<Movie Theater Ticketing System>

Software Requirements Specification

<Version>

<9/18/2025>

<Group 15>
<Christian De Leon, Jacob Tabalon, Max Caguioa>

Prepared for
CS 250- Introduction to Software Systems
Instructor: Gus Hanna, Ph.D.
Fall 2025

<Movie Theater Ticketing System>

Revision History

Date	Description	Author	Comments
<09/18/2 5>	<Version 1>	<Group 15>	<SRS Requirements 1.0>

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	

<Movie Theater Ticketing System>

Table of Contents

<Movie Theater Ticketing System>

<Movie Theater Ticketing System>

1. Introduction

The introduction of the Software Requirements Specification (SRS) details the requirements for the Movie Theatre Ticketing System(further referred to as MTTS). The MMTS is a web-based application which aims to provide customers a platform to purchase tickets. The application should also grant administrators to manage showtimes, pricing, and discounts, as well as be able to receive customer input and inquiries.

The aim of this document is to thoroughly detail the functions and requirements necessary for the MTTS to be usable, efficient, and reliable for customers, administrators, employees, and all that fall under the Movie Theatre business.

1.1 Purpose

The purpose of this SRS is to create a ticketing system for a movie theatre. The audience for which it is written is the general public interested in attending movie theatre showings.

1.2 Scope

The software for Theater Ticketing, will provide a way for customers to purchase tickets for a chain of 20 theaters within San Diego.

One goal is to provide a more convenient experience for buying tickets as they'll be available online, or on a kiosk, essentially eradicating any lines there may be at the theatres while also preventing movie tickets to be botted online ensuring the smoothest possible experience for a regular customer buying tickets for movies.

Another is to provide a more accurate and efficient way for the theatres to keep track of tickets sold/bought, can track/report data of movie sales, and just overall create a better way to track market data.

1.3 Definitions, Acronyms, and Abbreviations

FAQ	Answers
What is MTTS?	Abbreviated term for Movie Theatre Management System.
What is UI?	Abbreviated term for user interface.
What is SRS?	Abbreviated term for System Requirement Specifcations.

<Movie Theater Ticketing System>

1.4 References

- CS250 Template(Fall 2023)
- Lecture: Scenario and Use Cases
- Example System Requirement Specifications: Marvel Electronics and Home Entertainment

1.5 Overview

The rest of the SRS details the ticketing system itself, functions of the system, and how well it performs. The SRS further dives into the product's interfaces, constraints and various requirements. Our SRS from here on out will detail the product description, the specific requirements followed, and analysis models.

2. General Description

The Movie Theatre Ticketing System (MTTS) brings an innovative ticketing system available to customers through a browser-based system. The product aims to serve both the customer base and administration. Customers are able to access the system to view available movie times, tickets, and locations. Whereas administration will be able to manage showtimes, availability, promotions, and their respective details.

2.1 Product Perspective

The website will be a standalone application, however, it will be connected with the data/info base to different movie theatres in San Diego to receive new information back and forth without having to input it into different systems. It works in conjunction with other systems to provide a strong user experience.

2.2 Product Functions

This product should handle any customers who want to buy movie tickets at a theater, allowing for the user to easily purchase tickets either online or at a kiosk in the theater. If you are an administrator, you will be able to manage the website and edit as necessary. Furthermore, the

software will be able to handle loyalty rewards with the customer, properly storing purchasing data.

2.3 User Characteristics

This product targets two groups of users, the average movie watcher, and the employees of the theaters.

Movie goers are not expected to have any prior knowledge to handle the software, and should be able to navigate through the ticket buying process effortlessly.

Employees are expected to have moderate technical skills able to navigate the system properly to ensure a smooth experience for the customers. Higher management employees like managers should be expected to have advanced technical skills able to swiftly navigate the system and be able to manage schedules, update movie listings, track sales, and generate business reports. Of course the different levels of management will be separated by access codes.

2.4 General Constraints

The software is expected to be a website that works both at an in-person kiosk and online providing digital tickets that can be printed out but not replicated. It should be consistent and handle easy to interact with no matter the background, while performing on most devices.

2.5 Assumptions and Dependencies

The website should be dependent on the different movie theatres database containing all the movies listed, tickets available, etc. If the internal system of the movie theatres database should go down, information pertaining to that theatre will be unavailable but the site should not shut down. It will simply not list information about it or provide a system down message.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces
- Website interface
  - Movie listings
  - Different movie theatre locations
  - Checkout/payment screens
  - Maybe "featured" movies

- Seat selections
- Different menus
- Search bars
- Language chooser
- Something that asks for location permissions
- Page for customer input
- Loyalty system

3.1.2 Hardware Interfaces

- Theatres provide standard printers for printing tickets and corresponding receipts.
- Theatres provide electronic e-ticket scanners compatible with generated QR codes for online purchases.

3.1.3 Software Interfaces

- Showtime Database: Updates available showtimes, deletes expired showtimes, and creates new showtimes for upcoming showings.
- Ticket Database: Stores tickets from verified purchases and updates seat availability based on purchase.

3.1.4 Communications Interfaces

- Created in HTTPS for website
- APIs to communicate between databases or payment systems
- Auto generated emails for sending ticket confirmations
- QR code generators/validations for tickets purchased
- Location services

3.2 Functional Requirements

3.2.1 <Viewing for available showtimes>

3.2.1.1 Introduction
- This should allow the user to 'see' different showtimes for a variety of movies at different theatres within San Diego

3.2.1.2 Inputs
- Clicking on different movie theatres or different movies

3.2.1.3 Processing
- System redirects or shows seating and available tickets derived from database
- Filters out bought out seats and shows accessible ones

3.2.1.4 Outputs
- Displays a screen of available seats if tickets are available.

3.2.1.5 Error Handling
- If no showtimes are unavailable, all options or those that are unavailable will be greyed out and not accessible.

3.2.2 <Movie search>

3.2.2.1 Introduction

- This function will allow the users to search for available movies by title, genre, and or date

3.2.2.2 Inputs
- The user entering a movie title in the search bar
- Filtering out genres to see only 1 genre or only some
- Selecting a date on a calendar to see what movie is shown that day

3.2.2.3 Processing
- System connects to the database for movies matching all filters
- System doesn't show or greys out unavailable movies

3.2.2.4 Outputs
- A list of movies matching the filters selected

3.2.2.5 Error Handling
- States no movies found or any sort of message along those lines
- Or if just down, will state system is down.

3.2.3 <Pay screen>

3.2.3.1 Introduction
- This is a way for users to pay for movies that they have chosen to buy tickets for

3.2.3.2 Inputs
- Clicking onto "checkout" when selecting seats/tickets
- Inputting payment methods

3.2.3.3 Processing
- Connects to database to mark a seat as "bought"
- Connects to payment systems to process payments

3.2.3.4 Outputs
- Shows a message saying the ticket has been purchased
- Sends an email to provided email for receipt and movie information

3.2.3.5 Error Handling
- Will state payment is failed
- Make sure that payment does not go through
- Gives a message depending on the error reason, payment method failed, system is down, etc.

3.2.4 <Sending out QR codes and receipts to emails>

3.2.4.1 Introduction
- This function will send out QR codes that reciprocate to the connection tickets through emails and as well as receipts

3.2.4.2 Inputs
- The user completes their payment for an available ticket
- The user provides an email for receipt/QR code to be sent to

3.2.4.3 Processing
- Connecting to the database to process ticket
- Connecting to an automatic email generator that auto fills for QR code, ticket information, etc.

3.2.4.4 Outputs

- The user will receive an email with the receipt, QR code for their ticket, and relevant information/instructions.

3.2.4.5 Error Handling
- Will not let the user correctly follow through with payment if email could not be sent/error is found. Will state some sort of error message

3.2.5 <Seat selection>

3.2.5.1 Introduction
- This function allows users to accurately choose what seat they will occupy during a movie

3.2.5.2 Inputs
- User clicking on non greyed out seats
- User having clicked onto available showtimes

3.2.5.3 Processing
- Connects to movie database to review seats that have already been taken
- Sends out a message to database to state a seat has been reserved

3.2.5.4 Outputs
- Will give a message that a seat has been reserved if clicked on a greyed out seat
- Will give a message saying that seat successfully has been reserved please go to checkout
- Will grey out the seat that they have reserved

3.2.5.5 Error Handling
- If two users click on the same seat but one clicked on it sooner than the other, the later user will receive an error message that the seat has been reserved and refresh their page.

3.2.6 <Refund/Cancel booking>

3.2.6.1 Introduction
- This function allows the user to refund their ticket and/or cancel their booking

3.2.6.2 Inputs
- The user opens the site or to view their booking information
- Clicking on the refund options

3.2.6.3 Processing
- Accesses the database to check for ticket
- Sends back to site if ticket there
- Cancels ticket and booking in database
- Access payment method saved/used
- Returns payment

3.2.6.4 Outputs
- Receives email on refunded/canceled ticket/booking
- Gets taken to ticket refunded page

3.2.6.5 Error Handling
- If too late will send show a message
- If cannot find booking will ask to visit in person or email
- If ticket already refunded will let them know

<Movie Theater Ticketing System>

3.2.7 <Loyalty System>

    3.2.7.1 Introduction
-    This function allows the user's purchases to be tracked and allows the user to be rewarded based on their loyalty to the theater.

    3.2.7.2 Inputs
-    The user accesses their loyalty amount
-    The user's name/phone number
-    Verify the user identity

    3.2.7.3 Processing
-    Access the database to check previous purchases
-    Calculate loyalty points based on amount of money spent

    3.2.7.4 Outputs
-    Shows the user possible discounts based on the amount of loyalty points
-   Gives the user a discount if they choose to use it

    3.2.7.5 Error Handling
-    Ensure that the customer has enough points to use the loyalty discount
-

3.3 Use Cases

3.3.1 Use Case #1

A customer will be able to utilize the website to select movies and buy tickets while being able to check their booking status and request refunds. Payments are processed through the payment gateway and staff must verify tickets.
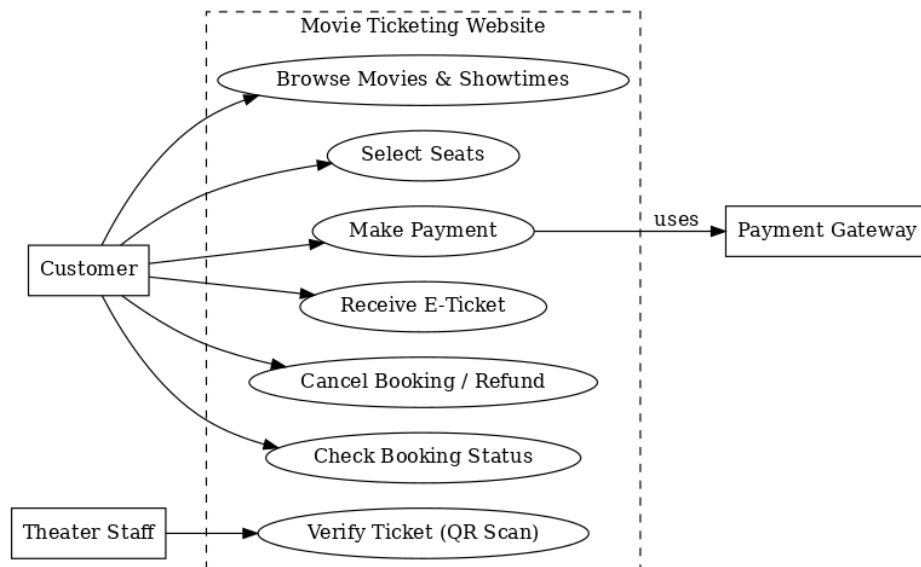
3.3.2 Use Case #2

Rather than using the website, a customer can use a kiosk found in the theater also to purchase tickets, browse movies, and receive an e-ticket. The kiosk mainly has ticket purchasing capabilities and payment is also processed through payment gate, tickets are still verified by staff.

Movie Ticketing Kiosk
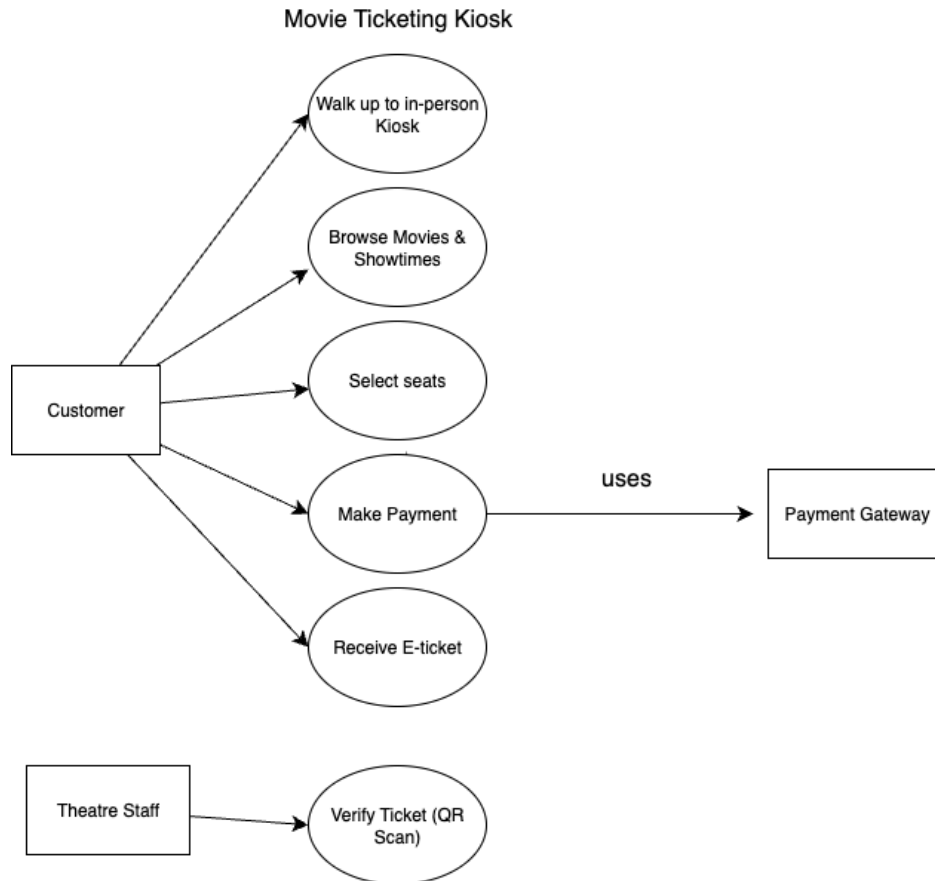
3.3.2 Use Case #3

The website and kiosk both give higher access to admins, mainly for bugfixing. They are able to enter admin mode and check for errors to fix and ensure functionality and also have access to update features on the site.

<Movie Theater Ticketing System>

Movie Ticketing Website/Kiosk Error Fix
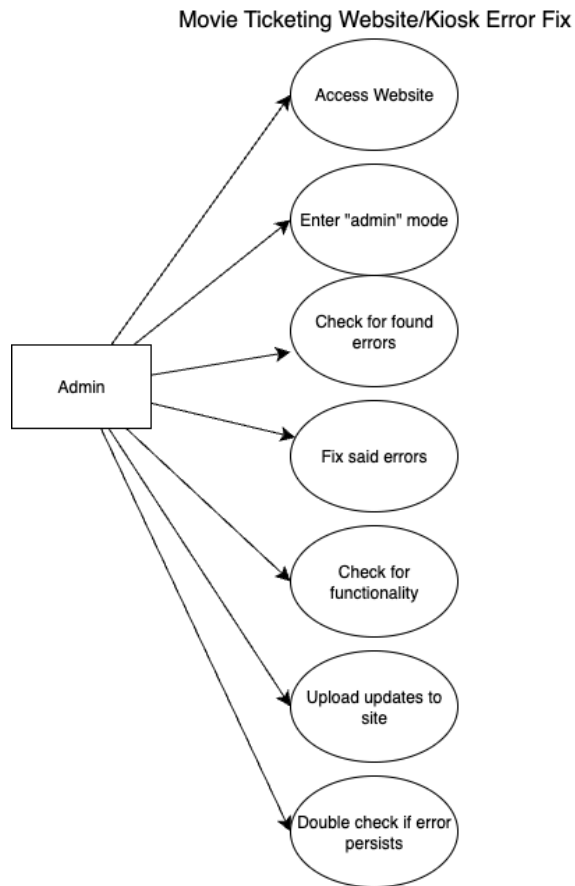
3.4 Classes / Objects

3.4.1 Customer

Customer
3.4.1.1 Attributes: - String name - String email - List<Ticket> orderHistory - List<String> feedback - int loyaltyPoints
3.4.1.2 Functions: - String getName(); - void setName(String name); - String getEmail(); - void setEmail(String email);

<Movie Theater Ticketing System>

```
-   List<Ticket> getOrderHIstory();
-   List<String> getFeedback();
-   void purchaseTickets(Showtime showtime, int quantity);
-   Void submitFeedback(String comment);
-   int getLoyalty();
-
```

3.4.2 Administrator

<div align="center">

Administrator

</div>

3.4.2.1 Attributes:
- String name
- String adminID
- String role
- String permissions

3.4.2.2 Functions:
- String getName();
- void setName(String name);
- String getAdminID();
- void setAdminID(String adminID);
- String getRole();
- void setRole(String role);
- String getPermissions():
- void setPermissions(String permissions);
- void manageSeatAvailability(Showtime showtime)
- void moderateFeedback(List<String> feedbackList);
- void implementDiscounts(List<Discount> discounts);
-

3.5 Non-Functional Requirements

Non-functional requirements may exist for the following attributes. Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the following sections in measurable terms (e.g., 95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, > 30 day MTBF value, etc).

3.5.1 Performance
-   The system should be able to handle at least one-thousand people at any given time.
-   The system should be able to limit the amount of tickets bought at once to twenty.
    -   The system should be able to block bots who are trying to buy tickets in bulk for high-demand movies.

- The system should allow purchase access between two weeks prior to the date of the showing and ten minus before showtime.
- The system should run efficiently; Pages should be 100% loaded within 3 seconds
- Purchases should be processed and completed within 5 seconds

3.5.2 Reliability

- Seats are reserved during the process of checkout and payment for up to a ten-minute period.
- Transactions should be made through customers providing card information, or through third-party payment methods.
- Transactions must be immediate and paid in full.
- The system should have nearly no downtime, downtime should be completed within one hour.

3.5.3 Availability

- The availability of the system should be all days of the week and twenty-four hours of each day.
- The system should only be unavailable for software updates; alternatively, tickets should still be available for purchase in person at the customers' desired location of attendance.
    - Refer to reliability for more details regarding downtime

3.5.4 Security

- The system should prevent bots trying to purchase a large volume of tickets.
- Payment information saved in customer accounts will be encrypted and protected.
- Admin actions and changes will all be documented.
- System will verify if a payment was actually made before providing any form of ticketing

3.5.5 Maintainability

- The system will keep updated with frequent maintenance.
- Admin will take in user feedback to increase functionality and usability.
- Updates can be deployed easily during a short downtime, being done in under an hour

3.5.6 Portability

- The system will be available on all devices and be able to run on all major browsers with ease.
- The UI will be responsive and adaptive based on the browser and device of either the customer or admin.

3.6 Inverse Requirements

*State any *useful* inverse requirements.*

- The system will not allow purchases of anything without providing a payment method
- The system will not allow access to the administrator controls without the password and login authentication of an admin
- The system will not allow double booking of the same seat of the same showtime of the same movie

- The system will not allow or show movie listings/showtimes that have already past
- The system will not allow the purchase of tickets for movies that have already past
- The system will not allow the user to keep their ticket after refunding their purchase
- The system will not allow users to log into accounts without the proper credentials needed

3.7 Design Constraints

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

- The system must be compatible with payment programs for secure processing
- The system must be compatible with all different kinds of browsers and computing systems
- The system must be able to handle high amounts of traffic without crashing
- System updates must comply with the company policies and laws of the state and country
- System must be developed under the budget provided by the company
- System must follow data privacy laws
-

3.8 Logical Database Requirements

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

- A database will be used for the Movie Theatre Ticketing System
- As for data formats, prices and payments will be stored in decimal or integer-cents formats to prevent rounding errors.
- Text data will be stored in standard text format.
- Dates and times will be saved in consistent format, and times will be specific to the location of the theatre for the consumer or admin.
- For storage capabilities, the database will back up data frequently to prevent data loss.
- Older data will be archived or deleted.
- Sensitive fields will be encrypted before storage.
- The database must be able to handle large volumes of transactions, both from high-demand showtimes and transactions built up in the database over time.
- In terms of data retention, order and payment records will be retained for a minimum of one year. This is so records can be audited or reported.
-

3.9 Other Requirements

Catchall section for any additional requirements.

4. Analysis Models

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

4.1 Sequence Diagrams (UML Diagram)

Necessary classes + fields

Movie
Fields
movieName: String
date: int
time: int
rating: double
location: String
genre : String
synopsis: String
seats: Map<bool>

Methods
setName(String): void
getName(): String
getFreeSeats(): Map<bool>
seatIsFree(index): bool

Location

Makepayment

accountInfo (users/employees)
Fields
username: String
password: String
name: String
emailAddress: String

adminInfo (employees)
Fields:
accessLevel: int

Methods:
deleteMovie(Movie): void

<Movie Theater Ticketing System>

setTime(Movie, int): void
setSynopsis(String): void
setGenre(String): void

customerInfo (users)

4.3 Data Flow Diagrams (SWA Diagram)

4.2 State-Transition Diagrams (STD)

5. Change Management Process

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

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

\<Movie Theater Ticketing System\>

Software Design Specification

\<Version\>

\<10/9/2025\>

\<Group 15\>
\<Christian De Leon, Jacob Tabalon, Max Caguioa\>

Prepared for
CS 250- Introduction to Software Systems
Instructor: Gus Hanna, Ph.D.
Fall 2025

<Movie Theater Ticketing System>

System Description

Brief overview of system

Software Architecture Overview

Architectural Diagram of All Major Components

UML Class Diagram

Description of Classes

Description of Attributes

Description of Operations

descriptions should be detailed and specify datatypes, function interfaces, parameters, etc..
NOTE: There's a difference between SRS document list of design diagrams required to include and those required to submit in the Rubric. Please, follow the Rubric's design diagram requirements.

Development Plan and Timeline

Partitioning of Tasks

<Movie Theater Ticketing System>

Team Member Responsibilities