High Level Design Document

Introduction

This High Level Design (HLD) document outlines the architecture and core components for **PropEase - Real Estate Listings & Booking API**. The project aims to deliver a Spring Boot backend for managing property listings, bookings, and user profiles, with secure RESTful APIs, robust validation, and JWT-based authentication. The backend will use MySQL for persistent storage and Docker for deployment.

---

1. System Architecture Overview

Architecture Description:
PropEase follows a layered architecture with clear separation of concerns. The system exposes RESTful APIs, processes business logic, manages data persistence, and ensures secure access.

Layer/Component	Role/Responsibility
API Layer	Exposes REST endpoints for clients
Service Layer	Implements business logic and validation
Data Access Layer	Handles CRUD operations with MySQL
Security Layer	Manages JWT authentication and authorization
Database	Stores users, properties, bookings
Docker Container	Encapsulates the application for deployment

---

2. Component Interactions

Sequence Step	Description
1. Client Request	Client sends HTTP request to API endpoint
2. Authentication	Security layer validates JWT token
3. Request Processing	API layer forwards request to Service layer
4. Business Logic	Service layer processes request, applies validation
5. Data Access	Data Access layer interacts with MySQL as needed
6. Response	Result is returned to client via API layer

---

3. Data Flow Overview

Data Flow	Source	Destination	Purpose
User Registration/Login	Client	API/Security	Create user, issue JWT

Property Listing Management	Client	API/Service	CRUD operations on property listings
Booking Creation/Management	Client	API/Service	Bookings linked to users and properties
Data Persistence	Service Layer	MySQL	Store/retrieve entities
Authentication/Authorization	Client	Security Layer	Secure API access with JWT

---

4. Technology Stack

Technology	Purpose
Java 17+	Core programming language
Spring Boot	Application framework
Spring Security	JWT authentication/authorization
MySQL	Relational database
Maven	Build and dependency management
Docker	Containerization and deployment

---

5. Scalability, Reliability & Security

- **Scalability:**
  - Stateless REST APIs enable horizontal scaling via container orchestration.
  - Database can be scaled with replication or managed services.
- **Reliability:**
  - Input validation and error handling at API and service layers.
  - Consistent exception management and logging.
- **Security:**
  - JWT-based authentication for all endpoints.
  - Role-based access control for sensitive operations.
  - Secure password storage and input sanitization.

End of Document