**Chapter 1**

**Introduction**

**Introduction**

Employee Tracking Application is an application which is going to track the employee by getting his current location. To get the current location we have used the Google Map Api. Using this api we got the current location.

In our application there are two different options to login one is for admin and the other is for employee. The admin will add employee by entering their valid credentials such as name, email address, password, phone number, and id and will give the email address and password to the employee’s for login. Admin will get the employee’s latitude and longitude from the database to view the employee’s current location. The employee will login with the email address and password and will view his current location. He can also search for the places in the app. Employee will save his current location to the database.

While developing it we have faced many of the problems related to installation, gradle building. As the time passed we learnt how to fix the gradle related problems and any problem related to the dependency.

**Planning**

The most important activity while developing a software is planning. Planning is the basic activity, also this is considered as planning of steps undertaken to develop a project or a product.

In our project we have taken following steps:

1. Collect information required for project development.
2. Calculate estimate for project requirement.
3. Develop working module of Android.
4. Using Android Studio write necessary code.
5. For storing purpose link project to the database used.
6. Test the application.

**Project Objective**

The objective of this project is to track the location of employee so that the admin will be able to track employee.

In travelling agencies our application will be useful to track the driver is going to the specified places or not. Admin will get to know driver is visiting the places as specified if not then admin can take actions.

**Feasibility Study**

Conducting a feasibility study is one of the key activities within the project initiation phase. It aims to analyze and justify the project in terms of technical feasibility, business viability and cost-effectiveness.

A **Feasibility Study Report (FSR)** is a formally documented output of feasibility study that summarizes results of the analysis and evaluations conducted to review the proposed solution and investigate project alternatives for the purpose of identifying if the project is really feasible, cost-effective and profitable. It describes and supports the most feasible solution applicable to the project.

This project can be implemented using affordable software and hardware technology making it economically, technically and operationally feasible.

**Economic Feasibility:**

This Employee Tracking project is an Android application which does not involve any extra things only requires android studio which is affordable and which make this project economically more feasible.

**Technical Feasibility:**

This project is based on android, which is world wide used technology. Therefore it is very much favored by the technology.

**Operational Feasibility:**

The Android application used to track the location of employee is very easy to use, it has user friendly interface so it will be easy to operate by anyone having little experience of using android phone. So it is operationally feasible.

**Chapter 2**

**Analysis**

**Problem Statement**

This system consist of android application where the user will be using the android application and admin as well as HR will also work with android application. All the data such as name of user, his/her e-mail address, password, phone no. and employee’s employee id.

This application is meant for field work Employers. The Employee will have this application in his android phone, when the user will login to the system then user is going to save the current location and this current location co-ordinates will be stored to the database. Only Admin can access that database to validate the employee.

In order to keep track of the attendance as well as payroll of the employee, this system plays a major role. The role of the admin is to add new employee by entering his personal details and admin will provide the employee with identity number and password to the user so that he can access the application in his android phone. Admin can view latitude and longitude of the GPS location sent by the employee. Admin can change the password of the employee. Since GPS location of the employee is tracked, so employee will not attempt to add proxy attendance.

**Software/hardware requirements**

Hardware Requirements:

1. Min Processor: i3 4th gen
2. Ram: 4GB
3. HDD Space: max 10 GB free
4. Internet Connection

Software Requirements:

Main software required for developing android projects is android studio.

**Android Studio**

Android studio is the official integrated development environment(IDE) for Google’s Android operating system, built on JetBrains’ IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (ADT) as the primary IDE for native Android Application development.

**Android Studio Installation**

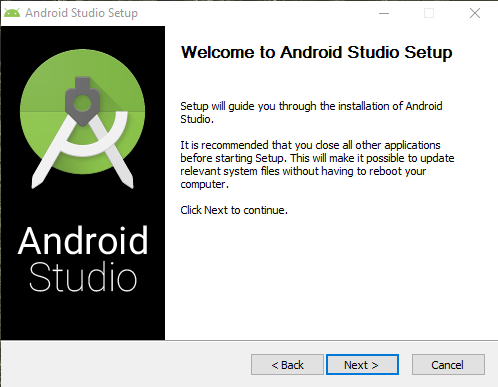
1. **Download setup file for android studio**

You can download Android studio from the Android studio homepage or u can directly visit the following link.

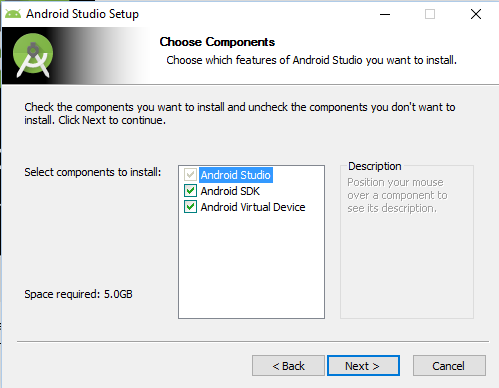
[**https://www.googleadservices.com/pagead/aclk?sa=L&ai=DChcSEwiZ8Lmly5zhAhXrle0KHXLbAZEYABABGgJkZw&ei=3mqYXOnrB4mU1fAPr42\_8Ac&ohost=www.google.com&cid=CAESEeD2A0lKhaU\_7Rm0XeiwG3Kp&sig=AOD64\_3vITlEcQpTiDHiqEXbbqsXBNrrUQ&adurl=&q=&ved=2ahUKEwjp0q-ly5zhAhUJShUIHa\_GD34QqyQoAHoECAoQBA**](https://www.googleadservices.com/pagead/aclk?sa=L&ai=DChcSEwiZ8Lmly5zhAhXrle0KHXLbAZEYABABGgJkZw&ei=3mqYXOnrB4mU1fAPr42_8Ac&ohost=www.google.com&cid=CAESEeD2A0lKhaU_7Rm0XeiwG3Kp&sig=AOD64_3vITlEcQpTiDHiqEXbbqsXBNrrUQ&adurl=&q=&ved=2ahUKEwjp0q-ly5zhAhUJShUIHa_GD34QqyQoAHoECAoQBA)

1. **Run the .exe file:**

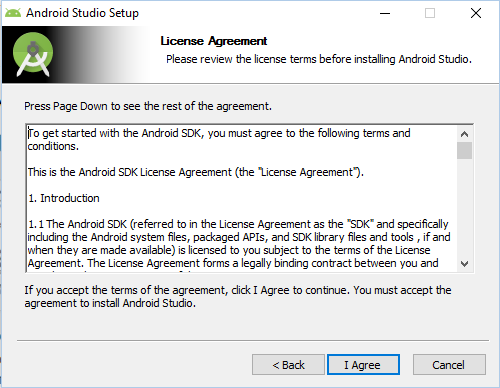
Now launch the .exe file .It will ask to provide Administrative access.



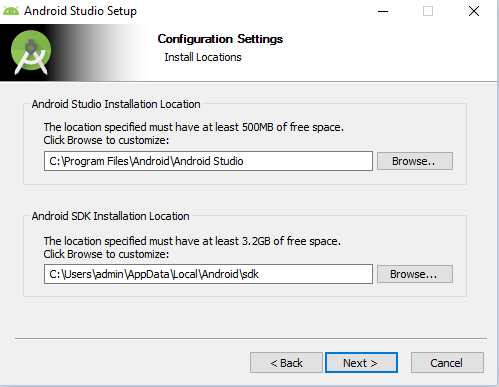
1. **Click next following window will popup [Choose Components ]**



1. **Accept the license agreement and click on ‘I Agree’**

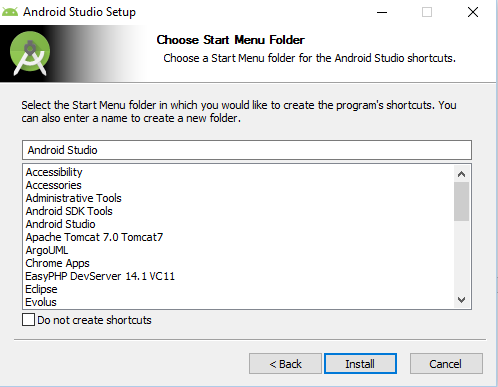


1. **Next step is to set location of installation.**

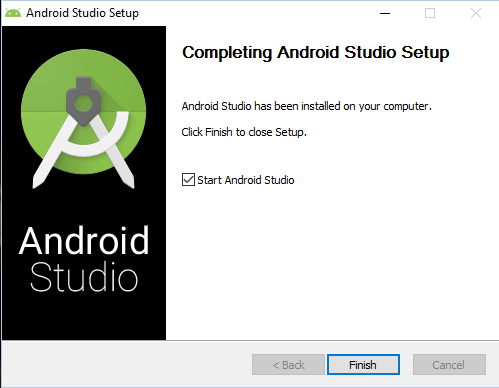


**6. Next step is to choose the start menu folder, where you want to create shortcut. If you don’t want to create a shortcut just mark. Do not create shortcut.**

**[Choose start menu folder]**



**7. Hit Install button.**



**Chapter 3**

**Design**

**I/O Design**

For taking input from employee various buttons, edittext, Textviews are used.

Output consist of stored information of employee and for that no extra work was needed because of the database used.

It has its own output design in which admin can see the details of employee.

**UI Design**

UI means what user sees when he first opens the app in his smartphone.

We used or developed a simple UI which will not create any confusion while using the app.

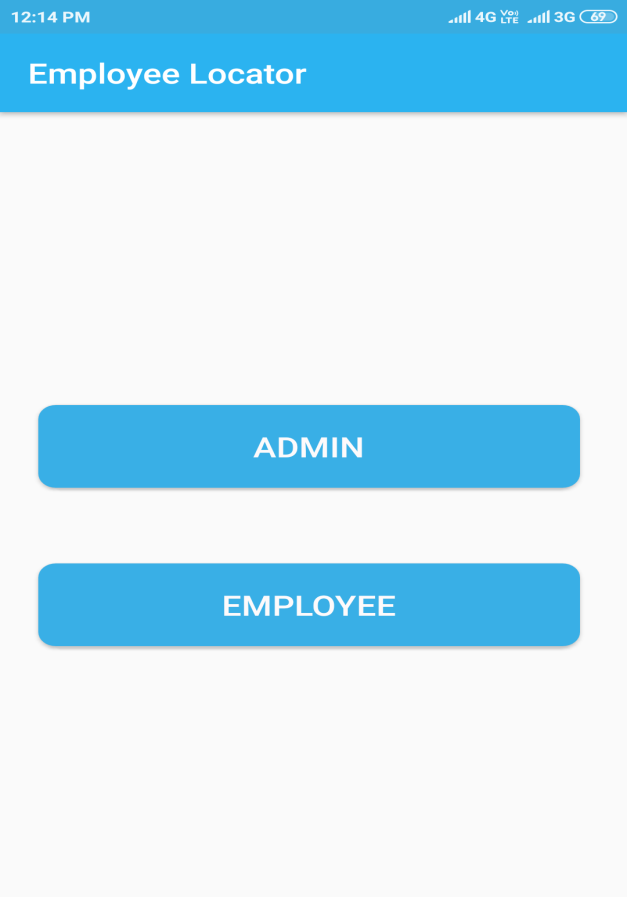
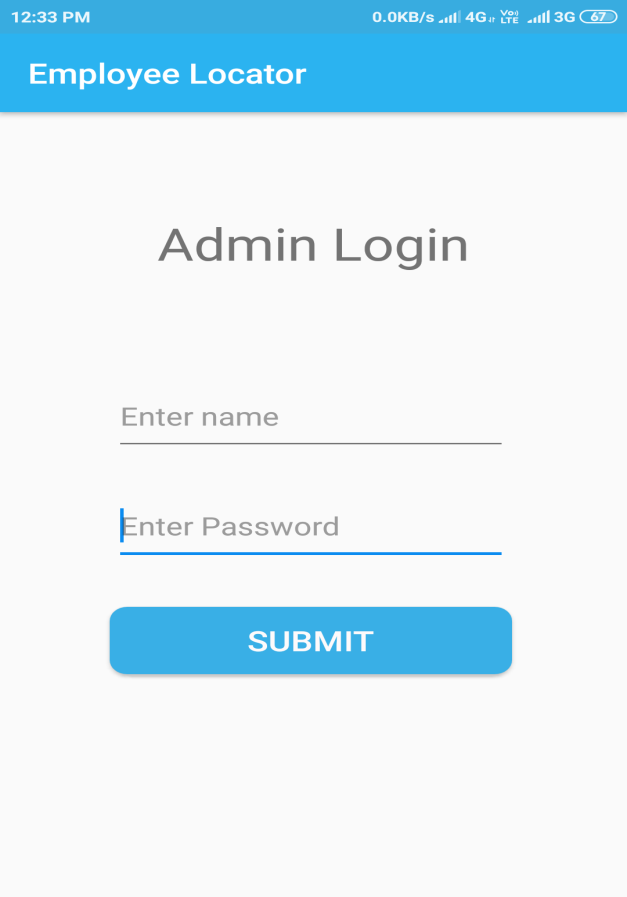
When we open the app the first activity displays two buttons one for Admin and the other for Employee.

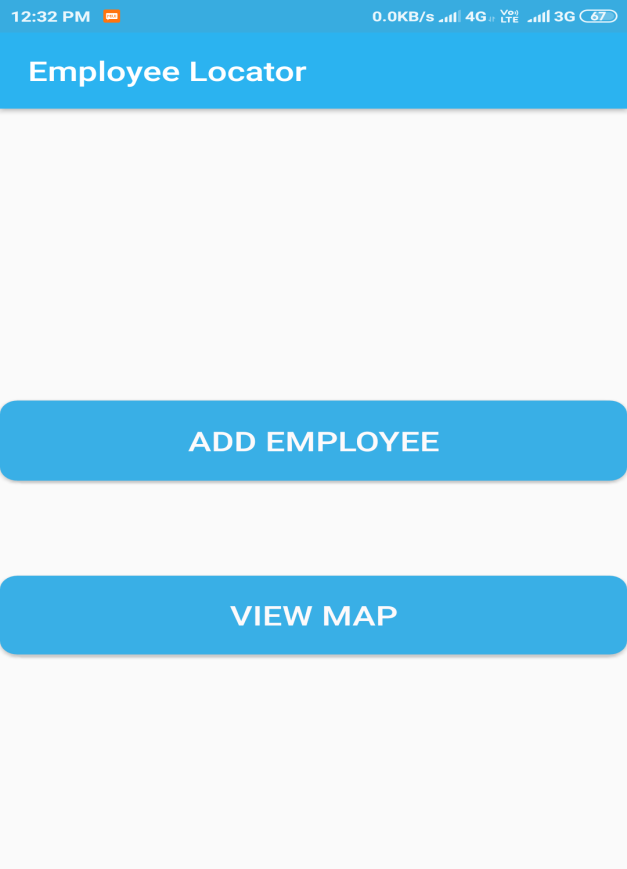
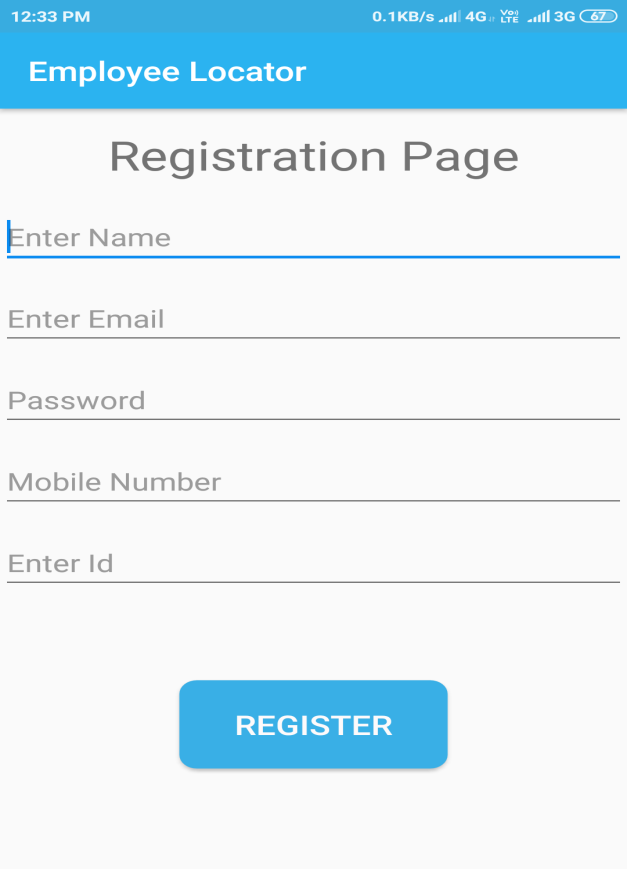
Admin will login and will have a register activity and one map activity to get employee location.

Employee will login and will have a maps activity with current location and will have another activity to save information.

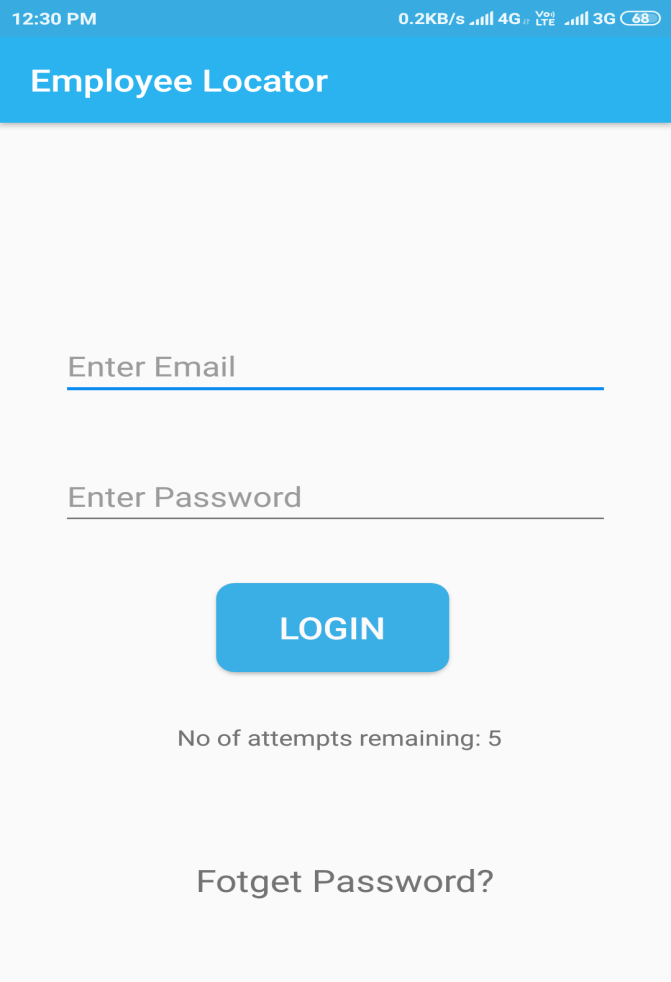
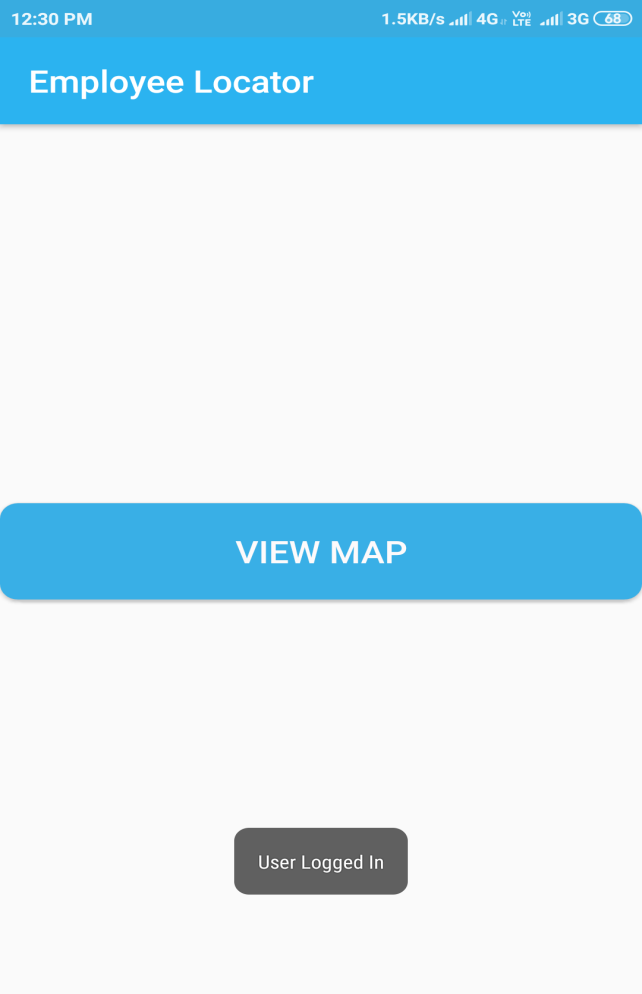
**Interface of App**

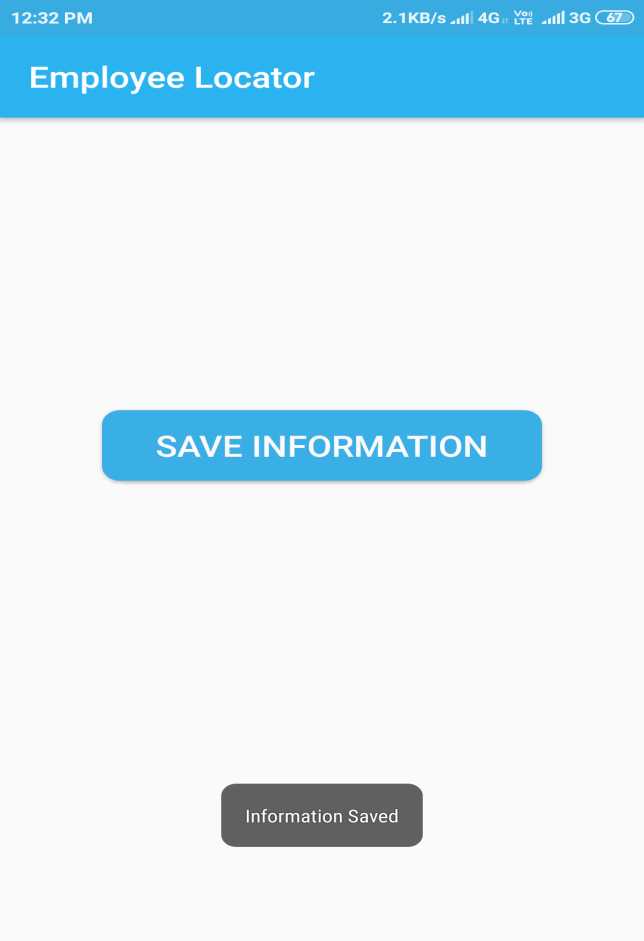
**Admin Interface**

** **

** **

**Employee Interface**

** **

** **

**Database Design:**

The Database we used for our application is firebase. The Firebase Real-time Database is a cloud-hosted database. Data is stored as JSON and synchronized in real-time to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Real-time Database instance and automatically receive updates with the newest data.

**Key capabilities**

**Real-time**

Instead of typical HTTP requests, the Firebase Real-time Database uses data synchronization—every time data changes, any connected device receives that update within milliseconds. Provide collaborative and immersive experiences without thinking about networking code.

**Offline**

Firebase apps remain responsive even when offline because the Firebase Real-time Database SDK persists your data to disk. Once connectivity is reestablished, the client device receives any changes it missed, synchronizing it with the current server state.

**Accessible from client devices**

The Firebase Real-time Database can be accessed directly from a mobile device or web browser; there’s no need for an application server. Security and data validation are available through the Firebase Real-time Database Security Rules, expression-based rules that are executed when data is read or written.

**How does it work?**

The Firebase Real-time Database lets you build rich, collaborative applications by allowing secure access to the database directly from client-side code. Data is persisted locally, and even while offline, real-time events continue to fire, giving the end user a responsive experience. When the device regains connection, the Real-time Database synchronizes the local data changes with the remote updates that occurred while the client was offline, merging any conflicts automatically.

The Real-time Database provides a flexible, expression-based rules language, called Firebase Real-time Database Security Rules, to define how your data should be structured and when data can be read from or written to. When integrated with Firebase Authentication, developers can define who has access to what data, and how they can access it.

The Real-time Database is a NoSQL database and as such has different optimizations and functionality compared to a relational database. The Real-time Database API is designed to only allow operations that can be executed quickly. This enables you to build a great real-time experience that can serve millions of users without compromising on responsiveness. Because of this, it is important to think about how users need to access your data and then structure it accordingly.

# Add Firebase to your Android project

**Adding Firebase using the Firebase Assistant**

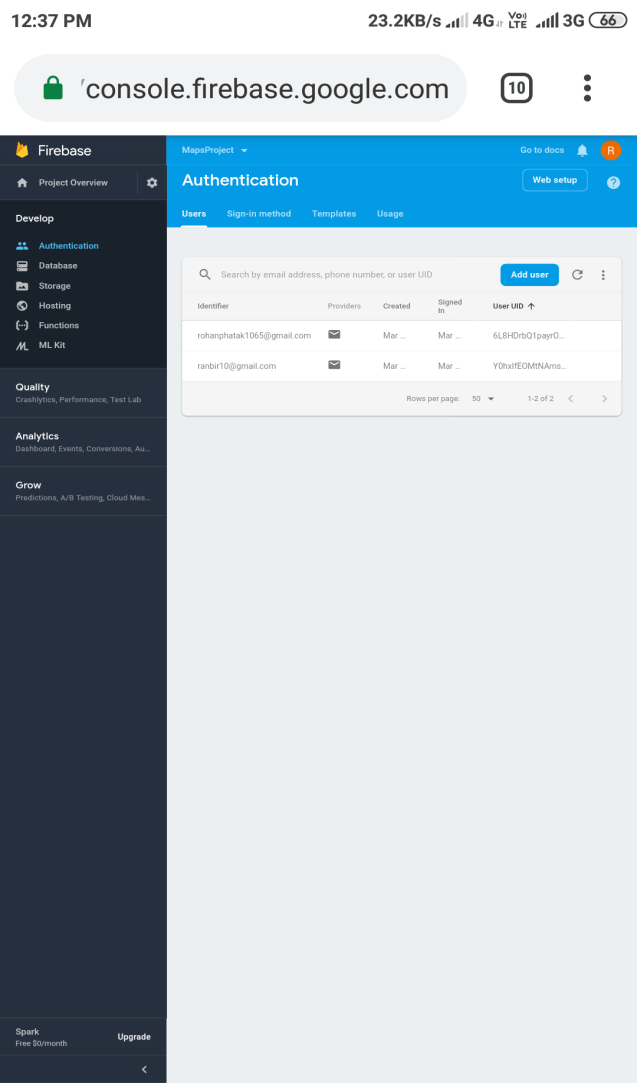
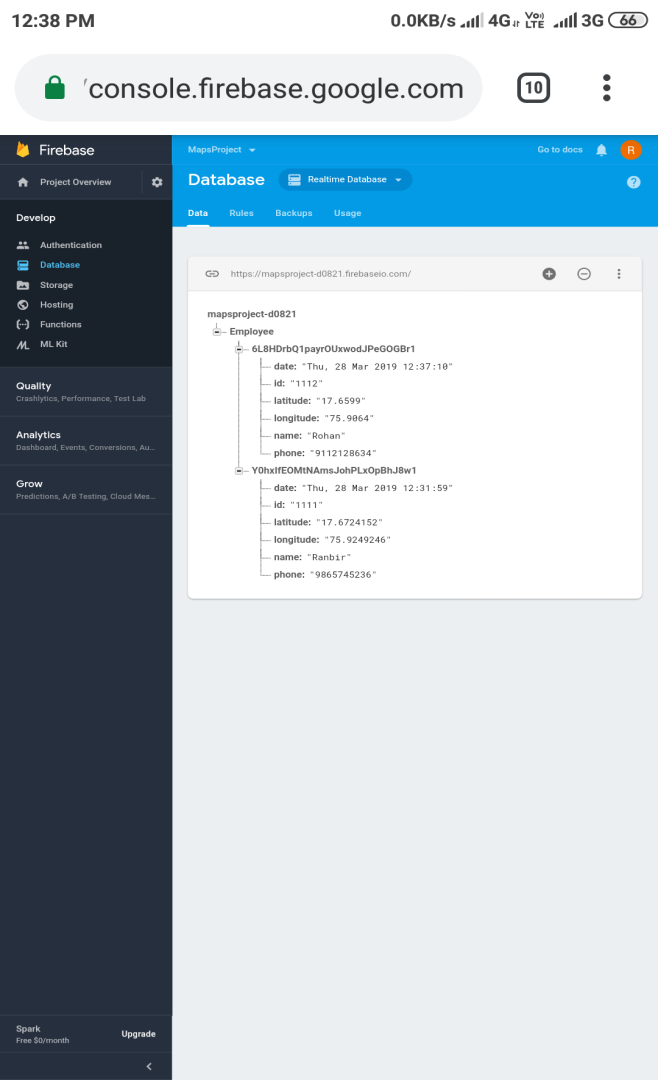
The Firebase Assistant registers your app with a Firebase project and adds the necessary Firebase files and code to your Android project all from within Android Studio.

1. Open your Android project in Android Studio.
2. Select **Tools > Firebase** to open the **Assistant** window.
3. Expand one of the listed Firebase products (for example, Analytics), then click the provided tutorial link (for example, Log an Analytics event).
4. Click **Connect to Firebase** to register your app with an existing or new Firebase project and to automatically add the necessary files and code to your Android project.
5. Check that your plugin and library versions are up-to-date:

* In your root-level (project-level) Gradle file (build.gradle), check that your Google Services plugin version is up-to-date (com.google.gms:google-services:4.2.0).
* In your module (app-level) Gradle file (usually app/build.gradle), check that your Firebase Android library versions are up-to-date.
* **Note:** If you're using Android Studio v3.2 or earlier, also make sure that each dependency line only has one version number specified.

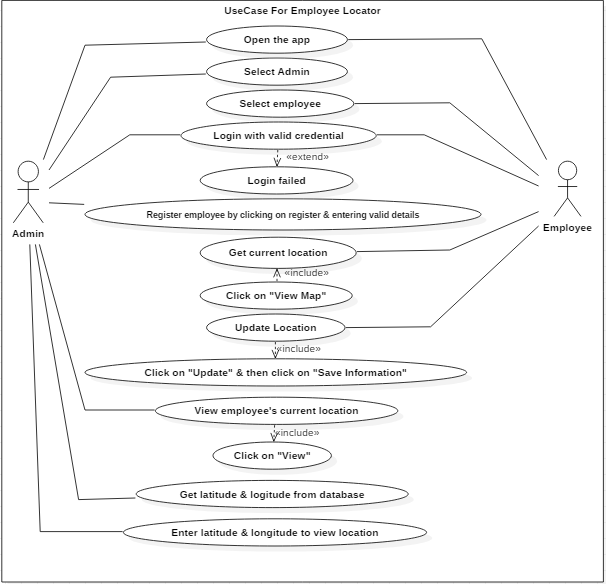
1. Sync your app to ensure that all dependencies have the necessary versions.
2. Configure your Analytics data sharing settings in the Firebase console [**Project settings**](https://console.firebase.google.com/project/_/settings/privacy/).   
   Enabling the sharing of Analytics data with other Firebase products is required to use Firebase products like Firebase Predictions or Firebase A/B Testing.
3. Run your app to send verification to Firebase that you've successfully integrated Firebase.

* Your device logs will display the Firebase verification that initialization is complete. If you ran your app on an emulator that has network access, the Firebase console notifies you that your app connection is complete.

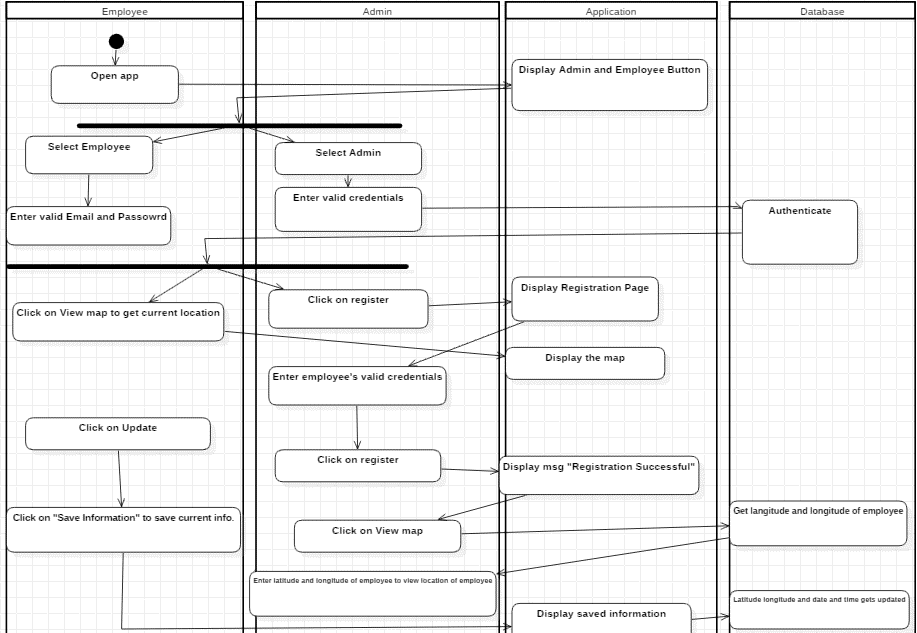
 

**List of Diagrams:**

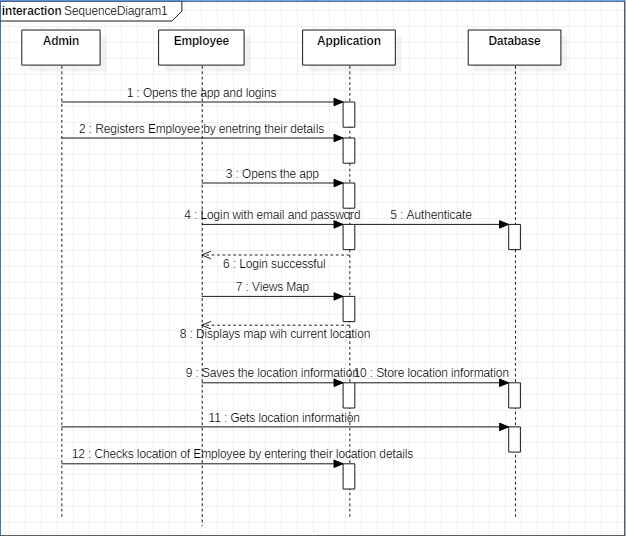
**1) UseCase Diagram**

****

**2) Activity Diagram**



**3) Sequence Diagram**

****

**Chapter 4**

**Coding**

**Languages / tools used:**

Our project is android based so we have used android studio which supports java language. Java language has many advantages over other language.

**Justification for Selection of languages/tools:**

Now for our project we have used Java.

The reason behind selecting Java is that it is the standard and Java is the official language of Android development and is supported by Android Studio. It has a steep learning curve however.

**Important modules**

**The project consist of various modules:**

1. Module for search functionality
2. Module for storing information into database
3. Module which shows google map along with current location of employee marked

**Application Programming Interface [API]**

An API is a set of methods and tools that can be used for building software application.

**Google Map API**

We have used google map api key in our app to access the google maps server. This api key should be restricted for android apps.

**Steps to create Google Maps Project**

1. Start Android Studio.
2. Create a new project as follows:

* If you see the **Welcome to Android Studio** dialog, choose **Start a new Android Studio project**, available under 'Quick Start' on the right of the dialog.
* Otherwise, click **File** in the Android Studio menu bar, then **New**, **New Project**.

1. Enter your app name, company domain, and project location, as prompted. Then click **Next**.
2. Select the form factors you need for your app. If you're not sure what you need, just select **Phone and Tablet**. Then click **Next**.
3. Select **Google Maps Activity** in the 'Add an activity to Mobile' dialog. Then click **Next**.
4. Enter the activity name, layout name and title as prompted. The default values are fine. Then click **Finish**.

Android Studio starts Gradle and builds your project. This may take a few seconds. When the build is finished, Android Studio opens the google\_maps\_api.xml and the MapsActivity.java files in the editor. (Note that your activity may have a different name, but it will be the one you configured during setup.) Notice that the google\_maps\_api.xml file contains instructions on getting a Google Maps API key before you try to run the application. The next section describes getting the API key in more detail.

**Steps to get Google Maps API Key**

Use the link provided in the google\_maps\_api.xml file that Android Studio created for you:

1. Copy the link provided in the google\_maps\_api.xml file and paste it into your browser. The link takes you to the Google Cloud Platform Console and supplies the required information to the Google Cloud Platform Console via URL parameters, thus reducing the manual input required from you.
2. Follow the instructions to create a new project on the Google Cloud Platform Console or select an existing project.
3. Create an Android-restricted API key for your project.
4. Copy the resulting API key, go back to Android Studio, and paste the API key into the <string> element in the google\_maps\_api.xml file.

**Searching**

For search we have used geocoder class.

Geocoding is the process of transforming a street address or other description of a location into a (latitude, longitude) coordinate. Reverse geocoding is the process of transforming a (latitude, longitude) coordinate into a (partial) address. The amount of detail in a reverse geocoded location description may vary, for example one might contain the full street address of the closest building, while another might contain only a city name and postal code. The Geocoder class requires a backend service that is not included in the core android framework. The Geocoder query methods will return an empty list if there no backend service in the platform. Use the isPresent() method to determine whether a Geocoder implementation exists.

**Method of GeoCoder class**

List<Address> getFromLocationName([String](https://developer.android.com/reference/java/lang/String.html) locationName, int maxResults)

Returns an array of Addresses that are known to describe the named location, which may be a place name such as "Dalvik, Iceland", an address such as "1600 Amphitheatre Parkway, Mountain View, CA", an airport code such as "SFO", etc.

**Chapter 5**

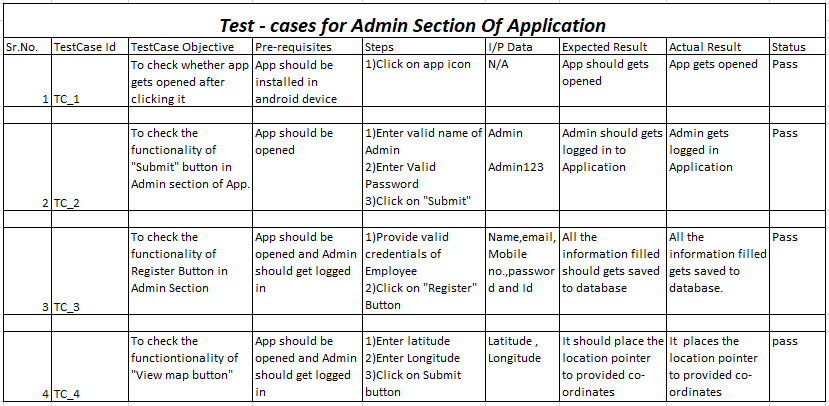
**Testing**

**Test Plan**

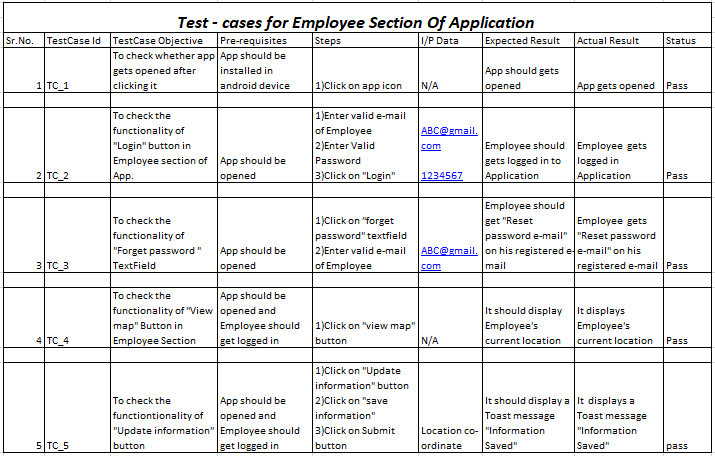
For the purpose of testing we decided that running the application in many mobile devices having different android versions. Running app in many versions proves that the developed app is compatible with all the versions which are above 4.2.2(Jellybean).

**Test Cases**

Admin



Employee



**Chapter 6**

**Reference**

**Guides:**

Our project guide helped us to improve the overall application by adding new features such as disabling of login button after 5 tries and sending link to an employee in case he/she forgets his/her login password.

**Websites:**

We have taken reference of following websites:

1. [www.developer.android.com](http://www.developer.android.com)
2. [www.firebase.google.com](http://www.firebase.google.com)
3. [www.stackoverflow.com](http://www.stackoverflow.com)
4. [www.quora.com](http://www.quora.com)
5. [www.androidauthority.com](http://www.androidauthority.com)

**Chapter 7**

**Conclusion**

We have successfully developed an android application which gives current location of employee using it.

Also we can change the password either by telling admin or by oneself.

By using our app admin can track the location of the employee.