Online Bus Booking and Tracking System for

SLTB Central Bus Station - *Colombo*

B.W.G.N Ranathunge

E111041050

Faculty of Information Technology

University of Moratuwa

2014

Online Bus Booking and Tracking System for

SLTB Central Bus Station - *Colombo*

B.W.G.N Ranathunge
E111041050

Dissertation submitted to the Faculty of Information Technology, University of Moratuwa, Sri Lanka in partial fulfillment of the requirements of the Degree of Bachelor of Information Technology (External) in Information Technology.

2014

Declaration

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

Name of Student

Signature of Student

Date:

Supervised by

Name of Supervisor

Signature of Supervisor

Date:

Acknowledgement

I take this opportunity to express my profound gratitude to University of Moratuwa for exemplary guidance, monitoring and constant encouragement throughout the Degree of Bachelor of Information Technology (External). And also I would like to my sincere gratitude to the Professor in-charge of the Final Year Projects, Prof. Asoka S. Karunananda for the sound knowledge given to me throughout the year regarding presentations and documentation. And also I would like to offer my sincere heartfelt gratitude to Dr. Lochandaka Ranathunga and Mr. Tharindu Gamage who are my project examiners for encouraging me.

I would like to offer my sincere heartfelt gratitude to the manager of SLTB Central Bus Station - *Colombo* and his core workers for their kind cooperation and assistance towards the success of the project from the beginning to the end.

I would like to present my heartfelt thanks and deep regards to Mr. Aruna Dissanayake who is my project supervisor for the constant support rendered to me at all stages of the project, right from the very beginning, till the submission of the final document. And also I wish to acknowledge the support and help given to my dear lecturer Ms. Shashini Pathiraja. I would like to pass my special thanks to my dear friend Mr. Dinesh Priyashantha for the support and assistance given to my project.

Very special thanks should be made to my dearest parents who have made my life successful in every way, for guiding me, for providing me with all the comfort and for providing me with the best education.

Abstract

It is very important to maintain efficient software to handle bus booking services in every place. This application provides a way to record the information and to access them in a simple way. The report presents the implementing details about web based online bus booking service system for SLTB in Colombo. In many centers, they do their whole process manually. For example recording information, process, maintain their information, etc. SLTB bus booking center in Colombo is one of them. Hence passengers have to come to the main which is located center at Colombo, even they are from remote and faraway places. This will in a total waste of time and money for the valuable passengers.

To overcome those drawbacks our aim is to design self-sufficient software, a web based online bus booking tracking system. The application is extendable, easily maintainable, cost effective, robust, user friendly and error free. The system is implemented using a web base technology. The system is able to track the location also. It is implemented using Android system. The system access is given according to their level of control to users such as administrator, system staffs, employees including conductor and passengers etc. The system facilitates you to do number of processes such as validate user login, analysis the available seats, calculate total ticket price, generate the bus ticket, analysis passenger data, generate admin reports, etc. The user will get a confirmation message when they complete there booking and payment.

The system presents additional features including generating reports as well. These reports included the analysis and design of the system and also it emphasizes the importance of having this kind of a system. This system could be implemented easily among the people.

Contents

List of Figures

List of Table

Chapter 1 – Introduction

1.1 Introduction

In Sri Lanka there are few bus booking centers. . Up to date almost the whole process of the centre is carried out manually. Therefore it takes much time to fill information and produce the ticket. Hence there are queues. And also it is not possible to do a bus booking at any time of the day (24 hours). Because it is open only at working hours. SLTB bus booking center is situated in Colombo. Therefore it is difficult to book your tickets without going for the center. It is a waste of both time and money for coming to the center for reserve your tickets. But without reserving the tickets it may also cause waste of time and lack of comfort. In such cases people require a possible solution. Having a complete manual system and less information technology they couldn't settle an appropriate method for above problems.

Web based online bus booking system is the solution for above problems. To the people who suffer to go to places to reserve their tickets, this system may be a great solution. And after reserving their ticket they do not need to wait at the bus stand until the bus comes, they could track the location of the bus and be at time. So it would be a quiet nice journey for all the passengers. Therefore this system is going to develop not only for online bus booking service and also to track the location. PHP [1], HTML [2], CSS [3], jQuery [4], Ajax [5], MySQL [6], android [7] and Google Cloud Platform [11] (for web service) would be used to develop this system.

1.2 Background and motivation

There are many aspects which forced me to select this problem. In Sri Lanka, nowadays travelling in a bus has been a difficult task for the passengers. Personally, I have faced to all the problems which I have mentioned above. And there are many changes in modern world. In this generation, IT has been using its power through all the areas. So when we look into travelling, in Sri Lanka we have to improve our technology resources. Technology and system design makes work easy and effective; therefore web based online bus booking and tracking system saves time. Time is the

most precious thing for the people nowadays. So the enthusiasm motivated me to do this project.

1.3 Aims & Objectives

Aim

The aim of this project is to achieve a standard web based online bus booking and tracking system which will be a great solution with many benefits to bus booking centers and passengers. This system will be implemented with the use of technologies like HTML, CSS, Ajax, jQuery, MySQL and android.

Objectives

- Researching and reviewing passengers' needs (easy, anywhere and anytime booking etc...).
- Researching and reviewing the manual bus booking system (SLTB).
- Generating the system as passengers could access it with least requirements to satisfy them. (Mobile phone or any devise to access internet is enough).
- Developing the system with a user friendly interface and easy procedures to attract passengers.
- Using easy payment methods (Ez cash [10], visa etc).
- Implementing solutions through system and making it more effective such as developing system to book bus ticket whenever or wherever the passengers are.
- Improving the growth of SLTB by using modern technology (web based and android) in luxury bus service. Therefore it would make changes in SLTB service.
- Study of using web technologies (PHP, HTML, CSS, Ajax, jQuery, MySQL) and android [7] Google Cloud Platform [11] (for web service) that develop the system.
- Manual booking system changing into automated web based system.

- Reinforce a variety of skills including concentration, problem solving effort, creativity to fulfill the vision.
- Evaluation of the web based system.
- Preparation of final documentation.

1.4 Web based Online Bus Booking and Tracking System for bus booking centers

In my project, the proposed solution would be the Web based online bus booking and tracking system for bus booking centers in order to eliminate the limitations of the existing manual system. The proposed solution, Web based online bus booking and tracking system designed for bus booking centers, to cover a wide range of bus transportation administration and management processes. It is an integrated end-to-end system that provides relevant information across the Sri Lanka Transportation Board to support effective transportation making for passengers, bus booking centers, administrations and profit increment, in a seamless flow. Web based online bus booking and tracking system provides the benefits of streamlining of operations, enhanced administration and control, improved response time, easily accessible for passengers, cost control and improved profitability. Mainly, the technologies such as PHP, HTML, CSS, Ajax, jQuery, MySQL, android and Google Cloud Platform [11] (for web service) will be used in developing the solution.

The proposed solution will develop using MVC technology [8]. It will use Ajax, HTML cascading style sheets. The system will use mysql database as the database management system. The system is platform independent. When implementing the solution, I will split my system into few modules such as Web client Module, Mobile client Module, Web server Module and its sub module as Web services Module and Database Module.

There are mainly four categories of users. The users are such as Administrator, System Operator, Conductor, and Booker. Administrator, System Operator has permit access the database. But there are two kinds of employees in System Operator. We will see more description about those users in further chapters. Conductors will View Passenger details, send message to passenger. And Booker have to enter there details and select the appropriate seats and confirm there tickets. As the output they would get a confirmation message.

Web based online bus booking and tracking system endorses to finalize their profit and other relevant reports such as number of passengers monthly. Another facility that system provides is they could reserve there tickets 24 hours of the day. Passenger can book the ticket at anytime without going to the bus booking center. The system administrator will handle the whole system access and maintenance. Information about the employees and viewing report is managed by the system administrator.

1.5 Remaining of the report

Rest of this final report consists of several chapters including current issues in manual bus booking system, technology adapted for the solution, the approach for the solution analysis and design for the system, implementation of the solution, and finally there will be a discussion on the evaluation of the system. The Chapter 2 describes reports on a critical literature review and defines the problem addressed in the project. Chapter 3 is a survey on technology used for similar problems with an emphasis on technology to be used in this project. Chapter 4 is about the approach which describes the users, inputs, outputs, process, technology and overall features of the proposed solution and Chapter 5 demonstrates the overall design of the system including the logical diagrams such as, top level design, ER Diagram, Data Flow Diagram, Use case, Class and Swim lanes etc. During the Chapter 6 of this report it will present the software, hardware, code segments, screen shots that are being used for implementing the solution. Finally, Chapter 7 provides a discussion on the further work.

Chapter 2 – Current issues in Bus Booking System

2.1 Introduction

This chapter gives a full description about background information of the project. Based on a literature survey, here it is state about other's approaches to solve similar problems, bus booking system. The chapter illustrates drawbacks of other alternative solutions also.

2.2 Alternative Solutions

2.2.1 Booking through Phone Line system

Some private centers use this kind of solutions. For example implement a solution through communication system to reserve seats. In this kind of a system all users have to have to make a phone call to book their tickets. So in this kind of way they do other works manually. They do have more than one connection to contact but it is very difficult to get the line, because phone connection is not like online connection. Multi users cannot access the system simultaneously. And there is no proper interconnection among the employees. The employees do not have a proper system therefore information sharing is not consisted. And due to manual work, the reliability is in a poor level. After booking the ticket if they have to stand until buses come, then they do not have a satisfying connection with system at anytime of the day to get the information about the bus. It does not have data security and integrity.

All the customers should use the phone connection therefore nowadays it's an easy way to access. So it is easily accessible but less in security. System is in a manual way thus data transferring is not a problem. Security problems during data transferring do not apply in this approach. As well as it has low reliability, data access does not depend on the network solutions. Therefore the database reliability depends only on the current manual system.

Although this solution has very few advantages it is not feasible enough for the client's requirements due to the following reasons:-

From the system, I expect to provide facility to reserve the tickets in your own and you can book your tickets whenever you want, and customers can follow the location of the bus through tracking system. So it is incompatible with the Telephone-system. And it is high reliable cause to web based, online and tracking system. Web base online bus booking and tracking system means all processes in center which are done by manually should do via the system. Among various users, some of the information should be shared between them. Hence standalone system is not applicable for this requirement.

2.2.2 Existing Current Manual Process in SLTB

Bus booking service is to reserve the books earlier to confirm our seats at the time of journey. If there are many passengers, one related person could reserve all there tickets earlier. So bus booking is very useful to make sure of the journey and feel comfort, and also time saving. Sri Lanka Transport Board Company is one of the leading Transport service in Sri Lanka. It provides various traveling methods. SLTB is currently having a manual system to store the details of buses, employee, and daily income. All the processes such as employees' attendance, customer details entering, daily transactions, and reports are done manually. Therefore it takes much time, more paper work and employees.

When it comes to bus booking, SLTB has a center in every district all over the country. In Colombo, the passengers should come to the main center in Pettah to reserve the tickets. Colombo is the capital of our country, and there will be many booking throughout the day. So it is not only a waste of time and money for passengers, it is also a disadvantage to SLTB. And when they have to get into the bus they should come to the main stand. Because they can't pickup everyone at there bus stands due to lack of information. Passengers can't get into the bus at there nearest bus halt because they can't stand there until bus comes. When they have to get information about time of the bus arrival, information about empty seats and bus route, it is not more efficient. They should contact with the center and get information; sometimes the passengers are dissatisfied.

So all these kind of process are done manually. Therefore it is both waste of time and money for passengers. And also manual bus booking is less in facilities in

modern world. Hence manual processing system is not applicable for this requirement.

2.3 The Benefits of Web Based Online Bus Booking and Tracking System

The web based system keeps all the details computerized such as bus details, passengers' details and process them electronically, to implement the manual system as web based online bus booking system. There is no need to keep all the records in paper work. It handles whole things in a proper way so it is capable of easy accessing. And it is easy to keep backups.

The information about the bus, employees, bus route, and passengers will be managed by administrator. The user can log into online system and book there tickets. Payment, transaction activities also will be managed by administrator. Users will get an email or message when you had completed your booking and payment. In web based system it provides facility such as passengers can get details from the web based online system, therefore no need to contact the center. The system creates the payment bills of the passenger. Web based system provides facility such as passengers can get details from the web based online system, therefore no need to contact the center. Another capability that can provide by the system is to communicate between department and bus. The system creates the connections between the centers so if they want administrators can share there common details. Web based system endorse to finalize their profit and other relevant reports such as number of passengers monthly, etc.

Another facility that the system provides is to book tour buses using the system. Passengers can book the bus without going to the center.

The web based systems have the following benefits.

- Reduced costs-Web based applications can dramatically lower costs due to reduced support and maintenance, lower requirements on the end user system and simplified architecture.
- Its efficiency in processing details and finds the destination, in booking the ticket and pay through online. Hence it reduces time waste.

- Profit increment – Due to Web based online bus booking and tracking system users would have a certain facilities in travelling, so there will be unimaginable increase in profit.
- No single failure location- Data is stored in multiple locations so it would be more disaster resistive to administer.
- Highly deployable-Due to the manageability and cross platform support deploying web applications to the end user is far easier. They are also ideal where bandwidth is limited and the system and data is remote to the user.
- Secured activities-Typically in larger more complex systems data is stored and moved around separate systems and data sources. In Web based online bus booking system these details and payment will be secured and guaranteed with an email /message which is sent by the system to your specific email address/telephone no.

Hence web based system is most applicable for this requirement.

2.4 Comparison between Manual vs. Web based online bus booking system

Criteria	Manual System	Computerized System
1. Storage	Stored in the manual file base system	Stored electronically in a database
2. Time consuming (Efficiency)	Time consuming is heavy due to the manual work	Less time consuming due to automated tasks
3.Querying information	Hard to manipulate queries based on different criteria	Easy query manipulation with MYSQL.
4. Report Generation	Hard to generate reports	Easy and extensive report generation functionality
5. Security	Less security to data since anyone can access the files.	High security due to role based access control and authentication mechanisms
6.Disaster Recovery	Files can be stolen, damaged or destroyed.	Data can be protected with backup mechanisms.

7. Portability	Data ,reports are not easily portable	Data, reports are portable due to web based online system

Table 2.1: Comparison between Manual vs. Web based online bus booking system

2.5 Summary

This chapter gave a full description about background information about the project.

Further, based on a literature survey other's approaches to solve similar problems have been discussed and compared those approaches with the approach of this project. Next chapter will discuss about the Technology adapted for this project.

Chapter 3 – Web Based Bus Booking System beyond Manual Booking System

3.1 Introduction

This chapter will discuss about the technology that can be adapted to solve the problem. The reasons for selecting such technologies will also be discussed. This is a project where we adapt technology to solve problems in manual system and design web based online and tracking system. Hence, selecting appropriate technologies would be important in order to facilitate the intended requirements successfully.

3.2 Web Based Bus Booking System exceeding manual processes

We have many problems in manual system therefore web based system is implementing as a solution with the use of web technologies. For web based online and tracking system we have used the technologies such as PHP [1], HTML, CSS, Ajax, jQuery [4] and MySQL, MVC, android [7] and Google Cloud Platform [11] (for web service) in developing the solution.

When the clients send a request from web page in his machine, it would be carried to the server, and it is done by HTTPS protocol. HTTPS protocol is being used because of its high security. When we do transaction through online, secure is very important.

Data manipulation would be an extremely well handled part of any system. As such, inserting, updating and deleting data to and from the database should be handled carefully since data is the most valuable part of the system. Keeping this in mind MySql has been selected for this project due to following reasons.

- The MySQL [6] is one of the most used open source databases best and the most-used database in the world for online applications.
- Continuously improved while remaining fast, secure and reliable
- Free from bugs, and it has GUI interface therefore easy to access.

- Elegantly support with the coding language for this project (PHP)
- PHP is a popular general-purpose scripting language that is especially suited for web development.

Java Script - JavaScript is generally used for client-side scripting. JavaScript works best for visual animation (such as changing an image when a user moves the mouse pointer over it) or for validating form fields. *Can minimums human error*.

Model–view–controller (MVC) [8] is a software pattern for implementing this system.

JQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML.

CSS provide the ability to change the appearance of text (such as fonts, colors, spacing) on Web pages.

Ajax is techniques used on the client-side to create asynchronous web applications. With Ajax, web applications can send data to, and retrieve data from, a server asynchronously (in the background) without interfering with the display and behavior of the existing page.

In manual system it takes more paper work and more employees to do all the process. And it takes much time to get the work done. Cost more than web based system. With the use of such technologies web based system helps to do bus booking more efficient, attractive interface, easy procedures, quick access, less paper work and high secured. These are the reasons why I am using those technologies to develop the system.

Apache Server

Purpose:- The order to execute the client site of Online Bus Booking System, the web server specified above is required as the provider of the client software at the server site.

PHP

Purpose: In the order to build web pages which work with MySQL database and Apache server.

Definition of the Interface: PHP is a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into HTML.

Macromedia Dreamweaver

Purpose: The web development tool specified above is required for designing and coding the project.

Definition of the Interface: Macromedia Dreamweaver is the industry-leading web development tool, enabling users to efficiently design, develop and maintain standards-based websites and applications.

MySQL

Purpose: Required as database server.

Definition of the Interface: MySQL is the world's most popular open source database software. With superior speed, reliability, and ease of use, MySQL has become the preferred choice of corporate IT Managers because it eliminates the major problems associated with downtime, maintenance, administration and support.

3.3 Summary

This chapter discussed about the technology that can be adapted for this project and how/why those technologies are appropriate to solve the problem. PHP, jQuery, Ajax language has been selected for coding the application; MYSQL Server has been selected as the Database.

Chapter 4 - Using Web Based Online Bus Booking and Tracking System for Built SLTB Extraordinary

4.1 Introduction

This section will talk about the technology that is used in this project, users that are involve, what are the inputs , what are the out puts and process which are going to involve in this development. And this is about how we have assimilated the technology to solve the said problems in SLTB. By changing manual system to Web Based Online Bus Booking and Tracking System will increase the efficiency, accuracy and performance of the SLTB.

This system will be referred to as (**OBBTS)** Web Based **O**nline **B**us **B**ooking **a**nd **T**racking **S**ystem in the future.

4.2 Users

The people who are interacted with the system can be defined as users. In this system, four types of users can be identified based on their access roles. They are namely Administrator, System Operator, Conductor and Booker.

Administrator

- Create System User and view report.

System Operator

- Enter all bus details, user details, ticket price and etc.

Conductor

- View Passenger details, send message to passenger and etc.

Booker (Online)

- Select destination and seat , enter passenger info, pay money and etc

Booker (Manual)

- Select destination and seat , enter passenger info, pay money and etc

4.3 Inputs

Bus details, Ticket Price, Passenger details, Payment details, etc.

There are many aspects which differentiate from other bus booking system. It is such as Bus Entry Point. In other booking system they do not have facilities to get into the bus at their nearest bus halt. But (OBBTS**)** have designed to overcome all those draw backs. They do have an option (Bus Entry Point) as input to select there nearest bus halt.

4.4 Outputs

- Available Bus, Available Seat.
- Receiving confirmation SMS, Print Ticket.
- Bus Location.
- Admin Report, etc.

As outputs we do have above actions. A major advantage we can locate the bus location through an android phone or internet connection.

4.5 Process

- Validate User Login.
- Analysis available seat.
- Calculate total ticket price.
- Generate bus ticket.
- Analysis passenger data.
- Generate admin report.

4.6 Technology

For web based online and tracking system we have used the technologies such as PHP [1], HTML, CSS, Ajax, jQuery [4], MySQL, MVC, android [7] and Google Cloud Platform [11] for web service in developing the solution. All those technologies used to design the system and achieve the vision of this system.

4.7 Features

There are many features such as

- Book the ticket within 24 hours of the journey.
- Selecting bus seats – You can select your own seat as your wish.
- Have a easy payment methods.
- Have a bus entry point.
- Find the location of the bus.

4.8 Summary

In this chapter described the approach for the proposed system. This chapter also discussed the technology, users, inputs, output, process and over all features for this system. The next chapter will discuss the analysis and design of the proposed system .It is one of the most important chapters in this report.

Chapter 5 – Analysis and Design of OBBTS

5.1 Introduction

Since, the project should gather, analyze and review the requirements to ensure that it meets the stated and implied needs of the SLTB, it is vital to have documented designs for understanding the system for development and further maintenance. Requirements shall form the basis for planning and development of subsequent work products. Every find out derived in this phase will be documented for the use of successful development of OBBTS and its maintenance.

5.2 Top Level Architecture of Web Based online bus booking and tracking System

In Top Level Architecture of OBBTS **(**shown in Figure 5.1**)** have mainly four modules. As figurer 5.1 displays they are Web client Module, Database Module, Mobile client Module, Web server Module and its sub module as Web services Module.

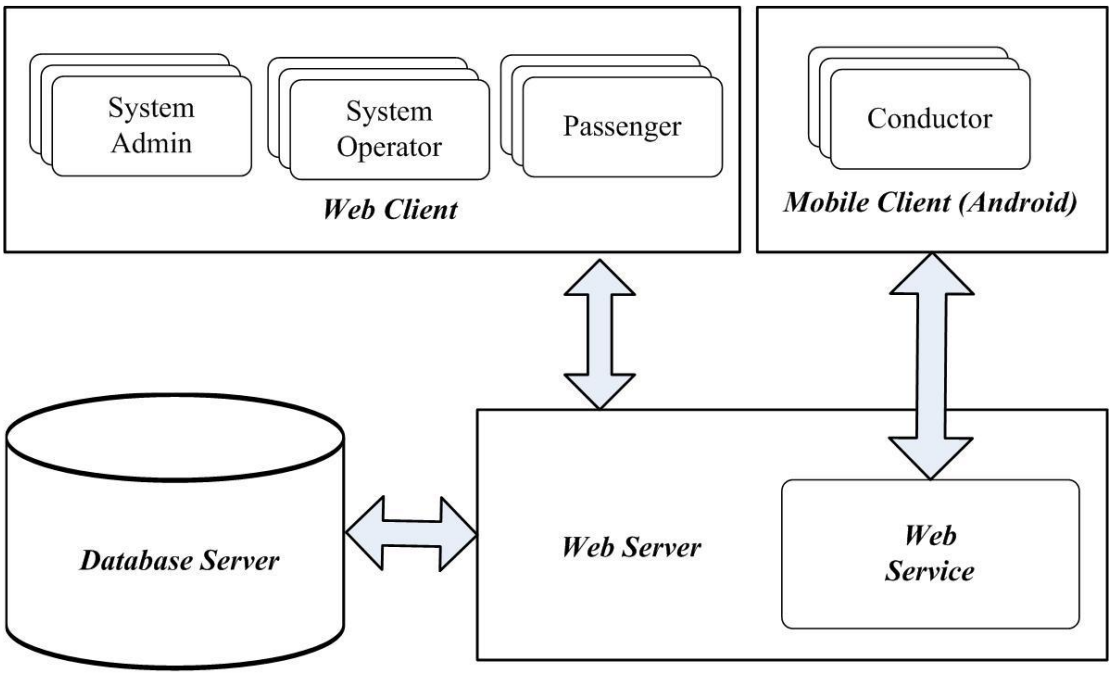


Figure 5.1 - Top Level Architecture

5.3 Use Case Diagram

Use case diagram shows the interaction between the user roles (actors) with the system. In this system actor such as Admin, System Operator, Booker, and Conductor interact with the system. The use case diagram for online bus booking system is shown in (Figure 5.2).

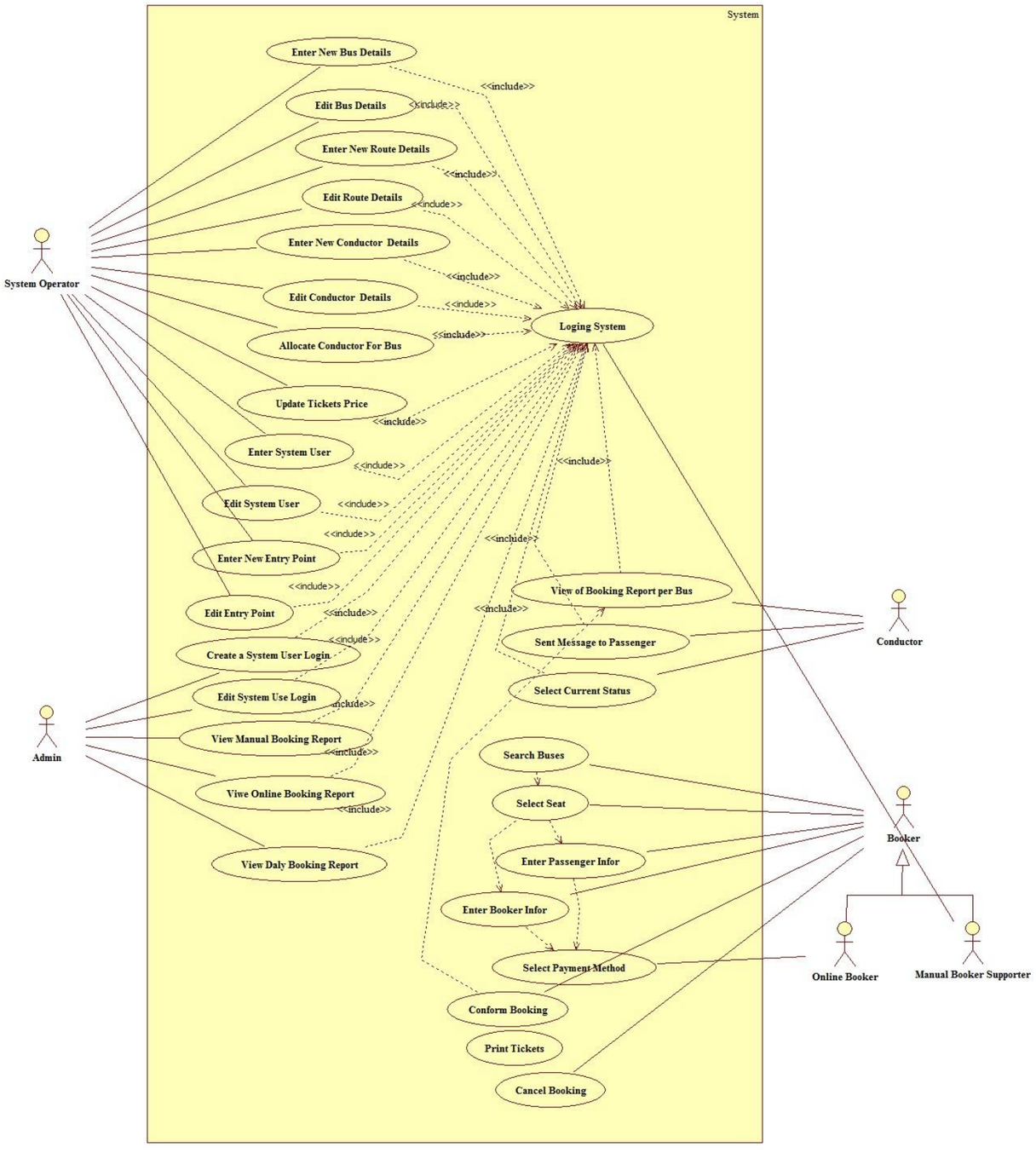


Figure 5.2- Use case diagram for Online Bus Booking and Tracking System

5.4 Class Diagram

Class diagrams show the classes of the system, their interrelationships (including inheritance, aggregation, and association), and the operations and attributes of the classes. Class diagrams are used for a wide variety of purposes, including both conceptual/domain modeling and detailed design modeling. The class diagram for online bus booking system is shown in (Figure 5.3).

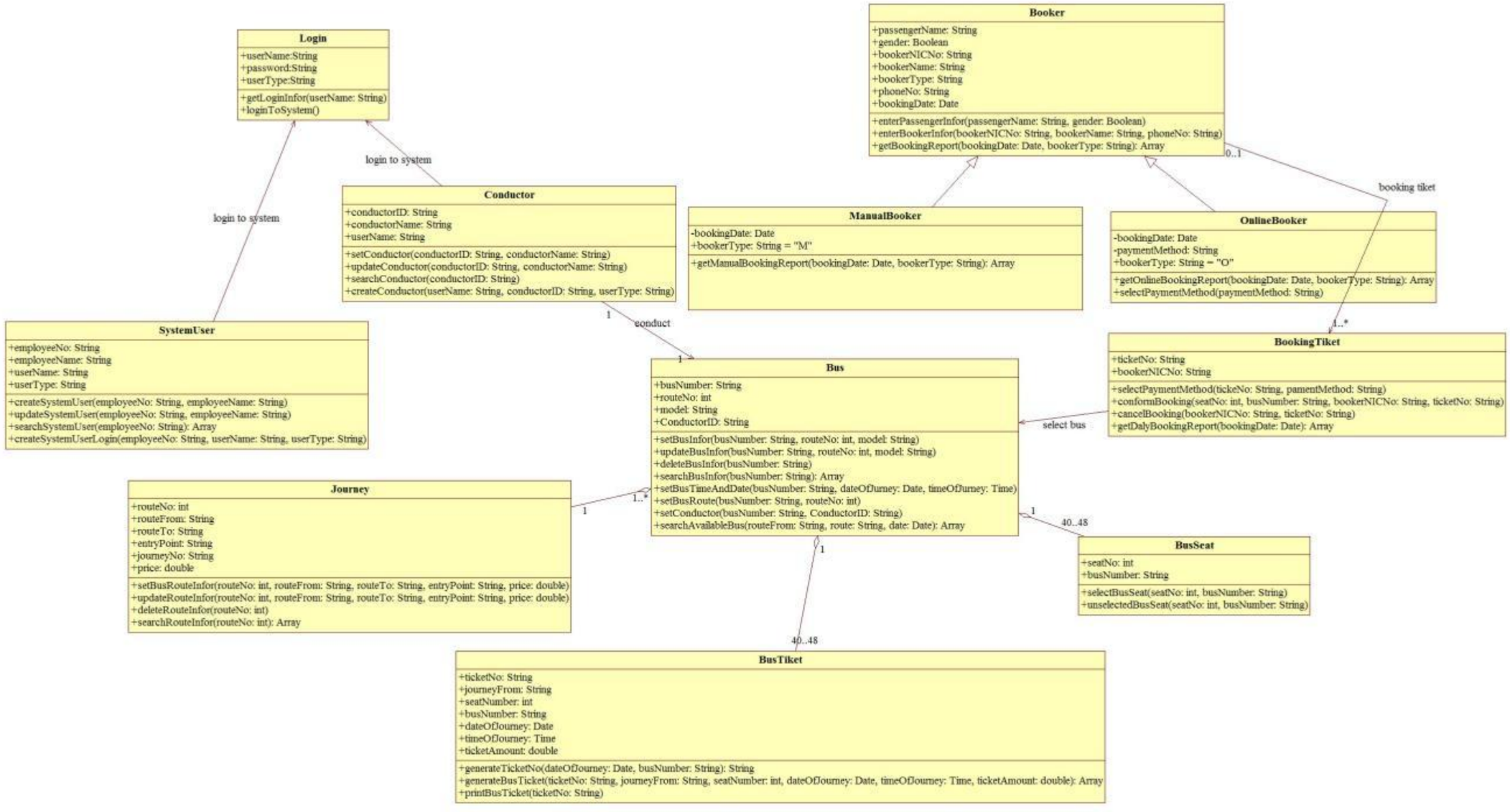


Figure 5.3 - Class diagram for Online Bus Booking Tracking System

5.5 Module Decomposition

This system comprises of following modules.

- Database Module
- System Operation Module
- Administration Module
- Booking Module
- Tracking Module

Firstly, the Administration Module is responsible for giving permission to users to access the system according to their responsibilities. Administrator controls the access level. There are two kinds of system operators. First type of operators input bus details, bus route, ticket price, etc. And other type operators do booking tickets for people who comes to center to book there ticket. They book the ticket through **(**OBBTS**).** So the administrator manages user levels of operators and view daily reports. System Operation Module is used to intake the inputs such as, bus details, bus route, ticket price, creating new users, dropping users from the system, update information to system and permits to access the system to the users to select their destinations, bus entry point and reserve the seats. Thirdly, the Booking Module is the most important module of this system. Through booking module booker can select destination and bus seats, enter details, pay payments and print tickets, etc. And Database Module will facilitate the functionalities stores and views data of all the users and other details, etc. Tracking Module is design to give location details for the passengers through a map in the system. They could observe it through internet or android phone.

Module 1-Database module

In this module it handles whole database which are related to this **(**OBBTS**)**. The information which has to store, are concerned in this module. This interacts with other module, Booking Module. To create this module supposed to use MySql. This module will be used by users such as administrator, system operators.

Because all processes which are done by this system may have taken/stored their information in Database module such as information of buses, destinations, employees, customers' details and payment details

ER diagram represents the system database architecture and its relationship. Good ER diagram will make the system run smoothly and allow performing faster. It speeds up data retrieval and saving. This make the system run faster. The entity relationship diagram for this module is shown in (Figure 5.4).

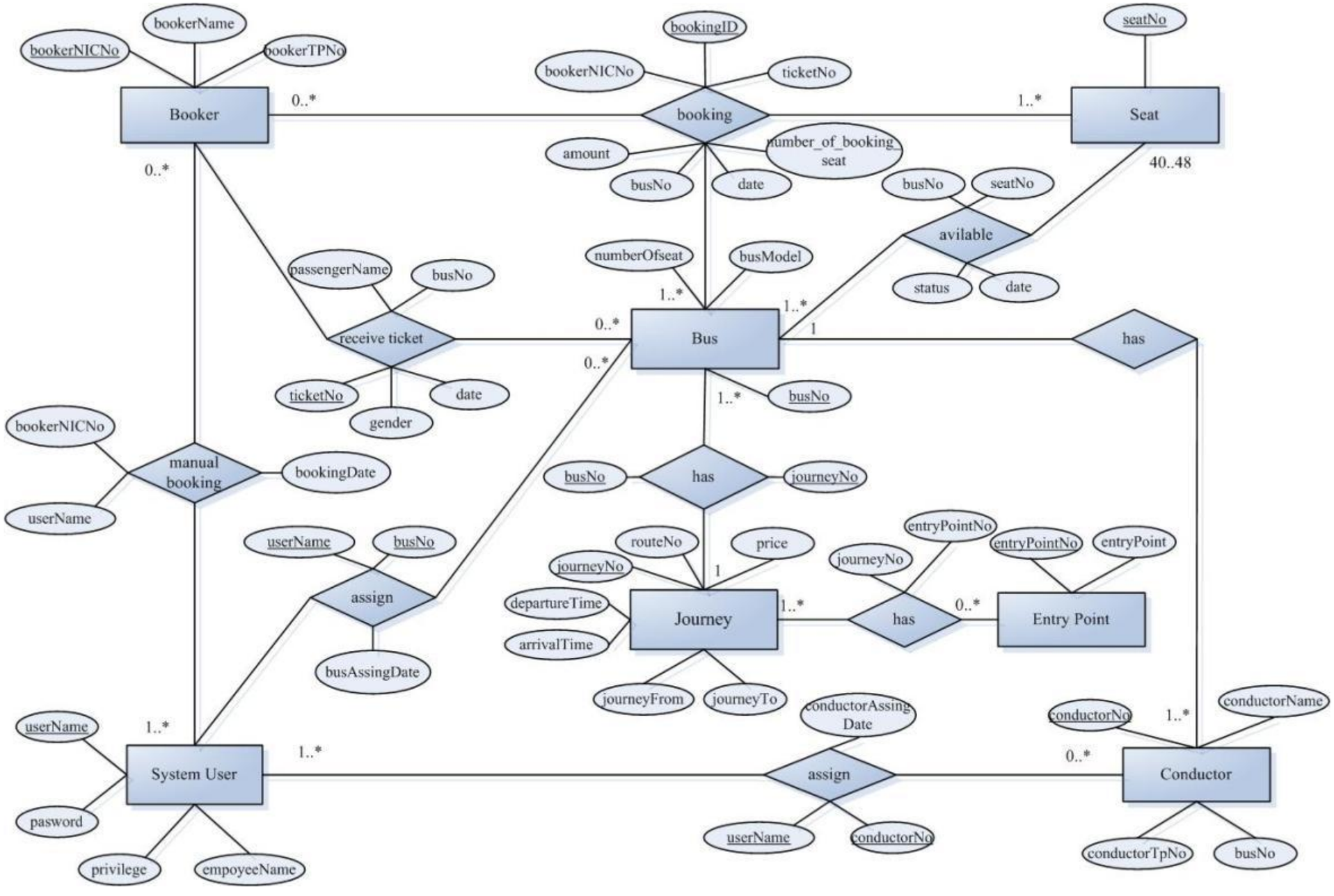


Figure 5.4- Entity Relationship diagram for database module

Module 2- System Operation Module

In this module it updates daily bus details such as bus routes, bus class, time schedules, ticket price, bus entry points and other important information. And it has access to create new users, drop users, etc.

The Swim lanes for System Operation are cited in “Appendix A”.

Module 3 - Administration Module

Administration Module is responsible for giving permission to users to access the system according to their responsibilities. Administrator controls the access level. And administrator view daily reports and manage the system. Administration module interacts with the system operator module by giving control of its access level. The Swim lanes for Administration Operation are cited in “Appendix B”.

Module 4 – Booking Module

Booking Module is the most important module of this system. Through booking module booker can select destination, bus seats, bus entry point and enter details, pay payments print tickets, etc. All those details are stored in database through database module. Therefore this module is also interacts with database module through https protocol.

The Swim lanes for Booking Seat are cited in “Appendix C”.

Module 5 – Tracking Module

Tracking Module is designed to give location details for the passengers through a map in the system. They could observe it through internet or android phone. Tracking module interacts with mobile client (android) and web services. Android technology cannot interact with web server. Therefore it interacts through web services. This is an important module in this system which differentiates from other alternative solutions for bus booking problems in nowadays. Uniquely it differentiates due to tracking the locations.

The Swim lanes for Tracking Bus are cited in “Appendix D”.

5.6 Summary

This chapter included the overall analysis and design part of the project. As such, it included several UML diagrams such as Use case diagram, Swim lanes and class diagram etc. The top level architecture shows the included modules of this system. Next Chapter will discuss about the implementation of the project thereby stepping to the developing stage of this project.

Chapter 6 - Implementation of Web Based online bus booking and tracking System

6.1 Introduction

In this chapter it is providing execution details of each module that is stated in the design diagram. In describing the implementation, it states about, software, hardware, and some pseudo code, code segments in modules in the design. Also the chapter describes why those software and hardware are choosing for the solution.

6.2 Resource Requirements

		Hardware	Software	Other
Client	Passenger	Computer, Mobile Phone	Web Browser	Internet access
	Conductor	*Android Device (*Minimum: 832 MHz, 290 MB RAM, 20 MB SD Card Space*)*	*Android OS*	
	Admin (SLTB)	Computer	Web Browser, pdf	
	System Operator (SLTB)	Computer	Web Browser,	
Server		3.06 GHz or faster processor, of 2 RAM space, 5 GB disk space	Apache Sarver, My Sql Database	
Developer		Computer(3.06 GHz or faster processor, of 2 RAM space) *Android Device (*Minimum: 832 MHz, 290 MB RAM*)*	Windows OS, My Sql, PHP, Apache Sarver, Android SDK, Java	

Table 6.1: Resource Requirements

6.3 Implementation of Web Based online bus booking and tracking System

Web based administration system consists of five modules as described in previous chapter such as database module, web client module, web server module and web mobile client module. These module are implementing use various technologies.

Module 1-Database module

Within this module it concerns about whole database which are related to Web Based online bus booking and tracking System. The information which has to store, are concerned in this module. The database is being implemented with MySQL Graphical User Interface (use "phpMyAdmin" web tool) create all tables.

The Some screen shot of the Create Database are cited in "Appendix E".

Pseudo code for Module

Module 2- System Operation Module

The implementation of this module is described through following details.

The pseudo code for System Operation is show below.

Enter and Edit Bus Data

```
Begin
Display Screen (Main Bus Page)
//Add Bus Data
IF click "Add" Button
        Display Screen (Enter Bus Form)
        //Input
        Input busNumber and Bus data from Text Fields
        IF click "Save" Button
                IF required fields are empty
```

```
                    Display Message "Field is Empty"
                ELSE
                        Read all Bus Data
                        //Open DB
                        Open Database
                        Insert all data to Bus Master Table
                        IF insert is Success
                            Display Message “Successful Save”
                        ELSE
                            Display Message “Not Save”
                        END IF
                        //Close DB
                        Close Database
                END IF
        END IF
//Update Bus Data
IF click “Update” Button
        //Input
        Enter busNumber
        IF click “Edit” Button
                IF Field is Empty
                    Display Message "Fields is Empty"
                ELSE
                    //Get the input Data
                    Read busNumber
                    //Open DB
                    Open Database
                    Get Bus Details to Bus Master Table
                    //Process
                    While (NOT End of Record)
                        Read Data
                        Fill Fields
                    END While
```

```
                    //Output
                    Output Display Bus Details on Fields
                    Input New Data
                    IF click “Update” Button
                        IF required fields are empty
                            Display Message "Field is Empty"
                        ELSE
                            Read all Bus Data
                            Update all data to Bus Master Table
                            IF Update is Success
                                Display Message “Successful Update”
                            ELSE
                                Display Message “Not Update”
                            END IF
                        END IF
                    END IF
                    //Close DB
                    Close Database
                END IF
            END IF
END IF
End
```

Enter and Edit Conductor Details

```
Begin
Display Screen (Main Page)
//Add Conductor Data
IF click “Add” Button
        Display Screen (Enter conductor Form)
        //Input
        Input conductorNo, conductorName, mobile, busNo data
        IF click “Save” Button
                IF required fields are empty
                        Display Message "Field is Empty"
                ELSE
                        Read all conductor Data
                        //Open DB
                        Open Database
                        Insert all data to conductor Master Table
                        IF insert is Success
                                Display Message “Successful Save”
                        ELSE
                                Display Message “Not Save”
                        END IF
                        //Close DB
                        Close Database
                END IF
        END IF
//Update conductor Data
IF click “Update” Button
        //Input
        Enter conductorNumber
        IF click “Edit” Button
                IF Field is Empty
                        Display Message "Fields is Empty"
```

```
            ELSE
                //Get the input Data
                Read conductorNumber
                //Open DB
                Open Database
                Get coductor Details to conductor Master Table
                //Process
                While (NOT End of Record)
                    Read Data
                    Fill Fields
                END While
                //Output
                Output Display conductor Details on Fields
                Input New Data
                IF click “Update” Button
                    IF required fields are empty
                        Display Message "Field is Empty"
                    ELSE
                        Read all Bus Data
                        Update all data to conductor Master Table
                        IF Update is Success
                            Display Message “Successful Update”
                        ELSE
                            Display Message “Not Update”
                        END IF
                    END IF
                END IF
                //Close DB
                Close Database
            END IF
        END IF
END IF
End
```

Enter and Edit Entry Point Details

```
Begin
Display Screen (Main Page)
//Add entry point Data
IF click “Add” Button
        Display Screen (Enter conductor Form)
        //Input
        Input entryPointNo, entryPointName
        IF click “Save” Button
                IF required fields are empty
                        Display Message "Field is Empty"
                ELSE
                        Read all entry Point Data
                        //Open DB
                        Open Database
                        Insert all data to entryPoint Master Table
                        IF insert is Success
                                Display Message “Successful Save”
                        ELSE
                                Display Message “Not Save”
                        END IF
                        //Close DB
                        Close Database
                END IF
        END IF
//Update entry Point Data
IF click “Update” Button
        //Input
        Enter entryPointNumber
        IF click “Edit” Button
                IF Field is Empty
                        Display Message "Fields is Empty"
```

```
            ELSE
                //Get the input Data
                Read entryPointNumber
                //Open DB
                Open Database
                Get entry Point Details to entryPoint Master Table
                //Process
                While (NOT End of Record)
                    Read Data
                    Fill Fields
                END While
                //Output
                Output Display conductor Details on Fields
                Input New Data
                IF click “Update” Button
                    IF required fields are empty
                        Display Message "Field is Empty"
                    ELSE
                        Read all Bus Data
                        Update all data to entryPoint Master Table
                        IF Update is Success
                            Display Message “Successful Update”
                        ELSE
                            Display Message “Not Update”
                        END IF
                    END IF
                END IF
                //Close DB
                Close Database
            END IF
        END IF
END IF
End
```

Enter and Edit New Route Details

```
Begin
Display Screen (Main Page)
//Add route Data
IF click “Add” Button
        Display Screen (Enter route Form)
        //Input
        Input routeNo, routeFrom, routeTo, pirce
        IF click “Save” Button
                IF required fields are empty
                        Display Message "Field is Empty"
                ELSE
                        Read all route Data
                        //Open DB
                        Open Database
                        Insert all data to route Master Table
                        IF insert is Success
                                Display Message “Successful Save”
                        ELSE
                                Display Message “Not Save”
                        END IF
                        //Close DB
                        Close Database
                END IF
        END IF
//Update route Data
IF click “Update” Button
        //Input
        Enter routeNumber
        IF click “Edit” Button
                IF Field is Empty
                        Display Message "Fields is Empty"
```

```
            ELSE
                //Get the input Data
                Read routeNumber
                //Open DB
                Open Database
                Get route Details to route Master Table
                //Process
                While (NOT End of Record)
                    Read Data
                    Fill Fields
                END While
                //Output
                Output Display route Details on Fields
                Input New Data
                 IF click “Update” Button
                    IF required fields are empty
                        Display Message "Field is Empty"
                    ELSE
                        Read all Bus Data
                        Update all data to route Master Table
                        IF Update is Success
                            Display Message “Successful Update”
                        ELSE
                            Display Message “Not Update”
                        END IF
                    END IF
                END IF
                //Close DB
                Close Database
            END IF
        END IF
END IF
End
```

Enter and Edit System User Details

```
Begin
Display Screen (Main Page)
//Add System User Data
IF click “Add” Button
        Display Screen (Enter route Form)
        //Input
        Input userName, emNo, emName, emMobileNo, passwaord, privilege
        IF click “Save” Button
                IF required fields are empty
                        Display Message "Field is Empty"
                ELSE
                        Read all route Data
                        //Open DB
                        Open Database
                        Insert all data to systemUser Master Table
                        IF insert is Success
                                Display Message “Successful Save”
                        ELSE
                                Display Message “Not Save”
                        END IF
                        //Close DB
                        Close Database
                END IF
        END IF
//Update System User Data
IF click “Update” Button
        //Input
        Enter username
        IF click “Edit” Button
                IF Field is Empty
                        Display Message "Fields is Empty"
```

```
            ELSE
                //Get the input Data
                Read userName
                //Open DB
                Open Database
                Get system user Details to systemuser Master Table
                //Process
                While (NOT End of Record)
                    Read Data
                    Fill Fields
                END While
                //Output
                Output Display system user  Details on Fields
                Input New Data
                IF click “Update” Button
                    IF required fields are empty
                        Display Message "Field is Empty"
                    ELSE
                        Read all Bus Data
                        Update all data to systemuser Master Table
                        IF Update is Success
                            Display Message “Successful Update”
                        ELSE
                            Display Message “Not Update”
                        END IF
                    END IF
                END IF
                //Close DB
                Close Database
            END IF
        END IF
END IF
End
```

Module 3 - Administration Module

Administrator controls the access level, view daily reports and manages the system. Administration module does all these implementation using following details. The pseudo code for Administration is show below.

Create System User

```
Begin
Display Screen (Main Bus Page)
//Create System User
IF click “Create System User” Tab
        Display Screen (Create System User Form)
        //Input
        Input userName, password, userType from Text Fields
        IF click “Save” Button
                IF required fields are empty
                        Display Message "Field is Empty"
                ELSE
                        Read all Bus Data
                        //Open DB
                        Open Database
                        Insert all data to Bus Master Table
                        IF insert is Success
                                Display Message “Successful Save”
                        ELSE
                                Display Message “Not Save”
                        END IF
                        //Close DB
                        Close Database
                END IF
        END IF
```

```
//Update System User
IF click “Update” Button
        //Input
        Enter username
        IF click “Edit” Button
                IF Field is Empty
                        Display Message "Fields is Empty"
                ELSE
                        //Get the input Data
                        Read userName
                        //Open DB
                        Open Database
                        Get Sytem User Details to Login Master Table
                        //Process
                        While (NOT End of Record)
                                Read Data
                                Fill Fields
                        END While
                        //Output
                        Output Display Bus Details on Fields
                        Input New password
                        IF click “Update” Button
                                IF required fields are empty
                                        Display Message "Field is Empty"
                                ELSE
                                        Read all Data
                                        Update all data to Login Master Table
                                        IF Update is Success
                                            Display Message “Successful Update”
                                        ELSE
                                            Display Message “Not Update”
                                        END IF
                                END IF
```

```
                    END IF
                    //Close DB
                    Close Database
                END IF
            END IF
END IF
End
```

Booking Report

```
Begin
        Display Screen (Report Page)
        //Input
                Enter bookingDate, bookingType
                IF click “Report” Button
                        IF Field is Empty
                                Display Message "Fields is Empty"
                        ELSE
                                //Get the input Data
                                Read bookingDate, bookingType
                                //Open DB
                                Open Database
                                Get Booking Details
                                //Process
                                Analysis Data
                                While (NOT End of Record)
                                        Read Data
                                        Fill row in table
                                END While
                                //Output
                                Output Display Booking Report
                                //Close DB
```

```
                    Close Database
                END IF
            END IF
End
```

Module 4 – Booking Module

Booking Module is the most important module of this system. Through booking module booker can select destination, bus seats, bus entry point and enter details, pay payments print tickets, etc. This module interacts with database module through https protocol.

The pseudo code for Booking is show below.

Search Available Bus

```
Begin
Display Screen (Search Bus Page)
//Input
    Enter journeyForm, journeyTo, journeyDate
    IF click “Search” Button
        IF Field is Empty
            Display Message "Fields is Empty"
        ELSE
            //Get the input Data
            Read journeyForm, journeyTo, journeyDate
            //Open DB
            Open Database
            Get Available Bus Details
            //Process
```

```
                    While (NOT End of Record)
                        Read Data
                        Fill row in table
                    END While
                    //Output
                    Output Display all Available Bus Details on the table
                    //Close DB
                    Close Database
                END IF
            END IF
End
```

Booking Bus Seat

```
Begin
Display Screen (Booking Seat Page)
Select Seat
IF Select Seat
    Holed Bus Seat
    //Output
    Selection Seat, Time for expire seat
    IF (NOT expired cookie)
        //Input
        Booking Data (passengerName, gender, booker NIC, mobileNo)
        IF Field is Empty
            Display Message "Fields is Empty"
        ELSE
            IF (NOT expired cookie)
                //Input
                Enter Payment Data
                IF (Valid Payment)
```

```
                                    Conform Booking
                                    //Open DB
                                    Open Database
                                    Update Data
                                    //Output
                                    Print Ticket
                            ELSE
                                    Reject Booking
                            END IF
                    ELSE
                            //Output
                            Display Message “Time is expired”
                    END IF
            END IF
        ELSE
            //Output
            Display Message “Time is expired”
        END IF
END IF
End
```

Module 5 – Tracking Module

In this module it generates longitudes, latitudes of the location to the tracking system. It interacts with the web services module. This is used to get details from web service modules such as customers’ details, destination, etc.

The pseudo code for Tracking is show below.

Track Bus Location

```
Begin
        Display Screen (Main App Window)
        On “GPS” Button
                IF check GPS on
                        Get Latitude & longitude
                END IF
        On “Send Latitude & longitude” Button
                IF check “GPS” AND “Latitude & longitude” Button
                        //Open DB
                        Open Database
                        Update Latitude & longitude data to Table
                END IF
End
```

Show Bus Location on Map

```
Begin
        Display Screen (Map)
        //Input
        busNumber
        IF click “Show Location” Button
                IF required fields are empty
                        Display Message "Field is Empty"
                ELSE
                        Read busNumber
                        //Open DB
                        Open Database
                        Search Latitude & longitude form Table
                        //Output
```

```
                    Show Bus location on Map
                END IF
            END IF
End
```

Screen shot and code for Module

Module 2 - System Operation Module

Login Page is shown in “Figure 6.1.1” and codes are cited in “Appendix F”.

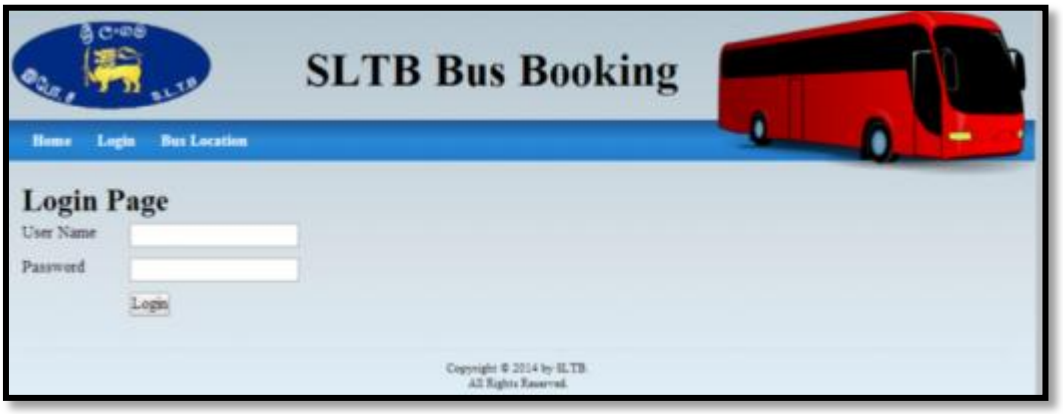


Figure 6.1.1 - Screen shot of Login Page

Display all Busses Page is shown in “Figure 6.1.2” and codes are cited in “Appendix G”.

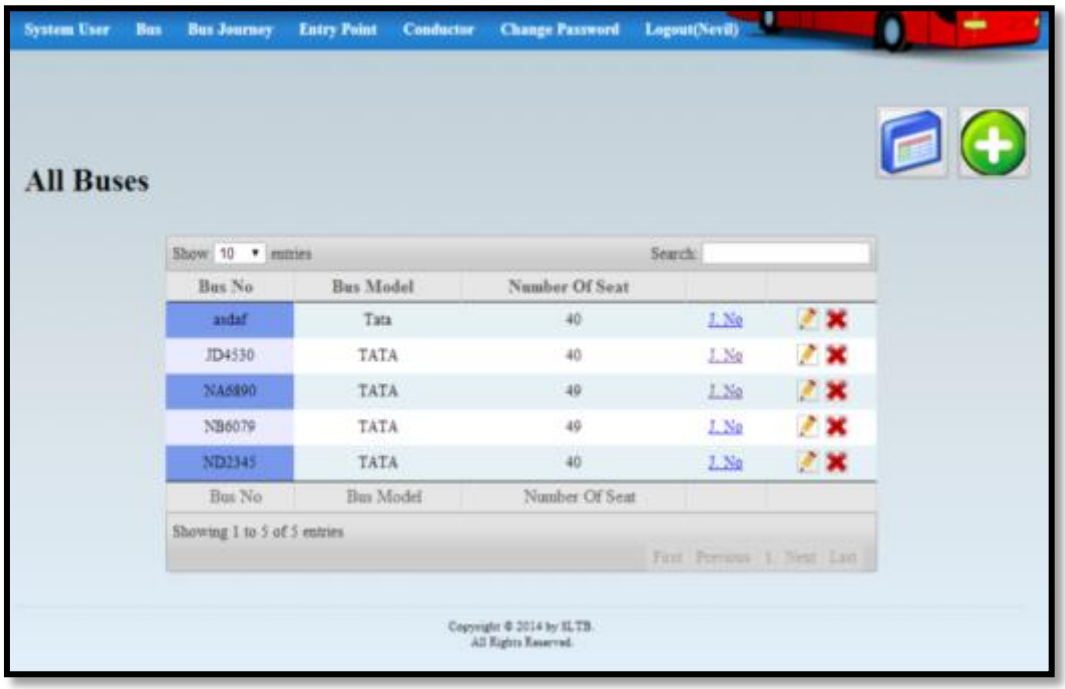


Figure 6.1.2 - Screen shot of Display all Buses Page

Enter New Bus Page is shown in “Figure 6.1.3” and codes are cited in “Appendix H”.

Figure 6.1.3 - Screen shot of Add new Bus

Edit Bus Page is shown in “Figure 6.1.4” and codes are cited in “Appendix I”.

Figure 6.1.4 - Screen shot of Update Bus

Changer Password is shown in “Figure 6.1.5” and codes are cited in “Appendix J”.

Figure 6.1.5 - Screen shot of Changer Password Interface

Bus Journey is shown in “Figure 6.1.6”.

Figure 6.1.6 - Screen shot of Display Bus Journey Interface

Entry Point is shown in “Figure 6.1.7”.

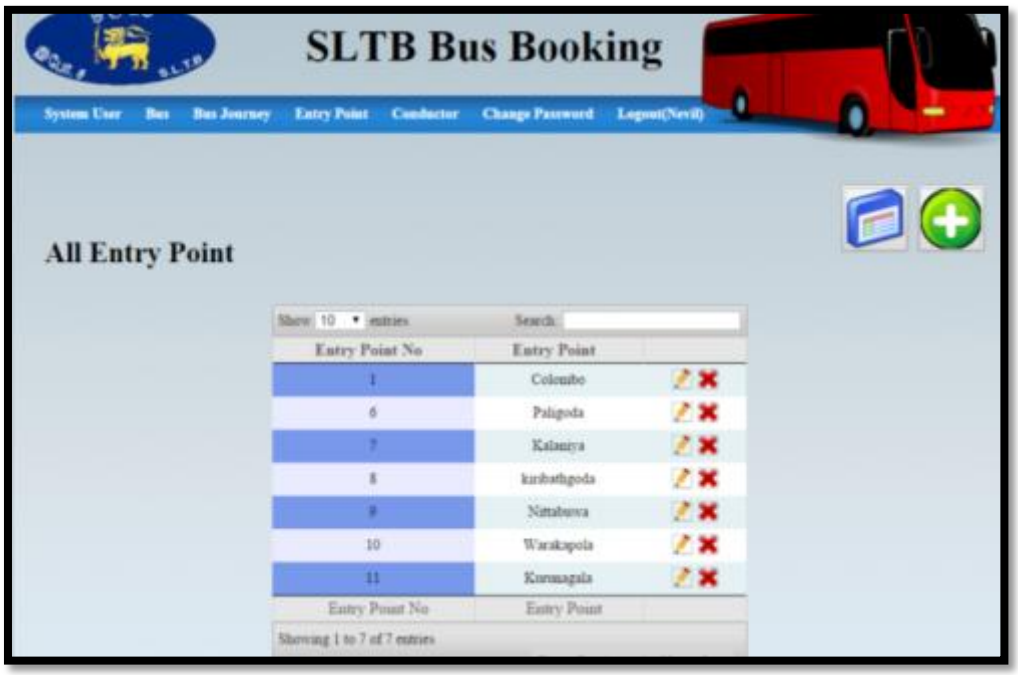


Figure 6.1.7 - Screen shot of Display Entry Point Interface

Conductor Information is shown in “Figure 6.1.8”.

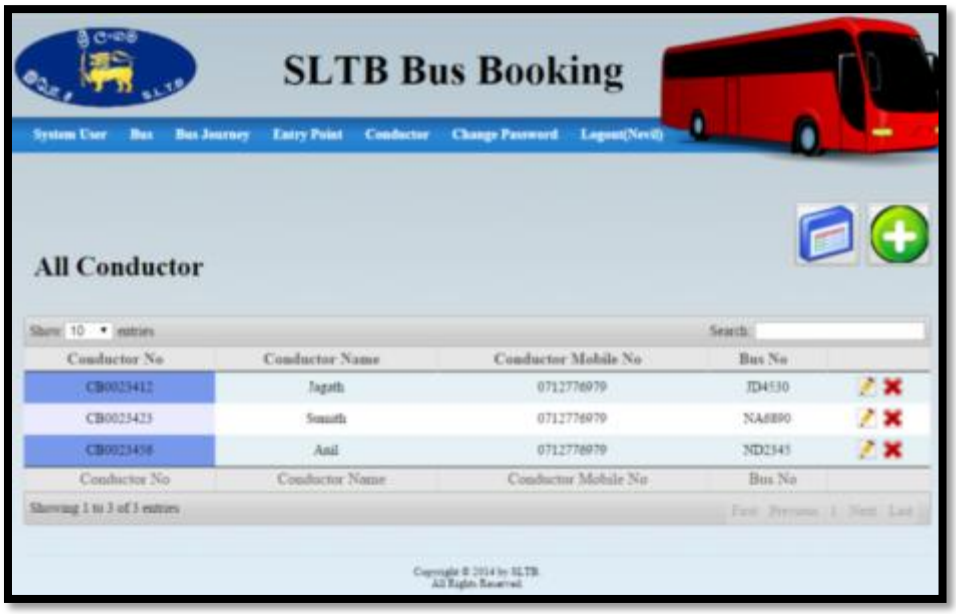


Figure 6.1.8 - Screen shot of Display Conductor Interface

Module 3 - Administration Module

Report Page is shown in “Figure 6.2.1”, Passenger Information Report is shown in “Figure 6.2.2”, Summary of Bus Booking Report is shown in “Figure 6.2.3” and codes are cited in “Appendix K”.

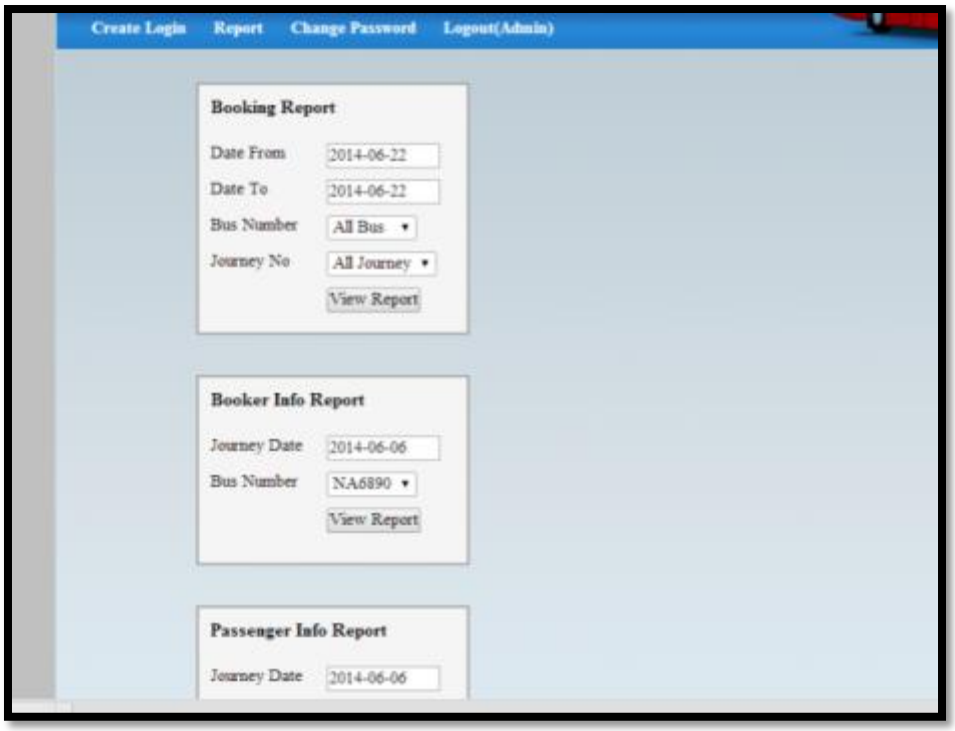
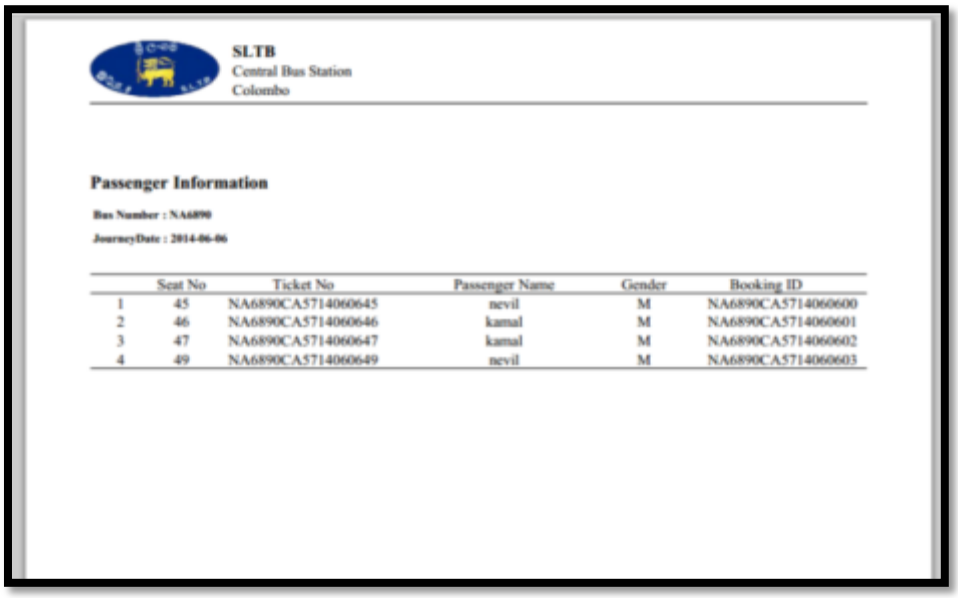


Figure 6.2.1 - Screen shot of View Report Interface



SLTB
Central Bus Station
Colombo

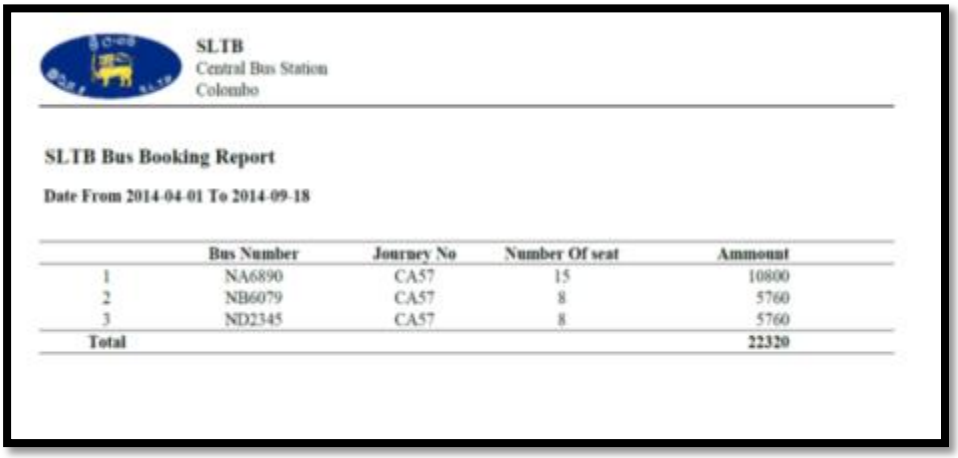
Passenger Information

Bus Number : NA6890

JourneyDate : 2014-06-06

	Seat No	Ticket No	Passenger Name	Gender	Booking ID
1	45	NA6890CA5714060645	nevil	M	NA6890CA5714060600
2	46	NA6890CA5714060646	kamal	M	NA6890CA5714060601
3	47	NA6890CA5714060647	kamal	M	NA6890CA5714060602
4	49	NA6890CA5714060649	nevil	M	NA6890CA5714060603

Figure 6.2.2 - Screen shot of Report



SLTB
Central Bus Station
Colombo

SLTB Bus Booking Report

Date From 2014-04-01 To 2014-09-18

	Bus Number	Journey No	Number Of seat	Ammount
1	NA6890	CA57	15	10800
2	NB6079	CA57	8	5760
3	ND2345	CA57	8	5760
Total				**22320**

Figure 6.2.3 - Screen shot of Summary of Bus Booking

Module 4 – Booking Module

Available Bus page is shown in “Figure 6.3.1” and codes are cited in “Appendix L”.

Figure 6.3.1 - Screen shot of Available Buses

Available Seat page is shown in “Figure 6.3.2” and codes are cited in “Appendix M”.

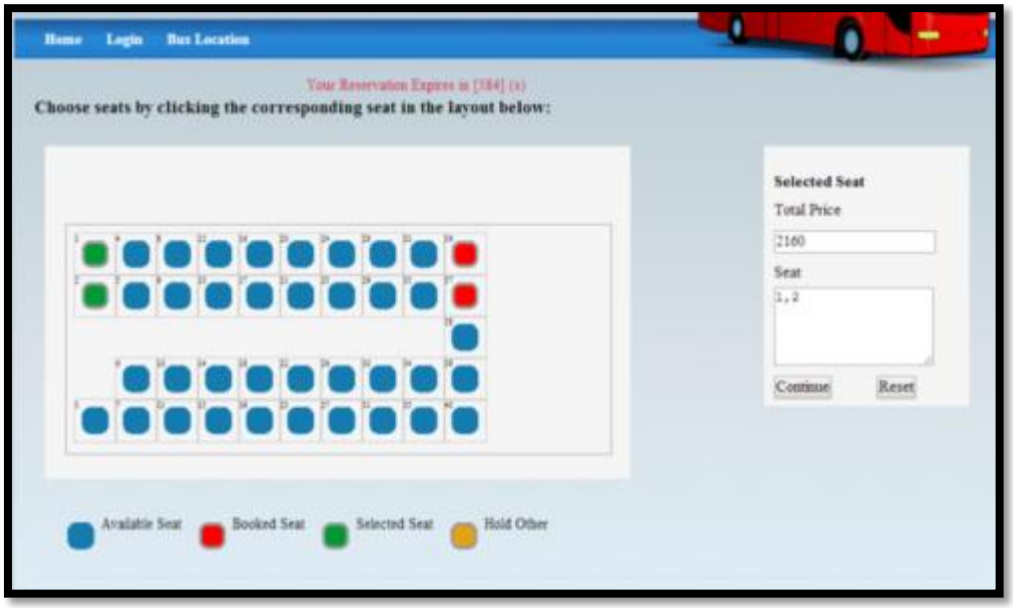


Figure 6.3.2 - Screen shot of Available Seat

Print Ticket is shown in “Figure 6.3.3” and codes are cited in “Appendix N”.

Figure 6.3.3 - Screen shot of Available Seat

Pay Payment

Figure 6.3.4 - Screen shot of Payment

Show Bus Location

Figure 6.3.5 - Screen shot of Show Bus Location

Module 5 – Tracking Module

Android Login Interface is shown in "Figure 6.4.1" and codes are cited in "Appendix O".

Figure 6.4.1 - Screen shot of Main Interface

Tracking Interface is shown in “Figure 6.4.2” and codes are cited in “Appendix P”.

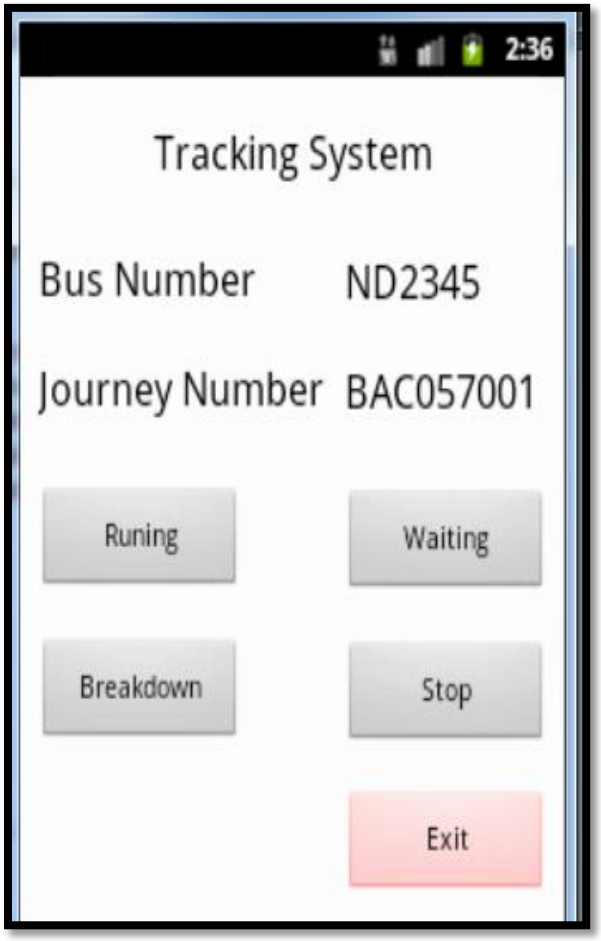


Figure 6.4.2 - Screen shot of Main Interface

Booked Seat Interface is shown in “Figure 6.4.3”

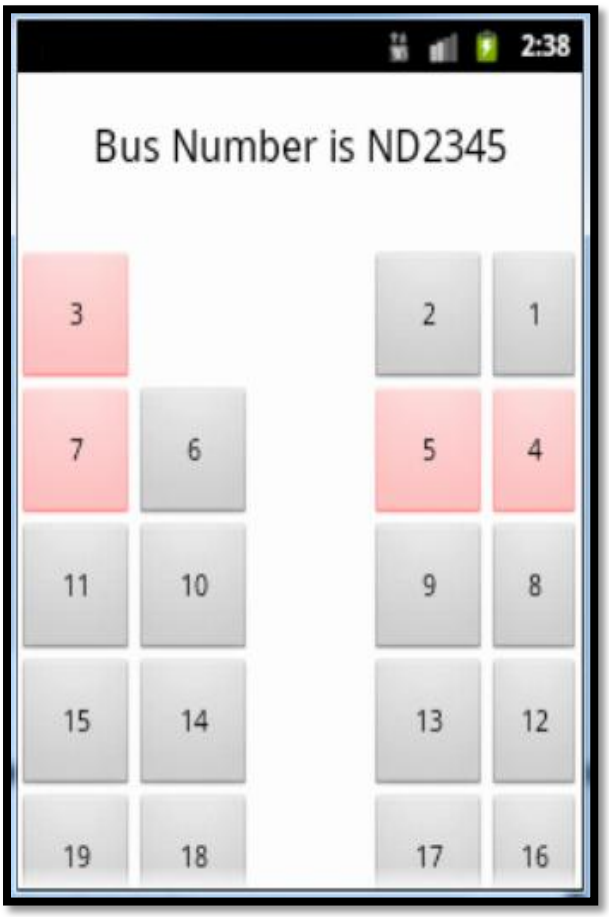


Figure 6.4.3 - Screen shot of Main Interface

6.4 Summary

This chapter state about, software, hardware, algorithms, pseudo codes, code segments for each module in the design and it shows the relation between design and implementing process.

Chapter 7 - System Testing and Evaluation

7.1 Introduction

In this chapter, testing of individual modules, integrated system testing and the observations are discussed. Also, in the first phase of the project, several objectives were set for the purpose of evaluation of final system. Therefore, this chapter is also devoted for examining the achievement of those objectives with respect to the evaluation criteria and outcomes.

7.2 Test Plan and Test Cases

System Operation Module

Login Interface

Test ID	Description	Expected Results	Status
1	Click Login Button without enter “Use name” and “Password”	Display Message “All required Filed”	Ok
2	Enter User Name	Show User Name on text filed as character.	Ok
3	Enter Password	Show Password on text filed as “******”	Ok
4	Enter incorrect “User name” and “Password” and Click Login Button	Redirect Login Interface and display Message “Invalid Use Name Or Password.”	Ok
5	Enter correct “User name” and “Password” and Click Login Button	Log into the Main Page.	Ok

Table 7.2.1 Test case for Login Interface

Add New Bus Interface

Test ID	Description	Expected Results	Status
1	Click Submit Button without enter Bus Data	Display Message “All required Filed”	Ok
2	Enter Bus Data with Privies “Bus No” and Click Button	Display Message “This Bus Already exist”	Ok
3	Enter Bus Data with difference “Bus No” and Click Button	Redirect Add New Bus Interface and Display Message “Add Data Successful”	Ok

Table 7.2.2 Test case for Add New Bus Interface

Update Bus Data Interface

Test ID	Description	Expected Results	Status
1	Click Edit Button	Go to Update Bus Page and Show Bus Data in Filed relevant Bus No	Ok
2	Change Bus Data and Click Submit Button	Redirect Update Bus Interface and Display Message “Update Data Successful”	Ok

Table 7.2.3 Test case for Update Bus Data Interface

Change Password Interface

Test ID	Description	Expected Results	Status
1	Click Login Button without enter Data	Display Message “All required Filed”	Ok
2	Enter “Current Password”	Show Current Password on text filed as “******”	Ok
3	Enter “New Password”	Show New Password on text filed as “******”	Ok
4	Enter “Conform New Password”	Show Conform New Password on text filed as “******”	Ok
5	Enter “Current Password” and “New Password” and enter incorrect “Conform New Password”	Display Message “New Password and Conform Password Not Match.”	Ok
6	Enter Incorrect “Current Password” and “New Password” and enter correct “Conform Password”	Display Message “Enter Correct Current Password”	Ok
7	Enter correct “Current Password” and “New Password” and enter correct “Conform Password”	Redirect Change Password Interface and Display Message “Password Change Successful”	Ok

Table 7.2.4 Test case for Change Password Interface

Administration Module

View Repot Interface

Test ID	Description	Expected Results	Status
1	Click Booking Report	Display Booking Report as a PDF document	Ok
2	Click Booker Info Report	Display Booker Info Report as a PDF document	Ok
3	Click Passenger Info Report	Display Passenger Info Report as a PDF document	Ok

Table 7.2.5 Test case for View Report Interface

Booking Module

Select Bus Seat Interface

Test ID	Description	Expected Results	Status
1	Click corresponding blue color(available seat) Seat	Changer Blue color as Green color (selecting seat).	Ok
2	Click brown color(hold other)	Display Message “this seat hold other”	Ok
3	Click red color seat(Booked seat)	Display message “ this seat already booked”	Ok

4	Click available seat and wait 10 minutes.	Selecting seat and after 10 minutes deselect it.	Ok
5	Click selecting seat	Deselect selecting seat	Ok
6	Click Reset Button	Deselect all selecting seat	Ok
7	Click number of available seat	Display total price on text filed and display all seat number on text area.	Ok

Table 7.2.6 Test case for Select Bus Seat Interface

Tracking Module

Select Tracking Interface

Test ID	Description	Expected Results	Status
1	Enter Bus No and click Submit Button	Show Tracking Interface and Display Message “Update: Longitude: 79.24242424 Latitude:6.92424242” and Update database	Ok
2	Click Runing Button	Display Message “Bus is Runing” and Update database	Ok
3	Click Waiting Button	Display Message “Bus is Waiting” and Update database	Ok
4	Click Breakdown Button	Display Message “Bus is Breakdown” and Update database	Ok
5	Click Stop Button	Display Message “Bus is Stop” and Update database	Ok
6	Click Exit Button	Exit Tracking Interface	Ok

Table 7.2.7 Test case for Tracking Interface

7.3 Test Data and Test results

Some of the actual data which was selected by random was used for testing purpose. Mainly users, who are having different roles with different capabilities have added and tested with the expected results. The administrator of the system must be preset as only one administrator allowed.

7.4 Acceptance Test

Acceptance testing is a type of black-box testing performed on a system. OBBTS was fully tested using actual data in actual environment.

7.5 Summary

Testing is necessary to give a quality output. Through this chapter it has described the main testing methods carried out in the evaluation process of OBBTS. Next chapter is the final chapter of this thesis and it will describe the overall achievements and further works of OBBTS.

Chapter 8 - Conclusion & Further work

7.1 Introduction

This is the final chapter of this thesis which examines the achievements of the overall project objectives with respect to the initial scope. Also the problems encountered when carrying out the project (OBBTS) and the solutions found are discussed. Thereafter, current system limitations are explained along with possible further enhancements.

7.2 - Conclusion

Initial scope of the project is described by the objectives set at early stages, which are listed in *Aim & Objectives of this Project* of Chapter 1 (*Section 1.3*). According to feedback received from testers that, the initial project scope is almost achieved by the end of final documentation. All functionality has been implemented with proper testing. This suggests that 99% of the whole system has been achieved. The system is under perfect working condition which can be made available for the production with fewer enhancements

7.3 - Problems encountered / limitations

In system design, problem encountered when drawing ER and Class diagram on identifying relevant entities and classes and their activities. Such ambiguity could be managed by referring relevant articles on the internet and ideas taken from the supervisor. And sending message after the confirmation to their phone was not in exact way. Because of getting a SMS gateway was expenditure. Getting a payment gateway is also expenditure therefore it would be done in further works. Creating a Mobile App for OBBTS does have a mean time of work, due to time I would develop it in future.

Time was a crucial factor to manage. It was a challenge to study a new subject in the context of bus booking system, gather the required skills, and utilize new

technology and implement such a system within the short span of time. After creating the system we have done system validation and verification process. However, almost all the requirements are gathered and analyzed, the design of the system is completely understood and thinking to implement a quality solution that goes beyond the expectations of the current issues of this system.

7.5 - Further work

The facility which does not included in this system is ticketing cancellation. In some other web based systems, they have provided an option to cancel the ticket. According to their company rules and regulations they will return the money. But here I have not included the cancellation facility. In future it may be included with the permission of SLTB. And OBBTS would be the exact way when SMS and payment gateway is purchased. The most important thing to finish is developing the Mobile App. It would make a revolution in online bus booking system.

References

Web References:

[1] Wikipedia. "PHP". Internet: http://en.wikipedia.org/wiki/PHP, Nov. 1, 2013 [Nov. 20, 2013]

[2] W3C. "HTML" . Internet: http://www.w3.org/standards/webdesign/htmlcss, [Nov. 26, 2013]

[3] W3C. "CSS" . Internet: http://www.w3.org/standards/webdesign/htmlcss, [Nov. 26, 2013]

[4] 2014 The jQuery Foundation. " jQuery ". Internet: http://jquery.com/, [Jan. 01, 2014]

[5] w3schools. "AJAX Introduction". Internet: http://www.w3schools.com/ajax/ajax_intro.asp, [Dec. 13, 2013].

[6] © 2014 Oracle Corporation. "Mysql". Internet: http://www.mysql.com/, [Accessed 3 - 01 - 2014]

[7] Apache. "Android". Internet: http://developer.android.com/reference/android/nfc/tech/package-summary.html, [Aug. 20, 2013]

[8] Wikipedia. "MVC". Internet: http://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller, Nov. 1, 2013 [Nov. 26, 2013]

[9] 2012 the Apache Software Foundation. "Apache". Internet: http://httpd.apache.org/download.cgi, [Dec. 20, 2013]

[10] Dialog Axiata PLC. "Ez cash". Internet: http://www.dialog.lk/personal/mobile/ez-cash/, [Dec. 5, 2013]

[11] Google. "Google Cloud Platform". Internet: https://cloud.google.com/, [Dec. 20, 2013]

Appendix A - Swim lanes for System Operation

Enter Bus Details

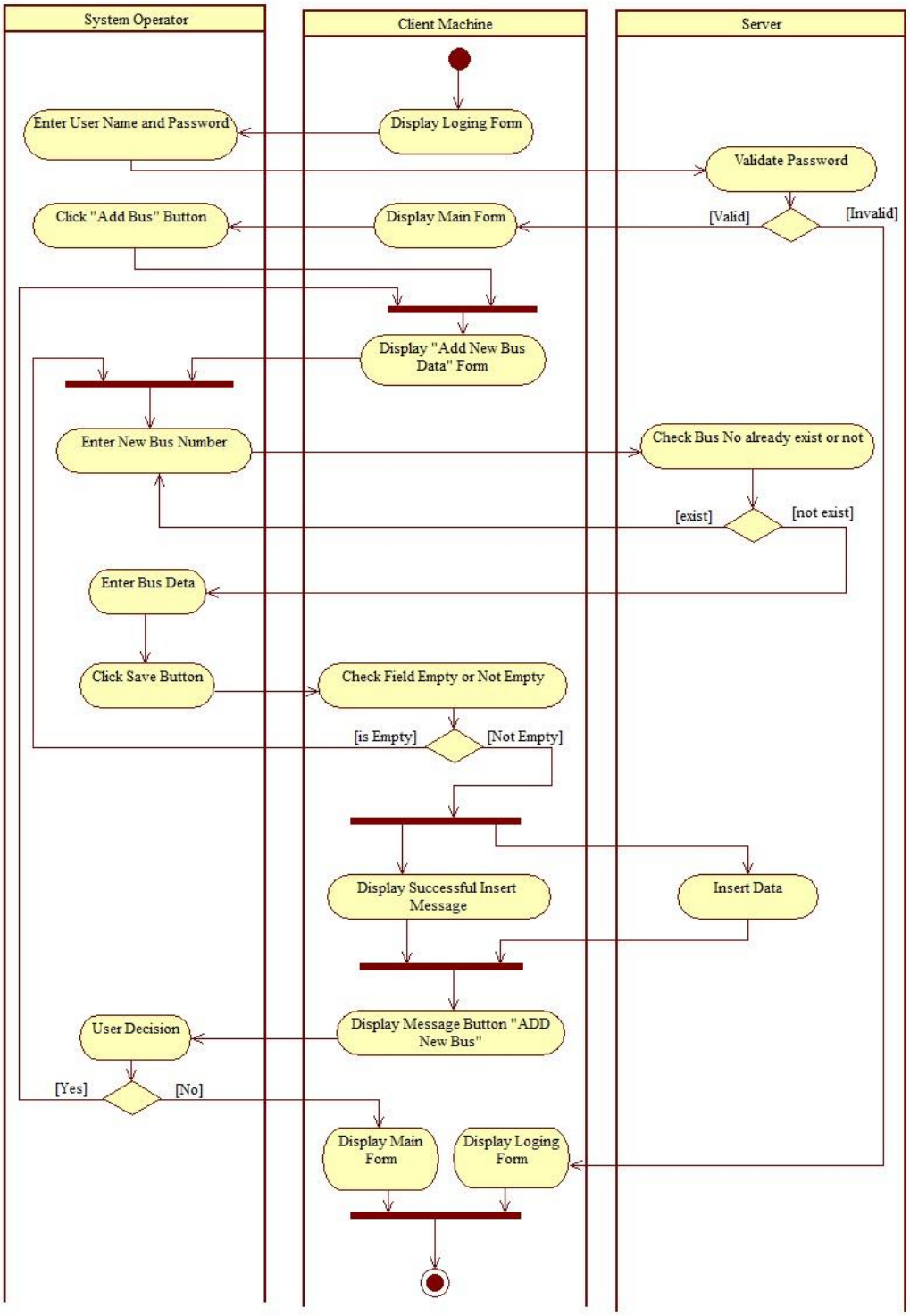


Figure 5.5.1- Enter Bus Details

Edit Bus Details

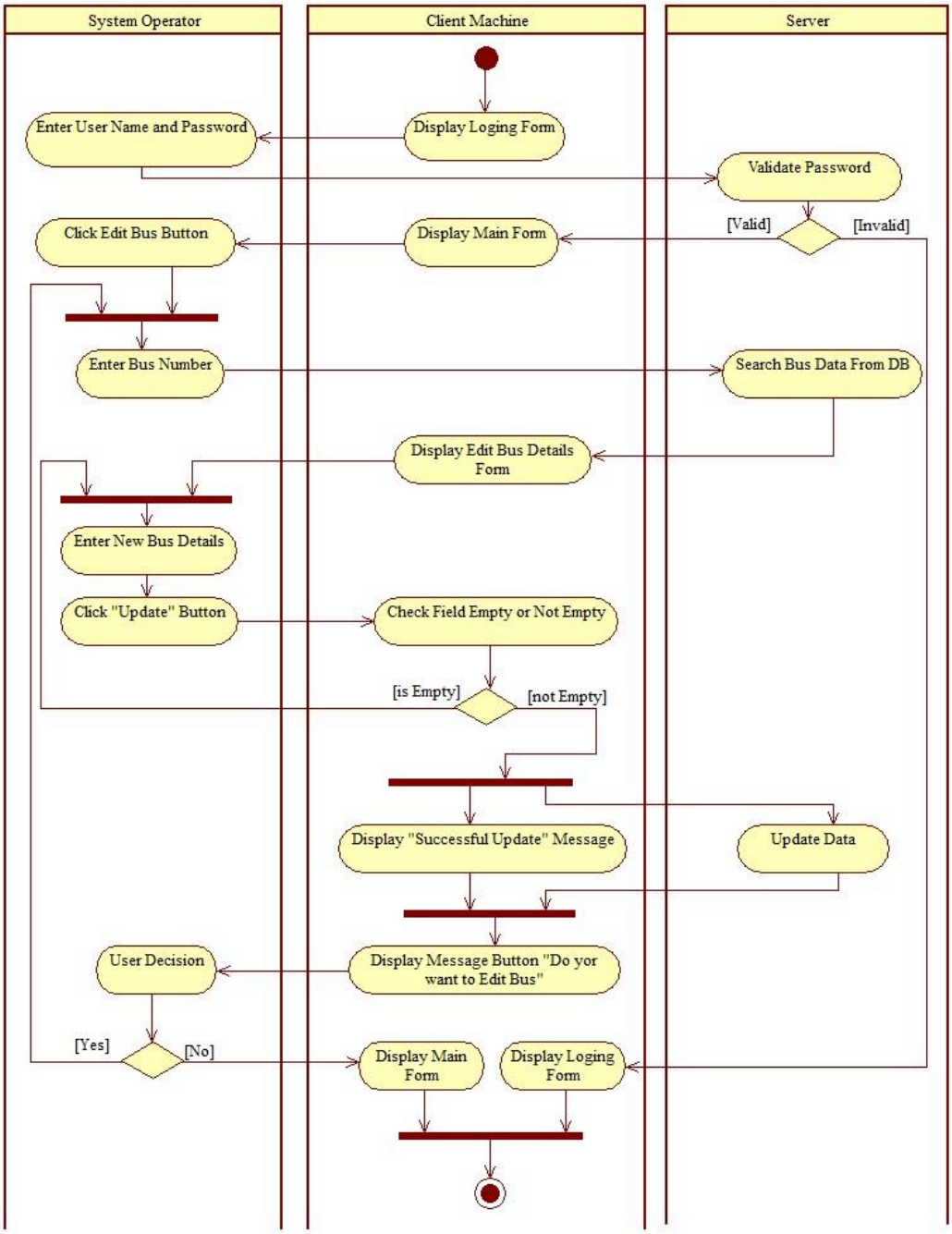


Figure 5.5.2- Edit Bus Details

Enter New Conductor Details

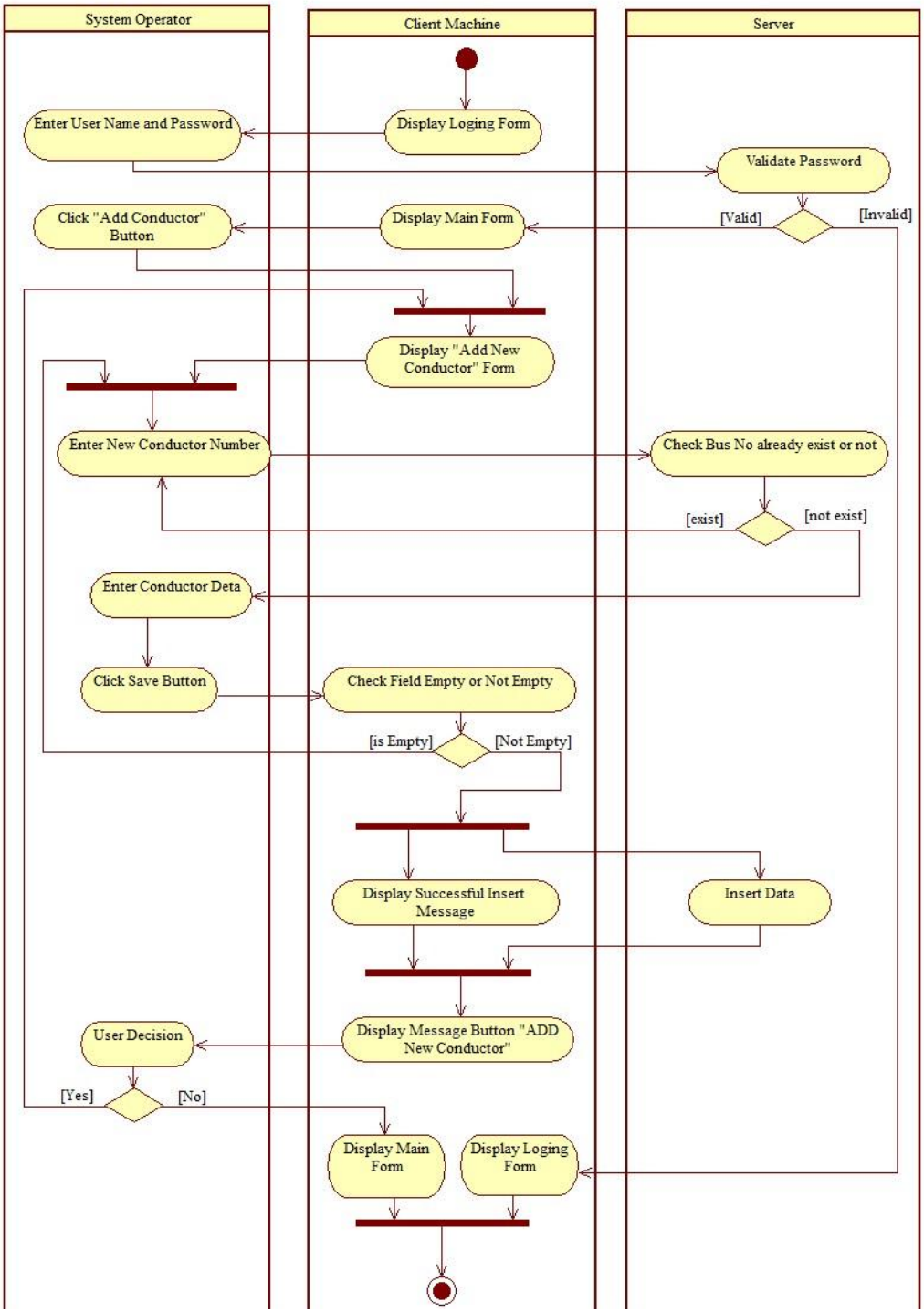


Figure 5.5.3 - Swim lanes for Enter Conductor Details

Edit Conductor Details

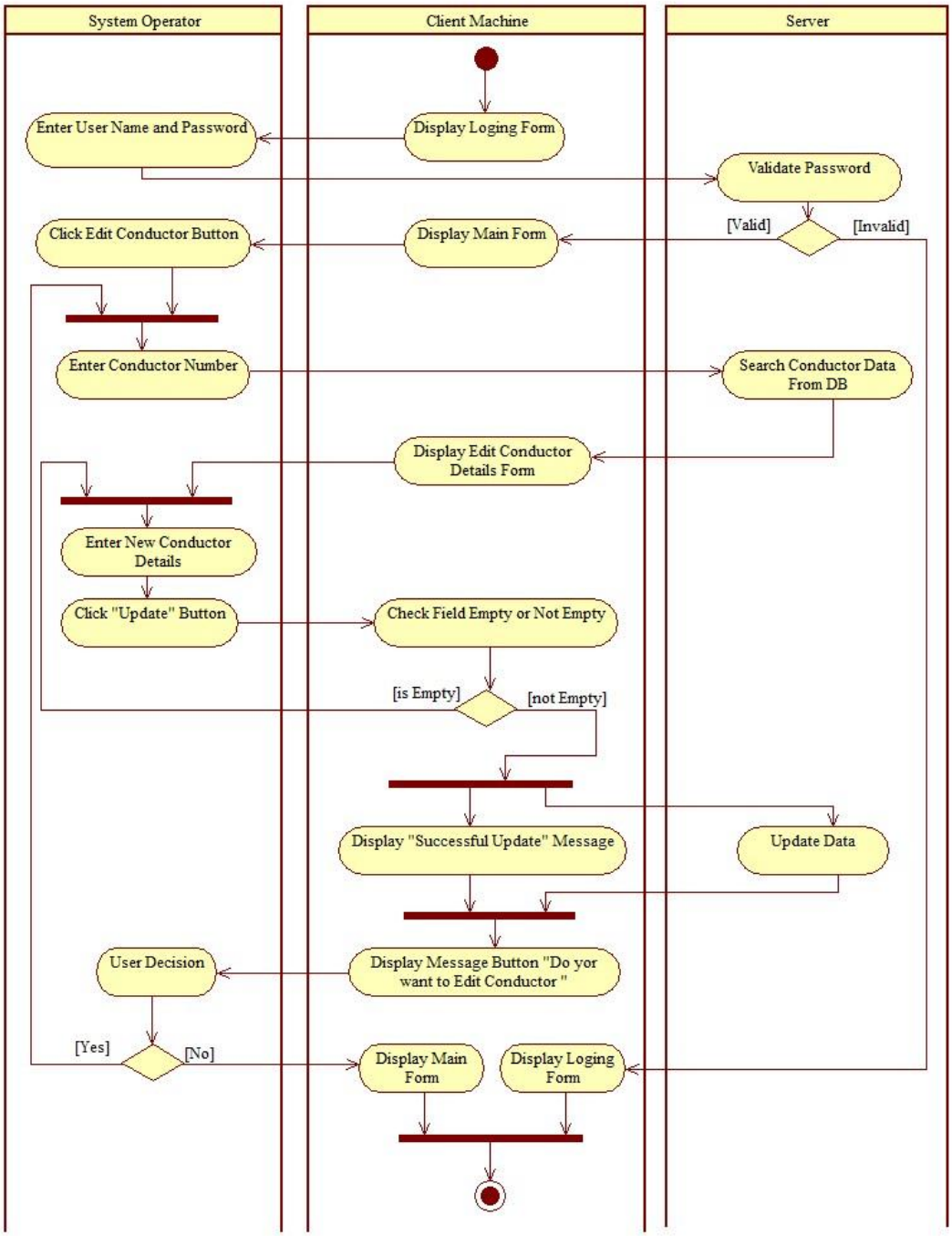


Figure 5.5.4 - Swim lanes for Edit Conductor Details

Allocate Conductor for Bus

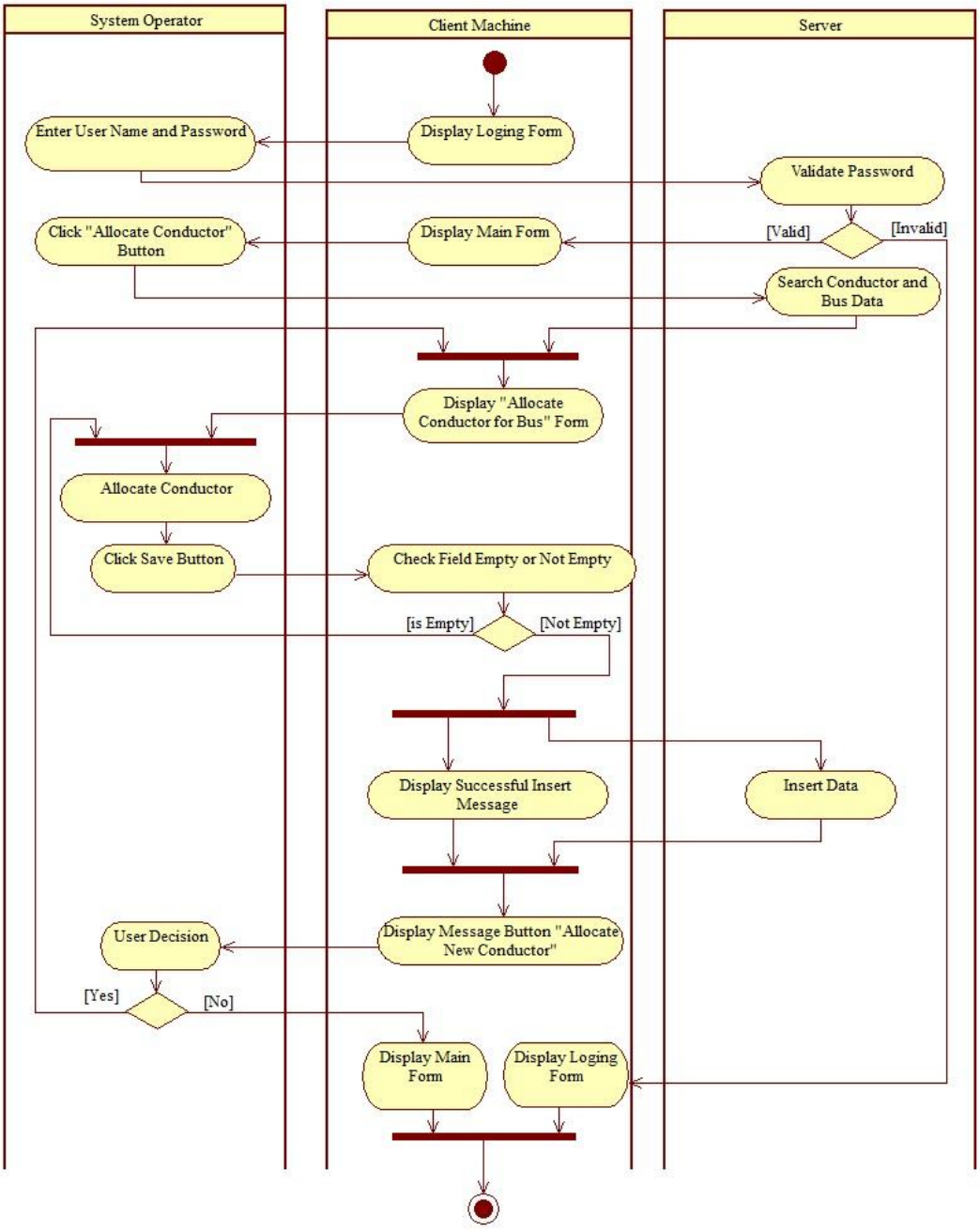


Figure 5.5.5 - Swim lanes for Allocate Conductor for Bus

Update Tickets Price

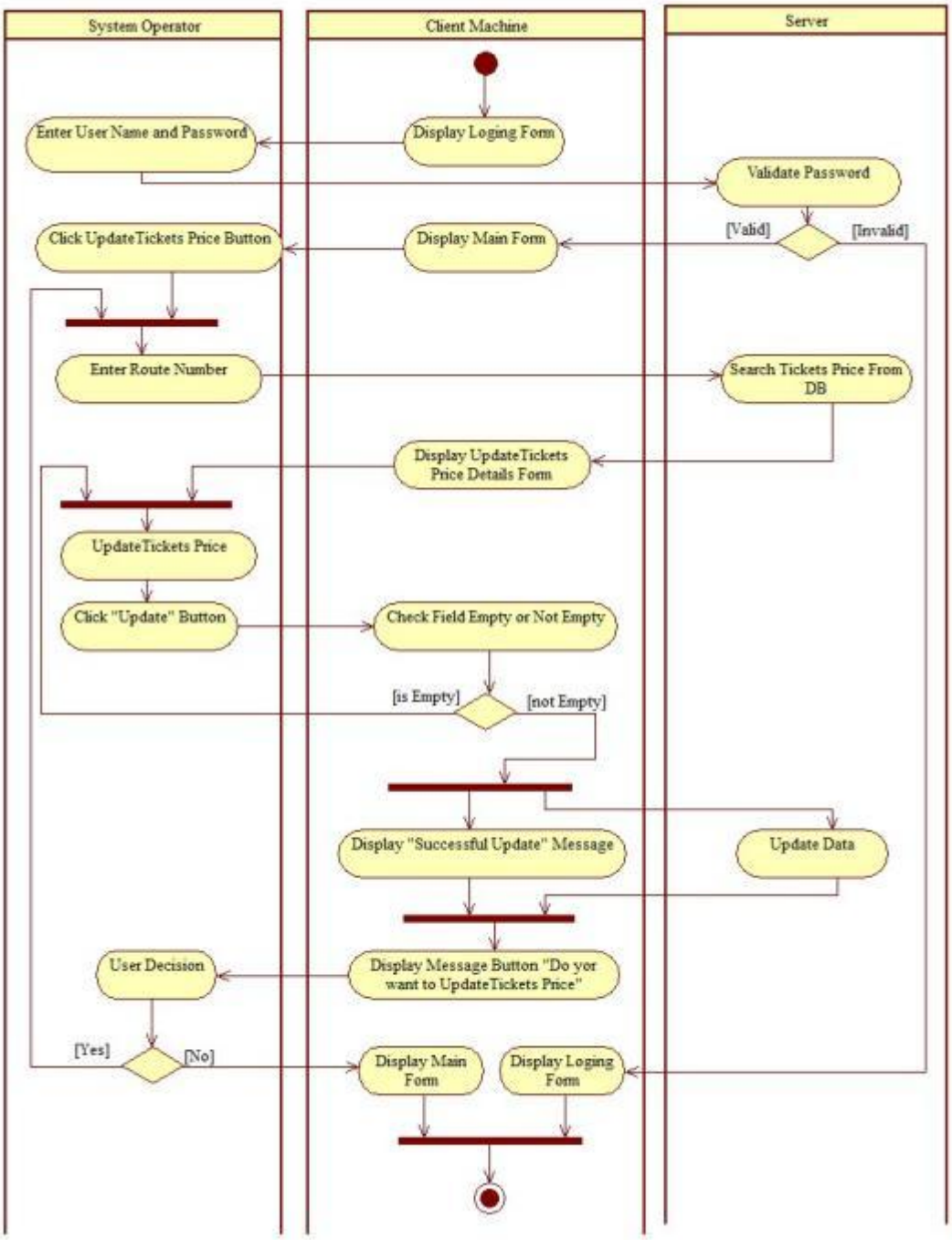


Figure 5.5.6 - Swim lanes for Update Tickets Price

Enter System User

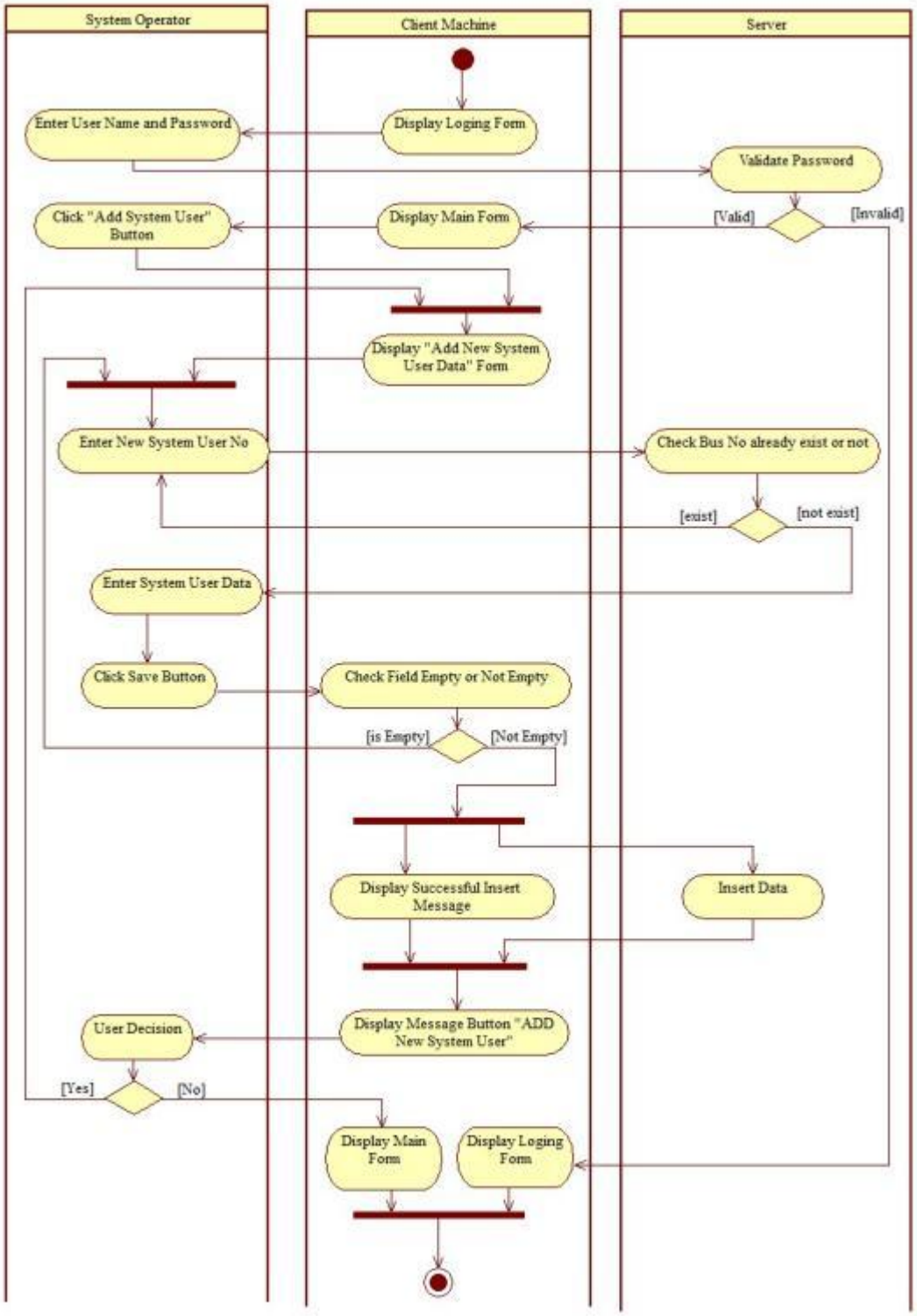


Figure 5.5.7 - Swim lanes for Enter System User

Edit System User

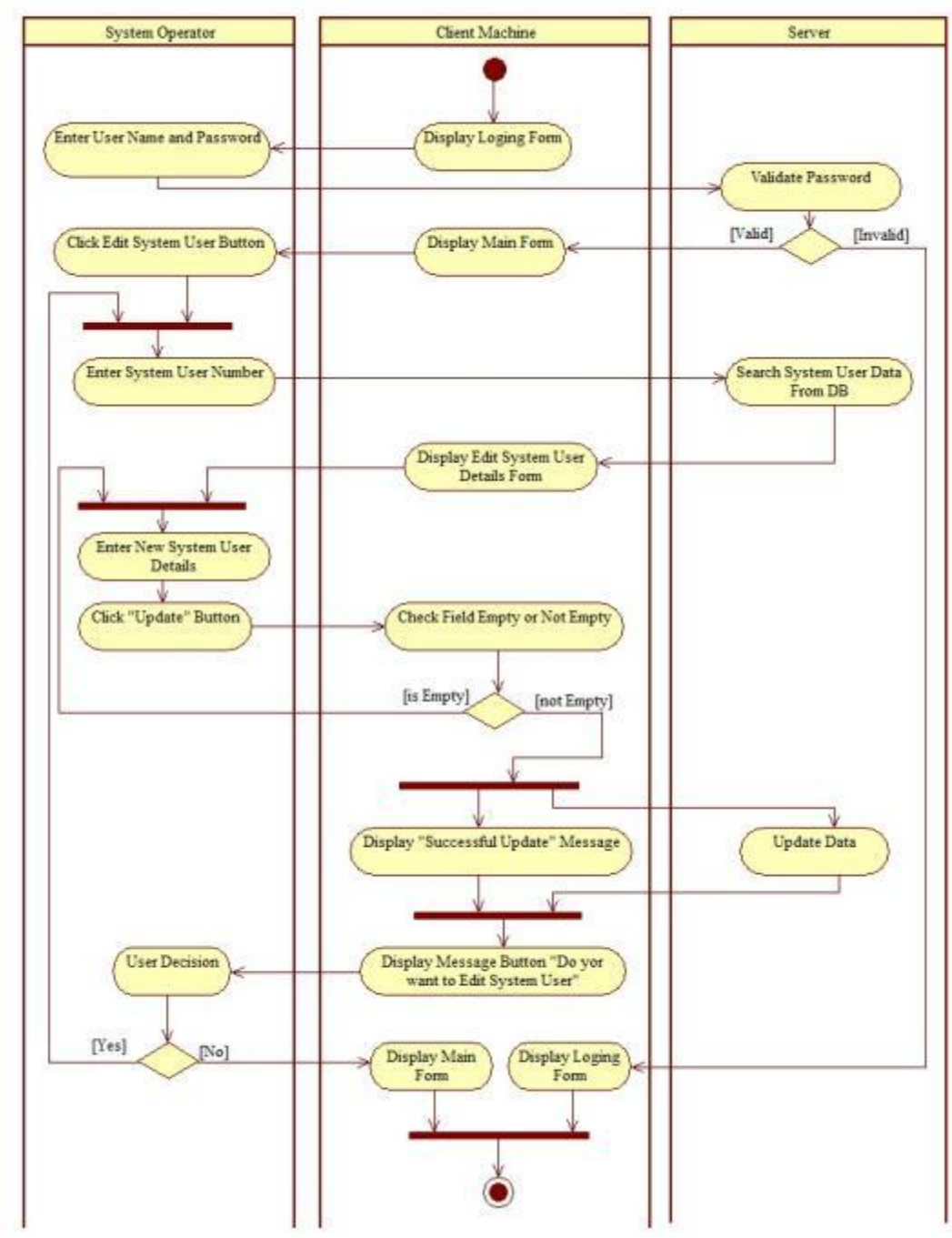


Figure 5.5.8 - Swim lanes for Update System User

Enter New Entry Point

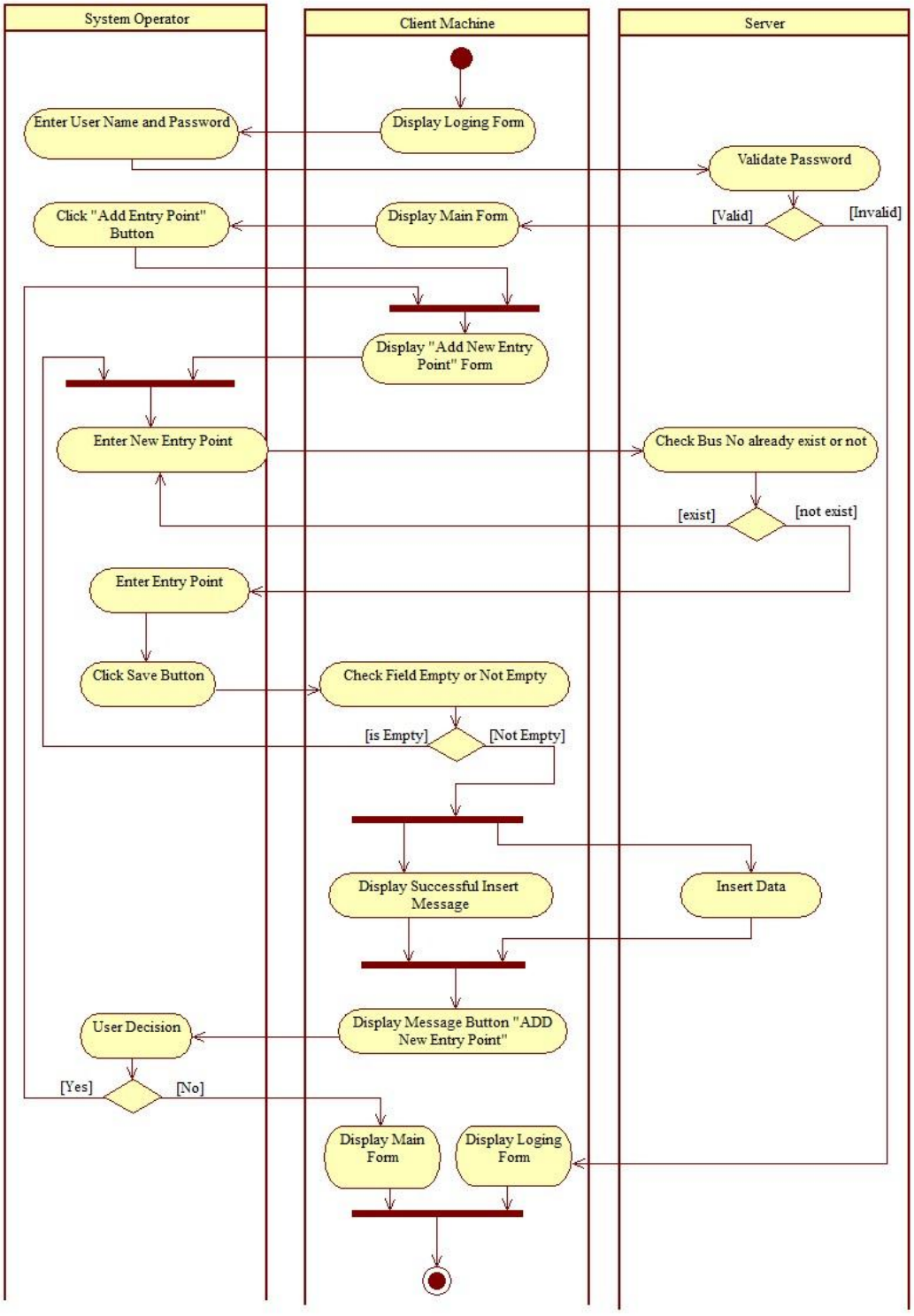


Figure 5.5.9 - Swim lanes for Enter New Entry Point

Edit Entry Point

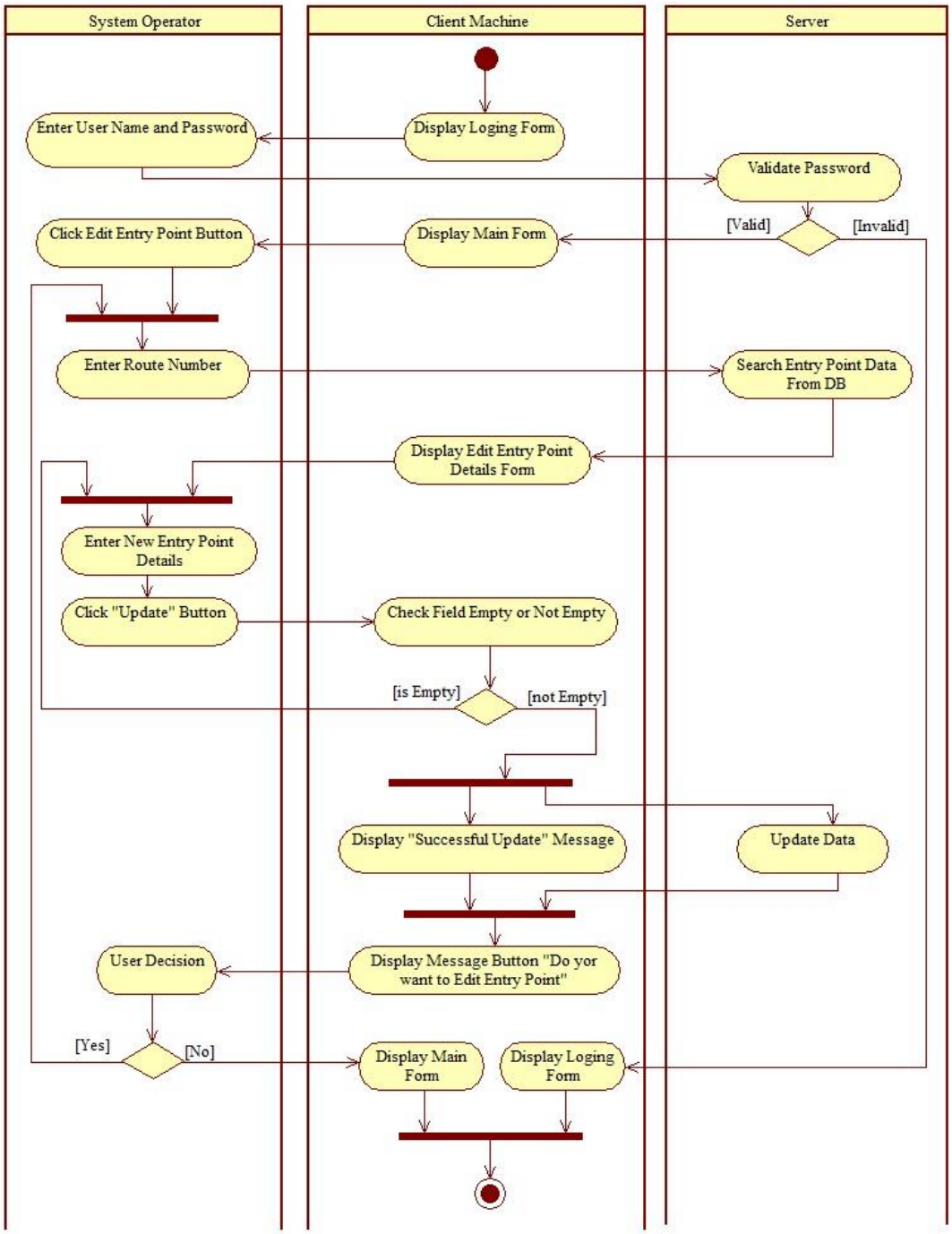


Figure 5.5.10 - Swim lanes for Edit New Entry Point

Appendix B - Swim lanes for Admin Operation

Create System User Login

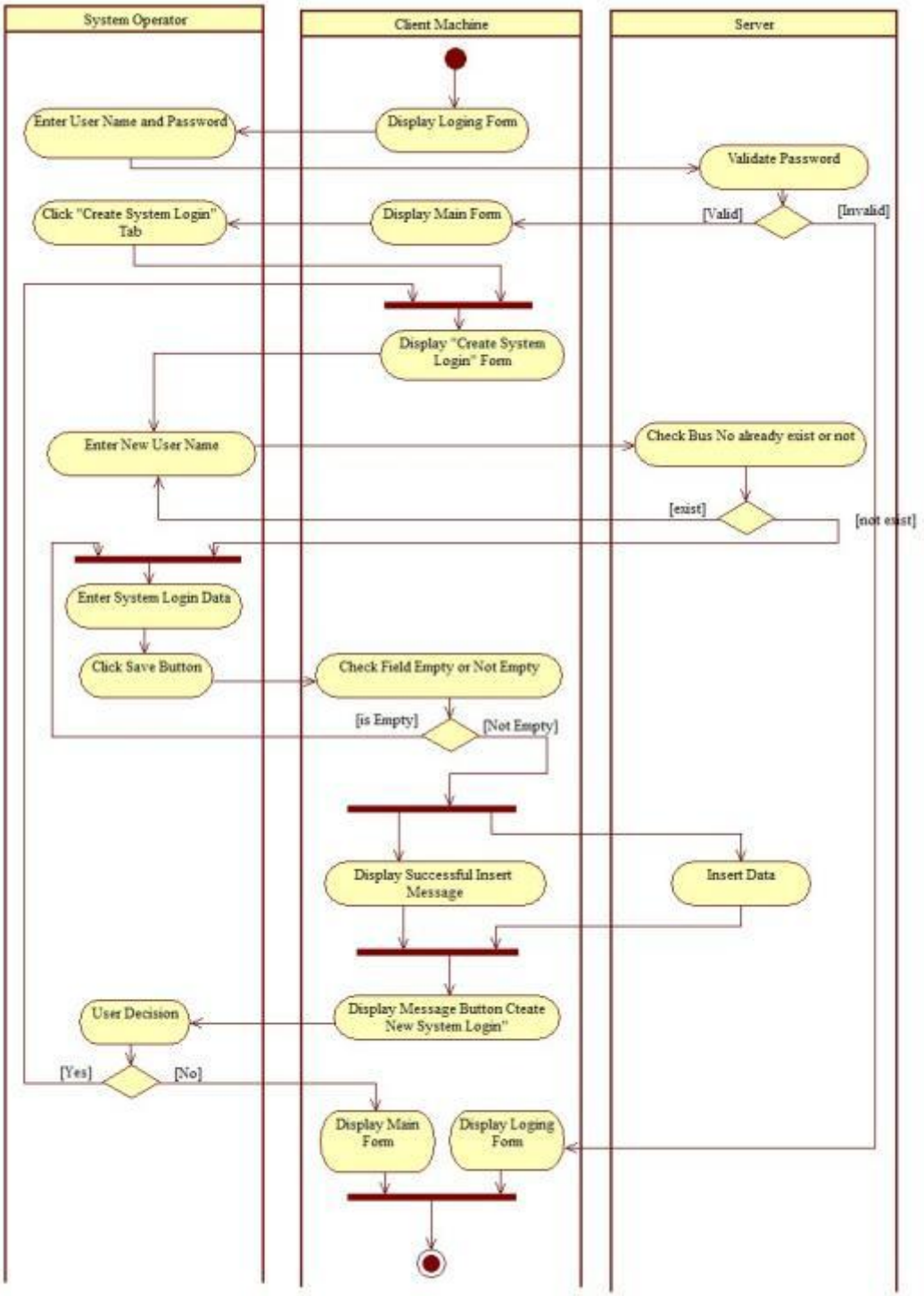


Figure 5.6.1 - Swim lanes for Create System User Login

Edit System Use Login

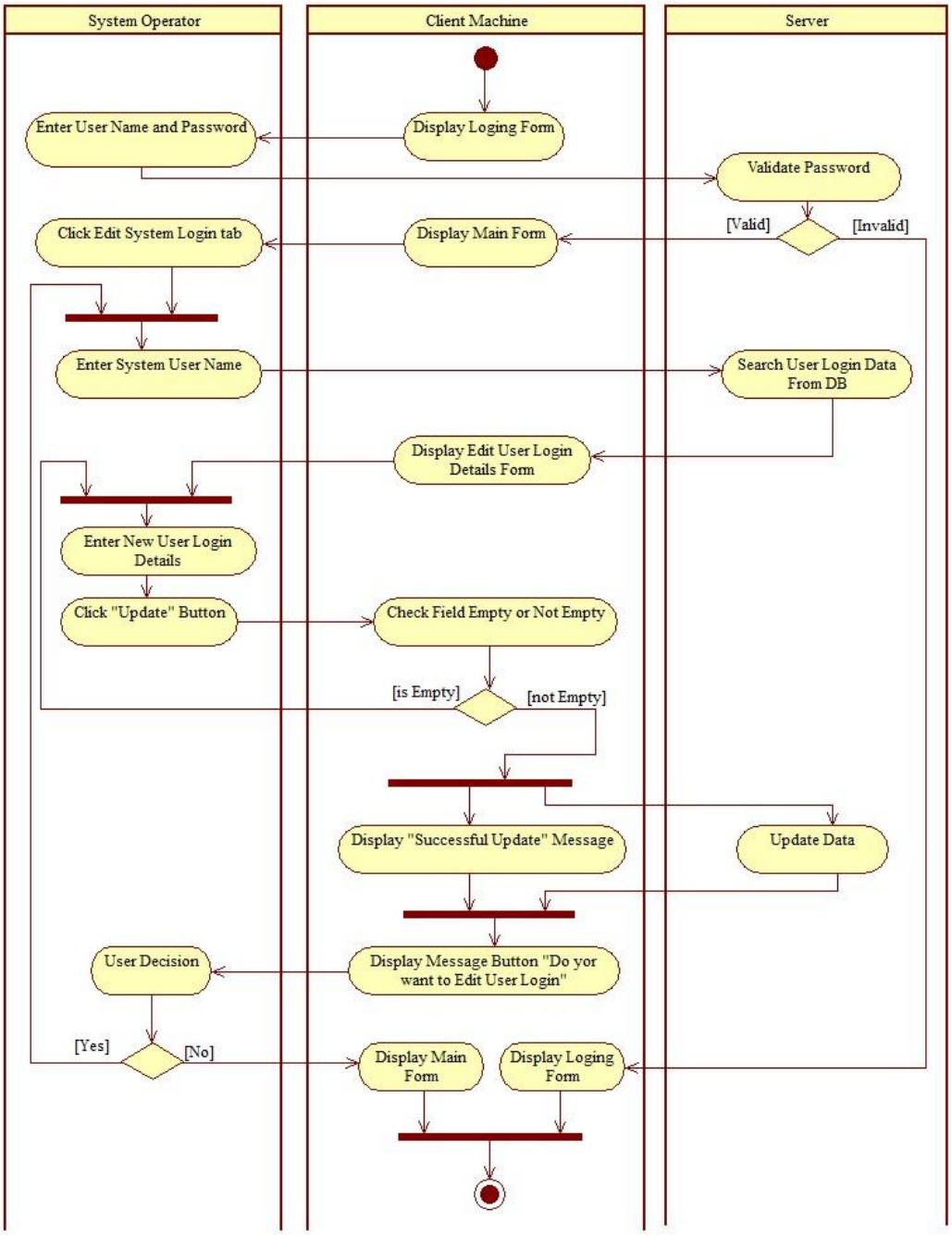


Figure 5.6.2 - Swim lanes for Edit System User Login

Manual Booking Report

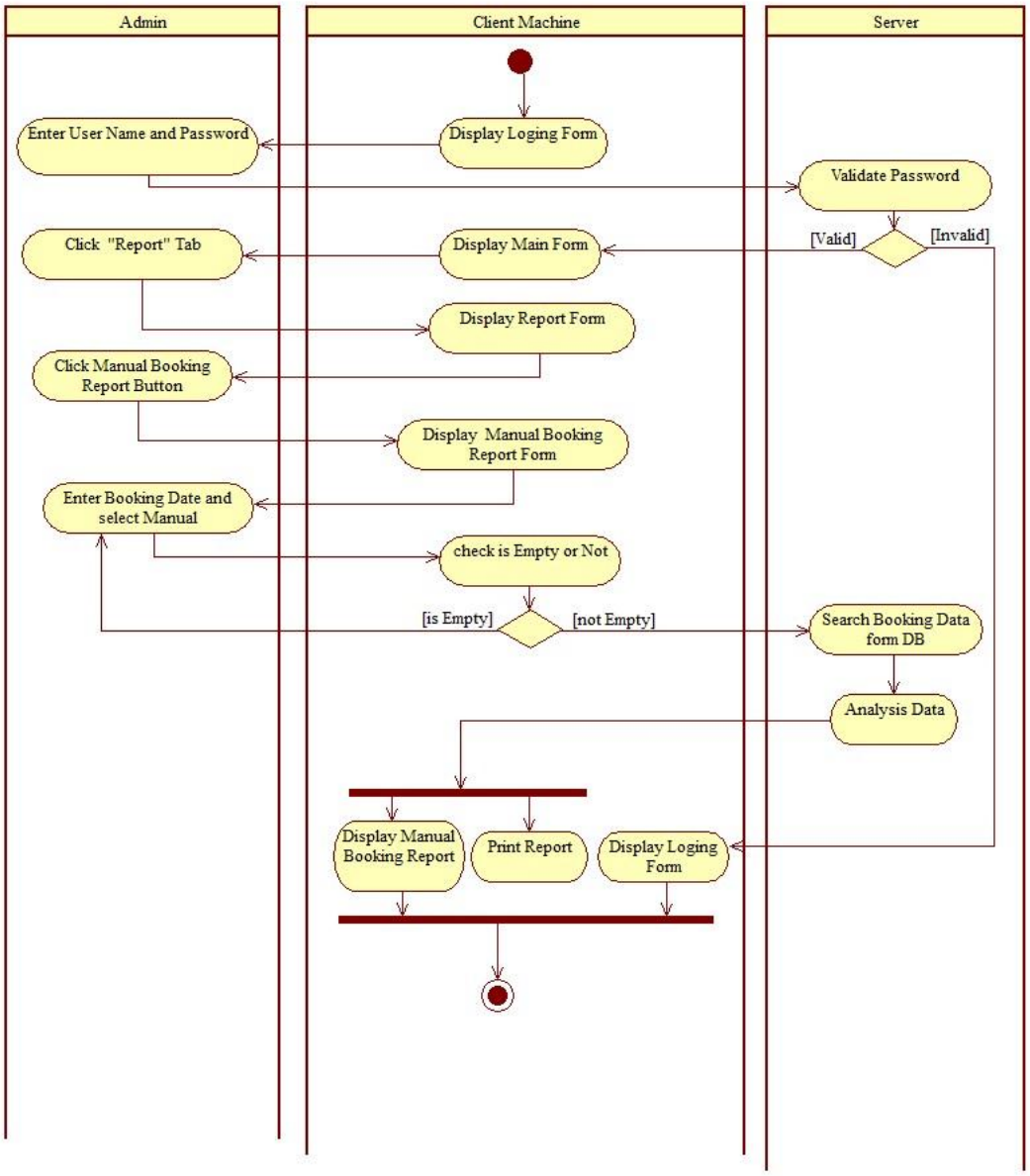


Figure 5.7.1 - Swim lanes for Manual Booking Report

Online Booking Report

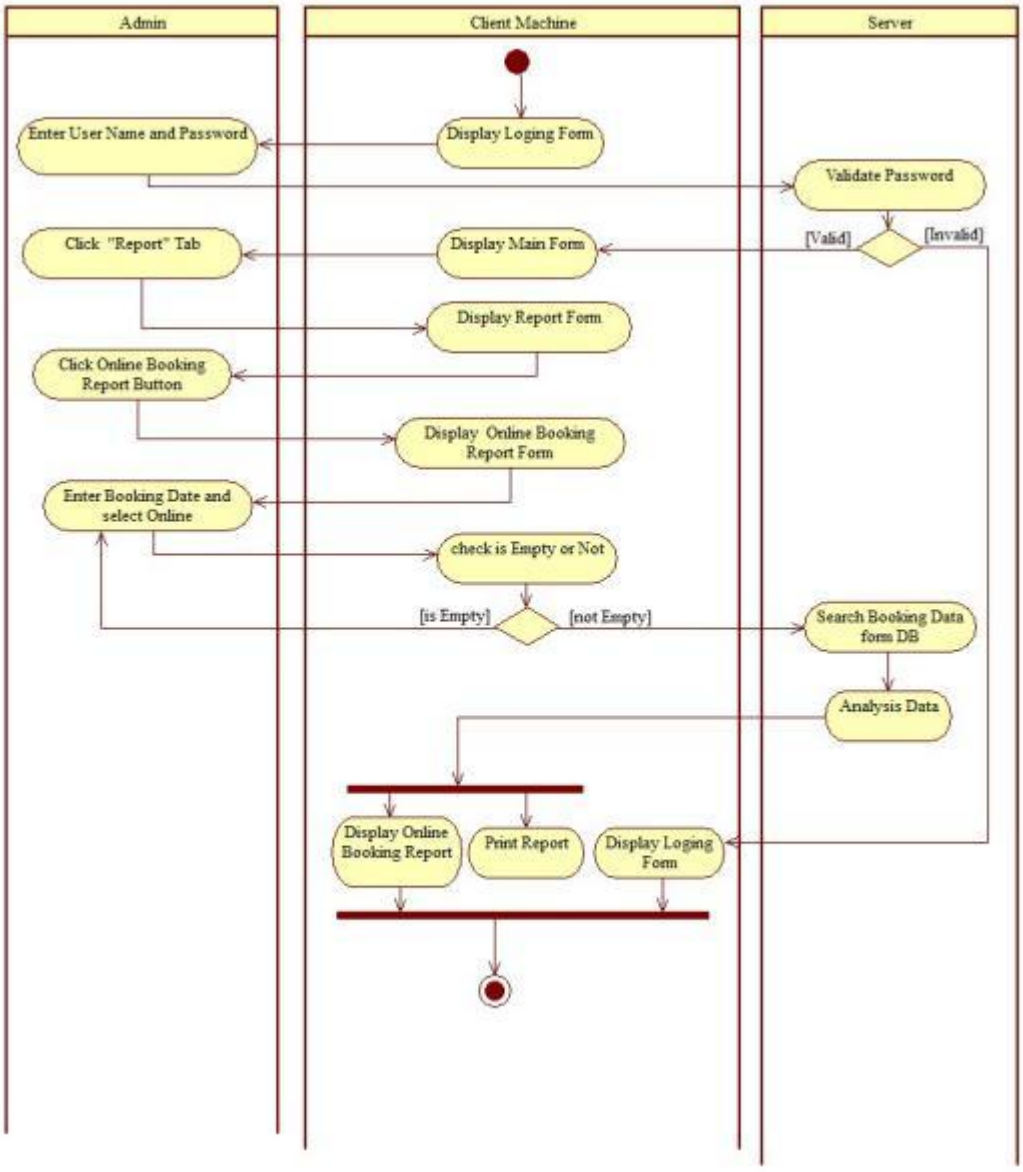


Figure 5.7.2 - Swim lanes for Online Booking Report

Daly Booking Report

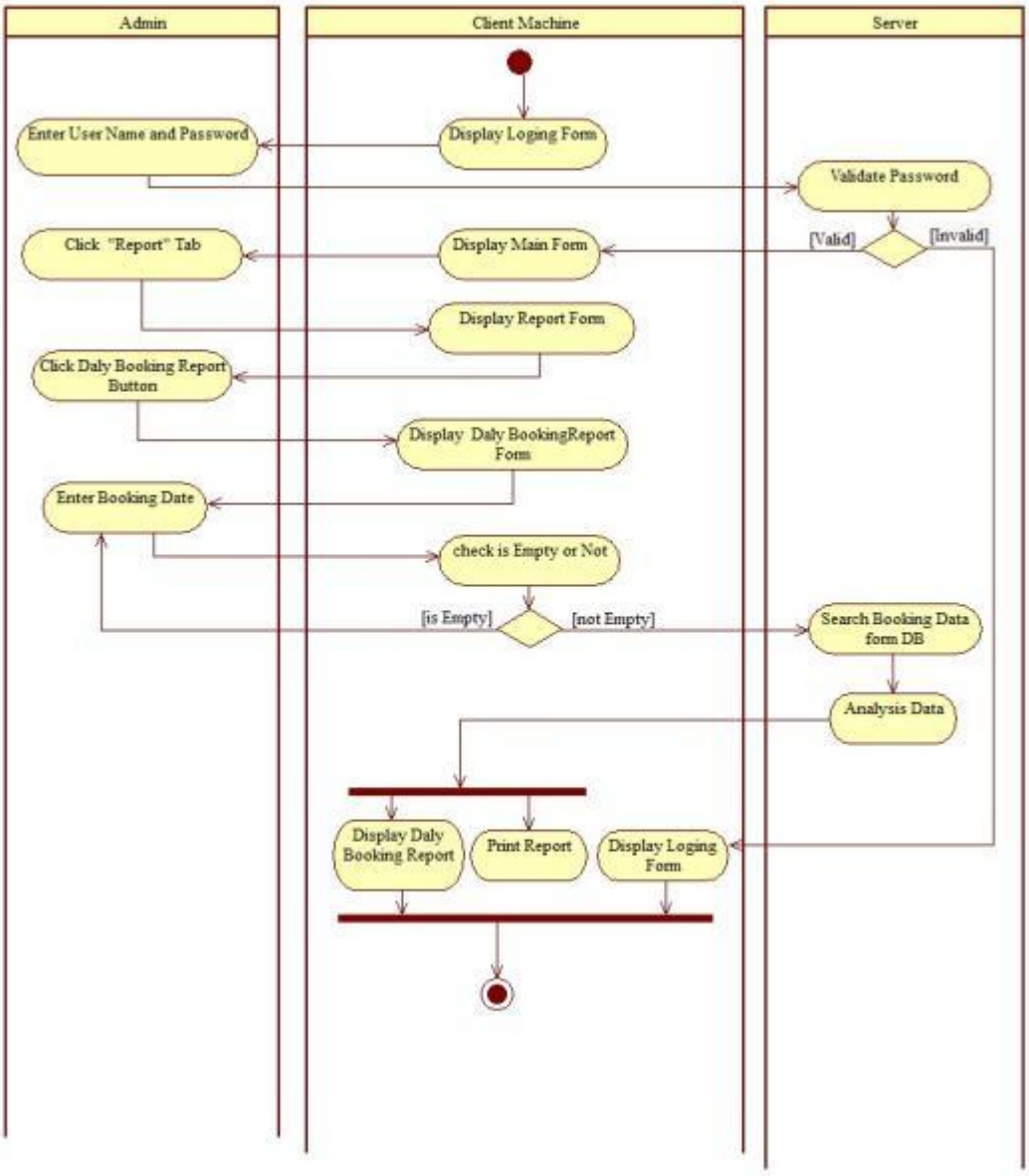


Figure 5.7.3 - Swim lanes for Daly Booking Report

Appendix C- Swim lanes for Booking Bus Seat

Search Available Bus

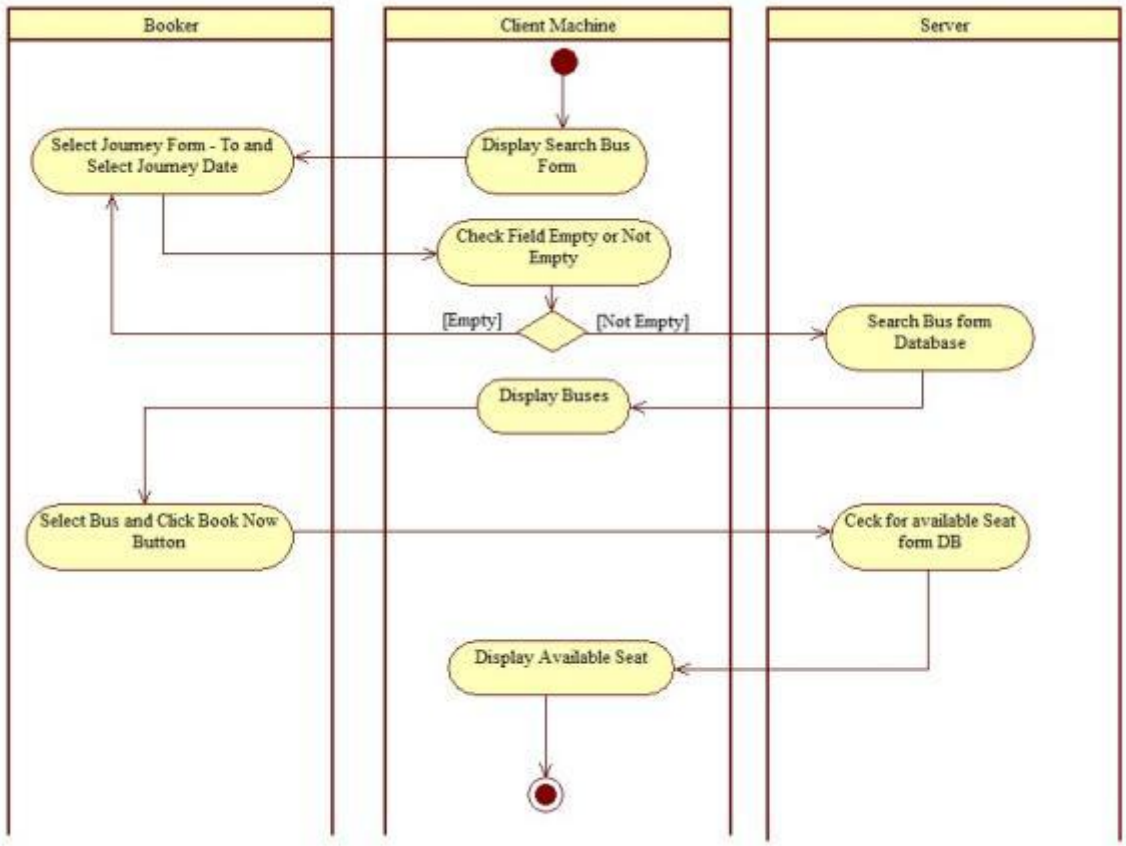


Figure 5.8.1 - Swim lanes for Search Available Bus

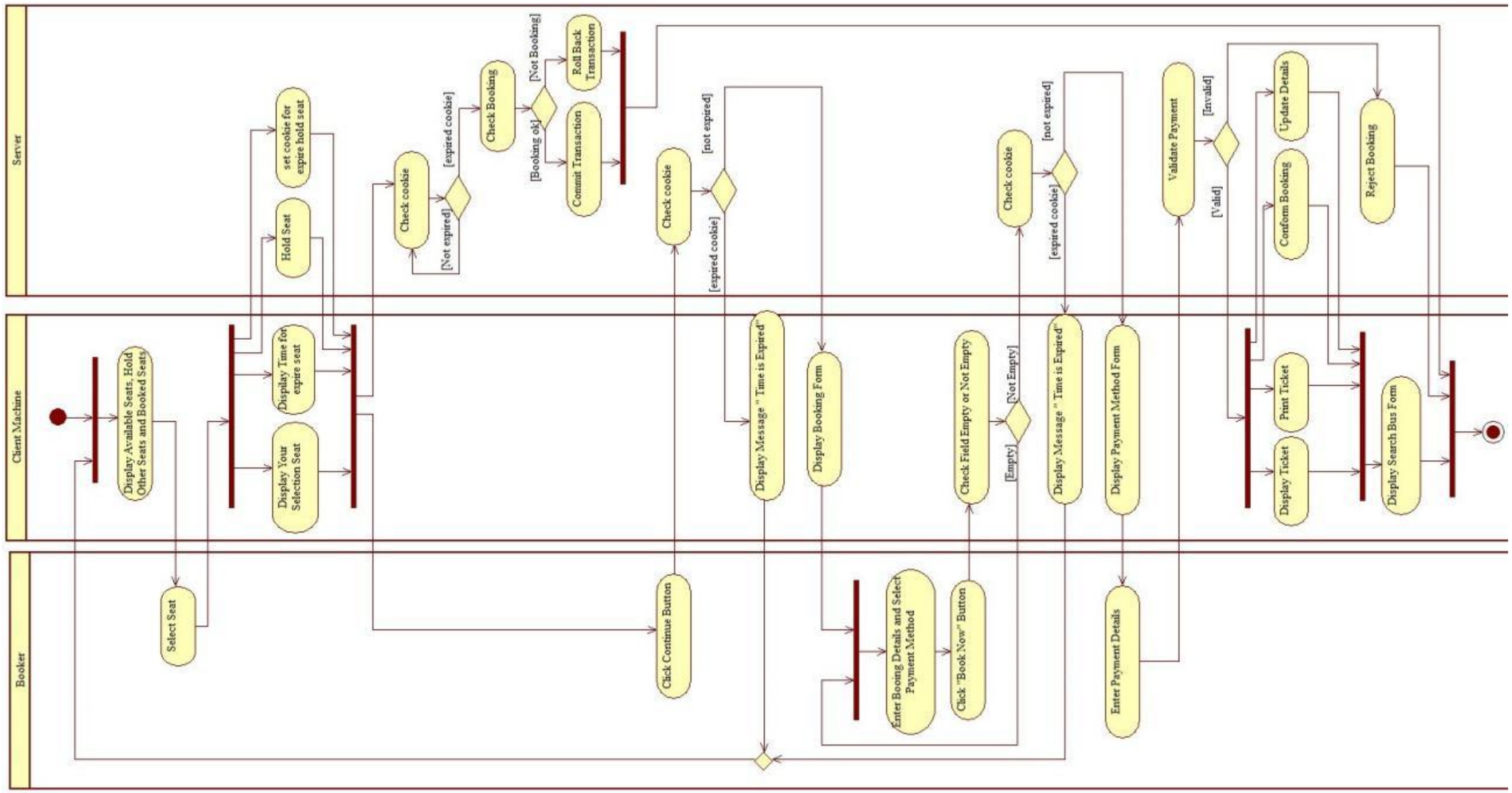


Figure 5.8.2 - Swim lanes for Booking Bus Seat

Appendix D - Swim lanes for Bus Tracking System

Track Bus Location

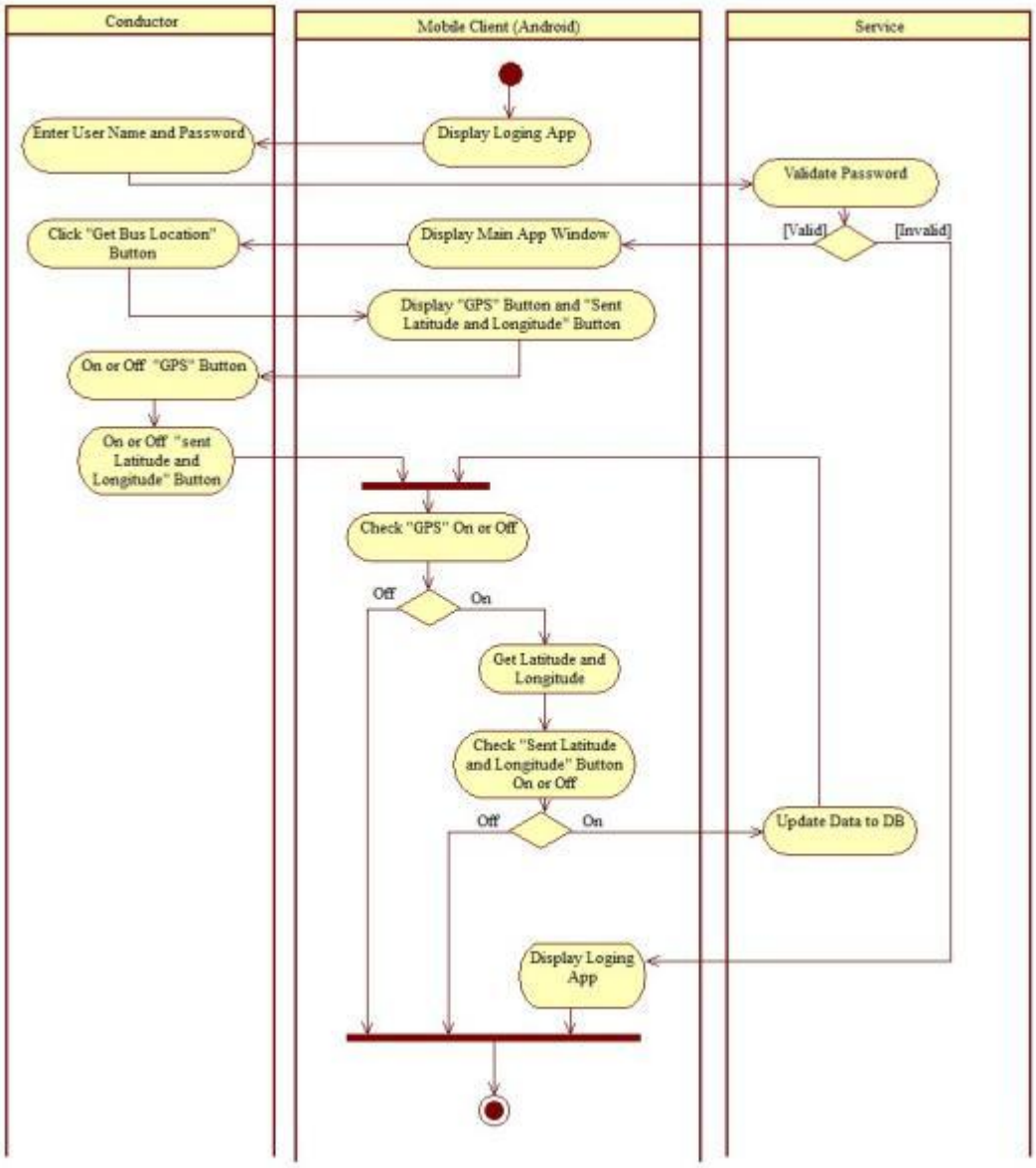


Figure 5.9.1 - Swim lanes for Track Bus Location

Show Bus Location on Map

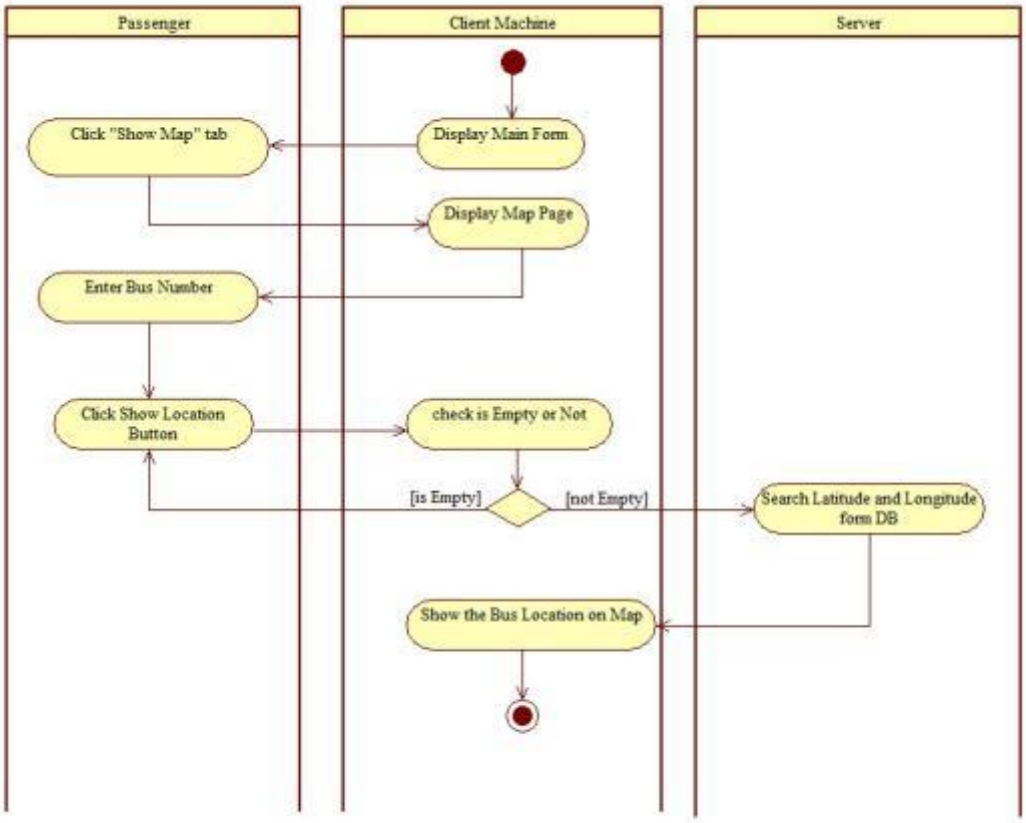


Figure 5.9.2 - Swim lanes for Show Bus Location on Map

Appendix E - Screen shot of the Database

sltb_booking Database

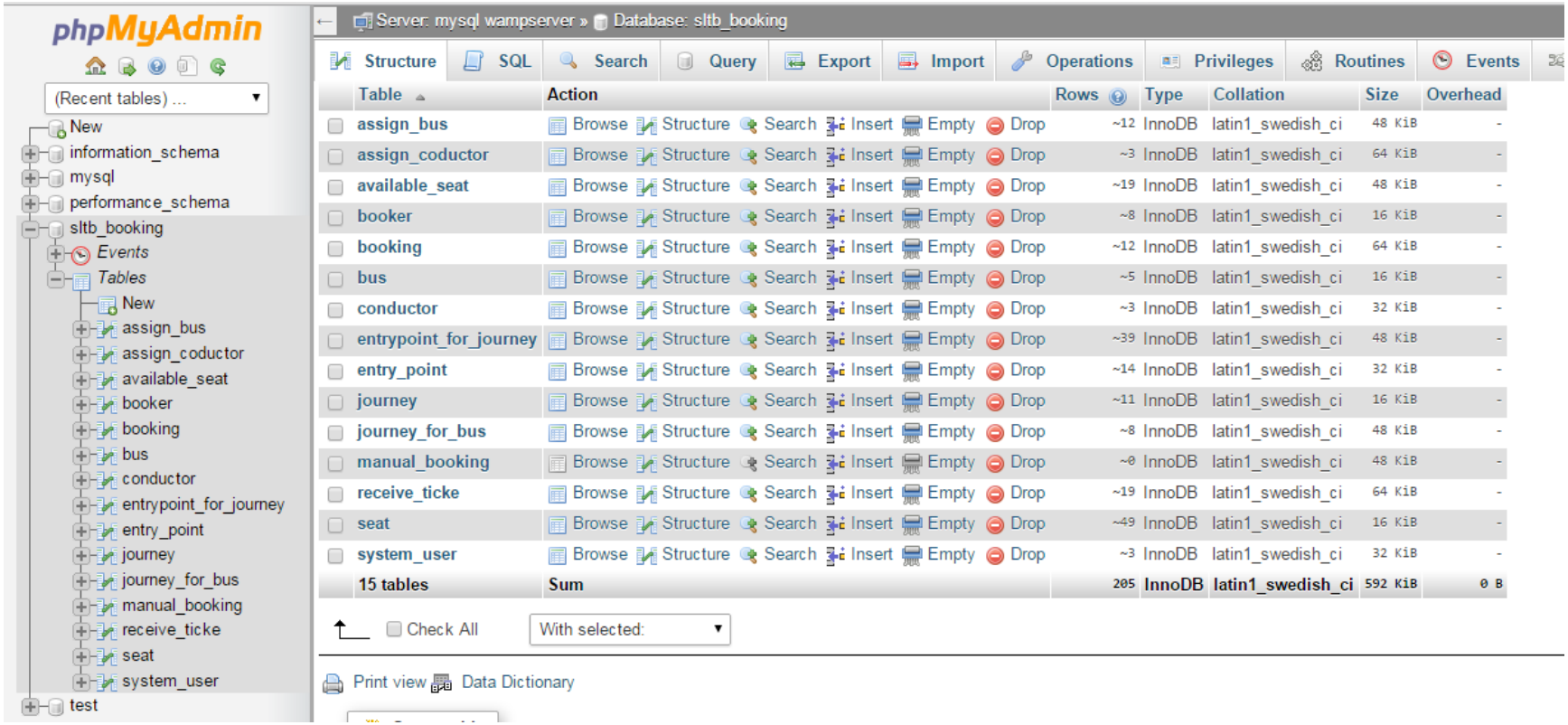


Figure 6.5.1 - Screen shot of the Database

assign_bus

Table comments: This Transaction Table is store who is assing Route for Bus

Column	Type	Null	Default	Comments
assign_bus_no	int(5)	No		this is primary key
userName	varchar(10)	No		System User Name
busNo	varchar(10)	No		Bus Route Number
date	date	No		Route assing Date
sql	varchar(1)	No		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	assign_bus_no	9	A	No	
userName	BTREE	No	No	userName	2	A	No	
				busNo	9	A	No	
routeNo	BTREE	No	No	busNo	9	A	No	

assign_coductor

Table comments: This Transaction Table is store who is assing conductor for Bus

Column	Type	Null	Default	Comments
assingConductorNo	int(11)	No		this is primary key
userName	varchar(10)	No		System User Name
conductorNo	varchar(10)	No		Conductor Number
date	date	No		Conductor assing Date

available_seat

Table comments: This Transaction Table is current Stauts a Bus Seat

Column	Type	Null	Default	Comments
seatNo	int(2)	No		Bus Seat Number
busNo	varchar(10)	No		SLTB Bus Number
journeyNo	varchar(7)	No		
status	varchar(2)	No		Seat Status
date	date	No		Status Date
time	time	No		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	seatNo	39	A	No	
				busNo	39	A	No	
				journeyNo	39	A	No	
				date	39	A	No	
seatNo	BTREE	No	No	seatNo	39	A	No	
				busNo	39	A	No	
busNo	BTREE	No	No	busNo	9	A	No	

booker

Table comments: This Master Table is store Bus Booker Data

Column	Type	Null	Default	Comments
bookerNICNo	varchar(10)	No		Bus Booker NIC Number
bookerName	varchar(20)	No		Booker Short Name
bookerMNo	varchar(10)	No		Booker Mobile Number

booking

Table comments: This Transaction Table is store Receive booking invoice

Column	Type	Null	Default	Comments
bookingID	varchar(20)	No		Bus Ticket Number
bookerNICNo	varchar(10)	No		Bus Booker NIC Number
busNo	varchar(10)	No		Bus Number
journeyNo	varchar(7)	No		
no_of_seat	int(2)	No		
entryPointNo	int(2)	No		
ammount	float	No		Total Amount of Bus ticket
date	date	No		Ticket receive Date
payment_status	varchar(2)	No	P	
s_bookin_time	time	No		

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	bookingID	21	A	No	
bookerNICNo	BTREE	No	No	bookerNICNo	21	A	No	
				busNo	21	A	No	
bookerNICNo_2	BTREE	No	No	bookerNICNo	21	A	No	
busNo	BTREE	No	No	busNo	7	A	No	

bus

Table comments: This Master Table is store Bus Data

Column	Type	Null	Default	Comments
busNo	varchar(10)	No		Bus Number
busModel	varchar(15)	No		Bus Model
numberOfSeat	int(2)	No		Number Of Seat

conductor

Table comments: This Master Table is store Conductor Data

Column	Type	Null	Default	Comments
conductorNo	varchar(10)	No		Conductor Number
conductorName	varchar(20)	No		Conductor Name
conductorMNo	varchar(10)	No		Conductor Mobile Number
busNo	varchar(10)	Yes	*NULL*	Assing Bus No

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	conductorNo	3	A	No	
busNo	BTREE	No	No	busNo	3	A	Yes	

entry_point

Table comments: This Master Table is store Entry Point for bus Route

Column	Type	Null	Default	Comments
entryPointNo	int(2)	No		Bus Entry Point No
entryPoint	varchar(15)	No		Bus Entry Point Name

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	entryPointNo	7	A	No	
entryPoint	BTREE	No	No	entryPoint	7	A	No	

entrypoint_for_journey

Table comments: This Transaction Table is assing Entry Point for Bus Route

Column	Type	Null	Default	Comments
entryPoint_for_journeyNo	int(3)	No		this is primary key
journeyNo	varchar(7)	No		Bus Route Number
entryPointNo	int(2)	No		Bus Entry Point Number

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	entryPoint_for_journeyNo	20	A	No	
entryPointNo	BTREE	No	No	entryPointNo	20	A	No	
journeyNo	BTREE	No	No	journeyNo	10	A	No	

journey

Table comments: This Master Table is store Bus Route Data

Column	Type	Null	Default	Comments
journeyNo	varchar(7)	No		
routeNo	varchar(5)	No		Bus Route Number
journeyFrom	varchar(10)	No		Bus Route Start Point
journeyTo	varchar(10)	No		Bus Route End Point
price	float	No		Bus Ticket Price

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	journeyNo	5	A	No	

journey_for_bus

Table comments: journey_for_bus

Column	Type	Null	Default	Comments
journey_for_bus_No	int(3)	No		
busNo	varchar(10)	No		
journeyNo	varchar(7)	No		
departureTime	varchar(8)	No		
arrivalTime	varchar(8)	No		

manual_booking

Table comments: This Transaction Table is store who is manual booking Booker

Column	Type	Null	Default	Comments
manualBookingNo	int(11)	No		this is primary key
userName	varchar(10)	No		System User Name
bookingID	varchar(20)	No		
date	date	No		ManualBooking Date

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	manualBookingNo	0	A	No	
userName	BTREE	No	No	userName	0	A	No	
				bookingID	0	A	No	
bookerNICNo	BTREE	No	No	bookingID	0	A	No	

receive_ticke

Table comments: This Transaction Table is store booking data

Column	Type	Null	Default	Comments
ticketNo	varchar(20)	No		
passengerName	varchar(20)	No		Passenger Name
seatNo	int(2)	No		Bus Seat Number
gender	varchar(1)	No		Passenger Gender
bookingID	varchar(20)	No		

seat

Table comments: This Master Table is store Bus Seat Number

Column	Type	Null	Default	Comments
seatNo	int(2)	No		Bus Seat Number

Indexes

Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
PRIMARY	BTREE	Yes	No	seatNo	49	A	No	

system_user

Table comments: This Master Table is store System User Data and Login Data

Column	Type	Null	Default	Comments
userName	varchar(10)	No		User Name for login to System
empolyeeNo	varchar(8)	No		Empoyee Number of System User
empolyeeName	varchar(20)	No		oyee Name of System User
empolyeeMNo	varchar(10)	No		oyee Mobile Number of System User
password	varchar(250)	Yes	*NULL*	Password for login to system
privilege	varchar(8)	No	Not User	

Appendix F - Codes for Login Page

The HTML code for design Login Interface

```
<html>
  <head>
  <body>
      <h1>Login Page</h1>
          <form action="http://localhost/SLTB/login/loginToSystem" method="post
          " id="log" class="has-validation-callback">
          <label>User Name</label>
          <input type="text" name="userName" data-validation="required"/>
          <br/>
          <label>Password</label>
          <input type="password" name="password" data-validation="required"/>
          <br/>
          <label/>
          <input type="submit" name="" id="" value="Login"/>
          </form>
      <div id="footer">
      </div>
  </body>
</html>
```

The CSS code for design Login Page

```
body {
        background:silver;
        font-family:'Times New Roman';
        font-size:14px;
}

form label {
```

```
        display:block;
        float:left;
        font-size:15px;
        margin-bottom:5px;
        margin-top:5px;
        width:100px;
}

form input, form select{
        font-family:'Times New Roman';
        font-size:15px;
}
```

The PHP code for Login to System

```
<?php

class Login_Model extends Model {

   public function __construct() {
      parent::__construct();
   }

   public function loginToSystem() {
      $sth = $this->db->prepare("SELECT userName, privilege FROM system_user
WHERE
            userName = :userName AND password = :password");
      $sth->execute(array(
         ':userName' => $_POST['userName'],
         ':password' => Hash::create('md5', $_POST['password'],
HASH_PASSWORD_KEY)
      ));
```

```
        $data = $sth->fetch();
        $count = $sth->rowCount();
        if ($count > 0) {
            Session::init();
            Session::set('privilege', $data['privilege']);
            Session::set('loggedIn', true);
            Session::set('userName', $data['userName']);
            if ($data['privilege'] == 'Booker')
                header('location: ../index');
            else
                header('location: ../systemUser');
        } else {
            header('location: ../login/index/User Name or Password is invalid .!');
        }
    }
}
?>
```

Appendix G - Codes for Display all Busses Page

The HTML and PHP code for all Buses Interface

```
<div class="main-button">
   <button class="btn"><a href="<?php echo URL; ?>bus"><img class="table-
   button"/></a></button>
   <button class="btn"><a href="<?php echo URL; ?>bus/create"><img class="add-
   button"/></a></button>
</div>

<div class="">
   <div id="bodyhead"><h1>All Buss</h1></div>
   <?php
   $url = explode('/', $_GET['url']);
   if (isset($url[2])) {
         echo '<P class="magNo"> ' . $url[2] . '</p>';
   }
   ?>
   <div id="tSize">
      <div class="demo_jui">
         <table cellpadding="0" cellspacing="0" border="0" class="display"
id="example">
            <thead>
               <tr>
                  <th>Bus No</th>
                  <th>Bus Model</th>
                  <th>Number Of Seat</th>
                  <th>Departure Time</th>
                  <th>Arrival Time</th>
                  <th></th>
                  <th></th>
               </tr>
```

```
            </thead>
            <tfoot>
               <tr>
                  <th>Bus No</th>
                  <th>Bus Model</th>
                  <th>Number Of Seat</th>
                  <th>Departure Time</th>
                  <th>Arrival Time</th>
                  <th></th>
                  <th></th>
               </tr>
            </tfoot>
            <tbody>
               <?php
               foreach ($this->searchAllBus as $key => $value) {
                  echo '<tr>';
                  echo '<td>' . $value['busNo'] . '</td>';
                  echo '<td>' . $value['busModel'] . '</td>';
                  echo '<td>' . $value['numberOfSeat'] . '</td>';
                  echo '<td>' . $value['departureTime'] . '</td>';
                  echo '<td>' . $value['arrivalTime'] . '</td>';
                  echo '<td><a href="' . URL . 'bus/addJourneytoBus/' . $value['busNo'] .
'">J. No</a></td>';
                  echo '<td>
                     <a href="' . URL . 'bus/updateFromTable/' . $value['busNo'] .
'"><img class="table-edit-button" alt="Update"/></a>
                     <a href="' . URL . 'bus/deleteBus/' . $value['busNo'] . '"><img
class="table-delete-button" alt="Delete"/></a>
                  </td>';
                  echo '</tr>';
               }
               ?>
            </tbody>
         </table>
```

```
        </div>
        <div class="spacer"></div>
    </div>
</div>
```

The PHP code for Search Bus Data

```
public function searchAllBus() {
        return $this->db->select('SELECT * FROM bus');
    }
```

Appendix H - Codes for Enter Bus Page

The HTML and PHP code for add new bus Interface

```
<div class="main-button">
   <button class="btn"><a href="<?php echo URL; ?>bus"><img class="table-
   button"/></a></button>
   <button class="btn"><a href="<?php echo URL; ?>bus/create"><img class="add-
   button"/></a></button>
</div>

<div class="main-form">
   <h1>Add New Bus</h1>
   <?php
   $url = explode('/', $_GET['url']);
   if (isset($url[2])) {
      if ($url[2] == 'Success')
         echo '<P class="magOk"> Data Add Successful .... ! </p>';
      if ($url[2] == 'Fail')
         echo '<P class="magNo"> Data Add Fail error is ' . $url[3] . '</p>';
   }
   ?>
   <form id="bus_create_form" action="<?php echo URL; ?>bus/createBus/"
   method="post">
      <label for="Bus_busNo" class="required">Bus No</label>
      <input size="10" maxlength="10" name="cBus_busNo" id="Bus_busNo"
      type="text"  data-validation="length" data-validation-length="max6"><br />

      <label for="Bus_busModel" class="required">Bus Model</label>
      <input size="15" maxlength="15" name="cBus_busModel" id="Bus_busModel"
      type="text" data-validation="required"><br />
      <label for="Bus_numberOfSeat" class="required">Number Of Seat</label>
```

```
        <input size="10" name="cBus_numberOfSeat" id="Bus_numberOfSeat"
        type="text" data-validation="number" data-validation-
        allowing="range[40;49]"><br />
        <label ></label>
        <input type="submit" name="addNewBus" id="addNewBus" value="Add
        Data">
    </form>
</div>
```

The PHP code for Insert Bus Data

```
class Bus_Model extends Model {
    public function __construct() {
        parent::__construct();
    }
    public function createBus($data) {
            $arrayData = array();
            $arrayData[0]['table'] = 'bus';
            $arrayData[0]['data'] = array(
            'busNo' => $data['cBus_busNo'],
            'busModel' => $data['cBus_busModel'],
                'numberOfSeat' => $data['cBus_numberOfSeat'],
            );
            $arrayData[1]['table'] = 'assign_bus ';
            $arrayData[1]['data'] = array(
                'userName' => Session::get('userName'),
                'busNo' => $data['cBus_busNo'],
                'date' => date("Y-m-d"),
                'sql' => 'I'
            );
            return $this->db->traInsert($arrayData);
        }
}
```

```
class Database extends PDO {
   public function __construct($DB_TYPE, $DB_HOST, $DB_NAME, $DB_USER,
   $DB_PASS) {
         parent::__construct($DB_TYPE . ':host=' . $DB_HOST . ';dbname=' .
         $DB_NAME, $DB_USER, $DB_PASS);
         $this->setAttribute(PDO::ATTR_ERRMODE,
         PDO::ERRMODE_EXCEPTION);
   }
   public function traInsert($arrayData) {
      $this->beginTransaction();
      try {
         foreach ($arrayData as $key => $value) {
            $table = $value['table'];
            $data = $value['data'];
            ksort($data);
            $fieldNames = implode('`, `', array_keys($data));
            $fieldValues = ':' . implode(', :', array_keys($data));
            $sth = $this->prepare("INSERT INTO $table (`$fieldNames`) VALUES
            ($fieldValues)");
            foreach ($data as $key => $value) {
               $sth->bindValue(":$key", $value);
            } $val = $sth->execute();
         }
         $this->commit();
         return $val;
      } catch (Exception $e) {
         $this->rollBack();
         return $e->getMessage();
      }
   }
}
```

Appendix I - Code for Update Bus Page

The HTML and PHP code for Update bus Interface

```
<div class="main-button">
   <button class="btn"><a href="<?php echo URL; ?>bus"><img class="table-
   button"/></a></button>
   <button class="btn"><a href="<?php echo URL; ?>bus/create"><img class="add-
   button"/></a></button>
</div>
<div class="main-form">
   <h1>Edit Bus</h1>
   <?php
   $url = explode('/', $_GET['url']);
   if (isset($url[2])) {
      if ($url[2] == 'Success')
         echo '<P class="magOk"> Data Edit Successful .... ! </p>';
      if ($url[2] == 'Fail')
         echo '<P class="magNo"> Data Edit Fail error is ' . $url[3] . '</p>';
   }
   ?>
      <form id="bus_update_form" action="<?php echo URL; ?>bus/updateBus/"
      method="post">
       <label for="Bus_busNo" class="required">Bus No</label>
       <input size="10" maxlength="10" name="uBus_busNo" value="<?php
       if (isset($this->bus[0]['busNo'])) {
       echo $this->bus[0]['busNo'];
        }
        ?>" id="uBus_busNo" type="text"  data-validation="required" ><br />

       <label for="Bus_busModel" class="required">Bus Model</label>
```

```
<input size="15" maxlength="15" name="uBus_busModel" value="<?php if
(isset($this-    >bus[0]['busModel'])) echo $this->bus[0]['busModel']; ?>"
id="uBus_busModel" type="text" data-validation="required"><br />
<label for="Bus_numberOfSeat" class="required">Number Of Seat</label>
<input size="10" name="uBus_numberOfSeat" value="<?php if (isset($this-
>bus[0]['numberOfSeat'])) echo $this->bus[0]['numberOfSeat']; ?>"
id="uBus_numberOfSeat" type="text" data-validation="number" data-validation-
allowing="range[40;49]"><br />
<label for="Bus_departureTime" class="required">Departure Time</label>
<input size="10" name="uBus_departureTime" value="<?php if (isset($this-
>bus[0]['departureTime'])) echo $this->bus[0]['departureTime']; ?>"
value="00:00 AM" id="uBus_departureTime" type="text" data-
validation="required"><br />
<label for="Bus_arrivalTime" class="required">Arrival Time</label>
<input  size="10"name="uBus_arrivalTime" value="<?php if (isset($this-
>bus[0]['arrivalTime'])) echo $this->bus[0]['arrivalTime']; ?>" value="12:00 PM"
id="uBus_arrivalTime" type="text" data-validation="required"><br />
<label ></label>
<input type="submit" name="updateNewBus" id="updateNewBus" value="Save
Data
  </form>
</div>
```

The PHP code for Update Bus Data

```
public function updateBus($data) {
    $postData_update = array(
        'busNo' => $data['uBus_busNo'],
        'busModel' => $data['uBus_busModel'],
        'numberOfSeat' => $data['uBus_numberOfSeat'],
    );
    $postData_insert = array(
        'userName' => Session::get('userName'),
```

```
            'busNo' => $data['uBus_busNo'],
            'date' => date("Y-m-d"),
            'sql' => 'U'
        );
        return $this->db->update_and_insert('bus', $postData_update, "`busNo` =
        '{$data['uBus_busNo']}'",'assign_bus', $postData_insert);
    }

class Database extends PDO {
    public function __construct($DB_TYPE, $DB_HOST, $DB_NAME, $DB_USER,
    $DB_PASS) {
            parent::__construct($DB_TYPE . ':host=' . $DB_HOST . ';dbname=' .
            $DB_NAME, $DB_USER, $DB_PASS);
            $this->setAttribute(PDO::ATTR_ERRMODE,
            PDO::ERRMODE_EXCEPTION);
    }
    public function traupdate($arraydata) {
        $this->beginTransaction();
        try {

            foreach ($arraydata as $key => $value) {
                $table = $value['table'];
                $data = $value['data'];
                $where = $value['where'];
                ksort($data);
                $fieldDetails = NULL;
                foreach ($data as $key => $value) {
                    $fieldDetails .= "`$key`=:$key,";
                }
                $fieldDetails = rtrim($fieldDetails, ',');
                $sth = $this->prepare("UPDATE $table SET $fieldDetails WHERE
                 $where");
                foreach ($data as $key => $value) {
                    $sth->bindValue(":$key", $value);
```

```
                }
                $val = $sth->execute();
            }
            $this->commit();
            return $val;
        } catch (Exception $e) {
            $this->rollBack();
            return $e->getMessage();
        }
    }
}
```

Appendix J - Codes for Changer Password

The HTML and PHP code for Changer Password Interface

```
<div class="main-form">
   <h1>Change  Password</h1>
   <?php
   $url = explode('/', $_GET['url']);
   if (isset($url[2])) {
      if ($url[2] == 'Success')
         echo '<P class="magOk"> Password Change Successful .... ! </p>';
      if ($url[2] == 'Fail')
         echo '<P class="magNo"> Password Change Fail error is ' . $url[3] . '</p>';
   }
   ?>
   <form id="update_password_form" action="<?php echo URL;
   ?>systemUser/updatePassword/" method="post">
      <label class="changer_p" for="currentPassword">Current password</label>
      <input size="15" maxlength="15" name="currentPassword"
      id="currentPassword" type="password" value="" data-validation="required"><br
      />
      <label for="newPassword" class="changer_p">New password</label>
      <input size="15" name="newPassword" id="newPassword" type="password"
      value="" data-validation="required"><br />
      <label for="confirmPassword" class="changer_p">Confirm new
      Password</label>
      <input size="15" name="confirmPassword" id="confirmPassword"
      type="password" value="" ddata-validation="required" ><br />
      <label class="changer_p"></label>
      <input type="submit" name="updatePassword" id="updatePassword"
      value="Save Data">
   </form>
</div>
```

The PHP code for Change Password

```
public function updatePassword($data) {
        $userName = Session::get('userName');
        $currentPassword = Hash::create('md5', $data['currentPassword'],
        HASH_PASSWORD_KEY);
        $newPassword = Hash::create('md5', $data['newPassword'],
        HASH_PASSWORD_KEY);
        $sth = $this->db->prepare("SELECT * FROM system_user WHERE
                userName = :userName AND password = :password");
        $sth->execute(array(
            ':userName' => $userName,
            ':password' => $currentPassword
        ));
        $count = $sth->rowCount();
        if ($count > 0) {
            $postData = array(
                'password' => $newPassword
            );
            return $this->db->update('system_user', $postData, "`userName` =
'{$userName}'");
        } else {
            return 'Enter Correct Current Password';
        }
}


    public function update($table, $data, $where) {
        $this->beginTransaction();
        try {
            ksort($data);
            $fieldDetails = NULL;
            foreach ($data as $key => $value) {
                $fieldDetails .= "`$key`=:$key,";
```

```
        }
        $fieldDetails = rtrim($fieldDetails, ',');
        $sth = $this->prepare("UPDATE $table SET $fieldDetails WHERE $where");
        foreach ($data as $key => $value) {
            $sth->bindValue(":$key", $value);
        } $val = $sth->execute();
        $this->commit();
        return $val;
    } catch (Exception $e) {
        $this->rollBack();
        return $e->getMessage();
    }
}
```

Appendix K - Codes for Report Page

The HTML and PHP code for View Report Interface

```
<div class="main">
   <h3>Booking Report</h3><br/>
   <form id="" target="_blank" action="<?php echo URL; ?>report/bookingData/"
       method="post">
      <label for="" class="repot_date_la">Date From</label>
      <input  size="10" name="date_from" id="" class="datepicker_repot" data-
       validation="required" value="<?php  echo date("Y-m-d");    ?>"><br/>
      <label for="" class="repot_date_la">Date To</label>
      <input  size="10" name="date_to" id="" class="datepicker_repot" data-
       validation="required" value="<?php  echo date("Y-m-d");    ?>"><br/>
      <label for="busNo" class="required">Bus Number</label>
      <select name="busNo" data-validation="required">
         <option value="AB">All Bus</option>
         <?php
         foreach ($this->searchAllBus as $key => $value) {
            echo '<option value="' . $value['busNo'] . '" > ' . $value['busNo'] .
       '</option>';
         }
         ?>
      </select><br/>
      <label for="journeyNo" class="required">Journey No</label>
      <select name="journeyNo" data-validation="required">
         <option value="AJ">All Journey</option>
         <?php
         foreach ($this->searchAllJourney as $key => $value) {
            echo '<option value="' . $value['journeyNo'] . '" > ' . $value['journeyNo'] .
       '</option>';
         }
         ?>
      </select><br/>
```

```
        <label ></label>
        <input type="submit" name="" id="" value="View Report">
    </form>
</div>

<div class="reportConducter1">
    <h3>Booker Info Report</h3><br/>
    <form id="bus_create_form" target="_blank" action="<?php echo URL;
        ?>report/bookerReport/" method="post">
        <label for="" class="repot_date_la">Journey Date</label>
        <input  size="10" name="journeyDate" id="" class="datepicker_repot" data-
        validation="required" value="<?php  echo date("Y-m-d");    ?>"><br/>
        <label for="busNo" class="required">Bus Number</label>
        <select name="busNo" data-validation="required">
            <option ></option>
            <?php
            foreach ($this->searchAllBus as $key => $value) {
                echo '<option value="' . $value['busNo'] . '" > ' . $value['busNo'] .
'</option>';
            }
            ?>
        </select><br/>
        <label ></label>
        <input type="submit" name="" id="" value="View Report">
    </form>
</div>

<div class="reportConducter2">
    <h3>Passenger Info Report</h3><br/>
    <form id="bus_create_form" target="_blank" action="<?php echo URL;
        ?>report/passengerReport/" method="post">
        <label for="" class="repot_date_la">Journey Date</label>
        <input  size="10" name="journeyDate" id="" class="datepicker_repot" data-
        validation="required" value="<?php  echo date("Y-m-d");    ?>"><br/>
```

```
        <label for="busNo" class="required">Bus Number</label>
        <select name="busNo" data-validation="required">
            <option ></option>
            <?php
            foreach ($this->searchAllBus as $key => $value) {
                echo '<option value="' . $value['busNo'] . '" > ' . $value['busNo'] .
        '</option>';
            }
            ?>
        </select><br/>
        <label ></label>
        <input type="submit" name="" id="" value="View Report">
    </form>
</div>
```

The PHP code for Search data

```
public function getPassengerData($journeyDate,$busNo) {
        return $this->db->select('SELECT
                                receive_ticke.seatNo,
                                receive_ticke.ticketNo,
                                receive_ticke.passengerName,
                                receive_ticke.gender,
                                receive_ticke.bookingID
                                FROM receive_ticke
                                INNER JOIN booking ON receive_ticke.bookingID =
         booking.bookingID
                                WHERE booking.busNo ="' . $busNo . '" AND booking.date
        ="' . $journeyDate . '" AND booking.payment_status ="Ok" ' );
    }

    public function getBookerrData($journeyDate,$busNo) {
```

```
        return $this->db->select('SELECT
                                booking.bookingID,
                                booking.bookerNICNo,
                                (SELECT bookerName FROM booker WHERE bookerNICNo
        = booking.bookerNICNo) AS bookerName,
                                (SELECT entryPoint FROM entry_point WHERE
        entryPointNo = booking.entryPointNo) AS entryPointNo,
                                booking.no_of_seat,
                                booking.ammount,
                                (SELECT bookerMNo FROM booker WHERE bookerNICNo
        = booking.bookerNICNo) AS bookerMNo
                                FROM booking
                                WHERE booking.busNo ='" . $busNo . "' AND booking.date
        ='" . $journeyDate . "' AND booking.payment_status ="Ok" ' );
    }
```

The PHP code for View pdf

```
    function view_report($main_tabale,$reportName,$size) {

        // Include the main TCPDF library (search for installation path).
        require_once('pdf/tcpdf_include.php');
        require_once('pdf/tcpdf.php');
        // create new PDF document
       $pdf = new TCPDF(PDF_PAGE_ORIENTATION, PDF_UNIT,
        PDF_PAGE_FORMAT, true, 'UTF-8', false);
        // set default header data
       $pdf->SetHeaderData(PDF_HEADER_LOGO,
        PDF_HEADER_LOGO_WIDTH, PDF_HEADER_TITLE . ",
        PDF_HEADER_STRING);
        // set header and footer fonts
       $pdf->setHeaderFont(Array(PDF_FONT_NAME_MAIN, ",
        PDF_FONT_SIZE_MAIN));
```

```
$pdf->setFooterFont(Array(PDF_FONT_NAME_DATA, '',
 PDF_FONT_SIZE_DATA));
 // set default monospaced font
$pdf->SetDefaultMonospacedFont(PDF_FONT_MONOSPACED);
 // set margins
$pdf->SetMargins(PDF_MARGIN_LEFT, PDF_MARGIN_TOP,
 PDF_MARGIN_RIGHT);
$pdf->SetHeaderMargin(PDF_MARGIN_HEADER);
$pdf->SetFooterMargin(PDF_MARGIN_FOOTER);
   // set auto page breaks
$pdf->SetAutoPageBreak(TRUE, PDF_MARGIN_BOTTOM);
   // set image scale factor
$pdf->setImageScale(PDF_IMAGE_SCALE_RATIO);
   // set some language-dependent strings (optional)
                    if (@file_exists(dirname(__FILE__) . '/lang/eng.php')) {
                            require_once(dirname(__FILE__) . '/lang/eng.php');
                            $pdf->setLanguageArray($l);
                    }
   // set font
$pdf->SetFont('times', '', 10);
$pdf->AddPage($size, 'A4');
   // output the HTML content
$pdf->writeHTML($main_tabale, true, false, true, false, '');
$pdf->Output($reportName.'.pdf', 'I');


   }
```

Appendix L - Codes for Available Bus page

The HTML and PHP code for Available Buses Interface

```
<div>
   <div id="bodyhead"><h1>Available Buss</h1></div>
   <form id="" action="<?php echo URL; ?>booker/booking/" method="post">
      <div id="tSize">
         <div class="demo_jui">
            <?php if(($this->bookDate) != null){?>
            <input type="hidden" name="book_date" id="book_date" value="<?php
            echo $this->bookDate ?>"/>
            <input type="hidden" name="book_journeyNo" id="book_journeyNo"
            value="<?php if(($this->journeyNo) != null){echo $this-
            >journeyNo[0]['journeyNo'];} ?>"/>
            <input type="hidden" name="book_busNo" id="book_busNo" value=""/>
            <input type="hidden" name="book_numberOfSeat"
            id="book_numberOfSeat" value=""/>
            <input type="hidden" name="book_price" id="book_price" value=""/>
            <?php }?>
            <table cellpadding="0" cellspacing="0" border="0" class="display"
            id="exampleBooker">
               <thead>
                  <tr>
                     <th>Bus No</th>
                     <th>No. of Seat</th>
                     <th>Route No</th>
                     <th>Dep. Time</th>
                     <th>Arr. Time</th>
                     <th>Entry Point</th>
                     <th>Price</th>
                     <th></th>
                  </tr>
               </thead>
```

```
                <tbody>
                    <?php
                    if (isset($this->availablelBus)) {
                        foreach ($this->availablelBus as $key => $value1) {
                            echo '<tr class="busData">';
                            echo '<td>' . $value1['busNo'] . '</td>';
                            echo '<td>' . $value1['numberOfSeat'] . '</td>';
                            echo '<td>' . $value1['routeNo'] . '</td>';
                            echo '<td>' . $value1['departureTime'] . '</td>';
                            echo '<td>' . $value1['arrivalTime'] . '</td>';
                            echo '<td><select>';
                            echo '<option>Entry Point</option>';
                            foreach ($value1['entry_Point'] as $key => $value2) {
                                echo '<option>' . $value2['entryPoint'] . '</option>';
                            }
                            echo '</select></td>';
                            echo '<td>' . $value1['price'] . '</td>';
                            echo '<td><input type="submit" name="bookNow" value="Book
                            Now"></td>';
                            echo '</tr>';
                        }
                    }
                    ?>
                </tbody>
            </table>
        </div>
        <div class="spacer"></div>
    </div>
</form>
</div>
```

The PHP code for Available Buses

```
public function searchAvailablelBus($from, $to) {
    $availablelBus = $this->db->select('SELECT
            bus.busNo,
            bus.departureTime,
            bus.arrivalTime,
            bus.numberOfSeat,
            journey.routeNo,
            journey.price,
            journey.journeyNo
            FROM journey
            INNER JOIN journey_for_bus ON journey.journeyNo =
            journey_for_bus.journeyNo
            INNER JOIN bus ON journey_for_bus.busNo = bus.busNo
            WHERE journey.journeyFrom =:journeyFrom AND journey.journeyTo
            =:journeyTo ORDER BY bus.departureTime ASC; ',
            array(':journeyFrom' => $from,':journeyTo' => $to));

            $arrayData = array();
            $arrayDataEntry = array();
            $i=0;

            if (isset($availablelBus)) {
            foreach ($availablelBus as $key => $value) {

               $arrayData[$i]['busNo'] = $value['busNo'];
               $arrayData[$i]['routeNo'] = $value['routeNo'];
               $arrayData[$i]['departureTime'] = $value['departureTime'];
               $arrayData[$i]['arrivalTime'] = $value['arrivalTime'];
               $arrayData[$i]['numberOfSeat'] = $value['numberOfSeat'];
               $arrayData[$i]['price'] = $value['price'];
               $entryPoint = $this->db->select('SELECT
                  entry_point.entryPoint
```

```
                FROM entry_point
                JOIN entrypoint_for_journey ON entry_point.entryPointNo =
            entrypoint_for_journey.entryPointNo
                WHERE journeyNo = "'.$value['journeyNo'].'" ');
            $j=0;
            if (isset($entryPoint)) {
                foreach ($entryPoint as $key => $value2) {
                    $arrayDataEntry[$j]['entryPoint']=$value2['entryPoint'];
                    $j++;
                }
            }
            $arrayData[$i]['entry_Point']=$arrayDataEntry;
           $i++;
        }
    return $arrayData;
  }
}
```

Appendix M - Codes for Available Seat

The HTML and PHP code for Available Seat Interface

```
<div id="area">
   <div id="print" style="text-align:center; color: #d14"></div>
   <div id="holder">
      <ul id="place">
        <?php
         Session::init();
         $this->busBooker = new Booker_Model();
         $busNo = $this->seatDara['busNo'];
         $noOfSeat = $this->seatDara['noOfSeat'];
         $bus_book_date = $this->seatDara['bus_book_date'];
         $journeyNo = $this->seatDara['journeyNo'];
         $bookSeat = array();
         $bookStatus = array();
         $selecting_s = array();

         if (isset($this->seatDara['seatNo']))
            $selecting_s = $this->seatDara['seatNo'];

         if (isset($this->seatDara['session']))
            if (($this->seatDara['session']) == 1)
               if ((Session::get('sessionforSelectin_s')) == true)
                  $selecting_s = Session::get('sessionforSelectin_s');
         /* call  searchAvailableSeat funtion get Available Seat bookStatus */
         $seat = $this->busBooker->xhrSearchAvailableSeat($busNo, $bus_book_date,
        $journeyNo);

         foreach ($seat as $key => $value) {
            $bookSeat[$key] = $value['seatNo'];
            $bookStatus[$key] = $value['status'];
         }
```

```
/* call  displayBusSeat funtion display Bus Seat on web page */
displayBusSeat($noOfSeat, $bookSeat, $bookStatus, $selecting_s);

function displayBusSeat($noOfSeat, $bookSeat, $bookStatus, $selecting_s) {
    $noOfSeat = $noOfSeat;
    $bookSeat = $bookSeat;
    $bookStatus = $bookStatus;
    $selecting_s = $selecting_s;
    $rows = 5;
    $cols = 13;
    $rows2 = 5;
    $cols2 = 10;
    $rowCssPrefix = 'row-';
    $colCssPrefix = 'col-';
    $seatWidth = 40;
    $seatHeight = 40;
    $seatCss = 'seat';
    $hidingSeatCss = 'hidingSeats';
    $selectedSeatCss = 'selectedSeat';
    $selectingSeatCss = 'selectingSeat';
    $pendingSeatCss = 'pendingSeat';
    $str = array();
    $seatNo;
    $className = '';
    $p;
    $x = 0;
                                if ($noOfSeat == 49) {
        for ($i = 0; $i < $cols; $i++) {
            for ($j = 0; $j < $rows; $j++) {
                $p = ($i * $rows + $j + 1);
                if ($p == 3 || $p == 8 || $p == 13 || $p == 18 || $p == 23 || $p == 28 ||
                $p == 33 || $p == 38 || $p == 43 || $p == 48 || $p == 53 || $p == 58 ||
                $p == 4 || $p == 5 || $p == 39 || $p == 40) {
```

```
    $seatNo = null;
    $className = $seatCss . ' ' . $hidingSeatCss . ' ' . $rowCssPrefix .
$i . ' ' . $colCssPrefix . $j;
 } else {
    $seatNo = (++$x);
    if ($bookSeat != null) {
       foreach ($bookSeat as $key => $value) {
          if ($x == $value) {
             if ($bookStatus[$key] == 'B')
                $className = $seatCss . ' ' . $rowCssPrefix . $i . ' ' .
                $colCssPrefix . $j . ' ' . $selectedSeatCss;
             else if ($bookStatus[$key] == 'P')
                $className = $seatCss . ' ' . $rowCssPrefix . $i . ' ' .
                $colCssPrefix . $j . ' ' . $pendingSeatCss;
             if ($selecting_s != null)
                foreach ($selecting_s as $key2 => $value2) {
                   if ($value2 == $x) {
                      $className = $className . ' ' . $selectingSeatCss;
                      break;
                   }
                }
             break;
          } else {
             $className = $seatCss . ' ' . $rowCssPrefix . $i . ' ' .
             $colCssPrefix . $j;
          }
       }
    }else{
       $className = $seatCss . ' ' . $rowCssPrefix . $i . ' ' .
        $colCssPrefix . $j;
    }
 }
```

```
                    echo '<li class="' . $className . '" seatno="' . $seatNo . '"
                    style="top:' . ($j * $seatHeight) . 'px; left:' . ($i * $seatWidth) .
                    'px"><a title="' . $seatNo . '"> ' . $seatNo . ' </a></li>';
                }
            }
        }
        }
        ?>
    </ul>
  </div>
</div>
```

The Ajax and jQuery code for Available Seat

```
$(function(){
  setSeat();

  function setSeat() { //alert(1);
    var session = 1;
    var busNo=document.getElementById("seat_book_busNo").value;
    var bus_book_date= document.getElementById("seat_book_date").value;
    var journeyNo= document.getElementById("seat_book_journeyNo").value;
    var noOfSeat= document.getElementById("seat_book_numberOfSeat").value;
    $.ajax({
      type:'POST',
      url:'/SLTB/booker/viewBusSeat',
      data:{
        busNo:busNo,
        noOfSeat:noOfSeat,
        journeyNo:journeyNo,
        bus_book_date:bus_book_date,
        session:session
      },
      success:function(o){ //alert(1);
```

```
            $('#viweSeat').html(o);
            startRefresh();
        }
    });
}

function getselectingSeat(handleData) {
    var seatNo =[];
    $.each($('#place li.selectingSeat '), function (index, value) {
        items = $(this).attr('seatNo');
        seatNo.push(items);
    });
    handleData(seatNo);
}

function startRefresh() {
    getselectingSeat(function(seatNo){
        var busNo=document.getElementById("seat_book_busNo").value;
        var bus_book_date= document.getElementById("seat_book_date").value;
        var journeyNo= document.getElementById("seat_book_journeyNo").value;
        var noOfSeat= document.getElementById("seat_book_numberOfSeat").value;
        $.ajax({
            type:'POST',
            url:'/SLTB/booker/viewBusSeat',
            data:{
                busNo:busNo,
                noOfSeat:noOfSeat,
                journeyNo:journeyNo,
                bus_book_date:bus_book_date,
                seatNo:seatNo
            },
            success:function(o){ //alert(1);
                $('#viweSeat').html(o);
            }
```

```
            });
        });
    setTimeout(startRefresh,1000);
    }
});

$(function(){
    cliarPrint();
    sessionforSelectin_s();
    function cliarPrint() {
        $('#print').html('');
    }
    function sessionforSelectin_s() {
      $.ajax({
            type:'POST',
            url:'/SLTB/booker/xhrgetSessionforSelectin_s',
            data:{
            },
            dataType:'json',
            beforeSend:function(o){
            },
            success:function(o){
                document.getElementById("selecting_seate_for_booker").value = o;
            },
            error : function(XMLHttpRequest, textStatus, errorThrown){
            }
        });

         $.ajax({
            type:'POST',
            url:'/SLTB/booker/xhrgetSessionforSelectin_s_tot',
            data:{
            },
            dataType:'json',
```

```
        beforeSend:function(o){
        },
        success:function(o){
           if(o != null)
           document.getElementById("total_price_for_selecting_seate").value = o;
        },
        error : function(XMLHttpRequest, textStatus, errorThrown){
        }
     });
  }
** ……function for session Time Out …….…… *
  sessionTimeOut();
  function sessionTimeOut() {
     timer = setTimeout(sessionTimeOut,1000);
     $.ajax({
        type:'POST',
        url:'/SLTB/booker/xhrSessionTimeOut',
        dataType:'json',
        beforeSend:function(o){
        },
        success:function(o){
           if(o.length > 0){
              clearAllSeateformDB(o);
              $('.timeOut').html('');
              clearTimeout(timer);
              $("#passenger_info").remove();
              $("#booker_info").remove();
              document.getElementById("total_price_for_selecting_seate").value = 0;
              document.getElementById("selecting_seate_for_booker").value = "";
              $.each($('#place li.selectingSeat '), function (index, value) {
                 $(this).removeClass('selectingSeat');
              });
           }else{
              $.each($('#place li.selectingSeat '), function (index, value) {
```

```
                if(($(this).hasClass('selectingSeat')))
                $('.timeOut').html('Your Reservation Expires in ['+parseInt(o)+'] (s)');
              });
            }
        },
        error : function(XMLHttpRequest, textStatus, errorThrown){
            clearTimeout(timer);
        }
    });
}
**……click on the seat ……*
$(".seat").live('click',function(){
    if ($(this).hasClass('selectedSeat')){
        $('#print').html('This seat is already reserved ...');
    }

    else if ($(this).hasClass('selectingSeat') && $(this).hasClass('pendingSeat')){
        var allSeatNo = [], selectArry=[], i=0;
        allSeatNo.push($(this).attr('seatNo'));
        clearAllSeateformDB(allSeatNo);
        setSubtractionSessionforSelectin_s(allSeatNo)
        $(this).removeClass('selectingSeat');
        $(this).removeClass('pendingSeat');
        $.each($('#place li.selectingSeat '), function (index, value) {
            selectArry[i++]=i;
        });
        if(selectArry.length==0){
            $('.timeOut').html('');
            clearTimeout(timer);
        }
    }
    else if ($(this).hasClass('pendingSeat')){
        if (!($(this).hasClass('selectingSeat')))
            $('#print').html('This seat is hold other ...');
```

```
    }
    else if(!($(this).hasClass('selectingSeat'))&& !($(this).hasClass('pendingSeat'))){
        $(this).toggleClass('pendingSeat');
        $(this).toggleClass('selectingSeat');
        var selectArry_ = [],j=0;
        $.each($('#place li.selectingSeat '), function (index, value) {
            selectArry_[j++]=j;
        });
        if(selectArry_.length==1){
            setSessionforTime();
        }
        insertSeattoTable($(this).attr('seatNo'));
        setIncrementSessionforSelectin_s($(this).attr('seatNo'));
    }
    else{
        $('#print').html('Pls wait ...');
    }
});
```

** ……. function set Session for all Increment Selecting Seat …….. *

```
function setIncrementSessionforSelectin_s(seatNo) {
    var price = document.getElementById("seat_book_price").value;
    var tot_price =
   document.getElementById("total_price_for_selecting_seate").value;
    tot_price = parseInt(tot_price) + parseInt(price);
    $.ajax({
        type:'POST',
        url:'/SLTB/booker/xhrIncrementSessionforSelectin_s',
        data:{
            seatNo:seatNo,
            tot_price:tot_price
        },
        dataType:'json',
```

```
            beforeSend:function(o){
            },
            success:function(o){
               document.getElementById("selecting_seate_for_booker").value = o;
               document.getElementById("total_price_for_selecting_seate").value =
             tot_price;
            },
            error : function(XMLHttpRequest, textStatus, errorThrown){
            }
          });
      }

      ** …… function set Session for all Subtraction Selecting Seat ……*
          function setSubtractionSessionforSelectin_s(seatNo) {
          var price = document.getElementById("seat_book_price").value;
          var tot_price =
document.getElementById("total_price_for_selecting_seate").value;
          tot_price = parseInt(tot_price) - parseInt(price);
            $.ajax({
            type:'POST',
            url:'/SLTB/booker/xhrSubtractionSessionforSelectin_s',
            data:{
               seatNo:seatNo,
               tot_price:tot_price
            },
            dataType:'json',
            beforeSend:function(o){
            },
            success:function(o){
               document.getElementById("selecting_seate_for_booker").value = o;
               document.getElementById("total_price_for_selecting_seate").value =
tot_price;
            },
            error : function(XMLHttpRequest, textStatus, errorThrown){
```

```
        }
    });
}
 ** …….. function insert Seat to Database ……..…… *
function insertSeattoTable(seatNo) {
    var busNo=document.getElementById("seat_book_busNo").value;
    var bus_book_date= document.getElementById("seat_book_date").value;
    var journeyNo= document.getElementById("seat_book_journeyNo").value;
    var status = 'P';
    $.ajax({
        type:'POST',
        url:'/SLTB/booker/xhrInsertBookingSeat',
        data:{
            seatNo:seatNo,
            busNo:busNo,
            journeyNo:journeyNo,
            bus_book_date:bus_book_date,
            status:status
        },
        dataType:'json',
        beforeSend:function(o){
        },
        success:function(o){
            var seatNo =[];
            $.each($('#place li.selectingSeat '), function (index, value) {
                items = $(this).attr('seatNo');
                seatNo.push(items);
            });
        //afterInsertSeat(seatNo);
        },
        error : function(XMLHttpRequest, textStatus, errorThrown){
        }
    });
}
```

```
 ** …….. function After Insert Seat view new Bus seat ……..*
function afterInsertSeat(seatNo) {
    var seatNo = seatNo;
    var busNo=document.getElementById("seat_book_busNo").value;
    var bus_book_date= document.getElementById("seat_book_date").value;
    var journeyNo= document.getElementById("seat_book_journeyNo").value;
    var noOfSeat= document.getElementById("seat_book_numberOfSeat").value;
    $.ajax({
        type:'POST',
        url:'/SLTB/booker/viewBusSeat',
        data:{
            busNo:busNo,
            noOfSeat:noOfSeat,
            journeyNo:journeyNo,
            bus_book_date:bus_book_date,
            seatNo:seatNo
        },
        success:function(o){ //alert(1);
            $('#viweSeat').html(o);
        }
    });
}

** ……function set Session for Time  …….*

function setSessionforTime() {
    $.ajax({
        type:'POST',
        url:'/SLTB/booker/xhrsetSession',
        beforeSend:function(o){
        },
        success:function(o){
            sessionTimeOut();
        },
```

```
        error : function(XMLHttpRequest, textStatus, errorThrown){
        }
    });
}
```

** ……function clear All Setae form DB ……*

```
function clearAllSeateformDB(allSeatNo) {
    var seatNo = allSeatNo;
    var status = 'P';
    var busNo=document.getElementById("seat_book_busNo").value;
    var bus_book_date= document.getElementById("seat_book_date").value;
    var journeyNo= document.getElementById("seat_book_journeyNo").value;
    $.ajax({
        type:'POST',
        url:'/SLTB/booker/xhrClearAllSeate',
        data:{
            seatNo:seatNo,
            busNo:busNo,
            journeyNo:journeyNo,
            bus_book_date:bus_book_date,
            status:status
        },
        dataType:'json',
        beforeSend:function(o){
        },
        success:function(o){
          },
        error : function(XMLHttpRequest, textStatus, errorThrown){
        }
    });
}
```

Appendix N - Codes for Print Ticket

The HTML and PHP code for Print Ticket Interface

```
<?php
echo '<h1>Print Tikect</h1>';

echo '<div class="printBookinName"><h3>Booking Tikect</h3></div>';
echo '<div id="booking_ticket_area_main">';
echo '<link href="http://localhost/SLTB/public/css/booker/ticket.css"
rel="stylesheet"></link>';
echo '<div id="booking_ticket_area">';
if(isset($this->bookingTicket)){
   foreach ($this->bookingTicket as $key => $value) {
   echo '<label class="b_ticket_la">Booking ID  </label><label class="">:
   '.$value['bookingID'].'</label><br/>';
   echo '<label class="b_ticket_la">Booker NIC  </label><label class="">:
   '.$value['bookerNICNo'].'</label><br/>';
   echo '<label class="b_ticket_la">Bus No  </label><label class="">:
   '.$value['busNo'].'</label><br/>';
   echo '<label class="b_ticket_la">Route No  </label><label class="">:
   '.$value['routeNo'].'</label><br/>';
   echo '<label class="b_ticket_la">Journey  </label>: <label class="">From -
   '.$value['journeyFrom'].'</label><br/>';
   echo '<label class="b_ticket_la"> .</label><label class=""> To -
   '.$value['journeyTo'].'</label><br/>';
   echo '<label class="b_ticket_la">Entry Point  </label><label class="">:
   '.$value['entryPoint'].'</label><br/>';
   echo '<label class="b_ticket_la">Number Of Seat  </label><label class="">:
   '.$value['no_of_seat'].'</label><br/>';
   echo '<label class="b_ticket_la">Ammount  </label><label class="">:
   '.$value['ammount'].'</label><br/>';
   echo '<label class="b_ticket_la">Date  </label><label class="">:
   '.$value['date'].'</label><br/><br/>';
```

```
    }
}
echo '</div>';
echo '</div>';
echo '<div class="printBookinbtn"><input type="button" name=""
id="printBookinbtn" value="Print"></div>';
echo '<div id="bus_ticket_area_main">';
echo '<h3>Bus Tikect</h3>';
echo '<div id="bus_ticket_sub_area">';
echo '<link href="http://localhost/SLTB/public/css/booker/ticket.css"
rel="stylesheet"></link>';
if(isset($this->busTicket)){
    foreach ($this->busTicket as $key => $value) {
    echo '<div id="bus_ticket_area">';
    echo '<label class="b_ticket_la">Ticket No : </label><label
    class="">'.$value['ticketNo'].'</label><br/>';
    echo '<label class="b_ticket_la"></label><label class="">From -
    '.$value['journeyFrom'] .'</label>';
    echo '<label class="b_ticket_la"></label><label class=""> To -
    '.$value['journeyTo'].'</label><br/>';
    echo '<label class="b_ticket_la">Seat No : </label><label
    class="">'.$value['seatNo'].'</label>';
    echo '<label class="">' , ' Gender : </label><label
    class="">'.$value['gender'].'</label><br/>';
    echo '<label class="b_ticket_la">Date : </label><label
    class="">'.$value['date'].'</label><br/>';
    echo '<label class="b_ticket_la">Booking ID : </label><label
    class="">'.$value['bookingID'].'</label><br/><br/>';
    echo '</div>';
    }
}
echo '</div>';
echo '</div>';
```

```
echo '<div class="printbusTicketsbtn"><label></label><input type="button" name=""
id="printbusTicketsbtn" value="Print"></div>';
?>
```

The PHP code for Search Ticket Data

```
public function searchbookingTicket($val) {
    return $this->db->select('SELECT
        booking.bookingID,
        booking.bookerNICNo,
        booking.busNo,
        booking.ammount,
        booking.no_of_seat,
        booking.date,
        booking.payment_status,
        journey.routeNo,
        journey.journeyFrom,
        journey.journeyTo,
        (SELECT entryPoint FROM entry_point WHERE entryPointNo =
        booking.entryPointNo) AS entryPoint
        FROM booking
        JOIN journey ON booking.journeyNo = journey.journeyNo
        WHERE booking.bookingID ="' . $val . '");

}

public function searchBusTicket($val) {
    return $this->db->select('SELECT
        receive_ticke.ticketNo,
        receive_ticke.bookingID,
        receive_ticke.seatNo,
        receive_ticke.gender,
```

```
            (SELECT journeyFrom FROM journey WHERE journeyNo =
            booking.journeyNo) AS journeyFrom,
            (SELECT journeyTo FROM journey WHERE journeyNo =
            booking.journeyNo) AS journeyTo,
            booking.date
            FROM receive_ticke
            JOIN booking ON receive_ticke.bookingID = booking.bookingID
            WHERE receive_ticke.bookingID ='" . $val . "");

    }
```

The jQuery code for Print Ticket

```
$('#printbusTicketsbtn').live('click',function(){
        var prtContent = document.getElementById("bus_ticket_sub_area");
        var WinPrint = window.open('', '',
       'letf=0,top=0,width=500,height=500,toolbar=0,scrollbars=0,status=0');
        WinPrint.document.write(prtContent.innerHTML);
        WinPrint.document.close();
        WinPrint.focus();
        WinPrint.print();
        WinPrint.close();
});
```

Appendix O - Codes for Android Login Interface

The XML code for design Main Interface

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/container"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context="com.sltbtrackingsystem.MainActivity"
   tools:ignore="MergeRootFrame" >

   <TextView
      android:id="@+id/textView1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_alignParentTop="true"
      android:layout_centerHorizontal="true"
      android:layout_marginTop="22dp"
      android:text="@string/bus_tracking_system"
      android:textAppearance="?android:attr/textAppearanceLarge" />

   <EditText
      android:id="@+id/editText1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_alignRight="@+id/button1"
      android:layout_alignTop="@+id/textView2"
      android:layout_marginLeft="25dp"
      android:layout_toRightOf="@+id/textView2"
      android:ems="10"
      android:inputType="text" >
      <requestFocus />
   </EditText>
```

```
<TextView
   android:id="@+id/textView2"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_alignParentLeft="true"
   android:layout_below="@+id/textView1"
   android:layout_marginLeft="20dp"
   android:layout_marginTop="70dp"
   android:text="@string/user_name"
   android:textAppearance="?android:attr/textAppearanceLarge" />

<TextView
   android:id="@+id/textView3"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_alignLeft="@+id/textView2"
   android:layout_below="@+id/editText1"
   android:layout_marginTop="24dp"
   android:text="@string/password"
   android:textAppearance="?android:attr/textAppearanceLarge" />

<EditText
   android:id="@+id/editText2"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_alignBaseline="@+id/textView3"
   android:layout_alignBottom="@+id/textView3"
   android:layout_alignLeft="@+id/editText1"
   android:layout_alignRight="@+id/button1"
   android:ems="10"
   android:inputType="textPassword" >

</EditText>
```

```
    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_below="@+id/editText2"
        android:layout_marginRight="15dp"
        android:layout_marginTop="25dp"
        android:text="@string/login" />

    <ProgressBar
        android:id="@+id/progressBar1"
        style="?android:attr/progressBarStyleLarge"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/textView3"
        android:layout_toRightOf="@+id/textView3"/>

</RelativeLayout>
```

The Java code for Enter Tracking Interface

```
package com.sltbtrackingsystem;

import java.util.ArrayList;
import java.util.List;
import com.parse.FindCallback;
import com.parse.Parse;
import com.parse.ParseException;
import com.parse.ParseObject;
import com.parse.ParseQuery;
import android.support.v7.app.ActionBarActivity;
```

```
import android.support.v7.app.ActionBar;
import android.support.v4.app.Fragment;
import android.content.Intent;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.view.View.OnClickListener;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;
import android.os.Build;
public class MainActivity extends ActionBarActivity {

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

                if (savedInstanceState == null) {
                        getSupportFragmentManager().beginTransaction()
                                        .add(R.id.container, new
         PlaceholderFragment()).commit();
        }

         Parse.initialize(this,"w6sOkx6IXZPloTIkWcdbfX3RxKrrPJZmWPIGHeYk",
         "3avqsVGqNOLwIQQMst9PMxlIArOfwzkYB6Ls9t1u");
```

```
ParseQuery<ParseObject> query = ParseQuery.getQuery("Tracking_Data
query.findInBackground(new FindCallback<ParseObject>()
@Override
        public void done(List<ParseObject> objs, ParseException e) {
                ArrayList<String> items=new ArrayList<String>();
                 if (e == null) {
                        for(ParseObject obj:objs){
                        items.add(obj.getString("bus_number"));
                        }
                } else {
                Toast.makeText(getBaseContext(),e.getMessage() ,
                Toast.LENGTH_LONG).show();
                }
                        Spinner spinnertech = (Spinner)
findViewById(R.id.planets_spinner);
                        ArrayAdapter<String> adapter = new
ArrayAdapter<String>(getBaseContext(),android.R.layout.simple_spinner_ite
m,items);
                        spinnertech.setAdapter(adapter);
                }
         });

        onSubmit();
}

private void onSubmit() {
        Button btnSubmit = (Button) findViewById(R.id.button1);
        final Spinner spinnertechObj = (Spinner)
        findViewById(R.id.planets_spinner);
        btnSubmit.setOnClickListener(new OnClickListener() {

                @Override
                public void onClick(View arg0) {
```

```
                    Intent intent = new
                    Intent(MainActivity.this,TrackingActivity.class);
                                        intent.putExtra("busNumber",
                    (String)spinnertechObj.getSelectedItem());
                    MainActivity.this.startActivity(intent);
                }
            });
        }

}
```

Appendix P - Codes for Tracking Interface

```
<?xml version="1.0" encoding="utf-8"?>
  <ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
      android:id="@+id/scrollView1"
      android:layout_width="fill_parent"
      android:layout_height="fill_parent"
      android:layout_alignParentBottom="true"
      android:layout_alignParentLeft="true"
      android:layout_alignParentTop="true" >

    <LinearLayout
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:orientation="vertical" >
        <TextView
            android:id="@+id/textView5"
            android:layout_width="300dp"
            android:layout_height="wrap_content"
            android:layout_marginTop="20dp"
            android:layout_marginLeft="10dp"
            android:layout_marginRight="10dp"
            android:gravity="center"
            android:text="@string/tracking_system"
            android:textAppearance="?android:attr/textAppearanceLarge"/>
          <TextView
            android:id="@+id/textView1"
            android:layout_width="wrap_content"
            android:layout_height="50dp"
            android:layout_marginTop="20dp"
            android:layout_marginLeft="10dp"
            android:layout_marginRight="10dp"
            android:gravity="center"
```

```
    android:text="@string/bus_number"
    android:textAppearance="?android:attr/textAppearanceLarge" />
<TextView
    android:id="@+id/textView3"
    android:layout_width="wrap_content"
    android:layout_height="50dp"
    android:layout_marginTop="-50dp"
    android:layout_marginLeft="190dp"
    android:layout_marginRight="10dp"
    android:gravity="center"
    android:text="@string/bno"
    android:textAppearance="?android:attr/textAppearanceLarge" />
<TextView
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="50dp"
    android:layout_marginTop="0dp"
    android:layout_marginLeft="10dp"
    android:layout_marginRight="10dp"
    android:gravity="center"
    android:text="@string/journey_number"
    android:textAppearance="?android:attr/textAppearanceLarge" />

<TextView
    android:id="@+id/textView6"
    android:layout_width="wrap_content"
    android:layout_height="50dp"
    android:layout_marginTop="-50dp"
    android:layout_marginLeft="190dp"
    android:layout_marginRight="10dp"
    android:gravity="center"
    android:text="@string/jno"
    android:textAppearance="?android:attr/textAppearanceLarge" />
<Button
```

```
    android:id="@+id/button2"
    android:layout_width="120dp"
    android:layout_height="50dp"
    android:layout_marginTop="20dp"
    android:layout_marginLeft="10dp"
    android:layout_marginRight="10dp"
    android:gravity="center"
    android:text="@string/runing" />
<Button
        android:id="@+id/button6"
        android:layout_width="120dp"
        android:layout_height="50dp"
        android:layout_marginLeft="190dp"
        android:layout_marginRight="10dp"
        android:layout_marginTop="-50dp"
        android:gravity="center"
        android:text="@string/waiting" />

<Button
        android:id="@+id/button4"
        android:layout_width="120dp"
        android:layout_height="50dp"
        android:layout_marginTop="20dp"
        android:layout_marginLeft="10dp"
        android:layout_marginRight="10dp"
        android:gravity="center"
        android:text="@string/breakdown" />

<Button
    android:id="@+id/button5"
    android:layout_width="120dp"
    android:layout_height="50dp"
    android:layout_marginLeft="190dp"
    android:layout_marginRight="10dp"
```

```
    android:layout_marginTop="-50dp"
    android:gravity="center"
    android:text="@string/stop" />
<ProgressBar
    android:id="@+id/progressBar2"
    style="?android:attr/progressBarStyleLarge"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="125dp"
    android:layout_marginRight="10dp"
    android:layout_marginTop="0dp"
    android:layout_marginBottom="0dp"
    android:gravity="center"/>

<Button
    android:id="@+id/button1"
    android:layout_width="120dp"
    android:layout_height="50dp"
    android:layout_marginLeft="190dp"
    android:layout_marginRight="10dp"
    android:layout_marginTop="20dp"
    android:layout_marginBottom="50dp"
    android:gravity="center"
    android:text="@string/exit" />

 <TextView
    android:id="@+id/textView4"
    android:layout_width="300dp"
    android:layout_height="50dp"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="10dp"
    android:layout_marginRight="10dp"
    android:gravity="center"
    android:text="@string/booked_seat"
```

```
            android:textAppearance="?android:attr/textAppearanceLarge" />

        <DatePicker
            android:id="@+id/datePicker1"
            android:layout_width="300dp"
            android:layout_height="120dp"
            android:layout_marginBottom="10dp"
            android:layout_marginLeft="10dp"
            android:layout_marginRight="10dp"
            android:layout_marginTop="10dp"
            android:gravity="center" />

        <Button
            android:id="@+id/button3"
            android:layout_width="120dp"
            android:layout_height="50dp"
            android:layout_marginLeft="190dp"
            android:layout_marginRight="10dp"
            android:layout_marginTop="0dp"
            android:layout_marginBottom="50dp"
            android:gravity="center"
            android:text="@string/view_seat" />

    </LinearLayout>
</ScrollView>
```

The Java code for Track the location

```
package com.sltbtrackingsystem;
import java.util.List;
import com.parse.FindCallback;
import com.parse.Parse;
import com.parse.ParseException;
```

```
import com.parse.ParseObject;
import com.parse.ParseQuery;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

public class TrackingActivity extends Activity implements LocationListener{

        private String busNumber = null;
        LocationListener locationListener;
        LocationManager locationManager;

        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.tracking);

                Intent intent = getIntent();
                busNumber = intent.getStringExtra("busNumber");
                 TextView txtBusNo = (TextView)
                 findViewById(R.id.textView2);
                txtBusNo.setText(busNumber);

                 locationManager = (LocationManager)
                 getSystemService(Context.LOCATION_SERVICE
```

```
        locationManager.requestLocationUpdates(LocationManager.G
        PS_PROVIDER,15000,0, this);

        Parse.initialize(this,"w6sOkx6IXZPloTIkWcdbfX3RxKrrPJZ
        mWPIGHeYk","3avqsVGqNOLwIQQMst9PMxlIArOfwzkYB
        6Ls9t1u
        onRuningClick();
        onWaitingClick();
        onBreakdownClick();
        onStopClick();
        onExitClick();

}

private void onRuningClick(){
        Button btnExit = (Button) findViewById(R.id.button2);
        btnExit.setOnClickListener(new OnClickListener() {

                @Override
                public void onClick(View v) {
                ParseQuery<ParseObject> query =
                ParseQuery.getQuery("Tracking_Data");
                 query.whereEqualTo("bus_number",busNumber);
                 query.findInBackground(new
                FindCallback<ParseObject>() {
                                        @Override
                                public void
                                done(List<ParseObject> objs,
                                ParseException e) {
                                 if (e == null) {
                                        if(objs.size()==0){
                                        }else{

                                for(ParseObject obj:objs){
```

```
                                        obj.put("status", "R");
                                        obj.saveInBackground();
                                        Toast.makeText(getBaseContext(
                                        ),"Bus is
                                        Runing",Toast.LENGTH_LONG)
                                        .show();
                                        }
                                        }
                                        else {
                                        }
                                    }
                                });
                            }
                        });
                }

                private void onWaitingClick(){
                        Button btnExit = (Button) findViewById(R.id.button3);
                        btnExit.setOnClickListener(new OnClickListener() {

                                @Override
                                public void onClick(View v) {
                                 ParseQuery<ParseObject> query =
                                ParseQuery.getQuery("Tracking_Data");
                                 query.whereEqualTo("bus_number",busNumber);
                                 query.findInBackground(new
                                FindCallback<ParseObject>() {
                                                @Override
                                                 public void done(List<ParseObject>
                                                 objs, ParseException e) {
                                                        if (e == null) {
                                                                if(objs.size()==0){
                                                                }else{
                                                                obj.put("status", "W");
```

```
                                        obj.saveInBackground();

                                        Toast.makeText(getBaseContext(),"Bus
                                        is
                                        Waiting",Toast.LENGTH_LONG).sho
                                        w();
                                                }
                                                }
                                        } else {
                                        Toast.makeText(getBaseContext(),e.get
                                        Message() ,
                                        Toast.LENGTH_LONG).show();
                                        }
                                        }
                                       }
                                });
                        }
                });
        }

        private void onBreakdownClick(){
                Button btnExit = (Button) findViewById(R.id.button4);
                btnExit.setOnClickListener(new OnClickListener() {

                        @Override
                        public void onClick(View v) {
                                ParseQuery<ParseObject> query =
                                ParseQuery.getQuery("Tracking_Data");
                               query.whereEqualTo("bus_number",busNumber);
                                query.findInBackground(new
                                FindCallback<ParseObject>() {
                                        @Override
                                        public void done(List<ParseObject>
                                        objs, ParseException e) {
```

```
                                    if (e == null) {
                                    if(objs.size()==0){
                                    }else{
                                            for(ParseObject obj:objs){
                                           obj.put("status", "B");
                                            obj.saveInBackground();
                                    Toast.makeText(getBaseContext(),"Bus
                                    is
                                    Breakdown",Toast.LENGTH_LONG).sho
                                    w();
                                     }
                                    }
                                     } else {
                                Toast.makeText(getBaseContext(),e.getMessage
                                () , Toast.LENGTH_LONG).show();
                                }
                                }
                         });
                        }
                });
        }

        private void onStopClick(){
                Button btnExit = (Button) findViewById(R.id.button5);
                btnExit.setOnClickListener(new OnClickListener() {
                        @Override
                        public void onClick(View v) {
                        ParseQuery<ParseObject> query =
                        ParseQuery.getQuery("Tracking_Data");
                         query.whereEqualTo("bus_number",busNumber);
                         query.findInBackground(new
                        FindCallback<ParseObject>() {
                                @Override
```

```
                    public void done(List<ParseObject> objs,
                    ParseException e) {
                    if (e == null) {
                    if(objs.size()==0){
                     }else{
                    for(ParseObject obj:objs){
                    obj.put("status", "S");
                     obj.saveInBackground();
                     Toast.makeText(getBaseContext(),"Bus is
                    Stop",Toast.LENGTH_LONG).show();
                      }
                      }
                 } else {
                 Toast.makeText(getBaseContext(),e.getMessage() ,
                Toast.LENGTH_LONG).show();
                                }
                               }
                      });
                }
        });
    }
    private void onExitClick(){
        Button btnExit = (Button) findViewById(R.id.button1);
        btnExit.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
                System.exit(0);
            }
        });
    }

@Override
public void onLocationChanged(final Location location) {
```

```
                ParseQuery<ParseObject> query =
                ParseQuery.getQuery("Tracking_Data");
                query.whereEqualTo("bus_number",busNumber);
                query.findInBackground(new FindCallback<ParseObject>() {
                        @Override
                        public void done(List<ParseObject> objs, ParseException e) {
                                String lon_lat = "Longitude:
                              "+location.getLongitude()+" Latitude:
                              "+location.getLatitude();
                                if (e == null) {
                                if(objs.size()==0){
                                }else{
                                for(ParseObject obj:objs){
                                        obj.put("longitude", location.getLongitude());
                                        obj.put("latitude", location.getLatitude());
                                        obj.saveInBackground();
                                Toast.makeText(getBaseContext(),"Update:
                              "+lon_lat,Toast.LENGTH_LONG).show();
                                }
                                }
                                } else {
                                Toast.makeText(getBaseContext(),e.getMessage() ,
                              Toast.LENGTH_LONG).show();
                                }
                        }
                });
        }
}
```

Supervisor Approval

FINAL REPORT SUBMISSION FORM

Student Name: .....

Student Registration no.: .....

Project title: .....

..... ……

Student Signature Date

Comments of supervisor:

.....

.....

.....

Name of supervisor: ……...

…… …….

Signature of supervisor Date