<Theater Ticketing System>

Software Design Specification

<Version >

<2/27/2025>

<Group 7>
<Anthony Nguyen, Steven To, Charlie Pham>

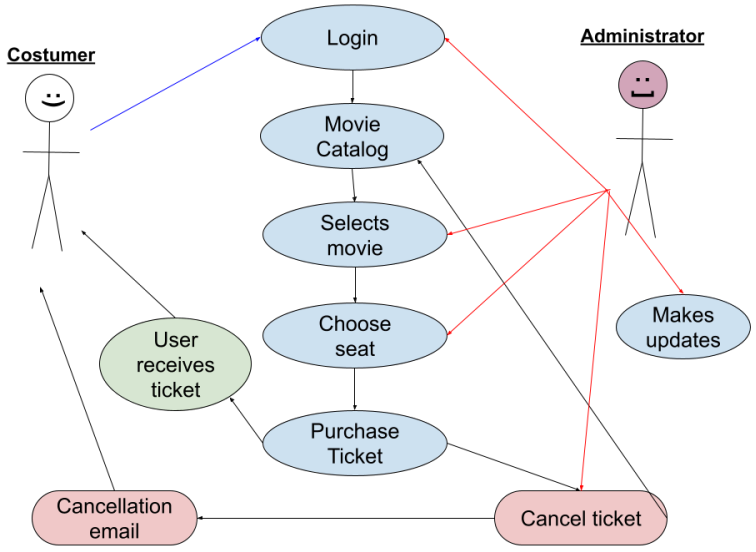
Prepared for
CS 250- Introduction to Software Systems
Instructor: Gus Hanna, Ph.D.
Fall 2023

System Description

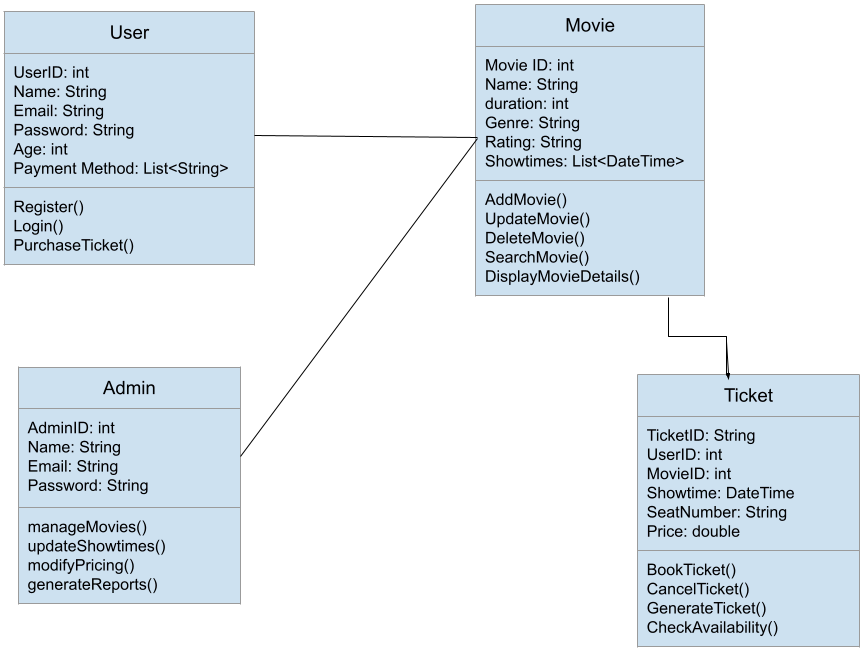
The theater ticketing system is a web-based application that allows users to browse available moving screenings, select seats, and purchase digital tickets. The system offers real-time seat availability, secure payment integration, and automated notifications. It also includes administrative functionalities for managing movie schedules and ticket pricing. The system aims to enhance convenience for customers and optimize theater operations.

Software Architecture Overview

- Architectural diagram of all major components



- UML Class Diagram



- Description of classes
- Description of attributes & Description of operations

User: Individual user who can register, log in, and purchase movie tickets	
3.4.1.1 Attributes	
UserID: int	A unique identifier for each user.
Name: String	The full name of the user.
Email: String	The user's email address, is used for account creation and communication.
Password: String	A secure password for account authentication.
Age: int	The user's age, is used to enforce the minimum age requirement (14+).
PaymentMethods: (List<string>)	A list of payment methods associated with the user.
3.4.1.2 Operations	
Register(name, email, password, age): void	Allows a new user to create an account.

User: Individual user who can register, log in, and purchase movie tickets	
3.4.1.1 Attributes	
Login(email, password:): bool	Authenticates the user and grants access to the system.
PurchaseTicket(movieID, showtime, seatNumber): Ticket	Purchases a movie ticket.

Movie: Stores details of movies available in the system	
3.4.2.1 Attributes	
MovieID: int	Unique identifier for the movie.
Title: string	Title of the movie.
Description: string	Brief synopsis of the movie.
Duration: int	Runtime of the movie in minutes.
Genre: string	Category of the movie (e.g., "Action", "Comedy").
Rating: string	Movie rating (e.g., "PG", "PG-13").
Showtimes: List<DateTime>	List of available showtimes.
3.4.2.2 Operations	
AddMovie(title, description, duration, genre, rating): void	Adds a new movie to the system.
UpdateMovie(movieID, title, description, duration, genre, rating): void	Updates movie details.
DeleteMovie(movieID): void	Removes a movie from the system.
SearchMovie(keyword): List<Movie>	Searches for movies by title, genre, or keyword.
DisplayMovieDetails(movieID): Movie	Displays detailed information about a movie.

Ticket: Handles movie ticket information and seat reservations	
3.4.3.1 Attributes	
TicketID: String	A unique identifier for each ticket.
UserID: int	The ID of the user who purchased the ticket.
MovieID: int	The ID of the movie associated with the ticket.
Showtime: DateTime	The date and time of the movie screening.
SeatNumber: String	The seat number assigned to the ticket.
Price: double	The price of the ticket is converted to the user's local currency.
3.4.3.2 Operations	
BookTicket(userID, movieID, showtime, seatNumber, price): Ticket	A user can book a ticket for a specific movie and showtime.
CancelTicket(ticketID): void	This enables a user to cancel a booked ticket.
GenerateTicket(ticketID): string	Generates a digital ticket for the user.
CheckAvailability(movieID, showtime): int	Check seat availability for a specific showtime.

Admin: Manages backend operations, including movie listings and reports	
3.4.3.1 Attributes	
AdminID: int	Unique identifier for administrators.
Name: String	Name of the administrator.
Email: String	Email for authentication.
Password: String	A secure password for account authentication.
3.4.3.2 Operations	
manageMovies(): void	Adds, edits, and removes movie listings.
updateShowtimes(): void	Modifies showtimes for movies.
modifyPricing(): void	Adjusts ticket pricing.
generateReports(): List<Report>	Produces sales and user activity reports.

- descriptions should be detailed and specify data types, 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
- Team member responsibilities