[**Acknowledgement**](#_u56apc2z15p6)1

[**Abstract**](#_6t7r3vufltwr)2

[**Chapter 1: Introduction**](#_1fob9te)3

[1.1 Introduction](#_3znysh7) 3

[1.2 Problem Statement](#_2et92p0) 3

[1.3 Scope and Objectives](#_tyjcwt) 4

[1.4 Project Features](#_3dy6vkm) 4

[1.5 Summary](#_1t3h5sf) 4

[**Chapter 2: Literature Review**](#_4d34og8)5

[2.1 Overview of Existing System](#_2s8eyo1) 5

[2.2 Limitation of Existing System](#_17dp8vu) 5

[**Chapter 3: System Analysis**](#_3rdcrjn)7

[3.1 Requirement Specification](#_26in1rg) 7

[3.1.1 Functional Requirement](#_2c2aplsiiip6) 7

[3.1.2 Non-Functional Requirement](#_vxrhjyb15f8j) 9

[**Chapter 4: System Design**](#_2xcytpi)11

[4.1 Context Diagram](#_1ci93xb) 11

[4.2 Data Flow Diagram](#_6w0zskv6xofd) 12

[4.3 Entity Relationship Diagram](#_hmau71stva3y) 13

[4.4 Class Diagram](#_zahysfavdyyr) 14

[4.5 Use Case Diagram](#_c8rune6j4565) 16

[**Chapter 5: System Development and Implementation**](#_2j3sbis81t6p)17

[5.1 Programming Platform](#_ovxatko1812i) 17

[5.2 Tools & Technologies](#_o4h56b1fx6vz) 17

[5.3 Testing & Debugging](#_1y95i7biks56) 18

[5.4 Implementation & Result](#_7a5cj2jvik3r) 20

[5.5 Maintenance](#_54fegtnl5uso) 20

[**Chapter 6: Conclusion and Future Enhancement**](#_blgmrvfefrb8)21

[**References**](#_3whwml4)21

[**Appendices**](#_aeso1o36aiqw)22

# **Acknowledgement**

In completing this graduate project we have been fortunate to have help, support and encouragement from many people. I would like to acknowledge them for their cooperation.

First, I would like to thank Mr. Kusal Niraula, my project supervisior, for guiding me through each and every step of the process with his knowledge and support. Thank you for your advice, guidance and assistance.

I would also like to thank Mr. Raju Kattel, our Head of Department, who showed immense patience and understanding throughout the project and provided suggestions.

Finally, I would like to thanks all of my friends who helped throughout the project by providing supporting materials and friendly advice to make this project a success.

# **Abstract**

Bus reservation is an online ticket booking platform that allows different travel agency to sell their bus tickets online. It utilizes business-to-consumer aspect of electronic commerce (e-commerce) which is the most visible business use of the World Wide Web. In this system, a user has the ability to search for specific route and choose affordable bus that travel through that route. In other words, a user can basically search for bus for a given route and and travel date, choose a bus from list of available buses and book it. They will also have the ability to choose the seat they want for their journey.

This project deals with developing an e-commerce based website for Online bus ticket reservation. It provides a user with access to different bus travel agency whose bus is available for booking. In order to facilitate payment for booked seats, a user can directly contact the travel agency where that bus belongs to. The system is implemented using a 3-tier approach, with a backend database (MySQL), a middle tier of Apache Server and PHP, and a web browser as the front end client. The application development framework used here is CodeIgniter.

This is a project with the objective to develop a basic website where a user is allowed to book seats on a bus of their choice. This document provides information about the technologies that has been used to develop such a system.

# 

# **Chapter 1: Introduction**

## **1.1 Introduction**

There are a lot of times when people are unable to buy a proper seat for their journey right before a festival seasons. People don't want get into a huge crowd just to get a ticket for their destination. On top of that most people donot have time to stay on a queue to get their ticket. Also, these days everyone is looking for a better and more easier ways to book a ticket for such occasions.

So, to address this problem, a new system has been developed. This system is called the “Bus Reservation”. In this system, a user has the ability to search buses for a given route on a specific date. In other words, a user can search, choose bus, pick a seat and finally pay for it using online payment portals. Basically, this allows a user to book and buy tickets without reaching to a ticket counter. Also, this system aims to allow all bus operators to come under a single platform so there won't a scarcity of buses online for booking. So in short, this system also allows a user to reserve a seat in a bus. Hence, the term “Bus Reservation” since people are allowed to secure a seat for their journey.

The major concept of this system is to allow business to consumer transaction. So, no central governing body has been implemented in this system which controls the transaction among users. But their will be a body responsible for techincally maintaining servers and other infrastruture for this system to be functional.

## **1.2 Problem Statement**

Generally, ticket booking was not done online (a few years back) mostly because due to lack of internet availability among many people. But even though we have reach to the internet, still, there are not many portal available that faciliate bus ticket booking operations. Moreover, there are even few portals that allowed actual payment transactions while bus ticket booking. But the major problem is, the service they provide is not of a satisfactory level as they didn't include a wide range of travel routes. On top of what is available, half of them were mostly broken in one way or the other such as broken links, insecure online transactions, lack of wide range of travel routes and so on. Most of the available ticket booking sites are own by a travel agency who rarely update their sites and keep track of the available routes. So, people still have to contact travel agencies to get their ticket booked.

Also people these days are quite busy to go and buy a ticket for their journey. And most of the time, even in off seasons, people cannot pick a bus of their choice due to lack of available options. To matter even worst, people have to sit on a seat that is provided by the agents of that particular bus agency. This particular case becomes an actual problem when a passenger has motion sickness and requires to travel next to a window seat.

Thus, the proposed system aims to address such issue by providing an online portal where one can easily search a bus, choose a seat of their choice and pay for it.

## **1.3 Scope and Objectives**

* To allow users to book bus seats of their choice.
* To bring different bus agencies into single platform.
* To make bus booking easier.

## **1.4 Project Features**

The following are list of features that has been implemented on this system:

* Bus searching facility
* Choose a seat for booking
* Search for ticket that was purchased by a customer
* Ability for an agency admins to keep track of their staff (requires agency admin role)
* Ability to set travel schedule (requires agency user role)
* Ability to add or remove bus and its schedules.

## **1.5 Summary**

Bus Reservation is an online platform that focuses on making bus ticket reservation easier. It has tried to implement business to consumer transaction. That means, there is no presence of “middle man” who will gain profit upon sale of a ticket. It has tried to achieve this by allowing user to directly search a bus, choose a seat and pay the price for that choice. The booked ticket can later be retrived using unique ticket id provided during payment.

In this, everyone can book a seat on an available bus with out the need of loging into the system. Also, they will have a choice to choose between many other bus operators and different price for same route. Thus, bus reservation has tried to take ticket booking to another level by providing user with such facilities.

# **Chapter 2: Literature Review**

## **2.1 Overview of Existing System**

There are quite a few existing portals that deals with online bus ticket booking. Two different system that has been considered as an existing system for this project. They are:-

* Westnepalbus.com
  + The features provided by this system are:
    - Print ticket facility
    - Contains wide range of travel routes
    - List of available bus agency
    - Belongs to an organization
* Busnepal.com
  + The features provided by this system are:
    - Print ticket facility
    - Contains wide range of travel routes
    - Support for payment gateway like esewa and ipay

## **2.2 Limitation of Existing System**

Like everything else, the system for this review phase also has some limitations. They are:

* Westnepalbus.com
  + Broken links
  + No result upon submiting the inputs
  + Need to call a provide phone number to check availability of seat
* Busnepal.com
  + Broken links
  + No result upon submiting the inputs
  + Popped error page every now and then

**2.2.1 The Causes**

Upon doing futher analysis, by visiting, similar website owners, I found that one of the major cause of such problems was lack of technical support. Once bus reservation system is up and running, many software developers and software development firms stops providing support and hands over the website to their owner.

Another issue I found was the website owner themself. They said they have a lot of customers during festival seasons so booking tickets via ticket counter only is not enough. So they only make their website up and running during festival seasons. And rest of the time, they leave the website unattendend since no one cares about online book rather people call the owner of the travel agency and simply book the ticket via phone.

**2.2.2 Conclusion**

In summary, many of the websites have no availability of bus for the route and date provided as an inputs. This indicates that the sites are not maintaned, updated, checked and verified properly in a regular intervals.

# **Chapter 3: System Analysis**

## **3.1 Requirement Specification**

## **3.1.1 Functional Requirement**

This section includes the requirements that specify all the fundamental actions of the software system. The list of all functional requirements are given below:

**3.1.1.1 Functional Requirement 1.1**

ID: FR1

TITLE: Bus Searching

DESC: A customer should be able to search buses that is posted on the website. User should be able to search bus based on the date and route provided.

DEP: None

**3.1.1.2 Functional Requirement 1.2**

ID: FR2

TITLE: Check Seat Availability

DESC: A customer should be able to choose seats from the chosen bus.

DEP: FR1

**3.1.1.3 Functional Requirement 1.3**

ID: FR3

TITLE: Ticket Booking

DESC: A customer should be able to get a the details of the booked ticket.

DEP: FR1

**3.1.1.4 Functional Requirement 1.4**

ID: FR4

TITLE: Ticket Printing

DESC: The customer should be able to print or download booked tickets through the website.

DEP: FR 3

**3.1.1.5 Functional Requirement 1.5**

ID: FR5

TITLE: User log-in

DESC: Given that a user has been registered, then the user should be able to log in to the system.

DEP: None

**3.1.1.6 Functional Requirement 1.6**

ID: FR6

TITLE: Buses (Bus Management) - Agency Member

DESC: Given that a user has successfully logged into the system, The user should be able to add new bus, remove old bus and update exsiting buses.

DEP: FR5

**3.1.1.7 Functional Requirement 2.1**

ID: FR7

TITLE: Schedules (Schedule Management) - Agency Memner

DESC: Given that a user has successfully logged into the system, then the user should be able to add new schedule, remove old schedule and update schedule.

DEP: FR5, FR6

**3.1.1.8 Functional Requirement 2.2**

ID: FR8

TITLE: Tickets (Ticket Management) - Agency Member

DESC: Given that a user has successfully logged into the system, then the user should be able to update the tickets of customer and remove the ticket if the booking needs cancellation.

DEP: FR5, FR3

**3.1.1.9 Functional Requirement 2.3**

ID: FR9

TITLE: Updating credentials - Agency Member

DESC: Given that a user has successfully logged into the system, then the user should be able to update their credentials. This includes username and password change.

DEP: FR5

**3.1.1.10 Functional Requirement 2.4**

ID: FR10

TITLE: Users (User management) - Agency Member Admin

DESC: Given that a user has successfully logged into the system, then the agency admin should be able to add, update and delete its agency member.

DEP: FR5

**3.1.1.11 Functional Requirement 3.1**

ID: FR11

TITLE: Managing Picture Sildeshow - Admin Module

DESC: Given that a admin has successfully logged into the system, then the admin should be able to add new picture to the picutre slider, update it and remove it. The admin should be able to view the currently active silde show picture with the system.

DEP: FR5

**3.1.1.12 Functional Requirement 3.2**

ID: FR12

TITLE: Places (Place Management)- Admin Module

DESC: Given that a admin has successfully logged into the system, then the admin should be able to add new places, update them and remove them.

DEP: FR5

**3.1.1.13 Functional Requirement 3.3**

ID: FR13

TITLE: Travel Agencies - Admin Module

DESC: Given that a admin has successfully logged into the system, then the admin should be able to add new agency, update it and remove it.

DEP: FR5

**3.1.1.14 Functional Requirement 3.4**

ID: FR14

TITLE: Updating credentials - Admin Module

DESC: Given that a admin has successfully logged into the system, then the admin should be able to update their own credentials. This includes username and password change.

DEP: FR5

**3.1.1.15 Functional Requirement 3.5**

ID: FR15

TITLE: Agency Admin Management - Admin Module

DESC: Given that a admin has successfully logged into the system, then the admin should be able to add new agency admin, update their credential and delete it.

DEP: FR5

### **3.1.2 Non-Functional Requirement**

This section includes the requirements that specify all the fundamental actions of the software system. The list of all functional requirements are given below:

**3.1.2.1 Non- Functional Requirement 1.1**

ID: NFR1

TITLE: Webpage Loading

DESC: The webpage should load faster so that user can quickly browse through the website to find what they are looking for.

DEP: none

**3.1.2.2 Non- Functional Requirement 1.2**

ID: NFR2

TITLE: Error Handling

DESC: The webpage should not show any major errors that could reveal its programming language. Instead, the website should redirect to a generic error page indicating some generic error information.

DEP: none

**3.1.2.3 Non- Functional Requirement 1.3**

ID: NFR3

TITLE: User Friendly Interface

DESC: The user interface should be easy to use. Website navigation should follow a proper pattern. It should also contain links at proper places so that user won't get lost within the website.

DEP: none

**3.1.2.4 Non- Functional Requirement 1.4**

ID: NFR4

TITLE: Security

DESC: The website should not be prone to attacks. It should be secure enough to handle attacks like cross site scripting and SQL injection.

DEP: none

**3.1.2.5 Non- Functional Requirement 1.5**

ID: NFR5

TITLE: Availability

DESC: The website should be hosted in a public network. The user should be able to access from anywhere around the world.

DEP: none

**3.1.2.6 Non- Functional Requirement 1.6**

ID: NFR6

TITLE: Reliablity

DESC: The website should be reliable enough to perform its tasks without crashing. But for any input from user that is beyond its predefined functionality, the website should show an error page indicating the error.

DEP: none

#### 

# **Chapter 4: System Design**

## **4.1 Context Diagram**

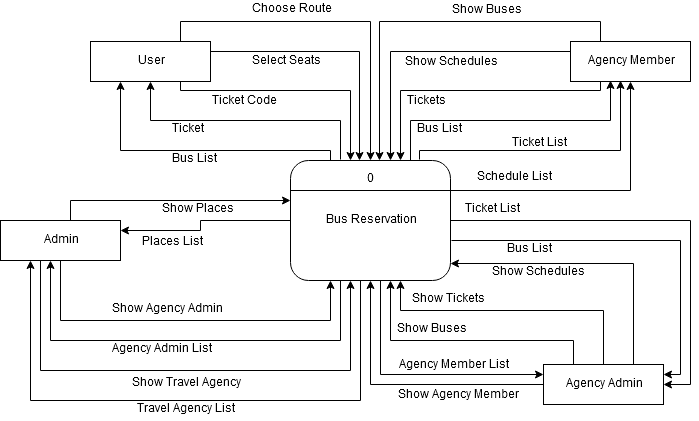


Fig 4.1: Context Diagram

## **4.2 Data Flow Diagram**

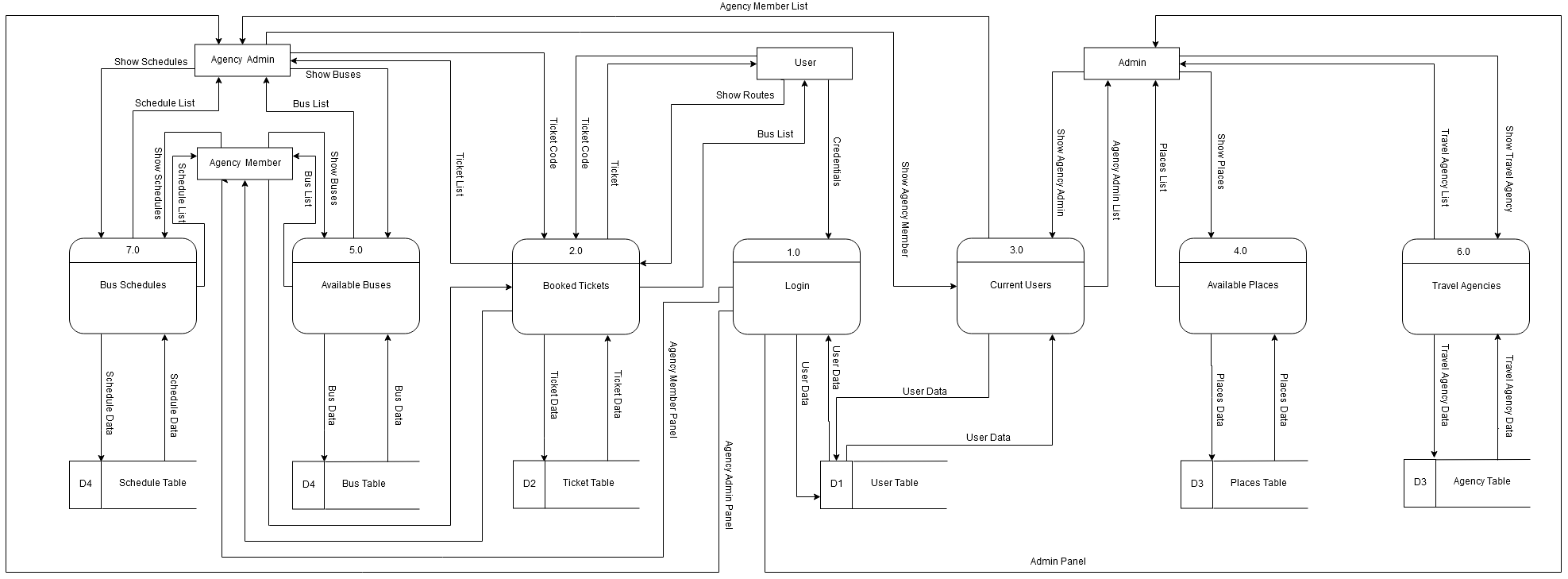


Fig 4.2: Data Flow Diagram

# 

## **4.3 Entity Relationship Diagram**

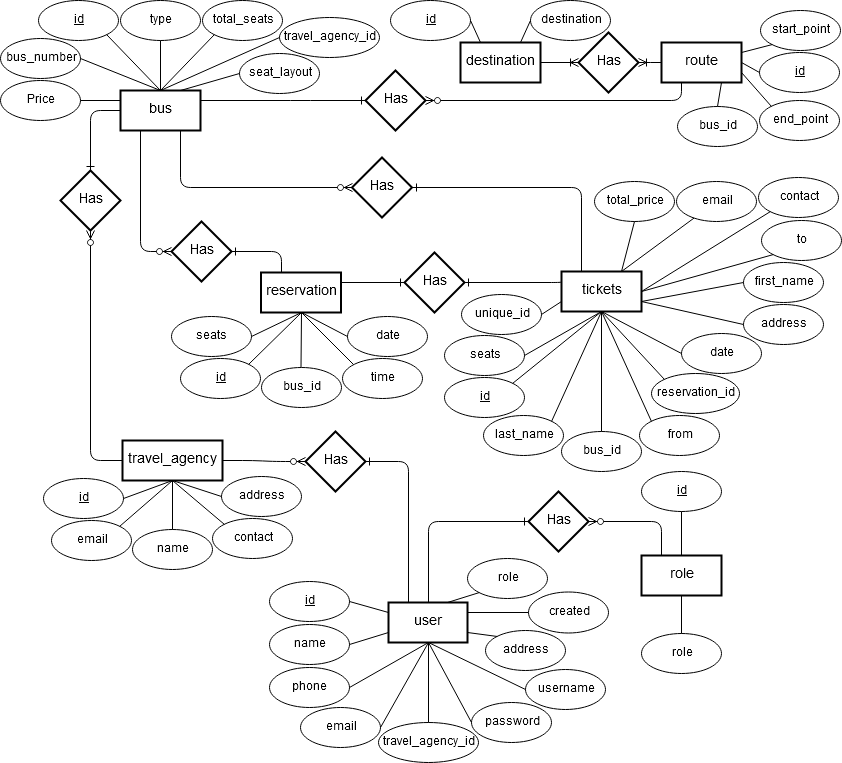


Fig 4.3: Entity Relationship Diagram

## **4.4 Class Diagram**

## 

Fig 4.4.1 Class Diagram - Models

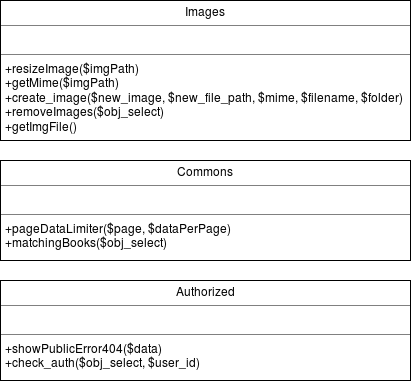


Fig 4.4.2: Class Diagram - Libraries

## **4.5 Use Case Diagram**

Working on it

Fig 4.5: Use Case Diagram

# **Chapter 5: System Development and Implementation**

## **5.1 Programming Platform**

This system has been built using PHP (hypertext preprocessor) language. It is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML. It has following features:

* It can run on both UNIX and windows servers.
* It works in combination with HTML to display dynamic elements on the page.
* It only parses PHP code and doesn't parse HTML tags so it is very efficient.
* It supports large number of relational database management system.

This system also uses a relational database to store all the data.The database that has been used here is MySQL.It has following features:

* It's easy to use since all we need to know is the basic knowledge of SQL.
* It's secure since rights can be set to allow some or all privileges to individuals and passwords are encrypted.
* It's inexpensive and is available by free download from MySQL Web site.
* It's fast and scalable since it can handle almost any amount of data, up to as much as 50 million rows or more.
* It manages memory very well and has been thoroughly tested to prevent memory leaks.
* It runs on many operating systems including Novell NetWare, Windows\* Linux\*, many varieties of UNIX\* (such as Sun\* Solaris\*, AIX, and DEC\* UNIX), OS/2, FreeBSD\*, and others.
* It supports several development interfaces including JDBC, ODBC, and scripting language such as PHP and Perl.

## **5.2 Tools & Technologies**

**5.2.1 Tools**

Even though PHP backed this system, there is yet another a type of tool/framework that has been used on top of PHP within this system. That framework is called ‘CodeIgniter’. CodeIgniter is a powerful PHP framework with a very small footprint, built for developers who need a simple and elegant toolkit to create full-featured web applications. It has following features:

* It is a framework with a small file size including complete user guide.
* It encourages MVC pattern but doesn't force it upon user.
* It is very fast as compared to other similar framework.
* It is consists of built-in protection against CSRF and XSS attacks.

**5.2.2 Technologies**

Other than CodeIgniter, this system utilizes 3 tier architecture technology. Each tier has following functionalities:

* **Presentation tier**

The top most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

* **Logic tier**

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

* **Data tier**

Here information is stored and retrieved from a database of file system. The information is the passed back to the logic tier for processing, and then eventually back to the user.

## **5.3 Testing & Debugging**

Testing is one of the crucial stages in Software Development Life Cycle. Testing is a mechanism of checking the functioning and usability of a system. Test plans are made as soon as the coding work was started. The types of testing has been done on this system is given below:

* **Function-level testing**

Under this testing level, each of the functions used in the development of the system were tested as and when implemented. Each individual component used in this project has been tested, before merging the codes.

The condition for this test to pass was to meet the functional requirement as mentioned earlier.

* **Module-level testing**

With the collection and integration of variety of functions, modules were developed which again had to be tested in order to recognize the mistakes underlying when being integrated.

This level of testing was also done after merging the codes together to see if the merging didn’t break anything within the system. Any issues that were encountered while merging or after merging were resolved immediately to reduce further issue that may be encountered in the future.

* **System-level testing / Integration testing**

It is done when the system is almost ready or when an integration task has been performed within the project. As mentioned before, the system was created by collecting and integrating required modules.

Moreover, this project has been built by reusing codes from previous project, this testing was necessary to ensure the integration didn’t hamper the system’s working mechanism.

* **Regression testing**

It is done when any changes that are made to an existing system/software. It has been done to determine whether the newly implemented fix or feature has created new problem/bug or not. This test has been performed after completing the project to see if each and everything is working fine before presenting the system to the project evaluator.

The condition required for this test to pass was not to encounter any issue while testing. Further during the testing period, there were no issue for the given set of inputs

* **Unit testing**

This test was performed using codeigniter's built-in unit testing library. To do this, below-mentioned steps needs to be followed:

* + Load built-in library: **$this->load->library('unit\_test');**
  + Once loaded, the Unit Test object will be available using **$this->unit**
  + Running a test involves supplying a test and an expected result in the following way: **$this->unit->run(‘test’, ‘expected result’, ‘test name’, ‘notes’);**
  + To run a full report of all tests, use this: **echo $this->unit->report();**

This type of test was successfully performed on all the classes within the project. All of the testing came positive during and at the end of the system development.

## **5.4 Implementation & Result**

**5.4.1 Implementation**

After development and testing, this is the final stage of this project. For actual implementation of the project, at least, following tools and technology must be used.

* **Hardware requirements**
  + **Processor:** Pentium IV or greater.
  + **Operating System:** any OS (32 bit or 64 bit)
  + **RAM:** at least 512 MB or greater
  + **HDD:** at least 200 MB for project deployment only
* **Software requirements**
  + **Database Server:** MySQL Ver. 5.1.36 or greater (for data storage)
  + **Application Server:** Apache 2.4.9 and PHP 5.5.12 or greater (for application deployment)
  + **Browser:** any modern browser (user’s choice)

**5.4.2 Result**

As mentioned in the **Requirement Specification** section, this project has tired to meet all of its functional as well as non functional requirements. This project has tried to bring different bus agency into a single platform. This has been achieved by providing separate panel for different agency admins and and member of that agency within this system.

## **5.5 Maintenance**

Like any other software project, this project also may contain unseen bugs and issue that may not have been covered by our testing methods. There is always chance of some issue to come over time. Since, we are already aware of such unseen possible issues, an oop language has been used to build this project to help prevent the hassle of doing maintenance task on the system. A simple but powerful framework will allow developers to the fix issues much faster compared to custom codes develop.

Not only that, the framework chosen is so easy to use that adding other enhancement to the system will be faster and easier. Thus, use of codeigniter framework has made it easy for the current developers as well as future developers to add new feature and fix issues within this system.

# **Chapter 6: Conclusion and Future Enhancement**

**6.1 Conclusion**

Based on the **Literature Review,** this type of system has already been created by other individuals prior this project. But it tries to solve one of the major problem. That problem is easy availablity of buses for journey. Compared to similar systems available on the market, this system tires its best to link bring various bus agency into a single syste,. As mentioned earlier (See **System Analysis** for more details), this system provides facility to browse bus, select seat of your desire and finally pay for the booked seats. Further, it allows user to search for their ticket using the unique ticket id provided during ticket purcase. This allows the user and the bus agency to cross check the booking details.

**6.2 Future Enhancement**

The following list of enhancements can be done to this project in the future:

* Adding support for mobile devices.
* Provide online payment system such as e-sewa and i-pay.
* Ability for bus agency to add their own type of seat layout.

# **References**

[1] Eckerson, Wayne W. "Three Tier Client/Server Architecture: Achieving Scalability, Performance, and Efficiency in Client Server Applications." Open Information Systems 10, 1 (January 1995): 3(20)

[2] Buschmann, Frank; Meunier, Regine; Rohnert, Hans; Sommerlad, Peter; Stal, Michael (1996-08). Pattern-Oriented Software Architecture, Volume 1, A System of Patterns. Wiley, August 1996. ISBN 978-0-471-95869-7. Retrieved from <http://www.wiley.com/WileyCDA/WileyTitle/productCd-0471958697.html>.

[3] Pigoski, Thomas M., 1997: Practical software maintenance: Best practices for managing your software investment. Wiley Computer Pub. (New York)

[4] Lientz B., Swanson E., 1980: Software Maintenance Management. Addison Wesley, Reading, MA

[5] Sarah, Geagea; Sheng, Zhang; Niclas,Sahlin; Faegheh, Hasibi; Farhan, Hameed; Elmira, Rafiyan; Magnus, Ekberg; (2010). Software Requirements Specification. Retrieved from <http://www.cse.chalmers.se/~feldt/courses/reqeng/examples/srs_example_2010_group2.pdf>

[6] Lerdorf, Rasmus (2007-04-26). "PHP on Hormones – history of PHP presentation by Rasmus Lerdorf given at the MySQL Conference in Santa Clara, California". The Conversations Network.

[7] "Rasmus Lerdorf, Senior Technical Yahoo: PHP, Behind the Mic". 2003-11-19. Archived from the original on 2013-07-28., cited at Felipe Ribeiro (Aug 7, 2012). "This Is Not Another "PHP Sucks" Article".

[8] Pawel Krawczyk (2013). "Most common attacks on web applications". IPSec.pl.

[9] Golemon, Sara (2006). Extending and Embedding PHP. ISBN 978-0-672-32704-9.

[10] https://www.quora.com/How-do-I-show-attributes-with-a-foreign-key-in-ER-diagrams