Project: Online Rental Property Management System

1. Introduction

This document outlines the Low-Level Design (LLD) for an **Online Rental Property Management System**, which allows property owners to list rental properties, tenants to search and apply for rentals, and property managers to track lease agreements and payments.

The system supports both **Spring Boot (Java)** and **ASP.NET Core (.NET)** for backend development.

2. Module Overview

2.1 Property Listing Module

- Enables property owners to list available rental properties.
- Supports uploading property images, descriptions, and rental terms.

2.2 Tenant Management Module

- Allows tenants to register, search for properties, and submit rental applications.

2.3 Lease Management Module

- Facilitates lease agreement creation and digital signatures.
- Manages lease renewal and termination processes.

2.4 Rent Payment Module

- Enables online rent payments via multiple payment methods.
- Tracks payment history for tenants and landlords.

2.5 Maintenance Request Module

- Allows tenants to request property maintenance and track resolution.
- Enables landlords to assign maintenance tasks to service providers.

3. Architecture Overview

3.1 Architectural Style

- **Frontend:** Angular or React.
- **Backend:** REST API-based architecture.

- **Database:** Relational Database (MySQL/PostgreSQL/SQL Server).

3.2 Component Interaction

- Frontend interacts with the backend through REST APIs.
- Backend connects to the relational database for data storage and retrieval.

4. Module-Wise Design

4.1 Property Listing Module

4.1.1 Features

- Property owners can list and update rental properties.
- Tenants can search for properties using filters like location, price, and size.

4.1.2 Entities

- **Property**
  - PropertyID
  - OwnerID
  - Address
  - RentAmount
  - AvailabilityStatus

4.2 Tenant Management Module

4.2.1 Features

- Tenants can register, create profiles, and apply for rentals.
- Landlords can review applications and approve or reject tenants.

4.2.2 Entities

- **Tenant**
  - TenantID
  - Name
  - ContactDetails
  - RentalHistory

4.3 Lease Management Module

4.3.1 Features

- Digital lease agreement generation and signing.
- Lease renewal and termination functionality.

4.3.2 Entities

- **Lease**
  - LeaseID
  - PropertyID
  - TenantID
  - StartDate
  - EndDate
  - RentAmount

4.4 Rent Payment Module

4.4.1 Features

- Online rent payment through credit/debit cards, bank transfers, and wallets.
- Late payment notifications and reminders.

4.4.2 Entities

- **Payment**
  - PaymentID
  - LeaseID
  - Amount
  - PaymentDate
  - Status

4.5 Maintenance Request Module

4.5.1 Features

- Tenants can submit maintenance requests.
- Landlords can assign tasks to service providers.

4.5.2 Entities

- **MaintenanceRequest**
  - RequestID

    - PropertyID
    - TenantID
    - IssueDescription
    - Status

5. Deployment Strategy

5.1 Local Deployment

- Frontend and backend deployed on developer machines for initial testing.

5.2 Testing Environments

- Use containerized setups for staging environments to ensure consistency.

6. Database Design

6.1 Tables and Relationships

- **Property**: Primary Key: PropertyID, Foreign Key: OwnerID.
- **Tenant**: Primary Key: TenantID.
- **Lease**: Primary Key: LeaseID, Foreign Keys: PropertyID, TenantID.
- **Payment**: Primary Key: PaymentID, Foreign Key: LeaseID.
- **MaintenanceRequest**: Primary Key: RequestID, Foreign Key: PropertyID, TenantID.

7. User Interface Design

7.1 Wireframes

- **Home Page**: Lists featured rental properties.
- **Property Details Page**: Displays property information and lease terms.
- **Tenant Dashboard**: Shows active leases, rent payments, and maintenance requests.

8. Non-Functional Requirements

8.1 Performance

- Capable of handling 1,000 concurrent users browsing properties.

8.2 Usability

- User-friendly interface for tenants and landlords.

8.3 Security

- Secure payment processing and encrypted tenant data.

8.4 Scalability

- Supports expansion to multiple cities and property types.

9. Assumptions and Constraints

9.1 Assumptions

- Property owners and tenants will access the system via the internet.

9.2 Constraints

- Initial phase targets only residential rentals, with commercial property support in future updates.