Swami Keshvanand Institute of Technology, Management & Gramothan, Jaipur

FACULTY KIT

Objective –

The objective of this faculty kit is to outline the roles, responsibilities, and tools necessary for the successful implementation of the project **Nomads: AI Trip Planner and Destination Explorer** by students. This system enables users—primarily tourists—to explore destinations, generate personalized itineraries using AI, view city insights, and manage their travel data. The kit provides faculty with guidelines for overseeing progress, ensuring timely evaluation, and maintaining project quality and adherence to academic standards.

Requirements Specification

The Nomads platform will include:

- **User Authentication**: Secure login and user management using Google Authentication.
- **Destination Information**: Facilitate access to detailed information about destinations, including attractions, hotels, and local services.
- **Itinerary Planning**: Gemini AI-powered itinerary generation based on user preferences.
- **Booking Services**: Integration of booking functionalities for hotels, flights, and local activities.
- **Payment Integration**: Secure payment gateway for transactions related to bookings and services.
- **Search Functionality**: Advanced search options for destinations, services, and availability.
- **Admin Control**: Admin will manage destination data, user roles, and bookings.

Technology Familiarization

The project will use **Spring Boot** for backend development, enabling the implementation of REST APIs and database interactions. The frontend will be developed using **React.js**, providing a dynamic, responsive user interface. For the database, **SQL** and **NoSQL** databases will be utilized to store user information, destination details, and booking data. The integration of payment functionality will use **Stripe**, **Razorpay**, or similar gateways.

Database Creation

The Travel Guide System will use relational and non-relational databases to handle different data types:

- **SQL Database** for structured data (user profiles, destinations, bookings).
- **NoSQL Database** for unstructured data such as user logs or multimedia related to destinations.

Database Entities:

1. **User**: User details, role (Admin, Travel Agent, Tourist).
2. **Destination:** Information about various destinations, including attractions, hotels, and local services.
3. **Booking:** Data regarding user bookings for services like flights, hotels, and activities.

High-Level and Detailed Design

System Overview:

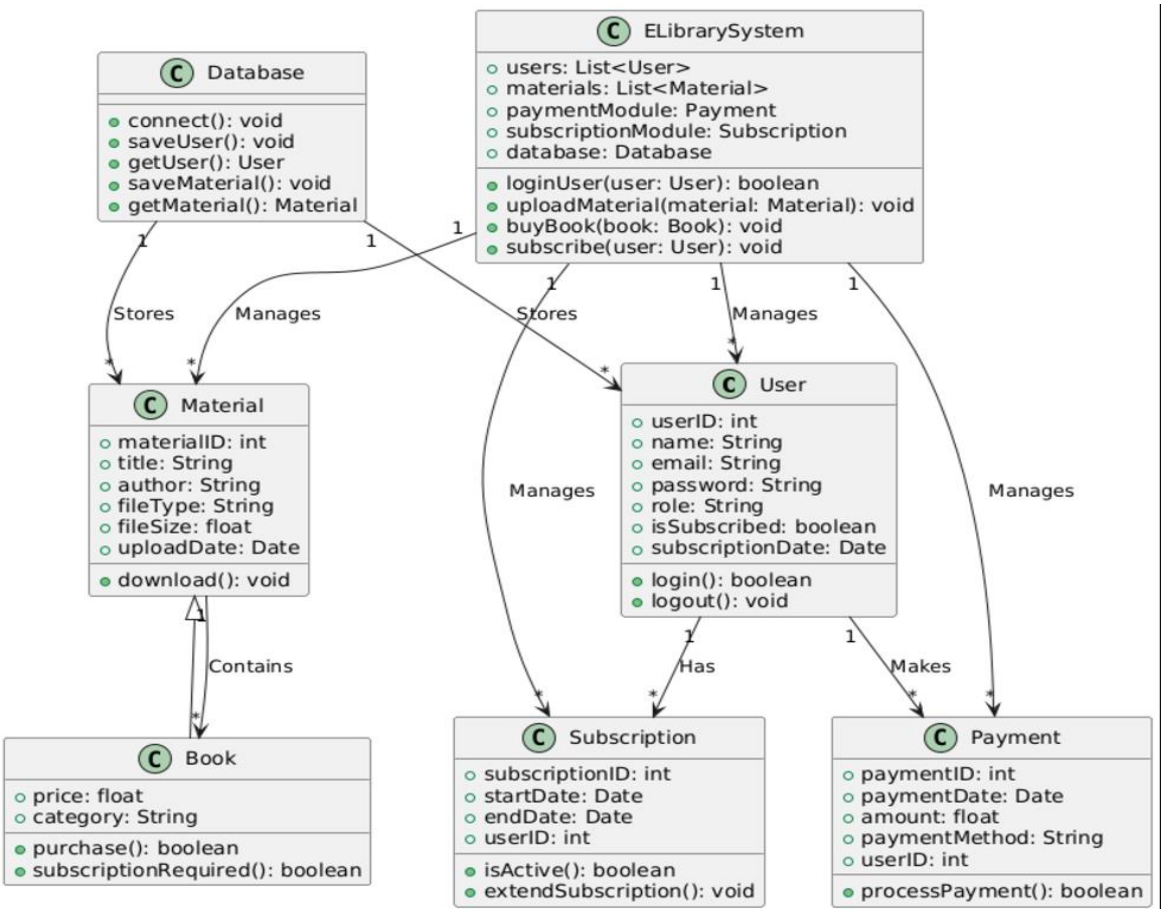
The system consists of three layers:

- **Frontend Layer**:
    - A React-based user interface allowing seamless interaction for exploring destinations, managing itineraries, making bookings, and processing payments.
- **Backend Layer**:
    - Spring Boot handles the server-side logic, database interactions, and communication with third-party services like payment gateways and booking APIs.
- **Database Layer**:
    - **SQL** (for structured data like user details, destinations, and bookings).

- **NoSQL** (for unstructured data like multimedia and logs) ensures scalability, fast processing, and reliable storage.

Detailed Design:

- **Frontend**:
  - The user interface will include interactive features for searching destinations, booking services, and creating itineraries.
  - User roles (Tourist, Travel Agent, Admin) will dictate the features accessible on the frontend.
- **Backend**:
  - RESTful APIs will handle user authentication, destination data management, itinerary planning, and booking integration.
  - Secure payment APIs will ensure reliable financial transactions.



Frontend Implementation –

React.js will be used to build the user interface for the Travel Guide System. The UI will have sections for login, registration, destination browsing, itinerary creation, and booking processing. Each user type (Tourist, Travel Agent, Admin) will have specific permissions and views:

- **Tourists**: Explore destinations, plan itineraries, and make bookings.
- **Travel Agents**: Manage travel services, provide recommendations, and oversee customer bookings.
- **Admins**: Monitor the system, manage user roles, and update destination details.

Integrating the Frontend with the Database

To integrate the frontend with the database, the backend will expose **REST APIs** using **Spring Boot**, handling HTTP requests for CRUD operations. These operations will include fetching destination details, updating booking statuses, and processing payments. **React.js** will consume these APIs using **Axios**, providing dynamic content to the user interface based on data stored in the databases.

Example functionalities:

1. **Destination Search**: The frontend fetches destination details via an API and displays results based on user queries.
2. **Itinerary Management**: Users can create, update, and view their itineraries, with data synced to the database.
3. **Booking Integration**: Payment and booking confirmations are dynamically updated in the UI after successful transactions.

Test Plan Review

Testing will be conducted at various stages to ensure all functionalities of the Travel Guide System work seamlessly:

- **Unit Testing**: Validate individual backend API logic for correctness (e.g., fetching destination data, booking services).
- **Integration Testing**: Ensure smooth interaction between the frontend and backend.
- **UI/UX Testing**: Test the user interface for responsiveness, usability, and clarity.
- **Performance Testing**: Verify the system's ability to handle high user traffic during peak seasons.

Final Review

At the project's conclusion, a final review will validate that the Travel Guide System meets all specified requirements. This includes:

1. **User Authentication**: Ensuring secure login for all roles (Tourist, Travel Agent, Admin).
2. **Destination Data**: Verifying that destination information is accurate and accessible.
3. **Itinerary and Booking**: Confirming that users can create and manage itineraries and complete bookings.
4. **Payment Module:** Testing secure payment gateway integration and successful transaction processing.
5. **Feedback Collection**: Gathering user feedback to identify areas for improvement.

Documents/References to Aid Evaluation

- **Spring Boot Documentation**: For backend implementation and API development.
- **React.js Documentation**: For frontend development and UI/UX handling.
- **SQL and NoSQL Database Tutorials**: For designing and integrating databases.
- **Stripe/Razorpay API Documentation**: For integrating the payment gateway.
- **Agile Methodology Resources**: For project management and sprint planning.

Conclusion

The Faculty Kit for the Nomads: AI Trip Planner and Destination Explorer serves as a comprehensive guide for faculty members overseeing the project's development and deployment. Through its structured approach, it ensures all aspects of the project, from initial planning to final review, are thoroughly addressed.

This kit highlights:

- **Technology Familiarization**: Preparing faculty to guide the adoption of Spring Boot, React.js, and SQL/NoSQL databases.
- **High-Level and Detailed Design**: Providing a clear roadmap for the system's technical architecture.
- **Frontend Implementation**: Outlining the creation of an intuitive and responsive user interface.
- **Backend Integration**: Ensuring seamless data flow between the frontend and backend.

With defined testing protocols and a thorough review process, faculty can evaluate the system's functionality, scalability, and security. The kit supports both the development and quality assurance phases, ensuring that the **Travel Guide System** provides an engaging, user-friendly, and efficient platform for travellers and administrators.

This kit is essential for achieving the project's objectives and offering students a valuable learning experience in system design and implementation.