Tourism Management System

Project Overview:

The **Tourism Management System** is a comprehensive web-based application designed to streamline the management of tourism services, including destinations, tour packages, trip plans, bookings, payments, discounts, and additional personalized features. The goal of this project is to provide a seamless experience for both users and administrators, integrating destination and package management with booking, payment, and promotional systems to ensure convenience and efficiency.

This system allows users to browse destinations, select tour packages, plan trips, make bookings, apply discounts or vouchers, complete payments online, and provide feedback after the trip is completed. On the admin side, the system enables managing destinations, tour packages, trip plans, bookings, payment statuses, discounts or vouchers, and feedback.

Core Features and Functionalities:

1. **User Registration and Authentication (User & Admin):**

- Users can create accounts to browse destinations and book packages.
- Admins can log in to manage destinations, packages, and bookings.
- Secure JWT-based authentication and role-based access control.
- Validation: Ensure unique email, strong password, and proper input format during registration.

2. **Destination Management:**

- Add, update, and delete destinations with details like name, country, city, description, highlights, and images.
- Search and filter destinations.
- Validation: Check mandatory fields (name, country, city), validate image formats, and prevent duplicate destinations.

3. **Tour Package Management:**

- Create and manage tour packages linked to destinations.
- Define duration, price, capacity, inclusions/exclusions, and trip plans.
- Validation:
  - Ensure price is positive and numeric.
  - Validate duration (must be greater than zero).
  - Capacity must be a positive integer.
  - Check that linked destination exists.

4. **Trip Planning:**

- Add day-wise trip plans for each tour package.
- Update and manage trip plans easily.
- Validation: Ensure day numbers are sequential, and descriptions are not empty.

5. **Age-Based Package Selection:**

- Packages can be customized or filtered based on the age group of travellers (e.g., family-friendly, senior citizen packages, adventure packages for youth).
- Validation: Ensure age group matches package eligibility.

6. **Traveller Details (Co-Travellers):**

- Users can add details of accompanying travellers (relatives or friends) during booking.
- Information includes name, age, gender, relationship, and special requirements.
- Validation: Ensure all mandatory fields are filled and age-based package rules apply to all travellers.

7. **Guide Assignment:**

- Users can request a tour guide during booking.
- Admins can assign guides to specific packages or trips.

- Guide details include name, language proficiency, experience, and availability.

8. Booking System:

- Users can book packages, view booking status, and cancel if needed.
- Admins can manage bookings and update statuses (Pending, Confirmed, Cancelled).
- Validation: Check availability of package capacity before confirming booking.

9. Payment Integration:

- Secure payment processing with status tracking (Initiated, Success, Failed).
- Supports payment intent and webhook handling.
- Validation: Verify amount matches booking total and currency is supported.

10. Discounts and Vouchers:

- Users can apply discount codes or vouchers during booking.
- Admins can create and manage promotional codes with validity dates and discount percentages.
- Validation:
- Ensure voucher is active and not expired.
- Validate discount amount does not exceed package price.

11. Feedback and Reviews:

- Users can provide feedback after completing a trip.
- Ratings and comments can be submitted for destinations, packages, and overall experience.
- Admins can view and manage feedback to improve services.
- Feedback can be displayed on destination and package pages for transparency.

12. API Testing & Security:

- **Postman Integration for API Testing:**
  All endpoints can be tested using Postman collections for easy validation and debugging.
- **JWT-based Security for All Endpoints:**
  Secure authentication and authorization using JSON Web Tokens (JWT). Users must include the token in the Authorization header as:
  Authorization: Bearer <token>.

User Roles:

1. User

- **Browse Destinations and Packages:**
  View destination details, tour packages, and trip plans.
- **Booking and Payment:**
  Make bookings, apply discounts/vouchers, and complete payments securely.
- **Track Status:**
  Monitor booking and payment status.
- **Add Co-Travellers:**
  Provide details of accompanying travellers (name, age, gender, relationship).
- **Request Guide:**
  Option to request a tour guide during booking.
- **Provide Feedback:**
  Rate and review destinations, packages, and overall experience after trip completion.

---

2. Admin

- **Manage Destinations:**
  Add, update, and delete destinations.
- **Manage Packages and Trip Plans:**
  Create, update, and validate packages and itineraries.
- **Manage Bookings:**
  View all bookings and update statuses (Pending, Confirmed, Cancelled).
- **Monitor Payments:**
  Track payment statuses and handle issues.
- **Manage Discounts and Vouchers:**
  Create and manage promotional codes.
- **Assign Guides:**
  Allocate guides to packages or trips.
- **Manage Feedback:**
  View and moderate user feedback.
- **User Management:**
  View and manage user accounts and roles.
- **System Analytics:**
  Monitor overall system performance and data.

Technology Stack:

Backend (Spring Boot):

- **Spring Boot:** For creating the RESTful API services to handle business logic, order processing, and user management.
- **Spring Security:** For implementing user authentication and authorization.
- **Spring Data JPA:** For database interactions and ORM mapping.

- **SQL (MySQL):** For relational database management, schema design, and query optimization.
- **JUnit:** For unit testing and integration testing of services, controllers, and business logic to ensure reliability and maintainability.
- **GitHub:** For version control, collaborative development, and CI/CD integration.

Project Flow:

1. Authentication Phase

- **User/Admin Login or Registration**
  - Validate credentials.
  - Generate **JWT token** for secure access.
  - Token used in all subsequent API calls.

---

2. User Journey

Step 1: Browse & Select

- View **Destinations** (name, country, city, images).
- Explore **Tour Packages** linked to destinations.
- Filter packages by **price, duration, age group** (family, senior, youth).
- Check **Trip Plans** (day-wise itinerary).

Step 2: Booking

- Select package → Enter travel dates.
- Add **Co-Travelers** (name, age, gender, relationship, special needs).
- Validate **age-based eligibility** for all travellers.
- Request **Guide** (optional).
- Apply **Discount/Voucher** (optional).
- Confirm booking → System checks **capacity**.

Step 3: Payment

- Initiate payment → Validate amount & currency.
- Complete payment → Update status (**Initiated → Success/Failed**).
- Generate booking confirmation & itinerary.

Step 4: After Trip

- Submit **Feedback** (ratings & comments for destination, package, overall experience).

---

3. Admin Journey

- **Manage Destinations**: Add, update, delete.
- **Manage Packages & Trip Plans**: Validate price, duration, capacity.
- **Manage Bookings**: Update status (Pending, Confirmed, Cancelled).
- **Assign Guides**: Based on availability & language.
- **Manage Discounts/Vouchers**: Create, update, validate.
- **Monitor Payments**: Track statuses & resolve issues.
- **Review Feedback**: Moderate and publish.
- **User Management & Analytics**: Monitor system performance.

---

4. Validation Rules

- Unique email & strong password for registration.
- Mandatory fields for destinations & packages.
- Age-based eligibility for packages.
- Capacity check before confirming booking.
- Voucher validity & discount limits.
- Payment amount matches booking total.

Models:

1. Account

- **Primary Key: id** – Unique identifier for each account.
- **fullName:** Complete name of the account holder.
- **email:** Unique email address for login.
- **passwordHash:** Encrypted password for security.
- **phone:** Contact number for communication.
- **role:** Defines access level (USER or ADMIN).
- **isActive:** Indicates if the account is active.
- **createdAt / updatedAt:** Timestamps for record creation and updates.

2. Destination

- **Primary Key: id** – Unique identifier for destination.
- **name:** Name of the destination.
- **country / city:** Location details.
- **description:** Overview of the destination.
- **highlights:** Key attractions or features.
- **images:** URLs or paths to destination images.
- **isActive:** Visibility status in the system.
- **createdAt / updatedAt:** Audit timestamps.

3. TourPackage

- **Primary Key: id** – Unique identifier for package.

- **title:** Name of the tour package.
- **destinationId:** Links package to a destination.
- **description:** Details of the package.
- **durationDays:** Number of days for the trip.
- **basePrice:** Price per traveller or booking.
- **capacity:** Maximum allowed travellers.
- **includes / excludes:** Items included or excluded.
- **ageEligibility:** Allowed age group (Family, Senior, Youth).
- **isActive:** Availability status.
- **createdAt / updatedAt:** Audit timestamps.

4. **TripPlan**
   - **Primary Key: id** – Unique identifier for trip plan.
   - **tourPackageId:** Links to parent package.
   - **dayNumber:** Day sequence in itinerary.
   - **title:** Short title for the day's plan.
   - **description:** Activities planned for the day.
   - **createdAt / updatedAt:** Audit timestamps.
5. **Guide**
   - **Primary Key:** id – Unique identifier for guide.
   - **fullName:** Guide's name.
   - **languages:** Languages spoken by the guide.
   - **yearsOfExperience:** Experience in years.

- **availabilityStatus:** Current availability (Available/Busy).
- **contactPhone:** Contact number.
- **isActive:** Indicates if guide is active.
- **createdAt / updatedAt:** Audit timestamps.

6. **Booking**

- **Primary Key: id** – Unique identifier for booking.
- **accountId:** Links booking to an account.
- **tourPackageId:** Links booking to a package.
- **startDate / endDate:** Travel dates.
- **travellersCount:** Number of travellers.
- **status:** Booking status (Pending, Confirmed, Cancelled).
- **guideRequested:** Indicates if guide was requested.
- **guideId:** Assigned guide (optional).
- **subtotalAmount / discountAmount / totalAmount:** Pricing details.
- **createdAt / updatedAt:** Audit timestamps.

7. Traveller

- **Primary Key: id** – Unique identifier for traveller.
- **bookingId:** Links traveller to a booking.
- **fullName:** Traveler's name.
- **age:** Age for eligibility.
- **gender:** Gender of traveller.
- **relationship:** Relation to account holder.

- **specialRequirements:** Dietary or medical needs.
- **createdAt / updatedAt:** Audit timestamps.

8. Voucher

- **Primary Key: id** – Unique identifier for voucher.
- **code:** Unique voucher code.
- **description:** Details of the offer.
- **discountType:** Percentage or flat discount.
- **discountValue:** Discount amount.
- **validFrom / validUntil:** Validity period.
- **usageLimit / usageCount:** Usage tracking.
- **isActive:** Indicates if voucher is active.
- **createdAt / updatedAt:** Audit timestamps.

9. Payment

- **Primary Key:** id – Unique identifier for payment.
- **bookingId:** Links payment to a booking.
- **provider:** Payment gateway used.
- **transactionRef:** Transaction reference number.
- **amount:** Amount paid.
- **status:** Payment status (Initiated, Success, Failed).
- **paidAt:** Timestamp of successful payment.
- **createdAt / updatedAt:** Audit timestamps.

10. Feedback

- **Primary Key: id** – Unique identifier for feedback.
- **bookingId:** Links feedback to a booking.
- **accountId:** Who submitted the feedback.
- **destinationId / tourPackageId:** Optional references.
- **rating:** Rating (1–5).
- **comments:** User review text.
- **isPublished:** Indicates if feedback is visible.
- **createdAt / updatedAt:** Audit timestamps.

Relationships with Cardinality:

Account ↔ Booking

- **One-to-Many:** One Account can have many Bookings.
- **Many-to-One:** Each Booking belongs to one Account.
- **Foreign Key:** Booking.accountId → Account.id

Destination ↔ TourPackage

- **One-to-Many:** One Destination can have many TourPackages.
- **Many-to-One:** Each TourPackage belongs to one Destination.
- **Foreign Key:** TourPackage.destinationId → Destination.id

---

TourPackage ↔ TripPlan

- **One-to-Many:** One TourPackage can have many TripPlans.
- **Many-to-One:** Each TripPlan belongs to one TourPackage.
- **Foreign Key:** TripPlan.tourPackageId → TourPackage.id

TourPackage ↔ Booking

- **One-to-Many:** One TourPackage can have many Bookings.
- **Many-to-One:** Each Booking belongs to one TourPackage.
- **Foreign Key:** Booking.tourPackageId → TourPackage.id

Booking ↔ Traveller

- **One-to-Many:** One Booking can have many Travellers.
- **Many-to-One:** Each Traveller belongs to one Booking.
- **Foreign Key:** Traveller.bookingId → Booking.id

Guide ↔ Booking

- **One-to-Many (Optional):** One Guide can be assigned to many Bookings.
- **Many-to-One (Optional):** Each Booking may have one Guide.
- **Foreign Key:** Booking.guideId → Guide.id

Booking ↔ Payment

- **One-to-Many:** One Booking can have many Payments.
- **Many-to-One:** Each Payment belongs to one Booking.
- **Foreign Key:** Payment.bookingId → Booking.id

Account ↔ Feedback

- **One-to-Many:** One Account can submit many Feedback entries.
- **Many-to-One:** Each Feedback belongs to one Account.
- **Foreign Key:** Feedback.accountId → Account.id

Booking ↔ Feedback

- **One-to-Many:** One Booking can have many Feedback entries.
- **Many-to-One:** Each Feedback belongs to one Booking.
- **Foreign Key:** Feedback.bookingId → Booking.id

Voucher ↔ Booking

- **Many-to-Many:** A Voucher can apply to many Bookings, and a Booking can have many Vouchers.
- **Join Table:** BookingVoucher with:
  - BookingVoucher.bookingId → Booking.id
  - BookingVoucher.voucherId → Voucher.id