**Hotel name**

Aniket Surendrakumar Dandekar

R.K. Talreja College

T.Y. BSc Computer Science

# **Contents**

---

# Chapter 1: Overview of project

## 1.1 Introduction

"Online Hotel Booking System" is a web-application that deals with Room booking/reservation in Hotel. Room booking and room management in hotel can be difficult to manage. This system automates day by day activity in a Hotel from managing available rooms to users booking a room in hotel. The system can be managed by manager or staff of the Hotel to manage guests/users and room availability in hotel. The services that are provided in the system are booking room/ reception hall through online booking. Managing availability and details of rooms and managing employees' details and customer details through admin panel. The guests are provided with detailed info of all rooms available in the hotel including the reception hall.

Main objective of this system is to provide easy room booking for the guests or the customers of the hotel with help of login. The guests is registered and then logged in. Once guest is logged in they are taken to booking page where guest can specify all the details of their stay and book a room.

## 1.2 Purpose of the Project

Online Hotel Booking System is system for managing Hotel Business. After successful login the customer can access the booking page. The main objective of developing this system is to provide guests or customers simple hassle-free booking system at the end of their fingertips with providing a simpler booking experience overall. Also to help hotel managers/administrators manage the whole process of booking and managing the hotel online. In the proposed system not only administrator's task is simplified but also user experience is simplified.

This system was developed to counter the problems like complicated booking, only in-hotel booking and no proper customer support, which are common problems that occur when guest is trying to book a room for stay at a hotel. This system keeps track of all the registered customers and their details. Also the rooms present in hotel are kept track of and if they are available for booking or not. This system also provides option to register staff details and manage them through admin panel.

The proposed system, through benefits of the online resources and solutions directly avoids the common problem of that occur Hotel industry, such as difficulty maintaining information of rooms, their prices, guests, staff etc. Since the system is online, it also requires

little no paper and register books commonly used by hotels to maintain all this data, making it good for the environment.

## 1.3 Scope of the project

The system proposed is an online system and is valuable for both hotel and its customers because it is simplifying the hotel booking process. The customer can register in this website with the help of a valid email address and log-in to the website. The website is up-to-date with all types of rooms available in the hotel for stay of the guests with their prices. Guests can look up availability of rooms from the website.

New customers while browsing the site can see all types of rooms available in the hotel inside the rooms section of the website. This page can be managed by hotel staff like changing/adding room photos, changing/adding description and prices of those rooms.

The proposed system provides guests with every detail about the room possible, so then the guest can select their requirements and costings are displayed on same page as the customer selects a type room they want to book and then book a room. Users are also provided with message box at booking page to mention additional details for their stay at the hotel. Here they can mention if they want to make special provisions for their stay, like decoration inside rooms, multiple room bookings, reception hall booking, specific menu details for their meals, etc.

All the details about rooms is managed by the staff/administrator. From is a particular room is available for use or not. What the prices of stay per night should cost for different types of rooms. Hotel inventory can also be managed via this system. Regular customers can be identified by the staff and giving special discounts to those customers. Staff details can be managed by administrator.

## 1.4 Existing system

The current system in the hotel industry is paper based, where papers are used to create sizeable register books. Not only using paper is not good for the environment but also it takes lot of physical storage space to store all these register books therefore costing more money. Also using paper for keeping records of all the hotel data and information means the data is vulnerable. The ink on the paper can be damaged fade away with time, paper itself can be damaged or destroyed with regards to various entities like fire or water, and hence the data will be lost forever and a loss of money too.

As the current Hotel system is offline, user has to go to hotel physically and check if the hotel has a room available for stay or not. This is both time consuming and lot of manual

work for the guest. This makes the whole experience bitter for the guest when it should have been a more relaxing or comforting experience.

If a customer contacts hotel for asking information about rooms or hotel, a member of the staff has to be available to answer that call. Which means more workforce is needed to manage the hotels business.

## 1.5 Proposed system

This system is aimed at helping both the customers in booking hotel room for stay and the administrators of the hotel in managing the hotel because current offline system has some loopholes and some basic problems discussed in previous point. The proposed system being online thus prevents those problems before even occurring. The online hotel room booking system site helps the users to check available rooms, what kind of services are provided and inside those hotel rooms and their prices without actually going or even needing to contact the hotel to provide the particular details, which wastes time of both customer and the staff answering the requested information.

The administrator can view can manage staff details. Also administrator can view user's login data and can block a particular user from logging in if he has unpleasant history with the hotel.

### Customer Registration Module

Customer registration module handles the user's information used to create login details for user. It first takes the data user entered and then checks if user is new or old and if user is a new user then the entered data is stored in the database. After this user can login with email address and password.

### Customer Login Module

Customer login module handles login information of guests. This module takes input from the login page and then checks with the database if the email address exists in database or not, if it does then it checks and verifies password and it credentials are accurate then user is logged in and can book room. This module also handle admin login for admin page.

### Forgot Password Module

Forgot password module is used when the customer has forgotten their password and needs to login again for room booking. This module when used sends password to user on the registered email address if they are already a logged-in member.

**Room Booking Module**

This module is used at the time of booking and is only used if logging in of user is done prior. The module takes various details as input for room booking like customer's full name, their contact details, the room they want to stay in, if they want to book multiple rooms, check-in and check-out date and the prices of rooms they have selected are shown to them in same page. All this data is then checked if similar data already exists in database or not to prevent multiple entries of same data and after this the data is stored in database and booking is done.

**Contact Module**

Contact module is used to for user contacting the hotel for information regarding their stay. This module is can be used to send booking information directly to hotel if user doesn't have email address for login. The data is sent in database and then staff can respond to the request and contact the user with contact number that user provided.

**Room Details Module**

This provides user browsing the site with useful information of rooms and their prices accordingly.

## 1.6 Importance of Hotel Booking System

Hotel booking and management is key to any hotels success. It takes a significant amount of time and resources to get the things just right. The booking system being offline significantly hinders the performance of hotel and is not very cost efficient of time efficient. It becomes very important for hotel to adopt a system which is cost effective, time efficient, simpler for users to understand and overall efficient.

Hotel industry thrives on the type of services they provide to their guests from booking to every little thing possible to make the user experience pleasant, hassle free and as simple as possible. Starting point of this pleasant experience starts from checking for hotel, here online website application of hotel comes into play. It provides all the information needs to know about the hotel at their fingertips.

Then comes the booking part, now this also needs to be simpler and online booking system is helping in the same. With little to no effort user can check if room is available or not and then book the room. The system being online also helps in attracting more new users than traditional system.

## 1.7 Benefits of Hotel Booking System

What are the benefits that are provided by the online hotel booking system? These are some of the benefits provided as follows.

**Keeping track of revenue**

The system can keep track of all the bookings and administrators can used that information to keep track of revenue.

**Improved efficiency**

Since the booking system is online less time and efforts are required to book and manage booked rooms, the efficiency is thus improved.

**Access to database remotely**

The administrators of hotel can view all data at any time from any location and manage that data from anywhere.

**Better customer service**

From making hotel room details available online to making booking very simple it all counts towards a better customer experience overall. And allowing custom bookings to users also provides better customer service.

**Cost saving**

Better and efficient management and time efficiency means better cost efficiency.

# Chapter 2: Project requirements and analysis

## 2.1 Project background

The hotel booking system is a computerized online system for managing the hotel booking and to some extent manage it also. The main theme of developing this system is to help hotel administrators to manage the hotel and help customers for online room booking. This project was developed to solve problems of offline hotel systems such as booking rooms beforehand and checking available hotel rooms. The previous system being offline required a lot a paperwork thus not being very efficient system overall from cost or time perspective. So, the Online Hotel Booking System will develop to help the Hotel Administrator to manage Hotel room bookings and guests of hotels. Also the customers to make their hotel experience much smoother and simpler. Consequently, this project will upgrade the manual system and make the business fast and systematic to manage business activities.

## 2.2 User requirements

To deliver the best services and experience to user we tried to find out what are the most common requirements that users of this system have, they are as follows:

**Administrator's aspect:**

- Monitoring the whole system from a panel
- Taking backup of the database
- Viewing all the records and modifying them
- Updating room costings time-to-time

**Customer's aspect**

- Seeing room details and availability at ease
- Signing in and signing out
- Booking a room for stay
- Forgot password help

## 2.3 User of the system

Every system have different type of users, users of this system are as follows

**Admin:** admin can has access to all data present in the system and can manage the data.

**Customer:** Customer is the most important user of the Hotel Booking System. Customer can book a room in hotel booking system after signing in.

## 2.4 Requirement analysis

The Hotel Booking System is a web-application which needs to be hosted on a server and can be accessed by any device which has web browser. The objective of hotel booking system is to change the way hotels worked by implementing online solution. To develop this System Some Software and Hardware Requirements are necessary, Requirements which are necessary are given below briefly.

<u>**Hardware Requirements**</u>:

    **Processor**:   At least 2.0 GHZ

    **RAM**:      At least 2 to 4GB

<u>**Software Requirements:**</u>

    **Compatible operating system**:  Windows 7 and higher, Mac

    **Front end design tool**:  HTML5, CSS3, JavaScript, Ajax, json, Bootstrap .

    **Back End**:  MySQL, jquery

    **Editor Tools**: Notepad++, Visual studio.

    **Other Graphics Tools**:  Adobe Photoshop. Adobe Illustrator

    **Web Browser**:  Google Chrome, Firefox, or any compatible update browser.

    **Web server software**:  Apache Tomcat, Wamp, Xampp.

    **Server side scripting too**l:  PHP-5.6.

    **Database tools**: MYSQL DBMS.

    .

## 2.5 Project Development Languages

There are two types of languages used in the "Hotel Booking System". One being Programming Language and the other is Database Language. In Front End Development JavaScript, HTML, CSS has been used & Back End Development MySQL Database Language and PHP for server-side scripting language is used in "Hotel Booking System"

**Programming Language & Mark-up Language**: HTML, PHP, JavaScript and Ajax

**Styling language:** CSS

**CSS Framework**: Bootstrap

**Library** jQuery

**Database Language**: MySQL

## 2.6 Why Developer used the above Languages to develop

**JavaScript:**

This language helps to makes Website truly feel alive. Developer has used JavaScript at Dashboard Website for "Slideshow, searching, line chart, delete data, view data, and various sections in website, validations etc."

**PHP**:

This language helps develop server side functions in websites and they are not shown to users accessing the site. Developer has used PHP in different section for creating function such as "Database Connection", "Admin User Login Validation", "Data Insert, Update, Retrieve and Delete".

**HTML**:

This is mark-up script Language for creating web pages. Developer has used it different section in my "Restaurant Management System" Such as for making "Header & Footer" in home page and dashboard,

**CSS**:

Cascading Style Sheet is a Style sheet language that is mostly used to design websites.

Developer has used this style sheet language in different sections at my project. Such as

Admin Panel, Home Page, Dashboard, Sidebar Navigation design.

## 2.7 Software Development Lifecycle
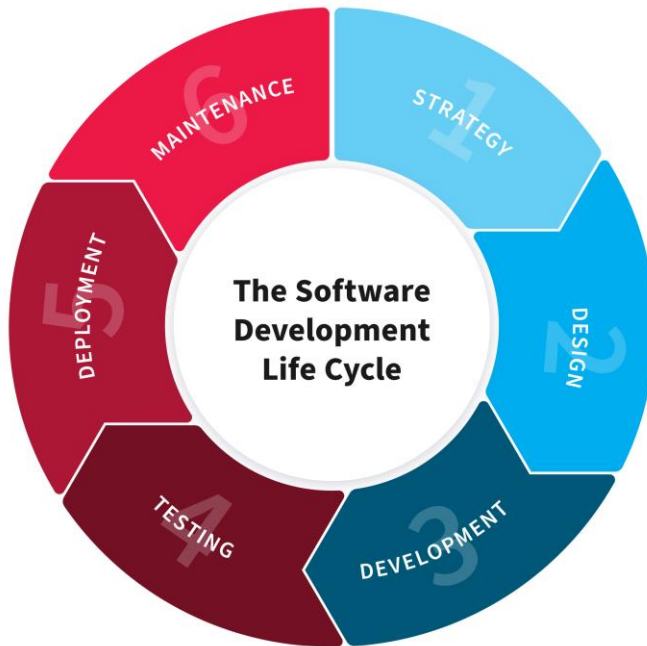
The software industry includes many different processes, for example, analysis, development, maintenance and publication of software. This industry also includes software services, such as training, documentation, and consulting. Our focus here is about software development life cycle (SDLC). So, due to that different types of projects have different requirements. Therefore, it may be required to choose the SDLC phases according to the specific needs of the project. These different requirements and needs give us various software development approaches to choose from during software implementation.

## 2.7.1 SDLC Models

There are various software development life cycle models defined and designed which are followed during the software development process. These models are also referred to as "Software Development Process Models". Each process model follows a Series of steps unique to its type, in order to ensure success in the process of software development. Following are the most important and popular SDLC models followed in the industry:

1. Waterfall model
2. V-Shaped model
3. Prototyping model
4. Spiral model
5. Iterative and Incremental model
6. Agile model
7. RAD model.

## 2.7.2 Waterfall Model

The Waterfall Model was the first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. Waterfall model is the earliest SDLC approach that was used for software development .The waterfall Model illustrates the software development process in a linear sequential flow; hence it is also referred to as a linear-sequential life cycle model. This means that any phase in the development process begins only if the previous phase is complete. In waterfall model phases do not overlap.

## 2.7.3 Waterfall Model Design

The Waterfall approach was the first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In the Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. Following is a diagrammatic representation of different phases of waterfall model
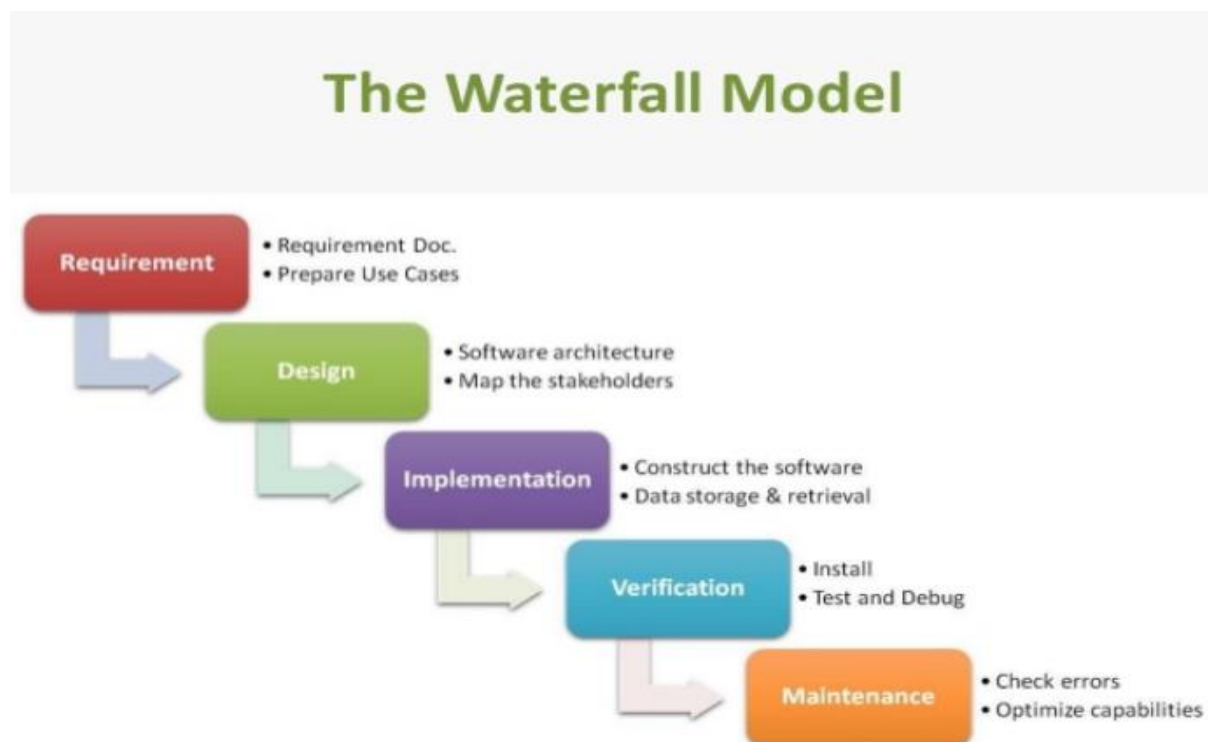
## 2.7.4 Waterfall Model Phases

The sequential phases in Waterfall model are:

1. **Requirement Gathering and analysi**s: All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.

2. **System Design:** The requirement specifications from the first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.

3. **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.

4. **Integration and Testing:** All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

5. **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.

6. **Maintenance**: There are some issues which come up in the client environment. To fix those issues patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment. All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for the previous phase and it is signed off, hence the name "Waterfall Model". In this model phases do not overlap

## 2.8 Advantages and Disadvantages of Waterfall Model

## 2.8.1 Advantages of Waterfall Model:

1) Waterfall model is simple to implement and also the amount of resources required for it are minimal.

2) In this model, output is generated after each stage (as seen before), therefore it has high visibility. The client and project manager gets a feel that there is considerable progress. Here it is important to note that in any project psychological factors also play an important role.

3) Project management, both at internal level and client's level, is easy again because of 9 visible outputs after each phase. Deadlines can be set for the completion of each phase and evaluation can be done from time to time, to check if the project is going as per milestones.

4) This methodology is significantly better than the haphazard approach to develop software. It provides a template into which methods of analysis, design, coding, testing and maintenance can be placed.

5) This methodology is preferred in projects where quality is more important as compared to schedule or cost.

### 2.8.2 Disadvantages of Waterfall Model

1) Real projects rarely follow the sequential flow and iterations in this model are handled indirectly. These changes can cause confusion as the project proceeds.

2) It is often difficult to get customer requirements explicitly. Thus specifications can't be freezed. If that case arises, a baseline approach is followed, wherein output of one phase is carried forward to the next phase. For example, even if SRS is not well defined and requirements can't be freezed, still design starts. Now if any changes are made in SRS then formal procedure is followed to put those changes in the baseline document.

3) In this model we freeze software and hardware. But as technology changes at a rapid pace, such freezing is not advisable especially in long-term projects.

4) This method is especially bad in case the client is not IT-literate as getting specifications from such a person is tough.

5) Even a small change in any previous stage can cause big problems for subsequent phases as all phases are dependent on each-other.

6) Going back a phase or two can be a costly affair.


## 2.9 Server Side Requirement

**Hardware Requirements:**

**Processor**: Quad-core or Hexa-core Intel i7/Intel 19/Thread ripper/Xeon/.

**Motherboard**: GPU that is compatible with OpenGL 3.2. (Integrated graphics cards

Intel HD 4000 or above).

**RAM**: 16GB RAM.

**SSD**: 500 MB

**Software Requirement:**

**Operating System**: Windows Server 2016, Ubuntu Server

**Network Requirement:**

**Bandwidth**: 10 MBPS (The number of 100 users)

# Chapter 3: Project requirements and analysis

## 3.1 Class diagram



**customer**

| | |
|---|---|
| id | int(10) |
| name | varchar(50) |
| email | varchar(30) |
| phone | varchar(10) |
| password | varchar(255) |
| dt | datetime |

**Room booking**

| | |
|---|---|
| res_id | int(10) |
| res_name | varchar(50) |
| res_email | varchar(50) |
| res_phone | int(10) |
| res_room | varchar(50) |
| res_no_of_roo m | varchar(60) |
| res_no_guests | varchar(60) |
| res_cidate | date |
| res_codate | date |
| res_message | varchar(255) |
| res_dt | datetime |

**contact**

| | |
|---|---|
| Cont_name | varchar(50) |
| Cont_email | varchar(50) |
| Cont_phone | varchar(10) |
| Cont_password | varchar(255) |
| Cont_dt | date |
| Cony_message | Datetime |

**admin**

| | |
|---|---|
| id | Int(5) |
| F_name | varchar(20) |
| L_name | varchar(20) |
| type | Int(3) |
| email | varchar(30) |
| password | varchar(255) |

**Figure: Class diagram of Hotel Booking System**

## 3.2 Use case diagram
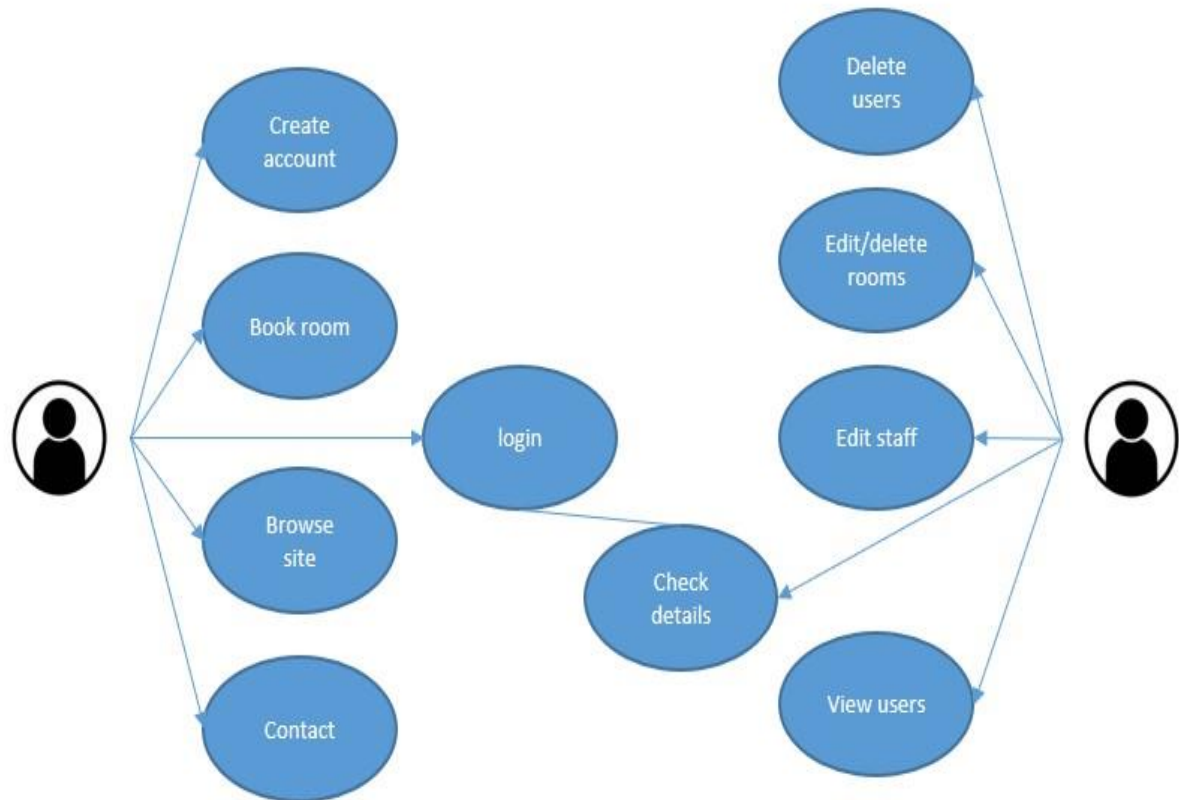


**Figure: Use case diagram of Hotel Booking System**

## 3.3 Activity diagram
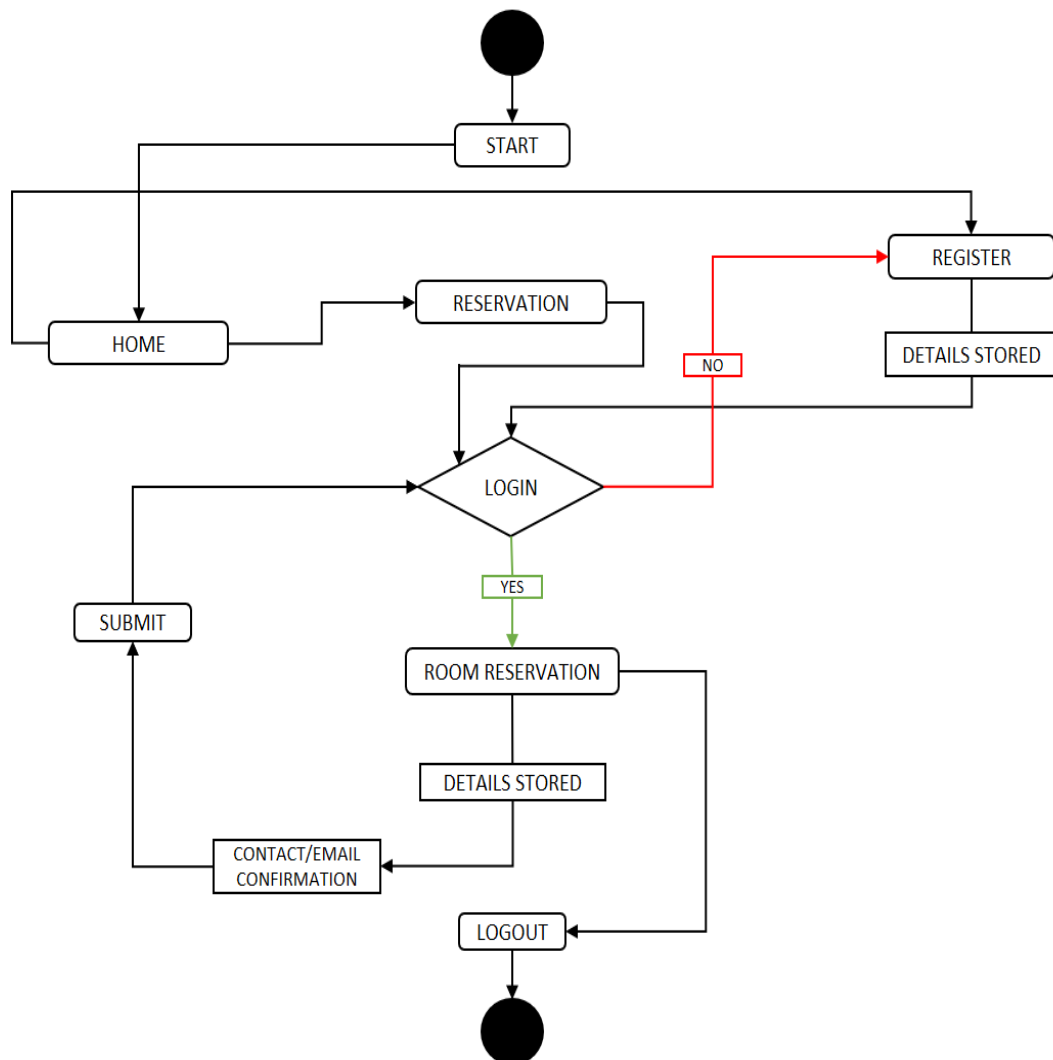


**Figure: Activity diagram of Hotel Booking System**

## 3.4 Login Sequence diagram



| User | login | Validate | Database |
|---|---|---|---|

Login start
Validate(email,password)
Email:found
Password:match

Validate
(password:matched)

Room booking
Login success

Alert message as required
Login denied
Validate
(password:unmatched)
(email:unmatched/exist)
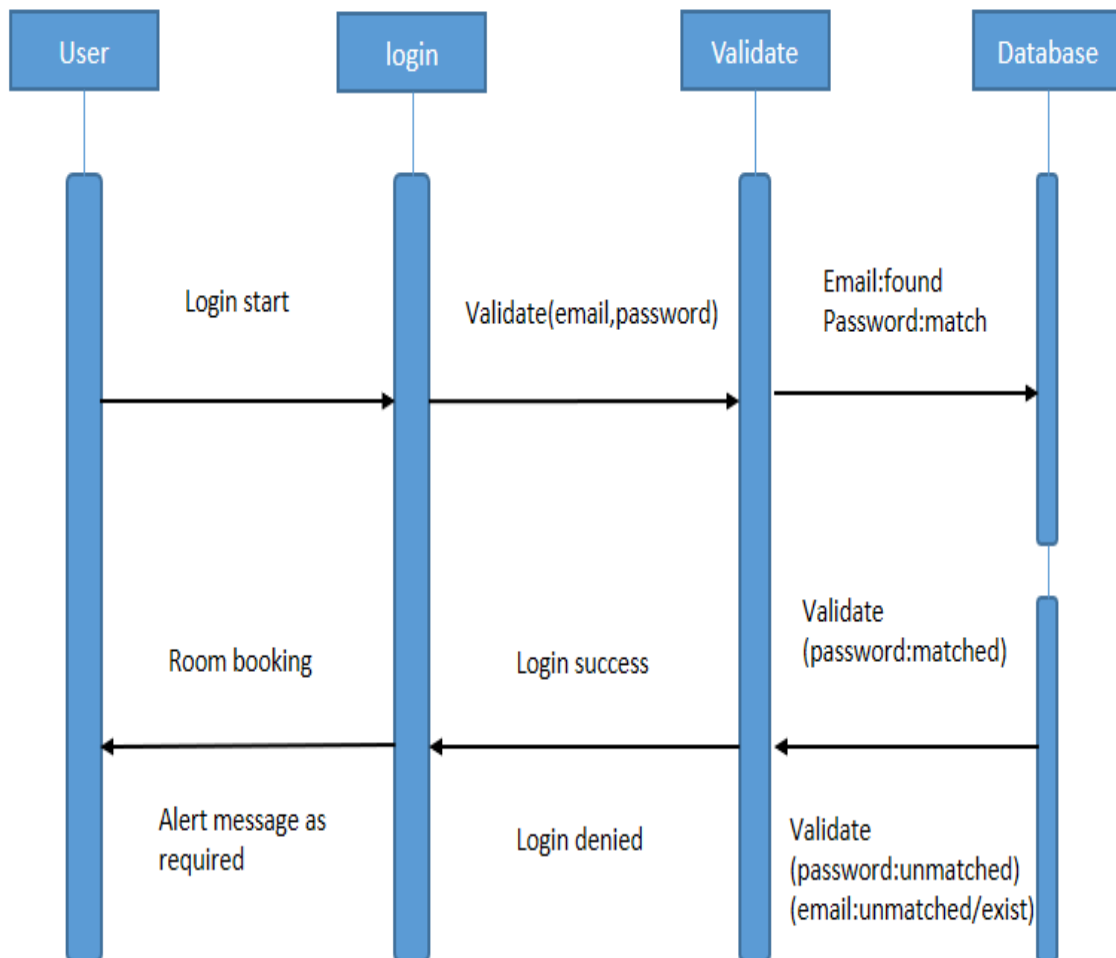
**Figure: Login sequence diagram of Hotel Booking System**

## 3.5 ER diagram
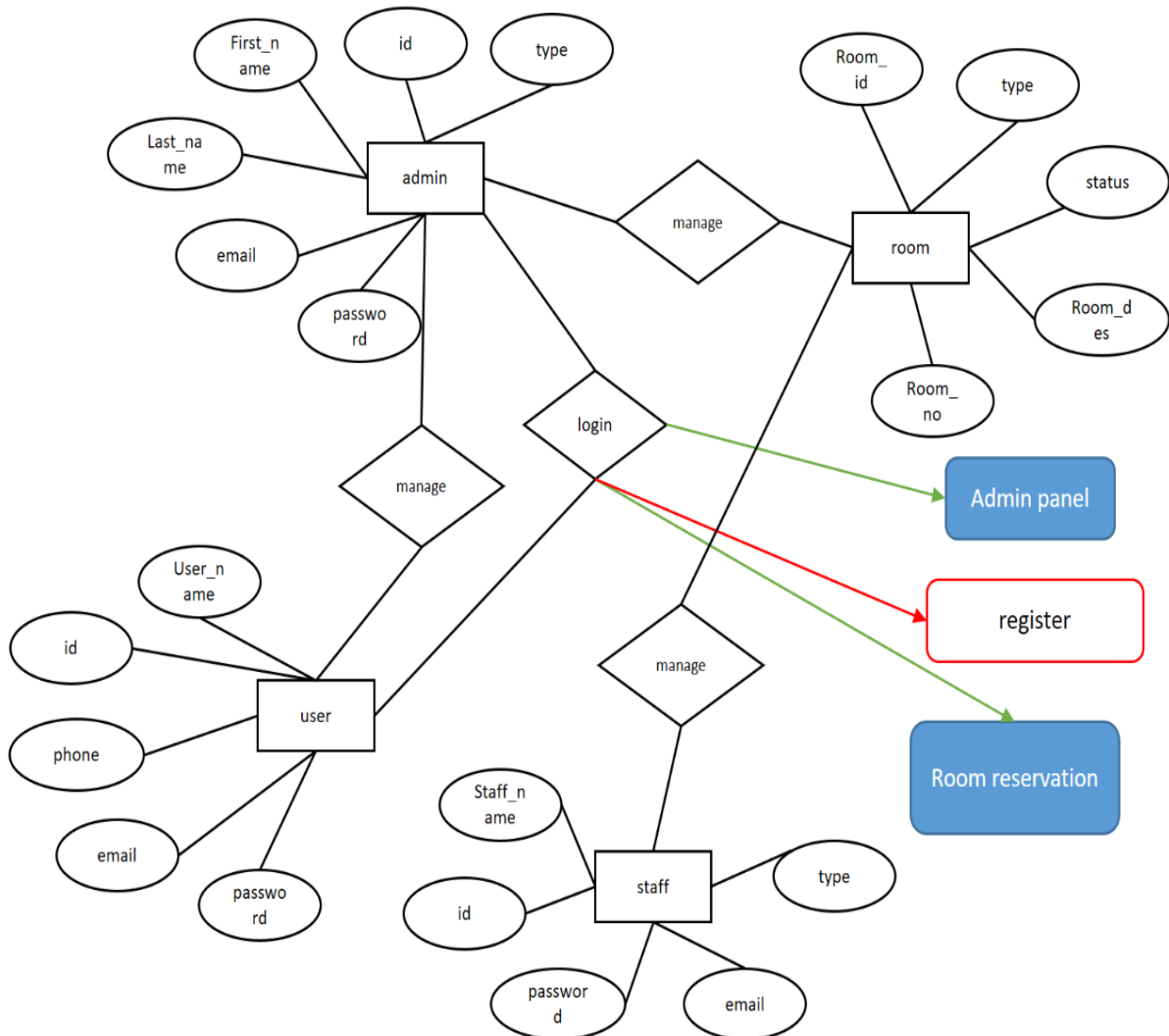
Figure: Entity relationship diagram of Hotel Booking System

## 3.6Collaboration diagram
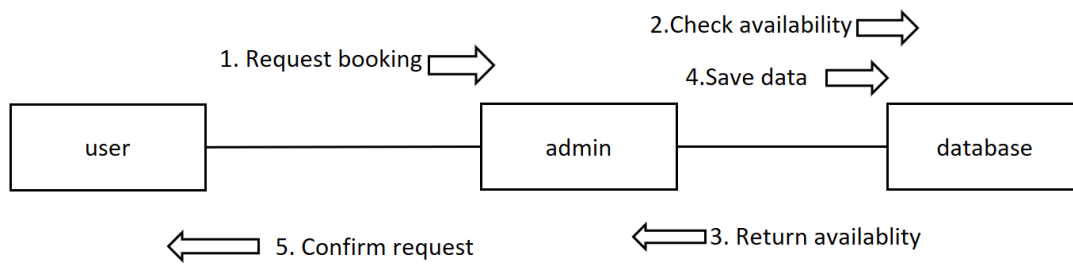


**Figure: Entity relationship diagram of Hotel Booking System**

# Chapter 4: Data definitions and Design snapshots

## 4.1Data definition

This section describes the database tables that are used in the hotel booking system.

## 4.2 Database tables of Hotel Booking system

Data tables used in the hotel booking system

### 4.2.1 Table structure for admin

This table contains the login details of admin

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action | | |
|---|------|------|-----------|------------|------|---------|----------|-------|--------|--|--|
| 1 | id | int(5) | | | No | None | | AUTO_INCREMENT | Change | Drop | More |
| 2 | first_name | varchar(20) | utf8mb4_general_ci | | No | None | | | Change | Drop | More |
| 3 | last_name | varchar(20) | utf8mb4_general_ci | | No | None | | | Change | Drop | More |
| 4 | type | int(3) | | | No | None | | | Change | Drop | More |
| 5 | email | varchar(30) | utf8mb4_general_ci | | No | None | | | Change | Drop | More |
| 6 | password | varchar(255) | utf8mb4_general_ci | | No | None | | | Change | Drop | More |

### 4.2.2 Table structure for users

This table contains the login details and other details of users

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action | | |
|---|------|------|-----------|------------|------|---------|----------|-------|--------|--|--|
| 1 | user_id | int(10) | | | No | None | | AUTO_INCREMENT | Change | Drop | More |
| 2 | user_name | varchar(50) | utf8mb4_general_ci | | No | None | | | Change | Drop | More |
| 3 | user_email | varchar(30) | utf8mb4_general_ci | | No | None | | | Change | Drop | More |
| 4 | user_password | varchar(255) | utf8mb4_general_ci | | No | None | | | Change | Drop | More |
| 5 | user_dt | datetime | | | No | None | | | Change | Drop | More |
| 6 | user_phone | varchar(10) | utf8mb4_general_ci | | No | None | | | Change | Drop | More |

### 4.2.3 Table structure for staff

This table contains the login details and other details of staff

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|------|------|-----------|------------|------|---------|----------|-------|--------|
| 1 | staff_id 🔑 | int(3) | | | No | None | | AUTO_INCREMENT | 🖉 Change ⊖ Drop ▾ More |
| 2 | staff_name | varchar(50) | utf8mb4_general_ci | | No | None | | | 🖉 Change ⊖ Drop ▾ More |
| 3 | satff_email | varchar(50) | utf8mb4_general_ci | | No | None | | | 🖉 Change ⊖ Drop ▾ More |
| 4 | satff_password | varchar(255) | utf8mb4_general_ci | | No | None | | | 🖉 Change ⊖ Drop ▾ More |
| 5 | satff_type | int(3) | | | No | None | | | 🖉 Change ⊖ Drop ▾ More |

### 4.2.4 Table structure for room types

This table contains the details of types of rooms

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|------|------|-----------|------------|------|---------|----------|-------|--------|
| 1 | r_id 🔑 | int(5) | | | No | None | | AUTO_INCREMENT | 🖉 Change ⊖ Drop ▾ More |
| 2 | r_name | varchar(20) | utf8mb4_general_ci | | No | None | | | 🖉 Change ⊖ Drop ▾ More |
| 3 | r_type | int(5) | | | No | None | | | 🖉 Change ⊖ Drop ▾ More |
| 4 | r_price | varchar(20) | utf8mb4_general_ci | | No | None | | | 🖉 Change ⊖ Drop ▾ More |

### 4.2.5 Table structure for all room details

This table contains the details of all of the rooms

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|------|------|-----------|------------|------|---------|----------|-------|--------|
| 1 | room_id 🔑 | int(5) | | | No | None | | AUTO_INCREMENT | 🖉 Change ⊖ Drop ▾ More |
| 2 | room_no | varchar(10) | utf8mb4_general_ci | | No | None | | | 🖉 Change ⊖ Drop ▾ More |
| 3 | room_type | varchar(20) | utf8mb4_general_ci | | No | None | refer room_type_details | | 🖉 Change ⊖ Drop ▾ More |
| 4 | room_status | varchar(20) | utf8mb4_general_ci | | No | None | 1=occupied, 0=unoccupied | | 🖉 Change ⊖ Drop ▾ More |
| 5 | room_des | varchar(100) | utf8mb4_general_ci | | No | None | | | 🖉 Change ⊖ Drop ▾ More |

**4.2.6 Table structure for all contact details**

This table contains the details of all of contact messages sent by users

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|------|------|-----------|------------|------|---------|----------|-------|--------|
| 1 | cont_id 🔑 | int(10) | | | No | None | | AUTO_INCREMENT | Change Drop More |
| 2 | cont_name | varchar(50) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 3 | cont_email | varchar(50) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 4 | cont_phone | varchar(10) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 5 | cont_number_guest | varchar(5) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 6 | cont_ci_date | date | | | No | None | | | Change Drop More |
| 7 | cont_co_date | date | | | No | None | | | Change Drop More |
| 8 | cont_message | varchar(255) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 9 | cont_date_time | datetime | | | No | current_timestamp() | | ON UPDATE CURRENT_TIMESTAMP() | Change Drop More |

**4.2.7 Table structure for all booking details**

This table contains the details of all of bookings by users

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|------|------|-----------|------------|------|---------|----------|-------|--------|
| 1 | res_id 🔑 | int(10) | | | No | None | | AUTO_INCREMENT | Change Drop More |
| 2 | res_name | varchar(50) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 3 | res_email | varchar(50) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 4 | res_phone | int(10) | | | No | None | | | Change Drop More |
| 5 | res_room | varchar(50) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 6 | res_no_of_room | varchar(60) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 7 | res_no_guests | varchar(60) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 8 | res_cidate | date | | | No | None | | | Change Drop More |
| 9 | res_codate | date | | | No | None | | | Change Drop More |
| 10 | res_message | varchar(255) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 11 | res_dt | datetime | | | No | None | | | Change Drop More |

# Chapter 5: System code and future scope

## 5.1 Login validation code

```php
// for existing user sign in
if(isset($_POST['si_email'])){
    $si_email = $_POST['si_email'];
    $si_password = base64_encode($_POST['si_password']);
    $sql = "SELECT * FROM `super_admin_login` WHERE `email`='".$si_email."'
AND `password`='".$si_password."';";
    // echo $sql;
    $result = mysqli_query($con,$sql);
    if(mysqli_num_rows($result)!=0){
        while($row=mysqli_fetch_array($result,MYSQLI_BOTH))
        {
            session_start();
            // $_SESSION['username'] = $row['name'];
            $_SESSION['username'] = 'admin';
            $_SESSION['email'] = $row['email'];
            $_SESSION['privil'] = $row['type'];
        }
        echo "<script>alert('Your login was
succefull!');window.location='../admin/dashboard.php';</script>";
    }
    else{
        $sql = "SELECT * FROM `staff_login` WHERE
`staff_email`='".$si_email."' AND `staff_password`='".$si_password."';";
        // echo $sql;
        $result = mysqli_query($con,$sql);
        if(mysqli_num_rows($result)!=0){
            while($row=mysqli_fetch_array($result,MYSQLI_BOTH))
            {
                session_start();
                // $_SESSION['username'] = $row['name'];
                $_SESSION['username'] = $row['staff_name'];
                $_SESSION['email'] = $row['satff_email'];
                $_SESSION['privil'] = $row['type'];
            }
            echo "<script>alert('Your login was
succefull!');window.location='../admin/dashboard.php';</script>";
        }
        else{
            $sql = "SELECT * FROM `user_data` WHERE
`user_email`='".$si_email."' AND `user_password`='".$si_password."';";
            // echo $sql;
            $result = mysqli_query($con,$sql);
            if(mysqli_num_rows($result)!=0){
```

```php
            while($row=mysqli_fetch_array($result,MYSQLI_BOTH))
            {
                session_start();
                $_SESSION['username'] = $row['user_name'];
                $_SESSION['email'] = $row['user_email'];
                $_SESSION['phone'] = $row['user_phone'];
            }
            echo "<script>alert('Your login was succefull! Book a room
now!');window.location='../default/book-room.php';</script>";
        }
        else{
            echo "<script>alert('Your login failed! try again or try
forgot password!');window.location='index.php';</script>";
        }
    }
}
}
```

## 5.2 Sign-Up code

```php
// for new user sign up
if(isset($_POST['su_email'])){
    $su_username = $_POST['su_username'];
    $su_email = $_POST['su_email'];
    $su_phone = $_POST['su_phone'];
    $su_password = base64_encode($_POST['su_password']);
    $sql = "SELECT * FROM `user_data` WHERE `user_email`='".$su_email."';";
    // echo $sql;
    $result = mysqli_query($con,$sql);
    if(mysqli_num_rows($result)==0){
        $sql = "INSERT INTO `user_data` (`user_name`, `user_email`,
`user_password`, `user_phone`, `user_dt`) VALUES ('$su_username', '$su_email',
'$su_password', '$su_phone', current_timestamp())";
        // echo $sql;
        $result = mysqli_query($con,$sql);

        session_start();
        $_SESSION['username'] = $su_username;
        $_SESSION['email'] = $su_email;
        $_SESSION['phone'] = $su_phone;
        echo "<script>alert('Your login was created succefully! you can now
book a room!');window.location='../default/book-room.php';</script>";
    }
    else{
```

```php
        echo "<script>alert('Your email already exists in our database, please
try forgot password!');window.location='forgot-password.php';</script>";
    }
}
```

## 5.3 Forgot password code

```php
// for forgot password
if(isset($_POST['fp_email'])){
    $fp_email = $_POST['fp_email'];
    $sql = "SELECT * FROM `user_data` WHERE `user_email`='".$fp_email."';";
    // echo $sql;
    $result = mysqli_query($con,$sql);
    if(mysqli_num_rows($result)!=0){
        $fpsql = "SELECT * FROM `user_data` WHERE `user_email`='".$fp_email."'
limit 1";
        // echo $fpsql;
        $fpresult = mysqli_query($con,$fpsql);
        if(mysqli_num_rows($fpresult)>0){
            while($row=mysqli_fetch_array($fpresult,MYSQLI_BOTH))
            {
                $fp_name = $row['user_name'];
                $fp_password = base64_decode($row['user_password']);
            }
            $to = $fp_email;
            $subject = "You used forgot password method to retrive password";
            $body = "Dear user ".$fp_name.", Thank you for using our services
your password is ".$fp_password."   -Mariana Hotel";
            $headers = "From: thunderninja1521@gmail.com";

            mail($to,$subject,$body,$headers);
            echo "<script>alert('Password has been sent to your email
address');window.location='index.php';</script>";
        }
        else{
            echo "<script>alert('There was an error, please try
again!');window.location='forgot-password.php';</script>";
        }
    }
    else{
        echo "<script>alert('Your email address does not exist in server, try
signing-up!');window.location='sign-up.php';</script>";
    }
}
```

## 5.4 Contact us code

```php
include("../includes/connection.php");

// Collect post variables
$name = $_POST['cont_name'];
$email = $_POST['cont_email'];
$phone = $_POST['cont_phone'];
$number_guests = $_POST['cont_number_guests'];
$cidate = $_POST['cont_ci_date'];
$codate = $_POST['cont_co_date'];
$message = $_POST['cont_message'];
$t = time();
// mysql query for no repeatation check
$sql = "SELECT * FROM `contact_details` WHERE `cont_name`='$name' AND
`cont_phone`='$phone' AND `cont_ci_date`='$cidate' AND
`cont_co_date`='$codate'";
$result = mysqli_query($con, $sql);

if(mysqli_num_rows($result)!=0){
    echo "<script>alert('Your enquiry has already been sent, please be patient
we will cantact you!');window.location='testing.php';</script>";
}

else{
    // mysql query for insertion
    $sql = "INSERT INTO `contact_details` (`cont_name`, `cont_email`,
`cont_phone`, `cont_number_guest`,  `cont_ci_date`, `cont_co_date`,
`cont_message`, `cont_date_time`) VALUES ('$name', '$email', '$phone',
'$number_guests', '$cidate', '$codate', '$message', current_timestamp())";
    // echo $sql;
    $result = mysqli_query($con, $sql);

    if ($result == 1) {
        echo "<script>alert('Your enquiry has been
sent!');window.location='testing.php';</script>";
    } else {
        echo "ERROR: $sql <br> $con->error";
    }
}
```

## 5.5 Room booking code

```php
include("../includes/connection.php");

// Collect post variables
$name = $_POST['res_name'];
$email = $_POST['res_email'];
$phone = $_POST['res_phone'];
$room_type = $_POST['res_room_type'];
$number_rooms = $_POST['res_number_rooms'];
$number_guests = $_POST['res_number_guests'];
$ci_date = $_POST['res_ci_date'];
$co_date = $_POST['res_co_date'];
$message = $_POST['res_message'];
$dt = time();
// mysql query for no repeatation check
$sql = "SELECT * FROM `room_booking` WHERE `res_name`='$name' AND
`res_phone`='$phone' AND `res_email`='$email' AND `res_cidate`='$ci_date' AND
`res_codate`='$co_date' AND `res_room`='$room_type' AND
`res_no_of_room`='$number_rooms' AND `res_no_guests`='$number_guests'";
$result = mysqli_query($con, $sql);

if(mysqli_num_rows($result)!=0){
    echo "<script>alert('Your booking is already
done!');window.location='index.php';</script>";
}

else{
    // mysql query for insertion
    $sql = "INSERT INTO `room_booking` (`res_name`, `res_email`, `res_phone`,
`res_room`, `res_no_of_room`, `res_no_guests`, `res_cidate`, `res_codate`,
`res_message`, `res_dt`) VALUES ('$name', '$email', '$phone', '$room_type',
'$number_rooms', '$number_guests', '$ci_date', '$co_date', '$message',
current_timestamp());";
    // echo $sql.'<br>';
    $result = mysqli_query($con, $sql);

    if ($result == 1) {
        echo "<script>alert('Your reservation was
successful!');window.location='index.php';</script>";
    } else {
        echo "ERROR: $sql <br> $con->error";
    }
}
```

## 5.6 Limitations

- There is no security protection such as SSL, Sitelock.
- SMS system is unavailable at the moment.
- There is no payment getaway active as of now.
- Receipt generation is still unavailable

## 5.7 Future scope

The future development of this project is possible and the project is scalable. Some of the modules can be upgraded as well as the site animation can be improved and refined too. Some of the main future scope improvements are as follows:

- Online payment system/ getaway.
- Phone confirmation system.
- Site can be more dynamic with some improvements.
- Multiple Hotel can be integrated together in this system after some amount of work.

# REFERENCES

1. [Online].available: Template design
   https://templatemo.com/

2. [Online].available: melsatar,

   "Software development life cycle models and methodologies," Mohamed  Sami, 2012 .[OnlineAvailable]:https://melsatar.wordpress.com/2012/03/15/software-development-life-cyclemodels-and-methodologies.

3. [Online].available: https://www.tutorialspoint.com/sdlc/sdlc_tutorial.pdf.

4. [Online].Available: P. Sparrow and 4 penna, "Waterfall model: Advantages and disadvantages of waterfall model,". [Online]. Available: http://www.ianswer4u.com/2011/11/advantages-and-disadvantagesof.html#axzz4M06BvkOA.

5. [Online].Available:
   https://www.online-image-editor.com/
   https://www.resizepixel.com/
   https://www.adobe.com/express/create/logo

6. [Online].Available:

   https://youtu.be/0yMtUkR1qcE

   https://youtu.be/

7. [Online].Available:

   Code with harry: https://youtu.be/1SnPKhCdlsU