Software Design Specification

Software Title

- Movie Theater Ticketing System (MTTS)

Team Members

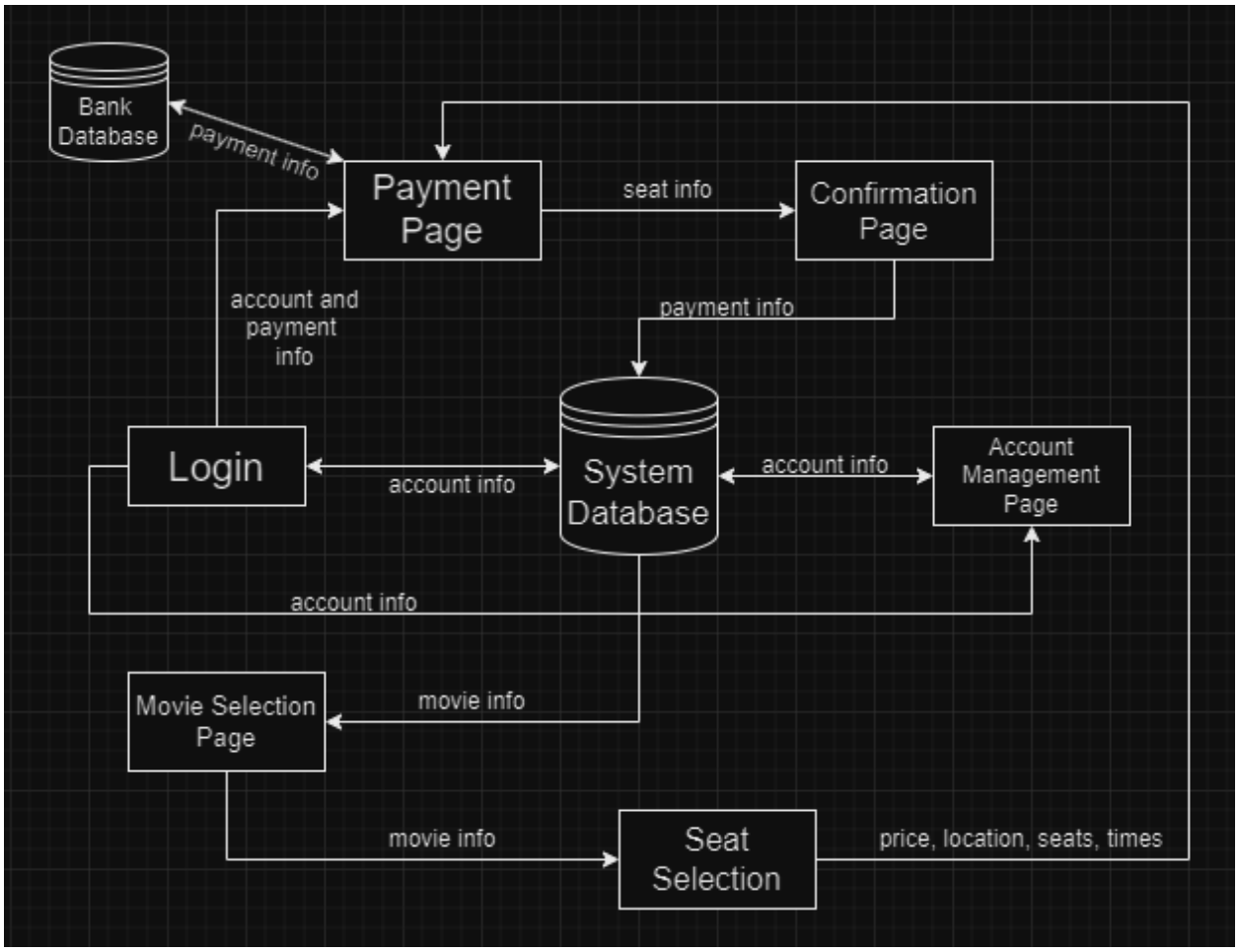
- Tim
- Ben
- Jahnavi

System Description

- The system allows customers to buy movie tickets online via accessing our website.
- The system allows smooth transactions between the customer and its designated bank service.
- The system keeps track of all necessary logs and receipts of transactions in case issues arise in the future.
- The system authenticates valid users by providing authentication tokens if account credentials are verified.
- The system does not hold card information due to security reasons.
- The system implements AWS Sage Maker to detect potential fraud transactions.

Software Architecture Overview

- Software Architecture Diagram



- Architectural Diagram of all Major Components

Data Management Strategy

Remarks

- Security is a critical aspect when dealing with data and databases (DB). With a significant increase of hacking activities worldwide, it is crucial to implement secure DB infrastructure for our project. Security is a challenge that evolves as time evolves. Therefore, although the following Data Management Strategy is fairly secure, we are constantly researching and trying to implement stronger security. Overall, it is best practice to add as many layers of security as possible. One practical way to implement this is by utilizing several DB instead of just 1 to split out data amongst the same topography.

Account Management DB

- This DB is responsible for withholding all valuable user information ranging from email address to their password. This DB is highly likely to be most targeted by malicious hackers due to the significant amount of valuable data. One way to mitigate this risk is by implementing another DB and splitting the amount of data in those two DB. By doing this, it poses another layer of security.
- This no-SQL database contains a user's email, username, hashed password, payment info, and their reward points.
- This database works closely with the user endpoint and Transaction Records Server (TRS).
- Once a valid user is authenticated via checking through this database, an authentication token is granted to the user via Account Management Server (AMS).
-

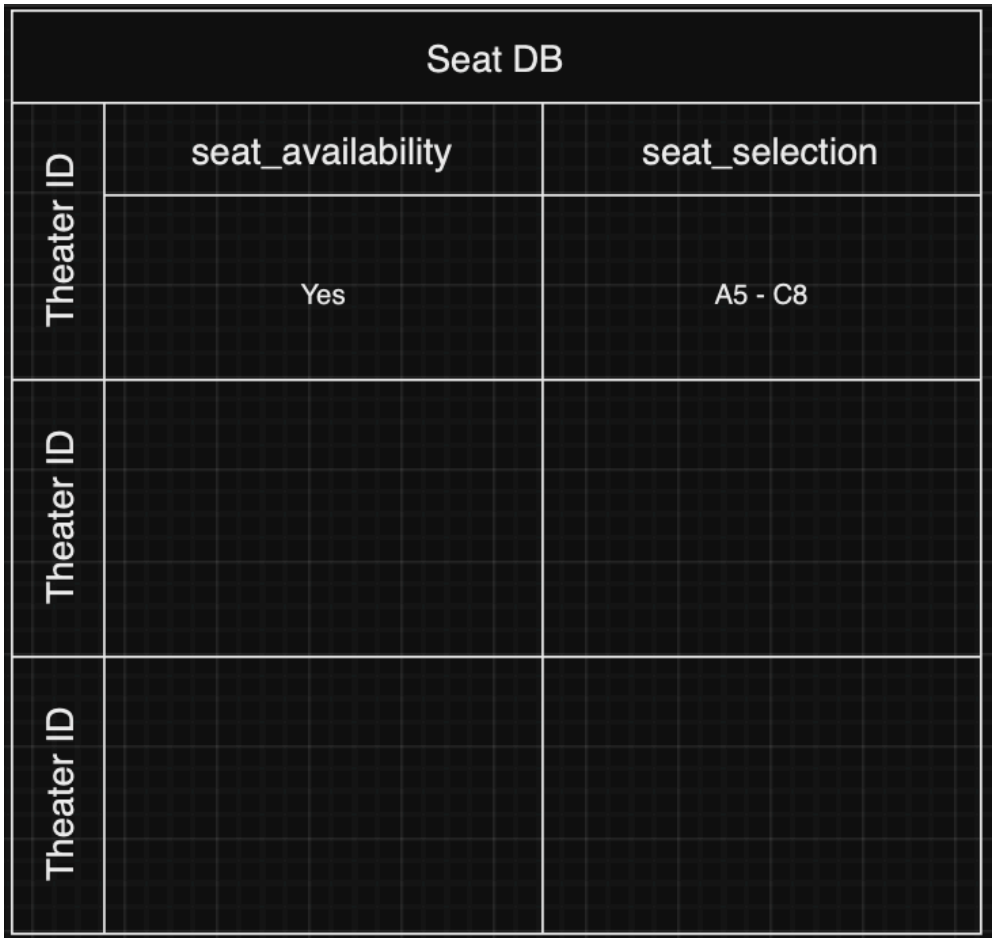
Account Management DB					
	email	username	password	payment_info	rewards
Actor 1	johnsmith@gmail.com	John	5f4dcc3b5aa765d61d8327deb882cf99	Credit/Debit	6900 points
Actor 2					
Actor 3					

Movie DB

- This DB is responsible for withholding all valuable movie information ranging from movie title to their movie duration. This DB is less likely a target for malicious hackers compared to Account Management DB, but still is a good practice to ensure an extra layer of security.
- This SQL database contains movie name, showtime, rating, genre, duration, theater ID, and MPAA rating.
- SQL database is used for this database since it provides faster categorization and filters according to the user's movie option.
- This database works closely with the user endpoint and returns movie options and availability.
- Once an available movie is available and selected, this database communicates with Seat DB to process ticketing.
- 

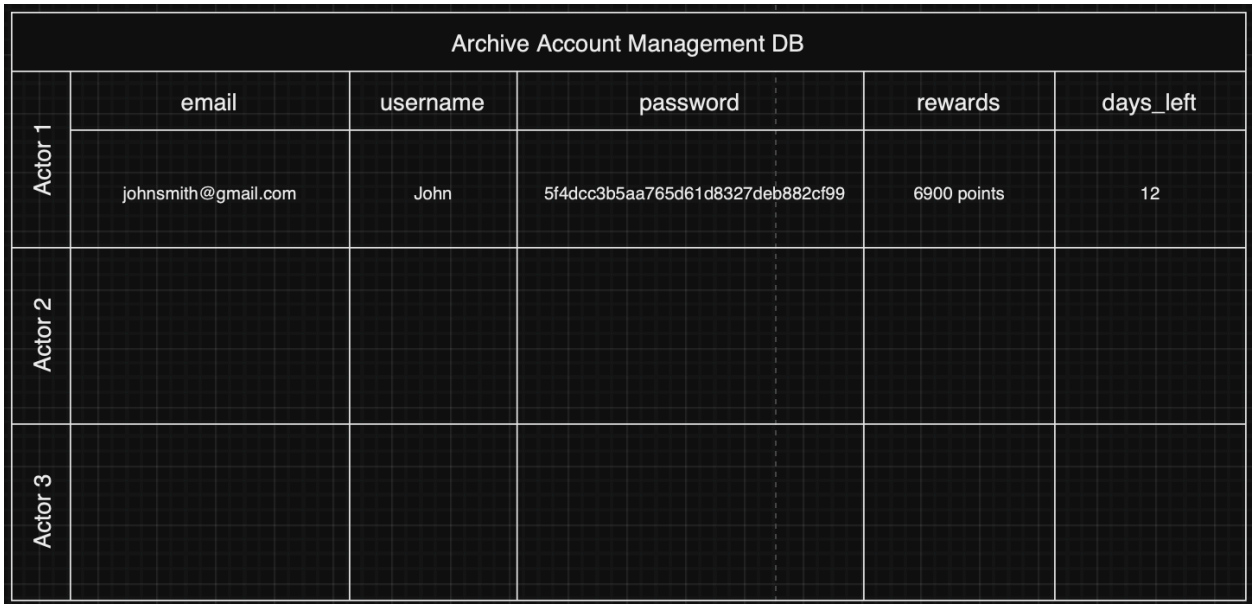
Seat DB

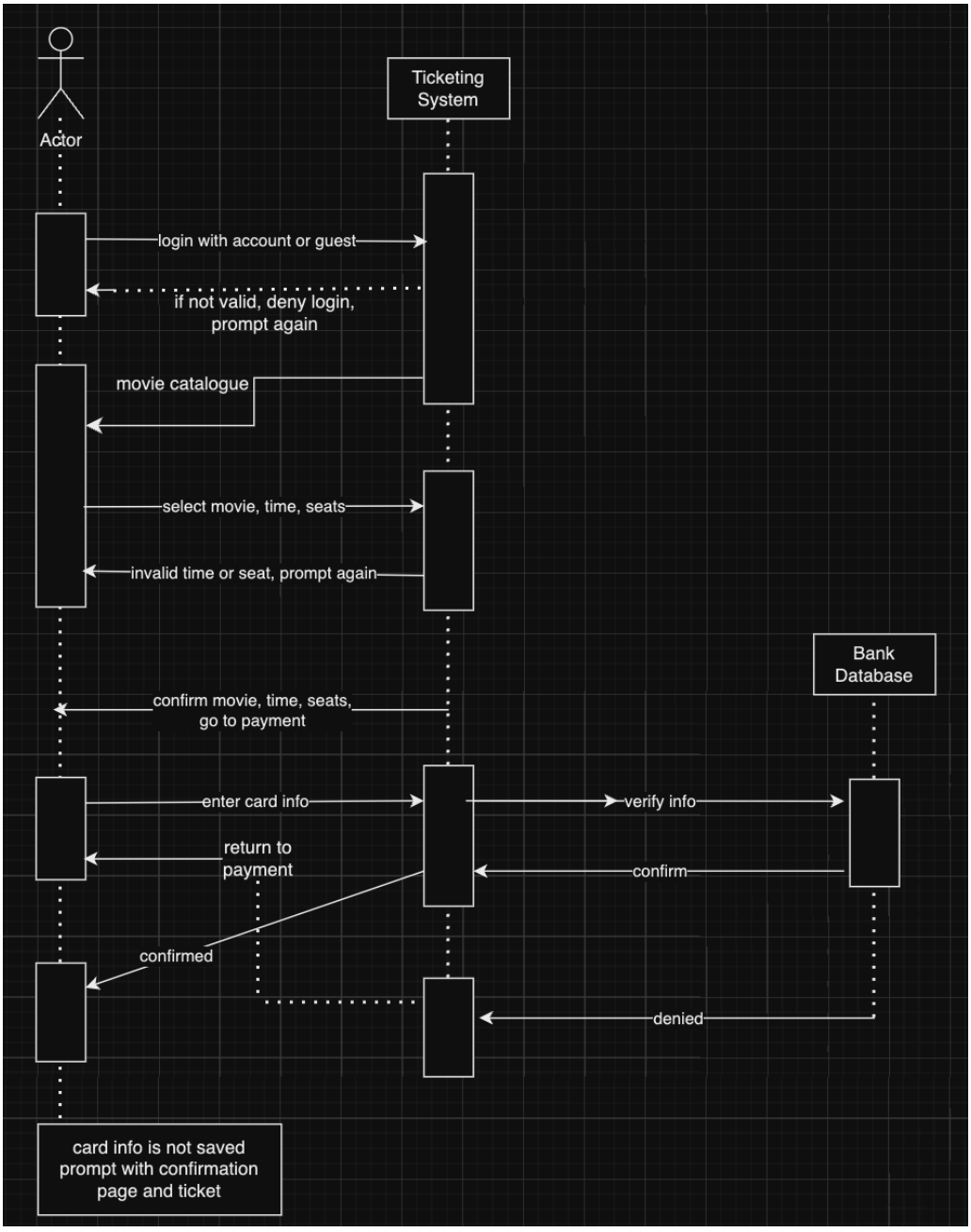
- This DB is responsible for withholding seat availability information ranging from A1 - F30. This DB is also less likely to be targeted by malicious hackers compared to Account Management DB since its data is less appealing. However, it may open up doors for potential hackers to gain privileged access. Therefore, it is always good practice to implement an extra layer of security.
- This no-SQL database contains unique theater ID and seat availability options.
- No-SQL database is used for this database since each seat chart has its unique seat code. Therefore, it is more compatible to use a no-SQL database to ensure a proper seat selection service.
- This database works closely with the user endpoint and the Movie DB.

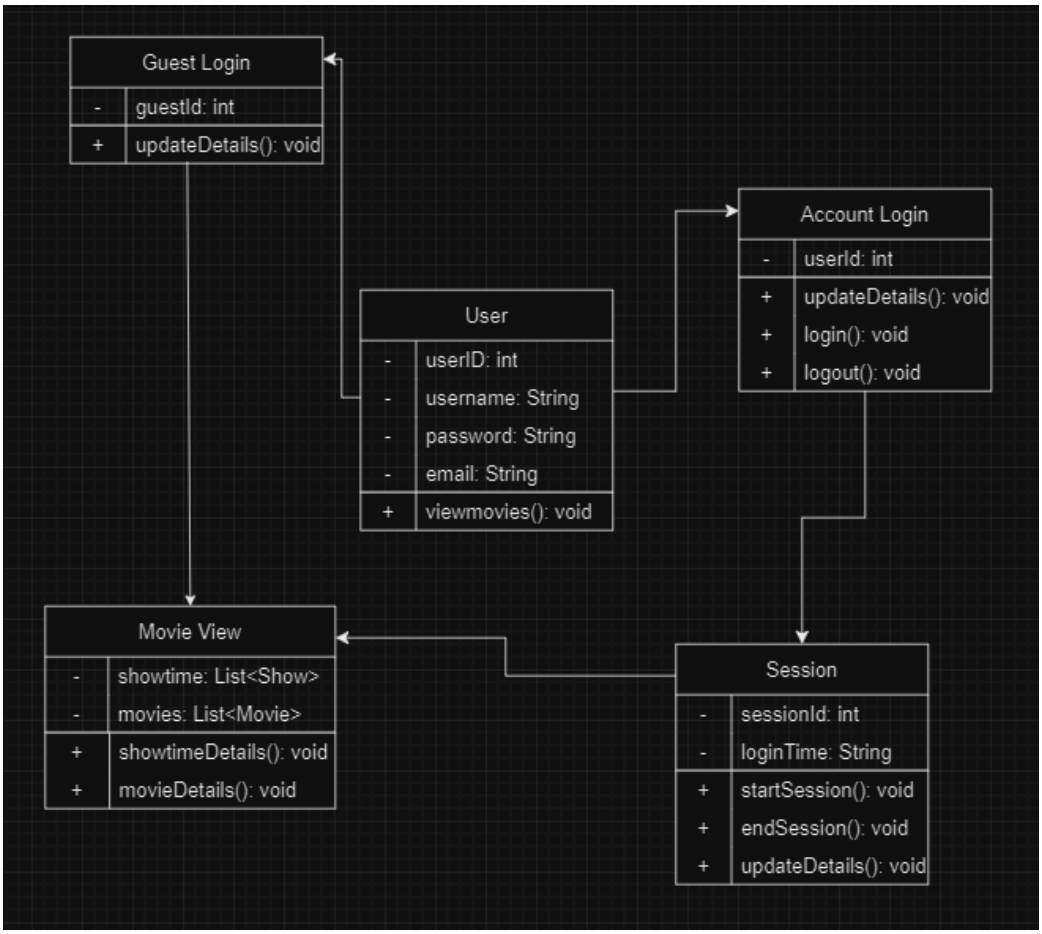
- 

Archive Account Management DB

- This DB is responsible for withholding data for deleted accounts. According to the Terms and Conditions our company and the user have signed, our company withholds the data of a deleted account for up to 30 days. After that, all data related to the deleted account is permanently deleted. The purpose of this is in case the user wants to reactivate their account again. In addition, due to legal reasons, it is recommended by the government to withhold user data for up to 30 days upon deletion request.
- This no-SQL database contains almost identical data with Account Management DB except “payment_info” and “days_left” category is added.
- No-SQL database is used for this database since each user account has specific days left so no structured query is necessary.
- This database works closely with the user endpoint and all other other DB.

- 
Archive Account Management DB
email
username
password
rewards
days_left
Actor 1
johnsmith@gmail.com
John
5f4dcc3b5aa765d61d8327deb882cf99
6900 points
12
Actor 2
Actor 3


Actor
Ticketing System
login with account or guest
if not valid, deny login, prompt again
movie catalogue
select movie, time, seats
invalid time or seat, prompt again
Bank Database
confirm movie, time, seats, go to payment
enter card info
verify info
return to payment
confirm
confirmed
denied
card info is not saved prompt with confirmation page and ticket

- UML Class Diagram
  - 

- Description of Classes
  - Stores information for the userID as an integer, stores the username and password as a string and calls viewmovies() method for the movie view class.
  - It branches to the Guest Login class when user info is not provided or goes to account login where it takes the userID and calls updateDetails() method to update info for the userID. Then calls login() or logout() depending on user input, which will log the user in or out.
  - After Account login it goes to session class where it contains the sessionID and loginTime which goes into the updateDetails(). startSession() info is recorded into the updateDetails() and endSession() will call logout().
  - The Movie View class is then called, which contains the showtime and movie lists and provides details for those lists through showtimeDetails() and movieDetails().
- Description of Attributes

    - Idle State: Welcome! Please log in to select your movie
        - Idle state leads to movie selection and movie time selection, if an error occurs it gets returned to the movie selection state.

    - Select Movie Time State:
        - Select Movie Time State leads to confirming selection and choosing seats, if an error occurs it goes back to choosing seats.

    - Choose seats state:
        - Choosing a seat state leads to confirming seat selection and selecting payment method, if an error occurs, it goes back to selecting payment method.

    - Select Payment state
        - Leads to confirmation, and sending payment info to the bank database, if an error occurs it resends to the bank database.

    - Payment going to bank state
        - Leads to confirmation, which leads to end, which results in going back to idle state.

- Description of Operations
    - Log-in / Guest
        - Users input the following information if they have an account: username and password.
        - Users bypass the login page if they click the “Login as Guest” button.
        - Assuming the user has an account, their account will be authenticated by the Account Management Server (AMS). If provided credentials are verified, an authentication token will be given to the user.
        - All of the traffic logs will be sent and stored in the Centralized Log Server (CLS).

    - Request Ticket
        - Users input the following information to buy a ticket/s: movie, time, location, and seat.
        - Those data will be sent to the Movie Management Server (MMS) to verify movie availability for the allotted data.
        - All of the traffic logs will be sent and stored in the Centralized Log Server (CLS).

    - Payment Process
        - Users input the following information: card number, full name, CVV, expiry date, and address.
        - These encrypted data will first be sent to the AWS Sage Maker Server to check for potential frauds. If detected, it will automatically deny payment and its log will be sent to the IT department.

        - All of the traffic logs will be sent and stored in the Centralized Log Server (CLS).
        - If it passes the potential fraud detection process, it will then be sent its encrypted data to the Designated Bank Server where the payment process will be held.
        - Once the bank approves the transaction, its receipt will be sent and stored in the Transaction Records Server (TRS) and it will be linked to the AMS of the account who purchased it.
        - All of the traffic logs will be sent and stored in the Centralized Log Server (CLS).

Development Plan and Timeline

- Tim
    - Designs Data Flow Diagram (DFD)
    - Data Management Strategy
- Ben
    - Designs Sequence Diagram
    - Designs UML Class Diagram
    - Software Architecture Diagram
- Jahnavi
    - Designs State-Transitions Diagram (STD)