Jashore University of Science and Technology

Department of Computer Science and Engineering

Course Title: Software Development Project II

Course Code: CSE-3208

A Project Report On

Online Movie Ticket Booking Website (Popcorn)

Submitted to	Submitted by
Mostafijur Rahman Akhond **Assistant Professor** Dept. of Computer Science and Engineering. Jashore University of Science and Technology.	**Md Shojib Hossain** Student ID: 190103 **Rukhsana Khatun** Student ID: 190148 3rd year, 2nd semester Dept. of Computer Science and Engineering Jashore University of Science and Technology.

Remarks:

Date of Submission: 13-01-2024

Index

Chapter 1: Introduction

1.1 Background

The backdrop against which the "Popcorn" Online Movie Ticket Booking System emerges is one of unprecedented transformation within the entertainment industry. Over the years, the way audiences consume media has evolved, with a significant shift toward digital platforms and online experiences. The background of "Popcorn" is steeped in the understanding that the contemporary moviegoer expects more than just a ticketing service – they crave an immersive, user-centric journey.

Digitalization has become synonymous with convenience, and "Popcorn" positions itself as a response to this burgeoning demand for streamlined, hassle-free movie experiences. The background narrative recognizes the increasing reliance on technology for entertainment choices and strives to bridge the gap between the cinematic world and the digital era.

In this era of on-demand content and instant gratification, "Popcorn" is not merely a ticketing system but a comprehensive ecosystem designed to meet the expectations of a tech-savvy audience. The background narrative underscores the platform's commitment to not only meet but exceed user expectations, creating a digital cinema experience that seamlessly integrates with the lifestyle and preferences of the modern movie enthusiast.

1.2 System Request

The impetus behind the "Popcorn" system is a response to a compelling system request that encapsulates the collective vision of stakeholders. It originates from a strategic recognition of the evolving landscape, where consumers seek instant, personalized, and secure services. The system request envisions a platform that goes beyond mere ticketing, aiming to create an immersive and interconnected experience for movie enthusiasts, hall managers, and administrators alike.

At its core, the system request is a call to redefine the cinematic journey. It envisions a platform that seamlessly integrates digital technologies, providing users with a personalized and intuitive interface. Simultaneously, it empowers administrators and hall managers with the tools necessary for efficient management, oversight, and adaptation to the dynamic expectations of the industry.

This strategic request aligns with the broader narrative of digital transformation in the entertainment sector, positioning "Popcorn" as a forward-thinking solution poised to meet the demands of a tech-savvy audience.

1.3 Methodology

The "Popcorn" system is shaped by a meticulous plan and methodology designed to navigate the complexities of development. Rooted in the agile philosophy, the project unfolds iteratively, allowing for continuous refinement. The chosen technology stack reflects a strategic blend of PHP for robust server-side processing, Laravel for streamlined development, JavaScript for dynamic client-side interactions, and Tailwind CSS for an aesthetic and responsive user interface. This approach ensures a harmonious integration of coding, testing, and deployment, with a focus on adaptability to evolving user needs.

Development Phases:

- **Requirements Gathering:** A comprehensive exploration of user needs, industry trends, and stakeholder expectations to shape the project scope.
- **Design:** The system's architecture is visualized through user interface mockups, system architecture diagrams, and a detailed database plan, ensuring a cohesive and intuitive design.
- **Implementation:** Coding commences based on the outlined requirements, employing PHP, Laravel, JavaScript, and Tailwind CSS. Code complexity is managed with an eye on maintainability.
- **Testing:** Rigorous testing protocols, including black box, white box, performance, and user acceptance testing, are implemented to ensure the reliability and functionality of the system.
- **Deployment:** The system transitions seamlessly from the development environment to a live production server, encompassing domain registration and configuration.
- **Monitoring and Maintenance:** Continuous vigilance, performance monitoring, and user feedback analysis ensure the sustained excellence of the system. Regular maintenance activities include updates and scalability assessments.

Chapter 2: Requirement Study

2.1 User Registration and Login

Functional Requirements:

- Users create accounts with unique usernames and secure passwords.
- Seamless verification and authentication of user credentials during login.

Non-Functional Requirements:

- Robust encryption and secure storage of user passwords.
- System response time for user authentication optimized to less than 1 second.

2.2 Movie Browsing and Selection

Functional Requirements:

- Intuitive interface for users to browse movies with details like title, genre, release date.
- Movie selection seamlessly initiates the ticket purchasing process.

Non-Functional Requirements:

- System designed to support a minimum of 500 movies in the database.
- Response time for loading movie details optimized to less than 2 seconds.

2.3 Ticket Purchase

Functional Requirements:

- Users can select tickets and make secure online transactions.
- System generates digital tickets upon successful purchase.

Non-Functional Requirements:

- Payment transactions encrypted using SSL.
- System designed to support a minimum of 1000 concurrent transactions.

2.4 Hall Management Tools

Functional Requirements:

- Hall managers update movie details, set pricing, and manage operational aspects.
- Admins approve or reject hall manager requests promptly.

Non-Functional Requirements:

- Hall manager requests processed within 24 hours.
- Hall manager tools feature an intuitive interface for efficient management.

2.5 User Profile Management

Functional Requirements:

- Users update personal information, view transaction history, and submit reviews.
- Admins have access to comprehensive user profiles for oversight.

Non-Functional Requirements:

- User profile updates reflect in real-time.
- Admin access to user profiles fortified with two-factor authentication.

2.6 Review and Rating

Functional Requirements:

- Users can submit reviews and ratings for movies and cinema halls.
- The system aggregates and displays average ratings for movies and halls.

Non-Functional Requirements:

- Reviews and ratings are stored securely, ensuring the integrity of user feedback.
- The system calculates and updates average ratings in real-time.

2.7 Notification System

Functional Requirements:

- Users receive notifications for ticket confirmation, movie releases, and special promotions.
- Admins can send broadcast notifications for system updates or announcements.

Non-Functional Requirements:

- Notification delivery is prompt and reliable.
- The system logs notification history for users and administrators.

Chapter 3: Design Diagrams

3.1 System Architecture Diagram

The system architecture diagram provides an overview of the components and their interactions within the "Popcorn" Online Movie Ticket Booking system. It outlines the flow of data between the user interface, server, database, and external services such as the Stripe payment gateway.

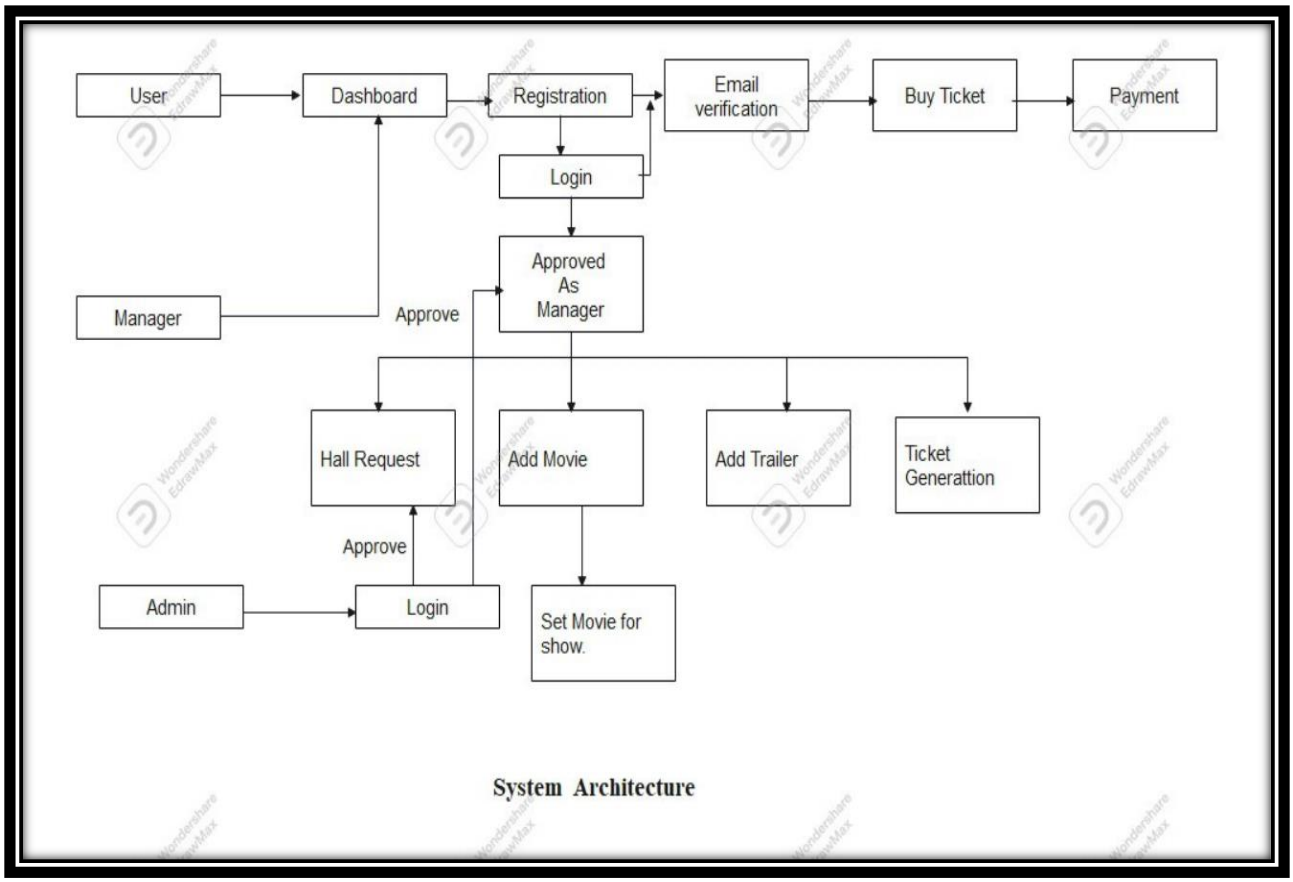

Diagram-3.1: System Architecture

3.2 Entity-Relationship (ER) Diagram

The Entity-Relationship (ER) Diagram for the "Popcorn" Online Movie Ticket Booking System provides a visual representation of the relationships among key entities within the system. It illustrates how different entities are interconnected and how data flows between them.

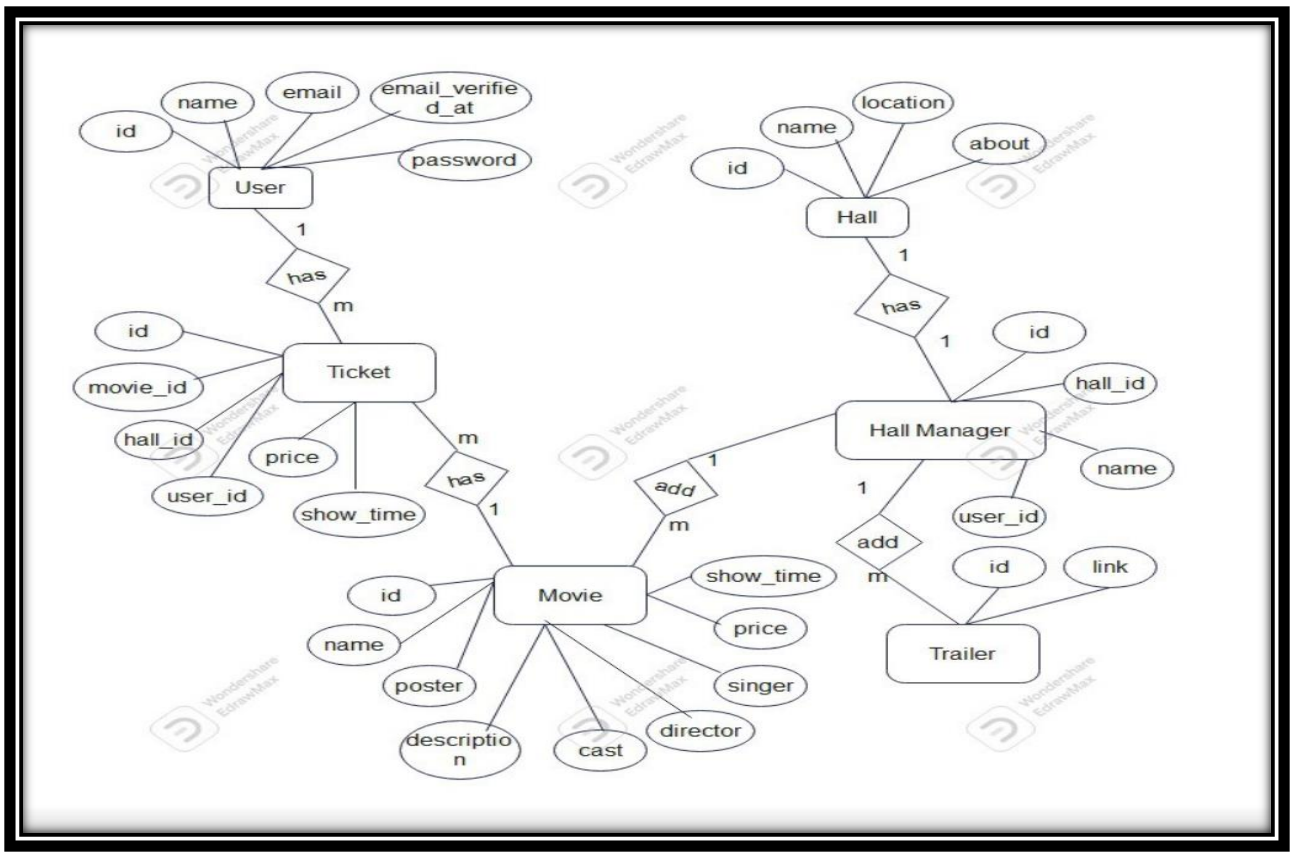

Diagram-3.2: Entity-Relationship (ER) Diagram

Entities:

User:

- Represents individuals registered in the system, storing user-specific information such as usernames, emails, and passwords.

Movie:

- Contains details about the movies available in the system, including titles, genres, and release dates.

Transaction:

- Records information about each user's transactions, including ticket purchases and payment details.

Hall:

- Stores information about cinema halls, including hall managers and specific operational details.

Review:

- Captures user reviews for different halls, associating feedback with specific transactions.

Relationships:

User and Transaction:

- A one-to-many relationship where each user can have multiple transactions, but each transaction is associated with a single user.

Movie and Transaction:

- A one-to-many relationship indicating that a movie can be associated with multiple transactions, but each transaction corresponds to a specific movie.

Hall and Transaction:

- A one-to-many relationship showcasing that a hall can have multiple transactions, but each transaction is linked to a specific hall.

Review and Transaction:

- A one-to-one relationship representing that each review is associated with a specific transaction.

Attributes:

User Attributes:

- UserID (Primary Key), Username, Email, Password

Movie Attributes:

- MovieID (Primary Key), Title, Genre, Release Date

Transaction Attributes:

- TransactionID (Primary Key), Date, Ticket Quantity, Total Amount

Hall Attributes:

- HallID (Primary Key), ManagerID (Foreign Key), Operational Details

Review Attributes:

- ReviewID (Primary Key), Rating, Comments, TransactionID (Foreign Key)

3.3 Schema Diagram

The Schema Diagram for the "Popcorn" Online Movie Ticket Booking System provides a visual representation of the structure and relationships within the database. It illustrates how different entities are organized and interconnected to support the seamless functioning of the system.

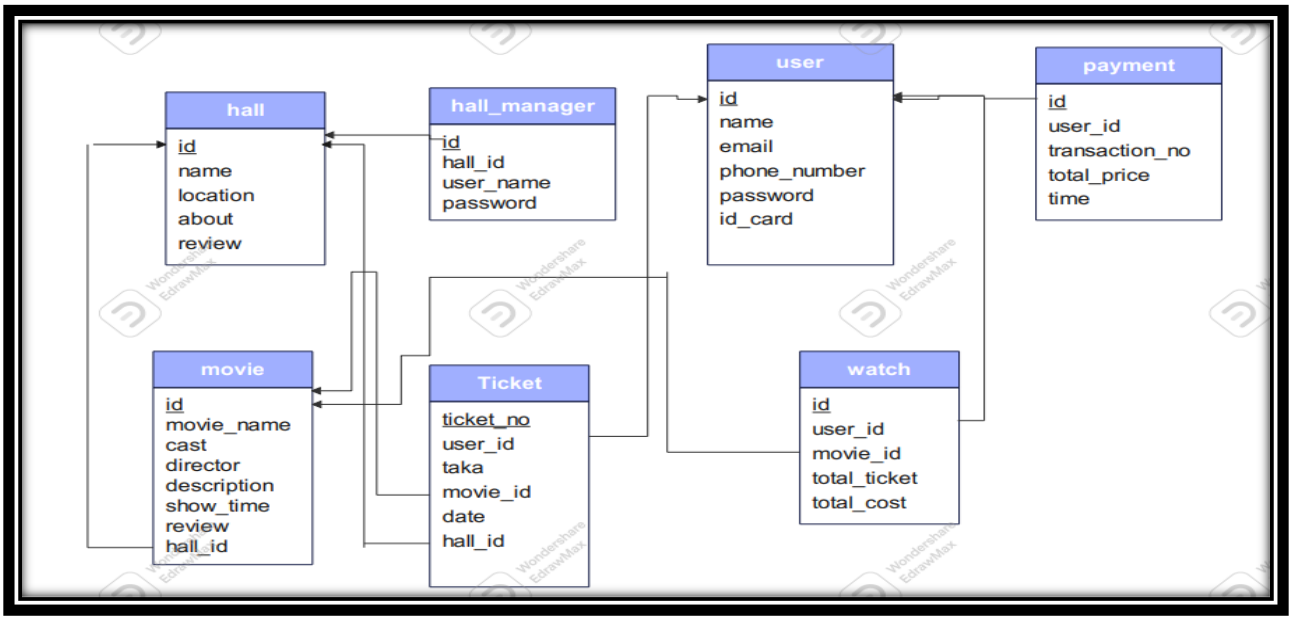

Diagram-3.3: Schema Diagram

Entities:

User:

- Represents individuals who register and use the system, storing user-specific information such as usernames, emails, and passwords.

Movie:

- Contains details about the movies available in the system, including titles, genres, and release dates.

Transaction:

- Records information about each user's transactions, including ticket purchases and payment details.

Hall:

- Stores information about cinema halls, including hall managers and specific operational details.

Review:

- Captures user reviews for different halls, associating feedback with specific transactions.

Relationships:

User and Transaction:

- A one-to-many relationship where each user can have multiple transactions, but each transaction is associated with a single user.

Movie and Transaction:

- A one-to-many relationship indicating that a movie can be associated with multiple transactions, but each transaction corresponds to a specific movie.

Hall and Transaction:

- A one-to-many relationship showcasing that a hall can have multiple transactions, but each transaction is linked to a specific hall.

Review and Transaction:

- A one-to-one relationship representing that each review is associated with a specific transaction.

Attributes:

User Attributes:

- Username, Email, Password

Movie Attributes:

- Title, Genre, Release Date

Transaction Attributes:

- Transaction ID, Date, Ticket Quantity, Total Amount

Hall Attributes:

- Hall ID, Manager ID, Operational Details

Review Attributes:

- Review ID, Rating, Comments

3.4 Use Case Diagram

A Use Case Diagram provides a visual representation of the functional requirements of the system and the interactions between its various actors (users or external systems) and use cases (system functionalities). In the context of the "Popcorn" Online Movie Ticket Booking System, the Use Case Diagram outlines key interactions and functionalities.

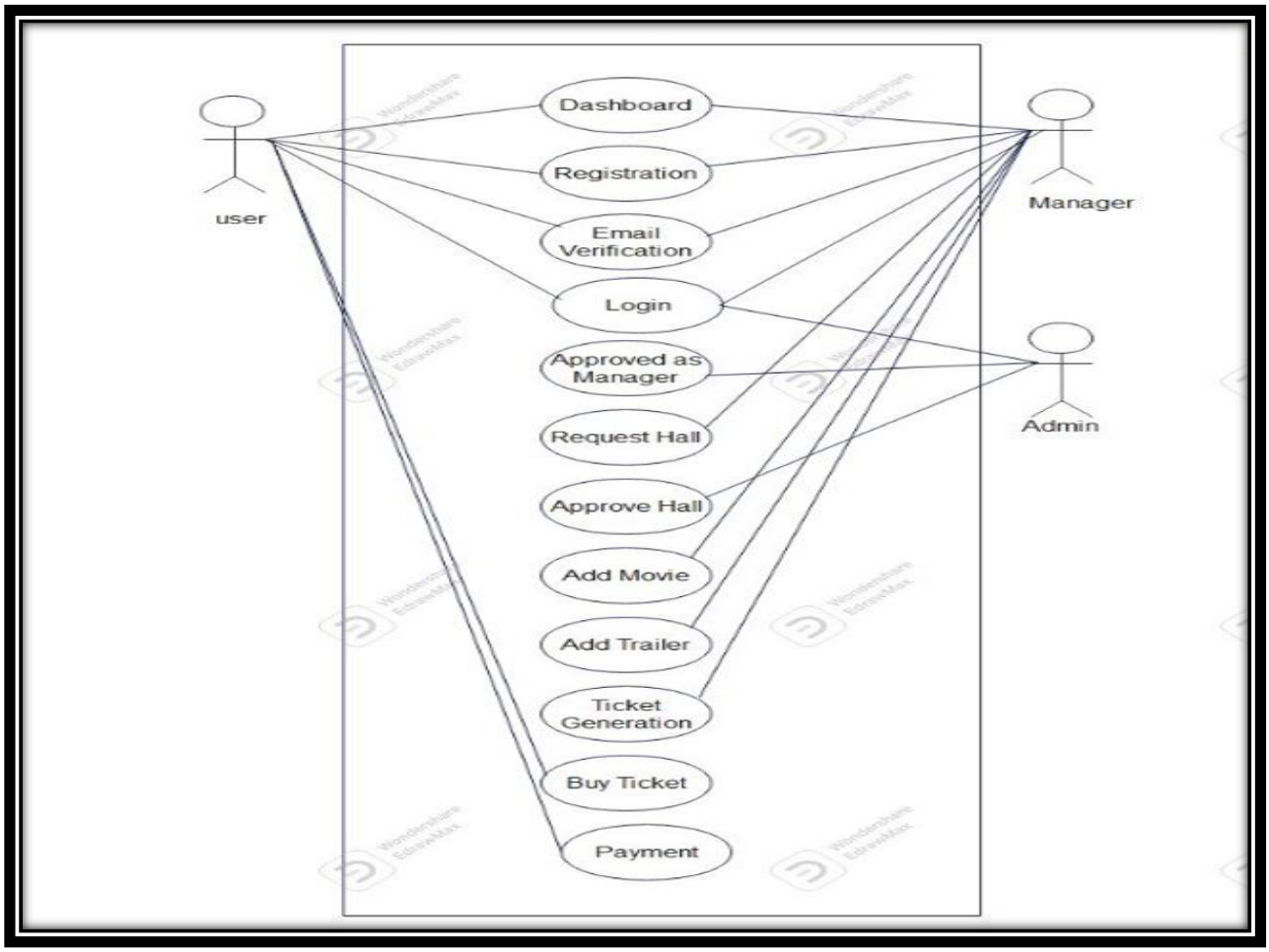

Diagram-3.4: Use Case Diagram

Actors:

User:

- Represents individuals accessing the system to browse movies, purchase tickets, and manage their profiles.

Hall Manager:

- Signifies the role responsible for overseeing and managing a specific cinema hall, including updating movie details and responding to user reviews.

Admin:

- Represents the system administrator who has comprehensive control over the entire system, managing halls and overseeing operations.

Use Cases:

Browse Movies:

- Allows users to view a list of available movies, explore details, and check showtimes.

Purchase Tickets:

- Enables users to select a movie, choose the number of tickets, and complete the transaction securely.

Manage User Profile:

- Permits users to update their personal information, view transaction history, and submit reviews.

Manage Hall Information:

- Enables hall managers to update movie details, set pricing, and manage hall-specific information.

Admin Dashboard:

- Provides administrators with a centralized view of system statistics and operations.

Approve Manager Requests:

- Allows the admin to approve or reject requests from individuals seeking hall manager access.

Relationships:

User to Use Cases:

- Users interact with the system through use cases such as Browse Movies, Purchase Tickets, and Manage User Profile.

Hall Manager to Use Cases:

- Hall managers engage with functionalities like Manage Hall Information to ensure accurate movie details and pricing.

Admin to Use Cases:

- The admin oversees system operations through the Admin Dashboard and manages hall-related requests.

3.5 Activity Diagram

The Activity Diagram for the "Popcorn" Online Movie Ticket Booking System provides a visual representation of the sequential flow of activities within the system. It illustrates the dynamic aspects of the processes and interactions among various components during a ticket purchase.

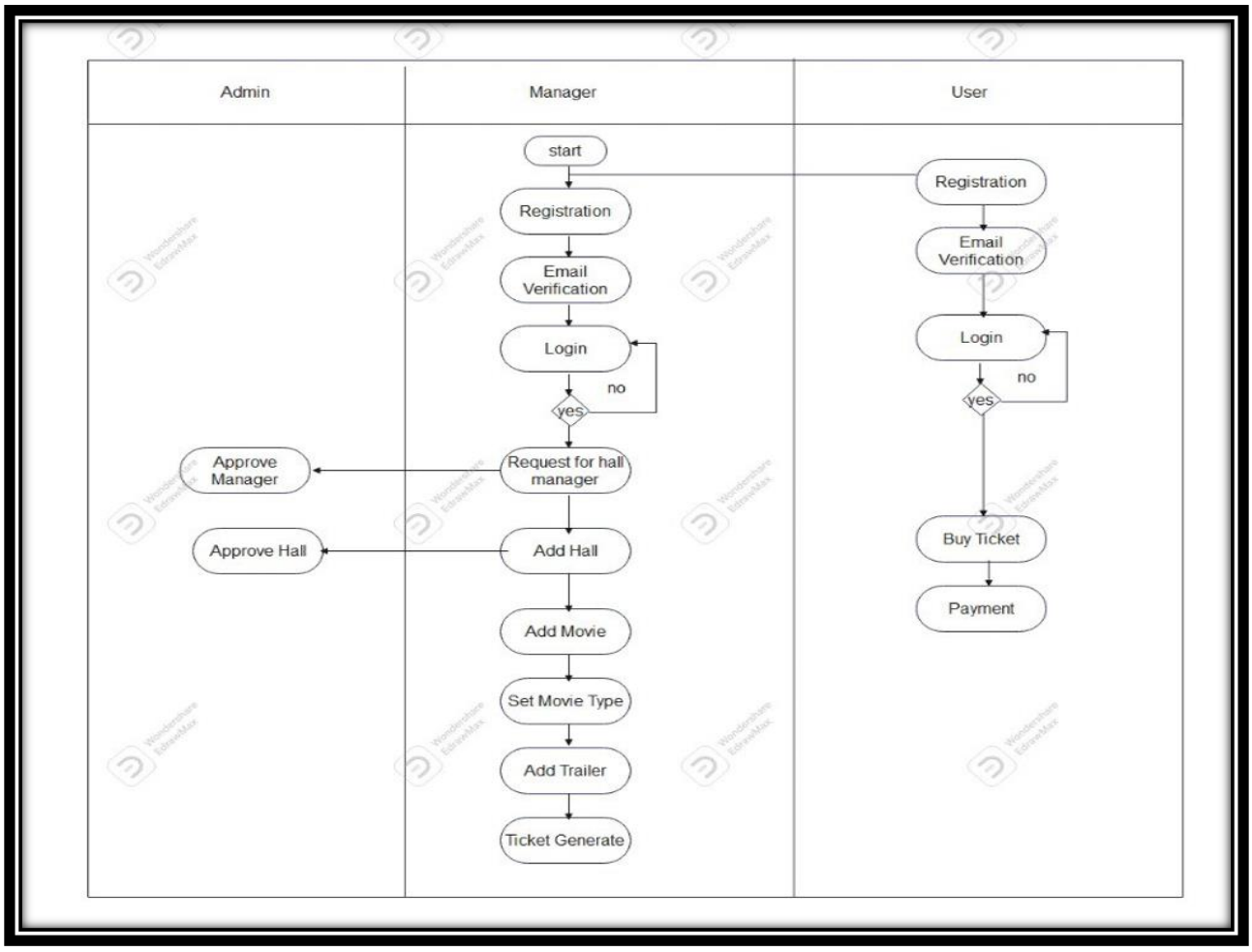

Diagram-3.5: Activity Diagram

Components:

User:

- Represents individuals interacting with the system, from selecting a movie to completing the ticket purchase.

Ticket Selling System:

- The central system responsible for handling user interactions, processing transactions, and updating relevant data.

Payment Gateway:

- External component facilitating secure payment processing.

Activities:

User Selects Movie:

- The process begins as the user selects a movie of interest.

Initiate Ticket Purchase:

- The user initiates the ticket purchase process, triggering the system to handle the transaction.

Check Ticket Availability:

- The system checks the inventory to ensure the selected number of tickets is available for the chosen movie.

Process Payment:

- The user's payment is securely processed through the payment gateway.

Generate Ticket:

- Upon successful payment, the system generates a digital ticket for the user.

Update Inventory:

- The system updates the inventory to reflect the reduced availability of tickets.

Send Confirmation:

- A confirmation, along with the digital ticket, is sent to the user.

Chapter 4: Implementation Details

4.1 User Authentication

4.1.1 Description

The user authentication process ensures that registered users can securely log in to the "Popcorn" system.

4.1.2 Leveling

Level 1:

- Validate input parameters.
- Query the database to check user credentials.

Level 2:

- Check if the user exists and credentials are valid.
- Return authentication status.

4.2 Ticket Purchase

4.2.1 Description

The ticket purchase process allows users to select a movie, choose the quantity of tickets, and securely complete the transaction.

4.2.2 Leveling

Level 1:

- Validate input parameters.
- Calculate the total amount based on ticket price and quantity.

Level 2:

- Create a transaction record in the database.
- Process payment using the Stripe gateway.
- Return transaction details.

4.3 Code Complexity

The code complexity of the "Popcorn" system is moderate. It involves CRUD operations, user authentication, and payment processing. The use of Laravel framework simplifies many aspects of development, reducing overall complexity.

4.4 Site Interfaces

4.4.1 Home Screen

Figure 4.4.1: The home screen displays a list of available movies and provides a user-friendly interface for navigation.

4.4.2 User Profile

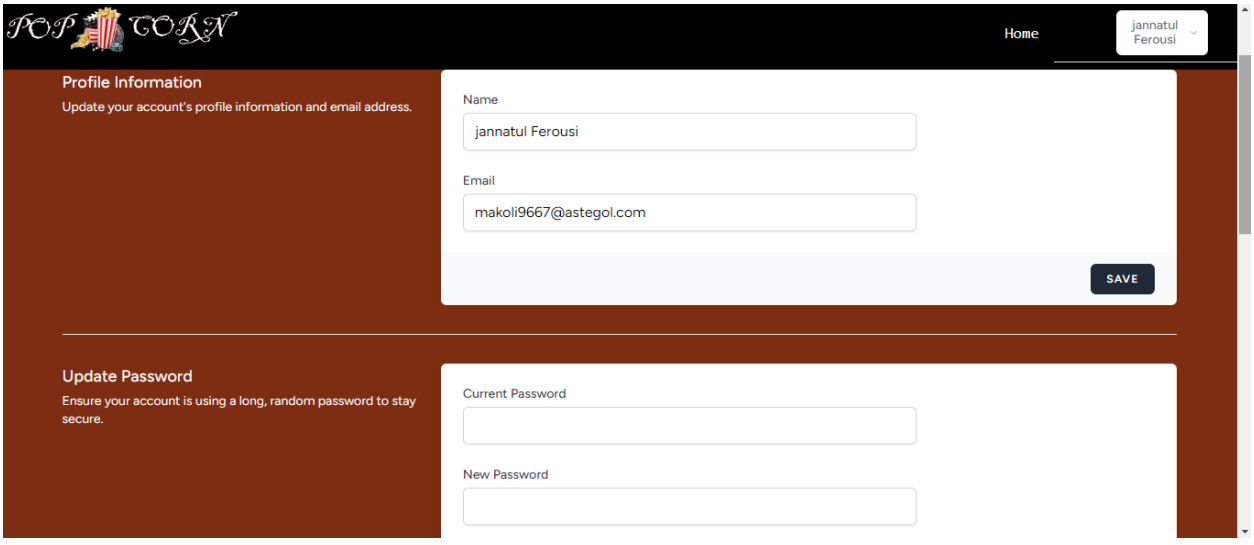

Figure 4.4.2: The user profile page allows users to manage their personal information, view transaction history, and submit reviews.

4.4.3 Admin Dashboard

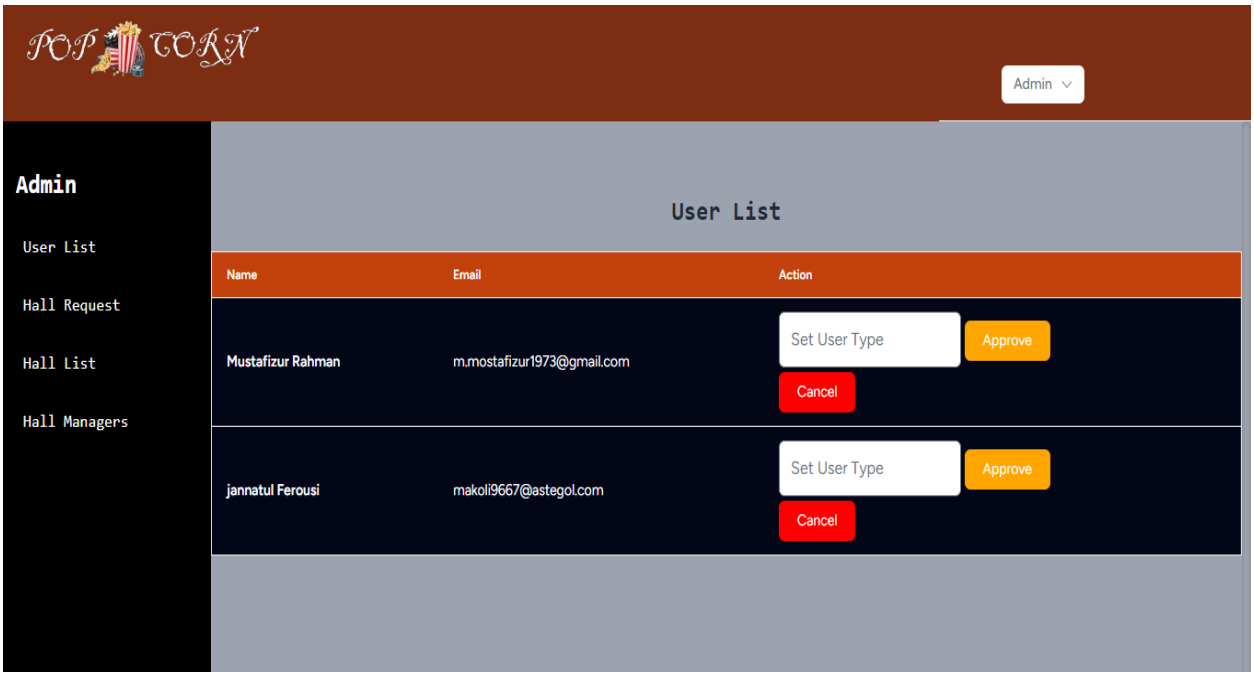

Figure 4.4.3: The system architecture illustrates the flow of data between the user interface, server, database, and external services.

Chapter 5: Testing

5.1 Black Box Testing

5.1.1 User Authentication

Test Case 1: Valid Login

- **Input**: Valid username and password.
- **Expected Output**: Successful login.
- **Result**: Pass

Test Case 2: Invalid Login

- **Input**: Invalid username or password.
- **Expected Output**: Authentication failure.
- **Result**: Pass

5.1.2 Ticket Purchase

Test Case 3: Successful Ticket Purchase

- **Input**: Valid movie selection, quantity, and payment.
- **Expected Output**: Successful transaction.
- **Result**: Pass

5.2 White Box Testing

5.2.1 User Authentication

Test Case 5: SQL Injection

- **Input**: Malicious input attempting SQL injection.
- **Expected Output**: Prevent SQL injection and return authentication failure.
- **Result**: Pass

5.2.2 Ticket Purchase

Test Case 6: Transaction Calculation

- **Input**: Valid movie selection and quantity.
- **Expected Output**: Correct calculation of the total amount.
- **Result**: Pass

Chapter 6: User Manual

Welcome to the "Popcorn" Online Movie Ticket Booking System! This user manual provides comprehensive guidance on using the system efficiently. Whether you are a user looking to purchase movie tickets, a hall manager managing your hall, or an admin overseeing the entire system, this manual will assist you in navigating through the features seamlessly.

6.1 User Section

6.1.1 Registering an Account

To access the "Popcorn" system, you need to register an account. Follow these steps:

- Click on the "Sign Up" or "Register" button on the homepage.
- Fill in the required information, including your username, email, and password.
- Click "Submit" to complete the registration.

6.1.2 Logging In

If you already have an account, follow these steps to log in:

- Click on the "Log In" button on the homepage.
- Enter your username and password.
- Click "Log In" to access your account.

6.1.3 Browsing Movies

- Navigate to the homepage.
- Browse through the list of available movies.
- Click on a movie to view details, including trailers and pricing.

6.1.4 Purchasing Tickets

- Choose the number of tickets and click "Purchase."
- Follow the on-screen instructions to complete the transaction.

6.1.5 Managing User Profile

- Go to the user profile section.
- Update your personal information and view transaction history.

6.1.6 Submitting Reviews

- After purchasing tickets, go to the user profile section.
- Find the "Submit Review" option.
- Provide a rating and optional comments for the hall.

6.2 Hall Manager Section

6.2.1 Requesting Manager Access

- Log in to your user account.
- Navigate to the admin-managed halls section.
- Click on "Request Manager Access" for the desired hall.

6.2.2 Managing Hall Information

- Log in to the system as a hall manager.
- Access the hall management dashboard.
- Update movie details, pricing, and hall-specific information.

6.2.3 Overseeing Hall Operations

- Monitor ticket sales and transaction history.
- Respond to user reviews and feedback.
-

6.3 Admin Section

6.3.1 Admin Dashboard

- Access the admin dashboard to view system-wide statistics.
- Manage halls, approve/reject manager requests, and oversee operations.

6.3.2 Managing Halls

- View a list of all halls in the system.
- Edit hall information, including managers and operational details.

6.3.3 Approving Hall Manager Requests

- Access the admin dashboard.
- Review pending hall manager requests.
- Approve or reject requests based on the provided information.

Chapter 7: Deployment

7.1 Deployment

7.1.1 Overview

The deployment process for the "Popcorn" Online Movie Ticket Booking System involves transferring the developed application from the development environment to a live production server. This documentation outlines the steps and considerations for a successful deployment.

7.1.2 Deployment Plan

Step 1: Pre-Deployment Preparation

- **Backup**: Ensure a backup of the entire application and database is created for recovery purposes.
- **Environment Check**: Verify that the production server meets the system requirements for hosting the application.

Step 2: Code Repository

- **Clone Repository**: Clone the latest version of the application from the version control system (e.g., Git).
- **Update Dependencies**: Run composer install to ensure all required dependencies are up to date.

Step 3: Configuration

- **Environment Variables**: Configure environment variables for the production environment, including database connection details, API keys, and other sensitive information.
- **File Permissions**: Set appropriate file permissions for security.

Step 4: Database Migration

- **Run Migrations**: Execute database migrations to create necessary tables and structures.
- **Seed Database (Optional)**: Populate the database with initial data if needed.

Step 5: Web Server Configuration

- **Virtual Host**: Set up a virtual host on the web server (e.g., Apache or Nginx) to point to the application's public directory.
- **SSL Configuration**: If applicable, configure SSL for secure communication.

Step 6: Security Measures

- **Firewall Settings**: Configure server firewall rules to restrict access.
- **Update Software**: Ensure all server software, including the operating system and web server, is up to date with the latest security patches.

Step 7: Testing

- **Health Check**: Perform a health check to ensure the application is running smoothly.
- **Browser Testing**: Conduct cross-browser testing to verify compatibility.

Step 8: Monitoring

- **Logging**: Set up logging mechanisms to monitor errors and application activities.
- **Performance Monitoring**: Implement tools for performance monitoring and issue tracking.

Step 9: Final Checks

- **Backup (Again)**: Take a final backup of the deployed application and database.
- **Domain Configuration**: Confirm that the domain is correctly configured to point to the application.

Step 10: Deployment

- **Deploy Application**: Transfer the application files to the production server.
- **Restart Services**: Restart the web server and any relevant services to apply changes.

7.1.3 Post-Deployment

- **Monitoring**: Continuously monitor the application for performance, security, and potential issues.
- **Scalability**: Plan for future scalability and consider load balancing if needed.

7.2 Domain

7.2.1 Domain Registration

- **Choose a Domain Name**: Select a relevant and memorable domain name.
- **Register Domain**: Use a domain registrar service to register the chosen domain.

7.2.2 Domain Configuration

- **DNS Configuration**: Set up Domain Name System (DNS) records to point the domain to the IP address of the production server.
- **SSL Certificate**: If using HTTPS, configure and install an SSL certificate for the domain.

7.3 Conclusion

The deployment of the "Popcorn" Online Movie Ticket Booking System involves careful planning, configuration, and testing to ensure a smooth transition from the development environment to the live production server. Continuous monitoring and updates are essential for maintaining the system's performance and security.