

# **TRAVELONE - RESORT MANAGEMENT SYSTEM**

*Project Report Submitted By*

**ANNAPOORNESWARI D**

**Reg. No.: AJC20MCA-2020**

*In Partial fulfillment for the Award of the Degree of*

**MASTER OF COMPUTER APPLICATIONS  
(REGULAR MCA)  
APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**

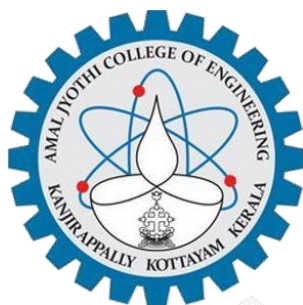


**AMAL JYOTHI COLLEGE OF ENGINEERING  
KANJIRAPPALLY**

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE,  
Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

**2021-2022**

**DEPARTMENT OF COMPUTER APPLICATIONS**  
**AMAL JYOTHI COLLEGE OF ENGINEERING**  
**KANJIRAPPALLY**



**CERTIFICATE**

This is to certify that the Project report, “**TRAVELONE – RESORT MANAGEMENT SYSTEM**” is the bonafide work of **ANNAPOORNESWARI D (Reg.No:AJC20MCA-2020)** in partial fulfillment of the requirements for the award of the Graduation of Regular Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2021-2022.

**Dr. Bijimol T K**  
**Internal Guide**

**Rev. Fr. Dr. Rubin Thottupurathu Jose**  
**Coordinator**

**Rev. Fr. Dr. Rubin Thottupurathu Jose**  
**Head of the Department**

## **DECLARATION**

I hereby declare that the project report “**TRAVELONE - RESORT MANAGEMENT SYSTEM**” is a bonafided work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Degree of Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2021-2022.

**Date:**

**KANJIRAPPALLY**

**ANNAPOORNESWARID**

**Reg. No: AJC20MCA-2020**

## ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department and the project coordinator **Rev. Fr. Dr. Rubin Thottupurathu Jose** for helping us and I extend my whole hearted thanks for his valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also like to express sincere gratitude to my guide, **Dr. Bijimol T K** for her inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions and encouragement to make this venture a success.

ANNAPOORNESWARI D

## **ABSTRACT**

The major goal of the resort management system project is to construct an online web portal for the atomization of the resort room booking system and to open up the possibility of receiving visitors from other regions. This application will assist in enhancing both visitor services and resort management's ability to generate income. TravelOne enables users to book the resort room of their choice. The end consumer can browse the resort, resort rooms, trip packages, restaurant menu, and events thanks to this website. Additionally, it enables visitors to register on the website and retain their own unique profile. By logging in, he or she can view and edit their profile.

The user account keeps track of the customer's personal information and preferred room. Additionally, the profile module allows you to view the reservation he or she has made. Reserving may be cancelled by user. TravelOne wants to provide customers with an effective, educational, and easy-to-use website where they can make reservations for their travels.

Currently, there is a manual process in place where visitors must call to inquire about the availability of rooms and places to visit in a given area. As internet usage has grown, creating online portals will be advantageous and atomized.

# CONTENT

<b>Sl. No</b>	<b>Topic</b>	<b>Page No</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
<b>1.1</b>	<b>PROJECT OVERVIEW</b>	<b>2</b>
<b>1.2</b>	<b>PROJECT SPECIFICATION</b>	<b>2</b>
<b>2</b>	<b>SYSTEM STUDY</b>	<b>3</b>
<b>2.1</b>	<b>INTRODUCTION</b>	<b>4</b>
<b>2.2</b>	<b>EXISTING SYSTEM</b>	<b>5</b>
<b>2.3</b>	<b>DRAWBACKS OF EXISTING SYSTEM</b>	<b>5</b>
<b>2.4</b>	<b>PROPOSED SYSTEM</b>	<b>5</b>
<b>2.5</b>	<b>ADVANTAGES OF PROPOSED SYSTEM</b>	<b>5</b>
<b>3</b>	<b>REQUIREMENT ANALYSIS</b>	<b>6</b>
<b>3.1</b>	<b>FEASIBILITY STUDY</b>	<b>7</b>
<b>3.1.1</b>	<b>ECONOMICAL FEASIBILITY</b>	<b>7</b>
<b>3.1.2</b>	<b>TECHNICAL FEASIBILITY</b>	<b>8</b>
<b>3.1.3</b>	<b>BEHAVIORAL FEASIBILITY</b>	<b>8</b>
<b>3.2</b>	<b>SYSTEM SPECIFICATION</b>	<b>9</b>
<b>3.2.1</b>	<b>HARDWARE SPECIFICATION</b>	<b>9</b>
<b>3.2.2</b>	<b>SOFTWARE SPECIFICATION</b>	<b>9</b>
<b>3.3</b>	<b>SOFTWARE DESCRIPTION</b>	<b>9</b>
<b>3.3.1</b>	<b>PHP</b>	<b>9</b>
<b>3.3.2</b>	<b>MYSQL</b>	<b>10</b>
<b>4</b>	<b>SYSTEM DESIGN</b>	<b>12</b>
<b>4.1</b>	<b>INTRODUCTION</b>	<b>13</b>
<b>4.2</b>	<b>UML DIAGRAM</b>	<b>13</b>
<b>4.2.1</b>	<b>USE CASE DIAGRAM</b>	<b>14</b>
<b>4.2.2</b>	<b>SEQUENCE DIAGRAM</b>	<b>17</b>
<b>4.2.3</b>	<b>STATE CHART DIAGRAM</b>	<b>20</b>
<b>4.2.4</b>	<b>ACTIVITY DIAGRAM</b>	<b>21</b>
<b>4.2.5</b>	<b>CLASS DIAGRAM</b>	<b>22</b>
<b>4.2.6</b>	<b>OBJECT DIAGRAM</b>	<b>23</b>
<b>4.2.7</b>	<b>COMPONENT DIAGRAM</b>	<b>24</b>
<b>4.2.8</b>	<b>DEPLOYMENT DIAGRAMS</b>	<b>25</b>
<b>4.3</b>	<b>USER INTERFACE DESIGN</b>	<b>26</b>

<b>4..4</b>	<b>DATA BASE DESIGN</b>	<b>32</b>
<b>5</b>	<b>SYSTEM TESTING</b>	<b>43</b>
<b>5.1</b>	<b>INTRODUCTION</b>	<b>44</b>
<b>5.2</b>	<b>TEST PLAN</b>	<b>45</b>
<b>5.2.1</b>	<b>UNIT TESTING</b>	<b>45</b>
<b>5.2.2</b>	<b>INTEGRATION TESTING</b>	<b>46</b>
<b>5.2.3</b>	<b>VALIDATION TESTING</b>	<b>46</b>
<b>5.2.4</b>	<b>USER ACCEPTANCE TESTING</b>	<b>46</b>
<b>6</b>	<b>IMPLEMENTATION</b>	<b>52</b>
<b>6.1</b>	<b>INTRODUCTION</b>	<b>53</b>
<b>6.2</b>	<b>IMPLEMENTATION PROCEDURE</b>	<b>53</b>
<b>6.2.1</b>	<b>USER TRAINING</b>	<b>54</b>
<b>6.2.2</b>	<b>TRAINING ON APPLICATION SOFTWARE</b>	<b>54</b>
<b>6.2.3</b>	<b>SYSTEM MAINTENANCE</b>	<b>54</b>
<b>7</b>	<b>CONCLUSION &amp; FUTURE SCOPE</b>	<b>55</b>
<b>7.1</b>	<b>CONCLUSION</b>	<b>55</b>
<b>8</b>	<b>BIBLIOGRAPHY</b>	<b>57</b>
<b>9</b>	<b>APPENDIX</b>	<b>59</b>
<b>9.1</b>	<b>SAMPLE CODE</b>	<b>60</b>
<b>9.2</b>	<b>SREENSHOTS</b>	<b>66</b>

## List of Abbreviation

IDE	-	Integrated Development Environment
HTML	-	Hyper Text Markup Language.
CSS	-	Cascading Style Sheet
SQL	-	Structured Query Language
UML	-	Unified Modeling Language



# **CHAPTER 1**

## **INTRODUCTION**

## **1.1 PROJECT OVERVIEW**

The major goal of the resort management system project is to construct an online web portal for the atomization of the resort room booking system and to open up the possibility of receiving visitors from other regions. This application will assist in enhancing both visitor services and resort management's ability to generate income. An internet tool called TravelOne enables users to book the resort room of their choice. The end consumer can browse the resort, resort rooms, trip packages, restaurant menu, and events thanks to this website.

## **1.2 PROJECT SPECIFICATION**

This project intends to provide a resort management system that clients may use to make online reservations for rooms and other facilities. The availability of rooms and other amenities can be checked by users.

The system includes 2 users. They are:

### **1. Admin**

This system requires an administrator login. He is in charge of the entire system. The administrator can manage user data, add or edit service details, etc. The administrator can view every user who has signed up and handle all of his data.

### **2. User**

Customers or users can register and edit profiles. They can look up and reserve available rooms and other amenities. Get information about nearby visiting places.

## **CHAPTER 2**

### **SYSTEM STUDY**

## 2.1 INTRODUCTION

The process of gathering and evaluating data, finding problems, and using the knowledge to recommend system modifications. There must be extensive contact between the system developers and the system users during this problem-solving process. Any system development process should begin with a system analysis or research. The system is carefully inspected and evaluated. The system analyst adopts the role of the interrogator and meticulously probes questions on how the current system functions. The input into the system is acknowledged, and the system is seen as a whole. The different processes can be related to the organizational outputs. System analysis involves understanding the issue, identifying the pertinent and important aspects, evaluating and synthesis the numerous components, and selecting the best or, at the very least, most acceptable course of action.

The procedure has to be carefully investigated utilizing a range of approaches, such as surveys and interviews. The data acquired by various sources has to be thoroughly evaluated in order to draw a conclusion. The conclusion is knowing how the system functions. This system is known as the present one. Trouble locations have been identified now that the problem with the existing system has been extensively analyzed. The designer now assumes the position of a problem-solver and makes an effort to address the issues the business is facing. Proposals are used in place of the solutions. Following an analytical comparison of the plan and the current system, the best choice is chosen. The user is given the chance to accept or reject the idea when it is presented to them. Depending on user feedback, the idea is examined to see whether any changes are necessary.

Preliminary research is the procedure of gathering and analyzing data in order to use it for upcoming system investigations. Initial research requires strong collaboration between system users and developers since it involves problem-solving. It carries out several feasibility studies. The system activities are roughly estimated by these studies, which may be utilized to choose the methods to employ for effective system research and analysis.

## **2.2 EXISTING SYSTEM**

Prior to the internet, the current method required those who wanted to use the amenities to physically visit the resort to reserve rooms and make inquiries. The resort's management must manually keep track of all the rooms.

Since the advent of the internet, both customers and resort management have had trust concerns.

## **2.3 DRAWBACKS OF EXISTING SYSTEM**

- The procedure is time-consuming.
- There is no guarantee that rooms will be available; it takes a lot of time.
- Paperwork necessitates a lot of storage space for the data.
- Problems with consumer and resort management trust.

## **2.4 PROPOSED SYSTEM**

To fix the issues with the manual system. Customers can speak with the Resort directly. The main goal of a resort management system is to make it possible to access all of the resort's amenities online. This program allows for online resort management and record-keeping.

## **2.5 ADVANTAGES OF PROPOSED SYSTEM**

- Allows guests to immediately reserve all resort amenities;
- Allows guests to book rooms whenever they want, from anywhere they have internet.
- Disclose information about resort amenities.
- Direct contact with the hotel
- Convenient.

## **CHAPTER 3**

### **REQUIREMENT ANALYSIS**

### **3.1 FEASIBILITY STUDY**

To ascertain if the project will ultimately achieve the objectives of the organization given the labour, effort, and time invested in it, a feasibility study is carried out. A feasibility study enables the project's designer to foresee the project's possible future and usefulness. The effectiveness of a system proposal is assessed by considering how it will impact the organization, user needs, and resource usage. As a result, a feasibility analysis is often conducted before a new proposal is approved for development.

The feasibility of the project is described in the paper, along with a list of the several aspects—such as technical, economic, and operational viability—that were carefully considered throughout the project's feasibility study. It has the following characteristics :-

#### **3.1.1 Economical Feasibility**

The growing system has to be supported by cost-benefit evaluations. The undertaking that attracts the most interest must also be the one that yields the best results the quickest. The expense of developing a new system is one of the variables.

Following are some important financial questions that were brought up during the original investigation:

- The expenses take the entire system into account.
- The cost of the hardware and software;
- The benefits in terms of lower costs or less costly errors.

The recommended solution was established as part of a project, thus there are no manual expenses associated with it. The system may also be put into place at a reasonable cost given the availability of the required resources.

Depending on the system employed and its development costs, the project's cost for the employment exchange system was divided. All calculations indicate that the project was developed at a modest cost. As open source software was used to develop it entirely.

### 3.1.2 Technical Feasibility

The system must go through a technical assessment first. An overview design of the system's requirements in terms of input, output, program, and procedures must be the foundation for the viability evaluation. The investigation must next identify an outline system and suggest the kind of equipment, necessary construction processes, and operation procedures for the system after it has been built.

Technical issues raised during the investigation are:

- Is the technology now available sufficient for the suggested work?
- Could the system grow if it were developed?

The project should be planned so that the required performance and functionality are met within the limits. The system may still be used even if the technology may become outdated over time since a newer version of the same software still works with an older version. There aren't many limitations on this project as a result. Technically, the project can be completed because the system was created utilizing PHP for the front end and a MySQL server for the back end. The pre-owned computer is equipped with an effective Pentium CPU, 8GB of RAM, and a 500GB hard drive.

### 3.1.3 Behavioral Feasibility

The following inquiries are part of the suggested system:

- Does the users' help meet their needs?
- Will the suggested system damage anyone?

Because it would accomplish the objectives after being developed and put into action, the project would be advantageous. After carefully examining all behavioral parameters, it is determined that the project is behaviorally viable.



---

## 3.2 SYSTEM SPECIFICATION

### 3.2.1 Hardware Specification

Processor - A M D R y z e n 5

RAM - 8 GB

Hard disk - 500 GB

### 3.2.2 Software Specification

Front End - HTML, CSS

Back end - MYSQL

Client on PC - Windows 11 and above.

Technologies used - JS, J Query, PHP

## 3.3 SOFTWARE DESCRIPTION

### 3.3.1 PHP

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language is now installed on more than 244 million websites and 2.1 million web servers. The PHP group is presently developing the reference version of PHP, which Rasmus Lerdorf invented in 1995. Personal Home Page, as it was once named, is now known as PHP: Instead of accessing an external file to process data, a web server uses Hypertext Preprocessor instructions that may be put directly into an HTML source document and understands the recursive acronym code to construct the final web page. It has also evolved to incorporate a command-line interface capability and may be used independently due to restrictions on the usage of the word PHP, rendering it incompatible with the GNU General Public License (GPL). The vast majority of web servers support installing PHP and running it as a standalone shell on virtually all platforms and operating systems.

### 3.3.2 MySQL

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. The MySQL Website provides the latest information about MySQL software.

- **MySQL is a database management system.**

A database is a planned collection of data. It might be anything, like a simple grocery list, a photo gallery, or the vast amount of information in a corporate network. Data included in a computer database must be added to, accessed, and processed using a database management system, such as MySQL Server. Because computers are so good at processing enormous amounts of data, database management systems—whether employed as standalone program or as a component of other applications—are crucial to computing.

- **MySQL databases are relational.**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You setup rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well- designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data. The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax. SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL92” refers to the standard released in 1992,

The terms “SQL:1999” and “SQL:2003” refer to the standard's previous and most recent iterations, respectively. The SQL Standard as it exists at any one moment is referred to as “the SQL standard.”

- **MySQL software is Open Source.**

Being open source, the application may be used and modified by anybody. Everybody may access and use the MySQL software for free online.

You have the right to go at the source code and make any required changes. The MySQL software is licensed under the GPL (GNU General Public License), which details what you may and cannot do with the program under specific conditions. If the GPL makes you uncomfortable or if you need to incorporate MySQL code into a for-profit application, you may buy a commercially licensed version from us. Check out the MySQL Licensing Overview for more information.

- **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

If it is what you're wanting, you should try it. MySQL Server may run without a hitch on a desktop or laptop and require little to no maintenance, in addition to your other program, web servers, and other software. If you use a whole machine for MySQL, you may change the settings to make full use of the RAM, CPU, and I/O.

- **MySQL Server works in client/server or embedded systems.**

A multi-threaded SQL server, several client applications and libraries, management tools, and a wide range of application programming interfaces are all included in the client/server MySQL Database Software (APIs). Additionally, you may link our MySQL Server integrated multi-threaded library to your application to produce a distinct project that is more manageable, speedier, and less in size.

## **CHAPTER 4**

### **SYSTEM DESIGN**

## 4.1 INTRODUCTION

Design is the first step in the creation of any technical system or product. Design is a creative process. A good design is the key to a system that works effectively. "Design" is the process of employing many approaches and concepts to thoroughly outline a process or a system so that it may be physically implemented. It may be summed up as the process of employing several concepts and strategies to specify a device, processor, or system with insufficient specificity to enable physical implementation. Software design serves as the technical foundation of the software engineering process, regardless of the development paradigm used. The system design generates the architectural detail required to build a system or product. This software underwent the best design phase imaginable, fine-tuning all efficiency, performance, and accuracy levels, as with any systematic approach. A user-oriented document becomes a document for programmers or database specialists during the design phase. The two stages of system design development are logical design and physical design.

## 4.2 UML DIAGRAM

A common language known as UML is used to describe, visualize, build, and record the software system artefacts. The Object Management Group (OMG) was responsible for developing UML, and a draught of the UML 1.0 definition was presented to the OMG in January 1997.

Unified Modeling Language, or UML, is distinct from other popular programming languages like C++, Java, COBOL, etc. As a general-purpose visual modelling language to visualize, design, develop, and record software systems, UML is a pictorial language used to create software blueprints. Although representing software systems is the most popular use of UML, it is not the only one. Simulating non-software systems is another use for it. the process flow in the manufacturing plant, etc. Despite not being a programming language, UML may be used with tools to generate code in a number of other languages. UML is intimately connected to the analysis and design of objects-oriented systems. UML has been standardized the degree where it is now an OMG standard. A complete UML diagram that shows a system is made up of all the pieces and relationships. The visual effect of the UML diagram is the most important factor in the entire process. Together with all the other parts, the next nine schematics are used to complete it.

- Class diagram
- Object diagram
- Use case diagram
- Sequence diagram
- Collaboration diagram
- Activity diagram
- State chart diagram
- Deployment diagram
- Component diagram

### **4.2.1 USE CASE DIAGRAM**

An illustration of the interactions between system components is a use case diagram. A use case is a method for locating, defining, and organizing system needs. The word "system" in this context refers to a thing that is being built or operated, such as a website for the selling of products and services through mail order. Use case diagrams are used in UML (Unified Representing Language), a standard notation for modelling real products and systems.

Planning general requirements, validating hardware designs, testing and debugging software products while they are still in development, creating online help resources, and finishing customer support-focused tasks are a few examples of system objectives. For instance, customer support, item ordering, catalogue updating, and payment processing are examples of use cases in a setting of product sales. A use case diagram consists of four components.

- The border, which isolates the system of interest from its surroundings.
- The performers, who are often system participants identified by the roles they play.
- The players within and around the system play the roles specified by the use cases.
- The connections and interactions between the actors and use cases.

Use case diagrams are created to depict a system's functional needs. To create an effective use case diagram after identifying the aforementioned things, we must adhere to the following rules.

- A use case's naming is highly significant. The name should be selected in a way that makes it clear what functions are being performed.
- Give actors a name that fits them.
- Clearly depict links and dependencies in the diagram.
- Keep in mind that the diagram's primary function is to indicate the needs; do not attempt to include all possible links.

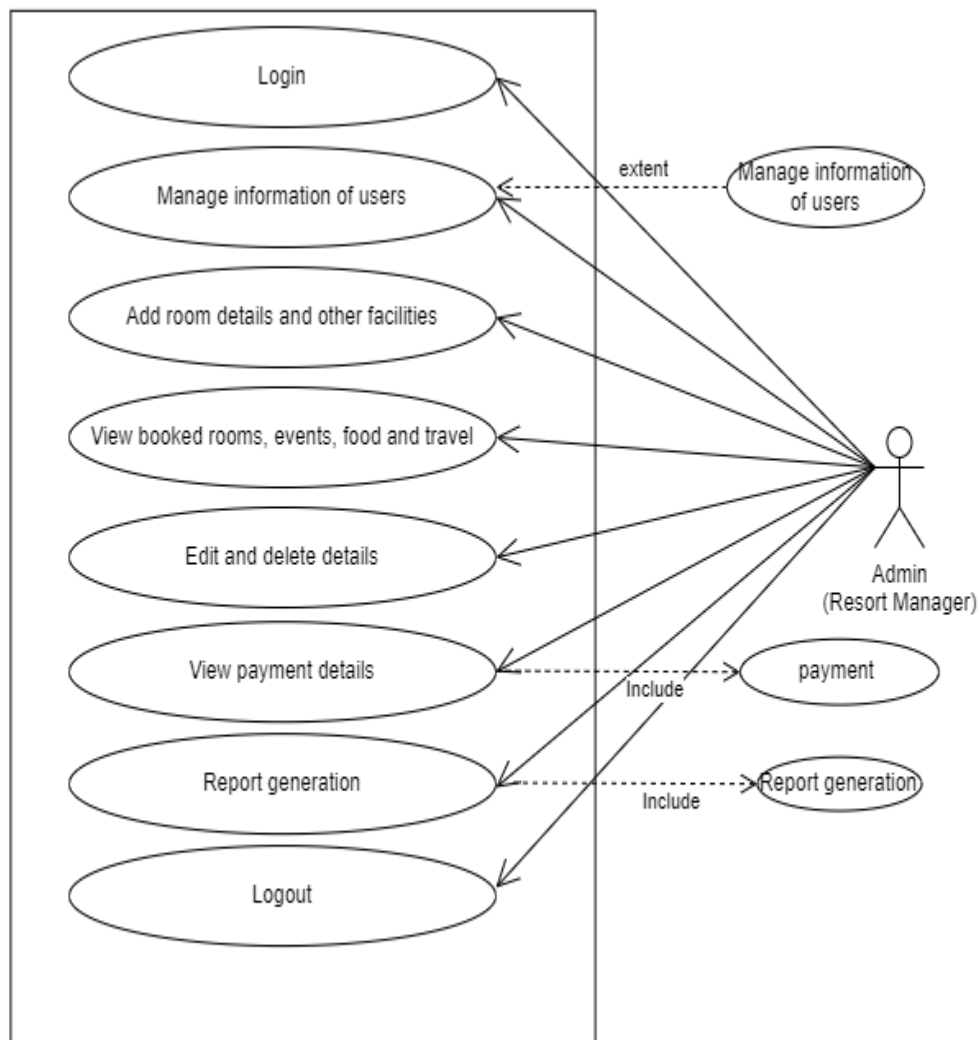


Figure 4.1.a Use case diagram for Admin

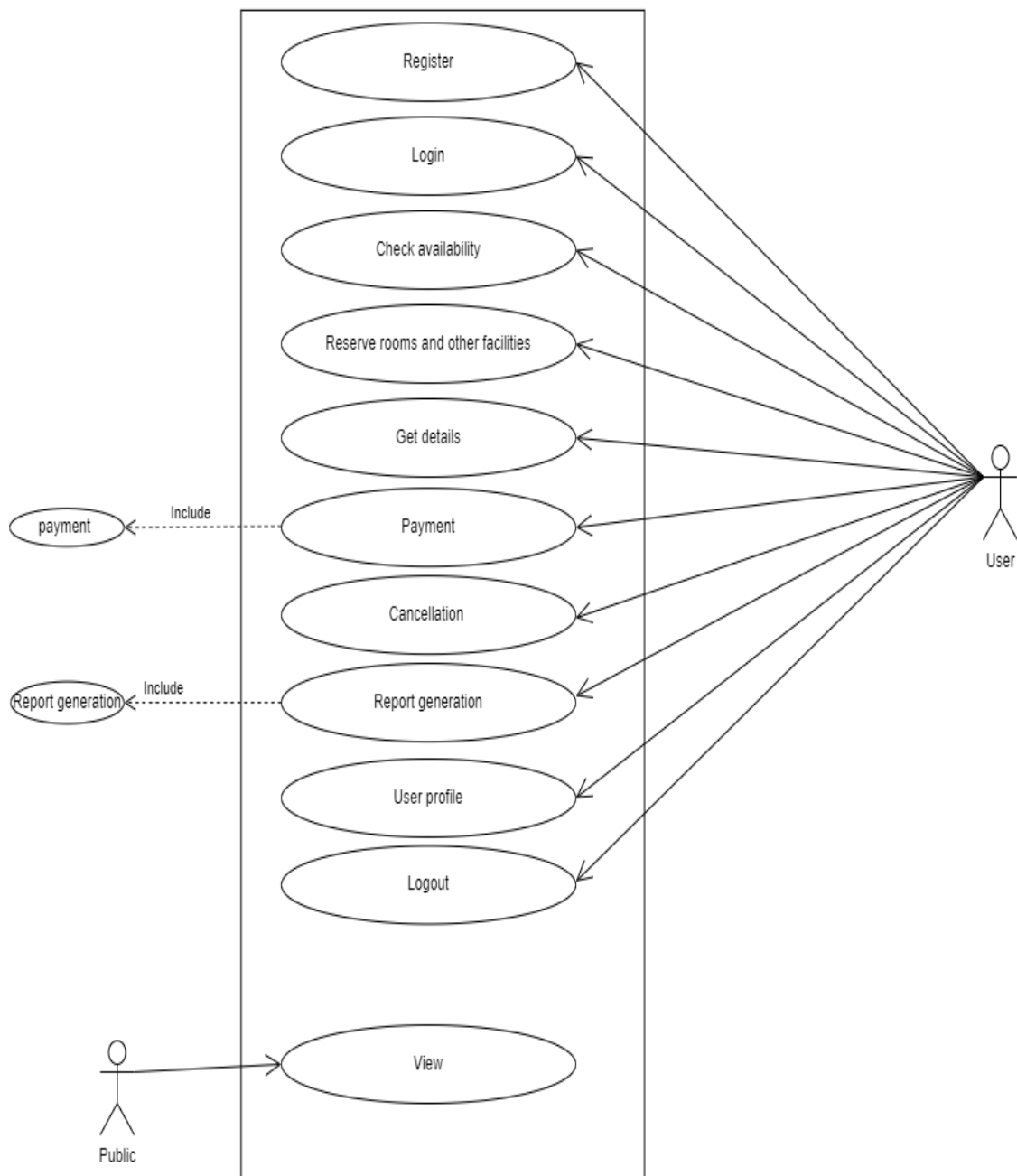


Figure 4.1.b Use case diagram for User



## 4.2.2 SEQUENCE DIAGRAM

A sequence diagram fundamentally depicts the sequential order in which events occur or how they interact with one another. Event diagrams and event scenarios are other names for sequence diagrams. Sequence diagrams display the activities performed by a system's parts in time order. These diagrams are widely used by businesspeople and software engineers to document and explain the requirements for new and existing systems.

### Sequence Diagram Notations –

- i. Actors** – An actor in a UML diagram symbolise a certain sort of role in which it interacts with the system's elements. An actor is never inside the scope of the system that we want to describe using the UML diagram. For a range of roles, including those of human users and other external topics, we use actors. An actor is shown using the stick person notation in a UML diagram. A sequence diagram could have several actors.
- ii. Lifelines** – A lifeline is a named component that displays a particular participant in a sequence diagram. Each episode is effectively represented by a lifeline in a sequence diagram. In a sequence diagram, the lifeline components are at the top.
- iii. Messages** – It is shown how things may communicate with one another through messages. The lifeline displays the messages in reverse chronological order. Messages are represented by arrows. The two major elements of a sequence diagram are lifelines and messages.

Messages can be broadly classified into the following categories:

- Synchronous messages
- Asynchronous Messages
- Create message
- Delete Message
- Self-Message
- Reply Message
- Found Message
- Lost Message

**iv. Guards** – In the UML, We use guards to simulate situations. We employ them when we need to restrict communication under the pretends that a criterion has been met. Software developers rely on guards to let them know when a system or particular technique has restrictions.

#### Uses of sequence diagrams –

- Used to model and visualize the logic behind a sophisticated function, operation or procedure.
- They are also used to show details of UML use case diagrams.
- Used to understand the detailed functionality of current or future systems.
- Visualize how messages and tasks move between objects or components in a system.

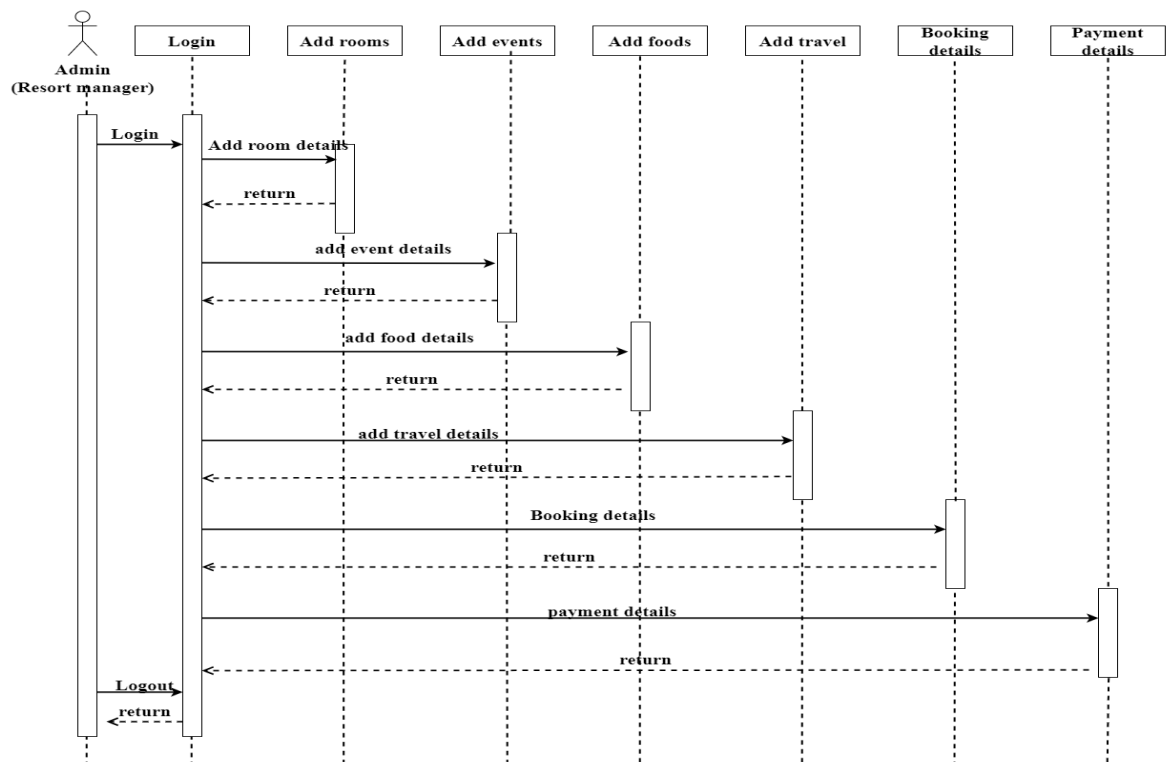


Figure 4.2.a Sequence diagram for Admin(Resort manager)

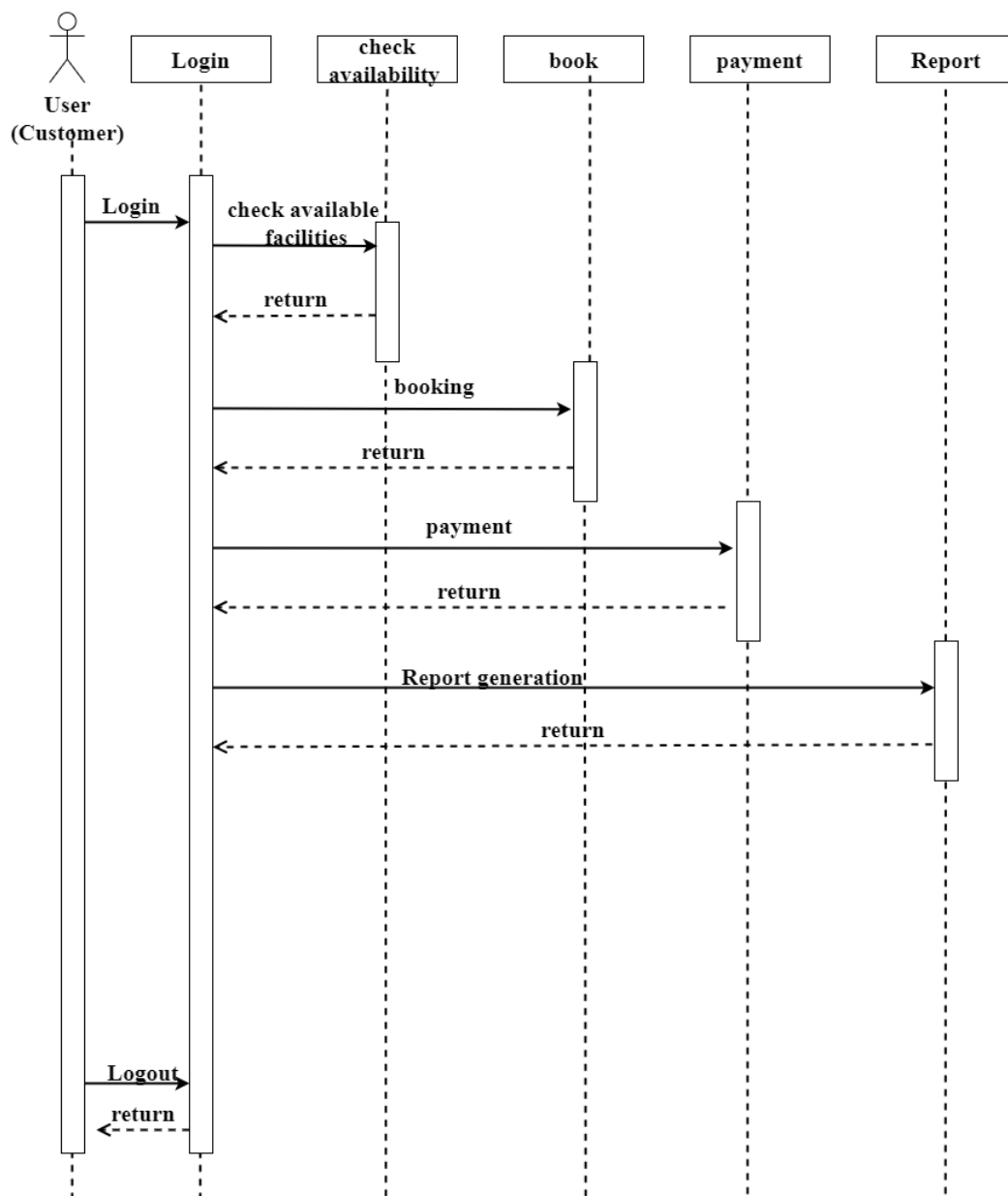


Figure 4.2.b Sequence diagram for User (Customer)

### 4.2.3 State Chart Diagram

State diagrams are a common tool for displaying a software system's behavior. A state machine diagram in a UML model can describe the behavior of a class, a subsystem, a package, or even an entire system. Other names for it include state charts and state transition diagrams. We can effectively describe the communications or interactions that occur between external entities and a system using state chart diagrams. These diagrams are used to model the event-based system. An event can be used to control an object's state. State chart diagrams are used in the application system to show the various states of an object.

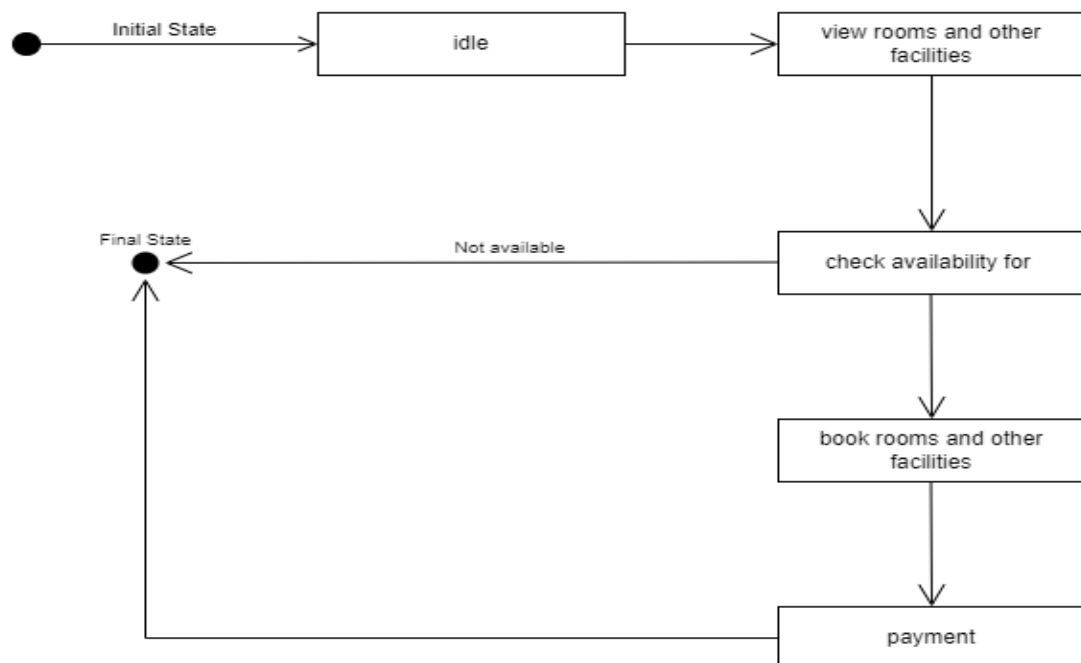


Figure 4.3 State diagram

### 4.2.4 Activity Diagram

Activity diagrams show how different levels of activity abstraction must work together to produce a service. An event often takes a few operations to complete, especially when those operations must coordinate several independent processes. Another common need is understanding how the events in a single use case relate to one another, particularly in use cases where activities may overlap and require coordination. It is also appropriate for modelling how a group of coordinated use cases might reflect business activity.

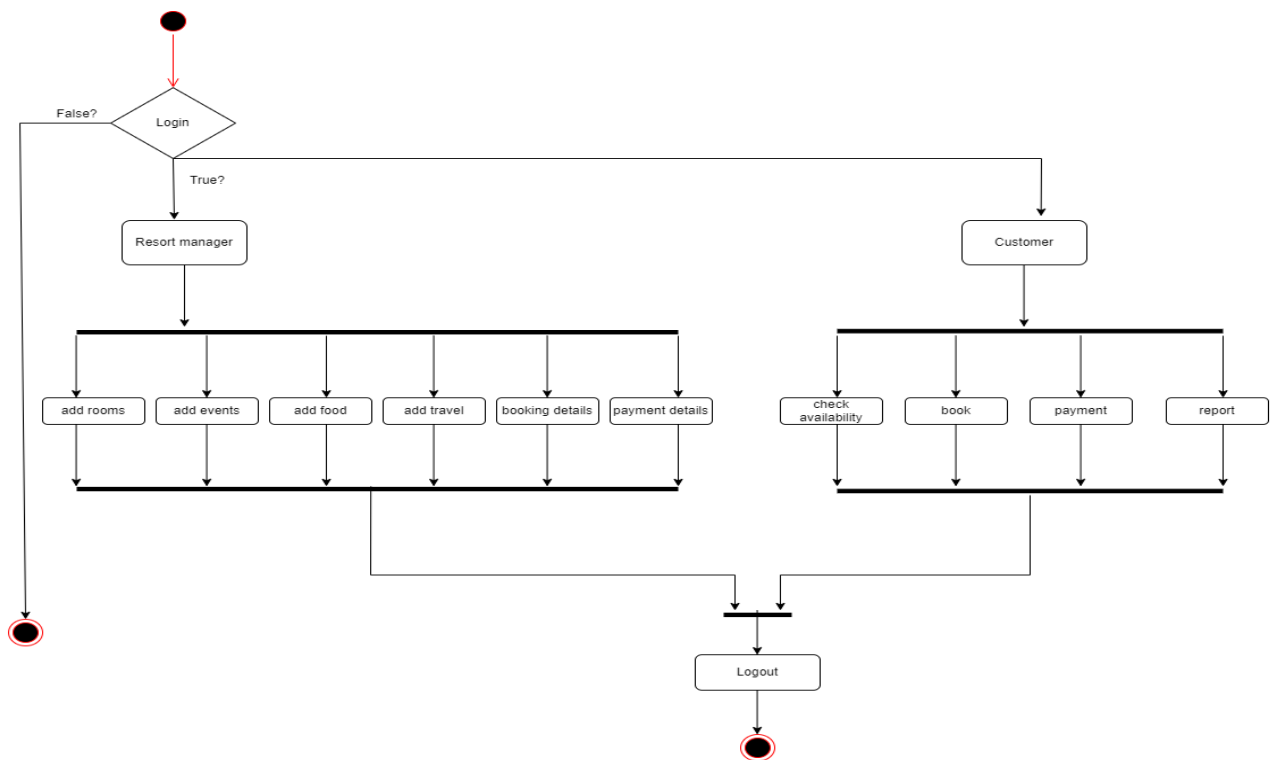


Figure 4.4 Activity diagram

### 4.2.5 Class Diagram

Static diagrams include class diagrams. It represents the application's static view. Class diagrams are helpful for developing executable code for software program as well as for visualizing, explaining, and documenting a variety of system components. A class diagram depicts a class's operations and characteristics together with the limitations placed on the system. Since class diagrams are the only UML diagrams that can be immediately translated using object-oriented languages, they are often employed in the modelling of object-oriented systems. An assortment of classes, interfaces, affiliations, collaborations, and restrictions are displayed in a class diagram. Alternatively called a "structural diagram,"

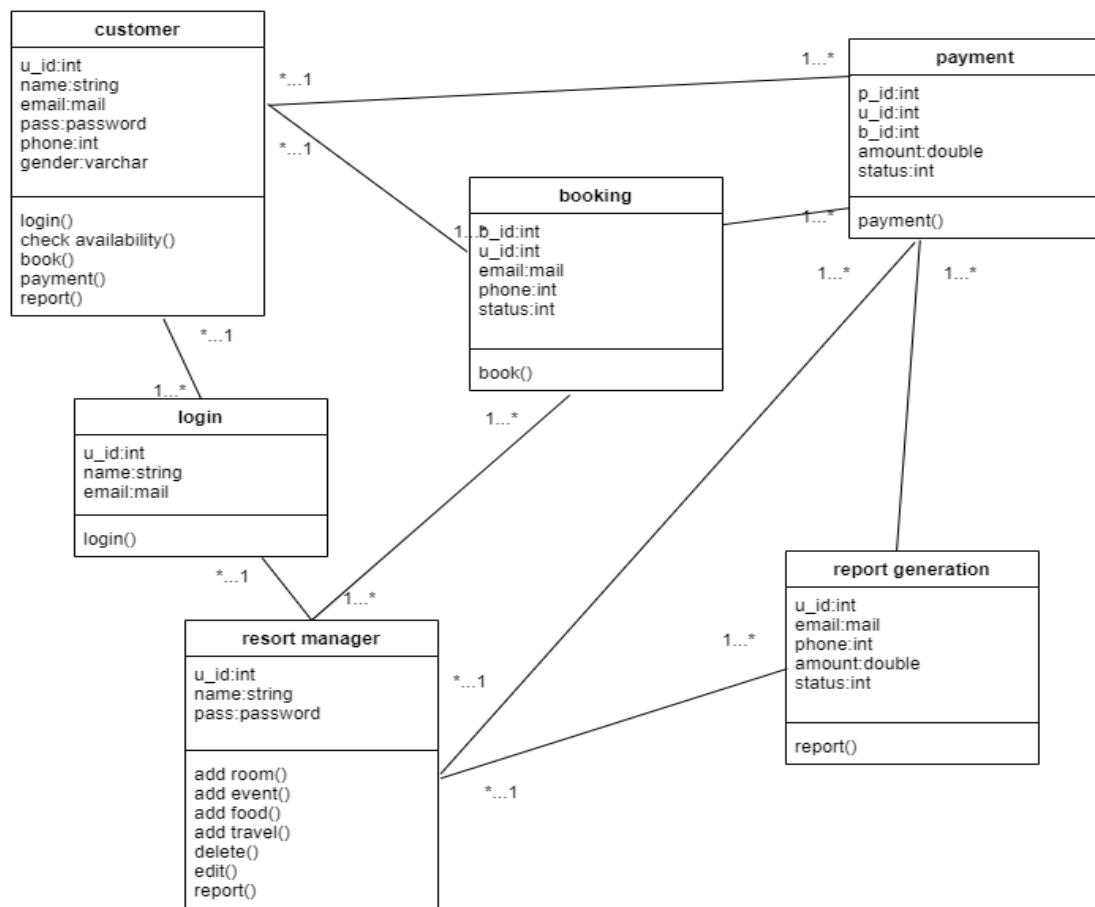


Figure 4.5 Class diagram

### 4.2.6 Object Diagram

Since object diagrams are the source of class diagrams, class diagrams are a need for object diagrams. A particular instance of a class diagram is shown in an object diagram. The core ideas in class and object diagrams are same. Additionally, object diagrams are used to show a system's static view, which is a fleeting snapshot of the system. Object diagrams are used, for instance, to show a group of items and how they are connected.

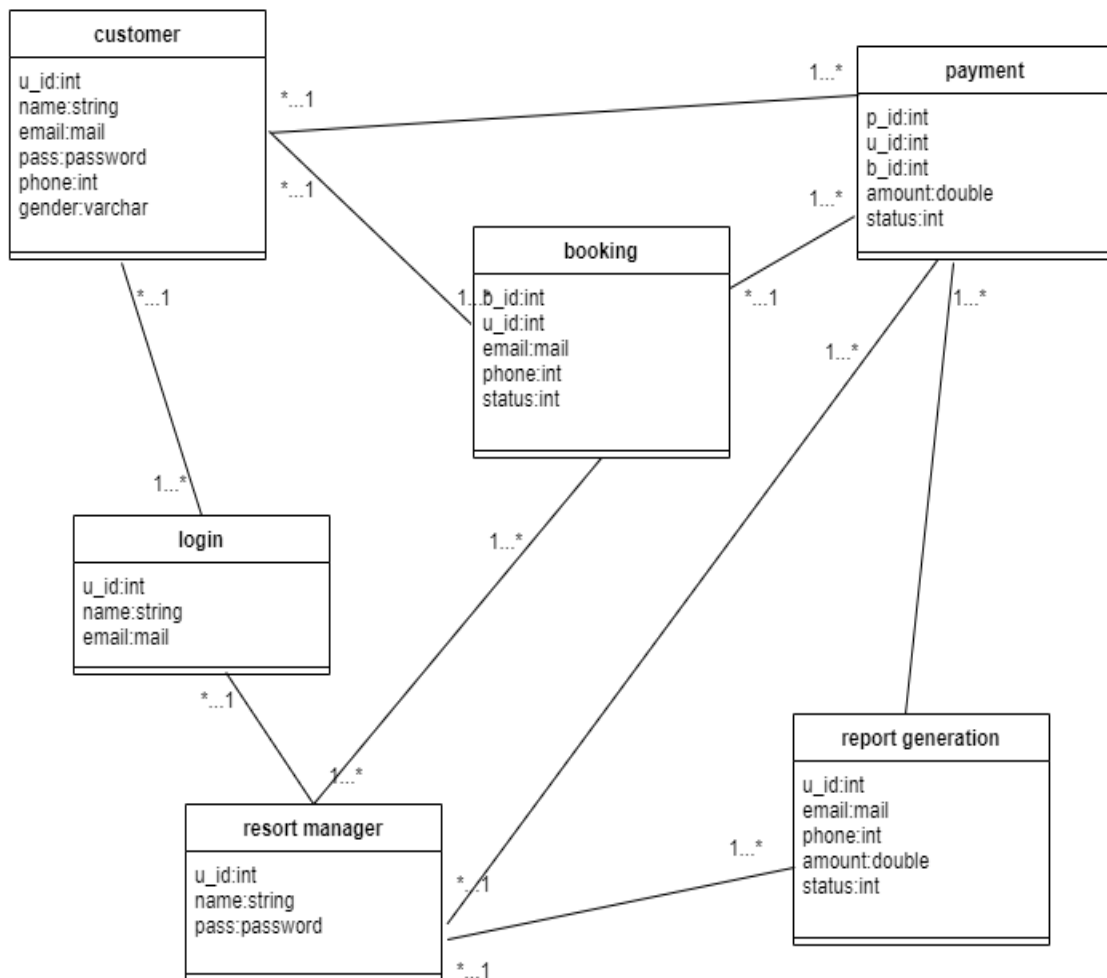


Figure 4.6 Object diagram

### 4.2.7 Component Diagram

Different behaviors and personalities may be found in component diagrams. Component diagrams are used to depict the system's physical components. Examples include things that are physically present in a node, such as executables, libraries, files, and documents. The connections and configuration of a system's components are displayed using component diagrams. Systems that can be put into use can also be built using these designs.

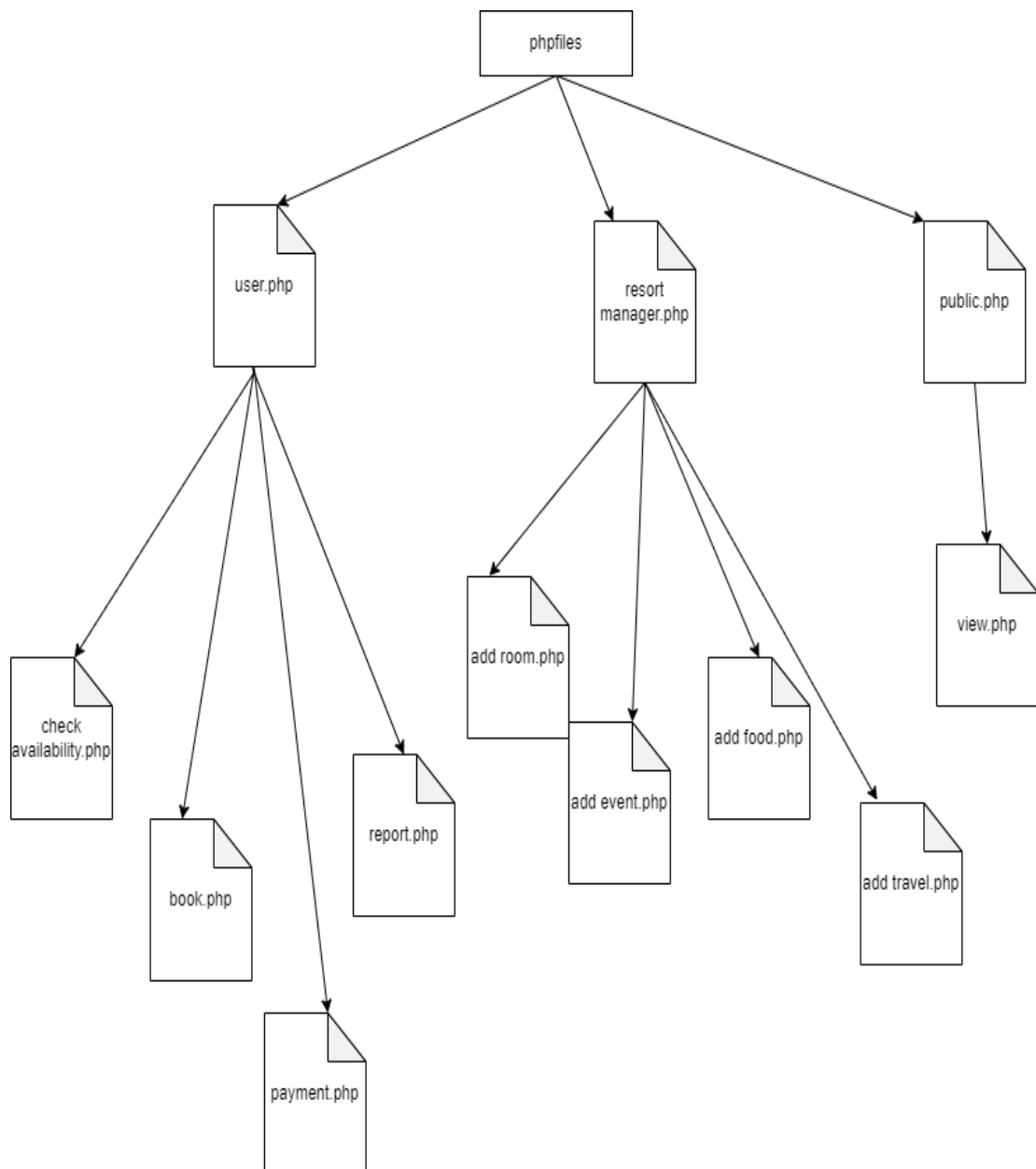
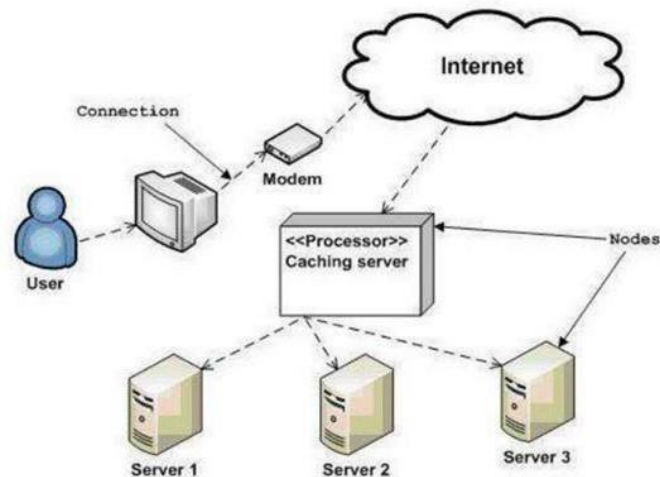


Figure 4.7 Component diagram



### 4.2.8 DEPLOYMENT DIAGRAM

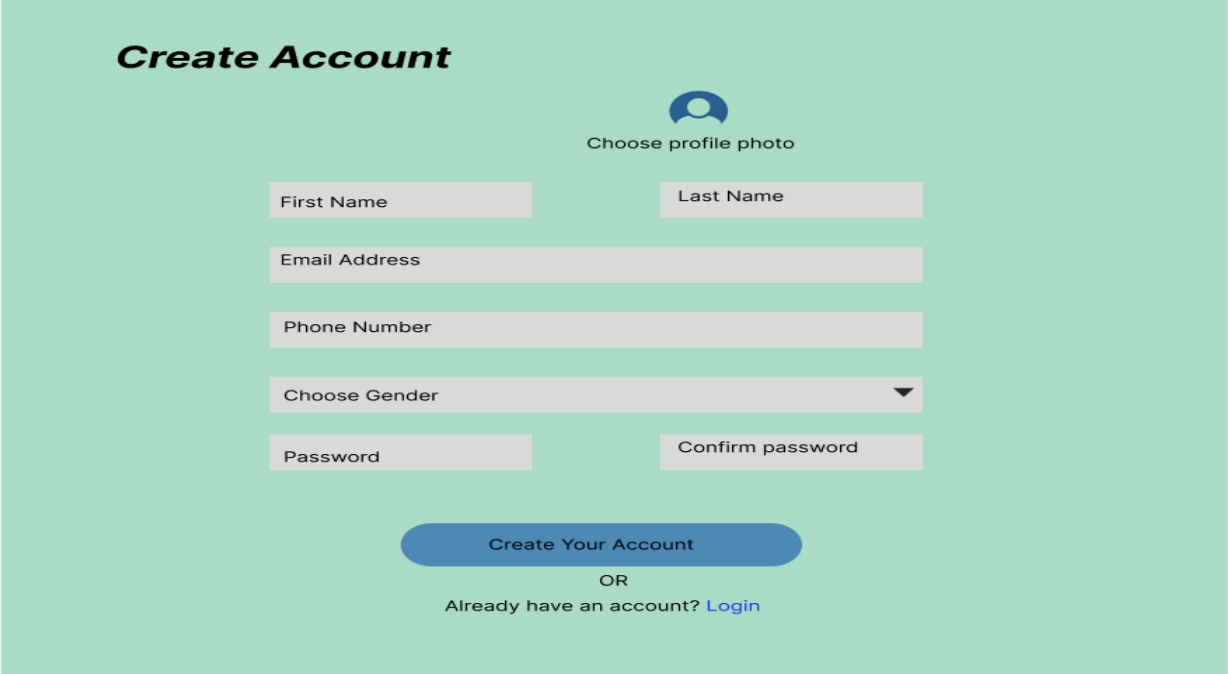
An execution architecture of a system, containing nodes like hardware or software execution environments, and the middleware linking them, is shown in a deployment diagram, a form of UML diagram. Typically, deployment diagrams are used to represent the actual hardware and software of a system.



## 4.3 USER INTERFACE DESIGN

### 4.3.1-INPUT DESIGN

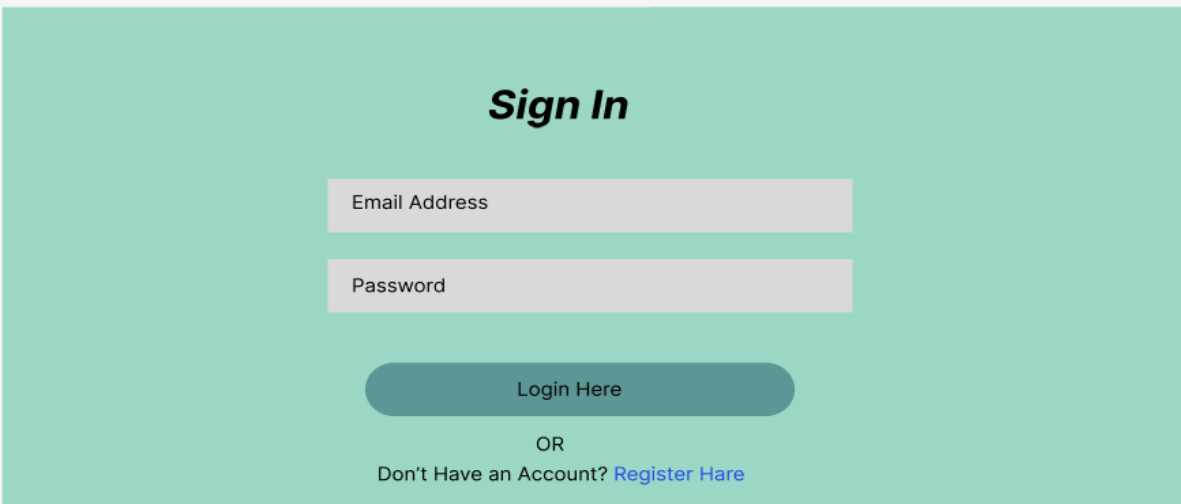
Form Name : User Registration



The 'Create Account' form is displayed on a light green background. It features a title 'Create Account' in bold black text. Below the title is a blue circular icon with a white person silhouette, followed by the text 'Choose profile photo'. The form contains several input fields: 'First Name' and 'Last Name' (side-by-side), 'Email Address' (full width), 'Phone Number' (full width), 'Choose Gender' (dropdown menu), 'Password' and 'Confirm password' (side-by-side). A blue rounded button labeled 'Create Your Account' is centered below the fields. Below the button is the text 'OR' and a link 'Already have an account? Login'.

Figure 4.8 User registration

Form Name : User Login



The 'Sign In' form is displayed on a light green background. It features a title 'Sign In' in bold black text. Below the title are two input fields: 'Email Address' and 'Password'. A blue rounded button labeled 'Login Here' is centered below the fields. Below the button is the text 'OR' and a link 'Don't Have an Account? Register Here'.

Figure 4.9 User login

Form Name : Home Page

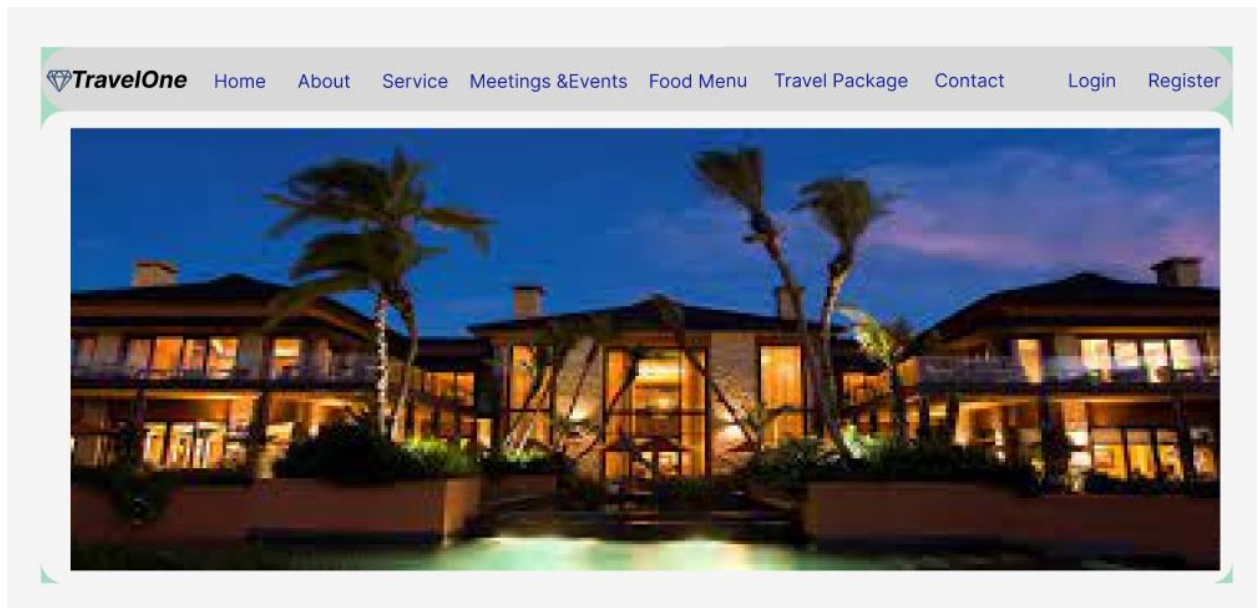


Figure 4.10 Home page

Form Name : User Page

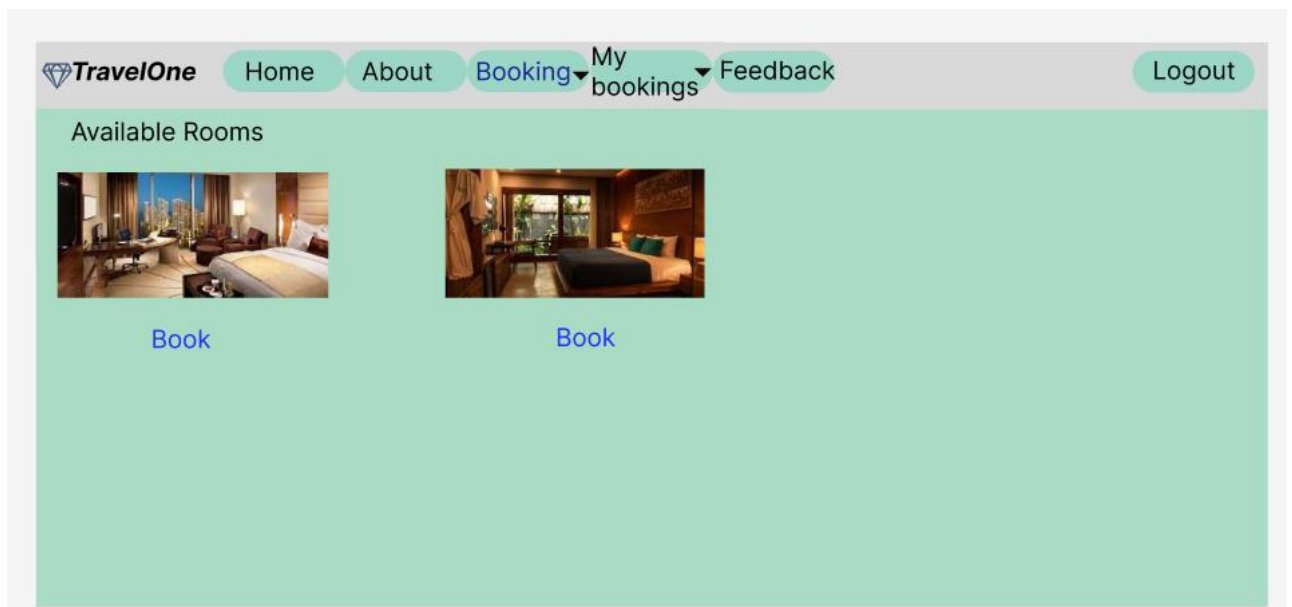


Figure 4.11 User Page

Form Name : Admin Dashboard

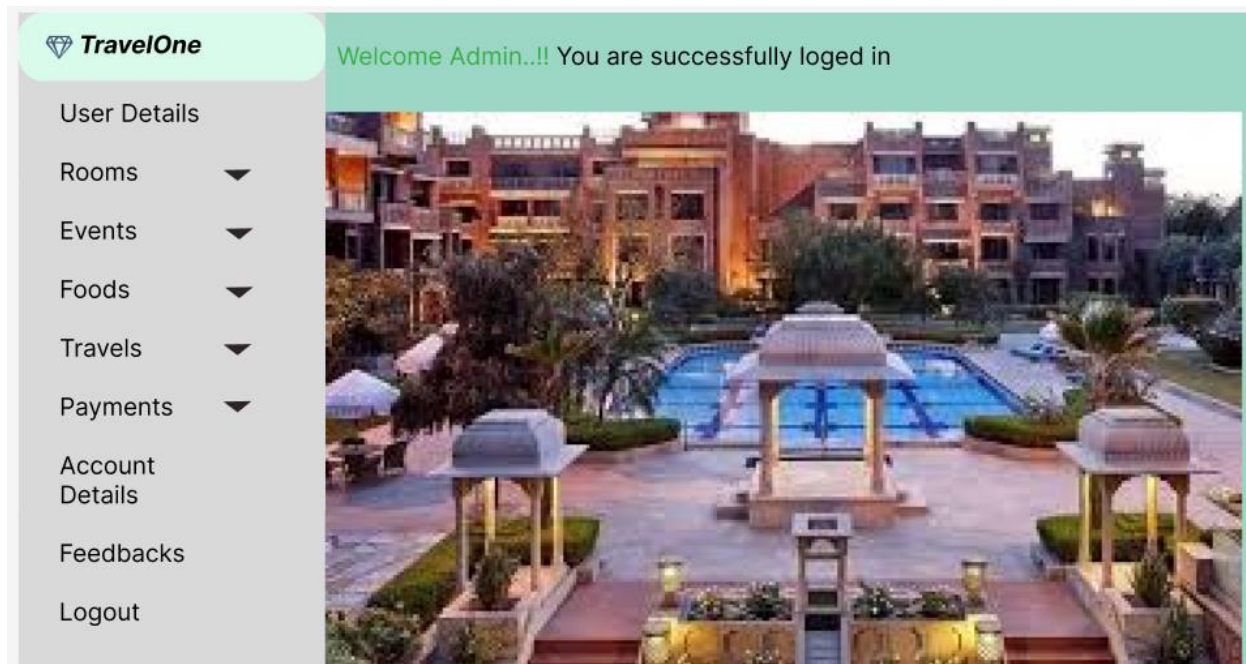
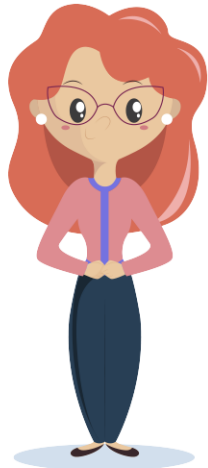


Figure 4.13 Admin dashboard

### 4.3.2 OUTPUT DESIGN

#### User Login



**Sign in**


**Sign in**


or

Don't have an account? [Sign up here!](#)

Figure 4.14 User login

#### User Registration



  
Choose Picture

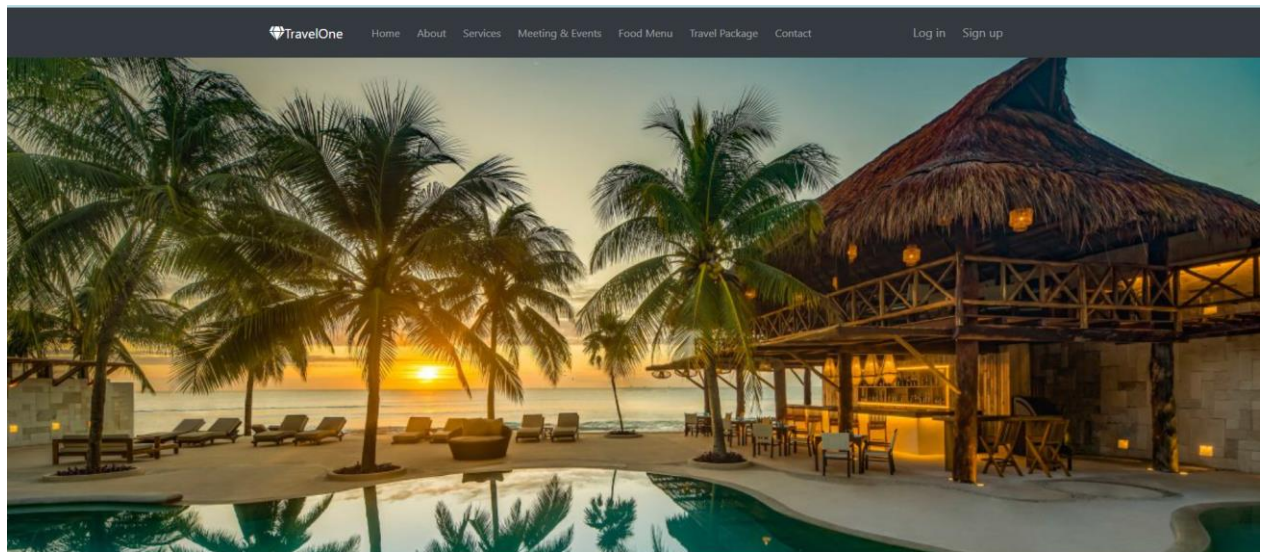
**Create your account**

OR

Already Registered? [Login](#)

Figure 4.15 User registration

## Home Page



TravelOne, Kochi.

Figure 4.16 Home Page

## About Us

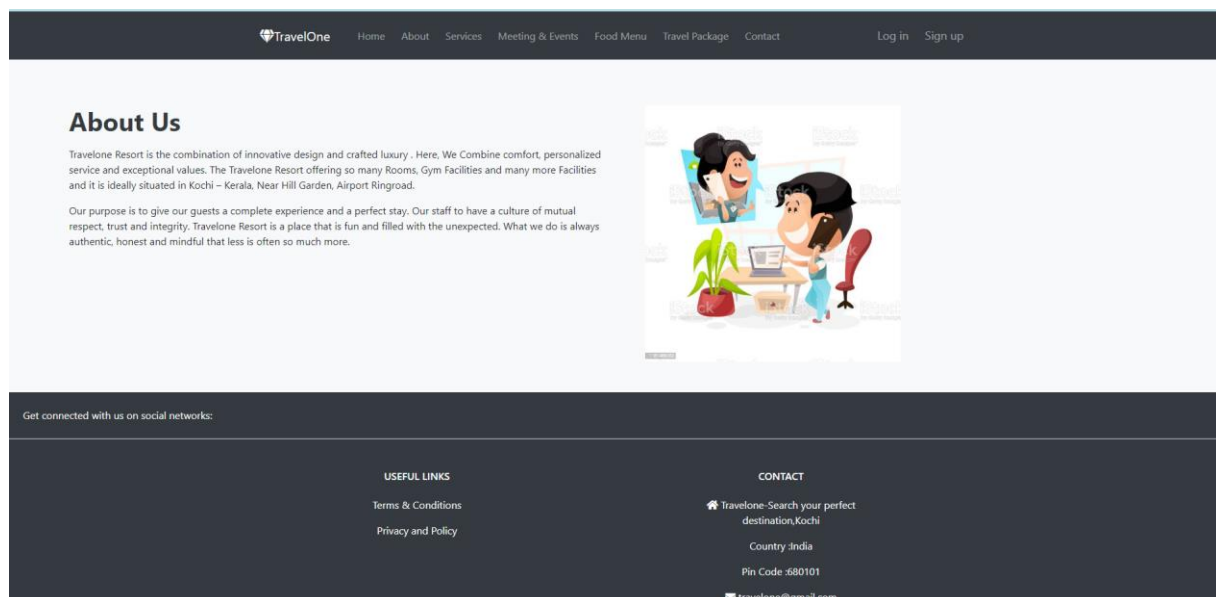


Figure 4.17 About us

## Contact Us

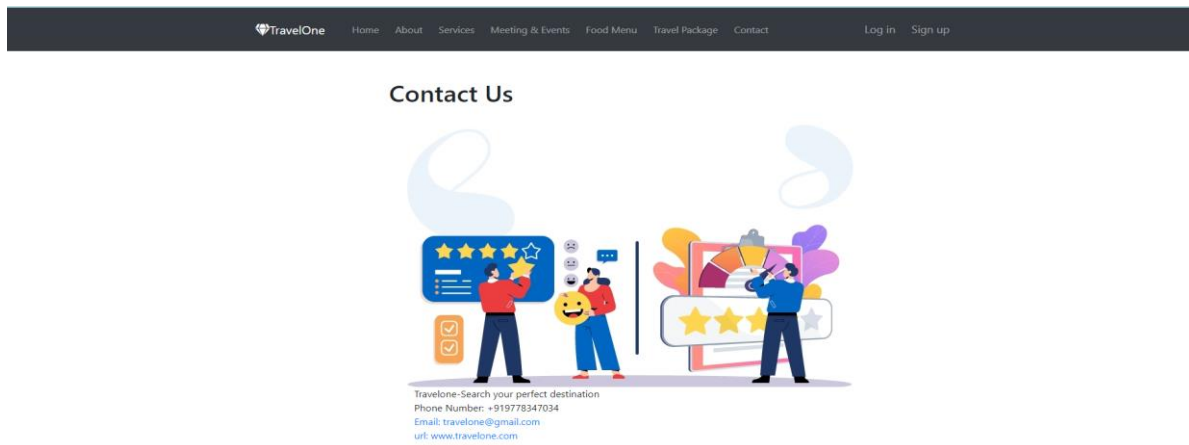


Figure 4.18 Contact us

## 4.4 DATABASE DESIGN

A database is an organized system that has the ability to store information and gives users rapid, efficient access to that information. The main objective of every database is its data, which must be secured.

The database design process is divided into two steps. In the first stage, the user demands are ascertained, and a database is made to as clearly as possible meet these requirements. Information level design is a procedure that is carried out separately from all DBMSs.

In the second step, the information level design for the specific DBMS that will be used to build the system in question is transformed into a design. The properties of the specific DBMS that will be used are discussed at the Physical Level Design stage. A database design runs parallel to the system design. The two major objectives of the database's data structure are outlined below.

- Data Integrity
- Data independence

### 4.4.1 Relational Database Management System (RDBMS)

In a relational paradigm, the database is shown as a collection of relations. To each relation, a table or file of records with values can be compared. A row is known as a tuple, a column heading is known as an attribute, and the table is known as a relation in formal relational model language. A relational data base is made up of several tables, each with a distinct name. A collection of connected values are reflected in each row of a story.

#### Relations, Domains & Attributes

A table is a connection. Tuples are the units of a table's rows. An ordered group of n items is referred to as a tuple. Attributes are referred to as columns. The database already has links between all of the tables. This guarantees the validity of both referential and entity links. A group of atomic values make up a domain D. Choosing a data type from which the domain's data values are derived is a popular way to define a domain. In order to make the domain's values easier to understand, it is also beneficial to give the domain a name. The values of a connection are all atomic and interdependent.



## Relationships

- Key is used to create table connections. Primary Key and Foreign Key are the two principal keys that are most crucial. With the use of these keys, relationships for entity integrity and referential integrity may be created.
- Entity Integrity forbids the use of null values for any Primary Key.
- No Primary Key may contain null values, according to Referential Integrity.
- Referential Integrity: A Primary Key value in the same domain must correspond to each unique Foreign Key value. Super Key and Candidate Keys are additional keys.

### 4.4.2 Normalization

The simplest possible grouping of data is used to put them together so that future modifications may be made with little influence on the data structures. The formal process of normalizing data structures in a way that reduces duplication and fosters integrity. Using the normalization approach, superfluous fields are removed and a huge table is divided into several smaller ones. Anomalies in insertion, deletion, and updating are also prevented by using it. Keys and relationships are two notions used in the standard style of data modelling. A row in a table is uniquely identified by a key. Primary keys and foreign keys are the two different kinds of keys. In a database, a primary key is an element, or group of elements, that is used to identify between entries belonging to the same table. A column in a database known as a foreign key is used to uniquely identify records from other tables. Up to the third normal form, all tables have been normalized.

As the name implies, it refers to arranging things in their natural state. The application developer uses normalization to provide a logical grouping of the data into suitable tables and columns so that users can immediately link names to the data. Normalization avoids data redundancy, which puts a large strain on the computer's resources by eliminating repeated groupings from the data. These consist of:

- Normalize the data.
- Choose proper names for the tables and columns.
- Choose the proper name for the data.

---

**First Normal Form**

Each attribute's domain must only include atomic values, and each attribute's value in a tuple must be a single value from that domain, according to the First Normal Form. Using relationships as attribute values within tuples or relations within relations is prohibited by 1NF, in other words. Under 1NF, only single atomic or indivisible values are allowed for attribute values. The data entry must be done using First Normal Form. To do this, data may be separated into tables in each table that are of a similar type. Depending on the needs of the project, either a Primary Key or a Foreign Key is assigned to each table. For each nested attribute or non-atomic attribute in this, we create additional relations. Repeated data categories were eliminated as a result. A relation is said to be in first normal form if it only satisfies the constraints that contain the primary key.

**Second Normal Form**

According to the Second Normal Form, no non-key attribute should be operationally dependent on any aspect of the primary key when the primary key has numerous qualities. This includes breaking down each partial key into its dependent attributes and generating a new relation for each one. Keep the original primary key and any attributes that are completely dependent on it in your database. This method allows for the removal of data that only needs a small portion of the key. Only when a connection's primary key alone satisfies the requirements for first normal form for the relation's main key and all of its non-primary key properties is it regarded as being in second NF.

**Third Normal Form**

A relation shouldn't have a non-key attribute that is functionally determined by another non-key property or by a collection of non-key characteristics, according to the Third Normal Form. The main key should not be transitively dependent, in other words. The non-key qualities that the deconstructed non-key characteristics functionally define are then organized in respect to those non-key attributes. The goal of this technique is to eliminate any dependencies on the primary key. Only when a relation is in second normal form and, more importantly, when none of its non-key qualities depend on any other non-key traits, is it said to be in third normal form.

## 4.5 TABLE DESIGN

### 1. tbl\_user\_details

Table Description: Table to store user registration details

Primary key : UserId

Foreign key: Null

Table 4.1 User Details

Field	Data Type	Constraints
UserId	Int(10)	Primary key
FirstName	Varchar(50)	Not null
LastName	Varchar(50)	Not null
Email	Text	Not null
Password	Varchar(60)	Not null
Phone	Int(10)	Not null
Gender	Varchar(20)	Not null
profileImage	Text	Not null
LoginTime	Timestamp	Not null
Status	Enum (active,in-active)	Not null
Role	Varchar(30)	Not null

### 2. tbl\_general\_settings

Table Description: Table to store resort details

Primary key : Id

Foreign key: Null

Table 4.2 General settings

Field	Data Type	Constraints
Id	Int(10)	Primary key
Name	Varchar(30)	Not null
Address line1	Varchar(30)	Not null
Address line 2	Varchar (50)	Not null
City	Varchar (30)	Not null
State	Varchar (30)	Not null
Country	Varchar (30)	Not null

Pin code	Int (10)	Not null
Email	Varchar (30)	Not null
Phone	Int (10)	Not null
Telephone	Int(9)	Not null
Description	Varchar (100)	Not null

### 3. tbl\_feedback

Table Description: Table to store feedback details

Primary key : FeedbackId

Foreign key:UserId

Table 4.3 Feedback

Field	Data Type	Constraints
FeedbackId	Int(10)	Primary key
UserId	int(20)	Foreign key
Date	Date	Not null
Message	Varchar(100)	Not null

### 4. tbl\_event\_type

Table Description: Table to store event type details

Primary key : EventTypeId

Foreign key:Null

Table 4.4 Event Type

Field	Data Type	Constraints
EventTypeId	Int(10)	Primary key
EventType	Varchar (30)	Not null
EventImage	Varchar(100)	Not null
Description	Varchar(100)	Not null
Cost	Double	Not null
Status	Enum(active, in-active)	Not null

### 5. tbl\_event\_list

Table Description: Table to store event list details

Primary key : EventId

---

Foreign key:EventTypeId

Table 4.5 Event List

Field	Data Type	Constraints
EventId	Int(10)	Primary key
EventTypeId	Int(10)	Foreign key
HallNo.	Int (10)	Not null
Status	Enum(active, in-active)	Not null
BookingStatus	Enum(booked,available)	Not null

## 6. tbl\_event\_book

Table Description: Table to store event book details

Primary key : BookingId

Foreign key: EventId and UserId

Table 4.6 Event booking

Field	Data Type	Constraints
BookingId	Int(10)	Primary key
EventId	Int(10)	Foreign key
UserId	Int(10)	Foreign key
BookedDate	Current Timestamp	Not null
EventDate	Date	Not null
NoOfGuest	Varchar(20)	Not null
HallNo.	Int(10)	Not null
EventTime	Time	Not null
TotalHours	Int(11)	Not null
Amount	Double	Not null
Email	Varchar(50)	Not null
Phone	Int(10)	Not null
Status	Enum(cancel,paid)	Not null

## 7. tbl\_event\_payment

Table Description: Table to store event payment details

Primary key : PaymentId

Foreign key: BookingId and UserId

Table 4.7 Event payment

Field	Data Type	Constraints
PaymentId	Int(10)	Primary key
BookingId	Int(10)	Foreign key
UserId	Int(10)	Foreign key
PaymentDate	Date	Not null
Amount	Double	Not null
Status	Enum(paid)	Not null

## 8. tbl\_food\_type

Table Description: Table to store food type details

Primary key : FoodTypeId

Foreign key: Null

Table 4.8 Food Type

Field	Data Type	Constraints
FoodTypeId	Int(10)	Primary key
FoodType	Varchar (30)	Not null
FoodImage	Text	Not null
Description	Varchar(100)	Not null
Cost	Double	Not null
Status	Enum(active, in-active)	Not null

## 9. tbl\_food\_booking

Table Description: Table to store food booking details

Primary key : BookingId

Foreign key: FoodTypeId and UserId

Table 4.9 Food booking

Field	Data Type	Constraints
BookingId	Int(10)	Primary key
FoodTypeId	Int(10)	Foreign key
UserId	Int(10)	Foreign key
BookedDate	Date CurrentTimestamp	Not null
Date	Date	Not null

Time	Time	Not null
Package	Varchar (50)	Not null
NoOfPackages	Int(10)	Not null
Amount	Double	Not null
Email	Varchar(50)	Not null
Phone	Int(10)	Not null
Status	Enum(cancel,paid)	Not null

## 10. tbl\_food\_payment

Table Description: Table to store food paymentdetails

Primary key : PaymentId

Foreign key: BookingId and UserId

Table 4.10 Food payment

Field	Data Type	Constraints
PaymentId	Int(10)	Primary key
BookingId	Int(10)	Foreign key
UserId	Int(10)	Foreign key
PaymentDate	Date	Not null
Amount	Double	Not null
Status	Enum(paid)	Not null

## 11. tbl\_room\_type

Table Description: Table to store room type details

Primary key : RoomTypeId

Foreign key:Null

Table 4.11 Room Type

Field	Data Type	Constraints
RoomTypeId	Int(10)	Primary key
RoomType	Varchar (30)	Not null
RoomImage	Text	Not null
Description	Varchar(100)	Not null
NoOfPersons	Int(10)	Not null
Cost	Double	Not null
Status	Enum(active, in-active)	Not null

**12. tbl\_room\_list**

Table Description: Table to store room list details

Primary key : RoomId

Foreign key: RoomTypeId

Table 4.12 Room List

Field	Data Type	Constraints
RoomId	Int(10)	Primary key
RoomTypeId	Int(10)	Foreign key
RoomNo	Int(10)	Not null
Status	Enum(active,in-active)	Not null
BookingStatus	Enum(booked, cancel)	Not null

**13. tbl\_room\_book**

Table Description: Table to store room book details

Primary key : BookingId

Foreign key: RoomId and UserId

Table 4.13 Room booking

Field	Data Type	Constraints
BookingId	Int(10)	Primary key
RoomId	Int(10)	Foreign key
UserId	Int(10)	Foreign key
BookedDate	Date Current timestamp	Not null
CheckIn	Date	Not null
CheckOut	Date	Not null
NoOfGuest	Int(10)	Not null
RoomNo.	Int(10)	Not null
Amount	Double	Not null
Email	Varchar(50)	Not null
Phone	Int(10)	Not null
Status	Enum(cancel,paid)	Not null



**14. tbl\_room\_payment**

Table Description: Table to store room payment details

Primary key : PaymentId

Foreign key: BookingId and UserId

Table 4.14 Room payment

Field	Data Type	Constraints
PaymentId	Int(10)	Primary key
BookingId	Int(10)	Foreign key
UserId	Int(10)	Foreign key
PaymentDate	Date	Not null
Amount	Double	Not null
Status	Enum(paid)	Not null

**15. tbl\_travel\_package**

Table Description: Table to store travel package details

Primary key :TravelPackageId

Foreign key:Null

Table 4.15 Travel package

Field	Data Type	Constraints
TravelPackageId	Int(10)	Primary key
TravelPackage	Varchar(50)	Not null
TravelImage	Text	Not null
Description	Varchar(100)	Not null
Cost	Double	Not null
Status	Enum(active, in-active)	Not null

**16. tbl\_travel\_book**

Table Description: Table to store travel booking details

Primary key : BookingId

Foreign key: TravelPackageId and UserId

Table 4.16 Travel booking

Field	Data Type	Constraints
BookingId	Int(10)	Primary key
TravelPackageId	Int(10)	Foreign key
UserId	Int(10)	Foreign key
BookedDate	Date Current timestamp	Not null
TravelingDate	Date	Not null
NoOfPersons	Int(10)	Not null
StartingTime	Time	Not null
TotalDuration	Int(10)	Not null
Amount	Double	Not null
Email	Varchar(50)	Not null
Phone	Int(10)	Not null
Status	Enum(cancel,paid)	Not null

## 17. tbl\_travel\_payment

Table Description: Table to store travel payment details

Primary key : PaymentId

Foreign key: BookingId and UserId

Table 4.17 Travel payment

Field	Data Type	Constraints
PaymentId	Int(10)	Primary key
BookingId	Int(10)	Foreign key
UserId	Int(10)	Foreign key
PaymentDate	Date	Not null
Amount	Double	Not null
Status	Enum(paid)	Not null

## **CHAPTER 5**

### **SYSTEM TESTING**

## 5.1 INTRODUCTION

Software testing is the process of closely monitoring the way software is used to see if it functions as intended. Verification and validation are two concepts that are commonly used in conjunction with software testing. A product, including software, is validated by being examined or evaluated to see if it conforms with all pertinent requirements. Software testing, one sort of verification, uses techniques including reviews, analyses, inspections, and walkthroughs as well. Validation is the process of ensuring that what has been specified matches what the user actually wants.

Other procedures that are typically connected to software testing include static analysis and dynamic analysis. Without actually running the code, static analysis examines the software's source code to look for errors and gather statistics. Dynamic analysis examines how software behaves while it is running in order to offer data like execution traces, timing profiles, and test coverage details.

The tasks that make up testing can be planned out in advance and carried out methodically. For computer-based systems, testing begins with individual modules and continues all the way through system integration. Numerous laws may be utilized as testing goals, and testing is required for the system testing goals to be effective. As follows:

Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has high possibility of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error.

A test that effectively accomplishes the aforementioned goals will discover software problems. Additionally, testing demonstrates that the program appears to function in line with the specification and that the performance criteria appear to have been met.

There are three ways to test program.

- For accuracy;
- For effectiveness of execution
- For the difficulty of computing

Test for correctness is supposed to verify that a program does exactly what it was designed to do.

This is much more difficult than it may at first appear, especially for large programs.

## 5.2 TEST PLAN

The procedures that must be followed in order to fulfil various testing techniques are suggested in a test plan. The test plan specifies the activities that must be completed. Software developers are responsible for creating a computer program, as well as any related documentation and data structures. It is always the software developers' job to test each of the program's individual parts to ensure that it serves the intended function. There is an impartial test group in order to address the problems of letting the developer assess what they have created (ITG). The specific objective soft testing need to do the best in terms of numbers. Defect density or frequency of occurrence, cost to detect and correct the problems, remaining test work hours per regression test, and time to failure should all be specified in the test plan.

The levels of testing include:

- Unit testing
- Integration Testing
- Data validation Testing
- Output Testing

### 5.2.1 Unit Testing

The smallest unit of software design, the software component or module, is the focus of unit testing, which focuses verification effort on it. For the purpose of verifying critical control routes and identifying flaws inside the module's boundary, the component level design description is utilised as a guide. the specified untested area for unit testing and the test complexity level. Unit testing may be carried out simultaneously for several components and is white-box oriented. The modular interface is examined to ensure that data enters and departs the software unit under test in a proper manner. The local data structure is reviewed to ensure that data temporarily stored maintains its integrity during each step of an algorithm's execution. Boundary conditions are assessed to confirm that each statement in a module has been executed. Finally, each method of error management is looked at. Testing of data flow through a module interface are important before beginning any additional tests. If data cannot correctly enter and exit the system, all other tests are useless. The unit test's selective analysis of execution routes is a crucial task. Good design must account for problem scenarios and build up error handling systems that may cleanly redirect or terminate. processing when an error does occur. Boundary testing is the last task of unit testing step.

In the Sell-Soft System, unit testing was carried out by considering each module as a distinct entity and subjecting them to a variety of test inputs. The internal logic of the modules had certain issues, which were fixed. Each module is tested and run separately after development. All unused code was eliminated, and it was confirmed that every module was functional and produced the desired outcome.

### **5.2.2 Integration Testing**

Integration testing is a methodical approach for creating the program's structure while also carrying out tests to find interface issues. The goal is to construct a program structure that has been determined by design using unit tested components. The software as a whole is tested. Correction is challenging since the size of the overall program makes it hard to isolate the reasons. As soon as these mistakes are fixed, new ones arise, and the process repeats itself in an apparently unending cycle. All modules were incorporated into the system once unit testing was completed in order to check for any interface consistency issues. A distinctive program structure also developed when discrepancies in program structures were eliminated.

### **5.2.3 Validation Testing or System Testing**

This marks the conclusion of the testing procedure. This required comprehensive testing of the whole system, including all forms, codes, modules, and class modules. System tests and black box testing are two common names for this kind of testing.

The primary focus of the black box testing technique is on the program's functional requirements. A software engineer may, for instance, use Black Box testing to generate sets of input conditions that would exhaustively test each program requirement. Erroneous or missing functions, interface flaws, data structure or external data access errors, performance defects, startup blunders, and termination faults are the kinds of issues that black box testing focuses on.

### **5.2.4 Output Testing or User Acceptance Testing**

The system under consideration is tested for user acceptance; in this case, it must satisfy the business' requirements. The software should consult the user and the perspective system while it is being developed in order to make any necessary adjustments. This was done in relation to the following things:

5.2.4.1 Input Screen Designs,

5.2.4.2 Output Screen Designs,

A variety of test data are used to conduct the aforementioned tests. The process of system testing requires the preparation of test data. After the test data preparation, the system under investigation is tested using the test data. The system's flaws are once more discovered during testing, repaired with the help of the aforementioned techniques, and recorded for future use.

### **Automation Testing**

Automatic tests are performed on software and other computer products to ensure they adhere to strict standards. It basically serves as a test to make sure the hardware or software works exactly as intended. It looks for mistakes, faults, and any other issues that might arise during the course of the product's construction. Automation testing can be done at any time of day. By means of predefined sequences, the software is examined. The information is then summarized, and this data can be compared to the outcomes of earlier test runs.

### **Benefits of Automation Testing**

Detailed reporting capabilities - Test cases for different scenarios are carefully built for automation testing. These planned sequences can cover a lot of ground and produce in-depth reports that are simply impossible for a human to produce.

Improved bug detection - Finding bugs and other flaws in a product is one of the key reasons to test it. This procedure can be made simpler with automation testing. Additionally, it has a greater test coverage analysis capability than people might have.

- Simplifies testing - Most SaaS and IT organizations routinely include testing in their daily operations. The key is to keep things as basic as you can. Automation has a lot of advantages. The test scripts can be reused when automating test tools.
- Quickens the testing procedure - Machines and automated technology operate more quickly than people. This is why we employ them, along with increased accuracy. Your software development cycles are subsequently shortened by this.

Reduces human intervention - Tests can be run at any time of day, even overnight, without needing humans to oversee it. Plus, when it's conducted automatically, this can also reduce the risk of human error

### **5.2.5 Selenium Testing**

Selenium is an open-source application that automates web browsers. You can write test scripts using a single interface in a variety of computer languages, including Ruby, Java, NodeJS, PHP, Perl, Python, and C#. The Selenium testing tool automates the cross-browser compatibility testing of web applications. It is used to ensure that web apps are of a high level, whether they are responsive, progressive, or standard. Selenium is a free piece of software.



## Test cases for a Login Page

Project Name: Resort Management system (TravelOne)					
Login Test Case					
Test Case ID: Fun_1			Test Designed By: Annapoorneswari D		
Test Priority (Low/Medium/High): High			Test Designed Date:19-07-2022		
Module Name: Login Screen			Test Executed By: Dr. Bijimol T K		
Test Title: Verify login with valid username and password			Test Execution Date: 19-07-2022		
Description: Test the Login Page					
Pre-Condition: User has valid username and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status (Pass/Fail)
1	Navigation to Login Page		Login Page should be display ed	Login page displayed	Pass
2	Provide Valid username	Username : annaammu2017@gmail.com	User should d be able to Login	User Logged in and navigated to User Dashboard	Pass
3	Provide Valid Password	Password:Ammu2625			
4	Click on Sign In button				
5	Provide Invalid username or password	Username:annaammu2017@gmail.com  Password: User12345	User should not be able to Login	Message for enter valid email id or password displayed	Pass
6	Provide Null username or Password	Username : null Password: null			
7	Click on Sign In button				

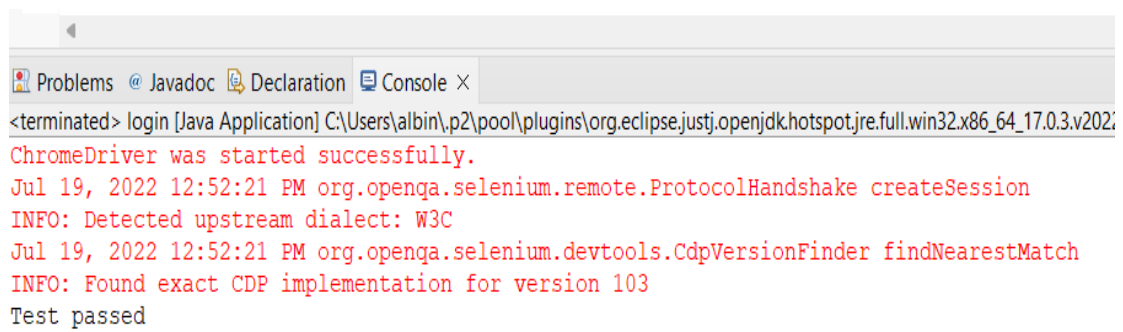
Table 5.2.1 Testing

**Post-Condition:** User is validated with database and successfully login into account. TheAccount session details are logged in database.

### Code package

```
package testing1;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
public class login {
public static void main(String[] args) {
System.setProperty("webdriver.chrome.driver","C:\\Users\\albin\\Downloads\\chromedriver.exe" );
WebDriver driver=new ChromeDriver();

driver.get("http://localhost/Main%20Project/Resort%20management%20system/login.php");
driver.findElement(By.id("uname")).sendKeys("annaammu2017@gmail.com");
driver.findElement(By.id("password")).sendKeys("Ammu2625");
driver.findElement(By.id("login")).click();
String
actualUrl="http://localhost/Main%20Project/Resort%20management%20system/customer/index.php?";
String expectedUrl= driver.getCurrentUrl();
if(actualUrl.equalsIgnoreCase(expectedUrl)) {
System.out.println("Test passed");
} else {
System.out.println("Test failed");
}
}
}
```



```
<terminated> login [Java Application] C:\Users\albin\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.3.v20220719\jre\bin\java.exe
ChromeDriver was started successfully.
Jul 19, 2022 12:52:21 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected upstream dialect: W3C
Jul 19, 2022 12:52:21 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
INFO: Found exact CDP implementation for version 103
Test passed
```

Figure 5.2.1 Testing Result

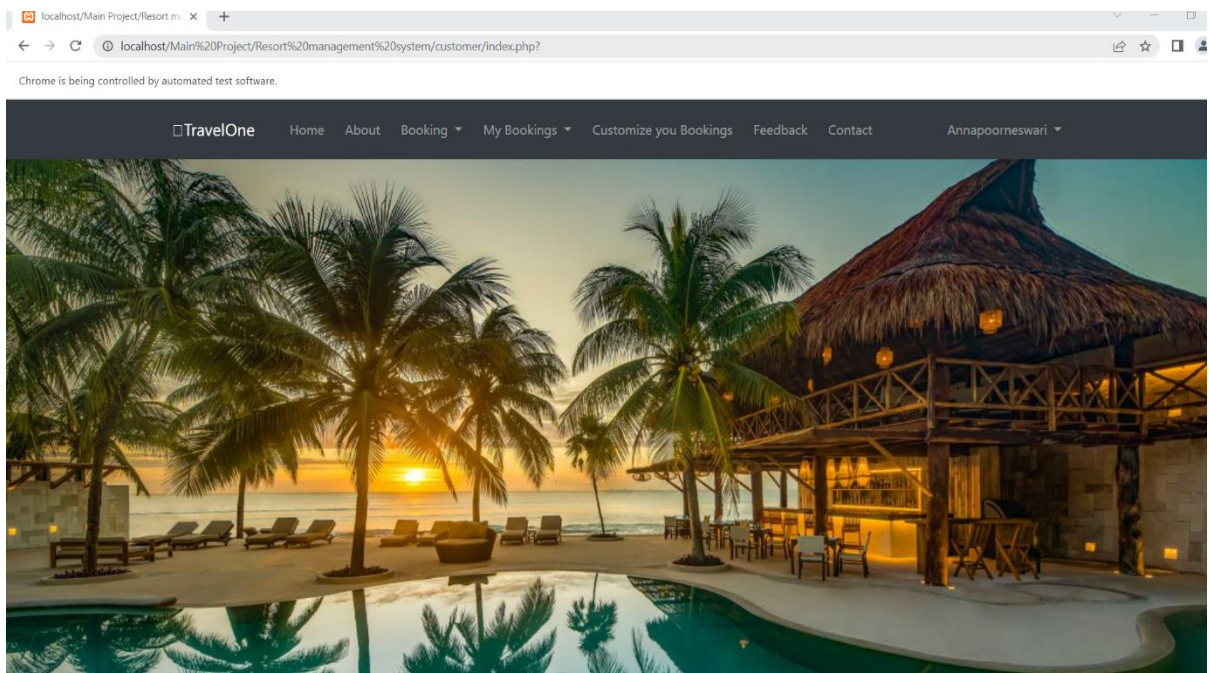


Figure 5.2.2 Testing Result

## **CHAPTER 6**

# **IMPLEMENTATION**

## 6.1 INTRODUCTION

The conceptual design is developed into a usable system during the project's execution phase. Since it assures users that the system will function as planned and be dependable and accurate, it may be viewed as the most crucial stage in developing a successful new system. Its primary concerns are training and user documentation. Conversion often occurs either during or following user training. Bringing a new system design into operation is referred to as implementation. Implementation is the process of making a new, updated system design a stand-alone operation. According to Haviland, the user department is currently responsible for the bulk of the work and has the most impact on the system. Confusion and mayhem may come from a poorly thought-out or managed implementation. The entire process of moving from the old system to the new one is referred to as implementation. The new system might replace a current human or automated system, be completely different, or be improved upon. To meet organizational needs, a dependable system must be appropriately deployed. System implementation describes the process of putting the created system into use. This covers all steps required to switch from the old to the new system. only in the event that thorough testing reveals the system to be in good working order. Employees of the system evaluate the system's feasibility. The complexity of the system being implemented will affect how much effort is required for system analysis and design in order to implement the three essential components of education and training, system testing, and changeover.

The implementation state involves the following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of methods to achieve the changeover.

## 6.2 IMPLEMENTATION PROCEDURES

The full installation of the package in the intended setting, as well as the system's usability and accomplishment of the planned applications, are all referred to as software implementation. It's common for someone who won't use the program to commission the development effort. People first have reservations about the program, yet it's crucial to prevent opposition from growing because:

- The active user must be aware of the benefits of using the new system.
- Their confidence in the software is built-up.
- Proper guidance is imparted to the user so that he is comfortable in using the application.

Before examining the system, the user must be aware that the server software has to be running on the server in order to access the results. The real procedure won't happen if the server object is not active and functioning on the server.

### **6.2.1 User Training**

To prepare the user to test and alter the system is the goal of user training. The participants must have confidence in their capacity to advance the objective and reap the benefits of the computer-based system. As systems get more complex, training becomes increasingly important. Through user training, the user learns how to enter data, respond to error messages, query the database, call up routines that will create reports, and carry out other crucial activities.

### **6.2.2 Training on the Application Software**

The user must first obtain the fundamental training in computer literacy, following which they must be taught how to operate the new application software. In addition to how the screens function, what sort of help is shown on them, what types of errors are created while entering data, how each input is checked, and how to update the data that was entered, this will explain the core concepts of how to use the new system. The information needed by the specific user or group to run the system or a particular component of the system should therefore be covered throughout the program's training on the application. This training may change based on the user group and organizational level.

### **6.2.3 System Maintenance**

The mystery of system development is maintenance. The riddle of system development is maintenance. A software product is actively functioning while it is in the maintenance stage of its lifespan. After being successfully implemented, a system has to be properly maintained. System maintenance is a crucial phase in the software development life cycle. Maintenance is necessary for a system to be adaptable to changes in the system environment.

Software maintenance is of course, far more than "Finding Mistakes".

## **CHAPTER 7**

### **CONCLUSION AND FUTURE SCOPE**

## **7.1 CONCLUSION**

In this project, users will be able to access the portal according to their preferences and availability while still receiving a variety of amenities with less effort. The activity focuses on the administration and support of the many adjustments that may be made in the various system components. At the centre of the database, it primarily handles resort management. The approach offers details on the many alternatives that are available and their availability in various circumstances. The database also keeps track of the atomic data for the many units that may be found inside a resort.

## **7.2 FUTURE SCOPE**

Expanding the project in different location around the globe. Transforming this to the mobile application for better portability and making it more reliable.



## **CHAPTER 8**

### **BIBLIOGRAPHY**

---

**REFERENCES:**

- Gary B. Shelly, Harry J. Rosenblatt, “*System Analysis and Design*”,2009.
- Roger S Pressman, “*Software Engineering*”,1994.
- PankajJalote, “*Software engineering: a precise approach*”,2006.
- James lee and Brent ware Addison, “Open source web development with LAMP”, 2003
- IEEE Std 1016 Recommended Practice for Software Design Descriptions.

**WEBSITES:**

- [www.w3schools.com](http://www.w3schools.com)
- [www.jquery.com](http://www.jquery.com)
- <http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf>
- <https://www.bing.com/search?q=php+tutorial&cvid=ecf6e80c06744639a6bf650f62a97f0d&aqs=edge.3.69i57j0l5j69i60l3.4955j0j1&pglt=43&FORM=ANNTA1&PC=U531#>

## **CHAPTER 9**

## **APPENDIX**

## 9.1 Sample Code

### room.php

```

<?php include('include/header.php') ;
include('../include/dbConnect.php');
?>
<!DOCTYPE html>
<html lang="en">
<head>
<body>
<div class="container">
    <a class="navbar-brand "><b><h1>Available Rooms</h1></a></b>
</div>
<div class="row product-categorie-box">

    <div class="tab-content">
        <div role="tabpanel" class="tab-pane fade show active" id="grid-view">
            <div class="row">
                <?php
                    $result = mysqli_query($con,"SELECT * FROM room_type INNER JOIN
room_list ON room_type.RoomTypeId=room_list.RoomTypeId where
Booking_status='Available'");
                    while ($row = mysqli_fetch_array($result)){
                        ?>
                        <div class="col-sm-6 col-md-6 col-lg-4 col-xl-4">
                            <form method="post" enctype="multipart/form-data" action="">
                                <div class="products-single fix">
                                    <div class="box-img-hover">
                                        <center>
                                            
                                        </center>
                                    </div>
                                    <div class="why-text">
                                        <h4 align="center"><?php echo $raw['RoomType']; ?></h4>
                                        <h4 align="center">Rs.<?php echo $raw['Cost']; ?></h4>
                                        <h6 align="center"><?php echo $raw['Description']; ?></h6>
                                        <h6 align="center">Maximum No Of Persons:<?php echo
$raw['No.of persons']; ?></h6>
                                        <h6 align="center"><?php echo $raw['Booking_status']; ?></h6>
                                        <input type="hidden" name="productid" value="<?php echo
$raw['RoomTypeId']; ?>">
                                        <a class="nav-link" href="roombooking.php?id=<?php echo
$raw['RoomTypeId']; ?>" aria-selected="true" align="center">Book Now</a>
                                    </div>
                                </div><br><br>
                            </form>
                        </div>
                    }
                </div>
            </div>
        </div>
    </div>

```

---

```

        <?php } ?>
    </div>
</div>

</div>
</div>

</body>
</html>

```

### roombooking.php

```

<?php include('include/header.php') ;
include('../include/dbConnect.php');
$roomid=$_GET['id'];
$userid= $_SESSION['loggedUserId'];
$sql=mysqli_query($con,"SELECT * FROM `room_type` WHERE `RoomTypeId`='$roomid'");

$rows=mysqli_fetch_array($sql);
$amount=$rows['Cost'];
$sql1=mysqli_query($con, "SELECT * FROM `room_list` WHERE `RoomTypeId`='$roomid' and
`Booking_status`='Available' ");
$rows1=mysqli_fetch_array($sql1);
$roomno=$rows1['RoomNumber'];

if(isset($_POST['bookRoom'])) {
    $checkin= mysqli_real_escape_string($con, $_POST['checkIn']);
    $checkout= mysqli_real_escape_string($con, $_POST['checkOut']);
    $noofguest= mysqli_real_escape_string($con, $_POST['no_of_guest']);
    $email= mysqli_real_escape_string($con, $_POST['email']);
    $phn= mysqli_real_escape_string($con, $_POST['contactno']);

    $room=mysqli_query($con,"INSERT INTO `room_book`(`RoomId`,`User_id`,`CheckIn`,
`CheckOut`,`NoOfGuest`,`Room_Number`,`Email`,`Phone_number`,`Status`)
VALUES ('$roomid','$userid','$checkin','$checkout','$noofguest','$roomno',
$email','$phn','Booked')");
    $z=mysqli_insert_id($con);
    $sqli=mysqli_query($con,"UPDATE `room_list` SET `Booking_status`='Booked' WHERE
`RoomNumber`='$roomno' ");

    if($room)
    {
        echo "<script>alert('Successfully Booked');window.location.href='roomcash.php?id=$roomid
&& cost=$amount && bid=$z';</script>";
    }
}

```

---

```

}

}

?>

<div class="container">
  <div class="row align-items-center my-5">
    <!-- For Demo Purpose -->
    <div class="col-md-5 pr-lg-5 mb-5 mb-md-0">
      

    </div>

    <!-- Booking Form -->
    <div class="col-md-7 col-lg-6 ml-auto">
      <form action="" method="POST" enctype="multipart/form-data" autocomplete="off">
        <div class="row">
          <div class="container mb-4">
            <h2 class="text-center">Make Your Booking</h2>

          </div>

          <input type="hidden" name="roomTypeId" />

          <!--roomType-->
          <div class="form-group col-lg-6 mb-4">

            <div class="ml-2">
              <label for="roomType">Room Type</label>
            </div>
            <div class="input-group ">
              <div class="input-group-prepend">
                <span class="input-group-text bg-white px-4 border-md border-right-0">

                </span>
              </div>
              <input id="roomType" type="text" name="roomType" value="{?php echo
$rows['RoomType'];?>" class="form-control bg-white border-left-0 border-md" required readonly>
            </div>
          </div>
          <div class="form-group col-lg-6 mb-4">

```

---

```

    <div class="ml-2">
      <label for="roomType">Room Number</label>
    </div>
    <div class="input-group ">
      <div class="input-group-prepend">
        <span class="input-group-text bg-white px-4 border-md border-right-0">

          </span>
        </div>
        <input id="roomType" type="text" name="roomType" value="<?php echo
$rows1['RoomNumber'];?>" class="form-control bg-white border-left-0 border-md" required
readonly>
      </div>
    </div>

    <!-- roomCost -->
    <div class="form-group col-lg-6 mb-4">

      <div class="ml-2">
        <label for="roomCost">Cost of Room /per-day</label>
      </div>
      <div class="input-group ">
        <div class="input-group-prepend">
          <span class="input-group-text bg-white px-4 border-md border-right-0">
            <i class="fa fa-inr"></i>
          </span>
        </div>
        <input id="roomCost" type="text" name="roomCost" value="<?php echo
$rows1['Cost'];?>" class="form-control bg-white border-left-0 border-md" required readonly>
      </div>
    </div>

    <!-- Email Address -->
    <div class="form-group col-lg-12 mb-4">

      <div class="ml-2">
        <label for="email">Enter Email Id</label>
      </div>
      <div class="input-group ">
        <div class="input-group-prepend">
          <span class="input-group-text bg-white px-4 border-md border-right-0" >
            <i class="fa fa-envelope text-muted"></i>
          </span>
        </div>
        <input id="email" type="email" name="email" placeholder="Email Address"
class="form-control bg-white border-left-0 border-md" required>
      </div>
    </div>

    <!-- Phone Number -->
    <div class="form-group col-lg-12 mb-4">

```

---

---

```

    <div class="ml-2">
      <label for="phoneNumber">Enter Phone Number</label>
    </div>
    <div class="input-group ">
      <div class="input-group-prepend">
        <span class="input-group-text bg-white px-4 border-md border-right-0">
          <i class="fa fa-phone-square text-muted"></i>
        </span>
      </div>

      <input id="contactno" type="tel" name="contactno" pattern="[789][0-9]{9}"
placeholder="Phone Number" class="form-control bg-white border-md border-left-0 pl-3"
required>
    </div>
  </div>

  <!-- number of guest -->
  <div class="form-group col-lg-12 mb-4">

    <div class="ml-2">
      <label for="no_of_guest">Number of Guest</label>
    </div>
    <div class="input-group ">
      <div class="input-group-prepend">
        <span class="input-group-text bg-white px-4 border-md border-right-0">
          <i class="fa fa-black-tie text-muted"></i>
        </span>
      </div>
      <select id="no_of_guest" name="no_of_guest" class="form-control custom-select
bg-white border-left-0 border-md" required>
        <option value="" selected="true" disabled="true">Choose number of
Guests</option>
        <option value="1">1</option>
        <option value="2">2</option>
        <option value="3">3</option>
        <option value="4">4</option>
      </select>
    </div>
  </div>

  <!--checkin -->
  <div class="form-group col-lg-6 mb-4">

    <div class="ml-2">
      <label for="checkIn">Check-In Date</label>
    </div>
    <div class="input-group ">
      <div class="input-group-prepend">
        <span class="input-group-text bg-white px-4 border-md border-right-0">

```

---



---

```

        <i class="fa fa-calendar" aria-hidden="true"></i>
      </span>
    </div>
    <input id="checkIn" type="date" name="checkIn" placeholder="Check-In Data"
class="form-control bg-white " required>
  </div>
</div>

<!--checkOut-->
<div class="form-group col-lg-6 mb-4">
  <div class="ml-2">
    <label for="checkOut">Check-Out Date</label>
  </div>
  <div class="input-group ">
    <div class="input-group-prepend">
      <span class="input-group-text bg-white px-4 border-md border-right-0">
        <i class="fa fa-calendar" aria-hidden="true"></i>
      </span>
    </div>

    <input id="checkOut" type="date" name="checkOut" placeholder="Check-Out
Data" class="form-control bg-white " required>
  </div>
</div>

<div class="form-group col-lg-12 mx-auto mb-0">
  <button type="submit" class="btn btn-primary btn-block py-2" name="bookRoom"
>
    <span class="font-weight-bold">Book</span>
  </button>
</div>

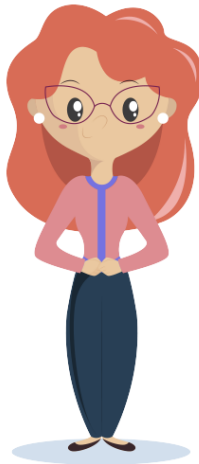
</div>
</form>
</div>
</div>
</div>

```

---

## 9.2 ScreenShots

### User Login



Sign in

Email

Password


Sign in

or

Don't have an account? [Sign up here!](#)

Figure 9.1 User login

### User Registration



Choose Picture

First Name

Last Name

Email Address

Phone Number

Choose your Gender

Password

Confirm Password

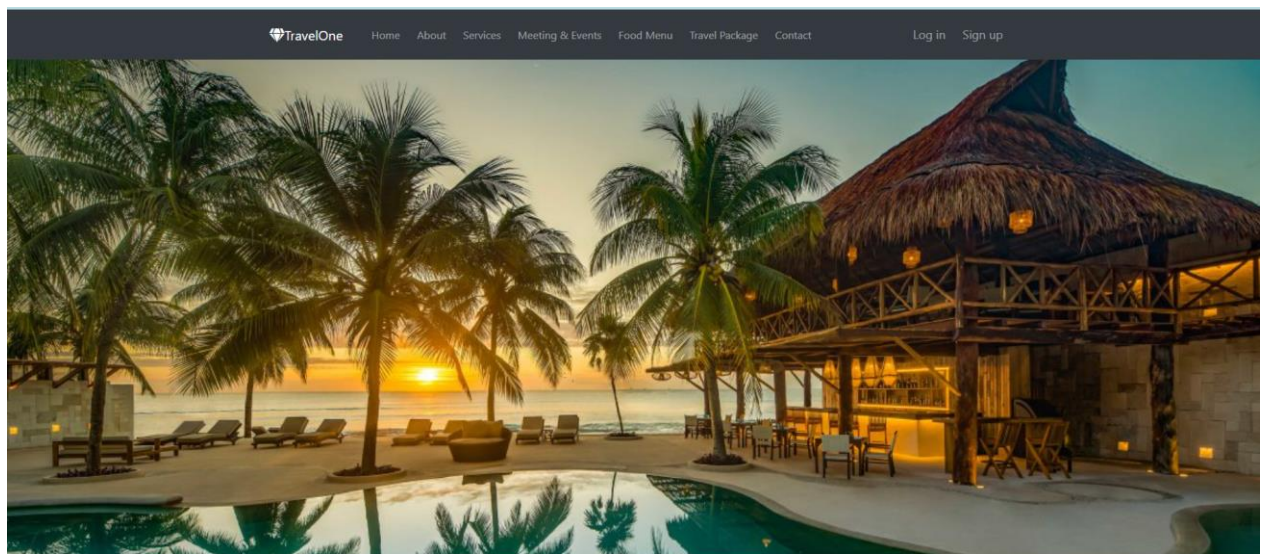
Create your account

OR

Already Registered? [Login](#)

Figure 9.2 User registration

## Home Page



TravelOne, Kochi.

Figure 9.3 Home Page

## About Us

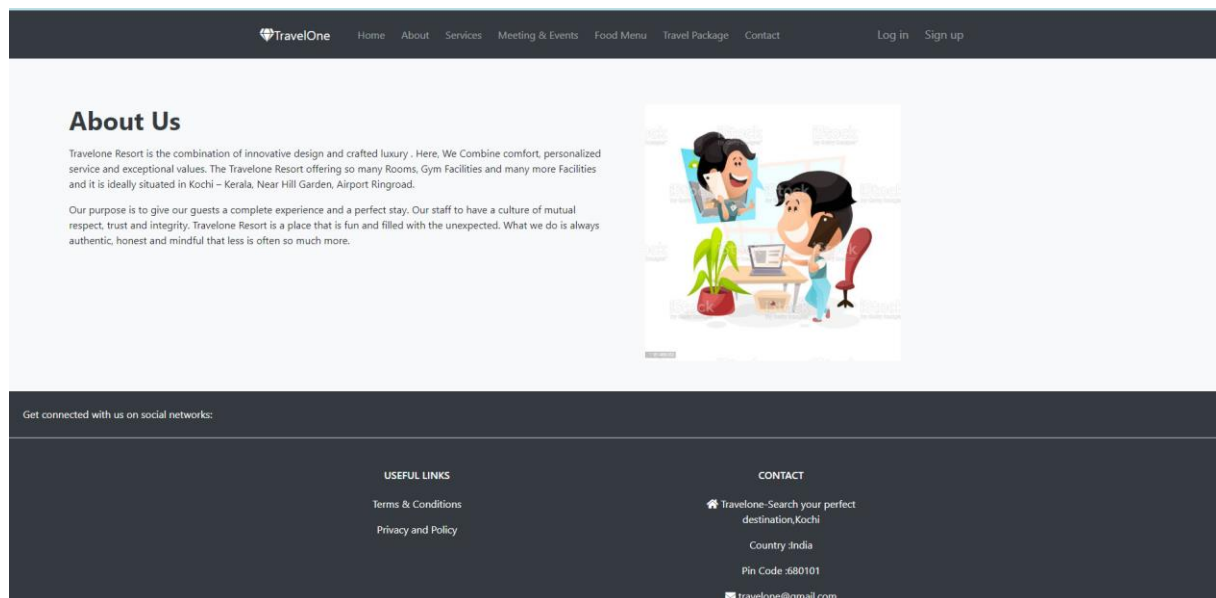


Figure 9.4 About us

## Contact Us

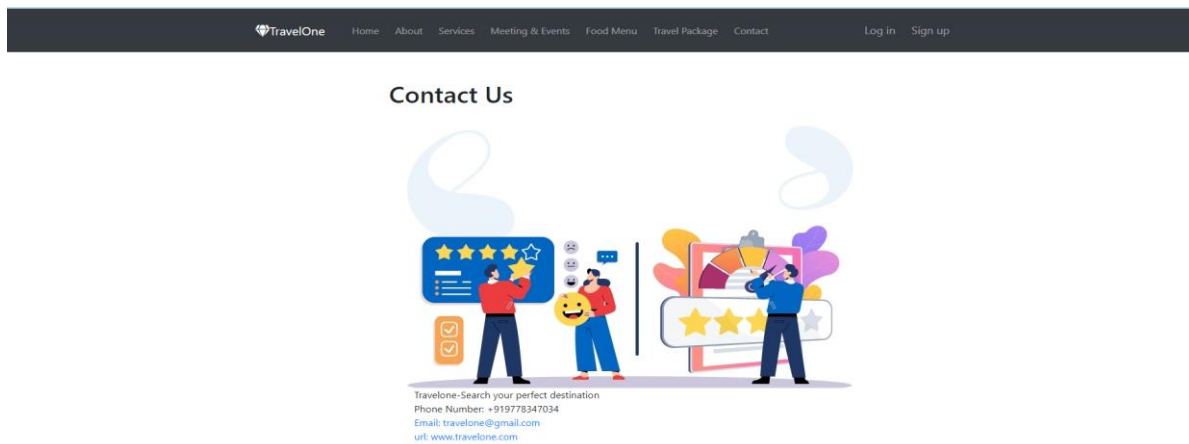
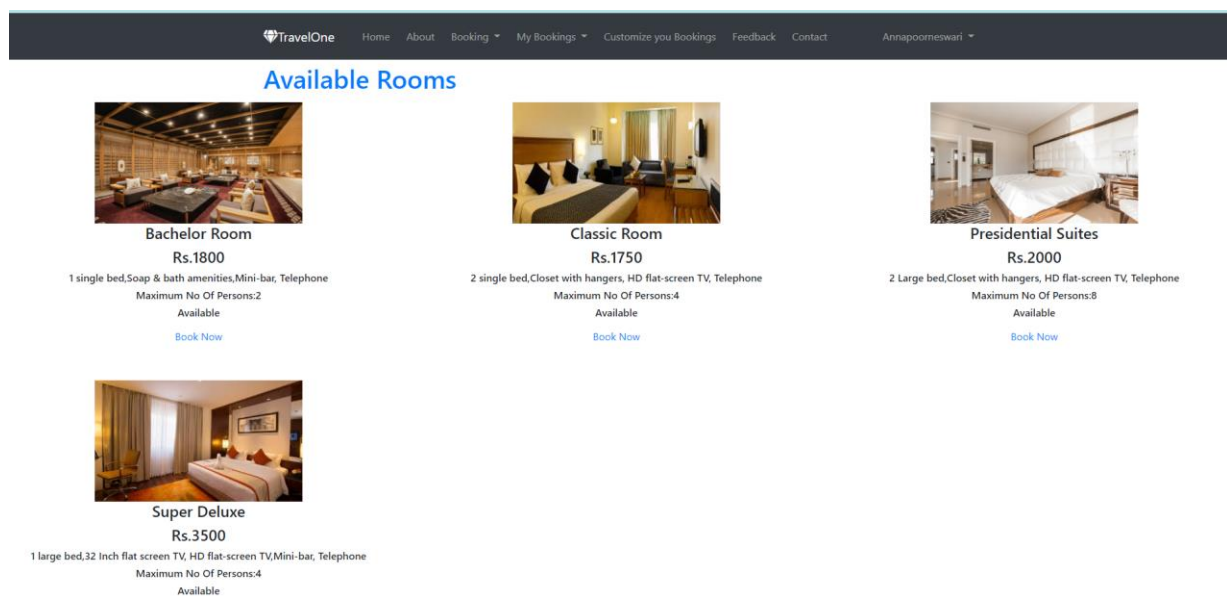


Figure 9.5 Contact us

## Available Rooms

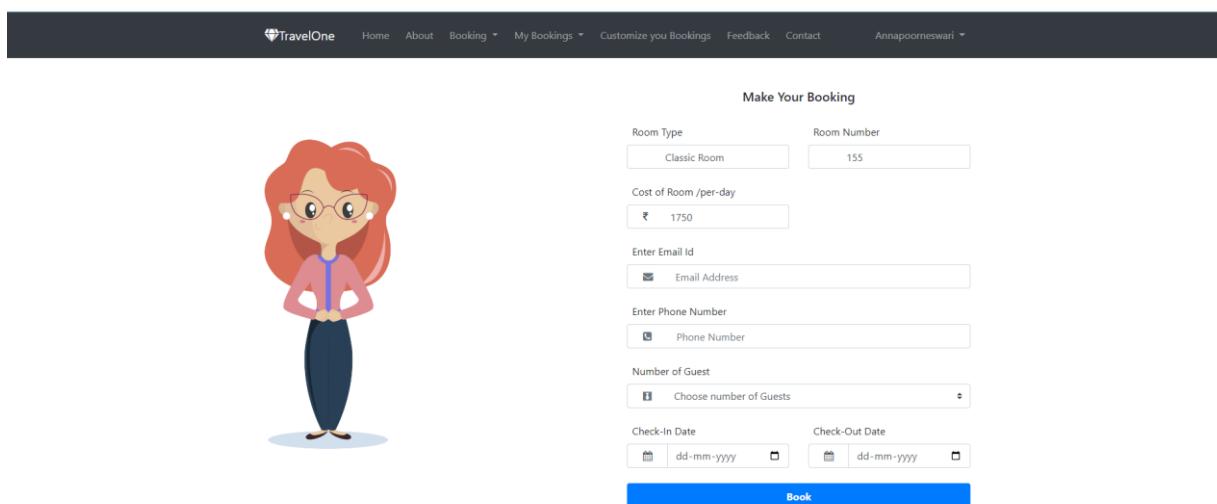


**Available Rooms**

Room Type	Price (Rs.)	Bed Configuration	Amenities	Maximum No Of Persons	Status	Action
Bachelor Room	Rs.1800	1 single bed	Soap & bath amenities, Mini-bar, Telephone	2	Available	<a href="#">Book Now</a>
Classic Room	Rs.1750	2 single bed	Closet with hangers, HD flat-screen TV, Telephone	4	Available	<a href="#">Book Now</a>
Presidential Suites	Rs.2000	2 Large bed	Closet with hangers, HD flat-screen TV, Telephone	8	Available	<a href="#">Book Now</a>
Super Deluxe	Rs.3500	1 large bed	32 Inch flat screen TV, HD flat-screen TV, Mini-bar, Telephone	4	Available	

Figure 9.6 Available rooms

## Room Booking



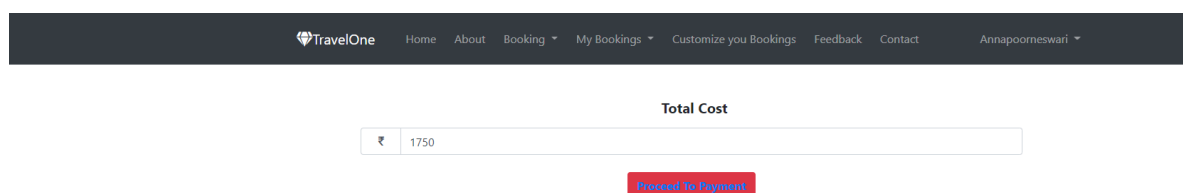
The screenshot shows the 'Make Your Booking' form on the TravelOne website. On the left is a cartoon illustration of a woman with red hair and glasses. The form fields are as follows:

- Room Type:** Classic Room
- Room Number:** 155
- Cost of Room /per-day:** ₹ 1750
- Enter Email Id:** Email Address
- Enter Phone Number:** Phone Number
- Number of Guest:** Choose number of Guests
- Check-In Date:** dd-mm-yyyy
- Check-Out Date:** dd-mm-yyyy

A blue 'Book' button is at the bottom right of the form.

Figure 9.7 Room booking

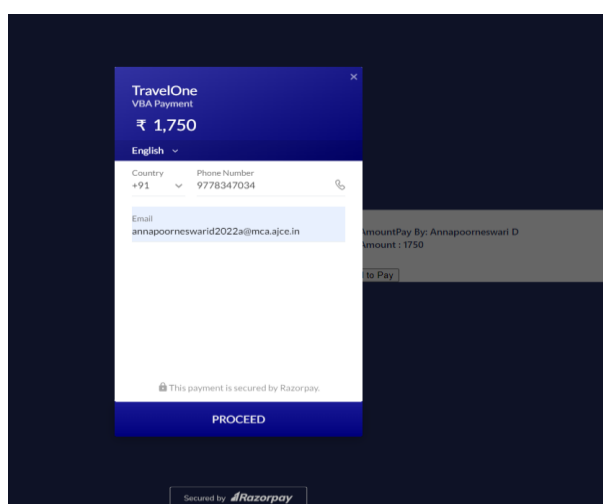
## Cost Calculation



The screenshot shows the 'Total Cost' section of the TravelOne website. It features a text input field with the value '₹ 1750' and a red 'Proceed to Payment' button below it.

Figure 9.8 cost calculation

## Payment



The screenshot shows a Razorpay payment modal for TravelOne. The modal displays the following information:

- TravelOne VISA Payment**
- ₹ 1,750**
- English** (language dropdown)
- Country:** +91
- Phone Number:** 9778347034
- Email:** annapoorneswarid2022a@mca.ajce.in
- Amount Pay By:** Annapoorneswari D
- Amount:** 1750

At the bottom, there is a 'PROCEED' button and a note: 'This payment is secured by Razorpay.' The modal is secured by Razorpay.

Figure 9.9 Payment

## Booking Details


<div>  <a href="#">Home</a> <a href="#">About</a> <a href="#">Booking</a> <a href="#">My Bookings</a> <a href="#">Customize you Bookings</a> <a href="#">Feedback</a> <a href="#">Contact</a> <a href="#">Annapoorneswari</a> </div>						
Room Bookings						
Sl no.	Room Type	Date	Room No	Amount	Status	Details
1	Family Room	2022-07-04	101	4500	Cancelled	<a href="#">Cancel</a>
2	Bachelor Room	2022-07-05	110	16200	Booked	<a href="#">Cancel</a>
3	Family Room	2022-07-06	101	4500	Booked	<a href="#">Cancel</a>
4	Family Room	2022-07-06	102	4500	Booked	<a href="#">Cancel</a>
5	Family Room	2022-07-06	0	4500	Booked	<a href="#">Cancel</a>
6	Family Room	2022-07-06	0	3000	Booked	<a href="#">Cancel</a>
7	Classic Room	2022-07-06	125	5250	Booked	<a href="#">Cancel</a>
8	Luxury	2022-07-06	121	7000	Cancelled	<a href="#">Cancel</a>
9	Luxury	2022-07-06	150	10500	Cancelled	<a href="#">Cancel</a>

Figure 9.10 Booking details

## User Profile








<div>  <a href="#">Home</a> <a href="#">About</a> <a href="#">Booking</a> <a href="#">My Bookings</a> <a href="#">Customize you Bookings</a> <a href="#">Feedback</a> <a href="#">Contact</a> <a href="#">Annapoorneswari</a> </div>	
Account Details	
<div>  <div>Choose Picture</div> </div>	
First Name	Last Name
 Annapoorneswari	 D
Email	
 annaammu2017@gmail.com	
Contact No	
 9778347034	
Gender	
 female	
<a href="#">Save Changes</a>	

Figure 9.11 User profile

## Admin Dashboard

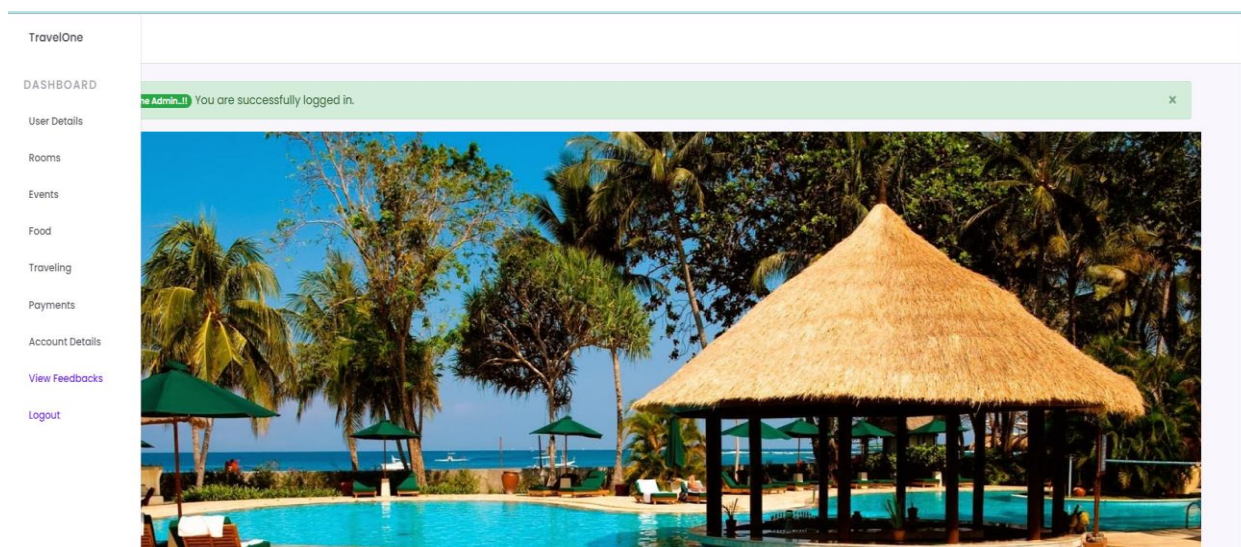


Figure 9.12 Admin dashboard

## User Details

TravelOne										
Users Details										
Sl no.	First Name	Last Name	Email	Password	Phone No	Gender	Image	Status	Action	Registered Date/Time
1	Bensy	Benny	bensy@gmail.com	bensy12	7896541236	female		active	<a href="#">Delete</a>	2022-07-02 18:14:22
2	Sujith	K S	sujith12@gmail.com	Sujith12	8596321478	male		active	<a href="#">Delete</a>	2022-07-02 18:14:22
3	Arya	Sasi	arya@gmail.com	Arya12	7896523698	female		active	<a href="#">Delete</a>	2022-07-02 18:14:22
4	Haritha	Krishnan	hari@gmail.com	Haril23	7896325412	female		active	<a href="#">Delete</a>	2022-07-02 18:14:22
5	Annapoorneswari	D	annaammu2017@gmail.com	Ammu2625	9778347034	female		active	<a href="#">Delete</a>	2022-07-04 08:56:13
6	Amal	Dath	amaldathad@gmail.com	Amal12	9947268768	male		active	<a href="#">Delete</a>	2022-07-06 06:19:01

Figure 9.13 User details

## Room Types

TravelOne

### Room Type

[Add New Room Type](#)

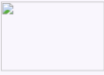




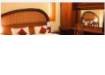
Sl no.	Room Type	Image	Description	No. of Persons	Cost	Status	Action
1	Family Room		1 Large bed,32 inch flat screen TV, Kitchen facilities,Towels,Dining tables	4	Rs.2e17	active	<a href="#">Edit</a> <a href="#">Delete</a> <a href="#">Status</a>
2	Bachelor Room		1 single bed,Soap & bath amenities,Mini-bar, Telephone	2	Rs.1800	active	<a href="#">Edit</a> <a href="#">Delete</a> <a href="#">Status</a>
3	Presidential Suites		2 Large bed,Closet with hangers, HD flat-screen TV, Telephone	8	Rs.2000	active	<a href="#">Edit</a> <a href="#">Delete</a> <a href="#">Status</a>
4	Classic Room		2 single bed,Closet with hangers, HD flat-screen TV, Telephone	4	Rs.1750	active	<a href="#">Edit</a> <a href="#">Delete</a> <a href="#">Status</a>
5	Club Room		1 large bed,Closet with hangers, 24 Hour room service,Computer and internet access	4	Rs.1680	active	<a href="#">Edit</a> <a href="#">Delete</a> <a href="#">Status</a>
6	Deluxe Room		1 large bed,Closet with hangers, 24 Hour room service,Computer and internet access	4	Rs.1900	active	<a href="#">Edit</a> <a href="#">Delete</a> <a href="#">Status</a>

Figure 9.14 Room type

## Room List

TravelOne

### Rooms

Sl no.	Room Type	Cost	Room No	Status	Booking Status	Action
1	Family Room	2e17	101	active	Booked	<a href="#">BookingStatus</a> <a href="#">Delete</a> <a href="#">Status</a>
2	Family Room	2e17	102	active	Booked	<a href="#">BookingStatus</a> <a href="#">Delete</a> <a href="#">Status</a>
3	Bachelor Room	1800	110	active	Booked	<a href="#">BookingStatus</a> <a href="#">Delete</a> <a href="#">Status</a>
4	Presidential Suites	2000	120	active	Booked	<a href="#">BookingStatus</a> <a href="#">Delete</a> <a href="#">Status</a>
5	Classic Room	1750	125	active	Booked	<a href="#">BookingStatus</a> <a href="#">Delete</a> <a href="#">Status</a>
6	Deluxe Room	1900	149	active	Booked	<a href="#">BookingStatus</a> <a href="#">Delete</a> <a href="#">Status</a>
7	Luxury	3500	121	active	Booked	<a href="#">BookingStatus</a> <a href="#">Delete</a> <a href="#">Status</a>
8	Bachelor Room	1800	111	active	Booked	<a href="#">BookingStatus</a> <a href="#">Delete</a> <a href="#">Status</a>
9	Bachelor Room	1800	115	active	Available	<a href="#">BookingStatus</a> <a href="#">Delete</a> <a href="#">Status</a>

Figure 9.15 Room list



## Room Booking details

TravelOne

### Room Bookings

Sl no.	User	Date	Room Type	Room No	Check In	Check Out	Amount	Status	Details
1	Annapoomeswari D	2022-07-04	Family Room	101	2022-07-05	2022-07-08	4500	Cancelled	<a href="#">Bill</a>
2	Annapoomeswari D	2022-07-05	Bachelor Room	110	2022-07-21	2022-07-30	16200	Booked	<a href="#">Bill</a>
3	Amal Dath	2022-07-06	Bachelor Room	110	2022-07-07	2022-07-09	3600	Booked	<a href="#">Bill</a>
4	Annapoomeswari D	2022-07-06	Family Room	101	2022-07-07	2022-07-10	4500	Booked	<a href="#">Bill</a>

Figure 9.16 Room booking

## Feedbacks

TravelOne

### Foodbacks

Sl no.	Name	Date	message	Action
1	Bensy	2022-06-28 09:52:54	good	<a href="#">Delete</a>
2	Annapoomeswari	2022-07-05 14:34:01	Great	<a href="#">Delete</a>
3	Amal	2022-07-06 06:25:44	Good atmosphere and nature friendly	<a href="#">Delete</a>
4	Gopala	2022-07-08 11:56:07	nice	<a href="#">Delete</a>

Figure 9.17 Feedbacks

## Account details

TravelOne

### Company Details

Company Name

Address Line 1

Address Line 2

City

State

Country

Pin Code

Description

### Contact Details

Email

Phone Number

Telephone Number

[Save Changes](#)

Figure 9.18 Account details