EAST WEST UNIVERSITY

Mini Project Report

Course Name: Information System Analysis and Design

Course Code: CSE347

Semester: Summer – 2025

Section: 05

Project Title:

Travel Touch: A Comprehensive Tourist Management System

Submitted To:

MD Sabbir Hossain

Lecturer

Department of Computer Science and Engineering

East West University

Submitted By (Group – 03) :

Name	Student ID
Hemary Ahmed	2022-3-60-008
Tasnia Tabassum Azra	2022-3-60-084
Mujahida Nazmoon Naher	2022-2-60-099
Md. Danish Ahmed	2020-2-60-085

Date of Submission: 03 September, 2025

1.Introduction

1.1 Purpose

The purpose of this project is to describe the requirements for the **Tourist Management System (Travel Touch).** This system is created to bring tourists, agents, and admins together on one platform. Tourists will be able to explore different destinations, book trips, and share feedback about their experiences. Agents will be responsible for adding new packages, uploading hotel and flight details, and promoting destinations. Admins will manage the whole system, check user accounts, approve listings, and make sure that everything is safe and reliable.

1.2 Scope

This system will mainly serve three types of users: tourists, agents, and admins.

- **Tourists** can create accounts, search for places to visit, book trips, make payments online, and provide ratings or complaints about their experience.
- **Agents** will create and manage travel packages, upload information about hotels and flights, and promote new or special destinations to attract more tourists.
- **Admins** will monitor all activities in the system. They will approve or reject new listings, manage user accounts, review transactions, and take action if any problems are reported.

With these features, the portal will make traveling more convenient, safe, and organized. Secure login, verified listings, reliable payment methods, and detailed booking memos will help improve the overall travel experience for everyone.

1.3 Intended Audience and Reading Suggestions

This document is written for different groups of people who will use or work with the system:

- **Admins**, who will manage and monitor the system to keep it safe and running smoothly.
- **Agents**, who will use the portal to create packages and manage their services.
- **Tourists**, who will use the system to find destinations, book trips, and give feedback.
- **Developers and Designers**, who will build the system and design its interface based on the requirements written here.
- **Testers**, who will check the system step by step to make sure all features are working properly.

2. Usage Scenario

East-West Tours & Travels is building an online Tourist Management System (Travel Touch) to make traveling easier for tourists and more organized for agents and admins.

When a new user signs up as a tourist, they provide their name, email, phone number, and password. A verification code is sent to their email, and once entered, the account becomes active. For security, tourists cannot change their email or phone number after verification.

On the login page, tourists see input fields for email/phone and password along with a login button. If they forget their password, they can reset it using their verified contact details. After logging in, tourists access their dashboard where they can explore destinations, check hotels, trip prices, and package options. They can add trips to a cart, pick dates, and pay through debit card, credit card, or bank transfer. Once payment is complete, a unique transaction ID is generated. After traveling, tourists can rate their experience, leave feedback, or file complaints if needed.

Agents also sign up, but their registration is stricter. They must provide NID number, bank account details, and a passport-size photo. After admin verification, agents can log in to upload and manage listings. They add new destinations, provide hotel and flight information, set package prices, and promote special offers. Each agent can upload a limited number of packages per month, and they earn 10–15% commission for every successful booking.

Admins use predefined credentials to log in. They have full control over the system: approving or rejecting agent registrations, verifying new listings, blocking suspicious accounts, and monitoring transactions. Admins also check complaints and can take action against agents with fake or poor listings. Monthly reports are generated to track performance and keep the system secure and reliable.

The system ensures that each group of users has their own secure role. Tourists enjoy an easy way to book trusted trips, agents get a platform to promote and manage their packages, and admins maintain safety, reliability, and quality. Overall, Travel Touch creates a smooth, organized, and trustworthy digital travel experience for everyone.

3. System Features and Requirements

System features are the key things the system can do for its users. They describe the services or tools that are built into the system to make it useful. In your Tourist Management System (Travel Touch), features are what tourists, agents, and admins can actually use.

System requirements are the rules and conditions that the system must follow to make those features work properly.

3.1 Functional Requirements

Stakeholder	Stakeholder Goal	Functional Requirement
Tourist	We want to create and use a personal account safely.	The system shall allow tourists to create an account with name, email, phone, and password, and verify it through email/phone before activation.
	We want secure access to my account.	The system shall require tourists to log in with their email/phone and password.
	We want to explore available destinations.	The system shall allow tourists to browse verified destinations with details such as hotels, packages, and prices.
	We want to book trips easily.	The system shall allow tourists to select trips, add them to a cart, choose dates, and confirm booking after payment.
	We want safe payment options.	The system shall support payments through credit/debit cards and bank transfers, generating a unique transaction code.
	We want to track my bookings.	The system shall generate a detailed memo with hotel, flight, and trip information after each booking.
	We want to share my experience.	The system shall allow tourists to rate destinations and submit complaints about poor service.
Agent	We want to join the system securely.	The system shall allow agents to create an account with personal details (NID, bank info, and photo), verified by the admin before approval.
	We want to manage my listings.	The system shall allow agents to upload and edit destinations, hotels, flights, and package prices.
	We want to promote my destinations.	The system shall allow agents to highlight or promote listings with offers and visibility boosts.
	We want to track my performance.	The system shall allow agents to view bookings made through their packages and calculate commissions (10–15%).

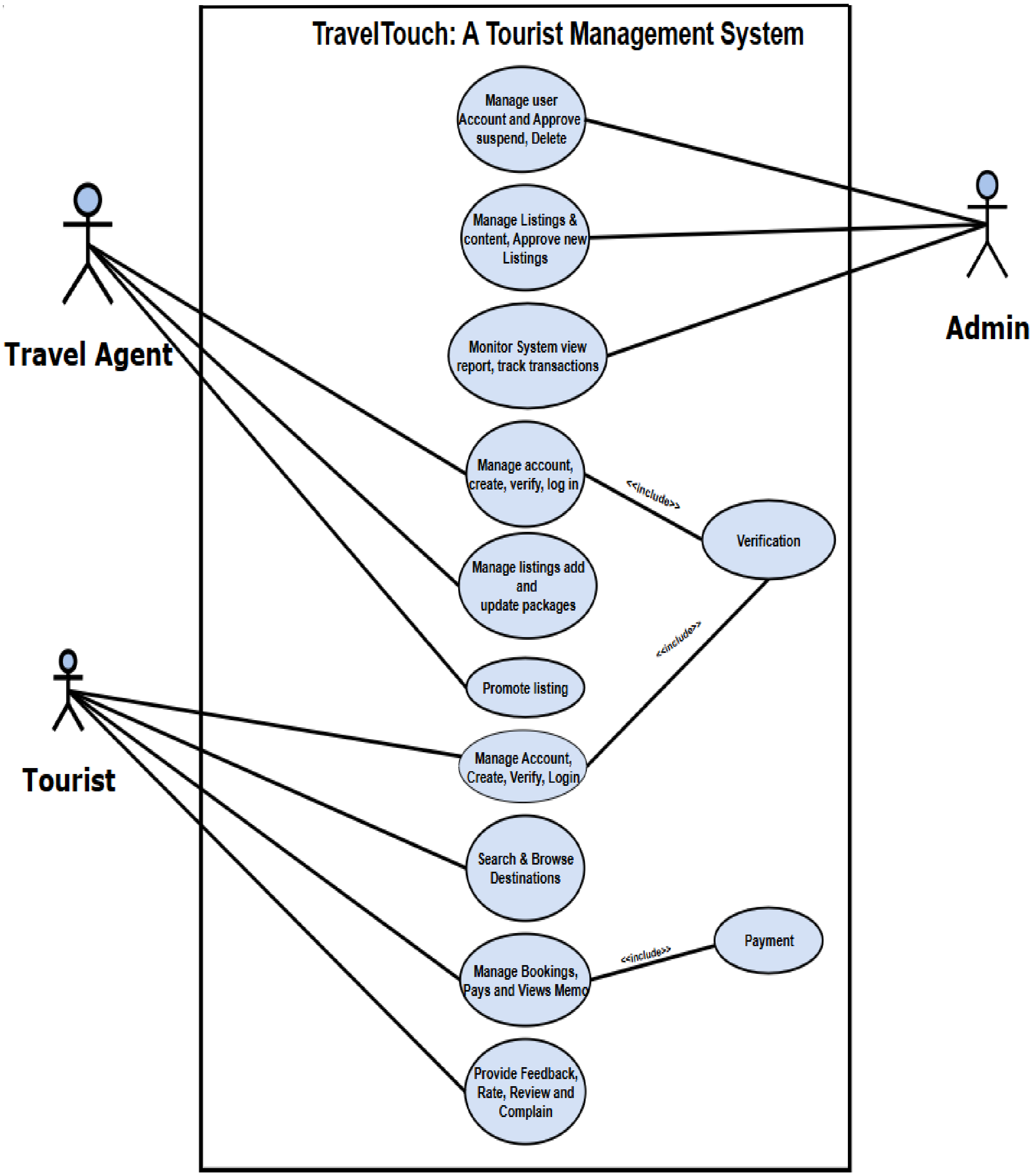
Admin	We want to oversee the entire system.	The system shall allow admins to log in with predefined credentials to access the admin panel.
	We want to review new accounts.	The system shall allow admins to approve or reject tourist and agent registrations.
	We want to ensure quality and trust.	The system shall allow admins to verify and approve listings before they appear for tourists.
	We want to manage users fairly.	The system shall allow admins to block or suspend suspicious accounts (tourists or agents).
	We want to monitor activity.	The system shall allow admins to track bookings, payments, complaints, and feedback.
	We want to generate system reports.	The system shall allow admins to create monthly reports for performance monitoring.

3.2 Non - Functional Requirements

Stakeholder	**Stakeholder Goal**	**Non-Functional Requirement**
Tourist	We want to log in and book trips from different devices.	The system shall support secure and simultaneous logins from multiple devices.
	We want smooth booking even when many people are online.	The system shall ensure fast response times and reliable performance during peak booking periods.
	We want the system to be simple and easy to use.	The system shall provide a clean, user-friendly, and intuitive interface for all users.
	We want my ratings and complaints to stay private.	The system shall guarantee anonymity for tourist feedback and complaints.
Agent	We want quick uploads of packages and listings.	The system shall allow fast uploading and processing of new destinations, hotel, and flight details.
	We want reliable commission tracking.	The system shall ensure accurate and timely calculation of agent commissions.
Admin	We want to keep user data safe.	The system shall ensure secure storage, encryption, and proper handling of all user information.
	We want to handle many users at once.	The system shall be scalable to support large numbers of simultaneous tourists and agents.
	We want full control and monitoring of activities.	The system shall provide admins with audit logs and monitoring tools for all actions.

4. Use Case Diagram

A use case diagram is a visual blueprint that shows a system's main functionalities, who uses them (actors), and how those users interact with the system. It's a high-level, top-down view used in the Unified Modeling Language (UML) to understand and communicate user needs and system goals without getting bogged down in implementation details.


TravelTouch: A Tourist Management System
Travel Agent
Tourist
Admin
Manage user Account and Approve suspend, Delete
Manage Listings & content, Approve new Listings
Monitor System view report, track transactions
Manage account, create, verify, log in
<<include>>
Verification
Manage listings add and update packages
<<include>>
Promote listing
Manage Account, Create, Verify, Login
Search & Browse Destinations
Manage Bookings, Pays and Views Memo
<<include>>
Payment
Provide Feedback, Rate, Review and Complain

4.1 Description of the Use Case Diagram

The diagram illustrates how different types of users—**Admins, Travel Agents, and Tourists**—interact with the travel management system. Each actor has specific roles and responsibilities within the system, ensuring smooth operation and user satisfaction.

4.1.1 Admin

Admins are responsible for overseeing and maintaining the entire system. Their tasks include:

- Managing user accounts, such as approving new registrations, suspending problematic accounts, or deleting inactive ones.
- Reviewing, approving, and managing travel-related listings or content submitted by travel agents.
- Monitoring system activity and tracking transactions to ensure security and reliability.

4.1.2 Travel Agent

Travel Agents act as service providers within the system. They:

- Create and manage their own accounts.
- Add, update, and maintain travel packages to attract customers.
- Promote their listings to reach more tourists.

4.1.3 Tourist

Tourists are the end-users who use the platform to plan and book their trips. They can:

- Create and manage personal accounts.
- Search for and browse travel destinations and packages.
- Book trips, make payments, and receive booking memos.
- Share their experiences by giving feedback, ratings, and reviews.

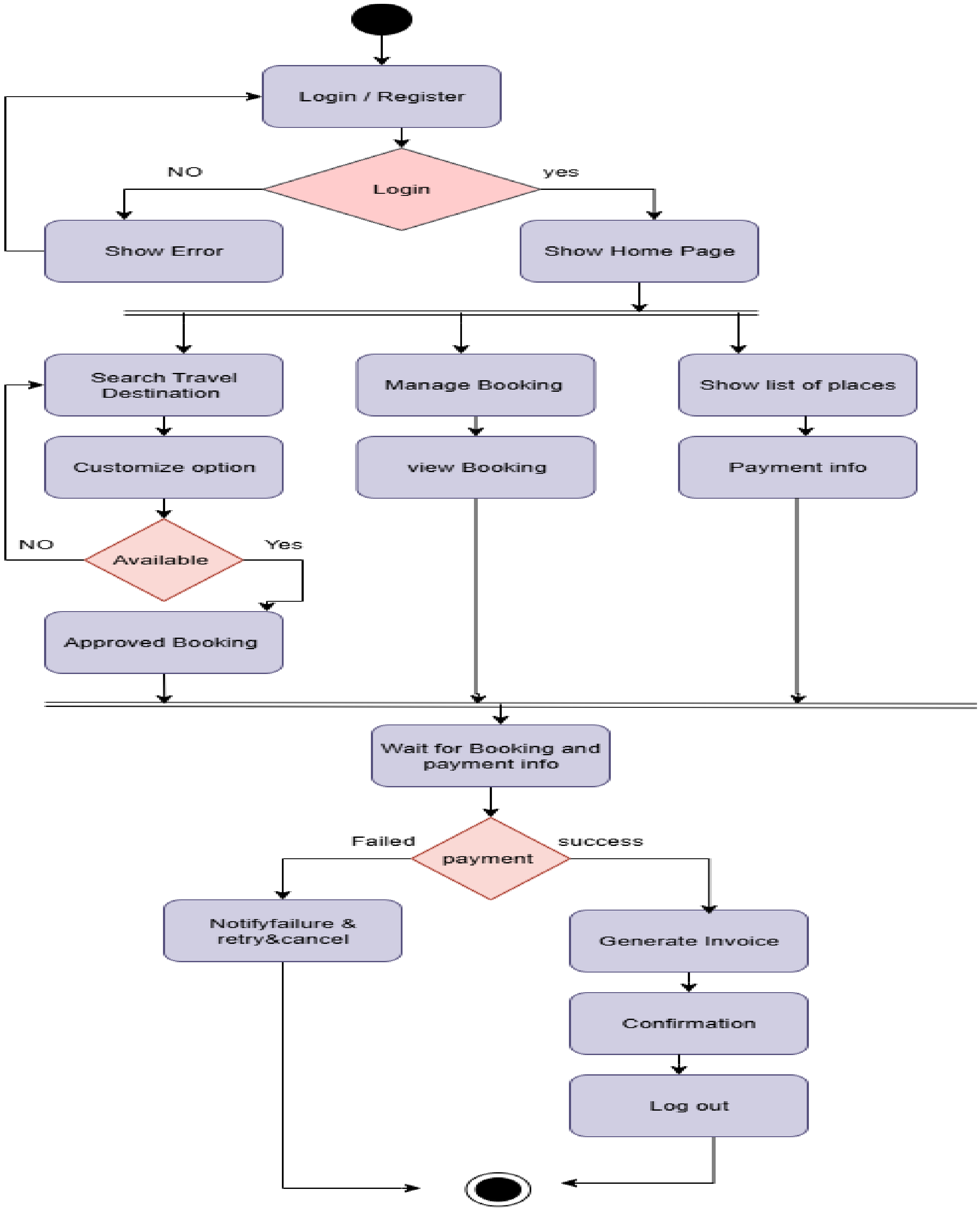
4.1.4 Additional Features

- **Verification**: Integrated into account management to ensure authenticity and security for all users.
- **Payment System**: Connected to booking management, providing safe and secure transaction processes.

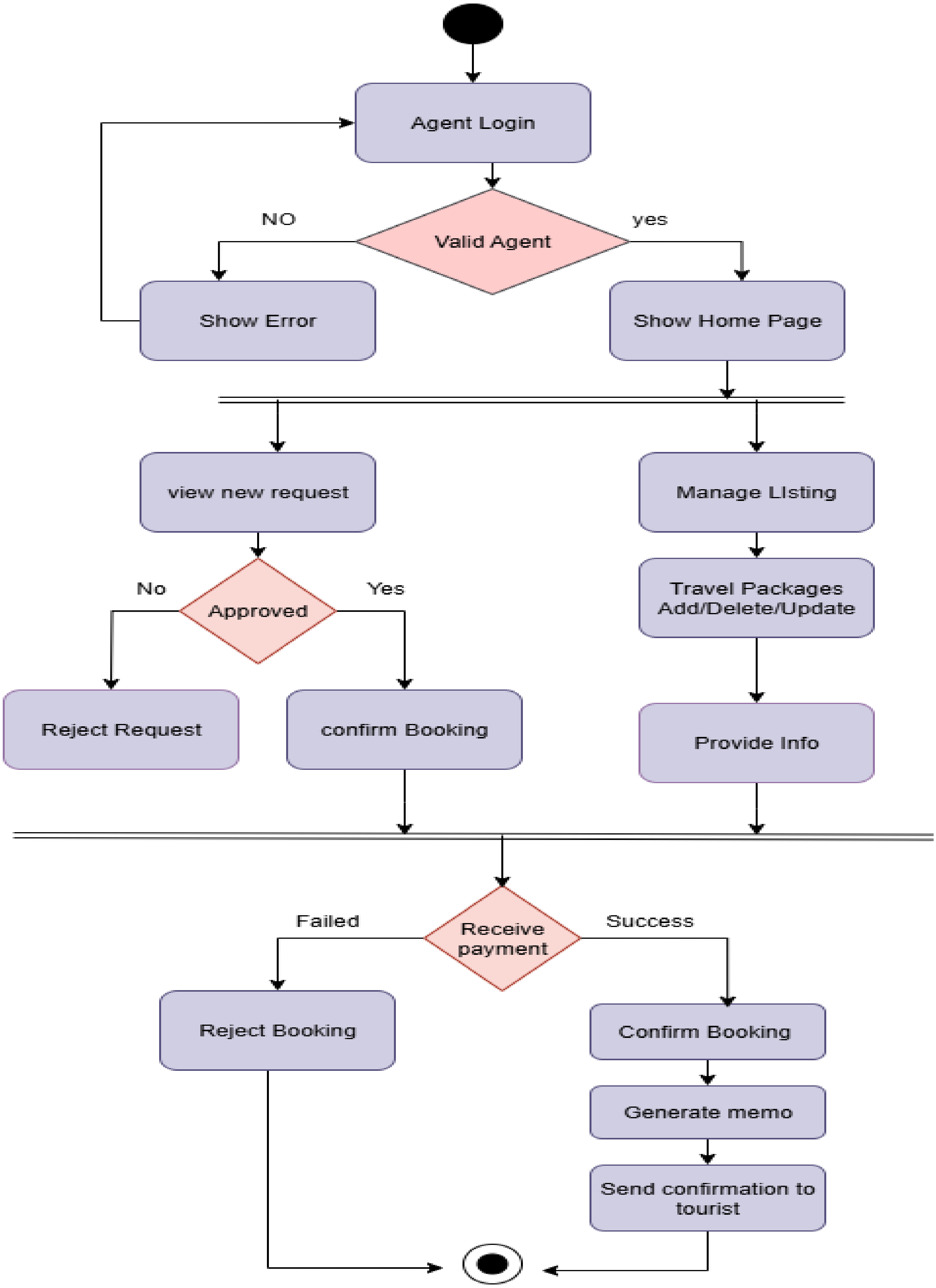
5. Activity Diagram

An activity diagram is a type of behavioral diagram in the Unified Modeling Language (UML) that visually represents the workflow of a system or process.

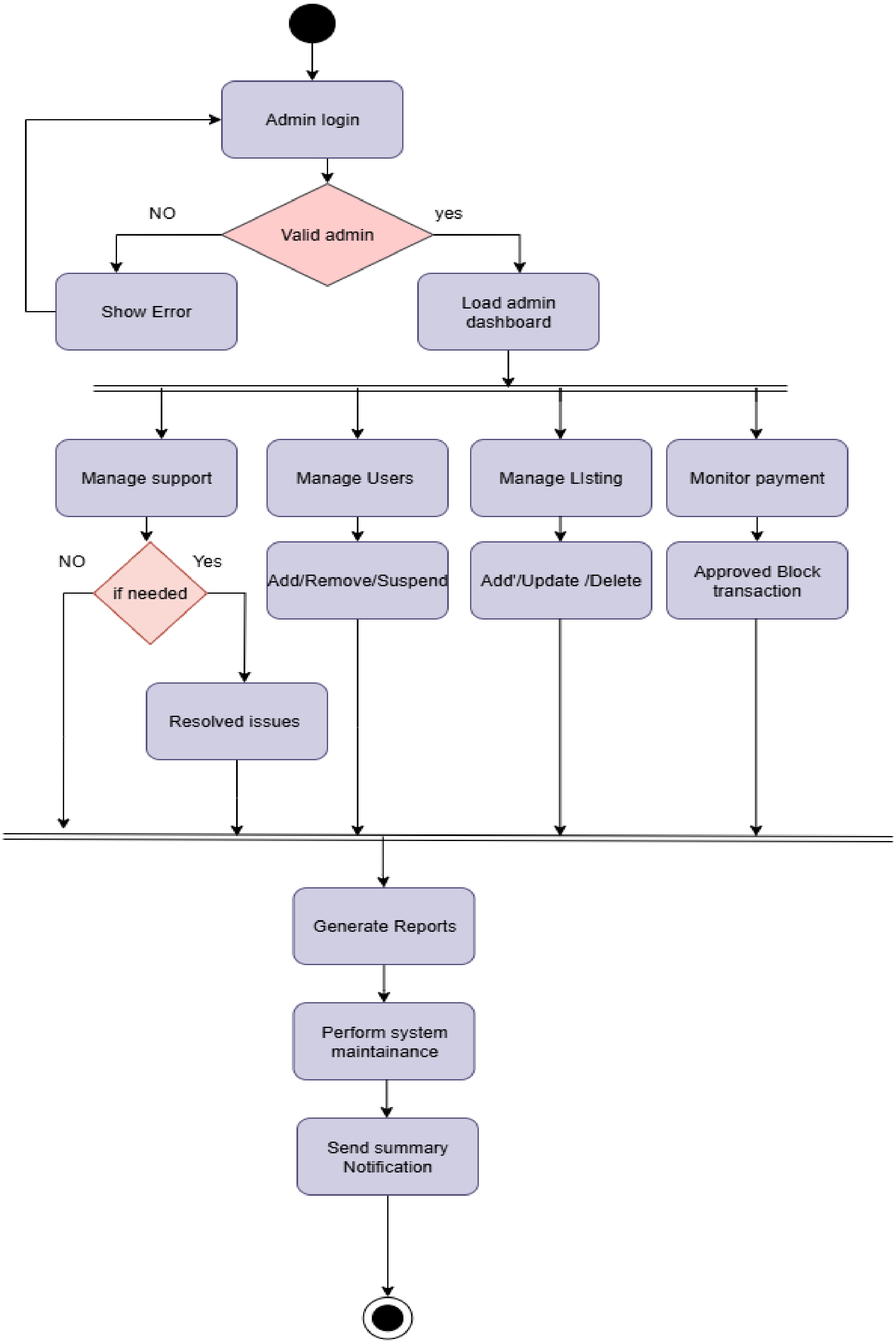
5.1 Activity Diagram for Tourist



5.2 Activity Diagram for Agent



5.3 Activity Diagram for Admin



• Tourist Activity Flow

A tourist starts by logging in or registering an account. If the login details are wrong, the system shows an error, but if correct, the tourist is taken to the home page. From there, they can search for destinations, check bookings, or see available places and payment details. When searching, tourists can adjust options, and if the destination is available, the booking gets approved. After that, they move to payment. If the payment fails, the system notifies them and lets them retry or cancel. If the payment works, the system creates an invoice, shows a confirmation, and the tourist can log out. This makes the whole process of searching, booking, and paying simple and organized.

• Agent Activity Flow

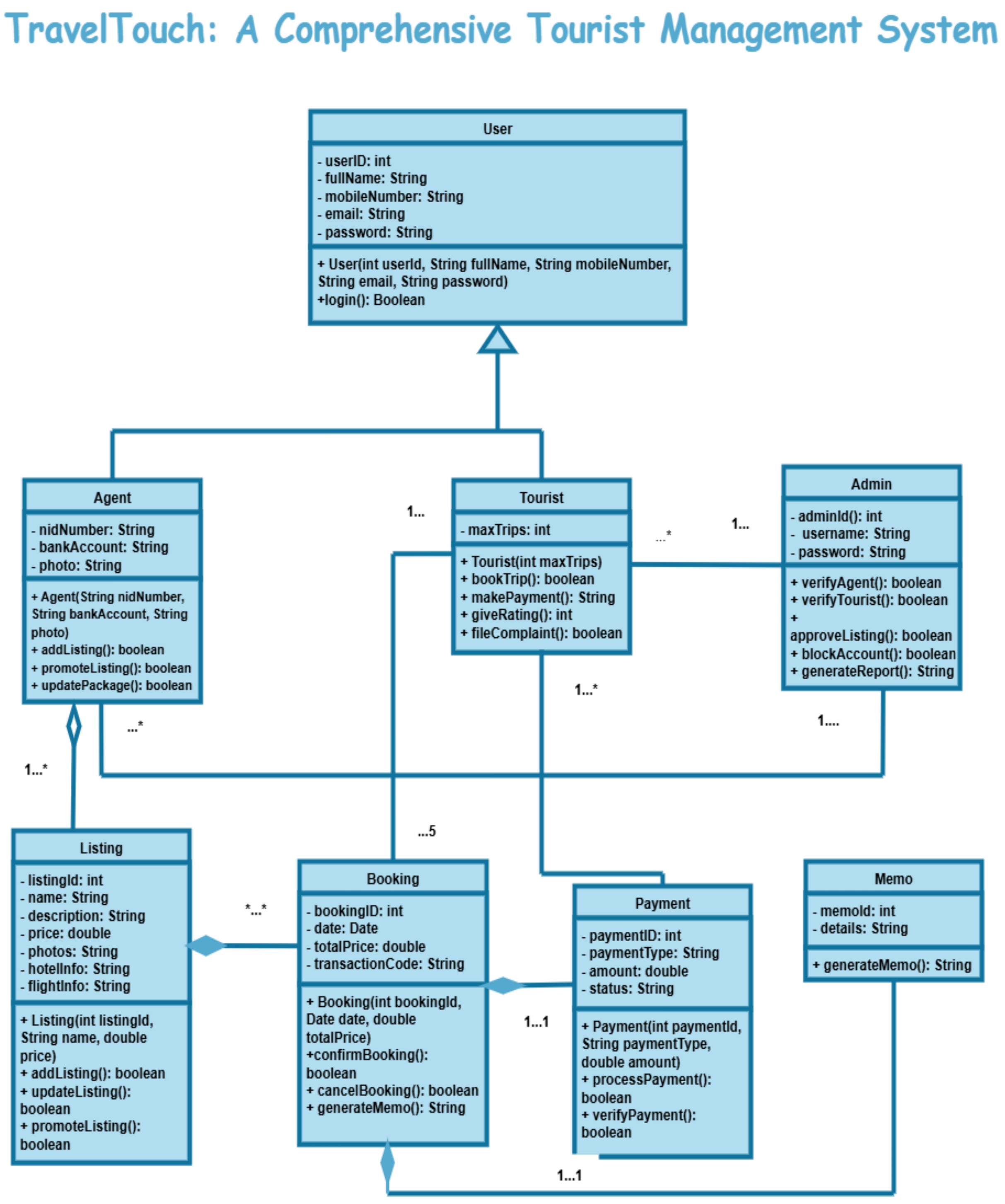
An agent begins by logging into the system. If the login is invalid, an error appears; if valid, they are directed to the home page. From there, agents can check new booking requests or update their travel listings. If a booking request comes in, the agent can approve or reject it. Approved bookings move forward for confirmation. Agents can also add, delete, or update details of travel packages. Once a booking is approved, payment is handled. If payment fails, the booking is rejected. If payment is successful, the agent confirms the booking, generates a memo, and sends confirmation to the tourist. This process helps agents manage their packages and keep tourists updated.

• Admin Activity Flow

The admin logs in using special credentials. If the login is wrong, an error is shown; if correct, the admin dashboard opens. From here, admins can handle system support, user accounts, listings, and payments. They can solve issues if needed, add or remove users, update or delete listings, and approve or block transactions. Admins also make sure that all accounts and listings are genuine and safe. At the end, admins generate reports, carry out system maintenance, and send summary notifications. This flow shows how admins control and monitor everything to keep the system reliable and secure.

6. Class Diagram

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.



This class diagram represents a Tourist Management System. It uses OOP concepts like: Classes, Attributes, Methods, Inheritance, Associations, Aggregations, Compositions.

There are total 8 classes. In this diagram the parent class is: User class and child classes are: Agent, Tourist and Admin.

- **User Class (Parent Class)**
  - Attributes: userID, fullName, mobileNumber, email, password
  - Constructor: User(int userId, String fullName, String mobileNumber, String email, String password + create account)
  - Methods: createAccount(), login()
  - Acts as the base class (or parent) for Admin, Agent and Tourist.
  - This means Agent, Tourist, and Admin inherit from User (they get the common features like account, login, etc.).
- **Agent Class (Child of User)**
  - Extra Attributes: nidNumber, bankAccount, photo
  - Constructor: Agent (String nidNumber, String bankAccount)
  - Methods:

  - addListing()

  - promoteListing()

  - updatePackage()

  - Role: Agents create and manage tourism packages/listings

- **Tourist Class (Child of User)**
  - Extra Attribute: maxTrips
  - Constructor: Tourist(int maxTrips)
  - Methods:

    - bookTrip()

    - makePayment()

    - giveRating()

    - fileComplaint()
  - Role: Tourists can book trips, pay, and give reviews.
- **Admin Class (Child of User)**
  - Extra Attributes: adminId, username, password
  - Methods:

    - verifyAgent()

    - verifyTourist()

    - approveListing()

    - blockAccount()

    - generateReport()
  - Role: Admin manages the system—approving, verifying, blocking accounts.

❖ **Listing Class**

- Attributes: listingId, name, description, price, photos, hotelInfo, flightInfo
- Constructor: Listing(int listingId, String Name, double Price)
- Methods:

  - addListing()

  - updateListing()

  - promoteListing()

- Represents tour packages created by Agents. One Agent can create many listings.

❖ **Booking Class**

- Attributes: bookingID, date, totalPrice, transactionCode
- Constructor: Booking(int bookingId, Date date, double totalPrice)
- Methods:

  - confirmBooking()

  - cancelBooking()

  - generateMemo()

- Represents when a Tourist books a Listing. One Tourist can have multiple Bookings.

❖ **Payment Class**

- Attributes: paymentID, paymentType, amount, status
- Consructor: Payment(int paymentId, String, paymentType, double amount)
- Methods:

  - processPayment()

  - verifyPayment()

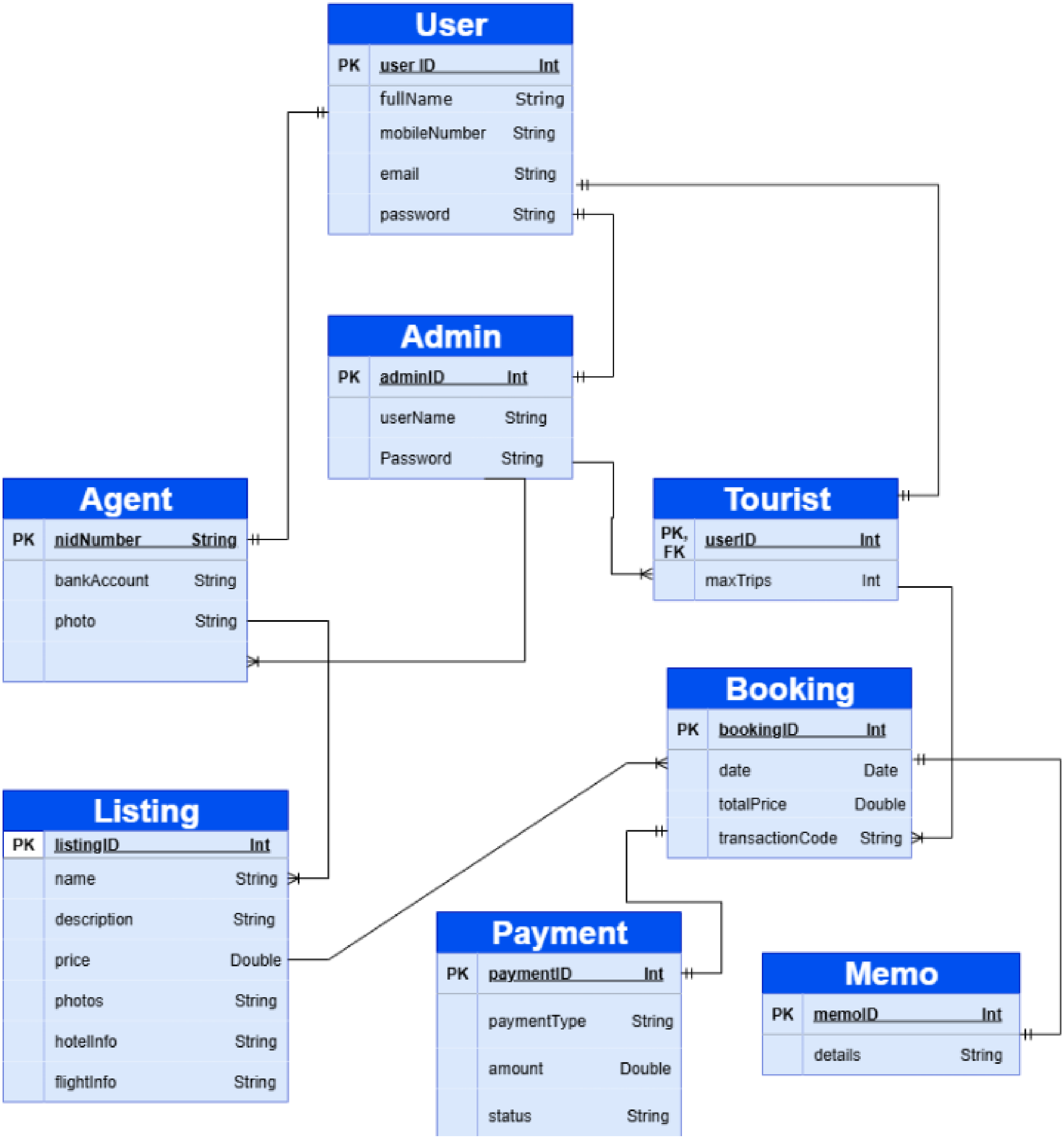
- Represents payments for bookings. Each Booking has one Payment.

- **Memo Class**
  - Attributes: memoId, details
  - Methods: generateMemo()
  - Represents receipts or memos generated after booking.

- **Relationships:**
  - **Inheritance:** User → Agent, Tourist.
  - **Association:**
    - ❑ Tourist ↔ Booking
    - ❑ Tourist ↔ Payment
    - ❑ Admin ↔ Agent
    - ❑ Admin ↔ Tourist
  - **Aggregation (◇):**
    - ❑ Agent ↔ Listing
  - **Composition (◆):**
    - ❑ Booking ↔ Memo
    - ❑ Booking ↔ Payment
    - ❑ Booking ↔ Memo

7. ER Diagram

An Entity-Relationship Diagram (ERD) is a visual blueprint for designing databases, illustrating the relationships between different entities (like "customers" or "products") and their attributes (properties) using specialized symbols.



The Entity–Relationship (ER) diagram of the Tourist Management System represents the main entities of the system and their relationships. It ensures proper organization of data related to tourists, agents, admins, listings, bookings, payments, and memos. The ER model is designed to maintain consistency, avoid redundancy, and provide smooth interaction between different system components.

Entities and Attributes

- **Users**

❑ Represents general user information for tourists, agents, and admins.

❑ Attributes: id, fullName, mobileNumber, email, password.

- **Tourists**

❑ Represents tourists who make bookings.

❑ Attributes: id, maxTrips, userID.

❑ Connected with the Users entity to access user information.

- **Agents**

❑ Represents agents who manage listings.

❑ Attributes: id, userID, nidNumber, bankAccount, photo.

❑ Linked with the Users entity for authentication and identity.

- **Admins**

❑ Represents administrators who verify agents, approve listings, and manage the system.

❑ Attributes: id, userID, adminID, username, password.

❑ Linked with Users for common account details.

- **Listings**

❑ Represents available trip packages or travel services.

❑ Attributes: id, agentID, name, description, price, photos, hotelInfo, flightInfo.

❑ Each listing is created and managed by an Agent.

- **Bookings**

❑ Represents tourist bookings for specific listings.

❑ Attributes: id, touristID, date, totalPrice, transactionCode, listingID.

❑ Connected to both Tourists and Listings.

- **Payments**

❑ Represents payment transactions for bookings.

❑ Attributes: id, paymentType, amount, status, bookingID.

❑ Linked to Bookings to track financial transactions.

- **Memos**

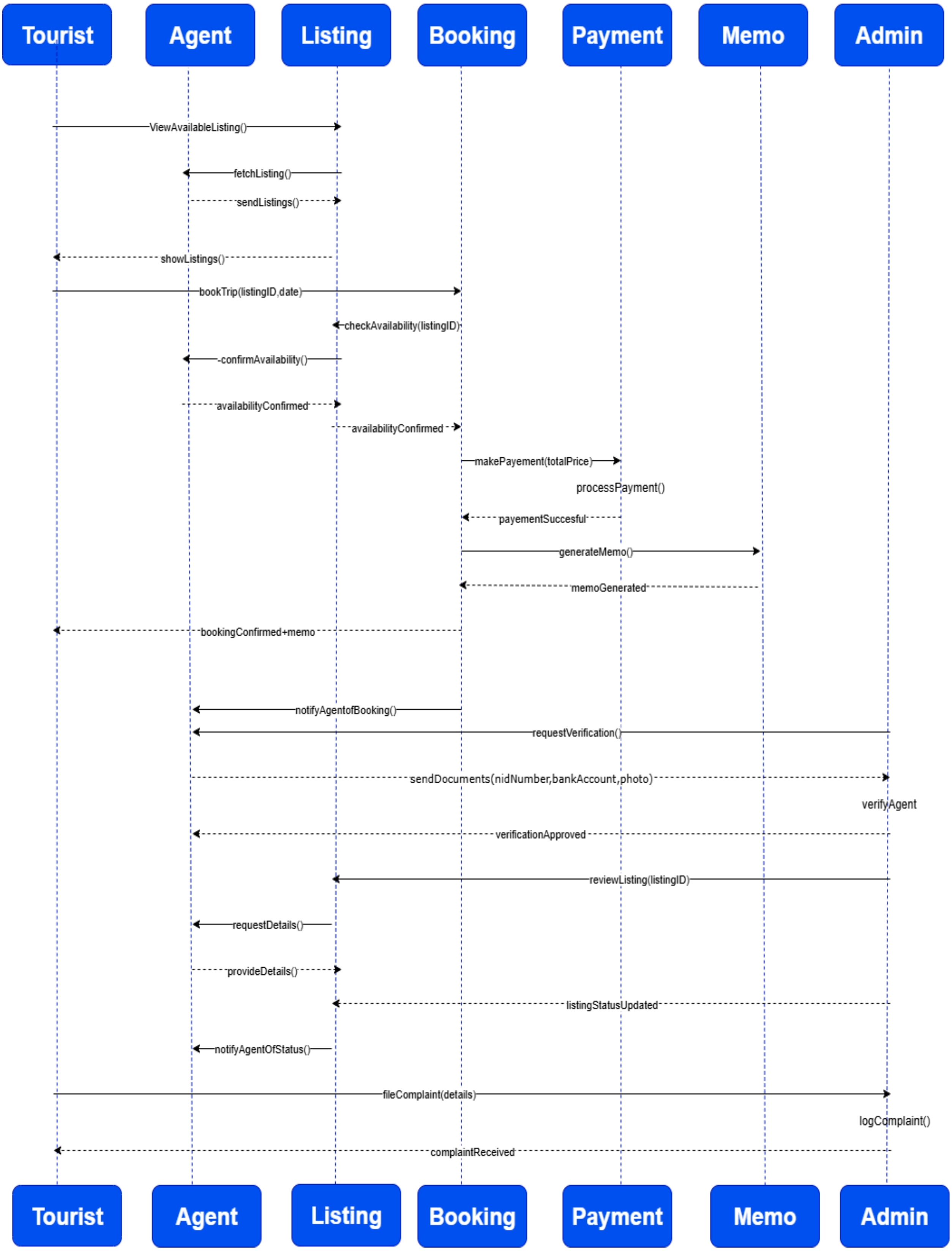
❑ Represents booking confirmation records or receipts.

❑ Attributes: id, bookingID, details.

❑ Directly connected to Bookings as proof of successful transactions.

8. Sequence Diagram

A sequence diagram is a type of UML diagram that shows how objects or parts of a system interact with each other step by step over time.



The sequence diagram illustrates the interaction between different actors and system components in the tourism booking process. It shows how a Tourist, Agent, Listing system, Booking system, Payment service, Memo generator, and Admin collaborate step by step.

7.1 Viewing Available Listings

- The Tourist initiates the process by requesting to view available listings.
- The Agent fetches listings from the Listing system and sends them back to the Tourist.
- The listings are then displayed to the Tourist.

7.2 Booking a Trip

- The Tourist selects a listing and makes a booking request with details such as listing ID and date.
- The Booking system checks the availability of the selected listing through the Listing system.
- If available, a confirmation is sent back to the Tourist.

7.3 Payment and Memo Generation

- Once availability is confirmed, the Tourist makes a payment by providing the total price.
- The Payment service processes the transaction and confirms payment success.
- A Memo is generated and linked with the booking, which is then sent to the Tourist as a booking confirmation.

7.4 Agent Notification and Verification

- The Agent is notified of the Tourist's booking.
- To validate the Agent, the system requests verification.
- The Agent provides documents (NID number, bank account, and photo).
- The Admin verifies these documents, and once approved, notifies the system.

7.5 Listing Review and Status Update

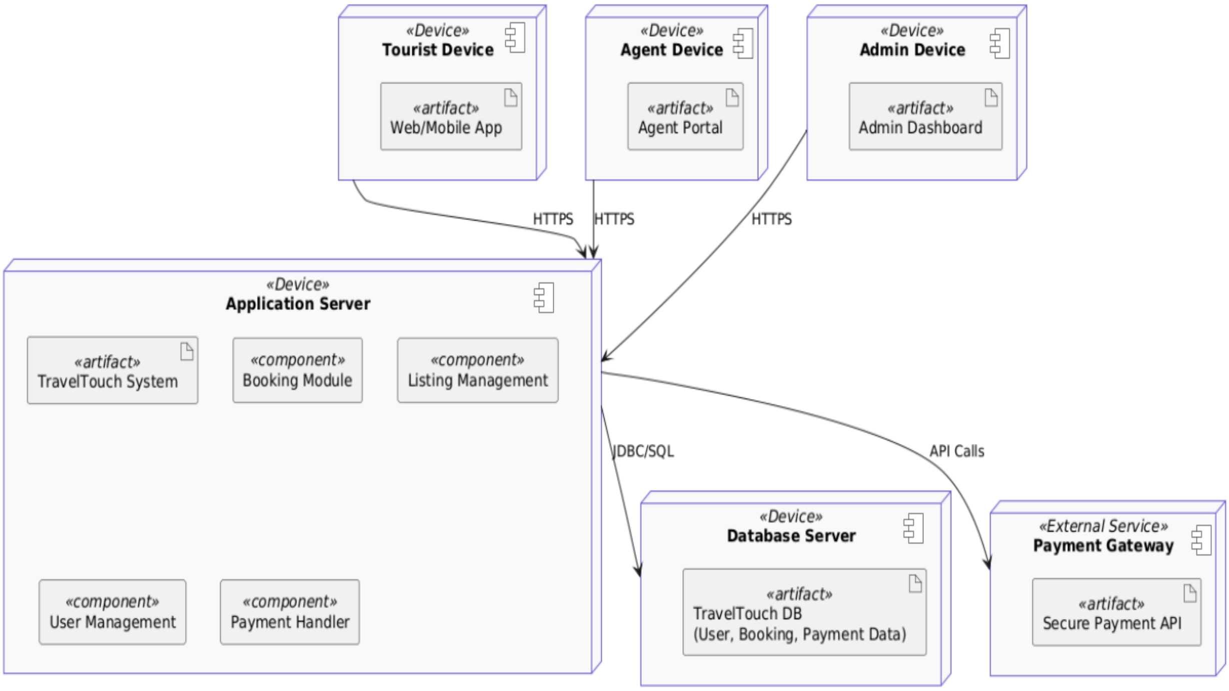
- The Booking system requests a review of the listing.
- The Listing system updates the listing status and notifies the Agent.

7.6 Complaint Management

- If the Tourist faces any issue, they can file a complaint with details.
- The Admin logs the complaint and ensures that the complaint is recorded in the system

9. Deployment Diagram

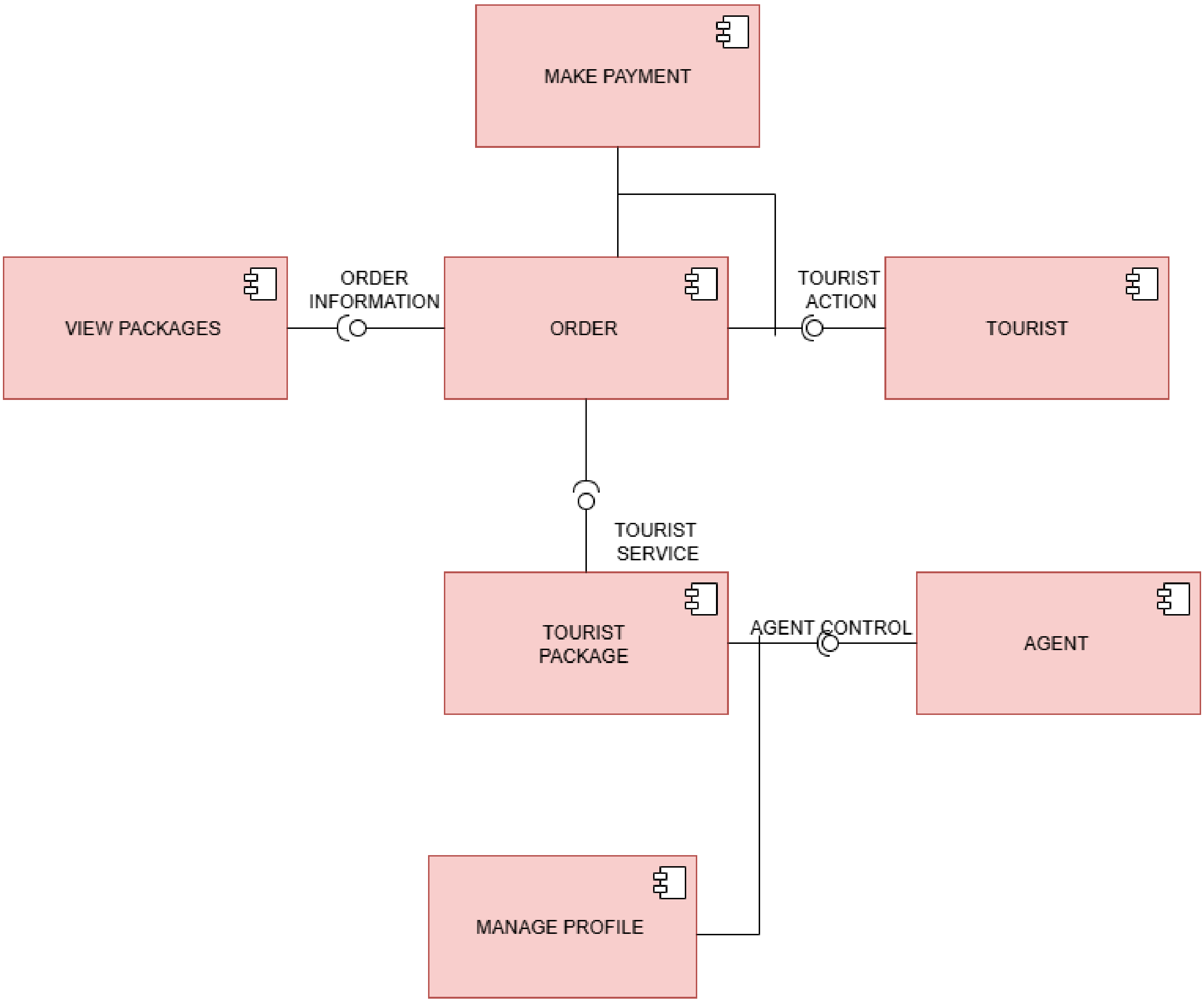
A deployment diagram allows you to illustrate how instances of software systems and/or containers in the static model are deployed on to the infrastructure within a given **deployment environment** (e.g. production, staging, development, etc). It's based upon a UML deployment diagram.



The deployment diagram shows how the Tourist Management System (TravelTouch) is set up and how different parts communicate. Tourists, agents, and admins use their devices to connect to the Application Server through HTTPS. Inside the application server, different modules handle booking, listing, user management, and payments. The Database Server stores all important data, such as user details, bookings, and payment records, while the Payment Gateway processes secure online transactions through API calls. This setup ensures safe access, smooth booking, and reliable management for all users.

10. Component Diagram

This diagram shows the main parts (**components**) of the Tourist Management System and how they work together:



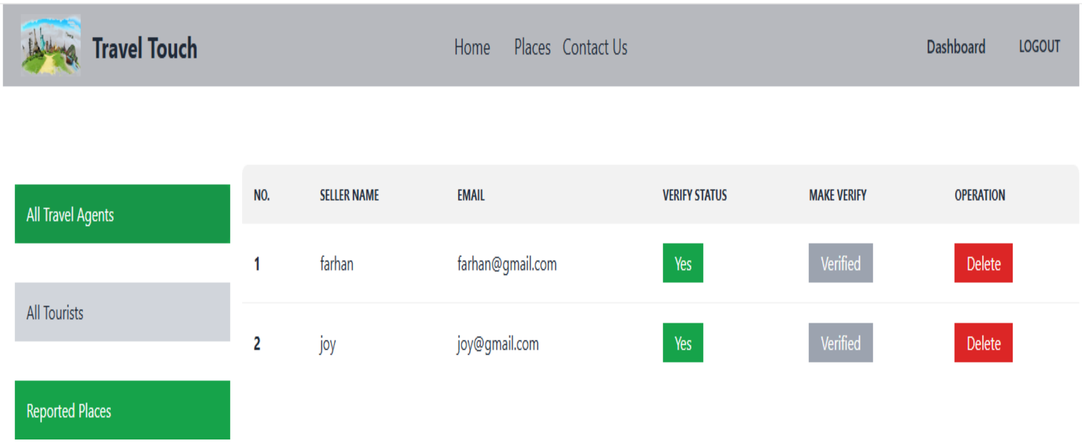
- **Components in the Diagram:**

1. **Tourist**
   - The main user who uses the system.
   - They can view packages, place orders, make payments, and manage their profile.
2. **Agent**
   - The service provider.
   - They can create, update, and manage tour packages and their own profile.
3. **View Packages**
   - A component that shows available **tour packages** to tourists.
4. **Order**
   - Handles bookings when a tourist decides to buy a package.
5. **Make Payment**
   - Manages online or offline payments for bookings.
6. **Tourist Package**
   - Stores all the details of different travel/tour packages.
   - Acts as the **main database** of the system.
7. **Manage Profile**
   - Lets both **Tourists** and **Agents** manage their personal details, preferences, and login information.

Project Snapshot

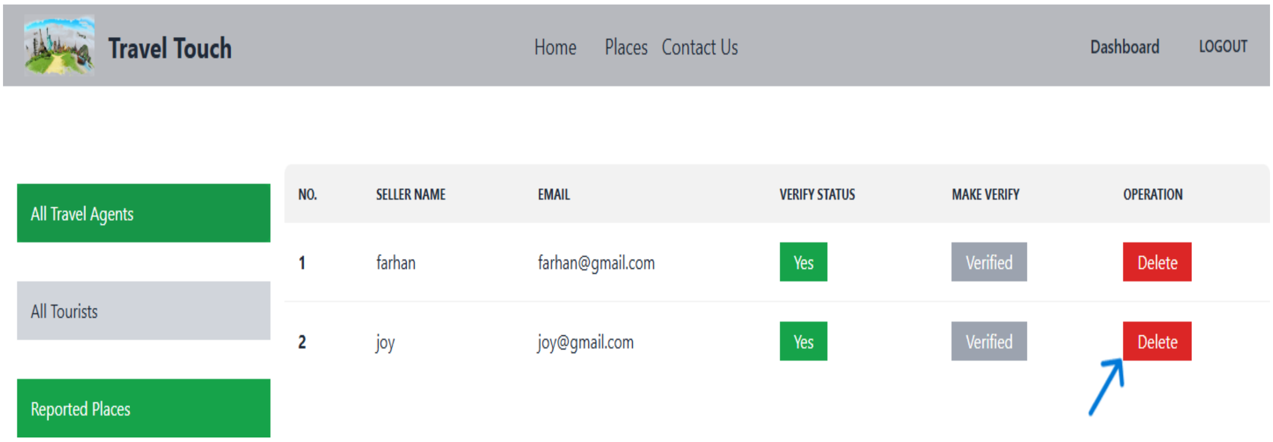
10.1 Admin Features:

- **View all Tourist and Agents**



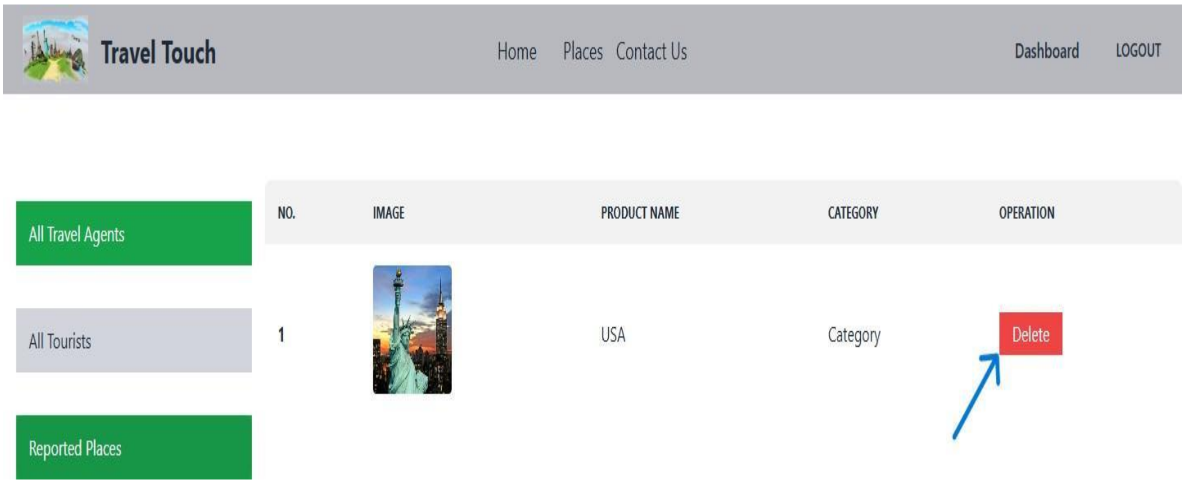
NO.	SELLER NAME	EMAIL	VERIFY STATUS	MAKE VERIFY	OPERATION
1	farhan	farhan@gmail.com	Yes	Verified	Delete
2	joy	joy@gmail.com	Yes	Verified	Delete

- **Delete faulty Agents and Tourists**



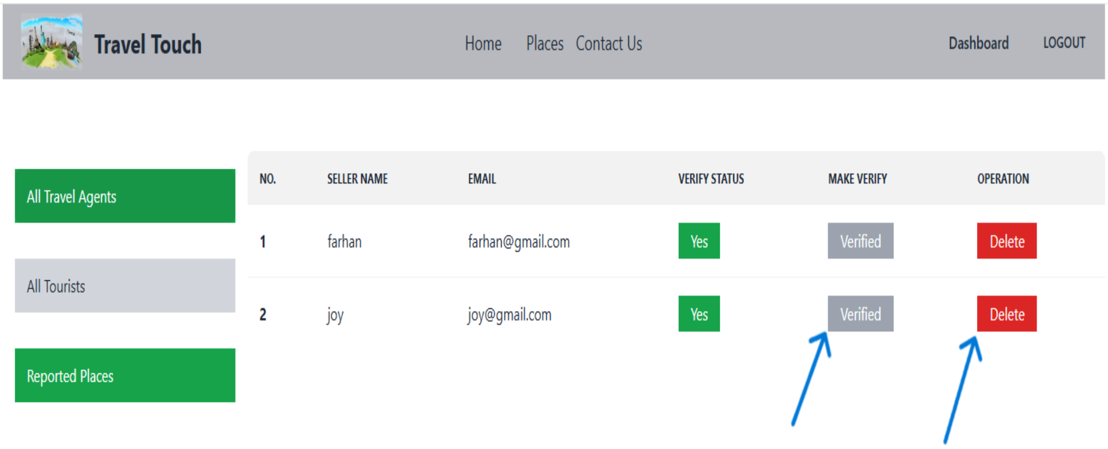
NO.	SELLER NAME	EMAIL	VERIFY STATUS	MAKE VERIFY	OPERATION
1	farhan	farhan@gmail.com	Yes	Verified	Delete
2	joy	joy@gmail.com	Yes	Verified	Delete

- **View all reported places.**



NO.	IMAGE	PRODUCT NAME	CATEGORY	OPERATION
1		USA	Category	Delete

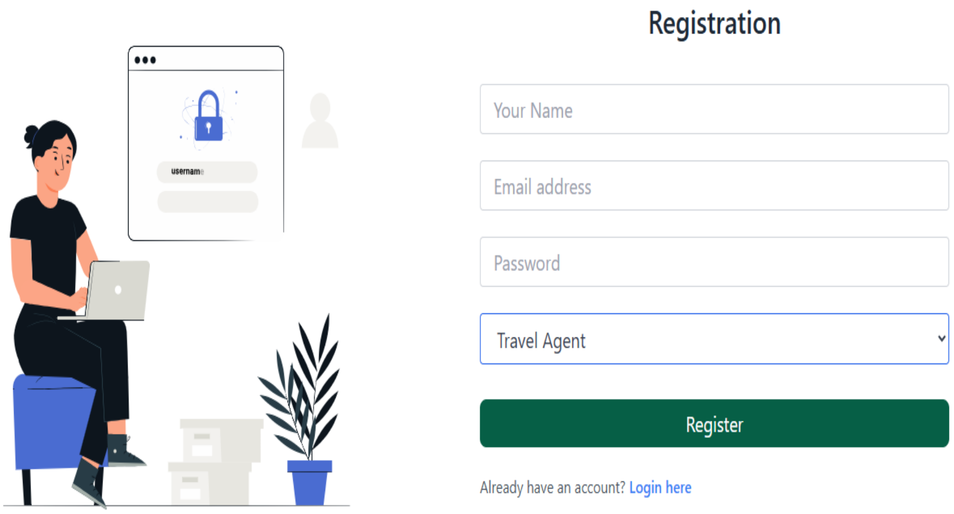
- **Verify or Delete places advertised by agents**



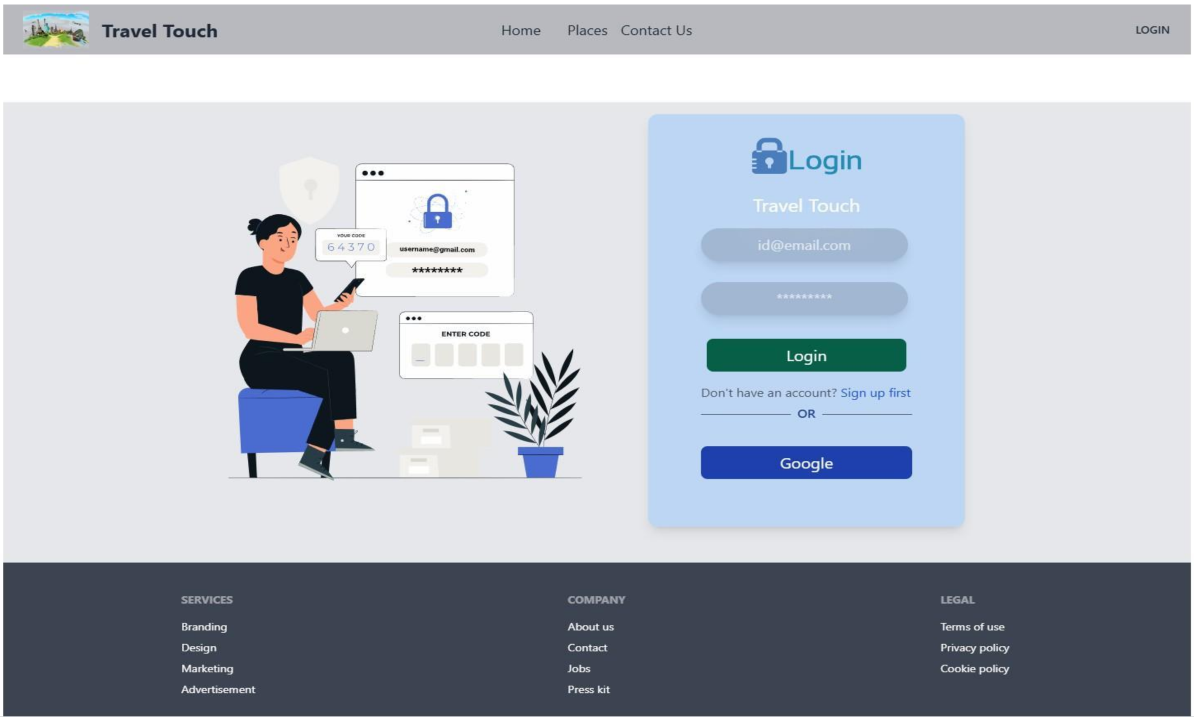
NO.	SELLER NAME	EMAIL	VERIFY STATUS	MAKE VERIFY	OPERATION
1	farhan	farhan@gmail.com	Yes	Verified	Delete
2	joy	joy@gmail.com	Yes	Verified	Delete

10.2 Agent Features

- **Sign up**

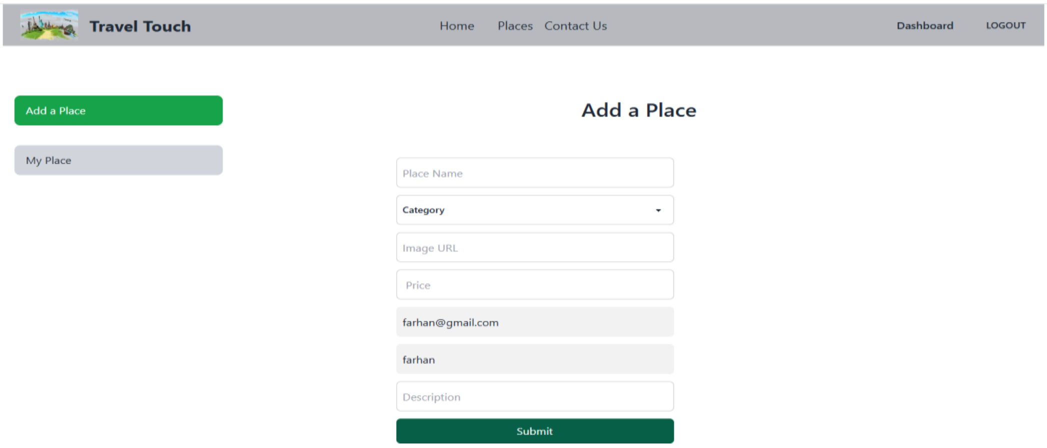


- **Login**



Don't have an account? Sign up first

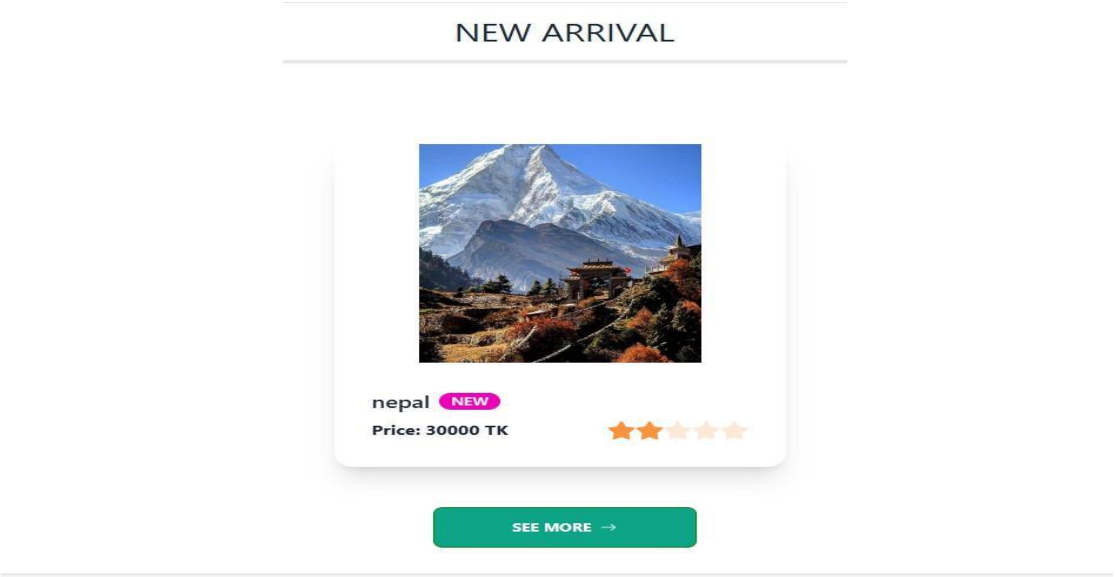
- **Add new places (Nepal, Indonesia etc.)**



farhan@gmail.com

farhan

- **Advertise places (shows up as new on the home page)**



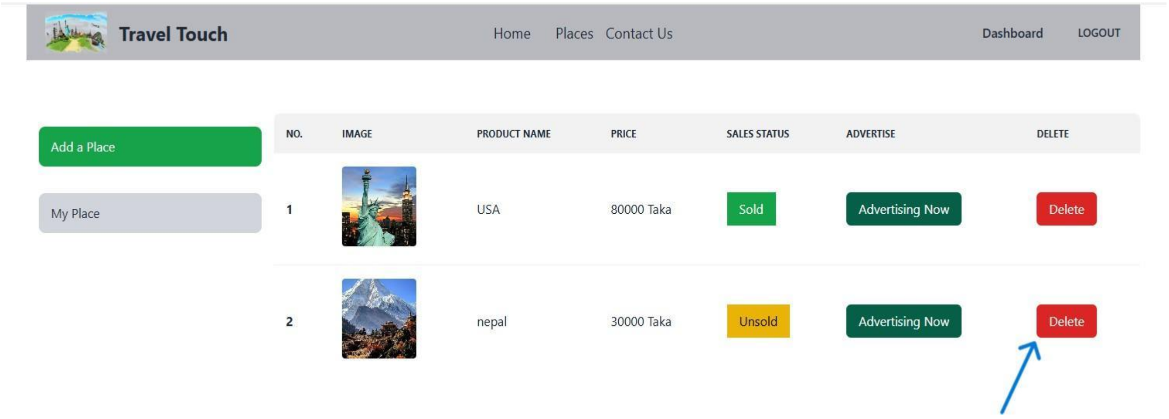
NEW ARRIVAL

nepal NEW

Price: 30000 TK ★★☆☆☆

SEE MORE →

- **Delete places from their dashboard**



Travel Touch Home Places Contact Us Dashboard LOGOUT

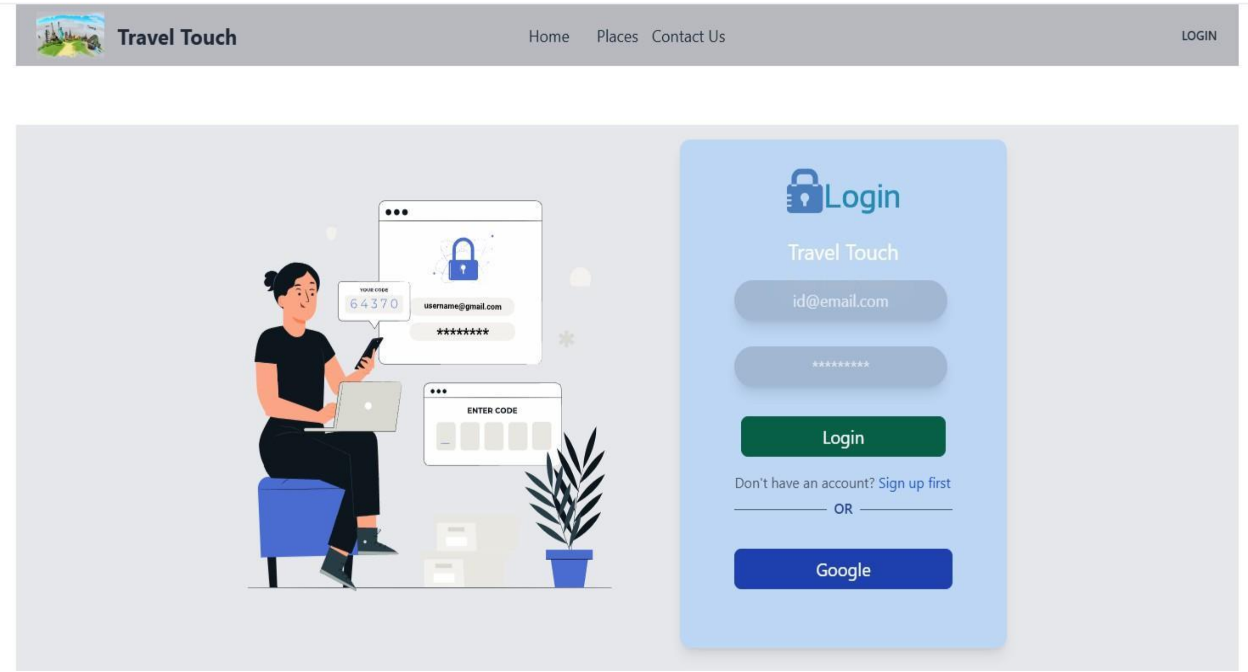
Add a Place

My Place

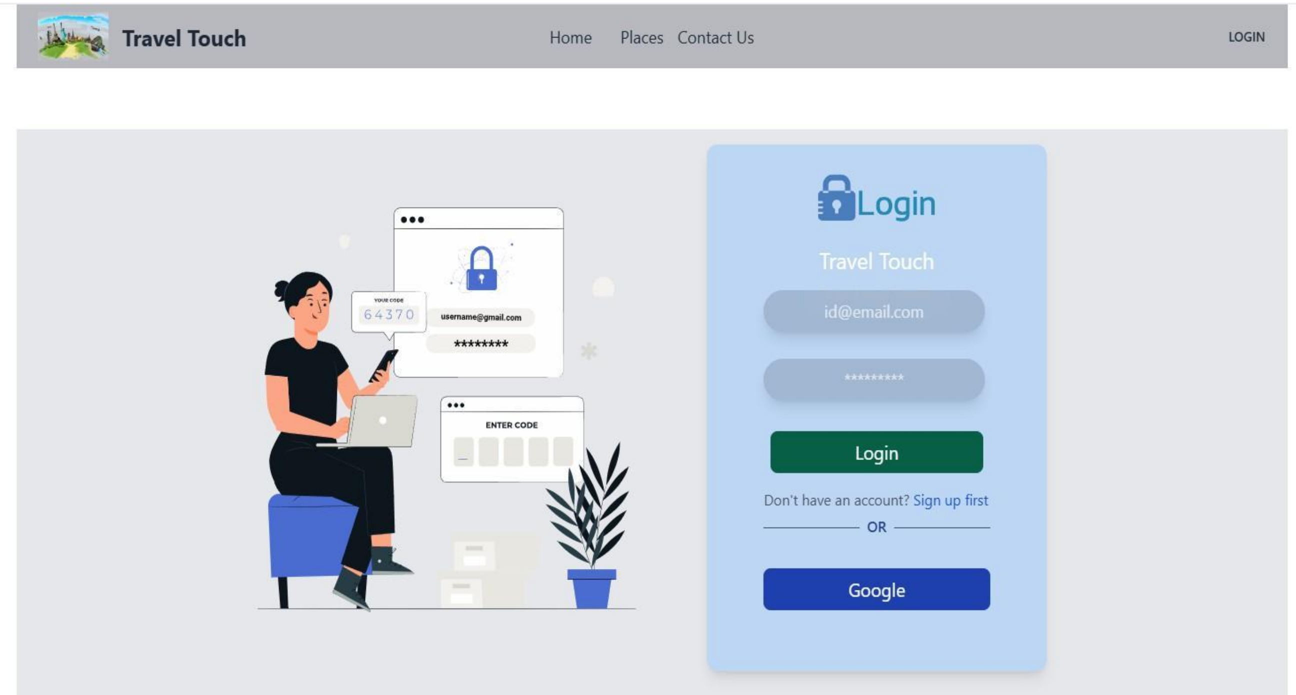
NO.	IMAGE	PRODUCT NAME	PRICE	SALES STATUS	ADVERTISE	DELETE
1		USA	80000 Taka	Sold	Advertising Now	Delete
2		nepal	30000 Taka	Unsold	Advertising Now	Delete

8.3 Tourist Features

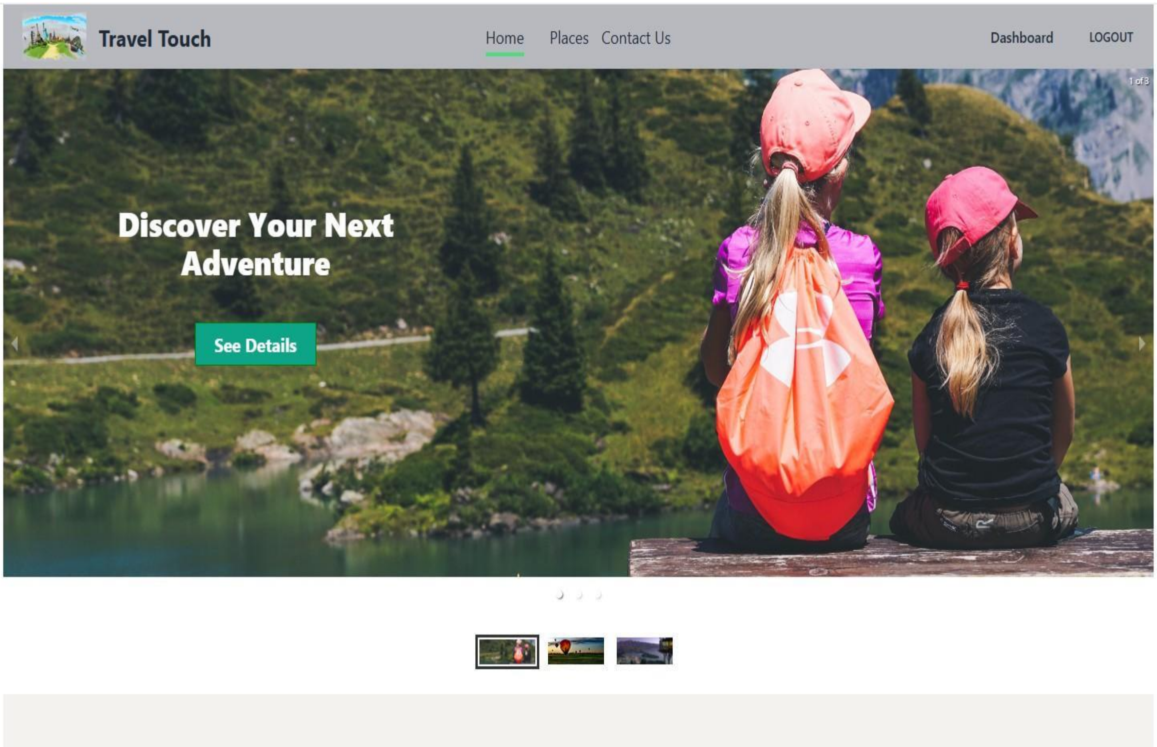
- **Sign up**



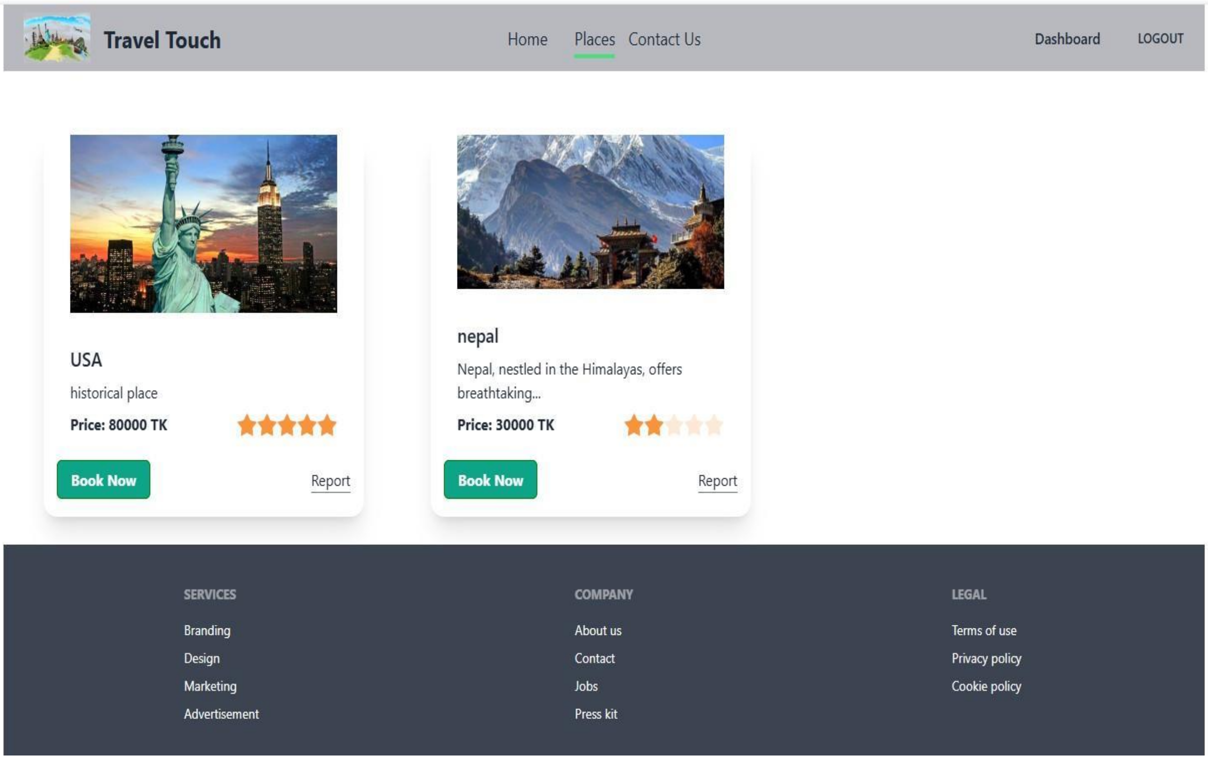
- **Log in**



- **View newly arrived places on the home page**



- **View all places (new and old) on the places page**



USA

historical place

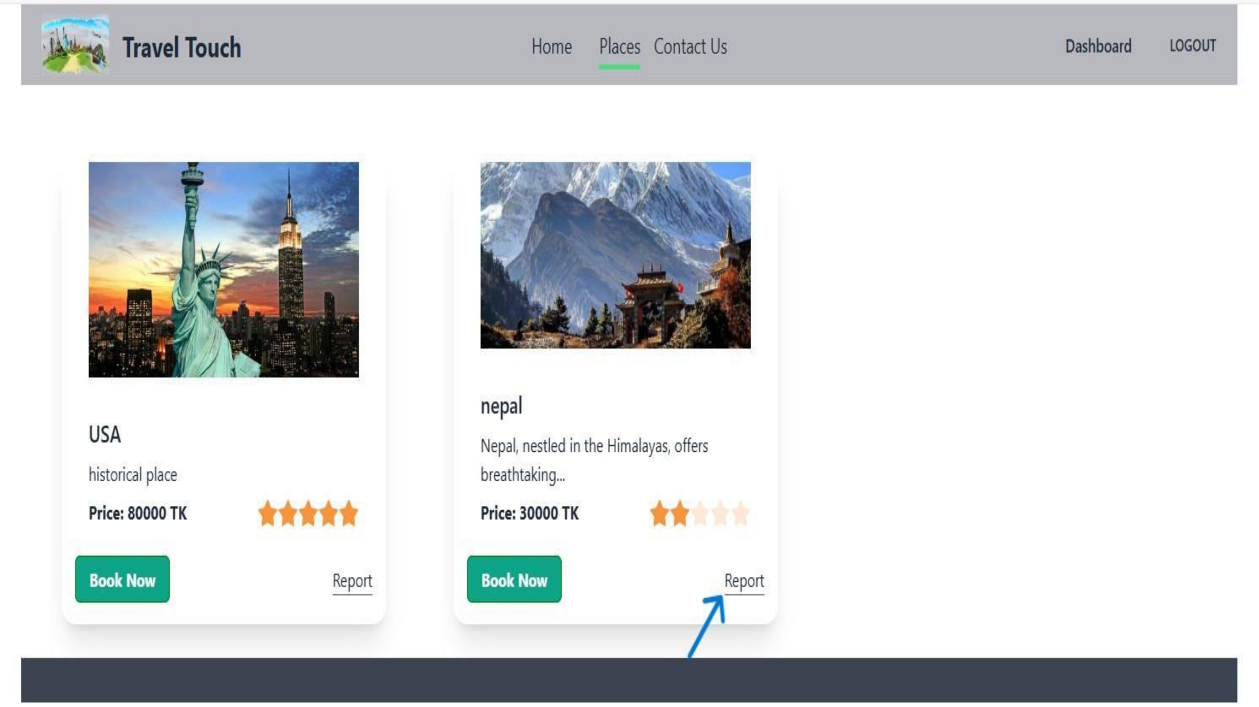
Price: 80000 TK

nepal

Nepal, nestled in the Himalayas, offers breathtaking...

Price: 30000 TK

- **Book a place from the Places Page**



USA

historical place

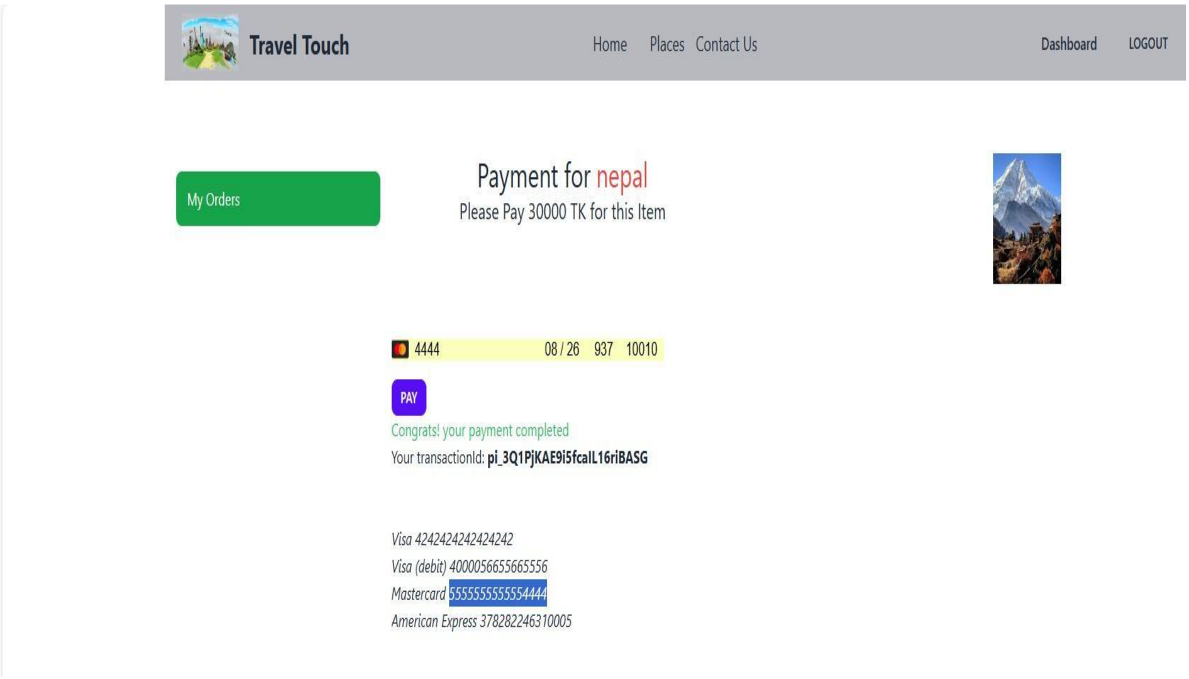
Price: 80000 TK

nepal

Nepal, nestled in the Himalayas, offers breathtaking...

Price: 30000 TK

- **Receive a transaction ID after successful payme**



Payment for nepal

Please Pay 30000 TK for this Item

4444 08 / 26 937 10010

Congrats! your payment completed

Your transactionId: **pi_3Q1PjKAE9i5fcalL16riBASG**

Visa 4242424242424242

Visa (debit) 4000056655665556

Mastercard 5555555555554444

American Express 378282246310005

11. Conclusion

The **Tourist Management System (Travel Touch)** is designed to create a single, easy-to-use platform that connects tourists, agents, and admins. Through this system, tourists can safely create accounts, explore verified destinations, book trips, make payments, and share their feedback. Agents are given the tools to add and promote travel packages, manage listings, and track their commissions. Admins play the key role of supervising the entire system by verifying accounts, approving listings, monitoring transactions, and ensuring quality and trust.

By combining secure login, verified listings, multiple payment options, and detailed booking records, the system provides a safe and reliable travel experience for everyone involved. It also ensures transparency through ratings, complaints, and admin monitoring. Overall, TravelTouch offers a modern and user-friendly solution for managing travel digitally, making tourism more organized, secure, and convenient for all stakeholders.