Bericht

**Public Cloud Services HS23**

Dozent: Sebastian Graf

Studenten: Fabian Heuberger  
Etienne Frei  
Yannick Hohler

CI Pipeline on AWS

Ein Bild, das Rad, orange enthält.

Automatisch generierte Beschreibung

# Inhaltsverzeichnis

[Use Case 3](#_Toc154850147)

[Architektur 4](#_Toc154850148)

[Umsetzung 5](#_Toc154850149)

[Erkenntnisse 6](#_Toc154850150)

[Fazit 6](#_Toc154850151)

# Use Case

# Architektur

Gewählte Architektur inkl. evt. Alternativen

Verargumentieren der Entscheidung anhand der gelernten Prinzipien

Ein Bild, das Text, Screenshot, Diagramm, Schrift enthält.

Automatisch generierte Beschreibung

Abbildung 1 Gesamte Architektur

# Umsetzung

In diesem Kapitel wird beschrieben wie die einzelne Services konfiguriert wurden und wie unser Service Stack erstellt wurde.

### IaC

Als Erstes haben wir versucht, Terraform als IaC-Tool zu verwenden. Es hat sich jedoch herausgestellt, dass es Funktionen gibt, die noch nicht unterstützt werden (Bug: [[Bug]: aws\_codebuild\_project queued\_timeout unsupported value when using lambda compute · Issue #34376 · hashicorp/terraform-provider-aws (github.com)](https://github.com/hashicorp/terraform-provider-aws/issues/34376)). Beim CodeBuild-Service haben wir AWS Lambda als "Compute Mode" gewählt. Dieser Modus unterstützt das Attribut "queued\_timeout" nicht. Beim Erstellen der Services wird ein Standardwert festgelegt, obwohl im Terraform-Template nichts definiert ist, was beim Bereitstellen der Services einen Fehler wirft.

Daher haben wir uns entschieden, AWS CloudFormation zu verwenden, um unsere Services zu erstellen. CloudFormation bietet folgende Vorteile:

* Es bietet eine tiefere Integration mit anderen AWS-Services und -Funktionen
* Die Syntax ist deklarativ, was bedeutet, dass Sie Ihre gewünschte Infrastruktur konfigurieren
* CloudFormation bietet eine grafische Benutzeroberfläche namens AWS CloudFormation Designer

### AWS CodePipeline

AWS CodePipeline ist ein Continuous-Delivery-Service, der zur Automatisierung von Software-Release-Prozessen eingesetzt wird. Unsere Pipeline besteht aus 3 Phasen:

* Source
* Build
* Deploy

Source Phase:

In dieser Phase greift der Pipeline-Prozess auf den neuesten Commit über eine Codestar-Verbindung, zieht die relevanten Dateien und speichert sie in die Artefakten-Bucket.

Build Phase:

Die Build-Phase hollt sich die Artefakte aus dem Artefakten-Bucket und buildet das Projekt gemäss den in der Buildspec-Datei im Projekt-Repository spezifizierten Anweisungen. Dieser Prozess resultiert in der Erstellung eines Release-Dist-Verzeichnisses. Das generierte Release wird als ZIP-Datei im Artefakten-Bucket abgelegt.

Deploy Phase:

In der Deployment-Phase wird das erstellte Release-ZIP aus dem Artefakten-Bucket abgerufen. Dieses ZIP wird entpackt, und die enthaltenen Dateien werden als statische Website im Website-Bucket bereitgestellt. Die Webseite ist unter folgender URL erreichbar: <http://pipeline-stack-websitebucket-q4j5omdxwwn5.s3-website-eu-west-1.amazonaws.com>

Durch diese Release Pipeline wird sichergestellt, dass jede Änderung im Quellcode erfolgreich gebaut, veröffentlicht und bereitgestellt wird, wodurch ein effizienter Development- und Delivery-Prozess gewährleistet wird.

### AWS S3

In unserem CodePipeline Projekt wurden folgende 3 S3 Buckets erstellt und verwendet:

* WebsiteBucket: ein öffenlicher Bucket für das deployen einer statischen Webseite: <http://pipeline-stack-websitebucket-q4j5omdxwwn5.s3-website-eu-west-1.amazonaws.com>
* ArtifactBucket: der Bucket wird verwendet um alle Source und Build Artifakte zu speichern.
* LambdaSourceBucket: in diesem Bucket ist der Quellcode der Lambda Funktion angelegt.

### AWS CodeBuild

AWS CodeBuild wurde für die Build Phase verwendet. Für das CodeBuild Project haben wir statt einer EC2 Instanz AWS Lambda für das Ausführen unsere Builds verwendet. AWS Lambda ermöglicht schnellere Builds aufgrund einer geringeren Startlatenz. AWS Lambda skaliert auch automatisch, sodass Builds nicht in der Warteschlange warten, bis sie ausgeführt werden.

Der Build wird durch die Buildspec Datei im Angular definiert:

Buildspe.yaml:

A screen shot of a computer program

Description automatically generated

### AWS Lambda

### Amazon EventBridge

# Erkenntnisse

# Fazit

# Anhang