Software Requirements Specification Template

Software Engineering

The following annotated template shall be used to complete the Software Requirements Specification (SRS) assignment.

Template Usage:
Text contained within angle brackets (‘<’, ‘>’) shall be replaced by your project-specific information and/or details. For example, <Project Name> will be replaced with either ‘Smart Home’ or ‘Sensor Network’.

Italicized text is included to briefly annotate the purpose of each section within this template. This text should not appear in the final version of your submitted SRS.

This cover page is not a part of the final template and should be removed before your SRS is submitted.

Theatre Ticketing System

Software Requirements Specification

Version 1.0

01/31/2023

Group 7
Andre Lasola
Jacen Salvador
Maksym Yelisyeyev

Prepared for
CS 250- Introduction to Software Systems
Instructor: Gus Hanna, Ph.D.
Fall 2023

Revision History

Date	Description	Author	Comments
<2/14>	<Version 1>	<Group 7>	<First Revision>
<2/28>	<Version 2>	<Group 7>	<Second Revision>
<3/13>	<Version 3>	<Group 7>	<Third Revision>

Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

Signature	Printed Name	Title	Date
	Dr. Gus Hanna	Instructor, CS 250	
	Dr. Gus Hanna	Instructor, CS 250	
	Dr. Gus Hanna	Instructor, CS 250	

Table of Contents

1. Introduction

This document will define and describe the requirements of the project, as well as the functionality and limitations of the software used.

1.1 Purpose

The purpose of this document is to create a ticketing system and collect ideas for the system with specifications to create a secure, stable, and user-friendly interface. From this, we will showcase and add different concepts and ideas that best fit our system as well as remove features as needed. Our document showcases the goals, features, and upcoming plans for the software. The intended audience is our developers to showcase requirements and assist the team in the software development process.

1.2 Scope

The software product is a theater ticketing system designed to facilitate the purchase of tickets for movies in San Diego chains. It will include online ticket purchasing, seat selection, and account management. The software products will allow users to search for movies, view showtimes, select seats, and purchase tickets online. You will be able to create an account for other options such as refunding tickets and exchanging them. The process will not handle physical ticket printing as well as any in-theater services. The application is intended to make the ticketing process easier and create a smooth ticket-purchasing process. The product aims to provide a user-friendly interface for purchasing tickets, ensure the security and integrity of ticket purchases and user information, support a large number of concurrent users to handle peak ticketing periods, integrate with external service for payment processing, and movie reviews, provide customer accounts to enhance user engagement and improve reporting theater capabilities.

1.3 Definitions, Acronyms, and Abbreviations

SRS (Software Requirements Specification), DBMS (Database Management System), API (Application Programming Interface), HTTP (Hypertext Transfer Protocol), HTTPS (Hypertext Transfer Protocol Secure), SMTP (Simple Mail Transfer Protocol), TCP/IP (Transmission Control Protocol/Internet Protocol), DNS (Domain Name System), CRM (Customer Relationship Management), APIs (Application Programming Interface), URL (Uniform Resource Locator), SSL (Secure Sockets Layer), PCI DDS (Payment Card Industry Data Security Standard), NFTs (Non-Fungible Tokens)

1.4 References

IEEE Guide to Software Requirements Specifications, IEEE Std 830-1998, IEEE Computer Society, October 20, 1998.

1.5 Overview

The Software Requirements Specification document outlines the requirements for the theater ticketing system. It contains detailed information about the functionality, constraints, and dependencies of the system.

The SRS is organized into an introduction which describes the overview and purpose of the entire document. It is then divided into a general description of the document which describes the general factors that affect the product including its perspective, functions, user characteristics, constraints, assumptions, and dependencies. It dives into the specific requirements which describe the specific requirements of the system, external interface requirements, functional requirements, use cases, classes and objects, and more. It then talks about the analysis models which go into detail about the diagrams used to create the system. It talks about the updating process and then talks about the marketing materials that support the SRS.

2. General Description

2.1 Product Perspective

The theater ticketing system is a standalone product that allows movie ticketing services for movies. It is designed to operate independently of other projects but can take external payment and reviews from other sites. This is a standalone system but it is part of a larger ecosystem of other ticketing systems in the industry. Since it is a wide ecosystem, it is important to create an industry-standard website and create a seamless experience for the users. The system is designed to create a flexible and scalable solution that can adapt to the needs of the theater chain and broaden the entertainment industry. It is made to enhance the experience for customers.

2.2 Product Functions

2.2.1 Sale of Tickets

The system shall allow users to purchase a maximum of 20 tickets.
The system shall allow users to purchase tickets 2 weeks before the premiere date.
The system shall allow users to purchase tickets 10 minutes after showtime.
The system shall allow users to choose and reserve specific seats with their tickets.
The system shall allow users to purchase tickets through the online website, or in person at a digital kiosk at the theater.

2.2.2 Pricing of Tickets

The system shall have a consistency of prices across the system.
The system shall offer discounts available for veterans, students, and senior citizens.

2.2.3 Delivery of Tickets

The system shall provide the tickets through either email or in person at the theater's box office.

2.2.4 Payment Options

The system shall allow payment methods including credit card, PayPal, and Bitcoin.

2.2.5 Return/Exchange Policy
The system shall allow tickets to be returned or exchanged before showing if purchased with a user account.

2.2.4 Protection against Shopping Bots
The system shall allow users to purchase a maximum of 20 tickets.
The system must block bot attempts to buy large numbers of tickets to high-demand movies.

2.2.5 Prevention against Scalping
The system shall establish security measures to ensure tickets are unique and non-replicable.

2.2.6 Customer Feedback
The system shall include a customer feedback system.

2.2.7 User Accounts
The system shall allow users to register loyalty accounts with personal/payment information, purchase history, and loyalty points.

2.3 User Characteristics

2.3.1 Users are expected to be literate, with little to no experience with technology.
2.3.2 Users should know how to complete online transactions.

2.4 General Constraints

2.4.1 Deadline
The project has a deadline of 4 months which can impact the scope of the features that can be implemented.

2.4.2 System Updates
The system must implement updates as few as possible to reduce server downtime.

2.4.3 Performance Requirements
The system must support many users which can cause constraints on the system architecture, scalability, and performance optimization.

2.4.4 User interface compatibility
The system must be compatible and limited to web browsers.

2.4.5 Regulatory Compliance
The system must comply with the ticket sale, data protection, and online transaction laws to enforce maximum security.

2.4.6 System Cost of Production
The production of the system must cost a maximum of $500,000.

2.5 Assumptions and Dependencies

2.5.1 Availability of Internet Connection
Users should have stable access to the Internet.

2.5.2 Availability of Theaters
The system shall only be available within the San Diego Area.
The system shall accommodate either Deluxe Theaters or Regular Theaters.

2.5.3 Availability of Seats
Deluxe Theaters can only accommodate a maximum of 75 seatings per theater.
Regular Theaters can only accommodate a maximum of 150 seatings per theater.

2.5.4 Compatibility with Web Browsers
The system assumes compatibility with common web browsers such as Safari, Chrome, Firefox, and many other browsers.

2.5.5 Ticket Purchase Limits:
The system assumes the users will adhere to the maximum ticket purchase limit. (20 tickets per purchase or 2 weeks before the premier

2.5.6 Server Capabilities
The system assumes that the server will support at least 1000 users at the same time and up to 10 million concurrent users.
The system will implement a queueing system for high volumes of requests for single showings.
The system shall be available in the following languages: English, Spanish, and Swedish.

2.5.7 User Literacy and Experience
The system assumes that the users have experience with online transactions and are expected to know how to complete purchases through the system

2.5.8 Operating System and Hardware
The system assumes that the hardware and software support accessing web browsers and internet connection

2.5.9 External Dependencies
The system may depend on external services for payment processing, email delivery and other interfaces which are assumed to be functional and available.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The system shall be compatible with any web browser that can access the internet such as Safari, Chrome, Firefox, etc.

3.1.2 Hardware Interfaces

Since the website must run on an internet browser, all the hardware shall require to connect internet will be hardware interface for the system.

3.1.3 Software Interfaces

The system shall interact with a database to retrieve data related to movies.
The system shall have a payment gateway to ensure user data security.
The system shall have an email service to mail and deliver the tickets after purchase.
The system shall have a web server to process and handle requests.
The system shall have external APIs to gather reviews from movie review sites.

3.1.4 Communications Interfaces

The system will utilize various communication interfaces to facilitate the interaction between the system and external entities such as HTTP/HTTPS, email, APSs, TCP/IP, and DNS

3.2 Functional Requirements

3.2.1 <Retention of Information>

3.2.1.1 Introduction

The Theater Ticketing System must maintain information about purchased tickets, seating availability, the schedule, and location of desired showings.

3.2.1.2 Inputs

The system shall consider the following as inputs: opening the website, selecting a desired showing, logging onto the website, and purchasing a ticket.

3.2.1.3 Processing

The system shall process the information provided by the user via JavaScript code.

3.2.1.4 Outputs

The system shall display a desired showing, seats taken, and information about the purchased ticket.

3.2.1.5 Error Handling

If the user provides incomplete/incorrect data, then provide an error message and prompt the user to enter data following requirements.

3.2.1.6 Database Structure

The system shall utilize one database for all theaters partitioned into different theaters to manage ticket inventory and sales data

3.2.1.7 Database Interface

The system shall interface with the database to retrieve showtimes with available ticket information.

3.2.2 <Logging onto the Website>

3.2.2.1 Introduction

The Theater Ticketing System must allow the user to create an account and log into an existing account.

3.2.2.2 Inputs

The system shall consider the following as inputs: Email Address, Username, Password, and Security Information.

3.2.2.3 Processing

The system shall process the information provided by the user via JavaScript code.

3.2.2.4 Outputs

The system must prompt the user to create an account, i.e. enter email address, phone number, and full name.

3.2.2.5 Error Handling

If the user provides incomplete/incorrect data, then provide an error message and prompt the user to enter data in accordance with requirements.

3.2.2 <User Personal Office>

3.2.2.1 Introduction

The System must allow the user to access their personal office, once an account is created, where information about purchased tickets would be shown.

3.2.2.2 Inputs

The System shall consider the following as inputs: Username, Password, and Security Information.

3.2.2.3 Processing

The system shall process the information provided by the user via JavaScript code.

3.2.2.4 Outputs

The system shall display purchased tickets and personal information that the user has provided.

3.2.2.5 Error Handling

If the user provides incomplete/incorrect data, then provide an error message and prompt the user to enter data in accordance with requirements. If the problem persists, ask the user if they forgot their password.

3.3 Use Cases

3.3.1

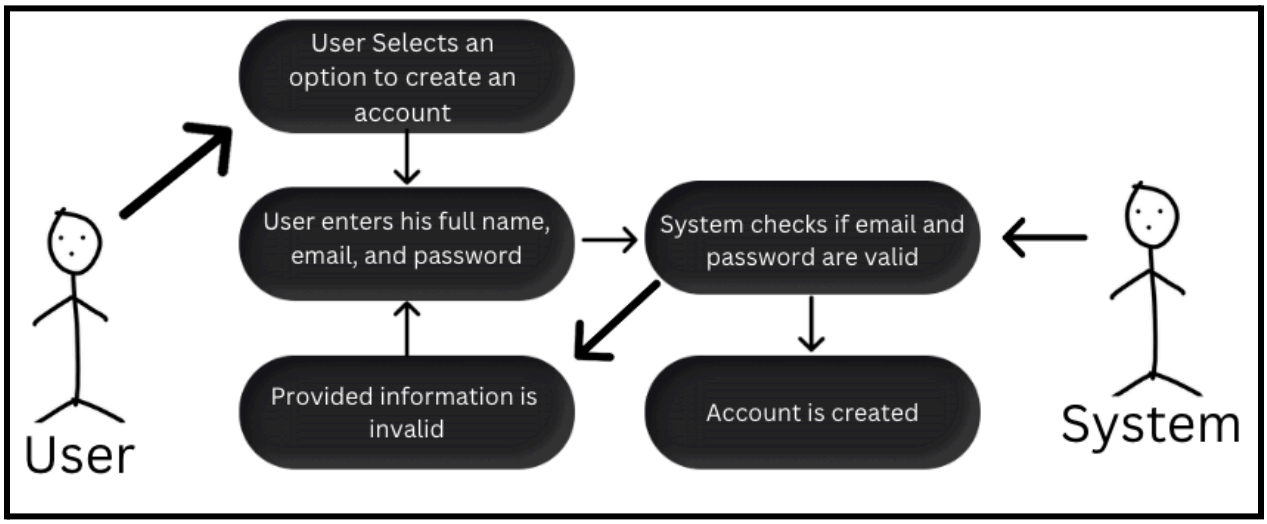
A. Name: Creating an Account
B. Brief Description: User Creates an account and chooses login details.
C. Actors: User, System
D. Basic flow:
    a. User selects an option to create a new account.
    b. User enters full name, email, and password.
    c. System confirms if email is valid and password complies with password requirements.
    d. System confirms account creation.
E. Alternate Flows:

a. User inputs an invalid email address.
b. User inputs an invalid password.

F. Pre-conditions:
   a. User has a connection to the internet.
   b. User is on the website.

G. Post-conditions
   a. User Account is created.
   b. User canceled account creation.

H. Special Requirements:
   a. System should have email address validation.
   b. System should have password requirements.



3.3.2 Purchasing a Ticket

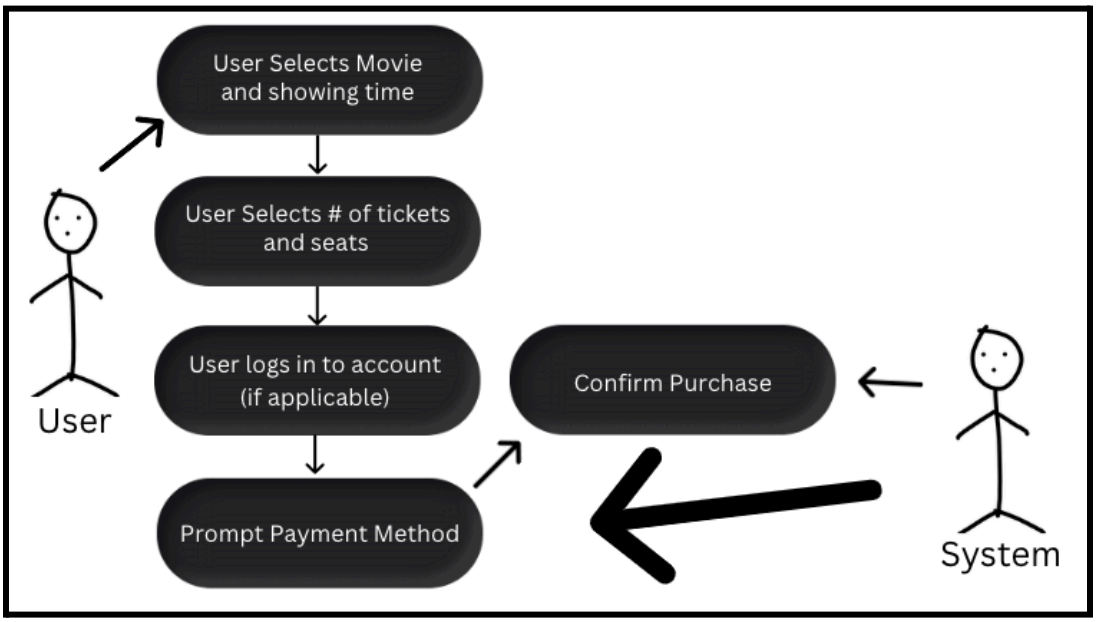
A. Name: Purchasing a Ticket
B. Brief description: User purchases a ticket and chooses seats, movie, and payment methods.
C. Actors: User, System
D. Basic Flow:
   a. User selects a movie.
   b. User selects a showing time.
   c. User selects number of tickets for the premiere.
   d. User selects desired seats.
   e. User logs in to account.
   f. User applies discounts.
   g. System prompts user for payment method.
   h. System confirms purchase.
   i. System provides purchase details.
E. Alternate Flows:
   a. User doesn't log in to the account.
   b. User chooses to log in to the account but uses wrong information.
   c. User doesn't apply discounts.

   d. User provides wrong payment information.

F. Pre-conditions:
   a. User has a connection to the internet.
   b. User is on the website.
   c. Possession of a payment method.
   d. There are available seats to the showing.
   e. User has sufficient funds to purchase ticket/s.

G. Post-conditions:
   a. User purchases a ticket.
   b. System rewards the user with loyalty points.

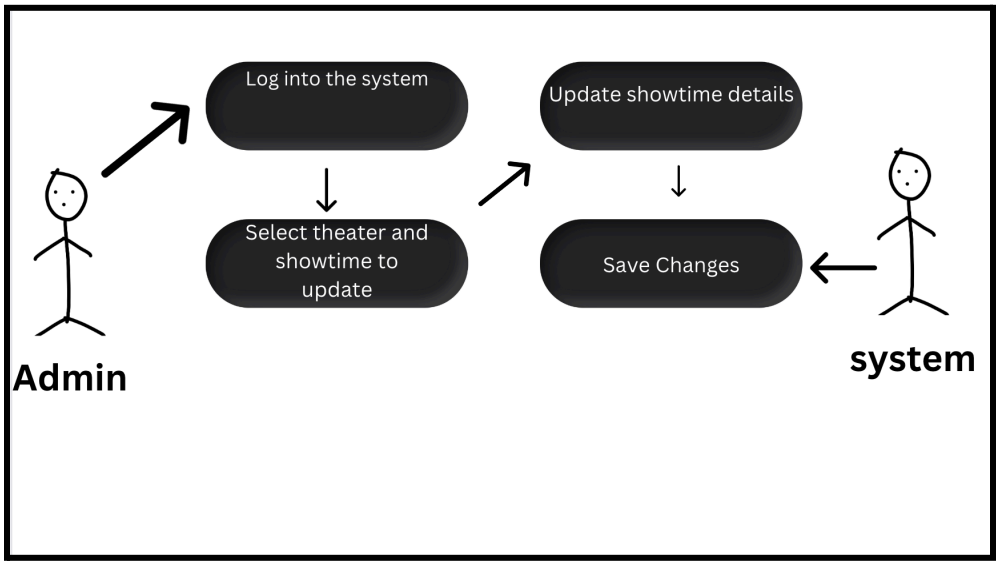
H. Special Requirements:
   a. System should have email address validation.
   b. System should have password requirements.
   c. System should have payment information validation.



3.3.3 Administration

A. Name: Administration Managing Showtimes
B. Brief Description: The administrator manages showtimes for a specific theatre by updating the showtime details in the system.
C. Actors: Administrator
D. Basic Flow:
   a. Administrator logs onto the system
   b. Administrator selects the specific show/theatre to be updated
   c. Administrator updates the showtime details such as the time and date
   d. Administrator saves the changes
E. Alternate Flows: None
F. Pre Conditions:
   a. Administrator has access to the system
   b. Showtime information for the theatre exists in the database

G. Post-conditions: Showtime details are updated in the database
H. Special Requirements: None



3.4 Classes / Objects

3.4.1 <General User>

3.4.1.1 Attributes
Average user, laptop, smartphone or PC.
3.4.1.2 Functions
General user should be able to navigate the website, browse for showings and purchase tickets.
General user cannot alter any information on the website, except for his personal information(refer to 3.2.2)
<Reference to functional requirements and/or use cases>

3.4.2 <Administrator>

3.4.2.1 Attributes
Administrator is a user utilizing a corporate account.
3.4.2.2 Functions
Administrators should be given permission to alter website information and access user information.

3.5 Non-Functional Requirements

3.5.1 Performance

The system shall process transactions quickly and accurately
The system shall support up to 10000 concurrent users
The system shall have a quick response time

3.5.2 Reliability

The system shall have an MTBF of at least 30 days

The system shall be able to recover from failure within 5 minutes
The system shall have a backup system in case of failure

3.5.3 Availability

The system shall be able to be used 24/7 except during maintenance
The system shall provide real-time updates for seat and showtimes

3.5.4 Security

The system shall encrypt all sensitive data and payment information
The system shall implement user authentication and mechanisms to ensure data privacy and security
The system shall keep daily logs system wide to ensure traceability of transactions

3.5.5 Maintainability

The system shall be modular and well documented to keep smooth maintenance
The system shall provide logging and monitoring capabilities to track system performance

3.5.6 Portability

The system shall be platform independent and compatible with different web browsers
The system shall be compatible with browsers on mobile devices.

3.6 Inverse Requirements

The system shall not allow user accounts to be logged in to multiple devices at the same time.

3.7 Design Constraints

Specify design constrains imposed by other standards, company policies, hardware limitation, etc. that will impact this software project.

3.8 Logical Database Requirements

Will a database be used? If so, what logical requirements exist for data formats, storage capabilities, data retention, data integrity, etc.

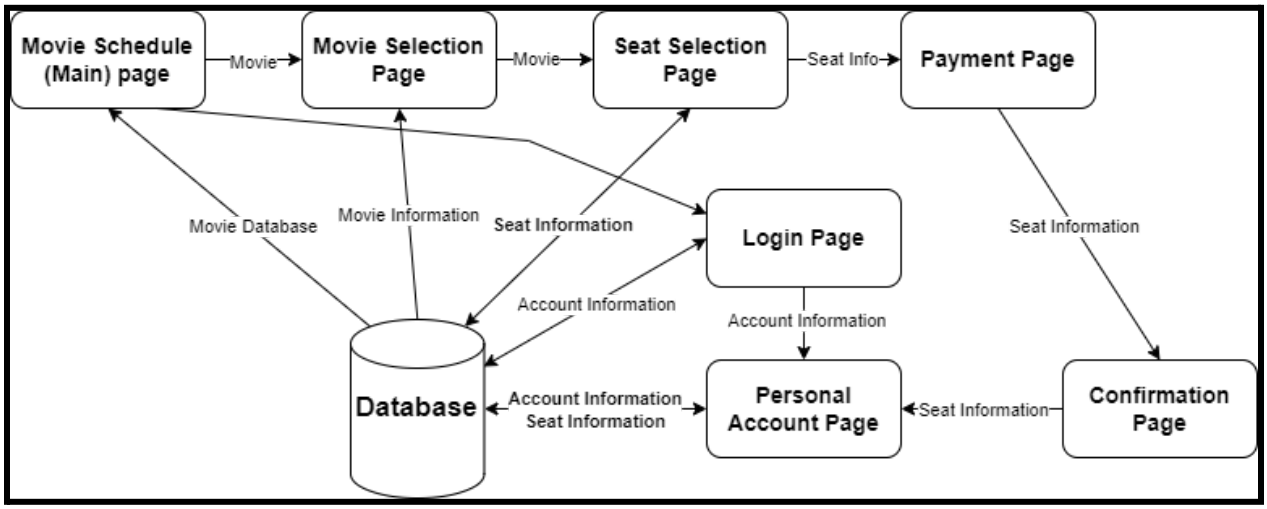
3.9 Other Requirements

Catchall section for any additional requirements.

4. Analysis Models

List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS's requirements.

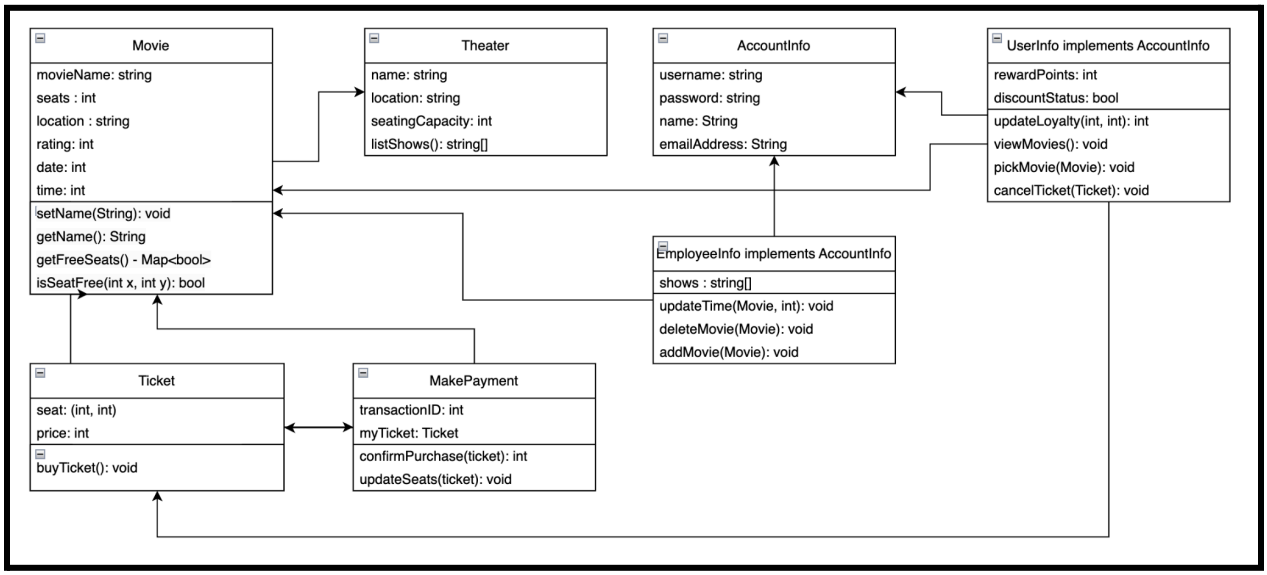
4.1 Architectural Diagram of All Major Components



4.2 Description of Architectural Diagram

- Database: This component is the center of all other major components, used to store information about movies, seats, and account information.
- Movie Schedule (Main) Page: This page is used as the main page whenever a user enters the website. Takes information from the database to display available movies to choose from.
- Login Page: This page is where the user enters their credentials to log into their personal account page. Takes information from the database about account information to validate/decline login attempts.
- Movie Selection Page: This page is used to select a movie. Takes information from movie schedule page
- Seat Selection Page: This page is used to select a seat. Takes information from the movie selection page for the movie and database for the seat information.
- Payment Page: This page is outputted whenever a user selects seats they are interested in. Sends the user to the Confirmation page on successful payment.
- Confirmation Page: This page is used to confirm payment made by the user and provides seat information to the user personal account page.
- Personal Account Page: This page is used whenever a user logs into their account. Provides seat information and account information to the database.

4.3 UML Class Diagram



4.4 Description of classes

- Movie: This allows you to see what movies are being screened, it includes the movies name, time of screening, the screen number, location and etc.
- Theatre: This represents the theater itself as it includes information about the movies, the layout and the seating arrangement.
- AccountInfo: This allows you to store user accounts and it includes details about the username, password, name and emails of the account owners.
- UserInfo: This allows you to represent information about the user and lets you look at the reward points, discount and watching habits.
- EmployeeInfo: This stores information about about who is working in the theatre and includes the ID, role and the contact information
- MakePayment: This shows the payment process for the movie and shows the ID of the transaction and the ticket being purchased.
- Ticket: This represents the ticket for the purchased movie and it talks about the movie information, seat number and the price of the ticket.

4.5 Description of attributes

- Movie: movieName (gives the name of the movie), date (shows the date of the movie), time (says what time the movie is at), screen (shows which screen the movie is at), rating (shows how people review the movie), location (shows where the movie is being screened), seats (shows the seats of the theatre)
- Theatre: movies (shows the movies that are being shown)
- AccountInfo: username (creates a name for a user), password (creates a way to access the username), name (allows you to save who you are), emailAddress (stores where you can get emailed for everything)

- UserInfo: rewardPoints (if you keep buying tickets, you will be able to gain points), discountStatus (depending on the amount of points, you can get a discount)
- EmployeeInfo:
- MakePayment: transactionId (gives you a receipt for the ticket), myTicket (shows you the ticket that you bought)

4.6 Description of operations

- Movie: setName(String): [this makes the name of the movie], getName(): [this gets the name of the movie], getFreeSeats() [this gets the list to see if there are any seats], isSeatFree(index) [this checks to see if the seat is taken]
- Theatre: updateMovie() [this changes the movie and allows you to update the details]
- AccountInfo:
- UserInfo: updateLoyalty(int, int) [see how often you watch movies and buy tickets], viewMovies() [you can see the movies that you seen], pickMovie(Movie) [you can choose the movie that you want to view that is on your list], cancelTicket(Ticket) [you can cancel your ticket if you don't want to watch the movie anymore]
- EmployeeInfo: updatetime(Movie, int) [you can change the time of the movie showing], deleteMovie(Movie) [you can remove a movie from the database], addMovie(Movie) [you can add a movie to the database]
- MakePayment: confirmPurchase(ticket) [you can confirm if you want to buy the ticket and go through with it], updateSeats(ticket) [the site will update the seat to show that it is taken]
- Ticket: buyTicket() [this allows you to initiate the make payment process]

5. Change Management Process

Identify and describe the process that will be used to update the SRS, as needed, when project scope or requirements change. Who can submit changes and by what means, and how will these changes be approved.

6. Test Cases

Refer to the attached document for the test case samples.

7. Data Management Strategy

This section will describe how the system will be managing its data.

7.1 Database Choices

There is one database that contains all the information (Movie Information, Seat information, and User Information). Having a singular database simplifies the process and makes it easier to find information on a single database rather than scattered throughout several databases.

7.2 Data Partitioning

The data will be split up into three categories: Movies, Seat Information, and Account Information.

- Movies: This category will have information about which movies are currently available, the times they are available, and which venue they are available at. Each will have a distinct code for the seating category.
- Seating: This category will have information about which seats are available at which venues. Each seating chart will have a distinct code depending on the venue.
- Account Information: This category will have information about the user such as first and last name, login, password, email address, and age.

7.3 Data Security

Users will be required to authenticate themselves before accessing the system. This will be achieved through various methods such as passwords, two-factor authentication (2FA), and single sign-on (SSO).

Once authenticated, users will only have access to the data and functionality that they are authorized to use.

7.4 Alternatives Considered

One alternative to the single database that is implemented could be replaced with multiple databases instead, distributing information throughout the separat e databases instead of having them all on one database. Through this, we could have Movie Information, Seat Information, and User Information all divided into three separate databases.

The tradeoff between having multiple databases over having just one would be that having one database would create a simple user interface for the user to see. We will be able to store information in multiple parts and only have to access one database to view all the information we need over going to different ones. The only downside of this would be that it would be a little less efficient as there would be lots of data stored in a singular database. The benefits of having multiple databases would be that we would be able to create the website to be more organized and have a bit more information.

A. Appendices

Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS's overall set of requirements.

Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.

A.1 Appendix 1

A.2 Appendix 2