

# **SMART TRANSIT SYSTEM**

## **A PROJECT REPORT**

*Submitted by*

**VIGNESH BABU S**

**(211614104208)**

**VARUN M**

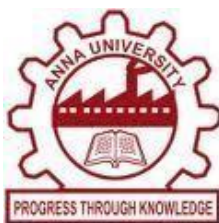
**(211614104200)**

*in partial fulfillment for the award of the degree  
of*

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**



**RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI**

**ANNA UNIVERSITY: CHENNAI 600 025**

## **BONAFIDE CERTIFICATE**

Certified that this project report “**SMART TRANSIT SYSTEM**” is the bonafide work of “**VIGNESH BABU S AND VARUN M**” who carried out the project work under my supervision.

### **SIGNATURE**

Dr. P.Kumar, Ph.D.,

### **HEAD OF THE DEPARTMENT**

Professor

Computer Science and Engineering

Rajalakshmi Engineering College

Rajalakshmi Nagar

Chennai-602105.

### **SIGNATURE**

Dr. K.Devaki, Ph.D.,

### **SUPERVISOR**

Professor

Computer Science and Engineering

Rajalakshmi Engineering College

Rajalakshmi Nagar

Chennai-602105.

This project report is submitted for viva voce examination to be held on

\_\_\_\_\_ .

**Internal Examiner**

**External Examiner**

## ACKNOWLEDGEMENT

First we thank the almighty god for the successful completion of the project. Our sincere thanks to our chairman **Mr. S.MEGANATHAN, B.E., F.I.E.**, for his sincere endeavor in educating us in his premier institution. We would also like to express our deep gratitude to our beloved Chairperson **Dr. (Mrs.). THANGAM MEGANATHAN, Ph.D.**, for her enthusiastic motivation which inspired us a lot in completing this project.

We also express our sincere gratitude to our college principal, **Dr.S.N.MURUGESAN, M.E., Ph.D.**, who helped us in providing the required facilities for completing the project.

We would like to thank and express our gratitude to **Dr. P.KUMAR, Ph.D., Head of the Department of Computer Science and Engineering** and our project guide **Dr.(Mrs.) Devaki.K, Ph.D.**, for her encouragement and guiding us throughout the project.

We would like to thank our project coordinator **Dr. K.Devaki, Ph.D.**, for her encouragement and guidance towards successful completion of this project.

We also extend our sincere thanks to all faculty members and supporting staffs for their direct and indirect involvement in successful completion of the project. We express our gratitude to our parents, friends and well-wishers for their encouragement and support.

**VIGNESH BABU S**

**VARUN M**

## **ABSTRACT**

Smart transit system (STS) is a major area in Smart city initiative of the Indian Government. A report published by the Union Transport Ministry on March 22, 2016, complains that some of the route buses are overcrowded as it carries twice the number of passengers than its seating capacity, overruling the RTO rules and restrictions. Hence this project is an imminent requirement for today's fast paced environment. The aim of this project is to develop a mobile application for Smart Transit System using IoT and Data Analysis technologies to provide info to the passengers about the bus's proximity, seat availability and arrival time estimate in advance and to optimize the number of buses plying through the routes for efficient fuel usage and human resources. The bus's proximity is determined by Automatic Vehicle Location (AVL) device which uses inbuilt GPS system of the smartphone or cell tower triangulation method to identify and upload the location information to the centralized server. The passenger request data for each bus is collected from the commuter interface and transmitted to the centralized database, which is further used in the analysis process, to obtain the final optimized number of buses that can be operated through that route. STS with such capabilities will be an integral part in the upcoming smart city development.

## TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	<b>ACKNOWLEDGEMENT</b>	<b>iii</b>
	<b>ABSTRACT</b>	<b>iv</b>
	<b>LIST OF TABLES</b>	<b>vii</b>
	<b>LIST OF FIGURES</b>	<b>viii</b>
	<b>LIST OF ABBREVIATIONS</b>	<b>x</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>1-3</b>
	1.1 SMART TRANSIT SYTEM	2
	1.2 OBJECTIVES	2
	1.3 ADVANTAGE	3
<b>2</b>	<b>LITERATURE REVIEW</b>	<b>4-6</b>
	2.1 EXISTING SYSTEM	4
	2.2 PROPOSED SYSTEM	6
<b>3</b>	<b>SYSTEM REQUIREMENTS SPECIFICATION</b>	<b>7-8</b>
	3.1 PLATFORM	7
	3.2 HARDWARE REQUIREMENTS	7
	3.3 SOFTWARE REQUIREMENTS	8
<b>4</b>	<b>SYSTEM DESIGN</b>	<b>9-18</b>
	4.1 SYSTEM ARCHITECTURE	9
	4.2 WORK FLOW OF THE PROPOSED SYSTEM	9
	4.3 USE CASE DIAGRAM	11
	4.4 MODULE DESCRIPTION	12
	4.4.1 PASSENGER MODULE	12
	4.4.2 CONDUCTOR MODULE	12

	4.4.3	BUS FREQUENCY OPTIMIZATION MODULE	13
	4.5	ALGORITHMS USED	14
	4.5.1	ARIMA ALGORITHM	14
	4.5.2	RANDOM FOREST TIME SERIES ALGORITHM	16
<b>5</b>		<b>SYSTEM IMPLEMENTATION AND TESTING</b>	<b>19-29</b>
	5.1	COMMUTER INTERFACE	19
	5.2	CONDUCTOR INTERFACE	21
	5.3	DATA ANALYSIS INTERFACE	24
	5.4	PROPOSED ALGORITHM	27
	5.5	UNIT TESTING	28
<b>6</b>		<b>RESULTS AND DISCUSSION</b>	<b>30-34</b>
	6.1	DATASET	30
	6.2	RESULTS	30
<b>7</b>		<b>CONCLUSION AND FUTURE ENHANCEMENTS</b>	<b>35</b>
<b>8</b>		<b>APPENDIX – SAMPLE CODE</b>	<b>37-40</b>
		<b>REFERENCES</b>	<b>41-42</b>

## **LIST OF TABLES**

<b>TABLE NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
5.1	Unit testing	28

## **LIST OF FIGURES**

<b>FIGURE NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
2.1	Functional requirements of Smart Transportation Application using Global Positioning System	5
2.2	Internal architecture of Smart phone application for connected vehicles and smart transportation	6
4.1	Architecture Diagram of STS project	9
4.2	Use case diagram of STS project	11
5.1	Search screen of passenger interface	19
5.2	Display of available buses	20
5.3	Route map and estimate time of selected bus	21
5.4	Conductor login interface	22
5.5	Booking screen	23
5.6	Closure screen for conductor booking	24
5.7	Welcome screen for analytic interface	25
5.8	Dataset viewing interface	26
6.1	Sample Trend for bus no 17D	30



6.2	Scatter plot for sample dataset	31
6.3	Optimization module	32
6.4	Optimization module	33
6.5	Dataset view step 1	33
6.6	Dataset view step 2	34
6.7	Dataset view step 3	34

## **LIST OF ABBREVIATIONS**

<b>API</b>	Application Programming Interface
<b>GPS</b>	Global Positioning System
<b>AVL</b>	Automatic Vehicle Location
<b>ARIMA</b>	Auto Regressive Integrated Moving Average
<b>AIC</b>	Akaike Information Criteria
<b>BIC</b>	Bayesian information criterion
<b>AICc</b>	Akaike Information Criteria with correction
<b>OS</b>	Operating System
<b>PC</b>	Personalized Computers
<b>UI</b>	User Interface

# **CHAPTER 1**

## **INTRODUCTION**

Cities are engines of growth for the economy of every nation, including India. With increasing urbanization, urban areas are expected to house 40% of India's population and contribute 75% of India's GDP by 2030. This requires the comprehensive development of physical, institutional, social and economic infrastructure. Development of Smart Cities is a step in that direction.

Public transportation is an integral part of smart city development. Therefore, over the years, a large number of initiatives have been undertaken by various State Governments and Central Ministries to usher in an era of the smart transport system. The government is seeking all possible measures to increase the profit obtained in public transport and also to cater the people's need. General public (i.e.) the commuters are unaware of the proximity of the buses and has to wait for a long time unsure of the bus status. These days' buses are not often tracked of their location. Moreover, the commuters have no idea about the exact timing that their required bus will reach their stop and also the seat(s) availability for them in the bus.

At the same time, the transport department is not able to determine when to leave a new service on that route, to maximize the profit as well as to satisfy the commuters need. Most of the time during the lean hours the bus goes empty and much of fuel and time is wasted.

This model introduces an IoT based application by providing clear information to the commuters availing the transport system in advance so it may avoid displeasure in their travel. And further, the emission of harmful air pollutants from the vehicles can be reduced if the traffic is properly analyzed.

Hence we proposed this method using latest technologies Raspberry Pi, smartphone, AVL(Automatic Vehicle Location) using GPS and data analytics, to provide optimization of the number of buses in operation, insights about the bus proximity for the commuters in advance of their travel, decrease in fuel expenditure and for comfortable travelling experience.

## **1.1 SMART TRANSIT SYSTEM**

This system is mainly focused on two objectives, which are, first to display the proximity of the bus to the commuters, seat availability and arriving time, second to optimize the number of buses plying through the predefined routes to reduce the loss of fuel expenditure.

The problem with the current system is that the buses either travel overloaded during peak hours (i.e.) a bus which has a capacity of carrying 72 people including 24 standings, carries twice the amount of people, or the bus travels with less occupancy rate during lean hours. On the commuters side, they are unaware of the bus proximity, arrival time and seat availability on the bus.

This system is designed to eliminate the above-mentioned problem by using the advanced technologies like IoT, Data Analytics and Cloud Computing.

This system comprises of three modules which are, commuter interface, conductor interface and the data analyst interface.

## **1.2 OBJECTIVES**

1. To develop a Smart Transit System that aim's to integrate information technology in the public transport system to provide a sophisticated solution to the commuters. The system helps the passengers to be aware of the time of

arrival of the bus they wanted to board and also to verify whether they have the required number of seats in that bus.

2. To display the updates about seats availability to the conductor in the bus.
3. Prediction of routes in which more number of commuters travel and suggest to increase the number of buses in that particular routes.
4. The commuters can access these information through their smart phones which increases the ease of access.
5. The commuters who don't possess a smart phone can make use of the IoT device installed in the bus stop to access the same set of features.
6. The exact location of the bus is also shown to the user at the time of selection of a bus.
7. To optimize the number of buses plying in the current routes and thereby reducing the CO<sub>2</sub> emission.
8. Efficient usage of the human resource (drivers & conductors).
9. To optimize the cost and thereby increase the revenue.

### **1.3 ADVANTAGE**

1. The commuter can pre-plan their travel according to the bus proximity, seat availability and arrival time.
2. Automatic updating of the bus location using the AVL system for the convenience of the commuter.
3. Efficient usage of the human resource and optimized number of buses for increased profit in the revenue.

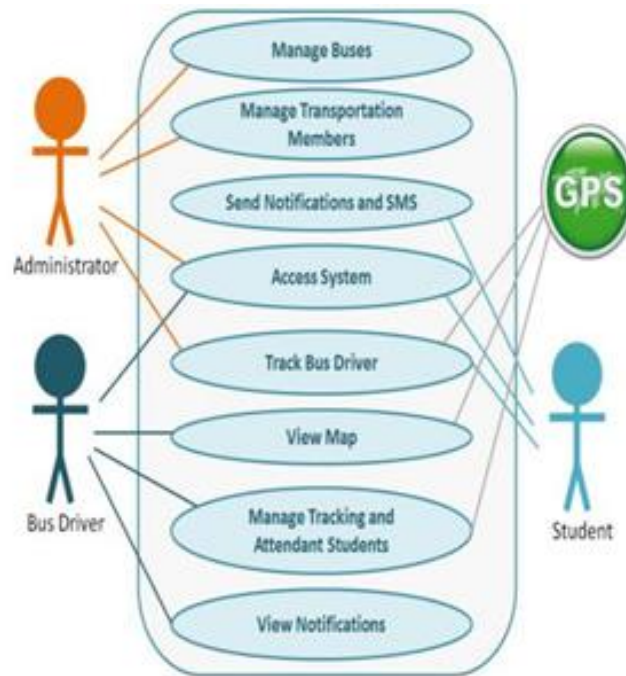
## **CHAPTER 2**

### **LITERATURE SURVEY**

#### **2.1 EXISTING SYSTEM**

1. Today's existing system implements the features independently, most applications are based on traffic analysis but doesn't consider the ease of access to the customer.
2. The data is often not recorded to gain useful insights. The availability of seat in a particular bus route at a particular time and the demand for a particular route is often neglected.
3. The supply of seat in a selected bus path at a selected time and the demand for a specific course is frequently not noted.

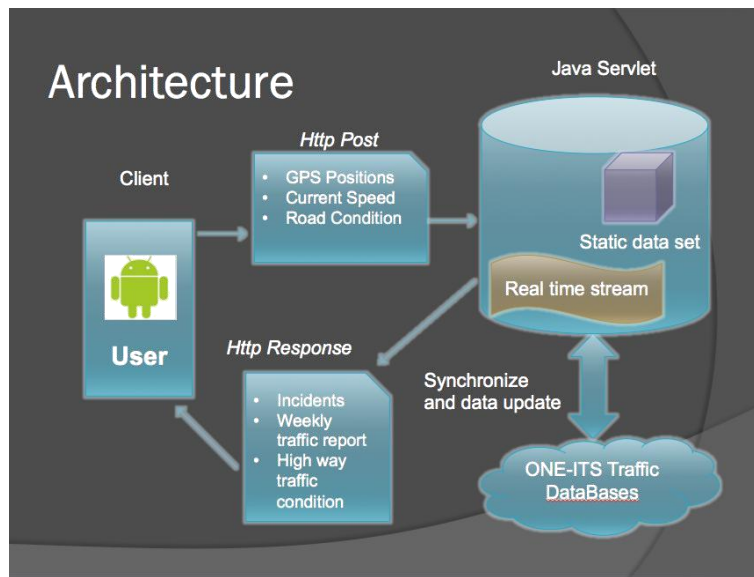
The advantages of the system as described in paper [1] is elaborated as such, Provides shortest route between source and destination. Displays approximate time of travel. Advanced reservation and seat availability details through short message service (SMS). The application is based on GPS and GNSS using mobile interface to access the information.



**Figure 2.1 Functional requirements of Smart Transportation Application using Global Positioning System**

This system falls short of some features like real time seat availability between any two stops as if it is not calculated so it can result in replication of seat availability. No optimization measures taken for the transport vehicles.

In [2] the Smart phone application for connected vehicles and smart transportation uses image processing and sensors installed at highways and signal to detect traffic congestion. It also gathers information about road conditions and time gap to move to next place according to GPS, to access the traffic intensity.



**Figure 2.2 Internal architecture of Smart phone application for connected vehicles and smart transportation**

This project is custom made for traffic identification and doesn't deal with seat availability and estimate time of boarding.

## 2.2 PROPOSED SYSTEM

1. Smart Transit System provides user interface through the users mobile as well as a touch screen installed at the bus station.
2. The GPS installed in each bus sends signal about its location to a centralized server, the server then provides information about the bus to the commuter's user interface for the user to plan their travel.
3. The seat availability is monitored and updated through the conductor's user interface (smartphone- web application).
4. Each request by the commuter is recorded in the centralized server to analyze the aggregated data and use time series analysis to identify the demand and optimize.



## **CHAPTER 3**

### **SYSTEM REQUIREMENT SPECIFICATIONS**

#### **3.1 PLATFORM**

This chapter describes the requirement analysis in accordance with the input and the resources and it also describes the implementation of the project with the technology used.

#### **3.2 HARDWARE REQUIREMENTS**

The hardware requirements may serve as the basis for a contract for the implementation of the system and should, therefore, be a complete and consistent specification of the whole system.

##### **PC Minimum Requirements: -**

- RAM : 4-8 GB
- Architecture : 32-bit
- Processor : Core 2 Duo
- Processor speed : Minimum 2 GHZ
- Hard disk : 500 GB

##### **Smart Transit System Requirements: -**

- RAM : 1GB
- Processor : Snapdragon 430
- Processor speed : 1.5GHZ
- Touch Screen Display
- Raspberry Pi

### **3.3 SOFTWARE REQUIREMENTS**

The software requirements are the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it.

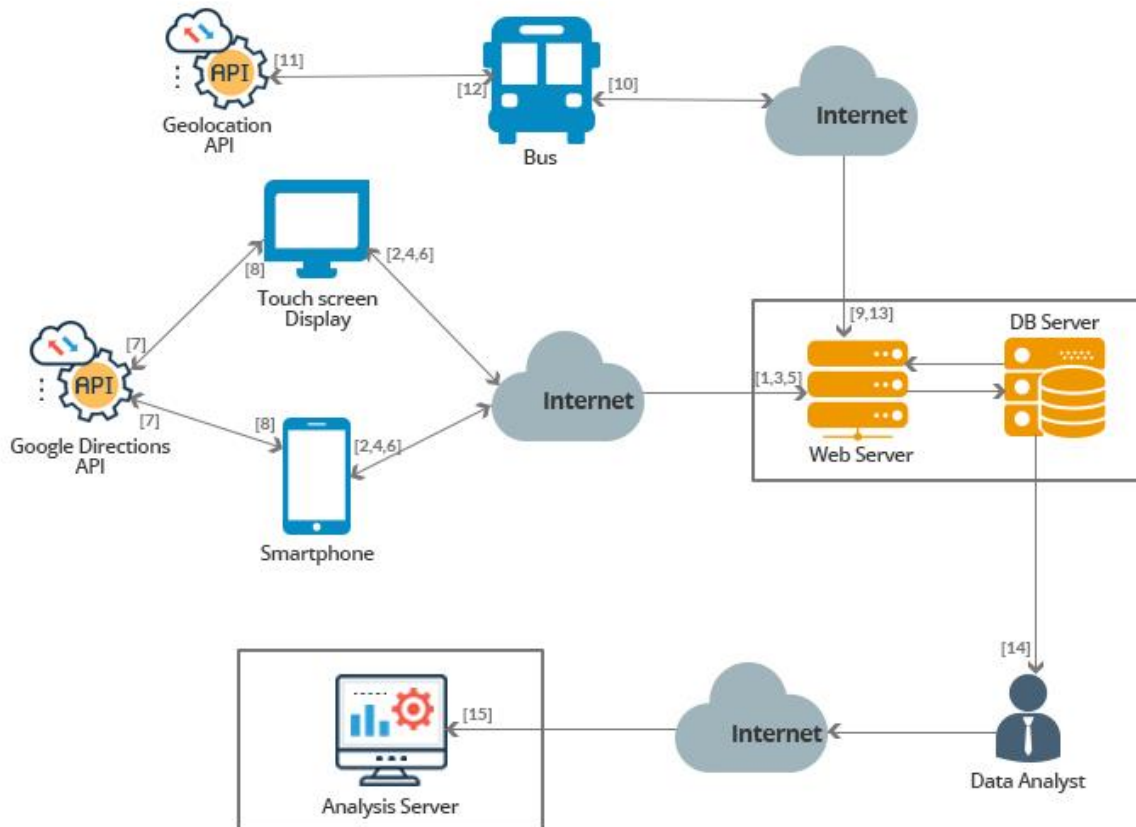
- Operating system : Windows 7/8/10
- Browser : Google Chrome/Mozilla Firefox
- Development Environment : Atom text editor, RStudio
- Languages : PHP, R, SHINY(Framework of R)
- Markup and styling : HTML 5.1 Ed. 2, CSS 4 , JavaScript
- Application Software : Microsoft excel

# CHAPTER 4

## SYSTEM DESIGN

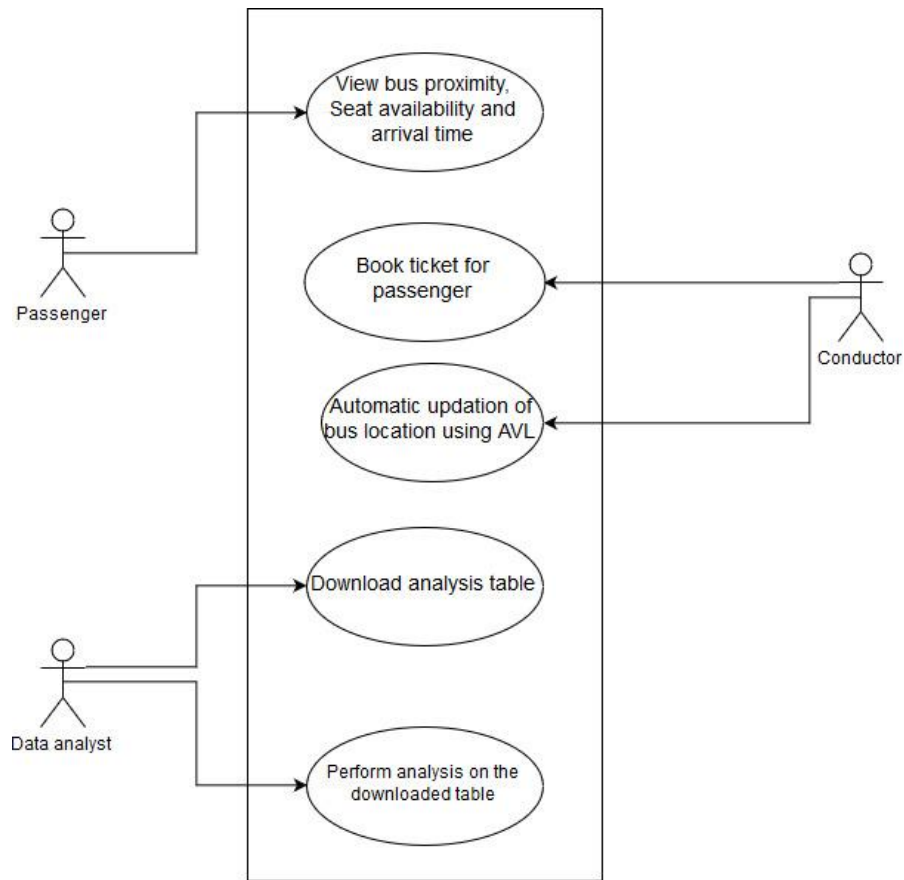
## SYSTEM DESIGN

## 4.1 SYSTEM ARCHITECTURE



- The request from each user are recorded in the server for the purpose of optimization using Data Mining algorithms.
- Each search is recorded with the date, segment of the day (e.g. Morning, Evening..).
- On identifying the bus the user boards it.
- The conductor while issuing the ticket has to enter the source, destination and the number of required tickets.
- The conductor on booking the ticket gets an update successful message, indicating the closure.
- In the analytic interface the transport department has some functionality
  - The user at transport department has to download the dataset which is recorded in the cloud server after entering the credentials.
  - The overall count for each bus is represented using a date vs count scatter plot.
  - The user can also determine the optimized number of buses to ply in a particular route by choosing the day, bus number and the segment of the day.

## 4.2 USE CASE DIAGRAM



**Figure 4.2 – Use case diagram of STS project**

The above use case diagram depicts that there are three characters(actors) involved in the system namely passenger, conductor and the data analyst.

The passenger can view the details of the bus at any instant. The details include the proximity of the bus, seat availability and ETA of the bus.

The conductor can book the ticket for the passenger onboard, consequently after the booking an algorithm runs at the backend which calculates the number of free seats left, which is to be viewed by the passenger from his/her interface.

The data analyst can download the table containing the data of the total number of request collected for each bus at a given time. With this table he then uploads it to

the analysis interface where the analysis is done with Time series algorithms and final results are displayed.

## **4.3 MODULE DESCRIPTION**

### **4.3.1 COMMUTER MODULE**

The passenger uses either the touch screen display connected to the raspberry pi or their smartphone as an interface for the passenger to query the database server to know the information about their boarding bus.

Step i) The user needs to enter his/her destination address into the UI.

Step ii) The list of buses which matches the source and destination stops entered by the passenger are displayed.

Step iii) The passenger needs to select a desired bus from the list displayed.

Step iv) On the next page the details like bus proximity, free seat availability and Estimated Arrival Time (ETA) of the selected bus will be displayed for the passenger's to know.

### **4.3.2 CONDUCTOR MODULE**

The conductor uses the smartphone connected to the web application as an interface for performing tasks. Each conductor is given a login username and password in advance. The conductor module updates the current location of the bus to the database automatically. Steps to be followed in this module are

Step i) The conductor has to enter the username and password on the login page.

Step ii) After submission of the credentials it directs to the bus's personalized dashboard page where the ticket booking form is displayed.

Step iii) The conductor has to enter the source, destination and the number of seats required by the passenger(s) and click on the Book button.

Step iv) The conductor will be displayed with an “update successful” prompt box. In the backend, the proposed algorithm will calculate the free seats remaining in the bus based on the input given by the conductor in the dashboard interface.

### **4.3.3 BUS FREQUENCY OPTIMIZATION MODULE**

One of the primary objective is to optimize the number of busses flying in a particular route. It is better to know the optimum frequency of busses to prevent wastage of fuel and to provide a hassle free ride to the prospective commuters. The admin dashboard created using Shiny R serves this purpose of optimizing the frequency. This optimization is done with the help of algorithms like Auto Regressive Integrated Moving Average and Random Forest Regression. The ARIMA algorithm is used to predict the trend depicting the frequency of commuters boarding a particular bus route. The Random Forest algorithm on the other hand is used to estimate the number of busses to fly in a particular. Parameters such as time interval, date, route etc. are considered. These data are obtained by accessing the database which gets updates as and when a commuter access the system. The data being fed into the regressor along with the custom inputs provided by the admin – the day and time interval – enable to produce the exact optimised number of busses. On inferring this output the admin can take necessary steps to add or reduce the number of busses for a route.

## 4.4 ALGORITHMS USED

### 4.4.1 Auto Regressive Integrated Moving Average Algorithm

ARIMA is a combination of 3 parts i.e. AR (*AutoRegressive*), I (*Integrated*), and MA (*Moving Average*). A convenient notation for ARIMA model is ARIMA(p,d,q). Here p,d, and q are the levels for each of the AR, I, and MA parts. Each of these three parts is an effort to make the final residuals display a white noise pattern (or no pattern at all). In each step of ARIMA modeling, time series data is passed through these 3 parts

#### 1st Pass of ARIMA to extract Information

**Integrated (I)** – subtract time series with its lagged series to extract trends from the data

In this pass of ARIMA juicer, we extract trend(s) from the original time series data. Differencing is one of the most commonly used mechanisms for extraction of trends. Here, the original series is subtracted with it's lagged series e.g. November's sales values are subtracted with October's values to produce trend-less residual series. The formulae for different orders of differencing are as follow:

No Differencing (d=0)	$Y'_t = Y_t$
1st Differencing (d=1)	$Y'_t = Y_t - Y_{t-1}$
2nd Differencing (d=2)	$Y'_t = Y_t - Y_{t-1} - (Y_{t-1} - Y_{t-2}) = Y_t - 2 \times Y_{t-1} + Y_{t-2}$



## 2nd Pass of ARIMA to extract Information

**AutoRegressive (AR)** – extract the influence of the previous periods' values on the current period

After the time series data is made stationary through the *integrated* (I) pass, the AR part of the ARIMA juicer gets activated. As the name auto-regression suggests, here we try to extract the influence of the values of previous periods on the current period

This is done through developing a regression model with the time lagged period values as independent or predictor variables. The general form of the equation for this regression model is shown below.

$$Y_t = c + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + e_t$$

AR model of order 1 i.e.  $p=1$  or ARIMA(1,0,0) is represented by the following regression equation

$$Y_t = c + \phi_1 Y_{t-1} + e_t$$

## 3rd Pass of ARIMA to Extract Information

**Moving Average (MA)** – extract the influence of the previous period's error terms on the current period's error

MA involves finding relationships between the previous periods' error terms on the current period's error term. *Moving Average* (MA) part of ARIMA is developed with the following simple multiple linear regression values with the lagged error values as independent or predictor variables.

$$Y_t = c + e_t + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \cdots + \theta_q e_{t-q}$$

MA model of order 1 i.e.  $q=1$  or ARIMA(0,0,1) is represented by the following regression equation

$$Y_t = c + e_t + \theta_1 e_{t-1}$$

The stochastic data is unpredictable and may change in different time frame, therefore Auto-ARIMA is used in the application. This chooses the best fit of ARIMA version by accessing the parameters. The data is initially converted into time series data and the ARIMA function takes these data as parameters along with other parameters such as approximation and trace. The approximation is taken a default value 'FALSE' so as to decrease the search time for the best fit ARIMA model. Trace is used to display the considered list of ARIMA models, therefore it is set as 'FALSE'.

The value parameters of ARIMA, such as number of autoregressive terms (p), number of non-seasonal differences needed for stationarity (d), number of lagged forecast errors in the prediction equation (q) are automatically decided by the obtained best AIC, AICc or BIC value. On application of this time series algorithm we get the trend for the commuters' usage data.

#### **4.4.2 RANDOM FOREST REGRESSION ALGORITHM**

Random Forests are similar to a famous Ensemble technique called Bagging but have a different tweak in it. In Random Forests the idea is to decorrelate the several trees which are generated on the different bootstrapped samples from training Data. Then we simply reduce the Variance in the Trees by averaging them.

Averaging the Trees helps us to reduce the variance and also improve the Performance of Decision Trees on Test Set and eventually avoid Overfitting.

The idea is to build lots of Trees in such a way to make the Correlation between the Trees smaller.

Another major difference is that we only consider a Random subset of predictors each time we do a split on training examples. Whereas usually in Trees we find all the predictors while doing a split and choose best amongst them. Typically  $m = p \cdot \sqrt{p}$  where  $p$  are the number of predictors.

Now it seems crazy to throw away lots of predictors, but it makes sense because the effect of doing so is that each tree uses different predictors to split data at various times. This means that 2 trees generated on same training data will have randomly different variables selected at each split, hence this is how the trees will get de-correlated and will be independent of each other. Another great thing about Random Forests and Bagging is that we can keep on adding more and more big bushy trees and that won't hurt us because at the end we are just going to average them out which will reduce the variance by the factor of the number of Trees  $T$  itself.

STEPS:

- loading the required packages
- Fitting the Random Forest to a regressor
- Use predict function with regressor and data frame's column value as parameter to predict the optimal busses.
- Plot the output

### **Formulation and Parameters:**

Ensemble method of regression helps in estimating a more accurate prediction by applying the decision tree regressor multiple times in the stochastic data which is gathered.

The parameters such as day of week as X and count Y is fed into the regressor along with number of trees.

$$F_n = \frac{1}{B} \sum_{b=1}^B f_b(x') = Y' \quad (1)$$

where  $x'$  consist of the training data antecedent X- Day of the week (training data) and consequent Y – count (training data),  $Y'$ -count of searches (response), B- number of bags

The summation of the predicted values  $f_b(x')$  (for each bag) is divided by the total number of bags to get the average of all the outputs. In this way high accuracy in estimation of bus is obtained by applying the algorithm.

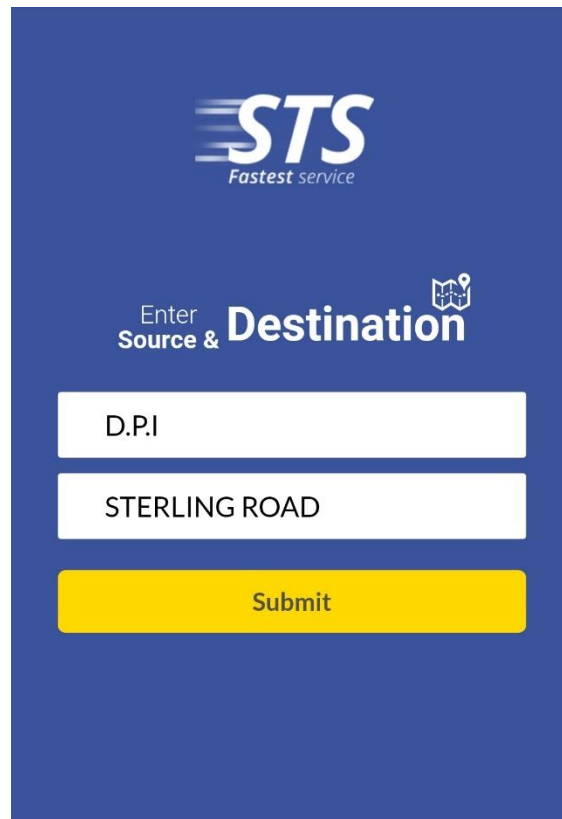
## CHAPTER 5

### SYSTEM IMPLEMENTATION AND TESTING

#### 5.1 COMMUTER INTERFACE

The commuter interface can be accessed from either the touchscreen display device connected to the raspberry pi installed at each bus stop or the passenger can use their own smartphones to access the web application. No authentication procedure is required for entering the interface, thus the user's personal data is secured.

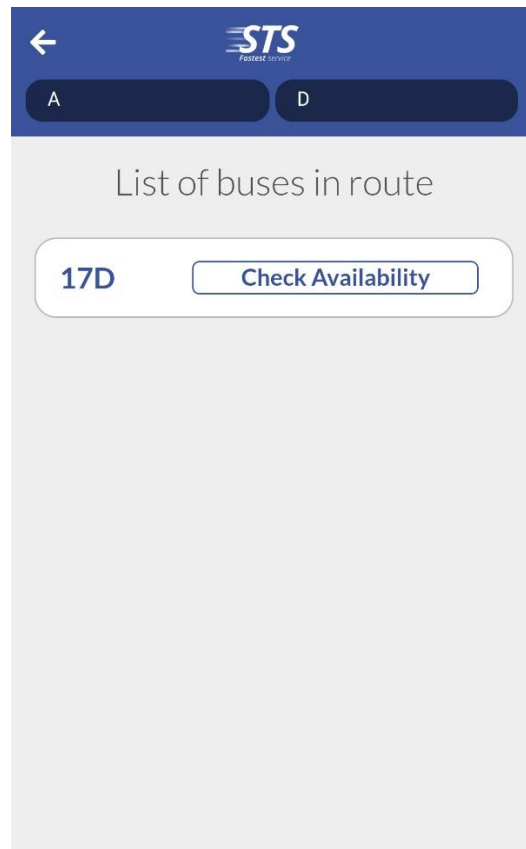
The implementation screens are shown below:

The image shows a digital interface for STS (Sterling Transport Service) with a blue background. At the top center is the STS logo, which includes the letters 'STS' in a large, white, sans-serif font with horizontal motion lines to the left, and the tagline 'Fastest service' in a smaller, italicized font below it. Below the logo, the text 'Enter Source &' is in white, followed by the word 'Destination' in a larger, bold white font. To the right of 'Destination' is a small white icon of a map with a location pin. Below this text are two white rectangular input fields. The first field contains the text 'D.P.I' and the second field contains 'STERLING ROAD'. Below these fields is a prominent yellow rectangular button with the word 'Submit' in black text.

**Figure 5.1 - Search screen of passenger interface**

This is the first screen that the passenger will be prompted. Here the passenger has to enter the source stop (i.e.) from where he wishes to board the bus and then enter the destination stop in the second input box. The stop names can be auto-completed by selecting an autocomplete suggestion as the user begins to enter the stop name.

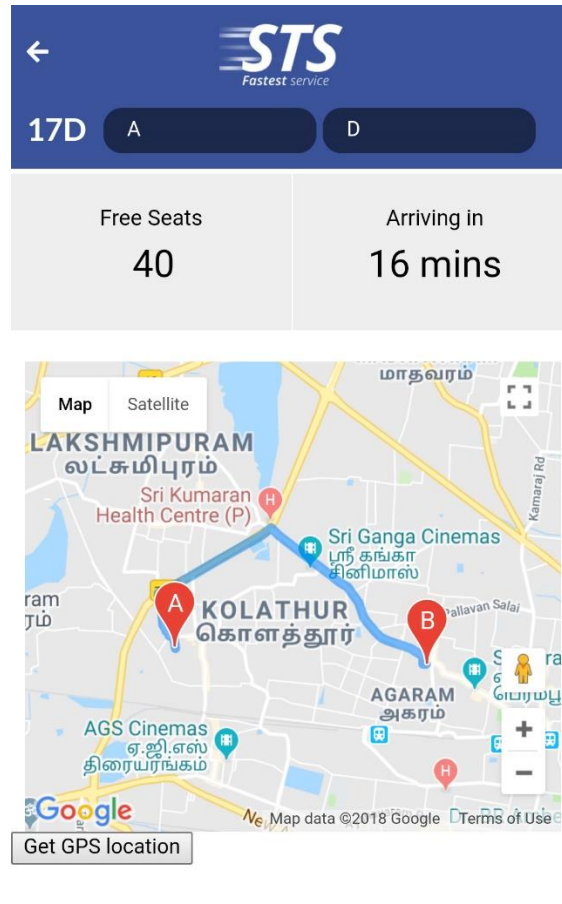
After entering the details, the user needs to press the submit button for passing the values to the database to be queried



**Figure 5.2 - Display of available buses**

This is the second screen which the user will be directed to after submitting the details in the [Figure 5.1]. This screen displays the list of bus numbers fetched from the database whose stop names match the source and destination stop name entered

by the user in [Figure 5.1]. The user can click on check availability button of any bus number to further know the details about that bus.



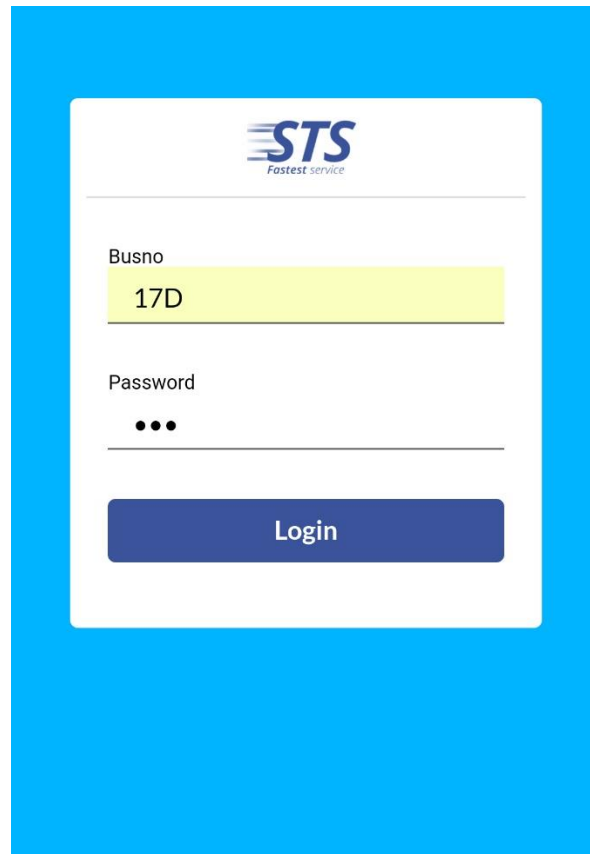
**Figure 5.3 - Route map and estimate time of selected bus**

This is the third page that the user will be directed to when he clicks the Check Availability button on Screen 2 [Figure 5.2]. On this screen, the user has to press the Get GPS location button initially to get the ETA and bus's current location proximity details. The user can also check the number of free seats available on the bus and plan their travel accordingly.

## 5.2 CONDUCTOR INTERFACE

The conductor interface can be accessed only by the conductor on their smartphones. Any other general user trying to enter into conductor interface will be prompted with a login page where the secret credentials should be entered for further access. The credentials are made known only to the conductor, in advance. The conductor interface also updates the current location of the bus (latitude, longitude) to the database automatically. This process happens at the backend automatically thus avoiding the conductor's effort.

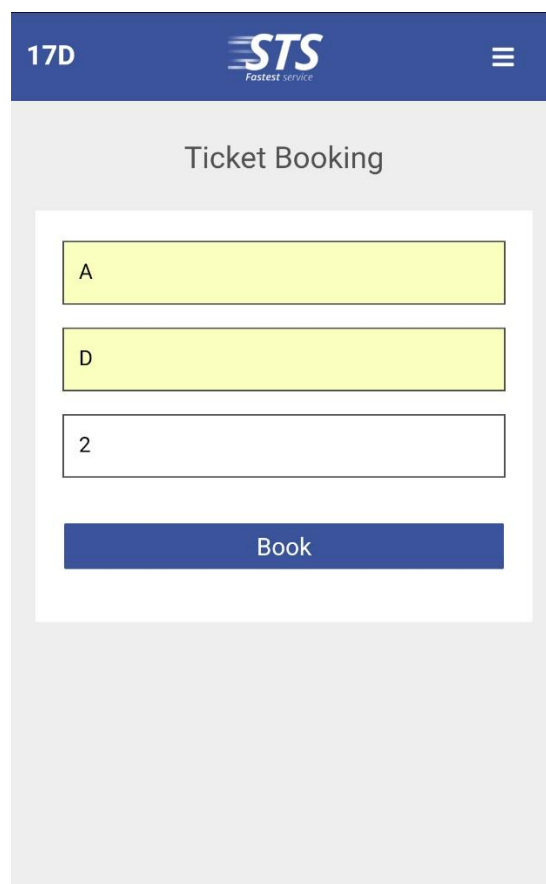
The implementation screens of the conductor interface are below:-

A screenshot of a mobile application login screen for a conductor. The screen has a bright blue background. In the center is a white rounded rectangle containing the login form. At the top of the white rectangle is the 'STS' logo with the tagline 'Fastest service' below it. Below the logo is a horizontal line. Underneath the line is the label 'Busno' followed by a yellow input field containing the text '17D'. Below this is the label 'Password' followed by a white input field with three black dots representing a masked password. Another horizontal line is below the password field. At the bottom of the white rectangle is a dark blue button with the word 'Login' in white text.

**Figure 5.4 - Conductor login interface**



This is the login page which will be displayed to each conductor at the beginning of the web application. The conductor has to enter the credentials (i.e.) the Bus number and the password into the input fields then click on the login button. The credentials will be validated with the database values. If the validation is successful they will be directed to next page, else the same page will be displayed again.



17D

STS  
Fastest service

Menu icon

Ticket Booking

A

D

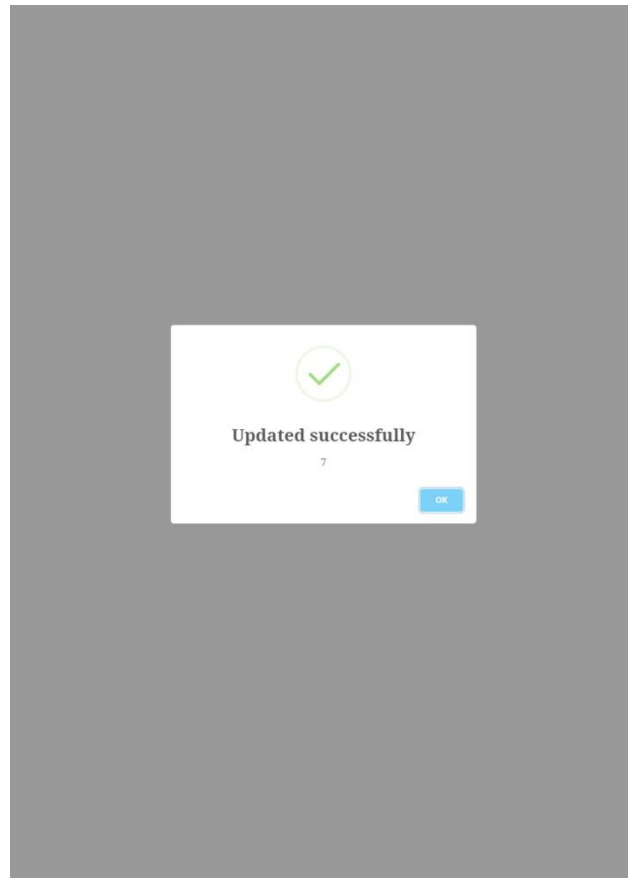
2

Book

**Figure 5.5 - Booking screen**

This page is the personalized page for each conductor according to their bus numbers. The page contains a ticket booking form where the conductor has to enter

the source and destination in the input fields and enter the number of seats requested by the passengers onboard, at last, the conductor has to click the Book button after entering the details above.

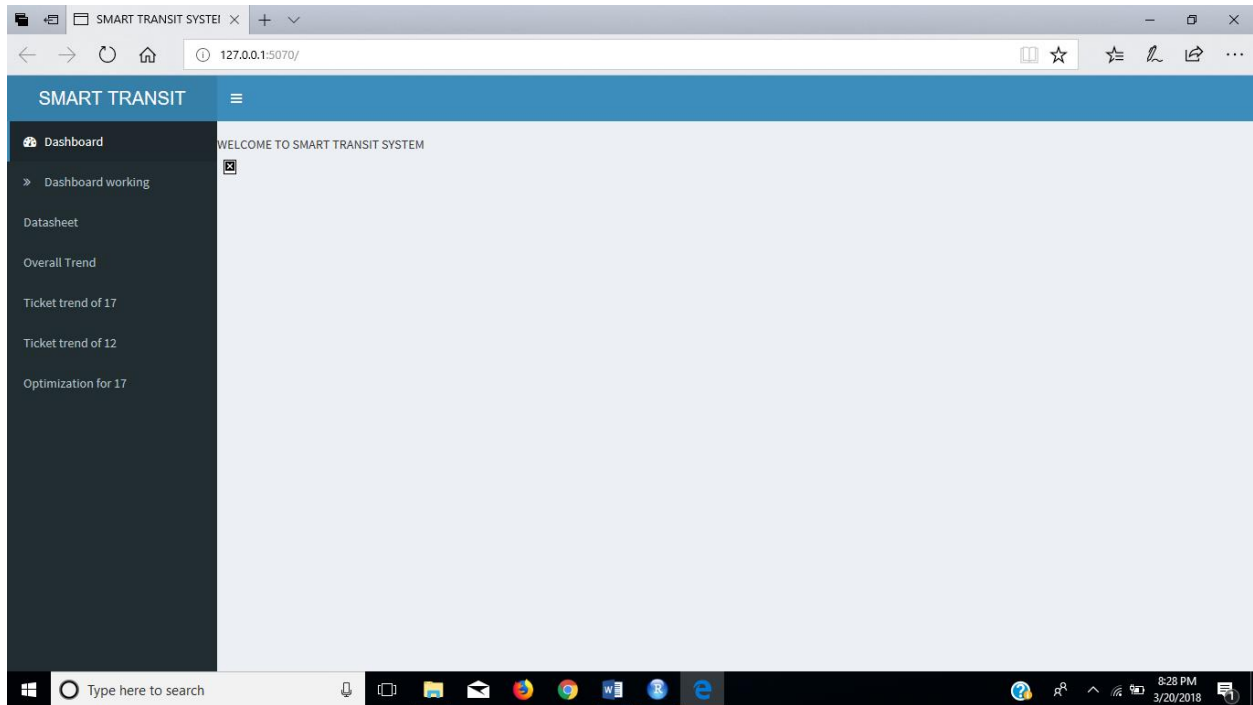


**Figure 5.6 - Closure screen for conductor booking**

This is a confirmation message box which contains the updated successful message when the booking is done successfully and the box also contains the information which says about the total number of passengers onboard currently on the vehicle. The conductor can press the ok button to redirect to the Screen 2[Figure 5.5], where he can continue to add next ticket to the following passenger on the bus.

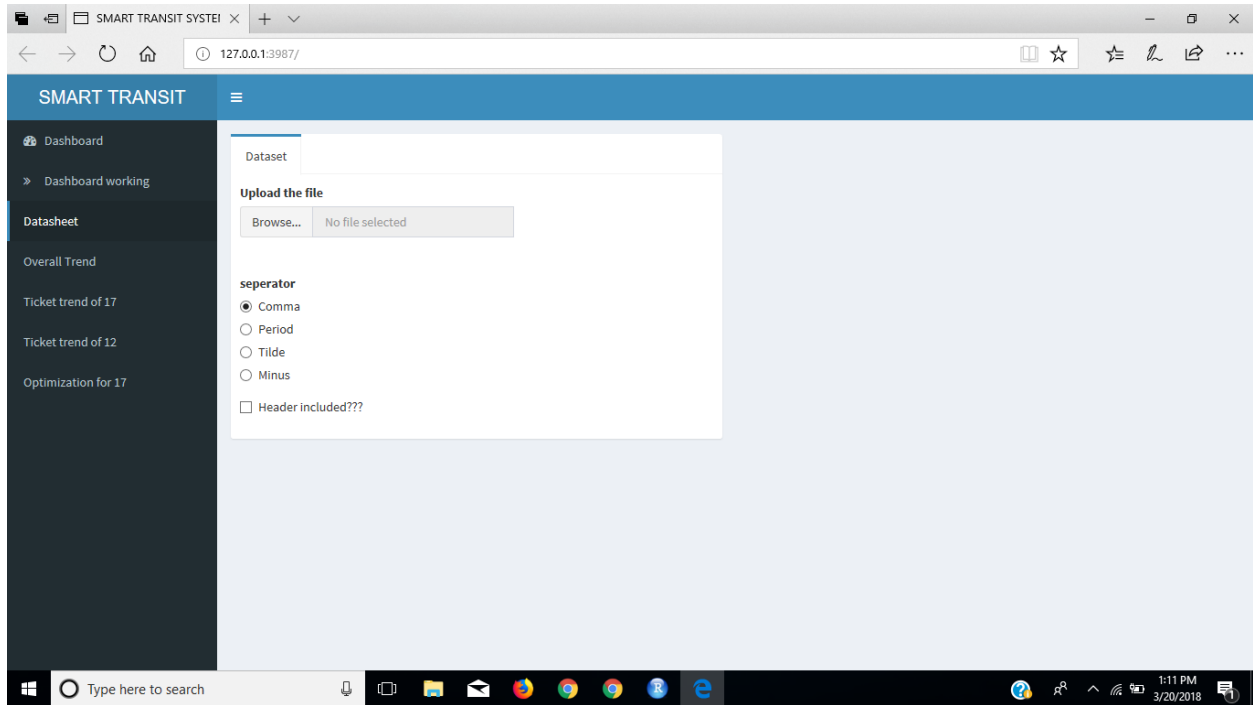
## 5.3 DATA ANALYSIS INTERFACE

The data analysis or the admin interface consist of certain functionalities which enable the administrator to view and infer results and eventually utilize those results in the decision making process. The inference is obtained by running predictive algorithms on the accumulated data.



**Figure 5.7 - Welcome screen for analytic interface**

The “Dashboard Working” tab displays all the requests which have been recorded in a graphical format. The bus number is represented by size of the points and the time interval is represented using colors. The count vs date plot is depicted.



**Figure 5.8 - Dataset viewing interface**

Datasheet tab is used to view the data which is being processed. It can accommodate comma separated value, tab separated value etc.. The admin has to browse through the folder and open the dataset.

The ticket trend for bus number 17 represents the predicted ticket sales for the upcoming dates. This is predicted with the help of ARIMA algorithm which handles the dataset. The blue projection represents the future estimate and the red and orange projection estimates the maximum and minimum limit.

The optimization for 17 helps to estimate the number of busses flying in that route for a particular day and time interval. The Random Forest regression model is run to produce the output for the decision making process. The optimal number of busses is displayed in the textbox after the admin clicks the radio buttons and submit button. The number which is displayed can be considered for increasing or decreasing the frequency.

## 5.4 PROPOSED ALGORITHM

This system uses an algorithm as given below for calculating the seat availability in the bus. The pre-requisites for the algorithm are the boarding count and the get down count values which will be updated each time a ticket is booked in the conductor component and the total number of seat capacity of the bus value will be predefined in the database by then administrator. The proposed algorithm is as follows:

Algorithm: Algorithm for Vacant seat count identification

Input: Boarding\_stop, Destination\_stop, Database D

Globals:

const Bs: Boarding\_stop

const Ls: Leaving\_stop

const TS: Total number of seats in the bus

TBC: Total\_boarding\_count = NULL

TLC: Total\_leaving\_count = NULL

Bc: Boarding\_count

Lc: Leaving\_count

Output: Free\_seats, available in the bus

Method:

(1) Fetch the Bc and Lc from the D

(2) Calculate sum of Bc and Lc

for (i=1; i<Bs; i++)

TBC = TBC + Bc[i];

$TLC = TLC + Lc[i];$

(3) To calculate the free seats

$Free\_seats = TS - TBC + TLC$

The above algorithm can be generically applied in problems where dynamic change in the input parameter value competing for available resources occurs and calculation of the present available resource need to be calculated.

## 5.5 UNIT TESTING

Table no 5.1 – Unit Testing table

Test case ID	Test Description	Test Data	Expected Result	Actual Result
Login_01	Entering wrong busnumber and password	Bus number:abcd Password:1234	The validation fails and same login page is reloaded	The validation fails and same login page is reloaded
Login_02	Entering correct busnumber but wrong password	Bus number :17D Password:idjssa	The validation fails and same login page is reloaded	The validation fails and same login page is reloaded

Login_03	Entering correct Busnumber and password	Bus number:17D  Password: 17D@2018	Validation successful. Redirected to next page.	Validation successful. Redirected to next page.
----------	--	--	--	--

## **CHAPTER 6**

### **RESULTS AND DISCUSSION**

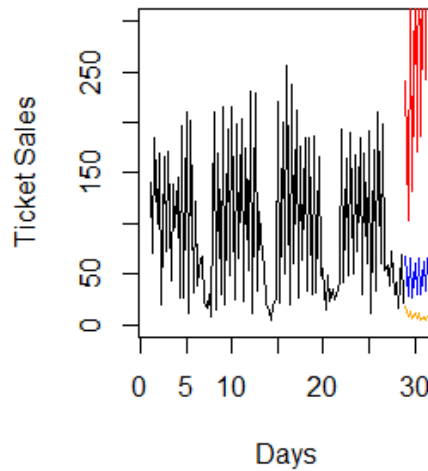
#### **6.1 DATASET**

The dataset which is accessed by the admin dashboard consist of attributes such as bus number, date, day, time interval of request, accumulated count. These attributes are obtained from the user interface (i.e.) on click of the bus number, while searching, accessing the location and checking the seat availability. Each search is recorded as request at that time interval. The aggregated count along with other parameters are considered for prediction and optimization.

#### **6.2 RESULTS**

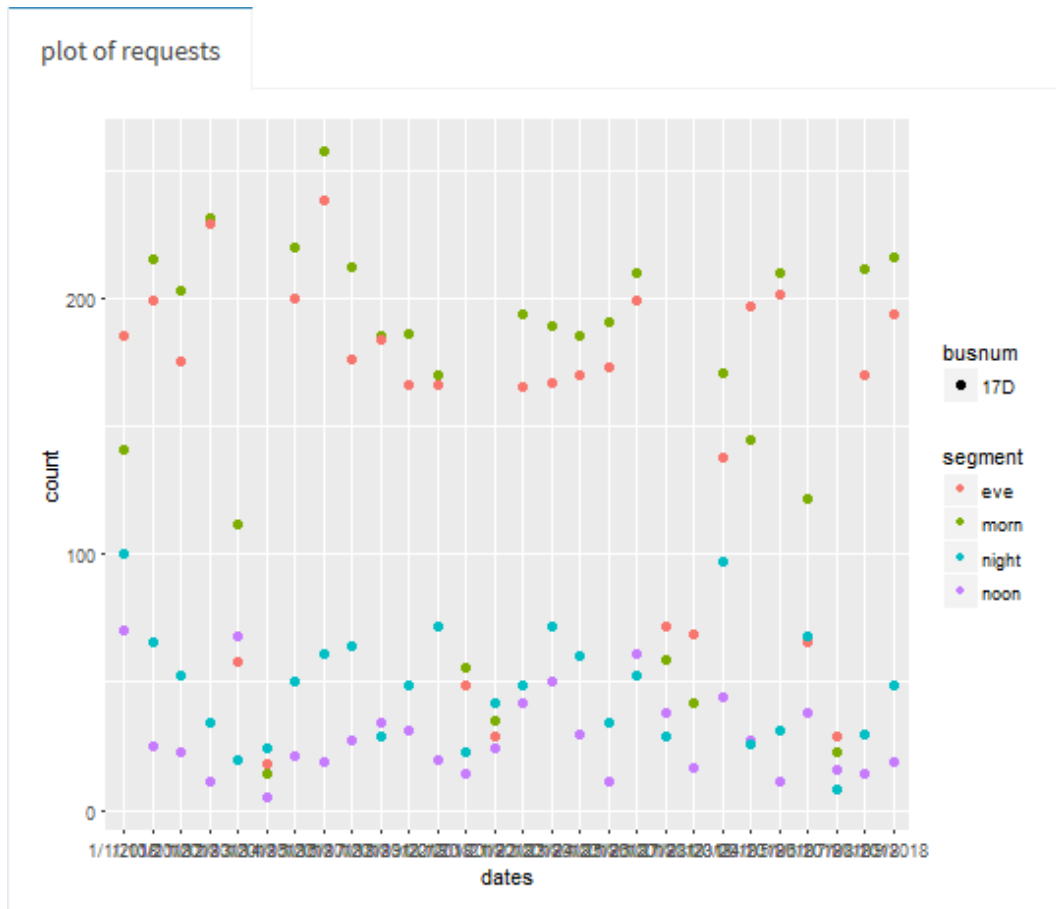
The trend for bus no 17 is represented. The black line represents the data considered and the blue line represents the trend projected. The orange line represents the maximum and minimum deviations.





**Figure 6.1 - Sample Trend for bus no 17D**

The graph represents the total requests accessed for all the busses. The time interval is represented by colors and the bus number to which each request pertains is represented by the size of the point. The x-axis represents the date of the requests and y-axis represents the total count of requests that has been accumulated for each bus.



**Figure 6.2 - Scatter plot for sample dataset**

The optimization of number of busses requires 2 controls, namely day and interval. These inputs are customized by using radio button which the admin is required to click. On click of the submit button the algorithm runs and produces the optimal count of busses.

SMART TRANSIT

☰

Dashboard
Dashboard working
Datasheet
Overall Trend
Ticket trend of 17
Ticket trend of 12
Optimization

for Day of week

☒ Mon
☐ Tue
☐ Wed
☐ Thur
☐ Fri
☐ Sat
☐ Sun

busnumber

☒ 17D
☐ 12G

for segment of week

☒ morn
☐ noon
☐ eve
☐ night

CHECK

4

**Figure 6.3 - Optimization module**

SMART TRANSIT

☰

Dashboard
Dashboard working
Datasheet
Overall Trend
Ticket trend of 17
Ticket trend of 12
Optimization

for Day of week

☒ Mon
☐ Tue
☐ Wed
☐ Thur
☐ Fri
☐ Sat
☐ Sun

busnumber

☒ 17D
☐ 12G

for segment of week

☐ morn
☒ noon
☐ eve
☐ night

CHECK

1

**Figure 6.4 - Optimization module**

33

The admin can also view the data which is being processed by browsing through the functionality provided.

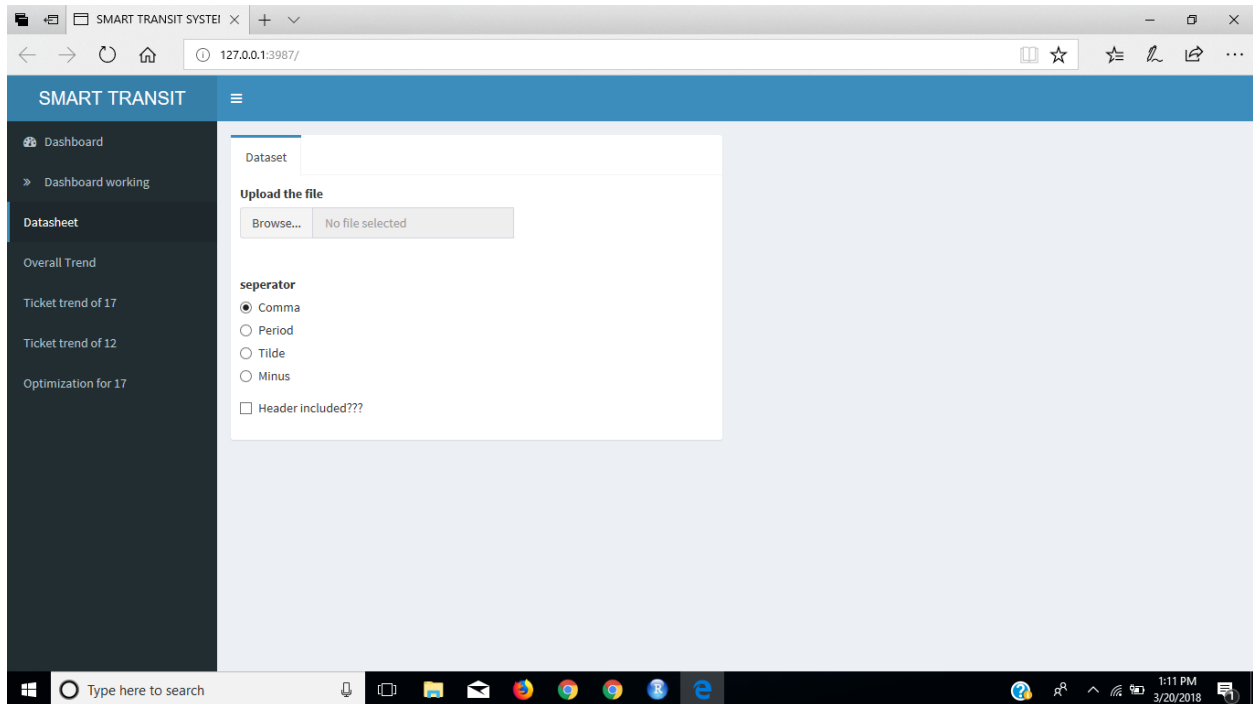


Figure 6.5 - Dataset view step 1

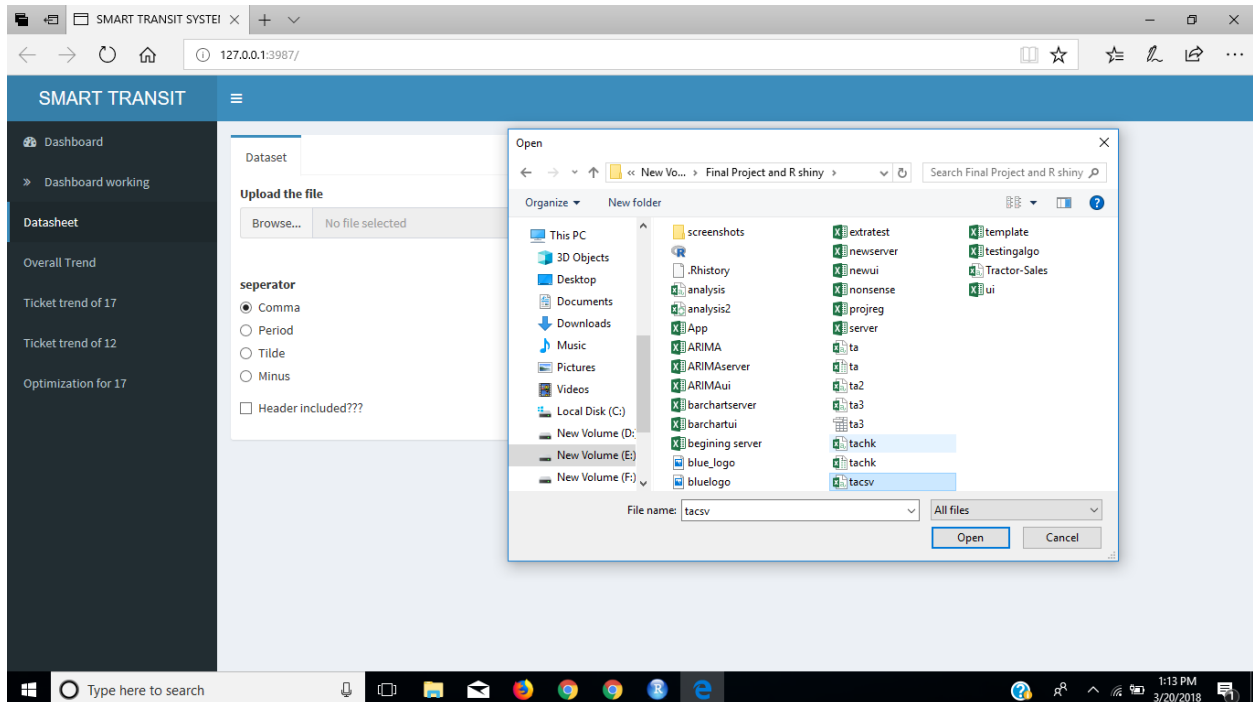
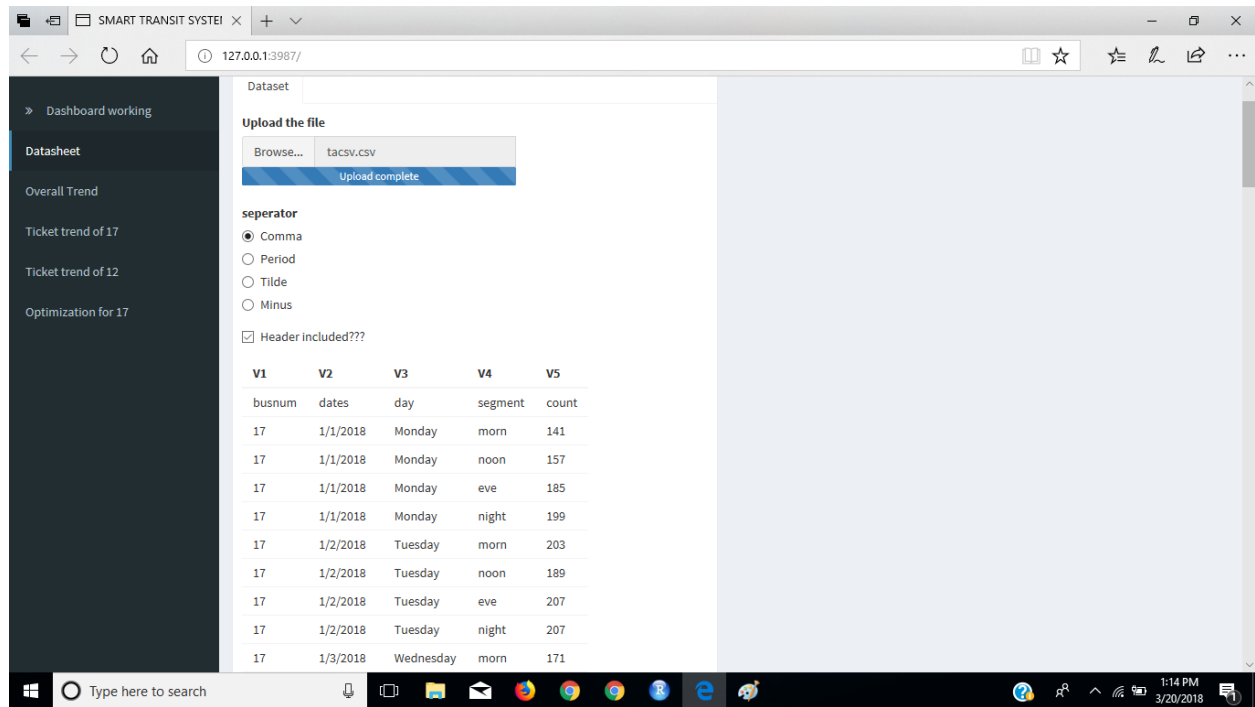


Figure 6.6 - Dataset view step 2



**Figure 6.7 - Dataset view step 3**

## **CHAPTER 7**

### **CONCLUSION AND FUTURE ENHANCEMENTS**

#### **7.1 CONCLUSION**

We have developed a mobile application for Smart Transit System to help the commuters to take an informed choice about their travel and to optimize the number of buses using data mining algorithms, thereby optimizing the cost of operation, fuel consumption, pollution, traffic management and at the same time serving the commuters requirement. The experiments conducted for the sample data set with the proposed method produced good results and predicted the optimized number of buses. The prediction made for optimizing the bus can be applied to a robust, scalable dataset.

#### **7.2 FUTURE ENHANCEMENTS**

- Real time data analytics report on public displays
- Multi Lingual support and localization of the apps and displays.
- Integration support with other IoT devices for travel data enhancement

## APPENDIX - SAMPLE CODE

PHP CODE:

```
<?php
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "smart";
    // Create connection
    $conn = mysqli_connect($servername, $username, $password , $dbname);
    // Check connection
    if (!$conn) {
        die("Connection failed: " . mysqli_connect_error());
    }
    session_start();
    if(isset($_GET["busno"])){
        $busno = $_GET["busno"];
        $_SESSION["busno"] = $busno;
    }
    if(isset($_SESSION["source"], $_SESSION["destination"])){
        $source = $_SESSION["source"];
        $destination = $_SESSION["destination"];
    }
    $count = 0;
    $sql0 = "SELECT stops FROM ".$busno." WHERE busno = '".$busno.'";";
    $res0 = mysqli_query($conn, $sql0);
    if(mysqli_num_rows($res0) > 0){
        while($row = mysqli_fetch_assoc($res0)){
```

```

        $stops[] = $row["stops"];
        $count++;
    }
}

$sql1 = "SELECT seq FROM ".$busno." WHERE stops = ''.$source.'";
$res1 = mysqli_query($conn , $sql1);
if(mysqli_num_rows($res1) > 0){
    while($row = mysqli_fetch_assoc($res1)){
        $source_seq = $row["seq"];
    }
}

$sql2 = "SELECT sum(bookcount),sum(getdowncount) FROM ".$busno."
WHERE seq BETWEEN 1 AND ".$source_seq."";
$res2 = mysqli_query($conn,$sql2);
if(mysqli_num_rows($res2) > 0){
    while($row = mysqli_fetch_assoc($res2)){
        $bookcountTotal = $row["sum(bookcount)"];
        $getdowncountTotal = $row["sum(getdowncount)"];
    }
}

$sql3 = "SELECT seat FROM listofbusses WHERE busno = ''.$busno.'";
$res3 = mysqli_query($conn,$sql3);
if(mysqli_num_rows($res3) > 0){
    while($row = mysqli_fetch_assoc($res3)){
        $TotalSeat = $row["seat"];
    }
}

$freeSeat = $TotalSeat - $bookcountTotal + $getdowncountTotal;
mysqli_close($conn); ?>

```



Shiny R code:

```
output$plot3<-renderPlot({
  analysis<-read.csv("tacsv123.csv")
  colnames(analysis)<-c("busnum","dates","day","segment","count")
  factor(analysis$segment)
  analysis$segment<-factor(analysis$segment)
  factor(analysis$busnum)
  analysis$busnum<-factor(analysis$busnum)
  data = analysis[which(analysis$busnum=='12G'),]
  data = data[,c(2,5)]
  data = ts(data[,2],start = c(1,1),frequency = 4)
  require(forecast)
  ARIMAfit = auto.arima(log10(data), approximation=FALSE,trace=FALSE)
  par(mfrow = c(1,1))
  pred = predict(ARIMAfit, n.ahead = 36)
  plot(data,type='l',xlim=c(1,31),ylim=c(100,800),xlab = 'Days',ylab = 'Ticket
Sales')
  lines(10^(pred$pred),col='blue')
  lines(10^(pred$pred+2*pred$se),col='red')
  lines(10^(pred$pred-2*pred$se),col='orange')
})
renderText({
  ds<-read.csv("tacsv123.csv",nrows=111)

  ds$day=factor(ds$day,levels=c('Monday','Tuesday','Wednesday','Thursday','Friday'
,'Saturday','Sunday'),
    labels=c(1,2,3,4,5,6,7))
  ds$day<-as.numeric(as.character(ds$day))
  ds$segment=factor(ds$segment,levels=c('morn','noon','eve','night'),labels
c(1,2,3,4))
  =
```

```

ds$segment<-as.numeric(as.character(ds$segment))
ds$busnum=factor(ds$busnum,levels=c('17D','12G'),labels = c(1,2))
ds$busnum<-as.numeric(as.character(ds$busnum))
pk= as.numeric(input$sec)
pk1=as.numeric(input$dow)
pk2=as.numeric(input$busno)
ds=ds[which(ds$busnum==pk2),]
ds=ds[which(ds$segment==pk),]
ds=ds[,c(3,5)]
require(caTools)
require(randomForest)
set.seed(1234)
regressor=randomForest(x=ds[1],y=ds$count,ntree = 1000)
ypred=predict(regressor,newdata = data.frame(day=pk1))
paste( as.character(round(ypred/45)) )
})

```

## REFERENCES

- [1] Albert Nagy, József Tick (2016), “Improving Transport Management with Big Data Analytics”, SISY 2016 • IEEE 14th International Symposium on Intelligent Systems and Informatics • August 29-31, 2016, Subotica, Serbia
- [2] Andy Liaw and Matthew Wiener (2002), “Classification and Regression by RandomForest”, Vol. 2/3, December, ResearchGate Publications
- [3] BOX, G.E.P. and Jenkins.G.M (1970) “Time series analysis: Forecasting and control”, San Francisco: Holden-Day
- [4] Bryan. A Weber, ARNP, PhD, Hossein Yarandi, PhD, Meredith A. Rowe, RN, PhD, Justus P. Weber (2005), “A comparison study: paper-based versus web-based data collection and management”, Applied Nursing Research 18, 182– 185
- [5] Chenghao Liu<sup>1,2</sup>, Steven C. H. Hoi<sup>2</sup>, Peilin Zhao<sup>3</sup>, Jianling Sun, “Online ARIMA Algorithms for Time Series Prediction”, Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)
- [6] Dharmo.E, Puka.LL (2010), “Using the R-package to forecast time series: ARIMA models and Application”, December, ResearchGate Publications
- [7] Hans-Arno Jacobsen, Kaiwen Zhang, Yingqi Yue (2013), “Smart Phone Application for Connected Vehicles and Smart Transportation”, Middleware 2013 Posters and Demos Track, December 9-13, Beijing, China
- [8] Mallia M. E and Simpson. K (2014), “Wireless global positioning system fleet tracking system” The university at Albany. Report No. C-11-12/ 14-27.

- [9] Nouf Mohammad Al Shammery and Abdul Khader Jilani Saudagar (2015), “Smart Transportation Application using Global Positioning System”, (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 6, No. 6.
- [10] Motahari. S, Zang. H, Bali. S, and Reuther. P (2012), “Mobile applications tracking wireless user location”, In Proceedings of IEEE Global Communications Conference (GLOBECOM), 3–7 December, Anaheim, CA, pp. 2006–2011
- [11] Geolocation API Specification 2nd Edition, W3C Recommendation 8 November 2016, <https://www.w3.org/TR/geolocation-API/>
- [12] <http://cran.r-project.org/web/views/TimeSeries.html>
- [13] <http://ucanalytics.com/blogs/step-by-step-graphic-guide-to-forecasting-through-arima-modeling-in-r-manufacturing-case-study-example/>