Theater Ticketing System Project

Software Requirements Specification

Version <3.0>

02/14/2024

Group #16
Kaylie Pham
Aditya Bhagat
Ryan Hanna

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

Revision History

Date	Description	Author	Comments
02/14/2024	Version <1.0>	Group #16	
02/28/2024	Version <2.0>	Group #16	
03/13/2024	Version <3.0>	Group #16	

Document Approval

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

Signature	Printed Name	Title	Date
	Kaylie Pham, Aditya Bhagat, & Ryan Hanna	Software Eng.	
	Dr. Gus Hanna	Instructor, CS 250	

Table of Contents

REVISION HISTORY..... 2

DOCUMENT APPROVAL..... 2

1. INTRODUCTION..... 5

1.1 Purpose..... 5
1.2 Scope..... 5
1.3 Definitions, Acronyms, and Abbreviations..... 5
1.4 References..... 5
1.5 Overview..... 5

2. GENERAL DESCRIPTION..... 6

2.1 Product Perspective..... 6
2.2 Product Functions..... 6
2.3 User Characteristics..... 6
2.4 General Constraints..... 6
2.5 Assumptions and Dependencies..... 6

3. SPECIFIC REQUIREMENTS..... 6

3.1 External Interface Requirements..... 6
3.1.1 User Interfaces..... 6
3.1.2 Hardware Interfaces..... 7
3.1.3 Software Interfaces..... 7
3.1.4 Communications Interfaces..... 7
3.2 Functional Requirements..... 7
3.2.1 Ticket Purchasing..... 7
3.2.2 Bot Blocking..... 7
3.2.3 Administrator Mode..... 7
3.2.4 Feedback System..... 8
3.3 Use Cases..... 8
3.3.1 Use Case #1..... 8
3.3.2 Use Case #2..... 8
3.3.3 Use Case #3..... 9
3.4 Classes / Objects.....
3.4.1 <Class / Object #1>..... 9
3.4.2 <Class / Object #2>..... 9
3.5 Non-Functional Requirements..... 10
3.5.1 Performance..... 10
3.5.2 Reliability..... 10
3.5.3 Availability..... 10
3.5.4 Security..... 10
3.5.5 Maintainability..... 10
3.5.6 Portability..... 10
3.6 Inverse Requirements..... 10
3.7 Design Constraints..... 10
3.8 Logical Database Requirements..... 10

3.9 Other Requirements..... 11

4. SOFTWARE DESIGN SPECIFICATION……11

4.1 Software Architecture Overview……11

4.1.1 Architectural Diagram……11

4.1.2 Explanation of Architectural Diagram……12

4.1.2 UML Diagram……13

4.1.3 Explanation of UML Diagram……13

5. TEST PLAN……..14

5.1 Verification test Plan……14

5.2 Test Case Samples……14

6. ANALYSIS MODELS..... 15

6.1 Sequence Diagrams..... 15

6.3 Data Flow Diagrams (DFD)..... 15

6.2 State-Transition Diagrams (STD)..... 15

7. CHANGE MANAGEMENT PROCESS..... 15

A. APPENDICES..... 15

A.1 Appendix 1..... 15

A.2 Appendix 2..... 15

1. Introduction

This Software Requirements Specification (SRS) document defines and lists the functional and nonfunctional requirements for the development of a Theater Ticketing System. This document is intended for project managers, stakeholders, software engineers, and software developers involved in the design, development, and implementation of the Theater Ticketing System.

1.1 Purpose

The purpose of this document is to provide a detailed specification of the requirements for the development of a Theater Ticketing System. This system aims to assist and simplify ticket purchasing for customers. This system allows customers to either purchase in person at the theater through a digital kiosk or purchase from an online website.

1.2 Scope

This document identifies the software product, its characteristics (functional and nonfunctional) and constraints. It describes the application of the software, including its benefits, objectives, and goals. The Theater Ticketing System will execute ticket purchasing for customers, whether in person at theater kiosks or through an online website.

1.3 Definitions, Acronyms, and Abbreviations

- *SRS: Software requirements specification*
- *DBMS: Database Management System*

1.4 References

The references are:

- "Software Requirements Specification document with example" by Ravi Bandakkanava, May 8, 2023 https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database
- "How to Write a Software Requirements Specification (SRS Document)" by Gerhard Krüger and Charles Lane, January 17, 2023 https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document
- "IEEE Recommended Practice for Software Requirements Specifications" by the IEEE Computer Society, October 20, 1998

1.5 Overview

This document contains specific requirements and information for the Theater Ticketing System, including general description, external interface requirements, functional requirements, non-functional requirements, design constraints, and logical database requirements.

2. General Description

2.1 Product Perspective

The Theater Ticketing System will interact with databases of showtimes, available tickets, and user accounts. The application operates independently from other software/hardware which means it will be a standalone web-based application.

2.2 Product Functions

The system will perform many different functions, the main being ticket purchasing and interfacing with databases. For security and management purposes, other functions include blocking bots from buying tickets in high-demand , managing through administrator mode, and collecting customer feedback.

2.3 User Characteristics

Users of the system include customers purchasing theater tickets, software engineers maintaining the system and software, and administrators that manage and control the system.

2.4 General Constraints

Constraints include handling at least 1000 users at once, preventing scalping/reselling and selling unique, non-replicable tickets, supporting multiple languages such as English, Spanish, and Swedish, and following discount policies.

2.5 Assumptions and Dependencies

Assumptions include the availability of hardware and software needed for this ticketing system to work and operate as intended.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

- *The system will have user interfaces for ticket purchasing, account management, and feedback submission.*
- *Language options include English, Spanish, and Swedish*.

3.1.2 Hardware Interfaces

- *For web browsing, the system will interface with standard hardware components.*

3.1.3 Software Interfaces

- *The system will interact with databases for tickets, showtimes, and customer accounts.*

3.1.4 Communications Interfaces

- *The system may use email for ticket delivery and communication.*

3.2 Functional Requirements

3.2.1 Ticket Purchasing

3.2.1.1 Introduction

- *Users can access the system via a digital kiosk in person or online through a web browser, where they will be presented with an interface that displays showtimes and ticket availability for either a deluxe or regular theaters depending on the theater's location.*

3.2.1.2 Inputs

- *The user selects a desired showtime and specifies the number of tickets they would like to purchase. If the theater is a deluxe theater, the user will then be prompted to select what specific seats they would like to claim. To complete the transaction, the user chooses an accepted payment method and whether they would like their ticket to be provided via email or in person at the box office.*

3.2.1.3 Processing

- *The system will process the payment method that the user inputs and check error handling for any issues that may have occurred during the input process.*

3.2.1.4 Outputs

- *The user receives confirmation for their purchase.*
- *The system will ask for feedback by displaying a range of smiley faces for the user to select.*

3.2.1.5 Error Handling

- *Users can purchase tickets for specific showtimes.*
- *A maximum of 20 tickets can be purchased in a single transaction.*
- *Tickets are available for purchase two weeks prior to showtime and up to 10 minutes after the show starts.*
- *Once the purchasing process begins, the system will initiate a timer with a 5 minute countdown. If the user does not complete the transaction within this time frame, their ticket will be abandoned.*

3.2.2 Bot Blocking

- *The system must be secure to protect users and to prevent automated bot purchases. Tickets purchased must be unique and non-replicable (tickets are NFTs).*

3.2.3 Administrator Mode

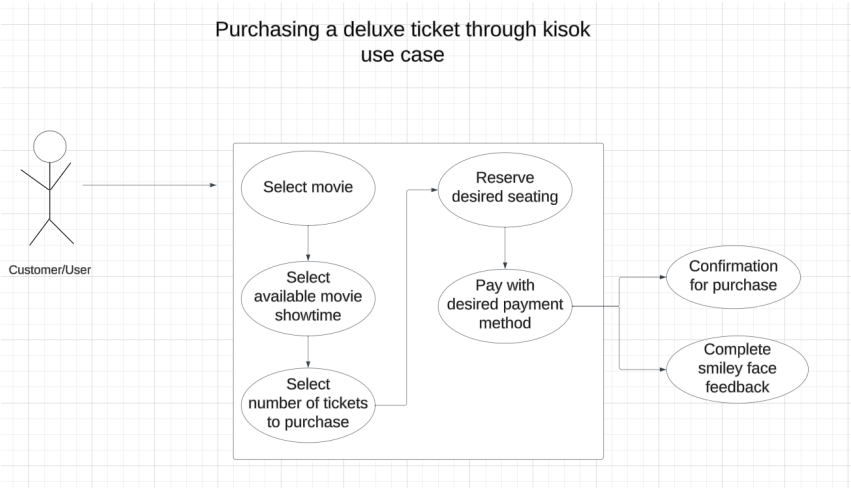
- *Administrators can access a secure mode for managing and controlling the ticketing system such as overriding errors made by users and setting showtimes and theaters.*

3.2.4 Feedback System

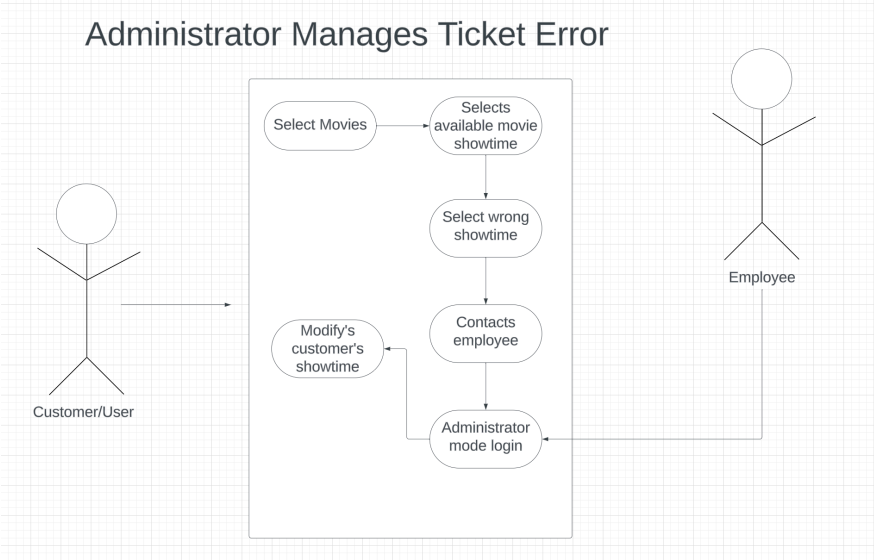
- *Customers can provide feedback on their ticketing experience. After each customer purchases tickets, whether via kiosk in-person or via the online website, smiley faces will be displayed for people to select to represent their evaluation of the theater.*

3.3 Use Cases

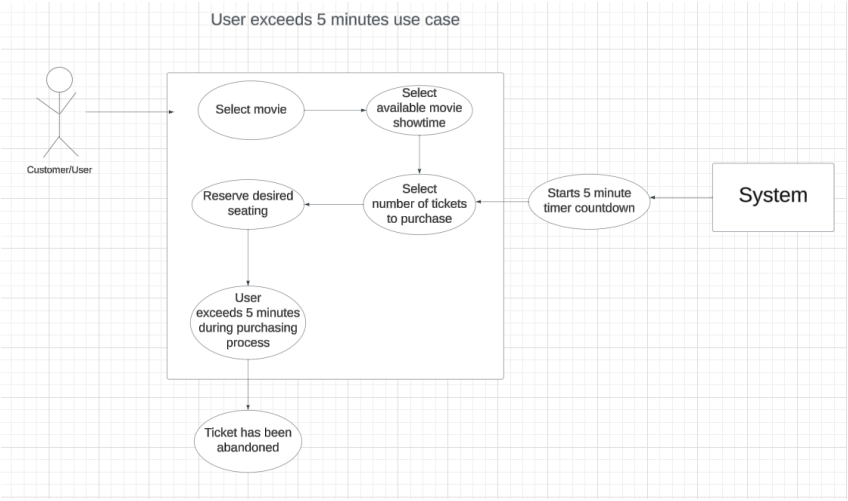
3.3.1 Use Case #1: User purchases deluxe tickets



3.3.2 Use Case #2: Administrator Manages Ticket Error



3.3.3 Use Case #3: User Exceeds 5 Minutes



3.4 Classes / Objects

3.4.1 User Class

3.4.1.1 Attributes

- *User account name and password*
- *Can register loyalty accounts to store payment/purchase history, personal information, and loyalty points*

3.4.1.2 Functions

- *Purchasing tickets by selecting showtime, theater, and number of tickets*
- *Pay with credit card, paypal, bitcoin*
- *Reserve seats*
- *Can return or exchange tickets before showtime*

3.4.2 Administrator Class

3.4.2.1 Attributes

- *Admin account name and password*
- *Access Control*

3.4.2.2 Functions

Overrides any errors made by users
Set showtimes and prices
Change theaters as needed
User management
Security Management
Financial management
Customer/Collaboration communication

3.5 Non-Functional Requirements

3.5.1 Performance: *The system must handle at least 1000 concurrent users efficiently and the system must be easy to use by customers. The system must also be secure and protect customer information. Tickets distributed through the system must be unique and non-replicable.*

3.5.2 Reliability: *In cases where there are a high volume of requests of more than 1000 people for a single showing at once, a queuing system will be used so that customers can wait in line to purchase their tickets according to when they attempted their purchases.*

3.5.3 Usability: *The user interface must be simple, straightforward, and easy to use by customers purchasing tickets. Customers will receive unique, non-replicable ticket(s).*

3.5.4 Security: *The user interface must be secure to prevent data breaches and unapproved access to customers' accounts and information. To protect customers and the potential for unauthorized purchases, only one device can log into a user account at once. Signing into a user account using a different device will log out the previous device utilized.*

3.5.5 Maintainability: T*he system will keep a systemwide daily log of ticket purchases and will have an administrator mode that is available. The administrator mode will override any errors made by customers and will set theaters, showtimes, and tickets available for purchase.*

3.5.6 Portability: *The entire theater ticketing system uses a single database that is divided by individual theaters allowing the system to function smoothly across different systems and individual theaters.*

3.6 Inverse Requirements

The system has various inverse requirements. The user has 5 minutes after starting the ticket purchasing process to finish purchasing their ticket before the ticket is abandoned. The range in which users may purchase tickets is a maximum of two weeks in advance and up to 10 minutes after showtime. Users may purchase up to a maximum of 20 tickets at once. User accounts can be logged into only one account at a time and can purchase using credit card, paypal, or bitcoin. There are also only assigned seats for deluxe theaters.

3.7 Design Constraints

- *The system must be web-based (not app-based) and accessed through standard web browsers.*
- *Ticket validation is limited to the San Diego theater chain (20 different theaters) and Pacific Time Zone.*

3.8 Logical Database Requirements

The system will use a database for storing customer information, user accounts, showtimes, tickets, reviews and critics quotes of the movies, and feedback data.

3.9 Other Requirements

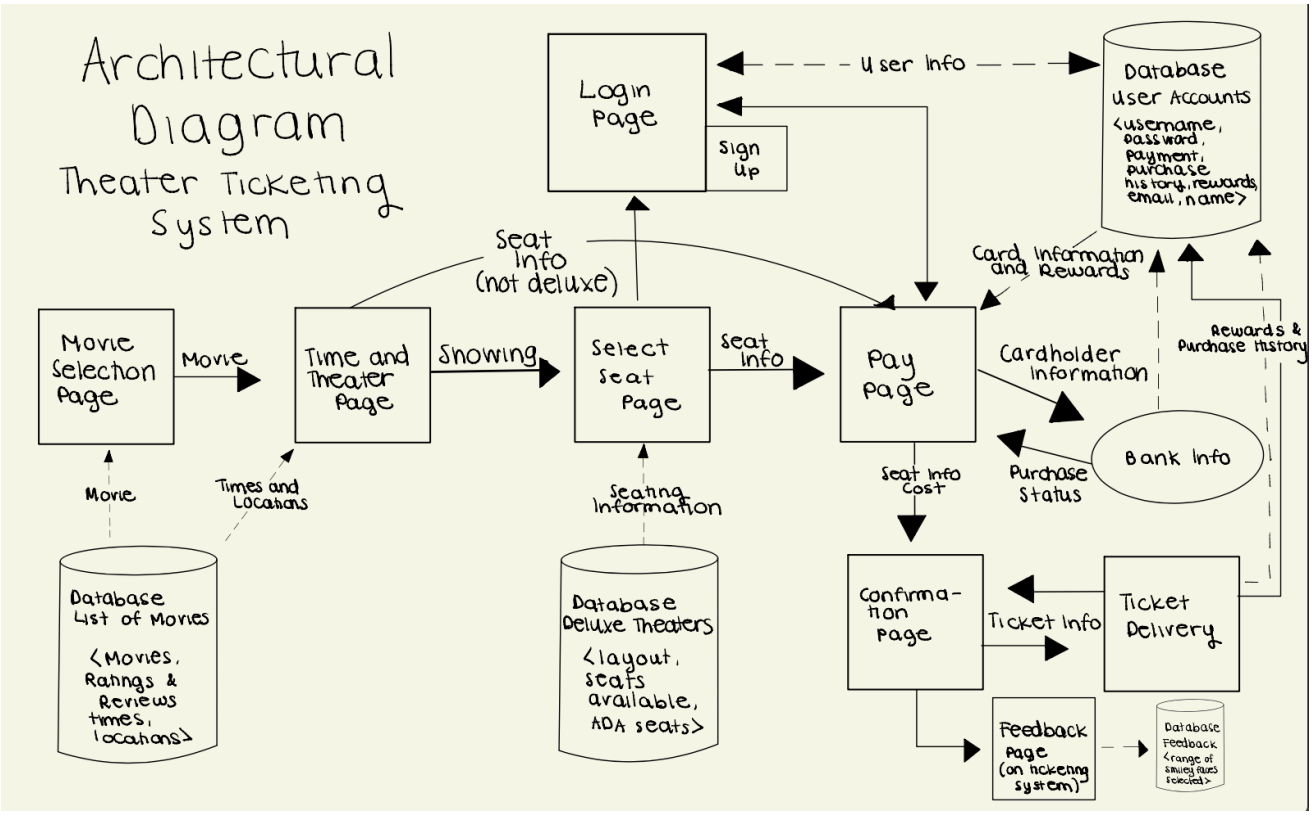
The system must offer discounts for students, veterans, and military during weekdays. The system must also allow users to register for loyalty accounts that store purchase history, personal information, payment information, and loyalty points. Users with accounts may return or exchange their theater tickets before the showing.

4. Software Design Specification

The Software Design Specification describes the software organization, development plan, and architectural design of the theater ticketing system. The user class and administrator class have access to different components of the system. While the user is able to purchase tickets, make an account, and select movies and showings of their choice, the administrator has access to all databases and information submitted on the web application and may make changes to the application. The Architectural Diagram focuses on the components, user interfaces, and connections between interfaces. The UML Diagram focuses on more detailed information about the software structure and behavior.

4.1 Software Architecture Overview

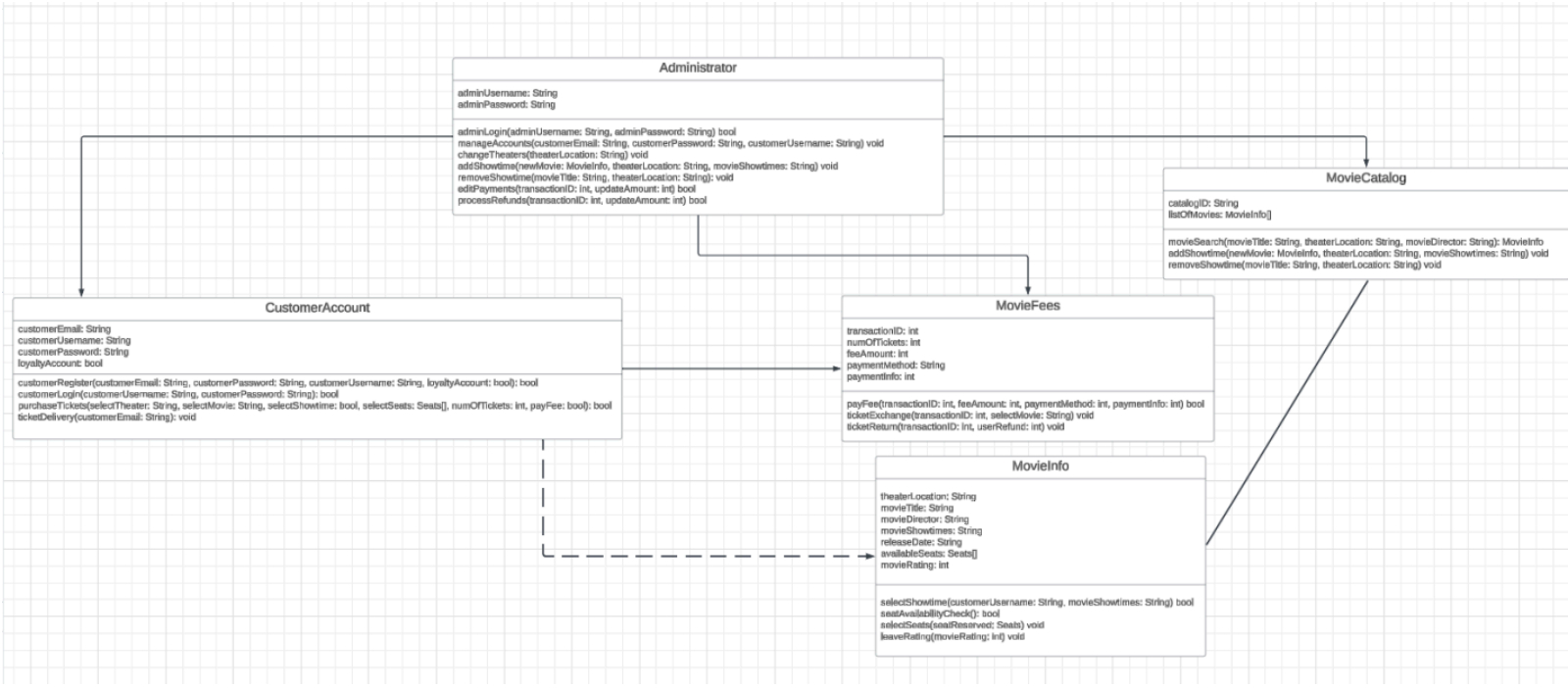
4.1.1 Architectural Diagram



4.1.2 Explanation of the Architectural Diagram

This architectural design serves as a blueprint for the Theater Ticketing System. It outlines the key elements of the system such as its components, interactions, and structure. Its components include databases, functions, and user interfaces. When users first open the web application, they are directed to the "Movie Selection Page" which includes information from the "List of Movies" database. The "List of Movies" database includes information about movies, their ratings and reviews, times, and locations. Users can then select a movie of their choice. This redirects them to the "Time and Theater Page" which contains information about the times and locations from the "List of Movies" database. On this page, users select an available showing with a time and location of the movie of their choice. If the user selects a deluxe theater, they are directed to the "Select Seat Page" because users are able to select their seat(s) if they select a showing in one of the deluxe theaters. The "Select Seat Page" contains seating information from the "Deluxe Theaters" database which includes information about the layout, seats available, and ADA (Americans with Disabilities Act) seating. Once the user selects their seating information, they are redirected to the "Pay Page." If the user does not select a showing from the deluxe theaters, they are not able to select their seats and are directly brought to the "Pay Page" from the "Time and Theater Page." From the "Pay Page," users are given the option to log in with an existing account or sign up with an account for the web application. User information from the log-in/sign-up page is stored in the "User Accounts" database which contains the user's username, password, payment information, purchase history, rewards, email, and name. Once the user is logged in, they are brought to the "Pay Page" to finish their transaction. If the "User Accounts" database does not already include the user's payment and bank information, the user is redirected to the "Bank Info" page to enter their card information. Once the payment information has been entered and the purchase status is confirmed, the user is once again directed to the "Pay Page" to finish their transaction. Once possible rewards, card information, cost details, and theater showing information are confirmed by the user, the user is directed to the "Confirmation Page" where they may open their ticket delivery information on the "Ticket Delivery Page" and/or open the "Feedback Page" where they may rate the ticketing system by picking from a range of smiley faces. The user feedback information is stored in the "Feedback" database which contains a range of smiley faces selected by users. Information from the "Ticket Delivery Page" such as the rewards earned and purchase history is stored in the "User Accounts" database. Finally, the user has successfully purchased their theater tickets. The administrator can access information from all databases and change information such as available movies, times, locations, prices, seatings, rewards, discounts, and more.

4.1.3 UML Diagram



4.1.4 Explanation of UML Class Diagram

The system's architecture is composed of multiple classes that manage the operations of a movie theater which has been displayed in the UML diagram above. The Administrator class is the backend management of the system, consisting of attributes such as adminUsername and adminPassword that allow for secure access control to managing aspects of the system. These administrator privileges are included as operations in the UML Diagram such as being able to manage customer accounts (manageAccounts), theater/movie information (changeTheaters, addShowtime, removeShowtime), and edit payments (editPayments, processRefunds).

The CustomerAccount class allows user interaction with the system, holding important information within attributes such as customerEmail, customerUsername, customerPassword, and a boolean for loyaltyAccount to denote whether the user would like to join the loyalty program. Allowing the system to store the aforementioned information allows for user account creation and sign-ins through the use of the customerRegister and customerLogin operations respectively. Along with this, this class is also responsible for letting users purchase tickets (purchaseTickets) and allows a user to set a delivery preference for receiving their ticket (ticketDelivery).

The MovieCatalog class serves as a directory of movies that are available at a designated theater, each uniquely identified by a catalogID. This class lists movies through an array of MovieInfo instances and includes functionality to search (movieSearch), add (addShowtime), and

remove showtimes (removeShowtime), thereby maintaining an up-to-date and accessible catalog of movies.

MovieFees is responsible for the monetary functions of the theater's transactions. The attributes that encapsulate these functions are transactionID, numOfTickets, feeAmount, paymentMethod, and paymentInfo which are all essentially steps that calculate what a person will owe before purchasing their tickets. The following operations use these attributes to process payments (payFee), allow for ticket exchanges (ticketExchange), and returns (ticketReturn), ensuring that users have an accurate financial experience within the system.

Within the MovieInfo class is a repository of detailed movie information listed as attributes such as theaterLocation, movieTitle, movieDirector, movieShowtimes, releaseDate, availableSeats, and movieRating designed to easily categorize movies by the most important information about them. This class is designed to facilitate customer decisions and engagement through a set of operations that allow them to select showtimes (selectShowtime), see proper seat availability (seatAvailabilityCheck), reserve specific seats (selectSeats), and provide feedback by rating the movie (leaveRating).

4.2 Development Plan and Timeline

As a team, we first analyzed the information provided about the theater ticketing system and described the system in more detail for the first part of this assignment for parts 1-3. We provided the system's purpose, overview, functions, characteristics, constraints, interfaces, functional and non-functional requirements, classes, and use cases. This was completed by February 14, 2024. Next, we completed the "Software Design Specification" in part 4 of the specification. As a team, we used our knowledge of the system to create the system's architectural diagram and UML diagram. These diagrams display the system's components, interactions, and functions in more detail. The architectural diagram displays different user interfaces and databases, and the UML diagram displays the classes, functions, and behaviors.

5. Test Cases

5.1 Verification Test Plan

5.2 Test Case Samples

6. 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 to the SRS's requirements.

6.1 Sequence Diagrams

6.3 Data Flow Diagrams (DFD)

6.2 State-Transition Diagrams (STD)

7. 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.

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