Software Requirements Specification Documentation

Theatre Ticketing System

9/21/2023

Created by:

Group #4

Moe Jawadi

Joel Alvarado

Ernesto Cornejo

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

Revision History

Date	Description	Author	Comments
<date>	<Version 1>	<Your Name>	<First Revision>
9/21/23	Requirement Specification	Moe Jawadi	
9/21/23	Requirement Specification	Joel Alvarado	
9/21/23	Requirement Specification	Ernesto Cornejo	

Document Approval

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

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

Table of Contents

1. Introduction & Overview

2. User Requirements

3. System Requirements

- **3.1 Functional Requirements**
- 3.1.1 - User Registration
- 3.1.2 - Profile Management
- 3.1.3 - Movie Availability
- 3.1.4 - Analytics and Reviews
- 3.1.5 - Payment Processing and Movie Selection
- 3.1.6 - Receipts and Purchase History
- 3.1.7 - Newsletter

- **3.2 Non-Functional Requirements**
- 3.2.1 - Security
- 3.2.2 - UI
- 3.2.3 - Servers and Response time
- 3.2.4 - Portability
- 3.2.5 - Reliability
- 3.2.6 - Logical Database Requirements
- 3.2.7 - Fraud
- 3.2.8 - Availability
- 3.2.9 - Inverse Requirements
- 3.2.10 - Maintainability
- 3.2.11 - Customer Support

4. Analysis Models

5. Conclusion

1. Introduction & Overview

The purpose of our website is to allow users to buy tickets to movies that are currently playing or upcoming. In order for the user to access our services they will be required to create an account. The user will be able to choose a time and day a specific movie plays, they will also be able to search through movies that are currently playing or upcoming movies. The user will also be able to filter the movies displayed on screen by choosing a specific genre, actor, showtime, day, and movie title. There will be no refunds however a user can receive credit in their account that doesn't expire which can be used to watch a different movie at any time. The user will be allowed to purchase and reserve up to 20 seats to a movie with our interactive seating chart. An optional newsletter with coupons will be sent out to its users twice a month, which can be disabled at any point.

2. User Requirements

- Access to the internet
- Computer or Laptop
- An Email
- An account on our website
- Debit Card on file or similar purchasing power that meet our requirements
- Optional(game console, or smart watch)

3. System Requirements

3.1 <u>Functional Requirements</u>

1. **User Registration**

- Users should be able to create an account and login with no pop ups or ads.
- System will ask to save password for future logins upon account creation.
- The System will provide many ways to login, (Login with Google), (Login with Apple).
- Users will be asked to enable and add a (2FA) authentication method upon logging in to help prevent account safety.

2. **Profile Management**

- Users will be able to change account preferences.
- Users will be able to save their favorite movies.
- Users will be able to save payment methods.
- Users will be able to update payment methods.
- Users will be able to customize their (2FA) authentication preferences

to fit wants and needs.

3. **Movie Availability**
   - The front page will display numerous movies.
   - The front page will include Showtimes, titles, and Genres.
   - The system will have a filter option to help the browsing experience.

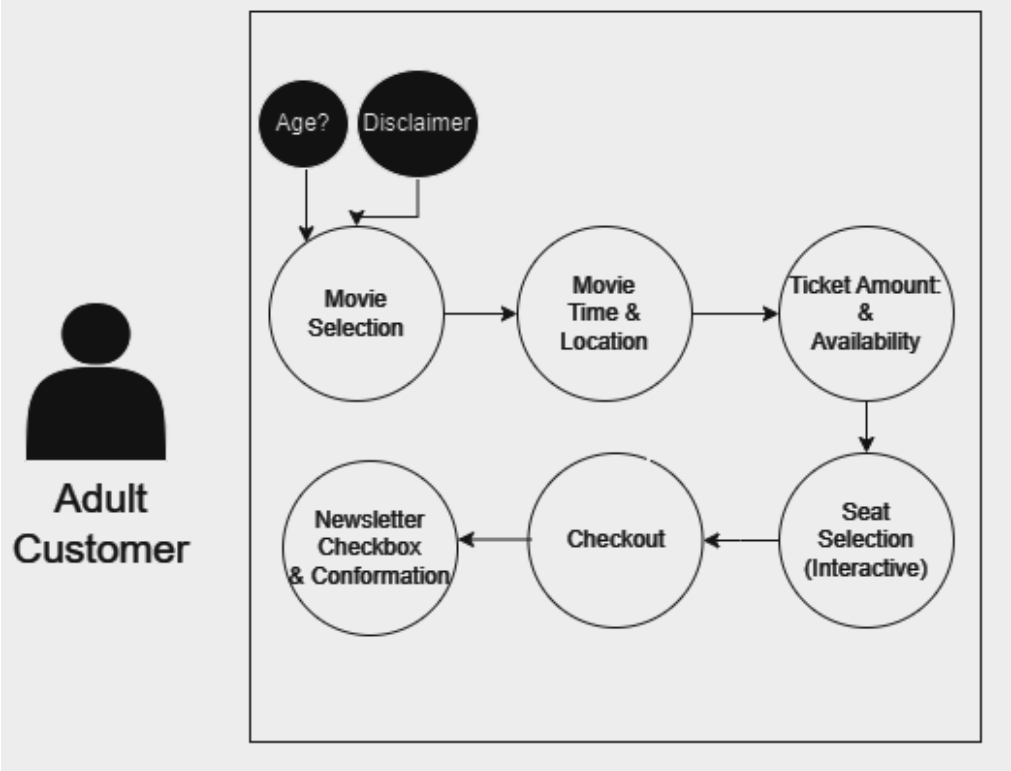
4. **Analytics & Reviews**
   - Data of how many tickets were sold within the previous hour for a specific show that is browsed
   - Data of what shows are trending, most popular, and best reviewed by our certified customers.
   - Ratings of every movie will be shown on the bottom right of every movie icon or (picture)
   - Customers will be able to see the positive/negative reviews on every show available by simply clicking on the show.

5. **Payment Processing & Movie Selection**
   - Payment Options of Debit cards, Credit Cards, Apple Pay, CryptoCurrency (BTC & ETH), and Paypal.
   - Users will have the ability to click on any movie, select any seat and time that is available upon searching.
   - Users will be able to view an interactive seating chart to view the seats they are interested in.
   - Users will be able to select up to (20) tickets of purchase at once upon each transaction to limit the amount of tickets that can be purchased by a single individual.

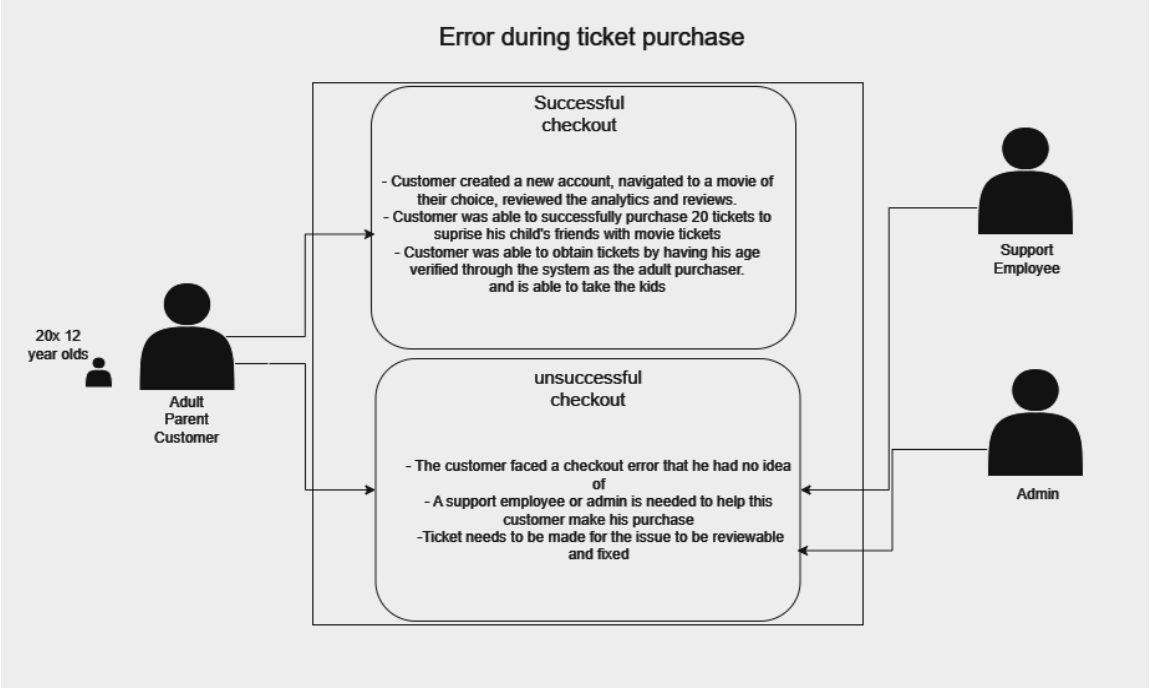
6. **Receipts & Purchase History**
   - After every purchase, a confirmation page will appear as well as a confirmation email will be sent.
   - Purchase history of all movies will be saved and found in the Profile Preferences under “Previous Purchases”.

7. **Email Newsletter**
   - Discounts and Coupons will be sent out commonly twice a month to loyal returning customers who agreed to the terms and conditions of the Email Newsletter upon purchase the newsletter can be disabled by the user at any point.

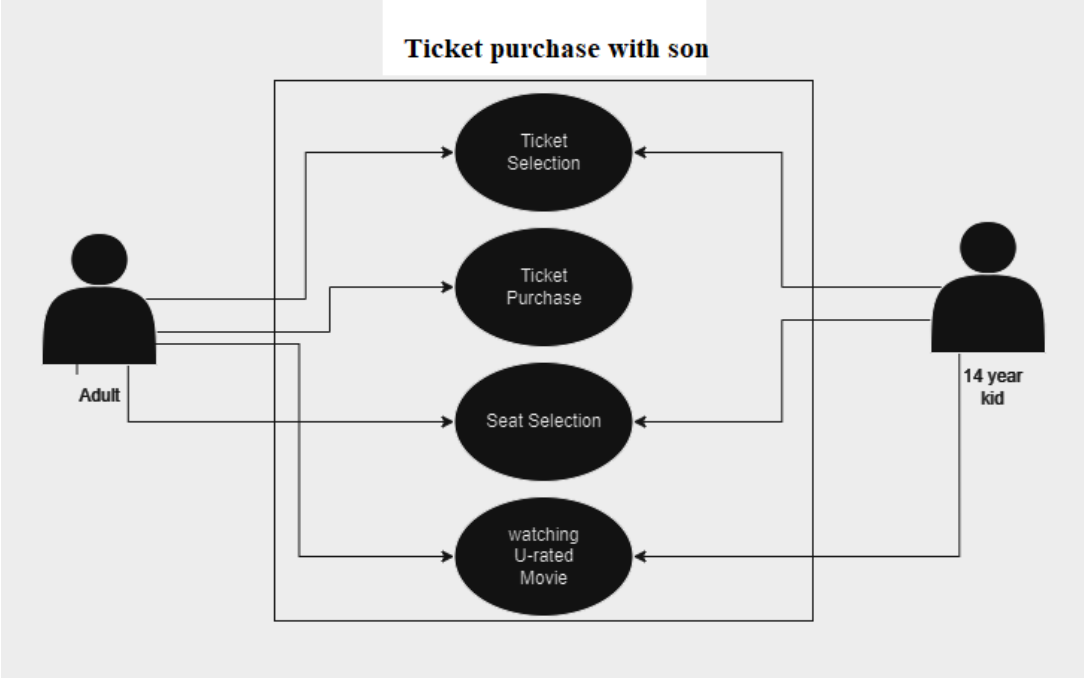
Use Case Diagram #1:



Use Case Diagram #2:



Use Case Diagram #3:



3.2 Non-Functional Requirements

1. **Security**
   - Information such as user information, passwords, or payment methods will be secured and preventable from any data leaks or breaches.
   - All customer data is encrypted, both in transit and stored. Admin access uses multi-factor authentication. Account brute force attacks are safeguarded against.
   - Customers have the ability to enable 2 step authentication (2FA) to take any precautions against any unpreventable attacks.

2. **UI Configuration**
   - UI will be very easy to use as it will be designed to be very friendly, easy to use and navigate.

3. **Servers & Response time**
   - The average website response rate is 2.5, our website's response time is 2.0, so it will be ahead of many websites and will be lag-free.
   - Server can handle over 10,000 orders during peak with no problem

4. **Portability**
   - UI can be accessed through all available web browsers.
   - UI can be accessed through all available devices including game consoles and smart watches.

5. **Reliability**
   - The system should maintain over 99.9% uptime measured monthly. No more than 1% of transactions should fail due to technical issues.

6. **Logical Database Requirements**
   - The System uses a SQL Database
   - Stores information such as shows, times, seatings, charts, availability, customer accounts and orders, payments, admin access and activity logs.
   - Database normalization principles reduce redundancy. Foreign keys enforce data relationships.

7. **Fraud**
   - Fraud detection identifies suspicious purchasing patterns. Customer service can access order histories to assist users. Notifications are sent when shows or sales are added.

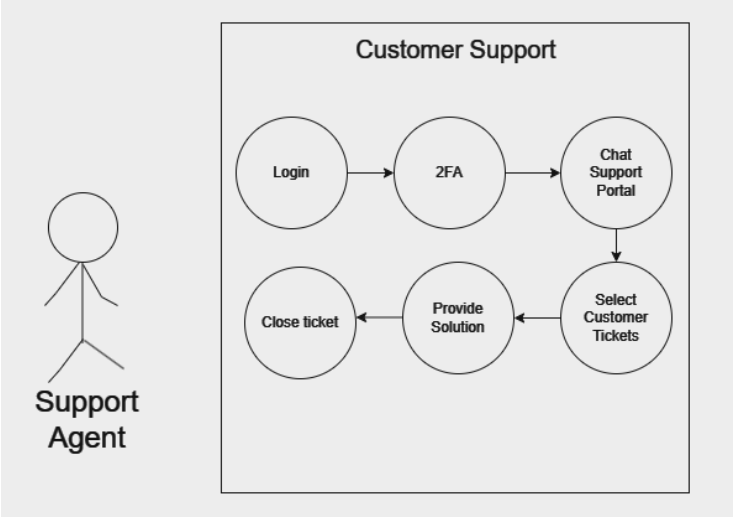
8. **Availability**
   - The site should be up 24/7 unless we have scheduled maintenance. We want unplanned downtime to be less than 1% of the year.
   - Movies will be updated frequently by administrators including movie times, seats and etc.

9. **Inverse Requirements**
   - The System will not sell your information to any person or company.
   - The System will have no security breaches to damage or have any stolen content or private information of customers.
   - The System will not automatically save your payment information without asking.
   - The System will not email you if you have not checked the news letter box upon a purchase.

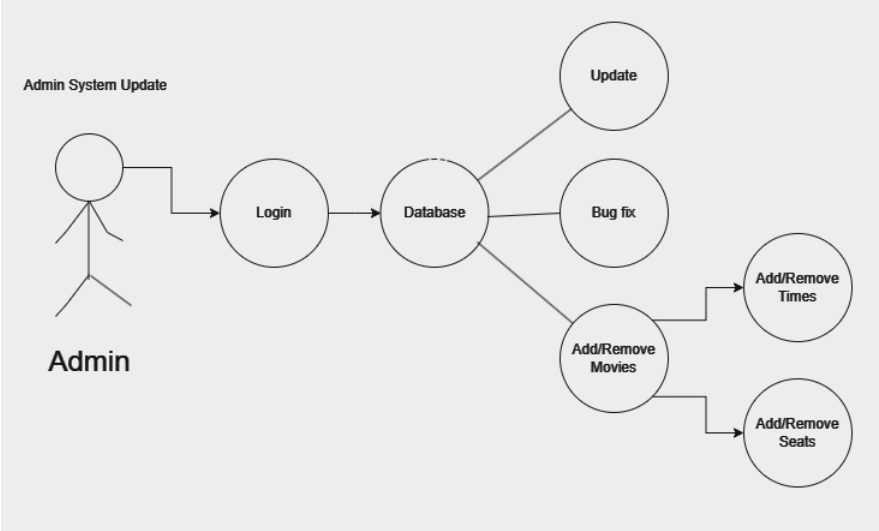
10. **Maintainability**
   - The web application will support regular updates to ensure compatibility with latest technology.
   - The web application will apply bug fixes regularly.
   - The web application will apply security patches when needed.

11. **Live Support**
   - On the bottom right of the website we have a live support chat that is 24/7 to help ensure customers get the help they need.
   - Live support agents can help with purchase problems, ticket problems and any errors the customer might have

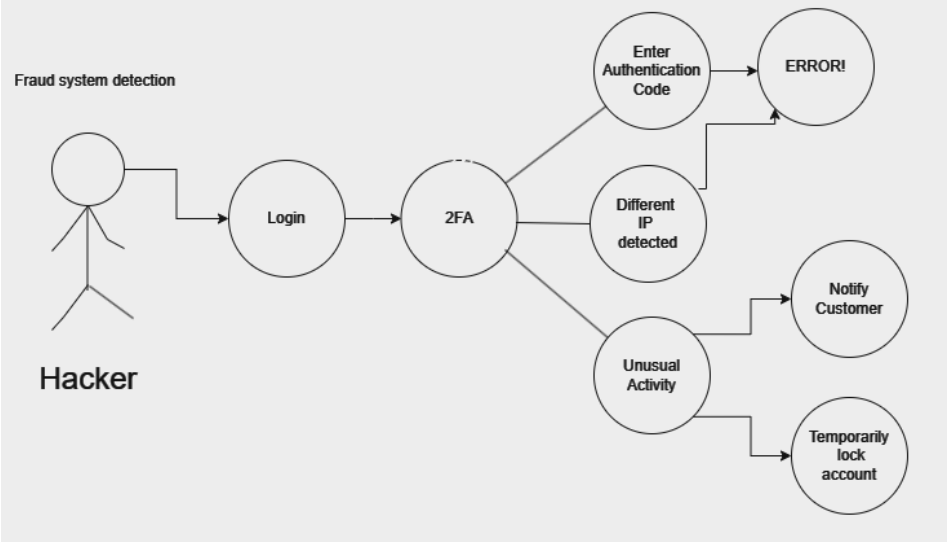
Use Case Diagram #1:



Use Case Diagram #2:



Use Case Diagram #3:



4. Analysis Model Diagram

5. Conclusion

In conclusion, we have one of the best websites that will cater to your movie ticket needs 24/7 with our filter tool that helps you search the right movie for you and your friends. Our website's response time, security, fraud detection, as well as a personalized account based on previous purchases is also something that will surely catch your attention.

Software Design Specification

Movie Ticket System

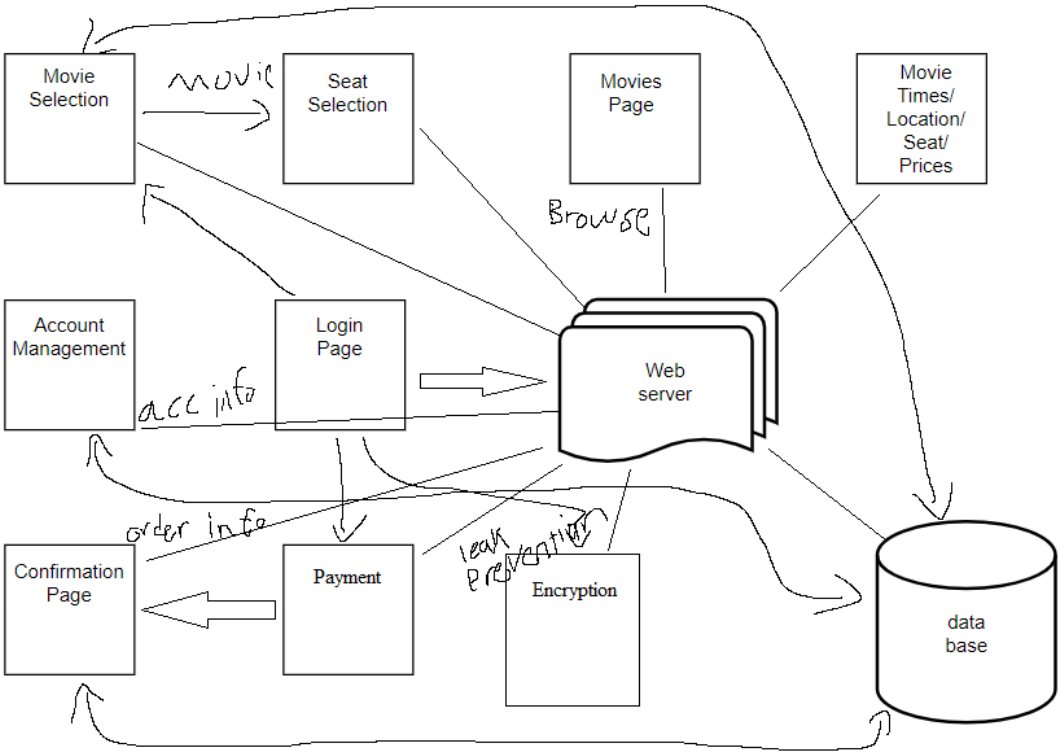
Moe Jawadi
Joel Alvarado
Ernesto Cornejo

System Description

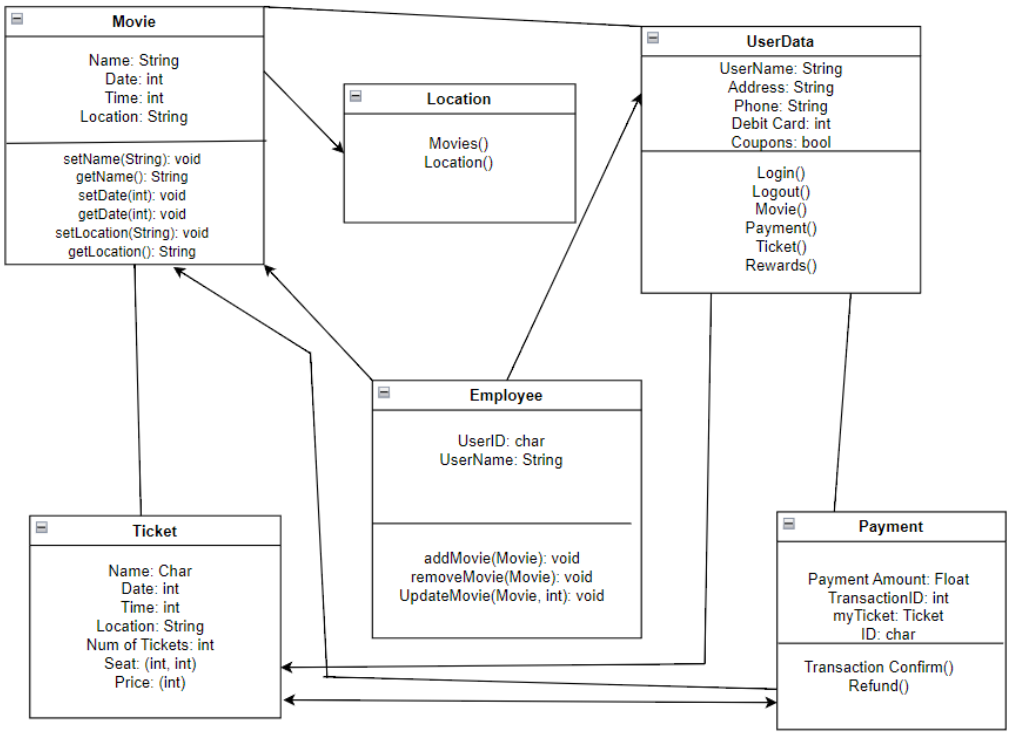
The movie ticket system is a web application that allows users to purchase movie tickets online. Users can browse movies by showtime, genre, title, etc and select seats on an interactive seating chart. The system manages user accounts, profiles, payment processing, and sends email receipts/confirmations. Admins can add new movies, showtimes, and seating availability. The movie ticket system was developed using a responsive design approach that will allow users to connect to our website via tablet, desktop, smartphone, laptop, smartwatches, and video game consoles. It will have a database that will store information about shows, times, seatings, customer accounts and their saved orders. Our system will have security measures such as two factor authorization, key encryption, and SSL certificate in order to protect user information.

Software Architecture Overview

ARCHITECTURAL DIAGRAM



UML DIAGRAM OF SOFTWARE DESIGN



- **Description of classes**

  Movie class: The Movie class is set to give us provide a lot of information about any movie that is selected.

  Location class tells us what is the movie and where is its location

  UserData class showcases any user based information of a customer

  Employee class gets special access to UserData and Movie

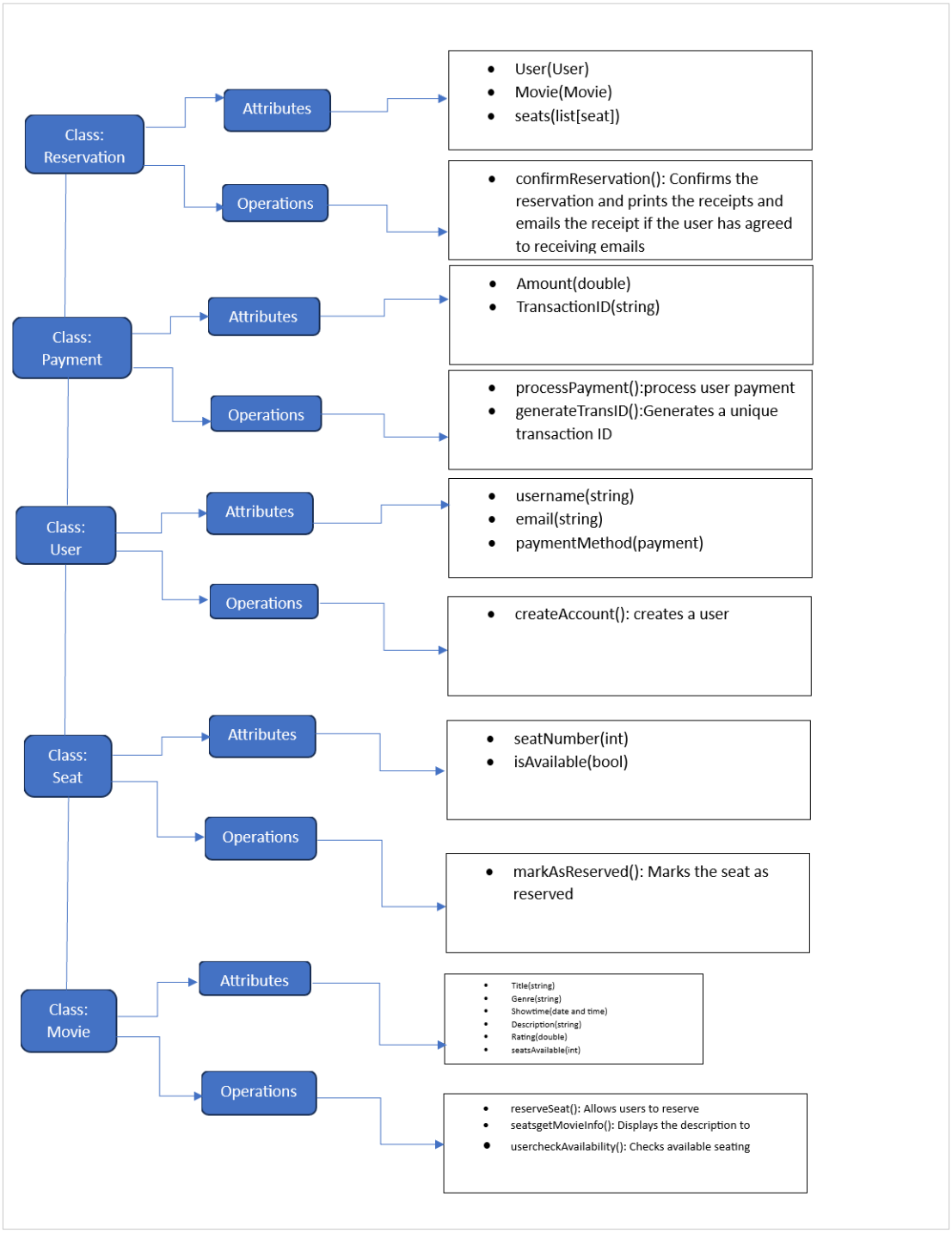
  Ticket class includes all information regarding purchased or browsed tickets.

  Payment class displays the payment amount, transaction ID and the payment ID.

- **Description of attributes**

  Attributes and characteristics of each class are showcased in the diagram above, when we set something to its value, such as Name to String, Date to int, Time to int, Location to String, add movie, remove movie, update movie and etc. we used these attributes to help us get the name of our movie, date of our movie, time of our movie, location of our movie, ability to update, add or remove a movie without any complications.

SWA DIAGRAM OF SOFTWARE DESIGN



Description of Classes

The createAccount() allows users to create an account

The reserveSeat() allows users to reserve up to 20 seats

The getMovieInfo() displays a synopsis of the movie

The userCheckAvailability() checks for any seats available for a particular movie and show time

The markAsReserved() marks seats that have been reserved in order to not allow a different user to choose that same seat

The confirmReservation() confirms the users reservation and prints the receipt or emails the receipt to the user

The processPayment() processes user payment

The generateTransID() generates a unique transaction ID that authenticates user purchase

Description of Attributes

We will start off with user attributes such as username which takes in a string value as well as email, but for paymentMethod we will take in a payment object. For the Movie attributes we will have a few string data types such as title, genre, and description. We will also have showtime which will be a date and time, we will also have rating which is of double data type, and seatsAvailable which is an int. Our attributes for seat are only seatNumber, which is an int value and isAvailable which is a boolean data type. Our Reservation has attributes that take in objects as their parameters such as User(User) which shows which user reserved certain seats, and Movie(Movie) which shows what movie the reservations were made for, and lastly Seats(list[seats]) Which holds the list of the seats that are reserved for a given movie. Our last attributes are of the payment class. The attributes are as follows, Amount which is of double data type and displays the total balance the user needs to pay, and TransactionID which is a string data type and is a unique transaction identifier.

Development Plan and Timeline

Partitioning of tasks

Moe: Implement user registration, login, profiles and settings pages

Joel: Implement movie browsing, selection, seating chart and payment pages

Ernesto: Implement database schema, movie/showtime admin features

Timeline

Week 1: Set up development environments, build basic User Interface pages

Week 2: Implement user account management features

Week 3: Implement movie browsing and selection features

Week 4: Implement seating chart, payment and order management

Week 5: Complete admin features, testing and deployment