A

Project Report

On

**ONLINE HOTEL BOOKING SYSTEM**

Developed by

**GOHIL KAPILBHAI S. (CE015)**

**CHUDASAMA AKSHAYRAJSINH (CE007)**

**Department of CE, DD University**

**Guided By:**

**Prof. Ankit P. Vaishnav**



**Department of Computer Engineering**

**Faculty of Technology, Dharmsinh Desai University**

**College Road, Nadiad-387001**



**Department of Computer Engineering**

**Faculty of Technology, Dharmsinh Desai University**

**CERTIFICATE**

This is to certify that the practical/term work carried out in the subject of **Web Service Development** and recorded in this journal is the bonafide work of **Gohil Kapilbhai, ID No: 22CEUOD002 , Chudasama Akshayrajsinh, ID No: 21CEUOG108**  of B.Tech semester **VI** in the branch of Computer Engineering during the academic year **2023-2024.**

Prof. Ankit P. Vaishnav Dr. C.K. Bhensdadia

Assistant Professor, Head,

Dept. of Computer Engineering, Dept. of Computer Engineering,

Faculty of Technology, Faculty of Technology,

Dharmsinh Desai University, Dharmsinh Desai University,

Nadiad Nadiad

# List Of Figures

[Figure 3.1 Use-Case Diagram 10](#_Toc163047792)

[Figure 4.1 Class Diagram 11](#_Toc163047793)

[Figure 5.1 E-R Diagram 12](#_Toc163047794)

[Figure 8.1 Login Page UI 25](#_Toc163047795)

[Figure 8.2 Register Page UI 25](#_Toc163047796)

[Figure 8.3 Home Page UI 26](#_Toc163047797)

[Figure 8.4 Search For Rooms UI 26](#_Toc163047798)

[Figure 8.5 Available Rooms UI 27](#_Toc163047799)

[Figure 8.6 Select Rooms UI 27](#_Toc163047800)

[Figure 8.7 Booking Details UI 28](#_Toc163047801)

[Figure 8.8 Booking Form UI 28](#_Toc163047802)

[Figure 8.9 Reservation UI 29](#_Toc163047803)

[Figure 8.10 Bill PDF 29](#_Toc163047804)

[Figure 8.11 Find Booking UI 30](#_Toc163047805)

# List Of Tables

[Table 5.1 User Data Dictionary 13](#_Toc163047806)

[Table 5.2 Room Data Dictionary 13](#_Toc163047807)

[Table 5.3 Reservation Data Dictionary 13](#_Toc163047808)

[Table 5.4 EachRoom Data Dictionary 14](#_Toc163047809)

[Table 5.5 Reservation-EachRoom Data Dictionary 14](#_Toc163047810)

[Table 7.1 Test Suite 24](#_Toc163047811)

# Table of Contents

[List Of Figures i](#_Toc163047812)

[List Of Tables ii](#_Toc163047813)

[Table of Contents iii](#_Toc163047814)

[1 Introduction 1](#_Toc163047815)

[1.1 Purpose 1](#_Toc163047816)

[1.2 Intended Audience 1](#_Toc163047817)

[1.3 Project Scope 1](#_Toc163047818)

[2 Software Requirement Specifications 3](#_Toc163047819)

[2.1 Product Perspective 3](#_Toc163047820)

[2.2 Product Functions 3](#_Toc163047821)

[2.3 User Classes and Characteristics 3](#_Toc163047822)

[2.3.1 Admin 4](#_Toc163047823)

[2.3.2 Customers 4](#_Toc163047824)

[2.4 Operating Environment 4](#_Toc163047825)

[2.5 Design And Implementation Constraints 4](#_Toc163047826)

[2.6 Assumptions And Dependencies 5](#_Toc163047827)

[2.7 External Interface Requirements 5](#_Toc163047828)

[2.7.1 User Interfaces 5](#_Toc163047829)

[2.7.2 Hardware Interface 5](#_Toc163047830)

[2.7.3 Software Interfaces 5](#_Toc163047831)

[2.7.4 Communication Interfaces 5](#_Toc163047832)

[2.8 System Features 6](#_Toc163047833)

[2.8.1 Functional Requirenments 6](#_Toc163047834)

[2.9 Other Nonfunctional Requirements 8](#_Toc163047835)

[2.9.1 Performance Requirements 8](#_Toc163047836)

[2.9.2 Safety Requirements 8](#_Toc163047837)

[2.9.3 Software Quality Attributes 8](#_Toc163047838)

[2.9.4 Goal Of Implementation 9](#_Toc163047839)

[3 Use-Case Diagram 10](#_Toc163047840)

[4 Class Diagram 11](#_Toc163047841)

[4.1 Class Diagram 11](#_Toc163047842)

[5 Database Schema 12](#_Toc163047843)

[5.1 E-R Diagram 12](#_Toc163047844)

[5.2 Data Dictionary 13](#_Toc163047845)

[5.2.1 User 13](#_Toc163047846)

[5.2.2 Room 13](#_Toc163047847)

[5.2.3 Reservation 13](#_Toc163047848)

[5.2.4 EachRoom: 14](#_Toc163047849)

[5.2.5 ReservationEachRoom: 14](#_Toc163047850)

[6 Implementation Details 15](#_Toc163047851)

[6.1 Web Services (Interfaces, Contracts) 15](#_Toc163047852)

[6.1.1 Iuserservice.Cs 15](#_Toc163047853)

[6.1.2 Iroomservice.Cs 15](#_Toc163047854)

[6.1.3 Ireservationservice.Cs 16](#_Toc163047855)

[6.2 Host (Form1.Cs) 16](#_Toc163047856)

[6.3 Client (App.Config) 19](#_Toc163047857)

[7 Testing 21](#_Toc163047858)

[7.1 Testing Methodology Used 21](#_Toc163047859)

[7.1.1 BlackBox Testing 21](#_Toc163047860)

[7.1.2 Advantages of Black-Box Testing 21](#_Toc163047861)

[7.2 Test Suite 21](#_Toc163047862)

[8 User Interface 25](#_Toc163047863)

[8.1 Login Page 25](#_Toc163047864)

[8.2 Register Page 25](#_Toc163047865)

[8.3 Home Page 26](#_Toc163047866)

[8.4 Search for Room Availability 26](#_Toc163047867)

[8.5 Available Rooms 27](#_Toc163047868)

[8.6 Select Rooms 27](#_Toc163047869)

[8.7 Booking Details 28](#_Toc163047870)

[8.8 Booking Form 28](#_Toc163047871)

[8.9 Reservation 29](#_Toc163047872)

[8.10 Bill Pdf 29](#_Toc163047873)

[8.11 Find Booking 30](#_Toc163047874)

[9 Conclusion 31](#_Toc163047875)

[10 Limitation and Future Extension 32](#_Toc163047876)

[10.1 Limitations 32](#_Toc163047877)

[10.2 Future Extension 32](#_Toc163047878)

[11 Bibliography 33](#_Toc163047879)

[11.1 Books or Some reading content 33](#_Toc163047880)

[11.2 Web 33](#_Toc163047881)

# Introduction

## Purpose

* The software should be useful for the hotel manager to manage the hotel in online mode and also for the growing business.
* The purpose of the online hotel booking system is to allow the customers to self-book the hotel rooms.
* The software should be used for securely storing and managing the data.
* Through the help of the software the customer can do emergency booking.
* This software saves the time from both sides (Hotel and Customer side).

## Intended Audience

* This system is intended for providing room facility in online mode so that the peoples having the online banking or online payment options can do booking in online mode.

## Project Scope

* The online hotel booking system project is intended for the online reservation of the rooms.
* It will be able to do various operation of the hotel booking automatic and in online mode like bill generation, room availability, data management etc.
* There are two user levels in hotel booking system: Customer, and Hotel Manager
* This software is consist of the booking management system and DBMS server.
* Customer will be able to check for room’s availability, select the rooms, and pay for the room.
* Manager will able to update room information such as cost and category.
* Hotel booking system is able to resolve the drawbacks of the reservation of room in offline mode.
* In future updates, software should be able to perform more tasks automatically like financial report generation, growth chart of the system, no of customers visited, shows the regular customer etc.

**1.4 SOFTWARE INTERFACES:**

* Front end technologies: HTML, CSS, JAVASCRIPT
* Back-end language: c#
* Development Tool: Visual Studio 2022
* Database server: MySQL
* Framework: .Net framework
* Architecture: Service based architecture

# Software Requirement Specifications

## Product Perspective

* The “Online Hotel Booking System” is a web application used for the book the room using internet via website.
* There manual method for the room booking has so many constrain and issues like more time Consumption, delay in processing, maintaining data (requires all the records of booking, recourses etc. Are maintained in books) etc.
* Hence sometime it is very difficult to book the room in emergency situation
* The purpose of this project is to develop and implement an online hotel booking system for hotels, that will replace the manual method of booking for hotel rooms.
* We are creating application that will used for the processing the online room booking.
* With the help of this web application hotel manager should be able to maintain the hotel with less time and also maintain and store data of the customer and hotel in proper manner without any time-consuming process.

## Product Functions

1. Registration
2. log in
3. display the available rooms
4. Book a room
5. Update Room Details
6. Manage Booking Details
7. Generate bill
8. Store the details of the user in database

## User Classes and Characteristics

There are 2 user classes in our Hotel Booking System:

1.Admin

2.Customer

### Admin

* Admin have every access to the hotel system.
* Admin is responsible for managing hotel resources and staffs.
* Admin can view information report such as customer information, booking information, and room information.

### Customers

* Customers are vital part of the system.
* Customer have access to view the vacant room information and price range.
* They should be able to confirm the booking and cancel it if necessary.
* Other than this customer can do the following actions:
* Registration
* Login
* Search room availability
* See the price of rooms and room details
* Booking
* Logout

## Operating Environment

* The online hotel booking system is web application so it will operate in all browsers like Google, Chrome, Microsoft Edge, Firefox etc.
* Operating environment for this system will be hosted virtually using any cloud service provider.

## Design And Implementation Constraints

* Internet connection is a constraint for this system because system is available from cloud therefore customer needs to have good network connection to connect to our web interface.

## Assumptions And Dependencies

* It is assumed that system will be developed using .net framework with My SQL database will work perfectly.
* If incase of any difficulties, SRS should be flexible enough to change accordingly.

## External Interface Requirements

### User Interfaces

* The user interface for system shall be compatible to any type of web browser such as Mozilla Firefox, Google Chrome, and Internet Explorer.

### Hardware Interface

#### Server side

* Monitor: resolution of 1024\*768 or above
* Processor: intel or AMD 2GHZ
* Ram: 4 GB or above
* Disk space: 10GB or above

#### Client side

* Monitor: resolution of 1024\*768 or above
* Processor: intel or AMD 1GHZ
* Ram: 512 MB or above
* Disk space: 2GB or above

### Software Interfaces

* Front end technologies: HTML, CSS, JAVASCRIPT
* Back-end language: c#
* Development Tool: Visual Studio 2022
* Database server: MySQL

### Communication Interfaces

* The System shall be using HTTP/HTTPS Or TCP/IP for communication over Internet

## System Features

### Functional Requirenments

* Our system has following general requirements:

1. MANAGING ROOM DETAILS
2. REGISTRATION
3. LOGIN
4. VIEW THE ROOM DETAILS:
5. ROOM BOOKING
6. **Managing Room Details**

**Description:** Update the room details.

* 1. **View available rooms**

**Input:** select view room details option

**Output:** rooms are displayed

* 1. **Update room details**

**Input:** New details for room provided

**Output:** changes made will be affected in room details

1. **Registration**

**Description:** Users can register themselves

* 1. **Select register option**

**Input:** Register option selected

**Output:** User will be asked to enter Full Name, Phone Number, Email, Gender, password and confirm password

* 1. **Provide details for registration**

**Input:** details provided

**Output:** User will be registered

**Processing:** If user has not already registered then registration process will be done.

1. **Login**

**Description:** User and admin can login to the system

* 1. **Select login option**

**Input:** Login option selected

**Output:** User or Admin will be asked to enter username and password

**Processing:** If user has already registered him/herself then only he/she will be logged in to the system

1. **View The Room Details**

**Description:** User can check the details about every room before booking

* 1. **Select book now option**

**Input:** view rooms option selected

**Output:** shows the details about rooms

* 1. **Select particular room to view the facilities**

**Input:** room selected

**Output:** details of the room displayed

1. **Room Booking**

**Description:** user can book room and make payment

* 1. **Select the room**

**Input:** room selected

**Output:** User prompted to provide details

* 1. **Provide the details**

**Input:** Details provided

**Output:** User prompted with payment option

* 1. **Select payment option.**

**Input:** Payment option selected.

**Output:** requested method will be provided to pay the bill

* 1. **bill generation**

**Input:** select bill generate option

**Output:** Bill will be generated

## Other Nonfunctional Requirements

### Performance Requirements

**NF1.**Results for cross checking of availability of rooms in internal database and customer’s choice shall be in 5 seconds.

**NF2.**Web page UI load time should within 3 seconds.

**NF3.**Redirection page load time should be within 3 seconds.

**NF4.**Data in database should be updated within 3 seconds.

**NF5.**Data in database should be updated within 2 seconds

### Safety Requirements

**NF6.** Customer’s personal details shall be encrypted.

### Software Quality Attributes

* **Correctness:** This system should satisfy the normal regular Hotel Management operations precisely to fulfil the end user objectives
* **Efficiency:** Enough resources to be implemented to achieve the particular task efficiently without any hassle.
* **Flexibility:** System should be flexible enough to provide space to add new features and to handle them conveniently.
* **Integrity:** System should focus on the customer information and avoid data losses as much as possible
* **Portability:** The system should run in any Microsoft windows environment.
* **Maintainability:** The system should be maintainable.
* **Testability**: The system should be able to be tested to confirm the performance and client’s specifications.

### Goal Of Implementation

In the future we are implementing the following function into our system.

* Alert and notification
* Multiple payment gateway
* Guest history
* Easy retrieved and storage of data and information
* Daily count of guest that arrive and departure from hotel
* Daily weekly and monthly calculation of guest booked, arrived and departed from hotel
* Improved data management efficiency

# Use-Case Diagram

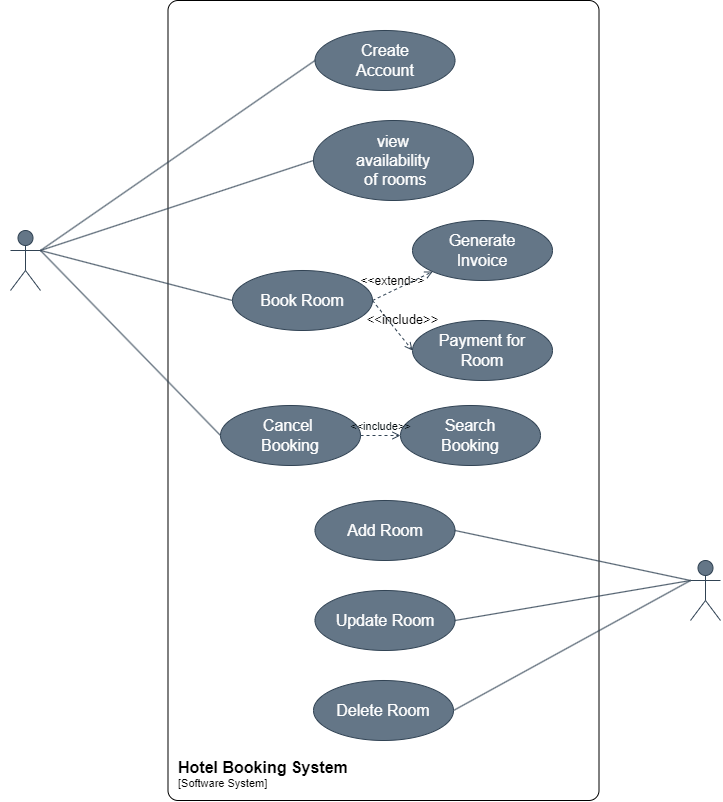
****

Figure 3.1 Use-Case Diagram

# Class Diagram

## Class Diagram

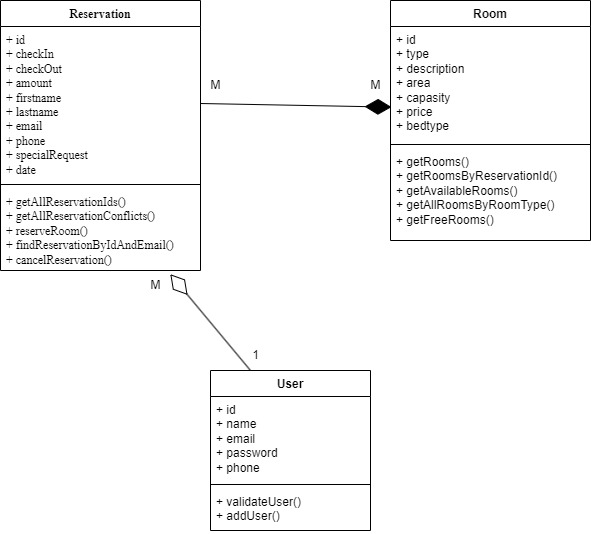
****

Figure 4.1 Class Diagram

# Database Schema

## E-R Diagram

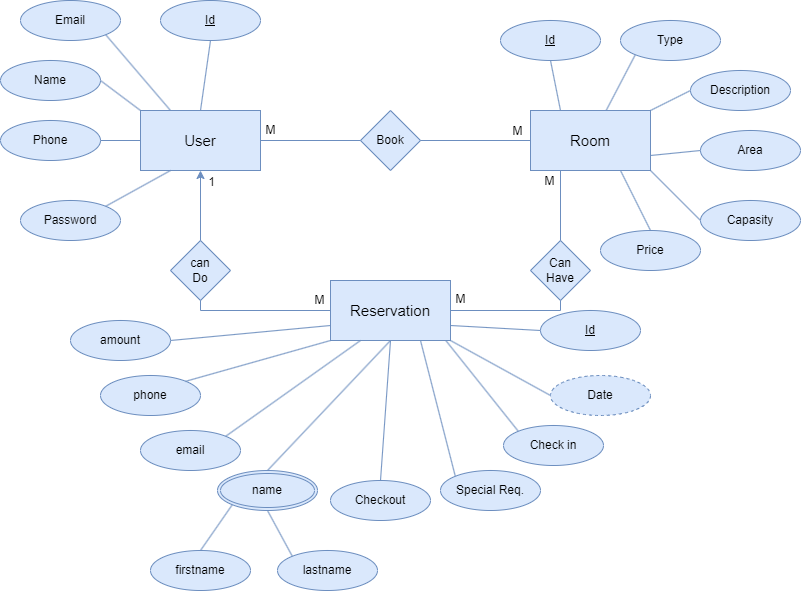


Figure 5.1 E-R Diagram

## Data Dictionary

### User

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr no | Fieldname | Datatype | Field Length | constraints |
| 1. | Id | Int | - | PK |
| 2. | Email | Varchar | 50 | Not-Null |
| 3. | Name | Varchar | 50 | Not-Null |
| 4. | Phone | Varchar | 50 | Not-Null |
| 5. | Password | Varchar | 50 | Not-Null |

Table 5.1 User Data Dictionary

### Room

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr no | Fieldname | Datatype | Field Length | constraints |
| 1. | Id | Int | - | PK |
| 2. | Type | Varchar | 50 | Not-Null |
| 3. | Desc | Varchar | 500 | Not-Null |
| 4. | Area | Varchar | 50 | Not-Null |
| 5. | Capacity | Numeric | (1,0) | Not-Null |
| 6. | Bed Type | Varchar | 50 | Not-Null |
| 7. | Price | Numeric | (18,0) | Not-Null |

Table 5.2 Room Data Dictionary

### Reservation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr no | Fieldname | Datatype | Field Length | constraints |
| 1. | Id | Int | - | PK |
| 2. | Checkin | Date | - | Not-Null |
| 3. | Checkout | Date | - | Not-Null |
| 4. | Amount | Int | - | Not-Null |
| 5. | Firstname | Varchar | 50 | Not-Null |
| 6. | Lastname | Varchar | 50 | Not-Null |
| 7. | Email | Varchar | 50 | Not-Null |
| 8. | Phone | Varchar | 50 | Not-Null |
| 9. | SpecialRequest | Varchar | 500 | - |
| 10. | Date | DateTime | - | Not-Null |

Table 5.3 Reservation Data Dictionary

### EachRoom:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr no | Fieldname | Datatype | Field Length | constraints |
| 1. | Id | Int | - | PK |
| 2. | RoomTypeId | Int | 50 | FK(Room.Id) |

Table 5.4 EachRoom Data Dictionary

### ReservationEachRoom:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr no | Fieldname | Datatype | Field Length | constraints |
| 1. | Id | Int | - | PK |
| 2. | ReservationId | Int | - | FK(Reservation.Id) |
| 3. | EachRoomId | Int | - | FK(EachRoom.Id) |
| 4. | Adult | Int | - | Null |
| 5. | Children | Int | - | Null |
| 6. | Price | nchar | 10 | Not-Null |

Table 5.5 Reservation-EachRoom Data Dictionary

# Implementation Details

## Web Services (Interfaces, Contracts)

### Iuserservice.Cs

namespace HotelService

{

[ServiceContract]

public interface IUserService

{

[OperationContract]

User ValidateUser(string email, string pass);

[OperationContract]

bool AddUser(User u);

}

}

### Iroomservice.Cs

namespace HotelService

{

[ServiceContract]

public interface IRoomService

{

[OperationContract]

List<Room> GetRooms();

[OperationContract]

List<RoomData> getRoomsByReservationId(int resId);

[OperationContract]

List<Room> getAvailabeRooms(int noOfRooms, int minCapasity, List<int> ConflictedResIds);

[OperationContract]

List<int> getAllEachRoomsByReservationId(int rid);

[OperationContract]

List<int> getAllRoomsByRoomTypeId(int roomTypeId);

[OperationContract]

int getFreeRoomIdByRoomHeading(string heading);

}

}

### Ireservationservice.Cs

namespace HotelService

{

[ServiceContract]

public interface IReservationService

{

[OperationContract]

bool cancelReservation(int resId);

[OperationContract]

ReservationFull FindReservationByIdAndEmail(int id, string email);

[OperationContract]

int BookRoom(FinalStorageData dt);

[OperationContract]

List<int> getAllReservationConflicts(DateTime checkin, DateTime checkout);

[OperationContract]

List<Reservation> GetALlReservationIds();

}

}

## Host (Form1.Cs)

namespace HotelHost

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

ServiceHost sh = null;

ServiceHost sh2 = null;

ServiceHost sh3 = null;

private void button1\_Click(object sender, EventArgs e)

{

if (button1.Text=="Stop")

{

sh.Close();

sh2.Close();

sh3.Close();

button1.Text = "Start";

label1.Text = "Services are not Running";

}

else

{

string baseAddress = "http://localhost/HotelBookingServices/";

Uri httpUri1 = new Uri(baseAddress+"RoomService");

Uri httpUri2 = new Uri(baseAddress + "ReservationService");

Uri httpUri3 = new Uri(baseAddress + "UserService");

sh = new ServiceHost(typeof(HotelService.RoomService), httpUri1);

sh2 = new ServiceHost(typeof(HotelService.ReservationService), httpUri2);

sh3 = new ServiceHost(typeof(HotelService.UserService), httpUri3);

BasicHttpBinding binding1 = new BasicHttpBinding();

BasicHttpBinding binding2 = new BasicHttpBinding();

BasicHttpBinding binding3 = new BasicHttpBinding();

ServiceMetadataBehavior mBehcave1 = new ServiceMetadataBehavior();

ServiceMetadataBehavior mBehcave2 = new ServiceMetadataBehavior();

ServiceMetadataBehavior mBehcave3 = new ServiceMetadataBehavior();

mBehcave1.HttpGetEnabled = true;

mBehcave2.HttpGetEnabled = true;

mBehcave3.HttpGetEnabled = true;

sh.Description.Behaviors.Add(mBehcave1);

sh2.Description.Behaviors.Add(mBehcave2);

sh3.Description.Behaviors.Add(mBehcave3);

sh.AddServiceEndpoint(typeof(IMetadataExchange), MetadataExchangeBindings.CreateMexHttpBinding(), "mex");

sh2.AddServiceEndpoint(typeof(IMetadataExchange), MetadataExchangeBindings.CreateMexHttpBinding(), "mex");

sh3.AddServiceEndpoint(typeof(IMetadataExchange), MetadataExchangeBindings.CreateMexHttpBinding(), "mex");

sh.AddServiceEndpoint(typeof(HotelService.IRoomService), binding1, httpUri1);

sh2.AddServiceEndpoint(typeof(HotelService.IReservationService), binding2, httpUri2);

sh3.AddServiceEndpoint(typeof(HotelService.IUserService), binding3, httpUri3);

sh.Open();

sh2.Open();

sh3.Open();

label1.Text = "Services are Running";

button1.Text = "Stop";

}

}

}

}

## Client (App.Config)

<?xml version="1.0" encoding="utf-8"?>

<!--

For more information on how to configure your ASP.NET application, please visit

https://go.microsoft.com/fwlink/?LinkId=169433

-->

<configuration>

<system.web>

<compilation debug="true" targetFramework="4.7.2" />

<httpRuntime targetFramework="4.7.2" />

</system.web>

<system.codedom>

<compilers>

<compiler language="c#;cs;csharp" extension=".cs" type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider, Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default /nowarn:1659;1699;1701" />

<compiler language="vb;vbs;visualbasic;vbscript" extension=".vb" type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider, Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default /nowarn:41008 /define:\_MYTYPE=\&quot;Web\&quot; /optionInfer+" />

</compilers>

</system.codedom>

<system.serviceModel>

<bindings>

<basicHttpBinding>

<binding name="BasicHttpBinding\_IRoomService" />

<binding name="BasicHttpBinding\_IUserService" />

<binding name="BasicHttpBinding\_IReservationService" />

</basicHttpBinding>

</bindings>

<client>

<endpoint address="http://localhost/HotelBookingServices/RoomService"

binding="basicHttpBinding" bindingConfiguration="BasicHttpBinding\_IRoomService"

contract="RoomService.IRoomService" name="BasicHttpBinding\_IRoomService" />

<endpoint address="http://localhost/HotelBookingServices/UserService"

binding="basicHttpBinding" bindingConfiguration="BasicHttpBinding\_IUserService"

contract="UserService.IUserService" name="BasicHttpBinding\_IUserService" />

<endpoint address="http://localhost/HotelBookingServices/ReservationService"

binding="basicHttpBinding" bindingConfiguration="BasicHttpBinding\_IReservationService"

contract="ReservationService.IReservationService" name="BasicHttpBinding\_IReservationService" />

</client>

</system.serviceModel>

<runtime>

<assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">

<dependentAssembly>

<assemblyIdentity name="System.Runtime.CompilerServices.Unsafe" publicKeyToken="b03f5f7f11d50a3a" culture="neutral" />

<bindingRedirect oldVersion="0.0.0.0-5.0.0.0" newVersion="5.0.0.0" />

</dependentAssembly>

</assemblyBinding>

</runtime>

</configuration>

# Testing

## Testing Methodology Used

### BlackBox Testing

In this section, we'll focus on the testing methodologies utilized in Share-Hub development, with a particular emphasis on black-box testing. Black-box testing is a software testing technique where the internal structure, design, or implementation details of the system are not known to the tester. Instead, the tester interacts with the system's external interfaces and observes its behaviour based on predefined inputs and expected outputs.

### Advantages of Black-Box Testing

**1. Independence from implementation details:** Testers do not need access to the source code or knowledge of the system's internal design, making black-box testing suitable for testing third-party software or components.

**2.Encourages thorough testing:** Black-box testing encourages testers to explore various input combinations and scenarios, ensuring comprehensive test coverage and uncovering potential defects or inconsistencies in the system.

**3.Emphasizes user perspective:** By focusing on inputs and outputs from a user's perspective, black-box testing helps ensure that the system meets user requirements and expectations.

## Test Suite

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test  Case Id | Test Case Objective | Pre  requisite | Steps | Expected  Output | Actual  Output | Status |
|  | Test for the user registration.  (For the user not having account.) | User should not have the account. | 1.Select the create account button.  2.Enter full name, email, phone and password  3.Submit. | User account should be created. | User account created. | PASS |
|  | Test for the user registration.  (For the user having existing account) | User should have account. | 1.Select the create account button.  2.Enter full name, email, phone and password  3.Submit. | The error message should be shown indicating user account already exist. | The error message shown indicating user account already exist. | PASS |
|  | Test for the login with incorrect mail and correct password. | No pre-  requisite | 1.Go to login page  2.Enter invalid email and valid password. | The error message should be shown regarding invalid credentials. | The error message shown indicating invalid credentials. | PASS |
|  | Test for the login with incorrect mail and incorrect password. | No pre-  requisite | 1.Go to login page  2.Enter invalid email and valid password. | The error message should be shown regarding invalid credentials. | The error message shown indicating invalid credentials. | PASS |
|  | Test for the login with correct mail and incorrect password. | No pre-  requisite | 1.Go to login page  2.Enter invalid email and valid password. | The error message should be shown regarding invalid credentials. | The error message shown indicating invalid credentials. | PASS |
|  | Test for the login with correct mail and correct password. | User should already have account. | 1.Go to login page  2.Enter valid email and valid password. | The user should be logged in and redirected to the main page. | Successful login and user redirected to the main page. | PASS |
|  | Test for the booking room in given dates.  (room already booked in given dates) | Room should be booked on give dates. | 1.Click book a stay.  2.Enter check-in, no of adults, no of Children, no of rooms, check-out date.  3.Search | The room should be shown as not available. | The room is not available for booking. | PASS |
|  | Test for the booking room when no of person is more than room capacity. | Room should be not booked but the capacity should be less than the no person entered by user. | 1.Click book a stay.  2.Enter check-in, no of adults, no of Children, no of rooms, check-out date.  3.Search | The room should be shown as not available as the room capacity is less than the peoples. | The room shown as not available. | PASS |
|  | Test for the booking room.  (room criteria matches) | Room should not be booked in given dates. | 1.Click book a stay.  2.Enter check-in, no of adults, no of Children, no of rooms, check-out date.  3.Search | The room should be shown for the reservation. | Room shown for the registration. | PASS |
|  | Test for booking room with the invalid number entered.  (number digit is less than 10) | User should have already selected the room and now filling booking form. | 1.Don’ fill all the fields.  2. Submit. | The error message should be shown indicating the field is required. | The error message shown indicating the field is required. | PASS |
|  | Test for booking room with the invalid number entered.  (number digit is less than 10) | User should have already selected the room and now filling booking form. | 1.Enter invalid number.  2. Submit. | The error message should be shown indicating the invalid mobile number. | The error message shown indicating the invalid mobile number. | PASS |
|  | Test for the print functionality. | User should have already booked the room. | 1.Click on print button. | User should be able to download the pdf of the bill generated. | User is able to view and download the pdf file for the reservation. | PASS |
|  | Test for the cancel reservation. | User should have already booked the room. | 1.Click on the cancel reservation button. | The booking should be cancelled and room should be available for the booking in given dates. | The booking cancelled and room is available for the booking in given dates. | PASS |
|  | Test for the searching room with wrong booking id. | User should have already booked the room. | 1.Click on manage room details.  2.Enter the invalid booking id and submit. | The error message should be shown indicating the invalid credentials. | The error message shown indicating the invalid credentials. | PASS |
|  | Test for searching room with valid email and booking id. | User should have already booked the room. | 1.Click on manage room details.  2.Enter the valid email and booking id and submit. | The booking details should be shown. | The booking details opened. | PASS |

Table 7.1 Test Suite

# User Interface

## Login Page

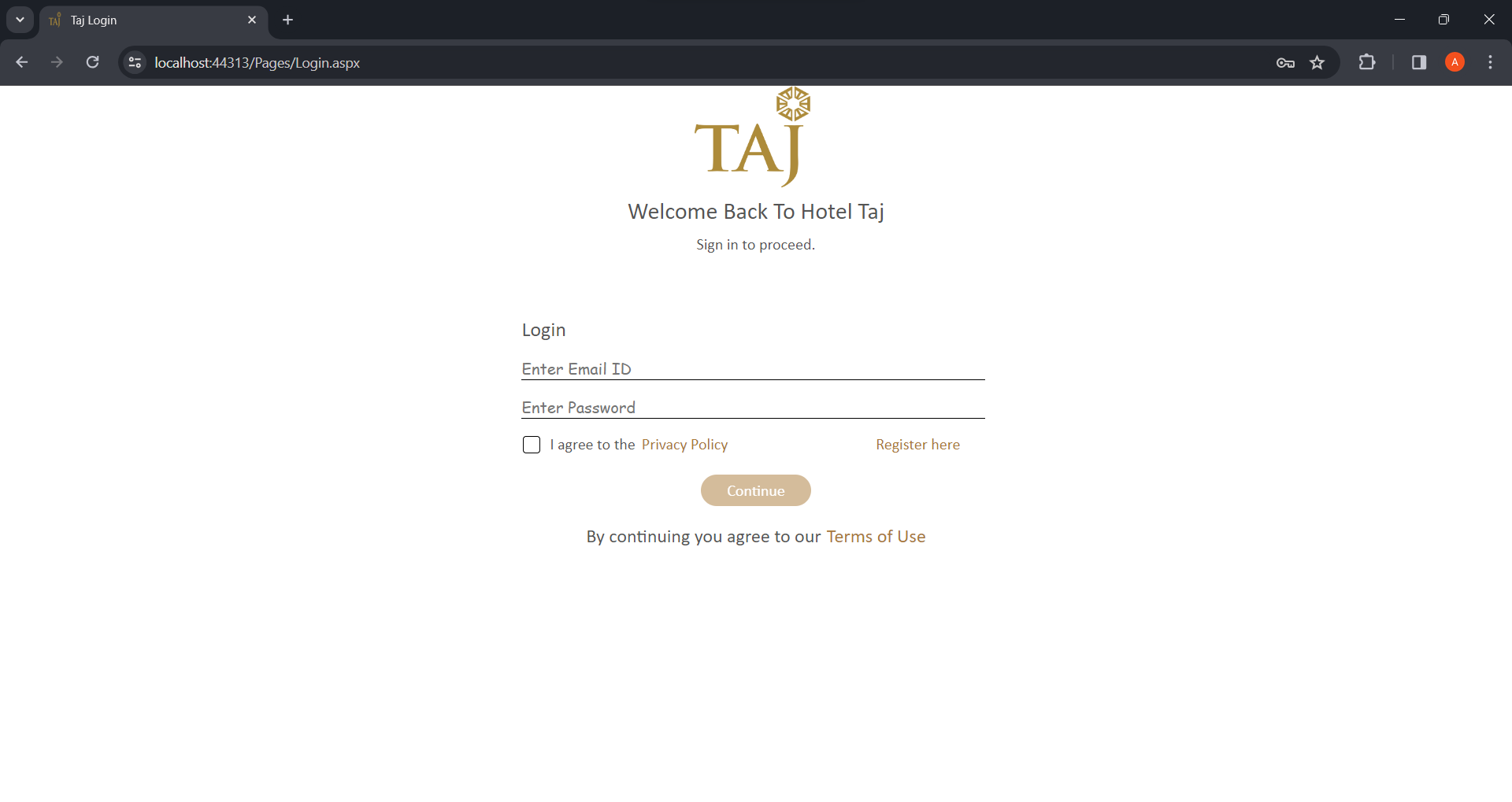


Figure 8.1 Login Page UI

## Register Page

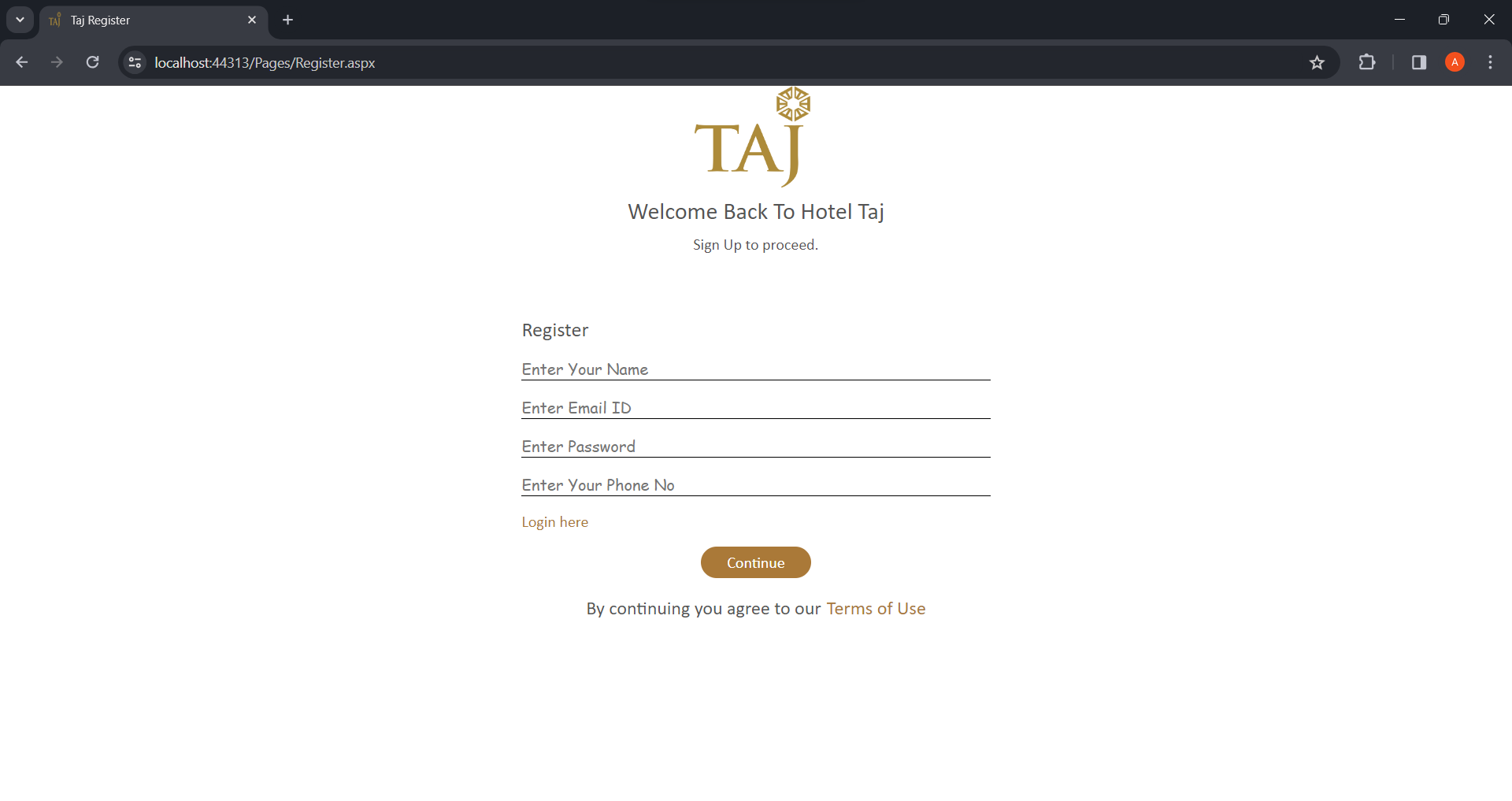


Figure 8.2 Register Page UI

## Home Page

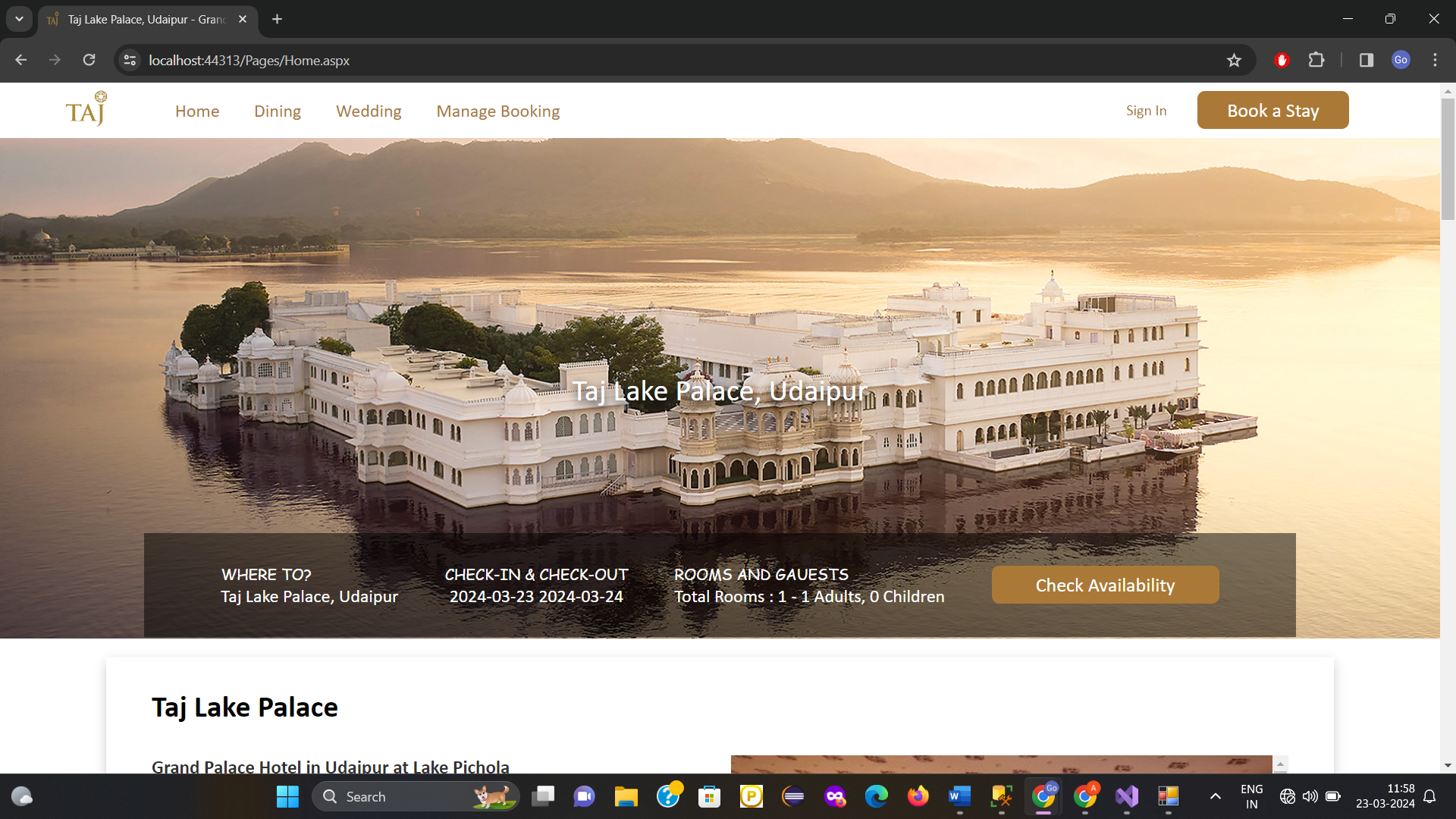


Figure 8.3 Home Page UI

## Search for Room Availability

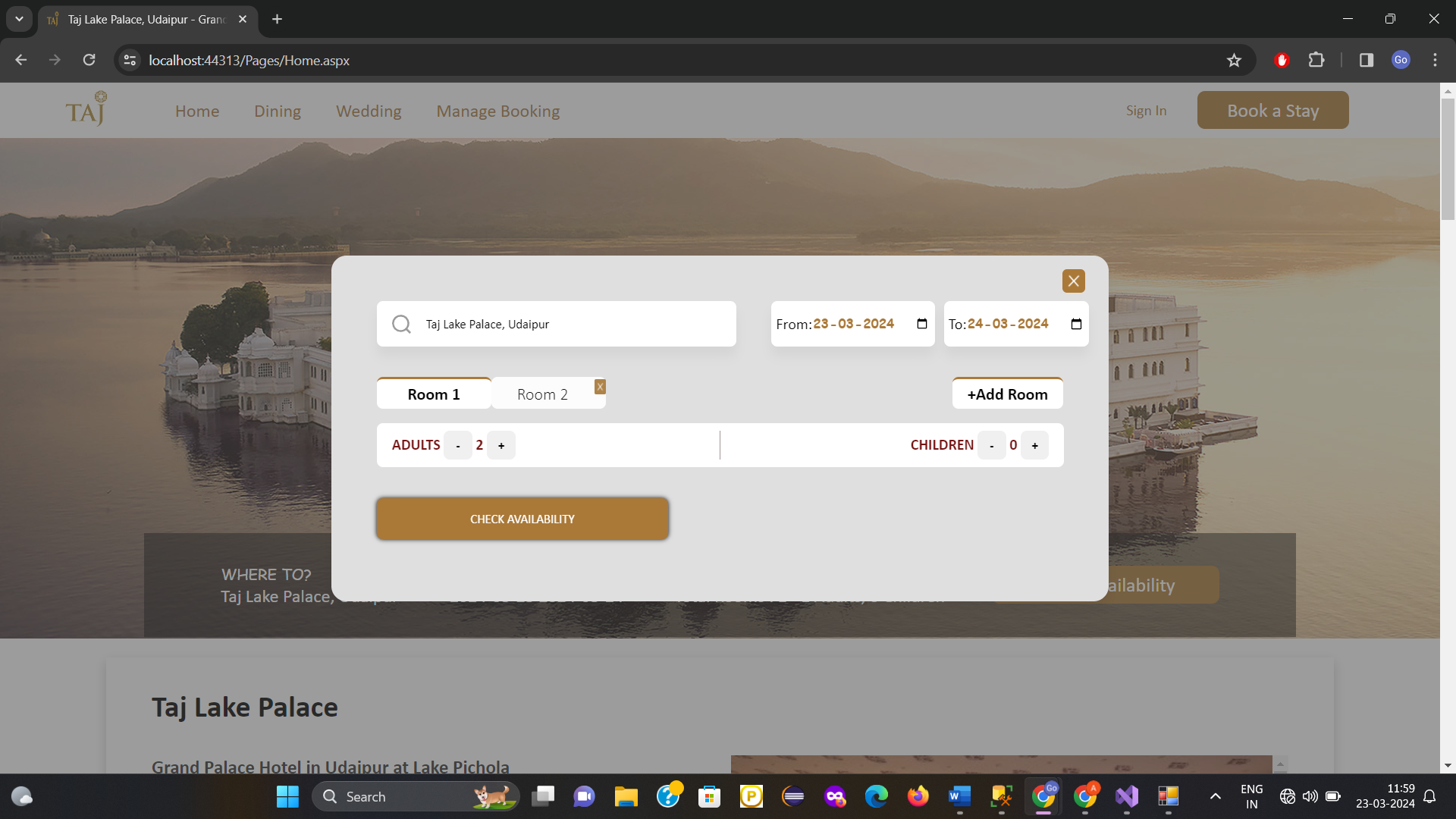


Figure 8.4 Search For Rooms UI

## Available Rooms

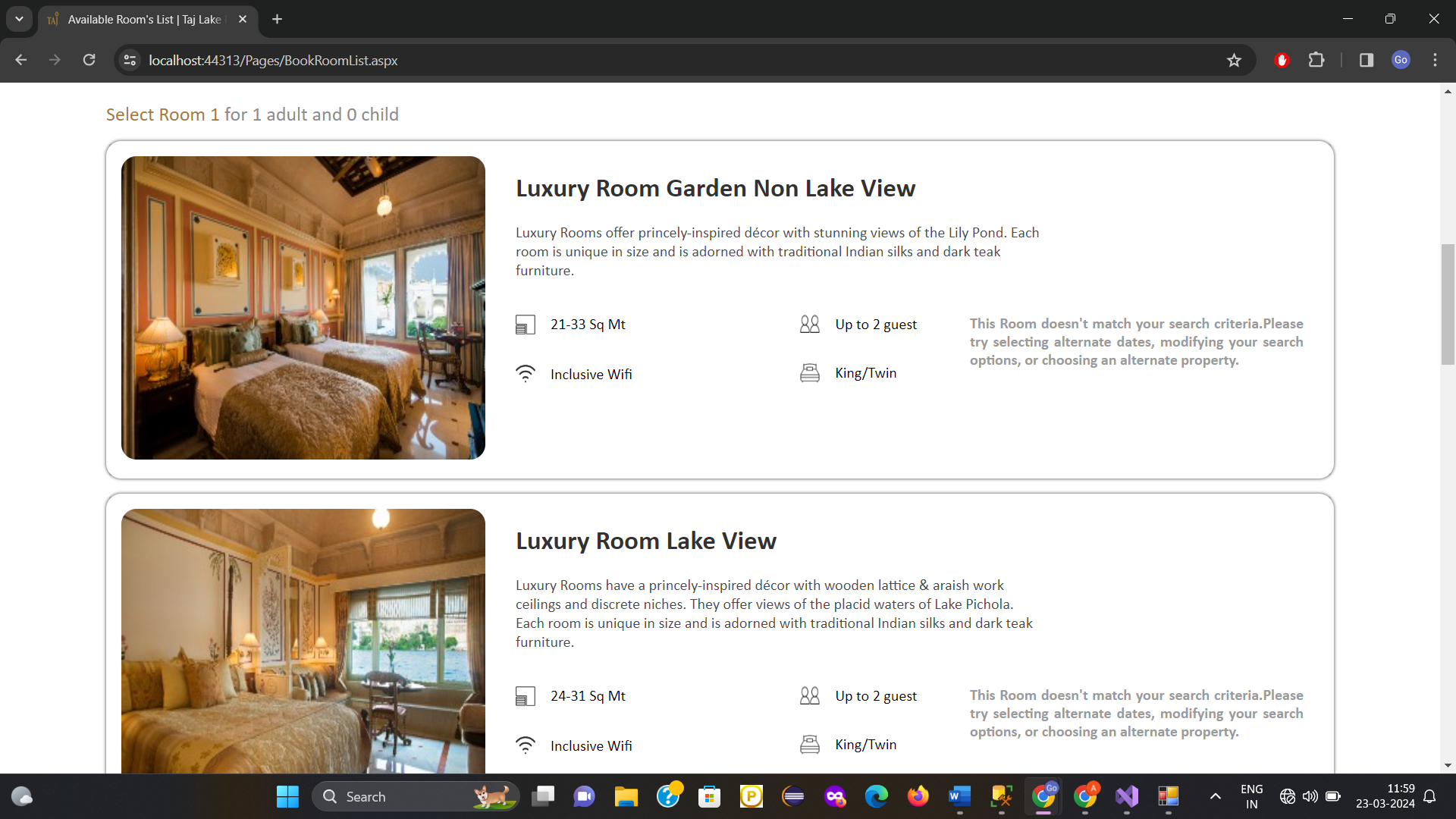


Figure 8.5 Available Rooms UI

## Select Rooms

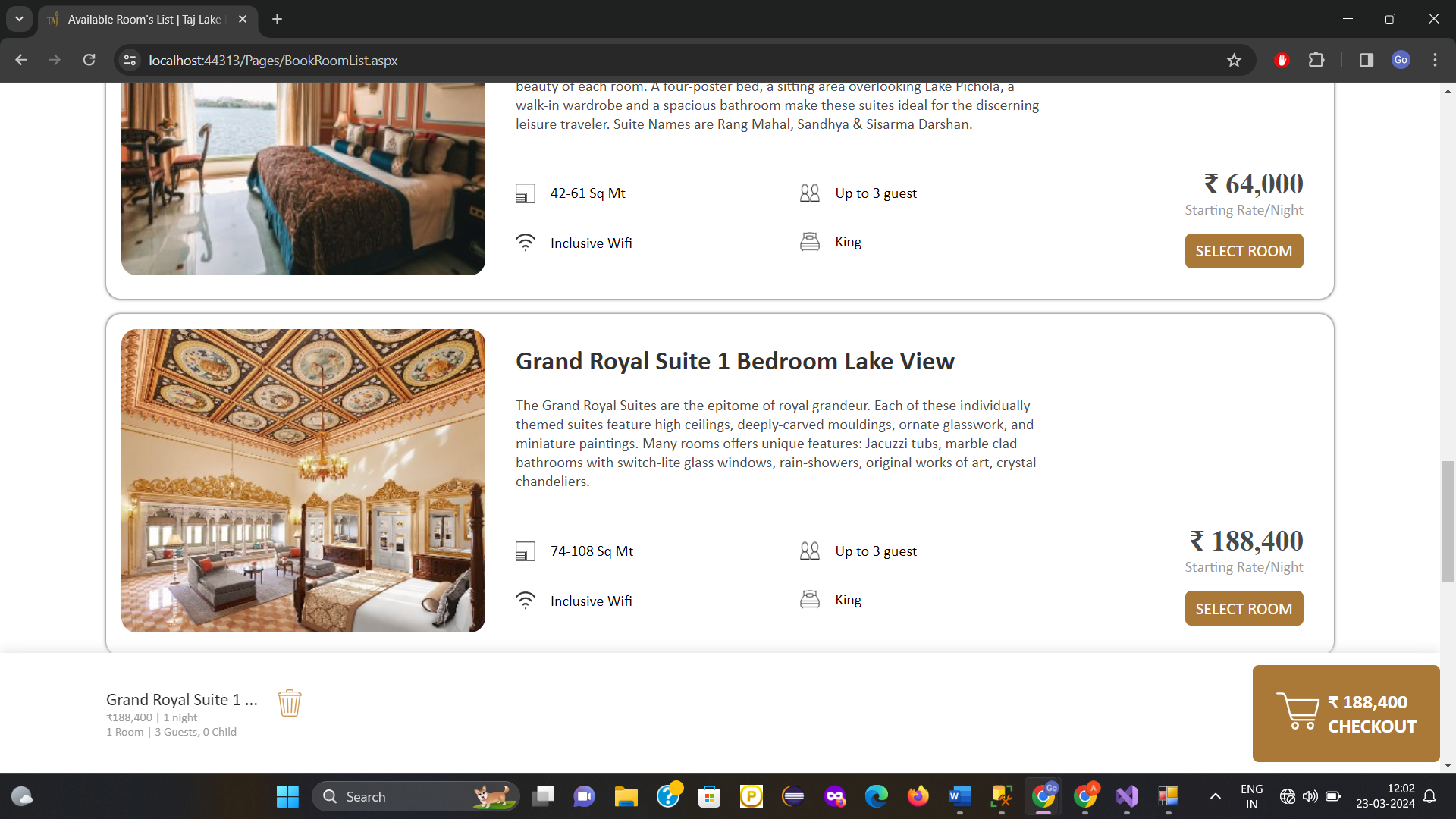


Figure 8.6 Select Rooms UI

## Booking Details

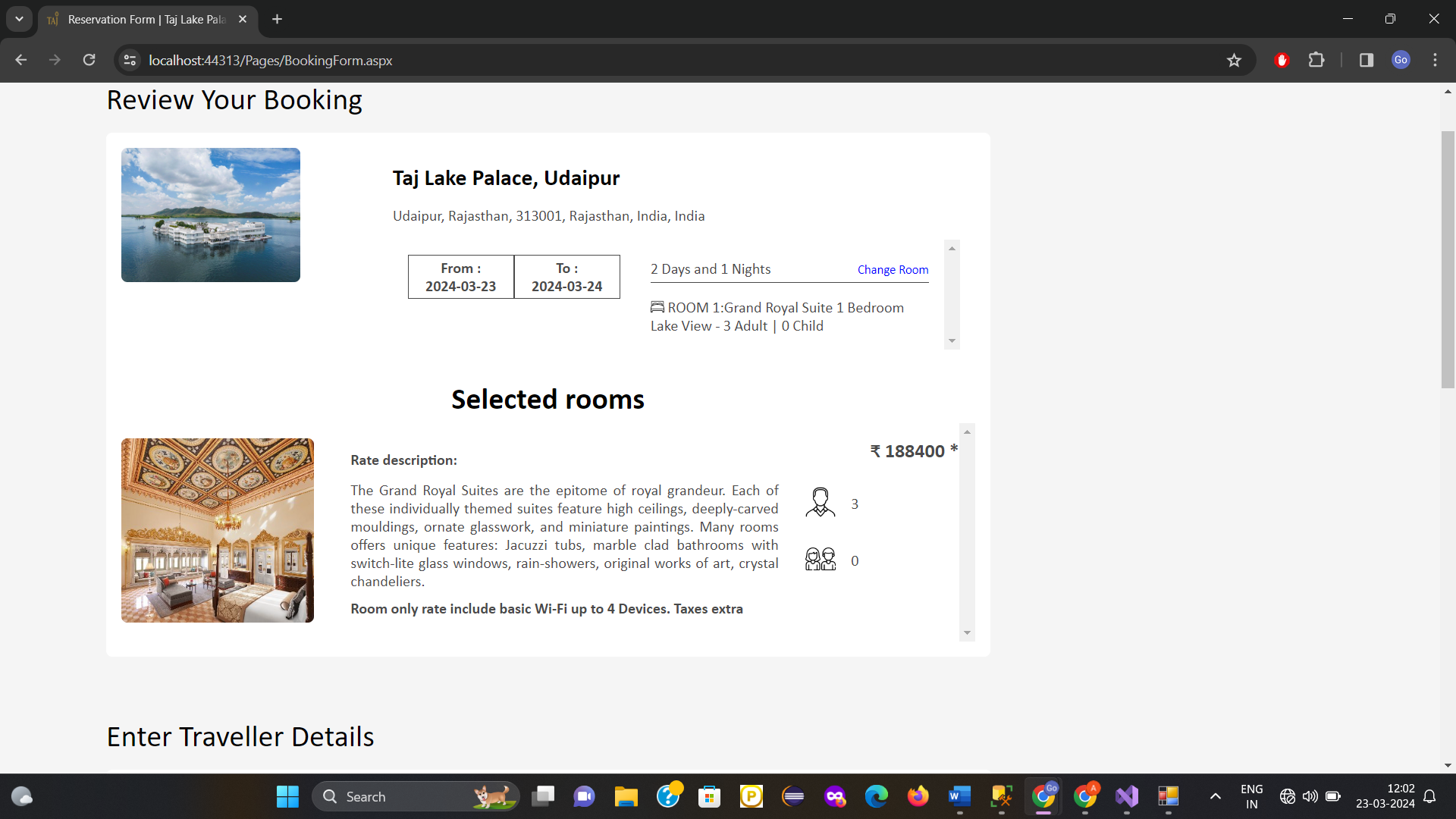


Figure 8.7 Booking Details UI

## Booking Form

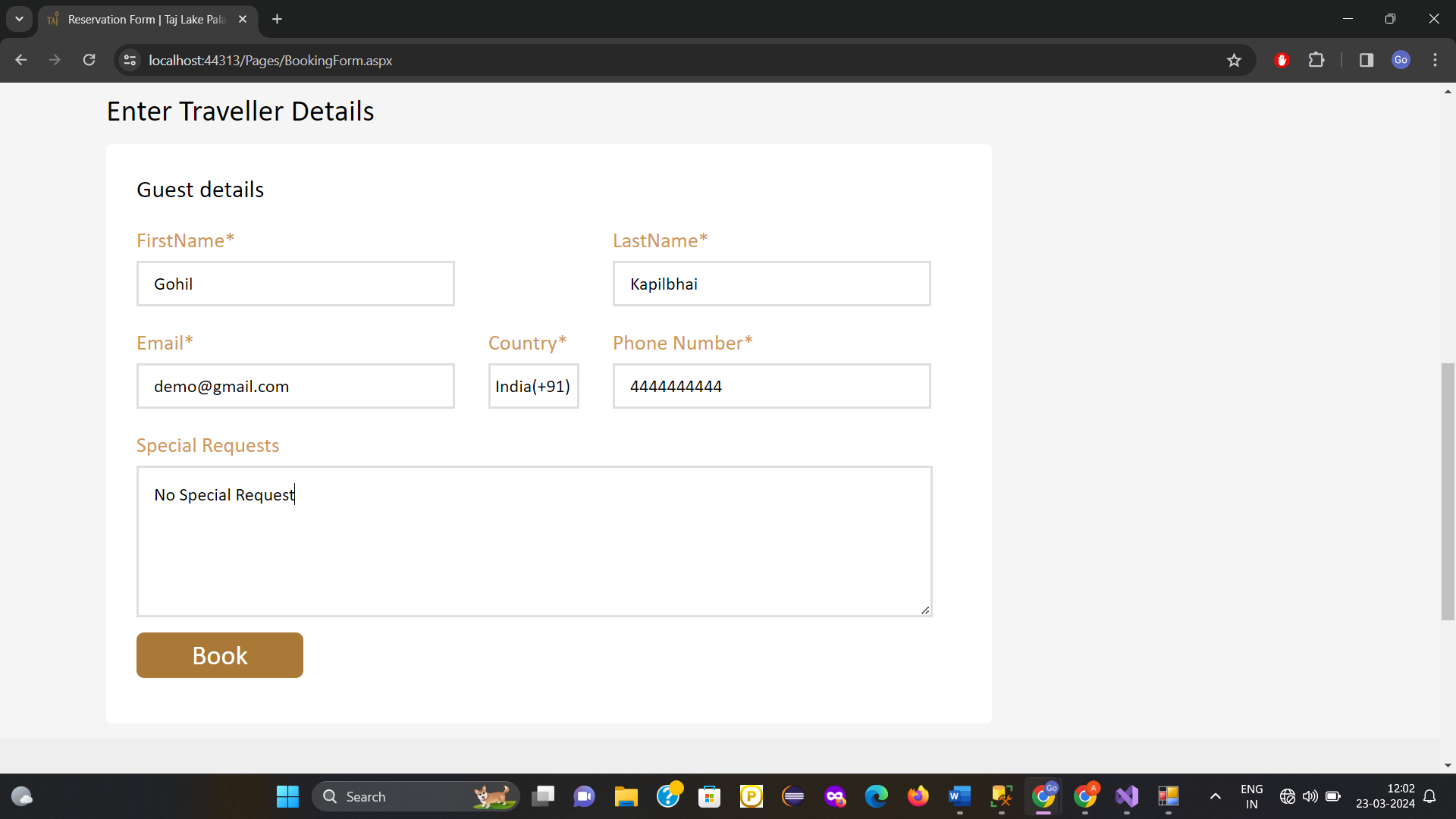


Figure 8.8 Booking Form UI

## Reservation

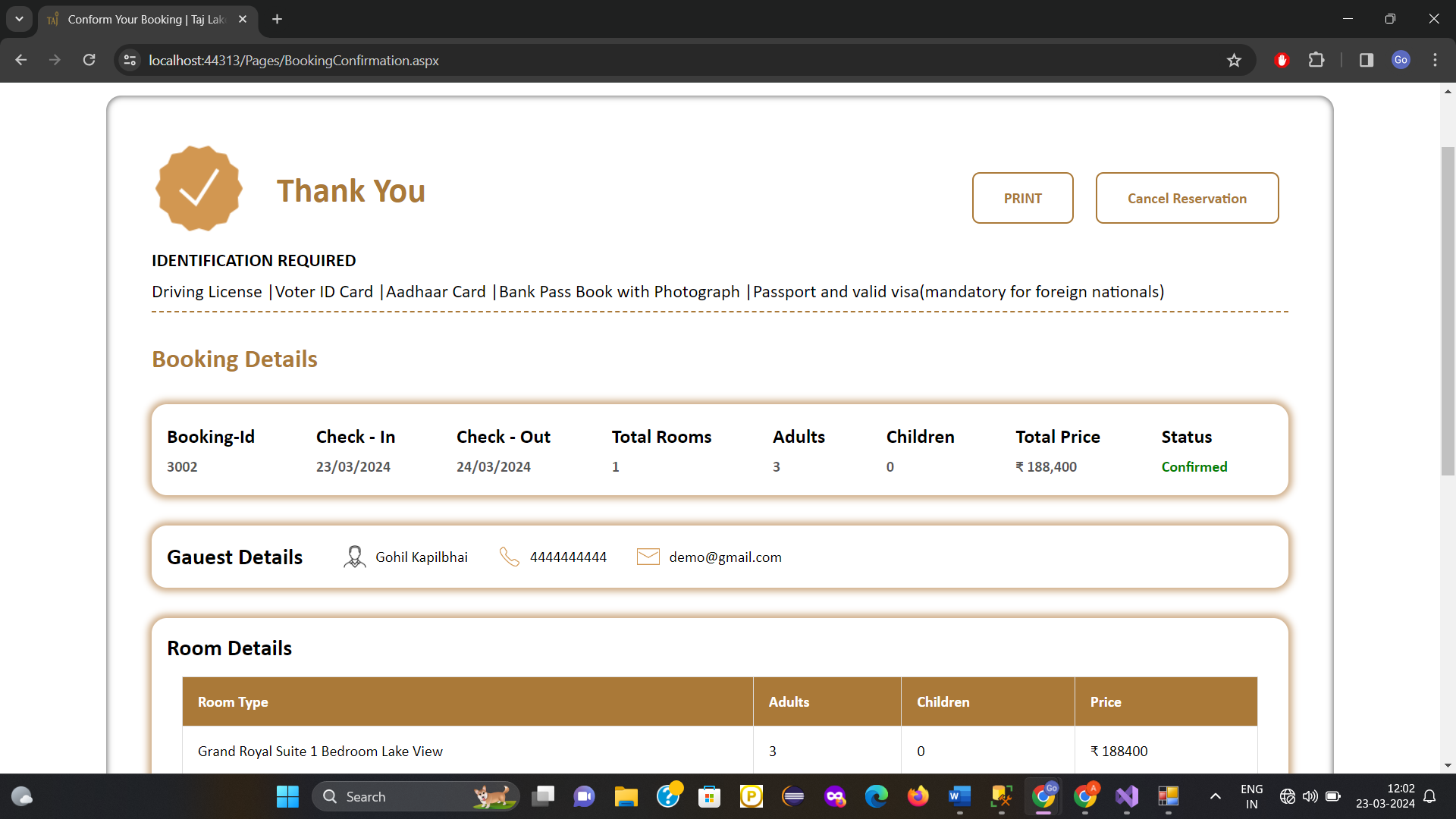


Figure 8.9 Reservation UI

## Bill Pdf

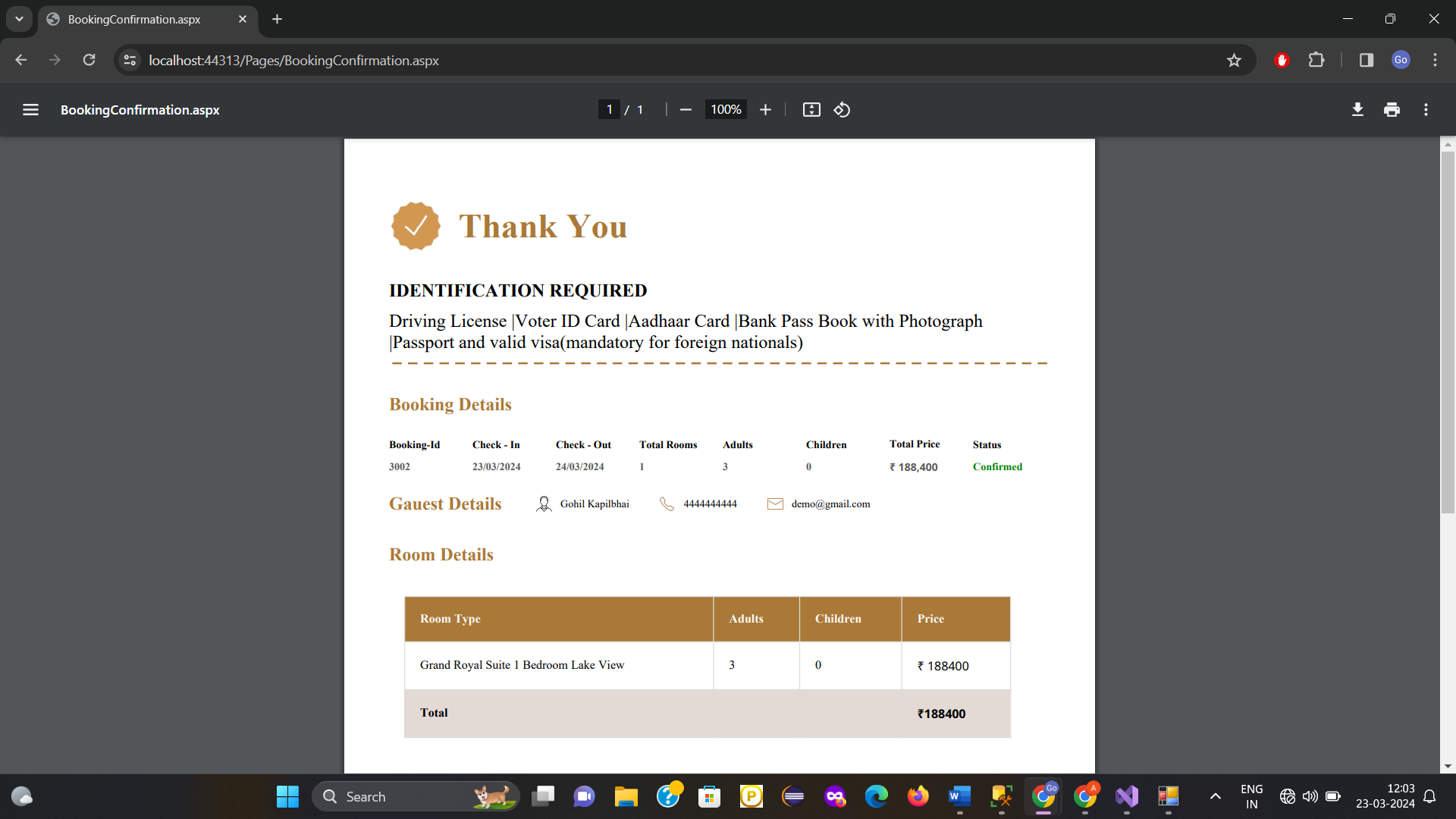


Figure 8.10 Bill PDF

## Find Booking

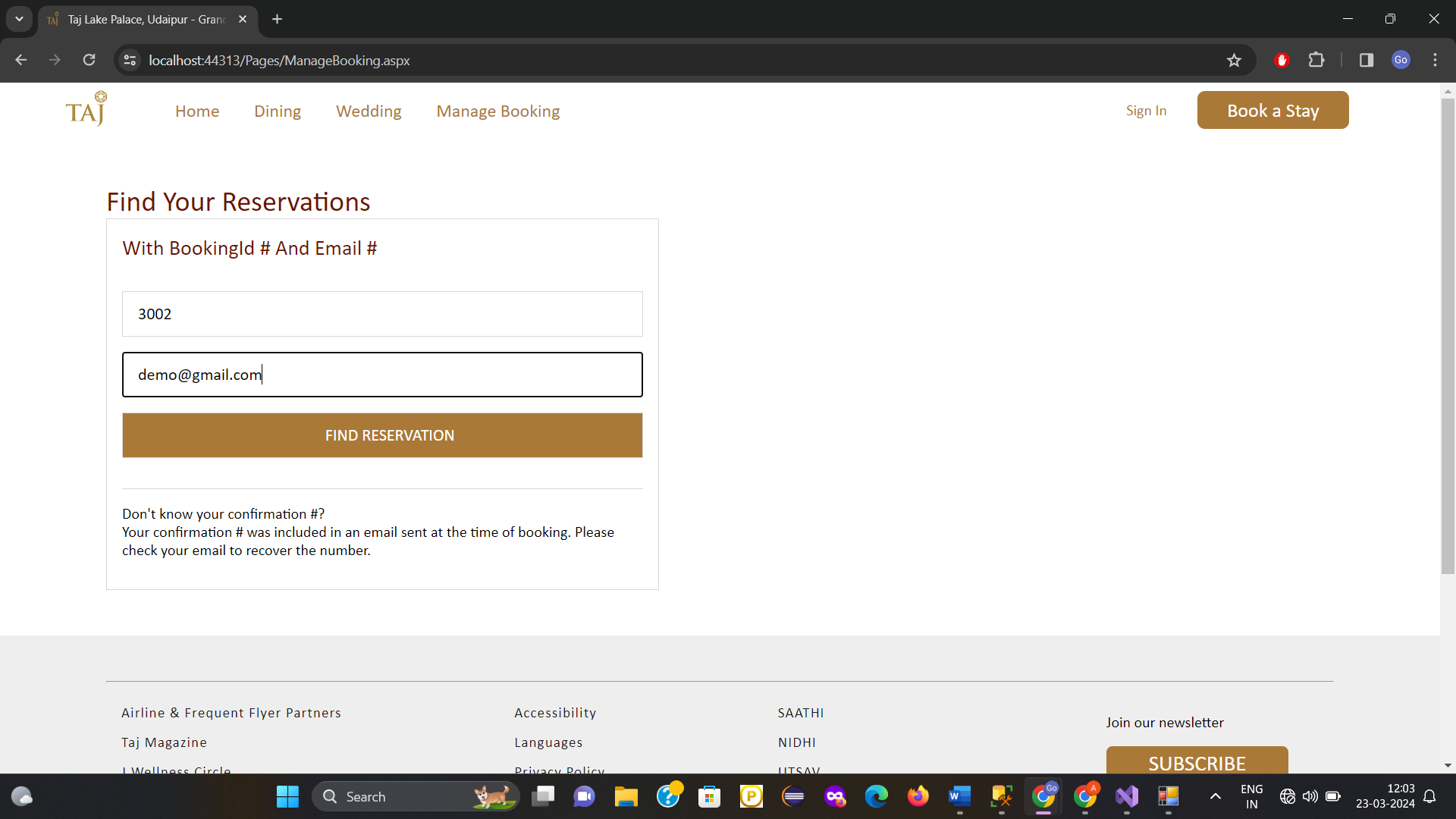


Figure 8.11 Find Booking UI

# Conclusion

After the end of the project we have successfully implemented the below functionalities

1. **Register and login**

* User can Register and login in the system

1. **Search room availability**

* User can search the room based on their criteria no of rooms, adults in one room, children in one room etc.

1. **Select multiple rooms**
   * User can book more than one rooms at a time of their own choice
2. **Book the room**
   * User can book the room by providing the necessary details like name, mobile no, email, any special request etc.
3. **Cancel the booking of the Room**
   * If user wants to cancel the booking then the user can cancel the booking by searching that reservation
4. **Search Booking by booking-id and e-mail**
   * User can search their booking by e-mail and booking id from there they can either print or cancel the reservation
5. **Print the bill for the reservation**
   * User can download the pdf copy of the bill once they have booked room successfully.

# Limitation and Future Extension

## Limitations

In our project there is some limitations which is listed below

1. We haven’t implemented the real payment gateway
2. Once the user have booked the room they can’t change the room and switch the booking for another room (they can do by cancelling the previous reservation and book again)
3. There is no charge for cancelling the booking so the ungenuine user can book rooms and cancel reservation any no of times.
4. This project is just for enhancing the skills on the .net webservices so it doesn’t put much aspect on the security (data encryption),Real time data ,admin site for the room management ,data processing , various charts for financial and data analysis etc.
5. Lack of authentication while booking room i.e. user can book room just with email and phone no. (not verifying identity like Aadhar no or pan at the time of booking)

At end we made our project for enhancing our skills on the service-based architecture so it may fail in the real time scenarios where authentication, security, performance, scalability would be the most needy things.

## Future Extension

Following extensions possible in our system.

1. Implement the real time payment gateway
2. Make booking functionality more flexible so user can customize or change the selection after booking.
3. Apply the appropriate charge upon cancelling the booking so the ungenuine user can’t just book and occupy rooms for times and at end just cancel room. This charge should be based on the no of days remaining for the room check-in date.
4. Add data processing, various financial charts so it would be easy for the admin to analyze the data and make important decision accordingly.

There can be many future extensions to the system according to the need of the Hotel Owner.

# Bibliography

## Books or Some reading content

Most of the reference taken from the google classroom material provided by the university teachers.

- <https://classroom.google.com/u/0/c/NTkwOTgwMjc5MzYx>

## Web

We also have followed the other resources from the web which are listed below

* <https://learn.microsoft.com/en-us/troubleshoot/developer/visualstudio/csharp/language-compilers/write-web-service>
* <https://www.c-sharpcorner.com/UploadFile/govind77/introduction-to-service-oriented-architecture/>