Movie Theater Ticketing System

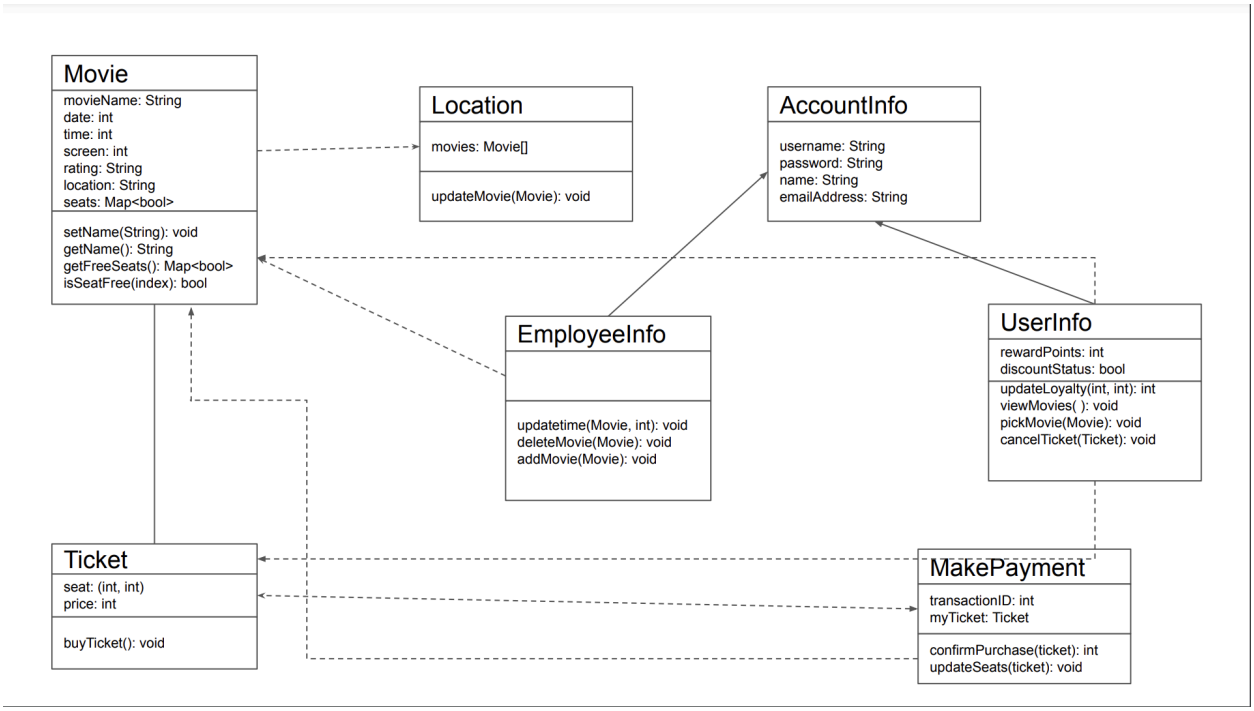
Team Members: Liam Hayes, Youngmin Park, Alex Colmenar

‘

Brief Overview:

The Movie Theater Ticketing System focuses on the stakeholders and applications that allow for online ticket sales, distribution, and marketing of movies. It will provide an accessible and intuitive interface for users to purchase tickets through a browser. The system will ensure it manages showtimes, ticket availability, and user transactions, ensuring consistency and security for the different movies. It will display movie reviews from critics updated constantly for live reviews and will not produce its own reviews.

UML Diagram:



Description:

Class: Movie

- Attributes:
  - `movieName: String` - The name of the movie.
  - `date: int` - The showing date of the movie.
  - `time: int` - The showing time of the movie.
  - `screen: int` - The screen number where the movie is shown.
  - `rating: String` - The rating of the movie (e.g., PG, PG-13, R).

        - `location: String` - The location of the movie showing.
        - `seats: Map<bool>` - A map representing the seats and their availability (true for available, false for taken).
- Operations:
    - `setName(String): void` - Sets the name of the movie.
    - `getName(): String` - Retrieves the name of the movie.
    - `getFreeSeats(): Map<bool>` - Returns a map of seats with their availability status.
    - `isSeatFree(index): bool` - Checks if a specific seat is free.

Class: Ticket

- Attributes:
    - `seat: (int, int)` - The row and column of the seat.
    - `price: int` - The price of the ticket.
- Operations:
    - `buyTicket(): void` - Processes the ticket purchase.

Class: Location

- Attributes:
    - `movies: Movie[]` - An array of `Movie` objects available at the location.
- Operations:
    - `updateMovie(Movie): void` - Updates the movie details.

Class: AccountInfo

- Attributes:
    - `username: String`
    - `password: String`
    - `name: String`
    - `emailAddress: String`

Class: EmployeeInfo

- Operations:
    - `updateTime(Movie, int): void` - Updates the time for a movie showing.
    - `deleteMovie(Movie): void` - Removes a movie from the schedule.
    - `addMovie(Movie): void` - Adds a new movie to the schedule.

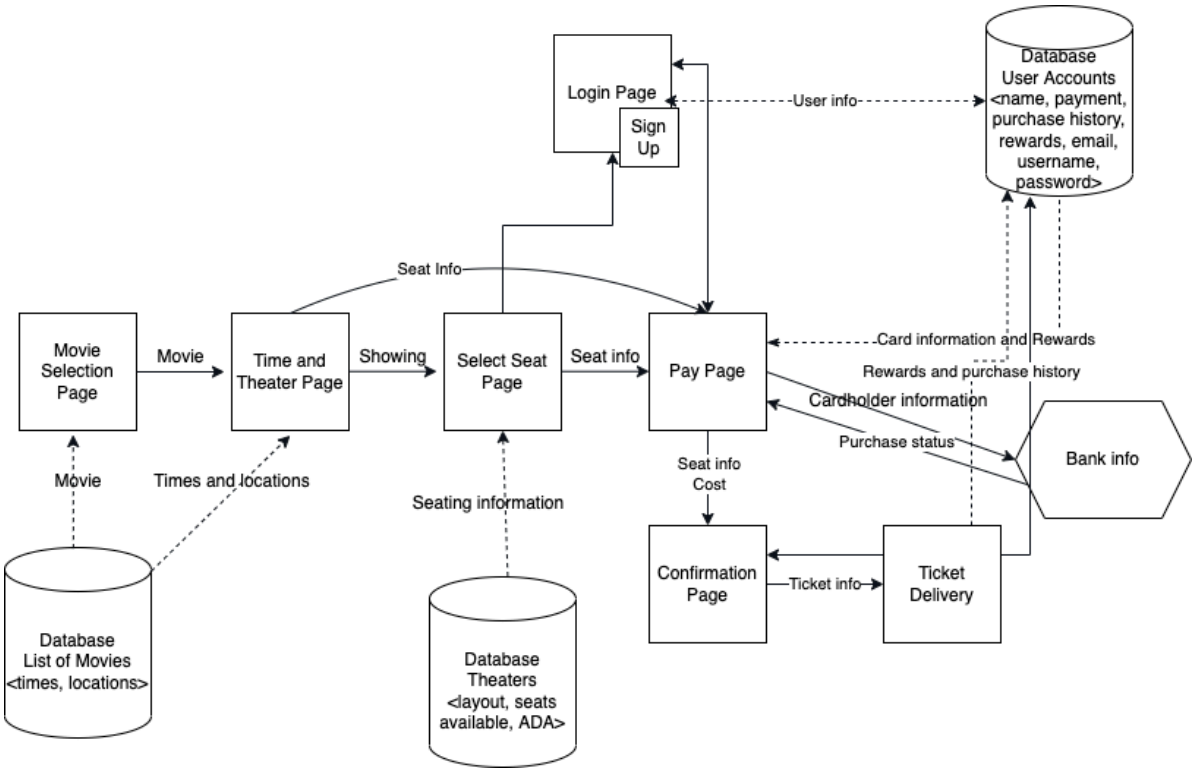
Class: UserInfo

- Attributes:
  - `rewardPoints: int` - The loyalty points of the user.
  - `discountStatus: bool` - Indicates if the user is eligible for a discount.
- Operations:
  - `updateLoyalty(int, int): int` - Updates loyalty points for the user.
  - `viewMovies(): void` - Displays available movies.
  - `pickMovie(Movie): void` - Selects a movie for ticket purchase.
  - `cancelTicket(Ticket): void` - Cancels a ticket purchase.

Class: MakePayment

- Attributes:
  - `transactionID: int` - The ID of the transaction.
  - `myTicket: Ticket` - The ticket associated with the payment.
- Operations:
  - `confirmPurchase(ticket): int` - Confirms the purchase of a ticket and returns a transaction ID.
  - `updateSeats(ticket): void` - Updates the seat availability after a ticket purchase.

SWA Diagram:



Description:

Movie Selection Page: Users start here to browse movies. This component interacts with a database to retrieve a list of movies, including their showtimes and locations.

Time and Theater Page: After selecting a movie, users are directed here to choose a specific time and theater location, pulling data from a database that lists movie times and locations.

Select Seat Page: Users pick their seats based on availability, which is checked against a seating database for the selected theater.

Pay Page: Payment information is collected and processed in this component, with secure transactions being a priority.

Confirmation Page: This component confirms the successful transaction, providing users with their ticket information and seat details.

Ticket Delivery: Manages the distribution of tickets, ensuring users receive them in their chosen format.

The databases—User Accounts, List of Movies, and Theaters—store all relevant data and interact with the system's pages to provide up-to-date information and maintain the integrity of user data and transactional records.

The Login Page and Sign Up are connected to the User Accounts Database, handling user authentication and account creation. Additionally, the system interfaces with external Bank Info for payment processing, ensuring secure financial transactions.

Connectors between these components represent data flow and user navigation paths, indicating how users move through the system and how data is exchanged to support functionalities like movie selection, seat reservation, and payment processing.

Development Plan and Timeline

- Partitioning of Tasks:
    - UI Design and Development
    - Backend System Development

        - Payment Integration
        - Testing and Quality Assurance
- Team Member Responsibilities:
    - Alex Colmenar: UI Design and Frontend Development
    - Liam Hayes: Backend System Development, Testing and Quality Assurance
    - Youngmin Park: Database Management and Integration
    - Team Member 4: Testing and Quality Assurance
- Timeline:
    - Week 1-2: Requirement Analysis and Planning
    - Week 3-4: UI Design
    - Week 5-8: Backend Development
    - Week 9-10: Integration and Testing
    - Week 11: Final Review and Deployment