**Guest House Management System**

****

Saugat K.C.

00172898

Email: *thelastresort35@gmail.com*

Computing Project

Level 5 Diploma in Computing

Softwarica College of IT & E-Commerce

Kathmandu, Nepal

January 4, 2019

Submitted to: Kiran Rana

# Abstract

Trisa Guest house management system is a web based application which manages the data of Trisa Guest House and where their customer can view details about the Guest House and use the facility provided. Customer can view available rooms accordingly to their specification and book the room for their specified date and time. This makes easier for customer to make their booking and absolute data of room can view easily without visiting the guest house.

For the development of this project Laravel Framework 5.8.3, XAMMP 3.2.2, and text editor sublime are used. For achieving the goal of the project different procedures and techniques have been implemented.

# Acknowledgement

I would like to express my special thanks of gratitude to my teacher Kiran Rana who gave me the golden opportunity to do this project on Trisa Guest House Management System which helped me a lot in learning new things and polishing my skill of research as well as communication skill.

Furthermore, I would like to thank my friend who helped me when I was stuck and taught me the things that I didn’t know and helping me to learn many new things. The efforts and guidance given by my teacher and friends came to the success with the completion of the project.

Contents

[Chapter 1- Introduction 2](#_Toc5136192)

[Project Introduction: 2](#_Toc5136193)

[Problem Statement 2](#_Toc5136194)

[Background of the project 2](#_Toc5136195)

[Aims and Objectives 2](#_Toc5136196)

[Features 3](#_Toc5136197)

[Chapter 2- Analysis 3](#_Toc5136198)

[Methodology used 3](#_Toc5136199)

[Design Pattern 5](#_Toc5136200)

[System Architecture 6](#_Toc5136201)

[Requirements 7](#_Toc5136202)

[MoSCoW Prioritization 7](#_Toc5136203)

[Functional and Non-Functional 8](#_Toc5136204)

[Use-Case diagram 9](#_Toc5136205)

[Initial class diagram 11](#_Toc5136206)

[Chapter 3- Design 12](#_Toc5136207)

[Structural design 12](#_Toc5136208)

[Final class diagram 12](#_Toc5136209)

[Behavioral Diagram 13](#_Toc5136210)

[Sequence Diagram 13](#_Toc5136211)

[Activity Diagram 13](#_Toc5136212)

[Data Model 15](#_Toc5136213)

[ER-Diagram 15](#_Toc5136214)

[UI design 15](#_Toc5136215)

[Chapter 4- Implementation 19](#_Toc5136216)

[Chapter 5- Testing 19](#_Toc5136217)

[Limitation and Future Work 20](#_Toc5136218)

[Limitation 20](#_Toc5136219)

[Future Work 20](#_Toc5136220)

[Risk and Configuration Management 20](#_Toc5136221)

[Risk Management 20](#_Toc5136222)

[6. Configuration Management 22](#_Toc5136223)

[Conclusion 22](#_Toc5136224)

[References 23](#_Toc5136225)

# Chapter 1- Introduction

## Project Introduction:

Guest house management system is a web based application designed for the guest house to manage their transactions as well as to give information about the guest house and get the required information’s that is available. Customer can view details about the Guest House and use the facility provided. Customer can view available rooms accordingly to their specification and book the room for their specified date and time. This makes easier for customer to make their booking and absolute data of room can view easily without visiting the guest house.

## Problem Statement

Giving information about the guest house to customers is not so smoot as well as the proper details of the facilities provided by the guest house to the customers. People often have to visit the guest house to book or view the room and know the prices of rooms, packages provided by the guest house. Some customers have to return with dissatisfaction if the facilities are not as they except. Customers sometime have to return due to all available rooms pack and the details of the rooms occupied or empty have to be saved in paper which gives employee little bit of hard time. With this project customers can view proper information’s about the guest house online and book the rooms online. Customer can also see if rooms are packed or not as well as makes the employee easier to save the information about the rooms.

### Background of the project

People trying to minimize their expenses when they are going to places far from home so likely than hotels they prefer guest houses. Using the website of the guest house customers and view the room picture and their prices and if like the room can book it online. Customers can see if the room is empty, booked or occupied. Customer can get other information of the guest house through the website like any packages and other.

## Aims and Objectives

**Aim:**

* Provide up-to date information to the customer
* Develop user friendly and helpful application
* Proper data management
* Give more facilities to the customer

**Objectives:**

* All information about the room their packages and facilities provided by the guest house including their price is shown.
* Information of booked room, occupied room and empty room are updated
* Information of the customer as well as the information of the guest house is properly saved as well as their expenses.
* reviews from customer are seen for further development of the guest house
* Rooms can be booked online if empty for their date required and select the room they prefer

## Features

The features of the project are as follows:

* Room :

Admin can add and edit and delete rooms in the website

* Review :

Customer can give review to the guest house

* Booking:

Customers can check for empty room and book it online

* Billing:

Expenses done by the customer is saved and afterward generated in bill

* Menu:

Customer can view the menu provided and order the items they want if they are currently checked in

* Menu Item:

Admin can add, update and delete the items on their menu

* Expenses:

Currently staying user can check their expenses

* Stays:

User can view details of their old stays, their booking and currently staying

* Bill:

User can view bills of their old stays

# Chapter 2- Analysis

## Introduction

Analysis is a phase from where project lifecycle begins. It is the phase for identifying the problems, errors and looking for the methods to solve them. Analysis gives the whole overview of the system as it helps to determine the problem domain.

## Methodology used

Development methodology is a framework that is used to plan, structure and control the process of developing an information system. For my development, I will use waterfall methodology. As it is a sequential design approach, it is easy to manage due to the rigidity of model.

**Waterfall model** is very simple methodology of software development. There are sequential steps which are proceeded after the completion of one due to which there is no overlapping during the process. The steps involved in this methodology are:

* **Requirement gathering and analysis:**The first phase involves understanding what need to be design and what is its function, purpose etc.
* **System Design:** The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and helps in defining overall system architecture.
* **Implementation:** With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.
* **Testing:** All the units developed in the implementation phase are integrated into a system after testing of each unit.  Testing is done so that the client does not face any problem during the installation of the software.
* **Deployment of System:** Once the functional and non-functional testing is done, the product is deployed in the customer environment or released into the market.
* **Maintenance:** This step occurs after installation and involves making modifications to the system to improve performance. These modifications arise due to change requested by the customer. Client is provided with regular maintenance and support for the developed software.

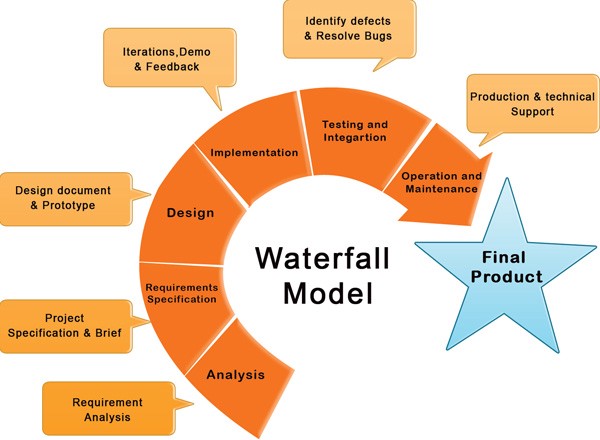


Figure 1. Waterfall Model

## Design Pattern

Among the various types of design pattern like Factory pattern, prototype pattern, Model View Controller, Singleton pattern etc. I have chosen the Model View Controller (MVC) pattern.

Model view controller is an architectural pattern commonly used for developing user interfaces that divides an application into three interconnected parts. This pattern is used to separate application’s concerns. In this pattern the model is known as the central component of the pattern which directly manages the data, logic and rules the application. A view can represent the information of chart diagram, bar chart etc. And the Controller accept the input which convert it to command for mode and view. The reasons to choose this pattern are as follows:

* **Faster development process:**
* **Modification does not affect the entire model**
* **Ability to provide multiple views**
* **Support for asynchronous technique.**

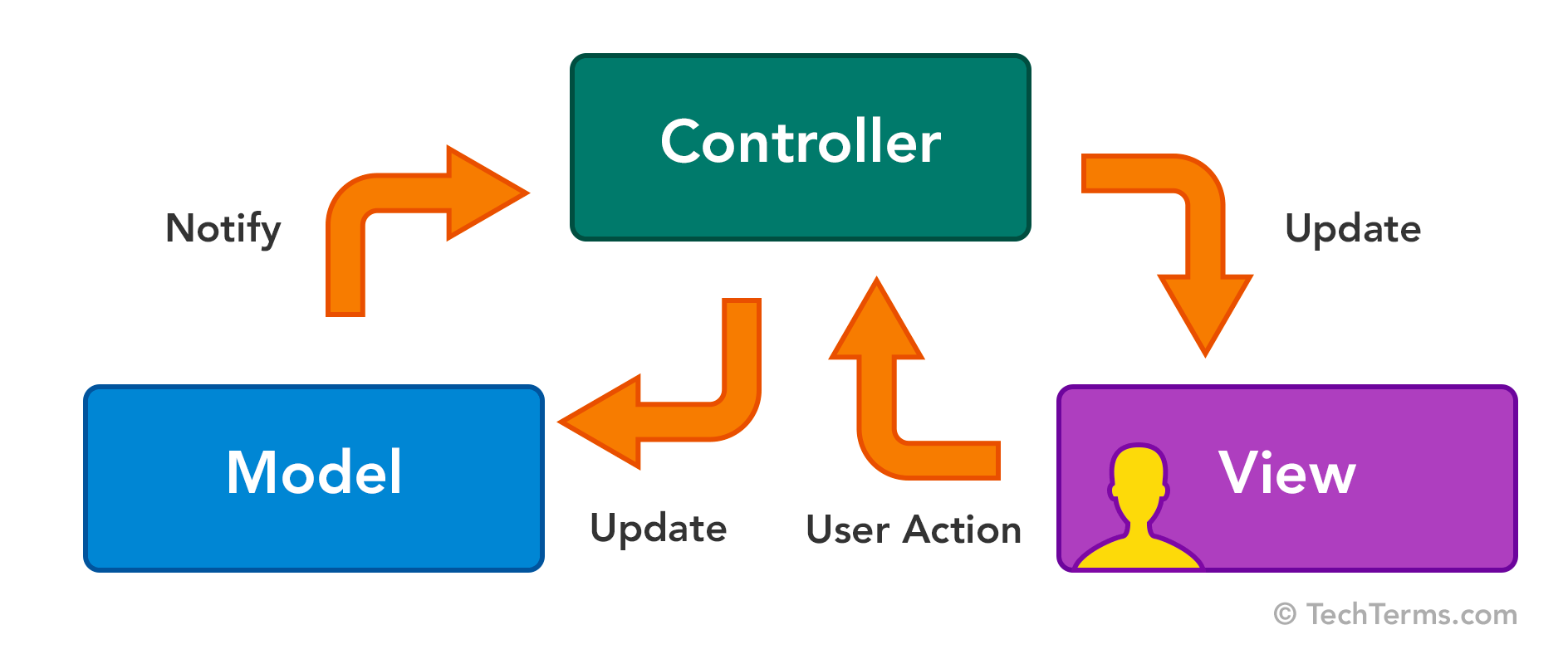


Figure 2. MVC

## System Architecture

I have decided to use three tire architecture. Three-tier architecture is a client-server architecture in which the functional process logic, data access, computer data storage and user interface are developed and maintain independent modules on separate platform. The three tier in this architecture are:

Client tier:

This is a top level which display the information related to the services which is available in this website.

Application tier:

It is also known as the middle tier of the application. This tier is pulled from the upper tier known as presentation tier and control the functionality by performing detail processing.

Data tier:

It is a server where the information is stored and retrieve. Data in this tier kept independent of application servers.



Figure 3. Three tire architecture

## Requirements

### MoSCoW Prioritization

In this phase, all the functional and non-functional requirements are prioritized. All of them are categorized according to their value to the system. Prioritization will finalize the system functionalities that needs to be developed at the beginning. Without the prioritization, all the requirements will be a mess and becomes difficult to make decision which are the important ones. (bawiki, 2019) MoSCoW prioritization is the most suited tool for prioritization of the requirements of system. It includes factors such as must have, should have, could have and won’t have which are briefly described below:

Must Have (Mo) – must have requirements falls under this priority.

Should Have (S) – the requirements that can be needed falls under this priority.

Could Have (Co) – the requirements that could have been on the system falls under this priority.

Won’t Have (W) – the requirements that doesn’t fit and won’t be implemented falls under this priority.

### Functional and Non-Functional

**Functional requirements** are those which are related to the technical **functionality** of the system. (stackoverflow.com, 2019)

|  |  |  |  |
| --- | --- | --- | --- |
| **SN** | **Requirements** | **Justification** | **MoSCoW** |
| F1 | Signup | Should be able to register user | Must Have |
| F2 | Admin Login | Admin should be able to login into admin control | Must Have |
| F3 | Customer login | Customer should be login to homepage | Must Have |
| F4 | Edit user details | User should be able to update their details | Should Have |
| F5 | Add new room | Admin should be able to add new rooms | Must Have |
| F6 | Show room details | All room details should be view by the admin | Must Have |
| F7 | Update room details | Admin should be able to update rooms details | Must Have |
| F8 | Search rooms | Customer should be able to search rooms for specific dates | Must Have |
| F9 | Book room | Customer should be able to book room for specific date | Must Have |
| F10 | View currently staying guest | Admin can view all the guest currently staying | Should Have |
| F11 | Customer reviews | Customers should be able to give feed back | Should Have |
| F12 | View users all booking | User should be able to view all their bookings till date | Should Have |
| F12 | View location | Location of the guest house should be shown | Could Have |
| F13 | Online order | Customer should be able to order online | Could Have |
| F14 | Manually order add | Orders can be added manually | Must Have |
| F15 | Create final bill | Final bill should be generated | Must Have |
| F16 | Online payment | Customer may be able to pay online | Could Have |

**Non**-**functional requirement** are those requirements that specifies criteria that can be used to judge the operation of a system in particular conditions, rather than specific behaviors (stackoverflow.com, 2019)

|  |  |  |  |
| --- | --- | --- | --- |
| **SN** | **Requirements** | **Justification** | **MoSCoW** |
| NF1 | User Friendly | The project should be user friendly | Should Have |
| NF2 | Customer privacy | Guests privacy should be maintained | Must Have |
| NF3 | Maintainability | The application should be maintainable | Must Have |
| NF4 | Data integrity | Data integrity should be maintained | Should Have |
| NF5 | Reliable | The application should be reliable | Must Have |
| NF6 | Security | Security should be maintained | Must Have |
| NF7 | Usability | The application should be useable | Must Have |

## Use-Case diagram

A **use case diagram** is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. (techtarget, 2019)

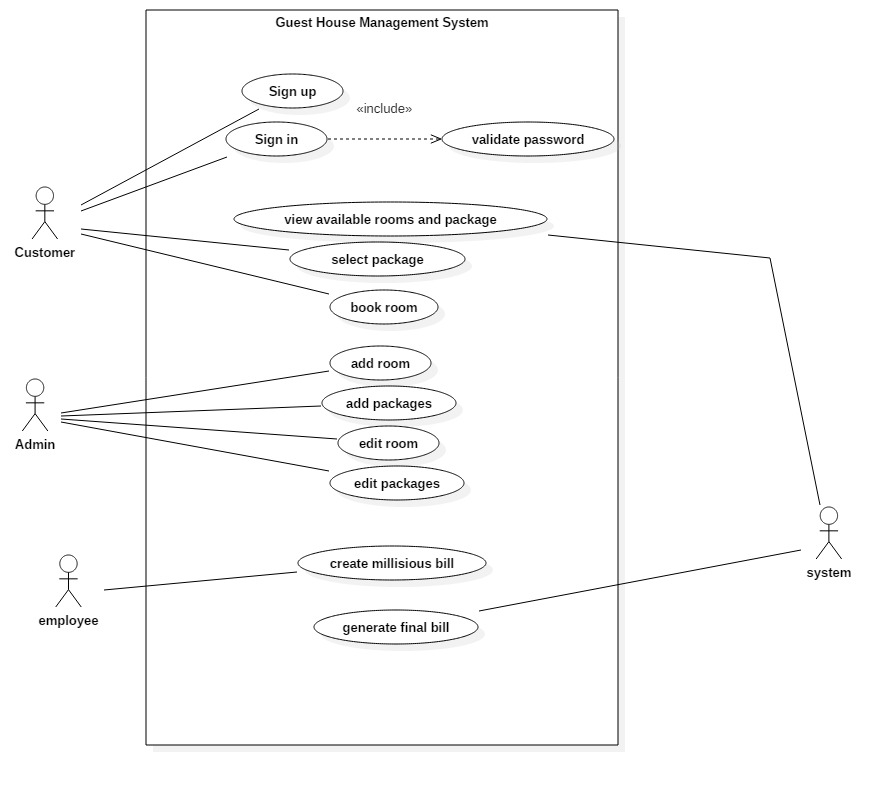


Figure 4 use case diagram

**Justification of Use case diagram**

This diagram is used to identify the functionality of system. The diagram includes boundary, actors, use cases and relations. A complete use case diagram defines the actions that can be performed in the system, which is called use cases, roles of individuals performing the actions which is actors and the relation between actors and use cases. Use case diagram is important for understanding the behavior of system being developed and describing how users accomplish their goals.

## Initial class diagram

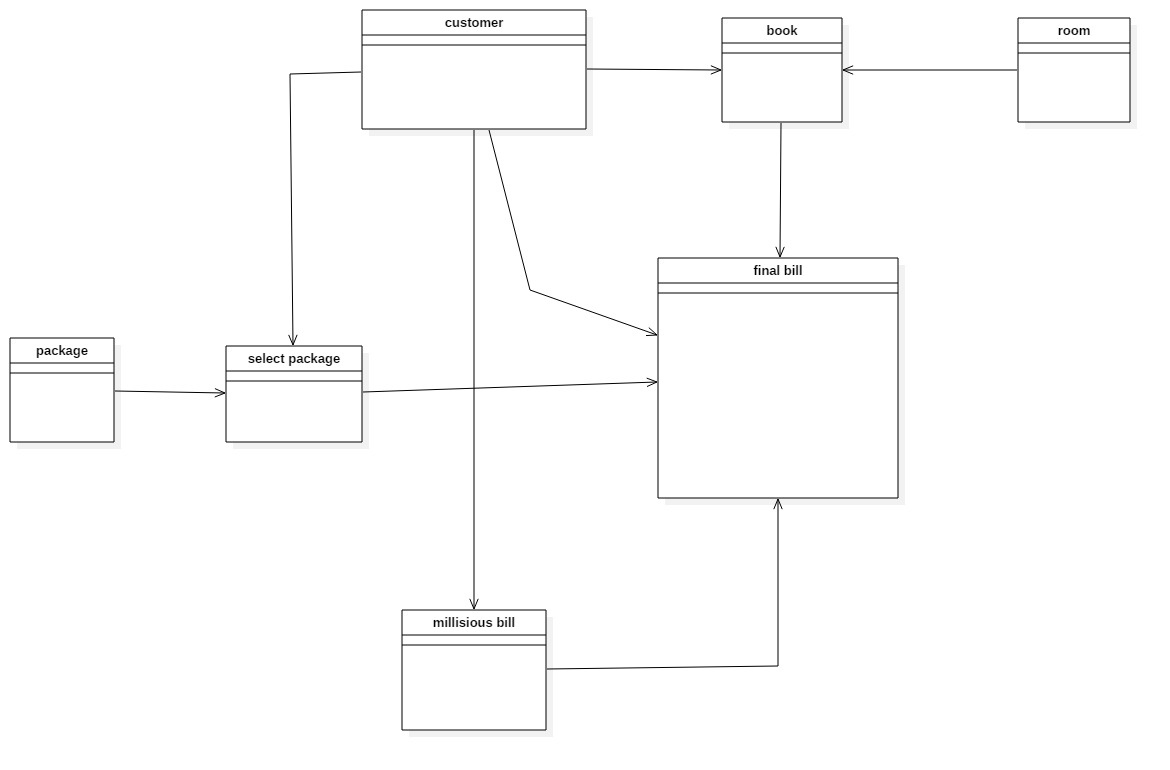


Figure 5 initial class diagram

# Chapter 3- Design

## Structural design

### Final class diagram

A class diagram is an illustration of the relationships and source code dependencies among classes in the Unified Modeling Language (UML). (techtarget, 2019)

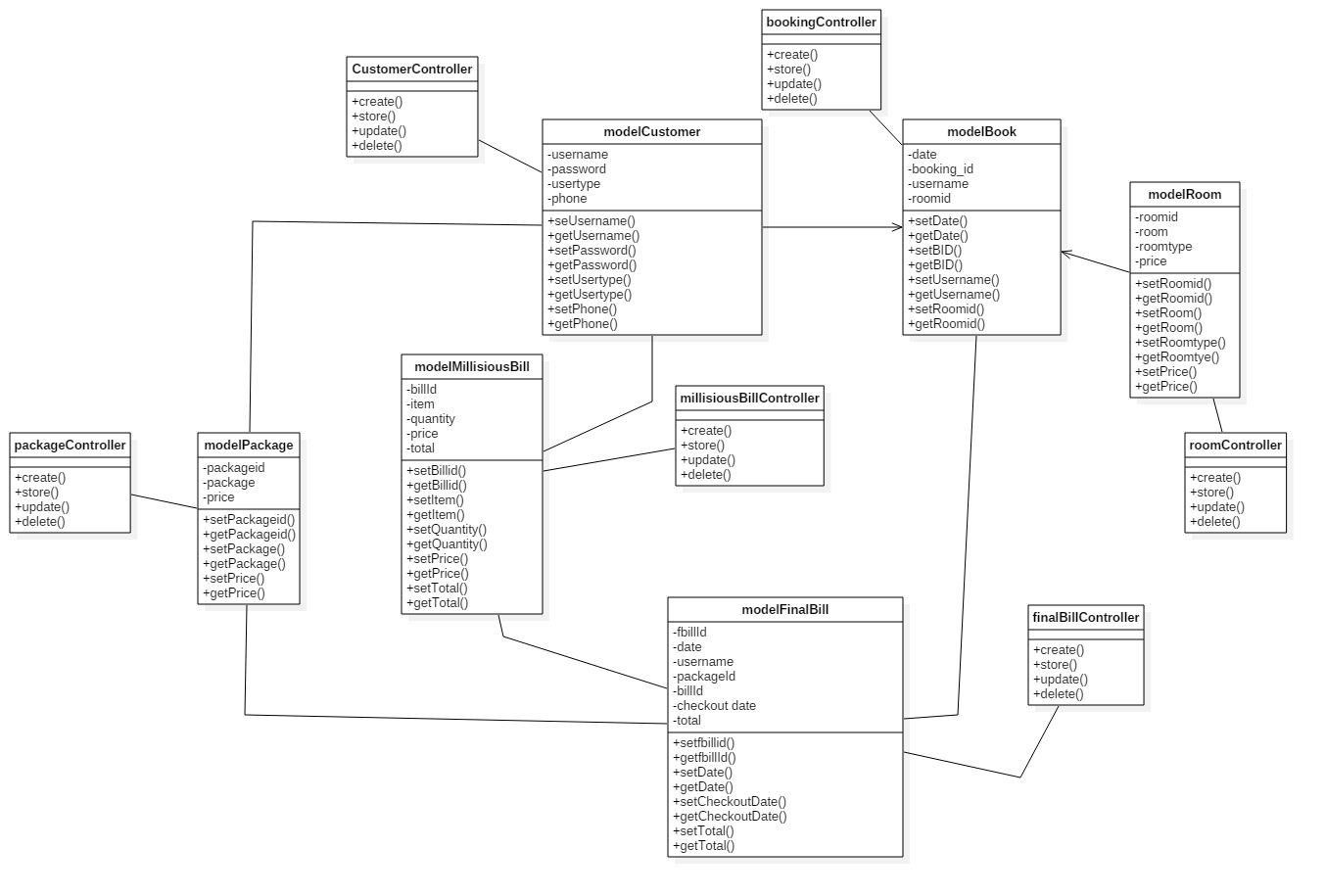
****

Figure 6 final Class Diagram

**Justification of class diagram**

This diagram helps us on describing the system by illustrating the relationship of its classes, their attributes and operations. It has different relationship like aggregation and association which helps us to visualize the path between the classes. So, to describe and understand the detail structure of system, class diagram is vitally important in software development.

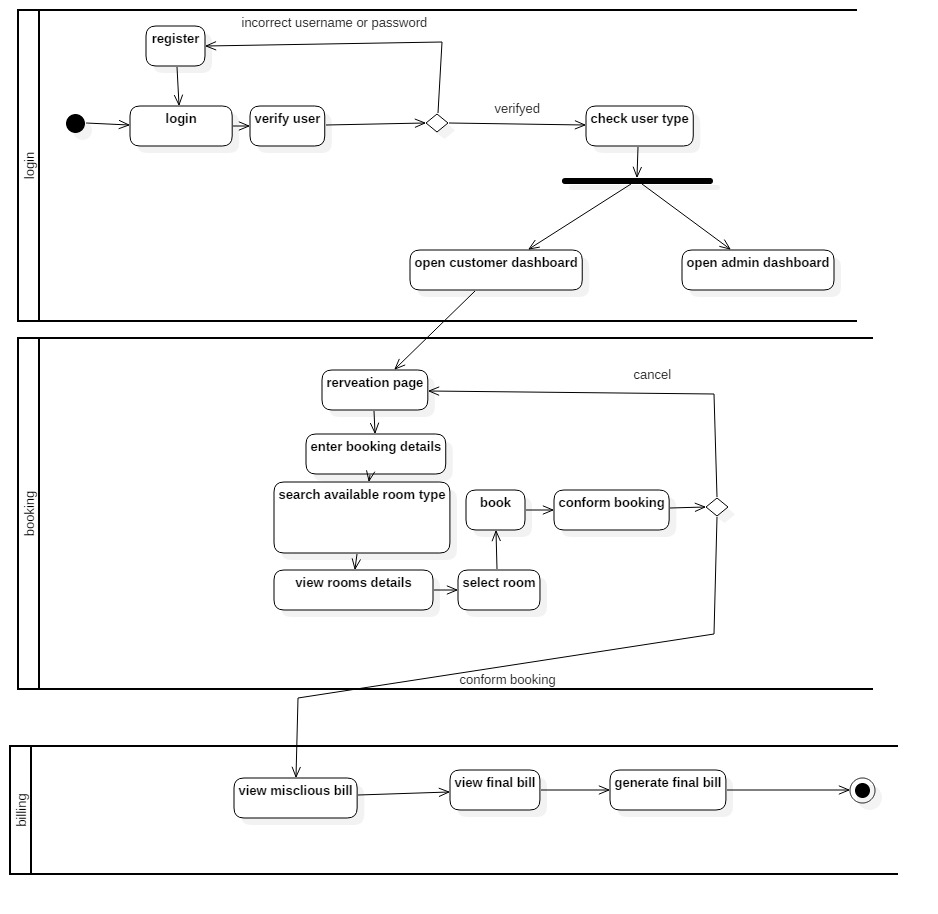
## Behavioral Diagram

### Sequence Diagram

**Sequence diagram** is kind of diagram that shows how the objects in a system interact among each other. This is time focused and shows the lifeline of objects**.** (Viasual-Paradigm, 2018)

### Activity Diagram

**Activity diagram** is another important diagram in UML to describe the dynamic aspects of the system. Activitydiagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. (tutorialspoint, 2019)

****

**Justification of Activity diagram**

This diagram is also known as an event diagram as it shows the events taking place in the system in a particular order. It shows the behavior of the objects that live simultaneously, and the messages exchanged between them. The diagram is important to design for the clear visualization of the objects of system interacting in a time sequence.

## Data Model

### ER-Diagram

An **entity-relationship diagram** (**ERD**) is a data modeling technique that graphically illustrates an information system's entities and the relationships between those entities. (techopedia, 2019 )

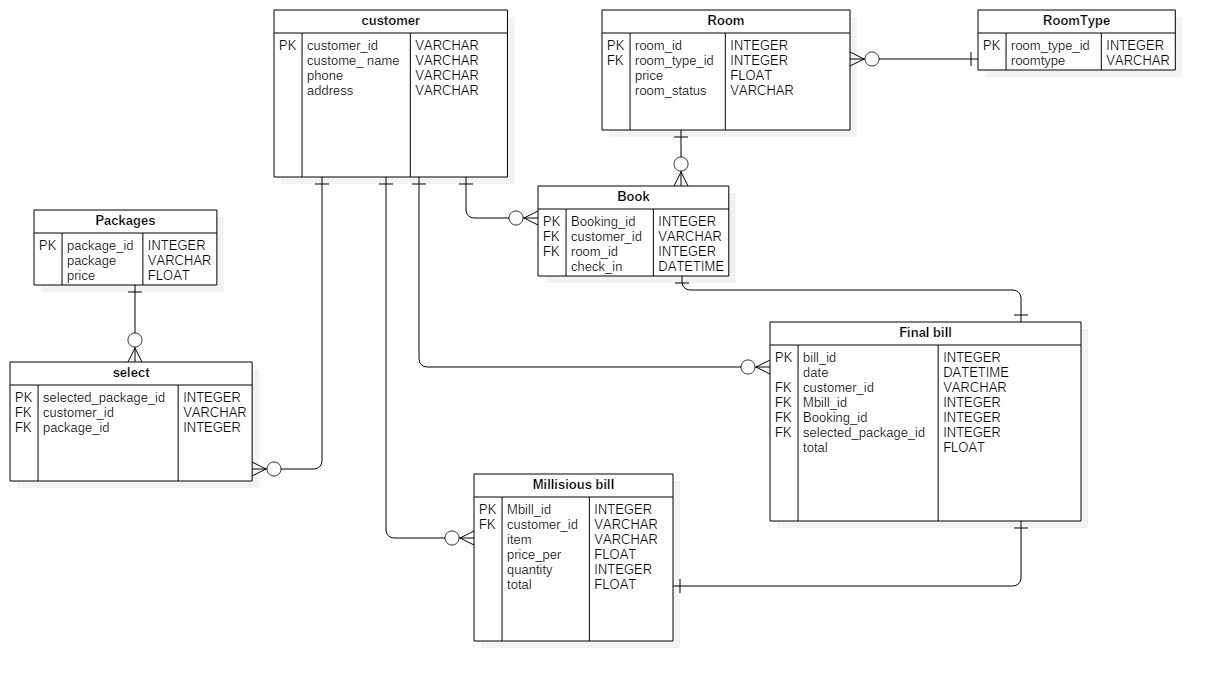


Figure 7 Er-diagram

**Justification of ER diagram**

This is also called conceptual model that represents infrastructure of entity framework. This is an important diagram for database design as it visualizes the entity relationship and shows the data structure as well as the logical database structure of the system.

## UI design

**User interface** (**UI**) **design** is the process of making interfaces in software or computerized devices with a focus on looks or style.

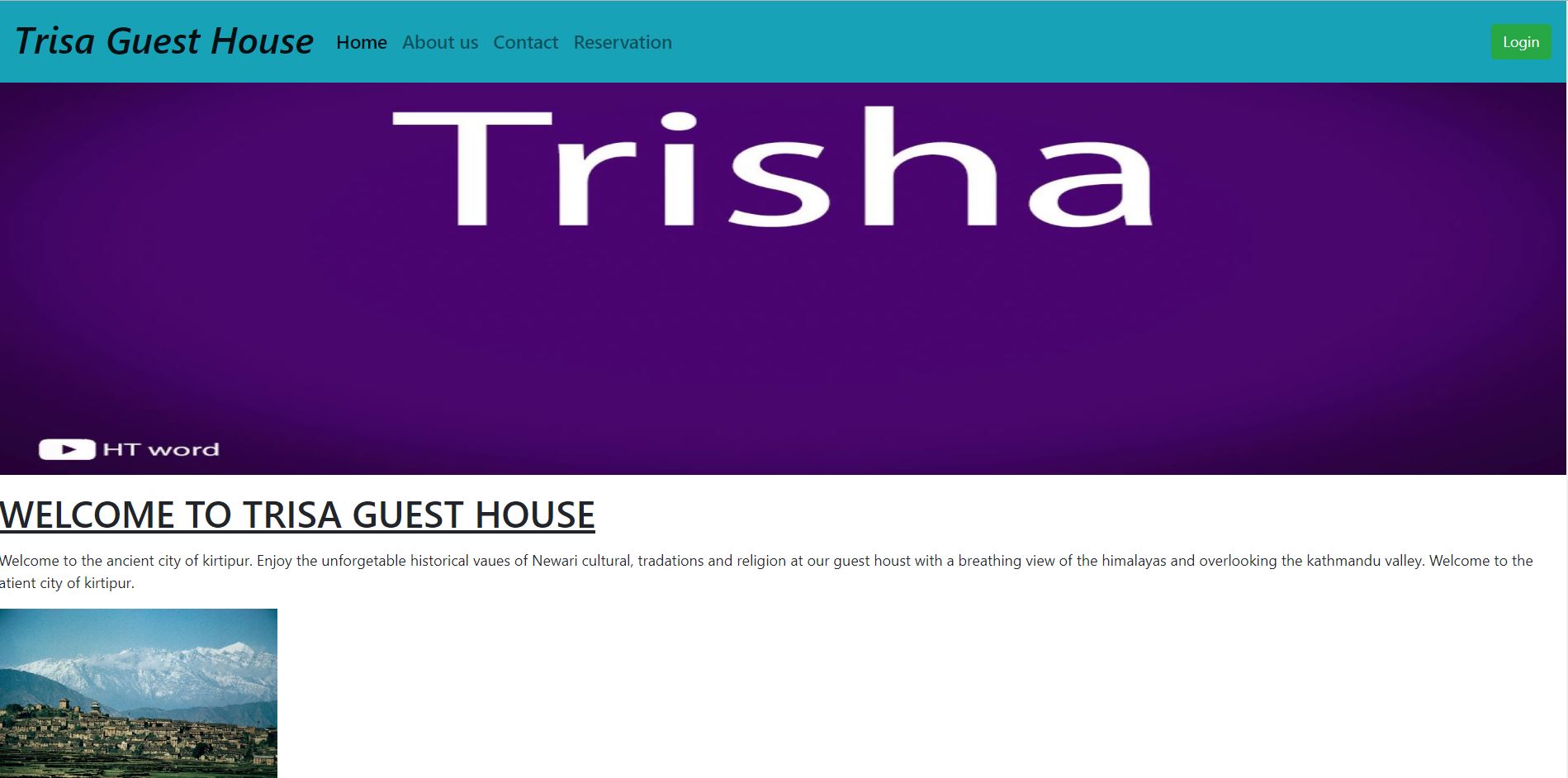


Figure 8: home

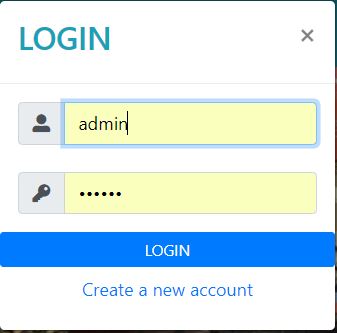


Figure 9 login

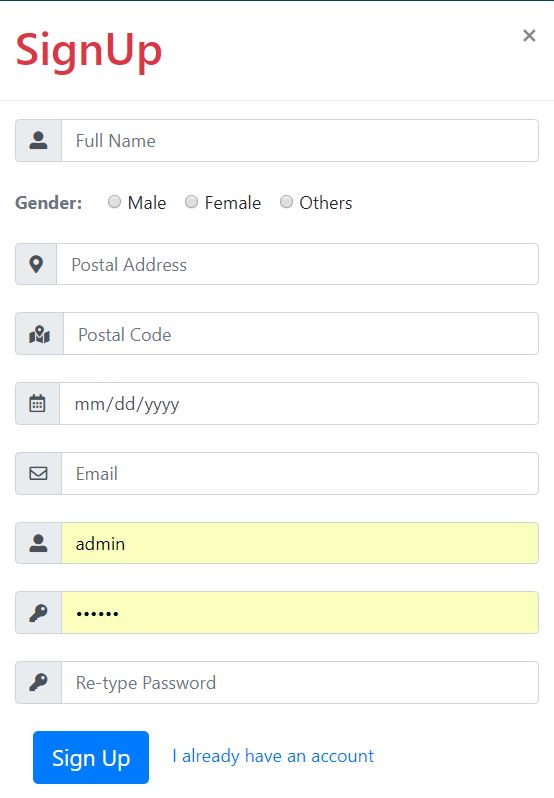


Figure 10: signup

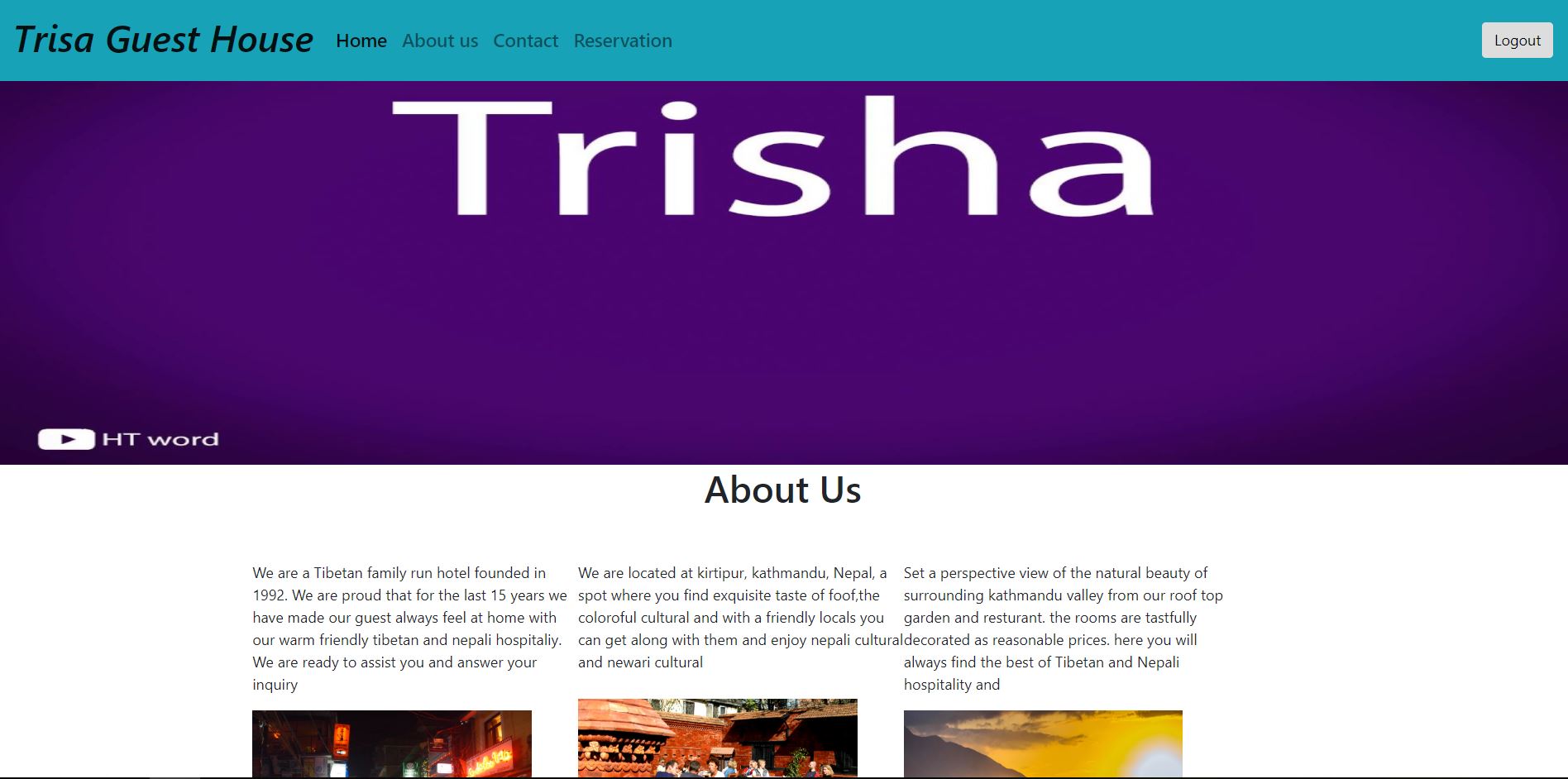


Figure 11 about us

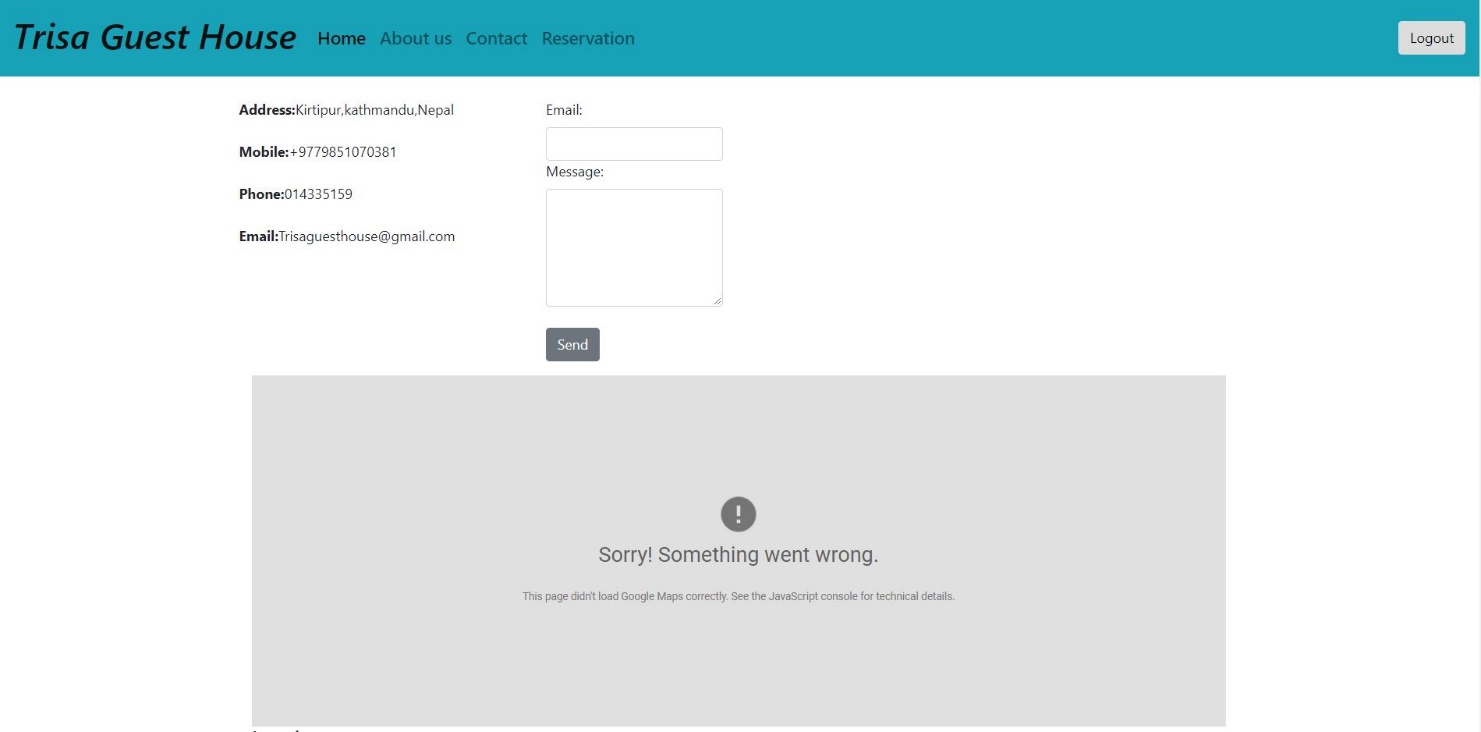


Figure 12 contact us

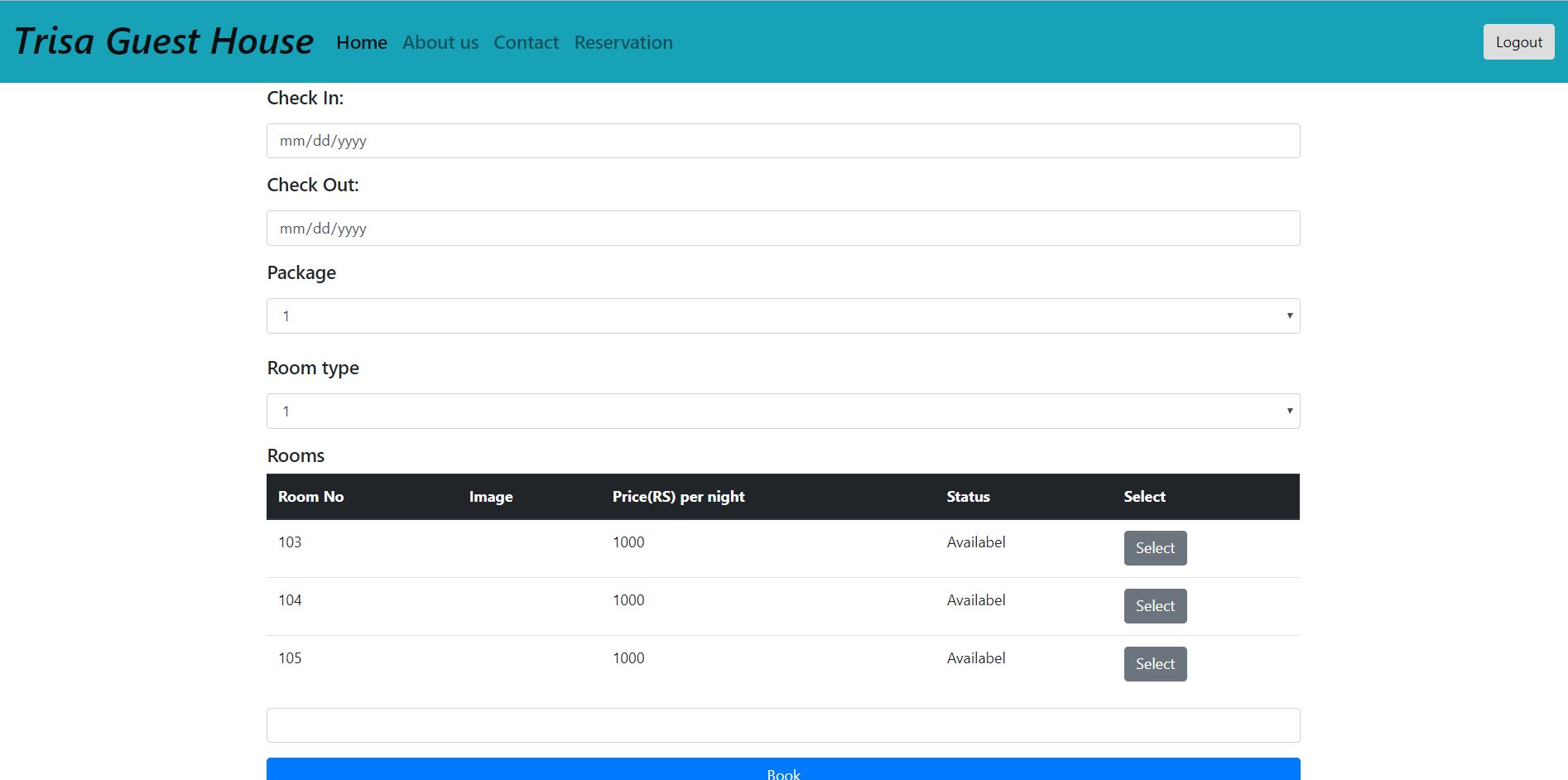


Figure 13 reservation

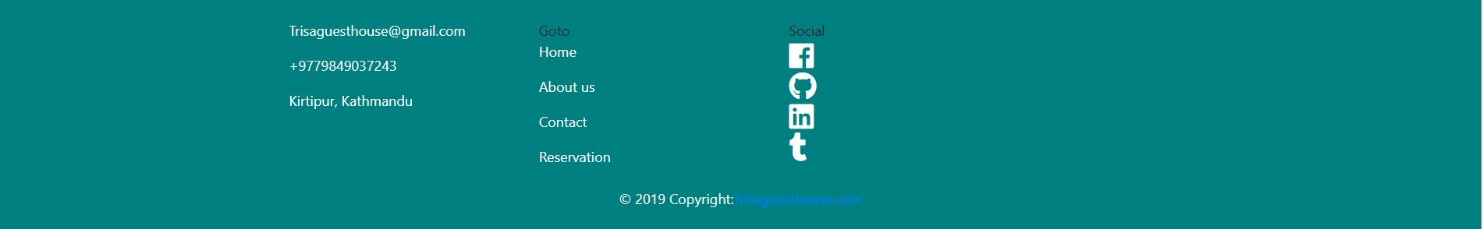


Figure 14 footer

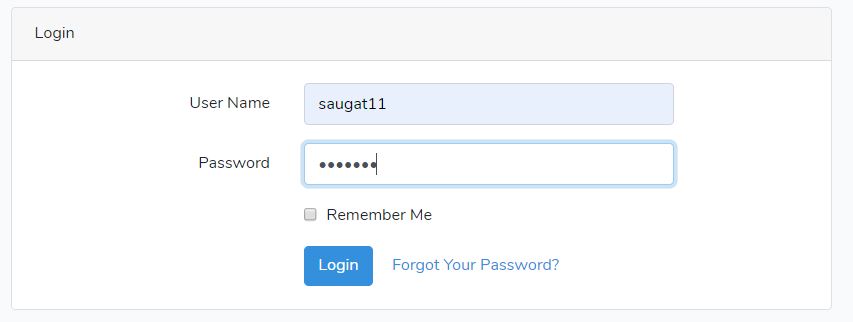
# Chapter 4- Implementation

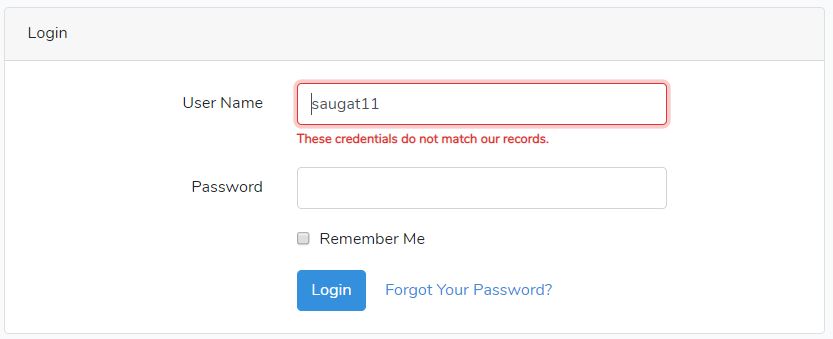
# Chapter 5- Testing

Testing is the process of evaluation of the functionality and the quality of the system. The process ensures that the actual result matches the expected result of the product. This is the fourth phase of the system development process which is very important to conduct. This provides the assurance to the stakeholders that the product is defect free as the errors are find out and removed. Testing helps to improve the overall system as it checks and maintains the quality, efficiency, reliability, and system performance. The process involves execution of system components for evaluating one or more properties.

|  |  |
| --- | --- |
| **Test case no.** | **1** |
| Purpose of the case | Checking if user can login or not |
| Test data | username:saugat11  password: saugatk |
| Class name | UserTest |
| Function name | testLogin |
| Expected result | Successful login |
| Actual result | Login unsuccessful |
| Conclusion | **No,** provided data was incorrect |

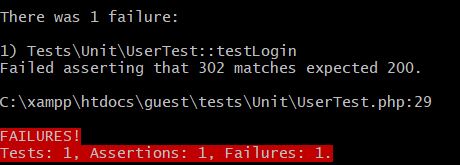
**Black Box**

****

****

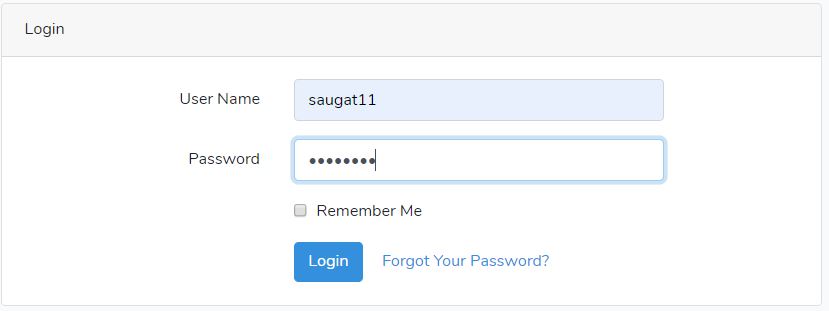
**Unit testing**

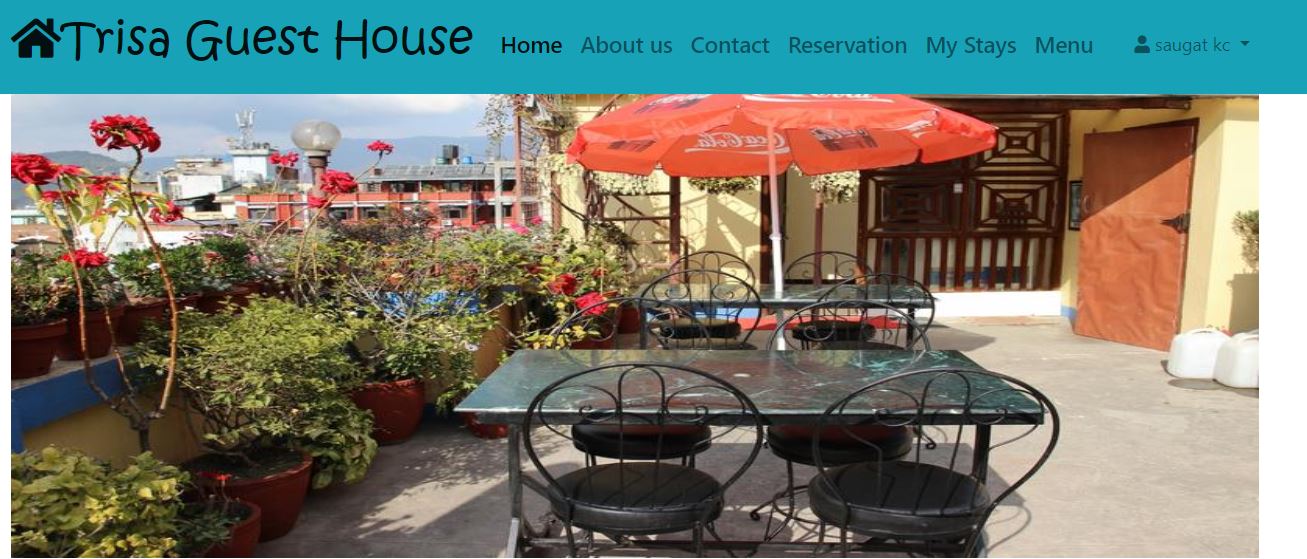
****

****

|  |  |
| --- | --- |
| **Test case no.** | **2** |
| Purpose of the case | Checking if user can login or not |
| Test data | username:saugat11  password: saugatkc |
| Class name | UserTest |
| Function name | testLogin2 |
| Expected result | Successful login |
| Actual result | Successful login |
| Conclusion | **Yes,** expected result matched actual result |

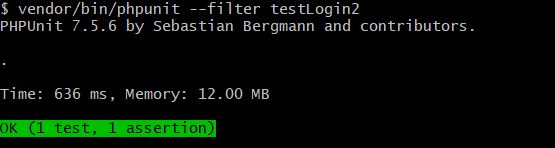
**Black box**

****

****

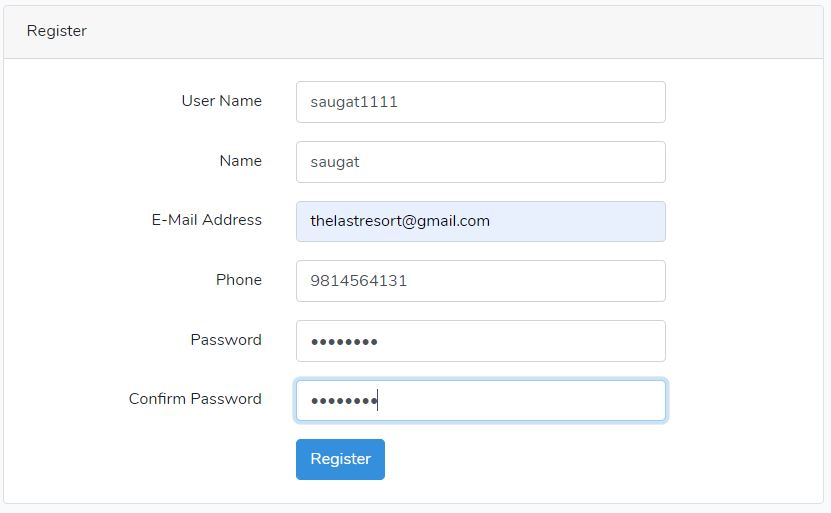
**Unit testing**

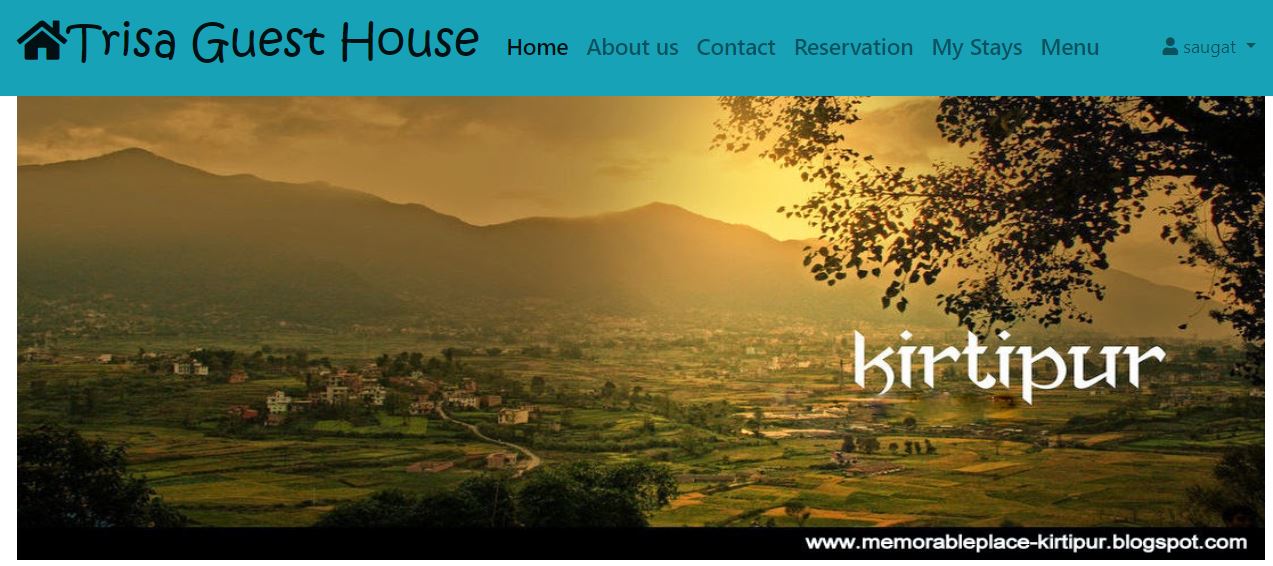
****

****

|  |  |
| --- | --- |
| **Test case no.** | **3** |
| Purpose of the case | Checking new registration |
| Test data | 'userName' => "saugat1111",  'name' => "saugat",  'email' => "thelastresort@gmail.com",  'phone' => "9814564131",  'password' => "saugatkc",  'password\_confirmation' => "saugatkc", |
| Class name | UserTest |
| Function name | testRegister |
| Expected result | Registration successful |
| Actual result | Registration successful |
| Conclusion | **Yes,** expected result matched actual result |

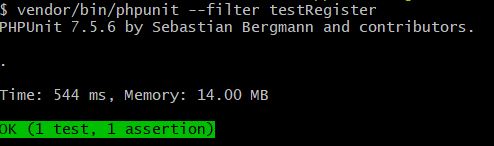
**Black box**

****

****

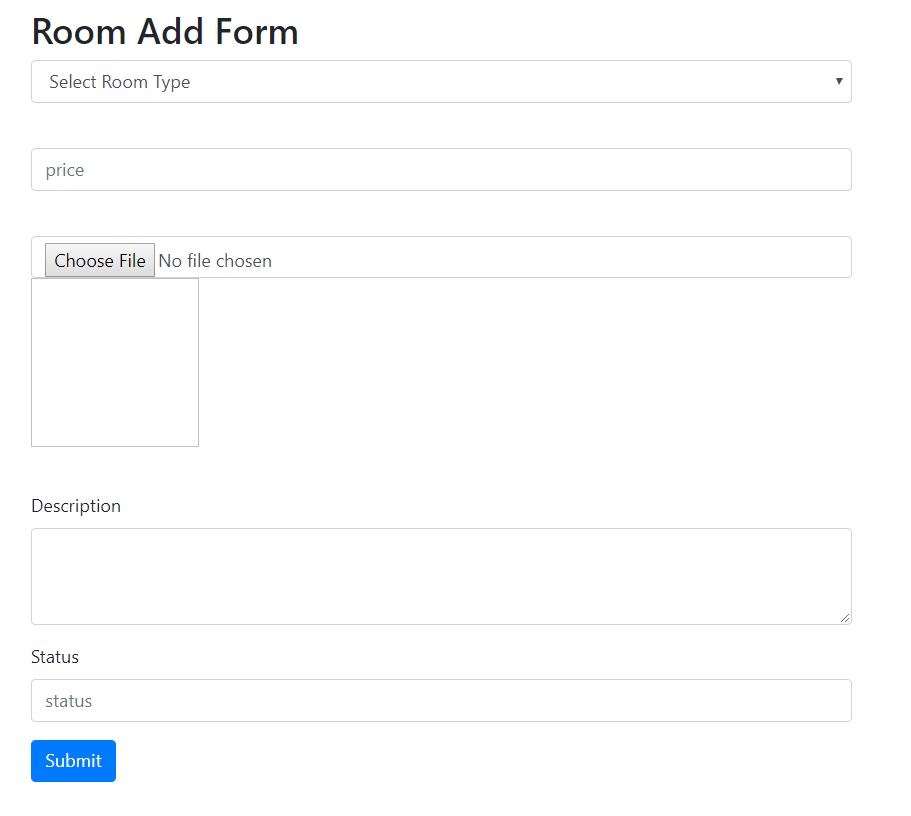
**Unit testing**

****

****

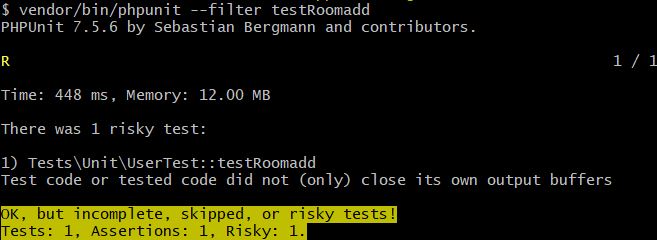
|  |  |
| --- | --- |
| **Test case no.** | **4** |
| Purpose of the case | Getting room add form |
| Test data |  |
| Class name | UserTest |
| Function name | testRoomform |
| Expected result | Fetch room add form |
| Actual result | Fetch room add form |
| Conclusion | **Yes,** expected result matched actual result |

**Black Box**

****

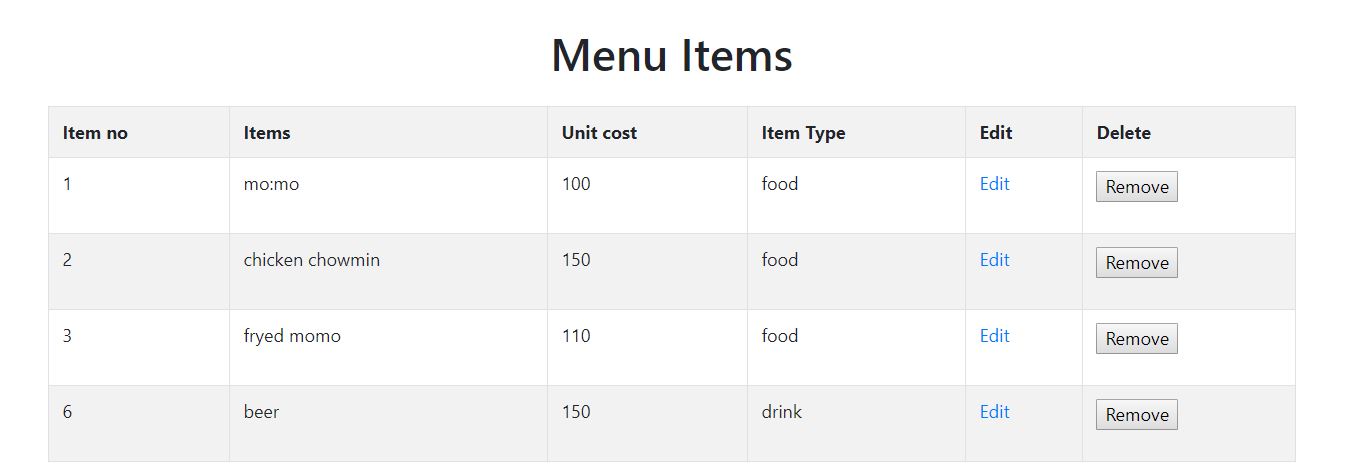
**Unit testing**

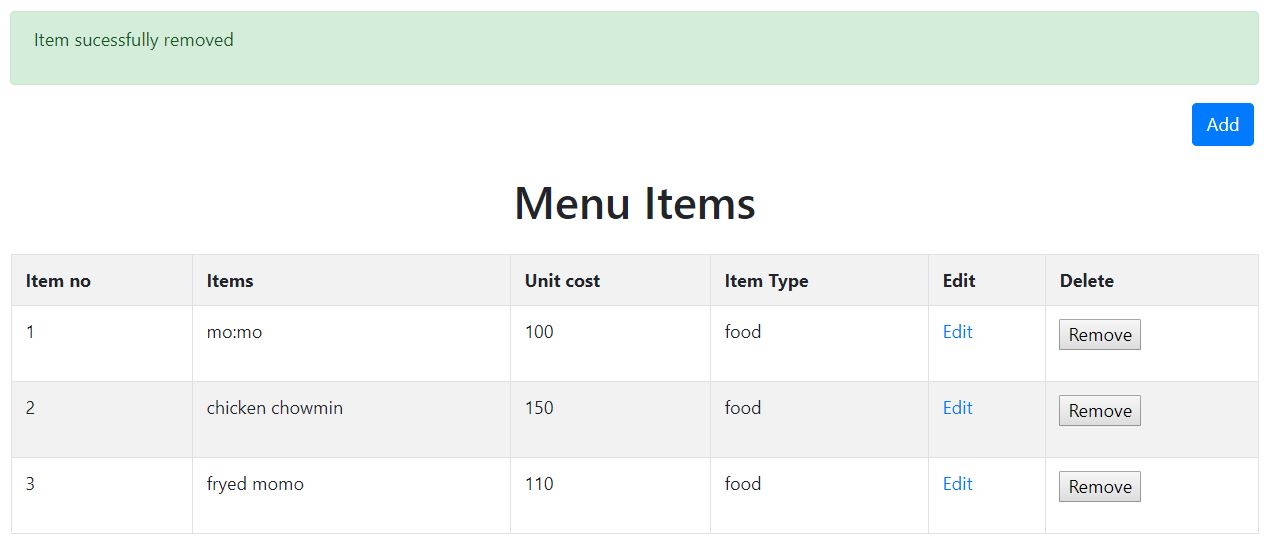
****

****

|  |  |
| --- | --- |
| **Test case no.** | **5** |
| Purpose of the case | Checking if selected menu item is deleted |
| Test data | 4 |
| Class name | UserTest |
| Function name | testItemDelete |
| Expected result | Item successfully deleted |
| Actual result | Item successfully deleted |
| Conclusion | **Yes,** expected result matched actual result |

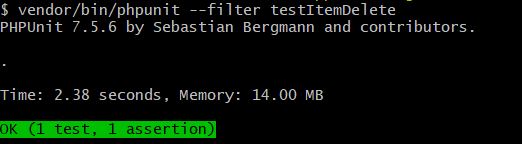
**Black Box**

****



**Unit testing**

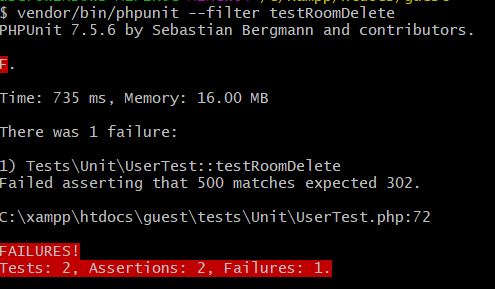
****

****

|  |  |
| --- | --- |
| **Test case no.** | **6** |
| Purpose of the case | Checking if room can be deleted |
| Test data | 1 |
| Class name | UserTest |
| Function name | testRoomDelete |
| Expected result | Delete room |
| Actual result | Room not deleted |
| Conclusion | **No,** not found |

**Unit testing**

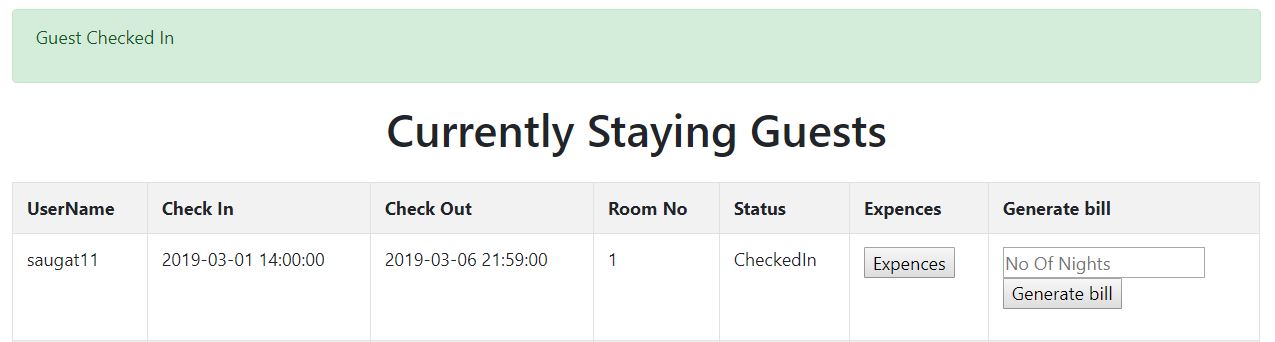
****

****

|  |  |
| --- | --- |
| **Test case no.** | **7** |
| Purpose of the case | Checking if the booked status changes to checked in |
| Test data | 1 |
| Class name | UserTest |
| Function name | testCheckIn |
| Expected result | Status changed to checkedIn |
| Actual result | Status changed to checkedIn |
| Conclusion | **Yes,** expected result matched actual result |

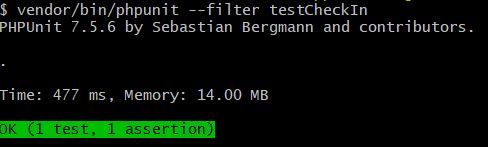
**Black Box**

****

****

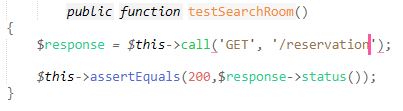
**Unit testing**

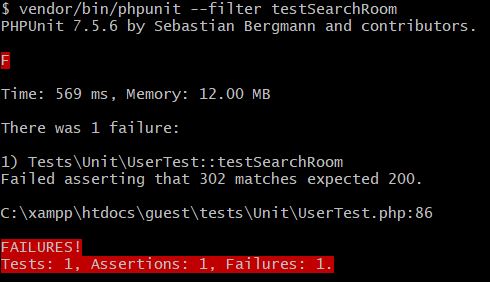
****

****

|  |  |
| --- | --- |
| **Test case no.** | **8** |
| Purpose of the case | Opening search room form |
| Test data |  |
| Class name | UserTest |
| Function name | testSearchRoom |
| Expected result | Open form |
| Actual result | Form not opened |
| Conclusion | **No,** requires user login to open the form |

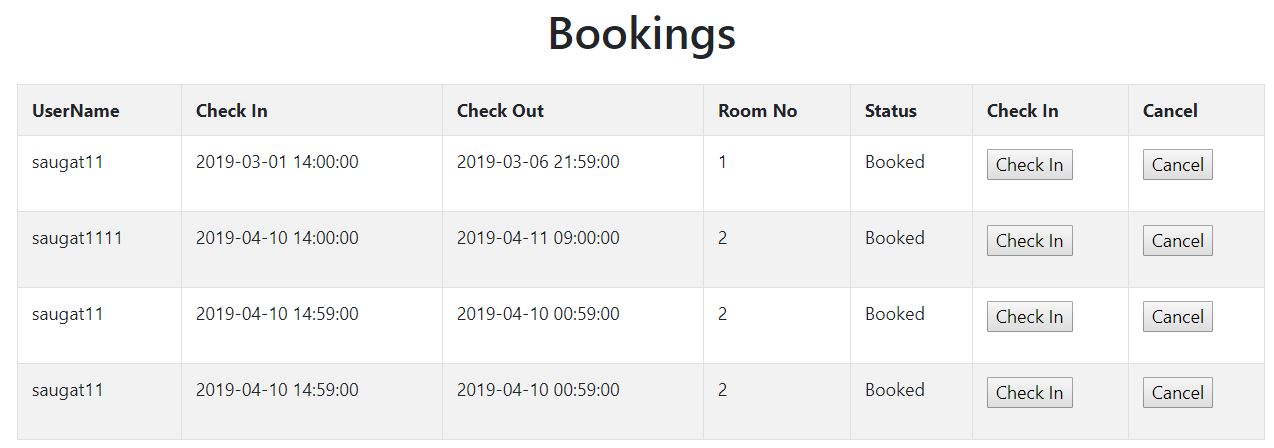
**Unit testing**

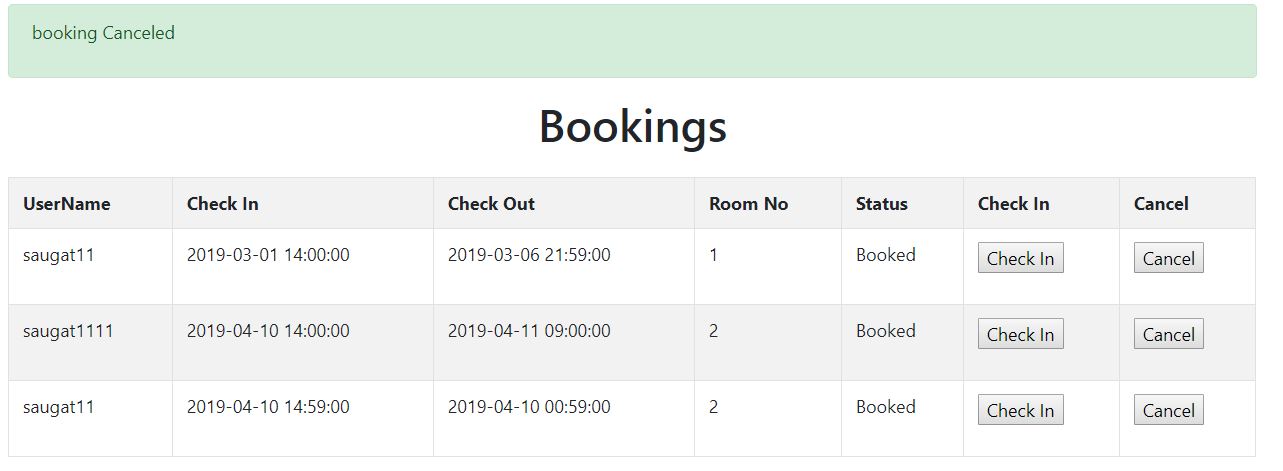
****

****

|  |  |
| --- | --- |
| **Test case no.** | **9** |
| Purpose of the case | Checking if admin can cancel a booking |
| Test data | bookingId= |
| Class name | UserTest |
| Function name | testViewOrders |
| Expected result | View orders |
| Actual result | View orders |
| Conclusion | **Yes,** expected result matched actual result |

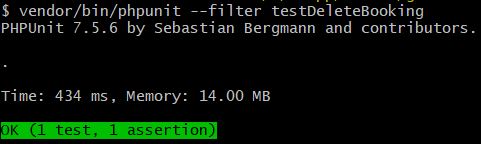
**Black Box**

****

****

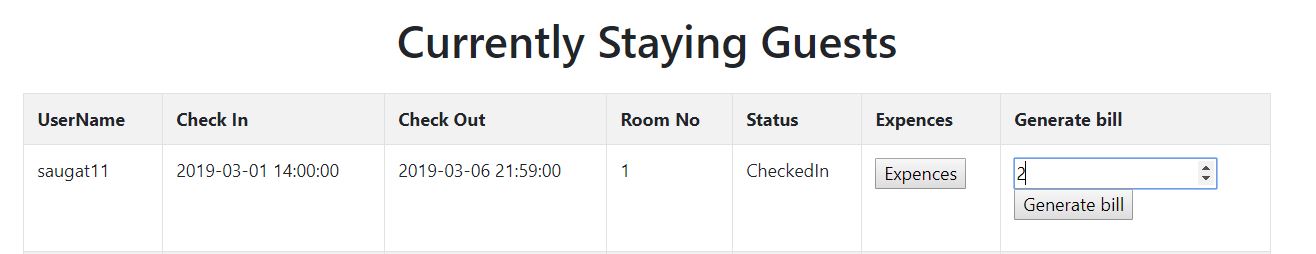
**Unit testing**

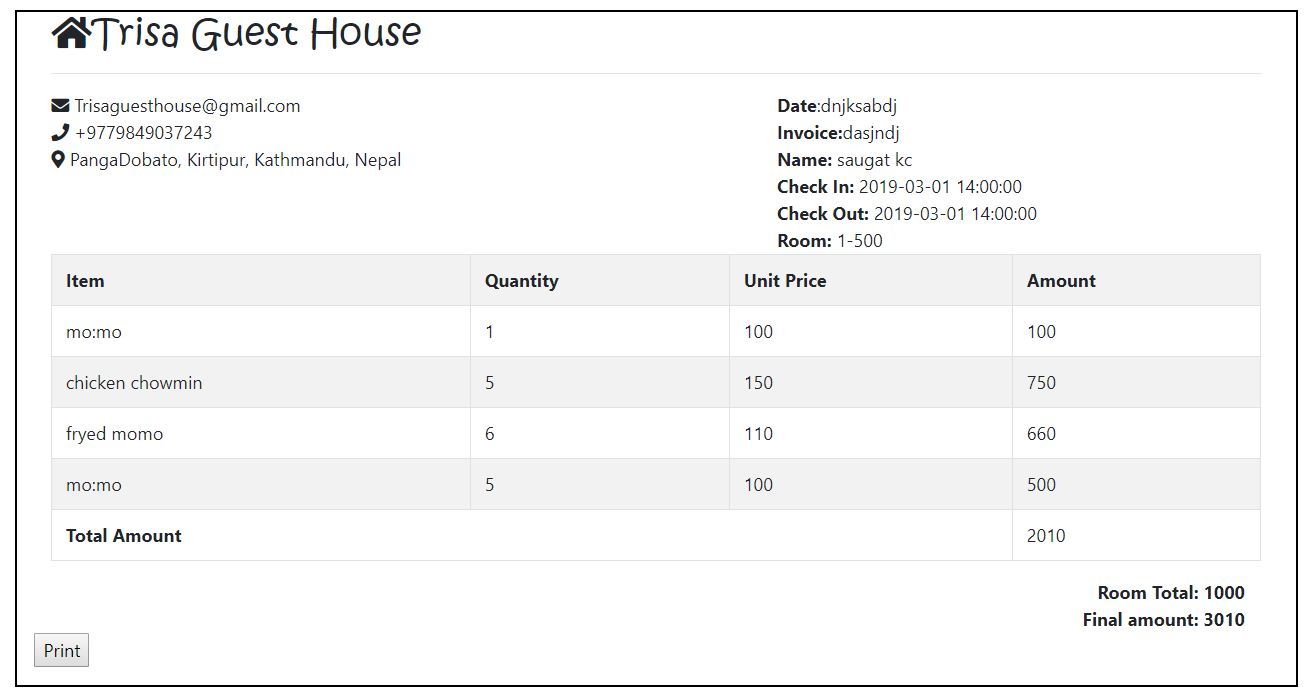
****

****

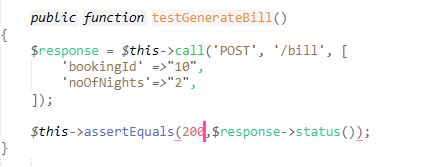
|  |  |
| --- | --- |
| **Test case no.** | **10** |
| Purpose of the case | Checking if bill is generated and change status to checked out |
| Test data | 'bookingId' => 10,  'noIfNights'=>2, |
| Class name | UserTest |
| Function name | testGenerateBill |
| Expected result | Generate a bill |
| Actual result | Generate a bill |
| Conclusion | **Yes,** expected result matched actual result |

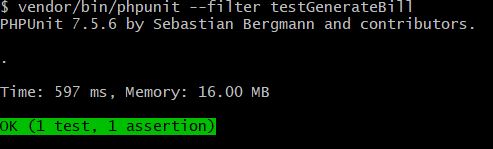
**Black Box**

****

****

**Unit testing**

****

****

# Limitation and Future Work

## Limitation

* Users cannot get access to the database to see background information.
* User do not have access to manipulate their booking details afterwards
* User cannot order items online.
* User cannot see details of other people in the guest house.
* Site cannot be executed in absence of internet.
* Online payment cannot be done.
* Rating and sorting is not available.

## Future Work

* Online Payment System
* Count visit of user and provide free stay after certain
* Add packages
* Booking cancelation
* Online order

# Risk and Configuration Management

## Risk Management

To identify and avoid the possible risk that may occur during the development of our project is risk management. It helps us to tackle the problem in real time when project should be implemented.

***Impact = Likelihood \* Consequence***

Risk Likelihood values are shown as follows

|  |  |
| --- | --- |
| **Likelihood** | **Value** |
| Low | 1 |
| Medium | 2 |
| High | 3 |

Risk Consequence values are shown below

|  |  |
| --- | --- |
| **Consequence** | **Value** |
| Very low | 1 |
| Low | 2 |
| Medium | 3 |
| High | 4 |
| Very High | 5 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S. No** | **Risks** | **Likelihood** | **Consequences** | **Impact** | **Solution** |
| 1 | Server failure | 1 | 5 | 5 | Suitable environment and equipment for server |
| 2 | DDoS attack | 2 | 2 | 4 | Use of proper firewall |
| 3 | Phishing | 2 | 3 | 6 | Proper planning is to be done in every phase. |
| 4 | Identity theft | 3 | 3 | 9 | Proper study of customer |
| 5 | Information leakage | 2 | 4 | 8 | Proper security |
| 6 | Hard disk crash | 2 | 5 | 10 | Data must be backed up in external drive or in clouds. |
| 7 | User error | 2 | 2 | 4 | User training |

# 

## Configuration Management

The process of keeping and tracking the detail data, so the updates can flow with the existing project. Files and folder should be safely stored in a systematic order, so that it can be easily excessed whenever we need to. When configuration management is done, the consistency of the projects becomes better. The given figure shows the configuration management of my project:

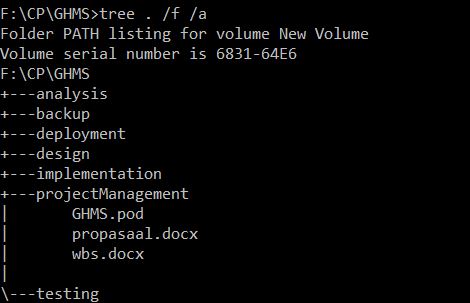


Figure 15.Directory structure

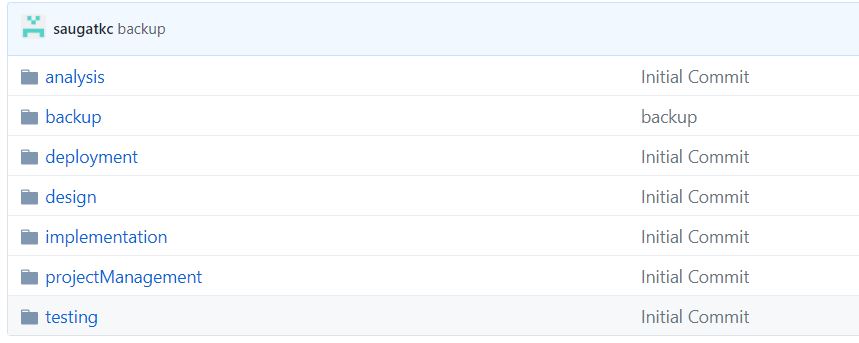


Figure 16.Configuration folder

# Conclusion

The project is developed to make easier for the user to get information of the guest house and their facilities as well as customer for checking the information and to book the rooms online. The application will be reliable, efficient and easier for a user to work on it. The application will be user friendly for the user. Hence, this proposal will be used as the guideline for the final project.

# References

https://www.toolsqa.com/software-testing/waterfall-model

[Accessed on January 3, 2019]

https://searchitoperations.techtarget.com/definition/configuration-management-CM

[Accessed on January 5, 2018]