[**Description of Mezcla**](#_srl6rq6wu4fx) **2**

[**Reasons for Mezcla**](#_tokvtqiolf7r) **2**

[**Feature List**](#_unhg0o2pdtsr) **3**

[**Roles & Contributions**](#_wq288ojk1dn0) **4**

[**User Guide**](#_1mvscsx6gf4o) **5**

[Registration Page and Login Page](#_wyz3gtuo6jik) 5

[Creation of Profile](#_fxu38qo0pnmm) 12

[Homepage - Usage of Fragments](#_3ocmuf4x14t) 13

[Search Functionalities](#_x376ltjjwmhu) 14

[Search By Ingredients](#_gsgmbju02322) 15

[Search By Name](#_cqkvjvy1w04g) 17

[Google Maps](#_3da6l31n0z6) 21

[Profile Settings](#_spgtcp1pvccx) 23

[**Problems Encountered**](#_s7gxb0ygptqu) **24**

[**Improvements made after first presentation**](#_1xqoineynjiq) **26**

[Night Mode](#_cacs47srql0e) 26

[Addition Search Function](#_naymv44vfazf) 27

[**Design and Ideation Stage (Initial Ideas)**](#_won6c474ycrx) **30**

Yong Kai En (S10173513B)

Nathan Tan Hong Da (S10175720H)

Tan Jian Zhen (S10177664C)

# **Description of Mezcla**

Mezcla is an application that uses Model View ViewModel (MVVM) software architecture pattern, which allows users to search for relevant cocktails based on the ingredients they have or to search for cocktails based on its name to find out more information such as its recipe and instructions to construct the cocktail. The application also provides users with the nearest nightlife locations that are close to their GPS location. The application utilizes “TheCocktailDB” API as its database. (<https://thecocktaildb.com/api.php>)

# **Reasons for Mezcla**

It is an Android application for cocktail lovers. Powered by “TheCocktailDB” API, the application has the capability to list hundreds of cocktails and their own individual unique recipes. In the situation that the user has some beverages and fruits and wonder what cocktails they can make - Mezcla will be able to help you with that. Additionally, if the user is interested in constructing a particular cocktail but am not sure of the instructions to do so, the application will be able to help them as well.

# **Feature List**

|  |  |
| --- | --- |
| Feature Name | Functionality |
| Login | User will need to enter their login credentials to access the application |
| Registration | User is able to register for an account using their email address and password |
| Profile Setup | After the user has successfully registered his account, they will need to select a display picture and enter a display name |
| Logout | The user is able to log out of their account in the Settings page |
| Search by ingredients | User can enter one or more ingredients in the search bar and the application will display a list of cocktails that contain the ingredient |
| Search by cocktail name | User can search for a particular cocktail by entering the name, and the application will display the search results based on the query |
| View all ingredients | The user is able to view all ingredients stored in the database by not entering any values in the search bar |
| View all cocktails | The user is able to view all cocktails stored in the database by not entering any values in the search bar |
| Nearest nightlife locations | User can view the nearest nightlife location and their own current location |

# **Roles & Contributions**

|  |  |
| --- | --- |
| Member | Contributions |
| Yong Kai En | Basic functionalities such as Login, Register, Edit Profile, View Profile and Maps etc  Validation Implementation; Error handling  Report  PPT Slides |
| Tan Jian Zhen | Bringing in the API and scrape relevant data that our application requires  Search functions such as “Search By Ingredient” and “Search By Cocktail Name” |
| Nathan Tan | Report, helped with designing of the application such as wireframing using Balsamiq Mockups 3  PPT Slides |

# **User Guide**

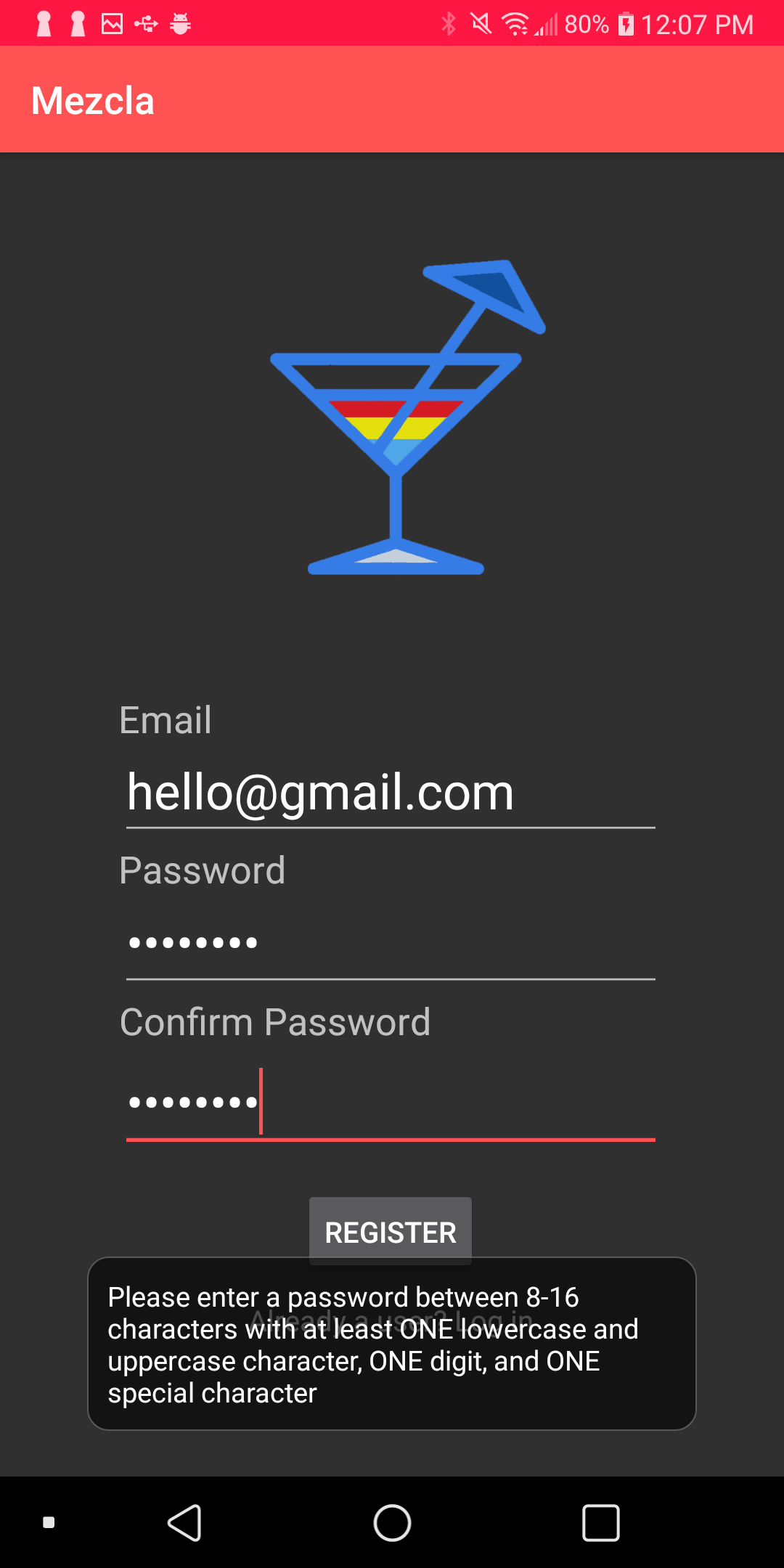
## **Registration Page and Login Page**

To access the application, the user is required to create an account. During the registration process, Mezcla will prompt the user for their email address and password. After their account has been registered, the user will also be required to set a display name and upload a display picture.

To ensure that users are registering with their email address and a strong password, we have implemented multiple error validations that will show different error messages to the users.

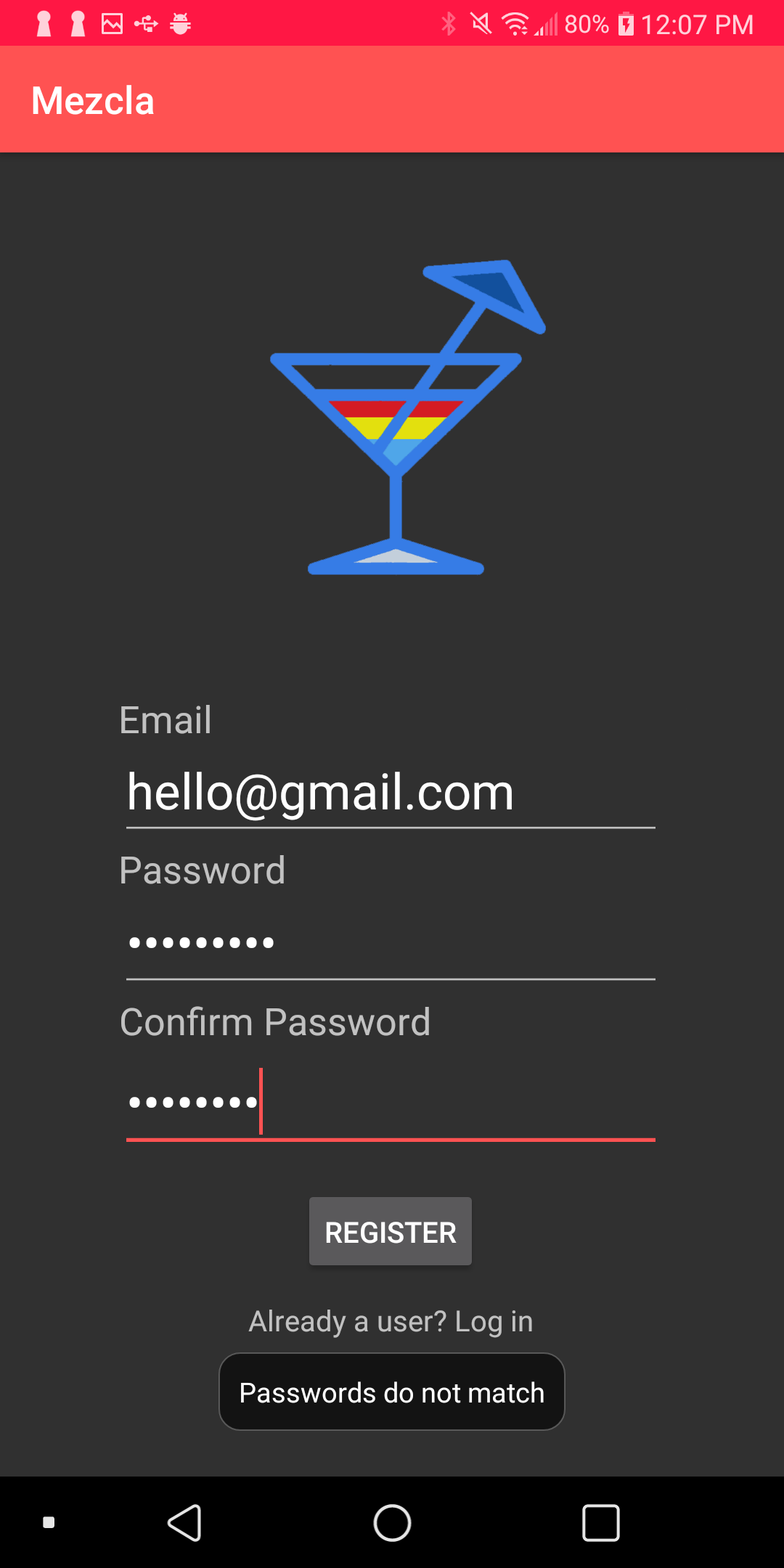
To create a new user, we have used Firebase’s pre-defined method “createUserWithEmailAndPassword” for the registration page.

On the other hand, we have used “signInWithEmailAndPassword” for the login page to log the user into the application.

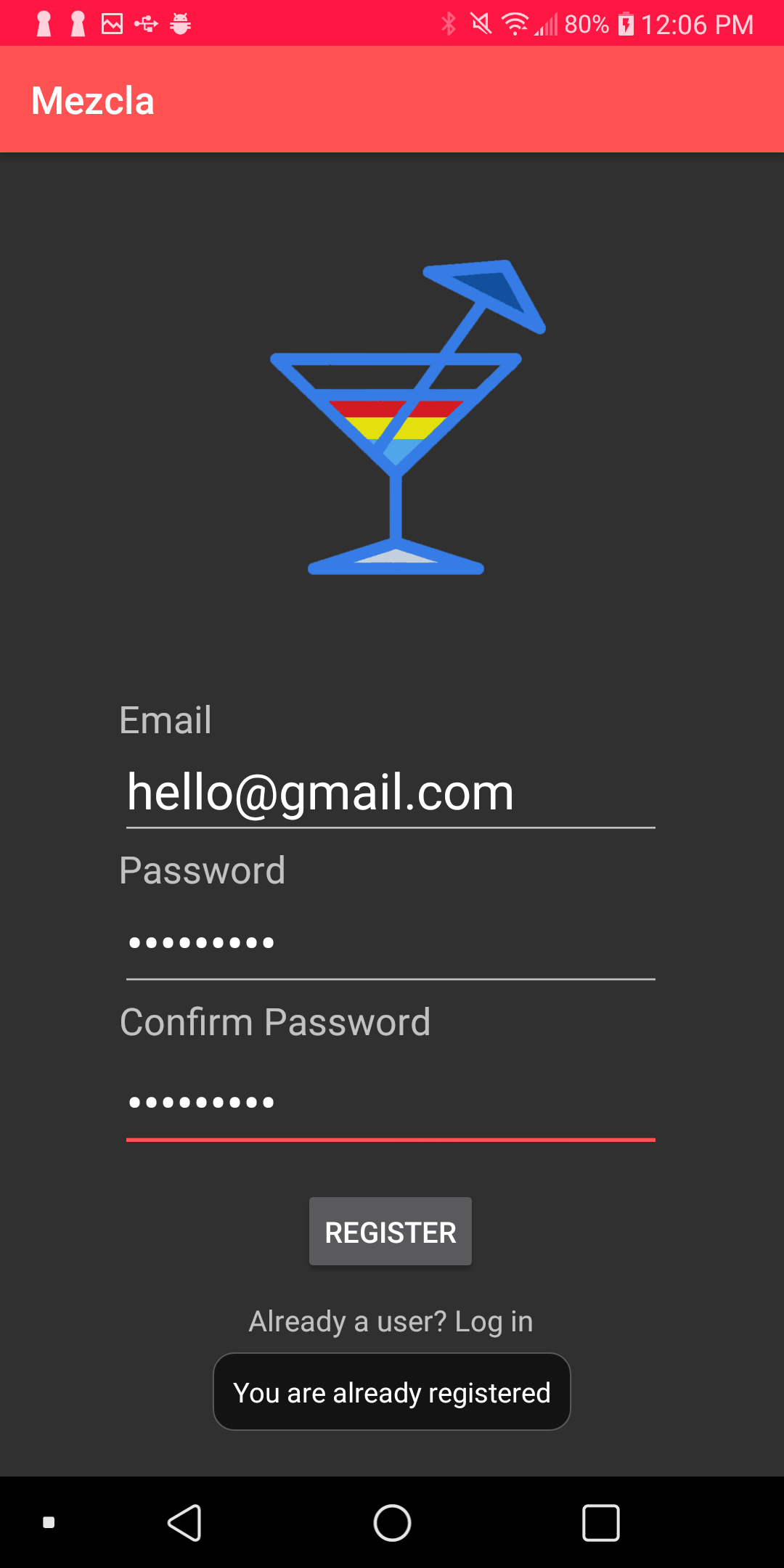


The registration page will display an error message near the bottom of the screen to users that their password does not contain lowercase, uppercase, digits, or special characters. We have masked the password field for privacy reasons. Additionally, we have utilized Regular Expression (Regex) syntax, a pattern that describes a set of strings that matches the pattern. A Regex accepts a certain set of strings and rejects the rest.

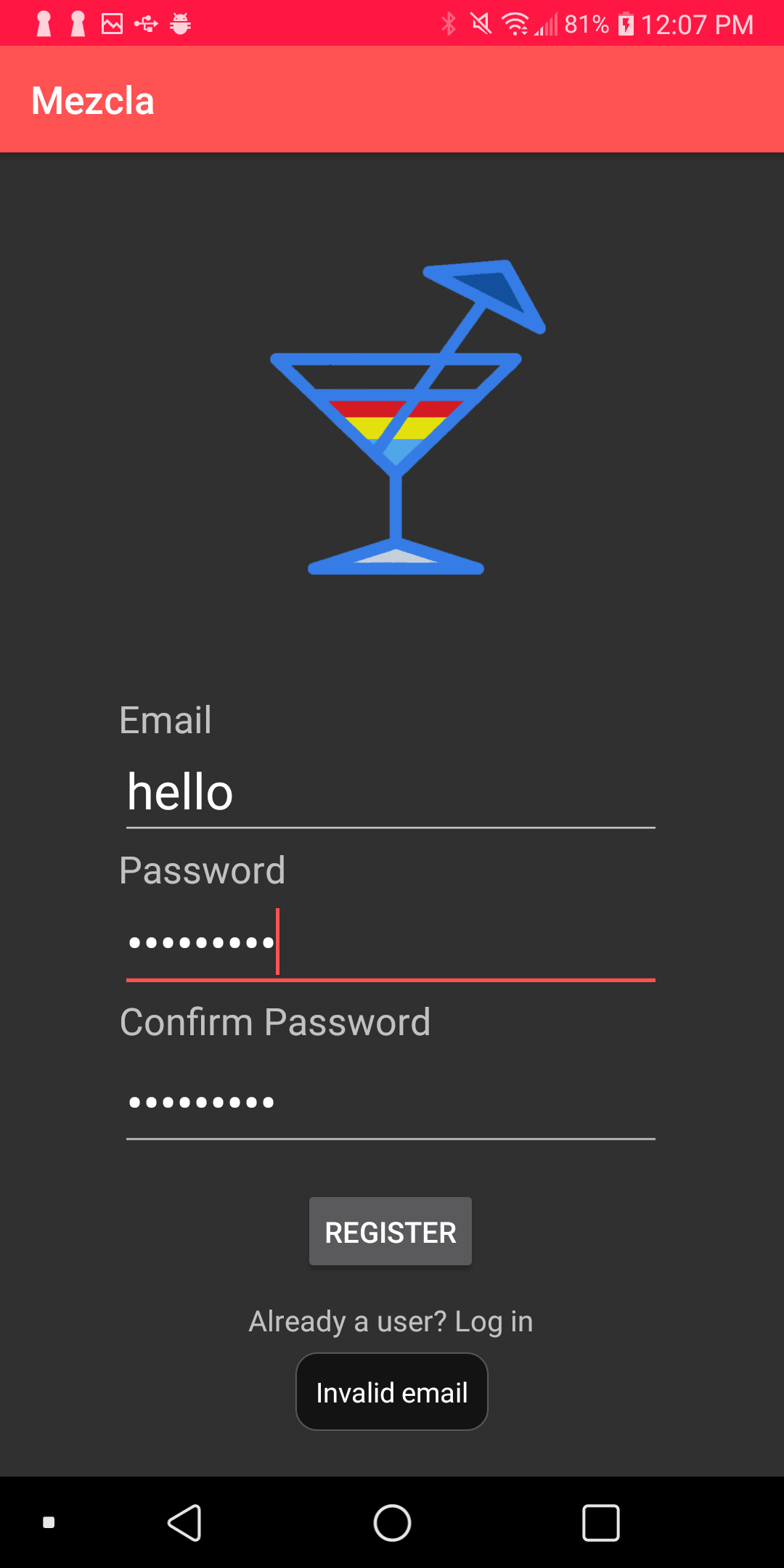
For our registration page, we have specified the Regex pattern to be "^(?=.\*[0-9])(?=.\*[a-z])(?=.\*[A-Z])(?=.\*[!@#$%^&+=]).{8,16}$". The pattern means that the application will only accept a password between 8 to 16 characters, with at least one lowercase and uppercase character, one digit and one special character. The reason why we have implemented this validation is to ensure that the user has set a password for higher security to prevent attackers from logging into their accounts easily.



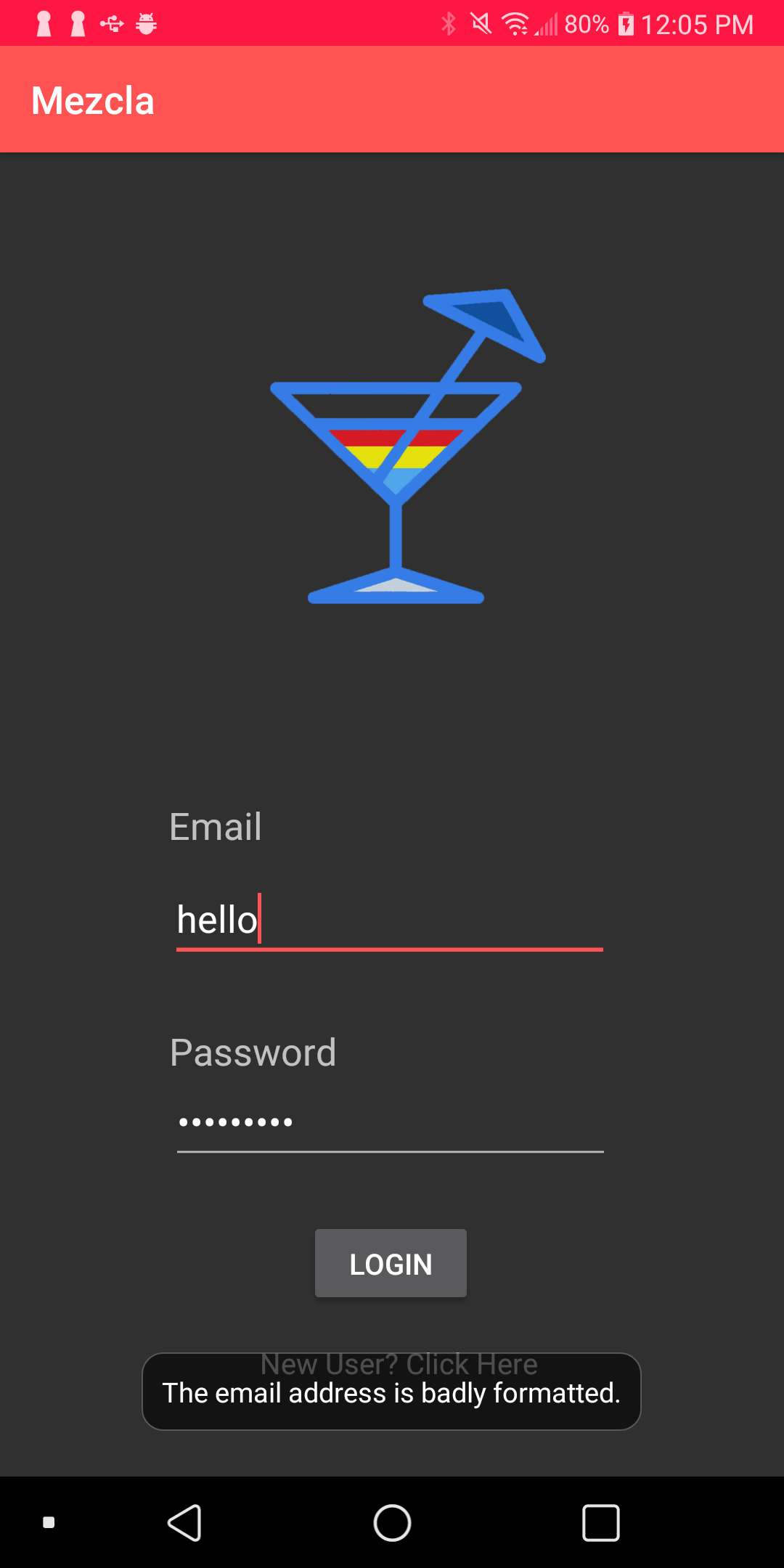
The registration page will display an error message at the bottom to users that have entered a different password. As the password field is masked, the application will ask for user’s password twice during registration to prevent any typing errors. To implement this is simple, we have compared the strings of both EditTexts.



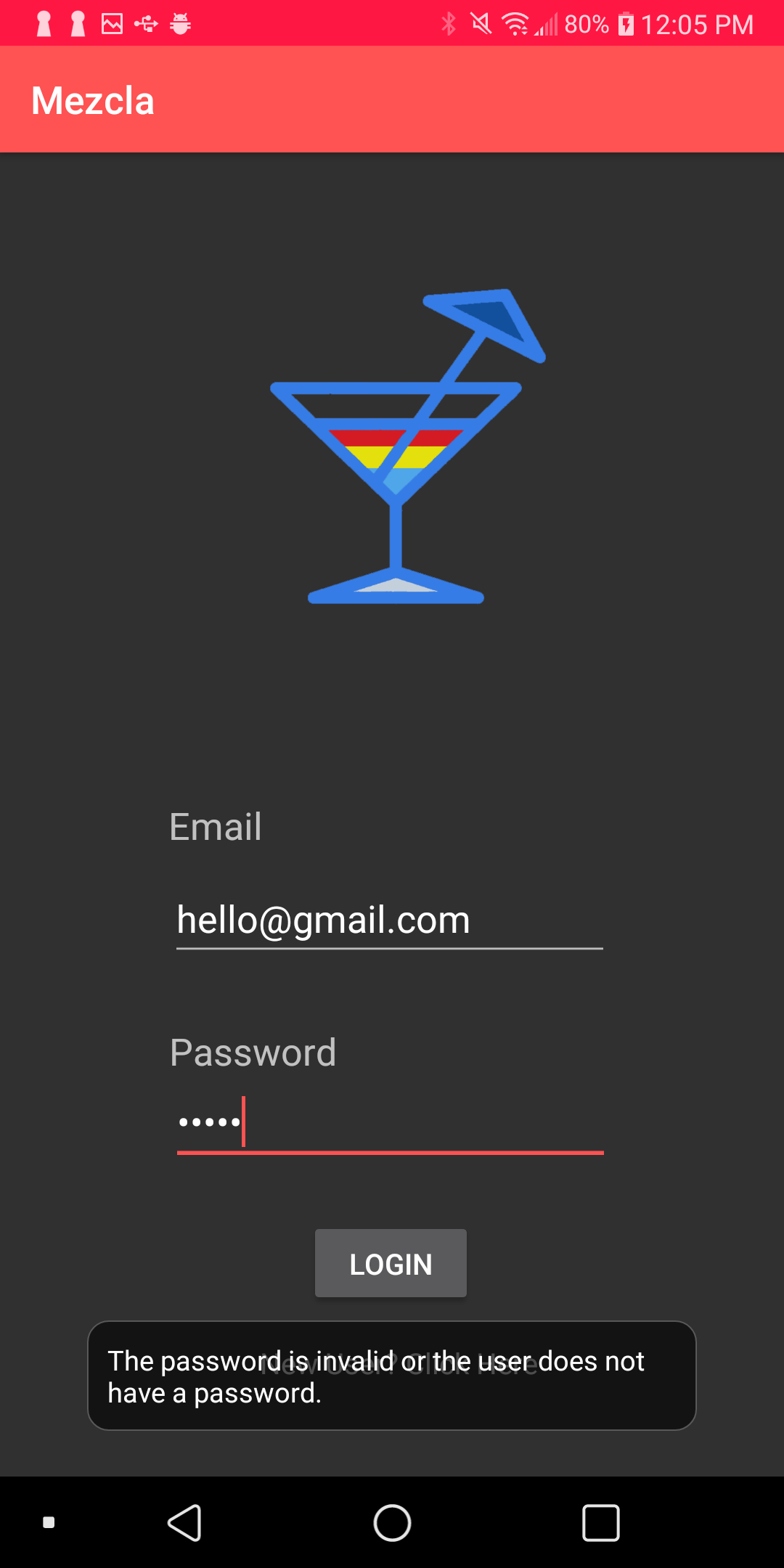
The registration page will display an error message near the bottom to users that have previously registered an account with the email address. This is implemented using FirebaseAuthUserCollisionException, which is an error exception if there is an existing user in Firebase.



The registration page will display an error message at the bottom to the user that have entered an invalid email address. The reason why we implemented this is to ensure that the user is entering their own correct email address.

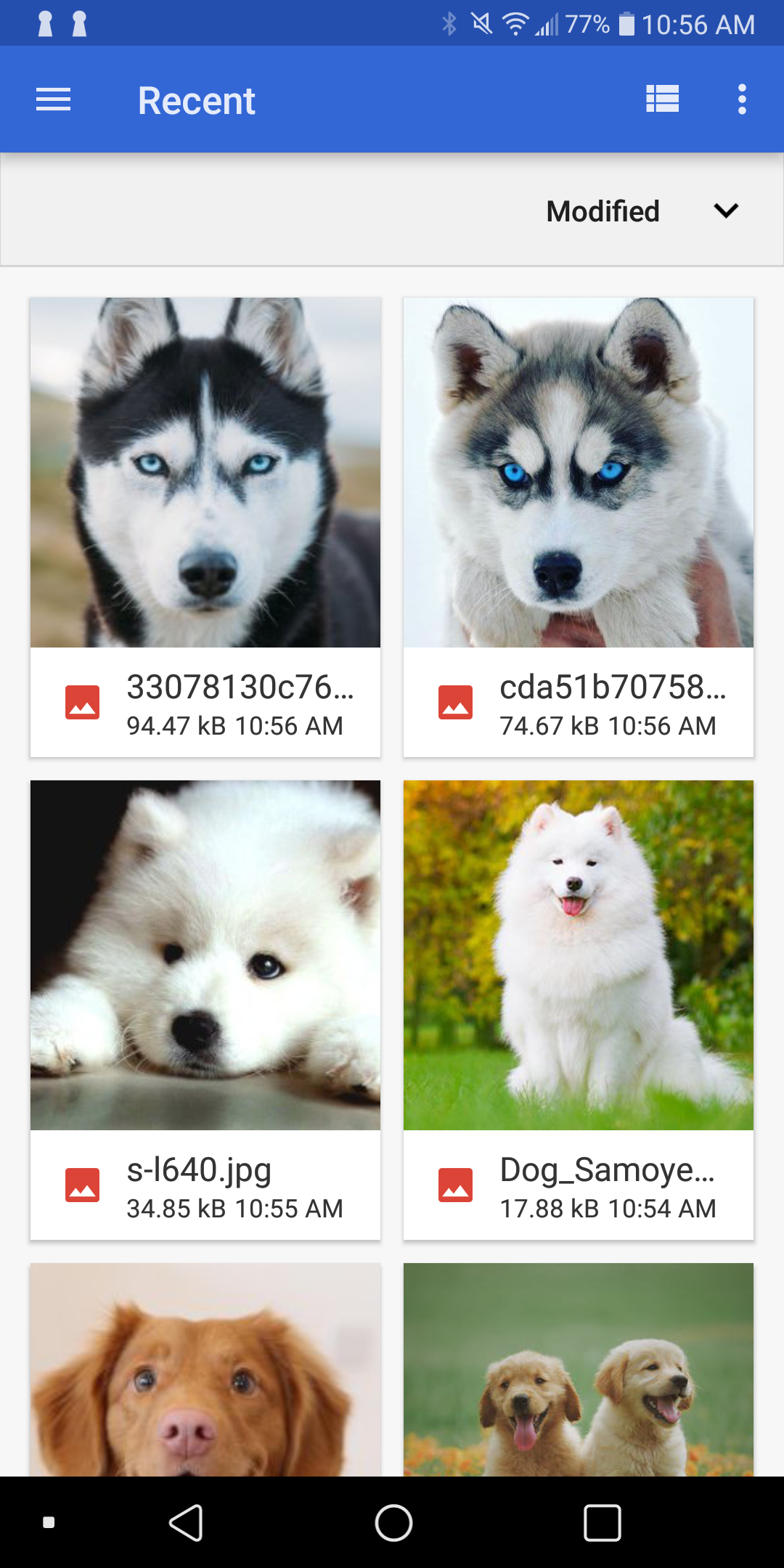
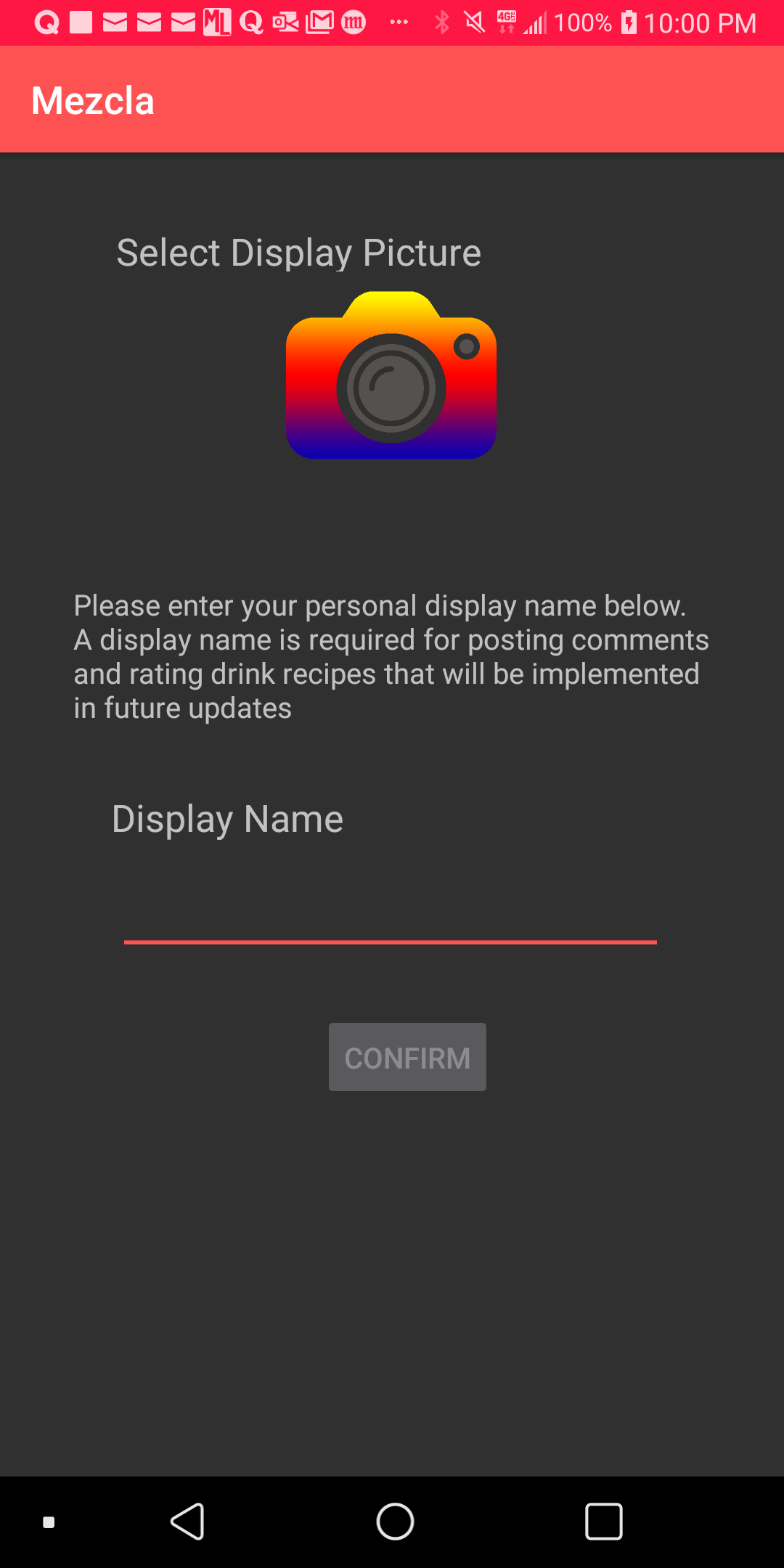


The login page will display an error message near the bottom of the screen when the user has entered an invalid email address. For the registration and login process, we have utilized Firebase Authentication to store, retrieve and verify customers’ data such as email address and password. The password of the user is not disclosed to anybody as it will be hashed by Firebase during the registration process.



The login page will display an error message at the bottom to users that have entered an unregistered account.

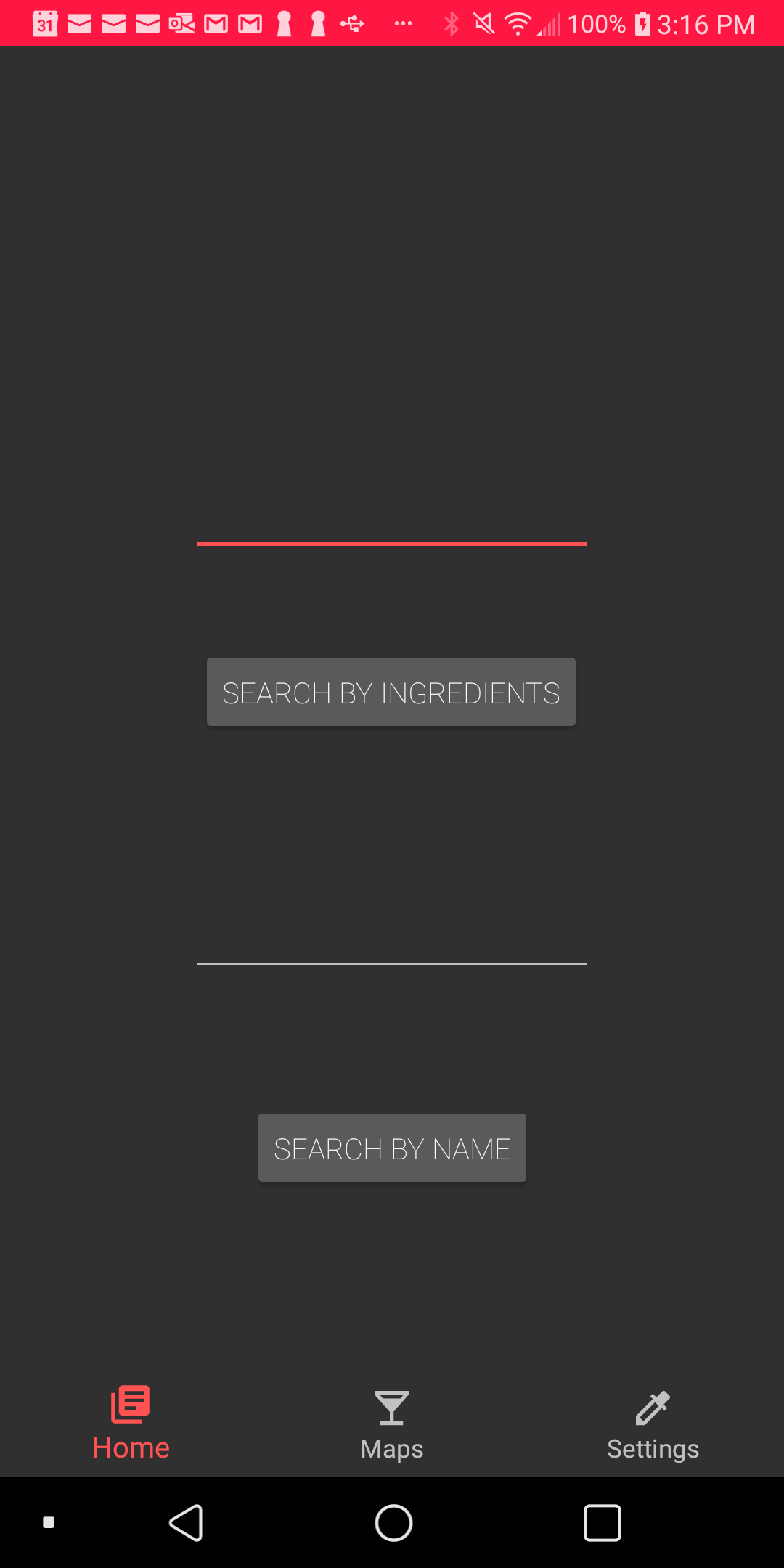
## **Creation of Profile**



After the user has successfully created an account, they will be redirected to the page shown in the left screenshot. They are able to upload their own unique profile picture by simply click on the “camera icon”, which will redirect them to their phone storage of images. After they have selected their ideal display picture, they can type their display name in the field provided.

For this implementation, we have used Glide and Firebase Storage. Glide is a fast and efficient image loading library. After loading the image into the application using Glide, it will then be stored in Firebase Storage. Once that is done, the account setup has been completed and the user will be able to use the services of the application.

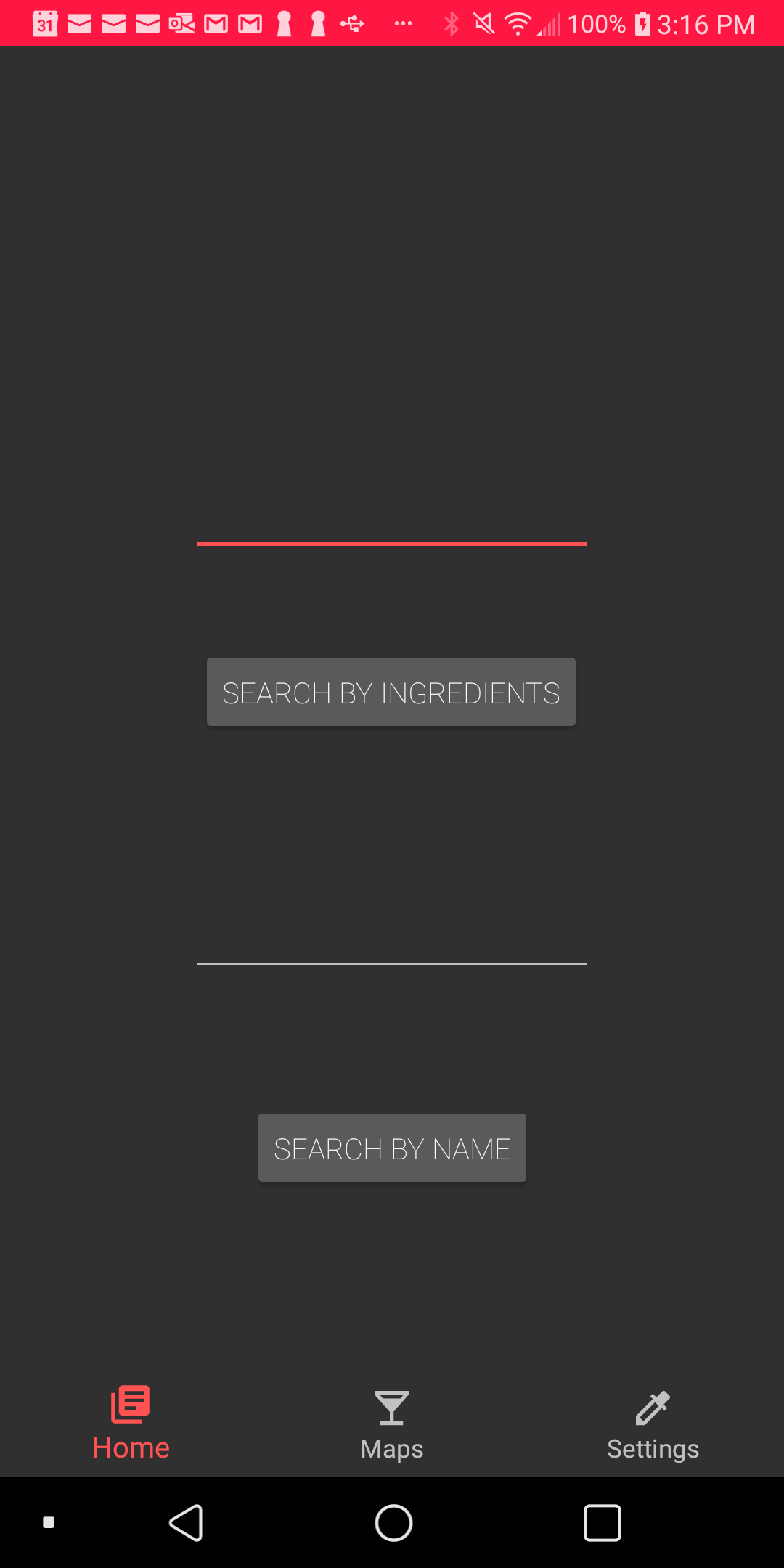
## **Homepage - Usage of Fragments**



As observed in the screenshot shown above, we have used a bottom navigation bar for users to navigate between features of the application. There are three navigation pages, “Home”, “Maps” and “Settings”. More information on each specific navigation page will be explained in the next few pages.

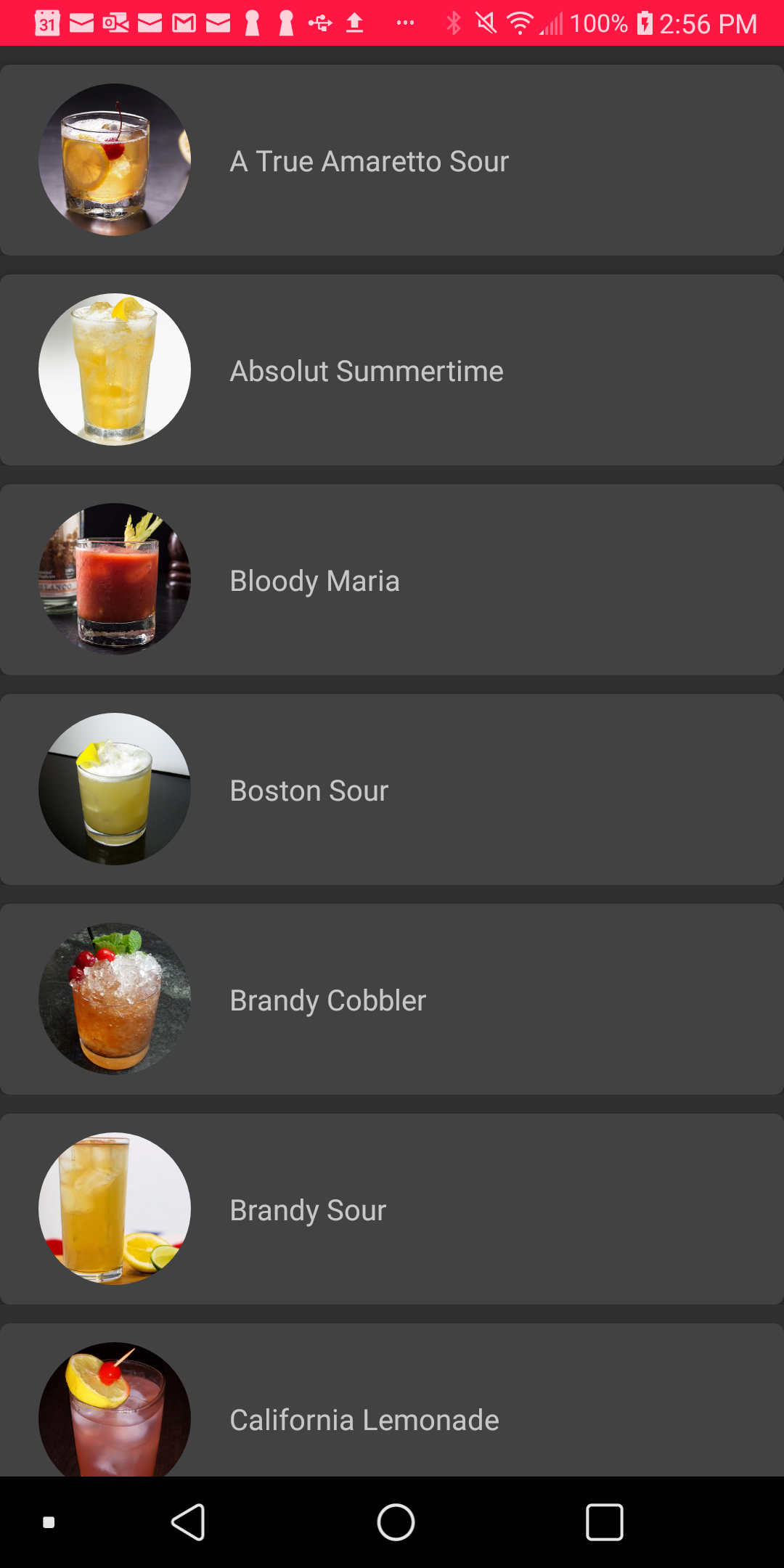
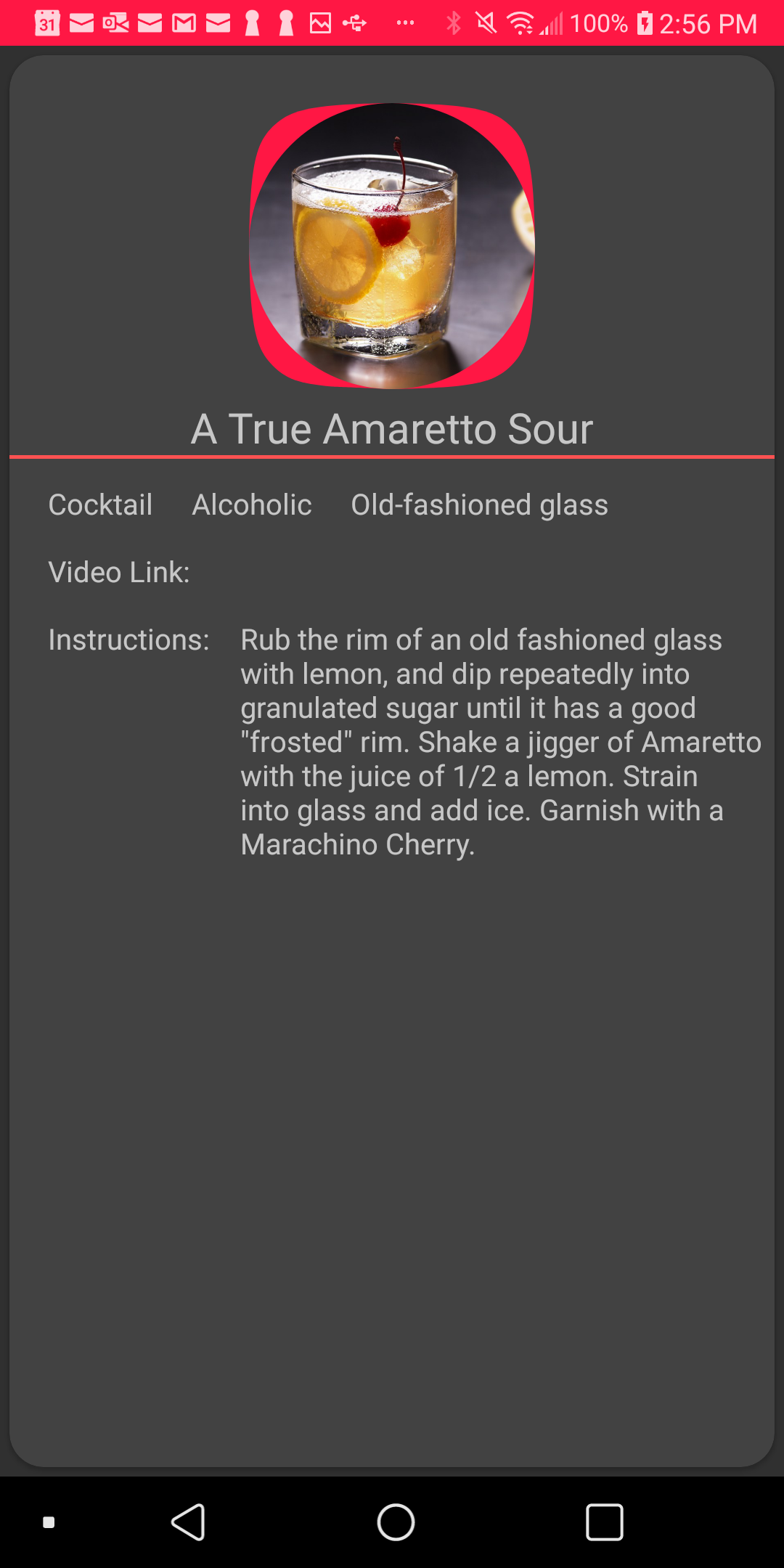
To implement this multi-pane user interface, we have used “Fragments” to combine multiple fragments in a single activity. The “Home”, “Maps” and “Settings” are all fragment implementations.

## **Search Functionalities**

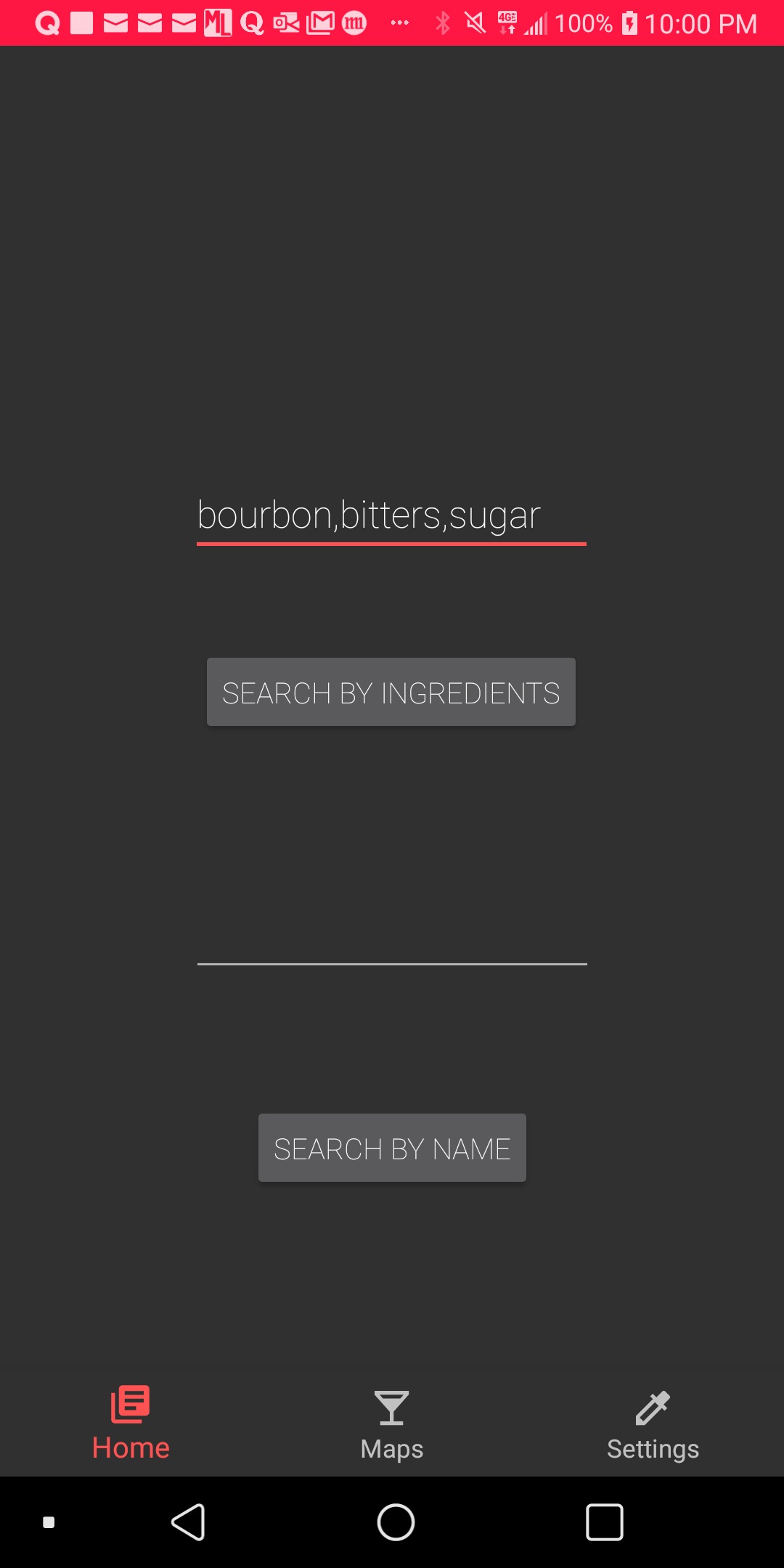


The Home page displays two search functionalities which users are able to either search for cocktails by its ingredients or by its name.

### **Search By Ingredients**

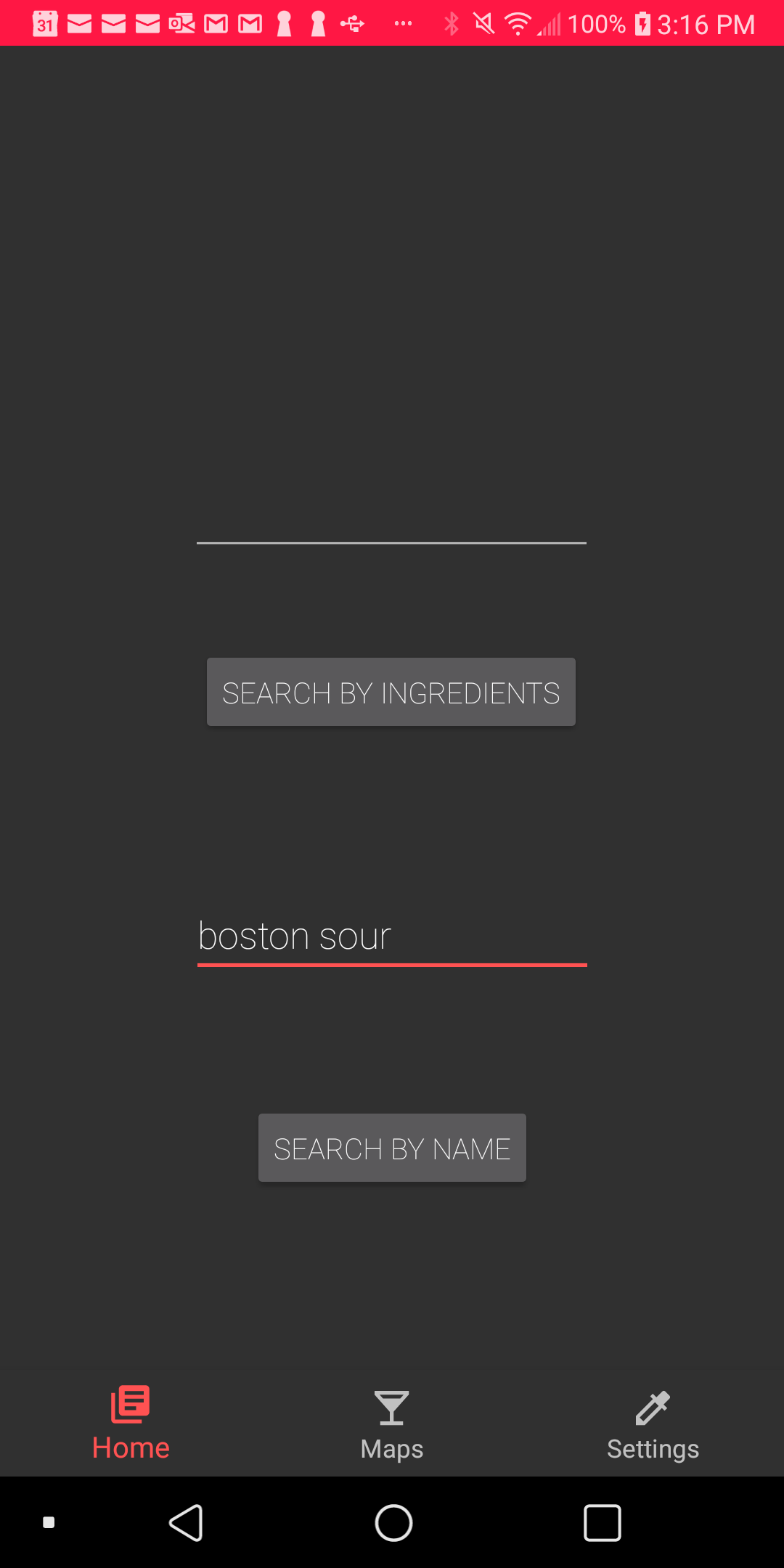
****

An example of how the “Search By Ingredients” search functionality can be used is when the user has an ingredient such as Lemon, but is unsure of what drink he can make - they are able to use the search functionality by entering the ingredient name. The user will then be redirected to their search result page, which will return them with a list of cocktails in an alphabetical order that contain “Lemon” in its recipe. Clicking on any of the cocktails will bring them to the recipe page, where they will be able to know more about the drink such as instructions to construct it.



Another example of how the “Search By Ingredients” search functionality can be used is when the user has multiple ingredients such as Bourbon, bitters and sugar on their table. Similarly, they are able to use the search functionality by entering the ingredients separated by commas. The users will then be redirected to their search result page, which will return them with the “Classic Old-Fashioned” cocktail.

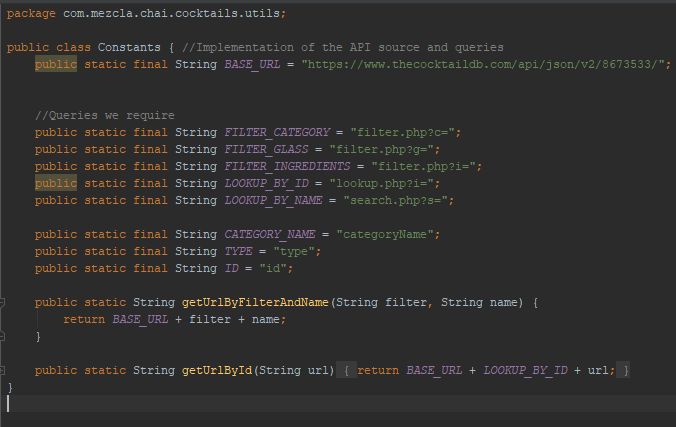
### **Search By Name**



In the screenshot on the left, when the user types in the name of the cocktail and click on “Search By Name” button, Mezcla will redirect users to the page seen in the screenshot on the right. This will be the page that displays the relevant cocktails according to the user search.

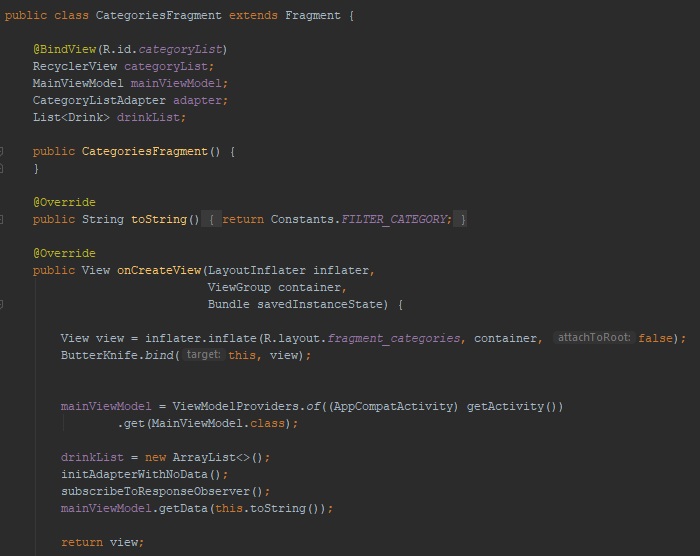
For example, the user would like to construct the “Boston Sour” cocktail but is unsure of its recipe. Therefore, they can search for the cocktail using the “Search By Name” functionality. The user will then be redirected to their search result page, which will return them the cocktail drink that they have searched for. Clicking on the cocktail will bring them to the recipe page, where they will be able to know more about the drink such as instructions to construct it.

The implementation behind the two search functionalities will be described in the next page.



Firstly, as we are using the “TheCocktailDB” API for the database of the cocktails, we will need to pull in the relevant data we need from the API. The above screenshot shows the implementation of the API source and the queries. Although the API has provided a lot more queries, we have only included queries that we require for the assignment.

After we have created the API source, the next step that we have to take is to consume the JSON data that the API has provided. To do this, we have used Retrofit HTTP client in our Android application. It makes the process of retrieving JSON data easy by parsing it into Plain Old Java Objects (POJO). As seen in the screenshot shown above, we have created a data model to parse the JSON data.

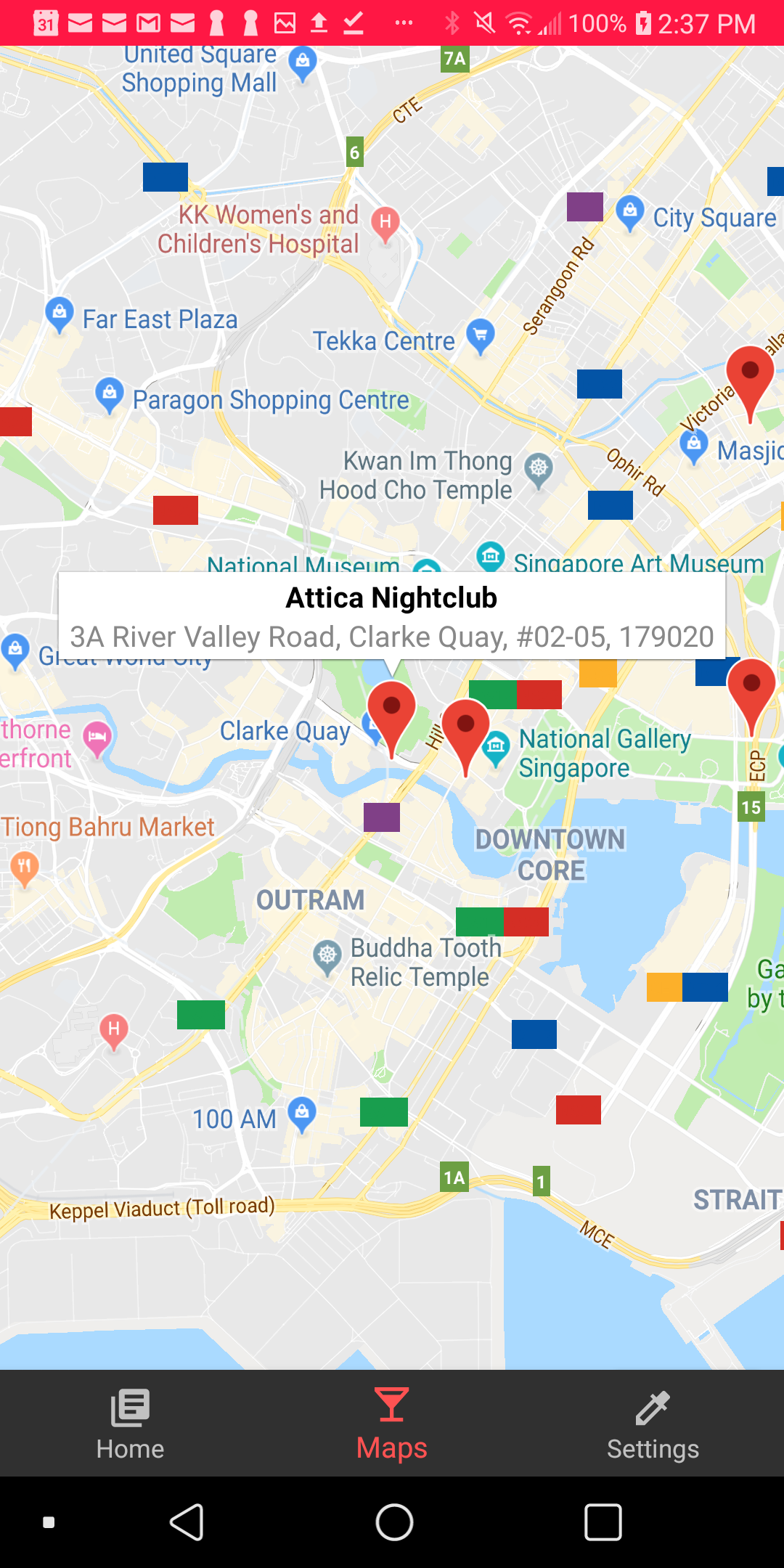


We have also implemented the built-in Observer pattern for the purpose to work hand-in-hand with the RecyclerView to display datasets. The Observer pattern defines one-to-many dependency between objects, so that when one object changes state (new cocktail added into the API), all of its dependents are notified and updated automatically as well. Additionally, we have implemented RecyclerView to display a scrolling list of elements based on large datasets. The reason why we used RecyclerView is because we are retrieving hundreds of cocktail data and that the data will frequently be updated by the API.

## **Google Maps**

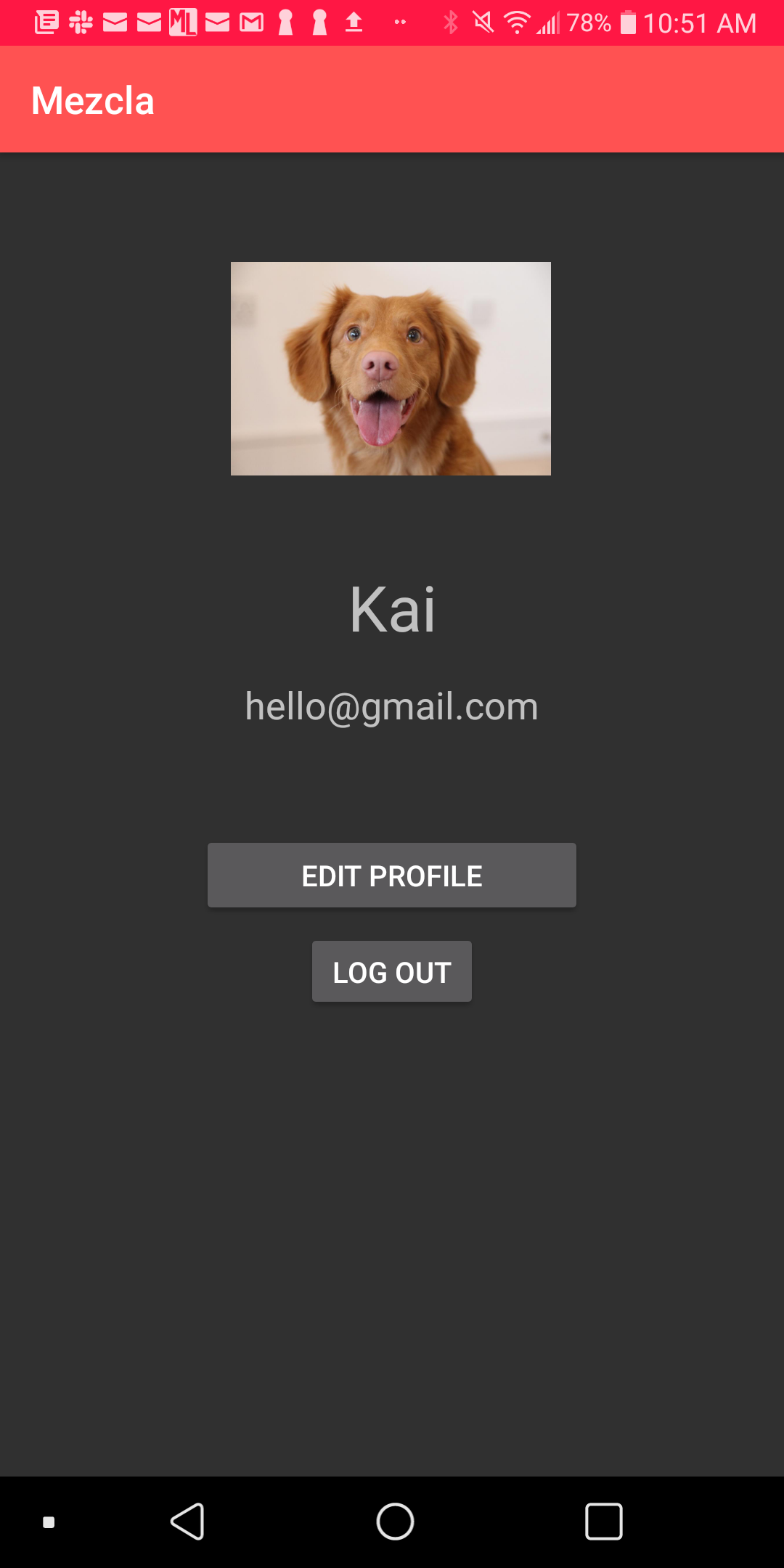


The above screenshot shows a Google Map, where there are markers on the map that locate nightlife locations in Singapore. We have also enabled users to locate their current location based on their phone’s GPS. However, before we could use their phone’s GPS, we will prompt a message to seek their consent.



The above screenshot shows that the user is able to know more about a marker such as the name and address of the nightlife location by clicking on each of the markers on the map.

## **Profile Settings**

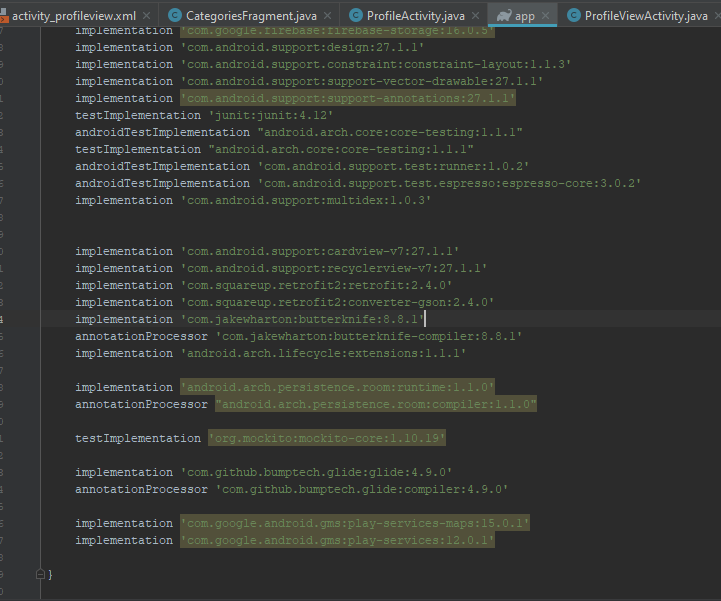


In Profile Settings, the user is able to check their account information such as their display picture, display name and email address. In the situation that they wish to edit their display picture and name, they will be able to do so by clicking on the “Edit Profile” button, which the application will redirect the user to Edit Profile page.

Additionally, there is a Logout button implemented for users to log out of their account. To implement this feature, we have used Firebase’s predefined signOut method.

# **Problems Encountered**

The first problem that we have encountered is gradle issues. As our application uses many libraries and APIs, it is important that all of the libraries and APIs are compatible with each other for the application to work successfully. However, this was not the case as we have faced gradle issues on multiple occasions. There were multiple solutions that we have implemented to fix this error. The first solution was to check on the documentation of the particular library or API if the required gradle files have been added. The second solution was to check for spelling mistakes as sometimes we may have accidentally entered some characters.

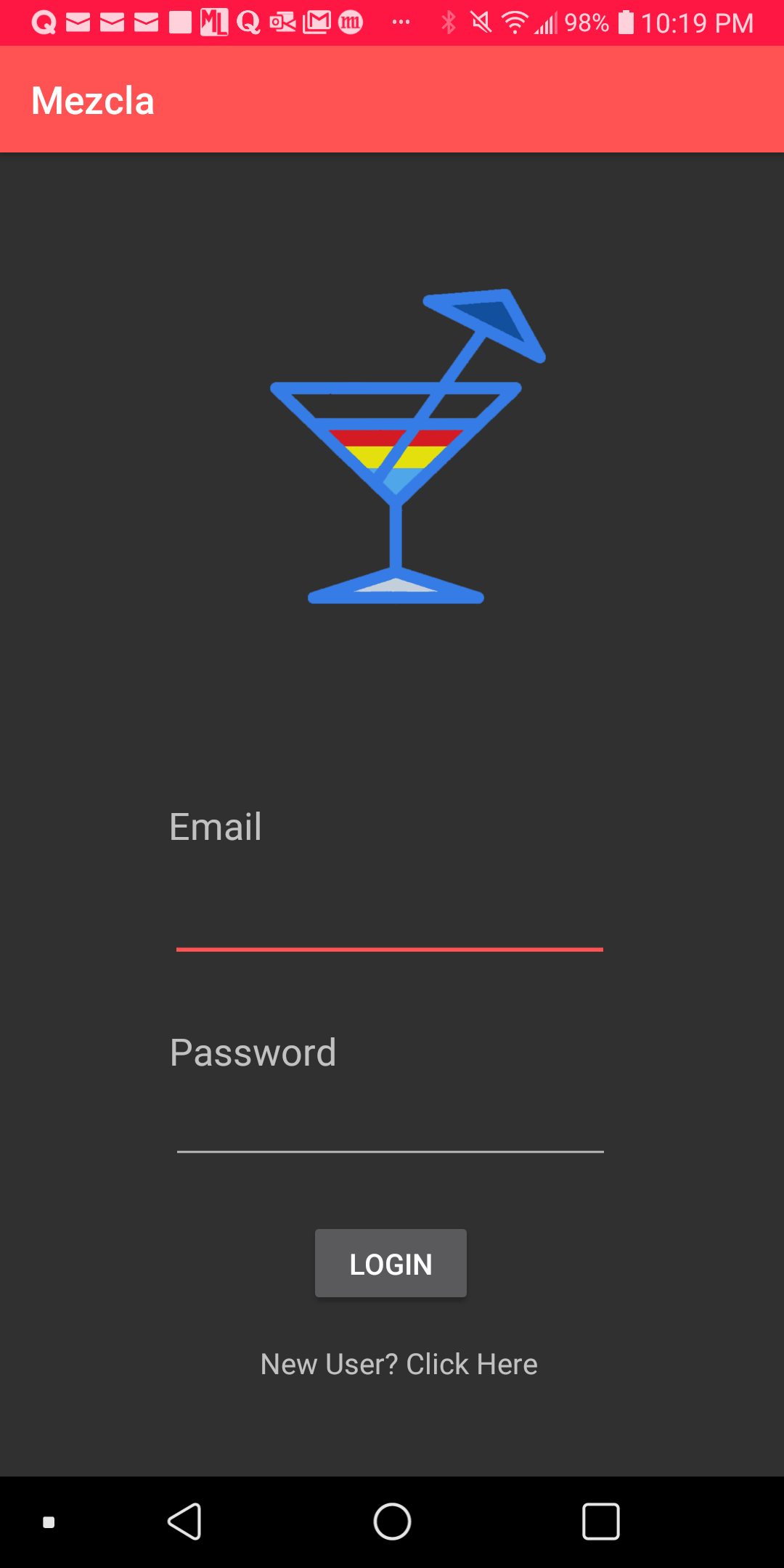


The second problem that we have faced while doing this assignment is when we tried to upload our Android application on Play Store. The error message was that our application was using the package name “com.example”, which was not allowed on Google Play. Therefore, we had to change our package name in order to upload the application. However, when we tried to change our application through refactoring and rebuilding of the project, we encountered multiple issues whereby some of the files were not updated with the new package correctly.

As Android Studio has yet to implement a proper solution for this issue, we used the “Find and Replace” method to ensure that all of the files have been updated with the new package name.

# **Improvements made after first presentation**

## **Night Mode**

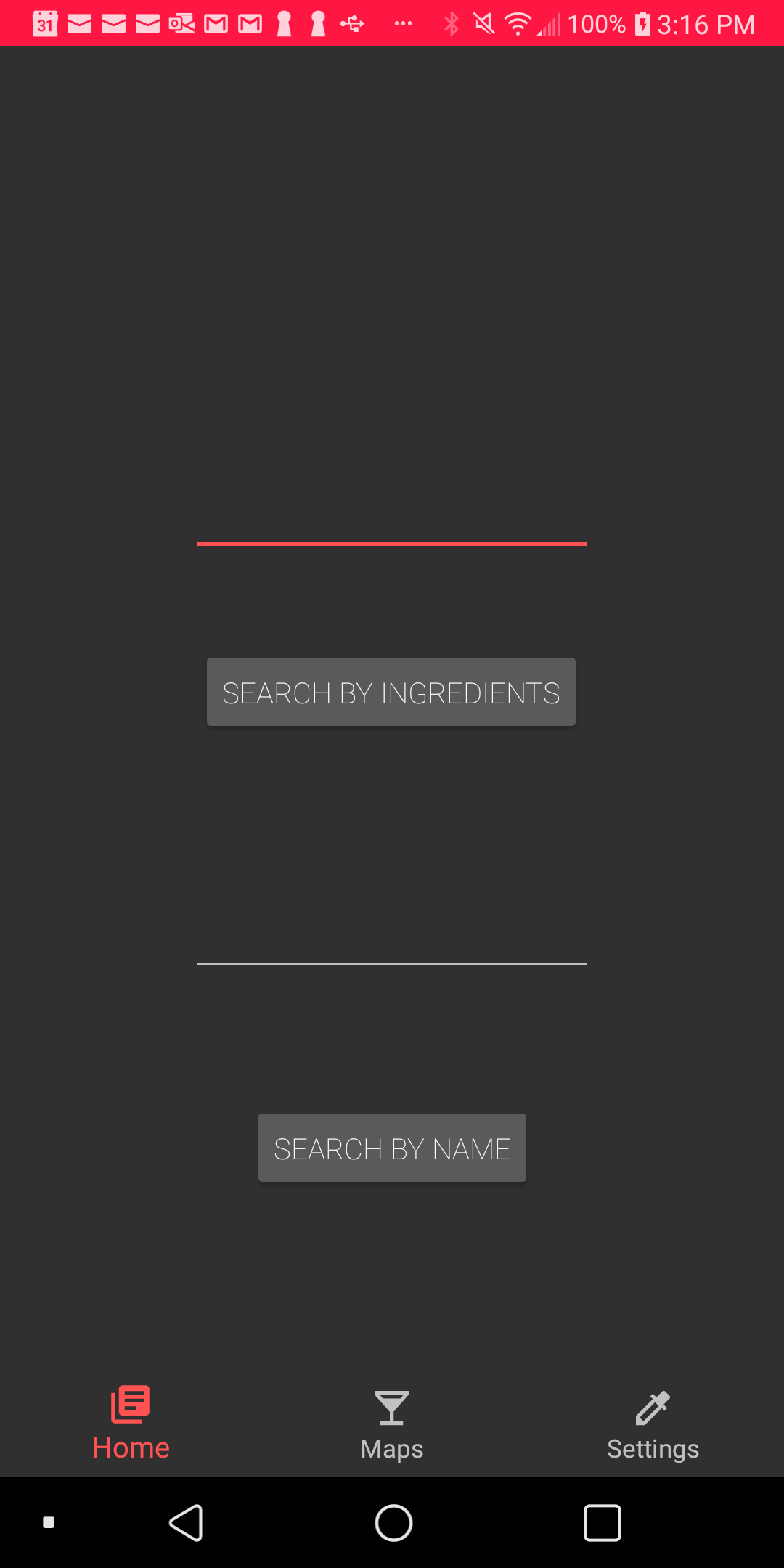


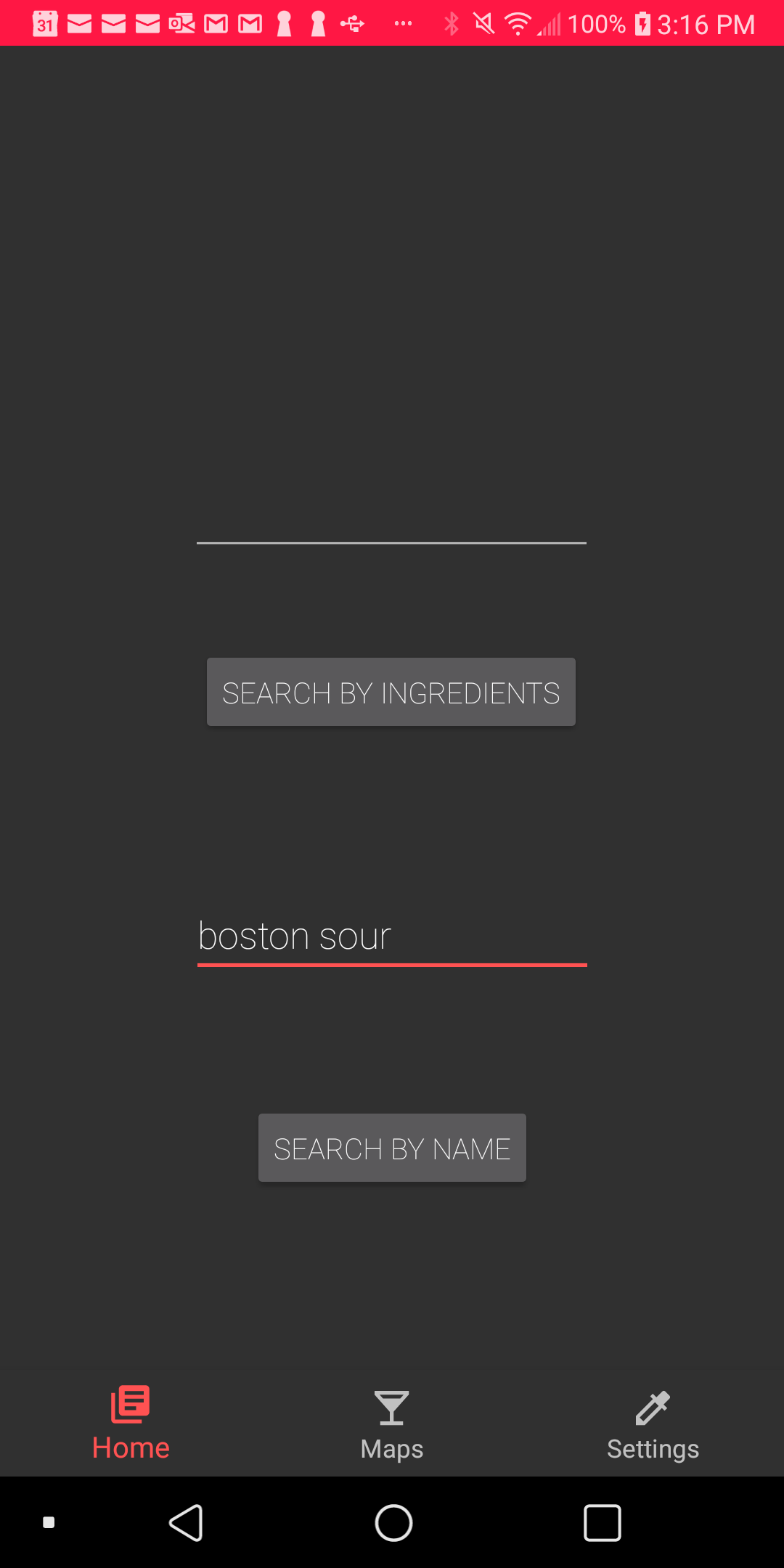
We have implemented the Dark Mode feature whereby the mobile device will automatically adjust the phone display so that it gives off a warmer tone and less blue light. On the left screenshot is the application with dark mode disabled, while on the right screenshot is with the mode enabled.

The use of dark mode provides benefits such as reducing eye strain as it filters out blue light and improve battery life of devices using OLED displays.

## **Addition Search Function**

During the first presentation in class, we have only implemented the “Search By Ingredients” search functionality. Therefore, we have added an additional search function named “Search By Name”, where users are able to search for a particular cocktail recipe by entering the name of the cocktail. This can be observed in the screenshot shown below.





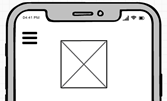
When the user type in the name of the cocktail and click on “Search By Name”, Mezcla will redirect user to another page.



This will be the page that displays the relevant cocktails according to the user search.

# **Design and Ideation Stage (Initial Ideas)**

For design and ideation stage, we have used Balsamiq Mockups 3 software for our wireframing. The following screenshots illustrate our initial ideas of the Mezcla application.



User can access to various features of the application by clicking on the top left corner of the hamburger bar which will display the features along with the username.

A list of relevant cocktail cocktails will be displayed on the screen once the user clicks on the “Search” button. Assuming there are no results of the relevant cocktails, user can click on the “Search Again” text to search for different ingredients.



When the user clicked on a particular cocktail from the search list, Mezcla will display the instructions and ingredients of the cocktail and user are allowed to like or dislike the cocktail.



The most popular cocktails are displayed from the most “likes” by the users and users can click on the “Learn More” button to find out more information of the cocktail. User can see the nightlife locations that are nearest to their current location in Google Map.



The most popular cocktails are displayed from the most “likes” by the users and users can click on the “Learn More” button to find out more information of the cocktail.