CSCI 5410 - Project Design Document

Team: **SApp_19**

(1) Hashik Donthineni - B00868314

(2) Fenil Nikeshkumar Shah - B00871894

(3) Pathik Kumar Patel – B00869765

(4) Dharaben Thakorbhai Gohil -B00884960

Application Architecture:

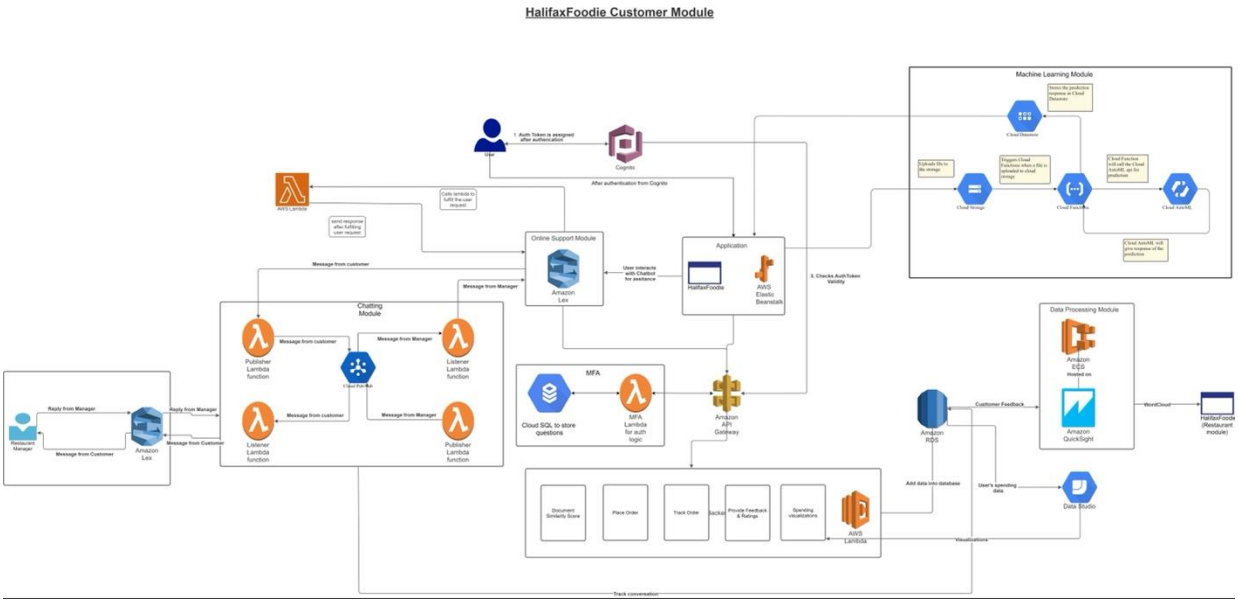

Figure 1 : Customer Application Architecture[1]

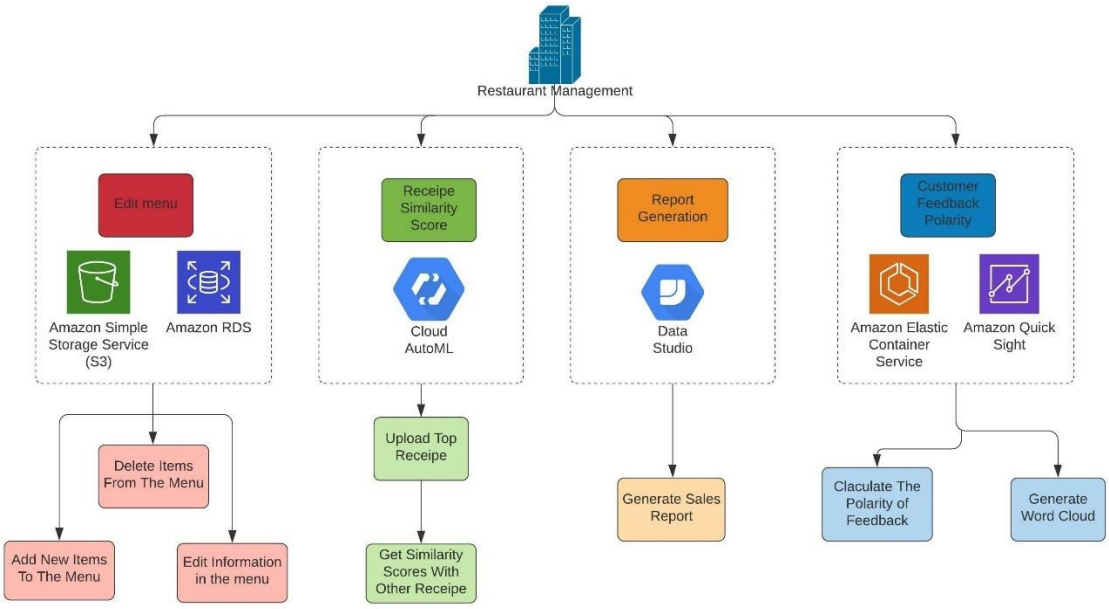

Figure 2: Restaurant Module Flowchart[1]

Components:

Application deployment on EBS (Elastic Beanstalk)

We will use Elastic Beanstalk to host our application. The front end will be using React.js and it will be hosted on EBS. All the other services will be connected to EBS, and the service request will be directed to appropriate service.

Design of Authentication Module

The AWS Cognito is used for user authentication with authorization rules set for the user-pool in the IAM

To authenticate a user request coming from the front-end to the API Gateway the Cognito is set as the authentication agent which would reject all the requests that aren't authenticated.

All the requests that need to go to a third-party API like the GCP API are still authenticated through the AWS API Gateway with Cognito, there are routing lambdas that would form the request and route the request to the third-party service/API.

Users' sign-up and sign-in using the AWS Cognito and after the user is successfully authenticated, on his first sign-up he'll be asked to set few security questions, after that's finished, we use those security questions as his MFA when he logs in next time.

Design of Chatting module using Amazon Lex, Cloud Pub/Sub, and Lambda function

The key components for the chatting module will be the **Amazon Lex**, **Cloud Pub/Sub** and a couple of **Lambda functions**.

Basically, the user will interact with the Amazon Lex and initiate the chatting/escalation request. After that a Lambda function will be invoked and subscriptions to the already created topics in Cloud Pub/Sub will be used. After that, on every user input, a Publisher Lambda function (Customer) will be invoked that will push the message to the Cloud Pub/Sub and then the configured Listener Lambda function (Restaurant team) will be invoked that will pull the message from the queue and display it to the restaurant team's application.

Similarly, when the restaurant representative will reply to that message via the front-end another Publisher Lambda function (Restaurant team) will push the message to the same topic in the Cloud Pub/Sub and then a Listener Lambda function (Customer) will pull the message from the queue and then it will be displayed to the customer in the UI. Similarly, the communication will continue until the fulfilment condition is satisfied by the Customer [2,3].

Data Processing for Food ratings word cloud using Amazon QuickSight and Amazon ECS

The key service that we will be using for the data processing module will be **Amazon QuickSight** and the service will be deployed on **Amazon Elastic Container service (ECS).**

A separate container-based application will be created to extract the named entities from customer food ratings stored in AWS RDS MySQL database and it will create a word cloud using Amazon QuickSight and the result will be displayed on the UI. This application will be deployed and managed in AWS ECS.

Data Storage and Retrieval using AWS RDS, API Gateway and Lambda functions

The key service that will be used to store the application data is the **AWS RDS MySQL** database.

All the data related to the food category, items, menu provided by the restaurant team, promotional (coupon) codes, etc. Will be stored in the AWS RDS MySQL database and the data will be retrieved using the backend **Lambda functions** invoked from the UI through the **API Gateway**.

Design of Chatbot using Amazon Lex and Lambda Functions

The customer will interact with the chatbot and ask their queries. Amazon Lex will get the user inputs and according to the received input, it will call the relevant lambda function to fulfill the request. After fulfillment of the user request, Lambda will give a suitable response to AWS Lex.

Design of Document similarity score module using GCP AutoML, Cloud Function, and Cloud Datastore

When the restaurant manager will upload the recipe. The recipe document will be uploaded to cloud storage. After uploading the recipe, it will trigger the cloud function. Cloud function will call the AutoML API and send data for prediction. AutoML will do the prediction based on the trained model and send the result back to the cloud Function. Cloud Function will store the results to Cloud Datastore in JSON format.

Design of Visualization module using Amazon RDS and Google Data Studio

When the customer wants to see the overall visualization of his/her spending based on the food items categories and restaurant, the **Google Data Studio** is used to represent the data in the graph format. This graph may be a pie chart, bar graph or line graph.

The data of the spending of the customers are stored in AWS RDS. **Google data studio** will fetch the data from the **AWS RDS Database** and convert it into useful data and represent it in the form of a graph. [4]

References

[1] "diagrams.net - free flowchart maker and diagrams online," Flowchart Maker & Online Diagram Software. [Online]. Available: https://app.diagrams.net/. [Accessed: 03-Jul-2021].

[2] "Use Pub/Sub to send and receive real-time messages," Google. [Online]. Available: https://cloud.google.com/community/tutorials/pubsub-quickstart. [Accessed: 03-Jul-2021].

[3] P. author B. Himanshu, Sriharsha says: and N. *, "Integrating PubSub Cloud with Lambda via API Gateway," a bit deployed. [Online]. Available: http://abitdeployed.com/2020/05/14/integrating-pubsub-cloud-with-lambda-via-api-gateway/. [Accessed: 03-Jul-2021].

[4] Data reporting & dashboarding key features – Google data Studio. (n.d.). Retrieved July 3, 2021, from Google.com website: https://marketingplatform.google.com/about/data-studio/features/