****

DEBRE MARKOS UNIVERSTIY BURIE CAMPUS

DEPARTMENT OF COMPUTER SCIENCE

PROJECT TITLE: ONLINE VEHICLE MANAGEMENT SYSTEM FOR BURIE BUS STATION

**Prepared by:**

**NAME ID NO**

1. **ENKUHANE BELAYNEH…………….. ………..048**
2. **GETANEH ADDISIE ……………. ..………… 050**
3. **GETANEH SHBABAW……………………. …. 051**
4. **GETASEW ADUGNAW ……………………. 052**

**Advisor name: msc Andargie M.**

**February, 2011E.c**

**BURIE, Ethiopia**

Table of Contents

[List of tables ii](#_Toc12491766)

[List of figure iii](#_Toc12491767)

[CHAPTER ONE 1](#_Toc12491768)

[1.1. Introduction 1](#_Toc12491769)

[1.2 .Background of the project 1](#_Toc12491771)

[1.3. Statement of Problem 2](#_Toc12491772)

[1.4 .OBJECTIVE OF THE PROJECT 3](#_Toc12491773)

[1.4.1 General objective 3](#_Toc12491774)

[1.4.2 Specific objective 3](#_Toc12491775)

[1.6 .Significance of the project 5](#_Toc12491776)

[1.7. Methodology 6](#_Toc12491777)

[1.7.1. System requirement 6](#_Toc12491778)

[1 .7 .2. Data collection 9](#_Toc12491783)

[2. System analysis 10](#_Toc12491784)

[2.1. Overview of existing system 10](#_Toc12491785)

[Proposed system 11](#_Toc12491786)

[2.2. System requirement specification 11](#_Toc12491787)

[2.2.1. Functional requirements 11](#_Toc12491788)

[The system shall allow:- 12](#_Toc12491789)

[2.2.2. Non- functional requirements 13](#_Toc12491790)

[2.2.3. Business rules 14](#_Toc12491791)

[2.2.4. Change cases 14](#_Toc12491792)

[2.3 .System Requirement Analysis 15](#_Toc12491793)

[2.3.1 .Actors and use case identification 15](#_Toc12491794)

[2.3.2. Sequence Diagram 37](#_Toc12491795)

[2.3.3. Activity Diagram 44](#_Toc12491796)

[2.3.4. Analysis of class diagram 49](#_Toc12491797)

[CHAPTER THREE 51](#_Toc12491798)

[3 .system Design 51](#_Toc12491799)

[3.1. Design of Class Diagram 51](#_Toc12491800)

[3.2. User interfaces design 57](#_Toc12491801)

[**CHAPTER FOUR** 63](#_Toc12491803)

[**4 . Implementation** 63](#_Toc12491804)

[**4.1 Overview of the programming language used** 63](#_Toc12491805)

[**4.2. Algorithms used for login** 63](#_Toc12491806)

[**4.3. Sample codes** 64](#_Toc12491807)

[**CHAPTER FIVE** 73](#_Toc12491808)

[**5. Testing** 73](#_Toc12491809)

[**5.3 .**Validation testing 75](#_Toc12491810)

[**CHAPTER SIX** 75](#_Toc12491811)

[**6. Conclusion and Recommendations** 75](#_Toc12491812)

[**6.1 Conclusion** 75](#_Toc12491813)

[**6.2Recommendation** 76](#_Toc12491814)

[**6.3 Future Enhancement** 76](#_Toc12491815)

**Reference**………………………………………………………………………………………………………………………………………………78

# List of tables

[Table 1: actors table 17](#_Toc12490292)

[Table 2: Use case description of login 22](#_Toc12490293)

[Table 3: Use case description of update account 25](#_Toc12490294)

[Table 4: Use case description of delete account 26](#_Toc12490295)

[Table 5: Use case description of view vehicle information 27](#_Toc12490296)

[Table 6: Use case description of register vehicles 28](#_Toc12490297)

[Table 7 : Use case description of assign vehicle 29](#_Toc12490298)

[Table 8: Use case description of check fuel balance 30](#_Toc12490299)

[Table 9: Use case description of request maintenance 31](#_Toc12490300)

[Table 10: Use case description of request services 32](#_Toc12490301)

[Table 11: Use case description of view maintenance request 34](#_Toc12490302)

[Table 12: Use case description of submit comments 36](#_Toc12490303)

[Table 13: Use case description of view comments 36](#_Toc12490304)

[Table 14: Use case description of logout 37](#_Toc12490305)

[Table 15: table of Assign vehicle method 54](#_Toc12490306)

[Table 16: description of comment class 54](#_Toc12490307)

[Table 17: description of comment method 54](#_Toc12490308)

[Table 18: description of vehicle registration class 54](#_Toc12490309)

[Table 19: vehicle registration method 55](#_Toc12490310)

[Table 20: check fuel balance class 55](#_Toc12490311)

[Table 21: check fuel balance method 55](#_Toc12490312)

[Table 22 : Manager class 56](#_Toc12490313)

[Table 23: Manager method 56](#_Toc12490314)

[Table 24: driver class 56](#_Toc12490315)

[Table 25 : driver method 56](#_Toc12490316)

[Table 26: Send report class 57](#_Toc12490317)

[Table 27: Send report method 57](#_Toc12490318)

[Table 28:Unit testing. 73](#_Toc12490319)

[Table 29: Integrated testing 73](#_Toc12490320)

[Table 30.System testing 74](#_Toc12490321)

# List of figure

[Figure 1software process 8](#_Toc3927647)

[Figure 2: use case diagram 19](#_Toc3927648)

[Figure 3: Sequence diagram for create account 39](#_Toc3927649)

[Figure 4 :Sequence diagram for login 39](#_Toc3927650)

[Figure 5: Sequence diagram for register vehicle 40](#_Toc3927651)

[Figure 6: Sequence diagram for view vehicle information 42](#_Toc3927652)

[Figure 7 Sequence diagram for logout 43](#_Toc3927653)

[Figure 8: Activity diagram for create account 45](#_Toc3927654)

[Figure 9:Activity diagram for Login 46](#_Toc3927655)

[Figure 10: Activity diagram for register vehicle 47](#_Toc3927656)

[Figure 11Activity diagram for calculate fuel balance 48](#_Toc3927657)

[Figure 12 :analysis of class diagram 51](#_Toc3927658)

[Figure 13: Design of class diagram 52](#_Toc3927659)

[Figure 14:user interfaces home page 58](#_Toc3927660)

[Figure 15: login form 59](#_Toc3927661)

[Figure 16: create account 60](#_Toc3927662)

[Figure 17:register vehicle 61](#_Toc3927663)

[Figure 18: Deployment Diagram 62](#_Toc3927664)

Acronym

**Br Business rule**

**UI User interface**

**ID Identification**

**PC Personal Computer**

**WAMP Window, Apache, MySQL, PHP**

**PHP Hypertext Preprocessor**

**MySQL Structured Query Language**

**CD Compact Disk**

**DVD Digital Video Device**

**RAM Random Access Memory**

**CSS Cascading Style Sheet**

**HTML Hyper Text Markup Language**

**MSc Master**

**GB Giga byte**

**GH Giga Hertz**

**GUI Graphical User Interface**

**UC Use Case**

**UML Unified Modeling Language**

**VMS Vehicle management system**

# CHAPTER ONE

## 1.1. Introduction

Online vehicle management system is a system which concerned with how vehicles could be managed and controlled in a web based way by the employees of the organization. Currently, the vehicle gives service use manual way of information gathering and documenting, no reliable communication between different offices, as well as there is lack of security, difficult to coordinate, and to identify the owner of the vehicle.

This project will intend to advocate for the need of BURIE Town bus station to change the manual system to automated system vehicle management system.

Automated systems make it possible to have a better accuracy, to increase the quality of the work, to reduce the time it takes, to minimize cost, to keep the security and organization of data in most advantageous condition, to make data transfer easier and also make it possible to save and back up all transactions in case of vehicle management to keep the data in a centralized way which is available to all the users simultaneously.

## 1.2 .Background of the project

Now a day, the introduction of new information systems is increasing at an alarming rate to bring radical change to the existing manual system, improve the performance of other systems and solve difficulties. This project will intend to advocate for the need of BURIE Town bus station to use automate vehicle management system. Because there is no computerized vehicle management system in the BURIE Town bus station.

Therefore employees of the organization should have to facilitate their activity in computerized way. In BURIE Town bus station, customers use transportation or get service from BURIE Town bus station. Currently BURIE Town bus station vehicle management system use manual system like scheduling to give service, by filling written form for vehicle maintenance, putting detail information of each vehicle. While giving those activities, it stores the data in document which creates inefficiency in terms of access, track, and store information in the working environment. The vehicle give service use manual way of information gathering and documenting, no reliable communication between different offices, as well as there is no optimized way to facilitate the service. Keeping in view of all such problems, the existing system translated to automated system for information process, management and distribution. The data processing structure should be centralized and function in accordance with hierarchal structure.

## 1.3. Statement of Problem

In the current vehicle management system, information exchange and service control is process in manual way. There are various problems that the BURIE Town bus station vehicle management system faces due to the file manual handling of its daily activity.

* Not sequenced flow of information during retrieve.
* It’s a time-consuming process to assign vehicles and to generate required reports.
* Difficulty to get the required detailed information about specific vehicle.
* Employees couldn’t get high satisfaction .
* Lack of security.
* There is no effective coordination between various employers.
* The organization does not give service for user in 24 hours.

## 1.4 .OBJECTIVE OF THE PROJECT

### 1.4.1 General objective

The main objective of this project is to develop a web based Vehicle Management System for BURIE Town bus station.

## 1.4.2 Specific objective

The specific objectives of the project, which will be done in order to achieve the general objective.

* To identify existing System work flow
* To analyze the existing systems problem
* Analyze gathered information using VMS document.
* Design the proposed system.
* Design interactive user interface for vehicle management system.
* Implementing the proposed system.
* Validation and Testing the System.

**1.5. Scope of the project**

The system contains modules that can handle vehicle and service requester like:

* **Login page**: The login page contains the user name and password and users should enter the correct username password and their privilege correctly to get the page that he wants.
* **Vehicle registration:** Allow the Manager to register vehicle.
* **Manage user account:** Manager create & deactivate account for users and users can update their account**.**
* **Get exit permission:** the manager prepares exit permission for the driver.
* **Make reservation**:-the system supports the actors of the system i.e. managers to make reservation for passengers.
* **Show availability of ticket**:-the proposed system of the project is support the passengers to see availability of tickets in the system.
* **Show availability of seat**:-the system show availability of seat numbers of the passengers so the passenger see his/her own seat numbers based on the

System required personal information from users.

* **View vehicles information:** See the detail information of the vehicles.
* **Approve request services:** allows the staff (i.e. employee under officer) request service to officer permission and the officer directly request vehicle service to manager.
* **Vehicle assignation**:It allows the manager to assign car for the requested service
* **Generate and send reports:** It allows manager generating report for the tasks performed and if there is any difficult problem.
* **View notification and comment**: It allows manager to see the comments that are submitted from the driver.
* **Vehicle maintenances:** Vehicles should be maintained when they are damaged or based on schedule maintenance report.

## 1.6 .Significance of the project

After development of the new system: it gives the following advantages.

* Easy to process requests.
* Easy to manage historical data in secure manner.
* Minimizes cost of operations and the work load.
* Avoiding improper resource consumption and data loss.
* To make easy vehicle assignation.
* Easy to get the required detailed information about specific vehicle.
* Employees get high satisfaction .
* High coordination between various employers.
* To have good authentication and security.
* To make easy and fast report generate.

## 1.7. Methodology

### 1.7.1. System requirement

#### 1.7.1.1 .Hardware Requirement

* Computer: To develop the documentation as well as the implementation.
* Flash: Used as move data from one to other.
* CD**:** Used as a backup.
* Printer**:** For printing the documentation.
* Paper : To store the documents in the form of hard copy

#### 1.7.1.2. Software requirement

* **Modeling software**
* EDRAW Max.
* Used to draw diagrams like use case diagram and activity diagram.
* **Designing Software**
* MySQL server
* This software will used for designing Database.
* **Implementation Software**
* Operating System: Windows 7
* To have good speed of operations or execution time of tasks.
* Notepad or Notepad++.
* To write PHP scripts.
* **Software process**
* **Iterative Development**

Our software follows Iterative development model aims to develop a system through building small portions of all the features, across all components. In each increment, a slice of system features is delivered, passing through the requirements till the deployment.

This process is then repeated, producing a new version of the software for each cycle of the model

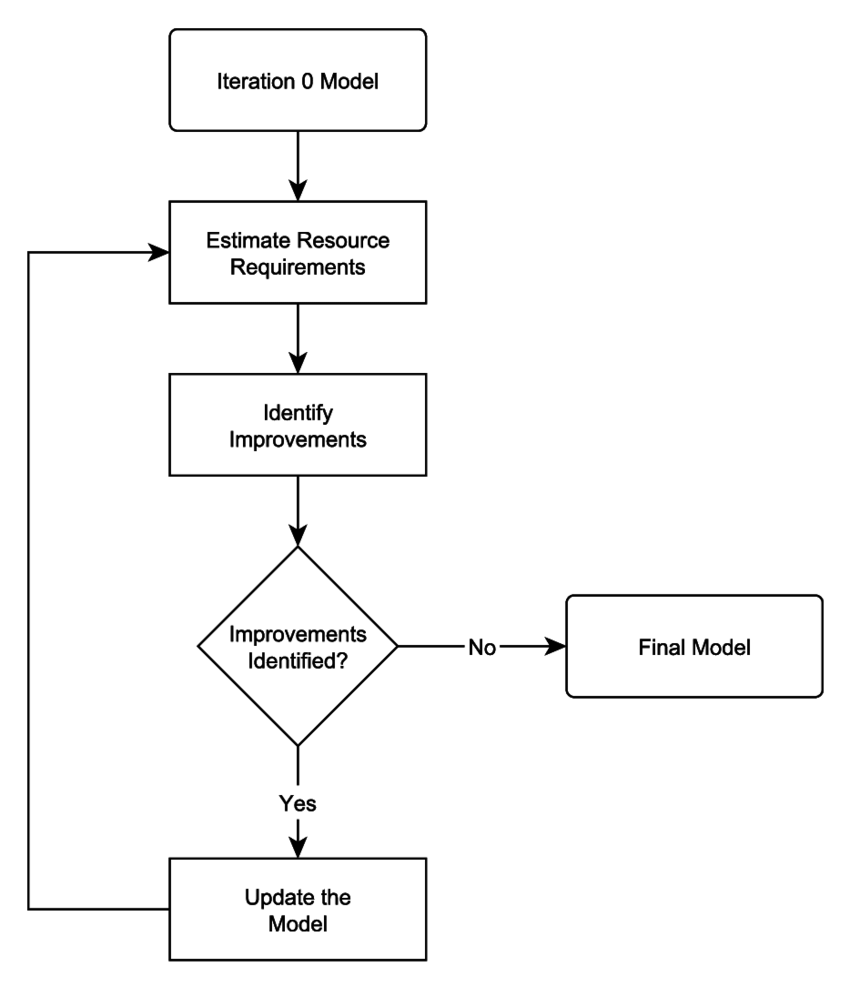


Figure 1: software process

1.7.1.3 .Programming Language

* PHP is a popular and widely used programming language which is utilized to build dynamic web applications with MySQL database connections. Because it is user Friendly and Easy to Learn, Cost user Friendly, It’s fast and easy, it accesses everything etc...
* **Back End**
* MySQL software of the data base system and PHP language was used in developing and managing the back end of the system.
* **Front End**
* The user interface had been developed using html, java script, CSS since it easily designing the front end and connected in to database easily.

### 1 .7 .2. Data collection

* **Interview**

We gathered information by asking person in vehicle management offices at BURIE bus station.

* **Document analysis**

- To get more secondary source information and ideas about the vehicle management systems, we referred project report documents and other reading materials that is helpful to develop this system.

* **Practical observation**:-

We observed all the activities that are performing and notice down how they did. It helps us to get real information how the organization performs its function and this helps to strength the data that gathered through interview and document analysis.

CHAPTER TWO

# 2. System analysis

## 2.1. Overview of existing system

* Currently the system in BURIE bus station Vehicle Management systems is use paper based documentation or manual system approach to record and report its file. There are activities and tasks that can be taking care off by the manager, driver and technician like:
* Report preparation:-Report can be prepared in the form of paper documentation.
* Document collection:-document can be collect from the prepared report according to their function and submitted time to allocated place (it may be shelf).
* Order driver:-based on the request service the manager order the driver by giving the word command or written command.
* Order technician:-The manager orders the mechanics by looking the car that need repair and maintain service for proper function.
* The officer Evaluate the requester question forward evaluated question to manager.
* Assign Car:-when the service requested, the manager arranges and assign the car to provide service.
* The staff request the car service to officer for his/her student for the Purpose of Training, trip and journey for the practical learning.
* The messenger circulates the forms and documented report from one office to another.

**Users of the existing system**

The followings are players/actors in the existing system:

* **Manager**: The listed below are the duties of the manager.
  + - Order driver
    - Give commands for mechanics according to their task.
    - Register the car
    - Calculate fuel balance
    - Prepare exit permission
* **Driver**: The duty of driver listed below
  + - Drive
    - Fill the gasoil
    - Get exit permission
    - Check the vehicle regularly
* **Officer**:
* Evaluate the requester question
* Forward evaluated question to manager.

.

* **Messenger**: the duty of the messenger includes:
* Circulate the forms and documented report from one office to another.

## Proposed system

In basis of understanding the current manual system and identifying all the problems occurred during over all activities of the existing system, the project team has decided to design an automated system model as solution. Since the client server model fully flagged online at any time, it will solve the problem and limitation of the current manual system of BURIE Town Bus Station vehicle management system.

* The administer manages all works related to the system like creating and deleting manager, driver, officer, staff and mechanics account, generate report and register employee.
* The Vehicle manager in system register vehicle, update vehicle record, view vehicles information, manage request, assign vehicle(i.e. the system give commands and response for the drivers, mechanics and officer).
* The one who drives the car, in this system the driver views notification and request maintenance record for the mechanic.
* Mechanics view maintenance request and generate maintained report.
* Staff will request vehicle service for officer and then officer request service for the manager and get the response to the system.

## 2.2. System requirement specification

### 2.2.1. Functional requirements

The functional requirements of the system describe the necessary functions for which the system is expected to fulfill. The system describes a single and well defines goal of online vehicle management system for BURIE Town Bus Station, and the steps involved to reach this goal. A function is described as a set of inputs, the behavior and output. The requirements specified are helpful to clearly understand the scope and the objective of the system, and consequently this will be helpful for designing the system effectively. The proposed system meets the following functionalities:-

### The system shall allow:-

* + - * **Account management:**
* Create account
* Update account
* Delete account
  + - * **Registration**
* Register vehicle
* **View information**
* View notification
* View maintenance request
* View comment
* View seat reservation
* Route of information
  + - * **Request service.**
* Request maintenance.
  + - * **Approve request**
* Send notification.
* get exit permission
* Give comment
  + - * **Assign vehicle.**
      * **Reserve ticket:-**
      * **Generate report.**
      * **Check fuel balance.**
      * **Send comment and View comment.**
      * **See availability of tickets.**
      * **See availability of seats.**

### 2.2.2. Non- functional requirements

Non-functional requirements that specifies criteria that can be used to judge the technical operation of a system, rather than specific characters. This should be opposes with that of functional requirement that define specific behavior.

In General,non-functional requirements define how a system is supposed to be.

* **Security**:
* **Authentication and authorization**: deals with identifying a user and what a user is allowed to do respectively.
* **Session**: once the user’s logout from the system then the session do not enable to return to the previous page
* The information provider by the user should be authentic which protect the system from external attack and spamming.
* **Performance**:
* The response time for loading and processing a given task is very fast and triggered by a single click.
* The system should provide the services in considerable time interval.
* **Reliability**:
  + - Increase reliability of the system by removing commonly made errors and feeding correct inputs for processes.
    - Provides to the user correct information.
    - The system notifies users to correct the input data when they enter wrong inputs.
    - The system will be able to meet specified objectives as well as the expectations of the customer.
* **Usability**: User interface will be user friendly, so user can familiar to the system and easy to use.
* **Accessibility**: Extent to which a consumer or user can obtain a good or service at the time it is needed.
* **Flexibility:** The system is flexible for all users when they inter the correct.

### 2.2.3. Business rules

It is the collection of rules and regulations of the vehicle management system that tells the users follow the activities.

* **Rule1**: Users of the system must have proper user name and password in order to login the system.
* **Rule2:** Users of the system must have proper user name and password in order to login the system.
* **Rule3**: The manager should check the fuel balance of the car while the car travelled.
* **Rule4:** The manager must generate report in case of some conditions occur.
* **Rule5:** Vehicles should be maintained when they are damaged or based on schedule maintenance report.
* **Rule6:** The manager should approve the request of the officer and staff.
* **Rule7:** System assigned the vehicle and automatically sends notification for driver and requester office.

### 2.2.4. Change cases

* Likely future changes (update) to either the system, in terms of its capabilities and properties are computable with the new version.
* The system will promote related international rules and regulations.
* Used to describe new potential requirement for a system or modification to existing requirements.
* Describes a potential requirement that your system may need to support in the future.

## 2.3 .System Requirement Analysis

### .Actors and use case identification

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **List Of Actors With Function** | | | | | | | |
|  | Admin | Manager | Driver | Officer | Staff | Mechanic | passengers |
| Request service |  |  |  |  |  |  |  |
| Manage request |  |  |  |  |  |  |  |
| Payment status |  |  |  |  |  |  |  |
| Assign vehicle |  |  |  |  |  |  |  |
| Manage account |  |  |  |  |  |  |  |
| Register vehicle |  |  |  |  |  |  |  |
| Submit comment |  |  |  |  |  |  |  |
| View comment |  |  |  |  |  |  |  |
| Check fuel balance |  |  |  |  |  |  |  |
| Update account |  |  |  |  |  |  |  |
| Maintenance request |  |  |  |  |  |  |  |
| View maintenance request |  |  |  |  |  |  |  |
| Route of information |  |  |  |  |  |  |  |
| Create user account |  |  |  |  |  |  |  |
| Manage account |  |  |  |  |  |  |  |
| View route of information: |  |  |  |  |  |  |  |
| Give comment |  |  |  |  |  |  |  |
| Generate report |  |  |  |  |  |  |  |
| Get exit permission |  |  |  |  |  |  |  |
| See availability of tickets. |  |  |  |  |  |  |  |
| See availability of seats. |  |  |  |  |  |  |  |
| Send report |  |  |  |  |  |  |  |
| View response |  |  |  |  |  |  |  |

Table 1: actors table

* **Use case identification**

Each Use Case describes the functionality to be built in the proposed system, which can include another Use Case's functionality or extend another Use Case with its own behavior.

The most important and basic use cases of this system are the following:-

* + Login
  + Create account
  + Delete account
  + Update account
  + Check fuel balance
  + Register vehicle
  + View vehicle Information
  + Generate report
  + View request maintenance
  + Request service
  + View Response
  + approve request
  + Assign vehicle
  + Get exit permission
  + Request maintenance
  + Send report
  + View notification
  + Sea availability seat
  + Sea availability of tickets
  + View route information
  + Payment status
  + Submit comment
  + View comment
  + Logout

**Use case Descriptions**

Manager

Driver

Officer

Mechanics

Manage

Account

Create

Register

Vehicle

Request

Maintenance

Send report

Submit comment

Delete

Account

View vehicle

Information

Assign

Vehicle

Approve

Request

Check fuel

Balance

Generate

Report

View response

View

Notification

Update account

Request service

Get exit

Permission

Login

Logout

View comment

Extend

Include

Extend

Extend

Extend

Include

Include

Include

Include

Include

Include

Include

Include

Include

Include

Include

Include

Include

Include

Include

Include

Administrator

Staff

View Request

Maintenance

Figure 2: use case diagram

**Use case description for Passenger**

Update account

Include

View route of information

Include

Give comment

Include

Payment status

Login

Include

Include

Route information

Include

Passenger

See availability of seats

Include

See availability of Tickets

Figure 3: use case diagram

Use case description includes use case Id, use case name, particular actors, descriptions of the use case, preconditions, basic course of action(actor action and system response), post conditions, flow of event and whatever which is important in modeling the user go.

|  |  |  |
| --- | --- | --- |
| **Use case name** | Login | |
| **UC ID** | UC 1 | |
| **Actor** | Administrator, Manager, driver, officer, passenger, staff and mechanics | |
| **Precondition** | They must have user account, user name and password. | |
| **Basic course of action** | User action | System response |
| 1.Users click on login button  3.User insert user name and password  4. User click login button.  6. The system displays the appropriate user page. | .  2. The system displays the login form.  5. System checks the user name and password.  7. Use case ends. |
| **Alternative course of action** | 5.1. If username and password is not validating, the system display login fail and return to basic course action 3. | |
| **Post condition** | The authenticated user gets the appropriate page. | |

Table 2: Use case description of login

|  |  |  |
| --- | --- | --- |
| Use Case ID: UC-ID2 | Use Case ID: UC-ID2 | |
| Use Case Name: | See availability of seats. | |
| Actor: Passenger | Actor: Passenger | |
| Description: | Description: The User must know detail of seat availability | |
| Preconditions: | First The user must login into the system and know the information | |
| Post conditions: | The availability of seat information must added into the database | |
| Normal Course of  Events: | User action | System action |
| 1.User opens the  documented page of  seat.  3. User clicks the view  button  6. Use case end. | 2. The system will display documented  page seats availability.  4. The system checks the correctness of  the input data.  5 The system display availability seats. |
| Alternative Courses: | A4: User enters invalid data  A4:1The system displays error messages  A4:2 Go to step 2 | |

*Table 3:use case description for availability of seats*

|  |  |  |
| --- | --- | --- |
| **Use case name:** | Manage account | |
| **Use case ID:** | UC 2. | |
| **Actor:** | Administrator | |
| **Description** | Administrator r creates account for deriver, officer, mechanics, and staff. | |
| **Pre-condition:** | Administrator open and login into the system | |
| User action | System response |
| **Basic course of action** | 1. Administrator select create user account link.   3. Administrator enters user information.  4. Administrator click submit button.  7. Use case end | 2.System display creates  Account form.  5. System checks user information.  6. System creates successfully user account. |
| **Alternative course of Action** | 5.1. The system display error messages that the input values are incorrect return to step 3. | |
| **Post-condition:** | User account is created. | |
|  |  | |

Table 4: Use case description of create account

|  |  |  |
| --- | --- | --- |
| Use Case ID: UC-ID5 | UC-ID5 | |
| Use Case Name | Use Case Name: view route information | |
| Actor | Passenger | |
| Description | The User should view the route information. | |
| Preconditions | The user must be registered and login into the system | |
| Post conditions | The route information details must added into the database | |
| Normal Course of  Events: | User action | System action |
| 1. User opens the  documented page  of route  information.  3. User fills the  form and submits.  6. Use case end. | 2. 2. The system wills the documented  information.  4. The system checks the correctness of the  input data.  5. The system display “fills successfully”. |
| Alternative Courses: | A4: User enters invalid data  A4:1The system displays error messages  A4:2 Go to step 2 | |

*Table 5:use case description of view route information*

|  |  |  |
| --- | --- | --- |
| **Use case name:** | Update account | |
| **Use case ID:** | UC 3 | |
| **Actor:** | Manager, driver, officer, staff, passenger , and mechanics | |
| **Description** | Manager, driver, officer and staff update their account | |
| **Pre-condition:** | User open and login into the system | |
| User Action | System Response |
| **Basic course of action** | 1. User select update user account link.   3. The user inserts account type, user name  e, password, and other user information.  4.User click update button  6. The systems save the new account  to the existing the database.  7.Use case end | 1. System display updates account form.   5. System checks the new account information with the existing account in database. |
| **Alternative course of Action** | 5.1. The system display error messages that the input values are incorrect. | |
| **Post-condition:** | Account is updated | |

Table 3: Use case description of update account

|  |  |  |  |
| --- | --- | --- | --- |
| **Use case name:** | Delete account | | |
| **Use case ID:** | UC 4 | | |
| **Actor:** | Administrator | | |
| **Description** | It allows the Administrator to delete user account from the system. | | |
| **Pre-condition:** | The Administrator activates and login in to the system | | |
| User action | | System response |
| **Basic course of action** | 1. Administrator Select deactivate account link.   3. Administrator select users and click deactivate button.  5. Administrator click yes button.  7.Use case end | 2.System displays  Deactivate user account information.  4.System display recommend page.  6.Systems deactivate user account | |
| **Alternative course of Action** | 5.1. If Administrator click no button, then system back into basic course of action 3. | | |
| **Post-condition:** | Account is deactivated. | | |

Table 4: Use case description of delete account

|  |  |  |
| --- | --- | --- |
| **Use case name:** | View Vehicles info. | |
| **Use case ID:** | UC 5 | |
| **Actor:** | Manager | |
| **Description** | It allows manager to see the detail information of the vehicles. | |
| **Pre-condition:** | Manager open and login into the system. | |
| User action | System response |
| **Basic course of action** | 1. Manager click on view vehicle Information link.  3. Manager fills the required fields.  4...Manager click search button.  6.Use case end | 2. The system displays the vehicle form.  5. The system displays the detail information about the vehicles. |
| **Alternative course of Action** | 5.1The system display error messages that the input values are incorrect or fill input value and display Basic course of action 4. | |
| **Post-condition:** | The managers get vehicle information. | |

Table 5: Use case description of view vehicle information

|  |  |  |
| --- | --- | --- |
| **Use case name:** | Register vehicle | |
| **Use case ID:** | UC 6 | |
| **Actor:** | Manager | |
| **Description** | Allow the Manager to register vehicle. | |
| **Pre-condition:** | The manager activates the system and login to the system. | |
| User action | System response |
| **Basic course of action** | 1. The manager   Selects the register car link.  3. The manager fills the required fields.  4. Manager click submit button.  7. Use case ends. | 2.The system will display  the vehicle registration form.  5.The system checks the  input data.  6.The system displays  the successful notification |
| **Alternative course of Action** | 5.1. The system displays incorrect entered data message and return to path 3 | |
| **Post-condition:** | The vehicle is registered. | |

Table 6: Use case description of register vehicles

|  |  |  |  |
| --- | --- | --- | --- |
| **Use case name:** | Assign vehicle. | | |
| **Use case ID:** | UC 7 | | |
| **Actor:** | Manager | | |
| **Description** | It allows the manager to assign car for the requested service | | |
| **Pre-condition:** | Manager open and login into the system | | |
| User Action | system response | |
| **Basic course of action** | 1. The manager click assign vehicle link.   3.Manager selects and inserts required data  4.The manager click assign button.  7. Use case ends. | | 2. System display assign vehicle page .  5. System validates selected and inserted data.  6. System assigned the vehicle and automatically sends notification for driver and requester office. |
| **Alternative course of Action** | 5.1.The system displays error message incorrect entered data message and return to path 3 | | |
| **Post-condition:** | Vehicle will be assigned. | | |

Table 7 : Use case description of assign vehicle

|  |  |  |
| --- | --- | --- |
| **Use case name:** | Calculate fuel balance | |
| **Use case ID:** | UC 8 | |
| **Actor:** | Manager | |
| **Description** | The fuel balance form takes and compares two reading of current entry and the previous saved reading and produces fuel balance data. | |
| **Pre-condition:** | Manager must get information of vehicles status. | |
| **User action** | **System response** |
| **Basic course of action** | 1. Manager enters username and password.  4. Manager enters input value.  6. The system save the difference and store the status of fuel.  7. Use case ends. | 2. System checks the validity of username and password and then checks for authentication and authorization.  3.System display fuel balance page  5. System checks the validity the input data.  . |
| **Alternative course of Action** | 2.1 If the user name and password is not validated and verified then the system responds error message and return.  5.1 If the input is invalid or incorrect the system displays “invalid input” message and return. | |
| **Post-condition:** | The fuel balance will be calculated. | |

Table 8: Use case description of check fuel balance

|  |  |  |
| --- | --- | --- |
| **Use case name:** | Request maintenance | |
| **Use case ID:** | UC 9 | |
| **Actor:** | Driver | |
| **Description** | A driver formulate request to repair vehicle. The request describes briefly the vehicle problem. | |
| **Pre-condition:** | Driver is registered. | |
| **User action** | **System response** |
| **Basic course of action** | 1The driver click request maintenance link.  .  3. Driver enters information related to the maintenance request for the vehicle.  6.Use case ends. | 2.System display vehicle maintenance form  4.System checks validity of the information  5.System stores the data in the database. |
| **Alternative course of Action** | 4.1. If the data is invalid or incorrect the system displays “invalid input” message. | |
| **Post-condition:** | Maintenance record will be created. | |

Table 9: Use case description of request maintenance

|  |  |  |
| --- | --- | --- |
| **Use case name:** | Request vehicle service | |
| **Use case ID:** | UC 10 | |
| **Actor:** | Officer and staff | |
| **Description** | It allows the staff (i.e. employee under officer) request service to officer and the officer directly request vehicle service to manager. | |
| **Pre-condition:** | Officer and staff enter user name and password. | |
|  |  |
| **Basic course of action** | 1. Officer and staff enter user name and password.  3. The officer and staff click the request service button.  5.Officer and staff fill required vehicle information on displayed form.  6. User clicks send request button.  8. Use case ends. | 2. System checks the validity of username and password and then checks for authentication and authorization.  4. The system displays request service form.  7.System checks validity of the information  8. The system displays sent message acknowledgement. |
| **Alternative course of Action** | 2.1. If the user name and password is not validated and verified then the system responds error message and return.  7.1 If the data is invalid or incorrect the system displays “invalid input” message. | |
| **Post-condition:** | The request is sent. | |

Table 10: Use case description of request services

|  |  |  |
| --- | --- | --- |
| **Use case name:** | Generate report | |
| **Use case ID:** | UC 11 | |
| **Actor:** | Manager | |
| **Description** | It allows administrator and manager generating report for the tasks performed and if there is any difficult problem. | |
| **Pre-condition:** | The admin and manager open the system and login to the system | |
| User action | System response |
| **Basic course of action** | 1. The user must log to his/her page 2. The user select generate report link.   4. The user selects the criteria from the given options and clicks on Display button.  7. Use case ends. | 3. The system displays the options (criteria).  5. System check selected criteria by the user.  6. The system displays the information to the user. |
| **Alternative course of Action** | 5.1. The system displays error message as invalid selection. | |
| **Post-condition:** | The report will be generated. | |

Table 12: Use case description of generate report

|  |  |  |
| --- | --- | --- |
| **Use case name:** | View maintenance request | |
| **Use case ID:** | UC 12 | |
| **Actor:** | Mechanic | |
| **Description** | This use case describes viewing maintenance request to repair the vehicle. | |
| **Pre-condition:** | The mechanics open the system and login to the system | |
| **User action** | **System response** |
| **Basic course of action** | 1. User enters username and password.   4. Mechanic click approve button.  6.Users click ok button  8. Use case ends. | 2. System checks the validity and then authentication and authorization of username and password.  3. System display viewing maintenance request page.  5. System display recommended page.  7. System automatically updates the vehicle condition and sends response to driver. |
| **Alternative course of Action** | 2.1. If the username and password is not validated and verified, system displays error message  7.1. If user click cancel button, the system not update and send . | |
| **Post-condition:** | The Vehicle’s problem will be viewed. | |

Table 11: Use case description of view maintenance request

|  |  |  |
| --- | --- | --- |
| **Use case name:** | Submit Comment | |
| **Use case ID:** | UC 13 | |
| **Actor:** | Driver, officer | |
| **Description** | The driver and officer can give comment or feedback to manager. | |
| **Pre-condition:** | Driver and officer must have valid Email address. | |
| **User action** | **System response** |
| **Basic course of action** | 1. The driver and officer initiates to give comment.  2. The driver and officer click on the comment link.  4. The driver and officer insert all the required fields.  7. Use case ends. | 3. The system displays the form.  5. The system validates the entered information.  6. The system display as your comments has been been sent. |
| **Alternative course of Action** | * 1. The system display error message and give a chance to retype. | |
| **Post-condition:** | The driver and officer sends comment to the manager. | |

Table 12: Use case description of submit comments

|  |  |  |
| --- | --- | --- |
| **Use case name:** | View Comment | |
| **Use case ID:** | UC 14 | |
| **Actor:** | Manager | |
| **Description** | It allows manager to see the comments that are submitted from the driver. | |
| **Pre-condition:** | Manager must have a full privilege to read the comments. | |
| **User action** | **System response** |
| **Basic course of action** | 1. User log to his/her page.  2. User clicks on view comment link.  4. User starts to view the comments.  5.Use case ends | 3. The system reorders the comments according to the time of delivery. |
| **Post-condition:** | The managers view the submitted comments. | |

Table 13: Use case description of view comments

|  |  |  |
| --- | --- | --- |
| **Use case name:** | Logout | |
| **Use case ID:** | UC 15 | |
| **Actor:** | Admin, manger, driver,passenger, officer and staff | |
| **Description** | The user logouts when he/she wants to  back home page or exit from the system | |
| **Pre-condition:** | Users finish their activities. | |
| **User action** | **System response** |
| **Basic course of action** | 1. User clicks the log out button   4.Use case ends | 2.The system responds to the requested action.  3.System returns the users to the login page. |
| **Post condition** | User logs out or sign out. | |

Table 14: Use case description of logout

### 2.3.2. Sequence Diagram

A sequence diagram shows object interactions arranged in time sequence. It is dynamic model of use cases, showing the interaction among classes during a specified time period. A sequence modeling in our system is used to formalize the behavior of the system and to visualize the communication among objects. It helped us to identify additional objects and participate in the use case. This phase of the document ties uses cases with objects and shows the behavior of a use case is distributed among its participating objects.

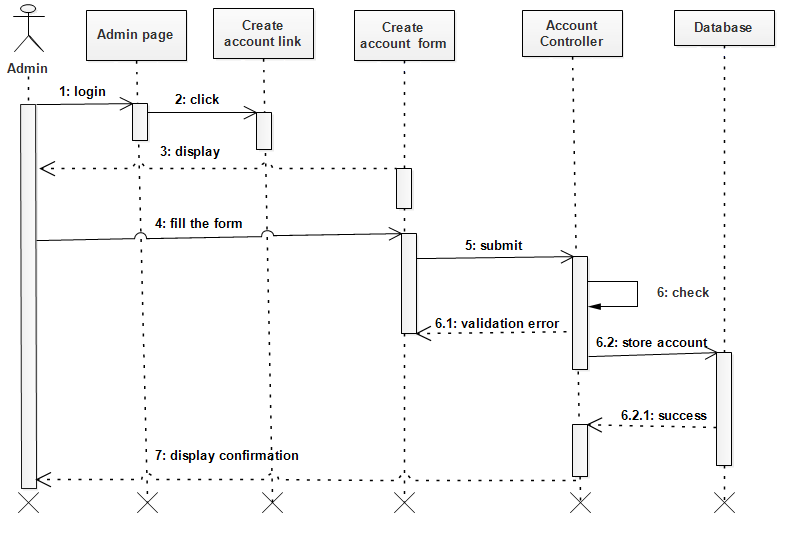


Figure 3: Sequence diagram for create account

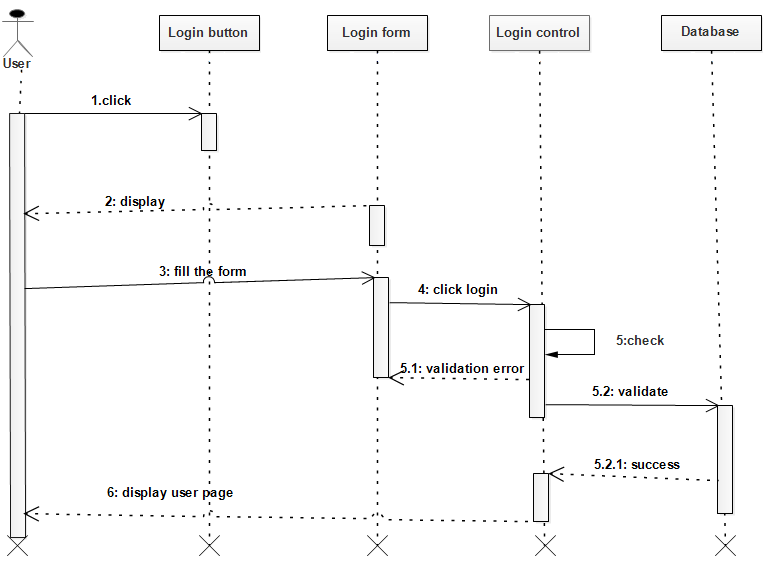


Figure 4 : Sequence diagram for login

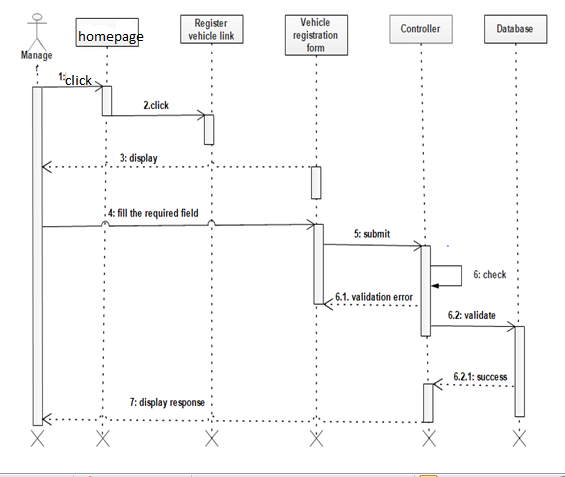


Figure 5: Sequence diagram for register vehicle

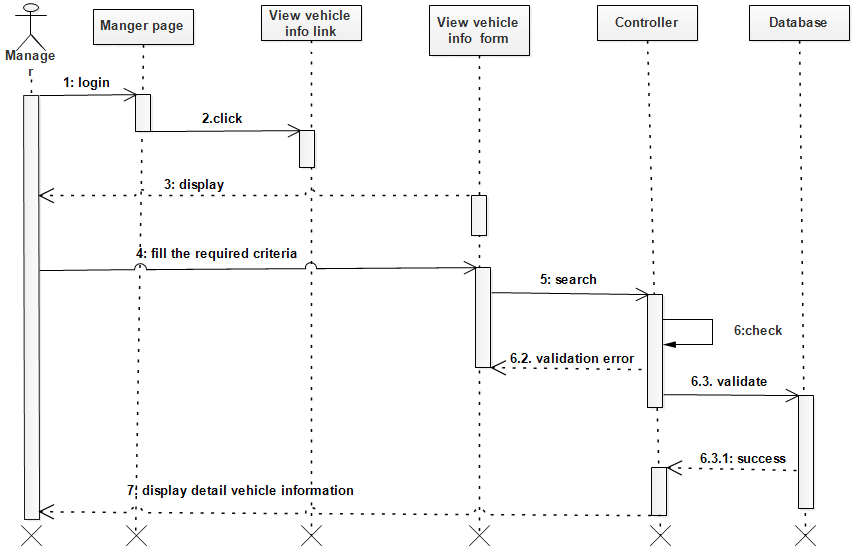


Figure 6: Sequence diagram for view vehicle information

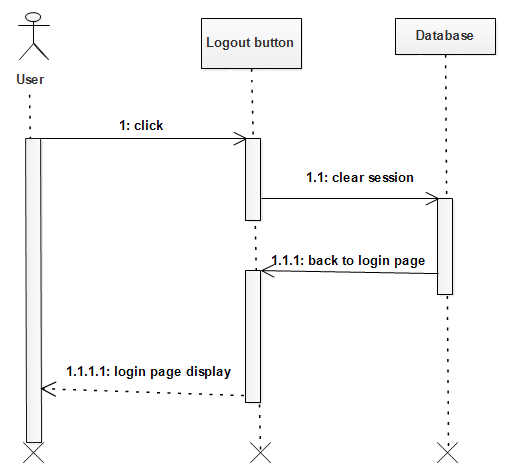


Figure 7 Sequence diagram for logout

### 2.3.3. Activity Diagram

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent. Activity diagrams deals with all type of flow control by using different elements like fork, join etc.

The purposes of activity diagram can be described as:

* Draw the activity flow of a system.
* Describe the sequence from one activity to another.
* Describe the parallel, branched and concurrent flow of the system.

Home page

Admin login page

Enter user name and

Password

Submit

Validate

Enter user

Information

Submit

Select Create Account

Page

System create user

Account

Validate

System display creates account form.

Incorrect

Correct

Incorrect

Correct

Home page

Figure 8: Activity diagram for create account

User login page

Enter user name and

password

submit

Validate

Display appropriate

page

Home page

correct

Incorrect

Figure 9:Activity diagram for Login

Home page

Manager login page

Enter user name and

Password

Submit

Validate

Fill the necessary data

Submit

Register vehicle

Information

Validate

Display vehicle registration page

Incorrect

Correct

Incorrect

Correct

Home page

Figure 10: Activity diagram for register vehicle

Enter user name and

Password

Submit

Validate

Enter input

Submit

Calculate and Store

Which Travelled Kilometers

Validate

Display Check fuel

Balance page

Incorrect

Correct

Incorrect

Correct

Home page

Manager login page

incorrect

Figure 11Activity diagram for calculate fuel balance

### 2.3.4. Analysis of class diagram

The class diagram is a static model. It 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 also 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.

i

**Manage account**

-create account

-update account

-delete account

**Assign vehicle**

-user\_id:varchar(45)

-plate no:varchar(45)

-Full name: :varchar(45)

-Starts: :varchar(45)

-Arrival: :varchar(45)

-outgoing:varchar(45)

-times: :varchar(45)

+assign vehicle()

+send notification()

User **User**

-Name:string

-user\_id:string

-sex:string

-phone no:string

-username: string

-passward:string

+login ()

+update account ()

**Comment**

-name:string

-email: :string

-date;datetime

submitcomment ()

view comment()

**Request maintenances**

-Full name: :varchar(45)

-Request:varchar(35) Date:Datetime

-Request Reason: :varchar(45)

-Vehicle Status:varchar

+view request()

+mainteinance\_request()

+send\_response()

-Create\_Account()

-update\_account()

-Delete\_Account()

**Manager**

+register\_vehicle()

+manage\_request()

+assign\_vehicle()

+view\_vehicle()

+check\_fujelbalance()

+view\_comment()

**Officer**

+request\_service()

+manage\_reques()

+view\_responce()

+submit\_comment()

**Staff**

+request\_service()

+view\_responce()

**Mechanic**

+viewmainteinance

request()

-plate no:varchar(45)

-Type:varchar(45)

-Model:varchar(45)

-Engin\_no:varchar(45)

-Owner:varchar(45)

-Condition:varchar(45)

-Capacity:varchar(45)

-Production\_date:varchar(45)

-insurance

+register\_car()

+view\_vehicleinfo()

**Fuel balance**

-Fuel\_id:varchar(45)

-User\_id:varchar(40)

-Fullname:varchar(20)

-Previouskms:varchar(35)

-Currentkms:varchar(29)

-Differencekm:varchar(45)

-KMperLitter:varchar(38)

-UsedFuel:varchar(34)

-Givenfuel:varchar(38)

-Fuelprice:varchar(25)

+Calculate\_vehicle\_fuel-balance()

+Calculate\_total\_fuel\_price()

+register\_vehicle\_traveledkms

**Approve request**

ove request

+Reject

+Accept

+Send\_responce

**Service request**

-User\_id:varchar(45)

-Traveler:varchar(45)

Requestdate:varchar(45)

-Placestart:varchar(45)

-Placearrival:varchar(45

+Request\_vehicle()

+View-request()

**Send report**

-full name:varchar(45)

-Plate no:varchar(45)

-Date:varchar(45)

-Time:varchar(45)

+Update\_Userstatus()

+Update\_vehiclestatus()

1

\*

**Driver**

+maintenance-request()

+submit comment()

+update account ()

+send report()

Tick Ticket

et

Ticket

Ticket\_id int

Bus\_no:int

Seat\_no:int

Price:int

Dep\_date:date

**Reservation**

-reservedate:date

-reservetime:time

-email addres:string

-phone int

ticket amount:int

ticket no :int

**Seat**

-sid:int

-seatno:int

+get sid()

+get seatno()

**Passenger**

Passenger

Passenger

Passenger

Passenger

r\_id:varchar(45)

-Traveler:varchar(45)

-Requestdate:varchar(45)

-Placestart:varchar(45)-Placearrival:varchar(40)

Inheritance

Search

Reserve

1

1

1

1

has

M

1

Give

Submit

1

M

M

1

request

1

M

send

1

M

View

1

M

View

request

request

manages

Manages

Checks

1

M

1

M

1

M

M

1

M

1

Register

1

M

manages

1

M

Assign

**Admin**

Passenger

Passenger

Passenger

Passenger

**Vehicle Registration**

Passenger

Passenger

Passenger

Passenger

Figure 12 : analysis of class diagram

# CHAPTER THREE

# 3 .system Design

## 3.1. Design of Class Diagram

The class diagram is a static model. It 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 also 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.

**Manage account**

-create account

-update account

-delete account

**Assign vehicle**

-user\_id:varchar(45)

-plate no:varchar(45)

-Full name: :varchar(45)

-Starts: :varchar(45)

-Arrival: :varchar(45)

-outgoing:varchar(45)

-times: :varchar(45)

+assign vehicle()

+send notification()

User **User**

-Name:string

-user\_id:string

-sex:string

-phone no:string

-username: string

-passward:string

+login ()

+update account ()

**Comment**

-name:string

-email: :string

-date;datetime

submitcomment ()

view comment()

**Request maintenances**

-Full name: :varchar(45)

-Request:varchar(35) Date:Datetime

-Request Reason: :varchar(45)

-Vehicle Status:varchar

+view request()

+mainteinance\_request()

+send\_response()

-Create\_Account()

-update\_account()

-Delete\_Account()

**Manager**

+register\_vehicle()

+manage\_request()

+assign\_vehicle()

+view\_vehicle()

+check\_fujelbalance()

+view\_comment()

**Officer**

+request\_service()

+manage\_reques()

+view\_responce()

+submit\_comment()

**Staff**

+request\_service()

+view\_responce()

**Mechanic**

+viewmainteinance

request()

-plate no:varchar(45)

-Type:varchar(45)

-Model:varchar(45)

-Engin\_no:varchar(45)

-Owner:varchar(45)

-Condition:varchar(45)

-Capacity:varchar(45)

-Production\_date:varchar(45)

-insurance

+register\_car()

+view\_vehicleinfo()

**Fuel balance**

-Fuel\_id:varchar(45)

-User\_id:varchar(40)

-Fullname:varchar(20)

-Previouskms:varchar(35)

-Currentkms:varchar(29)

-Differencekm:varchar(45)

-KMperLitter:varchar(38)

-UsedFuel:varchar(34)

-Givenfuel:varchar(38)

-Fuelprice:varchar(25)

+Calculate\_vehicle\_fuel-balance()

+Calculate\_total\_fuel\_price()

+register\_vehicle\_traveledkms

**Approve request**

ove request

+Reject

+Accept

+Send\_responce

**Service request**

-User\_id:varchar(45)

-Traveler:varchar(45)

Requestdate:varchar(45)

-Placestart:varchar(45)

-Placearrival:varchar(45

+Request\_vehicle()

+View-request()

**Send report**

-full name:varchar(45)

-Plate no:varchar(45)

-Date:varchar(45)

-Time:varchar(45)

+Update\_Userstatus()

+Update\_vehiclestatus()

1

\*

**Driver**

+maintenance-request()

+submit comment()

+update account ()

+send report()

Tick Ticket

et

Ticket

Ticket\_id int

Bus\_no:int

Seat\_no:int

Price:int

Dep\_date:date

**Reservation**

-reservedate:date

-reservetime:time

-email addres:string

-phone int

ticket amount:int

ticket no :int

**Seat**

-sid:int

-seatno:int

+get sid()

+get seatno()

**Passenger**

Passenger

Passenger

Passenger

Passenger

r\_id:varchar(45)

-Traveler:varchar(45)

-Requestdate:varchar(45)

-Placestart:varchar(45)-Placearrival:varchar(40)

Inheritance

Search

Reserve

1

1

1

1

has

M

1

Give

Submit

1

M

M

1

request

1

M

send

1

M

View

1

M

View

request

request

manages

Manages

Checks

1

M

1

M

1

M

M

1

M

1

Register

1

M

manages

1

M

Assign

**Admin**

Passenger

Passenger

Passenger

Passenger

**Vehicle Registration**

Passenger

Passenger

Passenger

Passenger

Table17: Decription of Assign vehicle class

|  |  |  |
| --- | --- | --- |
| **Attribute** | **purpose** | **type** |
| user\_id | To identify the user | Varchar(20) |
| Full  name | Represents the name of the user | Varchar(45) |
| Starts | Represent where the car starts | Varchar(45) |
| Arrival | Represents the destination place of the car | Varchar(45) |
| times | Represents when it reaches from starts to arrival | Varchar(45) |
| Reason | It describes for what purpose it go | Varchar(45) |

Table 15: table of Assign vehicle method

|  |  |
| --- | --- |
| method | purpose |
| assign vehicle() | Used to assign vehicle |
| send notification() | Used to send notification for users |

Table 16: description of comment class

|  |  |  |
| --- | --- | --- |
| attribute | purpose | type |
| Name | Represent lame of the user | Varchar(45) |
| email | Represent the address of the user | Varchar(25) |
| message | Represent the idea of the sender | Varchar(250) |
| date | Describes the date | Varchar(45) |

Table 17: description of comment method

|  |  |
| --- | --- |
| method | purpose |
| submit comment() | Used to send comment |
| view comment () | To view comment |

Table 18: description of vehicle registration class

|  |  |  |
| --- | --- | --- |
| attribute | purpose | type |
| model | Used to describe the model of the car | Varchar(20) |
| Plate no | Used to identify the car from the other car | Varchar(30) |
| Owner | Used to owner of the car | varchar(18) |
| Capacity | Used to describe the speed of car | varchar(23) |
| Production date | Used to when it is made | varchar(30) |
| insurance | Used to help help the owner of car during emergency | varchar(30 |

Table 19: vehicle registration method

|  |  |
| --- | --- |
| Method | purpose |
| register car() | Used to register car |
| viewi\_vehicleinfo() | Used to view vehicle info |

Table 20: check fuel balance class

|  |  |  |
| --- | --- | --- |
| **attribute** | purpose | type |
| Fuel\_id: | Used to identify fuel | int |
| User\_id | Used to identify the user | int |
| Full name | Represent the name of the user | Varchr(20) |
| Previouskms | Describe how far it travelled in km | float |
| Differences’kms | Represent the difference between it consumes and remain fuel | float |
| KMperLitter | How much it required km /litter | float |
| Fuel price | Represent the price of fuel | float |
| Used fuel | How much used fuel | float |

Table 21: check fuel balance method

|  |  |
| --- | --- |
| method | purpose |
| Calculate\_vehicle\_fuel-balance() | Used to calculate vehicle fuel |
| Calculate\_total\_fuel\_price() | Used to calculate totalfuel\_price |
| register\_vehicle\_traveledkms() | Used to register vehicle travelled kms |

Table 22 : Manager class

|  |  |  |  |
| --- | --- | --- | --- |
| attribute | purpose | | type |
| name | Represent manager name | | Varchar(30) |
| user\_id | Used to identify the manager | | Varchar(15) |
| Sex | Used to gender of manager | | Varchar(8) |
| Phone no | manager phone no | | Varchar(20) |
| username | Used to describe user name | | Varchar(30) |
| password | Used to log into the system | Varchar(80) | |

Table 23: Manager method

|  |  |
| --- | --- |
| method | purpose |
| login() | To enter the system |
| update account() | To change the account |

Table 24: driver class

|  |  |  |  |
| --- | --- | --- | --- |
| attribute | purpose | type | |
| name | Describes the driver name | Varchar(30) | |
| user\_id | Used to identify the driver | Varchar(15) | |
| Sex | Tells the gender of the driver | Varchar(8) | |
| Phone no | Used to know the phone no | Varchar(20) | |
| Username | Identify the username of the driver | Varchar(30) | |
| Password | Used to login and access the system | | Varchar(80) |

Table 25 : driver method

|  |  |
| --- | --- |
| Method | purpose |
| submit comment() | Used to send comment |
| update account() | Used to update comment |
| send report() | Used to send report |
| Maintenance request() | Used to ask maintenance request while vehicle damaged |

Table 26: Send report class

|  |  |  |
| --- | --- | --- |
| attribute | purpose | type |
| Full name | Describes the sender name | Varchar(30) |
| Plate no | which identifies where the vehicle was registered | varchar(30) |
| Date | This identifies when the report sent. | Varchar(45) |

Table 27: Send report method

|  |  |
| --- | --- |
| Methods | purpose |
| Update\_Userstatus | Used to update the user status |
| Update\_vehiclestatus | Used to update the vehicle status |

## 3.2. User interfaces design

* **User Interface**

In this system users will communicate with it through the following user interfaces.

* **Home Page:** This form contains some links which lead it to the concerned page, and if the user has an account he/she will directly go to concerned page by entering their username and password.



Figure 13: user interfaces home page

* **Login form:-**This form found immediately following the home page. Home page appears as the site on which the system is deployed is opened. All user will have their own password. Those forms appeared using password and user name will not accessible by other persons except for those who have privilege.

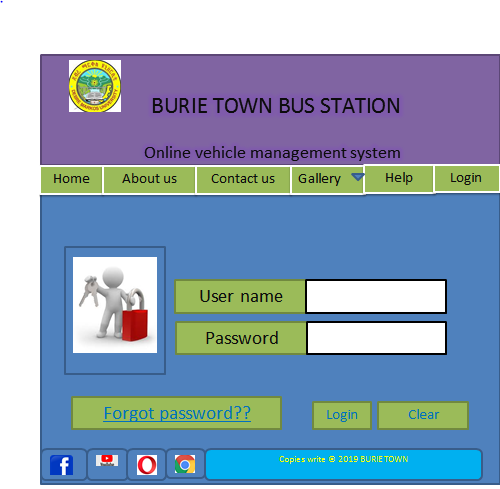


Figure 14: login form

* **Create Account:** this is creating account page in this page the administrator create accounts for the user (manager, driver, officer, staff and mechanic).

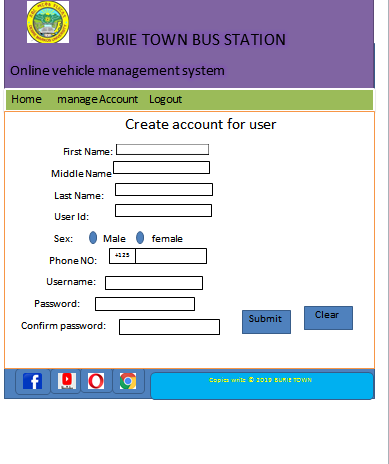


Figure 15: create account

* **Register vehicle :** The manager activates the system and login to the system. Allow the Manager to register vehicle

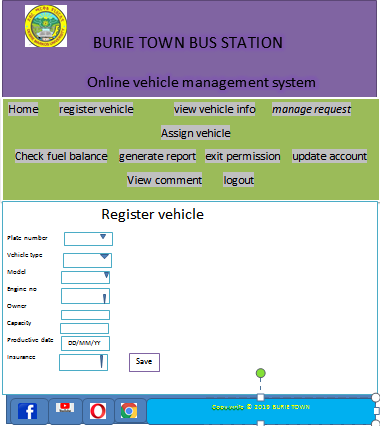


Figure 16: register vehicle

3.3 . Deployment Diagram

Deployment diagram is a UML diagram that used for describing how the hardware components are deployed. We used to visualize the topology of the physical components of our system where the mentioned software component diagrams are deployed. This diagram depicts a static view of the run-time configuration of processing nodes and the components that run on those nodes. In other words, deployment diagrams show the component of the software will installed on which the hardware machine and how they interact with each other. That is hardware and software part of the system work tog

Figure 17: Deployment Diagram

Manager

Officer

Staff

Driver

Mechanic

Generate report

Register vehicle

Check fuel balance

View comment

Assign vehicle

Submit comment

fdfddv

Manage account

Request maintenance

Request service

Approve request

View maintenance request

Database server

Application server

Client side



View vehicle info

VMS database server

Admin

Passenger

Route info

See seat

**CHAPTER FOUR**

**4 . Implementation**

**4.1 Overview of the programming language used**

* PHP is a popular and widely used programming language which is utilized to build dynamic web applications with MySQL database connections. Because it is user Friendly and Easy to Learn, Cost user Friendly, It’s fast and easy, it accesses everything etc...
* **Back End**
* MySQL software of the data base system and PHP language was used in developing and managing the back end of the system.
* **Front End**
* The user interface had been developed using html, java script, CSS since it easily designing the front end and connected in to database easily.

We use HTML and CSS for prepare user interface.

* We use java script for form validation
* We use MYSQL database for storage of data.

**4.2. Algorithms used for login**

**Login** (**Username, password**):- that pass/checks three arguments for login function.

* If length of username and password==0:- Check that Zero length is not allowed.

Display error message “username and password cannot be empty”.

Stay in the same page and the actor fills the form after he or she knows the field is required.

The system retrieves username and password which the actor submits.

* If username! =username or password! =password or status! =active: - Authentications for the right actor.

Display error message “Invalid username or password or your account is deactivate”.

* If his/her account detail is correct access the required page.

//End of the function login

**4.3. Sample codes**

Coding is the process whereby the physical design specification created by the designers is turned in to working computer code by the programmer. The code is made simple in such a way that another programmer can easily understand and work on that in future

**For register vehicle**

<?php

include("connection.php");

session\_start();

if(isset($\_SESSION['user\_id']))

{

$mail=$\_SESSION['user\_id'];

} else {

?>

<script>

alert('You Are Not Logged In !! Please Login to access this page');

alert(window.location='login.php');

</script>

<?php

}

?>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<title>Vehicle Management System</title>

<link href="tooplate\_style.css" rel="stylesheet" type="text/css" />

<link rel="shortcut icon" HREF="12.jpg" />

<!-- Start WOWSlider.com HEAD section-->

<link rel="stylesheet" type="text/css" href="engine1/style.css" />

<script type="text/javascript" src="engine1/jquery.js"></script>

<!-- End WOWSlider.com HEAD section -->

<SCRIPT language=Javascript>

<!--

function isNumberKey(evt)

{

var charCode = (evt.which) ? evt.which : event.keyCode

if (charCode > 31 && (charCode < 48 || charCode > 57)){

return false;

}

else{

return true;

}

}

//-->

</SCRIPT>

</head>

<body>

<div id="tooplate\_wrapper">

<div id="tooplate\_header">

<div id="site\_title"><h2 id="hheader"style="margin-left:180px;margin-top:40px;font-size:22px;font-family:Cooper Black;"><b><span style="font-size:36px;color:white"> Burie Town Bus Station</span><br/><br/></b>Online Vehicle Management System</h2></div>

</div> <!-- end of forever header -->

<div id="tooplate\_menu">

<ul>

<li><a href="manager.php" class="current">Home</a></li>

<li><a href="registervehicle.php">Register Vehicle</a></li>

<li><a href="vehicleinfo.php">View Vehicle info</a></li>

<li><a href="manage\_requests.php">Manage Request</a></li>

<li><a href="assign.php">Assign Vehicle</a></li>

<li><a href="fuel.php">Check Fuel Balance</a></li>

<li><a href="#">Generate Report</a>

<ul style="margin-left:10px;padding-right:4px;">

<li><a href="reportuser.php">Report For Users</a></li>

<li><a href="reportvehicle.php">Vehicle Report</a></li>

<li><a href="FuelReport.php">Fuel Information</a></li>

</ul>

</li>

<li><a href="permission.php">Get Exit permission</a></li>

<li><a href="update.php">Update account</a></li>

<li><a href="viewcomment.php">View Comment</a></li>

<li><a href="logout.php">Logout</a></li>

</ul>

</div> <!-- end of tooplate\_menu -->

<div id="tooplate\_main" class="shadow">

<div id="tooplate\_content">

<form name="form1" method="post" action="registervehicle.php">

<fieldset><table align="center"style="color:black">

<legend align =center><h2 align="right" style="color:Blue">Register Vehicle</h2></legend>

<tr>

<td style="padding-top:12px;">Plate Number:</td>

<td style="padding-top:12px;"><input type="text" name="pno" id="pno" size="20" pattern="\d{3,7}"onKeyPress="return isNumeric(event)" required x-moz-errormessage="Please Enter Number Only between 2 and 8 digit ! " title="Please Enter Number Only between 2 and 8 digit !"/></td>

</tr>

<tr>

<td style="padding-top:12px;">Vehicle Type :</td>

<td style="padding-top:12px;"><select name="vtype" required maxlength="8">

<option ></option>

<option value='Nissan patrol'>Nissan patrol</option>

<option value='Toyota PRADO'>Toyota PRADO</option>

<option value='Toyota single cub'>Toyota single cub</option>

<option value='Nissan pick up'>Nissan pick up</option>

<option value='Cacciamali bus'>Cacciamali bus</option>

<option value='Daewoo bus'>Daewoo bus</option>

<option value='Mercedes benz bus'>Mercedes benz bus</option>

<option value='Fiat-mini-bus'>Fiat-mini-bus</option>

<option value='Other-Type'>Other Type</option>

</select></td>

</tr>

<tr>

<td style="padding-top:12px;">Model :</td>

<td style="padding-top:12px;"><select name="model" required maxlength="5">

<option ></option>

<option value='TVTSLEFY61NRA'>TVTSLEFY61NRA</option>

<option value='KUN25L-PRMDHV'>KUN25L-PRMDHV</option>

<option value='HZJ79LTJMRS'>HZJ79LTJMRS</option>

<option value='CVRULCFD22NWN'>CVRULCFD22NWN</option>

<option value='IHZJ105L-GMRS'>IHZJ105L-GMRS</option>

<option value='KB 7TNJNML'>KB 7TNJNML</option>

<option value='BE637JLSH'>BE637JLSH</option>

<option value='BF120'>BF120</option>

<option value='9BM3840883B'>9BM3840883B</option>

<option value='Other-Model'>Other Model</option>

</select></td>

</tr>

<tr>

<td style="padding-top:12px;"> Engine Number:</td>

<td style="padding-top:12px;"><input type="text" name="eno" id="eno" size="20" pattern="\w{2,20}" required x-moz-errormessage="Please Enter engine Number from 2-20 digit ! " title="Please Enter Engine Number From 2-20 digit !"/></td>

</tr>

<tr>

<td style="padding-top:12px;"><font size="4">Owner :</font></td>

<td style="padding-top:12px;"><input type="text"pattern="\D{3,15}" required x-moz-errormessage="Please Enter Only Letter! at least 3 characters required" title="Please Enter Letter at least 3 characters required " name="owner" id="error12"size="20"/></td>

</tr>

<tr>

<td style="padding-top:12px;"> <font size="4">Capacity:<font></td>

<td style="padding-top:12px;"><input type="text" name="cap" id="cap" size="5" pattern="\d{1,2}"onKeyPress="return isNumeric(event)" required x-moz-errormessage="Please Enter Number of Capacity From 1-65 number! " title="Please Enter Number of Capacity From 1-65 number !"/></td>

</tr>

<tr>

<td style="padding-top:12px;"><font size="4">Productive Date:</font></td>

<td style="padding-top:12px;"><input type="date" name="date" title="Enter Id for search " id="date-pick" placeholder ="year-month-date" class="search" autocomplete="off"/></td>

</tr>

<tr>

<td style="padding-top:12px;"><font size="4">Insurance :</font></td>

<td style="padding-top:12px;"><select name="ins" style="width:148px;" required maxlength="5">

<option ></option>

<option value='Insured'>Insured</option>

<option value='Uninsured'>Uninsured</option>

</select></td>

</tr>

<tr> <td></td>

<td style="padding-top:12px;"><input type="submit" name="submit" value="Save" class="button\_example" /></td>

</tr>

</table>

</form></fieldset>

<?php

if(isset($\_POST['submit']))

{

//geting values

//$username=$\_SESSION['user'];

$pno=$\_POST['pno'];

$vtype=$\_POST['vtype'];

$model=$\_POST['model'];

$eno=$\_POST['eno'];

$owner=$\_POST['owner'];

$cap=$\_POST['cap'];

$ins=$\_POST['ins'];

$status=1;

$date=$\_POST['date'];

$query="SELECT \* FROM vehicleregister where Plate\_no='$\_POST[pno]'";

$resultw=mysql\_query($query);

$count=mysql\_num\_rows($resultw);

if($count==1){

while($row=mysql\_fetch\_array($resultw)){

$Plate\_no=$row[0];

}

if($Plate\_no==$\_POST['pno']

)

if($Plate\_no==$\_POST['pno'])

if($Plate\_no==$\_POST['pno'])

{

echo "<script>alert(' Plate\_no is used by another vehicle!');</script>";

echo'<meta content="3;registervehicle.php" http-equiv="refresh" />';

}

}

else

{

if($pno==$pno)

{

$sql="insert into vehicleregister values('$pno','$vtype','$model','$eno','$owner','$cap','$date','$ins','$status')";//

//echo"$insert";//

if (!mysql\_query($sql,$conn))

{

echo "<script>alert(' Already Registered!');</script>".mysql\_error();

echo' <meta content="6;registervehicle.php" http-equiv="refresh" />';

}

else

{

echo "<script>alert(' Vehicle Register is successfully!');</script>";

echo' <meta content="6;registervehicle.php" http-equiv="refresh" />';

}

}

}

}

?>

**CHAPTER FIVE**

**5. Testing**

Testing is a trial experience in which the deliverables of the project are checked with acceptable Standards in the project. We used unit testing, system testing to test the correctness of each Module and the compiled program.

Table 28:Unit testing.

|  |  |  |
| --- | --- | --- |
| Tested Form | Test Case | Expected Result |
| Login Form | validate user name and password entry as an input from each end users | Display a message when user didn’t fill user name or password and also when there is user name or password error. |
| All other forms | controlling the proper insertion of data | Display a message when user left some text fields, radio buttons, combo boxes or date and time unfilled and insert improper data in to the form try to save. |

**Integrated testing:** - all the modules will be combined together and tested it for its fitness with each other and with the systems functionality. If error occurs in combining them, the module with problem will be identified and recombined.

Table 29: Integrated testing

|  |  |  |
| --- | --- | --- |
| Tested Form | Test Case | Expected Result |
| Login Form | Check the correctness of the form to be displayed after login is succeeded | Display administrator or system members menu |
| Administrator menu | check proper display of selected options to be accessed | Display the selected form from the administrator form as menu |
| Report from | check whether the report will be generated or not | The selected report will be displayed |
| All forms | check the navigation functionality | The form required Will be displayed |

1. **System testing:** - the team member to performs over all functional testing by checking whether it meets the required target or not. Here the system is partially functional and reached its requirement.

Table 30.System testing

|  |  |  |  |
| --- | --- | --- | --- |
| Tested from | Test case | Expected result | Actual result |
| Login form | To validate the proper functionality of login by inserting username and password | To authenticate user | ser will be authenticated and if user is authorized enter to the system else confirm invalidity |
| Search from | To validate the functionality of search form | Search result | If the requested record exist display the result else if it doesn’t exist display the message about the status |
| Report from | To validate the functionality of report form | To generate report | generate the requested report if the request is valid, if request is invalid display message box that describes the invalidity |
| All forms | To validate the functionality of each form | To provide the function required by the form | The form is presented and the required function can operated using the form |

**5.3 .**Validation testing

When we say valid it is to mean that the software functions as it is intended. This is  
tested by giving real data and get real information from the software.

**CHAPTER SIX**

**6. Conclusion and Recommendations**

**6.1 Conclusion**

As the scope of this project described we developed the vehicle management system for BURIE bus station by making it more reliable and efficient. The system would register vehicles and perform additional task that would be performed by the system.

So the system would:-

* Minimize the time required to perform task
* Minimize the work load of employees
* Increase customer satisfaction

This document use the object oriented technology called UML (unified modeling language) that enable the user to understand the software easily.

We finally concluded that BURIE bus station vehicle management system would be benefited by the system developed, and accept it cheerfully. During working on this project, we members of the team had learned a lot.

**6.2Recommendation**

The system that we have developed involves web based vehicle management system for  
BURIE bus station. We recommend the following features need to be included in any further revision and extension attempt.

* May used the web based to change in to android or mobile based application.
* Use uninterruptible power supply or UPS if electric power is not available in university.
* Integrate with the traffics system.
* Adding the chatting system.
* Include GPS.
* They done all system in organization are automated
* Update this system to android based system or integrate with android and PHP.

Therefore, others who are interested to develop a new system on vehicle management system for BURIE bus station or other related systems can get some initial idea about the system. By focusing on the limitation and functional areas of the system they can also develop a better vehicle management system that automates all files managed in vehicles management office and other related thin.

**6.3 Future Enhancement**

For the future as the services of the associations becomes increased, this system should be improved by adding functions and using better technologies. So, future works that the project team proposed are:

* Maintaining the system according to the services of the organization after applying and testing the acceptance of the project by the organization
* Connecting the system with other bus station organizations
* Appling better security mechanisms.

**References**

1.**Jeffery A.Hoffer, Joey F.George.** *Modern systems analysis and design.* 2005.

2. **Jeffery L.Whitten, Lonnie.** *Systems analysis and design methods .* 2004 G.C.

3. **Bahrami, Ali.** *Object oriented systems development .* (1999 G.C).

4. http://www.setha.info/docs/ict/course/0001/chap08.pdf. [Online]

5. *Object Oriented and the UML.2nd rev. Ed England: The Cambridge University Press.*

6. **Bahrami, Ali.** *Object oriented systems development .* 1999G.C.