Movie Ticket Booking System - Project Report

1. Title Page

Course: MCA-II Semester (Jan-2025) - OOP-II [MCA 4221]

Assignment: IA-3 (Mini Project)

Project Title: Design and Implementation of a Movie Ticket Booking System

Submitted By:

- **Name:** Chandradip Roy
- **Registration No:** 240970063
- **Department:** Department of Data Science & Computer Applications

2. Overview (Problem Statement)

This project aims to design and develop a desktop application for a Movie Ticket Booking System using Java Swing for the graphical user interface (GUI) and MongoDB for database connectivity. The system allows users to log in, browse available movies and their showtimes, select seats visually, proceed through a simulated payment process, and receive a booking confirmation.

The core requirements include:

1. **User Authentication:** A secure login page for users to access the system.
2. **Movie Browsing:** Displaying a list of currently running movies with details like description, theatre, and available showtimes.
3. **Seat Selection:** An interactive interface showing the layout of seats for a selected showtime, indicating available, booked, and selected seats. Users can pick their desired seats.
4. **Booking & Payment:** A mechanism to finalize the seat selection, calculate the total cost, simulate payment processing, and save the booking details to the database.
5. **Confirmation:** Displaying a confirmation message to the user with all relevant booking details upon successful payment.

The application utilizes Java Swing components for building the user interface, Maven for project management and dependency handling, and the official MongoDB Java Driver for database interactions.

3. Project Setup and Execution Guide

This section provides instructions on how to set up the development environment and run the Movie Ticket Booking System application. Using Visual Studio Code with Maven is recommended for a smoother experience.

3.1 Prerequisites

Ensure the following software is installed on your system:

1. **Java Development Kit (JDK):** Version 11 or higher is required. You can download OpenJDK distributions like Adoptium Temurin or Oracle JDK. Verify installation by opening a terminal or command prompt and typing java -version.
2. **Apache Maven:** A build automation tool used for managing dependencies and building the project. Download from the official Apache Maven website and follow their installation instructions. Verify installation by typing mvn -v in a terminal.
3. **MongoDB Community Server:** The NoSQL database used by the application. Download and install it from the official MongoDB website. Ensure the MongoDB service (mongod) is running, typically accessible at mongodb://localhost:27017.
4. **Visual Studio Code (VS Code):** A lightweight code editor. Download from the official VS Code website.
5. **VS Code Java Extensions:** Install the "Extension Pack for Java" from Microsoft within VS Code's Extensions view (Ctrl+Shift+X). This pack includes support for Java development, Maven, debugging, etc.

3.2 Getting the Code

Obtain the project source code files, including the pom.xml file and the src directory containing all .java files organized into packages (com.ticketbook.movieapp, com.ticketbook.movieapp.db, etc.).

3.3 Maven Setup

- **pom.xml**: This file, located in the project's root directory (movieticketbooking/), is the core of the Maven project. It defines project information, dependencies (like the MongoDB driver), and build instructions.
- **Dependencies:** Maven reads the <dependencies> section in pom.xml and automatically downloads the required libraries (JAR files) when you build the project. You do not need to download JARs manually.
- **Verification:** Ensure Maven is correctly installed and accessible from your terminal by running mvn -v.

3.4 IDE Setup (Visual Studio Code)

1. **Open Project Folder:** Launch VS Code and select File > Open Folder.... Navigate to and select the **root** movieticketbooking folder (the one containing pom.xml and src).
2. **Extensions:** Make sure the "Extension Pack for Java" is installed and enabled.
3. **Project Recognition:** VS Code's Java extensions should automatically detect the pom.xml file and recognize it as a Maven project. You might see prompts to import the project; allow it. The "Java Projects" view in the Explorer sidebar should show your project structure and Maven dependencies.

3.5 MongoDB Setup

1. **Start MongoDB:** Ensure your MongoDB server instance is running. If installed as a service, it might start automatically. Otherwise, you may need to start the mongod process manually. The application expects it to be running on the default port 27017 on localhost.
2. **Populate Sample Data:** The application requires some initial data (users, movies) to function correctly.
   - Open a tool like MongoDB Compass or the mongosh command-line shell.
   - Connect to your local MongoDB instance.
   - Use the sample data insertion queries provided previously (artifact mongo_sample_data_all) to create the movie_booking_db database and populate the users and movies collections. You can optionally add sample bookings data as well, remembering to substitute placeholder ObjectIds.

3.6 Building the Project

Before running the application, you need to build it using Maven. This compiles your Java code and downloads dependencies.

1. **Open Terminal:** Use the integrated terminal in VS Code (Terminal > New Terminal).
2. **Navigate:** Ensure the terminal is in the project root directory (movieticketbooking).
3. **Run Build:** Execute the command:
   mvn clean install
4. **Check Output:** Look for *[INFO] BUILD SUCCESS* at the end. This confirms dependencies were downloaded and code was compiled into the target/classes directory.

3.7 Running the Application

There are multiple ways to run the application after a successful build:

1. **From VS Code (Recommended):**
    - Navigate to src/main/java/com/ticketbook/movieapp/App.java in the VS Code Explorer.
    - Right-click within the App.java editor window and select "Run Java".
    - Alternatively, click the small "Run" overlay icon that appears above the main method.
    - VS Code uses the Maven project context to run the application with the correct classpath.
2. **Using Maven Exec Plugin (Terminal):**
    - Ensure your terminal is in the project root directory.
    - Run the command:
      ```
      mvn exec:java
      ```
    - This uses the exec-maven-plugin defined in pom.xml to run the main class.
3. **Using the Executable JAR (Terminal):**
    - The mvn clean install command created an executable JAR in the target directory.
    - Navigate to the target directory: cd target
    - Run the JAR file (the exact filename might vary slightly based on version):
      ```
      java -jar movieticketbooking-1.0-SNAPSHOT-jar-with-dependencies.jar
      ```
    - This runs the application packaged with all its dependencies.

4. Flow Diagram

The system follows a sequential workflow initiated by the user:

1. **Start Application:** The user launches the application (App.java).
2. **Display Login:** The LoginFrame is displayed, prompting the user for username and password.
3. **Authenticate User:**
    - User enters credentials and clicks "Login".
    - The system verifies credentials against the users collection in the MongoDB database (handleLogin in LoginFrame).
    - **If Invalid:** An error message is shown. User remains on the Login screen.
    - **If Valid:** The LoginFrame is closed.
4. **Display Movie Selection:** The MovieSelectionFrame is displayed, populated with

movie data loaded from the movies collection in MongoDB (loadMoviesFromDatabase).

5. **Select Movie & Showtime:**
   - User selects a movie from the list (JList).
   - Movie details and available showtimes (from the selected movie document) are displayed.
   - User selects a showtime from the dropdown (JComboBox).
   - User clicks "Select Seats".

6. **Display Seat Selection:** The MovieSelectionFrame is closed, and the SeatBookingFrame is displayed for the chosen movie and showtime. Booked seats for this specific show are loaded from the bookings collection and disabled (loadBookedSeats). A "Back to Movies" button is available.
7. **Select Seats:**
   - User clicks on available seat buttons (JToggleButton). Selected seats are highlighted.
   - User clicks "Proceed to Payment".

8. **Display Payment/Summary:** The SeatBookingFrame is closed, and the PaymentFrame is displayed, showing a summary of the selected movie, showtime, seats, and total cost. "Back" and "Cancel Booking" buttons are available.
9. **Confirm Payment & Booking:**
   - User clicks "Confirm Booking & Payment".
   - The system simulates payment success.
   - A new booking document containing user ID, movie ID, showtime, seats, amount, and timestamp is inserted into the bookings collection in MongoDB (processBooking in PaymentFrame).
10. **Display Confirmation:** The PaymentFrame is closed, and a ConfirmationDialog (using JOptionPane) is shown with the booking details (including a unique booking ID).
11. **Return/Exit:** After the user closes the confirmation dialog, the application flow typically returns to the MovieSelectionFrame. The application also gracefully closes the MongoDB connection upon JVM shutdown via a shutdown hook (App.java).

5. Swing Components Used

The application utilizes various Java Swing components to build the graphical user interface. Key components include:

- **JFrame**: Represents the main window for each primary screen (LoginFrame, MovieSelectionFrame, SeatBookingFrame, PaymentFrame).
- **JPanel**: Used as a container to group and organize other components using various Layout Managers.
- **JLabel**: Displays static text labels for instructions, field names, informational messages, and booking summaries.
- **JTextField**: Allows single-line text input (e.g., username).
- **JPasswordField**: Allows masked single-line text input (e.g., password).
- **JButton**: Represents clickable buttons for actions ("Login", "Select Seats", "Proceed", "Confirm", "Cancel", "Back").
- **JList**: Displays a scrollable list of movies.
- **JComboBox**: A dropdown list for selecting showtimes.
- **JTextArea**: Displays multi-line text (e.g., movie descriptions).
- **JScrollPane**: Provides scroll bars for components like JList, JTextArea, and the seat panel.
- **JToggleButton**: A two-state button representing individual seats.
- **JOptionPane**: Displays standard dialog boxes for messages and confirmations.
- **Layout Managers** (GridBagLayout, BorderLayout, FlowLayout, BoxLayout): Control the arrangement of components within panels.

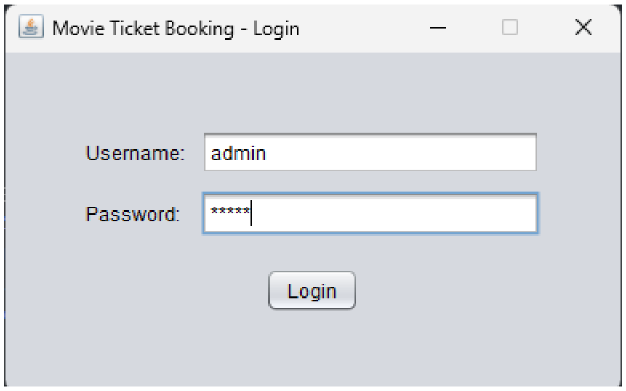
6. Event Handling

Event handling enables interactivity. The listener model is used: components fire events, and registered listeners execute handler code.

- **ActionListener**: Attached to JButton and JToggleButton. The actionPerformed method handles button clicks (Login, Select Seats, Proceed, Confirm, Cancel, Back, seat toggling).
- **ListSelectionListener**: Attached to the movie JList. The valueChanged method updates the details area and showtime combo box when a movie selection changes.
- **Event Dispatch Thread (EDT):** SwingUtilities.invokeLater ensures UI creation and updates happen safely on the EDT.

7. Screenshots

Screenshot 1: Login Screen

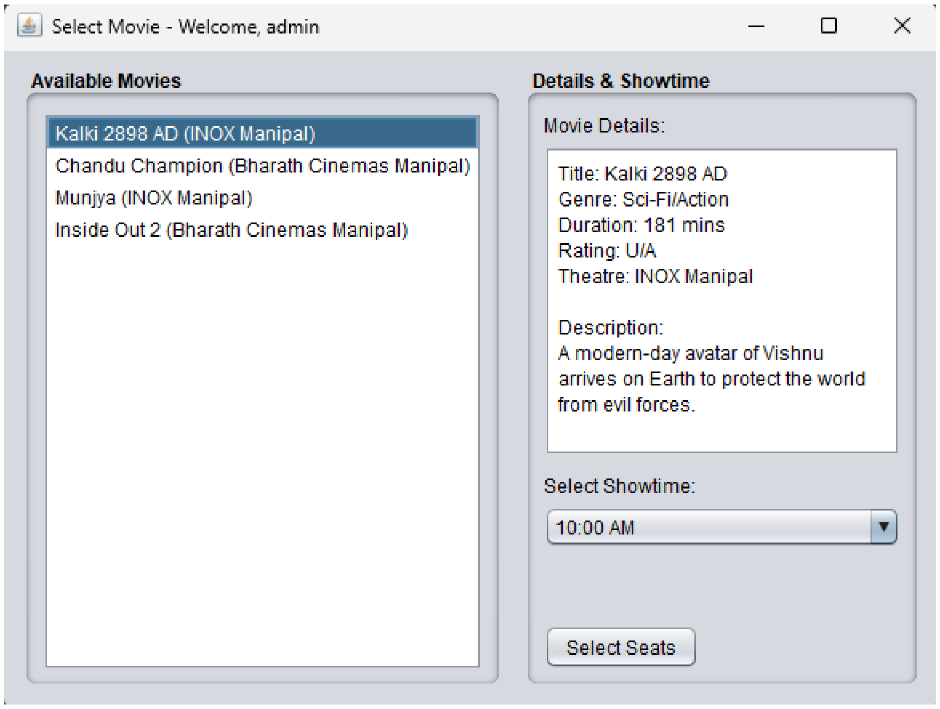

Movie Ticket Booking - Login

Username: admin

Password: *****

Caption: The initial login window.

Screenshot 2: Movie Selection Screen


Select Movie - Welcome, admin

Available Movies

- Kalki 2898 AD (INOX Manipal)
- Chandu Champion (Bharath Cinemas Manipal)
- Munjya (INOX Manipal)
- Inside Out 2 (Bharath Cinemas Manipal)

Details & Showtime

Movie Details:

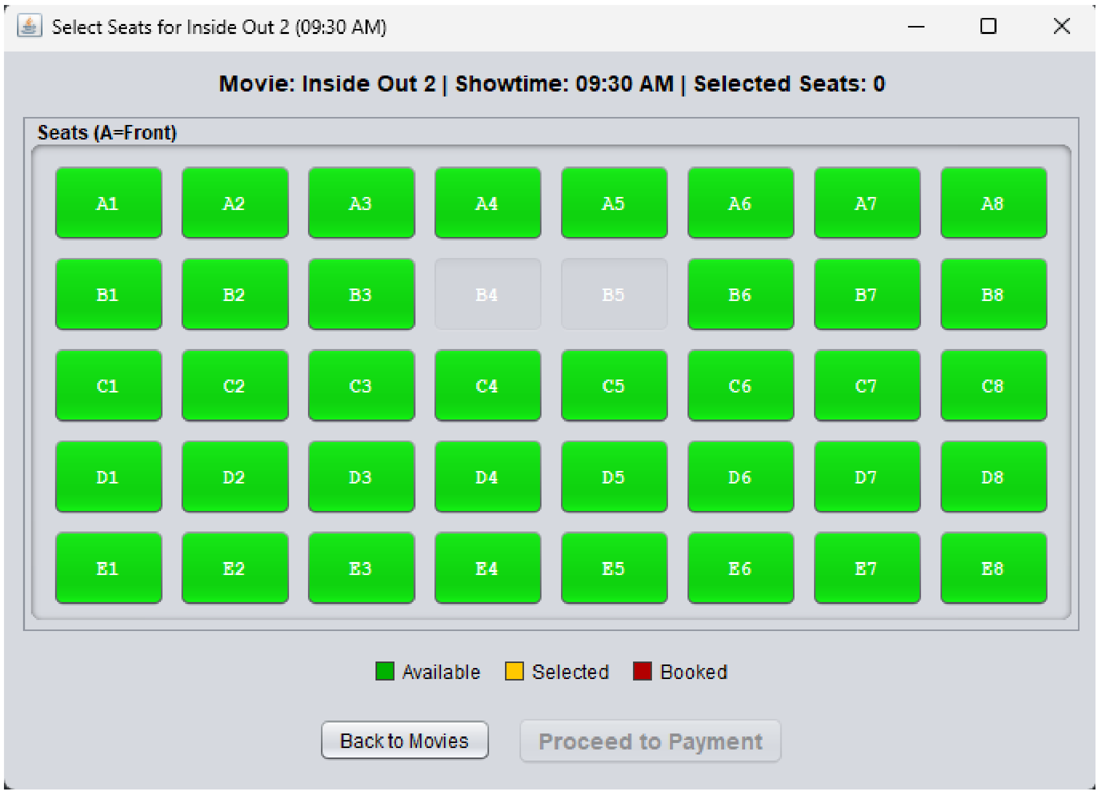
Title: Kalki 2898 AD
Genre: Sci-Fi/Action
Duration: 181 mins
Rating: U/A
Theatre: INOX Manipal

Description:
A modern-day avatar of Vishnu arrives on Earth to protect the world from evil forces.

Select Showtime: 10:00 AM

Caption: Movie listing and details.

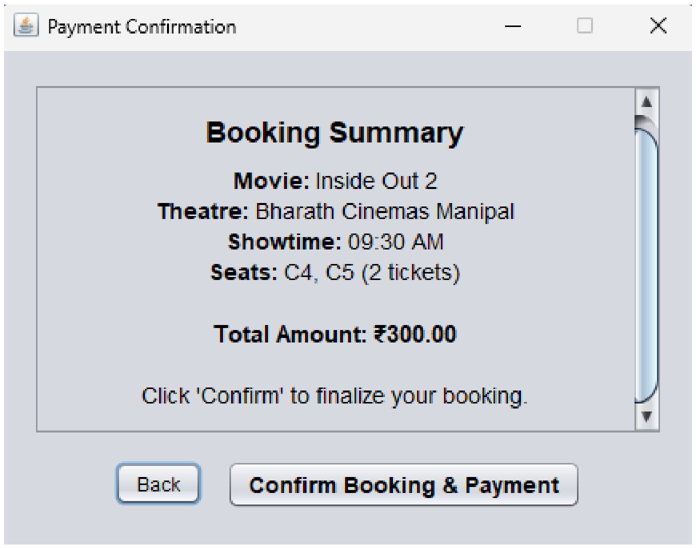
Screenshot 3: Seat Selection Screen


Select Seats for Inside Out 2 (09:30 AM)

Movie: Inside Out 2 | Showtime: 09:30 AM | Selected Seats: 0

Caption: Interactive seat map.

Screenshot 4: Payment Summary Screen


Payment Confirmation

Booking Summary

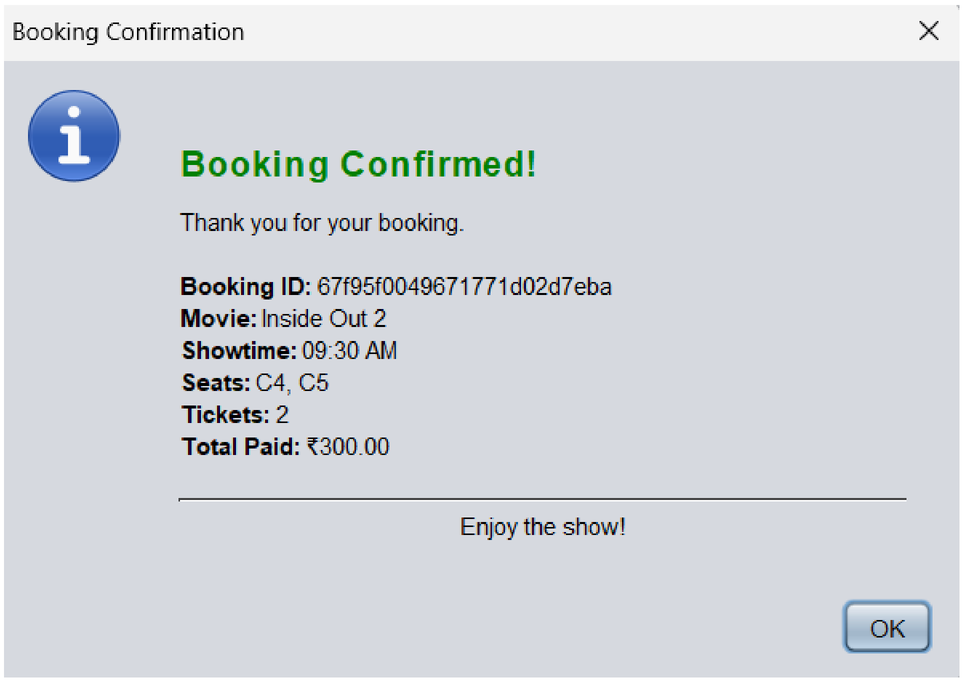
Movie: Inside Out 2
Theatre: Bharath Cinemas Manipal
Showtime: 09:30 AM
Seats: C4, C5 (2 tickets)

Total Amount: ₹300.00

Click 'Confirm' to finalize your booking.

Caption: Booking summary before confirmation.

Screenshot 5: Booking Confirmation Dialog



Booking Confirmed!

Thank you for your booking.

Booking ID: 67f95f0049671771d02d7eba
Movie: Inside Out 2
Showtime: 09:30 AM
Seats: C4, C5
Tickets: 2
Total Paid: ₹300.00

Enjoy the show!

Caption: Confirmation dialog after successful booking.

8. References

- Oracle Java Swing Documentation: https://docs.oracle.com/javase/tutorial/uiswing/
- MongoDB Java Driver Documentation: https://mongodb.github.io/mongo-java-driver/
- MongoDB Manual: https://docs.mongodb.com/manual/
- Apache Maven Project: https://maven.apache.org/
- Visual Studio Code: https://code.visualstudio.com/