ONLINE TICKET RESERVATION SYSTEM

1. Introduction

The Online TicketReservation System is a console-based software application developed using the C++ programming language. In modern times, ticket booking for movies, concerts, and sports events is mostly done online to save time and reduce manual effort. This project aims to simulate such a system in a simple and user-friendly way.

The system allows users to view available events, book tickets, and automatically update seat availability. The project focuses on implementing Object-Oriented Programming (OOP) concepts using only basic C++ libraries, making it suitable for beginners and academic learning.

2. Topic Description

Thisproject is designed to manage ticket reservations for different types of events. The system works through a menu-driven interface, where users can select options by entering their choice.

Main features of the system include:

- Displaying a list of available events with seat count and price
- Allowing users to book tickets for a selected event
- Checking seat availability before booking
- Updating remaining seats after successful booking
- Calculating the total cost of tickets

The system prevents invalid operations such as booking more seats than available or entering incorrect seat numbers.

3. Object-Oriented Programming (OOP) Concepts Used

1. Class and Object

A class named Event is created to represent an event.

Each event has its own data such as event ID, name, available seats, and ticket price.

Objects of the Event class represent individual events like movies, concerts, & sports matches.

2. Encapsulation

All data members of the Event class are declared as private.

This protects the data from direct access.

Public member functions are used to access and modify these values safely.

3. Constructor

Constructors are used to initialize event objects. A parameterized constructor assigns values to event ID, name, seats, and price at the time of object creation.

4. Data Abstraction

The internal details of seat booking and data handling are hidden from the user. The user interacts with the system using simple functions such as displaying events and booking tickets.

5. Array of Objects

An array of Event objects is used to store multiple events. This allows easy management of more than one event without using advanced data structures.

4. Implementation

The project is implemented using **C++** with only two header files:

- iostream for input and output
  string for handling text data

The Event class stores all event-related information and provides functions to display event details and book seats. The bookSeat() function checks whether the requested number of seats is valid and available before completing the booking.

The main() function controls the program flow. It initializes event data, displays a menu repeatedly using a loop, takes user input, and performs the required operation based on the selected option. After booking, the remaining seats are updated and displayed to the user.

- **CODE**

```
#include <iostream>

#include <string>

using namespace std;


// ----------------- Event Class -----------------

class Event { private:

   int id; string

   name; int

   seats; float

   price;
```

```
public:
  Event() {
    id    =    0;
    name  =  "";
    seats  =  0;
    price = 0;
  }

  Event(inti,string n, int s, float p) {
    id= i;
    name= n;
    seats = s;
    price=p;
  }

  voidshowEvent() {
    cout<<"ID:" << id
       <<"|Event: " << name
       <<"|Seats: " << seats
       <<"|Price: " << price << endl;
  }

  int getId() {
    return id;
    }

    intgetSeats() {
        return seats;
    }

    floatgetPrice() {
        return price;
    }

    stringgetName() {
        return name;
    }

    boolbookSeat(int s) {
        if(s> 0 && s <= seats) {
            seats = seats - s;
            return true;
        }
        return false;
    }
};

//------------------ Main ------------------
```

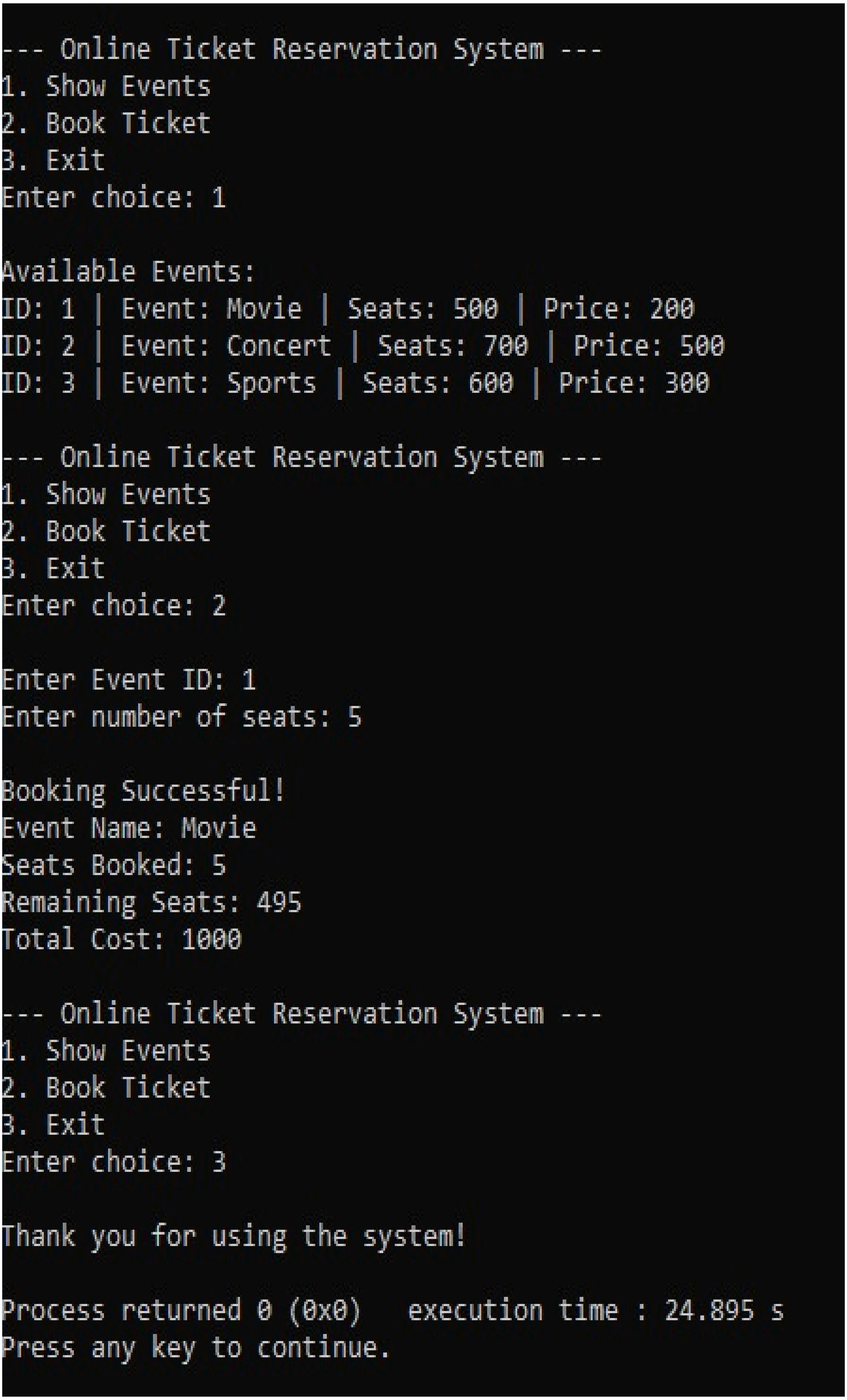
```
int main() {
   Event events[3];

   // Initialdata
   events[0]=Event(1,"Movie",500, 200);
   events[1]=Event(2,"Concert",700, 500);
   events[2]=Event(3,"Sports",600, 300);

   int choice;

   do{

   cout<<"\n---Online Ticket Reservation System ---\n";
   cout<<"1.ShowEvents\n";
   cout<<"2.BookTicket\n";
   cout<<"3.Exit\n";
   cout<<"Enterchoice: ";
   cin>>choice;



       if(choice==1) {
          cout<<"\nAvailable Events:\n";
          for(inti=0;i<3; i++) {
             events[i].showEvent();
          }
       }
```

```
elseif(choice==2) {
inteid,seatCount;
cout<<"\nEnter Event ID: ";
cin>>eid;
cout<<"Enternumber of seats: ";
cin>>seatCount;

    boolfound=false;

    for(inti=0;i<3; i++) {
        if(events[i].getId() == eid) {
            found=true;

            if(events[i].bookSeat(seatCount)) {
                cout<<"\nBooking Successful!";
                cout<<"\nEvent Name: " << events[i].getName();
                cout<<"\nSeats Booked: " << seatCount;
                cout<<"\nRemaining Seats: "
                     <<events[i].getSeats();
                cout<<"\nTotal Cost: "
                     <<seatCount * events[i].getPrice() << endl;
            }else {
                cout<<"\nInvalid seat number or not enough seats!\n";
            }
```

```
                    break;
                }
            }

            if(!found) {
                cout << "\nEvent not found!\n";
            }
        }
        elseif(choice == 3) {
            cout<< "\nThank you for using the system!\n";
        }
        else {
            cout<< "\nInvalid choice!\n";
        }

    }while(choice != 3);

    return 0;
}
```

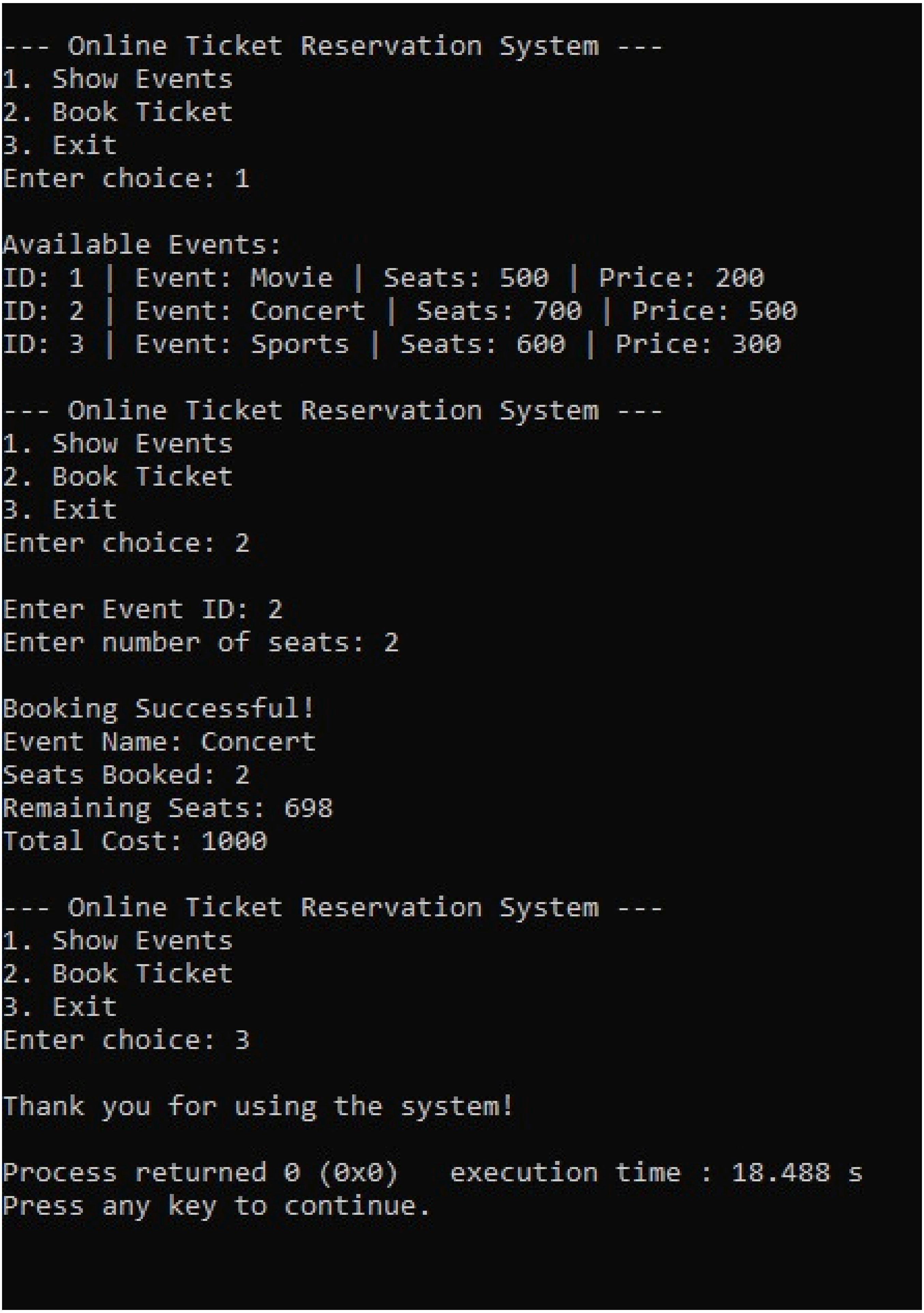
5. Output

Output 1:



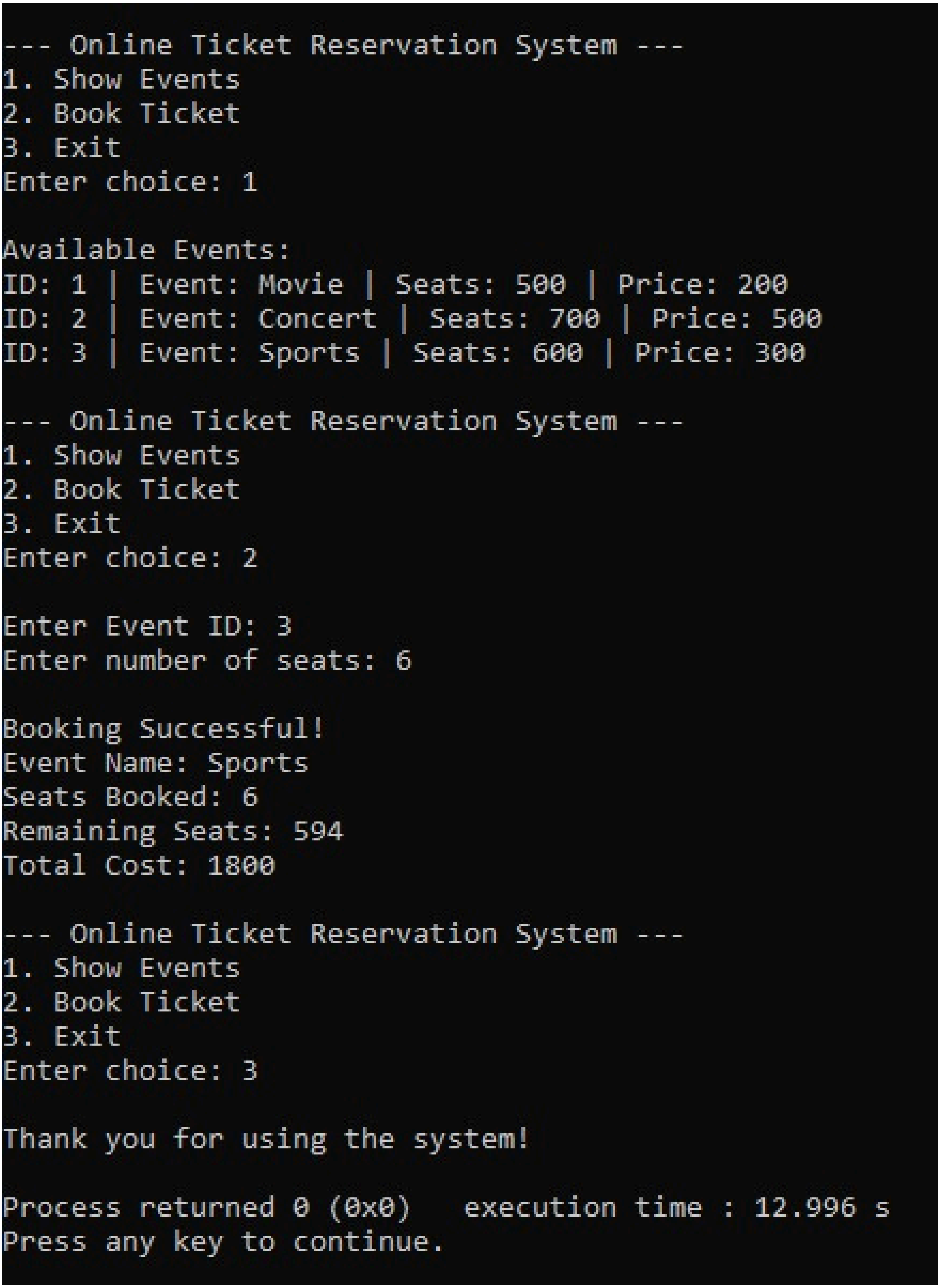
```
--- Online Ticket Reservation System ---
1. Show Events
2. Book Ticket
3. Exit
Enter choice: 1

Available Events:
ID: 1 | Event: Movie | Seats: 500 | Price: 200
ID: 2 | Event: Concert | Seats: 700 | Price: 500
ID: 3 | Event: Sports | Seats: 600 | Price: 300

--- Online Ticket Reservation System ---
1. Show Events
2. Book Ticket
3. Exit
Enter choice: 2

Enter Event ID: 1
Enter number of seats: 5

Booking Successful!
Event Name: Movie
Seats Booked: 5
Remaining Seats: 495
Total Cost: 1000

--- Online Ticket Reservation System ---
1. Show Events
2. Book Ticket
3. Exit
Enter choice: 3

Thank you for using the system!

Process returned 0 (0x0)   execution time : 24.895 s
Press any key to continue.
```

Output 2:



```
--- Online Ticket Reservation System ---
1. Show Events
2. Book Ticket
3. Exit
Enter choice: 1

Available Events:
ID: 1 | Event: Movie | Seats: 500 | Price: 200
ID: 2 | Event: Concert | Seats: 700 | Price: 500
ID: 3 | Event: Sports | Seats: 600 | Price: 300

--- Online Ticket Reservation System ---
1. Show Events
2. Book Ticket
3. Exit
Enter choice: 2

Enter Event ID: 2
Enter number of seats: 2

Booking Successful!
Event Name: Concert
Seats Booked: 2
Remaining Seats: 698
Total Cost: 1000

--- Online Ticket Reservation System ---
1. Show Events
2. Book Ticket
3. Exit
Enter choice: 3

Thank you for using the system!

Process returned 0 (0x0)   execution time : 18.488 s
Press any key to continue.
```

Output 3:



```
--- Online Ticket Reservation System ---
1. Show Events
2. Book Ticket
3. Exit
Enter choice: 1

Available Events:
ID: 1 | Event: Movie | Seats: 500 | Price: 200
ID: 2 | Event: Concert | Seats: 700 | Price: 500
ID: 3 | Event: Sports | Seats: 600 | Price: 300

--- Online Ticket Reservation System ---
1. Show Events
2. Book Ticket
3. Exit
Enter choice: 2

Enter Event ID: 3
Enter number of seats: 6

Booking Successful!
Event Name: Sports
Seats Booked: 6
Remaining Seats: 594
Total Cost: 1800

--- Online Ticket Reservation System ---
1. Show Events
2. Book Ticket
3. Exit
Enter choice: 3

Thank you for using the system!

Process returned 0 (0x0)   execution time : 12.996 s
Press any key to continue.
```

6. Conclusion

TheOnlineTicket Reservation System is a simple yet effective application that demonstrates the use of Object-Oriented Programming concepts in C++. The project successfully manages event details, ticket booking, and seat availability using basic programming constructs.

This project helps in understanding real-world application development using OOP principles. It can be further enhanced by adding features such as ticket cancellation, file handling, user authentication, or graphical interfaces.