

WOLLO UNIVERSITY

KOMBOLCHA INSTITUTE OF TECHNOLOGY

COLLEGE OF INFORMATICS

DEPARTMENT OF INFORMATION SYSTEM

**TITLE: - ANDROID BASED ONLINE BUS RESERVATION SYSTEM**

PREPEARED BY:

NAME IDNO

1. MUHAMED DESALEGN………………………….…WOUR/0749/11
2. SIMACHEW ALEHEGN ……………………………. WOUR/0792/11
3. ERMIYAS MAMO ……………………………………WOUR/0625/11

ADVISOR: - MR. MUHAMED JIBRIL (MSC)

KOMBOLCHA, ETHIOPIA

JULY/16 /2022

**Approval sheet**

This is to certify that the senior project titled “Android Based Online Bus Reservation System is

Prepared By: -

Name ID No Sign

1. MUHAMED DESALEGN WOUR/0749/11 \_\_\_\_\_\_\_\_\_\_

2.ERMIAS MAMO WOUR/0625/11 \_\_\_\_\_\_\_\_\_\_

3. SIMACHEW ALEHEGN WOUR/0792/11 \_\_\_\_\_\_\_\_\_\_

To the department of information system in wollo university for the partial fulfillment of the requirement of the Degree of Bachelor Science in Information system. The contents of this document in full, or in parts, have not been submitted to any other institute or university for the award of any degree or diploma.

Advisor Name\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Signature \_\_\_\_\_\_\_\_\_\_\_\_

This project has been examined with our approval as the project examiner.

*Examiner Name*:

1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *signature*\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*Signature*\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
3. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*Signature*\_\_\_\_\_\_\_\_\_\_\_\_\_

**ACKNOWLEDGEMENT**

First of all, we would like to thank almighty god for the strength, he has given us throughout our life and this project; nothing could happen without the help of god, Then we would like to express our special thanks of gratitude to our adviser Mr. Muhamed Jibril for his guidance and suggestion during the project work. Next we would like to express our thanks to our department for providing us this golden chance to work in group which help us in our future career.

**ABSTRACT**

*This Document delineates the design & development of a user-friendly Online Bus Ticketing  
System based on Android platform. In current Public transport system, Every Bus needs a  
conductor to collect money and issue ticket to each passenger; it is time consuming, manual error  
like improper distribution of ticket, passenger travelling without ticket & currency exchange many  
other problems. To overcome this many system have proposed such as ticket vending machine,  
Electronic ticketing system, and Smart ticketing system using; even they had drawbacks too. The  
main idea is to provide more convenient, cashless ticketing system i.e. an app is developed to  
generate digital ticket. This system eliminates the Man power, Paper tickets are eliminated, on  
that account it will make the passenger comfortable to trave.*

**Contents**

[Chapter One 1](#_Toc109785882)

[1. Introduction 1](#_Toc109785883)

[1.1 Background 2](#_Toc109785884)

[1.2 Statement of The Problem 2](#_Toc109785885)

[1.3 Objective of The Project 3](#_Toc109785886)

[1.3.1 General Objective 3](#_Toc109785887)

[1.3.2 Specific Objective 3](#_Toc109785888)

[1.4 Methodology 3](#_Toc109785889)

[1.4.1 The System Development Model 3](#_Toc109785890)

[1.4.2 System Analysis and Design Methodology 4](#_Toc109785891)

[1.4.3 Data Gathering Methodology 5](#_Toc109785892)

[1.4.3 Tools and Justification 6](#_Toc109785893)

[1.5 Feasibility of The Project 7](#_Toc109785894)

[1.5.1 Economic Feasibility 7](#_Toc109785895)

[1.5.2 Technical Feasibility 8](#_Toc109785896)

[1.5.3 Operational Feasibility 8](#_Toc109785897)

[1.5.4 Schedule Feasibility 9](#_Toc109785898)

[1.5.5 Legal Feasibility 9](#_Toc109785899)

[1.6 Scope and Limitation of The Project 9](#_Toc109785900)

[1.6.1. Scope of The Project 9](#_Toc109785901)

[1.6.2 Limitation of The Project 10](#_Toc109785902)

[1.7 Significance of The Project 10](#_Toc109785903)

[1.8 Organization of The Project 10](#_Toc109785904)

[1.8.1 Team Organization 11](#_Toc109785905)

[1.8.2 Target Group and Benefit of The Project 11](#_Toc109785906)

[1.9 Budget 11](#_Toc109785907)

[1.10 Communication Plan 12](#_Toc109785908)

[Chapter Two: Analysis 13](#_Toc109785909)

[2.1 Introduction 13](#_Toc109785910)

[2.2 Current System 13](#_Toc109785911)

[2.2.1 Limitations of The Current System 13](#_Toc109785912)

[2.2.2 Business Rules 14](#_Toc109785913)

[2.3 Description of the Proposed System 14](#_Toc109785914)

[2.3.1 Functional Requirements and Non-Functional Requirements 15](#_Toc109785915)

[2.3.2 Use Case Modeling 18](#_Toc109785916)

[2.3.2.1 Essential Use Case 19](#_Toc109785917)

[2.3.2.2 System Use Case 19](#_Toc109785918)

[2.3.2.2.1 Use Case Documentation 22](#_Toc109785919)

[2.3.3 Class Responsibility Collaboration Diagram (CRC) 35](#_Toc109785920)

[2.3.4 Sequence Diagram 37](#_Toc109785921)

[2.3.5 Activity Diagram 41](#_Toc109785922)

[2.3.6 Class Diagram 44](#_Toc109785923)

[2.3.7 User Interface Prototyping 45](#_Toc109785924)

[2.3.8 Identifying Change Cases 46](#_Toc109785925)

[Chapter Three: System Design 47](#_Toc109785926)

[3.1 Introduction 47](#_Toc109785927)

[3.2 Design Goals 47](#_Toc109785928)

[3.3 Design Class Diagram 48](#_Toc109785929)

[3.4 Current System Architecture 50](#_Toc109785930)

[3.5 Proposed System Architecture 50](#_Toc109785931)

[3.5.1 Subsystem Decomposition 50](#_Toc109785932)

[3.5.3 Component diagram 52](#_Toc109785933)

[3.5.4 Deployment diagram 52](#_Toc109785934)

[3.5.5 Persistent data management 55](#_Toc109785935)

[3.5.6 Access Control and Security 55](#_Toc109785936)

[3.5.7 Subsystem Service 56](#_Toc109785937)

[3.5.8 Boundary conditions and Exception Handling 56](#_Toc109785938)

[3.5.8.1 Boundary Conditions 56](#_Toc109785939)

[3.5.8.2 Exception Handling 57](#_Toc109785940)

[3.6 User Interface Design 57](#_Toc109785941)

[Chapter Four: Implementation 61](#_Toc109785942)

[4.1. Introduction 61](#_Toc109785943)

[4.2 Overview of the programming language used 61](#_Toc109785944)

[4.3 Algorithms Used 61](#_Toc109785945)

[4.4 Functional Test Specifications 66](#_Toc109785946)

[Chapter 5: Conclusion and Recommendation 68](#_Toc109785947)

[5.1 Conclusion 68](#_Toc109785948)

[5.2 Recommendation 68](#_Toc109785949)

[Reference 69](#_Toc109785950)

**List of tables**

[Table 1.1 data collecting techniques 5](#_Toc113782337)

[Table 1.2 tools 6](#_Toc113782338)

[Table 1.3 Schedule 9](#_Toc113782339)

[Table 1.4 budget 11](#_Toc113782340)

[Table 1.5 communication plan 12](#_Toc113782341)

[Table 2.6 Requirement table 16](#_Toc113782342)

[Table 2.7 use case description for log in use case. 21](#_Toc113782343)

[Table 2.8 use case description for Make Reservation use case. 22](#_Toc113782344)

[Table 2. 9 use case description for ticket cancelation use case 23](#_Toc113782345)

[Table 2.10 use case description for check ticket availability use case 24](#_Toc113782346)

[Table 2.11 use case description for view schedule use case 25](#_Toc113782347)

[Table 2.12 use case description for generate report use case 26](#_Toc113782348)

[Table 13 Use case description for get ticket use case 27](#_Toc113782349)

[Table 2.14 use case description for add Schedule use case. 28](#_Toc113782350)

[Table 2.15 use case diagram for delete route use case 28](#_Toc113782351)

[Table 2.16 use case of free chat description. 29](#_Toc113782352)

[Table 2.17 use case of see map description. 30](#_Toc113782353)

[Table 2.18 use case of see passenger. 31](#_Toc113782354)

[Table 2.19 use case of see passenger. 32](#_Toc113782355)

[Table 2.20 CRC for Admin 33](#_Toc113782356)

[Table 2.21 CRC for Passenger 34](#_Toc113782357)

[Table 2.22 CRC for Clerk 34](#_Toc113782358)

[Table 3.23 Passenger Table 53](#_Toc113782359)

[Table 3.24 Bus Table 53](#_Toc113782360)

[Table 3.25 Reservation Table 53](#_Toc113782361)

[Table 3.26 Blind Table 54](#_Toc113782362)

[Table 3.27 Payment Table 54](#_Toc113782363)

[Table 3.28 Clerk Table 54](#_Toc113782364)

**List of figures**

[Figure 1.1 Lifecycle of Agile 4](#_Toc113782365)

[Figure 2 2 Essential use case 19](#_Toc113782366)

[Figure 2.3 System Use Case 20](#_Toc113782367)

[Figure 2.4 Sequence diagram for login 36](#_Toc113782368)

[Figure 2.5 Sequence diagram for get ticket 37](#_Toc113782369)

[Figure 2.6 Sequence diagram for Deposit For Passenger 38](#_Toc113782370)

[Figure 2.7 Activity diagram for admin 39](#_Toc113782371)

[Figure 2.8 Activity diagram for passenger 40](#_Toc113782372)

[Figure 2.9 Activity diagram for clerk 41](#_Toc113782373)

[Figure 2.10 Class diagram 42](#_Toc113782374)

[Figure 2.11 User interface prototyping 43](#_Toc113782375)

[Figure 3.12 Design Class Diagram 47](#_Toc113782376)

[Figure 3.13 Software Archetecture 48](#_Toc113782377)

[Figure 3.14 Component Diagram 50](#_Toc113782378)

[Figure 3.15 Deployment Diagram 52](#_Toc113782379)

[Figure 3.16 Relationship Among Tables 55](#_Toc113782380)

**List of Acronyms and Abbreviations**

CRC: Class responsibility collaborator.

GUI: Graphical User Interface

HTML: Hyper Text Markup Language

MYSQL: Structured Query Language.

OOA: Object Oriented Analysis.

OOD: Object Oriented Design

PHP: Hyper Preprocessor Scripting Language

UML: Unified Modeling Language.

UC: Use case.

# Chapter One

## 1. Introduction

The Online Bus Ticket Reservation System is android -based application that allows visitors check bus ticket availability, buy bus ticket and pay the bus ticket online. This system is established for all the home/office users after gaining access from the administrator. Online Bus Reservation System provides bus transportation system, a facility to reserved seats, cancellation of seats and different types of enquiry which need an instant and quick reservation. This system can be used by the users in performing online reservation via internet for their all business purposes. Users can use this program directly on their mobiles by install it.

The current reservation system is still conducted manually and separately at each branch, contact must be made by each branch’s front-officer to the head office for each customer’s enquiry in order to get the latest update on schedule, seat availability and other reservation-related information; as well as to avoid duplicate bookings or over-capacity. There is also a physical limit to the reservation availability as each branch only operates during certain hours and reservations can only be made on-the-spot. use a means of transportation, or have access to some internet services. Android Based Bus Ticket Reservation System enables the customer to buy bus ticket, make payment, and ask for information online easily. Furthermore, staff can sell bus ticket using Bus Ticket Reservation System after check bus ticket availability for the customer. The method to solve this problem is to create an online buying bus ticket system. Customer can buy the bus ticket over the Internet, 24 hours a day, 7 days a week and the bus ticket can't be lost, stolen or left behind. In addition, the online system lets the customers check the availability of the bus ticket before they buy bus ticket. Furthermore, customers no need to pay cash to buy bus ticket because they can pay the bus ticket by using deposit slip number order.

## 1.1 Background

Transportation applications are no exception whether it is the convenience of a ride sourcing app, like Uber, transforming the ride-for-hire industry, or expanding gamification opportunities on traffic apps, like Waze to develop ever smarter driver routing, transportation apps are profoundly influencing how travelers interact with the transportation system and traveler behavior.

A way back before the emergence of transport mobility, due to smaller marketplace and lesser demand, it was easier to maintain and dispatch the transport and logistics deliveries. But with the recent rise in investment and development in infrastructure has triggered far-reaching changes. It has efficiently increased the revenue of the transport industry and developed its network across the globe. This constant growth in the industry has not only presented a wide range of traveling and logistics options to the people but also increased the complexity of choosing a best-fit.

As a result, customers are looking for software that can manage the transport services and provide updates features to the user. Smartphone Applications for Transportation helps business owners to increase productivity and efficiency. It also encourages them to provide a better customer experience.

In Ethiopia currently, staff at the bus ticket counter is using an internal system to sell tickets at the counter and customers who are unable to buy bus ticket online at this moment would have to go to the counter to a buy bus ticket. Sometimes, customers’ needs to queue up a long queue to buy bus ticket and ask for information and this brings a lot of inconveniences to customers however, Online Bus Ticket Reservation System enables the customer to buy bus ticket, make payment, and ask for information online easily. Furthermore, staff can sell bus ticket using Bus Ticket Reservation System after checking the bus ticket availability for the customer.

Online Bus ticketing system is a total internet ticketing operation offering the benefit of total in-house management of bus schedules, ticket bookings, ticket sales, report generation, and other business functions associated with ticket sales [1] .

## 1.2 Statement of The Problem

A problem statement always expresses the words that will be used to keep the effort on tracking things out to represent a solvable problem with the existing system [2]. Below are the problems with the current system.

* Systems that are used by the staff at the counter currently are manually and just used to sell the bus ticket at the different counter, this causes the service not easily accessible.
* Customer has to go the counter to buy the bus ticket or ask for schedule.
* Furthermore, customers need to pay cash when they buy the bus ticket and sometimes needs to queue up long time to get the bus ticket.
* Operation times of the bus station to book ticket are limited to the office hours only.
* Current ticketing operated by individual bus operators, does not provide enough

choices to a passenger such as different destinations and the availability of tickets are.

## 1.3 Objective of The Project

### 1.3.1 General Objective

The main purpose of this study is to develop a secure Android Based online bus booking system that facilitate efficient purchase of ticket and replace the manual ticketing system.

### 1.3.2 Specific Objective

To achieve the general objective mentioned above the following are specific objective:

* To identify the problem of the existing system.
* To perform requirement analysis to find out the system functional and non-functional requirements.
* To design the new system based on the identified requirements using object-oriented models in order to understand the system and implementation easily.
* To implement the new system based on the new system design.
* To test the functionality of the system.

## 1.4 Methodology

### 1.4.1 The System Development Model

There are various software development life cycle models defined and designed which are followed during the software development process. These models are also referred as “Software Development Process Models". Each process model follows a Series of steps unique to its type to ensure success in the process of software development. But the proposed system follows Agilemodel. It is an iterative and step-by-step software development methodology. Agile application development helps organize designing and planning methods, development, and testing methods during a software lifecycle. We use this model because using the agile method include increased customer satisfaction, lower defect rates, and faster development times. Additionally, the agile method is an answer to rapidly changing requirements, as it uses early feedback on technology features of project deliverables.



Figure 1. Lifecycle of Agile

### 1.4.2 System Analysis and Design Methodology

Among the different methodologies available we plan to use the object-oriented design methodology for the development of our system. Because it is best way to construct, manage and assemble objects that will be implemented in our system, and the composition of objects and collaboration between objects on the system. Object oriented design methodology has two phases:

* **Object Oriented Analysis (OOA**): During this phase we looked at the problem domain, and with the aim of producing a conceptual model of the information that exists in the area which will be analyzed, and Modeled the functions of the system (use case modeling), identifying the business objects, organize the objects and also the relationship between them and finally model the behavior of the objects.
* **Object Oriented Design (OOD):** During this phase Model object interactions and behaviors that support the use case scenario, and finally update object model to reflect the implementation environment, and also transforms the conceptual model produced in object-oriented analysis, so that we can use this phase to refine the use case model to reflect the implementation environment.

Among the different methodologies in practice we thoughtfully pay our time to work on Objected Oriented Programming (OOP) approach for achieving effective and reliable working system in the future. We will use OOP because of the following advantages:

* **Code reusability** - new objects can be derived from old objects.
* **Code Modularity** - Everything in OOPL is an object, they can be interchanged or removed to meet the user’s needs.
* **Easier maintenance** - Inheritance usually reduces maintenance.
* **Design stability** - Once you have developed a stable base class, new classes that you derive may have less errors and bugs.

### 1.4.3 Data Gathering Methodology

Data collection is the most important part of the project to find the main requirement of the system and to understand how the system does and also to understand how the overall organization work flow looks like.

Some of the methods that we apply to collect relevant information are: -

Table .1 data collecting techniques

|  |  |  |
| --- | --- | --- |
| **Technique** |  | **Key Facts** |
| **Interviews** |  |  Questions should be focused clear open-ended responses. |
|  |  |  Interviews are mainly qualitative in nature |
| **Document Analysis** |  |  Collect certain relevant information from written documents. |
| **Observation** |  |  frequency counts of target behaviors. |
|  |  |  Good source in providing additional information about a  particular assembly |

### 1.4.3 Tools and Justification

The programming tools that the project concentrates are the following:

Table 1. tools

|  |  |  |
| --- | --- | --- |
| **S/NO** | **TOOLS** | **JUSTIFICATION** |
| **1.** | **Java** | Java is a strange state, dynamic, pitifully wrote, model based, multiword, and deciphered programming dialect. |
| **2.** | **Android Studio** | Android studio It is the official condition for android application improvement. It is extremely easy to understand and has a simplified capacity which proves to be useful amid the framework execution. |
| **3.** | **000webhost** | 000webhost is a web growth environment; the developer used the tool to design web pages which were to be linked some of the application by using android studio. |
| **4.** | **PHP** | PHP is a server-side scripting dialect composed principally for web development yet in addition utilized as a universally useful programming language. |
| **5.** | **IDE** | Integrated Development Environment (IDE) is a software application that gives far reaching offices to PC software engineers for programming improvement. |
| **6.** | **MySQL** | MySQL is an open source social database administration framework (RDBMS). MySQL is a focal segment of the LAMP open source web application programming stack (and other "AMP" stacks). |

**Software requirements**:

* Android SDK
* MS word 2007: for documentation.
* Paint: for editing
* Window 10 Operating system
* MS power point for presentation
* Edrawmax, MS Visio, Visual paradigm for drawing different diagrams

**Hardware requirements**:

* Printer: for printing document
* Flash disks(4GB,2GB,8GB)
* Computer
* CD Rom

## 1.5 Feasibility of The Project

It is an analysis of the ability to complete a project successfully, taking into account legal, economic, technological, scheduling and other factors. Rather than just diving into a project and hoping for the best, a feasibility study allows project managers to investigate the possible negative and positive outcomes of a project before investing too much time and money.

### 1.5.1 Economic Feasibility

Assessing economic feasibility involves comparing the costs of the information systems development project with the benefits the information system is going to have for the organization once it has been developed. From this point of view our project is economically feasible due to the following reasons:

* Reduce the amount of money that we need to go to the bus station to check whether if the bus is available or not.
* free offline chat that saves money from using online chat
* It makes the bus station act like a bank because the passengers can deposit money in the bus station.

Due to the above reasons we will have a net benefit that makes our project economically feasible.

### 1.5.2 Technical Feasibility

This feasibility evaluation determines whether the technology needed for the proposed system to be developed is available or not.

* The system can be developed through tools and devices which are currently available on the market (example desktop computer, Edrawmax etc.).
* The project can be working with current equipment, existing software technology & available personnel.
* The needed equipment and software may also easily access and can successfully satisfy the user requirement.

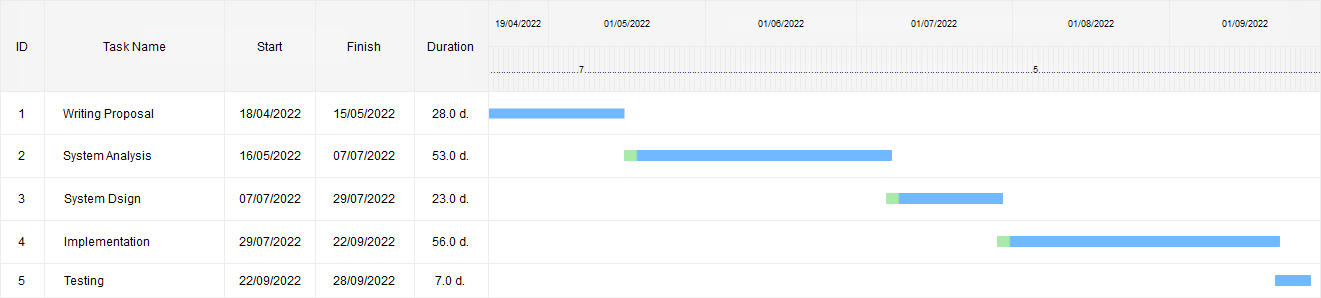
### 1.5.3 Operational Feasibility

Proposed applications are beneficial only if they can be turned into user friendly that meet the users’ requirements. Simply stated, this test of feasibility asks if the application will be worked when it is developed. Therefore, the system will be designed to be operationally feasible. That means, the system will operate without failure. Because of it is simplicity and easy access. In addition to this the system is practical, applicable and also the system operation is easy for the users. The new system that we develop require organization end user potential and skilled man power, also social acceptability that the system completely changed from manual system to computerized due to this potential and skilled man power of our team to operate the system is operationally feasible.

### 1.5.4 Schedule Feasibility

It involves checking if the project team can build the proposed system within the time allotted. Within the time duration, we have identified the activities of the project in order to accomplish the project objective within their schedule requirement which is on the table below.

Table 1. Schedule



### 1.5.5 Legal Feasibility

The proposed system not conflicts with legal requirements, the government/ company. It meets to the rule and regulations of the organization or the university or it is not conflict to each other.

## 1.6 Scope and Limitation of The Project

### 1.6.1. Scope of The Project

This Project work focuses on two parties, i.e. Staff and the customers. The staffs are the bus station administrator, clerk, and the transport authority. the system administrator may be specialized member of the company staff who administer other staff to this system, the administrator is able to add, edit delete, manage route, manage bus schedule, manage news. The other functions of staffs are to retrieve information, travelling issues and to add new buses to the bus station etc. This activity will help the company to evaluate its current position and to plan its company’s operations on how and what action to be taken in order to stay ahead in this competitive business world. The customer will be able to utilize this Online Bus Ticketing System portal to perform their transaction of purchasing bus tickets at their own hassle-free time, postpone/cancel ticket booked, and give feedback as a decision support system.

### 1.6.2 Limitation of The Project

Even if we try to do the project in good way our project has some limitations:

* Inconvenient for first time users because their account needs to have available balance before a ticket can be purchased.
* Printing out of ticket is unavailable.
* The system will not provide facility to pay the fine online by means of credit card/ banking.

## 1.7 Significance of The Project

* User friendly because the steps to purchase a ticket are simple and direct.
* Blinds can reserve seat number (get ticket) by just Taping the Screen.
* Help tourists with choosing of cities to visit.
* The users can check the schedule for the specified route.
* Payment method is reliable online.
* Online checking for bus schedule for the specified route.
* The admins or passengers can get the map of their current city and, they can get the driving road map from their current city to their destination city with kilo meters to reach the destination and also searching and viewing cities with map.
* You can book your Transport Tickets directly from your mobile phone.
* Provide free chat without needing of turning on mobile data or Bluetooth and without needing of connected to WIFI network.
* Handles the violation of law doing by the drivers like undue payment of the passengers.

## 1.8 Organization of The Project

* **Chapter one:** This chapter covers proposal part of the project and gives brief explanation about background of the project, how existing system works, proposed system as whole this portion deals about clearly describe existing system and proposed system and objectives of the project
* **Chapter two: -**This chapter contains all about SRS and system analysis parts of the system
* **Chapter three: -**This chapter deals about details of system design of the system
* **Chapter four: -**This portion deals about implementation design of the system that means graphical user interface and system database that data storage memory
* **Chapter five: -**This portion deals about Recommendation and conclusion

### 1.8.1 Team Organization

The team responsible for every activities of this project work is organized and grouped based on our talent. We are three in number, namely: - MUHAMED DESLEGN, ERMIAS MAMO and SEMACHEW ALEHEGN, but we do every activities of the system together.

### 1.8.2 Target Group and Benefit of The Project

The primary target of this project is to simplify the process of getting bus tickets and get full information about the travel.

**Benefit of the project:**

* Reduce time needed to get tickets.
* Reduce the amount of money of getting tickets by handling illegal overblown payments
* Reduce users load since the transaction is made online.

## 1.9 Budget

The following table lists budget required for the successful development of the proposed system.

Table 1. budget

|  |  |  |  |
| --- | --- | --- | --- |
| Item | quantity | unit price | total price |
| stationary materials | 3 pen,1desta paper | For pen =40 birr  For DESTA paper=450 birr | 490 |
| CD Rom disk | 1 | 2300 birr | 2300 |
| flash disk | 16Gb | 250 birrs | 250 |
| Internet access cost | - | 6000 birrs | 6000 |
| mucocutaneous cost | - | 1000 | 1000 |
| PC | 1 | - | 20000 |
| total cost | - | - | 30,040 birrs |

## 

## 1.10 Communication Plan

Table 1. communication plan

|  |  |  |
| --- | --- | --- |
| **Days** | **Team** | **Advisor** |
| Monday | MC900441310[1] | We meet our advisor  on Mondays and Fridays |
| Tuesday | MC900441310[1] |
| Wednesday | MC900441310[1] |
| Thursday | MC900441310[1] |
| Friday |  |
| Saturday | MC900441310[1] |
| Sunday |  |

# Chapter Two: Analysis

## 2.1 Introduction

The project development team uses an object-oriented system development methodology. Because the Object system development approach gives easier and natural way to break down problems into simple, small and manageable components so that it reduces the vague appearance of the big problem. Moreover, it is predominately used and popular method in present software development trend.

The major activities described in this chapter are Constructing a use case model, Documenting the use case course of events, constructing sequence and activity diagram analysis level class diagram and user proto type about the proposed system.

## 2.2 Current System

Currently Bus reservation in Ethiopia uses a manual ticketing system which is a passenger can reserve ticket by going to the ticket office physically and after waiting a long queue the ticket attendant asks question. These questions are listed below:

* What is there phone number?
* Where they want to go?
* When they want go?

After the question is completely answered the ticket attendant gives the ticket to the passenger. Currently company store passenger information’s in paper file.

### 2.2.1 Limitations of The Current System

The data gathered deduced that, most of the interview results agree that many problems are encountered with the current system these include insecurity, redundancy, inconsistency, delay in data accessing, ineffective retrieval of data and its often inefficient in time and cost etc., and off course there is need to overcome these problems.

It has several problems which is described below:

* The details information of the passenger is traditionally paper based and maintained on paper.
* Finding out details regarding any information is very difficult because information in paper-based form.
* Existing system require great amount of manual work has to be done. The amount of manual work increases rapidly with increase in bus services.
* Needs a lot of working staff and extra attention on all the records.
* It is time consuming.
* It is wasteful which means it require a lot of cost.
* It causes data loss because there is no proper handling of data.
* It is not easily accessible.
* The passengers must wait for their required bus without reserving

### 2.2.2 Business Rules

A business rule is a statement that defines or constrains some aspect of the business. It is intended to assert business structure or to control or influence the behavior of the business. The business rule regarding to the current bus ticketing are the following:

**BR1**: The user get ticket physically or using phone call.

**BR2**: To get ticket the customer must go to the counter office or the bus station.

**BR3**: The customer must be present early in the bus station to choose the seat.

**BR4**: If the customer wants to cancel travel, he must sell his ticket to another traveler unless the ticket will be disqualified.

## 2.3 Description of the Proposed System

Indeed, mobile bus ticketing system (MBTS) is the most noteworthy prospects in Ethiopia to reduce expenditures and increase traveler's accessibility. This project will reduce ticket processing flow, reduce usage of paper and allows greater convinces and flexibility to the traveler in Ethiopia cities and allow travel agent to make alterations to the journey. There are other important issues from the use of this technique such as the mobile ticket cannot be lost or stolen on the contrary of sending the ticket by mail also there is a probably of sending it to the wrong address. MBTS will make customers' lives easier, and can get the service by himself in anytime, anywhere and any devices [3].

In efforts to Developing bus e-ticketing systems in Ethiopia, observation on the problems and opportunities from the existing Manual ticketing in Ethiopia had been conducted With that, it is recommended an integrated system of solutions that attempts to rectify many of the existing problems in the current bus ticketing and propose an innovative way to enhance the services provided in the bus Manual ticketing system, Therefore, the goal of the proposed Online Bus Ticketing System is to provide a revolutionary way to interact effectively in a one stop venue.

In short, with the proposed system, bus operators can now involve in making bus ticketing purchase transactions convenient, easy, cost effective and finally can-do selling bus tickets online. Therefore, the proposed system will be superior and function as a catalyst in the competitive business environment regardless of the geographic barricades among the bus operators.

This Mobile ticketing (m-ticketing) can be broadly defined as ordering, purchasing, delivery and usage of a product or service using mobile technology such as Wireless Application Protocol (WAP) [4].

Generally, the mobile ticketing process can be defined in the following steps:

* Registration: Online ticketing requires a company to register with all the business and services information.
* Provisioning: Allow mobile phone application to interact with customers, allowing the purchase to take place.
* Validation: Validating and legitimate the ticket via online system between the company and the customer.
* Ticket check: Controller to accept the sales and display of the mobile ticket as a valid ticket for the passenger, according to the terms displayed on the ticket.

### 2.3.1 Functional Requirements and Non-Functional Requirements

**Functional requirement** is a function or feature that must be included in an information system to satisfy the business needs and user acceptance. Staff and Customers will use this proposed Online Bus Ticketing System online portal.

**Non-functional requirements** are requirements that are not directly related to the functional aspects of the system.

In the priority column, the following short hands may use:

* M: mandatory requirements (something the system must do).
* D: desirable requirements (something the system preferably should do).
* O: optional requirement.

Table 2. Requirement table

|  |  |  |  |
| --- | --- | --- | --- |
| **Functional Requirements** | | |  |
| **#** | Requirement ID | Requirement Description | Priority |
|  | **MBTS\_1** | **Check schedule** |  |
| 1 | MBTS \_1\_1 | The system should allowusers to check about ticket availability by seeing from where to where. | M |
| 2 | MBTS \_1\_2 | The system must allowthe users to find all the information’s before press book button. | M |
|  | **MBTS\_2** | **Login to the system** |  |
| 3 | MBTS\_2\_1 | The system must allowusers to login the system, the user must enter his valid username and password to access to the system, otherwise the system will show an error message. | M |
|  | **MBTS\_3** | **Manage Timetable** |  |
| 4 | MBTS\_3\_1 | The system shallallow admins to create a new timetable (include all the trip details) | M |
| 5 | MBTS\_3\_2 | The system shouldallow admins to manage (update, edit and delete) timetable | M |
| 7 | MBTS\_3\_3 | The system shallallow users to check the timetable schedule | M |
|  | **MBTS\_4** | **Edit Bus** |  |
| 8 | MBTS\_4\_1 | The system mustallow admins to create a new bus (include all the bus details) | M |
| 9 | MBTS\_4\_2 | The system must Allow Admins to manage (update, edit and delete) buses | M |
|  | **MBTS\_5** | **Edit Route** |  |
| 10 | MBTS\_5\_1 | The System Shall AllowAdmin to create a new route (include all the route details) | M |
| 11 | MBTS\_5\_2 | The system must allow admins to manage (update, edit and delete) routes | M |
|  | **MBTS\_6** | **Edit Station** |  |
| 12 | MBTS\_6\_2 | The system should allow admins to manage (update, edit and delete) stations data. |  |
|  | **MBTS\_8** | **Edit Staff** |  |
| 13 | MBTS\_8\_1 | The system shall allowadmins to create a new staff (include all the staff details) | M |
| 14 | MBTS\_8\_2 | The system must allowadmins to manage (update, edit and delete) staffs | M |
|  | **MBTS\_9** | **Manage Profile** |  |
| 15 | MBTS\_9\_1 | The system must allow **a**dmins to manage clerk profile | M |
| **Non-functional Requirements** | | |  |
|  | **MBTS\_1** | **Usability** |  |
| 1 | MBTS \_1\_1 | The system should be user friendly and efficiently | M |
| 2 | MBTS \_1\_2 | The system must be easy and understandable | M |
| 3 | MBTS \_1\_3 | The system should be expandable | M |
|  | **MBTS \_2** | **Reliable** |  |
| 4 | MBTS \_2\_1 | The System should be able to perform its required functions under stated conditions. | M |
| 5 | MBTS \_2\_2 | The system shall stay working smoothly without any interruption | M |
|  | **MBTS \_3** | **Security** |  |
| 6 | MBTS \_3\_1 | The system shall use SELINUX to enforce mandatory access control. | M |
|  | **MBTS \_4** | **Availability** |  |
| 7 | MBTS \_4\_1 | The system should have high up time. | M |
|  | **MBTS \_5** | **Performance** |  |
| 9 | MBTS \_5\_1 | The system must have a high speed of manipulation data and reply to the user requests. | M |

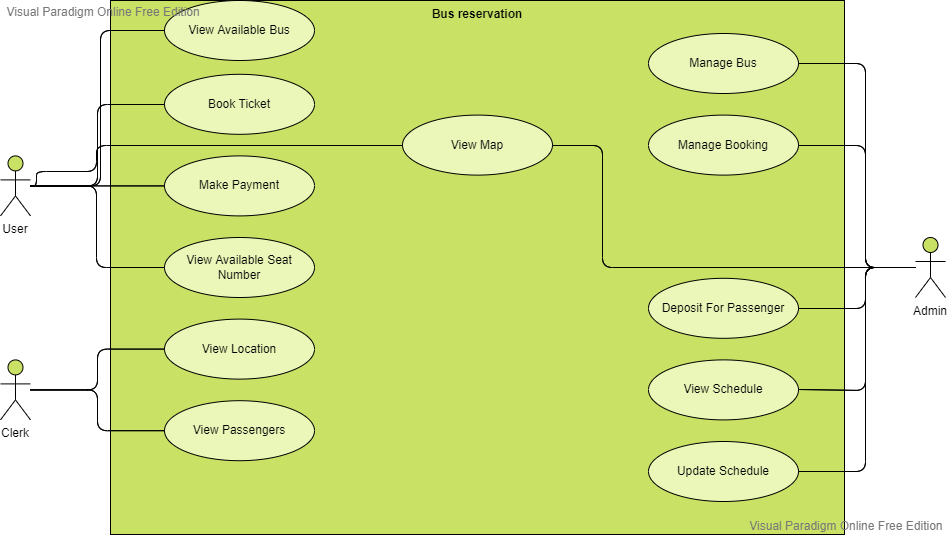
### 2.3.2 Use Case Modeling

Use case diagram in UML used to represent functional requirements, however, is familiar model artefact helping to specify the requirements of computer-based systems in the early phase of development. Figures Below which describe the main overall interaction between MBTS and (Passenger, Admin, and Clerk), as showing in figure. Each Use Case has a description which describes the functionality that will be built in the proposed system. It has two types:

* Essential use case
* System use case

#### 2.3.2.1 Essential Use Case

It is known as business or abstract use case. Intended to capture the essence of problems through technology free, idealized and abstract

Figure 2 Essential use case

#### 2.3.2.2 System Use Case

A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. The use case is made up of a set of possible sequences of interactions between systems and users in a particular environment and related to a particular goal. The use case should contain all system activities that have significance to the users. A use case can be thought of as a collection of possible scenarios related to a particular goal, indeed, the use case and goal are sometimes considered to be synonymous.

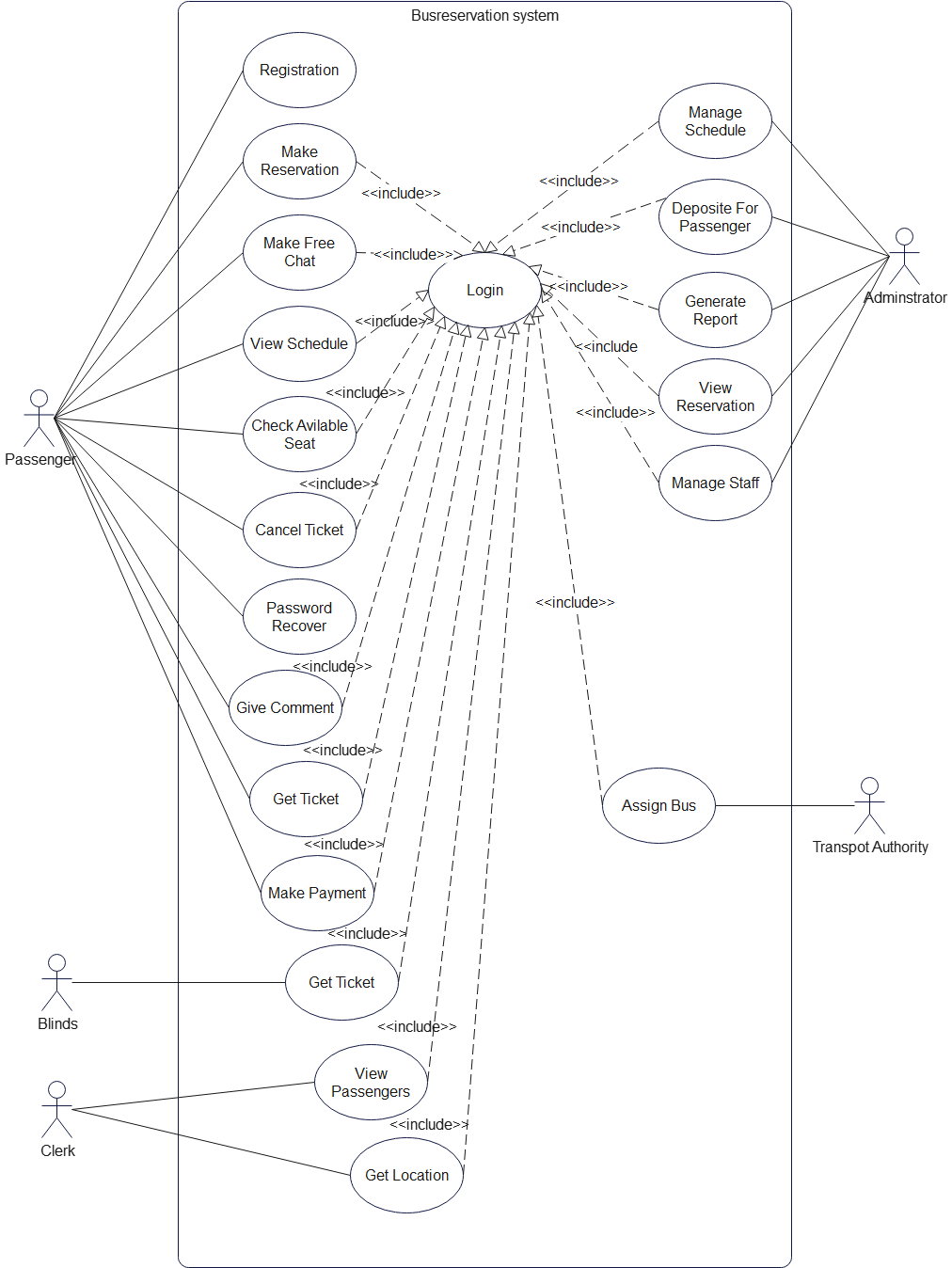


Figure 2. System Use Case

#### 2.3.2.2.1 Use Case Documentation

The third step is to document each of the above use case courses of events to determine the requirement use cases as described in the following section. So, the following consecutive tables show the use case documentation for each of the use cases that has identified in the above use case diagram. Each table contains the use case name, the actor which initiates and interacts with the use case, description of the use case and typical course of events that show the interaction between the actor and the use case which enable the team to easily depict the functions of the proposed system.

**1) Log in**

Table 2. use case description for log in use case.

|  |  |
| --- | --- |
| Use case name | Log in |
| Use case identifier | UC1 |
| Actor | Administrator(admin), or clerk |
| Description | It allows the existing user to login |
| Pre-condition | Clerk or admin or transport authority must have valid user name and password. |
| Post-condition | Clerk or Admins or clerk logs in to the system. |
| Basic course of action | 1. Transport authority or admin or clerk launch the system. 2. Transport authority or admin or clerk get different activity. 3. The system displays the login activity. 4. Transport authority or admin or clerk fills user name and password and submits it to the system. 5. The system checks the login information whether it is valid or not. 6. Transport authority or admin or clerk gets their profile. 7. Use case ends. |
| Alternative course of action | AC1: If step 4 is invalid  A2: Restart from step 3 |
| Special requirements | * A user must have username and password |

**2) Make Reservation**

Table 2. use case description for Make Reservation use case.

|  |  |
| --- | --- |
| Use case name | Make Reservation |
| Use case identifier | UC2 |
| Actor | Passenger |
| Description | It allows the user to enter the passenger details |
| Pre-condition | The passenger must log in to the system. |
| Post-condition | The passenger will reserve a ticket |
| Basic course of action | 1. Passenger launches the system. 2. The system displays the login activity. 3. Passenger log in to the system. 4. Passenger browses the reservation activity. 5. The system displays the reservation activity. 6. Passenger fills the destination and secrete code in the form then click button. 7. The system gives back the reservation verification and journey information. 8. Use case end |
| Alternative course of action | AC1: If step 6 is invalid.  AC2: The system displays not available bus message. |
| Special requirement | * A ticket must be available |

**3) Cancel ticket**

Table 2. use case description for ticket cancelation use case

|  |  |
| --- | --- |
| Use case name | Cancel ticket |
| Use case identifier | UC3 |
| Actor | Passenger |
| Description | It allows the passenger to cancel the reserved ticket |
| Pre-condition | The passenger must log in to the system. |
| Post-condition | The reserved ticket will be cancelled. |
| Basic course of action | 1. Passenger launches the system. 2. The system displays the passenger page. 3. Passenger log in to the system. 4. Passenger get the ticket passenger Activity. 5. The system displays the ticket cancelation button. 6. Passenger fills the required data and submits the form. 7. The system display confirmation of ticket cancelation. 8. Use case end. |
| Alternative course of action | AC1: If step 3 is invalid:  AC2: return to step 2. |
| Special requirements | * A passenger must reserve ticket |

**4) Check availability of seat number**

Table 2. use case description for check ticket availability use case

|  |  |
| --- | --- |
| Use case name | Check availability of seat Number |
| Use case identifier | UC4 |
| Actor | Passenger |
| Description | This use case allows the passenger to see Available seat Number. |
| Pre-condition | The passenger must log in to the system. |
| Post-condition | Shows the availability of Seat Number |
| Basic course of action | 1. Passenger launches the system. 2. The system displays the passenger activity. 3. Passenger access system via UC2. 4. Passenger gets the passengers Activity 5. The system displays Available Seat Number Button. 6. Passenger fills the required data then click button. 7. The system shows available seat number. 8. Use case end. |
| Alternative course of action | AC1: if step 5 is not available  AC2: The system displays not available bus message |

**5) View schedule**

Table 2. use case description for view schedule use case

|  |  |
| --- | --- |
| Use case name | View schedule |
| Use case identifier | UC5 |
| Actor | Passenger |
| Description | This use case allows the passenger to see information related with journeys. |
| Pre-condition | The passenger must log in to the system. |
| Post-condition | It will show fare and time of the journeys. |
| Basic course of action | 1. Passenger launches the system. 2. The system displays the login activity. 3. Passenger access the system via UC2. 4. Passenger gets schedule page. 5. The system displays the required page. 6. The system displays the information related with the bus. 7. Use case end. |
| Alternative course of action | AC1:   * If the passenger fills incorrect data the system displays error message and the view passenger activity. |

**6) Generate report**

Table 2. use case description for generate report use case

|  |  |
| --- | --- |
| Use case name | Generate report |
| Use case identifier | UC6 |
| Actor | Administrator |
| Description | This use case allows to generate report about every journey |
| Pre-condition | Administrator must log in to the system. |
| Post-condition | It will generate report |
| Basic course of action | 1. Administrator launches the system. 2. The system displays the login activity. 3. Administrator log in to the system. 4. Administrator gets the generate report button. 5. The system displays the requested activity. 6. Admin generate report. 7. The system displays message successfully compellation of use case. 8. Use case end. |
| Alternative course of action | None |

**7) Get ticket**

Table Use case description for get ticket use case

|  |  |
| --- | --- |
| Use case name | Get Ticket |
| Use case identifier | UC7 |
| Actor | Blinds |
| Description | This use case allows to get ticket for blinds |
| Pre-condition | No. |
| Post-condition | Make reservation |
| Basic course of action | 1. blinds launch the system. 2. Blinds make single tape on the mobile 3. Blinds follow the instruction telling by the system. 4. Ticket gated. 5. Use case end. |
| Alternative course of action | None |

**8) Add route**

Table 2. use case description for add Schedule use case.

|  |  |
| --- | --- |
| Use case name | Add Schedule |
| Use case identifier | UC8 |
| Actor | Administrator |
| Description | This use case allows addition of a new route |
| Pre-condition | Administrator must log in to the system. |
| Post-condition | New route will be added to the database. |
| Basic course of action | 1. Admin launches the system. 2. The system displays the login activity. 3. Admin log in to the system. 4. Admin gets’ the arrange bus button. 5. The system displays the requested activity. 6. Admin fills the required data then submit the activity. 7. The system displays successfully compellation message. 8. End of use case. |
| Alternative course of action | AC1: If step 6 is invalid  AC2: Return to step 4 |
| Special requirements |  |

**9) Deposit for passengers**

Table 2. use case diagram for delete route use case

|  |  |
| --- | --- |
| Use case name | Deposit for Passenger |
| Use case identifier | UC9 |
| Actor | Administrator |
| Description | This use case allows to deposit money for passengers |
| Pre-condition | Administrator must log in to the system. |
| Post-condition | Deposit for Passenger |
| Basic course of action | 1. Admin launches the system. 2. The system displays the login activity. 3. Admin log in to the system. 4. Admin gets’ the deposit for passenger route activity 5. The system displays the requested filling textboxes. 6. Admin fills the required data then submit the activity. 7. The system displays message successfully deposited. 8. End of use case. |
| Alternative course of action | AC1: If step 6 is invalid  AC2: Restart from step 4 |

**10) Make free chat**

Table 2. use case of free chat description.

|  |  |
| --- | --- |
| Use case name | Free Chat |
| Use case identifier | UC10 |
| Actor | Passenger |
| Description | This use case allows passengers to chat free from cost. |
| Pre-condition | passenger must log in to the system. |
| Post-condition | Chat with friends. |
| Basic course of action | 1. Passenger launches the system. 2. The system displays the passenger activity. 3. passenger go to free chat activity. 4. The system displays the requested activity. 5. Passenger send message to the guy he wants. 6. The system displays message successfully message. 7. End of use case. |
| Alternative course of action | No |

**11) See map**

Table 2. use case of see map description.

|  |  |
| --- | --- |
| Use case name | See Map |
| Use case identifier | UC11 |
| Actor | Passenger or Administrator |
| Description | This use case allows to see current location and different location. |
| Pre-condition | passenger or administrator must log in to the system. |
| Post-condition | See current location and destination location. |
| Basic course of action | 1. Passenger or administrator launches the system. 2. The system displays the passenger or administrator activity. 3. passenger or administrator go to Route activity. 4. The system displays the requested activity. 5. Passenger or administrator see their current location or destination location. 6. End of use case. |
| Alternative course of action | No |

12**) See passenger**

Table 2. use case of see passenger.

|  |  |
| --- | --- |
| Use case name | See Passenger |
| Use case identifier | UC12 |
| Actor | Clerk |
| Description | This use case allows The Clerks to See Passengers Who Get Ticket |
| Pre-condition | Clerk Must Open the System |
| Post-condition | See Passengers Who Get Ticket. |
| Basic course of action | 1. Clerk launches the system. 2. The system displays the Login Activity. 3. Clerk sign In to clerk account. 4. The system displays the requested activity. 5. Clerk select city. 6. The system displays Passengers Who get ticket for that city. 7. End of use case. |
| Alternative course of action | No |

13**) Get location**

Table 2. use case of see passenger.

|  |  |
| --- | --- |
| Use case name | Get Location |
| Use case identifier | UC13 |
| Actor | Clerk |
| Description | This Use case allows The Clerks to Get the Current Location of Blinds. |
| Pre-condition | Clerk Must Open the System |
| Post-condition | Get Locations of Blinds. |
| Basic course of action | 1. Clerk launches the system. 2. The system displays the Login Activity. 3. clerk sign in to clerk account. 4. The system displays the requested activity. 5. Clerk select city. 6. The system displays get passengers location who get ticket for that city. 7. Clerk click get location button. 8. System display current location of passenger. 9. End of use case. |
| Alternative course of action | No |

### 2.3.3 Class Responsibility Collaboration Diagram (CRC)

CRC is a collection of standard index cards that have been divided into three sections such as class, responsibility, and collaboration.

* A class represents a collection of similar objects
* Responsibility is something that a class knows or does and
* Collaborator is another class that a class interacts with to fulfil its responsibilities.

Identify the concepts and things that are important for the system and draw a CRC card for them. This helps to identify objects that the system deals with and how they collaborate/interact/ with each other.

Table 2. CRC for Admin

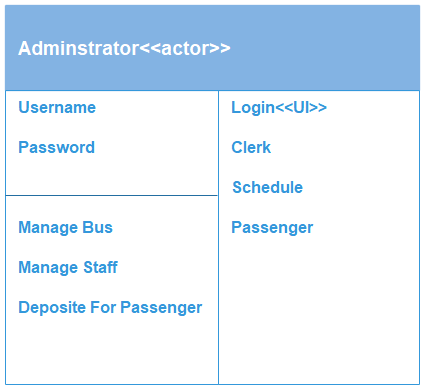


Table 2. CRC for Passenger

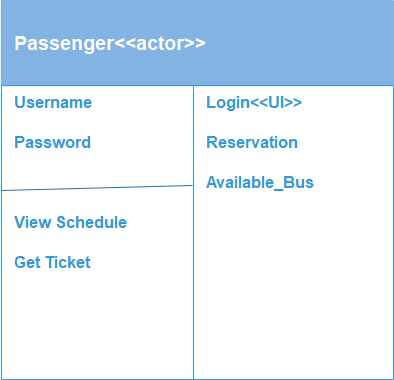
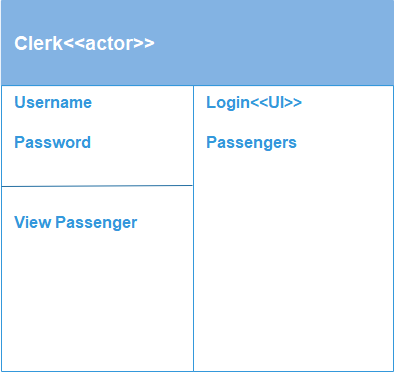


Table 2. CRC for Clerk



### 2.3.4 Sequence Diagram

Sequence diagrams are an easy and intuitive way of describing the behavior of a system by viewing the interaction behavior the system and its environment. A sequence diagram shows an interaction changed in a time sequence. It shows the object participating in the interaction by their lifelines and the messages they exchange arranged in time sequence. Generally, this diagram shows a succession of interaction between object instances over time.

**A sequence diagram has two dimensions:**

1. Vertical dimension –represent time.

2. Horizontal dimension represents different objects.

The vertical line is called the object life line. The life line represents the objects during the interaction an object is shown as a box at the top of a dashed vertical line. However, sequence diagram does not show the relationship among the roles (or) the association among the objects. Each message is represented by an arrow between the lifelines of two objects. The order in which these messages occur is shown top to bottom on the page. Each message is labeled with the message that an object sends to itself by sending the message arrow back to the same lifeline.

The sequence diagram is very simple and has immediate visual object appeal. This is its great strength. It is an alternative way to understand the overall flow of the control of a program. Instead of locking at the code and trying to find out the overall sequence of behavior, we can use diagram to quickly understand the sequence.

**Sequence Diagram for Login**

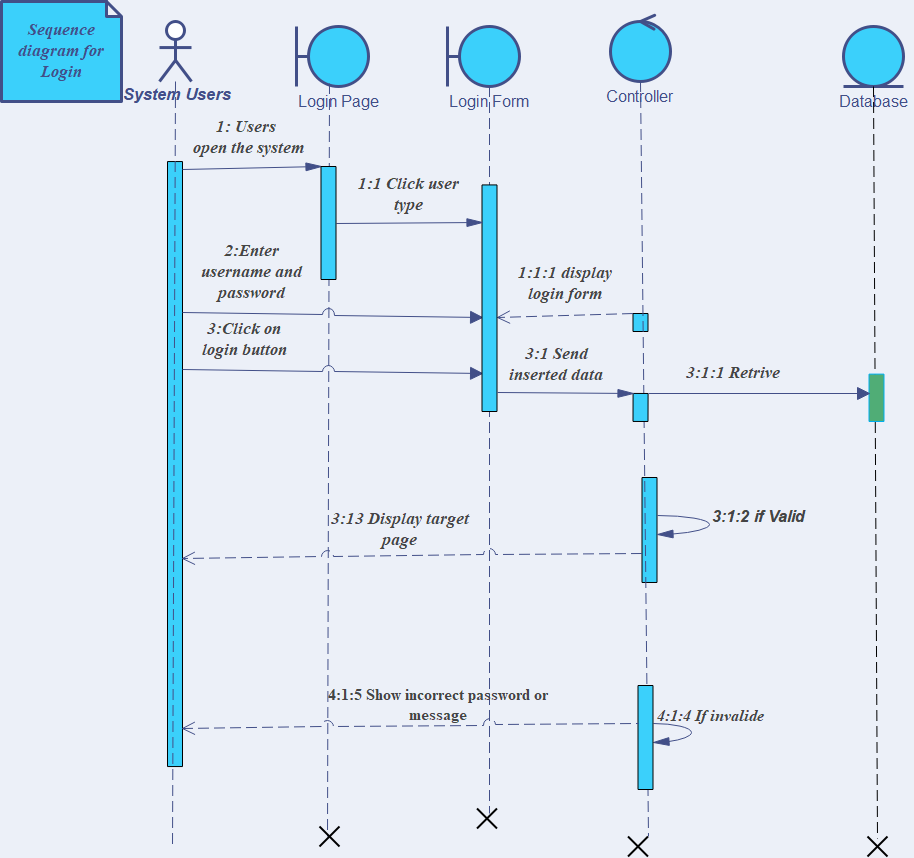


Figure 2. Sequence diagram for login

**Sequence Diagram for Get Ticket**

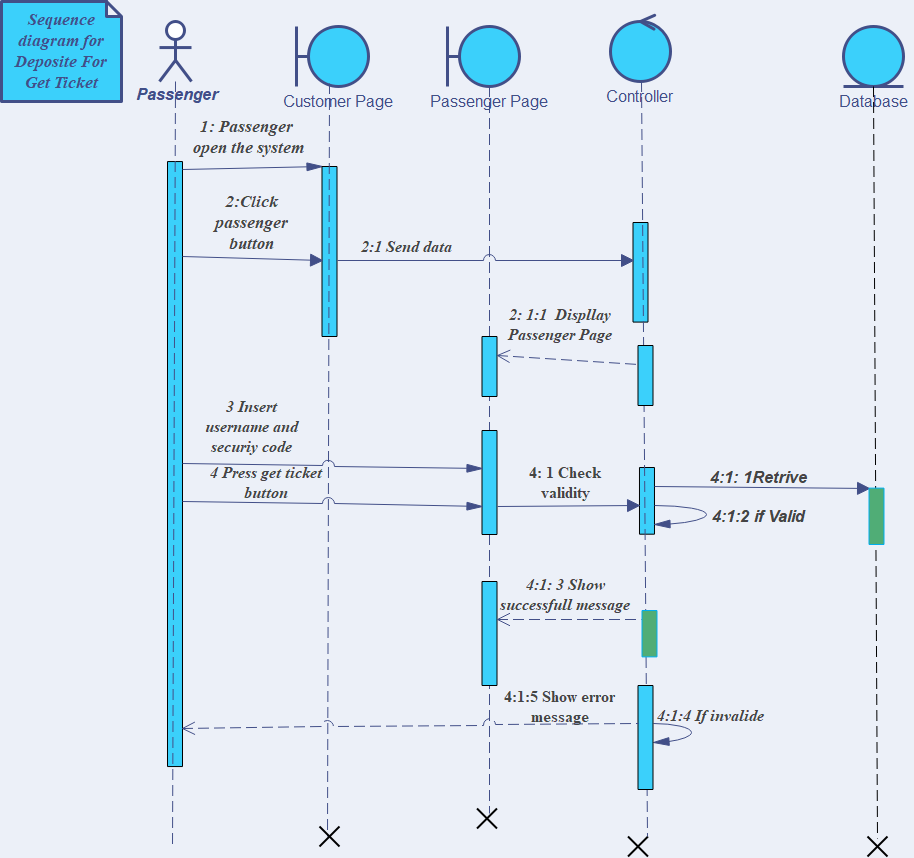


Figure 2. Sequence diagram for get ticket

**Sequence Diagram for Deposit for Passenger**

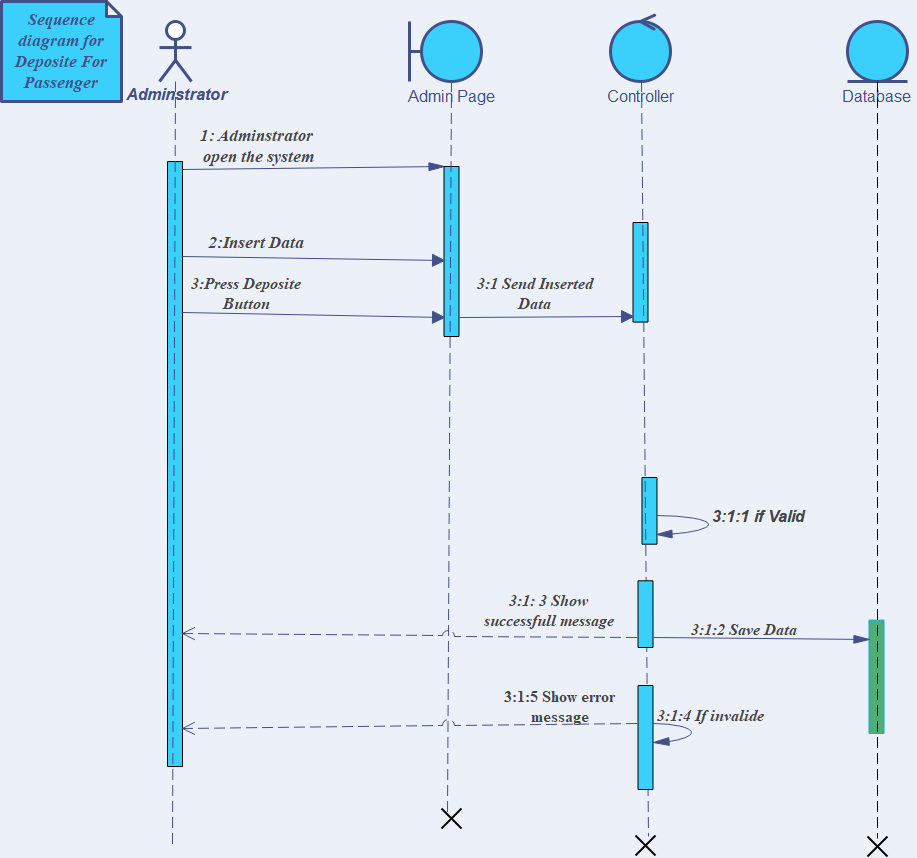


Figure 2. Sequence diagram for Deposit For Passenger

### 2.3.5 Activity Diagram

An activity diagram is essentially a flowchart, showing flow of control from activity to activity it involves:

* modelling the sequential (and possibly concurrent) steps in a computational process
* modelling the flow of an object as it moves from state to state at different points in the flow of control.
* Activity diagrams emphasize the flow of control from activity to activity.

**Activity Diagram for Admin**

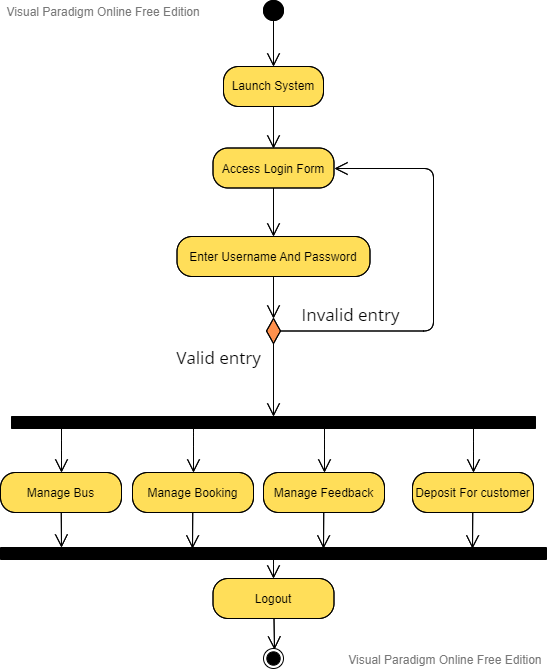


Figure 2. Activity diagram for admin

**Activity Diagram for Passenger**

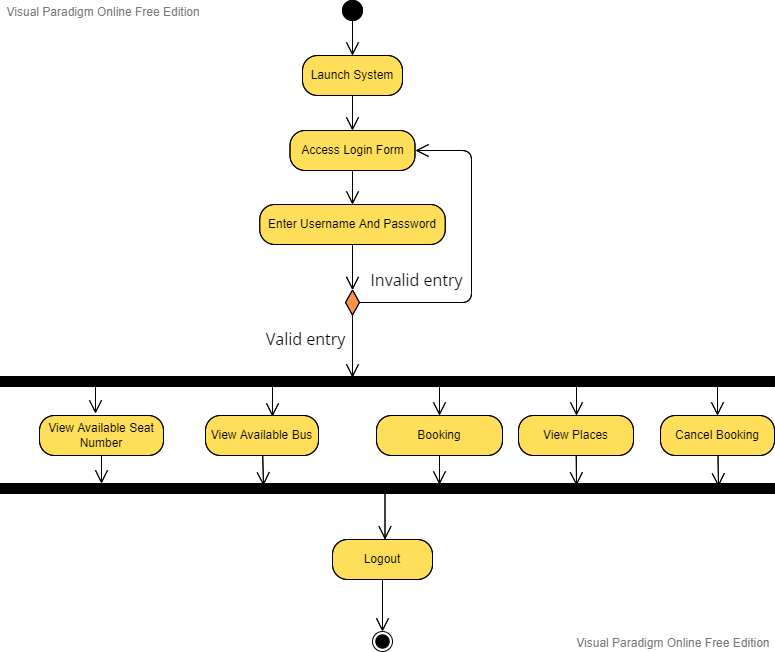


Figure 2. Activity diagram for passenger

**Activity Diagram for Clerk**

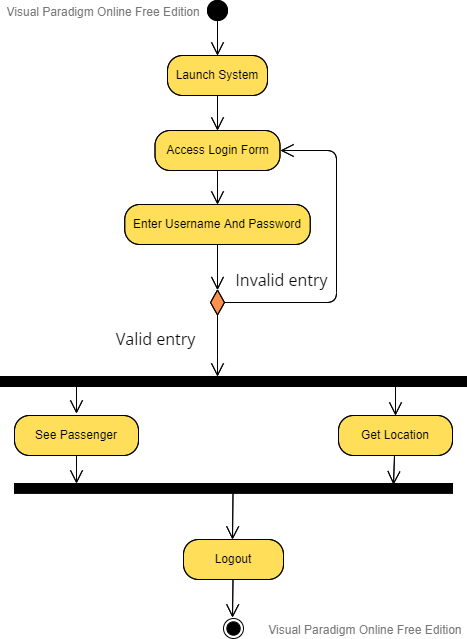


Figure 2. Activity diagram for clerk

### 2.3.6 Class Diagram

Class diagrams are used to describe the structure of this system. Classes are abstractions that specify the common structure and behavior of a set of objects in the new system.

It is a static model that shows the classes and the relationships (including inheritance and association) among classes and the operation and attributes of the classes.

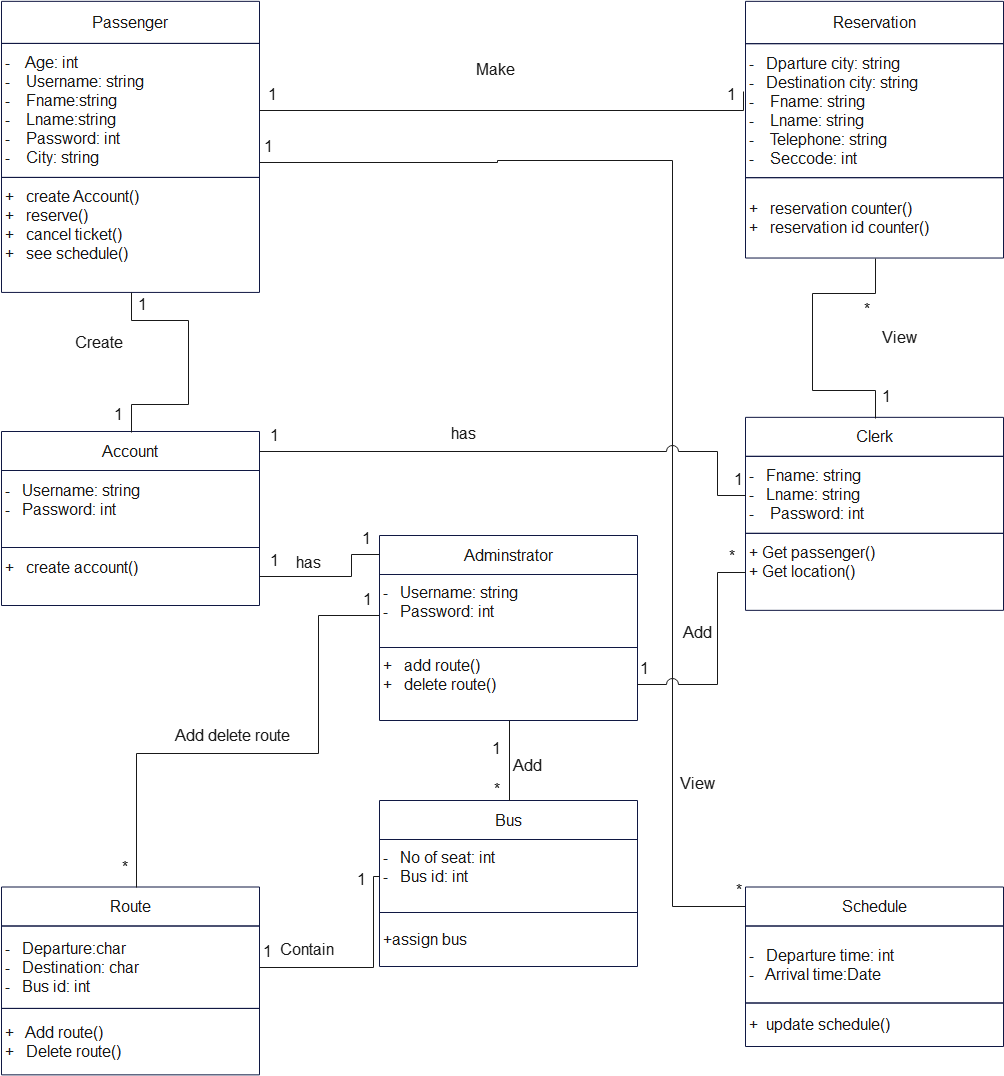


Figure 2. Class diagram

### 2.3.7 User Interface Prototyping

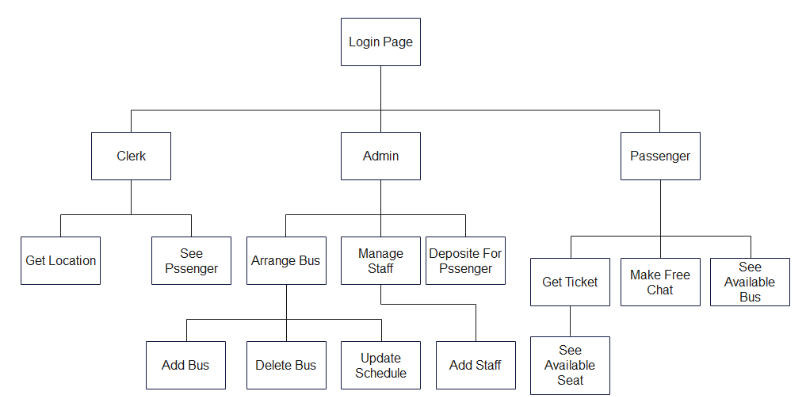


Figure 2. User interface prototyping

### 2.3.8 Identifying Change Cases

This application needs changes due to the following cases: -

* It needs modification as new Android devices are developed to enable it to run on newer versions of Android devices. The maximum SDK requirement should be updated up in the Android manifest code. In addition to modification in SDK, it may be crucial to include new libraries to add new features.
* To add speech recognition function using Amharic language.
* To add different payment methods.

# Chapter Three: System Design

## 3.1 Introduction

**S**ystem design is the transformation of the analysis model into a system design model. In the analysis phase the team describe the system completely from the actor point of view and serves as the basis of communication between the client and the developers. But the analysis does not contain information about the internal structure of the system and its hardware configuration. In generally how the system should be realized. So in system design phase the team describe in detail about the proposed system architecture, current software architecture, design goals and at last the services of subsystem.

## 3.2 Design Goals

The design goals describe the qualities of the system that developers should improve. Design goals are normally derived from the non-functional requirements of the system. So the followings section describe the design goals of the system:

**Performance**

* **Response time**: the response time for searching definitions should be fast. To attain this goal, coding should follow simple, efficient and fast algorithms. Especially looping statements shall be avoided (if possible) or minimized. For example, instead of using loop for retrieving a specific record form a table use SQL statement.
* **Memory**: Since the memory of mobiles is small in size, size of the application should not be large. To attain this store multimedia parts like image and pronunciation or audios on file server and access them through internet

**Dependability**

* **Reliability**: The system shall be tested after and during development process to verify that whether the specified services are available or not. And also it should give consistent and correct output for various type of input it is given.
* **Availability**: The multimedia contents shall be deployed on a file server that is available 24hrs. Therefore, sample care shall be taken when selecting the deployment server.
* **Fault Tolerance**: Error handlers and Record Set control objects shall be used to handle and tolerate fault that may arise during execution. For example, end of Record Set shall be checked during traversing.

**Maintenance**

* **Modifiability**: In order to make the system flexible and easy to modify its functionality some of its feature shall be modifiable.
* **Portability**: The application shall be developed to run on android environment therefore it is not portable to run on any environment which is not android enabled.
* **Readability**: Coding shall incorporate significant and clear comments describing each part of the code what it does and what its purpose is.
* **Traceability of Requirement**: The system shall be developed according to the system analysis and the design specification and it shall not incorporate anything beyond. Otherwise newly incorporated functionality shall be documented.

**End User**

* **Usability**: The application shall be developed to be easy for user understanding. Especially in developing the user interface it is better to keep the user need in mind.

## 3.3 Design Class Diagram

The class diagram represents the static view of an application. Class diagram is not only used for visualizing, describing and documenting different aspects of a system but also for constructing executable code of the software application. The class diagram describes the attributes and operations of a class and the constraints imposed on the system. The classes diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. The class diagram shows a collection of classes, interfaces, associations, collaborations and constraints. It is also known as structural diagram for this the team developed the following class diagram.

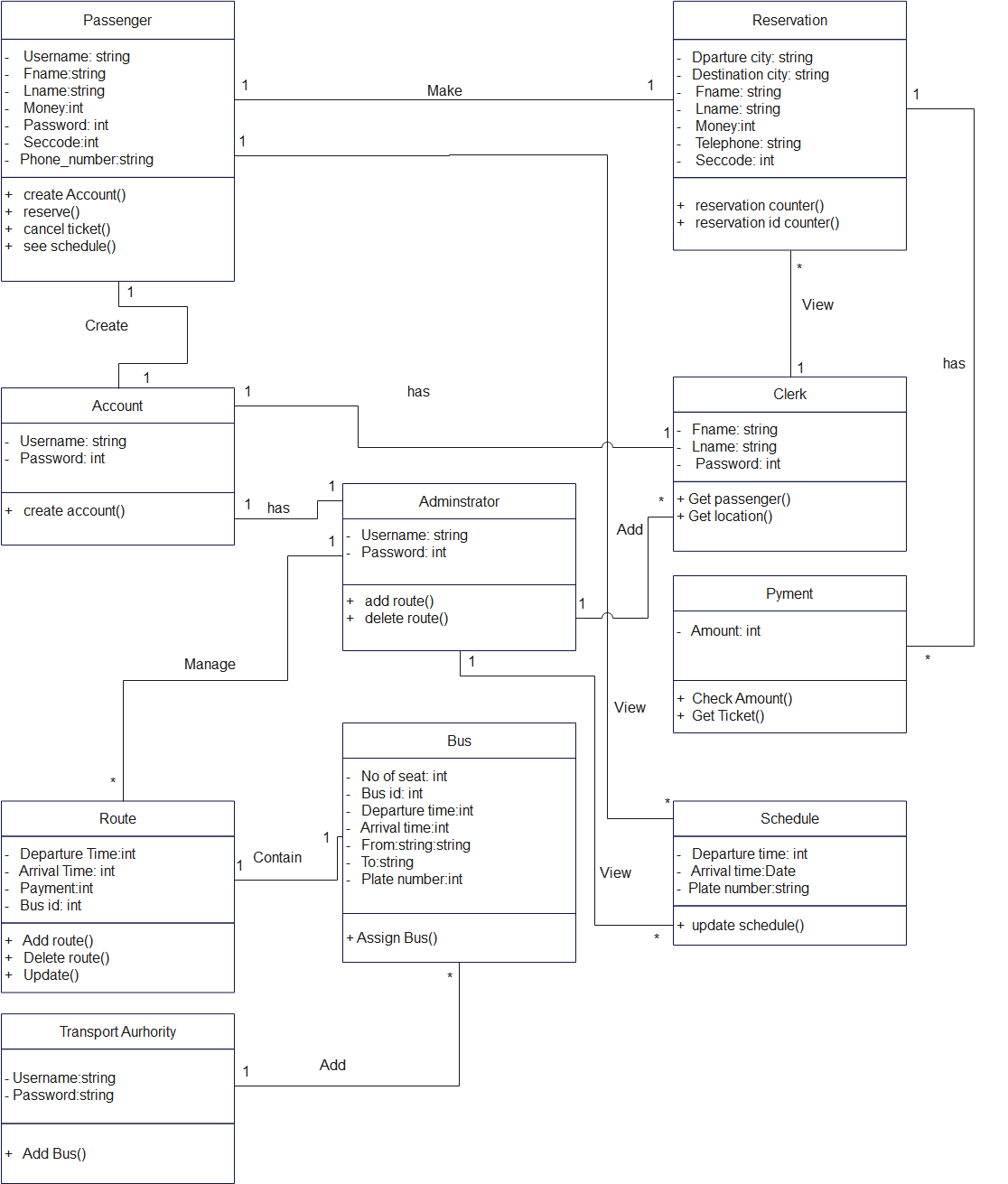


Figure 3. Design Class Diagram

## 3.4 Current System Architecture

Currently in Ethiopia bus stations uses a manual ticket reservation system. So that the manual system does not have any software architecture.

## 3.5 Proposed System Architecture

The proposed system is expected to replace the existing manual system by android-based ticket reservation which the software architecture used for the system is Repository architecture because subsystems access and modify data from a single data structure which is called the central repository. This architecture allows different user of the system to access the data from center database server.

The central repository of the proposed system is MySQL database server which is every data related with the system is stored.

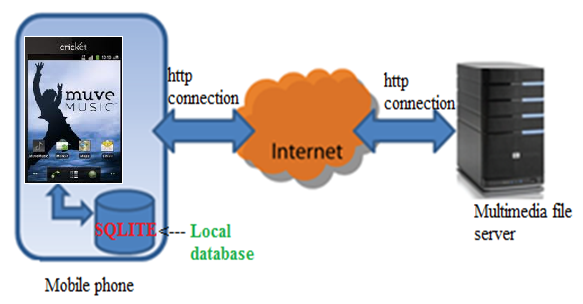


Figure 3. Software Archetecture

### 3.5.1 Subsystem Decomposition

Subsystem decompositions will help us to reduce the complexity of the system. So the team identifies the following subsystem from the main system:

**User Management Subsystem**

The user management subsystem controls the account of the system user.

The user here is a staff of bus station that would be allowed to use the system whose authorization properties would determine his/her type of access. This subsystem will allow for;

* Addition of new users
* Removal of users from get ticket list

**Bus Management System**

This manages the bus schedule and fare details for each service. This subsystem allows for the following;

* Assign buses to a route.
* Modification of bus fare.

**Reservation Management System**

Which provide and control reservation and cancelation of ticket and also uses to check the availability of ticket. It performs the following functions;

* Receives reservation request and send tickets for passengers.
* Cancel passengers’ ticket.
* Show available seat to passengers.

**Account Management Subsystem**

This subsystem is responsible for managing user account. It provides function for add user account, Administrator and Passenger is the only actor who can communicate with this sub system. This sub system uses the storage sub system for storing account’s information.

The operations provided by this sub system are:

* Create account ()
* Recover account ()

**Route Management Subsystem**

The route subsystem uses to control and manage addition and deletion of route.

**Schedule Management Subsystem**

Schedule subsystem which uses to control and manage the schedule of the bus.

**Database management subsystem**

This subsystem is responsible for managing all the relevant information’s stored in the database

### 3.5.3 Component diagram

Systems may be built from components in component-based architecture. Component diagram shows how objects (classes) in your system will grouped together and form components. The components interact with each other either in giving service to other components or requesting service from another component. Component diagrams are particularly useful with larger teams.

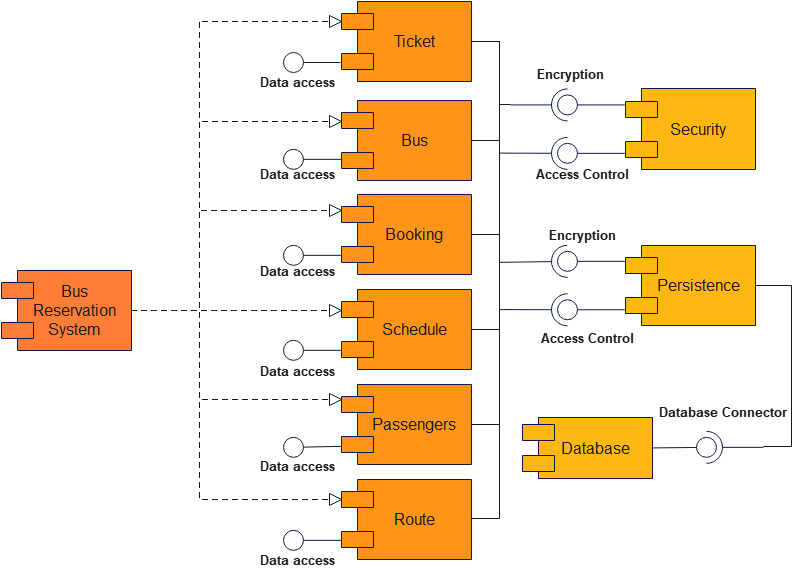


Figure 3. Component Diagram

### 3.5.4 Deployment diagram

Deployment diagram show how the system will deployed on computers. In other words, it shows which component of the software will installed on which machine and how they communicate with each other if they are on different machines. Deployment diagrams can also be created to explore the architecture of embedded systems, showing how the hardware and software components work together. You want to create a deployment diagram for applications deployed to several machines.

It is used to show the relationship among run-time components and hardware nodes. It is representing the allocation of components to different nodes and the dependencies among components.

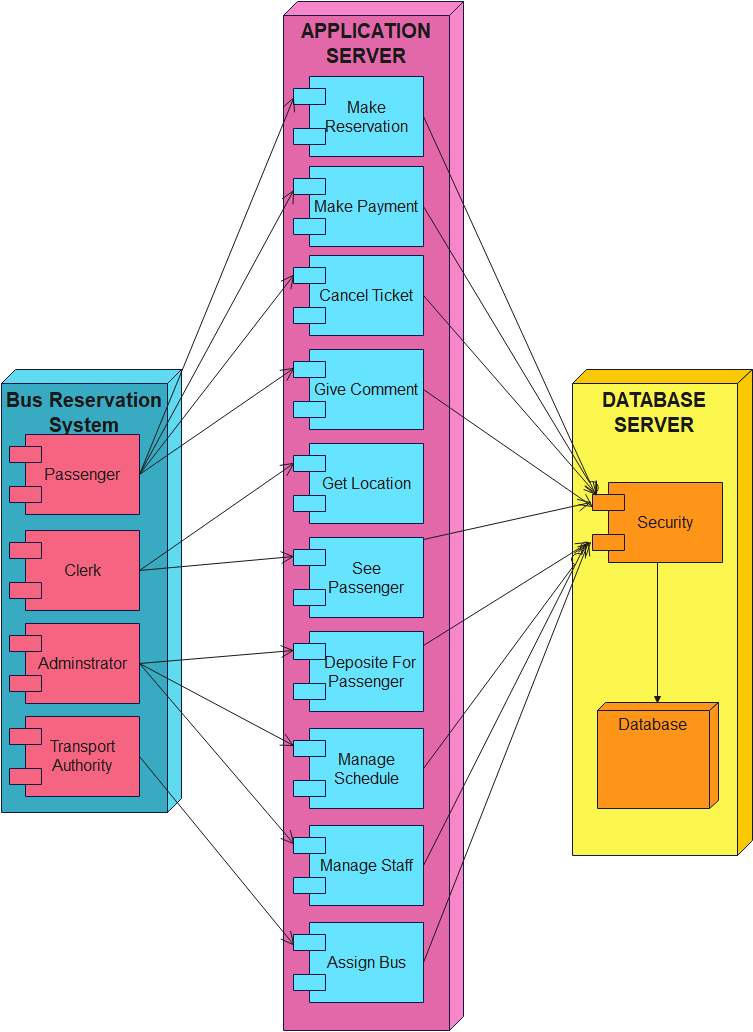


Figure 3. Deployment Diagram

### 3.5.5 Persistent data management

In this section the team describes how the persistent data stored by the system and the data management infrastructure required for it. The system will use the MYSQL database server for storing data. This will allow the database to be easily integrated with and accessed by the rest of the system. The database will retain passenger information (name, password etc.), and also retain configuration data such as authorized administrator. Each of these items will be store in a separate table. These tables below provide the complete database table details such as **Field Name, Descriptions, data types**, and **character lengths**.

**Table Name: Passenger**

Table 3. Passenger Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Length** |
| Sec Code (PK) | Password | Int | 11 |
| NAME | Full Name | Varchar | 255 |
| AMT | Amount of Money | int | 11 |
| PHONE | Phone Number | text | 100 |
| FF | Favorite Food | text | 100 |
| FC | Favorite City | text | 100 |

**Table Name: Bus**

Table 3. Bus Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Length** |
| SK (PK) | Schedule Key | Int | 100 |
| PN | Plate Number | Int | 100 |
| BS | Bus Seat | Varchar | 100 |
| FRM | From | text | 100 |
| TOO | To | text | 100 |
| PAYMENT | Payment | int | 100 |

**Table Name: Reservation**

Table 3. Reservation Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Length** |
| R\_ID (PK) | Reservation Id | Int | 11 |
| P\_ID (FK) | Passenger Id | Int | 11 |
| B\_ID | Buss Id | Int | 11 |
| FRM | From | text | 100 |
| TOOO | To | text | 100 |
| SEAT NO | Seat Number | Int | 11 |

**Table Name: Blind**

Table 3. Blind Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Length** |
| ID (PK) | ID | Int | 11 |
| FRM | From | Varchar | 255 |
| TOOO | To | Varchar | 255 |
| LAT | Latitude | Int | 100 |
| LONG | Longitude | Int | 100 |

**Table Name: Payment**

Table 3. Payment Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Length** |
| **P\_ID (PK)** | Payment ID | Int | 11 |
| **C\_ID (FK)** | Customer ID | Int | 11 |
| **R\_ID (FK)** | Reservation ID | Int | 11 |
| **P\_DATE** | Date of Payment | Date | 100 |

**Table Name: Clerk**

Table 3. Clerk Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Description** | **Type** | **Length** |
| **Password (PK)** | Password | Int | 11 |
| **Username** | Username Name | Varchar | 255 |

**Relationships among Tables**

This part is to describe and show the necessary relationships among the tables, which are selected to store the data persistently in the system.

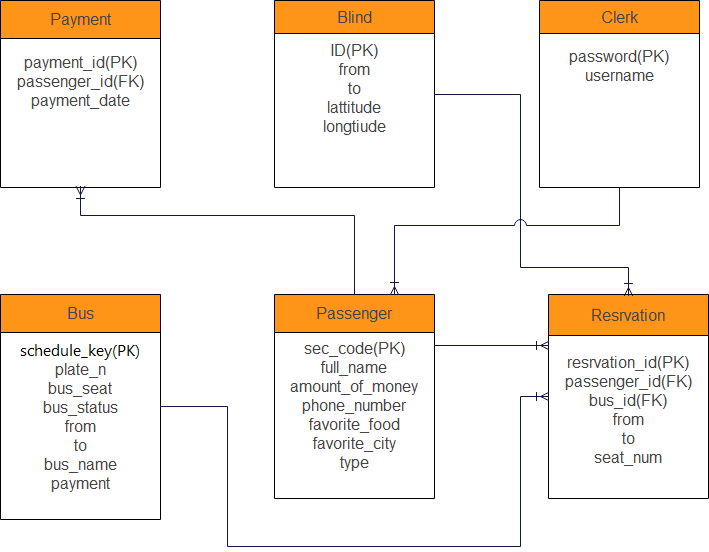


Figure 3. Relationship Among Tables

### 3.5.6 Access Control and Security

Many levels of security protect sensitive documents and files from unauthorized viewers. Each user has a security access level and each document has a sensitivity level. Depending upon the access level of the user, they will see only the list of documents that is appropriate for their security access level. Generally, all users have their own user names and passwords to control security access levels and document sensitivity level.

The system accessed by different account levels:

* The administrator has been guaranteed to update schedule, generate report, add and delete route and assign route to bus.
* The transport authority is guaranteed to assign new bus to the bus station.
* The ticket clerk has been guaranteed to review reservation and get the location of passengers.
* The passenger has been allowed to reserve a seat, check availability of bus, view schedule cancel reserved ticket.

### 3.5.7 Subsystem Service

As mentioned in the analysis phases the system provide different services so below are those services which is provided by subservices listed:

* Assigning bus to the route
* Updating and reviewing schedule
* Adding and deleting route
* Reserving and canceling of ticket

### 3.5.8 Boundary conditions and Exception Handling

#### 3.5.8.1 Boundary Conditions

Here it will try to describe some of the objects that act as a boundary object and exception handling mechanisms. The System Administrator initiates the App using the appropriate administrator account that enables him/her to add, modify and/or remove data available on the system such as, the bus station data, bus information, schedule, payment and others. It also enables to take the necessary backups from it for recovery and other essential purposes in case of system crash. Of course, the server side is giving 24/7 service unless it gets some problem that makes it down.

Any customer initiates the system to get a connection with the app using his/her mobile phone. As a result, after login the page administrator page, or clerk page, or passenger page, or transport authority page will be displayed as a boundary object to provide different services and access the pages available there. Based on the functionality the customer is interested, there are a number of boundary objects found there so as to facilitate the connection between the customer, the system administrator tasks. In addition to the administrator page, the following are some of the boundary objects found in this specific system. Passenger Page, Clerk Page, Transport authority page, get ticket page, available bus page, Login page and comment Page.

The System is a Client–Server architecture and allows a remote access. The following requirements are mandatory on both Client and Server side. For example:

###### **Client slide**

* Internet connection should be available on the client side
* Application is demanding to connect with the web server of the system
* The customer should be legitimate and having an account provided by the system
* The customer communicates the different pages using the passenger page.
* The Customer can get different service from viewing the available bus up to getting ticket and making payment.

###### **Server Side**

* The system administrator initiates and updates the system using his/her preferred privileges
* The server side should be registered on the local or any other service provider.
* It should have Internet connection and database driven-file for remote access.

#### 3.5.8.2 Exception Handling

* The system will display incorrect username or password message if it is tried to access using wrong/invalid account by checking against the account table.
* The passenger can’t get ticket that are not available in the schedule.
* If the passenger enters random security code, it checks and gives an incorrect message.
* It uses trusted web hosting servers in order to prevent data lose in case of system crashes.
* There will be a recovery mechanism so as to save temporary states in the case of network link failure.

## 3.6 User Interface Design

User interface design requires a good understanding of user needs. The goal of user interface design is to make the user's interaction as simple and efficient as possible, in terms of accomplishing user goals what is often called user-centered design. The following interface design pictures describe the logical characteristics of some interfaces between the system and the users. So, the sample interfaces are shown as follows.

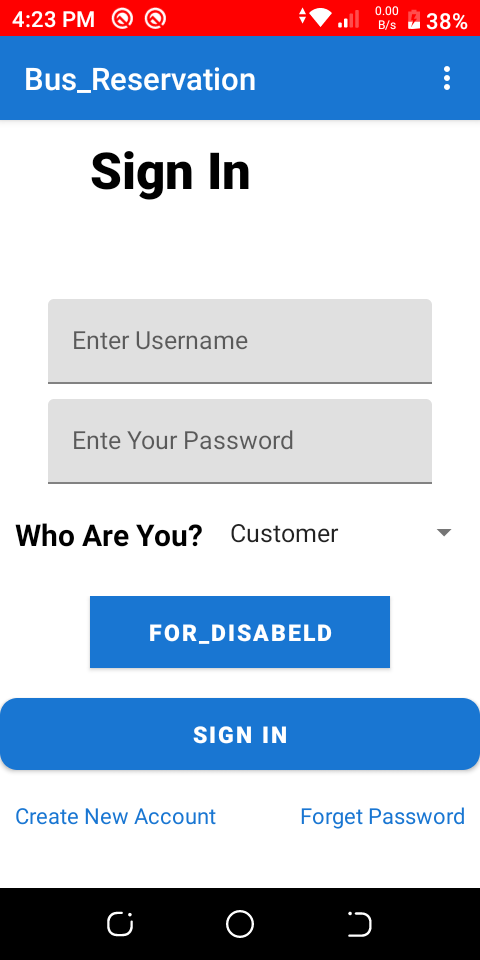


Figure 3. Login Page

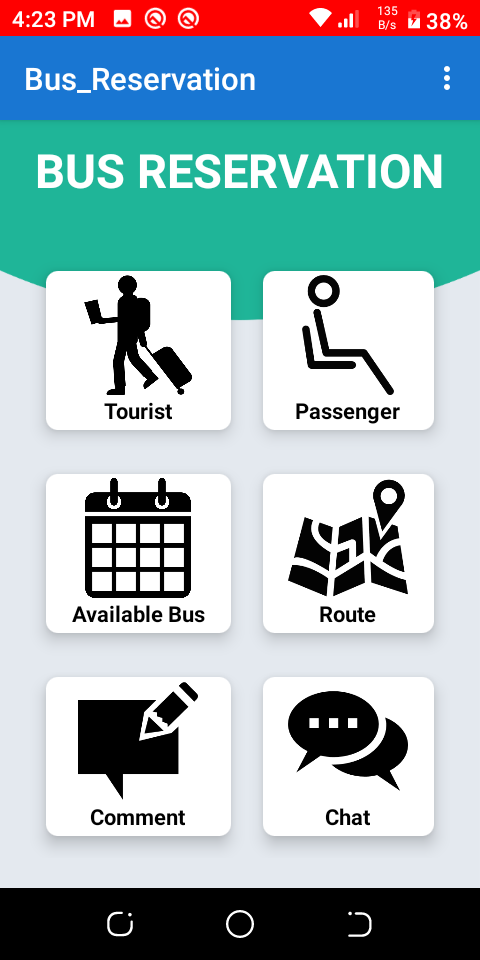


Figure 3. Passenger Page

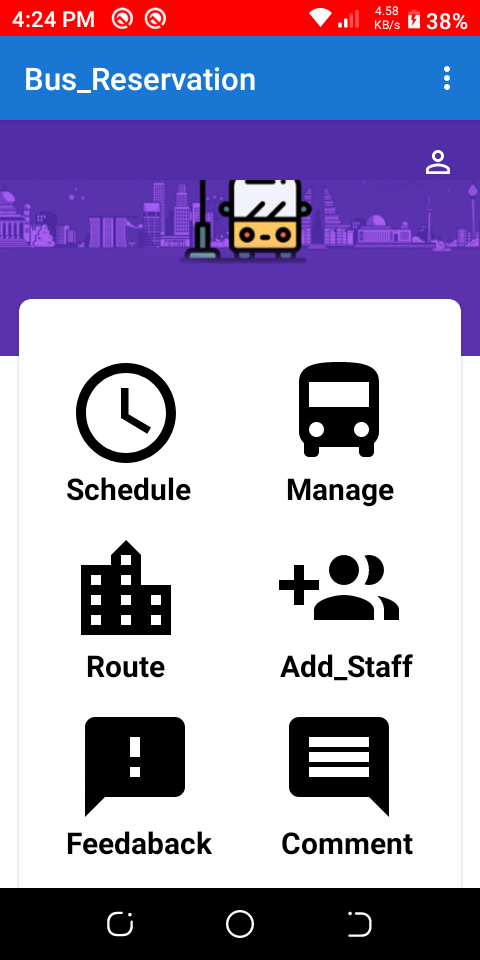


Figure 3. Administrator Page

# Chapter Four: Implementation

## 4.1. Introduction

Implementation in the system includes implementing the attributes and methods of each object and integrating all the objects in the system, to function as a single system the implementation activity spans the gap between the detailed objects designed model and a complete set of source code file that can be compiled together.

Implementation is defined as a specified set of activities designed to put into practice an activity or program of known dimensions. It is the carrying out, execution, or practice of a plan, a method, or any design, idea, model, specification, standard or policy for doing something. It is also a realization of a technical specification or algorithm as a program, software component, or other computer system through computer programing and deployment. It

## 4.2 Overview of the programming language used

This project used JAVA, PHP. PHP is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into Preprocessor (PHP) is a server-side scripting language designed for web development.

## 4.3 Algorithms Used

The project used **Random algorithm** andwe use **MD5** technique for encryption purpose.

* **MD5**: Stand for **Message-Digest Algorithm** and it is a cryptographic hash function whose main purpose is to verify that a file has been unaltered which is encrypted. MD5 has certain flaws and so it isn't useful for advanced encryption applications, but it's perfectly acceptable to use it for standard file verifications.

**4.4 Sample Codes**

**Sample code for view available bus**

package com.example. project\_beguin\_1;

import androidx.appcompat.app.AppCompatActivity;

import androidx.recyclerview.widget.LinearLayoutManager;

import androidx.recyclerview.widget.RecyclerView;

import android.content.Context;

import android.content.Intent;

import android.os.AsyncTask;

import android.os.Bundle;

import android.widget.RelativeLayout;

import android.widget.Toast;

import com.android.volley.AuthFailureError;

import com.android.volley.Request;

import com.android.volley.RequestQueue;

import com.android.volley.Response;

import com.android.volley.VolleyError;

import com.android.volley.toolbox.StringRequest;

import com.android.volley.toolbox.Volley;

import com.example.project\_beguin\_1.MyListAdapter;

import com.example.project\_beguin\_1.MyListData;

import com.example.project\_beguin\_1.R;

import org.apache.http.HttpResponse;

import org.apache.http.client.HttpClient;

import org.apache.http.client.methods.HttpGet;

import org.apache.http.impl.client.DefaultHttpClient;

import org.json.JSONArray;

import org.json.JSONObject;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.URI;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Date;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

public class Availbus extends AppCompatActivity {

List<MyListData> productList;

private MyListAdapter mMyAdapter;

RecyclerView recyclerView;

String st;

ArrayList<Integer> inte;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.availbus);

recyclerView = findViewById(R.id.recyclerView);

recyclerView.setHasFixedSize(true);

recyclerView.setLayoutManager(new LinearLayoutManager(this));

productList= new ArrayList<>();

new Connection().execute();

}

class Connection extends AsyncTask<String, String, String> {

@Override

protected String doInBackground(String... strings) {

String result = "";

String host = "https://muhamedd823.000webhostapp.com/available.php";

try {

HttpClient client = new DefaultHttpClient();

HttpGet request = new HttpGet();

request.setURI(new URI(host));

HttpResponse response = client.execute(request);

BufferedReader reader = new BufferedReader(new InputStreamReader(response.getEntity().getContent()));

StringBuffer stringBuffer = new StringBuffer("");

String line = "";

while ((line = reader.readLine()) != null) {

stringBuffer.append(line);

break;

}

reader.close();

result = stringBuffer.toString();

} catch (Exception e) {

Toast.makeText(Availbus.this, e.getMessage(), Toast.LENGTH\_SHORT).show();

}

return result;

}

@Override

protected void onPostExecute(String result) {

try {

int jj;

JSONObject jsonresult = new JSONObject(result);

int success = jsonresult.getInt("success");

if (success == 1) {

JSONArray cars = jsonresult.getJSONArray("cars");

for (int k=0;k<cars.length();k++){

JSONObject car = cars.getJSONObject(k);

String ca=car.getString("to\_t");

String fr=car.getString("from\_t");

if(ca.equals("")){

continue;

}

productList.add(new MyListData(

"From: "+fr,

"To:"+ca,

R.drawable.bu

));

}

MyListAdapter adapter = new MyListAdapter(productList);

recyclerView.setHasFixedSize(true);

recyclerView.setLayoutManager(new LinearLayoutManager(Availbus.this));

recyclerView.setAdapter(adapter);

} else {

Toast.makeText(Availbus.this, "no car", Toast.LENGTH\_SHORT).show();

}

} catch (Exception e) {

}

}}

}

**4.3 Choose your Testing Approach**

**Blackboxtesting*: -***To test our system, the tester may use black box testing, if he/she has not enough time to check internal modules or codes. By looking only input /output or user interface, the tester can test our systems functionalities without looking the internal code. We used this testing technique for the following reasons: -

* This testing type is more effective on larger units of code
* Tester needs no knowledge of implementation, including specific programming languages
* Tester and programmer are independent of each other
* Tests are done from a user's point of view

**White box testing: -** In this type of testing, skilled man in different programming languages tries to test the logic of our system. If the person who tests the system is not skilled, it is difficult to understand our systems functionality. If any failures occur while testing the system in all of the above testing methods, the team will take immediate correction where this fault occurred before jumping to next work. So, that it will meet the goal.

## 4.4 Functional Test Specifications

|  |  |
| --- | --- |
| **Test Case 1:** Get Ticket | |
| **Test case objective**: To Get Legitimate Ticket | |
| **Test case description:** users insert some required data, then presses get ticket button. Client program contacts with server, server contacts with the database, and database checks for account and sends successful message. | |
| **Requirements Verified:** Yes | |
| **Test Environment:** Database Should contain appropriate table and link must be established between server and client program. | |
| **Test Setup/Pre-Conditions:** all fields should be filled. | |
| **Actions** | **Expected Results** |
| The user inserts required data and press the get ticket button | Displays success message/ get verified ticket. |
| If some fields are not filled the system displays to fill the fields again/ error message. | |

|  |  |
| --- | --- |
| **Test Case 2**: Deposit | |
| **Test case objective**: To deposit money for passengers. | |
| **Test case description:** The administrator enters required data, then presses deposit button.  Client program contacts with server, server contacts with the database, and database checks for entered data and save the data. | |
| **Requirements Verified:** Yes | |
| **Test Environment:** Database Should contain appropriate table and link must be established between server and client program. | |
| **Test Setup/Pre-Conditions:** fields should be filled correctly. | |
| **Actions** | **Expected Results** |
| The administrator should fill all the input sections. | Display successful message. |
| If the field’s not fill correctly the system displays error message. | |

# Chapter 5: Conclusion and Recommendation

## 5.1 Conclusion

This project may rectify many disadvantages in ticket collecting system such as fare is debited from the recharged amount, hence cash is no longer necessary and also passengers no longer need to go to the bus station for asking schedule. The tickets are generated directly on passenger’s mobile phones so smart card usage and paper tickets are eliminated. On that account it will make the passenger comfortable to travel with this User-friendly system.

## 5.2 Recommendation

Research and development are continuous processes; this is the same in computer and software development. However, this work is recommended for different Transport Company, since their operation are still carried out manually. The system can contribute more on those bus staffs handling the account if it can generate reports by trip so that they will no longer go to a certain module to check the reservation and its details. Also, it will be more beneficial to both clients and bus representatives if clients can create an account just like in airlines websites. With that, the system can record the modifications made.

# Reference

|  |  |
| --- | --- |
| [1] | M. Oloyede, "Assignment 2 SAD," 17 April 2018. [Online]. Available: https://studylib.net/doc/25492048/assignment-2-sad. [Accessed 11 June 2022]. |
| [2] | C. Gyg, "Dummies," 6 July 2022. [Online]. Available: https://www.dummies.com/article/business-careers-money/business/project-management/how-to-write-a-problem-statement-for-six-sigma-170197/. [Accessed 1 July 2022]. |
| [3] | H. Belani, "ReserachGate," 2009. [Online]. Available: https://www.researchgate.net/publication/224611155\_Implementation\_and\_evaluation\_of\_  mobile\_ticket\_validation\_systems\_for\_value-added\_services. [Accessed 5 july 2022]. |
| [4] | Sanja, implementation of mobilt tcket system, harvoge belani, 2009. |