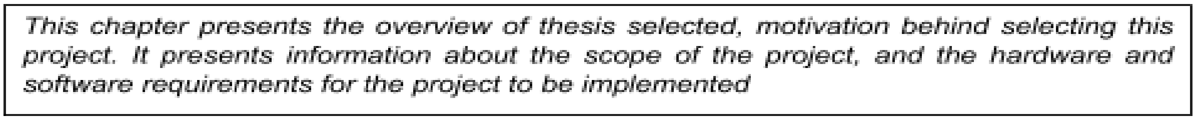
Chapter 1

Introduction



This chapter presents the overview of thesis selected, motivation behind selecting this project. It presents information about the scope of the project, and the hardware and software requirements for the project to be implemented

1.1 Overview of the system

The major pillar of any country is the transportation system which is used for transporting the goods for one place to another.This is done through air,water and roadways but our system focuses on only transportation through road. Goods transportation System needs to follow a systematic procedure for booking any goods transportation vehicle. A major issue which the goods transportation system suffers from this the movement of under capacity or unloaded truck moving places due to lack of orders or booking. Regardless the high demands in the market for transporting the goods, goods transportation system still suffers from lack of bookings.

Another problem that arises in the goods transportation system is that some goods gets damaged while its transportation due to the pollution and the climatic condition of certain routes. There are variety of goods which are transported, not all goods are suitable to certain climatic conditions.

Another problem which occurs in the current goods transportation system is the security system while the booking for any transportation vehicle.

To solve these problems, we are going to present an effective solution. We will present an application which will be help to book one of our various vehicle types. As this is an on-demand system, our servers will locate the drivers nearest to the customer and assign him to the customer. Thus, no vehicles will go under-capacity.

We will also find the least polluted route from the customer's source to the destination using time-series analysis of data. Thus, the customer will have the option of choosing from the various routes we present to him.

Five applications will be developed-Customer Application, Agent Application, Truck Driver Application, Truck Owner Application and Warehouse Owner Application. When the customer will book a vehicle, using his app, a request will be sent to the nearest driver. If he accepts this and arrives at the customer's location, the customer will have to enter an OTP as a form of authentication. This OTP will be encrypted at the server side using Elliptical Curve Cryptography and then sent to the customer's application where it will be decrypted. The driver will enter this OTP and it will then be encrypted and sent to the server where this password will be decrypted and verified, thus making the ride secure.

1.2 Scope

The scope of the project can be divided into two sections

The first section being the design and development of a small physical prototype for the Smart Goods Transportation system. We will be demonstrating how our system is aimed at working on our campus by creating prototypes of IOT devices and placing it at different parts of our campus. Each device will represent one of our applications-Customer, Driver, Truck Owner or Agent. If the customer books a cab, data will be sent through our servers to the nearest driver on our campus grounds. When this request will be accepted , a map will be displayed on the driver's IOT device which he will follow to the customer's location. To start the trip, the customer will be required to give the driver an OTP, which will be encrypted from the server and sent immediately. The driver will input and this OTP will then be verified.

The second section includes the design and development of a simulation software using time series analysis to find least polluted route. In this system, given the source and destination, the least polluted route will be found between them, based on the data that is collected previously.

The simulated software will show the various possible routes between the source and the destination with the different pollution levels and costs. The user will have the option of selecting one of these routes.

1.3 Motivation

The current system in India has three modes for goods transportation i.e by railways, airways, waterways and roadways. We are going to focus on the roadways of goods transportation. On studying the current goods transportation in India we have noticed that the goods transportation suffers various problems. The problem of unavailability of the vehicle or the possibility of empty or under-capacity vehicles moving to places due to lack of information of availability of the vehicles at the time of booking. Also there is possibility of not considering the shortest route or the route which is less polluted which will further help to preserve the goods that can be affected due to the climate condition.

Hence to solve this problem of Goods Transportation System there should be an application which would help the trucks owners as well as the customers to easily get the information to avoid empty trips or idle vehicles to increase productivity and to make the goods transport economical from all the users. Thus applying automation to the transport system.

1.4 Hardware and Software Requirements

Hardware Requirements for development:

- A computer/laptop for coding (android and Arduino uno Board)

  Minimum requirements to be met:

    - Intel i5 processor
  - 8 GB Ram
  - Running Windows 10
- Android phone for development and testing purposes
  - Running Android 5.0 +
  - 2 GB RAM

- Arduino Uno Wifi
- Sensors(MQ135)

Software Requirements for development:

- Operating System: Windows 10
- Android studio for app development
- Firebase Cloud to support android application.
- Arduino IDE for pushing code into board.
- Jupyter Notebook for Python development
- Flask Framework for website.
- Thingspeak cloud for storing data.

Hardware Requirements for implementation:

- A 2-3 mobile phones to run the applications
- Minimum requirements to be met:
    - Android version 5.0 and Above
    - 4 GB Ram
    - Internet connection

User Interfaces

Users are going to communicate through their respective android application. Information displayed will be:

- Drivers will see the route to customer and then the route to customer's destination.
- Customer will see the location of the selected vehicles, time and route for destination. Also it must have information about the driver.
- Vehicle owner's can see their respective vehicle location.
- Customer can track the vehicle until it reaches the destination.

-

Thus in chapter 1, which is the Introduction of the project, we have clearly defined the problem definition of the project, in which the issues and faults of current system are mentioned. Furthermore, pertaining to these problems, this thesis mentions the scope of the project and mentioning a clear way how we plan to solve the problems mentioned in the problem definition. We have also mentioned the motivation behind building this application. The scope includes exact details of what is to be implemented. Lastly, chapter 1 ends with mentioning the hardware and software requirements of the project with respect to both development and implementation of the project.

In the next chapter, that is Literature Survey, various research papers have been studied and analyzed and the analysis have been mentioned.

Chapter 2

Literature Survey

This chapter presents the research papers studied to gain knowledge about the current goods transportation system and how the smart goods transportation can be implemented to overcome the problems faced by current system.

A variety of research papers were studied to implement a solution to the above problem. The first research paper “Transactive control in Smart Cities” mainly focuses on improving the quality of the urban mobility. It shows how dynamic toll pricing and mobility on demand helps in transactive control. This paper introduces to the concept of dynamic routing for multi-customer transportation.

The second research paper “An Architecture of Smart Transportation System using modified RR algorithm and VANET” provides the methods for efficient traffic control systems considering the various factors affecting the traffic. In this paper, Vehicular Ad-hoc Network (VANET)architecture for real time traffic information exchange from one traffic signal to another traffic signal via vehicle is proposed.

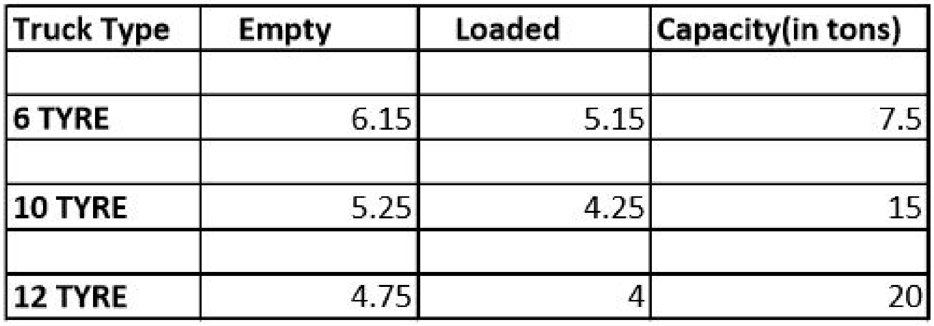
The third research paper “Smart City Implementation Models Based on IoT Technology” IoT (Internet of Things) is the network of physical objects-devices, vehicles, buildings and other items embedded with electronics, software, sensors, and network connectivity-that enables these objects to collect and exchange data. The internet of things allows objects to be sensed and controlled remotely across existing network infrastructure. 260 million objects will be connected by year 2020. This paper summarizes the tangible IoT based service models which are helpful to academic and industrial world to understand IoT business.

The fourth research paper “Dynamic route planning framework for minimal air pollution exposure in urban road transportation systems” this paper gives us information about how the shortest with least pollution can be used and what all factors needs to be considered for the same. This paper focuses on the reduction of traffic at the junctions.

There are various websites from where we have obtained information about the pricing systems for the different types of vehicle. While calculating trip costs real time diesel price, mileage, distance and total load of the vehicle will be considered.

ECC cryptography will be used to encrypt the OTP which will be sent by the backend software to the customer. ECC is a public key based , such as RSA , but it is sort of represented in an algebraic structure , ECC offers the same security than RSA but at a smaller footprint , also it’s less cpu intensive so it’s ideal for mobile devices and faster acting networks

- **Truck mileage as per amount of goods loaded.**

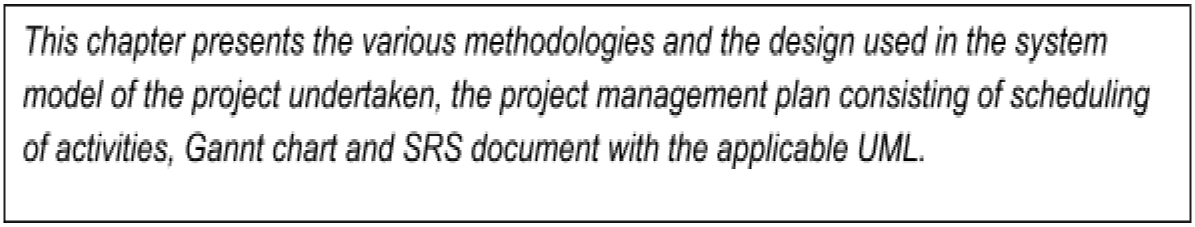


Truck Type	Empty	Loaded	Capacity(in tons)
6 TYRE	6.15	5.15	7.5
10 TYRE	5.25	4.25	15
12 TYRE	4.75	4	20

Figure 2.1 Mileage Diesel per goods loaded.

Chapter 3

Methodology



This chapter presents the various methodologies and the design used in the system model of the project undertaken, the project management plan consisting of scheduling of activities, Gannt chart and SRS document with the applicable UML.

3.1 Programming Languages Used :

1.Our apps were generated using the programming language of Java and were created in Android Studio.

-Elliptical Curve Cryptography, which is used for transmission of OTP was also coded in Java.

2.All our Arduino models that were used to collect pollution data through its various sensors were programmed in Arduino IDE using the Arduino Programming Language.

3.Time Series Analysis using the SARIMA model was also coded in Python.

4. The visualization of Time Series Analysis via website is done using Flask framework.

3.2 Other Facilities Used:

1.The data from all our apps in stored in Google's cloud server-Firebase[7]. This acts as our backend server.

2.The sign-in feature in all our apps uses the authentication provided from Firebase.

3.3 Proposed System Model

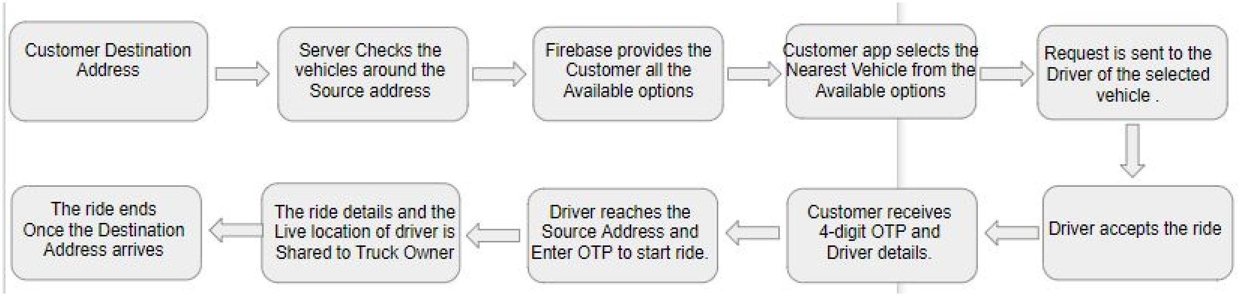

Customer Destination Address
Server Checks the vehicles around the Source address
Firebase provides the Customer all the Available options
Customer app selects the Nearest Vehicle from the Available options
Request is sent to the Driver of the selected vehicle .
Driver accepts the ride
Customer receives 4-digit OTP and Driver details.
Driver reaches the Source Address and Enter OTP to start ride.
The ride details and the Live location of driver is Shared to Truck Owner
The ride ends Once the Destination Address arrives

Figure 3.3.1 Proposed System Model

The working of the proposed system is as follows:

- A customer would book a vehicle and the request would be made to the driver application.
- The customer application looks for any vehicle near to his location. From the list of available vehicles the customer can select any type of vehicle. These data would be presented to customer by the reference of the data stored in Firebase.
- The customer select type of vehicle and clicks on book ride.
- The request would be sent to the nearest driver application. The driver can then either accept or reject the booking.
- Once the driver accepts the ride he is navigated to the source address and driver details are sent to the firebase cloud.
- The firebase will forward the driver details to the customer.
- The 4 digit OTP which will be displayed in the customer application which will be stored on the firebase in json format.
- After the driver reaches the source address the customer conveys the 4 digit OTP verbally to the driver. The driver enters the OTP and the entered OTP will be verified to the OTP stored.
- . If the code matches then the trip starts successfully.
- The driver will now get the destination address and would be navigated to the same. The Customer application,

- Vehicle owner application would get the live status of the truck.Customer and Driver app will be notified once the journey ends thus displaying the cost o.

The working of the Elliptical Curve Cryptography is as follows:

- The points satisfying the curve selected are generated.
- Randomly 10 points from the point generated are selected and then are mapped to the numbers 0-9 thus generating mapping table.
- The prime number is randomly selected.
- 4 digit OTP is randomly generated.
- A number n is selected and private key is selected randomly between 1,n-1.
- Each digit is extracted from 4 digit OTP and the point on the curve which is mapped to the digit is extracted and used for encryption.
- The cipher text, mapping table and point Q which is calculated using private key and local point is sent to the client.
- Using the above values the client decrypts the 4 digit OTP.

The process of collecting data on ThingSpeak is as follows:

- The module is made using Arduino Uno WiFi and MQ135 sensor.
- Using Arduino IDE the code is pushed to the Arduino board.
- The code consist of three parts:
  - Connecting to the WiFi.
  - Reading the sensed data using sensor.
  - Writing the sensed data to the ThingSpeak.
- In order to connect to the WiFi ESP8266WiFi, WiFiClient libraries are imported. Providing ssid and password of the network the board is connected to WiFi using WiFi.begin(ssid,password) function.
- The data is read into variable airpollution by using the analogread(A0), where A0 is the pin on the board at which MQ135 sensor is connected.
- In order to connect WiFi client to ThingSpeak server ThingSpeak library is imported. On ThingSpeak server channel is created. Using ThingSpeak.writeField(myChannelNumber,1 , airpollution, myWriteAPIKey); function data is uploaded to ThingSpeak channel. myChannelNumber and myWriteApiKey is the unique channel number assigned to every channel created.
- Arduino is supplied with power to upload data

The working of the SARIMA model for predicting pollution is as follows:

- The data of air pollution is collected using the Arduino Uno WiFi modules. MQ135 Gas sensor is used in order to get value of the air pollution.
- The modules were placed in Thane, Vidyavihar and Sion and data was collected on Thingspeak which is a cloud platform. The AQI was added to the database every 15 seconds.
- The python code is programmed which takes current time as input extracts data of that hour and gives the pollution value accordingly.

The cost of trip depends on the following factors:

1. Diesel Price
2. Mileage of the Vehicle
3. Distance between source and destination
4. Toll prices
5. Cost of vehicle according to the type of goods
6. Commission Percentage
7. Insurance

Formula:

Cost of trip = (Distance/Mileage)*Diesel Price + Cost of vehicle + Toll prices + Insurance

Total amount = Cost of trip+(Cost of trip * Commission percent)

3.4 Software Project Management Plan

3.4.1 Activity Plan

ID	**Activity**	**Duration (Weeks)**	**Predecessor**	**Start Date**	**Finish Date**

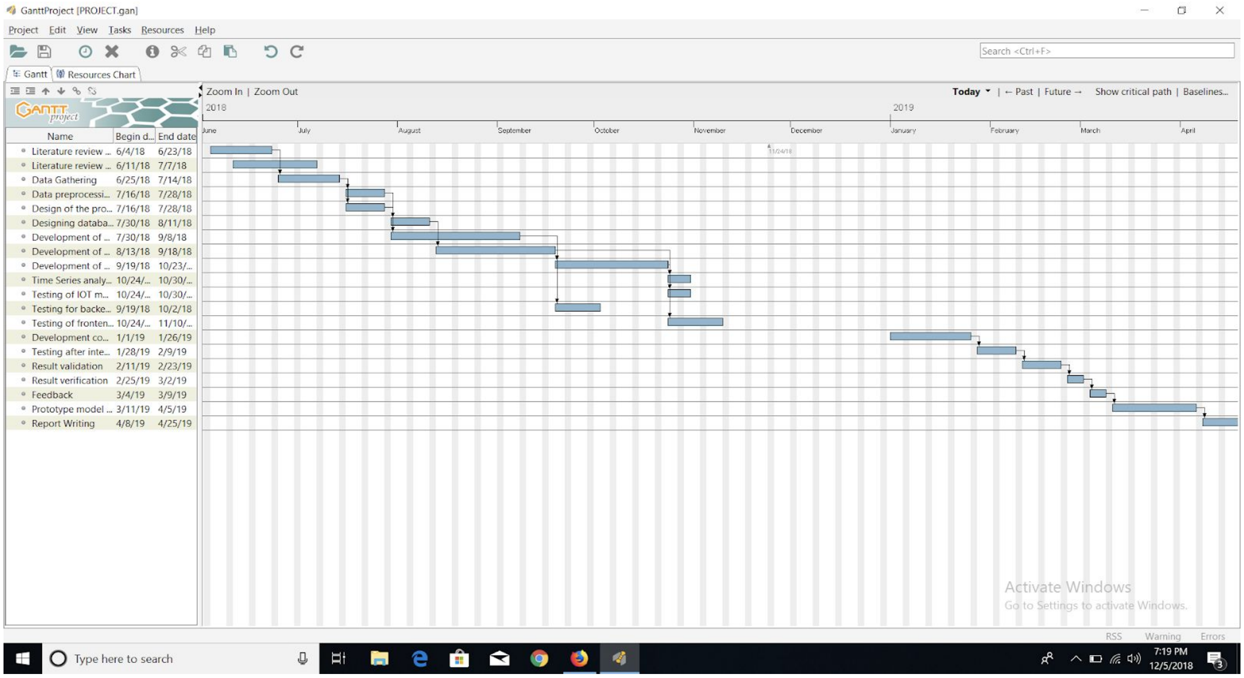
1	Literature review for the scope refinement of the problem statement	3	-	4-6-18	23-6-18
2	Literature review for finding methodology for prototype designing	4	-	11-6-18	7-7-18
3	Data Gathering	3	1	25-6-18	14-7-18
4	Data preprocessing	2	3	16-7-18	28-7-18
5	Design of the prototype	2	3	16-7-18	28-7-18
6	Designing database of time series data	2	4	30-7-18	11-8-18
7	Development of prototype(Iot module)	5	5	30-7-18	8-9-18
8	Development of prototype(Backend software)	5	6	13-8-18	18-9-18
9	Development of prototype(Frontend software)	4	7,8	10-9-18	13-10-18
10	Time Series analysis	1	8,9	15-10-18	20-10-18
11	Testing of IOT module	1	8,9	15-10-18	20-10-18
12	Testing for backend software	2	8	22-10-18	3-11-18
13	Testing of frontend software	1	9	5-11-18	10-11-18
14	Development complete integration of iot, backend, frontend and software	4		1-1-19	26-1-19
15	Testing after integration	2	14	28-1-19	9-2-19
16	Result validation	2	15	11-2-19	23-2-19
17	Result verification	1	16	25-2-19	2-3-19

18	Feedback	1	17	4-3-19	9-3-19
19	Prototype model correction and testing	2	18	25-3-19	5-4-19
20	Report Writing	2	19	11-4-19	25-4-19

Table 1. Activity Plan

The pictures above depict the software management plan for the project. The first picture contains the division of project into activities and the duration, early start, early finish, late start and late finish of every activity is written in a tabular format. Next two images depict the start and the end dates (tentative planning for each activity) .Next comes the gantt chart showing dependencies among various activities and the timeline for the project.

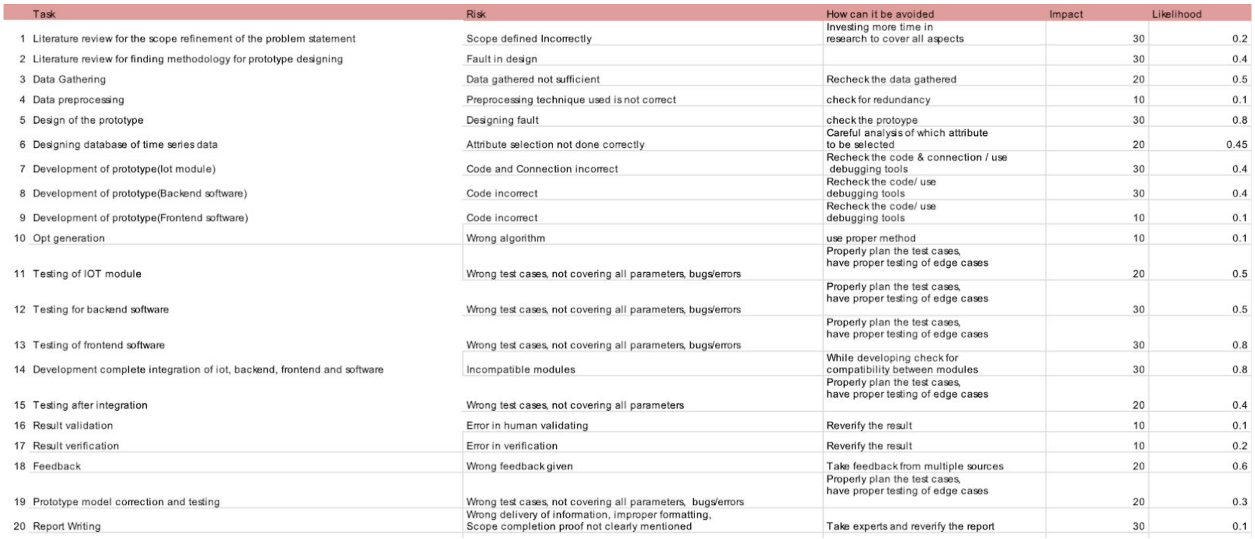
Gantt chart



Name	Begin d...	End date
Literature review ...	6/4/18	6/23/18
Literature review ...	6/11/18	7/7/18
Data Gathering	6/25/18	7/14/18
Data preprocessi...	7/16/18	7/28/18
Design of the pro...	7/16/18	7/28/18
Designing databa...	7/30/18	8/11/18
Development of ...	7/30/18	9/8/18
Development of ...	8/13/18	9/18/18
Development of ...	9/19/18	10/23/...
Time Series analy...	10/24/...	10/30/...
Testing of IOT m...	10/24/...	10/30/...
Testing for backe...	9/19/18	10/2/18
Testing of fronten...	10/24/...	11/10/...
Development co...	1/1/19	1/26/19
Testing after inte...	1/28/19	2/9/19
Result validation	2/11/19	2/23/19
Result verification	2/25/19	3/2/19
Feedback	3/4/19	3/9/19
Prototype model ...	3/11/19	4/5/19
Report Writing	4/8/19	4/25/19

Figure 3.4.1 Gantt Chart

3.4.2 Activity Risk Analysis Table



	Task	Risk	How can it be avoided	Impact	Likelihood
1	Literature review for the scope refinement of the problem statement	Scope defined Incorrectly	Investing more time in research to cover all aspects	30	0.2
2	Literature review for finding methodology for prototype designing	Fault in design		30	0.4
3	Data Gathering	Data gathered not sufficient	Recheck the data gathered	20	0.5
4	Data preprocessing	Preprocessing technique used is not correct	check for redundancy	10	0.1
5	Design of the prototype	Designing fault	check the protoype	30	0.8
6	Designing database of time series data	Attribute selection not done correctly	Careful analysis of which attribute to be selected	20	0.45
7	Development of prototype(Iot module)	Code and Connection incorrect	Recheck the code & connection / use debugging tools	30	0.4
8	Development of prototype(Backend software)	Code incorrect	Recheck the code/ use debugging tools	30	0.4
9	Development of prototype(Frontend software)	Code incorrect	Recheck the code/ use debugging tools	10	0.1
10	Opt generation	Wrong algorithm	use proper method	10	0.1
11	Testing of IOT module	Wrong test cases, not covering all parameters, bugs/errors	Properly plan the test cases, have proper testing of edge cases	20	0.5
12	Testing for backend software	Wrong test cases, not covering all parameters, bugs/errors	Properly plan the test cases, have proper testing of edge cases	30	0.5
13	Testing of frontend software	Wrong test cases, not covering all parameters, bugs/errors	Properly plan the test cases, have proper testing of edge cases	30	0.8
14	Development complete integration of iot, backend, frontend and software	Incompatible modules	While developing check for compatibility between modules	30	0.8
15	Testing after integration	Wrong test cases, not covering all parameters	Properly plan the test cases, have proper testing of edge cases	20	0.4
16	Result validation	Error in human validating	Reverify the result	10	0.1
17	Result verification	Error in verification	Reverify the result	10	0.2
18	Feedback	Wrong feedback given	Take feedback from multiple sources	20	0.6
19	Prototype model correction and testing	Wrong test cases, not covering all parameters, bugs/errors	Properly plan the test cases, have proper testing of edge cases	20	0.3
20	Report Writing	Wrong delivery of information, improper formatting, Scope completion proof not clearly mentioned	Take experts and reverify the report	30	0.1

Table 2. Risk Analysis Table

3.5 Software Requirement Specification Document

3.5.1 Introduction

3.5.1.1 Purpose

Nowadays, the goods transportation system suffers from the problem of unavailability of the trucks or the empty trucks moving to places because of lack of information of availability of the vehicles at the time of booking sometime. Hence to solve this problem of goods transportation system there should be an application which would help the trucks owners as well as the customers to easily get the information to avoid empty trips or idle vehicles to increase productivity and to make the goods transport economical from all the users.

3.5.1.2 Product Scope

- Design and develop a small physical prototype for Smart Goods Transportation.

- Design and develop simulation software using time series analysis to find least polluted route.
- Design and develop front-end application of software system and connect the same to firebase.
- Result gathering, testing and result analysis for the developed prototype.

3.5.2 Overall Description

3.5.2.1 Product Perspective

Goods transportation System needs to follow a systematic procedure for booking any goods transportation vehicle. The current goods transportation system consider only the one way trip thus increasing the trip cost. It fails to consider factors like return trip, proper pricing, drivers staying in inhuman condition and better technologies to be used to overcome the same. Our product will solve the above problems thus increasing the productivity of the goods transportation system.

3.5.2.2 Product Functions

The key functions of the software are:

- Calculating the trip cost accurately considering all the factors affecting the same.
- Give the least polluted route which will be computed using time series analysis
- Smartly choosing the best route for the journey.
- Providing appropriate vehicle options which are nearest to the source address.
- Sharing the real time location of the vehicle after the begins to the Customer and Vehicle Owner application.
- Giving proper share to car owner and driver.
- Proper pickup and drop services to user.

- OTP for verifying valid user

3.5.2.3 Operating Environment

We will be using Arduino IDE for Arduino Uno circuit , the Raspbian OS for Raspberry PI, Android Studio to develop the applications And Windows OS and Mac OS for running Raspberry Pi, Pycharm, Android Studio and other software.

3.5.2.4 Design and Implementation Constraints

The prototype will be demonstrated in a the college campus. It will not be feasible to create a prototype suitable for complex real world scenarios given the time constraint. Due to limitation that the pollution data for the complete route is not available we would develop an Arduino Uno circuit and place it at the junctions to collect the data regarding the same.

3.5.2.5 Assumptions and Dependencies

The assumption are:

- Every driver and the user has knowledge about and access to android application in their smart phones.
- The dataset which we have built will be accurate.

3.5.3 External Interface Requirements

3.5.3.1 User Interfaces

Users are going to communicate through their respective android application. Information displayed will be:

- Drivers will see the route to customer and then the route to customer’s destination.

- Customer will see the location of the selected vehicles, time and route for destination. Also it must have information about the driver.
- Vehicle owner’s can see their respective vehicle location.
- Customer can track the vehicle until it reaches the destination.
- Agent can see the location of the vehicle.

3.5.3.2 Hardware Interfaces

- Smartphone Devices to run the applications with GPS location service.
- Proper Internet connection
- Arduino Uno
- MQ135

3.5.3.3 Software Interfaces

- Operating System: Windows 10
- Android Studio for Application development. Android version should be 5.0 or above
- Programming language: Python, Java
- Raspbian IDE for sensing locations
- Arduino IDE

3.5.3.4 System Features

System must provide customer to book the vehicle according their need and must provide them time to reach and cost. Drivers must have proper route guidance. Owner must have exact location of their vehicles and can manage their vehicles. Each type of user must have proper guidance to use their application and also to manage their account.

3.5.4 Other Non-Functional Requirements

3.5.4.1 Performance Requirements

● Reliability: The system must be reliable i.e. it must give accurate prediction

● Scalability: System should be eventually to be scaled up to real world scenario.

● Capacity: System should be able to work endlessly without any storage issues.

● Availability: The system should be available to the authorized users at all times.

● Accuracy: The predicted travel time and calculated costs must have high accuracy and the user must be able to track the real time location of their goods via the application anytime and anywhere provided he is connected to the internet.

● Response time: A low response time is desirable to get real-time information, for which processing time must be as low as possible.

3.5.4.2 Security Requirements

The system must be ensured to the fact that no data is lost or leaked and the history of each user is confidential and would not be known to others and also the Integrity would be preserved.

3.5.4.3 Software Quality Requirements

Making sure all the functional requirements are met. The application should be reliable, user-friendly, with emphasis on maintainability to accommodate future modifications.

3.6 Software Design Document (UML)

3.6.1 Class Diagram:

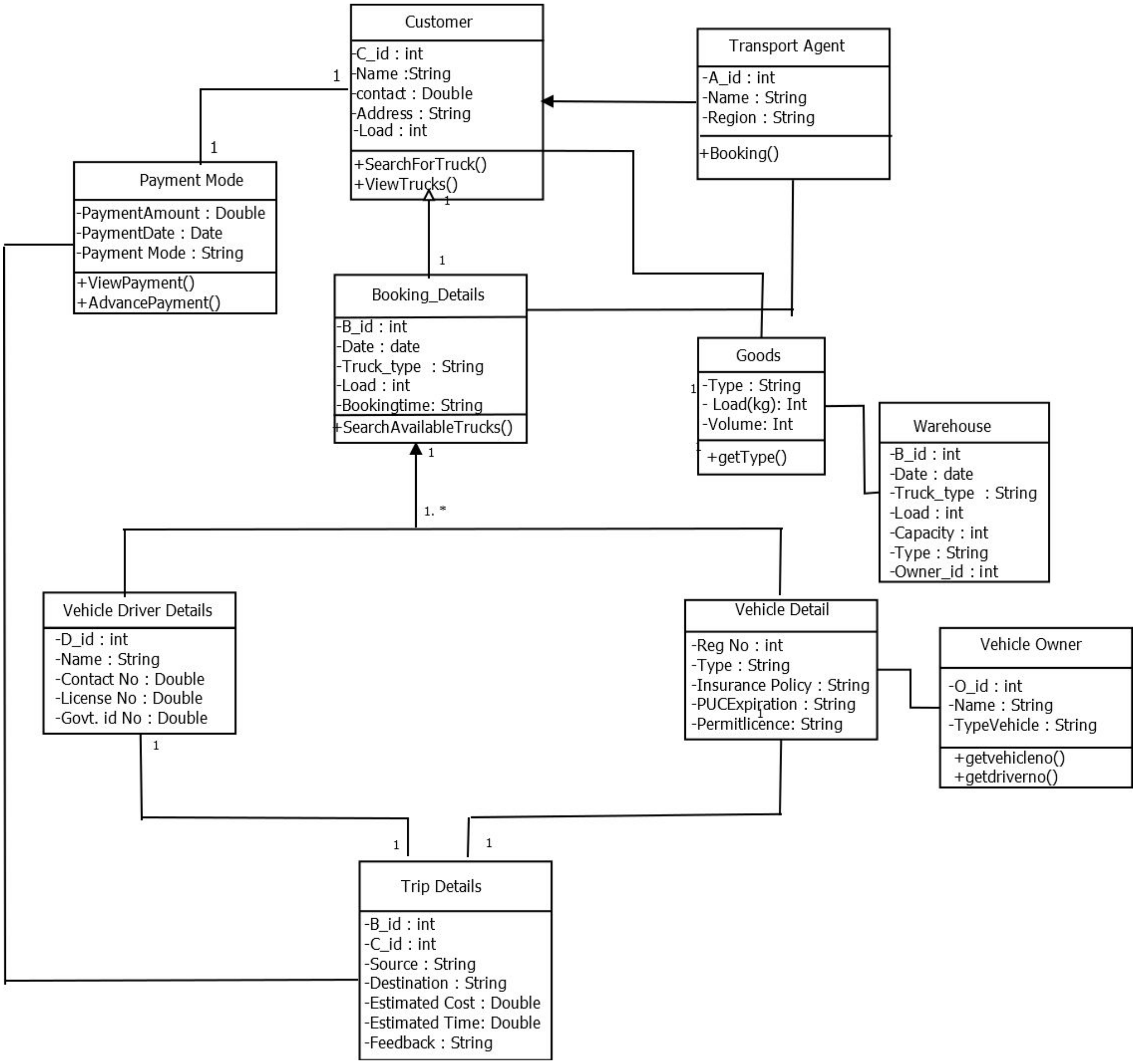


Figure 3.6.1 Class Diagram

3.6.2 Use Case:

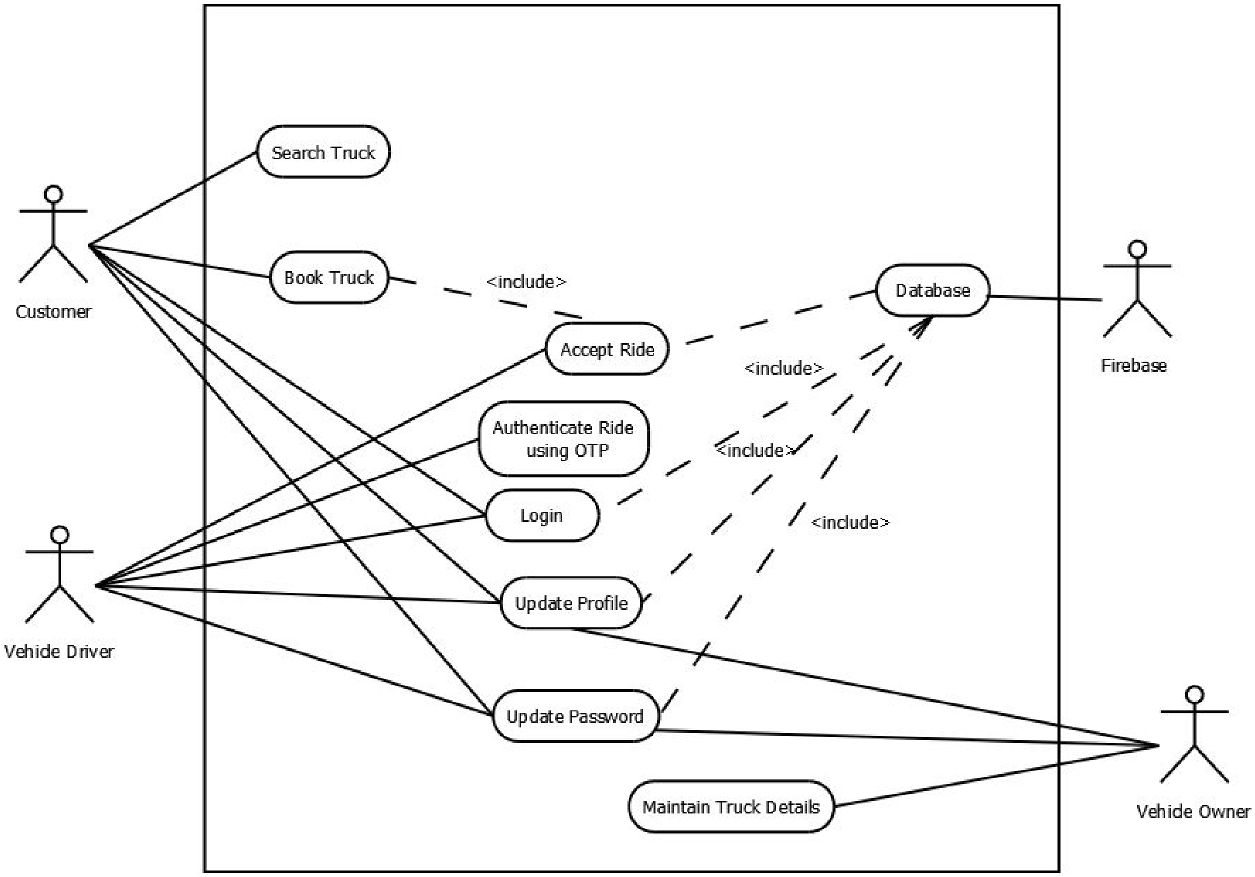


Figure 3.6.2 Use Case

Chapter 4

Implementation and Experimentation

This chapter presents the implementation roadmap and the experimental issues that were faced during the execution of the planned system. It highlights the milestones that were achieved in the progress of the project and also considers the solutions to the problems presented in the implementation of the project.

4.1 Proposed system model implementation

4.1.1 For Android Applications

Customer Application:

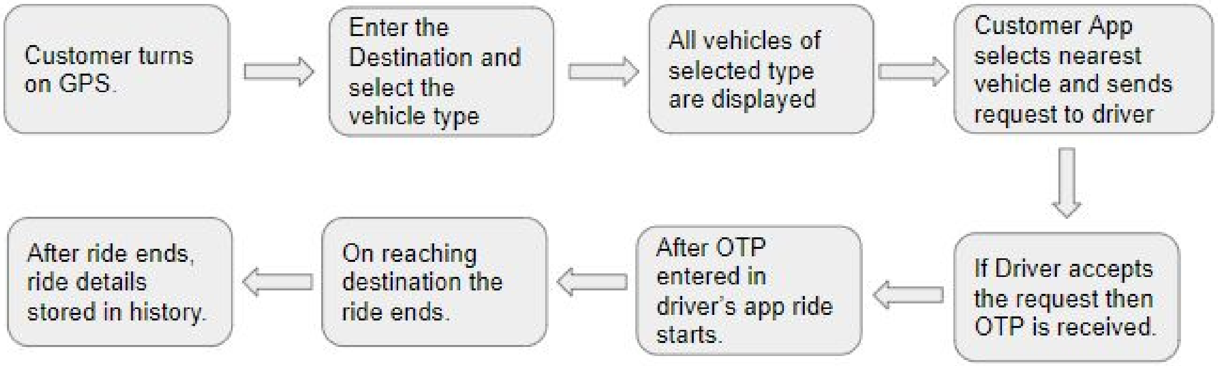

Customer turns on GPS.
Enter the Destination and select the vehicle type
All vehicles of selected type are displayed
Customer App selects nearest vehicle and sends request to driver
If Driver accepts the request then OTP is received.
After OTP entered in driver's app ride starts.
On reaching destination the ride ends.
After ride ends, ride details stored in history.

Figure 4.1.1.1 Block Diagram of Customer App

Working of Customer Application:

- Customer needs to login if account already exists else needs to register to create a new account.
- To book a vehicle the customer needs to turn on the GPS so as to get the current location as the source address .
- Now to book the vehicle the customer needs to enter the destination. The entered destination is verified i.e. if the address is present the database if not the user is asked to enter the destination again.
- The user selects the type of vehicle he/she want to book.

- Now, the customer app searches for all the vehicles of selected type which are in vicinity of the source location. The nearest one is sent the user request.
- If the driver accepts the request 4-digit OTP is displayed to the customer application and sent to the firebase in json format and the driver receives the customers contact details and source address.
- Once the driver reaches source address the OTP is entered by the driver which will be verified by comparing it to the json file stored. After successful verification driver receives the destination address and once the vehicle reach the destination the ride details are stored in ride history.

Driver Application:

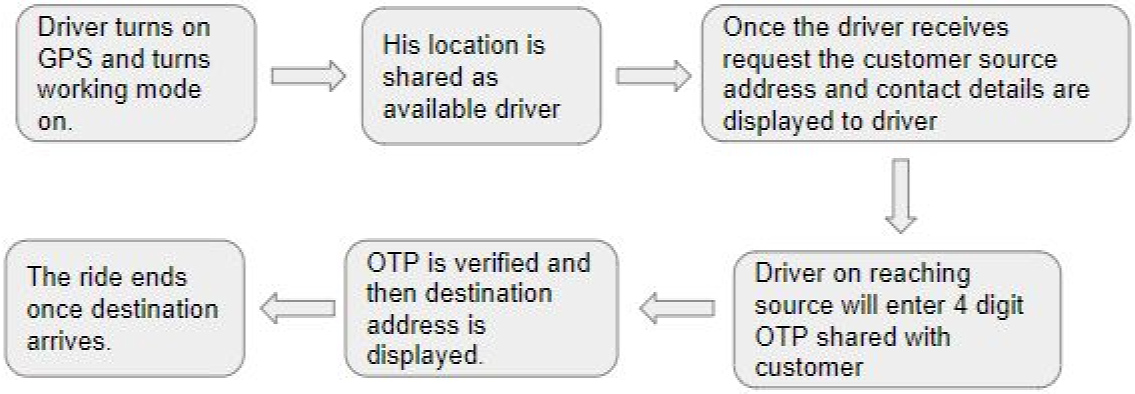

Driver turns on GPS and turns working mode on.
His location is shared as available driver
Once the driver receives request the customer source address and contact details are displayed to driver
Driver on reaching source will enter 4 digit OTP shared with customer
OTP is verified and then destination address is displayed.
The ride ends once destination arrives.

Figure 4.1.1.2 Block Diagram of Driver App

Working of Driver Application:

- Driver needs to login if account already exists else needs to register to create a new account.
- To get the bookings of a vehicle the driver needs to turn on the GPS so as to get the current location of the driver and working mode. The drivers current address is stored on firebase. Now that the working mode is turned on the driver's current location is displayed on the map.
- If the source address entered by the customer is in vicinity of the driver's current location then the user request will be received by the driver.

- The driver will receive the customer’s source address and contact details. Also 4 digit OTP will be displayed on the customer application which is further stored on firebase .
- Once the driver reaches the source address the customer conveys the OTP received which is entered by driver in driver application.
- The OTP entered is compared with the OTP stored. After OTP is verified driver receives details regarding destination address.
- If verification fails then the driver needs to re-enter the OTP.
- On reaching the destination the driver needs to end the ride by pressing end now button. On doing so the driver’s current location is fetched using GPS and driver’s location is updated in database and shown in the map.

Owner Application:

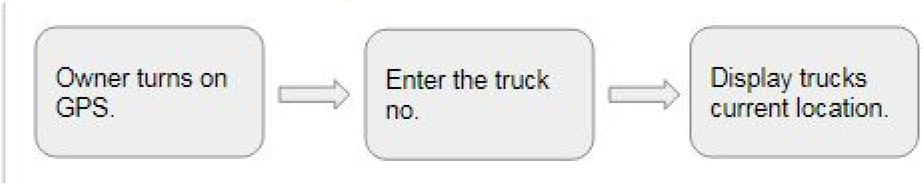


Figure 4.1.1.3 Block Diagram of Owner App

Working of Owner Application:

- Owner need to login if the account already exists else needs to register.
- Owner should update all the vehicle numbers of the vehicles which are owned by him.
- Owner then needs to turn on the GPS. And then click on my profile in order to check the current location of the truck. This redirects the owner to page where he/she needs to enter the vehicle number.
- The vehicle number fed by the owner is verified by checking the vehicle numbers of the vehicle under the owner on firebase .
- On successful verification the current location of the vehicle is displayed.
- If verification fails then the owner is asked to re-enter the vehicle number.

4.1.2 For Elliptical Curve Cryptography

- The server encrypts the 4 digit randomly generated OTP and sends to the client.

- The client decrypts the 4 digit OTP from the cipher text received.

4.1.3 For SARIMA Model

- **Arduino model using MQ135 as sensor for collecting Air pollution data:**

  Air quality sensor for detecting a wide range of gases, including NH3, NOx, alcohol, benzene, smoke and CO2. Ideal for use in office or factory. MQ135 gas sensor has high sensitivity to Ammonia, Sulfide and Benze steam, also sensitive to smoke and other harmful gases. It is with low cost and particularly suitable for Air quality monitoring application.

Block Diagram

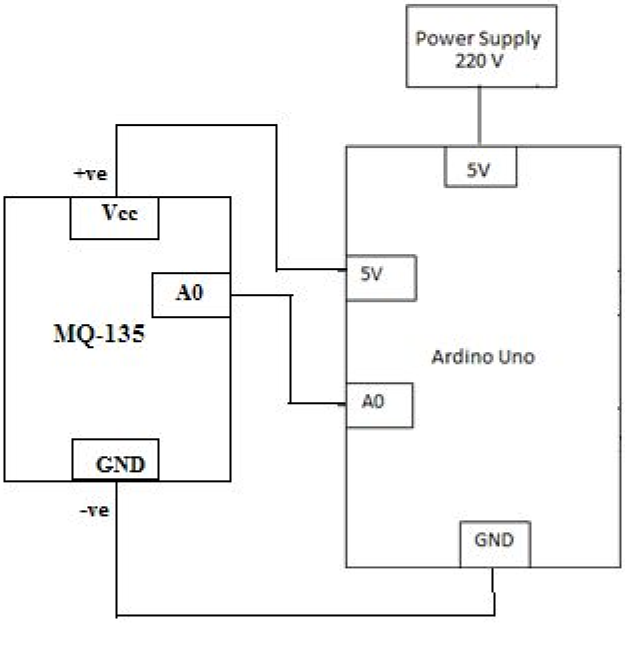


Figure 4.1.3.1 Block Diagram of Arduino Module using MQ135

The working of SARIMA model:

- Input data used is in csv format.
- Takes current date and time as input and extracts data of current hour from CSV file.
- Then SARIMA function is applied to get the predicted value as the result.

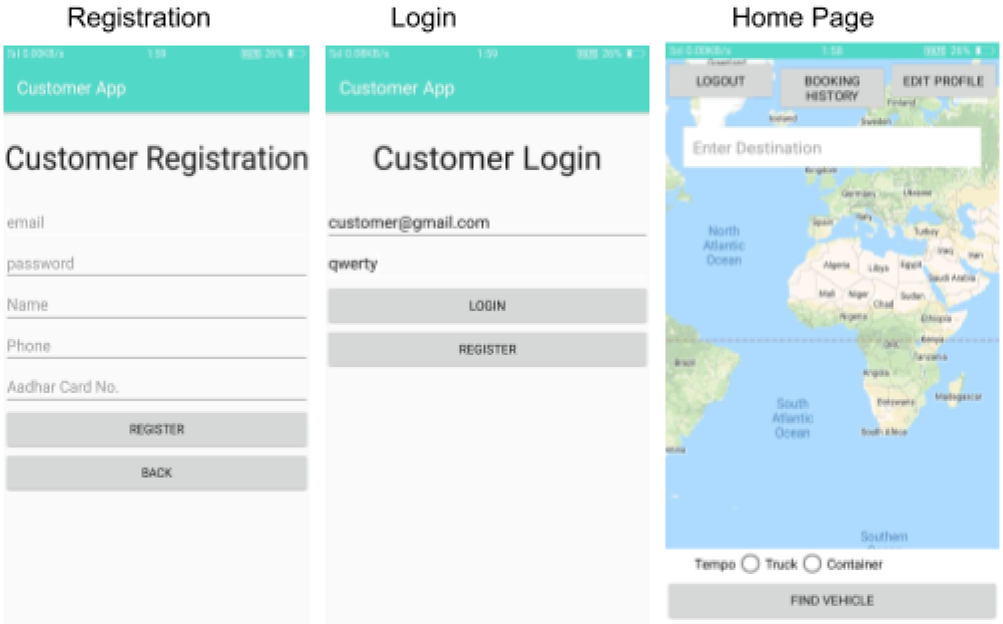
Chapter 5

Gathering of Results

This chapter presents the results of the system after implementation and how testing was done on the system.

5.1 Results

5.1.1 Customer Application:



Registration

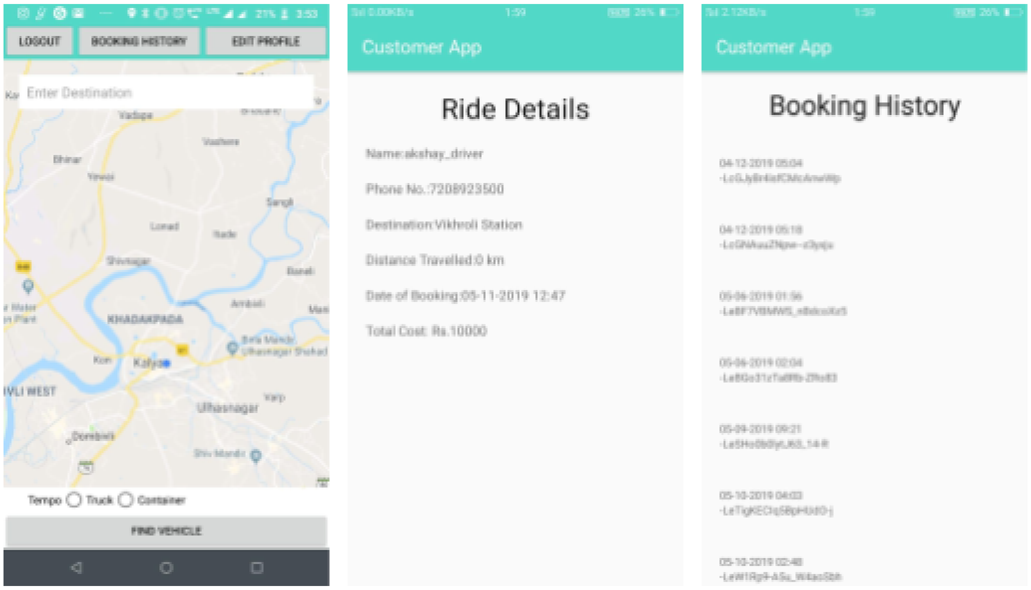
Login

Home Page

customer@gmail.com

qwerty

Figure 5.1.1.1 Login, Register and Home Page



Ride Details

Name:akshay_driver

Phone No.:7208923500

Destination:Vikhroli Station

Distance Travelled:0 km

Date of Booking:05-11-2019 12:47

Total Cost: Rs.10000

Booking History

04-12-2019 05:04
-LcGJyBn6sfCMcAnwWp

04-12-2019 05:18
-LcGhAsuZNpw--z3yqs

05-06-2019 01:56
-LeBF7VBMWS_xBdcoXz5

05-06-2019 02:04
-LeBGo31zTa8Rb-2Ro83

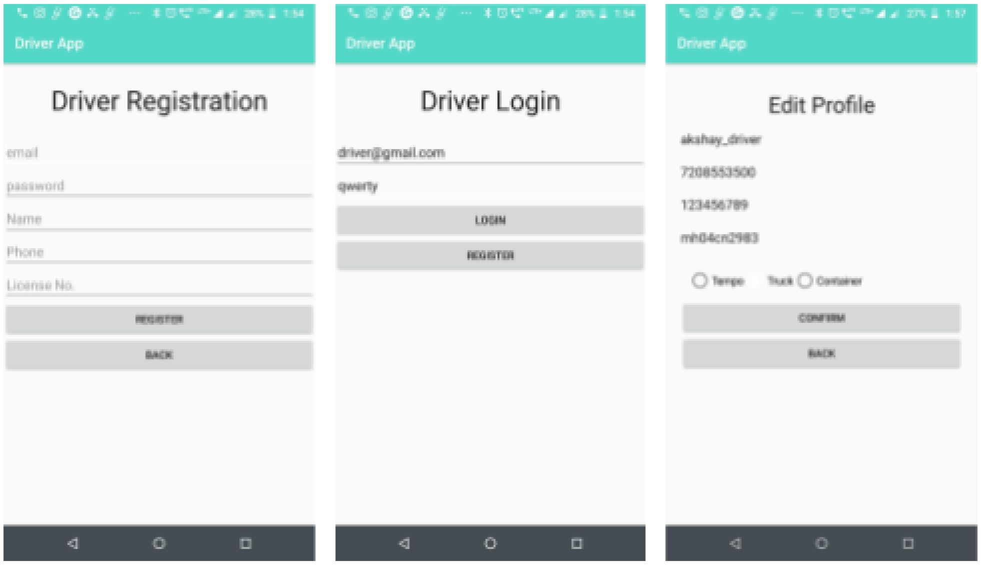
05-09-2019 09:21
-LeSHo0bDyLJ63_14-R

05-10-2019 04:03
-LeTigKECIq5BpHJd0-j

05-10-2019 02:48
-LeW1Rg9-A5u_W4acSbh

Figure 5.1.1.2 Get Ride, Ride Details, Booking History

5.1.2 Driver Application:



Driver Registration

email

password

Name

Phone

License No.

REGISTER

BACK

Driver Login

driver@gmail.com

qwerty

LOGIN

REGISTER

Edit Profile

akshay_driver

7208553500

123456789

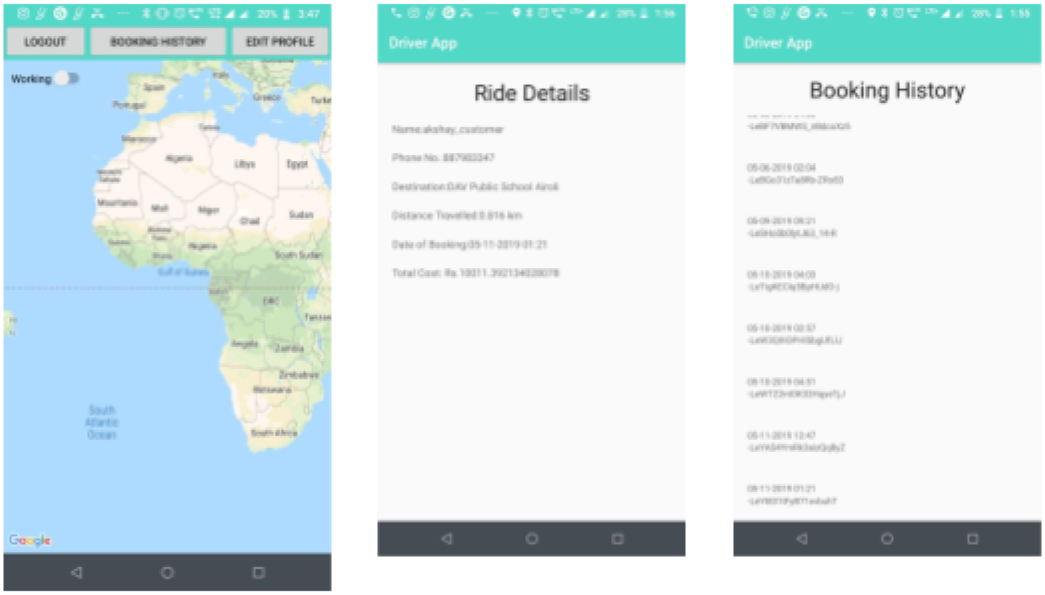
mh04cn2983

○ Tempo Truck ○ Container

CONFIRM

BACK

Figure 5.1.2.1 Registration, Login And Edit Profile



Ride Details

Name:akshay_customer

Phone No. 887903247

Destination:DAV Public School Airoli

Distance Travelled:0.816 km

Date of Booking:05-11-2019 01:21

Total Cost: Rs.10011.392134028078

Booking History

-[illegible]

05-06-2019 02:04
-[illegible]

06-09-2019 09:21
-[illegible]

05-10-2019 04:03
-[illegible]

05-10-2019 02:57
-[illegible]

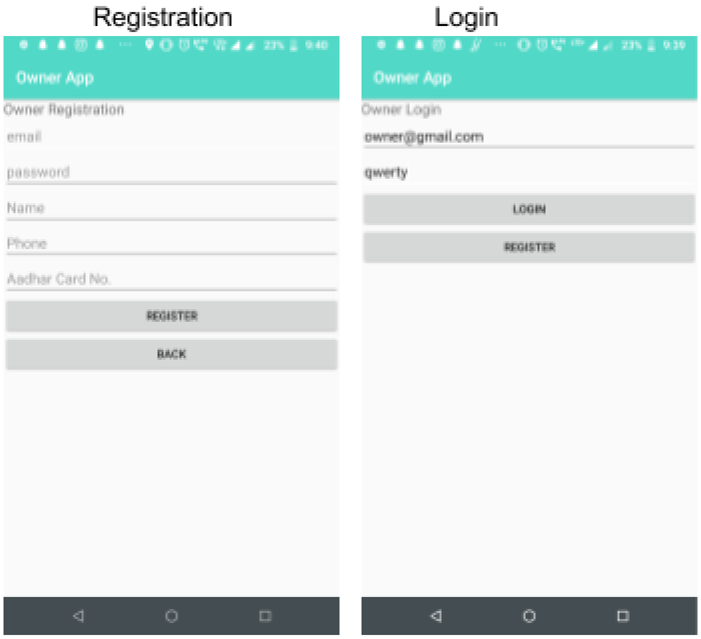
08-10-2019 04:51
-[illegible]

05-11-2019 12:47
-[illegible]

08-11-2019 01:21
-[illegible]

Figure 5.1.2.2 Current Location display, Ride Details, Booking History

5.1.3 Owner Application:



Registration

Owner Registration

email

password

Name

Phone

Aadhar Card No.

Login

Owner Login

owner@gmail.com

qwerty

Figure 5.1.3.1 Registration and Login Page Owner App

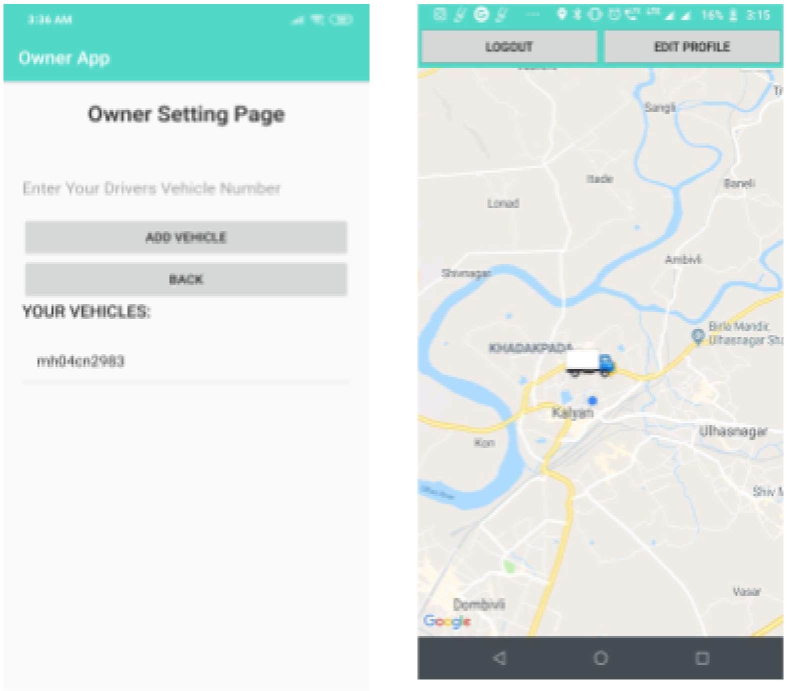
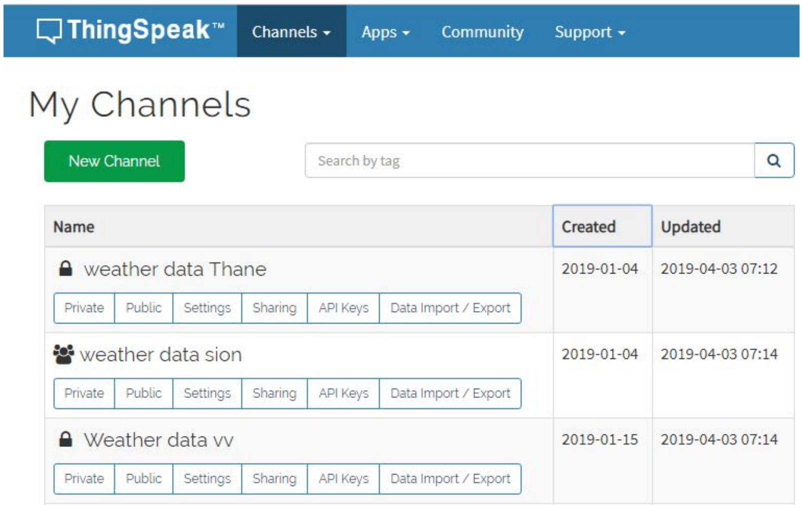


Figure 5.1.3.4 Edit Details and Driver Location display

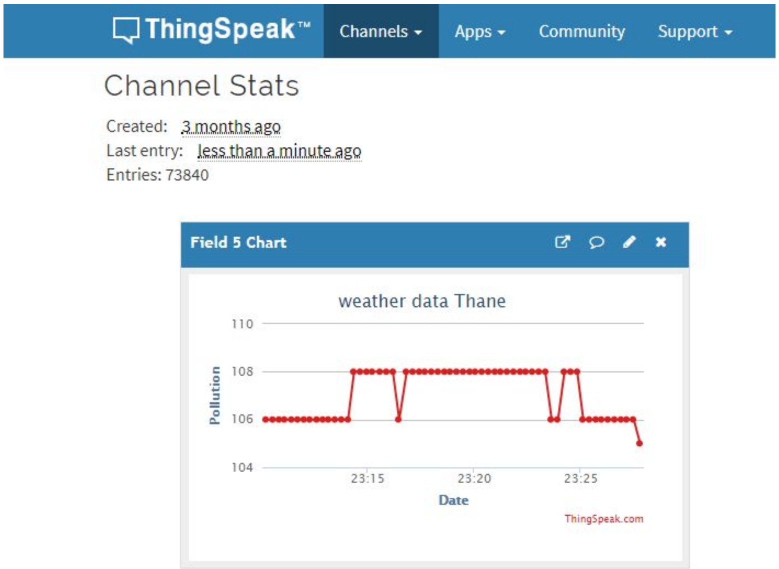
5.1.4 SARIMA model:



Name	Created	Updated
weather data Thane Private Public Settings Sharing API Keys Data Import / Export	2019-01-04	2019-04-03 07:12
weather data sion Private Public Settings Sharing API Keys Data Import / Export	2019-01-04	2019-04-03 07:14
Weather data vv Private Public Settings Sharing API Keys Data Import / Export	2019-01-15	2019-04-03 07:14

Figure 5.1.4.1 Channels On ThingSpeak

The graph generated by data uploaded:



Channel Stats

Created: 3 months ago

Last entry: less than a minute ago

Entries: 73840

Figure 5.1.4.2 Data in Thane Channel

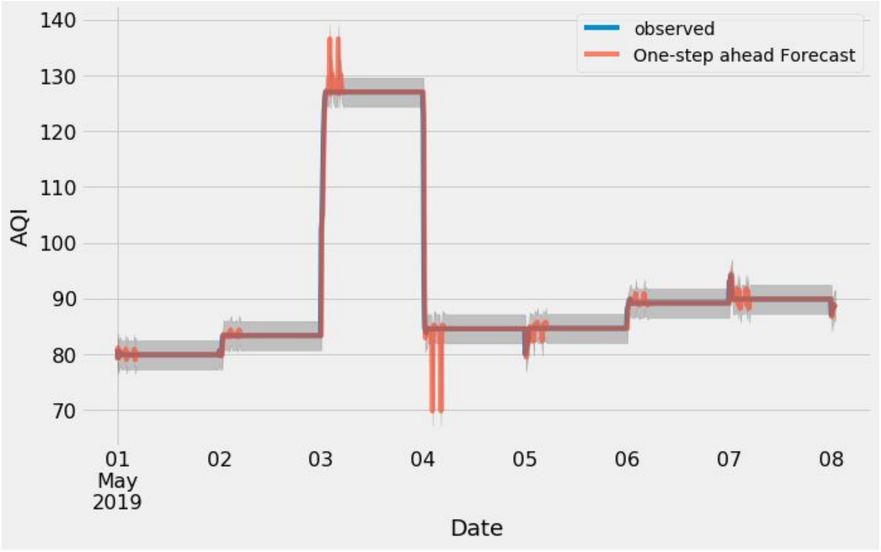
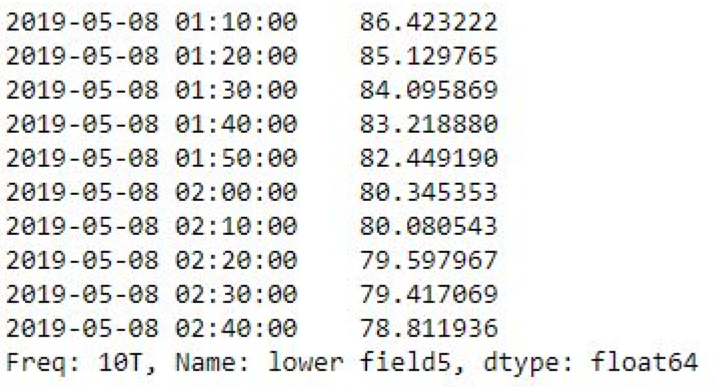


Figure 5.1.4.3 Comparison of Test Data and Predicted Value



```
2019-05-08 01:10:00    86.423222
2019-05-08 01:20:00    85.129765
2019-05-08 01:30:00    84.095869
2019-05-08 01:40:00    83.218880
2019-05-08 01:50:00    82.449190
2019-05-08 02:00:00    80.345353
2019-05-08 02:10:00    80.080543
2019-05-08 02:20:00    79.597967
2019-05-08 02:30:00    79.417069
2019-05-08 02:40:00    78.811936
Freq: 10T, Name: lower field5, dtype: float64
```

Figure 5.1.4.4 Forecasted Value

5.2 Software Testing (Software testing reports at various levels)

5.2.1 Test Plan

5.2.1.1 Introduction

Goods Transportation is important for businesses and the life of a city. However, the efficiency as well as the environmental impacts is negatively influenced by traffic congestion, scarcity of loading areas, sub-optimal delivery routes and idle return vehicles. The transport of dangerous goods is also a safety risk. The proposed system will make goods transportation smarter, more efficient, environmentally friendly and safe.

5.2.1.2 Objectives

Our project is an android application for which the work is divided into three phases. The test plan is actually developed for the three modules and is divided among the testers. The objective of the project is that the customer should be able to book the vehicle for transporting goods from source A to destination B using the customer application.

- **Phase 1** of the project will deliver the initial system with functionality to input the GPS location of the customer and also get the GPS location of the driver along with its distance from the source and time to reach the source and book the vehicle. The driver application will be able to receive the request only when they have turned on the working mode.
- **Phase 2** of the project will integrate the security features by adding Elliptical Curve Cryptography for sending OTP from backend server to customer application and from driver application to backend server.
- **Phase 3** of the project will deliver the time series data and prediction based on the time series analysis to give the least polluted route.

Team Members

Resource Name	Role
Simran Bhutani	Developer/ Tester
Nisha Patel	Developer/ Tester
Prashant Kadam	Developer/ Tester
Akshay Sigar	Developer/Tester

5.2.1.3 Scope

The scope of the project can be divided into two sections:

The first section being the de sign and development of a small physical prototype for the Smart Goods Transportation system. We will be demonstrating how our system is aimed at working Truck Owner or Agent. If the customer books a cab, data will be sent through our servers to the nearest driver on our campus grounds. When this request will be accepted, a map will be displayed on the driver's IOT device which he will follow to the customer's location. To start the trip, the customer will be required to give the driver an OTP, which will be encrypted from the server and sent immediately. The driver will input and this OTP will then be verified.

The second section includes the design and development of a simulation software using time series analysis to find least polluted route. In this system, given the source and destination, the least polluted route will be found between them, based on the data that is collected previously. The simulated software will show the various possible routes between the source and the destination with the different pollution levels and costs. The user will have the option of selecting one of these routes.

5.2.1.4 Assumptions / Risks

Assumptions

This section lists assumptions that are made specific to this project.

1. The user has registered with the app.
2. The driver has registered with a valid driving license and has a valid truck registration.
3. The truck owner should provide valid details of his trucks.

Risks

The following risks have been identified and the appropriate action has been identified to mitigate their impact on the project. The impact (or severity) of the risk is based on how the project would be affected if the risk was triggered. The trigger is what milestone or event would cause the risk to become an issue to be dealt with.

h#	Risk	Impact	Trigger	Mitigation Plan
1	Changes in government policy, may result in obsolescence of software or may require all the test cases to be changed to incorporate changes.	High	Delays in implementation date.	Each iteration, functionality will be closely monitored. Priorities will be set and discussed by stakeholders. Since the driver is functionality and not time, it may be necessary to push the date out.
2	Weekly delivery is not possible because any of the developers fall sick or take leave.	Medium	Product doesn't get delivered on schedule	Ensure the availability of extra developers(maybe from other departments), who are familiar with the product.
3	Data collected is inappropriate or missing	High	The prediction of the least polluted route will not be accurate	Ensure that the sensor is working properly.

Table 3. Risk Associated

5.2.1.5 Test Approach

1. The project is using an agile approach, with weekly iterations. At the end of each week the requirements identified for that iteration will be delivered to the team and will be tested.

The sprints with their deliverables are identified as follows:

Sprints for phase 1:

Unit testing for the first few units is:

(here sprints and units are considered to be the same)

a) If user data entered in the forms is stored in the database without any errors.

b) If the form directs the user correctly at all times when the user enters wrong values, or leaves a field blank by mistake.

c) If the user enters any incompetent values(wrong format values), appropriate dialog/message boxes should indicate that the format is wrong. For eg- if the user forgets to include '@' in the email field.

Sprints for phase 2:

Black box testing will be performed with the following as sprints:

1) The max booking a customer can do at a time.
2) If the driver refuses to go to destination of the customer.
3) If the driver cancels more than a cap of booking per day.
4) If the entered data is proper or not.
5) The behavior of application for different input combination.

White box testing will be performed with the following as sprints:

1) Testing the flow of activities using control flow graph.

5.2.1.6 Test Environment

- The testing environment will be different from the production environment.

- However the same servers used in the production environment will be used for the test environment also in order to ensure that the servers will be a fit for the prod environment.

5.2.1.7 Milestones / Deliverables

Test Schedule

Task Name	Start	Finish	Effort
Test Planning	29/7/18	2/8/18	4 days
Review Requirements documents	3/8/18	6/8/18	3 days
Create initial test estimates	7/8/18	9/8/18	2 days
Staff and train new test resources	11/8/18	30/8/18	20 days
First deploy to QA test environment	1/9/18	5/9/18	4 days
Unit testing – Lane Detection	6/9/18	14/9/18	9 days
Unit Testing - Object detection	15/9/18	25/9/18	10 days
Integration Testing	26/9/18	30/9/18	4 days
System testing	1/10/18	15/10/18	15 days
Regression testing	16/10/18	25/10/18	10 days
UAT	26/10/18	30/10/18	4 days
Resolution of final defects and final build testing	1/11/18	10/11/18	10 days
Deploy to Staging environment	11/11/18	16/11/18	5 days

Performance testing	17/11/18	25/11/18	7 days
Release to Production	26/11/18	28/11/18	2 days

Table 4. Test Schedule

5.3 Black Box Testing

Driver Application:

Login Module										
Decision Table	Conditions	Email id	T	T	T	T	F	F	F	F
		Password	T	T	F	F	T	T	F	F
		Submit Button	T	F	T	F	T	F	T	F
	Action	Page shown	Map Page	Same Page	Same Page	Same Page	Same Page	Same Page	Same Page	Same Page

Optimised Table	Conditions	Email id	T	T	T	F
		Password	T	T	F	NA
		Submit Button	T	F	NA	NA
	Action	Page shown	Map Page	Same Page	Same Page	Same Page

Driver Map Activity										
Decision Table	Conditions	Turn Gps On	T	T	T	T	F	F	F	F
		Select Working	T	T	F	F	T	T	F	F
		Edit Profile	T	F	T	F	T	F	T	F
	Action	Page shown	Map Page	Same Page	Same	Same Page	Same Page	Same Page	Same	Same Page

					Page				Page	

Optimised Table	Conditions	Turn Gps On	T	T	T	F
		Select Working	T	T	F	NA
		Edit Profile	T	F	NA	NA
	Action	Page shown	Map Page	Same Page	Same Page	Same Page

EQUIVALENCE CLASS				
Topic - Email				
Class Name	**Range**	**Type**	**Expected Output**	
C1	Minimum 1 [a-z] & [1-9] & .com & @	Valid Class	Accept	
C2	Minimum 1 [a-z] & [1-9] & .com & @ & special classes	Valid Class	Accept	
C3	[1-9] and special classes	Invalid Class	Reject with error invalid email	
C4	[A_Z} & .com & @	Invalid Class	Reject with error invalid email	
Test Number	**Test Value**	**Expected Output**	**Actual Output**	**Class Tested**
1	NISHACP@GMAIL.com	Reject with error invalid email	Reject with error invalid email	C4
2	1234@somaiya.edu	Reject with error invalid email	Reject with error invalid email	C3
3	akshay.sigar@somaiya.edu	Accept	Accept	C1
4	simran123@somaiya.edu	Accept	Accept	C1
5	prashant_kadam@somaiya.edu	Accept	Accept	C2
6	nisha.cp	Reject with error invalid email	Reject with error invalid email	C2

BOUNDARY VALUE ANALYSIS

Topic - Email(No. of characters before @)			
BV	***Value***	**Expected Output**	
Min-1	*0*	Reject with error invalid email	
Min	1	Accept	
Min+1	2	Accept	
Max-1	253	Accept	
Max	254	Accept	
Max+1	255	Reject with error invalid email	
Test Number	**Test Value**	**Expected Output**	**Actual Output**
1	@somaiya.edu	Reject with error invalid email	Reject with error invalid email
2	a@somaiya.edu	Accept	Accept
3	np@somaiya.edu	Accept	Accept
4	as(255 char)@somaiya.edu	Reject with error invalid email	Reject with error invalid email
5	s(252 char)@somaiya.edu	Accept	Accept
6	a(253char)@somaiya.edu	Accept	Accept

EQUVALENCE CLASS				
Topic - Password				
Class Name	**Range**	**Type**		
C1	Minimum 1 [a-z] & [1-9] & .com & @	Valid Class		

C2	6 -10 digits & 1<[1-9[& 1< special characters &<1 [a-z]	Valid Class		
C3	5-10 digits [a-z]	Valid Class		
C4	>10 digits	Invalid Class		
C5	<6 digits	Invalid Class		
C6	Null Value	Invalid Class		
Test Number	**Test Value**	**Expected Result**	**Actual Output**	**Class Tested**
1	Nisha	Accept	Accept	C3
2	Akshay_123	Accept	Accept	C1
3	Simran7!Bh	Accept	Accept	C2
4	Prashant123	Reject	Reject	C4
5	!23	Reject	Reject	C5
6	Null	Reject	Reject	C6

BOUNDARY VALUE ANALYSIS			
Topic - Password			
BV	***Value***	**Expected Output**	
Min-1	*4*	Reject	
Min	5	Accept	
Min+1	6	Accept	
Max-1	9	Accept	
Max	10	Accept	
Max+1	11	Reject	
Test Number	**Test Value**	**Expected Result**	**Actual Output**
1	Nish	Reject	Reject
2	aksha	Accept	Accept
3	akshay	Accept	Accept
4	akshaynish	Accept	Accept
5	simranpra	Accept	Accept
6	simranprasha	Reject	Reject

EQUVALENCE CLASS				
Topic - Contact No.				
Class Name	**Range**	**Type**		
C1	10 digits [1-9]	Valid Class		
C2	<10 digits	Invalid Class		
C3	>10 digits	Invalid Class		
C4	10 digit [a-z] & [1-9] & special characters	Invalid Class		
Test Number	**Test Value**	**Expected Result**	**Actual Output**	**Class Tested**
1	1234567890	Accept	Accept	C1
2	123456789	Reject	Reject	C2
3	12345678912	Reject	Reject	C3
4	12ab@31429	Reject	Reject	C4
5	abcdesladv	Reject	Reject	C4
6	!!!@@##$$^	Reject	Reject	C4

BOUNDARY VALUE ANALYSIS			
Topic - Contact No. (No. of digits)			
BV	***Value***	**Expected Output**	
Min-1,Max-1	9	Reject	
Max,Min	10	Accept	
Min-1,Max+1	11	Reject	
Test Number	**Test Value**	**Expected Result**	**Actual Output**
1	123456789	Reject	Reject
2	1234567890	Accept	Accept
3	12345678912	Reject	Reject

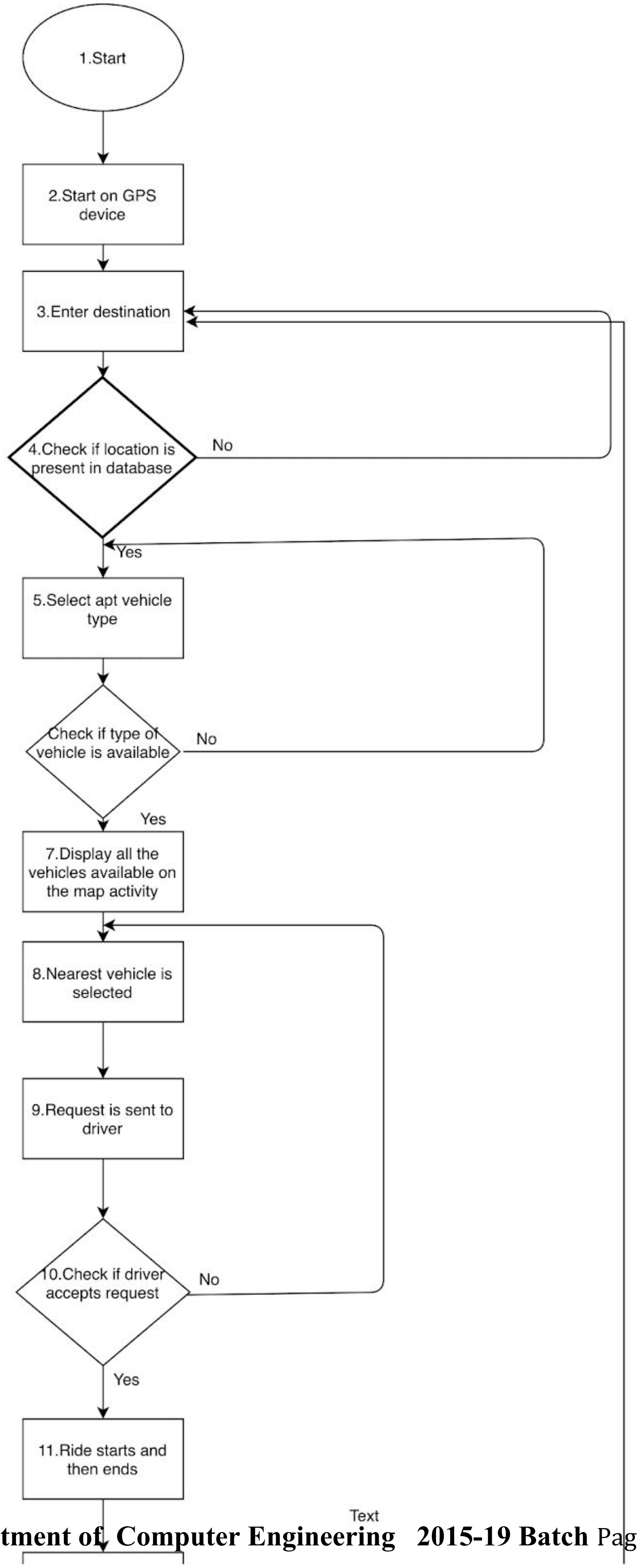
Table 5. Black Box Test Cases for Driver App

5.4 White Box Testing

5.4.1 Customer Application:

Psuedo-code:

```
1: Start
2: Turn on Gps
3:Enter Destination
4: If location present in database jump to 5 or else jump to 3.
5:Select Appropriate vehicle type.
6: if type of vehicle available then jump to 7 or else 5
7:Display the vehicles on map
8: Nearest Vehicle is selected.
9: Request is sent to driver
10: if driver accepts the request then jump to 11 else move to 8
11:Ride starts
12:Ride is stored in history log
13: if User presses logout then 14 else jump to 3
14:display login page
15:stop
```


1.Start
2.Start on GPS device
3.Enter destination
4.Check if location is present in database
No
Yes
5.Select apt vehicle type
Check if type of vehicle is available
No
Yes
7.Display all the vehicles available on the map activity
8.Nearest vehicle is selected
9.Request is sent to driver
10.Check if driver accepts request
No
Yes
11.Ride starts and then ends
Text

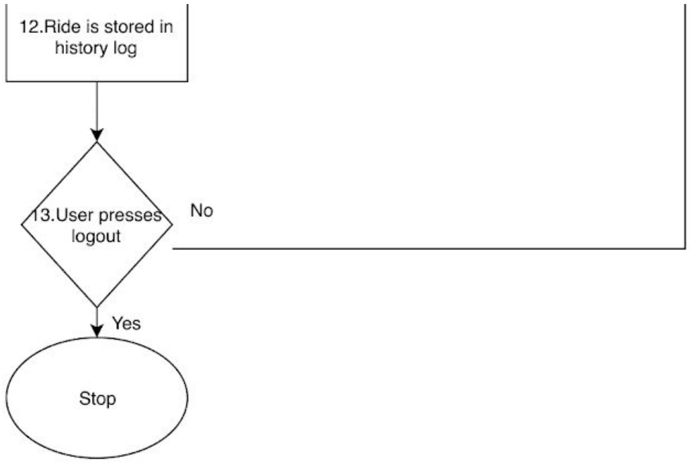


Figure 5.4.1.1 Control Flow Graph of Customer App

Paths:

Path 1: 1-2-3-4(T)-5-6(T)-7-8-9-10(T)-11-12-13(T)

Path 2: 1-2-3-4(F)-2-3-4(T)-5-6(T)-7-8-9-10(T)-11-12-13(T)

Path 3: 1-2-3-4(T)-5-6(F)-2-3-4(T)-5-6(T)-7-8-9-10(T)-11-12-13(T)

Path 4: 1-2-3-4(T)-5-6(T)-7-8-9-10(F)-8-9-10(T)-11-12-13(T)

Path 5: 1-2-3-4(T)-5-6(T)-7-8-9-10(T)-11-12-13(F)-2

Test Case	input	Expected Output	Actual Output	Result
Path1	1: GPS turned on. 2: Entered Destination. 3: Selected Vehicle type. 4: driver accepts request. 5: Press Logout.	1,2: Verifies if location is in database. 3: Verifies selected vehicle is available. Display all the vehicles available which are of selected type. Nearest one selected and request is sent. 4: Driver accepts request.	1,2: Verifies if location is in database. 3: Verifies selected vehicle is available. Display all the vehicles available which are of selected type. Nearest one selected and request is sent. 4: Driver accepts request.	Pass

		Ride starts and stored in ride history. 5: Goes to the login page	Ride starts and stored in ride history. 5: Goes to the login page	
Path 2	1: GPS turned on. 2: Entered Destination not in database.	1,2: Verifies if location is location is available. IF not Shows message please re-enter destination.	1,2: Verifies if location is location is available. IF not Shows message please re-enter destination.	Pass
Path 3	1: GPS turned on. 2: Entered Destination. 3: Selected Vehicle type not available.	1,2: Verifies if location is in database. 3: Verifies selected vehicle is available.If not again enter the destination and find available vehicle.	1,2: Verifies if location is in database. 3: Verifies selected vehicle is available.If not again enter the destination and find available vehicle.	Pass
Path 4	1: GPS turned on. 2: Entered Destination. 3: Selected Vehicle type. 4: driver rejects request.	1,2: Verifies if location is in database. 3: Verifies selected vehicle is available. Display all the vehicles available which are of selected type. Nearest one selected and request is sent. 4: driver rejects the request all available vehicles are displayed and ask user to select from the available vehicles.	1,2: Verifies if location is in database. 3: Verifies selected vehicle is available. Display all the vehicles available which are of selected type. Nearest one selected and request is sent. 4: driver rejects the request all available vehicles are displayed and ask user to select from the available vehicles.	Pass
Path 5	1: GPS turned on. 2: Entered Destination. 3: Selected	1,2: Verifies if location is in database. 3: Verifies selected vehicle is available. Display all the vehicles	1,2: Verifies if location is in database. 3: Verifies selected vehicle is available. Display all the vehicles	Pass

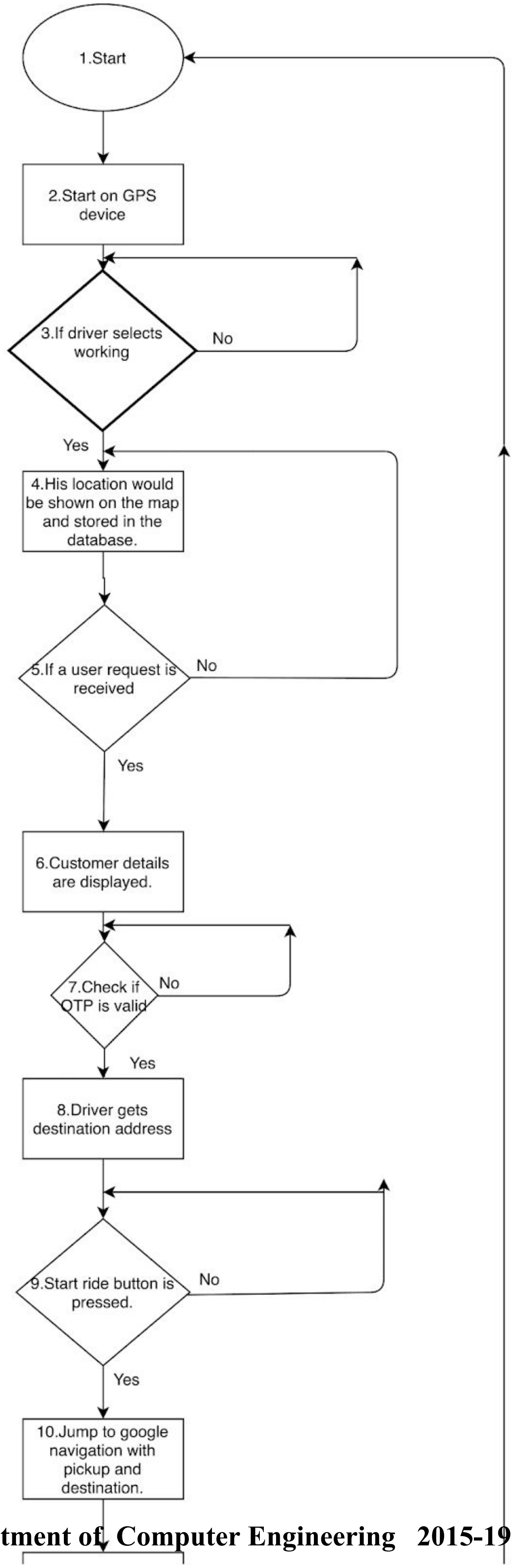
	Vehicle type. 4: driver accepts request. 5:Doesn’t Press Logout.	available which are of selected type. Nearest one selected and request is sent. 4: Driver accepts request. Ride starts and stored in ride history 5: Return to the enter destination page.	available which are of selected type. Nearest one selected and request is sent. 4: Driver accepts request. Ride starts and stored in ride history 5: Return to the enter destination page.	

Table 6. Test Cases Customer App

5.4.2 Driver Application:

Pseudo-code:

```
1: Start
2: Turn on Gps
3: if driver selects working mode then jump 4 else move to 3
 4: Display the location of driver on Map and it is stored in database.
5: if customer request is received then jump to 6 else jump to 4.
6: Display customers pickup location and contact no.
7: if otp is valid then jump to 8 else move to 7
8: Driver gets Destination address.
9: If driver has pressed start button then jump to 10 else move to 9
10: Navigate driver to destination using google navigation
11:Ride ends and all its details is stored in history log
12: if User presses logout then 14 else jump to 3
13:display login page
14:stop
```


1.Start
2.Start on GPS device
3.If driver selects working
No
Yes
4.His location would be shown on the map and stored in the database.
5.If a user request is received
No
Yes
6.Customer details are displayed.
7.Check if OTP is valid
No
Yes
8.Driver gets destination address
9.Start ride button is pressed.
No
Yes
10.Jump to google navigation with pickup and destination.

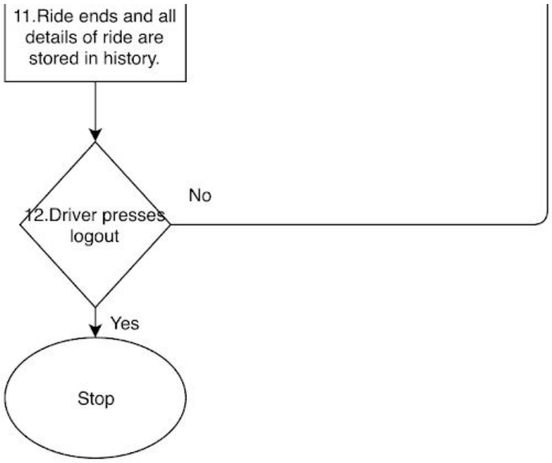


Figure 5.4.2.1 Control Flow Graph of Driver App

Paths:

Path1: 1-2-3(T)-4-5(T)-6-7(T)-8-9(T)-10-11-12(T)

Path2: 1-2-3(F)-3(T)-4-5(T)-6-7(T)-8-9(T)-10-11-12(T)

Path3: 1-2-3(T)-4-5(F)-4-5(T)-6-7(T)-8-9(T)-10-11-12(T)

Path4: 1-2-3(T)-4-5(T)-6-7(F)-7(T)-8-9(T)-10-11-12(T)

Path5: 1-2-3(T)-4-5(T)-6-7(T)-8-9(F)-9(T)-10-11-12(T)

Path6: 1-2-3(T)-4-5(T)-6-7(T)-8-9(T)-10-11-12(F)-4

Test Case	Input	Expected Output	Actual Output	Result
Path1	1: GPS turned on. 2: Working mode on. 3: User Request 4: Valid OTP 5: Start Ride pressed.	1,2: Display Driver's Location on Map. 3: Display Customer Details. 4: OTP verified and destination displayed. 5: Ride starts and driver is navigated using maps until	1,2: Display Driver's Location on Map. 3: Display Customer Details. 4: OTP verified. 5: Ride starts and driver is displayed destination address and navigated using maps until	Pass

	6: Log Out	destination arrives. 6: On log out Stops working mode of driver.	destination arrives. 6: On log out Stops working mode of driver..	
Path 2	1: GPS turned on. 2: Working mode not turned on.	1: Wait until Working mode is turned on and then display its location on map until user request arrives.	1: Wait until Working mode is turned on and then display its location on map until user request arrives.	Pass
Path 3	1: GPS turned on. 2: Working mode on. 3: No User Request	1,2: Display Driver's Location on Map. 3: Display Driver's Location on Map until request arrives.	1,2: Display Driver's Location on Map. 3: Display Driver's Location on Map until request arrives.	Pass
Path 4	1: GPS turned on. 2: Working mode on. 3:User Request 4: Invalid OTP	1,2: Display Driver's Location on Map until request arrives. 3: Get Customer Details 4: Re-enter OTP message.	1,2: Display Driver's Location on Map until request arrives. 3: Get Customer Details 4: Re-enter OTP message.	Pass
Path 5	1: GPS turned on. 2: Working mode on. 3:User Request 4: Valid OTP 5: Driver didn't presses start ride.	1,2: Display Driver's Location on Map. 3: Display Customer Details. 4: OTP verified and destination displayed. 5: Message saying "Press Start Ride button to start the ride".	1,2: Display Driver's Location on Map. 3: Display Customer Details. 4: OTP verified and destination displayed. 5: Message saying "Press Start Ride button to start the ride".	Pass
Path 6:	1: GPS turned on. 2: Working mode on. 3: User Request 4: Valid OTP	1,2: Display Driver's Location on Map. 3: Display Customer Details. 4: OTP verified and	1,2: Display Driver's Location on Map. 3: Display Customer Details. 4: OTP verified and	Pass

	5: Start Ride pressed. 6: Log Out not pressed.	destination displayed. 5: Ride starts and driver is navigated using maps until destination arrives. 6: Returns to 4 and wait for another user request.	destination displayed. 5: Ride starts and driver is navigated using maps until destination arrives. 6: Returns to 4 and wait for another user request.	

Table 7. Test Cases Driver App

5.4.3 Owner Application:

1: Start

2: Turn on Gps

3: Clicks my profile button

4: Enter Truck No.

5: if truck under the owner then jump to 6 else jump to 4.

6: Display current location of the truck.

7: if User presses logout then 8 else jump to 3

8:display login page

9: stop

Control Flow Graph:

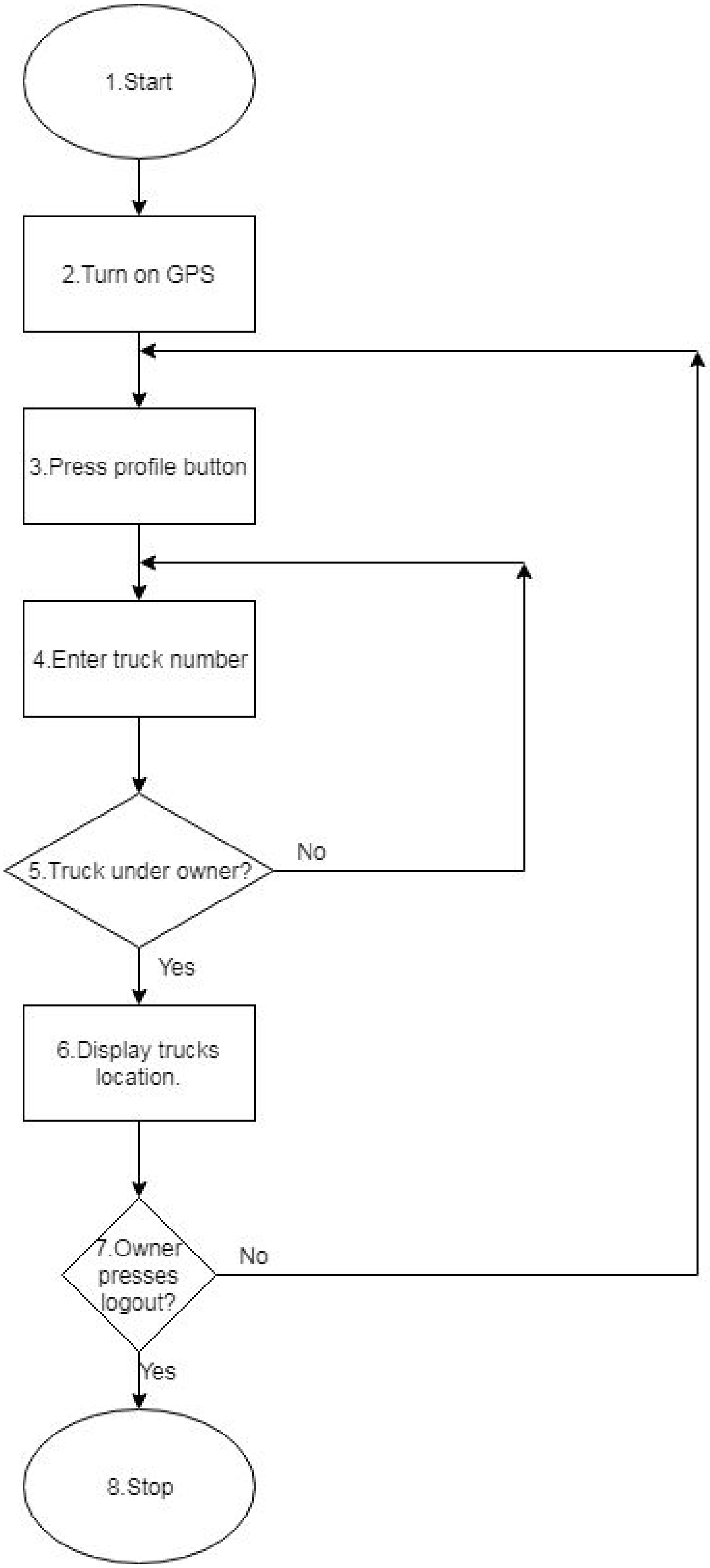


Figure 5.4.3.1 Control Flow Graph of Owner App

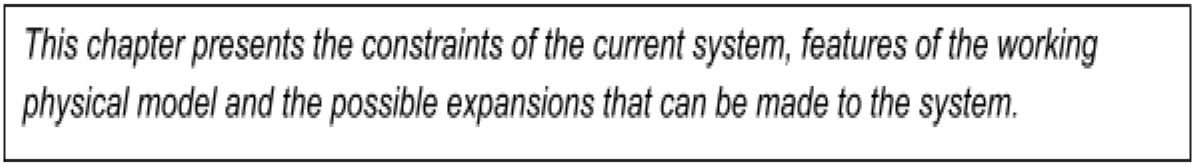
Paths:
Path1: 1-2-3-4-5(Y)-6-7(Y)-8
Path2: 1-2-3-4-5(N)-4-5(Y)-6-7(Y)-8
Path3: 1-2-3-4-5(Y)-6-7(N)-3-4-5(Y)-6-7(Y)-8

Test Case	input	Expected Output	Actual Output	Result
Path1	1: Enter truck no of truck which he owns . 2: Owner Presses logout.	1: Server matches the owner's profile and checks whether the truck is under the owner. If yes then displays the current location of the vehicle. 2: Redirects to the login page`	1: Server matches the owner's profile and checks whether the truck is under the owner. If yes then displays the current location of the vehicle. 2: Redirects to the login page`	Pass
Path 2	1: Enter truck no. of the truck which he doesn't own 2: Owner Presses logout.	1: Server matches the owner's profile and checks whether the truck is under the owner.If no then the owner is asked to re-enter the truck no. 2: Redirects to the login page	1: Server matches the owner's profile and checks whether the truck is under the owner.If no then the owner is asked to re-enter the truck no. 2: Redirects to the login page	Pass
Path 3	1: Enter truck no of truck which he owns . 2: Owner doesn't Presses logout.	1: Server matches the owner's profile and checks whether the truck is under the owner. If yes then displays the current location of the vehicle. 2: Redirects enter truck no. page.	1: Server matches the owner's profile and checks whether the truck is under the owner. If yes then displays the current location of the vehicle. 2: Redirects enter truck no. page.	Pass

Table 8. Test Cases Owner App

Chapter 6

Analysis of Results



This chapter presents the constraints of the current system, features of the working physical model and the possible expansions that can be made to the system.

6.1 Features of current system:

The most important features provided by our current system are:

1.Customer can book a ride from his desired location to his destination and choose between a truck, tempo or container as his ride. The desired vehicle nearest to his location, will be assigned to him and after picking him up the driver on entering an OTP provided by the customer will take him to the desired location.

2.Driver can switch on his app and make himself available. After a customer is assigned to him, he reaches the customer’s destination using the directions provided to him through Google Maps.After taking the customer to his desired location, the trip details will be stored in his application.

3.The owner of the vehicle can keep track of the driver and his current location using maps.Also each trip that the driver takes will be logged with the owner as the history of all the trips will also be stored with the owner.

4.The SARIMA model which is used to find the least-polluted route between the source and the destination predicts the value of pollution at that given point of time between the source and the destination.

6.2 Constraints:

1.The current system does not contain all the locations that will be used in the real-world application. As the location of the places we have chosen, had to be inputted manually through their latitude and longitude, we have done this only for a limited number of locations.

2.The current system does not integrate the results of Time Series Analysis using SARIMA model with the customer and driver applications.

3.The data collected for Time Series Analysis is only for one specific source and destination with different routes.Hence, currently we present a constrained analysis of only one route.

4.Like in a real world model, our system does not provide a method of payment using an
e-Wallet.

6.3 Possible Expansions:

1.As in a real-world application we require all locations, we will need to expand our database to use all the various locations possible.

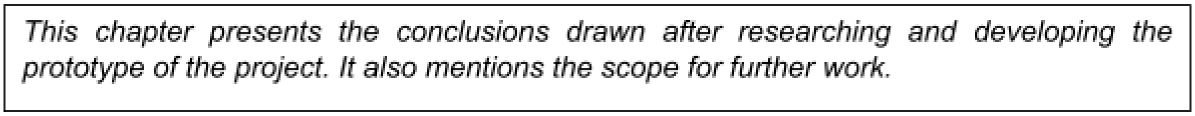
2.Integration of the SARIMA model and the customer and driver application will result in predicting the least polluted route in our application and will give customer the choice to pick his route depending upon the various levels of pollutions on different routes between the source and the destination.

3.To have SARIMA model predict pollution between every possible source and destination, we will be required to collect data using sensors on each and every route, which will then be used to feed into our SARIMA model.

4.We can link our system to an e-Wallet to make it more convenient for our customers who use our application.

Chapter 7

Conclusions and scope for further work



This chapter presents the conclusions drawn after researching and developing the prototype of the project. It also mentions the scope for further work.

7.1 Conclusions

Thus this thesis explains the problem statement, the scope, and the hardware/software requirements of the project to be implemented. Then various research papers have been studied and referenced, and three of them have been selected and conclusions have been drawn from them. These papers gave us an initial idea about how will we go about implementing our project and what various methods can be used to do so. Then the above experiments were performed to decide on the sensors to be used for monitoring the environmental condition. The data for time series analysis is collected using the arduino module as we need the hourly data. Implementation of Graphical User Interface for 3 application (Customer, Driver and Owner) has been implemented. The implementation of ECC encryption algorithm to encrypt the 4 digit OTP which will be randomly generated by the backend software. SARIMA model was developed in order to find the least polluted route.Black box, White box and manual testing was done to test all the above modules.

7.2 Scope for further work

The further work includes inclusion of 2 application such as Agent and Warehouse. The agent app will help us to adjust in real world Goods Transportation System where agent plays an important role.This app will help agent to get their booking done online effortlessly. The Warehouse app will help the warehouse owner to transport goods from one warehouse to other or from warehouse to some destination. This will also help warehouse owner to keep easy track of the goods present in the warehouse, how much is going to arrive or how much has been dispatched from the warehouse. The working prototype of the system will be shown in the Somaiya campus in order to show the working of the prototype. After the conclusion of implementation testing procedures will be applied on the project to test it against all kinds of data.

References:

[1] A. M. Annaswamy, Y. Guan, H. E. Tseng, H. Zhou, T. Phan and D. Yanakiev, "Transactive Control in Smart Cities," in *Proceedings of the IEEE*, vol. 106, no. 4, pp. 518-537, April 2018.

[2] Bag, Rajib & Das, Debasis & Kumar, Rajiv. (2017). An architecture of smart transportation system using modified RR algorithm and VANET.

[3] Byun, Jaehak & Kim, Sooyeop & Sa, Jaehun & Kim, Sangphil & Shin, Yong-Tae & Kim, Jong-Bae. (2016). Smart City Implementation Models Based on IoT Technology.

[4] B. Vamshi and R. V. Prasad, "Dynamic route planning framework for minimal air pollution exposure in urban road transportation systems," *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, Singapore, 2018, pp. 540-545.

[5] http://www.imdmumbai.gov.in/

[6] https://www.arduino.cc/

[7]https://firebase.google.com/