

# Chapter 1

## INTRODUCTION

### 1.1 Overview

Bike Tracking system is getting popular and widely used in a lot of countries worldwide. It has tons of advantages to users even more to the bike users in which it will make it easier for them to track their bikes. Nowadays, everyone cannot be separated from their smartphones. a number of five thousands individuals from USA, UK, South Korea, India, China, South Africa, Indonesia and Brazil took a survey regarding which was done by Time magazine. The result proved most of them is inseparable from their smartphones, eighty four per cent allegedly claimed that survive without their smartphones.

Another study shows that seventy five per cent of the market share is smartphone and a total of one hundred and six million smartphone were shipped in the second half of 2012. Smartphone became the top telecommunication medium in the market in the present time worldwide and it became the most popular used telecommunication medium known to man. So, from the above mentioned survey now it's clear that how smartphones became important and integral part of our modern day life, that's the reason to make this bike rental tracking system text message oriented so that we can take bike of rented bike in just one touch of our hand. Through smart phone we can track real time location of our bike with the help of internet connection. In such a manner, this tracking system designed so that users can have easy and user friendly interface to fetch their bike.

In recent years, the popularity of rental bikes has skyrocketed as an eco-friendly and convenient mode of transportation in many cities around the world. With the increasing demand for rental bikes, the need for effective management and tracking systems has become crucial for both rental service providers and users. This is where GPS (Global Positioning System) tracking technology comes into play.

GPS tracking allows rental bike companies to monitor and manage their fleet of bikes more efficiently. By equipping each bike with a GPS device, the rental service provider can accurately track the bikes' locations in real-time. This technology offers a wide range of benefits for both the rental service provider and the users

## 1.2 Plan of Report

The main plan of the bike rental tracking system is to track the rented bike given to other users/customers .

The following are the major objective of this application

To provide a bug-free application.

The main objective is to build a secured, robust Gps tracking of rental bikes System

The application will be developed using Android studio and back end will be managed in Firebase database. Application will have easy and feasible GUI for all type of users. User needs to Login at the initial phase, set his/her profile details including location, choices, email-id, etc. User can modify or change His/her profile at any stage.

## **Chapter 2**

### **SYSTEM STUDY**

#### **2.1 Existing System**

In the present day bike tracking is becoming essential for the purpose of improving our life condition. Convenience and ease of using bike is what home bike tracking is offering. Bike tracking offers a futuristic way of life in which an individual gets to control his bike using a smart phone, from tracking a bike /detecting accidental place of a bike; it also offers an efficient use of technology. But to get or acquire such system installed will cost a lot of money and that is the major reason of why bike tracking has not received much demand and attention, adding to that also the complexity of installing it and configuring it. Thus it is essential to make it cost effective and easy to configure, if this is granted to people then they will be willing to acquire it in their personal bikes, school buses and taxis/cabs etc. In other words, a system modification for the bike tracking is required in order to lower the price of applying it to bikes. Also this tracking project can be used to purpose of women safety as well as parents can be used to take bike of their child/kid for the safety or missing purpose or to track their activities for their future. 5 Even more realistically this project can be used to track airline baggage because as we know every year almost 13% airline baggage used to get missing by a worldwide survey.

#### **2.2 Pitfalls in the Existing System**

- It is time consuming.
- It needs lot of human effort.
- More time for accessing.
- Possibility of proxies.
- Possibility for human errors.

## 2.3 Proposed System

The proposed system is used for positioning and navigating the bike. The Exact location is indicated in the form of latitude and longitude along with the exact Navigated track on Google map. The system tracks the location of particular bike and sends to user's mobile in form of data and also to microcontroller. The arrived data, in the form of latitude and longitude is used to locate the bike on the Google maps.

## 2.4 Objective of the Project

So one of our main objectives is to create a platform that can improve our user's experience and boost the chance of our creators to reach the maximum of the audience as well as facilitate the access and the funding process for our funders, and provide a place where they can build there trust and look for promising talents and empowering entrepreneurs to help in creating new possibilities.

Firstly, we need to make sure that the code is maintainable for future alterations and additions by other programmers. For this, we will try to keep the files organized in a logical setup and we will comment our code where necessary, to understand its working

Tracking in India is mainly used by transport systems, taxi companies, traffic operators. Taxi operators use this to estimate how far the bike is from a particular area and send this information to call centres' and they can inform general public about the distance of the taxi location and time it takes tom come to them.

The main objective of this project is to track the bikes rented to the users or customers.

## **Chapter 3**

### **SYSTEM DESIGN AND IMPLEMENTATION**

#### **3.1 System Requirements**

System requirements are the configuration that a system must have in order for a hardware or software application to run smoothly and efficiently. Failure to meet these requirements can result in installation problems or performance problems. The former may prevent a device or application from getting installed, whereas the latter may cause a product to malfunction or perform below expectation or even to hang or crash.

##### **3.1.1 Software Requirements**

There is use of large software in developing this project.

Software is basically the logical program that handles different components which cannot be touched or felt and helps to interact with one another in a Hassle-free manner.

The software used here consists of:

- Operating system : Windows 10.
- Coding Language : JAVA.
- Tool : android studio
- Database : firebase

### 3.1.2 Hardware Requirements

Hardware basically refers to the item in a PC that can be touched and felt like keyboard, monitor, mouse and the system unit.

Additional hardware components that can be added to the PC are modem, printer, scanner etc.

The hardware thus used here are consists of:

- System : Pentium i3 Processor,mobile
- Hard Disk : 500 GB.
- Monitor : 15” LED
- Input Devices : Keyboard, Mouse
- Ram : 4 GB

### 3.2 About the Software

Software is a set of instructions, data or programs used to operate computers and execute specific tasks. It is the opposite of hardware, which describes the physical aspects of a computer. Software is a generic term used to refer to applications, scripts and programs that run on a device. It can be thought of as the variable part of a computer, while hardware is the invariable part.

The two main categories of software are application software and system software. An application is software that full fills a specific need or performs tasks. System software is designed to run a computer's hardware and provides a platform for applications to run on top of.

## What is JAVA?

Java is a widely-used programming language for coding web applications. It has been a popular choice among developers for over two decades, with millions of Java applications in use today. Java is a multi-platform, object-oriented, and network-centric language that can be used as a platform in itself. It is a fast, secure, reliable programming language for coding everything from mobile apps and enterprise software to big data applications and server-side technologies.

All programming languages are a means to communicate with machines. Machine hardware only responds to electronic communication. High-level programming languages like Java act as a bridge between human language and hardware language.

Java is a widely-used programming language for coding web applications. It has been a popular choice among developers for over two decades, with millions of Java applications in use today. Java is a multi-platform, object-oriented, and networkcentric language that can be used as a platform in itself. It is a fast, secure, reliable

programming language for coding everything from mobile apps and enterprise software to big data applications and server-side technologies.

Java defines the syntax and semantics of the Java programming language. This includes the basic vocabulary and rules used to write algorithms such as primitive data types, if/else blocks, loops, etc.

APIs are important software components bundled with the Java Platform. These are pre-written Java programs that can plug and play existing functionality into your own code. For example, you could use Java APIs to get the date and time, perform mathematical operations, or manipulate text.

Any Java application code written by a developer will typically combine new and pre-existing code from Java APIs and Java libraries.

The Java program was the first language to combine both methods above using a Java Virtual Machine (JVM). The Java code compiler is called the Java Virtual Machine. Any Java file is first compiled into bytecode. Java bytecode can only run in the JVM. The JVM then interprets the bytecode to run it on the underlying hardware platform. So if the application is running on a Windows machine, the JVM will interpret it for Windows. But if it is running on an open-source platform like Linux, the JVM will interpret it for Linux.

## Java Terminology

### 1. Java Virtual Machine(JVM):

This is generally referred to as JVM. There are three execution phases of a program. They are written, compile and run the program. Writing

Department of Computer Application 6Event Planner System

a program is done by a java programmer like you and me. The compilation is done by the JAVAC compiler which is a primary Java compiler included in the Java development kit (JDK). It takes the Java program as input and generates bytecode as output.

In the Running phase of a program, JVM executes the bytecode generated by the compiler.

Now, we understood that the function of Java Virtual Machine is to execute the bytecode produced by the compiler. Every Operating System has a different JVM but the output they produce after the execution of bytecode is the same across all the operating systems. This is why Java is known as a platform-independent language.

### 2. Bytecode in the Development Process:

As discussed, the Javac compiler of JDK compiles the java source code into bytecode so that it can be executed by JVM. It is

saved as .class file by the compiler. To view the bytecode, a disassembler like javap can be used.

### 3. Java Development Kit(JDK):

While we were using the term JDK when we learn about bytecode and JVM. So, as the name suggests, it is a complete Java development

kit that includes everything including compiler, Java Runtime Environment (JRE), java debuggers, java docs, etc. For the program to execute in java, we need to install JDK on our computer in order to create, compile and run the java program.



#### 4. Java Runtime Environment (JRE):

JDK includes JRE. JRE installation on our computers allows the java program to run, however, we cannot compile it. JRE includes

a browser, JVM, applet support, and plugins. For running the java program, a computer needs JRE.

#### 5. Garbage Collector:

In Java, programmers can't delete the objects. To delete or recollect that memory JVM has a program called Garbage Collector. Garbage

Collectors can recollect the objects that are not referenced. So Java makes the life of a programmer easy by handling memory management. However, programmers should be careful about their code whether they are using objects that have been used for a long time. Because Garbage cannot recover the memory of objects being referenced.

#### 6. ClassPath:

The classpath is the file path where the java runtime and Java compiler look for .class files to load. By default, JDK provides many libraries. If you want to

include external libraries they should be added to the classpath.

Department of Computer Application 7Event Planner System

#### Primary/Main Features of Java

**1. Platform Independent:** Compiler converts source code to bytecode and then the JVM executes the bytecode generated by the compiler. This bytecode can run on any platform be it Windows, Linux, or macOS which means if we compile a program on Windows, then we can run it on Linux and vice versa. Each operating system has a different JVM, but the output produced by all the OS is the same after the execution of the bytecode. That is why we call java a platform-independent language.

**2. Object-Oriented Programming Language:** Organizing the program in the terms of a collection of objects is a way of object-oriented programming, each of which represents an instance of the class.

The four main concepts of Object-Oriented programming are:

- Abstraction
- Encapsulation
- Inheritance
- Polymorphism

**3. Simple:** Java is one of the simple languages as it does not have complex features like pointers, operator overloading, multiple inheritances, and Explicit memory allocation.

**4. Robust:** Java language is robust which means reliable. It is developed in such a way that it puts a lot of effort into checking errors as early as possible, that is why the java compiler is able to detect even those errors that are not easy to detect by another programming language. The main features of java that make it robust are garbage collection, Exception Handling, and memory allocation.

**5. Secure:** In java, we don't have pointers, so we cannot access out-of-bound arrays i.e it shows `ArrayIndexOutOfBoundsException` Exception if we try to do so. That's why several security flaws like stack corruption or buffer overflow are impossible to exploit in Java. Also, java programs run in an environment that is independent of the os(operating system) environment which makes java programs more secure.

**6. Distributed:** We can create distributed applications using the java programming language. Remote Method Invocation and Enterprise Java Beans are used for creating Department of Computer Application 8 distributed applications in java. The java programs can be easily distributed on one or more systems that are connected to each other through an internet connection.

**7. Multithreading:** Java supports multithreading. It is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilization of the CPU.

**8. Portable:** As we know, java code written on one machine can be run on another machine. The platform-independent feature of java in which its platform-independent bytecode can be taken to any platform for execution makes java portable.

**9. High Performance:** Java architecture is defined in such a way that it reduces overhead during the runtime and at some times java uses Just In Time (JIT) compiler where the compiler compiles code on-demand basics where it only compiles those methods that are called making applications to execute faster.

**10. Dynamic flexibility:** Java being completely object-oriented gives us the flexibility to add classes, new methods to existing classes, and even create new classes through sub-classes. Java even supports functions written in other languages such as C, C++ which are referred to as native methods.

**11. Sandbox Execution:** Java programs run in a separate space that allows user to execute their applications without affecting the underlying system with help of a bytecode verifier. Bytecode verifier also provides additional security as its role is to check the code for any violation of access.

**12. Write Once Run Anywhere:** As discussed above java application generates a '.class' file that corresponds to our applications(program) but contains code in binary format. It provides ease to architecture-neutral ease as bytecode is not dependent on any machine architecture. It is the primary reason java is used in the enterprising IT industry globally worldwide.

**13. Power of compilation and interpretation:** Most languages are designed with the purpose of either they are compiled language or they are interpreted language. But java integrates arising enormous power as Java compiler compiles the source code to bytecode and JVM executes this bytecode to machine OS-dependent executable code.

**FIREBASE:**

The Firebase Realtime Database is a cloud-hosted NoSQL database that lets you store and sync data between your users in real-time.

Cloud Firestore enables you to store, sync and query app data at global scale.

Department of Computer Application 9Event Planner System

**Key features of Cloud Firestore include:**

- Documents and collections with powerful querying
- iOS, Android, and Web SDKs with offline data access
- Real-time data synchronization
- Automatic, multi-region data replication with strong consistency
- Node, Python, Go, and Java server SDKs

And of course, we have aimed for the simplicity and ease-of-use that is always top priority for Firebase, while still making sure that Cloud Firestore can scale to power even the largest apps.

Managing app data is still hard; you have to scale servers, handle intermittent connectivity, and deliver data with low latency.

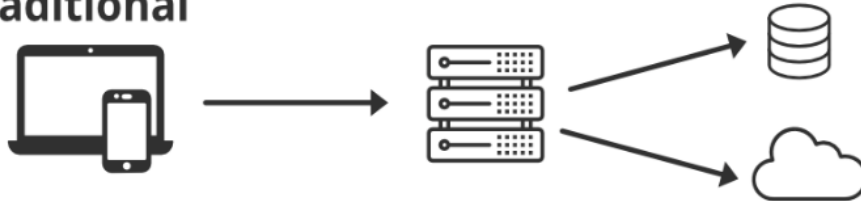
We've optimized Cloud Firestore for app development, so you can focus on delivering value to your users and shipping better apps, faster.

Cloud Firestore's client-side SDKs take care of the complex authentication and networking code you'd normally need to write yourself. Then, on the backend, we provide a powerful set of security rules so you can control access to your data. Security rules let you control which users can access which documents, and let you apply complex validation logic to your data as well. Combined, these features allow your mobile app to connect directly to your database.

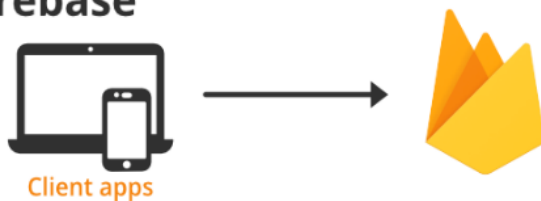
Firebase is a product of Google which helps developers to build, manage, and grow their apps easily. It helps developers to build their apps faster and in a more secure way. No programming is required on the firebase side which makes it easy to use its features

more efficiently. It provides services to android, ios, web, and unity. It provides cloud storage. It uses NoSQL for the database for the storage of data.

### Traditional



### Firebase

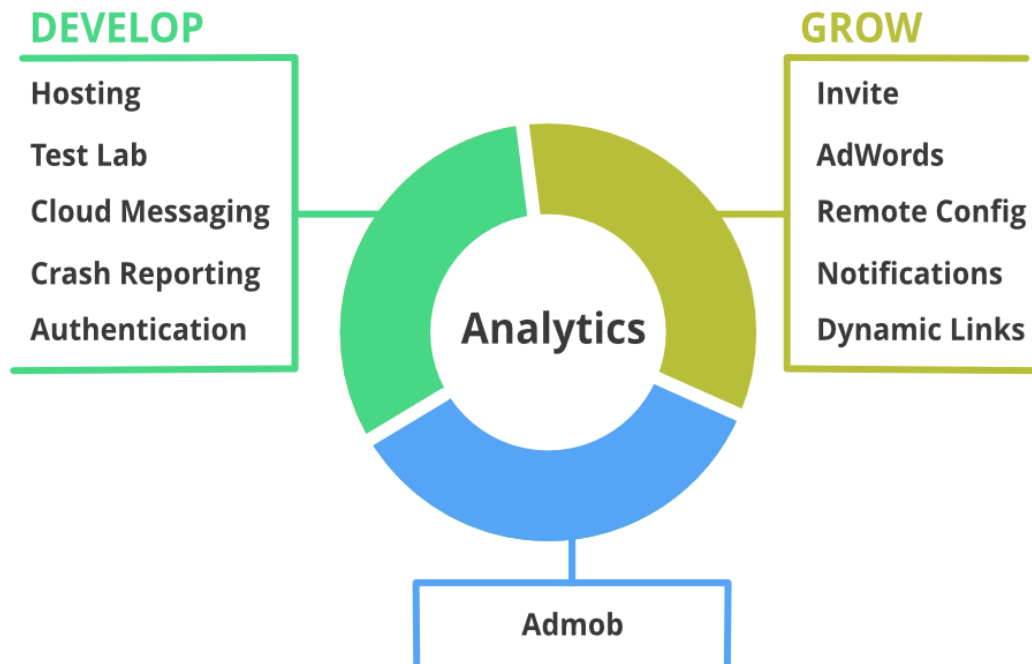


### Brief History of Firebase:

Firebase initially was an online chat service provider to various websites through API and ran with the name Envolv. It got popular as developers used it to exchange application data like a game state in real time across their users more than the chats. This resulted in the separation of the Envolv architecture and its chat system. The Envolv architecture was further evolved by its founders James Tamplin and Andrew Lee, to what modern day Firebase is in the year 2012.

## Features of Firebase:

Mainly there are 3 categories in which firebase provides its services.

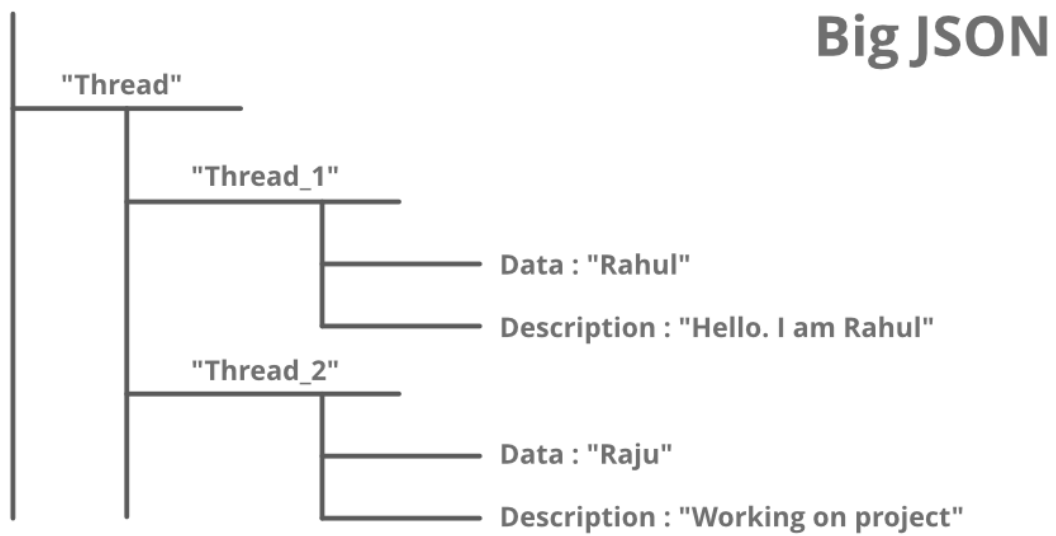


This feature mainly includes backend services that help developers to build and manage their applications in a better way.

Department of Computer Application 11

Services included under this feature are :

- **Realtime Database:** The Firebase Realtime Database is a cloud-based NoSQL database that manages your data at the blazing speed of milliseconds. In simplest term, it can be considered as a big JSON file.



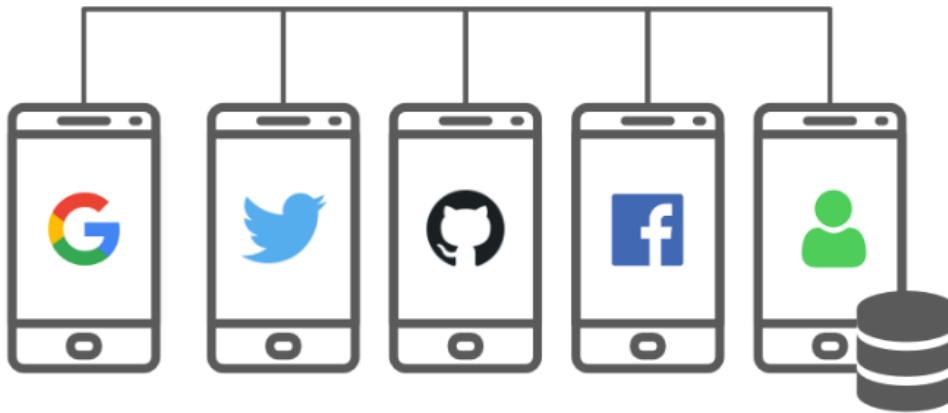
- **Cloud Firestore:** The cloud Firestore is a NoSQL document database that provides services like store, sync, and query through the application on a global scale. It stores data in the form of objects also known as Documents. It has a key-value pair and can store all kinds of data like, strings, binary data, and even JSON trees.



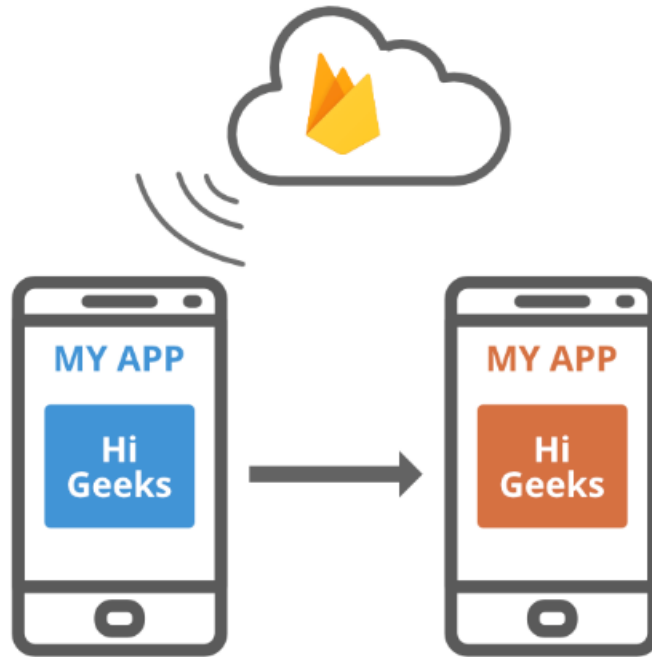
- **Authentication:** Firebase Authentication service provides easy to use UI libraries and SDKs to authenticate users to your app. It reduces the manpower and effort required to develop and maintain the user authentication service. It even handles tasks like merging accounts, which if done manually can be hectic.



# Firestore



- **Remote Config:** The remote configuration service helps in publishing updates to the user immediately. The changes can range from changing components of the UI to changing the behavior of the applications. These are often used while publishing seasonal offers and contents to the application that has a limited life.

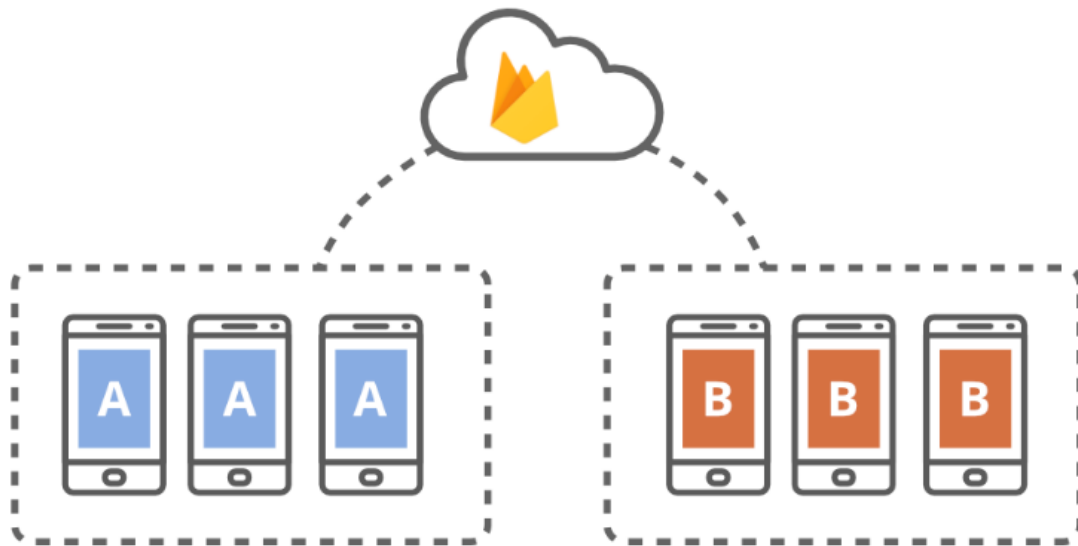


- **Hosting:** Firebase provides hosting of applications with speed and security.

It can be used to host Static or Dynamic websites and microservices. It has the capability of hosting an application with a single command.

Department of Computer Application 13Event Planner System

- **Firestore Database(Firestore):** The Firestore service provides a connection between the server and the application end users, which can be used to receive and send messages and notifications.



### Improve app quality:

Here majorly all the application performance and testing features are provided.

All the features required to check and manage before launching your application officially are provided in this section. Services included are:

- **Crashlytics:** It is used to get real-time crash reports. These reports can further be used to improve the quality of the application. The most interesting part of this service is that it gives a detailed description of the crash which is easier to analyze for the developers.
- **Performance monitoring:** This service gives an insight to the performance characteristics of the applications. The performance monitoring SDK can be used to receive performance data from the application, review them, and make changes to the application accordingly through the Firebase console.

- **Test lab:** This service helps to test your applications on real and virtual devices provided by Google which are hosted on the Google Data Centers. It is a cloud-based app-testing infrastructure which supports testing the application on a wide variety of devices and device configurations

- **App Distribution:** This service is used to pre-release applications that can be tested by trusted testers. It comes in handy as decreases the time required to receive feedback from the testers.

Department of Computer Application

Grow your app:

This feature provides your application analytics and features that can help you to interact with your user and make predictions that help you to grow your app. Services provided are:

- **Google analytics:** It is a Free app measurement service provided by Google that provides insight on app usage and user engagement. It serves unlimited reporting for up to 500 distinct automatic or user-defined events using the Firebase SDK.

- **Predictions:** Firebase Predictions uses machine learning to the application's analytics data, further creating dynamic user segments that are based on your user's behavior. These are automatically available to use for the application through Firebase Remote Config, the Notifications composer, Firebase In-App Messaging, and A/B Testing.

- **Dynamic Links:** Deep Links are links that directly redirects user to specific content. Firebase provides a Dynamic linking service that converts deep links into dynamic links which can directly take the user to a specified

content inside the application. Dynamic links are used for converting web users to Native app users. It also increases the conversion of user-to-user sharing. In addition, it can also be used to integrate social media networks, emails, and SMS to increase user engagement inside the application.

- **A/B Testing:** It is used to optimize the application's experience by making it run smoothly, scaling the product, and performing marketing.

### **ANDROID STUDIO:**

Android Studio is the official Integrated Development Environment (IDE) for Android app development. Based on the powerful code editor and developer tools from IntelliJ IDEA, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

- A flexible Gradle-based build system
- A fast and feature-rich emulator
- A unified environment where you can develop for all Android devices
- Apply Changes to push code and resource changes to your running app without restarting your app

Department of Computer Application 15Event Planner System

- Code templates and GitHub integration to help you build common app features and import sample code
- Extensive testing tools and frameworks
- Lint tools to catch performance, usability, version compatibility, and other problems
- C++ and NDK support
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine

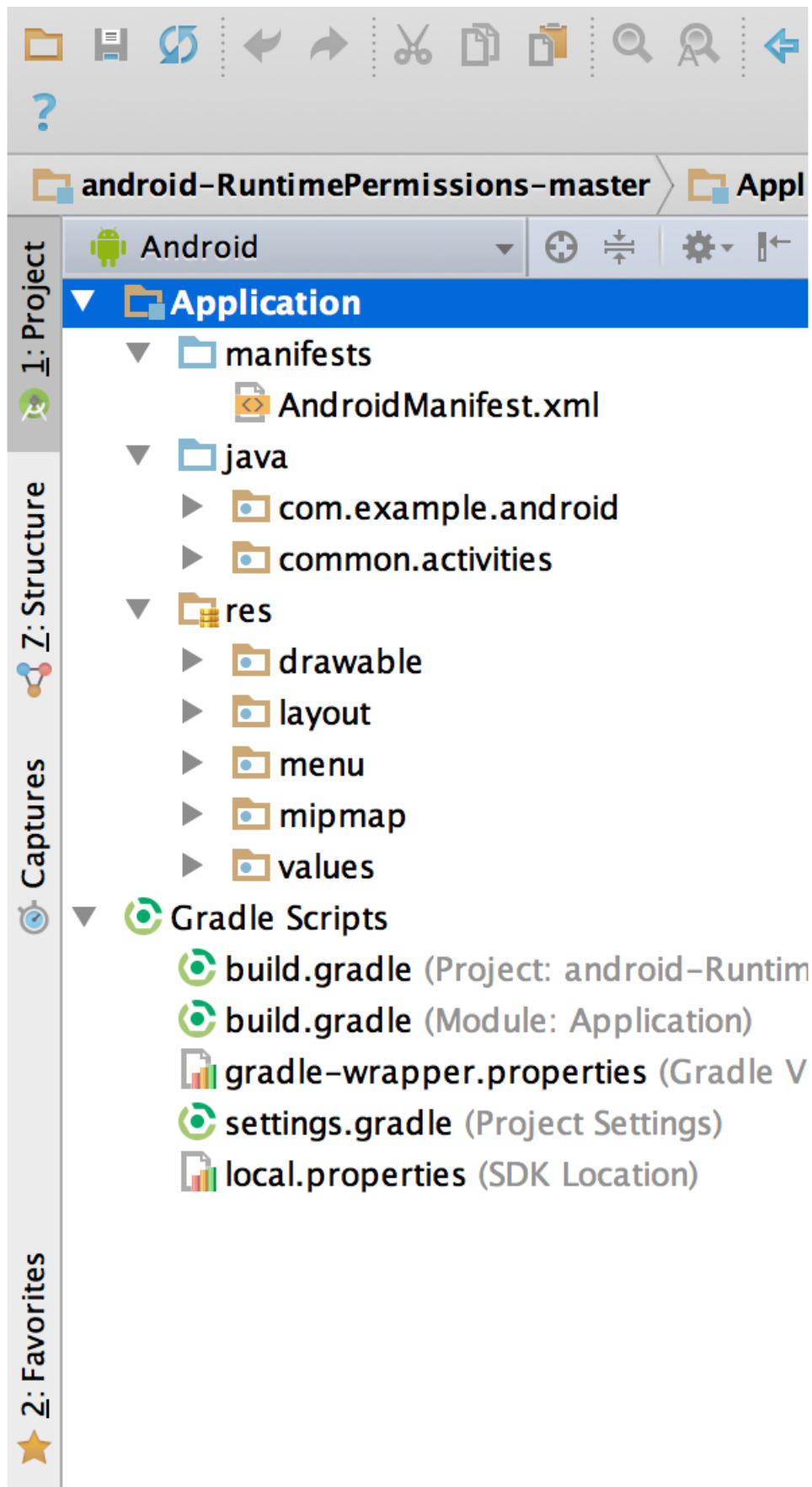
Google Cloud Messaging and App Engine

Each project in Android Studio contains one or more modules with source code

files and resource files. The types of modules include:

- Android app modules
- Library modules
- Google App Engine modules

By default, Android Studio displays your project files in the Android project view, as shown in figure 1. This view is organized by modules to provide quick access to your project's key source files.



All the build files are

visible at the top level, under Gradle Scripts.

#### Department of Computer Application

Figure 1. The project files in Android project view.

Each app module contains the following folders:

- **manifests:** Contains the AndroidManifest.xml file.
- **java:** Contains the Java and Kotlin source code files, including JUnit test code.
- **res:** Contains all non-code resources such as XML layouts, UI strings, and bitmap images.

The Android project structure on disk differs from this flattened representation.

To see the actual file structure of the project, select Project instead of Android from the Project menu.

You can also customize the view of the project files to focus on specific aspects of your app development. For example, select the Problems view of your project to display links to the source files containing any recognized coding and syntax errors, such as a missing XML element closing tag in a layout file.



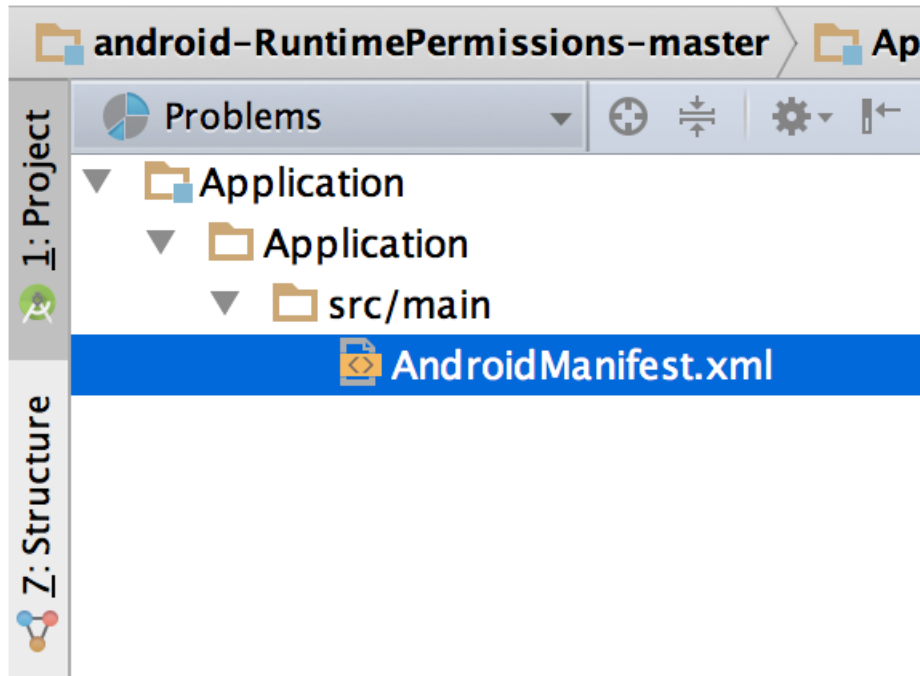


Figure 2. A layout file with a problem in Problems view.

Android Studio uses Gradle as the foundation of the build system, with more Android-specific capabilities provided by the Android Gradle plugin. This build system runs as an integrated tool from the Android Studio menu and independently from the command line. You can use the features of the build system to do the following:

- Customize, configure, and extend the build process.

Department of Computer Application 17

Department of Computer Application 18

- Create multiple APKs for your app with different features, using the same project and modules.
- Reuse code and resources across source sets.

By employing the flexibility of Gradle, you can achieve all of this without modifying your app's core source files.

Android Studio build files are named build.gradle. They are plain text files that use Groovy syntax to configure the build with elements provided by the Android Gradle

plugin. Each project has one top-level build file for the entire project and separate module-level build files for each module. When you import an existing project, Android Studio automatically generates the necessary build files.

Android Studio helps you debug and improve the performance of your code, including inline debugging and performance analysis tools.

## **What is NetBeans?**

NetBeans IDE is a free and open source integrated development environment for application development on Windows, Mac, Linux, and Solaris operating systems. The IDE simplifies the development of web, enterprise, desktop, and mobile applications that use the Java and HTML5 platforms.

NetBeans is an integrated development environment (IDE) specifically designed for Java programming. It provides a range of tools and features to assist Java developers in creating, debugging, and deploying Java applications.

Key features of NetBeans for Java development include:

**Code Editor:** NetBeans offers a rich code editor with features like syntax highlighting, code completion, and code templates to enhance productivity.

**Project Management:** NetBeans provides project templates and wizards to help you set up and manage Java projects. It supports various project types, such as Java SE, Java EE, and Java ME.

**GUI Builder:** NetBeans includes a graphical user interface (GUI) builder that allows developers to visually design user interfaces for their Java applications. It simplifies the process of creating Swing-based GUIs.

**Debugging:** NetBeans has powerful debugging capabilities, including breakpoints, step-by-step execution, variable inspection, and expression evaluation. It helps identify and resolve issues in your Java code.

**Profiling:** The IDE also includes profiling tools to analyze the performance of your Java applications. It helps identify bottlenecks and optimize your code for better execution.

**Version Control:** NetBeans integrates with popular version control systems like Git, SVN, and Mercurial. It allows you to manage your source code and collaborate with other developers effectively.

**Testing:** NetBeans supports unit testing frameworks like JUnit and TestNG. It provides features for creating and running tests, as well as generating test reports.

**Plugins and Extensions:** NetBeans has a vibrant plugin ecosystem that allows developers to extend the IDE's functionality. You can install additional plugins to integrate third-party libraries, frameworks, or tools into your Java development workflow.

NetBeans is a free and open-source IDE, and it is available for Windows, macOS, Linux, and Solaris operating systems. It has been widely used by Java developers for many years and continues to be a popular choice for Java development.

## **Why is NetBeans best for Java?**

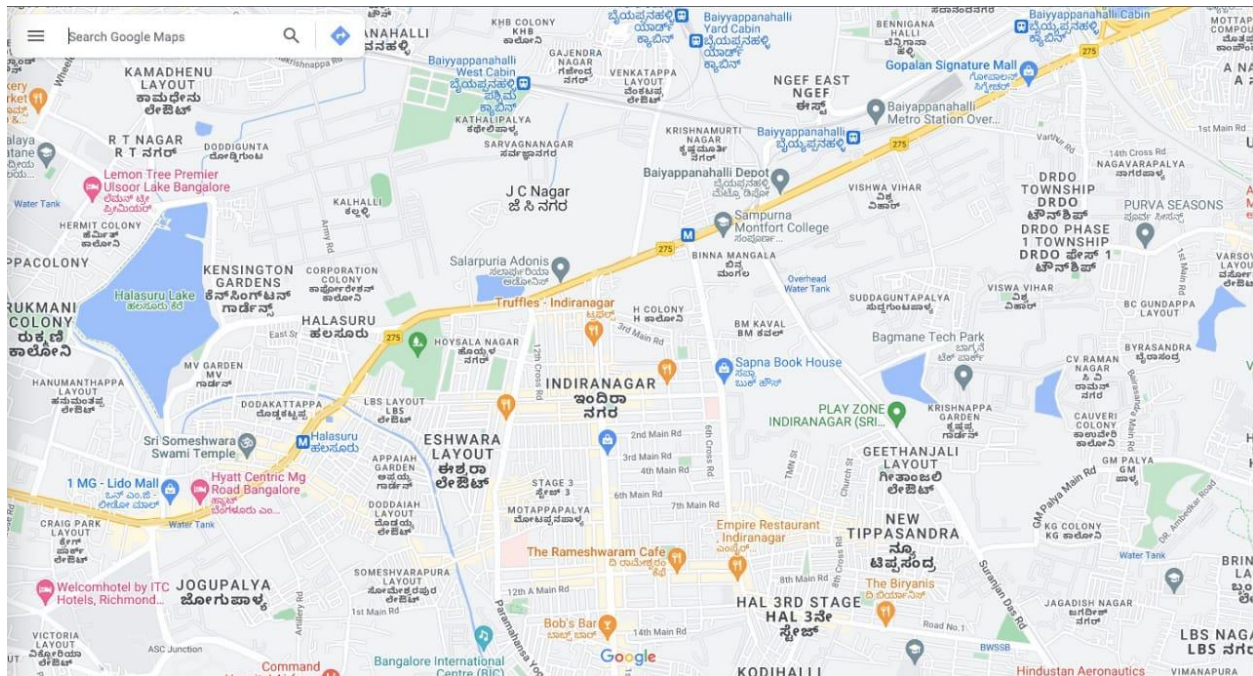
NetBeans IDE helps in creating Java standalone desktop applications, graphical user interface using drag-and-drop swing containers/controls and AWT elements, JavaFX UI, web applications, web services, Enterprise applications in Java.

## **GOOGLE MAPS**

Google Maps is a desktop and mobile web mapping service application and technology provided by google, offering satellite imagery, street maps, and Street View perspectives, as well as functions such as a route planner for traveling by foot, bike, bicycle (beta test), or with public transportation. Also supported are maps embedded on third-party websites via the Google Maps API, [1] and a locator for urban businesses and other organizations in numerous countries around the world. Google Maps satellite images are not updated in real time; however, Google adds data to their Primary Database on a regular basis. Google Earth support states that most of the images are no more than 3 years old.

Google Maps is a free online map from Google. It's accessible through your web browser or as an app for mobile devices. You can use Google Maps to get step-by-step directions, find information about local businesses, and a whole lot more! For example, ever wondered where we create the content for our website.

Google Maps is a widely used web mapping service developed by Google. It provides detailed maps, satellite imagery, street views, and real-time traffic conditions for various locations around the world. Google Maps can be accessed through its website or via the mobile app available on iOS and Android devices.



## Key Features of Google Maps:

**Maps:** Google Maps offers detailed maps for virtually any location worldwide. Users can view maps in different styles, including standard maps, satellite imagery, and terrain maps.

**Directions:** Users can get driving, walking, cycling, or public transit directions between two or more locations. Google Maps provides step-by-step instructions, estimated travel times, and even suggests alternative routes based on current traffic conditions.

**Street View:** With Street View, users can explore panoramic street-level imagery of specific locations. This feature allows you to virtually "walk" through streets and get a realistic view of the area.

**Real-Time Traffic:** Google Maps provides real-time traffic information, including congestion levels, accidents, and road closures. This helps users plan their routes accordingly and find the fastest way to their destination.

**Local Businesses and Points of Interest:** Google Maps includes a vast database of businesses, restaurants, landmarks, and other points of interest. Users can search for specific places or browse different categories to discover nearby options.

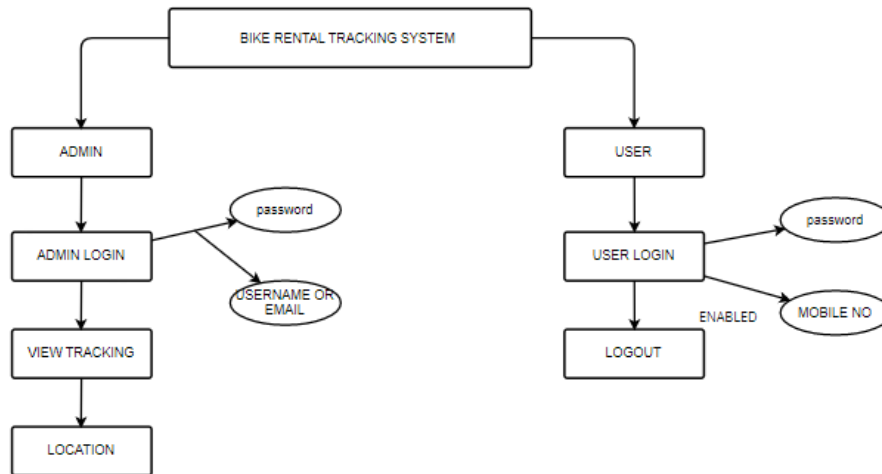
**Reviews and Ratings:** Users can read and write reviews for businesses and places they have visited. This feature helps others make informed decisions and provides valuable feedback to business owners.

**Indoor Maps:** Some buildings, such as airports, shopping malls, and museums, have indoor maps available on Google Maps. These maps help users navigate within these locations and find specific stores, gates, or amenities.

**Integration with Other Google Services:** Google Maps integrates with other Google services, such as Google Search and Google Street View. It also allows users to save favorite locations, create custom maps, and share directions with others.

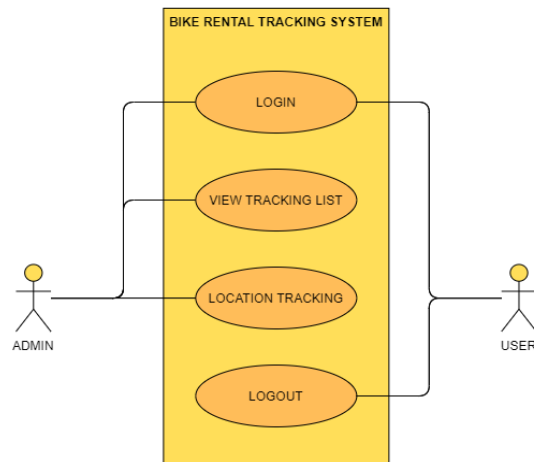
### 3.3 Block diagram

A block diagram is a visual representation of a system that uses simple, labeled blocks that represent single or multiple items, entities or concepts, connected by lines to show relationships between them.

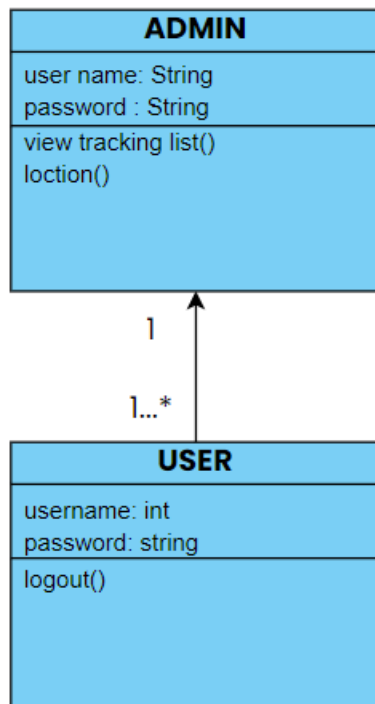


### 3.4 UML DIAGRAMS

#### 3.4.1 USE CASE DIAGRAM

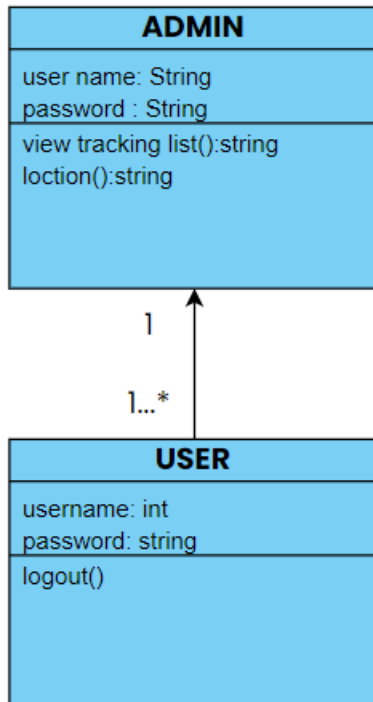


### 3.4.2 CLASS DIAGRAM

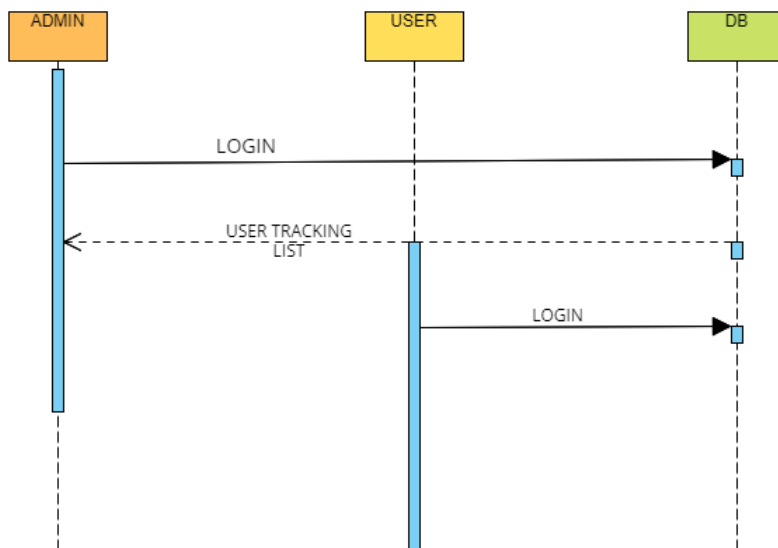




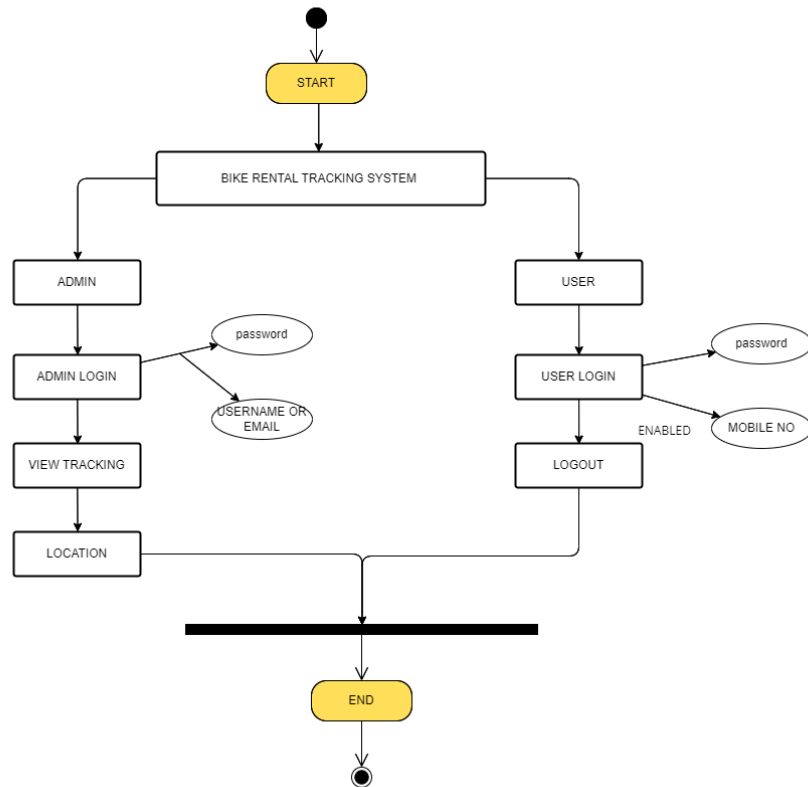
### 3.4.4 OBJECT DIAGRAM



### 3.4.4 SEQUENCE DIAGRAM



### 3.4.5 ACTIVITY DIAGRAM



### 3.4 DATABASE DESIGNS



**Warning:** Your security rules are defined as public, so anyone can steal, modify or delete data in your database

```

https://bike-track-380306-default-rtdb.firebaseio.com/
└─ Users
  └─ user1
    └─ Lat: 12.82891
    └─ Lng: 77.5881458
  └─ user2
    └─ Lat: 12.964678
    └─ Lng: 77.5323953
  
```

### 3.5 Sample Code (Only Code Snippets to be included)

#### Admin log activity

```
package com.bike.track;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.database.FirebaseDatabase;

import java.util.HashMap;

public class AdminLogActivity extends AppCompatActivity {
    EditText editEmail, editPass;
    Button btnLogin;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_adminlog);
    }
}
```

```

editEmail = findViewById(R.id.editEmail);
editPass = findViewById(R.id.editPass);
btnLogin = findViewById(R.id.btnLogin);

btnLogin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if (editEmail.getText().toString().equals("admin@gmail.com")
            && editPass.getText().toString().equals("admin")) {
            Intent i = new Intent(AdminLogActivity.this, UsersList.class);
            startActivity(i);
        }
        else {
            Toast.makeText(getApplicationContext(), "invalid Data", Toast.LENGTH_LONG).show();
        }
    }
});
}
}

```

Main activity

```

package com.bike.track;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

```

```
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    Button btnUser, btnAdmin;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        btnUser = findViewById(R.id.btnUser);
        btnAdmin = findViewById(R.id.btnAdmin);

        btnUser.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(MainActivity.this, UserLogActivity.class);
                startActivity(i);
            }
        });

        btnAdmin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(MainActivity.this, AdminLogActivity.class);
                startActivity(i);
            }
        });
    }
}
```

```
}
}
```

Map activity

```
package com.bike.track;
```

```
import androidx.appcompat.app.AppCompatActivity;
```

```
import android.os.Bundle;
```

```
import android.webkit.WebView;
```

```
public class MapActivity extends AppCompatActivity {
```

```
@Override
```

```
protected void onCreate(Bundle savedInstanceState) {
```

```
    super.onCreate(savedInstanceState);
```

```
    setContentView(R.layout.activity_map);
```

```
    WebView webView = (WebView) findViewById(R.id.webView);
```

```
    webView.getSettings().setJavaScriptEnabled(true);
```

```
    webView.loadUrl(
```

```
        "http://maps.google.com/maps?q=loc:" + 12.9581667 + "," + 77.7146389 + " (" + "Label which  
you want" + ")");
```

```
}
```

```
}
```

User



```
packagecom.bike.track;

importandroid.app.Activity;
importandroid.app.ActivityManager;
importandroid.app.Service;
importandroid.content.Context;
importandroid.content.Intent;
importandroid.content.pm.PackageManager;
importandroid.location.LocationManager;
importandroid.os.Bundle;
importandroid.util.Log;
importandroid.view.View;
importandroid.widget.Button;
importandroid.widget.Toast;

importandroidx.appcompat.app.AppCompatActivity;
importandroidx.core.app.ActivityCompat;
importandroidx.core.content.ContextCompat;

importcom.bike.track.gps.GPSTracker;

public class User extends AppCompatActivity {

    private static final String TAG = "HomeScreen";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_user);
    }
}
```

```

try {

    if ((ActivityCompat.checkSelfPermission(this,
        android.Manifest.permission.ACCESS_FINE_LOCATION)
        != PackageManager.PERMISSION_GRANTED)

        && (ActivityCompat.checkSelfPermission(this,
        android.Manifest.permission.ACCESS_COARSE_LOCATION)
        != PackageManager.PERMISSION_GRANTED)) {

        ActivityCompat.requestPermissions(this, new
        String[] {android.Manifest.permission.ACCESS_FINE_LOCATION,
        android.Manifest.permission.ACCESS_COARSE_LOCATION}, 101);
    }
} catch (Exception e){
    e.printStackTrace();
}

getLocation();

Button logout = findViewById(R.id.btn_logout);

logout.setOnClickListener(view -> {
    stopService(new Intent(this, GPSTracker.class));
    finish();
});
}

@Override
protected void onDestroy() {
    super.onDestroy();

```

```
Log.d(TAG, "onDestroy: called");
}
```

```
public void getLocation(){
Log.d(TAG, "getLocation: called");
```

```
if(!isMyServiceRunning(GPSTracker.class)) {
startService(new Intent(this, GPSTracker.class));
}else{
Log.d(TAG, "getLocation: service already running");
}
}
```

```
private boolean isMyServiceRunning(Class<?> serviceClass) {
    ActivityManager manager = (ActivityManager)
        getSystemService(Context.ACTIVITY_SERVICE);
    for (ActivityManager.RunningServiceInfo service :
        manager.getRunningServices(Integer.MAX_VALUE)) {
        if (serviceClass.getName().equals(service.service.getClassName())) {
            Log.d(TAG, "isMyServiceRunning: serviceRunning"+service.service.getClassName());
            return true;
        }
    }
    return false;
}
```

```
User data
package com.bike.track;
```

```
public class UserData {  
    private static UserData INSTANCE;  
    private String user = null;  
  
    privateUserData() {  
    }  
  
    public static UserData getInstance() {  
        if(INSTANCE == null) {  
            INSTANCE = new UserData();  
        }  
  
        return INSTANCE;  
    }  
  
    public String getUser() {  
        return user;  
    }  
  
    public void setUser(String user) {  
        this.user = user;  
    }  
}  
  
User log activity  
package com.bike.track;  
import android.content.Intent;  
import android.os.Bundle;
```

```
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class UserLogActivity extends AppCompatActivity {
    EditText editMob, editPass;
    Button btnLogin;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_userlog);
        editMob = findViewById(R.id.editMob);
        editPass = findViewById(R.id.editPass);
        btnLogin = findViewById(R.id.btnLogin);

        btnLogin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String mob1 = "1234567890";
                String pass1 = "akshay123";
                String mob2 = "0987654321";
```

```
String pass2 = "sonu123";
```

```

if(TextUtils.isEmpty(editMob.getText().toString())) {
    editMob.setError("Mobile number cannot be Empty");
    return;
}
if(TextUtils.isEmpty(editPass.getText().toString())) {
    editPass.setError("Password cannot be Empty");
    return;
}
if (editMob.getText().toString().equals(mob1) &&editPass.getText().toString().equals(pass1))
{
    UserData.getInstance().setUser("user1");
    Intent i = new Intent(UserLogActivity.this, User.class);
    startActivity(i);
}
if (editMob.getText().toString().equals(mob2) &&editPass.getText().toString().equals(pass2))
{
    UserData.getInstance().setUser("user2");
    Intent i = new Intent(UserLogActivity.this, User.class);
    startActivity(i);
}
}
});
}
}

```

User list

```
package com.bike.track;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.util.Log;
import android.widget.Button;
import android.widget.TextView;

import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

public class UsersList extends AppCompatActivity {

    private static final String TAG = "UsersList";
    private TextView tvLoc1, tvLoc2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_userslist);

        Button userView1 = findViewById(R.id.btn_view1);
```

```
Button userView2 = findViewById(R.id.btn_view2);
```

```
tvLoc1 = findViewById(R.id.tv_loc1);
```

```
tvLoc2 = findViewById(R.id.tv_loc2);
```

```
Location user1Location = fetchLocationData("user1");
```

```
Location user2Location = fetchLocationData("user2");
```

```
userView1.setOnClickListener(view -> {
    openMap(user1Location.lat, user1Location.lng, "user1");
});
```

```
userView2.setOnClickListener(view -> {
    openMap(user2Location.lat, user2Location.lng, "user2");
});
}
```

```
private Location fetchLocationData(String user) {
```

```
    Location location = new Location();
```

```
    DatabaseReferencedbRef
```

```
=
```

```
    FirebaseDatabase.getInstance().getReference().child("Users").child(user);
```

```
    dbRef.getRef().addValueEventListener(new ValueEventListener() {
```

```
        @Override
```

```
        public void onDataChange(@NonNull DataSnapshot snapshot) {
```

```
            try {
```

```
                location.lat = (double) snapshot.child("Lat").getValue();
```

```
                location.lng = (double) snapshot.child("Lng").getValue();
```

```
                if(user.equals("user1")) {
```

```
                    tvLoc1.setText("Lat: "+location.lat+", Lng: "+location.lng);
```

```
                }else if(user.equals("user2")) {
```



```

tvLoc2.setText("Lat: "+location.lat+", Lng: "+location.lng);
}
} catch (Exception e) {
e.printStackTrace();
}
}

```

```

@Override
public void onCancelled(@NonNull DatabaseError error) {
// Getting Post failed, log a message
Log.e(TAG, "loadPost:onCancelled", error.toException());
}
});

```

```

return location;
}

```

```

private void openMap(double lat, double lng, String user) {
String strUri = "http://maps.google.com/maps?q=loc:" + lat + "," + lng + " (" + user + ")";
Intent intent = new Intent(android.content.Intent.ACTION_VIEW, Uri.parse(strUri));

```

```

intent.setClassName("com.google.android.apps.maps",
"com.google.android.maps.MapActivity");

```

```

startActivity(intent);
}

```

```

public class Location {
double lat;

```

```
double lng;  
}  
}  
package com.bike.track;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.AdapterView;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.Spinner;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import com.google.firebase.database.FirebaseDatabase;  
  
import java.util.HashMap;  
  
public class AdminLogActivity extends AppCompatActivity {  
    EditText editEmail, editPass;  
    Button btnLogin;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);
```

```

setContentView(R.layout.activity_adminlog);
editEmail = findViewById(R.id.editEmail);
editPass = findViewById(R.id.editPass);
btnLogin = findViewById(R.id.btnLogin);

btnLogin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if (editEmail.getText().toString().equals("admin@gmail.com")
            && editPass.getText().toString().equals("admin")) {
            Intent i = new Intent(AdminLogActivity.this, UsersList.class);
            startActivity(i);
        }
        else {
            Toast.makeText(getApplicationContext(), "invalid Data", Toast.LENGTH_LONG).show();
        }
    }
});
}
}

```

Main activity

```

package com.bike.track;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;

```

```

import android.widget.Button;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    Button btnUser, btnAdmin;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        btnUser = findViewById(R.id.btnUser);
        btnAdmin = findViewById(R.id.btnAdmin);

        btnUser.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(MainActivity.this, UserLogActivity.class);
                startActivity(i);
            }
        });

        btnAdmin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(MainActivity.this, AdminLogActivity.class);
                startActivity(i);
            }
        });
    }
}

```

```
});
}
}
```

Map activity

```
package com.bike.track;
```

```
import androidx.appcompat.app.AppCompatActivity;
```

```
import android.os.Bundle;
```

```
import android.webkit.WebView;
```

```
public class MapActivity extends AppCompatActivity {
```

```
    @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.activity_map);
```

```
        WebView webView = (WebView) findViewById(R.id.webView);
```

```
        webView.getSettings().setJavaScriptEnabled(true);
```

```
        webView.loadUrl(
```

```
            "http://maps.google.com/maps?q=loc:" + 12.9581667 + "," + 77.7146389 + " (" + "Label which  
you want" + ")");
```

```
        }
```

```
    }
```

User

```
packagecom.bike.track;

importandroid.app.Activity;
importandroid.app.ActivityManager;
importandroid.app.Service;
importandroid.content.Context;
importandroid.content.Intent;
importandroid.content.pm.PackageManager;
importandroid.location.LocationManager;
importandroid.os.Bundle;
importandroid.util.Log;
importandroid.view.View;
importandroid.widget.Button;
importandroid.widget.Toast;

importandroidx.appcompat.app.AppCompatActivity;
importandroidx.core.app.ActivityCompat;
importandroidx.core.content.ContextCompat;

importcom.bike.track.gps.GPSTracker;

public class User extends AppCompatActivity {

    private static final String TAG = "HomeScreen";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```

setContentView(R.layout.activity_user);

try {
    if ((ActivityCompat.checkSelfPermission(this,
        android.Manifest.permission.ACCESS_FINE_LOCATION)
        != PackageManager.PERMISSION_GRANTED)

        && (ActivityCompat.checkSelfPermission(this,
        android.Manifest.permission.ACCESS_COARSE_LOCATION)
        != PackageManager.PERMISSION_GRANTED)) {

        ActivityCompat.requestPermissions(this, new
        String[] {android.Manifest.permission.ACCESS_FINE_LOCATION,
        android.Manifest.permission.ACCESS_COARSE_LOCATION}, 101);
    }
} catch (Exception e){
    e.printStackTrace();
}

getLocation();

Button logout = findViewById(R.id.btn_logout);

logout.setOnClickListener(view -> {
    stopService(new Intent(this, GPSTracker.class));
    finish();
});

@Override
protected void onDestroy() {

```

```

super.onDestroy();
Log.d(TAG, "onDestroy: called");
}

public void getLocation(){
Log.d(TAG, "getLocation: called");

if(!isMyServiceRunning(GPSTracker.class)) {
startService(new Intent(this, GPSTracker.class));
}else{
Log.d(TAG, "getLocation: service already running");
}
}

private boolean isMyServiceRunning(Class<?> serviceClass) {
    ActivityManager manager = (ActivityManager)
    getSystemService(Context.ACTIVITY_SERVICE);
    for (ActivityManager.RunningServiceInfo service :
    manager.getRunningServices(Integer.MAX_VALUE)) {
        if (serviceClass.getName().equals(service.service.getClassName())) {
            Log.d(TAG, "isMyServiceRunning: serviceRunning"+service.service.getClassName());
            return true;
        }
    }
    return false;
}
}

```

User data



```
packagecom.bike.track;

public class UserData {

    private static UserData INSTANCE;

    private String user = null;

    privateUserData() {

    }

    public static UserDatagetInstance() {
        if(INSTANCE == null) {
            INSTANCE = new UserData();
        }

        return INSTANCE;
    }

    public String getUser() {
        return user;
    }

    public void setUser(String user) {
        this.user = user;
    }
}

User log activity
packagecom.bike.track;
importandroid.content.Intent;
```

```

import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class UserLogActivity extends AppCompatActivity {
    EditText editMob, editPass;
    Button btnLogin;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_userlog);
        editMob = findViewById(R.id.editMob);
        editPass = findViewById(R.id.editPass);
        btnLogin = findViewById(R.id.btnLogin);

        btnLogin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String mob1 = "1234567890";
                String pass1 = "akshay123";
            }
        });
    }
}

```

```
String mob2 = "0987654321";
String pass2 = "sonu123";

if(TextUtils.isEmpty(editMob.getText().toString())) {
    editMob.setError("Mobile number cannot be Empty");
    return;
}

if(TextUtils.isEmpty(editPass.getText().toString())) {
    editPass.setError("Password cannot be Empty");
    return;
}

if (editMob.getText().toString().equals(mob1) && editPass.getText().toString().equals(pass1))
{
    UserData.getInstance().setUser("user1");
    Intent i = new Intent(UserLogActivity.this, User.class);
    startActivity(i);
}

if (editMob.getText().toString().equals(mob2) && editPass.getText().toString().equals(pass2))
{
    UserData.getInstance().setUser("user2");
    Intent i = new Intent(UserLogActivity.this, User.class);
    startActivity(i);
}

});
}
```

User list

```
package com.bike.track;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.util.Log;
import android.widget.Button;
import android.widget.TextView;

import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

public class UsersList extends AppCompatActivity {

    private static final String TAG = "UsersList";
    private TextView tvLoc1, tvLoc2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_userslist);
    }
}
```

```
Button userView1 = findViewById(R.id.btn_view1);
```

```
Button userView2 = findViewById(R.id.btn_view2);
```

```
tvLoc1 = findViewById(R.id.tv_loc1);
```

```
tvLoc2 = findViewById(R.id.tv_loc2);
```

```
Location user1Location = fetchLocationData("user1");
```

```
Location user2Location = fetchLocationData("user2");
```

```
userView1.setOnClickListener(view -> {
    openMap(user1Location.lat, user1Location.lng, "user1");
});
```

```
userView2.setOnClickListener(view -> {
    openMap(user2Location.lat, user2Location.lng, "user2");
});
}
```

```
private Location fetchLocationData(String user) {
```

```
    Location location = new Location();
```

```
    DatabaseReferencedbRef
```

```
=
```

```
    FirebaseDatabase.getInstance().getReference().child("Users").child(user);
```

```
    dbRef.getRef().addValueEventListener(new ValueEventListener() {
```

```
        @Override
```

```
        public void onDataChange(@NonNull DataSnapshot snapshot) {
```

```
            try {
```

```
                location.lat = (double) snapshot.child("Lat").getValue();
```

```
                location.lng = (double) snapshot.child("Lng").getValue();
```

```
                if(user.equals("user1")) {
```

```
                    tvLoc1.setText("Lat: "+location.lat+", Lng: "+location.lng);
```

```

} else if(user.equals("user2")) {
tvLoc2.setText("Lat: "+location.lat+", Lng: "+location.lng);
}
} catch (Exception e) {
e.printStackTrace();
}
}

```

```

@Override
public void onCancelled(@NonNull DatabaseError error) {
// Getting Post failed, log a message
Log.e(TAG, "loadPost:onCancelled", error.toException());
}
});

```

```

return location;
}

```

```

private void openMap(double lat, double lng, String user) {
String strUri = "http://maps.google.com/maps?q=loc:" + lat + "," + lng + " (" + user + ")";
Intent intent = new Intent(android.content.Intent.ACTION_VIEW, Uri.parse(strUri));

intent.setClassName("com.google.android.apps.maps",
"com.google.android.maps.MapActivity");

startActivity(intent);
}

```

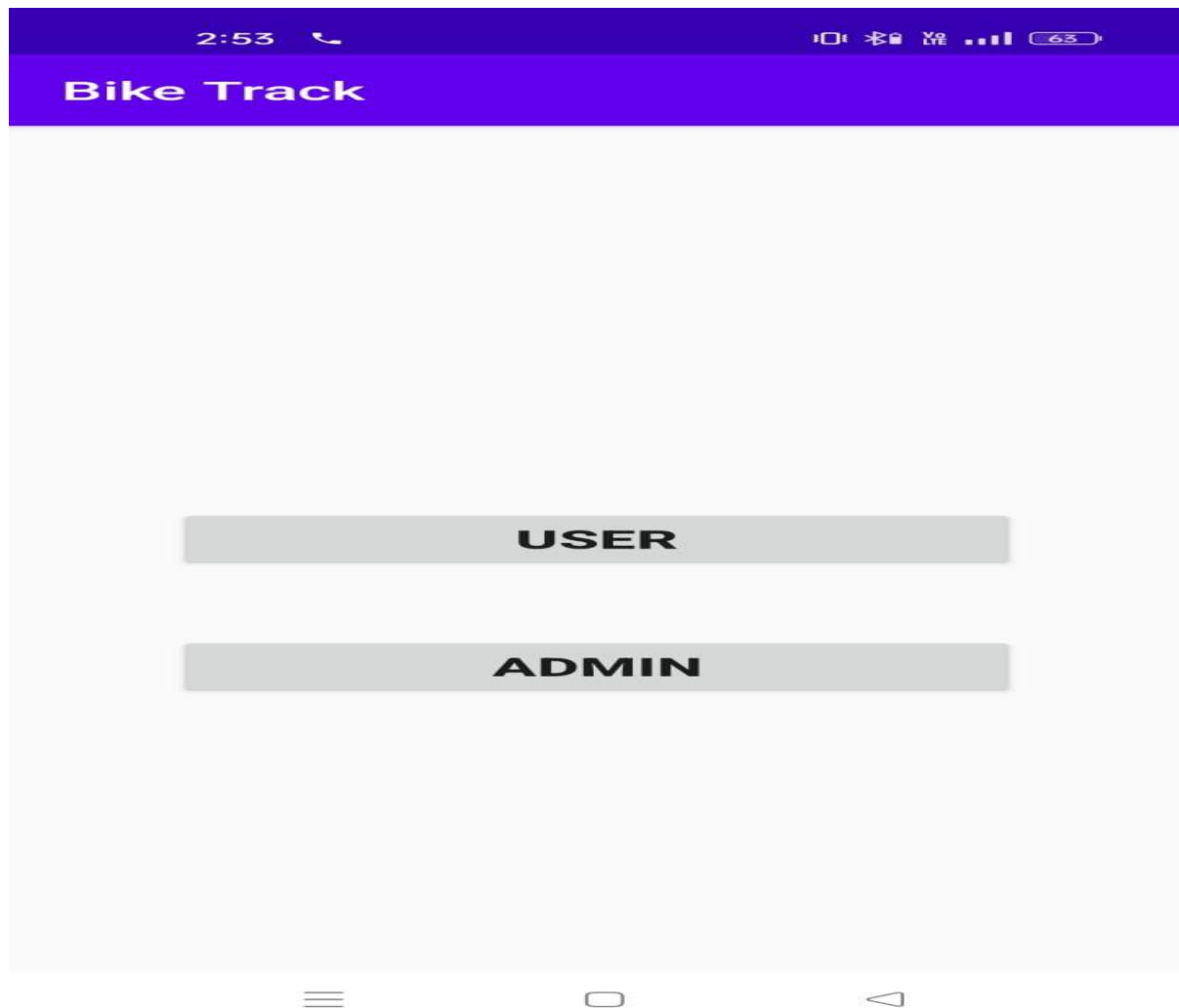
```
public class Location {  
    double lat;  
    double lng;  
}  
}
```

## Chapter 4

### RESULTS AND DISCUSSION

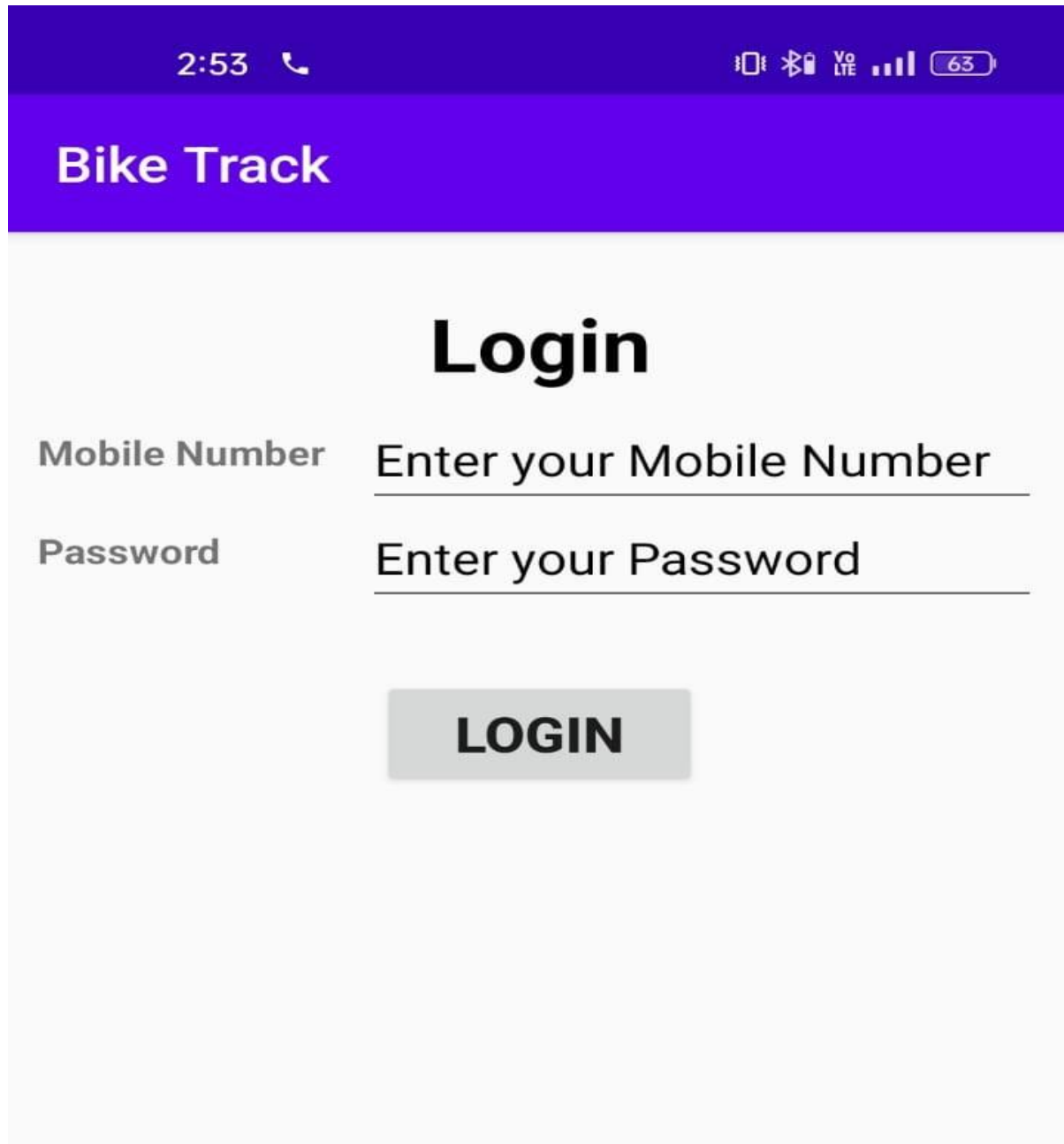
#### 4.1 INPUT AND OUTPUT FORMS

##### HOME PAGE





## LOGIN



The image shows a mobile application interface for 'Bike Track'. At the top, there is a status bar with the time 2:53, a phone icon, and various connectivity icons (Wi-Fi, Bluetooth, VoLTE, signal strength) along with a battery level of 63%. Below the status bar is a purple header with the text 'Bike Track' in white. The main content area is light gray and features the word 'Login' in a large, bold, black font. Below this, there are two input fields. The first is labeled 'Mobile Number' and contains the placeholder text 'Enter your Mobile Number'. The second is labeled 'Password' and contains the placeholder text 'Enter your Password'. At the bottom of the form is a gray button with the text 'LOGIN' in bold, black, uppercase letters.

2:53

Bike Track

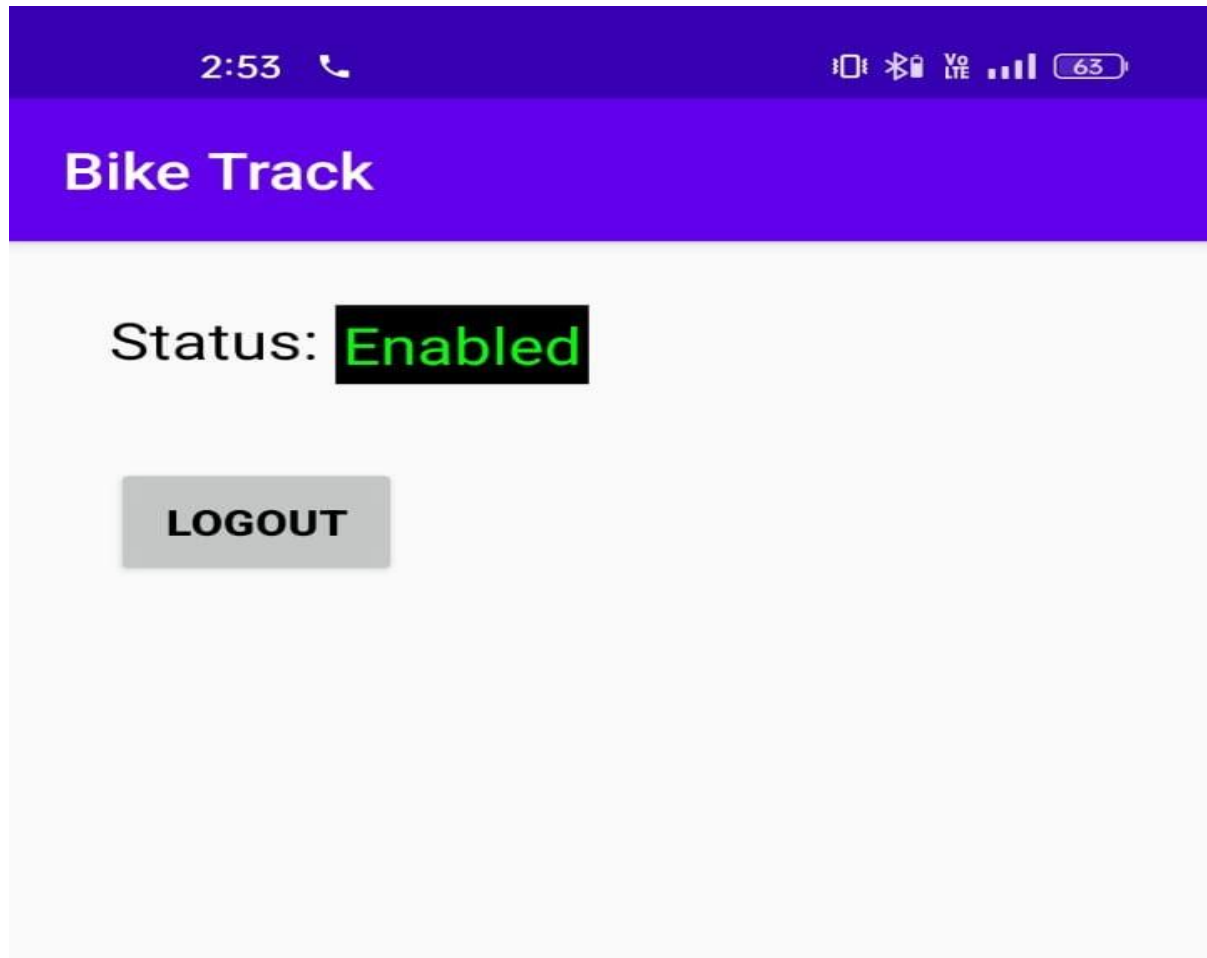
# Login

Mobile Number Enter your Mobile Number

Password Enter your Password






**LOGIN**

## STATUS PAGE



**BIKE TRACKING LIST**

2:54



63

Bike Track

User: Tejaswini

Mob:6361190326

Vehicle No: KA02 9113

TRACK

TextView

User: varshitha

Mob: 9741865928

Vehicle No: KA03 8113

TRACK

TextView

User: Sanjana

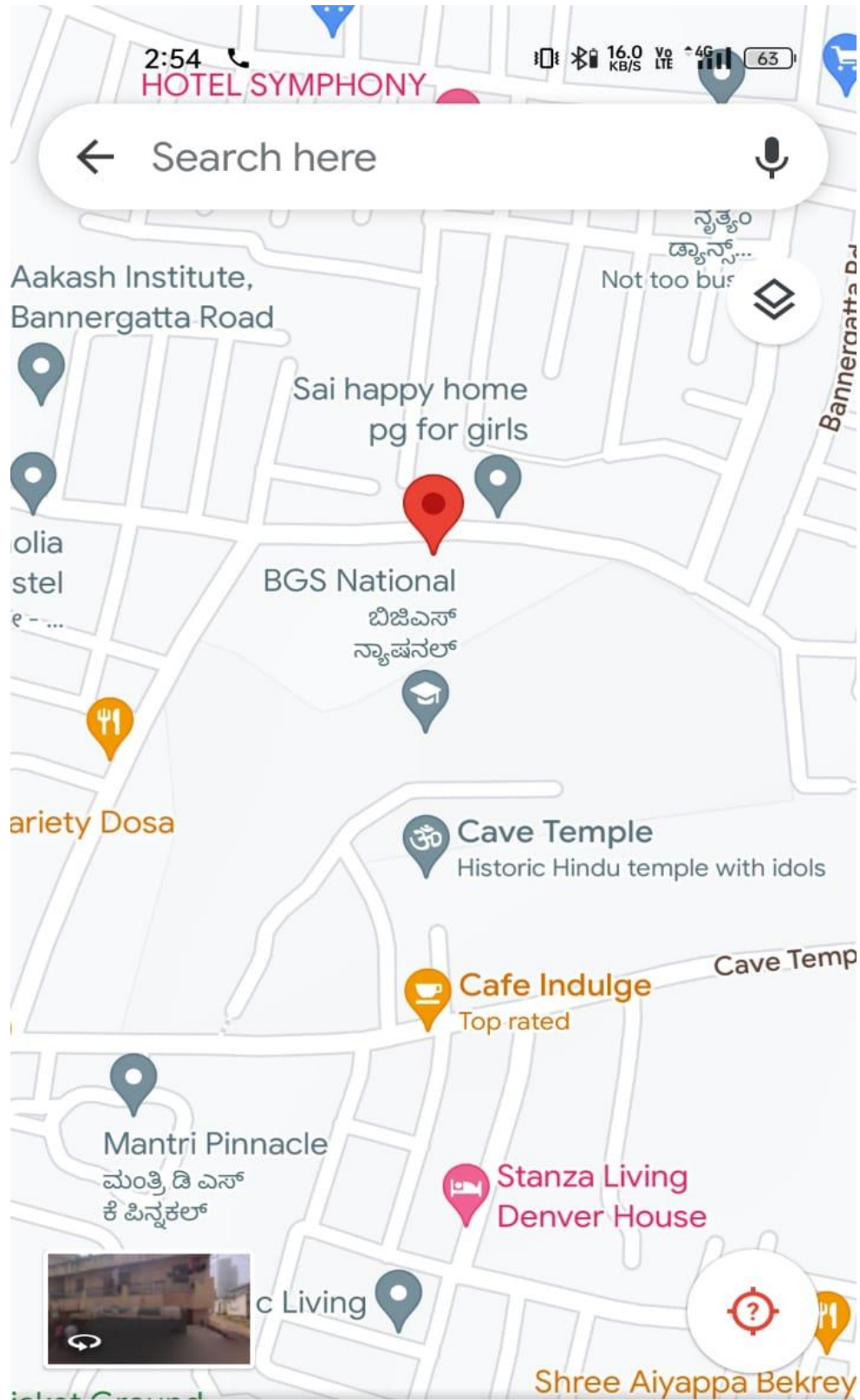
Mob: 9855446633

Vehicle No: KA04 2668

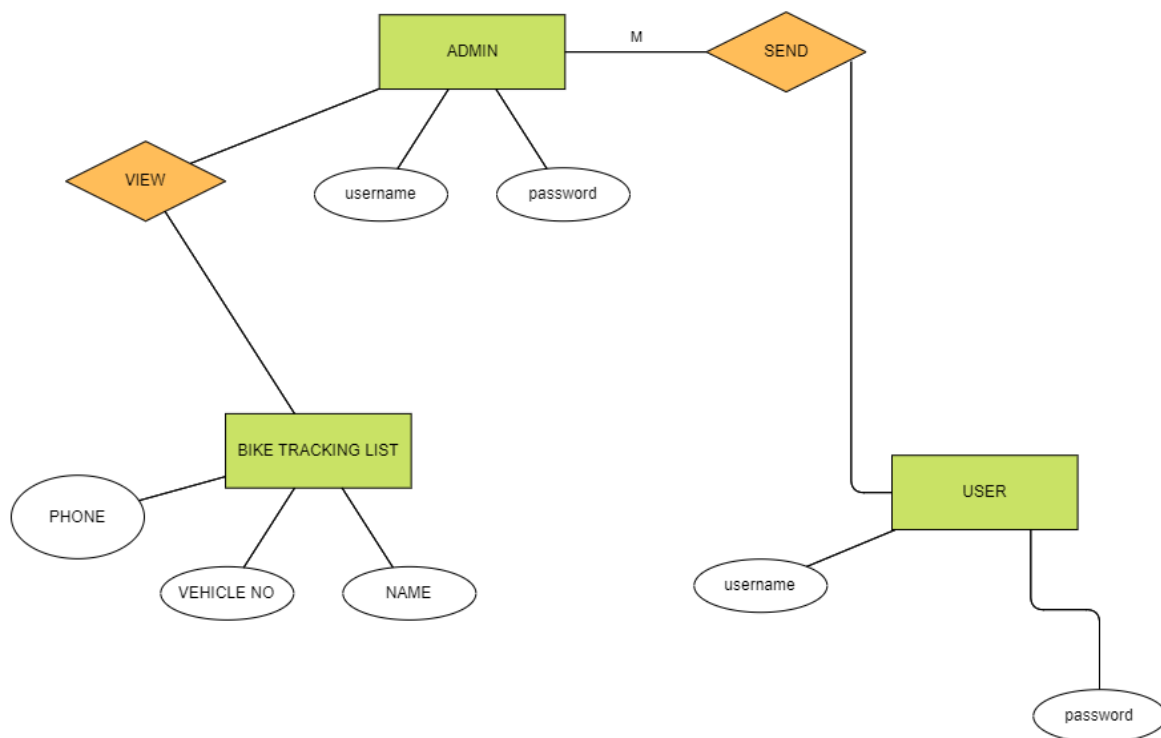
TRACK

TextView

## LOCATION



## 4.2 EXPLANATION ABOUT PROJECT FLOW



## Chapter 5

### CONCLUSION AND FUTURE SCOPE

#### 5.1 Conclusion

To conclude the description about the project, the project, developed using JAVA with MySQL is based on the requirement specification of the user and the analysis of the existing system, with flexibility for future enhancement. This project focuses on how to track bike. Finally we conclude that our project bike rental tracking system is an efficient system to track rented bikes. These systems is tested on different test cases and showed positive result. The Bike rental tracking system portal is developed to facilitate easy processing of tracking bike procedures. Manually, this consumes a lot of time, effort and paper work. And also it is possible to freely submit the feedback without any hesitation. So, this portal overcomes all these limitations and offers a great deal of help at each and every stage in the whole process of availing a leave.

So, bikes are the only solution that can come to the rescue here and help people reach their destination on time. Companies have started launching bike rental mobile apps as it is a great relief to the eco-system from vehicle pollution and the daily honking. Secondly, there will be considerably lesser cars seen on the roads, hence, less traffic.

Vehicle rental business is growing whether it's Car, Bus, Bike, E-scooter, Truck, etc. So, secure your vehicle by using GPS tracking software. Also, it helps the owner of the vehicle rental business to manage the fleet by using fleet management software.

## 5.2Future Scope

- We can use the EEPROM to store the previous Navigating positions up to 256 locations and we can navigate up to N number of locations by increasing its memory.
  - We can reduce the size of the kit by using GPS+GSM on the same module.
  - We can increase the accuracy up to 3m by increasing the cost of the Gps receivers.
  - We can use our kit for detection of bomb by connecting to the bomb detector.
  - With the help of high sensitivity vibration sensors, we can detect the accident. whenever bike unexpectedly had an accident on the road with help of vibration sensor, we can detect the accident and we can send the location to the owner, hospital and police.
  - We can use our kit to assist the traffic. By keeping the kits in the entire bikes and by knowing the locations of all the bikes.
  - If anybody steals our bike, we can easily find our bike around the globe. By keeping bike positioning bike on the bike.
  - The functioning of the motorbike rental apps is quite simple and similar to the taxi booking app. All a user has to do is look for the most suitable app for them on the Play Store or the App Store and install it from there.
- 
- Once the app is saved, they have to make their individual accounts by registering themselves with the application and saving their personal details. This helps the user save information like their location and the payment details. After they are done entering all their details, they can check the bikes available in proximity and select one according to their preferences. All the information entered o the app helps them get an estimate of the amount they would have to spend on a rental bike.
- 
- Further, they can either select it and go ahead with their bike ride or look for other options available in the mobile app.

## REFERENCES

- Parry, A. (2013). Event planners rejoice as a new type of event staff provider launches in London. Retrieved October 20, 2013 from <http://www.eventindustrynews.co.uk/2013/03/20/event-planners-rejoice-as-a-new-type-of-event-staff-provider-launches-in-london/> .
- Freidson, E. (2007). Professionalism the third logic. Cambridge: Polity press in association with Blackwell publishers.
- East Anglia Childrens Hospice. (2013). Overview of EACH. Retrieved October 19, 2013 from [http://www.each.org.uk/about\\_us/overview\\_of\\_each](http://www.each.org.uk/about_us/overview_of_each) .
- Lloyds Banking Group. (2012). The economic impact of the London 2012 Olympics and Paralympics games. Retrieved October 19, 2013 From [http://www.lloydsbankinggroup.com/media/pdfs/lbg/2012/Eco\\_impact\\_report.pdf](http://www.lloydsbankinggroup.com/media/pdfs/lbg/2012/Eco_impact_report.pdf).