Dokumentation M346

# Bildverkleinerung

Ein Bild, das Text, Person, schwarz enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Bildschirm enthält.

Automatisch generierte Beschreibung

Inhaltsverzeichnis

[Bildverkleinerung 1](#_Toc122730904)

[Einleitung 2](#_Toc122730905)

[Ablauf / Planung 3](#_Toc122730906)

[Code erklärt 4](#_Toc122730907)

[Reflexion 5](#_Toc122730908)

# Einleitung

Wir haben uns für das Projekt „Bildverkleinerung“ entschieden. Bei der Bildverkleinerung muss man die Fertigkeiten mit der Cloud praktisch aufzeigen. Wir haben in der Gruppe, welche aus Dean, Julia und Jan besteht, versucht diese Aufgabe zu lösen. In dieser Aufgabe mussten wir zwei Buckets auf Amazon S3 einrichten und einen Code dazu schreiben, welcher automatisch ein Bild, welches in den Bucket hochgeladen wurde über die Lambdafunktion verkleinert und dieses in einen zweiten Bucket abspeichert.

# Ablauf / Planung

Wir haben es mehrmals mit anderen Wegen probiert und zuerst mit dem Versuch die Kreierung der Buckets und den Cleanup dafür zu schreiben. Beim zweiten Commit haben wir die Kreierung und das Zippen der leeren JS-Datei für die Lambda-Funktion erstellt. Mit unserem dritten Commit haben wir der Lambda-Funktion eine Konfigurationsdatei hinzugefügt mit einer Limitierung, sodass die Datei nur auf .jpg enden darf. Auch haben wir eingefügt, dass das Ergebnis des Uploads in die Buckets angezeigt werden kann. Wir haben versucht die Lambda-Konfiguration einzusetzen, jedoch hat dies nicht funktioniert. Selbst mit einiger Fehlersuche nach dem Error: 'Unable to validate the following destination configurations'. Wir haben versucht eine Json-Datei zu erstellen, damit man die Funktion abspeichern kann und um anzusehen, was dann passiert. Wir haben die cleanup.sh neu geschrieben um die output.json und die Lambda-Funktion zu löschen.

Im zweiten Versuch haben wir im ersten Commit in Zusammenarbeit mit der ganzen Klasse versucht weiterzukommen, da fast alle an der 'put-bucket-notification-configuration' gescheitert sind, jedoch sind wir nicht weitergekommen. Die Separation der ARN's und der Namen haben wir als nächstes geschrieben. Im zweiten Commit haben wir uns für einen neuen Python-Skript anstelle vom JS-Skript entschieden und diesen programmiert.

# Code erklärt

Wir haben eine Datei erstellt „\_setup.sh“, in welcher wir zuerst den Folder „workspace“ aufrufen und danach die Namen der Funktionen instanziieren. Danach gibt es einen Text aus und erstellt die Buckets. Im nächsten Schritt lädt es die gezippte Datei „func.py“ hoch und konfiguriert die Rückmeldung. Mit der Datei „\_cleanup.sh“ werden die Buckets gelöscht und wieder zurückgesetzt. Wir haben den Code mit einem Template im .json Format erstellt. Mit der Datei „func.py“ haben wir die Dateien importiert und den S3-Client ausgewählt. In der Funktion lambda\_handler haben wir ein Bucket-Event erstellt und den Bucket und den Key ausgegeben. Das Bild wird dann mit dem lambda\_handler geöffnet und die Grösse verändert. Das Bild wird dann als ein neues File in einem neuen Bucket gespeichert. In der Datei func.js wird ein Event ausgelöst, in welchem der Inhalt der Json-Datei in einen String umgewandelt wird.

# Reflexion