Software Requirements Specification

for

TicketHub

Version 1.0

Prepared by Project Team 18

Name	PRN
Akshay Anil Nalkol	240840320015
Farah Deeba Khan	240840520025
Yugandhar Sudhir Deshmukh	240840320134
Harsh Kumar Bharti	240840320039
Shruti Sopan Bangar	240840520077

Centre for Development of Advanced Computing

Raintree Marg, Near Bharati Vidyapeeth, Opp. Kharghar Railway Station, Sector 7, CBD
Belapur, Navi Mumbai - 400 614 - Maharashtra (India)
Phone: +91-22-27565303
Fax: +91-22-2756-0004

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of this document is to provide a comprehensive description of **TicketHub**, a Ticket Booking and Management System. This document outlines the functional and non-functional requirements for the system and provides a detailed description of the system's functionalities, such as ticket booking, cancellation, event management, and user notifications. The SRS document also outlines the performance and security requirements for the system, ensuring a seamless, secure, and user-friendly experience for both end-users and administrators.

1.2 Intended Audience and Reading Suggestions

The document is intended for a broad audience, including developers, event organizers, and end-users.

For developers, the SRS document serves as a comprehensive guide for the design and development of the system. It outlines the functional and non-functional requirements, performance expectations, and security considerations, providing a clear understanding of the scope of the project. Developers should read the document carefully to ensure that their work aligns with the specified requirements and meets the expectations of all stakeholders.

For end-users or customers, the SRS document provides an overview of the system's capabilities, such as ticket booking, movie browsing, and notifications, helping them understand how to use the system effectively. They should read the document to gain a general understanding of the system's functionalities and how they can benefit from it.

For event organizers, the SRS document outlines the goals and objectives of the project and provides a clear understanding of the system's capabilities, helping them to determine whether the system is suitable for their needs. The document should be read by relevant decision-makers to ensure that the system meets their requirements and expectations.

1.3 Project Scope

Development of a platform for users to browse movies, book tickets, and for theater owner to manage their events seamlessly.

- Implementation of user authentication and authorization to ensure secure access and personalized experiences for users and theater owner.
- Integration of a booking and cancellation process that allows users to book, reschedule, or cancel tickets in real-time with automated updates on ticket availability.
- Implementation of a user-friendly interface for both end-users and theater owner to interact with the system intuitively.
- Development of a commenting/review system where users can leave feedback on events, with real-time updates of post time and date.
- Integration of automatic notification systems to keep users informed about booking confirmations, cancellations, or upcoming events.
- Additional Features: dynamic seat allocation, movie analytics for theater owner, and daily notifications about movie status or updates.

- Development of a secure system to protect user data, ticketing information, and payment details.

- Deployment of the platform on AWS to ensure high availability, scalability, and performance for all users.

1.4 References

- https://spring.io/
- https://www.javascript.com/
- https://developer.mozilla.org/en-US/docs/Web/JavaScript
- https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller
- Spring Boot + React: JWT Authentication with Spring Security - BezKoder

2. Overall Description

2.1 Product Perspective

The TicketHub System is designed to address the challenges faced in traditional methods of event ticket booking and management. With the increasing popularity of movies and the growing demand for efficient and streamlined ticketing solutions, the need for a modern and technology-driven system has become imperative.

The TicketHub System aims to provide a centralized platform where users can easily browse movies, book tickets, and receive real-time updates, while theater owner can efficiently manage their movies and ticket sales. By streamlining the ticket booking and movie management process, the project will enhance the overall user experience for attendees and improve the operational efficiency of theater owners.

2.2 Product Features

The following are the key features and requirements of the app:

- User authentication and authorization: This feature will allow users to log into the system using their credentials, ensuring secure access. The system will determine their access level (e.g., user, theater owner, admin) and permissions accordingly
- Movie Ticket Booking: Users can browse through available movies, view details, and book tickets. The app will also include real-time seat availability updates to enhance the booking experience.
- Movie Management: Theater owners can create, edit, and manage movies, including specifying movie details such as dates, venue, ticket pricing, and seating arrangements.
- Notification System: This feature will send real-time notifications to users, informing them of ticket bookings, cancellations, upcoming movie reminders, or updates to movie details.
- Ticket Cancellation and Refunds: Users will be able to cancel booked tickets within the allowed time frame, with automated refunds processed according to movie policies.
- Movie Insights and Analytics: Movie owners and administrators can generate detailed reports to track ticket sales, movie attendance, and overall performance metrics.
- User Management: Administrators will have the ability to manage all user accounts, including adding, editing, or removing users, and assigning specific roles or permissions (e.g., theater owner, admin).
- Data security and privacy: The system will implement robust measures to ensure the secure storage and transfer of user and payment data, complying with privacy regulations and industry standards.

2.3 User Classes and Characteristics

The TicketHub System has three main user classes:

1. **Customers:**
    - Characteristics: They are the primary users of the system and use it to browse events, book tickets, and manage their bookings.
    - Requirements: Customers should be able to log in to the system, view their

dashboard which displays upcoming events, booking status, and other relevant information. They should also be able to search for events, select tickets, make payments, and view booking history.

2. **Theater Owner:**
    - Characteristics: They are responsible for creating and managing movies and tracking ticket sales.
    - Requirements: Theater owner should be able to log in to the system, create and update movies with details like movie name, venue, date, time, and ticket pricing. They should also have access to real-time analytics, view booking statistics, and manage customer feedback or inquiries related to their movies.

3. **Admins:**
    - Characteristics: They are responsible for overseeing the system, managing users, and ensuring smooth operation
    - Requirements: Admins should have the ability to create and manage user accounts (both customers and theater owner), monitor all movies and ticket transactions, generate system-wide reports, and ensure compliance with data security and operational guidelines.

2.4 Operating Environment

The operating environment for the TicketHub project consists of the following hardware and software components:

Hardware:

- A machine with at least 8GB of RAM and a fast processor, such as Intel Core i5 or higher, to ensure smooth and efficient execution of the application.

Software:

- ReactJS for the frontend development.
- Spring Boot & MS.NET for the backend development
- MySQL for the database management.
- JWT Authentication for security.
- AWS for deployment and scalability.

Other Applications:

- Code editor (such as Visual Studio Code, Eclipse, STS, Visual Studio)
- Git version control software
- Command line interface (CLI) or terminal or terminal for project operations.
- A browser for testing the application.
- Postman for testing APIs.
- MySQL Workbench or another database management tool
- An AWS account for deployment.

2.5 Design and Implementation Constraints

- User interface is only in English. No other language option is available.
- Compatibility challenges may arise while integrating the frontend with the backend.
- Parallel operations such as simultaneous ticket booking may cause temporary performance lags.
- Limited to HTTPS.

2.6 User Documentation

User documentation will primarily consist of a Help menu within the application. This menu will provide detailed explanations about the project and its functionalities. If a user has any questions regarding any module or functionality, they can refer to the Help menu to understand how to operate the application. Additionally, this report serves as a

comprehensive guide, detailing all aspects of the project, including its functionality, users, software used, hardware requirements, and operating environment.

2.7 Assumptions and Dependencies

Assumptions:

- Users are familiar with basic online ticket booking processes.
- Theater owners have access to the necessary tools for uploading and managing event details.
- Users will follow the specified guidelines for booking and cancellation policies.
- Theater owners are responsible for maintaining accurate movie details in the system.

Dependencies:

- A functional email service is required for notifications to be sent.
- Access to GitHub API to verify the URL of the project and retrieve the list of branches.
- An active internet connection is necessary to access the application and receive notifications.
- AWS deployment infrastructure.
- ReactJS, Spring Boot, Ms.NET and MySQL components are required to run the application.

3. System Features

3.1 User Registration and Login

3.1.1 Description and Priority

- Description: Users can register for a TicketHub account and log in using their unique credentials.
- Priority: High

3.1.2 Stimulus/Response Sequences

- A new user clicks the "Sign Up" button, enters their details (name, email, password, etc), and submits the form. The app creates an account and sends a confirmation email.
- Returning users click "Login," enter their credentials, and are redirected to their dashboard upon successful authentication.

3.1.3 Functional Requirements

- User authentication: Ensure that the user's credentials are valid and match those in the app's database.
- Session management: Keep track of the user's session and log them out after a certain amount of time or when they log out manually.
- Data encryption: Ensure that the user's password is encrypted before it is stored in the database.

3.2 Dashboard

3.2.1 Description and Priority

- Description: Users will see a personalized dashboard displaying ongoing and upcoming movies, past bookings, payment history. The dashboard also allows users to update their personal information.
- Priority: High

3.2.2 Stimulus/Response Sequences

- The user logs in to the website and is taken to their dashboard. The dashboard displays their upcoming bookings, past booking history, and options to update personal information or view event details.

3.2.3 Functional Requirements

- Display booking history: Provide a list of past bookings with details such as movie name, date, time, and payment status.
- Update personal information: Allow users to edit their personal details, such as name, contact number, email address, and preferences.
- Interactive interface: Include clickable links to view detailed movie information, access tickets, or modify bookings.

3.3 Movie Discovery

3.3.1 Description and Priority

- Description: Users can browse and search for movies based on location, type, or keywords.
- Priority: High

3.3.2 Stimulus/Response Sequences

- A user visits the homepage, uses the search bar, or filters events by categories (e.g., concerts, movies) and location. A list of matching movies is displayed.

3.3.3 Functional Requirements

- Provide a search bar to find events by name or keywords.
- Implement filters for categories, location, and movie dates.
- Display movie details such as title, venue, date, price, and description.

3.4 Ticket Booking

3.4.1 Description and Priority

- Description: Users can book tickets for movies, select seats (if applicable), and make payments securely.
- Priority: High

3.4.2 Stimulus/Response Sequences

- The user selects an movie, chooses seats, and clicks "Book Tickets." The app redirects them to the payment page. Upon successful payment, tickets are confirmed and emailed to the user.

3.4.3 Functional Requirements

- Enable seat selection for events with reserved seating.
- Provide multiple payment options (credit/debit cards, UPI, wallets).
- Generate and email ticket confirmations with QR codes for event entry.

3.5 Ticket Cancellation and Refunds

3.5.1 Description and Priority

- Description: Users can cancel their tickets and receive refunds as per the movie's refund policy.
- Priority: Medium

3.5.2 Stimulus/Response Sequences

- The user visits their booking history, selects a ticket, and clicks "Cancel." The system processes the cancellation and updates the refund status.

3.5.3 Functional Requirements

- Allow users to cancel tickets from their account.
- Calculate refunds based on the movie's cancellation policy.
- Update the booking status to "Cancelled" and notify users about refunds.

3.6 Notifications

3.6.1 Description and Priority

- Description: Notify users about upcoming movies, ticket confirmations, and cancellations via email or app notifications.
- Priority: Medium

3.6.2 Stimulus/Response Sequences

- The app sends reminders for movies a day before the movie date or updates users on changes in movie schedules.

3.6.3 Functional Requirements

- Send email and in-app notifications for ticket confirmations, reminders, and updates.

- Notify users about special offers and discounts.

3.7 Reporting and Analytics

3.7.1 Description and Priority

- Description: The system generates reports for movie performance, ticket sales, and user engagement.
- Priority: Low

3.7.2 Stimulus/Response Sequences

- Theater owner select a date range and movie from the reporting dashboard.

3.7.3 Functional Requirements

- Provide customizable filters for report generation.
- Include metrics like ticket sales, user demographics, and revenue.

3.8 Group Bookings with Discounts

3.8.1 Description and Priority

- Description: Users can book tickets for groups and receive discounts based on the number of tickets purchased.
- Priority: Low

3.8.2 Stimulus/Response Sequences

- A user selects multiple tickets for an movie, and the app calculates and applies group discounts before checkout.

3.8.3 Functional Requirements

- Implement dynamic pricing for group bookings
- Display the discount amount before payment.
- Allow users to share group booking links for others to join.

4. External Interface Requirements

4.1 User Interfaces

The TicketHub system shall provide a user-friendly and intuitive interface for all users (customers, theatre owners, and administrators).

- **Customers Interface:**
  - A homepage displaying featured movies, categories, and search options.
  - A dashboard for users to view their booked tickets, cancellation options, and notifications.
  - A booking interface with dynamic seat selection and movie details.
  - A feedback section for users to leave reviews and rate events.
  - Mobile-responsive design for seamless access across devices.
- **Theater Owner Interface:**
  - A dashboard for managing movie details, ticket pricing, and seating arrangements.
  - Real-time analytics for ticket sales, movie attendance, and performance.
  - A notifications panel to communicate updates with customers.
- **Admins:**
  - Tools for user management, including adding, editing, and removing user accounts.
  - Comprehensive reports of system-wide statistics, including revenue and user activity.
  - Monitoring tools to ensure system compliance and operational health.

4.2 Hardware Interfaces

- **Server Side:**
  - The web application will be hosted on a web server which is listening on the web standard port, port 5173
- **Client Side:**
  - Monitor: The software shall display dynamic and interactive content using HTML5 and CSS3.
  - Input Devices: The system shall interact with a keyboard and mouse for desktop users, and touchscreen interfaces for mobile and tablet users.

4.3 Software Interfaces

- **Frontend:**
  - ReactJS for building interactive and dynamic user interfaces.
  - Bootstrap CSS for a responsive and aesthetically pleasing design.
- **Backend:**
  - Spring Boot and MS.NET for managing business logic and ensuring secure communication between the frontend and database.
  - MySQL database for storing user, ticket, and event information.
- **APIs:**
  - Integration with third-party APIs for payment processing (e.g., Razorpay, PayPal).
  - Email and SMS services for real-time notifications and confirmations.

4.4 Communications Interfaces

- Communication between the client and server will use HTTPS for secure data transmission.
- Notifications will be delivered via email and SMS using an external notification service.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The performance requirements of the project should outline the expected speed, reliability, scalability, and efficiency of the system. This may include the maximum response time for user actions, the expected uptime percentage, the ability to handle irrespective for the increasing numbers of users and bookings, and the resource usage of the system. These requirements should be realistic and achievable, taking into account factors such as hardware limitations and network bandwidth. They should also be measurable and verifiable, allowing the system to be tested and evaluated against the defined standards.

5.2 Safety Requirements

The system should be reliable and have minimal downtime to ensure that users can book shows, movies, and events in a proper manner. The user interface is very friendly to prevent errors or misunderstandings while using the app. The app should have robust error handling mechanisms to deal with unexpected situations and prevent any harm to the user or loss of data.

5.3 Security Requirements

The project would include measures to protect the confidentiality, integrity, and availability of sensitive information such as user personal information. Some of the security requirements are:

- Authentication: A secure authentication system that ensures only authorized users can access the system.
- Authorization: An authorization mechanism to determine what actions a user can perform within the system based on their role and permissions.
- Encryption: Data in transit and at rest should be encrypted to protect sensitive information from being intercepted or accessed by unauthorized parties.
- Access Control: A mechanism to control access to sensitive information within the system, including the ability to set permissions for different users and roles.

5.4 Software Quality Attributes

The software quality attributes for the project include the following:

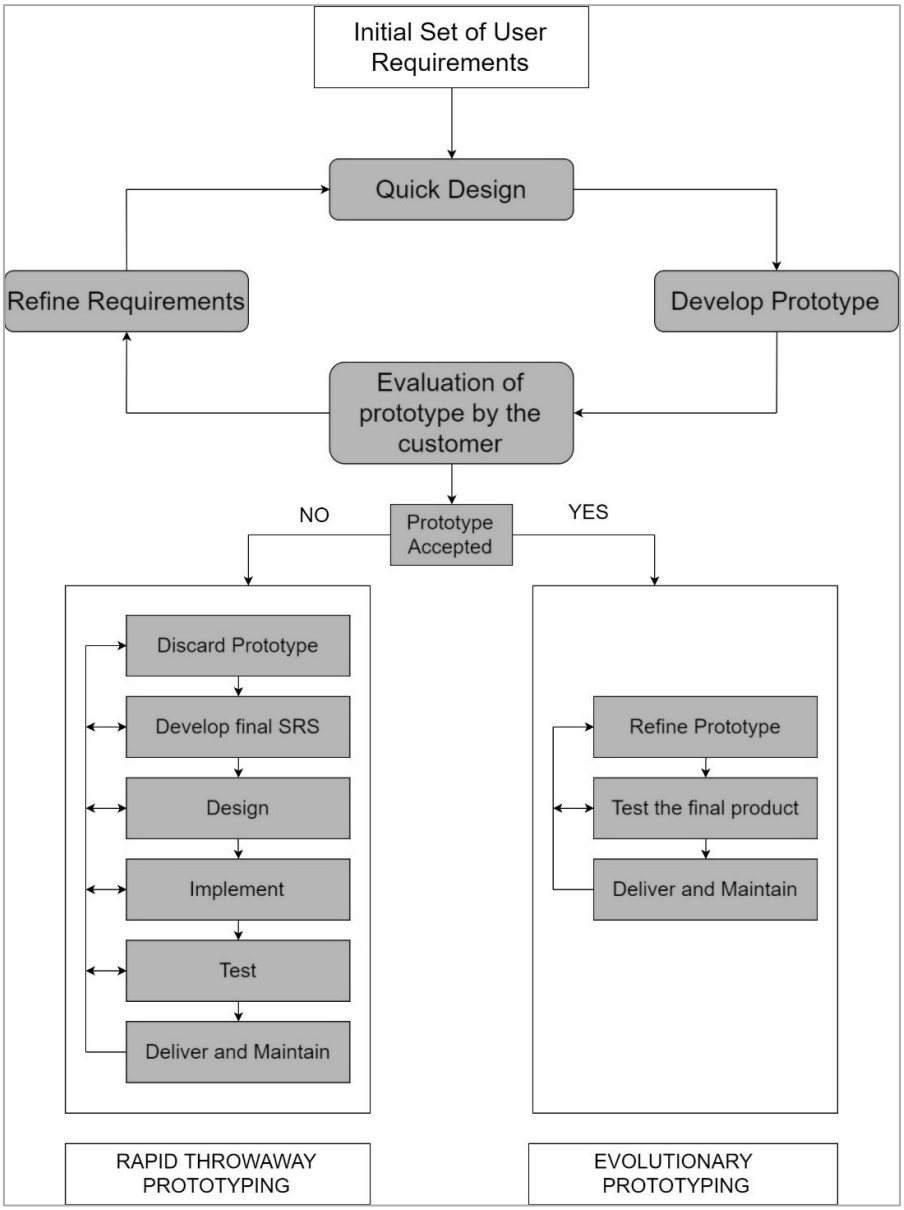
- Usability: The system shall follow a clean and intuitive design, ensuring users can complete actions (e.g., ticket booking) in a very ease manner.
- Reliability: The system shall handle for failed and other successful transactions or API calls.
- Scalability: The system shall accommodate a growing number of events and users without requiring significant architectural changes.
- Maintainability: The system should be maintainable, with easy-to-update code and well-documented processes, ensuring that updates and improvements can be made quickly and efficiently.
- Compliance: The system should comply with relevant regulations and standards, ensuring that it operates within the bounds of the law and industry best practices.

Appendix A: Glossary

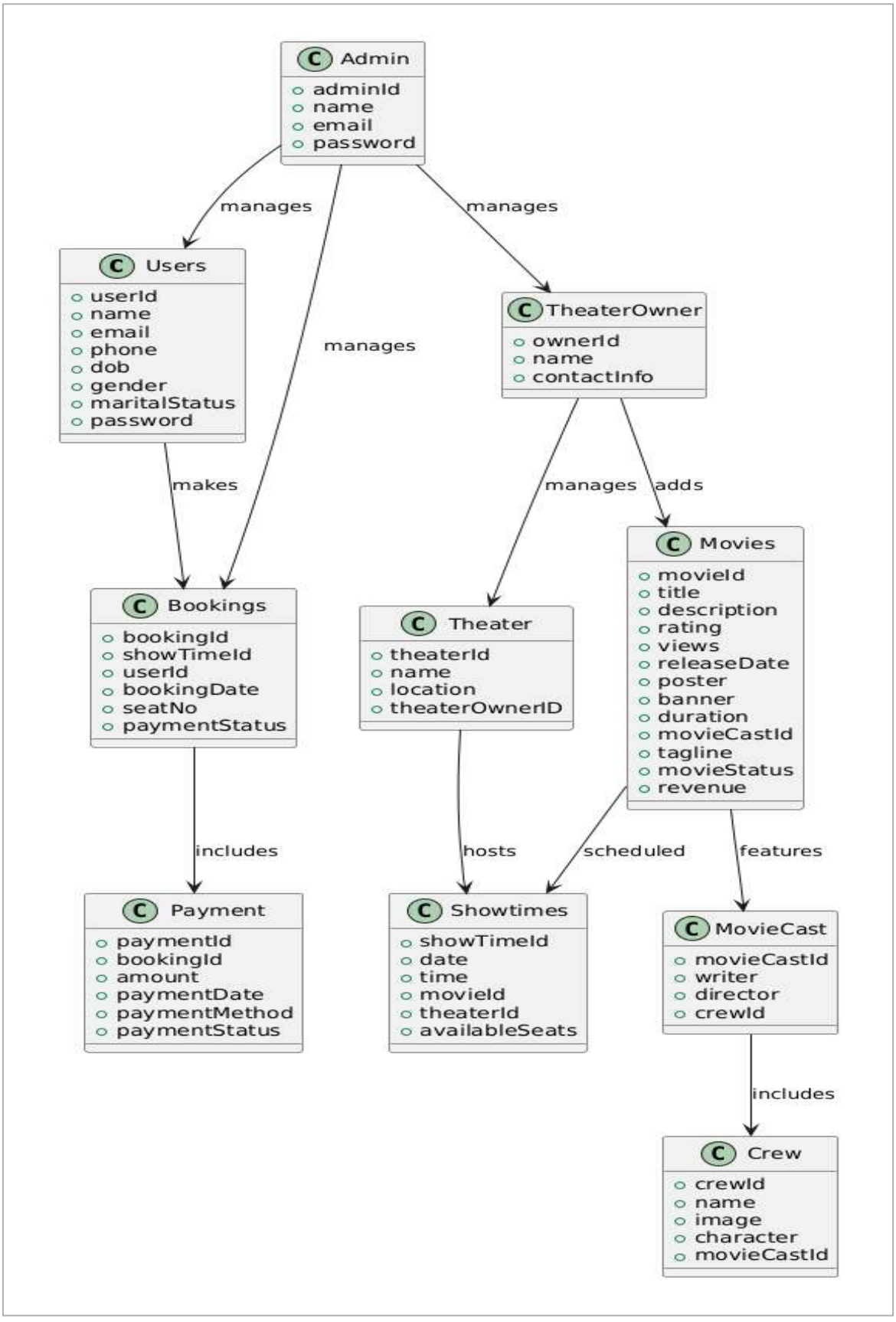
Sr. No.	Abbreviation	Full Form
1.	API	Application Programming Interface
2.	AWS	Amazon Web Service
3.	CLI	Command line Argument
4.	GB	Gigabyte
5.	HTML	Hypertext Markup Language
6.	HTTP / HTTPS	Hypertext Transfer Protocol / Hypertext Transfer Protocol Secure
7.	ID	Identification
8.	JS	JavaScript
9.	JWT	Java Web Token
10.	OS	Operating System
11.	RAM	Random Access Memory
12.	SQL	Structured Query Language
13.	SRS	Software Requirement Specification
14.	URL	Uniform Resource Locator
15.	ER	Entity Relationship

Appendix B: Analysis Models

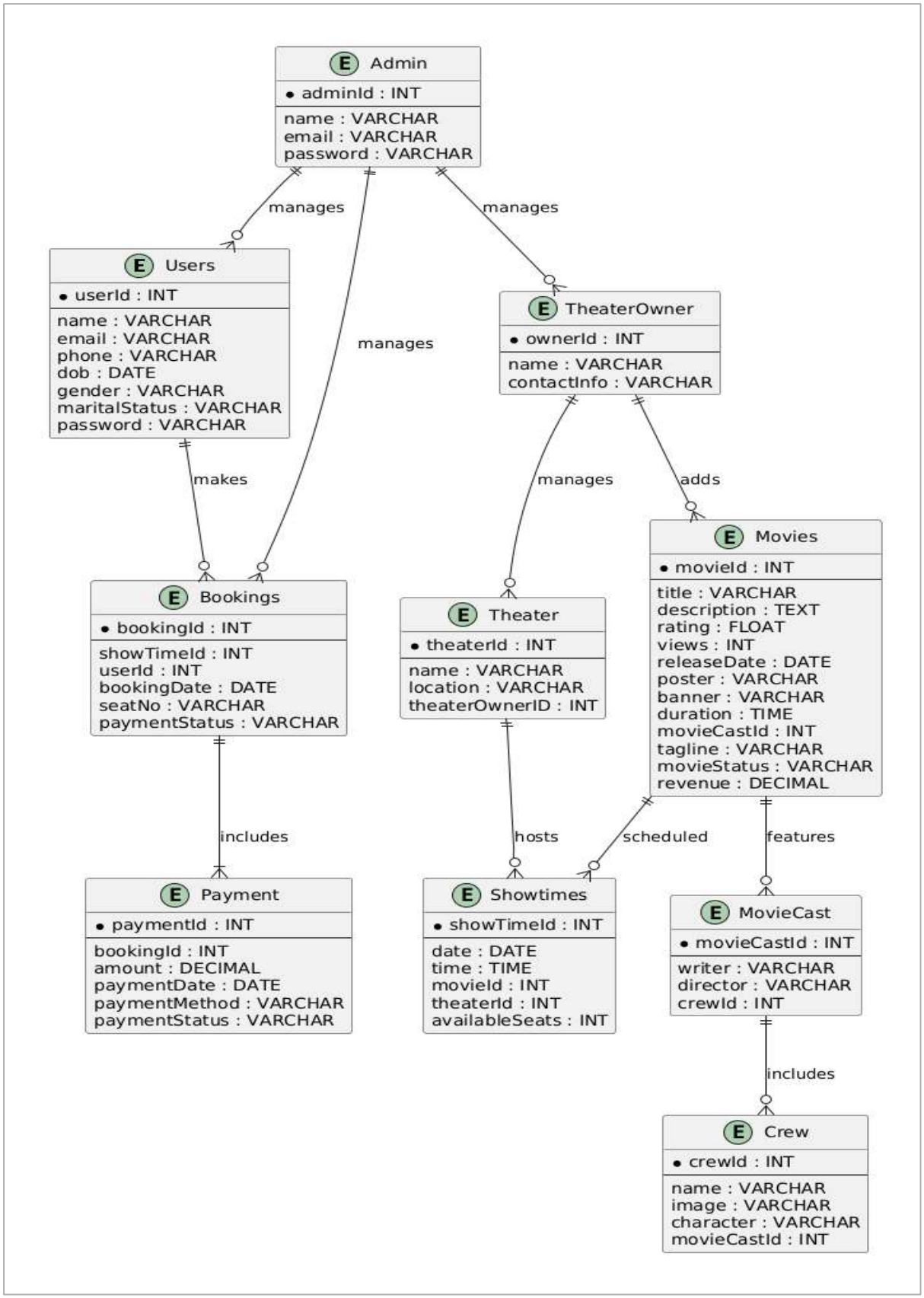
- **Software Development Approach in Our System**



- **Class Diagram**

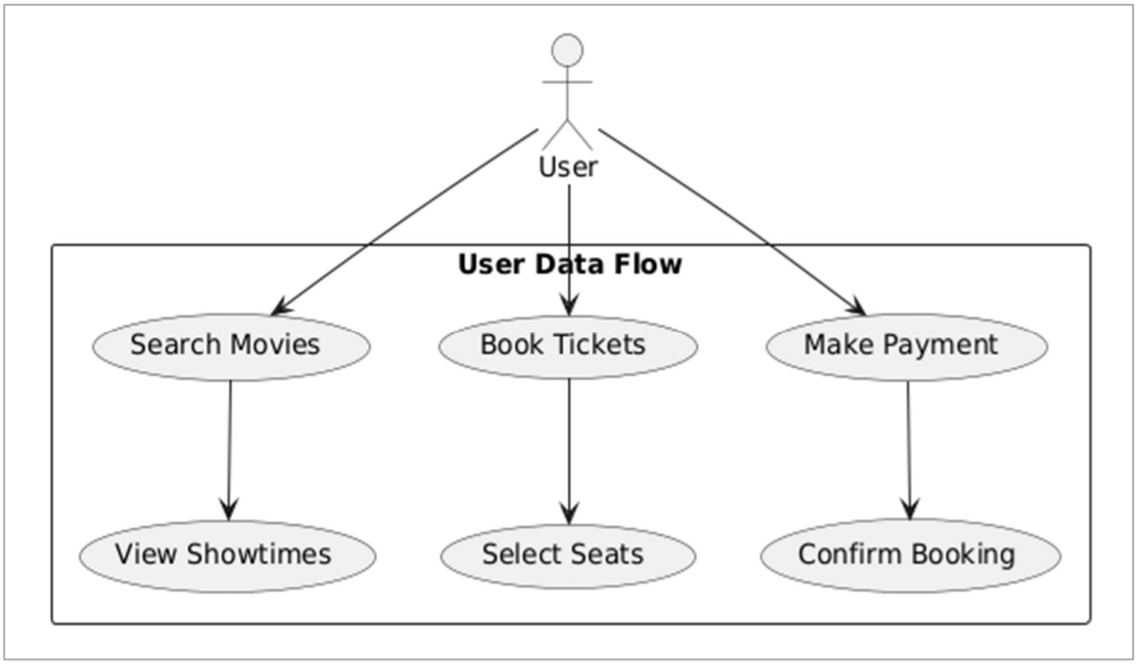


- **ER Diagram**

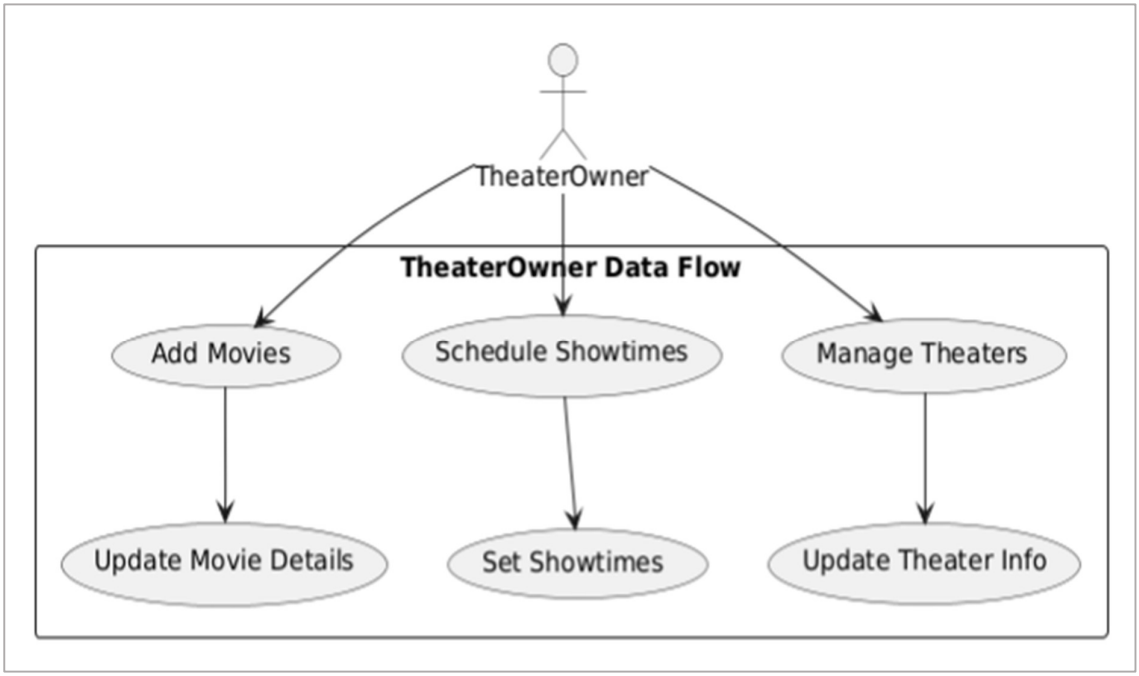

Admin
• adminId : INT
name : VARCHAR
email : VARCHAR
password : VARCHAR
manages
manages
manages
Users
• userId : INT
name : VARCHAR
email : VARCHAR
phone : VARCHAR
dob : DATE
gender : VARCHAR
maritalStatus : VARCHAR
password : VARCHAR
TheaterOwner
• ownerId : INT
name : VARCHAR
contactInfo : VARCHAR
makes
manages
adds
Movies
• movieId : INT
title : VARCHAR
description : TEXT
rating : FLOAT
views : INT
releaseDate : DATE
poster : VARCHAR
banner : VARCHAR
duration : TIME
movieCastId : INT
tagline : VARCHAR
movieStatus : VARCHAR
revenue : DECIMAL
Bookings
• bookingId : INT
showTimeId : INT
userId : INT
bookingDate : DATE
seatNo : VARCHAR
paymentStatus : VARCHAR
Theater
• theaterId : INT
name : VARCHAR
location : VARCHAR
theaterOwnerID : INT
includes
hosts
scheduled
features
Payment
• paymentId : INT
bookingId : INT
amount : DECIMAL
paymentDate : DATE
paymentMethod : VARCHAR
paymentStatus : VARCHAR
Showtimes
• showTimeId : INT
date : DATE
time : TIME
movieId : INT
theaterId : INT
availableSeats : INT
MovieCast
• movieCastId : INT
writer : VARCHAR
director : VARCHAR
crewId : INT
includes
Crew
• crewId : INT
name : VARCHAR
image : VARCHAR
character : VARCHAR
movieCastId : INT

- **Data-Flow Diagram**

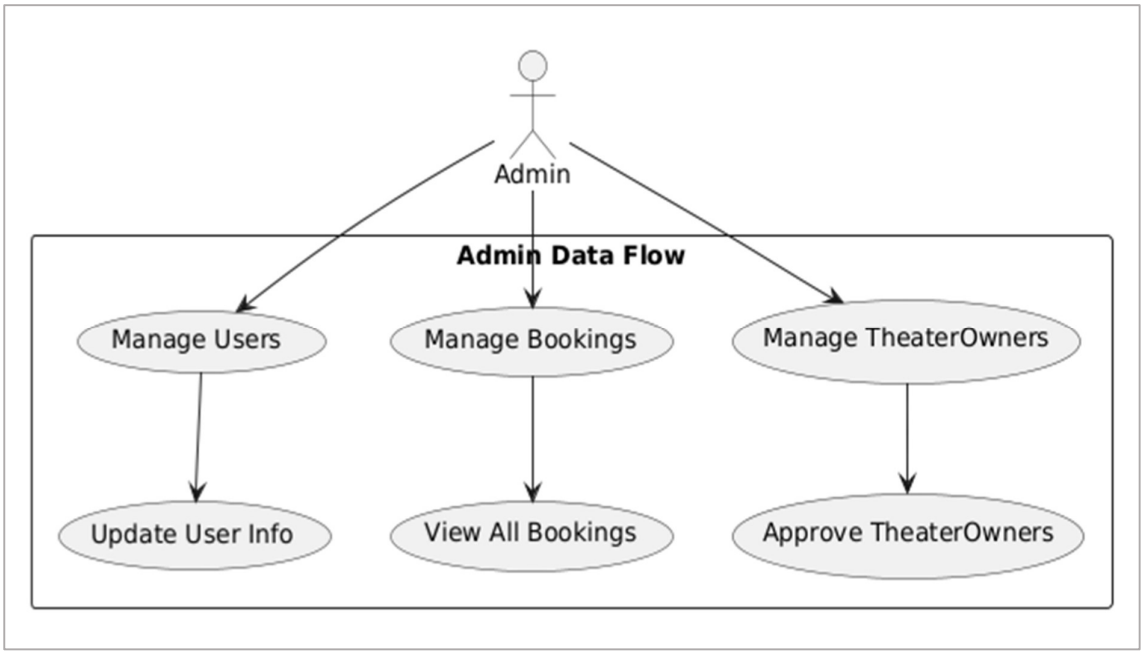
1. **User Module**



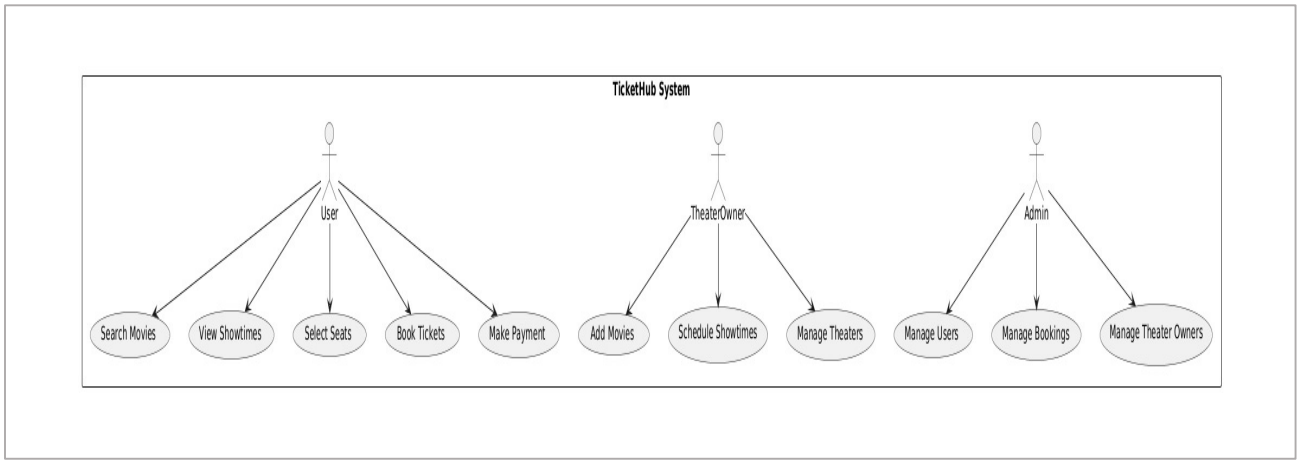
2. **TheaterOwner Module**



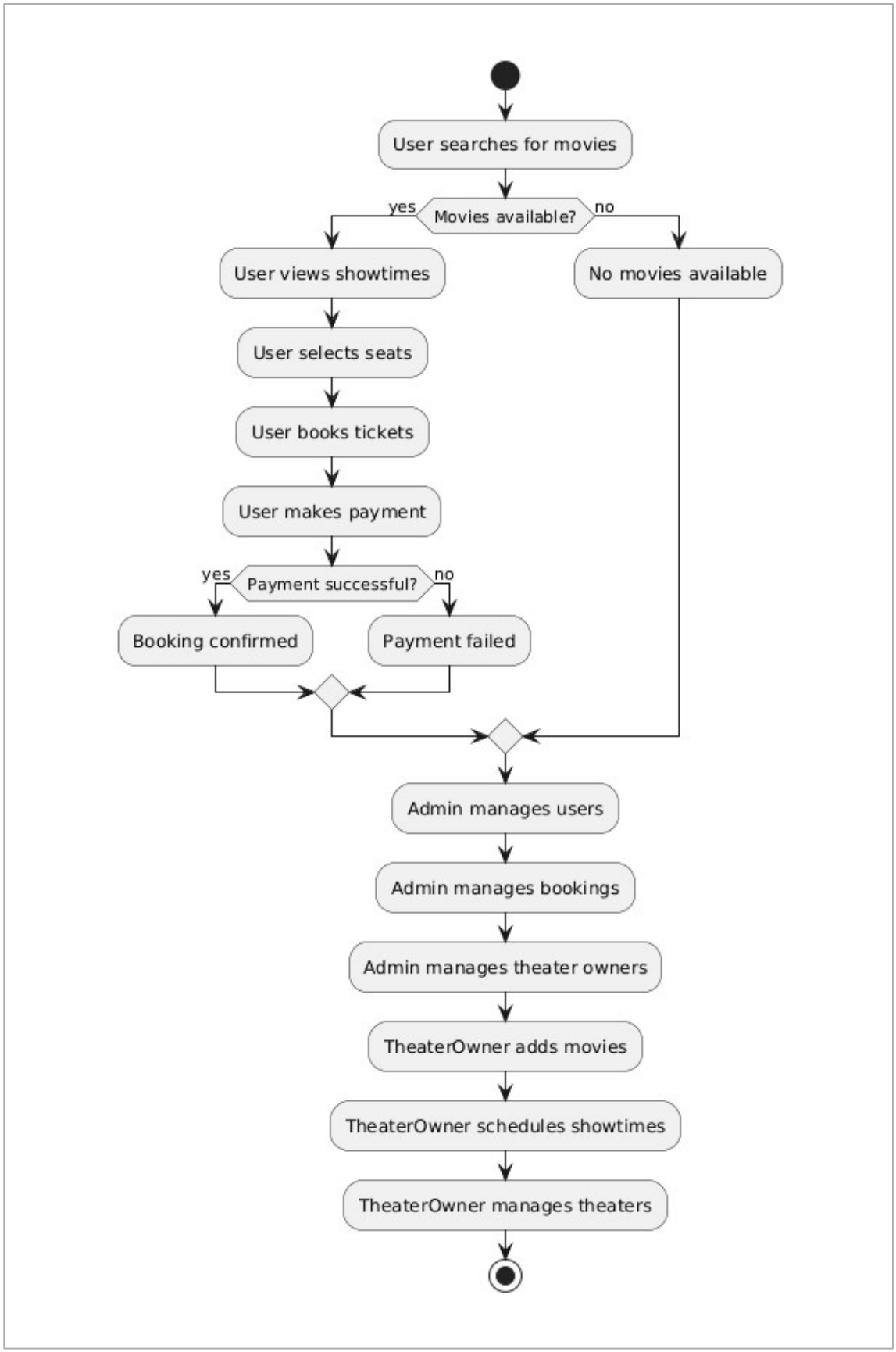
3. Admin Module



- **Use-Case Diagram**



- **Activity Diagram**



- **Sequence Diagram**