## **Github** - <https://github.com/AwwwRyan/Real-Estate-Platform>

### Team: Aryan Srivastava A073 Aryan Walia A078 Aayush Chawla A008 Priya Pandey A048

# **Real Estate Platform Documentation**

## **Introduction to the Project**

This project aims to develop a comprehensive real estate platform that connects property buyers and sellers in a seamless digital environment. The platform will serve as a marketplace where users can list properties, search for suitable properties, communicate with potential buyers/sellers, schedule property viewings, and share experiences through reviews and ratings.

The system is designed as a modern web application utilizing microservices architecture to ensure scalability, maintainability, and robust performance. The platform caters to three primary user types: property sellers who want to list their properties, potential buyers searching for properties, and general users who can browse and interact with the platform's features.

Key features include user authentication and verification, property listing management, advanced search and filtering capabilities, in-app messaging system, appointment scheduling for property viewings, and a comprehensive review and rating system for both properties and sellers.

## **Problem Statement**

The traditional real estate market faces several challenges that hinder efficient property transactions:

**For Property Sellers:**

* Limited reach and visibility for property listings
* Difficulty in managing inquiries and scheduling viewings
* Lack of centralized platform to showcase property details and images
* Time-consuming process of communicating with multiple potential buyers
* No standardized way to build credibility and trust with buyers

**For Property Buyers:**

* Scattered information across multiple platforms and sources
* Difficulty in finding properties that match specific criteria
* Lack of verified seller information and property authenticity
* Complex process of scheduling property viewings
* Limited access to genuine reviews and ratings from previous buyers

**General Market Issues:**

* Information asymmetry between buyers and sellers
* Lack of transparency in property transactions
* Time-consuming manual processes for property searches and communications
* Trust and security concerns in online property transactions
* Fragmented user experience across different platforms and services

## **Objectives**

### **Primary Objectives:**

1. **Centralized Property Marketplace:** Create a unified platform where users can list, search, and discover properties efficiently
2. **Enhanced User Experience:** Provide an intuitive and responsive interface for seamless property browsing and management
3. **Secure Communication:** Implement a secure in-app messaging system for buyer-seller interactions
4. **Trust and Transparency:** Establish a review and rating system to build trust and provide transparency in transactions
5. **Efficient Scheduling:** Enable easy appointment booking for property viewings and inspections

### **Secondary Objectives:**

1. **User Verification:** Implement document-based user verification to ensure authenticity and build trust
2. **Advanced Search:** Provide comprehensive search and filtering options based on location, price, amenities, and other criteria
3. **Scalable Architecture:** Design a microservices-based system that can handle growing user base and data volume
4. **Mobile Responsiveness:** Ensure the platform works seamlessly across desktop and mobile devices
5. **Data Security:** Implement robust authentication and authorization mechanisms to protect user data

### **Technical Objectives:**

1. **Microservices Implementation:** Develop independent, scalable services for different functionalities
2. **API-First Design:** Create well-documented RESTful APIs for service communication
3. **Database Optimization:** Design efficient database schemas for optimal performance
4. **Performance Optimization:** Ensure fast loading times and responsive user interactions
5. **Code Maintainability:** Follow best practices and design patterns for clean, maintainable code

## **Scope of the Project**

### **1. Authentication (Login/Signup/Logout) Screen**

**Modules:**

* Authentication Service
* Profile Service (after authentication)

**Flow:**

* **Login/Signup:** Client sends credentials to Authentication Service:
  + Issues JWT/session token on success.
  + All further client requests send this token (HTTP header).
* **Profile Retrieval:** After login, frontend calls Profile Service with user\_id to fetch user profile and verification status.

**Service Communication:**

* Profile Service validates authentication tokens with Authentication Service or uses a shared secret/public key for JWT validation.

### **2. Property Listing (Home/Search) Screen**

**Modules:**

* Property Service
* Messaging Service
* Review Service

**Flow:**

* **Property Grid/List:**
  + Frontend queries Property Service for available properties, with filters (location, price, amenities, etc.).
  + For each property, also fetches aggregate ratings from Review Service if summary/rating scores are displayed.
* **Property Images:** Property Service exposes images per property (joined/queried separately).

**Service Communication:**

* Review Service provides property average ratings to the Property Service via API or client-side joins.
* Messaging Service is called when a user wants to message a seller: Property Service provides seller\_id, which is fed into Messaging Service.

### **3. Single Property Detail Screen**

**Modules:**

* Property Service
* Profile Service
* Review Service
* Messaging Service

**Flow:**

* **Property Data:** Fetch from Property Service by property\_id.
* **Seller Info:** Profile Service fetches seller's details and verification status by seller\_id.
* **Images/Amenities:** Property Service provides all linked images and amenities for property\_id.
* **Reviews:** Pulls property and seller reviews using Review Service.
* **Contact/Message Seller:** Initiates a conversation via Messaging Service (creates/gets conversation between buyer\_id & seller\_id).
* **Book Appointment:** Initiates via Appointment Service with relevant user/property IDs.

**Service Communication:**

* Messaging Service interacts with Authentication to verify token for sender and receiver.
* Appointment Service checks property and seller validity via Property Service.

### **4. Create/Edit Property Listing (Seller Flow)**

**Modules:**

* Property Service

**Flow:**

* On create/edit, frontend submits listing data to Property Service with seller's auth token.
* Property images upload via dedicated endpoints/files service and linked in Property Service.

**Service Communication:**

* Property Service authenticates user token via Authentication Service (direct or JWT validation).

### **5. Appointment Booking & Management**

**Modules:**

* Appointment Service
* Property Service

**Flow:**

* To book, the buyer selects a property – frontend sends booking data to Appointment Service (includes property\_id, buyer\_id, seller\_id, date/time).
* Appointment Service verifies user/token (calls Authentication Service) and property validity (calls Property Service).
* Appointment data is shown under the buyer's/seller's dashboards using Appointment Service queries, with relevant enrichment from Property and Profile Services.

### **6. In-App Messaging Screen**

**Modules:**

* Messaging Service
* Authentication Service
* Profile Service

**Flow:**

* **List of conversations:** Messaging Service fetches them per user.
* **Messages within a conversation:** Fetched from Messaging Service.
* **For displaying the user avatars/names:** Messaging Service queries Profile Service by sender\_id/receiver\_id.

**Service Communication:**

* Messaging Service validates tokens with Authentication.

### **7. Review & Rating Screen**

**Modules:**

* Review Service
* Authentication Service
* Profile/Property Services

**Flow:**

* **Listing property or seller reviews:** Frontend queries Review Service (by property\_id or seller\_id).
* **Posting a review:** Authenticated user posts new review to Review Service, which checks validity (ownership, previous transactions) with Authentication, Appointment, Property Services.

**Service Communication:**

* Review Service may check transaction history with Appointment Service and user identity with Authentication.

### **8. User Profile Screen & Verification**

**Modules:**

* Profile Service
* Authentication Service
* Document (Profile) Service

**Flow:**

* Fetch user profile using Profile Service (by user\_id, authenticated).
* To upload/verify documents: Call Document Service sub-endpoints in Profile microservice.

**Service Communication:**

* Profile Service may check authentication tokens.

## **Technologies Used**

### **Backend Technologies:**

* **Java Spring Boot 3.x:** Core framework for building robust REST APIs and microservices
* **Spring Security:** Authentication and authorization management with JWT token implementation
* **Spring Data JPA:** Object-relational mapping and database operations
* **Spring Web:** RESTful web services development
* **Maven:** Dependency management and project build tool
* **Hibernate:** ORM framework for database interactions

### **Frontend Technologies:**

* **Next.js 15:** React-based framework for server-side rendering and static site generation
* **React 18:** Component-based user interface development
* **TypeScript:** Type-safe JavaScript for better code quality and developer experience
* **Tailwind CSS:** Utility-first CSS framework for responsive design
* **Axios:** HTTP client for API communication

### **Database:**

* **MySQL 8.0:** Primary relational database for data storage
* **MySQL Workbench:** Database design and administration tool

### **Development Tools:**

* **IntelliJ IDEA / VS Code:** Integrated development environments
* **Postman:** API testing and documentation
* **Git:** Version control system
* **Docker:** Containerization for development and deployment

## **Expected Outcome**

### **Functional Outcomes:**

**User Management:**

* Secure user registration and authentication system
* Profile management with document verification capabilities
* Role-based access control for different user types

**Property Management:**

* Comprehensive property listing system with image uploads
* Advanced search and filtering functionality
* Detailed property information display with virtual tour support
* CRUD operations for property listings

**Communication Features:**

* Real-time messaging system between buyers and sellers
* Conversation history and message management
* User-friendly chat interface

**Appointment System:**

* Seamless appointment booking for property viewings
* Calendar integration for scheduling management
* Appointment history and status tracking

**Review and Rating System:**

* Property and seller rating functionality
* Review management and display system
* Trust score calculation based on user feedback

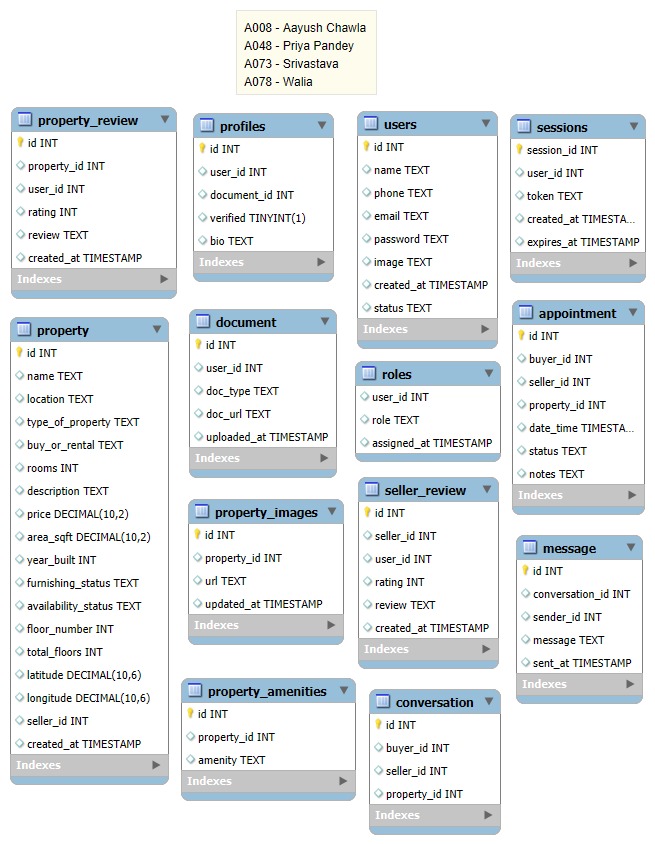
### **Technical Outcomes:**

**Architecture:**

* Scalable microservices architecture with clear service boundaries
* Well-defined API contracts between services
* Efficient database design with proper indexing and relationships

**Security:**

* Secure authentication and authorization implementation
* Data encryption and secure communication protocols
* Input validation and SQL injection prevention



### 