A
PROJECT REPORT
ON

TRAVEL AND TOURISM MANAGEMENT SYSTEM

SUBMITTED BY

Ms. Masal Pratibha Kakasaheb

SUBMITTED TO
SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE

FULFILLMENT OF DEGREE

MASTER OF COMPUTER APPLICATION (SEM-I)

UNDER THE GUIDANCE OF
Prof. Deepali Gavhane
Through,

Sadhu Vaswani Institute of Management Studies for Girls,
Koregaon Park, Pune 411001

2024-25

DECLARATION BY STUDENT

To,

The Director,

SVIMS, Koregaon Park, Pune

I, undersigned hereby declare that this project titled, **"Travel and Tourism Management System"** written and submitted by me to SPPU, Pune, in partial fulfilment of the requirement of the award of the degree of **MASTER OF COMPUTER APPLICATION (MCA-I)** under the guidance of **Prof. Deepali Gavhane,** is my original work.

I further declare that to the best of my knowledge and belief, this project has not been submitted to this or any other University or Institution for the award of any Degree.

Place: Pune

Date:

(Pratibha Kakasaheb Masal)

ACKNOWLEDGEMENT

I extend my sincere gratitude to Dr. B. H. Nanwani, Dr. Rajesh Kashyap and Prof. Deepali Gavhane for allowing me to carry out the study and for their constant encouragement, valuable suggestions, and guidance during the research work.

I extend my special thanks to Dr. Shveti Chandan and Dr. Neeta Raskar for their kind co-operation and inspiration.

I extend my special gratitude to my dearest family members and friends who encouraged and motivated me to complete the project report.

Place: Pune

Date:

Pratibha Kakasaheb Masal

CHAPTER 1:

1.1Introduction

The main objective of the Travel and Tourism Management System is to manage the details of Customer, Hotel Booking, Cancellation and Tourism places. It manages all the information about Users, Hotel, and Packages etc. The project is totally built at administrative end and thus only the administrator is guaranteed the access to the backend database. The purpose of this project is to build an application program to reduce the manual work for managing Tourists, Booking, Places etc.

This application will help in accessing the information related to the travel to the particular destination with great ease. The users can track the information related to their tours with great ease through this application. The travel agency information can also be obtained through this application.

Through this system, the propose system is highly automated and makes the travelling activities much easier and flexible. The user can get the very right information at the very right time. This system will include all the necessary fields which are required during online reservation time. This system will be easy to use and can be used by any person. The basic idea behind this project is to save data in a central database which can be accessed by any authorize person to get information and saves time and burden which are being faced by their customers.

Administrator can access and modify the information stored in the database of this system, this includes adding and updating of details, and it will give accurate information and simplifies manual work and also it minimizes the documentation related work. Provides up to date information. Finally booking confirmation notification will be send to the users.

Tourists can register by providing personal details, make new reservation and book only one hotel and package and can make cancellation.

1.2 Need for System:

- **Customer Support and Feedback**
- A support ticketing system or live chat for inquiries. Customer reviews and feedback functionality.
- **Admin Dashboard**
- Overview of bookings, finances, and user interactions. Tools for managing users, listings, and content.
- **Itinerary Management**
- Ability for users to create and manage their travel itineraries.
- Options to share itineraries with friends or family.
- **Destination Guides**
- Information about destinations including attractions, cultural sites, and local events.
- Maps, weather information, and tips for travelers.
- **Content Management System (CMS)**
- For managing blogs, articles, and media related to travel and tourism.
- SEO tools for website visibility.
- Multi-language and Currency Support
- **User Registration and Profiles**
- User accounts for tourists, tour operators, and service providers.
- Profiles with personal information, preferences, and past bookings.
- **Search and Booking System**
- Search functionality for hotels, tours, travel packages, and attractions.
- Date selection for availability checking.
- Booking management with confirmation emails.
- **Tour Packages Management**
- Creation and management of various tour packages.
- Pricing, itinerary details, duration, and special offers.
- **Accommodation Management**
- Listings of hotels, hostels, and alternative accommodations.
- Managing room types, prices, amenities, and availability.
- **Payment Gateway Integration**
- Secure payment processing for bookings.
- Support for various payment methods (credit card, PayPal, etc.).

1.3 Scope and feasibility of work:

- **Tourists**: General public looking for travel options, accommodations, and activities.
- **Tour Operators**: Businesses offering tours, excursions, and travel packages.
- **Accommodations**: Hotels, hostels, and other lodging providers wanting to manage bookings.
- **Administrators**: System managers responsible for maintaining the platform, managing user accounts, and ensuring operational efficiency.
- 2. Core Features
- **User Registration and Profiles**: Allow users to create and manage personal accounts.
- **Search and Discovery**: Enable users to search for and filter tours, accommodations, and points of interest based on various criteria (e.g., location, price, ratings).
- **Booking System**: Implement a secure booking process that includes payment options and confirmation notifications.
- **Itinerary Management**: Allow users to create, view, and modify travel itineraries.
- **Technical Feasibility**
    - Technology Requirements**:** Identify the technology stack for development (programming languages, databases, APIs).
    - System Architecture**:** Outline the overall architecture (cloud-based, on-premise).
    - Integration Needs: Assess integration with other systems (payment gateways, CRM, social media). ▪ **Operational Feasibility**
    - Organizational Readiness: Assess the organization's capacity to implement the new system.
    - Staffing Requirements: Identify staff roles needed for development, implementation, training, and ongoing support.
    - Training and Support: Plan for staff training and end-user support.
- **Financial Feasibility**
    - Cost Estimate: Provide an estimate of development costs, maintenance, marketing, and other operational expenses.
    - Funding Sources: Identify potential funding sources (internal budgeting, loans, investors).
    - Revenue Projections: Project potential revenue from system usage (subscription fees, transaction fees).
    - Cost-Benefit Analysis: Evaluate the financial benefits against the anticipated costs.

1.4 Operating Environment-H/w and S/w:

Table 1.4.1: Hardware requirements:

Processor	80 GB System type: 64-bit Operating System, x64-bassed processor
Hard Disk	Total size of Hard disk: 1 TB
RAM	1 GB Installed memory (RAM):8.00 GB (7.43 GB Usable)
Monitor	Colour Monitor
Other Hardware	Keyboard, Mouse etc.

Table 1.4.2: Software requirements:

Operating System	Windows 11, Microsoft windows 10.
Language	HTML,CSS,JAVA
Database	MySql
Tool Kit	Notepad++, Apache Netbeans, Mysql Workbench,

1.5 Architecture of System:

1. Presentation Layer (Frontend):

- This layer is responsible for the user interface, which interacts with users (tourists, tour operators, administrators).
- HTML, CSS, JavaScript for basic structure and styling.
- Frontend frameworks: React, Angular, Vue.js for a responsive and interactive user experience.
- Mobile compatibility: Consider using React Native or Flutter for a mobile app.
- User Registration/Login
- Search Functionality (tours, accommodations)
- Booking Management Dashboard
- Customer Feedback System
- Admin Panel for managing tours, users, and reviews

2. Application Layer (Backend):

- **Programming Languages:** Python (Django/Flask), PHP (Laravel), and Node.js (Express.js).
- **Authentication Module:** User login, registration, and profile management.
- Booking Management Module: Handles reservations, cancellations, and modifications.
- **Tour Management Module:** CRUD (Create, Read, Update, and Delete) functionality for tour packages.
- **Accommodation Management Module:** Similar functionality for hotels and other accommodations.
- **Analytics Module:** Gathers data on user interactions, bookings, etc., and generates reports.

3. Data Layer (Database)

- **Relational Databases:** MySQL, PostgreSQL for structured data.
- **NoSQL Databases:** MongoDB for flexible schema requirements, especially for user-generated content.
- **Users Table**: user_id, name, email, password_hash, user_type (tourist/operator/admin), preferences.
- **Tours Table**: tour_id, title, description, price, duration, itinerary, available_dates.
- **Accommodation Table**: accommodation_id, name, location, price, amenities, reviews.

1.6 Detail Description of Technology Used:

1. HTML (Hypertext Markup Language)

HTML is the standard markup language used to create the structure of web pages. It defines the elements of a web page and organizes content such as text, images, links, and multimedia.

HTML (HyperText Markup Language) is the standard language used to create and structure content on the web. It forms the foundation of every website, organizing elements like headings, paragraphs, images, links, and more using a system of tags. HTML tells the browser how to display text, where to place images, and how to link pages together. Although HTML alone doesn't make a website look fancy or interactive, it works together with CSS (for styling) and JavaScript (for behavior) to build fully functional and visually appealing web pages. Learning HTML is the first step toward becoming a web developer.

2. CSS (Cascading Style Sheets)

CSS is used for styling HTML documents and controlling the layout and presentation of web pages. It allows for the separation of content from design.

CSS (Cascading Style Sheets) is the language used to style and design web pages. While HTML structures the content, CSS controls how that content looks—like colors, fonts, spacing, and layout. With CSS, you can make a plain HTML page look visually appealing by customizing everything from backgrounds and buttons to animations and page layouts. It works by applying rules to HTML elements using selectors and properties. CSS is essential for creating modern, responsive, and attractive websites.

Key Features

Selectors: CSS uses selectors to apply styles to specific HTML elements (e.g., .class, #id, element).

Box Model: CSS uses the box model to define the dimensions, padding, margins, and borders of elements.

Responsive Design: Media queries enable responsive design, allowing web pages to adapt to different screen sizes and devices.

3. JavaScript

JavaScript is a scripting language that enables interactivity on web pages. It is used to manipulate the Document Object Model (DOM), validate forms, create animations, and handle events.

Key Features

Event Handling: JavaScript can respond to user actions like clicks, form submissions, and keyboard presses.

DOM Manipulation: JavaScript can create, read, update, and delete HTML elements dynamically.

Asynchronous Programming: With the use of AJAX and Promises, JavaScript can communicate with servers to fetch data without refreshing the page.

4. Core Java

Core Java typically refers to the foundational features of the Java programming language, which is oriented toward building robust, platform-independent applications. It is widely used for back-end development, including server-side programming.

Key Features

Object-Oriented: Java is an object-oriented programming language that emphasizes encapsulation, inheritance, and polymorphism.

Platform-Independent: Java code is compiled into bytecode, which can run on any platform that has a Java Virtual Machine (JVM).

Rich Standard Library: Java provides a vast standard library (Java API) that includes essential classes for data structures, networking, I/O, and more.

MySQL is an open-source relational database management system (RDBMS) based on Structured Query Language (SQL). It is widely used for managing and storing data in web applications and is known for its reliability, ease of use, and performance. Let's explore its features, basic concepts, and how it can be integrated with web technologies like HTML, CSS, JavaScript, and Java.

5. MySQL

Relational Database: MySQL organizes data into tables that can be related to each other, allowing for complex queries across different datasets.

SQL (Structured Query Language): It uses SQL for defining, querying, and modifying data, providing a standardized way to interact with the database.

Open Source: MySQL is available under the GNU General Public License, although commercial versions are also offered.

2. Key Features of MySQL

Scalability: MySQL can handle large databases (up to several terabytes) and thousands of concurrent clients, making it suitable for enterprise-level applications.

Performance: Optimized for speed and efficiency, MySQL can quickly process complex queries.

Data Integrity: Supports ACID compliance (Atomicity, Consistency, Isolation, Durability) which ensures reliable transactions.

Compatibility: MySQL works well with various operating systems, programming languages (like PHP, Java, Python), and frameworks.

Storage Engines: Provides different storage engines (InnoDB, MyISAM, etc.) that offer trade-offs between performance and features.

Chapter 2:

2.1 Proposed System:

The proposed Tourism Management System is an integrated, user-friendly platform designed to streamline the travel experience for tourists and service providers alike. By consolidating key features such as user registration, real-time search and booking capabilities, itinerary management, and customer feedback mechanisms, the system aims to enhance accessibility and efficiency in the tourism industry. The architecture comprises a responsive frontend developed using modern frameworks like React or Angular, paired with a robust backend powered by technologies such as Node.js or Python. Data is securely managed in relational databases like PostgreSQL, ensuring effective handling of user profiles, bookings, and accommodations. Integration with third-party APIs allows for seamless payment processing and access to valuable external information, such as weather forecasts and local attractions. Designed with scalability in mind, the system will be hosted on a cloud infrastructure, enabling it to handle increasing user demands and facilitate continuous updates. Overall, this proposed system aspires to improve customer satisfaction, promote local businesses, and drive operational efficiency within the tourism sector.

2.2 Objectives:

Enhance User Experience: Create an intuitive and user-friendly interface that allows tourists to easily search, compare, and book tours and accommodations.

Streamline Booking Processes: Implement efficient booking and reservation management features that facilitate secure transactions, reducing manual overhead for both users and service providers.

Centralize Information Access: Provide a comprehensive platform that aggregates tour offerings, accommodations, user reviews, and local attractions, enabling users to make informed decisions.

Support Multiple User Roles: Design functionalities tailored for various user roles—tourists, tour operators, and administrators—ensuring that each group has the necessary tools to manage their specific needs effectively.

Real-Time Updates: Incorporate real-time availability tracking for tours and accommodations, ensuring users have access to the most current information, thus minimizing booking conflicts.

Integrated Payment Solutions: Facilitate secure and seamless payment processing through integration with trusted payment gateways, ensuring transactional security and user trust.

Scope:

1. Target Users

- Tourists**:** General public looking for travel options, accommodations, and activities.
- Tour Operators: Businesses offering tours, excursions, and travel packages.
- Accommodations: Hotels, hostels, and other lodging providers wanting to manage bookings.
- Administrators: System managers responsible for maintaining the platform, managing user accounts, and ensuring operational efficiency.

2. Core Features

- User Registration and Profiles: Allow users to create and manage personal accounts.
- Search and Discovery: Enable users to search for and filter tours, accommodations, and points of interest based on various criteria (e.g., location, price, ratings).
- Booking System: Implement a secure booking process that includes payment options and confirmation notifications.
- Itinerary Management: Allow users to create, view, and modify travel itineraries.
- Reviews and Ratings: Provide a framework for users to leave feedback on services and accommodations.
- Promotional Tools: Enable tour and accommodation providers to promote special offers and discounts.
- Analytics Dashboard: Equip operators with insights into bookings, user behavior, and performance metrics.

3. Technology Stack

- Frontend: Use modern web technologies like React, Angular, or Vue.js for a responsive user interface.
- Backend: Utilize frameworks such as Node.js or Django for server-side functionality.
- Database: Implement relational databases like PostgreSQL or MySQL for data storage.
- API Integrations: Integrate third-party services for payment processing, location data, or weather information.

4. Geographical Scope

- Focus on local, regional, or international tourism, depending on the target markets of the involved service providers.

2.3 User Requirements:

2.3.1 Functional Requirements:

User Registration and Authentication:

- The system must allow tourists to create an account using an email address and password.
- Users should be able to log in and log out securely.
- Password recovery and reset functionality should be available.

Search and Filter:

- Tourists must be able to search for tours, accommodations, and attractions by various criteria (location, price, type of activity).
- The system should allow users to apply filters (e.g., price range, ratings, availability).

Booking Management:

- Users should be able to book tours or accommodations through the platform.
- The system must send confirmation emails detailing the booking information.
- Tourists should have the ability to view, modify, or cancel their bookings.

Itinerary Creation and Management:

- Tourists should be able to create and manage their travel itineraries within the system.
- The system should allow users to add booked tours and accommodation details to their itineraries.

Reviews and Ratings:

- Users should be able to leave reviews and rate tours and accommodations after their experience.
- The system must display user ratings and reviews on tour and accommodation listings.

Payment Processing:

- The system must support integrated payment gateways for secure online payments.
- Users should be able to save payment information for future use.

2.3.2 Non-functional Requirements:

Performance

Response Time: The system should provide responses to user requests (e.g., search results, booking confirmations) within 2 seconds under normal load conditions.

Throughput: The system should support a minimum of 500 concurrent users without degradation in response time.

Scalability: The system architecture must allow for horizontal or vertical scaling to accommodate increasing user volume and data load.

2. Reliability

Uptime: The system should ensure 99.9% uptime, allowing for minimal interruptions in services**.**

Error Recovery: In case of failure, the system should automatically recover its state without data loss, with a recovery time of no more than 5 minutes.

3. Usability

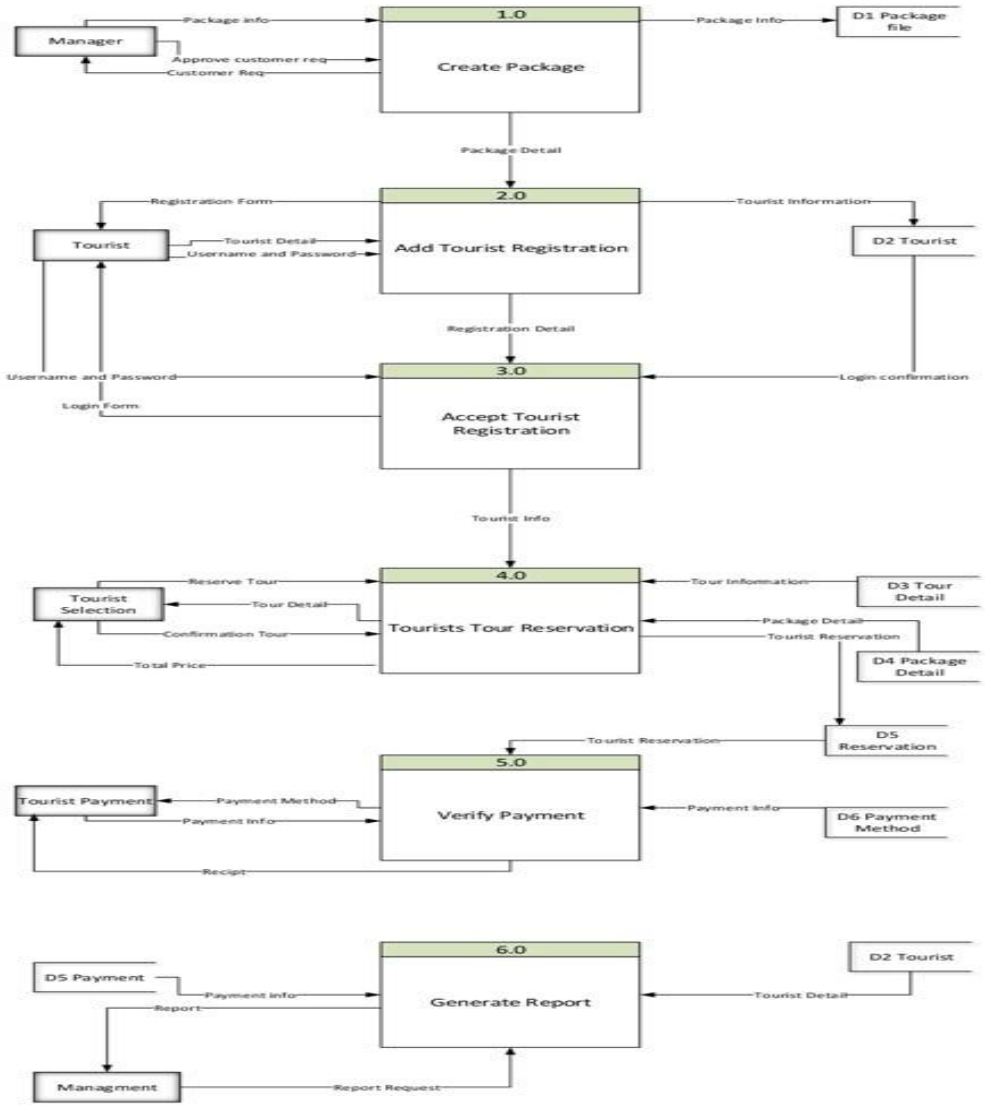
User Interface Design: The system should feature an intuitive and user-friendly interface that requires no more than three clicks to access any primary function.

Accessibility: The application should be accessible to users with disabilities in compliance with WCAG 2.1 Level AA guidelines.

Documentation: Comprehensive help resources, including user guides and FAQs, should be available within the application.

Chapter 3:

3.1 DFD:



Tourism Management System for DFD Diagram

3.2 Table Specification:

TABLE 3.2.1: ACCOUNT:

Field	Type	Null	Key	Default	Extra
username	varchar(30)	NO	PRI	NULL	
Name	varchar(30)	NO	PRI	NULL	
password	varchar(30)	NO		NULL	
security	varchar(30)	NO		NULL	
Answer	varchar(30)	NO		NULL	

TABLE 3.2.2: CUSTOMER:

Field	Type	Null	Key	Default	Extra
username	varchar(30)	NO	MUL	NULL	
id	varchar(30)	NO	PRI	NULL	
id_number	varchar(30)	NO		NULL	
name	varchar(30)	NO		NULL	
gender	varchar(30)	NO		NULL	
country	varchar(30)	NO		NULL	
address	varchar(30)	NO		NULL	
phone	varchar(30)	NO		NULL	
email	varchar(30)	NO		NULL	

TABLE 3.2.3: BOOK PACKAGE:

Field	Type	Null	Key	Default	Extra
username	varchar(30)	NO	MUL	NULL	
package	varchar(30)	NO		NULL	
persons	int(10)	NO		NULL	
id	varchar(30)	NO	MUL	NULL	
id_number	varchar(30)	NO		NULL	
phone	varchar(30)	NO	PRI	NULL	
price	varchar(30)	NO		NULL	

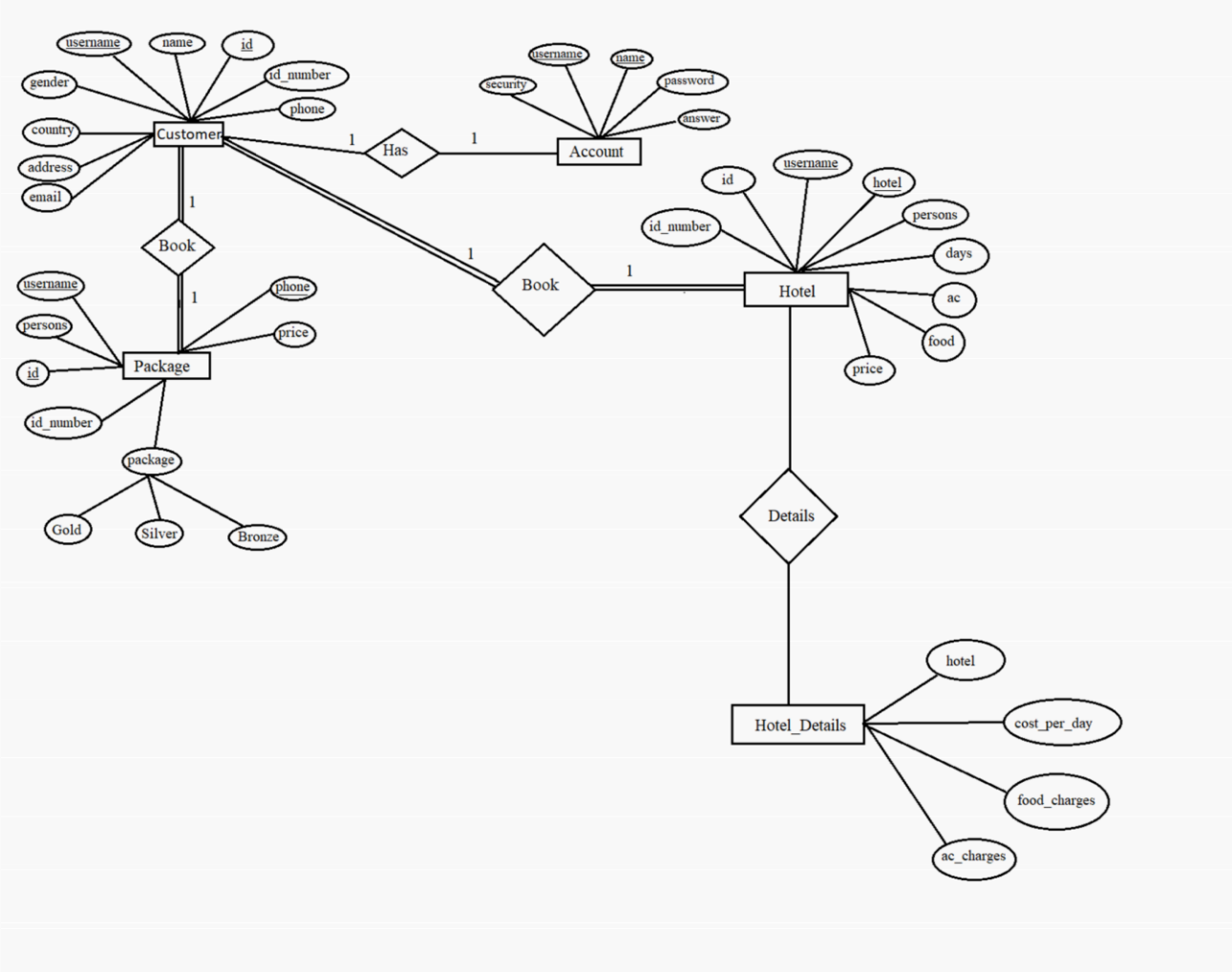
TABLE 3.2.4: BOOK HOTEL:

Field	Type	Null	Key	Default	Extra
username	varchar(30)	NO	MUL	NULL	
hotel	varchar(30)	NO	MUL	NULL	
persons	int(10)	NO		NULL	
days	int(10)	NO		NULL	
Ac	varchar(30)	NO		NULL	
food	varchar(30)	NO		NULL	
Id	varchar(30)	NO	MUL	NULL	
id_number	varchar(30)	NO		NULL	
phone	varchar(30)	NO	MUL	NULL	
price	varchar(30)	NO		NULL	

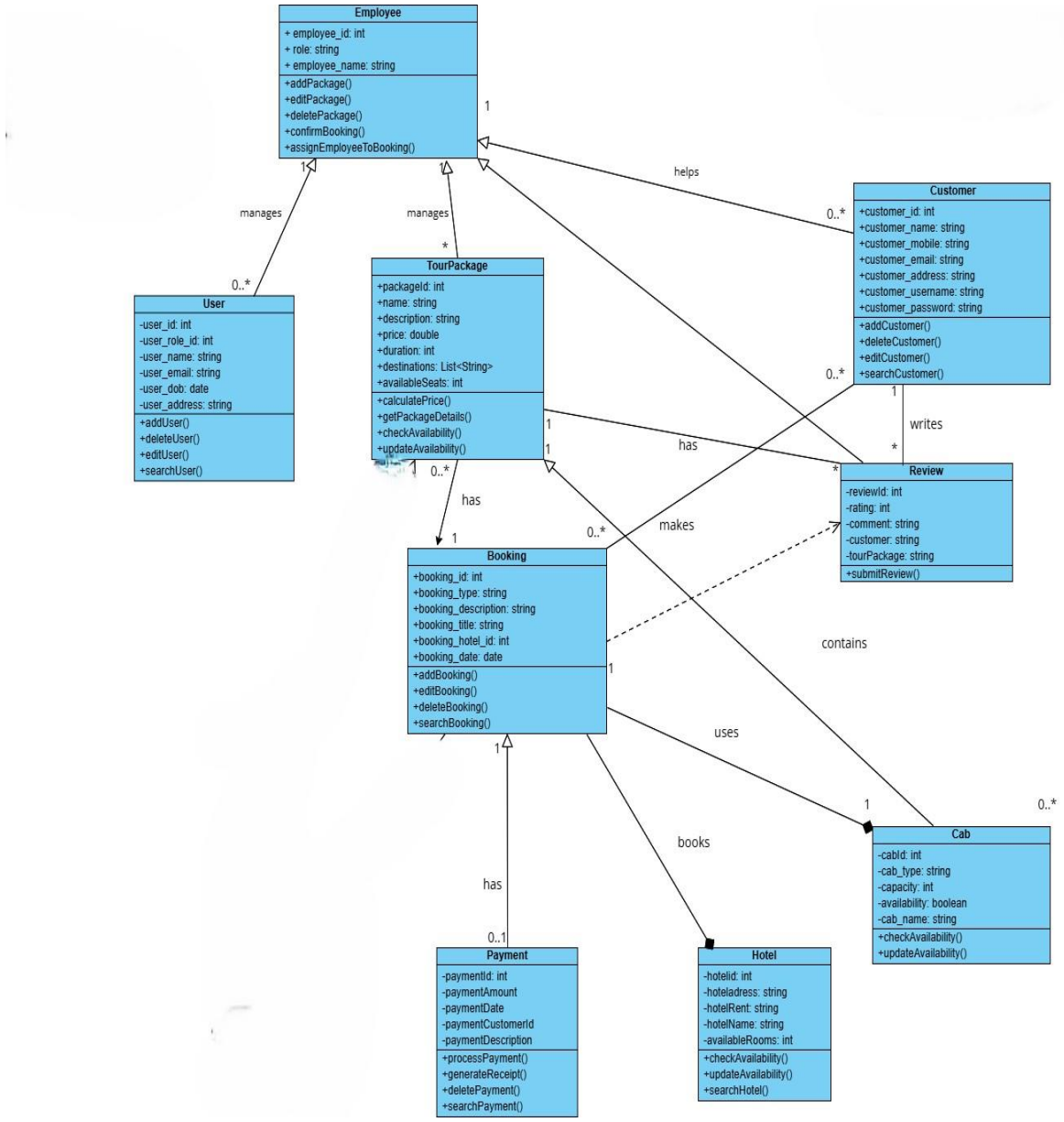
TABLE 3.2.5: HOTEL:

Field	Type	Null	Key	Default	Extra
hotel	varchar(30)	NO	PRI	NULL	
cost_per_day	int(10)	NO		NULL	
food_charges	int(10)	NO		NULL	
ac_charges	Int(10)	NO		NULL	

3.3 Entity Relationship Diagram:



3.4 Class Diagram:

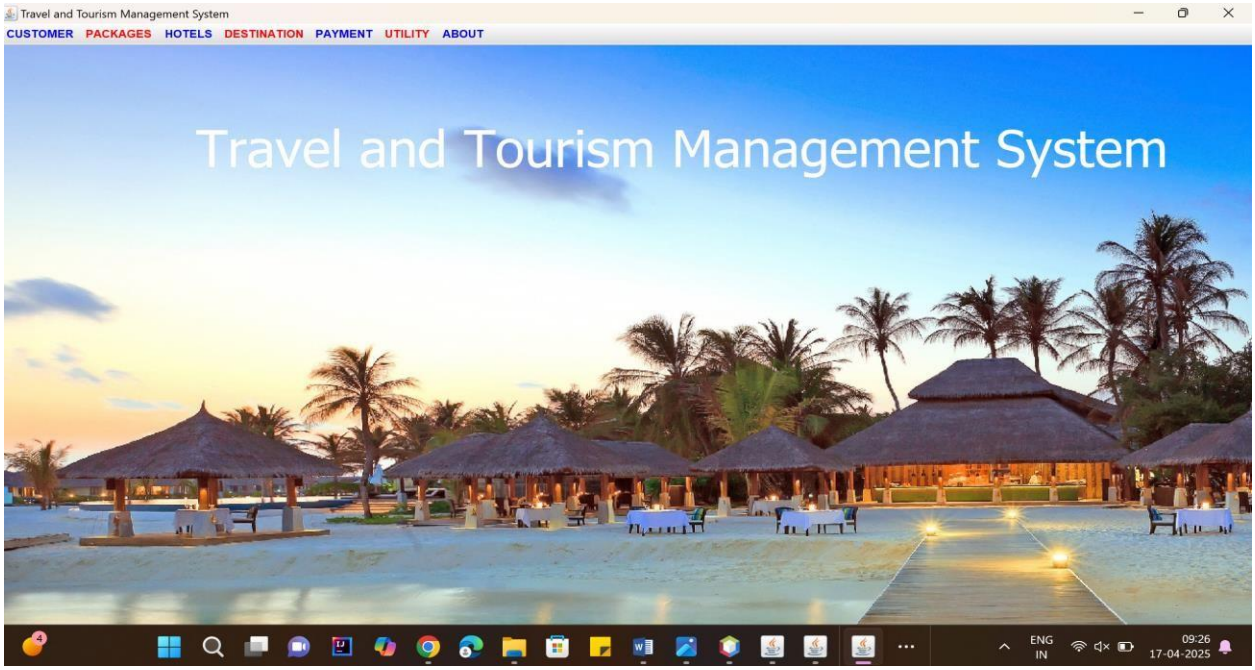


3.5 Use case diagram:

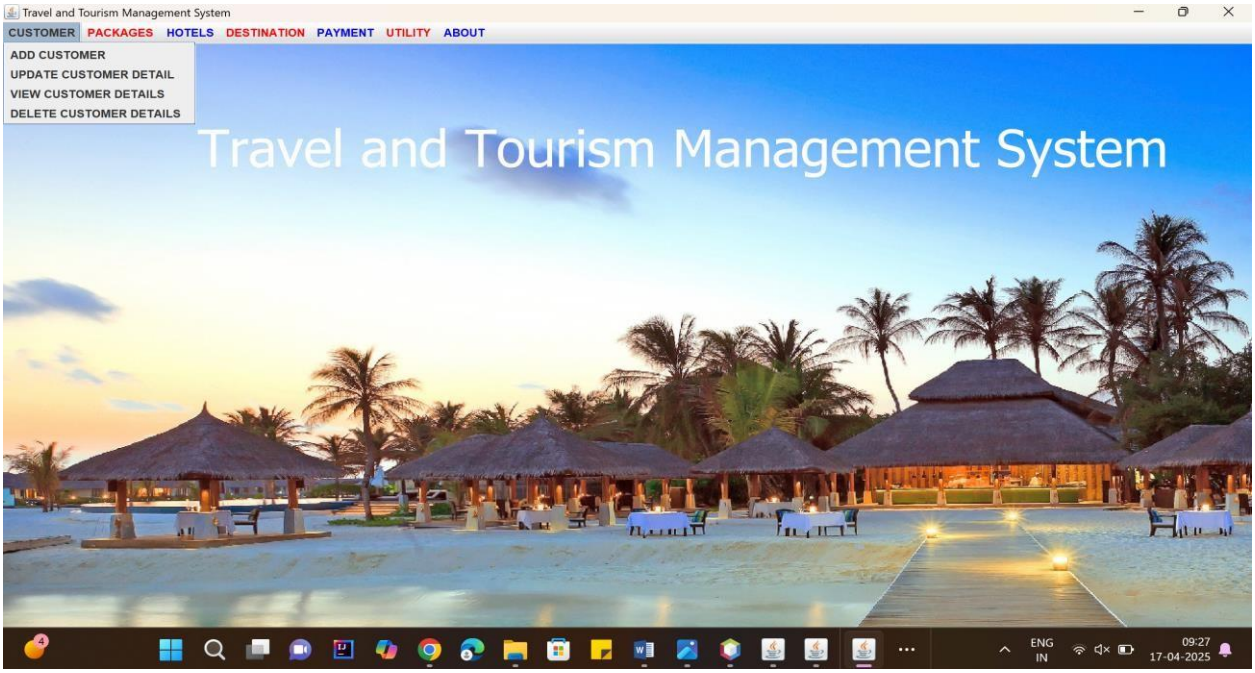
Chapter 4:

4.1 User Interface Design (Screen etc.)

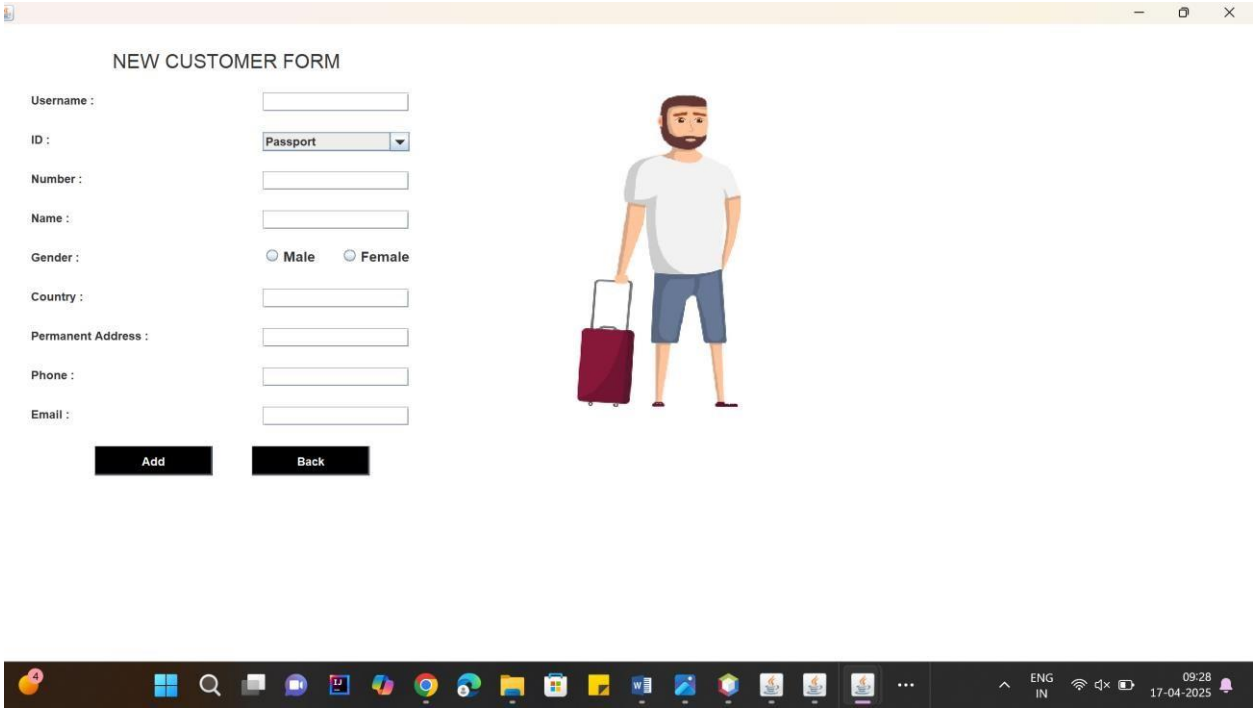
Home Page: Welcome Travel and Tourism Management system.



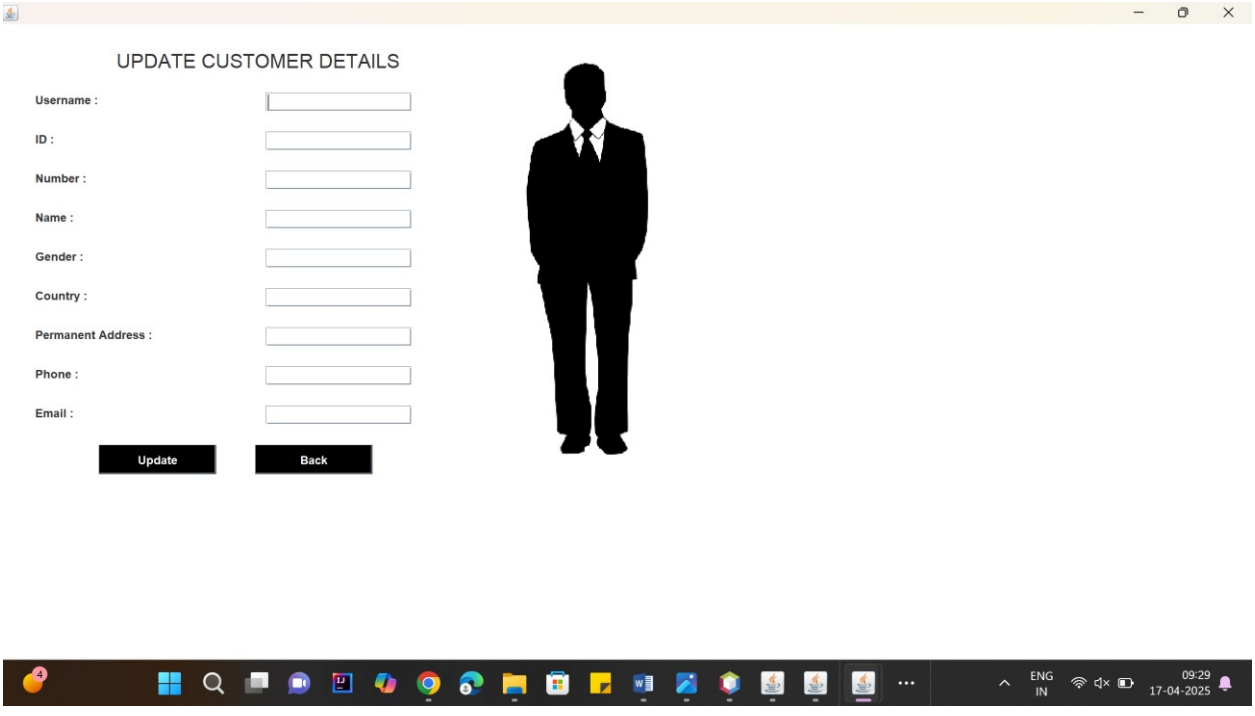
Customer: Go to add, update view, and delete customer information.



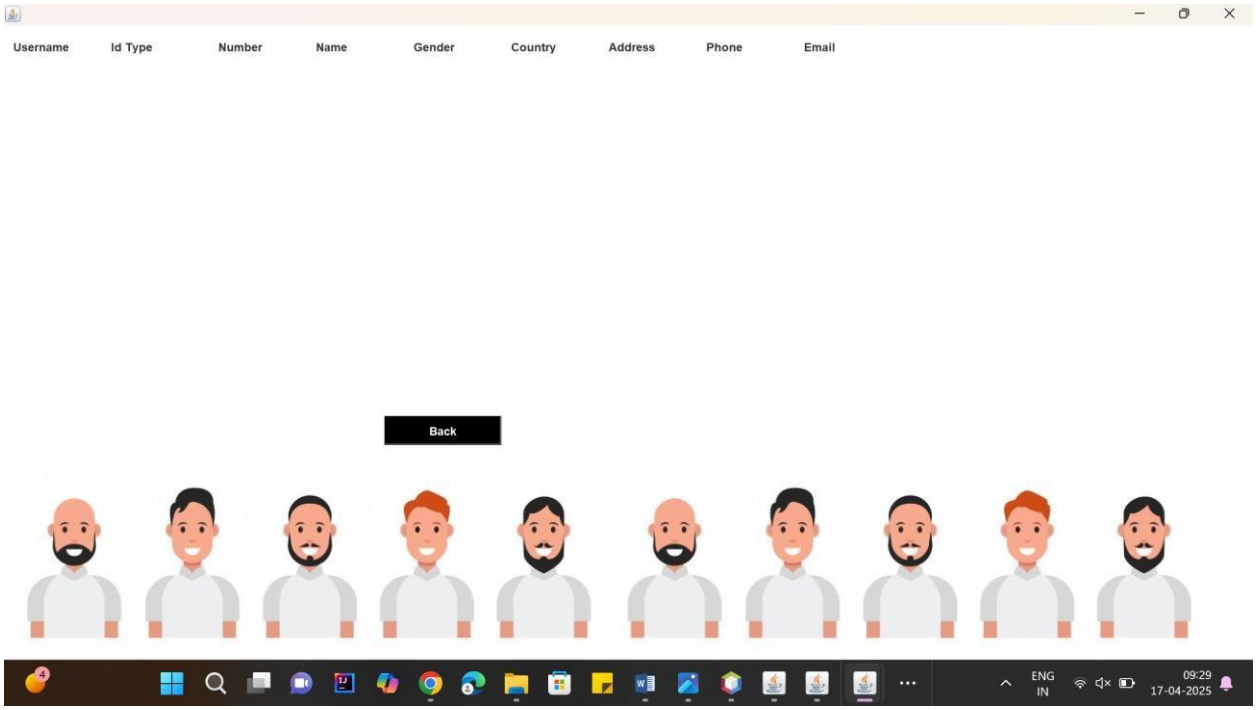
New Customer Form: Adding customer essential information.



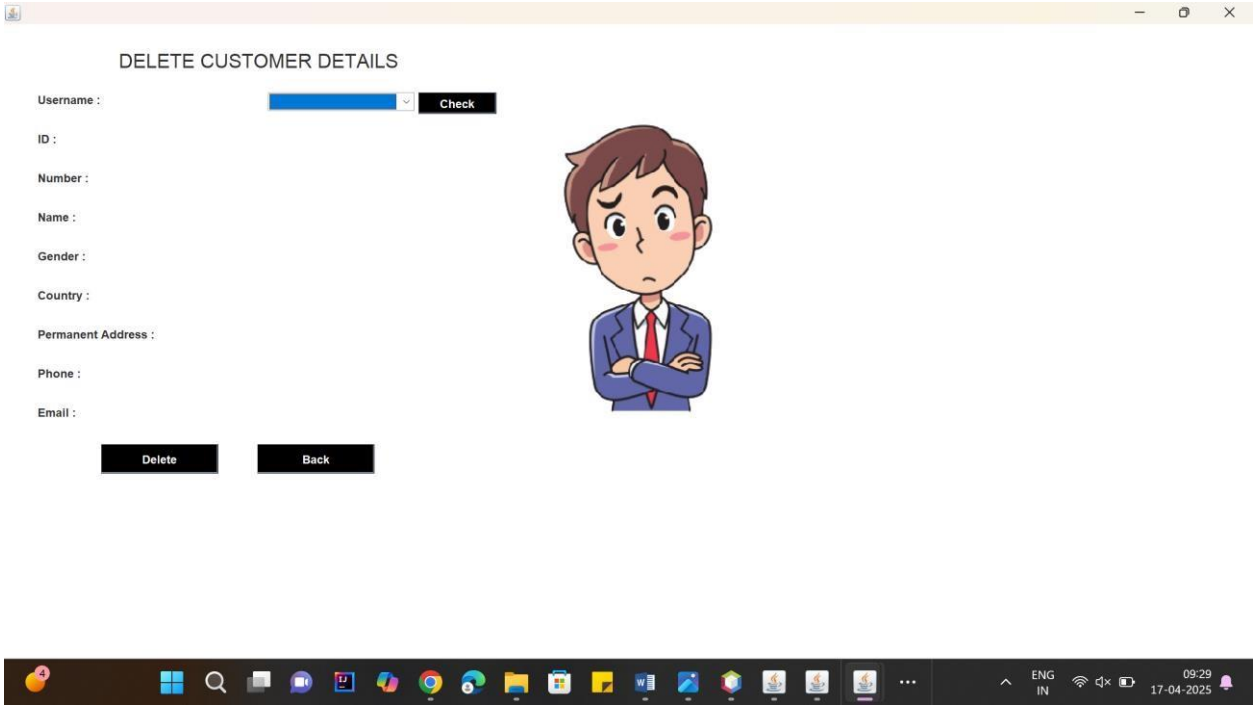
Update Customer Details: If there any changes do update customer details.



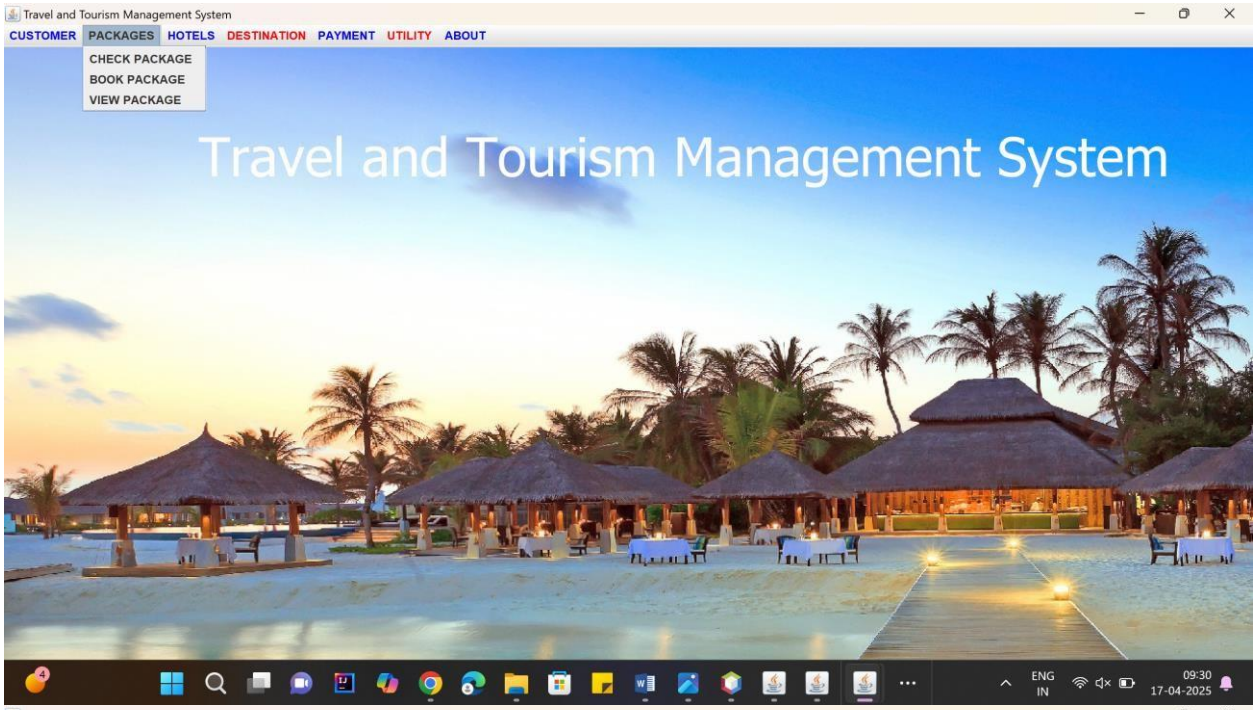
View Customer list: Showing all customers list.

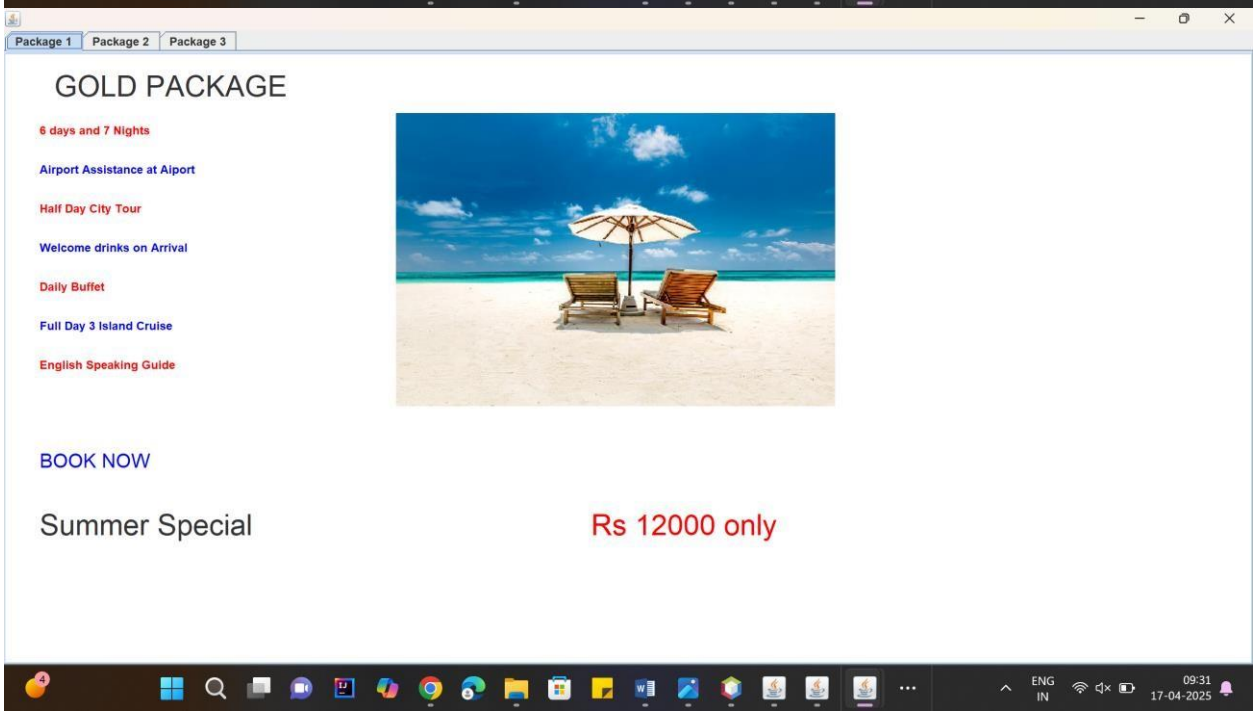


Delete Customer details: Delete customer profiles.

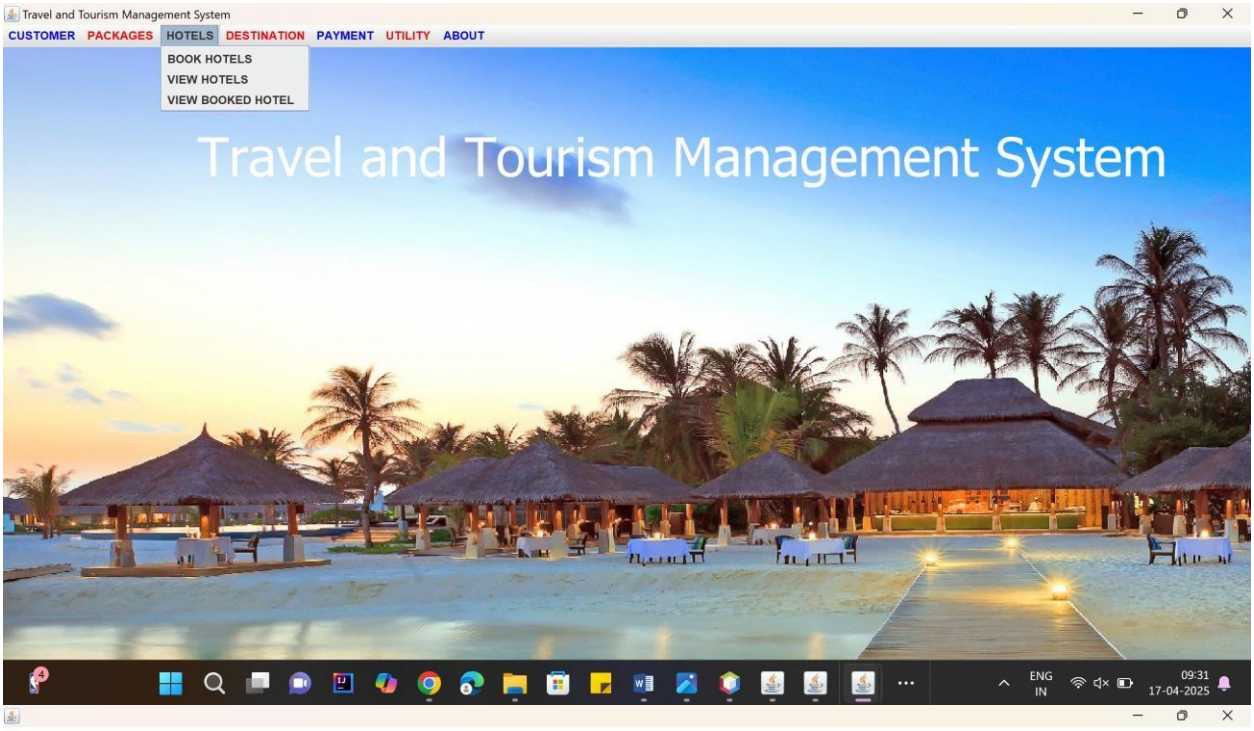


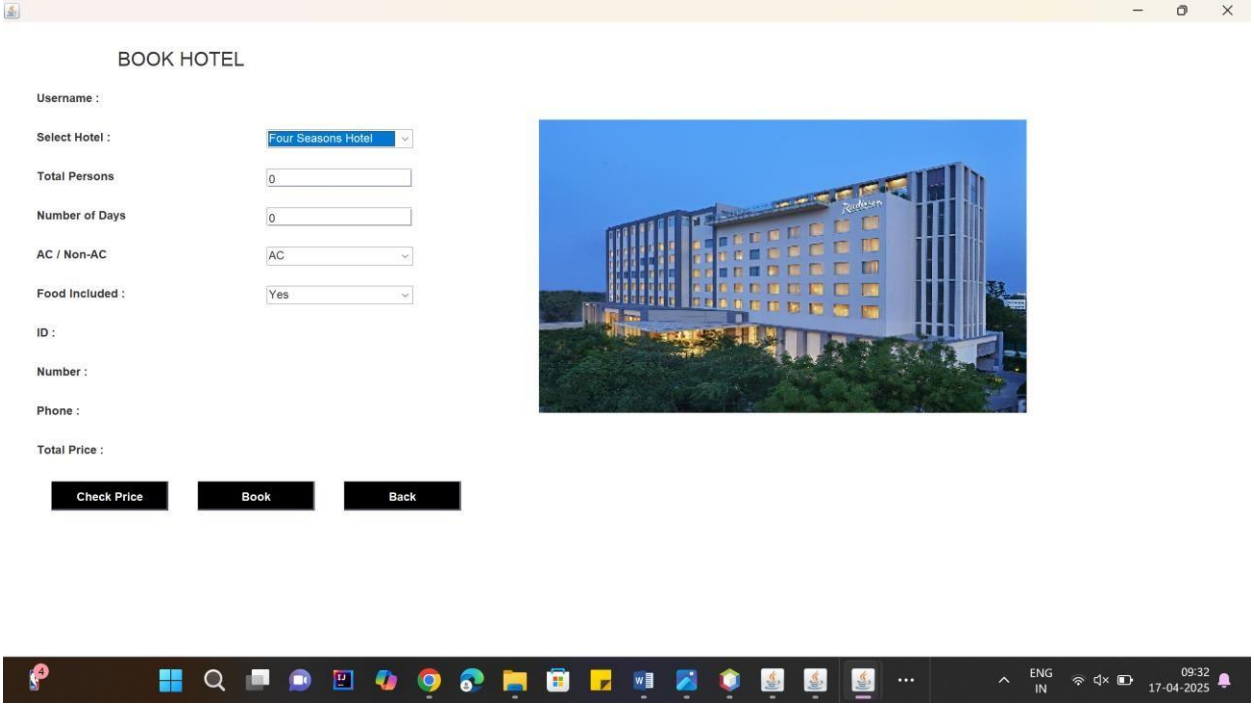
Packages: Showing packages details.

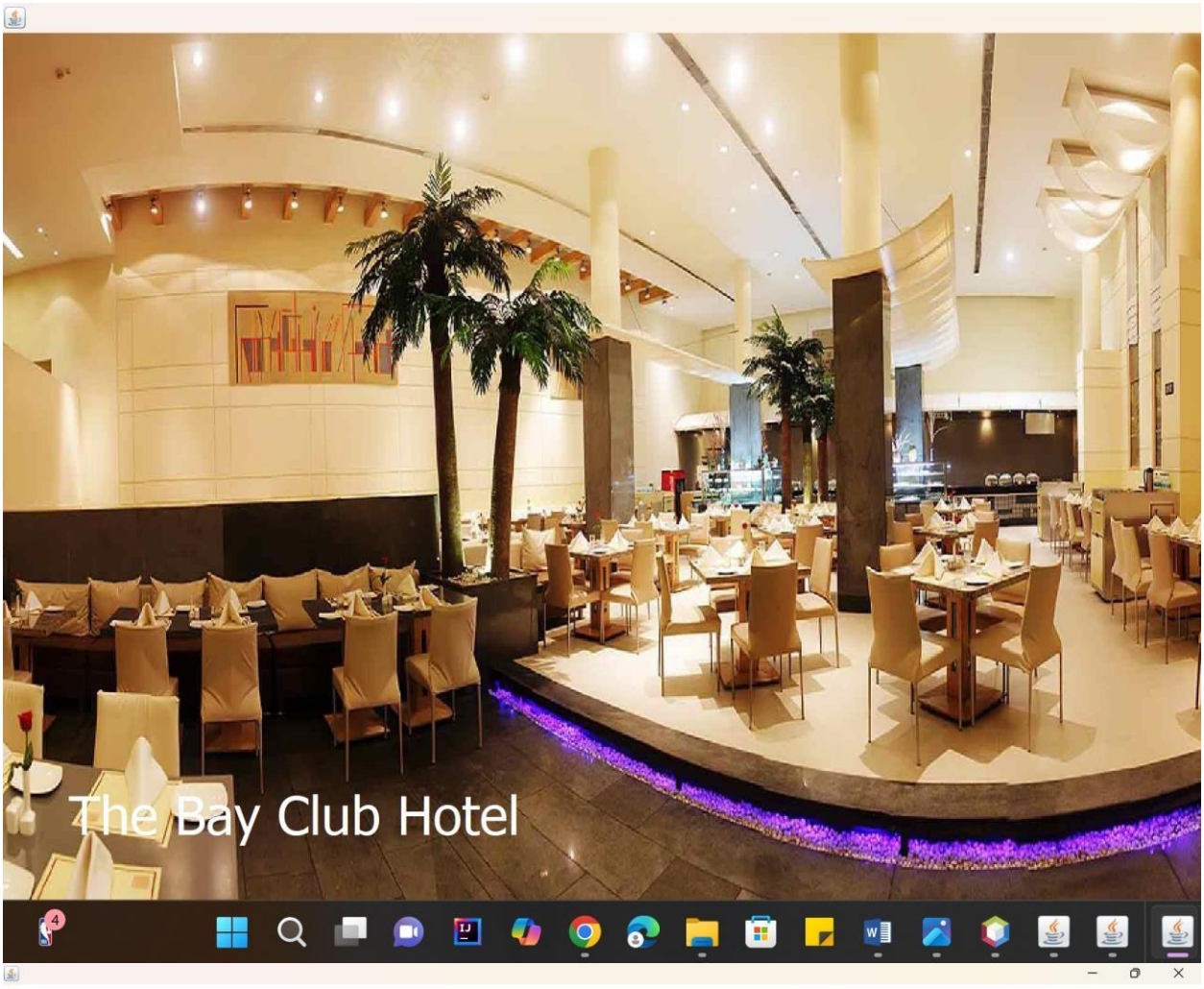




Hotels: Showing the information about hotels.







VIEW BOOKED HOTEL DETAILS

Username :

Name :

Number of Persons :

Number of Days :

AC / Non-AC :

Food Included (Yes/No) :

ID :

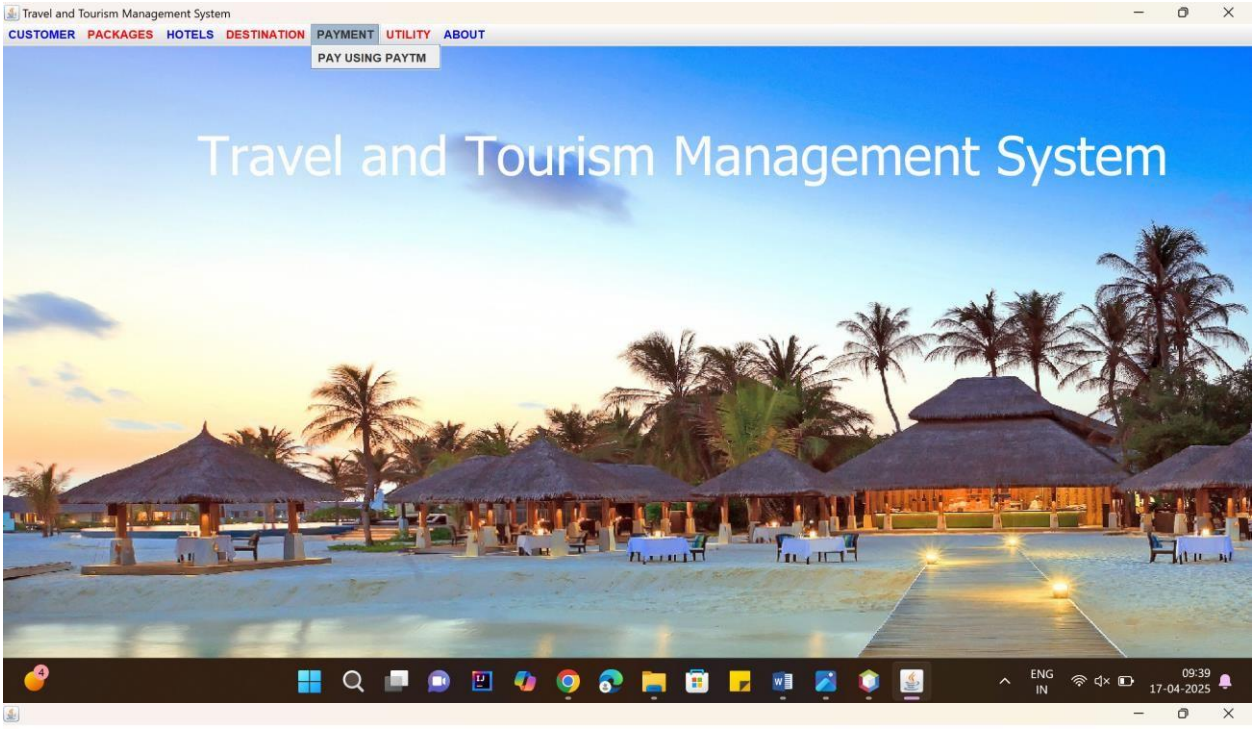
Number :

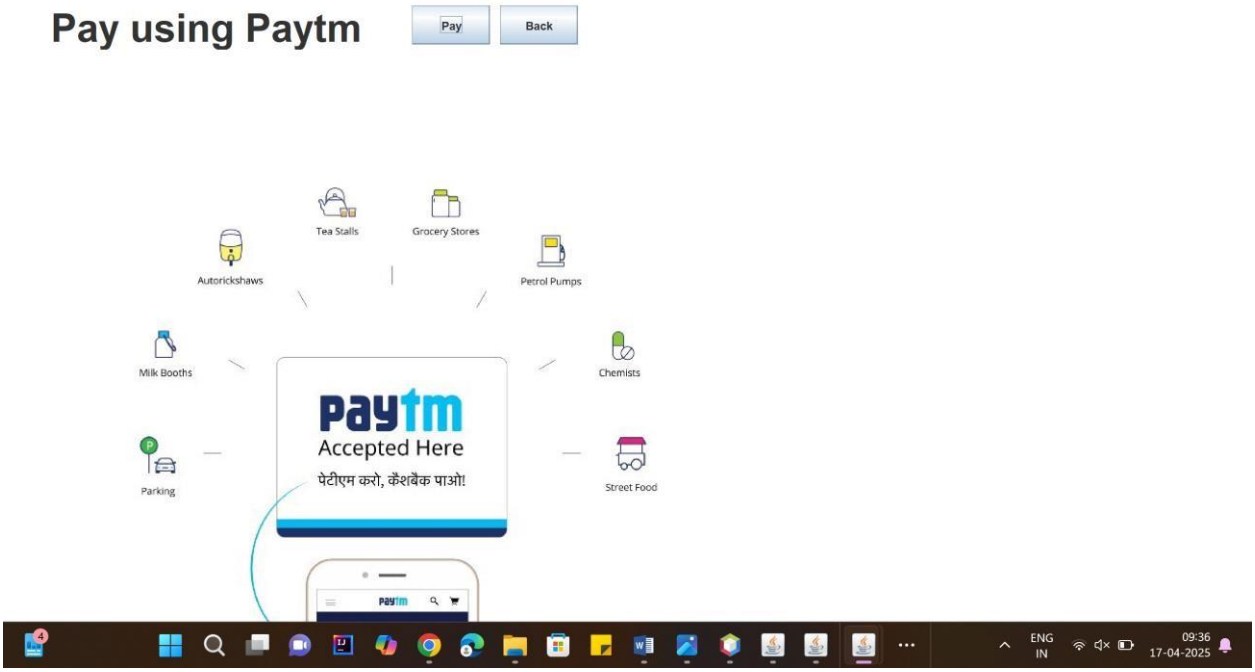
Phone :

Cost :

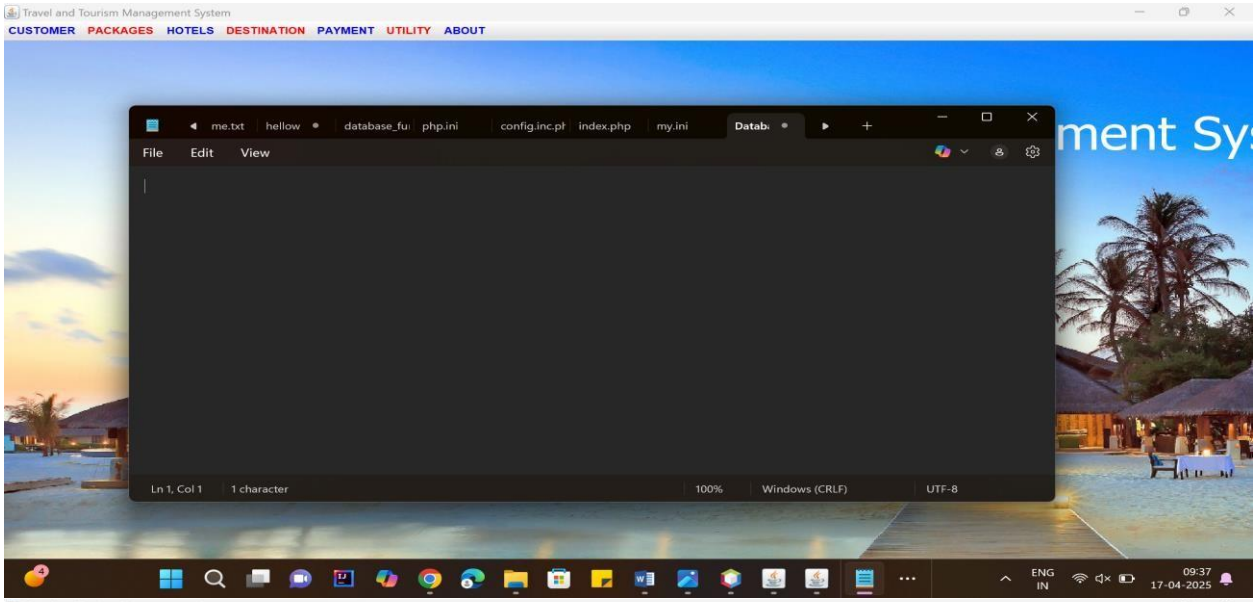
Back

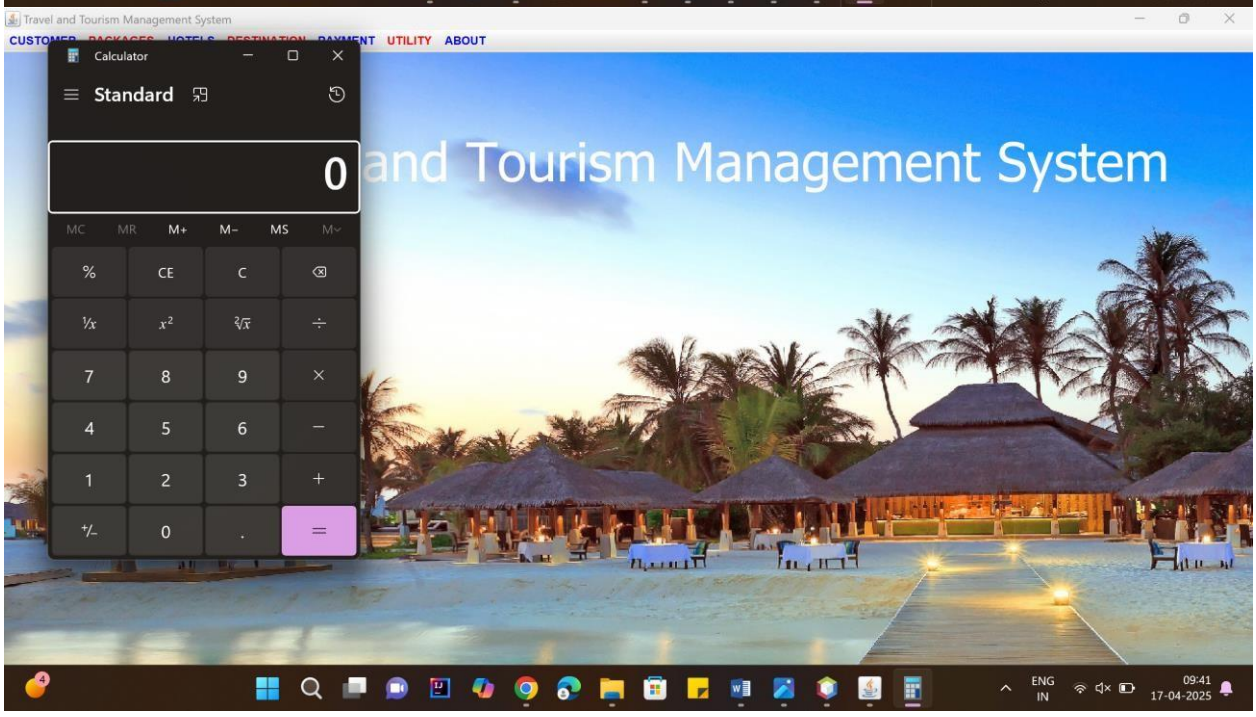
Payment: Do payment via paytm.



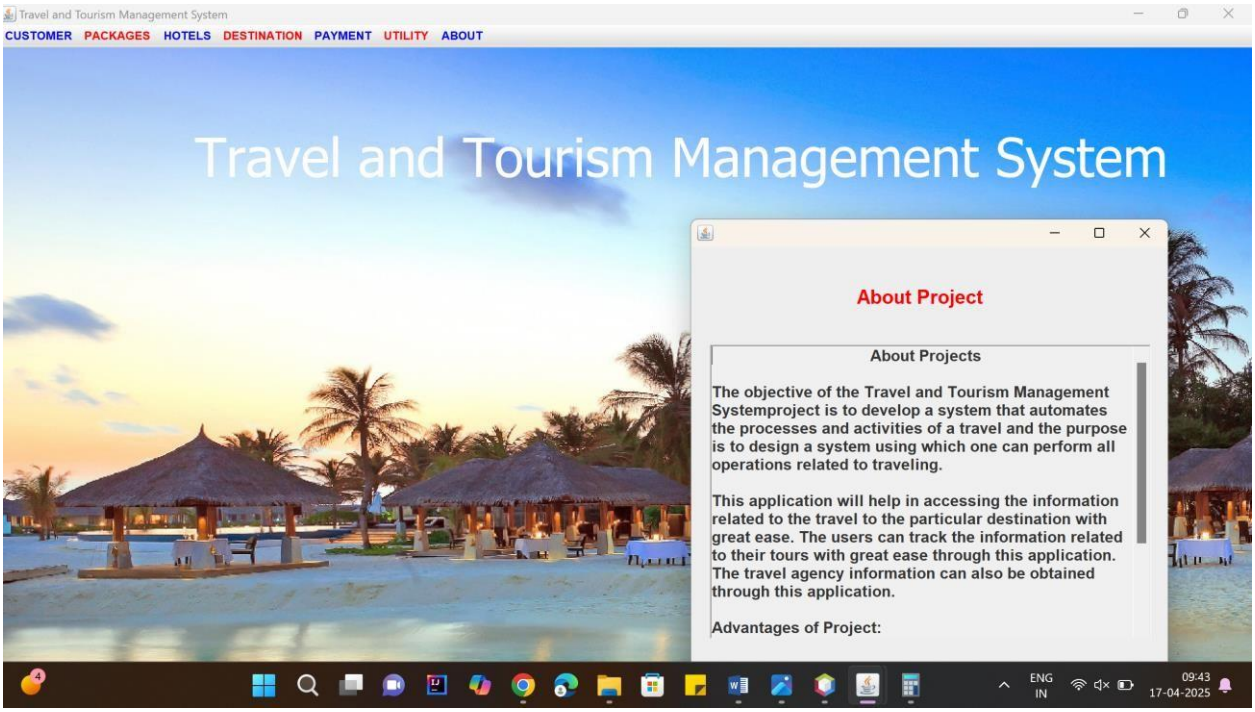


Utility: use notepad and calculator function.





About: About the information Travel and Tourism Management System.



4.2 Limitations:

1. Scalability Issues

- Performance Under Load: If the system is not designed to scale effectively, it may struggle to handle peak traffic times, such as holidays or special events, leading to slow response times and user frustration.
- Data Management: As user data and transaction volumes increase, managing and optimizing database performance can become challenging.

2. Integration Challenges

- Third-Party Services: Integrating with external services (e.g., payment gateways, travel APIs, or mapping services) can lead to dependencies that may affect system stability, performance, or security.
- Legacy Systems: Existing systems or databases may need to be integrated, which can introduce compatibility issues or data silos.

3. User Experience Limitations

- Accessibility: Despite best efforts, the system may not fully meet accessibility standards, making it difficult for users with disabilities to navigate and use the platform.
- User Training: Users may require training, especially if the interface is complex or no intuitive, which can limit adoption rates.

4.3 Future Enhancement:

1. Personalization and AI Integration

- **Smart Recommendations**: Use machine learning algorithms to analyze user behavior and preferences to provide personalized travel recommendations, accommodations, and itineraries.
- **Chabot's and Virtual Assistants**: Implement AI-driven chatbots to assist users in realtime, answering queries, making bookings, and providing support 24/7.

2. Mobile App Development

- **Native Applications**: Develop dedicated mobile applications for iOS and Android to provide a seamless user experience on mobile devices, including offline access and push notifications for deals and alerts.
- **Augmented Reality Features**: Incorporate AR technology for immersive user experiences, such as virtual tours or location-based information on landmarks.

3. Enhanced Payment Solutions

- **Multiple Payment Options**: Integrate various payment methods (cryptocurrency, buynow-pay-later services, mobile wallets) to cater to diverse user preferences and enhance convenience.
- **Payment Security Features**: Implement additional security features like biometric authentication or two-factor authentication for secure transactions.

4. Community and Social Features

- **User Reviews and Ratings**: Enhance review systems by adding verification for users (like booking history) and implementing social sharing features to encourage community building.
- **Itinerary Sharing**: Allow users to share travel itineraries with friends and family, promoting social planning and engagement.

Bibliography

1. Buhalis, D., & Law, R. (2008). **Progress in Information Technology and Tourism Management: 20 Years on and 10 Years After the Internet—The State of eTourism Research**. *Tourism Management*, 29(4), 609-623. https://doi.org/10.1016/j.tourman.2008.01.005
2. Chen, J., & Rahman, I. (2018). **Tourism Management: An Introduction**. In *Managing Tourism and Hospitality Services* (pp. 1-21). Routledge. https://doi.org/10.4324/9781351279974
3. Cooper, C., Fletcher, J., Gilbert, D., & Wanhill, S. (2005). **Tourism: Principles and Practice** (3rd ed.). Pearson Education Limited.
4. Gunter, U. (2016). **Applying Integrated IT Systems in the Tourism Sector**. *Journal of Tourism Management Research*, 3(1), 10-20. https://doi.org/10.2139/ssrn.2866582
5. Ivanov, S. H., & Zhecheva, Y. (2012). **The Role of Information Technology in Tourism Management: Possibilities and Challenges**. *Tourism & Management Studies*, 8(1), 55-
   65.
6. Komppula, R. (2014). **The Role of Technology in a Tourism Management System**. *Tourism Management*, 39, 149-157. https://doi.org/10.1016/j.tourman.2013.06.013
7. Neidhardt, J. (2021). **Trends and Innovations in Tourism Management**. In *Sustainable Tourism Management: A Global Perspective* (pp. 45-67). Emerald Publishing Limited. https://doi.org/10.1108/9781786356890-010

ANNEXTURE: Sample Code:

```
package travel.management.system;

import java.awt.*; import
javax.swing.*;
import java.awt.event.*;


public class Home extends JFrame{
String username;     public static void
main(String[] args) {
      new Home("").setVisible(true);
   }

   public Home(String username) {
      super("Travel and Tourism Management System");
this.username = username;
      setForeground(Color.CYAN);
      setLayout(null);


      ImageIcon i1 = new
ImageIcon(ClassLoader.getSystemResource("travel/management/system/icons/home.jpg"));
      Image i2 = i1.getImage().getScaledInstance(1950, 1000,Image.SCALE_DEFAULT);
      ImageIcon i3 = new ImageIcon(i2);
 JLabel NewLabel = new JLabel(i3);  NewLabel.setBounds(0, 0,
1950, 1000);
      add(NewLabel);

      JLabel l1 = new JLabel("Travel and Tourism Management System");
           l1.setForeground(Color.WHITE);
      l1.setFont(new Font("Tahoma", Font.PLAIN, 55));
           l1.setBounds(500, 60, 1000, 100);
           NewLabel.add(l1);


      JMenuBar menuBar = new JMenuBar();
           setJMenuBar(menuBar);

      JMenu m1 = new JMenu("CUSTOMER");
      m1.setForeground(Color.BLUE);
           menuBar.add(m1);
```

```
        JMenuItem mi1 = new JMenuItem("ADD CUSTOMER");
            m1.add(mi1);

        JMenuItem mi2 = new JMenuItem("UPDATE CUSTOMER DETAIL");
            m1.add(mi2);

        JMenuItem mi3 = new JMenuItem("VIEW CUSTOMER DETAILS");
            m1.add(mi3);

        JMenuItem mi4 = new JMenuItem("DELETE CUSTOMER DETAILS");
            m1.add(mi4);

        mi1.addActionListener(new ActionListener(){
           public void actionPerformed(ActionEvent ae){
try{
                new AddCustomer(username).setVisible(true);
            }catch(Exception e ){}
           }
             });

        mi2.addActionListener(new ActionListener(){
           public void actionPerformed(ActionEvent ae){
try{
                new UpdateCustomer(username).setVisible(true);
            }catch(Exception e ){}
           }
             });

        mi3.addActionListener(new ActionListener(){
           public void actionPerformed(ActionEvent ae){
try{
                new ViewCustomers().setVisible(true);
            }catch(Exception e ){}
           }
             });

        mi4.addActionListener(new ActionListener(){
           public void actionPerformed(ActionEvent ae){
try{
                new DeleteCustomer().setVisible(true);
            }catch(Exception e ){}
           }
             });
```

```
        JMenu m2 = new JMenu("PACKAGES");
      m2.setForeground(Color.RED);
        menuBar.add(m2);

      JMenuItem mi6 = new JMenuItem("CHECK PACKAGE");
        m2.add(mi6);

      JMenuItem mi7 = new JMenuItem("BOOK PACKAGE");
        m2.add(mi7);

      JMenuItem mi5 = new JMenuItem("VIEW PACKAGE");
        m2.add(mi5);


      mi6.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae){
try{
            new CheckPackage().setVisible(true);
          }catch(Exception e ){}
        }
          });


      mi7.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae){
try{
            new BookPackage(username).setVisible(true);
          }catch(Exception e ){}
        }
          });


          mi5.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae){
try{
            new ViewPackage(username).setVisible(true);
          }catch(Exception e ){}

        }
          });


      JMenu m3 = new JMenu("HOTELS");
```

```
        m3.setForeground(Color.BLUE);
            menuBar.add(m3);

        JMenuItem mi8 = new JMenuItem("BOOK HOTELS");
            m3.add(mi8);

        JMenuItem mi9 = new JMenuItem("VIEW HOTELS");
            m3.add(mi9);

        JMenuItem mi10 = new JMenuItem("VIEW BOOKED HOTEL");
                                                    m3.add(mi10);

        mi8.addActionListener(new ActionListener(){
public void actionPerformed(ActionEvent ae){
              new BookHotel(username).setVisible(true);
           }
             });



             mi9.addActionListener(new ActionListener(){
           public void actionPerformed(ActionEvent ae){
try{
                 new CheckHotels().setVisible(true);
              }catch(Exception e ){}
           }
             });

        mi10.addActionListener(new ActionListener(){
           public void actionPerformed(ActionEvent ae){
try{
                 new ViewBookedHotel(username).setVisible(true);
              }catch(Exception e ){}
           }
             });

        JMenu m4 = new JMenu("DESTINATION");
        m4.setForeground(Color.RED);
            menuBar.add(m4);

        JMenuItem mi11 = new JMenuItem("DESTINATION");
            m4.add(mi11);
```

```
      mi11.addActionListener(new    ActionListener(){
public    void    actionPerformed(ActionEvent    ae){
new Destination().setVisible(true);
         }
           });


      JMenu m5 = new JMenu("PAYMENT");
      m5.setForeground(Color.BLUE);
           menuBar.add(m5);

      JMenuItem mi12 = new JMenuItem("PAY USING PAYTM");
           m5.add(mi12);

      mi12.addActionListener(new ActionListener(){
         public void actionPerformed(ActionEvent ae){
            new Payment().setVisible(true);
         }
           });
```