# **Online Property Rental System**

( A Full-Stack Web Application for Secure Property Rentals )

Name: Ewomazino Great Ekwigbedi

**Placement:** Intermediate

**Role:** Backend Developer

**Date:** 9/4/2025

# **Overview:**

- A platform for landlords to list properties and for tenants to search, book, and manage rentals.
- Secure JWT-based authentication with separate dashboards for property owners and tenants.
- Multi-image upload using Cloudinary and responsive UI built in React.

# **Project Architecture**

# **\*** Backend:

- Java Servlets for business logic (registration, login, listing management, booking, etc.)
- **PostgreSQL** for data persistence (users, listings, bookings)
- **JWT Authentication** for stateless and secure sessions

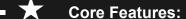
#### **★** Frontend:

- React for building a dynamic, responsive Single-Page Application (SPA)
- Axios for REST API calls
- React Router for client-side navigation

# **★** Cloud Integration:

Cloudinary for robust multi-image upload and hosting

#### **Core Features**



- Secure registration and login using JWT
- Role-based access: tenants and property owners

## **X** Property Listing Management:

- Support for multi-image uploads with a gallery view on the listing details page
- Landlord functionality to create, edit, and delete property listings

#### **★** Booking System:

- Tenants can search listings, send booking requests, and cancel bookings
- Landlords can manage booking requests (approve, decline, terminate)

#### **Weak of the Second Sec**

- **Tenant Dashboard:** Browse listings, search, filter, sort, and manage bookings
- o **Owner Dashboard:** Manage property listings, view and respond to booking requests

#### **★** Profile & Communication:

• User profile management for updating personal information

# **Tools & Technologies**

#### **Backend:**

- Java Servlets (Maven Webapp)
- PostgreSQL
- JSON Web Tokens (JWT)
- Cloudinary (for image upload and storage)

# Frontend:

- React
- Axios
- React Router

#### Other:

- IntelliJ IDEA (IDE)
- Git (Version Control)
- Apache Commons IO (for file I/O in Java)

# **Challenges Faced**

#### **Setting Up the Environment:**

- Issues with project structure in IntelliJ and Maven (e.g., marking source directories).
- Configuring the backend (Java Servlets) to work with PostgreSQL, JWT authentication, and Cloudinary.

#### File Upload Challenges:

- Integrating multi-image upload using Cloudinary.
- Handling InputStream conversions, especially with Java versions (e.g., readAllBytes() vs. Apache Commons iO).

#### **CORS & Routing Issues:**

- Troubleshooting CORS errors between the backend and React frontend.
- Ensuring the REST endpoints match the routes used on the frontend.

#### **Data Parsing & UI State:**

- Correctly processing a comma-separated image URL string on the frontend.
- Ensuring booking status persisted correctly across refreshes.

#### **How I Solved Them & Lessons Learned:**

#### **Environment Setup:**

- Resolved IntelliJ issues by correctly marking the source directories, learning the Maven project structure.
- Learned the importance of reading documentation and using community examples.

#### • File Upload Handling:

- Switched from Java 9's readAllBytes() to Apache Commons IO's IOUtils.toByteArray() for compatibility.
- Implemented proper logging at each stage of the upload to determine where issues occurred.
- This taught me to debug methodically and validate each integration point (API credentials, file inputs, and logs).

#### • CORS & Routing:

- Configured a global CORS filter and matched endpoint paths exactly.
- Learned that paying attention to request headers and origin settings is crucial for cross-domain communication.

#### **How I Solved Them & Lessons Learned:**

#### • Data Parsing on the Frontend:

- Enhanced the component to handle different data types (string vs. array) and added console logs to confirm values.
- Learned that good logging and defensive coding on the UI significantly eases troubleshooting.

#### • Overall Project Management:

- Each challenge taught me the importance of incremental testing, robust logging, and validating assumptions across the entire stack.
- I learned new libraries (like Cloudinary and Apache Commons IO) and integrated them into a cohesive system, adding to my professional skill set.

#### **Future Enhancements**

# • Online Payment Integration:

 Enable online rent payment with payment gateway integration.

# • Advanced Messaging System:

 Implement in-app messaging between tenants and landlords.

# • Enhanced Analytics:

 Detailed property analytics and booking trends.

# • Mobile App:

Develop a mobile version of the platform.

