Posted

1st Benson Gao
Computer Engineering and Computer Science
California State University Long Beach
Long Beach, USA
bnsn.gao@gmail.com

2nd Nick McDaniel
Computer Engineering and Computer Science
California State University Long Beach
Long Beach, USA
nickamcd@gmail.com

3rd Michael Ly
Computer Engineering and Computer Science
California State University Long Beach
Long Beach, USA
michaelly03@gmail.com

4th Julian Tran
Computer Engineering and Computer Science
California State University Long Beach
Long Beach, USA
juliantran097@gmail.com

***Abstract*—This document is to introduce and explain our mobile application: Posted. We will be discussing the motivation of our application, the problem it is addressing, emphasizing the features that our application has compared to others, and summarizing the design, implementation, and testing.**

I. Introduction

Our mobile application will let users around the world interact with each other to discuss various destinations. The key feature of our application is that "guides" may be able to appear under any given location that they have visited. This lets other users contact these to learn more about the place as well as any other locations that that guide has visited.

II. Related Work

What sets this application apart from other platforms like Yelp or TripAdvisor is the ability for users to receive more personal recommendations based on interests. For example, a user may want to visit a city that is not very well known or not considered a typical tourist location. The user may want to know about local events or relaxing/sightseeing places, but other platforms may not have information on this, especially if the locations the user are interested in are less known. This application can also benefit users interested in tourist locations with the ability to learn about different routes and shortcuts to take in the area at peak times, safe and dangerous areas, hidden gems catered to the users interest (such as a restaurant that may not have as much exposure), and many more depending on how the users intend to use it.

The way our application achieves this is by providing users to create and modify their profiles, and also have the ability to interact with other users. Each user with their profile are able to publicly provide their personal interests and connect with other users with similar interests. Two users can contact each other and communicate personally to get more insight and help about what they are looking for and interested in. Users also have the option to leave the conversation as they wish, and may also report or block the other user if any problems has arise. Furthermore, the users can rate others that they have interacted with based on the helpfulness of their conversation. Our application additionally connects with services such as Google Maps and Yelp for relevant information.

III. Design

On application start up, the user will be prompted to login. If the user does not already have an account, they may click "Not Registered? Sign up here." and follow the process of making an account. This can be seen in figure 1 below.

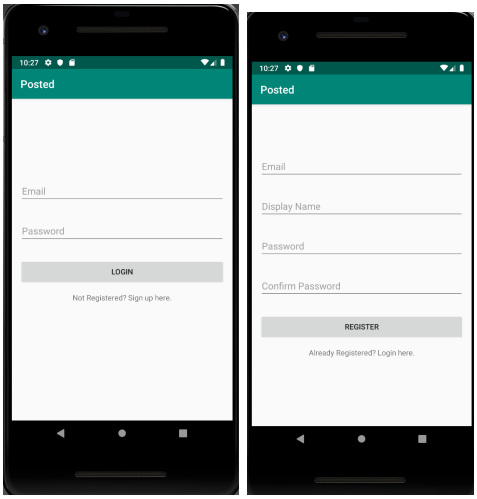

Fig. 1. Login/Register Page

Once the user has successfully logged on, they will be taken to the homepage of the application. By default, the homepage will display several locations that is nearby the user. This can be seen in figure 2 below.

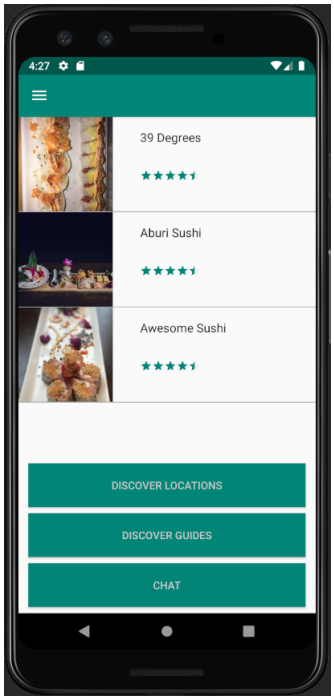

Fig. 2. Home Screen

At the homepage, the user has many different options to choose from. One of the options is to ”Discover Locations.” If the use decides to click on this button, they will be taken to a page where it will display a list of all possible locations. This can be seen in figure 3 below.

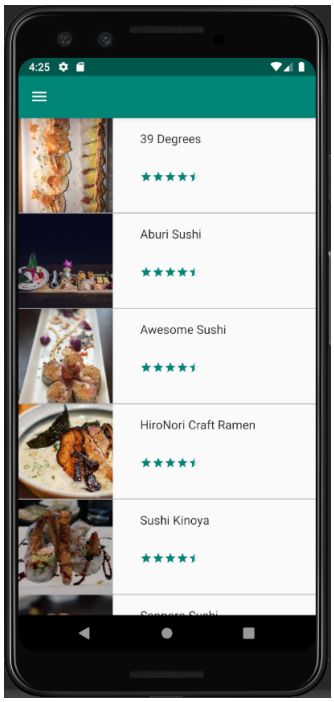

Fig. 3. Discover Locations

From here, the user can select a location they are interested in. Once a user has selected a location, they will be able to see a bit more information about that specific location and a list of guides that have been to this location right underneath. The user could then click the corresponding button to open the location in Yelp or in Google Maps. This will open up the native application if the user has it installed on their phone, otherwise it will open up in their browser. This can be seen in figure 4 below.

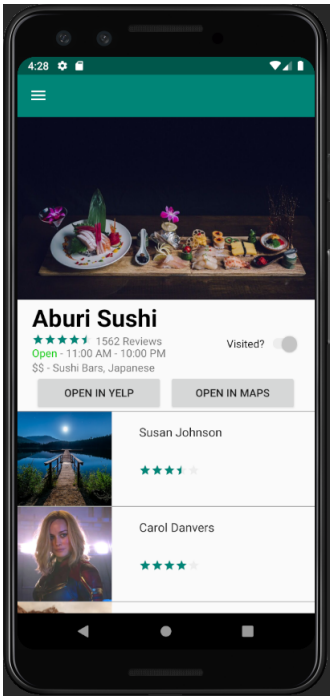

Fig. 4. Location

Back at the homepage (figure 2), the user also has the option to ”Discover Guides.” If the user clicks on this button, they will be taken to a page with a list of current guides. This can be seen in figure 5 below.

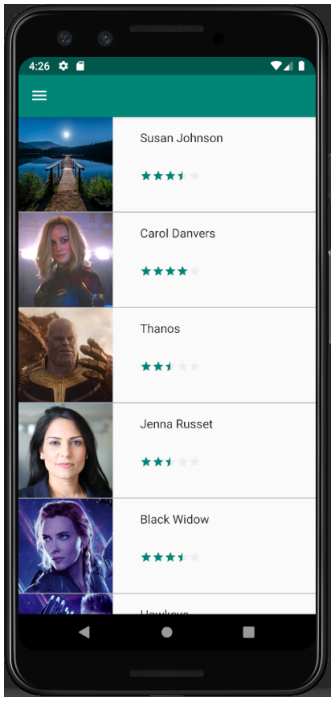

Fig. 5. Guides

The user may click on whichever guide they please. Once the user has clicked on a guide, they will be able to see a list of locations that the guide has been to. The user may click on a location, and it will bring them to figure 4 above. The user may also choose to send a message to the user. This will take the user and the guide to a chat room, and from there they will be able to message each other. This can be seen in figure 6 below.

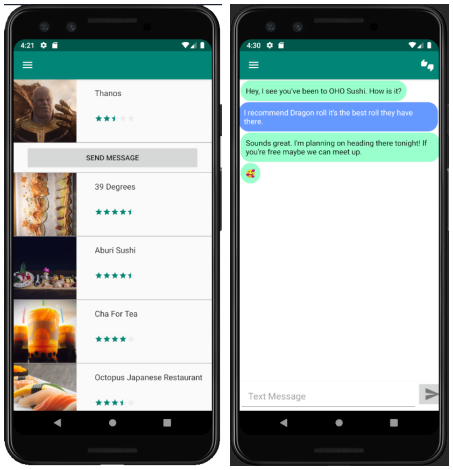

Fig. 6. Chat

Back at the homepage (figure 2), the user can also click on ”Chat.” This will take them to the page of their current chats that they have. The user can click on any chat and it will take them to the figure 6 seen above earlier. This can be seen in figure 7 below.

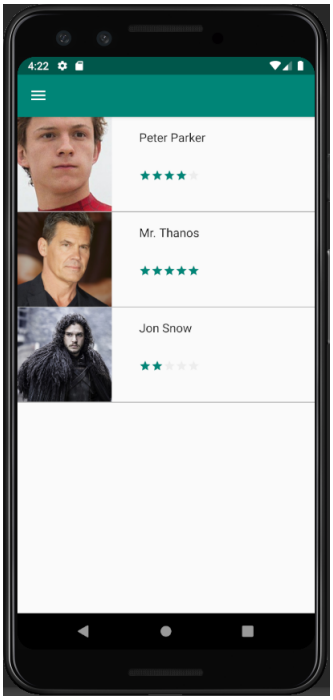

Fig. 7. List of Chats

IV. Implementation

The implementation of Posted was done primarily on Android Studio with some work being done on Firebase for the implementation of security rules.

A. Problems

We ran into numerous problems throughout the implementation process. Most of which we were able to overcome but some of which went unresolved through the end of the development time-frame.

1) Setup Issues: We ran into some technical issues at the start of the development time-frame with some developers having issues with Android Studio and setting up the IDE. Another member ran into hardware issues that required the purchase of a new laptop. Both these issues were resolved within a week and didn't incur much of a setback in the grand scheme of things.

2) Learning Curve: While we anticipated that Firebase, the Yelp API, and Android would be easy to learn and use based on our limited collective experiences with the technologies, each technology proved to be difficult in their own ways. Firebase presented us with issues due to the

complicated methods involved with retrieving information stored on our database as well as the complexity of our schema which was required to change and grow during Implementation to accommodate different features we did not expect to need. The Yelp API proved to be especially frustrating due to the library for Yelp's Android API not being updated in the last two years. The implementation ended up requiring the usage of a third-party fork of the API which still required some amount of modification due to not being updated for a year. Even then, the handling of data from Yelp also proved to be frustrating due to formatting mismatches. While Android wasn't as frustrating to deal with, our inexperience with the platform provided some of the largest challenges. The implementation of our application required the usage of many aspects of Android that we were unfamiliar with. A significant portion of our development time-frame was used up by reading and re-reading tutorials and documentation for the various features that we needed. Most notably among those were fragments which required a complete refactor of our code a month into development, shared preferences for tracking settings across the application, and recycler views to display lists of locations or guides.

3) Personnel Problems: We also ran into major scheduling issues where it was extremely difficult to find times to meet in person outside of class time. When some members were free, others had class or work or other obligations. Other classes and work were also a major obstruction to the amount of effort that could be given to the project. These issues combined with the fact that our group had fewer members than every other group resulted in the implementation not receiving the level of polish that we had hoped for during the design phase.

B. Results

The implementation results can be broken down into four major components: user authentication and accounts, guides and profiles, locations, and chat.

1) User Authentication and Accounts: User authentication's implementation was handled primarily through Firebase's FirebaseAuth and FirebaseUser libraries. These libraries allowed us to easily handle user login and registrations by passing through values based on the user's inputs. Upon account creation, we also initialize that user's entries in the database and local shared settings. This allows for users to sign out and log in to a different account while keeping settings and preferences on each account separated.

2) Guides and Profiles: Because guides are the defining feature of Posted, we wanted to ensure that the implementation was as solid as possible. Guides were constructed based on the profiles we built for every single user. Every profile keeps track of the user's preferences as well as locations they've marked as visited. When a user toggles their guide status, they are added to each location as well as our total guide pool. In each of these spots, a recycler view is used to populate a list displaying each guide's profile photo, display name, and rating. These lists are sorted through our algorithm to rank guides based on their compatibility with the user as well as their overall rating. Selecting a guide from one of these lists brings the user to the guide's profile page. From here, users can request to send a message and see all the locations that the guide has visited.

3) Locations: Locations are a similarly important feature of Posted and carry many similar attributes to guides. However, each location is built on pulling relevant information from the Yelp API rather than user input. Locations also use recycler views to populate lists of locations but they are not sorted in house. Their sorting is handled through Yelp and the lists are in whatever order Yelp's API retrieves locations from a query. The query to discover guides is built through passing through all of the user's selected suggestion preferences. While this may result in too wide a variety of results if a user selects a large number of suggestions, the user is expected to frequently change their suggestion preferences to match their current mood rather than a set and forget approach. Other queries are based on specific location IDs that are stored on our database.

4) Chat: Chat is the last major component for Posted and its connection to the guides feature means that it is also extremely important to the success of our application. The chat method functions in two parts, sending and fetching messages. To send messages, the class creates a database reference underneath the current users ID that is made up of the message receivers ID. Then the fragment takes the text that the current user types and creates a hashmap that also contains the time that the message was sent. It then creates a unique key for each message from these values and saves it into the database. For receiving messages, a database reference is created and is pointed directly to the level of children underneath the message receiver ID and adds a listener for any children underneath it. This way, whenever a message is sent, a child is added and the listener updates an arraylist that is used to display all of the messages in order of when they were received. Finally, the displaying of messages is handled by an adapter that checks to see if the message in the arraylist is was sent or received by the current user and will display them with the direct format accordingly.

V. Testing

To test our application, we first had to decide what features were important to test as well as which ones did not need to be tested. What we decided to test on goes as follows: Login Credentials, the Messaging System, the Guide Discovery Algorithm, Server Latency, API Integration, and Profile Modifications. From this list, we also derived the priority of features that need to be tested as to make sure that we put more time and effort into testing features that we felt must be completely functional for for the app to work as intended. Due to issues that arose within the development cycle, we were not able to perform rigorous

unit testing, but we were able to perform limited manual value testing to ensure that our product works as intended.

A. Login Credentials

The first of the high priority features to be tested was the Login Credentials system. To test this, we created multiple scenarios to validate that 1). An account was successfully created, 2). The information was correctly saved to our database, and 3). When logging in, the information is validated so that a user can only access an account with the correct email address and password. The authorization system also works with emails created with unicode symbols from other languages.

B. Messaging System

Another high priority function to test was the messaging system, as this is intended to be a form of social media which would require a functioning way for users to interact with each other. To test this functionality, we came up messages that would be sent with varying length as well as unicode symbols from other languages. No symbols that were tested caused issue with the messaging system or how the information was saved within the database.

C. Server Latency

Server Latency was tested by having multiple accounts send messages of large length at the same time to see if it would have an affect on the response time of the application when it came to updating information that a user requests. We found that there were no hindrances on the performance of the application or its ability to continue to gather information from the database.

D. API Integration

API Integration testing was done through sampling many of the locations provided by the Yelp API and cross-checking with the actual Yelp page to verify that the information pulled is accurate, is translated to the client facing language, and is being displayed properly. Through our testing, the only major issues that we encountered was the variety of dimensions on location photos and that the isClosed method does not return true if the business is closed for the day but rather permanently closed, which was not explained clearly in the documentation.

E. Profile Modifications

Profile Modification testing was performed by not only making changes on the client side, and checking to make sure that the changes reflect properly on our database but also to make changes directly to the database and ensure that the changes take place the next time the user opens the app. Testing was also done to make sure that profile modifications would only apply to one user and that they would not affect any other accounts being logged in on the same device.

VI. Conclusion

The objective of our application is to create a application that allows a user to find a local guide based upon the user's preferences. The user can find a guide that has visited a specific location or has similar interests. The user can then start a discussion with the guide to inquire more information about that specific restaurant or event. Future works involve creating a location based service, creating notification system, cleaning up unnecessary code. The current state of our program only finds locations with in Long Beach and currently doesn't notify the user when they receive a message.