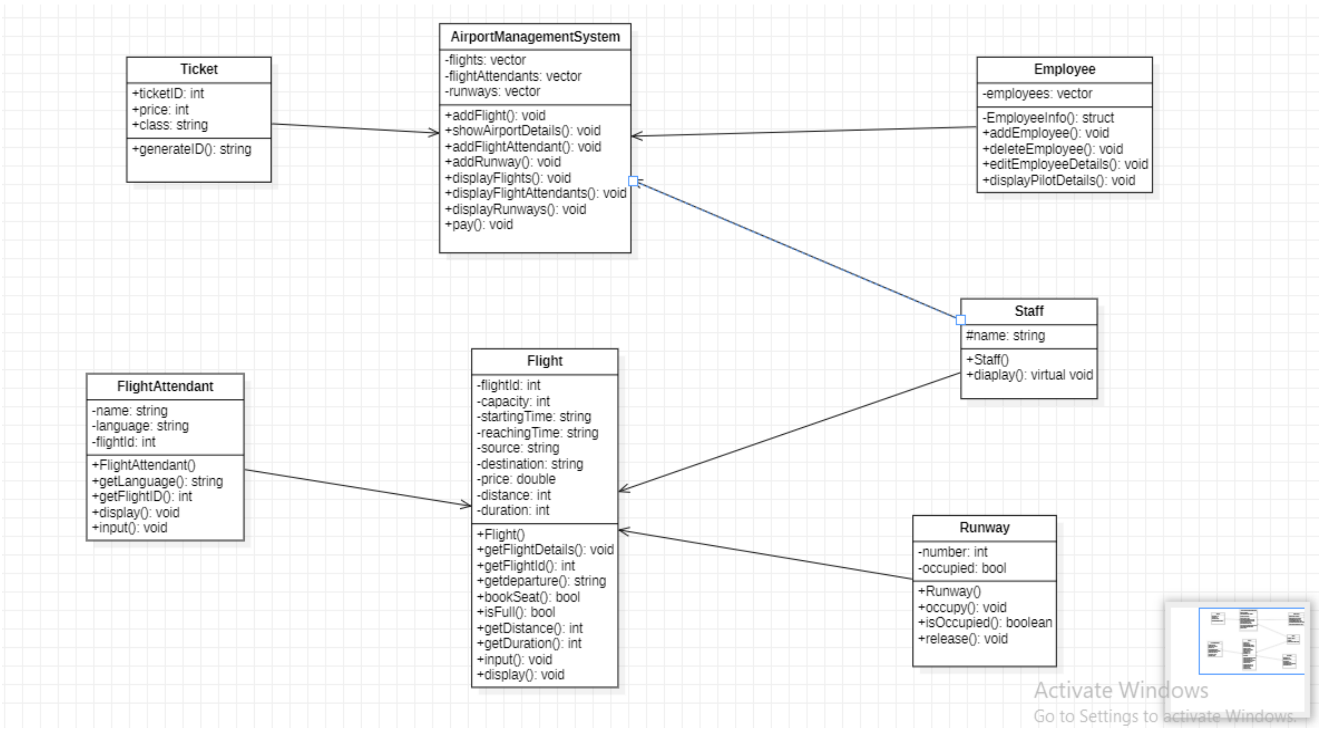
StarUml



Code

```
#include <bits/stdc++.h>
using namespace std;
class employee
{
private:
   struct EmployeeInfo
   {
      string name;
      string role;
      string licenseNumber;
   };
   vector<EmployeeInfo> employees;

public:
   void addEmployee()
```

```
{
    EmployeeInfo employee;
    cout << "Enter name: ";
    cin >> employee.name;
    cout << "Enter role: ";
    cin >> employee.role;
    cout << "Enter license number: ";
    cin >> employee.licenseNumber;
    employees.push_back(employee);
}

void deleteEmployee(const string &name)
{
    for (auto it = employees.begin(); it != employees.end(); ++it)
    {
        if (it->name == name)
        {
            employees.erase(it);
            cout << name << " deleted successfully." << endl;
            return;
        }
    }
    cout << name << " not found." << endl;
}

void editEmployeeDetails(const string &name)
{
    for (auto &employee : employees)
    {
        if (employee.name == name)
        {
            cout << "Enter new role: ";
            cin >> employee.role;
            cout << "Enter new license number: ";
            cin >> employee.licenseNumber;
            cout << name << "'s details updated successfully." << endl;
            return;
        }
    }
    cout << name << " not found." << endl;
}

void displayPilotDetails()
```

```
    {
        for (const auto &employee : employees)
        {
            cout << "Name: " << employee.name << ", Role: " << employee.role << ", License Number: " <<
employee.licenseNumber << endl;
        }
    }
};
class Runway
{
private:
    int number;
    bool occupied;
public:
    Runway(int n) : number(n), occupied(false) {}
    void occupy()
    {
        occupied = true;
    }
    void release()
    {
        occupied = false;
    }
    bool isOccupied() const
    {
        return occupied;
    }
};

class Staff
{
protected:
    string name;

public:
    Staff(string n) : name(n) {}
    virtual void display() const
    {
        cout << "Name: " << name << endl;
    }
};
class FlightAttendant
{
```

```
public:
   FlightAttendant() {}
   FlightAttendant(const string &name, const string &language, int flightID) : name(name),
language(language), flightID(flightID) {}

   string getName() const    {
      return name;
   }
   string getLanguage() const{
      return language;
   }
   int getFlightID() const
   {
      return flightID;
   }

   void display() const
   {
      cout << "Name: " << name << endl;
      cout << "Language: " << language << endl;
      cout << "Flight ID: " << flightID << endl;
   }

   void input()
   {
      cin.ignore(numeric_limits<streamsize>::max(), '\n');

      cout << "Enter name of the flight attendant: " << endl;
      getline(cin, name);

      cout << "Enter language spoken by the flight attendant: " << endl;
      getline(cin, language);

      cout << "Enter flight ID: " << endl;
      cin >> flightID;

      cin.ignore();
   }

private:
   string name;
   string language;
   int flightID;
```

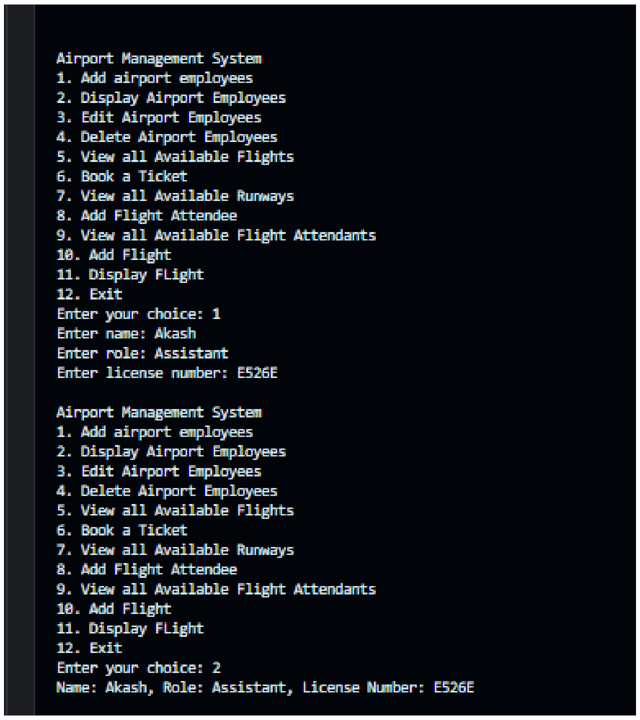
```
};

class Flight
{
private:
    int flightId;
    string departure;
    string destination;
    float price;
    int capacity;
    int bookedSeats;
    int flightNumber;
    int distance;
    int duration;

public:
    Flight(int id, string src, string dest, float p, int cap)
        : flightId(id), departure(src), destination(dest), price(p), capacity(cap), bookedSeats(0) {}

    int getFlightId() const { return flightId; }
    string getdeparture() const { return departure; }
    float getPrice() const { return price; }
    int getCapacity() const { return capacity; }
    int getBookedSeats() const { return bookedSeats; }

    bool bookSeat()
    {
        if (bookedSeats < capacity)
        {
            bookedSeats++;
            return true;
        }
        else
        {
            return false;
        }
    }

    bool isFull() const
    {
        return bookedSeats >= capacity;
    }
    Flight() {}
    Flight(int flightNumber, const string &departure, const string &destination, int distance, int duration)
        : flightNumber(flightNumber), departure(departure), destination(destination), distance(distance),
duration(duration) {}
    int getFlightNumber() const
    {
        return flightNumber;
    }

    string getDeparture() const
    {
        return departure;
    }

    string getDestination() const
    {
        return destination;
    }

    int getDistance() const
    {
        return distance;
    }

    int getDuration() const
    {
        return duration;
    }

    void input()
    {
        cout << "Enter flight number: ";
        cin >> flightNumber;

        cin.ignore();

        cout << "Enter departure city: ";
        getline(cin, departure);

        cout << "Enter destination city: ";
        getline(cin, destination);

        cout << "Enter distance (in miles): ";
        cin >> distance;

        cout << "Enter duration (in minutes): ";
        cin >> duration;

        cin.ignore();
    }

    void display() const
    {
        cout << "Flight Number: " << flightNumber << endl;
        cout << "Departure: " << departure << endl;
        cout << "Destination: " << destination << endl;
        cout << "Distance: " << distance << " miles" << endl;
        cout << "Duration: " << duration << " minutes" << endl;
    }
};

class AirportManagementSystem
{
private:
    vector<Flight> flights;
    vector<FlightAttendant> flightAttendants;
    vector<Runway> runways;

public:
    void addFlight()
    {
        Flight flight;
        flight.input();
        flights.push_back(flight);
    }

    void addFlightAttendant()
    {
        FlightAttendant attendant;
        attendant.input();
        flightAttendants.push_back(attendant);
    }

    void addRunway(const Runway &runway)
    {
        runways.push_back(runway);
```

```
}

void displayFlights() const
{
    cout << "Available Flights:\n";
    cout << "------------------------------------------\n";
    for (const Flight &flight : flights)
    {
        flight.display();
        cout << "------------------------------------------\n";
    }
}
void displayFlightAttendants() const
{
    cout << "Available Flight Attendants:\n";
    cout << "------------------------------------------\n";
    for (const FlightAttendant &attendant : flightAttendants)
    {
        attendant.display();
        cout << "------------------------------------------\n";
    }
}
void displayRunways() const
{
    cout << "Available Runways:\n";
    cout << "------------------------------------------\n";
    for (const Runway &runway : runways)
    {
        cout << "Runway Number: " << runway.isOccupied() << endl;
        cout << "------------------------------------------\n";
    }
}
void pay()
{

    int d, e, f, amo, y;
    cout << "Choose mode of payment :1.Net banking 2.Credit card 3.Debit card 4.Paytm" << endl;
    cin >> f;
    cout << "Enter amount to be paid" << endl;
    cin >> amo;
    cout << "Click 1. YES to proceed and 2.NO to go back\n";
    cin >> y;
```

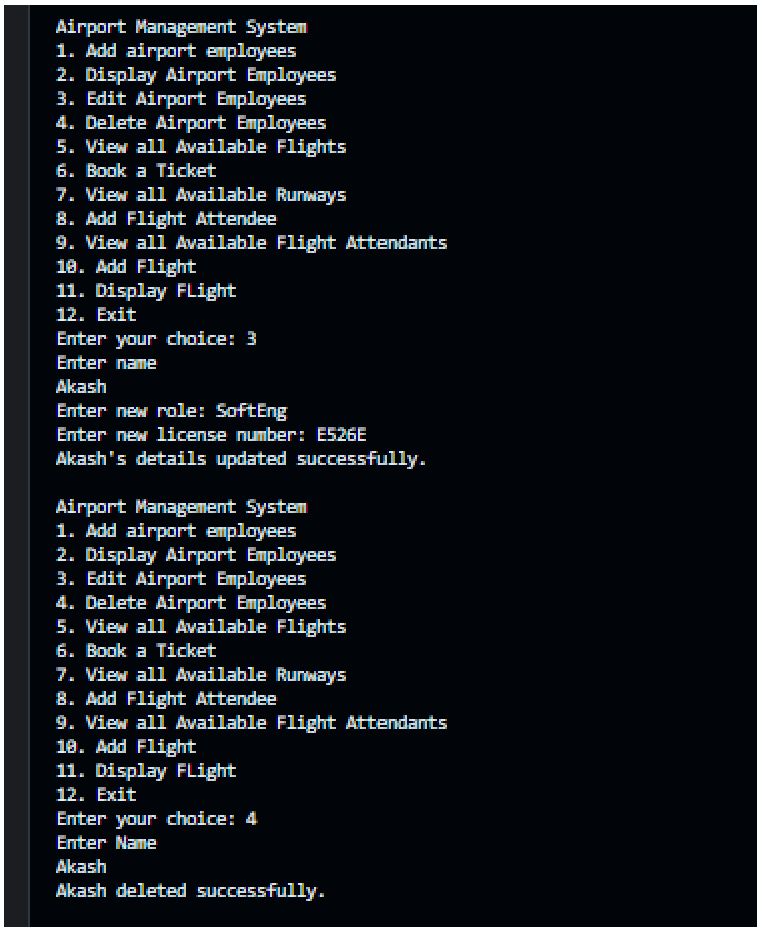
```
        cout << endl;
    }

    bool bookTicket(int flightId)
    {
        for (Flight &flight : flights)
        {
            if (flight.getFlightId() == flightId)
            {
                if (!flight.isFull())
                {
                    flight.bookSeat();
                    cout << "Your Ticket has been booked successfully!\n";
                    return true;
                }
                else
                {
                    cout << "Sorry, the flight is fully booked!\n";
                    return false;
                }
            }
        }
        cout << "Flight not found!\n";
        return false;
    }
};

int main()
{
    string s1, s2, s;
    AirportManagementSystem ams;

    ams.addRunway(Runway(1));
    ams.addRunway(Runway(2));
    employee e;
    int choice;
    do{
        cout << "\nAirport Management System\n";
        cout << "1. Add airport employees\n";
        cout << "2. Display Airport Employees\n";
        cout << "3. Edit Airport Employees\n";
        cout << "4. Delete Airport Employees\n";
        cout << "5. View all Available Flights\n";
cout << "6. Book a Ticket\n";
cout << "7. View all Available Runways\n";
cout << "8. Add Flight Attendee\n";
cout << "9. View all Available Flight Attendants\n";
cout << "10. Add Flight \n";
cout << "11. Display FLight\n";
cout << "12. Exit\n";
cout << "Enter your choice: ";
cin >> choice;

switch (choice)
{
case 1:
   e.addEmployee();
   break;
case 2:
   e.displayPilotDetails();
   break;
case 3:
   cout << "Enter name" << endl;
   cin >> s;
   e.editEmployeeDetails(s);
   break;
case 4:
   cout << "Enter Name" << endl;
   cin >> s;
   e.deleteEmployee(s);
   break;
case 5:
   ams.displayFlights();
   break;
case 6:
   int flightId;
   int time;
   cout << "Enter to and from location" << endl;
   cin >> s1 >> s2;
   cout << "Choose your preferred timing" << endl;
   cout << "1.9:00 hrs  2.13:20 hrs  3.22:00 hrs" << endl;
   cin >> time;
   cout << "Enter the ID of the flight you want to book: ";
   cin >> flightId;
   ams.pay();
   ams.bookTicket(flightId);
```

```
            break;

        case 7:
            ams.displayRunways();
            break;
        case 8:
            ams.addFlightAttendant();
            break;
        case 9:
            ams.displayFlightAttendants();
            break;
        case 10:
            ams.addFlight();
            break;
        case 11:
            ams.displayFlights();
            break;

        case 12:
            cout << "Thank you for using the Airport Management System. Have a nice day!\n";
            break;
        default:
            cout << "Invalid choice. Please try again.\n";
        }
    } while (choice != 5);

    return 0;
}
```

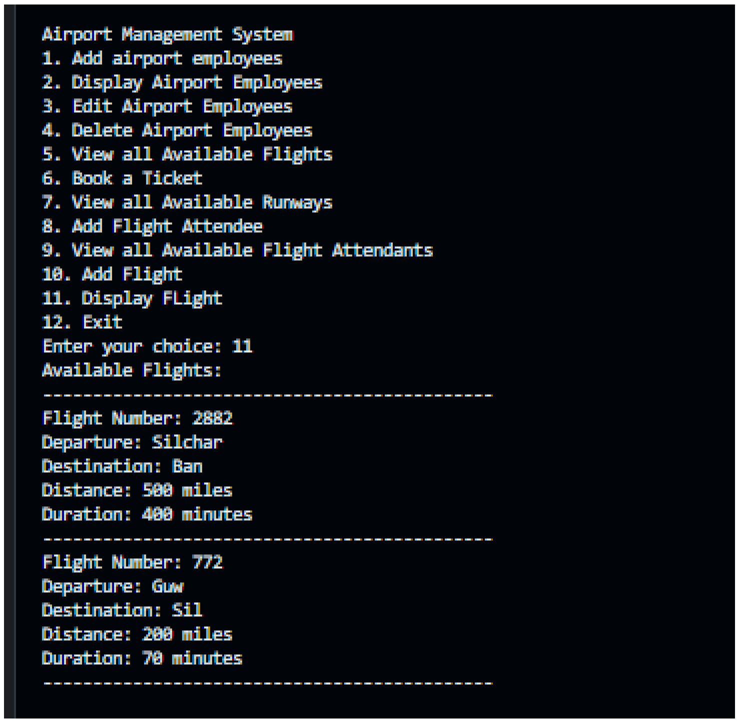
OUTPUT



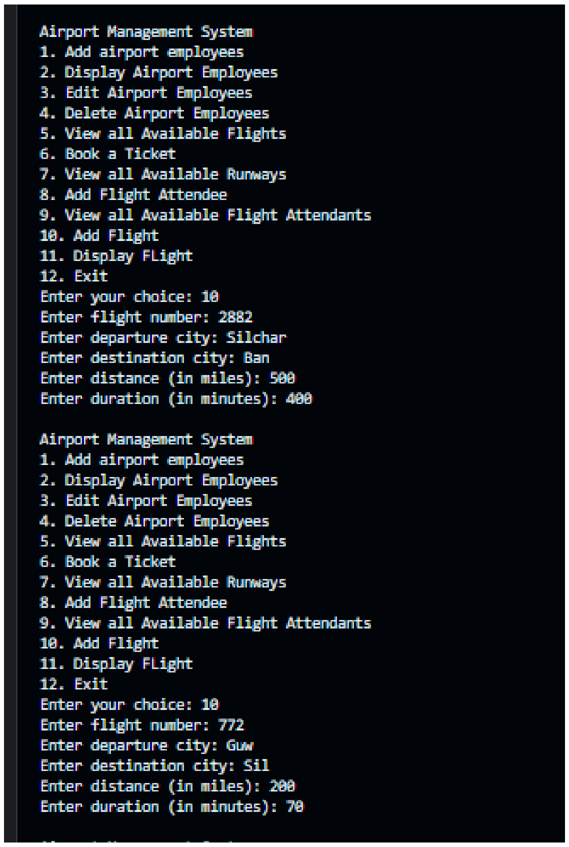
```
Airport Management System
1. Add airport employees
2. Display Airport Employees
3. Edit Airport Employees
4. Delete Airport Employees
5. View all Available Flights
6. Book a Ticket
7. View all Available Runways
8. Add Flight Attendee
9. View all Available Flight Attendants
10. Add Flight
11. Display FLight
12. Exit
Enter your choice: 1
Enter name: Akash
Enter role: Assistant
Enter license number: E526E

Airport Management System
1. Add airport employees
2. Display Airport Employees
3. Edit Airport Employees
4. Delete Airport Employees
5. View all Available Flights
6. Book a Ticket
7. View all Available Runways
8. Add Flight Attendee
9. View all Available Flight Attendants
10. Add Flight
11. Display FLight
12. Exit
Enter your choice: 2
Name: Akash, Role: Assistant, License Number: E526E
```



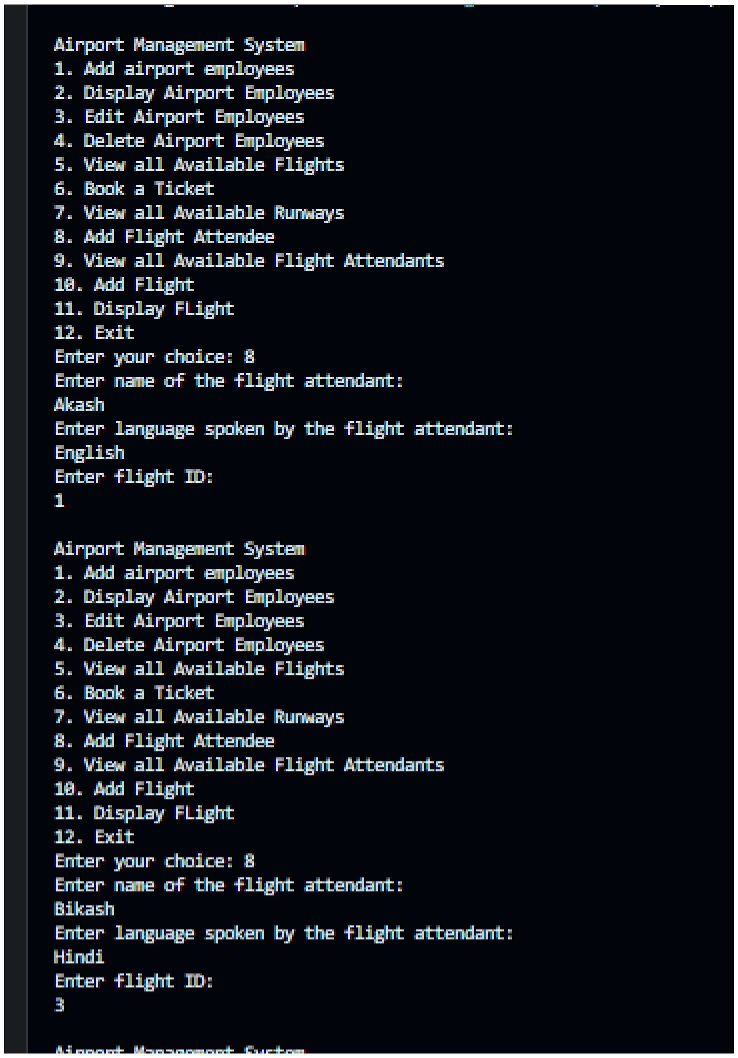
```
Airport Management System
1. Add airport employees
2. Display Airport Employees
3. Edit Airport Employees
4. Delete Airport Employees
5. View all Available Flights
6. Book a Ticket
7. View all Available Runways
8. Add Flight Attendee
9. View all Available Flight Attendants
10. Add Flight
11. Display FLight
12. Exit
Enter your choice: 3
Enter name
Akash
Enter new role: SoftEng
Enter new license number: E526E
Akash's details updated successfully.

Airport Management System
1. Add airport employees
2. Display Airport Employees
3. Edit Airport Employees
4. Delete Airport Employees
5. View all Available Flights
6. Book a Ticket
7. View all Available Runways
8. Add Flight Attendee
9. View all Available Flight Attendants
10. Add Flight
11. Display FLight
12. Exit
Enter your choice: 4
Enter Name
Akash
Akash deleted successfully.
```



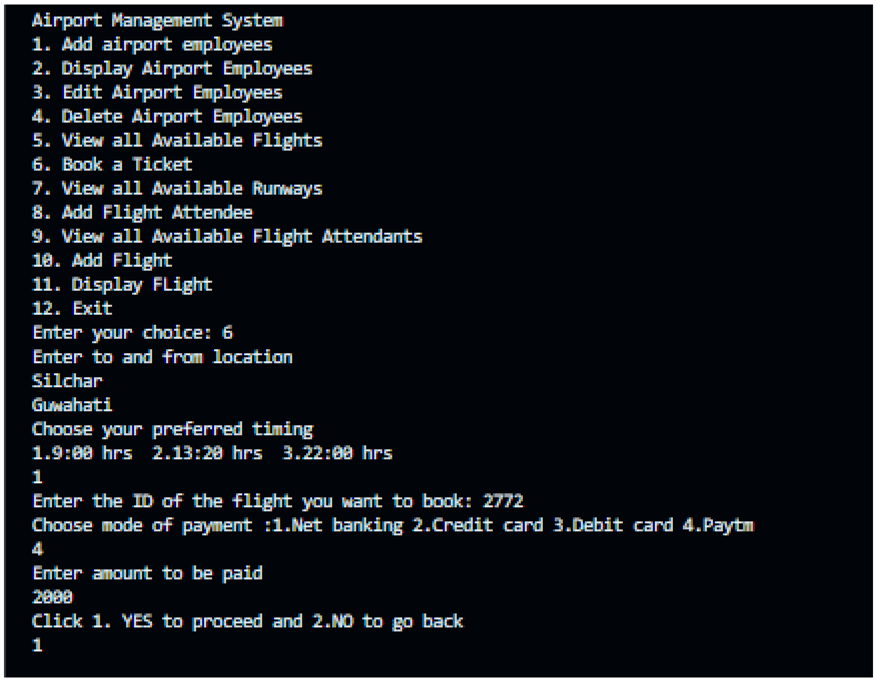
```
Airport Management System
1. Add airport employees
2. Display Airport Employees
3. Edit Airport Employees
4. Delete Airport Employees
5. View all Available Flights
6. Book a Ticket
7. View all Available Runways
8. Add Flight Attendee
9. View all Available Flight Attendants
10. Add Flight
11. Display FLight
12. Exit
Enter your choice: 11
Available Flights:
---------------------------------------------
Flight Number: 2882
Departure: Silchar
Destination: Ban
Distance: 500 miles
Duration: 400 minutes
---------------------------------------------
Flight Number: 772
Departure: Guw
Destination: Sil
Distance: 200 miles
Duration: 70 minutes
---------------------------------------------
```



```
Airport Management System
1. Add airport employees
2. Display Airport Employees
3. Edit Airport Employees
4. Delete Airport Employees
5. View all Available Flights
6. Book a Ticket
7. View all Available Runways
8. Add Flight Attendee
9. View all Available Flight Attendants
10. Add Flight
11. Display FLight
12. Exit
Enter your choice: 10
Enter flight number: 2882
Enter departure city: Silchar
Enter destination city: Ban
Enter distance (in miles): 500
Enter duration (in minutes): 400

Airport Management System
1. Add airport employees
2. Display Airport Employees
3. Edit Airport Employees
4. Delete Airport Employees
5. View all Available Flights
6. Book a Ticket
7. View all Available Runways
8. Add Flight Attendee
9. View all Available Flight Attendants
10. Add Flight
11. Display FLight
12. Exit
Enter your choice: 10
Enter flight number: 772
Enter departure city: Guw
Enter destination city: Sil
Enter distance (in miles): 200
Enter duration (in minutes): 70
```



```
Airport Management System
1. Add airport employees
2. Display Airport Employees
3. Edit Airport Employees
4. Delete Airport Employees
5. View all Available Flights
6. Book a Ticket
7. View all Available Runways
8. Add Flight Attendee
9. View all Available Flight Attendants
10. Add Flight
11. Display FLight
12. Exit
Enter your choice: 8
Enter name of the flight attendant:
Akash
Enter language spoken by the flight attendant:
English
Enter flight ID:
1

Airport Management System
1. Add airport employees
2. Display Airport Employees
3. Edit Airport Employees
4. Delete Airport Employees
5. View all Available Flights
6. Book a Ticket
7. View all Available Runways
8. Add Flight Attendee
9. View all Available Flight Attendants
10. Add Flight
11. Display FLight
12. Exit
Enter your choice: 8
Enter name of the flight attendant:
Bikash
Enter language spoken by the flight attendant:
Hindi
Enter flight ID:
3

Airport Management System
```



```
Airport Management System
1. Add airport employees
2. Display Airport Employees
3. Edit Airport Employees
4. Delete Airport Employees
5. View all Available Flights
6. Book a Ticket
7. View all Available Runways
8. Add Flight Attendee
9. View all Available Flight Attendants
10. Add Flight
11. Display FLight
12. Exit
Enter your choice: 6
Enter to and from location
Silchar
Guwahati
Choose your preferred timing
1.9:00 hrs  2.13:20 hrs  3.22:00 hrs
1
Enter the ID of the flight you want to book: 2772
Choose mode of payment :1.Net banking 2.Credit card 3.Debit card 4.Paytm
4
Enter amount to be paid
2000
Click 1. YES to proceed and 2.NO to go back
1
```