**Use Case: Reserve Seats in the Cinema**

* **Primary Actor:** Customer
* **Goal:** Allow the customer to select and reserve seats in the cinema based on the desired seat type.
* **Preconditions:** The system is operational and displaying a list of available seats for reservation.
* **Main Flow of Events:**
  1. The system presents the customer with an option to choose between regular or VIP seats.
  2. The customer selects the desired seat type (regular or VIP).
  3. The system displays a map of available seats in the cinema, marking the reserved seats.
  4. The customer selects the desired seats by inputting the seat labels or returns to the menu.
  5. The system checks the validity of the seats and reserves the selected ones if available.
  6. The system confirms the successful reservation of the selected seats.
* **Alternate or Exceptional Flows:**
  1. If the customer inputs an invalid or already reserved seat label, the system displays an error message and requests another valid seat.
  2. If the customer chooses to return to the menu during seat selection, the system terminates the reservation process and returns to the main menu.
* **Post-conditions:** The valid seats selected by the customer are reserved and marked as occupied in the system. The customer receives a confirmation of successful reservation.

**Use Case: Process Payment for Movie Tickets**

* **Primary Actor:** Customer
* **Goal:** Facilitate the customer's payment process for booking movie tickets using different payment methods.
* **Preconditions:** The customer has selected movie tickets and confirmed seat reservations.
* **Main Flow of Events:**
  1. The customer selects the preferred payment method (Card, Cash, or Loyalty Points).
  2. The system sets the payment state based on the chosen payment method.
  3. The system displays a confirmation message indicating the selected payment type.
  4. The system processes the payment based on the selected method using the appropriate payment state.
  5. The system calculates the total price of the selected movie tickets based on the reserved seats.
* **Alternate or Exceptional Flows:**
  1. If the customer selects an invalid payment method or enters an incorrect option, the system displays an error message indicating the invalid choice.
  2. If the number of reserved seats is zero, the system calculates the movie price based on the standard ticket price without any additional seats reserved.
* **Post-conditions:** The customer's payment is processed using the chosen payment method, and the total price of the movie tickets is calculated based on the number of reserved seats. The system confirms the payment method and the total amount to be paid by the customer.

**Use Case: Managing Food and Drink Selection**

**Actor:** Customer **Purpose:** To interactively manage the selection and modification of food and drink items.

**Main Flow**

1. Start
   * The customer initiates the food and drink selection process.
2. View Chosen Items
   * If no items are added yet:
     + Display a message indicating that no items have been selected.
   * If items are added:
     + Show the list of selected food or drink items.
3. Display Menu
   * Present menu options for different cuisines (Mexican, Italian, Indian).
   * For each cuisine:
     + Display the available food and drink items.
4. Delete From Order
   * If no items are added:
     + Show a message indicating that there are no items to remove.
   * If items are added:
     + Display the list of added food or drink items with their indices.
     + Prompt the user to input the index of the item to be deleted.
     + Remove the selected item from the list and display the updated list.
5. Add Food or Drink Items
   * Display menu choices for various cuisines.
   * Allow the customer to select a cuisine type (Mexican, Italian, Indian, or Exit).
   * Based on the chosen cuisine:
     + Present the corresponding menu items.
     + Enable the customer to select a specific food or drink item to add to the order.
     + Add the chosen item to the list of selected items.
6. End
   * Terminate the food and drink selection process.

**Alternative Flow**

* Invalid Deletion Attempt
  + If the customer inputs an invalid index during deletion:
    - Request the user to input a valid index for item removal.
* Invalid Menu Selection
  + If the customer selects an invalid menu type:
    - Prompt the customer to input a valid index within the specified range.

**Use Case: Select and Display Movie Information**

**Actor:** User  
**Description:** The user selects a movie from a specific genre and views its details.

1. **Start:** User interacts with the system to choose a movie.
2. **System:** Prompts the user to select a movie genre (Action, Comedy, Drama, Horror).
3. **User:** Enters the preferred movie genre.
4. **System:** Retrieves and displays available movies within the chosen genre.
5. **User:** Selects a movie by entering its corresponding number from the displayed list.
6. **System:** Validates the input and retrieves details of the chosen movie.
7. **System:** Displays movie details, including its name and type.
8. **End:** User receives information about the selected movie.

**Alternate Scenario - Invalid Input**

4a. User: Enters an invalid genre or number. 4b. System: Notifies the user about the invalid input and re-prompts for a valid choice. 4c. Back to Step 3: User re-enters a valid movie genre or number.

**Alternate Scenario - No Movie Selected**

5a. User: Chooses to exit or doesn't select any movie. 5b. System: Acknowledges the user's choice or displays an exit message. 5c. End: Use case ends without displaying specific movie details.

**Use Case: Buy Food, Drinks, and Souvenirs**

**Actor:** Customer  
**Description:** The customer interacts with the system to purchase food, drinks, and souvenirs.

1. **Start:** Customer initiates the process to buy food, drinks, or souvenirs.
2. **System:** Presents a menu with options:
   * See the Menu \
   * Add Items to the cart (Food, Drinks, and Souvenirs)
   * See all the chosen items
   * Remove Item from the cart
   * Exit
3. **User:** Selects an action from the presented menu.
4. **System:** Executes the chosen action.
   * Action 1 - See the Menu:
     + Prompts the user to choose between seeing the food/drinks menu or the souvenir menu.
     + Based on the selection, displaying 2 different menus.
   * Action 2 - Add Items to the Cart:
     + Prompts the user to choose between buying foods/drinks or souvenirs.
     + Based on the selection, allows the user to add items to their respective carts.
   * Action 3 - See Chosen Items:
     + Displays the list of currently chosen food, drinks, and souvenir items.
   * Action 4 - Remove Item from the Cart:
     + Asks the user whether they want to remove food/drink or souvenir.
     + Removes the selected item from the respective cart.
   * Action 5 - Exit:
     + Ends the interaction with the system and exits the process.
5. **End:** Interaction concludes based on the user's choice to exit or perform further actions.

**Alternate Scenario - Invalid Input**

3a. User: Enters an invalid menu option. 3b. System: Prompts the user to enter a valid choice. 3c. Back to Step 3: User re-enters a valid menu option.

**Alternate Scenario - Removing Item from an Empty Cart**

4a. User: Tries to remove an item when the cart is empty. 4b. System: Informs the user that the cart is empty, prompting no action or a message. 4c. Back to Step 4: User continues with another action or exits.

**Use Case: Manage Cinema Database**

**Actor:** Cinema Management System **Description**: Interaction with the cinema database to save and retrieve information.

1. **Start**: Initiate interactions with the cinema database.
2. **System:** Presents database-related options:
   * Establish a database connection
   * Save information to the database
   * Read data from the database
3. **User:** Selects an action from the available options.
4. **System:** Executes the chosen action.
   * Action 1 - Establish a Database Connection:
     + Establishes a connection to the cinema database using JDBC.
   * Action 2 - Save Information to the Database:
     + Saves information related to a cinema transaction into the database:
       - Date of transaction
       - Movie details
       - Category of the movie
       - Seat type and number
       - Purchased souvenirs
       - Food items
       - Transaction price and payment type
   * Action 3 - Read Data from the Database:
     + Retrieves and displays stored data from the database, listing transactions with details like:
       - Date
       - Movie
       - Movie category
       - Seat type and number
       - Souvenirs purchased
       - Food items ordered
       - Transaction price
       - Payment type
5. **End:** Interaction concludes based on the user's selection to perform further actions or exit.

**Alternate Scenario - Database Connection Failure**

2a. System: Fails to establish a connection to the database due to network issues or invalid credentials. 2b. System: Notifies the user about the connection failure and potential causes. 2c. Back to Step 2: User may retry establishing the connection or opt for other actions.

**Alternate Scenario - Data Retrieval Error**

4a. System: Encounters errors while retrieving data from the database, such as SQL syntax issues or data corruption. 4b. System: Alerts the user about the issue in data retrieval and its potential causes. 4c. Back to Step 4: User may re-initiate data retrieval or select other actions.