Project Proposal: Hospitality and Accommodation Booking Platform API Service

By Reem Ibrahim

Contents

Project Overview:

The aim of this project is to build the backend API service for a hospitality and accommodation booking platform. The API will provide endpoints for user management, property listings, booking operations, payments, content management, reviews, favorites, and notifications. It will be developed using modern backend frameworks (such as Node.js or Laravel), utilize relational databases (SQLite), and follow RESTful best practices.

Main Functionalities to Implement:

1. User Management System:

- **Registration & Authentication:** Secure user registration, login, logout, and password reset endpoints using OAuth2.
- **Role-Based Access Control:** Assign roles (admin, host, guest) and restrict access to endpoints based on user roles.
- **Profile Management:** Endpoints for viewing and updating user profiles.

2. Property & Booking Management:

- **CRUD Operations:** Endpoints for creating, reading, updating, and deleting property listings.
- **Booking API:** Endpoints for searching availability, creating bookings, modifying or cancelling reservations, and viewing booking history.
- **Availability & Pricing:** Logic for managing property availability and dynamic pricing.

3. Payment Integration:

- **Payment Gateway Integration:** Endpoints for processing payments, refunds, and viewing transaction history using third-party services.
- **Payment Validation:** Secure handling of sensitive payment data and confirmation flows.

4. Content Management API:

- **File Uploads:** Endpoints for uploading images and documents with proper validation.
- **Content CRUD:** Endpoints for hosts and admins to manage descriptions, amenities, policies, etc.

5. Real-Time Notification System:

- **Notifications API:** Endpoints for sending and retrieving notifications (booking confirmations, cancellations, updates).
- **Integration with Push Services:** Support for email and push notifications via third-party APIs.

6. API Documentation & Testing Suite:

- **Comprehensive Documentation:** Use Postman to document all endpoints, request/response schemas, authentication flows, and error codes.
- **Automated Testing:** Postman tests for all endpoints to ensure reliability and correctness.

7. Request Validation & Error Handling:

- **Validation Middleware:** Consistent validation of incoming requests for required parameters, types, and authorization.
- **Error Handling:** Structured error responses and logging for debugging and client feedback.

8. Third-Party API Integrations:

Integrate with external services for payments, notifications, maps/location, and possibly reviews.

9. Review System:

- **Add Reviews:** Endpoints for users to submit reviews and ratings for properties they have booked.
- **View Reviews:** Endpoints for retrieving reviews for properties.
- **Moderation:** Admin tools for moderating inappropriate reviews.

10. Favorites Functionality:

- **Add to Favorites:** Endpoints for users to save properties to their favorites list.
- **Remove from Favorites:** Endpoints to remove properties from favorites.
- **View Favorites:** Endpoint to retrieve the user's list of favorite properties.

Technology Used:

- **Backend Framework:** Laravel
- **Database:** SQLite
- **Authentication:** OAuth2
- **API Testing/Documentation:** Postman
- **Version Control:** Git (GitHub)

Development Approach:

1. **Requirements Gathering:** Collaborate with senior developers and team members to finalize API requirements and data models.
2. **Design:** Draft API specifications, entity-relationship diagrams, and authentication flows.
3. **Implementation:** Develop modules iteratively, starting with user management and authentication, followed by property, booking, payments, reviews, favorites, and notifications.
4. **Testing & Documentation:** Document endpoints with Postman; create automated tests and validation scripts.
5. **Code Review:** Participate in code reviews, refactor as needed, and follow team conventions for maintainable code.

Postman Collection:

1. **Authentication:**
    - POST: /api/test
    - POST: /api/register
    - POST: /api/login
    - POST: /api/logout
    - POST: /api/password-reset

2. **User Management:**
    - GET: /api/users/:id
    - PUT: /api/users/:id
    - DELETE: /api/users/:id

3. **Properties:**
    - POST: /api/properties
    - GET: /api/properties
    - GET: /api/properties/:id
    - PUT: /api/properties/:id
    - DELETE: /api/properties/:id

4. **Rooms:**
    - POST: /api/properties/:propertyId/rooms
    - GET: /api/properties/:propertyId/rooms
    - GET: /api/rooms/:id
    - PUT: /api/rooms/:id
    - DELETE: /api/rooms/:id

5. **Bookings:**
    - POST: /api/bookings
    - GET: /api/bookings
    - GET: /api/bookings/:id

- PUT: /api/bookings/:id
- DELETE: /api/bookings/:id

6. **Payments:**
    - POST: /api/payments
    - GET: /api/payments/:id
    - POST: /api/refunds

7. **Content Management:**
    - POST: /api/files/upload
    - GET: /api/content
    - PUT: /api/content/:id
    - DELETE: /api/content/:id

8. **Reviews:**
    - POST: /api/properties/:id/reviews
    - GET: /api/properties/:id/reviews
    - DELETE: /api/reviews/:id

9. **Favorites:**
    - POST: /api/properties/:id/favorite
    - DELETE: /api/properties/:id/favorite
    - GET: /api/users/:id/favorites

10. **Notifications:**
    - GET: /api/notifications
    - POST: /api/notifications/send

Planned Additional CRUD APIs

1. **Admin Management:**
    - GET: /api/admins
    - POST: /api/admins
    - PUT: /api/admins/:id
    - DELETE: /api/admins/:id

2. **Amenities:**
    - GET: /api/amenities
    - POST: /api/amenities
    - PUT: /api/amenities/:id
    - DELETE: /api/amenities/:id

3. **Roles:**
    - GET: /api/roles
    - POST: /api/roles
    - PUT: /api/roles/:id
    - DELETE: /api/roles/:id

Expected Outcomes:

- A secure, scalable backend API for accommodation booking.
- Comprehensive documentation and testing for seamless frontend integration.
- A functioning backend API for a multi-module web application.
- Full API documentation in Postman.
- Exposure to backend design, validation, testing, and integration.