🏠 Airbnb Clone Backend Project

🚀 Objective

The backend for the Airbnb Clone project is designed to provide a robust and scalable foundation for managing user interactions, property listings, bookings, and payments. This backend supports all core features of Airbnb, ensuring a smooth and secure experience for users and hosts.

🏆 Project Goals

- **User Management**: Secure registration, login, and profile management.
- **Property Management**: Host property listings with CRUD capabilities.
- **Booking System**: Reserve properties, manage booking details.
- **Payment Processing**: Handle and record payment transactions.
- **Review System**: Allow users to leave reviews and ratings.
- **Data Optimization**: Ensure fast, efficient data handling.

👥 Team Roles

🔧 Backend Developer

Implements API endpoints, business logic, and maintains REST/GraphQL interfaces.

🗃️ Database Administrator

Designs schemas, manages migrations, handles indexing and optimization.

🚀 DevOps Engineer

Sets up Docker, manages CI/CD pipelines, monitors deployments and scaling.

🧪 QA Engineer

Writes and runs automated and manual tests to ensure backend stability.

⚙️ Technology Stack

Technology	Purpose
Django	High-level Python framework for building robust web backends.
Django REST Framework	For building RESTful APIs with powerful features.
PostgreSQL	Relational database for storing structured project data.
GraphQL	Flexible query language for advanced frontend interaction.

Technology	Purpose
Celery	Handles asynchronous background tasks like emails or payments.
Redis	In-memory data store for caching and session management.
Docker	Containerizes the project for consistent development and deployment.
CI/CD (GitHub Actions)	Automates testing and deployment processes.

Database Design

Users

- `id`
- `username`
- `email`
- `password`
- `is_host`

Properties

- `id`
- `title`
- `description`
- `location`
- `price_per_night`
- `owner_id` (FK to User)

Bookings

- `id`
- `user_id` (FK to User)
- `property_id` (FK to Property)
- `start_date`
- `end_date`

Payments

- `id`
- `booking_id` (FK to Booking)
- `amount`
- `payment_status`

Reviews

- `id`
- `user_id` (FK to User)
- `property_id` (FK to Property)

- `rating`
- `comment`

🔁 Relationships

- A **User** can own multiple **Properties**.
- A **User** can make multiple **Bookings**.
- A **Booking** belongs to one **Property**.
- A **Review** is linked to one **User** and one **Property**.
- A **Payment** is tied to a **Booking**.

✨ Feature Breakdown

☑️ User Management

Handles registration, login, logout, and profile editing using secure authentication.

🏘️ Property Management

Hosts can create, update, retrieve, and delete property listings.

📅 Booking System

Users can view availability, make bookings, and manage their reservations.

💰 Payment Processing

Secure handling of payments, transactions logged per booking.

🌟 Review System

Guests can rate and review properties post-booking to help others decide.

⚡ Data Optimization

Indexes and caching mechanisms are used to ensure performance and scalability.

🔐 API Security

🔑 Authentication

JWT-based or session-based auth for verifying users.

🔐 Authorization

Role-based access control (Host vs. User) to restrict certain actions.

🧱 Rate Limiting

Prevents abuse by limiting the number of requests per IP/user.

HTTPS

All traffic is encrypted using HTTPS to protect data in transit.

> **Why it matters:**

- **User data** must be protected from unauthorized access.
- **Payments** must be secure and verifiable.
- **APIs** must be resilient to brute-force and DDoS attacks.

CI/CD Pipeline

What is CI/CD?

Continuous Integration and Continuous Deployment (CI/CD) automate the testing and deployment of new code changes.

Tools Used

- **GitHub Actions**: Automates tests and linting on every push.
- **Docker**: Ensures consistent environments across development and production.
- **PostgreSQL Service**: Runs in Docker during test phase.
- **Heroku/Vercel/Render**: Optional deployment platforms.

> Benefits:

- Faster development cycles
- Fewer bugs in production
- Reliable deployments

Repository

GitHub Repository: `airbnb-clone-project`
This repository contains the backend implementation and documentation for the Airbnb Clone project.
