Wi-Fi Mapper for the University of Cape Town network

Clayton Sibanda
Department of Computer Science
SBNCLA002@myuct.ac.za

David Kheri
Department of Computer Science
KHRDAV001@myuct.ac.za

Meluleki Dube
Department of Computer Science
DBXMEL004@myuct.ac.za

Abstract

The goal of the project is to create a University of Cape Town(UCT) eduroam Wi-Fi Mapper, which is an application to allow the eduroam users to be able to view the quality/strength of the eduroam signal at different areas/locations on campus. The deliverables required consisted of a android application and a web client application for administration use.

The two deliverables needed were produced and their uses are as follows:

- **Android Application for users:** This was the application that the basic users will be using. With this the user can see the map of the UCT upper campus with colors over the different areas/zones indicating the signal strengths. The colors used are as follows:
  - Red: bad signal strength of 0% - 30%
  - Orange: poor signal 30%- 50%
  - Yellow: average signal strength 50% - 60%
  - Light: Green Good signal strength 60% - 80 %
  - Dark: Green Excellent signal strength 80% - 100%

  The user phones are also used to record Wi-Fi strengths of the different location areas together and send it to a database hosted on Firebase to update the average Wi-Fi strength of the whole area. If the Wi-Fi name is not eduroam the data is discarded.

- **Web Client:** The Web Client has the view of the map which shows the strengths of the different areas. It also provides the functionality for creating reports for the Information Communication Technology Services(ICTS) to monitor how the Wi-Fi strengths are at different areas.

1. Introduction

The project was concerned with the creation of a Wi-Fi signal strength mapper of the UCT eduroam internet across the upper Campus. The stakeholders who were involved in the project included:

- **Product Owner/Client:** This is the person in-charge of the project and for whom the project is being built for. In this case the client came with the project Idea which we then implemented in a way we saw fit. *The project owner in this instance is Dr. Josiah Chavula*

- **Basic users of the application** This is everyone who will download the application on their phone are going to using the phone to view the Wi-Fi strength of the eduroam networks in different areas on on UCT upper campus.

The basic end user of the application will also be responsible for collecting the data that will be used in the future to determine the average strength of a area. This has to be done with their permission.

- **Advanced users/ICTS department** These users will mainly use the application to monitor the Wi-Fi of the network across campus to act on the data.

The Wi-Fi Mapper project consisted of two main deliverables,android application for the basic users together with a web application for advanced users.

The purpose of the application was to provide a way for the students to know which places on campus have the best Wi-Fi strength so that they can go to those areas to use the Wi-Fi for applications that require strong Wi-Fi signals to operate appropriately. The problems together with the opportunities that arise from this projects are:

- **Problems:**
  - One has to travel to a further distance to an access point with a strong wifi signal
  - New users of eduroam service do not have enough experience to know where strong signals can be found on campus
- **Opportunities:**
  - Give a wider range of locations with a strong signal strength
  - Maintenance easier for ICTS

The Wi-Fi mapper then serves as the platform to provide information and overview of the signal areas of the campus for the stake holders mentioned above.

Software engineering methods used

The project was developed in an iterative manner. Therefore the methodology used for this project was Agile. Agile development methods include developing the application in both iterative and incremental steps. Each end of an iteration is met with a milestone set upon by the client and the development team. The reason for going with this approach was so as to keep the client in the development process so as to not incorporate the change in requirements that the client might have and also to ensure that the client was happy with the progress at each development stage. At the end of each iteration the client was consulted to ensure the application being developed was what the client had envisioned,that way instant Client feedback was given back in the case things were not going as planned. The model used in the project was the evolutionary prototyping method where the application was built in iterations of prototypes. These prototypes where shown to the user at the end of each iteration.

2. Requirements Captured

The following are requirement divided into functional and non-functional requirement which where collected after consulting with the client.

Functionality requirements

The list of the requirements for the application where as follows:
Functional Requirements

- Rendering of the map on the user screen
- Have signal strength indicators on the map. These should be updated on a real-time basis.
- Save signal strength on the database for a specific location.
- Zoom in and zoom out of the campus map

Non-Functional Requirements

- Real-time receiving of data being updated in the database.
- Speed up on the rendering of the areas and their strengths on the map.

Use case

Below is a use case diagram to show the different operation that can be done on the system and the actors who are responsible for performing these actions. Firstly, there is a need to discuss the different users of the systems and the different systems that are used by our system. These make up the actors involved in the system. The different actors and their level of interaction with the application are listed below:
Primary Actor

- User
- ICTS (extending User)
- Phone

Secondary users

- Server (where the application data)
- User (for some instances.)

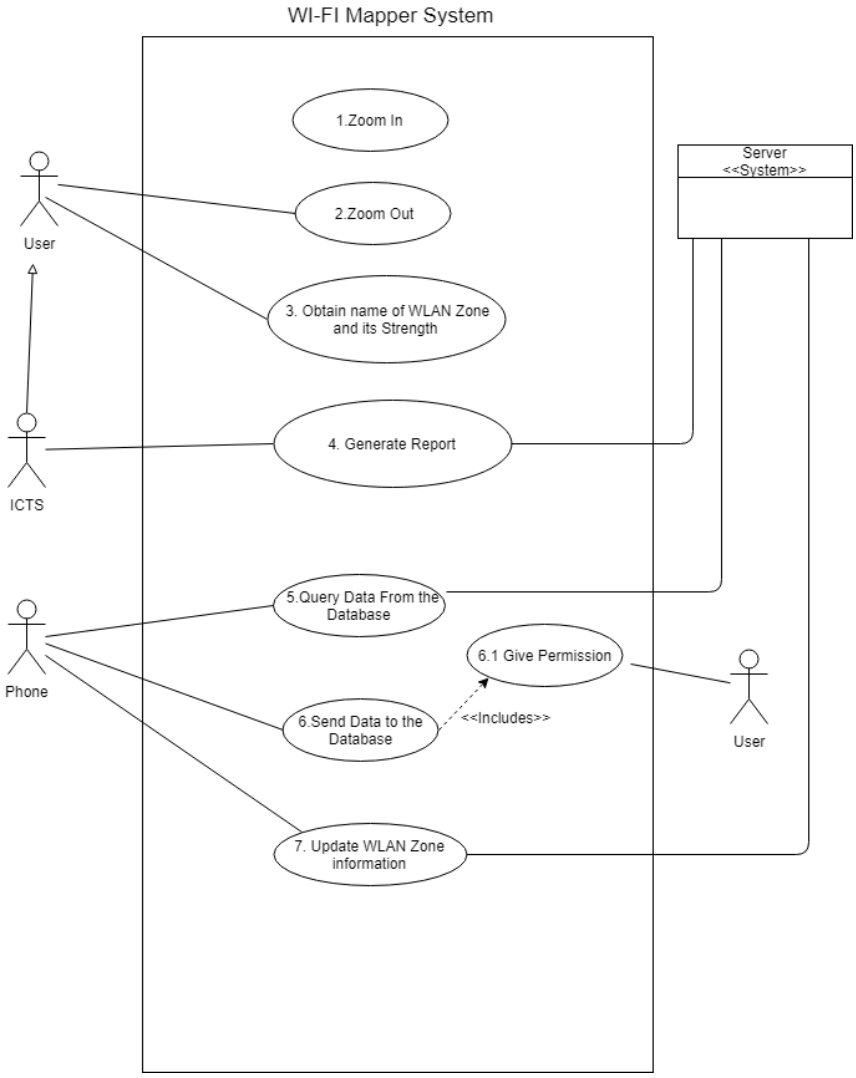

Figure 1: Showing the use case diagram of the Wi-Fi Mapper

The main use case for the system is update WLAN Zone information as it is the spawns many more functions that influence the user's view of the menu. The detailed flow of events for updating information of a particular zone are shown by the use case narrative below:

Use Case:	Update WLAN Zone	**ID:** 007	**Level:** High
Actors:	Users, ICTS, Wi-Fi Mapper Server		
Stakeholders and Interests	Josiah Chavula -The owner of the product, ICTS - they use the information provided by the system.		
Brief Description:	A user who is using the app opens the application to get the different eduroam strength signals. The system gives user the data for the whole UCT upper campus map. If location permissions are granted the user is then also able to send their data to the server and get the map zoomed in to give them strength signals on a few meters radius around them.		
Preconditions:	User connected to eduroam. User is at the locations that are supported by the application.		
Post Conditions:	Data is updated at the server. User has a data showing different signal strengths across the UCT Campus.		
Related Use Cases:	includes: Get user permissions		

Typical Course of Events	
Actor Action	**System Response**
1. Phone Queries the database for data for the supported areas and return to user.	2. Queries the database for data for the supported areas and return to user 3. Request user to give location permissions
4. Grants the system permission to use Location services	5. Get the current user location 6. Get the signal Strength of the users Wi-Fi 7. Set up Location Change listener
8. Continuously send the system data from where they are located.	9. Constantly accepts data from the users 10. Save the data 11. Use aggregate data from the users to calculate average strength signals for different locations.
12. User changes zoom level	13. System changes the focus level for the specific area.

Alternative Courses of Events	
Actor Action	**System Response**
4. (a) Refuse to give the system location permissions	5. (a) Continue showing the user the upper campus map without giving them information about where they are.
8. (a) User loses signal to send data to the server	9. (a) User loses signal to send data to the server (b) System notifies user that disconnection occurred

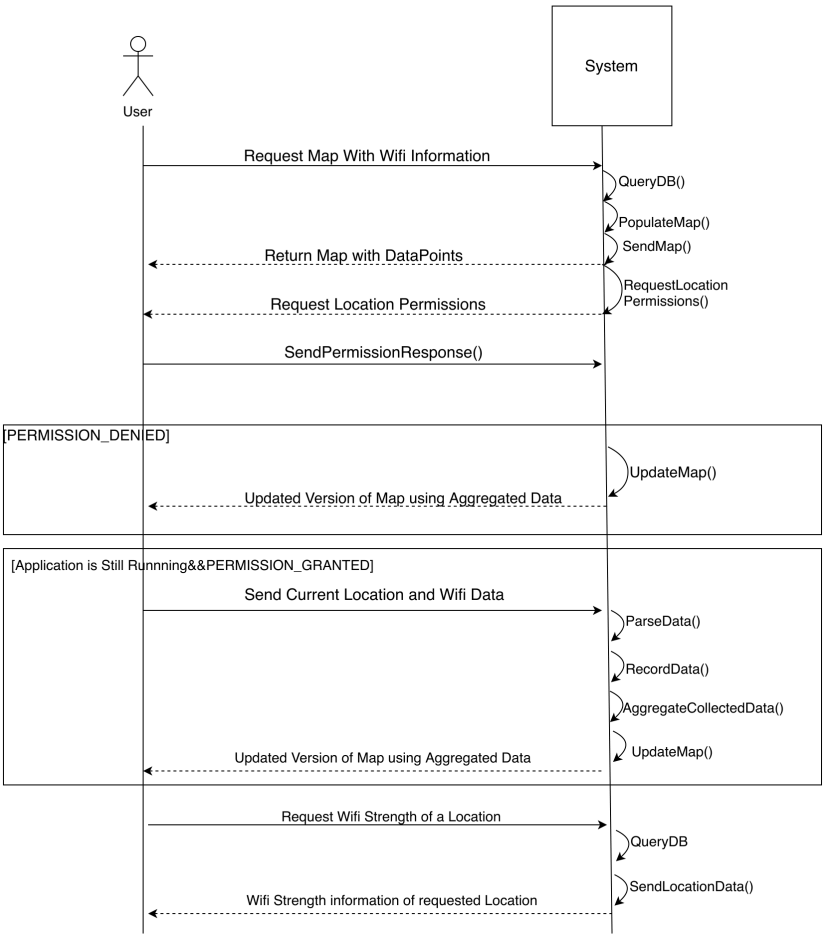

Figure 2: Sequence Diagram

Design Overview

This Section gives an overview of the system which entails showing how the system interacts with other systems in delivery of functionality to the user and introduce the basic functionality expected from the System,It will also describe what type of stakeholders will interact with the system and what functionality each should expect.

They are mainly Two ways users can interact with the System that is through a mobile Application(Android Only) and Web Application.

Mobile Application will need to communicate with the phone's GPS system so as to obtain a user's Location.The functionality provided by GPS is crucial as it pinpoints exactly which WLAN Zone data is collected and ensures accurate aggregate Data is displayed on each WLAN zone.

Wifi Manager will also be needed by the mobile application so as to obtain Wifi Strength of the router to which the mobile phone is connected to a certain point in time,as both the Wifi Strength and Location to which this data is obtained are the main pieces of information driving the System.

Since this is a data-centric Application,that means data needs to be stored in a database,both Web Application and Mobile Application will communicate with the database but in slightly different ways.Mobile Application will both query and send data to and from the database while the web Application will be restricted to just querying data from the database for displaying purposes.

Query Manager is used by both Mobile so as to validate the data collected corresponds to a WLAN zone or else its not written to the database and handles requests of data from both web and Mobile Application to obtain data to display.

Both Mobile and Web Application use the Mapping System so as to display a Map to the user which includes data collected from the users stored in the database aggregated into different WLAN zones so as to provide appropriate information to the user.

Cluster System is required by the Mapping System so as to cluster Data points collected which are very close to each other so as to prevent clutter on the Map which is undesirable to the user(Clustering occurs only at high zoom level).

Reporting system is used by the Web Client to generate a report which will provide useful information based on the collected data stored in the database.

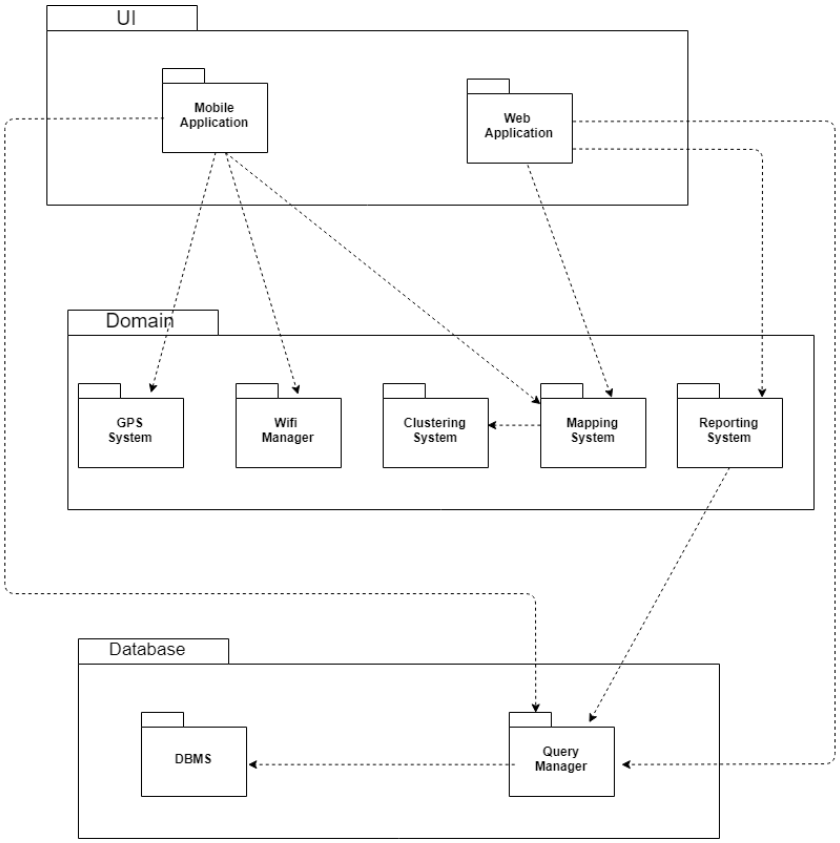

Figure 3: Architecture Diagram

3. Implementation

The major components of the application included, the database, the phone application and the web application version of the application. The application was implemented without a server. The implementation included a cloud database hosted on a third party computer in this case firebase. The phone application was developed for android phones to target the majority of users on campus as per the client's requirements. The Android application was therefore created in Java. The web application was created using JavaScript. Below follows the discussions of these components together with how the they came about and why they were settled on.

Database

Firebase, a NoSQL cloud database was used as the database option. Since a NoSQL database was used, this means that the data in the database was stored as large JSON documents. This helped in reduce the number of deliverable that were to be delivered and hence aided in helping the developers put more focus into other aspects of the problem spectrum. This also reduced the number of resources required to complete the project. The alternative to this would have required a server and a database. The server would have required to be constantly monitored and always on taking data from users and updating it on the database as well as sending new data to the users. With the current approach, we only have a cloud-hosted database that is hosted on Google servers that both applications will be getting data from and sending data to. The database will handle the updating of all users currently connected to it with the new relevant information and as well on the users request give provide the users with the data to populate the menu. The advantages of this approach include the following:

- There is no need for a server since firebase database will be able to act both as a server and as a persistent storage tool at the same time. It will assume the role of the server when it needs update the other clients connected to it of new data changes or send the client the data on the database on

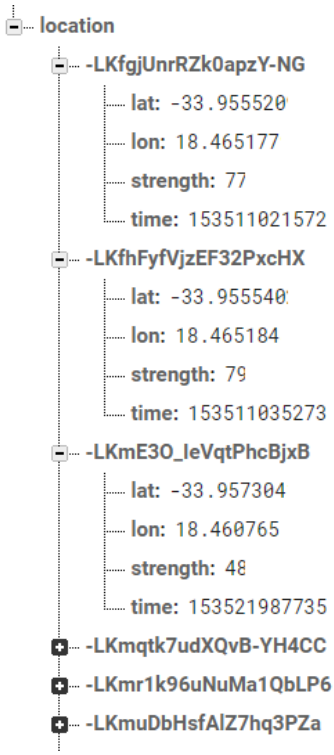

Figure 4: Showing the first design of the database developed.

opening of the application so as to populate the map. This also means there is no stressing about the hosting environment also.

- Firebase provide the connected users with realtime updates. This implies that when a value is updated by one user will be sent immediately to the other connected users.

Database Design

During the development cycle, the database had two main designs that where showed to the client. The first design was based on the miscommunication of the client requirements. In this design, the database only stored Location objects together with the signal strengths recorded for that location were stored. This is shown in the figure below As can be seen, there is not much structure in the way the was being stored. Each record had to have an id which made it unique so as to allow better queries. A time stamp was also added to the records on when being written to the database so as to allow future features to use it when they want to filter the strengths based on time. The location was stored as longitude and latitude objects.

The new database design now also include zone/area objects being stored. This store a list of 4 coordinates that define an area, the average strength of that area based on location points that are under that area that have recorded data, and lastly the name of the area. This design is shown below in next figure. The initial design was still used to maintain the list of all locations that have had data recorded from so as to be able to give the view of the places where the data was recorded for a WLAN zone if needed Now we get to the details.

Android application

The android application was written in Java using the Android Studio tools. The minimum Android version supported in the application was Android 5 API level 21. This version for android was installed in atleast 71% of the phones of people using android. The reason it was selected to be the minimum supported version was to manage to reach a large population of people at the same time taking advantage of the new features that are in the new APIs which may include better management of resources and

Figure 5: Showing the added document for storing area objects.

security issues for instance. The application built had only one Activity which was the main activity as well. This activity consisted of a map of UCT segmented into different areas based on the buildings and has each color with a specific color that relates into the signal strength in that area.

Web Application

Wifi mapper comes with a web client that consists of three views and these are map view, login view and the report view.

Wifi mapper comes with a web client that consists of three views and these are map view, login view and the report view.The technologies used to build the app are HTML,CSSS and JavaScript. A few frameworks were also include to improve team productivity.The map view displays a google map that contains the WLAN ZONEs over UCT upper campus. The app gets the map from the google map api obtained by including a script on the html file of the view.

The map view is set up such that the map fills the screen of the device fully. After rendering the map, the app asynchronously gets data from the firebase database using the 'on' firebase listener. The app sets the location center and defualt zoom. In order to get the data from the database firebase was added as a dependence on the app.After obtaining the data from the database, the app displays colored polygons on top of respective WLAN ZONEs. The colors of the polygons are calculated using the data returned by the database. Five colors are used to represent ranges between 0 and 100.

The map view also contains a button used to generate report for special users. On clicking this button, the login view is displayed on the screen.The login view consists of a login form with two input fields and a login button. The two input fields are for username and password. The password input field is such that the password is hidden while the user is typing it.

The login button has an onLogin listener attached to it. When the button is clicked/tapped an onLogin event is fired that sends a request to the server to check if the user is authenticated to view the report. If the user is authenticated then the app displays the wifi mapper report.

The report is used to display special statistics to the user. The app calculates the number of locations and signal strength then displays them at the top of the view. Number of WLAN ZONEs is also calculated

and displayed together with the number of locations.

Following the calculated numbers, the report displays bar ,pie and bubble charts for the data on firebase. The first bar chart shows the average wifi strength against the WLAN ZONEs on the map. The bar chart is followed by a pie chart that shows a different representation of the average wifi strength against the WLAN.

The app uses a library called chart.js to convert the data from javascript objects to charts. Calculations for the color are done by a single function for all the charts. The calculation is such the only bright colors are used for all chart components.The report view was developed using modern frameworks like bootstrap that makes the report responsive accross all devices.

Data Structure used

During Query Phase when the application is initially started data is obtained using different listener threads as data has to be obtained asynchronously from firebase and both individual Locations and WLAN Zones are stored each in a List Structure and a HashMap is created mapping each ID of WLAN-Zone with its actual representation through a Polygon object that will make it easier to update zones whenever new data is obtained that corresponds to that zone

On Location Added Event

Any change on the database will be caught by the listener threads and if its a location the object will be obtained and a marker will be added on the map at the corresponding location

on Area Changed Event

Any update on an area object will also be caught by listener and since the object might change but its ID is still same its used to query the HashMap and get the polygon object thats directly on the Map and change its color and signal strength to reflect the new average

User Interface

WLANZones are clearly defined on the map as essentially each zone covers a building and other common areas to students which will enable the user accurately know a WLAN Zone and its corresponding wifi Strength based on the color Scheme explained shown below

- Red : 0 - 29
- Orange : 30 - 49
- Yellow : 50 - 59
- Light Green : 60 - 79
- Green : 80 - 100

The colors were chosen as they easily map to what users are already used to in their day to day.The map used was a Satelilte Map as it did not have any extra clutter such as building names and their markers,as Zones in the application have names associated with them

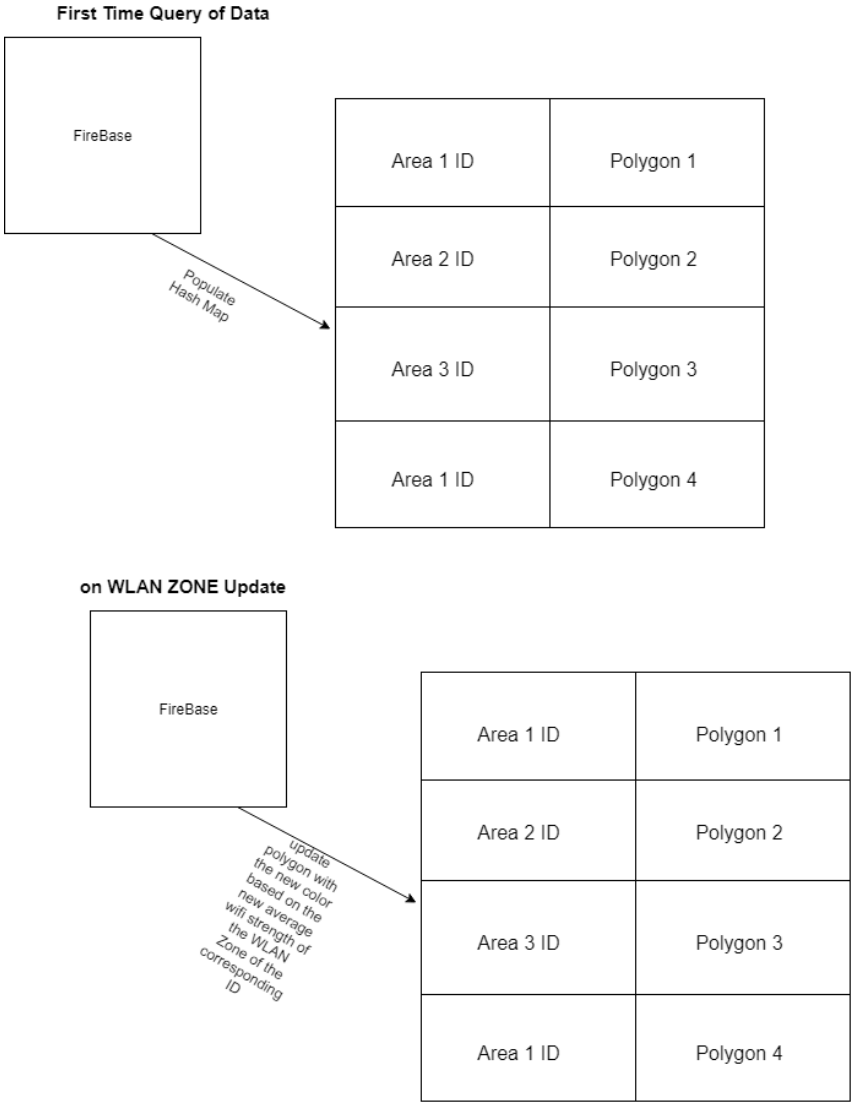

Figure 6: Area ID Polygon Mapping operations

4. Program Validation and Verification

Tell us how you tested the system and why you believe it works. Describe the Quality Management Plan for your project, that is, software testing plan. The plan should indicate the types of testing that was performed and detail how they were done. This must include the reasons on why the chosen testing protocol was considered effective.

Create a table that summarizes the testing plan (see Table 3).

Table 3: Summary Testing Plan. A table caption goes above the table.

Process	**Technique**
1. Class Testing: test methods and state behaviour of classes	Random, Partition and White-Box Tests
2. Integration Testing: test the interaction of sets of classes	Random and Behavioural Testing
3. Validation Testing: test whether customer requirements are satisfied	Use-case based black box and Acceptance tests
4. System Testing: test the behaviour of the system as part of a larger environment	Recovery, security, stress and performance tests

Describe all the steps taken to validate the correctness of the program.

If you had user tests then say what you did and what the results were. Describe why these test data were chosen (what test conditions the data was testing). Table 4 provides an example of the sorts of results we are looking for. The full detail of the test runs should be appended to the report.

Table 4: A table of tests. A table caption goes above the table.

Data Set and reason for its choice	**Test Cases**		
	Normal Functioning	*Extreme boundary cases*	*Invalid Data (program should not crash)*
Preliminary test (see Appendix 3)	Passed	n/a	Fell over

Follow your table of results with a discussions of them highlighting how useful and usable your system is for its intended purpose.

5. User Manual

5.1. Mobile Client(Android)

5.1.1. Getting started and Installation

To get started download and install the latest version of WiFi mapper from the website provided or from google play app store. If downloaded from the app store, the app will automatically install on your device. If downloaded from the website, tap on the apk file to begin installation. During installation WiFi mapper will ask for permissions to use location services of the device. Tap yes to allow the app to send data from your device. Tapping no will allow one to see the data collected by other users but will not allow the app to send the data to the database.

5.1.2. Map

On opening the app, a hybrid version of google maps is displayed on the screen. The map contains polygons on top of different buildings with colors depicting the wifi strength on respective WLAN ZONEs.

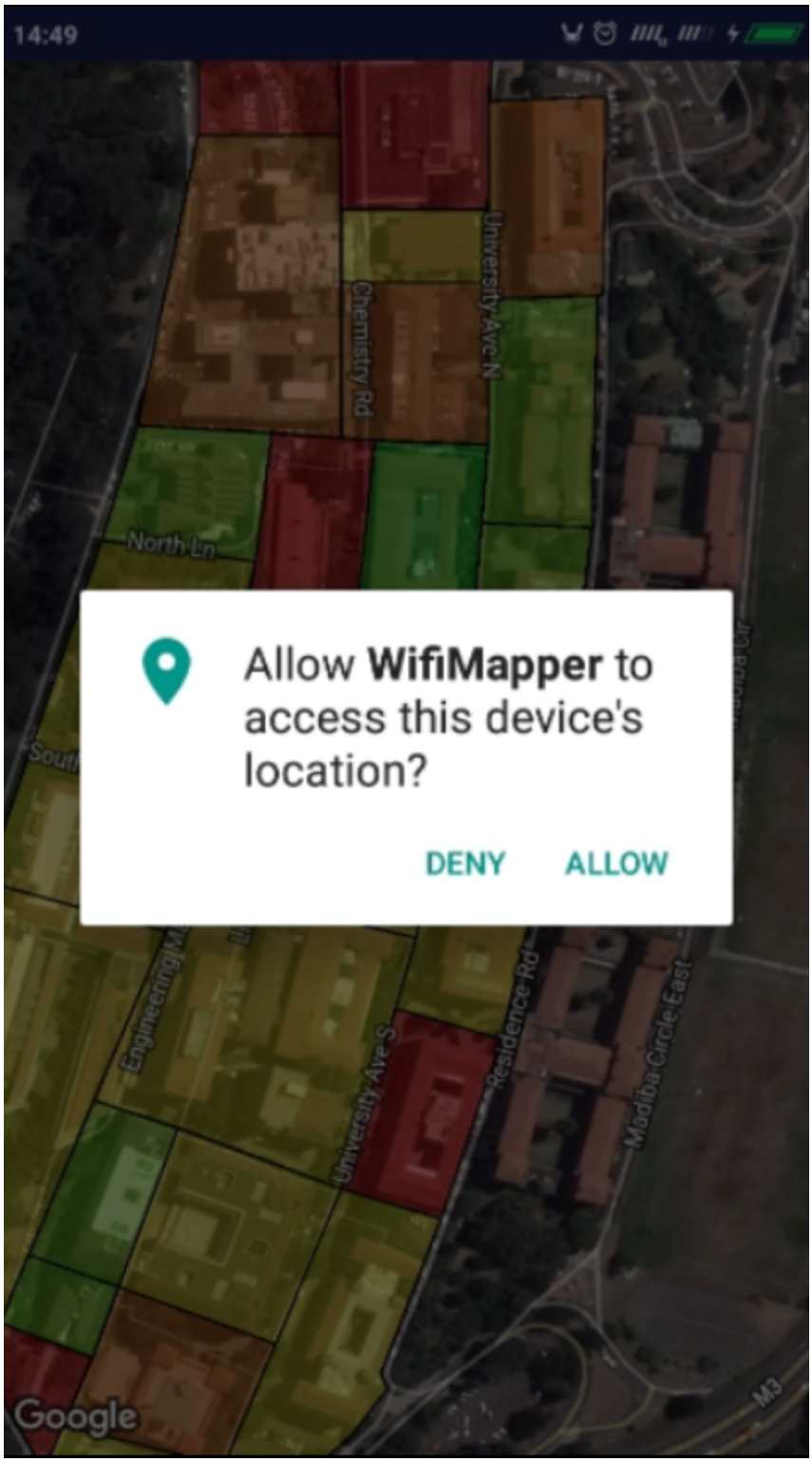

Figure 7: Asking For Permissions

The map is by default zoomed such that it is focused on the University Of Cape Town(UCT) upper campus.

5.1.3. Zoom

Wifi mapper allows users to zoomin and zoomout on the map. The app puts restrictions on the zoomout functionality so that the map is always focused at UCT upper campus. The zoomin functionality is also limited to a certain degree. To zoom in, pinch in with two fingers on the map screen. On zooming in the map replaces colored polygons with coloured data points. Zoomout by pinching out on the map screen,if the map was displaying datapoints it replaces those with coloured polygons.

5.1.4. Pan

To move to another area without zooming in, pan on the screen with a single finger. To move the map to the right push the map with a single finger to the left. To move the map to the left push on the map right using a single finger. Rate at which you push the map determines the rate at which the map moves in the specified direction.

5.1.5. Area tap

To get the name and wifi strength of an area a polygon is placed over, tap the screen once. On taping the screen the maps pans over to the tapped polygon or area.

5.2. Web Client

5.2.1. Getting started and Installation

To get started with the web client, open your browser and go to https://capestone-a8a58.firebaseapp.com/. Before going to the url first ensure that you are connected to the internet. If the network is slow, wait for the app to load fully.

5.2.2. Map

On opening the app, a hybrid version of google maps is displayed on the screen. The map contains polygons on top of different buildings with colors depicting the wifi strength on respective WLAN ZONEs. The map is by default zoomed such that it is focused on the University Of Cape Town(UCT) upper campus.

5.2.3. Zoom

Wifi mapper web client allows users to zoomin and zoomout on the map. The app puts restrictions on the zoomout functionality so that the map is always focused at UCT upper campus. The zoomin functionality is also limited to a certain degree. To zoom in, pinch in with two fingers on the map screen. On zooming in the map replaces colored polygons with coloured data points. Zoomout by pinching out on the map screen,if the map was displaying datapoints it replaces those with coloured polygons. On the webclient the zoomin and zoomout limits are different.

5.2.4. Pan

The WiFi mapper web client allows users to pan and navigate the map around. To pan use two fingers to move the map around. To move the map to the left push to the right using both fingers. To move them map to the right push on the map to the left with both fingers.

5.2.5. Area tap

The behaviour is the same as that of the android client

5.2.6. Generate Report

Wifi mapper web app provides capabilities to generate a report from the collected data. To generate the report, on the map click the Generate Report button at the top left corner of the map view. This will take you to the login page. To login the fill in the form fields and click login. If succesful you will be redirected to the report view that contains different charts for the data obtained from the database. On the

Figure 8: Showing the Map

Figure 9: zoomin

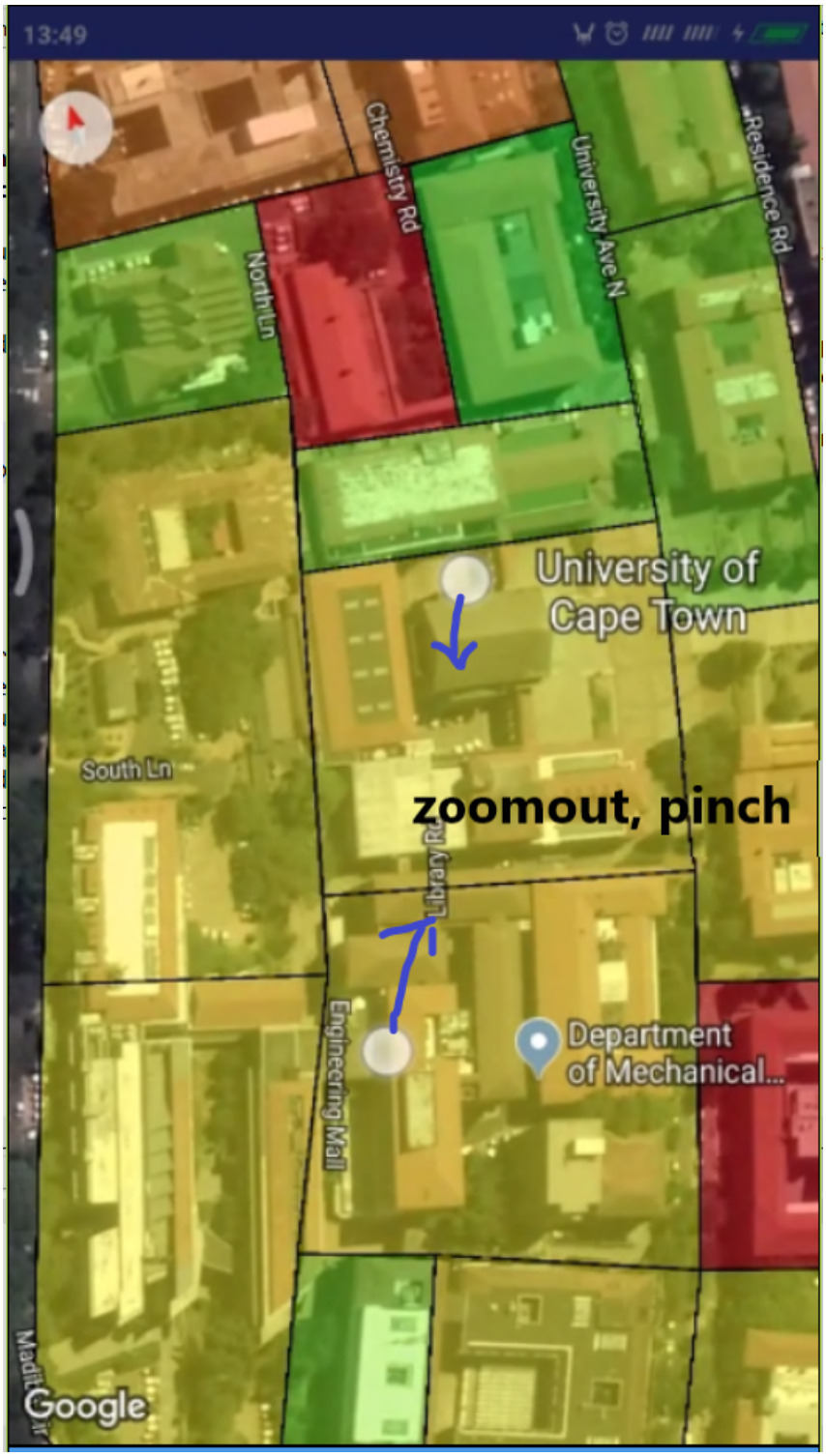

Figure 10: zoomout

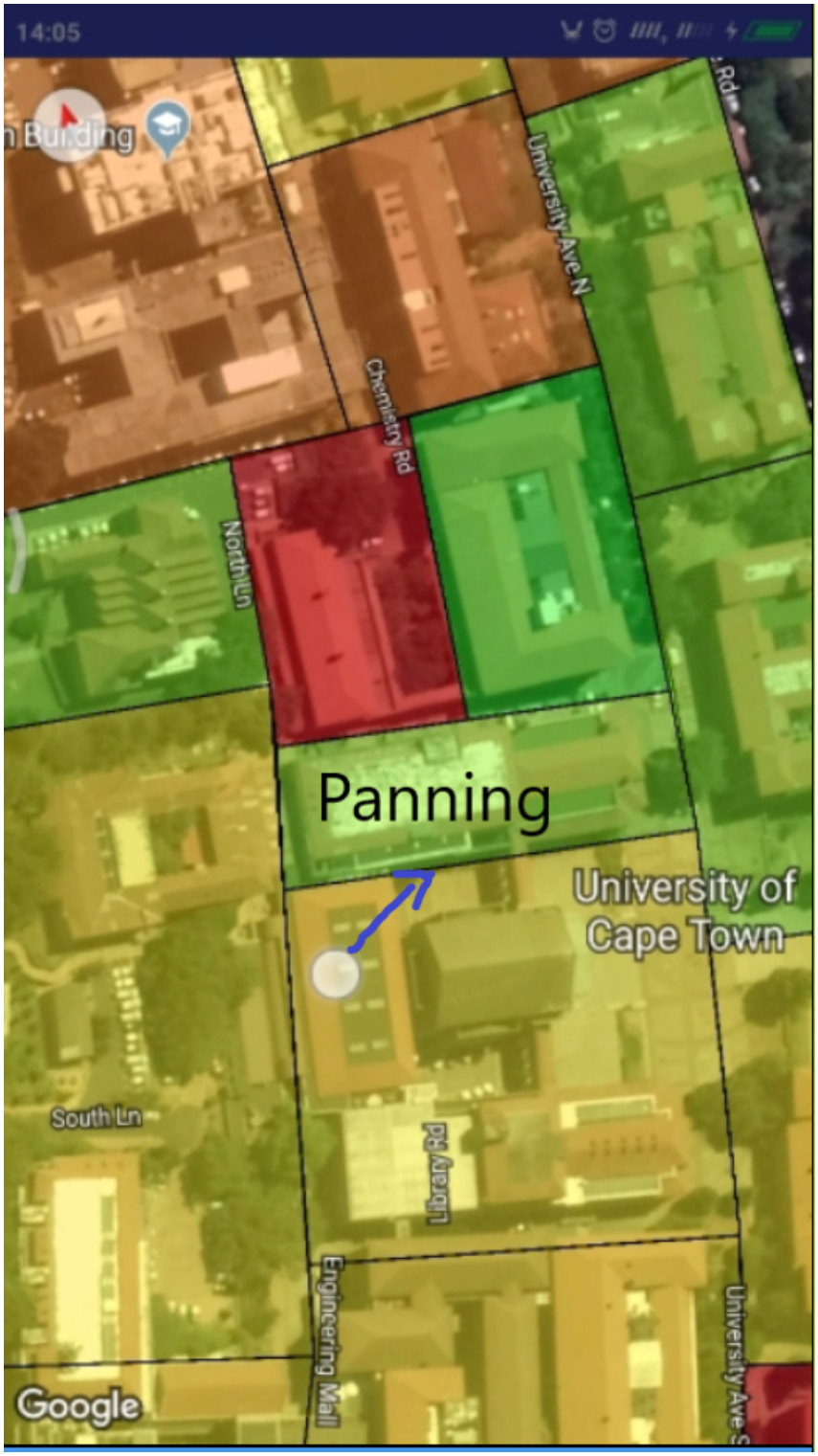

Figure 11: Panning

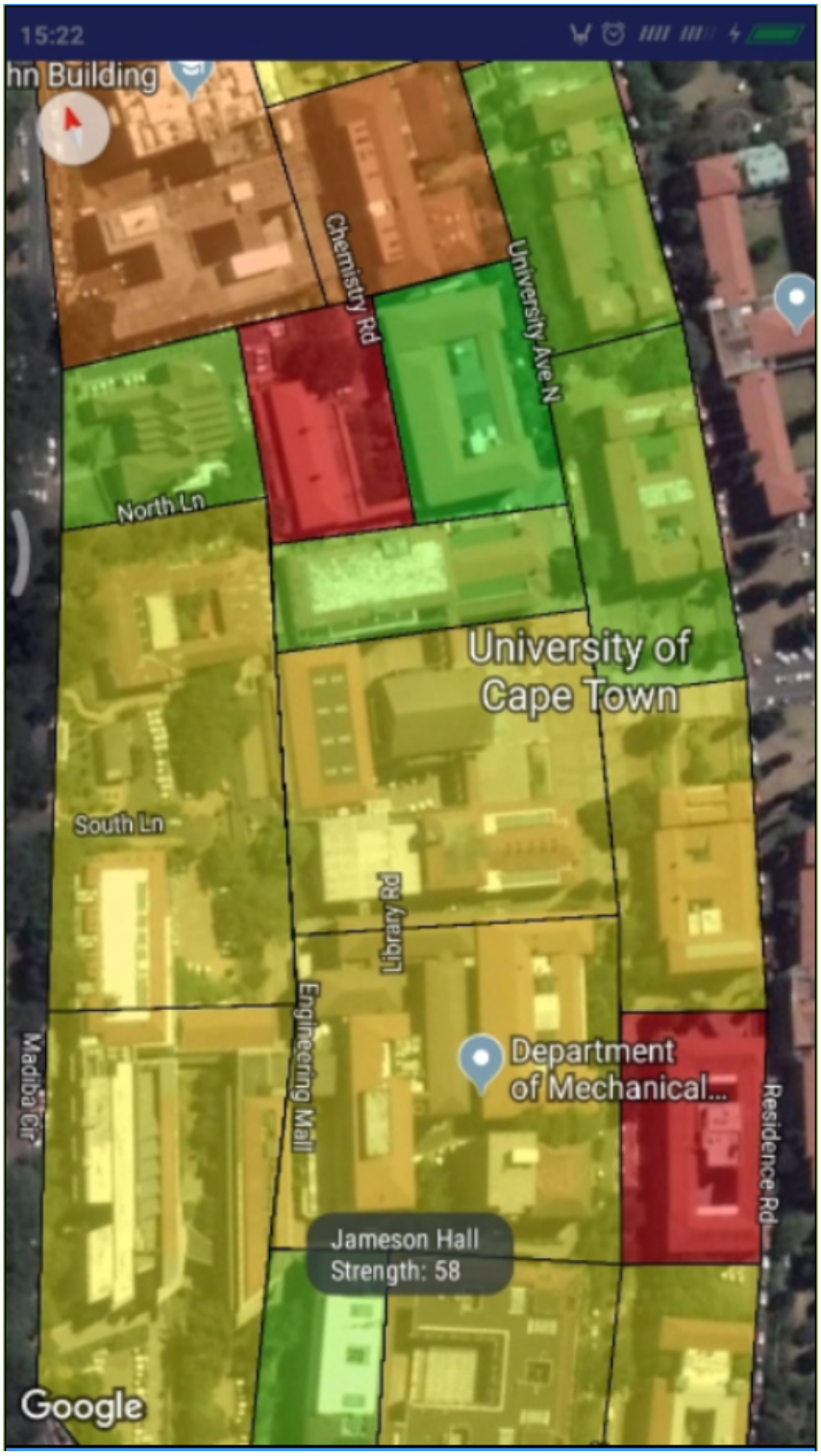

Figure 12: Tapping showing the name and strength of the tapped WLAN ZONE

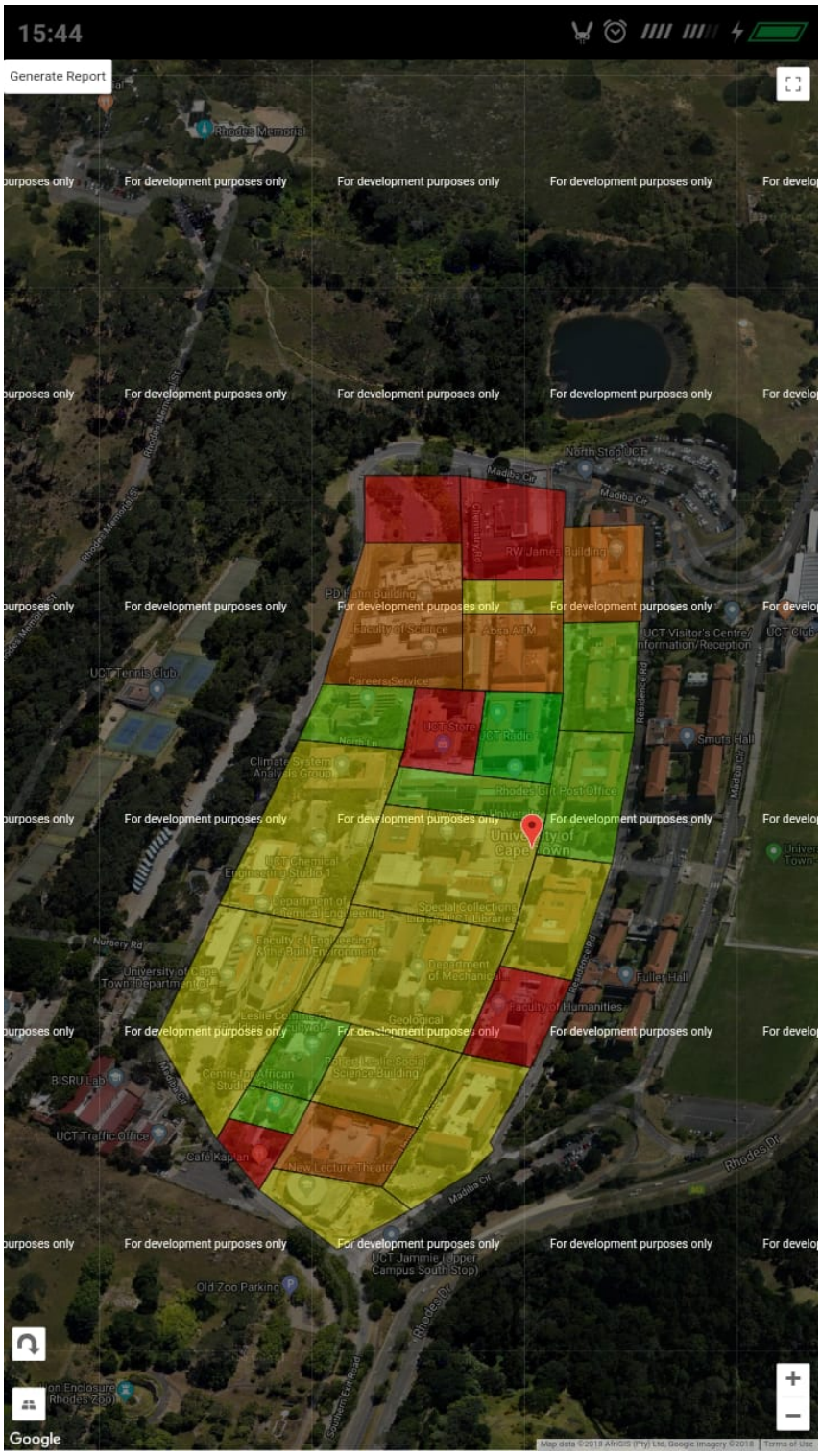

Figure 13: web client map

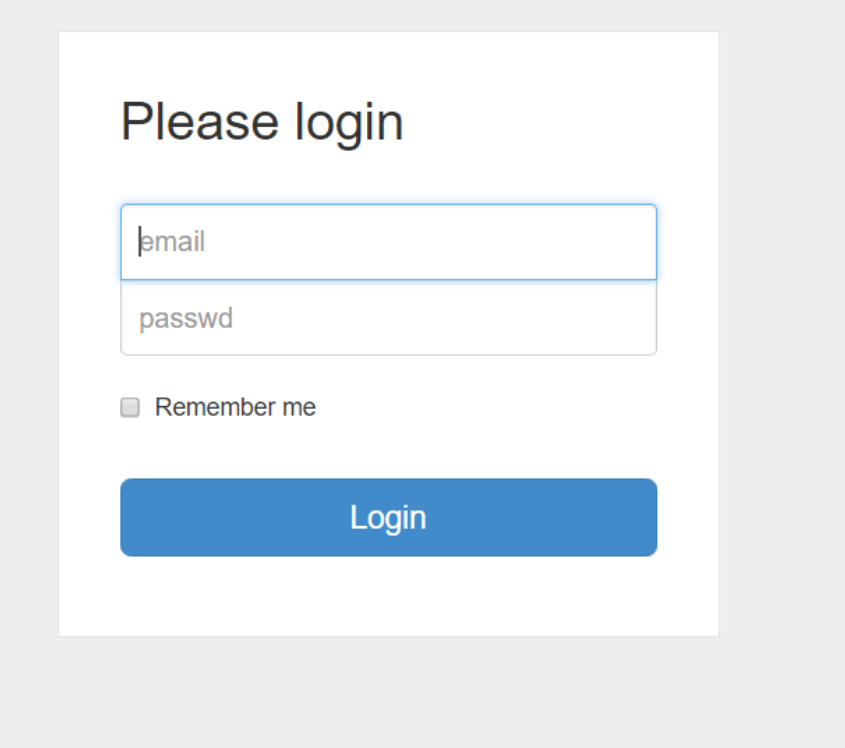

Figure 14: login page

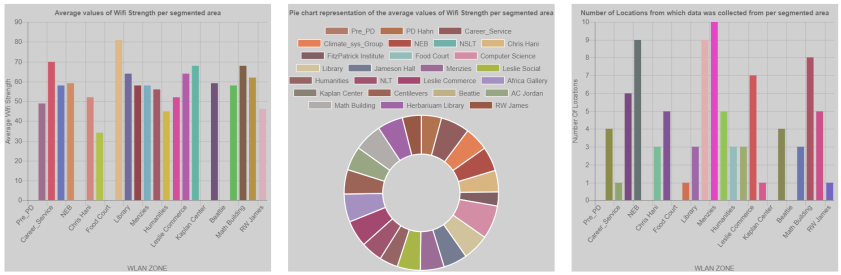

Figure 15: a part of the report page

report view you can interact by hovering over each element of the charts provided to get supplementary data of the chart.

6. Conclusion

This document has covered the major sections needed for your report. You will probably have each of the subsections 2.1–2.7 as major section in the report each with its own subsections.

A marking guide for the report will be provided later.

7. Code Legibility and Output

This is not strictly part of the report but is a requirement for the final hand-in.

- Each method should start wide a brief description of its function.
- Use indentation to display the structure within a method.
- Comments should be used extensively. They are best used to describe logical blocks of code rather than individual statements. Line-by-line comments have the drawbacks of not providing any overview and of decreasing readability.

- Meaningful identifiers should be chosen.
- Output should be pleasingly formatted and easy to read.

You do, of course, have the option to call in any of your favourite packages for setting maths, graphics, computer listings, etc.

References

[Kopka and Daly(2004)] Kopka, H. and Daly, P.W. (2004) *A Guide to LaTeX 2ε: Document Preparation for Beginners and Advanced Users* (4th edn). Addison-Wesley.

[Lamport(1994)] Lamport L. (1994) *LaTeX: A Document Preparation System* (2nd edn). Addison-Wesley.

[Mittelbach and Goossens(2004)] Mittelbach, F. and Goossens, M., (2004) *The LaTeX Companion* (2nd edn). Addison-Wesley.