SAFEDEPOSIT PROJECT DESIGN DOCUMENT

CSCI5410 – Serverless Data Processing

Dhrumil Amish Shah (B00857606)
Dhrumil Rakesh Shah (B00870600)
Sanket Ushangbhai Mehta (B00881783)

Instructor – Saurabh Dey

SafeDeposit Application Architecture

Figure 1 displays the architecture of the SafeDeposit application. The application consists of a frontend that will be built using the React.js framework and a backend built using Express.js, Node.js along with various Amazon Web Services [1] (AWS) and Google Cloud Platform [2] (GCP) Services.

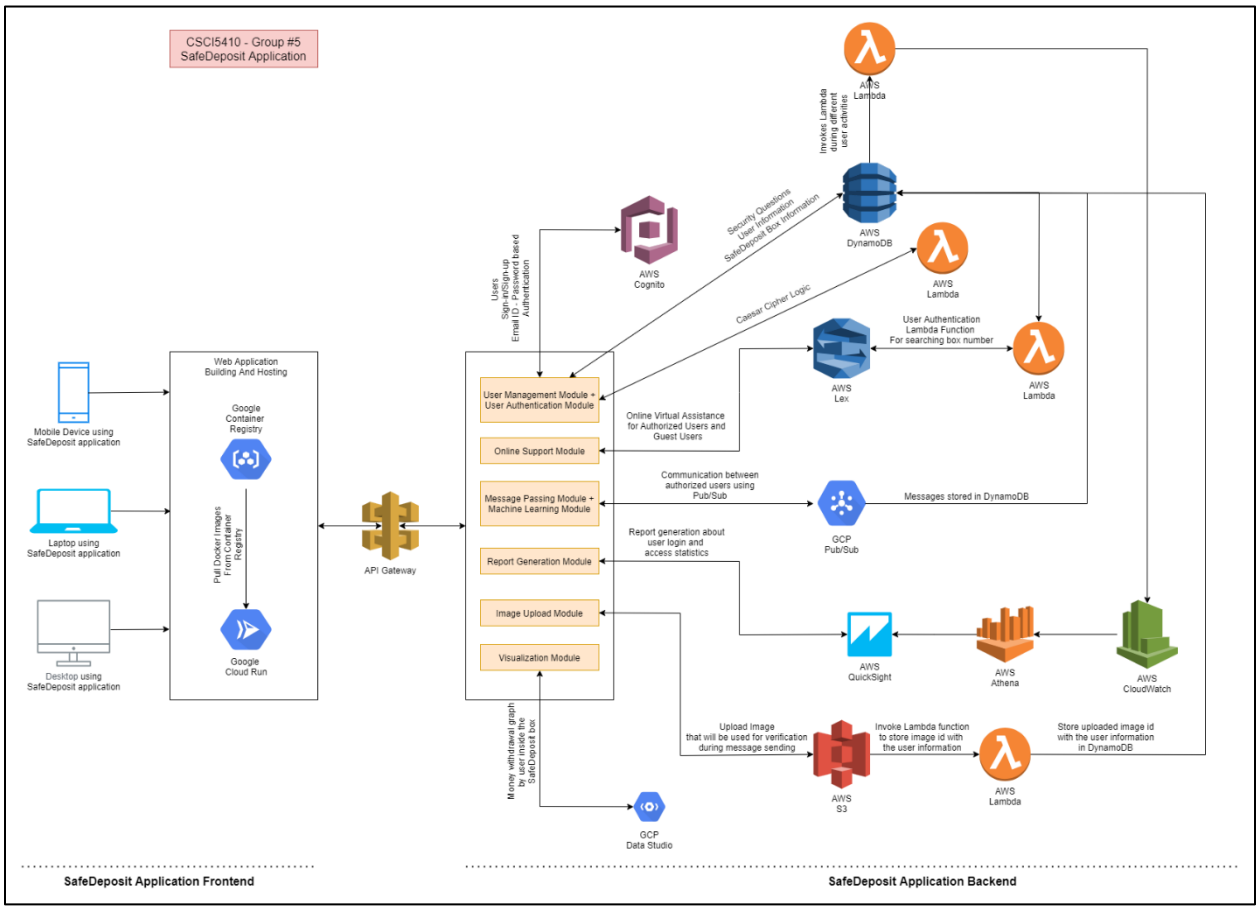

Figure 1 - SafeDeposit Application Architecture [3]

SafeDeposit Application Architecture Overview

The SafeDeposit application is a multi-cloud based serverless security deposit box that consists of a frontend built using React.js framework and a backend built using Express.js, Node.js along with various cloud-based services from AWS and GCP. The final application will be deployed to the Google Cloud Run (GCR). This application provides numerous features to the authorized users such as money withdrawal from the safe deposit, message sending to other authorized users, safe deposit box number searching based on a clue provided. Also, it provides a few features to the guest users such as virtual assistance for navigation. The final application will contain the below modules and will be built using the following services:

1. **User Management Module:**
    a. User information (Sign-up details) management & storing – AWS DynamoDB [4]

2. **User Authentication Module:**
    a. User-ID and Password – AWS Cognito [5]
    b. Questions and Answers – AWS DynamoDB + AWS Lambda [6]
    c. Caesar Cipher – AWS Lambda

3. **Online Support Module:**
    a. Online Virtual Assistance – AWS Lex [7]

4. **Message Passing Module:**
    a. Communication between authorized users – GCP Pub/Sub + AWS S3 [8]

5. **Machine Learning:**
    a. Image Classification Algorithm – GCP built-in image classification algorithm (GCP AutoML Vision)

6. **Web Application Building and Hosting:**
    a. Web Application Registry – Google Container Registry [9]
    b. Web Application Hosting – Google Cloud Run [10]

7. **Other Essential Modules:**
    a. Testing Module (Validations, Unit Tests) – AWS Lambda
    b. Report Generation Module – AWS CloudWatch [11] + AWS Athena [12] + AWS QuickSight [13]
    c. Visualization Module – Google Data Studio [14] (Google Cloud Studio)

The architecture is explained in detail in the given section:

1. **User Management Module:**
    a. This module mainly covers the User Registration part of the application along with dynamically assigning the User to a SafeDeposit Box and also securely storing and maintaining the user details. The application initially will display the home page where the user will either login to the application or can register themselves.
    b. If the user chooses to register themselves then the application will ask user for details like Full Name, Email ID (It will be a unique User ID), Password and Three Predefined Security Questions. There will be on-the go validations done for the details entered by the user like for a valid email or for a strong password.
    c. **SafeDeposit Box Logic:**
        i. So, initially at the very beginning when there are no users registered within the application there are Zero SafeDeposit boxes created or available. When first user registers with the application, the system checks if there are any

SafeDeposit boxes already created if no, then the very first SafeDeposit Box is created, and the user gets added to that SafeDeposit Box.

ii. As each SafeDeposit Box can contain up-to 3 users, so for the next two users that registers with the application will be added to the first SafeDeposit Box that was created.

iii. Now, if a fourth User registers with the application, then the application checks if there is any SafeDeposit Box available to add a new User, if no then a new SafeDeposit Box gets created and respectively an entry is made in the AWS DynamoDB database.

iv. To achieve this, we will be using in-code checks using AWS DynamoDB and AWS Lambda function to dynamically add User to respective SafeDeposit Boxes and create new SafeDeposit Boxes.

d. Now, all the information entered by the User, all the Security Questions and also the SafeDeposit Box Information is stored securely in the AWS DynamoDB.

e. Another thing that the user needs to do is upload an image that will be stored against that User and will be used as verification for the Message Passing between authorized users. The details of this are covered in the Image Upload Module.

2. User Authentication Module:

a. In this module the Registered Users are authenticated by using a three-factor authentication module. The three-factor authentication includes the ID-Password, Security Question and Caesar Cipher.

b. We are using AWS Cognito in our application for Email ID – Password authentication, whereas for Security Question authentication we are using AWS DynamoDB that has each User's Security Question and Answers stored. For Caesar Cipher we are using an AWS Lambda Function.

i. For Caesar Cipher we will provide user with the Caesar Cipher and the Key, the user will be required to decode the Caesar Cipher using the Key and enter the decoded message.

3. Online Support Module:

a. The Online Support Module is basically an Online Virtual Assistant that both Authorized and Guest Users can use. As, we are using React.js for frontend of our application, we can seamlessly integrate the AWS Lex and React.js.

b. For Online Virtual Assistant we are using AWS Lex along with AWS Lambda Function. The flow is as follows;

i. The user either a guest or an authorized user interacts with the Virtual Assistant, if the user is a guest, then he/she will only be shown navigation and normal application features.

ii. If the user is an authorized user, then he/she will be asked for authentication first by invoking an AWS Lambda function which will communicate with the AWS DynamoDB that will have the user's details stored.

1. If, the authentication is successful then the user can access various features via the Virtual Assistant like Searching his/her SafeDeposit Box number, User details and much more.

4. **Message Passing Module + Machine Learning Module:**
    a. In the Message Passing Module the main service that is being used is the GCP Pub/Sub service that enables the authorized users to send and receive messages to other authorized users using the application. These messages are stored in the AWS DynamoDB Database.
    b. For the Machine Learning Module, we are going to use the GCP built-in image classification algorithm which is the GCP AutoML Vision service that will help us in validating and comparing the images and then allowing the user to send message.

5. **Report Generation Module:**
    a. This module generates the report on user login and access statistics. User can generate report, which uses AWS QuickSight service- i.e., a business intelligence tool. AWS CloudWatch service would log each of the user activities and that information will be provided to another AWS service Athena which then would process this information and give it to QuickSight for report generation.

6. **Image Upload Module:**
    a. When user wants to send a message to another authorized user, he/she has to upload an image for sending the message and that image needs to be verified with the recipient's image in order to successfully send the message and the receiver can receive the message. The image would be stored into storage service provided by AWS S3. Whenever user sends a message lambda function is triggered to store the image id and verify it with the user information stored during registration into the DynamoDB. If the image match is successful then it is verified and user can send message successfully.
    b. This is an add-on module where user can upload an image in order to update their original uploaded image while registration. User will get an option to upload an image in the user interface which will take them to another dialog box or page to upload the image. This image will be stored into AWS S3 and then it will be updated in the DynamoDB for the user id associated with the user and original image will be replaced. A lambda function will be triggered when user uploads an image which does the storing of the image in DynamoDB.

7. **Testing Module:**
    a. We will be creating the validations and unit tests for our application. There will be various test cases to test the application such as test cases to test the image verification, to test if authorized user can perform operations restricted for them, to test if guest user can perform operations which are not restricted to them, to test whether user is able to send messages with or without uploading correct image, to test if the safe deposit balance is accurate after user performs transaction on it, to test if user is able to get the unique box numbers while registration and many other tests according to the functionalities. Testing will include all the possible test cases required by the application to run efficiently.

8. **Visualization Module:**
    a. User can generate a visualization for the money withdrawal in the form of graph using this module. This module makes use of Google Data Studio (Google Cloud Studio) to generate the graph for the data visualization. Whatever the transactions are made by the user, it will be shown in the graph generated by this service offered by Google.

Meeting Logs

Figure 2 displays the meeting log of Architecture Design Discussion.

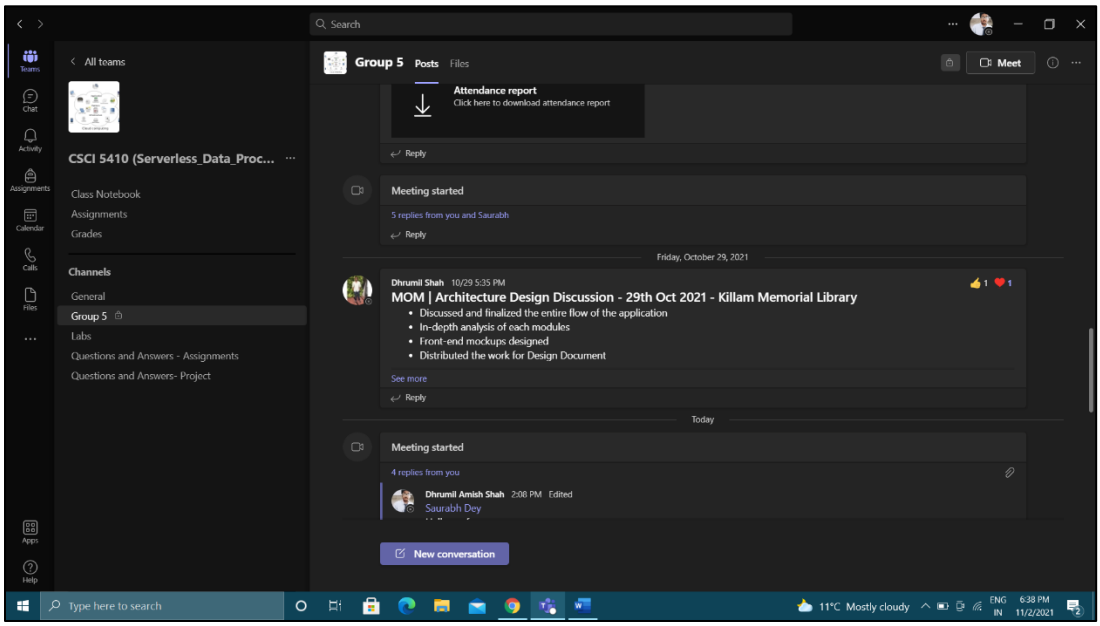

Figure 2 - Meeting regarding Architecture Design Discussion at Killam Memorial Library on 29th Oct 2021

Figure 3 displays the meeting log of Architecture Finalization.

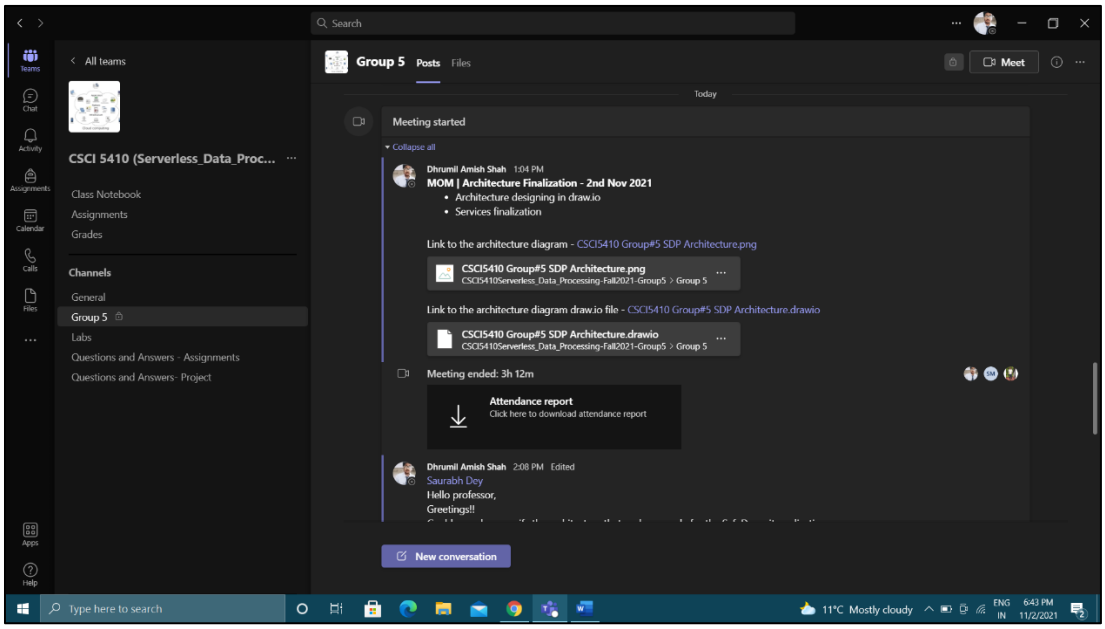

Figure 3 - Meeting regarding Architecture Finalization done Online on 2nd Nov 2021

References:

[1] Amazon and AWS, "Cloud Services - Amazon Web Services (AWS)," Amazon, [Online]. Available: https://aws.amazon.com/. [Accessed 02 November 2021].

[2] Google, "Google Cloud Platform," Google, [Online]. Available: https://console.cloud.google.com/. [Accessed 02 November 2021].

[3] draw.io, "Flowchart Maker & Online Diagram Software," draw.io, [Online]. Available: https://app.diagrams.net/. [Accessed 02 November 2021].

[4] AWS, "Amazon DynamoDB," Amazon, [Online]. Available: https://aws.amazon.com/dynamodb/. [Accessed 02 November 2021].

[5] AWS, "AWS Cognito," Amazon, [Online]. Available: https://aws.amazon.com/cognito/. [Accessed 02 November 2021].

[6] AWS, "AWS Lambda," Amazon, [Online]. Available: https://aws.amazon.com/lambda/. [Accessed 02 November 2021].

[7] AWS, "Amazon Lex," Amazon, [Online]. Available: https://aws.amazon.com/lex/. [Accessed 02 November 2021].

[8] AWS, "Amazon S3," Amazon, [Online]. Available: https://aws.amazon.com/s3/. [Accessed 02 November 2021].

[9] Google, "Container Registry," Google, [Online]. Available: https://cloud.google.com/container-registry. [Accessed 02 November 2021].

[10] Google, "Cloud Run," Google, [Online]. Available: https://cloud.google.com/run. [Accessed 02 November 2021].

[11] AWS, "Amazon CloudWatch," Amazon, [Online]. Available: https://aws.amazon.com/cloudwatch/. [Accessed 02 November 2021].

[12] AWS, "Amazon Athena," Amazon, [Online]. Available: https://aws.amazon.com/athena/. [Accessed 02 November 2021].

[13] AWS, "Amazon QuickSight," Amazon, [Online]. Available: https://aws.amazon.com/quicksight/. [Accessed 02 November 2021].

[14] Google, "Data Studio," Google, [Online]. Available: https://datastudio.google.com/. [Accessed 02 November 2021].