COVID-19 Community Mutual Assistance Forum

Project Report

ECE1779 Assignment 3 Serverless Computing

Zichuan Wang	Zhaohui Qu	Huan Qi
1000474300	1005783127	1006437214

1 Introduction

The spread of COVID-19 has made communities kick into high gear in preparation for supporting those most impacted by the virus. The goal of our application is to provide a way for our local community members to request and provide support to each other. Some key features that our application provides are as follow:

- **Mutual Assistance Forum:** As community members, we all have skills and resources to contribute to share with one another. Here is a platform for people to connect with each other and exchange, whether it's Food, Shelter, Medical Care, Utility Services, Child Care, Senior Care and More.
- **Community Resource Guide:** There has been a lot of announcements and changes to regular services around the country. This guide tries to bring all of this news together into one place and be regularly updated. You will find info about services related to food, housing, child care, transportation, utilities, and more.

2 How to Use

We first deploy the application using Zappa with the command `zappa deploy dev`. A Zappa settings JSON file has already been created and can be reused. If any changes need to be made after deployment, run the command same command again. Once the application is deployed onto AWS, it can be accessed through the link generated by Zappa. Following steps can be performed to realize the full functionality of the app:

1. After accessing the app through the link generated by Zappa, user will be directed to the home page. Home page consists of 2 sections including community resource guide and COVID-19 response timebank. In community resource guide, general information about COVID-19 related source guide can be found. In COVID-19 response timebank, links for request help and offer help portals can be found where community members can interact with each other.
2. If you are new to the website, click the register button on the top right corner to create a new user. If you are existing user, click the login button on the top right corner to login

3. After logging in, you will redirected to the home page. If you would like to offer help, click the offer help button on the navigation bar. In the offer help page, a list of all the existing request help posts will be displayed. You can filter the posts based on the distance between your current location and location of the help requestor, time window you would like to offer help and type of help you would like to provide. After you click Search, all qualified posts will be displayed along with their corresponding location as marked on the map. Detailed request information will be displayed if click on the marker. Also, filter request form can be hide by clicking the Filter Request button.

4. If you would like to request help, click the request help button on the navigation bar. In the request help page, you can edit the title of the post, time window you would like to receive help, help type, contact information, residential address and detailed request description. After Post button is clicked, your help request will be posted onto the website. You can also edit your posts if you changed your mind or delete your posts if you no longer wish to receive help.

5. Account information can be accessed and updated including profile image, username and email address by clicking on the Account button on the top right corner once you logged in.

3 Software Architecture

This application primarily consists of two lambda functions – one of which hosts the main web application, and the other conducts garbage collection after posts have expired or have been deleted. The lambda hosting the main web application is deployed with Zappa, and the secondary background lambda function is invoked whenever metadata of corresponding DynamoDB changes.

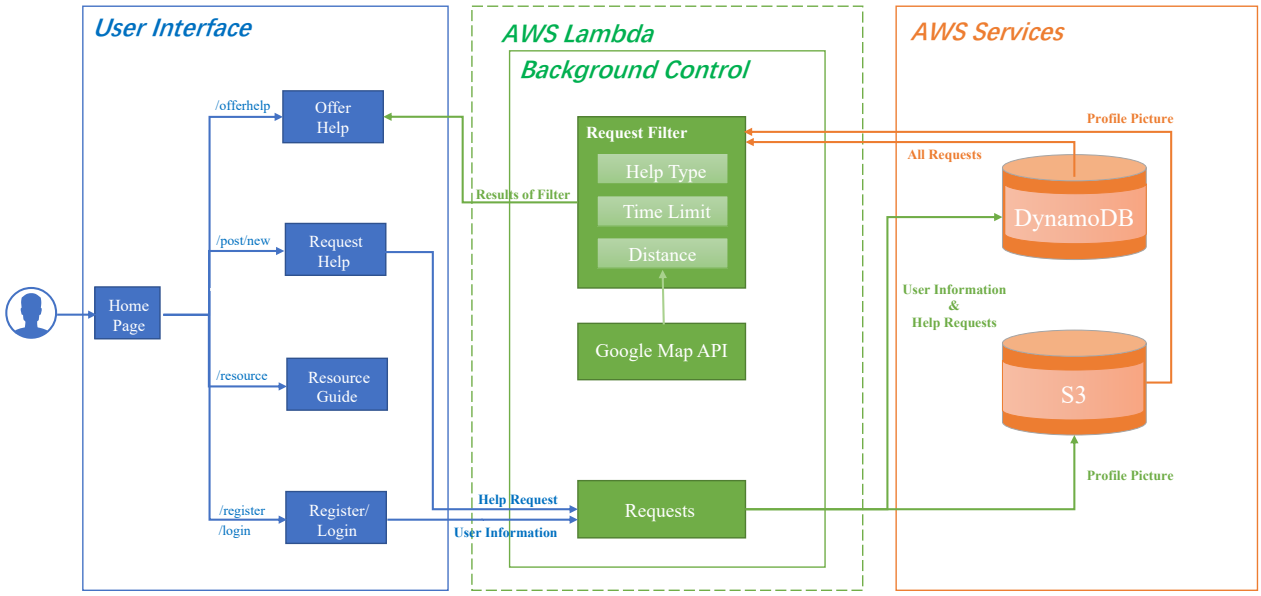

Figure 1: Software Structure

3.1 Primary Web Application

- Deployed by Zappa's automatic API gateway configuration, Lambda will be triggered by HTTP requests to the web application. The primary web application allows account register, log in, post creation, post update, post filter.
- User and post data are saved on DynamoDB, the application directly interacts with the DynamoDB to perform queries and make updates if required.
- User profile pictures and corresponding thumbnails are saved in S3 bucket.

3.2 DynamoDB

DynamoDB is used for storing user and post information, detailed structure of the user and post table are as follow:

User Table:

- username: stores the username of the corresponding account
- password: stores the password of the corresponding account
- email: stores the email of the corresponding account
- imagefile: stores the name of the profile picture stored in S3

Post Table:

- post id: stores post id for post query
- title: stores the title of the post
- type: stores the type of help requested
- status: stores current status of the post, entires of expired or manually deleted posts will be cleared using background lambda function.
- phone: stores the phone number of the help requestor
- email: stores email address of the help requestor
- content: stores detailed information about the requested help
- timestamp: stores the times that the post was created
- sdate: stores the starting date of the requested help
- fdate: stores the ending date of the requested help
- lat: stores the latitude information of the location for the helper requestor
- lng: stores the longitude information of the location for the helper requestor

3.3 S3

S3 is used for storing user profile picture.

3.4 Background Lambda Process

Background Lambda function is triggered whenever metadata of corresponding DynamoDB changes. Function will scan the database for "pending deletion" flags. Deletion will be performed for the database entries whose status flag equals "pending deletion". Function will also scan the database for "expired" flags. Deletion will be performed for database entries whose status flag equals "expired".

4 Cost Model

For the cost model, the resources used are:

- On demand DynamoDB
- Free tier S3 bucket
- AWS Lambda

Following assumptions are made about user behaviour such as requests per day:

- A user makes at most 1000 http requests per day with the web application
- A user makes at most 1000 write requests to the DynamoDB
- A user makes at most 5000 read requests to the DynamoDB

Daily Cost ($)				
Service	1	10	1000	1000,000
DynamoDB	0.0025	0.025	2.5	2500
S3	0.023	0.023	0.023	0.023
Lambda	0.012	0.12	12	12000
Total Cost	0.0375	0.168	14.532	14500.023
6 Months Cost ($)				
Service	1	10	1000	1000,000
Total Cost	6.75	30.24	2615.76	2610004.14

Table 1: Cost Model