Project 2 (Part 1): PaaS

Summary

In this project, a cloud application will be developed using Platform as a Service (PaaS) resources. Specifically, the application will be built with AWS Lambda and other supporting services from AWS. AWS Lambda is recognized as the first and most widely used function-based serverless computing service. With more experience in cloud programming, a more advanced application will be created compared to Project 1, taking advantage of the ease that PaaS offers for cloud development.

The PaaS application will provide face recognition as a service for video streams from clients (e.g., security cameras). This type of cloud service is valuable to many users, and the technologies and techniques learned in this project will be applicable to building similar services in the future.

Description

The project is divided into two parts. In the first part, we will focus on implementing the video-splitting function. The clients upload videos to the Input bucket. When a new video is uploaded, it triggers a video-splitting function that splits the video into frames and stores the Group-of-Pictures (GoP) in another bucket called Stage-1. The second part of the project will include the development of the other Lambda functions and S3 buckets needed for this application.

The architecture of the cloud application in Part 1 is shown in Figure 1.

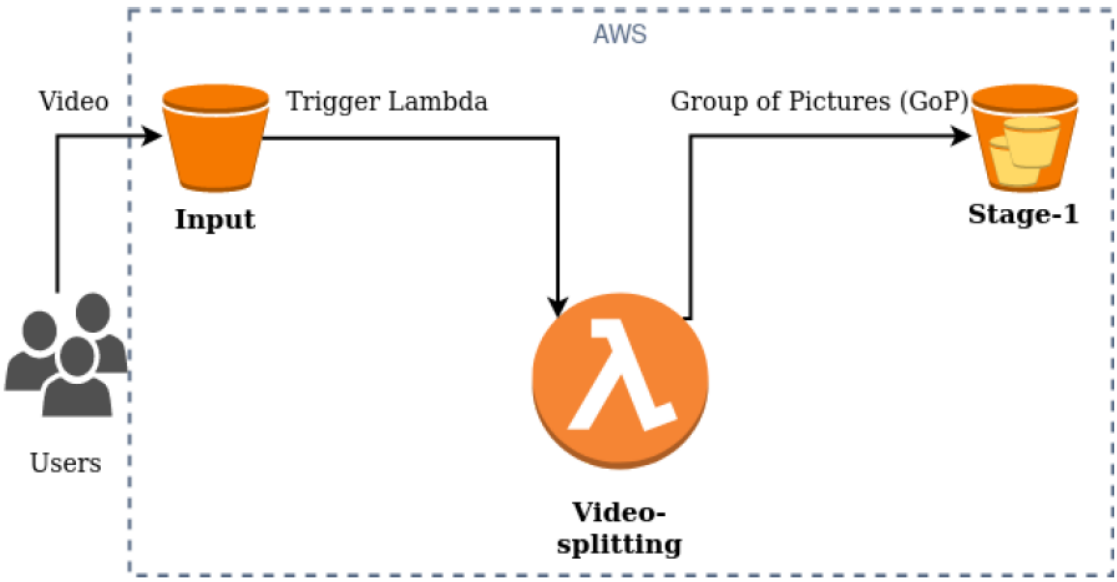

Fig 1: Architecture diagram of Project 2 (Part 1)

Input Bucket

- This bucket stores the videos uploaded by the workload generator.
- The name of the input bucket MUST be **<ID>-input**. Each student submission MUST have only one input bucket with the exact naming. For example, if your ID is "12345678910", your bucket name will be 12345678910-input.
- The input bucket must contain only .mp4 video files sent by the workload generator.
- (https://github.com/visa-lab/CSE546-Cloud-Computing/tree/main/Project_2/dataset/test_case_2)
- Each new video upload should trigger an invocation of the **video-splitting** Lambda function.

The video-splitting Function

- The name of this Lambda function MUST be "**video-splitting**".
- The function is triggered whenever a new video is uploaded to the "**<ID>-input**" bucket.
- The function gets the video file and splits the video in Group-of-Pictures (GoP) using the ffmpeg library.

The function splits the given input in frames using the following command:

```
ffmpeg -ss 0 -r 1 -i <input_video>.mp4 -vf fps=1/10 -start_number 0
-vframes 10 /tmp/test_0/output-%02d.jpg -y
```

- The function stores this group of pictures (GoP) in a single folder with the same name as the input video in the "**<ID>-stage-1**" bucket. E.g. If the <input_video> is test_00.mp4, then its corresponding group of pictures (GoP) is stored in a folder named test_00 in the "**<ID>-stage-1**" bucket.
- You can use the provided video-splitting code to implement your Lambda function.
- You can also use the provided templates for the Dockerfile and handler code. You need to add the required code/packages in the template code to run and compile the video-splitting function.

Stage-1 bucket

- This bucket stores the result of the video-splitting Lambda function.
- The name of the input bucket MUST be **< ID>-stage-1**. Each student submission MUST have only one "**<ID>-stage-1**" bucket with the exact naming.
- The <ID>-stage-1 bucket should have a folder with the exact name as the input video name without any extension. E.g. If the input bucket has 5 videos with the names test_0.mp4, test_1.mp4, test_2.mp4, test_3.mp4, test_4.mp4, then the stage-1 bucket should have 5 folders with the names exactly as the input video files: test-0, test_1, test_2, test_3, test_4.
- Each folder in <ID>-stage-1 contains images with the extension .jpg. It follows the naming convention "**output_xx.jpg**", where xx is the frame number.