USMAN INSTITUTE OF TECHNOLOGY

Affiliated with NED University of Engineering & Technology, Karachi

Department of Computer Science

B.S. Computer Science / Software Engineering

FINAL YEAR PROJECT REPORT

Batch-2019

VEHICLE DETECTION SYSTEM

By

ADNAN SAMAD 19B-011-SE

M.OMER SIDDIQUI 19B-004-SE

ALISHA HAMEED 19B-017-SE

HAFIZ TAHA SHARIF 19B-035-SE

Supervised by

DR SHARAF HUSSAIN

ST-13, Block 7, Gulshan-e-Iqbal, Abul Hasan Isphahani Road, Opposite Safari Park, P.O. Box 75300, Karachi, Pakistan. Phone: 34978274-5; 34994305; 34982476; http://www.uit.edu

Declaration

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at USMAN INSTITUTE OF TECHNOLOGY or other institutions.

ADNAN SAMAD (19B-011-SE)

Signature: _____

M.OMER SIDDIQUI (19B-004-SE)

Signature: _____

ALISHA HAMEED (19b-017-SE)

Signature: _____

HAFIZ TAHA SHAREEF (19B-035-SE)

Signature: _____

Date: _____

Acknowledgments

Abstract

Table of Contents

TABLE OF FIQURE

TABLE LIST

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION

As Technology is being used in our everyday life to overcome the problems. We use different things together to create inventions. There are a lot of technologies to control over crimes some are prefect and cheap solution but some are very expensive solution. A car detection system is a technology used to identify and track vehicles. It is typically used in traffic monitoring and control, as well as in automotive safety. Car detection systems can be used to monitor speed, identify vehicles that are in violation of traffic regulations, and track stolen vehicles. RFID technology is used for various purposes. For Example

- Asset management and inventory
- Patient identification
- Medical device tracking
- Control of infectious diseases
- Vehicle Detection system

RFID

RFID is an automatic identification method, relying on storing and remotely retrieving data using devices called RFID tags or transponders. An RFID tag is an object that can be applied to or incorporated into a product, animal, or person for the purpose of identification using radio waves. There are several types of RFID tags, but the most common is called passive RFID tags. Passive RFID tags have no internal battery, so power is supplied by the reader . They are much cheaper than active RFID tags and are the preferred choice for many applications. RFID is used in many applications including toll plaza, parking, access control, supply chain management, livestock tracking and many other tracking and identification applications.

RFID is a technology that uses radio waves to identify objects. RFID In car detection system is a system that uses RFID technology to detect the presence of cars. RFID In car detection system can be used to detect the presence of cars in a parking lot, garage, or other area. RFID In car detection system can also be used to track the movements of cars.

RFID works in car detection system by sending a radio signal to interrogate an electronic tag attached to a vehicle. The tag replies with a serial number, identifying the vehicle. The tag contains electronically stored information that can be read from a distance.

RFID can be used in car detection system in various ways. For example, it can be used to track the location of vehicles in a parking lot. It can also be used to identify vehicles that are entering or exiting a parking lot. Additionally, RFID can be used to track the movements of vehicles in a traffic congestion area.

1.2 PROBLEM STATEMENT

RFID car detection system is to design a system which can detect the presence of a car using RFID tags. The system should be able to distinguish between different cars and should be able to provide information about the car such as the make, model and year. The system will also be able to provide information about the location of the car.

RFID tag which can be attached to a car. The tag should be able to withstand the harsh environment and should be able to communicate with the reader even when the car is moving.

A reader which can be installed to reader the RFID tags should be able to read the tags of the cars and should be able to provide information about the car to the system.

A system which can process the information received from the reader and can provide information about the car to the user.

1.3 PROPOSED SOLUTION

In this project we will install the RFID reader on different instances which will be able to read the RFID tags install in cars. Every RFID tags will have their unique number once this RFID tag will be passes from any of the RFID reader the scanner will record the time, location and RFID Tag number and will send it to the database at the backhand. So that if the concern data will be required it can be access easily.

1. The car detection system should be able to detect cars in real-time.
2. The system should be able to identify the make, model, and color of the car.
3. The system should be able to track the car's movements.

4. The system should be able to automatically record the license plate number of the car.
5. The system should be able to generate a report.
6. Implement an RFID tag system on all vehicles.
7. Use RFID reader to scan for tags.
8. Record the tag number and time of entry for each vehicle.

1.4 PROJECT SCOPE

The RFID scanner can scan the number and match the information of the owner and match this to the excise database. Moreover, the location of the vehicle will be sent to the control center, which will be helpful to identify the actual location in the city.

The scope of this project includes and excludes the following items.

1.4.1 IN SCOPE:

- To secure the vehicles in a way that if it is stolen, it will be detected through scanner installed on different location with the help of RFID tags and map it on Google Map with the last scanned location.
- To provide the relevant information to the excise department of vehicle and taxation so that they could know the location of their vehicle.
- To provide all the information of the vehicles to excise department so that no-one could be able to run the vehicle without registering & without paying tax of the vehicle.

1.4.2 OUT OF SCOPE:

- Hand-held scanning devices for concern officers.
- Tax can be paid.
- Concern officers can add Tax challan of the vehicle.

1.5 MOTIVATION

The RFID car detection system is designed to automatically detect the presence of vehicles in a given area. The system can be used to monitor the flow of traffic in a parking lot or garage, or to

track the movements of vehicles in a fleet. The system is based on the principle of radio frequency identification (RFID), which uses radio waves to identify objects. RFID tags are attached to objects, and an RFID reader is used to read the tags. In the case of the car detection system, the reader is connected to a computer that is programmed to automatically detect the presence of vehicles in the given area. The system is designed to reduce the need for manual monitoring of traffic flow.

1.6 SYSTEM DIAGRAM

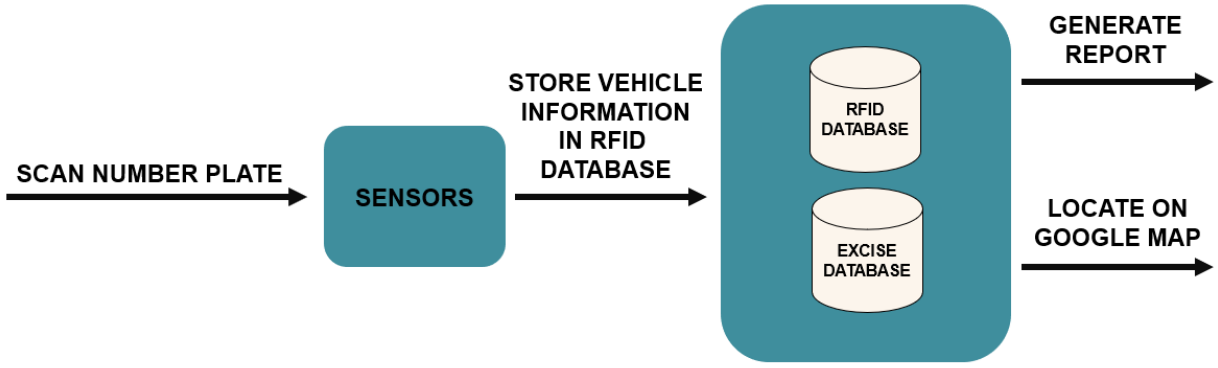


Figure 1 Working system diagram for the project

At first the scanner scans rfid card which has been pivoted in number plates. This sensor information then be sent to the server (database). The other taxation database stores information related to owner. At last the information generated through google maps and report.

1.7 BUSINESS VALUE OR BENEFITS OF PROJECT

The RFID car detection system provides a number of benefits for businesses, including:

1. Increased security: The system can help businesses to keep track of their vehicles and prevent theft.
2. Reduced costs: The system can help businesses to save money on insurance and fuel costs.
3. Improved efficiency: The system can help businesses to improve their fleet management and operations.

4. Enhanced customer service: The system can help businesses to provide better customer service by tracking vehicles and providing information about their location.
5. Increased sales: The system can help businesses to increase their sales by providing information about their vehicles' location and availability.

1.8 CONCLUSION OR FUTURE SCOPE

This system is using one of the most advanced technologies, Radio Frequency Identification or RFID. This technology is able to detect car, and to identify it. The system is able to detect a car from a marked point.

This system can be further improved by adding features like storing the count of vehicle in a non-volatile memory. This helps to count vehicles even if the system is switched off.

CHAPTER 2
BACKGROUND AND LITERATURE REVIEW

2.1 BACKGROUND AND LITERATURE REVIEW

Our system rely on RFID (Radio frequency identification) chips and scanner which is integrated with gps and this scanner senses the rfid number which is pivoted in the license plate and sends longitude and latitude information against this vehicle rfid and points this location on google maps similarly more scanners have to be deployed to track vehicle with the points and gives the path from where the vehicle passes. And verifies license plate RFID with the owner information. This will check the tax information which is connected with the excise database.

2.2 SIMILAR APPLICATIONS

A fleet management system is a system used to track and optimize the performance of a fleet of vehicles. The system typically includes GPS tracking, vehicle diagnostics, driver behavior analytics, and more. In our project, The goal is same to locate and track the vehicles but not for single company or organization but to provide solution for vehicle tracking in all over Pakistan.

YouTube video explanation of a similar project (1566) IVIS RFiD LIcense Plate - YouTube

(1566) Fleet Management System with Live GPS Tracking in PHP | Free Source Code Download - YouTube [1] (Gps) Fleet Management System with Live GPS Tracking in PHP Free Download is a Vehicle Management System for managing vehicles. Fleet Management System with Live GPS Tracking is system can able to manage vehicles and drivers, customers, keep a track of income and expenses and get detailed reports. Real time GPS tracking and Geofence is available in the system, which help to track location. Geofence help to track eye on inbound and outbound of particular marked location. This system gives a unique tracking URL of trip and can able to share, the URL to customer to keep tracking.

HOME - RFID PAKISTAN [2]

RFID vehicle tracking system, Vehicle asset tracking | RAMP (ramprfid.com) [3]

2.3 RFID SYSTEM (FLEET MANAGEMENT SYSTEM) PRODUCT LINKS AND COMMON COMPARISON

Title (archive.org) (history of rfid)

GPS vs RFID asset tracking system - WandResearch (wand-research.com)

RFID in Motion: High-Speed Applications - atlasRFIDstore

What is RFID? | The Beginner's Guide to How RFID Systems Work | atlasRFIDstore

HOME - RFID PAKISTAN

RFID vehicle tracking system, Vehicle asset tracking | RAMP (ramprfid.com)

2.4 STUDY FOR RFID IN CONTAINER AREA

The paper presents a new method for image retrieval based on the use of a bag of words representation. The method is tested on a database of images of outdoor scenes. The results show that the proposed method outperforms the state of the art in terms of accuracy and efficiency. Rfid adoption for the container area where it discusses about the test that passes by an rfid.

A study of RFID adoption for vehicle tracking in a container terminal (upc.edu) [4]

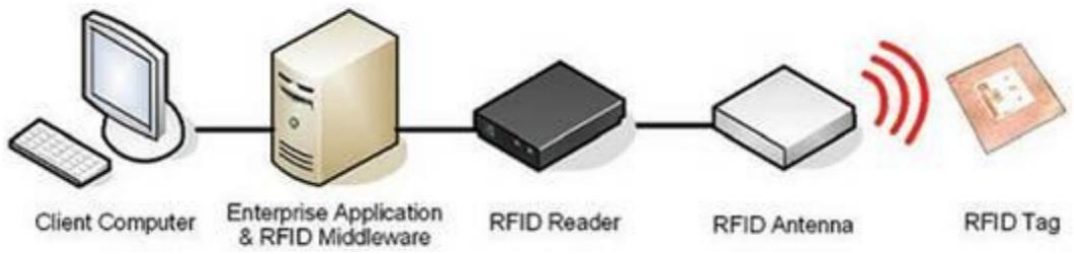


Figure 2 Basic Components of an RFID System

2.5 ARCHITECTURE DESIGN OF THE VEHICLE TRACKING SYSTEM BASED ON RFID

These are the samples of rfid vehicle tracking in the research article which is about cargo tracking but is similar to the vehicle tracking system which we are about to create in our system. Similarly, the scanners being placed on different locations to be able to track the vehicle at particular point. When the vehicle passes through this point scanners pins its location and inform the database that this vehicle passes by me at this time. And we are able to draw location on google map with the help of these points.

2317-3152-1-PB.pdf

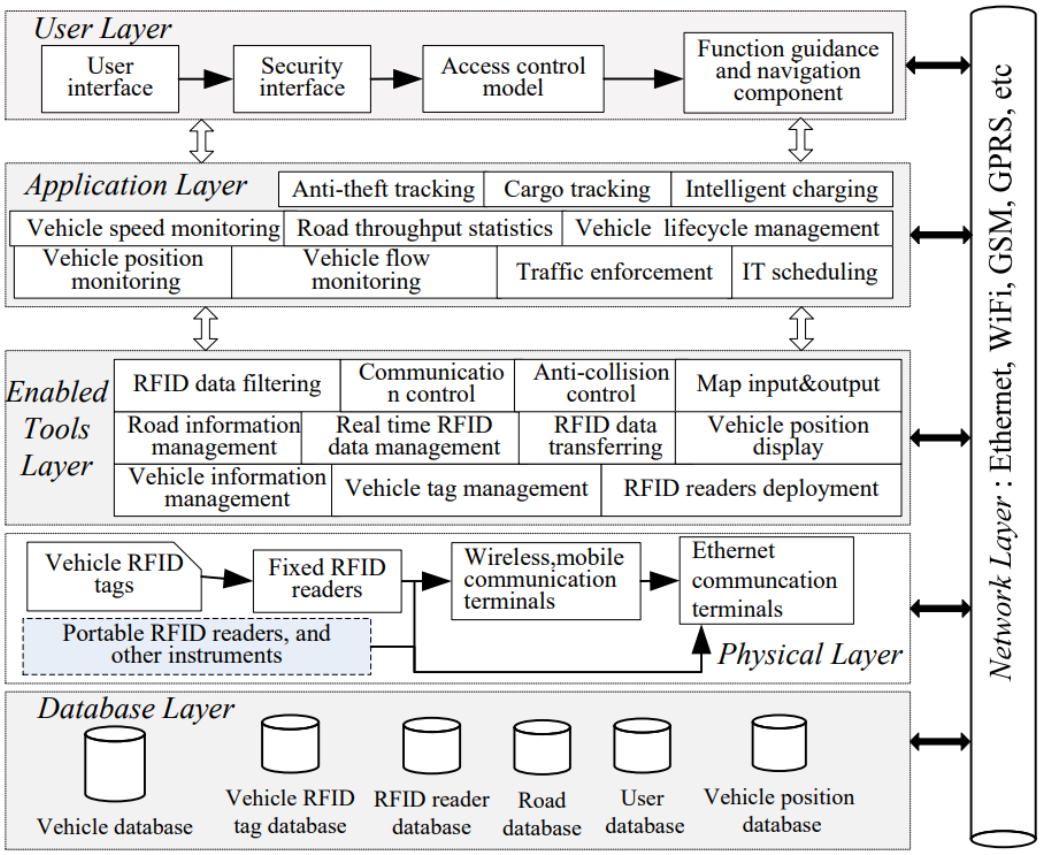


Figure 3 Vehicle system layer

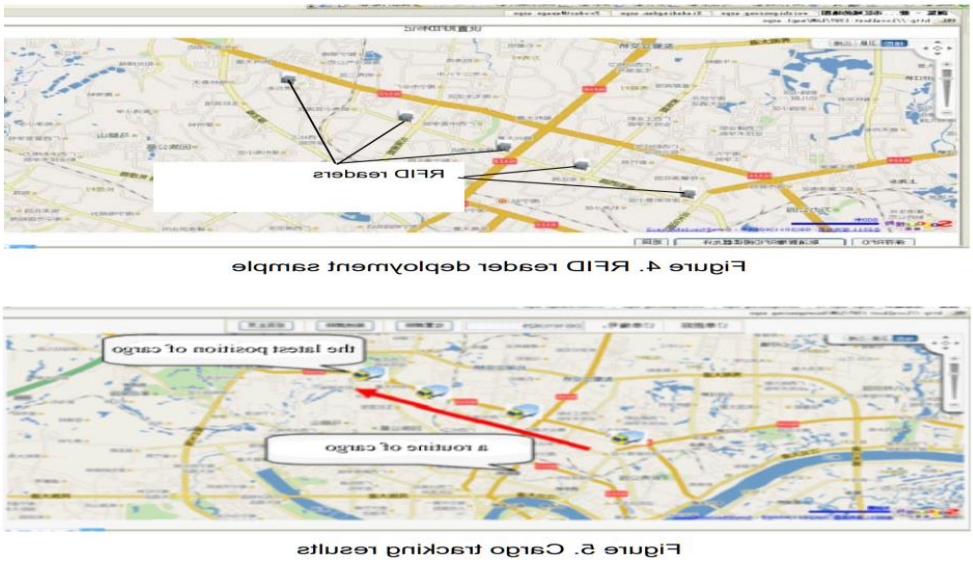


Figure 4 Google maps

2.6 RFID TRACKING SYSTEM FOR VEHICLES (RTSV)

RFID tracking system for vehicles (RTSV) is a system that uses radio frequency identification (RFID) technology to track the movement of vehicles. It is typically used to track the location of vehicles in a fleet, or to track the movement of vehicles through a toll road or other restricted area. The system consists of a network of RFID readers, which are installed at strategic locations along the route that the vehicles will be travelling. The readers are connected to a central database, which stores the information collected by the readers. When a vehicle passes one of the readers, the reader will capture the vehicle's identification information, as well as the time and location of the reading. This information is then transmitted to the central database, where it is stored. The system can be used to track the location of individual vehicles, or to track the movement of vehicles in a fleet. The system can also be used to monitor the performance of a fleet, or to track the usage of a toll road or other restricted area.

RFID Tracking System for Vehicles (RTSV) (researchgate.net) [5]

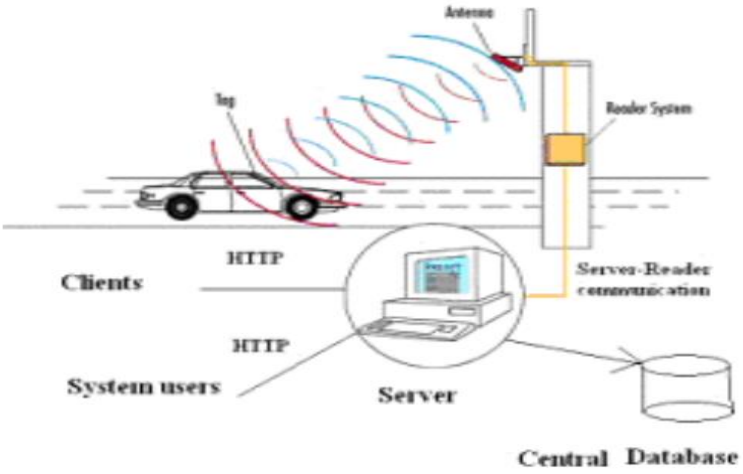


Figure 5 How the system works

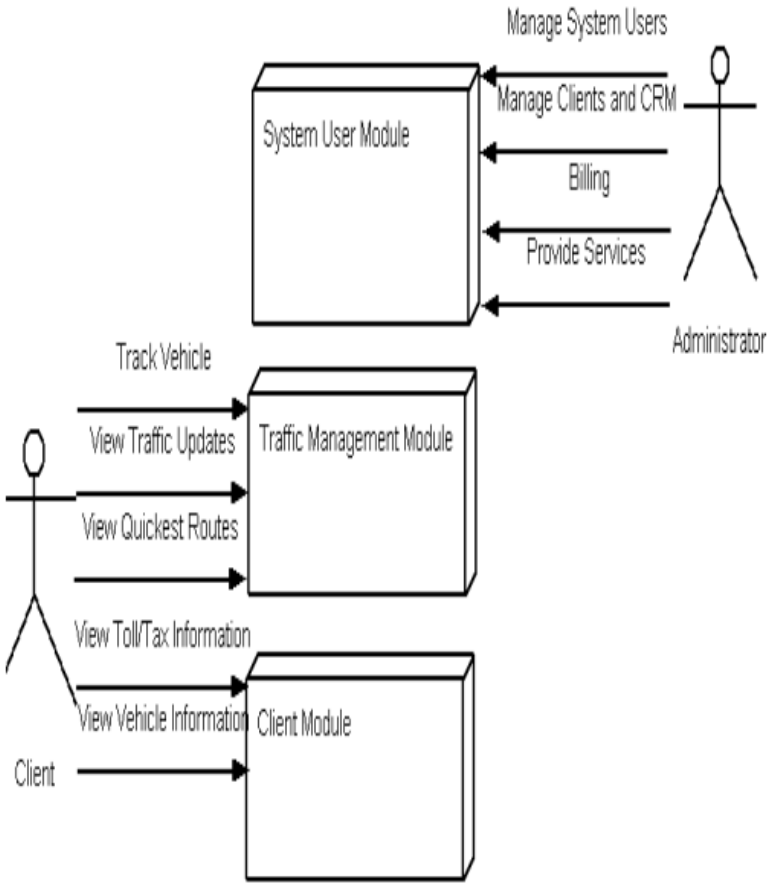


Figure 6 Component Diagram

2.7 RFID-BASED VEHICLE MONITORING SYSTEM

This research paper includes the literary work on monitoring inside a boundary with the rfid entering system. The paper presents a system for monitoring vehicles using radio frequency identification (RFID) tags. The system consists of an RFID reader, a microcontroller, and a GPS module. The RFID reader is used to read the RFID tags attached to the vehicles. The microcontroller is used to process the data from the RFID reader and the GPS module. The system is able to track the location of the vehicles and to provide information about the vehicles' status.

Paper Title (use style: paper title) (researchgate.net)

2.8 RFID INTRODUCTION

This paper describes how rfid works and the deployment issues that are being faced by the programmer at the time of implementation. Understanding of Radio frequency has been a great source when we are working on rfid implantation solutions.

untitled (roywant.com)

2.9 ANTI-THEFT SYSTEM

This paper is about the vehicle anti-theft location which is kind of similar to our project but uses different methodology of GPS and GSM integrated with android mobile and pyroelectric infrared sensor. The idea of this is to lock via rfid and if intruder still be managed to enter the vehicle, then pyroelectric sensor works and be able to identify by the heat produced by the person and pins the location of the vehicle with the help of android and GPS. This will send latitude and longitude information to the owner via SMS

untitled (xilirprojects.com)

2.10 ANTI-THEFT SYSTEM 2

In Bangladesh, Vehicle theft is a major problem. In this paper they propose a Smart Anti-theft vehicle tracking system. The entire system is used in two phases. In the first phase, they used a device that is embedded on a vehicle. The device collects data from the sensors and sends it to the cloud. The second phase uses the data to determine the location of the vehicle and uses GPS (Global Positioning System) to track the location of the vehicle. This system is based on Internet of things (IoT), which includes sensors, cloud, and GPS. The system is tested on the Android platform with a range of sensors. It is observed that the accuracy of the system is 100% in the case of vehicle theft.

https://www.researchgate.net/profile/Maruf-Islam/publication/322516489_Smart_anti-theft_vehicle_tracking_system_for_Bangladesh_based_on_Internet_of_Things/links/5c93134145851506d71f7728/Smart-anti-theft-vehicle-tracking-system-for-Bangladesh-based-on-Internet-of-Things.pdf). [6]

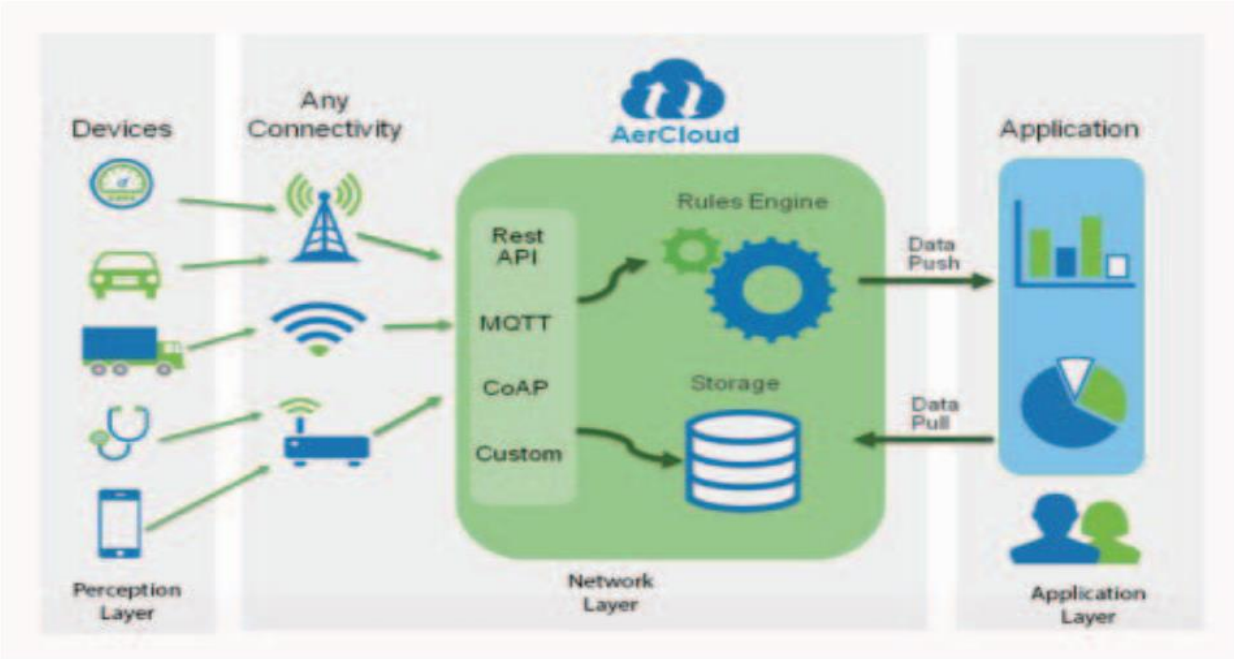


Figure 7 Anti thief Diagram

CHAPTER 3
AIM AND STATEMENT OF THE PROBLEM

3.1 EXPLAINATION OF PROJECT

The vehicle tracking system with RFID uses radio frequency identification (RFID) technology to track the location of vehicles. RFID tags are attached to each vehicle, and these tags emit radio signals that can be detected by RFID readers. The readers then relay the information to a central system, which can track the location of the vehicles in real-time. This system can be used to monitor the movement of vehicles in a fleet, or to track the location of individual vehicles.

3.2 DEFINATION OF THE PROBLEM

Now a days crime is increasing day by day similarly the vehicle are stolen rapidly and can't be recovered easily. There must be proper autonomous system which can track these activities and inform to the concern authorities.by today almost 169 cars has been stolen and not been recovered yet this report has been published by CPLC (citizen polices Liasion committee) these stats clearly showing us the deficiency of management of these vehicles.

3.3 LIMITATION OF SCOPE

The main limitation of the vehicle tracking system with RFID is the range of the RFID tags. The tags need to be within a certain range of the reader in order to be read, so if a vehicle goes outside of that range it will not be tracked. Another limitation is that RFID tags can be blocked by metal or other materials, so if a vehicle is parked in a garage or near a metal structure, the tag may not be readable. And RFID tags can be easily removed or tampered with, which could allow vehicles to avoid being tracked.

3.4 SOLUTION OF THE PROBLEM

1. The vehicle tracking system with RFID uses a tag or transponder that is attached to the vehicle.
2. The tag contains information about the vehicle, such as the make, model, and license plate number.
3. The RFID reader scans the tag and decodes the information.
4. The tracking system then uses the information to track the vehicle's location.

3.5 LACK OF TIME

One of the main challenges that the team may face is a lack of time to properly test and implement the system. This could lead to potential issues with the system's accuracy and reliability. If the team does not have enough time to work on the project, the quality of the RFID Vehicle detection system may be compromised. The team may not be able to finish developing and testing the system before the deadline. If the RFID Vehicle detection system project is not given enough time to be completed, it is likely that the system will not work as intended. The system may have errors, or it may not be able to properly detect vehicles. It is likely that it will not be completed correctly or to the fullest extent. This can cause major problems and setbacks for the company or organization that is implementing the system.

3.6 SCOPE CREEP

If the RFID vehicle detection system is changed, it may affect the project in a number of ways. For example, if the system is changed to a different type of RFID system, it may be incompatible with the existing hardware and software, and may require new hardware and software to be installed. Additionally, the change may require new training for personnel, and may cause disruptions to the project schedule. The RFID vehicle detection system may affect the budget and schedule for the project. The RFID vehicle detection system is used to track vehicles as they move through the detection zone. If the system is changed, it could affect the accuracy of the vehicle tracking and the overall performance of the project. Also the cost of the project will increase because the new system will be more expensive. The schedule for the project will be delayed because the new system will take longer to install. And the scope of the project will change because the new system will have different capabilities than the old system.

CHAPTER 4
HARDWARE, SOFTWARE ANALYSIS & REQUIREMENTS

4.1 FACT FINDING

The goal of this project is to detect vehicles with the help of RFID. The first step is to find an RFID tag that can be used to detect vehicles. The second step is to find an RFID reader that can be used to read the tag. The third step is to find an antenna that can be used to read the tag. The fourth step is to find a power source that can be used to power the tag and the reader. The fifth step is to find a way to connect the tag and the reader to the power source. The sixth step is to find a way to connect the tag and the reader to the antenna. The seventh step is to find a way to connect the tag and the reader to the vehicle. The eighth step is to find a way to connect the tag and the reader to the ground. The ninth step is to find a way to connect the tag and the reader to the computer. The tenth step is to find a way to connect the tag and the reader to the Internet.

The purpose of this fact finding is to determine the feasibility of using an RFID system for vehicle detection. RFID systems are already in use for a variety of applications, but their use for vehicle detection is relatively new. This fact finding will examine the potential benefits and drawbacks of using an RFID system for vehicle detection.

4.1.1 BENEFITS

1. RFID systems can provide real-time tracking of vehicles. This information can be used to improve traffic flow and reduce congestion.
2. RFID systems can be used to identify stolen vehicles.
3. RFID systems can be used to track the location of vehicles in case of an emergency.
4. RFID systems can be used to monitor the speed and movement of vehicles. This information can be used to improve safety on the roads.
5. RFID systems can be used to automatically toll vehicles. This can reduce the need for human toll collectors and improve efficiency.

4.1.2 DRAWBACKS

1. RFID systems require the installation of infrastructure, such as antennas and readers. This can be costly and time-consuming.
2. RFID systems can be susceptible to interference from other electronic devices.
3. RFID systems can be hacked

4.2 HARDWARE

The main hardware components used in a Vehicle Detection system with RFID would include an RFID reader, an antenna, and a tag. Additionally, you might also need a computer or other device to store and process the data collected by the RFID reader.

RFID

MFRC522 RC522 RFID READER

TAG

ESP32

GPS MODULE NEO6M V2 U BLOX

4.2.1 RFID

RFID is a short-range wireless communications technology that uses electromagnetic fields to identify and track objects. RFID technology can be used in a variety of applications, including access control, inventory management, asset tracking, and vehicle detection.

RFID technology can be used to detect vehicles in a variety of ways. For example, RFID tags can be affixed to the outside of vehicles, or RFID readers can be installed in strategic locations to detect the presence of vehicles. RFID technology can also be used to track the movement of vehicles in real-time, allowing for better traffic management and route planning. [1]

Figure 8 RFID

4.2.1.1 HOW RFID WORKS

Radio-frequency identification (RFID) is a technology that uses radio waves to identify and track objects. RFID tags are small devices that contain a chip and antenna that emit a signal that can be read by an RFID reader.

RFID tags are attached to objects such as vehicles, and the signal is used to identify and track the object. RFID readers can be handheld, mobile, or fixed, and they can be used to track objects in real time. [7]

4.2.1.2 ACTIVE RFID

Active RFID tag is an active transponder with a built-in antenna that can be read at long ranges. Active RFID tags use battery power to transmit a signal to a reader. The battery life of an active tag is usually 5 to 10 years. Active RFID tags are used in applications where long read ranges are required, such as in warehouse management and tracking high-value assets.

RFID is an automated identification technology that uses radio waves to identify people or objects. RFID technology can be used for a variety of purposes, including access control, inventory management, and asset tracking.

4.2.1.3 PASSIVE RFID

Passive RFID technology is used in a system where the reader does not require a power source. Instead, the tag is powered by the electromagnetic field generated by the reader. The power for the tag is typically supplied by a battery, which is used to store the data. The tag does not have its own power source, so it cannot be used to track the movements of an object.

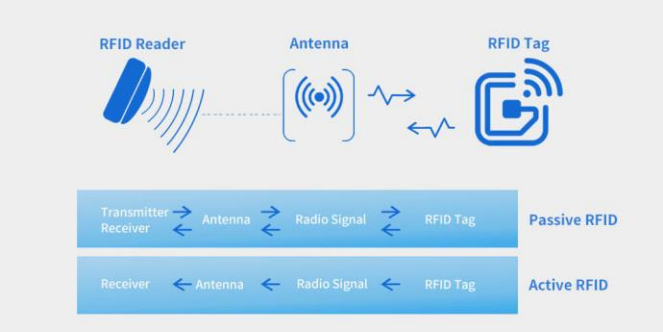


Figure 9 Passive RFID

4.2.2 MFRC522 RC522 RFID READER

RFID (Radio Frequency Identification) is a technology that allows us to identify certain objects without physical contact. The way it works is very simple. There is a tag or card which contains a chip with a unique serial number. This serial number is read with a reader, which is connected to a computer.

The use of RFID is not new. This technology has been used for many years in the field of logistics and transportation, where it has been used to track and trace products and goods. However, the use of RFID is not limited to logistics and transportation. It can be used in many other fields, such as healthcare, retail, security, and so on.

The main advantages of RFID are its speed and accuracy. With RFID, you can track and trace objects in a matter of seconds, which is not possible with other technologies such as barcodes. In addition, RFID is not affected by dirt, dust, or water, which means that it can be used in harsh environments.

The main disadvantage of RFID is its cost. RFID tags and readers are not cheap, and the infrastructure required to use RFID can be costly. In addition, RFID is a complex technology, and it requires trained personnel [8].

Figure 10 *MFRC522 RC522 RFID READER*

4.2.3 TAG

RFID tag is a kind of tag which uses RFID technology to facilitate the identification of objects. RFID tag comprises an antenna, a semiconductor chip and a substrate. The antenna is electrically connected to the RFID chip and is used to receive the signal transmitted by the reader, and then transmit the signal to the RFID chip; the RFID chip comprises a memory space, and the control unit includes an address space and a command space; the data in the memory space can be modified through the command space, and the address space is used to store the address of the command space; the substrate is used to fix the antenna and the RFID chip on the object to be identified.

RFID tag is a non-contact automatic identification technology. RFID tag does not need to be in visual line of sight, which can be identified at any time and any place. It is mainly based on radio frequency signal to automatically identify target and read and write data. It is mainly used in identification and tracking of products, people, vehicles, and animals in various fields. It is an important part of radio frequency identification technology.

RFID tag has three major components: antenna, chip and substrate. Antenna is mainly used to receive and transmit electromagnetic [8].

Figure 11 RFID Tag

4.2.4 ESP32

The ESP32 is a powerful, low cost, Wi-Fi, and Bluetooth system on a chip. It is a single chip solution for low power consumption wireless communication. It is designed for high performance applications such as Internet of Things (IoT), machine-to-machine (M2M), and wearable electronics. The ESP32 is an advanced, low power, wireless SoC with integrated dual-mode Bluetooth and Wi-Fi. It supports a range of peripherals, including UART, SPI, I2C, PWM, ADC, digital microphone, and touch sensor. It is an excellent choice for embedded applications that require robust networking and low power consumption. The ESP32 is ideal for applications that require high performance and low power, such as home automation, security systems, and IoT devices.

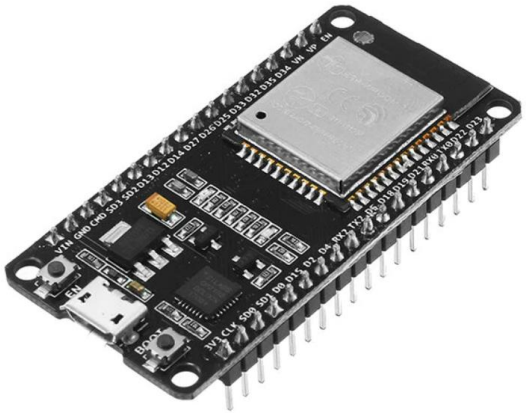


Figure 12 ESP32

4.2.5 GPS MODULE NEO6M V2 U BLOX

The NEO-6M GPS module is a highly sensitive and low power GPS receiver. It is based on the U-BLOX NEO-6M GPS module and has the same performance and features. The module is suitable for a wide range of applications including navigation, tracking, timing, surveying, and much more. It has an item size of 25*25mm and a very low power consumption of only 6mA. It also has an integrated antenna and a high level of accuracy with up to 10 meters. It is designed to be used in a wide range of applications and is compatible with most microcontrollers. It is designed to provide a reliable, accurate, and cost-effective solution for GPS applications.

Figure 13 GPS MODULE NEO6M V2 U BLOX

4.3 HARDWARE COST

In this section we will discuss the main and important points related to our project and compare the equipment, discuss detail further to explain why we use this kind of product in our project.

4.3.2 WORKING ASSUMPTION

How the system works integrated we use the Raspberry Pi can be used as a GPS tracker by connecting it to a GPS module. The GPS module will then be able to track the location of the Raspberry Pi. To send the location of the Raspberry Pi, you will need to use a communication module such as a cellular modem or Wi-Fi. The location can then be sent over the internet to a tracking service or website. We will also need to install a software package called gpsd, which provides a daemon that communicates with GPS modules and provides GPS data to applications.

Products	Estimated prices
RFID CHIP	50X10=500
RFID READER	350x1050
ESP32	950x3=2850
GPS MODULE NEO6M V2 U BLOX	1000x3=3000
JUMPER WIRES	5x200=1000
MB102 BREADBOARD	160x3=480

Table 1 Cost estimation

4.3.3 RFID VS AI (ARTIFICIAL INTELLIGENCE)

There is no clear winner when it comes to RFID vs Artificial Intelligence advantages and disadvantages. Each technology has its own set of advantages and disadvantages that need to be considered when deciding about which technology to use.

RFID advantages include its ability to track and manage inventory, its low cost, and its durability. RFID disadvantages include its potential for abuse and its reliance online-of-sight technology.

Artificial intelligence advantages include its ability to learn and improve over time, its ability to make decisions based on data, and its ability to automate tasks. Artificial intelligence disadvantages include its lack of common sense and its potential for misuse. And it is much expensive than RFID tags because of its computation and processing of data and equipment like expensive cameras to record high quality video.

4.4 UML DIAGRAMS

4.4.1 OPERATIONAL DIAGRAM

RFID's to be installed in number plate of cars and scanners are installed on various points of the city which then scan these chips. And send the scanned rfid to the database with its location which then be mapped on to google maps. The Rfid database store's location with date and time of the vehicle which it passes. this all information can be accessed by a dashboard. In the end authenticated by excise database.

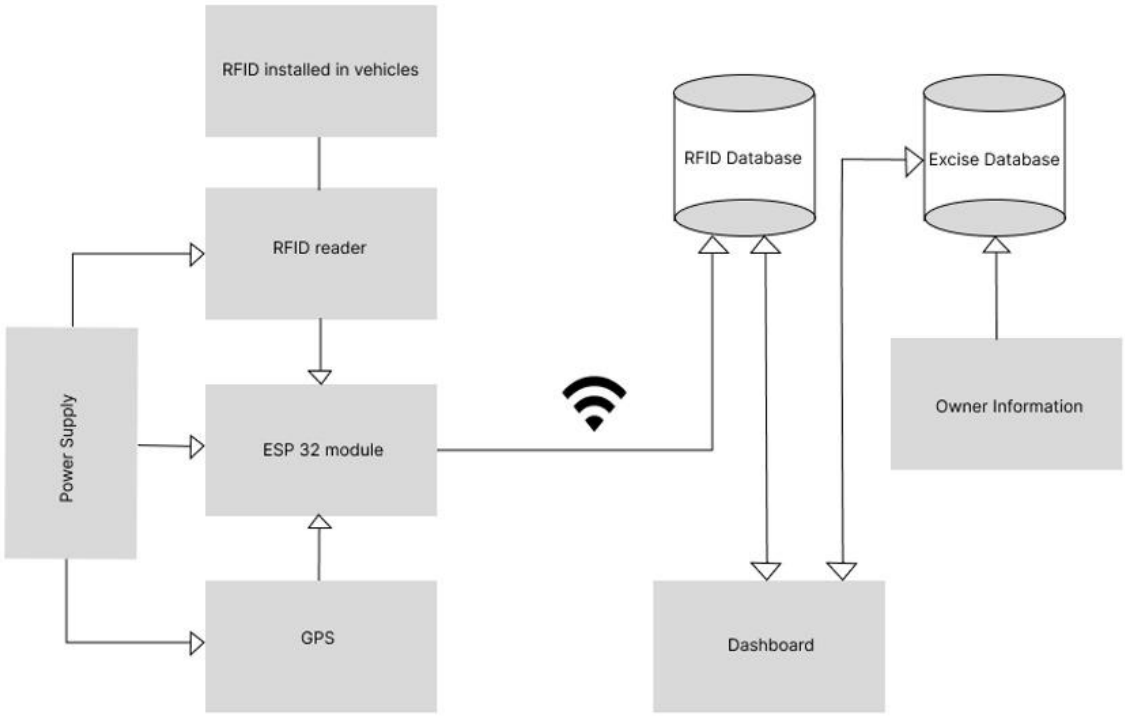


Figure 14 Operational Diagram

4.4.2 ACTIVITY DIAGRAM

The activity diagram for a vehicle tracking system with RFID would show the various activities that need to be performed to track a vehicle. These activities would include reading the RFID tag on the vehicle, determining the location of the vehicle, and then tracking the movement of the vehicle.

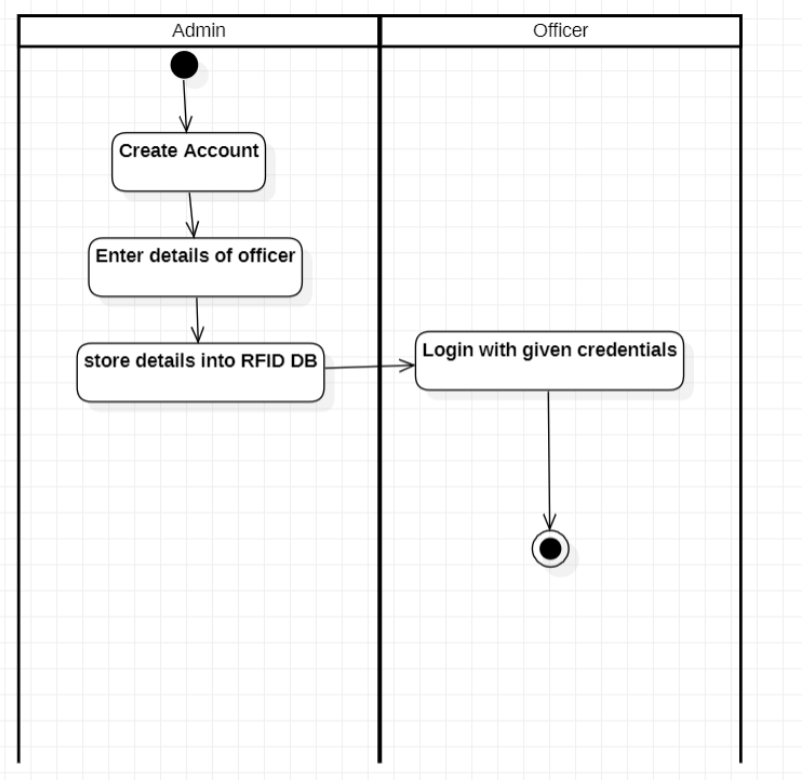


Figure 15 Activity Diagram 4.1

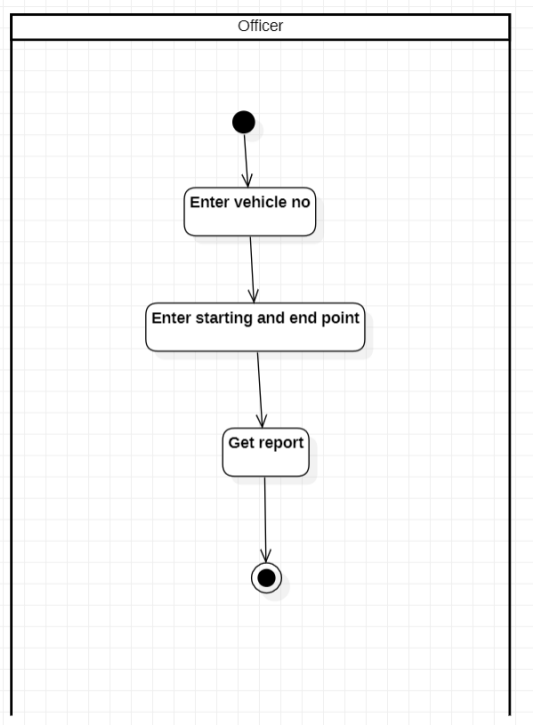


Figure 16 Activity Diagram 4.2

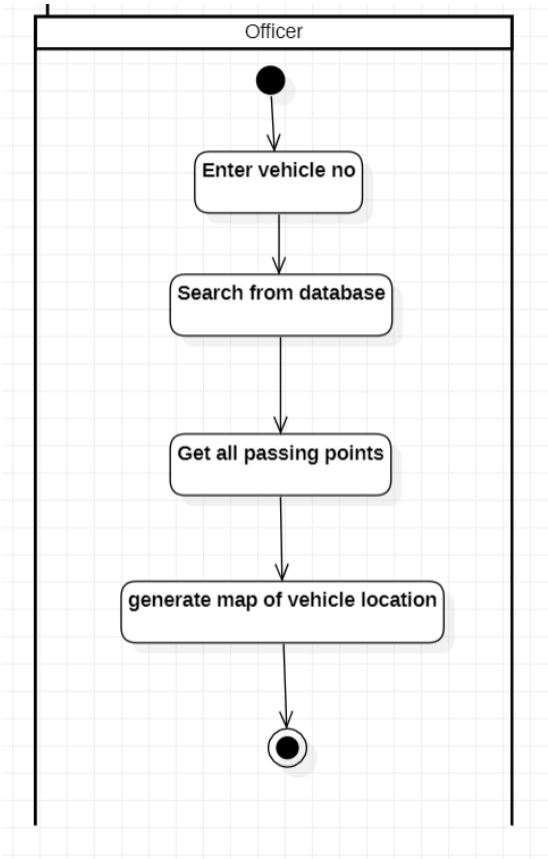


Figure 17 Activity Diagram 4.3

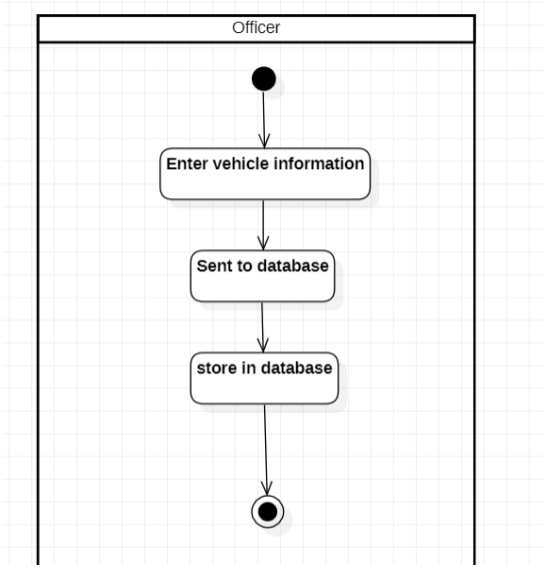


Figure 18 Activity Diagram 4.4

4.4.3 USECASE DIAGRAM

There are two users Admin who administer whole system including search,register,getting location and owner data. The other user is officer who maintain logs of admin and create or remove admins from the existing system.

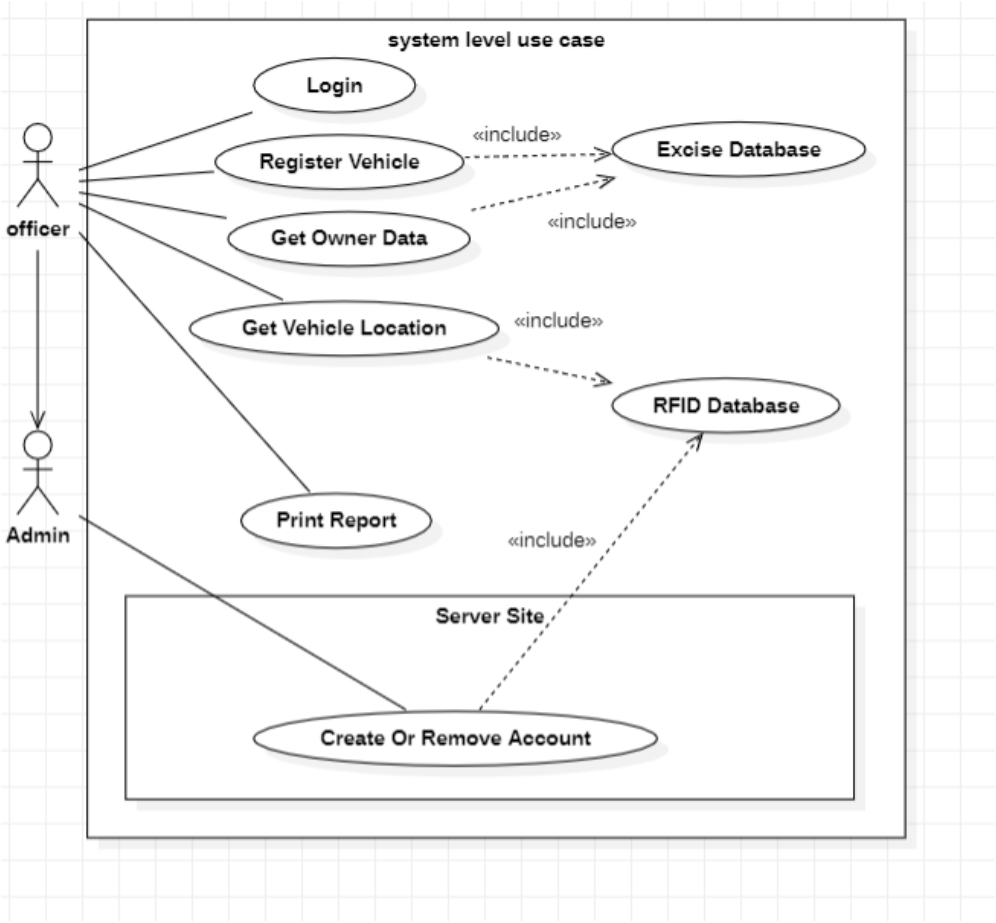


Figure 19 Usecase Diagram 4.1

CHAPTER 5
SOFTWARE DESIG AND MODELING

5.1 ARCHITECTURE

MVC (Model View Controller):

The Model-View-Controller (MVC) architecture is a design pattern that is commonly used to organize the code of an application. It is particularly useful for complex systems such as a vehicle detection system using RFID and GPS, as it allows for the separation of different aspects of the system into distinct components.

The Model is responsible for handling the data and logic of the system. In this case, it would handle the information received from RFID and GPS sensors and would process this data to determine if a vehicle is present. It would also handle any algorithms or logic required for the detection system to function properly.

The View is responsible for displaying the information to the user. In this case, it would handle the presentation of the vehicle detection data on a map or in a list. The view would also handle the user interface elements such as buttons, menus, and other controls that the user interacts with.

The Controller is responsible for handling the flow of information between the Model and the View. It would take the information from the Model and pass it to the View, and would also take user input from the View and pass it to the Model for processing. The controller would also handle any updates to the view based on new information from the Model.

By separating the system into these three components, it becomes easier to understand and maintain the code. Additionally, if one component needs to be updated or changed, it can be done independently of the other components, making the system more flexible and easier to scale.

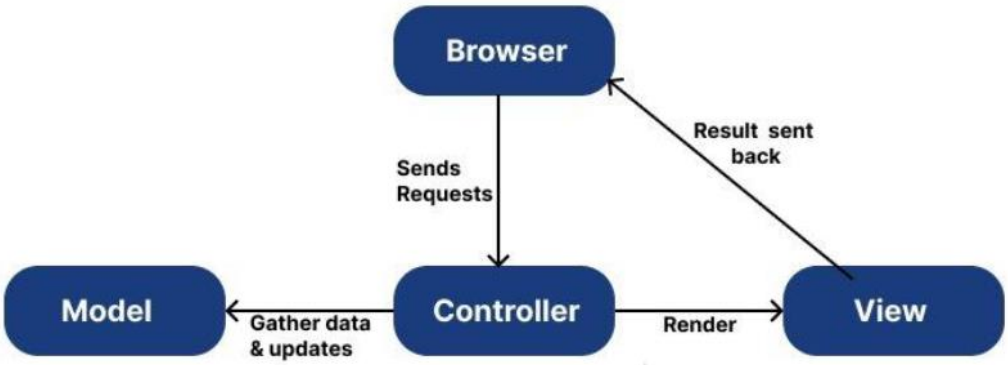


Figure 20 MVC

5.2 DIAGRAMS

5.2.1 USECASE DIAGRAM

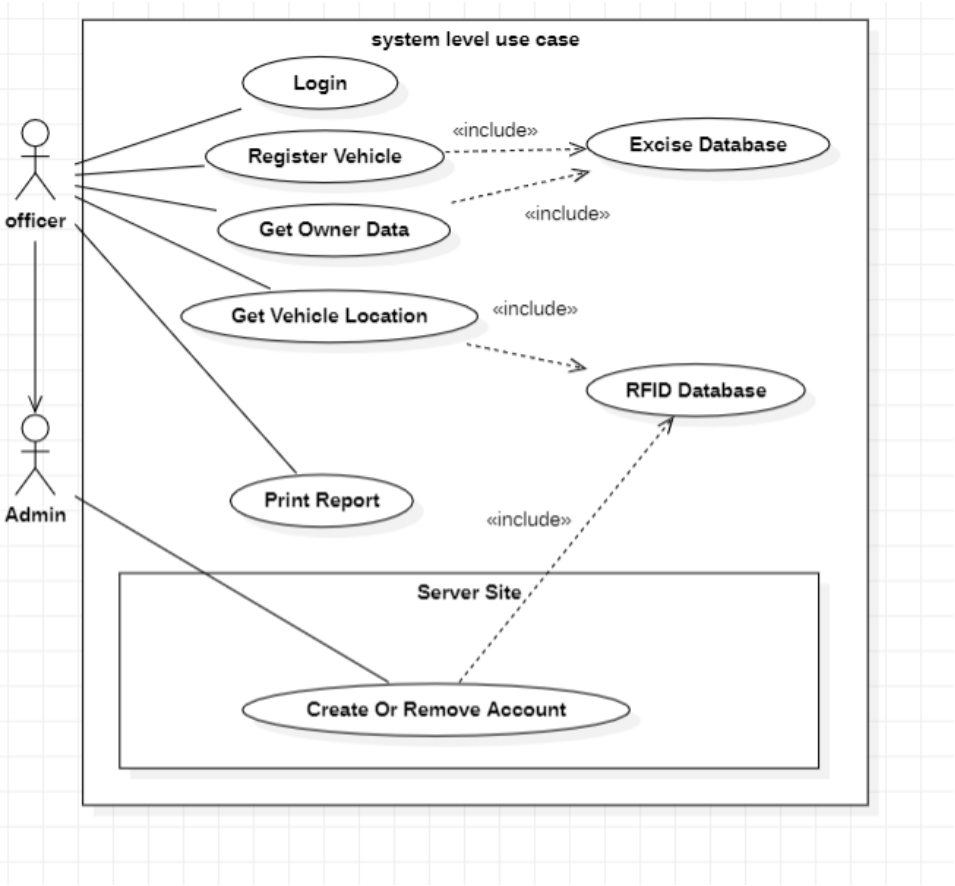


Figure 21 Usecase Diagram 5.1

A use case diagram for a vehicle detection system using RFID would show several actors and use cases. The actors in this system would include:

- Admin: responsible for creating and removing accounts for officers and managing the system's settings.
- Officer: responsible for registering vehicles, tracking vehicles using RFID technology, generating reports, and managing the system's data.

The use cases in this system would include:

- Create/Remove Officer Account: The admin can create or remove an account for an officer.
- Login: The officer can log in to the system using their credentials.
- Register Vehicle: The officer can register a vehicle by inputting the vehicle's registration number and other relevant information.
- Get Owner Data: The officer can retrieve the owner's information by inputting the vehicle's registration number.
- Get Vehicle Location: The officer can track the vehicle's location by using the RFID technology
- Print Report: The officer can generate, and print reports based on the system's data.

All these use cases are connected by the system and can be performed by the officer after successful login and by admin to manage the system.

5.2.2 OBJECT DIAGRAM

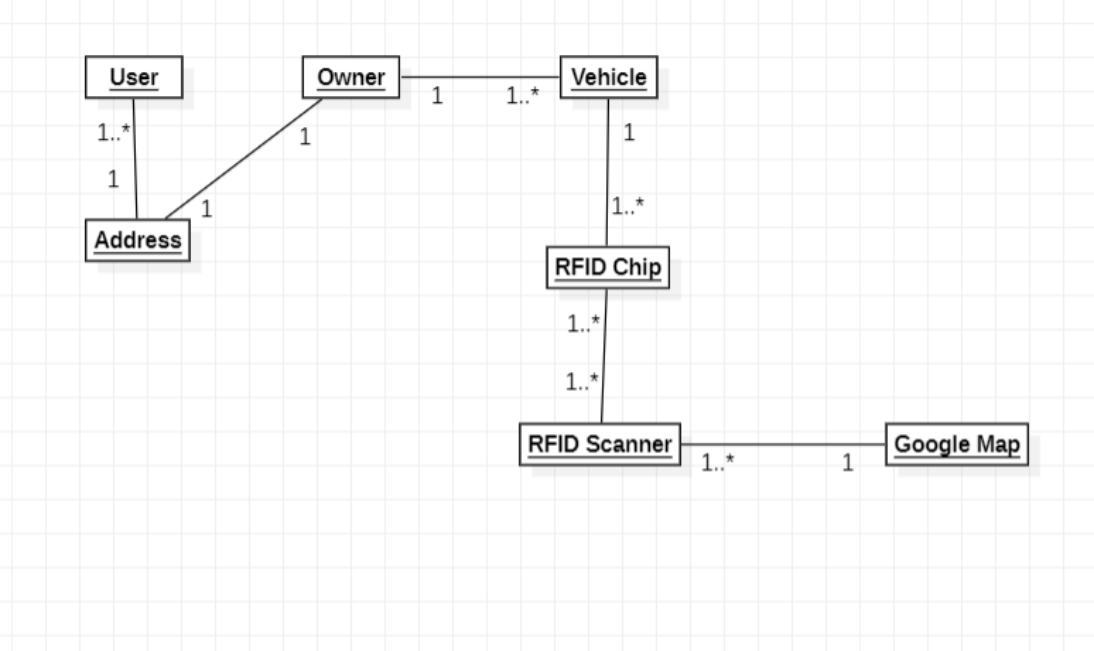


Figure 22 Object Diagram

Object diagrams in the vehicle detection system are used to model the static structure of a system. This diagram is used to show the objects and the relationships between them.

The main objects in the vehicle detection system include User, Owner, Vehicle, RFID Scanner, and software. Each of these objects has its own attributes such as location, performance, type of sensor

The relationships between these objects include the data flow between them. For example, the data from the sensors is sent to the processor, which then sends it to the software.

5.2.3 ACTIVITY DIAGRAM

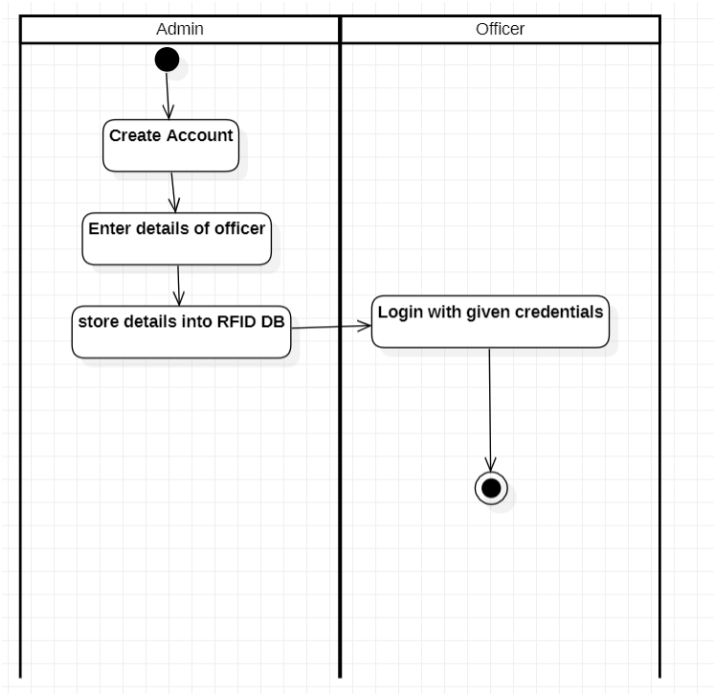


Figure 23 Activity Diagram 5.1

In this activity admin can create an account for an officer and the officer can login the account with the given credentials would involve several steps. The first step would be for the admin to input the officer's information, such as name, username, CNIC number, phone number and password. This information would then be stored in a database.

Once the account is created, the officer would be able to use their username and password to login to the system. The officer would be prompted to enter their credentials, such as username and password. This information would then be verified against the database to ensure that the officer has the correct credentials.

Once the officer is logged in, the officer would have access to the system's functions, such as registering vehicles, generating reports, and tracking vehicles.

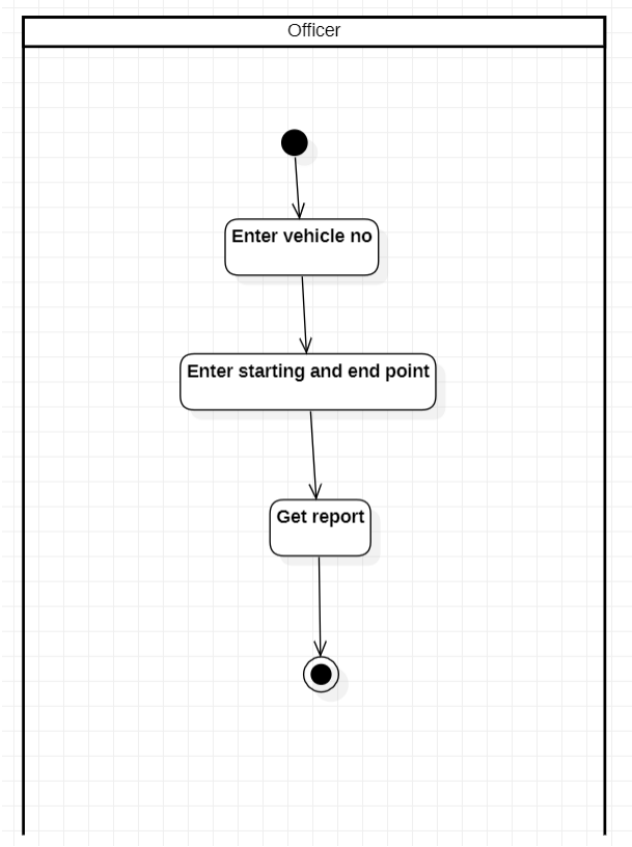


Figure 24 Activity Diagram 5.2

An activity diagram for a vehicle detection system using RFID would involve several steps. The first step would be for an officer to enter the vehicle number. This information would then be verified and matched with a database of registered vehicles.

Once the vehicle has been identified, the officer would enter the starting and end point for the vehicle's journey. This information would also be verified and matched with a database of authorized routes.

Once the starting and end point have been confirmed, the vehicle would be allowed to proceed, and its location would be tracked using RFID technology. This information would be recorded and stored in a database.

Finally, the officer would be able to generate reports based on the collected data, such as a list of vehicles that have passed through a certain point, or the routes taken by a specific vehicle.

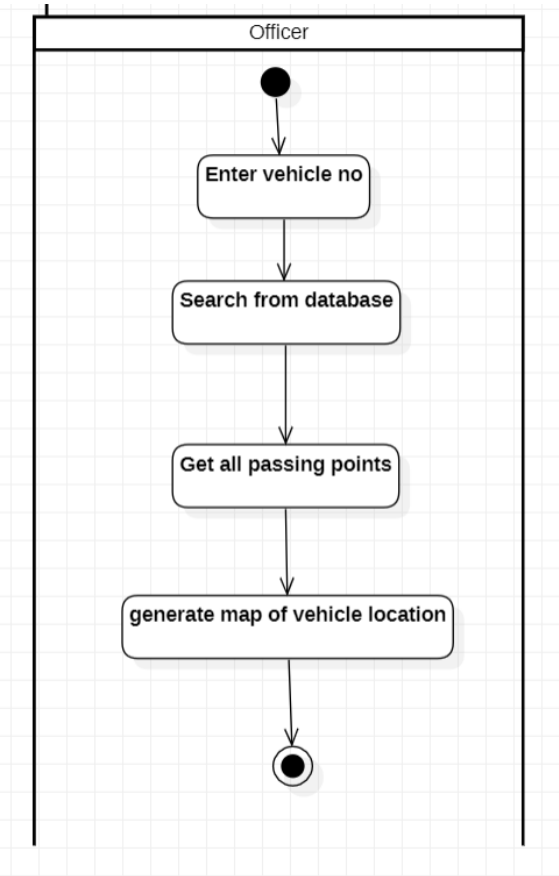


Figure 25 Activity Diagram 5.3

An activity diagram for a vehicle detection system using RFID would involve several steps. The first step would be for an officer to enter the vehicle number using an input device, such as a keyboard or scanner. This information would then be verified and matched with a database of registered vehicles.

Once the vehicle has been identified, the officer would enter the starting and end point for the vehicle's journey. This information would also be verified and matched with a database of authorized routes.

Once the starting and end point have been confirmed, the vehicle would be allowed to proceed, and its location would be tracked using RFID technology. This information would be recorded and stored in a database.

Finally, the officer would be able to generate map based on the collected data, such as a list of vehicles that have passed through a certain point, or the routes taken by a specific vehicle.

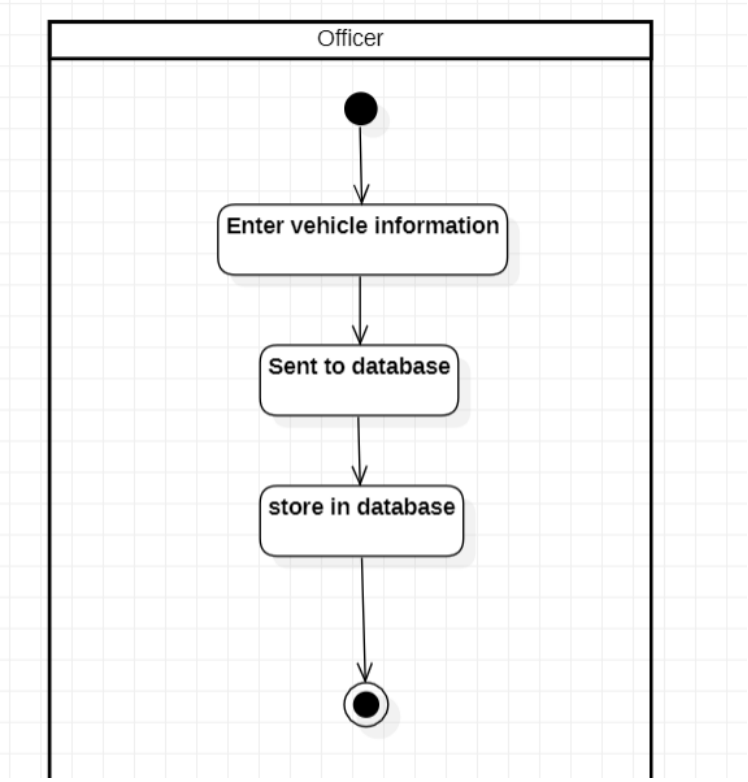


Figure 26 Activity Diagram 5.4

In this activity officer can register vehicles using a vehicle registration number would involve several steps. This information would then be verified and matched with a database of registered vehicles.

If the vehicle is not already registered, the officer would be prompted to enter additional information about the vehicle, such as make, model, and owner's details. This information would also be stored in the database.

Once the registration process is complete, the officer would assign a RFID tag to the vehicle, which would be affixed to the vehicle. The RFID tag would contain unique identification number that corresponds to the vehicle registration number and all the other details.

5.2.4 SEQUENCE DIAGRAM

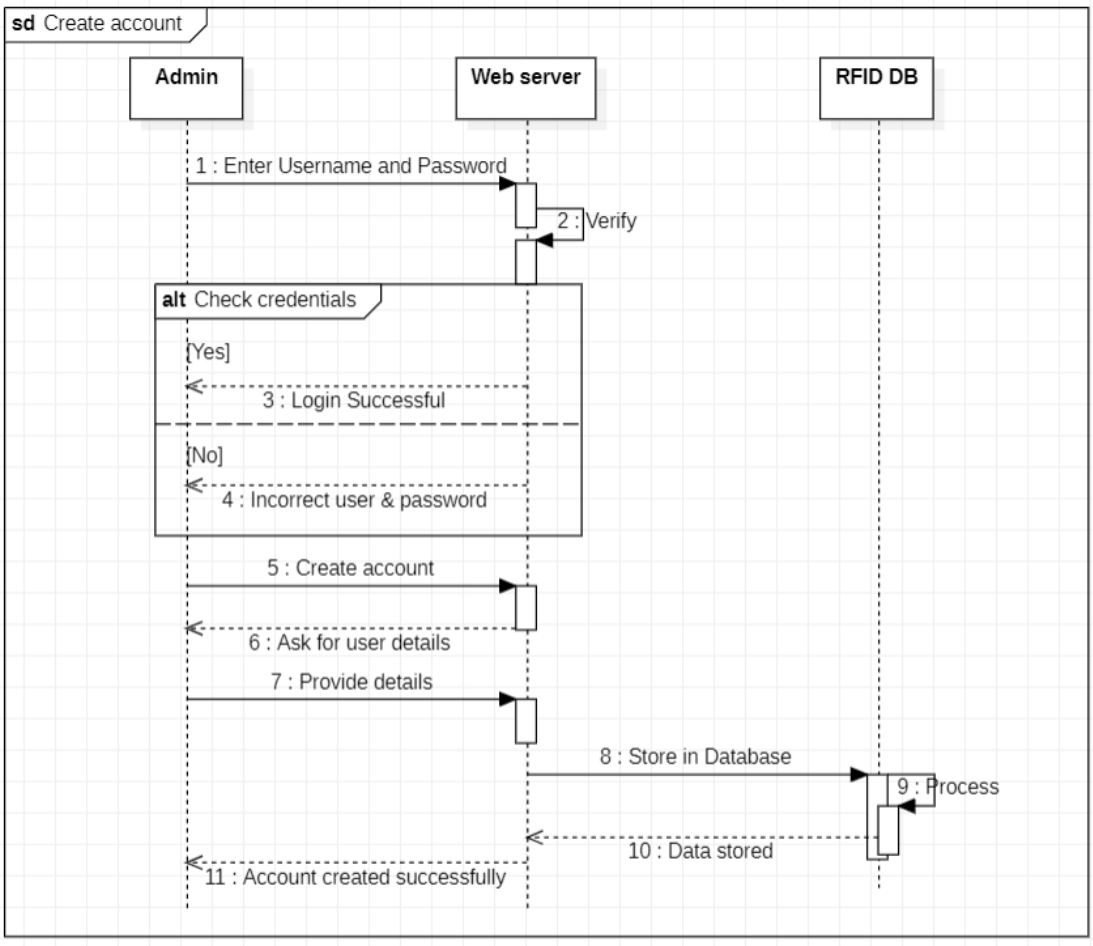


Figure 27 Sequence Diagram 5.1

When new officer will join the department, he/she will request the admin of the system to create his or her account the admin will get his personal information like CNIC number, address, phone number the system will then store his information in the database and create account so that a user can use the system by providing USERID and password. Now whenever the user enters his id password into the system the system will first verify the user and then will give access if valid.

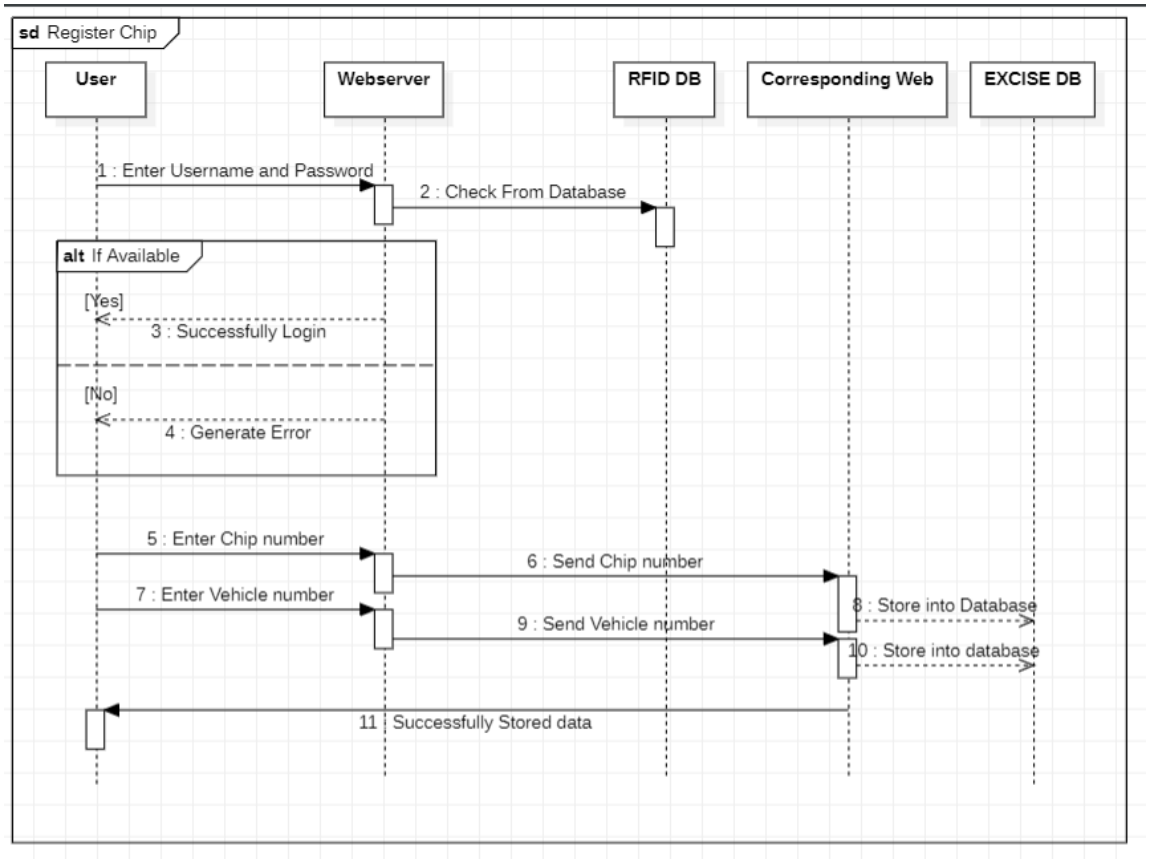


Figure 28 Sequence Diagram 5.2

The officer (User) can register the chip by sending the information of RFID chip and vehicle to the Excise database. The Excise Database will verify the data and registered the vehicle if valid information is provided

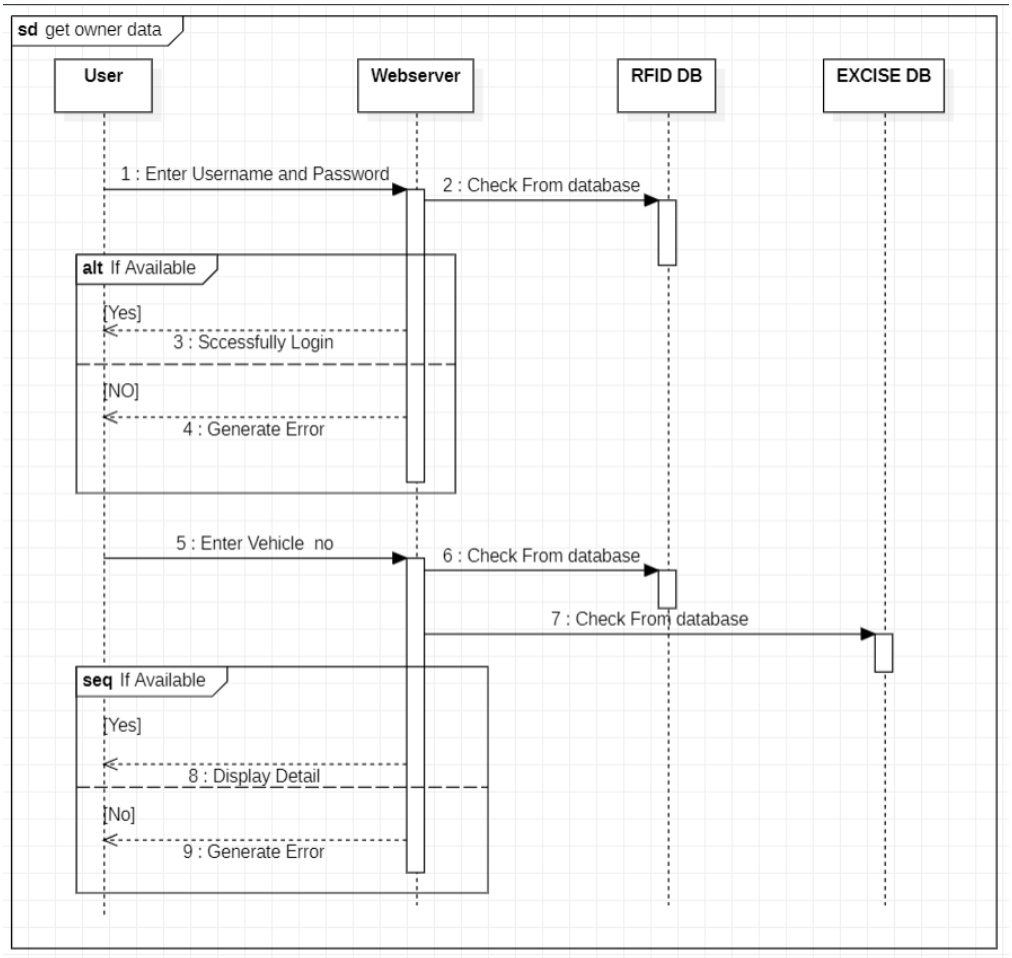


Figure 29 Sequence Diagram 5.3

The officer will login in to the system by entering his username and password the system will first verify the username and password from database then will act accordingly. For getting the owner data the officer will first enter the vehicle number in to the website the web application then fetch data from the RFID as well as EXCISE database if the data will be available then it will be displayed on screen

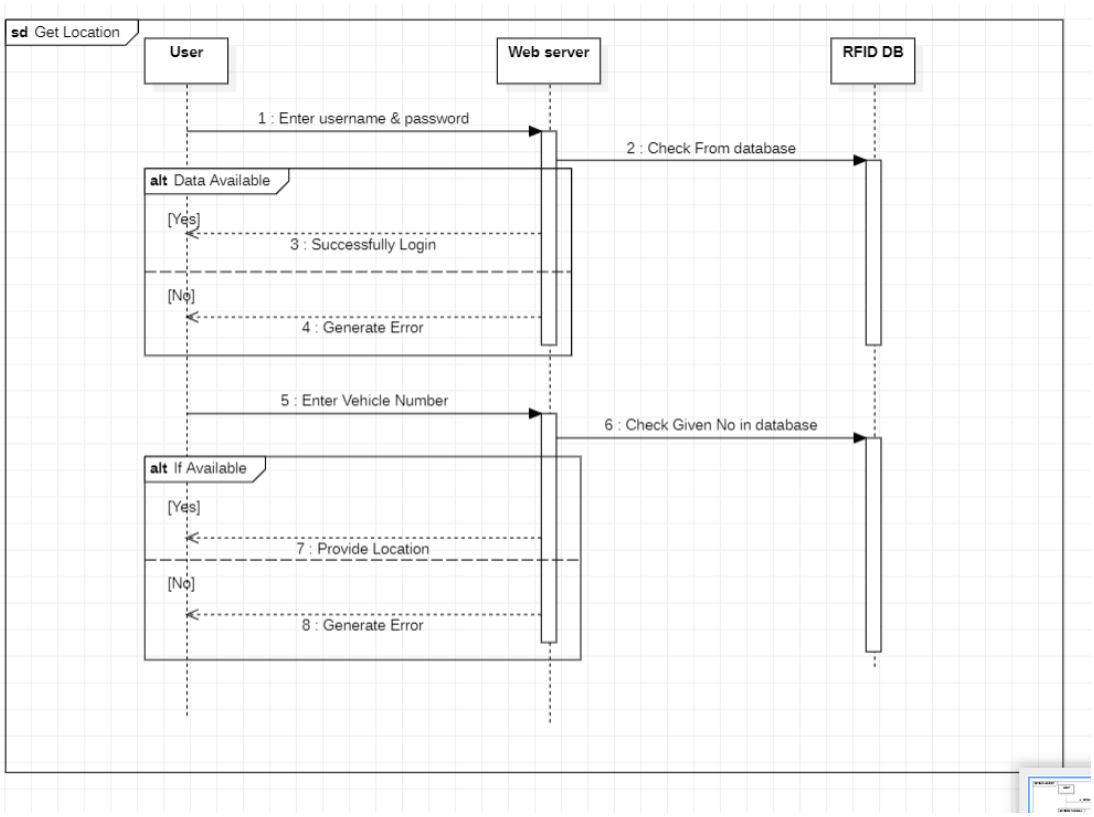


Figure 30 Sequence Diagram 5.4

In the same way the officer (User) can get the location of the vehicle. Whenever the Officer enter the data of the vehicle like Vehicle number & date the system will fetch the Location from the database that we created (RFID DATABASE) where the location is saved by RFID scanner.

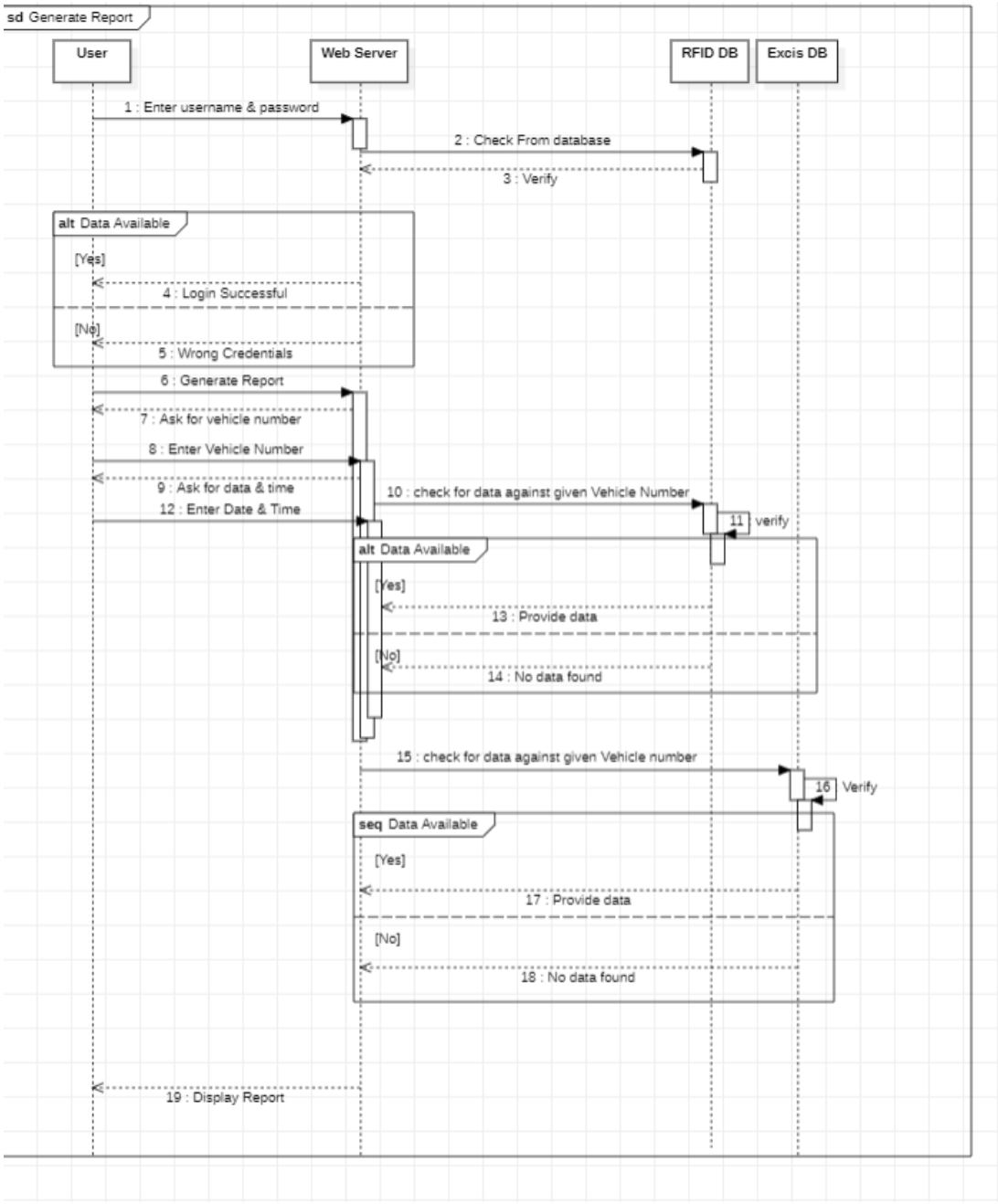


Figure 31 Sequence Diagram 5.5

The officer will first login in to the system for printing the report the user will enter vehicle number and date and time into the web server the web will check the data into the database if available then will provide the information and will display on the screen

5.2.4 CLASS DIAGRAM

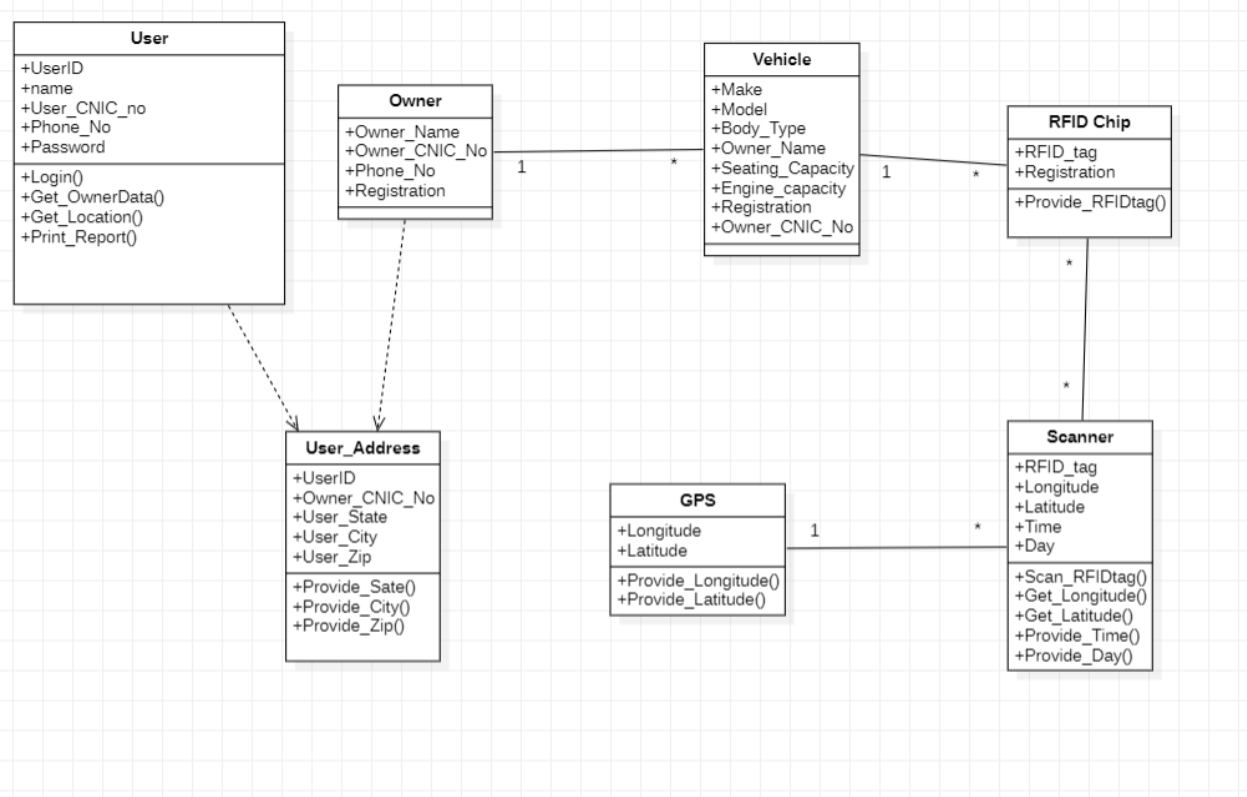


Figure 32 Class Diagram

I this Project Vehicle detection system the main objectives are Owner, User, Vehicle, RFID Chip, GPS, and Scanner

The above diagram shows that the User is able to Login in to the system, He can get the information of the Owner, Also get the location of the car using the system and he can also print the report of the data.

The scanner will scan the RFID Chip attach to the system and will provide the giving information like longitude, Latitude which will be provided by GPS module. Also, date and time.

5.2.5 ER DIAGRAM

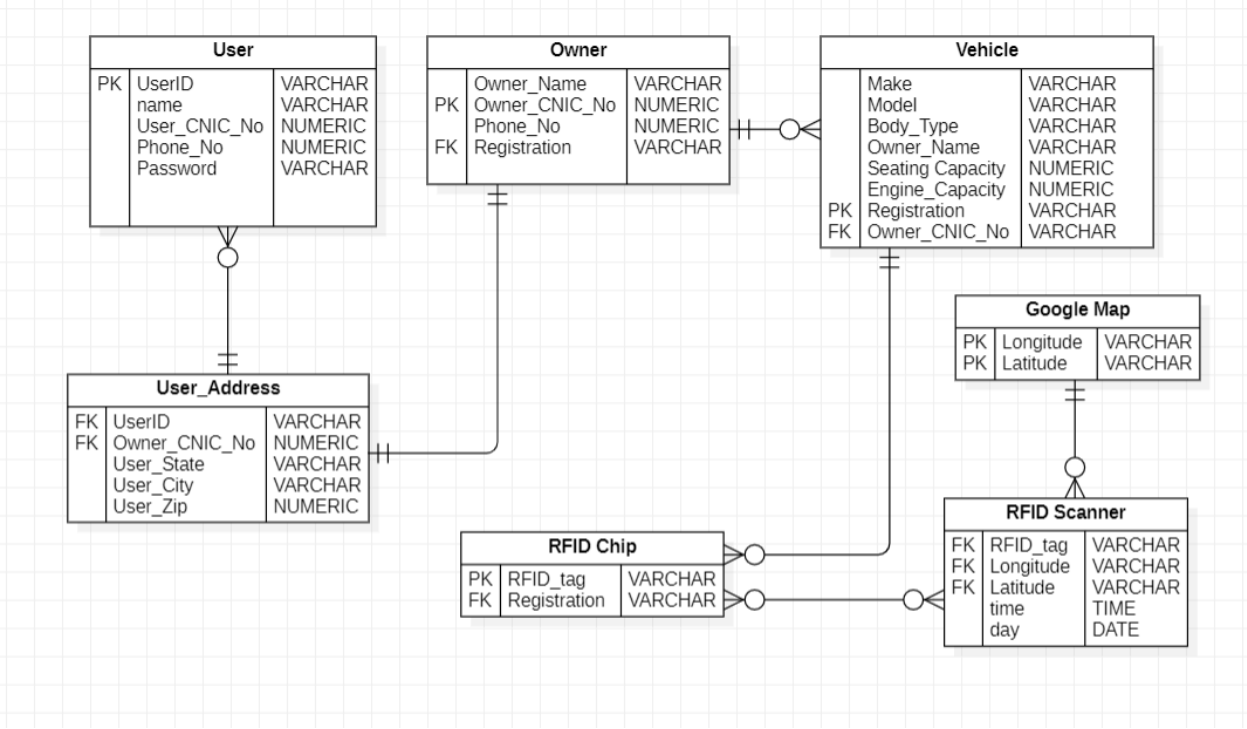
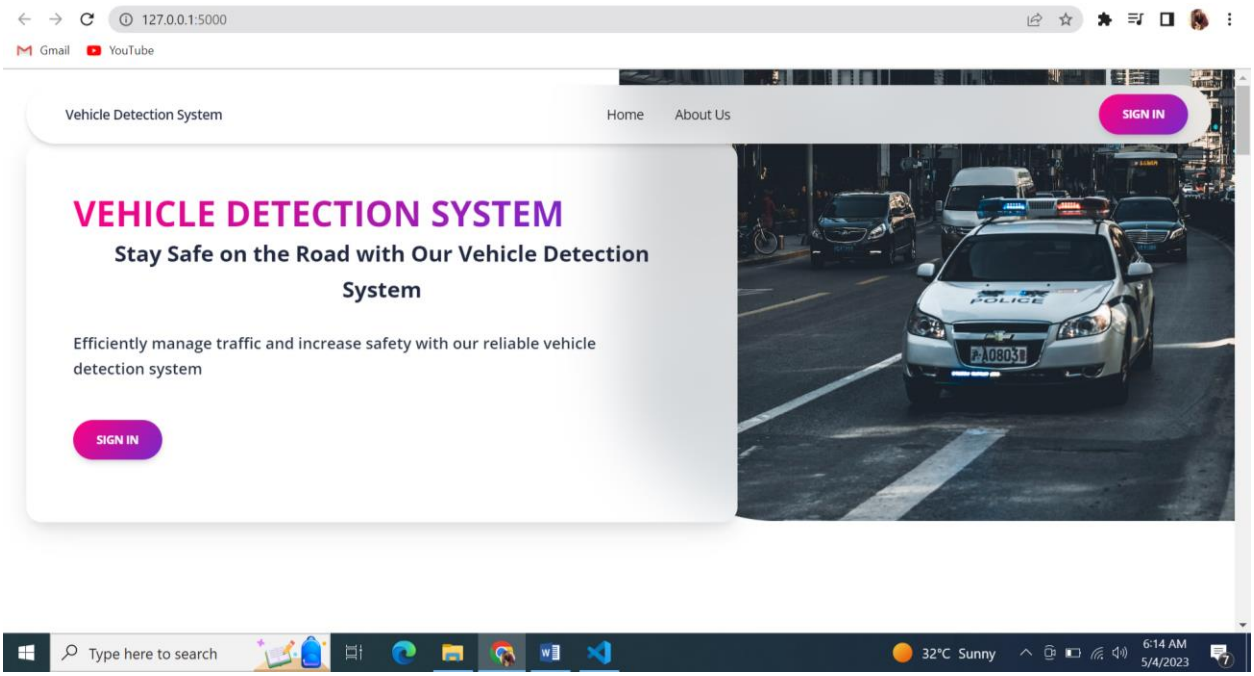


Figure 33 ER Diagram

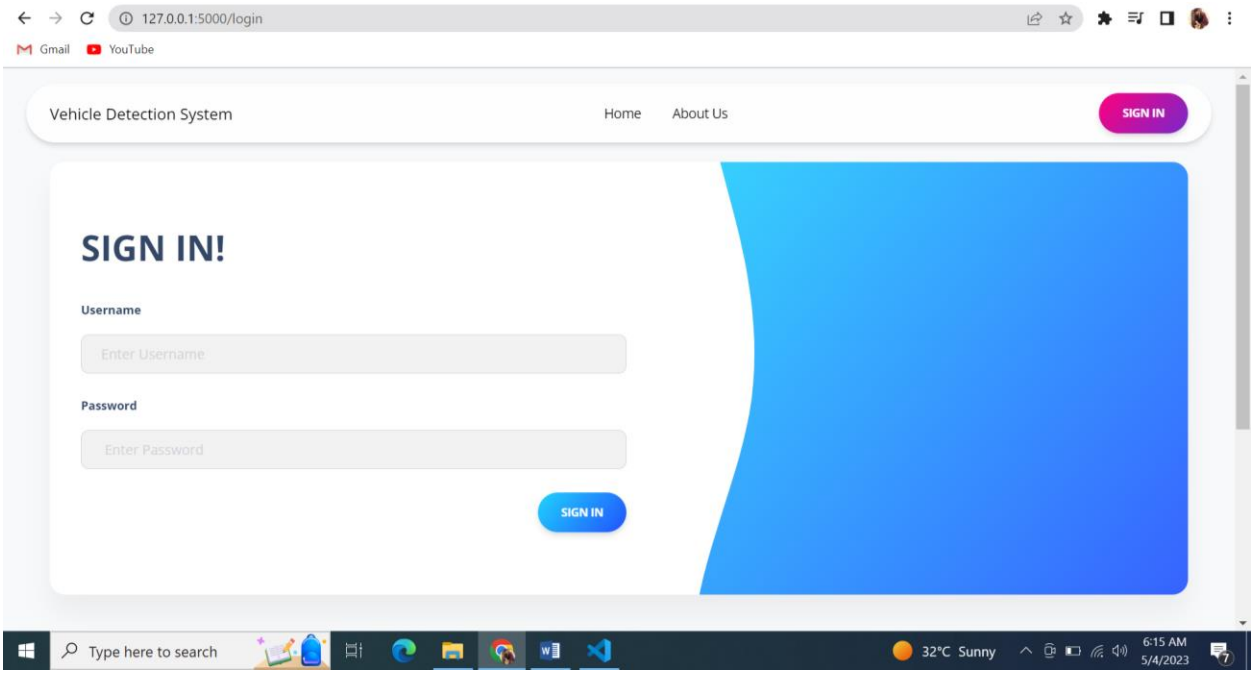
The ER Diagram shows that there will be multiple User using the system. Multiple User can have single address e.g., two brother living in the same house working with the same system. In the same way Multiple vehicles can have single owner and single owner will have single address.

One Vehicle can have multiple chips install in it. E.g., installing RFID chips into number plate as well as install in card dashboard. Multiple scanners can scan multiple RFID chip at the same time. Similarly Multiple Scanner can be installed at the same place means having the same Longitude and Latitude.

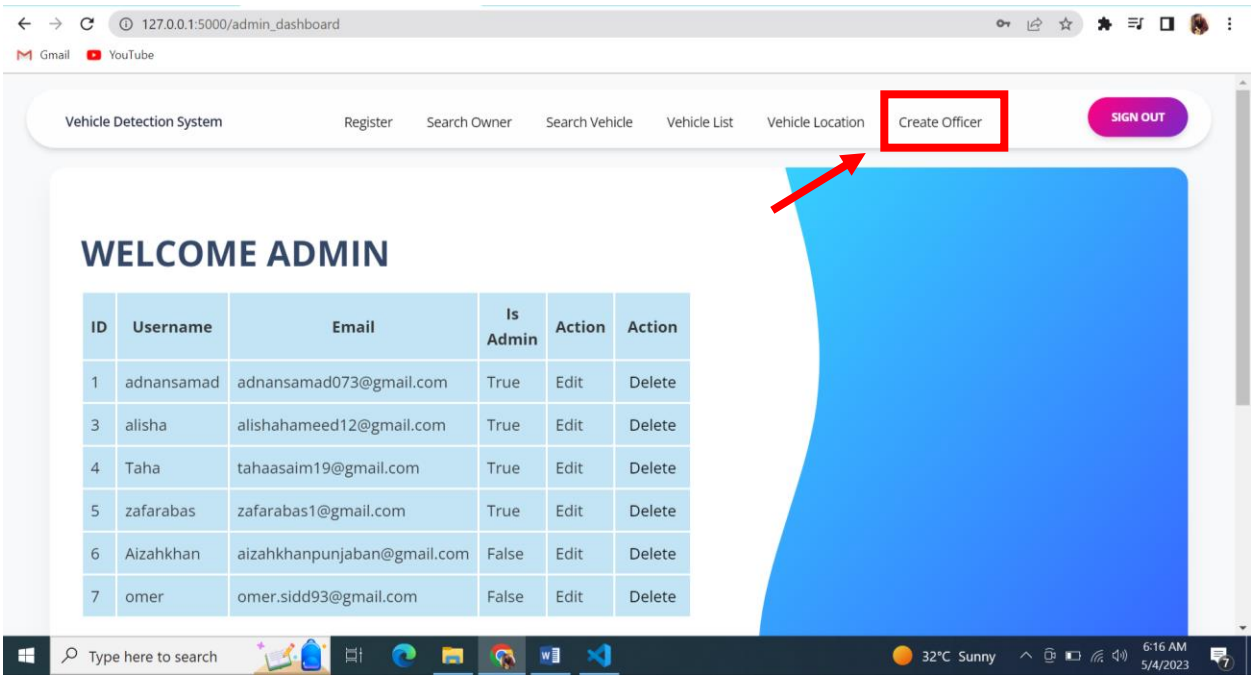
5.3 NON-WORKING PROTOTYPING



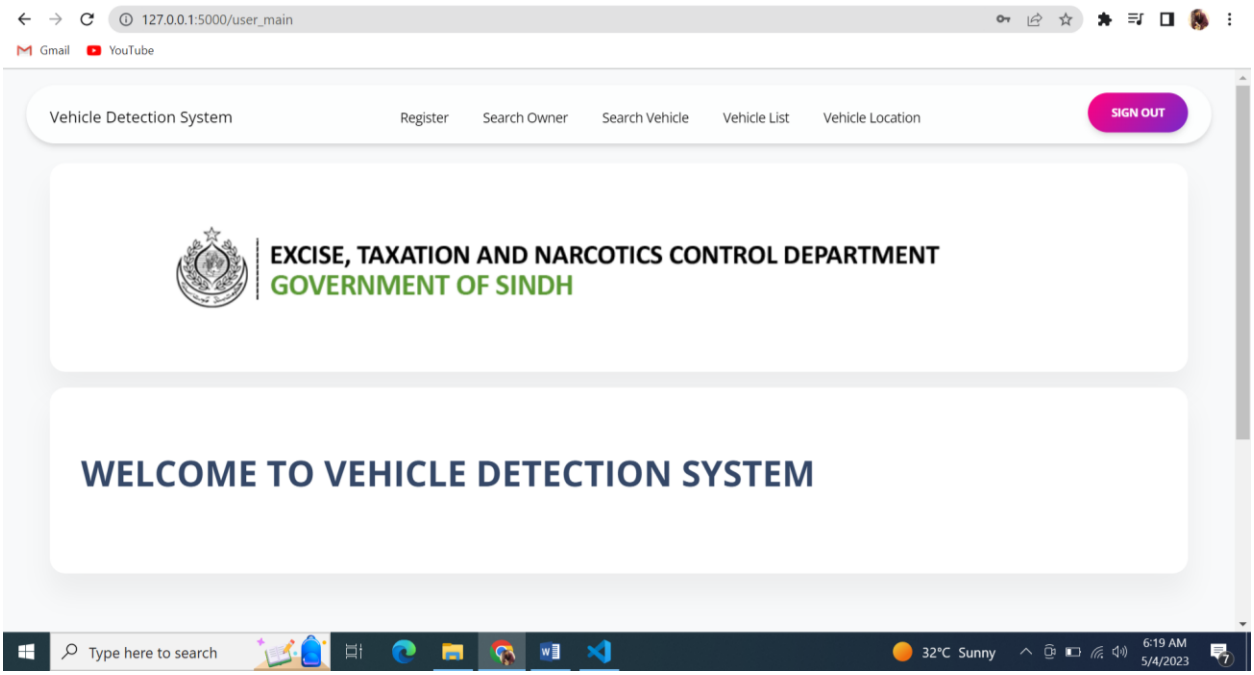
MAIN HOME PAGE



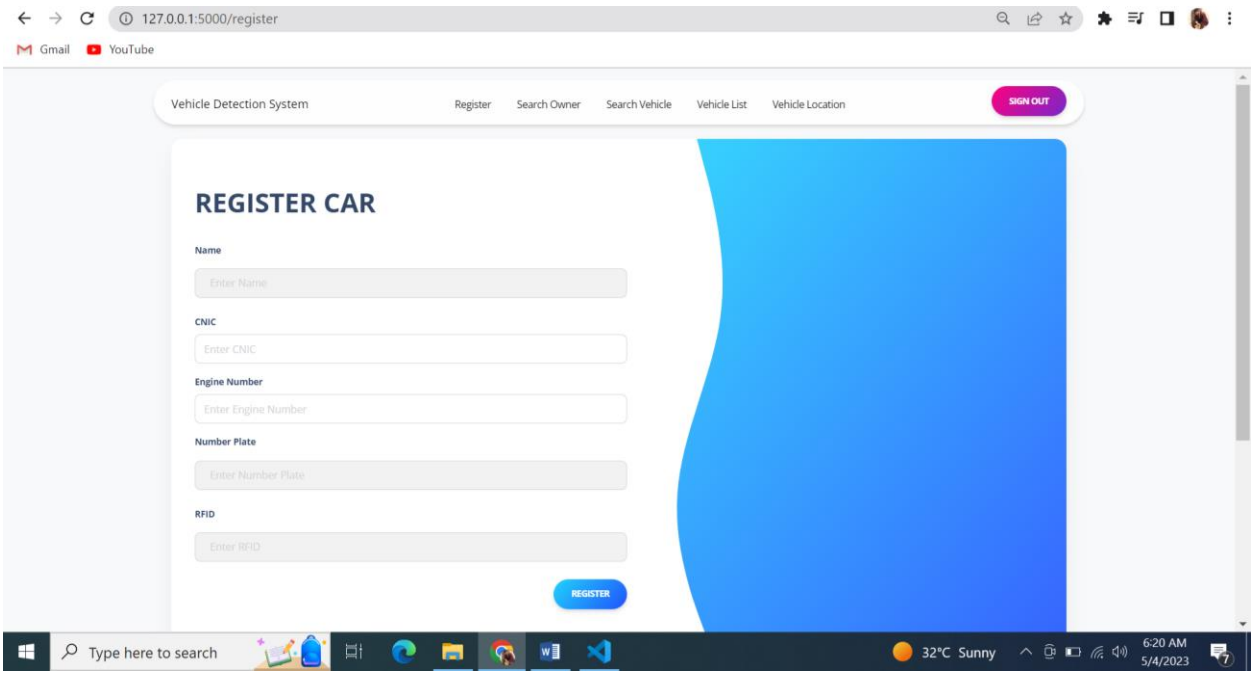
LOGIN PAGE



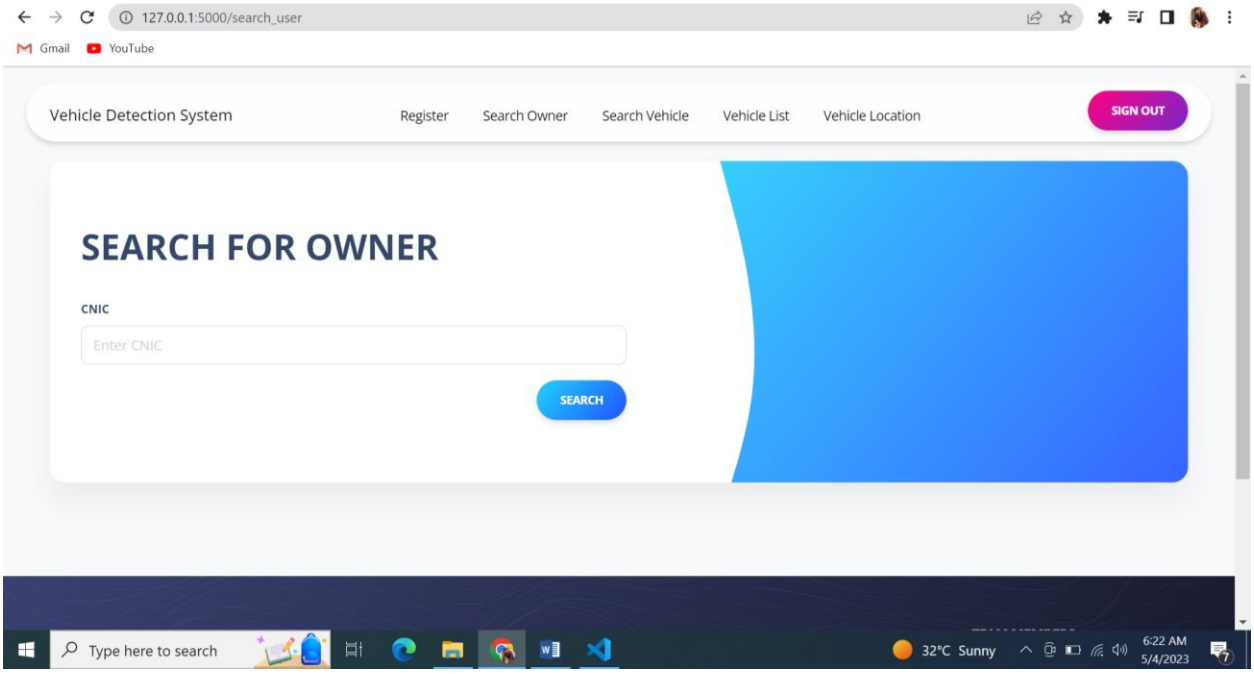
ADMIN DASHBOARD



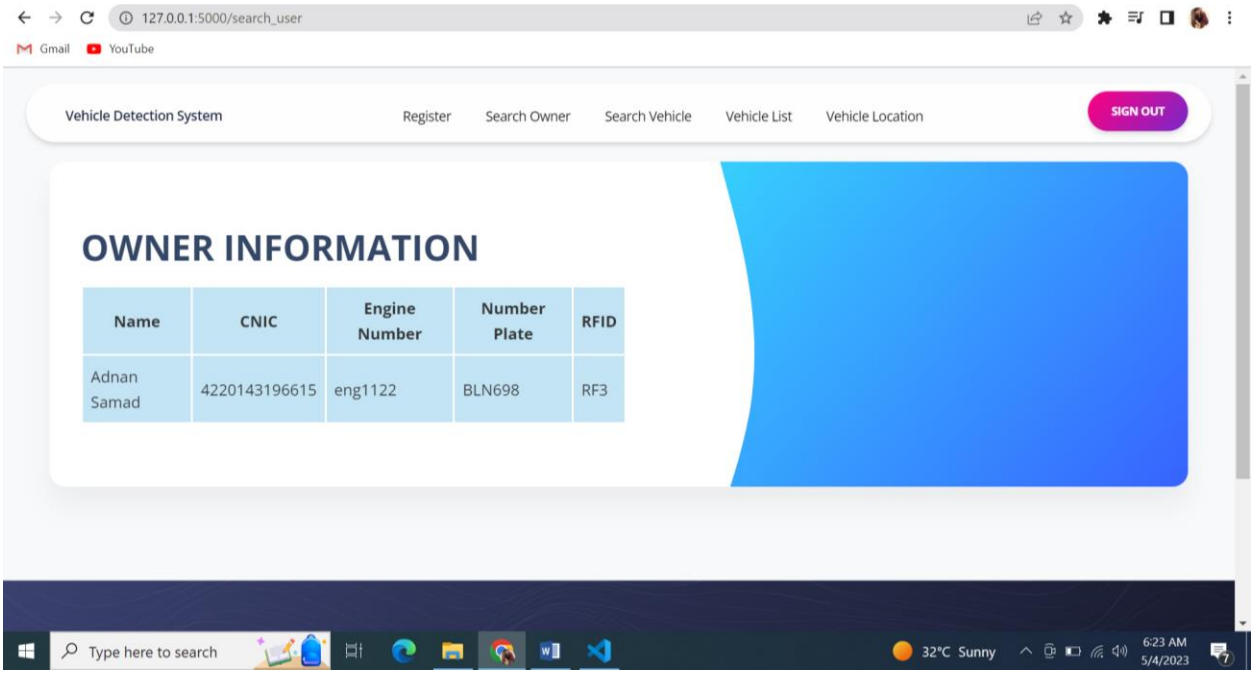
OFFICER DASHBOARD



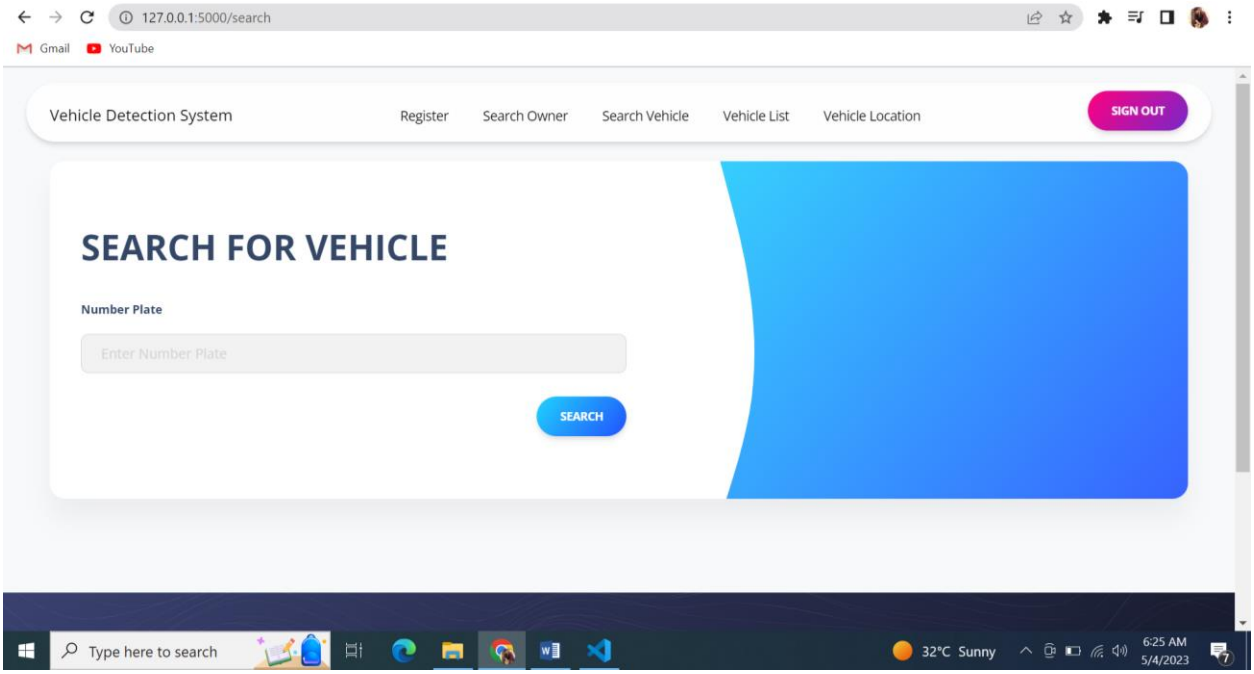
REGISTER CAR PAGE



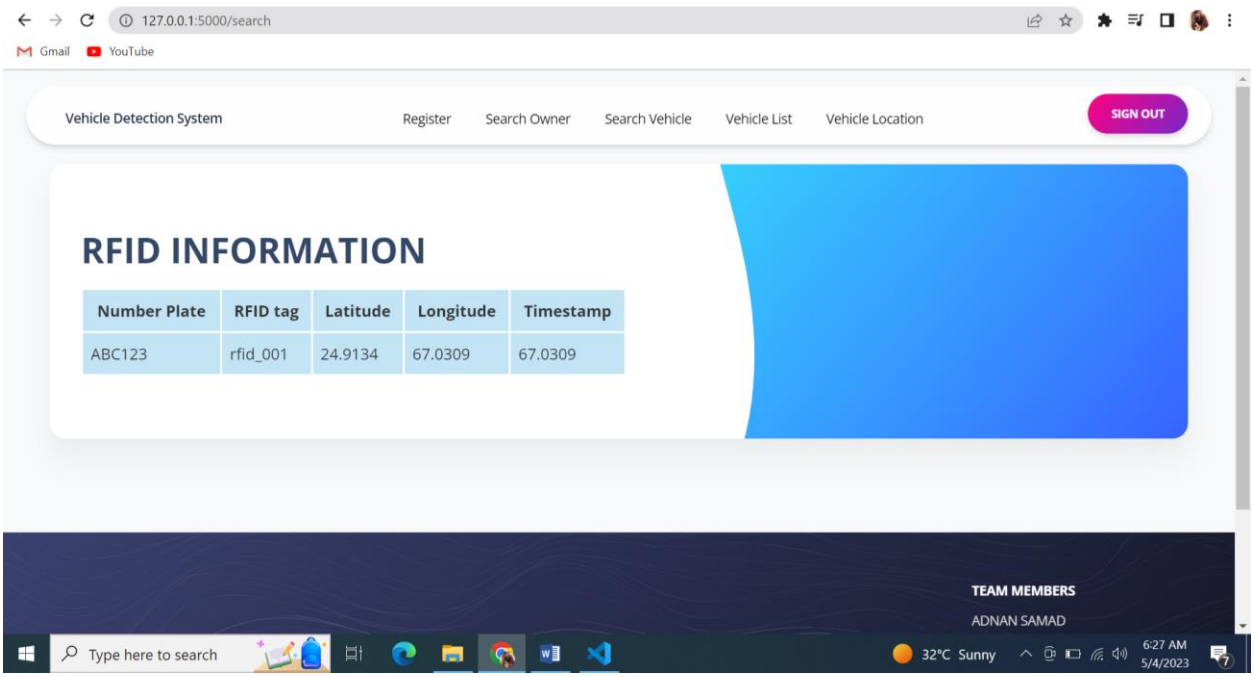
SEARCH PAGE FOR OWNER



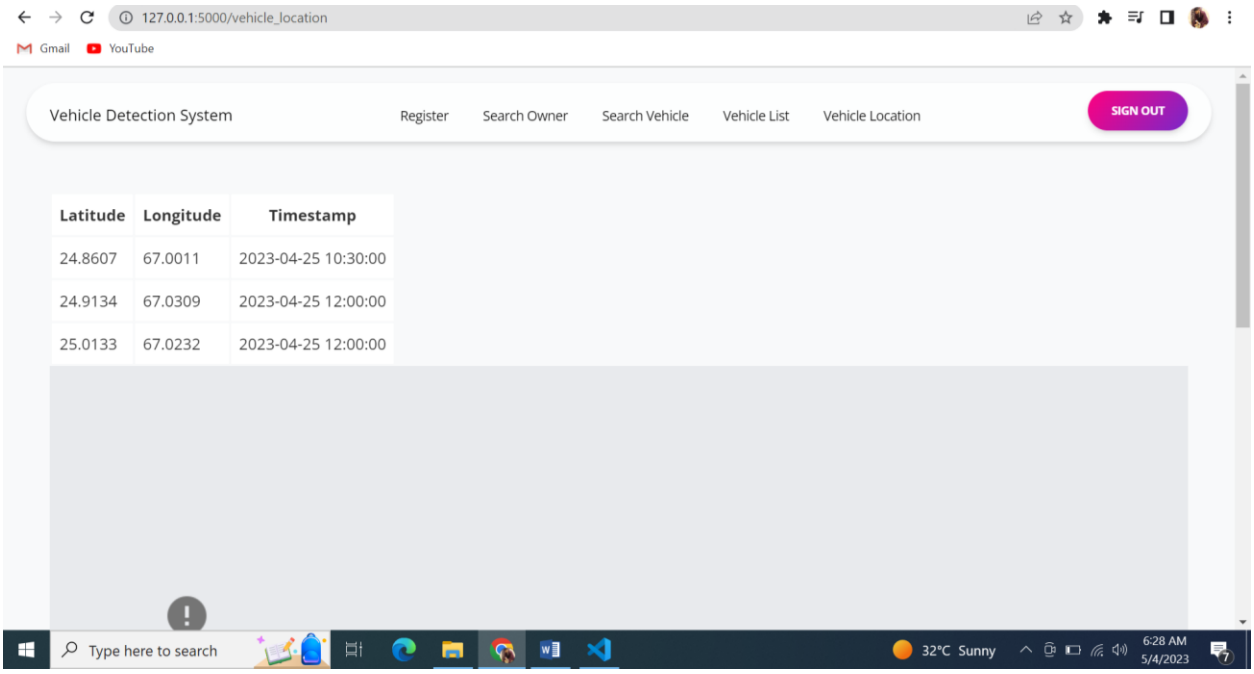
OWNER INFORMATION PAGE



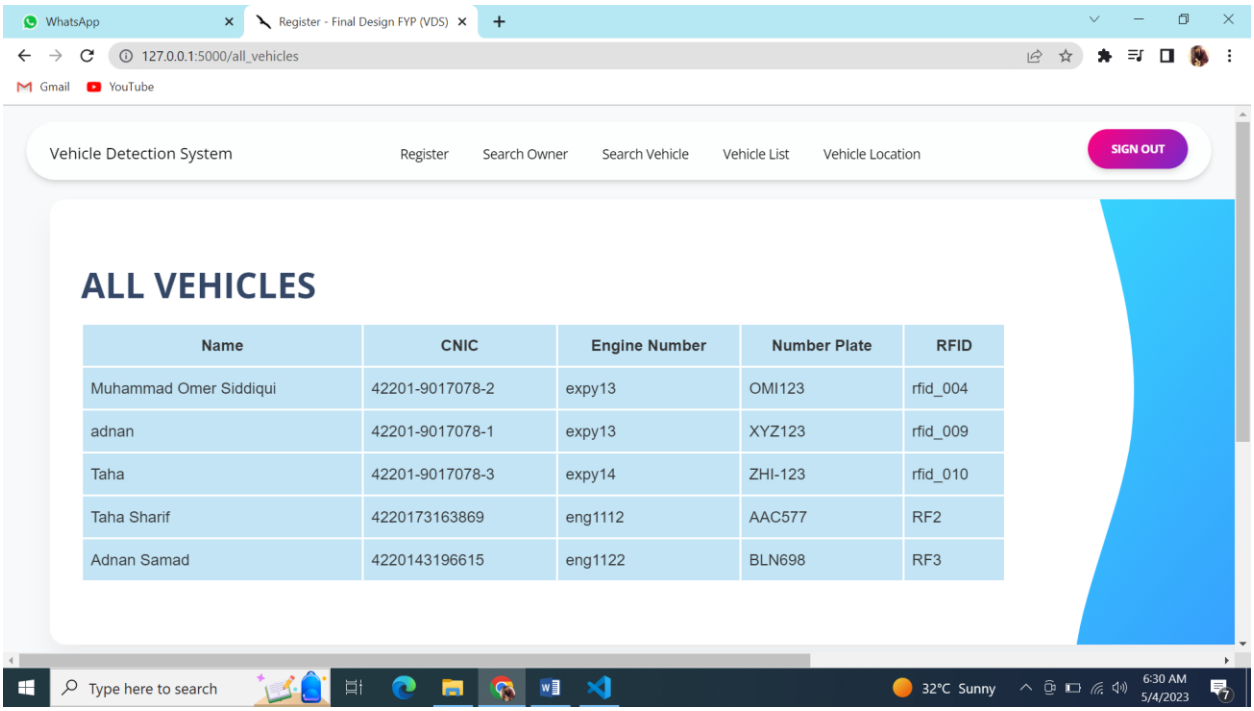
SEARCH PAGE VEHICLE



RFID INFORMATION PAGE



VEHICLE LOCATION PAGE



VEHICLE LIST PAGE

CHAPTER 6
ALGORITHM ANALYSIS AND COMPLEXITY

6.1 ALGORITHM

In our project we are using Reverse Geocoding because reverse geocoding is a process that takes geographic coordinates and uses them to find a descriptive address or place name. It is the opposite of forward geocoding, which attempts to find geographic coordinates based on an address or place name. Reverse geocoding is used in many applications, such as locating nearby amenities, finding nearby addresses and landmarks, and even pinpointing cell phone locations.

6.2 PSEUDOCODE

```
# Initialize RFID scanner and GPS module

rfid_scanner = RFIDScanner()

gps = GPS()

# Connect to database

db = Database()

while True:

    # Wait for a vehicle to pass by the scanner

    vehicle = rfid_scanner.wait_for_vehicle()

    # Get the registration number of the vehicle from RFID

    reg_num = vehicle.get_registration_number()

    # Get the current location of the vehicle from GPS

    location = gps.get_location()

    longitude = location.longitude

    latitude = location.latitude

    # Store the information in the database

    db.store_vehicle_data(reg_num, longitude, latitude)

# To retrieve the information from the database

def retrieve_vehicle_data(reg_num):

    data = db.get_vehicle_data(reg_num)

    return data
```

CHAPTER 7
IMPLEMENTATION

7.1 INTRODUCTION

In this chapter we will discuss the source code of our chapter. This chapter outlines the specific steps taken to create the system.

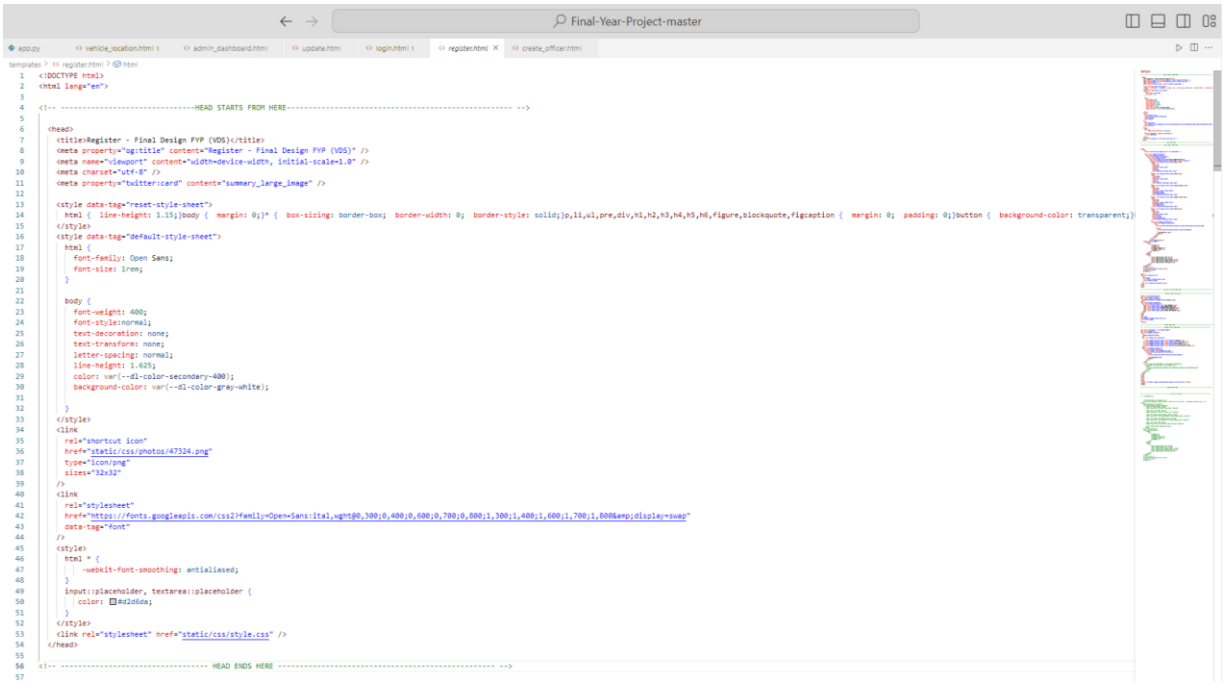
7.2 CODING

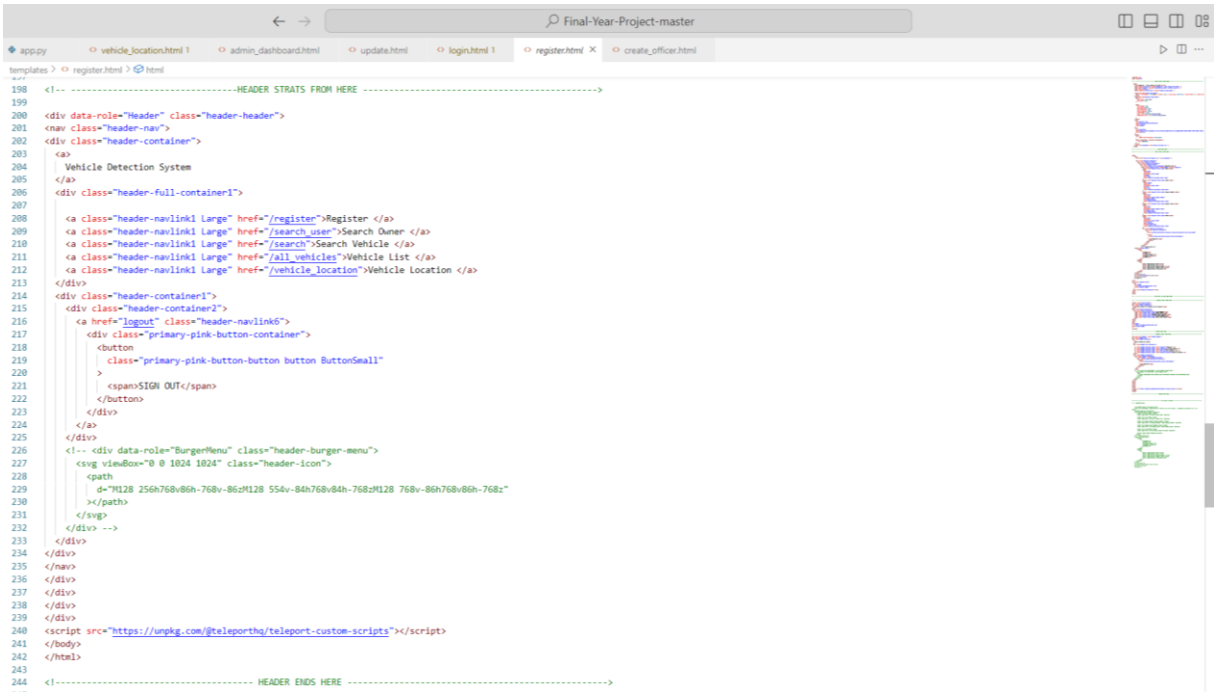
The Source code of our project is as follows.

7.2.1 HTML'S

7.2.1.1 BASE HTML

Base Html is created to give a common design to all of templates

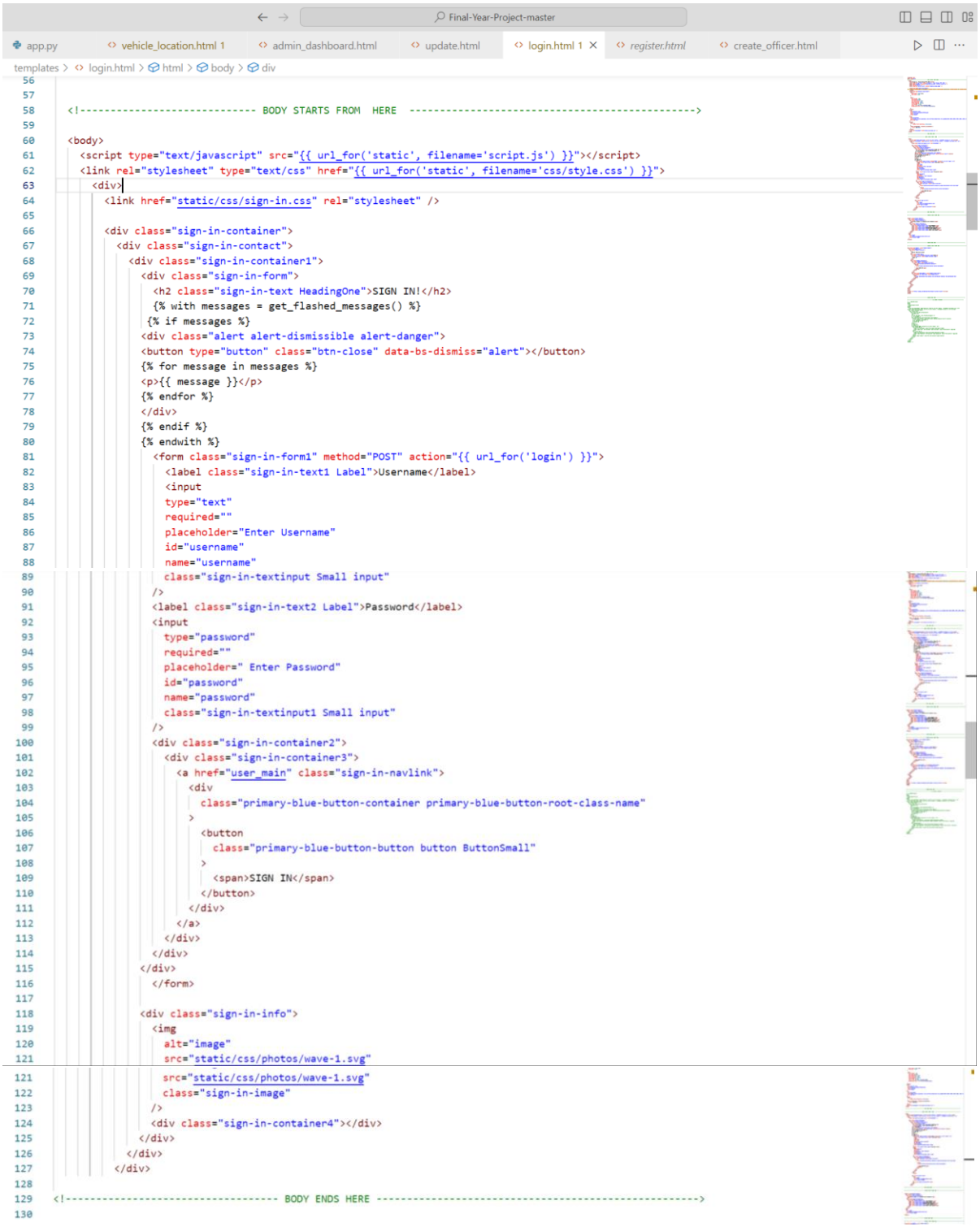




HEADER OF ALL THE TEMPALTES

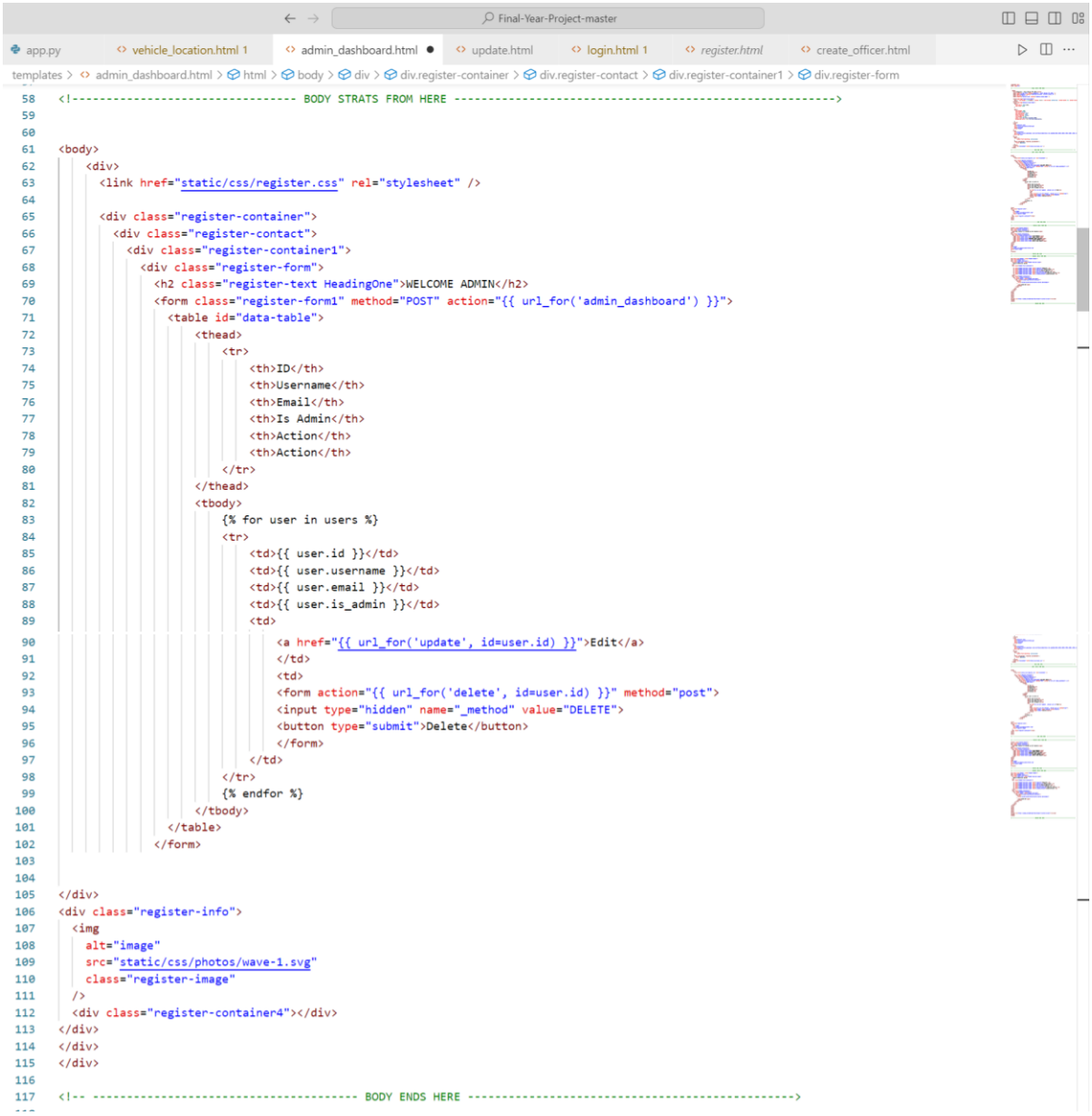
FOOTER OF ALL THE TEMPLATES

7.2.1.2 LOGIN HTML




```
<!------------------------------ BODY STARTS FROM  HERE  ---------------------------------------------->

<body>
  <script type="text/javascript" src="{{ url_for('static', filename='script.js') }}"></script>
  <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='css/style.css') }}">
    <div>
      <link href="static/css/sign-in.css" rel="stylesheet" />

      <div class="sign-in-container">
        <div class="sign-in-contact">
          <div class="sign-in-container1">
            <div class="sign-in-form">
              <h2 class="sign-in-text HeadingOne">SIGN IN!</h2>
              {% with messages = get_flashed_messages() %}
             {% if messages %}
            <div class="alert alert-dismissible alert-danger">
            <button type="button" class="btn-close" data-bs-dismiss="alert"></button>
            {% for message in messages %}
            <p>{{ message }}</p>
            {% endfor %}
            </div>
            {% endif %}
            {% endwith %}
              <form class="sign-in-form1" method="POST" action="{{ url_for('login') }}">
                <label class="sign-in-text1 Label">Username</label>
                <input
                type="text"
                required=""
                placeholder="Enter Username"
                id="username"
                name="username"
                class="sign-in-textinput Small input"
              />
              <label class="sign-in-text2 Label">Password</label>
              <input
                type="password"
                required=""
                placeholder=" Enter Password"
                id="password"
                name="password"
                class="sign-in-textinput1 Small input"
              />
              <div class="sign-in-container2">
                <div class="sign-in-container3">
                  <a href="user_main" class="sign-in-navlink">
                    <div
                      class="primary-blue-button-container primary-blue-button-root-class-name"
                    >
                      <button
                        class="primary-blue-button-button button ButtonSmall"
                      >
                        <span>SIGN IN</span>
                      </button>
                    </div>
                  </a>
                </div>
              </div>
            </div>
              </form>

            <div class="sign-in-info">
              <img
                alt="image"
                src="static/css/photos/wave-1.svg"
                src="static/css/photos/wave-1.svg"
                class="sign-in-image"
              />
              <div class="sign-in-container4"></div>
            </div>
          </div>
        </div>
      </div>

<!---------------------------------- BODY ENDS HERE ---------------------------------------------------->
```

7.2.1.3 ADMIN DASHBOARD HTML



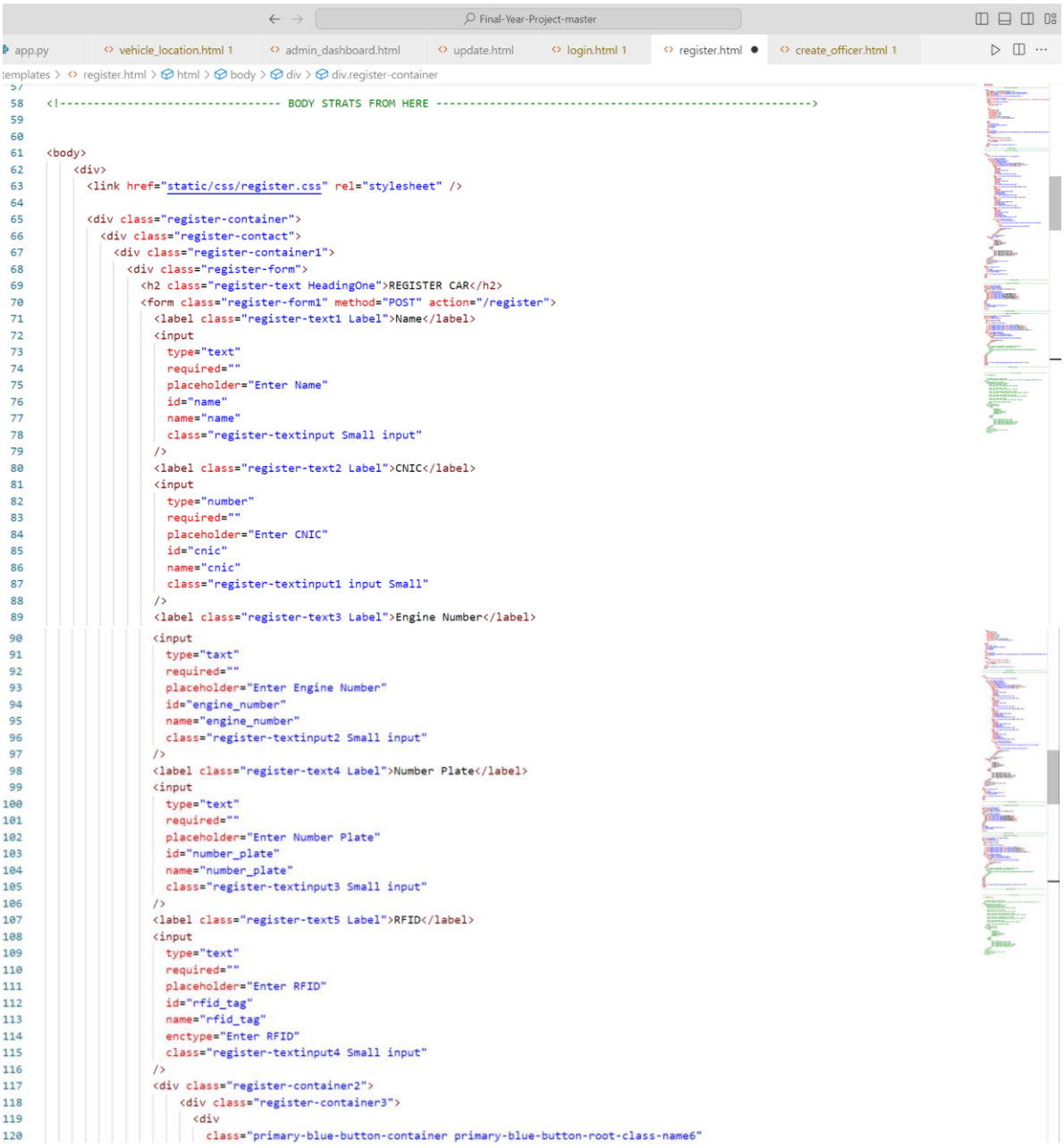
7.2.1.4 ADMIN UPDATE INFORMATION HTML


```
<!-------------------------------- BODY STRATS FROM HERE ---------------------------------------------------------->


<body>
    <div>
    <script type="text/javascript" src="{{ url_for('static', filename='script.js') }}"></script>
    <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='css/style.css') }}">

    </div >
      <div class="register-container">
        <div class="register-contact">
          <div class="register-container1">
            <div class="register-form">
              <h2 class="register-text HeadingOne">UPDATE INFO</h2>
              <form class="register-form1" method="POST">
                <label class="sign-in-text1 Label">Username</label>
                  <input
                  type="text"
                  required=""
                  placeholder="Enter Username"
                  value="{{ user.username }}"
                  name="username"
                  class="sign-in-textinput Small input"
                />
                <label class="sign-in-text2 Label">Password</label>
                <input
                  type="password"
                  required=""
                  placeholder=" Enter Password"
                  value="{{ user.password }}"
                  name="password"
                  class="sign-in-textinput1 Small input"
                />
                <label class="sign-in-text2 Label">Email</label>
                <input
                  type="email"
                  required=""
                  placeholder=" Enter Email"
                  id="email"
                  value="{{ user.email }}"
                  name="email"
                  class="sign-in-textinput1 Small input"
                />
                <input type="checkbox" name="is_admin" {% if user.is_admin %}checked{% endif %}
                  required=""
                  class="sign-in-textinput1 Small input"
                  >
                  <div class="register-container2">
                    <div class="register-container3">
                      <div
                        class="primary-blue-button-container primary-blue-button-root-class-name6"
                      >
                        <button
                          class="primary-blue-button-button button ButtonSmall"
                        >
                          <span>UPDATE</span>
                        </button>
                      </div>
                    </div>
                  </div>
              </form>
</div>
<div class="register-info">
  <img
    alt="image"
    src="/static/css/photos/wave-1.svg"
    class="register-image"
  />
  <div class="register-container4"></div>
</div>
</div>
</div>
      </div>
    </div>

<!-- ----------------------------------------BODY ENDS HERE ------------------------------------------------>
```

7.2.1.5 CREATE OFFICER HTML


```
<!---------------------------- BODY STARTS FROM  HERE  ---------------------------------------------->

<body>
  <script type="text/javascript" src="{{ url_for('static', filename='script.js') }}"></script>
  <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='css/style.css') }}">
    <div>
      <link href="static/css/sign-in.css" rel="stylesheet" />

      <div class="sign-in-container">
        <div class="sign-in-contact">
          <div class="sign-in-container1">
            <div class="sign-in-form">
              <h2 class="sign-in-text HeadingOne">REGISTER SYSTEM USER!</h2>
              {% with messages = get_flashed_messages() %}
              {% if messages %}
              <div class="alert alert-dismissible {% if 'success' in messages[0] %}alert-success{% else %}alert-danger{% endif %}">
              <button type="button" class="btn-close" data-bs-dismiss="alert"></button>
              {% for message in messages %}
              <p>{{ message }}</p>
              {% endfor %}
              </div>
              {% endif %}
              {% endwith %}



              <form class="sign-in-form1" method="POST" action="{{ url_for('create_officer') }}">
                <label class="sign-in-text1 Label">Username</label>
                <input
                type="text"
                required=""
                placeholder="Enter Username"
                id="username"
                name="username"
                class="sign-in-textinput Small input"
              />
              <label class="sign-in-text2 Label">Password</label>
              <input
                type="password"
                required=""
                placeholder=" Enter Password"
                id="password"
                name="password"
                class="sign-in-textinput1 Small input"
              />
              <label class="sign-in-text2 Label">Email</label>
              <input
                type="email"
                required=""
                placeholder=" Enter Email"
                id="email"
                name="email"
                class="sign-in-textinput1 Small input"
              />
              <div class="sign-in-container2">
                <div class="sign-in-container3">
                  <a href="main-admin-page.html" class="sign-in-navlink">
                    <div
                      class="primary-blue-button-container primary-blue-button-root-class-name"
                    >
                      <button
                        class="primary-blue-button-button button ButtonSmall"
                      >
                        <span>SIGN IN</span>
                      </button>
                    </div>
                  </a>
                </div>
              </div>
            </div>
              </form>
            <div class="sign-in-info">
              <img
                alt="image"
                src="static/css/photos/wave-1.svg"
                class="sign-in-image"
              />
              <div class="sign-in-container4"></div>
            </div>
          </div>
        </div>

<!---------------------------------- BODY ENDS HERE ------------------------------------------------->
```

7.2.1.6 REGISTER CAR HTML



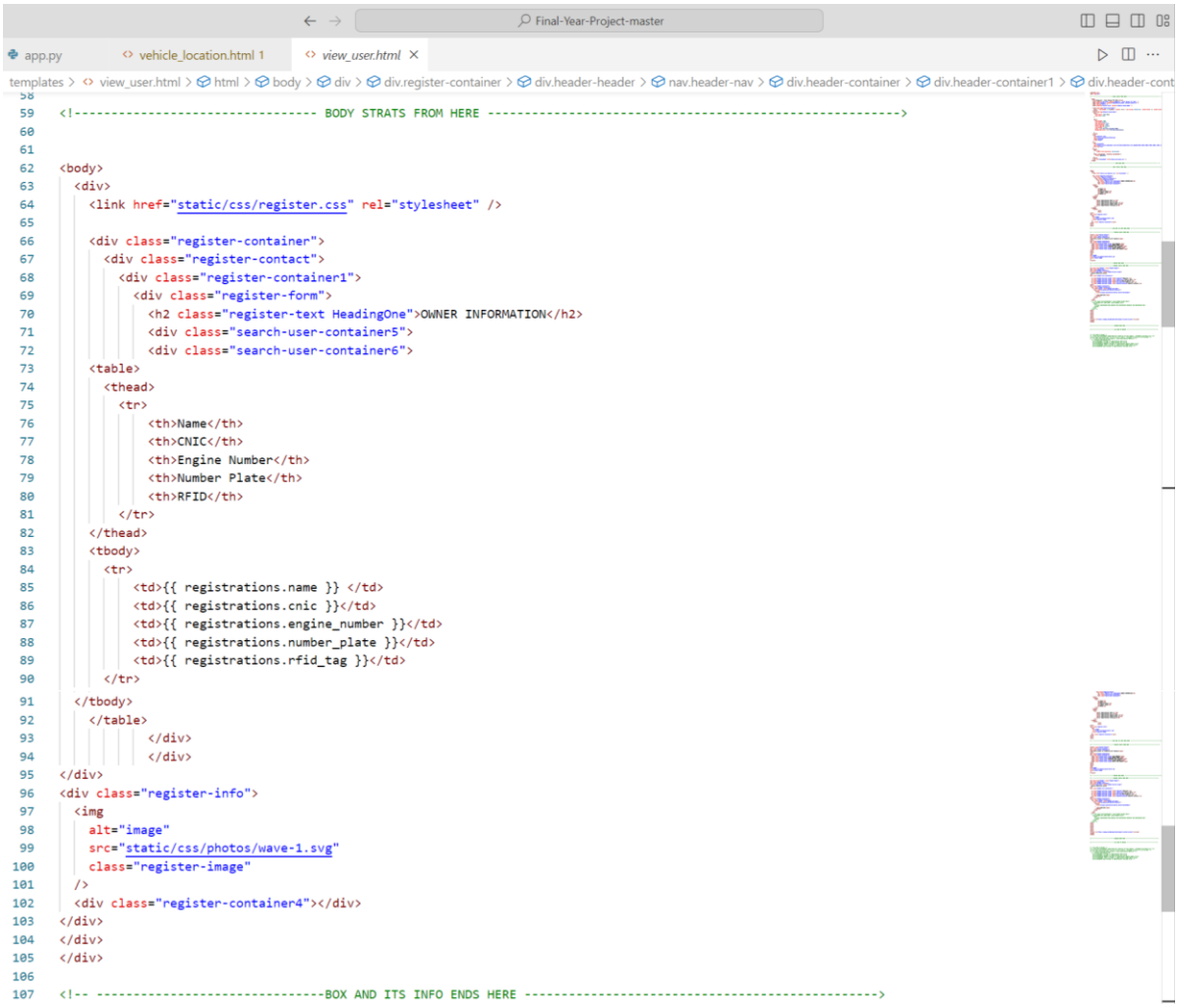

```
<!-------------------------------- BODY STRATS FROM HERE ---------------------------------------------------->


<body>
    <div>
      <link href="static/css/register.css" rel="stylesheet" />

      <div class="register-container">
        <div class="register-contact">
          <div class="register-container1">
            <div class="register-form">
              <h2 class="register-text HeadingOne">REGISTER CAR</h2>
              <form class="register-form1" method="POST" action="/register">
                <label class="register-text1 Label">Name</label>
                <input
                  type="text"
                  required=""
                  placeholder="Enter Name"
                  id="name"
                  name="name"
                  class="register-textinput Small input"
                />
                <label class="register-text2 Label">CNIC</label>
                <input
                  type="number"
                  required=""
                  placeholder="Enter CNIC"
                  id="cnic"
                  name="cnic"
                  class="register-textinput1 input Small"
                />
                <label class="register-text3 Label">Engine Number</label>
                <input
                  type="taxt"
                  required=""
                  placeholder="Enter Engine Number"
                  id="engine_number"
                  name="engine_number"
                  class="register-textinput2 Small input"
                />
                <label class="register-text4 Label">Number Plate</label>
                <input
                  type="text"
                  required=""
                  placeholder="Enter Number Plate"
                  id="number_plate"
                  name="number_plate"
                  class="register-textinput3 Small input"
                />
                <label class="register-text5 Label">RFID</label>
                <input
                  type="text"
                  required=""
                  placeholder="Enter RFID"
                  id="rfid_tag"
                  name="rfid_tag"
                  enctype="Enter RFID"
                  class="register-textinput4 Small input"
                />
                <div class="register-container2">
                    <div class="register-container3">
                      <div
                        class="primary-blue-button-container primary-blue-button-root-class-name6"
```

```
                    >
                      <button
                        class="primary-blue-button-button button ButtonSmall"
                      >
                        <span>Register</span>
                      </button>
                    </div>
                  </div>
                </div>
              </form>
              {% if registrations %}
    <table id="data-table">
        <thead>
            <tr>
                <th>Name</th>
                <th>CNIC</th>
                <th>Engine Number</th>
                <th>Number Plate</th>
                <th>RFID</th>
            </tr>
        </thead>
        <tbody>
            <tr>
                <td>{{ registrations.name }}</td>
                <td>{{ registrations.cnic }}</td>
                <td>{{ registrations.engine_number }}</td>
                <td>{{ registrations.number_plate }}</td>
                <td>{{ registrations.rfid_tag }}</td>
            </tr>
        </tbody>
    </table>
    {% elif error %}
    <p style="color:red">{{ error }}</p>
    <p>{{ error }}</p>
    {% endif %}

</div>
<div class="register-info">
  <img
    alt="image"
    src="static/css/photos/wave-1.svg"
    class="register-image"
  />
  <div class="register-container4"></div>
</div>
</div>
</div>

<!-- ----------------------------------------BODY ENDS HERE ---------------------------------------------->
```

7.2.1.7 SEARCH OWNER HTML


```
<!------------------------------ BODY STRATS FROM HERE ---------------------------------------------------->

<body>
    <div>
      <link href="static/css/register.css" rel="stylesheet" />

      <div class="register-container">
        <div class="register-contact">
          <div class="register-container1">
            <div class="register-form">
              <h2 class="register-text HeadingOne">SEARCH FOR OWNER </h2>

              {% if error %}
              <p style="color: red">{{ error }}</p>
              {% endif %}

              <form class="register-form1" id="search-form" method="POST" action="{{ url_for('search_user') }}">
                <label class="register-text1 Label">CNIC</label>
                <input
                  type="number"
                  required=""
                  placeholder="Enter CNIC"
                  id="cnic"
                  name="cnic"
                  class="register-textinput Small input"
                />
                <div class="register-container2">
                    <div class="register-container3">
                      <div
                        class="primary-blue-button-container primary-blue-button-root-class-name6"
                      >
                        <button
                          class="primary-blue-button-button button ButtonSmall"
                        >
                          <span>SEARCH</span>
                        </button>
                      </div>
                    </div>
                </div>
              </form>


              {% if registrations %}
              <table id="data-table">
                <thead>
                    <tr>
                        <th>Name</th>
                        <th>CNIC</th>
                        <th>Engine Number</th>
                        <th>Number Plate</th>
                        <th>RFID</th>
                    </tr>
                </thead>
                <tbody>
                    <tr>
                        <td>{{ registrations.name }}</td>
                        <td>{{ registrations.cnic }}</td>
                        <td>{{ registrations.engine_number }}</td>
                        <td>{{ registrations.number_plate }}</td>
                        <td>{{ registrations.rfid_tag }}</td>
                    </tr>
                </tbody>
              </table>
              {% elif error %}
              {% endif %}

    </div>
    <div class="register-info">
      <img
        alt="image"
        src="static/css/photos/wave-1.svg"
        class="register-image"
      />
      <div class="register-container4"></div>
    </div>
    </div>
    </div>

    <!-- ------------------------------BODY ENDS HERE ---------------------------------------------->
```

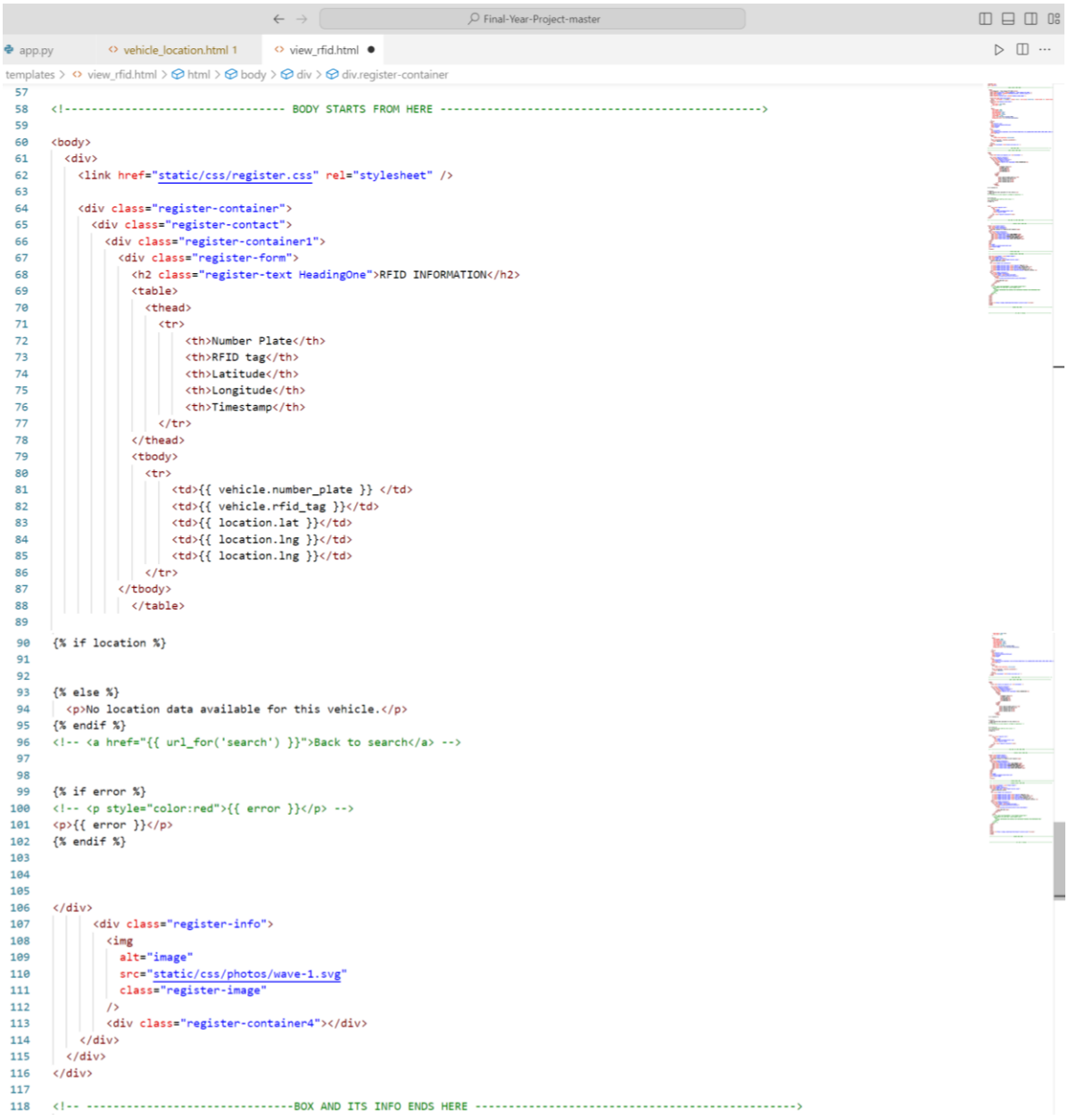
7.2.1.8 OWNER INFORMATION HTML



7.2.1.9 SEARCH VEHICLE HTML

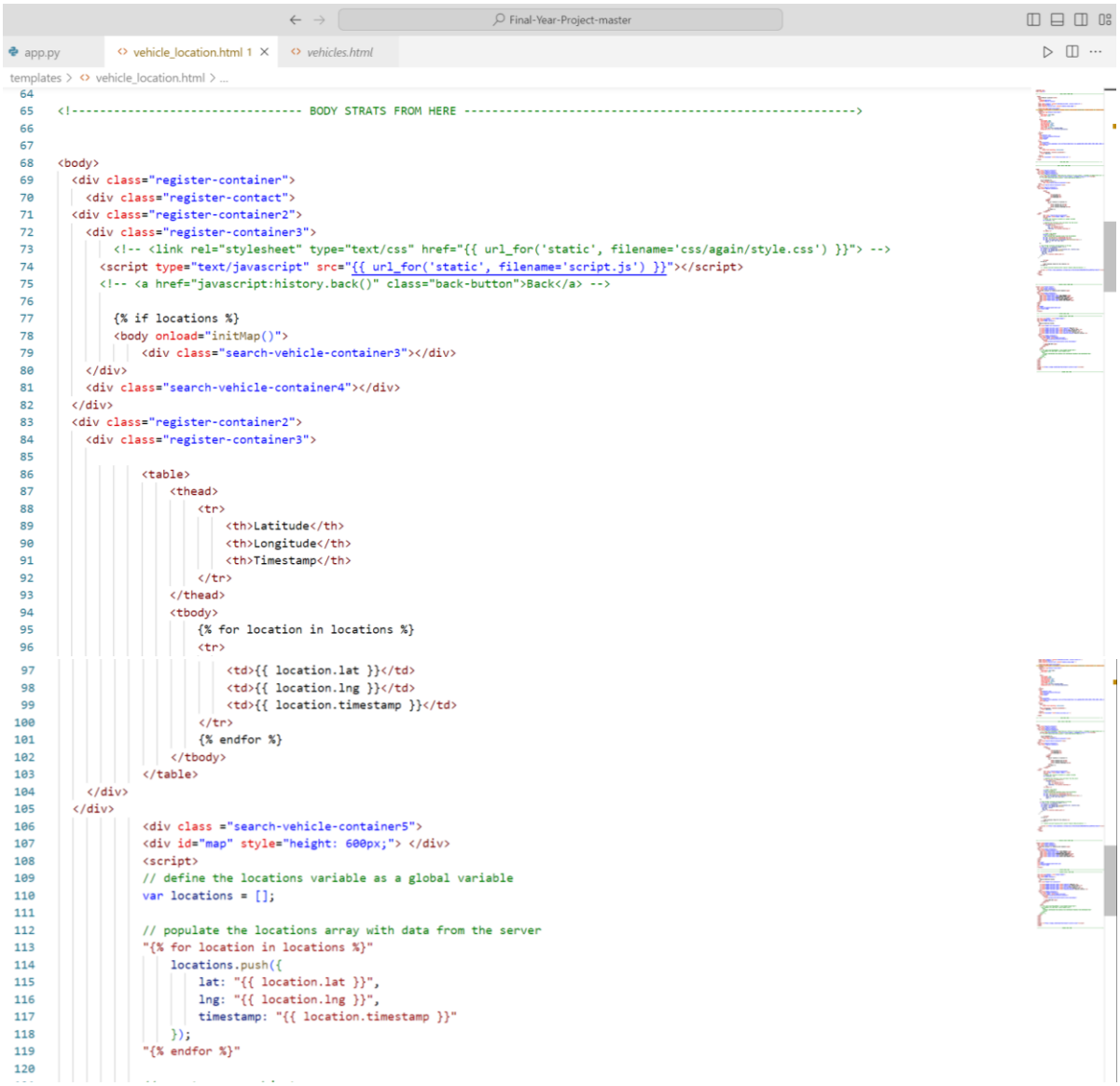
```
<!-------------------------------- BODY STRATS FROM HERE ---------------------------------------------------------->


<body>
  <script type="text/javascript" src="{{ url_for('static', filename='script.js') }}"></script>
  <div>
    <link href="static/css/register.css" rel="stylesheet" />

    <div class="register-container">
      <div class="register-contact">
        <div class="register-container1">
          <div class="register-form">
            <h2 class="register-text HeadingOne">SEARCH FOR VEHICLE </h2>

            {% if error %}
                <p style="color: red">{{ error }}</p>
                {% endif %}
                <form method="POST" class="register-form1">
                  <label class="register-text1 Label" for="number_plate">Number Plate</label>
                  <input
                    type="text"
                    required=""
                    id="number_plate"
                    name="number_plate"
                    placeholder="Enter Number Plate"
                    class="register-textinput Small input"
                  />
                  <div class="register-container2">
                    <div
                      class="primary-blue-button-container primary-blue-button-root-class-name6"
                    >
                      <button type="submit" class="primary-blue-button-button button ButtonSmall">
                        Search
                      </button>
                    </div>
                  </div>

                </form>


            {% if registrations %}
            <table id="data-table">
              <thead>
                <tr>
                    <th>Name</th>
                    <th>CNIC</th>
                    <th>Engine Number</th>
                    <th>Number Plate</th>
                    <th>RFID</th>
                </tr>
              </thead>
              <tbody>
                <tr>
                    <td>{{ registrations.name }}</td>
                    <td>{{ registrations.cnic }}</td>
                    <td>{{ registrations.engine_number }}</td>
                    <td>{{ registrations.number_plate }}</td>
                    <td>{{ registrations.rfid_tag }}</td>
                </tr>
              </tbody>
            </table>
            {% elif error %}
            {% endif %}


  </div>
  <div class="register-info">
    <img
      alt="image"
      src="static/css/photos/wave-1.svg"
      class="register-image"
    />
    <div class="register-container4"></div>
  </div>
  </div>
  </div>

  <!-- ------------------------------------BODY ENDS HERE ---------------------------------------------->
```

7.2.1.10 RFID INFORMATION HTML




```
<!-------------------------------- BODY STARTS FROM HERE ---------------------------------------------->

<body>
  <div>
    <link href="static/css/register.css" rel="stylesheet" />

    <div class="register-container">
      <div class="register-contact">
        <div class="register-container1">
          <div class="register-form">
            <h2 class="register-text HeadingOne">RFID INFORMATION</h2>
            <table>
              <thead>
                <tr>
                    <th>Number Plate</th>
                    <th>RFID tag</th>
                    <th>Latitude</th>
                    <th>Longitude</th>
                    <th>Timestamp</th>
                </tr>
              </thead>
              <tbody>
                <tr>
                    <td>{{ vehicle.number_plate }} </td>
                    <td>{{ vehicle.rfid_tag }}</td>
                    <td>{{ location.lat }}</td>
                    <td>{{ location.lng }}</td>
                    <td>{{ location.lng }}</td>
                </tr>
              </tbody>
                </table>

{% if location %}


{% else %}
  <p>No location data available for this vehicle.</p>
{% endif %}
<!-- <a href="{{ url_for('search') }}">Back to search</a> -->


{% if error %}
<!-- <p style="color:red">{{ error }}</p> -->
<p>{{ error }}</p>
{% endif %}



</div>
      <div class="register-info">
        <img
          alt="image"
          src="static/css/photos/wave-1.svg"
          class="register-image"
        />
        <div class="register-container4"></div>
    </div>
  </div>
</div>

<!-- ------------------------------BOX AND ITS INFO ENDS HERE ---------------------------------------------->
```

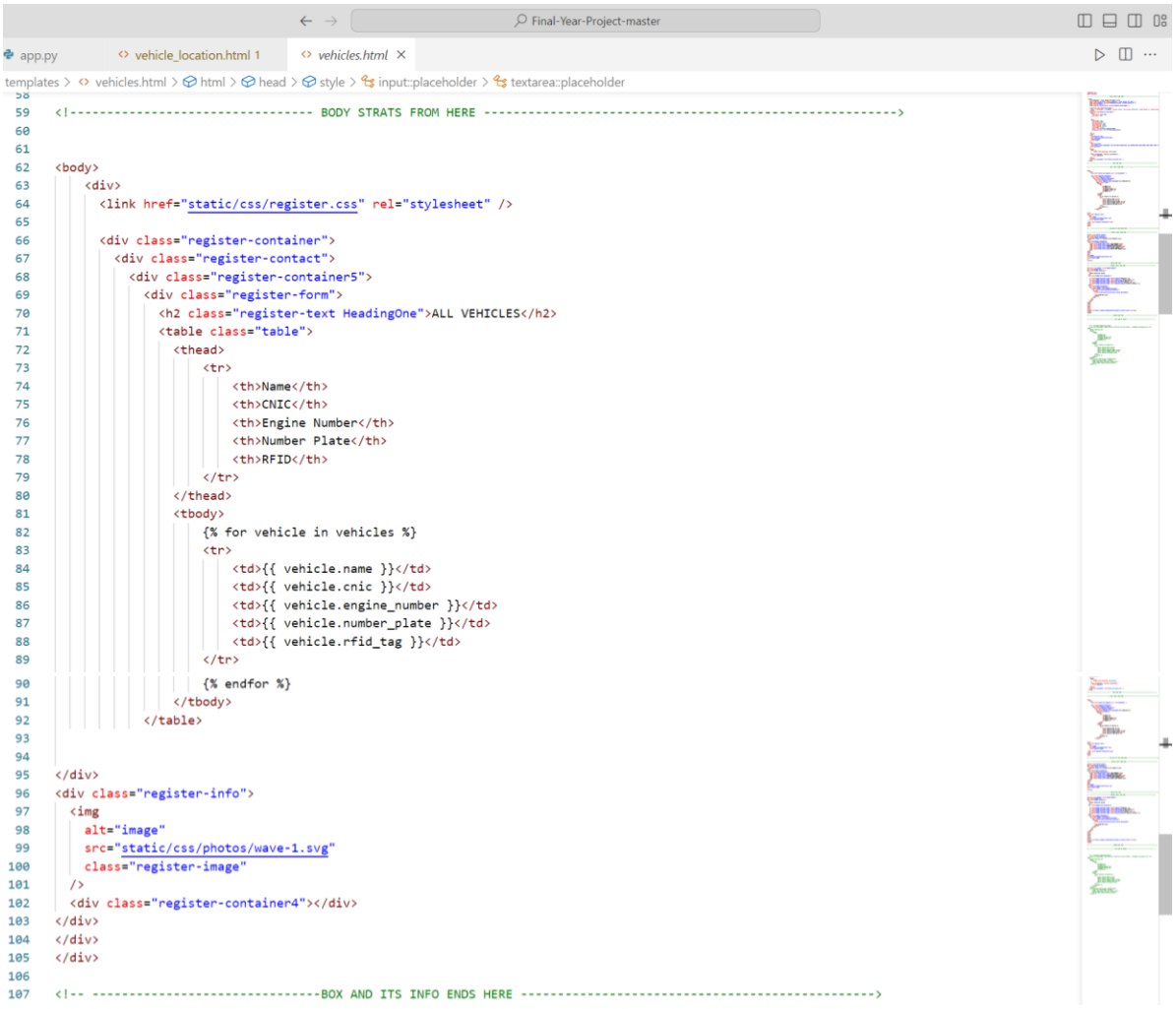
7.2.1.11 VEHICLE LOCATION HTML




```
<!-------------------------------- BODY STRATS FROM HERE ------------------------------------------------------>


<body>
  <div class="register-container">
    <div class="register-contact">
  <div class="register-container2">
    <div class="register-container3">
        <!-- <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='css/again/style.css') }}"> -->
      <script type="text/javascript" src="{{ url_for('static', filename='script.js') }}"></script>
      <!-- <a href="javascript:history.back()" class="back-button">Back</a> -->

        {% if locations %}
        <body onload="initMap()">
            <div class="search-vehicle-container3"></div>
    </div>
    <div class="search-vehicle-container4"></div>
  </div>
  <div class="register-container2">
    <div class="register-container3">

            <table>
                <thead>
                    <tr>
                        <th>Latitude</th>
                        <th>Longitude</th>
                        <th>Timestamp</th>
                    </tr>
                </thead>
                <tbody>
                    {% for location in locations %}
                    <tr>
                        <td>{{ location.lat }}</td>
                        <td>{{ location.lng }}</td>
                        <td>{{ location.timestamp }}</td>
                    </tr>
                    {% endfor %}
                </tbody>
            </table>
    </div>
  </div>
            <div class ="search-vehicle-container5">
            <div id="map" style="height: 600px;"> </div>
            <script>
            // define the locations variable as a global variable
            var locations = [];

            // populate the locations array with data from the server
            "{% for location in locations %}"
                locations.push({
                    lat: "{{ location.lat }}",
                    lng: "{{ location.lng }}",
                    timestamp: "{{ location.timestamp }}"
                });
            "{% endfor %}"
```

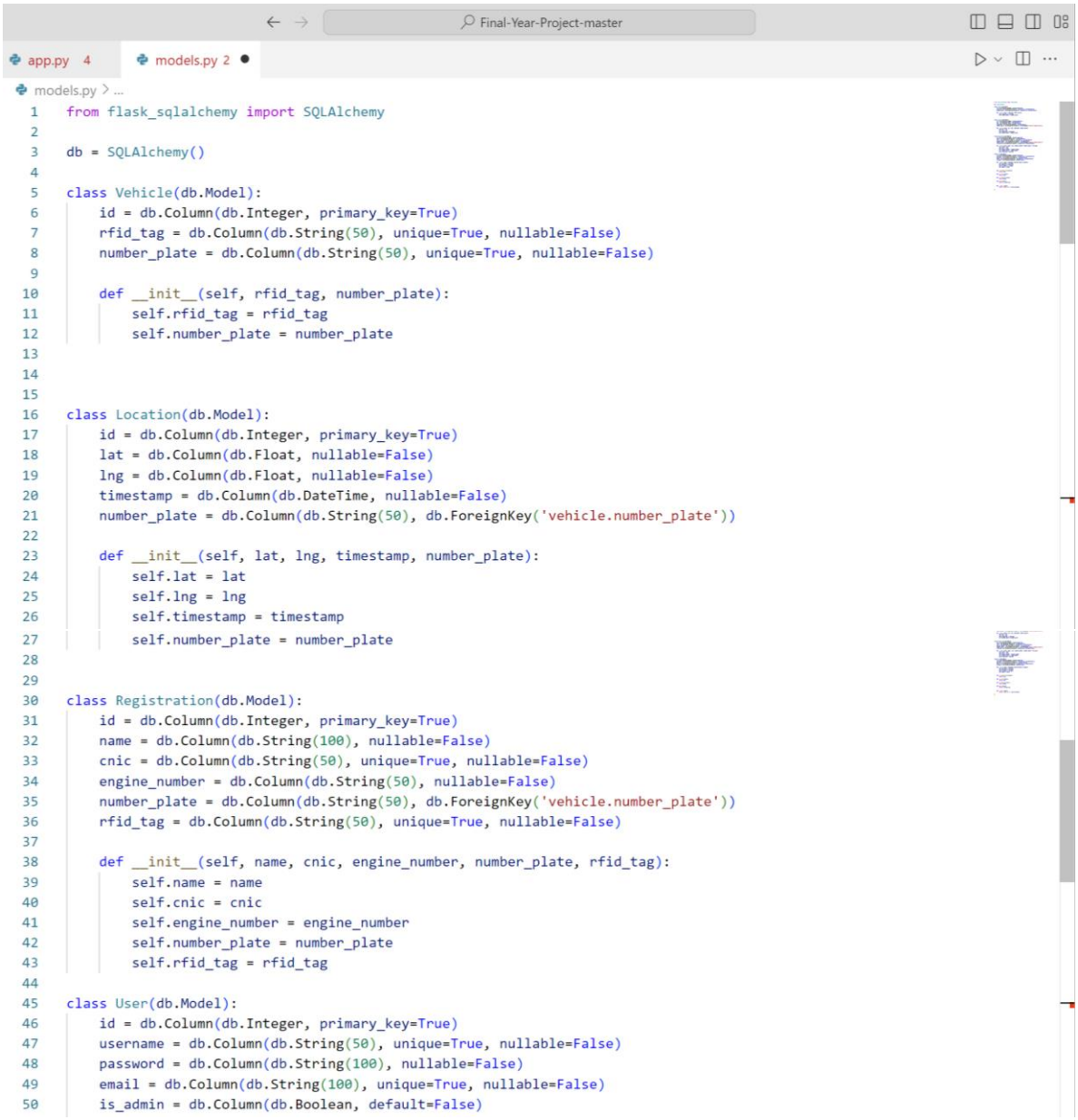
```
            // create a map object
            function initMap() {
            // get latitude and longitude values from the database
            var lat = parseFloat("{{ locations[0].lat }}");
            var lng = parseFloat("{{ locations[0].lng }}");
            var map = new google.maps.Map(document.getElementById('map'), {
                center: { lat: lat, lng: lng },
                zoom: 12
        });
        // loop through locations and add markers to the map
        for (var i = 0; i < locations.length; i++) {
          var location = locations[i];
          var latLng = new google.maps.LatLng(location.lat, location.lng);
          var marker = new google.maps.Marker({
            position: latLng ,
            map: map,
            title: '{{ locations.number_plate }}'
          });
        }
      }
            </script>
          {% else %}
            <p>No locations found for this vehicle.</p>
          {% endif %}
          <!-- <button onclick="location.href='/search'">Search Vehicle</button> -->
          <script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCcZREyA3Z0ALYAaciq1043Fya3-2dSL1E"></script>
      </div>
      </div>
    </div>
      </body>

<!-- ------------------------------------BODY ENDS HERE ------------------------------------------------>
```

7.2.1.12 VEHICLE LIST HTML




```
<!------------------------------ BODY STRATS FROM HERE ---------------------------------------------------->


<body>
    <div>
      <link href="static/css/register.css" rel="stylesheet" />

      <div class="register-container">
        <div class="register-contact">
          <div class="register-container5">
            <div class="register-form">
              <h2 class="register-text HeadingOne">ALL VEHICLES</h2>
              <table class="table">
                <thead>
                    <tr>
                        <th>Name</th>
                        <th>CNIC</th>
                        <th>Engine Number</th>
                        <th>Number Plate</th>
                        <th>RFID</th>
                    </tr>
                </thead>
                <tbody>
                    {% for vehicle in vehicles %}
                    <tr>
                        <td>{{ vehicle.name }}</td>
                        <td>{{ vehicle.cnic }}</td>
                        <td>{{ vehicle.engine_number }}</td>
                        <td>{{ vehicle.number_plate }}</td>
                        <td>{{ vehicle.rfid_tag }}</td>
                    </tr>
                    {% endfor %}
                </tbody>
              </table>


</div>
<div class="register-info">
  <img
    alt="image"
    src="static/css/photos/wave-1.svg"
    class="register-image"
  />
  <div class="register-container4"></div>
</div>
</div>
</div>

<!-- ----------------------------BOX AND ITS INFO ENDS HERE ------------------------------------------->
```

7.2.2 MODELS

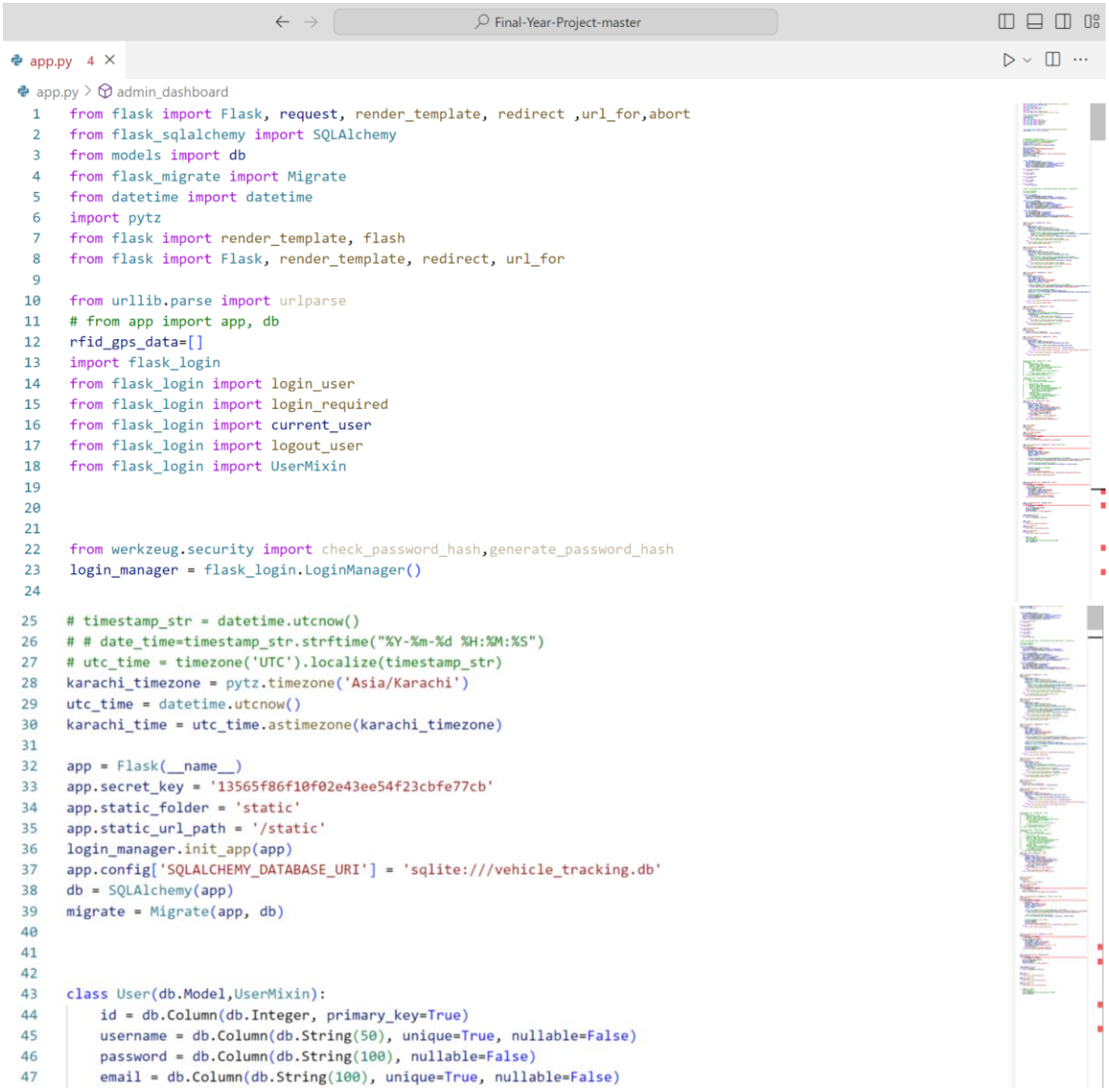



```python
from flask_sqlalchemy import SQLAlchemy

db = SQLAlchemy()

class Vehicle(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    rfid_tag = db.Column(db.String(50), unique=True, nullable=False)
    number_plate = db.Column(db.String(50), unique=True, nullable=False)

    def __init__(self, rfid_tag, number_plate):
        self.rfid_tag = rfid_tag
        self.number_plate = number_plate



class Location(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    lat = db.Column(db.Float, nullable=False)
    lng = db.Column(db.Float, nullable=False)
    timestamp = db.Column(db.DateTime, nullable=False)
    number_plate = db.Column(db.String(50), db.ForeignKey('vehicle.number_plate'))

    def __init__(self, lat, lng, timestamp, number_plate):
        self.lat = lat
        self.lng = lng
        self.timestamp = timestamp
        self.number_plate = number_plate


class Registration(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(100), nullable=False)
    cnic = db.Column(db.String(50), unique=True, nullable=False)
    engine_number = db.Column(db.String(50), nullable=False)
    number_plate = db.Column(db.String(50), db.ForeignKey('vehicle.number_plate'))
    rfid_tag = db.Column(db.String(50), unique=True, nullable=False)

    def __init__(self, name, cnic, engine_number, number_plate, rfid_tag):
        self.name = name
        self.cnic = cnic
        self.engine_number = engine_number
        self.number_plate = number_plate
        self.rfid_tag = rfid_tag

class User(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(50), unique=True, nullable=False)
    password = db.Column(db.String(100), nullable=False)
    email = db.Column(db.String(100), unique=True, nullable=False)
    is_admin = db.Column(db.Boolean, default=False)
```

```
51
52      def __init__(self, username, password,email,is_admin):
53          self.username = username
54          self.password = password
55          self.is_admin = is_admin
56          self.email = email
57
58
59      def is_authenticated(self):
60          return True
61
62      def is_active(self):
63          return True
64
65      def is_anonymous(self):
66          return False
67
68      def get_id(self):
69          return str(self.id)
70
71
72      def __repr__(self):
73          return '<User %r>' % self.username
```

7.2.3 FORMS

7.2.4 APP


Final-Year-Project-master
app.py 4
app.py > admin_dashboard
from flask import Flask, request, render_template, redirect ,url_for,abort
from flask_sqlalchemy import SQLAlchemy
from models import db
from flask_migrate import Migrate
from datetime import datetime
import pytz
from flask import render_template, flash
from flask import Flask, render_template, redirect, url_for
from urllib.parse import urlparse
# from app import app, db
rfid_gps_data=[]
import flask_login
from flask_login import login_user
from flask_login import login_required
from flask_login import current_user
from flask_login import logout_user
from flask_login import UserMixin
from werkzeug.security import check_password_hash,generate_password_hash
login_manager = flask_login.LoginManager()
# timestamp_str = datetime.utcnow()
# # date_time=timestamp_str.strftime("%Y-%m-%d %H:%M:%S")
# utc_time = timezone('UTC').localize(timestamp_str)
karachi_timezone = pytz.timezone('Asia/Karachi')
utc_time = datetime.utcnow()
karachi_time = utc_time.astimezone(karachi_timezone)
app = Flask(__name__)
app.secret_key = '13565f86f10f02e43ee54f23cbfe77cb'
app.static_folder = 'static'
app.static_url_path = '/static'
login_manager.init_app(app)
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///vehicle_tracking.db'
db = SQLAlchemy(app)
migrate = Migrate(app, db)
class User(db.Model,UserMixin):
id = db.Column(db.Integer, primary_key=True)
username = db.Column(db.String(50), unique=True, nullable=False)
password = db.Column(db.String(100), nullable=False)
email = db.Column(db.String(100), unique=True, nullable=False)

```
    is_admin = db.Column(db.Boolean, default=False)

def is_authenticated(self):
    return True

def is_active(self):
    return True

def is_anonymous(self):
    return False

def is_admin(self):
    return True

def get_id(self):
    return str(self.id)


# admin = User(username='admin', password=admin_password, email='a@g.com', is_admin=True)

# db.session.add(admin)
# db.session.commit()

class Vehicle(db.Model):
```

```
    id = db.Column(db.Integer, primary_key=True)
    rfid_tag = db.Column(db.String(50), unique=True, nullable=False)
    number_plate = db.Column(db.String(50), unique=True, nullable=False)

class Registration(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(100), nullable=False)
    cnic = db.Column(db.String(50), unique=True, nullable=False)
    engine_number = db.Column(db.String(50), nullable=False)
    number_plate = db.Column(db.String(50), db.ForeignKey('vehicle.number_plate'))
    rfid_tag = db.Column(db.String(50), unique=True, nullable=False)

class Location(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    lat = db.Column(db.Float, nullable=False)
    lng = db.Column(db.Float, nullable=False)
    timestamp = db.Column(db.DateTime, nullable=False,default=karachi_time)
    number_plate = db.Column(db.String(50), db.ForeignKey('vehicle.number_plate'))



@app.route('/search', methods=['GET', 'POST'])
@login_required
def search():
    if request.method == 'POST':
        number_plate = request.form['number_plate']
        # retrieve the vehicle information from the database
        vehicle = Vehicle.query.filter_by(number_plate=number_plate).first()
        if vehicle:
            # if the vehicle is found, retrieve its latest location from the database
            location = Location.query.filter_by(number_plate=number_plate).order_by(Location.timestamp.desc()).fi
            # display the vehicle and location information
            return render_template('view_rfid.html', vehicle=vehicle, location=location)
        else:
            # if the vehicle is not found, display an error message
            return render_template('search.html', error='Vehicle not found')
    else:
        # if the request method is GET, display the search form
        return render_template('search.html')


@app.route('/view_rfid', methods=['GET', 'POST'])
@login_required
def view_rfid():
    if request.method == 'POST':
        number_plate = request.form['number_plate']
        # retrieve the vehicle information from the database
```

```
        vehicle = Vehicle.query.filter_by(number_plate=number_plate).first()
        if vehicle:
            # if the vehicle is found, retrieve its associated RFID from the database
            rfid = Registration.query.filter_by(number_plate=number_plate).first().rfid_tag
            # display the vehicle and RFID information
            return render_template('view_rfid.html', vehicle=vehicle, rfid=rfid)
        else:
            # if the vehicle is not found, display an error message
            return render_template('view_rfid.html', error='Vehicle not found')
    else:
        # if the request method is GET, display the search form
        return render_template('view_rfid.html')



@app.route('/register', methods=['GET', 'POST'])
@login_required
def register():
    if request.method == 'POST':
        name = request.form['name']
        cnic = request.form['cnic']
        engine_number = request.form['engine_number']
        number_plate = request.form['number_plate']
        rfid_tag = request.form['rfid_tag']

        # check if the number plate or rfid tag already exist in the database
        if db.session.query(Vehicle).filter_by(number_plate=number_plate).first() or db.session.query(Registratio
            return render_template('register.html', error='Vehicle or RFID already exists')

        # create new vehicle and registration objects
        vehicle = Vehicle(rfid_tag=rfid_tag, number_plate=number_plate)
        registration = Registration(name=name, cnic=cnic, engine_number=engine_number, number_plate=number_plate,

        # add the new objects to the database
        db.session.add(vehicle)
        db.session.add(registration)
        db.session.commit()

        return render_template('register.html', success='Vehicle registered successfully')
    else:
        return render_template('register.html')


@app.route('/search_user', methods=['GET', 'POST'])
@login_required
def search_user():
    if request.method == 'POST':
        cnic = request.form['cnic']
        # retrieve the user information from the database
        registrations = db.session.query(Registration).filter_by(cnic=cnic).first()
        if registrations:
            # if the user is found, display their information
            return render_template('view_user.html', registrations=registrations)
        else:
            # if the user is not found, display an error message
            return render_template('search_user.html', error='User not found')
    else:
        # if the request method is GET, display the search form
        return render_template('search_user.html')


@app.route('/all_vehicles')
@login_required
def all_vehicles():
    vehicles = Registration.query.all()
    return render_template('vehicles.html', vehicles=vehicles)


@app.route('/vehicle_location', methods=['GET', 'POST'])
@login_required
def vehicle_location():
    if request.method == 'POST':
        number_plate = request.form['number_plate']
        vehicle = Vehicle.query.filter_by(number_plate=number_plate).first()
        if vehicle:
```

```
            locations = Location.query.filter_by(number_plate=number_plate).all()
            if locations:
                return render_template('vehicle_location.html', locations=locations)
            else:
                return render_template('vehicle_location.html', error='No locations found for this vehicle.')
        else:
            return render_template('search.html', error='Vehicle not found')
    else:
        return render_template('search.html')

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        remember_me = bool(request.form.get('remember_me'))
        user = User.query.filter_by(username=username).first()
        if user is None or not user.password==password:
            flash('Invalid username or password')
            return redirect(url_for('login'))
        login_user(user, remember=remember_me)
        if user.is_admin:
            return redirect(url_for('admin_dashboard'))
        else:
            return redirect(url_for('user_main'))
    return render_template('login.html', title='Sign In')



@app.route('/logout')
@login_required
def logout():
    logout_user()
    return redirect(url_for('index'))

@app.route('/admin_dashboard')
@login_required
def admin_dashboard():
    if not current_user.is_admin:
        abort(403)
    users = User.query.all()
    return render_template('admin_dashboard.html',users=users)



@app.route('/create_officer', methods=['GET', 'POST']) # type: ignore
@login_required
def create_officer():
    if not current_user.is_admin:
        abort(403)
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        email = request.form['email']
        is_admin = False

        # check if the number plate or rfid tag already exist in the database
        if db.session.query(User).filter_by(username=username).first() or db.session.query(User).filter_by(email=
            return render_template('create_officer.html', error='username or email already exists')

        # create new vehicle and registration objects
        user = User(username=username, password=password, email=email, is_admin=is_admin)


        # add the new objects to the database
        db.session.add(user)
        db.session.commit()
        flash('Officer created successfully!')
        return render_template('create_officer.html', success='Officer registered successfully')
    else:
        return render_template('create_officer.html')

```

```python
@app.route('/update/<int:id>', methods=['GET', 'POST'])
def update(id):
    if not current_user.is_admin:
        abort(403)
    user = User.query.get_or_404(id)
    if request.method == 'POST':
        user.username = request.form['username']
        user.password = request.form['password']
        user.email = request.form['email']
        user.is_admin = request.form.get('is_admin') == 'on'
        db.session.commit()
        return redirect(url_for('admin_dashboard'))
    return render_template('update.html', user=user)




@app.route('/delete/<int:id>', methods=['POST'])
def delete(id):
    if not current_user.is_admin:
        abort(403)
    user = User.query.get_or_404(id)
    db.session.delete(user)
    db.session.commit()
    return redirect(url_for('admin_dashboard'))


@login_manager.user_loader
def load_user(user_id):
    return User.query.get(int(user_id))


@app.route('/')
def index():
    return render_template('index.html')

@app.route('/about_us')
def about_us():
    return render_template('about_us.html')

@app.route('/user_main')
def user_main():
    return render_template('user_main.html')


if __name__ == '__main__':
    # db.create_all()
    # app.run(debug=True)
    # app.run(debug=True, host='192.168.43.206',port=5000)
    app.run(debug=True)
```

7.3 OPERATIONAL DIAGRAM

RFID's to be installed in number plate of cars and scanners are installed on various points of the city which then scan these chips. And send the scanned rfid to the database with its location which then be mapped on to google maps. The Rfid database store's location with date and time of the vehicle which it passes. this all information can be accessed by a dashboard. In the end authenticated by excise database.

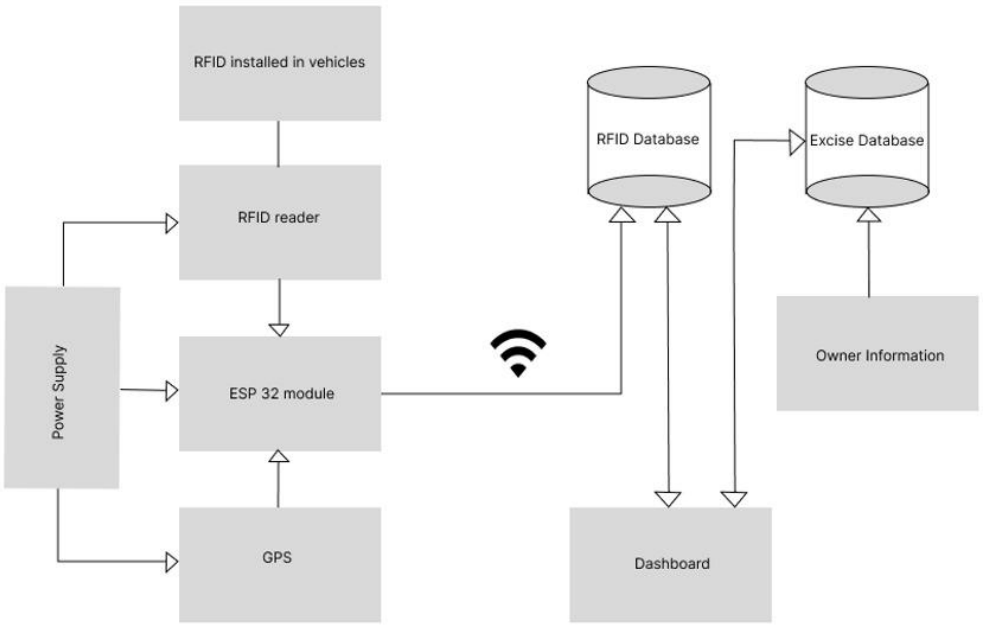


Figure 34 OPERATIONAL DIAGRAM

7.4 COMPONENT DIAAGRAM

A component diagram in a Vehicle Detection System (VDS) with RFID is a graphical representation that shows the various software and hardware components that make up the system, and how they interact with each other. The component diagram helps to visualize the overall architecture of the system and the dependencies between the various components, making it easier to understand the system's design and implementation. It also helps to identify potential areas of improvement or modifications needed to the system.

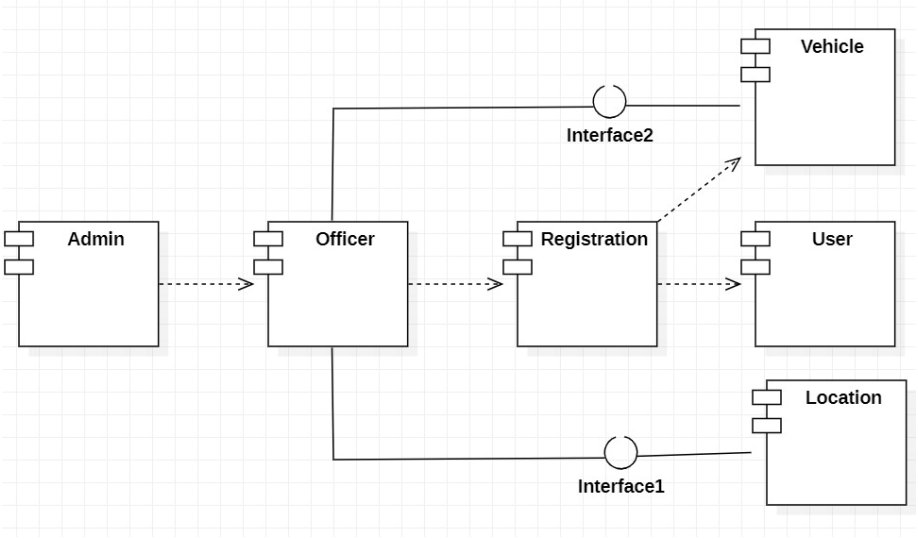


Figure 35 COMPONENT DIAGRAM

7.5 DEPLOYMENT DIARGAM

A deployment diagram in a Vehicle Detection System (VDS) with RFID is a graphical representation that shows the physical deployment of the system's components, such as hardware and software, on various computing nodes and devices. The deployment diagram helps to visualize the physical deployment of the system's components and the communication channels between them, making it easier to understand how the system is deployed in the physical environment. It also helps to identify potential issues related to the deployment, such as network latency, bandwidth constraints, or hardware limitations.

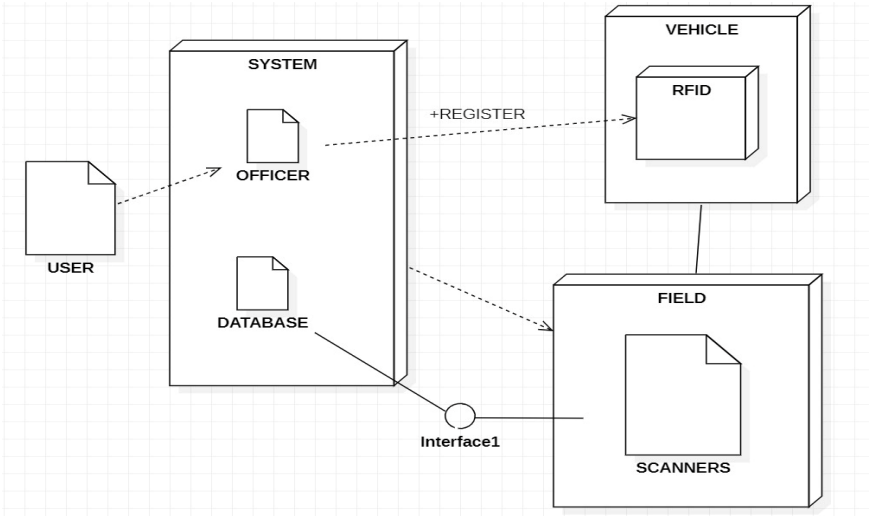


Figure 36 DEPLOYMENT DIAGRAM

CHAPTER 8
TESTING

8.1 INTRODUCTION

Testing is the process of evaluating a software application or system to ensure that it meets its intended requirements and functions correctly. Testing is typically performed by a software tester or a team of testers, who use various techniques and tools to identify defects, bugs, and other issues in the software. The goal of testing is to find and fix any defects before the software is released to end-users, to ensure that it is reliable, secure, and performs as expected.

8.2 WHITE BOX TESTING

White box testing is a type of software testing that involves testing the internal workings or code of a software application. In this type of testing, the tester has access to the source code and is able to see the software's internal structures and processes. The purpose of white box testing is to evaluate the internal logic, structure, and design of the software to ensure that it functions correctly, efficiently, and reliably. It is also known as structural testing, clear-box testing, or open-box testing. White box testing can be conducted at various levels of the software development lifecycle, including unit testing, integration testing, and system testing.

8.1.1 TEST CASE (USER LOGIN)

TEST CASE	PRE-CONDITION	TEST STEP	EXPECTED RESULT	SUCCESS CRITERIA
To ensure that the user can successfully sign-in to the system using the credentials provided by the admin.	The system is installed and configured properly. The database is set up and contains the necessary user information. The user has been provided with valid credentials by the admin.	Launch the system and navigate to the user sign-in page. Enter the valid username and password provided by the admin. Click on the "Sign In" button. Verify that the user is	The user should be able to log in to the system using valid credentials. The user should not be able to log in to the system using invalid credentials. Appropriate error messages should be displayed for	The test case passes if the user is able to log in to the system using valid credentials, and appropriate error messages are displayed for invalid credentials. The test case fails if the user is unable to log in to the

		successfully logged in to the system and the user's dashboard is displayed. Log out from the user's account. Repeat the above steps with invalid username and password combinations. Verify that appropriate error messages are displayed for incorrect credentials.	invalid credentials.	system using valid credentials or no error messages are displayed for invalid credentials.

8.2.2 TEST CASES (SEARCH VEHICLE USING NUMBER PLATE)

TEST CASE	PRE-CONDITION	TEST STEP	EXPECTED RESULT	SUCCESS CRITERIA
To verify that the system can accurately search for a vehicle using its number plate.	The system is installed and configured properly. The user has valid login credentials to access the system.	Log in to the system with valid login credentials. Navigate to the "Search Vehicle" option. Enter a valid vehicle	The system should accurately display the details of the searched vehicle for a valid number plate. An appropriate error message	The test case passes if the system accurately displays the details of the searched vehicle for a valid number plate, and appropriate error messages

	There are one or more vehicles registered in the system.	number plate in the search field. Click on the "Search" button. Verify that the details of the searched vehicle are displayed, including its make, model, color, and other relevant information. Repeat the above steps with an invalid number plate format (e.g., wrong pattern, special characters, etc.). Verify that an appropriate error message is displayed for invalid number plate format. Repeat the above steps with a non-existent number plate. Verify that an appropriate	should be displayed for invalid number plate format. An appropriate message should be displayed stating that the number plate was not found for a non-existent number plate.	are displayed for an invalid number plate format and non-existent number plates. The test case fails if the system does not display the details of the searched vehicle for a valid number plate or does not display an appropriate error message for an invalid number plate format or non-existent number plates.

		message is displayed stating that the number plate was not found.		

8.2.3 TEST CASES (SEARCH VEHICLE OWNER USING CNIC)

TEST CASE	PRE-CONDITION	TEST STEP	EXPECTED RESULT	SUCCESS CRITERIA
verify that the system can accurately display the owner details of a vehicle using their CNIC number.	The system is installed and configured properly. The user has valid login credentials to access the system. The owner of the vehicle whose details are to be searched has a valid CNIC number registered in the system.	Log in to the system with valid login credentials. Go to the "Search Vehicle Owner" option. Enter a valid CNIC number in the search field. Click on the "Search" button. Verify that the owner details of the searched CNIC number are accurately displayed. Repeat the above steps with an	The system should accurately display the owner details of the searched CNIC number. An appropriate error message should be displayed for an invalid CNIC number.	The test case passes if the system accurately displays the owner details of the searched CNIC number and an appropriate error message is displayed for an invalid CNIC number. The test case fails if the system does not display the owner details accurately or does not display an appropriate error message for an invalid CNIC number.

		invalid CNIC number. Verify that an appropriate error message is displayed for an invalid CNIC number.		

8.2.4 TEST CASES (SEARCH VEHICLE LOCATION USING MAP)

TEST CASE	PRE-CONDITION	TEST STEP	EXPECTED RESULT	SUCCESS CRITERIA
To verify that the system can accurately display the location of a vehicle on a map.	The system is installed and configured properly. The user has valid login credentials to access the system. The vehicle whose location is to be searched has a RFID chip installed and is registered to the system.	Log in to the system with valid login credentials. Go to the "Search Vehicle option" page. Enter a valid vehicle number plate in the search field. Click on the "Search" button. Verify that the location of the searched vehicle is accurately displayed on a map.	The system should accurately display the location of the searched vehicle on a map for a valid number plate. An appropriate error message should be displayed for an invalid number plate.	The test case passes if the system accurately displays the location of the searched vehicle on a map for a valid number plate and an appropriate error message is displayed for an invalid number plate. The test case fails if the system does not display the location of the searched vehicle accurately or does not display an

		Repeat the above steps with an invalid number plate. Verify that an appropriate error message is displayed for an invalid number plate.		appropriate error message for an invalid number plate.

8.2.5 TEST CASES (VIEW ALL VEHICLES LIST)

TEST CASE	PRE-CONDITION	TEST STEP	EXPECTED RESULT	SUCCESS CRITERIA
To verify that the system can accurately display the list of all vehicles to both the user and the admin.	The system is installed and configured properly. The user has valid login credentials to access the system. The admin has valid login credentials to access the system. There are multiple vehicles registered in the system.	Log in to the system with valid login credentials. Go to the "View All Vehicles" option. Verify that the list of all vehicles is accurately displayed. Verify that the user can only view the list and cannot edit or delete any vehicle details.	The system should accurately display the list of all vehicles to both the user and the admin. The user and admin should able to view the list and vehicle details.	The test case passes if the system accurately displays the list of all vehicles to both the user and the admin. The test case fails if the system does not display the list of all vehicles accurately.

8.2.6 TEST CASES (REGISTER A VEHICLE)

TEST CASE	PRE-CONDITION	TEST STEP	EXPECTED RESULT	SUCCESS CRITERIA
To verify that the system can accurately register a new vehicle by providing the name, CNIC, engine number, number plate, and RFID number.	The system is installed and configured properly. The user has valid login credentials to access the system. The user has the necessary information, including the name, CNIC, engine number, number plate, and RFID number, for registering a new vehicle.	Log in to the system with valid login credentials. Navigate to the "Register a Vehicle" page. Enter the name, CNIC, engine number, number plate, and RFID number of the new vehicle. Click on the "Register" button. Verify that the new vehicle is successfully registered and the system displays a success message. Navigate to the "View All Vehicles" page and verify that the new vehicle is accurately displayed.	The system should successfully register the new vehicle and display a success message. The new vehicle should be accurately displayed on the "View All Vehicles" page. An appropriate error message should be displayed for invalid or incomplete information.	The test case passes if the system successfully registers the new vehicle, accurately displays it on the "View All Vehicles" page, and displays an appropriate error message for invalid or incomplete information. The test case fails if the system fails to register the new vehicle, displays incorrect information, or does not display an appropriate error message for invalid or incomplete information.

		Repeat the above steps with invalid or incomplete information. Verify that the system displays an appropriate error message for invalid or incomplete information.		

8.2.7 TEST CASES (ADD NEW USER)

TEST CASE	PRE-CONDITION	TEST STEP	EXPECTED RESULT	SUCCESS CRITERIA
To verify that the system can accurately add a new user to the system by the admin.	The system is installed and configured properly. The admin has valid login credentials to access the system. The admin has the necessary information, including name, email, and password, for adding a new user to the system.	Log in to the system with valid admin login credentials. Navigate to the "Add New User" page. Enter the necessary information, including name, email, and password, of the new user. Click on the "Add User" button.	The system should successfully add the new user and display a success message. An appropriate error message should be displayed for invalid or incomplete information.	The test case passes if the system successfully adds the new user and displays a success message. An appropriate error message should be displayed for invalid or incomplete information. The test case fails if the system fails to add the new user, displays incorrect

		Verify that the new user is successfully added to the system and the system displays a success message. Repeat the above steps with invalid or incomplete information. Verify that the system displays an appropriate error message for invalid or incomplete information.		information, or does not display an appropriate error message for invalid or incomplete information.

8.3 BLACK BOX TESTING

Black box testing is a technique used in software testing to check the functionality of a software application without knowing how it works internally. It is done by testers who only focus on making sure that the software meets its intended functional requirements. To achieve this, testers use different techniques such as equivalence partitioning, boundary value analysis, and decision table testing. The main objective of black box testing is to ensure that the software is reliable, robust, and meets the needs of its users by checking its behavior without any knowledge of its internal workings.

8.3.1 TEST CASES

Case	Input	Expected Output	Actual Output	Status
Verify that a user can register and login to their account with all the required information.	Navigate to the user registration page. Fill in all the required fields, including name, email, password, and contact information. Click the "Register" button. Verify that the system displays a success message confirming that the user account has been created. Navigate to the user login page. Enter the email and password used during registration. Click the "Login" button. Verify that the system redirects the user to their	The user is able to successfully register and login to their account with all the required information.	The user is able to successfully register and login to their account with all the required information.	Valid

	dashboard page.			
Verify that users can register their vehicles with all the required information.	Navigate to the vehicle registration option. Fill in all the required fields, Click the "Register Vehicle" button. Verify that the system displays a success message confirming that the vehicle has been registered.	The user is able to successfully register their vehicle with all the required information.	The user is able to successfully register their vehicle with all the required information.	Valid
Verify that searching for a vehicle using number plate is working as intended, and the system returns the correct vehicle information.	Navigate to the search option. Enter the number plate of the vehicle. Click the "Search" button. Verify that the system returns the correct vehicle information, including the name, CNIC, engine number, number plate, and RFID number.	The system should return the correct vehicle information for the given search criteria.	The system returns the correct vehicle information for the given search criteria.	Valid

Verify that adding new users to the system by an admin is working as intended, and admins can create new user accounts with all the required information.	Navigate to the user management page. Click the "Add New User" option. Fill in all the required fields, including the name, email, username, password. Click the "Create User" button. Verify that the system displays a success message confirming that the user has been created.	The admin should be able to create a new user account with all the required information.	The admin is able to create a new user account with all the required information.	Valid
Verify that users and admins can view the list of all registered vehicles in the system.	Navigate to the vehicle list option. Verify that the system displays a list of all registered vehicles in the system, including their name, CNIC, engine number, number plate, and RFID number.	The system should display a list of all registered vehicles in the system, and users and admins should be able to view accurate and up-to-date information for each vehicle.	The system displays a list of all registered vehicles in the system, and users and admins can view accurate and up-to-date information for each vehicle.	Valid

	Check that the list is properly sorted and searchable. Verify that the system displays accurate and up-to-date information for each vehicle.			

CHAPTER 9
CONCLUSION

9.1 INTRODUCTION

A Vehicle Detection System (VDS) is a technology used to detect the presence of vehicles entering and exiting a particular location. VDS can use various technologies such as Radio Frequency Identification (RFID), ultrasonic sensors, infrared sensors, or magnetic sensors to detect the vehicles.

In the case of RFID-based VDS, a passive RFID tag is attached to the vehicle, and an RFID reader is installed at the entry and exit points of the location. As the vehicle passes through the reader, the RFID tag transmits a unique identifier to the reader, which is then processed by the VDS software.

The VDS software can then track the movement of the vehicle, providing real-time information about the vehicle's entry and exit times. The system can also generate reports on vehicle traffic patterns, parking availability, and revenue generation.

9.2 CONCLUSION

a Vehicle Detection System (VDS) with RFID technology is an innovative solution that can significantly improve the efficiency of traffic management and vehicle tracking. With its ability to automatically detect and identify vehicles using RFID tags, the VDS can provide accurate and real-time information about vehicle movements, entry and exit times, and parking availability.

By implementing the VDS, organizations such as traffic police, parking management authorities, and transportation agencies can optimize their operations, reduce congestion, and enhance security. The system is also user-friendly, with a simple and intuitive graphical user interface that makes it easy for operators to monitor and manage the system.

Furthermore, the VDS is a cost-effective solution that requires minimal maintenance and can be easily integrated with other systems. With its high reliability and accuracy, the VDS can provide a significant return on investment and contribute to the overall success of an organization.

Overall, the VDS with RFID technology is a powerful tool that can improve the efficiency and effectiveness of traffic management and vehicle tracking. Its numerous benefits make it an essential system for any organization that wants to optimize its operations and enhance its services.

CHAPTER 10
FUTURE WORK

10.1 INTRODUCTION

VDS with RFID is the project the can be further modify in future like we can also use this system in parking areas parking system uses RFID technology to monitor parking facilities and provide real-time information about available parking spaces. It can also be used to enforce parking regulations and improve parking management. Similar it can be used to detect who didn't pay the tax.

10.2 FUTURE ENHANCEMENT

Mobile app integration: The VDS software could be integrated with a mobile app, allowing users to view real-time information about vehicle traffic patterns and parking availability from their smartphones.

Machine learning-based analytics: By leveraging machine learning algorithms, the VDS software could provide more accurate and insightful data analysis, such as predicting peak traffic periods and optimizing parking allocation.

Multi-site integration: The VDS system could be expanded to cover multiple sites, providing a centralized view of vehicle traffic across multiple locations.

Integration with smart city systems: VDS could be integrated with smart city systems, allowing for better coordination of traffic flow and optimizing overall city operations.

Integration with autonomous vehicles: As autonomous vehicles become more common, VDS could be modified to support automatic detection and tracking of these vehicles.

Overall, these enhancements could provide even greater benefits in terms of traffic management, security, and revenue generation, making VDS with RFID technology an even more valuable investment for organizations in the future.

CHAPTER 11
ACHIEVEMENT

11.1 INTRODUCTION

In this chapter we will discuss the skill experience that we gain by completing such project.

11.2 ACTIVITIES

11.2.1 PROJECT PLANNING

This involves defining the project goals, objectives, scope, and timeline, as well as identifying the resources required and the risks involved. Which we have done at the start of the project.

11.2.2 SYSTEM ANALYSIS AND DESIGN

This involves analyzing the requirements of the VDS and designing the system architecture, components, and functionalities. Also include selecting the appropriate RFID hardware and software, database management, and user interface design. We had several meeting with our supervisor to select appropriate hardware

11.2.3 SYSTEM IMPLEMENTATION

This involves developing and coding the software, hardware components, and integrating them into a working system. All of our team members sit together to work on the project and to overcome the problems

11.2.4 SYSTEM TESTING AND VALIDATION

This involves testing the VDS system to ensure that it meets the specified requirements and functions as intended. All of our members discuss the project to make case studies and test cases accordingly

11.2.5 DOCUMENTATION

This involves creating documentation, such as project plans, technical specifications, user manuals, and system documentation, to support the development and implementation of the VDS. The documentation had place from the very first day of the project.

11.2.6 PRESENTATION AND DEMONSTRATION

This involves presenting the VDS project to the project stakeholders, demonstrating the functionality of the system, and obtaining feedback on the project outcomes. We demonstrate our working in very milestone.

11.3 IMPROVEMENT

11.3.1 TECHNICAL SKILLS

By completing a VDS project improve experience in designing and developing complex software and hardware systems, which will help us to develop technical skills in areas such as RFID technology, database management, and system integration.

11.3.2 PROBLEM-SOLVING ABILITIES

Developing a VDS project requires a significant amount of problem-solving and critical thinking skills, which students can apply to other real-world scenarios and challenges. By completing VDS project our skills of Problem solving has improved

11.3.3 INDUSTRY-RELEVANT EXPERIENCE

The VDS project is a practical and industry-relevant project that prepares students for future career opportunities in areas such as transportation, logistics, and security.

11.3.4 TEAMWORK AND COMMUNICATION

By completing VDS project we gain valuable experience working in a team and communicating effectively with team members, stakeholders, and clients.

11.3.5 ACADEMIC RECOGNITION

Completing a VDS project also provide us with academic recognition in the form of course credits, grades, and awards, which can enhance our academic and professional credentials.

APPENDIX A

EXCISE, TAXATION AND NARCOTICS CONTROL DEPARTMENT

A Vehicle Detection System (VDS) with RFID technology can be helpful for the Excise, Taxation and Narcotics Control Department in Pakistan in several ways:

Vehicle Registration: VDS with RFID technology can help the department manage vehicle registration by automating the process of capturing vehicle data, such as make, model, and registration number. This can help the department maintain accurate records of registered vehicles and reduce the risk of fraud.

Vehicle Taxation: VDS with RFID technology can help the department monitor vehicle usage and calculate taxes based on distance traveled or fuel consumption. This can help the department collect taxes more efficiently and reduce the risk of tax evasion.

Anti-Smuggling: VDS with RFID technology can be used to track the movement of vehicles suspected of smuggling illicit goods, such as drugs or contraband. This can help the department identify and intercept illegal activity more effectively.

Security and Surveillance: VDS with RFID technology can be used for security and surveillance purposes, such as monitoring the movement of vehicles near sensitive locations or tracking the movement of suspected criminals. This can help the department improve public safety and reduce crime.

Overall, a Vehicle Detection System with RFID technology can help the Excise, Taxation and Narcotics Control Department in Pakistan manage vehicle registration and taxation more efficiently, reduce the risk of fraud and tax evasion, and improve security and surveillance.

Real-Time Traffic Monitoring: VDS with RFID technology can provide real-time data on vehicle movement, which can help the traffic police department monitor traffic flow and identify congestion hotspots. This can help them make informed decisions about traffic management and deploy resources more effectively.

Automated Traffic Violation Detection: VDS with RFID technology can be integrated with automated traffic violation detection systems, such as speed cameras and red light cameras, to

identify and penalize traffic violators. This can help the traffic police department enforce traffic laws and improve road safety.

Parking Management: VDS with RFID technology can be used to manage parking facilities, such as public parking lots and street parking zones, by tracking the number of vehicles entering and leaving the facility and enforcing parking regulations. This can help the traffic police department regulate parking and reduce traffic congestion caused by illegal parking.

Emergency Response: VDS with RFID technology can help the traffic police department respond more quickly to emergencies, such as accidents and road closures, by providing real-time data on traffic flow and congestion. This can help them divert traffic away from affected areas and minimize disruption to the public.

APPENDIX B
ASSET TRACKING SYSTEM

Asset Management: An ATS can help organizations manage their assets more efficiently by providing real-time information about their location and status. This can help reduce the risk of theft or loss and improve asset utilization.

Inventory Management: An ATS can be used to track inventory in real-time, allowing organizations to optimize their inventory levels and reduce waste.

Maintenance Management: An ATS can be used to track the maintenance history of assets, such as equipment, and schedule preventive maintenance. This can help organizations reduce downtime and extend the lifespan of their assets.

Compliance Management: An ATS can help organizations comply with regulations related to asset tracking and management, such as those related to environmental or safety standards.

APPENDIX C
HARDWARE PACIFICATION

RFID Tags: Passive RFID tags can be attached to the vehicles, and active RFID tags can be installed in the vehicles. These tags contain a unique identifier that can be read by RFID readers.

RFID Readers: RFID readers can be installed at the entry and exit points of the parking facility or on the gantries of the highway. These readers can communicate with the RFID tags and collect data about the location and movement of the vehicles.

Antennas: RFID antennas are used to transmit and receive signals between the RFID readers and tags. They can be installed in various locations to ensure maximum coverage of the facility or highway.

Control Unit: The control unit is used to manage and process the data collected by the RFID readers. It can be a standalone device or integrated with a software application.

Software: The software application is used to manage and analyze the data collected by the RFID readers. It can provide real-time information about the location and status of vehicles and generate reports for parking management or traffic control.

Network Infrastructure: A network infrastructure is required to connect the RFID readers, control unit, and software application. This can be a wired or wireless network, depending on the location and size of the facility or highway.

APPENDIX D
DESIGN DOCUMENTS

System Design Document: This document provides an overview of the system architecture, components, and interfaces. It includes a detailed description of the hardware and software components required for the VDS, such as RFID tags, readers, antennas, control unit, software, and network infrastructure. It also outlines the data flow and communication protocols between the components.

User Interface Design Document: This document describes the graphical user interface (GUI) of the VDS software. It includes wireframes, mockups, and user interface specifications. The document defines the layout, navigation, and functionality of the GUI.

Database Design Document: This document outlines the database schema and data model used by the VDS software. It includes a description of the database tables, fields, and relationships. The document also defines the data validation rules and constraints.

Network Design Document: This document outlines the network infrastructure required for the VDS. It includes a description of the network topology, devices, and protocols. The document also defines the security policies and access controls.

Integration Design Document: This document describes the integration of the VDS with other systems, such as a parking management system or traffic control system. It includes a description of the data exchange protocols, data format, and data transformation rules.

Test Plan Document: This document outlines the testing strategy and test cases for the VDS. It includes a description of the testing environment, test data, and test procedures. The document also defines the acceptance criteria and quality metrics.

REFERENCES

[1] (1566) IVIS RFiD LIcense Plate - YouTube

[2] HOME - RFID PAKISTAN

[3] RFID vehicle tracking system, Vehicle asset tracking | RAMP (ramprfid.com)

[4] A study of RFID adoption for vehicle tracking in a container terminal (upc.edu)

[5] RFID Tracking System for Vehicles (RTSV) (researchgate.net)

[6] https://www.researchgate.net/profile/Maruf-Islam/publication/322516489_Smart_anti-theft_vehicle_tracking_system_for_Bangladesh_based_on_Internet_of_Things/links/5c93134145851506d71f7728/Smart-anti-theft-vehicle-tracking-system-for-Bangladesh-based-on-Internet-of-Things.pdf

[7] https://www.asiarfid.com/what-is-rfid.html

[8] https://www.sparkfun.com/rfid

[9] https://lastminuteengineers.com/how-rfid-works-rc522-arduino-tutorial/

[10] Fareastsun 1-3meters Long Range Scanner Impinj R2000 Chip Handheld Wireless Portable Uhf Rfid Reader To Connect Android Phone - Buy Bluetooth Rfid Reader,Rfid Reader To Android,Uhf Rfid Handheld Reader Product on Alibaba.com

Turnitin Originality Report

Processed on: 15-Jan-2023 18:10 PKT
ID: 1992324011
Word Count: 7850
Submitted: 2

report By Alisha Hameed

Similarity Index	Similarity by Source	
15%	Internet Sources:	10%
	Publications:	5%
	Student Papers:	8%

2% match (Internet from 09-Mar-2016)
http://louisdl.louislibraries.org/cdm/compoundobject/collection/p120701coll25/id/101780/rec/11

1% match (student papers from 11-Jan-2023)
Submitted to Higher Education Commission Pakistan on 2023-01-11

1% match (Internet from 29-May-2022)
https://www.campcodes.com/projects/php/fleet-management-system-with-live-gps-tracking/

1% match (Internet from 05-Mar-2014)
http://www.infonomics-society.org/IJRFIDSC/IJRFIDSC_Paper%201.pdf

1% match (Internet from 20-Nov-2022)
https://worldwidescience.org/topicpages/f/field+uhf+rfid.html

< 1% match (student papers from 17-Apr-2018)
Submitted to Higher Education Commission Pakistan on 2018-04-17

< 1% match (Internet from 20-Dec-2022)
https://www.researchgate.net/publication/269309084_Design_and_implementation_of_vehicle_tracking_system_using_GPSGSMGPRS_technology_and