. Verifique se tudo está funcionando corretamente conferindo o console para encontrar a instância do Amazon EC2 em execução. Conecte-se a ela usando o Gerenciador de sessões.
3. Depois de se conectar à instância do Amazon EC2, use os seguintes comandos:

```
sudo su
cd /m2-anywhere/tomcat.gapwalk/velocity/logs
cat catalina.log
```

4. Certifique-se de que não haja exceções ou erros no log.
5. Depois, verifique se a aplicação está respondendo usando este comando:

```
curl http://localhost:8080/gapwalk-application/
```

Você verá a mensagem “A aplicação Jics está em execução”.

Etapa 7: Validar que as etapas anteriores foram concluídas corretamente

Nas próximas etapas, usaremos o AWS Mainframe Modernization Application Testing para reproduzir e comparar conjuntos de dados criados pelo aplicativo. CardDemo Essas etapas dependem da conclusão com êxito de todas as etapas anteriores deste tutorial. Valide o seguinte antes de continuar:

1. Você criou com sucesso a instância AWS Blu Age na Amazon EC2 por meio do AWS CloudFormation modelo.
2. O serviço Tomcat no AWS Blu Age no Amazon EC2 está funcionando, sem exceções.

Ao executar a instância do EC2 com o CardDemo aplicativo, conclua as etapas a seguir no console do Application Testing para executar a repetição e a comparação de conjuntos de dados em lote.

Etapa 8: criar o caso de teste

Nesta etapa, você cria o caso de teste que será usado para comparar os conjuntos de dados criados na aplicação Card Demo.

1. Crie um novo caso de teste. Dê a ele um nome e uma descrição.
2. Especifique `CREASTMT.JCL` como o nome do JCL.
3. Adicione os seguintes conjuntos de dados à definição do caso de teste:

Nome	CCSID	RecordFormat	RecordLength
AWS.M2.CARDDEMO.STATEMNT.PS	“037”	FB	80
AWS.M2.CARDDEMO.STATEMNT.HTML	“037”	FB	100

> **Note**
>
> O nome do JCL e os detalhes do conjunto de dados devem ser correspondentes.

Etapa 9: criar uma suíte de testes

1. Crie uma nova suíte de testes e forneça um nome e uma descrição para ela.
2. Adicione o caso de teste que você criou na etapa anterior à sua suíte de testes.
3. Depois que a suíte de testes for criada, capture os casos de teste no mainframe e faça o upload dos dados de referência do mainframe para o AWS Application Testing.
4. Escolha Criar suíte de testes.

Etapa 10: criar uma configuração de ambiente de teste

1. Crie uma nova configuração de ambiente de teste e forneça um nome e uma descrição para ela.
2. Adicione seu CloudFormation modelo. Você também pode adicionar o nome e o valor do parâmetro de entrada do seu CloudFormation modelo.
3. Escolha o AWS serviço de modernização de mainframe AWS Blu Age não gerenciado como seu tempo de execução.

4. Adicione o nome da variável de saída para o nome para o ID do aplicativo de modernização do AWS mainframe, o nome da variável de saída para o ID do serviço do VPC endpoint, o nome da variável de saída para a porta do Listener e o nome da variável de saída para o nome. WebApp

   > **Note**
   >
   > Os nomes desses campos devem corresponder aos nomes das variáveis de saída do CloudFormation modelo que serão retornados da modernização do AWS mainframe durante a criação da pilha.

5. (Opcional) Escolha o nome da variável de saída para o ARN da tarefa DMS (Database Migration Service) e a localização do URI S3 do banco de dados de origem DDL (Database definition language).
6. (Opcional) Personalize sua chave do Serviço de Gerenciamento de Chaves (KMS). Para obter mais informações, consulte Managing access to customer managed keys (Administrando o acesso a chaves gerenciadas pelo cliente) no Guia do desenvolvedor do AWS Key Management Service .
7. Escolha Criar configuração do ambiente de teste.

Etapa 11: Carregue seus dados de entrada no conjunto de testes

Nesta etapa, você executa casos de teste na fonte. Para fazer isso:

1. Baixe e execute os conjuntos de dados que se originaram da execução do aplicativo no mainframe. CardDemo
2. Faça upload da pasta descompactada para seu bucket do Amazon S3. Esse bucket do Amazon S3 deve estar na mesma região que os outros recursos do Application Testing.

   > **Note**
   >
   > Deve haver dois arquivos com nomes correspondentes aos nomes dos conjuntos de dados passados no caso de teste anterior.

3. Na página de visão geral da suíte de testes, escolha o botão Carregar.
4. Na página Carregar dados de referência, especifique a localização do Amazon S3 para onde você fez o upload dos conjuntos de dados obtidos do mainframe de origem.
5. Escolha Carregar para iniciar o processo de upload.

> **Note**
>
> Aguarde a conclusão da gravação antes de reproduzir e comparar.

Etapa 12: Reproduzir e comparar

Execute a suíte de testes e os casos de teste no ambiente AWS AWS Blu Age de destino no Amazon EC2. O Application Testing capturará os conjuntos de dados produzidos pela reprodução e os comparará com os conjuntos de dados de referência que foram registrados no mainframe.

1. Escolha Repetir e compare. A criação da CloudFormation pilha e a comparação devem levar cerca de três minutos.

Quando tudo for concluído, você deverá ter resultados de comparação com algumas diferenças criadas intencionalmente para o propósito desta demonstração.

AWS Modernização de mainframe, testes de aplicativos, conjuntos de dados suportados, páginas de código

Use a tabela a seguir para determinar se o identificador de conjunto de caracteres codificado (CCSID) de seus dados é compatível com testes de aplicativos. AWS Se os dados usarem um CCSID incompatível, recomendamos que você os converta em um CCSID compatível ou entre em contato conosco para obter ajuda.

CCSID	Conjunto de caracteres	Descrição
37	IBM037, IBM-037, Cp037	Host: EUA, Canadá (ESA), Holanda, Portugal, Brasil, Austrália, Nova Zelândia
273	IBM273, IBM-273, Cp273	Host: Áustria, Alemanha
277	IBM277, IBM-277, Cp277	Host: Dinamarca, Noruega
278	IBM278, IBM-278, Cp278	Host: Finlândia, Suécia

CCSID	Conjunto de caracteres	Descrição
280	IBM280, IBM-280, Cp280	Host: Itália
284	IBM284, IBM-284, Cp284	Host: Espanha, América Latina (espanhol)
285	IBM285, IBM-285, Cp285	Host: Reino Unido
297	IBM297, IBM-297, Cp297	Host: França
300	IBM-300	DB EBCDIC JAPÃO
301	IBM-301	Dados do PC: DB Japão
437	IBM437, IBM-437, US-ASCII, ASCII, Cp437, US-ASCII	Dados do PC: PC Base USA, muitos outros países
500	IBM500, IBM-500, Cp500	Host: Bélgica, Canadá (AS/400), Suíça, Latin-1 Internacional
720	IBM-720	MSDOS ÁRABE
737	IBM-737, x-IBM737	MSDOS GREGO
775	IBM775, IBM-775	MSDOS BÁLTICO
808	IBM-808	Dados do PC: cirílico, Rússia, com euro
813	ISO-8859-7, ISO8859_7	ISO 8859-7: Grécia
819	ISO-8859-1, ISO8859_1	ISO 8859-1: países de Latin-1
833	IBM-833	EBCDIC COREANO
834	IBM-834, x-IBM834	DB EBCDIC COREANO
835	IBM-835	DB EBCD CHINÊS TRADICIONAL

CCSID	Conjunto de caracteres	Descrição
836	IBM-836	EBCDIC CHINÊS SIMPLIFIC ADO
837	IBM-837	EBCDIC CHINÊS SIMPLIFIC ADO
850	IBM850, IBM-850, Cp850	Dados de PC: países de Latin-1
855	IBM855, IBM-855, Cp855	Dados do PC: cirílico
856	IBM-856, x-IBM856, Cp856	Dados do PC: hebraico
858	IBM00858, IBM-858, Cp858	Dados de PC: países de Latin-1, com euro
859	IBM-859	Dados do PC: LATIN-9
860	IBM860, IBM-860	Dados do PC: português
861	IBM861, IBM-861	Dados do PC: Islândia
862	IBM862, IBM-862, Cp862	Dados do PC: hebraico (migração)
863	IBM863, IBM-863	Dados do PC: Canadá
865	IBM865, IBM-865, Cp865	Dados do PC: Dinamarca/ Noruega
866	IBM866, IBM-866, Cp866	Dados do PC: cirílico, Rússia
867	IBM-867	Dados do PC: hebraico com euro
870	IBM870, IBM-870, Cp870	Host: Latin-2 multilíngue
871	IBM871, IBM-871, Cp871	Host: Islândia

CCSID	Conjunto de caracteres	Descrição
874	x-IBM874	Dados do PC: tailandês
875	IBM-875, x-IBM875, Cp875	Host: Grécia
897	IBM-897	Dados do PC: Japan SB
912	ISO-8859-2, ISO8859_2	ISO 8859-2: Latin-2 multilíng ue
915	ISO-8859-5, ISO8859_5	ISO 8859-5: cirílico
916	ISO-8859-8, ISO8859_8	ISO 8859-8: hebraico
918	IBM918, IBM-918, Cp918	Host: urdu
920	ISO-8859-9, ISO8859_9	ISO 8859-9: Latin-5 (ECMA-128, Turquia TS-5881)
921	IBM-921, x-IBM921, Cp921	Dados do PC: Letônia, Lituânia
922	IBM-922, x-IBM922, Cp922	Dados do PC: Estônia
923	ISO-8859-15, Cp923, ISO8859_15_FDIS	ISO 8859-15: Latin-9
924	IBM-924	ISO 8859-15: Latin-9
927	IBM-927	Dados do PC: chinês tradicion al
930	IBM-930, x-IBM930, Cp930	Host Katakana: SBCS estendido. Host Kanji: DBCS incluindo 4.370 caracteres definidos pelo usuário
932	IBM-932	Dados do PC: Japão Mix

CCSID	Conjunto de caracteres	Descrição
933	IBM-933, x-IBM933, Cp933	Host: SBCS estendido. Host: DBCS incluindo 1.880 caracteres definidos pelo usuário e 11.172 caracteres hangul completos
935	IBM-935, x-IBM935, Cp935	Host: SBCS estendido. Host Kanji: DBCS incluindo 1.880 caracteres definidos pelo usuário.
937	IBM-937, x-IBM937, Cp937	Host: SBCS estendido. Host Kanji: DBCS incluindo 6.204 caracteres definidos pelo usuário
939	IBM-939, x-IBM939, Cp939	Host latino: SBCS estendido . Host Kanji: DBCS incluindo 4.370 caracteres definidos pelo usuário.
942	IBM-942, IBM-942C, x-IBM942, x-IBM942C, Cp942, Cp942C	Dados do PC: SBCS estendido. Host Kanji: DBCS incluindo 1.880 caracteres definidos pelo usuário
943	IBM-943, IBM-943C, Shift_JIS, windows-31j, windows-932, x-IBM943, x-IBM943C, Cp943, Cp943C, MS932	Dados do PC: SBCS. Host Kanji: DBCS para ambiente aberto incluindo 1.880 caracteres definidos pelo usuário do IBM
947	IBM-947	BIG-5 CHINÊS TRADICIONAL

CCSID	Conjunto de caracteres	Descrição
948	IBM-948, x-IBM948, Cp948	Dados do PC: SBCS estendido. Host Kanji: DBCS incluindo 6.204 caracteres definidos pelo usuário
949	IBM-949, IBM-949C, x-IBM949, x-IBM949C, Cp949, Cp949C	Código IBM KS, dados do PC: SBCS. Código IBM KS, dados do PC: DBCS incluindo 1.880 caracteres definidos pelo usuário
950	Big5, IBM-950, x-IBM950, Cp950	Dados do PC: SBCS (IBM BIG5). Dados do PC: DBCS incluindo 13.493 CNS, 566 selecionados da IBM, 6.204 caracteres definidos pelo usuário
951	IBM-951	Dados do PC: IBM KS
954	EUC-JP, IBM-954, IBM-954C	G0: JIS X201 romano. G1: JIS X208-1990. G1: JIS X201 Katakana. G1: JIS X212
964	EUC-TW, IBM-964, x-IBM964, Cp964	G0: ASCII. G1: CNS 11.643 plano 1. G1: CNS 11.643 plano 2.
970	EUC-KR, x-IBM970, Cp970	G0: ASCII. G1: KSC X5601-1989 incluindo 1.880 caracteres definidos pelo usuário
971	IBM-971	EUC COREANO
1006	IBM-1006, x-IBM1006, Cp1006	ISO-8: Urdu

CCSID	Conjunto de caracteres	Descrição
1025	IBM-1025, x-IBM1025, Cp1025	Host: cirílico multilíngue
1026	IBM1026, IBM-1026, Cp1026	Host: Latin-5 (Turquia)
1027	IBM-1027	EBCD JAPÃO LATINO
1041	IBM-1041	Dados do PC: Japão
1043	IBM-1043	Dados do PC: chinês tradicional
1046	IBM-1046, IBM-1046S, x-IBM1046	ÁRABE: PC
1047	IBM1047, IBM-1047	Host: Latin-1
1051	hp-roman8	EMULAÇÃO HP
1088	IBM-1088	Dados do PC: KS Coreia
1089	ISO-8859-6, ISO8859_6	ISO 8859-6: Árabe
1097	IBM-1097, x-IBM1097, Cp1097	Host: Farsi
1098	IBM-1098, x-IBM1098, Cp1098	Dados do PC: farsi
1112	IBM-1112, x-IBM1112, Cp1112	Host: Letônia, Lituânia
114	IBM-1114	Dados do PC: SB T-CH
1115	IBM-1115	Dados do PC: SB S-CH
1122	IBM-1122, x-IBM1122, Cp1122	Host: Estônia

CCSID	Conjunto de caracteres	Descrição
1123	IBM-1123, x-IBM1123, Cp1123	Host: Ucrânia cirílica
1124	IBM-1124, x-IBM1124, Cp1124	8 bits: cirílico, Bielorrússia
1140	IBM01140, IBM-1140, Cp1140	Host: EUA, Canadá (ESA), Holanda, Portugal, Brasil, Austrália, Nova Zelândia com euro
1141	IBM01141, IBM-1141, Cp1141	Host: Áustria, Alemanha, com euro
1142	IBM01142, IBM-1142, Cp1142	Host: Dinamarca, Noruega, com euro
1143	IBM01143, IBM-1143, Cp1143	Host: Finlândia, Suécia, com euro
1144	IBM01144, IBM-1144, Cp1144	Host: Itália, com euro
1145	IBM01145, IBM-1145, Cp1145	Host: Espanha, América Latina (espanhol)
1146	IBM01146, IBM-1146, Cp1146	Host: Reino Unido, com euro
1147	IBM01147, IBM-1147, Cp1147	Host: França, com euro
1148	IBM01148, IBM-1148, Cp1148	Host: Bélgica, Canadá (AS/400), Suíça, Latin-1 Internacional, com euro
1149	IBM01149, IBM-1149, Cp1149	Host: Islândia, com euro

CCSID	Conjunto de caracteres	Descrição
1200	UTF-16BE	Unicode com conjunto de caracteres 65535. Na ausência de uma marca de ordem de byte (BOM), presume-se que seja UTF-16 BE (big-endian).
1202	UTF-16LE	UTF-16 LE com IBM PUA
1204	UTF-16	UTF-16 com IBM PUA
1208	UTF-8, UTF-8J, UTF8	Unicode com conjunto de caracteres 65535. UTF-8.
1232	UTF-32BE	UTF-32 BE com IBM PUA
1234	UTF-32LE	UTF-32 LE com IBM PUA
1236	UTF-32	UTF-32 com IBM PUA
1351	IBM-1351	JAPÃO ABERTO
1362	IBM-1362	MS-WIN COREANO
1363	IBM-1363, IBM-1363C, windows-949, MS949	Dados do PC: MS Windows SBCS Coreano. Dados do PC: MS Windows DBCS Coreano incluindo 11.172 Hangul completos
1364	IBM-1364	Host: SBCS estendido. Host: DBCS incluindo 1.880 caracteres definidos pelo usuário e 11.172 caracteres hangul completos

CCSID	Conjunto de caracteres	Descrição
1370	IBM-1370	Dados do PC: SBCS estendido, com euro. Dados do PC: DBCS incluindo 6.204 caracteres definidos pelo usuário, com euro
1371	IBM-1371	Host: SBCS estendido, com euro. Host: DBCS incluindo 6.204 caracteres definidos pelo usuário, com euro
1375	Big5-HKSCS	Ext Big-5 Misto para HKSCS
1380	IBM-1380	Dados do PC: SB S-CH
1381	IBM-1381, x-IBM1381, Cp1381	Dados do PC: SBCS estendido (IBM GB). Dados do PC: DBCS (IBM GB) incluindo 31 selecionados da IBM, 1.880 caracteres definidos pelo usuário
1382	IBM-1382	EUC CHINÊS SIMPLIFICADO
1383	EUC-CN, GB2312, IBM-1383, x-IBM1383, Cp1383	G0: ASCII. G1: conjunto GB 2312-80
1385	IBM-1385	Dados do PC: GBK S-CH
1386	GBK, IBM-1386, windows-936, MS936	Dados do PC: GBK Chinês Simplificado e IBM BIG-5 Chinês Tradicional. Dados do PC: GBK Chinês Simplificado

CCSID	Conjunto de caracteres	Descrição
1388	IBM-1388	Host: SBCS estendido. Host Kanji: DBCS incluindo 1.880 caracteres definidos pelo usuário
1390	IBM-1390	Host Katakana: SBCS estendido, com euro. Host Kanji: DBCS incluindo 6.205 caracteres definidos pelo usuário
1399	IBM-1399	Host Latino: SBCS estendido , com euro. Host Kanji: DBCS incluindo 4.370 caracteres definidos pelo usuário, com euro
5050	JIS0201, JIS0208, JIS0212, JIS0201, JIS0208, JIS0212	G0: JIS X201 romano. G1: JIS X208-1990. G1: JIS X201 Katakana. G1: JIS X212
5054	ISO-2022-JP	TCP JAPONÊS
5346	windows-1250, Cp1250	MS Windows: Latin-2, versão 2 com euro
5347	windows-1251, Cp1251	MS Windows: cirílico, versão 2 com euro
5348	windows-1252, Cp1252	MS Windows: países de Latin-1, versão 2 com euro
5349	windows-1253, Cp1253	MS Windows: Grécia, versão 2 com euro
5350	windows-1254, Cp1254	MS Windows: Turquia, versão 2 com euro

CCSID	Conjunto de caracteres	Descrição
5351	windows-1255, Cp1255	MS Windows: Hebraico, versão 2 com euro
5352	windows-1256, windows-1256S, Cp1256	MS Windows: Árabe, versão 2 com euro
5353	windows-1257, Cp1257	MS Windows: Borda do Báltico, versão 2 com euro
5354	windows-1258, Cp1258	MS Windows: vietnamita, versão 2 com euro
5488	GB18030	GB18030, dados de 1 byte GB18030, dados de 2 bytes GB18030, dados de 4 bytes
9030	IBM-838, Cp838	Host: SBCS tailandês estendido.
9066	IBM-874, Cp874	Dados do PC: SBCS tailandês estendido
9400	CESU-8	CESU-8 com IBM PUA
2546	ISO-2022-KR	TCP COREANO
33722	IBM-33722, IBM-33722C	IBMeucJP

Proteção de dados em testes de AWS aplicativos de modernização de mainframe

O modelo de responsabilidade AWS compartilhada O modelo de se aplica à proteção de dados no AWS Mainframe Modernization Application Testing. Conforme descrito neste modelo, AWS é responsável por proteger a infraestrutura global que executa todos os Nuvem AWS. Você é responsável por manter o controle sobre seu conteúdo hospedado nessa infraestrutura. Você também é responsável pelas tarefas de configuração e gerenciamento de segurança dos Serviços

da AWS que usa. Para obter mais informações sobre a privacidade de dados, consulte as Perguntas Frequentes sobre Privacidade de Dados.. Para obter mais informações sobre a proteção de dados na Europa, consulte a postagem do blog AWS Shared Responsibility Model and GDPR no Blog de segurança da AWS .

Recomendamos que você proteja Conta da AWS as credenciais e configure usuários individuais com AWS IAM Identity Center ou AWS Identity and Access Management (IAM). Como resultado, cada usuário recebe apenas as permissões necessárias para cumprir suas tarefas. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com os recursos. AWS Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure a API e o registro de atividades do usuário com AWS CloudTrail.
- Use soluções de AWS criptografia, juntamente com todos os controles de segurança padrão Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sigilosos armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-2 ao acessar AWS por meio de uma interface de linha de comando ou de uma API, use um endpoint FIPS. Para ter mais informações sobre endpoints do FIPS disponíveis, consulte Federal Information Processing Standard (FIPS) 140-2.

Recomendamos que você evite usar informações confidenciais ou sigilosas, como endereços de e-mail de seus clientes, em tags ou campos de texto de formato livre (por exemplo, campo Nome). Isso inclui quando você trabalha com testes de aplicativos de modernização de AWS mainframe ou outros Serviços da AWS usando o console, a API ou AWS os AWS CLI SDKs. Todos os dados inseridos em tags ou campos de texto de formato livre usados para nomes podem ser usados para registros de faturamento ou diagnóstico. Se você fornecer uma URL para um servidor externo, evite usar as informações de credenciais na URL para validar sua solicitação para esse servidor.

Dados coletados pelo AWS Mainframe Modernization Application Testing

AWS O teste de aplicativos de modernização de mainframe coleta vários tipos de dados de você:

- `Resource definition`: a definição do recurso indica os dados passados para o Application Testing quando você cria ou atualiza um recurso do tipo caso de teste, suíte de testes ou configuração de teste.
- `Scripts for replay`: esses são scripts passados para testes de aplicativos em seu aplicativo de modernização de AWS mainframe.
- `Data for comparison`: são conjuntos de dados ou arquivos de captura de dados de alteração de banco de dados (CDC) passados para testes de aplicativos para comparação.

AWS O Mainframe Modernization Application Testing armazena esses dados nativamente em. AWS Os dados que coletamos de você são armazenados em um bucket Amazon S3 gerenciado pelo AWS Mainframe Modernization Application Testing. Quando você exclui um recurso, os dados associados são removidos do bucket do Amazon S3.

Quando você inicia uma execução de teste para realizar a repetição para testar cargas de trabalho interativas, o AWS Mainframe Modernization Application Testing baixa o script em um contêiner Fargate gerenciado pelo Amazon ECS e baseado em armazenamento temporário para realizar a repetição. O arquivo de script é excluído quando a reprodução é concluída e o arquivo de saída gerado pelo script é armazenado no bucket Amazon S3 gerenciado pelo Application Testing em sua conta. O arquivo de saída de repetição é excluído do bucket do Amazon S3 quando você exclui a execução do teste.

Da mesma forma, quando você inicia um teste para comparar arquivos (conjuntos de dados ou alterações no banco de dados), o AWS Mainframe Modernization Application Testing baixa os arquivos em um contêiner Fargate gerenciado pelo Amazon ECS com armazenamento temporário para realizar a comparação. Os arquivos baixados são excluídos assim que a operação de comparação for concluída. Os dados de saída da comparação são armazenados no bucket Amazon S3 gerenciado pelo Application Testing em sua conta. Os dados de saída são excluídos do bucket do S3 quando você exclui a execução do teste.

Você pode usar todas as opções de criptografia do Amazon S3 disponíveis para proteger seus dados ao colocá-los no bucket do Amazon S3 AWS que o Mainframe Modernization Application Testing usa para comparar arquivos.

Criptografia de dados em repouso para o teste de aplicativos de modernização do AWS mainframe

AWS O Mainframe Modernization Application Testing se integra ao AWS Key Management Service (KMS) para fornecer criptografia transparente do lado do servidor (SSE) em todos os recursos dependentes que armazenam dados permanentemente. Exemplos de recursos incluem Amazon Simple Storage Service, Amazon DynamoDB e Amazon Elastic Block Store. AWS O Mainframe Modernization Application Testing cria e gerencia AWS KMS chaves de criptografia simétricas para você em. AWS KMS

A criptografia de dados em repouso por padrão ajuda a reduzir a sobrecarga operacional e a complexidade envolvidas na proteção de dados confidenciais. Ao mesmo tempo, ele permite que você teste aplicativos que exigem rigorosa conformidade com criptografia e requisitos regulatórios.

Você não pode desativar essa camada de criptografia nem selecionar um tipo de criptografia alternativo ao criar casos de teste, suítes de testes ou configurações de teste.

Você pode usar sua própria chave gerenciada pelo cliente para comparar arquivos e AWS CloudFormation modelos para criptografar o Amazon S3. Você pode usar essa chave para criptografar todos os recursos criados para execuções de teste no Application Testing.

> **Note**
>
> Os recursos do DynamoDB são sempre criptografados usando Chave gerenciada pela AWS uma conta de serviço na Application Testing. Não é possível criptografar recursos do DynamoDB usando uma chave gerenciada pelo cliente.

AWS O teste de aplicativos de modernização de mainframe usa sua chave gerenciada pelo cliente para as seguintes tarefas:

- Exportação de conjuntos de dados do Application Testing para o Amazon S3.
- Carregar arquivos de saída de comparação para o Amazon S3.

Para obter mais informação, consulte Chaves gerenciadas pelo cliente no Guia do desenvolvedor do AWS Key Management Service .

Criar uma chave gerenciada pelo cliente

Você pode criar uma chave simétrica gerenciada pelo cliente usando as AWS Management Console ou as AWS KMS APIs.

Para criar uma chave simétrica gerenciada pelo cliente

Siga as etapas de Criar uma chave simétrica gerenciada pelo cliente no Guia do desenvolvedor do AWS Key Management Service .

Política de chave

As políticas de chaves controlam o acesso à chave gerenciada pelo seu cliente. Cada chave gerenciada pelo cliente deve ter exatamente uma política de chaves, que contém declarações que determinam quem pode usar a chave e como pode usá-la. Ao criar a chave gerenciada pelo cliente, é possível especificar uma política de chaves.

A seguir está um exemplo de política de chaves com acesso limitado ViaService que permite que o Application Testing grave dados gerados por repetição e comparação em sua conta. Você deve anexar essa política à função do IAM ao invocar a `StartTestRun` API.

Example

```
{
    "Sid": "TestRunKmsPolicy",
    "Action": ["kms:Decrypt", "kms:GenerateDataKey"],
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/TestRunRole"
    },
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "kms:ViaService": ["s3.amazonaws.com"]
        },
        "ForAnyValue:StringEquals": {
            "kms:EncryptionContextKeys": "aws:apptest:testrun"
        }
    }
}
```

Para obter mais informações, consulte Managing access to customer managed keys (Administrando o acesso a chaves gerenciadas pelo cliente) no Guia do desenvolvedor do AWS Key Management Service .

Para obter mais informações sobre solução de problemas de acesso à chave, consulte o Guia do Desenvolvedor do AWS Key Management Service .

Especificação de uma chave gerenciada pelo cliente para testes de aplicativos de modernização de AWS mainframe

Ao criar uma configuração de teste, você pode especificar uma chave gerenciada pelo cliente inserindo uma ID da CHAVE. O teste de aplicativos é usado para criptografar os dados enviados para o bucket do Amazon S3 durante a execução do teste.

- KEY ID — Um identificador de chave para uma chave gerenciada pelo cliente. Insira uma ID de chave, um ARN de chave, um nome de alias ou um ARN de alias.

Para adicionar sua chave gerenciada pelo cliente ao criar uma configuração de teste com o AWS CLI, especifique o `kmsKeyId` parâmetro da seguinte forma:

```
create-test-configuration --name test \
--resources '[{
    "name": "TestApplication",
    "type": {
        "m2ManagedApplication": {
            "applicationId": "wqju4m2dcz3rhny5fpdozrsdd4",
            "runtime": "MicroFocus"
        }
    }
}]' \
--service-settings '{
    "kmsKeyId": "arn:aws:kms:us-west-2:111122223333:key/05d467z6-c42d-40ad-
b4b7-274e68b14013"
}'
```

AWS Modernização de mainframe, teste de aplicativos, contexto de criptografia

Um contexto de criptografia é um conjunto opcional de pares chave-valor que pode conter informações contextuais adicionais sobre os dados.

AWS KMS usa o contexto de criptografia como dados autenticados adicionais para oferecer suporte à criptografia autenticada. Quando você inclui um contexto de criptografia em uma solicitação para criptografar dados, AWS KMS vincula o contexto de criptografia aos dados criptografados. Para descriptografar os dados, você inclui o mesmo contexto de criptografia na solicitação.

AWS Modernização de mainframe, teste de aplicativos, contexto de criptografia

AWS O teste de aplicativos de modernização de mainframe usa o mesmo contexto de criptografia em todas as operações AWS KMS criptográficas relacionadas a uma execução de teste, em que a chave está `aws:apptest:testrun` e o valor é o identificador exclusivo da execução do teste.

Example

```
"encryptionContext": {
        "aws:apptest:testrun": "u3qd7uhdandgdkhhi44qv77iwq"
}
```

Uso do contexto de criptografia para monitoramento

Ao usar uma chave simétrica gerenciada pelo cliente para criptografar sua execução de teste, você também pode usar o contexto de criptografia em registros e registros de auditoria para identificar como a chave gerenciada pelo cliente está sendo usada ao carregar dados para o Amazon S3.

Monitorando suas chaves de criptografia para testes de AWS aplicativos de modernização de mainframe

Ao usar uma chave gerenciada pelo AWS KMS cliente com seus recursos de teste de aplicativos de modernização de AWS mainframe, você pode usá-la AWS CloudTrailpara rastrear solicitações que o teste de aplicativos de modernização de AWS mainframe envia para o Amazon S3 ao carregar objetos.

Criptografia em trânsito

Para casos de teste que definem etapas para testar cargas de trabalho transacionais, as trocas de dados entre o emulador de terminal gerenciado do Application Testing que executa seus scripts selenium e os endpoints do aplicativo de modernização do AWS mainframe não são criptografadas em trânsito. AWS O Mainframe Modernization Application Testing usa AWS PrivateLink para se conectar ao endpoint do seu aplicativo para trocar dados de forma privada sem expor o tráfego pela Internet pública.

AWS O teste de aplicativos de modernização de mainframe usa HTTPS para criptografar as APIs do serviço. Todas as outras comunicações no AWS Mainframe Modernization Application Testing são protegidas pelo serviço VPC ou pelo grupo de segurança, bem como pelo HTTPS.

A criptografia básica em trânsito é configurada por padrão, mas não se aplica aos testes de carga de trabalho interativos baseados em `TN3270` protocolo.

Transferência de arquivos na modernização AWS do mainframe

O AWS Mainframe Modernization File Transfer permite transferir e converter conjuntos de dados de mainframe para o Amazon S3 para casos de uso de modernização, migração e aumento de mainframe.

Tópicos

O que é o Transferência de Arquivos do AWS Mainframe Modernization?

Com o AWS Mainframe Modernization File Transfer, você pode transferir e converter conjuntos de dados e arquivos com um serviço totalmente gerenciado para acelerar e simplificar os casos de uso de modernização, migração e aumento para o serviço de AWS modernização de mainframe e o Amazon S3.

Tópicos

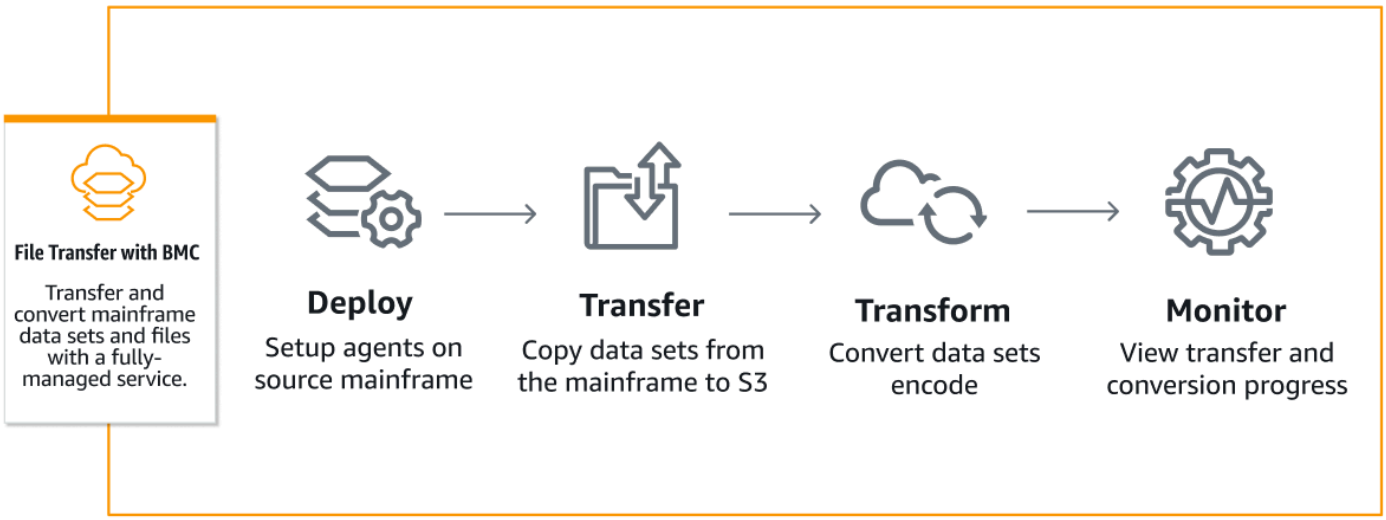
Benefícios do Transferência de Arquivos do AWS Mainframe Modernization

O Transferência de Arquivos do AWS Mainframe Modernization ajuda você a transferir conjuntos de dados do mainframe para o Amazon S3. Alguns benefícios incluem:

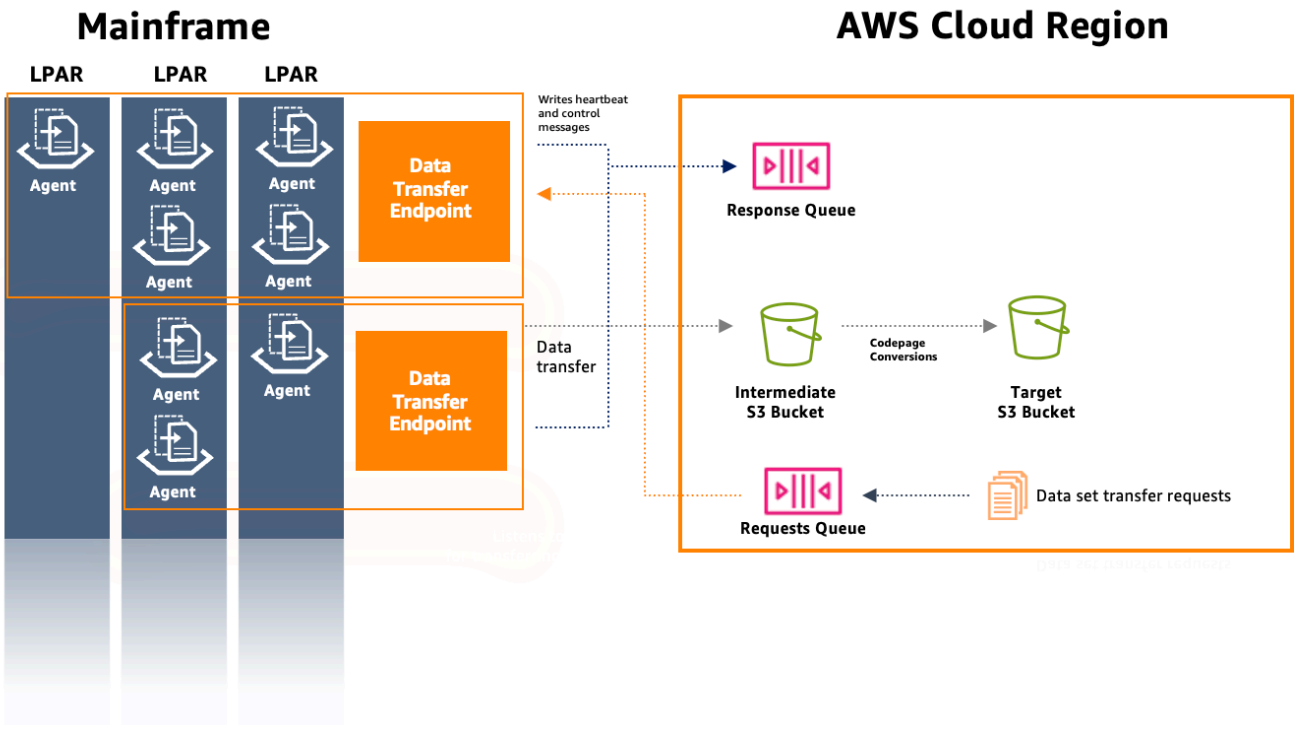
- Descoberta de conjuntos de dados e artefatos do mainframe de origem
- Transferências automatizadas e conversão de conjuntos de dados
- Escalabilidade, eficiência e velocidade para obter transferências mais rápidas de conjuntos de dados para a AWS

Como funciona o Transferência de Arquivos do AWS Mainframe Modernization

A figura a seguir é uma visão geral de como funciona o Transferência de Arquivos do AWS Mainframe Modernization em um nível conceitual.



A figura a seguir é uma visão geral da arquitetura do recurso Transferência de Arquivos do AWS Mainframe Modernization.



Instalar um agente do Transferência de Arquivos

Você pode usar este documento como um step-by-step guia para instalar um agente no mainframe de origem.

Tópicos

- Etapa 1: criar um conjunto de dados zFS para o agente M2
- Etapa 2: formatar o conjunto de dados como zFS
- Etapa 3: montar o sistema de arquivos
- Etapa 4: verificar a montagem
- Etapa 5: insira o OMVS
- Etapa 6: definir a variável de ambiente do diretório de instalação do agente
- Etapa 7: definir a variável de ambiente do diretório de trabalho
- Etapa 8: criar o diretório de trabalho
- Etapa 9: Copiar o arquivo tar do agente e copiar o diretório de trabalho
- Etapa 10: suponha o usuário root
- Etapa 11: Concluir a instalação do agente

Etapa 1: criar um conjunto de dados zFS para o agente M2

Crie um zFS para a instalação do M2-Agent usando o JCL abaixo.

```
//DEFINE   EXEC   PGM=IDCAMS
//SYSPRINT DD     SYSOUT=A
//SYSIN    DD     *
 DEFINE CLUSTER (NAME(yourhlq.M2AGENT.ZFS) -
 VOLUMES(*) -
 LINEAR CYL(1000 200))
```

Etapa 2: formatar o conjunto de dados como zFS

Depois de criar o conjunto de dados, formate-o como um sistema de arquivos zFS.

Uma maneira de fazer isso é usando a seguinte Job Control Language (JCL):

```
//FORMAT  EXEC PGM=IOEAGFMT,PARM='AGGRNAME(yourhlq.M2AGENT.ZFS),FORMAT,AGGRSIZE(1200)'
//SYSPRINT DD  SYSOUT=A
```

Envie esse trabalho e verifique se ele foi concluído com êxito.

Etapa 3: montar o sistema de arquivos

Para montar o sistema de arquivos, use o comando `MOUNT`. É possível montar o sistema de arquivos na linha de comando no ISPF ou em lote.

Por exemplo: .

```
MOUNT FILESYSTEM('yourhlq.M2AGENT.ZFS') TYPE(ZFS) MODE(RDWR) MOUNTPOINT('/usr/lpp/aws/
m2-agent')
```

Etapa 4: verificar a montagem

Verifique se o sistema de arquivos foi montado corretamente usando o comando `D OMVS,F` ou verificando no Unix System Service (USS).

Etapa 5: insira o OMVS

Use o comando a seguir para entrar no OMVS:

```
TSO OMVS
```

Etapa 6: definir a variável de ambiente do diretório de instalação do agente

Use o seguinte comando para definir o ambiente do diretório de instalação do agente:

```
export AGENT_DIR=/usr/lpp/aws/m2-agent
```

> **Note**
>
> O ponto de montagem é definido na etapa 3.

Etapa 7: definir a variável de ambiente do diretório de trabalho

Use o seguinte comando para definir a variável de ambiente do diretório de trabalho:

```
export WORK_DIR=$AGENT_DIR/tmp
```

Etapa 8: criar o diretório de trabalho

Use o seguinte comando para definir a variável de ambiente do diretório de trabalho:

```
mkdir -p $WORK_DIR
```

Etapa 9: Copiar o arquivo tar do agente e copiar o diretório de trabalho

Faça o download do arquivo tar do agente da AWS usando o link do agente M2.

O mecanismo de transferência dependerá do seu ambiente, mas certifique-se de que o arquivo tar seja transferido no modo binário.

Etapa 10: suponha o usuário root

Use o seguinte comando para assumir o usuário raiz:

```
su
```

Etapa 11: Concluir a instalação do agente

Siga estas etapas para concluir a instalação do agente.

1. Defina a variável de ambiente da versão m2-agent para a versão que está sendo instalada no momento usando o seguinte comando:

```
export M2_AGENT_VERSION=1.0.0
```

2. Crie o pacote tar do agente usando o seguinte comando:

```
tar -xpf m2-agent-package-$M2_AGENT_VERSION.tar -C $AGENT_DIR
```

3. Crie um link `current-version` simbólico para o diretório atual de instalação do agente com o seguinte comando:

```
ln -s $AGENT_DIR/m2-agent-v$M2_AGENT_VERSION $AGENT_DIR/current-version
```

4. Atualize e envie `CPY#PDS` para criar os conjuntos de dados do agente de transferência de arquivos.

> **Note**
>
> A JCL usa o. `SYS2.AWS.M2 HLQ`

Para criar o agente do Transferência de Arquivos, defina as linhas de parâmetros 000006 a 000012. Além disso, atualize as três variáveis simbólicas `HLQ`, `VOLSER` e `AGNTPATH` para serem usadas posteriormente na JCL:

```
oedit $AGENT_DIR/current-version/installation/CPY#PDS
submit $AGENT_DIR/current-version/installation/CPY#PDS
```

> **Note**
>
> Essa JCL é personalizada para configurar certos aspectos da instalação do agente no mainframe. Ele aloca os conjuntos de dados necessários e, em seguida, copia arquivos específicos do sistema de arquivos Unix para esses conjuntos de dados.

Configurar um agente de transferência de arquivos

Depois de instalar um agente de transferência de arquivos, siga estas etapas para configurar o agente. Se precisar instalar um novo agente, siga as instruções na the section called “Instalar um agente do Transferência de Arquivos” página.

Tópicos

Etapa 1: Configurar permissões e iniciar o controle de tarefas (STC)

1. Atualize e envie uma dos `SYS2.AWS.M2.SAMPLIB(SEC#RACF)` (para configurar as permissões do RACF) ou `SYS2.AWS.M2.SAMPLIB(SEC#TSS)` (para configurar as permissões do TSS) de acordo com suas instruções. Esses membros foram criados pela etapa anterior de `CPY#PDS`.

   > **Note**
   >
   > `SYS2.AWS.M2`é o qualificador de alto nível (HLQ) que foi escolhido durante a instalação.

2. Atualize a exportação de PWD no STC JCL `SYS2.AWS.M2.SAMPLIB(M2AGENT)`, se o caminho padrão do diretório do agente do Transferência de Arquivos (`/usr/lpp/aws/m2-agent`) tiver sido alterado.
3. Atualize e copie o JCL `SYS2.AWS.M2.SAMPLIB(M2AGENT)` no `SYS1.PROCLIB`.
4. Adicione `SYS2.AWS.M2.LOADLIB` à lista de APF usando o seguinte comando:

```
SETPROG APF ADD DSNAME(SYS2.AWS.M2.LOADLIB) SMS
```

5. Defina o grupo e o proprietário das pastas `logs` e `diag` do agente como o usuário/grupo do agente (M2USER/M2GROUP). Use o seguinte comando:

```
chown -R M2USER:M2GROUP $AGENT_DIR/current-version/logs
chown -R M2USER:M2GROUP $AGENT_DIR/current-version/diag
```

Etapa 2: criar buckets do Amazon S3

O AWS Mainframe Modernization File Transfer exige um bucket intermediário do Amazon S3 como área de trabalho. Recomendamos criar um bucket especificamente para isso.

Opcionalmente, crie um novo bucket Amazon S3 de destino para os conjuntos de dados transferidos. Caso contrário, você também pode usar seu bucket Amazon S3 existente. Para obter mais informações sobre a criação de buckets do Amazon S3, consulte Criação de um bucket.

Etapa 3: criar uma chave gerenciada pelo AWS KMS cliente para criptografia

Para criar uma chave gerenciada pelo cliente em AWS KMS

1. Abra o AWS KMS console emhttps://console.aws.amazon.com/kms.
2. Escolha Chaves gerenciadas pelo cliente no painel de navegação esquerdo.
3. Escolha Create key (Criar chave).
4. Em Configurar chave, escolha Tipo de chave como Simétrico e Uso da chave como criptografia e descriptografia. Use outras configurações padrão.
5. Em Adicionar rótulos, adicione um alias e uma descrição para sua chave.
6. Escolha Próximo.
7. Em Definir as principais permissões administrativas, escolha pelo menos um usuário e uma função do IAM que administra essa chave.
8. Escolha Próximo.
9. Na página Revisar, adicione a seguinte sintaxe à política de chaves. Isso permite que o serviço de modernização do AWS mainframe leia e use essas chaves para criptografia/descriptografia.

> **Important**
>
> Adicione a declaração às declarações existentes. Não substitua o que já está na política.

```
{
    "Sid" : "Enable AWS M2 File Transfer Permissions",
    "Effect" : "Allow",
    "Principal" : {
        "Service" : "m2.amazonaws.com"
    },
    "Action" : [
        "kms:Encrypt",
        "kms:Decrypt"
    ],
   "Resource" : "*"
},
```

Salve o ARN da chave gerenciada pelo cliente depois que ela for criada. Ele será usado na política posteriormente.

Etapa 4: criar um AWS Secrets Manager segredo para as credenciais do mainframe

As credenciais do mainframe são necessárias para acessar os conjuntos de dados a serem transferidos e devem ser armazenadas como um AWS Secrets Manager segredo.

Para criar um AWS Secrets Manager segredo

1. Abra o console do gerenciador Secrets emhttps://console.aws.amazon.com/secretsmanager.
2. Em Escolher tipo de segredo, escolha Outro tipo de segredo.
3. Use o valor da chave `userId` para o UserID do mainframe que tem acesso aos conjuntos de dados.
4. Use o valor da chave `password` para o campo de senha.
5. Em Chave de criptografia, escolha a chave gerenciada pelo AWS cliente criada anteriormente.
6. Escolha Próximo.

7. Na página Configurar segredo, forneça um nome e uma descrição.
8. Na mesma página, edite as permissões do recurso e use a política de recursos a seguir para que o serviço de modernização do AWS mainframe possa acessá-la.

```
{
  "Version" : "2012-10-17",
  "Statement" : [ {
    "Effect" : "Allow",
    "Principal" : {
        "Service" : "m2.amazonaws.com"
    },
    "Action" : [ "secretsmanager:GetSecretValue",
                 "secretsmanager:DescribeSecret" ],
    "Resource" : "*"
  } ]
}
```

9. Escolha Salvar para salvar as permissões atualizadas antes de escolher Avançar.
10. Passe pela página Configurar rotações e escolha Avançar.
11. Na página Revisar, verifique todas as configurações e escolha Armazenar para salvar o segredo.

> **Important**
>
> As `userId` chaves `password` secretas fazem distinção entre maiúsculas e minúsculas e devem ser inseridas conforme mostrado.

Etapa 5: criar uma política do IAM

Para criar uma nova política com as permissões necessárias para o agente

1. Mude do editor visual para o editor JSON e substitua o conteúdo pelo seguinte modelo:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
         "Sid": "FileTransferAgentSQSReceive",
```

```
            "Effect": "Allow",
            "Action": [
                "sqs:DeleteMessage",
                "sqs:ReceiveMessage"
            ],
            "Resource": "arn:aws:sqs:*:111122223333:m2-*-request-queue.fifo"
        },
        {
         "Sid": "FileTransferAgentSQSSend",
            "Effect": "Allow",
            "Action": "sqs:SendMessage",
            "Resource": "arn:aws:sqs:*:111122223333:m2-*-response-queue.fifo"
        },
        {
         "Sid": "FileTransferWorkingS3",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "<file-transfer-endpoint-intermediate-bucket-arn>/*"
        },
        {
         "Sid": "FileTransferAgentKMSDecrypt",
            "Effect": "Allow",
            "Action": "kms:Decrypt",
            "Resource": "<kms-key-arn>"
        }
    ]
}
```

2. Substitua os `111122223333` ARNs da fila de solicitações e da fila de respostas pela sua conta.

> **Note**
>
> Esses são ARNs curingas que correspondem às duas filas do Amazon SQS criadas durante a inicialização do endpoint de transferência de dados. Depois de criar um endpoint de transferência de arquivos, substitua opcionalmente esses ARNs pelos valores reais do Amazon SQS.

3. `file-transfer-endpoint-intermediate-bucket-arn`Substitua pelo ARN do bucket de transferência criado anteriormente. Deixe o caractere curinga “/*” no final.
4. `kms-key-arn`Substitua pelo ARN da AWS KMS chave criada anteriormente.

Etapa 6: criar um usuário do IAM com credenciais de acesso de longo prazo

Crie um usuário do IAM que permita que o agente do mainframe se conecte à sua AWS conta. O agente se conectará a esse usuário e, em seguida, assumirá uma função que você define com permissões para usar as filas de resposta e solicitação do Amazon SQS e para salvar conjuntos de dados nos buckets do Amazon S3.

Para criar esse usuário do IAM

1. Navegue até o console do AWS IAM emhttps://console.aws.amazon.com/iam.
2. Nas opções de Permissões, escolha a opção Anexar políticas diretamente, mas não anexe nenhuma política de permissões. Essas permissões serão gerenciadas por uma função que será anexada.
3. Depois que o usuário for criado, escolha o usuário e abra a guia Credenciais de segurança.
4. Em Criar chave de acesso, escolha Outro quando solicitado para Caso de uso.
5. Copie e salve com segurança a chave de acesso e a chave de acesso secreta geradas. Eles serão usados posteriormente.

Para obter mais informações sobre a criação de chaves de acesso do IAM, consulte Gerenciar chaves de acesso para usuários do IAM.

> **Important**
>
> Salve a chave de acesso e a chave de acesso secreta exibidas na última página do assistente de criação da chave de acesso, antes de escolher Concluído. Essas chaves são usadas para configurar o agente de mainframe.

> **Note**
>
> Salve o ARN do usuário do IAM usado para configurar um relacionamento de confiança com um perfil do IAM.

Etapa 7: criar uma função do IAM para o agente assumir

Para criar uma nova função do IAM para o agente

1. Escolha Funções no console do IAM emhttps://console.aws.amazon.com/iam.
2. Selecione Criar função.
3. Na página Selecionar entidade confiável, escolha Política de confiança personalizada para o tipo de entidade confiável.
4. Substitua a política de confiança personalizada pela seguinte e `<iam-user-arn>` substitua pelo ARN do usuário criado anteriormente.

```
{
    "Version": "2012-10-17",
    "Statement": [ {
         "Sid": "FileTransferAgent",
         "Effect": "Allow",
         "Principal": {
            "AWS": "<IAM-User-arn>"
         },
         "Action": "sts:AssumeRole"
    } ]
}
```

5. Escolha Próximo.
6. Em Adicionar permissões, filtre o nome da política que você criou anteriormente e escolha-o.
7. Escolha Próximo.
8. Dê um nome à função e escolha Criar função.

> **Note**
>
> Salve o nome da função, que será usado posteriormente para configurar o agente de mainframe.

Etapa 8: configuração do agente

Para configurar o agente de transferência de arquivos

1. Acesse `$AGENT_DIR/current-version/config`.
2. Edite o arquivo de configuração do agente `appication.properties` para adicionar uma configuração de ambientes usando o seguinte comando:

```
oedit $AGENT_DIR/current-version/config/application.properties
```

Por exemplo: .

```
agent.environments[0].account-id=<AWS_ACCOUNT_ID>
agent.environments[0].agent-role-name=<AWS_IAM_ROLE_NAME>
agent.environments[0].access-key-id=<AWS_IAM_ROLE_ACCESS_KEY>
agent.environments[0].secret-access-id=<AWS_IAM_ROLE_SECRET_KEY>
agent.environments[0].bucket-name=<AWS_S3_BUCKET_NAME>
agent.environments[0].environment-name=<AWS_REGION>
agent.environments[0].region=<AWS_REGION>
zos.complex-name=<File_Transfer_Endpoint_Name>
```

Em que:

- `AWS_ACCOUNT_ID`é o ID da AWS conta.
- `AWS_IAM_ROLE_NAME` é o nome do perfil do IAM criado em the section called “Etapa 7: criar uma função do IAM para o agente assumir”.
- `AWS_IAM_ROLE_ACCESS_KEY` é a chave de acesso do usuário do IAM criado em the section called “Etapa 6: criar um usuário do IAM com credenciais de acesso de longo prazo”.
- `AWS_IAM_ROLE_SECRET_KEY` é a chave secreta de acesso para o usuário do IAM criado em the section called “Etapa 6: criar um usuário do IAM com credenciais de acesso de longo prazo”.
- `AWS_S3_BUCKET_NAME` é o nome do bucket de transferência criado com o endpoint de transferência de dados.
- `AWS_REGION` é a região na qual você configura o agente do Transferência de Arquivos.

> **Note**
>
> Você pode fazer com que o agente de transferência de arquivos seja transferido para várias regiões e contas AWS definindo vários ambientes.

- (Opcional). `zos.complex-name`é o nome complexo que você criou ao criar um endpoint de transferência de arquivos.

> **Note**
>
> Esse campo é necessário somente se você quiser personalizar o nome complexo (cujo padrão é o nome do sysplex) que é o mesmo que você definiu ao criar seu endpoint de transferência de arquivos. Para ter mais informações, consulte the section called “Endpoints de transferência de dados”.

> **Important**
>
> Pode haver várias seções desse tipo, desde que o índice entre colchetes (`[0]`) seja incrementado para cada uma.

Reinicie o agente para que as alterações entrem em vigor.

Requisitos

1. Quando um parâmetro é adicionado ou removido, o agente deve ser interrompido e iniciado. Inicie o agente do Transferência de Arquivos usando o seguinte comando na CLI:

   ```
   /S M2AGENT
   ```

   Para interromper o agente M2, use o seguinte comando na CLI:

   ```
   /P M2AGENT
   ```

2. Você pode fazer com que o agente de transferência de arquivos seja transferido para várias regiões e contas AWS definindo vários ambientes.

> **Note**
>
> Substitua os valores pelos valores dos parâmetros que você criou e configurou anteriormente.

```
#Region 1
agent.environments[0].account-id=AWS_ACCOUNT_ID
agent.environments[0].agent-role-name=AWS_IAM_ROLE_NAME
agent.environments[0].access-key-id=AWS_IAM_ROLE_ACCESS_KEY
agent.environments[0].secret-access-id=AWS_IAM_ROLE_SECRET_KEY
agent.environments[0].bucket-name=AWS_S3_BUCKET_NAME
agent.environments[0].environment-name=AWS_REGION
agent.environments[0].region=AWS_REGION

#Region 2
agent.environments[1].account-id=AWS_ACCOUNT_ID
agent.environments[1].agent-role-name=AWS_IAM_ROLE_NAME
agent.environments[1].access-key-id=AWS_IAM_ROLE_ACCESS_KEY
agent.environments[1].secret-access-id=AWS_IAM_ROLE_SECRET_KEY
agent.environments[1].bucket-name=AWS_S3_BUCKET_NAME
agent.environments[1].environment-name=AWS_REGION
agent.environments[1].region=AWS_REGION
```

Endpoints de transferência de dados

Os endpoints de transferência de dados permitem a conectividade com o mainframe de origem e oferecem suporte à alta disponibilidade, escalabilidade e gerenciamento simplificado de agentes. Agentes individuais são instalados em LPARs do mainframe e podem ser agrupados em um endpoint de transferência de dados. Quando uma solicitação é feita para transferir um conjunto de dados, um agente no endpoint de transferência de dados cuidará dessa transferência específica. Para iniciar as transferências de dados, pelo menos um agente no endpoint de transferência de dados deve estar on-line.

Esse procedimento pressupõe que você tenha concluído as etapas em Configurar a modernização AWS do mainframe e configurado o agente do Transferência de Arquivos no mainframe de origem.

Criar um endpoint de transferência de dados

Para criar endpoints de transferência de dados para transferência de arquivos, siga estas etapas no console de modernização do AWS mainframe.

Como criar um endpoint de transferência de dados

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região para a qual você deseja transferir arquivos do seu mainframe para um bucket do Amazon S3.
3. Na página Endpoints de transferência de dados, em Transferência de arquivos, escolha Criar endpoint de transferência de dados.
4. Na página Pré-requisitos do endpoint de transferência de dados, leia todas as instruções para garantir que você tenha concluído essas etapas no mainframe de origem. Depois de confirmado, escolha Próximo.
5. Na página Configurar endpoint de transferência de dados, adicione as informações básicas sobre seu endpoint de transferência de dados.

    1. Na seção de informações básicas, insira o nome do endpoint de transferência de dados.

        > **Note**
        >
        > O nome do endpoint de transferência de dados deve corresponder ao nome do Sysplex, a menos que você especifique um nome complexo na configuração do agente.

    2. Uma descrição opcional.
    3. A chave KMS usada para criptografar o segredo.

        > **Note**
        >
        > Você deve adicionar a seguinte política baseada em recursos para o KMS para que o serviço AWS Mainframe Modernization possa ler e usar essas chaves para a criptografia/descriptografia:

        ```
        {
          "Sid" : "Enable AWS M2 Permissions",
          "Effect" : "Allow",
        ```

```
  "Principal" : {
      "Service" : "m2.amazonaws.com"
            },
   "Action" : [
       "kms:Encrypt",
       "kms:Decrypt"
              ],
   "Resource" : "*"
}
```

4. Especifique a localização do S3 para dados intermediários, que é o local intermediário do S3 em que os conjuntos de dados transferidos do mainframe são armazenados antes de serem convertidos e transferidos para o bucket do Amazon S3 de destino.

   .

   > **Note**
   >
   > É recomendável criar um novo bucket do Amazon S3 para suas tarefas de transferência. Para obter mais informações, consulte Criação de um bucket. Também é possível pesquisar seus buckets do Amazon S3 existentes escolhendo a opção Procurar no S3.

5. Depois de inserir os campos obrigatórios, escolha Próximo.
6. Na página Revisar e criar endpoint de transferência de dados, verifique se você preencheu os pré-requisitos e revise as informações básicas. Depois de confirmado, escolha Criar ponto final de transferência de dados.

Você será redirecionado para a página Visão geral dos endpoints de transferência de dados, onde poderá ver a lista de todos os endpoints de transferência de dados. Também será possível ver os endpoints de transferência de dados que estão disponíveis ou que tiveram falha.

Também será possível pesquisar endpoints de transferência de dados por nome e acessar informações adicionais para cada agente disponível.

Tarefas de transferência

As tarefas de transferência são usadas para especificar os conjuntos de dados a serem transferidos do mainframe para o Amazon S3 e permitem que você escolha as opções de conversão da página de código.

Essas instruções pressupõem que você concluiu as etapas Configurar a modernização AWS do mainframe e criouthe section called “Endpoints de transferência de dados”.

Tópicos

- Criar tarefas de transferência
- Visualizar tarefas de transferência

Criar tarefas de transferência

Para criar tarefas de transferência no File Transfer, siga estas etapas no console de modernização do AWS mainframe.

Criar uma tarefa de transferência

> **Important**
>
> Você deve ter pelo menos um endpoint de transferência de dados para criar tarefas de transferência.

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região para a qual você deseja transferir arquivos do seu mainframe para um bucket do Amazon S3.
3. Na página Transferir tarefas, você pode escolher qualquer ponto final de transferência de dados para criar tarefas de transferência.
4. Na página Criar tarefa de transferência, configure as propriedades da sua tarefa de transferência. Se você não criou nenhuma tarefa de transferência anteriormente, você pode criar a primeira escolhendo a opção Criar tarefa de transferência.
    - Nessa página, insira as informações básicas da sua tarefa de transferência, incluindo o nome, a descrição e a chave secreta da tarefa de transferência.

> **Note**
>
> - Criptografe o segredo usando a chave KMS definida com o endpoint de transferência de dados. O segredo deve conter as credenciais de mainframe necessárias para acessar conjuntos de dados no mainframe usando as chaves e. `userId` `password` Para obter mais informações, consulte o segredo do AWS Secrets Manager.
> - Você deve configurar a chave secreta com a seguinte política baseada em recursos para que o serviço de modernização do AWS mainframe possa acessá-la para realizar a tarefa de transferência de dados.
>
> ```
> {
>   "Version" : "2012-10-17",
>   "Statement" : [ {
>     "Effect" : "Allow",
>     "Principal" : {
>         "Service" : "m2.amazonaws.com"
>     },
>     "Action" : [ "secretsmanager:GetSecretValue",
>                  "secretsmanager:DescribeSecret" ],
>     "Resource" : "*"
>   } ]
> }
> ```

> **Note**
>
> O tamanho máximo atual do conjunto de dados suportado para transferência é 90 GB.

- Em seguida, selecione o local de destino do bucket Amazon S3 para o qual os conjuntos de dados de destino do mainframe serão transferidos.
- O endpoint de transferência de dados escolhido anteriormente será selecionado. Você também pode selecionar outro endpoint dentre os endpoints disponíveis.

5. Selecione Next (Próximo).
6. Na página Adicionar conjuntos de dados, insira sua consulta em Pesquisar conjuntos de dados no mainframe para pesquisar no mainframe os conjuntos de dados a serem incluídos na sua tarefa de transferência. Selecione Exibir conjuntos de dados.

Os seguintes símbolos curinga podem ser usados como parte dos critérios de pesquisa do conjunto de dados para mainframe:

- Um único asterisco (*) como qualificador (entre períodos ou após o período final) corresponde a um único qualificador nessa posição.
- Um único asterisco (*) dentro de um qualificador corresponde a zero ou mais caracteres nessa posição.
- Um asterisco duplo (**) como qualificador (entre períodos ou após o período final) corresponde a zero ou mais qualificadores nessa posição.
- Um asterisco duplo (**) dentro de um qualificador não é uma consulta válida.
- Um único sinal de porcentagem (%) corresponde a qualquer caractere alfanumérico ou nacional nessa posição. Você pode usar sinais de até oito por cento em cada qualificatória.

> **Important**
>
> Sugerimos sempre terminar seus critérios de pesquisa com um ponto seguido por um asterisco duplo (.**) e, em seguida, refinar ainda mais a pesquisa, se necessário.

Para obter mais informações sobre regras curinga, consulte Filtragem de nomes de conjuntos de dados na documentação da IBM.

7. Esses conjuntos de dados serão carregados na seção Conjuntos de dados de mainframe, onde você poderá pesquisar ou escolher um ou mais conjuntos de dados para os quais deseja configurar as conversões de páginas de código. Esses conjuntos de dados escolhidos serão exibidos na seção Conjuntos de dados adicionados.

> **Note**
>
> Você pode selecionar conjuntos de dados de várias consultas de pesquisa e adicioná-los à sua tarefa de transferência.

8. Na seção Conjuntos de dados adicionados, você precisa selecionar manualmente a página do código-fonte e a página do código-alvo para cada conjunto de dados escolhido. A página do código-fonte é o formato do conjunto de dados de origem, e a página do código de destino

é o formato do conjunto de dados de destino usado para converter os conjuntos de dados e armazená-los no bucket do Amazon S3 de destino.

9. Depois de confirmar as páginas de código-fonte e de destino, escolha Avançar.
10. Na página Revisar e criar, é possível revisar ou editar informações da tarefa de transferência.
11. Em seguida, escolha Criar tarefa de transferência.

Visualizar tarefas de transferência

Para visualizar as tarefas de transferência no File Transfer, você deve seguir estas etapas no console de modernização do AWS mainframe.

Como visualizar tarefas de transferência

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. No Região da AWS seletor, escolha a região para a qual você deseja transferir arquivos do seu mainframe para um bucket do Amazon S3.
3. Na página Transferir tarefas, selecione o endpoint de transferência de dados para visualizar suas tarefas de transferência.
4. Para endpoints que têm tarefas de transferência preexistentes, elas serão exibidas na seção Transferir tarefas. Você pode optar por visualizar os detalhes de qualquer tarefa de transferência nessa lista.

Tutorial: Conceitos básicos do Transferência de Arquivos do AWS Mainframe Modernization

O AWS Mainframe Modernization File Transfer permite transferir e converter conjuntos de dados de mainframe para casos de uso de modernização, migração e aumento de mainframe.

Siga as etapas deste tutorial para entender como o Transferência de Arquivos do AWS Mainframe Modernization funciona.

Visão geral

O Transferência de Arquivos consiste no seguinte:

1. Um agente a ser instalado no mainframe de origem.

2. Acesso aos recursos de descoberta, transferência e conversão de conjuntos de dados diretamente do console do serviço de gerenciamento de modernização do AWS mainframe.

Como usuário, você pode transferir conjuntos de dados do mainframe para o bucket do Amazon S3.

Tópicos

Etapa 1: Transferir o pacote tar dos binários do agente AWS para a partição lógica do mainframe

Baixe os arquivos tar do link tar do M2-Agent.

Etapa 2: Configurar o agente do Transferência de Arquivos no mainframe de origem

Nesta etapa, você configura e inicia o agente do Transferência de Arquivos do AWS Mainframe Modernization no mainframe de origem. O agente deve facilitar a comunicação entre o recurso do serviço Transferência de Arquivos e o mainframe de origem. É necessário pelo menos um agente por mainframe. Mais de um agente pode ser iniciado para obter alta disponibilidade e escalabilidade aprimorada.

Siga as instruções no guia the section called “Instalar um agente do Transferência de Arquivos” para concluir a instalação do agente do Transferência de Arquivos no mainframe.

Etapa 3: Criar um endpoint de transferência de dados

Siga as etapas na página the section called “Endpoints de transferência de dados” para criar um endpoint de transferência de dados.

Etapa 4: Criar uma tarefa de transferência

Siga as etapas na página the section called “Tarefas de transferência” para criar e gerenciar suas tarefas de transferência.

Etapa 5: Visualizar o progresso da tarefa de transferência

Você pode ver o progresso da sua tarefa de transferência no console de modernização do AWS mainframe. Para obter mais detalhes, consulte a seção the section called “Visualizar tarefas de transferência”.

O AWS Mainframe Modernization File Transfer suportou as codificações de origem e destino

O AWS Mainframe Modernization File Transfer oferece suporte a vários tipos de conjuntos de dados e opções de conversão de páginas de código.

Tipos de conjuntos de dados de mainframe

O AWS Mainframe Modernization File Transfer oferece suporte aos seguintes tipos de conjuntos de dados de mainframe:

- Não VSAM: Sequencial (PS), PDS, GDS, GDG
- Tipos de VSAM: KSDS

Páginas de código suportadas

O AWS Mainframe Modernization File Transfer oferece suporte às seguintes páginas de código para conversão de conjuntos de dados (de/para):

“BIG5", “BIG5_HKSCS”, “CESU_8", “EUC_JP”, “EUC_KR”, “GB18030", “GB2312", “GBK”, “IBM00858", “IBM01140", “IBM01141", “IBM01142", “IBM01143", “IBM01143", “IBM01143" 144", “IBM01145", “IBM01146", “IBM01147", “IBM01148", “IBM01149", “IBM037", “IBM1026", “IBM1047", “IBM273", “IBM277", “IBM278", “IBM280", ““IBM284", “IBM285", “IBM290", “IBM297", “IBM420", “IBM424", “IBM437", “IBM500", “IBM775", “IBM850", “IBM852", “IBM855", “IBM857", “IBM860", “IBM861", “IBM862", “IBM863", “IBM864", “IBM865", “IBM866", “IBM868", “IBM869", “IBM870", “IBM871" , “IBM918", “IBM_THAI”, “ISO_2022_CN”, “ISO_2022_JP”, “ISO_2022_JP_2",

“ISO_2022_KR”, “ISO_8859_1", “ISO_8859_13", “ISO_8859_15", “ISO_8859_16", “ISO_8859_16" _2", “ISO_8859_3", “ISO_8859_4", “ISO_8859_5", “ISO_8859_6", “ISO_8859_7", “ISO_8859_8", “ISO_8859_9", "JIS_X0201", “JIS_X0212_1990", “KOI8_R”, “KOI8_U”, “SHIFT_JIS”, “TIS_620", “US_ASCII”, “UTF_16", “UTF_16BE”, “UTF_16LE”, “UTF_32", “UTF_32BE”, “UTF_32LE”, “UTF_8", “WINDOWS_1250", “WINDOWS_1250" _1251", “WINDOWS_1252", “WINDOWS_1253", "WINDOWS_1254", “WINDOWS_1255", “WINDOWS_1256", “WINDOWS_1257", “WINDOWS_1258", “WINDOWS_31J”, “X_BIG5_HKSCS_2001", “X_BIG5_SOLARIS”, “X_EUCJP_OPEN”, “X_EUC_JP_LINUX”, “X_EUC_TW”, “X_IBM1006", “X_IBM1006", “X_IBM1006" 1025", “X_IBM1046", “X_IBM1097", “X_IBM1098", “X_IBM1112", “X_IBM1122", “X_IBM1123", “X_IBM1124", “X_IBM1129", “X_IBM1166", “X_IBM136" 64", “X_IBM1381", “X_IBM1383", “X_IBM29626C”, “X_IBM300", “X_IBM33722", “X_IBM737", “X_IBM833", “X_IBM834", “X_IBM856", “X_IBM874", “X_IBM875", “X_IBM921", “X_IBM922", “X_IBM930", “X_IBM933", “X_IBM935", “X_IBM937", “X_IBM939", “X_IBM942", “X_IBM 942C”, “X_IBM943", “X_IBM943C”, “X_IBM948", “X_IBM949", “X_IBM949C”, “X_IBM950", “X_IBM964", “X_IBM970", “X_ISCII91", “X_ISO_2020" 22_CN_CNS”, “X_ISO_2022_CN_GB”, “X_ISO_8859_11", “X_JIS0208", “X_JISAUTODETECT”, “X_JOHAB”, “X_MACARABIC”, “X_MACCENTRALEUROPE”, “X_MACCROATIAN”, “X_MACCYRILLIC”, “X_MACDINGBAT”, “X_MACGREEK”, “X_MACHEBREW”, “X_MACICELAND”, “X_MACROMAN”, “X_MACROMANIA”, “X_MACSYMBOL”, “X_MACTHAI”, “X_MACTURKISH”, “X_MACUKRAINE”, “X_MS932_0213", “X_MS950_HKSCS”, “X_MS950_HKSCS_XP”, “X_MSWIN_936", “X_PCK”, “X_SJIS_0213", “X_ UTF_16LE_BOM”, “X_UTF_32BE_BOM”, “X_UTF_32LE_BOM”, “X_WINDOWS_50220", “X_WINDOWS_50221", “X_WINDOWS_874", “X_WINDOWS_949", “X_WINDOWS_950", “X_WINDOWS_ISO2022J”

Segurança na modernização AWS do mainframe

A segurança na nuvem AWS é a maior prioridade. Como AWS cliente, você se beneficia de uma arquitetura de data center e rede criada para atender aos requisitos das organizações mais sensíveis à segurança.

A segurança é uma responsabilidade compartilhada entre você AWS e você. O modelo de responsabilidade compartilhada descreve isso como a segurança da nuvem e a segurança na nuvem:

- Segurança da nuvem — AWS é responsável por proteger a infraestrutura que executa AWS os serviços na AWS nuvem. AWS também fornece serviços que você pode usar com segurança. Auditores terceirizados testam e verificam regularmente a eficácia de nossa segurança como parte dos Programas de Conformidade Programas de AWS de . Para saber mais sobre os programas de conformidade que se aplicam à modernização do AWS mainframe, consulte AWS Services in Scope by Compliance Program AWS Program.
- Segurança na nuvem — Sua responsabilidade é determinada pelo AWS serviço que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade dos dados, os requisitos da empresa e as leis e os regulamentos aplicáveis

Essa documentação ajuda você a entender como aplicar o modelo de responsabilidade compartilhada ao usar a modernização do AWS mainframe. Ele mostra como configurar a modernização do AWS mainframe para atender aos seus objetivos de segurança e conformidade. Você também aprenderá a usar outros AWS serviços que ajudam a monitorar e proteger seus recursos de modernização de AWS mainframe.

AWS A modernização do mainframe fornece seus próprios recursos protegidos pelo IAM (aplicativo, ambiente, implantação etc.), que são os recursos administrativos da modernização do AWS mainframe, nos quais qualquer ação deve ser permitida pelas políticas do IAM.

AWS A modernização do mainframe para reformulação de plataformas também é garantida pelo IAM. O IAM concede ou nega permissão a um diretor para uma ação específica em um recurso definido, derivado do ambiente original do mainframe, também por meio de políticas padrão do IAM. O tempo de execução da replataforma da modernização do AWS mainframe chama o serviço de autorização do IAM quando um aplicativo tenta tal ação em um recurso protegido. O IAM retornará permitir ou negar com base nos mecanismos padrão de avaliação de políticas do IAM.

Conteúdo

Proteção de dados na modernização AWS do mainframe

O modelo de responsabilidade AWS compartilhada O modelo se aplica à proteção de dados na modernização do AWS mainframe. Conforme descrito neste modelo, AWS é responsável por proteger a infraestrutura global que executa todos os Nuvem AWS. Você é responsável por manter o controle sobre seu conteúdo hospedado nessa infraestrutura. Você também é responsável pelas tarefas de configuração e gerenciamento de segurança dos Serviços da AWS que usa. Para obter mais informações sobre a privacidade de dados, consulte as Perguntas Frequentes sobre Privacidade de Dados.. Para obter mais informações sobre a proteção de dados na Europa, consulte a postagem do blog AWS Shared Responsibility Model and GDPR no Blog de segurança da AWS .

Para fins de proteção de dados, recomendamos que você proteja Conta da AWS as credenciais e configure usuários individuais com AWS IAM Identity Center ou AWS Identity and Access Management (IAM). Dessa maneira, cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com os recursos. AWS Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure a API e o registro de atividades do usuário com AWS CloudTrail.
- Use soluções de AWS criptografia, juntamente com todos os controles de segurança padrão Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sigilosos armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-2 ao acessar AWS por meio de uma interface de linha de comando ou de uma API, use um endpoint FIPS. Para ter

mais informações sobre endpoints do FIPS disponíveis, consulte Federal Information Processing Standard (FIPS) 140-2.

É altamente recomendável que nunca sejam colocadas informações de identificação confidenciais, como endereços de e-mail dos seus clientes, em marcações ou campos de formato livre, como um campo Nome. Isso inclui quando você trabalha com a modernização de AWS mainframe ou outros Serviços da AWS usando o console, a API ou AWS os AWS CLI SDKs. Quaisquer dados inseridos em tags ou campos de texto de formato livre usados para nomes podem ser usados para logs de faturamento ou de diagnóstico. Se você fornecer um URL para um servidor externo, recomendemos fortemente que não sejam incluídas informações de credenciais no URL para validar a solicitação a esse servidor.

Dados que a modernização AWS do mainframe coleta

AWS A modernização do mainframe coleta vários tipos de dados de você:

- `Application configuration`: é um arquivo JSON que você cria para configurar seu aplicativo. Ele contém suas opções para as diferentes opções que a modernização do AWS mainframe oferece. O arquivo também contém informações sobre AWS recursos dependentes, como os caminhos do Amazon Simple Storage Service, nos quais os artefatos do aplicativo são armazenados, ou o Amazon Resource Name (ARN) onde as credenciais do banco AWS Secrets Manager de dados são armazenadas.
- `Application executable (binary)`: esse é um binário que você compila e pretende implantar na modernização do AWS mainframe.
- `Application JCL or scripts`: esse código-fonte gerencia trabalhos em lotes ou outros processamentos em nome do seu aplicativo.
- `User application data`: quando você importa conjuntos de dados, a modernização do AWS mainframe os armazena no banco de dados relacional para que seu aplicativo possa acessá-los.
- `Application source code`: Por meio do Amazon AppStream 2.0, a modernização do AWS mainframe fornece um ambiente de desenvolvimento para você escrever e compilar código.

AWS A modernização do mainframe armazena esses dados nativamente em. AWS Os dados que coletamos de você são armazenados em um AWS Mainframe Modernization-managed Amazon S3 bucket. Quando você implanta um aplicativo, a modernização do AWS mainframe baixa os dados em uma instância do Amazon Elastic Compute Cloud baseada na Amazon Elastic Block Store. Quando

a limpeza é acionada, os dados são removidos do volume do Amazon EBS e do Amazon S3. Os volumes do Amazon EBS são de inquilino único, o que significa que uma instância é usada para um cliente. As instâncias nunca são compartilhadas. Quando você Guia do usuário ambiente de runtime, o volume do Amazon EBS também é excluído. Quando você exclui uma aplicação, os artefatos e a configuração são excluídos do Amazon S3.

Os registros do aplicativo são armazenados na Amazon CloudWatch. As mensagens de registro do aplicativo do cliente também são CloudWatch exportadas para. Os CloudWatch registros podem conter dados confidenciais do cliente, como dados comerciais ou informações de segurança em mensagens de depuração). Para ter mais informações, consulte Monitorando a modernização AWS do mainframe com a Amazon CloudWatch.

Além disso, se você optar por anexar um ou mais sistemas de arquivos Amazon Elastic File System ou Amazon FSx ao seu ambiente de runtime, os dados dentro desses sistemas serão armazenados na AWS. Você precisará limpar esses dados se decidir parar de usar os sistemas de arquivos.

Você pode usar todas as opções de criptografia do Amazon S3 disponíveis para proteger seus dados ao colocá-los no bucket do Amazon S3 AWS que o Mainframe Modernization usa para implantação de aplicativos e importações de conjuntos de dados. Além disso, você pode usar as opções de criptografia do Amazon EFS e do Amazon FSx se anexar um ou mais desses sistemas de arquivos ao seu ambiente de runtime.

Criptografia de dados em repouso para o serviço de modernização AWS de mainframe

AWS A modernização do mainframe se integra AWS Key Management Service para fornecer criptografia transparente do lado do servidor (SSE) em todos os recursos dependentes que armazenam dados permanentemente, ou seja, Amazon Simple Storage Service, Amazon DynamoDB e Amazon Elastic Block Store. AWS A modernização do mainframe cria e gerencia AWS KMS chaves de criptografia simétricas para você em. AWS KMS

A criptografia de dados em repouso por padrão ajuda a reduzir a sobrecarga operacional e a complexidade envolvidas na proteção de dados confidenciais. Ao mesmo tempo, ele permite que você migre aplicações que exigem rigorosa conformidade com criptografia e requisitos regulatórios.

Você não pode desativar essa camada de criptografia nem selecionar um tipo de criptografia alternativo ao criar ambientes e aplicativos de tempo de execução.

Você pode usar sua própria chave gerenciada pelo cliente para aplicativos de modernização de AWS mainframe e ambientes de execução para criptografar os recursos do Amazon S3 e do Amazon EBS.

Para seus aplicativos de modernização de AWS mainframe, você pode usar essa chave para criptografar a definição do seu aplicativo, bem como outros recursos do aplicativo, como arquivos JCL, que são salvos no bucket do Amazon S3 criado na conta do serviço. Para ter mais informações, consulte Cria uma aplicação .

Para seus ambientes de execução de modernização de AWS mainframe, a modernização de AWS mainframe usa sua chave gerenciada pelo cliente para criptografar o volume do Amazon EBS que ele cria e anexa à sua instância AWS Amazon EC2 de modernização de mainframe, que também está na conta do serviço. Para ter mais informações, consulte Criar um ambiente de runtime.

> **Note**
>
> Os recursos do DynamoDB são sempre criptografados usando Chave gerenciada pela AWS uma conta de serviço na modernização AWS do mainframe. Não é possível criptografar recursos do DynamoDB usando uma chave gerenciada pelo cliente.

AWS A modernização do mainframe usa sua chave gerenciada pelo cliente para as seguintes tarefas:

- Implantação de uma aplicação.
- Substituindo uma instância de modernização de AWS mainframe do Amazon EC2.

AWS A modernização do mainframe não usa sua chave gerenciada pelo cliente para criptografar bancos de dados Amazon Relational Database Service ou Amazon Aurora, filas do Amazon Simple Queue Service e caches da ElastiCache Amazon criados para dar suporte a AWS um aplicativo de modernização de mainframe, porque nenhum deles contém dados do cliente.

Para obter mais informação, consulte Chaves gerenciadas pelo cliente no Guia do desenvolvedor do AWS Key Management Service .

A tabela a seguir resume como a modernização do AWS mainframe criptografa seus dados confidenciais.

Tipo de dados	Chave gerenciada pela AWS criptografia	Criptografia de chave gerenciada pelo cliente
`Definition`	Habilitado	Habilitado

Tipo de dados	Chave gerenciada pela AWS criptografia	Criptografia de chave gerenciada pelo cliente
Contém a definição de uma aplicação específica.		
`EnvironmentSummary` Contém informações sobre o ambiente de runtime.	Habilitado	Habilitado
`ApplicationSummary` Contém informações sobre o aplicativo de modernização do AWS mainframe.	Habilitado	Habilitado
`DeploymentSummary` Contém informações sobre a implantação de um aplicativo de modernização de AWS mainframe.	Habilitado	Habilitado

> **Note**
>
> AWS A modernização do mainframe ativa automaticamente a criptografia em repouso, usando Chaves gerenciadas pela AWS para proteger seus dados confidenciais sem nenhum custo. No entanto, AWS KMS cobranças são aplicadas pelo uso de uma chave gerenciada pelo cliente. Para obter mais informações sobre precificação, consulte Precificação do AWS Key Management Service.

Para obter mais informações sobre AWS KMS, consulte AWS Key Management Service.

Como a modernização AWS do mainframe usa subsídios em AWS KMS

AWS A modernização do mainframe exige uma concessão para usar sua chave gerenciada pelo cliente.

Quando você cria um aplicativo ou ambiente de execução, ou implanta um aplicativo na Modernização do AWS Mainframe criptografado com uma chave gerenciada pelo cliente, a Modernização do AWS Mainframe cria uma concessão em seu nome enviando uma solicitação para. CreateGrant AWS KMS As concessões AWS KMS são usadas para dar à modernização do AWS mainframe acesso a uma chave KMS em uma conta de cliente.

AWS A modernização do mainframe exige que a concessão use sua chave gerenciada pelo cliente para as seguintes operações internas:

- Envie DescribeKeysolicitações AWS KMS para verificar se a ID simétrica da chave gerenciada pelo cliente inserida ao criar um aplicativo, ambiente de execução ou implantação de aplicativo é válida.
- Envie GenerateDataKeysolicitações para AWS KMS criptografar o volume do Amazon EBS anexado às instâncias do Amazon EC2 que AWS hospedam ambientes de execução de modernização de mainframe.
- Envie solicitações de descriptografia para AWS KMS descriptografar conteúdo criptografado no Amazon EBS.

AWS A modernização do mainframe usa AWS KMS concessões para decifrar seus segredos armazenados no Secrets Manager e ao criar um ambiente de tempo de execução, criar ou reimplantar um aplicativo e criar uma implantação. Os subsídios criados pela modernização do AWS mainframe dão suporte às seguintes operações:

- Criar ou atualizar uma concessão de ambiente de runtime:
  - Decrypt
  - Encrypt
  - ReEncryptFrom
  - ReEncryptTo
  - GenerateDataKey
  - DescribeKey
  - CreateGrant
- Crie ou reimplante uma concessão de aplicação:
  - GenerateDataKey
- Crie uma concessão de implantação:
  - Decrypt

É possível revogar o acesso à concessão, ou remover o acesso do serviço à chave gerenciada pelo cliente a qualquer momento. Se você fizer isso, a modernização do AWS mainframe não poderá acessar nenhum dado criptografado pela chave gerenciada pelo cliente, o que afeta as operações que dependem dos dados. Por exemplo, se a modernização do AWS mainframe tentasse acessar uma definição de aplicativo criptografada por uma chave gerenciada pelo cliente sem a concessão dessa chave, a operação de criação do aplicativo falharia.

AWS A modernização do mainframe coleta configurações de aplicativos do usuário (arquivos JSON) e artefatos (binários e executáveis). Ele também cria metadados que rastreiam várias entidades usadas para a operação do AWS Mainframe Modernization e cria logs e métricas. Os logs e métricas que são visíveis para o cliente incluem:

- CloudWatch registros que refletem o aplicativo e o mecanismo de tempo de execução (AWS Blu Age ou Micro Focus).
- CloudWatch métricas para painéis de operação.

Além disso, a modernização do AWS mainframe coleta dados e métricas de uso para medição, relatórios de atividades e assim por diante sobre os serviços. Esses dados não são visíveis para o cliente.

AWS A modernização do mainframe armazena esses dados em locais diferentes, dependendo do tipo de dados. Os dados do cliente que você carrega são armazenados em um bucket do Amazon S3. Os dados do serviço são armazenados no Amazon S3 e no DynamoDB. Quando você implanta uma aplicação, tanto os dados quanto os dados do serviço são baixados nos volumes do Amazon EBS. Se você optar por conectar o armazenamento Amazon EFS ou Amazon FSx ao seu ambiente de runtime, os dados armazenados nesses sistemas de arquivos também serão baixados para o volume do Amazon EBS.

A criptografia em repouso é configurada por padrão. Você não pode desativá-lo ou alterá-lo. No momento, também não é possível alterar a configuração.

Criar uma chave gerenciada pelo cliente

Você pode criar uma chave simétrica gerenciada pelo cliente usando as AWS Management Console ou as AWS KMS APIs.

Para criar uma chave simétrica gerenciada pelo cliente

Siga as etapas de Criar uma chave simétrica gerenciada pelo cliente no Guia do desenvolvedor do AWS Key Management Service .

Política de chave

As políticas de chaves controlam o acesso à chave gerenciada pelo seu cliente. Cada chave gerenciada pelo cliente deve ter exatamente uma política de chaves, que contém declarações que determinam quem pode usar a chave e como pode usá-la. Ao criar a chave gerenciada pelo cliente, você pode especificar uma política de chaves. Para obter mais informações, consulte Managing access to customer managed keys (Administrando o acesso a chaves gerenciadas pelo cliente) no Guia do desenvolvedor do AWS Key Management Service .

Para usar sua chave gerenciada pelo cliente com seus recursos de modernização de AWS mainframe, as seguintes operações de API devem ser permitidas na política de chaves:

- `kms:CreateGrant`: Adiciona uma concessão a uma chave gerenciada pelo cliente. Concede acesso de controle a uma chave KMS especificada, o que permite o acesso às operações de concessão exigidas pela modernização do AWS mainframe. Para obter mais informações sobre Utilizar concessões, consulte o Guia do desenvolvedor do AWS Key Management Service .

  Isso permite que a modernização do AWS mainframe faça o seguinte:

  - Ligar para `GenerateDataKey` para gerar uma chave de dados criptografada e armazená-la, porque a chave de dados não é usada imediatamente para criptografar.
  - Ligar para `Decrypt` para usar a chave de dados criptografada armazenada para acessar os dados criptografados.
  - Configure uma entidade principal aposentada para permitir que o serviço para `RetireGrant`.
- `kms:DescribeKey`— fornece os principais detalhes gerenciados pelo cliente para permitir que a modernização do AWS mainframe valide a chave.

AWS A modernização do mainframe exige `kms:CreateGrant` `kms:DescribeKey` permissões na política principal do cliente. AWS A modernização do mainframe usa essa política para criar uma concessão para si mesma.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "Enable IAM User Permissions",
        "Effect": "Allow",
```

```
        "Principal": {
            "AWS": "arn:aws:iam::AccountId:role/ExampleRole"
        },
        "Action": [
            "kms:CreateGrant",
            "kms:DescribeKey"
        ],
        "Resource": "*"
    }]
}
```

> **Note**
>
> A função mostrada `Principal` no exemplo anterior é aquela que você usa para operações de modernização de AWS mainframe, como e. `CreateApplication` `CreateEnvironment`

Para obter mais informações sobre specifying permissions in a policy (especificações de permissões em uma política), consulte o Guia do desenvolvedor do AWS Key Management Service .

Para obter mais informações sobre solução de problemas de acesso à chave, consulte o Guia do Desenvolvedor do AWS Key Management Service .

Especificação de uma chave gerenciada pelo cliente para o AWS Mainframe Modernization

Você pode especificar uma chave gerenciada pelo cliente para os seguintes recursos:

- Aplicativo
- Ambiente

Ao criar um recurso, você pode especificar a chave inserindo uma ID do KMS, que a Modernização do AWS Mainframe usa para criptografar os dados confidenciais armazenados pelo recurso.

- KMS ID - Um identificador de chave para uma chave gerenciada pelo cliente. Insira uma ID de chave, um ARN de chave, um nome de alias ou um ARN de alias.

Você pode especificar uma chave gerenciada pelo cliente usando o AWS Management Console ou AWS CLI o.

Para especificar sua chave gerenciada pelo cliente ao criar um ambiente de tempo de execução no AWS Management Console, consulteCrie um ambiente de tempo de execução de modernização de AWS mainframe. Para especificar sua chave gerenciada pelo cliente ao criar um aplicativo no AWS Management Console, consulteCrie um aplicativo de modernização AWS de mainframe.

Para adicionar sua chave gerenciada pelo cliente ao criar um ambiente de tempo de execução com o AWS CLI, especifique o `kms-key-id` parâmetro da seguinte forma:

```
aws m2 create-environment –engine-type microfocus –instance-type M2.m5.large
--publicly-accessible –engine-version 7.0.3 –name test
--high-availability-config desiredCapacity=2
--kms-key-id myEnvironmentKey
```

Para adicionar sua chave gerenciada pelo cliente ao criar um aplicativo com o AWS CLI, especifique o `kms-key-id` parâmetro da seguinte forma:

```
aws m2 create-application –name test-application –description my description
--engine-type microfocus
--definition content="$(jq -c . raw-template.json | jq -R)"
--kms-key-id myApplicationKey
```

AWS Contexto de criptografia da modernização do mainframe

Um contexto de criptografia é um conjunto opcional de pares chave-valor que pode conter informações contextuais adicionais sobre os dados.

AWS KMS usa o contexto de criptografia como dados autenticados adicionais para oferecer suporte à criptografia autenticada. Quando você inclui um contexto de criptografia em uma solicitação para criptografar dados, AWS KMS vincula o contexto de criptografia aos dados criptografados. Para descriptografar os dados, você inclui o mesmo contexto de criptografia na solicitação.

AWS Contexto de criptografia da modernização do mainframe

AWS A modernização do mainframe usa o mesmo contexto de criptografia em todas as operações AWS KMS criptográficas relacionadas a um aplicativo (criar aplicativo e criar implantação), onde a chave está `aws:m2:app` e o valor é o identificador exclusivo do aplicativo.

Example

```
"encryptionContextSubset": {
        "aws:m2:app": "a1bc2defabc3defabc4defabcd"
}
```

Uso do contexto de criptografia para monitoramento

Quando você usa uma chave simétrica gerenciada pelo cliente para criptografar suas aplicações ou ambientes de runtime, também pode usar o contexto de criptografia em registros e logs de auditoria para identificar como a chave gerenciada pelo cliente está sendo usada.

Uso do contexto de criptografia para controlar o acesso à chave gerenciada pelo cliente

Você pode usar o contexto de criptografia nas políticas de chaves e políticas do IAM como `conditions` e controlar o acesso à sua chave simétrica gerenciada pelo cliente. Você também pode usar restrições no contexto de criptografia em uma concessão.

AWS A modernização do mainframe usa uma restrição de contexto de criptografia nas concessões para controlar o acesso à chave gerenciada pelo cliente em sua conta ou região. A restrição de concessão exige que as operações permitidas pela concessão usem o contexto de criptografia especificado. O exemplo a seguir é uma concessão que a modernização do AWS mainframe aproveita para criptografar o artefato do aplicativo ao criar um aplicativo.

```
//This grant is retired immediately after create application finish
{
   "grantee-principal": m2.us-west-2.amazonaws.com,
   "retiring-principal": m2.us-west-2.amazonaws.com,
   "operations": [
       "GenerateDataKey"
   ]
   "condition": {
        "encryptionContextSubset": {
            “aws:m2:app”: “a1bc2defabc3defabc4defabcd”
   }
}
```

Monitoramento de suas chaves de criptografia para AWS Mainframe Modernization

Ao usar uma chave gerenciada pelo AWS KMS cliente com seus recursos de modernização de AWS mainframe, você pode usar o AWS CloudTrailAmazon CloudWatch Logs para rastrear solicitações enviadas pela modernização de AWS mainframe. AWS KMS

Exemplos de ambientes de runtime

Os exemplos a seguir são AWS CloudTrail eventos para`DescribeKey`, `CreateGrantGenerateDataKey`, e `Decrypt` para monitorar as operações do KMS chamadas pela modernização do AWS mainframe para acessar dados criptografados pela chave gerenciada pelo cliente:

DescribeKey

AWS A modernização do mainframe usa a `DescribeKey` operação para verificar se a chave gerenciada pelo AWS KMS cliente associada ao seu ambiente de tempo de execução existe na conta e na região.

O evento de exemplo a seguir registra a operação `DescribeKey`:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T19:40:26Z",
                "mfaAuthenticated": "false"
```

```
            }
        }
    },
    "eventTime": "2022-12-06T20:23:43Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DescribeKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "205.251.233.182",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "keyId": "00dd0db0-0000-0000-ac00-b0c000SAMPLE"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```

CreateGrant

Quando você usa uma chave gerenciada pelo AWS KMS cliente para criptografar seu ambiente de execução, a modernização do AWS mainframe envia várias `CreateGrant` solicitações em seu nome para realizar as operações necessárias do KMS. Algumas das concessões criadas pela modernização do AWS mainframe são retiradas imediatamente após o uso. Outros são retirados quando você exclui o ambiente de runtime.

O evento de exemplo a seguir registra a operação `CreateGrant` da função de execução do Lambda associada ao fluxo de trabalho Create Environment.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T20:11:45Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T20:23:09Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "operations": [
            "Encrypt",
            "Decrypt",
            "ReEncryptFrom",
            "ReEncryptTo",
            "GenerateDataKey",
            "GenerateDataKey",
```

```
            "DescribeKey",
            "CreateGrant"
        ],
        "granteePrincipal": "m2.us-west-2.amazonaws.com",
        "retiringPrincipal": "m2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

O exemplo de evento a seguir registra a operação `CreateGrant` para o perfil vinculado ao serviço do grupo do Auto Scaling. A função de execução do Lambda associada ao fluxo de trabalho Create Environment chama essa operação `CreateGrant`. Ele concede permissão para que o perfil de execução crie uma subconcessão em relação ao perfil vinculado ao serviço do grupo do Auto Scaling.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65MZFUPM4JO:EnvironmentWorkflow-alpha-
CreateEnvironmentLambda7-HfxDj5zz86tr",
```

```
        "arn": "arn:aws:sts::111122223333:assumed-role/EnvironmentWorkflow-
alpha-CreateEnvironmentLambdaS-1AU4A8VNQEEKN/EnvironmentWorkflow-alpha-
CreateEnvironmentLambda7-HfxDj5zz86tr",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/EnvironmentWorkflow-alpha-
CreateEnvironmentLambdaS-1AU4A8VNQEEKN",
                "accountId": "111122223333",
                "userName": "EnvironmentWorkflow-alpha-
CreateEnvironmentLambdaS-1AU4A8VNQEEKN"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T20:22:28Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-12-06T20:23:09Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "54.148.236.160",
    "userAgent": "aws-sdk-java/2.18.21 Linux/4.14.255-276-224.499.amzn2.x86_64
 OpenJDK_64-Bit_Server_VM/11.0.14.1+10-LTS Java/11.0.14.1 vendor/Amazon.com_Inc. md/
internal exec-env/AWS_Lambda_java11 io/sync http/Apache cfg/retry-mode/legacy",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "operations": [
            "Encrypt",
            "Decrypt",
            "ReEncryptFrom",
            "ReEncryptTo",
            "GenerateDataKey",
            "GenerateDataKey",
            "DescribeKey",
            "CreateGrant"
        ],
        "granteePrincipal": "m2.us-west-2.amazonaws.com",
```

```
        "retiringPrincipal": "m2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    }
}
}
```

GenerateDataKey

Quando você habilita uma chave gerenciada pelo AWS KMS cliente para seu recurso de ambiente de tempo de execução, o Auto Scaling cria uma chave exclusiva para criptografar o volume do Amazon EBS associado ao ambiente de tempo de execução. Ele envia uma `GenerateDataKey` solicitação AWS KMS que especifica a chave gerenciada pelo AWS KMS cliente para o recurso.

O evento de exemplo a seguir registra a operação `GenerateDataKey`:

```
{
    "eventVersion": "1.08",
```

```
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65EEXVIEH7D:AutoScaling",
        "arn": "arn:aws:sts::111122223333:assumed-role/AWSServiceRoleForAutoScaling/
AutoScaling",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling",
                "accountId": "111122223333",
                "userName": "AWSServiceRoleForAutoScaling"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T20:23:16Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "autoscaling.amazonaws.com"
    },
    "eventTime": "2022-12-06T20:23:18Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "autoscaling.amazonaws.com",
    "userAgent": "autoscaling.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "aws:ebs:id": "vol-080f7a32d290807f3"
        },
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "numberOfBytes": 64
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
```

```
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Decrypt

Quando você acessa um ambiente de runtime criptografado, o Amazon EBS chama a operação `Decrypt` para usar a chave de dados criptografados armazenada para acessar os dados criptografados.

O evento de exemplo a seguir registra a operação `Decrypt`:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ebs.amazonaws.com"
    },
    "eventTime": "2022-12-06T20:23:22Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ebs.amazonaws.com",
    "userAgent": "ebs.amazonaws.com",
    "requestParameters": {
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "encryptionContext": {
            "aws:ebs:id": "vol-080f7a32d290807f3"
        }
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
```

```
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "sharedEventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventCategory": "Management"
}
```

Exemplos de aplicações

Os exemplos a seguir são AWS CloudTrail eventos para `CreateGrant` e `GenerateDataKey` para monitorar as operações do KMS chamadas pela modernização do AWS mainframe para acessar dados criptografados pela chave gerenciada pelo cliente:

CreateGrant

Quando você usa uma chave gerenciada pelo AWS KMS cliente para criptografar os recursos do seu aplicativo, a função de execução do Lambda envia `CreateGrant` uma solicitação em seu nome para acessar a chave KMS em sua conta. AWS A concessão permite que a função de execução do Lambda carregue recursos de aplicações do cliente para o Amazon S3 usando sua chave gerenciada pelo cliente. Esse subsídio é retirado imediatamente após a criação da aplicação.

O evento de exemplo a seguir registra a operação `CreateGrant`:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
```

```
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T21:51:45Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T22:47:04Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "constraints": {
            "encryptionContextSubset": {
                "aws:m2:app": "a1bc2defabc3defabc4defabcd"
            }
        },
        "retiringPrincipal": "m2.us-west-2.amazonaws.com",
        "operations": [
            "GenerateDataKey"
        ],
        "granteePrincipal": "m2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
```

```
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

GenerateDataKey

Quando você habilita uma chave gerenciada pelo AWS KMS cliente para seu recurso de aplicativo, a função de execução do Lambda cria uma chave que é usada para criptografar e carregar dados do cliente para o Amazon Simple Storage Service. A função de execução do Lambda envia uma `GenerateDataKey` solicitação AWS KMS que especifica a chave gerenciada pelo AWS KMS cliente para o recurso.

O evento de exemplo a seguir registra a operação `GenerateDataKey`:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65CLCEKKC7Z:ApplicationWorkflow-alpha-
CreateApplicationVersion-CstWZUn5R4u6",
        "arn": "arn:aws:sts::111122223333:assumed-role/ApplicationWorkflow-
alpha-CreateApplicationVersion-1IZRBZYDG20B/ApplicationWorkflow-alpha-
CreateApplicationVersion-CstWZUn5R4u6",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/ApplicationWorkflow-alpha-
CreateApplicationVersion-1IZRBZYDG20B",
                "accountId": "111122223333",
                "userName": "ApplicationWorkflow-alpha-
CreateApplicationVersion-1IZRBZYDG20B"
```

```
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T23:28:32Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T23:29:08Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "aws:m2:app": "a1bc2defabc3defabc4defabcd",
            "aws:s3:arn": "arn:aws:s3:::supernova-processedtemplate-111122223333-us-
west-2/111122223333/a1bc2defabc3defabc4defabcd/1/cics-transaction/ZBNKE35.so"
        },
        "keySpec": "AES_256",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Exemplos de implantações

Os exemplos a seguir são AWS CloudTrail eventos para `CreateGrant` e `Decrypt` para monitorar as operações do KMS chamadas pela modernização do AWS mainframe para acessar dados criptografados pela chave gerenciada pelo cliente:

CreateGrant

Quando você usa uma chave gerenciada pelo AWS KMS cliente para criptografar seus recursos de implantação, a modernização do AWS mainframe envia duas `CreateGrant` solicitações em seu nome. A primeira concessão é atribuída à função de execução atual do Lambda a ser chamada ListBatchJobScriptFiles e é retirada imediatamente após o término da implantação. A segunda concessão é destinada ao perfil de instância com escopo reduzido do Amazon EC2, para que o Amazon EC2 possa baixar recursos de aplicações de clientes do Amazon S3. Essa concessão é retirada quando a aplicação é excluída do ambiente de runtime.

O evento de exemplo a seguir registra a operação `CreateGrant`:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T21:51:45Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
```

```
    "eventTime": "2022-12-06T23:40:07Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "operations": [
            "Decrypt"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:m2:app": "a1bc2defabc3defabc4defabcd"
            }
        },
        "granteePrincipal": "m2.us-west-2.amazonaws.com",
        "retiringPrincipal": "m2.us-west-2.amazonaws.com",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Decrypt

Quando você acessa uma implantação, o Amazon EC2 chama a operação `Decrypt` para usar a chave de dados criptografada armazenada para descriptografar e baixar dados criptografados do cliente do Amazon S3.

O evento de exemplo a seguir registra a operação `Decrypt`:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65BSPZ37E6G:m2-hm-bqe367dxtfcpdbzmnhfzranisu",
        "arn": "arn:aws:sts::111122223333:assumed-role/
SupernovaEnvironmentInstanceScopeDownRole/m2-hm-bqe367dxtfcpdbzmnhfzranisu",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/
SupernovaEnvironmentInstanceScopeDownRole",
                "accountId": "111122223333",
                "userName": "SupernovaEnvironmentInstanceScopeDownRole"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T23:19:29Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T23:40:15Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "aws:m2:app": "a1bc2defabc3defabc4defabcdm",
```

```
            "aws:s3:arn": "arn:aws:s3:::supernova-processedtemplate-111122223333-us-
west-2/111122223333/a1bc2defabc3defabc4defabcdm/1/cics-transaction/BBANK40P.so"
        },
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


Saiba mais

Os recursos a seguir fornecem mais informações sobre a criptografia de dados em pausa.

- Para obter mais informações sobre conceitos básicos do AWS Key Management Service, consulte o Guia do desenvolvedor do AWS Key Management Service .
- Para obter mais informações sobre as melhores práticas de segurança para o AWS Key Management Service, consulte o Guia do desenvolvedor do AWS Key Management Service .

Criptografia em trânsito

Para aplicativos interativos que fazem parte de cargas de trabalho transacionais, as trocas de dados entre o emulador de terminal e o endpoint do serviço de modernização de AWS mainframe para o protocolo TN3270 não são criptografadas em trânsito. Se a aplicação exigir criptografia em trânsito, talvez você queira implementar alguns mecanismos adicionais de tunelamento.

AWS A modernização do mainframe usa HTTPS para criptografar as APIs do serviço. Todas as outras comunicações na modernização do AWS mainframe são protegidas pelo serviço VPC ou pelo grupo de segurança, bem como pelo HTTPS. AWS A modernização do mainframe transfere artefatos, configurações e dados do aplicativo. Os artefatos da aplicação são copiados de um bucket do Amazon S3 que você possui, assim como os dados da aplicação. Você pode fornecer configurações de aplicações usando um link para o Amazon S3 ou fazendo o upload de um arquivo localmente.

A criptografia básica em trânsito é configurada por padrão, mas não se aplica ao protocolo TN3270. AWS A modernização do mainframe usa HTTPS para endpoints de API, que também são configurados por padrão.

Identity and Access Management para modernização AWS do mainframe

AWS Identity and Access Management (IAM) é uma ferramenta AWS service (Serviço da AWS) que ajuda o administrador a controlar com segurança o acesso aos AWS recursos. Os administradores do IAM controlam quem pode ser autenticado (conectado) e autorizado (tem permissões) a usar os recursos de modernização do AWS mainframe. O IAM é um AWS service (Serviço da AWS) que você pode usar sem custo adicional.

Tópicos

Público

A forma como você usa o AWS Identity and Access Management (IAM) difere, dependendo do trabalho que você faz na modernização do AWS mainframe.

Usuário do serviço — Se você usa o serviço de modernização do AWS mainframe para fazer seu trabalho, seu administrador fornecerá as credenciais e as permissões de que você precisa. À medida que você usa mais recursos de modernização de AWS mainframe para fazer seu trabalho, talvez precise de permissões adicionais. Entender como o acesso é gerenciado pode ajudar você a solicitar as permissões corretas ao seu administrador. Se você não conseguir acessar um recurso no AWS Mainframe Modernization, consulte Solução de problemas de identidade e AWS acesso à modernização do mainframe.

Administrador de serviços — Se você é responsável pelos recursos de modernização de AWS mainframe em sua empresa, provavelmente tem acesso total à modernização de AWS mainframe. É seu trabalho determinar quais recursos e recursos de modernização de AWS mainframe seus usuários do serviço devem acessar. Assim, você deve enviar solicitações ao administrador do IAM para alterar as permissões dos usuários de seu serviço. Revise as informações nesta página para entender os Introdução ao IAM. Para saber mais sobre como sua empresa pode usar o IAM com a modernização do AWS mainframe, consulte. Como a modernização AWS do mainframe funciona com o IAM

Administrador do IAM — Se você for administrador do IAM, talvez queira saber detalhes sobre como criar políticas para gerenciar o acesso à modernização do AWS mainframe. Para ver exemplos de políticas baseadas em identidade de modernização de AWS mainframe que você pode usar no IAM, consulte. Exemplos de políticas baseadas em identidade para AWS modernização de mainframe

Autenticando com identidades

A autenticação é a forma como você faz login AWS usando suas credenciais de identidade. Você deve estar autenticado (conectado AWS) como o Usuário raiz da conta da AWS, como usuário do IAM ou assumindo uma função do IAM.

Você pode entrar AWS como uma identidade federada usando credenciais fornecidas por meio de uma fonte de identidade. AWS IAM Identity Center Usuários (IAM Identity Center), a autenticação de login único da sua empresa e suas credenciais do Google ou do Facebook são exemplos de identidades federadas. Quando você faz login como identidade federada, o administrador já configurou anteriormente a federação de identidades usando perfis do IAM. Ao acessar AWS usando a federação, você está assumindo indiretamente uma função.

Dependendo do tipo de usuário que você é, você pode entrar no AWS Management Console ou no portal de AWS acesso. Para obter mais informações sobre como fazer login em AWS, consulte Como fazer login Conta da AWS no Guia do Início de Sessão da AWS usuário.

Se você acessar AWS programaticamente, AWS fornece um kit de desenvolvimento de software (SDK) e uma interface de linha de comando (CLI) para assinar criptograficamente suas solicitações usando suas credenciais. Se você não usa AWS ferramentas, você mesmo deve assinar as solicitações. Para obter mais informações sobre como usar o método recomendado para assinar solicitações por conta própria, consulte Assinatura de solicitações de AWS API no Guia do usuário do IAM.

Independente do método de autenticação usado, também pode ser exigido que você forneça informações adicionais de segurança. Por exemplo, AWS recomenda que você use a autenticação multifator (MFA) para aumentar a segurança da sua conta. Para saber mais, consulte Autenticação Multifator no Guia do Usuário do AWS IAM Identity Center . Usar a autenticação multifator (MFA) na AWS no Guia do Usuário do IAM.

Conta da AWS usuário root

Ao criar uma Conta da AWS, você começa com uma identidade de login que tem acesso completo a todos Serviços da AWS os recursos da conta. Essa identidade é chamada de usuário Conta da AWS raiz e é acessada fazendo login com o endereço de e-mail e a senha que você usou para criar a conta. É altamente recomendável não usar o usuário raiz para tarefas diárias. Proteja as credenciais do usuário raiz e use-as para executar as tarefas que somente ele puder executar. Para obter a lista completa das tarefas que exigem login como usuário raiz, consulte Tarefas que exigem credenciais de usuário raiz no Guia do Usuário do IAM.

Identidade federada

Como prática recomendada, exija que usuários humanos, incluindo usuários que precisam de acesso de administrador, usem a federação com um provedor de identidade para acessar Serviços da AWS usando credenciais temporárias.

Uma identidade federada é um usuário do seu diretório de usuários corporativo, de um provedor de identidade da web AWS Directory Service, do diretório do Identity Center ou de qualquer usuário que acesse usando credenciais fornecidas Serviços da AWS por meio de uma fonte de identidade. Quando as identidades federadas são acessadas Contas da AWS, elas assumem funções, e as funções fornecem credenciais temporárias.

Para o gerenciamento de acesso centralizado, recomendamos usar o AWS IAM Identity Center. Você pode criar usuários e grupos no IAM Identity Center ou pode se conectar e sincronizar com um conjunto de usuários e grupos em sua própria fonte de identidade para uso em todos os seus Contas

da AWS aplicativos. Para obter mais informações sobre o Centro de Identidade do IAM, consulte O que é o Centro de Identidade do IAM? no Manual do Usuário do AWS IAM Identity Center .

Usuários e grupos do IAM

Um usuário do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas para uma única pessoa ou aplicativo. Sempre que possível, recomendamos contar com credenciais temporárias em vez de criar usuários do IAM com credenciais de longo prazo, como senhas e chaves de acesso. No entanto, se você tiver casos de uso específicos que exijam credenciais de longo prazo com usuários do IAM, recomendamos alternar as chaves de acesso. Para obter mais informações, consulte Alterne as chaves de acesso regularmente para casos de uso que exijam credenciais de longo prazo no Guia do Usuário do IAM.

Um grupo do IAM é uma identidade que especifica uma coleção de usuários do IAM. Não é possível fazer login como um grupo. É possível usar grupos para especificar permissões para vários usuários de uma vez. Os grupos facilitam o gerenciamento de permissões para grandes conjuntos de usuários. Por exemplo, você pode ter um grupo chamado IAMAdmins e conceder a esse grupo permissões para administrar recursos do IAM.

Usuários são diferentes de perfis. Um usuário é exclusivamente associado a uma pessoa ou a uma aplicação, mas um perfil pode ser assumido por qualquer pessoa que precisar dele. Os usuários têm credenciais permanentes de longo prazo, mas os perfis fornecem credenciais temporárias. Para saber mais, consulte Quando criar um usuário do IAM (em vez de um perfil) no Guia do usuário do IAM.

Perfis do IAM

Uma função do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas. Ele é semelhante a um usuário do IAM, mas não está associado a uma pessoa específica. Você pode assumir temporariamente uma função do IAM no AWS Management Console trocando de funções. Você pode assumir uma função chamando uma operação de AWS API AWS CLI ou usando uma URL personalizada. Para obter mais informações sobre métodos para o uso de perfis, consulte Utilizar perfis do IAM no Guia do usuário do IAM.

Funções do IAM com credenciais temporárias são úteis nas seguintes situações:

- Acesso de usuário federado: para atribuir permissões a identidades federadas, você pode criar um perfil e definir permissões para ele. Quando uma identidade federada é autenticada, essa identidade é associada ao perfil e recebe as permissões definidas pelo mesmo. Para obter mais

informações sobre perfis para federação, consulte Criar um perfil para um provedor de identidades de terceiros no Guia do Usuário do IAM. Se você usar o Centro de identidade do IAM, configure um conjunto de permissões. Para controlar o que suas identidades podem acessar após a autenticação, o Centro de identidade do IAM correlaciona o conjunto de permissões a um perfil no IAM. Para obter informações sobre conjuntos de permissões, consulte Conjuntos de Permissões no Manual do Usuário do AWS IAM Identity Center .

- Permissões temporárias para usuários do IAM –- um usuário ou um perfil do IAM pode presumir um perfil do IAM para obter temporariamente permissões diferentes para uma tarefa específica.
- Acesso entre contas –- é possível usar um perfil do IAM para permitir que alguém (uma entidade principal confiável) em outra conta acesse recursos em sua conta. Os perfis são a principal forma de conceder acesso entre contas. No entanto, com alguns Serviços da AWS, você pode anexar uma política diretamente a um recurso (em vez de usar uma função como proxy). Para saber a diferença entre perfis e políticas baseadas em recurso para acesso entre contas, consulte Acesso a recursos entre contas no IAM no Guia do usuário do IAM.
- Acesso entre serviços — Alguns Serviços da AWS usam recursos em outros Serviços da AWS. Por exemplo, quando você faz uma chamada em um serviço, é comum que esse serviço execute aplicativos no Amazon EC2 ou armazene objetos no Amazon S3. Um serviço pode fazer isso usando as permissões do principal de chamada, usando um perfil de serviço ou um perfil vinculado a serviço.
  - Sessões de acesso direto (FAS) — Quando você usa um usuário ou uma função do IAM para realizar ações AWS, você é considerado principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O FAS usa as permissões do diretor chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) para fazer solicitações aos serviços posteriores. As solicitações do FAS são feitas somente quando um serviço recebe uma solicitação que requer interações com outros Serviços da AWS ou com recursos para ser concluída. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Encaminhar sessões de acesso.
  - Função de serviço: um perfil de serviço é um perfil do IAM que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM.
  - Função vinculada ao serviço — Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um. AWS service (Serviço da AWS) O serviço pode presumir a função de executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em você Conta da AWS

e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não editar as permissões para funções vinculadas ao serviço.

- Aplicativos em execução no Amazon EC2 — Você pode usar uma função do IAM para gerenciar credenciais temporárias para aplicativos que estão sendo executados em uma instância do EC2 e fazendo AWS CLI solicitações de API. AWS É preferível fazer isso a armazenar chaves de acesso na instância do EC2. Para atribuir uma AWS função a uma instância do EC2 e disponibilizá-la para todos os seus aplicativos, você cria um perfil de instância anexado à instância. Um perfil de instância contém o perfil e permite que os programas em execução na instância do EC2 obtenham credenciais temporárias. Para mais informações, consulte Utilizar um perfil do IAM para conceder permissões a aplicações em execução nas instâncias do Amazon EC2 no Guia do usuário do IAM.

Para saber se deseja usar perfis do IAM, consulte Quando criar um perfil do IAM (em vez de um usuário) no Guia do usuário do IAM.

Gerenciando acesso usando políticas

Você controla o acesso AWS criando políticas e anexando-as a AWS identidades ou recursos. Uma política é um objeto AWS que, quando associada a uma identidade ou recurso, define suas permissões. AWS avalia essas políticas quando um principal (usuário, usuário raiz ou sessão de função) faz uma solicitação. As permissões nas políticas determinam se a solicitação será permitida ou negada. A maioria das políticas é armazenada AWS como documentos JSON. Para obter mais informações sobre a estrutura e o conteúdo de documentos de políticas JSON, consulte Visão geral das políticas JSON no Guia do Usuário do IAM.

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

Por padrão, usuários e funções não têm permissões. Para conceder aos usuários permissões para executar ações nos recursos que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM aos perfis e os usuários podem presumir os perfis.

As políticas do IAM definem permissões para uma ação independente do método usado para executar a operação. Por exemplo, suponha que você tenha uma política que permite a ação `iam:GetRole`. Um usuário com essa política pode obter informações de função da AWS Management Console AWS CLI, da ou da AWS API.

Políticas baseadas em identidade

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário do IAM, grupo de usuários ou perfil. Essas políticas controlam quais ações os usuários e perfis podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte Criando políticas do IAM no Guia do Usuário do IAM.

As políticas baseadas em identidade podem ser categorizadas ainda adicionalmente como políticas em linha ou políticas gerenciadas. As políticas em linha são anexadas diretamente a um único usuário, grupo ou perfil. As políticas gerenciadas são políticas autônomas que você pode associar a vários usuários, grupos e funções em seu Conta da AWS. As políticas AWS gerenciadas incluem políticas gerenciadas e políticas gerenciadas pelo cliente. Para saber como escolher entre uma política gerenciada ou uma política em linha, consulte Escolher entre políticas gerenciadas e políticas em linha no Guia do Usuário do IAM.

Políticas baseadas em recursos

Políticas baseadas em recursos são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as políticas de bucket do Amazon S3. Em serviços que suportem políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o recurso ao qual a política está anexada, a política define quais ações um principal especificado pode executar nesse recurso e em que condições. Você deve especificar uma entidade principal em uma política baseada em recursos. Os diretores podem incluir contas, usuários, funções, usuários federados ou. Serviços da AWS

Políticas baseadas em recursos são políticas em linha localizadas nesse serviço. Você não pode usar políticas AWS gerenciadas do IAM em uma política baseada em recursos.

Listas de controle de acesso (ACLs)

As listas de controle de acesso (ACLs) controlam quais entidades principais (membros, usuários ou perfis da conta) têm permissões para acessar um recurso. As ACLs são semelhantes as políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

O Amazon S3 e o Amazon VPC são exemplos de serviços que oferecem suporte a ACLs. AWS WAF Para saber mais sobre ACLs, consulte Visão geral da lista de controle de acesso (ACL) no Guia do Desenvolvedor do Amazon Simple Storage Service.

Outros tipos de política

AWS oferece suporte a tipos de políticas adicionais menos comuns. Esses tipos de política podem definir o máximo de permissões concedidas a você pelos tipos de política mais comuns.

- Limites de permissões: um limite de permissões é um atributo avançado no qual você define o máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM (usuário ou perfil do IAM). É possível definir um limite de permissões para uma entidade. As permissões resultantes são a interseção das políticas baseadas em identidade de uma entidade com seus limites de permissões. As políticas baseadas em recurso que especificam o usuário ou o perfil no campo `Principal` não são limitadas pelo limite de permissões. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações sobre limites de permissões, consulte Limites de permissões para identidades do IAM no Guia do Usuário do IAM.
- Políticas de controle de serviço (SCPs) — SCPs são políticas JSON que especificam as permissões máximas para uma organização ou unidade organizacional (OU) em. AWS Organizations AWS Organizations é um serviço para agrupar e gerenciar centralmente vários Contas da AWS que sua empresa possui. Se você habilitar todos os atributos em uma organização, poderá aplicar políticas de controle de serviço (SCPs) a qualquer uma ou a todas as contas. O SCP limita as permissões para entidades nas contas dos membros, incluindo cada uma Usuário raiz da conta da AWS. Para obter mais informações sobre o Organizações e SCPs, consulte Como os SCPs Funcionam no Manual do Usuário do AWS Organizations .
- Políticas de sessão: são políticas avançadas que você transmite como um parâmetro quando cria de forma programática uma sessão temporária para um perfil ou um usuário federado. As permissões da sessão resultante são a interseção das políticas baseadas em identidade do usuário ou do perfil e das políticas de sessão. As permissões também podem ser provenientes de uma política baseada em atributo. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações, consulte Políticas de sessão no Guia do Usuário do IAM.

Vários tipos de política

Quando vários tipos de política são aplicáveis a uma solicitação, é mais complicado compreender as permissões resultantes. Para saber como AWS determinar se uma solicitação deve ser permitida quando vários tipos de políticas estão envolvidos, consulte Lógica de avaliação de políticas no Guia do usuário do IAM.

Como a modernização AWS do mainframe funciona com o IAM

Antes de usar o IAM para gerenciar o acesso à modernização do AWS mainframe, saiba quais recursos do IAM estão disponíveis para uso com a modernização do AWS mainframe.

Recursos do IAM que você pode usar com a modernização AWS do mainframe

Atributo do IAM	AWS Suporte à modernização do mainframe
Políticas baseadas em identidade	Sim
Políticas baseadas em recursos	Não
Ações das políticas	Sim
Atributos de políticas	Sim
Chaves de condição de políticas	Sim
ACLs	Não
ABAC (tags em políticas)	Sim
Credenciais temporárias	Sim
Sessões de acesso direto (FAS)	Sim
Perfis de serviço	Sim
Funções vinculadas a serviço	Sim

Para ter uma visão de alto nível de como a modernização do AWS mainframe e outros AWS serviços funcionam com a maioria dos recursos do IAM, consulte AWS os serviços que funcionam com o IAM no Guia do usuário do IAM.

Políticas baseadas em identidade para AWS modernização do mainframe

Suporta políticas baseadas em identidade	Sim

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário, grupo de usuários ou perfil do IAM. Essas políticas controlam quais ações os usuários e perfis podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte Criando políticas do IAM no Guia do Usuário do IAM.

Com as políticas baseadas em identidade do IAM, é possível especificar ações ou recursos permitidos ou negados, assim como as condições sob as quais as ações são permitidas ou negadas. Você não pode especificar a entidade principal em uma política baseada em identidade porque ela se aplica ao usuário ou perfil ao qual ela está anexada. Para saber mais sobre todos os elementos que podem ser usados em uma política JSON, consulte Referência de elementos da política JSON do IAM no Guia do Usuário do IAM.

Exemplos de políticas baseadas em identidade para AWS modernização de mainframe

Para ver exemplos de políticas baseadas em identidade de modernização de AWS mainframe, consulte. Exemplos de políticas baseadas em identidade para AWS modernização de mainframe

Políticas baseadas em recursos na modernização do mainframe AWS

Oferece compatibilidade com políticas baseadas em recursos	Não

Políticas baseadas em recursos são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as políticas de bucket do Amazon S3. Em serviços que suportem políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o recurso ao qual a política está anexada, a política define quais ações um principal especificado pode executar nesse recurso e em que condições. Você deve especificar uma entidade principal em uma política baseada em recursos. Os diretores podem incluir contas, usuários, funções, usuários federados ou. Serviços da AWS

Para permitir o acesso entre contas, você pode especificar uma conta inteira ou as entidades do IAM em outra conta como a entidade principal em uma política baseada em atributo. Adicionar uma entidade principal entre contas à política baseada em recurso é apenas metade da tarefa de estabelecimento da relação de confiança. Quando o principal e o recurso são diferentes Contas da

AWS, um administrador do IAM na conta confiável também deve conceder permissão à entidade principal (usuário ou função) para acessar o recurso. Eles concedem permissão ao anexar uma política baseada em identidade para a entidade. No entanto, se uma política baseada em recurso conceder acesso a uma entidade principal na mesma conta, nenhuma política baseada em identidade adicional será necessária. Consulte mais informações em Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

Ações políticas para modernização AWS do mainframe

Oferece compatibilidade com ações de políticas	Sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento `Action` de uma política JSON descreve as ações que você pode usar para permitir ou negar acesso em uma política. As ações de política geralmente têm o mesmo nome da operação de AWS API associada. Existem algumas exceções, como ações somente de permissão, que não têm uma operação de API correspondente. Algumas operações também exigem várias ações em uma política. Essas ações adicionais são chamadas de ações dependentes.

Incluem ações em uma política para conceder permissões para executar a operação associada.

Para ver uma lista das ações de modernização do AWS mainframe, consulte Ações definidas pela modernização do AWS mainframe na Referência de autorização de serviço.

As ações de política na modernização do AWS mainframe usam o seguinte prefixo antes da ação:

```
m2
```

Para especificar várias ações em uma única instrução, separe-as com vírgulas.

```
"Action": [
      "m2:StartApplication",
      "m2:StopApplication"
         ]
```

Você também pode especificar várias ações usando caracteres-curinga (*). Por exemplo, para especificar todas as ações que começam com a palavra `List`, inclua a seguinte ação:

```
"Action": "m2:List*"
```

Para ver exemplos de políticas baseadas em identidade de modernização de AWS mainframe, consulte. Exemplos de políticas baseadas em identidade para AWS modernização de mainframe

Recursos de políticas para a modernização AWS do mainframe

Oferece compatibilidade com recursos de políticas	Sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento de política JSON `Resource` especifica o objeto ou os objetos aos quais a ação se aplica. As instruções devem incluir um elemento `Resource` ou `NotResource`. Como prática recomendada, especifique um recurso usando seu nome do recurso da Amazon (ARN). Isso pode ser feito para ações que oferecem compatibilidade com um tipo de recurso específico, conhecido como permissões em nível de recurso.

Para ações que não oferecem compatibilidade com permissões em nível de recurso, como operações de listagem, use um curinga (*) para indicar que a instrução se aplica a todos os recursos.

```
"Resource": "*"
```

Você pode restringir o acesso a recursos específicos de modernização de AWS mainframe usando seus ARNs para identificar o recurso ao qual a política do IAM se aplica. Para obter mais informações sobre o formato de ARNs, consulte Nomes de recurso da Amazon (ARNs) na Referência geral da AWS.

Por exemplo, um ambiente de modernização de AWS mainframe tem o seguinte ARN.

```
"Resource": "arn:aws:m2:regionId:accountId:env/service-generated-unique-identifier"
```

Um aplicativo de modernização de AWS mainframe tem o seguinte ARN.

```
"Resource": "arn:aws:m2:regionId:accountId:app/service-generated-unique-identifier"
```

Nem todas as ações de modernização do AWS mainframe oferecem suporte a permissões em nível de recurso. Para ações que não suportam permissões em nível de recurso, você deve usar o curinga (*).

As seguintes ações de modernização do AWS mainframe não oferecem suporte a permissões em nível de recurso.

```
ListApplications
            ListApplicationVersions
            ListBatchJobDefinitions
            ListBatchJobExecutions
            ListDataSetImportHistory
            ListDataSets
            ListDeployments
            ListEngineVersions
            ListEnvironments
            ListTagsForResource
```

Para ver uma lista dos tipos de recursos de modernização do AWS mainframe e seus ARNs, consulte Recursos definidos pela modernização do AWS mainframe na Referência de autorização de serviço. Para saber com quais ações você pode especificar o ARN de cada recurso, consulte Ações definidas pela modernização do AWS mainframe.

Para ver exemplos de políticas baseadas em identidade de modernização de AWS mainframe, consulte. Exemplos de políticas baseadas em identidade para AWS modernização de mainframe

AWS Permissões da API de modernização de mainframe: referência de ações, recursos e condições

Ao escrever políticas de permissões que podem ser anexadas a uma identidade do IAM (políticas baseadas em identidade), você pode usar a tabela a seguir como referência. A tabela inclui o seguinte:

- Cada operação da API AWS de modernização de mainframe.
- As ações correspondentes para as quais você pode conceder permissões para realizar a ação.
- O AWS recurso para o qual você pode conceder as permissões.

Especifique as ações no campo `Action` da política e o valor do recurso no campo `Resource` da política.

Você pode usar chaves de condição AWS globais em suas políticas de modernização de AWS mainframe para expressar condições. Para obter uma lista completa das AWS chaves, consulte Chaves de condição globais disponíveis no Guia do usuário do IAM.

> **Note**
>
> Para especificar uma ação, use o prefixo `m2:` seguido do nome da operação da API (por exemplo, `m2:CreateApplication`).

AWS API de modernização de mainframe e permissões necessárias para ações

AWS Operações de API de modernização de mainframe	Permissões obrigatórias (ações de API)	Recursos
CancelBatchJobExecution		Aplicativo

CreateApplication	`iam:PassRole`
	`kms:DescribeKey`
	`kms:CreateGrant`
	`s3:GetObject`
	`s3:ListBucket`
		Aplicativo

CreateDataSetImportTask	`m2:CreateDataSetImportTask` `s3:GetObject`	Aplicativo
CreateDeployment	`elasticloadbalancing:AddTags` `elasticloadbalancing:CreateListener`	Aplicativo

AWS Operações de API de modernização de mainframe	Permissões obrigatórias (ações de API)	Recursos

	`elasticloadbalancing:CreateTargetGroup`
	
	`elasticloadbalancing:RegisterTargets`	

AWS Operações de API de modernização de mainframe	Permissões obrigatórias (ações de API)	Recursos
CreateEnvironment	ec2:CreateNetworkInterface ec2:CreateNetworkInterfacePermission ec2:DescribeNetworkInterfaces ec2:DescribeSecurityGroups ec2:DescribeSubnets ec2:DescribeVpcAttribute ec2:DescribeVpcs ec2:ModifyNetworkInterfaceAttribute elasticfilesystem:DescribeMountTargets elasticloadbalancing:AddTags elasticloadbalancing:CreateLoadBalancer elasticloadbalancing:DeleteLoadBalancer kms:DescribeKey kms:CreateGrant fsx:DescribeFileSystems	Ambiente

AWS Operações de API de modernização de mainframe	Permissões obrigatórias (ações de API)	Recursos
	`iam:CreateServiceLinkedRole`	

DeleteApplication	`elasticloadbalancing:DeleteListener`
	`elasticloadbalancing:DeleteTargetGroup`
	`logs:DeleteLogDelivery`
		Aplicativo

DeleteApplicationFromEnvironment	`elasticloadbalancing:DeleteListener` `elasticloadbalancing:DeleteTargetGroup`	Aplicativo Ambiente
DeleteEnvironment	`elasticloadbalancing:DeleteLoadBalancer`	Ambiente
GetApplication		Aplicativo
GetApplicationVersion		Aplicativo
GetBatchJobExecution		Aplicativo
GetDataSetDetails		Aplicativo
GetDataSetImportTask		Aplicativo
GetDeployment		Aplicativo
GetEnvironment		Ambiente
ListApplications		*

AWS Operações de API de modernização de mainframe	Permissões obrigatórias (ações de API)	Recursos
ListApplicationVersions		*
ListBatchJobDefinitions		*
ListBatchJobExecutions		*
ListDataSetImportHistory		*
ListDataSets		*
ListDeployments		*
ListEngineVersions		*
ListEnvironments		*
ListTagsForResource		*
StartApplication		Aplicativo
StartBatchJob		Aplicativo
StopApplication		Aplicativo
TagResource		*
UntagResource		*
UpdateApplication	`s3:GetObject` `s3:ListBucket`	Aplicativo
UpdateEnvironment	`kms:DescribeKey`	Ambiente

Chaves de condição de política para a AWS modernização do mainframe

Suporta chaves de condição de política específicas de serviço	Sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos, e em que condições.

O elemento `Condition` (ou bloco `Condition`) permite que você especifique condições nas quais uma instrução estiver em vigor. O elemento `Condition` é opcional. É possível criar expressões condicionais que usem agentes de condição, como “igual a” ou “menor que”, para fazer a condição da política corresponder aos valores na solicitação.

Se você especificar vários elementos `Condition` em uma instrução ou várias chaves em um único `Condition` elemento, a AWS os avaliará usando uma operação lógica `AND`. Se você especificar vários valores para uma única chave de condição, AWS avalia a condição usando uma `OR` operação lógica. Todas as condições devem ser atendidas antes que as permissões da instrução sejam concedidas.

Você também pode usar variáveis de espaço reservado ao especificar condições. Por exemplo, é possível conceder a um usuário do IAM permissão para acessar um atributo somente se ele estiver marcado com seu nome de usuário do IAM. Para obter mais informações, consulte Elementos da política do IAM: variáveis e tags no Guia do usuário do IAM.

AWS suporta chaves de condição globais e chaves de condição específicas do serviço. Para ver todas as chaves de condição AWS globais, consulte as chaves de contexto de condição AWS global no Guia do usuário do IAM.

As seguintes chaves de condição são específicas para a modernização AWS do mainframe

```
m2:EngineType
            m2:InstanceType
```

Para ver uma lista das chaves de condição de modernização de AWS mainframe, consulte Chaves de condição para modernização de AWS mainframe na Referência de autorização de serviço. Para saber com quais ações e recursos você pode usar uma chave de condição, consulte Ações definidas pela modernização do AWS mainframe.

Para ver exemplos de políticas baseadas em identidade de modernização de AWS mainframe, consulte. Exemplos de políticas baseadas em identidade para AWS modernização de mainframe

Listas de controle de acesso (ACLs) no AWS Mainframe Modernization

Oferece compatibilidade com ACLs	Não

As listas de controle de acesso (ACLs) controlam quais entidades principais (membros, usuários ou perfis da conta) têm permissões para acessar um recurso. As ACLs são semelhantes as políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

Controle de acesso baseado em atributos (ABAC) com modernização do mainframe AWS

Oferece compatibilidade com ABAC (tags em políticas)	Sim

O controle de acesso por atributo (ABAC) é uma estratégia de autorização que define permissões com base em atributos. Em AWS, esses atributos são chamados de tags. Você pode anexar tags a entidades do IAM (usuários ou funções) e a vários AWS recursos. A marcação de entidades e atributos é a primeira etapa do ABAC. Em seguida, você cria políticas de ABAC para permitir operações quando a tag da entidade principal corresponder à tag do recurso que ela estiver tentando acessar.

O ABAC é útil em ambientes que estão crescendo rapidamente e ajuda em situações onde o gerenciamento de políticas se torna um problema.

Para controlar o acesso baseado em tags, forneça informações sobre as tags no elemento de condição de uma política usando as `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* ou chaves de condição `aws:TagKeys`.

Se um serviço oferecer suporte às três chaves de condição para cada tipo de recurso, o valor será Sim para o serviço. Se um serviço oferecer suporte às três chaves de condição somente para alguns tipos de recursos, o valor será Parcial

Para obter mais informações sobre o ABAC, consulte O que é ABAC? no Guia do Usuário do IAM. Para visualizar um tutorial com etapas para configurar o ABAC, consulte Utilizar controle de acesso baseado em atributos (ABAC) no Guia do usuário do IAM.

Usando credenciais temporárias com a modernização do AWS mainframe

Oferece compatibilidade com credenciais temporárias	Sim

Alguns Serviços da AWS não funcionam quando você faz login usando credenciais temporárias. Para obter informações adicionais, incluindo quais Serviços da AWS funcionam com credenciais temporárias, consulte Serviços da AWS “Trabalhe com o IAM” no Guia do usuário do IAM.

Você está usando credenciais temporárias se fizer login AWS Management Console usando qualquer método, exceto um nome de usuário e senha. Por exemplo, quando você acessa AWS usando o link de login único (SSO) da sua empresa, esse processo cria automaticamente credenciais temporárias. Você também cria automaticamente credenciais temporárias quando faz login no console como usuário e, em seguida, alterna perfis. Para obter mais informações sobre como alternar funções, consulte Alternar para um perfil (console) no Guia do usuário do IAM.

Você pode criar manualmente credenciais temporárias usando a AWS API AWS CLI ou. Em seguida, você pode usar essas credenciais temporárias para acessar AWS. AWS recomenda que você gere credenciais temporárias dinamicamente em vez de usar chaves de acesso de longo prazo. Para mais informações, consulte Credenciais de segurança temporárias no IAM.

Sessões de acesso direto para modernização AWS do mainframe

Suporte para o recurso Encaminhamento de sessões de acesso (FAS)	Sim

Quando você usa um usuário ou uma função do IAM para realizar ações AWS, você é considerado principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O FAS usa as permissões do diretor chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) para fazer solicitações aos serviços posteriores. As solicitações do FAS são feitas somente quando um serviço recebe uma solicitação que requer interações com outros Serviços da AWS ou com recursos para ser concluída.

Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Encaminhar sessões de acesso.

> **Important**
>
> Esses tokens dão à Modernização do AWS Mainframe acesso aos dados do cliente sem seu acordo explícito; por exemplo, a Modernização do AWS Mainframe implanta artefatos de aplicativos com dados comerciais associados de um bucket do Amazon S3 sem obter permissão explícita do cliente. Talvez seja necessário atualizar qualquer documentação de conformidade de acordo.

Perfis de serviço para o AWS Mainframe Modernization

Oferece compatibilidade com funções de serviço	Sim

O perfil de serviço é um perfil do IAM que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM.

AWS A modernização do mainframe oferece suporte a funções de serviço para ganchos de atividades (transações/tarefas pendentes ou concluídas, etc.).

> **Warning**
>
> Alterar as permissões de uma função de serviço pode interromper a funcionalidade de modernização do AWS mainframe. Edite as funções de serviço somente quando a modernização do AWS mainframe fornecer orientação para fazer isso.

Escolha de uma função do IAM na modernização AWS do mainframe

Se você já criou uma função do IAM que seus aplicativos executados no Amazon EC2 podem assumir, você pode escolher essa função ao criar um modelo de execução ou uma configuração de execução. AWS A modernização do mainframe fornece uma lista de funções para você escolher.

Ao criar essas funções, é importante associar políticas do IAM de privilégio mínimo que restrinjam o acesso às chamadas de API específicas necessárias para a aplicação. Para obter mais informações, consulte Função do IAM para aplicativos que são executados em instâncias do Amazon EC2 no Guia do usuário do Amazon EC2 Auto Scaling.

Funções vinculadas a serviços para AWS modernização do mainframe

Oferece suporte a perfis vinculados ao serviço	Sim

Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um. AWS service (Serviço da AWS) O serviço pode presumir a função de executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em você Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não editar as permissões para funções vinculadas ao serviço.

Para obter detalhes sobre como criar ou gerenciar funções vinculadas ao serviço de modernização de AWS mainframe, consulte. Usar funções vinculadas ao serviço do

Para obter detalhes sobre como criar ou gerenciar perfis vinculados a serviços, consulte Serviços da AWS que funcionam com o IAM. Encontre um serviço na tabela que inclua um `Yes` na coluna Função vinculada ao serviço. Escolha o link Sim para visualizar a documentação do perfil vinculado a serviço desse serviço.

Exemplos de políticas baseadas em identidade para AWS modernização de mainframe

Por padrão, usuários e funções não têm permissão para criar ou modificar recursos de modernização de AWS mainframe. Eles também não podem realizar tarefas usando a AWS API AWS Management Console, AWS Command Line Interface (AWS CLI) ou. Para conceder aos usuários permissão para executar ações nos recursos de que precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM aos perfis e os usuários podem presumir os perfis.

Para saber como criar uma política baseada em identidade do IAM usando esses exemplos de documento de política JSON, consulte Criação de políticas do IAM no Guia do Usuário do IAM.

Para obter detalhes sobre ações e tipos de recursos definidos pela modernização do AWS mainframe, incluindo o formato dos ARNs para cada um dos tipos de recursos, consulte Ações,

recursos e chaves de condição para a modernização do AWS mainframe na Referência de autorização de serviço.

Tópicos

- Melhores práticas de política
- Usando o console de modernização AWS de mainframe
- Permitir que usuários visualizem suas próprias permissões

Melhores práticas de política

As políticas baseadas em identidade determinam se alguém pode criar, acessar ou excluir recursos de modernização de AWS mainframe em sua conta. Essas ações podem incorrer em custos para seus Conta da AWS. Ao criar ou editar políticas baseadas em identidade, siga estas diretrizes e recomendações:

- Comece com as políticas AWS gerenciadas e avance para as permissões de privilégios mínimos — Para começar a conceder permissões aos seus usuários e cargas de trabalho, use as políticas AWS gerenciadas que concedem permissões para muitos casos de uso comuns. Eles estão disponíveis no seu Conta da AWS. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo AWS cliente que sejam específicas para seus casos de uso. Para obter mais informações, consulte Políticas gerenciadas pela AWS ou Políticas gerenciadas pela AWS para funções de trabalho no Guia do Usuário do IAM.
- Aplique permissões de privilégio mínimo –- ao definir permissões com as políticas do IAM, conceda apenas as permissões necessárias para executar uma tarefa. Você faz isso definindo as ações que podem ser executadas em atributos específicos sob condições específicas, também conhecidas como permissões de privilégio mínimo. Para obter mais informações sobre como usar o IAM para aplicar permissões, consulte Políticas e permissões no IAM no Guia do Usuário do IAM.
- Use condições nas políticas do IAM para restringir ainda mais o acesso –- você pode adicionar uma condição às políticas para limitar o acesso a ações e recursos. Por exemplo, você pode gravar uma condição de política para especificar que todas as solicitações devem ser enviadas usando SSL. Você também pode usar condições para conceder acesso às ações de serviço se elas forem usadas por meio de uma ação específica AWS service (Serviço da AWS), como AWS CloudFormation. Para obter mais informações, consulte Elementos da política JSON do IAM: Condição no Guia do usuário do IAM.

- Use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais –- o IAM Access Analyzer valida as políticas novas e existentes para que elas sigam a linguagem de política do IAM (JSON) e as práticas recomendadas do IAM. O IAM Access Analyzer oferece mais de 100 verificações de política e recomendações acionáveis para ajudá-lo a criar políticas seguras e funcionais. Para obter mais informações, consulte Validação de políticas do IAM Access Analyzer no Guia do Usuário do IAM.
- Exigir autenticação multifator (MFA) — Se você tiver um cenário que exija usuários do IAM ou um usuário root, ative Conta da AWS a MFA para obter segurança adicional. Para exigir MFA quando as operações de API forem chamadas, adicione condições de MFA às suas políticas. Para obter mais informações, consulte Configuração de acesso à API protegido por MFA no Guia do Usuário do IAM.

Para obter mais informações sobre as práticas recomendadas do IAM, consulte Práticas Recomendadas de Segurança no IAM no Guia do Usuário do IAM.

Usando o console de modernização AWS de mainframe

Para acessar o console de modernização do AWS mainframe, você deve ter um conjunto mínimo de permissões. Essas permissões devem permitir que você liste e visualize detalhes sobre os recursos de modernização do AWS mainframe em seu. Conta da AWS Se você criar uma política baseada em identidade que seja mais restritiva do que as permissões mínimas necessárias, o console não funcionará como pretendido para entidades (usuários ou perfis) com essa política.

Você não precisa permitir permissões mínimas do console para usuários que estão fazendo chamadas somente para a API AWS CLI ou para a AWS API. Em vez disso, permita o acesso somente a ações que correspondam a operação de API que estiverem tentando executar.

Para garantir que os usuários e as funções ainda possam usar o console de modernização do AWS mainframe, anexe também a política `ReadOnly` AWS gerenciada `ConsoleAccess` ou a modernização do AWS mainframe às entidades. Para obter mais informações, consulte Adicionando Permissões a um Usuário no Guia do Usuário do IAM.

Permitir que usuários visualizem suas próprias permissões

Este exemplo mostra como criar uma política que permita que os usuários do IAM visualizem as políticas gerenciadas e em linha anexadas a sua identidade de usuário. Essa política inclui permissões para concluir essa ação no console ou programaticamente usando a API AWS CLI ou AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Solução de problemas de identidade e AWS acesso à modernização do mainframe

Use as informações a seguir para ajudá-lo a diagnosticar e corrigir problemas comuns que você pode encontrar ao trabalhar com a modernização do AWS mainframe e o IAM.

Tópicos

- Não estou autorizado a realizar iam: PassRole
- Quero permitir que pessoas de fora da minha tenham acesso Conta da AWS aos meus recursos de modernização de AWS mainframe

Não estou autorizado a realizar iam: PassRole

Se você receber um erro informando que não está autorizado a realizar a `iam:PassRole` ação, suas políticas devem ser atualizadas para permitir que você passe uma função para a modernização do AWS mainframe.

Alguns Serviços da AWS permitem que você passe uma função existente para esse serviço em vez de criar uma nova função de serviço ou uma função vinculada ao serviço. Para fazer isso, é preciso ter permissões para passar o perfil para o serviço.

O exemplo de erro a seguir ocorre quando um usuário IAM chamado `marymajor` tenta usar o console para executar uma ação no AWS Mainframe Modernization. No entanto, a ação exige que o serviço tenha permissões concedidas por um perfil de serviço. Mary não tem permissões para passar o perfil para o serviço.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Nesse caso, as políticas de Mary devem ser atualizadas para permitir que ela realize a ação `iam:PassRole`.

Se precisar de ajuda, entre em contato com seu AWS administrador. Seu administrador é a pessoa que forneceu suas credenciais de login.

Quero permitir que pessoas de fora da minha tenham acesso Conta da AWS aos meus recursos de modernização de AWS mainframe

Você pode criar um perfil que os usuários de outras contas ou pessoas fora da sua organização podem usar para acessar seus recursos. Você pode especificar quem é confiável para assumir o perfil. Para serviços que oferecem compatibilidade com políticas baseadas em recursos ou listas de controle de acesso (ACLs), você pode usar essas políticas para conceder às pessoas acesso aos seus recursos.

Para saber mais, consulte:

- Para saber se a modernização do AWS mainframe é compatível com esses recursos, consulte. Como a modernização AWS do mainframe funciona com o IAM
- Para saber como fornecer acesso aos seus recursos em todos os Contas da AWS que você possui, consulte Como fornecer acesso a um usuário do IAM em outro Conta da AWS que você possui no Guia do usuário do IAM.
- Para saber como fornecer acesso aos seus recursos a terceiros Contas da AWS, consulte Como fornecer acesso Contas da AWS a terceiros no Guia do usuário do IAM.
- Para saber como conceder acesso por meio da federação de identidades, consulte Conceder acesso a usuários autenticados externamente (federação de identidades) no Guia do usuário do IAM.
- Para saber a diferença entre o uso de perfis e políticas baseadas em recurso para acesso entre contas, consulte Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

Usar funções vinculadas ao serviço do

AWS Mainframe Modernization usa funções vinculadas ao serviço AWS Identity and Access Management (IAM). Um perfil vinculado ao serviço é um tipo especial de perfil do IAM vinculado diretamente ao Mainframe Modernization. As funções vinculadas ao serviço são predefinidas pela modernização do mainframe e incluem todas as permissões que o serviço exige para chamar outros AWS serviços em seu nome.

Um perfil vinculado ao serviço facilita a configuração do Mainframe Modernization porque você não precisa adicionar as permissões necessárias manualmente. O Mainframe Modernization define as permissões das funções vinculadas ao serviço e, exceto se definido de outra forma, somente o Mainframe Modernization pode assumir suas funções. As permissões definidas incluem a política de confiança e a política de permissões, e essa política não pode ser anexada a nenhuma outra entidade do IAM.

Uma função vinculada ao serviço poderá ser excluída somente após excluir seus recursos relacionados. Isso protege seus recursos do Mainframe Modernization, pois você não pode remover por engano as permissões para acessar os recursos.

Para obter informações sobre outros serviços compatíveis com perfis vinculados aos serviços, consulte Serviços da AWS que funcionam com o IAM e procure os serviços que apresentam Sim na coluna Funções vinculadas aos serviços. Escolha um Sim com um link para visualizar a documentação da função vinculada a esse serviço.

Permissões de perfil vinculado ao serviço para o Mainframe Modernization

A modernização do mainframe usa a função vinculada ao serviço chamada AWSServiceRoleForAWSM2 — configure a rede para se conectar à sua VPC e acessar recursos, como sistemas de arquivos.

A função vinculada a AWSServiceRoleForAWSM 2 serviços confia nos seguintes serviços para assumir a função:

- `m2.amazonaws.com`

A política de permissões de função denominada AWSM2ServicePolicy permite que a modernização do mainframe conclua as seguintes ações nos recursos especificados:

- Crie, exclua, descreva e anexe permissões às interfaces de rede do Amazon EC2 para o ambiente do Mainframe Modernization a fim de estabelecer conectividade com a VPC do cliente.
- Registre ou cancele o registro de entradas do Elastic Load Balancing, que é como os clientes se conectam ao ambiente do Mainframe Modernization.
- Descreva o sistema de arquivos Amazon EFS ou Amazon FSx, se usado.
- Emita métricas para o cliente a CloudWatch partir do ambiente de execução.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Action": [
    "ec2:DescribeSubnets",
    "ec2:CreateNetworkInterface",
    "ec2:DeleteNetworkInterface",
    "ec2:DescribeNetworkInterfaces",
    "ec2:CreateNetworkInterfacePermission",
    "ec2:ModifyNetworkInterfaceAttribute"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
```

```
    "elasticfilesystem:DescribeMountTargets"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "elasticloadbalancing:RegisterTargets",
    "elasticloadbalancing:DeregisterTargets"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "fsx:DescribeFileSystems"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "cloudwatch:PutMetricData"
   ],
   "Resource": "*",
   "Condition": {
    "StringEquals": {
     "cloudwatch:namespace": [
      "AWS/M2"
     ]
    }
   }
  }
 ]
}
```

Você deve configurar permissões para que uma entidade do IAM (por exemplo, um usuário, grupo ou função) crie, edite ou exclua uma função vinculada a serviço. Para obter mais informações, consulte Permissões de perfil vinculado a serviços no Guia do usuário do IAM.

Criar um perfil vinculado a serviço para o Mainframe Modernization

Não é necessário criar manualmente uma função vinculada ao serviço. Quando você cria um ambiente de tempo de execução na AWS Management Console, na ou na AWS API AWS CLI, a modernização do mainframe cria a função vinculada ao serviço para você.

Se excluir essa função vinculada ao serviço e precisar criá-la novamente, você poderá usar esse mesmo processo para recriar a função em sua conta. Quando você cria um ambiente de runtime, o Mainframe Modernization cria o perfil vinculado ao serviço para você novamente.

Editar um perfil vinculado ao serviço para o Mainframe Modernization

A modernização do mainframe não permite que você edite as AWSServiceRoleForAWSM duas funções vinculadas ao serviço. Depois que criar um perfil vinculado ao serviço, você não poderá alterar o nome do perfil, pois várias entidades podem fazer referência a ele. No entanto, será possível editar a descrição do perfil usando o IAM. Para ter mais informações, consulte Editar uma função vinculada a serviço no Guia do usuário do IAM.

Excluir um perfil vinculado ao serviço para o Mainframe Modernization

Se você não precisar mais usar um atributo ou serviço que requer uma função vinculada a serviço, é recomendável excluí-la. Dessa forma, você não tem uma entidade não utilizada que não seja monitorada ativamente ou mantida. No entanto, você deve limpar os recursos de sua função vinculada ao serviço antes de excluí-la manualmente.

> **Note**
>
> Se o serviço Mainframe Modernization estiver usando a função quando você tentar excluir os recursos, a exclusão poderá falhar. Se isso acontecer, espere alguns minutos e tente a operação novamente.

Para excluir os recursos de modernização do mainframe usados pelos 2 AWSServiceRoleForAWSM

- Exclua os ambientes de runtime no Mainframe Modernization. Certifique-se de excluir aplicações de um ambiente antes de excluir o próprio ambiente.

Como excluir manualmente a função vinculada a serviço usando o IAM

Use o console do IAM AWS CLI, o ou a AWS API para excluir as AWSServiceRoleForAWSM duas funções vinculadas ao serviço. Para obter mais informações, consulte Excluir uma função vinculada ao serviço no Guia do usuário do IAM.

Regiões compatíveis com perfis vinculados ao serviço do Mainframe Modernization

O Mainframe Modernization oferece suporte a perfis vinculados ao serviço em todas as regiões em que o serviço está disponível. Para obter mais informações, consulte Regiões e endpoints da AWS.

Validação de conformidade para AWS modernização do mainframe

Auditores terceirizados avaliam a segurança e a conformidade da modernização do AWS mainframe como parte de vários AWS programas de conformidade. Isso inclui SOC, PCI, FedRAMP, HIPAA e outros.

Para obter uma lista de AWS serviços no escopo de programas de conformidade específicos, consulte Serviços da AWS no escopo do programa de conformidade . Para obter informações gerais, consulte Programas de Conformidade da AWS.

Você pode baixar relatórios de auditoria de terceiros usando AWS Artifact. Para obter mais informações, consulte Baixar relatórios em AWS Artifact .

Sua responsabilidade de conformidade ao usar a modernização do AWS mainframe é determinada pela confidencialidade de seus dados, pelos objetivos de conformidade da sua empresa e pelas leis e regulamentações aplicáveis. AWS fornece os seguintes recursos para ajudar na conformidade:

- Guias de início rápido de segurança e compatibilidade: estes guias de implantação abordam as considerações de arquitetura e fornecem etapas para implantação de ambientes de linha de base focados em compatibilidade e segurança na AWS.
- Documento técnico sobre arquitetura para segurança e conformidade com a HIPAA — Este whitepaper descreve como as empresas podem usar para criar aplicativos compatíveis com a HIPAA. AWS
- AWS Recursos de https://aws.amazon.com/compliance/resources/ de conformidade — Essa coleção de pastas de trabalho e guias pode ser aplicada ao seu setor e local.
- Avaliação de recursos com regras no Guia do AWS Config Desenvolvedor — AWS Config avalia o quão bem suas configurações de recursos estão em conformidade com as práticas internas, as diretrizes do setor e os regulamentos.

- AWS Security Hub— Esse AWS serviço fornece uma visão abrangente do seu estado de segurança interno, AWS que ajuda você a verificar sua conformidade com os padrões e as melhores práticas do setor de segurança.

Resiliência na modernização do AWS mainframe

A infraestrutura AWS global é construída em torno de AWS regiões e zonas de disponibilidade. As regiões fornecem várias zonas de disponibilidade separadas e isoladas fisicamente, que são conectadas com baixa latência, alta throughput e redes altamente redundantes. Com as zonas de disponibilidade, é possível projetar e operar aplicações e bancos de dados que automaticamente executam o failover entre as zonas sem interrupção. As zonas de disponibilidade são mais altamente disponíveis, tolerantes a falhas e escaláveis que uma ou várias infraestruturas de data center tradicionais.

Para obter mais informações sobre AWS regiões e zonas de disponibilidade, consulte Infraestrutura AWS global.

Segurança da infraestrutura em AWS Mainframe Modernization

Como serviço gerenciado, AWS Mainframe Modernization é protegido pela segurança de rede AWS global. Para obter informações sobre serviços AWS de segurança e como AWS proteger a infraestrutura, consulte AWS Cloud Security. Para projetar seu AWS ambiente usando as melhores práticas de segurança de infraestrutura, consulte Proteção de infraestrutura no Security Pillar AWS Well-Architected Framework.

Você usa chamadas de API AWS publicadas para acessar a modernização do mainframe por meio da rede. Os clientes devem oferecer suporte para:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Conjuntos de criptografia com perfect forward secrecy (PFS) como DHE (Ephemeral Diffie-Hellman) ou ECDHE (Ephemeral Elliptic Curve Diffie-Hellman). A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

Além disso, as solicitações devem ser assinadas usando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou você pode usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

Acesso AWS Mainframe Modernization usando um endpoint de interface ()AWS PrivateLink

Você pode usar AWS PrivateLink para criar uma conexão privada entre sua VPC e. AWS Mainframe Modernization Você pode acessar a modernização do mainframe como se estivesse em sua VPC, sem o uso de um gateway de internet, dispositivo NAT, conexão VPN ou conexão. AWS Direct Connect As instâncias em sua VPC não precisam de endereços IP públicos para acessar o Mainframe Modernization.

Você estabelece essa conectividade privada criando um endpoint de interface, desenvolvido pelo AWS PrivateLink. Criaremos um endpoint de interface de rede em cada sub-rede que você habilitar para o endpoint de interface. Essas são interfaces de rede gerenciadas pelo solicitante que servem como ponto de entrada para o tráfego destinado ao Mainframe Modernization.

Para obter mais informações, consulte Acesso Serviços da AWS por meio AWS PrivateLink do AWS PrivateLink Guia.

Considerações sobre o Mainframe Modernization

Antes de configurar um endpoint de interface para o Mainframe Modernization, consulte Considerations no Guia do AWS PrivateLink .

O Mainframe Modernization é compatível com chamadas para todas as ações de API da interface por meio do endpoint da interface.

Criar um endpoint de interface para o Mainframe Modernization

Você pode criar um endpoint de interface para modernização de mainframe usando o console Amazon VPC ou o (). AWS Command Line Interface AWS CLI Para obter mais informações, consulte Criar um endpoint de interface no Guia do usuário do AWS PrivateLink .

Crie um endpoint de interface para o Mainframe Modernization usando o seguinte nome de serviço:

```
com.amazonaws.region.m2
```

Se você habilitar o DNS privado para o endpoint de interface, poderá fazer solicitações de API para o Mainframe Modernization usando seu nome DNS regional padrão. Por exemplo, `m2.us-east-1.amazonaws.com`.

Crie uma política de endpoint para seu endpoint de interface.

Política de endpoint é um recurso do IAM que você pode anexar ao endpoint de interface. A política de endpoint padrão permite acesso total ao Mainframe Modernization por meio do endpoint de interface. Para controlar o acesso permitido ao Mainframe Modernization a partir de sua VPC, anexe uma política de endpoint personalizada ao endpoint da interface.

Uma política de endpoint especifica as seguintes informações:

- As entidades principais que podem executar ações (Contas da AWS, usuários e funções do IAM).
- As ações que podem ser executadas.
- Os recursos nos quais as ações podem ser executadas.

Para obter mais informações, consulte Controlar o Acesso a Serviços Usando Políticas de Endpoint no AWS PrivateLink Guia.

Exemplo: política de endpoint da VPC para ações do Mainframe Modernization

O exemplo a seguir refere-se a uma política de endpoint personalizada. Quando você anexa essa política ao endpoint de interface, ela concede acesso às ações do Mainframe Modernization listadas para todas as entidades principais em todos os recursos.

```
//Example of an endpoint policy where access is granted to the
//listed AWS Mainframe Modernization actions for all principals on all resources
{"Statement": [
      {"Principal": "*",
         "Effect": "Allow",
         "Action": [
            "m2:ListApplications",
            "m2:ListEnvironments",
            "m2:ListDeployments"
         ],
         "Resource":"*"
      }
   ]
}

//Example of an endpoint policy where access is denied to all the
//AWS Mainframe Modernization CREATE actions for all principals on all resources
{"Statement": [
      {"Principal": "*",
```

```
         "Effect": "Deny",
         "Action": [
            "m2:Create*"
         ],
         "Resource":"*"
      }
   ]
}
```

Monitorando a AWS modernização do mainframe

O monitoramento é uma parte importante da manutenção da confiabilidade, disponibilidade e desempenho da modernização do AWS mainframe e de suas outras soluções da AWS. A AWS fornece as seguintes ferramentas de monitoramento para observar a modernização do AWS mainframe, relatar quando algo está errado e realizar ações automáticas quando apropriado:

- A Amazon CloudWatch monitora seus AWS recursos e os aplicativos em que você executa AWS em tempo real. É possível coletar e rastrear métricas, criar painéis personalizados e definir alarmes que o notificam ou que realizam ações quando uma métrica especificada atinge um limite definido. Por exemplo, você pode CloudWatch rastrear o uso da CPU ou outras métricas de suas instâncias do Amazon EC2 e iniciar automaticamente novas instâncias quando necessário. Para obter mais informações, consulte o Guia CloudWatch do usuário da Amazon.
- O Amazon CloudWatch Logs permite que você monitore, armazene e acesse seus arquivos de log a partir de instâncias do Amazon EC2 e de outras fontes. CloudTrail CloudWatch Os registros podem monitorar as informações nos arquivos de log e notificá-lo quando determinados limites forem atingidos. É possível também arquivar seus dados de log em armazenamento resiliente. Para obter mais informações, consulte o Guia do usuário do Amazon CloudWatch Logs.
- AWS CloudTrailcaptura chamadas de API e eventos relacionados feitos por ou em nome de sua AWS conta e entrega os arquivos de log para um bucket do Amazon S3 que você especificar. Você pode identificar quais usuários e contas ligaram AWS, o endereço IP de origem a partir do qual as chamadas foram feitas e quando elas ocorreram. Para mais informações, consulte o Guia do usuário do AWS CloudTrail.

Monitorando a modernização AWS do mainframe com a Amazon CloudWatch

Você pode monitorar a modernização do AWS mainframe usando CloudWatch, que coleta dados brutos e os processa em métricas legíveis, quase em tempo real. Essas estatísticas são mantidas por 15 meses, de maneira que você possa acessar informações históricas e ter uma perspectiva melhor de como o aplicativo web ou o serviço está se saindo. Você também pode definir alarmes que observam determinados limites e enviam notificações ou realizam ações quando esses limites são atingidos. Para obter mais informações, consulte o Guia CloudWatch do usuário da Amazon.

As tabelas a seguir listam as métricas e dimensões da modernização AWS do mainframe. O namespace para essas métricas é `AWS/M2`.

Mainframe Metrics

Métrica	Descrição
CPUUtilization	A utilização da CPU das instâncias no ambiente. Dimensão: environmentId Unidades: percentual Estatísticas válidas: média, mínimo, máximo
InboundNetworkThroughput	Taxa de transferência de rede de entrada de instâncias no ambiente. Dimensão: environmentId Unidades: bytes por segundo Estatísticas válidas: média, mínimo, máximo
MemoryUtilization	A utilização da memória das instâncias no ambiente. Dimensão: environmentId Unidades: percentual Estatísticas válidas: média, mínimo, máximo
OutboundNetworkThroughput	Taxa de transferência de rede de saída das instâncias no ambiente. Dimensão: environmentId Unidades: bytes por segundo

Métrica	Descrição
	Estatísticas válidas: média, mínimo, máximo

Métricas de aplicativo

Métrica	Descrição
BatchJobCompletedCount	O número de trabalhos concluídos durante o intervalo de tempo. Essa métrica está disponível para a Micro Focus e para o AWS Blu Age 3.7.0 e versões posteriores. Dimensões: applicationId Unidades: contagem Estatística válida: soma
BatchJobFailedCount	O número de trabalhos com falha durante o intervalo de tempo. Essa métrica está disponível para a Micro Focus e para o AWS Blu Age 3.7.0 e versões posteriores. Dimensões: applicationId Unidades: contagem Estatística válida: soma
JvmMemoryFree	A quantidade de memória disponível que não está sendo usada atualmente pela Java Virtual Machine. Essa métrica está disponível somente para o mecanismo de tempo de execução do AWS Blu

Métrica	Descrição
	Age. Ele está disponível para o AWS Blu Age 3.7.0 e versões posteriores. Dimensões: applicationId Unidades: bytes Estatísticas válidas: média, mínimo, máximo
JvmMemoryMax	A quantidade máxima de memória permitida para a Java Virtual Machine. Essa métrica está disponível somente para o mecanismo de tempo de execução do AWS Blu Age. Ele está disponível para o AWS Blu Age 3.7.0 e versões posteriores. Dimensões: applicationId Unidades: bytes Estatísticas válidas: média, mínimo, máximo
JvmMemoryUsed	A quantidade de memória usada ativamente pela Máquina Virtual Java. Essa métrica está disponível somente para o mecanismo de tempo de execução do AWS Blu Age. Ele está disponível para o AWS Blu Age 3.7.0 e versões posteriores. Dimensões: applicationId Unidades: bytes Estatísticas válidas: média, mínimo, máximo

Métrica	Descrição
ProcessesActiveCount	O número ativo de processos de execução simultânea de serviços que estão processando solicitações. Essa métrica só está disponível para o mecanismo de runtime da Micro Focus. Dimensões: applicationId Unidades: contagem Estatística válida: soma
SessionCount	O número de sessões HTTP para a aplicação. Essa métrica está disponível somente para o mecanismo de tempo de execução do AWS Blu Age. Ele está disponível para o AWS Blu Age 3.7.0 e versões posteriores. Dimensões: applicationId Unidades: contagem Estatísticas válidas: média, mínimo, máximo
SharedMemoryFree	A memória disponível para que o servidor corporativo armazene todas as informaçõ es necessárias para executar transações e trabalhos. Essa métrica só está disponível para o mecanismo de runtime da Micro Focus. Dimensões: applicationId Unidades: contagem Estatísticas válidas: média, mínimo, máximo

Métrica	Descrição
ThreadActiveCount	O número de threads do mecanismo que estão processando solicitações. Essa métrica está disponível somente para o mecanismo de tempo de execução do AWS Blu Age. Ele está disponível para o AWS Blu Age 3.7.0 e versões posteriores. Dimensões: applicationId Unidades: contagem Estatísticas válidas: média, mínimo, máximo
TransactionCompletedCount	O número de transações confirmadas durante o intervalo de tempo. Essa métrica está disponível para a Micro Focus e para o AWS Blu Age 3.7.0 e versões posteriores. Dimensões: applicationId Unidades: contagem Estatística válida: soma
TransactionFailedCount	O número de transações com falha durante o intervalo de tempo. Essa métrica está disponível para a Micro Focus e para o AWS Blu Age 3.7.0 e versões posteriores. Dimensões: applicationId Unidades: contagem Estatística válida: soma

Métrica	Descrição
TransactionResponseTime	A quantidade de tempo desde o momento em que um usuário envia uma solicitação até o momento em que a aplicação indica que a solicitação foi concluída. Essa métrica está disponível para a Micro Focus e para o AWS Blu Age 3.7.0 e versões posteriores. Dimensões: applicationId Unidade: milissegundos Estatísticas válidas: média, mínimo, máximo

Dimensões

Dimensão	Descrição
applicationId	Essa dimensão filtra a métrica para a aplicação identificada por ID.
environmentId	Essa dimensão filtra a métrica para o ambiente identificado por ID.

Registrando AWS chamadas da API de modernização de mainframe usando AWS CloudTrail

AWS A modernização do mainframe é integrada com AWS CloudTrail um serviço que fornece um registro das ações realizadas por um usuário, função ou AWS serviço na modernização do AWS mainframe. CloudTrail captura todas as chamadas de API para modernização AWS do mainframe como eventos. As chamadas capturadas incluem chamadas do console de modernização do AWS mainframe e chamadas de código para as operações da API de modernização do AWS mainframe. Se você criar uma trilha, poderá habilitar a entrega contínua de CloudTrail eventos para um bucket

do Amazon S3, incluindo eventos para modernização do AWS mainframe. Se você não configurar uma trilha, ainda poderá ver os eventos mais recentes no CloudTrail console no Histórico de eventos. Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita para a modernização do AWS mainframe, o endereço IP do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais.

Para saber mais sobre isso CloudTrail, consulte o Guia AWS CloudTrail do usuário.

AWS Informações sobre modernização do mainframe em CloudTrail

CloudTrail é ativado em sua AWS conta quando você cria a conta. Quando a atividade ocorre na modernização do AWS mainframe, essa atividade é registrada em um CloudTrail evento junto com outros eventos de AWS serviço no histórico de eventos. Você pode visualizar, pesquisar e baixar eventos recentes em sua AWS conta. Para obter mais informações, consulte Visualização de eventos com histórico de CloudTrail eventos.

Para um registro contínuo dos eventos em sua AWS conta, incluindo eventos para modernização do AWS mainframe, crie uma trilha. Uma trilha permite CloudTrail entregar arquivos de log para um bucket do Amazon S3. Por padrão, quando você cria uma trilha no console, ela é aplicada a todas as regiões da AWS. A trilha registra eventos de todas as regiões na AWS partição e entrega os arquivos de log ao bucket do Amazon S3 que você especificar. Além disso, você pode configurar outros AWS serviços para analisar e agir com base nos dados de eventos coletados nos CloudTrail registros. Para mais informações, consulte:

- Visão geral da criação de uma trilha
- CloudTrail serviços e integrações suportados
- Configurando notificações do Amazon SNS para CloudTrail
- Recebendo arquivos de CloudTrail log de várias regiões
- Recebendo arquivos de CloudTrail log de várias contas

Todas as ações de modernização do AWS mainframe são registradas CloudTrail e documentadas na Referência da API de modernização do AWS mainframe. Por exemplo, chamadas para o `CreateApplication` `CreateEnvironment` e `CreateDeployment` as ações geram entradas nos arquivos de CloudTrail log.

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar:

- Se a solicitação foi feita com credenciais de usuário raiz ou credenciais de usuário.
- Se a solicitação foi feita com credenciais de segurança temporárias de um perfil ou de um usuário federado.
- Se a solicitação foi feita por outro AWS serviço.

Para obter mais informações, consulte o elemento CloudTrail userIdentity.

Entendendo as entradas do arquivo de log de modernização do AWS mainframe

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log para um bucket do Amazon S3 que você especificar. CloudTrail os arquivos de log contêm uma ou mais entradas de log. Um evento representa uma única solicitação de qualquer fonte e inclui informações sobre a ação solicitada, a data e a hora da ação, os parâmetros da solicitação e assim por diante. CloudTrail os arquivos de log não são um rastreamento de pilha ordenado das chamadas públicas de API, portanto, eles não aparecem em nenhuma ordem específica.

O exemplo a seguir mostra uma entrada de CloudTrail registro que demonstra a `CreateApplication` ação.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAII6WZTHGYAEXAMPLE",
        "arn": "arn:aws:sts::444455556666:assumed-role/Admin/Mary_Major",
        "accountId": "444455556666",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAII6WZTHGYAEXAMPLE",
                "arn": "arn:aws:iam::444455556666:role/Admin",
                "accountId": "444455556666",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-06-01T20:38:22Z",
                "mfaAuthenticated": "false"
```

```
            }
        }
    },
    "eventTime": "2022-06-01T20:40:39Z",
    "eventSource": "m2.amazonaws.com",
    "eventName": "CreateApplication",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "72.21.196.65",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:91.0) Gecko/20100101
 Firefox/91.0",
    "requestParameters": {
        "clientToken": "1abc23de-f45g-6789-h01i-jkl2m3456789",
        "name": "MyApp",
        "description": "",
        "engineType": "microfocus",
        "definition": {
            "content": "{}"
        },
        "tags": {}
    },
    "responseElements": {
        "applicationVersion": 1,
        "Access-Control-Expose-Headers": "x-amzn-RequestId,x-amzn-ErrorType,x-amzn-
ErrorMessage,Date",
        "applicationArn": "arn:aws:m2:us-east-1:444455556666:app/
lsfhmwhw7fffrosff2lncwqcua",
        "applicationId": "lsfhmwhw7fffrosff2lncwqcua"
    },
    "requestID": "36982d38-fcde-4bfe-a89a-7bd78d43c926",
    "eventID": "d7f0fc36-46ae-4157-9a79-c79f385fda98",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management"
}
```

Solução de problemas

Use as informações desta seção para ajudá-lo a solucionar erros comuns em aplicativos de modernização de AWS mainframe e ambientes de execução usando os mecanismos AWS Blu Age e Micro Focus.

Tópicos

Erro: Tempo limite ao esperar que o nome do conjunto de dados seja desbloqueado

Esta página descreve como você pode resolver seu erro ao ver que outro aplicativo em um ambiente está bloqueando um conjunto de dados compartilhado.

- Motor: AWS Blu Age
- Componente: Blusam

Se você ver esse erro nos CloudWatch registros da Amazon para um aplicativo de modernização de AWS mainframe usando o mecanismo AWS Blu Age e executado em um ambiente com o padrão de alta disponibilidade, isso indica que outro aplicativo está bloqueando um conjunto de dados compartilhado. Normalmente, essa situação ocorre se a outra aplicação falhar ou falhar e não liberar o bloqueio.

Procure um aplicativo com falha e verifique se ele usa o mesmo conjunto de dados mencionado na mensagem de erro. Verifique se a aplicação está sendo executada em um ambiente de runtime com o padrão de alta disponibilidade. A aplicação que gerou a exceção de tempo limite não pode continuar e exibirá o status `Failed`.

Causa comum

A aplicação `example-app-1` tenta bloquear um registro `example-record-1` para uma operação de gravação. Essa operação cria um bloqueio no conjunto de dados`example-dataset-1`, que possui`example-record-1`, e um bloqueio em `example-record-1` si mesmo. Agora, outra aplicação, `example-app-2`, tenta bloquear o mesmo registro `example-record-1`. O conjunto de dados e o registro já estão bloqueados, então `example-app-2` espere que o bloqueio seja liberado. Se `example-app-1` falhar, o bloqueio retido no conjunto de dados `example-dataset-1` ainda existe, o que faz com que `example-app-2` a tentativa de gravação seja cancelada e gere uma exceção de tempo limite. Essa situação de impasse impede que todas as aplicações alcancem `example-dataset-1`.

Resolução

Para resolver a situação imediatamente, você pode forçar a liberação da trava. Para evitar que uma situação semelhante ocorra no futuro, você pode configurar dois parâmetros que controlam o mecanismo de reparo automático Blusam.

Forçar a liberação da trava

O gerenciador de bloqueio Blusam usa o Amazon ElastiCache for Redis para fornecer bloqueios compartilhados entre aplicativos. Para liberar bloqueios ElastiCache, use o utilitário CLI do Redis. Não é possível excluir uma bloqueios de registro individual. Você deve remover todos os bloqueios do conjunto de dados proprietário. Execute as etapas a seguir:

1. Conecte-se ao seu ElastiCache usando o seguinte comando:

   ```
   redis-cli -h hostname -p port
   ```

   Você pode encontrar seus detalhes ElastiCache no ElastiCache console em https://console.aws.amazon.com/elasticache/.
2. Insira a senha.
3. Digite o comando que você deseja executar, da seguinte forma:

Comando	Finalidade
`KEYS *`	Obtenha todas as chaves existentes.

Comando	Finalidade
CHAVES * *YOUR_DATASET_NAME*	Obtenha uma chave de bloqueio do conjunto de dados.
EXCLUA *THE_RETURNED_KEY*	Exclua um bloqueio de conjunto de dados.
FLUSHDB	Limpe todo o Redis. **Warning** Todos os dados no cache do Redis serão perdidos. Se o Redis for usado para outras finalidades, como o tratamento de sessões http, talvez você não queira usar FLUSHDB.

Configure o mecanismo de reparo automático Blusam

O gerenciador de bloqueios Blusam inclui um mecanismo de reparo automático para evitar bloqueios em conjuntos de dados ou registros. Você pode ajustar os seguintes parâmetros na definição da aplicação (`application-main.yml`) para configurar o mecanismo de reparo automático:

- `locksDeadTime`: refere-se ao tempo máximo que uma aplicação pode manter um bloqueio. Quando esse tempo passa, o bloqueio é declarado expirado e liberado imediatamente. O `locksDeadTime` valor está em milissegundos e o valor padrão é 1000.
- `locksCheck`: define a estratégia do gerenciador de bloqueios Blusam para verificar bloqueios. Todos os bloqueios Blusam ElastiCache têm data e hora de validade. O valor do `locksCheck` parâmetro determina se os bloqueios expirados são removidos.
  - `off`: nenhuma verificação é executada em nenhum momento. Podem ocorrer impasses. (Não recomendado)
  - `reboot`: as verificações são executadas quando uma instância do aplicativo de modernização de AWS mainframe em execução em um ambiente de tempo de execução de modernização de AWS mainframe é iniciada ou reinicializada. Todos os bloqueios expirados são liberados imediatamente. (Padrão)

- `timeout`: as verificações são executadas quando uma instância do aplicativo de modernização de AWS mainframe em execução em um ambiente de execução de modernização de AWS mainframe é iniciada ou reinicializada, ou quando um tempo limite expira durante uma tentativa de bloquear um conjunto de dados. Os bloqueios expirados são liberados imediatamente.

Para obter mais informações sobre a definição do aplicativo AWS Blu Age, consulteAWS Exemplo de definição do aplicativo Blu Age.

Gerenciar bloqueios

No contexto de um ambiente de execução de modernização de AWS mainframe usando o padrão de alta disponibilidade, um aplicativo AWS Blu Age pode ser implantado várias vezes. Para as aplicações que lidam com conjuntos de dados Blusam, problemas de acesso simultâneo podem ocorrer. O gerenciador de bloqueios Blusam garante a integridade dos dados e gerencia o acesso de leitura e gravação a registros e conjuntos de dados, fornecendo bloqueios compartilhados entre os aplicativos que usam ElastiCache. Esse mecanismo permite que mais de uma aplicação leia o registro simultaneamente e garante que somente uma aplicação por vez grave o registro.

Bloqueios de gravação

Para atualizar ou excluir um registro específico, a aplicação deve primeiro bloquear o conjunto de dados que possui o registro e, em seguida, bloquear o próprio registro. Quando o registro é bloqueado, o bloqueio do conjunto de dados é liberado e outros registros do mesmo conjunto de dados ficam disponíveis para uso. Quando a operação de atualização ou exclusão for concluída, o bloqueio de registro retido será liberado. Somente uma aplicação por vez pode atualizar o registro, o que impede que outras aplicações leiam ou gravem até que o bloqueio seja liberado, se a política de aplicação definida permitir aguardar a liberação.

Bloqueios de leitura

Desde que nenhum bloqueio de gravação seja mantido no registro ou no conjunto de dados, várias aplicações podem ler os mesmos registros ao mesmo tempo. Para bloquear um registro para uma operação de gravação, todos os bloqueios de leitura devem ser liberados.

> **Note**
>
> O gerenciador de bloqueios Blusam manipula o acesso de vários threads em uma determinada aplicação usando o mesmo mecanismo de bloqueio.

Não consigo acessar o URL de um aplicativo

Esta página descreve como você pode resolver seu erro quando não consegue acessar o URL de um aplicativo de modernização de AWS mainframe em execução.

- Motor: AWS Blu Age e Micro Focus
- Componente: aplicações

Se você não conseguir acessar a URL de um aplicativo de modernização de AWS mainframe em execução que você criou e implantou em um ambiente de tempo de execução de modernização de AWS mainframe, talvez seja necessário configurar as regras de entrada no grupo de segurança que você associou ao ambiente de tempo de execução.

Causa comum

Quando você cria um ambiente de runtime, o grupo de segurança fornecido, incluindo o grupo de segurança padrão, deve ter regras de entrada configuradas para permitir o tráfego para as aplicações implantadas de fora da VPC, se você quiser permitir esse tipo de acesso.

Resolução

Verifique se o grupo de segurança da Amazon VPC associado ao ambiente de runtime permite tráfego para o ambiente nas portas apropriadas da aplicação. Para verificar as regras do grupo de segurança, conclua as etapas a seguir:

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. Na barra de navegação à esquerda, selecione Ambientes.
3. Escolha o ambiente de runtime que hospeda a aplicação ao qual você deseja conectar-se.
4. Escolha Configurações.
5. Em Segurança e rede, escolha o grupo de segurança. O link abre os detalhes do grupo de segurança no console da Amazon VPC.
6. Se necessário, escolha Editar regras de entrada e adicione a seguinte regra, se ainda não estiver presente:

   Tipo

   TCP personalizado

Port (Porta)

8196 ou a porta que corresponde às propriedades do receptor especificadas na definição da aplicação. Para ter mais informações, consulte Etapa 2: criar a aplicação.

Origem

O endereço IP de onde você está chamando a aplicação. Você pode escolher myIP no menu suspenso. Se você ainda tiver problemas de tempo limite, tente escolher Anywhere IPV4 ou Anywhere IPV6. Certifique-se de interromper a aplicação e iniciá-la novamente depois de adicionar a regra de entrada no grupo de segurança.

Para obter mais informações, consulte Trabalhar com regras de grupo de segurança no Guia do usuário da Amazon VPC.

AWS O Blu Insights não abre no console

Esta página descreve como você pode resolver a página do Blu Insights que não está sendo aberta a partir do console de modernização do AWS mainframe.

- Motor: AWS Blu Age
- Componente: Blu Insights

Quando você tenta acessar o Blu Insights a partir do console de modernização do AWS Mainframe, ele não abre e a nova guia é fechada imediatamente.

Causa comum

A função que você está usando para acessar o Blu Insights não tem permissões suficientes.

Resolução

Anexe uma política do IAM à função para permitir que ela acesse o Blu Insights. Certifique-se de que a política inclua pelo menos as seguintes permissões.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                "m2:GetSignedBluinsightsUrl"
            ],
            "Resource": "*"
            }
    ]
}
```

Certifique-se de substituir `region` e `account` com o Região da AWS Conta da AWS e. correto

Ambiente insalubre

Esta página descreve como você pode resolver seu erro ao receber uma notificação de que um de seus ambientes de modernização de AWS mainframe não está íntegro.

- Motor: AWS Blu Age e Micro Focus
- Componente: ambientes

Se você receber uma notificação informando que um dos seus ambientes de modernização de AWS mainframe não está íntegro, isso se aplica a você. Você é notificado por meio de uma dessas fontes:

- O status do ambiente não íntegro é mostrado no console de modernização AWS do mainframe.
- Notificação por e-mail sobre o status do ambiente insalubre de AWS Health.
- Você vê um evento relacionado da Modernização do AWS Mainframe em seu AWS Health painel, em Integridade da sua conta.

Causa comum

O erro ocorre quando os recursos da sua AWS conta associados ao ambiente de modernização do AWS mainframe estão inacessíveis. Um motivo comum para esse problema é que os recursos relacionados ao ambiente estão sendo modificados ou excluídos.

Resolução

Para obter orientação específica, use o código de erro fornecido no e-mail do AWS Health ou por meio do console de modernização de AWS mainframe.

Código de erro:

- Armazenamento inacessível

Esse erro indica que o armazenamento conectado (sistemas de arquivos Amazon Elastic File System ou Amazon FSx) do ambiente falhou ao ser montado corretamente. Para verificar detalhes sobre um ambiente insalubre, conclua as seguintes etapas:

1. Abra o console de modernização do AWS mainframe em https://console.aws.amazon.com/m2/.
2. Selecione o ambiente não íntegro e escolha Configuração.
3. Escolha Attached Storage para visualizar os recursos de armazenamento associados a esse ambiente.
4. Verifique as configurações relacionadas à rede, como o grupo de segurança, a sub-rede e a Amazon VPC associada ao armazenamento. Se essas configurações estiverem incorretas, tente restaurá-las para resolver esse problema.

> **Note**
>
> Se o armazenamento tiver sido excluído, o ambiente não poderá ser recuperado. Nesse caso, você deve considerar a exclusão do ambiente insalubre.

Histórico de documentos do Guia do usuário da modernização do AWS mainframe

A tabela a seguir descreve as versões da documentação para a modernização AWS do mainframe.

Alteração	Descrição	Data
Versão do Application Testing GA	Documentos de disponibilidade geral para testes de aplicativos. AWS O Mainframe Modernization Application Testing fornece testes automatizados de equivalência funcional para seus projetos de migração. Essa versão inclui a página de proteção de dados, fluxos de trabalho do console e atualizações em outras páginas de documentos desde a pré-visualização.	12 de junho de 2024
Tutorial atualizado do Managed Runtime para Micro Focus	Este tutorial mostra como implantar e executar o aplicativo de CardDemo amostra em um ambiente de tempo de execução gerenciado de modernização de AWS mainframe com o mecanismo de tempo de execução da Micro Focus.	5 de fevereiro de 2024
Notas de lançamento do AWS Blu Age Runtime and Modernization Tools versão 3.9.0.	Esta versão do AWS Blu Age Runtime and Modernization Tools está focada em vários aprimoramentos transversais em todo o produto, buscando	18 de dezembro de 2023

	aumentar o desempenh o em arquiteturas de alta disponibilidade, juntament e com novos recursos para elevar a execução de tarefas a um novo patamar.	
Transferir arquivos entre mainframe e AWS	Novo recurso lançado para transferir arquivos do mainframe de origem para a AWS.	27 de novembro de 2023
Gerencie transações para aplicações	Novo recurso lançado para exibir e editar transações de aplicações para o AWS Mainframe Modernization.	16 de outubro de 2023
Notas de lançamento do AWS Blu Age Runtime and Modernization Tools versão 3.6.0.	Esta versão do AWS Blu Age Runtime and Moderniza tion Tools fornece novos recursos para migrações legadas do zOS e do AS400, principalmente orientadas para expandir os mecanismos de suporte do CICS, complemen tar os recursos da JCL, otimizar o desempenho em recursos simultâneos e de alto volume e adicionar recursos. multi-data-source	4 de agosto de 2023

Agora você poderá implantar uma nova versão de uma aplicação quando ela for interrompida.	Anteriormente, para implantar uma nova versão de uma aplicação, era necessário excluir a versão implantada. Agora você pode simplesme nte interromper a versão implantada e implantar uma nova versão.	26 de julho de 2023
AWS Runtime do Blu Age empacotado para uma implantação mais fácil do Amazon EC2	AWS A modernização do mainframe com o tempo de execução AWS Blu Age agora está disponível com mais flexibilidade para configurar a pilha completa e a implantaç ão em instâncias do Amazon EC2 em seu. Conta da AWS	6 de julho de 2023
Login único no Blu Age AWS Blu Insights.	AWS O Blu Age Blu Insights está disponível AWS Management Console por meio de login único.	31 de março de 2023
Lançamento do GA	Versão GA do Guia do Usuário da Modernização do AWS Mainframe.	8 de junho de 2022
Lançamento inicial	Versão inicial (prévia pública) do Guia do usuário de modernização do AWS mainframe.	30 de novembro de 2021

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.