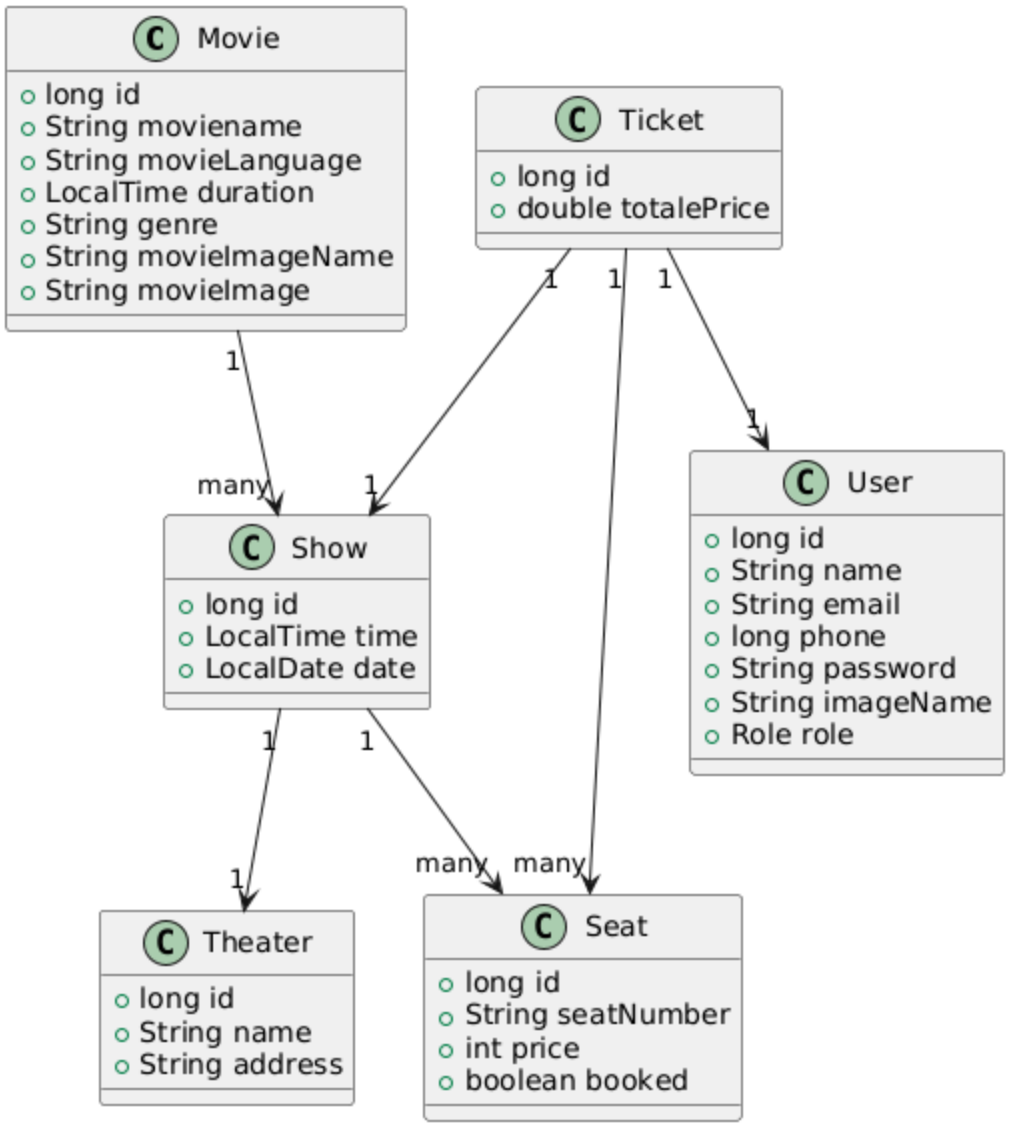
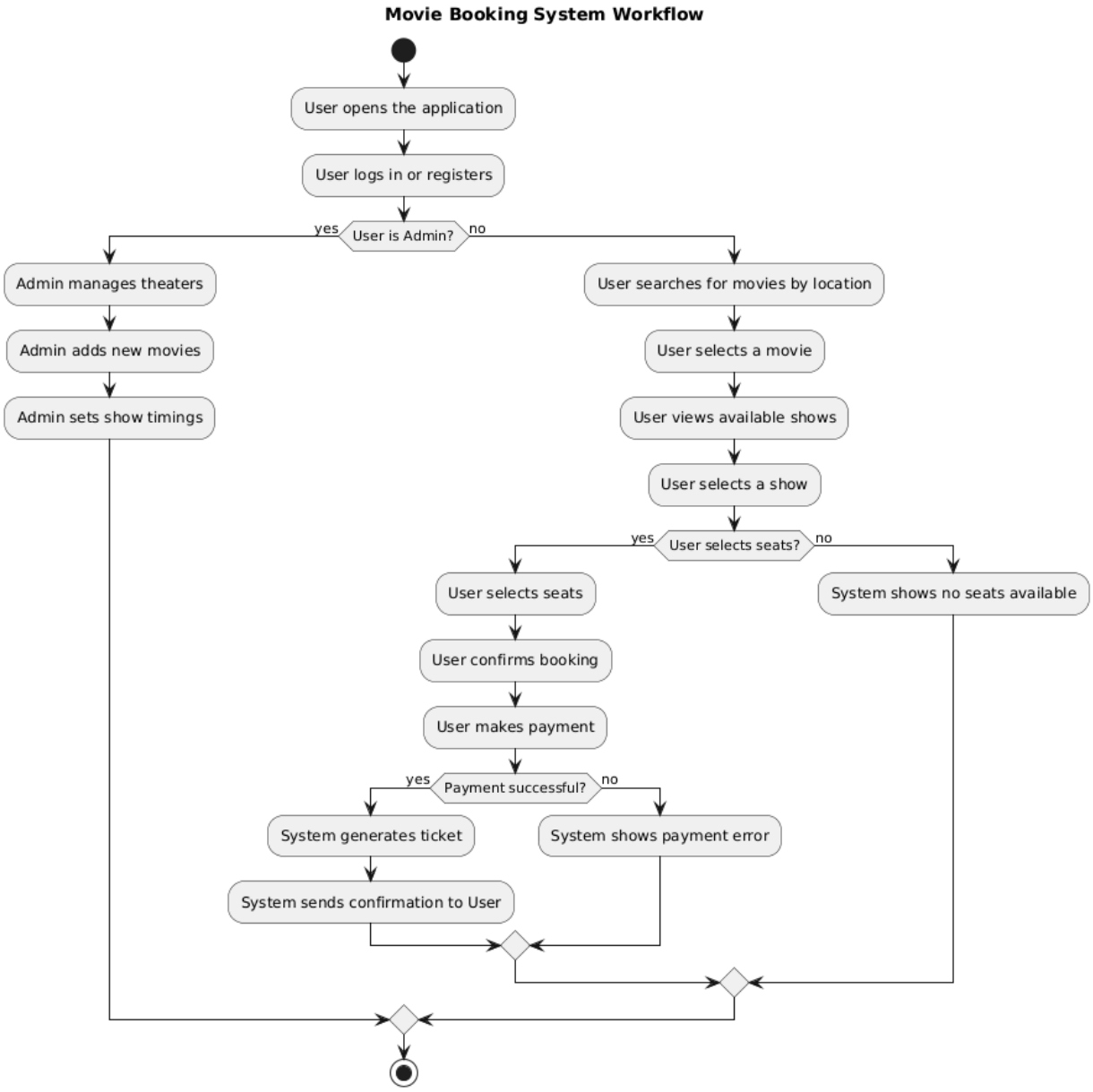
1. **User Registration/Login:**
    - The user creates an account or logs in with their credentials. They provide details like their name, email, phone, and password, which will be stored in the `User` entity.
2. **Browse Movies:**
    - The logged-in user can browse a list of available movies (`Movie` entity). Each movie displays relevant information such as its name, genre, language, duration, and movie poster.
3. **Select a Show:**
    - After selecting a movie, the user is shown the available show timings from the `Show` entity, which are tied to a particular `Theater` entity. Each show has a scheduled time (`time`), date (`date`), and is associated with a specific theater.
4. **Choose a Seat:**
    - The user selects a show and is presented with the available seats (`Seat` entity). Each seat has a seat number (`seatNumber`), price (`price`), and booking status (`booked`).
5. **Booking Confirmation:**
    - Once the user selects their seats, a `Ticket` entity is created that holds details like the total price (`totalePrice`), selected show, and associated user.
    - The selected seats are marked as booked, and a confirmation is generated.
6. **Payment and Final Confirmation:**
    - The user completes the payment for the ticket.
    - Upon successful payment, the ticket is confirmed, and details are stored in the `Ticket` entity. The system marks the selected seats as booked in the `Seat` entity.

Detailed Workflow Steps

1. **User Registration & Login**
    - Input: `User` entity
    - Workflow:
        - User registers or logs in.
        - If registration, `User` entity is populated with user details (e.g., name, email, password).
        - On login, the system validates the user credentials and grants access.
2. **Movie Browsing**
    - Input: `Movie` entity
    - Workflow:
        - System fetches the list of available movies.
        - Displays movie details from the `Movie` entity (moviename, genre, movieImage).

3. **Show Selection**
    - Input: `Show` entity
    - Workflow:
        - After the user selects a movie, the system displays all available shows linked to that movie.
        - Each show is connected to a `Theater` and has attributes like time, date, and associated movie.
4. **Seat Selection**
    - Input: `Seat` entity
    - Workflow:
        - For the selected show, the system fetches available seats (`Seat` entity).
        - User chooses a seat, and the seat status is updated to `booked = true`.
5. **Ticket Creation and Booking**
    - Input: `Ticket`, `Seat`, and `User` entities
    - Workflow:
        - Once the user selects seats, the total price is calculated, and a `Ticket` is generated.
        - The selected seats are linked to this ticket, and the user is linked as the ticket owner.
6. **Final Payment**
    - Input: `Ticket` and `User` entities
    - Workflow:
        - The user proceeds to payment.
        - Upon successful payment, the system confirms the booking, and the `Ticket` entity is finalized with details of the show, seats, and the user.


Movie
long id
String moviename
String movieLanguage
LocalTime duration
String genre
String movieImageName
String movieImage
Ticket
long id
double totalePrice
User
long id
String name
String email
long phone
String password
String imageName
Role role
Show
long id
LocalTime time
LocalDate date
Theater
long id
String name
String address
Seat
long id
String seatNumber
int price
boolean booked
1
many
1
1
1
1
1
1
1
1
1
1
many
many



High-Level Design (HLD)

1. **Project Overview**
    - Purpose and Scope of the Movie Booking System
2. **System Architecture**
    - Overview of System Components and Interactions
3. **Technology Stack**
    - Frontend: React (or any other chosen framework)
    - Backend: Spring Boot, Java

    - Database: MySQL/PostgreSQL
4. **Core Functional Modules**
    - User Management (Registration, Login)
    - Movie and Show Management
    - Theater and Seat Management
    - Booking and Ticketing System
    - Payment Gateway Integration
5. **Database Design**
    - Key Entities and Relationships
    - ER Diagram
6. **Security Considerations**
    - User Authentication and Authorization
    - Data Encryption for sensitive information
7. **Scalability and Performance**
    - System Scaling Strategy
    - Caching and Load Balancing Mechanisms
8. **External Integrations**
    - Payment Gateway Integration
    - Notification Services (Email/SMS for booking confirmation)
9. **API Overview**
    - API Endpoints for Frontend-Backend Communication

Low-Level Design (LLD)

1. **Class Diagrams**
    - Entity Relationships (Movie, Show, Seat, Theater, Ticket, User)
2. **Detailed Component Design**
    - Detailed Description of Core Components (Controllers, Services, Repositories)
3. **Database Schema**
    - Detailed SQL Table Structure and Queries
4. **Sequence Diagrams**
    - Workflow for User Registration, Movie Selection, Show Booking, and Ticketing
5. **Flowcharts**
    - Flowcharts for Core Processes (Seat Selection, Ticket Generation)
6. **API Design**
    - Request and Response Formats for all API Endpoints
7. **Service Layer Design**
    - Details of Service Classes for Business Logic
8. **Error Handling Mechanism**
    - Strategy for Exception Handling and Logging
9. **UI/UX Design Considerations**
    - Mockups or Wireframes (if available)
    - User Flow for Movie Booking