CA 3: Experiential Learning

Group Members:

Sr. No.	PRN	Name of Student	Mail id
1	22070122198	Shilpi Biswal	shilpi.biswal.btech2022@sitpune.edu.in
2	22070122199	Shivani Shashidhar	shivani.shashidhar.btech2022@sitpune.edu.in
3	22070122249	Vidhi Binwal	vidhi.binwal.btech2022@sitpune.edu.in

Problem Statement:

Traditional methods of purchasing movie tickets can be difficult and time-consuming, which can often make users frustrated. As a solution to this, our platform seeks to provide a more straightforward, user friendly and convenient method for booking movie tickets and seats, thus enhancing the overall booking experience for the user.

Explanation:

Title: Online Movie Ticket Booking System

Description of System:

Our system is an effective and user-friendly programme created to make the procedure of making movie ticket reservations simpler and less time consuming. Through the provision of a digital platform that enables customers to select movies, choose seats, and make safe payments, it aims to improve the experience of purchasing movie tickets traditionally.

Key Features:

- Movie Selection: Users can look through the list of movies and showtimes to decide the movie they want to see.
- Seat Reservation: The system offers a simple interface that enables customers to choose their seat column and row.
- Users have the choice of paying for their tickets either with a credit card or a UPI payment option.
- E-Ticket Generation: Following a successful reservation, the system generates an eye-catching e-ticket that includes a summary of the movie information, showtime and seat details.

Working Of The System:

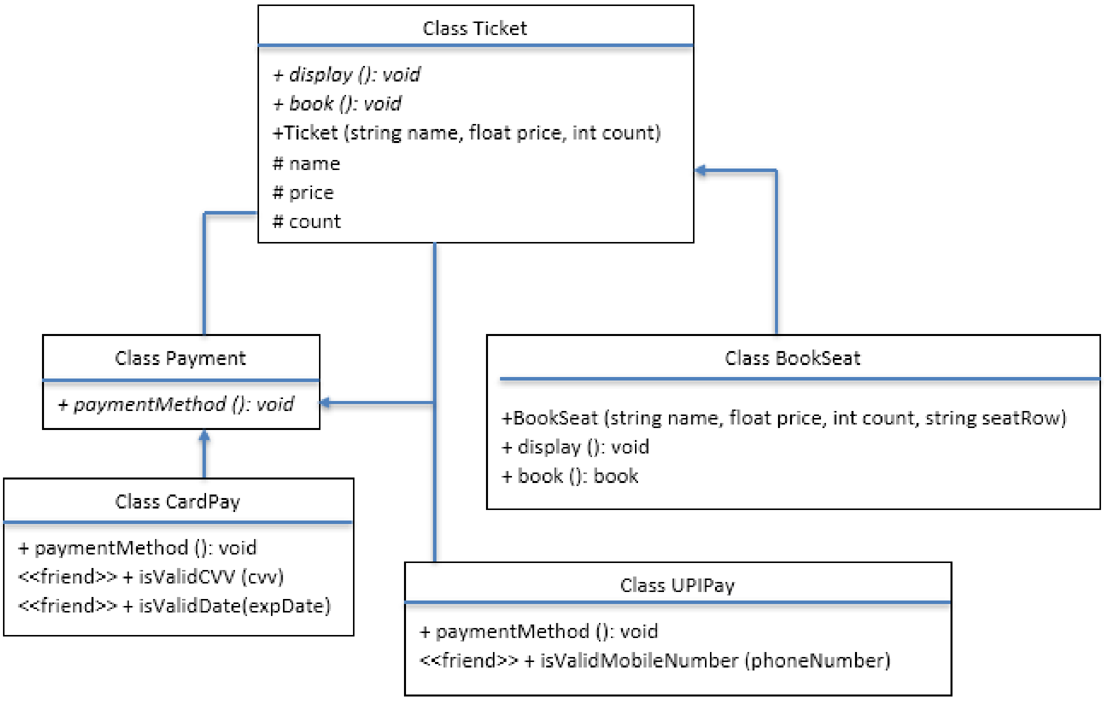
1. User makes decision of selecting desired movie and showtime.
2. They enter the number of tickets they want to buy and choose the row and column in which they want to sit.
3. Users enter payment data, such as credit card information or a UPI phone number.
4. The system creates an electronic ticket containing the movie and showtime information after a successful payment.

Technology Used: C++ for code development

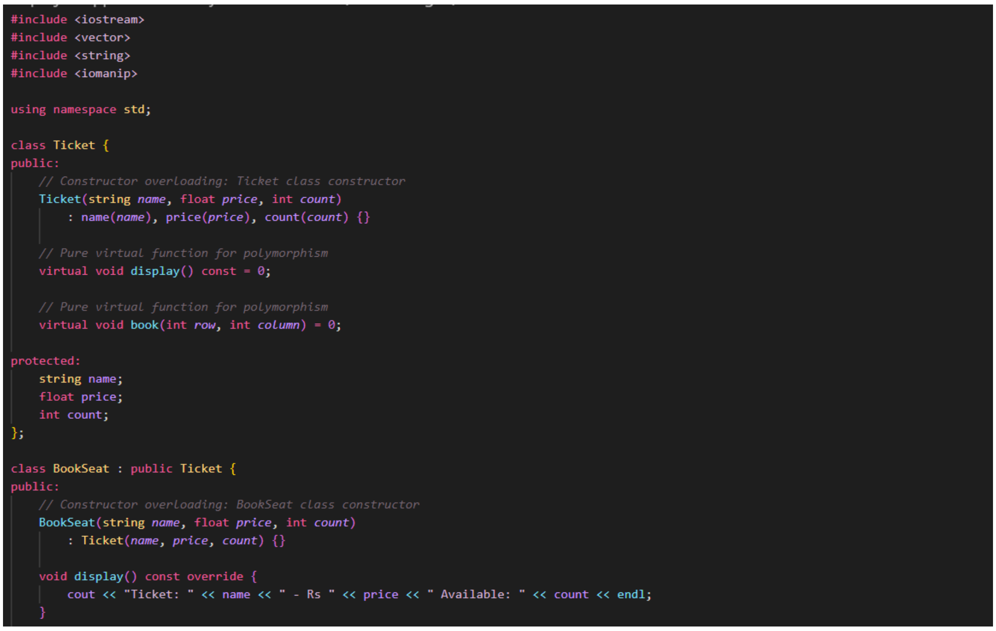
Code Structure:

The code is organised around classes for tickets and payments, and it includes methods for booking, displaying, and processing payments. Additionally, it uses functions for displaying the seating arrangement and generating electronic tickets.

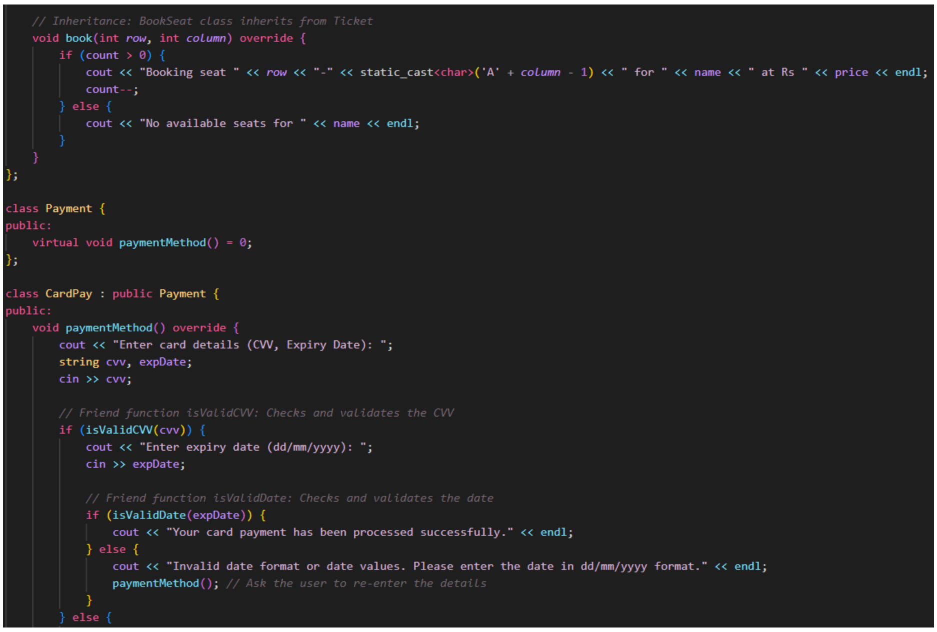
Class Diagram:



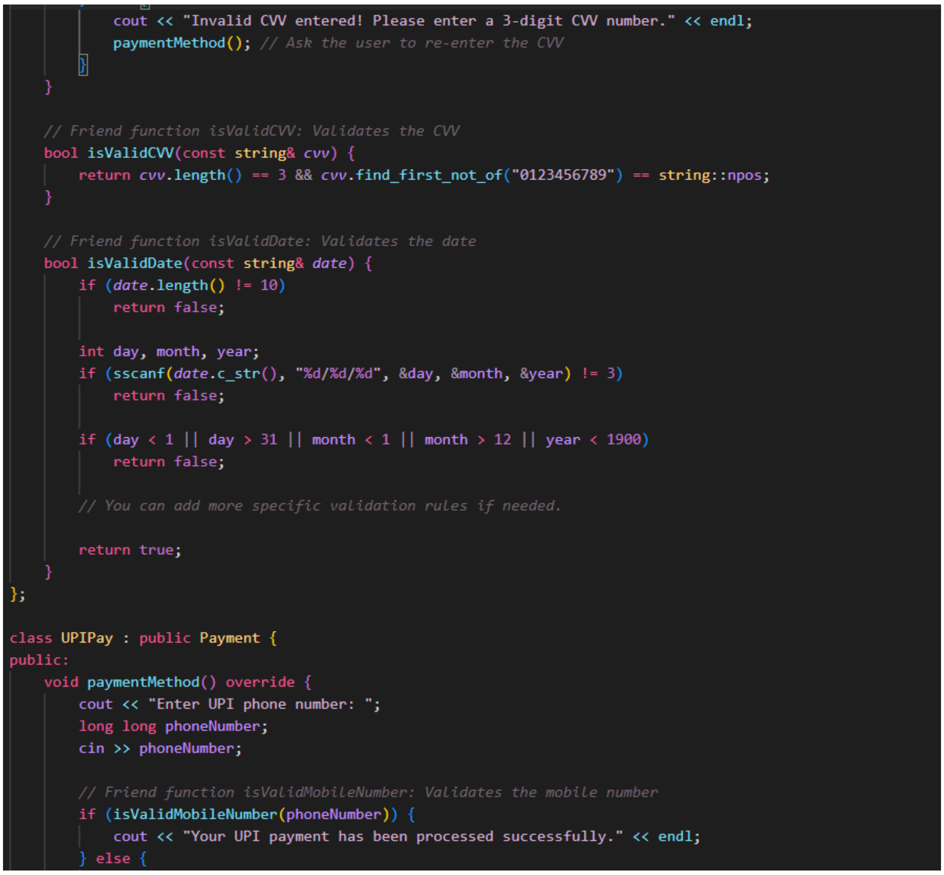
Code snippets:



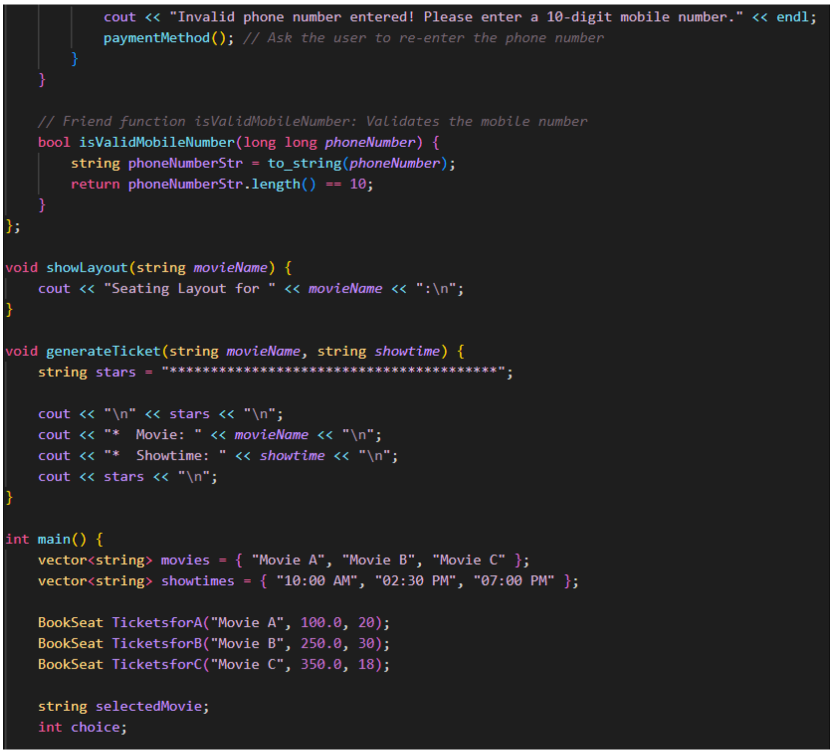
```
#include <iostream>
#include <vector>
#include <string>
#include <iomanip>

using namespace std;

class Ticket {
public:
    // Constructor overloading: Ticket class constructor
    Ticket(string name, float price, int count)
        : name(name), price(price), count(count) {}

    // Pure virtual function for polymorphism
    virtual void display() const = 0;

    // Pure virtual function for polymorphism
    virtual void book(int row, int column) = 0;

protected:
    string name;
    float price;
    int count;
};

class BookSeat : public Ticket {
public:
    // Constructor overloading: BookSeat class constructor
    BookSeat(string name, float price, int count)
        : Ticket(name, price, count) {}

    void display() const override {
        cout << "Ticket: " << name << " - Rs " << price << " Available: " << count << endl;
    }
```



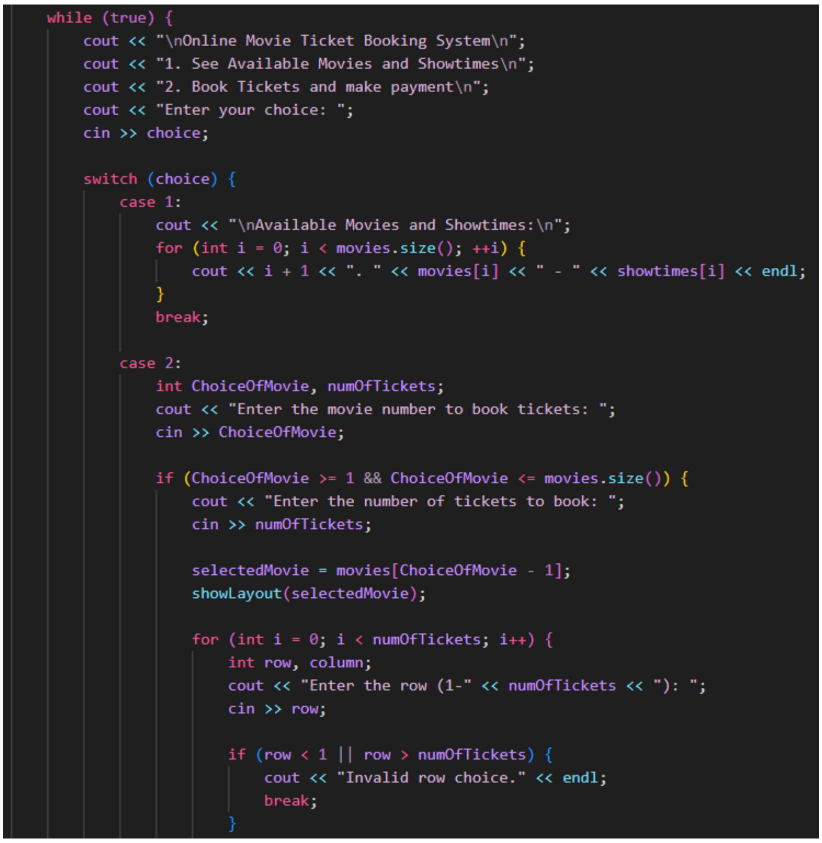
```
    // Inheritance: BookSeat class inherits from Ticket
    void book(int row, int column) override {
        if (count > 0) {
            cout << "Booking seat " << row << "-" << static_cast<char>('A' + column - 1) << " for " << name << " at Rs " << price << endl;
            count--;
        } else {
            cout << "No available seats for " << name << endl;
        }
    }
};

class Payment {
public:
    virtual void paymentMethod() = 0;
};

class CardPay : public Payment {
public:
    void paymentMethod() override {
        cout << "Enter card details (CVV, Expiry Date): ";
        string cvv, expDate;
        cin >> cvv;

        // Friend function isValidCVV: Checks and validates the CVV
        if (isValidCVV(cvv)) {
            cout << "Enter expiry date (dd/mm/yyyy): ";
            cin >> expDate;

            // Friend function isValidDate: Checks and validates the date
            if (isValidDate(expDate)) {
                cout << "Your card payment has been processed successfully." << endl;
            } else {
                cout << "Invalid date format or date values. Please enter the date in dd/mm/yyyy format." << endl;
                paymentMethod(); // Ask the user to re-enter the details
            }
        } else {
```



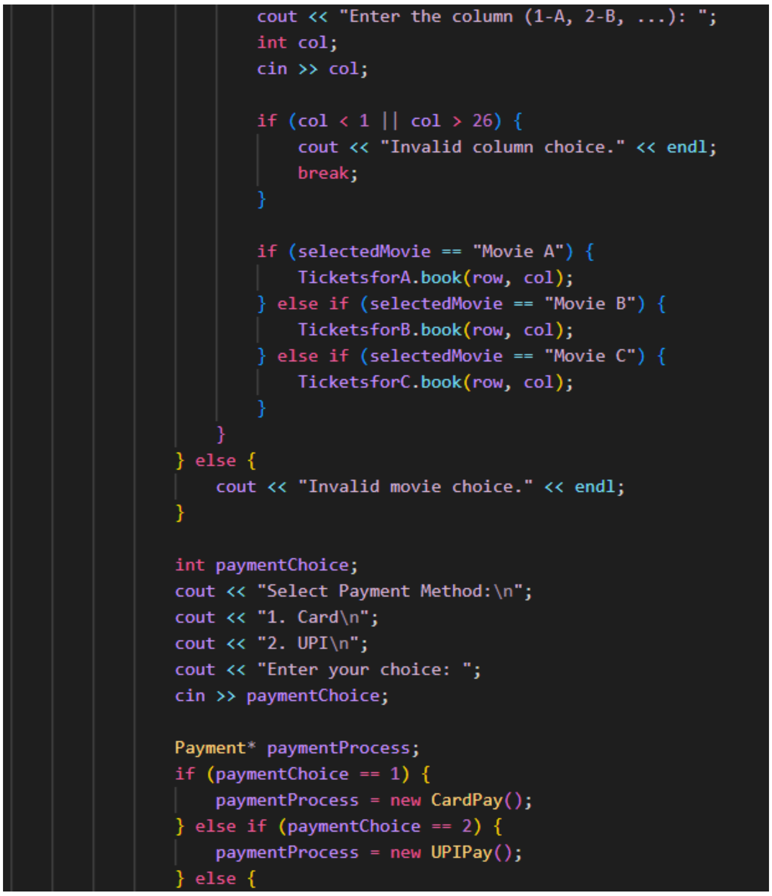
```
            cout << "Invalid CVV entered! Please enter a 3-digit CVV number." << endl;
            paymentMethod(); // Ask the user to re-enter the CVV
        }
    }

    // Friend function isValidCVV: Validates the CVV
    bool isValidCVV(const string& cvv) {
        return cvv.length() == 3 && cvv.find_first_not_of("0123456789") == string::npos;
    }

    // Friend function isValidDate: Validates the date
    bool isValidDate(const string& date) {
        if (date.length() != 10)
            return false;

        int day, month, year;
        if (sscanf(date.c_str(), "%d/%d/%d", &day, &month, &year) != 3)
            return false;

        if (day < 1 || day > 31 || month < 1 || month > 12 || year < 1900)
            return false;

        // You can add more specific validation rules if needed.

        return true;
    }
};

class UPIPay : public Payment {
public:
    void paymentMethod() override {
        cout << "Enter UPI phone number: ";
        long long phoneNumber;
        cin >> phoneNumber;

        // Friend function isValidMobileNumber: Validates the mobile number
        if (isValidMobileNumber(phoneNumber)) {
            cout << "Your UPI payment has been processed successfully." << endl;
        } else {
```



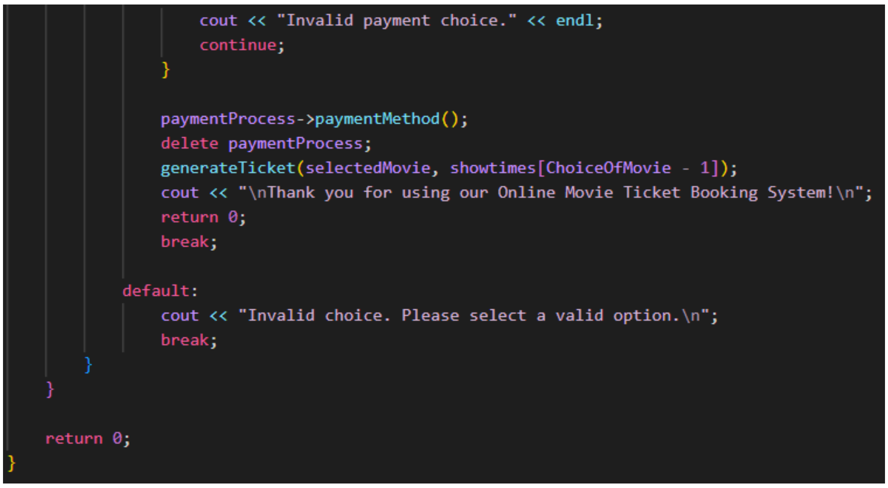
```cpp
            cout << "Invalid phone number entered! Please enter a 10-digit mobile number." << endl;
            paymentMethod(); // Ask the user to re-enter the phone number
        }
    }

    // Friend function isValidMobileNumber: Validates the mobile number
    bool isValidMobileNumber(long long phoneNumber) {
        string phoneNumberStr = to_string(phoneNumber);
        return phoneNumberStr.length() == 10;
    }
};

void showLayout(string movieName) {
    cout << "Seating Layout for " << movieName << ":\n";
}

void generateTicket(string movieName, string showtime) {
    string stars = "****************************************";

    cout << "\n" << stars << "\n";
    cout << "*  Movie: " << movieName << "\n";
    cout << "*  Showtime: " << showtime << "\n";
    cout << stars << "\n";
}

int main() {
    vector<string> movies = { "Movie A", "Movie B", "Movie C" };
    vector<string> showtimes = { "10:00 AM", "02:30 PM", "07:00 PM" };

    BookSeat TicketsforA("Movie A", 100.0, 20);
    BookSeat TicketsforB("Movie B", 250.0, 30);
    BookSeat TicketsforC("Movie C", 350.0, 18);

    string selectedMovie;
    int choice;
```



```cpp
    while (true) {
        cout << "\nOnline Movie Ticket Booking System\n";
        cout << "1. See Available Movies and Showtimes\n";
        cout << "2. Book Tickets and make payment\n";
        cout << "Enter your choice: ";
        cin >> choice;

        switch (choice) {
            case 1:
                cout << "\nAvailable Movies and Showtimes:\n";
                for (int i = 0; i < movies.size(); ++i) {
                    cout << i + 1 << ". " << movies[i] << " - " << showtimes[i] << endl;
                }
                break;

            case 2:
                int ChoiceOfMovie, numOfTickets;
                cout << "Enter the movie number to book tickets: ";
                cin >> ChoiceOfMovie;

                if (ChoiceOfMovie >= 1 && ChoiceOfMovie <= movies.size()) {
                    cout << "Enter the number of tickets to book: ";
                    cin >> numOfTickets;

                    selectedMovie = movies[ChoiceOfMovie - 1];
                    showLayout(selectedMovie);

                    for (int i = 0; i < numOfTickets; i++) {
                        int row, column;
                        cout << "Enter the row (1-" << numOfTickets << "): ";
                        cin >> row;

                        if (row < 1 || row > numOfTickets) {
                            cout << "Invalid row choice." << endl;
                            break;
                        }
```

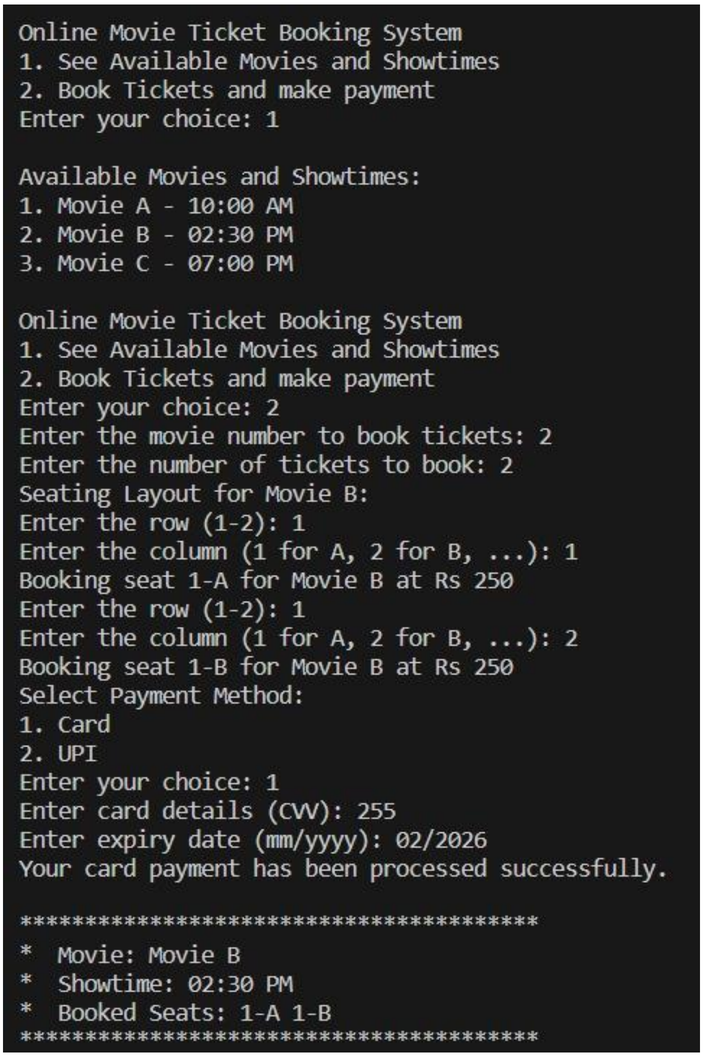


```
                        cout << "Enter the column (1-A, 2-B, ...): ";
                        int col;
                        cin >> col;

                        if (col < 1 || col > 26) {
                            cout << "Invalid column choice." << endl;
                            break;
                        }

                        if (selectedMovie == "Movie A") {
                            TicketsforA.book(row, col);
                        } else if (selectedMovie == "Movie B") {
                            TicketsforB.book(row, col);
                        } else if (selectedMovie == "Movie C") {
                            TicketsforC.book(row, col);
                        }
                    }
                } else {
                    cout << "Invalid movie choice." << endl;
                }

                int paymentChoice;
                cout << "Select Payment Method:\n";
                cout << "1. Card\n";
                cout << "2. UPI\n";
                cout << "Enter your choice: ";
                cin >> paymentChoice;

                Payment* paymentProcess;
                if (paymentChoice == 1) {
                    paymentProcess = new CardPay();
                } else if (paymentChoice == 2) {
                    paymentProcess = new UPIPay();
                } else {
```



```
                    cout << "Invalid payment choice." << endl;
                    continue;
                }

                paymentProcess->paymentMethod();
                delete paymentProcess;
                generateTicket(selectedMovie, showtimes[ChoiceOfMovie - 1]);
                cout << "\nThank you for using our Online Movie Ticket Booking System!\n";
                return 0;
                break;

            default:
                cout << "Invalid choice. Please select a valid option.\n";
                break;
        }
    }

    return 0;
}
```

Input/Output:



```
Online Movie Ticket Booking System
1. See Available Movies and Showtimes
2. Book Tickets and make payment
Enter your choice: 1

Available Movies and Showtimes:
1. Movie A - 10:00 AM
2. Movie B - 02:30 PM
3. Movie C - 07:00 PM

Online Movie Ticket Booking System
1. See Available Movies and Showtimes
2. Book Tickets and make payment
Enter your choice: 2
Enter the movie number to book tickets: 2
Enter the number of tickets to book: 2
Seating Layout for Movie B:
Enter the row (1-2): 1
Enter the column (1 for A, 2 for B, ...): 1
Booking seat 1-A for Movie B at Rs 250
Enter the row (1-2): 1
Enter the column (1 for A, 2 for B, ...): 2
Booking seat 1-B for Movie B at Rs 250
Select Payment Method:
1. Card
2. UPI
Enter your choice: 1
Enter card details (CVV): 255
Enter expiry date (mm/yyyy): 02/2026
Your card payment has been processed successfully.

****************************************
*  Movie: Movie B
*  Showtime: 02:30 PM
*  Booked Seats: 1-A 1-B
****************************************
```

Github repository link:

https://github.com/c0upsattaca/Movie-Ticket-Booking