<Movicast>

Software Requirements Specification

<Version 1>

<2/1/2024>

<Group 10>
<Santiago Verdugo, Hung Quach, Nguyen Pham>

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

Revision History

Document Approval

Date	Description	Author	Comments
<02/01/24>	<Version 1>	<Name>	<First Revision>
		Santiago Verdugo	
		Hung Quach	
		Nguyen Pham	

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

1.1 Purpose

The system allows customers to buy movie tickets from various movie theaters. Provided as an online service, the system allows users of all ages to find and buy tickets of movies that are currently available in theaters.

1.2 Scope

The product is an online movie ticket booking system called Movicast. Movicast will allow users to browse a catalog of past, current, and upcoming movies. Users will be able to book and purchase movie tickets entirely online. There will also be options to edit ticket orders as well as cancel them.

Movicast provides a quick and easy way to buy movie tickets no matter where users are. Whether at home or on the go, users will be able to access Movicast anywhere they are. When making a booking, a purchase, or a cancellation, Movicast will send users an Email confirmation of their decisions. In addition, there is also an option for users to receive automated replies on their mobile devices if they wish to do so.

1.3 Definitions, Acronyms, and Abbreviations

Contract: A legal binding document that serves as an agreement between two parties. An enforceable contract includes an offer from a supplier, the acceptance from the customer, and mutual consent between the two parties. It also lists the cost, requirements, and schedule for the product.

Customer: The person, or people, who pay for the product. Usually, but not always, the customer will specify the requirements for the product that they desire.

Kiosk: A small stand alone device that provides access to various information and services on a computer screen.

Supplier: The person, or people, who produce the product for the customer.

User: The person, or people, who operates or directly interacts with the product. In many cases, the user(s) and the customer(s) are not the same people.

1.4 References

IEEE Recommended Practice for Software Requirements Specifications, 20 October 1998, Software Engineering Standards Committee

Software Requirements Template, 20 October 2009, ShockForce Software Team

Theater Ticketing Requirements, Unknown

1.5 Overview

The rest of the SRS will provide a general description of the ticketing system, specific requirements for users, and non-functional requirements. The SRS will be organized respectively in the order of the mentioned features.

2. General Description

2.1 Product Perspective

This website will be similar to other movie ticket websites, such as that of Fandango, Atom Tickets, and MovieTickets. With similar functionality, users will be able to browse various catalogs, bookmark upcoming movies, add tickets into a cart where users can later view and checkout their purchases, and many more actions.

2.2 Product Functions

Users will be able to search for movies currently showing at a movie theater, as well as being able to purchase tickets for a certain movie at a specific showtime. The website will also provide movie information, such as reviews and critic quotes. Customers will have an option to leave feedback for changes/improvement, and the website will support administrator mode for an easier way to make changes to it.

2.3 User Characteristics

Users will be general audiences that are familiar with online shopping and the movie theater experience.

2.4 General Constraints

The website will be optimized to allow fast performance for users. The website will have strong security, providing protection for stored user information. Furthermore, the website will be able to handle different scales of traffic as well as being compatible with multiple types of web browsers.

2.5 Assumptions and Dependencies

To be able to make bookings and purchases on Movicast, users are expected to have reliable access to a wifi network. It is also assumed that users will be using newer generations of compatible devices, such as computers, laptops, and smartphones in order to assure quick

performance. Older generations of the mentioned devices can access Movicast, however, performance will vary and not be optimal.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

Movicast will be web-based and accessible via web browsers, such as that of Google Chrome, Firefox, and Internet explorer. It will also be accessible on mobile applications for both iOS and Android devices. Furthermore, a self-service kiosk interface will be accessible at the theater for on-site ticket purchases and seat selection.

3.1.2 Hardware Interfaces

Movicast will be available on Windows, Microsoft, and Apple devices. It can be accessed through the usage of computers, laptops, and smartphones. To guarantee the best performance, it is recommended to use technologies from newer generations, preferably from 2018 and after.

3.1.3 Software Interfaces

Movicast will work with the theater's scheduling and event management software to provide real-time show schedules, seating availability, and event information.

3.1.4 Communications Interfaces

Integration with email and SMS gateways allows consumers to get booking confirmations, reminders, and promotional messages.

3.2 Functional Requirements

3.2.1 <Functional Requirement or Feature #1>

3.2.1.1 Introduction
Movicast will be able to handle above 100,000 users at any given time.

3.2.1.2 Inputs
Users will be able to request browsing and purchasing movie tickets. There will be options to search tickets via name, popularity, alphabetical order, and more.

3.2.1.3 Processing
The system will process user requests, manage ticket inventory, and update the database.

3.2.1.4 Outputs
Displaying available tickets, confirming purchases, and providing electronic tickets.

3.2.1.5 Error Handling
Handling errors such as payment failures, invalid inputs, and system crashes.

3.3 Use Cases

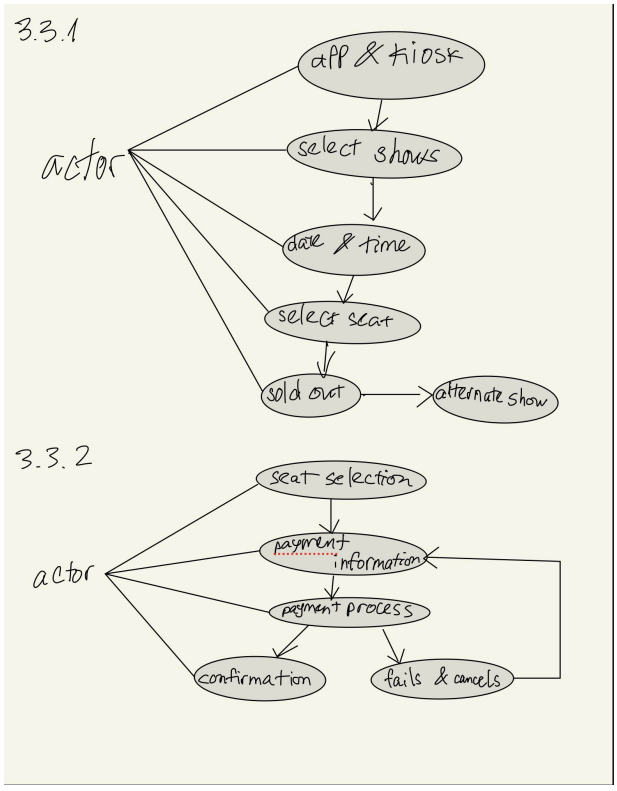
3.3.1 Use Case #1: Browsing movies and seat selection

Description: Users can browse movies and each movie with detailed information like time, date, duration as well as choosing seats from an interactive theater map.
Actors: Customers, users, audiences
Basic Flow: Customers use the mobile app or go through theater's website or purchase from self-service kiosks. The "Select Shows" section on the screen displays available shows for viewing and purchasing. The system will display all information on selected shows, which include the date and time of showtime, duration of the movie, and more. An interactive theater then will appear for customers to select seats. The system will update all user information and display total price.
Alternate Flow: In case of sold out, the system will recommend alternate shows.

3.3.2 Use Case #2: Purchasing tickets

Description: After selecting the movie, users can securely purchase tickets. After payment is confirmed, the system will send booking information to users.
Actors: Customers, Users
Basic Flow: After seat selection, the system will direct to checkout. System will ask for payment information. The system will process payment and return the user to booking confirmation if the payment is successful.
Alternate Flow: If the payment transaction fails, the system will have alternative payment options. In case the user cancels the transaction, the system will return to the payment step.

Diagrams for both 3.3.1 and 3.3.2



3.4 Classes / Objects

3.4.1 <Class / Object #1>

User

3.4.1.1 Attributes
Username
User information: Unique ID, first and last name, phone number
Payment information
User settings
Preferred theater
Location

3.4.1.2 Functions
Change username
Change account info
Select preferred theater

3.4.2 <Class / Object #2>

Movie

3.4.2.1 Attributes
Name
Runtime
Cast
Reviews
Showtimes
Media (Poster/Trailer)

3.4.2.2 Functions
Select movie
Edit movie details (Admin)
Add media (Admin)

3.4.3 <Class / Object #3>

Theater

3.4.3.1 Attributes
Name
Location
Current showings
Capacity

3.4.3.2 Functions
Select theater
View showtimes
View available seats
Select seats

3.4.4<Class / Object #4>

Ticket

3.4.4.1 Attributes
Movie name
Showtime
Runtime
Seat number
QR Code?

3.4.4.2 Functions
View ticket
Refund
Security access

3.5 Non-Functional Requirements

3.5.1 Performance

Movicast is guaranteed to have a fast running speed. While using Movicast, whether it is searching the catalog or making transactions, all actions will be processed in under a second. If issues were to occur, the system will be down no more than 30 seconds until said issues are fixed

3.5.2 Reliability

Movicast will be up to date with all available ticket options. All movie tickets shown in the catalog are legally authorized and licensed by movie labels and organizations.

3.5.3 Availability

Movicast will be online twenty-four hours a day. Users will be able to access it as long as they have an account. Like many applications, the system will also require users to have an internet connection.

3.5.4 Security

User data, such personal information, credit card information, and password, will be encrypted and stored securely with up to date and reliable systems.

3.5.5 Maintainability

Movicast is easily maintainable with modular code and clear documentation.

3.5.6 Portability

Movicast will be compatible with major web browsers and operating systems, allowing users to access at home or on the go.

3.6 Inverse Requirements

Movicast will have access to ticket inventory unless authorization is provided. Furthermore, it will not allow access to or display user information to others.

3.7 Design Constraints

User Interface Consistency: Movicast's user interface design will be consistent across all platforms for ease of use and familiarity.

Data Security: Movicast uses up to date data encryption and security systems to protect sensitive customer information that complies with industry regulations.

Performance Optimization: Design the system to handle peak loads efficiently and minimize response times, considering hardware limitations such as server capacity.

Data Backup: Movicast does regular data backup in order to prevent data loss and ensure system reliability.

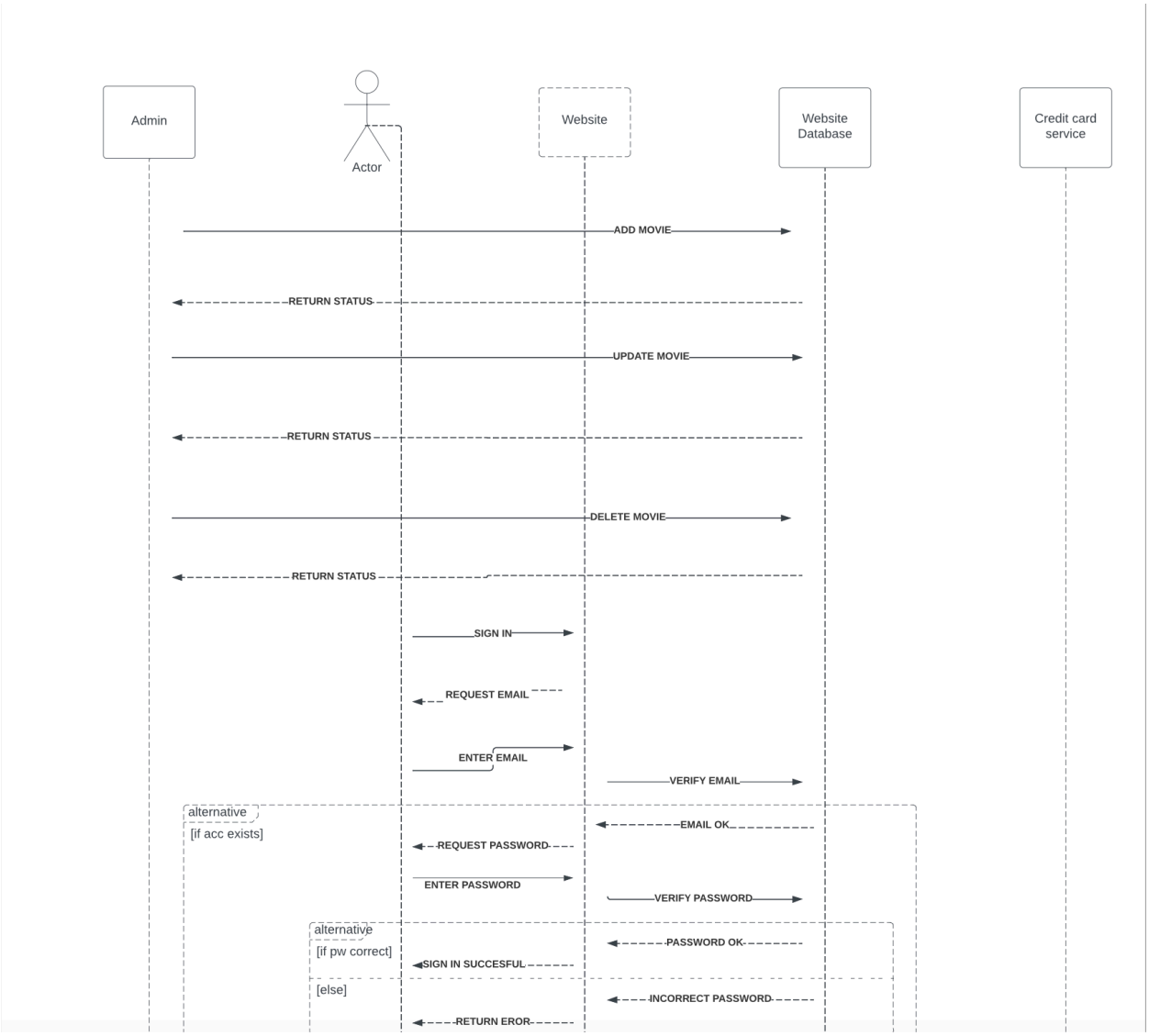
Error Handling: Movicast uses up to data error handling mechanisms in order to ensure fast recovery from errors that users may encounter.

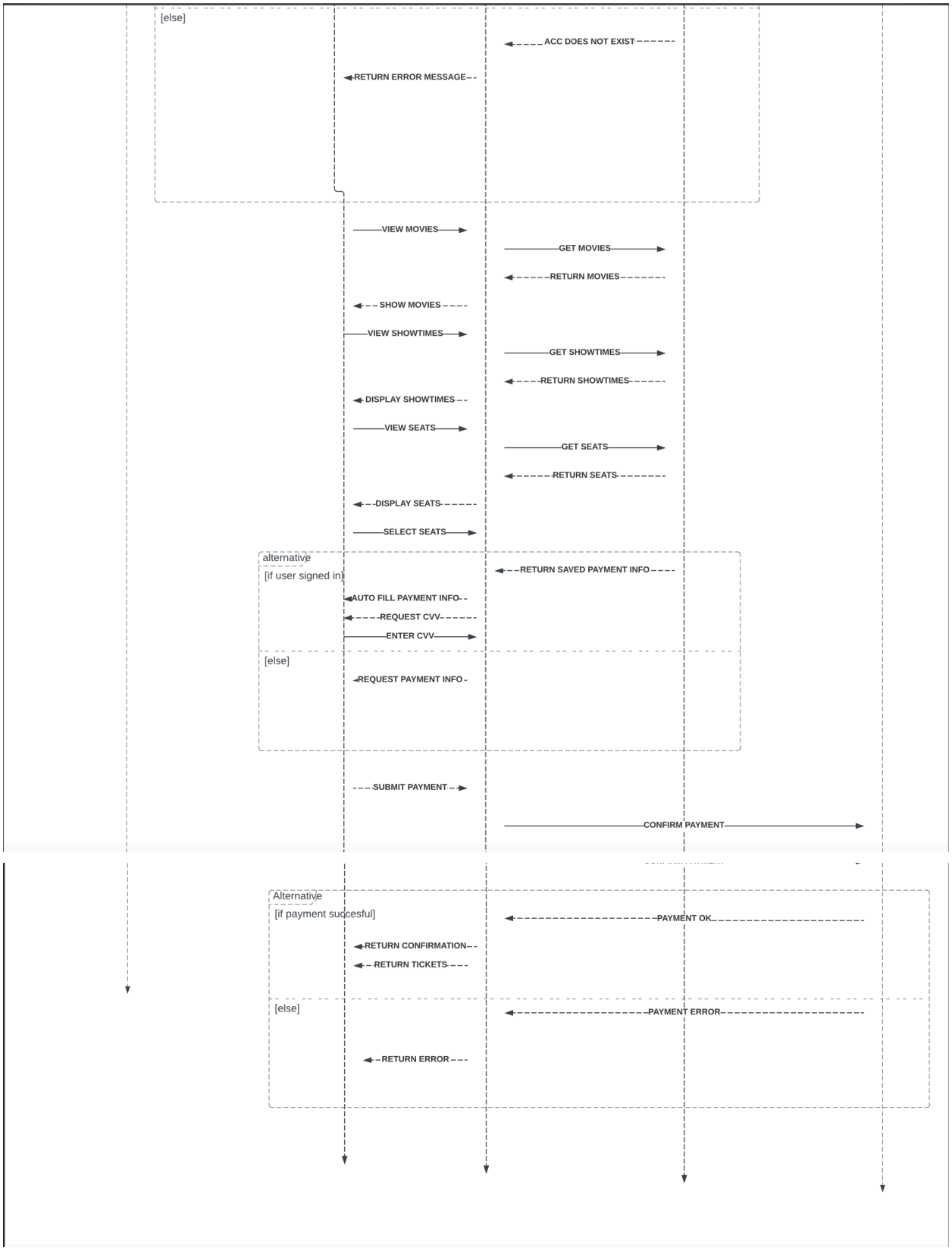
3.8 Logical Database Requirements

3.9 Other Requirements

4. Analysis Models

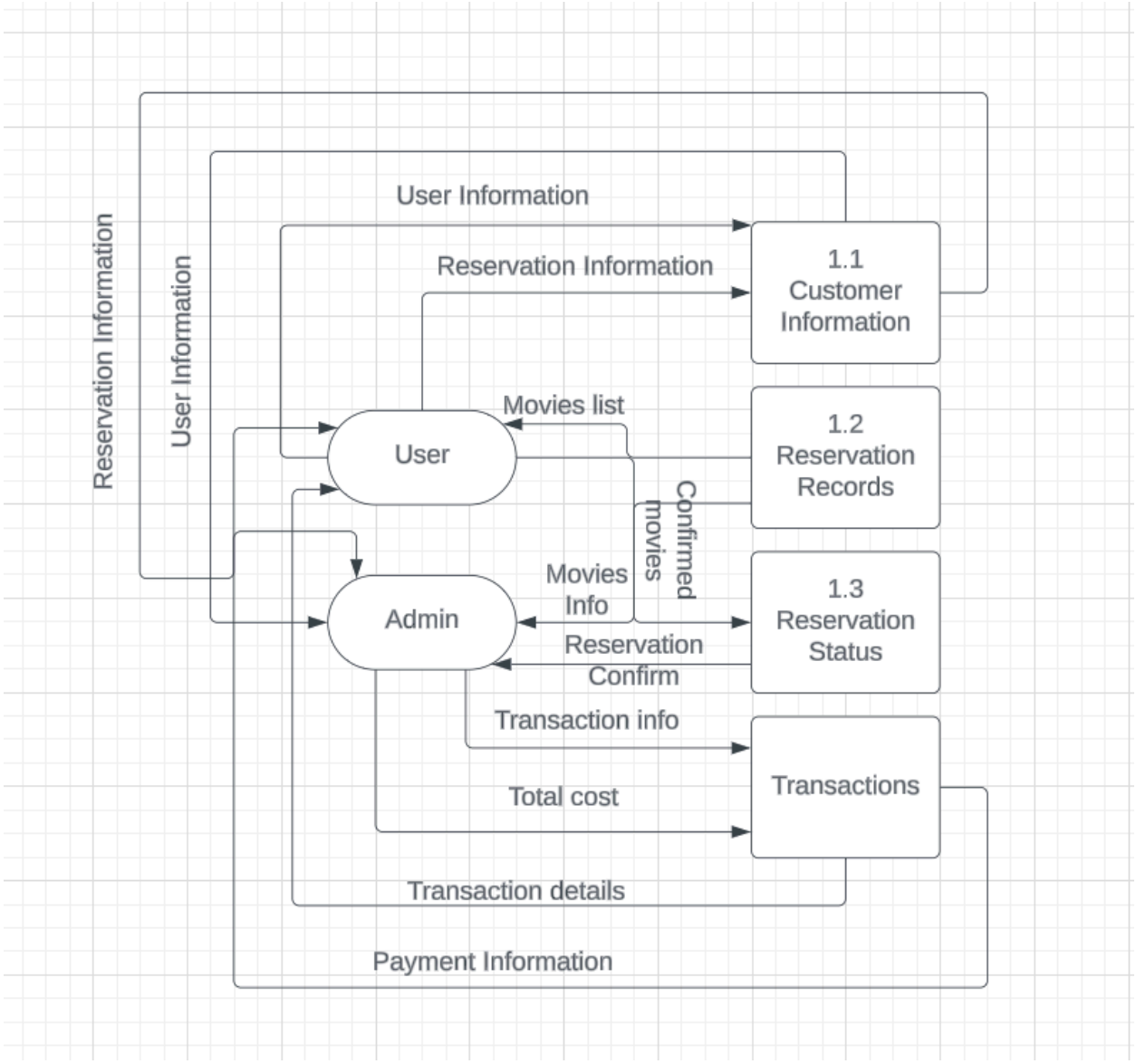
4.1 Sequence Diagrams




[else]
ACC DOES NOT EXIST
RETURN ERROR MESSAGE
VIEW MOVIES
GET MOVIES
RETURN MOVIES
SHOW MOVIES
VIEW SHOWTIMES
GET SHOWTIMES
RETURN SHOWTIMES
DISPLAY SHOWTIMES
VIEW SEATS
GET SEATS
RETURN SEATS
DISPLAY SEATS
SELECT SEATS
alternative
[if user signed in]
RETURN SAVED PAYMENT INFO
AUTO FILL PAYMENT INFO
REQUEST CVV
ENTER CVV
[else]
REQUEST PAYMENT INFO
SUBMIT PAYMENT
CONFIRM PAYMENT
Alternative
[if payment succesful]
PAYMENT OK
RETURN CONFIRMATION
RETURN TICKETS
[else]
PAYMENT ERROR
RETURN ERROR

This diagram explains the possible interactions a user would go through when navigating the ticketing website. It explains multiple different features, such as signing in to autofill payment method, and an error message when email address or password are incorrect. It also includes the features available for website admins, such as adding, removing or editing existing movies in the website database. It is displayed in chronological order, showing from top to bottom how a user would purchase tickets. Lastly, it illustrates how each portion of the system interacts with each other, including the website, its database and the banking software used to verify payment.

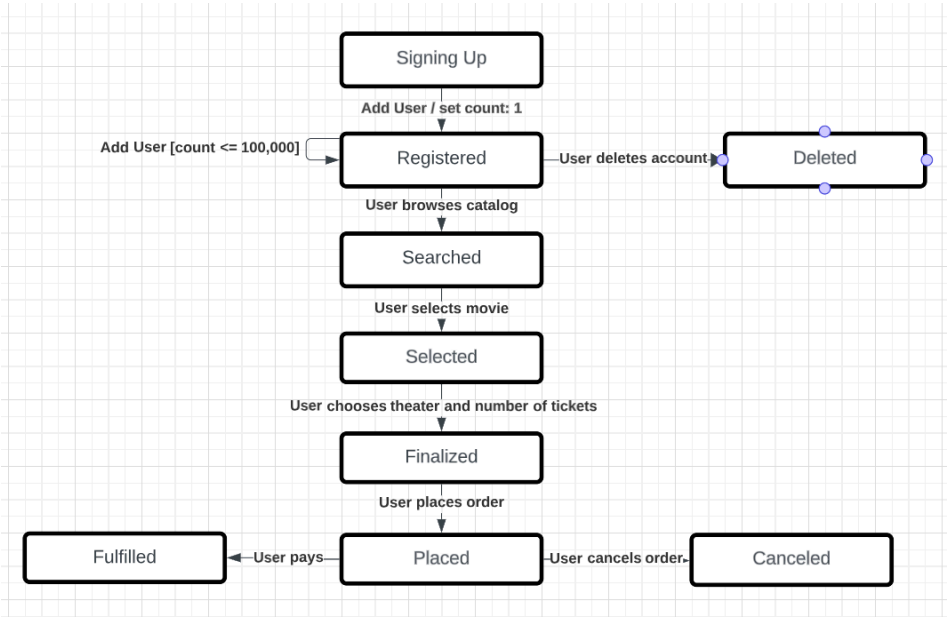
4.3 Data Flow Diagrams (DFD)



This diagram consisted of 4 sections: Customer Information, Reservation Records, Reservation Status, Transactions. Customer Information is a database and can be used to store customer information for future uses. Reservation Records hold existing items that customers can access

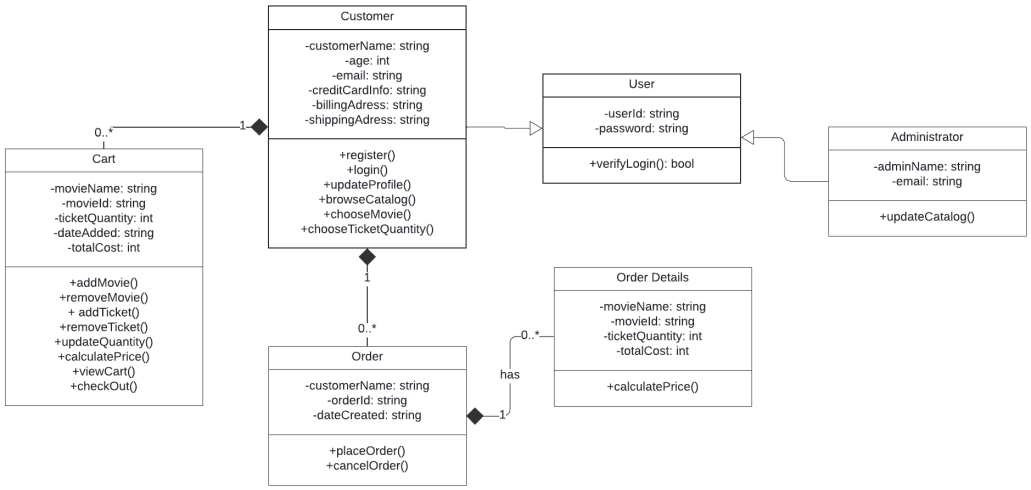
and choose from. Reservation Status confirming customer order before going to transaction. Transactions will give customer payment information and return to the admin if transactions fail. The flow of data starts from the user to reservation and returns to the admin for final transaction.

4.2 State-Transition Diagrams (STD)



This diagram portrays how Movicast is going to be used. Users first sign up and create an account. The maximum number of accounts that can be registered is 100,000. Users can delete their account if they choose to do so. Once users are registered, they are able to browse the catalog provided by Movicast. Users are then able to select their choice of movie, choose the theater that they want to watch at, and the number of tickets that they want to buy. Once these details are finalized, users are able to place their order. When orders are placed, users are able to cancel their order if they wish to do so. Otherwise, once they pay, the order is fulfilled.

4.3 UML Diagram



This diagram demonstrates the general structure of Movicast. All users will have an userId and a password, and will be prompted to sign in. With an account, users will be able to enter their information in order to utilize Movicast, such as their name, age, email, and payment information. Users will be able to update their profile, browse the movie catalog provided by Movicast, choose a movie to order, and the number of tickets. When ordering, users are able to add movies to a cart for later checkout. Items in the cart can be updated, such as adjusting what movies to purchase and the number of tickets. Users can then finalize their purchase, having an option of placing or canceling an order. Users will be provided with the order details.

5. Change Management Process

A. Appendices

A.1 Appendix 1

A.2 Appendix 2

6. Other Information

6.1 Software System Overview

Customers will interact with the movie ticketing website, browsing through a database of available movies, showtimes and seats. Website database also allows customers to sign in to access saved information such as billing information, address, preferred theater, etc. After selecting the desired movie and seats, customers will proceed to checkout. Once payment is successful, customers will see the ticket QR code and order number, which will also be sent through email.

6.2 List of Tasks

- Design main pages
- Implement movie database
- Implement available showtimes and seats to database
- Implement search feature
- Implement payment software
- Add option to create or sign in to account
- Allow users to save personal information
- Implement QR code generator, linked to order #
- Implement email service to send order receipt

6.3 Timeline

- Finish website prototype (Main pages without interactables) ~ 2 weeks. Members contributing: Santiago Verdugo, Nguyen Pham, Hung Quach
- Add interactables (Ways to navigate website) ~ 1 week. Members contributing: Santiago Verdugo
- Implement movie database and search (Add movies for users to browse through) ~ 2 weeks. Members contributing: Nguyen Pham, Hung Quach
- Implement showtimes and seats (Allow users to select desired seats) ~ 2 weeks. Members contributing: Santiago Verdugo, Nguyen Pham
- Implement payment options (Allow users to pay using debit/credit cards) ~ 1 week. Members contributing: Hung Quach
- Implement account features (Allow users to create accounts and save information) ~ 2 weeks. Members contributing: Santiago Verdugo, Hung Quach
- Implement post-order features (Generate ticket QR code, email receipt) ~ 2 weeks. Members contributing: Santiago Verdugo, Hung Quach, Nguyen Pham.

6.4 Verification Test Plan

For the test plan, refer to the following link:
https://docs.google.com/document/d/1P7E-8stB6kR7zKUyPDEU3iyILKqrh85wwiWYZWPfoKY/edit

6.5 Test Cases

For test cases, refer to the following link:
https://github.com/cs250group10/movicast/blob/main/CS250_Group__10_TestCases.xlsx

GitHub Links

Group Link: https://github.com/orgs/cs250group10/repositories
Nguyen Pham: https://github.com/spection01?tab=repositories
Hung Quach: https://github.com/HungQuachHQ?tab=repositories
Santiago Verdugo: https://github.com/svrdgo?tab=repositories