Hussein Madan Migadde 27403645 COS1511 Assignment 2 777488

Input:

```
#include <iostream>
#include <string>
#include <vector>
#include <iomanip>
#include <cctype>
#include <limits>
#include <algorithm>
using namespace std;
// Constants
const int NUM_SEATS = 50;
const int FIRST_CLASS_ROWS = 4;
const double ECONOMY_COST = 1600.00;
const double FIRST_CLASS_MULTIPLIER = 1.20;
const int TICKET_WIDTH = 60; // Define a constant for ticket width
// Flight structure to hold flight details
struct Flight {
    string departureTime;
    vector<bool> seats;
    int bookings;
    Flight(string time) : departureTime(time), seats(NUM_SEATS,
false), bookings(0) {}
};
// Booking structure to hold each booking's details
struct Booking {
    string fullName;
    int flightIndex;
    int seatNumber;
};
// Function to display banner
void displayBanner(const string& title, int width) {
    string border(width, '*');
    cout << border << endl;
    cout << "*" << string(width - 2, ' ') << "*" << endl;
    // Calculate centering
    int padding = (width - 2 - title.length()) / 2;
    string paddedTitle = string(padding, ' ') + title +
string(padding, ' ');
    // Adjust if odd length
    if ((width - 2 - title.length()) % 2 != 0) {
        paddedTitle += " ";
    }
    cout << "*" << paddedTitle << "*" << endl;
    cout << "*" << string(width - 2, ' ') << "*" << endl;
    cout << border << endl;
}
// Function to print banner
void printAsterisks(int count) {
    for (int i = 0; i < count; ++i) {
        cout << "*";
    }
    cout << endl;
}
void displayTicketBanner() {
```

```
    printAsterisks(TICKET_WIDTH);
    cout << "Travel ticket for FLIGHT" << endl;
    printAsterisks(TICKET_WIDTH);
}
void displayTicketBanner2(double ticketPrice) {
    printAsterisks(TICKET_WIDTH);
    cout << "Amount: R" << ticketPrice << " Thank you for booking
with COS1511. "
         << "\nYour travel agent for this query is Hussein Madan" <<
endl;
    printAsterisks(TICKET_WIDTH);
}
// Function to print a line of dashes of given length
void printDashLine(int length) {
    for (int i = 0; i < length; ++i) {
        cout << "-";
    }
    cout << "\n";
}
// Function prototypes
void displayMenu(const vector<Flight>& flights);
void displaySeating(const Flight& flight);
int getSeatNumber(const Flight& flight);
void bookSeat(Flight& flight, int seatNumber);
void displayBookingTicket(const string& fullName, const Flight&
flight,
                          int seatNumber);
void displayBookingSummary(const vector<Flight>& flights,
                           const vector<Booking>& bookings);
// Function to validate name
bool isValidName(const string& name) {
    return all_of(name.begin(), name.end(), [](char c) {
        return isalpha(c) || isspace(c);
    });
}
int main() {
    // Initialize flight times
    vector<Flight> flights;
    flights.push_back(Flight("07:00"));
    flights.push_back(Flight("09:00"));
    flights.push_back(Flight("11:00"));
    flights.push_back(Flight("13:00"));
    flights.push_back(Flight("15:00"));
    vector<Booking> bookings; // Store all bookings
    string fullName;
    char continueBooking;
    displayBanner("Welcome to COS1511 Flight Booking System", 50);
    do {
        cout << "Enter full name: ";
        getline(cin, fullName);
        if (fullName.empty()) {
            cout << "Full name cannot be empty. Please try again.\n";
        } else if (!isValidName(fullName)) {
            cout << "Invalid name! Please enter only letters"
                 << " and spaces.\n";
        } else {
            break; // Exit the loop if the name is valid
        }
    } while (true);
    do {
        displayMenu(flights);
```

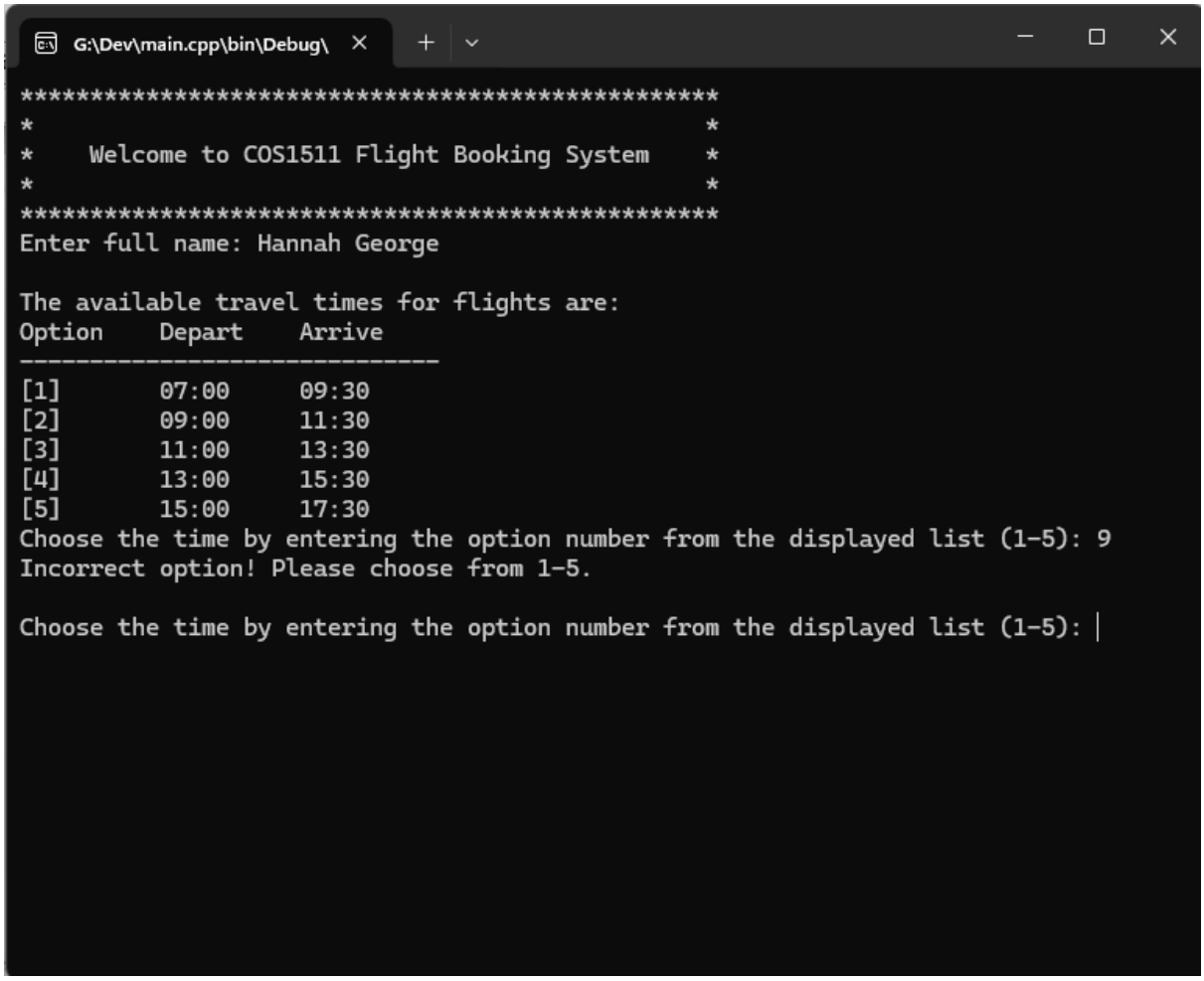
```
        int choice;
        while (true) {
            cout << "Choose the time by entering the option number"
                 << " from the displayed list (1-5): ";
            cin >> choice;
            if (cin.fail() || choice < 1 || choice > 5) {
                cin.clear();
                cin.ignore(numeric_limits<streamsize>::max(), '\n');
                cout << "Incorrect option! Please choose from"
                     << " 1-5.\n" << endl;
            } else {
                cin.ignore(numeric_limits<streamsize>::max(), '\n');
                break;
            }
        }
        // Validate input range 1-5
        while (choice < 1 || choice > 5) {
            cout << "Incorrect option! Please choose from 1-5: ";
            cin >> choice;
        }
        Flight& selectedFlight = flights[choice - 1];
        displaySeating(selectedFlight);
        int seatNumber = getSeatNumber(selectedFlight);
        bookSeat(selectedFlight, seatNumber);
        displayBookingTicket(fullName, selectedFlight, seatNumber);
        selectedFlight.bookings++;
        // Store booking details
        Booking newBooking;
        newBooking.fullName = fullName;
        newBooking.flightIndex = choice - 1;
        newBooking.seatNumber = seatNumber;
        bookings.push_back(newBooking);
        cout << "Do you want to make another booking? (Y/N): ";
        cin >> continueBooking;
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
        if (toupper(continueBooking) == 'Y') {
            do {
                cout << "Enter full name: ";
                getline(cin, fullName);
                if (fullName.empty()) {
                    cout << "Full name cannot be empty. Please try
again.\n";
                } else if (!isValidName(fullName)) {
                    cout << "Invalid name! Please enter only letters"
                         << " and spaces.\n";
                } else {
                    break; // Exit the loop if the name is valid
                }
            } while (true);
        }
    } while (toupper(continueBooking) == 'Y');
    displayBookingSummary(flights, bookings);
    cout << "\nThank you for using the COS1511 Flight Booking
System!\n";
    return 0;
}
// Function to display the flight menu
void displayMenu(const vector<Flight>& flights) {
    cout << "\nThe available travel times for flights are:\n";
    cout << left << setw(10) << "Option" << setw(10) << "Depart" <<
setw(10)
```

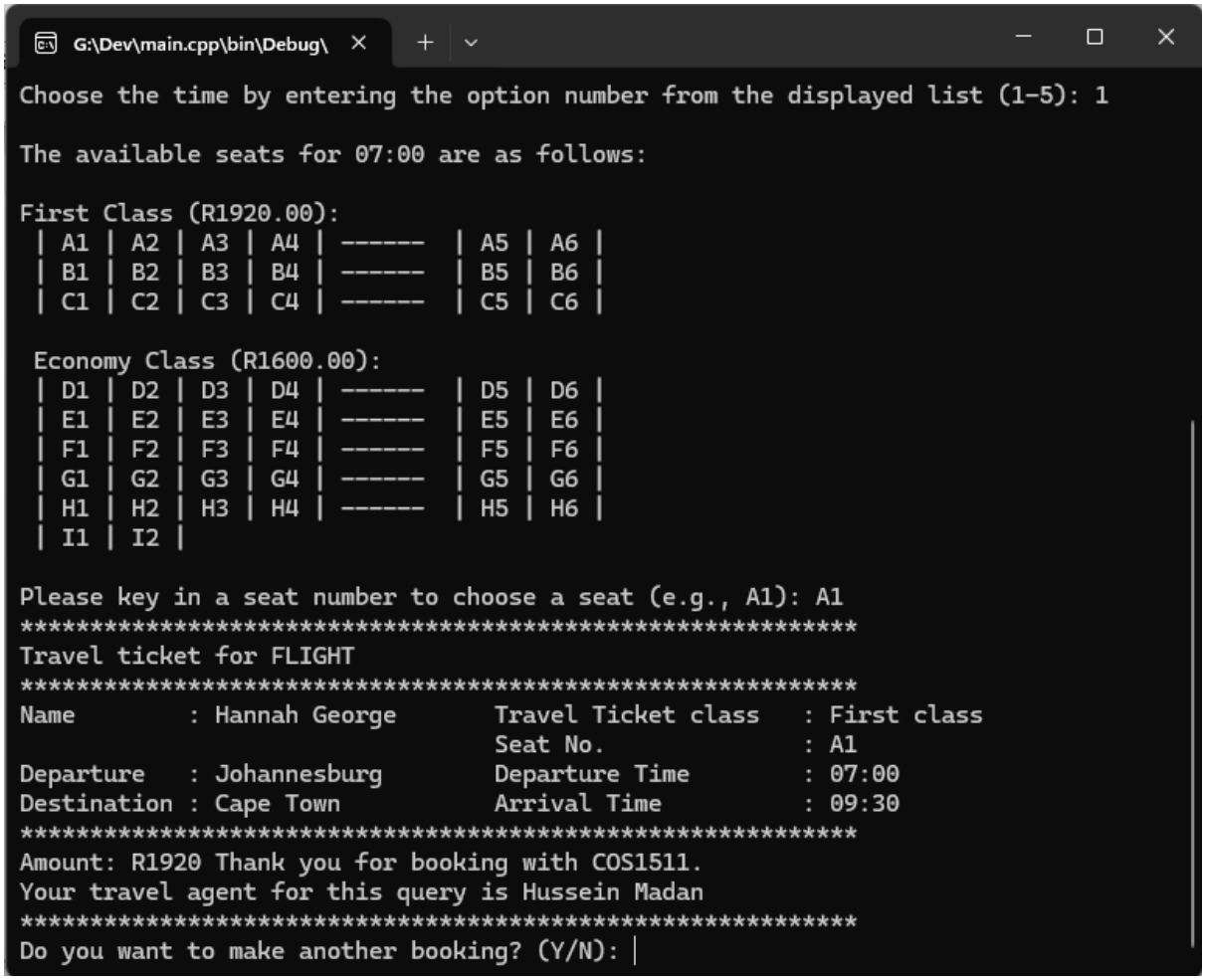
```
         << "Arrive" << endl;
    printDashLine(30);
    for (size_t i = 0; i < flights.size(); ++i) {
        int departureHour = stoi(flights[i].departureTime.substr(0,
2));
        int arrivalHour = (departureHour + 2) % 24;
        string arrivalTime = (arrivalHour < 10 ? "0" : "") +
to_string(arrivalHour) + ":30";
        cout << left << setw(10) << ("[" + to_string(i + 1) + "]") <<
setw(10)
             << flights[i].departureTime << setw(10) << arrivalTime
<< endl;
    }
}
// Function to display the seating arrangement
void displaySeating(const Flight& flight) { cout << "\nThe available
seats for " << flight.departureTime << " are as follows:\n\n";
char rowLabel = 'A';
    for (int i = 0; i < NUM_SEATS; ++i) {
        // Show section labels
        if (i == 0) {
            cout << "First Class (R1920.00):\n";
        }
        if (i == 18) {
            cout << "\n Economy Class (R1600.00):\n";
        }
        int seatInRow = i % 6;
        string seatLabel = string(1, rowLabel) + to_string(seatInRow
+ 1);
        cout << " | " << (flight.seats[i] ? "**" : seatLabel);
        // Aisle separator after seat 3
        if (seatInRow == 3) {
            cout << " | ------ ";
        }
        // End of row OR last seat in the array
        if (seatInRow == 5 || i == NUM_SEATS - 1) {
            cout << " |\n";
            rowLabel++;
        }
    }
}
// Function to get the seat number from the user
int getSeatNumber(const Flight& flight) {
    string seatInput;
    bool validSeat = false;
    int seatNumber = 0;
    while (!validSeat) {
        // Check if any seat is booked
bool anyBooked = false;
for (bool booked : flight.seats) {
    if (booked) {
        anyBooked = true;
        break;
    }
}
if (anyBooked) {
    cout << "\nSeats that are already taken are indicated with an
asterisk (**)";
}
```

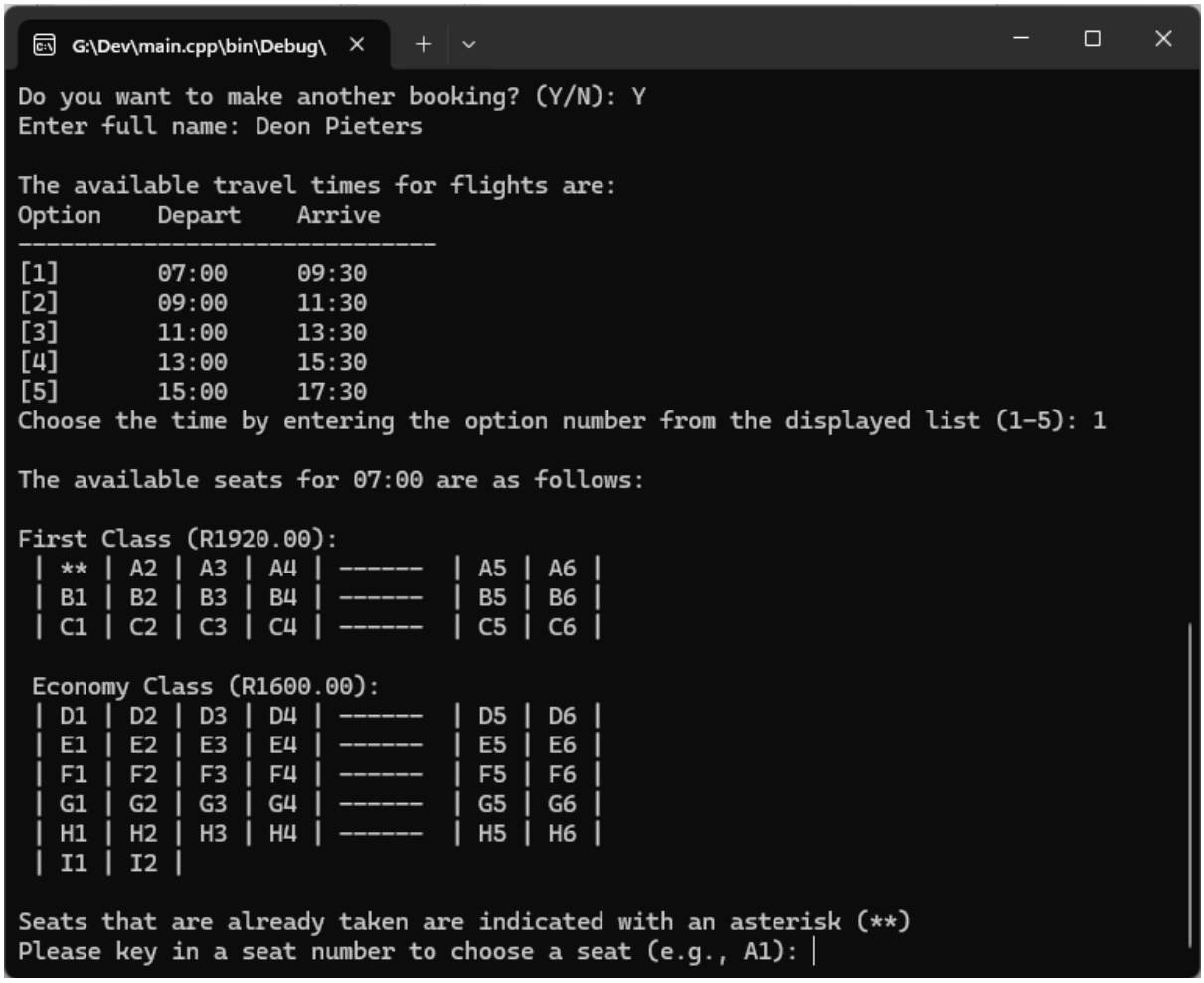
```
        cout << "\nPlease key in a seat number to choose a seat
(e.g., A1): ";
        cin >> seatInput;
        cin.ignore();
        // Validate input format (e.g., A1, B2 etc.)
        if (seatInput.length() != 2) {
            cout << "Invalid format! Please enter a seat number like
A1,"
                 << " B2, etc.\n";
            continue;
        }
        char row = toupper(seatInput[0]);
        int column = seatInput[1] - '0';
        // Validate row (A-I) and column (1-6)
        if (row < 'A' || row > 'I') {
            cout << "Invalid row! Please choose from A to I.\n";
            continue;
        }
        if (column < 1 || column > 6) {
            cout << "Invalid column! Please choose from 1 to 6.\n";
            continue;
        }
        // Calculate the seat index in the vector
        int rowIndex = row - 'A';
        seatNumber = rowIndex * 6 + (column - 1);
        // Check if seat number exceeds total seats (0-49)
        if (seatNumber >= NUM_SEATS) {
            cout << "Invalid seat! This seat does not exist.\n";
            continue;
        }
        // Check if the seat is already booked
        if (flight.seats[seatNumber]) {
            cout << "That seat is already booked. Please choose
another seat.\n";
        } else {
            validSeat = true;
        }
    }
    return seatNumber + 1; // Return 1 - based seat number
}
// Function to book the seat
void bookSeat(Flight& flight, int seatNumber) {
    flight.seats[seatNumber - 1] = true;
}
// Function to display the booking ticket
void displayBookingTicket(const string& fullName, const Flight&
flight,
                          int seatNumber) {
    double ticketPrice;
    string seatClass;
    // Convert seat number to letter + number format
    char row = 'A' + (seatNumber - 1) / 6;
    int column = ((seatNumber - 1) % 6) + 1;
    string seatLabel = string(1, row) + to_string(column);
    if (seatNumber <= FIRST_CLASS_ROWS * 6) {
        ticketPrice = ECONOMY_COST * FIRST_CLASS_MULTIPLIER;
        seatClass = "First class";
    } else {
        ticketPrice = ECONOMY_COST;
        seatClass = "Economy";
    }
```

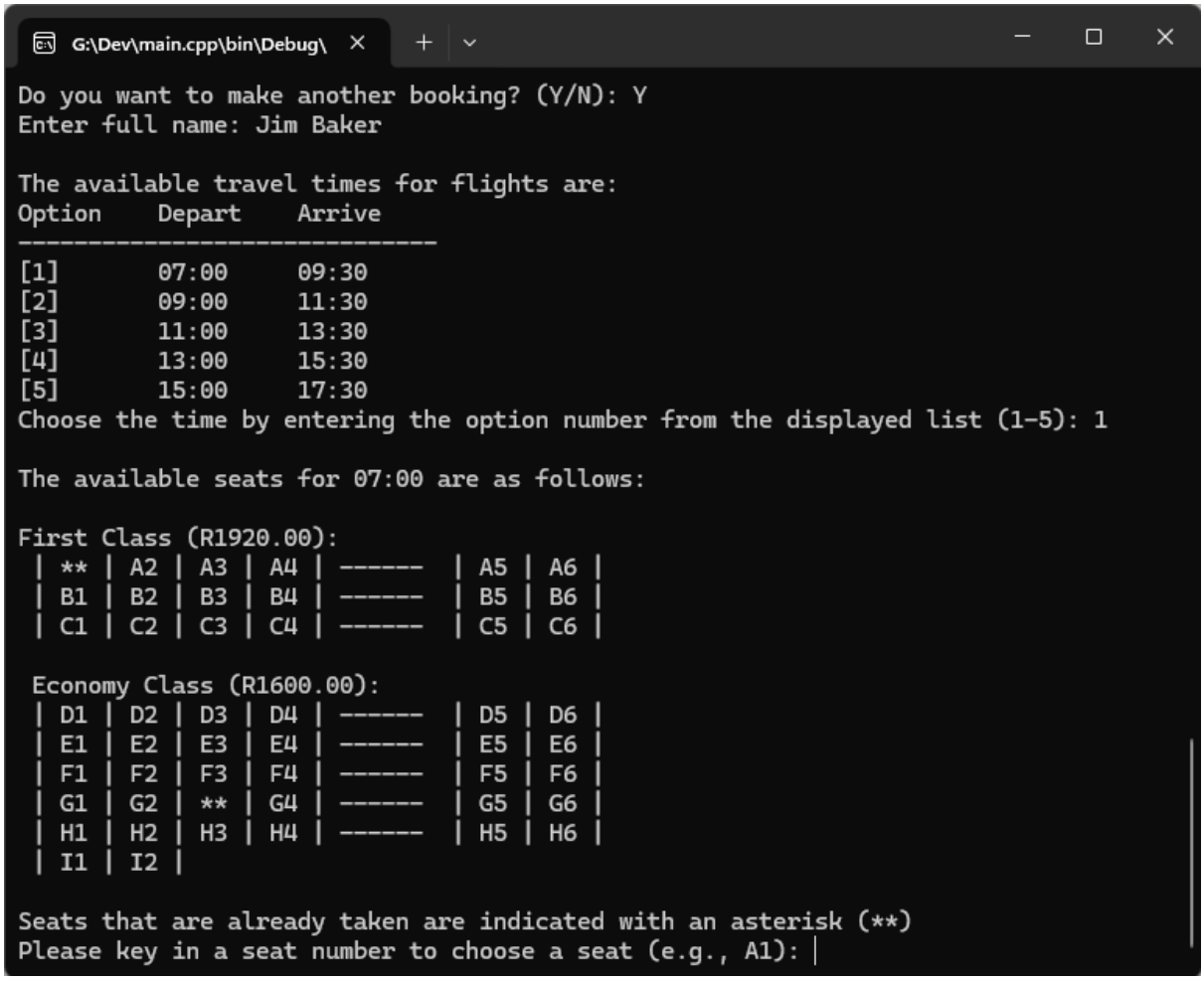
```
    int departureHour = stoi(flight.departureTime.substr(0, 2));
    // Calculate arrival time (2.5 hours)
    int arrivalHour = (departureHour + 2) % 24;
    int arrivalMinute =  30; //Always 30 minutes
    // Convert to string with leading zero if needed
    string arrivalHourStr = (arrivalHour < 10 ? "0" : "") +
to_string(arrivalHour);
    string arrivalMinuteStr = (arrivalMinute < 10 ? "0" : "") +
to_string(arrivalMinute);
    displayTicketBanner();
    cout << left << setw(12) << "Name" << ": " << setw(20) <<
fullName << "Travel Ticket class \t: "
         << seatClass << endl;
    cout << left << setw(34) << "" << "Seat No." << "\t\t: " <<
seatLabel << endl;
    cout << left << setw(12) << "Departure" << ": " << setw(20) <<
"Johannesburg"
         << "Departure Time \t: " << flight.departureTime << endl;
    cout << left << setw(12) << "Destination" << ": " << setw(20) <<
"Cape Town"
         << "Arrival Time \t\t: " << arrivalHourStr << ":" <<
arrivalMinuteStr << endl;
    displayTicketBanner2(ticketPrice);
}
// Function to display booking summary (with detailed bookings)
void displayBookingSummary(const vector<Flight>& flights,
                           const vector<Booking>& bookings) {
    cout << "\nBooking Summary:\n";
    printDashLine(39);
    // Display total bookings per flight
    for (size_t i = 0; i < flights.size(); ++i) {
        cout << "Number of bookings made for " <<
flights[i].departureTime
             << ": " << flights[i].bookings << "\n";
    }
    cout << "\nDetailed Bookings:\n";
    printDashLine(39);
    if (bookings.empty()) {
        cout << "No bookings have been made.\n";
        return;
    }
    for (const Booking& b : bookings) {
        const Flight& flight = flights[b.flightIndex];
        // Convert seat number to letter + number format
        char row = 'A' + (b.seatNumber - 1) / 6;
        int column = ((b.seatNumber - 1) % 6) + 1;
        string seatLabel = string(1, row) + to_string(column);
        cout << left << setw(25) << b.fullName
             << "Flight: " << setw(6) << flight.departureTime
             << " Seat: " << seatLabel << endl;
    }
}
```

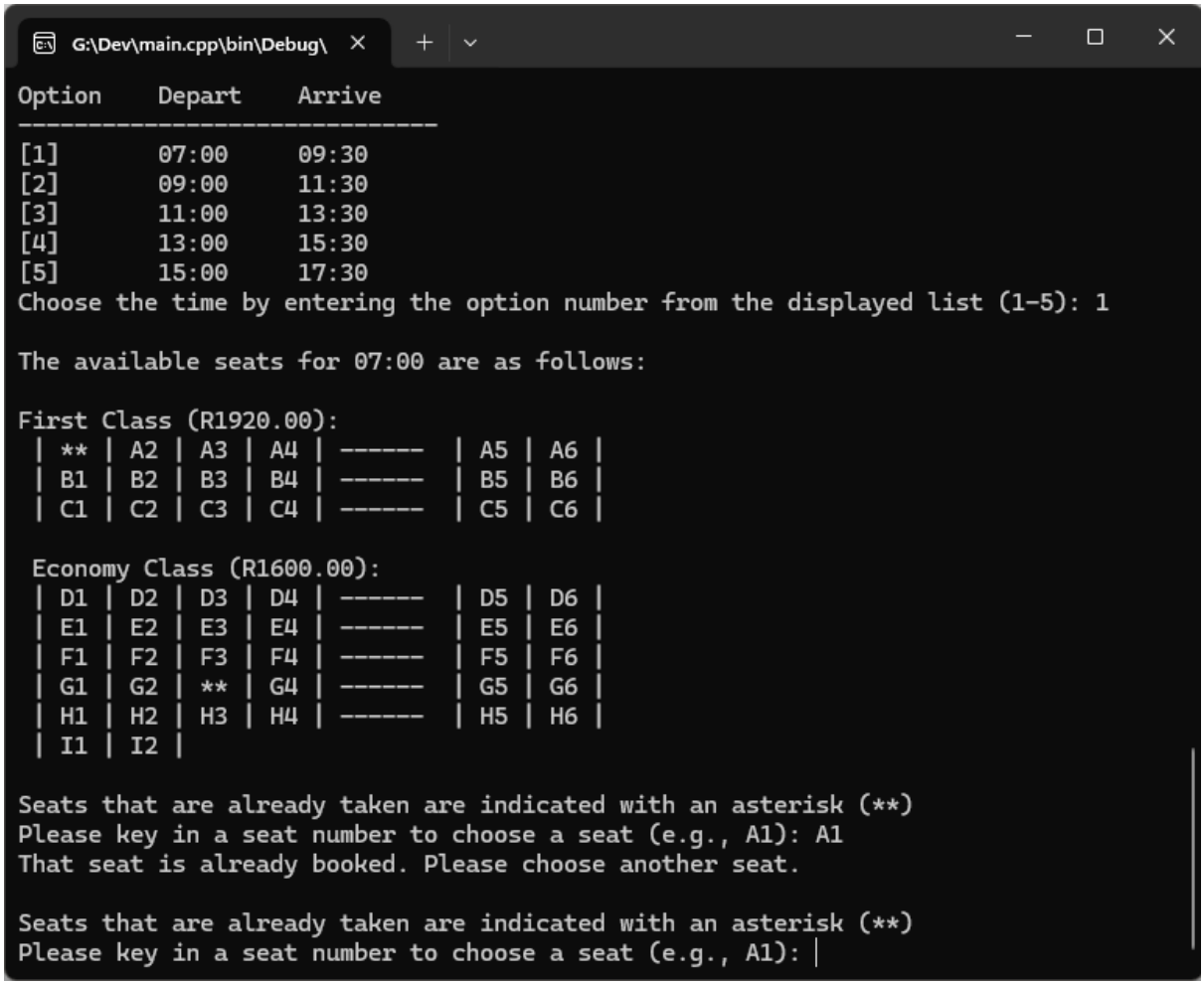
Output:

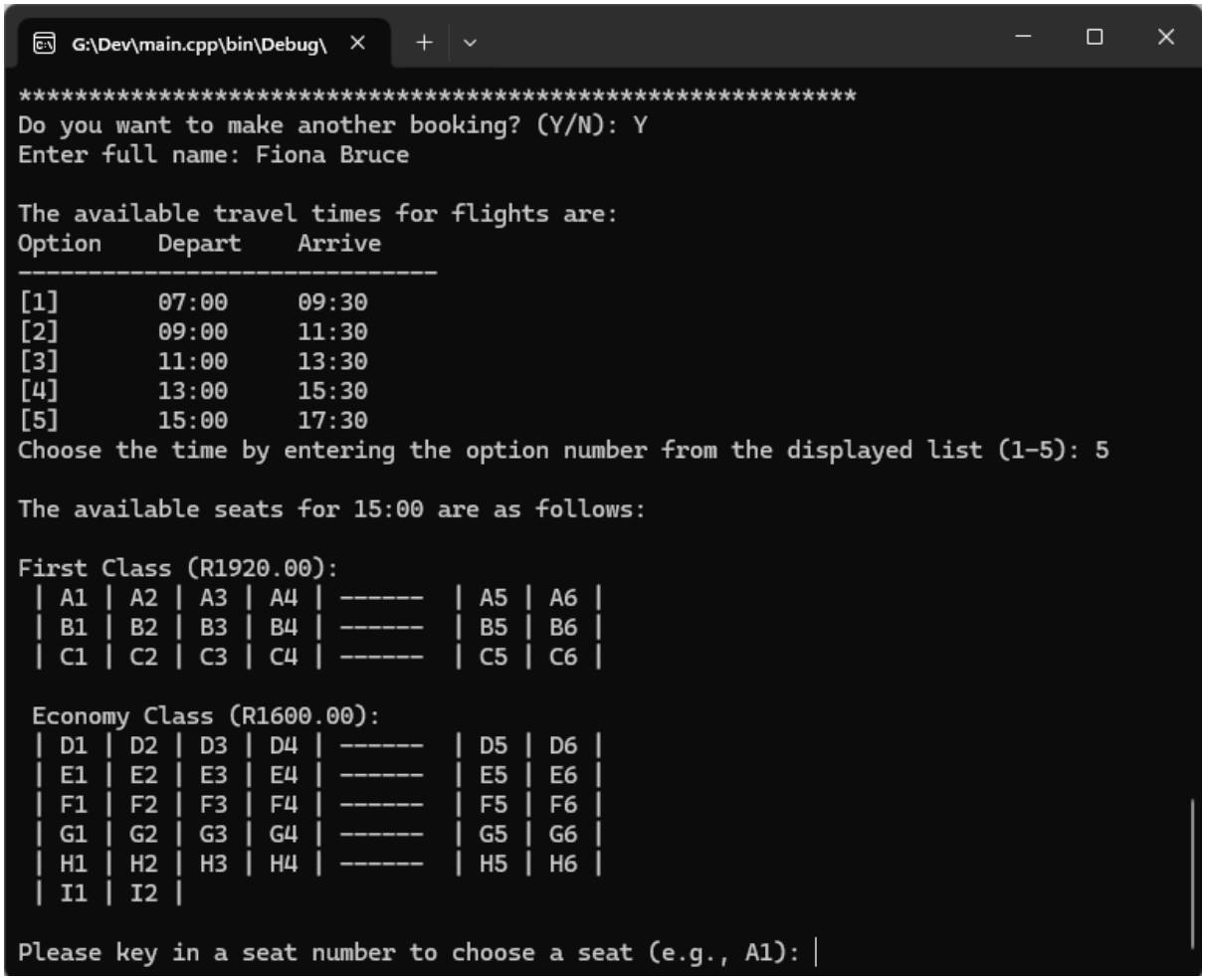

G:\Dev\main.cpp\bin\Debug\
****
*
*   Welcome to COS1511 Flight Booking System
*
****
Enter full name: Hannah George
The available travel times for flights are:
Option   Depart   Arrive
[1]   07:00   09:30
[2]   09:00   11:30
[3]   11:00   13:30
[4]   13:00   15:30
[5]   15:00   17:30
Choose the time by entering the option number from the displayed list (1-5): 9
Incorrect option! Please choose from 1-5.
Choose the time by entering the option number from the displayed list (1-5):

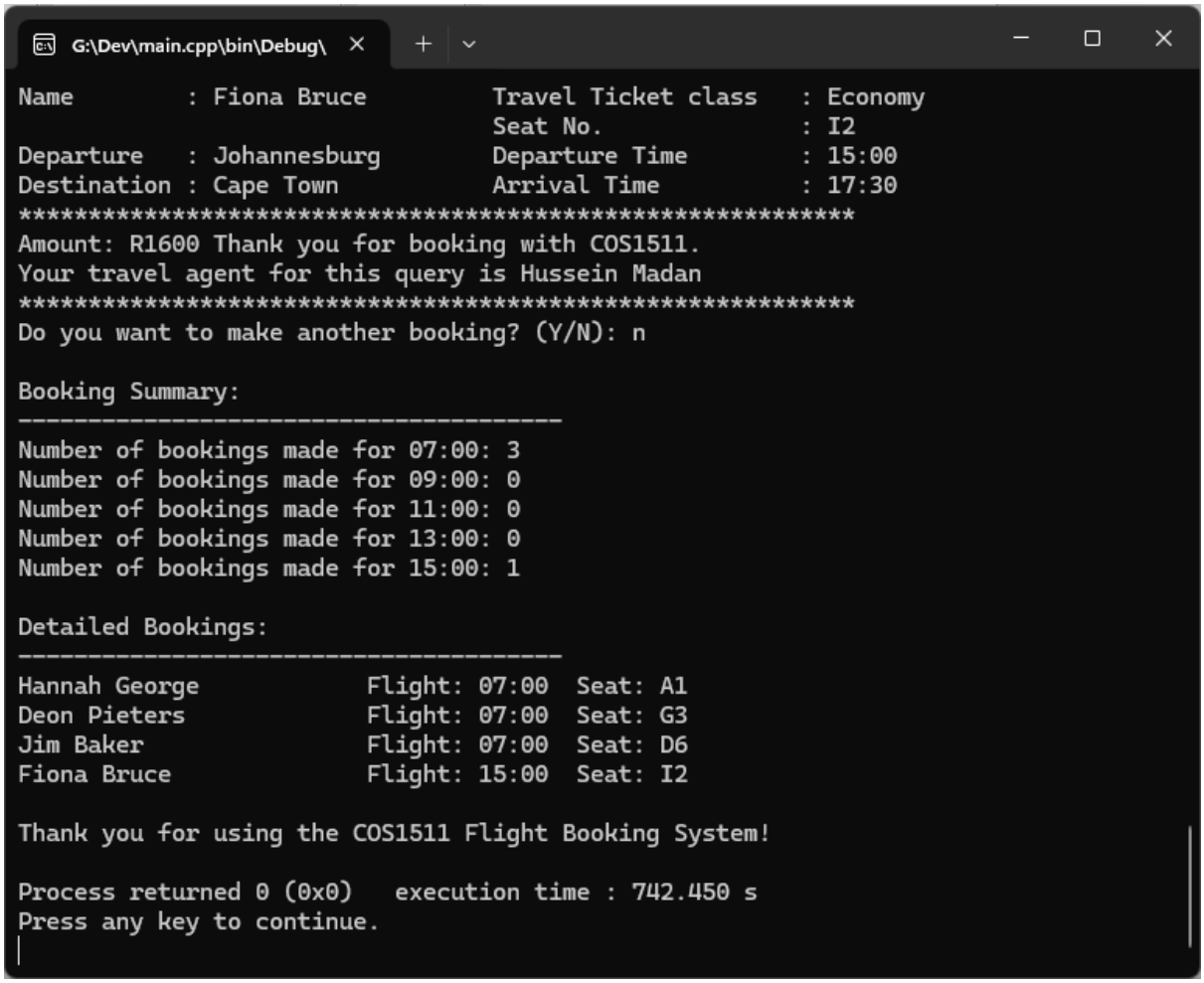

G:\Dev\main.cpp\bin\Debug\
Choose the time by entering the option number from the displayed list (1-5): 1
The available seats for 07:00 are as follows:
First Class (R1920.00):
A1	A2	A3	A4	------	A5	A6
B1	B2	B3	B4	------	B5	B6
C1	C2	C3	C4	------	C5	C6
Economy Class (R1600.00):
D1	D2	D3	D4	------	D5	D6
E1	E2	E3	E4	------	E5	E6
F1	F2	F3	F4	------	F5	F6
G1	G2	G3	G4	------	G5	G6
H1	H2	H3	H4	------	H5	H6
I1	I2
Please key in a seat number to choose a seat (e.g., A1): A1
Travel ticket for FLIGHT
Name : Hannah George
Travel Ticket class : First class
Seat No. : A1
Departure : Johannesburg
Departure Time : 07:00
Destination : Cape Town
Arrival Time : 09:30
Amount: R1920 Thank you for booking with COS1511.
Your travel agent for this query is Hussein Madan
Do you want to make another booking? (Y/N):


G:\Dev\main.cpp\bin\Debug\
Do you want to make another booking? (Y/N): Y
Enter full name: Deon Pieters
The available travel times for flights are:
Option Depart Arrive
[1] 07:00 09:30
[2] 09:00 11:30
[3] 11:00 13:30
[4] 13:00 15:30
[5] 15:00 17:30
Choose the time by entering the option number from the displayed list (1-5): 1
The available seats for 07:00 are as follows:
First Class (R1920.00):
**	A2	A3	A4	------	A5	A6
B1	B2	B3	B4	------	B5	B6
C1	C2	C3	C4	------	C5	C6
Economy Class (R1600.00):
D1	D2	D3	D4	------	D5	D6
E1	E2	E3	E4	------	E5	E6
F1	F2	F3	F4	------	F5	F6
G1	G2	G3	G4	------	G5	G6
H1	H2	H3	H4	------	H5	H6
I1	I2
Seats that are already taken are indicated with an asterisk (**)
Please key in a seat number to choose a seat (e.g., A1):


G:\Dev\main.cpp\bin\Debug\
Do you want to make another booking? (Y/N): Y
Enter full name: Jim Baker
The available travel times for flights are:
Option Depart Arrive
[1] 07:00 09:30
[2] 09:00 11:30
[3] 11:00 13:30
[4] 13:00 15:30
[5] 15:00 17:30
Choose the time by entering the option number from the displayed list (1-5): 1
The available seats for 07:00 are as follows:
First Class (R1920.00):
**	A2	A3	A4	------	A5	A6
B1	B2	B3	B4	------	B5	B6
C1	C2	C3	C4	------	C5	C6
Economy Class (R1600.00):
D1	D2	D3	D4	------	D5	D6
E1	E2	E3	E4	------	E5	E6
F1	F2	F3	F4	------	F5	F6
G1	G2	**	G4	------	G5	G6
H1	H2	H3	H4	------	H5	H6
I1	I2
Seats that are already taken are indicated with an asterisk (**)
Please key in a seat number to choose a seat (e.g., A1):


G:\Dev\main.cpp\bin\Debug\
Option Depart Arrive
[1] 07:00 09:30
[2] 09:00 11:30
[3] 11:00 13:30
[4] 13:00 15:30
[5] 15:00 17:30
Choose the time by entering the option number from the displayed list (1-5): 1
The available seats for 07:00 are as follows:
First Class (R1920.00):
**	A2	A3	A4	------	A5	A6
B1	B2	B3	B4	------	B5	B6
C1	C2	C3	C4	------	C5	C6
Economy Class (R1600.00):
D1	D2	D3	D4	------	D5	D6
E1	E2	E3	E4	------	E5	E6
F1	F2	F3	F4	------	F5	F6
G1	G2	**	G4	------	G5	G6
H1	H2	H3	H4	------	H5	H6
I1	I2
Seats that are already taken are indicated with an asterisk (**)
Please key in a seat number to choose a seat (e.g., A1): A1
That seat is already booked. Please choose another seat.
Seats that are already taken are indicated with an asterisk (**)
Please key in a seat number to choose a seat (e.g., A1):


G:\Dev\main.cpp\bin\Debug\
****
Do you want to make another booking? (Y/N): Y
Enter full name: Fiona Bruce
The available travel times for flights are:
Option Depart Arrive
[1] 07:00 09:30
[2] 09:00 11:30
[3] 11:00 13:30
[4] 13:00 15:30
[5] 15:00 17:30
Choose the time by entering the option number from the displayed list (1-5): 5
The available seats for 15:00 are as follows:
First Class (R1920.00):
A1	A2	A3	A4	------	A5	A6
B1	B2	B3	B4	------	B5	B6
C1	C2	C3	C4	------	C5	C6
Economy Class (R1600.00):
D1	D2	D3	D4	------	D5	D6
E1	E2	E3	E4	------	E5	E6
F1	F2	F3	F4	------	F5	F6
G1	G2	G3	G4	------	G5	G6
H1	H2	H3	H4	------	H5	H6
I1	I2
Please key in a seat number to choose a seat (e.g., A1):


G:\Dev\main.cpp\bin\Debug\
Name : Fiona Bruce Travel Ticket class : Economy
Seat No. : I2
Departure : Johannesburg Departure Time : 15:00
Destination : Cape Town Arrival Time : 17:30
****
Amount: R1600 Thank you for booking with COS1511.
Your travel agent for this query is Hussein Madan
****
Do you want to make another booking? (Y/N): n
Booking Summary:
Number of bookings made for 07:00: 3
Number of bookings made for 09:00: 0
Number of bookings made for 11:00: 0
Number of bookings made for 13:00: 0
Number of bookings made for 15:00: 1
Detailed Bookings:
Hannah George Flight: 07:00 Seat: A1
Deon Pieters Flight: 07:00 Seat: G3
Jim Baker Flight: 07:00 Seat: D6
Fiona Bruce Flight: 15:00 Seat: I2
Thank you for using the COS1511 Flight Booking System!
Process returned 0 (0x0) execution time : 742.450 s
Press any key to continue.