Group number 26
Group members:
Shiyu Xiu, 1004724872
Yuangan Zou, 1004761778
Qian Tang, 1005228656
Assignment 3

Description of Application

Our application is an online image storage system. This system allows users to upload their images for storage in order to save local storage space. It also notifies the user by email through Amazon SES when a user exceeds the storage limit. The user is able to download the image he uploads, view the images online as well as organize the images he uploads by grouping them into albums through Amazon Rekognition. There is also a cache assigned to the user in order to speed up the process of retrieving images. The user is able to configure the cache size as well as the replacement policy. Our online storage system ensures that each user can only view and manage the images in his own account after logging in by applying Amazon Cognito.

User Instructions

1. Log into the system: if you are new to our system, please sign up with your email in order to obtain a new account.

2. After logging in, you can upload your images through the “Upload” function. You are able to view the current available space before uploading. If you exceed the available space, the newly uploaded file will be automatically deleted and you will receive a notification email from us later. By default, each user will have 10MB of free storage.

3. For a successful upload, you will have to assign a key to the image. Please note that if you enter a used key, the image associated with the key will be replaced by the newly uploaded one. Our system accepts keys in both string and integer format. You can check all the keys you have stored in our system by going to “ShowAllKeys”.

4. To retrieve an image you uploaded, please go to “Show Image” and enter the key associated with the image.

5. In order to delete the images you uploaded, go to “Show All Keys” and you can delete them by clicking on the button.

6. Our system will automatically group our images into albums using Amazon Rekognition[1], you can view the grouped albums in “ShowImageByLabel”.

7. If you would like to configure the cache, go to “ConfigMem” to enter a capacity in MB and choose a replacement policy between Least Recently Used (LRU) and Random. Note that this step is not mandatory for using our system. The default setting for the cache is capacity = 5MB and replacement policy = random.

Architecture of Application

The following graph contains the general architecture of our application:

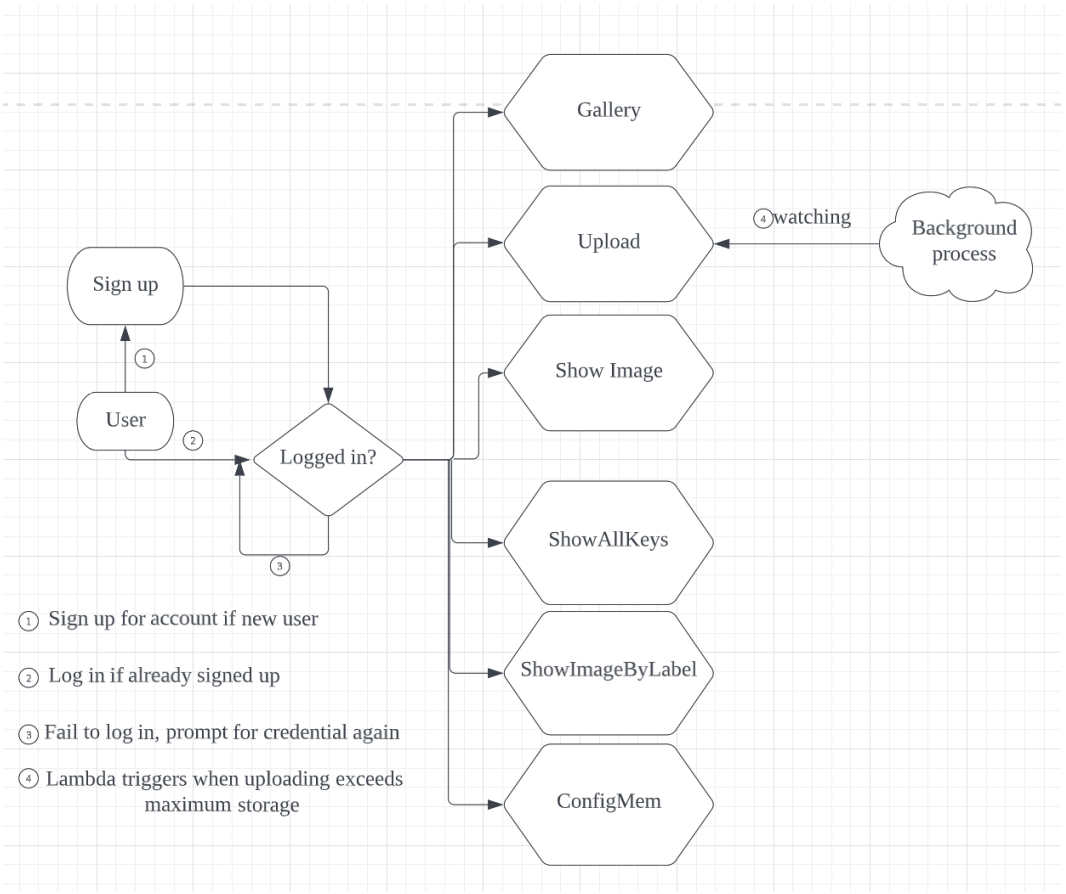


Figure1: General structure of application

The login module at the front will check if the user has provided correct credentials. After a user is logged in, he will have access to the 6 functions that our application provides. The functions are defined as:

- Gallery: A page that shows all the images that the user has uploaded. The user is able to choose a suitable image grid size to ultimate the viewing experience.

- Upload: A page that allows a user to upload an image with a key specified, system also shows the remaining space available.

- ShowImage: A page that shows the image with a specific key.

- ShowAllKeys: A page that shows all keys uploaded, together with the file name, file size as well as date modified. This page also allows the user to delete an uploaded image.

- ShowImageByLabel: This page groups the uploaded images into albums, the names of albums are automatically generated by Amazon Recoknition.

- ConfigMem: A page for the user to configure memcache settings. Specifically, the user can set cache capacity, replacement policy as well as clear the contents in cache.
- Background process: The background process is implemented by Lambda and it watches the Upload function constantly. It is triggered when the user has exceeded the maximum storage space. In such case, the background process will delete the newly uploaded image and send the user an email notification using Amazon SES.

AWS Cost Model

We predict the cost for running our application by estimating the cost for AWS services required. Below is a summary of the services we used as well as their costs.[2]

AWS service	Cost per month
S3	$0.023/GB for first 50TB, $0.022/GB for next 450 TB, $0.021/GB for over 500TB, $0.0004/1000 requests
dynamoDB	$1.25/million write requests, $0.25/million read requests, $0.25/GB after 25GB
Simple Email Service	$0.1/1000 emails
Lambda/EventBridge	Depends on the memory size of lambda function, please see [3]
Amazon Cognito	$0.0055/user after 50000, $0.0046/user for next 900,000, $0.00325/user for next 9,000,000, $0.0025/user for greater than 10,000,000
Amazon Rekognition	$0.001/img for first 1M, $0.0008/img for next 4M, $0.0006/img for next 30M, $0.0004/img for over 35M
API gateway	$1/Million requests for the first 300M, $0.9/Million requests for over 300M

Table1: AWS service cost table

In order to calculate the cost, we make the following assumptions:

1. Each user stores 10MB of images, each image is on average 0.5MB, so each user will have 20 images stored.

2. All users will upload 20 images in sequence until the storage is full, they will not delete any of the stored images and they will only try to exceed the maximum storage limit once by trying to upload the 21st image.

3. Each user will try to retrieve each of the 20 stored images once, view the gallery once and view each album once.

4. All users will use the default settings for memcache. (capacity = 5MB, replacement policy = random)

With the above assumption, we know that for one user:
#requests for S3 = 20 (upload) + 20 (check repeat key) + 20 (retrieves) + 20 (retrieves for gallery) + 20 (retrieves for albums)+1 (try to add 21st image) + 1 (delete 21st image due to exceeding storage) = 102 requests
S3 storage = 10MB
dynamodB storage = 32kB based on testing
#requests for dynamoDB = 20(check existing) + 20(upload) + 40(retrieves for gallery) +40 (retrieves for album) = 120 requests
#email = 1
rekognition = 21(upload) + 20(retrieve) + 20(albums) = 61 images
API gateway = 102 (s3) + 120 (dynamodB) + 61(rekognito) + 20 (signup&login) = 303 requests

Assume that we have 10 users, then the cost after 6 months would be:

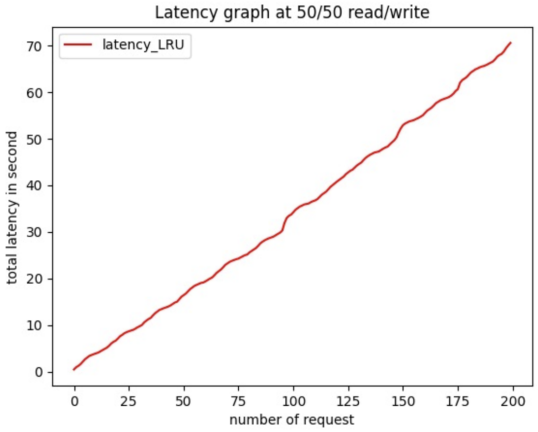
Cost type	Cost Amount in CAD
S3	$0.0004
dynamoDB	$0.0005
Simple Email Service	$0.001
Lambda/EventBridge	$33.13
Amazon Cognito	$0
Amazon Rekognition	$0.61
API gateway	$0.003

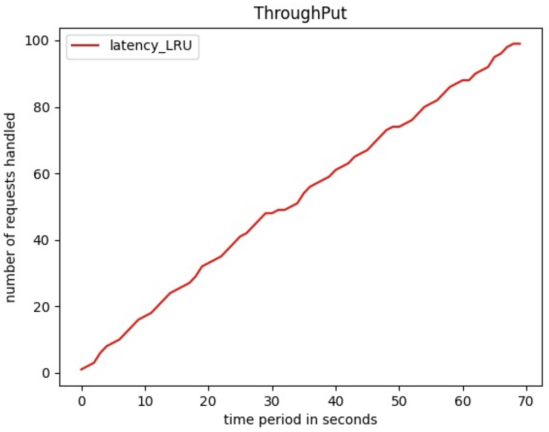
We observe that the Total cost = $33.74. Similarly, if we have 1000 users, the total cost would be $2691.98 and if we have 1,000,000 users, the cost would rise to $2,607,113.92.

Performance

We measure the latency and throughput for our application as follows.

The experimental set up is: memcache Read-write ratio 50:50, memcache size = 5MB and replacement policy = LRU.





We see that the latency of our application changes approximately linearly with the number of requests at an read/write ratio of 50:50. We tested this by selecting 10 images, first upload them and then try to retrieve each of them once. Since the first retrieve will go to S3 for obtaining the image, then the latency separates uniformly across different retrieves. If we increase the read/write ratio by doing a second retrieve from memcache instead of S3, then we expect that the slope of the latency graph to decrease, making it more flat. This is because retrieving from memcache is much faster, therefore introducing less latency.

We also observe that under the replacement policy LRU, the throughput increases linearly with time. This is expected since we are having 50:50 read/write rate which means that the throughput is also distributed uniformly across multiple upload/retrieve operations.

Reference

[1] A. Mishra, “Machine learning in the AWS cloud: Add intelligence to applications with Amazon Sagemaker and Amazon Rekognition,” *Amazon*, 2019. [Online]. Available: https://docs.aws.amazon.com/rekognition/latest/dg/what-is.html. [Accessed: 17-Dec-2022].

[2]S. Usher, “Free,” *Amazon*, 2021. [Online]. Available: https://aws.amazon.com/free/?trk=c8882cbf-4c23-4e67-b098-09697e14ffd9&sc_channel=ps&s_kwcid=AL%214422%213%21453053794317%21e%21%21g%21%21aws+pricing&ef_id=CjwKCAiA7vWcBhBUEiwAXieItnOEuHdP9bcikn2wI3DaGfPIS6EoN6APVLbY-aL-EryTa_kNbyjJXhoCoS8QAvD_BwE%3AG%3As&all-free-tier.sort-by=item.additionalFields.SortRank&all-free-tier.sort-order=asc&awsf.Free+Tier+Types=%2Aall&awsf.Free+Tier+Categories=%2Aall. [Accessed: 18-Dec-2022].

[3]D. Musgrave, “Lambda,” *Amazon*, 2022. [Online]. Available: https://aws.amazon.com/lambda/pricing/. [Accessed: 18-Dec-2022].