Kerangka kerja, platform, protokol, dan alat AI Agentic di AWS

AWS Panduan Preskriptif

AWS Panduan Preskriptif: Kerangka kerja, platform, protokol, dan alat AI Agentic di AWS

Table of Contents

Kerangka kerja, platform, protokol, dan alat AI Agentic di AWS

Aaron Sempf, Ansley Verzosa, dan Joshua Samuel, Amazon Web Services ()AWS

Januari 2026 (sejarah dokumen)

Agentic AI adalah paradigma yang kuat di persimpangan AI, sistem terdistribusi, dan rekayasa perangkat lunak. Ini adalah kelas sistem cerdas yang terdiri dari agen perangkat lunak asinkron otonom yang menggunakan model AI dan terintegrasi dengan alat dan sumber daya. Agen pameran, dapat memahami konteks, alasan atas tujuan, membuat keputusan, dan mengambil tindakan yang disengaja atas nama pengguna atau sistem. Agen-agen ini beroperasi secara independen, seringkali secara kolaboratif, dalam lingkungan terdistribusi dan dirancang untuk mengejar tujuan yang didelegasikan dengan kecerdasan, memori, dan niat yang tertanam.

Pada AWS, organisasi dapat memanfaatkan AI agen untuk mengotomatiskan alur kerja yang kompleks, meningkatkan proses pengambilan keputusan, dan menciptakan sistem yang lebih responsif. Panduan ini memberikan informasi tentang komponen utama yang diperlukan untuk membangun solusi AI agen yang efektif:

- Kerangka kerja memprofilkan kerangka kerja AI agen saat ini, termasuk ulasan tentang manfaat dan kasus penggunaannya. Pelajari bagaimana kerangka kerja ini mengurangi pengangkatan berat yang tidak berdiferensiasi di seluruh pola, protokol, dan alat. Pahami kriteria pemilihan utama untuk memilih kerangka kerja yang tepat untuk kebutuhan Anda.
- Platform memberikan gambaran umum tentang platform AI agen (agen terkelola, orkestrasi sumber terbuka, dan hibrida) dan pertimbangan untuk pemilihan atau desain.
- Protokol mengeksplorasi protokol komunikasi standar penting untuk interaksi agen. Agent-to-agentprotokol yang muncul, seperti open source Model Context Protocol (MCP) dan Agent2Agent (A2A), bersama dengan implementasi proprietary lainnya. Temukan bagaimana protokol umum memungkinkan protokol yang berbeda untuk berinteraksi dengan mulus.
- Tools menyediakan informasi tentang alat berbasis protokol (seperti MCP), framework-native tools, dan meta-tools. Organizations dapat membangun toolkit yang terintegrasi dengan sistem kunci dalam alur kerja mereka, memungkinkan alur kerja agen pengguna akhir dan berbasis server.

Audiens yang dituju

Panduan ini ditujukan untuk arsitek, pengembang, dan pemimpin teknologi yang ingin memanfaatkan kekuatan agen perangkat lunak berbasis AI dalam aplikasi cloud-native modern.

Tujuan

Panduan ini membantu Anda melakukan hal berikut:

- Bandingkan kerangka kerja AI agen yang berbeda untuk memilih yang paling tepat untuk kasus penggunaan Anda.
- Pelajari tentang platform AI agen yang menyediakan kemampuan untuk mengubah agen individu menjadi sistem adaptif yang terkoordinasi.
- Memahami keuntungan dari protokol terbuka untuk membangun arsitektur AI agen yang berkelanjutan.
- Buat strategi integrasi alat yang tepat saat membangun sistem agen.

Tentang seri konten ini

Panduan ini adalah bagian dari seri tentang AI agen di AWS. Untuk informasi lebih lanjut dan untuk melihat panduan lain dalam seri ini, lihat Agentic AI di situs web AWS Prescriptive Guidance.

Kerangka Kerja

Fondasi AI agen pada AWS memeriksa pola inti dan alur kerja yang memungkinkan perilaku otonom yang diarahkan pada tujuan. Inti penerapan pola-pola ini terletak pada pilihan kerangka kerja. Kerangka kerja adalah fondasi perangkat lunak dari kode yang telah ditulis sebelumnya yang menyediakan lingkungan terstruktur dan fungsionalitas umum untuk membangun dan mengelola, alat, dan kemampuan orkestrasi yang diperlukan untuk membangun agen AI otonom siap produksi.

Kerangka kerja AI agen yang efektif menyediakan beberapa kemampuan penting yang mengubah interaksi model bahasa besar mentah (LLM) menjadi sistem terkoordinasi dan cerdas yang mampu melakukan penalaran, kolaborasi, dan tindakan:

- Orkestrasi agen mengoordinasikan arus informasi dan pengambilan keputusan di seluruh agen tunggal atau ganda untuk mencapai tujuan yang kompleks tanpa campur tangan manusia.
- Integrasi alat memungkinkan agen untuk berinteraksi dengan sistem eksternal, API, dan sumber data untuk memperluas kemampuan mereka di luar pemrosesan bahasa. Untuk informasi selengkapnya, lihat Ikhtisar Alat dalam Strands Agents dokumentasi.
- Manajemen memori menyediakan keadaan persisten atau berbasis sesi untuk mempertahankan konteks lintas interaksi, penting untuk tugas yang berjalan lama atau adaptif. Kerangka kerja yang lebih canggih menggabungkan memori jangka panjang untuk menyimpan ringkasan dan preferensi pengguna, memungkinkan pengalaman agen yang dipersonalisasi dan sadar kontekstual. Untuk informasi selengkapnya, lihat Cara berpikir tentang kerangka kerja agen di LangChain Blog.
- Definisi alur kerja mendukung pola terstruktur seperti rantai, perutean, paralelisasi, dan loop refleksi yang memungkinkan penalaran otonom yang canggih.
- Penyebaran dan pemantauan memfasilitasi transisi dari pengembangan ke produksi dengan observabilitas untuk sistem otonom. Untuk informasi selengkapnya, lihat pengumuman ketersediaan AgentCore umum Amazon Bedrock.

Kemampuan ini diimplementasikan dengan berbagai pendekatan dan penekanan di seluruh lanskap kerangka kerja, masing-masing menawarkan keuntungan berbeda untuk kasus penggunaan agen otonom dan konteks organisasi yang berbeda.

Bagian ini membuat profil dan membandingkan kerangka kerja terkemuka untuk membangun solusi AI agen, dengan fokus pada kekuatan, keterbatasan, dan kasus penggunaan ideal untuk operasi otonom:

- Agen Helai
- LangChain dan LangGraph
- CrewAI
- AutoGen
- ???
- Membandingkan kerangka kerja AI agen

> **Note**
>
> Bagian ini mencakup kerangka kerja yang secara khusus mendukung agensi AI dan tidak mencakup antarmuka frontend atau AI generatif tanpa agensi.

Strands Agents

Strands Agentsadalah SDK open-source yang awalnya dirilis oleh AWS, seperti yang dijelaskan dalam AWS Open Source Blog. Strands Agentsdirancang untuk membangun agen AI otonom dengan pendekatan model-first Ini menyediakan kerangka kerja yang fleksibel dan dapat diperluas yang dirancang untuk bekerja dengan mulus Layanan AWS sambil tetap terbuka untuk integrasi dengan komponen pihak ketiga. Strands Agents sangat ideal untuk membangun solusi yang sepenuhnya otonom.

Fitur utama dari Strands Agents

Strands Agentstermasuk fitur utama berikut:

- Model-first desain — Dibangun di sekitar konsep bahwa model fondasi adalah inti dari intelijen agen, memungkinkan penalaran otonom yang canggih. Untuk informasi selengkapnya, lihat Agent Loop dalam Strands Agents dokumentasi.
- Multi-agent pola kolaborasi — model Built-in koordinasi seperti pola Swarm, Graph, dan Workflow yang memungkinkan kolaborasi dan tata kelola yang dapat diskalakan di seluruh jaringan agen terdistribusi. Untuk informasi selengkapnya, lihat Multi-agent Pola dalam dokumentasi Strands Agents.
- Integrasi MCP — Dukungan asli untuk Model Context Protocol (MCP), memungkinkan penyediaan konteks standar ke LLM untuk operasi otonom yang konsisten.

- Layanan AWS integrasi - Koneksi tanpa batas ke Amazon Bedrock,, AWS Lambda AWS Step Functions, dan lainnya Layanan AWS untuk alur kerja otonom yang komprehensif. Untuk informasi lebih lanjut, lihat Roundup AWS Mingguan (AWS Blog).
- Pemilihan model foundation - Mendukung berbagai model pondasi termasuk Anthropic Claude, Amazon Nova (Premier, Pro, Lite, dan Micro) di Amazon Bedrock, dan lainnya untuk mengoptimalkan kemampuan penalaran otonom yang berbeda. Untuk informasi selengkapnya, lihat Amazon Bedrock di Strands Agents dokumentasi.
- Integrasi LLM API - Integrasi fleksibel dengan antarmuka layanan LLM yang berbeda termasuk Amazon Bedrock, OpenAI, dan lainnya untuk penyebaran produksi. Untuk informasi selengkapnya, lihat Amazon Bedrock Basic Usage dalam Strands Agents dokumentasi.
- Kemampuan multimodal — Dukungan untuk berbagai modalitas termasuk pemrosesan teks, ucapan, dan gambar untuk interaksi agen otonom yang komprehensif. Untuk informasi selengkapnya, lihat Amazon Bedrock Multimodal Support di dokumentasi. Strands Agents
- Ekosistem alat - Kumpulan alat yang kaya untuk Layanan AWS interaksi, dengan ekstensibilitas untuk alat khusus yang memperluas kemampuan otonom. Untuk informasi selengkapnya, lihat Ikhtisar Alat dalam Strands Agents dokumentasi.

Kapan harus menggunakan Strands Agents

Strands Agentssangat cocok untuk skenario agen otonom termasuk:

- Organizations yang membangun AWS infrastruktur yang menginginkan integrasi asli dengan Layanan AWS alur kerja otonom
- Tim yang memerlukan fitur keamanan, skalabilitas, dan kepatuhan tingkat perusahaan untuk sistem otonom produksi
- Proyek yang membutuhkan fleksibilitas dalam pemilihan model di berbagai penyedia untuk tugas otonom khusus
- Gunakan kasus yang memerlukan integrasi ketat dengan AWS alur kerja dan sumber daya yang ada untuk proses otonom ujung ke ujung

Pendekatan implementasi untuk Strands Agents

Strands Agentsmenyediakan pendekatan implementasi langsung untuk pemangku kepentingan bisnis, sebagaimana diuraikan dalam Panduan Mulai Cepat untuk dan Panduan Mulai Cepat untuk. PythonTypescript Kerangka kerja ini memungkinkan organisasi untuk:

- Pilih model foundation seperti Amazon Nova (Premier, Pro, Lite, atau Micro) di Amazon Bedrock berdasarkan persyaratan bisnis tertentu.
- Tentukan alat khusus yang terhubung ke sistem perusahaan dan sumber data.
- Memproses beberapa modalitas termasuk teks, gambar, dan ucapan.
- Menyebarkan agen yang dapat secara mandiri menanggapi pertanyaan bisnis dan melakukan tugas.

Pendekatan implementasi ini memungkinkan tim bisnis untuk dengan cepat mengembangkan dan menyebarkan agen otonom tanpa keahlian teknis yang mendalam dalam pengembangan model AI.

Real-world contoh dari Strands Agents

AWS Transform untuk penggunaan .NET Strands Agents untuk memperkuat kemampuan modernisasi aplikasinya, seperti yang dijelaskan dalam AWS Transform untuk .NET, layanan AI agen pertama untuk memodernisasi aplikasi.NET dalam skala besar (Blog).AWS Layanan produksi ini mempekerjakan beberapa agen otonom khusus. Para agen bekerja sama untuk menganalisis aplikasi.NET lama, merencanakan strategi modernisasi, dan mengeksekusi transformasi kode ke arsitektur cloud-native tanpa campur tangan manusia. AWS Transform untuk .NET menunjukkan kesiapan produksi Strands Agents untuk sistem otonom perusahaan.

LangChain dan LangGraph

LangChainadalah salah satu kerangka kerja paling mapan dalam ekosistem AI agen. LangGraphmemperluas kemampuannya untuk mendukung alur kerja agen yang kompleks dan stateful seperti yang dijelaskan dalam Blog. LangChain Bersama-sama, mereka memberikan solusi komprehensif untuk membangun agen AI otonom yang canggih dengan kemampuan orkestrasi yang kaya untuk operasi independen.

Fitur utama LangChain dan LangGraph

LangChaindan LangGraph termasuk fitur utama berikut:

- Ekosistem komponen — Perpustakaan ekstensif komponen pra-bangun untuk berbagai kemampuan agen otonom, memungkinkan pengembangan agen khusus yang cepat. Untuk informasi selengkapnya, lihat Mulai cepat di LangChain dokumentasi.
- Pemilihan model foundation - Support untuk beragam model pondasi termasuk Anthropic Claude, Amazon Nova model (Premier, Pro, Lite, dan Micro) di Amazon Bedrock, dan lainnya untuk

kemampuan penalaran yang berbeda. Untuk informasi selengkapnya, lihat Input dan output dalam dokumentasi. LangChain

- Integrasi API LLM — Antarmuka standar untuk beberapa penyedia layanan model bahasa besar (LLM) termasuk Amazon Bedrock,OpenAI, dan lainnya untuk penerapan yang fleksibel. Untuk informasi selengkapnya, lihat LLM di LangChain dokumentasi.
- Pemrosesan multimodal — Built-in dukungan untuk pemrosesan teks, gambar, dan audio untuk memungkinkan interaksi agen otonom multimodal yang kaya. Untuk informasi selengkapnya, lihat Multimodalitas dalam dokumentasi. LangChain
- Graph-based alur kerja — LangGraph memungkinkan mendefinisikan perilaku agen otonom yang kompleks sebagai mesin negara, mendukung logika keputusan yang canggih. Untuk informasi selengkapnya, lihat pengumuman LangGraphPlatform GA.
- Abstraksi memori — Beberapa opsi untuk manajemen memori jangka pendek dan jangka panjang, yang penting untuk agen otonom yang mempertahankan konteks dari waktu ke waktu. Untuk informasi selengkapnya, lihat Cara menambahkan memori ke chatbots dalam dokumentasi. LangChain
- Integrasi alat — Ekosistem integrasi alat yang kaya di berbagai layanan dan API, memperluas kemampuan agen otonom. Untuk informasi selengkapnya, lihat Alat dalam LangChain dokumentasi.
- LangGraph platform - Solusi penyebaran dan pemantauan terkelola untuk lingkungan produksi, mendukung agen otonom yang sudah berjalan lama. Untuk informasi selengkapnya, lihat pengumuman LangGraphPlatform GA.

Kapan menggunakan LangChain dan LangGraph

LangChaindan LangGraph sangat cocok untuk skenario agen otonom termasuk:

- Alur kerja penalaran multi-langkah kompleks yang membutuhkan orkestrasi canggih untuk pengambilan keputusan otonom
- Proyek yang membutuhkan akses ke ekosistem besar komponen prebuilt dan integrasi untuk beragam kemampuan otonom
- Tim dengan infrastruktur dan keahlian machine learning (ML) Python berbasis yang ada yang ingin membangun sistem otonom
- Gunakan kasus yang memerlukan manajemen negara yang kompleks di seluruh sesi agen otonom yang berjalan lama

Pendekatan implementasi untuk LangChain dan LangGraph

LangChaindan LangGraph memberikan pendekatan implementasi terstruktur untuk pemangku kepentingan bisnis, sebagaimana dirinci dalam dokumentasi. LangGraph Kerangka kerja ini memungkinkan organisasi untuk:

- Tentukan grafik alur kerja canggih yang mewakili proses bisnis.
- Buat pola penalaran multi-langkah dengan poin keputusan dan logika bersyarat.
- Mengintegrasikan kemampuan pemrosesan multimodal untuk menangani beragam tipe data.
- Menerapkan kontrol kualitas melalui mekanisme peninjauan dan validasi bawaan.

Pendekatan berbasis grafik ini memungkinkan tim bisnis untuk memodelkan proses keputusan yang kompleks sebagai alur kerja otonom. Tim memiliki visibilitas yang jelas ke dalam setiap langkah proses penalaran dan kemampuan untuk mengaudit jalur keputusan.

Real-world contoh LangChain dan LangGraph

Vodafonetelah menerapkan agen otonom menggunakan LangChain (danLangGraph) untuk meningkatkan rekayasa data dan alur kerja operasinya, sebagaimana dirinci dalam studi kasus LangChain Enterprise mereka. Mereka membangun asisten AI internal yang secara mandiri memantau metrik kinerja, mengambil informasi dari sistem dokumentasi, dan menyajikan wawasan yang dapat ditindaklanjuti — semuanya melalui interaksi bahasa alami.

VodafoneImplementasinya menggunakan pemuat dokumen LangChain modular, integrasi vektor, dan dukungan untuk beberapa LLM (OpenAI, LLaMA 3, danGemini) untuk membuat prototipe dan benchmark jaringan pipa ini dengan cepat. Mereka kemudian digunakan LangGraph untuk menyusun orkestrasi multi-agen dengan menggunakan sub agen modular. Agen-agen ini melakukan tugas pengumpulan, pemrosesan, peringkasan, dan penalaran. LangGraphmengintegrasikan agen-agen ini melalui API ke dalam sistem cloud mereka.

CrewAI

CrewAIadalah kerangka kerja sumber terbuka yang berfokus secara khusus pada orkestrasi multi-agen otonom, tersedia di. GitHub Ini memberikan pendekatan terstruktur untuk menciptakan tim agen otonom khusus yang berkolaborasi untuk menyelesaikan tugas-tugas kompleks tanpa campur tangan manusia. CrewAImenekankan koordinasi berbasis peran dan delegasi tugas.

Fitur utama dari CrewAI

CrewAImenyediakan fitur utama berikut:

- Role-based desain agen — Agen otonom didefinisikan dengan peran, tujuan, dan cerita belakang tertentu untuk memungkinkan keahlian khusus. Untuk informasi selengkapnya, lihat Membuat Agen Efektif dalam CrewAI dokumentasi.
- Delegasi tugas — Built-in mekanisme untuk secara mandiri menetapkan tugas kepada agen yang sesuai berdasarkan kemampuan mereka. Untuk informasi selengkapnya, lihat Tugas dalam CrewAI dokumentasi.
- Kolaborasi agen — Kerangka kerja untuk komunikasi antar-agen otonom dan berbagi pengetahuan tanpa mediasi manusia. Untuk informasi selengkapnya, lihat Kolaborasi dalam CrewAI dokumentasi.
- Manajemen proses — Alur kerja terstruktur untuk eksekusi tugas otonom sekuensial dan paralel. Untuk informasi selengkapnya, lihat Proses dalam CrewAI dokumentasi.
- Pemilihan model foundation — Support untuk berbagai model foundation termasuk Anthropic Claude, Amazon Nova model (Premier, Pro, Lite, dan Micro) di Amazon Bedrock, dan lainnya untuk mengoptimalkan tugas penalaran otonom yang berbeda. Untuk informasi selengkapnya, lihat LLM di CrewAI dokumentasi.
- Integrasi API LLM - Integrasi fleksibel dengan beberapa antarmuka layanan LLM termasuk Amazon Bedrock,OpenAI, dan penerapan model lokal. Untuk informasi selengkapnya, lihat Contoh Konfigurasi Penyedia dalam CrewAI dokumentasi.
- Dukungan multimodal — Kemampuan yang muncul untuk menangani teks, gambar, dan modalitas lainnya untuk interaksi agen otonom yang komprehensif. Untuk informasi selengkapnya, lihat Menggunakan Agen Multimodal dalam CrewAI dokumentasi.

Kapan harus menggunakan CrewAI

CrewAIsangat cocok untuk skenario agen otonom termasuk:

- Masalah kompleks yang mendapat manfaat dari keahlian khusus berbasis peran yang bekerja secara mandiri
- Proyek yang membutuhkan kolaborasi eksplisit antara beberapa agen otonom
- Gunakan kasus di mana dekomposisi masalah berbasis tim meningkatkan pemecahan masalah otonom

- Skenario yang membutuhkan pemisahan kekhawatiran yang jelas antara peran agen otonom yang berbeda

Pendekatan implementasi untuk CrewAI

CrewAImenyediakan implementasi berbasis peran pendekatan tim agen AI untuk pemangku kepentingan bisnis, sebagaimana dirinci dalam Memulai dalam dokumentasi. CrewAI Kerangka kerja ini memungkinkan organisasi untuk:

- Tentukan agen otonom khusus dengan peran, tujuan, dan bidang keahlian tertentu.
- Tetapkan tugas kepada agen berdasarkan kemampuan khusus mereka.
- Menetapkan dependensi yang jelas antara tugas untuk membuat alur kerja terstruktur.
- Mengatur kolaborasi antara beberapa agen untuk memecahkan masalah yang kompleks.

Pendekatan berbasis peran ini mencerminkan struktur tim manusia, sehingga intuitif bagi para pemimpin bisnis untuk memahami dan menerapkan. Organizations dapat membuat tim otonom dengan bidang keahlian khusus yang berkolaborasi untuk mencapai tujuan bisnis, mirip dengan bagaimana tim manusia beroperasi. Namun, tim otonom dapat bekerja terus menerus tanpa campur tangan manusia.

Real-world contoh dari CrewAI

AWS telah menerapkan sistem multi-agen otonom menggunakan CreWai yang terintegrasi dengan Amazon Bedrock, sebagaimana dirinci dalam studi kasus yang CrewAI diterbitkan. AWS dan CrewAI mengembangkan kerangka kerja yang aman dan netral vendor. Arsitektur “aliran dan kru” CrewAI sumber terbuka terintegrasi dengan mulus dengan model fondasi Amazon Bedrock, sistem memori, dan pagar pembatas kepatuhan.

Elemen kunci dari implementasi meliputi:

- Cetak biru dan sumber terbuka — AWS dan CrewAI merilis desain referensi yang memetakan agen CrewAI ke model Amazon Bedrock dan alat observabilitas. Mereka juga merilis sistem contoh seperti kru audit AWS keamanan multi-agen, arus modernisasi kode, dan otomatisasi back-office consumer packaged goods (CPG).
- Integrasi tumpukan observabilitas — Solusi ini menyematkan pemantauan dengan Amazon CloudWatch, dan AgentOpsLangFuse, memungkinkan penelusuran dan debugging dari pembuktian konsep hingga produksi.

- Demonstrated Return on Investment (ROI) — Pilot awal menunjukkan peningkatan besar — eksekusi 70 persen lebih cepat untuk proyek modernisasi kode besar dan sekitar 90 persen pengurangan waktu pemrosesan untuk aliran back-office CPG.

AutoGen

AutoGenadalah kerangka kerja sumber terbuka yang dirilis awalnya olehMicrosoft. AutoGenberfokus pada mengaktifkan agen AI otonom percakapan dan kolaboratif. Ini menyediakan arsitektur yang fleksibel untuk membangun sistem multi-agen dengan penekanan pada interaksi asinkron, berbasis peristiwa antara agen untuk alur kerja otonom yang kompleks.

Fitur utama dari AutoGen

AutoGenmenyediakan fitur utama berikut:

- Agen percakapan — Dibangun di sekitar percakapan bahasa alami antara agen otonom, memungkinkan penalaran canggih melalui dialog. Untuk informasi selengkapnya, lihat Kerangka Multi-agent Percakapan dalam AutoGen dokumentasi.
- Arsitektur asinkron — Event-driven desain untuk interaksi agen otonom non-pemblokiran, mendukung alur kerja paralel yang kompleks. Untuk informasi selengkapnya, lihat Memecahkan Beberapa Tugas dalam Urutan Obrolan Async dalam dokumentasi. AutoGen
- Human-in-the-loop— Dukungan kuat untuk partisipasi manusia opsional dalam alur kerja agen otonom bila diperlukan. Untuk informasi selengkapnya, lihat Mengizinkan Umpan Balik Manusia di Agen dalam AutoGen dokumentasi.
- Pembuatan dan eksekusi kode — Kemampuan khusus untuk agen otonom yang berfokus pada kode yang dapat menulis dan menjalankan kode. Untuk informasi selengkapnya, lihat Eksekusi Kode dalam AutoGen dokumentasi.
- Perilaku yang dapat disesuaikan - Konfigurasi agen otonom yang fleksibel dan kontrol percakapan untuk beragam kasus penggunaan. Untuk informasi selengkapnya, lihat agentchat.conversable_agent di dokumentasi. AutoGen
- Pemilihan model foundation — Support untuk berbagai model foundation termasuk Anthropic Claude, Amazon Nova model (Premier, Pro, Lite, dan Micro) di Amazon Bedrock, dan lainnya untuk kemampuan penalaran otonom yang berbeda. Untuk informasi selengkapnya, lihat Konfigurasi LLM dalam AutoGen dokumentasi.

- Integrasi API LLM - Konfigurasi standar untuk beberapa antarmuka layanan LLM termasuk Amazon Bedrock,, dan. OpenAI Azure OpenAI Untuk informasi selengkapnya, lihat oai.openai_utils di Referensi API. AutoGen
- Pemrosesan multimodal — Dukungan untuk pemrosesan teks dan gambar untuk memungkinkan interaksi agen otonom multimodal yang kaya. Untuk informasi selengkapnya, lihat Terlibat dengan Model Multimodal: GPT-4V AutoGen dalam dokumentasi. AutoGen

Kapan harus menggunakan AutoGen

AutoGensangat cocok untuk skenario agen otonom termasuk:

- Aplikasi yang membutuhkan aliran percakapan alami antara agen otonom untuk penalaran yang kompleks
- Proyek yang membutuhkan operasi otonom penuh dan kemampuan pengawasan manusia opsional
- Gunakan kasus yang melibatkan pembuatan kode otonom, eksekusi, dan debugging tanpa campur tangan manusia
- Skenario yang membutuhkan pola komunikasi agen otonom yang fleksibel dan asinkron

Pendekatan implementasi untuk AutoGen

AutoGenmenyediakan pendekatan implementasi percakapan untuk pemangku kepentingan bisnis, sebagaimana dirinci dalam Memulai dalam dokumentasi. AutoGen Kerangka kerja ini memungkinkan organisasi untuk:

- Buat agen otonom yang berkomunikasi melalui percakapan bahasa alami.
- Menerapkan interaksi asinkron yang digerakkan oleh peristiwa antara beberapa agen.
- Gabungkan operasi otonom penuh dengan pengawasan manusia opsional bila diperlukan.
- Mengembangkan agen khusus untuk berbagai fungsi bisnis yang berkolaborasi melalui dialog.

Pendekatan percakapan ini membuat penalaran sistem otonom transparan dan dapat diakses oleh pengguna bisnis. Decision-makers dapat mengamati dialog antara agen untuk memahami bagaimana kesimpulan dicapai dan secara opsional berpartisipasi dalam percakapan ketika penilaian manusia diperlukan.

Real-world contoh dari AutoGen

Magentic-Oneadalah sistem multi-agen generalis open source yang dirancang untuk menyelesaikan tugas multi-langkah yang kompleks secara mandiri di berbagai lingkungan, seperti yang dijelaskan dalam blog AI Frontiers. Microsoft Pada intinya adalah agen Orchestrator, yang menguraikan tujuan tingkat tinggi dan melacak kemajuan dengan menggunakan buku besar terstruktur. Agen ini mendelegasikan subtugas kepada agen khusus (sepertiWebSurfer,, FileSurferCoder, danComputerTerminal) dan beradaptasi secara dinamis dengan perencanaan ulang bila diperlukan.

Sistem ini dibangun di atas AutoGen kerangka kerja dan model-agnostik, default ke GPT-4o. Ini mencapai kinerja mutakhir di seluruh tolok ukur seperti,, dan —semuanya tanpa penyetelan khusus tugas. GAIA AssistantBench WebArena Selain itu, mendukung ekstensibilitas modular dan evaluasi yang ketat melalui saran. AutoGenBench

LlamaIndex

LlamaIndexadalah kerangka data yang dirancang khusus untuk menghubungkan model bahasa besar (LLM) dengan sumber data eksternal untuk memungkinkan Retrieval Augmented Generation (RAG) dan aplikasi AI agen yang canggih. Kerangka kerja ini menyediakan abstraksi dan alur kerja pengembangan yang dipercepat untuk sistem agen, pola orkestrasi khusus, dan integrasi sistem yang mengurangi waktu produksi untuk solusi AI berbasis pengetahuan.

Fitur utama dari LlamaIndex

LlamaIndexmenyediakan serangkaian kemampuan komprehensif yang membuatnya sangat cocok untuk aplikasi AI agen perusahaan:

- Data-centric Arsitektur — Unggul dalam menelan, mengindeks, dan mengambil informasi dari lebih dari 100 format data termasuk PDF, dokumen Microsoft Word, spreadsheet, dan banyak lagi. Kerangka kerja ini mengubah data perusahaan menjadi basis pengetahuan yang dapat dikueri yang dioptimalkan untuk agen AI. Lihat informasi yang lebih lengkap dalam dokumentasi LlamaIndex.
- Production-ready deployment — LlamaIndex menawarkan kerangka kerja sumber terbuka dan layanan terkelola melaluiLlamaCloud, menyediakan fitur kelas perusahaan termasuk kontrol keamanan, skalabilitas, integrasi observabilitas, dan fleksibilitas penerapan. Untuk informasi selengkapnya, lihat dokumentasi LlamaIndex kerangka kerja.
- Pemrosesan dokumen lanjutan — LlamaCloud menyediakan kemampuan penguraian dokumen, ekstraksi, pengindeksan, dan pengambilan yang menangani tata letak kompleks, tabel bersarang,

konten multi-modal, dan bahkan catatan tulisan tangan. Penguraian canggih ini memungkinkan agen untuk bekerja secara efektif dengan dokumen perusahaan dunia nyata yang berisi bagan, diagram, dan pemformatan kompleks. Lihat informasi yang lebih lengkap dalam dokumentasi LlamaCloud.

- Orkestrasi alur kerja - LlamaAgents menyediakan mesin orkestrasi async-first yang digerakkan oleh peristiwa untuk membangun sistem agen multi-langkah. Alur kerja mendukung pola kompleks termasuk loop, eksekusi paralel, percabangan bersyarat, dan dimulainya kembali stateful, menjadikannya ideal untuk interaksi agen yang canggih. Untuk informasi selengkapnya, lihat dokumentasi LlamaIndex alur kerja.
- Kemampuan pengambilan agen — Mode pengambilan lanjutan termasuk pencarian hibrida, pencarian semantik, dan perutean otomatis yang secara cerdas menentukan strategi pengambilan terbaik untuk setiap kueri. Kerangka kerja ini mendukung pengambilan komposit di beberapa basis pengetahuan dengan reranking untuk meningkatkan akurasi. Untuk informasi selengkapnya, lihat dokumentasi LlamaIndex RAG.
- Observabilitas dan evaluasi — LlamaIndex terintegrasi dengan berbagai alat observabilitas dan evaluasi. Kemampuan integrasi ini membantu Anda melacak dan men-debug aplikasi Anda, mengevaluasi kinerjanya, dan memantau biaya. Untuk informasi selengkapnya, lihat dokumentasi Tracing dan Debugging dan LlamaIndex Evaluating.

Kapan harus menggunakan LlamaIndex

LlamaIndexsangat cocok untuk skenario AI agen yang menekankan alur kerja intensif data dan manajemen pengetahuan:

- Document-heavy aplikasi yang mengharuskan agen untuk memproses, menganalisis, dan mengekstrak wawasan dari sejumlah besar dokumen perusahaan seperti kontrak, laporan, manual, dan pengajuan peraturan
- Pembuatan prototipe cepat ke skenario produksi di mana organisasi ingin dengan cepat membangun dan menyebarkan agen yang berpusat pada dokumen tanpa overhead manajemen infrastruktur yang luas
- RAG-first arsitektur yang memprioritaskan akurasi pengambilan dan relevansi konteks, terutama ketika bekerja dengan dokumen multi-modal yang kompleks yang berisi tabel, gambar, dan data terstruktur
- Multi-agent alur kerja dokumen yang memerlukan agen khusus untuk berbagai aspek pemrosesan dokumen, seperti penguraian, analisis, ringkasan, dan pemeriksaan kepatuhan

Pendekatan implementasi untuk LlamaIndex

LlamaIndex menyediakan blok bangunan tingkat rendah dan abstraksi tingkat tinggi yang mengakomodasi pendekatan implementasi yang berbeda:

- Perkembangan pesat aplikasi RAG fungsional hanya dalam beberapa baris kode dengan menggunakan API LlamaIndex tingkat tinggi. Pendekatan ini dapat LlamaIndex diakses oleh tim bisnis dan pengembang yang baru mengenal AI agen.
- Integrasi perusahaan melalui LlamaHub sistem perusahaan populer termasuk SharePoint, Amazon Simple Storage Service (Amazon S3), database, dan API. Pendekatan ini memungkinkan integrasi tanpa batas dengan infrastruktur data yang ada.
- Opsi penyebaran yang fleksibel antara penerapan yang di-host mandiri sumber terbuka untuk kontrol maksimum, atau layanan LlamaCloud terkelola untuk mengurangi overhead operasional dan fitur perusahaan.
- Aplikasi dapat dimulai dengan mesin kueri sederhana dan secara progresif menambahkan kemampuan agen, orkestrasi multi-agen, dan alur kerja yang kompleks seiring dengan berkembangnya persyaratan.

Real-world contoh dari LlamaIndex

Contoh ini berfokus pada anak perusahaan dari perusahaan kedirgantaraan yang berspesialisasi dalam navigasi penerbangan dan solusi operasi. Mereka perlu mengatasi tantangan yang berkembang yang melibatkan uji coba chatbot AI yang tidak terkoordinasi. Uji coba menghasilkan pekerjaan berulang, siklus pengembangan yang panjang, hambatan kepatuhan, dan implementasi yang terisolasi di seluruh organisasi.

Mereka mengembangkan kerangka agen terpadu, solusi berbasis template yang dapat digunakan kembali yang dibangun di atas kerangka kerja LlamaIndex sumber terbuka yang membuat pembuatan agen jauh lebih efisien. Mereka membandingkan beberapa kerangka kerja yang bersaing, baik berorientasi rantai maupun berbasis grafik. Pada akhirnya, mereka LlamaIndex memilih tiga keunggulan penting: desainnya yang fleksibel, komponen modular, dan kontrol orkestrasi siap produksi.

Platform ini mengurangi waktu pengembangan dan penyebaran agen sebesar 87% dari 512 menjadi 64 jam. Pengurangan ini dicapai dengan memungkinkan tim untuk membangun agen dengan sekitar 50 baris kode dan file konfigurasi JSON. Tim memanfaatkan kerangka kerja terpadu dengan

keamanan bawaan, kepatuhan, dan akses sistem istimewa. Untuk detail lebih lanjut, lihat studi kasus LlamaIndex pelanggan.

Membandingkan kerangka kerja AI agen

Saat memilih kerangka kerja AI agen untuk pengembangan agen otonom, pertimbangkan bagaimana setiap opsi selaras dengan persyaratan spesifik Anda. Pertimbangkan tidak hanya kemampuan teknisnya tetapi juga kecocokan organisasinya, termasuk keahlian tim, infrastruktur yang ada, dan persyaratan pemeliharaan jangka panjang. Banyak organisasi mungkin mendapat manfaat dari pendekatan hibrida, memanfaatkan beberapa kerangka kerja untuk berbagai komponen ekosistem AI otonom mereka.

Tabel berikut membandingkan tingkat kematangan (terkuat, kuat, memadai, atau lemah) dari setiap kerangka kerja di seluruh dimensi teknis utama. Untuk setiap kerangka kerja, tabel juga mencakup informasi tentang opsi penerapan produksi dan kompleksitas kurva pembelajaran.

Kerangka	AWS integrasi	Dukungan multi-agen otonom	Kompleksitas alur kerja otonom	Kemampuan multimodal	Pemilihan model pondasi	Integrasi API LLM	Penyebaran produksi	Kurva belajar
AutoGen	Lemah	Kuat	Kuat	Memadai	Memadai	Kuat	Lakukan sendiri (DIY)	Curam
CrewAI	Lemah	Kuat	Memadai	Lemah	Memadai	Memadai	DIY	Sedang
LangChain / LangGraph	Memadai	Kuat	Terkuat	Terkuat	Terkuat	Terkuat	Platform atau DIY	Curam
LlamaIndex	Memadai	Memadai	Kuat	Memadai	Kuat	Kuat	Platform atau DIY	Sedang
Strands Agents	Terkuat	Kuat	Terkuat	Kuat	Kuat	Terkuat	DIY	Sedang

Pertimbangan dalam memilih kerangka kerja AI agen

Saat mengembangkan agen otonom, pertimbangkan faktor-faktor kunci berikut:

- AWS Integrasi infrastruktur — Organizations yang banyak diinvestasikan AWS akan mendapat manfaat besar dari integrasi asli Strands Agents dengan Layanan AWS untuk alur kerja otonom. Untuk informasi lebih lanjut, lihat Roundup AWS Mingguan (AWS Blog).
- Pemilihan model foundation - Pertimbangkan kerangka kerja mana yang memberikan dukungan terbaik untuk model fondasi pilihan Anda (misalnya, model Amazon Nova di Amazon Bedrock atau Anthropic Claude), berdasarkan persyaratan penalaran agen otonom Anda. Untuk informasi lebih lanjut, lihat Membangun Agen Efektif di Anthropic situs web.
- Integrasi API LLM - Evaluasi kerangka kerja berdasarkan integrasinya dengan antarmuka layanan model bahasa besar (LLM) pilihan Anda (misalnya, Amazon Bedrock atauOpenAI) untuk penerapan produksi. Untuk informasi selengkapnya, lihat Antarmuka Model dalam Strands Agents dokumentasi.
- Persyaratan multimodal — Untuk agen otonom yang perlu memproses teks, gambar, dan ucapan, pertimbangkan kemampuan multimodal dari setiap kerangka kerja. Untuk informasi selengkapnya, lihat Multimodalitas dalam dokumentasi. LangChain
- Kompleksitas alur kerja otonom - Alur kerja otonom yang lebih kompleks dengan manajemen negara yang canggih mungkin mendukung kemampuan mesin negara bagian yang canggih. LangGraph
- Kolaborasi tim otonom — Proyek yang membutuhkan kolaborasi otonom berbasis peran eksplisit antara agen khusus dapat memperoleh manfaat dari arsitektur berorientasi tim. CrewAI
- Paradigma pengembangan otonom — Tim yang lebih menyukai pola percakapan dan asinkron untuk agen otonom mungkin lebih menyukai arsitektur berbasis peristiwa. AutoGen
- Pendekatan terkelola atau berbasis kode - Organizations yang menginginkan pengalaman yang dikelola sepenuhnya dengan pengkodean minimal harus mempertimbangkan Amazon Bedrock Agents. Organizations yang memerlukan kustomisasi lebih dalam mungkin lebih suka Strands Agents atau kerangka kerja lain dengan kemampuan khusus yang lebih selaras dengan persyaratan agen otonom tertentu.
- Kesiapan produksi untuk sistem otonom — Pertimbangkan opsi penyebaran, kemampuan pemantauan, dan fitur perusahaan untuk agen otonom produksi.

Platform

Platform AI Agentic menyediakan lapisan runtime, orkestrasi, dan integrasi dasar yang diperlukan untuk menerapkan, menskalakan, dan mengelola sistem agen tingkat produksi. Kerangka kerja mendefinisikan bagaimana agen dibangun dan protokol mengatur bagaimana mereka berkomunikasi. Platform menyediakan lingkungan di mana agen-agen ini beroperasi, berkolaborasi, dan berkembang dengan aman dalam skala besar.

Platform agen menggabungkan eksekusi model, manajemen konteks, integrasi alat, observabilitas, dan kemampuan tata kelola ke dalam lingkungan terpadu. Platform ini memungkinkan organisasi untuk beralih dari eksperimen ke penyebaran skala perusahaan.

Di bagian ini:

Mengapa platform penting

Platform AI agen sangat penting bagi organisasi yang ingin mengoperasionalkan sistem otonom dalam produksi. Mereka menawarkan kemampuan berikut:

- Menyediakan orkestrasi runtime untuk hosting, penskalaan, dan agen koordinasi.
- Kelola status, konteks, dan memori di seluruh alur kerja multi-agen.
- Menawarkan kontrol keamanan, identitas, dan tata kelola yang selaras dengan standar perusahaan.
- Integrasikan dengan ekosistem perkakas dan sistem eksternal melalui standar APIs atau protokol.
- Aktifkan observabilitas dan auditabilitas di seluruh interaksi agen dan arus peristiwa.
- Mendukung interoperabilitas lintas model, memungkinkan agen untuk menggunakan beberapa model pondasi dalam satu lingkungan.

Kemampuan ini mengubah agen individu menjadi sistem adaptif yang terkoordinasi yang dapat beroperasi dengan andal dalam batas perusahaan dan peraturan.

Jenis platform AI agen

Platform AI agen biasanya termasuk dalam satu atau lebih kategori berikut:

- Agen terkelola - Platform yang dikelola sepenuhnya menyediakan infrastruktur, memori, dan kemampuan orkestrasi bawaan. Mereka mengurangi overhead operasional dan mempercepat waktu produksi.
- Orkestrasi sumber terbuka — Platform agen sumber terbuka menawarkan fleksibilitas dan transparansi bagi organisasi yang lebih menyukai lingkungan yang dapat disesuaikan atau penerapan lokal.
- Hybrid Enterprise — Platform hybrid mengintegrasikan komponen yang dikelola dan di-host sendiri, menggabungkan skalabilitas layanan yang dikelola cloud dengan kontrol sistem perusahaan.

Pertimbangan pemilihan platform

Saat memilih atau merancang platform AI agen, organisasi harus mempertimbangkan hal berikut:

- Integration depth — Mengevaluasi seberapa baik platform terintegrasi dengan sumber data, alat, dan protokol yang ada.
- Skalabilitas — Pastikan platform dapat menskalakan secara dinamis untuk mendukung beban kerja otonom dan kolaborasi multi-agen.
- Keamanan dan kepatuhan - Menilai privasi data, enkripsi, dan fitur tata kelola terhadap persyaratan organisasi dan regional.
- Ekstensibilitas — Pilih platform dengan arsitektur modular yang memungkinkan alat, model, atau agen baru ditambahkan dari waktu ke waktu.
- Observabilitas — Lebih suka platform yang menyediakan telemetri terperinci, keterlacakan, dan log audit untuk interaksi agen.
- Efisiensi biaya - Pertimbangkan model tanpa server atau berbasis penggunaan untuk mengoptimalkan biaya untuk beban kerja variabel.

Agen Batuan Dasar Amazon

Amazon Bedrock Agents adalah layanan terkelola penuh yang memungkinkan Anda membangun dan mengonfigurasi agen otonom dalam aplikasi Anda. Ini dapat mengatur interaksi antara model dasar, sumber data, aplikasi perangkat lunak, dan percakapan pengguna. Pendekatannya yang efisien untuk membuat agen tidak mengharuskan Anda menyediakan kapasitas, mengelola infrastruktur, atau menulis kode khusus.

Fitur utama dari Amazon Bedrock Agents

Amazon Bedrock Agents mencakup fitur-fitur utama berikut:

- Layanan yang dikelola sepenuhnya - Manajemen infrastruktur lengkap tanpa perlu menyediakan kapasitas atau mengelola sistem yang mendasarinya. Untuk informasi selengkapnya, lihat Mengotomatiskan tugas di aplikasi Anda menggunakan agen AI di dokumentasi Amazon Bedrock.
- Pengembangan berbasis API — Tentukan dan jalankan agen melalui panggilan API sederhana dengan menentukan model, instruksi, alat, dan parameter konfigurasi. Untuk informasi selengkapnya, lihat Membuat dan mengonfigurasi agen secara manual di dokumentasi Amazon Bedrock.
- Grup tindakan — Tentukan tindakan spesifik yang dapat dilakukan agen Anda dengan membuat grup tindakan dengan skema API. Untuk informasi selengkapnya, lihat Menggunakan grup tindakan untuk menentukan tindakan yang akan dilakukan agen Anda di dokumentasi Amazon Bedrock.
- Integrasi basis pengetahuan — Terhubung dengan mulus ke Pangkalan Pengetahuan Amazon Bedrock untuk meningkatkan respons agen dengan data organisasi Anda. Untuk informasi selengkapnya, lihat Pembuatan respons tambahan untuk agen Anda dengan basis pengetahuan di dokumentasi Amazon Bedrock.
- Template prompt lanjutan - Sesuaikan perilaku agen melalui templat prompt untuk pra-pemrosesan, orkestrasi, pembuatan respons basis pengetahuan, dan pasca-pemrosesan. Untuk informasi selengkapnya, lihat Meningkatkan akurasi agen menggunakan templat prompt lanjutan di Amazon Bedrock di dokumentasi Amazon Bedrock.
- Penelusuran dan pengamatan - Lacak proses step-by-step penalaran agen menggunakan kemampuan penelusuran bawaan. Untuk informasi selengkapnya, lihat Lacak proses step-by-step penalaran agen menggunakan jejak di dokumentasi Amazon Bedrock.

- Pembuatan versi dan alias — Buat beberapa versi agen Anda dan terapkan melalui alias untuk peluncuran terkontrol. Untuk informasi selengkapnya, lihat Menerapkan dan menggunakan agen Amazon Bedrock di aplikasi Anda di dokumentasi Amazon Bedrock.

Kapan menggunakan Amazon Bedrock Agents

Amazon Bedrock Agents sangat cocok untuk skenario agen otonom termasuk:

- Organizations yang menginginkan pengalaman yang dikelola sepenuhnya untuk membangun dan menyebarkan agen tanpa mengelola infrastruktur
- Proyek yang membutuhkan pengembangan dan penyebaran agen yang cepat melalui konfigurasi daripada kode
- Kasus penggunaan yang mendapat manfaat dari integrasi ketat dengan kemampuan Amazon Bedrock lainnya seperti Pangkalan Pengetahuan dan Pagar Pembatas
- Tim tanpa sumber daya internal untuk membangun agen dari awal tetapi membutuhkan kemampuan otonom siap produksi

Pendekatan implementasi untuk Amazon Bedrock Agents

Amazon Bedrock Agents menyediakan pendekatan implementasi berbasis konfigurasi untuk pemangku kepentingan bisnis. Layanan ini memungkinkan organisasi untuk:

- Tentukan agen melalui panggilan Konsol Manajemen AWS atau API tanpa menulis kode yang rumit.
- Buat grup tindakan yang menentukan APIs dan operasi yang dapat dilakukan agen.
- Hubungkan basis pengetahuan untuk memberikan informasi spesifik domain kepada agen.
- Uji dan ulangi perilaku agen melalui antarmuka visual.

Pendekatan terkelola ini memungkinkan tim bisnis untuk dengan cepat mengembangkan dan menyebarkan agen otonom tanpa keahlian teknis yang mendalam dalam pengembangan model AI atau manajemen infrastruktur.

Contoh dunia nyata dari Amazon Bedrock Agents

Solusi operasi keuangan (FinOps) yang dijelaskan dalam posting AWS blog ini menggunakan kerangka kerja multi-agen Amazon Bedrock untuk membuat asisten manajemen biaya cloud berbasis

AI. Model yayasan Amazon Nova yang hemat biaya mendukung solusi di mana agen FinOps Supervisor pusat mendelegasikan tugas kepada agen khusus. Agen ini mengambil dan menganalisis data AWS pengeluaran dengan menggunakan AWS Cost Explorer dan menghasilkan rekomendasi penghematan biaya dengan menggunakan. AWS Trusted Advisor

Sistem ini mencakup akses pengguna yang aman melalui Amazon Cognito, front-end yang dihosting, dan grup AWS Lambda tindakan untuk AWS Amplify analisis dan peramalan waktu nyata. Tim keuangan dapat mengajukan pertanyaan bahasa alami seperti “Berapa biaya saya pada Februari 2025?” Sistem merespons dengan perincian terperinci, saran pengoptimalan, dan prakiraan — semuanya dalam arsitektur tanpa server yang dapat diskalakan yang digunakan dengan menggunakan. AWS CloudFormation

Batuan Dasar Amazon AgentCore

Amazon Bedrock AgentCore adalah platform agen untuk membangun, menyebarkan, dan mengoperasikan agen berkemampuan tinggi dengan aman dalam skala menggunakan kerangka kerja, model, atau protokol apa pun. Menggunakan AgentCore, Anda dapat melakukan hal berikut, semua tanpa manajemen infrastruktur:

- Membangun agen lebih cepat.
- Memungkinkan agen untuk mengambil tindakan di seluruh alat dan data.
- Jalankan agen dengan aman dengan latensi rendah dan runtime yang diperpanjang.
- Memantau agen dalam produksi.

AgentCore menghilangkan pengangkatan berat yang tidak berdiferensiasi dari membangun infrastruktur agen khusus, memungkinkan Anda mempercepat agen Anda ke produksi. Layanannya dapat digunakan bersama atau secara independen dan kompatibel dengan kerangka kerja apa pun, termasukCrewAI,LangGraph,LlamaIndex, danStrands Agents. AgentCorejuga kompatibel dengan model foundation apa pun yang tersedia di dalam atau di luar Amazon Bedrock, memberikan fleksibilitas tertinggi.

AgentCore terdiri dari beberapa layanan utama:

- Amazon Bedrock AgentCore Runtime - Menyediakan lingkungan yang aman, tanpa server, dan dapat diskalakan untuk meng-host dan menjalankan agen Anda, tanpa perlu mengelola infrastruktur apa pun yang diperlukan untuk menerapkan dan menjalankan agen atau alat AI.

- Amazon Bedrock AgentCore Memory - Menawarkan sistem memori terkelola, memungkinkan agen untuk mempertahankan konteks dari interaksi untuk percakapan yang lebih personal dan koheren dengan mempertahankan pengetahuan langsung dan jangka panjang.
- Amazon Bedrock AgentCore Gateway — Menyederhanakan proses pembuatan, pengamanan, dan menemukan alat yang tepat untuk agen. Dengan AgentCore Gateway, pengembang dapat mengonversi APIs, fungsi Lambda, dan layanan yang ada menjadi alat yang kompatibel dengan Model Context Protocol (MCP) dan membuatnya tersedia untuk agen.
- Amazon Bedrock AgentCore Identity - Menyediakan identitas agen yang aman dan terukur serta layanan manajemen akses yang mempercepat pengembangan agen AI. Dengan AgentCore Identity, Anda dapat menetapkan identitas unik dan dapat diverifikasi ke agen, memungkinkan kontrol akses yang halus dan interaksi yang didukung agen yang aman dengan sistem perusahaan.
- Alat AgentCore bawaan Amazon Bedrock - Memungkinkan Anda menggunakan alat bawaan untuk meningkatkan alur kerja pengembangan dan pengujian Anda. Gunakan alat ini untuk berinteraksi dengan aplikasi Anda secara efektif, memungkinkan agen AI untuk menulis dan mengeksekusi kode dengan aman di lingkungan kotak pasir. Gunakan alat browser untuk memungkinkan agen AI berinteraksi dengan situs web dalam skala besar.
- Amazon Bedrock AgentCore Observability — Memberikan kemampuan pencatatan dan pemantauan, memberi Anda visibilitas real-time ke kinerja dan perilaku agen Anda untuk memfasilitasi debugging dan pengoptimalan.

Fitur utama AgentCore

AgentCore termasuk fitur utama berikut:

- Dikelola sepenuhnya dan dapat diperluas - AgentCore adalah layanan yang dikelola sepenuhnya, yang berarti AWS menangani infrastruktur dan pemeliharaan yang mendasarinya. Ini juga dapat diperluas, yang memungkinkan Anda untuk menyesuaikan dan meningkatkan fungsionalitas agen Anda. Untuk informasi selengkapnya, lihat Memulai AgentCore Runtime di AgentCore dokumentasi.
- Memori jangka panjang dan jangka pendek — Memberikan interaksi yang lebih personal dan relevan dengan melengkapi agen dengan sistem memori untuk mengingat konteks dari percakapan saat ini dan pengetahuan jangka panjang. Untuk informasi selengkapnya, lihat Memulai AgentCore Memori di AgentCore dokumentasi.

- Pengembangan dan integrasi alat yang disederhanakan - Memungkinkan agen Anda menemukan dan menggunakan alat melalui satu titik akhir yang aman. Ubah sumber daya perusahaan Anda dengan cepat menjadi alat yang siap agen hanya dengan beberapa baris kode, membebaskan pengembang untuk fokus membangun kemampuan unik. Untuk informasi selengkapnya, lihat Memulai AgentCore Gateway di AgentCore dokumentasi.
- Infrastruktur yang aman dan terukur - AgentCore menyediakan lingkungan yang aman dan terukur untuk menyebarkan dan mengoperasikan agen. Ini mencakup fitur untuk identitas dan manajemen akses, enkripsi data, dan keamanan jaringan. Untuk informasi selengkapnya, lihat Memulai AgentCore Identitas dalam AgentCore dokumentasi.
- Integrasi dengan berbagai alat - Memungkinkan Anda mengintegrasikan agen Anda dengan berbagai alat, termasuk penerjemah kode dan alat browser yang dapat Anda buat dengan menggunakan alat AgentCore bawaan. Untuk informasi selengkapnya, lihat Memulai dengan Penerjemah AgentCore Kode dan Memulai AgentCore Browser di AgentCore dokumentasi.
- Pengamatan dan pemantauan komprehensif — Dapatkan visibilitas mendalam ke agen Anda dengan alat komprehensif untuk melacak, men-debug, dan memantau kinerja mereka dalam produksi. Visualisasikan seluruh jalur eksekusi agen untuk mengaudit alasannya dan menyelesaikan kegagalan. Gunakan dasbor real-time dan data telemetri standar untuk melacak metrik operasional utama. Untuk informasi selengkapnya, lihat Menambahkan observabilitas ke AgentCore resource Amazon Bedrock Anda di dokumentasi. AgentCore

Kapan harus menggunakan AgentCore

AgentCore sangat cocok untuk skenario agen otonom termasuk:

- Organizations yang ingin mempercepat pengembangan dan menurunkan biaya operasional dengan layanan terkelola penuh yang menangani infrastruktur, keamanan, alat bawaan, observabilitas, dan penskalaan
- Proyek yang membutuhkan fleksibilitas dengan layanan modular yang bekerja sama atau independen dan kompatibel dengan kerangka kerja apa pun, seperti CrewAI atauLangGraph, dan model fondasi apa pun dari sumber mana pun
- Gunakan kasus yang membutuhkan agen percakapan stateful yang perlu mempertahankan konteks dan belajar dari interaksi masa lalu untuk memberikan tanggapan yang dipersonalisasi dan relevan
- Agen memungkinkan untuk melakukan tugas-tugas kompleks melalui integrasi sederhana dengan beragam aplikasi, sumber data, dan APIs

Pendekatan implementasi untuk AgentCore

AgentCore dirancang untuk organisasi yang ingin memindahkan agen AI dari bukti konsep, dibangun menggunakan kerangka kerja open source atau agen kustom, ke produksi. Dengan AgentCore, organisasi dapat melakukan hal berikut:

- Menyebarkan agen secara aman pada infrastruktur tanpa server, mendukung kerangka kerja dan model apa pun, dengan isolasi sesi dan identitas bawaan serta manajemen akses untuk end-to-end keamanan dan kepatuhan. Buat agen AgentCore Runtime dengan cepat untuk kerangka kerja agen terkemuka dengan menggunakan toolkit starter.
- Tingkatkan agen dengan mengintegrasikan memori persisten untuk retensi konteks, menyederhanakan pengembangan dan integrasi alat melalui AgentCore Gateway. Manfaatkan alat browser bawaan dan penerjemah kode untuk alur kerja tingkat lanjut.
- Lacak, debug, dan pantau agen AI dalam produksi menggunakan dasbor observabilitas yang didukung oleh Amazon CloudWatch Application Insights danOpenTelemetry, melacak metrik utama AgentCore sumber daya (runtime, memori, gateway, dan alat).
- Mempercepat penyebaran dan inovasi dengan layanan modular yang dikelola sepenuhnya, blok yang dapat dikomposisi bersama-sama atau secara independen, dengan kerangka kerja agen dan penyedia model apa pun. Fleksibilitas ini membantu organisasi beralih dari prototipe ke produksi lebih cepat.

Pendekatan terkelola ini memungkinkan organisasi untuk dengan cepat dan aman membangun, menyebarkan, dan menjalankan agen AI tingkat perusahaan dan sistem multi-agen pada skala apa pun.

Contoh dunia nyata AgentCore

AWS Telah mengamati bahwa salah satu bank terbesar di Amerika Latin telah digunakan AI/ML selama bertahun-tahun untuk memberikan pengalaman perbankan digital yang sangat personal dan aman. Bank memperluas layanan AI agen dengan menggunakan AgentCore untuk menyediakan pelanggan dengan interaksi intuitif, keamanan yang ditingkatkan, dan otomatisasi yang lebih besar. Menurut CTO, AgentCore diharapkan dapat mendukung upaya mereka untuk memenuhi komitmen pelanggan dalam skala besar. AgentCore memberikan pengembang mereka alat dan fleksibilitas untuk membangun dan mengelola agen, sambil membantu memastikan kepatuhan terhadap peraturan keuangan.

Protokol

Agen AI memerlukan protokol komunikasi standar untuk berinteraksi dengan agen dan layanan lain. Organizations menerapkan arsitektur agen menghadapi tantangan signifikan seputar interoperabilitas, independensi vendor, dan pemeriksaan masa depan investasi mereka.

Bagian ini membantu Anda menavigasi lanskap agent-to-agent protokol dengan fokus pada standar terbuka yang memaksimalkan fleksibilitas dan interoperabilitas. (Untuk informasi tentang agent-to-tool protokol, lihat Strategi integrasi alat nanti dalam panduan ini.)

Bagian ini menyoroti Model Context Protocol (MCP), standar terbuka yang awalnya dikembangkan pada Anthropic tahun 2024. Hari ini, AWS secara aktif mendukung MCP melalui kontribusi untuk pengembangan dan implementasi protokol. AWS Berkolaborasi dengan kerangka kerja agen open-source terkemuka, termasuk,, dan LangGraph CrewAILlamaIndex, untuk membentuk masa depan komunikasi antar-agen pada protokol. Untuk informasi selengkapnya, lihat Protokol Terbuka untuk Interoperabilitas Agen Bagian 1: Komunikasi Antar Agen di MCP (Blog).AWS

Di bagian ini:

- Mengapa pemilihan protokol penting
- Agent-to-agent protokol
- Memilih protokol agen
- Strategi implementasi untuk protokol agen
- Memulai dengan MCP
- ???

Mengapa pemilihan protokol penting

Pemilihan protokol secara fundamental membentuk bagaimana Anda dapat membangun dan mengembangkan arsitektur agen AI Anda. Dengan memilih protokol yang mendukung portabilitas antar kerangka kerja agen, Anda mendapatkan fleksibilitas untuk menggabungkan berbagai sistem agen dan alur kerja untuk memenuhi kebutuhan spesifik Anda.

Protokol terbuka memungkinkan Anda mengintegrasikan agen di beberapa kerangka kerja. Misalnya, gunakan LangChain untuk prototyping cepat dan implementasikan sistem produksi denganStrands Agents, berkomunikasi melalui protokol umum, seperti MCP atau protokol Agent2Agent (A2A). Fleksibilitas ini mengurangi ketergantungan pada penyedia AI tertentu, menyederhanakan integrasi

dengan sistem yang ada, dan memungkinkan Anda meningkatkan kemampuan agen dari waktu ke waktu.

Protokol yang dirancang dengan baik juga menetapkan pola keamanan yang konsisten untuk otentikasi dan otorisasi di seluruh ekosistem agen Anda. Yang terpenting, portabilitas protokol menjaga kebebasan Anda untuk mengadopsi kerangka kerja dan kemampuan agen baru saat muncul. Memilih protokol terbuka melindungi investasi Anda dalam pengembangan agen sambil mempertahankan interoperabilitas dengan sistem pihak ketiga.

Keuntungan dari protokol terbuka

Saat menerapkan ekstensi Anda sendiri atau membangun sistem agen kustom, protokol terbuka menawarkan keuntungan yang menarik:

- Dokumentasi dan transparansi — Biasanya menyediakan dokumentasi yang komprehensif dan implementasi transparan
- Dukungan komunitas — Akses ke komunitas pengembang yang lebih luas untuk pemecahan masalah dan praktik terbaik
- Jaminan interoperabilitas - Jaminan yang lebih baik bahwa ekstensi Anda akan bekerja di berbagai implementasi
- Kompatibilitas di masa mendatang - Mengurangi risiko melanggar perubahan atau penghentian
- Pengaruh pada pembangunan — Kesempatan untuk berkontribusi pada evolusi protokol

Agent-to-agent protokol

Tabel berikut memberikan ikhtisar protokol agen yang memungkinkan beberapa agen untuk berkolaborasi, mendelegasikan tugas, dan berbagi informasi.

Protokol	Ideal untuk	Pertimbangan-pertimbangan
Komunikasi antar agen MCP	Organizations mencari pola kolaborasi agen yang fleksibel	• Perpanjangan untuk Model Context Protocol (MCP) yang diusulkan oleh AWS yang dibangun di atas fondasi yang ada untuk komunikasi agent-to-agent

		• Memungkinkan kolaboras i agen yang mulus dengan keamanan OAuth berbasis
Protokol A2A	Ekosistem agen lintas platform	• Didukung oleh Google • Standar yang lebih baru dengan adopsi yang lebih terbatas dibandingkan dengan MCP

Memutuskan di antara opsi protokol

Saat menerapkan agent-to-agent komunikasi, sesuaikan persyaratan komunikasi spesifik Anda dengan kemampuan protokol yang sesuai. Pola interaksi yang berbeda memerlukan fitur protokol yang berbeda. Tabel berikut menguraikan pola komunikasi umum dan merekomendasikan pilihan protokol yang paling sesuai untuk setiap skenario.

Pola	Deskripsi	Pilihan protokol yang ideal
Permintaan dan tanggapan sederhana	Interaksi satu kali antar agen	MCP dengan aliran stateless
Dialog stateful	Percakapan yang sedang berlangsung dengan konteks	MCP dengan manajemen sesi
Kolaborasi multi-agen	Interaksi kompleks antara beberapa agen	MCP antar-agen atau AutoGen
Alur kerja berbasis tim	Tim agen hierarkis dengan peran yang ditentukan	MCP antar-agen,, atau CrewAI AutoGen

Di luar pola komunikasi, beberapa faktor teknis dan organisasi dapat memengaruhi pemilihan protokol Anda. Tabel berikut menguraikan pertimbangan utama yang dapat membantu Anda mengevaluasi protokol mana yang paling sesuai dengan persyaratan implementasi spesifik Anda.

Pertimbangan	Deskripsi	Contoh

Model keamanan	Persyaratan otentikasi dan otorisasi	OAuth 2.0 di MCP
Lingkungan penyebaran	Dimana agen akan menjalankan dan berkomunikasi	Mesin terdistribusi atau tunggal
Kompatibilitas ekosistem	Integrasi dengan kerangka kerja agen yang ada	LangChain atau Strands Agents
Kebutuhan skalabilitas	Pertumbuhan yang diharapkan dalam interaksi agen	Kemampuan streaming MCP

Memilih protokol agen

Untuk sebagian besar organisasi yang membangun sistem agen produksi, Model Context Protocol (MCP) menawarkan fondasi komunikasi yang paling komprehensif dan didukung dengan baik. agent-to-agent MCP mendapat manfaat dari kontribusi pengembangan aktif dari AWS dan komunitas open-source.

Memilih protokol agen yang tepat penting bagi organisasi yang ingin menerapkan AI agen secara efektif. Pertimbangan berbeda berdasarkan konteks organisasi.

Pertimbangan pemilihan protokol agen

Organizations harus mempertimbangkan praktik terbaik berikut saat memilih protokol untuk sistem AI agen:

- Prioritaskan standar terbuka — Organizations harus mengadopsi protokol terbuka seperti MCP untuk membantu memastikan interoperabilitas jangka panjang, ekstensibilitas, dan untuk mengurangi risiko penguncian vendor.
- Menyeimbangkan kecepatan dan fleksibilitas — Startup dan pengadopsi awal dapat dimulai dengan protokol kepemilikan yang didukung dengan baik untuk pengembangan yang cepat tetapi harus menentukan jalur migrasi ke standar terbuka saat sistem matang.
- Menerapkan lapisan abstraksi — Perusahaan harus menerapkan abstraksi protokol untuk menyederhanakan migrasi, mengaktifkan adopsi hibrida, dan strategi integrasi masa depan.

- Menekankan keamanan dan kepatuhan — Organizations dalam industri yang diatur harus memilih protokol dengan otentikasi, enkripsi, dan kemampuan audit yang kuat untuk memenuhi persyaratan tata kelola dan kepatuhan.
- Evaluasi kematangan ekosistem — Semua organisasi harus menilai kesehatan, adopsi, dan dukungan masyarakat dari setiap protokol untuk memastikan keberlanjutan dan meminimalkan utang teknis.
- Terlibat dalam pengembangan standar - Organizations harus berpartisipasi dalam badan standar atau komunitas sumber terbuka untuk membantu membentuk evolusi protokol dan memengaruhi praktik terbaik.
- Perhitungan kedaulatan data — Pemerintah dan sektor yang diatur harus memastikan pilihan protokol selaras dengan persyaratan residensi dan kedaulatan data di seluruh wilayah penyebaran.
- Manfaatkan layanan terkelola — Jika memungkinkan, gunakan implementasi protokol agen yang dikelola atau tanpa server untuk mengurangi kompleksitas operasional dan mempercepat penyebaran.

Strategi implementasi untuk protokol agen

Untuk menerapkan protokol agen secara efektif di seluruh organisasi Anda, pertimbangkan langkah-langkah strategis berikut:

1. Mulai dengan penyelarasan standar - Mengadopsi protokol terbuka yang sudah mapan jika memungkinkan.
2. Buat lapisan abstraksi — Menerapkan adaptor antara sistem Anda dan protokol tertentu.
3. Berkontribusi pada standar terbuka - Berpartisipasi dalam komunitas pengembangan protokol.
4. Pantau evolusi protokol —Tetap terinformasi tentang standar dan pembaruan yang muncul.
5. Uji interoperabilitas secara teratur — Verifikasi bahwa implementasi Anda tetap kompatibel.

Memulai dengan MCP

AWS secara aktif mendukung Model Context Protocol (MCP) melalui kontribusi untuk pengembangan dan implementasi protokol. AWS Berkolaborasi dengan kerangka kerja agen open-source terkemuka, termasuk,, dan LangGraph CrewAILlamaIndex, untuk membentuk masa depan komunikasi antar-agen pada protokol.

Untuk mengimplementasikan MCP dalam arsitektur agen Anda, lakukan tindakan berikut:

1. Jelajahi implementasi MCP dalam kerangka kerja seperti SDK. Strands Agents
2. Tinjau dokumentasi teknis Protokol Konteks Model.
3. Baca Protokol Terbuka untuk Interoperabilitas Agen Bagian 1: Komunikasi Antar Agen di MCP (AWS Blog) untuk mempelajari tentang interoperabilitas agen.
4. Bergabunglah dengan komunitas MCP untuk memengaruhi evolusi protokol.

MCP menyediakan lapisan komunikasi yang memungkinkan agen berinteraksi dengan data dan layanan eksternal dan juga dapat digunakan untuk memungkinkan agen berinteraksi dengan agen lain. Implementasi transportasi HTTP Streamable protokol memberi pengembang serangkaian pola interaksi yang komprehensif tanpa harus menemukan kembali roda. Pola-pola ini mendukung request/response arus stateless dan manajemen sesi stateful dengan persisten. IDs

Dengan mengadopsi protokol terbuka seperti MCP, Anda memposisikan organisasi Anda untuk membangun sistem agen yang tetap fleksibel, interoperable, dan mudah beradaptasi seiring berkembangnya teknologi AI. Untuk informasi tentang implementasi agent-to-tool protokol, lihat Strategi integrasi alat nanti dalam panduan ini.

Memulai dengan A2A

Protokol Agent2Agent (A2A) memungkinkan kolaborasi terdesentralisasi antar agen melalui lapisan semantik bersama. Alih-alih merutekan semua pekerjaan melalui orkestrator pusat, A2A memungkinkan agen untuk menemukan satu sama lain, mengiklankan kemampuan mereka, menegosiasikan tugas, dan berbagi konteks menggunakan protokol berbasis JSON yang ringan. Setiap agen menerbitkan manifes kemampuan.

Contoh berikut menunjukkan manifes kemampuan A2A yang disederhanakan yang mengiklankan tindakan yang didukung agen, input yang diperlukan, dan metadata operasional untuk memungkinkan penemuan dan negosiasi tugas:

```
{
  "can": ["summarize.text", "extract.keywords"],
  "needs": ["document.input"],
  "meta": { "version": "1.0.3", "latencyMs": 120 }
}
```

Model ini memungkinkan pencocokan kemampuan dinamis, delegasi tugas tengah, dan kolaborasi lintas organisasi. Agen dapat mengatur diri sendiri di sekitar tugas, membentuk kelompok kerja sementara, dan beradaptasi ketika kemampuan baru masuk atau keluar dari sistem.

A2A mendukung interaksi mulai dari permintaan stateless sederhana hingga sesi negosiasi multi-langkah, termasuk:

- peer-to-peerPesan langsung untuk kolaborasi latensi rendah
- Negosiasi tugas semantik, di mana agen memilih rekan yang paling cocok
- Penemuan berbasis kemampuan, memungkinkan pembagian kerja yang muncul
- Penahan sesi untuk interaksi multi-langkah stateful

Dengan mengadopsi protokol agen-asli terbuka seperti A2A, organisasi menciptakan sistem AI yang modular, interoperable, dan mampu berkolaborasi lintas batas. A2A memastikan bahwa ekosistem agen tetap fleksibel dan dapat berkembang saat agen, tim, atau sistem eksternal baru diperkenalkan, tanpa memerlukan lapisan orkestrasi yang kaku atau kopling sebelumnya.

Untuk mengimplementasikan protokol A2A dalam arsitektur agen Anda, lakukan tindakan berikut:

1. Tinjau Spesifikasi Protokol A2A — Baca versi terbaru Spesifikasi Protokol Agent2Agent (A2A) untuk mempelajari bagaimana kemampuan memanifestasikan, alur negosiasi, dan jabat tangan agen beroperasi.
2. Jelajahi runtime yang kompatibel dengan A2A — Evaluasi kerangka kerja seperti Strands Agents SDK atau lapisan runtime khusus yang mendukung manifes dan negosiasi kemampuan gaya A2A. peer-to-peer
3. Terapkan manifes kemampuan untuk agen Anda — Tentukan setiap agen`can`,`needs`, dan `meta` bidang untuk memungkinkan penemuan, perjodohan, dan kolaborasi tingkat niat.
4. Bereksperimenlah dengan pola negosiasi A2A — Gunakan loop permintaan—tawaran-terima, kueri kemampuan terstruktur, atau penemuan berbasis gosip untuk memahami bagaimana agen bernalar tentang siapa yang harus menangani tugas.
5. Uji A2A dalam lingkungan infrastruktur campuran — Gabungkan negosiasi sejawat A2A dengan perutean acara yang asli melalui AWS Amazon untuk mengevaluasi pola koordinasi hibrida. EventBridge
6. Bergabunglah dengan komunitas A2A — Terlibat dengan kelompok kerja terbuka untuk tetap up to date dengan ekstensi, rekomendasi keamanan, dan peningkatan interoperabilitas lintas vendor, dan berkontribusi pada pengembangan protokol.

Alat

Agen AI memberikan nilai dengan berinteraksi dengan alat eksternal APIs, dan sumber data untuk melakukan tugas yang bermanfaat. Strategi integrasi alat yang tepat secara langsung memengaruhi kemampuan agen Anda, postur keamanan, dan fleksibilitas jangka panjang.

Bagian ini membantu Anda menavigasi lanskap integrasi alat dengan fokus pada standar terbuka yang memaksimalkan kebebasan dan fleksibilitas Anda. Bagian ini menyoroti Model Context Protocol (MCP) untuk integrasi alat dan meninjau alat khusus kerangka kerja dan meta-tool khusus yang meningkatkan alur kerja agen.

Di bagian ini:

- Kategori alat
- Alat berbasis protokol
- Alat asli kerangka kerja
- Alat meta
- Strategi integrasi alat
- Praktik terbaik keamanan untuk integrasi alat

Kategori alat

Sistem agen bangunan melibatkan tiga kategori utama alat.

Alat berbasis protokol

Alat berbasis protokol menggunakan protokol standar untuk komunikasi: agent-to-tool

- Alat MCP — Buka alat standar yang bekerja di seluruh kerangka kerja dengan opsi eksekusi lokal dan jarak jauh.
- OpenAIpemanggilan fungsi — Alat berpemilik yang khusus untuk OpenAI model.
- Anthropicalat — Variasi pada OpenAI fungsi yang memanggil alat berpemilik yang khusus untuk model Anthropic Claude.

Alat asli kerangka kerja

Alat Framework-native dibangun langsung ke dalam kerangka kerja agen tertentu:

- Strands Agents alat - Ringan, quick-to-implement alat yang khusus untuk Strands Agents kerangka kerja.
- LangChainalat Python berbasis alat yang terintegrasi erat dengan LangChain ekosistem.
- LlamaIndexalat — Alat yang dioptimalkan untuk pengambilan dan pemrosesan data di dalamnyaLlamaIndex.

Alat meta

Meta-tools meningkatkan alur kerja agen tanpa langsung mengambil tindakan eksternal:

- Alat alur kerja - Kelola alur eksekusi agen, logika percabangan, dan manajemen status.
- Alat grafik agen - Mengkoordinasikan beberapa agen dalam alur kerja yang kompleks.
- Alat memori — Menyediakan penyimpanan persisten dan pengambilan informasi di seluruh sesi agen.
- Alat refleksi — Memungkinkan agen untuk menganalisis dan meningkatkan kinerja mereka sendiri.

Alat berbasis protokol

Saat mempertimbangkan alat berbasis protokol, Model Context Protocol (MCP) memberikan fondasi yang paling komprehensif dan fleksibel untuk integrasi alat. Sebagaimana dinyatakan dalam posting blog AWS Open Source tentang interoperabilitas agen, AWS telah merangkul MCP sebagai protokol strategis, secara aktif berkontribusi pada pengembangannya.

Tabel berikut menjelaskan opsi untuk penerapan alat MCP.

Model penyebaran	Deskripsi	Ideal untuk	Implementasi
Berbasis stdio lokal	Alat berjalan dalam proses yang sama dengan agen	Pengembangan, pengujian, dan alat sederhana	Cepat diimpleme ntasikan tanpa overhead jaringan

Acara terkirim server lokal (SSE) berbasis	Alat berjalan secara lokal tetapi berkomunikasi melalui HTTP	Alat lokal yang lebih kompleks dengan pemisahan kekhawatiran	Isolasi yang lebih baik tetapi latensi masih rendah
Remote HTTP Streamable	Alat berjalan di server jarak jauh	Lingkungan produksi dan alat bersama	Dapat diskalakan dan dikelola secara terpusat

MCP resmi SDKs tersedia untuk membangun alat MCP:

- PythonSDK — Implementasi komprehensif dengan dukungan protokol penuh
- TypeScriptSDK —JavaScript/TypeScriptimplementasi untuk aplikasi web
- JavaSDK - Implementasi Java untuk aplikasi perusahaan

Ini SDKs menyediakan blok bangunan untuk membuat alat yang kompatibel dengan MCP dalam bahasa pilihan Anda, dengan implementasi spesifikasi protokol yang konsisten.

Selain itu, AWS telah menerapkan MCP di Strands AgentsSDK. Strands AgentsSDK menyediakan cara mudah untuk membuat dan menggunakan alat yang kompatibel dengan MCP. Dokumentasi komprehensif tersedia di Strands Agents GitHub repositori. Untuk kasus penggunaan yang lebih sederhana atau ketika bekerja di luar Strands Agents kerangka kerja, MCP resmi SDKs menawarkan implementasi langsung protokol dalam berbagai bahasa.

Fitur keamanan alat MCP

Fitur keamanan alat MCP meliputi:

- OAuth 2.0/2.1 otentikasi — otentikasi standar industri
- Pelingkupan izin - Kontrol akses berbutir halus untuk alat
- Penemuan kemampuan alat — Penemuan dinamis alat yang tersedia
- Penanganan kesalahan terstruktur - Pola kesalahan yang konsisten

Memulai dengan alat MCP

Untuk menerapkan MCP untuk integrasi alat, lakukan tindakan berikut:

1. Jelajahi Strands AgentsSDK untuk implementasi MCP siap produksi.
2. Tinjau dokumentasi teknis MCP untuk memahami konsep inti.
3. Gunakan contoh praktis yang dijelaskan dalam posting Blog AWS Open Source ini.
4. Mulailah dengan alat lokal sederhana sebelum melanjutkan ke alat jarak jauh.
5. Bergabunglah dengan komunitas MCP untuk memengaruhi evolusi protokol.

Jelajahi AgentCore Gateway

Amazon Bedrock AgentCore Gateway menyediakan cara yang mudah dan aman bagi pengembang untuk membangun, menerapkan, menemukan, dan terhubung ke alat MCP, dan titik akhir target lainnya dalam skala besar. Dengan AgentCore Gateway, pengembang dapat mengonversi APIs, AWS Lambda fungsi, dan layanan yang ada menjadi alat yang kompatibel dengan MCP. Kemudian, hanya dengan beberapa baris kode, mereka dapat membuat alat ini tersedia untuk agen melalui titik akhir AgentCore Gateway. AgentCore Gateway mendukungOpenAPI,Smithy, dan Lambda sebagai tipe input, dan merupakan satu-satunya solusi yang menyediakan otentikasi ingress komprehensif dan otentikasi keluar dalam layanan yang dikelola sepenuhnya.

Alat asli kerangka kerja

Meskipun Model Context Protocol (MCP) memberikan fondasi yang paling fleksibel, alat kerangka kerja asli menawarkan keuntungan untuk kasus penggunaan tertentu.

Strands AgentsSDK menawarkan alat Python berbasis yang dicirikan oleh desainnya yang ringan yang membutuhkan overhead minimal untuk operasi sederhana. Mereka memungkinkan implementasi cepat dan memungkinkan pengembang untuk membuat alat hanya dengan beberapa baris kode. Selain itu, mereka terintegrasi erat untuk bekerja dengan mulus dalam Strands Agents kerangka kerja.

Contoh berikut menunjukkan cara membuat alat cuaca sederhana menggunakanStrands Agents. Pengembang dapat dengan cepat mengubah Python fungsi menjadi alat yang dapat diakses agen dengan overhead kode minimal dan secara otomatis menghasilkan dokumentasi yang sesuai dari docstring fungsi.

```
#Example of a simple Strands native tool

@tool
```

```
def weather(location: str) -> str:

"""Get the current weather for a location""" #

Implementation here

return f"The weather in {location} is sunny."
```

Untuk pembuatan prototipe cepat atau kasus penggunaan sederhana, alat kerangka kerja asli dapat mempercepat pengembangan. Namun, untuk sistem produksi, alat MCP memberikan interoperabilitas yang lebih baik dan fleksibilitas future daripada framework-native tools.

Tabel berikut memberikan ikhtisar alat khusus kerangka kerja lainnya.

Kerangka	Jenis alat	Keuntungan	Pertimbangan-pertimbangan
AutoGen	Definisi fungsi	Dukungan multi-agen yang kuat	Microsoftekosistem
LangChain	Pythonkelas	Ekosistem besar alat pra-bangun	Kerangka kerja terkunci
LlamaIndex	Fungsi Python	Dioptimalkan untuk operasi data	Terbatas untuk LlamaIndex

Alat meta

Meta-tools tidak berinteraksi langsung dengan sistem eksternal. Sebaliknya, mereka meningkatkan kemampuan agen dengan menerapkan pola agen. Bagian ini membahas alur kerja, grafik agen, dan meta-tools memori.

Meta-tools alur kerja

Alur kerja meta-tools mengelola alur eksekusi agen:

- Manajemen negara - Pertahankan konteks di berbagai interaksi agen
- Logika percabangan - Aktifkan jalur eksekusi bersyarat
- Mekanisme coba lagi — Tangani kegagalan dengan strategi coba ulang yang canggih

Contoh kerangka kerja dengan meta-tools alur kerja termasuk LangGraphdan kemampuan alur kerja. Strands Agents

Meta-tools grafik agen

Meta-tools grafik agen mengoordinasikan beberapa agen yang bekerja bersama:

- Delegasi tugas - Menetapkan subtugas ke agen khusus
- Agregasi hasil - Menggabungkan output dari beberapa agen
- Resolusi konflik — Menyelesaikan ketidaksepakatan antar agen

Kerangka kerja menyukai AutoGendan CrewAImengkhususkan diri dalam koordinasi grafik agen.

Alat meta memori

Meta-tools memori menyediakan penyimpanan dan pengambilan persisten:

- Riwayat percakapan - Pertahankan konteks di seluruh sesi
- Basis pengetahuan — Menyimpan dan mengambil informasi khusus domain
- Toko vektor - Aktifkan kemampuan pencarian semantik

Sistem sumber daya MCP menyediakan cara standar untuk mengimplementasikan meta-alat memori yang bekerja di berbagai kerangka kerja agen.

Strategi integrasi alat

Pilihan strategi integrasi alat Anda secara langsung memengaruhi apa yang dapat dicapai agen Anda dan seberapa mudah sistem Anda dapat berkembang. Prioritaskan protokol terbuka seperti Model Context Protocol (MCP) sementara secara strategis menggunakan framework-native tools dan meta-tools. Dengan begitu, Anda dapat membangun ekosistem alat yang tetap fleksibel dan kuat seiring kemajuan teknologi AI.

Pendekatan strategis berikut untuk integrasi alat memaksimalkan fleksibilitas sambil memenuhi kebutuhan mendesak organisasi Anda:

1. Mengadopsi MCP sebagai fondasi Anda - MCP menyediakan cara standar untuk menghubungkan agen ke alat dengan fitur keamanan yang kuat. Mulailah dengan MCP sebagai protokol alat utama Anda untuk:

- Alat strategis yang akan digunakan di beberapa implementasi agen.
- Alat sensitif keamanan yang memerlukan otentikasi dan otorisasi yang kuat.
- Alat yang membutuhkan eksekusi jarak jauh di lingkungan produksi.

2. Gunakan alat framework-native bila perlu — Pertimbangkan alat framework-native untuk:
   - Prototyping cepat selama pengembangan awal.
   - Alat non-kritis sederhana dengan persyaratan keamanan minimal.
   - Fungsionalitas khusus kerangka kerja yang memanfaatkan kemampuan unik.
3. Menerapkan meta-tools untuk alur kerja yang kompleks — Tambahkan meta-tools untuk menyempurnakan arsitektur agen Anda:
   - Mulai sederhana dengan pola alur kerja dasar.
   - Tambahkan kompleksitas saat kasus penggunaan Anda matang.
   - Standarisasi antarmuka antara agen dan meta-tools.
4. Rencanakan evolusi — Bangun dengan mempertimbangkan fleksibilitas masa depan:
   - Antarmuka alat dokumen secara independen dari implementasi.
   - Buat lapisan abstraksi antara agen dan alat.
   - Menetapkan jalur migrasi dari protokol eksklusif ke protokol terbuka.

Praktik terbaik keamanan untuk integrasi alat

Integrasi alat secara langsung memengaruhi postur keamanan Anda. Bagian ini menguraikan praktik terbaik untuk dipertimbangkan untuk organisasi Anda.

Autentikasi dan otorisasi

Manfaatkan kontrol akses yang kuat berikut:

- Gunakan OAuth 2.0/2.1 — Menerapkan otentikasi standar industri untuk alat jarak jauh.
- Terapkan hak istimewa paling sedikit — Berikan alat hanya izin yang mereka butuhkan.
- Putar kredensyal — Perbarui kunci API dan token akses secara teratur.

Perlindungan data

Untuk membantu melindungi data, lakukan langkah-langkah berikut:

- Validasi input dan output - Menerapkan validasi skema untuk semua interaksi alat.
- Enkripsi data sensitif — Gunakan TLS untuk semua komunikasi alat jarak jauh.
- Menerapkan minimisasi data — Hanya berikan informasi yang diperlukan ke alat.

Pemantauan dan audit

Pertahankan visibilitas dan kontrol dengan menggunakan mekanisme ini:

- Catat semua pemanggilan alat — Pertahankan jejak audit yang komprehensif.
- Memantau anomali — Mendeteksi pola penggunaan alat yang tidak biasa.
- Menerapkan pembatasan tarif - Mencegah penyalahgunaan melalui panggilan alat yang berlebihan.

Model keamanan Model Context Protocol (MCP) mengatasi masalah ini secara komprehensif. Untuk informasi selengkapnya, lihat Pertimbangan keamanan dalam dokumentasi MCP.

Kesimpulan

Lanskap AI agen terus berkembang pesat, menawarkan kepada organisasi cara baru yang kuat untuk membangun sistem yang cerdas dan otonom. Panduan ini telah mengeksplorasi tiga komponen penting untuk implementasi yang sukses: kerangka kerja yang menyediakan fondasi, platform yang menyediakan lingkungan, protokol yang memungkinkan komunikasi, dan alat yang memperluas kemampuan.

Saat kerangka kerja matang, Anda dapat mengharapkan peningkatan interoperabilitas, standardisasi di sekitar protokol seperti Model Context Protocol (MCP), dan kemampuan orkestrasi yang lebih canggih untuk agen otonom. Organizations yang membangun keahlian dengan kerangka kerja ini hari ini akan memiliki posisi yang baik untuk membangun agen yang semakin otonom dan cerdas yang memberikan nilai bisnis yang signifikan.

Platform menyediakan lingkungan eksekusi, tata kelola, dan siklus hidup tempat sistem agen beroperasi. Mereka menangani masalah seperti identitas, batas keamanan, observabilitas, manajemen memori, landasan sesi, dan interaksi yang aman dengan alat dan data. Di AWS lingkungan, platform seperti runtime agen terkelola dan layanan orkestrasi memungkinkan organisasi untuk menyebarkan, memantau, mengembangkan, dan mengatur agen otonom dan sistem agen dalam skala besar. Platform menjembatani kerangka kerja dasar dengan persyaratan operasional dunia nyata.

Pilihan protokol agen merupakan keputusan strategis yang menyeimbangkan kebutuhan pembangunan segera dengan fleksibilitas jangka panjang dan interoperabilitas. Dengan memprioritaskan protokol terbuka dan menciptakan lapisan abstraksi yang sesuai, organisasi dapat membangun sistem agen yang tetap dapat beradaptasi dengan teknologi yang berkembang sambil memenuhi persyaratan bisnis saat ini.

Bagi sebagian besar organisasi, MCP merupakan fondasi yang kuat karena standarnya yang terbuka, ekosistem yang berkembang, dukungan untuk pola agent-to-agent komunikasi, dan kemampuan integrasi alat. AWS telah merangkul MCP dan Agent2Agent (A2A) sebagai protokol strategis, secara aktif berkontribusi pada pengembangan mereka dan menerapkannya di seluruh layanan seperti SDK. Strands Agents Dengan menggunakan MCP atau A2A bersama dengan alat dan meta-tool framework-native yang sesuai, Anda dapat membangun sistem agen yang memberikan nilai langsung sambil tetap dapat beradaptasi dengan inovasi masa depan.

Sumber daya

Gunakan sumber daya berikut AWS dan lainnya yang terkait dengan pengembangan agen otonom.

AWS Blog

- Amazon Bedrock AgentCore Memory: Membangun agen sadar konteks
- Praktik terbaik untuk membangun aplikasi AI generatif yang kuat dengan Amazon Bedrock Agents - Bagian 1
- Praktik terbaik untuk membangun aplikasi AI generatif yang kuat dengan Amazon Bedrock Agents - Bagian 2
- Bangun jaringan pipa RAG yang kuat dengan dan LlamaIndex Amazon Bedrock
- Bangun agen AI tepercaya dengan Amazon Bedrock Observability AgentCore
- Evaluasi tanggapan RAG dengan Amazon Bedrock, LlamaIndex dan RAGAS
- Memperkenalkan Penerjemah AgentCore Kode Batuan Dasar Amazon
- Memperkenalkan Amazon Bedrock AgentCore Gateway: Mengubah pengembangan alat agen AI perusahaan
- Memperkenalkan Amazon Bedrock AgentCore Identity: Mengamankan AI agen dalam skala besar
- MemperkenalkanStrands Agents, SDK Agen AI Sumber Terbuka
- Protokol Terbuka untuk Interoperabilitas Agen Bagian 1: Komunikasi Antar Agen di MCP
- Meluncurkan dan menskalakan agen dan alat Anda dengan aman di Amazon Bedrock AgentCore Runtime
- AWS Transform untuk .NET, layanan AI agen pertama untuk memodernisasi aplikasi.NET dalam skala besar
- AWS Roundup Mingguan: Strands Agents

AWS Bimbingan Preskriptif

- Mengoperasionalkan AI agen pada AWS
- Yayasan AI agen pada AWS
- Pola dan alur kerja Agentic AI di AWS
- Membangun arsitektur tanpa server untuk AI agen AWS

- Membangun arsitektur multi-tenant untuk AI agen di AWS
- Keamanan untuk AI agen aktif AWS
- Pengambilan opsi dan arsitektur Augmented Generation di AWS

AWS sumber daya

- Dokumentasi Amazon Bedrock
- Dokumentasi Amazon Bedrock AgentCore
- Perangkat AgentCore Pemula Amazon Bedrock (repositori) GitHub
- Dokumentasi Amazon Nova
- AWS Server MCP (GitHubrepositori)

Sumber daya lainnya

- AutoGendokumentasi (Microsoft)
- Membangun agen yang efektif (Anthropic)
- CrewAI GitHubrepositori
- Dokumentasi LangChain
- LangGraphplatform
- Dokumentasi LlamaIndex
- Dokumentasi Protokol Konteks Model
- Dokumentasi Strands Agents
- Strands AgentsIkhtisar Alat
- Strands AgentsPanduan Memulai Cepat

Riwayat dokumen

Tabel berikut menjelaskan perubahan signifikan pada panduan ini. Jika Anda ingin diberi tahu tentang pembaruan masa depan, Anda dapat berlangganan umpan RSS.

Perubahan	Deskripsi	Tanggal
Bagian baru	Bagian Platform Ditambahkan	Januari 16, 2026
Publikasi awal	—	Juli 14, 2025

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.