Movie Theater Ticketing System

Software Requirements Specification

Version 4

8/4/25

Ethan Loun

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

Revision History

Date	Description	Author	Comments
7/15/25	Version 1	Ethan Loun	First Revision
7/22/25	Version 2	Ethan Loun	Second Revision
7/29/25	Version 3	Ethan Loun	Third Revision
8/4/25	Version 4	Ethan Loun	Fourth Revision

Document Approval

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

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

Table of Contents

1. Introduction

The introduction to the Software Requirement Specification (SRS) document should provide an overview of the complete SRS document with purpose, scope, definitions, acronyms, abbreviations, references, and overview of the SRS. This document aims to analyze and report on the complete Movie Theater Ticketing System by outlining the requirements in detail.

1.1 Purpose

This document's purpose is to compile and specify concepts that shape the system from the consumer's perspective. It gives an overview of the product, including its functionality and parameters, to understand the project. This document outlines the concepts for future additions and those that may be changed as the projects progress. The SRS presents the system's functionality, use cases, target audience, user interface, hardware, and software requirements. Furthermore, this document serves as a guide for designers and developers throughout the development life cycle.

1.2 Scope

The software will be able to provide users with a web browser that can search for movies, purchase tickets, select seats, showtimes, receive customer feedback, and implement reviews of the movies. This website will be available to operating systems such as Windows, Mac, and Linux to ensure diverse audiences. The platform will support two user roles: customers and administrators. The customers will manage their bookings and tickets online, while the administrators will manage the backend such as movie listings, monitoring bookings, and showtime listings. This software is intended to be used through the web browser, so specialized hardware and software is not required for the users. The goal is to create a secure, user-friendly movie theater ticketing website. The SRS outlines the requirements for the software to be developed and can be used as a reference.

1.3 Definitions, Acronyms, and Abbreviations

SRS	Software Requirements Specification
HTTPS	Hypertext Transfer Protocol Secure
API	Application Programming Interface
PCI DSS	Payment Card Industry Data Security Standard
Backend	Server-side that contains the data and infrastructure

1.4 References

- Bandakkanavar, Ravi. “Software Requirements Specification Document with Example.” *Krazytech*, 28 May 2025, krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database.
- IEEE Recommended Practice for Software Requirements Specifications
- Software Requirements Specification – Example, SRS4.0
- Theater Ticketing Requirements
- Theater Ticketing System

1.5 Overview

After the introduction, section 2 provides a general overview of the ticketing system, covering its features and user attributes. The specific requirements, including use cases, interface standards, and functional and non-functional requirements, are described in section 3. Section 4 presents supporting models and diagrams.

2. General Description

2.1 Product Perspective

The movie ticketing system is a new web-based solution designed to replace the client's existing ticketing infrastructure. While the current system meets minimal operational requirements, it suffers performance issues and lacks scalability and modern security features. The new system will meet the following needs.

- **Access**:
  The new system will support both online access via desktop and mobile web browsers and in-person uses via digital kiosk, able to support up to 1000 concurrent users and be scalable to support as much as needed.
- **Database**:
  The new system will interface with a centralized, partitioned database shared amongst all theaters, managing real-time ticket availability, seat reservations, and user interactions. It includes administrative capabilities for operational control and keeps systemwide daily logs of ticket purchases.
- **Payment & Review**:
  The new system will integrate with various payment gateways, review aggregation sources, and user feedback mechanisms.

2.2 Product Functions

The primary functions of the system are as shown in below:

- Browsing available movies and showtimes
- Selecting and reserving specific seats (deluxe only) or general admission (standard theaters)
- Processing transactions, with a limit of 20 tickets per transaction, and integrating customer discounts (student, senior, military)
- Handle secure transaction processing through multiple methods
- Enforcing purchase windows (tickets available from two weeks before showtime to 10 minutes after showtime starts)
- Supporting optional customer account registration and management, including loyalty programs, payment information storage, purchase history, and loyalty points
- Prevent bot-driven or automated bulk purchases and limit access to one device per user account at a time
- Support a queuing system for high volumes of requests for singles showings
- Enforce ticket uniqueness to prevent scalping and reselling through making each ticket an NFT
- Maintain a centralized log of daily transactions
- Display real-time ticket availability and enforce sales windows
- Provide customer feedback mechanisms after purchase
- Display aggregated movie reviews and critic quotes sourced from online review platforms
- Support multi-language interfaces (in English, Spanish, and Swedish)
- Support administrator functionalities such as overriding transactions, modifying showtimes, and managing theaters.

2.3 User Characteristics

The system is intended for use by general customers and administrators.

The General Customer purchases tickets online or at in-person digital kiosks. Users are expected to have basic web browsing skills and to access the system in supported languages (English, Spanish, Swedish).
The Administrator will have access to both customer and administrator functions. They will have access to information stored in the system database and management functions for movie showtimes, theaters, and transactions.

2.4 General Constraints

- The system must operate as a web-based platform accessible via standard web browsers on PC or mobile devices, as well as through in-person digital kiosks.
- It must support up to 1000 concurrent users to ensure scalability and avoid downtime during high-demand events.
- It must comply with security requirements, including prevention of ticket replication and bot-driven bulk purchases.
- The system will use a single centralized database partitioned per theater and must remain consistent and synchronized in real time.
- Only one device can be logged into a user account at any given time.

2.5 Assumptions and Dependencies

- The system assumes stable internet connectivity at all theaters.
- External review data sources will remain accessible and provide APIs or scraping capabilities as needed.
- Theaters provide accurate and timely updates on showtimes and seat availability to the centralized database.
- Payment gateways will be continuously operational and available for integration.
- Theaters will maintain dedicated staff for handling in-person refunds and physical ticket issues.
- Future expansion beyond the San Diego area or integration with additional theater chains is not within the current scope. If scalability beyond these requirements is needed, new requirements may be introduced.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

A graphical user interface will be used by the software. Through user-friendly visual tools, administrators and customers can control movie browsing and ticket management. These interfaces are enabled by the API which ensures the user interface and backend system communicate.

3.1.2 Hardware Interfaces

The web browser will be supported on devices such as smartphones, tablets, laptops, and PCs.

3.1.3 Software Interfaces

The ticketing system will run on standard operating systems like Windows, Mac, and Linux. It will be integrated with web browsers, APIs, and payment gateways for functionality.

3.1.4 Communications Interfaces

To focus on the browser-based function, the system will use HTTPS to ensure secure and reliable connections between clients and servers.

3.2 Functional Requirements

3.2.1 Scalable User Handling

3.2.1.1 Introduction: The system can support up to 1000 users accessing and performing operations simultaneously.
3.2.1.2 Inputs: The system should report inputs such as managing customer bookings through the web-based browser.
3.2.1.3 Processing: The system utilizes load balancing to optimize queries by the users.
3.2.1.4 Outputs: The system shows data through GUI in a timely response during peak usage.
3.2.1.5 Error Handling: The system will display a queue if there is an overload in user logins.

3.2.2 Web Browser

3.2.2.1 Introduction: The system will only be available through a web browser, and not an application.
3.2.2.2 Inputs: The system will use HTTPS on browsers such as Chrome, Safari, and Edge.
3.2.2.3 Processing: The system processes the request from the users through web technologies
3.2.2.4 Outputs: The system will support all features through the web browser with no need for external applications.
3.2.2.5 Error Handling: The system will notify the user if the browser is unsupported and will provide alternative solutions.

3.2.3 Bot Prevention

3.2.3.1 Introduction: The system will prevent scalpers and reseller from utilizing bots to purchase many high-demand tickets at once
3.2.3.2 Inputs: The system will allow ticket purchasing requests
3.2.3.3 Processing: The system will employ bot detection with behavioral analysis. The use of CAPTCHA, sourcing out IP addresses, and alerting non-human activity will mitigate bots
3.2.3.4 Outputs: The system will only allow legitimate purchases to be processed
3.2.3.5 Error Handling: The system will be alerted if there are bot-like behaviors and will lock out users if exceeded

3.2.4 Showtime and Ticket Availability

3.2.4.1 Introduction: The system will show the showtimes for the movies and how many tickets are available
3.2.4.2 Inputs: The users request from the interface to view availability
3.2.4.3 Processing: The system will use SQL databases to retrieve and process information about showtimes and tickets
3.2.4.4 Outputs: The system will show real-time updates on to the users
3.2.4.5 Error Handling: The system will notify the user if retrieval of data fails and ask the user to refresh the page for up-to-date information

3.2.5 Ticket Limitations

3.2.5.1 Introduction: The system will force customers to have a maximum purchase limit of 20 tickets. The tickets will be available to purchase 2 weeks in advance and 10 minutes after the showtime
3.2.5.2 Inputs: The number of tickets carted by the customer
3.2.5.3 Processing: The system will verify that each transaction will not exceed the 20-ticket limit and check the specific time window
3.2.5.4 Outputs: The system will send a confirmation or denial message whether the transaction was complete or not
3.2.5.5 Error Handling: The system will notify the user if the ticket limit was exceeded or the transaction was attempted outside of the time window

3.2.6 Administrator Mode

3.2.6.1 Introduction: The system will provide an administrative mode that allows authorized users access to management features
3.2.6.2 Inputs: The system request for the administrators' credentials
3.2.6.3 Processing: The system will allow the user to request sensitive actions and configuration settings
3.2.6.4 Outputs: The system will display the admins controls and tools
3.2.6.5 Error Handling: The system will display an authentication error if the credentials are incorrect and lock out users if several attempts are made

3.2.7 Customer Feedback

3.2.7.1 Introduction: The system will collect feedback from the users
3.2.7.2 Inputs: Through the web browser the user can submit feedback including ratings and comments
3.2.7.3 Processing: The system will store the data from the users and notify the administrator of entries
3.2.7.4 Outputs: The system will display a confirmation message to the user upon a successful submission
3.2.7.5 Error Handling: If the feedback is invalid then an error message will be sent to the user, and the admin will be notified

3.2.8 Discounts

3.2.8.1 Introduction: The system will handle student, military/veteran, and senior discounts on the weekdays
3.2.8.2 Inputs: The system will prompt the user to show a valid student ID, military/veteran ID, or drivers licenses
3.2.8.3 Processing: The system verifies that the information is valid, and it is on a weekday then proceeds to apply the discount to the transaction
3.2.8.4 Outputs: The system display the total discounted ticket price on their order summary, and they receive a confirmation
3.2.8.5 Error Handling: The system will notify the user of an invalid eligibility for the discount

3.2.9 Gathering Online Reviews

3.2.9.1 Introduction: The system will gather reviews and critic quotes from reputable sources
3.2.9.2 Inputs: Automations will collect data from sources when new data is released
3.2.9.3 Processing: The data will be formatted and stored when retrieved from trusted sources

3.2.9.4 Outputs: The system will display the up-to-date reviews and critic quotes about the movies on the details page
3.2.9.5 Error Handling: The system will display a message stating that the reviews are unavailable

3.2.10 Supported Languages

3.2.10.1 Introduction: The system will support English, Spanish, and Swedish
3.2.10.2 Inputs: The system will prompt the user to select a preferred language
3.2.10.3 Processing: The system will select the appropriate language depending on the user's inputs or default to the browsers settings
3.2.10.4 Outputs: The system will display the language chosen by the user
3.2.10.5 Error Handling: The system will default to English if the translation is not available

3.2.11 Unique Tickets

3.2.11.1 Introduction: Tickets are non-replicable and cannot be duplicated, each ticket is tied to individual transactions
3.2.11.2 Inputs: The system will prompt the user through the web interface to provide payment information
3.2.11.3 Processing: The system will create a unique ID for each transaction that will be linked to the ticket to prevent fraud and reusability
3.2.11.4 Outputs: The user will be provided with a downloadable ticket with a unique identifier
3.2.11.5 Error Handling: The system will detect invalid tickets and deny entry

3.2.12 San Diego Theater Chain

3.2.12.1 Introduction: The system will be available 20 theaters across San Diego
3.2.12.2 Inputs: The system will allow users to input the ZIP code or area name to determine which theater is optimal
3.2.12.3 Processing: The system organizes showtimes and ticket availability depending on the location of the user, it will handle location comparisons and provide real-time updates
3.2.12.4 Outputs: The system will display nearby locations with relevant information the user queried
3.2.12.5 Error Handling: Users will be shown an error message if they select an invalid location and suggest nearby theaters

3.2.13 Reserved Seating

3.2.13.1 Introduction: Users will be able to select and reserve seating for each showtime
3.2.13.2 Inputs: The system will display an overview of the seating in the theater, an interactive chart will be created for each showtime
3.2.13.3 Processing: The system will provide real-time availability for the seating chart; seats will lock once it has been taken and unlock if deselected
3.2.13.4 Outputs: The users will receive a confirmation of their reserved seats and will be shown on their order summary
3.2.13.5 Error Handling: The system will display a message to users if a seat is unavailable and will be prompted to select a different seat

3.2.14 Payment System

3.2.14.1 Introduction: Credit cards, PayPal, and Bitcoin are viable options
3.2.14.2 Inputs: During the checkout, the user will be prompted to select their payment method

3.2.14.3 Processing: Credit card transactions are processed through PCI DSS compliant gateway, PayPal is processed through its API, Bitcoin is processed through a secure crypto payment service
3.2.14.4 Outputs: The payment confirmation page will be sent to the user and as well as the receipt
3.2.14.5 Error Handling: The system will notify the user if the payment declines or is invalid and prompt the user to try a different payment method

3.2.15 User Registration
3.2.15.1 Introduction: Allow users to create accounts
3.2.15.2 Inputs: Ask user to provide an email address and password
3.2.15.3 Processing: The system will verify if that email is already in use or the password does not meet the requirements
3.2.15.4 Outputs: The user will get an email confirmation that their account has been created and then asked to sign in with their credentials
3.2.15.5 Error Handling: The system will show a message to the user that the email is in use, or the password does not comply

3.2.16 Refunds
3.2.16.1 Introduction: Customers can refund their tickets up to 2 hours prior to the movie showtime
3.2.16.2 Inputs: Refunds are done in person at the kiosk
3.2.16.3 Processing: The barcode from the confirmation email is scanned and will be retrieved by the system to verify that it is valid
3.2.16.4 Output: The system will send the amount back to the original payment method
3.2.16.5 Error Handling: If the user's barcode is invalid then the refund cannot be refunded, but the credentials are correct then they will receive instore credit

3.3 Use Cases

3.3.1 Use Case #1: Creating an Account

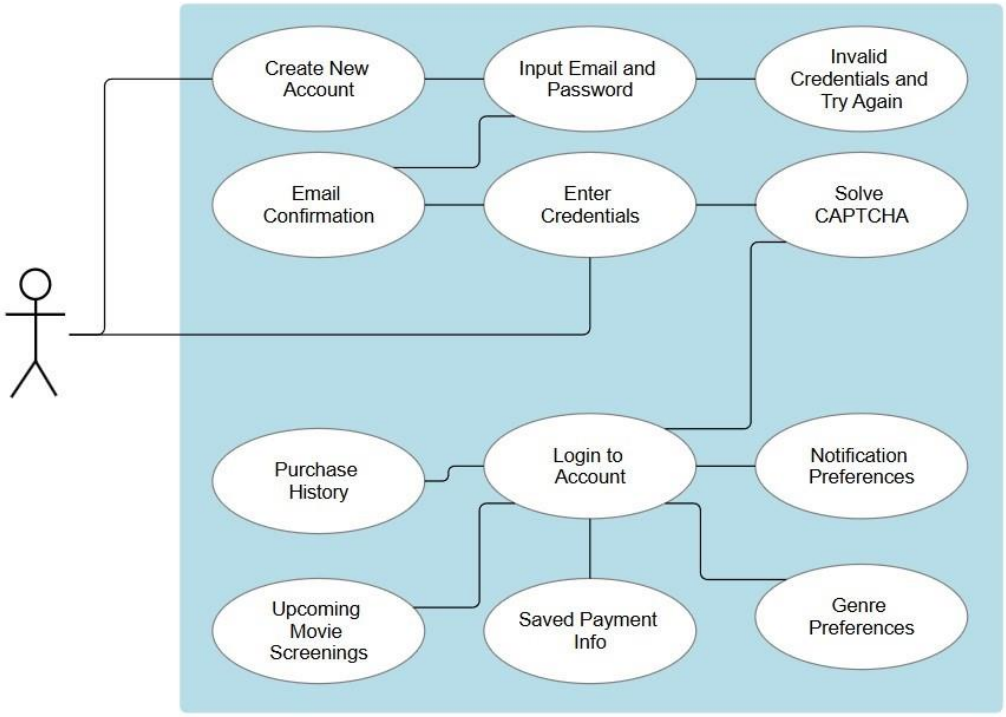
3.3.1.1 Pre-Conditions:

- Customer has device connected to the internet and browser open

3.3.1.2 Basic Flow:

1. When the user first enters, they will be prompted to create an account or login to their existing account
2. New account
    a. They will be asked to input their email which is also their username and password
        i. The email will be verified by the system that it is not already in use
        ii. The password must contain at least one uppercase letter
        iii. The password must contain at least one lowercase letter
    b. Then they will receive an email confirmation and be asked to sign into their new account
3. Enter Credentials
4. Solve CAPTCHA
5. Login to Account
    a. Once in their account they are redirected to the settings page
        i. Change notification preferences

ii. Change payment info
iii. Change genre preferences
iv. View purchase history
v. View upcoming movie screenings they have purchased



3.3.1.3 Alternative Flows:

- If the user inputs an invalid email or password when first creating an account, then the system will output an error message telling the user to try again
- If the user enters invalid credentials when they already have an account, then the system will output a message telling the user either the email or password is incorrect
- If the user fails to solve the CAPTCHA after 3 attempts, they will be locked out for 10 minutes

3.3.1.4 Post- Conditions:

- The user successfully created an account
- The user can configure their settings to their liking

3.3.2 Use Case #2: Buying a Ticket

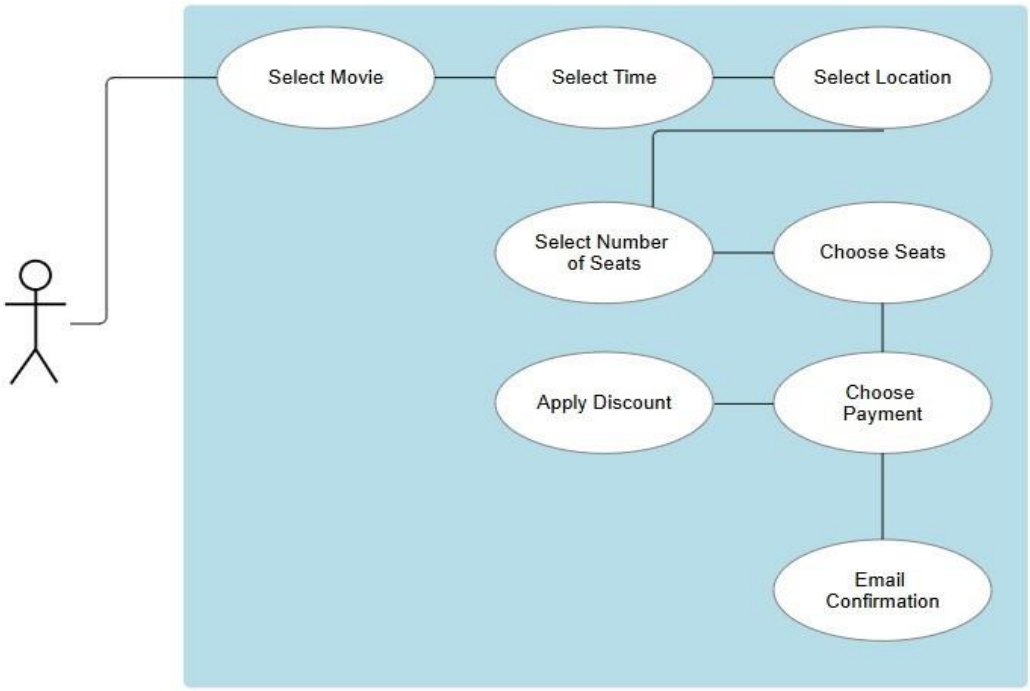
3.3.2.1 Pre-Conditions:

- Customer has device connected to the internet and browser open
- Customer has created an account and is logged in

3.3.2.2 Basic Flow:

1. The user first chooses which movie they want to watch
2. Then chooses from the available times
    a. The system will automatically detect if the listings are withing the 2-week prior period or 10 minutes after and only list those
3. Then they will choose from one of the 20 theaters located in San Diego
    a. By entering their ZIP code or area, it will display which locations are closest
4. Then choose the number of tickets/seats

    a. Each customer is limited to 20 tickets
5. Then choose which seats
    a. It will show the user how many tickets left they have
    b. It will tell the user to choose more seats if they have not used them all
6. Apply discount if applicable
    a. Student
    b. Military
    c. Senior
7. Choose payment method
    a. Credit card
        i. Visa, Mastercard, Discover, Amex
    b. PayPal
    c. Bitcoin
8. Customer gets an email confirmation containing
    a. Movie title, showtime, location, seat numbers, and payment receipt
    b. It will also contain a barcode that the employee at the kiosk will scan to admit the customer



3.3.2.3 Alternate Flows:

- If the user enters a ZIP code or area that is beyond 50 miles from one of the available locations, the system will tell the user that they are out of range
- If the use attempts to choose more than 20 tickets the system will alert them that the maximum number of tickets is 20
- If the user does not have valid proof of verification for the discount, then the system will alert the user that the discount is not applicable
- If the user's payment method is declined for insufficient funds or invalid method, then the system will alert the user that the payment was not successful

3.3.2.4 Post-Conditions:

- Customer gets an email confirmation of their movie showtime purchase
- Payment was successful

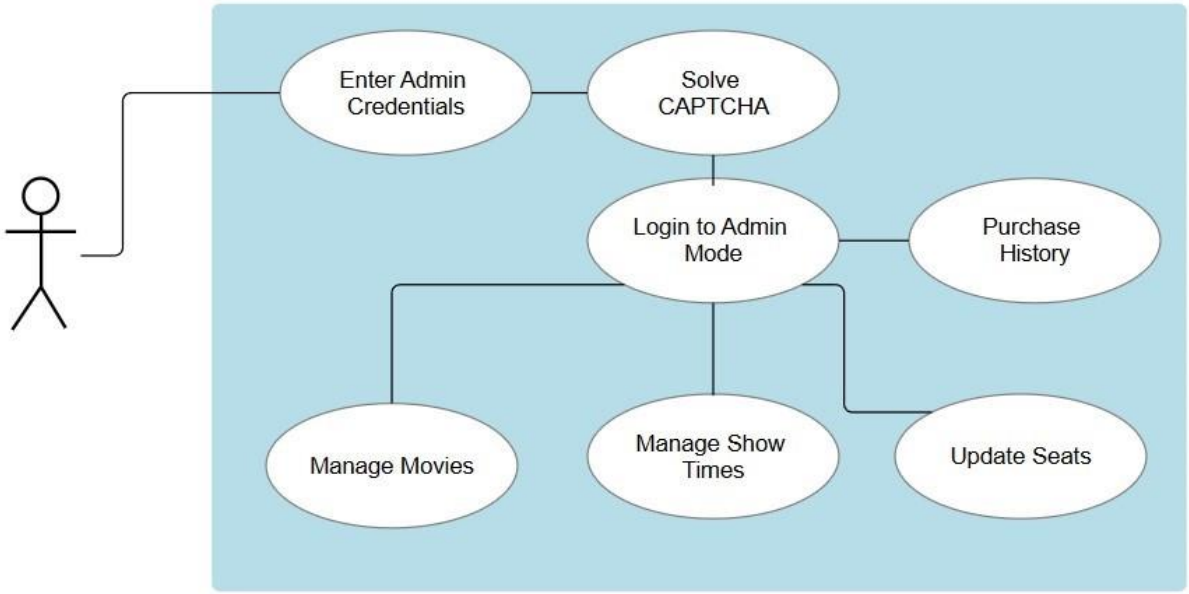
3.3.3 Use Case #3: Administrator Login

3.3.3.1 Pre-Conditions:

- User has device connected to the internet and browser open

3.3.3.2 Basic Flow:

1. Similar to logging into a normal account, the admin will be prompted to enter credentials and solve CAPTCHA
2. Administrator Privileges
    a. Manage movies such as upload and remove
    b. Manage show times such as upload and remove
    c. Manage seats such as if a customer has a request for a change
    d. View and manage purchase history from customers



3.3.3.3 Alternate Flows:

- If the user inputs an invalid email or password when first creating an account, then the system will output an error message telling the user to try again
- If the user enters invalid credentials when they already have an account, then the system will output a message telling the user either the email or password is incorrect
- If the user fails to solve the CAPTCHA after 3 attempts, they will be locked out for 10 minutes

3.3.4 Use Case #4: Refunds

3.3.4.1 Pre-Conditions:

- Customers will need to login to their account
- Customers will be at the theater with their movie confirmation email ready
- It is 2 hours before the showtime of the movie

3.3.4.2 Basic Flow:

1. Ask an employee at the kiosk for a refund for a ticket they bought

2. The employee will ask for proof of purchase from the account
    a. Scan the barcode/transaction id
3. The refund will be sent to the original form of payment
    a. Can take up to 5-7 business days
4. A refund receipt will be sent to the customers email and printed out

3.3.4.3 Alternate Flows:

- The barcode is invalid because it has passed the 2-hour grace period
- The payment method is now invalid so the refund will be instore credit

3.3.4.4 Post-Conditions:

- The refund is successful, and the customer receives their money back into their account
- The customer receives an email confirmation, and it is also shown on their purchase history on their account

3.3.5 Use Case #5: Customer Feedback

3.3.5.1 Pre-Conditions:

- Customers will need to login to their account and email
- Customer has device connected to the internet and browser open
- Customer has finished watching the movie

3.3.5.2 Basic Flow:

1. After the customer watches the movie; they will receive an email asking them for a review of their viewing experience
    a. Rate from 1 to 5 stars
    b. Comments
2. The user will receive a confirmation email once their review has been completed
3. The admin will collect and review the reviews

3.3.5.3 Alternate Flow:

- If the customer posts an inappropriate review, then the admin will flag, report, and delete the post

3.3.5.4 Post-Conditions:

- The customer refreshes their page and can see that their review has been posted
- Other users can see the reviews and the system

3.4 Classes / Objects

3.4.1 User Class

3.4.1.1 Attributes

- Email: Email ID of the user
- Password: Password string that the user sets
- Name: First and Last name of user
- PreferredLanguage: English, Spanish, Swedish
- PaymentMethods: Saved payments of user
- NotificationPreferences: Different notifications settings
- GenrePreferences: Different genres

3.4.1.2 Functions

- Register: Create a new account
- Login: Log into existing account
- SetPreferredLanguage: English, Spanish, Swedish
- SetPaymentMethods: Change payment methods
- GetPurchaseHistory: Returns purchase history
- SetNotificationPreferences: Change notifications
- SetGenrePreferences: Change genres

3.4.2 Transaction Class

3.4.2.1 Attributes

- TotalAmount: Total amount paid during transaction
- ConfirmationEmail: Email associated with customer
- PaymentMethod: Credit card, PayPal, Bitcoin

3.4.2.2 Functions

- CreateTransaction: Sets up transaction
- ProcessPayment: Process the payment of user
- SendConfirmationEmail: Sends confirmation email to user

3.4.3 Movie Class

3.4.3.1 Attributes

- Title: Movie title
- Genre: Genre of movie
- Description: Description of the movie
- Rating: Rating of the movie from third party sources

3.4.3.2 Functions

- AddTitle: Creates a new movie
- RemoveTitle: Removes a movie
- UpdateMovie: Modify movie info

3.4.4 Theater Class

3.4.4.1 Attributes

- Theater: Name of theater
- Address
- City
- ZipCode

3.4.4.2 Functions:

- AddTheater: Creates theaters in San Diego area
- RemoveTheater: Removes a theater
- UpdateTheater: Modify theater info

3.4.5 Discount Class

3.4.5.1 Attributes

- Discount: Discounts for students, military, senior
- DiscountPercent: Percentage taken off total amount

3.4.5.2 Functions

- SetDiscount: Sets the discount rate for the transaction
- CreateDiscount: Creates new discount

3.4.6 Admin Class

3.4.6.1 Attributes

- AdminUser: Admin username
- AdminPassword: Admin password

3.4.6.2 Functions

- UploadMovie: Add a movie
- DeleteMovie: Remove a movie
- UploadShowtime: Add a showtime
- RemoveShowtime: Remove a showtime
- UpdateSeats: Change seat for customer
- ViewPurchaseHistory: Display purchase history of customer
- LoginAdmin: Logs into admin account

3.4.7 Tickets Class

3.4.7.1 Attributes

- TicketID: Unique id for each ticket
- NumberOfTickets: Number of tickets customer wants to purchase
- SeatID: The specific seat chosen
- TheaterID: The specific theater chosen
- MovieID: The specific movie chosen

3.4.7.2 Functions

- CreateTicket: Creates a ticket for the customer to purchase
- CreateBarcode: Creates barcode for the ticket

3.4.8 Refunds Class

3.4.8.1 Attributes

- OriginalTicketID: Original ticket ID
- ReturnAmount: Refund amount
- ReturnMethod: Original payment, store credit

3.4.8.2 Functions

- CreateRefund: Creates a refund request
- ProcessRefund: Executes refund to customer
- RefundConfirmation: Sends refund receipt to customer

3.5 Non-Functional Requirements

3.5.1 Performance

- 95% of web page request are completed with in less than a second
- 99% of transactions are completed within 3 seconds from payment submission to confirmation
- System can support up to 1000 current users without fault

- Updates to seat availability and showtimes are shown within 3 seconds of any change

3.5.2 Reliability

- The Mean Time Between Failure is greater than 30 days
- The Mean Time to Repair is less than 4 hours
- Backup systems ensure it is automated every hour
- Centralized database maintains consistency with all 20 locations to ensure delays do not exceed 2 seconds
- System has 99% transaction success rate or higher

3.5.3 Availability

- System shall maintain 99.5% uptime annually
- Users will be notified of scheduled maintenance at least 24 hours in advance
    - Maintenance will occur during non-peak hours (2:00 AM – 6:00 AM)
- System will restore to full functionality within 1 hour in the event of a failure
- Ticket purchasing is available 24/7 with 99% uptime

3.5.4 Security

- User authentication will be verified within 2 seconds upon valid credentials
- Failed login attempts are locked out after 3 attempts for 10 minutes
- The web browers utilizes HTTPS for encryption between clients and servers
- All payments comply to PCI DSS standards
- CAPTCHA has 99% success rate for valid users
- Customer data is encrypted in transit and at rest with AES-256
- Each ticket shall have a unique identifier with 99% success in fraud prevention

3.5.5 Maintainability

- The system updates with zero downtime using a blue-green deployment
- The API is version controlled and backwards compatible
- Transactions are logged and retained for a minimum 1-year periods
- System health is monitored, and alerts are detected in less than 2 minutes

3.5.6 Portability

- The system is compatible with 95% of web browsers(Chrome, Firefox, Safari, Edge)
- Optimized for mobile devices as well
    - Screen resolution
    - Touch interfaces
    - Low power processing requirements
- The system shall support at least 95% accuracy for multilingual content presentation
- In-store kiosk uses the same ticking backend system to ensure consistency

3.6 Inverse Requirements

*State any *useful* inverse requirements.*

3.7 Design Constraints

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

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.

3.9 Other Requirements

Catchall section for any additional requirements.

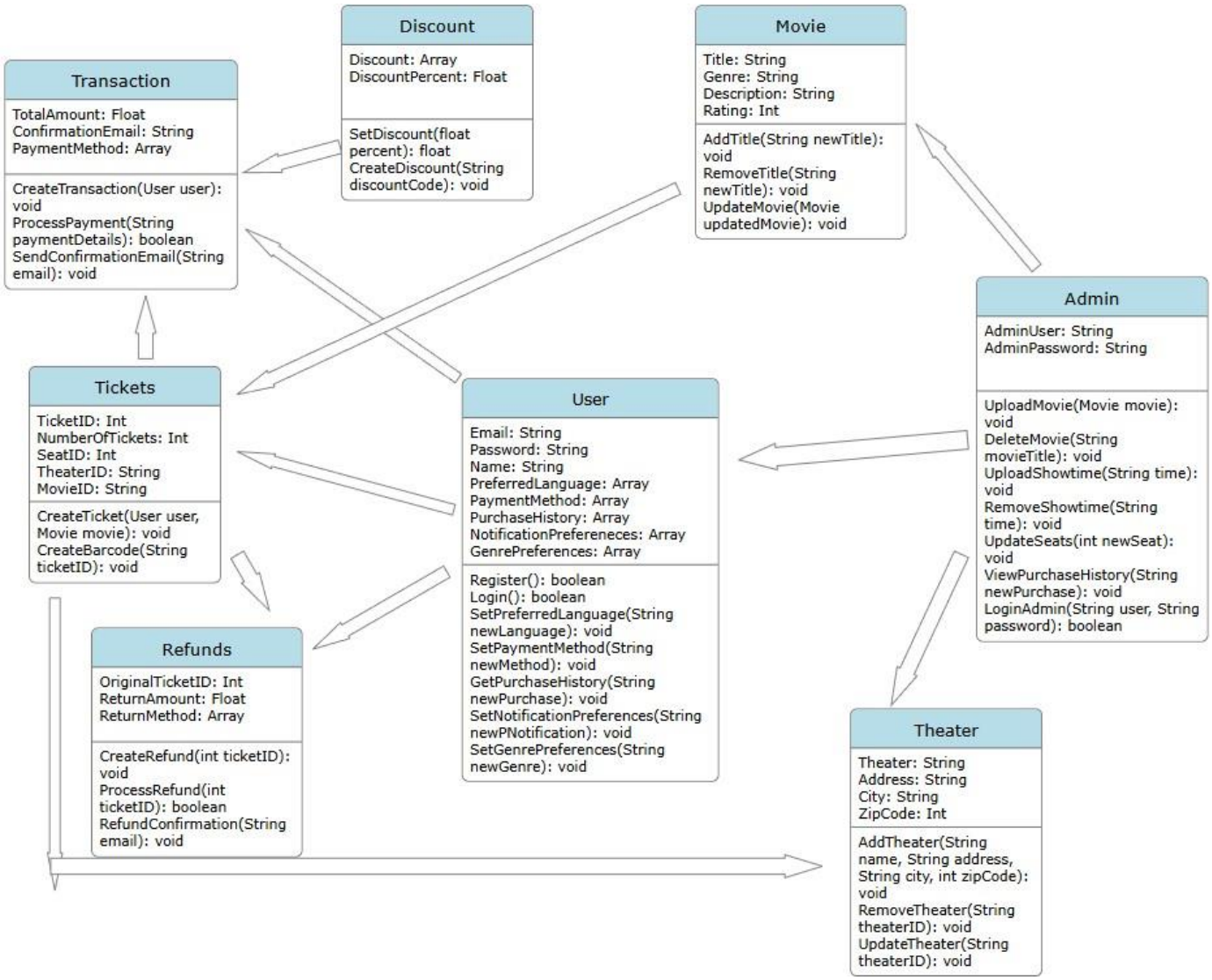
4. Analysis Models

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 Description

The Software Requirement Specification outlines the comprehensive design and implementation of the Movie Theater Ticketing System. It defines a web-based solution designed for ticketing infrastructure for a chain of 20 theaters across the San Diego County. This system can support up to 1000 concurrent users with scalability. Additionally, it allows customers to browse movies, select showtimes, reserve seats, and process secure payments. With a variation of payment methods; credit card, PayPal, and Bitcoin. Other key features include multilanguage support, real-time seat availability updates, and select discounts. The system supports both customer and administrator roles, with customers managing their booking through a user-friendly GUI and administrators controlling backend operations such as movie listings, showtime management, and transaction oversight. It incorporates security standards like HTTPS encryption, PCI DSS compliance, and unique ticket identifiers. The system ensures 99.5% uptime with backup systems and centralized database management across the theaters.

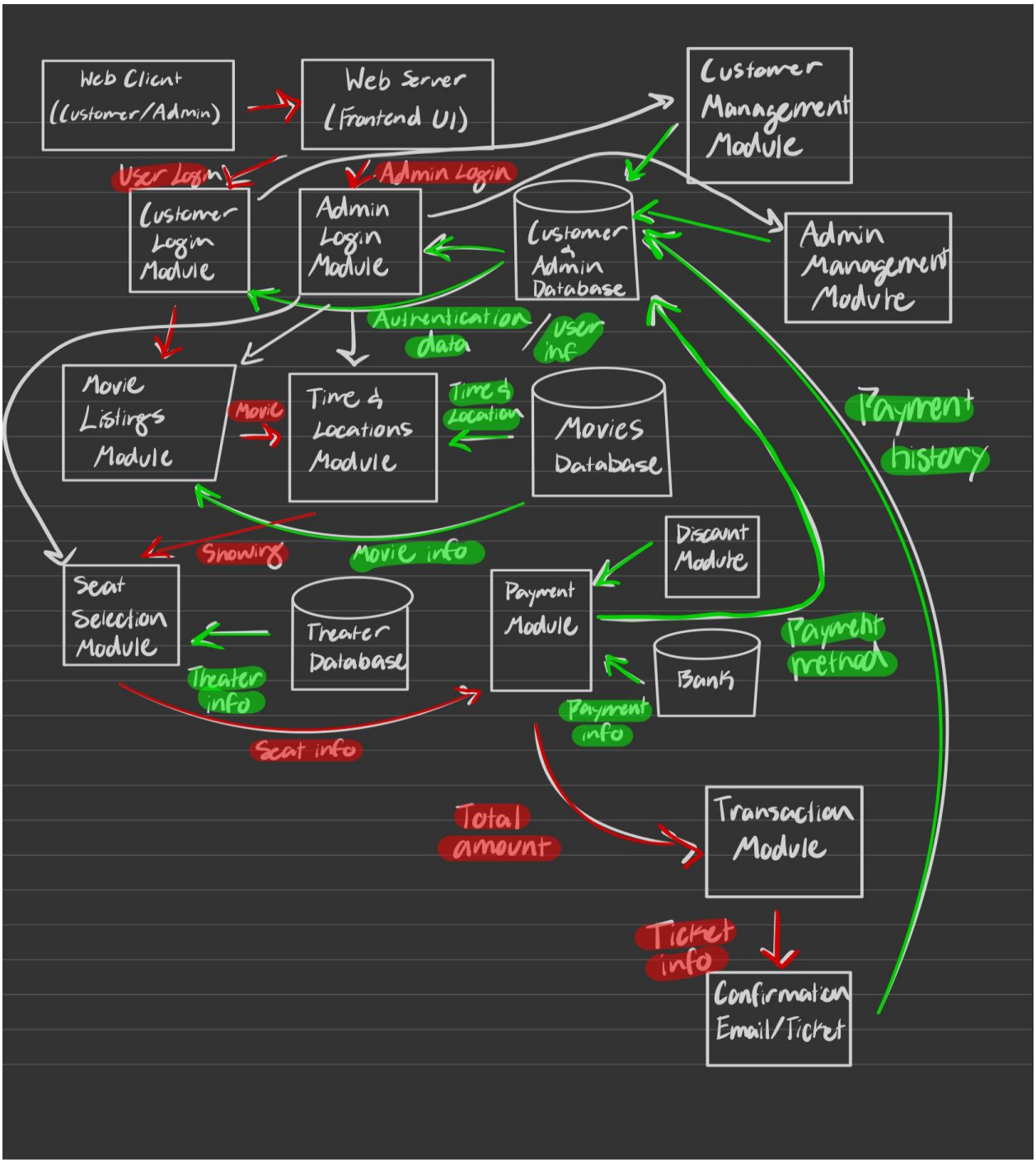
4.2 UML Diagram and Explanation



User: This represents the user who interacts with the application to browse and purchase movie tickets.
Attributes: This includes the credentials such as *Email* and *Password*, user preferences such as *PrefererredLanguage, NotificationPreferences, GenrePreferences,* and transaction data such as *PaymentMethod* and *PurchaseHistory*
Operations: Allows the user to register, login, update language and genre preferences, set payment methods, view purchase history, and modify notifications preferences
Admin: The administrator logs in using secure credentials and has broad control over theater and movie data.
Attributes: This stores the admin credentials *AdminUser* and *AdminPassword*
Operations: Allows the Admin to upload, delete, and update movies, modify seating information, view users' purchase histories, and perform login authentication
Movie: This holds the details of the individual movies available in the system
Attributes: This includes the *Title, Genre, Description*, and *Rating*
Operations: Allows updating movie data such as the title, modifying details, and removing movies from the catalog
Tickets: Tracks tickets and links them with specific movies, theaters, and users

Attributes: This holds the ticket data such as the *TicketID, NumberOfTickets, SeatID, TheaterID,* and *MovieID*
Operations: Allows the creation of tickets for a user and movie, it also generates a unique barcode associated with the user
Transactions: This represents the transaction record of a purchase made by the user
Attributes: This holds the *TotalAmount* paid, sends the *ConfirmationEmail*, and retrieves the *PaymentMethod* used
Operations: Allows transactions to be created, processes payments, and sends confirmation emails after the purchase
Discount: This manages the valid discounts that can be used during transactions
Attributes: This holds the list of discounts and their *DiscountPercent*
Operations: Allows the creation of a new discount and update discount values for the transaction
Refunds: This is where the user can initiate a refund which can be processed and confirmed.
Attributes: This holds the *OriginalTicketID, ReturnAmount*, and the *ReturnMethod*
Operations: Allows the initiation and processing of a refund, also it sends the confirmation of the refund status
Theater: This stores the physical location of the theaters, which includes the name, address, city, and zip code.
Attributes: This holds the information such as the *Theater* name, *Address, City*, and *ZipCode*
Operations: Allows adding of new theaters, updating existing ones, and removing theaters from the system

4.3 Software Architecture Diagram and Explanation



Web Client(Customer/Admin): This provides browser-based access to the system via desktop or mobile.
Web Server(Frontend UI): This allows the user to interact with an interface which sends and receives data to and from backend servers.
Customer Login Module: Allows the customer to login to their account.
Admin Login Module: Allows the administrator to login to their account and manage movies, showtimes, seats, and view purchase histories.

Customer and Admin Database: For the customer it holds their payment information, purchase history, preferences, and language. For the admin, it holds management information for movies, showtimes, seats, and purchase histories.
Customer Management Module: This allows the user to manage their payment information, purchase history, preferences, and language.
Admin Management Module: This allows the user to manage movies, showtimes, seats, and view purchase histories.
Movie Listing Module: This allows the user to browse and search for movies.
Time and Location Module: This displays the available times and filter locations by ZIP code or area.
Movies Database: This holds the real-time showtimes and updates availability across the 20 theaters.
Seat Selection Module: This allows the user to select the number of seats and specific seats they want.
Theater Database: This holds the layout details and seat availability of each theater.
Payment Module: This collects payment methods, verifies it, and ensures PCI DSS compliance
Discount Module: This connects to the payment module and verifies the eligibility of the student, military, or senior discount
Bank: This is the payment information of the user.
Transaction Module: This creates and confirms ticket transactions.
Confirmation Email/Ticket: This sends a digital receipt and a unique barcode ticket to the user's email confirming their purchase
Following the Red Arrows:
User/Admin Login: From the frontend UI, the user uses their credentials to login to the system.
Movie: The movie selection synchronizes with the database to show available showtimes and locations.
Showing: From the time and location selection, the theater database shows the available showtimes for the movie.
Total Amount: The payment module sends information to the transaction module to calculate the total amount.
Ticket Info: The transaction then initiates the Confirmation/Email ticket to send that info to the user.

4.4 Development Plan and Timeline

Week 1-2 Planning and System Design: Finalize the SRS, design UI mockups, create UML diagrams, define classes
Week 3-4 Frontend and Backend Setup: Build basic frontend (user and admin interfaces). Set up backend server and database
Week 5-6 Functionality Implementation: Browsing, ticket booking, showtimes, seating, movie selections, etc.
Week 7-8 Admin Features, Bot Prevention: Add admin login and management controls. CAPTCHA
Week 9-10 Payment, Reviews, Feedback: Add credit cards, PayPal, Bitcoin, movie reviews, and customer feedback
Week 11-12 Testing and Deployment: Conduct full system testing, security testing, finalize documentation

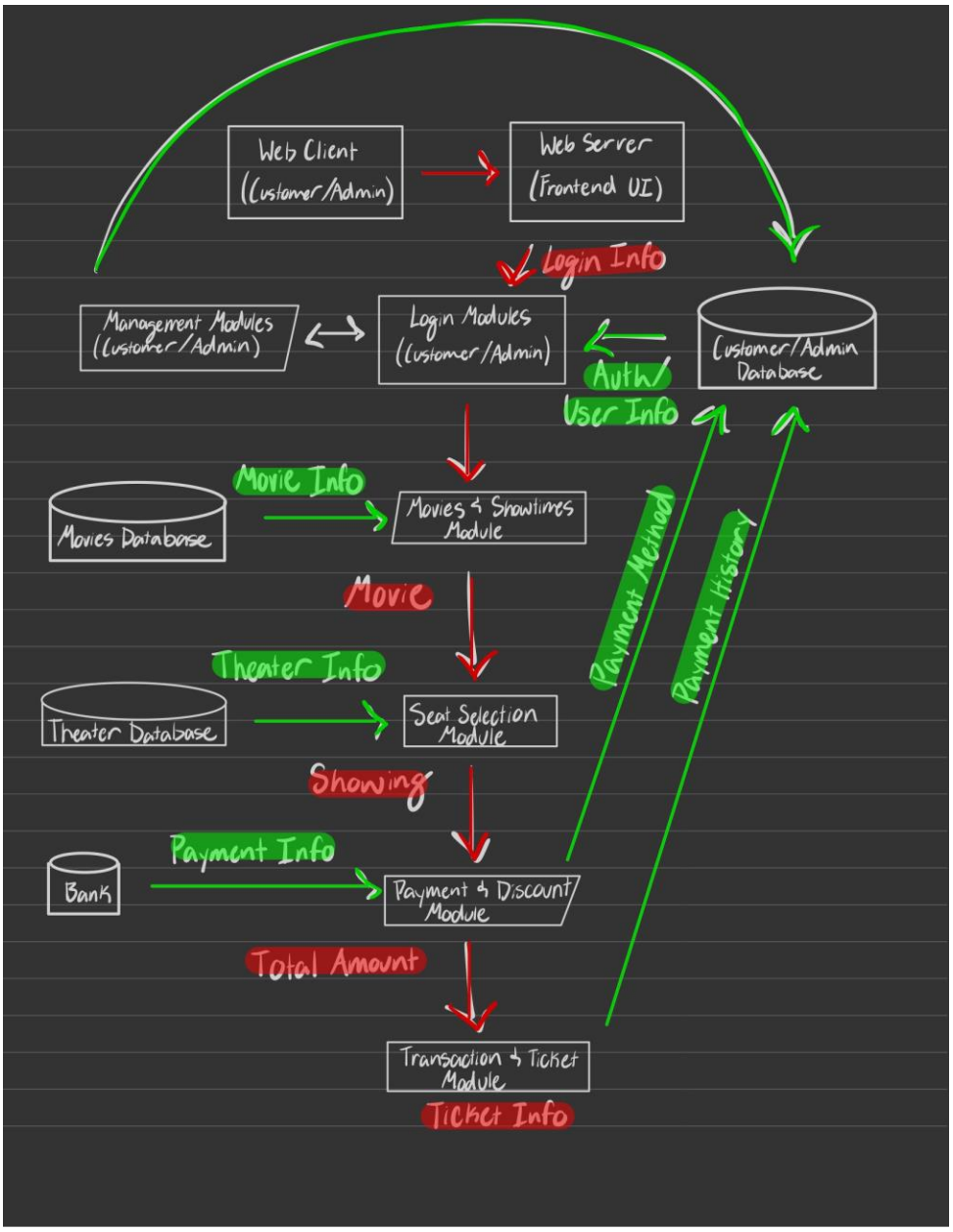
5. Test Plans

https://github.com/ethanloun/CS250-SRS.git

UML diagram and description includes appropriate content. Planning to update SWA diagram and description in next section/assignment (Architecture Design with Data Management).

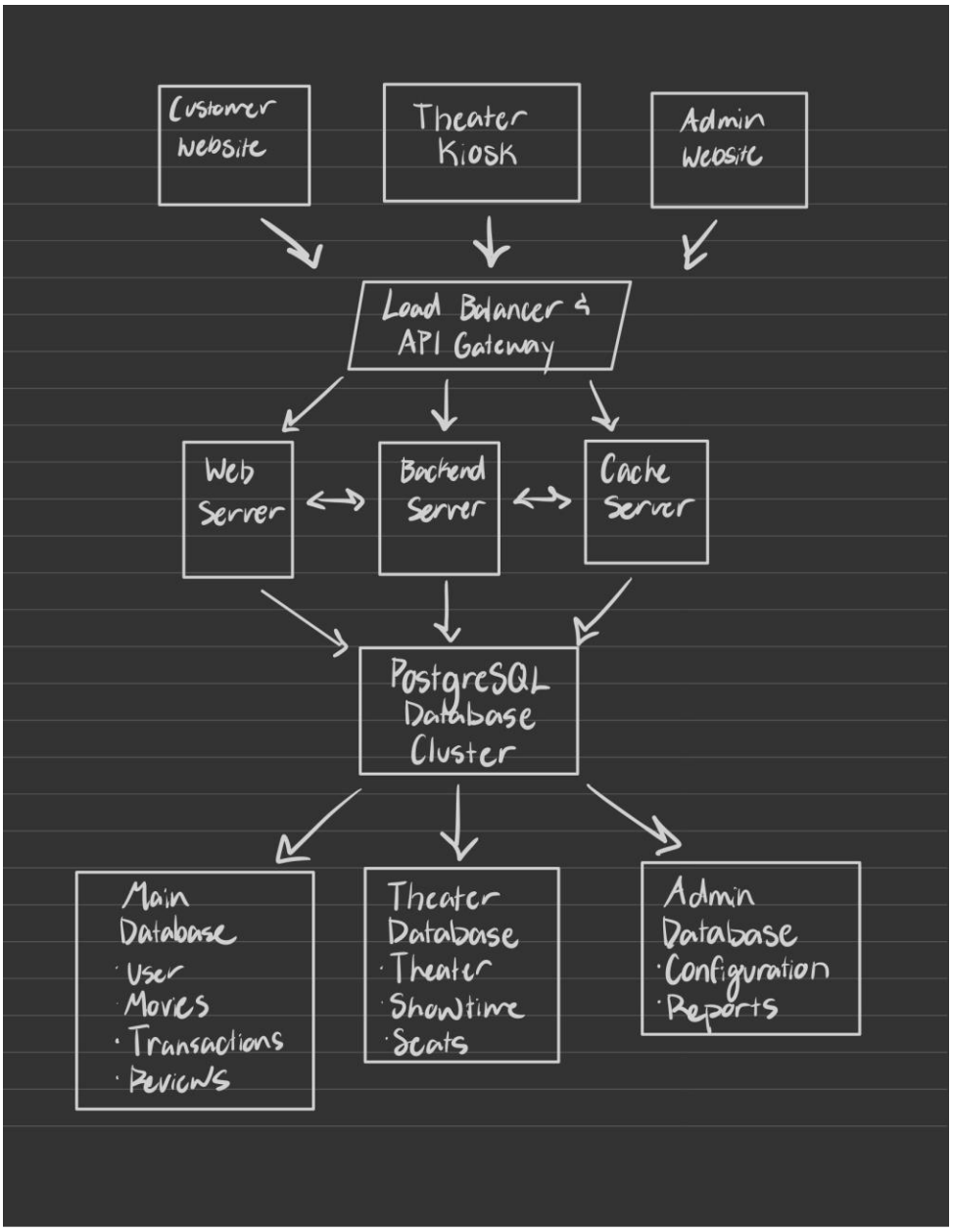
6. Architecture Design with Data Management

6.1 Updated Software Architecture Diagram and Explanation



Description remains mostly the same. For simplicity, the Customer and Admin Login Module has been combined into one, and the Management Module. The Payment and Discount have been combined into one. The Transaction and Ticket Module has been combined into one, too.

6.2 Data Management Strategy Diagram and Explanation



At the top are the Customer, Admin, and Theater website interfaces. The Load Balancer and API Gateway acts as the central hub which handles traffic distribution, authentication, and ensures optimal performance. The middle tier contains three interconnected servers. The Web server manages the interface and static content. The Backend manages logic, transactions, and data validation. The Cache Server provides access to frequently requested data. The PostgreSQL Cluster partitions and organizes the data into three databases. The Main Database maintains control over data, including user accounts, movie catalogs, transactions, and customer reviews. The Theater Database has location specific data such as the theater name, showtimes, and seats. The Admin Database handles operational data such as configuration and reports.

6.3 Tradeoff Discussion

6.4.1 SQL: The system uses PostgreSQL. The advantages of this database are its ACID compliance, integrity, and structure.

6.4.1.1 ACID:

Atomicity: This is the all or nothing function, where the system will complete the entire task or do nothing at all so that there are no partial bookings or errors

Consistency: This helps to ensure the real-time seat availability is accurate and follows the rules

Isolation: This ensures that there is no interference between users such as the seat selection module where someone cannot take your seat if you already have it reserved

Durability: This ensures that the payment confirmations and tickets are permanently saved even if there is a system crash

6.4.1.2 Integrity:

This ensures that each ticket is unique and cannot be duplicated, blocks invalid inputs such as attempting to purchase over 20 tickets, and changes are saved if everything follows the rules which prevents the system from being left in a broken state

6.4.1.3 Structure:

The modules have relational structure such as the users can have multiple transactions, the transactions contain tickets, the tickets are linked to specific movies, theaters, and showtimes.

6.4.2 non-SQL: The tradeoffs of not using a non-SQL database such as MongoDB include its flexibility and scalability. It can store different data shapes and spread data across multiple servers. This is better for unstructured data. But our system needs precise control, strong data links, and strict rules.

6.4.3 Single Database vs Multiple Databases:

This system uses a PostgreSQL Cluster with three partitioned databases: Main, Theater, Admin. The separation ensures that each database has a distinct function. Since the Admin database separates from customer data this increases security and maintains access control. Furthermore, the partitions aid with performance and load balancing. The tradeoffs of not using a single database are its simplicity to set up and maintenance. They are also easier to update, lower costs, and have faster development times. But less does not always mean better.

6.4.4 Table Contents: Allows for managing multiple types and layers of data

User: Stores information about the customer. This allows for specific preferences, enables login and checkout features.

Admin: Stores information about the admin. This allows for management of tickets and movies.

Tickets: Tracks the tickets that are sold. This links to the user which prevents fraud and ensures validation.

Movies: Stores information about the movie. Allows browsing and links movies with theaters and showtimes.

Theaters: Stores physical location of the theater. Allows users to search by area and ZIP code, also links to seats and showtimes.

Showtime: Tracks the movie schedule for a specific theater. Shows what is playing and when, also links to seats and tickets.

Seats: Stores the seat layout and availability. Allows for real-time seating updates.

Transactions: This logs the payments and confirmations for the users. This links to the user, discounts, and refunds.

Discounts: Defines valid discounts. This tracks and allows the correct discount amounts.
Reviews: Stores customer feedback. This allows the customers to review the site and improve engagement.
This system allows scalability, prevents redundancy, and the frequently accessed items are supplied by the cache server to improve performance. The tradeoff that can be provided by a JSON Blob is that it does not break up the data into individual columns. Which means that you can modify fields without changing the table structure. This is an all-in-one storage solution.