A PROJECT REPORT ON

"Automated Toll Payment and Vehicle Tracking System".

Submitted for partial fulfillment of award of the degree.

BACHELOR OF TECHNOLOGY

(Computer Science & Engineering)

BY

Swapnil Nimbalkar (MITU17CS0137)
Rahul Nanwani (MITU17CS0135)
Vijay Gupta (MITU17CS0077)

Under the guidance of

Prof. Kiran Bidua

Department of Computer Science and Engineering

MIT School of Engineering

MIT Art, Design and Technology University, Pune

MAEER's Rajbaug Campus, Loni-Kalbhor, Pune 412201

2020 - 2021

MIT SCHOOL OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

MAEER's Rajbaug Campus, Loni-Kalbhor, Pune - 412201

CERTIFICATE

This is to certify that the project report entitled.

"Automated Toll Payment and Vehicle Tracking System."

Submitted by

Swapnil Nimbalkar (MITU17CS0137)
Rahul Nanwani (MITU17CS0135)
Vijay Gupta (MITU17CS0077)

is a bonafide work carried out by students under the supervision of Prof. Kiran Bidua and it is submitted towards the fulfillment of the requirement of MIT-ADT University, Pune for the award of the degree of Bachelor of Technology (Computer Science & Engineering)

Prof. Kiran Bidua
Guide

Prof. Dhanelakshmi P.Y.
Co-Guide

Prof. Dr. Reena Pagare
Project Coordinator

Prof. Dr. Rajneeshkaur Sachdeo
Professor & Head of the Department

Prof. Dr. Kishore Ravande
Principal

Seal/Stamp of the College
Place: Pune
Date: 07/07/2021

DECLARATION

We, the team members

Swapnil Nimbalkar	MITU17BTCS0137
Rahul Nanwani	MITU17BTCS0135
Vijay Gupta	MITU17BTCS0077

Hereby declare that the project work incorporated in the present project entitled "Automated Toll Payment and Vehicle Tracking System" is original work. This work (in part or in full) has not been submitted to any University for the award or a Degree or a Diploma. We have properly acknowledged the material collected from secondary sources wherever required. We solely own the responsibility for the originality of the entire content.

Date: 07/07/2021

Name & Signature of the Team Members

Swapnil Nimbalkar

Rahul Nanwani

Vijay Gupta

Name & Signature of the Guide

Prof. Kiran Bidua

Seal/Stamp of the College

Place: Pune

Date: 02/05/2021

ACKNOWLEDGEMENT

We express my profound thanks to my Guide **Prof. Kiran Bidua** for her expert guidance, encouragement, and inspiration during this project work.

We would like to thank **Dr. Reena Pagare**, Project Coordinator, Department Computer Science & Engineering for extending all support during the execution of the project work.

We sincerely thank to **Prof. Dr. Rajneeshkaur Sachdeo**, Head, Department of Computer Science & Engineering, MIT School of Engineering, MIT ADT University, Pune, for providing necessary facilities in completing the project.

We are grateful to **Prof. Dr. Kishore Ravande**, Principal, MIT School of Engineering, MIT ADT University, Pune, for providing the facilities to carry out my project work.

We also thank all the faculty members in the Department for their support and advice.

Swapnil Nimbalkar, MITU17BTCS0137
Rahul Nanwani, MITU17BTCS0135
Vijay Gupta, MITU17BTCS0077

Abstract

The necessity for vehicles to stop or slow down for the toll payment results in traffic congestion and reduces fuel efficiency. Hence, a system that enables road users to pay the toll without stopping or slowing down is proposed. Automatic process of toll collection will save time, effort, and manpower. In this work we propose a low cost and efficient technique called Electronic Toll Collection using GPS module, microcontroller system (raspberry pie) and GSM module that automatically collects the toll based on distance travelled by the vehicle on highways. Software android development kit google maps and python programming language will be used in designing of proposed system. This developed system presents a different approach for highway toll collection which eliminates travel delays and construction of expensive gantries or toll booths.

Toll way road detection and alert driver to note highway and toll rates is an important issue in Automated Toll Payment and Vehicle Tracking System. To solve toll collection issue and to speed up real-time performance, we propose an Android-based solution for highway detection and automated toll payment. As a fast, reliable, and emerging operating system Android becomes the best choice for such automated system. Integrated with google maps platform using Kotlin as an integration language, Android application can track a vehicle in real-time. So, study of technologies like Android, Google Maps Platform and Kotlin is an important task. Such technologies have great developer support which makes them easy to implement. In contrast to traditional approaches, our solution, solving toll payment collection issue with a real-time performance roughly doubled, shows much improvement to existing toll collection systems.

KEYWORDS: Toll Payment, GPS Technology, Android development kit, Open Street Map.

CONTENTS

Chapter 1

Introduction

1.1 Overview

1.1.1 Introduction to the Project

Historically, tolling has been used for revenue collection on major high-capacity arterial highways (e.g., in France and Italy), on some interstate highways in the United States, and for tunnels and bridges. Generally, toll revenues are used to maintain the highway and to recover the costs of construction. Revenue from tunnels and bridges usually can repay only a portion of the construction cost, being used first to finance necessary maintenance and operations.

At present, revenue collection procedures at most toll facilities require a driver to stop his/her car, open the window or door, and find correct coins or a valid card before continuing his/her journey. As the use of tolls becomes more widely accepted, the drawbacks of this conventional toll collection method will be emphasized. Tollbooths suffer from being land intensive, labour intensive (owing to the hiring of toll operators), and time intensive. Electronic toll collection (ETC) systems are superior to manual methods from the perspective of both the toll agency and the user. The technological evolution of this process is slow and incremental.

1.1.2 Motivation behind project topic

By employing automated toll pay system, Driver of vehicles need not to stop at a window or waste time by waiting in a long queue to pay their tolls. This will reduce consumption of fuel; reduce congestion; increase road safety.

1.1.3 Aim and objectives of the work.

Project Aim: The aim of this project is to reduce vehicle congestion and to enhance toll collection thus reducing fuel consumption and emission.

Project Objectives:

- Literature survey of automated Vehicle Tracking system.
- Literature survey of Toll Payment System.
- Proposing a prototype for integrated Vehicle Tracking and Toll Payment System.

- Identification and study of required hardware and software tools.
- Design and implementation of proposed system.
- Performance evaluation of proposed system.

1.1.4 Software Development Model

"Agile software development" refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve via collaboration between self-organizing cross-functional teams. The term was coined in the year 2001 when the Agile Manifesto was formulated.

Agile software development uses iterative development as a basis but advocates a lighter and more people-centric viewpoint than traditional approaches. Agile processes fundamentally incorporate iteration and the continuous feedback that it provides to successively refine and deliver a software system.

There are many agile methodologies, including:

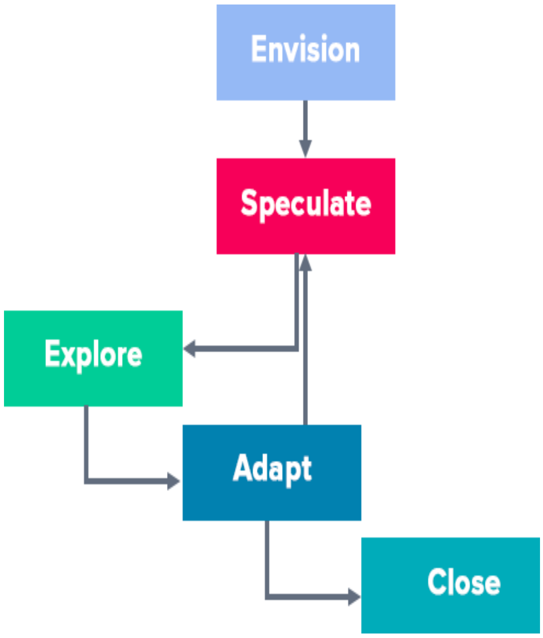
- Dynamic systems development method (DSDM)
- Kanban
- Scrum

Scrum is an agile process framework for managing complex knowledge work, with an initial emphasis on software development, although it has been used in other fields and is slowly starting to be explored for other complex work, research, and advanced technologies. It is designed for teams of ten or fewer members, who break their work into goals that can be completed within timeboxed iterations, called sprints, no longer than one month and most commonly two weeks, then track progress and re-plan in 15-minute time-boxed stand-up meetings, called daily scrums.

Scrum is a lightweight, iterative, and incremental framework for managing complex work. The framework challenges assumptions of the traditional, sequential approach to product development, and enables teams to self-organize by encouraging physical co-location or close online collaboration of all team members, as well as daily face-to-face communication among all team members and disciplines involved.

A key principle of Scrum is the dual recognition that customers will change their minds about what they want or need (often called requirements volatility) and that there will be unpredictable challenge for which a predictive or planned approach is not suited.

As such, Scrum adopts an evidence-based empirical approach accepting that the problem cannot be fully understood or defined up front, and instead focusing on how to maximize the team’s ability to deliver quickly, to respond to emerging requirements, and to adapt to evolving technologies and changes in market conditions.



There are only three roles in SCRUM:

- Product Owner-

  The product owner, representing the product’s stakeholders and the voice of the customer (or may represent the desires of a committee), is responsible for delivering good business results. Hence, the product owner is accountable for the product backlog and for maximizing the value that the team delivers.

- Development Team-

  The development team has from three to nine members who carry out all tasks required to build increments of valuable output every sprint.

- Scrum Master-

  Scrum is facilitated by a scrum master, who is accountable for removing impediments to the ability of the team to deliver the product goals and deliverables. The scrum master is not a traditional team lead or project manager but acts as a buffer between the team and any distracting influences.

Chapter 2

Literature Review

Ying He, Nan Zhao, and Hongxi Yin,” Integrated Networking, Caching, and Computing for Connected Vehicles: A Deep Reinforcement Learning Approach” published in IEEE Transactions on Vehicular Technology, vol. 67, 2018. In this paper, they proposed an integrated framework that can enable dynamic orchestration of networking, caching, and computing resources to improve the performance of next generation vehicular networks. They formulated the resource allocation strategy in this framework as a joint optimization problem, where the gains of not only networking but also caching and computing are taken into consideration in the proposed framework.

Tianyi Song, Nicholas Capurso, Xiuzhen Cheng, Jiguo Yu, Biao Chen, and Wei Zhao,” Enhancing GPS with Lane level Navigation to Facilitate Highway Driving” challenged IEEE Transactions on Vehicular Technology, 2017 by proposing a GPS aiding system that can sense and track a vehicle’s lane position. The system leverages smart phones’ computing capability, rear cam- eras, and inertial motion sensors. With little extra computational overhead, the system applies computer vision techniques to achieve lane level positioning. They also designed a machine learning based algorithm to detect and track lane switching.

Ruipeng Gao, Mingmin Zhao, Tao Ye, Fan Ye, Yizhou Wang, and Guojie Luo,” Smartphone-based Real Time Vehicle Tracking in Indoor Parking Structures” published in IEEE Transactions on Mobile Computing, 2017. According to this paper, they developed algorithms in a Sequential Monte Carlo framework to represent vehicle states probabilistically, and harness constraints by the garage map and detected landmarks to robustly infer the vehicle location. They also proposed landmark (e.g., speed bumps, turns) recognition methods reliable against noises, disturbances from bumpy rides and even hand-held movements.

Y.A. Quintero, G. Patino and J.E. Aedo,” Path-Tracking Algorithm for Intelligent Transportation Systems” published in IEEE Latin America Trans- actions, vol. 14, no. 6, June 2016. This paper describes a path tracking algorithm implemented in an Intelligent Transportation System to validate if a vehicle is following properly a certain path previously assigned. The algorithm is based on a geometrical analysis and GPS data specified to trace in real-time the current position of the vehicle according to a previous digital map of the path to be tracked.

Gabriel Svennerberg,” Beginning Google Maps API 3”, Appress Publica- tion,1st Edition, this book provides the reader with the skills and knowledge necessary to incorporate Google Maps version 3 on web pages in both desktop and mobile applications.

Lan Lake and Reto Meier,” Professional Android”, Wrax Publication,4th Edition, it is comprehensive developer guide to the latest Android features and capabilities Professional Android, it also shows developers how to leverage the latest features of Android to create robust and compelling mobile apps integrated with google API’s like Google Maps.

Marcin Moskala and IgorWojda,” Android Development with Kotlin”, Packt Publication, 1st Edition, this book provides methodology to make Android development much faster using a variety of Kotlin features, from basics to advanced, to write better quality code. This book Leverage specific features of Kotlin to ease Android application development.ax.

Chapter 3

Requirement and Analysis

3.1 **Assumptions and Dependencies**

3.1.1 The user should have knowledge about the system installed in vehicle.

3.1.2 The User should have basic knowledge about UI of application.

3.1.3 The system is developed in Kotlin platform.

3.1.4 The system works on GPS and Internet Facility.

3.1.5 The user should have internet service provider for the vehicle.

3.2 **Functional Requirements**

3.2.1 System Feature

- Automated Toll deduction.
- Theft detection.
- Real-time vehicle tracking.

3.3 **External Interface Requirements**

3.3.1 User Interfaces

- Home Page
- Login Panel.
- Registration Panel.
- Admin Panel.
- User Panel.
- Map Interface.
- Billing Interface.
- Enforcement Panel.

3.3.2 Hardware Interfaces
The entire software requires a completely equipped android system with working GPS and Internet Facility.

3.3.3 Software Interfaces
The system can use Android as the operating system platform. System also makes use of certain GUI tools. To run this application, we need SQLite as database and Apache tomcat as server. Google map API is used to implement map related functions in the application. To store data on server we need MySQL database.

3.3.4 Communication Interfaces

- Communication to the server using Kotlin and servlet APIs through Internet.
- User interface - the display, touchscreen, menus of the system.
- Software interface - Kotlin, Google Map API's and SQLite.

3.4 **Non-functional Requirements**

3.4.1 Performance Requirements
The performance of the system lies in the way it is handled. Every user must be given proper guidance regarding how to use the system. The other factor which affects the performance is the absence of any of the suggested requirements.

3.4.2 Safety Requirements
To ensure the safety of the system, perform regular monitoring of the system to trace the proper working of the system. An authenticated user is only able to access system.

3.4.3 Security Requirements
Any unauthorized user should be prevented from accessing the system. Password authentication can be introduced.

3.4.4 Software Quality Attributes

- Accuracy:
- The level of accuracy in the proposed system will be higher. All operation would be done correctly, and it ensures that whatever information is coming from the center is accurate. Result is organic results.

- Reliability:

  The reliability of the proposed system will be high due to the above stated reasons. The reason for the increased reliability of the system is that now there will be proper storage of information and tweet analysis model.

3.5 **System Requirements**

3.5.1 Database Requirements

- MySQL
- SQLite

3.5.2 Software Requirements (Platform)

- Operating System - Android 5.1 and above
- Application Server - Apache Tomcat
- Front End - Kotlin, XML
- Database - My SQL 5.0, SQLite
- IDE - Android Studio
- API - Google Maps API.

3.5.3 Hardware Requirements

- Processor - Any with clock Speed - 2.1 GHz
- RAM - 1 GB (min)
- Internal Storage - 8 GB (min)
- Display - 5 inches and above

Chapter 4

Technologies Used

4.1 Android

Android is a mobile operating system for touch-screen mobile devices such as smartphones and tablets, based on a modified version of the Linux kernel and additional open-source applications. The Android software stack is made up of a Linux kernel and a set of C/C++ libraries that are exposed through an application framework that handles services, application management, and run time.

Android includes a basic development kit. A full set of development tools is included in the Android software development kit (SDK). A debugger, libraries, a handset emulator based on Quick EMUlator (QEMU), documentation, sample code, and tutorials are among the tools available. On the Android OS, programmes are packed in .apk file and placed in the /data/app subdirectory.

4.2 Open Street Routing Machine

The Open Street Routing Machine is a high-performance routing engine for shortest paths in road networks written in C++. It blends advanced routing algorithms with the OpenStreetMap (OSM) project's open and free road network data. If you do not use a so-called speedup-technique, computing the shortest path on a continental-sized network can take several seconds.

4.3 Apache HTTP Server

The Apache HTTP Server, also known as Apache, is a cross-platform web server that is free and open source. Apache has many features, many of which are implemented as built modules that enhance the basic functionality. Authentication systems and server-side programming languages like Perl, Python, Tcl, and PHP are just a few examples. Apache has adjustable error messages, authentication databases based on relational databases, content negotiation, and numerous graphical user interfaces (GUIs). It accepts both password and digital certificate authentication.

Anyone can adjust the server for unique needs because the source code is publicly available, and there is a vast public library of Apache add-ons. Instead of adopting a single architecture, Apache provides several Multiprocessing Modules (MPMs) that enable it to function in a process-based mode, a hybrid mode, or an event-hybrid (process and thread) mode, depending on the needs of the infrastructure.

4.4 Retrofit

Retrofit is a type-safe REST client for Android and Java that attempts to make RESTful web service consumption easier. Retrofit makes consuming JSON or XML data that has been parsed into Plain Old Java Objects (POJOs). Retrofit is the class responsible for converting your API interfaces into callable objects. Your HTTP API becomes a Java interface with Retrofit.

Retrofit 2 is built on top of OkHttp, which is used as the networking layer by default. Retrofit uses a POJO (Plain Old Java Object) to automatically serialise the JSON response, which must be defined in advance for the JSON Structure. Interceptors in OkHttp are a sophisticated method for monitoring, rewriting, and retrying calls.

4.5 MySQL

MySQL is an open-source relational database management system (RDBMS) based on Structured Query Language that is backed by Oracle (SQL). The LAMP web application software stack (and others) includes MySQL. LAMP stands for Linux, Apache, MySQL, Perl/PHP/Python. MySQL has a client-server architecture.

MySQL's fundamental component is the MySQL server, which manages all database commands (or commands). The MySQL server is available as a standalone programme for usage in a client-server networked environment, as well as a library that may be integrated (or linked) into other programmes. Data can be stored and accessed in MySQL using a variety of storage engines, including InnoDB, CSV, and NDB. MySQL may also replicate data and divide tables for improved efficiency and longevity.

Chapter 5

Design

5.1 System Architecture

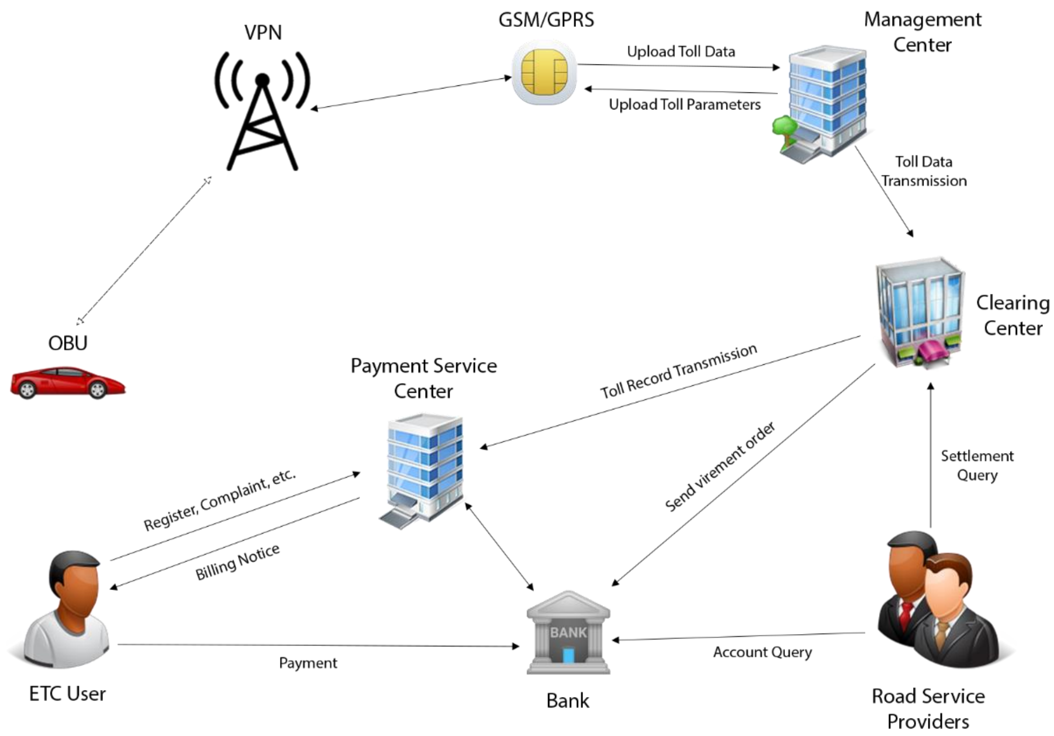


Figure 5.1 System Architecture

5.1.1 OBU (On-Board Unit)

Consisting of a motherboard, GPS module and GSM module. It will be mounted on user vehicle to track the vehicle.

5.1.2 Connectivity Framework

It will be used to establish connectivity between the user vehicle and management center. Data collected from user vehicle OBU will be shared with Management Center through Connectivity Framework.

5.1.3 Management Center

It will determine the toll charged according to the vehicle profile and data received and maintain a record of the vehicle regarding distance travelled paid receipts.

5.1.4 Clearing Center

Data collected from management center will be cleaned and analysed to get more knowledge on traffic patterns and need to implement good strategies to control traffic.

5.1.5 Payment Service Center

It will generate bills according to distance travelled by vehicle and process the payment cycle and fund management.

5.1.6 Road Service Providers

Road service provides are highway authorities and RTO authorities which will share the information regarding toll charges and vehicle registration documents.

5.2 **Flowcharts**

5.2.1 System Workflow

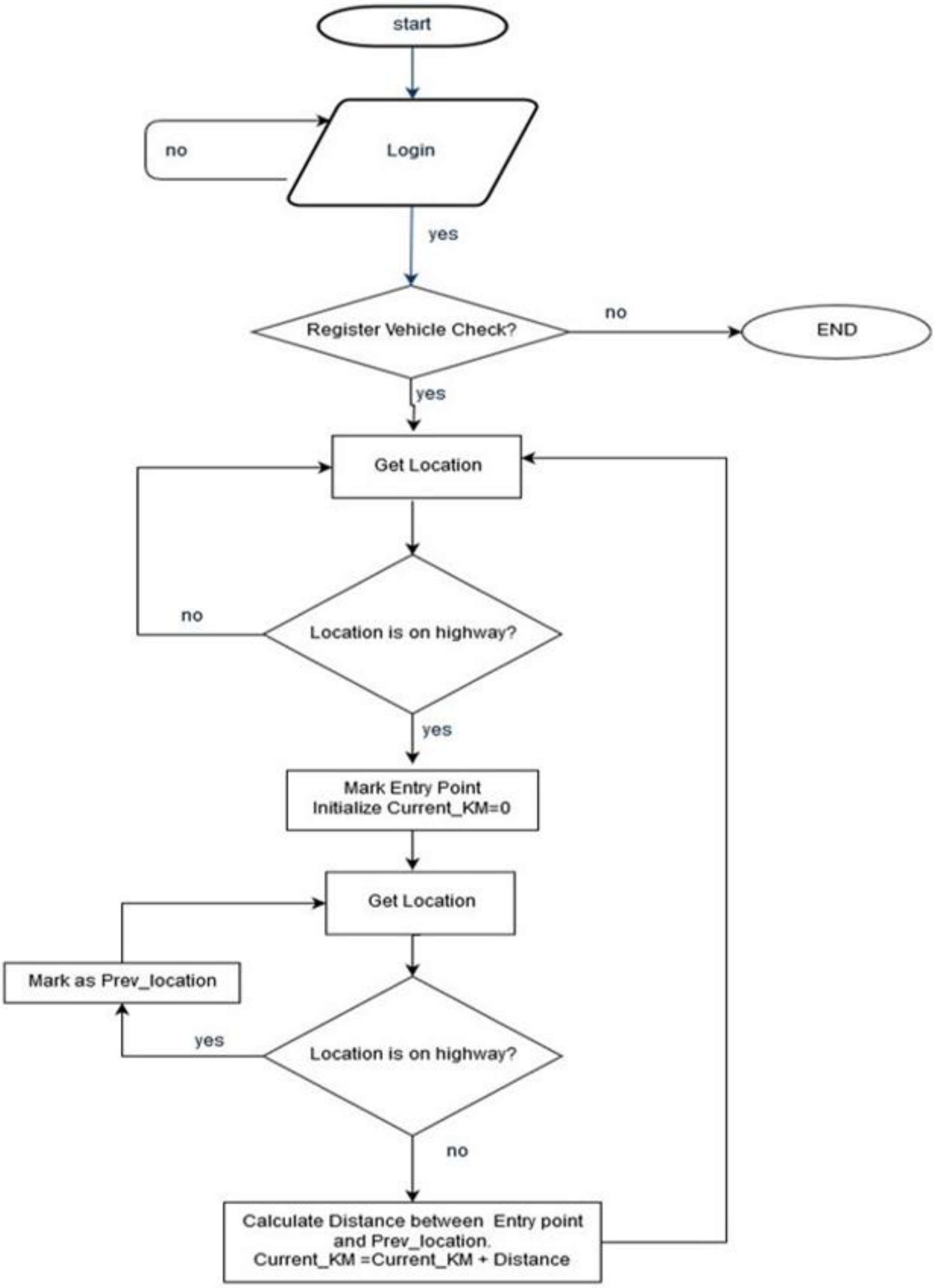


Figure 5.2 System Workflow

5.2.2 Payment Cycle

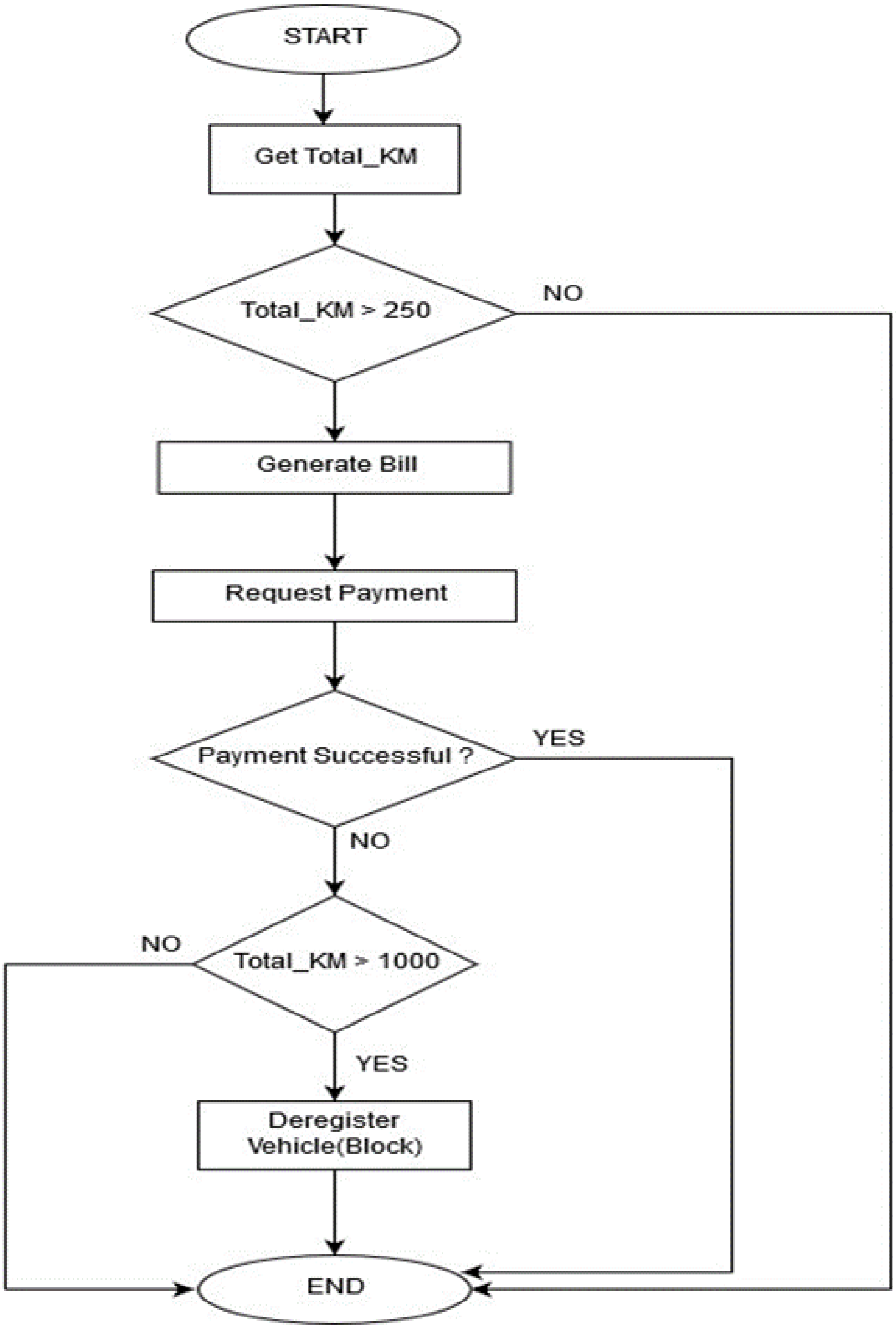


Figure 5.3 Payment Cycle

5.3 **UML Diagrams**

5.3.1 Use Case Diagram

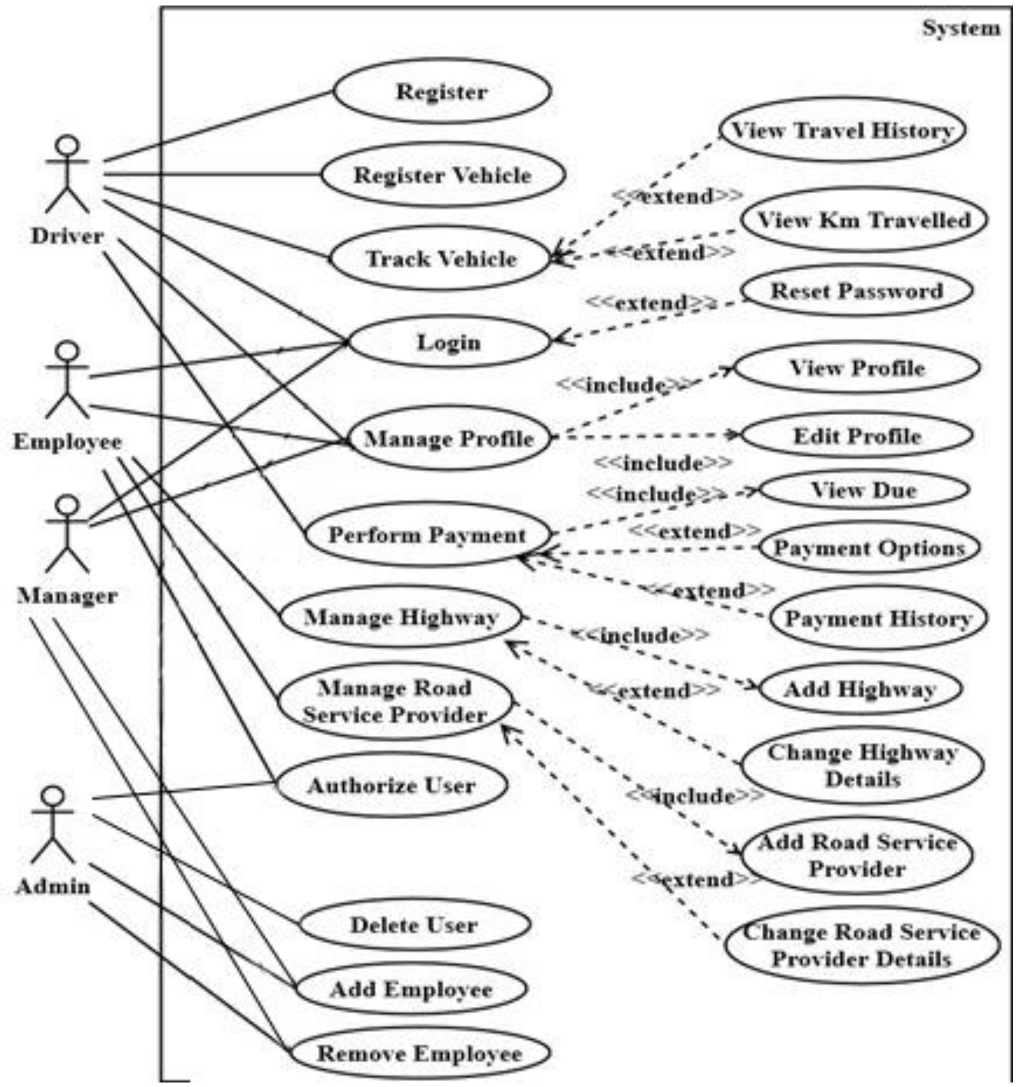


Figure 5.4 Use Case Diagram

5.3.2 Class Diagram

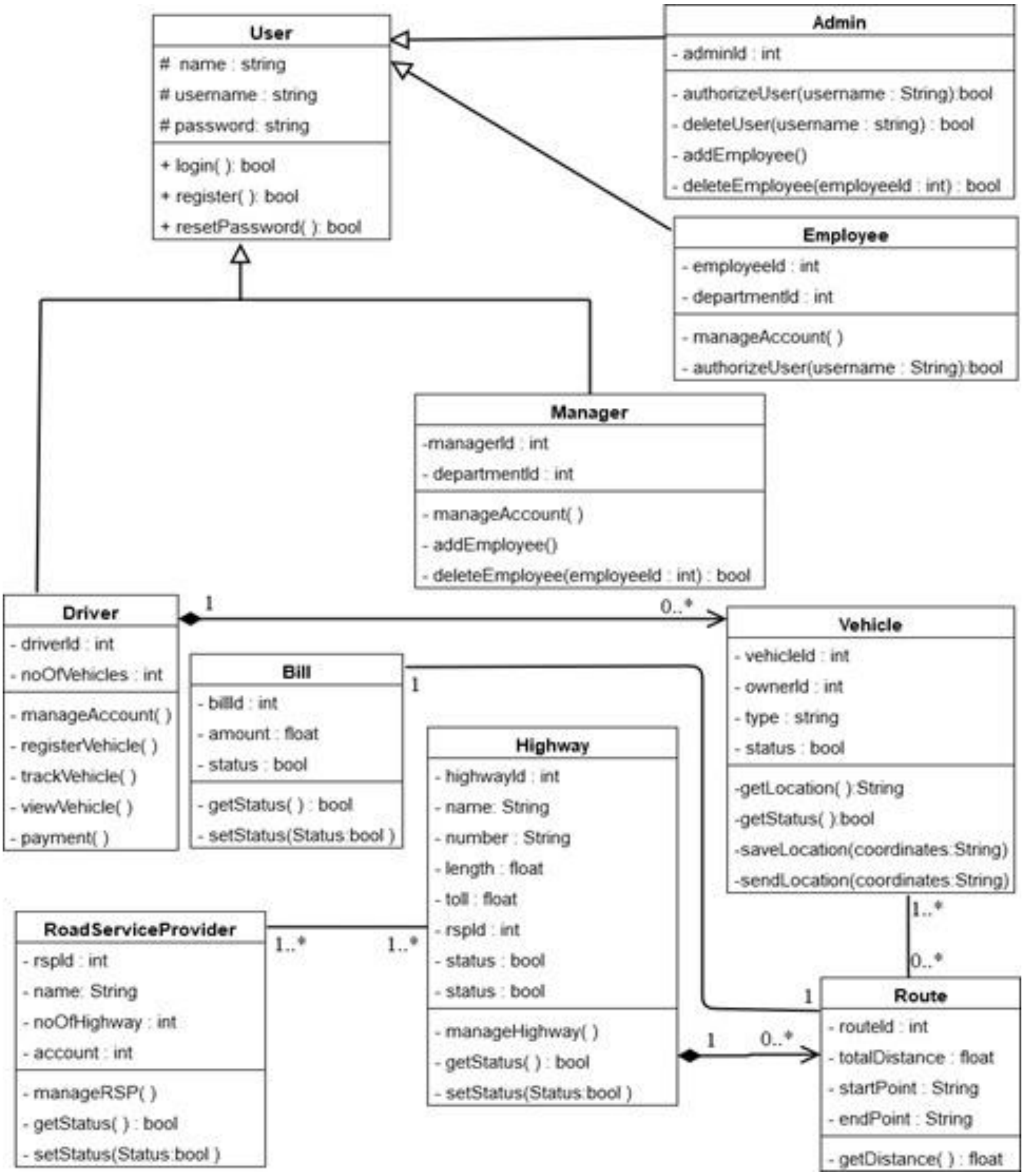


Figure 5.5 Class Diagram

5.3.3 Activity Diagram

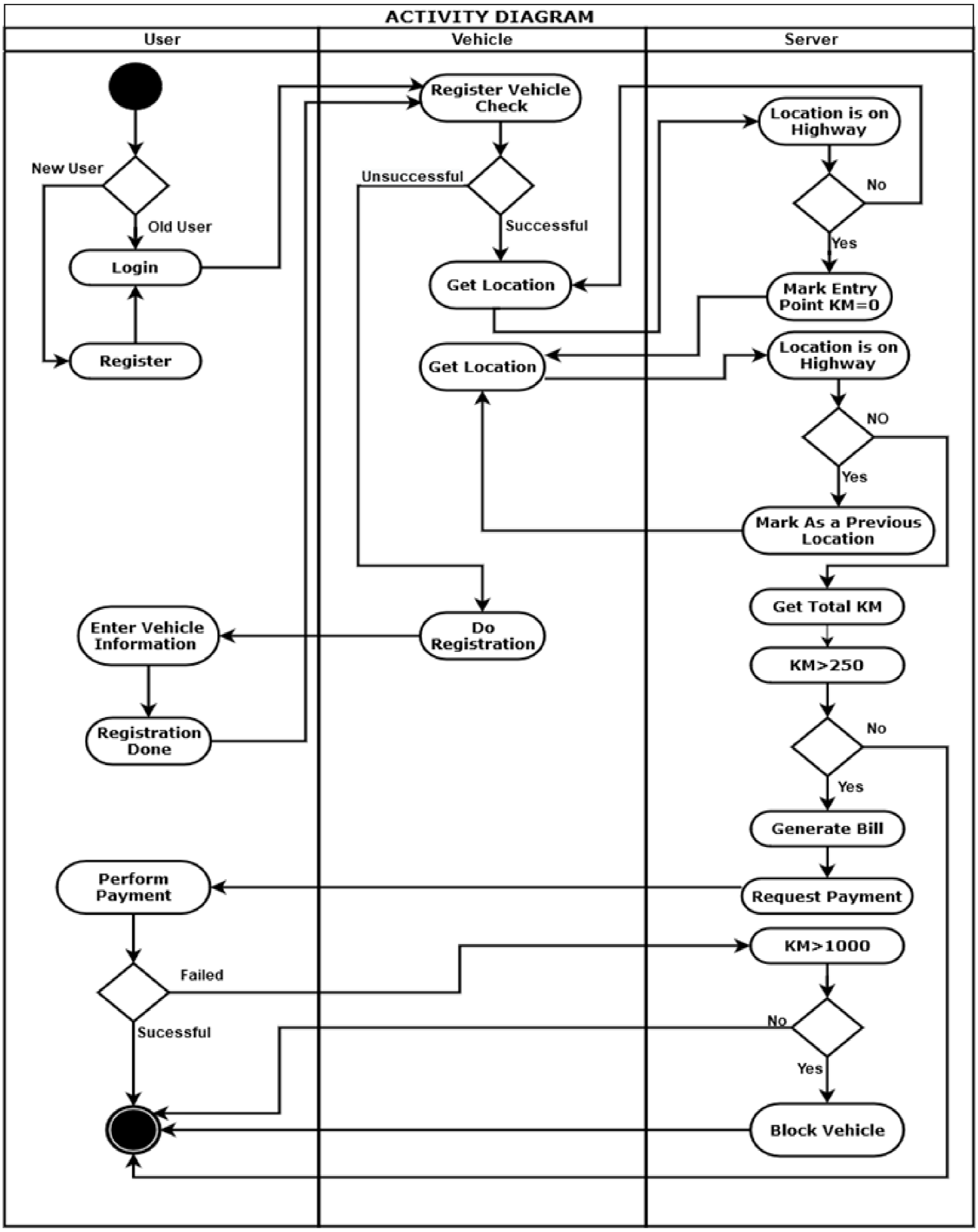


Figure 5.6 Activity Diagram

Chapter 6

Modules

6.1 **Login**

Input: Username, Password

Output: login status

This module will authenticate the user and allow to access and monitor all vehicle profiles under his name. Username and password will be verified using server-side database.

6.2 **Register Vehicle**

Input: Username, Vehicle Details

Output: Vehicle ID

Module will validate and enter vehicle details in database under username. If user tried to register same vehicle twice the module will return existing vehicle ID. This module can deregister and block the vehicle as well.

6.3 **Get Location**

Input: Vehicle ID

Output: GPS Data

Module will validate vehicle ID collect the GPS data from GPS module. After formatting the GPS data in longitude and latitude format, it will send it to the server.

6.4 **Highway Check**

Input: GPS Data

Output: Highway ID

Module will collect the GPS data send it to Roads API. Roads API will return a universal highway ID to Highway Check module which in turns will return this to the server.

6.5 **Trace Route**

Input: Highway ID

Output: Total kms

This Module will keep track of GPS data and will trace the route of vehicle on highway. It will calculate the distance travelled on highway and send Total KM to Server.

6.6 **Payment**

Input: Total kms

Output: Payment Status

Module will validate vehicle ID; Km travelled and perform the payment processing. The status of payment and Bill ID will be sent to user and updated on the database.

The Automated toll payment and vehicle tracking system provides accurate data in real time that makes it possible for the user to track the vehicle, enables road users to pay the toll without stopping or slowing down and it also enable an early retrieval if the car is stolen. Implementation of GPS tracker in vehicle can certainly bring revolutionary change in developing country like India where there is very high urban as well as rural vehicular transition every day. This will result in reduced traffic congestion and improvised fuel efficiency. Implementation of this system ensures fair toll collection policy.

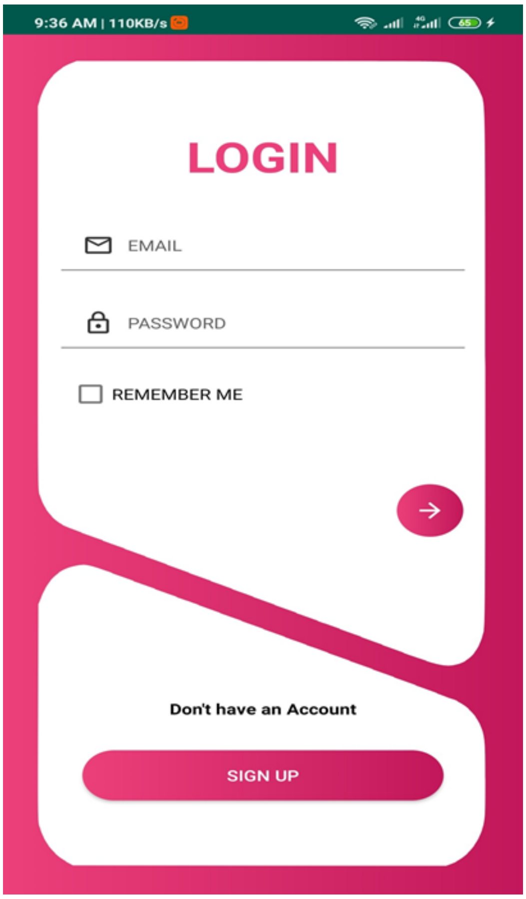
There can be various other applications that can be built over the existing platform to make this system more efficient. The features of this system are not only limited, but more technical features can also be developed to enhance this tracking purpose. For example, a camera can be placed inside the car along with this tracing device, with the aim to protect the vehicle from unauthorized users. On installing a camera inside, it will enable to have photos of rightful owner along with audio and video, with the photos being sent over the M2M link. Even the videos or pictures of current location of the vehicle can be sent using such technology to track down the vehicle if it gets to an unauthorized user or to an unauthorized location. Therefore, a wide range of applications can be implemented to make automated toll payment and vehicle tracking system a more desirable and beneficial as well.

Chapter 7

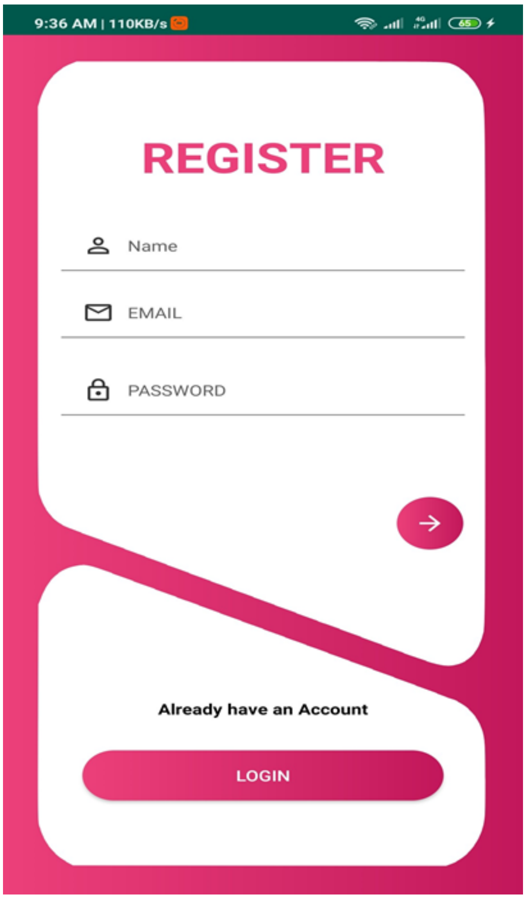
GUI

7.1 GUI for User Application

7.1.1 Login Screen

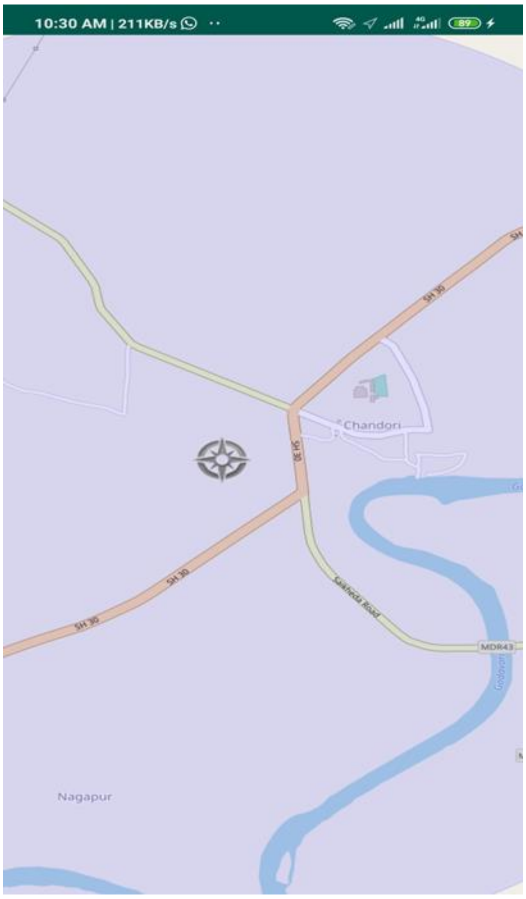


7.1.2 Register Screen



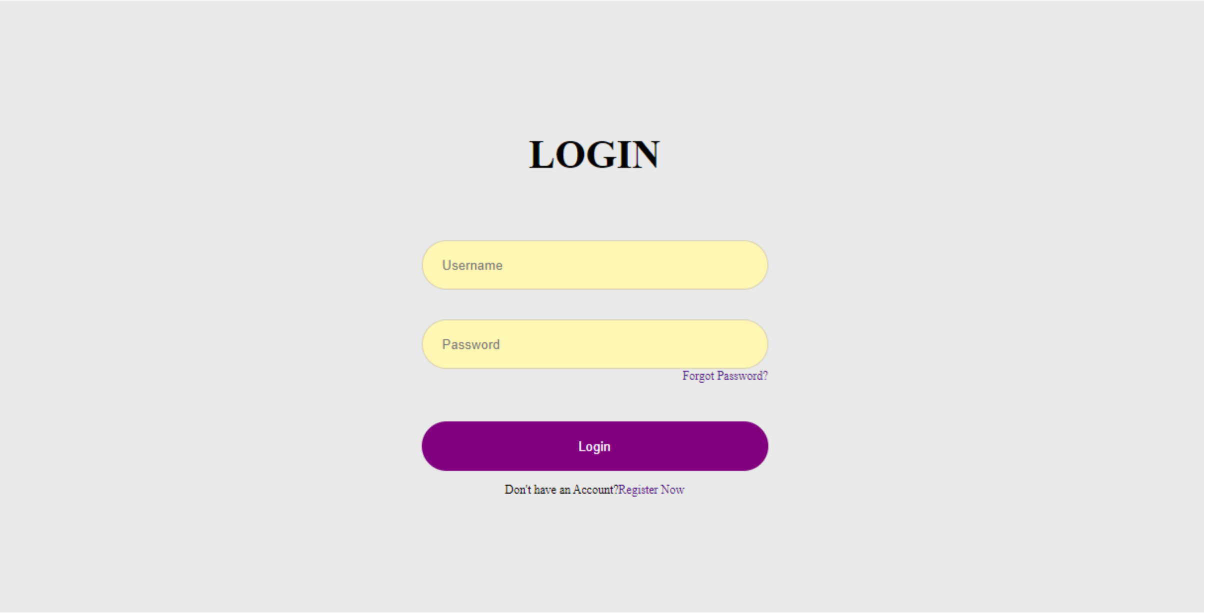
7.1.3 Dashboard Screen

7.1.4 Map Screen

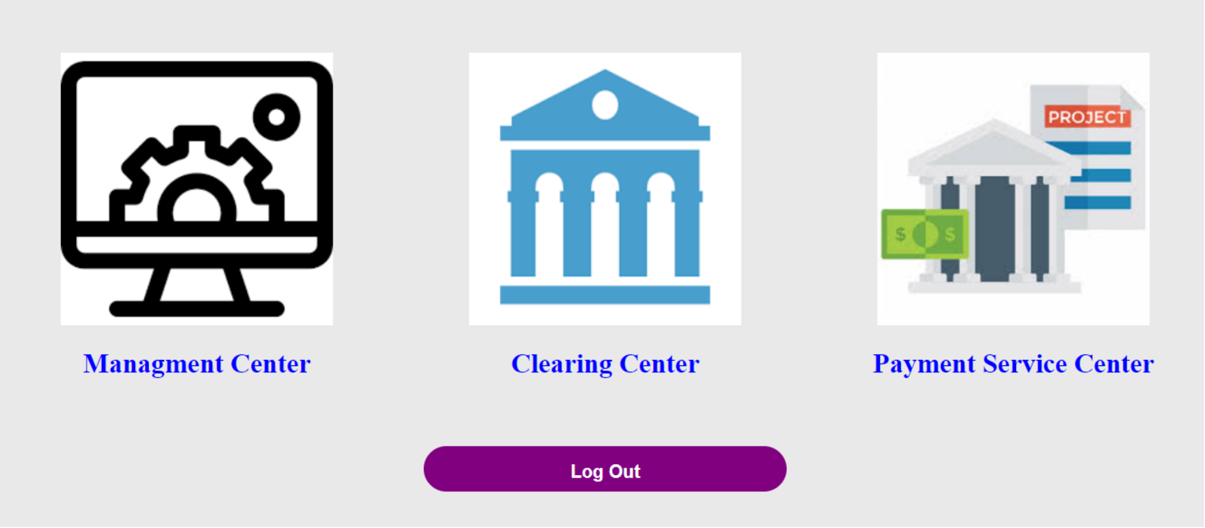


7.2 GUI for Management Authority

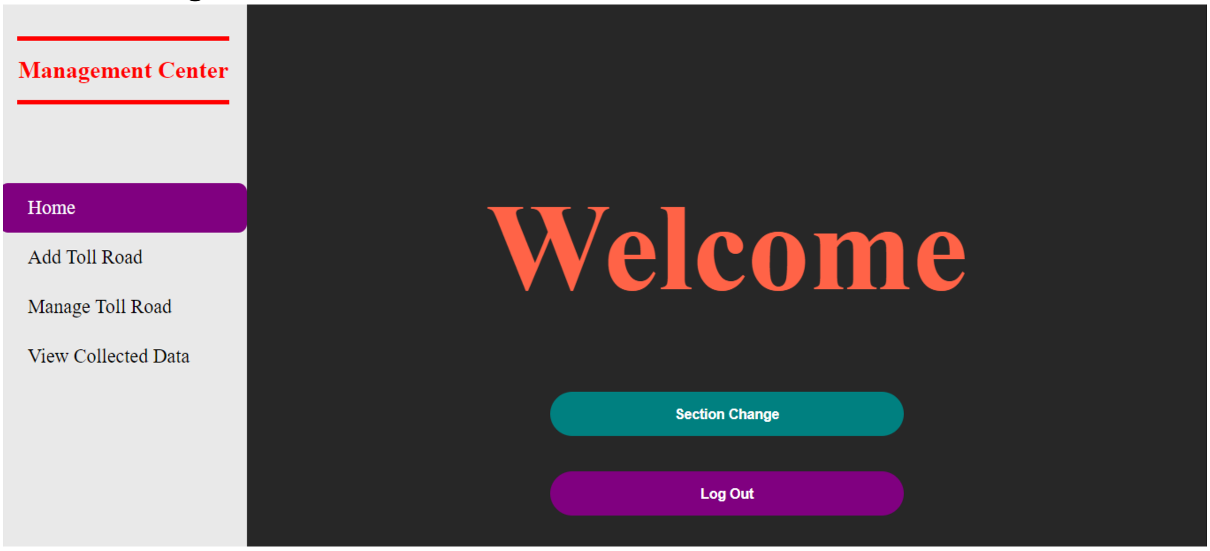
7.2.1 Login Screen



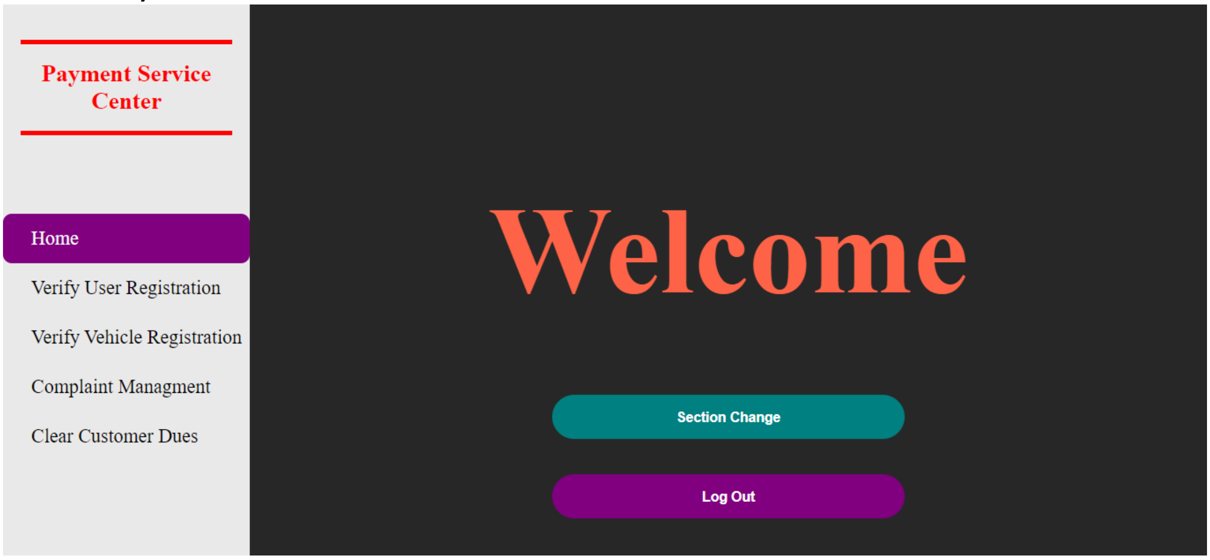
7.2.2 Choose Center Screen



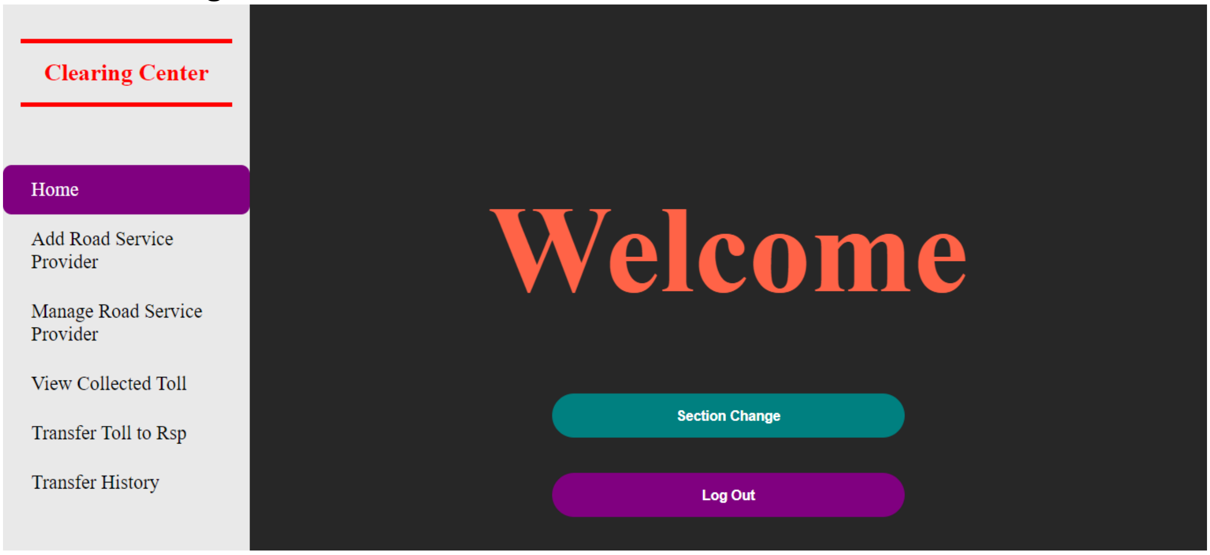
7.2.3 Management Center Screen



7.2.4 Payment Center Screen



7.2.5 Clearing Center Screen



Chapter 8

Implementation

8.1 **Methodology**

Implementing GUI, developing backend procedures, connecting to the server, and data processing are the four key steps of the implementation. We'll now go over each of the phases in more detail.

8.1.1 Implementing GUI

Designing and developing design code for each user interface is part of the GUI implementation process. The GUI is implemented using three languages: XML, HTML, and CSS.

The GUI screen of the android app that will be utilised by the Vehicle Owner is designed using XML. The actual UI (User interface) of our application is defined by layout xml files. It contains all the views (or tools) that we used in our application. TextViews, Button, and other UI elements are examples. The layout XML files are then joined to the android activity to display the GUI interface. HTML and CSS are used in conjunction with one another. They are used to develop a web-based graphical user interface (GUI) that will be utilised for system management. Hyper Text Mark-up Language is the abbreviation for Hyper Text Mark-up Language. It describes a Web page's structure. It is made up of several components. These elements instruct the browser on how to render the content. CSS specifies how HTML elements should appear on a screen, in print, or in other media.

8.1.2 Developing backend procedures

The backend method for an Android app is developed using Activities and Services. An activity is the application's entry point for communicating with the user. A single screen with a user interface is represented by an activity. A service is a background component that performs long-running activities. Services are used to continuously track the location of a vehicle while running in the background.

Python is being used to create a server-side service that handles vehicle tracking and routing. This service is always running in the background, processing the GPS data received by automobiles.

8.1.3 Connection with server

PHP is a server-side scripting language that allows you to connect to the server. PHP scripts running on the server are utilised to allow the android app to communicate with the server to share GPS data.

The Android app connects to the server and uses retrofit to access the PHP scripts running on the server. Retrofit is an Android and Java HTTP client that is type-safe. JSON or XML files are used to transfer data between client and server, and Retrofit makes handling them simple.

8.1.4 Data processing

The data processing received by the vehicle in operation is handled by the python service running on the server. The MySQL-connector is used to retrieve data from the server by the service. It uses routing engine capabilities like route and match, which are available in open street routing machines, to process GPS data. Following data processing, vehicle routes are obtained and saved in the database. This service also manages the processing of travel routes and the generating of related bills.

Chapter 9

Results

Vehicle tracking begins as soon as the vehicle moves. GPS coordinates are captured and stored in a central database utilising the GPS in the vehicle. The coordinates are recorded to the database with the current timestamp as they are received.

While driving on the highway, the following coordinates were obtained (NH65).

id	longitude	latitude	timestamp	vehicle_id
2	73.96944522857667	18.493569437829354	2021-07-06 16:37:38.000000	1
3	73.96947205056215	18.493747494423037	2021-07-06 16:37:51.000000	1
4	73.96949887254762	18.49392555101672	2021-07-06 16:37:54.000000	1
5	73.96952569453309	18.494103607610402	2021-07-06 16:37:56.000000	1
6	73.96955251651856	18.494281664204085	2021-07-06 16:37:59.000000	1
7	73.96957933850403	18.494459720797767	2021-07-06 16:38:01.000000	1
8	73.9696061604895	18.49463777739145	2021-07-06 16:38:03.000000	1
9	73.96963298247498	18.494815833985133	2021-07-06 16:38:05.000000	1
10	73.96965980446045	18.494993890578815	2021-07-06 16:38:08.000000	1
11	73.96968662644592	18.495171947172498	2021-07-06 16:38:10.000000	1

Coordinates saved in database are then processed using open street routing machine by service running on the server. Coordinates are then mapped to nearest route on the map.

Following the mapping, sets of coordinates belonging to the same highway/road are joined to establish a vehicle route, and the distance travelled is calculated and saved in a database.

After generating routes, the following findings are obtained.

id	highway_id	meters	start	end	vehicle_id	timestamp	status
6	1	5832	18.49540089,73.95350	18.49554334,74.01062	1	2021-07-06 16:02:42.294083	nobill

CONCLUSION AND FUTURE WORK

The Automated toll payment and vehicle tracking system provides accurate data in real time that makes it possible for the user to track the vehicle, enables road users to pay the toll without stopping or slowing down and it also enable an early retrieval if the car is stolen. Implementation of GPS tracker in vehicle can certainly bring revolutionary change in developing country like India where there is very high urban as well as rural vehicular transition every day. This will result in reduced traffic congestion and improvised fuel efficiency. Implementation of this system ensures fair toll collection policy.

There can be various other applications that can be built over the existing platform to make this system more efficient. The features of this system are not only limited, but more technical features can also be developed to enhance this tracking purpose. For example, a camera can be placed inside the car along with this tracing device, with the aim to protect the vehicle from unauthorized users. On installing a camera inside, it will enable to have photos of rightful owner along with audio and video, with the photos being sent over the M2M link. Even the videos or pictures of current location of the vehicle can be sent using such technology to track down the vehicle if it gets to an unauthorized user or to an unauthorized location. Therefore, a wide range of applications can be implemented to make automated toll payment and vehicle tracking system a more desirable and beneficial as well.

BIBLIOGRAPHY

[I] Ying He, Nan Zhao, and Hongxi Yin," Integrated Networking, Caching, and Computing for Connected Vehicles: A Deep Reinforcement Learning Approach", IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, VOL. 67, 2018.

[II] Tianyi Song, Nicholas Capurso, Xiuzhen Cheng, Jiguo Yu, Biao Chen, and Wei Zhao, Enhancing GPS with Lane-level Navigation to Facilitate Highway Driving, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, 2017.

[III] Ruipeng Gao, Mingmin Zhao, Tao Ye, Fan Ye, Yizhou Wang, and Guojie Luo, Smartphone-based Real Time Vehicle Tracking in Indoor Parking Structures, IEEE Transactions on Mobile Computing,2017.

[IV] SeokJu Lee, Girma Tewolde, Jaerock Kwon, Design and Implementation of Vehicle Tracking System Using GPS/GSM/GPRS Technology and Smartphone Application, IEEE World Forum on Internet of Things (WF-IoT), 2014 Gabriel Svennerberg," Beginning Google Maps API 3", Appress.

[V] Lan Lake and Reto Meier, Professional Android, John Wiley and Sons Publication, 4th Edition, 2018.

[VI] Marcin Moskala and Igor Wojda, Android Development with Kotlin, Packet Publication, 1st Edition, 2017.

[VII] Marcin Moskala and IgorWojda," Android Development with Kotlin" Packet Publication, 1st Edition, 2017.

[VIII] Samra J. Comparing Performance of Plain PHP and Four of Its Popular Frameworks. Thesis Project. Linnaeus University, Sweden.

[IX] Nylén. H. PHP Framework Performance for Web Development Between Codeigniter and CakePHP. Bachelor Thesis.