

Leitfaden

AWS Amplify Hosten



AWS Amplify Hosten: Leitfaden

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Die Marken und Handelsmarken von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, die geeignet ist, Kunden irrezuführen oder Amazon in irgendeiner Weise herabzusetzen oder zu diskreditieren. Alle anderen Handelsmarken, die nicht Eigentum von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise zu Amazon gehören oder nicht, mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

Was ist AWS Amplify Hosting?	1
Unterstützte Frameworks	1
Amplify Hosting-Funktionen	2
Erste Schritte mit Amplify Hosting	2
Erstellen Sie ein Backend	3
Amplify Hosting-Preise	3
Erste Schritte	4
Voraussetzungen	4
Schritt 1: Ein Repository Connect	4
Schritt 2: Bestätigen Sie die Build-Einstellungen	5
Schritt 3: Stellen Sie die Anwendung bereit	6
Schritt 4: (Optional) Ressourcen bereinigen	7
Füge Funktionen zu deiner App hinzu	7
Serverseitiges Rendern (SSR)	9
Was ist serverseitiges Rendern	9
SSR-Framework-Unterstützung	10
Stellen Sie eine SSR-App für Amplify bereit	11
Verwenden Sie einen Framework-Adapter	12
Verwendung der Bereitstellungsspezifikation	13
Bereitstellungsspezifikation	14
Bereitstellen eines Express-Servers	38
Bildoptimierung für SSR-Apps	44
Verwenden eines benutzerdefinierten Image-Loaders	45
Integration der Bildoptimierung für Framework-Autoren	45
Grundlegendes zur Bildoptimierungs-API	46
Unterstützung der Version Node.js für Apps der Version Next.js	54
Fehlerbehebung bei SSR-Bereitstellungen	54
Sie verwenden einen Framework-Adapter	54
Edge-API-Routen führen dazu, dass Ihr Build Next.js fehlschlägt	54
Die inkrementelle statische Regeneration auf Abruf funktioniert für Ihre App nicht	55
Die Build-Ausgabe Ihrer App überschreitet die maximal zulässige Größe	55
Ihr Build schlägt mit einem Fehler fehl, dass nicht genügend Arbeitsspeicher zur Verfügung steht	57
Die Größe der HTTP-Antwort ist zu groß	58

Amplify Sie die Unterstützung für Next.js	58
Unterstützung für die Funktion Next.js	58
Preise für Next.js Apps	60
Bereitstellung einer Next.js App mit Amplify	60
Migrieren einer Next.js 11-App auf Amplify Hosting Compute	63
SSR-Funktionalität zu einer statischen Next.js App hinzufügen	65
Umgebungsvariablen für serverseitige Laufzeiten zugänglich machen	67
Bereitstellen einer Next.js -App in einem Monorepo	69
Amazon CloudWatch Logs für SSR-Apps	70
Amplify Next.js 11-Unterstützung	70
Einrichtung benutzerdefinierter Domains	80
Verstehen der DNS-Terminologie und -Konzepte	81
DNS-Terminologie	81
DNS-Überprüfung	82
Amplify Hosting-Aktivierungsprozess für benutzerdefinierte Domains	82
Verwenden von SSL/TLS-Zertifikaten	83
Fügen Sie eine benutzerdefinierte Domain hinzu, die von Amazon Route 53 verwaltet wird	84
Fügen Sie eine benutzerdefinierte Domain hinzu, die von einem DNS-Drittanbieter verwaltet wird	85
Aktualisieren Sie die DNS-Einträge für eine Domain, die verwaltet wird von GoDaddy	91
Aktualisieren Sie die DNS-Einträge für eine von Google Domains verwaltete Domain	94
Aktualisieren Sie das SSL/TLS-Zertifikat für eine Domain	98
Subdomains verwalten	99
Nur um eine Subdomain hinzuzufügen	99
Um eine mehrstufige Subdomain hinzuzufügen	99
Um eine Subdomain hinzuzufügen oder zu bearbeiten	100
Wildcard-Unterdomänen	100
Um eine Wildcard-Subdomain hinzuzufügen oder zu löschen	101
Automatische Subdomains für eine benutzerdefinierte Amazon Route 53-Domain einrichten ...	101
Webvorschauen mit Subdomains	102
Problembehandlung bei benutzerdefinierten Domains	102
Wie kann ich überprüfen, ob mein CNAME aufgelöst wird?	103
Meine bei einem Drittanbieter gehostete Domain befindet sich im Status „Ausstehende Überprüfung“	104
Meine mit Amazon Route 53 gehostete Domain befindet sich im Status „Ausstehende Überprüfung“	104

Ich erhalte einen CNAME-Fehler AlreadyExistsException	106
Ich erhalte die Fehlermeldung „Zusätzliche Überprüfung erforderlich“	106
Ich erhalte eine 404-Fehlermeldung für die URL CloudFront	107
Ich erhalte beim Besuch meiner Domain SSL-Zertifikat- oder HTTPS-Fehler	107
Build-Einstellungen konfigurieren	109
Befehle und Einstellungen für Build-Spezifikationen	109
Branchenspezifische Build-Einstellungen	112
Zu einem Unterordner navigieren	113
Bereitstellung des Backends mit dem Frontend für eine Gen-1-App	113
Einstellung des Ausgabeordners	114
Pakete als Teil eines Builds installieren	114
Verwenden Sie eine private NPM-Registrierung	114
Installieren von OS-Paketten	115
Schlüssel-Wert-Speicher für jeden Build	115
Überspringe den Build für einen Commit	116
Deaktivieren Sie automatische Builds	116
Aktiviert oder deaktiviert das auf Diff basierende Frontend-Build und Deployment	116
Aktivieren oder deaktivieren Sie diff-basierte Backend-Builds für eine Gen-1-App	117
Monorepo-Build-Einstellungen	118
Monorepo-Build-Spezifikation, YAML-Syntax	119
Einstellung der Umgebungsvariablen AMPLIFY_MONOREPO_APP_ROOT	122
Konfiguration der Turborepo- und pnpm-Monorepo-Apps	124
Bereitstellungen in Feature-Banches	125
Team-Workflows mit Full-Stack-Apps von Amplify Gen 2	126
Team-Workflows mit Full-Stack-Apps von Amplify Gen 1	126
Workflow für Funktionsverzweigungen	126
GitFlow Arbeitsablauf	132
Sandbox pro Entwickler	133
Musterbasierte Feature-Branch-Bereitstellungen	135
Musterbasierte Feature-Branch-Bereitstellungen für eine App, die mit einer benutzerdefinierten Domain verbunden ist	136
Automatische Generierung der Amplify-Konfiguration während der Erstellung (nur Gen 1- Apps)	136
Bedingte Backend-Builds (nur Apps der Generation 1)	138
Verwenden Sie Amplify-Backends für mehrere Apps (nur Gen-1-Apps)	139
Backends wiederverwenden, wenn Sie eine neue App erstellen	139

Verwenden Sie Backends wieder, wenn Sie eine Filiale mit einer vorhandenen App verbinden	140
Bearbeiten Sie ein vorhandenes Frontend so, dass es auf ein anderes Backend verweist ...	141
Ein Backend erstellen	143
Erstellen Sie ein Backend für eine Gen 2-App	143
Erstellen Sie ein Backend für eine Gen 1-App	143
Voraussetzungen	143
Schritt 1: Stellen Sie ein Frontend bereit	144
Schritt 2: Erstellen Sie ein Backend	145
Schritt 3: Backend mit Frontend Connect	147
Nächste Schritte	149
Manuelle Bereitstellungen	150
Manuelles Bereitstellen per Drag-and-Drop	150
Manuelles Bereitstellen von Amazon S3 oder URL	151
Fehlerbehebung beim Amazon S3 S3-Bucket-Zugriff	152
Schaltfläche „Bereitstellen“ mit einem Klick	153
Fügen Sie die Schaltfläche Deploy to Amplify Hosting zu einem Repository oder Blog hinzu	153
GitHub Zugriff einrichten	155
Installation und Autorisierung der Amplify GitHub App für eine neue Bereitstellung	155
Migration einer vorhandenen OAuth App zur Amplify GitHub App	157
Einrichtung der Amplify GitHub App für AWS CloudFormation CLI- und SDK-Bereitstellungen ..	158
Webvorschauen mit der Amplify GitHub App einrichten	159
Vorschauen von Pull-Requests	160
Webvorschauen aktivieren	160
Zugriff auf die Webvorschau mit Subdomains	162
E-Prüfung end-to-end	163
Tutorial: Richten Sie end-to-end Tests mit Cypress ein	163
Fügen Sie Tests zu Ihrer bestehenden Amplify-App hinzu	163
Tests deaktivieren	165
Weiterleitungen verwenden	167
Arten von Weiterleitungen	167
Weiterleitungen erstellen und bearbeiten	169
Reihenfolge der Weiterleitungen	170
Abfrageparameter	170
Einfache Weiterleitungen und Umschreibungen	171
Weiterleitungen für einseitige Web-Apps (SPA)	173

Reverse-Proxy umschreiben	173
Schrägstriche am Ende und saubere URLs	174
Platzhalter	174
Abfragen von Zeichenketten und Pfadparametern	175
Regionsbasierte Weiterleitungen	176
Platzhalterausdrücke in Weiterleitungen und Umschreibungen	176
Beschränken Sie den Zugriff	178
Umgebungsvariablen	180
Umgebungsvariablen Amplify	180
Festlegen von Umgebungsvariablen	186
Greifen Sie während der Erstellung auf Umgebungsvariablen zu	187
Umgebungsvariablen für serverseitige Laufzeiten zugänglich machen	188
Erstellen Sie eine neue Backend-Umgebung mit Authentifizierungsparametern für die Anmeldung über soziale Netzwerke	188
Umgebungsvariablen des Frontend-Frameworks	190
Verwaltung von Umgebungsgeheimnissen	190
Legen Sie Umgebungsgeheimnisse für eine Gen-1-App fest und greifen Sie darauf zu	190
Auf Geheimnisse der Umgebung zugreifen	191
Amplify Sie die Geheimnisse der Umgebung	191
Benutzerdefinierte Header	193
Benutzerdefiniertes Header-YAML-Format	193
Benutzerdefinierte Header einrichten	194
Migrieren von benutzerdefinierten Headern	196
Benutzerdefinierte Monorepo-Header	198
Beispiel für Sicherheitsheader	198
Benutzerdefinierte Cache-Control-Header	199
Eingehende Webhooks	200
Überwachen	201
Überwachung mit CloudWatch	201
Metriken	201
Alarmer	204
Amazon CloudWatch Logs für SSR-Apps	205
Zugriffsprotokolle	206
Analyse von Zugriffsprotokollen	207
Benachrichtigungen erstellen	208
Richten Sie E-Mail-Benachrichtigungen ein	208

Benutzerdefinierte Builds	209
Benutzerdefinierte Build-Images	209
Anforderungen an ein benutzerdefiniertes Build-Image	209
Konfiguration eines benutzerdefinierten Build-Images	210
Live-Paket-Updates	211
Konfiguration von Live-Paket-Updates	211
Eine Servicerolle hinzufügen	213
Erstellen einer Servicerolle	213
Confused-Deputy-Prävention	214
Verwaltung der App-Leistung	215
Verwenden von Headern zur Steuerung der Cache-Dauer	215
Einstellung des Cache-Control Headers zur Steigerung der Anwendungsleistung	216
Protokollieren von Amplify-API-Aufrufen mit AWS CloudTrail	217
Amplify Sie Informationen in CloudTrail	217
Verstehen von Amplify-Protokolldateieinträgen	218
Sicherheit	222
Identitäts- und Zugriffsverwaltung	222
Zielgruppe	223
Authentifizierung mit Identitäten	224
Verwalten des Zugriffs mit Richtlinien	228
So funktioniert Amplify mit IAM	231
Beispiele für identitätsbasierte Richtlinien	238
Von AWS verwaltete Richtlinien	242
Fehlerbehebung	257
Datenschutz	260
Verschlüsselung im Ruhezustand	261
Verschlüsselung während der Übertragung	261
Verwaltung von Verschlüsselungsschlüsseln	261
Compliance-Validierung	261
Sicherheit der Infrastruktur	263
Protokollierung und Überwachung	264
Serviceübergreifende Confused-Deputy-Prävention	265
Bewährte Methoden für die Gewährleistung der Sicherheit	267
Verwendung von Cookies mit der Amplify-Standarddomain	267
Kontingente	268
Fehlerbehebung	271

Allgemeine Probleme	271
HTTP 429-Statuscode (Zu viele Anfragen)	271
AL2023 Build-Image	272
Wie führe ich Amplify-Funktionen mit der Python-Laufzeit aus?	272
Wie führe ich Befehle aus, für die Superuser- oder Root-Rechte erforderlich sind	273
Benutzerdefinierte Domänen	273
Serverseitiges Rendern (SSR)	273
AWS AmplifyReferenz zum Hosting	274
AWS CloudFormation-Support	274
AWS Command Line Interface-Support	274
Unterstützung für das Tagging von Ressourcen	274
Amplify Hosting-API	274
Dokumentverlauf	275
.....	cclxxxix

Willkommen bei AWS Amplify Hosting

Amplify Hosting bietet einen Git-basierten Workflow für das Hosten serverloser Full-Stack-Webanwendungen mit kontinuierlicher Bereitstellung. Amplify stellt Ihre App im AWS globalen Content Delivery Network (CDN) bereit. Dieses Benutzerhandbuch enthält die Informationen, die Sie für den Einstieg in Amplify Hosting benötigen.

Unterstützte Frameworks

Amplify Hosting unterstützt viele gängige SSR-Frameworks, Single-Page-Application-Frameworks (SPA) und statische Site-Generatoren, einschließlich der folgenden.

SSR-Frameworks

- Next.js
- Nuxt
- Astro mit einem Community-Adapter
- SvelteKit mit einem Community-Adapter
- Jedes SSR-Framework mit einem benutzerdefinierten Adapter

SPA-Frameworks

- React
- Winkel
- Vue.js
- Ionisch
- Glut

Generatoren für statische Websites

- Elfter
- Gatsby
- Hugo
- Jekyll

- VuePress

Amplify Hosting-Funktionen

[Feature-Zweige](#)

Verwalten Sie Produktions- und Staging-Umgebungen für Ihr Frontend und Backend, indem Sie neue Filialen verbinden.

[Benutzerdefinierte Domänen](#)

Connect Sie Ihre Anwendung mit einer benutzerdefinierten Domain.

[Vorschauen von Pull-Requests](#)

Eine Vorschau der Änderungen während der Code-Reviews anzeigen.

[end-to-end E-Prüfung](#)

Verbessern Sie die Qualität Ihrer App mit end-to-end Tests.

[Passwortgeschützte Filialen](#)

Schützen Sie Ihre Web-App mit einem Passwort, damit Sie an neuen Funktionen arbeiten können, ohne sie öffentlich zugänglich zu machen.

[Leitet um und schreibt neu](#)

Richten Sie Umschreibungen und Weiterleitungen ein, um das SEO-Ranking aufrechtzuerhalten und den Traffic auf der Grundlage der Anforderungen Ihrer Kunden-App weiterzuleiten.

Atomare Bereitstellungen

Atomare Bereitstellungen eliminieren Wartungsfenster, indem sie sicherstellen, dass Ihre Web-App erst aktualisiert wird, wenn die gesamte Bereitstellung abgeschlossen ist. Dadurch entfallen Szenarien, in denen Dateien nicht korrekt hochgeladen werden.

Erste Schritte mit Amplify Hosting

Informationen zu den ersten Schritten mit Amplify Hosting finden Sie im [Erste Schritte mit Amplify Hosting](#) Tutorial. Nach Abschluss des Tutorials wissen Sie, wie Sie eine Web-App in einem Git-Repository (GitHub, BitBucket GitLab, oder AWS CodeCommit) verbinden und sie mit kontinuierlicher Bereitstellung auf Amplify Hosting bereitstellen.

Erstellen Sie ein Backend

AWS Amplify Gen 2 bietet ein TypeScript basiertes Entwicklererlebnis, bei dem der Code an erster Stelle steht, um Backends zu definieren. Um zu erfahren, wie Sie Amplify Gen 2 verwenden, um ein Backend zu erstellen und mit Ihrer App zu verbinden, siehe [Backend erstellen und verbinden](#) in den Amplify-Dokumenten.

Wenn Sie nach der Dokumentation zum Erstellen von Backends für eine Gen 1-App mithilfe der CLI und Amplify Studio suchen, finden Sie in den Gen 1 Amplify-Dokumenten unter [Build & Connect-Backend](#) weitere Informationen.

Amplify Hosting-Preise

AWS Amplify ist Teil der Kostenloses AWS-Kontingent. Sie können kostenlos starten und dann nutzungsabhängig zahlen, sobald Sie die Limits des kostenlosen Kontingents überschritten haben. Informationen zu den Gebühren für Amplify Hosting finden Sie unter [AWS Amplify Preise](#).

Erste Schritte mit Amplify Hosting

Um Ihnen zu helfen, zu verstehen, wie Amplify Hosting funktioniert, führt Sie dieses Tutorial durch das Erstellen und Bereitstellen einer Next.js Anwendung aus einem Git-Repository.

Themen

- [Voraussetzungen](#)
- [Schritt 1: Ein Git-Repository Connect](#)
- [Schritt 2: Bestätigen Sie die Build-Einstellungen](#)
- [Schritt 3: Stellen Sie die Anwendung bereit](#)
- [Schritt 4: \(Optional\) Ressourcen bereinigen](#)
- [Füge Funktionen zu deiner App hinzu](#)

Voraussetzungen

Bevor Sie mit diesem Tutorial beginnen, müssen Sie die folgenden Voraussetzungen erfüllen.

Melden Sie sich an für eine AWS-Konto

Wenn Sie noch kein AWS Kunde sind, müssen Sie [einen erstellen](#), AWS-Konto indem Sie den Online-Anweisungen folgen. Durch die Registrierung können Sie auf Amplify und andere AWS Dienste zugreifen, die Sie mit Ihrer Anwendung verwenden können.

Erstellen einer Anwendung

Erstellen Sie anhand der [create-next-app](#)Anweisungen in der Dokumentation zu Next.js eine grundlegende Anwendung Next.js, die Sie für dieses Tutorial verwenden können.

Erstellen Sie ein Git-Repository

Amplify unterstützt GitHub, Bitbucket GitLab, und AWS CodeCommit Schieben Sie Ihre create-next-app Anwendung in Ihr Git-Repository.

Schritt 1: Ein Git-Repository Connect

In diesem Schritt verbinden Sie Ihre Anwendung Next.js in einem Git-Repository mit Amplify Hosting.

So verbinden Sie eine App in einem Git-Repository

1. Öffnen Sie die [Amplify-Konsole](#).
2. Wenn Sie Ihre erste App in der aktuellen Region bereitstellen, beginnen Sie standardmäßig auf der AWS AmplifyServiceseite.

Wählen Sie oben auf der Seite Neue App erstellen aus.

3. Wählen Sie auf der Seite Start building with Amplify Ihren Git-Repository-Anbieter aus und klicken Sie dann auf Weiter.

Für GitHub Repositories verwendet Amplify die GitHub Apps-Funktion, um den Amplify-Zugriff zu autorisieren. Weitere Informationen zur Installation und Autorisierung der App finden Sie unter.

GitHub [Einrichtung des Amplify-Zugriffs auf GitHub Repositories](#)

Note

Nachdem Sie die Amplify-Konsole mit Bitbucket, oder AWS CodeCommit autorisiert haben GitLab, ruft Amplify ein Zugriffstoken vom Repository-Anbieter ab, speichert das Token jedoch nicht auf den Servern. AWS Amplify greift auf Ihr Repository nur mit Bereitstellungsschlüsseln zu, die in einem bestimmten Repository installiert sind.

4. Gehen Sie auf der Seite Repository-Zweig hinzufügen wie folgt vor:
 - a. Wählen Sie den Namen des Repositories aus, zu dem Sie eine Verbindung herstellen möchten.
 - b. Wählen Sie den Namen des Repository-Zweigs aus, zu dem eine Verbindung hergestellt werden soll.
 - c. Wählen Sie Weiter aus.

Schritt 2: Bestätigen Sie die Build-Einstellungen

Amplify erkennt automatisch die Reihenfolge der Build-Befehle, die für den Zweig ausgeführt werden sollen, den Sie bereitstellen. In diesem Schritt überprüfen und bestätigen Sie Ihre Build-Einstellungen.

Um die Build-Einstellungen für eine App zu bestätigen

1. Suchen Sie auf der Seite mit den App-Einstellungen den Abschnitt Build-Einstellungen.

Stellen Sie sicher, dass der Frontend-Build-Befehl und das Build-Ausgabeverzeichnis korrekt sind. Für diese Beispiel-App Next.js ist das Build-Ausgabeverzeichnis auf `.next` eingestellt.

2. Das Verfahren zum Hinzufügen einer Servicerolle hängt davon ab, ob Sie eine neue Rolle erstellen oder eine vorhandene verwenden möchten.
 - Um eine neue Rolle zu erstellen:
 - Wählen Sie Neue Servicerolle erstellen und verwenden aus.
 - Um eine bestehende Rolle zu verwenden:
 - a. Wählen Sie Eine bestehende Rolle verwenden aus.
 - b. Wählen Sie in der Liste der Servicerollen die zu verwendende Rolle aus.
3. Wählen Sie Weiter aus.

Schritt 3: Stellen Sie die Anwendung bereit

In diesem Schritt stellen Sie Ihre App im AWS globalen Content Delivery Network (CDN) bereit.

Um eine App zu speichern und bereitzustellen

1. Vergewissern Sie sich auf der Seite „Überprüfen“, dass Ihre Repository-Details und App-Einstellungen korrekt sind.
2. Wählen Sie Save and deploy (Speichern und Bereitstellen) aus. Ihr Frontend-Build dauert in der Regel 1 bis 2 Minuten, kann aber je nach Größe der App variieren.
3. Wenn die Bereitstellung abgeschlossen ist, können Sie Ihre App über den Link zur `amplifyapp.com` Standarddomäne anzeigen.

Note

Um die Sicherheit Ihrer Amplify-Anwendungen zu erhöhen, ist die Domain `amplifyapp.com` in der [Public Suffix List \(PSL\)](#) registriert. Aus Sicherheitsgründen empfehlen wir Ihnen, Cookies mit einem `__Host-` Präfix zu verwenden, falls Sie jemals sensible Cookies im Standard-Domainnamen für Ihre Amplify-Anwendungen einrichten müssen. Diese Vorgehensweise hilft Ihnen dabei, Ihre Domain vor CSRF-Versuchen (Cross-Site Request Forgery Attempts,

Anforderungsfälschung zwischen Websites) zu schützen. Weitere Informationen finden Sie auf der [Set-Cookie](#)-Seite im Mozilla Developer Network.

Schritt 4: (Optional) Ressourcen bereinigen

Wenn Sie die App, die Sie für das Tutorial bereitgestellt haben, nicht mehr benötigen, können Sie sie löschen. Mit diesem Schritt wird sichergestellt, dass Ihnen keine Ressourcen in Rechnung gestellt werden, die Sie nicht nutzen.

Um eine App zu löschen

1. Wählen Sie im Navigationsbereich im Menü App-Einstellungen die Option Allgemeine Einstellungen aus.
2. Wählen Sie auf der Seite Allgemeine Einstellungen die Option App löschen aus.
3. Geben Sie im Bestätigungsfenster ein **delete**. Wählen Sie dann App löschen.

Füge Funktionen zu deiner App hinzu

Nachdem Sie eine App für Amplify bereitgestellt haben, können Sie einige der folgenden Funktionen erkunden, die für Ihre gehostete Anwendung verfügbar sind.

Umgebungsvariablen

Anwendungen benötigen zur Laufzeit häufig Konfigurationsinformationen. Bei diesen Konfigurationen kann es sich um Datenbankverbindungsdetails, API-Schlüssel oder Parameter handeln. Umgebungsvariablen bieten eine Möglichkeit, diese Konfigurationen während der Erstellung verfügbar zu machen. Weitere Informationen finden Sie unter [Umgebungsvariablen](#).

Benutzerdefinierte Domänen

In diesem Tutorial hostet Amplify Ihre App für Sie auf der `amplifyapp.com` Standarddomain mit einer URL wie `https://branch-name.d1m7bkiki6tdw1.amplifyapp.com`. Wenn Sie Ihre App mit einer benutzerdefinierten Domain verbinden, sehen Benutzer, dass Ihre App auf einer benutzerdefinierten URL gehostet wird, z. B. `https://www.example.com`. Weitere Informationen finden Sie unter [Benutzerdefinierte Domains einrichten](#).

Vorschauen von Pull-Requests

Web-Pull-Request-Vorschauen bieten Teams die Möglichkeit, eine Vorschau der Änderungen anhand von Pull Requests (PRs) anzuzeigen, bevor sie den Code mit einem Produktions- oder Integrationszweig zusammenführen. Weitere Informationen finden Sie unter [Webvorschauen](#) für Pull-Requests.

Verwalten mehrerer Umgebungen

Um zu erfahren, wie Amplify mit Feature Branches und GitFlow Workflows arbeitet, um mehrere Bereitstellungen zu unterstützen, siehe [Feature Branch-Bereitstellungen und Team-Workflows](#).

Stellen Sie serverseitig gerenderte Apps mit Amplify Hosting bereit

Sie können AWS Amplify damit Web-Apps bereitstellen und hosten, die serverseitiges Rendering (SSR) verwenden. Amplify Hosting erkennt automatisch Anwendungen, die mit dem Framework Next.js erstellt wurden, und Sie müssen keine manuelle Konfiguration in der AWS Management Console vornehmen. Amplify unterstützt auch jedes Javascript-basierte SSR-Framework mit einem Open-Source-Build-Adapter, der die Build-Ausgabe einer Anwendung in die Verzeichnisstruktur umwandelt, die Amplify Hosting erwartet.

In den folgenden Themen erfahren Sie, wie Amplify SSR unterstützt.

Themen

- [Was ist serverseitiges Rendern](#)
- [Amplify Sie die Unterstützung für SSR-Frameworks](#)
- [Verwenden der Amplify Hosting-Bereitstellungsspezifikation zur Konfiguration der Build-Ausgabe](#)
- [Bildoptimierung für SSR-Apps](#)
- [Unterstützung der Version Node.js für Apps der Version Next.js](#)
- [Fehlerbehebung bei SSR-Bereitstellungen](#)
- [Amplify Sie die Unterstützung für Next.js](#)

Was ist serverseitiges Rendern

Amplify unterstützt die Bereitstellung und das Hosting von statischen Web-Apps, die mit Single-Page-Application-Frameworks (SPA) wie React erstellt wurden, und von Apps, die mit einem Static Site Generator (SSG) wie Gatsby erstellt wurden. Statische Web-Apps bestehen aus einer Kombination von Dateien wie HTML, CSS und JavaScript Dateien, die in einem Content Delivery Network (CDN) gespeichert sind. Wenn ein Clientbrowser eine Anfrage an die Website stellt, sendet der Server eine Seite mit einer HTTP-Antwort an den Client zurück, und der Clientbrowser interpretiert den Inhalt und zeigt ihn dem Benutzer an.

Amplify unterstützt auch Web-Apps mit serverseitigem Rendering (SSR). Wenn ein Client eine Anfrage an eine SSR-Seite sendet, wird der HTML-Code für die Seite bei jeder Anfrage auf dem Server erstellt. SSR ermöglicht es einem Entwickler, eine Website pro Anfrage und pro Benutzer

anzupassen. Darüber hinaus kann SSR die Leistung und die Suchmaschinenoptimierung (SEO) einer Website verbessern.

Amplify Sie die Unterstützung für SSR-Frameworks

Amplify Hosting unterstützt jedes JavaScript basierte SSR-Framework mit einem Bereitstellungspaket, das der von Amplify erwarteten Build-Ausgabe entspricht. Amplify Hosting bietet eine Bereitstellungsspezifikation, die die Dateien und die Verzeichnisstruktur für die Ausgabe des Builds einer Anwendung für jedes SSR-Framework standardisiert.

Framework-Autoren können die dateisystembasierte Bereitstellungsspezifikation verwenden, um Open-Source-Build-Adapter zu entwickeln, die auf ihre spezifischen Frameworks zugeschnitten sind. Diese Adapter verwandeln die Build-Ausgabe einer App in ein Bereitstellungspaket, das der erwarteten Verzeichnisstruktur von Amplify Hosting entspricht. Dieses Bereitstellungspaket enthält alle zum Hosten einer App erforderlichen Dateien und Ressourcen, einschließlich der Laufzeitkonfiguration, z. B. Routing-Regeln.

Wenn Sie kein Framework oder keinen Framework-Adapter verwenden, können Sie Ihre eigene Lösung entwickeln, um ein Bereitstellungspaket zu generieren, das der erwarteten Verzeichnisstruktur von Amplify Hosting entspricht.

Amplify Hosting unterstützt die folgenden primitiven Typen: Statische Assets, Compute, Bildoptimierung und Routing-Regeln. Sie können diese primitiven Typen nutzen, um Anwendungen mit umfangreicheren Funktionen bereitzustellen. Ausführliche Informationen zu den einzelnen primitiven Typen finden Sie unter [Amplify Sie die Unterstützung des primitiven SSR-Typs](#).

Sie können aus den folgenden Szenarien wählen, um mit der Bereitstellung einer SSR-App auf Amplify zu beginnen.

Stellen Sie eine Next.js App bereit

Amplify unterstützt Anwendungen, die mit Next.js erstellt wurden, ohne dass ein Adapter oder eine manuelle Konfiguration in der Konsole erforderlich sind. Weitere Informationen finden Sie unter [Amplify Sie die Unterstützung für Next.js](#).

Stellen Sie eine App bereit, die einen Framework-Adapter verwendet

Sie können auf jeden verfügbaren Open-Source-Framework-Adapter verweisen, um Ihre SSR-App auf Amplify Hosting bereitzustellen. Weitere Informationen finden Sie unter [Verwenden Sie einen Framework-Adapter](#).

Für das Nuxt-Framework ist ein Adapter verfügbar. Weitere Informationen zur Verwendung dieses Adapters finden Sie in der [Nuxt-Dokumentation](#).

Erstellen Sie einen Framework-Adapter

Framework-Autoren, die Funktionen integrieren möchten, die ein Framework bietet, können die Amplify Hosting-Bereitstellungsspezifikation verwenden, um Ihre Build-Ausgabe so zu konfigurieren, dass sie der von Amplify erwarteten Struktur entspricht. Weitere Informationen finden Sie unter [Bereitstellen eines Express-Servers mithilfe des Deployment-Manifests](#).

Konfigurieren Sie ein Post-Build-Skript

Sie können die Amplify Hosting-Bereitstellungsspezifikation verwenden, um Ihre Build-Ausgabe nach Bedarf für bestimmte Szenarien zu bearbeiten. Weitere Informationen finden Sie unter [Verwenden der Amplify Hosting-Bereitstellungsspezifikation zur Konfiguration der Build-Ausgabe](#). Ein Beispiel finden Sie unter [Bereitstellen eines Express-Servers mithilfe des Deployment-Manifests](#).

Stellen Sie eine SSR-App für Amplify bereit

Sie können die Anweisungen in diesem Thema verwenden, um eine App bereitzustellen, die mit einem beliebigen Framework erstellt wurde, mit einem Bereitstellungspaket, das der von Amplify erwarteten Build-Ausgabe entspricht. Wenn Sie eine Next.js -Anwendung bereitstellen, ist kein Adapter erforderlich.

Wenn Sie eine SSR-App bereitstellen, die einen Framework-Adapter verwendet, müssen Sie zuerst den Adapter installieren und konfigurieren. Anweisungen finden Sie unter [Verwenden Sie einen Framework-Adapter](#).

Um eine SSR-App für Amplify Hosting bereitzustellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie auf der Seite Alle Apps die Option Neue App erstellen aus.
3. Wählen Sie auf der Seite Start building with Amplify Ihren Git-Repository-Anbieter aus und klicken Sie dann auf Weiter.
4. Gehen Sie auf der Seite Repository-Zweig hinzufügen wie folgt vor:
 - a. Wählen Sie den Namen des Repositorys aus, zu dem Sie eine Verbindung herstellen möchten.

- b. Wählen Sie den Namen des Repository-Zweigs aus, zu dem eine Verbindung hergestellt werden soll.
 - c. Wählen Sie Weiter aus.
5. Auf der Seite mit den App-Einstellungen erkennt Amplify automatisch Next.js SSR-Apps.

Wenn Sie eine SSR-App bereitstellen, die einen Adapter für ein anderes Framework verwendet, müssen Sie Amazon CloudWatch Logs explizit aktivieren. Öffnen Sie den Abschnitt Erweiterte Einstellungen und wählen Sie dann im Abschnitt Serverseitiges Rendering (SSR) -Bereitstellung die Option SSR-App-Logs aktivieren aus.

6. Für die App ist eine IAM-Servicerolle erforderlich, von der Amplify übernimmt, um Protokolle an Ihre zu senden. AWS-Konto

Das Verfahren zum Hinzufügen einer Servicerolle hängt davon ab, ob Sie eine neue Rolle erstellen oder eine bestehende verwenden möchten.

- Um eine neue Rolle zu erstellen:
 - Wählen Sie Neue Servicerolle erstellen und verwenden aus.
 - Um eine bestehende Rolle zu verwenden:
 - a. Wählen Sie Eine bestehende Rolle verwenden aus.
 - b. Wählen Sie in der Liste der Servicerollen die zu verwendende Rolle aus.
7. Wählen Sie Weiter aus.
 8. Wählen Sie auf der Seite Überprüfen die Option Speichern und bereitstellen aus.

Verwenden Sie einen Framework-Adapter

Sie können jeden SSR-Framework-Build-Adapter installieren und verwenden, der für die Integration mit Amplify Hosting erstellt wurde. Jedes Framework, das einen Adapter anbietet, bestimmt, wie der Adapter konfiguriert und mit seinem Build-Prozess verbunden ist. In der Regel installieren Sie den Adapter als NPM-Entwicklungsabhängigkeit.

Nachdem Sie eine App mit einem Framework erstellt haben, erfahren Sie in der Dokumentation des Frameworks, wie Sie den Amplify Hosting-Adapter installieren und in der Konfigurationsdatei Ihrer Anwendung konfigurieren.

Erstellen Sie als Nächstes eine `amplify.yml` Datei im Stammverzeichnis Ihres Projekts. Stellen Sie in der `amplify.yml` Datei `baseDirectory` das Build-Ausgabeverzeichnis Ihrer Anwendung ein.

Das Framework führt den Adapter während des Build-Prozesses aus, um die Ausgabe in das Amplify Hosting-Bereitstellungspaket umzuwandeln.

Der Name des Build-Ausgabeverzeichnis kann beliebig sein, aber der `.amplify-hosting` Dateiname hat Bedeutung. Amplify sucht zuerst nach einem Verzeichnis, das `baseDirectory` als definiert ist. Falls es existiert, sucht Amplify dort nach der Build-Ausgabe. Wenn das Verzeichnis nicht existiert, sucht Amplify darin nach der Build-Ausgabe `.amplify-hosting`, auch wenn sie nicht vom Kunden definiert wurde.

Das Folgende ist ein Beispiel für die Build-Einstellungen für eine App. Die Einstellung `baseDirectory` gibt `.amplify-hosting` an, dass sich die Build-Ausgabe im `.amplify-hosting` Ordner befindet. Solange der Inhalt des `.amplify-hosting` Ordners der Amplify Hosting-Bereitstellungsspezifikation entspricht, wird die App erfolgreich bereitgestellt.

```
version: 1
frontend:
  preBuild:
    commands:
      - npm install
  build:
    commands:
      - npm run build
  artifacts:
    baseDirectory: .amplify-hosting
```

Nachdem Ihre App für die Verwendung eines Framework-Adapters konfiguriert wurde, können Sie sie auf Amplify Hosting bereitstellen. Detaillierte Anweisungen finden Sie unter [Stellen Sie eine SSR-App für Amplify bereit](#).

Verwenden der Amplify Hosting-Bereitstellungsspezifikation zur Konfiguration der Build-Ausgabe

Verwenden Sie die Amplify-Bereitstellungsspezifikation, um die Build-Ausgabe für ein SSR-Framework zu konfigurieren, das Sie in Amplify Hosting integrieren möchten. Wenn Sie ein Framework-Autor sind, können Sie anhand der Deployment-Spezifikation verstehen, wie Sie die von Amplify erwartete Build-Ausgabe strukturieren. Wenn Sie kein Framework verwenden, können Sie Ihre eigene Lösung entwickeln, um eine Build-Ausgabe zu generieren, die Amplify erwartet.

Spezifikation für die Bereitstellung von Amplify Hosting

Die Amplify Hosting-Bereitstellungsspezifikation ist eine dateisystembasierte Spezifikation, die die Verzeichnisstruktur definiert, die Bereitstellungen für Amplify Hosting erleichtert. Ein Framework kann diese erwartete Verzeichnisstruktur als Ausgabe seines Build-Befehls generieren, sodass das Framework die Service-Primitivtypen von Amplify Hosting nutzen kann. Amplify Hosting versteht die Struktur des Bereitstellungspakets und stellt es entsprechend bereit.

Das Folgende ist ein Beispiel für die Ordnerstruktur, die Amplify für das Bereitstellungspaket erwartet. Auf einer höheren Ebene hat es einen Ordner mit dem Namen `static`, einen Ordner mit dem Namen `compute` und eine Bereitstellungsmanifestdatei mit dem Namen `deploy-manifest.json`.

```
.amplify-hosting/
### compute/
#   ### default/
#     ### chunks/
#     #   ### app/
#     #     ### _nuxt/
#     #     #   ### index-xxx.mjs
#     #     #   ### index-styles.xxx.js
#     #     ### server.mjs
#     ### node_modules/
#     ### server.js
### static/
#   ### css/
#   #   ### nuxt-google-fonts.css
#   ### fonts/
#   #   ### font.woff2
#   ### _nuxt/
#   #   ### builds/
#   #   #   ### latest.json
#   #   ### entry.xxx.js
#   ### favicon.ico
#   ### robots.txt
### deploy-manifest.json
```

Amplify Sie die Unterstützung des primitiven SSR-Typs

Die Amplify Hosting-Bereitstellungsspezifikation definiert einen Vertrag, der den folgenden primitiven Typen genau entspricht.

Statische Vermögenswerte

Bietet Frameworks die Möglichkeit, statische Dateien zu hosten.

Datenverarbeitung

Bietet Frameworks die Möglichkeit, einen Node.js HTTP-Server auf Port 3000 auszuführen.

Bildoptimierung

Stellt Frameworks einen Dienst zur Optimierung von Bildern zur Laufzeit zur Verfügung.

Routing-Regeln

Stellt Frameworks einen Mechanismus zur Verfügung, mit dem eingehende Anforderungspfade bestimmten Zielen zugeordnet werden können.

Das `.amplify-hosting/static` Verzeichnis

Sie müssen alle öffentlich zugänglichen statischen Dateien, die über die Anwendungs-URL bereitgestellt werden sollen, im `.amplify-hosting/static` Verzeichnis ablegen. Die Dateien in diesem Verzeichnis werden über den primitiven Typ `Static Assets` bereitgestellt.

Auf statische Dateien kann im Stammverzeichnis (`/`) der Anwendungs-URL zugegriffen werden, ohne dass ihr Inhalt, ihr Dateiname oder ihre Erweiterung geändert werden. Darüber hinaus werden Unterverzeichnisse in der URL-Struktur beibehalten und vor dem Dateinamen angezeigt. Als Beispiel `.amplify-hosting/static/favicon.ico` wird von `https://myAppId.amplify-hostingapp.com/favicon.ico` und `.amplify-hosting/static/_nuxt/main.js` wird bedient von `https://myAppId.amplify-hostingapp.com/_nuxt/main.js`

Wenn ein Framework die Möglichkeit unterstützt, den Basispfad der Anwendung zu ändern, muss es den Basispfad den statischen Assets innerhalb des `.amplify-hosting/static` Verzeichnisses voranstellen. Wenn der Basispfad beispielsweise lautet `/folder1/folder2`, dann `main.css` wird die Build-Ausgabe für ein statisches Asset aufgerufen. `.amplify-hosting/static/folder1/folder2/main.css`

Das `.amplify-hosting/compute` Verzeichnis

Eine einzelne Rechenressource wird durch ein einzelnes Unterverzeichnis mit dem Namen repräsentiert, das innerhalb des `.amplify-hosting/compute` Verzeichnisses `default` enthalten ist. Der Pfad ist `.amplify-hosting/compute/default`. Diese Rechenressource ist dem primitiven `Compute`-Typ von Amplify Hosting zugeordnet.

Der Inhalt des `default` Unterverzeichnisses muss den folgenden Regeln entsprechen.

- Im Stammverzeichnis des `default` Unterverzeichnisses muss eine Datei vorhanden sein, die als Einstiegspunkt zur Rechenressource dient.
- Bei der Einstiegspunktdatei muss es sich um ein Modul `Node.js` handeln und sie muss einen HTTP-Server starten, der auf Port 3000 lauscht.
- Sie können andere Dateien im `default` Unterverzeichnis platzieren und über den Code in der Einstiegspunktdatei auf sie verweisen.
- Der Inhalt des Unterverzeichnisses muss eigenständig sein. Der Code im Einstiegspunktmodul kann auf keine Module außerhalb des Unterverzeichnisses verweisen. Beachten Sie, dass Frameworks ihren HTTP-Server nach Belieben bündeln können. Wenn der Rechengvorgang mit dem `node server.js` Befehl, wo sich der Name der Eintragsdatei `server.js` befindet, aus dem Unterverzeichnis heraus initiiert werden kann, geht Amplify davon aus, dass die Verzeichnisstruktur der Deployment-Spezifikation entspricht.

Amplify Hosting bündelt und stellt alle Dateien im `default` Unterverzeichnis auf einer bereitgestellten Rechenressource bereit. Jeder Rechenressource werden 512 MB flüchtiger Speicher zugewiesen. Dieser Speicher wird nicht von allen Ausführungsinstanzen gemeinsam genutzt, sondern von nachfolgenden Aufrufen innerhalb derselben Ausführungsinstanz gemeinsam genutzt. Ausführungsinstanzen sind auf eine maximale Ausführungszeit von 15 Minuten begrenzt, und der einzige beschreibbare Pfad innerhalb der Ausführungsinstanz ist das Verzeichnis `/tmp`. Die komprimierte Größe jedes Rechenressourcenpakets darf 220 MB nicht überschreiten. Beispielsweise darf das `.amplify/compute/default` Unterverzeichnis im komprimierten Zustand 220 MB nicht überschreiten.

Die Datei `.amplify-hosting/deploy-manifest.json`

Verwenden Sie die `deploy-manifest.json` Datei, um die Konfigurationsdetails und Metadaten für eine Bereitstellung zu speichern. Eine `deploy-manifest.json` Datei muss mindestens ein `version` Attribut, das `routes` Attribut mit einer angegebenen Catch-All-Route und das Attribut mit den `framework` angegebenen Framework-Metadaten enthalten.

Die folgende Objektdefinition veranschaulicht die Konfiguration für ein Bereitstellungsmanifest.

```
type DeployManifest = {  
  version: 1;  
  routes: Route[];  
  computeResources?: ComputeResource[];
```

```
imageSettings?: ImageSettings;  
framework: FrameworkMetadata;  
};
```

In den folgenden Themen werden die Details und die Verwendung der einzelnen Attribute im Bereitstellungsmanifest beschrieben.

Verwenden des Versionsattributs

Das `version` Attribut definiert die Version der Bereitstellungsspezifikation, die Sie implementieren. Derzeit ist Version 1 die einzige Version für die Amplify Hosting-Bereitstellungsspezifikation. Das folgende JSON-Beispiel demonstriert die Verwendung des `version` Attributs.

```
"version": 1
```

Verwenden des Route-Attributs

Das `routes` Attribut ermöglicht es Frameworks, den primitiven Typ der Amplify Hosting-Routing-Regeln zu nutzen. Routing-Regeln bieten einen Mechanismus zum Weiterleiten eingehender Anforderungspfade an ein bestimmtes Ziel im Bereitstellungspaket. Routing-Regeln geben nur das Ziel einer eingehenden Anfrage vor und werden angewendet, nachdem die Anfrage durch Umschreib- und Umleitungsregeln transformiert wurde. Weitere Informationen darüber, wie Amplify Hosting mit Umschreibungen und Weiterleitungen umgeht, finden Sie unter [Weiterleitungen verwenden](#)

Routing-Regeln schreiben oder transformieren die Anfrage nicht. Wenn eine eingehende Anfrage dem Pfadmuster für eine Route entspricht, wird die Anfrage unverändert an das Ziel der Route weitergeleitet.

Die im `routes` Array angegebenen Routing-Regeln müssen den folgenden Regeln entsprechen.

- Es muss eine Sammelroute angegeben werden. Eine Catch-All-Route hat das `/*` Muster, das allen eingehenden Anfragen entspricht.
- Das `routes` Array kann maximal 25 Elemente enthalten.
- Sie müssen entweder eine `Static Route` oder eine `Compute Route` angeben.
- Wenn Sie eine `Static Route` angeben, muss das `.amplify-hosting/static` Verzeichnis existieren.
- Wenn Sie eine `Compute Route` angeben, muss das `.amplify-hosting/compute` Verzeichnis existieren.

- Wenn Sie eine `ImageOptimization` Route angeben, müssen Sie auch eine `Compute` Route angeben. Dies ist erforderlich, da die Bildoptimierung für rein statische Anwendungen noch nicht unterstützt wird.

Die folgende Objektdefinition veranschaulicht die Konfiguration für das `Route` Objekt.

```
type Route = {
  path: string;
  target: Target;
  fallback?: Target;
}
```

In der folgenden Tabelle werden die Eigenschaften des `Route` Objekts beschrieben.

Schlüssel	Typ	Erforderlich	Beschreibung
Pfad	String	Ja	<p>Definiert ein Muster, das den Pfaden eingehender Anfragen entspricht (ohne Querystring).</p> <p>Die maximale Pfadlänge beträgt 255 Zeichen.</p> <p>Ein Pfad muss mit einem Schrägstrich / beginnen.</p> <p>Ein Pfad kann jedes der folgenden Zeichen enthalten: [A-Z], [a-z], [0-9], [_.*\$/~"@: +].</p> <p>Für den Mustervergleich werden nur die folgenden Platzhalterzeichen unterstützt:</p>

Schlüssel	Typ	Erforderlich	Beschreibung
			<ul style="list-style-type: none"> • *(entspricht 0 oder mehr Zeichen) • Das /* Muster wird als Catch-All-Muster bezeichnet und entspricht allen eingehenden Anfragen.
Ziel	Ziel	Ja	<p>Ein Objekt, das das Ziel definiert, an das die übereinstimmende Anfrage weitergeleitet werden soll.</p> <p>Wenn eine Compute Route angegeben ist, ComputeResource muss eine entsprechende vorhanden sein.</p> <p>Wenn eine ImageOptimization Route angegeben ist, imageSettings muss diese ebenfalls angegeben werden.</p>

Schlüssel	Typ	Erforderlich	Beschreibung
Reserve	Ziel	Nein	<p>Ein Objekt, das das Ziel definiert, auf das zurückgegriffen werden soll, falls das ursprüngliche Ziel einen 404-Fehler zurückgibt.</p> <p>Die <code>target</code> Art und die <code>fallback</code> Art können für eine angegebene Route nicht identisch sein. Beispielsweise ist ein Fallback von <code>Static</code> bis nicht <code>Static</code> zulässig. Fallbacks werden nur für GET-Anfragen unterstützt, die keinen Hauptteil haben. Wenn in der Anfrage ein Hauptteil vorhanden ist, wird er während des Fallbacks gelöscht.</p>

Die folgende Objektdefinition veranschaulicht die Konfiguration für das Target Objekt.

```
type Target = {  
  kind: TargetKind;  
  src?: string;  
  cacheControl?: string;  
}
```

In der folgenden Tabelle werden die Eigenschaften des Target Objekts beschrieben.

Schlüssel	Typ	Erforderlich	Beschreibung
nett	Zieltyp	Ja	Und enum das definiert den Zieltyp. Gültige Werte sind <code>Static</code> , <code>Compute</code> und <code>ImageOptimization</code> .
src	String	Ja für Compute Nein für andere primitive Typen	Eine Zeichenfolge, die den Namen des Unterverzeichnisses im Bereitstellungspaket angibt, das den ausführbaren Code des primitiven Typs enthält. Gültig und nur für den primitiven Compute-Typ erforderlich. Der Wert muss auf eine der Rechenressourcen verweisen, die im Bereitstellungspaket vorhanden sind. Derzeit ist der einzige unterstützte Wert für dieses Feld <code>default</code> .
CacheControl	String	Nein	Eine Zeichenfolge, die den Wert des Cache-Control-Headers angibt, der auf die Antwort angewendet werden soll. Gilt

Schlüssel	Typ	Erforderlich	Beschreibung
			<p>nur für die Typen Static und ImageOptimization Primitive.</p> <p>Der angegebene Wert wird durch benutzerdefinierte Header überschrieben.</p> <p>Weitere Informationen zu den Kunden-Headern von Amplify Hosting finden Sie unter. Benutzerdefinierte Header</p> <div data-bbox="1187 890 1508 1495" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px;"><p> Note</p><p>Dieser Cache-Control-Header wird nur auf erfolgreiche Antworten angewendet, deren Statuscode auf 200 (OK) gesetzt ist.</p></div>

Die folgende Objektdefinition veranschaulicht die Verwendung der Aufzählung `TargetKind`.

```
enum TargetKind {  
    Static = "Static",  
    Compute = "Compute",  
    ImageOptimization = "ImageOptimization"  
}
```

Die folgende Liste spezifiziert die gültigen Werte für die TargetKind Aufzählung.

Statisch

Leitet Anfragen an den primitiven Typ statischer Objekte weiter.

Datenverarbeitung

Leitet Anfragen an den primitiven Typ Compute weiter.

ImageOptimization

Leitet Anfragen an den primitiven Typ der Bildoptimierung weiter.

Das folgende JSON-Beispiel zeigt die Verwendung des `routes` Attributs mit mehreren angegebenen Routing-Regeln.

```
"routes": [  
  {  
    "path": "/_nuxt/image",  
    "target": {  
      "kind": "ImageOptimization",  
      "cacheControl": "public, max-age=3600, immutable"  
    }  
  },  
  {  
    "path": "/_nuxt/builds/meta/*",  
    "target": {  
      "cacheControl": "public, max-age=31536000, immutable",  
      "kind": "Static"  
    }  
  },  
  {  
    "path": "/_nuxt/builds/*",  
    "target": {  
      "cacheControl": "public, max-age=1, immutable",  
      "kind": "Static"  
    }  
  },  
  {  
    "path": "/_nuxt/*",  
    "target": {  
      "cacheControl": "public, max-age=31536000, immutable",  
      "kind": "Static"  
    }  
  }  
]
```



```
    }
  },
  {
    "path": "/*.*",
    "target": {
      "kind": "Static"
    },
    "fallback": {
      "kind": "Compute",
      "src": "default"
    }
  },
  {
    "path": "/*",
    "target": {
      "kind": "Compute",
      "src": "default"
    }
  }
]
]
```

Weitere Informationen zur Angabe von Routing-Regeln in Ihrem Bereitstellungsmanifest finden Sie unter [Bewährte Methoden für die Konfiguration von Routing-Regeln](#)

Verwenden des ComputeResources-Attributs

Das `computeResources` Attribut ermöglicht es Frameworks, Metadaten zu den bereitgestellten Rechenressourcen bereitzustellen. Jeder Rechenressource muss eine entsprechende Route zugeordnet sein.

Die folgende Objektdefinition demonstriert die Verwendung für das `ComputeResource` Objekt.

```
type ComputeResource = {
  name: string;
  runtime: ComputeRuntime;
  entrypoint: string;
};

type ComputeRuntime = 'nodejs16.x' | 'nodejs18.x' | 'nodejs20.x';
```

In der folgenden Tabelle werden die Eigenschaften des `ComputeResource` Objekts beschrieben.

Schlüssel	Typ	Erforderlich	Beschreibung
name	String	Ja	<p>Gibt den Namen der Rechenressource an. Der Name muss mit dem Namen des Unterverzeichnisses innerhalb von übereinstimmen. <code>.amplify-hosting/compute directory</code></p> <p>Für Version 1 der Bereitstellungsspezifikation ist default der einzig gültige Wert.</p>
runtime	ComputeRuntime	Ja	<p>Definiert die Laufzeit für die bereitgestellte Rechenressource.</p> <p>Gültige Werte sind <code>nodejs16.x</code>, <code>nodejs18.x</code> und <code>nodejs20.x</code>.</p>
Einstiegspunkt	String	Ja	<p>Gibt den Namen der Startdatei an, von der aus Code für die angegebene Rechenressource ausgeführt wird. Die Datei muss sich in dem Unterverzeichnis</p>

Schlüssel	Typ	Erforderlich	Beschreibung
			befinden, das eine Rechenressource darstellt.

Wenn Sie eine Verzeichnisstruktur haben, die wie folgt aussieht.

```
.amplify-hosting
|---compute
|   |---default
|       |---index.js
```

Das JSON für das computeResource Attribut wird wie folgt aussehen.

```
"computeResources": [
  {
    "name": "default",
    "runtime": "nodejs16.x",
    "entrypoint": "index.js",
  }
]
```

Verwenden des ImageSettings-Attributs

Das imageSettings Attribut ermöglicht es Frameworks, das Verhalten des primitiven Typs der Bildoptimierung anzupassen, der eine On-Demand-Optimierung von Bildern zur Laufzeit ermöglicht.

Die folgende Objektdefinition veranschaulicht die Verwendung des ImageSettings Objekts.

```
type ImageSettings = {
  sizes: number[];
  domains: string[];
  remotePatterns: RemotePattern[];
  formats: ImageFormat[];
  mininumCacheTTL: number;
  dangerouslyAllowSVG: boolean;
};

type ImageFormat = 'image/avif' | 'image/webp' | 'image/png' | 'image/jpeg';
```

In der folgenden Tabelle werden die Eigenschaften des ImageSettings Objekts beschrieben.

Schlüssel	Typ	Erforderlich	Beschreibung
Größen	Nummer []	Ja	Eine Reihe unterstützter Bildbreiten.
domains	Zeichenfolge []	Ja	Eine Reihe zulässiger externer Domänen, für die die Bildoptimierung verwendet werden kann. Lassen Sie das Array leer, damit nur die Bereitstellungsdomäne die Bildoptimierung verwenden kann.
RemotePatterns	RemotePattern[]	Ja	Eine Reihe zulässiger externer Muster, für die Bildoptimierung verwendet werden kann. Ähnlich wie Domänen, bietet jedoch mehr Kontrolle mit regulären Ausdrücken (Regex).
Formate	ImageFormat[]	Ja	Eine Reihe von zulässigen Ausgabebildformaten.
Minimaler CacheTTL	Zahl	Ja	Die Cache-Dauer in Sekunden für die optimierten Bilder.

Schlüssel	Typ	Erforderlich	Beschreibung
SVG ist gefährlich	Boolesch	Ja	Erlaubt SVG-Eingabe-Bild-URLs. Dies ist aus Sicherheitsgründen standardmäßig deaktiviert.

Die folgende Objektdefinition demonstriert die Verwendung für das RemotePattern Objekt.

```
type RemotePattern = {
  protocol?: 'http' | 'https';
  hostname: string;
  port?: string;
  pathname?: string;
}
```

In der folgenden Tabelle werden die Eigenschaften des RemotePattern Objekts beschrieben.

Schlüssel	Typ	Erforderlich	Beschreibung
Protokoll	String	Nein	Das Protokoll des zulässigen Remote-Patterns. Gültige Werte sind http oder https.
hostname	String	Ja	Der Hostname des zulässigen Remote-Patterns. Sie können einen Literalwert oder einen Platzhalter angeben. Ein einzelnes `*` entspricht einer einzelnen Subdomain

Schlüssel	Typ	Erforderlich	Beschreibung
			. Ein doppeltes `**` entspricht einer beliebigen Anzahl von Subdomains. Amplify erlaubt keine pauschalen Platzhalter, wenn nur `**` angegeben ist.
port	String	Nein	Der Port des erlaubten Remote-Musters.
Pfadname	String	Nein	Der Pfadname des zulässigen Remote-Patterns.

Das folgende Beispiel demonstriert das `imageSettings` Attribut.

```
"imageSettings": {
  "sizes": [
    100,
    200
  ],
  "domains": [
    "example.com"
  ],
  "remotePatterns": [
    {
      "protocol": "https",
      "hostname": "example.com",
      "port": "",
      "pathname": "/*",
    }
  ],
  "formats": [
    "image/webp"
  ],
}
```

```
"minumumCacheTTL": 60,
  "dangerouslyAllowSVG": false
}
```

Verwenden des Framework-Attributs

Verwenden Sie das `framework` Attribut, um Framework-Metadaten anzugeben.

Die folgende Objektdefinition veranschaulicht die Konfiguration für das `FrameworkMetadata` Objekt.

```
type FrameworkMetadata = {
  name: string;
  version: string;
}
```

In der folgenden Tabelle werden die Eigenschaften des `FrameworkMetadata` Objekts beschrieben.

Schlüssel	Typ	Erforderlich	Beschreibung
<code>name</code>	String	Ja	Der Name des Frameworks.
<code>version</code>	String	Ja	Die Version des Frameworks. Es muss sich um eine gültige semantische Versionierungszeichenfolge (Semver) handeln.

Bewährte Methoden für die Konfiguration von Routing-Regeln

Routingregeln bieten einen Mechanismus zum Weiterleiten eingehender Anforderungspfade an bestimmte Ziele im Bereitstellungspaket. In einem Bereitstellungspaket können Framework-Autoren Dateien an die Build-Ausgabe ausgeben, die für eines der folgenden Ziele bereitgestellt werden:

- Primitiver Typ statischer Objekte — Dateien sind im `.amplify-hosting/static` Verzeichnis enthalten.
- Primitiven Typ berechnen — Dateien sind im `.amplify-hosting/compute/default` Verzeichnis enthalten.

Framework-Autoren stellen in der Deploy-Manifestdatei auch eine Reihe von Routing-Regeln bereit. Jede Regel im Array wird nacheinander mit der eingehenden Anfrage abgeglichen, bis eine Übereinstimmung gefunden wird. Wenn es eine passende Regel gibt, wird die Anfrage an das in der Abgleichsregel angegebene Ziel weitergeleitet. Optional kann für jede Regel ein Fallback-Ziel angegeben werden. Wenn das ursprüngliche Ziel einen 404-Fehler zurückgibt, wird die Anfrage an das Ausweichziel weitergeleitet.

Gemäß der Bereitstellungsspezifikation muss es sich bei der letzten Regel in der Durchlaufreihenfolge um eine Sammelregel handeln. Eine Catch-All-Regel wird mit dem `/*` Pfad angegeben. Wenn die eingehende Anfrage mit keiner der vorherigen Routen im Routing-Regel-Array übereinstimmt, wird die Anfrage an das Ziel der Catch-All-Regel weitergeleitet.

Für SSR-Frameworks wie Nuxt.js muss das Ziel der Catch-All-Regel der primitive Compute-Typ sein. Das liegt daran, dass SSR-Anwendungen serverseitig gerenderte Seiten mit Routen haben, die zum Zeitpunkt der Erstellung nicht vorhersehbar sind. Zum Beispiel, wenn eine Nuxt.js Anwendung eine Seite hat, auf der `/blog/[slug]` sich ein dynamischer Routenparameter `[slug]` befindet. Das Ziel der Catch-All-Regel ist die einzige Möglichkeit, Anfragen an diese Seiten weiterzuleiten.

Im Gegensatz dazu können spezifische Pfadmuster verwendet werden, um auf Routen zu zielen, die zum Zeitpunkt der Erstellung bekannt sind. Stellt beispielsweise statische Nuxt.js Elemente aus dem `/_nuxt` Pfad bereit. Das bedeutet, dass auf den `/_nuxt/*` Pfad eine bestimmte Routing-Regel angewendet werden kann, die Anfragen an den primitiven Typ der statischen Objekte weiterleitet.

Routing öffentlicher Ordner

Die meisten SSR-Frameworks bieten die Möglichkeit, veränderbare statische Assets aus einem `public` Ordner bereitzustellen. Dateien wie `favicon.ico` und `robots.txt` werden normalerweise im `public` Ordner gespeichert und über die Stamm-URL der Anwendung bereitgestellt. Die `favicon.ico` Datei wird beispielsweise von bereitgestellt `https://example.com/favicon.ico`. Beachten Sie, dass es für diese Dateien kein vorhersehbares Pfadmuster gibt. Sie werden fast ausschließlich durch den Dateinamen bestimmt. Die einzige Möglichkeit, Dateien innerhalb des `public` Ordners als Ziel zu verwenden, besteht darin, die Catch-All-Route zu verwenden. Das Catch-All-Route-Ziel muss jedoch vom Typ Compute Primitive sein.

Wir empfehlen einen der folgenden Ansätze für die Verwaltung Ihres `public` Ordners.

1. Verwenden Sie ein Pfadmuster, um auf Anforderungspfade zu zielen, die Dateierweiterungen enthalten. Sie können es beispielsweise als Ziel für alle Anforderungspfade verwenden `/*.*`, die eine Dateierweiterung enthalten.

Beachten Sie, dass dieser Ansatz unzuverlässig sein kann. Wenn der `public` Ordner beispielsweise Dateien ohne Dateierweiterungen enthält, fallen sie nicht unter diese Regel. Ein weiteres Problem, das Sie bei diesem Ansatz beachten sollten, besteht darin, dass die Anwendung Seiten mit Punkten im Namen enthalten kann. Beispielsweise `/blog/2021/01/01/hello.world` wird die `/*.*` Regel als Zielseite für eine Seite verwendet. Dies ist nicht ideal, da es sich bei der Seite nicht um ein statisches Asset handelt. Sie können dieser Regel jedoch ein Fallback-Ziel hinzufügen, um sicherzustellen, dass bei einem 404-Fehler vom statischen primitiven Typ die Anfrage auf den primitiven Compute-Typ zurückfällt.

```
{
  "path": "/*.*",
  "target": {
    "kind": "Static"
  },
  "fallback": {
    "kind": "Compute",
    "src": "default"
  }
}
```

2. Identifizieren Sie bei der Erstellung die Dateien im `public` Ordner und geben Sie für jede Datei eine Routing-Regel aus. Dieser Ansatz ist nicht skalierbar, da die Bereitstellungsspezifikation eine Obergrenze von 25 Regeln vorschreibt.

```
{
  "path": "/favicon.ico",
  "target": {
    "kind": "Static"
  }
},
{
  "path": "/robots.txt",
  "target": {
    "kind": "Static"
  }
}
```

```
}
```

3. Empfehlen Sie Ihren Framework-Benutzern, alle veränderbaren statischen Assets in einem Unterordner innerhalb des `public` Ordners zu speichern.

Im folgenden Beispiel kann der Benutzer alle veränderlichen statischen Assets im Ordner `public/assets` speichern. Anschließend `/assets/*` kann eine Routing-Regel mit dem Pfadmuster verwendet werden, um auf alle veränderbaren statischen Assets innerhalb des `public/assets` Ordners abzielen.

```
{
  "path": "/assets/*",
  "target": {
    "kind": "Static"
  }
}
```

4. Geben Sie einen statischen Fallback für die Catch-All-Route an. Dieser Ansatz hat Nachteile, die im nächsten Abschnitt ausführlicher beschrieben werden. [Catch-All-Fallback-Routing](#)

Catch-All-Fallback-Routing

Für SSR-Frameworks Nuxt.js, bei denen beispielsweise eine Catch-All-Route für das Compute-Primitiv-Typziel angegeben ist, könnten Framework-Autoren erwägen, einen statischen Fallback für die Catch-All-Route anzugeben, um das Ordnerrouting-Problem zu lösen. Diese Art von Routing-Regel verstößt jedoch gegen serverseitig gerenderte 404-Seiten. Wenn der Endbenutzer beispielsweise eine Seite besucht, die nicht existiert, rendert die Anwendung eine 404-Seite mit dem Statuscode 404. Wenn die Catch-All-Route jedoch einen statischen Fallback hat, wird die 404-Seite nicht gerendert. Stattdessen fällt die Anfrage auf den statischen primitiven Typ zurück und endet immer noch mit einem 404-Statuscode, aber die 404-Seite wird nicht gerendert.

```
{
  "path": "/*",
  "target": {
    "kind": "Compute",
    "src": "default"
  },
  "fallback": {
    "kind": "Static"
  }
}
```

```
}
```

Basispfad-Routing

Von Frameworks, die die Möglichkeit bieten, den Basispfad der Anwendung zu ändern, wird erwartet, dass sie den Basispfad den statischen Assets innerhalb des `.amplify-hosting/static` Verzeichnisses voranstellen. Wenn der Basispfad beispielsweise lautet `/folder1/folder2`, dann lautet die Build-Ausgabe für ein statisches Asset namens `main.css`. `.amplify-hosting/static/folder1/folder2/main.css`

Das bedeutet, dass auch die Routing-Regeln aktualisiert werden müssen, um den Basispfad widerzuspiegeln. Wenn der Basispfad beispielsweise lautet `/folder1/folder2`, dann sieht die Routing-Regel für die statischen Objekte im `public` Ordner wie folgt aus.

```
{
  "path": "/folder1/folder2/*.*",
  "target": {
    "kind": "Static"
  }
}
```

In ähnlicher Weise muss auch serverseitigen Routen der Basispfad vorangestellt werden. Wenn der Basispfad beispielsweise lautet `/folder1/folder2`, dann sieht die Routing-Regel für die `/api` Route wie folgt aus.

```
{
  "path": "/folder1/folder2/api/*",
  "target": {
    "kind": "Compute",
    "src": "default"
  }
}
```

Der Basispfad sollte jedoch nicht der Sammelroute vorangestellt werden. Wenn der Basispfad beispielsweise lautet `/folder1/folder2`, bleibt die Catch-All-Route wie folgt.

```
{
  "path": "/*",
  "target": {
    "kind": "Compute",
    "src": "default"
  }
}
```

```
}  
}
```

Beispiele für Routen in Nuxt.js

Im Folgenden finden Sie eine `deploy-manifest.json` Beispieldatei für eine Nuxt-Anwendung, die zeigt, wie Routing-Regeln angegeben werden.

```
{  
  "version": 1,  
  "routes": [  
    {  
      "path": "/_nuxt/image",  
      "target": {  
        "kind": "ImageOptimization",  
        "cacheControl": "public, max-age=3600, immutable"  
      }  
    },  
    {  
      "path": "/_nuxt/builds/meta/*",  
      "target": {  
        "cacheControl": "public, max-age=31536000, immutable",  
        "kind": "Static"  
      }  
    },  
    {  
      "path": "/_nuxt/builds/*",  
      "target": {  
        "cacheControl": "public, max-age=1, immutable",  
        "kind": "Static"  
      }  
    },  
    {  
      "path": "/_nuxt/*",  
      "target": {  
        "cacheControl": "public, max-age=31536000, immutable",  
        "kind": "Static"  
      }  
    },  
    {  
      "path": "/*.*",  
      "target": {  
        "kind": "Static"  
      }  
    }  
  ]  
}
```

```
    },
    "fallback": {
      "kind": "Compute",
      "src": "default"
    }
  },
  {
    "path": "/*",
    "target": {
      "kind": "Compute",
      "src": "default"
    }
  }
],
"computeResources": [
  {
    "name": "default",
    "entrypoint": "server.js",
    "runtime": "nodejs18.x"
  }
],
"framework": {
  "name": "nuxt",
  "version": "3.8.1"
}
}
```

Im Folgenden finden Sie eine `deploy-manifest.json` Beispieldatei für Nuxt, die zeigt, wie Routing-Regeln einschließlich Basispfaden angegeben werden.

```
{
  "version": 1,
  "routes": [
    {
      "path": "/base-path/_nuxt/image",
      "target": {
        "kind": "ImageOptimization",
        "cacheControl": "public, max-age=3600, immutable"
      }
    },
    {
      "path": "/base-path/_nuxt/builds/meta/*",
      "target": {
```

```
    "cacheControl": "public, max-age=31536000, immutable",
    "kind": "Static"
  }
},
{
  "path": "/base-path/_nuxt/builds/*",
  "target": {
    "cacheControl": "public, max-age=1, immutable",
    "kind": "Static"
  }
},
{
  "path": "/base-path/_nuxt/*",
  "target": {
    "cacheControl": "public, max-age=31536000, immutable",
    "kind": "Static"
  }
},
{
  "path": "/base-path/*.*",
  "target": {
    "kind": "Static"
  },
  "fallback": {
    "kind": "Compute",
    "src": "default"
  }
},
{
  "path": "/*",
  "target": {
    "kind": "Compute",
    "src": "default"
  }
}
],
"computeResources": [
  {
    "name": "default",
    "entrypoint": "server.js",
    "runtime": "nodejs18.x"
  }
],
"framework": {
```

```
"name": "nuxt",
"version": "3.8.1"
}
}
```

Weitere Informationen zur Verwendung des `routes` Attributs finden Sie unter [Verwenden des Route-Attributs](#).

Bereitstellen eines Express-Servers mithilfe des Deployment-Manifests

In diesem Beispiel wird erklärt, wie ein einfacher Express-Server mithilfe der Amplify Hosting-Bereitstellungsspezifikation bereitgestellt wird. Sie können das bereitgestellte Bereitstellungsmanifest nutzen, um Routing, Rechenressourcen und andere Konfigurationen anzugeben.

Richten Sie vor der Bereitstellung auf Amplify Hosting lokal einen Express-Server ein

1. Erstellen Sie ein neues Verzeichnis für Ihr Projekt und installieren Sie Express und Typescript.

```
mkdir express-app
cd express-app

# The following command will prompt you for information about your project
npm init

# Install express, typescript and types
npm install express --save
npm install typescript ts-node @types/node @types/express --save-dev
```

2. Fügen Sie dem Stammverzeichnis Ihres Projekts eine `tsconfig.json` Datei mit dem folgenden Inhalt hinzu.

```
{
  "compilerOptions": {
    "target": "es6",
    "module": "commonjs",
    "outDir": "./dist",
    "strict": true,
    "esModuleInterop": true,
    "skipLibCheck": true,
    "forceConsistentCasingInFileNames": true
  },
}
```

```
"include": ["src/**/*.ts"],
"exclude": ["node_modules"]
}
```

- Erstellen Sie ein Verzeichnis mit dem Namen `src` in Ihrem Projektstamm.
- Erstellen Sie eine `index.ts` Datei im `src` Verzeichnis. Dies ist der Einstiegspunkt zu der Anwendung, die einen Express-Server startet. Der Server sollte so konfiguriert sein, dass er auf Port 3000 lauscht.

```
// src/index.ts
import express from 'express';

const app: express.Application = express();
const port = 3000;

app.use(express.text());

app.listen(port, () => {
  console.log(`server is listening on ${port}`);
});

// Homepage
app.get('/', (req: express.Request, res: express.Response) => {
  res.status(200).send("Hello World!");
});

// GET
app.get('/get', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-get-header", "get-header-value").send("get-response-from-compute");
});

//POST
app.post('/post', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-post-header", "post-header-value").send(req.body.toString());
});

//PUT
app.put('/put', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-put-header", "put-header-value").send(req.body.toString());
});
```



```
//PATCH
app.patch('/patch', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-patch-header", "patch-header-
value").send(req.body.toString());
});

// Delete
app.delete('/delete', (req: express.Request, res: express.Response) => {
  res.status(200).header("x-delete-header", "delete-header-value").send();
});
```

5. Fügen Sie Ihrer `package.json` Datei die folgenden Skripts hinzu.

```
"scripts": {
  "start": "ts-node src/index.ts",
  "build": "tsc",
  "serve": "node dist/index.js"
}
```

6. Erstellen Sie ein Verzeichnis mit dem Namen `public` im Stammverzeichnis Ihres Projekts. Erstellen Sie dann eine Datei `hello-world.txt` mit dem folgenden Namen.

```
Hello world!
```

7. Fügen Sie Ihrem Projektstamm eine `.gitignore` Datei mit dem folgenden Inhalt hinzu.

```
.amplify-hosting
dist
node_modules
```

Richten Sie das Amplify-Bereitstellungsmanifest ein

1. Erstellen Sie eine Datei mit dem Namen `deploy-manifest.json` im Stammverzeichnis Ihres Projekts.
2. Kopieren Sie das folgende Manifest und fügen Sie es in Ihre `deploy-manifest.json` Datei ein.

```
{
  "version": 1,
  "framework": { "name": "express", "version": "4.18.2" },
```

```
"imageSettings": {
  "sizes": [
    100,
    200,
    1920
  ],
  "domains": [],
  "remotePatterns": [],
  "formats": [],
  "minimumCacheTTL": 60,
  "dangerouslyAllowSVG": false
},
"routes": [
  {
    "path": "/_amplify/image",
    "target": {
      "kind": "ImageOptimization",
      "cacheControl": "public, max-age=3600, immutable"
    }
  },
  {
    "path": "/*.*",
    "target": {
      "kind": "Static",
      "cacheControl": "public, max-age=2"
    },
    "fallback": {
      "kind": "Compute",
      "src": "default"
    }
  },
  {
    "path": "/*",
    "target": {
      "kind": "Compute",
      "src": "default"
    }
  }
],
"computeResources": [
  {
    "name": "default",
    "runtime": "nodejs18.x",
    "entrypoint": "index.js"
  }
]
```

```
    }  
  ]  
}
```

Das Manifest beschreibt, wie Amplify Hosting die Bereitstellung Ihrer Anwendung handhaben sollte. Die primären Einstellungen sind die folgenden.

- **Version** — Gibt die Version der Bereitstellungsspezifikation an, die Sie verwenden.
- **Framework** — Passen Sie dies an, um Ihr Express Server-Setup zu spezifizieren.
- **ImageSettings** — Dieser Abschnitt ist für einen Express Server optional, sofern Sie sich nicht mit der Bildoptimierung befassen.
- **Routen** — Diese sind entscheidend, um den Verkehr in die richtigen Bereiche Ihrer App zu leiten. Die "kind": "Compute" Route leitet den Verkehr an Ihre Serverlogik weiter.
- **ComputeResources** — Verwenden Sie diesen Abschnitt, um die Laufzeit und den Express Einstiegspunkt Ihres Servers anzugeben.

Als Nächstes richten Sie ein Post-Build-Skript ein, das die erstellten Anwendungsartefakte in das Bereitstellungspaket verschiebt. `.amplify-hosting` Die Verzeichnisstruktur entspricht der Amplify Hosting-Bereitstellungsspezifikation.

Richten Sie das Post-Build-Skript ein

1. Erstellen Sie ein Verzeichnis mit dem Namen `bin` in Ihrem Projektstamm.
2. Erstellen Sie eine Datei mit dem Namen `postbuild.sh` im `bin` Verzeichnis. Fügen Sie der Datei `postbuild.sh` die folgenden Inhalte hinzu.

```
#!/bin/bash  
  
rm -rf ./amplify-hosting  
  
mkdir -p ./amplify-hosting/compute  
  
cp -r ./dist ./amplify-hosting/compute/default  
cp -r ./node_modules ./amplify-hosting/compute/default/node_modules  
  
cp -r public ./amplify-hosting/static  
  
cp deploy-manifest.json ./amplify-hosting/deploy-manifest.json
```

3. Fügen Sie Ihrer `package.json` Datei ein `postbuild` Skript hinzu. Die Datei sollte wie folgt aussehen.

```
"scripts": {
  "start": "ts-node src/index.ts",
  "build": "tsc",
  "serve": "node dist/index.js",
  "postbuild": "chmod +x bin/postbuild.sh && ./bin/postbuild.sh"
}
```

4. Führen Sie den folgenden Befehl aus, um Ihre Anwendung zu erstellen.

```
npm run build
```

5. (Optional) Passen Sie Ihre Routen für Express an. Sie können die Routen in Ihrem Bereitstellungsmanifest an Ihren Express-Server anpassen. Wenn Sie beispielsweise keine statischen Assets im `public` Verzeichnis haben, benötigen Sie möglicherweise nur die Catch-All-Route, die zu `"path": "/*"` Compute führt. Dies hängt von der Einrichtung Ihres Servers ab.

Ihre endgültige Verzeichnisstruktur sollte wie folgt aussehen.

```
express-app/
### .amplify-hosting/
#   ### compute/
#   #   ### default/
#   #       ### node_modules/
#   #       ### index.js
#   ### static/
#   #   ### hello.txt
#   ### deploy-manifest.json
### bin/
#   ### .amplify-hosting/
#   #   ### compute/
#   #   #   ### default/
#   #   #   ### static/
#   ### postbuild.sh*
### dist/
#   ### index.js
### node_modules/
### public/
```

```
#   ### hello.txt
### src/
#   ### index.ts
### deploy-manifest.json
### package.json
### package-lock.json
### tsconfig.json
```

Stellen Sie Ihren Server bereit

1. Pushen Sie Ihren Code in Ihr Git-Repository und stellen Sie Ihre App dann auf Amplify Hosting bereit.
2. Aktualisieren Sie Ihre Build-Einstellungen `.amplify-hosting` so, dass `baseDirectory` sie auf Folgendes verweisen. Während des Builds erkennt Amplify die Manifest-Datei im `.amplify-hosting` Verzeichnis und stellt Ihren Express-Server wie konfiguriert bereit.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - nvm use 18
        - npm install
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: .amplify-hosting
    files:
      - '**/*'
```

3. Um zu überprüfen, ob Ihre Bereitstellung erfolgreich war und Ihr Server ordnungsgemäß läuft, besuchen Sie Ihre App unter der von Amplify Hosting bereitgestellten Standard-URL.

Bildoptimierung für SSR-Apps

Amplify Hosting bietet eine integrierte Bildoptimierungsfunktion, die alle SSR-Apps unterstützt. Mit der Bildoptimierung von Amplify können Sie qualitativ hochwertige Bilder im richtigen Format, der richtigen Größe und Auflösung für das Gerät, das auf sie zugreift, liefern und gleichzeitig die kleinstmögliche Dateigröße beibehalten.

Derzeit können Sie entweder die Bildkomponente Next.js verwenden, um Bilder bei Bedarf zu optimieren, oder Sie können einen benutzerdefinierten Bildlader implementieren. Wenn Sie Next.js 13 oder höher verwenden, müssen Sie keine weiteren Maßnahmen ergreifen, um die Bildoptimierungsfunktion von Amplify zu verwenden. Wenn Sie einen benutzerdefinierten Loader implementieren, finden Sie weitere Informationen unter [Verwenden eines benutzerdefinierten Image-Loaders](#).

Verwenden eines benutzerdefinierten Image-Loaders

Wenn Sie einen benutzerdefinierten Bildlader verwenden, erkennt Amplify den Loader in der `next.config.js` Datei Ihrer Anwendung und nutzt die integrierte Bildoptimierungsfunktion nicht. Weitere Informationen zu den benutzerdefinierten Loadern, die Next.js unterstützt, finden Sie in der [Bildokumentation zu Next.js](#).

Integration der Bildoptimierung für Framework-Autoren

Framework-Autoren können die Bildoptimierungsfunktion von Amplify mithilfe der Amplify Hosting-Bereitstellungsspezifikation integrieren. Um die Bildoptimierung zu aktivieren, muss Ihr Bereitstellungsmanifest eine Routing-Regel enthalten, die auf den Bildoptimierungsdienst abzielt. Das folgende Beispiel zeigt, wie die Routingregel konfiguriert wird.

```
// .amplify-hosting/deploy-manifest.json

{
  "routes": [
    {
      "path": "/images/*",
      "target": {
        "kind": "ImageOptimization",
        "cacheControl": "public, max-age=31536000, immutable"
      }
    }
  ]
}
```

Weitere Informationen zur Konfiguration der Einstellungen für die Bildoptimierung mithilfe der Bereitstellungsspezifikation finden Sie unter [Spezifikation für die Bereitstellung von Amplify Hosting](#).

Grundlegendes zur Bildoptimierungs-API

Die Bildoptimierung kann zur Laufzeit über die Domain-URL einer Amplify-App auf dem in der Routing-Regel definierten Pfad aufgerufen werden.

```
GET https://{appDomainName}/{path}?{queryParams}
```

Bei der Bildoptimierung gelten die folgenden Regeln für Bilder.

- Amplify kann die Formate GIF, APNG und SVG nicht optimieren oder in ein anderes Format konvertieren.
- SVG-Bilder werden nur bereitgestellt, wenn die `dangerouslyAllowSVG` Einstellung aktiviert ist.
- Die Breite oder Höhe von Quellbildern darf 11 MB oder 9.000 Pixel nicht überschreiten.
- Die Größenbeschränkung für ein optimiertes Bild beträgt 4 MB.
- HTTP oder HTTPS ist das einzige Protokoll, das für die Beschaffung von Bildern mit Remote-URLs unterstützt wird.

HTTP-Header

Der HTTP-Header der Accept-Anfrage wird verwendet, um die Bildformate, ausgedrückt als MIME-Typen, anzugeben, die vom Client (normalerweise ein Webbrowser) zugelassen werden. Der Bildoptimierungsdienst versucht, das Bild in das angegebene Format zu konvertieren. Der für diesen Header angegebene Wert hat eine höhere Priorität als der Formatabfrageparameter. Ein gültiger Wert für den Accept-Header ist beispielsweise `image/png`, `image/webp`, `*/*`. Die im Amplify-Bereitstellungsmanifest angegebene Formateinstellung beschränkt die Formate auf die in der Liste aufgeführten. Selbst wenn der Accept-Header nach einem bestimmten Format fragt, wird es ignoriert, wenn das Format nicht in der Zulassungsliste enthalten ist.

URI-Anfrageparameter

In der folgenden Tabelle werden die URI-Anforderungsparameter für die Bildoptimierung beschrieben.

Abfrageparameter	Typ	Erforderlich	Beschreibung	Beispiel
URL	String	Ja	Ein relativer Pfad oder eine absolute URL zum Quellbild. Für eine Remote-URL werden die Protokolle http und https unterstützt. Der Wert muss URL-codiert sein.	?url=http%3A%2F%2Fwww.example.com%2Fbuffalo.png
width	Zahl	Ja	Die Breite des optimierten Bilds in Pixeln.	?width=800
height	Zahl	Nein	Die Höhe des optimierten Bilds in Pixeln. Wenn nicht angegeben, wird das Bild auto an die Breite angepasst.	?height=600
fit	Aufzählungswerte: cover, contain, inside, outside	Nein	Wie das Bild an die angegebene Breite und Höhe angepasst wird.	?width=800&height=600&fit=cover
position	Aufzählungswerte: center, top, bottom, left, right	Nein	Eine Position, die verwendet werden soll,	?fit=contain&position=centre

Abfrageparameter	Typ	Erforderlich	Beschreibung	Beispiel
	right bottom left		wenn fit cover oder contain ist.	
trim	Zahl	Nein	Schneidet Pixel an allen Kanten ab, die Werte enthalten, die der angegebenen Hintergru ndfarbe des Pixels oben links ähneln.	?trim=50
erweitern	Object	Nein	Fügt den Bildrändern Pixel hinzu, wobei die Farbe verwendet wird, die von den Pixeln am nächsten Rand abgeleitet wurde. Beim Format gibt jeder Wert die Anzahl der Pixel {top}_{ri ght}_{bot tom}_{lef t} an, die hinzugefügt werden sollen.	?extend=1 0_0_5_0

Abfrageparameter	Typ	Erforderlich	Beschreibung	Beispiel
extract	Object	Nein	Schneidet das Bild auf das angegebene Rechteck zu, das durch oben, links, Breite und Höhe begrenzt ist. Das Format ist {left} _ {top} _ {width} _ {right}, wobei jeder Wert die Anzahl der Pixel ist, die zugeschnitten werden sollen.	?extract=10_0_5_0
Format	String	Nein	Das gewünschte Ausgabeformat für das optimierte Bild.	?format=webp
quality	Zahl	Nein	Die Qualität des Bildes, von 1 bis 100. Wird nur bei der Konvertierung des Bildformats verwendet.	?quality=50
rotate	Zahl	Nein	Dreht das Bild um den angegebenen Winkel in Grad.	?rotate=45

Abfrageparameter	Typ	Erforderlich	Beschreibung	Beispiel
Flip	Boolesch	Nein	Spiegelt das Bild vertikal (von oben nach unten) auf der X-Achse. Dies geschieht immer vor der Drehung, falls vorhanden.	?flip
Flop	Boolesch	Nein	Spiegelt das Bild horizontal (links-rechts) auf der Y-Achse. Dies geschieht immer vor der Drehung, falls vorhanden.	?flop
schärfen	Zahl	Nein	Durch das Schärfen wird die Definition der Kanten im Bild verbessert. Gültige Werte liegen zwischen 0,000001 und 10.	?sharpen=1
median	Zahl	Nein	Wendet einen Medianfilter an. Dadurch werden Bildrauschen entfernt oder die Kanten eines Bilds geglättet.	?sharpen=3

Abfrageparameter	Typ	Erforderlich	Beschreibung	Beispiel
verwischen	Zahl	Nein	Wendet eine Gaußsche Unschärfe mit dem angegebenen Sigma an. Gültige Werte liegen zwischen 0,3 und 1.000.	?blur=20
gamma	Zahl	Nein	Wendet eine Gammakorrektur an, um die wahrgenommene Helligkeit eines Bildes mit geänderter Größe zu verbessern. Der Wert muss zwischen 1,0 und 3,0 liegen.	?gamma=1
negieren	Boolesch	Nein	Kehrt die Farben des Bildes um.	?negate
normalisieren	Boolesch	Nein	Verbessert den Bildkontrast, indem die Luminanz so ausgedehnt wird, dass ein voller Dynamikbereich abgedeckt wird.	?normalize

Abfrageparameter	Typ	Erforderlich	Beschreibung	Beispiel
threshold	Zahl	Nein	Ersetzt jedes Pixel im Bild durch ein schwarzes Pixel, wenn dessen Intensität unter dem angegebenen Schwellenwert liegt. Oder mit einem weißen Pixel, wenn es über dem Schwellenwert liegt. Gültige Werte liegen zwischen 0 und 255.	?threshold=155
Farbton	String	Nein	Färbt das Bild mit dem bereitgestellten RGB-Wert unter Beibehaltung der Bildeuchtdichte.	?tint=#7743CE
Graustufen	Boolesch	Nein	Wandelt das Bild in Graustufen (Schwarzweiß) um.	?grayscale

Statuscodes für Antworten

In der folgenden Liste werden die Antwortstatuscodes für die Bildoptimierung beschrieben.

Erfolg — HTTP-Statuscode 200

Die Anfrage wurde erfolgreich erfüllt.

BadRequest - HTTP-Statuscode 400

- Ein Eingabeabfrageparameter wurde falsch angegeben.
- Die Remote-URL ist in der `remotePatterns` Einstellung nicht als zulässig aufgeführt.
- Die Remote-URL wird nicht in ein Bild aufgelöst.
- Die angeforderte Breite oder Höhe sind in der `sizes` Einstellung nicht als zulässig aufgeführt.
- Das angeforderte Bild ist SVG, aber die `dangerouslyAllowSvg` Einstellung ist deaktiviert.

Nicht gefunden — HTTP-Statuscode 404

Das Quellbild wurde nicht gefunden.

Inhalt zu groß — HTTP-Statuscode 413

Entweder das Quellbild oder das optimierte Bild überschreiten die maximal zulässige Größe in Byte.

Caching

Amplify Hosting speichert optimierte Bilder auf unserem CDN zwischen, sodass nachfolgende Anfragen an dasselbe Bild mit denselben Abfrageparametern vom Cache aus bedient werden. Die Gültigkeitsdauer (TTL) des Caches wird durch den Header gesteuert. `Cache-Control` In der folgenden Liste werden Ihre Optionen für die Angabe des `Cache-Control` Headers beschrieben.

- Verwenden des `Cache-Control` Schlüssels in der Routing-Regel, die auf die Bildoptimierung abzielt.
- Verwenden von benutzerdefinierten Headern, die in der Amplify-App definiert sind.
- Bei Remote-Bildern wird der vom Remote-Image zurückgegebene `Cache-Control` Header berücksichtigt.

Die in den Bildoptimierungseinstellungen `minimumCacheTTL` angegebenen Werte definieren die Untergrenze der `Cache-Control max-age` Direktive. Wenn beispielsweise eine Remote-Bild-URL mit `antwortetCache-Control s-max-age=10`, der Wert jedoch 60 `minimumCacheTTL` ist, dann wird 60 verwendet.

Unterstützung der Version Node.js für Apps der Version Next.js

Wenn Amplify eine Next.js Compute-App erstellt und bereitstellt, verwendet es die Node.js Laufzeitversion, die der Hauptversion entspricht Node.js, die zum Erstellen der App verwendet wurde.

Sie können die Node.js Version angeben, die in der Funktion zum Überschreiben von Live-Paketen in der Amplify-Konsole verwendet werden soll. Weitere Informationen zur Konfiguration von Live-Paket-Updates finden Sie unter [Live-Paket-Updates](#). Sie können die Node.js Version auch mithilfe anderer Mechanismen angeben, z. B. mit nvm Befehlen. Wenn Sie keine Version angeben, verwendet Amplify standardmäßig die aktuelle Version, die vom Amplify-Build-Container verwendet wird.

Fehlerbehebung bei SSR-Bereitstellungen

Wenn bei der Bereitstellung einer SSR-App mit Amplify Hosting Compute unerwartete Probleme auftreten, lesen Sie die folgenden Themen zur Fehlerbehebung. Wenn Sie hier keine Lösung für Ihr Problem finden, lesen Sie den [Leitfaden zur Fehlerbehebung bei SSR-Webcomputern im Repository Amplify Hosting GitHub Issues](#).

Themen

- [Sie verwenden einen Framework-Adapter](#)
- [Edge-API-Routen führen dazu, dass Ihr Build Next.js fehlschlägt](#)
- [Die inkrementelle statische Regeneration auf Abruf funktioniert für Ihre App nicht](#)
- [Die Build-Ausgabe Ihrer App überschreitet die maximal zulässige Größe](#)
- [Ihr Build schlägt mit einem Fehler fehl, dass nicht genügend Arbeitsspeicher zur Verfügung steht](#)
- [Die Größe der HTTP-Antwort ist zu groß](#)

Sie verwenden einen Framework-Adapter

Wenn Sie Probleme bei der Bereitstellung einer SSR-App haben, die einen Framework-Adapter verwendet, finden Sie weitere Informationen unter [Amplify Sie die Unterstützung für SSR-Frameworks](#).

Edge-API-Routen führen dazu, dass Ihr Build Next.js fehlschlägt

Derzeit unterstützt Amplify keine Next.js Edge-API-Routen. Sie müssen beim Hosten Ihrer App mit Amplify APIs und Middleware verwenden, die keine Edge-Komponenten sind.

Die inkrementelle statische Regeneration auf Abruf funktioniert für Ihre App nicht

Ab Version 12.2.0 unterstützt Next.js die inkrementelle statische Regeneration (ISR), um den Cache von Next.js für eine bestimmte Seite manuell zu löschen. Amplify unterstützt derzeit jedoch kein On-Demand-ISR. Wenn Ihre App die On-Demand-Revalidierung von Next.js verwendet, funktioniert diese Funktion nicht, wenn Sie Ihre App auf Amplify bereitstellen.

Die Build-Ausgabe Ihrer App überschreitet die maximal zulässige Größe

Derzeit beträgt die maximale Build-Ausgabegröße, die Amplify für SSR-Apps unterstützt, 220 MB. Wenn Sie eine Fehlermeldung erhalten, die besagt, dass die Größe der Build-Ausgabe Ihrer App die maximal zulässige Größe überschreitet, müssen Sie Maßnahmen ergreifen, um sie zu reduzieren.

Um die Größe der Build-Ausgabe einer App zu reduzieren, können Sie die Build-Artefakte der App überprüfen und alle großen Abhängigkeiten identifizieren, die aktualisiert oder entfernt werden müssen. Laden Sie zunächst die Build-Artefakte auf Ihren lokalen Computer herunter. Überprüfen Sie dann die Größe der Verzeichnisse. Das `node_modules` Verzeichnis kann beispielsweise Binärdateien wie die Server-Runtime-Dateien von Next.js enthalten `@swc` und auf `@esbuild` die von diesen verwiesen wird. Da diese Binärdateien in der Laufzeit nicht erforderlich sind, können Sie sie nach dem Build löschen.

Verwenden Sie die folgenden Anweisungen, um die Build-Ausgabe einer App herunterzuladen und die Größe der Verzeichnisse mithilfe der AWS Command Line Interface (CLI) zu überprüfen.

Um die Build-Ausgabe für eine Next.js App herunterzuladen und zu überprüfen

1. Öffnen Sie ein Terminalfenster und führen Sie den folgenden Befehl aus. Ändern Sie die App-ID, den Zweignamen und die Job-ID nach Ihren eigenen Angaben. Verwenden Sie für die Job-ID die Buildnummer für den fehlgeschlagenen Build, den Sie untersuchen.

```
aws amplify get-job --app-id abcd1234 --branch-name main --job-id 2
```

2. Suchen Sie in der Terminalausgabe im `stepName: "BUILD"` Abschnitt, nach der URL für vorsignierte Artefakte. `job steps` Die URL ist in der folgenden Beispielausgabe rot hervorgehoben.

```
"job": {  
  "summary": {
```



```

    "jobArn": "arn:aws:amplify:us-west-2:111122223333:apps/abcd1234/main/
jobs/0000000002",
    "jobId": "2",
    "commitId": "HEAD",
    "commitTime": "2024-02-08T21:54:42.398000+00:00",
    "startTime": "2024-02-08T21:54:42.674000+00:00",
    "status": "SUCCEED",
    "endTime": "2024-02-08T22:03:58.071000+00:00"
  },
  "steps": [
    {
      "stepName": "BUILD",
      "startTime": "2024-02-08T21:54:42.693000+00:00",
      "status": "SUCCEED",
      "endTime": "2024-02-08T22:03:30.897000+00:00",
      "logUrl": "https://aws-amplify-prod-us-west-2-artifacts.s3.us-
west-2.amazonaws.com/abcd1234/main/0000000002/BUILD/log.txt?X-Amz-Security-
Token=IQoJb3JpZ2luX2V...Example

```

3. Kopieren Sie die URL und fügen Sie sie in ein Browserfenster ein. Eine `artifacts.zip` Datei wird auf Ihren lokalen Computer heruntergeladen. Dies ist Ihre Build-Ausgabe.
4. Führen Sie den Befehl `du disk usage` aus, um die Größe der Verzeichnisse zu überprüfen. Der folgende Beispielfehler gibt die Größe der `static` Verzeichnisse `compute` und zurück.

```
du -csh compute static
```

Im Folgenden finden Sie ein Beispiel für die Ausgabe mit Größeninformationen für die `static` Verzeichnisse `compute` und.

```

29M    compute
3.8M   static
33M    total

```

5. Öffnen Sie das `compute` Verzeichnis und suchen Sie den `node_modules` Ordner. Suchen Sie in Ihren Abhängigkeiten nach Dateien, die Sie aktualisieren oder entfernen können, um die Größe des Ordners zu verringern.
6. Wenn Ihre App Binärdateien enthält, die in der Runtime nicht benötigt werden, löschen Sie sie nach dem Build, indem Sie die folgenden Befehle zum Build-Abschnitt der `amplify.yml` App-Datei hinzufügen.

```
- rm -f node_modules/@swc/core-linux-x64-gnu/swc.linux-x64-gnu.node
- rm -f node_modules/@swc/core-linux-x64-musl/swc.linux-x64-musl.node
```

Im Folgenden finden Sie ein Beispiel für den Abschnitt mit Build-Befehlen in einer `amplify.yml` Datei, in dem diese Befehle nach der Ausführung eines Produktions-Builds hinzugefügt wurden.

```
frontend:
  phases:
    build:
      commands:
        -npm run build

        // After running a production build, delete the files
        - rm -f node_modules/@swc/core-linux-x64-gnu/swc.linux-x64-gnu.node
        - rm -f node_modules/@swc/core-linux-x64-musl/swc.linux-x64-musl.node
```

Ihr Build schlägt mit einem Fehler fehl, dass nicht genügend Arbeitsspeicher zur Verfügung steht

Next.js ermöglicht es Ihnen, Build-Artefakte zwischenspeichern, um die Leistung bei nachfolgenden Builds zu verbessern. Darüber hinaus komprimiert der AWS CodeBuild Container von Amplify diesen Cache und lädt ihn in Ihrem Namen auf Amazon S3 hoch, um die nachfolgende Build-Leistung zu verbessern. Dies könnte dazu führen, dass Ihr Build mit einem Fehler wegen unzureichenden Speichers fehlschlägt.

Führen Sie die folgenden Aktionen aus, um zu verhindern, dass Ihre App das Speicherlimit während der Erstellungsphase überschreitet. Entfernen Sie es zunächst `.next/cache/**/*` aus dem Abschnitt `cache.paths` Ihrer Build-Einstellungen. Als Nächstes entfernen Sie die `NODE_OPTIONS` Umgebungsvariable aus Ihrer Build-Einstellungsdatei. Stellen Sie stattdessen die `NODE_OPTIONS` Umgebungsvariable in der Amplify-Konsole ein, um das maximale Speicherlimit für den Knoten zu definieren. Weitere Informationen zum Einstellen von Umgebungsvariablen mit der Amplify-Konsole finden Sie unter [Festlegen von Umgebungsvariablen](#).

Nachdem Sie diese Änderungen vorgenommen haben, versuchen Sie es erneut mit Ihrem Build. Wenn dies erfolgreich ist, fügen Sie `.next/cache/**/*` dem Abschnitt `cache.paths` Ihrer Build-Einstellungsdatei etwas hinzu.

Weitere Informationen zur Cache-Konfiguration von Next.js zur Verbesserung der Build-Leistung finden Sie unter [AWS CodeBuild](#) auf der Website Next.js.

Die Größe der HTTP-Antwort ist zu groß

Derzeit beträgt die maximale Antwortgröße, die Amplify für Apps von Next.js 12 oder höher unterstützt, die die Web Compute-Plattform verwenden, 5,72 MB. Antworten, die dieses Limit überschreiten, geben 504 Fehler ohne Inhalt an die Clients zurück.

Amplify Sie die Unterstützung für Next.js

Amplify unterstützt die Bereitstellung und das Hosting von serverseitig gerenderten (SSR) Web-Apps, die mit Next.js erstellt wurden. Next.js ist ein React-Framework zum Erstellen von Fullstack-Webanwendungen. Sie können mit Next.js 14 erstellte Apps mit Funktionen wie Bildoptimierung und Middleware bereitstellen.

Entwickler können Next.js verwenden, um Static Site Generation (SSG) und SSR in einem einzigen Projekt zu kombinieren. SSG-Seiten werden zum Zeitpunkt der Erstellung vorgerendert, und SSR-Seiten werden zum Zeitpunkt der Anfrage vorgerendert.

Durch das Prerendern können die Leistung und die Suchmaschinenoptimierung verbessert werden. Da Next.js alle Seiten auf dem Server vorrendert, ist der HTML-Inhalt jeder Seite bereit, sobald er den Browser des Clients erreicht. Dieser Inhalt kann auch schneller geladen werden. Schnellere Ladezeiten verbessern die Erfahrung des Endbenutzers mit einer Website und wirken sich positiv auf das SEO-Ranking der Website aus. Das Prerendering verbessert auch die Suchmaschinenoptimierung, indem es Suchmaschinen-Bots ermöglicht, den HTML-Inhalt einer Website einfach zu finden und zu crawlen.

Next.js bietet integrierte Analyseunterstützung für die Messung verschiedener Leistungskennzahlen wie Time to First Byte (TTFB) und First Contentful Paint (FCP). Weitere Informationen zu Next.js finden Sie unter [Erste Schritte](#) auf der Website Next.js.

Unterstützung für die Funktion Next.js

Amplify Hosting Compute verwaltet vollständig das serverseitige Rendering (SSR) für Apps, die mit den Versionen 12, 13 und 14 von Next.js erstellt wurden. Wenn Sie vor der Veröffentlichung von Amplify Hosting Compute eine Next.js App für Amplify bereitgestellt haben, verwendet Ihre App den vorherigen SSR-Anbieter von Amplify, Classic (nur Next.js 11). Amplify Hosting Compute unterstützt

keine Apps, die mit Next.js Version 11 oder früher erstellt wurden. Wir empfehlen dringend, dass Sie Ihre Next.js 11-Apps zum Compute Managed SSR-Anbieter von Amplify Hosting migrieren.

In der folgenden Liste werden die spezifischen Funktionen beschrieben, die der Amplify Hosting-Compute-SSR-Anbieter unterstützt.

Unterstützte Features

- Serverseitig gerenderte Seiten (SSR)
- Statische Seiten
- API-Routen
- Dynamische Routen
- Erfasse alle Routen
- SSG (Statische Generierung)
- Inkrementelle statische Regeneration (ISR)
- Internationalisiertes (i18n) Unterpfad-Routing
- Internationalisiertes (i18n) Domain-Routing
- Middleware
- Umgebungsvariablen
- Bildoptimierung
- Next.js 13 App-Verzeichnis

Nicht unterstützte Funktionen

- Edge-API-Routen (Edge-Middleware wird nicht unterstützt)
- Inkrementelle statische Regeneration (ISR) auf Anfrage
- Internationalisierte (i18n) automatische Erkennung von Gebietsschemas
- Next.js streamen
- Middleware auf statischen Assets und optimierten Bildern ausführen

Next.js Bilder

Die maximale Ausgabegröße eines Bildes darf 4,3 MB nicht überschreiten. Sie können eine größere Bilddatei irgendwo speichern und sie mit der Image-Komponente Next.js skalieren und für ein Webp- oder AVIF-Format optimieren und sie dann in einer kleineren Größe bereitstellen.

Beachten Sie, dass in der Dokumentation zu Next.js empfohlen wird, das Sharp-Bildverarbeitungsmodul zu installieren, damit die Bildoptimierung in der Produktion ordnungsgemäß funktioniert. Dies ist jedoch für Amplify-Bereitstellungen nicht erforderlich. Amplify stellt Sharp automatisch für Sie bereit.

Preise für Next.js Apps

Bei der Bereitstellung Ihrer SSR-App Next.js 12 oder höher verwaltet Amplify Hosting Compute die Ressourcen, die für die Bereitstellung der SSR-App für Sie erforderlich sind. Informationen zu den Rechengebühren von Amplify Hosting finden Sie unter [AWS Amplify Preise](#).

Bereitstellung einer Next.js App mit Amplify

Standardmäßig stellt Amplify neue SSR-Apps mithilfe des Compute-Service von Amplify Hosting mit Unterstützung für Next.js 12, 13 und 14 bereit. Amplify Hosting Compute verwaltet die Ressourcen, die für die Bereitstellung einer SSR-App erforderlich sind, vollständig. SSR-Apps in Ihrem Amplify-Konto, die Sie vor dem 17. November 2022 bereitgestellt haben, verwenden den klassischen SSR-Anbieter (nur Next.js 11).

Wir empfehlen dringend, Apps mit Classic (nur Next.js 11) SSR zum Amplify Hosting Compute SSR-Anbieter zu migrieren. Amplify führt keine automatischen Migrationen für Sie durch. Sie müssen Ihre App manuell migrieren und dann einen neuen Build initiieren, um das Update abzuschließen. Anweisungen finden Sie unter [Migrieren einer Next.js 11-App auf Amplify Hosting Compute](#).

Verwenden Sie die folgenden Anweisungen, um eine neue App Next.js bereitzustellen.

So stellen Sie mithilfe des Amplify Hosting-Compute-SSR-Anbieters eine Next.js App für Amplify bereit

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie auf der Seite Alle Apps die Option Neue App erstellen aus.
3. Wählen Sie auf der Seite Start building with Amplify Ihren Git-Repository-Anbieter aus und klicken Sie dann auf Weiter.
4. Gehen Sie auf der Seite Repository-Zweig hinzufügen wie folgt vor:
 - a. Wählen Sie den Namen des Repositorys aus, zu dem Sie eine Verbindung herstellen möchten.
 - b. Wählen Sie den Namen des Repository-Zweigs aus, zu dem eine Verbindung hergestellt werden soll.

- c. Wählen Sie Weiter aus.
5. Die App erfordert eine IAM-Servicerolle, die Amplify übernimmt, wenn es in Ihrem Namen andere Dienste anruft. Sie können entweder Amplify Hosting Compute erlauben, automatisch eine Servicerolle für Sie zu erstellen, oder Sie können eine Rolle angeben, die Sie erstellt haben.
 - Um Amplify zu ermöglichen, automatisch eine Rolle zu erstellen und sie an Ihre App anzuhängen:
 - Wählen Sie Neue Servicerolle erstellen und verwenden.
 - Um eine zuvor erstellte Servicerolle anzuhängen:
 - a. Wählen Sie Bestehende Servicerolle verwenden aus.
 - b. Wählen Sie die zu verwendende Rolle aus der Liste aus.
6. Wählen Sie Weiter aus.
7. Wählen Sie auf der Seite „Überprüfen“ die Option „Speichern und bereitstellen“ aus.

Einstellungen für die Datei package.json

Wenn Sie eine Next.js App bereitstellen, überprüft Amplify das Build-Skript der App in der package.json Datei, um festzustellen, ob es sich bei der App um SSR oder SSG handelt.

Das Folgende ist ein Beispiel für das Build-Skript für eine Next.js SSR-App. Das Build-Skript "next build" gibt an, dass die App sowohl SSG- als auch SSR-Seiten unterstützt.

```
"scripts": {  
  "dev": "next dev",  
  "build": "next build",  
  "start": "next start"  
},
```

Im Folgenden finden Sie ein Beispiel für das Build-Skript für eine SSG-App von Next.js. Das Build-Skript "next build && next export" gibt an, dass die App nur SSG-Seiten unterstützt.

```
"scripts": {  
  "dev": "next dev",  
  "build": "next build && next export",  
  "start": "next start"  
},
```

Build-Einstellungen Amplify

Nachdem Sie die `package.json` Datei Ihrer App überprüft haben, um festzustellen, ob Sie eine SSG- oder SSR-App bereitstellen, überprüft Amplify die Build-Einstellungen für die App. Sie können die Build-Einstellungen in der Amplify-Konsole oder in einer `amplify.yml` Datei im Stammverzeichnis Ihres Repositorys speichern. Weitere Informationen finden Sie unter [Build-Einstellungen konfigurieren](#).

Wenn Amplify feststellt, dass Sie eine Next.js SSR-App bereitstellen und keine `amplify.yml` Datei vorhanden ist, generiert es eine Buildspez für die App und setzt auf `baseDirectory: .next`. Wenn Sie eine App bereitstellen, in der eine `amplify.yml` Datei vorhanden ist, überschreiben die Build-Einstellungen in der Datei alle Build-Einstellungen in der Konsole. Daher müssen Sie den Wert `.next` in der Datei manuell `baseDirectory` auf setzen.

Im Folgenden finden Sie ein Beispiel für die Build-Einstellungen für eine App, für die der Wert auf `baseDirectory: .next` eingestellt ist. Dies weist darauf hin, dass die Build-Artefakte für eine Next.js -App bestimmt sind, die SSG- und SSR-Seiten unterstützt.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: .next
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
```

Wenn Amplify feststellt, dass Sie eine SSG-App bereitstellen, generiert es eine Buildspez für die App und setzt sie auf `baseDirectory: out`. Wenn Sie eine App bereitstellen, in der eine `amplify.yml` Datei vorhanden ist, müssen Sie den Wert in der `baseDirectory` Datei manuell auf `out` setzen.

Im Folgenden finden Sie ein Beispiel für die Build-Einstellungen für eine App, bei der der Wert auf `baseDirectory` ist `out`. Dies weist darauf hin, dass die Build-Artefakte für eine Next.js -App bestimmt sind, die nur SSG-Seiten unterstützt.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: out
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
```

Migrieren einer Next.js 11-App auf Amplify Hosting Compute

Wenn Sie eine neue Next.js App bereitstellen, verwendet Amplify standardmäßig die neueste unterstützte Version von Next.js. Derzeit unterstützt der Compute SSR-Anbieter Amplify Hosting Next.js Version 14.


Die Amplify-Konsole erkennt Apps in Ihrem Konto, die vor der Veröffentlichung des Amplify Hosting-Rechendienstes mit voller Unterstützung für die Versionen 12, 13 und 14 von Next.js bereitgestellt wurden. Auf der Konsole wird ein Informationsbanner angezeigt, das Apps mit Filialen identifiziert, die über den vorherigen SSR-Anbieter von Amplify, Classic, bereitgestellt wurden (nur Next.js 11). Wir empfehlen dringend, dass Sie Ihre Apps zum Amplify Hosting-Compute-SSR-Anbieter migrieren.

Sie müssen die App und alle ihre Produktionszweige gleichzeitig manuell migrieren. Eine App kann nicht sowohl Classic (nur Next.js 11) als auch Next.js 12, 13 oder 14 Branches enthalten.

Verwenden Sie die folgenden Anweisungen, um eine App zum Amplify Hosting-Compute-SSR-Anbieter zu migrieren.

Um eine App zum Amplify Hosting-Compute-SSR-Anbieter zu migrieren

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App Next.js aus, die Sie migrieren möchten.

 Note

Bevor Sie eine App in der Amplify-Konsole migrieren, müssen Sie zunächst die `package.json`-Datei der App aktualisieren, um Next.js Version 12, 13 oder 14 zu verwenden.

3. Wählen Sie im Navigationsbereich App-Einstellungen, Allgemein aus.
4. Auf der Startseite der App zeigt die Konsole ein Banner an, wenn die App über Zweige verfügt, die über den SSR-Anbieter Classic (nur Next.js 11) bereitgestellt wurden. Wählen Sie auf dem Banner Migrieren aus.
5. Wählen Sie im Bestätigungsfenster für die Migration die drei Anweisungen aus und wählen Sie Migrieren aus.
6. Amplify erstellt Ihre App und stellt sie erneut bereit, um die Migration abzuschließen.

Eine SSR-Migration rückgängig machen

Wenn Sie eine Next.js App bereitstellen, erkennt Amplify Hosting die Einstellungen in Ihrer App und legt den internen Plattformwert für die App fest. Es gibt drei gültige Plattformwerte. Eine SSG-App ist auf den Plattformwert WEB eingestellt. Eine SSR-App, die Next.js Version 11 verwendet, ist auf den Plattformwert gesetzt. WEB_DYNAMISC Eine SSR-App Next.js 12 oder höher ist auf den Plattformwert eingestellt. WEB_COMPUTE

Wenn Sie eine App mithilfe der Anweisungen im vorherigen Abschnitt migrieren, ändert Amplify den Plattformwert Ihrer App von WEB_DYNAMISC auf WEB_COMPUTE. Nachdem die Migration zu Amplify Hosting Compute abgeschlossen ist, können Sie die Migration in der Konsole nicht mehr rückgängig machen. Um die Migration rückgängig zu machen, müssen Sie den verwenden, AWS Command Line Interface um die Plattform der App wieder zu ändern. WEB_DYNAMISC Öffnen Sie ein Terminalfenster und geben Sie den folgenden Befehl ein, um die App-ID und die Region mit Ihren eindeutigen Informationen zu aktualisieren.

```
aws amplify update-app --app-id abcd1234 --platform WEB_DYNAMISC --region us-west-2
```

SSR-Funktionalität zu einer statischen Next.js App hinzufügen

Sie können SSR-Funktionalität zu einer vorhandenen statischen (SSG) Next.js App hinzufügen, die mit Amplify bereitgestellt wird. Bevor Sie mit der Konvertierung Ihrer SSG-App in SSR beginnen, aktualisieren Sie die App so, dass sie Next.js Version 12, 13 oder 14 verwendet, und fügen Sie SSR-Funktionen hinzu. Anschließend müssen Sie die folgenden Schritte ausführen.

1. Verwenden Sie die AWS Command Line Interface , um den Plattfortmtyyp der App zu ändern.
2. Fügen Sie der App eine Servicerolle hinzu.
3. Aktualisieren Sie das Ausgabeverzeichnis in den Build-Einstellungen der App.
4. Aktualisieren Sie die `package.json` Datei der App, um anzugeben, dass die App SSR verwendet.

Aktualisiere die Plattform

Es gibt drei gültige Werte für den Plattfortmtyyp. Eine SSG-App ist auf den Plattfortmtyyp WEB eingestellt. Eine SSR-App, die Next.js Version 11 verwendet, ist auf den Plattfortmtyyp `WEB_DYNAMIC` eingestellt. Für Apps, die auf Next.js 12 oder höher mithilfe von SSR bereitgestellt werden, das von Amplify Hosting Compute verwaltet wird, ist der Plattfortmtyyp auf `WEB_COMPUTE` eingestellt.

Wenn Sie Ihre App als SSG-App bereitgestellt haben, hat Amplify den Plattfortmtyyp auf `WEB` gesetzt. Verwenden Sie die AWS CLI , um die Plattform für Ihre App zu ändern. Öffnen Sie ein Terminalfenster und geben Sie den folgenden Befehl ein. Aktualisieren Sie dabei den roten Text mit Ihrer eindeutigen App-ID und Region.

```
aws amplify update-app --app-id abcd1234 --platform WEB_COMPUTE --region us-west-2
```

Fügen Sie eine Servicerolle hinzu

Eine Servicerolle ist die AWS Identity and Access Management (IAM) -Rolle, die Amplify übernimmt, wenn es in Ihrem Namen andere Dienste anruft. Folgen Sie diesen Schritten, um einer SSG-App, die bereits mit Amplify bereitgestellt wurde, eine Servicerolle hinzuzufügen.

Um eine Servicerolle hinzuzufügen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wenn Sie in Ihrem Amplify-Konto noch keine Servicerolle erstellt haben, finden Sie Informationen unter [Hinzufügen einer Servicerolle](#), um diesen vorausgesetzten Schritt abzuschließen.

3. Wählen Sie die statische App Next.js aus, zu der Sie eine Servicerolle hinzufügen möchten.
4. Wählen Sie im Navigationsbereich App-Einstellungen, Allgemein aus.
5. Wählen Sie auf der Seite mit den App-Details die Option Bearbeiten
6. Wählen Sie unter Servicerolle den Namen einer vorhandenen Servicerolle oder den Namen der Servicerolle, die Sie in Schritt 2 erstellt haben.
7. Wählen Sie Speichern.

Aktualisieren Sie die Build-Einstellungen

Bevor Sie Ihre App erneut mit SSR-Funktionalität bereitstellen, müssen Sie die Build-Einstellungen für die App aktualisieren, um das Ausgabeverzeichnis auf festzulegen. `.next` Sie können die Build-Einstellungen in der Amplify-Konsole oder in einer in Ihrem Repo gespeicherten `amplify.yml` Datei bearbeiten. Weitere Informationen finden Sie unter [Build-Einstellungen konfigurieren](#).

Das Folgende ist ein Beispiel für die Build-Einstellungen für eine App, bei der auf eingestellt `baseDirectory` ist. `.next`

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: .next
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
```

Aktualisieren Sie die Datei package.json

Nachdem Sie eine Servicerolle hinzugefügt und die Build-Einstellungen aktualisiert haben, aktualisieren Sie die Datei der App. `package.json` Stellen Sie wie im folgenden Beispiel das Build-

Skript so ein, dass "next build" es angibt, dass die App Next.js sowohl SSG- als auch SSR-Seiten unterstützt.

```
"scripts": {  
  "dev": "next dev",  
  "build": "next build",  
  "start": "next start"  
},
```

Amplify erkennt die Änderung an der `package.json` Datei in Ihrem Repo und stellt die App mit SSR-Funktionalität erneut bereit.

Umgebungsvariablen für serverseitige Laufzeiten zugänglich machen

Amplify Hosting unterstützt das Hinzufügen von Umgebungsvariablen zu den Builds Ihrer Anwendung, indem Sie sie in der Projektkonfiguration in der Amplify-Konsole festlegen. Eine Serverkomponente von Next.js hat jedoch standardmäßig keinen Zugriff auf diese Umgebungsvariablen. Dieses Verhalten dient dazu, alle Geheimnisse zu schützen, die in Umgebungsvariablen gespeichert sind, die Ihre Anwendung während der Erstellungsphase verwendet.

Um bestimmte Umgebungsvariablen für Next.js zugänglich zu machen, können Sie die Amplify-Build-Spezifikationsdatei so ändern, dass sie in den Umgebungsdateien festgelegt werden, die Next.js erkennt. Dadurch kann Amplify diese Umgebungsvariablen laden, bevor die Anwendung erstellt wird. Das folgende Beispiel für eine Build-Spezifikation zeigt, wie Umgebungsvariablen im Abschnitt mit den Build-Befehlen hinzugefügt werden.

```
version: 1  
frontend:  
  phases:  
    preBuild:  
      commands:  
        - npm ci  
    build:  
      commands:  
        - env | grep -e DB_HOST -e DB_USER -e DB_PASS >> .env.production  
        - env | grep -e NEXT_PUBLIC_ >> .env.production  
        - npm run build  
  artifacts:  
    baseDirectory: .next
```

```
files:
  - '**/*'
cache:
  paths:
    - node_modules/**/*
    - .next/cache/**/*
```

In diesem Beispiel enthält der Abschnitt mit den Build-Befehlen zwei Befehle, die Umgebungsvariablen in die `.env.production` Datei schreiben, bevor der Build der Anwendung ausgeführt wird. Amplify Hosting ermöglicht Ihrer Anwendung den Zugriff auf diese Variablen, wenn die Anwendung Datenverkehr empfängt.

Die folgende Zeile aus dem Abschnitt mit den Build-Befehlen im vorherigen Beispiel zeigt, wie Sie eine bestimmte Variable aus der Build-Umgebung nehmen und sie der `.env.production` Datei hinzufügen.

```
- env | grep -e DB_HOST -e DB_USER -e DB_PASS >> .env.production
```

Wenn die Variablen in Ihrer Build-Umgebung existieren, enthält die `.env.production` Datei die folgenden Umgebungsvariablen.

```
DB_HOST=localhost
DB_USER=myuser
DB_PASS=myspassword
```

Die folgende Zeile aus dem Abschnitt mit den Build-Befehlen im vorherigen Beispiel zeigt, wie Sie der `.env.production` Datei eine Umgebungsvariable mit einem bestimmten Präfix hinzufügen. In diesem Beispiel `NEXT_PUBLIC_` werden alle Variablen mit dem Präfix hinzugefügt.

```
- env | grep -e NEXT_PUBLIC_ >> .env.production
```

Wenn in der Build-Umgebung mehrere Variablen mit dem `NEXT_PUBLIC_` Präfix existieren, sieht die `.env.production` Datei wie folgt aus.

```
NEXT_PUBLIC_ANALYTICS_ID=abcdefghijkl
NEXT_PUBLIC_GRAPHQL_ENDPOINT=uowelalsmlsadf
NEXT_PUBLIC_SEARCH_KEY=asdfiojslf
NEXT_PUBLIC_SEARCH_ENDPOINT=https://search-url
```

SSR-Umgebungsvariablen für Monorepos

Wenn Sie eine SSR-App in einem Monorepo bereitstellen und bestimmte Umgebungsvariablen für Next.js zugänglich machen möchten, müssen Sie der `.env.production` Datei Ihr Anwendungsstammverzeichnis voranstellen. Die folgende Beispiel-Build-Spezifikation für eine Next.js -App in einem Nx-Monorepo zeigt, wie Umgebungsvariablen im Abschnitt mit den Build-Befehlen hinzugefügt werden.

```
version: 1
applications:
  - frontend:
      phases:
        preBuild:
          commands:
            - npm ci
        build:
          commands:
            - env | grep -e DB_HOST -e DB_USER -e DB_PASS >> apps/app/.env.production
            - env | grep -e NEXT_PUBLIC_ >> apps/app/.env.production
            - npx nx build app
      artifacts:
        baseDirectory: dist/apps/app/.next
        files:
          - '**/*'
      cache:
        paths:
          - node_modules/**/*
      buildPath: /
      appRoot: apps/app
```

Die folgenden Zeilen aus dem Abschnitt mit den Build-Befehlen im vorherigen Beispiel zeigen, wie bestimmte Variablen aus der Build-Umgebung entnommen und zur `.env.production` Datei für eine App in einem Monorepo mit dem Anwendungsstamm hinzugefügt werden. `apps/app`

```
- env | grep -e DB_HOST -e DB_USER -e DB_PASS >> apps/app/.env.production
- env | grep -e NEXT_PUBLIC_ >> apps/app/.env.production
```

Bereitstellen einer Next.js -App in einem Monorepo

Amplify unterstützt Apps in generischen Monorepos sowie Apps in Monorepos, die mit `npm workspace`, `pnpm workspace`, `Yarn workspace`, `Nx` und `Turborepo` erstellt wurden. Wenn Sie

Ihre App bereitstellen, erkennt Amplify automatisch das von Ihnen verwendete Monorepo-Build-Framework. Amplify wendet automatisch Build-Einstellungen für Apps in einem NPM-Workspace, Yarn-Workspace oder Nx an. Beachten Sie, dass pnpm- und Turborepo-Apps eine zusätzliche Konfiguration erfordern. Weitere Informationen finden Sie unter [Monorepo-Build-Einstellungen](#).

Ein detailliertes Nx-Beispiel finden Sie im Blogbeitrag [Share code between Next.js apps with Nx on AWS Amplify Hosting](#).

Amazon CloudWatch Logs für SSR-Apps

Amplify sendet Informationen über Ihre Laufzeit von Next.js an Amazon CloudWatch Logs in Ihrem AWS-Konto. Wenn Sie eine SSR-App bereitstellen, benötigt die App eine IAM-Dienstrolle, die Amplify übernimmt, wenn es andere Dienste in Ihrem Namen aufruft. Sie können entweder Amplify Hosting Compute erlauben, automatisch eine Servicerolle für Sie zu erstellen, oder Sie können eine Rolle angeben, die Sie erstellt haben.

Wenn Sie Amplify erlauben, eine IAM-Rolle für Sie zu erstellen, verfügt die Rolle bereits über die Berechtigungen zum Erstellen CloudWatch von Protokollen. Wenn Sie Ihre eigene IAM-Rolle erstellen, müssen Sie Ihrer Richtlinie die folgenden Berechtigungen hinzufügen, damit Amplify auf Amazon CloudWatch Logs zugreifen kann.

```
logs:CreateLogStream
logs:CreateLogGroup
logs:DescribeLogGroups
logs:PutLogEvents
```

Weitere Informationen zu Servicerollen finden Sie unter [Eine Servicerolle hinzufügen](#).

Amplify Next.js 11-Unterstützung

Wenn Sie vor der Veröffentlichung von Amplify Hosting Compute am 17. November 2022 eine Next.js App für Amplify bereitgestellt haben, verwendet Ihre App den vorherigen SSR-Anbieter von Amplify, Classic (nur Next.js 11). Die Dokumentation in diesem Abschnitt gilt nur für Apps, die mit dem SSR-Anbieter Classic (Next.js 11 nur) bereitgestellt wurden.

Note

Wir empfehlen dringend, dass Sie Ihre Next.js 11-Apps zum Compute Managed SSR-Anbieter von Amplify Hosting migrieren. Weitere Informationen finden Sie unter [Migrieren einer Next.js 11-App auf Amplify Hosting Compute](#).

In der folgenden Liste werden die spezifischen Funktionen beschrieben, die der SSR-Anbieter Amplify Classic (nur Next.js 11) unterstützt.

Unterstützte Funktionen

- Serverseitig gerenderte Seiten (SSR)
- Statische Seiten
- API-Routen
- Dynamische Routen
- Erfasse alle Routen
- SSG (Statische Generierung)
- Inkrementelle statische Regeneration (ISR)
- Internationalisiertes (i18n) Unterpfad-Routing
- Umgebungsvariablen

Nicht unterstützte Funktionen

- Bildoptimierung
- Inkrementelle statische Regeneration (ISR) auf Anfrage
- Internationalisiertes (i18n) Domain-Routing
- Internationalisierte (i18n) automatische Erkennung von Gebietsschemas
- Middleware
- Edge-Middleware
- Edge-API-Routen

Preise für Next.js 11-Apps

Bei der Bereitstellung Ihrer Next.js 11 SSR-App erstellt Amplify zusätzliche Backend-Ressourcen in Ihrem AWS Konto, darunter:

- Ein Amazon Simple Storage Service (Amazon S3) -Bucket, der die Ressourcen für die statischen Ressourcen Ihrer App speichert. Informationen zu den Amazon S3 S3-Gebühren finden Sie unter [Amazon S3 S3-Preise](#).
- Eine CloudFront Amazon-Distribution zur Bereitstellung der App. Informationen zu CloudFront Gebühren finden Sie unter [CloudFront Amazon-Preise](#).
- Vier [Lambda @Edge -Funktionen](#) zur Anpassung der Inhalte, die bereitgestellt werden CloudFront .

AWS Identity and Access Management Berechtigungen für Next.js 11 SSR-Apps

Amplify benötigt AWS Identity and Access Management (IAM-) Berechtigungen, um eine SSR-App bereitzustellen. Ohne die erforderlichen Mindestberechtigungen erhalten Sie eine Fehlermeldung, wenn Sie versuchen, Ihre SSR-App bereitzustellen. Um Amplify die erforderlichen Berechtigungen zu gewähren, müssen Sie eine Servicerolle angeben.

Informationen zum Erstellen einer IAM-Dienstrolle, die Amplify übernimmt, wenn Sie andere Dienste in Ihrem Namen aufrufen, finden Sie unter [Eine Servicerolle hinzufügen](#) Diese Anweisungen zeigen, wie Sie eine Rolle erstellen, die die verwaltete Richtlinie anhängt. AdministratorAccess-Amplify

Die AdministratorAccess-Amplify verwaltete Richtlinie bietet Zugriff auf mehrere AWS Dienste, einschließlich IAM-Aktionen. Sie sollte als ebenso leistungsfähig wie die Richtlinie angesehen werden. AdministratorAccess Diese Richtlinie bietet mehr Berechtigungen, als für die Bereitstellung Ihrer SSR-App erforderlich sind.

Es wird empfohlen, die bewährte Methode der Gewährung der geringsten Rechte zu befolgen und die der Servicerolle gewährten Berechtigungen zu reduzieren. Anstatt Administratorzugriffsberechtigungen für Ihre Servicerolle zu gewähren, können Sie Ihre eigene, vom Kunden verwaltete IAM-Richtlinie erstellen, die nur die für die Bereitstellung Ihrer SSR-App erforderlichen Berechtigungen gewährt. Anweisungen [zur Erstellung einer vom Kunden verwalteten Richtlinie finden Sie unter Erstellen von IAM-Richtlinien](#) im IAM-Benutzerhandbuch.

Wenn Sie Ihre eigene Richtlinie erstellen, finden Sie in der folgenden Liste die Mindestberechtigungen, die für die Bereitstellung einer SSR-App erforderlich sind.

```
acm:DescribeCertificate
acm:ListCertificates
acm:RequestCertificate
cloudfront:CreateCloudFrontOriginAccessIdentity
cloudfront:CreateDistribution
cloudfront:CreateInvalidation
cloudfront:GetDistribution
cloudfront:GetDistributionConfig
cloudfront:ListCloudFrontOriginAccessIdentities
cloudfront:ListDistributions
cloudfront:ListDistributionsByLambdaFunction
cloudfront:ListDistributionsByWebACLId
cloudfront:ListFieldLevelEncryptionConfigs
cloudfront:ListFieldLevelEncryptionProfiles
cloudfront:ListInvalidations
cloudfront:ListPublicKeys
cloudfront:ListStreamingDistributions
cloudfront:UpdateDistribution
cloudfront:TagResource
cloudfront:UntagResource
cloudfront:ListTagsForResource
cloudfront>DeleteDistribution
iam:AttachRolePolicy
iam:CreateRole
iam:CreateServiceLinkedRole
iam:GetRole
iam:PutRolePolicy
iam:PassRole
iam:UpdateAssumeRolePolicy
iam>DeleteRolePolicy
lambda:CreateFunction
lambda:EnableReplication
lambda>DeleteFunction
lambda:GetFunction
lambda:GetFunctionConfiguration
lambda:PublishVersion
lambda:UpdateFunctionCode
lambda:UpdateFunctionConfiguration
lambda:ListTags
lambda:TagResource
lambda:UntagResource
lambda:ListEventSourceMappings
lambda>CreateEventSourceMapping
```

```
route53:ChangeResourceRecordSets
route53:ListHostedZonesByName
route53:ListResourceRecordSets
s3:CreateBucket
s3:GetAccelerateConfiguration
s3:GetObject
s3:ListBucket
s3:PutAccelerateConfiguration
s3:PutBucketPolicy
s3:PutObject
s3:PutBucketTagging
s3:GetBucketTagging
sqs:CreateQueue
sqs>DeleteQueue
sqs:GetQueueAttributes
sqs:SetQueueAttributes
amplify:GetApp
amplify:GetBranch
amplify:UpdateApp
amplify:UpdateBranch
```

Problembehandlung bei Bereitstellungen von Next.js 1.1

Wenn bei der Bereitstellung einer Classic (nur Next.js 11) SSR-App mit Amplify unerwartete Probleme auftreten, lesen Sie sich die folgenden Themen zur Fehlerbehebung durch.

Themen

- [Ihr Ausgabeverzeichnis wurde überschrieben](#)
- [Nach der Bereitstellung Ihrer SSR-Site wird ein 404-Fehler angezeigt](#)
- [In Ihrer App fehlt die Rewrite-Regel für CloudFront SSR-Distributionen](#)
- [Ihre App ist zu groß, um bereitgestellt zu werden](#)
- [Ihr Build schlägt mit einem Fehler fehl, dass nicht genügend Arbeitsspeicher zur Verfügung steht](#)
- [Ihre App hat sowohl SSR- als auch SSG-Zweige](#)
- [Ihre App speichert statische Dateien in einem Ordner mit einem reservierten Pfad](#)
- [Ihre App hat ein CloudFront Limit erreicht](#)
- [Umgebungsvariablen werden nicht in Lambda-Funktionen übertragen](#)
- [Lambda @Edge -Funktionen werden in der Region USA Ost \(Nord-Virginia\) erstellt](#)
- [Ihre App Next.js verwendet Funktionen, die nicht unterstützt werden](#)

- [Bilder in Ihrer Next.js App werden nicht geladen](#)
- [Nicht unterstützte Regionen](#)

Ihr Ausgabeverzeichnis wurde überschrieben

Das Ausgabeverzeichnis für eine mit Amplify bereitgestellte App Next.js muss auf `.next` eingestellt sein. Wenn das Ausgabeverzeichnis Ihrer App überschrieben wird, überprüfen Sie die Datei `next.config.js`. Wenn das Build-Ausgabeverzeichnis standardmäßig den Wert haben soll `.next`, entfernen Sie die folgende Zeile aus der Datei:

```
distDir: 'build'
```

Stellen Sie sicher, dass das Ausgabeverzeichnis `.next` in Ihren Build-Einstellungen auf eingestellt ist. Informationen zum Anzeigen der Build-Einstellungen Ihrer App finden Sie unter [Build-Einstellungen konfigurieren](#).

Im Folgenden finden Sie ein Beispiel für die Build-Einstellungen für eine App, für die die Einstellung auf festgelegt `baseDirectory` ist `.next`.

```
version: 1
frontend:
  phases:
    preBuild:
      commands:
        - npm ci
    build:
      commands:
        - npm run build
  artifacts:
    baseDirectory: .next
    files:
      - '**/*'
  cache:
    paths:
      - node_modules/**/*
```

Nach der Bereitstellung Ihrer SSR-Site wird ein 404-Fehler angezeigt

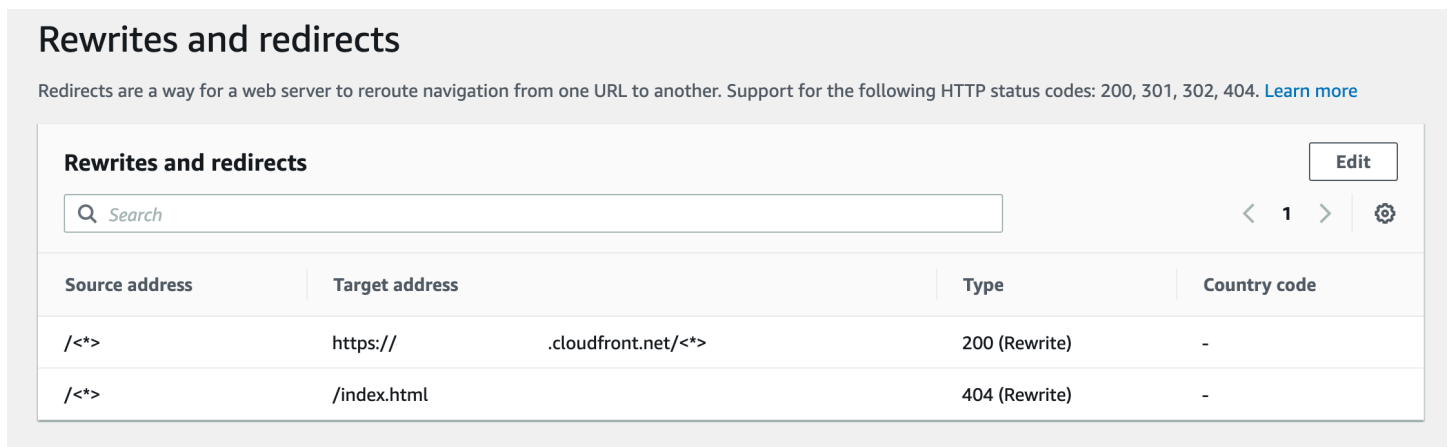
Wenn Sie nach der Bereitstellung Ihrer Site einen 404-Fehler erhalten, kann das Problem daran liegen, dass Ihr Ausgabeverzeichnis überschrieben wurde. Gehen Sie wie im vorherigen Thema

beschrieben vor, um Ihre `next.config.js` Datei und das richtige Build-Ausgabeverzeichnis in der Build-Spezifikation Ihrer App zu überprüfen. [Ihr Ausgabeverzeichnis wurde überschrieben](#)

In Ihrer App fehlt die Rewrite-Regel für CloudFront SSR-Distributionen

Wenn Sie eine SSR-App bereitstellen, erstellt Amplify eine Rewrite-Regel für Ihre CloudFront SSR-Distributionen. Wenn Sie in einem Webbrowser nicht auf Ihre App zugreifen können, stellen Sie sicher, dass die CloudFront Rewrite-Regel für Ihre App in der Amplify-Konsole existiert. Wenn sie fehlt, können Sie sie entweder manuell hinzufügen oder Ihre App erneut bereitstellen.

Um die Umschreib- und Umleitungsregeln einer App in der Amplify-Konsole anzuzeigen oder zu bearbeiten, wählen Sie im Navigationsbereich App-Einstellungen und dann Umschreibungen und Weiterleitungen aus. Der folgende Screenshot zeigt ein Beispiel für die Rewrite-Regeln, die Amplify für Sie erstellt, wenn Sie eine SSR-App bereitstellen. Beachten Sie, dass in diesem Beispiel eine CloudFront Rewrite-Regel existiert.



Rewrites and redirects

Redirects are a way for a web server to reroute navigation from one URL to another. Support for the following HTTP status codes: 200, 301, 302, 404. [Learn more](#)

Rewrites and redirects Edit

< 1 > ⚙️

Source address	Target address	Type	Country code
/ <code><*></code>	<code>https://.cloudfront.net/<*></code>	200 (Rewrite)	-
/ <code><*></code>	<code>/index.html</code>	404 (Rewrite)	-

Ihre App ist zu groß, um bereitgestellt zu werden

Amplify begrenzt die Größe einer SSR-Bereitstellung auf 50 MB. Wenn Sie versuchen, eine Next.js SSR-App für Amplify bereitzustellen und eine `RequestEntityTooLargeException` Fehlermeldung erhalten, ist Ihre App zu groß für die Bereitstellung. Sie können versuchen, dieses Problem zu umgehen, indem Sie Ihrer Datei Code zur Cache-Bereinigung hinzufügen. `next.config.js`

Im Folgenden finden Sie ein Beispiel für Code in der `next.config.js` Datei, der die Cache-Bereinigung durchführt.

```
module.exports = {
  webpack: (config, { buildId, dev, isServer, defaultLoaders, webpack }) => {
```

```
    config.optimization.splitChunks.cacheGroups = { }
    config.optimization.minimize = true;
    return config
  },
}
```

Ihr Build schlägt mit einem Fehler fehl, dass nicht genügend Arbeitsspeicher zur Verfügung steht

Next.js ermöglicht es Ihnen, Build-Artefakte zwischenzuspeichern, um die Leistung bei nachfolgenden Builds zu verbessern. Darüber hinaus komprimiert der AWS CodeBuild Container von Amplify diesen Cache und lädt ihn in Ihrem Namen auf Amazon S3 hoch, um die nachfolgende Build-Leistung zu verbessern. Dies könnte dazu führen, dass Ihr Build mit einem Fehler wegen unzureichenden Speichers fehlschlägt.

Führen Sie die folgenden Aktionen aus, um zu verhindern, dass Ihre App das Speicherlimit während der Erstellungsphase überschreitet. Entfernen Sie es zunächst `.next/cache/**/*` aus dem Abschnitt `cache.paths` Ihrer Build-Einstellungen. Als Nächstes entfernen Sie die `NODE_OPTIONS` Umgebungsvariable aus Ihrer Build-Einstellungsdatei. Stellen Sie stattdessen die `NODE_OPTIONS` Umgebungsvariable in der Amplify-Konsole ein, um das maximale Speicherlimit für den Knoten zu definieren. Weitere Informationen zum Einstellen von Umgebungsvariablen mit der Amplify-Konsole finden Sie unter [Festlegen von Umgebungsvariablen](#).

Nachdem Sie diese Änderungen vorgenommen haben, versuchen Sie es erneut mit Ihrem Build. Wenn dies erfolgreich ist, fügen Sie `.next/cache/**/*` dem Abschnitt `cache.paths` Ihrer Build-Einstellungsdatei etwas hinzu.

Weitere Informationen zur Cache-Konfiguration von Next.js zur Verbesserung der Build-Leistung finden Sie unter [AWS CodeBuild](#) auf der Website Next.js.

Ihre App hat sowohl SSR- als auch SSG-Zweige

Sie können keine App bereitstellen, die sowohl über SSR- als auch über SSG-Zweige verfügt. Wenn Sie sowohl SSR- als auch SSG-Branches bereitstellen müssen, müssen Sie eine App bereitstellen, die nur SSR-Branches verwendet, und eine andere App, die nur SSG-Zweige verwendet.

Ihre App speichert statische Dateien in einem Ordner mit einem reservierten Pfad

Next.js kann statische Dateien aus einem Ordner mit dem Namen `public` bereitstellen, der im Stammverzeichnis des Projekts gespeichert ist. Wenn Sie eine Next.js App mit Amplify bereitstellen und hosten, darf Ihr Projekt keine Ordner mit dem Pfad `public/static` enthalten. Amplify behält

sich den `public/static` Pfad für die Verwendung bei der Verteilung der App vor. Wenn Ihre App diesen Pfad enthält, müssen Sie den `static` Ordner umbenennen, bevor Sie ihn mit Amplify bereitstellen.

Ihre App hat ein CloudFront Limit erreicht

[CloudFront Servicekontingenten](#) beschränken Ihr AWS Konto auf 25 Distributionen mit angehängten Lambda @Edge -Funktionen. Wenn Sie dieses Kontingent überschreiten, können Sie entweder alle ungenutzten CloudFront Distributionen aus Ihrem Konto löschen oder eine Erhöhung des Kontingents beantragen. Weitere Informationen finden Sie unter [Beantragen einer Kontingenterhöhung](#) im Service-Quotas-Benutzerhandbuch.

Umgebungsvariablen werden nicht in Lambda-Funktionen übertragen

Umgebungsvariablen, die Sie in der Amplify-Konsole für eine SSR-App angeben, werden nicht auf die Funktionen der AWS Lambda App übertragen. Ausführliche Anweisungen zum Hinzufügen von Umgebungsvariablen [Umgebungsvariablen für serverseitige Laufzeiten zugänglich machen](#), auf die Sie in Ihren Lambda-Funktionen verweisen können, finden Sie unter ,.

Lambda @Edge -Funktionen werden in der Region USA Ost (Nord-Virginia) erstellt

Wenn Sie eine Next.js App bereitstellen, erstellt Amplify Lambda @Edge -Funktionen, um den bereitgestellten Inhalt anzupassen. CloudFront Lambda @Edge -Funktionen werden in der Region USA Ost (Nord-Virginia) erstellt, nicht in der Region, in der Ihre App bereitgestellt wird. Dies ist eine Lambda @Edge -Beschränkung. Weitere Informationen zu Lambda @Edge -Funktionen finden Sie unter [Einschränkungen für Edge-Funktionen](#) im Amazon CloudFront Developer Guide.

Ihre App Next.js verwendet Funktionen, die nicht unterstützt werden

Mit Amplify bereitgestellte Apps unterstützen die Hauptversionen von Next.js bis Version 11. Eine ausführliche Liste der Funktionen von Next.js, die von Amplify unterstützt und nicht unterstützt werden, finden Sie unter. [supported features](#)

Wenn Sie eine neue Next.js App bereitstellen, verwendet Amplify standardmäßig die neueste unterstützte Version von Next.js. Wenn Sie über eine bestehende Next.js App verfügen, die Sie mit einer älteren Version von Next.js auf Amplify bereitgestellt haben, können Sie die App zum Amplify Hosting Compute SSR-Anbieter migrieren. Anweisungen finden Sie unter [Migrieren einer Next.js 11-App auf Amplify Hosting Compute](#).

Bilder in Ihrer Next.js App werden nicht geladen

Wenn Sie Ihrer App Next.js mithilfe der `next/image` Komponente Bilder hinzufügen, darf die Größe des Bilds 1 MB nicht überschreiten. Wenn Sie die App auf Amplify bereitstellen, geben Bilder, die größer als 1 MB sind, einen 503-Fehler zurück. Dies wird durch ein Lambda @Edge -Limit verursacht, das die Größe einer Antwort, die von einer Lambda-Funktion generiert wird, einschließlich Header und Hauptteil, auf 1 MB beschränkt.

Das Limit von 1 MB gilt für andere Artefakte in Ihrer App, wie PDF- und Dokumentdateien.

Nicht unterstützte Regionen

Amplify unterstützt die klassische (nur Next.js 11) SSR-App-Bereitstellung nicht in jeder AWS Region, in der Amplify verfügbar ist. Classic (nur Next.js 11) SSR wird in den folgenden Regionen nicht unterstützt: Europa (Mailand) eu-south-1, Naher Osten (Bahrain) me-south-1 und Asien-Pazifik (Hongkong) ap-east-1.

Einrichtung benutzerdefinierter Domains

Sie können eine App, die Sie mit Amplify Hosting bereitgestellt haben, mit einer benutzerdefinierten Domain verbinden. Wenn Sie Amplify verwenden, um Ihre Web-App bereitzustellen, hostet Amplify sie für Sie auf der `amplifyapp.com` Standarddomain mit einer URL wie `https://branch-name.d1m7bkiki6tdw1.amplifyapp.com`. Wenn Sie Ihre App mit einer benutzerdefinierten Domain verbinden, sehen Benutzer, dass Ihre App auf einer benutzerdefinierten URL gehostet wird, z. `https://www.example.com`.

Sie können eine benutzerdefinierte Domain über einen akkreditierten Domain-Registrar wie Amazon Route 53 oder GoDaddy erwerben. Route 53 ist der Domain Name System (DNS) -Webservice von Amazon. Weitere Informationen zur Verwendung von Route 53 finden Sie unter [Was ist Amazon Route 53](#). Eine Liste der akkreditierten Domain-Registrierer von Drittanbietern finden Sie im [Verzeichnis der akkreditierten Registrare](#) auf der ICANN-Website.

Wenn Sie Ihre benutzerdefinierte Domain einrichten, können Sie das verwaltete Standardzertifikat verwenden, das Amplify für Sie bereitstellt, oder Sie können Ihr eigenes benutzerdefiniertes Zertifikat verwenden. Sie können das für die Domain verwendete Zertifikat jederzeit ändern. Ausführliche Informationen zur Verwaltung von Zertifikaten finden Sie unter [Verwenden von SSL/TLS-Zertifikaten](#).

Bevor Sie mit der Einrichtung einer benutzerdefinierten Domäne fortfahren, stellen Sie sicher, dass Sie die folgenden Voraussetzungen erfüllen.

- Sie besitzen einen registrierten Domainnamen.
- Sie haben ein Zertifikat, das von ausgestellt oder importiert wurde AWS Certificate Manager.
- Sie haben Ihre App auf Amplify Hosting bereitgestellt.

Weitere Informationen zum Ausführen dieses Schritts finden Sie unter [Erste Schritte mit Amplify Hosting](#).

- Sie verfügen über Grundkenntnisse in Domänen und DNS-Terminologie.

Weitere Informationen zu Domänen und DNS finden Sie unter [Verstehen der DNS-Terminologie und -Konzepte](#).

Themen

- [Verstehen der DNS-Terminologie und -Konzepte](#)

- [Verwenden von SSL/TLS-Zertifikaten](#)
- [Fügen Sie eine benutzerdefinierte Domain hinzu, die von Amazon Route 53 verwaltet wird](#)
- [Fügen Sie eine benutzerdefinierte Domain hinzu, die von einem DNS-Drittanbieter verwaltet wird](#)
- [Aktualisieren Sie die DNS-Einträge für eine Domain, die verwaltet wird von GoDaddy](#)
- [Aktualisieren Sie die DNS-Einträge für eine von Google Domains verwaltete Domain](#)
- [Aktualisieren Sie das SSL/TLS-Zertifikat für eine Domain](#)
- [Subdomains verwalten](#)
- [Wildcard-Unterdomänen](#)
- [Automatische Subdomains für eine benutzerdefinierte Amazon Route 53-Domain einrichten](#)
- [Problembehandlung bei benutzerdefinierten Domains](#)

Verstehen der DNS-Terminologie und -Konzepte

Wenn Sie mit den Begriffen und Konzepten im Zusammenhang mit Domain Name System (DNS) nicht vertraut sind, können Ihnen die folgenden Themen helfen, die Verfahren zum Hinzufügen benutzerdefinierter Domänen zu verstehen.

DNS-Terminologie

Im Folgenden finden Sie eine Liste von Begriffen, die im DNS gebräuchlich sind. Sie können Ihnen helfen, die Verfahren zum Hinzufügen benutzerdefinierter Domänen zu verstehen.

CNAME

Ein Canonical Record Name (CNAME) ist eine Art von DNS-Eintrag, der die Domain für eine Reihe von Webseiten maskiert und sie so aussehen lässt, als ob sie sich woanders befinden würden. Ein CNAME verweist eine Subdomain auf einen vollqualifizierten Domainnamen (FQDN). Sie können beispielsweise einen neuen CNAME-Eintrag erstellen, um die Subdomain `www.example.com`, wobei `www` die Subdomain ist, der FQDN-Domain `branch-name.d1m7bkiki6tdw1.cloudfront.net` zuzuordnen, die Ihrer App in der Amplify-Konsole zugewiesen wurde.

ANAME

Ein ANAME-Record ist wie ein CNAME-Record, allerdings auf Stammebene. Ein ANAME verweist das Stammverzeichnis Ihrer Domain auf einen FQDN. Dieser FQDN verweist auf eine IP-Adresse.

Namensserver

Ein Nameserver ist ein Server im Internet, der darauf spezialisiert ist, Anfragen zum Standort der verschiedenen Dienste eines Domainnamens zu bearbeiten. Wenn Sie Ihre Domain in Amazon Route 53 einrichten, ist Ihrer Domain bereits eine Liste von Nameservern zugewiesen.

NS-Eintrag

Ein NS-Eintrag verweist auf Nameserver, die Ihre Domaindetails nachschlagen.

DNS-Überprüfung

Ein Domain Name System (DNS) ist wie ein Telefonbuch, das menschenlesbare Domainnamen in computerfreundliche IP-Adressen übersetzt. Wenn Sie etwas **https://google.com** in einen Browser eingeben, wird im DNS-Anbieter ein Suchvorgang ausgeführt, um die IP-Adresse des Servers zu ermitteln, der die Website hostet.

DNS-Anbieter verfügen über die Datensätze der Domänen und die zugehörigen IP-Adressen. Die am häufigsten verwendeten DNS-Einträge sind CNAME-, ANAME- und NS-Einträge.

Amplify verwendet einen CNAME-Eintrag, um zu überprüfen, ob Sie Eigentümer Ihrer benutzerdefinierten Domain sind. Wenn Sie Ihre Domain bei Route 53 hosten, erfolgt die Überprüfung automatisch in Ihrem Namen. Wenn Sie Ihre Domain jedoch bei einem Drittanbieter hosten GoDaddy, müssen Sie die DNS-Einstellungen Ihrer Domain manuell aktualisieren und einen neuen CNAME-Eintrag hinzufügen, der von Amplify bereitgestellt wird.

Amplify Hosting-Aktivierungsprozess für benutzerdefinierte Domains

Wenn Sie mit Amplify Hosting eine benutzerdefinierte Domain hinzufügen, müssen Sie eine Reihe von Schritten ausführen, bevor Sie Ihre App mit Ihrer benutzerdefinierten Domain anzeigen können. In der folgenden Liste werden die einzelnen Schritte bei der Einrichtung der Domain beschrieben.

SSL/TLS-Erstellung

Wenn Sie ein verwaltetes Zertifikat verwenden, AWS Amplify stellt es ein SSL/TLS-Zertifikat für die Einrichtung einer sicheren benutzerdefinierten Domain aus.

SSL/TLS-Konfiguration und -Überprüfung

Vor der Ausstellung eines verwalteten Zertifikats überprüft Amplify, ob Sie der Eigentümer der Domain sind. Für Domains, die von Amazon Route 53 verwaltet werden, aktualisiert Amplify

automatisch den DNS-Bestätigungseintrag. Für Domains, die außerhalb von Route 53 verwaltet werden, müssen Sie den in der Amplify-Konsole bereitgestellten DNS-Bestätigungseintrag manuell zu Ihrer Domain bei einem DNS-Drittanbieter hinzufügen.

Wenn Sie ein benutzerdefiniertes Zertifikat verwenden, sind Sie für die Überprüfung des Domainbesitzes verantwortlich.

Aktivierung der Domain

Die Domain wurde erfolgreich verifiziert. Für Domains, die außerhalb von Route 53 verwaltet werden, müssen Sie die in der Amplify-Konsole bereitgestellten CNAME-Einträge manuell zu Ihrer Domain bei einem DNS-Drittanbieter hinzufügen.

Verwenden von SSL/TLS-Zertifikaten

Ein SSL/TLS-Zertifikat ist ein digitales Dokument, das es Webbrowsern ermöglicht, verschlüsselte Netzwerkverbindungen zu Websites mithilfe des sicheren SSL/TLS-Protokolls zu identifizieren und herzustellen. Wenn Sie Ihre benutzerdefinierte Domain einrichten, können Sie das verwaltete Standardzertifikat verwenden, das Amplify für Sie bereitstellt, oder Sie können Ihr eigenes benutzerdefiniertes Zertifikat verwenden.

Mit einem verwalteten Zertifikat stellt Amplify ein SSL/TLS-Zertifikat für alle mit Ihrer App verbundenen Domains aus, sodass der gesamte Datenverkehr über HTTPS/2 gesichert ist. Das von AWS Certificate Manager (ACM) generierte Standardzertifikat ist 13 Monate gültig und verlängert sich automatisch, solange Ihre App bei Amplify gehostet wird.

Warning

Amplify kann das Zertifikat nicht verlängern, wenn der CNAME-Bestätigungseintrag in den DNS-Einstellungen Ihres Domain-Anbieters geändert oder gelöscht wurde. Sie müssen die Domain in der Amplify-Konsole löschen und erneut hinzufügen.

Um ein benutzerdefiniertes Zertifikat verwenden zu können, müssen Sie ein Zertifikat von einer Drittanbieter-Zertifizierungsstelle Ihrer Wahl erwerben. Als Nächstes importieren Sie das Zertifikat in AWS Certificate Manager. ACM ist ein Dienst, mit dem Sie auf einfache Weise öffentliche und private SSL/TLS-Zertifikate für die Verwendung mit und Ihre internen verbundenen Ressourcen bereitstellen, verwalten AWS-Services und bereitstellen können. Stellen Sie sicher, dass Sie das Zertifikat in der Region USA Ost (Nord-Virginia) (us-east-1) anfordern oder importieren.

Stellen Sie sicher, dass Ihr benutzerdefiniertes Zertifikat alle Subdomains abdeckt, die Sie hinzufügen möchten. Sie können einen Platzhalter am Anfang Ihres Domainnamens verwenden, um mehrere Subdomains abzudecken. Wenn Ihre Domain beispielsweise `istexample.com`, können Sie die Wildcard-Domain einbeziehen. `*.example.com` Dies gilt für Subdomains wie `product.example.com` und `api.example.com`

Sobald Ihr benutzerdefiniertes Zertifikat in ACM verfügbar ist, können Sie es bei der Einrichtung der Domain auswählen. Anweisungen zum Importieren von Zertifikaten in AWS Certificate Manager finden Sie unter [Zertifikate importieren AWS Certificate Manager](#) in im AWS Certificate Manager Benutzerhandbuch.

Wenn Sie Ihr benutzerdefiniertes Zertifikat in ACM erneuern oder erneut importieren, aktualisiert Amplify die mit Ihrer benutzerdefinierten Domain verknüpften Zertifikatsdaten. Bei importierten Zertifikaten verwaltet ACM die Verlängerungen nicht automatisch. Sie sind dafür verantwortlich, Ihre benutzerdefinierten Zertifikate zu erneuern und erneut zu importieren.

Sie können das für eine Domain verwendete Zertifikat jederzeit ändern. Sie können beispielsweise vom verwalteten Standardzertifikat zu einem benutzerdefinierten Zertifikat oder von einem benutzerdefinierten Zertifikat zu einem verwalteten Zertifikat wechseln. Darüber hinaus können Sie das verwendete benutzerdefinierte Zertifikat in ein anderes benutzerdefiniertes Zertifikat ändern. Anweisungen zum Aktualisieren von Zertifikaten finden Sie unter [Aktualisieren des SSL/TLS-Zertifikats für eine Domain](#).

Fügen Sie eine benutzerdefinierte Domain hinzu, die von Amazon Route 53 verwaltet wird

Um eine benutzerdefinierte Domäne hinzuzufügen, die von Route 53 verwaltet wird

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie Ihre App aus, die Sie mit einer benutzerdefinierten Domain verbinden möchten.
3. Wählen Sie im Navigationsbereich Hosting, Benutzerdefinierte Domains aus.
4. Wählen Sie auf der Seite Benutzerdefinierte Domains die Option Domain hinzufügen aus.
5. Geben Sie den Namen Ihrer Root-Domain ein. Wenn der Name Ihrer Domain beispielsweise `https://example.com` lautet, geben Sie ein **example.com**.

Wenn Sie mit der Eingabe beginnen, werden alle Stammdomänen, die Sie bereits in Route 53 verwalten, in der Liste angezeigt. Sie können die Domain, die Sie verwenden möchten, aus der

Liste auswählen. Wenn Sie die Domain noch nicht besitzen und sie verfügbar ist, können Sie die Domain in [Amazon Route 53](#) erwerben.

6. Nachdem Sie Ihren Domainnamen eingegeben haben, wählen Sie Domain konfigurieren.
7. Standardmäßig erstellt Amplify automatisch zwei Subdomaineinträge für Ihre Domain. Wenn Ihr Domainname beispielsweise example.com lautet, werden Ihnen die Subdomains <https://www.example.com> und <https://example.com> mit einer Umleitung von der Root-Domain zur www-Subdomain angezeigt.

(Optional) Sie können die Standardkonfiguration ändern, wenn Sie nur Subdomains hinzufügen möchten. Um die Standardkonfiguration zu ändern, wählen Sie im Navigationsbereich Umschreibungen und Weiterleitungen aus und konfigurieren Sie dann Ihre Domain.

8. Wählen Sie das zu verwendende SSL/TLS-Zertifikat aus. Sie können entweder das verwaltete Standardzertifikat verwenden, das Amplify für Sie bereitstellt, oder ein benutzerdefiniertes Drittanbieterzertifikat, in das Sie importiert AWS Certificate Manager haben.
 - Verwenden Sie das von Amplify verwaltete Standardzertifikat.
 - Wählen Sie Amplify Managed Certificate.
 - Verwenden Sie ein benutzerdefiniertes Zertifikat eines Drittanbieters.
 - a. Wählen Sie Benutzerdefiniertes SSL-Zertifikat.
 - b. Wählen Sie das zu verwendende Zertifikat aus der Liste aus.
9. Wählen Sie Domain hinzufügen.

Note

Es kann bis zu 24 Stunden dauern, bis der DNS sich verbreitet und das Zertifikat ausgestellt hat. Hilfe zur Behebung von aufgetretenen Fehlern finden Sie unter.

[Problembehandlung bei benutzerdefinierten Domains](#)

Fügen Sie eine benutzerdefinierte Domain hinzu, die von einem DNS-Drittanbieter verwaltet wird

Wenn Sie Amazon Route 53 nicht zur Verwaltung Ihrer Domain verwenden, können Sie Ihrer mit Amplify bereitgestellten App eine benutzerdefinierte Domain hinzufügen, die von einem DNS-Drittanbieter verwaltet wird.

Wenn Sie unsere Google-Domains verwenden GoDaddy , finden Sie unter [the section called “Aktualisieren Sie die DNS-Einträge für eine Domain, die verwaltet wird von GoDaddy”](#) oder Informationen zu [the section called “Aktualisieren Sie die DNS-Einträge für eine von Google Domains verwaltete Domain”](#) den für diese Anbieter spezifischen Verfahren.

So fügen Sie eine benutzerdefinierte Domain hinzu, die von einem DNS-Drittanbieter verwaltet wird

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie Ihre App aus, zu der Sie eine benutzerdefinierte Domain hinzufügen möchten.
3. Wählen Sie im Navigationsbereich Hosting, Benutzerdefinierte Domains aus.
4. Wählen Sie auf der Seite Benutzerdefinierte Domains die Option Domain hinzufügen aus.
5. Geben Sie den Namen Ihrer Root-Domain ein. Wenn der Name Ihrer Domain beispielsweise `https://example.com` lautet, geben Sie ein **example.com**.
6. Amplify erkennt, dass Sie keine Route 53-Domain verwenden, und bietet Ihnen die Möglichkeit, eine gehostete Zone in Route 53 zu erstellen.
 - Um eine gehostete Zone in Route 53 zu erstellen
 - a. Wählen Sie Hosting-Zone auf Route 53 erstellen aus.
 - b. Wählen Sie Domain konfigurieren aus.
 - c. Die Nameserver der gehosteten Zone werden in der Konsole angezeigt. Rufen Sie die Website Ihres DNS-Anbieters auf und fügen Sie die Nameserver zu Ihren DNS-Einstellungen hinzu.
 - d. Wählen Sie Ich habe die oben genannten Nameserver zu meiner Domainregistrierung hinzugefügt.
 - e. Fahren Sie mit Schritt sieben fort.
 - Um mit der manuellen Konfiguration fortzufahren
 - a. Wählen Sie Manuelle Konfiguration
 - b. Wählen Sie Domain konfigurieren.
 - c. Fahren Sie mit Schritt 7 fort.
7. Standardmäßig erstellt Amplify automatisch zwei Subdomaineinträge für Ihre Domain. Wenn Ihr Domainname beispielsweise `example.com` lautet, werden Ihnen die Subdomains `https://www.example.com` und `https://example.com` mit einer Umleitung von der Root-Domain zur `www`-Subdomain angezeigt.

(Optional) Sie können die Standardkonfiguration ändern, wenn Sie nur Subdomains hinzufügen möchten. Um die Standardkonfiguration zu ändern, wählen Sie im Navigationsbereich Umschreibungen und Weiterleitungen aus und konfigurieren Sie Ihre Domain.

8. Wählen Sie das zu verwendende SSL/TLS-Zertifikat aus. Sie können entweder das verwaltete Standardzertifikat verwenden, das Amplify für Sie bereitstellt, oder ein benutzerdefiniertes Drittanbieterzertifikat, in das Sie importiert AWS Certificate Manager haben.
 - Verwenden Sie das von Amplify verwaltete Standardzertifikat.
 - Wählen Sie Amplify Managed Certificate.
 - Verwenden Sie ein benutzerdefiniertes Zertifikat eines Drittanbieters.
 - a. Wählen Sie Benutzerdefiniertes SSL-Zertifikat.
 - b. Wählen Sie das zu verwendende Zertifikat aus der Liste aus.
9. Wählen Sie Domain hinzufügen.
10. Wenn Sie in Schritt 6 die Option Gehostete Zone auf Route 53 erstellen ausgewählt haben, fahren Sie mit Schritt 15 fort.

Wenn Sie in Schritt 6 die Option Manuelle Konfiguration ausgewählt haben, müssen Sie Ihre DNS-Einträge bei Ihrem Drittanbieter für Domains aktualisieren.

Wählen Sie im Menü Aktionen die Option DNS-Einträge anzeigen aus. Der folgende Screenshot zeigt die DNS-Einträge, die in der Konsole angezeigt werden.

DNS Records

Verify records in your domain registrar match these records.

Verification record

Hostname	Type	Data/URL
<code>_39e1e8d7e0aedc8165cf52a176612124.testexample.com.</code>	CNAME	<code>_40404fb1d5a2a1bdec5b4ad98de4cfbb.mhbtsbpdnt.acm-validations.aws.</code>

Subdomain records

Hostname	Type	Data/URL
@	ANAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>
www	CNAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>

11. Führen Sie eine der folgenden Aktionen aus:

- Wenn Sie verwenden GoDaddy, gehen Sie zu [Aktualisieren Sie die DNS-Einträge für eine Domain, die verwaltet wird von GoDaddy](#).
- Wenn Sie Google Domains verwenden, gehen Sie zu [Aktualisieren Sie die DNS-Einträge für eine von Google Domains verwaltete Domain](#).
- Wenn Sie einen anderen DNS-Drittanbieter verwenden, fahren Sie mit dem nächsten Schritt in diesem Verfahren fort.

12. Gehen Sie zur Website Ihres DNS-Anbieters, melden Sie sich bei Ihrem Konto an und suchen Sie nach den DNS-Verwaltungseinstellungen für Ihre Domain. Sie werden zwei CNAME-Einträge konfigurieren.

13. Konfigurieren Sie den ersten CNAME-Eintrag so, dass er Ihre Subdomain auf den AWS Validierungsserver verweist.

Wenn die Amplify-Konsole einen DNS-Eintrag zur Überprüfung der Inhaberschaft Ihrer Subdomain anzeigt, z. B. `_c3e2d7eaf1e656b73f46cd6980fdc0e.example.com`, geben Sie nur für den CNAME-Record den Subdomainnamen ein. **`_c3e2d7eaf1e656b73f46cd6980fdc0e`**

Der folgende Screenshot zeigt den Speicherort des zu verwendenden Bestätigungsdatensatzes.

DNS Records

Verify records in your domain registrar match these records.

Verification record

Hostname	Type	Data/URL
<code>_c3e2d7eaf1e656b73f46cd6980fdc0e.testexample.com.</code>	CNAME	<code>_40404fb1d5a2a1bdec5b4ad98de4cfbb.mhbtsbpdnt.acm-validations.aws.</code>

Subdomain records

Hostname	Type	Data/URL
@	ANAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>
www	CNAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>

Wenn die Amplify-Konsole einen ACM-Validierungsservereintrag wie `_cjhvou20vhu2exampleuw20vuyb2ovb9.j9s73ucn9vy.acm-`

validations.aws anzeigt, geben Sie als CNAME-Datensatzwert ein.

`_cjhvou20vhu2exampleuw20vuyb2ovb9.j9s73ucn9vy.acm-validations.aws`

Der folgende Screenshot zeigt den Speicherort des zu verwendenden ACM-Bestätigungsdatensatzes.

DNS Records ×

Verify records in your domain registrar match these records.

Verification record

Hostname	Type	Data/URL
<code>_39e1e8d7e0aedc8165cf52a176612124.testexample.com.</code>	CNAME	<code>_40404fb1d5a2a1bdec5b4ad98de4cfbb.mhbtspbndt.acm-validations.aws.</code>

Subdomain records

Hostname	Type	Data/URL
@	ANAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>
www	CNAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>

Amplify verwendet diese Informationen, um die Inhaberschaft Ihrer Domain zu überprüfen und ein SSL/TLS-Zertifikat für Ihre Domain zu generieren. Sobald Amplify den Besitz Ihrer Domain bestätigt hat, wird der gesamte Traffic über HTTPS/2 bedient.

Note

Das von AWS Certificate Manager (ACM) generierte Standard-Amplify-Zertifikat ist 13 Monate gültig und verlängert sich automatisch, solange Ihre App bei Amplify gehostet wird. Amplify kann das Zertifikat nicht erneuern, wenn der CNAME-Bestätigungsdatensatz geändert oder gelöscht wurde. Sie müssen die Domain in der Amplify-Konsole löschen und erneut hinzufügen.

Important

Es ist wichtig, dass Sie diesen Schritt kurz nach dem Hinzufügen Ihrer benutzerdefinierten Domain in der Amplify-Konsole ausführen. Das AWS Certificate Manager (ACM) versucht sofort, die Inhaberschaft zu überprüfen. Mit der Zeit werden die

Kontrollen seltener. Wenn Sie Ihre CNAME-Einträge einige Stunden nach der Erstellung Ihrer App hinzufügen oder aktualisieren, kann dies dazu führen, dass Ihre App im Status „Ausstehende Überprüfung“ hängen bleibt.

- Konfigurieren Sie einen zweiten CNAME-Eintrag, um Ihre Subdomains auf die Amplify-Domain zu verweisen. Wenn Ihre Subdomain beispielsweise `www.example.com` ist, geben Sie `www` als Subdomainnamen ein.

Wenn die Amplify-Konsole die Domain für Ihre App als `d111111abcdef8.cloudfront.net` anzeigt, geben Sie für die Amplify-Domain ein. **`d111111abcdef8.cloudfront.net`**

Wenn Sie Produktions-Traffic haben, empfehlen wir Ihnen, diesen CNAME-Eintrag zu aktualisieren, nachdem Ihr Domainstatus in der Amplify-Konsole als VERFÜGBAR angezeigt wird.

Der folgende Screenshot zeigt den Speicherort des zu verwendenden Domainnamen-Eintrags.

DNS Records ×

Verify records in your domain registrar match these records.

Verification record

Hostname	Type	Data/URL
<code>_39e1e8d7e0aedc8165cf52a176612124.testexample.com.</code>	CNAME	<code>_40404fb1d5a2a1bdec5b4ad98de4cfbb.mhbtsbpdnt.acm-validations.aws.</code>

Subdomain records

Hostname	Type	Data/URL
@	ANAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>
www	CNAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>

- Konfigurieren Sie den ANAME/ALIAS-Eintrag so, dass er auf die Root-Domain Ihrer App verweist (z. B. `https://example.com`). Ein ANAME-Datensatz verweist das Stammverzeichnis Ihrer Domain auf einen Hostnamen. Wenn Sie Produktionsdatenverkehr haben, empfehlen wir Ihnen, Ihren ANAME-Eintrag zu aktualisieren, nachdem Ihr Domainstatus in der Konsole als VERFÜGBAR angezeigt wird. DNS-Anbietern, die keine ANAME/ALIAS-Unterstützung bieten, empfehlen wir dringend, Ihr DNS auf Route 53 zu migrieren. Weitere Informationen finden Sie unter [Amazon Route 53 als Ihren DNS-Service konfigurieren](#).

Note

Die Überprüfung des Domainbesitzes und der DNS-Weitergabe für Domains von Drittanbietern kann bis zu 48 Stunden dauern. Hilfe zur Behebung von aufgetretenen Fehlern finden Sie unter [Problembehandlung bei benutzerdefinierten Domains](#).

Aktualisieren Sie die DNS-Einträge für eine Domain, die verwaltet wird von GoDaddy

Um eine benutzerdefinierte Domain hinzuzufügen, die verwaltet wird von GoDaddy

1. Bevor Sie Ihre DNS-Einträge mit aktualisieren können GoDaddy, führen Sie die Schritte eins bis neun des Verfahrens durch [the section called "Fügen Sie eine benutzerdefinierte Domain hinzu, die von einem DNS-Drittanbieter verwaltet wird"](#).
2. Loggen Sie sich in Ihr GoDaddy Konto ein.
3. Suchen Sie in Ihrer Domainliste nach der Domain, die Sie hinzufügen möchten, und wählen Sie Manage DNS aus.
4. GoDaddy zeigt auf der DNS-Seite im Bereich DNS-Einträge eine Liste der Einträge für Ihre Domain an. Sie müssen zwei neue CNAME-Einträge hinzufügen.
5. Erstellen Sie den ersten CNAME-Eintrag, um Ihre Subdomains auf die Amplify-Domain zu verweisen.
 - a. Wählen Sie im Abschnitt DNS-Einträge die Option Neuen Eintrag hinzufügen aus.
 - b. Wählen Sie als Typ die Option CNAME aus.
 - c. Geben Sie als Name nur die Subdomain ein. Wenn Ihre Subdomain beispielsweise `www.example.com` ist, geben Sie `www` als Namen ein.
 - d. Sehen Sie sich für Value Ihre DNS-Einträge in der Amplify-Konsole an und geben Sie dann den Wert ein. Wenn die Amplify-Konsole die Domain für Ihre App als `d111111abcdef8.cloudfront.net` anzeigt, geben Sie Value ein.
d111111abcdef8.cloudfront.net

Der folgende Screenshot zeigt den Speicherort des zu verwendenden Domainnamen-Datensatzes.

DNS Records ×

Verify records in your domain registrar match these records.

Verification record

Hostname	Type	Data/URL
<code>_39e1e8d7e0aedc8165cf52a176612124.testexample.com.</code>	CNAME	<code>_40404fb1d5a2a1bdec5b4ad98de4cfbb.mhbtspbndt.acm-validations.aws.</code>

Subdomain records

Hostname	Type	Data/URL
@	ANAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>
www	CNAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>

- e. Wählen Sie Speichern.
6. Erstellen Sie den zweiten CNAME-Eintrag, der auf den AWS Certificate Manager (ACM) - Validierungsserver verweist. Ein einzelnes validiertes ACM generiert ein SSL/TLS-Zertifikat für Ihre Domain.
 - a. Wählen Sie als Typ die Option CNAME aus.
 - b. Geben Sie als Name die Subdomain ein.

Wenn der DNS-Eintrag in der Amplify-Konsole zur Überprüfung der Inhaberschaft Ihrer Subdomain beispielsweise `_c3e2d7eaf1e656b73f46cd6980fdc0e.example.com` lautet, geben Sie nur Name ein. **`_c3e2d7eaf1e656b73f46cd6980fdc0e`**

Der folgende Screenshot zeigt den Speicherort des zu verwendenden Bestätigungsdatensatzes.

DNS Records

Verify records in your domain registrar match these records.

Verification record

Hostname	Type	Data/URL
<code>_39e1e8d7e0aedc8165cf52a176612124.testexample.com.</code>	CNAME	<code>_40404fb1d5a2a1bdec5b4ad98de4cfbb.mhbtspbndt.acm-validations.aws.</code>

Subdomain records

Hostname	Type	Data/URL
@	ANAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>
www	CNAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>

- c. Geben Sie unter Value das ACM-Validierungszertifikat ein.

Wenn der Validierungsserver beispielsweise

`_cjhvou20vhu2exampleuw20vuyb2ovb9.j9s73ucn9vy.acm-validations.aws` ist, geben Sie `_cjhvou20vhu2exampleuw20vuyb2ovb9.j9s73ucn9vy.acm-validations.aws` für Value ein.

Der folgende Screenshot zeigt den Speicherort des zu verwendenden ACM-Bestätigungsdatensatzes.

DNS Records

Verify records in your domain registrar match these records.

Verification record

Hostname	Type	Data/URL
<code>_39e1e8d7e0aedc8165cf52a176612124.testexample.com.</code>	CNAME	<code>_40404fb1d5a2a1bdec5b4ad98de4cfbb.mhbtspbndt.acm-validations.aws.</code>

Subdomain records

Hostname	Type	Data/URL
@	ANAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>
www	CNAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>

- d. Wählen Sie Speichern.

Note

Das von AWS Certificate Manager (ACM) generierte Standard-Amplify-Zertifikat ist 13 Monate gültig und verlängert sich automatisch, solange Ihre App bei Amplify gehostet wird. Amplify kann das Zertifikat nicht erneuern, wenn der CNAME-Bestätigungsdatensatz geändert oder gelöscht wurde. Sie müssen die Domain in der Amplify-Konsole löschen und erneut hinzufügen.

7. Dieser Schritt ist für Subdomains nicht erforderlich. GoDaddy unterstützt keine ANAME/ALIAS-Einträge. Für DNS-Anbieter ohne ANAME/ALIAS-Unterstützung wird dringend empfohlen, Ihr DNS zu Amazon Route 53 zu migrieren. Weitere Informationen finden Sie unter [Amazon Route 53 als Ihren DNS-Service konfigurieren](#).

Wenn Sie Ihren Anbieter behalten GoDaddy und die Root-Domain aktualisieren möchten, fügen Sie Forwarding hinzu und richten Sie eine Domain-Forward ein:

- a. Suchen Sie auf der DNS-Seite nach dem Menü oben auf der Seite und wählen Sie Weiterleitung.
- b. Wählen Sie im Bereich Domain die Option Weiterleitung hinzufügen aus.
- c. Wählen Sie http:// und geben Sie dann als Ziel-URL den Namen Ihrer Subdomain ein, zu der Sie weiterleiten möchten (z. B. www.example.com).
- d. Wählen Sie als Weiterleitungstyp die Option Temporär (302) aus.
- e. Wählen Sie „Speichern“.

Aktualisieren Sie die DNS-Einträge für eine von Google Domains verwaltete Domain

Um eine benutzerdefinierte Domain hinzuzufügen, die von Google Domains verwaltet wird

1. Bevor Sie Ihre DNS-Einträge mit Google Domains aktualisieren können, führen Sie die Schritte eins bis neun des Verfahrens [So fügen Sie eine benutzerdefinierte Domain hinzu, die von einem DNS-Drittanbieter verwaltet wird](#).
2. Melden Sie sich unter <https://domains.google.com> bei Ihrem Konto an und wählen Sie im linken Navigationsbereich Meine Domains aus.

3. Suchen Sie in Ihrer Domainliste nach der Domain, die Sie hinzufügen möchten, und wählen Sie Verwalten aus.
4. Wählen Sie im linken Navigationsbereich DNS aus. Google zeigt die Ressourceneinträge für Ihre Domain an. Sie müssen zwei neue CNAME-Einträge hinzufügen.
5. Erstellen Sie den ersten CNAME-Eintrag, um alle Subdomains wie folgt auf die Amplify-Domain zu verweisen:
 - a. Geben Sie als Hostname nur den Namen der Subdomain ein. Wenn Ihre Subdomain beispielsweise `www.example.com` lautet, geben Sie `www` als Hostnamen ein.
 - b. Wählen Sie als Typ die Option CNAME aus.
 - c. Geben Sie für Daten den Wert ein, der in der Amplify-Konsole verfügbar ist.

Wenn die Amplify-Konsole die Domain für Ihre App als `d111111abcdef8.cloudfront.net` anzeigt, geben Sie `d111111abcdef8.cloudfront.net` für Daten ein.

Der folgende Screenshot zeigt den Speicherort des zu verwendenden Domainnamen-Datensatzes.

DNS Records ×

Verify records in your domain registrar match these records.

Verification record

Hostname	Type	Data/URL
<code>_39e1e8d7e0aedc8165cf52a176612124.testexample.com.</code>	CNAME	<code>_40404fb1d5a2a1bdec5b4ad98de4cfbb.mhbtspbndt.acm-validations.aws.</code>

Subdomain records

Hostname	Type	Data/URL
<code>@</code>	ANAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>
<code>www</code>	CNAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>

6. Erstellen Sie den zweiten CNAME-Eintrag, der auf den AWS Certificate Manager (ACM) - Validierungsserver verweist. Ein einzelnes validiertes ACM generiert ein SSL/TLS-Zertifikat für Ihre Domain.
 - a. Geben Sie als Hostname die Subdomain ein.

Wenn der DNS-Eintrag in der Amplify-Konsole zur Überprüfung der Inhaberschaft Ihrer Subdomain beispielsweise `_c3e2d7eaf1e656b73f46cd6980fdc0e.example.com` lautet, geben Sie nur `_c3e2d7eaf1e656b73f46cd6980fdc0e` als Hostname ein.

Der folgende Screenshot zeigt den Speicherort des zu verwendenden Bestätigungsdatensatzes.

DNS Records

Verify records in your domain registrar match these records.

Verification record

Hostname	Type	Data/URL
<code>_39e1e8d7e0aedc8165cf52a176612124.testexample.com.</code>	CNAME	<code>_40404fb1d5a2a1bdec5b4ad98de4cfbb.mhbtsbpdnt.acm-validations.aws.</code>

Subdomain records

Hostname	Type	Data/URL
@	ANAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>
www	CNAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>

- b. Wählen Sie als Typ die Option CNAME aus.
- c. Geben Sie für Daten das ACM-Validierungszertifikat ein.

Wenn der Validierungsserver beispielsweise `_cf1z2npwt9vzexample93c1j4xzc92wl.2te3iym6kzr.acm-validations.aws` ist, geben Sie `_cf1z2npwt9vzexample93c1j4xzc92wl.2te3iym6kzr.acm-validations.aws` ein. für Data.

Der folgende Screenshot zeigt den Speicherort des zu verwendenden ACM-Bestätigungsdatensatzes.

DNS Records ×

Verify records in your domain registrar match these records.

Verification record

Hostname	Type	Data/URL
<code>_39e1e8d7e0aedc8165cf52a176612124.testexample.com.</code>	CNAME	<code>_40404fb1d5a2a1bdec5b4ad98de4cfbb.mhbtspbndt.acm-validations.aws.</code>

Subdomain records

Hostname	Type	Data/URL
@	ANAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>
www	CNAME	<code>d1zp5qtgx0mgpb.cloudfront.net</code>

7. Wählen Sie Speichern.

Note

Das von AWS Certificate Manager (ACM) generierte Standard-Amplify; -Zertifikat ist 13 Monate gültig und verlängert sich automatisch, solange Ihre App bei Amplify gehostet wird. Amplify kann das Zertifikat nicht erneuern, wenn der CNAME-Bestätigungsdatensatz geändert oder gelöscht wurde. Sie müssen die Domain in der Amplify-Konsole löschen und erneut hinzufügen.

- Die Unterstützung von Google Domains für ANAME/ALIAS-Datensätze befindet sich in der Vorschauphase. Für DNS-Anbieter ohne ANAME/ALIAS-Unterstützung wird dringend empfohlen, Ihr DNS zu Amazon Route 53 zu migrieren. Weitere Informationen finden Sie unter [Amazon Route 53 als Ihren DNS-Service konfigurieren](#). Wenn Sie Google Domains als Ihren Anbieter behalten und die Root-Domain aktualisieren möchten, richten Sie eine Subdomain-Weiterleitung ein. Suchen Sie die Website-Seite für Ihre Google-Domain. Wählen Sie dann Domain weiterleiten und konfigurieren Sie Ihre Weiterleitung auf der Seite Webweiterleitung.

Note

Es kann bis zu 48 Stunden dauern, bis Aktualisierungen Ihrer DNS-Einstellungen für eine Google-Domain wirksam werden. Hilfe zur Behebung von aufgetretenen Fehlern finden Sie unter [Problembehandlung bei benutzerdefinierten Domains](#).

Aktualisieren Sie das SSL/TLS-Zertifikat für eine Domain

Sie können das SSL/TLS-Zertifikat, das für eine Domain verwendet wird, jederzeit ändern. Sie können beispielsweise von der Verwendung eines verwalteten Zertifikats zur Verwendung eines benutzerdefinierten Zertifikats wechseln. Sie können auch das benutzerdefinierte Zertifikat ändern, das für die Domain verwendet wird. Weitere Informationen zu Zertifikaten finden Sie unter [Verwenden von SSL/TLS-Zertifikaten](#).

Gehen Sie wie folgt vor, um den Zertifikatstyp oder das benutzerdefinierte Zertifikat zu aktualisieren, das für eine Domain verwendet wird.

Um das Zertifikat einer Domain zu aktualisieren

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie Ihre App aus, die Sie aktualisieren möchten.
3. Wählen Sie im Navigationsbereich Hosting, Benutzerdefinierte Domains aus.
4. Wählen Sie auf der Seite Benutzerdefinierte Domänen die Option Domänenkonfiguration aus.
5. Suchen Sie auf der Detailseite für Ihre Domain den Abschnitt Benutzerdefiniertes SSL-Zertifikat. Das Verfahren zur Aktualisierung Ihres Zertifikats hängt von der Art der Änderung ab, die Sie vornehmen möchten.
 - Um von einem benutzerdefinierten Zertifikat zum standardmäßigen verwalteten Amplify-Zertifikat zu wechseln
 - Wählen Sie Amplify Managed Certificate.
 - Um von einem verwalteten Zertifikat zu einem benutzerdefinierten Zertifikat zu wechseln
 - a. Wählen Sie Benutzerdefiniertes SSL-Zertifikat.
 - b. Wählen Sie das zu verwendende Zertifikat aus der Liste aus.
 - Um ein benutzerdefiniertes Zertifikat in ein anderes benutzerdefiniertes Zertifikat zu ändern
 - Wählen Sie für Benutzerdefiniertes SSL-Zertifikat das neue zu verwendende Zertifikat aus der Liste aus.
6. Wählen Sie Speichern. Die Statusdetails für die Domain geben an, dass Amplify den SSL-Erstellungsprozess für ein verwaltetes Zertifikat oder den Konfigurationsprozess für ein benutzerdefiniertes Zertifikat eingeleitet hat.

Subdomains verwalten

Eine Subdomain ist der Teil Ihrer URL, der vor Ihrem Domainnamen steht. Zum Beispiel ist `www` die Subdomain von `www.amazon.com` und `aws` ist die Subdomain von `aws.amazon.com`. Wenn Sie bereits eine Produktionswebsite haben, möchten Sie möglicherweise nur eine Subdomain verbinden. Subdomains können auch mehrstufig sein, zum Beispiel hat `beta.alpha.example.com` die mehrstufige Subdomain `beta.alpha`.

Nur um eine Subdomain hinzuzufügen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie Ihre App aus, zu der Sie eine Subdomain hinzufügen möchten.
3. Wählen Sie im Navigationsbereich Hosting und dann Benutzerdefinierte Domains aus.
4. Wählen Sie auf der Seite Benutzerdefinierte Domains die Option Domain hinzufügen aus.
5. Geben Sie den Namen Ihrer Root-Domain ein und wählen Sie dann Domain konfigurieren. Wenn der Name Ihrer Domain beispielsweise `https://example.com` lautet, geben Sie `example.com` ein.
6. Wählen Sie Exclude root und ändern Sie den Namen der Subdomain. Wenn die Domain beispielsweise `example.com` lautet, können Sie sie so ändern, dass nur die Subdomain Alpha hinzugefügt wird.
7. Wählen Sie Domain hinzufügen.

Um eine mehrstufige Subdomain hinzuzufügen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie Ihre App aus, zu der Sie eine mehrstufige Subdomain hinzufügen möchten.
3. Wählen Sie im Navigationsbereich Hosting und dann Benutzerdefinierte Domains aus.
4. Wählen Sie auf der Seite Benutzerdefinierte Domains die Option Domain hinzufügen aus.
5. Geben Sie den Namen einer Domain mit einer Subdomain ein, wählen Sie Exclude root aus und ändern Sie die Subdomain, um eine neue Ebene hinzuzufügen.

Wenn Sie beispielsweise eine Domain namens `alpha.example.com` haben und eine mehrstufige Subdomain `beta.alpha.example.com` erstellen möchten, würden Sie `beta` als Subdomain-Wert eingeben.

6. Wählen Sie Domain hinzufügen.

Um eine Subdomain hinzuzufügen oder zu bearbeiten

Nachdem Sie einer App eine benutzerdefinierte Domain hinzugefügt haben, können Sie eine bestehende Subdomain bearbeiten oder eine neue Subdomain hinzufügen.

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie Ihre App aus, für die Sie Subdomains verwalten möchten.
3. Wählen Sie im Navigationsbereich Hosting und dann Benutzerdefinierte Domains aus.
4. Wählen Sie auf der Seite Benutzerdefinierte Domänen die Option Domänenkonfiguration aus.
5. Im Abschnitt Subdomains können Sie Ihre vorhandenen Subdomains nach Bedarf bearbeiten.
6. (Optional) Um eine neue Subdomain hinzuzufügen, wählen Sie Neu hinzufügen.
7. Wählen Sie Speichern.

Wildcard-Unterdomänen

Amplify Hosting unterstützt jetzt Wildcard-Subdomains. Eine Wildcard-Subdomain ist eine Sammel-Subdomain, mit der Sie bestehende und nicht existierende Subdomains auf einen bestimmten Zweig Ihrer Anwendung verweisen können. Wenn Sie einen Platzhalter verwenden, um alle Subdomains in einer App einem bestimmten Zweig zuzuordnen, können Sie den Benutzern Ihrer App in jeder Subdomain denselben Inhalt bereitstellen und müssen nicht jede Subdomain einzeln konfigurieren.

Um eine Wildcard-Subdomain zu erstellen, geben Sie ein Sternchen (*) als Subdomännennamen an. Wenn Sie beispielsweise die Platzhalter-Subdomain `*.example.com` für einen bestimmten Zweig Ihrer App angeben, wird jede URL, die mit `example.com` endet, an den Zweig weitergeleitet. In diesem Fall werden Anfragen für `dev.example.com` und an die `prod.example.com` Subdomain weitergeleitet. `*.example.com`

Beachten Sie, dass Amplify Platzhalter-Subdomains nur für eine benutzerdefinierte Domain unterstützt. Sie können diese Funktion nicht mit der Standarddomain verwenden. `amplifyapp.com`

Die folgenden Anforderungen gelten für Wildcard-Subdomänen:

- Der Subdomainname darf nur mit einem Sternchen (*) angegeben werden.
- Sie können keinen Platzhalter verwenden, um einen Teil eines Subdomain-Namens wie folgt zu ersetzen: `*domain.example.com`.
- Sie können eine Subdomain nicht in der Mitte eines Domainnamens ersetzen, wie hier: `subdomain*.example.com`.

- Standardmäßig decken alle von Amplify bereitgestellten Zertifikate alle Subdomains für eine benutzerdefinierte Domain ab.

Um eine Wildcard-Subdomain hinzuzufügen oder zu löschen

Nachdem Sie einer App eine benutzerdefinierte Domain hinzugefügt haben, können Sie eine Wildcard-Subdomain für einen App-Branch hinzufügen.

1. Melden Sie sich bei der [Amplify Hosting-Konsole](#) an AWS Management Console und öffnen Sie sie.
2. Wählen Sie Ihre App aus, für die Sie Wildcard-Subdomains verwalten möchten.
3. Wählen Sie im Navigationsbereich Hosting und dann Benutzerdefinierte Domains aus.
4. Wählen Sie auf der Seite Benutzerdefinierte Domänen die Option Domänenkonfiguration aus.
5. Im Abschnitt Subdomains können Sie Wildcard-Subdomains hinzufügen oder löschen.
 - Um eine neue Wildcard-Subdomain hinzuzufügen
 - a. Wählen Sie Add new (Neuen hinzufügen) aus.
 - b. Geben Sie für die Subdomain eine ein. *
 - c. Wählen Sie für Ihren App-Branch einen Zweignamen aus der Liste aus.
 - d. Wählen Sie Speichern.
 - Um eine Wildcard-Subdomain zu löschen
 - a. Wählen Sie neben dem Namen der Subdomain die Option Entfernen aus. Der Datenverkehr zu einer Subdomain, die nicht explizit konfiguriert ist, wird gestoppt, und Amplify Hosting gibt für diese Anfragen einen 404-Statuscode zurück.
 - b. Wählen Sie Speichern.

Automatische Subdomains für eine benutzerdefinierte Amazon Route 53-Domain einrichten

Nachdem eine App mit einer benutzerdefinierten Domain in Route 53 verbunden wurde, können Sie mit Amplify automatisch Subdomains für neu verbundene Filialen erstellen. Wenn Sie beispielsweise Ihren Dev-Branch verbinden, kann Amplify automatisch `dev.exampledomain.com` erstellen. Wenn Sie einen Branch löschen, werden alle zugehörigen Subdomains automatisch gelöscht.

Um die automatische Erstellung von Subdomains für neu verbundene Filialen einzurichten

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie eine App aus, die mit einer benutzerdefinierten Domain verbunden ist, die in Route 53 verwaltet wird.
3. Wählen Sie im Navigationsbereich Hosting und dann Benutzerdefinierte Domänen aus.
4. Wählen Sie auf der Seite Benutzerdefinierte Domänen die Option Domänenkonfiguration aus.
5. Aktivieren Sie im Abschnitt Automatische Erstellung von Subdomains die Funktion.

Note

Diese Funktion ist nur für Root-Domains verfügbar, z. B. `exampledomain.com`. Die Amplify-Konsole zeigt dieses Kontrollkästchen nicht an, wenn Ihre Domain bereits eine Subdomain ist, z. B. `dev.exampledomain.com`.

Webvorschauen mit Subdomains

Nachdem Sie die automatische Erstellung von Subdomains anhand der obigen Anweisungen aktiviert haben, können Sie auf die Pull-Request-Webvorschauen Ihrer App auch mit automatisch erstellten Subdomains zugreifen. Wenn ein Pull-Request geschlossen wird, werden der zugehörige Branch und die Subdomain automatisch gelöscht. Weitere Informationen zum Einrichten von Webvorschauen für Pull Requests finden Sie unter [Webvorschauen für Pull-Requests](#)

Problembehandlung bei benutzerdefinierten Domains

Wenn beim Hinzufügen einer benutzerdefinierten Domain zu einer App in der AWS Amplify Konsole Probleme auftreten, finden Sie in den folgenden Themen in diesem Abschnitt Hilfe zur Problembehandlung.

Wenn Sie hier keine Lösung für Ihr Problem finden, wenden Sie sich an AWS Support. Weitere Informationen finden Sie unter [Erstellen eines Support-Falls](#) im Benutzerhandbuch von AWS Support

Themen

- [Wie kann ich überprüfen, ob mein CNAME aufgelöst wird?](#)



- [Meine bei einem Drittanbieter gehostete Domain befindet sich im Status „Ausstehende Überprüfung“](#)
- [Meine mit Amazon Route 53 gehostete Domain befindet sich im Status „Ausstehende Überprüfung“](#)
- [Ich erhalte einen CNAME-Fehler AlreadyExistsException](#)
- [Ich erhalte die Fehlermeldung „Zusätzliche Überprüfung erforderlich“](#)
- [Ich erhalte eine 404-Fehlermeldung für die URL CloudFront](#)
- [Ich erhalte beim Besuch meiner Domain SSL-Zertifikat- oder HTTPS-Fehler](#)

Wie kann ich überprüfen, ob mein CNAME aufgelöst wird?

1. Nachdem Sie Ihre DNS-Einträge bei Ihrem Domain-Drittanbieter aktualisiert haben, können Sie ein Tool wie [dig](#) oder eine kostenlose Website wie <https://www.whatsmydns.net/> verwenden, um zu überprüfen, ob Ihr CNAME-Eintrag korrekt aufgelöst wird. Der folgende Screenshot zeigt, wie Sie [whatsmydns.net](https://www.whatsmydns.net/) verwenden, um Ihren CNAME-Eintrag für die Domain www.example.com zu überprüfen.



2. Wählen Sie Suchen und [whatsmydns.net](https://www.whatsmydns.net/) zeigt die Ergebnisse für Ihren CNAME an. Der folgende Screenshot ist ein Beispiel für eine Ergebnisliste, die bestätigt, dass der CNAME korrekt in eine cloudfront.net-URL aufgelöst wird.

 Dallas TX, United States Speakeasy	d1e0xkpcedddpz.cloudfront.net ✓
 Reston VA, United States Sprint	d1e0xkpcedddpz.cloudfront.net ✓
 Atlanta GA, United States Speakeasy	d1e0xkpcedddpz.cloudfront.net ✓

Meine bei einem Drittanbieter gehostete Domain befindet sich im Status „Ausstehende Überprüfung“

1. Wenn bei Ihrer benutzerdefinierten Domain der Status „Überprüfung ausstehend“ nicht mehr angezeigt wird, überprüfen Sie, ob CNAME Ihre Daten behoben werden. Anweisungen zur Durchführung dieser Aufgabe finden Sie im vorherigen Thema zur [CNAMEProblembeseitigung, Wie überprüfe ich, ob meine Probleme behoben](#) sind?
2. Wenn Ihre CNAME Datensätze nicht aufgelöst werden, überprüfen Sie bei Ihrem Domain-Anbieter, ob der CNAME Eintrag in Ihren DNS-Einstellungen vorhanden ist.

Important

Es ist wichtig, dass Sie Ihre CNAME Einträge aktualisieren, sobald Sie Ihre benutzerdefinierte Domain erstellt haben. Nachdem Ihre App in der Amplify-Konsole erstellt wurde, wird Ihr CNAME Datensatz alle paar Minuten überprüft, um festzustellen, ob er aufgelöst wird. Wenn es nach einer Stunde nicht behoben wird, wird die Überprüfung alle paar Stunden durchgeführt, was zu einer Verzögerung bei der Einsatzbereitschaft Ihrer Domain führen kann. Wenn Sie Ihre CNAME Datensätze einige Stunden nach der Erstellung Ihrer App hinzugefügt oder aktualisiert haben, ist dies der wahrscheinlichste Grund dafür, dass Ihre App im Status „Ausstehende Überprüfung“ hängen bleibt.

3. Wenn Sie überprüft haben, dass der CNAME Eintrag existiert, liegt möglicherweise ein Problem mit Ihrem DNS-Anbieter vor. Sie können sich entweder an den DNS-Anbieter wenden, um zu diagnostizieren, warum die DNS-Überprüfung CNAME nicht funktioniert, oder Sie können Ihr DNS auf Route 53 migrieren. Weitere Informationen finden Sie unter [Amazon Route 53 zum DNS-Service für eine bestehende Domain machen](#).

Meine mit Amazon Route 53 gehostete Domain befindet sich im Status „Ausstehende Überprüfung“

Wenn Sie Ihre Domain zu Amazon Route 53 übertragen haben, ist es möglich, dass Ihre Domain andere Nameserver hat als die, die von Amplify bei der Erstellung Ihrer App ausgestellt wurden. Führen Sie die folgenden Schritte aus, um die Ursache des Fehlers zu diagnostizieren.

1. Melden Sie sich bei der [Amazon Route 53 53-Konsole](#) an

- Wählen Sie im Navigationsbereich Hosted Zones und dann den Namen der Domain aus, mit der Sie eine Verbindung herstellen möchten.
- Notieren Sie sich die Nameserver-Werte aus dem Abschnitt Hosted Zone Details. Sie benötigen diese Werte, um den nächsten Schritt abzuschließen. Der folgende Screenshot der Route 53-Konsole zeigt die Position der Nameserver-Werte in der unteren rechten Ecke.

The screenshot shows the 'Hosted Zone Details' panel in the AWS Route 53 console. The 'Name Servers' field is highlighted with a red box and contains the following list of nameservers:

- ns-2003.awsdns-58.co.uk
- ns-70.awsdns-08.com
- ns-1173.awsdns-18.org
- ns-805.awsdns-36.net

- Klicken Sie im Navigationsbereich auf Registered domains (Registrierte Domains). Stellen Sie sicher, dass die im Abschnitt Registrierte Domänen angezeigten Nameserver mit den Nameserver-Werten übereinstimmen, die Sie im vorherigen Schritt im Abschnitt Details zur Hosting-Zone aufgezeichnet haben. Wenn sie nicht übereinstimmen, bearbeiten Sie die Nameserver-Werte so, dass sie mit den Werten in Ihrer Hostzone übereinstimmen. Der folgende Screenshot der Route 53-Konsole zeigt die Position der Nameserver-Werte auf der rechten Seite.

Registered domains > designaws.com

The screenshot shows the 'Registered domains' page for 'designaws.com'. The 'Name servers' field is highlighted with a red box, and a red arrow points to it with the text 'Modify this to match NameServers in your hosted zone.' The nameservers listed are:

- ns-294.awsdns-36.com
- ns-1886.awsdns-43.co.uk
- ns-953.awsdns-55.net
- ns-1192.awsdns-21.org

The 'DNSSEC status' is shown as 'Not available'.

- Wenn das Problem dadurch nicht behoben wird, wenden Sie sich an AWS Support. Weitere Informationen finden Sie unter [Erstellen eines Support-Falls](#) im Benutzerhandbuch von AWS Support .

Ich erhalte einen CNAME-Fehler AlreadyExistsException

Wenn Sie einen AlreadyExistsExceptionCNAME-Fehler erhalten, bedeutet dies, dass einer der Hostnamen, mit denen Sie versucht haben, eine Verbindung herzustellen (eine Subdomain oder die Apex-Domain), bereits für eine andere Amazon-Distribution bereitgestellt ist. CloudFront Führen Sie die folgenden Schritte aus, um die Ursache des Fehlers zu diagnostizieren.

1. Melden Sie sich bei der [CloudFrontAmazon-Konsole](#) an und vergewissern Sie sich, dass Sie diese Domain nicht für eine andere Distribution bereitgestellt haben. Ein einzelner CNAME Datensatz kann jeweils an eine CloudFront Distribution angehängt werden.
2. Wenn Sie die Domain zuvor für eine CloudFront Distribution bereitgestellt haben, müssen Sie sie entfernen.
 - a. Wählen Sie im linken Navigationsmenü Distributionen aus.
 - b. Wählen Sie den Namen der Distribution aus, die bearbeitet werden soll.
 - c. Wählen Sie die Registerkarte Allgemein. Wählen Sie im Abschnitt Settings (Einstellungen) die Option Edit (Bearbeiten) aus.
 - d. Entfernen Sie den Domainnamen aus dem alternativen Domainnamen (CNAME). Wählen Sie dann „Änderungen speichern“.
3. Überprüfe, ob diese Domain mit einer anderen Amplify-App verbunden ist, die du besitzt. Stellen Sie in diesem Fall sicher, dass Sie nicht versuchen, einen der Hostnamen wiederzuwenden. Wenn Sie `www.example.com` für eine andere App verwenden, können Sie `www.example.com` nicht mit der App verwenden, mit der Sie gerade eine Verbindung herstellen. Sie können andere Subdomains verwenden, z. B. `blog.example.com`.
4. Wenn diese Domain erfolgreich mit einer anderen App verbunden und dann innerhalb der letzten Stunde gelöscht wurde, versuchen Sie es nach mindestens einer Stunde erneut. Wenn Sie diese Ausnahme nach 6 Stunden immer noch sehen, wenden Sie sich an AWS Support. Weitere Informationen finden Sie unter [Erstellen eines Support-Falls](#) im Benutzerhandbuch von AWS Support .

Ich erhalte die Fehlermeldung „Zusätzliche Überprüfung erforderlich“

Wenn Sie die Fehlermeldung „Zusätzliche Überprüfung erforderlich“ erhalten, bedeutet dies, dass AWS Certificate Manager (ACM) zusätzliche Informationen benötigt, um diese Zertifikatsanforderung zu bearbeiten. Dies kann als Betrugsschutzmaßnahme geschehen, z. B. wenn die Domäne unter den [Top 1000 Websites von Alexa](#) rangiert. Um diese Informationen bereitzustellen, kontaktieren Sie

[über das](#) Support-Center AWS Support. Wenn Sie über keinen Supportplan verfügen, veröffentlichen Sie einen neuen Thread im [ACM-Diskussionsforum](#).

Note

Sie können kein Zertifikat für Amazon-eigene Domainnamen, wie solche, die mit amazonaws.com, cloudfront.net oder elasticbeanstalk.com enden, anfordern.

Ich erhalte eine 404-Fehlermeldung für die URL CloudFront

Um Traffic bereitzustellen, verweist Amplify Hosting über einen CNAME-Eintrag auf eine CloudFront URL. Beim Verbinden einer App mit einer benutzerdefinierten Domain zeigt die Amplify-Konsole die CloudFront URL für die App an. Über diese CloudFront URL können Sie jedoch nicht direkt auf Ihre Anwendung zugreifen. Es wird ein 404-Fehler zurückgegeben. Ihre Anwendung wird nur mithilfe der Amplify-App-URL (z. B. `https://main.d5udybEXAMPLE.amplifyapp.com`) oder Ihrer benutzerdefinierten Domain (zum Beispiel `www.example.com`) aufgelöst.

Amplify muss Anfragen an den richtigen bereitgestellten Branch weiterleiten und verwendet dazu den Hostnamen. Sie können beispielsweise die Domain `www.example.com` konfigurieren, die auf den Mainline-Zweig einer App verweist, aber auch konfigurieren, `dev.example.com` dass sie auf den Entwicklungszweig derselben App verweist. Daher müssen Sie Ihre Anwendung auf der Grundlage der konfigurierten Subdomains aufrufen, damit Amplify die Anfragen entsprechend weiterleiten kann.

Ich erhalte beim Besuch meiner Domain SSL-Zertifikat- oder HTTPS-Fehler

Wenn Sie DNS-Einträge für Certificate Authority Authorization (CAA) bei Ihrem DNS-Drittanbieter konfiguriert haben, kann AWS Certificate Manager (ACM) möglicherweise keine Zwischenzertifikate für Ihr benutzerdefiniertes Domain-SSL-Zertifikat aktualisieren oder neu ausstellen. Um dieses Problem zu lösen, müssen Sie einen CAA-Eintrag hinzufügen, um mindestens einer der Zertifizierungsstellen-Domains von Amazon als vertrauenswürdig einzustufen. Das folgende Verfahren beschreibt die Schritte, die Sie ausführen müssen.

So fügen Sie einen CAA-Eintrag hinzu, um einer Amazon-Zertifizierungsstelle zu vertrauen

1. Konfigurieren Sie einen CAA-Eintrag bei Ihrem Domain-Anbieter, um mindestens einer der Amazon-Zertifizierungsstellen-Domains zu vertrauen. Weitere Informationen zur Konfiguration des CAA-Eintrags finden Sie unter [Probleme mit der Zertifizierungsstellen-Autorisierung \(CAA\)](#) im AWS Certificate Manager Benutzerhandbuch.

2. Verwenden Sie eine der folgenden Methoden, um Ihr SSL-Zertifikat zu aktualisieren:

- Manuelles Update mit der Amplify-Konsole.

Note

Diese Methode führt zu Ausfallzeiten für Ihre benutzerdefinierte Domain.

- a. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
- b. Wählen Sie Ihre App aus, zu der Sie einen CAA-Eintrag hinzufügen möchten.
- c. Wählen Sie im Navigationsbereich App-Einstellungen, Domainverwaltung aus.
- d. Löschen Sie auf der Seite Domainverwaltung die benutzerdefinierte Domain.
- e. Connect deine App erneut mit der benutzerdefinierten Domain. Durch diesen Vorgang wird ein neues SSL-Zertifikat ausgestellt, und die Zwischenzertifikate können jetzt von ACM verwaltet werden.

Verwenden Sie eines der folgenden Verfahren, das dem von Ihnen verwendeten Domain-Anbieter entspricht, um Ihre App wieder mit Ihrer benutzerdefinierten Domain zu verbinden.

- [Fügen Sie eine benutzerdefinierte Domain hinzu, die von Amazon Route 53 verwaltet wird.](#)
 - [Fügen Sie eine benutzerdefinierte Domain hinzu, die von einem DNS-Drittanbieter verwaltet wird.](#)
 - [Aktualisieren Sie die DNS-Einträge für eine Domain, die verwaltet wird von GoDaddy.](#)
 - [Aktualisieren Sie die DNS-Einträge für eine von Google Domains verwaltete Domain.](#)
- Wenden Sie sich AWS Support an uns, um Ihr SSL-Zertifikat erneut ausstellen zu lassen.

Build-Einstellungen konfigurieren

Wenn Sie eine App mit Amplify Hosting bereitstellen, erkennt sie automatisch das Front-End-Framework und die zugehörigen Build-Einstellungen, indem sie die `package.json` Datei in Ihrem Repository überprüft. Sie haben die folgenden Optionen zum Speichern der Build-Einstellungen Ihrer App:

- Speichern Sie die Build-Einstellungen in der Amplify-Konsole — Die Amplify-Konsole erkennt die Build-Einstellungen automatisch und speichert sie, sodass über die Amplify-Konsole auf sie zugegriffen werden kann. Amplify wendet diese Einstellungen auf alle Ihre Branches an, es sei denn, in Ihrem Repository ist eine `amplify.yml` Datei gespeichert.
- Speichern Sie die Build-Einstellungen in Ihrem Repository — Laden Sie die `amplify.yml` Datei herunter und fügen Sie sie dem Stammverzeichnis Ihres Repositories hinzu.

Sie können die Build-Einstellungen einer App in der Amplify-Konsole bearbeiten, indem Sie im Navigationsbereich Hosting und dann Build-Einstellungen auswählen. Die Build-Einstellungen werden auf alle Branches in Ihrer App angewendet, mit Ausnahme der Branches, für die eine `amplify.yml` Datei im Repository gespeichert ist.

Note

Die Build-Einstellungen sind nur dann im Hosting-Menü der Amplify-Konsole sichtbar, wenn eine App für die kontinuierliche Bereitstellung eingerichtet und mit einem Git-Repository verbunden ist. Anweisungen zu dieser Art der Bereitstellung finden Sie unter [Erste Schritte](#).

Befehle und Einstellungen für Build-Spezifikationen

Die Build-Spezifikation YAML enthält eine Sammlung von Build-Befehlen und zugehörigen Einstellungen, die Amplify verwendet, um Ihren Build auszuführen. In der folgenden Liste werden diese Einstellungen und ihre Verwendung beschrieben.

`version`

Die Amplify YAML-Versionsnummer.

AppRoot

Der Pfad innerhalb des Repositorys, in dem sich diese Anwendung befindet. Wird ignoriert, sofern nicht mehrere Anwendungen definiert sind.

env

Fügen Sie diesem Abschnitt Umgebungsvariablen hinzu. Sie können Umgebungsvariablen auch mithilfe der Konsole hinzufügen.

Backend

Führen Sie Amplify CLI-Befehle aus, um im Rahmen einer kontinuierlichen Bereitstellung ein Backend bereitzustellen, Lambda-Funktionen oder GraphQL-Schemas zu aktualisieren.

Frontend

Führen Sie Frontend-Build-Befehle aus.

Test

Befehle während einer Testphase ausführen. Erfahren Sie, wie Sie [Ihrer App Tests hinzufügen](#).

Phasen erstellen

Das Frontend, das Backend und der Test bestehen aus drei Phasen, die die Befehle darstellen, die während jeder Sequenz des Builds ausgeführt werden.

- PreBuild — Das PreBuild-Skript wird ausgeführt, bevor der eigentliche Build gestartet wird, aber nachdem Amplify Abhängigkeiten installiert hat.
- Build: Ihre Build-Befehle.
- PostBuild — Das Post-Build-Skript wird ausgeführt, nachdem der Build abgeschlossen ist und Amplify alle erforderlichen Artefakte in das Ausgabeverzeichnis kopiert hat.

Buildpath

Der Pfad, der zum Ausführen des Builds verwendet werden soll. Amplify verwendet diesen Pfad, um Ihre Build-Artefakte zu finden. Wenn Sie keinen Pfad angeben, verwendet Amplify beispielsweise den Monorepo-App-Root. apps/app

Artefakte>Basisverzeichnis

Das Verzeichnis, in dem Ihre Build-Artefakte existieren.

Artefakte>Dateien

Geben Sie Dateien aus Ihren Artefakten an, die Sie bereitstellen möchten. Geben Sie ein `**/*`, um alle Dateien einzubeziehen.

Cache

Das Cache-Feld der Buildspezifikation wird verwendet, um Abhängigkeiten von der Buildzeit wie den Ordner `node_modules` zwischenspeichern. Es wird automatisch basierend auf dem Paketmanager und dem Framework vorgeschlagen, in dem die App des Kunden integriert ist. Während des ersten Builds werden alle Pfade hier zwischengespeichert, und bei nachfolgenden Builds füllen wir den Cache erneut auf und verwenden diese zwischengespeicherten Abhängigkeiten, wo immer möglich, um die Build-Zeit zu verkürzen.

Die folgende Beispiel-Build-Spezifikation demonstriert die grundlegende YAML-Syntax:

YAML-Syntax für die Build-Spezifikation

```
version: 1
env:
  variables:
    key: value
backend:
  phases:
    preBuild:
      commands:
        - *enter command*
    build:
      commands:
        - *enter command*
    postBuild:
      commands:
        - *enter command*
frontend:
  buildpath:
  phases:
    preBuild:
      commands:
        - cd react-app
        - npm ci
    build:
      commands:
```



```
    - npm run build
artifacts:
  files:
    - location
    - location
  discard-paths: yes
  baseDirectory: location
cache:
  paths:
    - path
    - path
test:
  phases:
    preTest:
      commands:
        - *enter command*
    test:
      commands:
        - *enter command*
    postTest:
      commands:
        - *enter command*
artifacts:
  files:
    - location
    - location
  configFile: *location*
  baseDirectory: *location*
```

Branchenspezifische Build-Einstellungen

Sie können Bash-Shell-Skripts verwenden, um verzweigungsspezifische Build-Einstellungen festzulegen. Das folgende Skript verwendet beispielsweise die Systemumgebungsvariable `$AWS_BRANCH`, um einen Befehlssatz auszuführen, wenn der Branch-Name `main` ist, und einen anderen Befehlssatz, wenn der Branchname `dev` ist.

```
frontend:
  phases:
    build:
      commands:
        - if [ "${AWS_BRANCH}" = "main" ]; then echo "main branch"; fi
```

```
- if [ "${AWS_BRANCH}" = "dev" ]; then echo "dev branch"; fi
```

Zu einem Unterordner navigieren

Bei Monorepos möchten Benutzer in der Lage sein, `cd` in einen Ordner zu gelangen, um den Build auszuführen. Nachdem Sie den `cd` Befehl ausgeführt haben, gilt er für alle Phasen Ihres Builds, sodass Sie den Befehl nicht in separaten Phasen wiederholen müssen.

```
version: 1
env:
  variables:
    key: value
frontend:
  phases:
    preBuild:
      commands:
        - cd react-app
        - npm ci
    build:
      commands:
        - npm run build
```

Bereitstellung des Backends mit dem Frontend für eine Gen-1-App

Note

Dieser Abschnitt gilt nur für Amplify Gen 1-Anwendungen. Ein Gen 1-Backend wird mit Amplify Studio und der Amplify-Befehlszeilenschnittstelle (CLI) erstellt.

Der `amplifyPush` Befehl ist ein Hilfsskript, das Ihnen bei Backend-Bereitstellungen hilft. Die Build-Einstellungen unten ermitteln automatisch die richtige Backend-Umgebung für die Bereitstellung der aktuellen Verzweigung.

```
version: 1
env:
  variables:
    key: value
```

```
backend:
  phases:
    build:
      commands:
        - amplifyPush --simple
```

Einstellung des Ausgabeordners

Mit den folgenden Build-Einstellungen wird der öffentliche Ordner als Ausgabeordner festgelegt.

```
frontend:
  phases:
    commands:
      build:
        - yarn run build
  artifacts:
    baseDirectory: public
```

Pakete als Teil eines Builds installieren

Sie können die `yarn` Befehle `npm` oder verwenden, um Pakete während des Builds zu installieren.

```
frontend:
  phases:
    build:
      commands:
        - npm install -g <package>
        - <package> deploy
        - yarn run build
  artifacts:
    baseDirectory: public
```

Verwenden Sie eine private NPM-Registrierung

Sie können eine private Registrierung als Umgebungsvariable hinzufügen oder Verweise darauf in Ihren Build-Einstellungen hinzufügen.

```
build:
```

```
phases:
  preBuild:
    commands:
      - npm config set <key> <value>
      - npm config set registry https://registry.npmjs.org
      - npm config set always-auth true
      - npm config set email hello@amplifyapp.com
      - yarn install
```

Installieren von OS-Paketen

Das AL203-Image von Amplify führt Ihren Code mit einem nicht privilegierten Benutzer namens `aus.amplify`. Amplify gewährt diesem Benutzer Rechte, Betriebssystembefehle mit dem `sudo` Linux-Befehl auszuführen. Wenn Sie Betriebssystempakete für fehlende Abhängigkeiten installieren möchten, können Sie Befehle wie `yum` und `rpm` mit `sudo` verwenden.

Der folgende Beispiel-Build-Abschnitt demonstriert die Syntax für die Installation eines Betriebssystempakets mithilfe des `sudo` Befehls.

```
build:
  phases:
    preBuild:
      commands:
        - sudo yum install -y <package>
```

Schlüssel-Wert-Speicher für jeden Build

Der `envCache` stellt die Speicherung von Schlüsselwerten bei der Erstellung bereit. Werte, die in gespeichert sind, `envCache` können nur während eines Builds geändert und beim nächsten Build wiederverwendet werden. Mithilfe von können wir Informationen über die bereitgestellte Umgebung speichern und sie dem Build-Container in aufeinanderfolgenden Builds zur Verfügung stellen. `envCache` Im Gegensatz zu Werten `envCache`, die in der gespeichert sind, werden Änderungen an Umgebungsvariablen während eines Builds nicht für future Builds beibehalten.

Beispielverwendung:

```
envCache --set <key> <value>
envCache --get <key>
```

Überspringe den Build für einen Commit

Um einen automatischen Build für einen bestimmten Commit zu überspringen, fügen Sie den Text `[skip-cd]` am Ende der Commit-Nachricht ein.

Deaktivieren Sie automatische Builds

Sie können Amplify so konfigurieren, dass automatische Builds bei jedem Code-Commit deaktiviert werden. Wählen Sie zur Einrichtung App-Einstellungen, Branch-Einstellungen und suchen Sie dann den Abschnitt Branches, in dem die verbundenen Branches aufgelistet sind. Wählen Sie einen Zweig aus und klicken Sie dann auf Aktionen, auto Erstellung deaktivieren. Neue Commits für diesen Branch führen nicht mehr zu einem neuen Build.

Aktiviert oder deaktiviert das auf Diff basierende Frontend-Build und Deployment

Sie können Amplify so konfigurieren, dass diff-basierte Frontend-Builds verwendet werden. Wenn diese Option aktiviert ist, versucht Amplify zu Beginn jedes Builds standardmäßig `appRoot`, einen Diff für Ihren oder den `/src/` Ordner auszuführen. Wenn Amplify keine Unterschiede feststellt, überspringt es die Schritte zum Erstellen, Testen (falls konfiguriert) und Bereitstellen des Frontends und aktualisiert Ihre gehostete App nicht.

Um das diff-basierte Frontend zu konfigurieren, zu erstellen und bereitzustellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die Sie den Diff-basierten Frontend-Build und die Bereitstellung konfigurieren möchten.
3. Wählen Sie im Navigationsbereich Hosting, Umgebungsvariablen aus.
4. Wählen Sie im Abschnitt Umgebungsvariablen die Option Variablen verwalten aus.
5. Das Verfahren zur Konfiguration der Umgebungsvariablen hängt davon ab, ob Sie das diff-basierte Frontend-Build und Deploy aktivieren oder deaktivieren.
 - Um das diff-basierte Frontend zu aktivieren, erstellen und implementieren Sie es
 - a. Geben **AMPLIFY_DIFF_DEPLOY** Sie im Abschnitt Variablen verwalten unter Variable den Wert ein.
 - b. Geben Sie für Wert `true` ein.

- Um das auf Diff basierende Frontend-Build und Deployment zu deaktivieren
 - Führen Sie eine der folgenden Aktionen aus:
 - Suchen **AMPLIFY_DIFF_DEPLOYS** im Abschnitt Variablen verwalten nach. Geben Sie für Wert `false` ein.
 - Entfernen Sie die `AMPLIFY_DIFF_DEPLOY` Umgebungsvariable.

6. Wählen Sie Speichern.

Optional können Sie die `AMPLIFY_DIFF_DEPLOY_ROOT` Umgebungsvariable so einstellen, dass sie den Standardpfad durch einen Pfad überschreibt, der sich auf das Stammverzeichnis Ihres Repositorys bezieht, z. B. `dist`

Aktivieren oder deaktivieren Sie diff-basierte Backend-Builds für eine Gen-1-App

Note

Dieser Abschnitt gilt nur für Amplify Gen 1-Anwendungen. Ein Gen 1-Backend wird mit Amplify Studio und der Amplify-Befehlszeilenschnittstelle (CLI) erstellt.

Sie können Amplify Hosting so konfigurieren, dass es diff-basierte Backend-Builds verwendet, indem Sie die `AMPLIFY_DIFF_BACKEND` Umgebungsvariable verwenden. Wenn Sie diff-basierte Backend-Builds aktivieren, versucht Amplify zu Beginn jedes Builds, einen Diff für den `amplify` Ordner in Ihrem Repository auszuführen. Wenn Amplify keine Unterschiede feststellt, überspringt es den Backend-Build-Schritt und aktualisiert Ihre Backend-Ressourcen nicht. Wenn Ihr Projekt keinen `amplify` Ordner in Ihrem Repository hat, ignoriert Amplify den Wert der `AMPLIFY_DIFF_BACKEND` Umgebungsvariablen.

Wenn Sie derzeit benutzerdefinierte Befehle in den Build-Einstellungen Ihrer Backend-Phase angegeben haben, funktionieren bedingte Backend-Builds nicht. Wenn Sie möchten, dass diese benutzerdefinierten Befehle ausgeführt werden, müssen Sie sie in der Datei Ihrer App in die Frontend-Phase Ihrer Build-Einstellungen verschieben. `amplify.yml`

Um diff-basierte Backend-Builds zu konfigurieren

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).

2. Wählen Sie die App aus, für die diff-basierte Backend-Builds konfiguriert werden sollen.
3. Wählen Sie im Navigationsbereich Hosting, Umgebungsvariablen aus.
4. Wählen Sie im Abschnitt Umgebungsvariablen die Option Variablen verwalten aus.
5. Das Verfahren zur Konfiguration der Umgebungsvariablen hängt davon ab, ob Sie diff-basierte Backend-Builds aktivieren oder deaktivieren.
 - So aktivieren Sie diff-basierte Backend-Builds
 - a. Geben **AMPLIFY_DIFF_BACKENDS** im Abschnitt Variablen verwalten unter Variable den Wert ein.
 - b. Geben Sie für Wert `true` ein.
 - Um Diff-basierte Backend-Builds zu deaktivieren
 - Führen Sie eine der folgenden Aktionen aus:
 - Suchen **AMPLIFY_DIFF_BACKENDS** im Abschnitt Variablen verwalten nach. Geben Sie für Wert `false` ein.
 - Entfernen Sie die `AMPLIFY_DIFF_BACKEND` Umgebungsvariable.
6. Wählen Sie Speichern.

Monorepo-Build-Einstellungen

Wenn Sie mehrere Projekte oder Microservices in einem einzigen Repository speichern, wird dies als Monorepo bezeichnet. Sie können Amplify Hosting verwenden, um Anwendungen in einem Monorepo bereitzustellen, ohne mehrere Build-Konfigurationen oder Branch-Konfigurationen zu erstellen.

Amplify unterstützt Apps in generischen Monorepos sowie Apps in Monorepos, die mit `npm workspace`, `pnpm workspace`, `Yarn workspace`, `Nx` und `Turborepo` erstellt wurden. Wenn Sie Ihre App bereitstellen, erkennt Amplify automatisch das von Ihnen verwendete Monorepo-Build-Tool. Amplify wendet automatisch Build-Einstellungen für Apps in einem NPM-Workspace, Yarn-Workspace oder `Nx` an. `Turborepo`- und `pnpm`-Apps erfordern eine zusätzliche Konfiguration. Weitere Informationen finden Sie unter [Konfiguration der Turborepo- und pnpm-Monorepo-Apps](#).

Sie können die Build-Einstellungen für ein Monorepo in der Amplify-Konsole speichern oder die `amplify.yml` Datei herunterladen und zum Stammverzeichnis Ihres Repositorys hinzufügen. Amplify wendet die in der Konsole gespeicherten Einstellungen auf alle Ihre Branches an, es sei denn, es findet eine `amplify.yml` Datei in Ihrem Repository. Wenn eine `amplify.yml` Datei

vorhanden ist, überschreiben ihre Einstellungen alle in der Amplify-Konsole gespeicherten Build-Einstellungen.

Monorepo-Build-Spezifikation, YAML-Syntax

Die YAML-Syntax für eine Monorepo-Build-Spezifikation unterscheidet sich von der YAML-Syntax für ein Repository, das eine einzelne Anwendung enthält. Bei einem Monorepo deklarieren Sie jedes Projekt in einer Liste von Anwendungen. Sie müssen für jede Anwendung, die Sie in Ihrer Monorepo-Build-Spezifikation deklarieren, den folgenden zusätzlichen `appRoot` Schlüssel angeben:

AppRoot

Das Stammverzeichnis innerhalb des Repositorys, in dem die Anwendung gestartet wird. Dieser Schlüssel muss vorhanden sein und denselben Wert wie die `AMPLIFY_MONOREPO_APP_ROOT` Umgebungsvariable haben. Anweisungen zum Einstellen dieser Umgebungsvariablen finden Sie unter [Einstellung der Umgebungsvariablen AMPLIFY_MONOREPO_APP_ROOT](#).

Das folgende Beispiel für eine Monorepo-Build-Spezifikation zeigt, wie mehrere Amplify-Anwendungen im selben Repo deklariert werden. Die beiden `Appsreact-app`, und `angular-app` sind in der Liste deklariert. `applications` Der `appRoot` Schlüssel für jede App gibt an, dass sich die App im `apps` Stammordner des Repositorys befindet.

Das `buildpath` Attribut ist so eingestellt, dass / die App vom Monorepo-Projektstamm aus ausgeführt und erstellt wird.

Monorepo-Build-Spezifikation, YAML-Syntax

```
version: 1
applications:
  - appRoot: apps/react-app
    env:
      variables:
        key: value
    backend:
      phases:
        preBuild:
          commands:
            - *enter command*
        build:
          commands:
            - *enter command*
```



```
    postBuild:
      commands:
        - *enter command*
frontend:
  buildPath: / # Run install and build from the monorepo project root
  phases:
    preBuild:
      commands:
        - *enter command*
        - *enter command*
    build:
      commands:
        - *enter command*
  artifacts:
    files:
      - location
      - location
    discard-paths: yes
    baseDirectory: location
  cache:
    paths:
      - path
      - path
  test:
    phases:
      preTest:
        commands:
          - *enter command*
      test:
        commands:
          - *enter command*
      postTest:
        commands:
          - *enter command*
    artifacts:
      files:
        - location
        - location
      configFile: *location*
      baseDirectory: *location*
- appRoot: apps/angular-app
env:
  variables:
    key: value
```

```
backend:
  phases:
    preBuild:
      commands:
        - *enter command*
    build:
      commands:
        - *enter command*
    postBuild:
      commands:
        - *enter command*
frontend:
  phases:
    preBuild:
      commands:
        - *enter command*
        - *enter command*
    build:
      commands:
        - *enter command*
artifacts:
  files:
    - location
    - location
  discard-paths: yes
  baseDirectory: location
cache:
  paths:
    - path
    - path
test:
  phases:
    preTest:
      commands:
        - *enter command*
    test:
      commands:
        - *enter command*
    postTest:
      commands:
        - *enter command*
artifacts:
  files:
    - location
```

```
- location
configFilePath: *location*
baseDirectory: *location*
```

Einstellung der Umgebungsvariablen AMPLIFY_MONOREPO_APP_ROOT

Wenn Sie eine in einem Monorepo gespeicherte App bereitstellen, muss die AMPLIFY_MONOREPO_APP_ROOT Umgebungsvariable der App denselben Wert haben wie der Pfad des App-Stammverzeichnisses, relativ zum Stammverzeichnis Ihres Repositorys. Zum Beispiel ein Monorepo `ExampleMonorepo` mit dem Namen eines Stammordners `namensapps`, der `app1``app2`, enthält und die folgende Verzeichnisstruktur `app3` hat:

```
ExampleMonorepo
  apps
    app1
    app2
    app3
```

In diesem Beispiel ist der Wert der AMPLIFY_MONOREPO_APP_ROOT Umgebungsvariablen für `app1`. `apps/app1`

Wenn Sie eine Monorepo-App mit der Amplify-Konsole bereitstellen, legt die Konsole die AMPLIFY_MONOREPO_APP_ROOT Umgebungsvariable automatisch anhand des Werts fest, den Sie für den Pfad zum Stammverzeichnis der App angeben. Wenn Ihre Monorepo-App jedoch bereits in Amplify vorhanden ist oder mit Amplify bereitgestellt wird AWS CloudFormation, müssen Sie die Umgebungsvariable im Abschnitt **AMPLIFY_MONOREPO_APP_ROOT** Umgebungsvariablen in der Amplify-Konsole manuell festlegen.

Automatisches Einstellen der Umgebungsvariablen AMPLIFY_MONOREPO_APP_ROOT während der Bereitstellung

Die folgenden Anweisungen zeigen, wie Sie eine Monorepo-App mit der Amplify-Konsole bereitstellen. Amplify legt die AMPLIFY_MONOREPO_APP_ROOT Umgebungsvariable automatisch mithilfe des Stammordners der App fest, den Sie in der Konsole angeben.

Um eine Monorepo-App mit der Amplify-Konsole bereitzustellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).

2. Wählen Sie in der oberen rechten Ecke Neue App erstellen.
3. Wählen Sie auf der Seite „Mit Amplify erstellen“ Ihren Git-Anbieter und dann Weiter aus.
4. Gehen Sie auf der Seite Repository-Zweig hinzufügen wie folgt vor:
 - a. Wählen Sie den Namen Ihres Repositories aus der Liste aus.
 - b. Wählen Sie den Namen der Filiale, die Sie verwenden möchten.
 - c. Wählen Sie Meine App ist ein Monorepo
 - d. Geben Sie den Pfad zu Ihrer App in Ihrem Monorepo ein, zum Beispiel. **apps/app1**
 - e. Wählen Sie Weiter aus.
5. Auf der Seite mit den App-Einstellungen können Sie die Standardeinstellungen verwenden oder die Build-Einstellungen für Ihre App anpassen. Im Abschnitt Umgebungsvariablen setzt Amplify `AMPLIFY_MONOREPO_APP_ROOT` auf den Pfad, den Sie in Schritt 4d angegeben haben.
6. Wählen Sie Weiter aus.
7. Wählen Sie auf der Seite „Überprüfen“ die Option Speichern und bereitstellen.

Einstellung der Umgebungsvariablen `AMPLIFY_MONOREPO_APP_ROOT` für eine bestehende App

Verwenden Sie die folgenden Anweisungen, um die `AMPLIFY_MONOREPO_APP_ROOT` Umgebungsvariable für eine App manuell festzulegen, die bereits auf Amplify bereitgestellt wurde oder mit CloudFormation der erstellt wurde.

Um die Umgebungsvariable `AMPLIFY_MONOREPO_APP_ROOT` für eine bestehende App festzulegen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie den Namen der App, für die Sie die Umgebungsvariable festlegen möchten.
3. Wählen Sie im Navigationsbereich Hosting und dann Umgebungsvariablen aus.
4. Wählen Sie auf der Seite Umgebungsvariablen die Option Variablen verwalten aus.
5. Gehen Sie im Abschnitt Variablen verwalten wie folgt vor:
 - a. Wählen Sie Add new (Neuen hinzufügen) aus.
 - b. Geben Sie für Variable den Schlüssel ein `AMPLIFY_MONOREPO_APP_ROOT`.
 - c. Geben Sie unter Value beispielsweise den Pfad zur App ein **apps/app1**.

- d. Für Branch wendet Amplify standardmäßig die Umgebungsvariable auf alle Branches an.
6. Wählen Sie Speichern.

Konfiguration der Turborepo- und pnpm-Monorepo-Apps

Die Monorepo-Build-Tools Turborepo und pnpm Workspace rufen Konfigurationsinformationen aus Dateien ab. `.npmrc` Wenn Sie eine Monorepo-App bereitstellen, die mit einem dieser Tools erstellt wurde, benötigen Sie eine `.npmrc` Datei in Ihrem Projekt-Stammverzeichnis.

Stellen Sie in der `.npmrc` Datei den Linker für die Installation von Node-Paketen auf ein. `hoisted` Sie können die folgende Zeile in Ihre Datei kopieren.

```
node-linker=hoisted
```

Weitere Informationen zu `.npmrc` Dateien und Einstellungen finden Sie unter [pnpm .npmrc](#) in der pnpm-Dokumentation.

Pnpm ist nicht im Standard-Build-Container von Amplify enthalten. Für pnpm workspace- und Turborepo-Apps müssen Sie in der `preBuild` Phase der Build-Einstellungen Ihrer App einen Befehl zur Installation von pnpm hinzufügen.

Der folgende Beispielauszug aus einer Build-Spezifikation zeigt eine `preBuild` Phase mit einem Befehl zur Installation von pnpm.

```
version: 1
applications:
  - frontend:
    phases:
      preBuild:
        commands:
          - npm install -g pnpm
```

Bereitstellungen mit Funktionsverzweigungen und Team-Workflows

Amplify Hosting wurde für die Verwendung mit Feature-Branched und GitFlow Workflows entwickelt. Amplify verwendet Git-Branched, um jedes Mal, wenn Sie einen neuen Branch in Ihrem Repository verbinden, ein neues Deployment zu erstellen. Nachdem Sie Ihren ersten Branch verbunden haben, erstellen Sie weitere Feature-Branched.

Um einer App einen Zweig hinzuzufügen

1. Wählen Sie die App aus, zu der Sie einen Zweig hinzufügen möchten.
2. Wählen Sie App-Einstellungen und dann Filialeinstellungen.
3. Wählen Sie auf der Seite mit den Filialeinstellungen die Option Filiale hinzufügen aus.
4. Wählen Sie einen Branch aus Ihrem Repository aus.
5. Wählen Sie Zweig hinzufügen.
6. Stellen Sie Ihre App erneut bereit.

Nachdem Sie einen Branch hinzugefügt haben, sind für Ihre App zwei Bereitstellungen in den Amplify-Standarddomänen verfügbar, z. B. <https://main.appid.amplifyapp.com> und <https://dev.appid.amplifyapp.com>. Dies kann davon abweichen team-to-team, aber in der Regel verfolgt der Hauptzweig den Veröffentlichungscode und ist Ihr Produktionszweig. Die develop-Verzweigung wird als Integrationsverzweigung zum Testen von neuen Funktionen verwendet. Auf diese Weise können Betatester unveröffentlichte Funktionen in der Bereitstellung in der Entwicklungsabteilung testen, ohne dass sich dies auf die Endanwender in der Hauptniederlassung auswirkt.

Themen

- [Team-Workflows mit Full-Stack-Apps von Amplify Gen 2](#)
- [Team-Workflows mit Full-Stack-Apps von Amplify Gen 1](#)
- [Musterbasierte Feature-Branch-Bereitstellungen](#)
- [Automatische Generierung der Amplify-Konfiguration während der Erstellung \(nur Gen 1-Apps\)](#)
- [Bedingte Backend-Builds \(nur Apps der Generation 1\)](#)
- [Verwenden Sie Amplify-Backends für mehrere Apps \(nur Gen-1-Apps\)](#)

Team-Workflows mit Full-Stack-Apps von Amplify Gen 2

AWS Amplify Gen 2 bietet ein TypeScript basiertes Entwicklererlebnis, bei dem der Code an erster Stelle steht, um Backends zu definieren. Weitere Informationen zu Fullstack-Workflows mit Amplify Gen 2-Anwendungen finden Sie unter [Fullstack-Workflows](#) in den Amplify-Dokumenten.

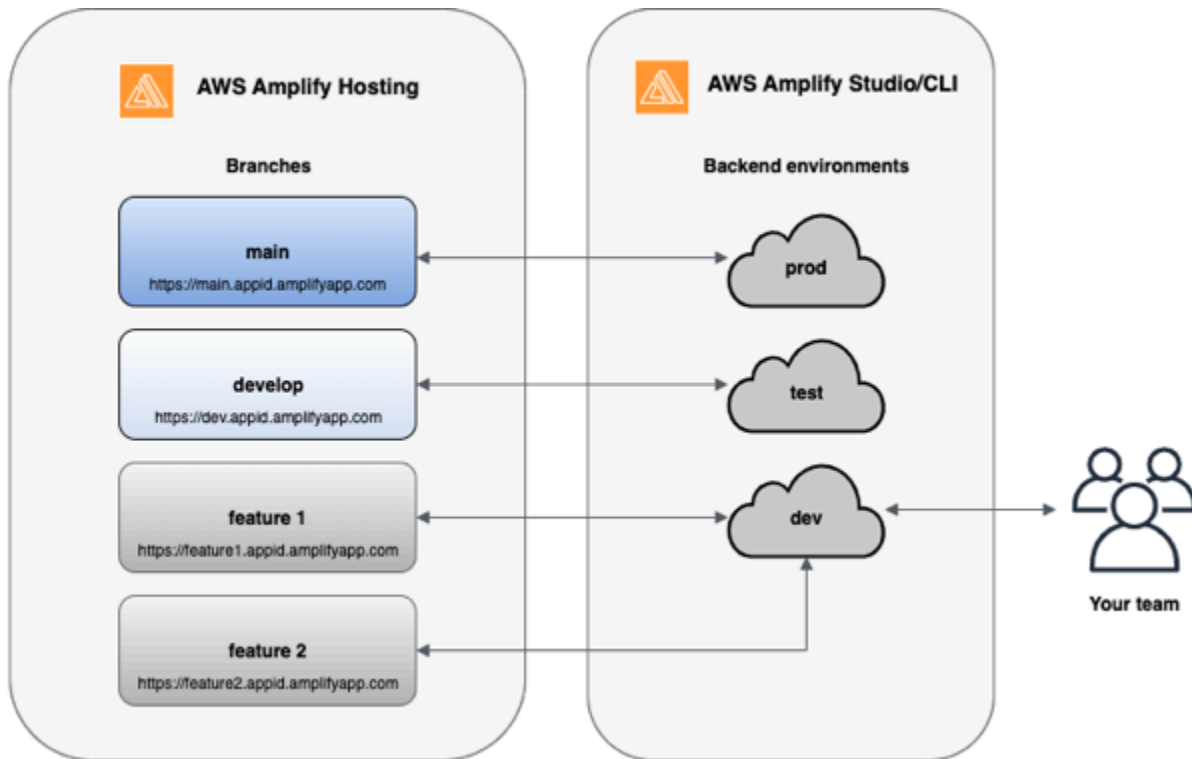
Team-Workflows mit Full-Stack-Apps von Amplify Gen 1

Eine Feature-Branch-Bereitstellung besteht aus einem Frontend und einer optionalen Backend-Umgebung. Das Frontend wird erstellt und in einem globalen Content Delivery Network (CDN) bereitgestellt, während das Backend von Amplify Studio oder der Amplify CLI bereitgestellt wird. AWS Informationen zur Einrichtung dieses Bereitstellungsszenarios finden Sie unter [Ein Backend für eine Anwendung erstellen](#)

Amplify Hosting stellt kontinuierlich Backend-Ressourcen wie GraphQL-APIs und Lambda-Funktionen mit Ihren Feature-Branch-Bereitstellungen bereit. Sie können die folgenden Verzweigungsmodelle verwenden, um Ihr Backend und Frontend mit Amplify Hosting bereitzustellen.

Workflow für Funktionsverzweigungen

- Erstellen Sie mit Amplify Studio oder der Amplify CLI Backend-Umgebungen für Produktion, Test und Entwicklung.
- Ordnen Sie das Prod-Backend dem Hauptzweig zu.
- Ordnen Sie das Test-Backend dem Entwicklungszweig zu.
- Teammitglieder können die Dev-Backend-Umgebung zum Testen einzelner Feature-Branches verwenden.



1. Installieren Sie die Amplify-CLI, um ein neues Amplify-Projekt zu starten.

```
npm install -g @aws-amplify/cli
```

2. Initialisieren Sie eine prod-Backend-Umgebung für Ihr Projekt. Wenn Sie kein Projekt haben, erstellen Sie eines mit Bootstrap-Tools wie create-react-app oder Gatsby.

```
create-react-app next-unicorn
cd next-unicorn
amplify init
? Do you want to use an existing environment? (Y/n): n
? Enter a name for the environment: prod
...
amplify push
```

3. Fügen Sie test- und dev-Backend-Umgebungen hinzu.

```
amplify env add
? Do you want to use an existing environment? (Y/n): n
? Enter a name for the environment: test
...
amplify push
```



```
amplify env add
? Do you want to use an existing environment? (Y/n): n
? Enter a name for the environment: dev
...
amplify push
```

4. Push Code in ein Git-Repository deiner Wahl (in diesem Beispiel gehen wir davon aus, dass du auf Main gepusht hast).

```
git commit -am 'Added dev, test, and prod environments'
git push origin main
```

5. Besuchen Sie Amplify in der AWS Management Console , um Ihre aktuelle Backend-Umgebung zu sehen. Navigieren Sie vom Breadcrumb aus eine Ebene nach oben, um eine Liste aller Backend-Umgebungen anzuzeigen, die auf der Registerkarte Backend-Umgebungen erstellt wurden.


quick-notes

The app homepage lists all deployed frontend and backend environments.

Frontend environments | **Backend environments**

Each backend environment is a container for all of the cloud capabilities added to your app. An Amplify backend environment contains the list of categories enabled such as API, auth, and storage.

prod



Categories added


- Authentication
- API

Deployment status

✔ Deployment completed 11/14/2019, 11:29:07 AM

▶ Edit backend

test



Categories added


- Authentication
- API

Deployment status

✔ Deployment completed 11/14/2019, 11:29:07 AM

▶ Edit backend

dev



Categories added

- Authentication
- API

Deployment status

✔ Deployment completed 11/14/2019, 11:29:07 AM

▶ Edit backend

6. Wechseln Sie zur Registerkarte Frontend-Umgebungen und verbinden Sie Ihren Repository-Anbieter und Ihren Hauptzweig.
7. Wählen Sie im Bildschirm mit den Build-Einstellungen eine bestehende Backend-Umgebung aus, um eine kontinuierliche Bereitstellung mit dem Hauptzweig einzurichten. Wählen Sie prod aus der

Drop-down-Liste aus und weisen Sie Amplify die Servicerolle zu. Wählen Sie Save and deploy (Speichern und Bereitstellen) aus. Nach Abschluss des Builds erhalten Sie eine Bereitstellung in der Hauptabteilung, die unter <https://main.appid.amplifyapp.com> verfügbar ist.

Configure build settings

App build settings

App name
Pick a name for your app.

Name cannot contain periods

Existing Amplify backend detected
Connect your backend to continuously deploy changes to both your frontend and backend

Would you like Amplify Console to deploy changes to these resources with your frontend?

Yes - choose an existing environment or create a new one

Create new environment

Select a backend environment for this branch:


- dev
- test
- prod

8. Connect den Entwicklungszweig in Amplify (gehen Sie davon aus, dass Develop und Main Branch zu diesem Zeitpunkt identisch sind). Wählen Sie die test-Backend-Umgebung aus.

Add repository branch

AWS CodeCommit

Repository service provider

 AWS CodeCommit

Branch
Select a branch from your repository.

develop

Backend environment
Select a backend environment for this branch.

test

9. Amplify ist jetzt eingerichtet. Sie können jetzt neue Funktionen in einer Funktionsverzweigung verwenden. Fügen Sie Backend-Funktionen mithilfe der dev-Backend-Umgebung aus Ihrer lokalen Workstation hinzu.

```
git checkout -b newinternet
amplify env checkout dev
amplify add api
...
amplify push
```

10. Wenn die Arbeit mit der Funktion abgeschlossen ist, führen Sie einen Commit für Ihren Code durch und erstellen Sie eine Pull-Anforderung zur internen Überprüfung.

```
git commit -am 'Decentralized internet v0.1'
git push origin newinternet
```

11. Um eine Vorschau der Änderungen anzuzeigen, rufen Sie die Amplify-Konsole auf und verbinden Sie Ihren Feature-Branch. Hinweis: Wenn Sie das auf Ihrem System AWS CLI installiert haben (nicht die Amplify-CLI), können Sie eine Filiale direkt von Ihrem Terminal aus verbinden. Um nach Ihrer appid zu suchen, gehen Sie zu App settings > General > AppARN (App-Einstellungen > Allgemein > AppARN: arn:aws:amplify:<region>:<region>:apps/<appid>

```
aws amplify create-branch --app-id <appid> --branch-name <branchname>
aws amplify start-job --app-id <appid> --branch-name <branchname> --job-type RELEASE
```

12. Sie können Ihre Funktion über <https://newinternet.appid.amplifyapp.com> für Ihre Teamkollegen freigeben. Wenn alles korrekt ist, führen Sie die Pull-Anforderung mit der develop-Verzweigung zusammen.

```
git checkout develop
git merge newinternet
git push
```

13. Damit wird ein Build gestartet, der sowohl das Backend als auch das Frontend in Amplify mit einer Filialbereitstellung unter <https://dev.appid.amplifyapp.com> aktualisiert. Sie können diesen Link für interne Beteiligte freigeben, damit sie die neue Funktion überprüfen können.

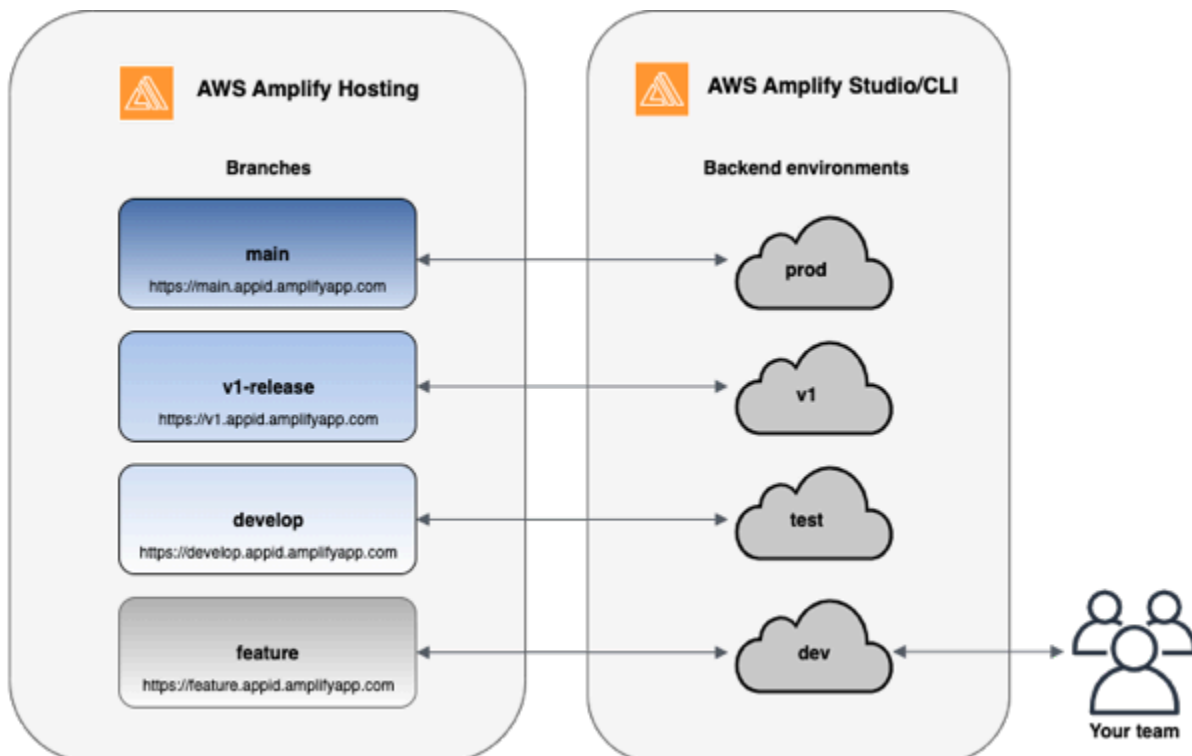
14. Löschen Sie Ihren Feature-Branch aus Git, Amplify und entfernen Sie die Backend-Umgebung aus der Cloud (Sie können jederzeit eine neue einrichten, indem Sie „amplify env checkout prod“ ausführen und „amplify env add“ ausführen).

```
git push origin --delete newinternet
aws amplify delete-branch --app-id <appid> --branch-name <branchname>
amplify env remove dev
```

GitFlow Arbeitsablauf

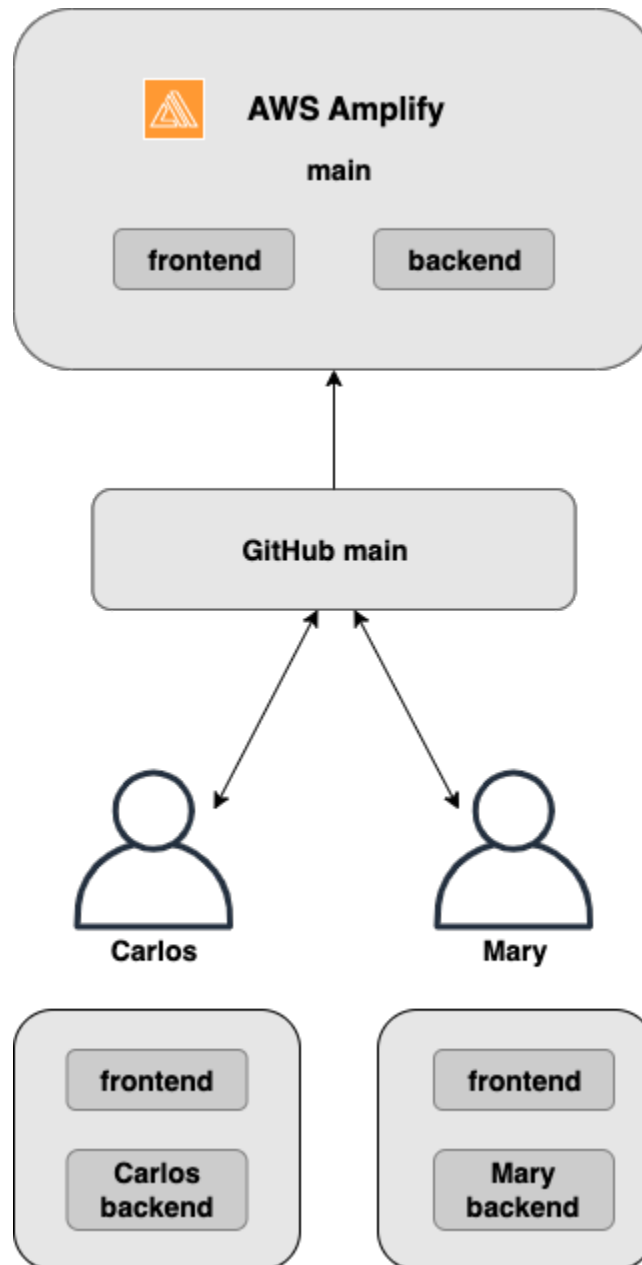
GitFlow verwendet zwei Zweige, um den Verlauf des Projekts aufzuzeichnen. Der Hauptzweig verfolgt nur den Release-Code, und der Entwicklungszweig wird als Integrationszweig für neue Funktionen verwendet. GitFlow vereinfacht die parallel Entwicklung, indem neue Entwicklungen von abgeschlossenen Arbeiten isoliert werden. Neue Entwicklungen (z. B. Funktionen und normale Fehlerbehebungen) erfolgen in feature-Verzweigungen. Wenn der Entwickler den Code zur Veröffentlichung freigeben möchte, wird die feature-Verzweigung wieder mit der develop-Verzweigung zur Integration zusammengeführt. Die einzigen Commits für den Hauptzweig sind Zusammenführungen von Release-Branched und Hotfix-Branched (um Fehler im Notfall zu beheben).

Das folgende Diagramm zeigt ein empfohlenes Setup mit GitFlow. Führen Sie einfach die im obigen Abschnitt „Workflow für Funktionsverzweigungen“ beschriebenen Schritte aus.



Sandbox pro Entwickler

- Jeder Entwickler in einem Team erstellt eine Sandbox-Umgebung in der Cloud, die vom lokalen Computer unabhängig ist. Auf diese Weise können Entwickler isoliert voneinander arbeiten, ohne die Änderungen anderer Teammitglieder zu überschreiben.
- Jeder Zweig in Amplify hat sein eigenes Backend. Dadurch wird sichergestellt, dass Amplify das Git-Repository als zentrale Informationsquelle für die Implementierung von Änderungen verwendet, anstatt sich darauf zu verlassen, dass Entwickler im Team ihr Backend oder Frontend manuell von ihren lokalen Computern aus in die Produktion bringen.



1. Installieren Sie die Amplify-CLI, um ein neues Amplify-Projekt zu starten.

```
npm install -g @aws-amplify/cli
```

2. Initialisieren Sie eine Mary-Backend-Umgebung für Ihr Projekt. Wenn Sie kein Projekt haben, erstellen Sie eines mit Bootstrap-Tools wie create-react-app oder Gatsby.

```
cd next-unicorn
amplify init
? Do you want to use an existing environment? (Y/n): n
```

```
? Enter a name for the environment: mary
...
amplify push
```

3. Push Code in ein Git-Repository deiner Wahl (in diesem Beispiel gehen wir davon aus, dass du auf Main gepusht hast).

```
git commit -am 'Added mary sandbox'
git push origin main
```

4. Connect dein Repo > Main mit Amplify.
5. Die Amplify-Konsole erkennt Backend-Umgebungen, die von der Amplify-CLI erstellt wurden. Wählen Sie in der Dropdownliste die Option Neue Umgebung erstellen aus und weisen Sie Amplify die Servicerolle zu. Wählen Sie Save and deploy (Speichern und Bereitstellen) aus. Nach Abschluss des Builds erhalten Sie unter <https://main.appid.amplifyapp.com> eine Bereitstellung für die Hauptniederlassung mit einer neuen Backend-Umgebung, die mit der Filiale verknüpft ist.
6. Connect den Entwicklungszweig in Amplify (gehen Sie davon aus, dass Develop und Main Branch zu diesem Zeitpunkt identisch sind) und wählen Sie Create

Musterbasierte Feature-Branch-Bereitstellungen

Mit musterbasierten Branch-Bereitstellungen können Sie automatisch Branches, die einem bestimmten Muster entsprechen, für Amplify bereitstellen. Produktteams, die Feature Branch oder GitFlow Workflows für ihre Releases verwenden, können jetzt Muster wie „release***“ definieren, um Git-Banches, die mit „release“ beginnen, automatisch auf einer gemeinsam nutzbaren URL bereitzustellen. In [diesem Blogpost](#) wird beschrieben, wie diese Funktion mit verschiedenen Team-Workflows verwendet wird.

1. Wähle „App-Einstellungen“ > „Branch-Einstellungen“ > „Bearbeiten“.
2. Wählen Sie Automatische Branch-Erkennung, um Zweige, die einem Mustersatz entsprechen, automatisch mit Amplify zu verbinden.
3. Geben Sie im Feld Automatische Branch-Erkennung — Muster die Muster für die automatische Bereitstellung von Zweigen ein.
 - *— Stellt alle Branches in Ihrem Repository bereit.
 - **release***— Stellt alle Branches bereit, die mit dem Wort „release“ beginnen.
 - **release*/**— Stellt alle Branches bereit, die einem 'release /'-Muster entsprechen.

- Geben Sie mehrere Muster in einer durch Kommas getrennten Liste an. z. B. `release*`, `feature*`.
4. Richten Sie den automatischen Kennwortschutz für alle Zweige ein, die automatisch erstellt werden, indem Sie die Zugriffskontrolle für automatische Branches auswählen.
 5. Für Gen-1-Anwendungen, die mit einem Amplify-Backend erstellt wurden, können Sie wählen, ob Sie für jede verbundene Filiale eine neue Umgebung erstellen oder alle Zweige auf ein vorhandenes Backend verweisen möchten.
 6. Wählen Sie Speichern.

Musterbasierte Feature-Branch-Bereitstellungen für eine App, die mit einer benutzerdefinierten Domain verbunden ist

Sie können musterbasierte Feature-Branch-Bereitstellungen für eine App verwenden, die mit einer benutzerdefinierten Amazon Route 53-Domain verbunden ist.

- Anweisungen zur Einrichtung musterbasierter Feature-Branch-Bereitstellungen finden Sie unter [Automatische Subdomains für eine benutzerdefinierte Amazon Route 53-Domain einrichten](#)
- Anweisungen zum Verbinden einer Amplify-App mit einer benutzerdefinierten Domain, die in Route 53 verwaltet wird, finden Sie unter [Fügen Sie eine benutzerdefinierte Domain hinzu, die von Amazon Route 53 verwaltet wird](#)
- Weitere Informationen zur Verwendung von Route 53 finden Sie unter [Was ist Amazon Route 53](#)

Automatische Generierung der Amplify-Konfiguration während der Erstellung (nur Gen 1-Apps)

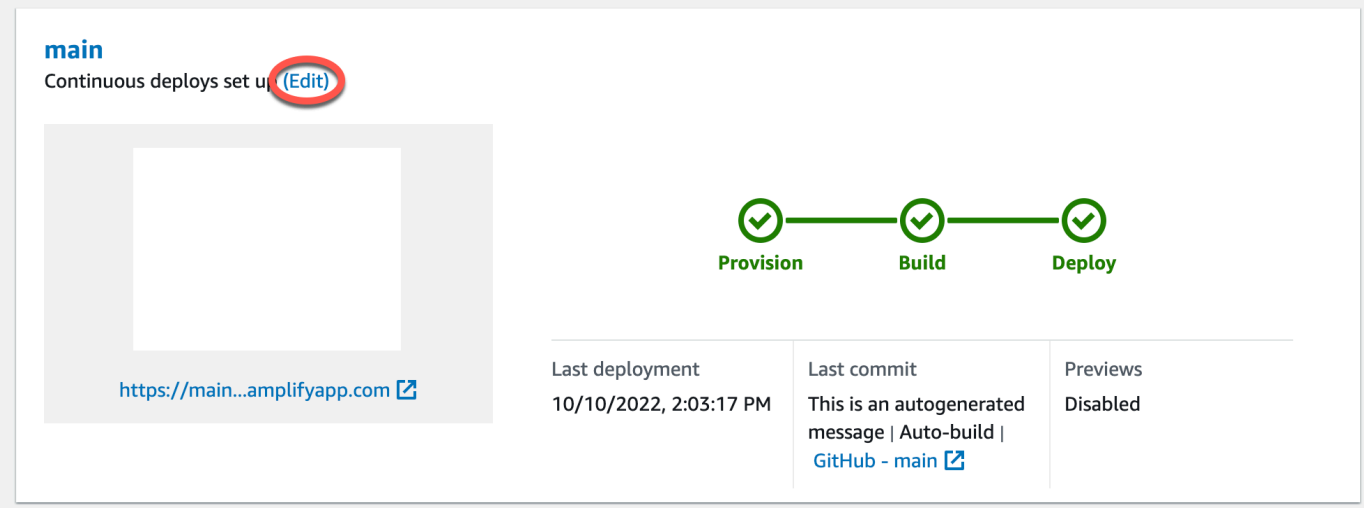
Note

Die Informationen in diesem Abschnitt gelten nur für Apps der 1. Generation. Wenn Sie automatisch Infrastruktur- und Anwendungscodeänderungen aus Feature-Banches für eine Gen-2-App bereitstellen möchten, finden Sie in den Amplify-Dokumenten weitere Informationen unter [Fullstack-Branch-Bereitstellungen](#).

Amplify unterstützt die automatische Build-Time-Generierung der `aws-exports.js` Amplify-Konfigurationsdatei für Gen-1-Apps. Indem Sie Full-Stack-CI/CD-Bereitstellungen deaktivieren, ermöglichen Sie Ihrer App, die `aws-exports.js` Datei automatisch zu generieren und sicherzustellen, dass zur Build-Zeit keine Aktualisierungen an Ihrem Backend vorgenommen werden.

Zur automatischen Generierung während der Erstellung `aws-exports.js`

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, die Sie bearbeiten möchten.
3. Wählen Sie den Tab Hosting-Umgebungen.
4. Suchen Sie den Zweig, den Sie bearbeiten möchten, und wählen Sie Bearbeiten.



Last deployment	Last commit	Previews
10/10/2022, 2:03:17 PM	This is an autogenerated message Auto-build GitHub - main	Disabled

5. Deaktivieren Sie auf der Seite Ziel-Backend bearbeiten die Option Enable Full-Stack Continuous Deployments (CI/CD), um Full-Stack-CI/CD für dieses Backend zu deaktivieren.

Edit target backend

Select a backend environment to use with this branch

App name

Example-Amplify-App (this app) ▼

Environment

dev ▼

Enable full-stack continuous deployments (CI/CD)

Full-stack CI/CD allows you to continuously deploy frontend and backend changes on every code commit

- Wählen Sie eine bestehende Servicerolle aus, um Amplify die erforderlichen Berechtigungen zu erteilen, um Änderungen an Ihrem App-Backend vorzunehmen. Wenn Sie eine Servicerolle erstellen müssen, wählen Sie Neue Rolle erstellen. Weitere Informationen zum Erstellen einer Servicerolle finden Sie unter [Eine Servicerolle hinzufügen](#).
- Wählen Sie Speichern. Amplify wendet diese Änderungen an, wenn Sie die App das nächste Mal erstellen.

Bedingte Backend-Builds (nur Apps der Generation 1)

Note

Die Informationen in diesem Abschnitt gelten nur für Apps der 1. Generation. Amplify Gen 2 bietet ein TypeScript basiertes Entwicklererlebnis, bei dem der Code an erster Stelle steht. Daher ist diese Funktion für Backends der zweiten Generation nicht erforderlich.

Amplify unterstützt bedingte Backend-Builds für alle Branches in einer Gen-1-App. Um bedingte Backend-Builds zu konfigurieren, setzen Sie die `AMPLIFY_DIFF_BACKEND` Umgebungsvariable auf `true`. Durch die Aktivierung bedingter Backend-Builds können Builds beschleunigt werden, bei denen Änderungen nur am Frontend vorgenommen werden.

Wenn Sie diff-basierte Backend-Builds aktivieren, versucht Amplify zu Beginn jedes Builds, einen Diff für den `amplify` Ordner in Ihrem Repository auszuführen. Wenn Amplify keine Unterschiede feststellt, überspringt es den Backend-Build-Schritt und aktualisiert Ihre Backend-Ressourcen nicht. Wenn Ihr Projekt keinen `amplify` Ordner in Ihrem Repository hat, ignoriert Amplify den Wert der `AMPLIFY_DIFF_BACKEND` Umgebungsvariablen. Anweisungen zum Einstellen der `AMPLIFY_DIFF_BACKEND` Umgebungsvariablen finden Sie unter [Aktivieren oder deaktivieren Sie diff-basierte Backend-Builds für eine Gen-1-App](#)

Wenn Sie derzeit benutzerdefinierte Befehle in den Build-Einstellungen Ihrer Backend-Phase angegeben haben, funktionieren bedingte Backend-Builds nicht. Wenn Sie möchten, dass diese benutzerdefinierten Befehle ausgeführt werden, müssen Sie sie in der Datei Ihrer App in die Frontend-Phase Ihrer Build-Einstellungen verschieben. `amplify.yml` Weitere Informationen zum Aktualisieren der `amplify.yml` Datei finden Sie unter [Befehle und Einstellungen für Build-Spezifikationen](#).

Verwenden Sie Amplify-Backends für mehrere Apps (nur Gen-1-Apps)

Note

Die Informationen in diesem Abschnitt gelten nur für Apps der 1. Generation. Wenn Sie Backend-Ressourcen für eine Gen 2-App gemeinsam nutzen möchten, finden Sie in den Amplify-Dokumenten weitere Informationen [unter Ressourcen branchenübergreifend teilen](#)

Mit Amplify können Sie bestehende Backend-Umgebungen für all Ihre Gen 1-Apps in einer bestimmten Region wiederverwenden. Sie können dies tun, wenn Sie eine neue App erstellen, einen neuen Zweig mit einer vorhandenen App verbinden oder ein vorhandenes Frontend so aktualisieren, dass es auf eine andere Backend-Umgebung verweist.

Backends wiederverwenden, wenn Sie eine neue App erstellen

Um ein Backend wiederzuverwenden, wenn Sie eine neue Amplify-App erstellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Gehen Sie wie folgt vor, um ein neues Backend für dieses Beispiel zu erstellen:
 - a. Wählen Sie im Navigationsbereich Alle Apps aus.
 - b. Wählen Sie Neue App, App erstellen aus.
 - c. Geben Sie einen Namen für Ihre App ein, z. **Example-Amplify-App B**.
 - d. Wählen Sie Bereitstellung bestätigen aus.
3. Um ein Frontend mit Ihrem neuen Backend zu verbinden, wählen Sie den Tab Hosting-Umgebungen.
4. Wähle deinen Git-Anbieter und dann Connect branch.
5. Wähle auf der Seite Repository-Zweig hinzufügen für Kürzlich aktualisierte Repositorys deinen Repository-Namen aus. Wählen Sie für Branch den Branch aus Ihrem Repository aus, zu dem Sie eine Verbindung herstellen möchten.
6. Gehen Sie auf der Seite mit den Build-Einstellungen wie folgt vor:

- a. Wählen Sie unter App-Name die App aus, die zum Hinzufügen einer Backend-Umgebung verwendet werden soll. Sie können die aktuelle App oder eine andere App in der aktuellen Region auswählen.
 - b. Wählen Sie unter Umgebung den Namen der Backend-Umgebung aus, die Sie hinzufügen möchten. Sie können eine bestehende Umgebung verwenden oder eine neue erstellen.
 - c. Standardmäßig ist Full-Stack-CI/CD ausgeschaltet. Wenn Sie Full-Stack-CI/CD ausschalten, wird die App nur im Pull-Only-Modus ausgeführt. Zur Build-Zeit generiert Amplify automatisch nur die `aws-exports.js` Datei, ohne Ihre Backend-Umgebung zu ändern.
 - d. Wählen Sie eine bestehende Servicerolle aus, um Amplify die erforderlichen Berechtigungen zu erteilen, um Änderungen an Ihrem App-Backend vorzunehmen. Wenn Sie eine Servicerolle erstellen müssen, wählen Sie Neue Rolle erstellen. Weitere Informationen zum Erstellen einer Servicerolle finden Sie unter [Eine Servicerolle hinzufügen](#).
 - e. Wählen Sie Weiter aus.
7. Wählen Sie Save and deploy (Speichern und Bereitstellen) aus.

Verwenden Sie Backends wieder, wenn Sie eine Filiale mit einer vorhandenen App verbinden

Um ein Backend wiederzuverwenden, wenn Sie eine Filiale mit einer vorhandenen Amplify-App verbinden

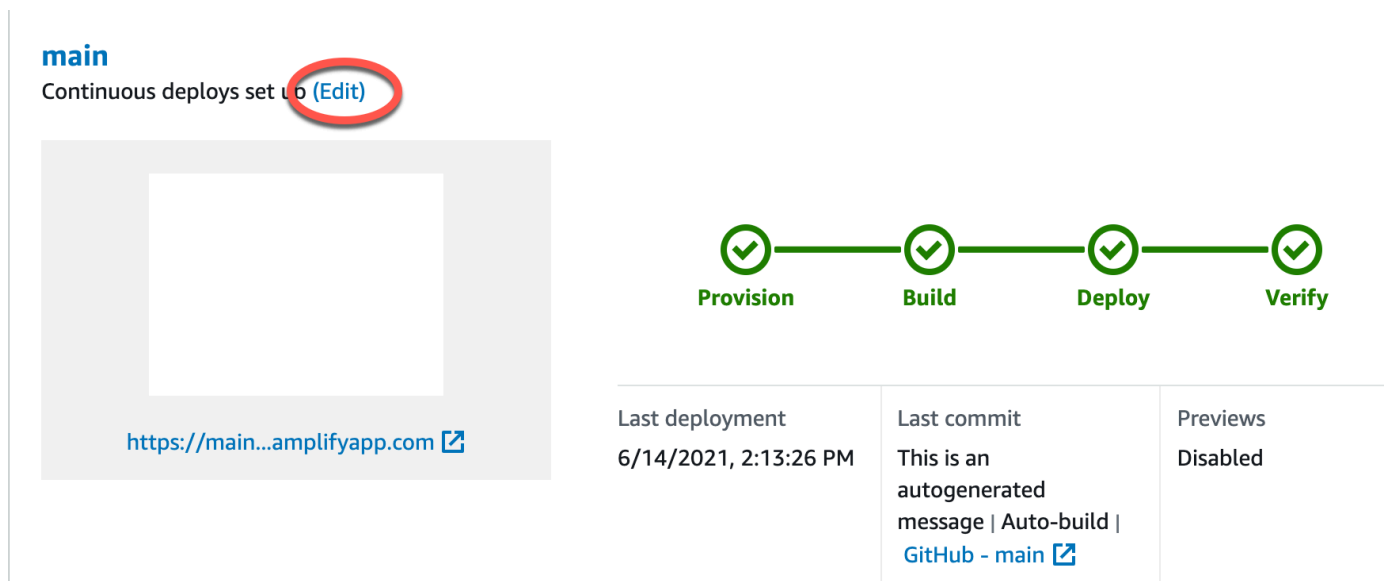
1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, mit der Sie eine neue Filiale verbinden möchten.
3. Wählen Sie im Navigationsbereich App-Einstellungen, Allgemein aus.
4. Wählen Sie im Abschnitt Filialen die Option Filiale Connect aus.
5. Wählen Sie auf der Seite Repository-Zweig hinzufügen für Branch den Branch aus Ihrem Repository aus, zu dem Sie eine Verbindung herstellen möchten.
6. Wählen Sie als App-Name die App aus, die zum Hinzufügen einer Backend-Umgebung verwendet werden soll. Sie können die aktuelle App oder eine andere App in der aktuellen Region auswählen.
7. Wählen Sie unter Umgebung den Namen der Backend-Umgebung aus, die Sie hinzufügen möchten. Sie können eine bestehende Umgebung verwenden oder eine neue erstellen.

8. Wenn Sie eine Servicerolle einrichten müssen, um Amplify die erforderlichen Berechtigungen zu erteilen, um Änderungen an Ihrem App-Backend vorzunehmen, werden Sie von der Konsole aufgefordert, diese Aufgabe auszuführen. Weitere Informationen zum Erstellen einer Servicerolle finden Sie unter [Eine Servicerolle hinzufügen](#).
9. Standardmäßig ist Full-Stack-CI/CD ausgeschaltet. Wenn Sie Full-Stack-CI/CD ausschalten, wird die App nur im Pull-Only-Modus ausgeführt. Zur Build-Zeit generiert Amplify automatisch nur die `aws-exports.js` Datei, ohne Ihre Backend-Umgebung zu ändern.
10. Wählen Sie Weiter aus.
11. Wählen Sie Save and deploy (Speichern und Bereitstellen) aus.

Bearbeiten Sie ein vorhandenes Frontend so, dass es auf ein anderes Backend verweist

Um eine Frontend-Amplify-App so zu bearbeiten, dass sie auf ein anderes Backend verweist

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die Sie das Backend bearbeiten möchten.
3. Wählen Sie den Tab Hosting-Umgebungen.
4. Suchen Sie den Zweig, den Sie bearbeiten möchten, und wählen Sie Bearbeiten.



Last deployment 6/14/2021, 2:13:26 PM	Last commit This is an autogenerated message Auto-build GitHub - main	Previews Disabled
--	--	----------------------

5. Wählen Sie auf der Seite Wählen Sie eine Backend-Umgebung aus, die mit diesem Zweig verwendet werden soll, als App-Name die Frontend-App aus, für die Sie die Backend-Umgebung

bearbeiten möchten. Sie können die aktuelle App oder eine andere App in der aktuellen Region auswählen.

6. Wählen Sie unter Backend-Umgebung den Namen der Backend-Umgebung aus, die hinzugefügt werden soll.
7. Standardmäßig ist Full-Stack-CI/CD aktiviert. Deaktivieren Sie diese Option, um Full-Stack-CI/CD für dieses Backend auszuschalten. Wenn Sie Full-Stack-CI/CD ausschalten, wird die App nur im Pull-Only-Modus ausgeführt. Zur Build-Zeit generiert Amplify automatisch nur die `aws-exports.js` Datei, ohne die Backend-Umgebung zu ändern.
8. Wählen Sie Speichern. Amplify wendet diese Änderungen an, wenn Sie die App das nächste Mal erstellen.

Ein Backend für eine Anwendung erstellen

Mit können AWS Amplify Sie eine Fullstack-Anwendung mit Daten-, Authentifizierungs-, Speicher- und Frontend-Hosting erstellen, für die Sie bereitgestellt werden. AWS

AWS Amplify Gen 2 bietet ein TypeScript basiertes Entwicklererlebnis, bei dem der Code an erster Stelle steht, um Backends zu definieren. Informationen dazu, wie Sie mit Amplify Gen 2 ein Backend erstellen und mit Ihrer App verbinden, finden Sie unter Backend [erstellen und verbinden](#) in den Amplify-Dokumenten.

Wenn Sie nach der Dokumentation zum Erstellen eines Backends für eine Gen 1-App mithilfe der CLI und Amplify Studio suchen, finden Sie in den Gen 1 Amplify-Dokumenten unter [Build & Connect Backend](#).

Themen

- [Erstellen Sie ein Backend für eine Gen 2-App](#)
- [Erstellen Sie ein Backend für eine Gen 1-App](#)

Erstellen Sie ein Backend für eine Gen 2-App

Ein Tutorial, das Sie durch die Schritte zur Erstellung einer Amplify Gen 2-Fullstack-Anwendung mit einem TypeScript basierten Backend führt, finden [Sie unter Erste](#) Schritte in den Amplify-Dokumenten.

Erstellen Sie ein Backend für eine Gen 1-App

In diesem Tutorial richten Sie einen Full-Stack-CI/CD-Workflow mit Amplify ein. Sie werden eine Frontend-App für Amplify Hosting bereitstellen. Anschließend erstellen Sie mit Amplify Studio ein Backend. Schließlich verbinden Sie das Cloud-Backend mit der Frontend-App.

Voraussetzungen

Bevor Sie mit diesem Tutorial beginnen, müssen Sie die folgenden Voraussetzungen erfüllen.

Melden Sie sich für eine an AWS-Konto

Wenn Sie noch kein AWS Kunde sind, müssen Sie [einen erstellen](#), AWS-Konto indem Sie den Online-Anweisungen folgen. Durch die Registrierung können Sie auf Amplify und andere AWS Dienste zugreifen, die Sie mit Ihrer Anwendung verwenden können.

Erstellen Sie ein Git-Repository

Amplify unterstützt GitHub, Bitbucket GitLab, und. AWS CodeCommit Schieben Sie Ihre Anwendung in Ihr Git-Repository.

Installieren Sie die Amplify Command Line Interface (CLI)

Anweisungen finden [Sie unter Installieren der Amplify CLI](#) in der Amplify Framework-Dokumentation.

Schritt 1: Stellen Sie ein Frontend bereit

Wenn Sie eine bestehende Frontend-App in einem Git-Repository haben, die Sie für dieses Beispiel verwenden möchten, können Sie mit den Anweisungen zur Bereitstellung einer Frontend-App fortfahren.

Wenn Sie eine neue Frontend-App für dieses Beispiel erstellen müssen, können Sie den Anweisungen [Create React App in der Dokumentation Create React App](#) folgen.

Um eine Frontend-App bereitzustellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie auf der Seite Alle Apps die Option Neue App und dann Web-App hosten in der oberen rechten Ecke aus.
3. Wähle deinen GitHub, Bitbucket- oder AWS CodeCommit Repository-Anbieter aus und wähle dann Weiter. GitLab
4. Amplify autorisiert den Zugriff auf Ihr Git-Repository. Für GitHub Repositorien verwendet Amplify jetzt die GitHub Apps-Funktion, um den Amplify-Zugriff zu autorisieren.

Weitere Informationen zur Installation und Autorisierung der App finden Sie unter. GitHub [Einrichtung des Amplify-Zugriffs auf GitHub Repositories](#)

5. Gehen Sie auf der Seite Repository-Zweig hinzufügen wie folgt vor:

- a. Wählen Sie in der Liste „Kürzlich aktualisierte Repositorys“ den Namen des Repositorys aus, mit dem Sie eine Verbindung herstellen möchten.
 - b. Wählen Sie in der Branch-Liste den Namen des Repository-Branche aus, zu dem Sie eine Verbindung herstellen möchten.
 - c. Wählen Sie Weiter aus.
6. Wählen Sie auf der Seite Build-Einstellungen konfigurieren die Option Weiter aus.
 7. Wählen Sie auf der Seite Überprüfen die Option Speichern und bereitstellen aus. Wenn die Bereitstellung abgeschlossen ist, können Sie Ihre App in der `amplifyapp.com` Standarddomäne anzeigen.

Note

Um die Sicherheit Ihrer Amplify-Anwendungen zu erhöhen, ist die Domain `amplifyapp.com` in der [Public Suffix List \(PSL\)](#) registriert. Aus Sicherheitsgründen empfehlen wir Ihnen, Cookies mit einem `__Host-` Präfix zu verwenden, falls Sie jemals sensible Cookies im Standard-Domainnamen für Ihre Amplify-Anwendungen einrichten müssen. Diese Vorgehensweise hilft Ihnen dabei, Ihre Domain vor CSRF-Versuchen (Cross-Site Request Forgery Attempts, Anforderungsfälschung zwischen Websites) zu schützen. Weitere Informationen finden Sie auf der [Set-Cookie](#)-Seite im Mozilla Developer Network.

Schritt 2: Erstellen Sie ein Backend

Nachdem Sie eine Frontend-App für Amplify Hosting bereitgestellt haben, können Sie ein Backend erstellen. Verwenden Sie die folgenden Anweisungen, um ein Backend mit einer einfachen Datenbank und einem GraphQL-API-Endpunkt zu erstellen.

Um ein Backend zu erstellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie auf der Seite Alle Apps die App aus, die Sie in Schritt 1 erstellt haben.
3. Wählen Sie auf der Startseite der App den Tab Backend-Umgebungen und dann Erste Schritte aus. Dadurch wird der Einrichtungsprozess für eine Standard-Staging-Umgebung eingeleitet.
4. Wählen Sie nach Abschluss der Einrichtung Launch Studio, um auf die Staging-Backend-Umgebung in Amplify Studio zuzugreifen.

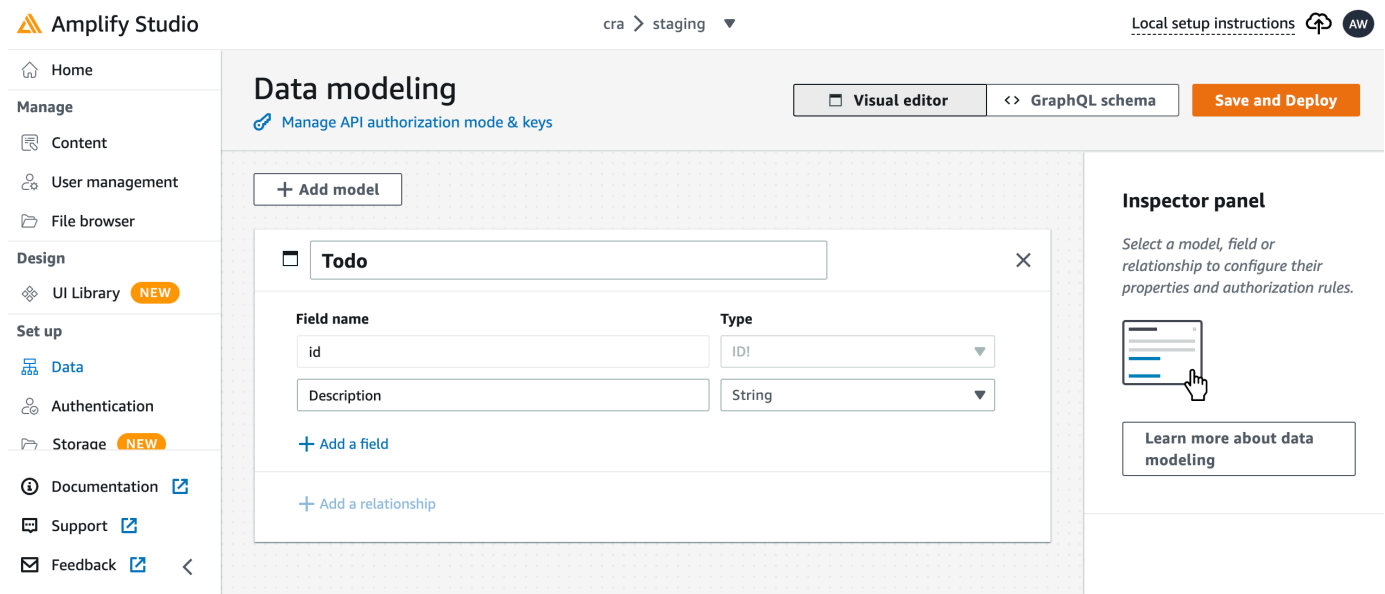
Amplify Studio ist eine visuelle Oberfläche, mit der Sie Ihr Backend erstellen und verwalten und die Entwicklung Ihrer Frontend-Benutzeroberfläche beschleunigen können. Weitere Informationen zu Amplify Studio finden Sie in der [Amplify Studio-Dokumentation](#).

Verwenden Sie die folgenden Anweisungen, um eine einfache Datenbank mit der Visual Backend Builder-Oberfläche von Amplify Studio zu erstellen.

Erstellen Sie ein Datenmodell

1. Wählen Sie auf der Startseite der Staging-Umgebung Ihrer App die Option Datenmodell erstellen aus. Dadurch wird der Datenmodell-Designer geöffnet.
2. Wählen Sie auf der Seite Datenmodellierung die Option Modell hinzufügen aus.
3. Geben Sie für den Titel ein **Todo**.
4. Wählen Sie Feld hinzufügen aus.
5. Geben Sie als Feldname ein **Description**.

Der folgende Screenshot ist ein Beispiel dafür, wie Ihr Datenmodell im Designer aussehen wird.



6. Wählen Sie Speichern und Bereitstellen.
7. Kehren Sie zur Amplify Hosting-Konsole zurück und die Bereitstellung der Staging-Umgebung wird fortgesetzt.

Während der Bereitstellung erstellt Amplify Studio alle erforderlichen AWS Ressourcen im Backend, einschließlich einer AWS AppSync GraphQL-API für den Datenzugriff und einer Amazon DynamoDB-

Tabelle zum Hosten der Todo-Elemente. Amplify verwendet AWS CloudFormation , um Ihr Backend bereitzustellen, sodass Sie Ihre Backend-Definition als speichern können. `infrastructure-as-code`

Schritt 3: Backend mit Frontend Connect

Nachdem Sie nun ein Frontend bereitgestellt und ein Cloud-Backend erstellt haben, das ein Datenmodell enthält, müssen Sie sie verbinden. Verwenden Sie die folgenden Anweisungen, um Ihre Backend-Definition mit der Amplify CLI in Ihr lokales App-Projekt zu übertragen.

Um ein Cloud-Backend mit einem lokalen Frontend zu verbinden

1. Öffnen Sie ein Terminalfenster und navigieren Sie zum Stammverzeichnis Ihres lokalen Projekts.
2. Führen Sie den folgenden Befehl im Terminalfenster aus und ersetzen Sie den roten Text durch die eindeutige App-ID und den Namen der Backend-Umgebung für Ihr Projekt.

```
amplify pull --appId abcd1234 --envName staging
```

3. Folgen Sie den Anweisungen im Terminalfenster, um die Einrichtung des Projekts abzuschließen.

Jetzt können Sie den Build-Prozess so konfigurieren, dass das Backend zum Workflow für die kontinuierliche Bereitstellung hinzugefügt wird. Verwenden Sie die folgenden Anweisungen, um einen Frontend-Zweig mit einem Backend in der Amplify Hosting-Konsole zu verbinden.

Um einen Frontend-App-Branch und ein Cloud-Backend zu verbinden

1. Wählen Sie auf der Startseite der App den Tab Hosting-Umgebungen aus.
2. Suchen Sie den Hauptzweig und wählen Sie Bearbeiten.

Last deployment	Last commit	Previews
10/10/2022, 2:03:17 PM	This is an autogenerated message Auto-build GitHub - main	Disabled

3. Wählen Sie im Fenster Ziel-Backend bearbeiten unter Umgebung den Namen des Backends aus, zu dem eine Verbindung hergestellt werden soll. Wählen Sie in diesem Beispiel das Staging-Backend aus, das Sie in Schritt 2 erstellt haben.

Standardmäßig ist Full-Stack-CI/CD aktiviert. Deaktivieren Sie diese Option, um Full-Stack-CI/CD für dieses Backend auszuschalten. Wenn Sie Full-Stack-CI/CD ausschalten, wird die App nur im Pull-Only-Modus ausgeführt. Zur Build-Zeit generiert Amplify automatisch nur die `aws-exports.js` Datei, ohne Ihre Backend-Umgebung zu ändern.

4. Als Nächstes müssen Sie eine Servicerolle einrichten, um Amplify die erforderlichen Berechtigungen zu erteilen, um Änderungen an Ihrem App-Backend vorzunehmen. Sie können entweder eine bestehende Servicerolle verwenden oder eine neue erstellen. Anweisungen finden Sie unter [Eine Servicerolle hinzufügen](#).
5. Nachdem Sie eine Servicerolle hinzugefügt haben, kehren Sie zum Fenster Ziel-Backend bearbeiten zurück und wählen Sie Speichern.
6. Um die Verbindung des Staging-Backends mit dem Hauptzweig der Frontend-App abzuschließen, führen Sie einen neuen Build Ihres Projekts durch.

Führen Sie eine der folgenden Aktionen aus:

- Pushen Sie aus Ihrem Git-Repository Code, um einen Build in der Amplify-Konsole zu initiieren.
- Navigieren Sie in der Amplify-Konsole zur Seite mit den Build-Details der App und wählen Sie Diese Version erneut bereitstellen.

Nächste Schritte

Richten Sie Feature-Branch-Bereitstellungen ein

Folgen Sie unserem empfohlenen Arbeitsablauf, um [Feature-Branch-Bereitstellungen mit mehreren Backend-Umgebungen einzurichten](#).

Erstellen Sie eine Frontend-Benutzeroberfläche in Amplify Studio

Verwenden Sie Studio, um Ihre Frontend-Benutzeroberfläche mit einer Reihe von ready-to-use UI-Komponenten zu erstellen, und verbinden Sie sie dann mit Ihrem App-Backend. Weitere Informationen und Tutorials finden Sie im Benutzerhandbuch für [Amplify Studio in der Amplify Framework-Dokumentation](#).

Manuelle Bereitstellungen

Manuelle Bereitstellungen ermöglichen es Ihnen, Ihre Web-App mit Amplify Hosting zu veröffentlichen, ohne einen Git-Anbieter zu verbinden. Sie können einen Ordner per Drag & Drop von Ihrem Desktop ziehen und Ihre Site in Sekundenschnelle hosten. Alternativ können Sie auf Assets in einem Amazon S3 S3-Bucket verweisen oder eine öffentliche URL zu dem Speicherort angeben, an dem Ihre Dateien gespeichert sind.

Für Amazon S3 können Sie auch AWS Lambda Auslöser einrichten, um Ihre Website jedes Mal zu aktualisieren, wenn neue Assets hochgeladen werden. Weitere Informationen zur [Einrichtung dieses Szenarios finden Sie im Blogbeitrag Bereitstellen von auf Amazon S3, Dropbox oder Ihrem Desktop auf der AWS Amplify Konsole gespeicherten Dateien](#).

Amplify Hosting unterstützt keine manuellen Bereitstellungen für serverseitig gerenderte (SSR) Apps. Weitere Informationen finden Sie unter [Stellen Sie serverseitig gerenderte Apps mit Amplify Hosting bereit](#).

Manuelles Bereitstellen per Drag-and-Drop

Um eine App manuell per Drag & Drop bereitzustellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie in der oberen rechten Ecke Neue App erstellen.
3. Wählen Sie auf der Seite Start building with Amplify die Option Deploy without Git aus. Wählen Sie anschließend Weiter.
4. Geben Sie im Abschnitt Manuelle Bereitstellung starten für App-Name den Namen Ihrer App ein.
5. Geben Sie unter Branchennamen einen aussagekräftigen Namen ein, z. B. **development** oder **production**.
6. Wählen Sie als Methode die Option Drag & Drop.
7. Ziehen Sie entweder einen Ordner per Drag & Drop von Ihrem Desktop in die Drop-Zone oder wählen Sie die Datei mit der Option „zip-Ordner auswählen“ von Ihrem Computer aus. Bei der Datei, die Sie ziehen und ablegen oder auswählen, muss es sich um einen ZIP-Ordner handeln, der den Inhalt Ihrer Build-Ausgabe enthält.
8. Wählen Sie Save and deploy (Speichern und Bereitstellen) aus.

Manuelles Bereitstellen von Amazon S3 oder URL

Um eine App manuell von Amazon S3 oder einer öffentlichen URL aus bereitzustellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie in der oberen rechten Ecke Neue App erstellen.
3. Wählen Sie auf der Seite Start building with Amplify die Option Deploy without Git aus. Wählen Sie anschließend Weiter.
4. Geben Sie im Abschnitt Manuelle Bereitstellung starten für App-Name den Namen Ihrer App ein.
5. Geben Sie unter Branchennamen einen aussagekräftigen Namen ein, z. B. **development** oder **production**.
6. Wählen Sie als Methode entweder Amazon S3 oder Beliebige URL aus.
7. Das Verfahren zum Hochladen Ihrer Dateien hängt von der Upload-Methode ab.
 - Amazon S3
 - a. Wählen Sie für Amazon S3 S3-Bucket den Namen des Amazon S3 S3-Buckets aus der Liste aus. Zugriffskontrolllisten (ACLs) müssen für den ausgewählten Bucket aktiviert sein. Weitere Informationen finden Sie unter [Fehlerbehebung beim Amazon S3 S3-Bucket-Zugriff](#).
 - b. Wählen Sie unter Zip-Datei den Namen der Zip-Datei aus, die bereitgestellt werden soll.
 - Beliebige URL
 - Geben Sie unter Ressourcen-URL die URL zu der komprimierten Datei ein, die bereitgestellt werden soll.
8. Wählen Sie Save and deploy (Speichern und Bereitstellen) aus.

Note

Achten Sie beim Erstellen des ZIP-Ordners darauf, dass Sie den Inhalt Ihrer Build-Ausgabe komprimieren und nicht den Ordner der obersten Ebene. Wenn Ihre Build-Ausgabe beispielsweise einen Ordner mit dem Namen „build“ oder „public“ generiert, navigieren Sie zunächst zu diesem Ordner, wählen Sie den gesamten Inhalt aus und komprimieren Sie ihn von dort aus. Wenn Sie dies nicht tun, wird die Fehlermeldung „Zugriff verweigert“ angezeigt, da das Stammverzeichnis der Site nicht ordnungsgemäß initialisiert wird.

Fehlerbehebung beim Amazon S3 S3-Bucket-Zugriff

Wenn Sie einen Amazon S3-Bucket erstellen, verwenden Sie dessen Einstellung Amazon S3 Object Ownership, um zu steuern, ob Zugriffskontrolllisten (ACLs) für den Bucket aktiviert oder deaktiviert sind. Um eine App manuell aus einem Amazon S3 S3-Bucket auf Amplify bereitzustellen, müssen ACLs für den Bucket aktiviert sein.

Wenn Sie bei der Bereitstellung aus einem Amazon S3 S3-Bucket eine `AccessControlList` Fehlermeldung erhalten, wurde der Bucket mit deaktivierten ACLs erstellt und Sie müssen sie in der Amazon S3 S3-Konsole aktivieren. Anweisungen finden Sie unter [Objektbesitz für einen vorhandenen Bucket festlegen](#) im Amazon Simple Storage Service-Benutzerhandbuch.

Schaltfläche „Auf Amplify bereitstellen“

Mit der Schaltfläche Deploy to Amplify Hosting können Sie GitHub Projekte öffentlich oder innerhalb Ihres Teams teilen. Das Folgende ist ein Bild der Schaltfläche:



Fügen Sie die Schaltfläche Deploy to Amplify Hosting zu einem Repository oder Blog hinzu

Fügen Sie die Schaltfläche zu Ihrer GitHub README.md-Datei, Ihrem Blogbeitrag oder einer anderen Markup-Seite hinzu, die HTML rendert. Die Schaltfläche besteht aus den folgenden zwei Komponenten:

1. Ein SVG-Bild befindet sich unter der URL `https://oneclick.amplifyapp.com/button.svg`
2. Die URL der Amplify-Konsole mit einem Link zu Ihrem GitHub Repository. Sie können entweder die URL Ihres Repositories kopieren `https://github.com/username/repository`, z. B., oder Sie können einen Deep-Link in einen bestimmten Ordner bereitstellen, z. `https://github.com/username/repository/tree/branchname/folder` Amplify Hosting stellt den Standard-Branch in Ihrem Repository bereit. Nach dem Verknüpfen der App können weitere Verzweigungen verknüpft werden.

Verwenden Sie das folgende Beispiel, um die Schaltfläche zu einer Markdown-Datei wie Ihrer GitHub README.md hinzuzufügen. Ersetze es `https://github.com/username/repository` durch die URL zu deinem Repository.

```
[![amplifybutton](https://oneclick.amplifyapp.com/button.svg)](https://console.aws.amazon.com/amplify/home#/deploy?repo=https://github.com/username/repository)
```

Verwenden Sie das folgende Beispiel, um die Schaltfläche zu einem beliebigen HTML-Dokument hinzuzufügen. `https://github.com/username/repository` Ersetzen Sie es durch die URL zu Ihrem Repository.

```
<a href="https://console.aws.amazon.com/amplify/home#/deploy?repo=https://github.com/username/repository">
```

```

</a>
```

Einrichtung des Amplify-Zugriffs auf GitHub Repositorys

Amplify verwendet jetzt die GitHub Apps-Funktion, um den Nur-Lesezugriff von Amplify auf GitHub Repositorys zu autorisieren. Mit der Amplify GitHub App sind die Berechtigungen genauer abgestimmt, sodass Sie Amplify nur Zugriff auf die von Ihnen angegebenen Repositorys gewähren können. Weitere Informationen zu GitHub Apps findest du auf der GitHub Website unter [Über GitHub Apps](#).

Wenn Sie eine neue App verbinden, die in einem GitHub Repo gespeichert ist, verwendet Amplify standardmäßig die GitHub App, um auf das Repo zuzugreifen. Bestehende Amplify-Apps, die Sie zuvor über GitHub Repos verbunden haben, verwenden jedoch OAuth für den Zugriff. CI/CD funktioniert weiterhin für diese Apps, wir empfehlen jedoch dringend, sie zu migrieren, um die neue Amplify GitHub App zu verwenden.

Wenn Sie eine neue App bereitstellen oder eine vorhandene App mithilfe der Amplify-Konsole migrieren, werden Sie automatisch zum Installationsort der GitHub Amplify-App weitergeleitet. Um manuell auf die Installations-Landingpage für die App zuzugreifen, öffnen Sie einen Webbrowser und navigieren Sie nach Regionen zur App. Verwenden Sie das Format `https://github.com/apps/aws-amplify-REGION` und ersetzen Sie *REGION* durch die Region, in der Sie Ihre Amplify-App bereitstellen werden. Um beispielsweise die Amplify GitHub App in der Region USA West (Oregon) zu installieren, navigieren Sie zum Beispiel `aws-amplify-us-west` aus. `https://github.com/apps/`

Themen

- [Installation und Autorisierung der Amplify GitHub App für eine neue Bereitstellung](#)
- [Migration einer vorhandenen OAuth App zur Amplify GitHub App](#)
- [Einrichtung der Amplify GitHub App für AWS CloudFormation CLI- und SDK-Bereitstellungen](#)
- [Webvorschauen mit der Amplify GitHub App einrichten](#)

Installation und Autorisierung der Amplify GitHub App für eine neue Bereitstellung

Wenn Sie aus vorhandenem Code in einem GitHub Repo eine neue App für Amplify bereitstellen, verwenden Sie die folgenden Anweisungen, um die GitHub App zu installieren und zu autorisieren.

Um die Amplify GitHub App zu installieren und zu autorisieren

1. Melden Sie sich bei der [Amplify-Konsole](#) an AWS Management Console und öffnen Sie sie.
2. Wähle auf der Seite Alle Apps die Option Neue App und dann Host-Web-App aus.
3. Wählen Sie auf der Seite Erste Schritte mit Amplify Hosting die Option GitHub und dann Weiter aus.
4. Wenn Sie zum ersten Mal eine Verbindung zu einem GitHub Repository herstellen, wird in Ihrem Browser auf GitHub .com eine neue Seite geöffnet, auf der Sie um die Genehmigung zur Autorisierung AWS Amplify in Ihrem GitHub Konto gebeten werden. Klicken Sie auf Authorize.
5. Als Nächstes müssen Sie die Amplify GitHub App in Ihrem GitHub Konto installieren. Auf GitHub.com wird eine Seite geöffnet, auf der Sie um die Erlaubnis zur Installation und Autorisierung AWS Amplify in Ihrem GitHub Konto gebeten werden.
6. Wählen Sie GitHub aus, wo Sie die Amplify GitHub App installieren möchten.
7. Führen Sie eine der folgenden Aktionen aus:
 - Um die Installation auf alle Repositorys anzuwenden, wählen Sie Alle Repositorys.
 - Um die Installation auf die von Ihnen ausgewählten Repositorys zu beschränken, wählen Sie Nur ausgewählte Repositorys. Stellen Sie sicher, dass Sie das Repo für die App, die Sie migrieren, in die von Ihnen ausgewählten Repos aufnehmen.
8. Wählen Sie „Installieren und autorisieren“.
9. Sie werden auf die Zweigseite Repository hinzufügen für Ihre App in der Amplify-Konsole weitergeleitet.
10. Wählen Sie in der Liste der kürzlich aktualisierten Repositorys den Namen des Repositorys aus, zu dem eine Verbindung hergestellt werden soll.
11. Wählen Sie in der Branch-Liste den Namen des Repository-Zweigs aus, zu dem eine Verbindung hergestellt werden soll.
12. Wählen Sie Next (Weiter).
13. Wählen Sie auf der Seite „Build-Einstellungen konfigurieren“ die Option Weiter aus.
14. Wählen Sie auf der Seite Überprüfen die Option Speichern und bereitstellen aus.

Migration einer vorhandenen OAuth App zur Amplify GitHub App

Bestehende Amplify-Apps, die Sie zuvor über GitHub Repositories verbunden haben, verwenden OAuth für den Repo-Zugriff. Wir empfehlen nachdrücklich, dass Sie diese Apps für die Verwendung der Amplify GitHub App migrieren.

Verwenden Sie die folgenden Anweisungen, um eine App zu migrieren und den entsprechenden OAuth-Webhook in Ihrem GitHub Konto zu löschen. Beachten Sie, dass das Migrationsverfahren davon abhängt, ob die GitHub Amplify-App bereits installiert ist. Nachdem Sie Ihre erste App migriert und die GitHub App installiert und autorisiert haben, müssen Sie nur die Repository-Berechtigungen für nachfolgende App-Migrationen aktualisieren.

Um eine App von OAuth zur GitHub App zu migrieren

1. Melden Sie sich bei der [Amplify-Konsole](#) an AWS Management Console und öffnen Sie sie.
2. Wählen Sie aus.
3. Suchen Sie auf der Informationsseite der App die blaue Meldung Zu unserer GitHub App migrieren und wählen Sie Migration starten aus.
4. Wählen Sie auf der Seite GitHub App installieren und autorisieren die Option GitHub App konfigurieren aus.
5. In Ihrem Browser auf GitHub .com wird eine neue Seite geöffnet, auf der Sie um Erlaubnis zur Autorisierung AWS Amplify in Ihrem GitHub Konto gebeten werden. Klicken Sie auf Authorize.
6. Wählen Sie GitHub aus, wo Sie die Amplify GitHub App installieren möchten.
7. Führen Sie eine der folgenden Aktionen aus:
 - Um die Installation auf alle Repositories anzuwenden, wählen Sie Alle Repositories.
 - Um die Installation auf die von Ihnen ausgewählten Repositories zu beschränken, wählen Sie Nur ausgewählte Repositories. Stellen Sie sicher, dass Sie das Repo für die App, die Sie migrieren, in die von Ihnen ausgewählten Repositories aufnehmen.
8. Wählen Sie „Installieren und autorisieren“.
9. Sie werden auf die Seite GitHub App installieren und autorisieren für Ihre App in der Amplify-Konsole weitergeleitet. Wenn die GitHub Autorisierung erfolgreich war, sehen Sie eine Erfolgsmeldung. Wählen Sie „Weiter“.
10. Wählen Sie auf der Seite Vollständige Installation die Option Vollständige Installation aus. In diesem Schritt wird Ihr vorhandener Webhook gelöscht, ein neuer erstellt und die Migration abgeschlossen.

Einrichtung der Amplify GitHub App für AWS CloudFormation CLI- und SDK-Bereitstellungen

Bestehende Amplify-Apps, die Sie zuvor über GitHub Repositories verbunden haben, verwenden OAuth für den Repo-Zugriff. Dies kann Apps einschließen, die Sie über die -Befehlszeilenschnittstelle (-CLI) oder die -SDKs bereitgestellt haben. Wir empfehlen nachdrücklich, dass Sie diese Apps für die Verwendung der neuen Amplify GitHub App migrieren. Die Migration muss in der Amplify-Konsole in der durchgeföhrt werden AWS Management Console. Detaillierte Anweisungen finden Sie unter [Migration einer vorhandenen OAuth App zur Amplify GitHub App](#).

Sie können die Amplify CLI und die SDKs verwenden AWS CloudFormation, um eine neue Amplify-App bereitzustellen, die die GitHub App für den Repo-Zugriff verwendet. Für diesen Vorgang müssen Sie zuerst die Amplify GitHub App in Ihrem GitHub Konto installieren. Als Nächstes müssen Sie in Ihrem GitHub Konto ein persönliches Zugriffstoken generieren. Stellen Sie abschließend hin und her.

Installieren Sie die Amplify GitHub App in Ihrem Konto

1. Öffnen Sie einen Webbrowser und navigieren Sie zum Installationsort für die Amplify GitHub App in der AWS Region, in der Sie Ihre App bereitstellen werden.

Verwenden Sie das Format `https://github.com/apps/aws-amplify-REGION/installations/new` und ersetzen Sie *REGION* durch Ihre eigene Eingabe. Wenn Sie Ihre App beispielsweise in der Region USA West (Oregon) installieren, geben Sie an `https://github.com/apps/aws-amplify-us-west-2/installations/new`.

2. Wählen Sie GitHub aus, wo Sie die Amplify GitHub App installieren möchten.
3. Föhren Sie eine der folgenden Aktionen aus:
 - Um die Installation auf alle Repositories anzuwenden, wählen Sie Alle Repositories.
 - Um die Installation auf die von Ihnen ausgewählten Repositories zu beschränken, wählen Sie Nur ausgewählte Repositories. Stellen Sie sicher, dass Sie das Repo für die App, die Sie migrieren, in die von Ihnen ausgewählten Repos aufnehmen.
4. Wählen Sie Installieren aus.

Generieren Sie ein persönliches Zugriffstoken in Ihrem GitHub Konto

1. Loggen Sie sich in Ihr GitHub Konto ein.
2. Suchen Sie in der oberen rechten Ecke Ihr Profilfoto und wählen Sie im Menü Einstellungen aus.

Webvorschauen für Pull-Requests

Webvorschauen bieten Teams für Entwicklung und Qualitätssicherung (QA) die Möglichkeit, eine Vorschau der Änderungen anhand von Pull Requests (PRs) anzuzeigen, bevor der Code in einen Produktions- oder Integrationszweig übertragen wird. Mithilfe von Pull-Requests können Sie andere über Änderungen informieren, die Sie an einen Branch in einem Repository übertragen haben. Nachdem ein Pull Request geöffnet wurde, kannst du die potenziellen Änderungen mit deinen Mitarbeitern besprechen und überprüfen und Folge-Commits hinzufügen, bevor deine Änderungen mit dem Basis-Branch zusammengeführt werden.

Bei einer Webvorschau wird jede Pull-Anfrage, die an dein Repository gestellt wird, an eine eindeutige Vorschau-URL gesendet, die sich völlig von der URL unterscheidet, die deine Hauptseite verwendet. Für Apps mit Backend-Umgebungen, die über die Amplify CLI oder Amplify Studio bereitgestellt werden, erstellt jede Pull-Anfrage (nur private Git-Repositorys) ein temporäres Backend, das gelöscht wird, wenn der PR geschlossen wird.

Wenn die Webvorschau für Ihre App aktiviert ist, wird jeder PR auf das Amplify-Kontingent von 50 Filialen pro App angerechnet. Um zu vermeiden, dass dieses Kontingent überschritten wird, stellen Sie sicher, dass Sie Ihre PRs schließen. Weitere Informationen zu Kontingenten finden Sie unter [Amplify Kontingente für Hosting-Dienste](#).

Note

Derzeit ist die `AWS_PULL_REQUEST_ID` Umgebungsvariable nicht verfügbar, wenn Sie sie AWS CodeCommit als Repository-Anbieter verwenden.

Webvorschauen aktivieren

Für Apps, die in einem GitHub Repo gespeichert sind, verwenden Vorschauen die Amplify GitHub App für den Repo-Zugriff. Wenn Sie Webvorschauen in einer vorhandenen Amplify-App aktivieren, die Sie zuvor über ein GitHub Repository mit OAuth für den Zugriff bereitgestellt haben, müssen Sie zuerst die App migrieren, um die Amplify-App zu verwenden. GitHub Anweisungen zur Migration finden Sie unter. [Migration einer vorhandenen OAuth App zur Amplify GitHub App](#)

⚠ Important

Aus Sicherheitsgründen können Sie Webvorschauen für alle Apps mit privaten Repositorys aktivieren, aber nicht für alle Apps mit öffentlichen Repositorys. Wenn Ihr Git-Repository öffentlich ist, können Sie Vorschauen nur für Apps einrichten, für die keine IAM-Servicerolle erforderlich ist.

Beispielsweise benötigen Apps mit Backends und Apps, die auf der WEB_COMPUTE Hosting-Plattform bereitgestellt werden, eine IAM-Servicerolle. Daher können Sie für diese Arten von Apps keine Webvorschauen aktivieren, wenn ihr Repository öffentlich ist.

Amplify erzwingt diese Einschränkung, um zu verhindern, dass Dritte beliebigen Code einreichen, der mit den IAM-Rollenberechtigungen Ihrer App ausgeführt werden würde.

Um Webvorschauen für Pull-Requests zu aktivieren

1. Wählen Sie Hosting und dann Vorschauen aus.

ℹ Note

Vorschauen sind im Menü mit den App-Einstellungen nur sichtbar, wenn eine App für die kontinuierliche Bereitstellung eingerichtet und mit einem Git-Repository verbunden ist. Anweisungen zu dieser Art der Bereitstellung findest du unter [Erste Schritte mit vorhandenem Code](#).

2. Gehen Sie nur für GitHub Repositorys wie folgt vor, um die Amplify GitHub App in Ihrem Konto zu installieren und zu autorisieren:
 - a. Wählen Sie im Fenster GitHub App installieren, um Vorschauen zu aktivieren die Option App installieren. GitHub
 - b. Wählen Sie das GitHub Konto aus, in dem Sie die Amplify GitHub App konfigurieren möchten.
 - c. Auf GitHub.com wird eine Seite geöffnet, auf der Sie die Repository-Berechtigungen für Ihr Konto konfigurieren können.
 - d. Führen Sie eine der folgenden Aktionen aus:
 - Um die Installation auf alle Repositorys anzuwenden, wählen Sie Alle Repositorys.
 - Um die Installation auf die von Ihnen ausgewählten Repositorys zu beschränken, wählen Sie Nur ausgewählte Repositorys. Stellen Sie sicher, dass das Repo für die

App, für die Sie Webvorschauen aktivieren, in die von Ihnen ausgewählten Repositorys aufgenommen wird.

- e. Wählen Sie Speichern.
3. Nachdem Sie die Vorschauen für Ihr Repo aktiviert haben, kehren Sie zur Amplify-Konsole zurück, um Vorschauen für bestimmte Branches zu aktivieren. Wählen Sie auf der Vorschauseite einen Zweig aus der Liste aus und klicken Sie auf Einstellungen bearbeiten.
4. Aktiviere auf der Seite „Vorschauereinstellungen verwalten“ die Option „Pull-Request-Vorschauen“. Wählen Sie dann Confirm (Bestätigen) aus.
5. Führen Sie für Fullstack-Anwendungen einen der folgenden Schritte aus:
 - Wählen Sie „Neue Backend-Umgebung für jeden Pull-Request erstellen“. Mit dieser Option können Sie Änderungen testen, ohne die Produktion zu beeinträchtigen.
 - Wählen Sie Alle Pull-Requests für diesen Branch auf eine bestehende Umgebung verweisen aus.
6. Wählen Sie Bestätigen aus.

Wenn Sie das nächste Mal eine Pull-Anfrage für die Filiale einreichen, erstellt Amplify Ihre PR und stellt sie auf einer Vorschau-URL bereit. Nachdem die Pull-Anfrage geschlossen wurde, wird die Vorschau-URL gelöscht und jede temporäre Backend-Umgebung, die mit der Pull-Anfrage verknüpft ist, wird gelöscht. Nur für GitHub Repositorys kannst du direkt über die Pull-Anfrage in deinem Konto auf eine Vorschau deiner GitHub URL zugreifen.

Zugriff auf die Webvorschau mit Subdomains

Webvorschauen für Pull-Requests sind über Subdomains für eine Amplify-App zugänglich, die mit einer von Amazon Route 53 verwalteten benutzerdefinierten Domain verbunden ist. Wenn der Pull-Request geschlossen wird, werden die mit dem Pull-Request verknüpften Branches und Subdomains automatisch gelöscht. Dies ist das Standardverhalten für Webvorschauen, nachdem Sie musterbasierte Feature-Branch-Bereitstellungen für Ihre App eingerichtet haben. Anweisungen zur Einrichtung automatischer Subdomains finden Sie unter [Automatische Subdomains für eine benutzerdefinierte Amazon Route 53-Domain einrichten](#)

Fügen Sie end-to-end Cypress-Tests zu Ihrer Amplify-App hinzu

Sie können end-to-end (E2E) -Tests in der Testphase Ihrer Amplify-App ausführen, um Regressionen catch, bevor Sie den Code in die Produktion bringen. Die Testphase kann in der Build-Spezifikation YAML konfiguriert werden. Derzeit können Sie während eines Builds nur das Cypress-Testframework ausführen.

Tutorial: Richten Sie end-to-end Tests mit Cypress ein

Cypress ist ein JavaScript basiertes Test-Framework, mit dem Sie E2E-Tests in einem Browser ausführen können. Ein Tutorial, das zeigt, wie Sie E2E-Tests einrichten, finden Sie im Blogbeitrag [Ausführen von end-to-end Cypress-Tests für Ihre Fullstack-CI/CD-Bereitstellung](#) mit Amplify.

Fügen Sie Tests zu Ihrer bestehenden Amplify-App hinzu

Sie können Cypress-Tests zu einer vorhandenen App hinzufügen, indem Sie die Build-Einstellungen der App in der Amplify-Konsole aktualisieren. Die Build-Spezifikation YAML enthält eine Sammlung von Build-Befehlen und zugehörigen Einstellungen, die Amplify verwendet, um Ihren Build auszuführen. Verwenden Sie den `test` Schritt, um alle Testbefehle zur Build-Zeit auszuführen. Für E2E-Tests bietet Amplify Hosting eine tiefere Integration mit Cypress, mit der Sie einen UI-Bericht für Ihre Tests erstellen können.

In der folgenden Liste werden die Testeinstellungen und deren Verwendung beschrieben.

Vor dem Test

Installieren Sie die Abhängigkeiten, die für die Ausführung von Cypress-Tests erforderlich sind. Amplify Hosting verwendet [Mochawesome](#), um einen Bericht zu erstellen, in dem Sie Ihre Testergebnisse einsehen können, und [warten Sie, bis der Localhost-Server](#) während des Builds eingerichtet ist.

Test

Führen Sie Cypress-Befehle aus, um Tests mit Mochawesome durchzuführen.

Nach dem Test

Der Mochawesome-Bericht wird aus der JSON-Ausgabe generiert. Beachten Sie, dass Sie, wenn Sie Yarn verwenden, diesen Befehl im unbeaufsichtigten Modus ausführen müssen, um den Mochawesome-Bericht zu generieren. Für Yarn können Sie den folgenden Befehl verwenden.

```
yarn run --silent mochawesome-merge cypress/report/mochawesome-report/  
mochawesome*.json > cypress/report/mochawesome.json
```

Artefakte > Basisverzeichnis

Das Verzeichnis, von dem aus Tests ausgeführt werden.

Artefakte> configFilePath

Die generierten Testberichtsdaten.

Artefakte>Dateien

Die generierten Artefakte (Screenshots und Videos) stehen zum Herunterladen zur Verfügung.

Der folgende Beispielauszug aus einer `amplify.yml` Build-Spezifikationsdatei zeigt, wie Sie Ihrer App Cypress-Tests hinzufügen.

```
test:  
  phases:  
    preTest:  
      commands:  
        - npm ci  
        - npm install -g pm2  
        - npm install -g wait-on  
        - npm install mocha mochawesome mochawesome-merge mochawesome-report-generator  
        - pm2 start npm -- start  
        - wait-on http://localhost:3000  
    test:  
      commands:  
        - 'npx cypress run --reporter mochawesome --reporter-options  
"reportDir=cypress/report/mochawesome-  
report,overwrite=false,html=false,json=true,timestamp=mmddyyyy_HHMMss"  
      postTest:  
        commands:
```

```
- npx mochawesome-merge cypress/report/mochawesome-report/mochawesome*.json >
cypress/report/mochawesome.json
- pm2 kill
artifacts:
  baseDirectory: cypress
  configFile: '**/mochawesome.json'
  files:
    - '**/*.png'
    - '**/*.mp4'
```

Tests deaktivieren

Nachdem die Testkonfiguration zu Ihren `amplify.yml` Build-Einstellungen hinzugefügt wurde, wird der `test` Schritt für jeden Build in jedem Zweig ausgeführt. Wenn Sie die Ausführung von Tests global deaktivieren oder nur Tests für bestimmte Zweige ausführen möchten, können Sie die `USER_DISABLE_TESTS` Umgebungsvariable verwenden, ohne Ihre Build-Einstellungen zu ändern.

Um Tests für alle Branches global zu deaktivieren, fügen Sie die `USER_DISABLE_TESTS` Umgebungsvariable mit dem Wert `true for all Branches` hinzu. Der folgende Screenshot zeigt den Abschnitt Umgebungsvariablen in der Amplify-Konsole mit deaktivierten Tests für alle Zweige.

Environment Variables [Manage variables](#)

Environment variables are key/value pairs that contain any constant values your app needs at build time. For instance, database connection details or third party API keys. [Learn more](#)


Branch	Variable	Value
All branches	USER_DISABLE_TESTS	True

Rows per page: 15 <> 1 <>





Um Tests für einen bestimmten Zweig zu deaktivieren, fügen Sie die `USER_DISABLE_TESTS` Umgebungsvariable mit dem Wert `false` für alle Zweige hinzu und fügen Sie dann für jeden Zweig, den Sie deaktivieren möchten, eine Überschreibung mit dem Wert von `true` hinzu. Im folgenden Screenshot sind Tests für den Hauptzweig deaktiviert und für jeden anderen Zweig aktiviert.

Environment Variables

[Manage variables](#)

Environment variables are key/value pairs that contain any constant values your app needs at build time. For instance, database connection details or third party API keys. [Learn more](#) 

Branch ▾	Variable ▾	Value ▾
All branches	USER_DISABLE_TESTS	False
main	USER_DISABLE_TESTS	True

Rows per page 15 ▾   1  

Das Deaktivieren von Tests mit dieser Variablen führt dazu, dass der Testschritt während eines Builds vollständig übersprungen wird. Um Tests wieder zu aktivieren, setzen Sie diesen Wert auf `false` oder löschen Sie die Umgebungsvariable.

Weiterleitungen verwenden

Umleitungen ermöglichen einem Webserver das Umleiten der Navigation von einer URL zu einer anderen. Zu den häufigsten Gründen für die Verwendung von Weiterleitungen gehören das Anpassen des Erscheinungsbilds einer URL, die Vermeidung defekter Links, das Verschieben des Hosting-Speicherorts einer App oder Website, ohne deren Adresse zu ändern, und das Ändern einer angeforderten URL in das von einer Web-App benötigte Formular.

Arten von Weiterleitungen

Amplify unterstützt die folgenden Umleitungstypen in der Konsole.

Dauerhafte Umleitung (301)

301-Umleitungen sind für dauerhafte Änderungen am Ziel einer Webadresse vorgesehen. Der Verlauf des Suchmaschinenrankings der ursprünglichen Adresse gilt für die neue Zieladresse. Die Umleitung erfolgt auf der Clientseite, daher wird in einer Browser-Navigationsleiste nach der Umleitung die Zieladresse angezeigt.

Häufige Gründe für die Verwendung von 301-Umleitungen:

- Vermeiden eines fehlerhaften Links beim Ändern der Adresse einer Seite
- Vermeiden eines fehlerhaften Links, wenn ein Benutzer einen vorhersehbare Tippfehler in einer Adresse macht

Temporäre Umleitung (302)

302-Umleitungen sind für temporäre Änderungen am Ziel einer Webadresse vorgesehen. Der Verlauf des Suchmaschinen-Rankings der ursprünglichen Adresse gilt nicht für die neue Zieladresse. Die Umleitung erfolgt auf der Clientseite, daher wird in einer Browser-Navigationsleiste nach der Umleitung die Zieladresse angezeigt.

Häufige Gründe für die Verwendung von 302-Umleitungen:

- Bereitstellen eines Umleitungsziels, während an einer ursprünglichen Adresse Reparaturen vorgenommen werden
- Um Testseiten für den A/B-Vergleich einer Benutzeroberfläche bereitzustellen.

Note

Wenn Ihre App eine unerwartete 302-Antwort zurückgibt, wird der Fehler wahrscheinlich durch Änderungen verursacht, die Sie an der Weiterleitung und der benutzerdefinierten Header-Konfiguration Ihrer App vorgenommen haben. Um dieses Problem zu beheben, stellen Sie sicher, dass Ihre benutzerdefinierten Header gültig sind, und aktivieren Sie dann erneut die standardmäßige 404-Rewrite-Regel für Ihre App.

Umschreibung (200)

200-Umleitungen (Umschreibungen) dienen zum Anzeigen von Inhalt an der Zieladresse, als würde er von der ursprünglichen Adresse bereitgestellt. Der Verlauf des Suchmaschinenrankings gilt weiterhin für die ursprüngliche Adresse. Die Umleitung erfolgt auf der Serverseite, daher wird in einer Browser-Navigationsleiste nach der Umleitung die ursprüngliche Adresse angezeigt. Häufige Gründe für die Verwendung von 200-Umleitungen:

- Umleiten einer gesamten Website an einen neuen Hostingstandort, ohne die Adresse der Website zu ändern
- Umleiten des gesamten Datenverkehrs an die Seite „index.html“ einer Single Page Web App (SPA) zur Verarbeitung durch eine clientseitige Routerfunktion

Nicht gefunden (404)

404-Weiterleitungen treten auf, wenn eine Anfrage auf eine Adresse verweist, die nicht existiert. Die Zielseite einer 404-Umleitung wird anstatt der angeforderten Seite angezeigt. Häufige Gründe für 404-Umleitungen:

- Vermeiden einer Meldung vom Typ „Fehlerhafter Link“, wenn ein Benutzer eine ungültige URL eingibt
- Verweisen von Anforderungen an nicht vorhandene Seiten einer Web-App an die Seite „index.html“ zur Verarbeitung durch eine clientseitige Routerfunktion

Weiterleitungen erstellen und bearbeiten

Sie können Weiterleitungen für eine App in der Amplify-Konsole erstellen und bearbeiten. Bevor Sie beginnen, benötigen Sie die folgenden Informationen zu den Teilen einer Weiterleitung.

Eine Originaladresse

Die Adresse, die der Benutzer angefordert hat.

Eine Zieladresse

Die Adresse, die tatsächlich den Inhalt bereitstellt, den der Benutzer sieht.

Ein Umleitungstyp

Zu den Typen gehören eine permanente Weiterleitung (301), eine temporäre Weiterleitung (302), eine Umschreibung (200) oder eine nicht gefundene Weiterleitung (404).

Ein aus zwei Buchstaben bestehender Ländercode (optional)

Ein Wert, den Sie angeben können, um die Benutzererfahrung Ihrer App nach geografischer Region zu segmentieren.

Um eine Weiterleitung in der Amplify-Konsole zu erstellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die Sie eine Weiterleitung erstellen möchten.
3. Wählen Sie im Navigationsbereich Hosting und anschließend Umschreibungen und Weiterleitungen aus.
4. Wählen Sie auf der Seite Umschreibungen und Weiterleitungen die Option Weiterleitungen verwalten aus.
5. Das Verfahren zum Hinzufügen einer Weiterleitung hängt davon ab, ob du Regeln einzeln hinzufügen oder eine Massenbearbeitung durchführen möchtest:
 - Um eine individuelle Weiterleitung zu erstellen, wähle Umschreibung hinzufügen.
 - a. Geben Sie unter Quelladresse die ursprüngliche Adresse ein, die der Benutzer angefordert hat.
 - b. Geben Sie unter Zieladresse die Zieladresse ein, über die der Inhalt an den Benutzer gesendet wird.
 - c. Wählen Sie unter Typ den Typ der Weiterleitung aus der Liste aus.

- d. (Optional) Geben Sie als Ländercode eine Bedingung für die Landesvorwahl ein, die aus zwei Buchstaben besteht.
- Um Weiterleitungen massenweise zu bearbeiten, wählen Sie Texteditor öffnen.
 - Fügen Sie Weiterleitungen im JSON-Editor für Umschreibungen und Weiterleitungen manuell hinzu oder aktualisieren Sie sie.
6. Wählen Sie Speichern.

Reihenfolge der Weiterleitungen

Weiterleitungen werden vom Anfang der Liste nach unten angewendet. Stellen Sie sicher, dass Ihre Reihenfolge das vorgesehene Ergebnis erzielt. Beispiel: Die folgende Reihenfolge von Umleitungen bewirkt, dass alle Anforderungen für einen bestimmten Pfad unter `/docs/` an denselben Pfad unter `/documents/` umgeleitet werden, mit Ausnahme von `/docs/bestimmter-dateiname.html`, das an `/documents/anderer-dateiname.html` umgeleitet wird:

```
/docs/specific-filename.html /documents/different-filename.html 301
/docs/<*> /documents/<*>
```

Bei der folgenden Umleitungsreihenfolge wird die Umleitung von `bestimmter-dateiname.html` an `anderer-dateiname.html` ignoriert:

```
/docs/<*> /documents/<*>
/docs/specific-filename.html /documents/different-filename.html 301
```

Abfrageparameter

Sie können Abfrageparameter verwenden, um mehr Kontrolle über Ihre URL-Übereinstimmungen zu erhalten. Amplify leitet alle Abfrageparameter für 301- und 302-Weiterleitungen an den Zielpfad weiter, mit den folgenden Ausnahmen:

- Wenn die ursprüngliche Adresse eine Abfragezeichenfolge enthält, die auf einen bestimmten Wert festgelegt ist, leitet Amplify keine Abfrageparameter weiter. In diesem Fall gilt die Umleitung nur für Anfragen an die Ziel-URL mit dem angegebenen Abfragewert.
- Wenn die Zieladresse für die Abgleichsregel Abfrageparameter enthält, werden Abfrageparameter nicht weitergeleitet. Wenn die Zieladresse für die Umleitung beispielsweise lautet `https://example-target.com?q=someParam`, werden Abfrageparameter nicht weitergeleitet.

Einfache Weiterleitungen und Umschreibungen

Dieser Abschnitt enthält Beispielcode für gängige Umleitungsszenarien.

Note

Beim Abgleich der ursprünglichen Adressdomäne wird nicht zwischen Groß- und Kleinschreibung unterschieden.

Sie können den folgenden Beispielcode verwenden, um eine bestimmte Seite dauerhaft an eine neue Adresse umzuleiten.

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
/original.html	/destination.html	permanent redirect (301)	

```
JSON [{"source": "/original.html", "status": "301", "target": "/destination.html", "condition": null}]
```

Sie können den folgenden Beispielcode verwenden, um einen beliebigen Pfad in einem Ordner an den gleichen Pfad in einem anderen Ordner umzuleiten.

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
/docs/<*>	/documents/<*>	permanent redirect (301)	

```
JSON [{"source": "/docs/<*>", "status": "301", "target": "/documents/<*>", "condition": null}]
```

Sie können den folgenden Beispielcode verwenden, um den gesamten Datenverkehr als Umschreibung an „index.html“ umzuleiten. In diesem Szenario wird die Seite dem Benutzer so angezeigt, als befände er sich an der ursprünglichen Adresse.

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
<code>/<*></code>	<code>/index.html</code>	<code>rewrite (200)</code>	

```
JSON [{"source": "/<*>", "status": "200", "target": "/index.html", "condition": null}]
```

Sie können den folgenden Beispielcode verwenden, um mithilfe einer Umschreibung die Unterdomäne zu ändern, die dem Benutzer angezeigt wird.

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
<code>https://mydomain.com</code>	<code>https://www.mydomain.com</code>	<code>rewrite (200)</code>	

```
JSON [{"source": "https://mydomain.com", "status": "200", "target": "https://www.mydomain.com", "condition": null}]
```

Sie können den folgenden Beispielcode verwenden, um zu einer anderen Domain mit einem Pfadpräfix umzuleiten.

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
<code>https://mydomain.com</code>	<code>https://www.mydomain.com/documents</code>	<code>temporary redirect (302)</code>	

```
JSON [{"source": "https://mydomain.com", "status": "302", "target": "https://www.mydomain.com/documents/", "condition": null}]
```

Sie können den folgenden Beispielcode verwenden, um Pfade unter einem Ordner, der nicht gefunden werden kann, auf eine benutzerdefinierte 404-Seite umzuleiten.

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
<code>/<*></code>	<code>/404.html</code>	not found (404)	

```
JSON [{"source": "/<*>", "status": "404", "target": "/404.html", "condition": null}]
```

Weiterleitungen für einseitige Web-Apps (SPA)

Die meisten SPA-Frameworks unterstützen HTML5 `history.pushState()`, um den Browserstandort zu ändern, ohne eine Serveranfrage auszulösen. Diese Methode funktioniert für Benutzer, die an der Stamm-URL (oder `/index.html`) beginnen, aber nicht für Benutzer, die direkt zu einer anderen Seite navigieren.

Im folgenden Beispiel werden reguläre Ausdrücke verwendet, um ein 200-Umschreiben für alle Dateien in `index.html` einzurichten, mit Ausnahme der im regulären Ausdruck angegebenen Dateierweiterungen.

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
<code></^[^.]�+\$ \.(?!(css gif ico jpg js png txt svg woff woff2 ttf map json webp)\$)([^\.]�+\$)/></code>	<code>/index.html</code>	200	

```
JSON [{"source": "</^[^.]�+$|\.(?!(css|gif|ico|jpg|js|png|txt|svg|woff|woff2|ttf|map|json|webp)$)([^\.]�+$)/>", "status": "200", "target": "/index.html", "condition": null}]
```

Reverse-Proxy umschreiben

Im folgenden Beispiel wird ein Rewrite verwendet, um Inhalt von einem anderen Ort aus als Proxy zu verwenden, sodass der Benutzer den Eindruck hat, dass sich die Domain nicht geändert hat.

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
/images/<*>	https://images.otherdomain.com/<*>	rewrite (200)	

```
JSON [{"source": "/images/<*>", "status": "200", "target": "https://images.otherdomain.com/<*>", "condition": null}]
```

Schrägstriche am Ende und saubere URLs

Zum Erstellen bereinigter URL-Strukturen wie `about` anstelle von `about.html` generieren Generatoren von statischen Seiten wie z. B. Hugo Verzeichnisse für Seiten mit „`index.html`“ (`/about/index.html`). Amplify erstellt automatisch saubere URLs, indem es bei Bedarf einen abschließenden Schrägstrich hinzufügt. In der folgenden Tabelle sind verschiedene Szenarien hervorgehoben:

Benutzereingaben im Browser	URL in der Adressleiste	Bereitgestelltes Dokument
/about	/about	/about.html
/about (when about.html returns 404)	/about/	/about/index.html
/about/	/about/	/about/index.html

Platzhalter

Sie können den folgenden Beispielcode verwenden, um Pfade in einer Ordnerstruktur an eine entsprechende Struktur in einem anderen Ordner umzuleiten.

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
/docs/<year>/<month>/<date>/<itemid>	/documents/<year>/<month>/<date>/<itemid>	permanent redirect (301)	

```
JSON [{"source": "/docs/<year>/<month>/<date>/<itemid>", "status": "301", "target": "/documents/<year>/<month>/<date>/<itemid>", "condition": null}]
```

Abfragen von Zeichenketten und Pfadparametern

Sie können den folgenden Beispielcode verwenden, um einen Pfad an einen Ordner mit einem Namen umzuleiten, der dem Wert eines Abfragezeichenfolgeelements in der ursprünglichen Adresse entspricht:

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
/docs?id=<my-blog-id-value>	/documents/<my-blog-post-id-value>	permanent redirect (301)	

```
JSON [{"source": "/docs?id=<my-blog-id-value>", "status": "301", "target": "/documents/<my-blog-id-value>", "condition": null}]
```

Note

Amplify leitet alle Abfragezeichenfolgeparameter für 301- und 302-Weiterleitungen an den Zielpfad weiter. Wenn die ursprüngliche Adresse jedoch eine Abfragezeichenfolge enthält, die auf einen bestimmten Wert festgelegt ist, wie in diesem Beispiel gezeigt, leitet Amplify keine Abfrageparameter weiter. In diesem Fall gilt die Umleitung nur für Anfragen an die Zieladresse mit dem angegebenen Abfragewertid.

Sie können den folgenden Beispielcode verwenden, um alle Pfade, die auf einer bestimmten Ebene einer Ordnerstruktur nicht gefunden werden können, zu `index.html` in einem angegebenen Ordner umzuleiten.

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
<code>/documents/ <folder>/ <child-folder>/ <grand-child- folder></code>	<code>/documents/ index.html</code>	not found (404)	

```
JSON [{"source": "/documents/<x>/<y>/<z>", "status": "404", "target": "/documents/index.html", "condition": null}]
```

Regionsbasierte Weiterleitungen

Sie können den folgenden Beispielcode verwenden, um Anforderungen basierend auf der Region umzuleiten.

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
<code>/documents</code>	<code>/documents/us/</code>	temporary redirect (302)	<code><US></code>

```
JSON [{"source": "/documents", "status": "302", "target": "/documents/us/", "condition": "<US>"}]
```

Platzhalterausrücke in Weiterleitungen und Umschreibungen

Sie können den Platzhalterausrück, `<*>`, in der ursprünglichen Adresse für eine Umleitung oder ein Umschreiben verwenden. Sie müssen den Ausdruck am Ende der ursprünglichen Adresse platzieren und er muss eindeutig sein. Amplify ignoriert Originaladressen, die mehr als einen Platzhalterausrück enthalten, oder verwendet ihn an einer anderen Stelle.

Im Folgenden finden Sie ein Beispiel für eine gültige Weiterleitung mit einem Platzhalterausrück.

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
/docs/<*>	/documents/<*>	permanent redirect (301)	

Die folgenden beiden Beispiele zeigen ungültige Weiterleitungen mit Platzhalterausrücken.

Ursprüngliche Adresse	Zieladresse	Umleitungsart	Country Code (Ländercode)
/docs/<*>/ content	/documents/<*>/ content	permanent redirect (301)	
/docs/<*>/ content/<*>	/documents/<*>/ content/<*>	permanent redirect (301)	

Beschränkung des Zugangs zu Filialen

Wenn Sie an unveröffentlichten Funktionen arbeiten, können Sie Feature-Branched mit einem Passwort schützen, um den Zugriff auf bestimmte Benutzer zu beschränken. Wenn die Zugriffskontrolle für einen Branch eingerichtet ist, werden Benutzer aufgefordert, einen Benutzernamen und ein Passwort einzugeben, wenn sie versuchen, auf die URL für den Branch zuzugreifen.

Sie können ein Passwort festlegen, das für eine einzelne Filiale oder global für alle verbundenen Filialen gilt. Wenn die Zugriffskontrolle sowohl auf Filialebene als auch auf globaler Ebene aktiviert ist, hat das Passwort auf Filialebene Vorrang vor einem Passwort auf globaler (Anwendungs-) Ebene.

Um Passwörter für Feature-Branched festzulegen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die Sie Feature-Branch-Passwörter einrichten möchten.
3. Wählen Sie im Navigationsbereich Hosting und dann Access Control aus.
4. Wählen Sie im Abschnitt Einstellungen für die Zugriffskontrolle die Option Zugriff verwalten aus.
5. Führen Sie auf der Seite „Zugriffskontrolle verwalten“ einen der folgenden Schritte aus.
 - Um einen Benutzernamen und ein Passwort festzulegen, die für alle verbundenen Filialen gelten
 - Aktivieren Sie die Option Zugriff für alle Filialen verwalten. Wenn Sie beispielsweise Main -, Dev - und Feature-Branched miteinander verbunden haben, können Sie für alle Branches denselben Benutzernamen und dasselbe Passwort verwenden.
 - Um einen Benutzernamen und ein Passwort auf einen einzelnen Branch anzuwenden
 - a. Deaktivieren Sie die Option Zugriff für alle Filialen verwalten.
 - b. Suchen Sie die Filiale, die Sie verwalten möchten. Wählen Sie unter Zugriffseinstellungen die Option Eingeschränktes Passwort erforderlich aus.
 - c. Geben Sie unter Nutzernamen einen Nutzernamen ein.
 - d. Geben Sie in das Feld Passwort ein Passwort ein.
 - Wählen Sie Speichern.

6. Wenn Sie die Zugriffskontrolle für eine serverseitig gerenderte (SSR) App verwalten, stellen Sie die App erneut bereit, indem Sie einen neuen Build aus Ihrem Git-Repository ausführen. Dieser Schritt ist erforderlich, damit Amplify Ihre Einstellungen für die Zugriffskontrolle anwenden kann.

Umgebungsvariablen

Umgebungsvariablen sind Schlüssel-Wert-Paare, die Sie zu den Einstellungen Ihrer Anwendung hinzufügen können, um sie für Amplify Hosting verfügbar zu machen. Als bewährte Methode können Sie Umgebungsvariablen verwenden, um Anwendungskonfigurationsdaten verfügbar zu machen. Alle Umgebungsvariablen, die Sie hinzufügen, sind verschlüsselt, um unbefugten Zugriff zu verhindern.

Amplify erzwingt die folgenden Einschränkungen für die von Ihnen erstellten Umgebungsvariablen.

- Amplify erlaubt Ihnen nicht, Umgebungsvariablenamen mit einem AWS Präfix zu erstellen. Dieses Präfix ist nur für den internen Gebrauch von Amplify reserviert.
- Der Wert einer Umgebungsvariablen darf 5500 Zeichen nicht überschreiten.

Important

Verwenden Sie keine Umgebungsvariablen zum Speichern von Geheimnissen. Verwenden Sie für eine Gen 2-App die Secret-Verwaltungsfunktion in der Amplify-Konsole. Weitere Informationen finden Sie unter [Secrets and environment vars](#) in der Amplify-Dokumentation. Speichern Sie für eine Gen-1-App Geheimnisse in einem Umgebungsgeheimnis, das mit dem AWS Systems Manager Parameter Store erstellt wurde. Weitere Informationen finden Sie unter [Verwaltung von Umgebungsgeheimnissen](#).

Umgebungsvariablen Amplify

Auf die folgenden Umgebungsvariablen kann standardmäßig in der Amplify-Konsole zugegriffen werden.

Variablenname	Beschreibung	Beispielwert
<code>_BUILD_TIMEOUT</code>	Die Dauer des Build-Timeouts in Minuten	30
<code>_LIVE_UPDATES</code>	Das Tool wird auf die neueste Version aktualisiert.	<code>[{"name": "Amplify CLI", "pkg": "@aws-amplify/cli", "type"</code>

Variablenname	Beschreibung	Beispielwert
		<code>:"npm", "version": "latest"]]</code>
USER_DISABLE_TESTS	<p>Der Testschritt wird während eines Builds übersprungen. Sie können Tests für alle Zweige oder bestimmte Zweige in einer App deaktivieren.</p> <p>Diese Umgebungsvariable wird für Apps verwendet, die während der Buildphase Tests durchführen. Weitere Informationen zum Setzen dieser Variablen finden Sie unter Tests deaktivieren.</p>	true
AWS_APP_ID	Die App-ID des aktuellen Builds	abcd1234
AWS_BRANCH	Der Verzweigungsname des aktuellen Builds	main, develop, beta, v2.0
AWS_BRANCH_ARN	Der Branch-Arizona-Ressourcenname (ARN) des aktuellen Builds	aws:arn:amplify:us-west-2:123456789012:appname/branch/...
AWS_CLONE_URL	Die Klon-URL zum Abrufen der Git-Repository-Inhalte	git@github.com:<user-name>/<repo-name>.git
AWS_COMMIT_ID	Die Commit-ID des aktuellen Builds „HEAD“ für Rebuilds	abcd1234

Variablenname	Beschreibung	Beispielwert
AWS_JOB_ID	<p>Die Auftrags-ID des aktuellen Builds.</p> <p>Dies beinhaltet eine gewisse Polsterung von '0', sodass es immer dieselbe Länge hat.</p>	0000000001
AWS_PULL_REQUEST_ID	<p>Die Pull-Request-ID des Pull-Request-Webvorschau-Builds.</p> <p>Diese Umgebungsvariable ist nicht verfügbar, wenn Sie sie AWS CodeCommit als Repository-Anbieter verwenden.</p>	1
AWS_PULL_REQUEST_SOURCE_BRANCH	Der Name des Feature-B ranches für eine Pull-Request-Vorschau, der an einen Anwendungsweig in der Amplify-Konsole gesendet wird.	featureA
AWS_PULL_REQUEST_DESTINATION_BRANCH	Der Name des Anwendungszweigs in der Amplify-K onsole, an den eine Feature-Branch-Pull-Anfrage gesendet wird.	main
AMPLIFY_AMAZON_CLIENT_ID	Die Amazon-Kunden-ID	123456
AMPLIFY_AMAZON_CLIENT_SECRET	Das Geheimnis des Amazon-Kunden	example123456
AMPLIFY_FACEBOOK_CLIENT_ID	Die Facebook-Kunden-ID	123456

Variablenname	Beschreibung	Beispielwert
AMPLIFY_FACEBOOK_CLIENT_SECRET	Das geheime Facebook-Kundegeheimnis	example123456
AMPLIFY_GOOGLE_CLIENT_ID	Die Google-Kunden-ID	123456
AMPLIFY_GOOGLE_CLIENT_SECRET	Das Geheimnis des Google-Clients	example123456
AMPLIFY_DIFF_DEPLOY	Aktivieren oder deaktivieren Sie die diff-basierte Frontend-Bereitstellung. Weitere Informationen finden Sie unter Aktiviert oder deaktiviert das auf Diff basierende Frontend-Build und Deployment .	true
AMPLIFY_DIFF_DEPLOY_ROOT	Der Pfad, der für Vergleichsvergleiche von Frontend-Deployments verwendet werden soll, relativ zum Stammverzeichnis Ihres Repositorys.	dist
AMPLIFY_DIFF_BACKEND	Aktiviert oder deaktiviert diff-basierte Backend-Builds. Dies gilt nur für Apps der 1. Generation. Weitere Informationen finden Sie unter Aktivieren oder deaktivieren Sie diff-basierte Backend-Builds für eine Gen-1-App .	true

Variablenname	Beschreibung	Beispielwert
AMPLIFY_BACKEND_PULL_ONLY	Amplify verwaltet diese Umgebungsvariable. Dies gilt nur für Apps der 1. Generation. Weitere Informationen finden Sie unter Bearbeiten Sie ein vorhandenes Frontend so, dass es auf ein anderes Backend verweist .	true
AMPLIFY_BACKEND_APP_ID	Amplify verwaltet diese Umgebungsvariable. Dies gilt nur für Apps der 1. Generation. Weitere Informationen finden Sie unter Bearbeiten Sie ein vorhandenes Frontend so, dass es auf ein anderes Backend verweist .	abcd1234
AMPLIFY_SKIP_BACKEND_BUILD	Wenn Ihre Build-Spezifikation keinen Backend-Abschnitt enthält und Sie Backend-Builds deaktivieren möchten, setzen Sie diese Umgebungsvariable auf <code>true</code> . Dies gilt nur für Apps der 1. Generation.	true
AMPLIFY_ENABLE_DEBUG_OUTPUT	Setzen Sie diese Variable auf <code>true</code> um einen Stack-Trace in den Protokollen zu drucken. Dies ist hilfreich beim Debuggen von Backend-Build-Fehlern.	true

Variablenname	Beschreibung	Beispielwert
AMPLIFY_MONOREPO_APP_ROOT	Der Pfad, der verwendet werden soll, um das App-Stammverzeichnis einer Monorepo-App relativ zum Stammverzeichnis Ihres Repositorys anzugeben.	apps/react-app
AMPLIFY_USERPOOL_ID	Die ID für den Amazon Cognito Cognito-Benutzerpool, der für die Authentifizierung importiert wurde	us-west-2_example
AMPLIFY_WEBCLIENT_ID	Die ID für den App-Client, der von Webanwendungen verwendet werden soll Der App-Client muss mit Zugriff auf den Amazon Cognito Cognito-Benutzerpool konfiguriert sein, der in der Umgebungsvariablen AMPLIFY_USERPOOL_ID angegeben ist.	123456
AMPLIFY_NATIVECLIENT_ID	Die ID für den App-Client, der von nativen Anwendungen verwendet werden soll Der App-Client muss mit Zugriff auf den Amazon Cognito Cognito-Benutzerpool konfiguriert sein, der in der Umgebungsvariablen AMPLIFY_USERPOOL_ID angegeben ist.	123456

Variablenname	Beschreibung	Beispielwert
AMPLIFY_IDENTITYPOOL_ID	Die ID für den Amazon Cognito Cognito-Identitätspool	example-identitypool-id
AMPLIFY_PERMISSIONS_BOUNDARY_ARN	Der ARN für die IAM-Richtlinie, der als Berechtigungsgrenze verwendet werden soll und für alle von Amplify erstellten IAM-Rollen gilt. Weitere Informationen finden Sie unter IAM-Berechtigungen für von Amplify generierte Rollen .	arn:aws:iam::123456789012:policy/example-policy
AMPLIFY_DESTRUCTIVE_UPDATES	Setzen Sie diese Umgebungsvariable auf true, damit eine GraphQL-API mit Schemaoperationen aktualisiert werden kann, die möglicherweise zu Datenverlust führen können.	true

Note

Die Umgebungsvariablen `AMPLIFY_AMAZON_CLIENT_ID` und die `AMPLIFY_AMAZON_CLIENT_SECRET` Umgebungsvariablen sind OAuth-Token, kein AWS Zugriffsschlüssel und kein geheimer Schlüssel.

Festlegen von Umgebungsvariablen

Verwenden Sie die folgenden Anweisungen, um Umgebungsvariablen für eine Anwendung in der Amplify-Konsole festzulegen.

Note

Umgebungsvariablen sind nur dann im App-Einstellungsmenü der Amplify-Konsole sichtbar, wenn eine App für die kontinuierliche Bereitstellung eingerichtet und mit einem Git-Repository verbunden ist. Anweisungen zu dieser Art der Bereitstellung finden Sie unter [Erste Schritte mit vorhandenem Code](#).

So legen Sie Umgebungsvariablen fest

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie in der Amplify-Konsole Hosting und dann Umgebungsvariablen aus.
3. Wählen Sie auf der Seite Umgebungsvariablen die Option Variablen verwalten aus.
4. Geben Sie unter Variable Ihren Schlüssel ein. Geben Sie unter Wert Ihren Wert ein. Standardmäßig wendet Amplify die Umgebungsvariablen auf alle Zweige an, sodass Sie Variablen nicht erneut eingeben müssen, wenn Sie einen neuen Zweig verbinden.
5. (Optional) Um eine Umgebungsvariable speziell für einen Zweig anzupassen, fügen Sie wie folgt eine Branch-Override hinzu:
 - a. Wählen Sie „Aktionen“ und anschließend „Variablenüberschreibung hinzufügen“.
 - b. Sie verfügen jetzt über einen spezifischen Satz von Umgebungsvariablen für Ihre Verzweigung.
6. Wählen Sie Speichern.

Greifen Sie während der Erstellung auf Umgebungsvariablen zu

Für den Zugriff auf eine Umgebungsvariable während eines Builds bearbeiten Sie Ihre Build-Einstellungen so, dass die Umgebungsvariable in Ihren Build-Befehlen enthalten ist.

Jeder Befehl in Ihrer Build-Konfiguration wird in einer Bash-Shell ausgeführt. Weitere Informationen zur Arbeit mit Umgebungsvariablen in Bash finden Sie unter [Shell-Erweiterungen](#) im GNU Bash-Handbuch.

Um Build-Einstellungen so zu bearbeiten, dass sie eine Umgebungsvariable einschließen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie in der Amplify-Konsole Hosting und dann Build-Einstellungen aus.

3. Wählen Sie im Abschnitt App-Build-Spezifikation die Option Bearbeiten aus.
4. Fügen Sie die Umgebungsvariable Ihrem Build-Befehl hinzu. Jetzt sollte der Zugriff auf Ihre Umgebungsvariable während Ihres nächsten Builds möglich sein. Dieses Beispiel ändert das Verhalten von npm (BUILD_ENV) und fügt einer Umgebungsdatei ein API-Token (TWITCH_CLIENT_ID) für einen externen Dienst zur späteren Verwendung hinzu.

```
build:
  commands:
    - npm run build:$BUILD_ENV
    - echo "TWITCH_CLIENT_ID=$TWITCH_CLIENT_ID" >> backend/.env
```

5. Wählen Sie Speichern.

Umgebungsvariablen für serverseitige Laufzeiten zugänglich machen

Eine Serverkomponente von Next.js hat standardmäßig keinen Zugriff auf die Umgebungsvariablen Ihrer App. Dieses Verhalten dient dazu, alle Geheimnisse zu schützen, die in Umgebungsvariablen gespeichert sind, die Ihre Anwendung während der Erstellungsphase verwendet.

Um bestimmte Umgebungsvariablen für Next.js zugänglich zu machen, müssen Sie die Amplify-Build-Spezifikationsdatei ändern, um die Umgebungsvariablen in den Umgebungsdateien festzulegen, die Next.js erkennt. Dadurch kann Amplify die Umgebungsvariablen laden, bevor die Anwendung erstellt wird. Weitere Informationen zum Ändern Ihrer Build-Spezifikation finden Sie in den Beispielen zum [Hinzufügen von Umgebungsvariablen im Abschnitt Build-Befehle](#).

Erstellen Sie eine neue Backend-Umgebung mit Authentifizierungsparametern für die Anmeldung über soziale Netzwerke

Um eine Filiale mit einer App zu verbinden

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Das Verfahren zum Verbinden einer Filiale mit einer App hängt davon ab, ob Sie eine Filiale mit einer neuen App oder einer vorhandenen App verbinden.
 - Eine Filiale mit einer neuen App verbinden

- a. Suchen Sie auf der Seite mit den Build-Einstellungen den Abschnitt Wählen Sie eine Backend-Umgebung aus, die mit diesem Zweig verwendet werden soll. Wählen Sie für Umgebung die Option Neue Umgebung erstellen aus und geben Sie den Namen Ihrer Backend-Umgebung ein. Der folgende Screenshot zeigt den Abschnitt Wählen Sie eine Backend-Umgebung aus, die mit diesem Zweig verwendet werden soll auf der Seite mit den Build-Einstellungen, in dem für die Backend-Umgebung ein Name **backend** eingegeben wurde.

Select a backend environment to use with this branch

App name

Environment

If you don't provide a value in this field, your branch name will be used by default.

Enable full-stack continuous deployments (CI/CD)
Full-stack CI/CD allows you to continuously deploy frontend and backend changes on every code commit

Select an existing service role or create a new one so Amplify Hosting may access your resources.

i Create a new service role. In the window that opens, accept the pre-selected defaults on each screen to create a new service role.

- b. Erweitern Sie den Abschnitt Erweiterte Einstellungen auf der Seite mit den Build-Einstellungen und fügen Sie Umgebungsvariablen für Social Sign-In-Schlüssel hinzu. **AMPLIFY_FACEBOOK_CLIENT_SECRET** ist beispielsweise eine gültige Umgebungsvariable. Eine Liste der standardmäßig verfügbaren Amplify-Systemumgebungsvariablen finden Sie in [Umgebungsvariablen Amplify](#) der Tabelle unter.
 - Einen Zweig mit einer vorhandenen App verbinden
 - a. Wenn Sie eine neue Filiale mit einer vorhandenen App verbinden, legen Sie die Umgebungsvariablen für die Anmeldung über soziale Netzwerke fest, bevor Sie die Filiale verbinden. Wählen Sie im Navigationsbereich App-Einstellungen, Umgebungsvariablen aus.
 - b. Wählen Sie im Abschnitt Umgebungsvariablen die Option Variablen verwalten aus.
 - c. Wählen Sie im Abschnitt Variablen verwalten die Option Variable hinzufügen aus.

- d. Geben Sie unter Variable (Schlüssel) Ihre Client-ID ein. Geben Sie unter Value Ihr Client-Geheimnis ein.
- e. Wählen Sie „Speichern“.

Umgebungsvariablen des Frontend-Frameworks

Wenn Sie Ihre App mit einem Frontend-Framework entwickeln, das seine eigenen Umgebungsvariablen unterstützt, ist es wichtig zu verstehen, dass diese nicht mit den Umgebungsvariablen identisch sind, die Sie in der Amplify-Konsole konfigurieren. Mit React (mit dem Präfix REACT_APP) und Gatsby (mit dem Präfix GATSBY) können Sie beispielsweise Laufzeitumgebungsvariablen erstellen, die diese Frameworks automatisch in Ihren Frontend-Produktions-Build bündeln. Informationen zu den Auswirkungen der Verwendung dieser Umgebungsvariablen zum Speichern von Werten finden Sie in der Dokumentation für das von Ihnen verwendete Frontend-Framework.

Das Speichern sensibler Werte, wie API-Schlüssel, in diesen Umgebungsvariablen des Frontend-Frameworks mit Präfix ist keine bewährte Methode und es wird dringend davon abgeraten. Ein Beispiel für die Verwendung der Build-Time-Umgebungsvariablen von Amplify für diesen Zweck finden Sie unter [Greifen Sie während der Erstellung auf Umgebungsvariablen zu](#)

Verwaltung von Umgebungsgeheimnissen

Mit der Veröffentlichung von Amplify Gen 2 wurde der Workflow für Umgebungsgeheimnisse optimiert, um die Verwaltung von Geheimnissen und Umgebungsvariablen in der Amplify-Konsole zu zentralisieren. Anweisungen zum Einrichten und Zugreifen auf Geheimnisse für eine Amplify Gen 2-App finden Sie unter [Secrets and environment vars](#) in der Amplify-Dokumentation.

Umgebungsgeheimnisse für eine Gen-1-App ähneln Umgebungsvariablen, es handelt sich jedoch um Schlüsselwertpaare für den AWS Systems Manager Parameterspeicher, die verschlüsselt werden können. Einige Werte müssen verschlüsselt werden, z. B. die Anmeldung mit dem privaten Apple-Schlüssel für Amplify.

Legen Sie Umgebungsgeheimnisse für eine Gen-1-App fest und greifen Sie darauf zu

Verwenden Sie die folgenden Anweisungen, um mithilfe der AWS Systems Manager Konsole ein Umgebungsgeheimnis für eine Amplify-App der Generation 1 festzulegen.

Um ein Umgebungsgeheimnis festzulegen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [AWS Systems Manager Konsole](#).
2. Wählen Sie im Navigationsbereich Application Management und dann Parameter Store aus.
3. Wählen Sie auf der Seite AWS Systems Manager Parameter Store die Option Create parameter aus.
4. Gehen Sie auf der Seite Parameter erstellen im Abschnitt Parameterdetails wie folgt vor:
 - a. Geben Sie für Name einen Parameter im Format `an/amplify/{your_app_id}/{your_backend_environment_name}/{your_parameter_name}`.
 - b. Wählen Sie für Type (Typ) die Option SecureString aus.
 - c. Wählen Sie unter KMS-Schlüsselquelle die Option Mein aktuelles Konto aus, um den Standardschlüssel für Ihr Konto zu verwenden.
 - d. Geben Sie unter Wert Ihren geheimen Wert für die Verschlüsselung ein.
5. Wählen Sie „Parameter erstellen“.

Note

Amplify hat nur Zugriff auf die Schlüssel unter dem Build `/amplify/{your_app_id}/{your_backend_environment_name}` für die spezifische Umgebung. Sie müssen die Standardeinstellung angeben AWS KMS key , damit Amplify den Wert entschlüsseln kann.

Auf Geheimnisse der Umgebung zugreifen

Der Zugriff auf ein Umgebungsgeheimnis für eine Gen-1-App während eines Builds ähnelt dem [Zugriff auf Umgebungsvariablen](#), außer dass Umgebungsgeheimnisse `process.env.secrets` als JSON-Zeichenfolge gespeichert werden.

Amplify Sie die Geheimnisse der Umgebung

Geben Sie einen Systems Manager Manager-Parameter im Format `an/amplify/{your_app_id}/{your_backend_environment_name}/AMPLIFY_SIWA_CLIENT_ID`.

Sie können die folgenden Umgebungsgeheimnisse verwenden, auf die standardmäßig in der Amplify-Konsole zugegriffen werden kann.

Variablenname	Beschreibung	Beispielwert
AMPLIFY_SIWA_CLIENT_ID	Die Anmeldung mit der Apple-Client-ID	com.yourapp.auth
AMPLIFY_SIWA_TEAM_ID	Das Melden Sie sich mit der Apple-Team-ID an	ABCD123
AMPLIFY_SIWA_KEY_ID	Die Anmeldung mit der Apple-Schlüssel-ID	ABCD123
AMPLIFY_SIWA_PRIVATE_KEY	Die Anmeldung mit dem privaten Apple-Schlüssel	-----BEGINNE MIT DEM PRIVATEN SCHLÜSSEL----- **** -----ENDE DES PRIVATEN SCHLÜSSELS-----

Benutzerdefinierte Header

Mit benutzerdefinierten HTTP-Headern können Sie Header für jede HTTP-Antwort angeben. Response-Header können für Debugging-, Sicherheits- und Informationszwecke verwendet werden. Sie können Header in der Amplify-Konsole angeben oder indem Sie die `customHttp.yml` Datei einer App herunterladen, bearbeiten und im Stammverzeichnis des Projekts speichern. Die detaillierten Schritte finden Sie unter [Benutzerdefinierte Header einrichten](#).

Bisher wurden benutzerdefinierte HTTP-Header für eine App angegeben, indem entweder die Build-Spezifikation (Buildspec) in der bearbeitet wurde AWS Management Console oder indem die `amplify.yml` Datei heruntergeladen und aktualisiert und im Stammverzeichnis des Projekts gespeichert wurde. Auf diese Weise angegebene benutzerdefinierte Header sollten aus der Buildspec und der Datei migriert werden. `amplify.yml` Anweisungen finden Sie unter [Migrieren von benutzerdefinierten Headern](#).

Benutzerdefiniertes Header-YAML-Format

Geben Sie benutzerdefinierte Header im folgenden YAML-Format an:

```
customHeaders:
  - pattern: '*.json'
    headers:
      - key: 'custom-header-name-1'
        value: 'custom-header-value-1'
      - key: 'custom-header-name-2'
        value: 'custom-header-value-2'
  - pattern: '/path/*'
    headers:
      - key: 'custom-header-name-1'
        value: 'custom-header-value-2'
```

Verwenden Sie für ein Monorepo das folgende YAML-Format:

```
applications:
  - appRoot: app1
    customHeaders:
      - pattern: '**/*'
        headers:
          - key: 'custom-header-name-1'
```

```
    value: 'custom-header-value-1'  
  - appRoot: app2  
    customHeaders:  
      - pattern: '/path/*.json'  
        headers:  
          - key: 'custom-header-name-2'  
            value: 'custom-header-value-2'
```

Wenn Sie Ihrer App benutzerdefinierte Header hinzufügen, geben Sie Ihre eigenen Werte für Folgendes an:

pattern

Benutzerdefinierte Header werden auf alle URL-Dateipfade angewendet, die dem Muster entsprechen.

Header

Definiert die Header, die dem Dateimuster entsprechen.

Schlüssel

Der Name des benutzerdefinierten Headers.

Wert

Der Wert des benutzerdefinierten Headers.

Weitere Informationen zu HTTP-Headern finden Sie in Mozillas Liste der [HTTP-Header](#).

Benutzerdefinierte Header einrichten

Es gibt zwei Möglichkeiten, benutzerdefinierte HTTP-Header für eine Amplify-App anzugeben. Sie können Header in der Amplify-Konsole angeben oder Sie können Header angeben, indem Sie die `customHttp.yml` Datei einer App herunterladen, bearbeiten und im Stammverzeichnis Ihres Projekts speichern.

Um benutzerdefinierte Header für eine App festzulegen und sie in der Konsole zu speichern

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die Sie benutzerdefinierte Header festlegen möchten.

3. Wählen Sie im Navigationsbereich Hosting und anschließend Benutzerdefinierte Header aus.
4. Wählen Sie auf der Seite Benutzerdefinierte Kopfzeilen die Option Bearbeiten aus.
5. Geben Sie im Fenster Benutzerdefinierte Header bearbeiten die Informationen für Ihre benutzerdefinierten Header im YAML-Format für [benutzerdefinierte Header](#) ein.
 - a. Geben Sie für `pattern` das passende Muster ein.
 - b. Geben Sie für `key` den Namen der benutzerdefinierten Kopfzeile ein.
 - c. Geben Sie für `value` den Wert des benutzerdefinierten Headers ein.
6. Wählen Sie Speichern.
7. Stellen Sie die App erneut bereit, um die neuen benutzerdefinierten Header anzuwenden.
 - Navigieren Sie bei einer CI/CD-App zu dem Zweig, den Sie bereitstellen möchten, und wählen Sie Diese Version erneut bereitstellen aus. Sie können auch einen neuen Build von Ihrem Git-Repository aus ausführen.
 - Für eine App zur manuellen Bereitstellung stellen Sie die App erneut in der Amplify-Konsole bereit.

Um benutzerdefinierte Header für eine App festzulegen und sie im Stammverzeichnis Ihres Repositories zu speichern

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die Sie benutzerdefinierte Header festlegen möchten.
3. Wählen Sie im Navigationsbereich Hosting und anschließend Benutzerdefinierte Header aus.
4. Wählen Sie auf der Seite Benutzerdefinierte Header die Option YML herunterladen aus.
5. Öffnen Sie die heruntergeladene `customHttp.yml` Datei im Code-Editor Ihrer Wahl und geben Sie die Informationen für Ihre benutzerdefinierten Header im YAML-Format für [benutzerdefinierte Header](#) ein.
 - a. Geben Sie für `pattern` das passende Muster ein.
 - b. Geben Sie für `key` den Namen der benutzerdefinierten Kopfzeile ein.
 - c. Geben Sie für `value` den Wert des benutzerdefinierten Headers ein.
6. Speichern Sie die bearbeitete `customHttp.yml` Datei im Stammverzeichnis Ihres Projekts. Wenn Sie mit einem Monorepo arbeiten, speichern Sie die `customHttp.yml` Datei im Stammverzeichnis Ihres Repos.
7. Stellen Sie die App erneut bereit, um die neuen benutzerdefinierten Header anzuwenden.

- Führen Sie für eine CI/CD-App einen neuen Build aus Ihrem Git-Repository aus, der die neue `customHttp.yml` Datei enthält.
- Für eine App mit manueller Bereitstellung stellen Sie die App erneut in der Amplify-Konsole bereit und fügen Sie die neue `customHttp.yml` Datei mit den Artefakten hinzu, die Sie hochladen.

Note

Benutzerdefinierte Header, die in der `customHttp.yml` Datei festgelegt und im Stammverzeichnis der App bereitgestellt werden, überschreiben benutzerdefinierte Header, die im Abschnitt Benutzerdefinierte Header in der Amplify-Konsole definiert sind.

Migrieren von benutzerdefinierten Headern

Bisher wurden benutzerdefinierte HTTP-Header für eine App entweder durch Bearbeitung der Buildspec in der Amplify-Konsole oder durch Herunterladen und Aktualisieren der `amplify.yml` Datei und Speichern im Stammverzeichnis des Projekts angegeben. Es wird dringend empfohlen, dass Sie Ihre benutzerdefinierten Header aus der Buildspec und der Datei migrieren. `amplify.yml`

Geben Sie Ihre benutzerdefinierten Header im Abschnitt Benutzerdefinierte Header der Amplify-Konsole an oder indem Sie die Datei herunterladen und bearbeiten. `customHttp.yml`

Um benutzerdefinierte Header zu migrieren, die in der Amplify-Konsole gespeichert sind

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die die benutzerdefinierte Header-Migration durchgeführt werden soll.
3. Wählen Sie im Navigationsbereich Hosting, Build-Einstellungen aus. Im Abschnitt App-Build-Spezifikation können Sie die Buildspec Ihrer App überprüfen.
4. Wählen Sie Herunterladen, um eine Kopie Ihrer aktuellen Buildspec zu speichern. Sie können später auf diese Kopie verweisen, wenn Sie Einstellungen wiederherstellen müssen.
5. Wenn der Download abgeschlossen ist, wählen Sie Bearbeiten.
6. Notieren Sie sich die benutzerdefinierten Header-Informationen in der Datei, da Sie sie später in Schritt 9 verwenden werden. Löschen Sie im Bearbeitungsfenster alle benutzerdefinierten Header aus der Datei und wählen Sie Speichern.

7. Wählen Sie im Navigationsbereich Hosting, Benutzerdefinierte Header aus.
8. Wählen Sie auf der Seite Benutzerdefinierte Header die Option Bearbeiten aus.
9. Geben Sie im Fenster Benutzerdefinierte Kopfzeilen bearbeiten die Informationen für Ihre benutzerdefinierten Kopfzeilen ein, die Sie in Schritt 6 gelöscht haben.
10. Wählen Sie Speichern.
11. Stellen Sie alle Branches erneut bereit, auf die die neuen benutzerdefinierten Header angewendet werden sollen.

Um benutzerdefinierte Header von `amplify.yml` zu `CustomHttp.yml` zu migrieren

1. Navigieren Sie zu der `amplify.yml` Datei, die derzeit im Stammverzeichnis Ihrer App bereitgestellt ist.
2. Öffnen Sie `amplify.yml` im Code-Editor Ihrer Wahl.
3. Notieren Sie sich die benutzerdefinierten Header-Informationen in der Datei, da Sie sie später in Schritt 8 verwenden werden. Löschen Sie die benutzerdefinierten Header in der Datei. Speichern und schließen Sie die Datei.
4. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
5. Wählen Sie die App aus, für die Sie benutzerdefinierte Header festlegen möchten.
6. Wählen Sie im Navigationsbereich Hosting, Benutzerdefinierte Header aus.
7. Wählen Sie auf der Seite Benutzerdefinierte Header die Option Herunterladen aus.
8. Öffnen Sie die heruntergeladene `customHttp.yml` Datei im Code-Editor Ihrer Wahl und geben Sie die Informationen für Ihre benutzerdefinierten Header ein, aus denen Sie `amplify.yml` in Schritt 3 gelöscht haben.
9. Speichern Sie die bearbeitete `customHttp.yml` Datei im Stammverzeichnis Ihres Projekts. Wenn Sie mit einem Monorepo arbeiten, speichern Sie die Datei im Stammverzeichnis Ihres Repos.
10. Stellen Sie die App erneut bereit, um die neuen benutzerdefinierten Header anzuwenden.
 - Führen Sie für eine CI/CD-App einen neuen Build aus Ihrem Git-Repository aus, der die neue `customHttp.yml` Datei enthält.
 - Für eine App mit manueller Bereitstellung stellen Sie die App erneut in der Amplify-Konsole bereit und fügen Sie die neue `customHttp.yml` Datei mit Artefakten hinzu, die Sie hochladen.

Note

Benutzerdefinierte Header, die in der `customHttp.yml` Datei festgelegt und im Stammverzeichnis der App bereitgestellt werden, überschreiben die benutzerdefinierten Header, die im Abschnitt Benutzerdefinierte Header der Amplify-Konsole definiert sind.

Benutzerdefinierte Monorepo-Header

Wenn Sie benutzerdefinierte Header für eine App in einem Monorepo angeben, beachten Sie die folgenden Einrichtungsanforderungen:

- Es gibt ein spezielles YAML-Format für ein Monorepo. Die korrekte Syntax finden Sie unter [Benutzerdefiniertes Header-YAML-Format](#)
- Sie können benutzerdefinierte Header für eine Anwendung in einem Monorepo mithilfe des Abschnitts Benutzerdefinierte Header der Amplify-Konsole angeben. Sie müssen Ihre Anwendung erneut bereitstellen, um die neuen benutzerdefinierten Header anzuwenden.
- Als Alternative zur Verwendung der Konsole können Sie benutzerdefinierte Header für eine App in einem Monorepo in einer Datei angeben. `customHttp.yml` Sie müssen die `customHttp.yml` Datei im Stammverzeichnis Ihres Repos speichern und dann die Anwendung erneut bereitstellen, um die neuen benutzerdefinierten Header anzuwenden. In der `customHttp.yml` Datei angegebene benutzerdefinierte Header überschreiben alle benutzerdefinierten Header, die im Abschnitt Benutzerdefinierte Header der Amplify-Konsole angegeben wurden.

Beispiel für Sicherheitsheader

Benutzerdefinierte Sicherheitsheader ermöglichen die Durchsetzung von HTTPS, die Verhinderung von XSS-Angriffen und den Schutz Ihres Browsers vor Clickjacking. Verwenden Sie die folgende YAML-Syntax, um benutzerdefinierte Sicherheitsheader auf Ihre App anzuwenden.

```
customHeaders:
  - pattern: '**'
    headers:
      - key: 'Strict-Transport-Security'
        value: 'max-age=31536000; includeSubDomains'
      - key: 'X-Frame-Options'
        value: 'SAMEORIGIN'
```

```
- key: 'X-XSS-Protection'  
  value: '1; mode=block'  
- key: 'X-Content-Type-Options'  
  value: 'nosniff'  
- key: 'Content-Security-Policy'  
  value: "default-src 'self'"
```

Benutzerdefinierte Cache-Control-Header

Apps, die mit Amplify gehostet werden, berücksichtigen die vom Ursprung gesendeten Cache-Control Header, es sei denn, Sie überschreiben sie mit benutzerdefinierten Headern, die Sie definieren. Amplify wendet benutzerdefinierte Cache-Control-Header nur für erfolgreiche Antworten mit einem Statuscode an. 200 OK Dadurch wird verhindert, dass Fehlerantworten zwischengespeichert und anderen Benutzern, die dieselbe Anfrage stellen, zugestellt werden.

Sie können die `s-maxage` Direktive manuell anpassen, um mehr Kontrolle über die Leistung und Bereitstellungsverfügbarkeit Ihrer App zu haben. Um beispielsweise die Dauer zu verlängern, für die Ihre Inhalte am Edge zwischengespeichert bleiben, können Sie die Gültigkeitsdauer (Time to Live, TTL) manuell erhöhen, indem Sie `s-maxage` auf einen Wert aktualisieren, der länger als die Standardeinstellung 600 Sekunden (10 Minuten) ist.

Verwenden Sie das folgende YAML-Formats `-maxage`, um einen benutzerdefinierten Wert für anzugeben. In diesem Beispiel wird der zugehörige Inhalt 3600 Sekunden (eine Stunde) lang am Edge zwischengespeichert.

```
customHeaders:  
  - pattern: '/img/'  
    headers:  
      - key: 'Cache-Control'  
        value: 's-maxage=3600'
```

Weitere Informationen zur Steuerung der Anwendungsleistung mithilfe von Headern finden Sie unter [Verwenden von Headern zur Steuerung der Cache-Dauer](#)

Eingehende Webhooks

Richten Sie in der Amplify-Konsole einen eingehenden Webhook ein, um einen Build zu starten, ohne Code in Ihr Git-Repository zu übertragen. Sie können Webhook-Trigger mit Headless-CMS-Tools (wie Contentful oder GraphCMS) verwenden, um einen Build zu starten, wenn sich der Inhalt ändert, oder um tägliche Builds mit Diensten wie Zapier durchzuführen.

Um einen eingehenden Webhook zu erstellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die Sie einen Webhook erstellen möchten.
3. Wählen Sie im Navigationsbereich Hosting und dann Build-Einstellungen aus.
4. Scrollen Sie auf der Seite mit den Build-Einstellungen nach unten zum Abschnitt Eingehende Webhooks und wählen Sie Webhook erstellen aus.
5. Gehen Sie im Dialogfeld „Webhook erstellen“ wie folgt vor:
 - a. Geben Sie unter Webhook-Name einen Namen für den Webhook ein.
 - b. Wählen Sie für Branch to build den Branch aus, der auf eingehenden Webhook-Anfragen aufgebaut werden soll.
 - c. Wählen Sie Webhook erstellen.
6. Führen Sie im Abschnitt Eingehende Webhooks einen der folgenden Schritte aus:
 - Kopieren Sie die Webhook-URL und stellen Sie sie einem Headless-CMS-Tool oder einem anderen Dienst zur Verfügung, um Builds zu initiieren.
 - Führen Sie den Befehl curl in einem Terminalfenster aus, um einen neuen Build zu starten.

Überwachen

AWS Amplify sendet Metriken über Amazon aus CloudWatch und stellt Zugriffsprotokolle mit detaillierten Informationen zu Anfragen an Ihre App bereit. In den Themen in diesem Abschnitt erfahren Sie, wie Sie diese Metriken und Protokolle zur Überwachung Ihrer App verwenden können.

Themen

- [Überwachung mit CloudWatch](#)
- [Zugriffsprotokolle](#)

Überwachung mit CloudWatch

AWS Amplify ist in Amazon integriert CloudWatch, sodass Sie die Metriken für Ihre Amplify-Anwendungen nahezu in Echtzeit überwachen können. Sie können Alarmer erstellen, die Benachrichtigungen senden, wenn eine Metrik einen von Ihnen festgelegten Schwellenwert überschreitet. Weitere Informationen zur Funktionsweise des CloudWatch Service finden Sie im [CloudWatch Amazon-Benutzerhandbuch](#).

Metriken

Amplify unterstützt sechs CloudWatch Metriken im `AWS/AmplifyHosting` Namespace zur Überwachung von Verkehr, Fehlern, Datenübertragung und Latenz für Ihre Apps. Diese Metriken werden in Intervallen von einer Minute aggregiert. CloudWatch Die Monitoring-Metriken sind kostenlos und werden nicht auf die [CloudWatch Servicequoten](#) angerechnet.

Nicht alle verfügbaren Statistiken gelten für jede Metrik. In der folgenden Tabelle sind die relevantesten Statistiken in der Beschreibung der einzelnen Metriken aufgeführt.

Metriken	Beschreibung
Anforderungen	Die Gesamtzahl der Zuschaueranfragen, die Ihre App erhalten hat. Die relevanteste Statistik ist <code>Sum</code> . Verwenden Sie die <code>Sum</code> Statistik, um die Gesamtzahl der Anfragen zu ermitteln.

Metriken	Beschreibung
BytesDownloaded	<p>Die Gesamtmenge der Daten, die von Zuschauern für, und OPTIONS Anfragen aus Ihrer App übertragen (heruntergeladen) wurden GETHEAD, in Byte.</p> <p>Die relevanteste Statistik istSum.</p>
BytesUploaded	<p>Die Gesamtmenge der Daten, die mithilfe von POST und PUT Anfragen in Ihre App übertragen (hochgeladen) wurden, in Byte.</p> <p>Die relevanteste Statistik istSum.</p>
4xxErrors	<p>Die Anzahl der Anfragen, bei denen ein Fehler im Bereich des HTTP-Statuscodes 400-499 zurückgegeben wurde.</p> <p>Die relevanteste Statistik ist. Sum Verwenden Sie die Sum Statistik, um die Gesamthäufigkeit dieser Fehler zu ermitteln.</p>
5xxErrors	<p>Die Anzahl der Anfragen, bei denen ein Fehler im Bereich des HTTP-Statuscodes 500-599 zurückgegeben wurde.</p> <p>Die relevanteste Statistik ist. Sum Verwenden Sie die Sum Statistik, um die Gesamthäufigkeit dieser Fehler zu ermitteln.</p>

Metriken	Beschreibung
Latency	<p>Die Zeit bis zum ersten Byte in Sekunden. Dies ist die Gesamtzeit zwischen dem Empfang einer Anfrage durch Amplify Hosting und der Rückgabe einer Antwort an das Netzwerk. Dies beinhaltet nicht die Netzwerklatenz, die auftritt, wenn eine Antwort das Gerät des Betrachters erreicht.</p> <p>Die relevantesten Statistiken sind AverageMaximum,Minimum,p10,p50,p90,p95, undp100.</p> <p>Verwenden Sie die Average Statistik, um die erwarteten Latenzen zu bewerten.</p>

Amplify bietet die folgenden CloudWatch metrischen Abmessungen.

Dimension	Beschreibung
App	Metrische Daten werden per App bereitgestellt.
AWS-Konto	Metrische Daten werden für alle Apps in der bereitgestellt AWS-Konto.

Sie können auf CloudWatch Metriken AWS Management Console unter <https://console.aws.amazon.com/cloudwatch/> zugreifen. Alternativ können Sie mit dem folgenden Verfahren auf Metriken in der Amplify-Konsole zugreifen.

So greifen Sie in der Amplify-Konsole auf Metriken zu

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die Sie Metriken anzeigen möchten.
3. Wählen Sie im Navigationsbereich App-Einstellungen, Überwachung aus.
4. Wählen Sie auf der Seite Überwachung die Option Metriken aus.

Alarme

Sie können in der Amplify-Konsole CloudWatch Alarme erstellen, die Benachrichtigungen senden, wenn bestimmte Kriterien erfüllt sind. Ein Alarm überwacht eine einzelne CloudWatch Metrik und sendet eine Amazon Simple Notification Service-Benachrichtigung, wenn die Metrik den Schwellenwert für eine bestimmte Anzahl von Bewertungszeiträumen überschreitet.

Sie können erweiterte Alarme erstellen, die metrische mathematische Ausdrücke in der CloudWatch Konsole oder mithilfe der CloudWatch APIs verwenden. Sie können beispielsweise einen Alarm erstellen, der Sie benachrichtigt, wenn der Prozentsatz in drei aufeinanderfolgenden Perioden 15% 4XXErrors überschreitet. Weitere Informationen finden Sie unter [Erstellen eines CloudWatch Alarms auf der Grundlage eines metrischen mathematischen Ausdrucks](#) im CloudWatch Amazon-Benutzerhandbuch.

Für Alarme gelten die CloudWatch Standardpreise. Weitere Informationen finden Sie unter [CloudWatchAmazon-Preise](#).

Gehen Sie wie folgt vor, um einen Alarm in der Amplify-Konsole zu erstellen.

Um einen CloudWatch Alarm für eine Amplify-Metrik zu erstellen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die Sie einen Alarm einrichten möchten.
3. Wählen Sie im Navigationsbereich App-Einstellungen, Überwachung aus.
4. Wählen Sie auf der Seite Überwachung die Option Alarme aus.
5. Wählen Sie Alarm erstellen aus.
6. Konfigurieren Sie Ihren Alarm im Fenster Alarm erstellen wie folgt:
 - a. Wählen Sie unter Metrik den Namen der zu überwachenden Metrik aus der Liste aus.
 - b. Geben Sie unter Name des Alarms einen aussagekräftigen Namen für den Alarm ein. Wenn Sie beispielsweise Anfragen überwachen, könnten Sie dem Alarm einen Namen geben **HighTraffic**. Der Name darf nur ASCII-Zeichen enthalten.
 - c. Gehen Sie für Benachrichtigungen einrichten wie folgt vor:
 - i. Wählen Sie Neu, um ein neues Amazon SNS SNS-Thema einzurichten.
 - ii. Geben Sie unter E-Mail-Adresse die E-Mail-Adresse des Empfängers der Benachrichtigungen ein.

- iii. Wählen Sie Neue E-Mail-Adresse hinzufügen, um weitere Empfänger hinzuzufügen.
- - i. Wählen Sie Existing, um ein Amazon SNS SNS-Thema wiederzuverwenden.
 - ii. Wählen Sie unter SNS-Thema den Namen eines vorhandenen Amazon SNS SNS-Themas aus der Liste aus.
- d. Stellen Sie für Whenever the Statistic of Metric die Bedingungen für Ihren Alarm wie folgt ein:
 - i. Geben Sie an, ob die Metrik größer, kleiner oder gleich dem Schwellenwert sein muss.
 - ii. Geben Sie den Schwellenwert an.
 - iii. Geben Sie die Anzahl der aufeinanderfolgenden Evaluierungsperioden an, die sich im Alarmstatus befinden müssen, um den Alarm auszulösen.
 - iv. Geben Sie die Dauer des Evaluierungszeitraums an.
- e. Wählen Sie Alarm erstellen aus.

Note

Jeder Amazon SNS SNS-Empfänger, den Sie angeben, erhält eine Bestätigungs-E-Mail von AWS Notifications. Die E-Mail enthält einen Link, dem der Empfänger folgen muss, um sein Abonnement zu bestätigen und Benachrichtigungen zu erhalten.

Amazon CloudWatch Logs für SSR-Apps

Amplify sendet Informationen über Ihre Laufzeit von Next.js an Amazon CloudWatch Logs in Ihrem AWS-Konto. Wenn Sie eine SSR-App bereitstellen, benötigt die App eine IAM-Dienstrolle, die Amplify übernimmt, wenn es andere Dienste in Ihrem Namen aufruft. Sie können entweder Amplify Hosting Compute erlauben, automatisch eine Servicerolle für Sie zu erstellen, oder Sie können eine Rolle angeben, die Sie erstellt haben.

Wenn Sie Amplify erlauben, eine IAM-Rolle für Sie zu erstellen, verfügt die Rolle bereits über die Berechtigungen zum Erstellen CloudWatch von Protokollen. Wenn Sie Ihre eigene IAM-Rolle erstellen, müssen Sie Ihrer Richtlinie die folgenden Berechtigungen hinzufügen, damit Amplify auf Amazon CloudWatch Logs zugreifen kann.

```
logs:CreateLogStream
```

```
logs:CreateLogGroup
logs:DescribeLogGroups
logs:PutLogEvents
```

Weitere Informationen zu Servicerollen finden Sie unter [Eine Servicerolle hinzufügen](#). Weitere Informationen zur Bereitstellung serverseitig gerendeter Apps finden Sie unter [Stellen Sie serverseitig gerenderte Apps mit Amplify Hosting bereit](#)

Zugriffsprotokolle

Amplify speichert Zugriffsprotokolle für alle Apps, die Sie in Amplify hosten. Zugriffsprotokolle enthalten Informationen über Anfragen, die an Ihre gehosteten Apps gestellt werden. Amplify speichert alle Zugriffsprotokolle für eine App, bis Sie die App löschen. Alle Zugriffsprotokolle für eine App sind in der Amplify-Konsole verfügbar. Jede einzelne Anfrage nach Zugriffsprotokollen ist jedoch auf einen von Ihnen angegebenen Zeitraum von zwei Wochen begrenzt.

Amplify verwendet niemals CloudFront Distributionen zwischen Kunden wieder. Amplify erstellt CloudFront Distributionen im Voraus, sodass Sie bei der Bereitstellung einer neuen App nicht darauf warten müssen, dass eine CloudFront Distribution erstellt wird. Bevor diese Distributionen einer Amplify-App zugewiesen werden, erhalten sie möglicherweise Traffic von Bots. Sie sind jedoch so konfiguriert, dass sie immer als Nicht gefunden antworten, bevor sie zugewiesen werden. Wenn die Zugriffsprotokolle Ihrer App Einträge für einen Zeitraum enthalten, bevor Sie Ihre App erstellt haben, beziehen sich diese Einträge auf diese Aktivität.

Important

Wir empfehlen, dass Sie die Protokolle verwenden, um die Art der Anfragen für Ihre Inhalte zu verstehen, nicht als eine vollständige Buchführung aller Anfragen. Amplify liefert Zugriffsprotokolle nach bestem Wissen. Der Protokolleintrag für eine bestimmte Anfrage wird möglicherweise viel später übermittelt, als die Anfrage tatsächlich verarbeitet wurde; in seltenen Fällen kann es auch sein, dass ein Protokolleintrag gar nicht übermittelt wird. Wenn ein Protokolleintrag in den Zugriffsprotokollen weggelassen wird, entspricht die Anzahl der Einträge in den Zugriffsprotokollen nicht der Nutzung, die in den AWS Abrechnungs- und Nutzungsberichten angegeben ist.

Gehen Sie wie folgt vor, um Zugriffsprotokolle für eine App abzurufen.

Um Zugriffsprotokolle anzuzeigen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die Sie Zugriffsprotokolle anzeigen möchten.
3. Wählen Sie im Navigationsbereich Hosting und anschließend Überwachung aus.
4. Wählen Sie auf der Seite Überwachung die Option Zugriffsprotokolle aus.
5. Wählen Sie Zeitraum bearbeiten aus.
6. Gehen Sie im Fenster Zeitraum bearbeiten wie folgt vor.
 - a. Geben Sie als Startdatum den ersten Tag des zweiwöchigen Intervalls an, für das Protokolle abgerufen werden sollen.
 - b. Wählen Sie unter Startzeit die Uhrzeit am ersten Tag aus, an dem der Protokollabruf gestartet werden soll.
 - c. Wählen Sie Bestätigen aus.
7. Die Amplify-Konsole zeigt die Protokolle für den angegebenen Zeitraum im Abschnitt Zugriffsprotokolle an. Wählen Sie Herunterladen, um die Protokolle im CSV-Format zu speichern.

Analyse von Zugriffsprotokollen

Um Zugriffsprotokolle zu analysieren, können Sie die CSV-Dateien in einem Amazon S3 S3-Bucket speichern. Eine Möglichkeit, Ihre Zugriffsprotokolle zu analysieren, ist die Verwendung von Athena. Athena ist ein interaktiver Abfragedienst, mit dem Sie Daten für AWS Dienste analysieren können. Sie können den [step-by-step Anweisungen hier](#) folgen, um eine Tabelle zu erstellen. Sobald Ihre Tabelle erstellt wurde, können Sie Daten wie folgt abfragen.

```
SELECT SUM(bytes) AS total_bytes
FROM logs
WHERE "date" BETWEEN DATE '2018-06-09' AND DATE '2018-06-11'
LIMIT 100;
```


E-Mail-Benachrichtigungen für Builds

Sie können E-Mail-Benachrichtigungen für eine AWS Amplify App einrichten, um Stakeholder oder Teammitglieder zu benachrichtigen, wenn ein Build erfolgreich ist oder fehlschlägt. Amplify Hosting erstellt ein Amazon Simple Notification Service (SNS) -Thema in Ihrem Konto und verwendet es zur Konfiguration von E-Mail-Benachrichtigungen. Benachrichtigungen können so konfiguriert werden, dass sie für alle Zweige oder bestimmte Zweige einer Amplify-App gelten.

Richten Sie E-Mail-Benachrichtigungen ein

Verwenden Sie die folgenden Verfahren, um E-Mail-Benachrichtigungen für alle Filialen oder bestimmte Zweige einer Amplify-App einzurichten.

So richten Sie E-Mail-Benachrichtigungen für eine Amplify-App ein

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die Sie E-Mail-Benachrichtigungen einrichten möchten.
3. Wählen Sie im Navigationsbereich Hosting, Benachrichtigungen erstellen aus. Wählen Sie auf der Seite Benachrichtigungen erstellen die Option Benachrichtigungen verwalten aus.
4. Wählen Sie auf der Seite Benachrichtigungen verwalten die Option Neue hinzufügen aus.
5. Führen Sie eine der folgenden Aktionen aus:
 - Um Benachrichtigungen für eine einzelne Filiale zu senden, geben Sie unter E-Mail die E-Mail-Adresse ein, an die Benachrichtigungen gesendet werden sollen. Wählen Sie unter Filiale den Namen der Filiale aus, für die Benachrichtigungen gesendet werden sollen.
 - Um Benachrichtigungen für alle verbundenen Filialen zu senden, geben Sie unter E-Mail die E-Mail-Adresse ein, an die Benachrichtigungen gesendet werden sollen. Wählen Sie für Filiale die Option Alle Filialen aus.
6. Klicken Sie auf Speichern.

Benutzerdefinierte Build-Images und Live-Paket-Updates

Themen

- [Benutzerdefinierte Build-Images](#)
- [Live-Paket-Updates](#)

Benutzerdefinierte Build-Images

Sie können ein benutzerdefiniertes Build-Image verwenden, um eine benutzerdefinierte Build-Umgebung für eine Amplify-App bereitzustellen. Wenn Sie bestimmte Abhängigkeiten haben, deren Installation während eines Builds mithilfe des Standardcontainers von Amplify lange dauert, können Sie Ihr eigenes Docker-Image erstellen und während eines Builds darauf verweisen. Bilder können auf Amazon Elastic Container Registry Public gehostet werden.

Note

Die Build-Einstellungen sind nur dann im Hosting-Menü der Amplify-Konsole sichtbar, wenn eine App für die kontinuierliche Bereitstellung eingerichtet und mit einem Git-Repository verbunden ist. Anweisungen zu dieser Art der Bereitstellung finden Sie unter [Erste Schritte mit vorhandenem Code](#).

Anforderungen an ein benutzerdefiniertes Build-Image

Damit ein benutzerdefiniertes Build-Image als Amplify-Build-Image funktioniert, muss es die folgenden Anforderungen erfüllen:

1. Eine Linux-Distribution, die die GNU C-Bibliothek (Glibc) unterstützt, z. B. Amazon Linux, die für die x86-64-Architektur kompiliert wurde.
2. cURL: Wenn wir Ihr benutzerdefiniertes Image starten, laden wir unseren Build Runner in Ihrem Container herunter, weshalb cURL vorhanden sein muss. Wenn diese Abhängigkeit fehlt, schlägt der Build sofort ohne Ausgabe fehl, da unser Build-Runner keine Ausgabe erzeugen kann.
3. Git: Zum Klonen des Git-Repositorys muss Git im Image installiert sein. Wenn diese Abhängigkeit fehlt, schlägt der Schritt zum Klonen des Repositorys fehl.

4. OpenSSH: Um Ihr Repository sicher zu klonen, benötigen wir OpenSSH, um den SSH-Schlüssel während des Builds vorübergehend einzurichten. Das OpenSSH-Paket stellt die Befehle bereit, die der Build-Runner dafür benötigt.
5. Bash und The Bourne Shell: Diese beiden Hilfsprogramme werden verwendet, um Befehle während der Build-Zeit auszuführen. Wenn sie nicht installiert sind, schlagen Ihre Builds möglicherweise vor dem Start fehl.
6. Node.js+npm: Unser Build-Runner installiert Node nicht. Stattdessen ist es darauf angewiesen, dass Node und NPM im Image installiert sind. Dies ist nur für Builds erforderlich, die NPM-Pakete oder Node-spezifische Befehle erfordern. Wir empfehlen jedoch dringend, sie zu installieren, da der Amplify Build Runner diese Tools verwenden kann, um die Build-Ausführung zu verbessern, wenn sie vorhanden sind. Die Paket-Override-Funktion von Amplify verwendet NPM, um das Hugo-Extended-Paket zu installieren, wenn Sie eine Überschreibung für Hugo festlegen.

Die folgenden Pakete sind nicht erforderlich, wir empfehlen Ihnen jedoch dringend, sie zu installieren.

1. NVM (Node Version Manager): Wir empfehlen Ihnen, diesen Versionsmanager zu installieren, wenn Sie mit verschiedenen Versionen von umgehen müssenNode. Wenn Sie eine Überschreibung festlegen, werden mit der Paket-Override-Funktion von Amplify NVM die Versionen von Node.js vor jedem Build geändert.
2. Wget: Amplify kann das Wget Hilfsprogramm verwenden, um Dateien während des Build-Prozesses herunterzuladen. Wir empfehlen, dass Sie es in Ihrem benutzerdefinierten Image installieren.
3. Tar: Amplify kann das Tar Hilfsprogramm verwenden, um heruntergeladene Dateien während des Build-Prozesses zu dekomprimieren. Wir empfehlen, dass Sie es in Ihrem benutzerdefinierten Image installieren.

Konfiguration eines benutzerdefinierten Build-Images

So konfigurieren Sie ein in Amazon ECR gehostetes benutzerdefiniertes Build-Image

1. Informationen zum Einrichten eines öffentlichen Amazon ECR-Repositorys mit einem Docker-Image finden Sie unter [Erste Schritte](#) im Amazon ECR Public User Guide.
2. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
3. Wählen Sie die App aus, für die Sie ein benutzerdefiniertes Build-Image konfigurieren möchten.
4. Wählen Sie im Navigationsbereich Hosting, Build-Einstellungen aus.

5. Wählen Sie auf der Seite mit den Build-Einstellungen im Abschnitt Build-Image-Einstellungen die Option Bearbeiten aus.
6. Erweitern Sie auf der Seite mit den Build-Image-Einstellungen bearbeiten das Menü Build-Image und wählen Sie Benutzerdefiniertes Build-Image aus.
7. Geben Sie den Namen des Amazon ECR Public Repos ein, das Sie in Schritt eins erstellt haben. Hier wird Ihr Build-Image gehostet. Wenn der Name Ihres Repos beispielsweise ecr-exemplerepo lautet, würden Sie Folgendes eingeben. **public.ecr.aws/xxxxxxx/ecr-exemplerepo**
8. Wählen Sie Speichern.

Live-Paket-Updates

Live-Paket-Updates ermöglichen es Ihnen, Versionen von Paketen und Abhängigkeiten anzugeben, die im Amplify-Standard-Build-Image verwendet werden sollen. Das Standard-Build-Image enthält mehrere vorinstallierte Pakete und Abhängigkeiten (z. B. Hugo, Amplify CLI, Yarn usw.). Bei Live-Paket-Updates können Sie die Version dieser Abhängigkeiten überschreiben und entweder eine bestimmte Version angeben oder sicherstellen, dass immer die neueste Version installiert ist.

Wenn Live-Paket-Updates aktiviert sind, aktualisiert der Build Runner vor der Ausführung Ihres Builds zunächst die angegebenen Abhängigkeiten (oder führt ein Downgrade durch). Dadurch erhöht sich die Erstellungszeit proportional zu der Zeit, die für die Aktualisierung der Abhängigkeiten benötigt wird. Der Vorteil besteht jedoch darin, dass Sie sicherstellen können, dass dieselbe Version einer Abhängigkeit zum Erstellen Ihrer App verwendet wird.

Warning

Wenn Sie die Version Node.js auf die neueste Version setzen, schlagen Builds fehl. Stattdessen müssen Sie eine exakte Version von Node.js angeben, z. B. 1821.5, oder v0.1.2.

Konfiguration von Live-Paket-Updates

Um Live-Paket-Updates zu konfigurieren

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die Sie Live-Paket-Updates konfigurieren möchten.

3. Wählen Sie im Navigationsbereich Hosting, Build-Einstellungen aus.
4. Wählen Sie auf der Seite mit den Build-Einstellungen im Abschnitt Build-Image-Einstellungen die Option Bearbeiten aus.
5. Wählen Sie auf der Seite Build-Image-Einstellungen bearbeiten in der Liste Live-Paket-Updates die Option Neu hinzufügen aus.
6. Wählen Sie unter Package die Abhängigkeit aus, die Sie überschreiben möchten.
7. Behalten Sie für Version entweder die neueste Standardversion bei oder geben Sie eine bestimmte Version der Abhängigkeit ein. Wenn Sie Latest verwenden, wird die Abhängigkeit immer auf die neueste verfügbare Version aktualisiert.
8. Klicken Sie auf Speichern.

Eine Servicerolle hinzufügen

Amplify benötigt Berechtigungen, um Backend-Ressourcen mit Ihrem Frontend bereitzustellen. Dazu verwenden Sie eine Servicerolle. Eine Servicerolle ist die AWS Identity and Access Management (IAM) -Rolle, die Amplify übernimmt, wenn es in Ihrem Namen andere Dienste anruft. In diesem Handbuch erfahren Sie, wie Sie eine Amplify-Servicerolle erstellen, die über Kontoverwaltungsrechte verfügt und explizit direkten Zugriff auf Ressourcen ermöglicht, die Amplify-Anwendungen für die Bereitstellung, Erstellung und Verwaltung von Backends benötigen.

Erstellen einer Servicerolle

So erstellen Sie eine Servicerolle

1. [Öffnen Sie die IAM-Konsole](#) und wählen Sie in der linken Navigationsleiste Rollen und dann Rolle erstellen aus.
2. Wählen Sie auf der Seite Vertrauenswürdige Entitäten auswählen die Option AWS -Service aus. Wählen Sie unter Anwendungsfall die Option Amplify und dann Weiter aus.
3. Wählen Sie auf der Seite Add permissions (Berechtigungen hinzufügen) die Option Next (Weiter) aus.
4. Geben Sie auf der Seite Name, Ansicht und Erstellung für Rollennamen einen aussagekräftigen Namen ein, z. B. **AmplifyConsoleServiceRole-AmplifyRole**
5. Akzeptieren Sie alle Standardeinstellungen und wählen Sie „Rolle erstellen“.
6. Kehren Sie zur Amplify-Konsole zurück, um die Rolle an Ihre App anzuhängen.
 - Wenn Sie gerade dabei sind, eine neue App bereitzustellen
 - a. Aktualisieren Sie die Liste der Servicerollen.
 - b. Wählen Sie die Rolle aus, die Sie gerade erstellt haben. In diesem Beispiel sollte es so aussehen AmplifyConsoleServiceRole- AmplifyRole
 - c. Wählen Sie Weiter und folgen Sie den Schritten, um Ihre App-Bereitstellung abzuschließen.
 - Wenn Sie eine bestehende App haben
 - a. Wählen Sie im Navigationsbereich App-Einstellungen und dann Allgemeine Einstellungen aus.
 - b. Wählen Sie auf der Seite Allgemeine Einstellungen die Option Bearbeiten aus.

- c. Wählen Sie auf der Seite Allgemeine Einstellungen bearbeiten die Rolle, die Sie gerade erstellt haben, aus der Liste der Servicerollen aus.
 - d. Wählen Sie Speichern.
7. Die Amplify-Konsole verfügt jetzt über Berechtigungen zum Bereitstellen von Backend-Ressourcen für Ihre App.

Confused-Deputy-Prävention

Das Problem des verwirrten Stellvertreters ist ein Sicherheitsproblem, bei dem eine Entität, die keine Berechtigung zur Durchführung einer Aktion hat, eine privilegiertere Entität zur Durchführung der Aktion zwingen kann. Weitere Informationen finden Sie unter [Serviceübergreifende Confused-Deputy-Prävention](#).

Derzeit erzwingt die Standard-Vertrauensrichtlinie für die Amplify-Backend Deployment Servicerolle die Schlüssel `aws:SourceArn` und die `aws:SourceAccount` globalen Kontextbedingungen, um zu verhindern, dass der Stellvertreter verwirrt wird. Wenn Sie jedoch zuvor eine Amplify-Backend Deployment Rolle in Ihrem Konto erstellt haben, können Sie die Vertrauensrichtlinie der Rolle aktualisieren, um diese Bedingungen hinzuzufügen, um vor verwirrtem Stellvertreter zu schützen.

Verwenden Sie das folgende Beispiel, um den Zugriff auf Apps in Ihrem Konto einzuschränken. Ersetzen Sie die Region und die Anwendungs-ID im Beispiel durch Ihre eigenen Informationen.

```
"Condition": {
  "ArnLike": {
    "aws:SourceArn": "arn:aws:amplify:us-east-1:123456789012:apps/*"
  },
  "StringEquals": {
    "aws:SourceAccount": "123456789012"
  }
}
```

Anweisungen zum Bearbeiten der Vertrauensrichtlinie für eine Rolle mithilfe von finden Sie unter [Ändern einer Rolle \(Konsole\)](#) im IAM-Benutzerhandbuch. AWS Management Console

Verwaltung der App-Leistung

Die Standard-Hosting-Architektur von Amplify optimiert das Gleichgewicht zwischen Hosting-Leistung und Bereitstellungsverfügbarkeit. Für die meisten Kunden empfehlen wir, die Standardarchitektur zu verwenden.

Wenn Sie eine genauere Kontrolle über die Leistung einer App benötigen, können Sie den `Cache-Control` HTTP-Header manuell so einrichten, dass er die Hosting-Leistung optimiert, indem Inhalte für einen längeren Zeitraum am Rand des Content Delivery Network (CDN) zwischengespeichert werden.

Verwenden von Headern zur Steuerung der Cache-Dauer

`Cache-Control` HTTP-Header `max-age` und `s-maxage` -Direktiven wirken sich auf die Dauer des Inhalts-Cachings für Ihre App aus. Die `max-age` Direktive teilt dem Browser mit, wie lange (in Sekunden) Inhalte im Cache verbleiben sollen, bevor sie vom Ursprungsserver aktualisiert werden. Die `s-maxage` Direktive überschreibt `max-age` und ermöglicht es Ihnen, anzugeben, wie lange (in Sekunden) Inhalte am CDN-Edge verbleiben sollen, bevor sie vom Ursprungsserver aktualisiert werden.

Apps, die mit Amplify gehostet werden, berücksichtigen die vom Ursprung gesendeten `Cache-Control` Header, es sei denn, Sie überschreiben sie mit benutzerdefinierten Headern, die Sie definieren. Amplify wendet nur `Cache-Control` benutzerdefinierte Header für erfolgreiche Antworten mit einem `200 OK` Statuscode an. Dadurch wird verhindert, dass Fehlerantworten zwischengespeichert und anderen Benutzern, die dieselbe Anfrage stellen, zugestellt werden.

Sie können die `s-maxage` Direktive manuell anpassen, um mehr Kontrolle über die Leistung und Bereitstellungsverfügbarkeit Ihrer App zu haben. Um beispielsweise die Dauer zu verlängern, für die Ihre Inhalte am Edge zwischengespeichert bleiben, können Sie die Gültigkeitsdauer (Time to Live, TTL) manuell erhöhen, indem Sie `s-maxage` auf einen Wert aktualisieren, der länger als die Standardeinstellung 600 Sekunden (10 Minuten) ist.

Sie können benutzerdefinierte Header für eine App im Abschnitt Benutzerdefinierte Header der Amplify-Konsole definieren. Ein Beispiel für das YAML Format finden Sie unter [Benutzerdefinierte Cache-Control-Header](#)

Einstellung des Cache-Control Headers zur Steigerung der Anwendungsleistung

Gehen Sie wie folgt vor, um die `s-maxage` Direktive so einzustellen, dass Inhalte 24 Stunden lang am CDN-Edge zwischengespeichert werden.

So legen Sie einen benutzerdefinierten Header fest Cache-Control

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die [Amplify-Konsole](#).
2. Wählen Sie die App aus, für die Sie benutzerdefinierte Header festlegen möchten.
3. Wählen Sie im Navigationsbereich Hosting, Benutzerdefinierte Header aus.
4. Wählen Sie auf der Seite Benutzerdefinierte Header die Option Bearbeiten aus.
5. Geben Sie im Fenster Benutzerdefinierte Kopfzeilen bearbeiten die Informationen für Ihre benutzerdefinierte Kopfzeile wie folgt ein:
 - a. Geben Sie für `pattern`, `**/*` für alle Pfade ein.
 - b. Geben Sie unter `key` den Wert **Cache-Control** ein.
 - c. Geben Sie unter `value` den Wert **s-maxage=86400** ein.
6. Wählen Sie Speichern.
7. Stellen Sie die App erneut bereit, um den neuen benutzerdefinierten Header anzuwenden.

Protokollieren von Amplify-API-Aufrufen mit AWS CloudTrail

AWS Amplify ist in einen Dienst integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen eines Benutzers, einer Rolle oder eines AWS Dienstes in Amplify bereitstellt. CloudTrail erfasst alle API-Aufrufe für Amplify als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der Amplify-Konsole und Code-Aufrufe der Amplify-API-Operationen. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen an einen Amazon S3 S3-Bucket aktivieren, einschließlich Ereignissen für Amplify. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse trotzdem in der CloudTrail Konsole im Ereignisverlauf anzeigen. Anhand der CloudTrail gesammelten Informationen können Sie die Anfrage, die an Amplify gestellt wurde, die IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde, und weitere Details ermitteln.

Weitere Informationen CloudTrail dazu finden Sie im [AWS CloudTrail Benutzerhandbuch](#).

Amplify Sie Informationen in CloudTrail

CloudTrail ist in Ihrem AWS Konto standardmäßig aktiviert. Wenn in Amplify Aktivitäten auftreten, wird diese Aktivität zusammen mit anderen AWS Serviceereignissen in der CloudTrail Ereignishistorie in einem Ereignis aufgezeichnet. Sie können die neusten Ereignisse in Ihr AWS -Konto herunterladen und dort suchen und anzeigen. Weitere Informationen finden Sie im AWS CloudTrail Benutzerhandbuch unter [Ereignisse mit CloudTrail Ereignisverlauf anzeigen](#).

Für eine fortlaufende Aufzeichnung von Ereignissen in Ihrem AWS Konto, einschließlich Ereignissen für Amplify, erstellen Sie einen Trail. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket. Wenn Sie einen Trail in der Konsole anlegen, gilt dieser für alle AWS -Regionen. Der Trail protokolliert Ereignisse aus allen Regionen der AWS Partition und übermittelt die Protokolldateien an den von Ihnen angegebenen Amazon S3 S3-Bucket. Darüber hinaus können Sie andere AWS Dienste konfigurieren, um die in den CloudTrail Protokollen gesammelten Ereignisdaten weiter zu analysieren und darauf zu reagieren. Weitere Informationen finden Sie in folgenden Themen im AWS CloudTrail -Benutzerhandbuch:

- [Einen Trail für dein AWS Konto erstellen](#)
- [CloudTrail unterstützte Dienste und Integrationen](#)
- [Konfiguration von Amazon SNS SNS-Benachrichtigungen für CloudTrail](#)
- [Empfangen von CloudTrail Protokolldateien aus mehreren Regionen](#) und [Empfangen von CloudTrail Protokolldateien von mehreren Konten](#)

Alle Amplify-Operationen werden von der [AWS Amplify Console API Reference](#), der [AWS Amplify Admin UI Reference CloudTrail](#) und der [Amplify UI Builder API Reference](#) protokolliert und dokumentiert. Beispielsweise generieren Aufrufe von `DeleteApp` und `DeleteBackendEnvironment` Operationen Einträge in den CloudTrail Protokolldateien. `CreateApp`

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Wurde die Anfrage mit Root- oder AWS Identity and Access Management (IAM-) Benutzeranmeldedaten gestellt?
- Wurde die Anfrage mit temporären Sicherheitsanmeldedaten für eine Rolle oder einen Verbundbenutzer gestellt?
- Wurde die Anfrage von einem anderen AWS Dienst gestellt.

Weitere Informationen finden Sie unter dem [Element CloudTrail userIdentity](#) im AWS CloudTrail Benutzerhandbuch.

Verstehen von Amplify-Protokolldateieinträgen

Ein Trail ist eine Konfiguration, die die Übertragung von Ereignissen als Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket ermöglicht. CloudTrail Protokolldateien enthalten einen oder mehrere Protokolleinträge. Ein Ereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar und enthält Informationen über die angeforderte Aktion, Datum und Uhrzeit der Aktion, Anforderungsparameter usw. CloudTrail Protokolldateien sind kein geordneter Stack-Trace der öffentlichen API-Aufrufe, sodass sie nicht in einer bestimmten Reihenfolge angezeigt werden.

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der den [ListApps](#) Vorgang „AWS Amplify Console API Reference“ demonstriert.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/Mary_Major",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
```

```

    "userName": "Mary_Major",
    "sessionContext": {
      "sessionIssuer": {},
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-01-12T05:48:10Z"
      }
    }
  },
  "eventTime": "2021-01-12T06:47:29Z",
  "eventSource": "amplify.amazonaws.com",
  "eventName": "ListApps",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.255",
  "userAgent": "aws-internal/3 aws-sdk-java/1.11.898
Linux/4.9.230-0.1.ac.223.84.332.metal1.x86_64 OpenJDK_64-Bit_Server_VM/25.275-b01
java/1.8.0_275 vendor/Oracle_Corporation",
  "requestParameters": {
    "maxResults": "100"
  },
  "responseElements": null,
  "requestID": "1c026d0b-3397-405a-95aa-aa43aexample",
  "eventID": "c5fca3fb-d148-4fa1-ba22-5fa63example",
  "readOnly": true,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "444455556666"
}

```

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der den [ListBackendJobs](#) Vorgang AWS Amplify Admin UI API Reference demonstriert.

```

{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/Mary_Major",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "Mary_Major",

```

```
    "sessionContext": {
      "sessionIssuer": {},
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-01-13T00:47:25Z"
      }
    }
  },
  "eventTime": "2021-01-13T01:15:43Z",
  "eventSource": "amplifybackend.amazonaws.com",
  "eventName": "ListBackendJobs",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.255",
  "userAgent": "aws-internal/3 aws-sdk-java/1.11.898
Linux/4.9.230-0.1.ac.223.84.332.metal1.x86_64 OpenJDK_64-Bit_Server_VM/25.275-b01
java/1.8.0_275 vendor/Oracle_Corporation",
  "requestParameters": {
    "appId": "d23mv2oexample",
    "backendEnvironmentName": "staging"
  },
  "responseElements": {
    "jobs": [
      {
        "appId": "d23mv2oexample",
        "backendEnvironmentName": "staging",
        "jobId": "ed63e9b2-dd1b-4bf2-895b-3d5dcexample",
        "operation": "CreateBackendAuth",
        "status": "COMPLETED",
        "createTime": "1610499932490",
        "updateTime": "1610500140053"
      },
      {
        "appId": "d23mv2oexample",
        "backendEnvironmentName": "staging",
        "jobId": "06904b10-a795-49c1-92b7-185dfexample",
        "operation": "CreateBackend",
        "status": "COMPLETED",
        "createTime": "1610499657938",
        "updateTime": "1610499704458"
      }
    ],
    "appId": "d23mv2oexample",
    "backendEnvironmentName": "staging"
  }
}
```

```
  },  
  "requestID": "7adfabd6-98d5-4b11-bd39-c7deaexample",  
  "eventID": "68769310-c96c-4789-a6bb-68b52example",  
  "readOnly": false,  
  "eventType": "AwsApiCall",  
  "managementEvent": true,  
  "eventCategory": "Management",  
  "recipientAccountId": "444455556666"  
}
```

Sicherheit in Amplify

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von Rechenzentren und Netzwerkarchitekturen, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame AWS Verantwortung von Ihnen und Ihnen. Das [Modell der geteilten Verantwortung](#) beschreibt dies als Sicherheit der Cloud selbst und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS Dienste in der AWS Cloud ausführt. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Externe Prüfer testen und verifizieren regelmäßig die Wirksamkeit unserer Sicherheitsmaßnahmen im Rahmen der [AWS](#) . Weitere Informationen zu den Compliance-Programmen, die für gelten AWS Amplify, finden Sie unter [AWS Services im Umfang nach Compliance-Programmen AWS](#) .
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Dienst, den Sie nutzen. Sie sind auch für andere Faktoren verantwortlich, etwa für die Vertraulichkeit Ihrer Daten, für die Anforderungen Ihres Unternehmens und für die geltenden Gesetze und Vorschriften.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der gemeinsamen Verantwortung bei der Verwendung von Amplify anwenden können. Die folgenden Themen zeigen Ihnen, wie Sie Amplify konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie lernen auch, wie Sie andere AWS Dienste nutzen können, mit denen Sie Ihre Amplify-Ressourcen überwachen und sichern können.

Themen

- [Identity and Access Management für Amplify](#)
- [Datenschutz in Amplify](#)
- [Konformitätsprüfung für AWS Amplify](#)
- [Sicherheit der Infrastruktur in AWS Amplify](#)
- [Protokollierung und Überwachung von Sicherheitsereignissen in Amplify](#)
- [Serviceübergreifende Confused-Deputy-Prävention](#)
- [Bewährte Sicherheitsmethoden für Amplify](#)

Identity and Access Management für Amplify

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf Ressourcen sicher zu kontrollieren. AWS IAM-Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), Amplify-Ressourcen zu verwenden. IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Themen

- [Zielgruppe](#)
- [Authentifizierung mit Identitäten](#)
- [Verwalten des Zugriffs mit Richtlinien](#)
- [So funktioniert Amplify mit IAM](#)
- [Beispiele für identitätsbasierte Richtlinien für Amplify](#)
- [AWS verwaltete Richtlinien für AWS Amplify](#)
- [Fehlerbehebung bei Amplify Identity and Access](#)

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von der Arbeit ab, die Sie in Amplify ausführen.

Dienstbenutzer — Wenn Sie den Amplify-Dienst für Ihre Arbeit verwenden, stellt Ihnen Ihr Administrator die Anmeldeinformationen und Berechtigungen zur Verfügung, die Sie benötigen. Da Sie für Ihre Arbeit mehr Amplify-Funktionen verwenden, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Funktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anfordern müssen. Wenn Sie in Amplify nicht auf eine Funktion zugreifen können, finden Sie weitere Informationen unter [Fehlerbehebung bei Amplify Identity and Access](#).

Serviceadministrator — Wenn Sie in Ihrem Unternehmen für die Amplify-Ressourcen verantwortlich sind, haben Sie wahrscheinlich vollen Zugriff auf Amplify. Es ist Ihre Aufgabe, zu bestimmen, auf welche Amplify-Funktionen und -Ressourcen Ihre Servicebenutzer zugreifen sollen. Sie müssen dann Anträge an Ihren IAM-Administrator stellen, um die Berechtigungen Ihrer Servicenutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um die Grundkonzepte von IAM nachzuvollziehen. Weitere Informationen darüber, wie Ihr Unternehmen IAM mit Amplify nutzen kann, finden Sie unter [So funktioniert Amplify mit IAM](#)

IAM-Administrator — Wenn Sie ein IAM-Administrator sind, möchten Sie vielleicht mehr darüber erfahren, wie Sie Richtlinien schreiben können, um den Zugriff auf Amplify zu verwalten. Beispiele für identitätsbasierte Amplify-Richtlinien, die Sie in IAM verwenden können, finden Sie unter [Beispiele für identitätsbasierte Richtlinien für Amplify](#)

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen als IAM-Benutzer authentifiziert (angemeldet AWS) sein oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich AWS als föderierte Identität anmelden, indem Sie Anmeldeinformationen verwenden, die über eine Identitätsquelle bereitgestellt wurden. AWS IAM Identity Center (IAM Identity Center) -Benutzer, die Single Sign-On-Authentifizierung Ihres Unternehmens und Ihre Google- oder Facebook-Anmeldeinformationen sind Beispiele für föderierte Identitäten. Wenn Sie sich als Verbundidentität anmelden, hat der Administrator vorher mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet. Wenn Sie über den Verbund darauf zugreifen AWS, übernehmen Sie indirekt eine Rolle.

Je nachdem, welcher Benutzertyp Sie sind, können Sie sich beim AWS Management Console oder beim AWS Zugangportal anmelden. Weitere Informationen zur Anmeldung finden Sie AWS unter [So melden Sie sich bei Ihrem an AWS-Konto](#) im AWS-Anmeldung Benutzerhandbuch.

Wenn Sie AWS programmgesteuert darauf zugreifen, AWS stellt es ein Software Development Kit (SDK) und eine Befehlszeilenschnittstelle (CLI) bereit, um Ihre Anfragen mithilfe Ihrer Anmeldeinformationen kryptografisch zu signieren. Wenn Sie keine AWS Tools verwenden, müssen Sie Anfragen selbst signieren. Weitere Informationen zur Verwendung der empfohlenen Methode, um Anfragen selbst zu [signieren, finden Sie im IAM-Benutzerhandbuch unter AWS API-Anfragen signieren](#).

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen angeben. AWS empfiehlt beispielsweise, die Multi-Faktor-Authentifizierung (MFA) zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. Weitere Informationen finden Sie unter [Multi-Faktor-Authentifizierung](#) im AWS IAM Identity Center - Benutzerhandbuch und [Verwenden der Multi-Faktor-Authentifizierung \(MFA\) in AWS](#) im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie ein AWS-Konto erstellen, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services Ressourcen im Konto hat. Diese Identität wird als AWS-Konto Root-Benutzer bezeichnet. Sie können darauf zugreifen, indem Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, mit denen Sie das Konto erstellt haben. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen und verwenden Sie diese, um die Aufgaben auszuführen, die nur der Root-Benutzer ausführen kann. Eine vollständige Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter [Aufgaben, die Root-Benutzer-Anmeldeinformationen erfordern](#) im IAM-Benutzerhandbuch.

Verbundidentität

Als bewährte Methode sollten menschliche Benutzer, einschließlich Benutzer, die Administratorzugriff benötigen, für den Zugriff AWS-Services mithilfe temporärer Anmeldeinformationen den Verbund mit einem Identitätsanbieter verwenden.

Eine föderierte Identität ist ein Benutzer aus Ihrem Unternehmensbenutzerverzeichnis, einem Web-Identitätsanbieter AWS Directory Service, dem Identity Center-Verzeichnis oder einem beliebigen Benutzer, der mithilfe AWS-Services von Anmeldeinformationen zugreift, die über eine Identitätsquelle bereitgestellt wurden. Wenn föderierte Identitäten darauf zugreifen AWS-Konten, übernehmen sie Rollen, und die Rollen stellen temporäre Anmeldeinformationen bereit.

Für die zentrale Zugriffsverwaltung empfehlen wir Ihnen, AWS IAM Identity Center zu verwenden. Sie können Benutzer und Gruppen in IAM Identity Center erstellen, oder Sie können eine Verbindung zu einer Gruppe von Benutzern und Gruppen in Ihrer eigenen Identitätsquelle herstellen und diese synchronisieren, um sie in all Ihren AWS-Konten Anwendungen zu verwenden. Informationen zu IAM Identity Center finden Sie unter [Was ist IAM Identity Center?](#) im AWS IAM Identity Center - Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein [IAM-Benutzer](#) ist eine Identität innerhalb Ihres Unternehmens AWS-Konto, die über spezifische Berechtigungen für eine einzelne Person oder Anwendung verfügt. Wenn möglich, empfehlen wir, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben. Bei speziellen Anwendungsfällen, die langfristige Anmeldeinformationen mit IAM-Benutzern erfordern, empfehlen wir jedoch, die Zugriffsschlüssel zu rotieren. Weitere Informationen finden Sie unter [Regelmäßiges](#)

[Rotieren von Zugriffsschlüsseln für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern](#) im IAM-Benutzerhandbuch.

Eine [IAM-Gruppe](#) ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht als Gruppe anmelden. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzer gleichzeitig angeben. Gruppen vereinfachen die Verwaltung von Berechtigungen, wenn es zahlreiche Benutzer gibt. Sie könnten beispielsweise einer Gruppe mit dem Namen IAMAdmins Berechtigungen zum Verwalten von IAM-Ressourcen erteilen.

Benutzer unterscheiden sich von Rollen. Ein Benutzer ist einer einzigen Person oder Anwendung eindeutig zugeordnet. Eine Rolle kann von allen Personen angenommen werden, die sie benötigen. Benutzer besitzen dauerhafte Anmeldeinformationen. Rollen stellen temporäre Anmeldeinformationen bereit. Weitere Informationen finden Sie unter [Erstellen eines IAM-Benutzers \(anstatt einer Rolle\)](#) im IAM-Benutzerhandbuch.

IAM-Rollen

Eine [IAM-Rolle](#) ist eine Identität innerhalb Ihres Unternehmens AWS-Konto, die über bestimmte Berechtigungen verfügt. Sie ist einem IAM-Benutzer vergleichbar, ist aber nicht mit einer bestimmten Person verknüpft. Sie können vorübergehend eine IAM-Rolle in der übernehmen, AWS Management Console indem Sie die Rollen [wechseln](#). Sie können eine Rolle übernehmen, indem Sie eine AWS CLI oder AWS API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter [Verwenden von IAM-Rollen](#) im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- **Verbundbenutzerzugriff** – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert, so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter [Erstellen von Rollen für externe Identitätsanbieter](#) im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter [Berechtigungssätze](#) im AWS IAM Identity Center -Benutzerhandbuch.
- **Temporäre IAM-Benutzerberechtigungen** – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.

- **Kontoübergreifender Zugriff** – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. Bei einigen können Sie AWS-Services jedoch eine Richtlinie direkt an eine Ressource anhängen (anstatt eine Rolle als Proxy zu verwenden). Informationen zum Unterschied zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter [Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch](#).
- **Serviceübergreifender Zugriff** — Einige verwenden Funktionen in anderen. AWS-Services AWS-Services Wenn Sie beispielsweise einen Aufruf in einem Service tätigen, führt dieser Service häufig Anwendungen in Amazon-EC2 aus oder speichert Objekte in Amazon-S3. Ein Dienst kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.
- **Forward Access Sessions (FAS)** — Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anfrage, Anfragen an AWS-Service nachgelagerte Dienste zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter [Zugriffssitzungen weiterleiten](#).
- **Servicerolle** – Eine Servicerolle ist eine [IAM-Rolle](#), die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter [Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service](#) im IAM-Benutzerhandbuch.
- **Dienstbezogene Rolle** — Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Servicebezogene Rollen erscheinen in Ihrem Dienst AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.
- **Auf Amazon EC2 ausgeführte Anwendungen** — Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2-Instance ausgeführt werden und API-Anfragen stellen AWS CLI . AWS Das ist eher zu empfehlen, als Zugriffsschlüssel innerhalb der EC2-Instance zu speichern. Um einer EC2-Instance eine AWS Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie

ein Instance-Profil, das an die Instance angehängt ist. Ein Instance-Profil enthält die Rolle und ermöglicht, dass Programme, die in der EC2-Instance ausgeführt werden, temporäre Anmeldeinformationen erhalten. Weitere Informationen finden Sie unter [Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon-EC2-Instances ausgeführt werden](#) im IAM-Benutzerhandbuch.

Informationen dazu, wann Sie IAM-Rollen oder IAM-Benutzer verwenden sollten, finden Sie unter [Erstellen einer IAM-Rolle \(anstatt eines Benutzers\)](#) im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie ist ein Objekt, AWS das, wenn es einer Identität oder Ressource zugeordnet ist, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Prinzipal (Benutzer, Root-Benutzer oder Rollensitzung) eine Anfrage stellt. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu Struktur und Inhalten von JSON-Richtliniendokumenten finden Sie unter [Übersicht über JSON-Richtlinien](#) im IAM-Benutzerhandbuch.

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das bedeutet, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

IAM-Richtlinien definieren Berechtigungen für eine Aktion unabhängig von der Methode, die Sie zur Ausführung der Aktion verwenden. Angenommen, es gibt eine Richtlinie, die Berechtigungen für die `iam:GetRole`-Aktion erteilt. Ein Benutzer mit dieser Richtlinie kann Rolleninformationen von der AWS Management Console AWS CLI, der oder der AWS API abrufen.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern,

welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter [Erstellen von IAM-Richtlinien](#) im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können weiter als Inline-Richtlinien oder verwaltete Richtlinien kategorisiert werden. Inline-Richtlinien sind direkt in einen einzelnen Benutzer, eine einzelne Gruppe oder eine einzelne Rolle eingebettet. Verwaltete Richtlinien sind eigenständige Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem System zuordnen können AWS-Konto. Zu den verwalteten Richtlinien gehören AWS verwaltete Richtlinien und vom Kunden verwaltete Richtlinien. Informationen dazu, wie Sie zwischen einer verwalteten Richtlinie und einer eingebundenen Richtlinie wählen, finden Sie unter [Auswahl zwischen verwalteten und eingebundenen Richtlinien](#) im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie [einen Prinzipal angeben](#). Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Zugriffssteuerungslisten (ACLs)

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3 und Amazon VPC sind Beispiele für Services, die ACLs unterstützen. AWS WAF Weitere Informationen“ zu ACLs finden Sie unter [Zugriffskontrollliste \(ACL\) – Übersicht](#) (Access Control List) im Amazon-Simple-Storage-Service-Entwicklerhandbuch.

Weitere Richtlinientypen

AWS unterstützt zusätzliche, weniger verbreitete Richtlinientypen. Diese Richtlinientypen können die maximalen Berechtigungen festlegen, die Ihnen von den häufiger verwendeten Richtlinientypen erteilt werden können.

- **Berechtigungsgrenzen** – Eine Berechtigungsgrenze ist ein erweitertes Feature, mit der Sie die maximalen Berechtigungen festlegen können, die eine identitätsbasierte Richtlinie einer IAM-Entität (IAM-Benutzer oder -Rolle) erteilen kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die daraus resultierenden Berechtigungen sind der Schnittpunkt der identitätsbasierten Richtlinien einer Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter [Berechtigungsgrenzen für IAM-Entitäten](#) im IAM-Benutzerhandbuch.
- **Service Control Policies (SCPs)** — SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in festlegen. AWS Organizations AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung mehrerer Objekte AWS-Konten , die Ihrem Unternehmen gehören. Wenn Sie innerhalb einer Organisation alle Features aktivieren, können Sie Service-Kontrollrichtlinien (SCPs) auf alle oder einzelne Ihrer Konten anwenden. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen Entitäten. Root-Benutzer des AWS-Kontos Weitere Informationen zu Organizations und SCPs finden Sie unter [Funktionsweise von SCPs](#) im AWS Organizations -Benutzerhandbuch.
- **Sitzungsrichtlinien** – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter [Sitzungsrichtlinien](#) im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn mehrere auf eine Anforderung mehrere Richtlinientypen angewendet werden können, sind die entsprechenden Berechtigungen komplizierter. Informationen darüber, wie AWS bestimmt wird,

ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie im IAM-Benutzerhandbuch unter [Bewertungslogik für Richtlinien](#).

So funktioniert Amplify mit IAM

Bevor Sie IAM verwenden, um den Zugriff auf Amplify zu verwalten, sollten Sie sich darüber informieren, welche IAM-Funktionen mit Amplify verwendet werden können.

IAM-Funktionen, die Sie mit Amplify verwenden können

IAM-Feature	Unterstützung Amplify
Identitätsbasierte Richtlinien	Ja
Ressourcenbasierte Richtlinien	Nein
Richtlinienaktionen	Ja
Richtlinienressourcen	Ja
Bedingungsschlüssel für die Richtlinie	Ja
ACLs	Nein
ABAC (Tags in Richtlinien)	Teilweise
Temporäre Anmeldeinformationen	Ja
Forward Access Sessions (FAS)	Ja
Servicerollen	Ja
Service-verknüpfte Rollen	Nein

Einen allgemeinen Überblick darüber, wie Amplify und andere AWS Dienste mit den meisten IAM-Funktionen funktionieren, finden Sie im [IAM-Benutzerhandbuch unter AWS Dienste, die mit IAM funktionieren](#).

Identitätsbasierte Richtlinien für Amplify

Unterstützt Richtlinien auf Identitätsbasis. Ja

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter [Erstellen von IAM-Richtlinien](#) im IAM-Benutzerhandbuch.

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Sie können den Prinzipal nicht in einer identitätsbasierten Richtlinie angeben, da er für den Benutzer oder die Rolle gilt, dem er zugeordnet ist. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der [IAM-Referenz für JSON-Richtlinienelemente](#) im IAM-Benutzerhandbuch.

Beispiele für identitätsbasierte Richtlinien für Amplify

Beispiele für identitätsbasierte Richtlinien von Amplify finden Sie unter [Beispiele für identitätsbasierte Richtlinien für Amplify](#)

Ressourcenbasierte Richtlinien innerhalb von Amplify

Unterstützt ressourcenbasierte Richtlinien Nein

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie [einen Prinzipal angeben](#). Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Um kontoübergreifenden Zugriff zu ermöglichen, können Sie ein gesamtes Konto oder IAM-Entitäten in einem anderen Konto als Prinzipal in einer ressourcenbasierten Richtlinie angeben. Durch das Hinzufügen eines kontoübergreifenden Auftraggebers zu einer ressourcenbasierten Richtlinie ist nur die halbe Vertrauensbeziehung eingerichtet. Wenn sich der Prinzipal und die Ressource unterscheiden AWS-Konten, muss ein IAM-Administrator des vertrauenswürdigen Kontos auch der Prinzipalentsität (Benutzer oder Rolle) die Berechtigung zum Zugriff auf die Ressource erteilen. Sie erteilen Berechtigungen, indem Sie der juristischen Stelle eine identitätsbasierte Richtlinie anfügen. Wenn jedoch eine ressourcenbasierte Richtlinie Zugriff auf einen Prinzipal in demselben Konto gewährt, ist keine zusätzliche identitätsbasierte Richtlinie erforderlich. Weitere Informationen finden Sie unter [Kontenübergreifender Ressourcenzugriff in IAM](#) im IAM-Benutzerhandbuch.

Politische Maßnahmen für Amplify

Unterstützt Richtlinienaktionen

Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Richtlinienaktionen haben normalerweise denselben Namen wie der zugehörige AWS API-Vorgang. Es gibt einige Ausnahmen, z. B. Aktionen, die nur mit Genehmigung durchgeführt werden können und für die es keinen passenden API-Vorgang gibt. Es gibt auch einige Operationen, die mehrere Aktionen in einer Richtlinie erfordern. Diese zusätzlichen Aktionen werden als abhängige Aktionen bezeichnet.

Schließen Sie Aktionen in eine Richtlinie ein, um Berechtigungen zur Durchführung der zugeordneten Operation zu erteilen.

Eine Liste der Amplify-Aktionen finden Sie unter [Actions defined by AWS Amplify](#) in der Service Authorization Reference.

Richtlinienaktionen in Amplify verwenden das folgende Präfix vor der Aktion:

```
amplify
```

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie mit Kommata:

```
"Action": [  
  "amplify:action1",  
  "amplify:action2"  
]
```

Beispiele für identitätsbasierte Richtlinien von Amplify finden Sie unter. [Beispiele für identitätsbasierte Richtlinien für Amplify](#)

Politische Ressourcen für Amplify

Unterstützt Richtlinienressourcen

Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das bedeutet die Festlegung, welcher Prinzipal Aktionen für welche Ressourcen unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Anweisungen müssen entweder ein `Resource` oder ein `NotResource`-Element enthalten. Als bewährte Methode geben Sie eine Ressource mit dem zugehörigen [Amazon-Ressourcennamen \(ARN\)](#) an. Sie können dies für Aktionen tun, die einen bestimmten Ressourcentyp unterstützen, der als Berechtigungen auf Ressourcenebene bezeichnet wird.

Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, z. B. Auflistungsoperationen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Eine Liste der Amplify-Ressourcentypen und ihrer ARNs finden Sie unter [Ressourcentypen definiert von AWS Amplify](#) in der Service Authorization Reference. Informationen zu den Aktionen, mit denen Sie den ARN einzelner Ressourcen angeben können, finden Sie unter [Von AWS Amplify definierte Aktionen](#).

Beispiele für identitätsbasierte Richtlinien von Amplify finden Sie unter. [Beispiele für identitätsbasierte Richtlinien für Amplify](#)

Schlüssel zur Richtlinienbedingung für Amplify

Unterstützt servicespezifische Richtlinienbedingungsschlüssel Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Condition` (oder `Condition block`) ermöglicht Ihnen die Angabe der Bedingungen, unter denen eine Anweisung wirksam ist. Das Element `Condition` ist optional. Sie können bedingte Ausdrücke erstellen, die [Bedingungsoperatoren](#) verwenden, z. B. `ist gleich` oder `kleiner als`, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt.

Wenn Sie mehrere `Condition`-Elemente in einer Anweisung oder mehrere Schlüssel in einem einzelnen `Condition`-Element angeben, wertet AWS diese mittels einer logischen AND-Operation aus. Wenn Sie mehrere Werte für einen einzelnen Bedingungsschlüssel angeben, wertet die Bedingung mithilfe einer logischen OR Operation aus. Alle Bedingungen müssen erfüllt werden, bevor die Berechtigungen der Anweisung gewährt werden.

Sie können auch Platzhaltervariablen verwenden, wenn Sie Bedingungen angeben. Beispielsweise können Sie einem IAM-Benutzer die Berechtigung für den Zugriff auf eine Ressource nur dann gewähren, wenn sie mit dessen IAM-Benutzernamen gekennzeichnet ist. Weitere Informationen finden Sie unter [IAM-Richtlinienelemente: Variablen und Tags](#) im IAM-Benutzerhandbuch.

AWS unterstützt globale Bedingungsschlüssel und dienstspezifische Bedingungsschlüssel. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter [Kontextschlüssel für AWS globale Bedingungen](#) im IAM-Benutzerhandbuch.

Eine Liste der Amplify-Bedingungsschlüssel finden Sie unter [Bedingungsschlüssel für AWS Amplify](#) in der Service Authorization Reference. Informationen zu den Aktionen und Ressourcen, mit denen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter [Aktionen definiert von AWS Amplify](#).

Beispiele für identitätsbasierte Richtlinien von Amplify finden Sie unter [Beispiele für identitätsbasierte Richtlinien für Amplify](#)

Zugriffskontrolllisten (ACLs) in Amplify

Unterstützt ACLs	Nein
------------------	------

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Attributbasierte Zugriffskontrolle (ABAC) mit Amplify

Unterstützt ABAC (Tags in Richtlinien)	Teilweise
--	-----------

Die attributbasierte Zugriffskontrolle (ABAC) ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen definiert werden. In werden diese AWS Attribute als Tags bezeichnet. Sie können Tags an IAM-Entitäten (Benutzer oder Rollen) und an viele AWS Ressourcen anhängen. Das Markieren von Entitäten und Ressourcen ist der erste Schritt von ABAC. Anschließend entwerfen Sie ABAC-Richtlinien, um Operationen zuzulassen, wenn das Tag des Prinzipals mit dem Tag der Ressource übereinstimmt, auf die sie zugreifen möchten.

ABAC ist in Umgebungen hilfreich, die schnell wachsen, und unterstützt Sie in Situationen, in denen die Richtlinienverwaltung mühsam wird.

Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungelement einer [Richtlinie Tag-Informationen](#) an, indem Sie die Schlüssel `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, oder Bedingung `aws:TagKeys` verwenden.

Wenn ein Service alle drei Bedingungsschlüssel für jeden Ressourcentyp unterstützt, lautet der Wert für den Service Ja. Wenn ein Service alle drei Bedingungsschlüssel für nur einige Ressourcentypen unterstützt, lautet der Wert Teilweise.

Weitere Informationen zu ABAC finden Sie unter [Was ist ABAC?](#) im IAM-Benutzerhandbuch. Um ein Tutorial mit Schritten zur Einstellung von ABAC anzuzeigen, siehe [Attributbasierte Zugriffskontrolle \(ABAC\)](#) verwenden im IAM-Benutzerhandbuch.

Temporäre Anmeldeinformationen mit Amplify verwenden

Unterstützt temporäre Anmeldeinformationen	Ja
--	----

Einige funktionieren AWS-Services nicht, wenn Sie sich mit temporären Anmeldeinformationen anmelden. Weitere Informationen, einschließlich Informationen, die mit temporären Anmeldeinformationen AWS-Services [funktionieren AWS-Services](#) , [finden Sie im IAM-Benutzerhandbuch unter Diese Option funktioniert mit IAM](#).

Sie verwenden temporäre Anmeldeinformationen, wenn Sie sich mit einer anderen AWS Management Console Methode als einem Benutzernamen und einem Passwort anmelden. Wenn Sie beispielsweise AWS über den Single Sign-On-Link (SSO) Ihres Unternehmens darauf zugreifen, werden bei diesem Vorgang automatisch temporäre Anmeldeinformationen erstellt. Sie erstellen auch automatisch temporäre Anmeldeinformationen, wenn Sie sich als Benutzer bei der Konsole anmelden und dann die Rollen wechseln. Weitere Informationen zum Wechseln von Rollen finden Sie unter [Wechseln zu einer Rolle \(Konsole\)](#) im IAM-Benutzerhandbuch.

Mithilfe der AWS API AWS CLI oder können Sie temporäre Anmeldeinformationen manuell erstellen. Sie können diese temporären Anmeldeinformationen dann für den Zugriff verwenden AWS. AWS empfiehlt, temporäre Anmeldeinformationen dynamisch zu generieren, anstatt langfristige Zugriffsschlüssel zu verwenden. Weitere Informationen finden Sie unter [Temporäre Sicherheitsanmeldeinformationen in IAM](#).

Weiterleitungszugriffssitzungen für Amplify

Unterstützt Forward Access Sessions (FAS)	Ja
---	----

Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, kombiniert mit der Anforderung, Anfragen an nachgelagerte Dienste AWS-Service zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter [Zugriffssitzungen weiterleiten](#).

Servicerollen für Amplify

Unterstützt Servicerollen	Ja
---------------------------	----

Eine Servicerolle ist eine [IAM-Rolle](#), die ein Service annimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter [Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service](#) im IAM-Benutzerhandbuch.

Warning

Das Ändern der Berechtigungen für eine Servicerolle kann die Amplify-Funktionalität beeinträchtigen. Bearbeiten Sie Servicerollen nur, wenn Amplify Sie dazu anleitet.

Serviceverknüpfte Rollen für Amplify

Unterstützt serviceverknüpfte Rollen

Nein

Eine serviceverknüpfte Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Dienstbezogene Rollen werden in Ihrem Dienst angezeigt AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

Einzelheiten zum Erstellen oder Verwalten von dienstbezogenen Rollen finden Sie im [AWS IAM-Benutzerhandbuch unter Dienste, die mit IAM funktionieren](#). Suchen Sie in der Tabelle nach einem Service mit einem Yes in der Spalte Service-linked role (Serviceverknüpfte Rolle). Wählen Sie den Link Ja, um die Dokumentation zu serviceverknüpften Rollen für diesen Service aufzurufen.

Beispiele für identitätsbasierte Richtlinien für Amplify

Standardmäßig haben Benutzer und Rollen keine Berechtigung, Amplify-Ressourcen zu erstellen oder zu ändern. Sie können auch keine Aufgaben mithilfe der AWS Management Console, AWS Command Line Interface (AWS CLI) oder AWS API ausführen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter [Erstellen von IAM-Richtlinien](#) im IAM-Benutzerhandbuch.

Einzelheiten zu den von Amplify definierten Aktionen und Ressourcentypen, einschließlich des Formats der ARNs für jeden Ressourcentyp, finden Sie unter [Aktionen, Ressourcen und Bedingungsschlüssel für AWS Amplify](#) in der Service Authorization Reference.

Themen

- [Bewährte Methoden für Richtlinien](#)
- [Verwenden der Amplify-Konsole](#)
- [Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer](#)

Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand Amplify-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder diese löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um damit zu beginnen, Ihren Benutzern und Workloads Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter [AWS -verwaltete Richtlinien](#) oder [AWS -verwaltete Richtlinien für Auftrags-Funktionen](#) im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter [Richtlinien und Berechtigungen in IAM](#) im IAM-Benutzerhandbuch.
- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn

diese für einen bestimmten Zweck verwendet werden AWS-Service, z. AWS CloudFormation
B. Weitere Informationen finden Sie unter [IAM-JSON-Richtlinienelemente: Bedingung](#) im IAM-Benutzerhandbuch.

- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter [Richtlinienvvalidierung zum IAM Access Analyzer](#) im IAM-Benutzerhandbuch.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter [Konfigurieren eines MFA-geschützten API-Zugriffs](#) im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter [Bewährte Methoden für die Sicherheit in IAM](#) im IAM-Benutzerhandbuch.

Verwenden der Amplify-Konsole

Um auf die AWS Amplify Konsole zugreifen zu können, benötigen Sie ein Mindestmaß an Berechtigungen. Diese Berechtigungen müssen es Ihnen ermöglichen, Details zu den Amplify-Ressourcen in Ihrem AWS-Konto aufzulisten und anzuzeigen. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (Benutzer oder Rollen) mit dieser Richtlinie.

Sie müssen Benutzern, die nur die AWS CLI oder die AWS API aufrufen, keine Mindestberechtigungen für die Konsole gewähren. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die der API-Operation entsprechen, die die Benutzer ausführen möchten.

Mit der Veröffentlichung von Amplify Studio sind für das Löschen einer App oder eines Backends beides `amplify` und `amplifybackend` Berechtigungen erforderlich. Wenn eine IAM-Richtlinie nur `amplify` Berechtigungen vorsieht, erhält ein Benutzer beim Versuch, eine App zu löschen, einen Berechtigungsfehler. Wenn Sie ein Administrator sind, der Richtlinien schreibt, sollten Sie die richtigen Berechtigungen für Benutzer festlegen, die Löschaktionen ausführen müssen.

Um sicherzustellen, dass Benutzer und Rollen die Amplify-Konsole weiterhin verwenden können, fügen Sie den Entitäten auch die Amplify ConsoleAccess - oder ReadOnly AWS verwaltete Richtlinie hinzu. Weitere Informationen finden Sie unter [Hinzufügen von Berechtigungen zu einem Benutzer](#) im IAM-Benutzerhandbuch.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie beinhaltet Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der OR-API. AWS CLI AWS

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsWithUser",
        "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": "*"
    }
  ]
}
```

```
]
}
```

AWS verwaltete Richtlinien für AWS Amplify

Eine AWS verwaltete Richtlinie ist eine eigenständige Richtlinie, die von erstellt und verwaltet AWS wird. AWS Verwaltete Richtlinien dienen dazu, Berechtigungen für viele gängige Anwendungsfälle bereitzustellen, sodass Sie damit beginnen können, Benutzern, Gruppen und Rollen Berechtigungen zuzuweisen.

Beachten Sie, dass AWS verwaltete Richtlinien für Ihre speziellen Anwendungsfälle möglicherweise keine Berechtigungen mit den geringsten Rechten gewähren, da sie allen AWS Kunden zur Verfügung stehen. Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie [kundenverwaltete Richtlinien](#) definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind.

Sie können die in AWS verwalteten Richtlinien definierten Berechtigungen nicht ändern. Wenn die in einer AWS verwalteten Richtlinie definierten Berechtigungen AWS aktualisiert werden, wirkt sich das Update auf alle Prinzidentitäten (Benutzer, Gruppen und Rollen) aus, denen die Richtlinie zugeordnet ist. AWS aktualisiert eine AWS verwaltete Richtlinie höchstwahrscheinlich, wenn eine neue Richtlinie eingeführt AWS-Service wird oder neue API-Operationen für bestehende Dienste verfügbar werden.

Weitere Informationen finden Sie unter [Von AWS verwaltete Richtlinien](#) im IAM-Benutzerhandbuch.

Von AWS verwaltete Richtlinie: AdministratorAccess -Amplify

Sie können die AdministratorAccess-Amplify-Richtlinie an Ihre IAM-Identitäten anfügen. Amplify verknüpft diese Richtlinie auch mit einer Servicerolle, die es Amplify ermöglicht, Aktionen in Ihrem Namen durchzuführen.

Wenn Sie ein Backend in der Amplify-Konsole bereitstellen, müssen Sie eine Amplify-Backend Deployment Servicerolle erstellen, die Amplify zum Erstellen und Verwalten von Ressourcen verwendet. AWS IAM fügt die AdministratorAccess-Amplify verwaltete Richtlinie der Servicerolle hinzu. Amplify-Backend Deployment

Diese Richtlinie gewährt Kontoverwaltungsberechtigungen und ermöglicht gleichzeitig ausdrücklich den direkten Zugriff auf Ressourcen, die Amplify-Anwendungen zum Erstellen und Verwalten von Backends benötigen.

Details zu Berechtigungen

Diese Richtlinie ermöglicht den Zugriff auf mehrere AWS Dienste, einschließlich IAM-Aktionen. Diese Aktionen ermöglichen es Identitäten, für die diese Richtlinie gilt, zu verwenden, AWS Identity and Access Management um andere Identitäten mit beliebigen Berechtigungen zu erstellen. Dies ermöglicht eine Eskalation von Berechtigungen, und diese Richtlinie sollte als ebenso wirksam angesehen werden wie die Richtlinie. `AdministratorAccess`

Diese Richtlinie erteilt der `iam:PassRole` Aktion die Genehmigung für alle Ressourcen. Dies ist erforderlich, um die Konfiguration von Amazon Cognito Cognito-Benutzerpools zu unterstützen.

Die Berechtigungen für diese Richtlinie finden Sie unter [AdministratorAccess-Amplify](#) in der Referenz für AWS verwaltete Richtlinien.

AWS verwaltete Richtlinie: AmplifyBackendDeployFullAccess

Sie können die `AmplifyBackendDeployFullAccess`-Richtlinie an Ihre IAM-Identitäten anfügen.

Diese Richtlinie gewährt Amplify volle Zugriffsberechtigungen für die Bereitstellung von Amplify-Backend-Ressourcen mithilfe von. AWS Cloud Development Kit (AWS CDK) Berechtigungen werden auf die AWS CDK Rollen übertragen, die über die erforderlichen Richtlinienberechtigungen verfügen. `AdministratorAccess`

Details zu Berechtigungen

Diese Richtlinie umfasst Berechtigungen für die folgenden Aktionen.

- `Amplify`— Rufen Sie Metadaten über bereitgestellte Anwendungen ab.
- `AWS CloudFormation`— Erstellen, aktualisieren und löschen Sie von Amplify verwaltete Stacks.
- `SSM`— Erstellen, aktualisieren und löschen Sie den von Amplify verwalteten SSM-Parameterspeicher `String` und `SecureString` die Parameter.
- `AWS AppSync`— AWS AppSync Schema-, Resolver- und Funktionsressourcen aktualisieren und abrufen. Der Zweck besteht darin, die Sandbox-Hotswapping-Funktionalität der zweiten Generation zu unterstützen.
- `Lambda`— Aktualisieren Sie die Konfiguration für von Amplify verwaltete Funktionen und rufen Sie sie ab. Der Zweck besteht darin, die Sandbox-Hotswapping-Funktionalität der zweiten Generation zu unterstützen.
- `Amazon S3`— Rufen Sie die Implementierungsressourcen von Amplify ab.
- `AWS Security Token Service`— Ermöglicht der AWS Cloud Development Kit (AWS CDK) CLI, die Bereitstellungsrolle zu übernehmen.

- Amazon RDS— Liest Metadaten von DB-Instances, Clustern und Proxys.
- Amazon EC2— Lesen Sie die Informationen zur Verfügbarkeitszone für ein Subnetz.

Die Berechtigungen für diese Richtlinie finden Sie [AmplifyBackendDeployFullAccess](#) in der Referenz zu AWS verwalteten Richtlinien.

Amplify Sie die Aktualisierungen AWS verwalteter Richtlinien

Sehen Sie sich Details zu Aktualisierungen der AWS verwalteten Richtlinien für Amplify an, seit dieser Dienst begonnen hat, diese Änderungen zu verfolgen. Um automatische Warnungen über Änderungen an dieser Seite zu erhalten, abonnieren Sie den RSS-Feed auf der [Dokumenthistorie für AWS Amplify](#)-Seite.

Änderung	Beschreibung	Datum
AmplifyBackendDeployFullAccess – Aktualisierung auf eine bestehende Richtlinie	Fügen Sie der <code>arn:aws:iam::*:parameter/cdk-bootstrap/*</code> Ressource Lesezugriff hinzu, damit Amplify die CDK-Bootstrap-Version im Konto eines Kunden erkennen kann.	31. Mai 2024
AmplifyBackendDeployFullAccess – Aktualisierung auf eine bestehende Richtlinie	Fügen Sie eine neue <code>AmplifyDiscoverRDSVpcConfig</code> Richtlinienerklärung mit Leseberechtigungen für Amazon RDS und Amazon EC2 hinzu, die sowohl von den Ressourcen als auch von den Kontobedingungen abhängig sind. Diese Berechtigungen unterstützen den <code>npx amplify generate schema-from-database</code> Befehl Amplify	17. April 2024

Änderung	Beschreibung	Datum
	<p>Gen 2, mit dem Kunden ein Typescript-Datenschema aus einer vorhandenen SQL-Datenbank generieren können.</p> <p>Fügen Sie die Berechtigungen <code>rds:DescribeDBProxies</code>, <code>rds:DescribeDBInstances</code>, <code>rds:DescribeDBClusters</code>, <code>rds:DescribeDBSubnetGroups</code>, und hinzu. <code>ec2:DescribeSubnets</code>. Der <code>npmx amplify generate schema-from-database</code> Befehl benötigt diese Berechtigungen, um zu überprüfen, ob ein bestimmter DB-Host in Amazon RDS gehostet wird, und um automatisch die Amazon VPC-Konfiguration zu generieren, die für die Bereitstellung der anderen Ressourcen erforderlich ist, die für die Einrichtung einer von einer SQL-Datenbank gestützten AWS AppSync API erforderlich sind.</p>	

Änderung	Beschreibung	Datum
<p>AmplifyBackendDeployFullAccess – Aktualisierung auf eine bestehende Richtlinie</p>	<p>Fügen Sie die <code>cloudformation:DeleteStack</code> Richtlinienaktion hinzu, um das Löschen von Stacks zu unterstützen, wenn die <code>DeleteBranch</code> API aufgerufen wird.</p> <p>Fügen Sie die <code>lambda:GetFunction</code> Richtlinienaktion hinzu, um Hotswapping-Funktionen zu unterstützen.</p> <p>Fügen Sie die <code>lambda:UpdateFunctionConfiguration</code> Richtlinienaktion hinzu, um Updates für die Lambda-Funktion zu unterstützen.</p>	<p>5. April 2024</p>
<p>AdministratorAccess-Amplify — Aktualisierung einer bestehenden Richtlinie</p>	<p>Fügen Sie die <code>cloudformation:UntagResource</code> Berechtigungen <code>cloudformation:TagResource</code> und hinzu, um API-Aufrufe zu AWS CloudFormation unterstützen.</p>	<p>4. April 2024</p>

Änderung	Beschreibung	Datum
AmplifyBackendDeployFullAccess – Aktualisierung auf eine bestehende Richtlinie	<p>Fügen Sie die <code>lambda:InvokeFunction</code> Richtlinienaktion zur Unterstützung von AWS Cloud Development Kit (AWS CDK) Hotswapping hinzu. Der AWS CDK ruft direkt eine Lambda-Funktion auf, um Amazon S3-Asset-Hotswapping durchzuführen.</p> <p>Fügen Sie die <code>lambda:UpdateFunctionCode</code> Richtlinienaktion zur Unterstützung von Hotswapping-Funktionen hinzu.</p>	02. Januar 2024
AmplifyBackendDeployFullAccess – Aktualisierung auf eine bestehende Richtlinie	<p>Fügen Sie Richtlinienaktionen hinzu, um den <code>UpdateApiKey</code> Vorgang zu unterstützen. Dies ist erforderlich, um nach dem Beenden und Neustarten der Sandbox eine erfolgreiche App-Bereitstellung zu ermöglichen, ohne Ressourcen zu löschen.</p>	17. November 2023
AmplifyBackendDeployFullAccess – Aktualisierung auf eine bestehende Richtlinie	<p>Fügen Sie die <code>amplify:GetBackendEnvironment</code> Erlaubnis hinzu, die Bereitstellung von Amplify-Apps zu unterstützen.</p>	6. November 2023

Änderung	Beschreibung	Datum
AmplifyBackendDeployFullAccess – Neue Richtlinie.	Amplify hat eine neue Richtlinie mit den Mindestberechtigungen hinzugefügt, die für die Bereitstellung von Amplify-Backend-Ressourcen erforderlich sind.	8. Oktober 2023
AdministratorAccess-Amplify — Aktualisierung einer bestehenden Richtlinie	Fügen Sie die <code>ecr:DescribeRepositories</code> Berechtigung hinzu, die für die Amplify Command Line Interface (CLI) erforderlich ist.	01. Juni 2023

Änderung	Beschreibung	Datum
<p>AdministratorAccess-Amplify — Aktualisierung einer bestehenden Richtlinie</p>	<p>Fügen Sie eine Richtlinie hinzu, um das Entfernen von Tags aus einer AWS AppSync Ressource zu unterstützen.</p> <p>Fügen Sie eine Richtlinie hinzu, um die Unterstützung der Amazon Polly Polly-Ressource zu unterstützen.</p> <p>Fügen Sie eine Richtlinie hinzu, um die Aktualisierung der OpenSearch Domain-Konfiguration zu unterstützen.</p> <p>Fügen Sie eine Richtlinie hinzu, um das Entfernen von Tags aus einer AWS Identity and Access Management Rolle zu unterstützen.</p> <p>Fügen Sie eine Richtlinie hinzu, um das Entfernen von Tags aus einer Amazon DynamoDB DynamoDB-Ressource zu unterstützen.</p> <p>Fügen Sie dem <code>CLISDKCalls</code> Anweisungsblock die <code>cloudfront:GetCloudFrontOriginAccessIdentityConfig</code></p>	<p>24. Februar 2023</p>

Änderung	Beschreibung	Datum
	<p>Berechtigungen <code>cloudfront:GetCloudFrontOriginAccessIdentity</code> und hinzu, um die Veröffentlichungs- und Hosting-Workflows von Amplify zu unterstützen.</p> <p>Fügen Sie dem <code>s3:PutBucketPublicAccessBlock</code> <code>CLIManageviaCFNPolicy</code> Anweisungsblock die Erlaubnis hinzu, damit der die bewährte Amazon S3 S3-Sicherheitsmethode AWS CLI zur Aktivierung der Amazon S3 S3-Funktion <code>Block Public Access</code> für interne Buckets unterstützen kann.</p> <p>Fügen Sie dem <code>cloudformation:DescribeStacks</code> <code>CLISDKCalls</code> Anweisungsblock die Berechtigung hinzu, um das Abrufen von AWS CloudFormation Kundenstapeln bei Wiederholungsversuchen im Amplify-Backend-Prozessor zu unterstützen, um doppelte Ausführungen zu vermeiden, wenn ein Stack aktualisiert wird.</p> <p>Fügen Sie die Erlaubnis zum Anweisungsblock hinzu.</p>	

Änderung	Beschreibung	Datum
	<code>cloudformation:ListStacks</code> <code>CLICloudformationPolicy</code> Diese Erlaubnis ist erforderlich, um die <code>CloudFormation DescribeStacks</code> Aktion vollständig zu unterstützen.	
AdministratorAccess-Amplify — Aktualisierung einer bestehenden Richtlinie	Fügen Sie Richtlinienaktionen hinzu, damit die serverseitige Rendering-Funktion von Amplify Anwendungsmetriken CloudWatch in die eines Kunden übertragen kann. AWS-Konto	30. August 2022
AdministratorAccess-Amplify — Aktualisierung einer bestehenden Richtlinie	Fügen Sie Richtlinienaktionen hinzu, um den öffentlichen Zugriff auf den Amazon S3 S3-Bucket für die Amplify-Bereitstellung zu blockieren.	27. April 2022

Änderung	Beschreibung	Datum
AdministratorAccess-Amplify — Aktualisierung einer bestehenden Richtlinie	<p>Fügen Sie eine Aktion hinzu, mit der Kunden ihre serverseitig gerenderten Apps (SSR) löschen können. Dadurch kann auch die entsprechende CloudFront Distribution erfolgreich gelöscht werden.</p> <p>Fügen Sie eine Aktion hinzu, damit Kunden mithilfe der Amplify-CLI eine andere Lambda-Funktion angeben können, um Ereignisse aus einer vorhandenen Ereignisquelle zu verarbeiten. Mit diesen Änderungen kann AWS Lambda die Aktion ausgeführt werden. UpdateEventSourceMapping</p>	17. April 2022
AdministratorAccess-Amplify — Aktualisierung einer bestehenden Richtlinie	Fügen Sie eine Richtlinienaktion hinzu, um Amplify UI Builder-Aktionen für alle Ressourcen zu aktivieren.	2. Dezember 2021

Änderung	Beschreibung	Datum
AdministratorAccess-Amplify — Aktualisierung einer bestehenden Richtlinie	<p>Fügen Sie Richtlinienaktionen hinzu, um die Amazon Cognito Cognito-Authentifizierungsfunktion zu unterstützen, die Anbieter sozialer Identitäten verwendet.</p> <p>Fügen Sie eine Richtlinieaktion zur Unterstützung von Lambda-Schichten hinzu.</p> <p>Fügen Sie eine Richtlinieaktion zur Unterstützung der Amplify Storage-Kategorie hinzu.</p>	8. November 2021

Änderung	Beschreibung	Datum
<p>AdministratorAccess-Amplify — Aktualisierung einer bestehenden Richtlinie</p>	<p>Fügen Sie Amazon Lex Lex-Aktionen hinzu, um die Kategorie Amplify Interactions zu unterstützen.</p> <p>Fügen Sie Amazon Rekognition Rekognition-Aktionen hinzu, um die Kategorie Amplify Predictions zu unterstützen.</p> <p>Fügen Sie eine Amazon Cognito Cognito-Aktion hinzu, um die MFA-Konfiguration in Amazon Cognito Cognito-Benutzerpools zu unterstützen.</p> <p>Fügen Sie CloudFormation Aktionen zur Unterstützung hinzu. AWS CloudFormation StackSets</p> <p>Fügen Sie Amazon Location Service Service-Aktionen hinzu, um die Amplify Geo-Kategorie zu unterstützen.</p> <p>Fügen Sie eine Lambda-Aktion hinzu, um Lambda-Schichten in Amplify zu unterstützen.</p> <p>Fügen Sie CloudWatch Log-Aktionen zur Unterstützung von Ereignissen hinzu. CloudWatch</p>	<p>27. September 2021</p>

Änderung	Beschreibung	Datum
	<p>Fügen Sie Amazon S3 S3-Aktionen hinzu, um die Kategorie Amplify Storage zu unterstützen.</p> <p>Fügen Sie Richtlinienaktionen hinzu, um serverseitig gerenderte Apps (SSR) zu unterstützen.</p>	

Änderung	Beschreibung	Datum
<p>AdministratorAccess-Amplify — Aktualisierung einer bestehenden Richtlinie</p>	<p>Konsolidieren Sie alle Amplify-Aktionen in einer einzigen <code>amplify:*</code> Aktion.</p> <p>Fügen Sie eine Amazon S3 S3-Aktion hinzu, um die Verschlüsselung von Amazon S3 S3-Buckets von Kunden zu unterstützen.</p> <p>Fügen Sie IAM-Berechtigungsaktionen hinzu, um Amplify-Apps zu unterstützen, für die Berechtigungsregeln aktiviert sind.</p> <p>Fügen Sie Amazon SNS SNS-Aktionen hinzu, um das Anzeigen von Ausgangstelefonnummern und das Anzeigen, Erstellen, Überprüfen und Löschen von Zieltelefonnummern zu unterstützen.</p> <p>Amplify Studio: Fügen Sie Amazon Cognito, AWS Lambda IAM und AWS CloudFormation Richtlinienaktionen hinzu, um die Verwaltung von Backends in der Amplify-Konsole und Amplify Studio zu ermöglichen.</p> <p>Fügen Sie eine AWS Systems Manager (SSM) -Richtlinie</p>	28. Juli 2021

Änderung	Beschreibung	Datum
	<p>ienenerklärung hinzu, um Amplify-Umgebungsgemeinschaften zu verwalten.</p> <p>Fügen Sie eine AWS CloudFormation <code>ListResources</code> Aktion hinzu, um Lambda-Ebenen für Amplify-Apps zu unterstützen.</p>	
Amplify hat begonnen, Änderungen zu verfolgen	Amplify begann, Änderungen für seine AWS verwalteten Richtlinien nachzuverfolgen.	28. Juli 2021

Fehlerbehebung bei Amplify Identity and Access

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit Amplify und IAM auftreten können.

Themen

- [Ich bin nicht berechtigt, eine Aktion in Amplify durchzuführen](#)
- [Ich bin nicht berechtigt, IAM auszuführen: PassRole](#)
- [Ich möchte Personen außerhalb meines AWS Kontos den Zugriff auf meine Amplify-Ressourcen ermöglichen](#)

Ich bin nicht berechtigt, eine Aktion in Amplify durchzuführen

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht zur Durchführung einer Aktion berechtigt sind, müssen Ihre Richtlinien aktualisiert werden, damit Sie die Aktion durchführen können.

Der folgende Beispielfehler tritt auf, wenn der IAM-Benutzer `mateojackson` versucht, über die Konsole Details zu einer fiktiven `my-example-widget`-Ressource anzuzeigen, jedoch nicht über `amplify:GetWidget`-Berechtigungen verfügt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
amplify:GetWidget on resource: my-example-widget
```

In diesem Fall muss die Richtlinie für den Benutzer mateojackson aktualisiert werden, damit er mit der `amplify:GetWidget`-Aktion auf die `my-example-widget`-Ressource zugreifen kann.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Mit der Veröffentlichung von Amplify Studio sind für das Löschen einer App oder eines Backends beides `amplify` und `amplifybackend` Berechtigungen erforderlich. Wenn ein Administrator eine IAM-Richtlinie geschrieben hat, die nur `amplify` Berechtigungen bereitstellt, wird beim Versuch, eine App zu löschen, ein Berechtigungsfehler angezeigt.

Der folgende Beispielfehler tritt auf, wenn der mateojackson IAM-Benutzer versucht, mit der Konsole eine fiktive `example-amplify-app` Ressource zu löschen, aber nicht über die entsprechenden Berechtigungen verfügt. `amplifybackend:RemoveAllBackends`

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
amplifybackend:RemoveAllBackends on resource: example-amplify-app
```

In diesem Fall bittet Mateo seinen Administrator um die Aktualisierung seiner Richtlinien, um unter Verwendung der Aktion `example-amplify-app` auf die Ressource `amplifybackend:RemoveAllBackends` zugreifen zu können.

Ich bin nicht berechtigt, IAM auszuführen: PassRole

Wenn Sie die Fehlermeldung erhalten, dass Sie nicht berechtigt sind, die `iam:PassRole` Aktion durchzuführen, müssen Ihre Richtlinien aktualisiert werden, damit Sie eine Rolle an Amplify übergeben können.

Einige AWS-Services ermöglichen es Ihnen, eine bestehende Rolle an diesen Dienst zu übergeben, anstatt eine neue Servicerolle oder eine dienstverknüpfte Rolle zu erstellen. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in Amplify auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich möchte Personen außerhalb meines AWS Kontos den Zugriff auf meine Amplify-Ressourcen ermöglichen

Sie können eine Rolle erstellen, die Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation für den Zugriff auf Ihre Ressourcen verwenden können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Im Fall von Diensten, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (Access Control Lists, ACLs) verwenden, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen darüber, ob Amplify diese Funktionen unterstützt, finden Sie unter [So funktioniert Amplify mit IAM](#).
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, AWS-Konten die Ihnen gehören, finden Sie im IAM-Benutzerhandbuch unter [Gewähren des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto, den Sie besitzen](#).
- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie [AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte](#).
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter [Gewähren von Zugriff für extern authentifizierte Benutzer \(Identitätsverbund\)](#) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie im IAM-Benutzerhandbuch unter [Kontenübergreifender Ressourcenzugriff in IAM](#).

Datenschutz in Amplify

AWS Amplify entspricht dem [Modell der AWS geteilten Verantwortung](#) mit der , das Vorschriften und Richtlinien für den Datenschutz beinhaltet. AWS ist verantwortlich für den Schutz der globalen Infrastruktur, die alle AWS Dienste betreibt. AWS behält die Kontrolle über die auf dieser Infrastruktur gehosteten Daten, einschließlich der Sicherheitskonfigurationskontrollen für den Umgang mit Kundeneinhalten und personenbezogenen Daten. AWS Kunden und APN-Partner, die entweder als Datenverantwortliche oder als Datenverarbeiter agieren, sind für alle personenbezogenen Daten verantwortlich, die sie in die AWS Cloud stellen.

Aus Datenschutzgründen empfehlen wir, dass Sie Ihre AWS-Konto Anmeldeinformationen schützen und einzelne Benutzer mit AWS IAM Identity Center oder AWS Identity and Access Management (IAM) einrichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem sollten Sie die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
- Verwenden Sie SSL/TLS, um mit Ressourcen zu kommunizieren. AWS
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein. AWS CloudTrail
- Verwenden Sie AWS Verschlüsselungslösungen zusammen mit allen Standardsicherheitskontrollen innerhalb der AWS Dienste.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu sichern.

Wir empfehlen dringend, in Freitextfeldern wie z. B. im Feld Name keine sensiblen, identifizierenden Informationen wie Kontonummern von Kunden einzugeben. Dies gilt auch, wenn Sie mit Amplify oder anderen AWS Diensten über die Konsole AWS CLI, API oder AWS SDKs arbeiten. Alle Daten, die Sie in Amplify oder andere Dienste eingeben, werden möglicherweise zur Aufnahme in Diagnoseprotokolle aufgenommen. Wenn Sie eine URL für einen externen Server bereitstellen, schließen Sie keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL ein.

Weitere Informationen zum Datenschutz enthält der Blog-Beitrag [AWS Shared Responsibility Model and GDPR](#) im AWS -Sicherheitsblog.

Verschlüsselung im Ruhezustand

„Verschlüsselung im Ruhezustand“ bezieht sich auf den Schutz Ihrer Daten vor unbefugtem Zugriff durch deren Verschlüsselung während der Speicherung. Amplify verschlüsselt standardmäßig die Build-Artefakte einer App mithilfe AWS KMS keys von Amazon S3, die von der verwaltet werden. AWS Key Management Service

Amplify verwendet Amazon CloudFront , um Ihre App für Ihre Kunden bereitzustellen. CloudFront verwendet SSDs, die für Edge Location Points of Presence (POPs) verschlüsselt sind, und verschlüsselte EBS-Volumes für Regional Edge Caches (RECs). Der Funktionscode und die Konfiguration in CloudFront Functions werden immer in einem verschlüsselten Format auf den verschlüsselten SSDs an den PoPs am Edge-Standort und an anderen Speicherorten gespeichert, die von verwendet werden. CloudFront

Verschlüsselung während der Übertragung

Der Begriff „Verschlüsselung während der Übertragung“ bedeutet, dass Ihre Daten davor geschützt werden, abgefangen zu werden, während sie zwischen Kommunikationsendpunkten verschoben werden. Amplify Hosting bietet standardmäßig Verschlüsselung für Daten während der Übertragung. Die gesamte Kommunikation zwischen Kunden und Amplify sowie zwischen Amplify und seinen nachgelagerten Abhängigkeiten ist durch TLS-Verbindungen geschützt, die mit dem Signature Version 4-Signaturprozess signiert wurden. Alle Amplify Hosting-Endpunkte verwenden SHA-256-Zertifikate, die von der Private Certificate Authority verwaltet AWS Certificate Manager werden. Weitere Informationen finden Sie unter [Signaturprozess mit Signaturversion 4](#) und [Was ist ACM PCA?](#).

Verwaltung von Verschlüsselungsschlüsseln

AWS Key Management Service (KMS) ist ein verwalteter Dienst zur Erstellung und Steuerung der Verschlüsselungsschlüssel AWS KMS keys, die zur Verschlüsselung von Kundendaten verwendet werden. AWS Amplify generiert und verwaltet kryptografische Schlüssel zur Verschlüsselung von Daten im Auftrag von Kunden. Es gibt keine Verschlüsselungsschlüssel, die Sie verwalten müssen.

Konformitätsprüfung für AWS Amplify


Externe Prüfer bewerten die Sicherheit und Einhaltung von Vorschriften im AWS Amplify Rahmen mehrerer AWS Compliance-Programme. Dazu gehören SOC, PCI, ISO, HIPAA, MTCS, C5, K-ISMS, ENS High, OSPAR, HITRUST CSF und FINMA.

Informationen darüber, ob ein in den Geltungsbereich bestimmter Compliance-Programme AWS-Service fällt, finden Sie unter Umfang nach Compliance-Programm unter [AWS-Services Umfang nach Compliance-Programm](#). Wählen Sie dort das Compliance-Programm AWS-Services aus, an sind. Allgemeine Informationen finden Sie unter [AWS Compliance-Programme AWS](#).

Sie können Prüfberichte von Drittanbietern unter herunterladen AWS Artifact. Weitere Informationen finden Sie unter [Berichte herunterladen unter](#).

Ihre Verantwortung für die Einhaltung der Vorschriften bei der Nutzung AWS-Services hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab. AWS stellt die folgenden Ressourcen zur Verfügung, die Sie bei der Einhaltung der Vorschriften unterstützen:

- [Schnellstartanleitungen zu Sicherheit und Compliance](#) — In diesen Bereitstellungsleitfäden werden architektonische Überlegungen erörtert und Schritte für die Bereitstellung von Basisumgebungen beschrieben AWS, bei denen Sicherheit und Compliance im Mittelpunkt stehen.
- [Architecting for HIPAA Security and Compliance on Amazon Web Services](#) — In diesem Whitepaper wird beschrieben, wie Unternehmen HIPAA-fähige Anwendungen erstellen AWS können.

 Note

AWS-Services Nicht alle sind HIPAA-fähig. Weitere Informationen finden Sie in der [Referenz für HIPAA-berechtigte Services](#).

- [AWS Compliance-Ressourcen](#) — Diese Sammlung von Arbeitsmapen und Leitfäden gilt möglicherweise für Ihre Branche und Ihren Standort.
- [AWS Leitfäden zur Einhaltung von Vorschriften für Kunden](#) — Verstehen Sie das Modell der gemeinsamen Verantwortung aus dem Blickwinkel der Einhaltung von Vorschriften. In den Leitfäden werden die bewährten Verfahren zur Sicherung zusammengefasst AWS-Services und die Leitlinien den Sicherheitskontrollen in verschiedenen Frameworks (einschließlich des National Institute of Standards and Technology (NIST), des Payment Card Industry Security Standards Council (PCI) und der International Organization for Standardization (ISO)) zugeordnet.
- [Evaluierung von Ressourcen anhand von Regeln](#) im AWS Config Entwicklerhandbuch — Der AWS Config Service bewertet, wie gut Ihre Ressourcenkonfigurationen den internen Praktiken, Branchenrichtlinien und Vorschriften entsprechen.

- [AWS Security Hub](#)— Auf diese AWS-Service Weise erhalten Sie einen umfassenden Überblick über Ihren internen Sicherheitsstatus. AWS Security Hub verwendet Sicherheitskontrollen, um Ihre AWS -Ressourcen zu bewerten und Ihre Einhaltung von Sicherheitsstandards und bewährten Methoden zu überprüfen. Eine Liste der unterstützten Services und Kontrollen finden Sie in der [Security-Hub-Steuerungsreferenz](#).
- [Amazon GuardDuty](#) — Dies AWS-Service erkennt potenzielle Bedrohungen für Ihre Workloads AWS-Konten, Container und Daten, indem es Ihre Umgebung auf verdächtige und böswillige Aktivitäten überwacht. GuardDuty kann Ihnen helfen, verschiedene Compliance-Anforderungen wie PCI DSS zu erfüllen, indem es die in bestimmten Compliance-Frameworks vorgeschriebenen Anforderungen zur Erkennung von Eindringlingen erfüllt.
- [AWS Audit Manager](#)— Auf diese AWS-Service Weise können Sie Ihre AWS Nutzung kontinuierlich überprüfen, um das Risikomanagement und die Einhaltung von Vorschriften und Industriestandards zu vereinfachen.

Sicherheit der Infrastruktur in AWS Amplify

Als verwalteter Dienst AWS Amplify ist er durch AWS globale Netzwerksicherheit geschützt. Informationen zu AWS Sicherheitsdiensten und zum AWS Schutz der Infrastruktur finden Sie unter [AWS Cloud-Sicherheit](#). Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter [Infrastructure Protection](#) in Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS veröffentlichte API-Aufrufe, um über das Netzwerk auf Amplify zuzugreifen. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit [AWS Security Token Service](#) (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Protokollierung und Überwachung von Sicherheitsereignissen in Amplify

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von Amplify und Ihren anderen AWS Lösungen. AWS bietet die folgenden Überwachungstools, um Amplify zu beobachten, zu melden, wenn etwas nicht stimmt, und gegebenenfalls automatische Maßnahmen zu ergreifen:

- Amazon CloudWatch überwacht in Echtzeit Ihre AWS Ressourcen und die Anwendungen, auf denen Sie laufen AWS. Sie können Kennzahlen sammeln und verfolgen, benutzerdefinierte Dashboards erstellen und Alarme einrichten, die Sie benachrichtigen oder Maßnahmen ergreifen, wenn eine bestimmte Metrik einen von Ihnen festgelegten Schwellenwert erreicht. Sie können beispielsweise die CPU-Auslastung oder andere Kennzahlen Ihrer Amazon Elastic Compute Cloud (Amazon EC2) -Instances CloudWatch verfolgen und bei Bedarf automatisch neue Instances starten. Weitere Informationen zur Verwendung von CloudWatch Metriken und Alarmen mit Amplify finden Sie unter [Überwachen](#).
- Mit Amazon CloudWatch Logs können Sie Ihre Protokolldateien von Amazon EC2 EC2-Instances und anderen Quellen überwachen AWS CloudTrail, speichern und darauf zugreifen. CloudWatch Logs können Informationen in den Protokolldateien überwachen und Sie benachrichtigen, wenn bestimmte Schwellenwerte erreicht werden. Sie können Ihre Protokolldaten auch in einem sehr robusten Speicher archivieren. Weitere Informationen finden Sie im [Amazon CloudWatch Logs-Benutzerhandbuch](#).
- AWS CloudTrailerfasst API-Aufrufe und zugehörige Ereignisse, die von oder im Namen Ihres AWS Kontos getätigt wurden, und übermittelt die Protokolldateien an einen von Ihnen angegebenen Amazon Simple Storage Service (Amazon S3) -Bucket. Sie können feststellen, welche Benutzer und Konten angerufen wurden AWS, von welcher Quell-IP-Adresse aus die Aufrufe getätigt wurden und wann die Aufrufe erfolgten. Weitere Informationen finden Sie unter [Protokollieren von Amplify-API-Aufrufen mit AWS CloudTrail](#).
- Amazon EventBridge ist ein serverloser Event-Bus-Service, der es einfach macht, Ihre Anwendungen mit Daten aus einer Vielzahl von Quellen zu verbinden. EventBridge liefert einen Stream von Echtzeitdaten aus Ihren eigenen Anwendungen, oftware-as-a S-Service (SaaS) -Anwendungen und AWS -Diensten und leitet diese Daten an Ziele weiter, wie AWS Lambda z. Auf diese Weise können Sie Ereignisse überwachen, die in Diensten auftreten, und ereignisgesteuerte Architekturen erstellen. Weitere Informationen finden Sie im [EventBridge Amazon-Benutzerhandbuch](#).

Serviceübergreifende Confused-Deputy-Prävention

Das Problem des verwirrten Stellvertreters ist ein Sicherheitsproblem, bei dem eine Entität, die keine Berechtigung zur Durchführung einer Aktion hat, eine privilegiere Entität zur Durchführung der Aktion zwingen kann. In AWS kann ein dienstübergreifendes Identitätswechsels zum Problem des verwirrten Stellvertreters führen. Ein dienstübergreifender Identitätswechsel kann auftreten, wenn ein Dienst (der Anruf-Dienst) einen anderen Dienst anruft (den aufgerufenen Dienst). Der aufrufende Service kann manipuliert werden, um seine Berechtigungen zu verwenden, um Aktionen auf die Ressourcen eines anderen Kunden auszuführen, für die er sonst keine Zugriffsberechtigung haben sollte. Um dies zu verhindern, bietet AWS Tools, mit denen Sie Ihre Daten für alle Services mit Serviceprinzipalen schützen können, die Zugriff auf Ressourcen in Ihrem Konto erhalten haben.

Wir empfehlen, die Kontextschlüssel [aws:SourceArn](#) und die [aws:SourceAccount](#) globalen Bedingungsschlüssel in Ressourcenrichtlinien zu verwenden, um die Berechtigungen einzuschränken, die der AWS Amplify Ressource einen anderen Dienst gewähren. Wenn Sie beide globalen Bedingungskontextschlüssel verwenden, müssen der `aws:SourceAccount`-Wert und das Konto im `aws:SourceArn`-Wert dieselbe Konto-ID verwenden, wenn sie in derselben Richtlinienanweisung verwendet werden.

Der Wert von `aws:SourceArn` muss der Branch-ARN der Amplify-App sein. Geben Sie diesen Wert im Format `arn:Partition:amplify:Region:Account:apps/AppId/branches/BranchName` an.

Der effektivste Weg, um sich vor dem Confused-Deputy-Problem zu schützen, ist die Verwendung des globalen Bedingungskontext-Schlüssels `aws:SourceArn` mit dem vollständigen ARN der Ressource. Wenn Sie den vollständigen ARN der Ressource nicht kennen oder wenn Sie mehrere Ressourcen angeben, verwenden Sie den globalen Bedingungskontext-Schlüssel `aws:SourceArn` mit Platzhaltern (*) für die unbekannt Teile des ARN. z. B. `arn:aws:servicename::123456789012:*`.

Das folgende Beispiel zeigt eine Rollenvertrauensrichtlinie, die Sie anwenden können, um den Zugriff auf jede Amplify-App in Ihrem Konto einzuschränken und das Problem mit dem verwirrten Stellvertreter zu vermeiden. Um diese Richtlinie zu verwenden, ersetzen Sie den rot kursiv geschriebenen Text in der Beispielrichtlinie durch Ihre eigenen Informationen.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
```

```

"Effect": "Allow",
"Principal": {
  "Service": [
    "amplify.me-south-1.amazonaws.com",
    "amplify.eu-south-1.amazonaws.com",
    "amplify.ap-east-1.amazonaws.com",
    "amplifybackend.amazonaws.com",
    "amplify.amazonaws.com"
  ]
},
"Action": "sts:AssumeRole",
"Condition": {
  "ArnLike": {
    "aws:SourceArn": "arn:aws:amplify:us-east-1:123456789012:apps/*"
  },
  "StringEquals": {
    "aws:SourceAccount": "123456789012"
  }
}
}
}
}

```

Das folgende Beispiel zeigt eine Rollenvertrauensrichtlinie, die Sie anwenden können, um den Zugriff auf eine bestimmte Amplify-App in Ihrem Konto einzuschränken und das Problem mit dem verwirrten Stellvertreter zu verhindern. Um diese Richtlinie zu verwenden, ersetzen Sie den rot kursiv gedruckten Text in der Beispielrichtlinie durch Ihre eigenen Informationen.

```

{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": [
        "amplify.me-south-1.amazonaws.com",
        "amplify.eu-south-1.amazonaws.com",
        "amplify.ap-east-1.amazonaws.com",
        "amplifybackend.amazonaws.com",
        "amplify.amazonaws.com"
      ]
    },
    "Action": "sts:AssumeRole",
    "Condition": {

```

```
    "ArnLike": {
      "aws:SourceArn": "arn:aws:amplify:us-east-1:123456789012:apps/d123456789/
branches/*"
    },
    "StringEquals": {
      "aws:SourceAccount": "123456789012"
    }
  }
}
```

Bewährte Sicherheitsmethoden für Amplify

Amplify bietet eine Reihe von Sicherheitsfunktionen, die Sie bei der Entwicklung und Implementierung Ihrer eigenen Sicherheitsrichtlinien berücksichtigen sollten. Die folgenden bewährten Methoden stellen allgemeine Richtlinien und keine vollständige Sicherheitslösung dar. Da diese bewährten Methoden für Ihre Umgebung möglicherweise nicht angemessen oder ausreichend sind, sollten Sie sie als hilfreiche Empfehlungen und nicht als bindend ansehen.

Verwendung von Cookies mit der Amplify-Standarddomain

Wenn Sie Amplify verwenden, um eine Web-App bereitzustellen, hostet Amplify sie für Sie auf der `amplifyapp.com` Standarddomain. Sie können Ihre App auf einer URL anzeigen, die als formatiert ist. `https://branch-name.d1m7bkiki6tdw1.amplifyapp.com`

Um die Sicherheit Ihrer Amplify-Anwendungen zu erhöhen, ist die Domain `amplifyapp.com` in der [Public Suffix List \(PSL\)](#) registriert. Aus Sicherheitsgründen empfehlen wir Ihnen, Cookies mit einem `__Host-` Präfix zu verwenden, falls Sie jemals sensible Cookies im Standard-Domainnamen für Ihre Amplify-Anwendungen einrichten müssen. Diese Vorgehensweise hilft Ihnen dabei, Ihre Domain vor CSRF-Versuchen (Cross-Site Request Forgery Attempts, Anforderungsfälschung zwischen Websites) zu schützen. Weitere Informationen finden Sie auf der [Set-Cookie](#)-Seite im Mozilla Developer Network.

Amplify Kontingente für Hosting-Dienste

Im Folgenden sind die Servicekontingente für das AWS Amplify Hosting aufgeführt.

Servicekontingenten (früher als Limits bezeichnet) sind die maximale Anzahl von Serviceressourcen oder Vorgängen für Sie AWS-Konto.

Neu AWS-Konten sind die Kontingente für Apps und gleichzeitige Jobs reduziert. AWS erhöht diese Kontingente automatisch auf der Grundlage Ihrer Nutzung. Sie können auch eine Kontingenterhöhung beantragen.

Die Service-Quotas-Konsole enthält Informationen zu Kontingenten für Ihr Konto. Sie können die Service Quotas-Konsole verwenden, um Standard-Kontingente anzuzeigen und [Kontingenterhöhungen für einstellbare Kontingente anzufordern](#). Weitere Informationen finden Sie unter [Beantragen einer Kontingenterhöhung](#) im Service Quotas-Benutzerhandbuch.

Name	Standard	Anpas	Beschreibung
Apps	Jede unterstützte Region: 25	Ja	Die maximale Anzahl von Apps, die Sie in AWS Amplify Console in diesem Konto in der aktuellen Region erstellen können.
Branches pro App	Jede unterstützte Region: 50	Nein	Die maximale Anzahl von Branches pro App, die Sie in diesem Konto in der aktuellen Region erstellen können.
Größe des Build-Artefakts	Jede unterstützte Region: 5 Gigabyte	Nein	Die maximale Größe (in GB) eines App-Build-Artefakts. Ein Build-Artefakt wird nach einem Build von AWS Amplify Console bereitgestellt.

Name	Standard	Anpas	Beschreibung
Cache-Artifaktgröße	Jede unterstützte Region: 5 Gigabyte	Nein	Die maximale Größe (in GB) eines Cache-Artifakts.
Gleichzeitige Aufgaben	Jede unterstützte Region: 5	Yes (Ja)	Die maximale Anzahl von gleichzeitigen Aufträgen, die Sie in diesem Konto in der aktuellen Region erstellen können.
Domains pro App	Jede unterstützte Region: 5	Yes (Ja)	Die maximale Anzahl von Domains pro App, die Sie in diesem Konto in der aktuellen Region erstellen können.
Umgebungscache-Artifaktgröße	Jede unterstützte Region: 5 Gigabyte	Nein	Die maximale Größe (in GB) des Umgebungs-Cache-Artifakts.
Größe der manuell bereitgestellten ZIP-Datei	Jede unterstützte Region: 5 Gigabyte	Nein	Die maximale Größe (in GB) einer manuell bereitgestellten ZIP-Datei.
Maximale Anzahl von App-Erstellungen pro Stunde	Jede unterstützte Region: 25	Nein	Die maximale Anzahl von Apps, die Sie in AWS Amplify Console pro Stunde in diesem Konto in der aktuellen Region erstellen können.

Name	Standard	Anpas	Beschreibung
Tokens pro Sekunde anfordern	Jede unterstützte Region: 20 000	Ja	Die maximale Anzahl von Anforderungstoken pro Sekunde für eine App. Amplify Hosting weist Anfragen Token zu, basierend auf der Menge der Ressourcen (Verarbeitungszeit und Datenübertragung), die sie verbrauchen.
Subdomains pro Domain	Jede unterstützte Region: 50	Nein	Die maximale Anzahl von Subdomains pro Domain, die Sie in diesem Konto in der aktuellen Region erstellen können.
Webhooks pro App	Jede unterstützte Region: 50	Ja	Die maximale Anzahl von Webhooks pro App, die Sie in diesem Konto in der aktuellen Region erstellen können.

Weitere Informationen zu Amplify-Servicekontingenten finden Sie unter [AWS Amplify Endpunkte und Kontingente](#) in der. Allgemeine AWS-Referenz

Fehlerbehebung bei Amplify Hosting

Wenn Sie bei der Arbeit mit Amplify Hosting auf Fehler oder Bereitstellungsprobleme stoßen, lesen Sie die Themen in diesem Abschnitt.

Themen

- [Behebung allgemeiner Amplify-Probleme](#)
- [Behebung von Problemen mit dem Amazon Linux 2023 Build Image](#)
- [Problembehandlung bei benutzerdefinierten Domänen](#)
- [Fehlerbehebung bei serverseitig gerenderten Anwendungen](#)

Behebung allgemeiner Amplify-Probleme

Die folgenden Informationen können Ihnen bei der Behebung allgemeiner Probleme mit Amplify Hosting helfen.

Themen

- [HTTP 429-Statuscode \(Zu viele Anfragen\)](#)

HTTP 429-Statuscode (Zu viele Anfragen)

Amplify steuert die Anzahl der Anfragen pro Sekunde (RPS) an Ihre Website auf der Grundlage der Verarbeitungszeit und der Datenübertragung, die eingehende Anfragen verbrauchen. Wenn Ihre Anwendung einen HTTP-Statuscode 429 zurückgibt, überschreiten eingehende Anfragen die für Ihre Anwendung vorgesehene Bearbeitungszeit und Datenübertragung. Dieses Anwendungslimit wird durch die Servicequote von Amplify verwaltet. `REQUEST_TOKENS_PER_SECOND` Weitere Informationen zu Kontingenten finden Sie unter [Amplify Kontingente für Hosting-Dienste](#).

Um dieses Problem zu beheben, empfehlen wir, Ihre Anwendung zu optimieren, um die Anforderungsdauer und die Datenübertragung zu reduzieren und so den RPS der App zu erhöhen. Bei denselben 20.000 Tokens kann beispielsweise eine hochoptimierte SSR-Seite, die innerhalb von 100 Millisekunden reagiert, höhere RPS unterstützen als eine Seite mit einer Latenz von mehr als 200 Millisekunden.

In ähnlicher Weise verbraucht eine Anwendung, die eine Antwortgröße von 1 MB zurückgibt, mehr Token als eine Anwendung, die eine Antwortgröße von 250 KB zurückgibt.

Wir empfehlen Ihnen außerdem, den CloudFront Amazon-Cache zu nutzen, indem Sie Cache-Control Header so konfigurieren, dass die Zeit, in der eine bestimmte Antwort im Cache aufbewahrt wird, maximiert wird. Anfragen, die vom CloudFront Cache aus bedient werden, werden nicht auf das Ratenlimit angerechnet. Jede CloudFront Distribution kann bis zu 250.000 Anfragen pro Sekunde verarbeiten, sodass Sie Ihre App mithilfe des Caches sehr stark skalieren können. Weitere Informationen zum CloudFront Cache finden Sie unter [Optimizing Caching and Availability](#) im Amazon CloudFront Developer Guide.

Behebung von Problemen mit dem Amazon Linux 2023 Build Image

Die folgenden Informationen können Ihnen bei der Behebung von Problemen mit dem Build-Image von Amazon Linux 2023 (AL2023) helfen.

Themen

- [Wie führe ich Amplify-Funktionen mit der Python-Laufzeit aus?](#)
- [Wie führe ich Befehle aus, für die Superuser- oder Root-Rechte erforderlich sind](#)

Wie führe ich Amplify-Funktionen mit der Python-Laufzeit aus?

Amplify Hosting verwendet jetzt standardmäßig das Amazon Linux 2023 Build-Image, wenn Sie eine neue Anwendung bereitstellen. AL2023 ist mit den Python-Versionen 3.8, 3.9, 3.10 und 3.11 vorinstalliert.

Aus Gründen der Abwärtskompatibilität mit dem Amazon Linux 2-Image sind auf dem AL2023-Build-Image Symlinks für ältere Versionen von Python vorinstalliert. Daher müssen Sie die Build-Befehle in der Build-Spezifikation Ihrer Anwendung nicht mehr anhand der Anweisungen in den [Amplify Hosting GitHub FAQ](#) aktualisieren.

Standardmäßig wird Python Version 3.10 global verwendet. Um Ihre Funktionen mit einer bestimmten Python-Version zu erstellen, führen Sie die folgenden Befehle in der Build-Spezifikationsdatei Ihrer Anwendung aus.

```
version: 1
backend:
  phases:
    build:
```

```
commands:
  # use a python version globally
  - pyenv global 3.11
  # verify python version
  - python --version
  # install pipenv
  - pip install --user pipenv
  # add to path
  - export PATH=$PATH:/root/.local/bin
  # verify pipenv version
  - pipenv --version
  - amplifyPush --simple
```

Wie führe ich Befehle aus, für die Superuser- oder Root-Rechte erforderlich sind

Wenn Sie das Amazon Linux 2023 Build-Image verwenden und bei der Ausführung von Systembefehlen, die Superuser- oder Root-Rechte erfordern, eine Fehlermeldung erhalten, müssen Sie diese Befehle mit dem `sudo` Linux-Befehl ausführen. Wenn bei der Ausführung beispielsweise ein Fehler auftritt `yum install -y gcc`, verwenden Sie `sudo yum install -y gcc`.

Das Amazon Linux 2-Build-Image verwendete den Root-Benutzer, aber das AL2023-Image von Amplify führt Ihren Code mit einem benutzerdefinierten `amplify` Benutzer aus. Amplify gewährt diesem Benutzer Rechte, Befehle mit dem `sudo` Linux-Befehl auszuführen. Es hat sich bewährt, es `sudo` für Befehle zu verwenden, für die Superuser-Rechte erforderlich sind.

Problembehandlung bei benutzerdefinierten Domänen

Wenn beim Verbinden einer benutzerdefinierten Domain mit Ihrer Amplify-Anwendung Probleme auftreten, finden Sie unter Hilfe [Problembehandlung bei benutzerdefinierten Domains](#) weitere Informationen.

Fehlerbehebung bei serverseitig gerenderten Anwendungen

Wenn Sie Probleme bei der Bereitstellung einer SSR-App [Fehlerbehebung bei SSR-Bereitstellungen](#) für Amplify haben, finden Sie Hilfe unter.

AWS AmplifyReferenz zum Hosting

Verwenden Sie die Themen in diesem Abschnitt, um detailliertes Referenzmaterial für zu findenAWS Amplify.

Themen

- [AWS CloudFormation-Support](#)
- [AWS Command Line Interface-Support](#)
- [Unterstützung für das Tagging von Ressourcen](#)
- [Amplify Hosting-API](#)

AWS CloudFormation-Support

BenutzenAWS CloudFormationVorlagen zur Bereitstellung von Amplify-Ressourcen, die wiederholbare und zuverlässige Implementierungen von Webanwendungen ermöglichen.AWS CloudFormationstellt Ihnen eine gemeinsame Sprache zur Beschreibung und Bereitstellung aller Infrastrukturressourcen in Ihrer Cloud-Umgebung zur Verfügung und vereinfacht den Rollout auf mehrereAWSKonten und/oder Regionen mit nur wenigen Klicks.

Informationen zu Amplify Hosting finden Sie in[VerstärkenCloudFormationDokumentation](#).

Informationen zu Amplify Studio finden Sie in[Amplify UI BuilderCloudFormationDokumentation](#).

AWS Command Line Interface-Support

Benutze dieAWS Command Line Interfaceum Amplify-Apps programmgesteuert über die Befehlszeile zu erstellen. Informationen finden Sie auf der[AWS CLIdokumentation](#).

Unterstützung für das Tagging von Ressourcen

Sie können das verwendenAWS Command Line Interfaceum Amplify-Ressourcen zu taggen. Weitere Informationen finden Sie auf der[AWS CLIDokumentation zur Tag-Ressource](#).

Amplify Hosting-API

Diese Referenz enthält Beschreibungen der Aktionen und Datentypen für die Amplify Hosting API. Weitere Informationen finden Sie auf der[Amplify API-Referenzdokumentation](#).

Dokumenthistorie für AWS Amplify

In der folgenden Tabelle werden die wichtigen Änderungen an der Dokumentation seit der letzten Version von AWS Amplify beschrieben.

- Letzte Aktualisierung der Dokumentation: 31. Mai 2024

Änderung	Beschreibung	Datum
Das Thema verwaltete Richtlinien wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	31. Mai 2024
Das Thema verwaltete Richtlinien wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	17. April 2024
Das Kapitel „Erste Schritte“ wurde aktualisiert	Das Erste Schritte mit Amplify Hosting Kapitel wurde aktualisiert und verwendet nun eine Beispielanwendung von Next.js im Tutorial.	12. April 2024
Das Thema verwaltete Richtlinien wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	5. April 2024

Änderung	Beschreibung	Datum
Das Thema verwaltete Richtlinien wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	4. April 2024
Neues Kapitel zur Fehlerbehebung	Das Fehlerbehebung bei Amplify Hosting Kapitel wurde hinzugefügt, in dem beschrieben wird, wie Probleme behoben werden können, die bei Anwendungen auftreten, die auf Amplify Hosting bereitgestellt werden.	2. April 2024
Neue Unterstützung für benutzerdefinierte SSL/TLS-Zertifikate	Dem Einrichtung benutzerdefinierter Domains Kapitel wurde das Verwenden von SSL/TLS-Zertifikaten Thema hinzugefügt, das die Amplify-Unterstützung für benutzerdefinierte SSL/TLS-Zertifikate beschreibt, wenn eine App mit einer benutzerdefinierten Domain verbunden wird.	20. Februar 2024
Das Thema verwaltete Richtlinien wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	2. Januar 2024

Änderung	Beschreibung	Datum
Neue Unterstützung für SSR-Frameworks	Das Amplify Sie die Unterstützung für SSR-Frameworks Thema zur Beschreibung der Amplify-Unterstützung für jedes JavaScript-basierte SSR-Framework mit einem Open-Source-Adapter wurde hinzugefügt.	19. November 2023
Neue Unterstützung für den Start der Bildoptimierungsfunktion	Das Bildoptimierung für SSR-Apps Thema zur Beschreibung der integrierten Unterstützung für die Bildoptimierung für serverseitig gerenderte Apps wurde hinzugefügt.	19. November 2023
Das Thema „Verwaltete Richtlinien“ wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	17. November 2023
Das Thema verwaltete Richtlinien wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	6. November 2023

Änderung	Beschreibung	Datum
Neues Thema für Wildcard-Subdomains	Das Wildcard-Unterdomänen Thema zur Beschreibung der Unterstützung von Wildcard-Subdomänen in benutzerdefinierten Domänen wurde hinzugefügt.	6. November 2023
Neue von verwaltete Richtlinie	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die neue AmplifyBackendDeployFullAccess AWS verwaltete Richtlinie für Amplify zu beschreiben.	8. Oktober 2023
Neue Unterstützung für Monorepo-Frameworks-Funktionen	Das Monorepo-Build-Einstellungen Thema wurde aktualisiert, um die Unterstützung für die Bereitstellung von Apps in Monorepos zu beschreiben, die mit npm workspace, pnpm workspace, Yarn workspace, Nx und Turborepo erstellt wurden.	19. Juni 2023
Das Thema „Verwaltete Richtlinien“ wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	01. Juni 2023

Änderung	Beschreibung	Datum
Das Thema verwaltete Richtlinien wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	24. Februar 2023
Das Kapitel zum serverseitigen Rendern wurde aktualisiert	Das Stellen Sie serverseitig gerenderte Apps mit Amplify Hosting bereit Kapitel wurde aktualisiert, um die jüngsten Änderungen an der Unterstützung von Amplify für die Versionen 12 und 13 von Next.js zu beschreiben.	17. November 2022
Das Thema „Verwaltete Richtlinien“ wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	30. August 2022
Das Thema verwaltete Richtlinien wurde aktualisiert	Das Ein Backend für eine Anwendung erstellen Thema wurde aktualisiert, um zu beschreiben, wie ein Backend mit Amplify Studio bereitgestellt wird.	23. August 2022

Änderung	Beschreibung	Datum
Das Thema „Verwaltete Richtlinien“ wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	27. April 2022
Das Thema verwaltete Richtlinien wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	17. April 2022
Einführung einer neuen GitHub App-Funktion	Das Einrichtung des Amplify-Zugriffs auf GitHub Repositories Thema zur Beschreibung der neuen GitHub App zur Autorisierung des Amplify-Zugriffs auf Ihr GitHub Repository wurde hinzugefügt.	5. April 2022
Einführung der neuen Amplify Studio-Funktion	Das Willkommen bei AWS Amplify Hosting Thema wurde aktualisiert, um die Updates für Amplify Studio zu beschreiben, die einen visuellen Designer zum Erstellen von UI-Komponenten bieten, die Sie mit Ihren Backend-Daten verbinden können.	2. Dezember 2021

Änderung	Beschreibung	Datum
Das Thema „Verwaltete Richtlinien“ wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zur Unterstützung von Amplify Studio zu beschreiben.	2. Dezember 2021
Das Thema verwaltete Richtlinien wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	8. November 2021
Das Thema verwaltete Richtlinien wurde aktualisiert	Das AWS verwaltete Richtlinien für AWS Amplify Thema wurde aktualisiert, um die jüngsten Änderungen an den AWS verwalteten Richtlinien für Amplify zu beschreiben.	27. September 2021
Neues Thema „Verwaltete Richtlinien“	Das AWS verwaltete Richtlinien für AWS Amplify Thema zur Beschreibung der AWS verwalteten Richtlinien für Amplify und der jüngsten Änderungen an diesen Richtlinien wurde hinzugefügt.	28. Juli 2021

Änderung	Beschreibung	Datum
Das Kapitel zum serverseitigen Rendern wurde aktualisiert	Das Stellen Sie serverseitig gerenderte Apps mit Amplify Hosting bereit Kapitel wurde aktualisiert, um die neue Unterstützung für Next.js Version 10 zu beschreiben. x. x und Next.js Version 11.	22. Juli 2021
Das Kapitel „Konfiguration der Build-Einstellungen“ wurde aktualisiert	Das Monorepo-Build-Einstellungen Thema wurde hinzugefügt, um zu beschreiben, wie die Build-Einstellungen und die neue AMPLIFY_MONOREPO_APP_ROOT Umgebungsvariable konfiguriert werden, wenn eine Monorepo-App mit Amplify bereitgestellt wird.	20. Juli 2021

Änderung	Beschreibung	Datum
Das Kapitel über Feature-Branch-Bereitstellungen wurde aktualisiert	<p>Es wurde ein Automatische Generierung der Amplify-Konfiguration während der Erstellung (nur Gen 1-Apps) Thema hinzugefügt, das beschreibt, wie die <code>aws-exports.js</code> Datei während der Erstellung automatisch generiert wird. Das Bedingte Backend-Builds (nur Apps der Generation 1) Thema zur Beschreibung der Aktivierung bedingter Backend-Builds wurde hinzugefügt. Es wurde ein Verwenden Sie Amplify-Backends für mehrere Apps (nur Gen-1-Apps) Thema hinzugefügt, das beschreibt, wie vorhandene Backends wiederverwendet werden können, wenn Sie eine neue App erstellen, einen neuen Branch mit einer vorhandenen App verbinden oder ein vorhandenes Frontend aktualisieren, sodass es auf eine andere Backend-Umgebung verweist.</p>	30. Juni 2021

Änderung	Beschreibung	Datum
Das Kapitel Sicherheit wurde aktualisiert	Das Datenschutz in Amplify Thema wurde hinzugefügt, in dem beschrieben wird, wie das Modell der gemeinsamen Verantwortung angewendet wird und wie Amplify Verschlüsselung einsetzt, um Ihre Daten im Ruhezustand und bei der Übertragung zu schützen.	3. Juni 2021
Neue Unterstützung für den Start der SSR-Funktion	Das Stellen Sie serverseitig gerenderte Apps mit Amplify Hosting bereit Kapitel zur Beschreibung der Amplify-Unterstützung für Web-Apps, die serverseitiges Rendering (SSR) verwenden und mit Next.js erstellt wurden, wurde hinzugefügt.	18. Mai 2021
Neues Kapitel bezüglich der Sicherheit	Es wurde ein Sicherheit in Amplify Kapitel hinzugefügt, in dem beschrieben wird, wie Sie das Modell der gemeinsamen Verantwortung bei der Verwendung von Amplify anwenden und wie Sie Amplify konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen.	26. März 2021

Änderung	Beschreibung	Datum
Das Thema benutzerdefinierte Builds wurde aktualisiert	Das Thema Custom Build-Images und Live-Paket-Updates wurde aktualisiert und beschreibt nun, wie ein Custom Build-Image konfiguriert wird, das in Amazon Elastic Container Registry Public gehostet wird.	12. März 2021
Das Thema Überwachung wurde aktualisiert	Das Thema Überwachung wurde aktualisiert und beschreibt nun, wie Sie auf Amazon CloudWatch Metrics-Daten zugreifen und Alarme einrichten können.	2. Februar 2021
Neues Thema zur CloudTrail Protokollierung	Es wurde das AWS CloudTrail Thema Logging Amplify API calls using hinzugefügt, um zu beschreiben, wie alle API-Aktionen für die AWS Amplify Konsolen-API-Referenz und die AWS Amplify Admin-UI-API-Referenz AWS CloudTrail erfasst und protokolliert werden.	2. Februar 2021

Änderung	Beschreibung	Datum
Einführung einer neuen Admin-UI-Funktion	Das Willkommen bei AWS Amplify Hosting Thema wurde aktualisiert, um die neue Admin-Benutzeroberfläche zu beschreiben, die eine visuelle Oberfläche für Frontend-Web- und Mobilentwickler bietet, um App-Backends außerhalb von zu erstellen und zu verwalten. AWS Management Console	1. Dezember 2020
Start der neuen Funktion im Leistungsmodus	Das Thema „ App-Leistung verwalten “ wurde aktualisiert und beschreibt nun, wie der Leistungsmodus aktiviert werden kann, um eine schnellere Hosting-Leistung zu erzielen.	4. November 2020
Das Thema „Benutzerdefinierte Header“ wurde aktualisiert	Das Thema Benutzerdefinierte Header wurde aktualisiert, um zu beschreiben, wie benutzerdefinierte Header für eine Amplify-App mithilfe der Konsole oder durch Bearbeiten einer YAML-Datei definiert werden.	28. Oktober 2020

Änderung	Beschreibung	Datum
Einführung der neuen Funktion für auto Subdomains	<p>Das Thema Automatische Subdomänen für eine benutzerdefinierte Route 53-Domain einrichten wurde hinzugefügt, um zu beschreiben, wie musterbasierte Feature-Branch-Bereitstellungen für eine App verwendet werden, die mit einer benutzerdefinierten Amazon Route 53-Domain verbunden ist. Das Thema Webvorschauzugriff mit Subdomänen wurde hinzugefügt, um zu beschreiben, wie Webvorschauen anhand von Pull-Requests eingerichtet werden, sodass sie mit Subdomänen zugänglich sind.</p>	20. Juni 2020
Neues Thema für Benachrichtigungen	<p>Das Thema Benachrichtigungen wurde hinzugefügt, um zu beschreiben, wie E-Mail-Benachrichtigungen für eine Amplify-App eingerichtet werden, um Stakeholder oder Teammitglieder zu benachrichtigen, wenn ein Build erfolgreich ist oder fehlschlägt.</p>	20. Juni 2020

Änderung	Beschreibung	Datum
Das Thema „Benutzerdefinierte Domains“ wurde aktualisiert	Das Einrichtung benutzerdefinierter Domains Thema wurde aktualisiert, um die Verfahren zum Hinzufügen benutzerdefinierter Domains in Amazon Route 53 und Google Domains zu verbessern. GoDaddy Dieses Update enthält auch neue Informationen zur Fehlerbehebung bei der Einrichtung benutzerdefinierter Domains.	12. Mai 2020
AWS Amplify Veröffentlichung	In dieser Version wird Amplify eingeführt.	26. November 2018

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.