Theater Ticketing System

Software Requirements Specification

Version 1

09/21/2023

Group 3
Emre Yikilmaz
Jose Urrutia
Jake Stonebraker

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

Revision History

Date	Description	Author	Comments
09/21/23	Version 1	Team 3	All sections are not complete

Document Approval

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

Signature	Printed Name	Title	Date
		Software Eng.	
	Dr. Gus Hanna	Instructor, CS 250	09/21/23

Table of Contents

Revision History **3**
Document Approval **3**
1. Introduction **1**
1.1 Purpose 1
1.2 Product 1
1.3 Scope 1
1.4 Overview 1
2. General Description **2**
2.1 Product Functions 2
2.2 User Characteristics 2
2.3 General Constraints 2
2.4 Assumptions and Dependencies 2
3. Specific Requirements **3**
3.1 Functional Requirements 3
3.1.1 Theater Selection 3
3.1.2 Listing of Movies and Showtimes 3
3.1.3 Booking a Ticket Online 3
3.1.4 Discounted Tickets Verification 4
3.1.5 Booking a Ticket with In-Person Kiosk 4
3.1.6 Seat Reservation Timeout 4
3.1.7 Payment Processing 4
3.1.8 Email Notification 5
3.1.9 Refund Process 5
3.1.10 Administrative Access 5
3.1.11 Employee Access 5
3.1.12 Search and Filter 6
3.1.13 User Registration 6
3.1.14 User Profiles 6
3.2 Use Cases 6
3.2.1 Use Case #1 - Customer Wants to Purchase Ticket Through the Theater Kiosk 6
3.2.2 Use Case #2 - Customer Requests a Refund (In-Person) 9
3.2.3 Use Case #3 - Creating User Account 11
3.3 Non-Functional Requirements 11
3.3.1 Performance 13
3.3.2 Reliability 13
3.3.3 Availability 13
3.3.4 Security Requirements 13
3.3.5 Safety Requirements 13
3.3.6 Maintainability 14

3.3.7 Portability..... 14
3.3.8 Usability..... 14
4. Analysis Models External Interface Requirements..... 14
4.1 User Interfaces..... 14
4.2 Hardware Interfaces..... 14
4.3 Software Interfaces..... 14
4.4 Communications Interfaces..... 14
3.7 Design Constraints..... 15
3.8 Logical Database Requirements..... 15
3.9 Other Requirements..... 15
4. Analysis Models..... 15
4.1 Sequence Diagrams..... 15
4.3 Data Flow Diagrams (DFD)..... 15
4.2 State-Transition Diagrams (STD)..... 15
5. Change Management Process..... 15
A. Appendices..... 15
A.1 Appendix 1..... 15
A.2 Appendix 2..... 15

1. Introduction

The introduction of the Software Requirements Specification (SRS) provides an overview of the entire SRS with purpose, product description, scope, and a further overview of other sections of this document. The aim of this document is to analyze and give an in-depth insight of the complete **Theater Ticketing System** by defining gathered requirements in detail.

1.1 Purpose

The purpose of the document is to collect and specify all collected ideas that have come up to define the product, its requirements with respect to consumers. This document describes the functionalities and use cases of the software system and its interface, hardware and software requirements. It should help any designer and developer to adequately design and implement the ideas as specified in this document.

1.2 Product

The product is a ticketing system for cinemas in a specific area. The system will enable consumers to buy tickets through a web browser. This includes selecting movies, showtimes, seats, payment and more. Consumers are the main user group but also an enhanced employee mode and administrator mode with additional features will be introduced. Besides buying tickets as a guest the system will also enable consumers to create an account to log in to get more options like returning tickets or saving personal information.

1.3 Scope

The scope of this document pertains to the ticketing system web browser project for 20 cinemas in the San Diego area. It focuses on the process of purchasing tickets online using the web browser, roles in that process and other dependencies within the system. This includes detailed use cases and the requirements for functional and non-functional features and the resulting steps taken by the assigned roles in these processes.

1.4 Overview

After the introduction a general description of the ticketing system will be discussed, which is section 2 of this document. This is followed by the specific requirements in section 3 which consist of functional requirements, use cases and non-functional requirements. After that section 4 describes the external interface requirements.

2. General Description

This section provides context and general factions for easier understanding of the cinema ticketing system. This includes the current process of cinema ticketing of the client company and challenges that arise from that. The product functions will address these challenges and represent the new process built by the ticketing system created. User characteristics will describe the specified roles within this system.

2.1 Product Functions

The product enables ticket buyers to select movies and their showtimes through the web browser and purchase tickets online. This avoids having to stand in line in the cinema for ticket purchasing. Employees of the cinema can see and manage bookings from ticket buyers. Besides that, cinema administrators can manage user accounts and extract sales reports. Using the online ticketing system the ticket purchasing process becomes easier and more efficient for both ticket buyers and the cinemas. Enhanced features like account registration enable opportunities for loyalty programs and discounts for ticket buyers.

2.2 User Characteristics

This system is mainly created for the user group of ticket buyers. It enables the ticket buyers to conveniently buy tickets online by using the ticketing system through a web browser. Supporting roles for this process are the user groups from the cinema consisting of an employee role and an administrator role. The employee role can access the information processed by the online ticketing system in order to check bookings and access the information about the purchase. The administrator role enables enhanced functionalities like managing user accounts or generating reports for booking history and total sales

2.3 General Constraints

As the client company only operates in the San Diego area with 20 cinemas the ticketing system will be created with the focus on this local area. Other constraints include an already installed web browser together with an existing internet connection to use the online ticketing system. Furthermore an easy to use interface for the ticketing system is needed. the program through forms, a Windows platform or, at bare minimum, a Mac with Access and Excel for Mac installed. Also, it must be constructed in Access, Excel, or another related program that is easily learnable.

2.4 Assumptions and Dependencies

Besides purchasing tickets online before coming to the cinema a tablet kiosk will be implemented to the cinemas. This requires a floor stand and a tablet with internet access where the customers that are already present at the cinema can use the same online ticketing system through the web browser to buy tickets. Every cinema from the client should include this hardware to the facility. The kiosk has a ticket printer available for a physical copy of the ticket.

3. Specific Requirements

3.1 Functional Requirements

3.1.1 Theater Selection

3.1.1.1Description

- The System will only support 20 theaters in the San Diego area working in the Pacific Time Zone
- The system should have one database for all 20 locations but partitioned in a way that has a specific section for each location.
- The system should prompt the user to select a specific theater before displaying movies and showtimes
    - The system should allow the user to search for theaters closest to them by providing a zip code and search radius (5 miles, 10 miles, 25 miles).
    - The system should display a list of nearby theaters with the closest theater being listed first
    - The system should allow the user to click on “Select” next to the desired theater name
    - The Theater name will have its address displayed underneath it

3.1.2 Listing of Movies and Showtimes

3.1.2.1 Description

- The system will have the user select a theater before displaying movies and showtimes
- This system should be able to display a list of current movies and their showtimes.
- This system should be able to provide detailed information for each movie including: Title, Description, Genre, and Run Time.
- This system should display movie showtimes and available dates up to 2 weeks from the current date.

3.1.3 Booking a Ticket Online

3.1.3.1 Description

- This system will require the user to select a specific theater before displaying movie showtimes
- This system should allow users to select a movie and showtime for a specific date up to 2 weeks in advance and 10 minutes after the selected showtime.
- Users should be able to choose the type of ticket (adult or child)
    - The child ticket will be from ages 3-11
- Users should have the option to purchase discounted tickets (See 3.1.4)
- Users should be able to choose their seat from a seating chart
    - This system should allow users to select multiple seats
    - Users are allowed to select up to 20 seats in a single booking
- This system should be able to calculate and display the total price after the user confirms the number of seats they want to purchase.

- This system should allow users to pay for their bookings only after they have confirmed the number of seats they want to purchase.

3.1.4 Discounted Tickets Verification

3.1.4.1 Description

- To get the discount for students and veterans during weekdays the user has to be registered with a profile (see 3.1.13 & 3.1.14).
- Through the profile the user needs to upload proof for the student or veteran status once.
- The discount will be calculated automatically at the payment step.

3.1.5 Booking a Ticket with In-Person Kiosk

3.1.5.1 Description

- Kiosk Interface
    - The kiosk will provide a user-friendly touch screen interface
    - The interface will be easy to navigate for all ages
- The kiosk will contain all of the functional requirements listed in:
    - **Listing of Movies and Showtimes (3.1.2)**
        - Movies and showtimes will be displayed for the specific theater the kiosk is located at
    - **Search and Filter (3.1.12)**
    - **Seat Reservation Timeout (3.1.6)**
    - **Payment Processing (3.1.7)**
    - **Email Notification (3.1.8)**
- User Assistance
    - The system will display a button the user can press if they encounter problems with the booking process
    - Theater staff will be notified if the user requires assistance

3.1.6 Seat Reservation Timeout

3.1.6.1 Description

- This system should implement a mechanism that releases a reserved seat 5 minutes after the payment process has started

3.1.7 Payment Processing

3.1.7.1 Description

- This system should be able to ensure that every payment is encrypted for secure transactions
- This system should be able to accept multiple payment methods including: Visa Debit/Credit, PayPal, Discover, and Mastercard.
- This system should provide the user with a confirmation receipt containing a confirmation number after their payment has been successfully processed.

3.1.8 Email Notification

3.1.8.1 Description

- This system should be able to send the user an email notification after a successful booking
- This system should be able to remind the user of their purchased showtime 24 hours before the scheduled showtime

3.1.9 Refund Process

3.1.9.1 Description

- Refunds can be processed online through the theater ticketing system or in-person at the customer service desk inside the theater the user purchased the ticket from.
- The system will be able to verify that the ticket has not been claimed and is still valid
- The system will be able to verify that the date and showtime on the ticket has not yet passed
- The system will allow the user to request a refund one hour after purchase of the ticket up to 2 hours before the scheduled date and showtime.
- The user will have the option to exchange for a different movie and showtime instead of a direct refund
- The in-person employee handling the refund transaction will access the system using their employee login information.
    - The system will prompt the employee to confirm customer name, confirmation number, and last 4 digits of the original form of payment.
    - The system will interact with the payment gateway to issue the refund to the user's bank within 3-5 business days

3.1.10 Administrative Access

3.1.10.1 Description

- Administrator should have the ability to add, edit, delete, or update movie showtimes
- Administrator should have the ability to manage user accounts in terms of account suspension or deletion
- The administrative access function should have the ability to generate daily, monthly, and yearly reports for booking history and total sales.
- The system will allow the administrator to view individual employee refund transactions (3.1.9).

3.1.11 Employee Access

3.1.11.1 Description

- The system will have an employee access function that allows employees of the theater the ability to search for previous bookings based on customer name and confirmation number
- The system will allow the employee to issue refunds to the user after confirming the user name, confirmation number, and last 4 digits of the payment method.

3.1.12 Search and Filter

3.1.12.1 Description

- This system should provide a search bar that allows users to search for movies by title, genre, or specific actor/actress.
- This system should allow the user to sort movies by popularity and release date.
- This system should allow users to filter movies based on their genre and third-party rating.

3.1.13 User Registration

3.1.13.1 Description

- This system should allow users to create accounts.
    - This system should make it mandatory for users to provide a name and email address and phone number to be added to their personal information.
    - This system will have a username verification implementation that will alert the user if the username they provided is already taken from another user.
    - To finish and confirm the registration an email is sent to the provided email address with a link that needs to be activated for confirmation.
- This system should allow users the option to login using their email/username and password before selecting seats.
- This system should allow users to reset their passwords through the provided email address.

3.1.14 User Profiles

3.1.14.1 Description

- If a user successfully purchased tickets with their account, their booking will be added to their user profile where they will be able to view confirmation number, movie showtime, and date.
    - This system will maintain a user's previous booking history and transactions.
- The user will be able to print their previous booking history
- This system will allow the user to update their profile information (name, email, username, password)
    - Just like the **User Registration** requirement (3.1.13.1), the username verification system will check to see if the user's new username is already in-use by another user

3.2 Use Cases

3.2.1 Use Case #1 - Customer Wants to Purchase Ticket Through the Theater Kiosk

3.2.1.1 Description

Priority: High

<Theater Ticketing System>

Main Actor: Customer
Secondary Actors: Employee (if needed)

Pre-Conditions:

- The customer is present at the theater of their choosing
- The ticketing kiosk is turned on, operational, and connected to the internet
- Movie listings and show-times are current and up-to-date within the database
- Payment Terminal is turned on and functional
- The ticket printer is turned on and functional

Trigger: The customer walks to the ticketing kiosk to purchase their movie ticket

3.2.1.2 Basic Flow:

1. The customer approaches the ticketing kiosk inside the movie theater of their choosing and presses anywhere on the touch-screen user interface to begin.
2. The interface will display a user-friendly list of movies that will contain details including: genre, rating, and showtimes
    - The customer has the option to filter movies based on genre and rating
3. The customer will select a movie of their choosing
4. The interface will display the showtimes of the chosen movie including the amount of seats available for each showtime.
5. The customer will then choose a specific showtime for their chosen movie.
6. A visual seating chart will then display both available and occupied seats for the customer to choose from. The customer has the option to zoom in and out with their fingers for better visibility
7. The customer will then choose how many tickets they want to purchase
    - On the top right of the interface, there will be a display showing the number of adult, child, and discounted tickets they want to purchase (20 MAX TOTAL)
    - Discounted tickets will be verified by the employee stationed at the kiosks
8. The customer will then select from the available seats from the visual seating chart
    - The customer will not be allowed to leave a single seat between two selected seats (an error message will appear if they try to do this)
9. The system will display a number of how many seats left to select to ensure the number of seats selected matches the number of tickets the customer chose to purchase
10. The system will calculate the total price after the customer is done selecting their seats and display the total on the screen.
11. The customer has the option to choose different seats before pressing CONFIRM to bring them to the payment page

12. The customer will be prompted to select their preferred method of payment:
    CREDIT/DEBIT: The customer will insert their chip or tap to pay. That's when the ticketing kiosk will securely process the transaction using the extended interface payment gateway.
13. The system will ensure that the sensitive card information is securely processed and not stored.
14. After successful transaction, the user has two options:
    - To print out a physical, unique, and non-replicable movie ticket which includes the title, showtime, seat number, and barcode for the employee to scan
    - To have their ticket delivered via email which includes the title, showtime, seat number, and QR code for the employee to scan
15. After successful delivery of the movie ticket, the user has two options:
    - To have a physical receipt printed containing their total, number of tickets they purchased, date and time of purchase, and confirmation number
    - To have their receipt delivered via email containing their total, number of tickets they purchased, date and time of purchase, and confirmation number.
16. After successful delivery of the customer's receipt, the system will update the seat availability for the showtime and movie the customer purchased tickets for. The selected seats will then be marked as taken.
17. The customer receives their ticket, either physical or digital, and the transaction will be marked as complete.

3.2.1.3 Alternate Flow # 1: Account Creation/Customer Login

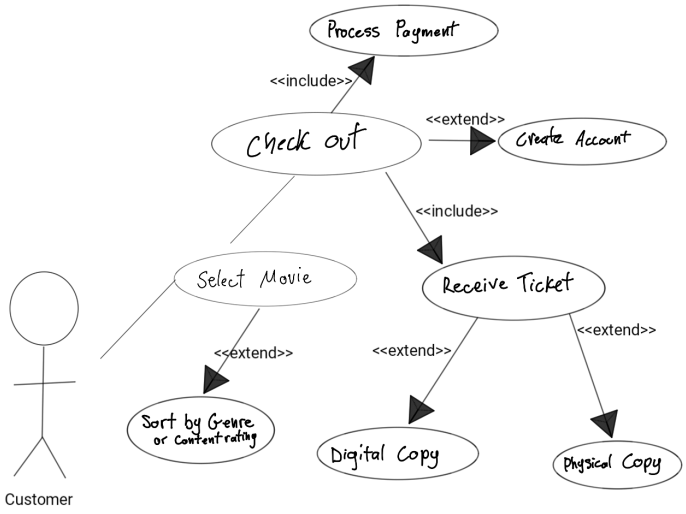
- Anytime during the checkout process, the customer has the ability to either login or create an account which will utilize Functional Requirements **3.1.13** and **3.1.14**. Once the account is created or the customer has successfully logged in, they can proceed through the purchasing process as normal. Once the transaction is marked as completed, the customer transaction history will be recorded to their User Profile.

3.2.1.4 Alternate Flow # 2: Failed Payment

- If at some point during the transaction the payment processing fails due to insufficient funds or the card declining, the system will display an error message that will tell the user to try again using the same form of payment or to choose an alternative payment method.

Post-Conditions:

- The customer receives a printed valid movie ticket OR The customer receives a digital ticket via email with confirmation number attached (Both are non-replicable and unique).
- The database is updated in real time to remove the seat(s) that they purchased
- Their payment was processed successfully



3.2.2 Use Case #2 - Customer Requests a Refund (In-Person)

3.2.2.1 Description

- This use case details the process where the user (customer) wishes to request a refund in-person at the theater they purchased their ticket from. Users will have the option to request refunds both online or in-person.

Priority: High

Main Actor: Customer: The individual who purchased the tickets requesting a refund in-person
Secondary Actors: Employee: The individual working at the customer service counter issuing the refund

Pre-Conditions:

- The customer is allowed to request a refund up to 2 hours before the scheduled showtime labeled on their ticket
- The tickets must be purchased through the theater ticketing system
- The ticketing system must be fully functional and connected to the customer service counter POS

Trigger: The customer approaches the theater customer service desk with valid tickets they wish to refund OR They click the “request refund” option in the ticketing system

3.2.2.2 Basic Flow:

1. 1.The customer tells the customer service representative that they would like to request a refund for tickets they purchased through the theater ticketing system
2. The employee first logs in to the theater ticketing system with their employee access User ID and Password
3. The employee requests the following information from the customer:
    - First and Last Name
    - Date, movie, and showtime
    - Confirmation number for their transaction
4. The employee uses this information to verify the customer's transaction in the theater ticketing system.
5. The employee ensures that the customer refund request is in line with the theater's established refund policy and that the refund is within the allowed refund time period.
6. Once the refund request is deemed valid, the employee will begin the refund process:
    - The employee must first ensure the ticket barcode is canceled and unusable
7. The employee will then initiate the refund transaction:
    - The employee will explain the refund process to the customer and inform them how many business days it will take for the funds to be posted to their bank
    - The employee must verify the total number of tickets being refunded with the customer
    - The employee will verify the total being refunded and must ask the customer if they would like it as theater credit or refunded to the original form of payment
8. If the customer chooses **Original Form of Payment**:
    - The customer will provide the original form of payment and the employee will verify that the last 4 digits of the card match the digits marked on the original transaction
    - Once verified, the customer will either tap or insert their card to the payment terminal for the funds to be refunded to their bank
9. If the customer chooses **Theater Credit**:
    - The employee will transfer the refund amount onto a theater gift card
    - The gift card can be used to purchase tickets from the theater ticketing system either online or through the theater's ticketing kiosk
10. Once approved, the employee will print out a physical refund receipt with the confirmation number for the refund transaction
    - The information on the receipt will include transaction number and the date and time the refund was initiated
11. The employe will also print out a physical receipt for the customer to sign, acknowledging the refund process
    - The employee will keep the receipt for the theater records
12. Once the transaction is completed, the customer will go on their way

3.2.2.3 Alternate Flow # 1: Refund Request Denied

- If the customer's refund request is denied because it does not match the theater's established refund policy, the employee must respectfully inform the customer of the denial while explaining in detail the refund policy established by the theater. They will then offer an alternative of offering theater credit to be put on a theater gift card for the customer to use to purchase tickets online or through the theater ticketing kiosk.

Post-Conditions:

- The theater's ticketing system records the successfully process refund in the history logs for administrators to access
- The ticketing system will update the virtual seating chart and make the seat(s) available for other customers to purchase
- The customer receives either physical or emailed receipt with the confirmation number for their refund
- If the customer linked the original ticket transaction to their user account, the refund transaction will appear in their purchase history.

3.2.3 Use Case #3 - Creating User account

3.2.3.1 Description

- This use case details the process where the user (customer) creates an account to the theater ticketing system website. This is optional and is not required to purchase tickets. The user creates this account in order to save purchases and view past transaction history

Priority: Low

Main Actor: Customer: The individual wanting to make their own user account

Pre-Conditions:

- The customer has a laptop or desktop computer connected to the internet

Trigger: The customer visits the theater ticketing website and will click on "Create an Account"

3.2.3.2 Basic Flow:

1. Once the customer clicks on "Create an Account", the system will ask them to enter the following information:

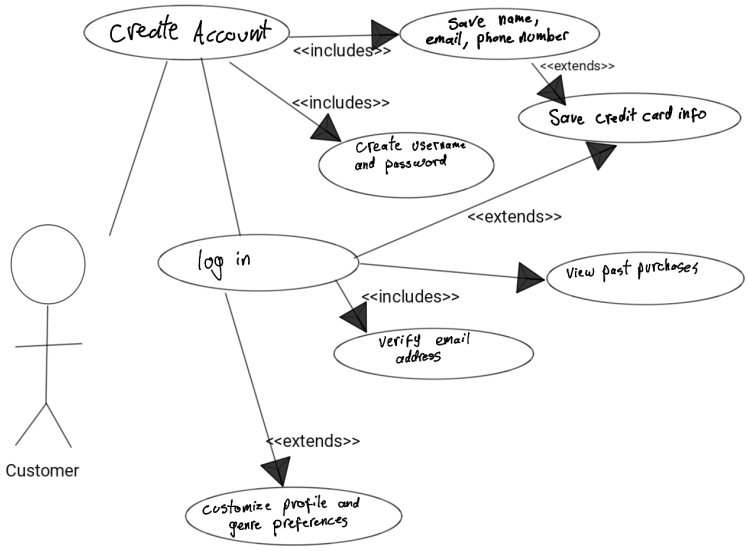
    - Email Address
    - Full Name
    - Phone Number
2. After entering the following information, the system will verify that the Email Address is not yet in use.
3. The customer will create a unique username and password where the system will prompt them to meet the following minimum requirements when creating a password:
    - At least 8 characters long
    - Must contain at least one uppercase letter
    - Must contain at least one lowercase letter
    - Must contain at least one number
    - Most contain at least one of the following special characters:
        - # ! $ @ *
4. The system will then verify that the username is not yet in use by another user
5. Once the username is verified and the password meets the minimum requirements, the system will send the user a verification email where they will click on a link to verify their account.
6. After verifying their account, the system will bring the user to their profile page where they will have the option to add a profile picture, set their movie preferences (genre, content rating, favorite theater), and save their card information that will be securely stored to expedite their purchase experience.

3.2.3.3 Alternate Flow # 1: Password does not meet minimum requirements

- If the user inputs a password that does not meet the minimum requirements, the system will display an error message and will notify the user which password requirement they need to fulfill.

Post-Conditions:

- The customer is registered to the theater ticketing system where they are able to view past purchase history, set their preferences, and optimally save their card information.



3.3 Non-Functional Requirements

3.3.1 Performance

3.3.1.1 Ensure that data for customers is only stored once to save storage space and overridden when necessary for updates to credit card numbers, email/physical addresses, etc.

3.3.2 Reliability

3.3.2.1 System will be regularly and thoroughly tested for reliability by team members
3.3.2.2 System will have 98% uptime with downtime for maintenance done during lowest traffic hours

3.3.3 Availability

3.3.3.1 Queueing system for high volume requests for single showings
3.3.3.2 System will hold seats for customers during the check-out stage for 5 minutes. Ticket will be released to other customers in the queue after 5 minutes.

3.3.4 Security Requirements

3.3.4.1 Each ticket is unique and non-replicable (NFT)
3.3.4.2 Tickets cannot be exchanged or gifted (except to people with verified user accounts)
3.3.4.3 Only one device can concurrently login to a user account
3.3.4.4 Kiosks will allow users to create a user account but will not store any credit card information.

3.3.5 Safety Requirements

3.3.5.1 System will keep systemwide logs of all purchases
3.3.5.2 System will regularly back up data in case of a significant system failure, such as a disk crash, and recover a previous copy of the database and update the system with it. The system

should then continue to recover and copy each new transaction after the recovery date and time according to the log until the time of failure.

3.3.6 Maintainability
3.3.6.1 System must be able to accept regular updates to keep up with ever-changing security requirements, performance upgrades, web browser compatibility and bug fixes.

3.3.7 Portability
3.3.7.1 Ticketing system should be able to run on popular web browsers (Chrome, Edge, Firefox, Safari)
3.3.7.2 Ticketing system should be able to run on tablet for Kiosk purchases.
3.3.7.3 System should read user device resolution and auto-scale to user device

3.3.8 Usability
3.3.8.1 The GUI has to be easy to use and intuitive.
3.3.8.2 The GUI elements like buttons and input fields have to be fully functional and bug free.
3.3.8.3 The transitions between different screens have to be delay-free.
3.3.8.4 The GUI elements and fonts need to be large enough to ensure accessibility to people with slight (sight) disabilities.

4. Analysis Models External Interface Requirements

4.1 User Interfaces

The user interface for the software must be compatible with any browser, such as Internet Explorer, Mozilla or Chrome, through which the user can access the system.
The user interface can be implemented using any tool or software package such as Java Applet, MS Front Page, EJB, etc.

4.2 Hardware Interfaces

Since the ticketing system must run over the web browser, all the hardware required to connect to the internet will be hardware interface for the system. As for e.g. Modem, WAN – LAN, Ethernet Cross-Cable.

The kiosk where the users can buy tickets in the cinema requires a tablet and a floor stand where the web browser can be accessed.

4.3 Software Interfaces

The ticketing system must be connected to the movie database and current ticket stock. Also to the cinema software system and its counter where the ticket is purchased.

4.4 Communications Interfaces

A connection to the respective cinema system has to be ensured.

This is the start of Homework 2

Theater Ticketing System

Software Design Specification

Version 1

October 5th, 2023

Group #3
Jake Stonebraker
Jose Urrutia
Emre Yikilmaz

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

Revision History

Date	Description	Author	Comments
October 5th, 2023	Version 1	Group 3	<First Revision>

Document Approval

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

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

Table of Contents

Revision History **2**
Document Approval **2**
1. Introduction **3**
2. System Overview **4**
3. System Architecture **5**
3.1 UML Class Diagram 5
3.2 Individual Classes of System 5
3.2.1 Theater 5
3.2.2 Person 6
3.2.3 Account 6
3.2.4 Admin 7
3.2.5 Customer 8
3.2.6 Employee 8
3.2.7 Showtime 9
3.2.8 Movie 10
3.2.9 Seat 11
3.2.10 Reservation 12
3.2.11 Ticket 14
3.2.12 TicketType 16
3.2.13 Payment 17
3.2.14 Receipt 18
3.3 Software Architecture 19
3.3.1 Software Architecture Diagram 19
3.3.2 Software Architecture Description 20
4. Development Plan and Timeline and Partitioning of Tasks **24**
4.1 Timeline 24
4.1.1 Phase 1: Project Initiation 25
4.1.2 Phase 2: Design Planning 25
4.1.3 Phase 3: Development 25
4.1.4 Phase 4: Deployment 26
4.1.5 Phase 5: Launch 26

1. Introduction

The introduction of the Software Design Specification (SDS) provides an overview of the SDS with purpose, product description, scope, and the objective of this document. The aim of this software design specification is to describe the software organization and development plan.

1.1 Purpose

The purpose of the document is to define the design for the overall software architecture and implementation of the theater ticketing system . The classes, data structures, together with all major functions and their parameters will be organized and specified. It should help the developers to adequately implement the software components as specified in this document.

1.2 Scope

This software design specification is to be used by our Software Engineers as a template definition of the design and architecture that is to be used in implementing the Theater Ticketing System.

1.3 Objective

The Theater Ticketing System is meant to provide the customer and theater employee the ability to view, select, reserve, purchase, and refund tickets. Customer accounts will also be created (if customer wishes to do so) to remember personal information, preferences, and payment methods.

2. System Overview

2.1 Product Perspective

The product enables users (customer/employee) to view, sort, select, reserve, purchase, and refund tickets for theaters the web browser and purchase tickets online. Employees of the cinema can see and manage bookings from ticket buyers. Besides that, cinema administrators can manage user accounts. Using the online ticketing system the ticket purchasing process becomes easier and more efficient for both ticket buyers and the cinemas. Enhanced features like account registration enable opportunities for loyalty programs and discounts for ticket buyers.

2.2 Product Functions

2.2.1 ***Theater class functions****: displayTheaterDetails()*
2.2.2 ***Person class functions****: n/a*
2.2.3 ***Account class functions****: resetPassword(), validate_Username()*
2.2.4 ***Admin class functions****: addMovie(), addShowtime(), deleteUser()*
2.2.5 ***Customer class functions****: getReservation(), initiateRefund()*
2.2.6 ***Employee class functions****: searchReservation(), initiateRefund()*
2.2.7 ***Showtime class functions****: displayShowtime()*
2.2.8 ***Movie class functions****: sortMovies(), getMovieRating()*
2.2.9 ***Seat class functions****: isOccupied(), occupy(), vacate()*
2.2.10 ***Reservation class functions****: calculatePrice(), addSeat(), removeSeat(), timeLimit(), cancelReservation(), createReservation()*
2.2.11 ***Ticket class functions****: getUser(), getShowTime(), getSeatInfo(), checkTicketType()*
2.2.12 ***Ticket type functions:*** *verifyDiscount(), setPrice()*
2.2.13 ***Payment class functions****: wipeCardInfo(), processPayment(), refund()*
2.2.14 ***Receipt class functions****: generateReceipt()*

2.3 User Characteristics

The intended user is any customer looking to reserve and purchase a ticket online or at a kiosk at any of the theater locations. Employees will be able to make reservations and purchases on customer's behalf when necessary.

2.4 General Constraints

This application will only run on a system that supports a GNU C++ compiler. Is meant to be run on the three most popular web browsers: Google Chrome, Firefox, Safari.

2.5 Assumptions and Dependencies

The user will have basic knowledge of how to navigate a website using a mainstream web browser, have a valid email address, locate their desired theater, select the movie of interest, enter personal and payment information, and present their ticket at the movie theater.

3. System Architecture

3.1 UML Class Diagram

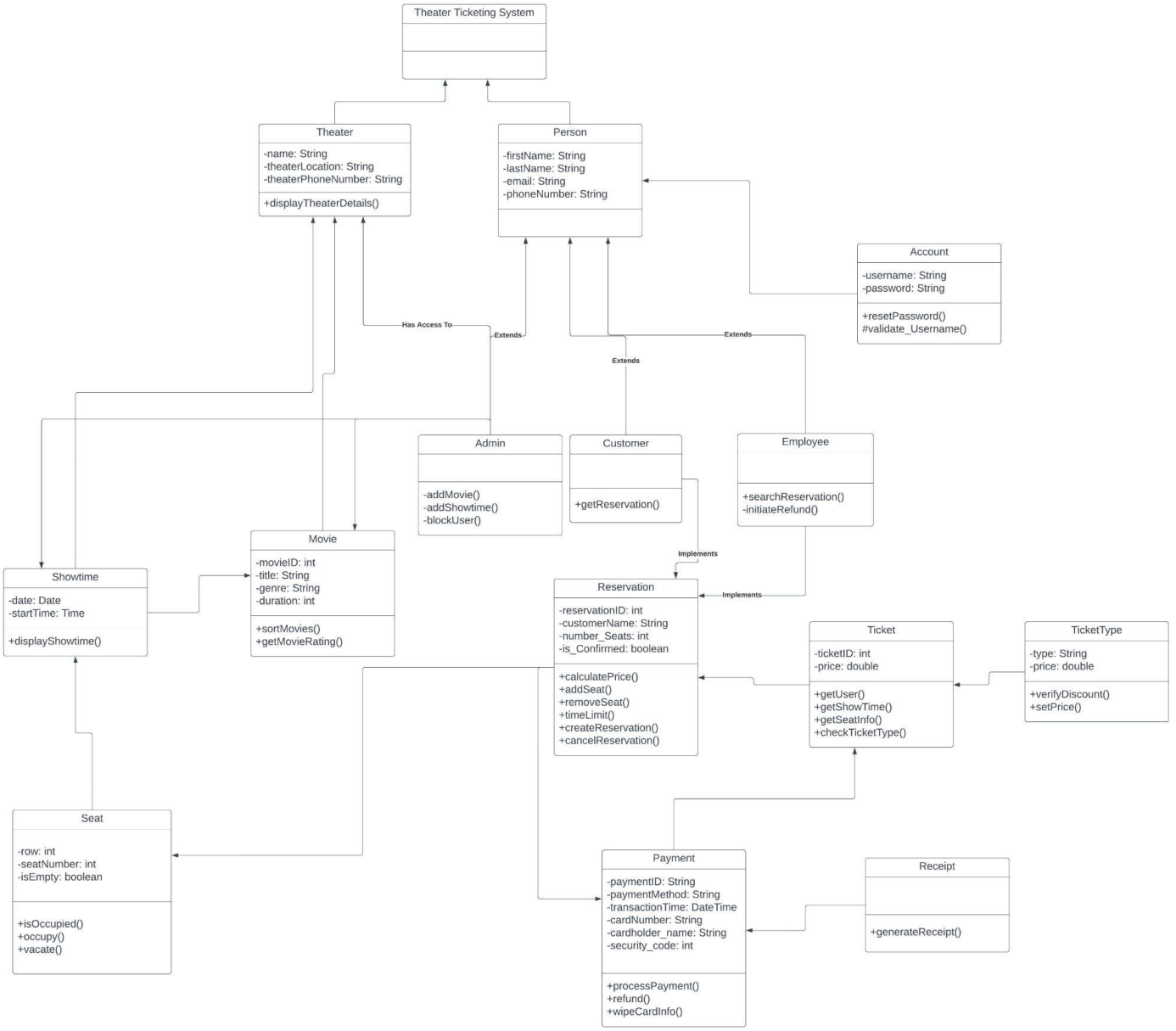


Figure 1: UML Class Diagram containing the class objects and their relationships

3.2 Individual Classes of System

3.2.1 Theater

the Theater class is will be the object class that can create instances of the individual theaters within the theater ticketing system

3.2.1.1 Attributes of Theater class

3.2.1.1.1 theaterName
the name of the theater will be held in data type: String

3.2.1.1.2 theaterLocation
the address of the theater will be held in data type: String

3.2.1.1.3 theaterPhoneNumber
the phone number of the specific theater will be held in data type: String

3.2.1.2 Functions available in Theater class

3.2.1.2.1 displayTheaterDetails()

this function will serve to display the attributes of the specific theater

Arguments:
this function will contain no arguments as it will directly handle the attributes within the class

Return Type:
this function will have a void return type as it will simply display the details of the theater in a neat manner

3.2.2 Person
the Person class will be the object class that will create instances of each individual actor within the Theater Ticketing System

3.2.2.1 Attributes of Person class

3.2.2.1.1 firstName
the first name of the person will be held in data type: String

3.2.2.1.2 lastName
the last name of the person will be held in data type: String

3.2.2.1.3 email
the email of the person will be held in data type: String

3.2.2.1.3 phoneNumber
the phone number of the person will be held in data type: String

3.2.3 Account
the Account class will have an associative relationship with the Person class as each instance of the person class will have an account corresponding to the type of "Person"

3.2.3.1 Attributes of Account class

3.2.3.1.1 username

the username of the instance of the person class will be held in data type: String

3.2.3.2.1 password

the password of the instance of the person class will be held in data type: String

3.2.3.2 Functions available in Account class

3.2.3.2.1 resetPassword()

this function will allow the instance of the person class (whether a customer, admin, or employee) to reset their password when they have forgotten it or wish to change it for security reasons

Arguments:

-new_password (str): the new password for the user

Return Type:

-boolean: Will return true if the password reset is successful, false if not successful

3.2.3.2.2 validate_Username()

this function will check whether or not the username given by the user is taken by another user

Arguments:

there are no arguments required for this function as it will directly deal with the username attribute of this class

Return Type:

-boolean: will return true if username is available, false if username is not available

3.2.4 Admin

the Admin class will be the specific object class that creates instances of each individual that has administrator privileges in the Theater Ticketing System. They will have access to the Showtime and Movie classes in order to add movies and showtimes for each Theater object

3.2.4.1 Attributes of Admin class

the admin class will inherit the attributes of the Person class

3.2.4.2 Functions available in Admin class

3.2.4.2.1 addMovie()

this function will allow the admin to create the instance of a Movie Object that will be added to the list of movies for the specific Theater object they want to add it to

Arguments:

this function has no arguments as the admin class will have access to the movie class and its attributes

Return Type:

-void: the function will return nothing

3.2.4.2.2 addShowtime()

this function will allow the admin to create the instance of a Showtime Object that will be added to the list of showtimes for the specific Movie object they want to add it to

Arguments:

this function has no arguments as the admin class will have access to the Showtime class and its attributes

Return Type:

-void: the function will return nothing

3.2.4.2.3 deleteUser()

this function will allow the admin to delete Customer Class instances in the person database

Arguments:

this function has no arguments as the admin class will have access to the person class and its attributes

Return Type:

-void: the function will return nothing

3.2.5 Customer

the Customer class will be the specific Object class that will create instances of each individual wanting to purchase tickets using the Theater Ticketing System

3.2.5.1 Attributes of Customer class

the Customer class will inherit the attributes of the Person class

3.2.5.2 Functions available in Customer class

3.2.5.2.1 getReservation()

this function will allow the Customer to search for their existing reservation

Arguments:

-reservationID (int): This function will use the reservation ID to use in a search algorithm in the Theater Ticketing System database

Return Type:

-boolean: the function will return true if the reservation is found or false if it is not found.

3.2.6 Employee

3.2.6.1 Attributes of Employee class

the Customer class will inherit the attributes of the Person class

3.2.6.2 Functions available in Employee class

3.2.6.2.1 searchReservation()

this function will allow the Employee to search for a customer's reservation in order to print tickets or initiate a refund.

Arguments:

-reservationID (int): This function will use the reservation ID to use in a search algorithm in the Theater Ticketing System database

Return Type:

-boolean: the function will return true if the reservation is found or false if it is not found.

3.2.6.2.2 initiateRefund()

this function will allow the Employee to initiate a refund to the Customer if a reservation is found and it abides by the refund policy as described in the functional requirements. If the argument captured in the parameters is true, the function will initiate the refund

Arguments:

-reservationFound (bool): This function will use the reservation ID to use in a search algorithm in the Theater Ticketing System database and capture the boolean from the searchReservation() function within the same Employee Class

Return Type:

-void: the function will return nothing.

3.2.7 Showtime

an instance of the Showtime class will be created by the administrator to be added under the movie database for the specific instance of the Theater class. It will contain the date and start time for the specific movie. It will also have access to the attributes of the Movie class that will be displayed to the user

3.2.7.1 Attributes of Showtime class

3.2.7.1.1 date

the date will contain the date of the instance of the Showtime object using data type: Date

3.2.7.1.2 startTime

the startTime will be initialized with the data type: Time

3.2.7.2 Functions available in Showtime class

3.2.7.2.1 displayShowtime()

this function will allow the Employee to initiate a refund to the Customer if a reservation is found and it abides by the refund policy as described in the functional requirements. If the argument captured in the parameters is true, the function will initiate the refund

Arguments:

-reservationFound (bool): This function will use the reservation ID to use in a search algorithm in the Theater Ticketing System database and capture the boolean from the searchReservation() function within the same Employee Class

Return Type:

-void: the function will return nothing.

3.2.8 Movie

the Movie class will hold the information necessary for the primary actor to view and search for the specific movie they want to see

3.2.8.1 Attributes of Movie class

3.2.8.1.1 movieID

each movie instance will have a unique ID that will have a data type: int

3.2.8.1.2 title

each movie instance will have a title that will use the data type: String

3.2.8.1.3 genre

each movie instance will have a genre classifier that will use the data type: String

3.2.8.1.4 duration

each movie instance will have a run time that will be in minutes using the data type: int

3.2.8.2 Functions available in Movie class

3.2.8.2.1 sortMovies()

this function will allow the user to filter movies based on genre or movie rating

Arguments:

-this function has no parameters as the logic inside the function will determine whether the movie will be filtered by genre or movie rating, which will call the getMovieRating() function within the same class.

Return Type:

-void: the function will return nothing.

3.2.8.2.2 getMovieRating()

this function will interact with a third-party website such as IMDB or Rotten Tomatoes to get information on a specific movie's rating

Arguments:

-title (Movie Object): This function will use the title from an instance of a Movie object to search for the movie rating from the third-party website

Return Type:

-the function will return the rating of the movie which will be displayed with a data type: String

3.2.9 Seat

the Seat class is responsible for creating an instance for each individual seat for a specific Showtime instance. It will interact with the Reservation class for when either a customer or employee wants create a reservation for a specific movie

3.2.9.1 Attributes of Seat class

3.2.9.1.1 row

the row for the specific seat will be initialized with the data type: int

3.2.9.1.2 seatNumber

the actual seat number will be initialized with the data type: int

3.2.9.1.3 isEmpty

this attribute will flag whether or not the seat is empty with data type: boolean

3.2.9.2 Functions available in Seat class

3.2.9.2.1 isOccupied()

this function will interact with the seating chart interface and will display the seat in black if it is already purchased or red if it is still available to purchase

Arguments:

-this function has no parameters since it will interact directly with the attributes within the class

Return Type:

-boolean: the function will return true if the seat is occupied or false if the seat is empty

3.2.9.2.2 occupy()

this function will interact with the seating chart interface to mark that is empty as occupied after a reservation is confirmed with payment confirmation

Arguments:

-isOccupied (bool): This function will call the isOccupied() function within the class to see if the seat is empty and ready for purchase.

-reservationConfirmed(bool): this function will also call the reservationConfirmed() function to make sure that payment was processed before marking the seat as occupied

Return Type:

-boolean: the function will return true if occupied successfully and false, if otherwise

3.2.9.2.3 vacate()

this function will interact with the seating chart interface and will turn an occupied seat back to red (previously black) after a successful reservation cancellation

Arguments:

-this function has no arguments as it will be called within the cancelReservation() method within the Reservation class

Return Type:

-void: the function will return nothing.

3.2.10 Reservation

the Reservation class will hold the attributes and functions of each single unique reservation instance created by either the Customer or the Employee. It will include a unique numerical ID, the name of the customer, and the number of seats they purchased

3.2.10.1 Attributes of Reservation class

3.2.10.1.1 reservationID

the reservation ID will be unique to the customer using a random number generator of type: int

3.2.10.1.2 customerName

the name of the Purchaser will be initialized by user input with data type: String

3.2.10.1.3 is_Confirmed

this will be a flag for when the reservation is confirmed: type boolean

3.2.10.1.4 number_seats

this will represent the number of seats the user intends to purchase with data type: int

3.2.10.2 Functions available in Reservation class

3.2.10.2.1 calculatePrice()

this function will calculate the total price of the reservation based on the number of seats chosen for each Ticket Type. It will also include the sales tax and include a reservation fee calculated in the total

Arguments:

-this function will have no parameters as it will interact directly with the variables within the class itself.

Return Type:

-double: the function will a floating point double containing the total price of the reservation

3.2.10.2.2 addSeat()

this function will see what Ticket Type the user wishes to purchase and will increment number_seats variable by one after each time the program takes user input of a mouse click on a plus sign on the website interface

Arguments:

-this function will take a user input of a single mouse click on the plus sign located on the website interface

Return Type:

-void: the function will return nothing.

3.2.10.2.3 removeSeat()

this function will see what Ticket Type the user wishes to purchase and will decrement number_seats variable by one after each time the program takes user input of a mouse click on a minus sign on the website interface

Arguments:

-this function will take a user input of a single mouse click on the minus sign located on the website interface

Return Type:

-void: the function will return nothing.

3.2.10.2.4 timeLimit()

this function will start a 10 minute timer once the user chooses to create a reservation. It will kick them out of the reservation window once time runs out

Arguments:

-This function contains no parameters

Return Type:

-void: the function will return nothing.

3.2.10.2.5 cancelReservation()

this function will allow the user to cancel a reservation by providing their unique reservation ID. The function will search the reservation database and will return a boolean notifying the user whether or not their reservation was found. The function will interact with the refund method to initiate the refund process

Arguments:

-reservationID (int): This function will use the reservation ID to use in a search algorithm in the Theater Ticketing System database.

Return Type:

-boolean: the function will return true if reservation is found and refund is processed

3.2.10.2.6 createReservation()

this function will initiate the process of creating a new reservation prompted by either the Customer or the Employee. It will interact with the Seat Class for them to be able to choose a Movie and Showtime. Once chosen, the function will interact with the functions within the Reservation class to add or remove seats and to calculate the total price. Once total price is calculated, the Payment class will be called where the processPayment() function will be invoked to process the payment.

Arguments:

-this function takes no arguments

Return Type:

void: the function will return nothing. It will change the is_Confirmed variable to true which will trigger the system to prompt the user if they want an email or print receipt

3.2.11 Ticket

The instance of a Ticket class will be generated once a reservation has been made in the Reservation class. The amount of instances created will depend on the number of seats purchased.

3.2.11.1 Attributes of Ticket class

3.2.11.1.1 ticketID

Each ticket will have a unique ID which has the data type: int

3.2.11.1.2 price

the price of each ticket displayed will be determined by the Ticket Type. This variable will have the data type: double

3.2.11.2 Functions available in Ticket class

3.2.11.2.1 getUser()

this function will retrieve the customer name from the Customer class to be displayed on the ticket

Arguments:

-this function has no arguments

Return Type:

-void: the function will return nothing.

3.2.11.2.2 getShowTime()

this function will retrieve the Showtime information from the Showtime class to be displayed on the ticket

Arguments:

-this function has no arguments

Return Type:

-void: the function will return nothing.

3.2.11.2.3 getSeatInfo()

this function will retrieve the seat information from the Seat class to be displayed on the ticket

Arguments:

-this function has no arguments

Return Type:

-void: the function will return nothing.

3.2.11.2.4 checkTicketType()

this function will verify the ticket type: adult, child, student, or senior. This will invoke the verifyDiscount() method in the TicketType subclass. If the ticket type is verified, it will apply the discount

Arguments:

This function has no arguments

Return Type:

-boolean: the function will return true if the ticket type is a discounted type and verified correctly

3.2.12 TicketType

the TicketType class is a subclass of the Ticket Class that will create an instance based on the type of ticket the user chooses. If a discounted ticket is chosen, the verifyDiscount() function will be called

3.2.12.1 Attributes of TicketType class

3.2.12.1.1 type
it could either be Adult, Child, Student, Military, or Senior. Will be data type: String

3.2.12.1.2 price
the price will be based on the ticket type. Will invoke setPrice() method that returns type: double

3.2.12.2 Functions available in TicketType class

3.2.12.2.1 verifyDiscount()

this function will interact with a third-party verification website such as ID.me to verify student, military, or senior status. If the verification fails, the user will be notified.

Arguments:
This function has no arguments

Return Type:
-boolean: the function will return true if the verification was returned as successful.

3.2.12.2.2 setPrice()

this function will set the price based on the type of ticket the user chooses. If it's either adult or child ticket type, no verification is needed and the price will be set accordingly. If the user requests a discounted ticket type, the verifyDiscount() method will be invoked and the discounted price will be set based on the discounted ticket type. if the method returns false, the adult ticket price will be set by default.

Arguments:
This function has no arguments

Return Type:
-void: this function returns nothing.

3.2.13 Payment
The Payment Class will create an instance of the class whenever a reservation is created or a refund is requested in terms of canceling the reservation. The functions within this class will interact with a external payment gateway connected to the banking systems that will process payments and refunds. Security is a top priority to protect the information of the user

3.2.13.1 Attributes of Payment class

3.2.13.1.1 paymentID
each transaction will have a unique ID that will be logged and tracked using the data type: int. It will utilize a random number generator to create the unique ID

3.2.13.1.2 paymentMethod

payment method type will be initialized based on what type of card they use for the transaction. The name of the card will be of data type: String

3.2.13.1.3 transactionTime

date and time of transaction will be recorded with type: DateTime

3.2.13.1.4 cardNumber

cardNumber will be sent to payment gateway with type: String

3.2.13.1.5 cardholder_name

will be sent to payment gateway with type: String

3.2.13.1.6 security_code

will be sent to payment gateway with type: int

3.2.13.2 Functions available in Payment class

3.2.13.2.1 wipeCardInfo()

this function will set all of the attributes to null after the transaction is successfully completed, the attributes in the class will be set to null. The information will be recorded in the database beforehand. The Payment object can then be deleted from memory to protect user information.

Arguments:

-the function contains no arguments

Return Type:

-void: the function will return nothing.

3.2.13.2.2 processPayment()

this function is responsible for handling the payment transaction when a customer makes a purchase in a theater ticketing system. It will interact with the payment gateway to communicate with the banks to process the transaction

Arguments:

-customer (Customer): the customer making the payment

-totalPrice (double): the total amount to be charged for the ticket booking including taxes

-paymentMethod (String): payment method used for the transaction

-paymentDetails (String, String, int): the cardholder name, card number, and security code

Return Type:

-boolean: returns true if the payment was successful, false otherwise

3.2.13.2.3 refund()

this function is responsible for processing refunds for a payment transaction. It will interact with the payment gateway that will communicate with the banks to process the refund.

Arguments:

-Transaction ID (String): the unique identifier of the payment transaction

-refundAmount (double): the amount to be refunded to the customer

Return Type:

-boolean: the function will return true if refund is successful, false otherwise

3.2.14 Receipt

3.2.14.1 Attributes of Receipt class

the Receipt class will inherit all of the attributes of the Payment and will also have access to the variables in the Ticket class containing reservation ID

3.2.14.2 Functions available in Receipt class

3.2.14.2.1 generateReceipt()

this function will generate either a printed or emailed receipt based off of the user input. If asking for an email receipt, it will prompt the user to enter their email address to be sent the receipt with all of the information. Otherwise, it will trigger the print function on their desktop to print a physical receipt

Arguments:

-paymentSuccessful(boolean): if true, will send a receipt, if false, will return a message saying payment was not processed correctly

-deliveryType (String): this function will take in an argument of type String that the user will input as being either an email or printed receipt.

Return Type:

-void: the function will return nothing.

3.3 Software Architecture

3.3.1 Software Architecture Diagram

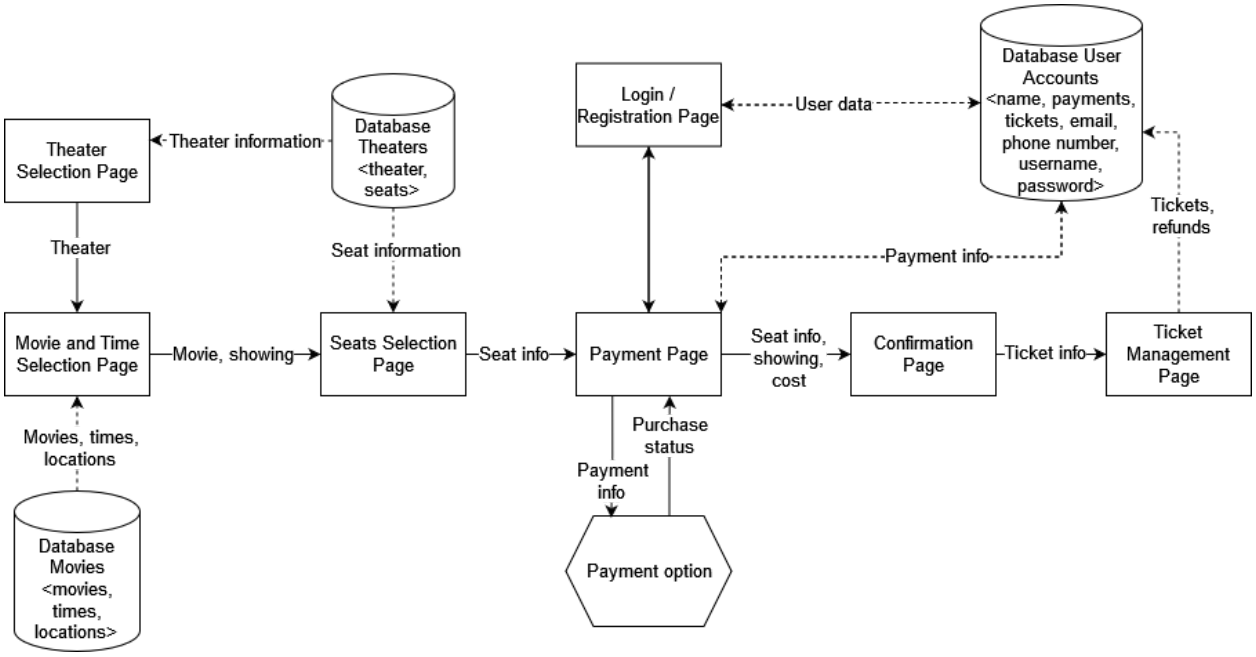


Figure 2: Software Architecture Diagram of the Theater Ticketing System including components and connectors

3.3.2 Software Architecture Description

The next section will provide a complete explanation of the entire SWA diagram, including an overview of components and their connectors.

3.3.2.1 Theater Selection Page

Description: On this page the user is selecting one of the theaters of the client from the San Diego Area.

Functionality: Selecting the theater enables the user to go further in the process of ticket buying. Once a theater is selected it will be the default theater shown in the drop-down selection.

Connectors: Theater Selection Page gets its theater information from the theater database which stores information about the theaters that can be selected. Once the theater is selected the user is enabled to select the movies and showtimes that are available in the theater.

3.3.2.2 Movie and Time Selection Page

Description: On this page the movie and showtime for the selected movie will be selected.

Functionality: Every movie currently available in the selected theater has different showtimes available. Showtimes that are booked out can't be selected and should be using a disabled secondary button.

Connectors: The movie database provides the Movie and Time Selection Page with data about available movies, the showtimes and the location where the movie is shown inside the theater that was selected on the Theater Selection Page.

3.3.2.3 Seats Selection Page

Description: Seats for the selected movie and showtime can be selected on this page

Functionality: This page allows the user to select how many seats he wants to purchase. The default value is 1 and can be increased and decreased with a plus and minus sign UI element or through clicking and writing an integer between 1 and 20 into the input field where the default value of 1 is shown. The maximum number of seats available per purchase are limited to 20.

Connectors: With the input from the Movie and Time Selection Page with the movie and showtime and the seat information from the Theater Database the seat selection is enabled. The seat information from this page will be directed to the next page for the payment.

3.3.2.4 Payment Page

Description: On the Payment Page the user can see his previous selections in the process and pay for the selected tickets.

Functionality: The Payment Page enables the user to see the selected theater, movie, location, showtime and seat information. Different payment options are available to choose from. Once a payment is confirmed the payment provider checks the validity of the payment and returns the payment status. If there is any complication the payment will be declined with an error message, if everything is correct the payment will be confirmed. Besides the option of proceeding as a guest the user can register or log in with an account.

Connectors: Through the seat selection all information about the purchase is displayed. The Registration / Login Page is connected to the Payment page as well. The payment option gets the payment request with the payment information and returns the payment status as confirmed or declined. The purchase information with the costs and tickets will be forwarded to the Confirmation Page. If an user account exists the payment info will be stored in the User Account Database.

3.3.2.5 Login / Registration Page

Description: The Login / Registration Page enables the user to sign up or log in with an user account.

Functionality: When proceeding with the payment process there is the option to sign up or log in with a user account. The user data is stored in the User Account Database. Name, username and password are required fields but phone number is an optional input. The username will be an email address and the password has to be at least eight characters including minimum one number, uppercase letter, lowercase letter and a special character.

Connectors: There is a connection between the Login / Registration page and the Payment Page. When logging in with the user account the previously used payment option will be set as a default. All the user data from the registration will be stored in the User Account Database.

3.3.2.6 Confirmation Page

Description: After a successful purchase of tickets the Confirmation Page will be displayed summing up the purchase information.

Functionality: The Confirmation Page will display the theater, movie, showtime, seats and the location inside the selected theater where the movie is shown. Also the seat information and the costs will be shown on this page.

Connectors: The information of the purchase and tickets comes from the connection to the Payment Page. The ticket information will be passed on to the Ticket Management Page.

3.3.2.7 Ticket Management Page

Description: The Ticket Management Page will display the purchased tickets with the information about the purchased tickets. Also the option of refunding will be found on this page.

Functionality: This page enables the user to view his tickets with the ticket information (seats, movie, showtime and location) and can be used to show the purchased tickets at the theater for entry. Also there is a button that enables the user to refund the tickets.

Connectors: The purchase information is retrieved from the Confirmation Page. The connection to the User Account Database facilitates the process of refunding as the ticket information from the user is stored in the database as well.

3.3.2.8 Payment Option

Description: After the request for the payment the payment provider checks the payments validity and returns a purchase status.

Functionality: The request for the payment from the Payment Page will be checked from the payment provider and will return a confirmation or decline the payment. The status of confirmation or declinement will be returned to the Payment Page with a corresponding message on the UI of the Payment Page.

Connectors: The Payment Option is only connected to the Payment Page getting the payment request and returning the payment status from the payment provider.

3.3.2.9 Database Theaters

Description: The Database for theaters stores information about theaters of the client together with seat information.

Functionality: The theater database provides data about the theaters for the Theater Selection Page and seat information to the Seats Selection Page.

Connectors: see 3.3.2.9 Functionality.

3.3.2.10 Database Movies

Description: The movie database stored data about the movies, showtimes and locations (halls) where the movie is shown.

Functionality: The movie database provides the Movie and Time Selection Page with data about availability of movies in the selected theater together with showtimes for each movie and locations inside the theater where the movie is shown.

Connectors: see 3.3.2.10 Functionality.

3.3.2.11 Database User Accounts

Description: The user accounts database stores information about user accounts and related activities such as payments.

Functionality: From the registration page the user data like name, username, password, email address and phone number is retrieved. Payment information comes from the connection to the Payment Page. Previously stored payment information can also be retrieved from the database to the payment page when the user logs into the user account. Refunds that are requested in the Ticket Management Page are checked with the information in the user account database.

Connectors: see 3.3.2.11 Functionality.

4. Development Plan and Timeline and Partitioning of Tasks

4.1 Timeline

4.1.1 Phase 1: Project Initiation (Project Manager, Business Analyst)

Timeline: Weeks 1-2

1. **Project Start (Week 1)**

- Define scope and objectives
- Hire project manager and create development team
- Obtain deliverables
- Resource Allocation

2. **Requirements Gathering (1-2)**
- Document requirements for theater ticketing system
- Create features and functionalities

4.1.2 Phase 2: Design Planning (System Architect, Database Designer, UI/UX Designer)

Timeline: Weeks 3-6

1. **Software Architecture (3-4)**
- Create software architecture design
- Create software architecture diagram

2. **Database (4-6)**
- Define UML Class Design
- Create UML Class Diagram
- Plan for securing data

3. **User Interface (4-6)**
- Plan for visual interface and layout
- Gather feedback on UI

4.1.3 Phase 3: Development (Frontend/Backend Developers, Quality Assurance/Test Engineers, DevOps Engineers)

Timeline: Weeks 7-23

1.**Frontend Development (Weeks 7-13)**
- Develop the user interface based on UI designs.
- Implement user authentication and authorization.

2. **Backend Development (Weeks 13-19)**
- Build the server-side application.

- Implement payment processing.

3. **Database Implementation (Weeks 19-21)**
- Create the database using the planned design
- Implement data storage and retrieval functionalities.

4. **Testing and Quality Assurance (Weeks 21-23)**
- Conduct testing
- Fix bugs experienced during testing
- Make sure the system is secure

4.1.4 Phase 4: Deployment (Training Specialist, Support & Maintenance Team, Marketing & Promotion team)

Timeline: Weeks 24-27

1. **Deployment and System Integration (Weeks 24-26)**
- Deploy the ticketing system to production servers
- Integrate with third-party services (IMDB, Rotten Tomatoes)
- Initiate performance testing

2. **Training and Documentation (Week 26)**
- Train the theater staff on how to use the system
- Create user manuals and documentation

3. **Soft Launch and Feedback Gathering (Week 27)**
- Perform a soft launch with audience being stockholders/family
- Gather feedback

4.1.5 Phase 5: Launch (Support/Maintenance Team, Marketing & Promotion Team, Product Manager)

Timeline: Weeks 27-30

1. **Full-Scale Launch (Week 27)**
- Launch the Theater Ticketing System to the general public
- Promote the new ticketing system through advertising

2. **Monitoring and Maintenance (Weeks 28-30)**
- Regularly monitor the system performance and keep obtaining feedback
- Address any issues and release regular system updates (security, bug fixes, etc.)
- Plan for weekly to bi-weekly maintenance and support for the system