# Chapter-2 Analysis

## 2.1 Requirements

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

**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

**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.

Requirements

|  |  |  |
| --- | --- | --- |
| **Functional/non-functional** | **Requirements** | **MoSCoW** |
| F | Signup | M |
| F | Admin Login | M |
| F | Customer login | M |
| F | CURD room | M |
| F | CURD package | M |
| F | View available rooms | S |
| F | View package | S |
| F | Booking | M |
| F | View location | C |
| F | Online order | C |
| F | Create malicious bill | M |
| F | Create final bill | M |
| NF | Security | M |
| F | Rate rooms | C |
| NF | User friendly | M |
| F | Photo gallery | S |
| NF | User feedback | C |
| NF | Customer privacy | M |
| NF | Data integrity | S |
| NF | Maintainability | s |

## 2.2 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)

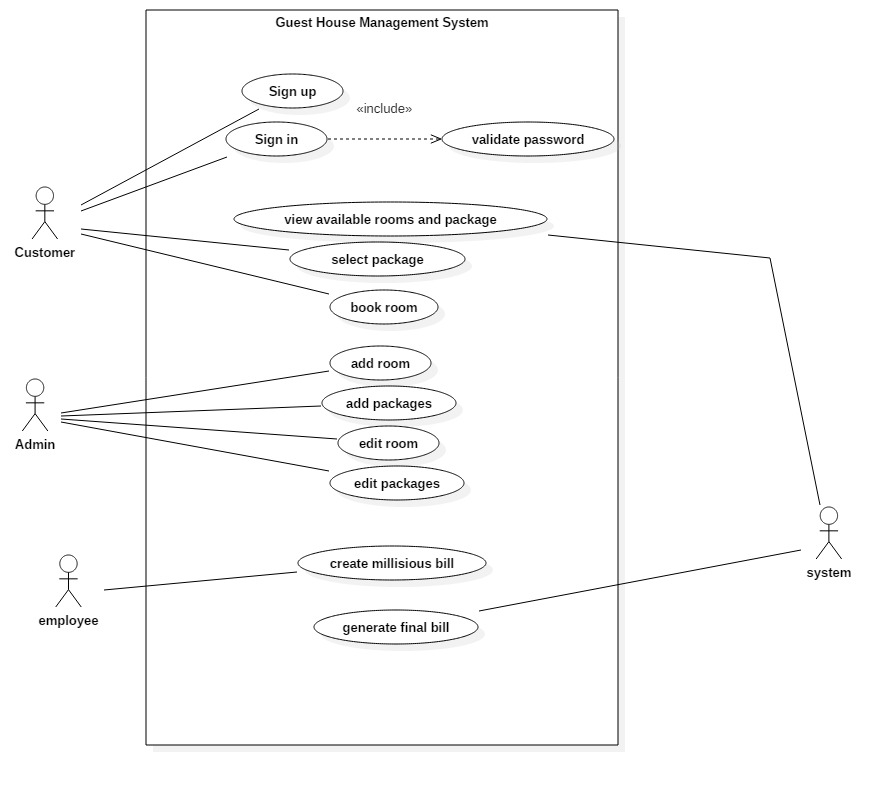


Figure 1 use case diagram

**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.

# 2.3 Initial class diagram

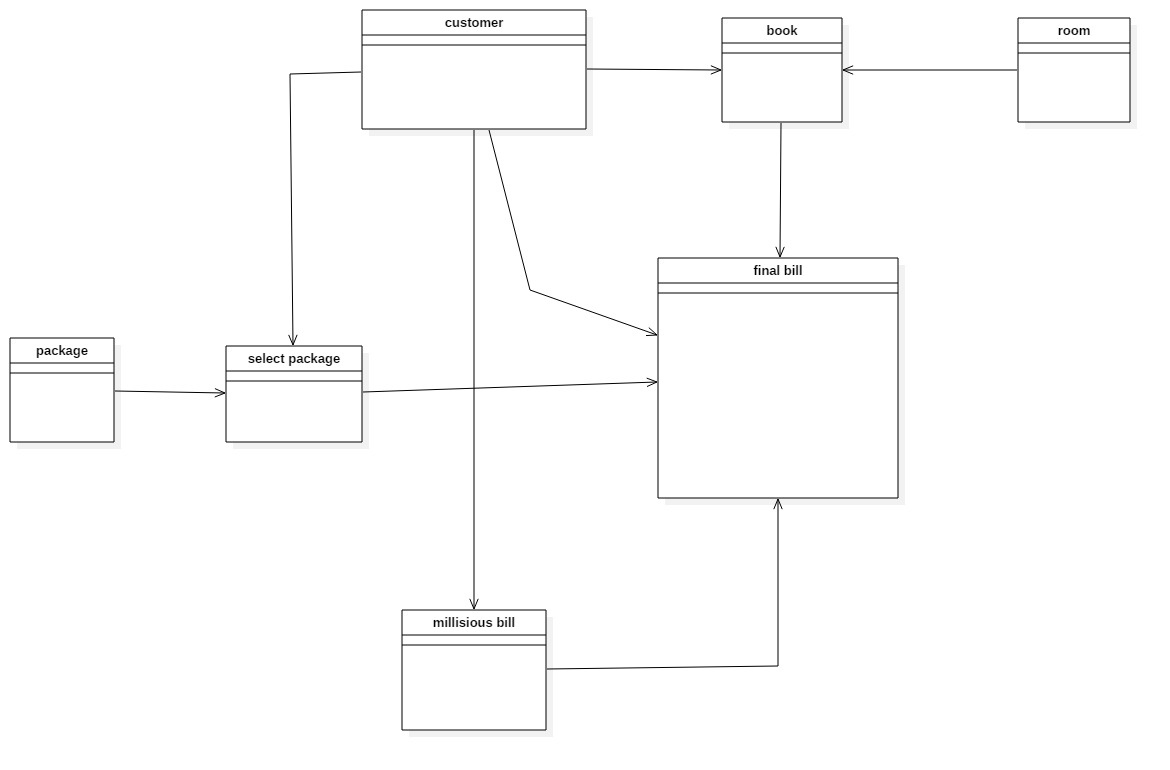
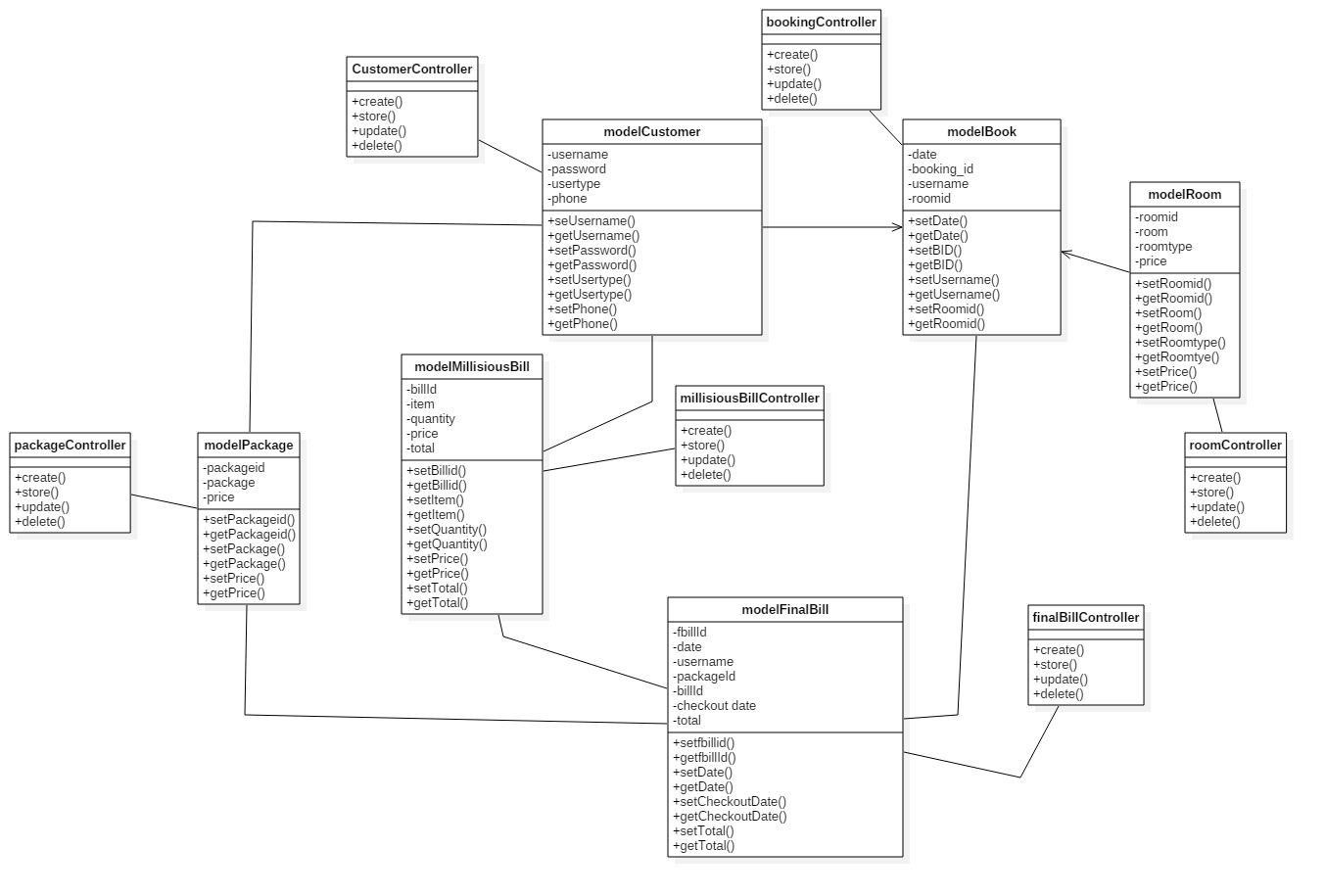


Figure 2 initial class diagram

# Chapter 3 – 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)

****

**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.

## 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 )

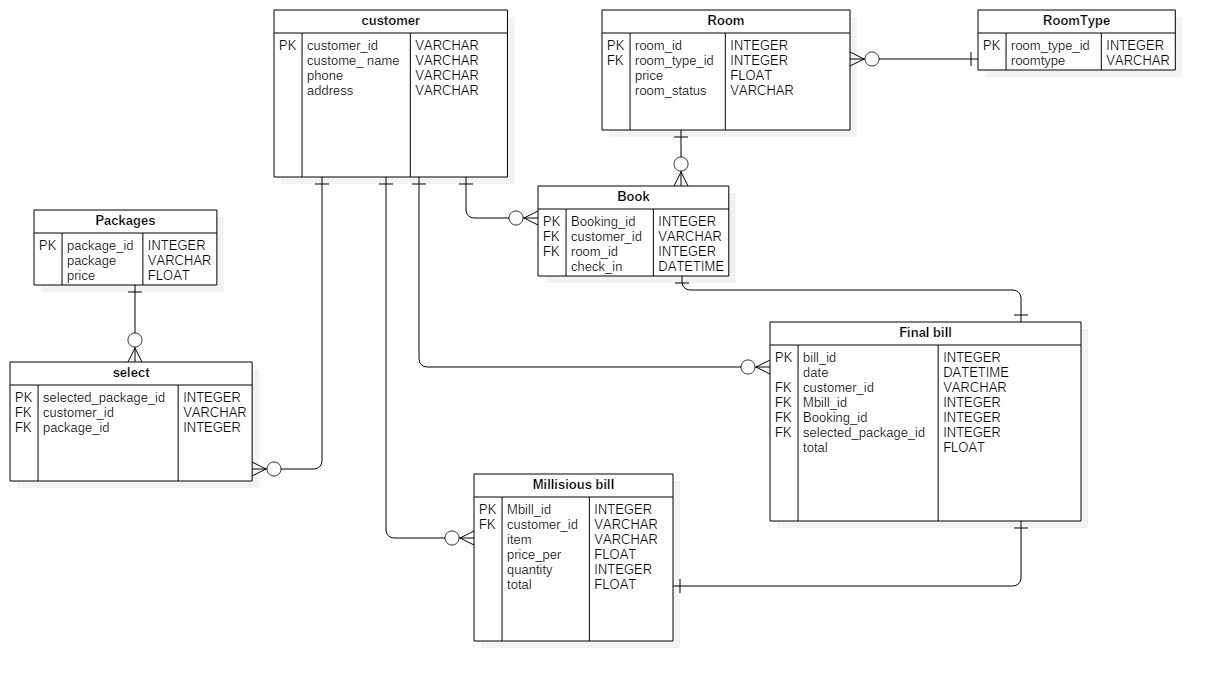


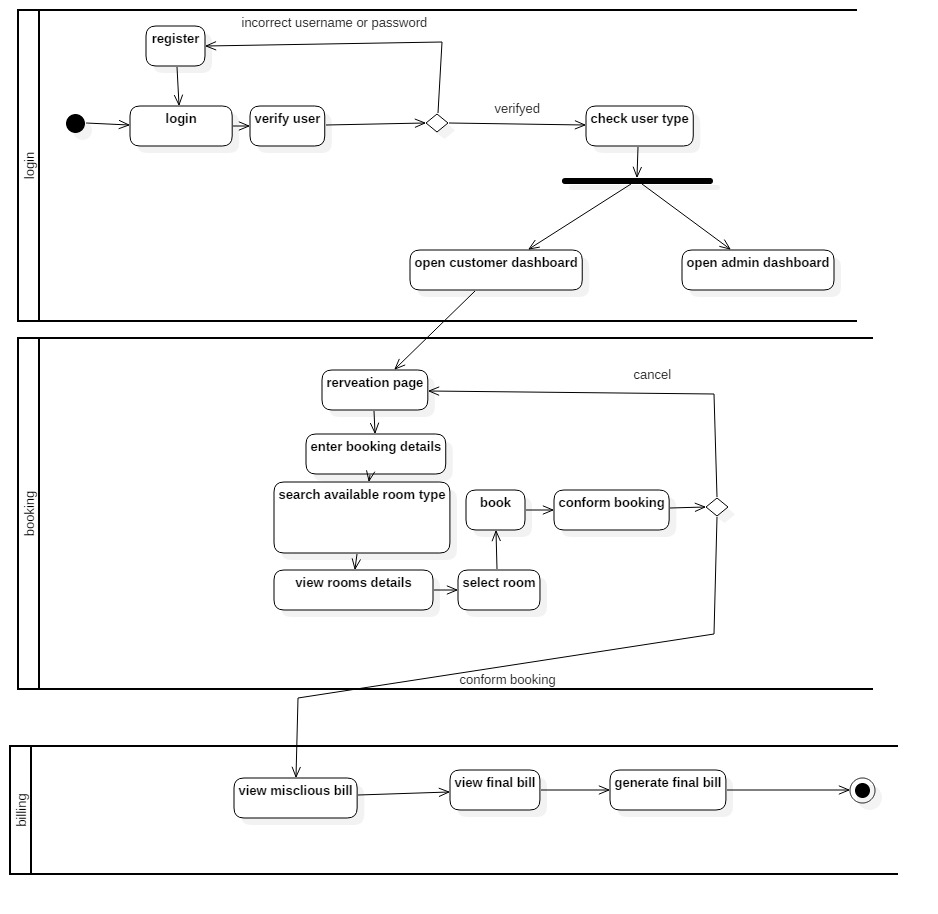
Figure 3 Er-diagram

**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.

## 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.

## UI design

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

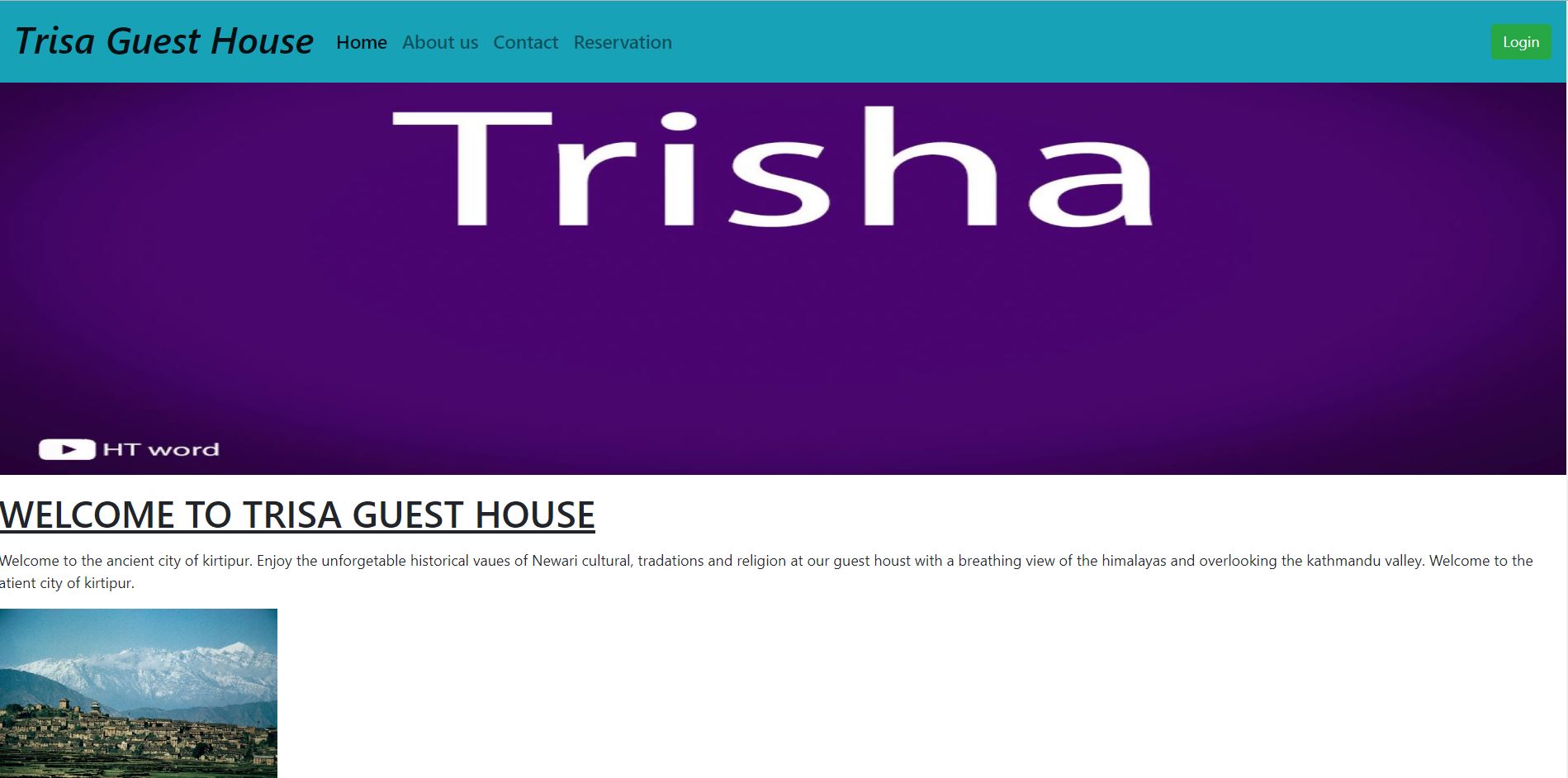


Figure 4: home

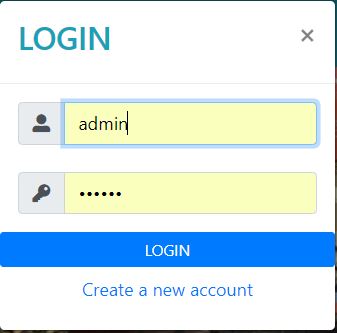


Figure 5 login

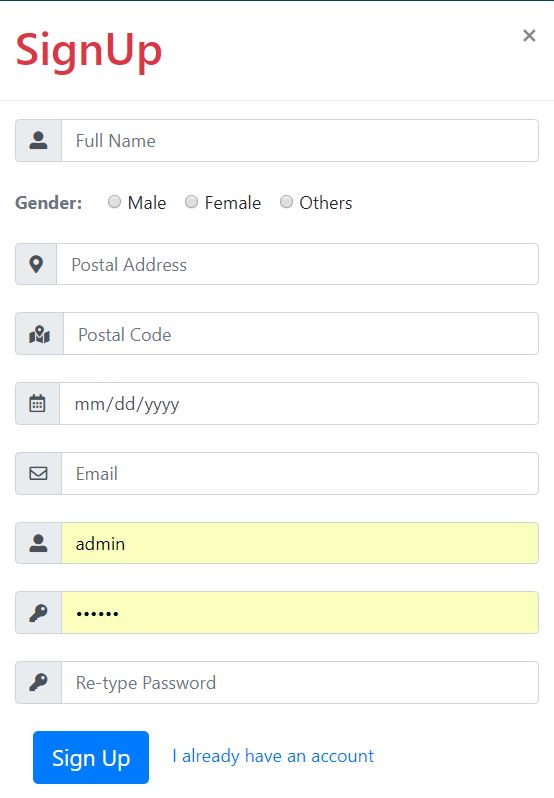


Figure 6: signup

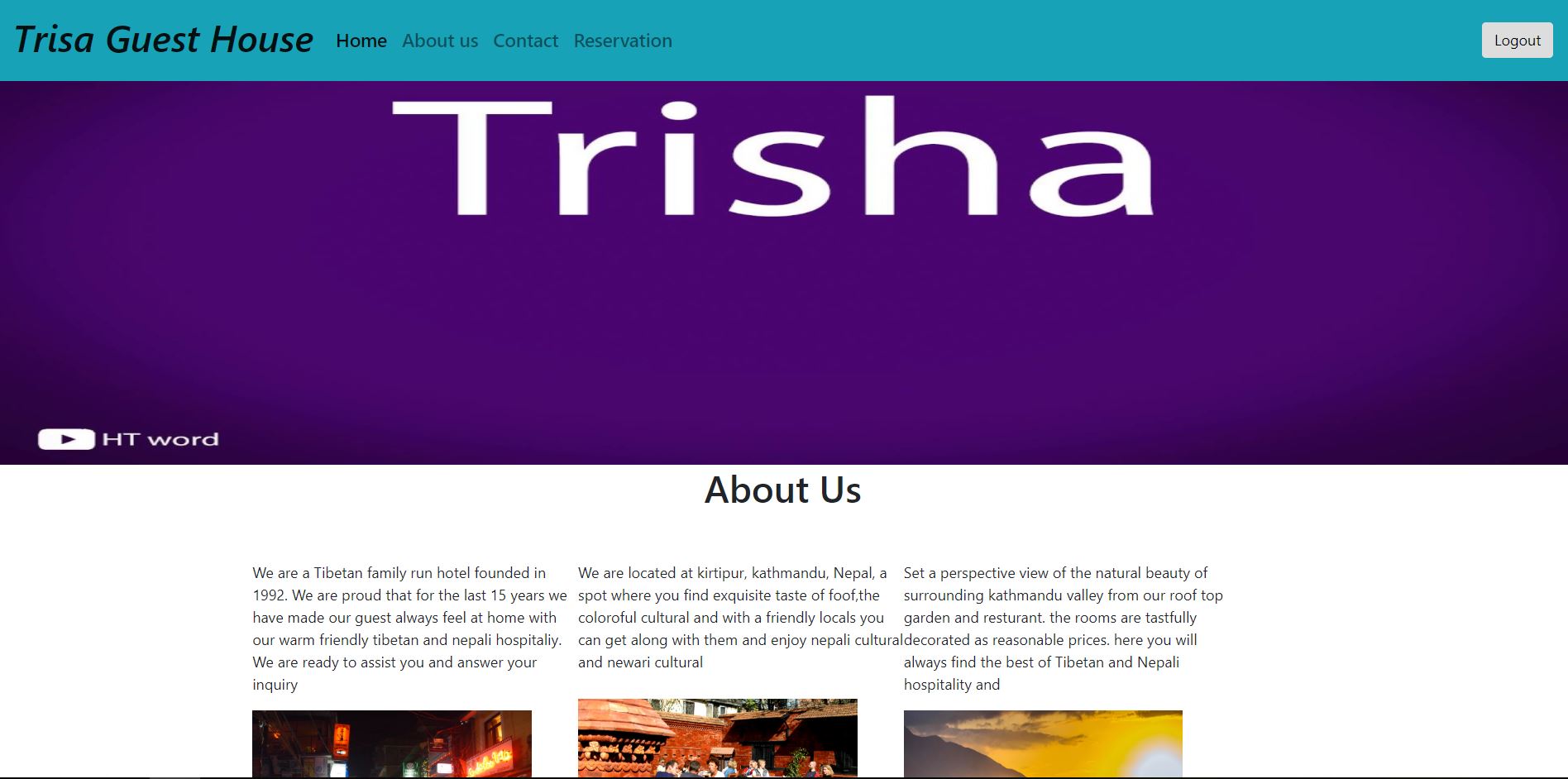


Figure 7 about us

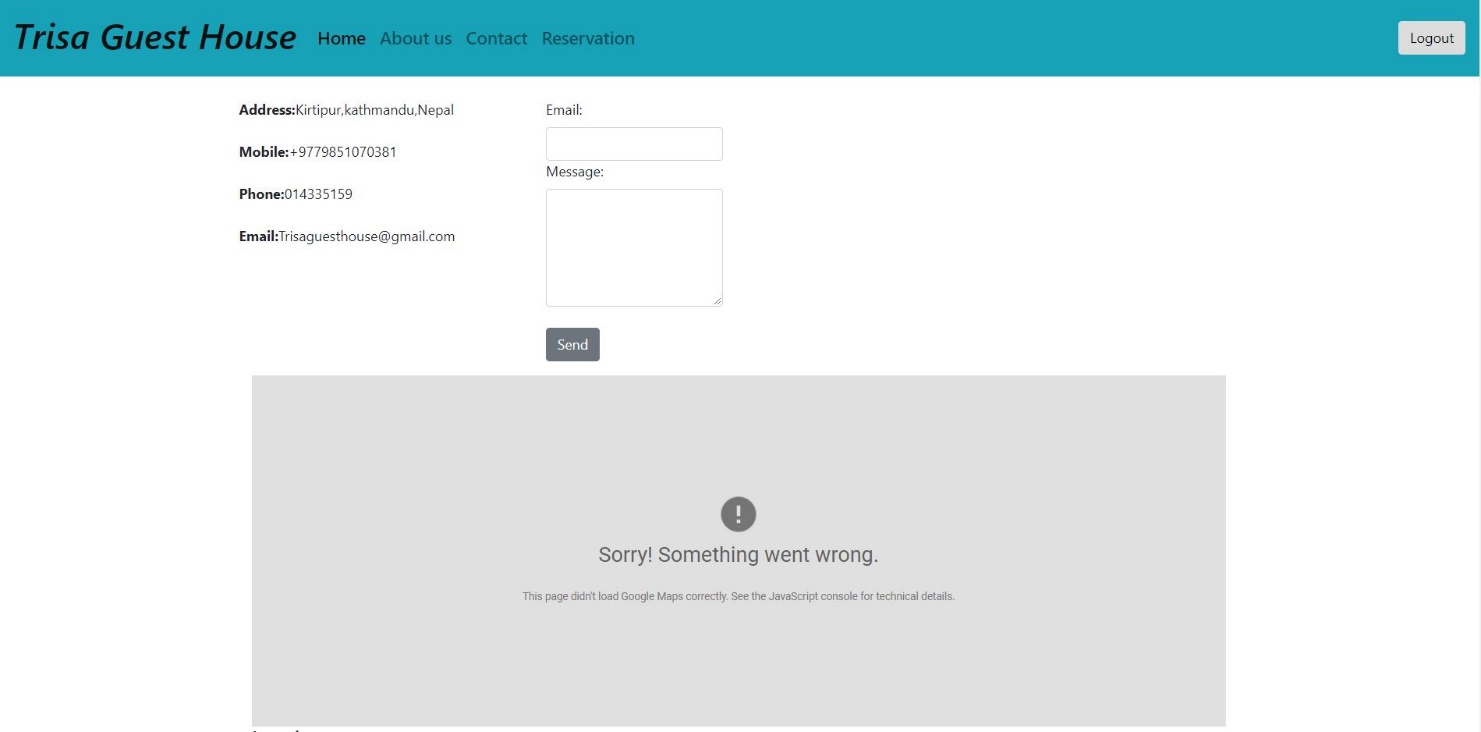


Figure 8 contact us

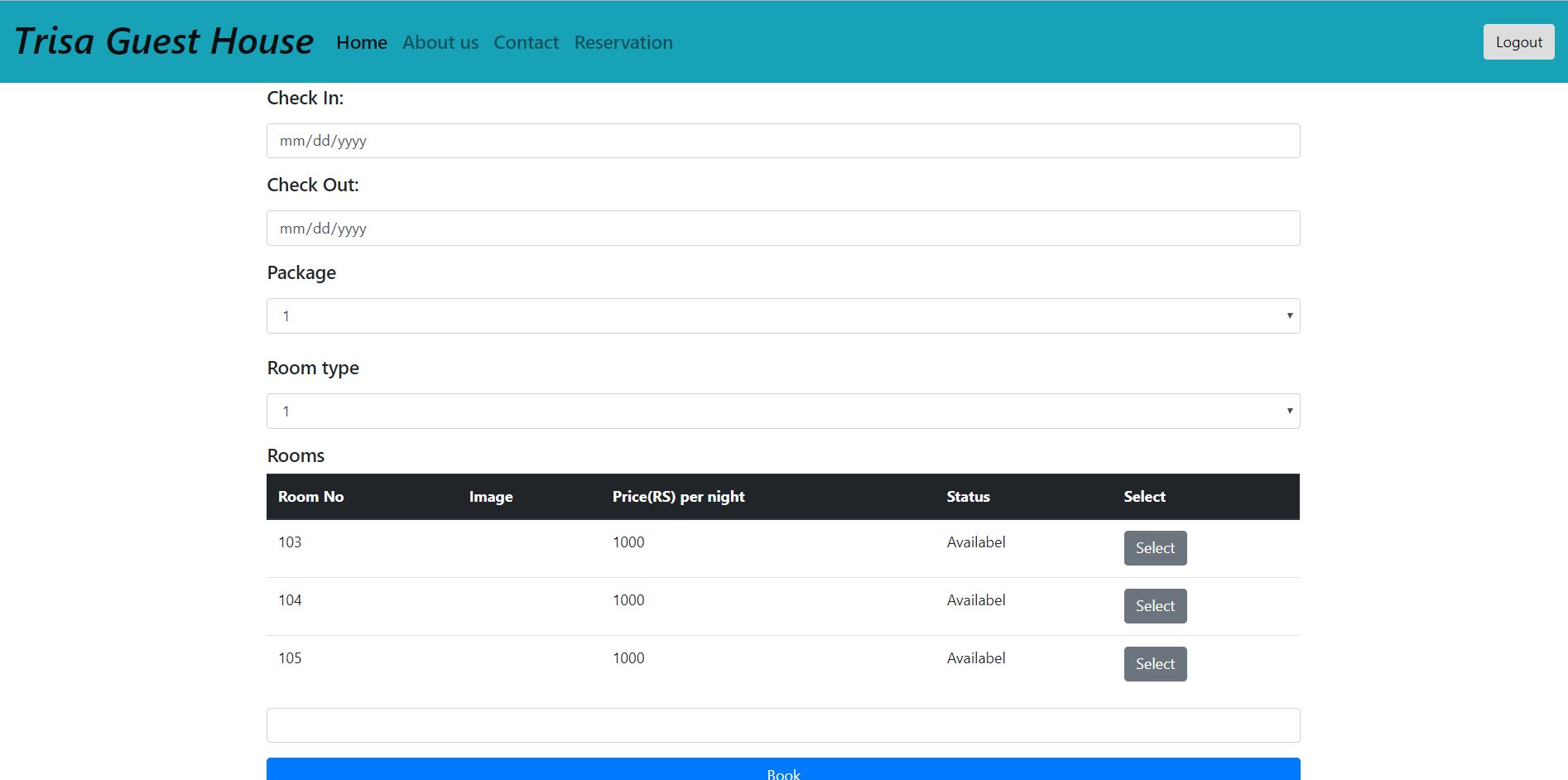


Figure 9 reservation

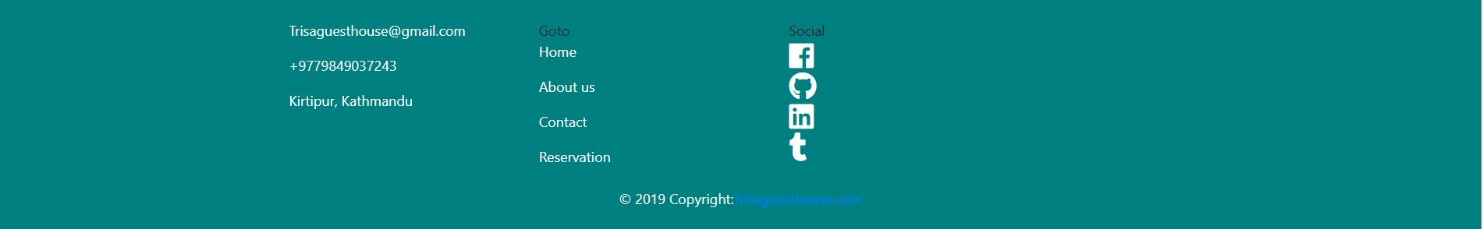


Figure 10 footer