Spring 2018, CS5551:Advanced Software Engineering, Department of Computer Science & Electrical Engineering, University of Missouri-Kansas City

[**GitHub URL**](https://github.com/benamreview/CS5551-Team7-Project/tree/master/SourceCode/FixItUpApp)

[**ZenHub URL**](https://github.com/benamreview/CS5551-Team7-Project/tree/master/SourceCode/FixItUpApp#workspaces/cs5551-team7-project-5b903c84ee716b0ad3ad16ec/reports?report=home)

[**Project Video URL**](https://www.youtube.com/watch?v=3nHzO0CH5kM)

**Members: Duy Ho, Sireesha Keesara, Kartheek Katta, Rishitha Bobba**

**FIXITUP**

# PROJECT TITLE : FIXITUPP ANDROID APP

## TEAM

## TEAM MEMBERS:

## 1.DUY HO

## 2.SIREESHA KEESARA

## 3.RISHITHA BOBBA

## 4. KARTHEEK KATTA

**PROJECT GOAL:**

To create an android app which acts as a platform to connect users (who needs help with household, auto, or mechanical services) with the actual technicians who are able to perform and provide those services.

**MOTIVATION:**

Technology has transformed our lives tremendously. When we are in need to fix a problem immediately, browsing a lot of websites for technician’s information and point of contact is a time-consuming process. Hence, it is advised that we have more centralized method of gathering information about these technicians/service providers in order to provide quicker and more convenient satisfaction to customers, which drives the need for developing an app which provides solution to many categories of problems.

**SIGNIFICANCE:**

* FixItUpp provides a single platform where we can fix our problems online. • Our app is unique because it lists explicitlythe availability of the technicians in terms of their current workload. For instance, when a user needs a plumber immediately and if the plumber is currently busy, the app instantly acknowledges it and notifies the user so that the user can opt out for another plumber. • FixItUpp also provides a useful live chat option within the app where the user can communicate with the technician about the problem and other related matter.

**OBJECTIVES:**

* The main objective of "FixItUpp" is to develop an android application where the user enters his/her category of issues(either in electricity, vehicle, pipe system, household appliances) and the comprehensive list of technicians and their expertise will be displayed afterwards together with all the appropriate information. In addition, the user can also identify the location of each technician on Google Maps. We have an option called "availability" where the user can see the approximate waiting time for a particular technician. Moreover, the user will be able to rate and review each technician. As far as the problem’s details are concerned, the user will be able to contact the technician directly to ask whether that technician is able to repair it.

**SYSTEM FEATURES:**

* Searching and selecting: Users can search for what they need help with and who they wish to contact first • Payment:Users can choose to pay provider up front if there is already a price. • Reviews and ratings: after each service session, the user will be able to rate the technician based on their satisfaction level. • Maps: if user allows current location tracking, he/she will be able to view nearby technicians on Google Maps. Otherwise, the user can enter a zip code instead. • Text: basic communication to the technicians. This does not guarantee reply since the technician may be working elsewhere. • Waiting time: Shows the current waiting time of a user for a particular service. Should be updated in real time. • Contact details: a webpage or info page for the technician, if available.

## First Increment Report

**Existing Services/ REST API**

Up to the end of this increment, we are not implementing any REST API (APIs that use get and parameters to retrieve information such as Weather Underground and Nutritionix). However, we are using:

**Google Firebase:** as cloud database/storage, and cloud authentication using User Email/ Password. This eliminates the constraints of validating against the database and Firebase has a built-in verification method for itself using FireBaseAuth. **Google Maps for Android SDK:** acts to show users where they are currently located and other technicians' current location as well. This will be useful as it gives the customers/technicians a better picture of waiting time/remaining distance from one another.

## Second Increment Report

Up to the end of this increment, we have done validations for Login and Registration activities for customers/technicians. And done some UI modifications to the application. And whenever a customer logs in and it takes to the customer home page where it displays the list of technicians that are available database. GeoFire: Updates real-time location of customers/technicians to Firebase Database and queries the storage of locations based on distance range and requested location of customer.

## Detail Design of Features

As a result, Increment 1 does not promise to show a lot of tangible features and output due to lack of team communication at initial stages and a need for more efficient workflow. Also, we spent a lot of time discussing design, version control, workflow among team members for future long-term development.

Our Mockup Design would look similar to Uber's platform in that we have sign for Technicians and Customers as well.

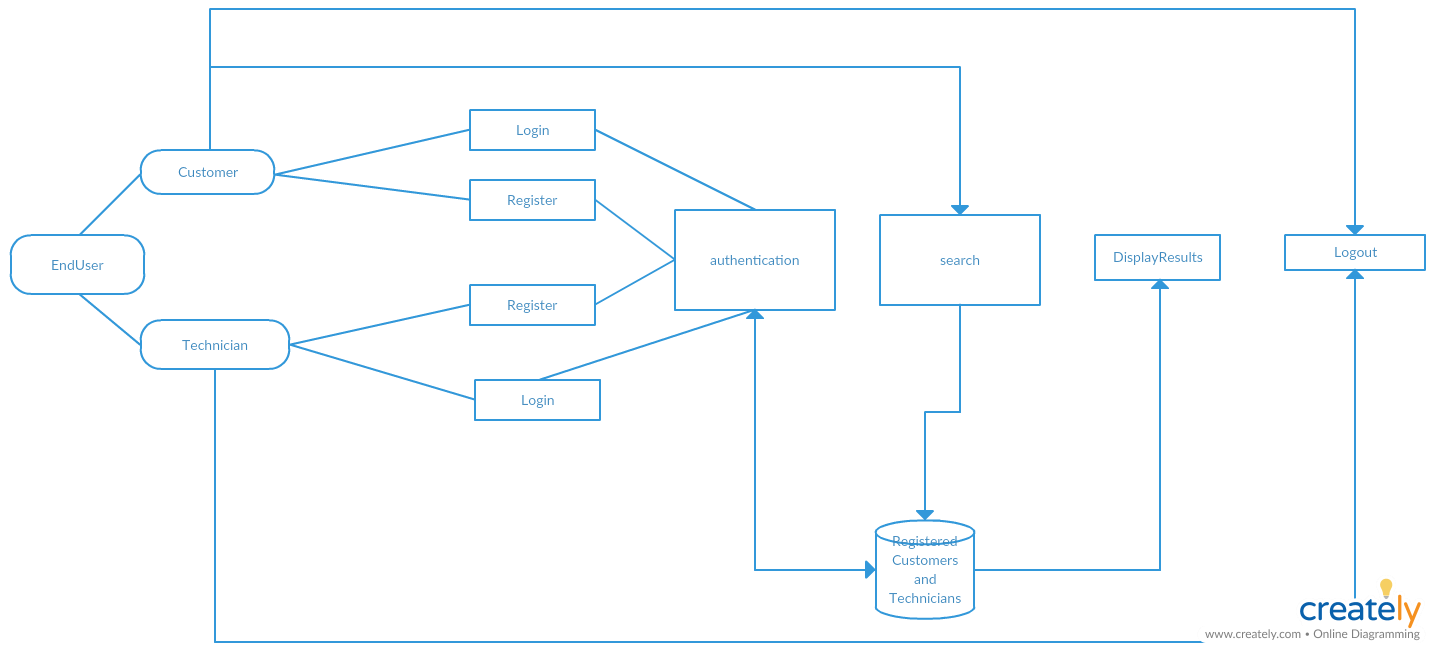
These two targeted users' paths will then diverge significantly in terms of UI design, content, and features:

Customers will be able to search for what they are currently needing help with, enter their current location (or other locations that they want using zip code), and opt for the technician that they see fit. There is an option for payment if the problem (service) is clearly identified and the technician has given a specific quote for that service; if not, they can always pay at the end of their session with the technician (however, this will not be our liability anymore). At the end of each session with the technicians, the customers will also be able to rate the service quality which would then be taken into account for the technician's overall rating as well.

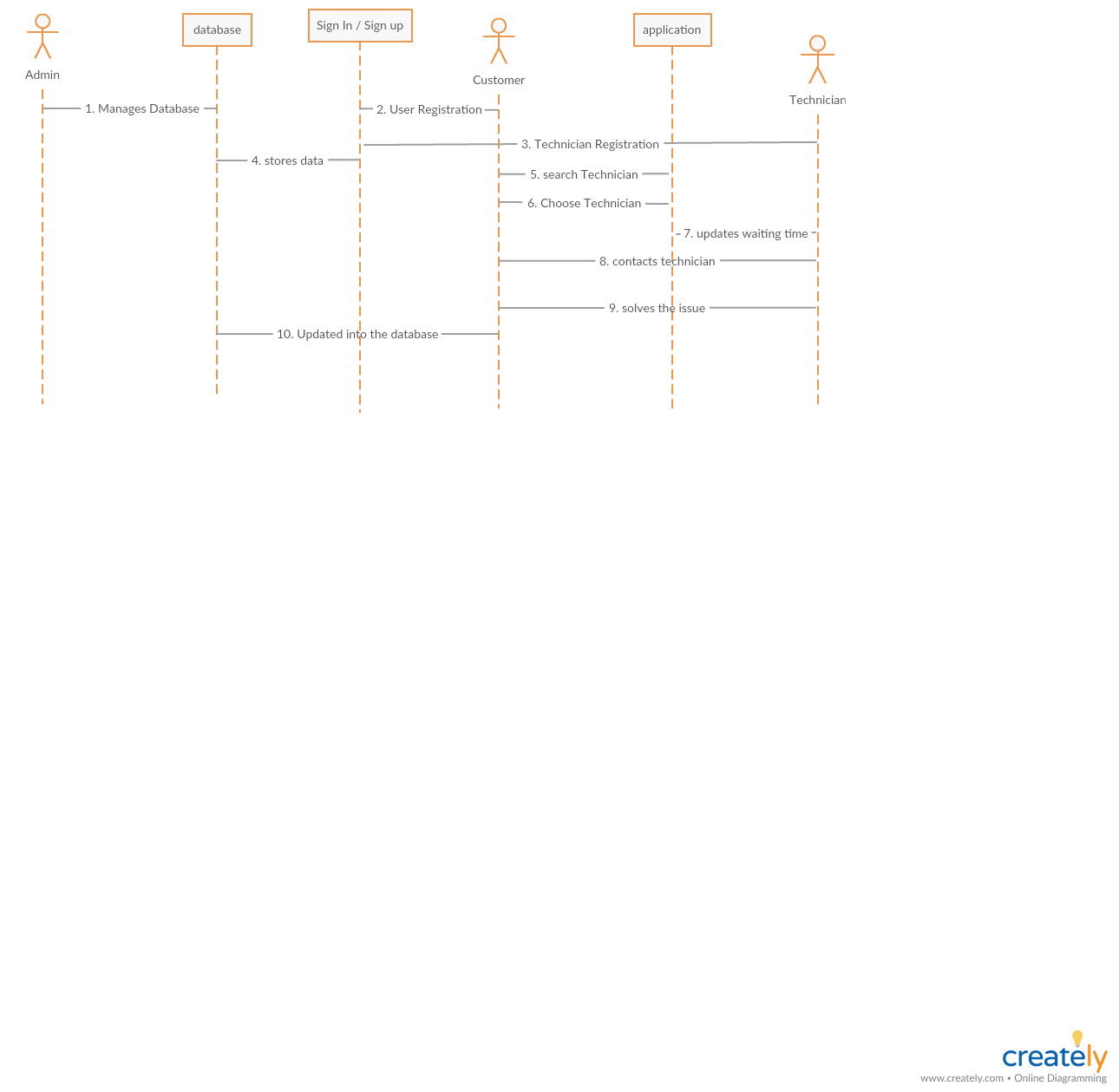
Technicians, on the other hand, would see what services have been requested (in case a technician is experienced in multiple services/ has multiple fields of expertise), what is the current queue is a particular service, and finally set his/her current waiting time for the customers to see. For the waiting time, we may ask the technician to enter a default waiting time per person so that the system can be show the customers its estimated waiting time in case the technician does not enter his own time estimation. Technicians may turn on and off his current location services in order for the customers to have a better idea of his current progress (ON is recommended for technicians).

Moreover, to have a better idea, here is the four different views of the FixItUp Application:

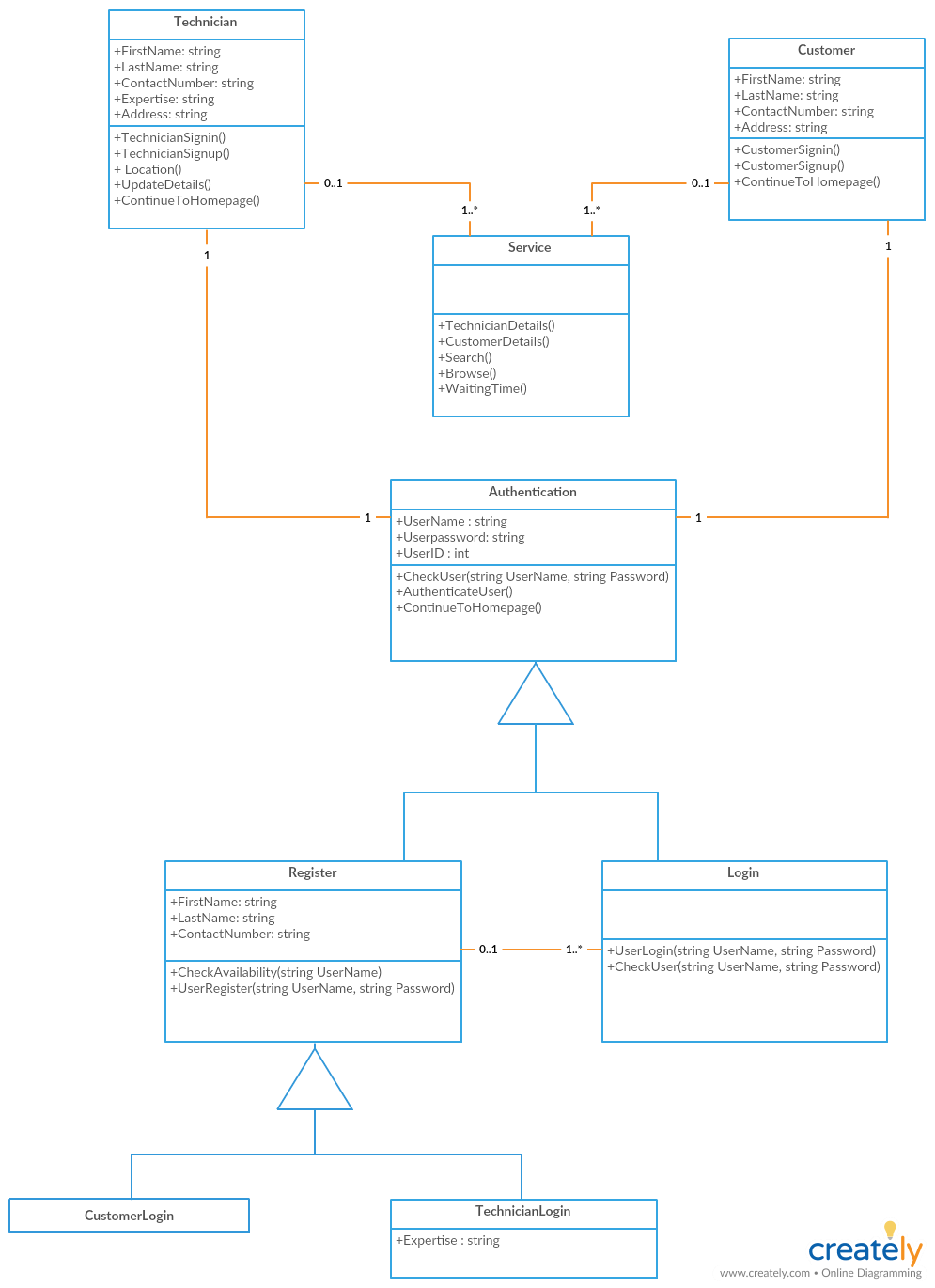
**Architecture Diagram:**



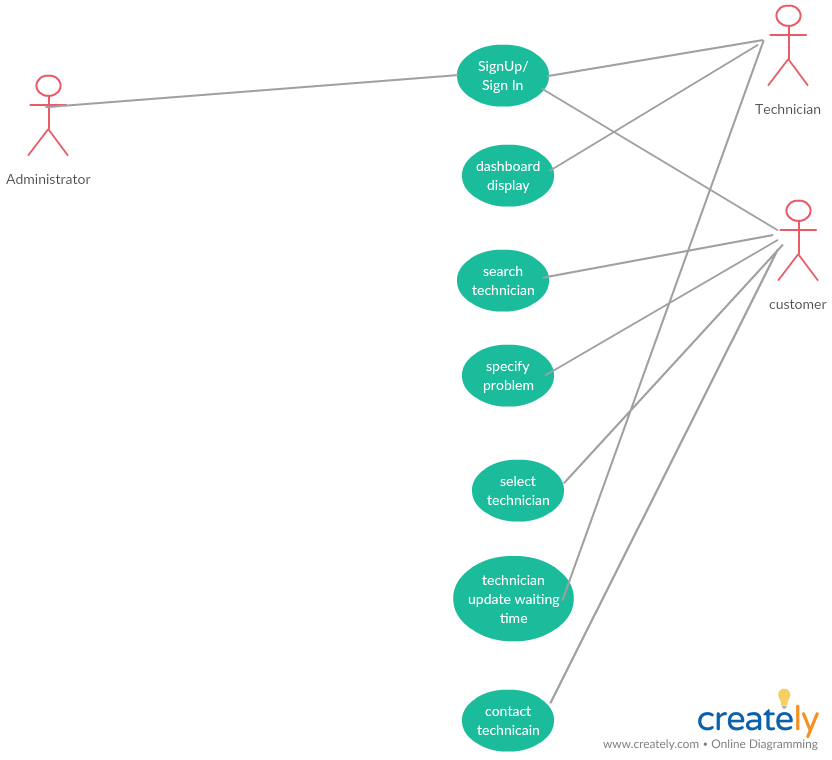
**Sequence Diagram:**



**Class Diagram:**



**Use case Diagram:**



**Testing:**

Unit Testing is not implemented for this increment due to time constraint and the unfamiliarity with the JUnit testing tool. Nonetheless, the team members did a lot of debugging to make sure that the core functions are executing properly without side effects.

**Implementation:**

The form of Application we are building is native application.

**Platform:**

Android

**Tool:**

Android Studio version 3.1.x

**Approach:**

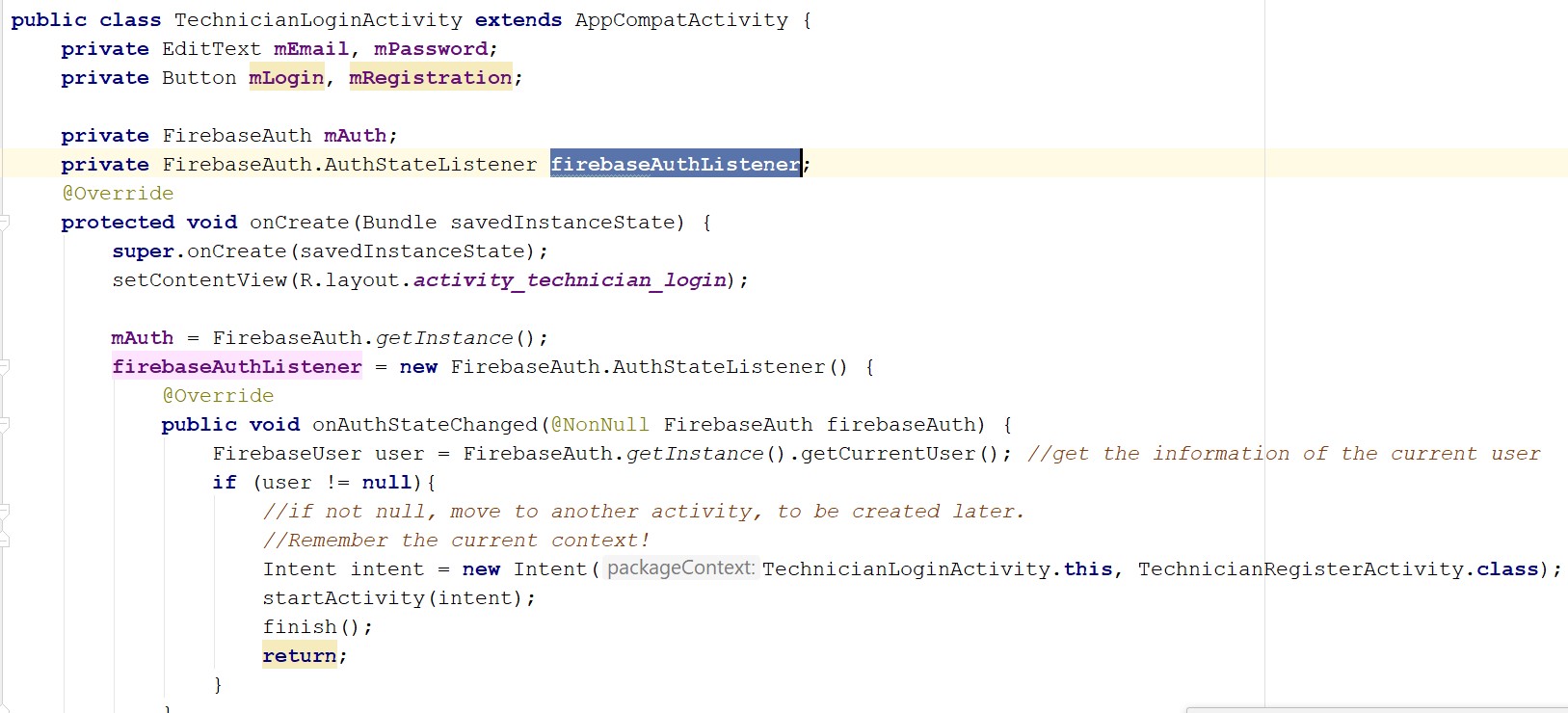
For this increment, we have created 3 main pages that encompasses 2 main features: authentication and registration.

For both purposes, we are using the provided service from Google Firebase to do the identity validation and storage. We include the firebase implementation in the gradle in order for Firebase source code and modules to work in Android Studio. Afterwards, we created two buttons, each for customers and technicians. If the user is a customer, then he/she would click on that and it will direct to another customer-related activity such as customerlogin activity. The same process applies to the technicians.

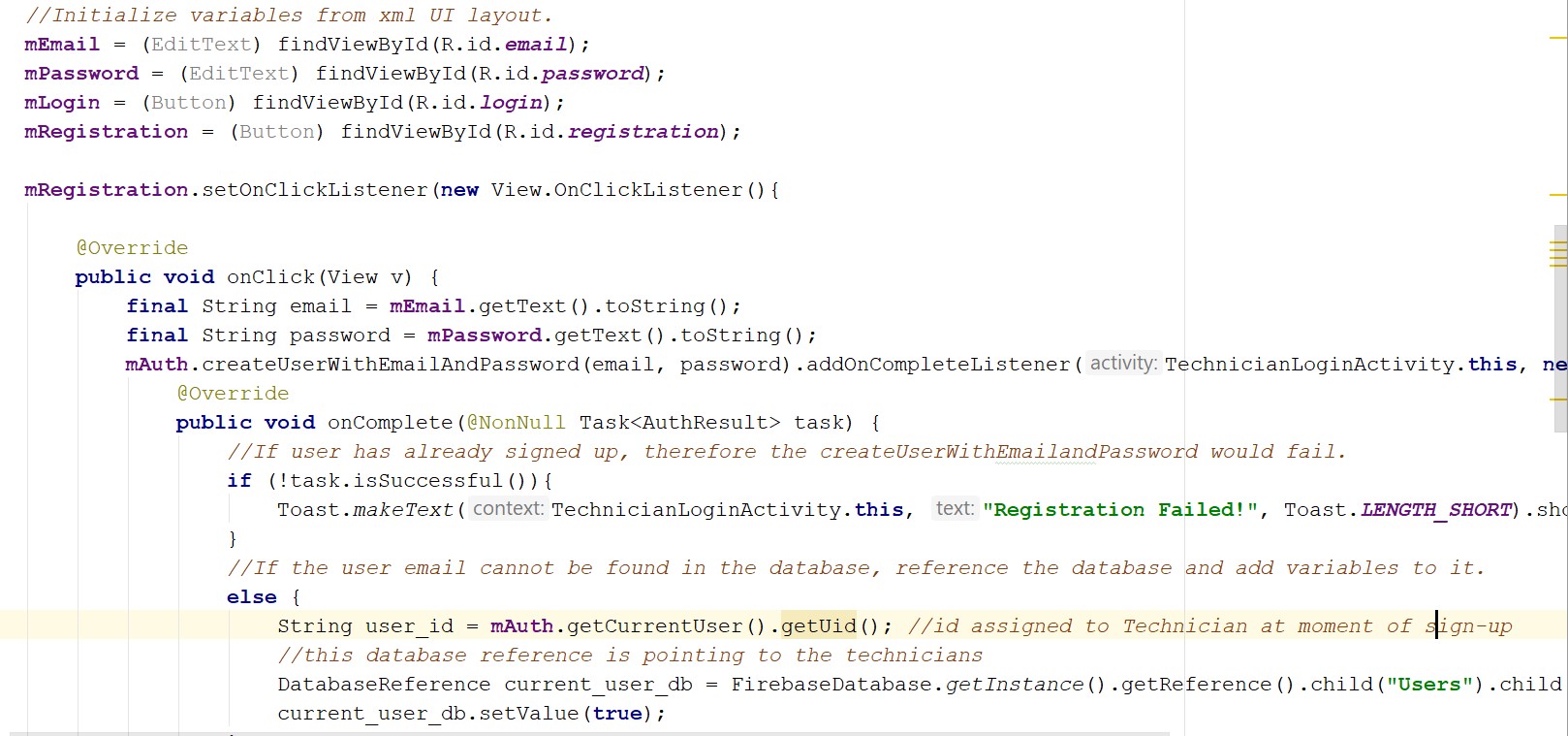
At the end of this increment, we have accomplished designing the paths for customers and technicians by not mixing them together and use identity validation. Instead, we let them diverge to their own activities. The same strategy is used in the Firebase database by creating 2 separate columns such as "Technicians" and "Customers" under "Users" column to explicitly distinguish between customers and technicians.

**Screenshots:**

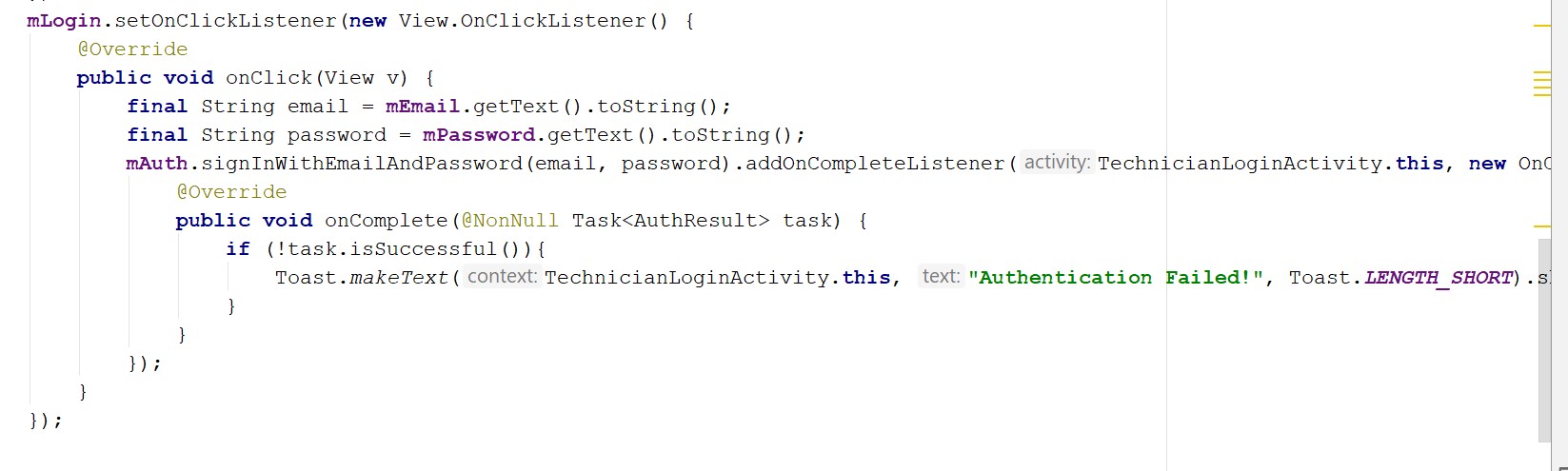
*This is the Firebase authentication in the TechnicianLoginActivity. The authentication is executed directly through the FirebaseAuth and FirebaseAuthListener. When it returns user's valid info (or invalid), we will direct the user to the right activity*



*This will be triggered when user clicks the registration button. Afterwards, the user will be able to enter the credentials which will then be validated against the Firebase authentication directory. If not exists, it will add the user id to the database.*

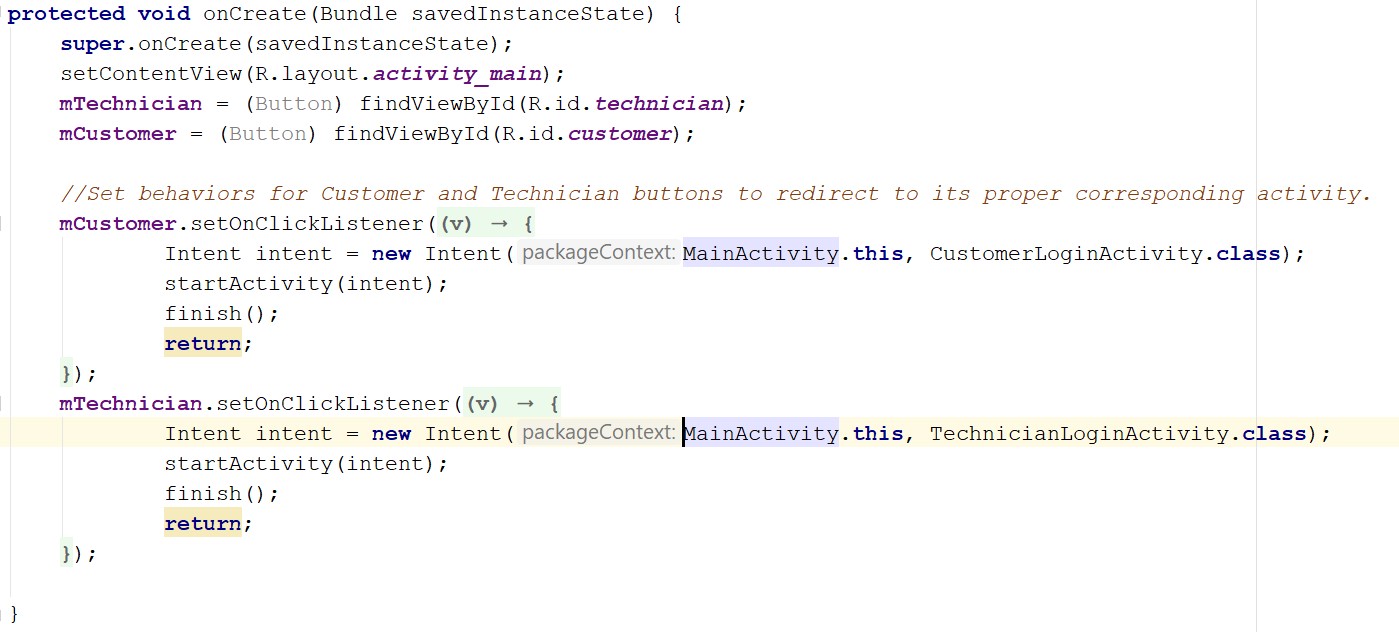


*The same logic happens in Login mechanism. However, for Login, we don't need to do anything if login is successful because FirebaseAuthListener will take care of valid info/state change.*

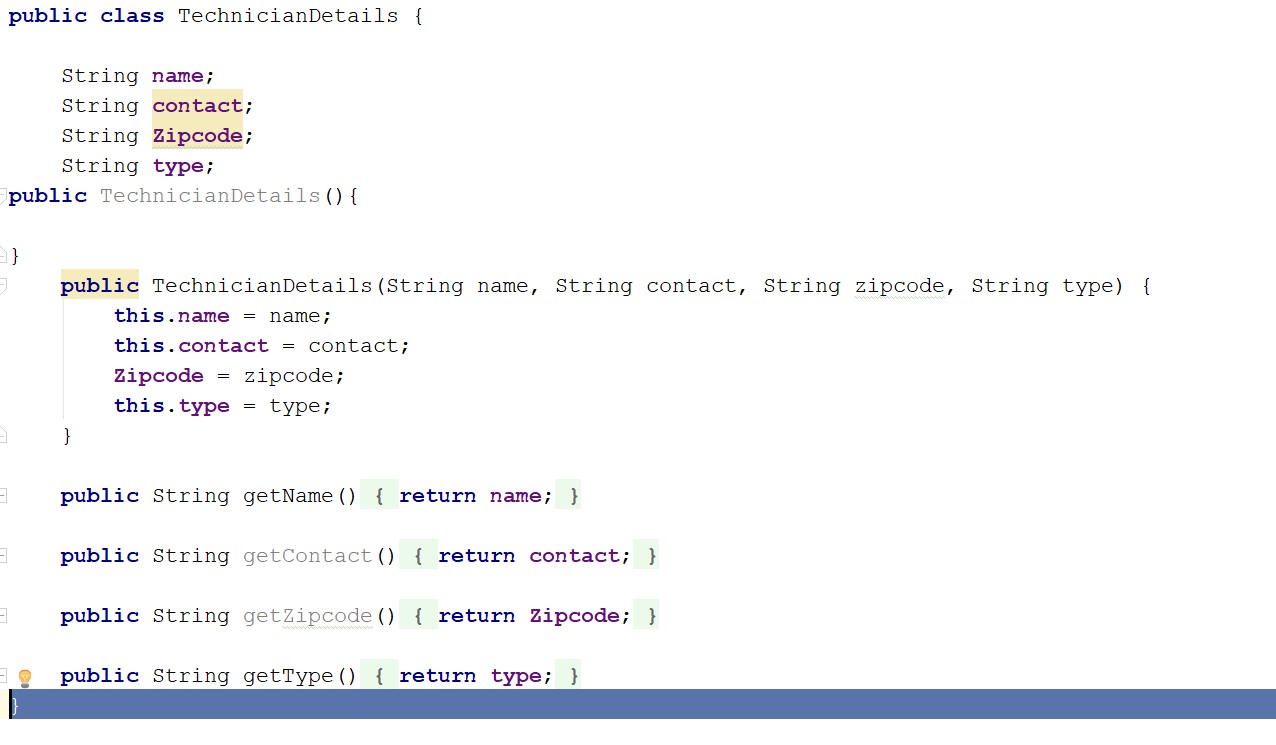


*This is the landing page of the app where the user will be able to identify himself/herself as a customer or a technician.*

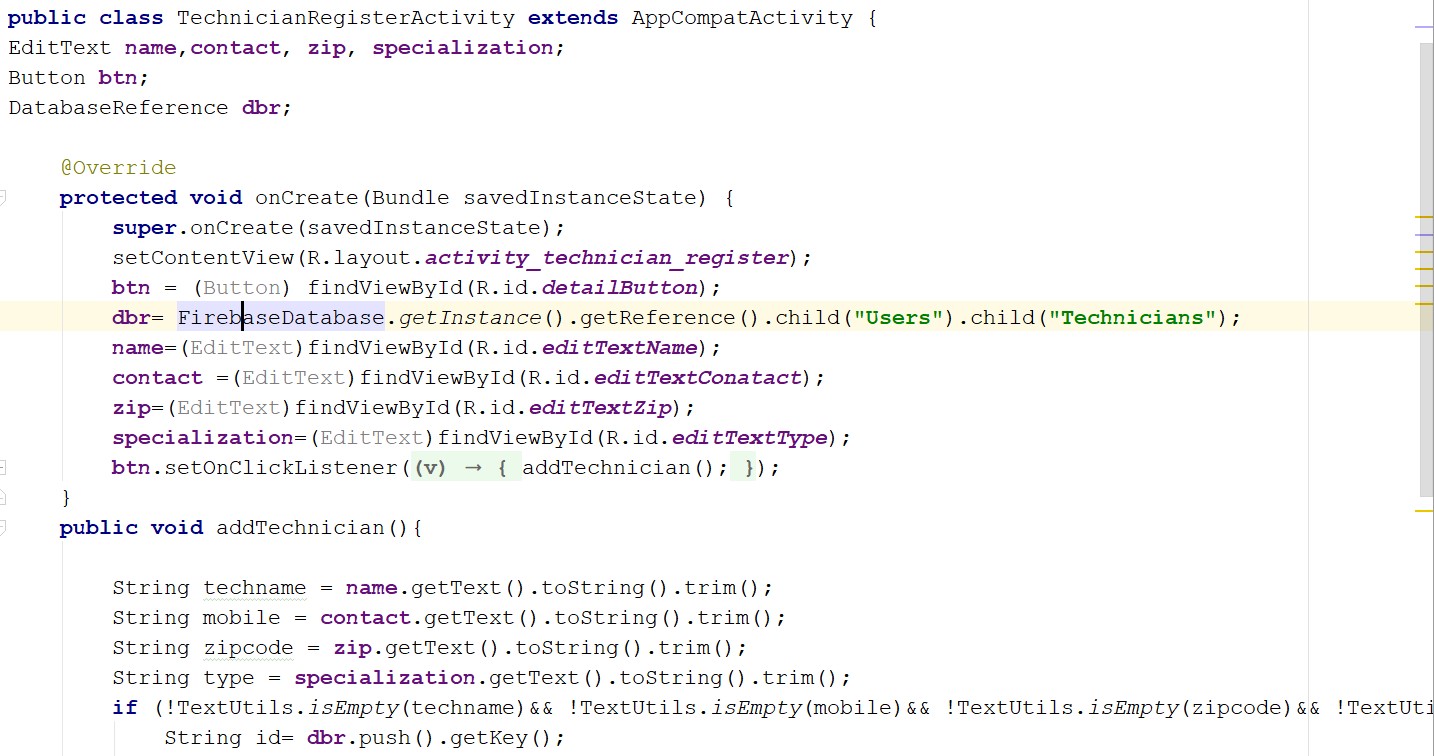
*Upon clicking the buttons, they will be directed to the right path/activity/login pages.*



*To simplify the process of pushing and adding information to Firebase Database for storage, we packaged our users' info in a class which contains various attributes/fields so that the whole object can be delivered.*



*To further continue the logic of the previous screenshot, this activity will be used to process the input from user, package it into a TechnicianDetails object, and pass it to the database. As of now, the ID is not consistent because the ID that the user initially logins is stored in a different key than the ID that is packaged here. It should be the same. This will be fixed in Increment 2*



*This is how the User IDs are stored in the FireBase real-time database. Whenever there is a registration, it will automatically update with the user id and their relative detailed information package.*



**Deployment:**

Our project is up-to-date with our GitHub as far as the most stable release is concerned. Other ongoing development and features need to be further tested, analyzed, and discussed. They are not yet published and are intended to be released in the upcoming increments.

[GitHub Repository URL](https://github.com/benamreview/CS5551-Team7-Project/)

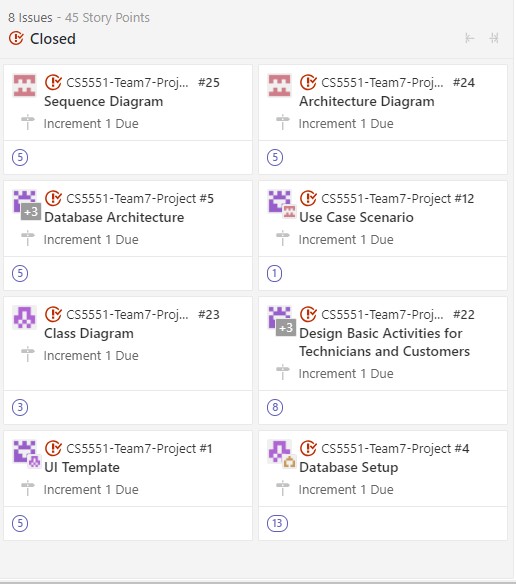
[Wiki Link](https://github.com/benamreview/CS5551-Team7-Project/wiki/)

## Project Management

**Implementation Status Report**

**Work Completed**

*Description:* There are 8 main issues that we have completed for this increment:



*Responsiblity:*

Database Architecture, Design, and Setup: Duy and Sireesha: 5 hours Code Design and Implementation: Duy, Sireesha, Karthik, Rishitha - 15 hours

Diagrams:

Class and Use case Diagrams: Karthik - 5 hours

Sequence and Architecture Diagrams: Rishita - 5 hours Wiki, Documentation, and Report: Duy - 4 hours

**Work to be Completed :**

For increment 2: We will expand our scope to:

Populate and retrieve data in Firebase database

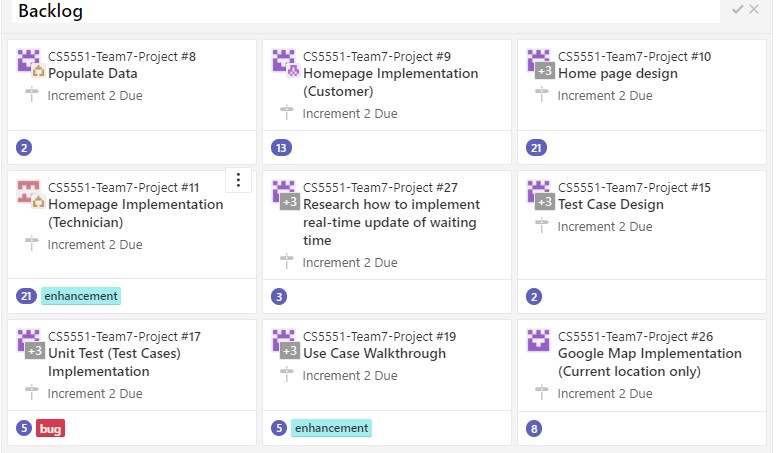
Provide homepages for both customer and technicians:

For Customer: search bar, technician information, rating, and request a service

For Technician: home page, update information, update status, and current request queue.

Provide a map of current location for both customers and technicians: Customers will be able to see their current location, Technicians will be able to see their location.

\*\*NOTE: For increment 2: Customers and Technicians may not be able to see each other's location yet. \*\*



## Bibliography

[Firebase Tutorial](https://firebase.google.com/docs/android/setup?authuser=0)

[Android Tutorial](https://html2pdf.com/files/2r6iwaonctbca64n/o_1ctjkbr6n1qrp16281vvub2t1h1ha/www.simcoder.com)

[Google Maps Tutorial](https://www.javatips.net/api/LOST-master/lost/src/main/java/com/mapzen/android/lost/api/LocationCallback.java)

[Android Resources](https://developer.android.com/reference/android/content/res/Resources)

# Second Increment Report

## Existing Services/ REST API

Up to the end of this increment, we are not implementing any REST API (APIs that use get and parameters to retrieve information such as Weather Underground and Nutritionix). However, we are using:

Google Firebase: as cloud database/storage, and cloud authentication using User Email/ Password. This eliminates the constraints of validating against the database and Firebase has a built-in verification method for itself using FireBaseAuth

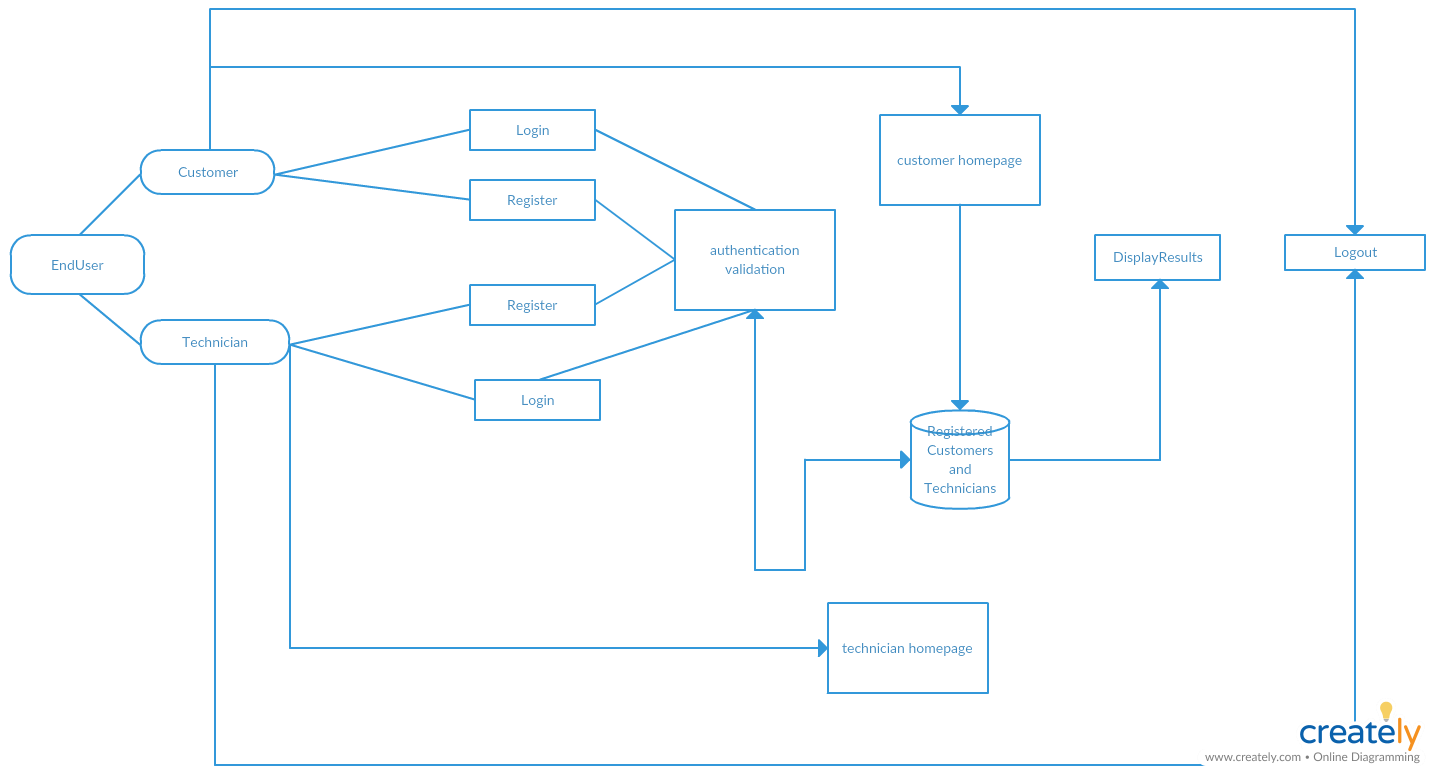
Google Maps for Android SDK: acts to show users where they are currently located and other technicians' current location as well. This will be useful as it gives the customers/technicians a better picture of waiting time/remaining distance from one another.

GeoFire: Updates real-time location of customers/technicians to Firebase Database and queries the storage of locations based on distance range and requested location of customer.

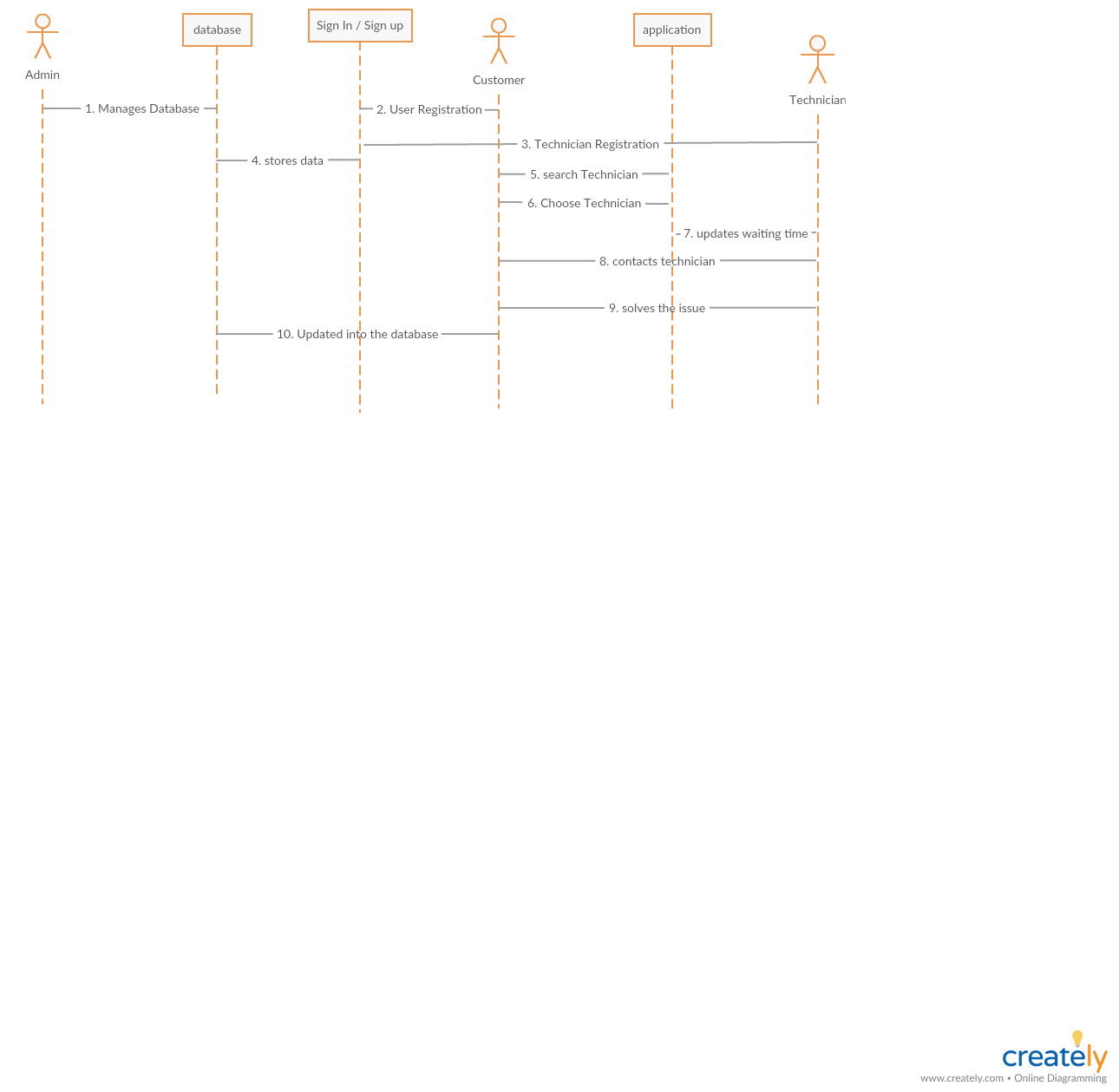
## Detail Design of Features

Here is the four different views of the FixItUp Application:

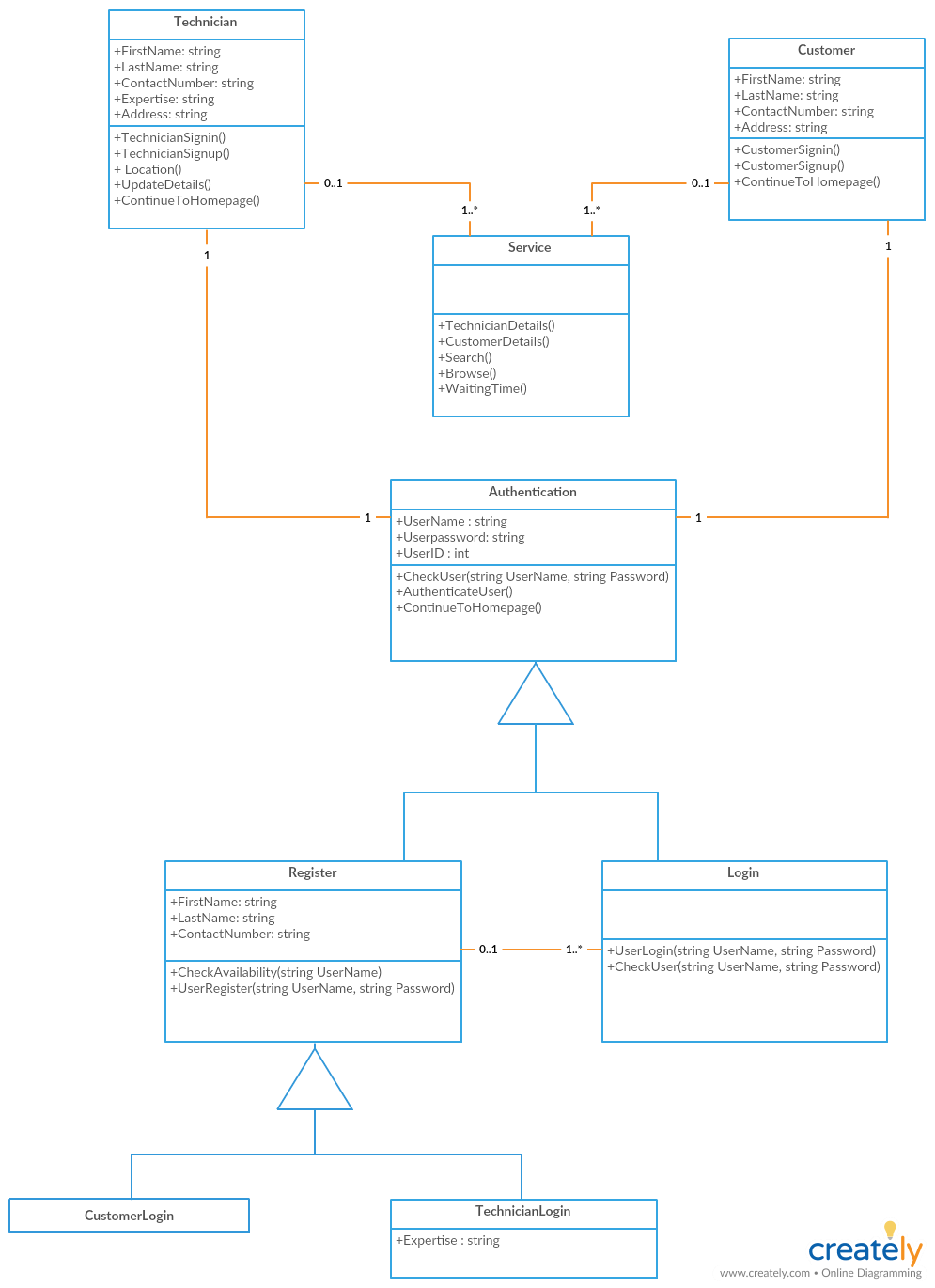
**Architecture Diagram:**



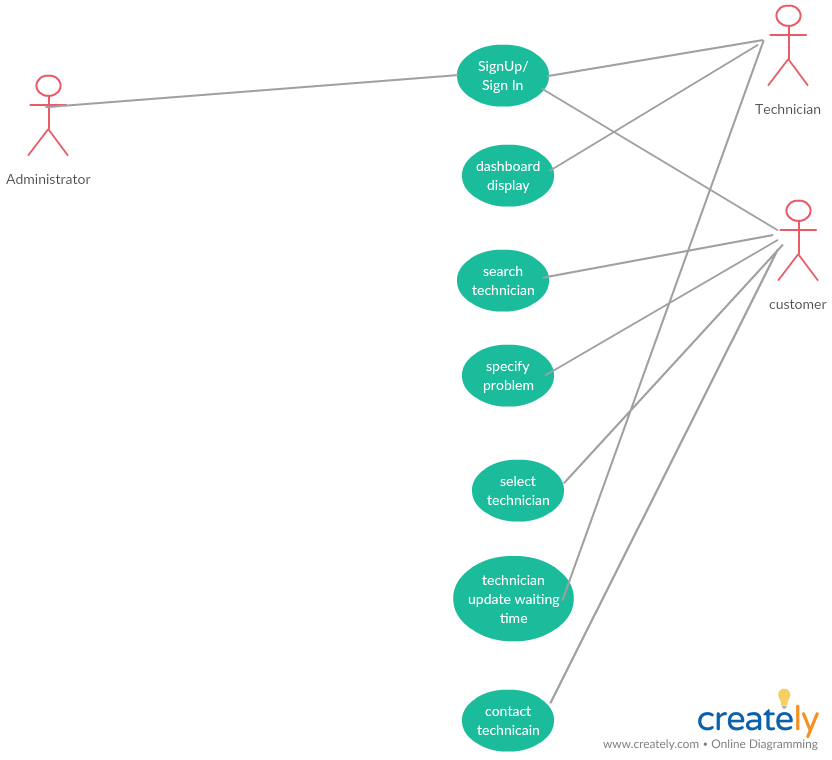
**Sequence Diagram:**



**Class Diagram:**



**Use case Diagram:**



**Testing:**

Unit Testing is not implemented for this increment due to time constraint and the unfamiliarity with the JUnit testing tool. Nonetheless, the team members did a lot of debugging to make sure that the core functions are executing properly without side effects.

**Implementation:**

The form of Application we are building is native application.

**Platform:**

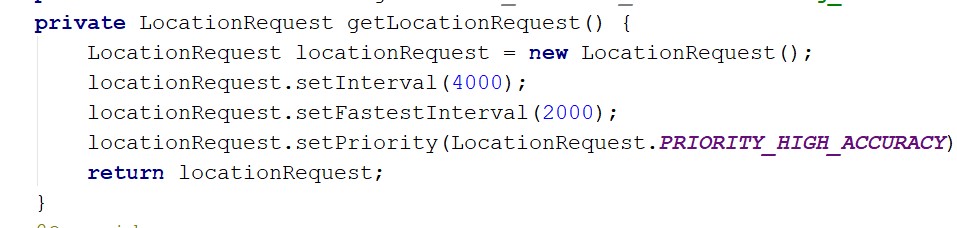
Android

**Tool:**

Android Studio version 3.1.x For UI layout improvement:

We have created a custom logo for both technician and customer for the homepage We have added a dimen resource file and style.xml file in the values and added the images and background.xml files for choosing the background.

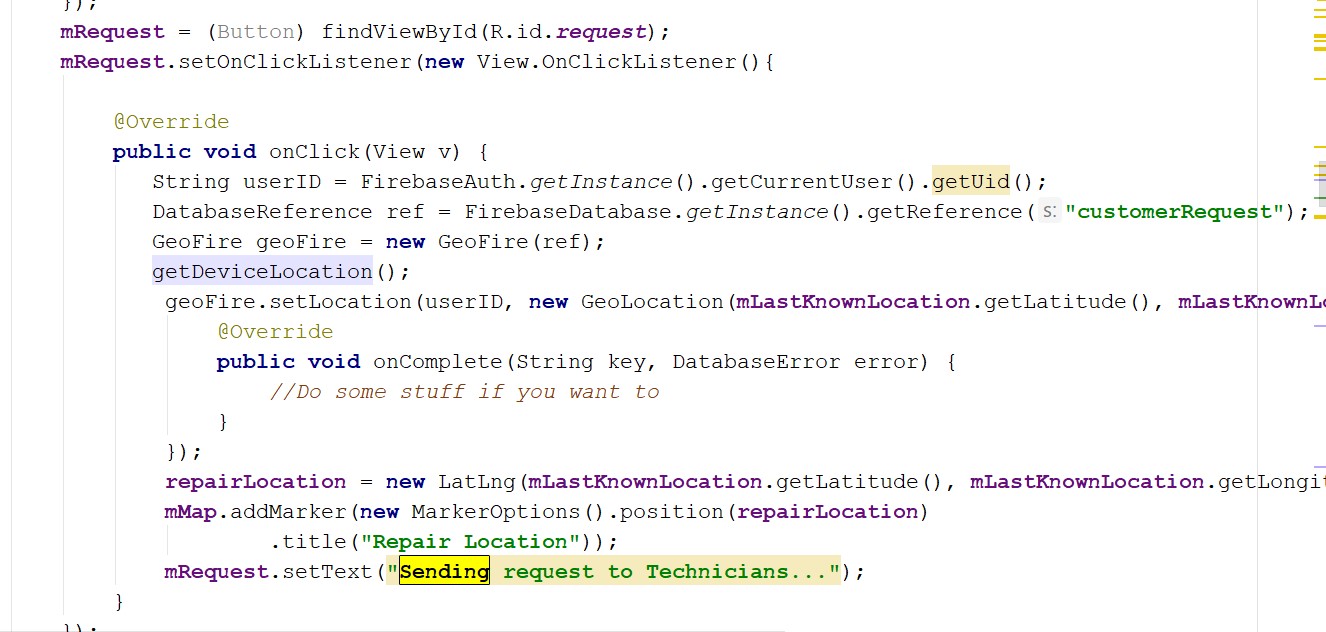
getLocationRequest() is responsible for determining the frequency of updating the real-time location. For FixitUp, we set it to be around 4000 milliseconds, or 4 seconds to be the average. The fastest it can update is 2 seconds.



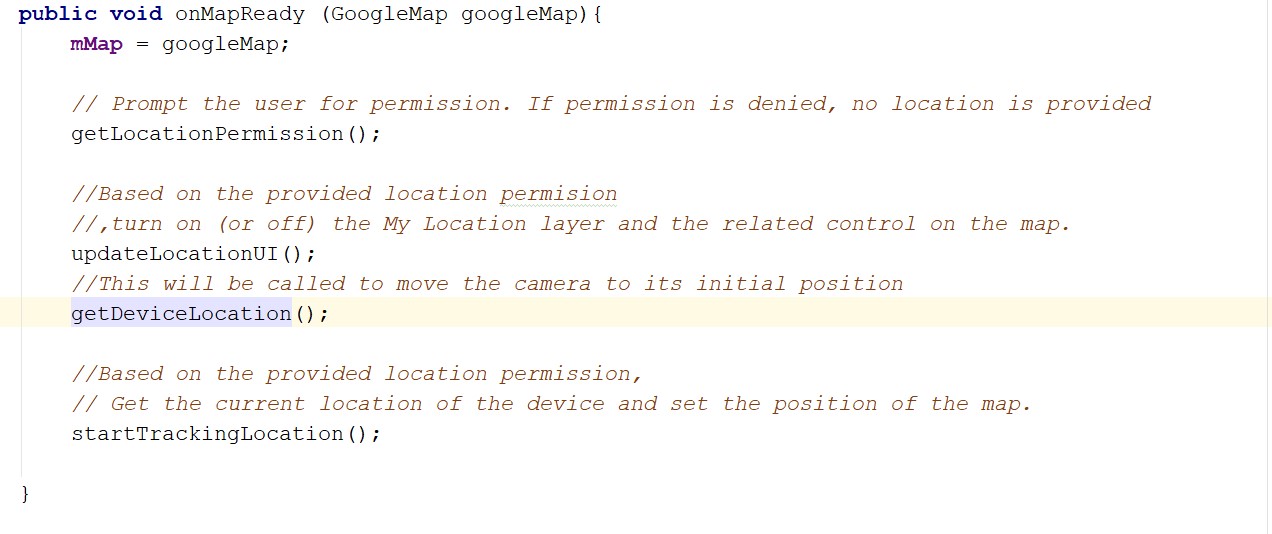
LocationCallback will be returned as soon as the getLocationRequest() finishes. This will update the UI with a marker that indicates the current location and also updates the last known location of the technician to the database



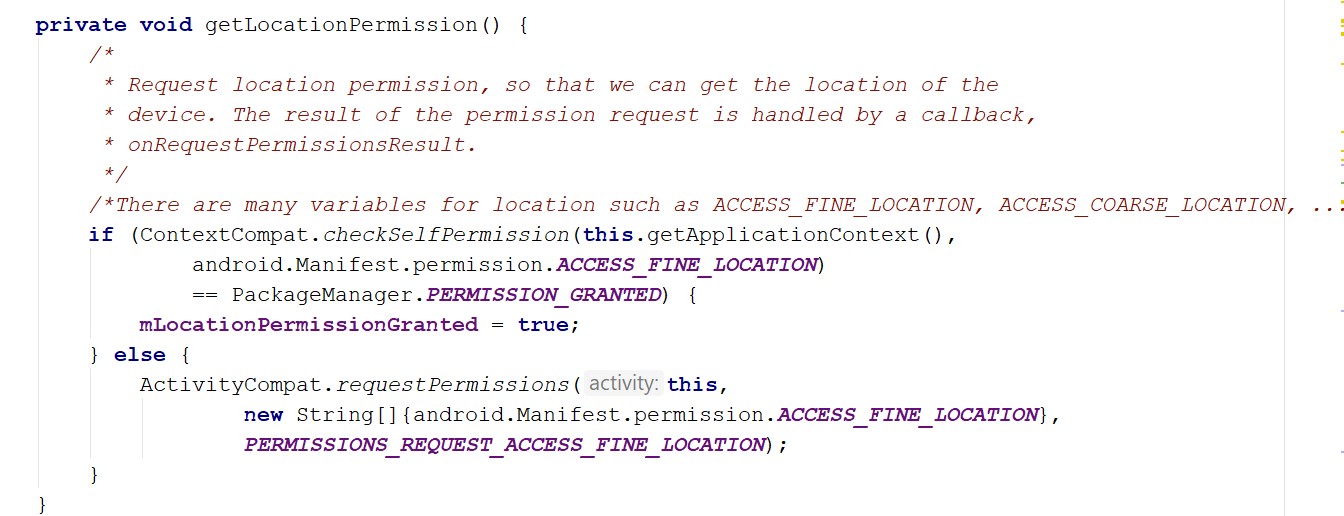
This request button is placed within the Customer Map activity with a click listener that will pinpoint the current location as the repair location and also update that location to the Firebase database



When Google Map is ready, it has four big functions to call, each of which is a wrapper function of its own auxiliary functions. getLocationPermission() will ask for user’s permission updateLocationUI() and getDeviceLocation() will update the UI and location based on the permission. If permission is denied, default location of Sydney, Australia is used startTrackingLocation() will be used to listen to the user’s location changes and update the Firebase database accordingly.



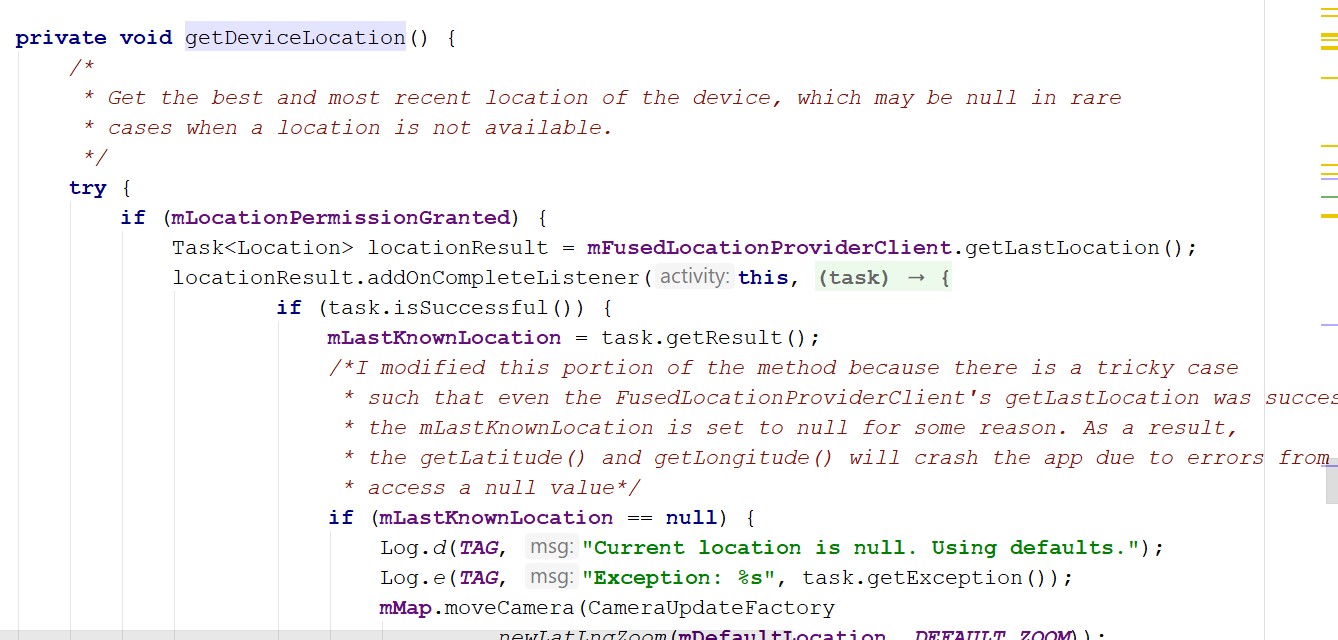
getLocationPermission() will request user’s permission to access fine location and check whether permission is granted



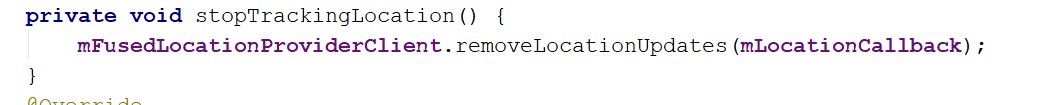
startTrackingLocation() takes care of listening to location changes, updating the latest locations on the map, and updating that location to the Firebase database



getDeviceLocation() will be called to update the latest location to update the initial location based on permission. startTrackingLocation() is not meticulous in tracking permission and default location.



stopTrackingLocation() is called when the application is paused or closed.



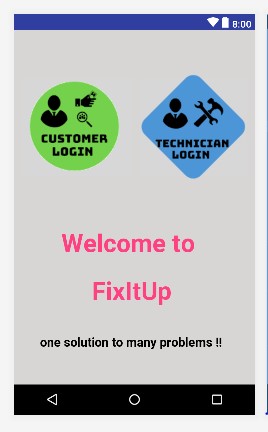
**Deployment:**

Our project is up-to-date with our GitHub as far as the most stable release is concerned. Other ongoing development and features need to be further tested, analyzed, and discussed. They are not yet published and are intended to be released in the upcoming increments.

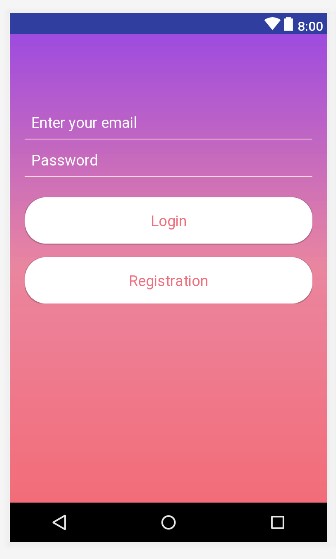
[GitHub Repository URL](https://github.com/benamreview/CS5551-Team7-Project/tree/master/SourceCode/FixItUpApp)

[Wiki Link](https://github.com/benamreview/CS5551-Team7-Project/wiki/)

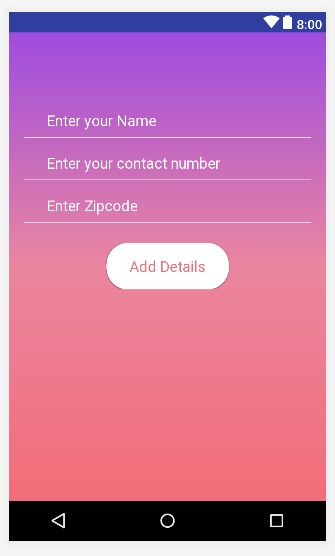
Homepage



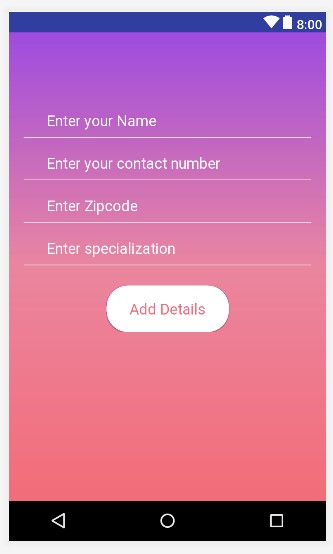
Login page for customer and technician



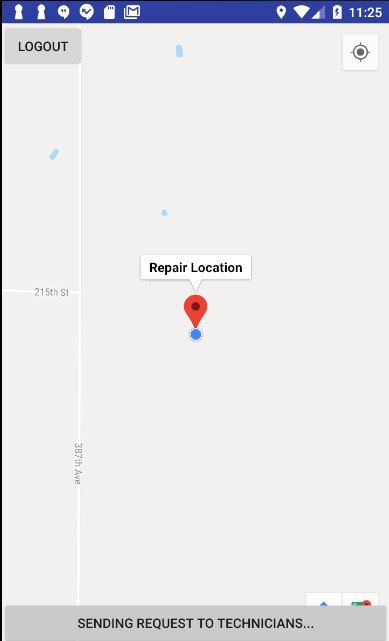
Customer Registration



Technician Registration



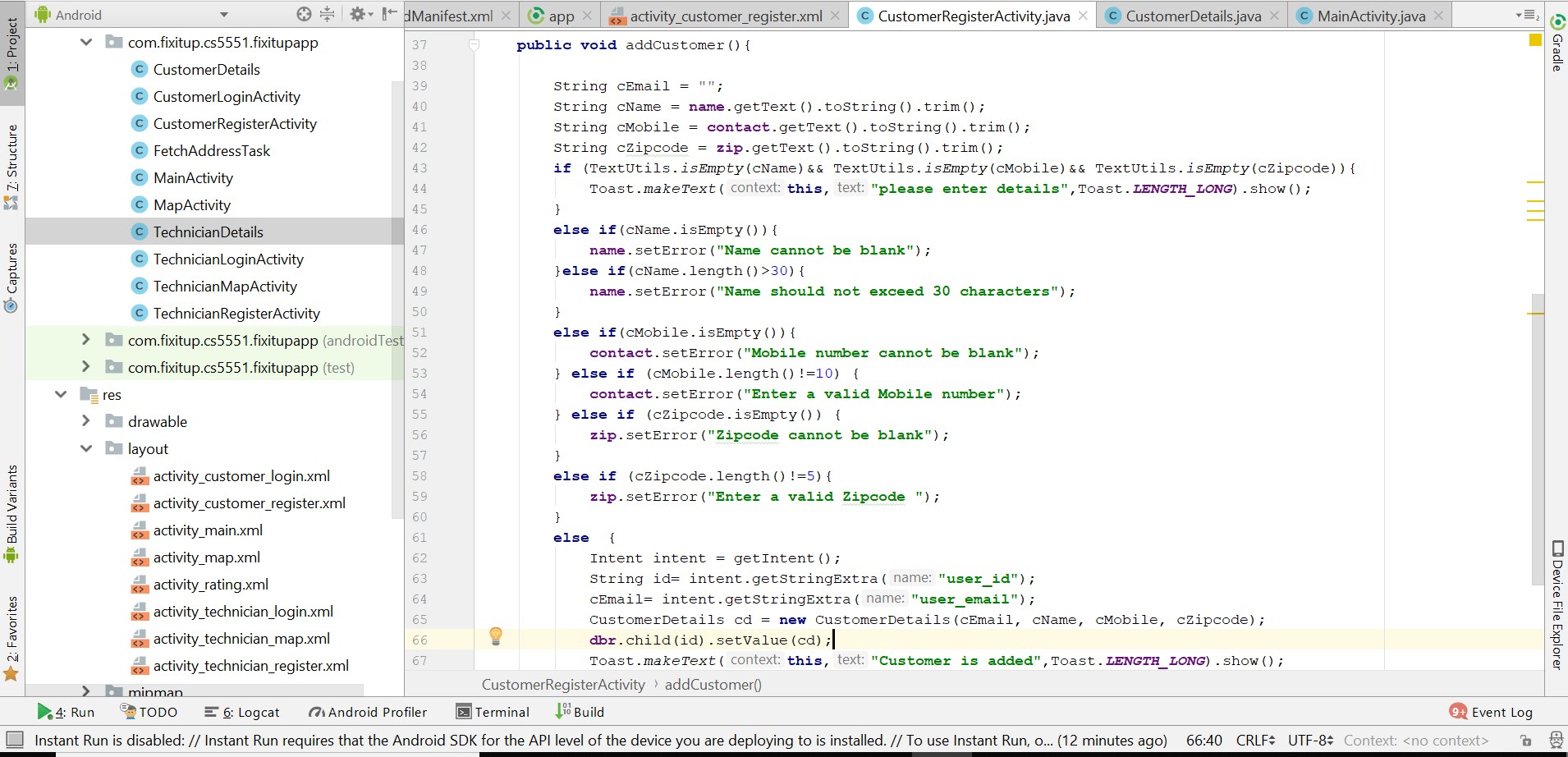
Demo of Map Feature 2 (Request button is pressed)



Firebase Database’s stored latitude and longitude. Technician’s stored location will be deleted as soon as he leaves the application so that the location is always fresh and updated.



Customer Validation

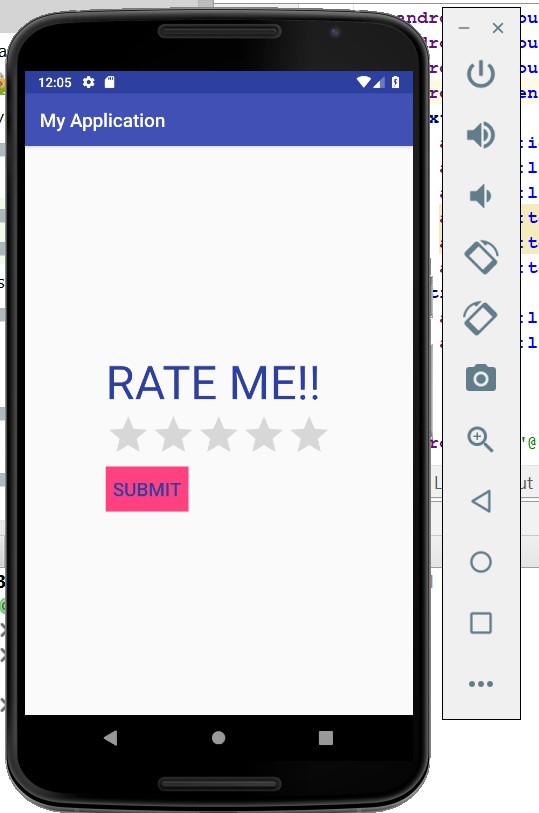


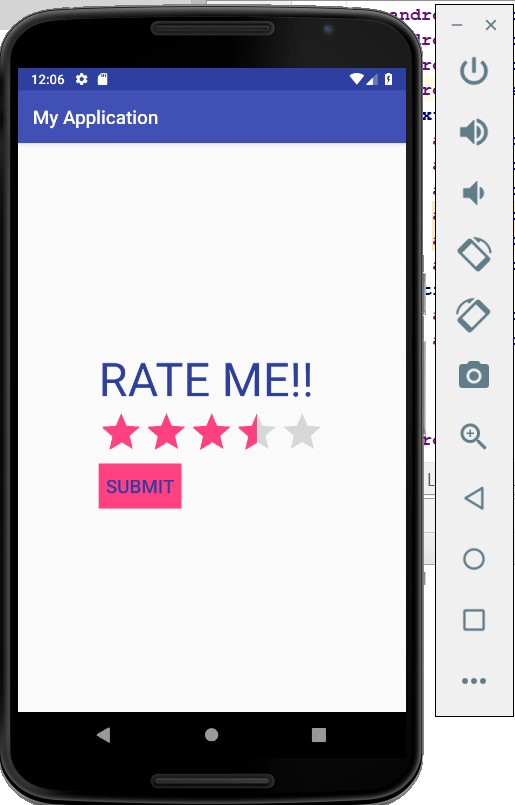
Rating For a Technician

Customer is given chance to rate the technician based on his performance. Once the rating is done, it is stored into the database and the average of all the ratings will be display for a particular Technician.



Demo of Rating





## Project Management

**Implementation Status Report**

**Work Completed**

*Description:*

Map Feature (realtime display, realtime update to database)

Logout function

UI layout improvement

Registration data validation

Database retrieval in Home page

*Responsiblity:*

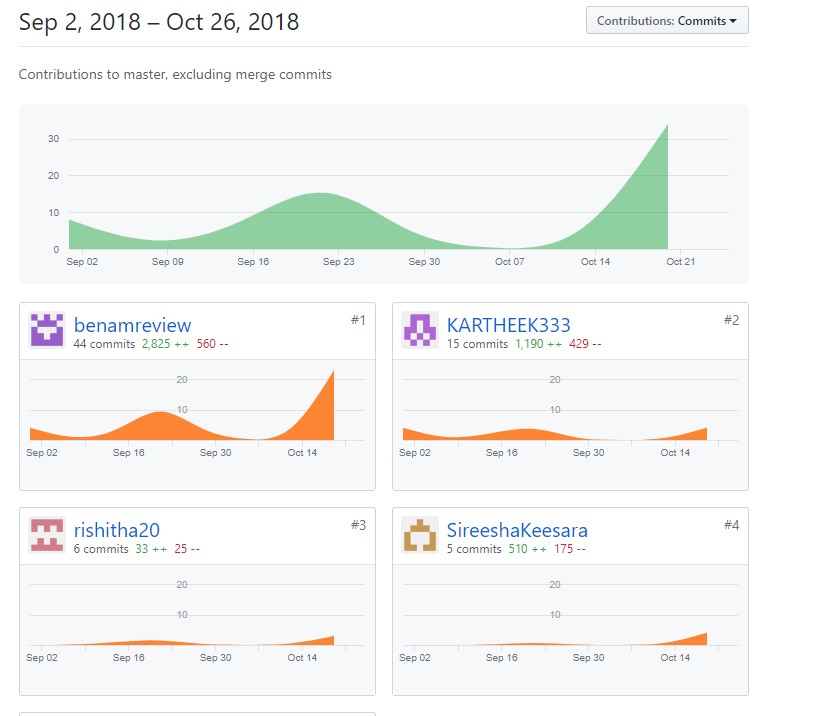
Duy: Map Feature, Log out function, Version control

Karthik: UI layout for all activities, Logo Design

Sireesha: Database Retrieval, connection between Customer and Technician activity.

Rishitha: Customer data validation

Time Taken: 45 hours Contributions:



**Work to be Completed :**

Description:

Home page improvement for Customers and Technicians

Rating and Feedback

Request the closest technician

Waiting time, Queue

Pricing and charges

Distance and Time Estimation

Scheduling

Server side setup

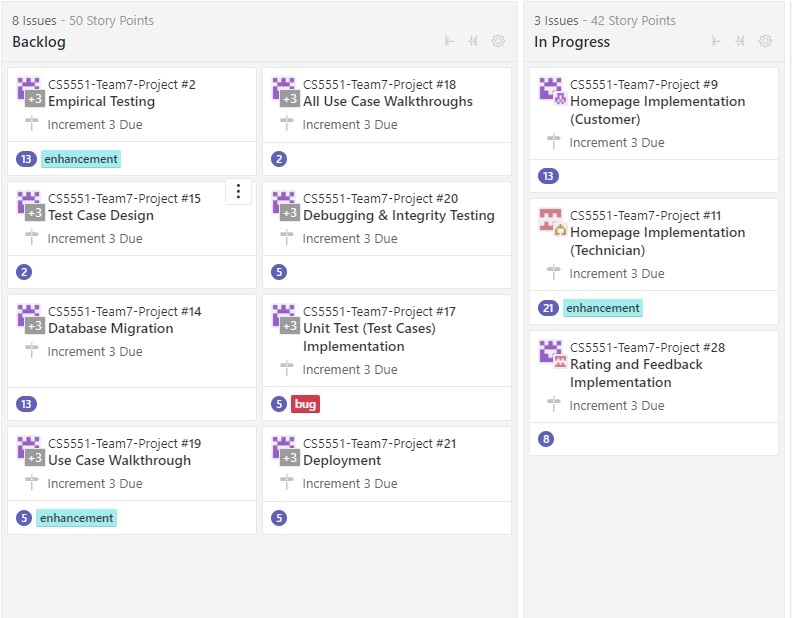
UI Improvement and interaction Unit Testing

Responsibility:

Duy: Home page improvement for Customers and Technicians, Request the closest technician, Server setup Karthik: UI improvement and interaction

Sireesha: Waiting time, Queue, Pricing and charges

Rishitha: Rating and feedback, Home page improvement for Customers and Technicians, Testing



## Third Increment Report

## Existing Services/ REST API

Up to the end of this increment, we are not implementing any REST API (APIs that use get and parameters to retrieve information such as Weather Underground and Nutritionix). However, we are using:

Google Firebase: as cloud database/storage, and cloud authentication using User Email/ Password. This eliminates the constraints of validating against the database and Firebase has a built-in verification method for itself using FireBaseAuth

Google Maps for Android SDK: acts to show users where they are currently located and other technicians' current location as well. This will be useful as it gives the customers/technicians a better picture of waiting time/remaining distance from one another.

GeoFire: Updates real-time location of customers/technicians to Firebase Database and queries the storage of locations based on distance range and requested location of customer. Up to the end of this increment we have made changes to the customer home page which now displays the technicians with the searched specialization and enables the customer to order the technician. Using GeoFire requesting a technician nearby. And also implemented cancellation of orders.

**Implementation:**

The form of Application we are building is native application.

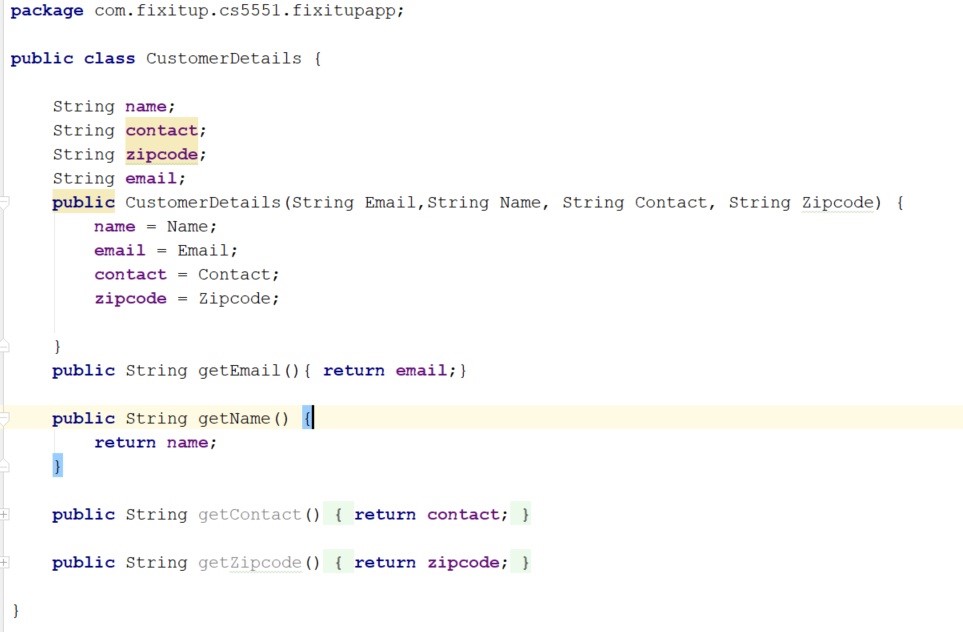
**Platform:**

Android

**Tool:**

Android Studio version 3.1.x For UI layout improvement:

CustomerDetails page wit hname, contact, zipcode, and email. This object will be used to push to the Firebase database



Search function is implemented with A listview and an adapter that converts the object retrieved from the Firebase to a

TechnicianDetails object to be displayed as a list on the Customer Home page

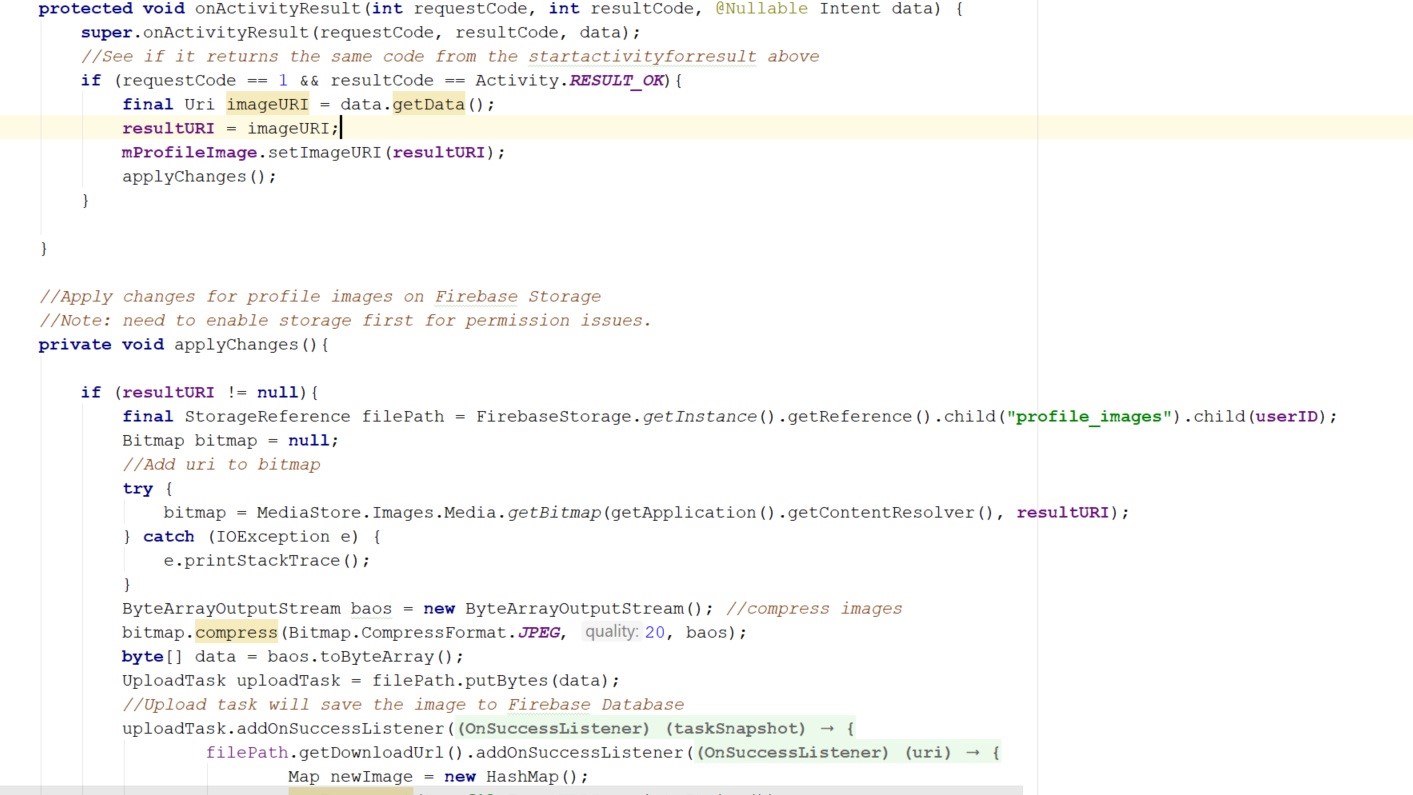


3 new features that have been added, including Setting buttons (for changing display and personal information), Logout to end the current user session, and Profile Image for displaying the user’s image from photos or gallery



Implementation of code that sets the profile image. Image is stored as bitmap and compressed to JPEG and uploaded to

Firebase Storage which will then later store the image and provide a URL in the Firebase Database



The Old Customer Login activity has been split into CustomerLoginActivity, CustomerLoginScreenActivity, and CustomerSignUp activity for better separation of concerns.



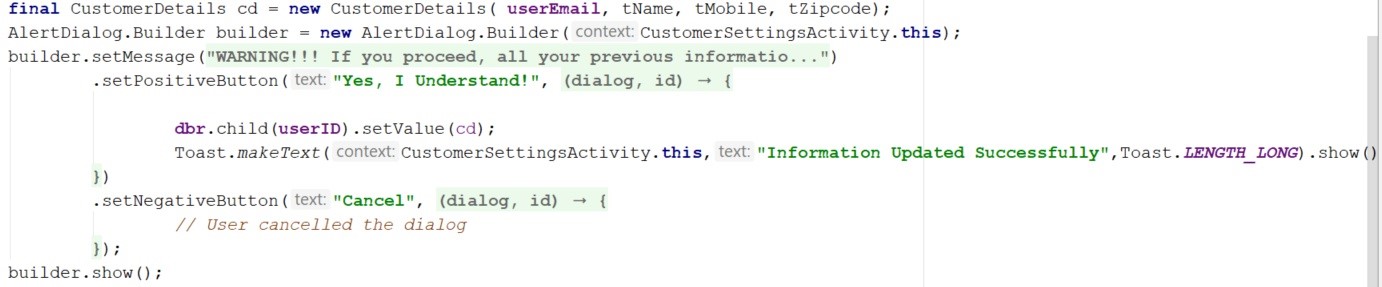
The implementation of the rating bar and its corresponding translation of number of stars to text



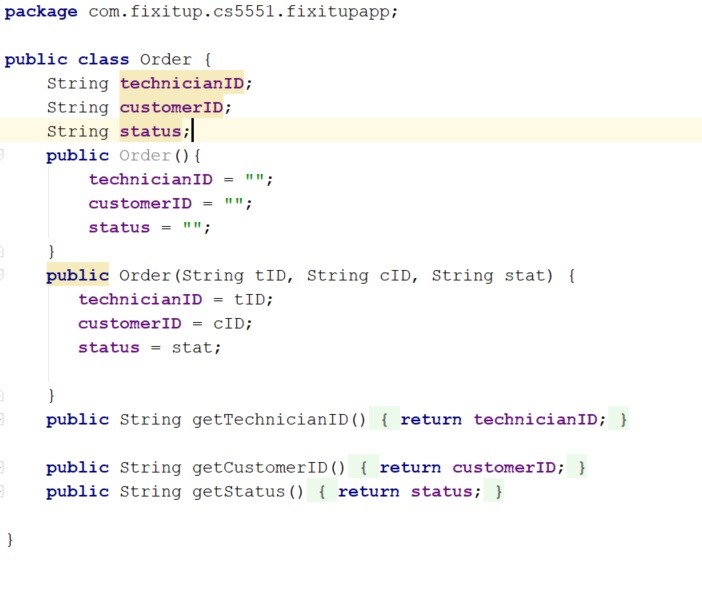
Implementation of feedback: feedback is not stored to Firebase yet, but the rating will be stored under the profile of the respective Technician/Customer



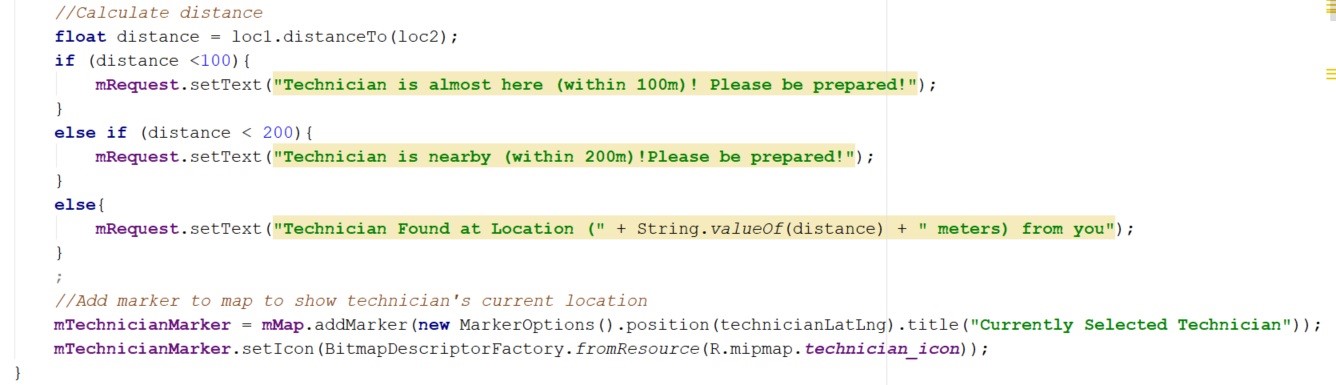
Dialog box that warns the user that information will be changed permanently.



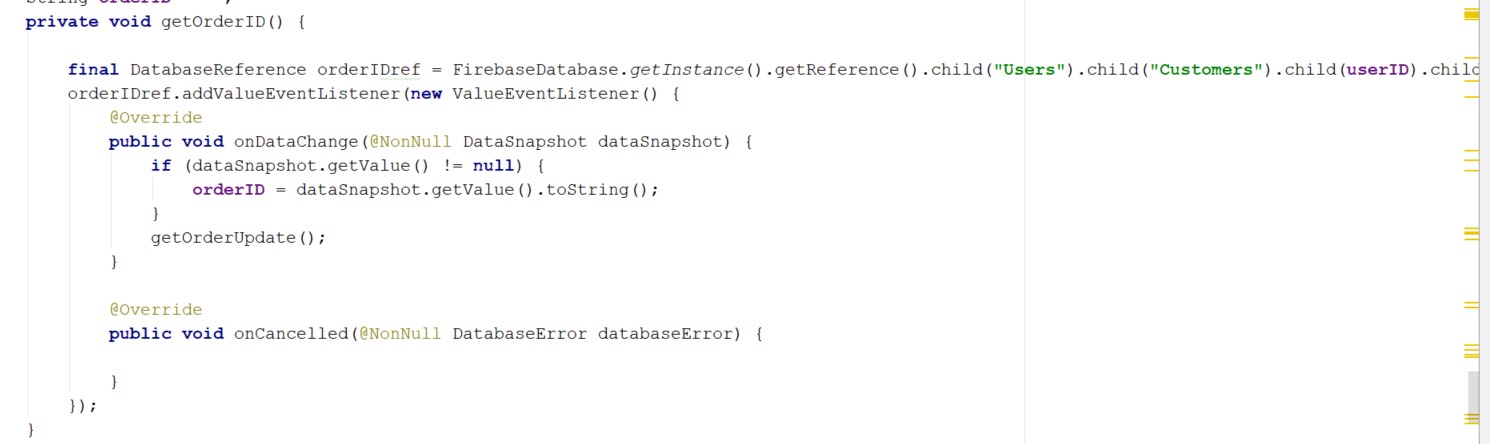
Order object that contains the TechnicianID, customerID, and status



Distance notification whether a technician or customer is nearby



This function is placed in the customermapactivity where the activity is constantly looking for changes in the customer table to see if the technician has started the session or not. If yes, the orderID will be available afterwards



Notification on the TechnicianMapActivity that is invisible in the beginning and only shows when technician arrives at the repair location. Used to display a button for technician to start the session



Send feedback to the Customers table of the Firebase database



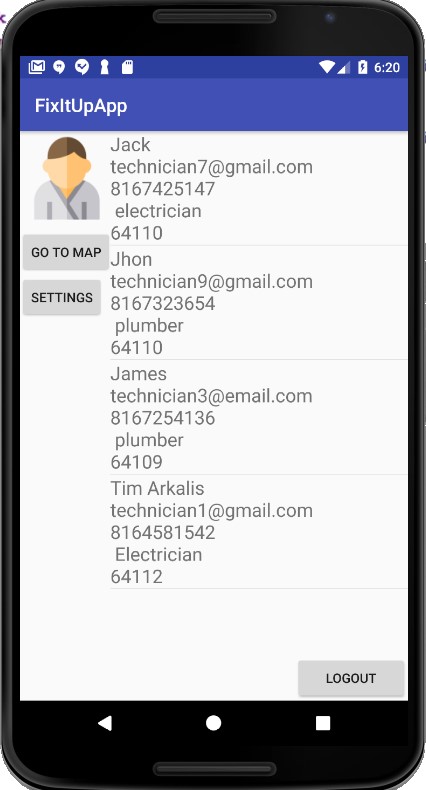
**Deployment:**

Our project is up-to-date with our GitHub as far as the most stable release is concerned. Other ongoing development and features need to be further tested, analyzed, and discussed. They are not yet published and are intended to be released in the upcoming increments.

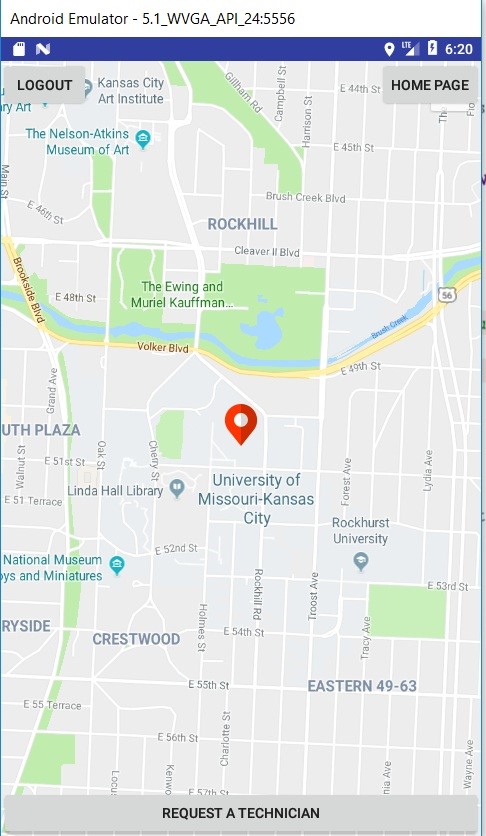
[GitHub Repository URL](https://github.com/benamreview/CS5551-Team7-Project/tree/master/SourceCode/FixItUpApp)

[Wiki Link](https://github.com/benamreview/CS5551-Team7-Project/wiki/)

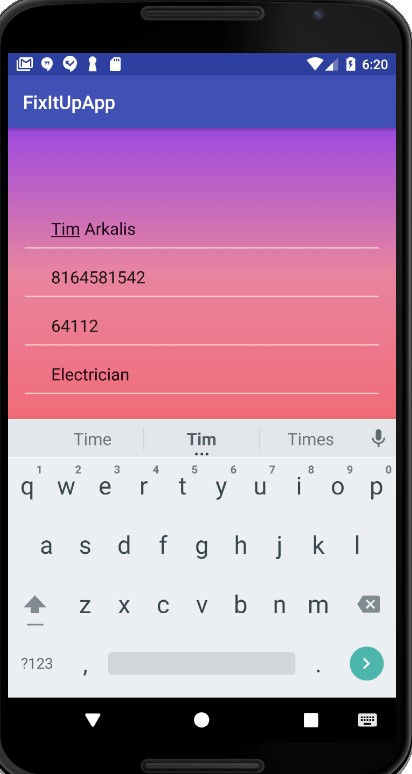
Customer and Technician Home Page’s layout



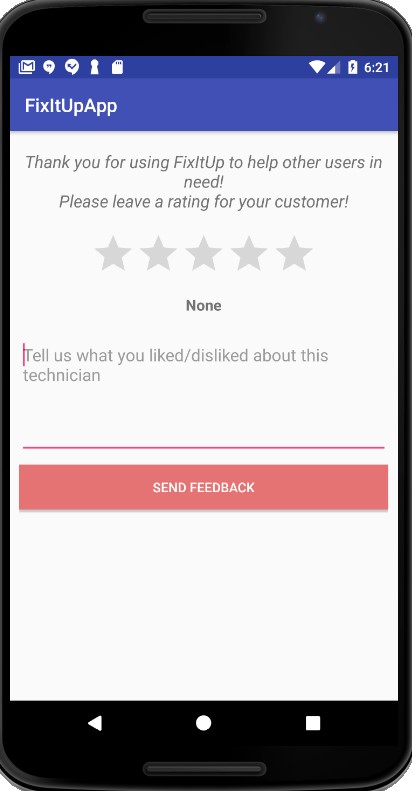
Customer Map Activity (default)



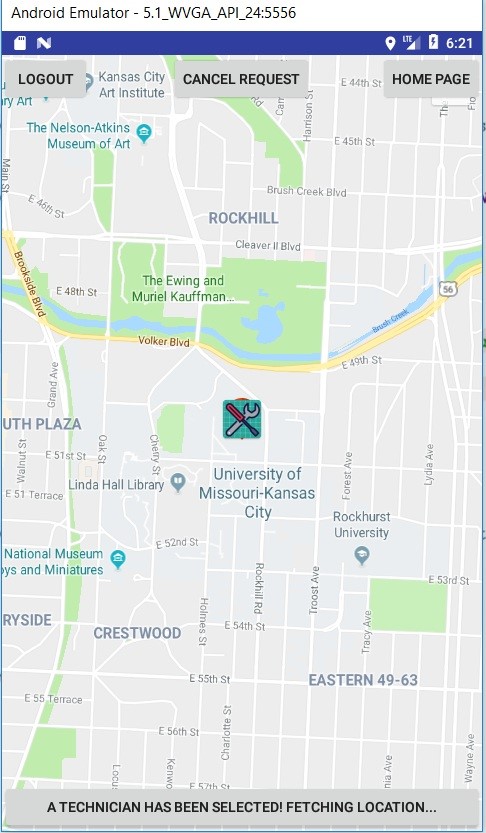
Technician Settings Activity



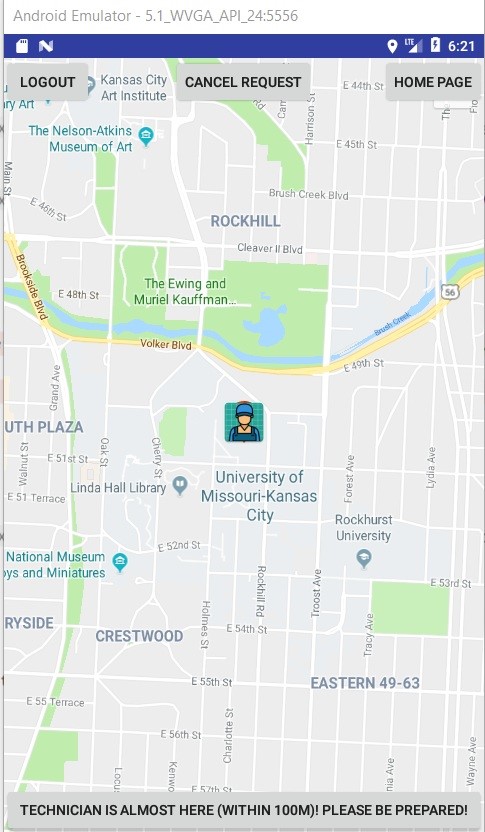
Technician and Customer Rating Activity



Customer: Fetching Location of Technician (displayed after clicking Request a Technician)



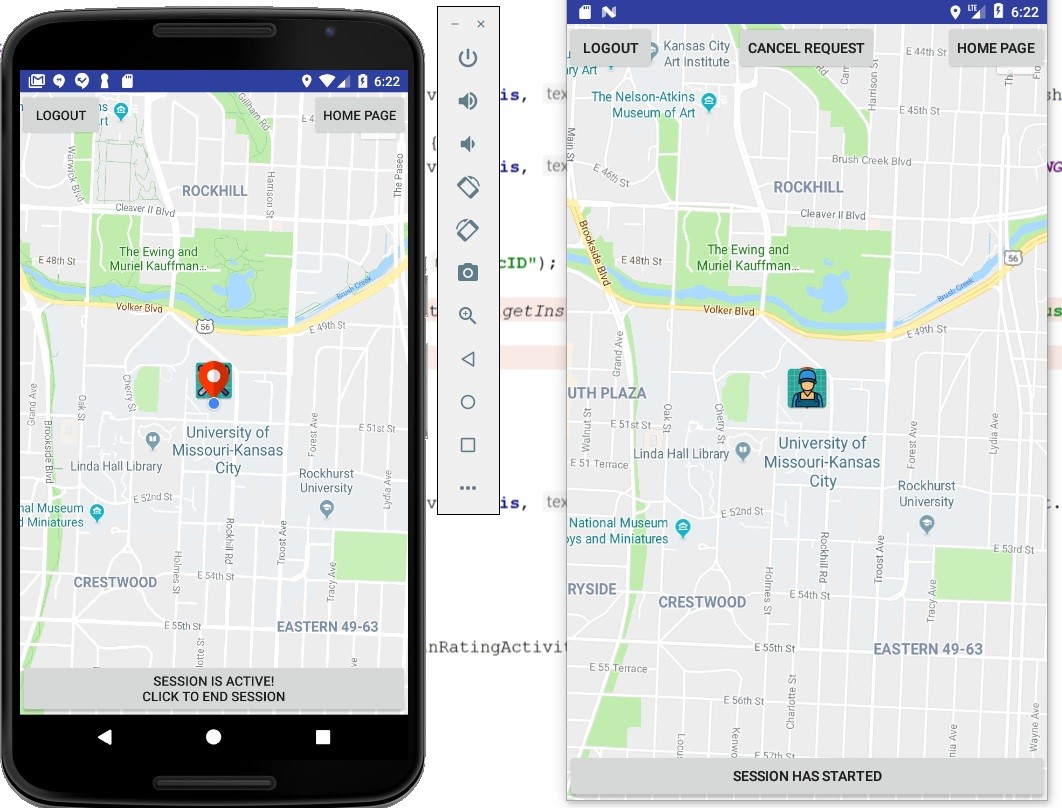
Customer: Tracking Location of Technician (after a technician has been matched)



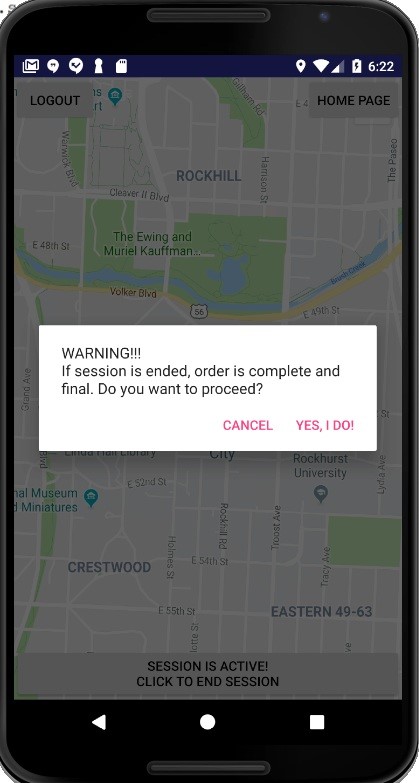
Technician: tracking location of Customer (when arrived)



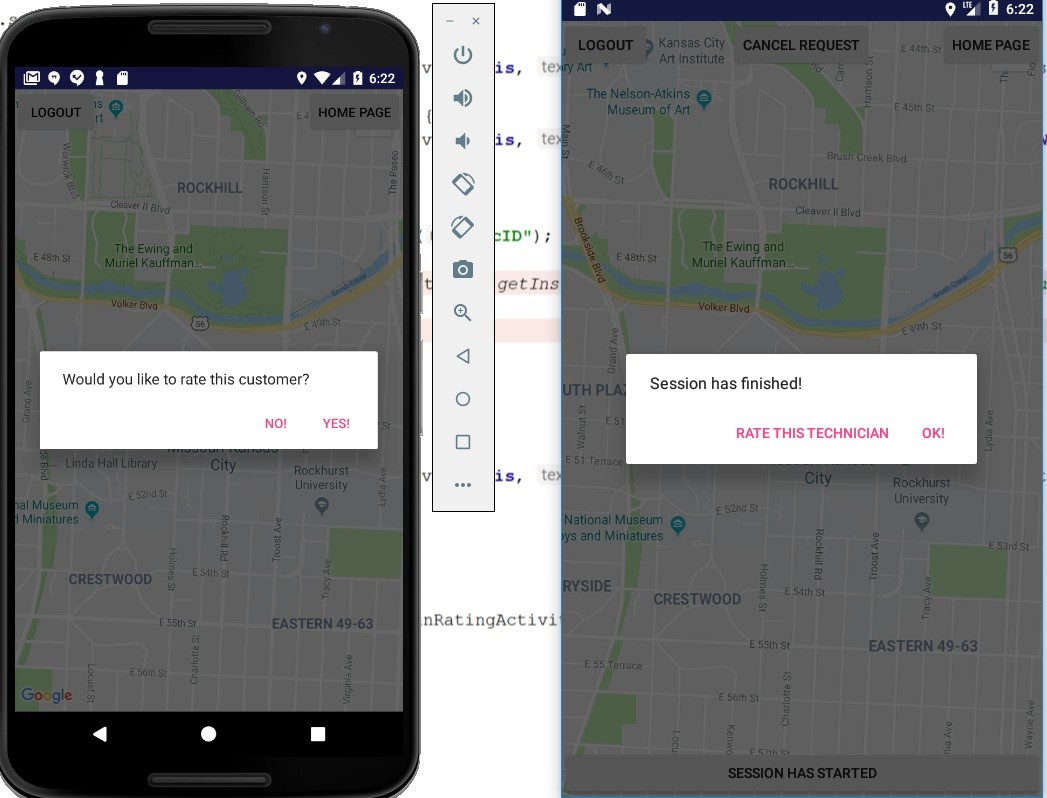
Technician and Customer: active session



Technician: session ending’s warning

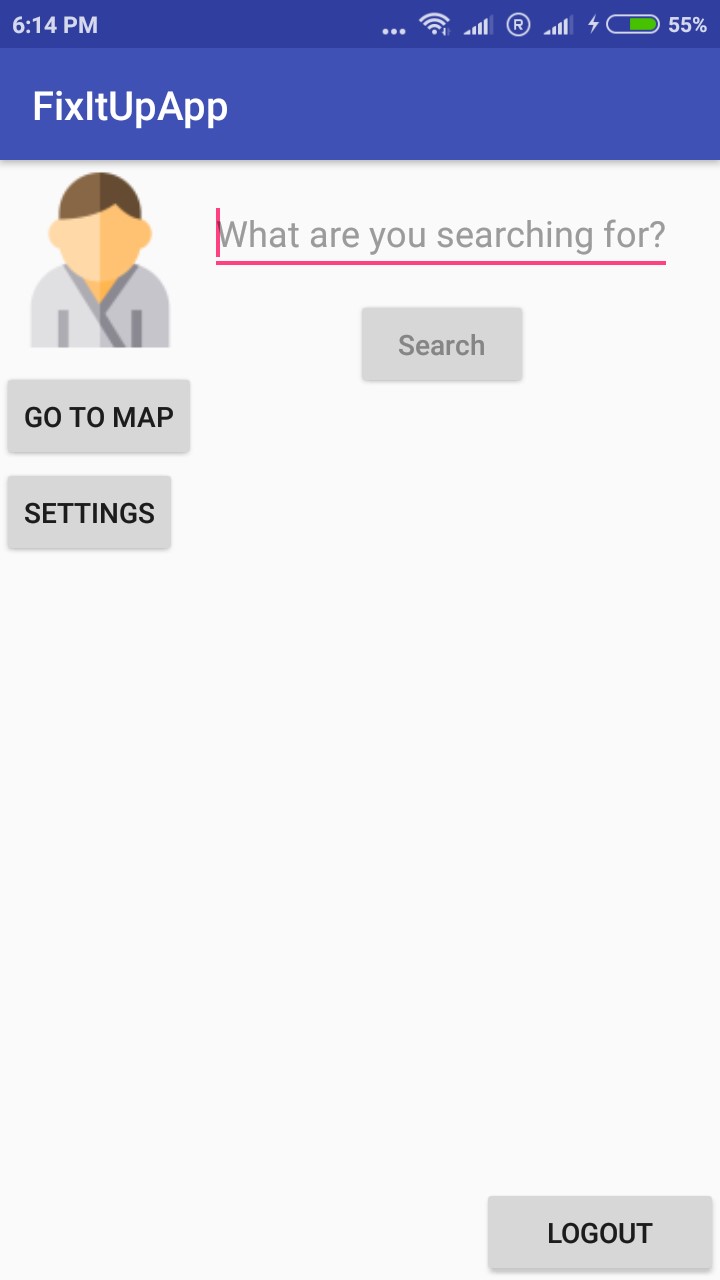


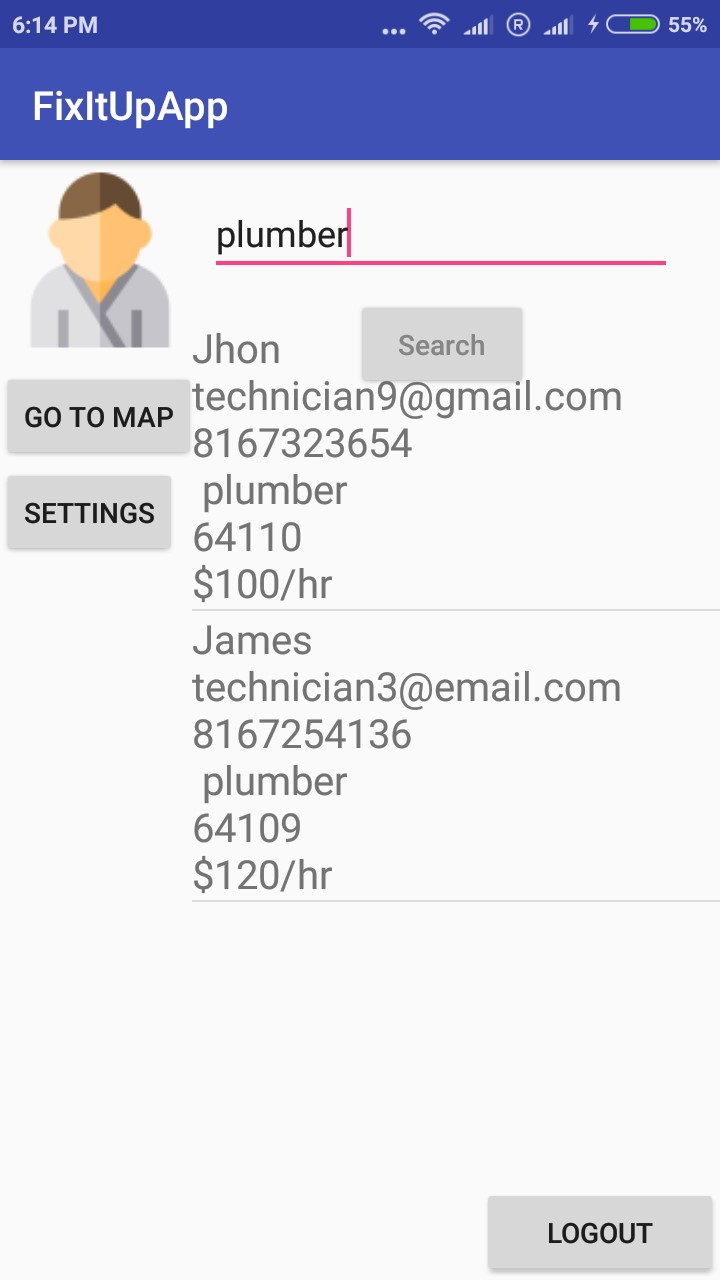
Technician and Customer: Session ending notification

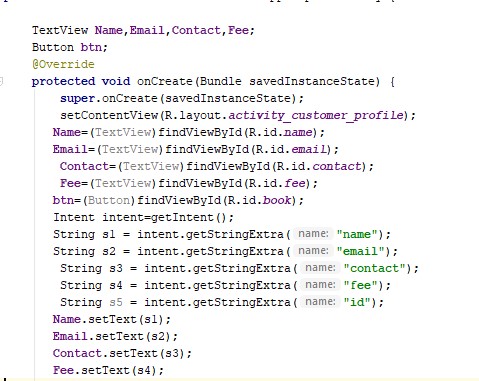


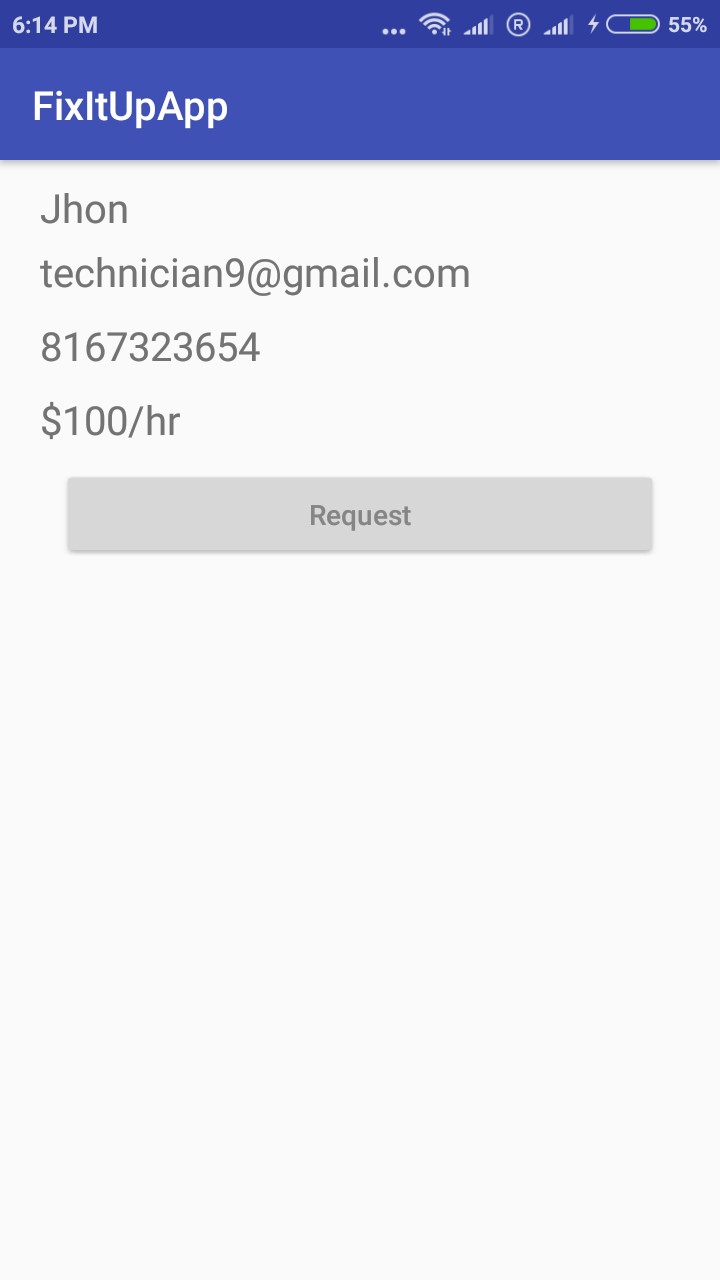
//added by sireesha











## Project Management

**Implementation Status Report**

*Responsiblity:*

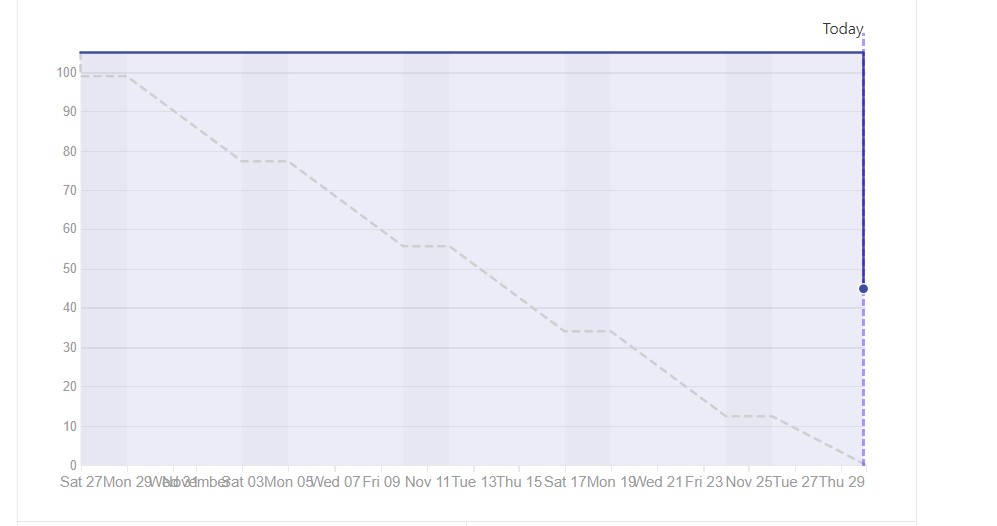
Duy: Location binding between Customers/Technicians, Completed order and session binding (cancel, started,

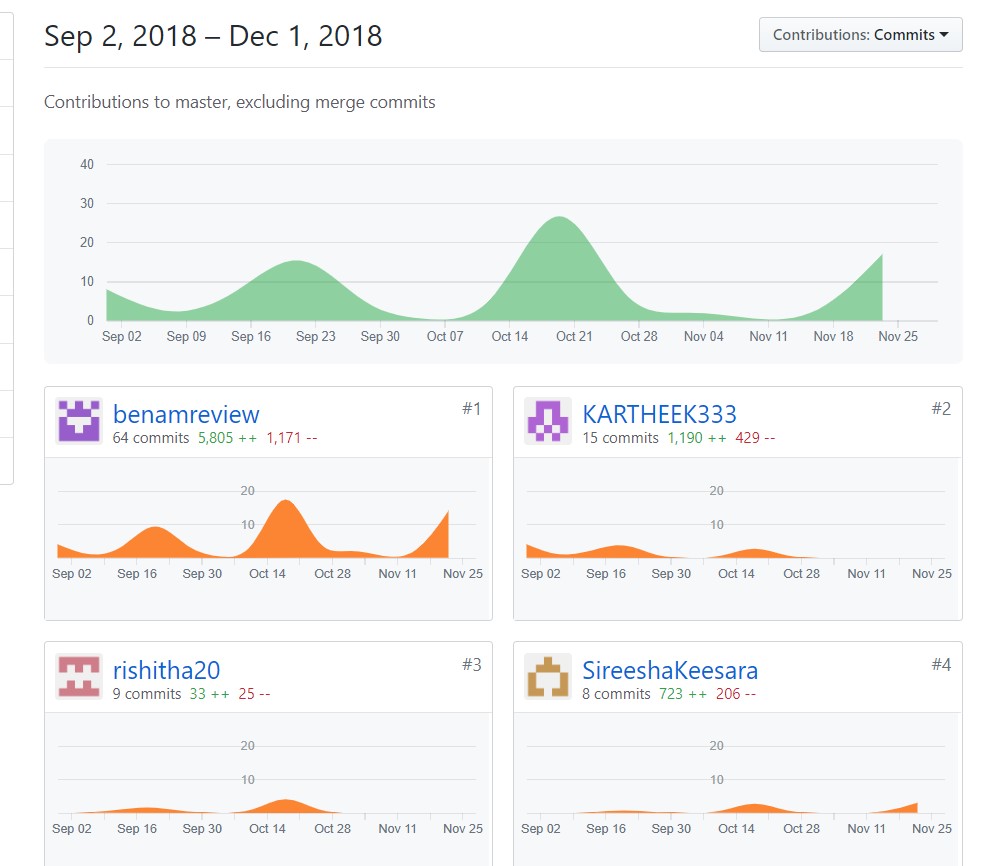
completed), user’s profile image, Customer and Technician ratings, Restructured app logic, layout, and flow.

Karthik:

Sireesha: Search function for customer, Customer Home Page, Customer Orders Technician.

Rishitha: Estimated number of hours: 50





## Bibliography

[Firebase Tutorial](https://firebase.google.com/docs/android/setup?authuser=0)

[Android Tutorial](https://html2pdf.com/files/2r6iwaonctbca64n/o_1ctjkbr6n1qrp16281vvub2t1h1ha/www.simcoder.com)

[Google Maps Tutorial](https://www.javatips.net/api/LOST-master/lost/src/main/java/com/mapzen/android/lost/api/LocationCallback.java)

[Android Resources](https://developer.android.com/reference/android/content/res/Resources)

*"The work has been completed under the guidance of Dr. Yugi Lee, Rajaram Anantharaman, and TAs (Ruthvic Punyamurtula, Bhargavi Nadendla, Sravanthi Gogadi) in CS5551 Advanced Software Engineering, University of Missouri -Kansas City), Fall 2018.*