Theater Ticketing System

Software Requirements Specification
Version 4.0
November 2nd, 2023

Group #7
Jesus Bolanos
Destiny Tudara
Alina Antonova

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

Revision History

Date	Description	Author	Comments
9/21/23	Version 1	Group 7	Theater Ticketing System
10/05/23	Version 2	Group 7	Software Development Specification
10/19/23	Version 3	Group 7	Verification Test Plan
11/2/23	Version 4	Group 7	Data Management Strategy

Document Approval

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

Signature	Printed Name	Title	Date
	<Your Name>	Software Eng.	
	Dr. Gus Hanna	Instructor, CS 250	

Table of Contents

1. Introduction

The Theater Ticketing System is a comprehensive software solution aimed at facilitating ticket booking for multiple theaters within the San Diego theater chain. This document outlines the functional and non-functional requirements for the system's development.

1.1 Purpose

The primary aim of this SRS is to furnish a comprehensive and well-organized delineation of the Theater Ticketing System's requisites. It functions as a point of reference for the development team and stakeholders, fostering a shared comprehension of the project's extent and aims

1.2 Scope

1.2.1 Software Product(s)

- *The Theater Ticketing System*

1.2.2 Description

- *The Theater Ticketing System allows customers to book tickets online and using an in-person digital kiosk.*
- *Customers can book multiple tickets within San Diego's theater chain.*
- *The Theater Ticketing System will allow users to select multiple languages to include English,Spanish, and Swedish.*
- *Up to a 10 million user load will be supported by the system*
- *Users can receive their tickets digitally and/or physically through a printable copy.*
- *Security measures will be enforced to prevent*

1.2.3 Application

- *The Theater Ticketing System will be used by customers that visit theaters all across San Diego in order to book tickets.*

1.2.4 Objectives

- *To provide a user-friendly interface for customers to book tickets.*
- *Allow for multiple languages to cater to a wide-range of customers*
- *Create unique ticket identification numbers to prevent fraud*
- *Handle a large amount of customers simultaneously to prevent the system crashing*

1.4 References

This subsection should:
(1) Provide a complete list of all documents referenced elsewhere in the SRS, or in a separate, specified document.
(2) Identify each document by title, report number - if applicable - date, and publishing organization.
(3) Specify the sources from which the references can be obtained.
This information may be provided by reference to an appendix or to another document.

1.5 Overview

1.5.1 Content

- The following sections of this Software Requirements Specification (SRS) will delve into the Theater Ticketing System's functional and non-functional needs. It will cover distinct use cases, describe user interactions and system interactions, outline the data the system retains, and enumerate the system's anticipated constraints and obligations. Additionally, the SRS will touch upon any presumptions and dependencies that might affect the system's design and functioning.

1.5.1.2 SRS Organization

- **Section 2. General Description**
- **Section 3. Specific Requirements**
- **Section 4. Analysis Models**
- **Section 5. Change Management Process**
- **Appendices**

2. General Description

The Theater Ticketing System is a software development tool designed to transform the theatergoing experience for the San Diego theater chain's audience. As the trend toward digital solutions continues to grow, there's an imperative for a system that addresses the diverse needs of theatergoers while streamlining operations for theater administrators.

2.1 Product Perspective

The Theater Ticketing System is an independent product that should be used for integration with existing systems in theaters

2.2 Product Functions

- *The Theater Ticketing System allows for booking tickets, processing payments, and providing a digital or physical copy of the ticket.*
- *Customers will be able to create accounts and partake in a loyalty program.*
- *Users are able to select specific seats during their reservations*
- *Customers are allowed to use a variety of payment methods to include PayPal, Debit Card, Credit Card, and CryptoCurrency.*

2.3 User Characteristics

The system provides a user-friendly experience to in-person and online customers that is not dependent on having a technical background.

2.4 General Constraints

- *The Theater Ticketing System must be able to handle 10 million users simultaneously*
- *There must be security measures in place to prevent theft and fraudulent activity*

2.5 Assumptions and Dependencies

This section delineates the assumptions and dependencies that have the potential to influence the requirements detailed in this SRS. These elements denote external conditions or alterations that could have an impact on the design and operation of the Theater Ticketing System. Diligent attention to these assumptions and dependencies is imperative to ensure the effective development and deployment of the system.

2.5.1 Assumptions

- We assume that the designated hardware for the software product will be equipped with the necessary operating system. Any variance from this assumption may require modifications to the SRS
- It is presupposed in this SRS that external payment services, including credit card processors, PayPal, and Bitcoin gateways, will persist in their availability and functionality throughout the system's operational phase.
- The assumption made within this SRS is that internet connectivity will be consistently accessible for online users. Any constraints or interruptions in internet access could potentially impact the system's usability.

2.5.2 Dependencies

- The Theater Ticketing System is dependent on third-party APIs for tasks such as currency conversion and retrieval of online review data. Any alterations or interruptions in the functioning of these APIs have the potential to influence the system's performance.
- The hardware specifications associated with devices utilized within the theater environment, encompassing digital kiosks, serve as critical dependencies. Any alterations or enhancements to these hardware specifications may necessitate corresponding adjustments within the system
- The functionality of the Theater Ticketing System hinges on the continued availability and optimal performance of the designated database, which stores information related to showtimes, ticket availability, and customer data. Any challenges related to the database's availability can impact the system's operation.
- The operational functionality of the system is contingent upon adherence to pertinent legal regulations, including those governing data privacy and industry standards. Revisions to these regulatory frameworks may obligate modifications to the system to ensure ongoing compliance.

3. Specific Requirements

This section outlines detailed specifications for the Theater Ticketing System, driven by customer needs in Section 2. These requirements ensure a correct system design, testability, and user *satisfaction without overly restricting development. Organized into subsections, they cover external interfaces, functionality, non-functional aspects, and constraints, facilitating effective communication between stakeholders and the development team.*

3.1 External Interface Requirements

3.1.1 User Interfaces

3.1.1.1 Ticket Selection Interface

- The Ticket Selection User Interface must allow customers to accurately select movie tickets.
- This requirement is tied to the customer's need for interacting with ticketing features, as outlined in Section 2
- Present movie options, showtimes, and ticket prices in a clear and straightforward manner
- Ensure that customers can successfully choose tickets for their preferred movie and showtime.
- This function holds high priority and is fundamental for customer satisfaction, increasing sales,and to reduce cart abandonment rates.
- Provide options for selecting both regular and deluxe tickets, along with seat preferences.
- Maintain uniformity through consistent layout and user interactions across all theaters in San Diego.

3.1.1.2 Transaction Processing Interface

- The Transaction Processing Interface must accurately and securely process customer payments for selected tickets.
- This requirement is aligned with the necessity for customers to securely complete payment transactions, as specified in Section 2.
- Clearly present payment options, including credit card, PayPal, and Cryptocurrency, to customers in an unambiguous manner.
- Implement secure and dependable payment processing mechanisms to confirm the successful completion of customer transactions.
- This requirement holds a high priority as it is fundamental for the successful completion of ticket purchases.
- The interface should enable customers to review their ticket selections, apply discounts, and confirm their purchase before initiating the transaction
- Maintain a consistent payment processing flow across all theaters within the system to ensure a consistent user experience.

3.1.1.3 Booking Confirmation Interface

- The Booking Confirmation Interface must precisely confirm customer reservations, including movie selection, showtime, and seat assignments.

- This requirement is in sync with the customer's need for clear and dependable booking confirmations, as outlined in Section 2.
- Present booking confirmations to customers in a format that is easily comprehensible, including essential details such as booking date and time, movie title, showtime, and seat information.
- Implement mechanisms to consistently and reliably deliver booking confirmations to customers following successful reservations.
- This is a top-priority requirement as it directly influences the customer's experience and provides them with confidence in their reservation.
- The interface should encompass all pertinent booking details and instructions for customers, ensuring they have a comprehensive grasp of their reservation
- Maintain a consistent process for booking confirmations across all theaters within the system to guarantee a standardized and dependable user experience.

3.1.1.4 Account Registration Interface

- The Account Registration Interface must precisely and securely collect customer information during the registration process, encompassing personal details and contact information.
- This requirement is in sync with the customer's need to establish user accounts, as elaborated in Section 2, enabling them to unlock personalized features and advantages.
- Present the registration process in an intuitive and transparent manner, guiding customers through each step to input the required information.
- Implement verification procedures, such as email confirmation, to validate the authenticity of newly registered accounts.
- This is a top-priority requirement, as it lays the groundwork for customer interaction and the delivery of personalized services within the system.
- The interface should collect all pertinent customer details, including name, email, and password, to create a comprehensive user account.
- Maintain a consistent account registration process across all theaters within the system, ensuring a standardized and dependable user experience.

3.1.2 Hardware Interfaces

3.1.2.1 Digital Kiosks Interface

- Ensure the Theater Ticketing System operates seamlessly with theater-installed digital kiosks, enabling customers to purchase tickets and access system features in person with precision.
- Maintain a uniform and dependable user experience across all digital kiosks, aligning with customer expectations for reliable in-person ticket transactions.
- Design the digital kiosk interface to be clear and intuitive, simplifying the ticket purchasing process for customers.

- Implement mechanisms for verifying successful ticket transactions through digital kiosks, instilling confidence in customers.
- Given the critical role of digital kiosks in in-person ticket purchases, this requirement holds a top priority.
- Ensure that digital kiosks offer a complete range of ticket options, including regular and deluxe tickets, to cater to customers' preferences

3.1.2.2 Theater Hardware Interface

- Guarantee the compatibility and precision of the Theater Ticketing System with theater-specific hardware components, including ticket printers and scanners. This ensures accurate ticket validation and access control.
- This requirement aligns with the necessity for customers to have confidence in the precision and reliability of hardware interactions during movie screenings.
- Design theater hardware interfaces to facilitate clear and efficient ticket validation, elevating the overall customer experience.
- Implement mechanisms to confirm the successful validation of tickets using theater-specific hardware, ensuring seamless access to movie screenings.
- Given the pivotal role of precise hardware interactions in delivering a seamless theater experience, this requirement holds paramount importance.
- Theater hardware interfaces must encompass all facets of ticket validation and access control, eliminating any room for ambiguity
- Maintain a uniform process for hardware interactions across all theaters within the system, ensuring a standardized and reliable customer experience

3.1.2.3 Mobile Devices Interface

- Ensure customers can securely and accurately access the Theater Ticketing System via their mobile devices, such as smartphones and tablets. This includes delivering a responsive and user-friendly experience across different mobile platforms.
- This requirement corresponds to the need for customers to enjoy consistent and dependable access to the system's features through their mobile devices
- Design the mobile device interface to be clear and user-friendly, facilitating easy navigation and utilization for customers.
- Implement mechanisms for verifying successful ticket transactions and interactions on mobile devices, instilling customer confidence in their bookings.
- Given the paramount importance of mobile accessibility in modern ticketing systems, this requirement holds a high priority.
- Mobile interfaces should encompass all relevant features and functions available on other platforms, delivering customers a comprehensive experience
- Maintain a uniform user interface and functionality across various mobile platforms, ensuring consistency and reliability.

3.1.2.4 Box Office Hardware Interface

- Ensure the Theater Ticketing System harmonizes seamlessly with box office hardware, including point-of-sale systems. This enables precise in-person ticket sales and efficient refunds by theater staff.
- This requirement corresponds to the need for customers to trust the accuracy and reliability of box office transactions
- Design the box office hardware interface to facilitate transparent and efficient ticket transactions, streamlining the in-person ticketing process
- Implement mechanisms for confirming the successful completion of box office transactions, instilling customer confidence during in-person interactions.
- Given the vital role of box office interactions in customer service and in-person ticketing, this requirement holds a high priority
- The box office hardware interface must encompass all aspects of ticket sales, refunds, and customer service, leaving no functional gaps.
- Maintain a consistent process for in-person ticket transactions and customer service across all theaters within the system, ensuring a standardized and reliable experience.

3.1.3 Software Interfaces

3.1.3.1 Database Integration Interface

- The system should seamlessly interact with the database to retrieve and update information related to showtimes, ticket availability, customer profiles, and other relevant data. This integration must prioritize data accuracy, integrity, and security.

3.1.3.2 Payment Integration Interface

- The system should establish interfaces with external payment services, including credit card processors, PayPal, and Bitcoin gateways, to ensure the secure processing of customer payments for ticket purchases. These interfaces must prioritize the correctness and security of financial transactions.

3.1.3.3 Third Party API Interface

- The system's functionality relies on the integration and maintenance of third-party APIs, which are essential for tasks such as currency conversion and retrieval of online movie reviews from reputable sources. These integrations must uphold the accuracy of currency conversion and access to reliable movie reviews.

3.1.3.4 Loyalty Program Management Interface

- For customers enrolled in loyalty programs, such as the AMC membership card, the system should offer an interface to efficiently manage and validate membership accounts. This feature allows users to seamlessly access reserved tickets and associated benefits

3.1.4 Communications Interfaces

3.1.4.1 Email Communication Interface

- Ensure that the information contained in the emails, such as booking confirmations and updates, is factually accurate and error-free. Any inaccuracies could lead to confusion and dissatisfaction among customers.
- Establish a clear link between each email and the specific actions or interactions of the customer. This ensures that the system can track which events triggered the emails and helps with auditing and customer support.
- Craft the email content in a way that is easy to understand and visually pleasing. Use clear language, formatting, and design to enhance readability and comprehension.
- Implement mechanisms to verify that the emails are not only sent but also successfully delivered to the intended recipients. This confirmation ensures that customers receive the information they need.
- Prioritize the sending of emails based on their relevance and timeliness. For instance, booking confirmations should take precedence over general promotional emails.
- Include all the necessary types of emails that customers may expect, such as booking confirmations, updates on ticket availability, and promotional offers.
- Maintain a consistent style and format for emails across all theaters within the system. This uniformity ensures that customers have a standardized and reliable email experience

3.1.4.2 In-App Notifications Interface

- Ensure that the content of in-app notifications, such as updates on new movie releases or loyalty program benefits, is accurate and up-to-date. Misinformation can lead to user confusion.
- Associate each in-app notification with the specific user actions or system events that triggered them. This tracking helps users understand why they received a particular notification.
- Design the in-app notifications to be visually clear and concise, so users can quickly grasp their meaning and take appropriate actions if necessary.
- Implement mechanisms to verify that in-app notifications are not only sent but also properly displayed to users. This confirmation ensures that users receive and see the notifications.
- Prioritize the delivery of in-app notifications based on their relevance and urgency. For example, time-sensitive updates should be displayed promptly.
- Ensure that all relevant types of notifications, such as movie release updates or loyalty program alerts, are covered to provide users with a comprehensive experience.
- Maintain a consistent design and functionality for in-app notifications across various mobile platforms. This consistency ensures that users have a uniform and dependable experience regardless of the device they use

3.1.4.3 Customer Support Interface

- Provide customers with accurate and reliable support responses to address their inquiries and concerns. Incorrect or misleading information can lead to dissatisfaction

- Establish a clear connection between customer support interactions and the specific inquiries or issues raised by customers. This traceability helps in tracking and resolving customer concerns effectively
- Communicate with customers in a clear and understandable manner to facilitate issue resolution and enhance the overall support experience.
- Implement mechanisms to verify that customer support requests are addressed and resolved satisfactorily. This verification ensures that customers receive the assistance they need
- Prioritize responsive and effective customer support to ensure that customers' inquiries and issues are handled promptly and professionally.
- Cover all necessary communication channels for customer support, such as live chat and email, to accommodate various customer preferences and needs
- Maintain a consistent approach to customer support interactions across all theaters within the system. This consistency ensures that customers receive a standardized and reliable level of support

3.1.4.4 Admin Notification Interface

- Deliver accurate and reliable admin notifications regarding system events and issues. Admins rely on these notifications to make informed decisions
- Establish clear links between admin notifications and the specific system conditions or activities that triggered them. This traceability aids in diagnosing and addressing system-related issues effectively
- Present admin notifications in a clear and actionable manner to enable administrators to take prompt and appropriate actions in response to the notifications.
- Implement mechanisms to verify that admin notifications are not only sent but also acknowledged and acted upon. This verification ensures that system administrators receive and act on important notifications.
- Prioritize the delivery of admin notifications based on their urgency and relevance to system monitoring and issue resolution
- Ensure that all relevant system events and issues are covered by admin notifications to provide administrators with a comprehensive overview of system health
- Maintain a consistent approach to presenting admin notifications across all theaters within the system. This consistency ensures that administrators have a standardized and dependable experience when monitoring the system

3.2 Functional Requirements

This section delineates precise features and functionalities that are imperative for the Theater Ticketing System's design, development, and testing processes. These functional requirements offer a comprehensive understanding of the system's objectives in delivering effective solutions to meet customer demands

3.2.1 User-Friendly Interfaces

3.2.1.1 Introduction

- This functional requirement centers on shaping the user interface of the Theater Ticketing System. The paramount objective is to ensure that the system is easy for customers to engage with, facilitating actions such as searching for movies, verifying showtimes, selecting seats, and completing ticket purchases. Establishing a user-friendly interface is pivotal in laying the foundation for a positive user experience, which in turn is vital for both customer satisfaction and the overall success of the system.

3.2.1.2 Inputs

- User inputs encompass the information and actions provided by users during their interactions with the system. Within the context of user-friendly interfaces, this encompasses user activities like choosing a movie, specifying a showtime, indicating seating preferences, and entering payment details. It is imperative to design input methods that are instinctive, appropriately labeled, and responsive to user actions to minimize errors and alleviate any potential frustrations

3.2.1.3 Processing

- Processing refers to the system's handling of user inputs and requests. In the context of user-friendly interfaces, streamlined processing is critical in providing users with swift responses and seamless interactions. For instance, when a user selects seats, the system should swiftly update seat availability and pricing information. Effective processing contributes significantly to a polished and responsive interface

3.2.1.4 Outputs

- Outputs encompass the information and feedback that the system provides to users. In user-friendly interfaces, outputs should offer clear and concise information about movie choices, showtimes, ticket pricing, seat availability, and booking confirmations. The presentation of this information should be visually appealing, well-organized, and readily comprehensible to users

3.2.1.5 Error Handling

- Error handling is an indispensable component for addressing any issues or missteps that users might encounter while using the interface. User-friendly interfaces should incorporate informative error messages and notifications that are specific, guiding users on the necessary steps to rectify the error. Effective error resolution not only prevents user frustration but also helps users navigate the system with ease.

3.2.2 Secure Payment Processing

3.2.1.1 Introduction

- This functional requirement centers on the paramount task of upholding the security of payment processing within the Theater Ticketing System. It is of utmost importance to protect sensitive financial information and transactions, as this is crucial for earning and retaining the trust of customers. The assurance of secure payment processing serves as a bulwark for safeguarding customer data and thwarting any fraudulent activities.

3.2.1.2 Inputs

- Inputs, within the realm of secure payment processing, encompass the sensitive information that users furnish during transactions. This includes details such as credit card credentials, PayPal account particulars, or information related to Bitcoin wallets. The stipulation here is to establish secure channels through which users can input this information, ensuring robust encryption and shielding it from any unauthorized access

3.2.1.3 Processing

- Processing pertains to how the system manages payment information. To ensure secure payment processing, it is imperative to implement robust encryption protocols and adhere to industry security standards (e.g., PCI DSS for credit card transactions). Processing must guarantee that payment data is securely conveyed to payment gateways, meticulously verified for accuracy, and processed without exposure to potential threats

3.2.1.4 Outputs

- Outputs, in this context, denote the outcomes of payment processing. This encompasses confirmation messages for successful transactions and the receipts that customers receive. In the realm of secure payment processing, it is imperative that these outputs are generated and disseminated securely to customers, preserving the sanctity of their payment details in the process.

3.2.1.5 Error Handling

- Error handling assumes a critical role in the domain of secure payment processing, particularly for addressing complications that may arise during transactions. These scenarios may involve declined payments or payment processing failures. Effective error handling should guarantee that sensitive payment data remains concealed within error messages and that users are provided with lucid instructions on how to securely rectify any issues that may surface

3.2.3 Reservation and Booking System

3.2.1.1 Introduction

- The Ticket Reservation and Booking System constitute a critical module within the Theater Ticketing System. Its primary function is to facilitate the seamless acquisition of movie tickets, enabling customers to efficiently plan and secure their theater visits

3.2.1.2 Inputs

- Within the realm of this system, user inputs serve as the keystones of customer interactions. Customers submit information encompassing movie selections, showtimes, seat preferences, and any applicable promotional codes or discounts. The system's responsibility is to adeptly parse and process these inputs, ensuring the facilitation of booking transactions with precision

3.2.1.3 Processing

- This aspect involves the intricate processing steps that the system undertakes. It encompasses seat availability checks, the dynamic calculation of pricing based on discounts, and the secure execution of payment transactions. In this orchestration, precision and speed are paramount. The system must execute these operations accurately and expeditiously to deliver a seamless booking experience

3.2.1.4 Outputs

- In the context of the Ticket Reservation and Booking System, outputs represent the culmination of the user journey. These outputs encompass booking confirmations, reservation particulars, and the issuance of electronic tickets. It falls upon the system to ensure the timely and accurate generation and delivery of these outputs to customers, thereby furnishing them with the requisite information for their theater experience.

3.2.1.5 Error Handling

- Even in the most finely tuned systems, anomalies can arise. Error handling and resolution constitute the system's ability to gracefully manage issues that may surface during booking transactions. Be it payment discrepancies, seat unavailability, or technical glitches, the system should respond with articulate error messages and offer guidance to customers for issue resolution, ensuring a positive and frictionless customer experience

3.2.4 Customer Loyalty Program

3.2.1.1 Introduction

- The "Customer Loyalty Program" functional requirement introduces a pivotal component within the Theater Ticketing System. This loyalty program is crafted to elevate the customer experience by acknowledging and retaining devoted patrons. In this section, we outline the precise prerequisites for the efficient implementation and management of the loyalty program.

3.2.1.2 Inputs

- Input parameters for the loyalty program encompass customer data, comprising personal details and purchase history. It also incorporates user interactions within the system. These inputs play a critical role in ascertaining eligibility for the loyalty program, monitoring user behavior, and furnishing tailored benefits and incentives.

3.2.1.3 Processing

- The aspect of program logic pertains to the system's algorithms for computing and granting loyalty points. These points are awarded based on various customer interactions, such as ticket acquisitions, referrals, or participation in promotional campaigns.

Furthermore, the system should proficiently process requests for redeeming accrued loyalty points, enabling customers to access rewards like discounts or complimentary tickets.

3.2.1.4 Outputs

- Outputs stemming from the loyalty program encompass notifications sent to customers regarding accumulated points, available rewards, and redemption choices. The system must be adept at producing and conveying these outputs promptly and informatively. This fosters customer engagement and encourages active participation in the loyalty program.

3.2.1.5 Error Handling

- In the context of the loyalty program, issue resolution entails addressing complications linked to point accrual, redemption, or program eligibility. For instance, if a customer encounters difficulties while redeeming loyalty points, the system should provide unambiguous instructions for resolution. Proficient error handling ensures that customers maintain a positive and gratifying experience during their involvement in the loyalty program

3.2.5 Communication and Notification System:

3.2.1.1 Introduction

- The functional requirement concerning the "Communication and Notification System" brings forth a pivotal element of the Theater Ticketing System. This system shoulders the responsibility of enabling efficient communication among the system, customers, and other stakeholders. It encompasses a spectrum of communication modes, spanning emails, in-app notifications, and customer support interactions. This section delineates the precise criteria governing the design and functionality of this system.

3.2.1.2 Inputs

- Inputs to the Communication and Notification System comprise a tapestry of customer data, system events, customer inquiries, and user interactions. These inputs serve as catalysts for triggering and tailoring diverse forms of communication, guaranteeing the delivery of pertinent and well-timed information to customers.

3.2.1.3 Processing

- The operational facet encapsulates the system's intelligence in determining the nature, timing, and content of communications. It encompasses the establishment of rules and algorithms governing the creation and dispatch of emails, in-app notifications, and customer support responses. Furthermore, the system must possess the capacity to prioritize communications based on factors like urgency and relevance.

3.2.1.4 Outputs

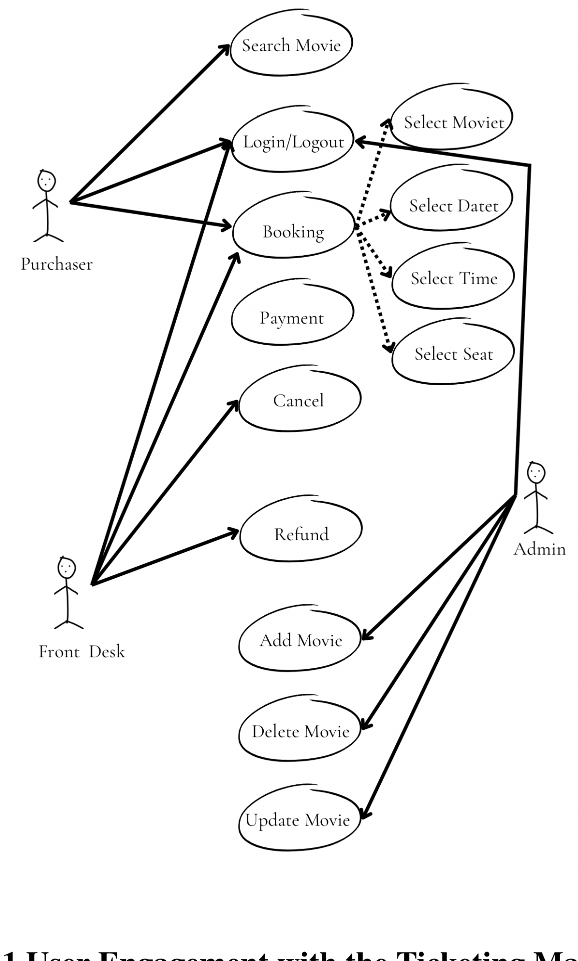
- Outputs emerging from the Communication and Notification System encompass a repertoire of emails, in-app notifications, and customer support interactions. These outputs are forged and disseminated to customers and stakeholders, acting as conduits for the transmission of information, updates, promotions, and support resolutions. The system bears the responsibility of ensuring that these outputs are lucid, pertinent, and reach their intended recipients effectively

3.2.1.5 Error Handling

- Within the precincts of the Communication and Notification System, issue resolution entails the handling of challenges linked to unsuccessful communication attempts, undelivered messages, or customer support queries. The system should incorporate mechanisms for error detection, notification, and remediation, thereby upholding the ethos of efficient communication and customer contentment

3.3 Use Cases

Create a program that encompasses various functions to streamline the process of managing a movie theater. The functions include, but are not limited to, purchasing tickets, keeping track of ticket availability, view and select showings, process payments, as well as administrative tasks.



3.3.1 User Engagement with the Ticketing Management System

This use case covers all the main interactions a user might experience with the ticketing system

3.3.1.1 Primary actor

- Purchaser

3.3.1.2 Secondary actor

- System admin, Front Desk team

3.3.1.3 Interfaces

- In-person Theater kiosk and online platform

3.3.1.4 Preconditions

- User has access to the online platform or the digital kiosk
- User has an account registered or they have selected a movie and showtime

3.3.1.5 Postconditions

- User has either a digital or printed ticket
- Proof of payment has been confirmed

3.3.1.6 Normal Course

1. User browses available movies and showtimes
2. User selects their desired showtime and seat
3. User continues to payment options
4. User selects a method of payment
5. User receives a digital or physical ticket

3.3.1.7 Alternative Courses

- User decides to cancel their transaction and exits the program
- User requests refund or exchange

3.3.2 Front Desk Management System

This use case outlines all the fundamental actions the Front Desk can execute with the ticketing system

3.3.2.1 Primary actor

- Front desk

3.3.2.2 Secondary actor

- User, System Admin

3.3.2.3 Interfaces

- Ticket booth system

3.3.2.4 Preconditions

- Front Desk staff are logged into the ticket booth system
- Purchaser approaches the front desk

3.3.2.5 Postconditions

- Customers no longer need assistance with booking, refunds, etc.

3.3.2.6 Normal Course

1. Staff assist customers walking in to book tickets
2. Aids to in-person questions about showtimes and availability
3. Handles refunds and exchanges
4. Provide physical tickets to customers

3.3.2.7 Alternative Courses

- The Front Desk directs customers to the online platform for exclusive deals or reservations.
- The Front Desk reaches out to the System Admin regarding technical issues

3.3.3 System Administrator Management System

This use case centers on the backend activities and system administration functions carried out by the System Admin

3.3.3.1 Actors

- System Admin

3.3.3.2 Interfaces

- Admin Graphical User Interface

3.3.3.3 Preconditions

- System Administrators are logged into the admin portal

3.3.3.4 Postconditions

- System configurations are modified, user disagreements are settled, technical problems are rectified, and essential data is either updated or backed up.

3.3.3.5 Normal Course

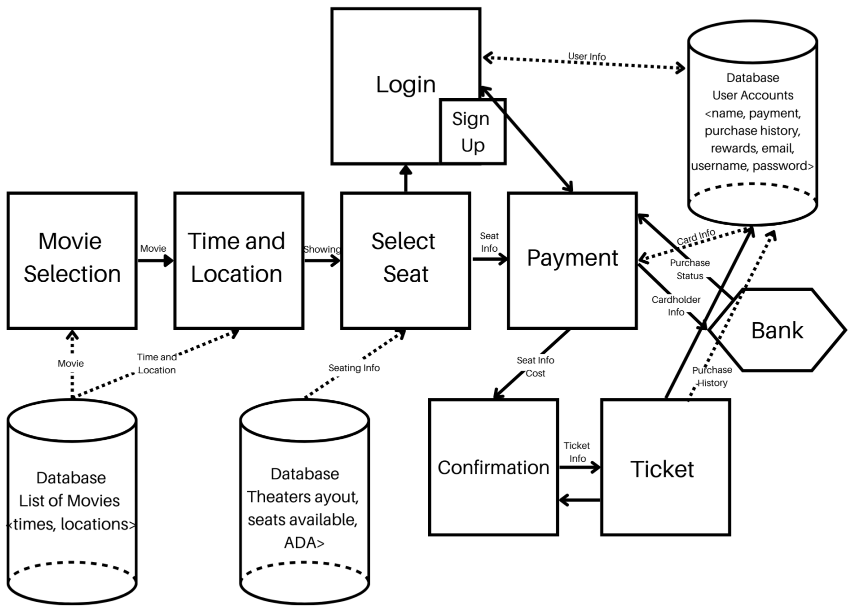
1. Modify system configurations as required.
2. Manage and settle user disagreements or ticketing concerns
3. Tackle any system-related technical glitches
4. Oversee and maintain security protocols
5. Refresh show timings or oversee show schedules

3.3.3.6 Alternative Courses

- System Admin reverts changes because of unforeseen mistakes
- System Admin liaises with external vendors regarding API or payment gateway concerns.

3.4 Classes / Objects

Figure 1. Architectural diagram of all major components



1. Login and Sign Up:

At the starting point, users can either "Login" or "Sign Up".

The "Login" component interacts with the "Database" containing user info to authenticate user credentials, which includes details like name, payment info, purchase history, rewards, email, username, and password.

2. Movie Selection:

    After logging in, users move to the "Movie Selection" component.
    This component communicates with a "Database" containing a "List of Movies", including their times and locations. Users select their desired movie from this list.

3. Time and Location:

    After selecting a movie, users are directed to choose the "Time and Location" for the Showing.

4. Select Seat:

    Following the time and location selection, users proceed to the "Select Seat" component. This step interacts with another "Database" detailing the theater's layout and seats available. Users select their desired seats based on seating information and availability.

5. Payment:

    After seat selection, users are redirected to the "Payment" phase.
    Here, the seat information and its associated cost become relevant.
    The payment phase also has a connection to the "Bank", which handles the actual financial transactions. The Bank's component showcases interactions related to "Card Info", "Purchase Status", and "Cardholder Info".
    The Bank keeps a "Purchase History" which likely tracks all transactions.

6. Confirmation:

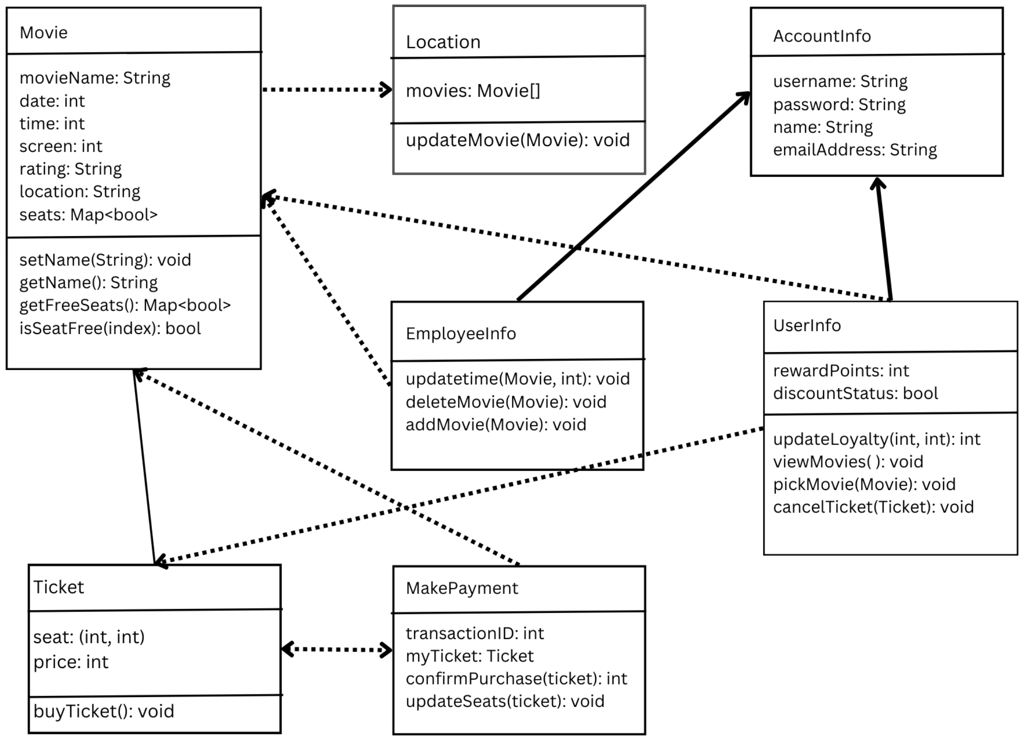
    Once the payment is processed and approved, users are moved to the "Confirmation" Step.

7. Ticket:

    After confirmation, the system generates a "Ticket" for the user, detailing the chosen movie, time, location, and seat.
    The "Ticket" component interacts with the "Confirmation" component to retrieve and display the ticket's details.

The flow of data between different steps is displayed using arrows, which indicate the progression from one step to the next. Additionally, dotted lines are used to show the interaction and data exchange between different processes and databases or external entities like the bank.

Figure 2. UML Class Diagram



1. Movie Class:
    - Attributes:
        - movieName: String – The title of the movie.
        - date: int – The date of the screening.
        - time: int – The time of the screening.
        - screen: int - The screen or auditorium number.
        - rating: String - The movie rating.
        - location: String - The venue or cinema location.
        - seats: Map<bool> - A mapping of seats to their availability status.
    - Methods:
        - setName (String): void - Sets the movie name.
        - getName (): String - Retrieves the movie name.
        - getFreeSeats () : Map<bool> - Returns a mapping of free seats.
        - isSeatFree (index): bool - Checks if a specific seat is available.
2. Location Class:
    - Attributes:
        - movies: Movie[]- An array containing movies available at this location.
    - Methods:
        - updatoMovie (Movie): void - Updates details of a specific movie.
3. AccountInfo Class:

Attributes:
- username: String - The user's username for login.
- password: String - The user's password.
- name: String - The real name of the user.
- emailAddress: String - The user's email address

4. EmployeeInfo Class:
   - Methods:
     - updatetime (Movie, int): void - Updates the screening time of a movie.
     - deleteMovie (Movie): void - Removes a movie from the listing.
     - addMovie(Movie): void - Adds a new movie to the listing.
5. UserInfo Class:
   - Attributes:
     - rewardPoints: int - Points accrued by the user for loyalty or promotions.
     - discountStatus: bool - A flag indicating whether the user is eligible for discounts.
   - Methods:
     - updateLoyalty(int, int): int -Updates the user's loyalty points.
     - viewMovies (): vold - Allows the user to view movie listings.
     - pickMovie (Movie): void - Lets the user select a movie.
     - cancelTicket (Ticket): void - Allows the user to cancel a booked ticket.
6. Ticket Class:
   - Attributes:
     - seat: (int, int) - A tuple representing the row and column of the booked seat.
     - price: int - The cost of the ticket.
   - Methods:
     - buyTicket (): void - Facilitates the ticket purchasing process.
7. MakePayment Class:
   - Attributes:
     - transactionID: int - A unique ID for the payment transaction.
     - myTicket: Ticket - The ticket linked to the payment.
   - Methods:
     - confirmPurchase(ticket): int - Confirms the purchase of a ticket.
     - updateSeats (ticket): void - Updates the seating arrangement post-purchase.

The 'Movie' class has a relationship with the 'Location' class, indicating that a location can have multiple movies.
Both 'UserInfo' and 'EmployeeInfo' classes interact with the 'Movie' class, indicating that both users and employees can view and modify movie details.
The 'Ticket' class is related to the 'MakePayment' class, illustrating that a ticket is linked to its respective payment transaction.

The 'UserInfo' class also has interactions with the 'AccountInfo' class, suggesting that each user's profile is tied to their account credentials.

3.4.1 Customer Class

3.4.1.1 Customer Class Attributes

Name

- Data Type: string
- Description: Stores the full name of the customer. Utilized for personalizing user interaction and communication.

UserID

- Data Type: string
- Description: A unique identifier assigned to each customer. Utilized to uniquely identify each user within the system.

3.4.1.2 Customer Class Functions

RegisterAccount(string name, string email, string password) : bool

Parameters:

- name: string. Full name of the user.
- email: string. Used for login and communication.
- password: string. Must adhere to specified rules (e.g., minimum length, special character).
- Return Type: bool. Returns true if account creation is successful, otherwise false.
- Description: Enables user account creation by validating input, ensuring email uniqueness, and storing user data in the database.

3.4.2 Ticket Class

3.4.2.1 Ticket Class Attributes

TicketID (string)

- Data Type: string
- Description: A unique identifier for each ticket, facilitating specific ticket retrieval, and management.

SeatNumber (int)

- Data Type: integer
- Description: The designated seat assigned upon ticket purchase, ensuring organized seating and avoiding double-booking.

3.4.2.2 Ticket Class Functions

ProduceTicket(string userID, int seatNumber, DateTime bookingDate) : string

- Parameters:
- userID: string. Identifies which user the ticket is assigned to.
- seatNumber: integer. Designates the specific seat assigned.

- bookingDate: DateTime. Timestamp of when the booking was made.
- Return Type: string. Returns the generated TicketID.
- Description: Upon payment confirmation, generates a ticket with a unique TicketID, associating it with the user, seat, and booking time.

3.4.3 Payment Class

3.4.3.1 Payment Class Attributes

PaymentID (string)

- Data Type: string
- Description: Uniquely identifies a specific payment transaction, facilitating tracking and management of each transaction.

Total (float)

- Data Type: float
- Description: Represents the total monetary amount of the purchase, ensuring accurate transaction and billing records.

3.4.3.2 Payment Class Functions

ProcessPayment(string userID, string paymentMethod, float total) : bool

- Parameters:
- userID: string. Associate the payment with a specific user.
- paymentMethod: string. Specifies the chosen method of payment (e.g., Credit, Debit).
- total: float. The total amount to be charged.
- Return Type: bool. Returns true if payment is processed successfully, otherwise false.
- Description: Initiates and processes the payment, ensuring accurate financial transactions and secure handling of user payment information.

RefundPayment(string paymentID) : bool

- Parameters:
- paymentID: string. Specifies the payment transaction to be refunded.
- Return Type: bool. Returns true if the refund is successful, otherwise false.
- Description: Manages the refund process for a specified payment, ensuring user satisfaction and accurate financial record-keeping.

3.5 Non-Functional Requirements

Non-functional requirements play a vital role in assessing the overall quality and performance of the Theater Ticketing System. They concentrate on characteristics that affect the system at a broader level, rather than delving into individual components or functions. These requirements are articulated using measurable criteria to guarantee a clear assessment and compliance with system-wide benchmarks.

3.5.1 Performance

- Clearly define specific response time objectives for activities like website loading or payment processing.
- Specify the number of transactions or operations that the system should handle within a given time frame.
- Detail how the system should adapt to accommodate increased user activity during peak periods while maintaining responsiveness
- Establish thresholds for the utilization of system resources such as CPU, memory, and bandwidth

3.5.2 Reliability

- Define the expected duration of uninterrupted operation before a failure is anticipated.
- Explain how the system should manage errors and continue functioning in the presence of faults.
- Specify the percentage of time the system must be available for use

3.5.3 Availability

- Establish the minimum acceptable uptime percentage, such as 99.9%
- Outline the maximum permissible downtime within a specified time frame, e.g., no more than one minute per day.

3.5.4 Security

- Specify the robustness of user authentication methods, such as password complexity or multi-factor authentication.
- Define the standards for encrypting data during transmission and storage
- Detail who can access specific system features and data, along with the associated conditions
- Ensure that the system adheres to pertinent data protection and privacy regulations

3.5.5 Maintainability

- Specify how easily the system can be altered to accommodate changes or incorporate new features.
- Define criteria for evaluating the system's components and functions during testing
- Describe the depth and comprehensiveness of system documentation designed to assist with maintenance

3.5.6 Portability

- The web-based interfaces of the system should seamlessly integrate with standard web browsers, such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari. The system must remain agile in accommodating shifts in browser technologies and versions to sustain its portability.
- The mobile applications associated with the system should demonstrate compatibility across a spectrum of mobile devices, encompassing varying screen sizes and resolutions.

The system's ability to acclimatize to alterations and advancements in mobile device technologies is pivotal

- The system should facilitate the integration of third-party services, APIs, and components that are conventionally utilized within the industry. It should possess the flexibility to adjust to shifts in third-party interfaces and technologies.
- The system should be equipped with robust localization support, enabling its utilization across diverse regions and languages. It should possess the versatility to incorporate new languages and account for cultural nuances.
- Where applicable, the system must offer cross-platform functionality, affording users seamless access across a myriad of devices and platforms. Future-proofing considerations should encompass compatibility with emerging platforms and technologies.

3.6 Inverse Requirements

- In cases where system updates or maintenance are necessary, the system should strive to minimize downtime to prevent disruptions to user services. Whenever feasible, downtime should be scheduled during off-peak hours.
- The system should be designed to avert the presentation of incorrect ticket prices to customers, assuring that they are accurately charged based on their selections.
- The system should implement safeguards to forestall data corruption, ensuring the accuracy and dependability of customer information, bookings, and transactions.
- The system should refrain from processing identical payment transactions more than once to avert potential overcharging of customers or financial discrepancies.
- The system should incorporate safeguards to avert overbooking of movie screenings, thereby ensuring that customers do not purchase more tickets than the available seating capacity permits.
- The system must establish robust measures to thwart unauthorized access to customer accounts and sensitive data, guaranteeing that only authenticated users can engage in actions like ticket booking and payment processing
- The system should take proactive measures to alleviate performance bottlenecks during peak hours or periods of high user traffic, ensuring that user interactions remain responsive.

3.7 Design Constraints

- Ensuring strict compliance with data privacy regulations, such as GDPR or CCPA, stands as a paramount concern. This constraint significantly influences the handling, storage, and protection of customer data within the system. Non-compliance may result in legal repercussions and harm to the system's reputation
- Conforming to established industry standards governing ticketing and payment processing holds utmost importance for fostering interoperability and trust. Failure to

meet these standards can lead to compatibility issues, which could, in turn, impact the system's ability to collaborate with other systems and services.
- Seamless integration with theater hardware components, including digital kiosks and ticket scanners, assumes critical significance in delivering a streamlined user experience. This constraint ensures that the software effectively communicates with theater equipment, enabling on-site ticket purchases and validation.

3.8 Logical Database Requirements

- Prescribe distinct data formats for various categories of information to guarantee uniformity and precision in data storage. This encompasses formats for customer particulars, movie specifics, showtimes, and payment data.
- Institute regulations and restrictions to uphold data integrity within the database. This entails defining validation protocols, unique identifiers, and interrelationships among data tables to preclude erroneous or incomplete data from infiltrating the system.
- Enforce robust access control mechanisms to shield sensitive data. Specify user roles and permissions to manage who possesses the authority to view, modify, or delete data within the database, thus ensuring the security and confidentiality of data
- Lay out protocols for periodic database backups and data retrieval procedures. This is indispensable for shielding data against unforeseen disruptions or data loss, thereby guaranteeing data accessibility and the uninterrupted operation of business processes

3.9 Other Requirements

- Detail the security requisites, encompassing authentication, authorization, encryption, and safeguards against common vulnerabilities like SQL injection and cross-site scripting.
- Precisely stipulate performance benchmarks, response times, and capacity requirements to assure the system's optimal performance under diverse circumstances
- Define the duration for which data will be retained, delineate archival or purging timelines, and take into account legal or compliance mandates concerning data retention.

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 Sequence Diagrams

4.3 Data Flow Diagrams (DFD)

4.2 State-Transition Diagrams (STD)

5. Development Plan and Timeline

5.1 Partitioning of Tasks

This section outlines the structured breakdown of the project into distinct phases: "Requirement Analysis," "System Design," "Implementation," "Testing," "Deployment," and "Maintenance and Support." Each phase is strategically defined with specific durations and milestones to streamline the development process, ensuring a systematic and organized approach towards the completion of the theater ticketing system.

5.1.1 Requirement Analysis and Finalization
Duration: 2 weeks
Detail: Analyze and finalize the functional and non-functional requirements of the system. This might involve discussions and meetings with stakeholders to ensure that all necessary features and constraints are considered.

5.1.2 System Design
Duration: 4 weeks
Detail: During this phase, the theoretical foundations for the system are crafted:

- Database Design: Define entities, relationships, and integrity rules to ensure coherent data management.
- UI/UX Design: Develop wireframes and prototypes, considering user journeys from login to ticket purchase and confirmation.
- System Architecture Design: Outline how different system components will interact, ensuring it supports all requirements and is scalable, maintainable, and reliable.

5.1.3 Implementation/Development
Duration: 3-4 weeks
Detail: Developers build the system based on the preceding designs, focusing on the following modules, among others:

- User Authentication: Develop secure login, registration, and profile management functionalities.
- Ticket Booking: Allow users to select shows, seats, and proceed to booking.
- Payment Processing: Implement a secure payment gateway for transactions.
- Admin Panel: Develop an admin panel to manage shows, schedules, and view booking history.

5.1.4 Testing
Duration: 8 weeks
Detail: Ensure the developed system is rigorously tested:

- Perform different types of testing to uncover and fix issues and ensure the system works flawlessly across all intended devices and platforms.

*5.1.5 Deploymen*t
Duration: 12 weeks
Detail: Deploy the system to a live environment where real users will interact with it. Monitor the system's performance and resolve any issues promptly.

5.1.6 Maintenance and Support
Ongoing
Detail: Regularly update the system, fix issues, and potentially add new features based on user feedback and system performance.

5.2 Team Member Responsibilities

This subsection delineates the specific roles and duties of each team member throughout the development of the theater ticketing system. From the Project Manager, who ensures adherence to timelines and budgets, to Developers, UI/UX Designers, Database Administrators, and more, each role is articulated to establish accountability and clarify expectations. This structure ensures that every aspect of the project, from conception to deployment, is underpinned by clear responsibility and expert attention.

5.2.1 Project Manager (PM)
Key Responsibility: Ensuring that the project progresses on time, within scope and budget.
Detailed Tasks: Identifying and managing risks, ensuring resource availability and allocation, communicating with stakeholders, and ensuring that the project objectives are met.

5.2.2 System Analyst
Key Responsibility: Ensuring the system is designed to meet user and business needs.
Detailed Tasks: Engaging with stakeholders to understand and document requirements, translating these requirements into technical specifications, and ensuring that the proposed solutions meet the identified needs.

5.2.3 UI/UX Designer
Key Responsibility: Ensuring the system is user-friendly and provides a positive user experience.
Detailed Tasks: Creating and iterating on wireframes and prototypes, conducting user testing sessions to validate designs, and ensuring the final design is implemented accurately during development.

5.2.4 Database Administrator (DBA)
Key Responsibility: Managing and ensuring the integrity of the database.
Detailed Tasks: Designing the database schema, ensuring data security, managing data back-ups, and optimizing database performance.

5.2.5 Developers
Key Responsibility: Building the system in accordance with design and requirements.

Detailed Tasks: Writing, testing, and deploying code, ensuring that it adheres to coding and performance standards, and documenting their work for future reference.

5.2.6 Quality Assurance (QA) Analyst
Key Responsibility: Ensuring the system works as intended and is free of bugs.
Detailed Tasks: Developing test cases and scripts, conducting testing, documenting results, and ensuring issues are resolved.

5.2.7 IT Support/DevOps
Key Responsibility: Ensuring continuous deployment and system availability.
Detailed Tasks: Managing system deployments, ensuring system uptime, and troubleshooting and resolving issues as they arise.

5.2.8 Customer Support and Training
Key Responsibility: Supporting users and ensuring they can effectively use the system.
Detailed Tasks: Managing user inquiries and issues, providing training and documentation to users, and communicating feedback to the development team.

6. Verification Test Plan

6.1 Features to be Tested

6.1.1 User Management

- Registration of new users.
- Logging in and out.
- Resetting forgotten passwords.
- Updating user profile information.

6.1.2 Movie Display and Selection

- Displaying currently showing movies.
- Filtering movies based on genre, rating, or date.
- Selecting a specific movie to view available showtimes.

6.1.3 Seat Selection and Booking

- Viewing available seats for a specific showtime.
- Selecting one or multiple seats.
- Holding the selected seats for a limited time during the booking process.

6.1.4 Payment Processing

- Providing multiple payment options (credit card, e-wallets, etc.).
- Secured payment data processing.
- Handling payment failures and successes.

6.1.5 E-ticket issuance and management

- Generating an e-ticket upon successful payment.
- Sending the e-ticket to the user's registered email.
- Allowing users to view their booked tickets within the app/website.

6.2 Test Sets/Vectors

6.2.1 User Account Creation and Management Feature

6.2.1.1 Management Feature Test Sets/Vectors

- Registration with valid details.
- Logging in with valid credentials.
- Logging out from an active session.
- Test Cases & Coverage

6.2.1.2 Management Feature Test Cases & Coverage

- UserRegistration_001: Validates the registration process using valid details.
- UserLogin_001: Ensures users can log in with valid credentials.
- UserLogout_001: Validates the logout mechanism ensuring a user can end an active session.

6.2.2 Display and Selection of Movies Feature

6.2.2.1 Movie Feature Test Sets/Vectors

- Viewing available movies.
- Selecting a specific movie for further details.

6.2.2.2 Movie Feature Test Cases & Coverage

- MovieSelect_001: Confirms that users can browse through available movies and select a specific one for more details or booking.

6.2.3 Seat Selection and Reservation Feature

6.2.3.1 Seat Selection and Reservation Test Sets/Vectors:

- Displaying available showtimes for a selected movie.
- Picking a specific showtime for booking.

6.2.3.2 Seat Selection and Reservation Test Cases & Coverage

- ShowTimeSelect_001: Demonstrates the user's ability to see available showtimes and choose one for further actions like seat selection.

6.2.4 Payment and Ticketing Feature

6.2.4.1 Payment and Ticketing Test Sets/Vectors

- Entering payment information.
- Integrating and processing payment through the gateway.

6.2.4.2 Payment and Ticketing Test Cases & Coverage

- PaymentInfoEntry_001: Validates that users can correctly input their payment details.
- Payment_Gateway: Ensures that the system can process payments through the integrated payment gateway successfully.

6.2.5 Discount & Promotion Feature

6.2.5.1 Discount & Promotion Test Sets/Vectors

- Applying a discount code during payment.

6.2.5.2 Discount & Promotion Test Cases & Coverage

- DiscountCodeApply_001: Checks the system's ability to recognize, validate, and apply discount codes during the payment process.

6.2.6 E-Ticket Generation Feature

6.2.6.1 E-Ticket Generation Test Sets/Vectors

- Successful Generation: Generating an e-ticket immediately after a successful payment.
- E-Ticket Details: Ensuring the e-ticket contains all necessary details like movie name, showtime, seat number, theater, and barcode or QR code for scanning.
- E-Ticket Delivery: Delivery mechanism – whether the ticket is displayed directly on the website, sent via email, or provided through an app.
- E-Ticket Storage: The capability for users to view their past e-tickets in their account.
- E-Ticket Cancellation/Modification: Allowing users to cancel or modify their bookings, and reflect these changes on the e-ticket.

6.2.6.2 E-Ticket Generation Test Cases & Coverage

- E-Ticket_Generation_Success: Validates the "Successful Generation" vector, ensuring that after a successful transaction, an e-ticket is instantly produced.
- E-Ticket_Details_Validation: Checks the "E-Ticket Details" vector to ensure that the generated e-ticket encompasses all essential details required for entry and verification.
- E-Ticket_Delivery_Method: Validates the "E-Ticket Delivery" vector, ensuring the delivery mechanism works as expected (like emailing the ticket or showing it within an app).
- E-Ticket_Storage_Access: Tests the "E-Ticket Storage" vector, allowing users to retrieve past e-tickets from their accounts.
- E-Ticket_Modification: Validates the "E-Ticket Cancellation/Modification" vector, allowing users to make changes to their bookings and verifying these changes reflect on the e-ticket.

7. Data Management Strategy

7.1 Database Choice: Relational Database (SQL)

7.1.1 Detailed Rationale

- Structured Schema: SQL databases provide a fixed schema ensuring data consistency.
- ACID Transactions: Ensures data remains consistent and reliable even in the event of failures.
- Query Optimization: SQL databases are optimized for complex queries beneficial for joining data from multiple tables.

7.2 Database Design

7.2.1 Tables

- Users Table: Stores authenticated users' data.
- Theaters Table: Stores information on different theaters.
- Shows Table: Information on shows including timing and theater details.
- Seats Table: Details on seating arrangement in each theater.
- Bookings Table: Tracks users' bookings, linking users, shows, and seats.
- Relationships:Entity-Relationship Diagram: Visually represents relationships between tables.

7.2.2 Database Schema and Relationships

- Users Table: Stores information such as UserID (Primary Key), Name, Email, and PasswordHash.
- Theaters Table: Contains details like TheaterID (Primary Key), Name, Location, and TotalSeats.
- Shows Table: Holds data on ShowID (Primary Key), TheaterID (Foreign Key), DateTime, MovieName, etc.
- Seats Table: Manages SeatID (Primary Key), TheaterID (Foreign Key), Row, Column, and Status.
- Bookings Table: Keeps track of BookingID (Primary Key), UserID (Foreign Key), ShowID (Foreign Key), SeatID (Foreign Key), Timestamp, and PaymentStatus.
- Relationships: These tables are interlinked, ensuring a relational structure that can efficiently model the theater booking domain. For instance, the Bookings table has foreign keys linking to Users, Shows, and Seats, creating relations that are essential for functionalities like viewing past bookings.

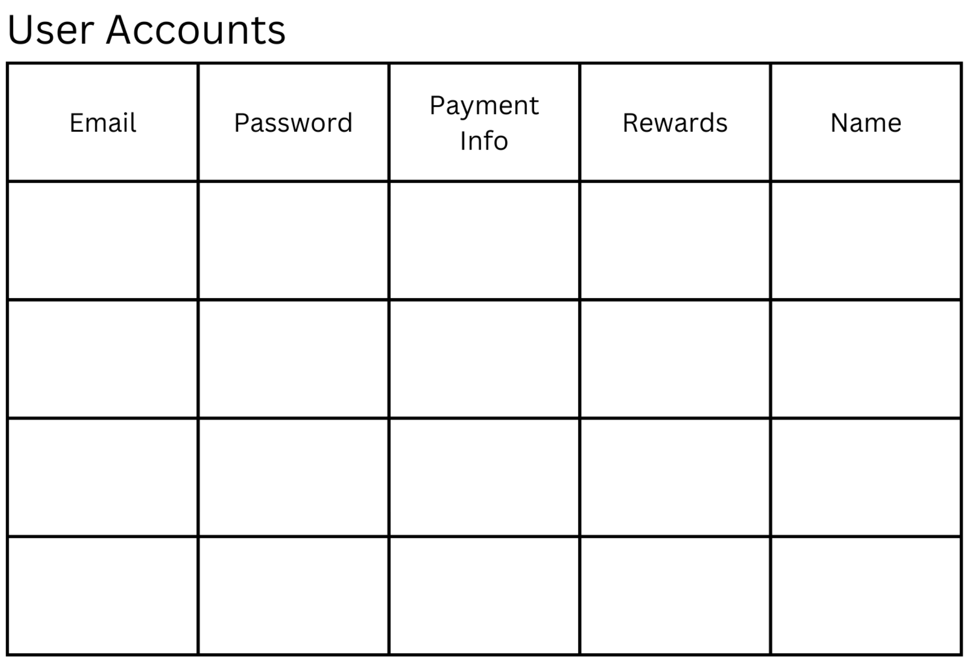
SQL Table

Theather Database

Theater ID	# of Seats	Layouts	ADA	Location
01	20	Reg	Y	San Diego
02	10	Delux	Y	Del Mar
03	50	Reg	Y	Solona Beach
04	100	Reg	Y	UTC

Movie Database

Name	Time	Rating	Genre	Duration	Theather ID	Available Seats
Barbie	5pm	★★★★★	Comedy	A1	01	1
Spider Man	5pm	☆☆☆☆☆	Comic	A2	02	0
Frozen	6pm	★★★★☆	...	A1	03	2
Moana	7pm	★★★☆☆	...	A2	04	3



User Accounts

Email	Password	Payment Info	Rewards	Name

7.3 Data Splitting

7.3.1 Partitioning

- Vertical Partitioning: Ensures tables contain only relevant data.
- Horizontal Partitioning (Sharding): Distributes rows based on a shard key across multiple database nodes.

7.4 Data Security

7.4.1 Encryption & Hashing

- Passwords: Hashing and salting passwords using algorithms like bcrypt.
- Data Encryption: Use encryption for sensitive data both in transit and at rest.

7.4.2 Access Control

- Role-Based Access Control (RBAC): Define roles and assign permissions.

7.5 Backup and Recovery

7.5.1 Strategies

- Regular Backups: Schedule daily or hourly backups.
- Recovery Plan: Ensure quick restoration of data.

7.6 Alternatives and Trade-offs

7.6.1 NoSQL Alternatives

- MongoDB: Offers flexibility but may require effort for data consistency.

7.6.2 Data Organization

- Denormalization: Could improve read performance but may lead to data inconsistency.

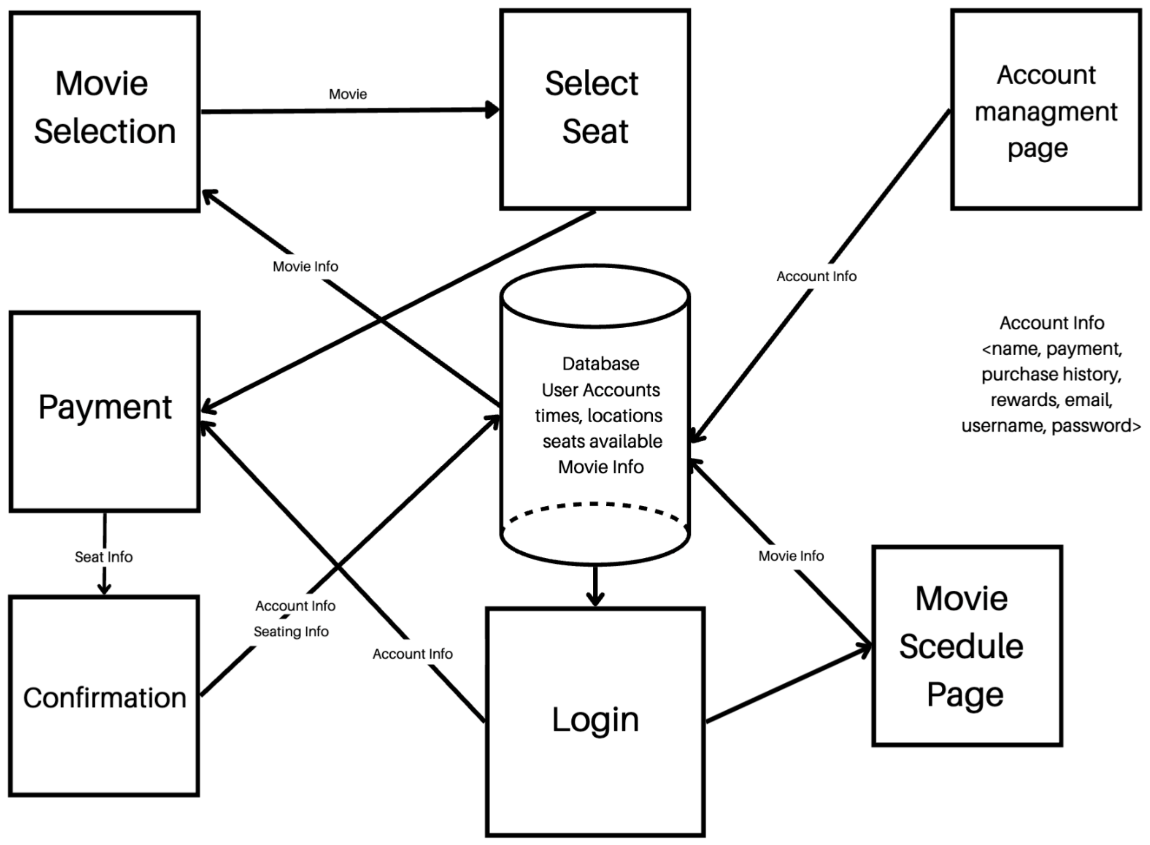
7.7 Trade-offs

7.7.1 SQL vs. NoSQL

- Schema and Relationships: SQL provides a clear schema and relationships.
- Scalability: NoSQL is more scalable but may require additional efforts for data integrity.

8. Design Specification

8.1 Updated Software Architecture Diagram



Description:

1. Movie Selection: This is the initial module where users choose the movie they wish to watch.

Output: The choice of movie, which is then relayed to the "Select Seat" module.
Interaction: This module fetches the movie info from the database to display available movies to the users.

2. Select Seat: After choosing a movie, users move to this module to pick their desired seats.
Output: Details about the chosen seat/s.
Interaction: The seat info is provided to the "Payment" module and also updates the database about which seats are now occupied for that specific movie and time.

3. Payment: Here, users proceed with the transaction for the movie tickets.
Input: Details about the selected seats and the specific movie.
Output: Payment status (successful or failed) and other related details.
Interaction: This module might interact with the database to store transaction details and confirm the seats' status.

4. Confirmation: After successful payment, this module provides a confirmation of the booking to the user.
Input: Information from the "Payment" module, seat details, and possibly the user's account details.
Output: A confirmation message or ticket details.
Interaction: Fetches seating and account information from the database to provide accurate confirmation details.

5. Account Management Page: This section offers features for users to view and manage their account details.
Input/Output: User account details like name, payment methods, purchase history, rewards, email, username, and password.
Interaction: Connects with the database to retrieve, display, or update user account details.

6. Movie Schedule Page: This module shows users the available schedules for different movies.
Input: Movie data from the database.
Output: Displayed schedules for the user.
Interaction: Fetches movie information, times, and locations from the database.

7. Login: A security checkpoint where users validate their credentials to access the system.
Input: User credentials (like username and password).
Output: Authentication status.
Interaction: Verifies user credentials against the stored data in the database.

8. Database: Central storage represented by a cylinder.

Contains:

User Accounts: Information related to registered users, including their personal and transactional details.

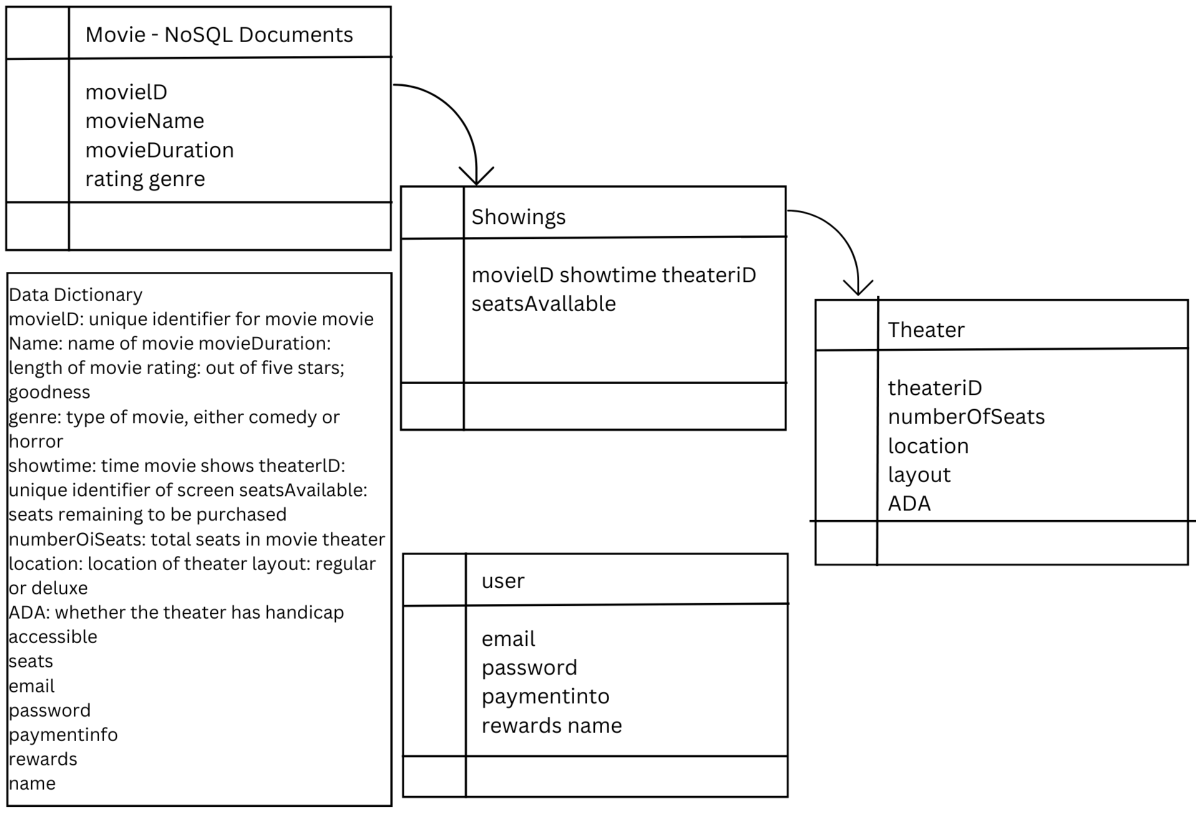
Times & Locations: Details of movie timings and possible venues or locations.

Seats Available: Keeps track of the seats that are booked or available for a particular movie and time.

Movie Info: Details about movies, such as title, duration, actors, etc.

Interaction: It interacts with almost all modules, providing necessary data and updating itself based on user actions.

8.2 Updated EntityRelationshipDiagram



Data Dictionary
movielD: unique identifier for movie movie
Name: name of movie movieDuration:
length of movie rating: out of five stars;
goodness
genre: type of movie, either comedy or
horror
showtime: time movie shows theaterlD:
unique identifier of screen seatsAvailable:
seats remaining to be purchased
numberOiSeats: total seats in movie theater
location: location of theater layout: regular
or deluxe
ADA: whether the theater has handicap
accessible
seats
email
password
paymentinfo
rewards
name

Description:

1. Movie - NoSQL Documents:

This document captures details of movies. Its attributes are:

movieID: This is a unique identifier assigned to each movie.

movieName: The name of the movie.

movieDuration: Represents how long the movie lasts, typically in minutes.

rating: It indicates the movie's rating, which is out of five stars.
genre: It specifies the type of movie. The given options in the diagram are comedy or horror.

2. Showings:
This document provides information about when a particular movie is being shown. Its attributes are:
movieID: Refers to the movie being shown. It links back to the "Movie" document to get movie details.
showtime: The time when the movie starts.
theaterID: This is an identifier to tell in which theater the movie is being shown.
seatsAvailable: The number of seats that remain unpurchased for that particular showing.

3. Theater:
This document provides details about the theaters. Attributes include:
theaterID: A unique identifier for each theater.
numberOfSeats: Total number of seats available in the theater.
location: Specifies where the theater is located.
layout: Describes the seat arrangement or style of the theater, for instance, regular or deluxe seating.
ADA: A boolean (True/False) value indicating if the theater is accessible to people with disabilities (handicap accessible).

4. User:
This document captures information about users or customers. Its attributes are:
email: The email address of the user.
password: User's password to access the system.
paymentinfo: Contains information related to user's payment methods.

9. Change Management Process

A. Appendices

A.1 Appendix 1

A.2 Appendix 2