DALHOUSIE UNIVERSITY

FACULTY OF COMPUTER SCIENCE

CSCI 5410 Serverless Data Processing

Project Conception Report

Group 4

Submitted by:

Adarsh Kannan (B00900913)
Fenil Parmar (B00895684)
Jenish Patel (B00897765)
Kunj Patel (B00894376)
Meet Patel (B00899517)
Ruhi Rajnish Tyagi (B00872269)

Gitlab: https://git.cs.dal.ca/kvpatel/csci5410_group4

Contents

Table of Figures

1. Project Overview:

"Serverless is changing the future software development model and process and is the future of cloud computing." [1] Serverless applications have a plethora of benefits when it comes to business and software development expenses. Enterprises employ fully managed cloud services in a Serverless architecture, which is comparable to developing applications using upper-layer APIs. Different components and services are integrated by cloud companies. Enterprises can afford to make low-cost technological decisions and implementations, focusing on how to adapt application logic to cloud service functions. They also have a low Operations and Management (O&M) workload and deliver value hourly [1].

The main goal of this project is to create a cloud plumbing system in which an application will be built utilizing serverless technologies to process data (mainly on-demand data) that can be used by multiple clients. To develop the system, we'll employ a variety of back-end services and a simple front-end application.

The primary services of the application are:

- *Customers:* This service oversees registering/logging in customers. In addition, the service communicates with other services to register feedback, arrange tours, and place food orders, among other things.
- *Hotel Management:* This service oversees registering/logging in customers, reserving rooms or beds, interacting with *Kitchen* to acknowledge its service, receiving a report on consumer comments, and making changes to a service.
- *Kitchen:* This service oversees receiving customer orders, sending invoices to hotel management, and preparing meals (only breakfast). Meal preparation necessitates access to inventory, cooking, scheduling, and order management, among other things.
- *Tour Operator:* This is a service that always listens to customer requests and creates custom tour services. It creates tickets/passes and sends them to customers via email.

DALSoft5410 has chosen a serverless application to reduce project development and operating costs. They must maintain and configure the backend service in a server-oriented architecture, which they are unable to do due to resource constraints if they choose server-oriented application. As a result, serverless is the only viable option. They've got two types of AWS accounts and one GCP account, which they can use to develop, test, and deploy their application employing an agile methodology.

2. Core Components

User Management Module:

The main functionality of this module is to handle user registration process. The AWS Cognito will be used for the authentication, authorization, and user management. The users will be able to sign in using their username and password. User pool and Identity pool will be used together to provide the essential services for the application. Moreover, AWS DynamoDB service will be utilized to store the information of these users.

Authentication Module

This module is responsible for authentication. We are planning to use AWS Cognito. The module will utilize three ways such as ID-Password, Security Questions & Answers and d Caesar Cipher for the authentication. The security questions & answers are stored in the DynamoDB. Additionally, the CloudFunction service of Google cloud platform will be used to cipher and decipher the plaintext with Caesar Cipher algorithm.

Online Support Module

The module is responsible to provide online assistance to the authenticated and guest users. For this module, AWS Lex service will be used. Various scenarios of user interactions with the application will be fed to the service to provide the accurate response to assist the users in possible best manner.

Message Passing Module:

For the message passing module, the GCP Pub/Sub service will be used that enables the authorized users to communicate with the Hotel Management and Tour Operators.

Machine Learning Module:

For the machine learning module, we are going to use the one of the built-in algorithms of GCP called AutoML Tabular to identify the similarities of staying duration of the customers and the polarity of their feedback to add an appropriate score.

Web Application Building and Hosting:

We are going to use ReactJS for the front-end development and NodeJS with Express framework for the API connection as a back-end environment. Users will interact with the web application via front-end module. The application will be hosted on AWS Elastic Beanstalk service as it provides real time load balancing by auto scaling.

Other Essential Modules: Report Generation/Visualization:

To generate the user login history and access statistics reports, we are planning to use the AWS QuickSight service which is equipped with Business Intelligence dashboards with the applications. Moreover, the embedded ML plugin can be used to view and analyze the generated reports.

Visualization of various data such as customer bookings, food orders, profit or income charts etc, will be provided by the service called Google Data Studio as it is convenient to report on data from a wide variety of sources, without programing.

3. Class Diagram

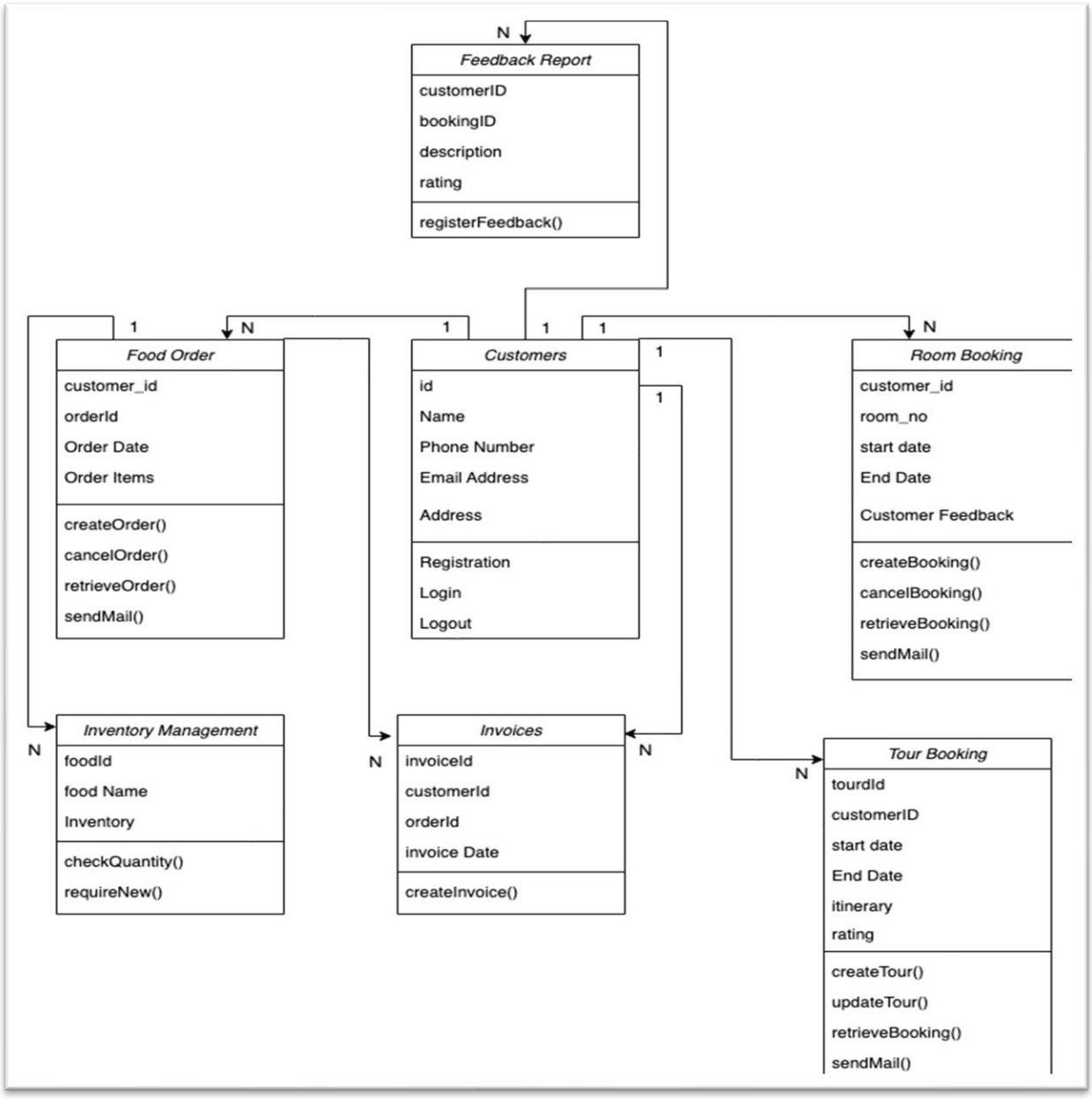

Figure 1 Class diagram of the application [3]

4. Cloud Architecture

The below image depicts the cloud architecture of the application. The diagram is designed using draw.io [3] by gathering knowledge from sources Google Cloud Documentation [4], and AWS Documentation [5]. As we can see in the image, the users will interact with the front-end of the application developed in ReactJS. The front-end of the application is hosted on AWS Elastic Beanstalk for better scalability. AWS API gateway is used to navigate the requests to various services such as AWS Lambda, AWS LEX, Cognito, DynamoBD, Pub/Sub, AutoML, Data Studio and AWS QuickSight.

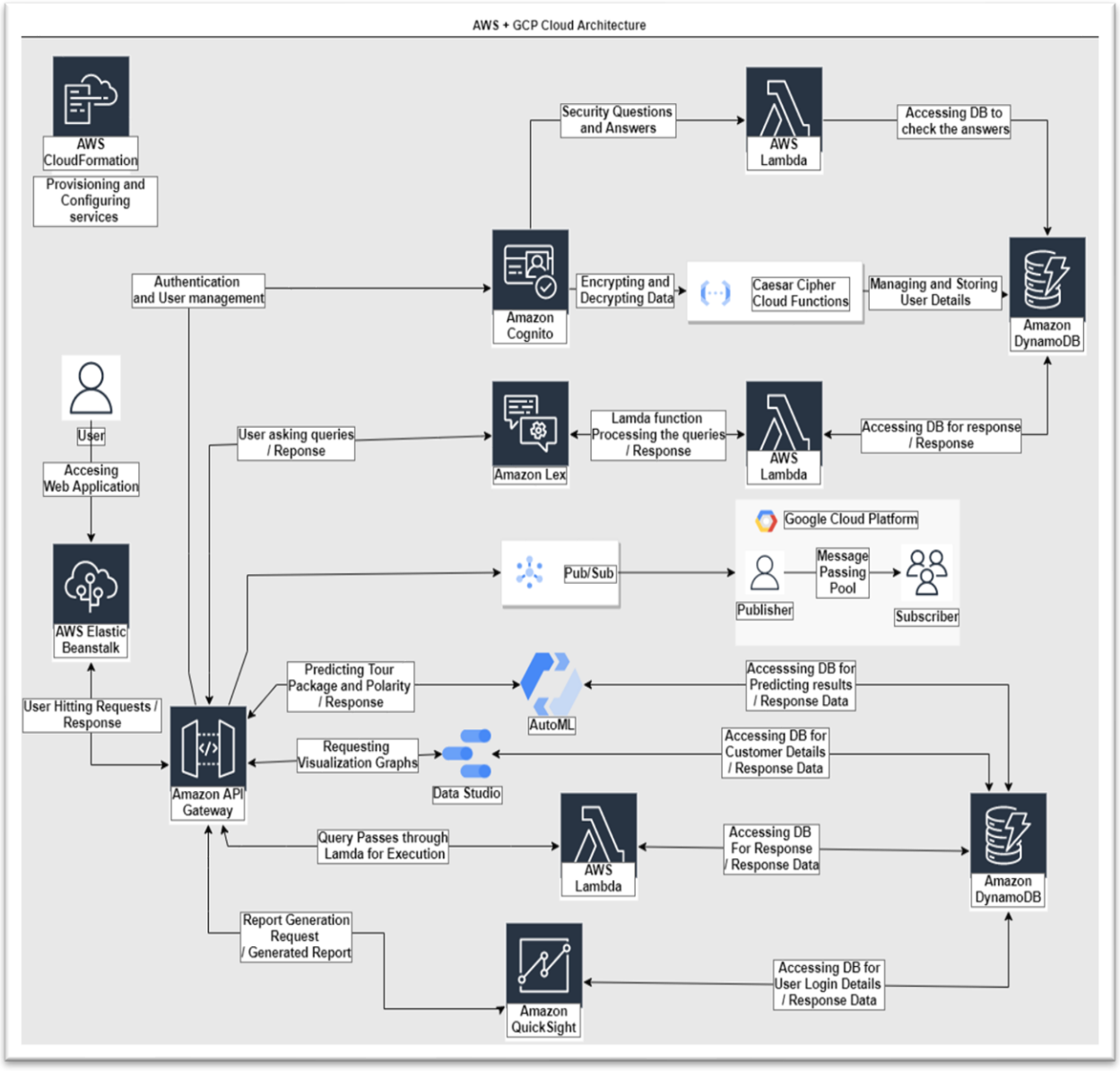

Figure 2 Cloud Architecture of the Application [3], [4], [5]

5. Challenges

Typical challenges faced when building a serverless application:

NOW:

- While Serverless platform providers are constantly increasing the performance of their underlying infrastructure, latency will always be an issue due to the highly distributed, loosely linked nature of Serverless application [2].
- Going Serverless entails losing complete control over the software stack that your code runs on. The lack of complete configuration control is another limitation of Serverless. For example, there are only a few configuration parameters accessible in the AWS Lambda FaaS platform, and no control over JVM or operating system runtime parameters [2].
- It can be difficult to thoroughly test a complicated Serverless application without creating a separate account with the platform provider, to guarantee that testing does not interfere with production resources and that testing does not exceed account-wide platform limits [2].
- On top of all of this, most of us are new to cloud technologies like AWS right now, and the learning curve for a beginner is steep and may be exhausting at times. However, as a group, we continue to communicate with one another and address our concerns.

THEN:

- Other than the above problems, all of them can be labeled “unknown” since not having the experience to build serverless applications is another challenge we are about to embark on as a team.
- One thing that might limit our options will be vendor lock-in. Different Serverless platform manufacturers use different integration methods, APIs, and documentation to enforce different amounts of lock-in [2]. However, this can also work in our favor if we pick a suitable service for a part of the application.
- API Gateway, for example, has come a long way in its first 18 months, but it still lacks several functionalities that we would expect from a universal web server (such as web sockets), and some of the functions it does have are difficult to use [2].

6. Timeline

Below table represents the development timeline of the project.

Table 1 Timeline for the project development

Date	Modules	Submissions
17th May 2022 - 31st May 2022	• Getting to know the Project Specifications • Understanding and getting hands on with cloud services like AWS and GCP	
1st June 2022 - 14th June 2022	• User Management Module • Authentication Module	Project Conception Report - 2nd June 2022
15th June 2022 - 28th June 2022	• Message Passing Module • Online Support Module	
29th June 2022 - 13th July 2022	• Machine Learning Module • Web Application Building and Hosting Module • Testing Module	Project Design Document - 5th July 2022
14th July 2022 - 25th July 2022	• Report Generation Module • Visualization Module • Project Documentation and Demo	Project Execution Q&A - 18th July 2022 - 22nd July 2022 Project Demo - 24th July 2022 Final Project Report - 25th July 2022

7. Meeting Logs

Meeting 1

Date: 17th May 2022

Title: Team Introduction

Meeting 2

Date: 25th May 2022

Title: Project Specifications Discussion

Figure 3 Meeting 2 log

Meeting 3

Date: 29th May 2022

Title: Cloud Services Discussion and Tasks Distribution

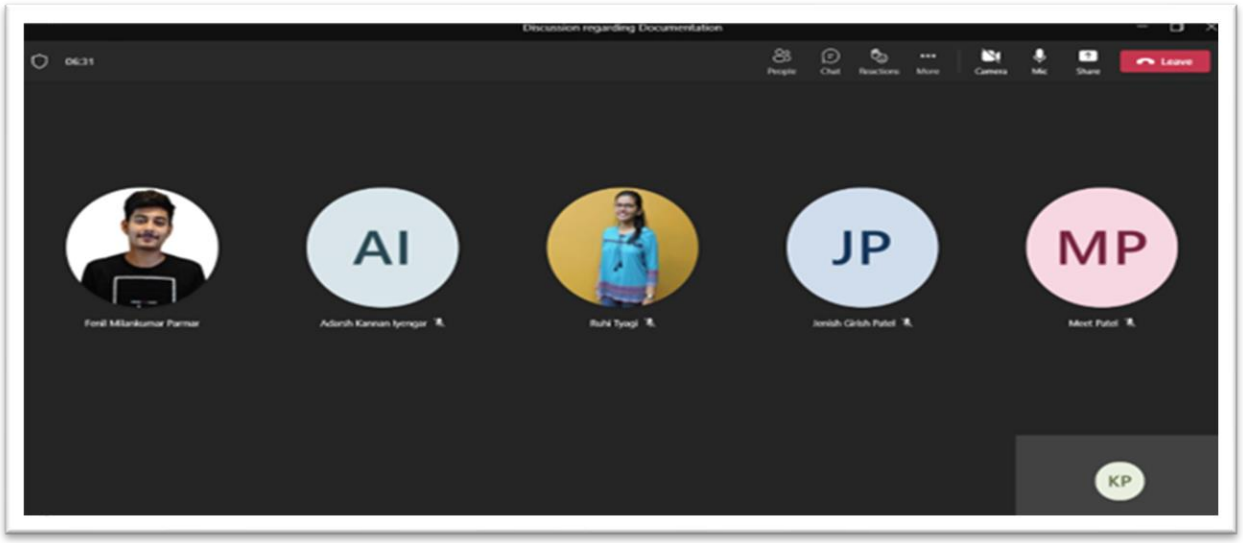

Figure 4 Meeting 3 log

Meeting 4

Date: 1st June 2022

Title: Conceptual Design Report Review

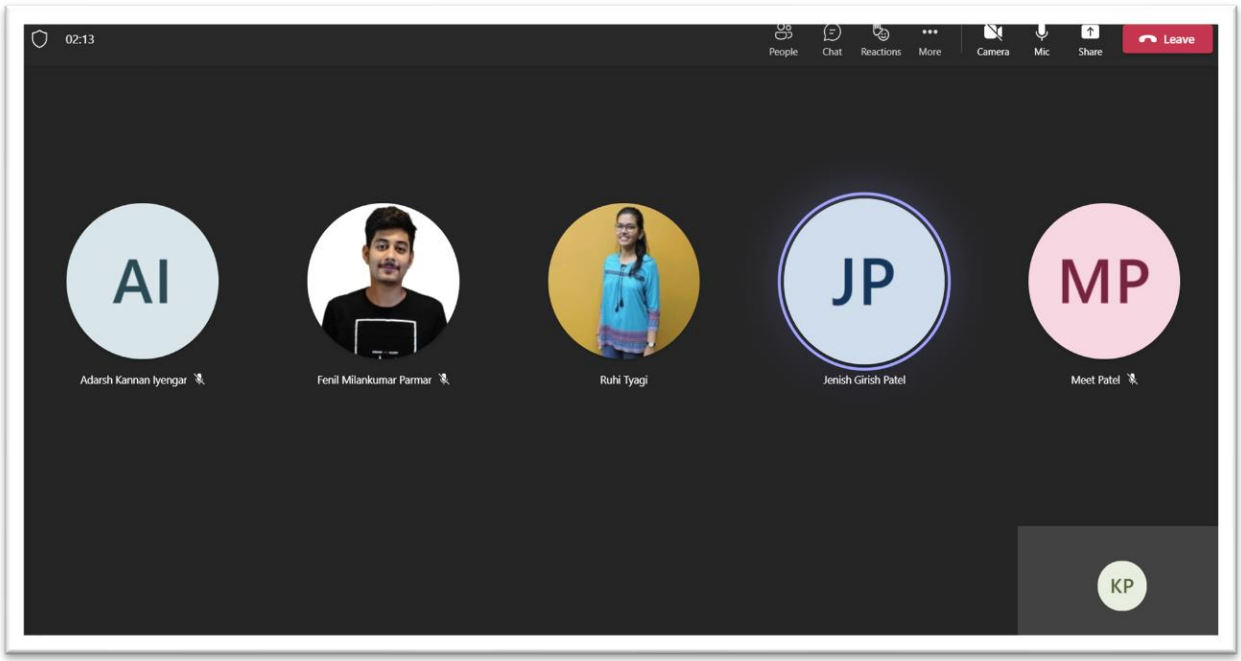

Figure 5 Meeting 4 log

8. References

[1] "Why Is Serverless the Future of Cloud Computing?", *Alibaba Cloud Community*, 2022. [Online]. Available: https://www.alibabacloud.com/blog/why-is-serverless-the-future-of-cloud-computing_597191. [Accessed: 30- May- 2022].

[2] M. Roberts and J. Chapin, "What Is Serverless?", *O'Reilly Online Learning*, 2022. [Online]. Available: https://www.oreilly.com/library/view/what-is-serverless/9781491984178/ch04.html. [Accessed: 30- May- 2022].

[3] "Flowchart Maker & Online Diagram Software", *App.diagrams.net*, 2022. [Online]. Available: https://app.diagrams.net/. [Accessed: 01- Jun- 2022].

[4] "Google Cloud documentation | Documentation", *Google Cloud*, 2022. [Online]. Available: https://cloud.google.com/docs. [Accessed: 01- Jun- 2022].

[5] "AWS Documentation", *AWS Docs*, 2022. [Online]. Available: https://docs.aws.amazon.com/. [Accessed: 01- Jun- 2022].