**HARDWARE AND SOFTWARE REQUIREMENTS**

I.OPERATING SYSTEM : WINDOWS 7 AND ABOVE

II. PROCESSOR : PENTIUM(ANY) OR AMD

ATHALON (3800+- 4200+ DUALCORE)

III. MOTHERBOARD : 1.845 OR 915,995 FOR PENTIUM 0R MSI

K9MM-V VIAK8M800+8237R PLUS CHIPSET FOR AMD ATHALON

IV. RAM : 512MB+

V. Hard disk : SATA 40 GB OR ABOVE

VI. CD/DVD r/w multi drive combo: (If back up required)

VII. FLOPPY DRIVE 1.44 MB : (If Backup required)

VIII. MONITOR 14.1 or 15 -17 inch

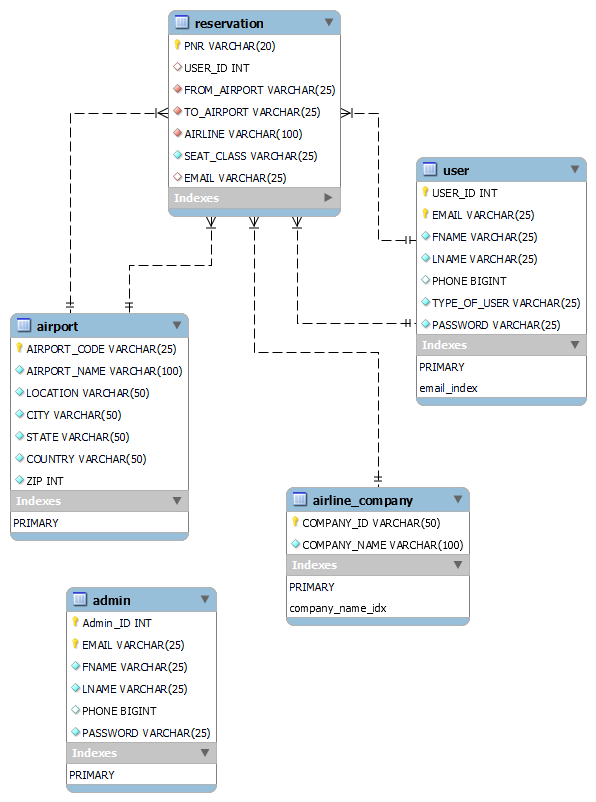
IX. Key board and mouse

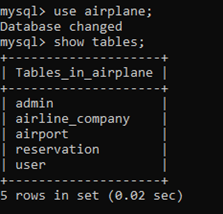
X. Printer : (if print is required – [Hard copy])

**SOFTWARE REQUIREMENTS:**

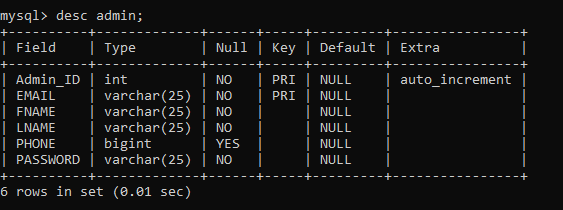
1. Windows OS
2. Python

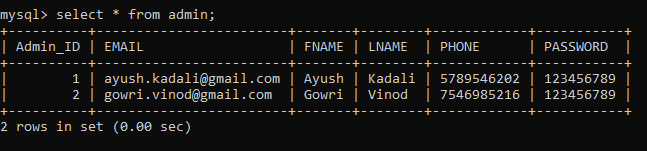
**DATABASE(TABLES)**

****

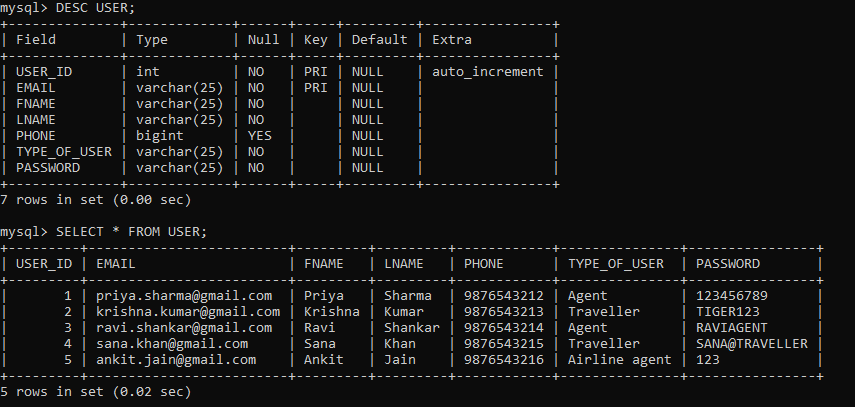
****

**ADMIN TABLE**

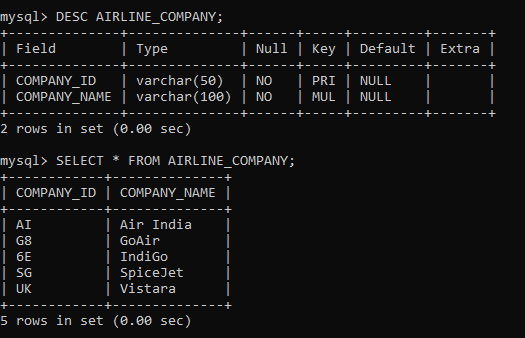
****

****

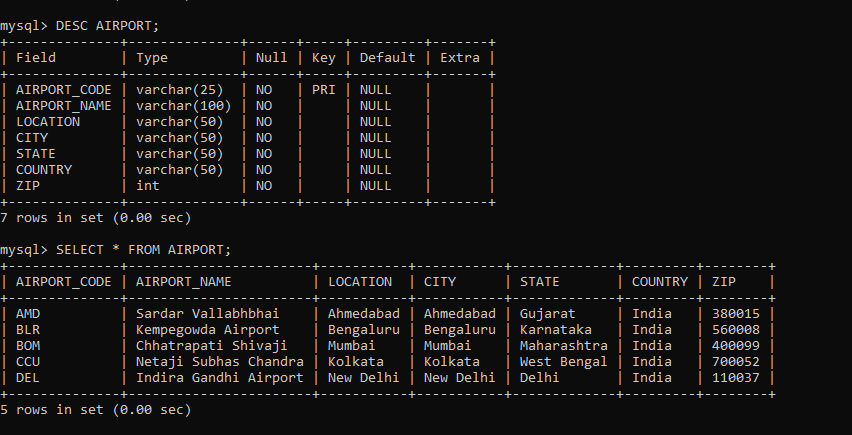
**USER**

****

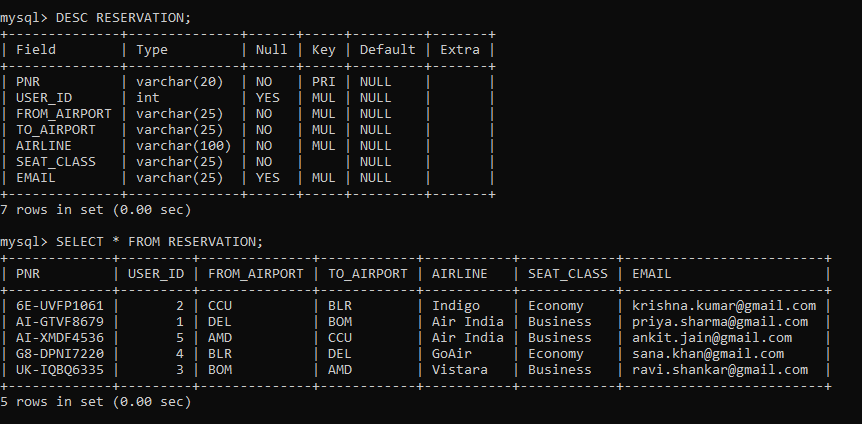
**AIRLINE COMPANY**

****

**AIRPORT**

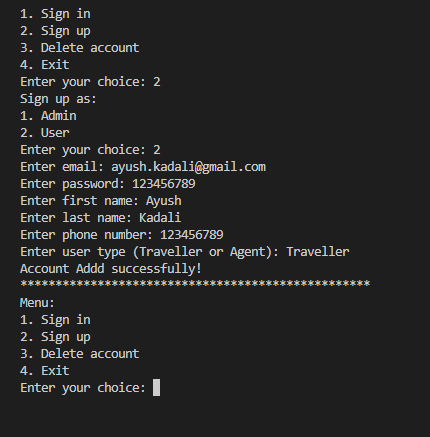
****

**RESERVATION**

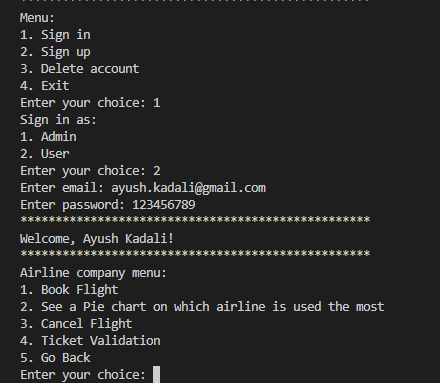
****

**INPUT & OUTPUT SCREENS**

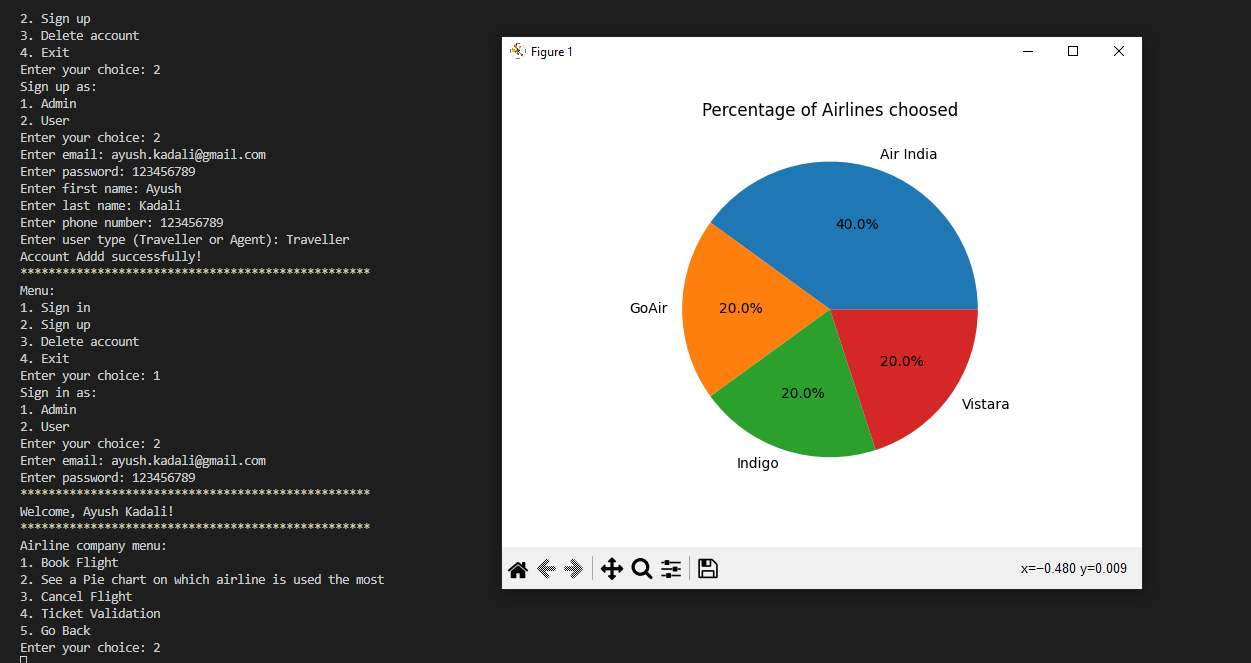
1) Signing up as a user



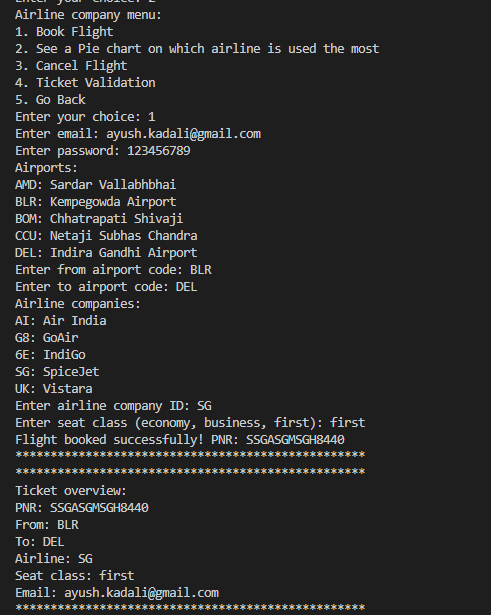
2) Signing in as a user



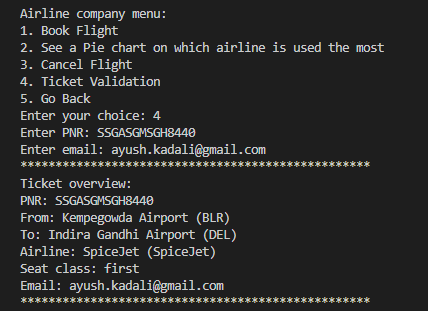
1. Seeing a pie chart of which airline is used the most



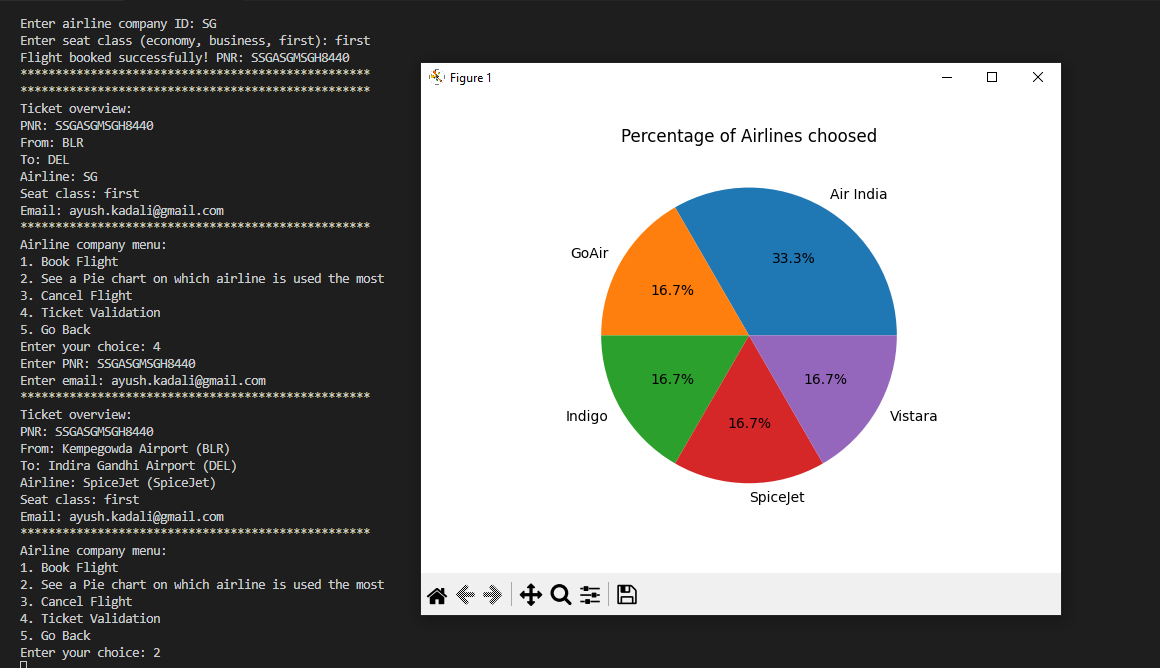
1. Booking a flight From BLR to DEL in first class with SPICE JET



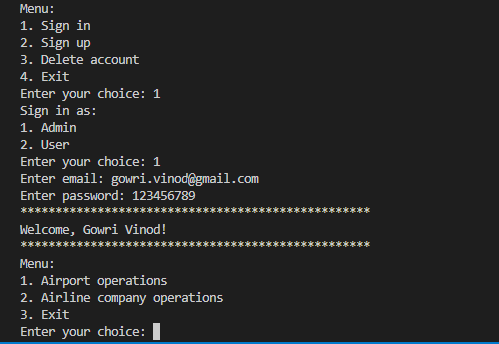
1. Ticket Validation



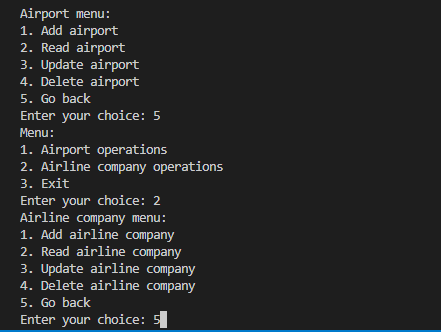
1. See the Pie chart again



1. Signing in as an admin



1. Available ADMIN options



**Project Code**

**Python Code:**

import mysql.connector

import matplotlib.pyplot as plt

import random

import string

import sys

# Connect to the database

cnx = mysql.connector.connect(user='root',

                              password='ayush@1254',

                              host='localhost',

                              database='AIRPLANE')

# Add a cursor object

cursor = cnx.cursor()

def main():

    options = {

        1: sign\_in,

        2: sign\_up,

        3: delete\_account,

        4: exit

    }

    while True:

        print("Menu:")

        print("1. Sign in")

        print("2. Sign up")

        print("3. Delete account")

        print("4. Exit")

        choice = int(input("Enter your choice: "))

        options[choice]()

def sign\_in():

    options = {

        1: sign\_in\_admin,

        2: sign\_in\_user

    }

    print("Sign in as:")

    print("1. Admin")

    print("2. User")

    choice = int(input("Enter your choice: "))

    options[choice]()

def sign\_in\_admin():

    # Get the email and password to sign in with

    global email

    global password

    email = input("Enter email: ")

    password = input("Enter password: ")

    # Execute a SELECT query to get the admin with the given email and password

    query = f"SELECT \* FROM ADMIN WHERE EMAIL='{email}' AND PASSWORD='{password}'"

    cursor.execute(query)

    admin = cursor.fetchone()

    if admin is None:

        print("Invalid email or password. Please try again.")

        print("\*"\*50)

    else:

        print("\*"\*50)

        print(f"Welcome, {admin[2]} {admin[3]}!")

        print("\*"\*50)

        admin\_menu()

def sign\_in\_user():

    # Get the email and password to sign in with

    global email

    global password

    email = input("Enter email: ")

    password = input("Enter password: ")

    # Execute a SELECT query to get the user with the given email and password

    query = f"SELECT \* FROM USER WHERE EMAIL='{email}' AND PASSWORD='{password}'"

    cursor.execute(query)

    user = cursor.fetchone()

    if user is None:

        print("Invalid email or password. Please try again.")

    else:

        print("\*"\*50)

        print(f"Welcome, {user[2]} {user[3]}!")

        print("\*"\*50)

        user\_menu()

def sign\_up():

    options = {

        1: sign\_up\_admin,

        2: sign\_up\_user

    }

    print("Sign up as:")

    print("1. Admin")

    print("2. User")

    choice = int(input("Enter your choice: "))

    options[choice]()

def sign\_up\_admin():

    # Get the email and password to sign up with

    email = input("Enter email: ")

    password = input("Enter password: ")

    first\_name = input("Enter first name: ")

    last\_name = input("Enter last name: ")

    phone = input("Enter phone number: ")

    # Execute an INSERT query to add a new admin to the ADMIN table

    query = f"INSERT INTO ADMIN (EMAIL, PASSWORD, FNAME, LNAME, PHONE) VALUES ('{email}', '{password}', '{first\_name}', '{last\_name}', '{phone}')"

    cursor.execute(query)

    cnx.commit()

    print("Account Addd successfully!")

    print("\*"\*50)

def sign\_up\_user():

    # Get the email and password to sign up with

    email = input("Enter email: ")

    password = input("Enter password: ")

    first\_name = input("Enter first name: ")

    last\_name = input("Enter last name: ")

    phone = input("Enter phone number: ")

    user\_type = input("Enter user type (Traveller or Agent): ")

    # Execute an INSERT query to add a new user to the USER table

    query = f"INSERT INTO USER (EMAIL, PASSWORD, FNAME, LNAME, PHONE, TYPE\_OF\_USER) VALUES ('{email}', '{password}', '{first\_name}', '{last\_name}', '{phone}', '{user\_type}')"

    cursor.execute(query)

    cnx.commit()

    print("Account Addd successfully!")

    print("\*"\*50)

def delete\_account():

    options = {

        1: delete\_admin\_account,

        2: delete\_user\_account

    }

    print("Delete account for:")

    print("1. Admin")

    print("2. User")

    choice = int(input("Enter your choice: "))

    options[choice]()

def delete\_admin\_account():

    # Get the email and password to delete the account for

    email = input("Enter email: ")

    password = input("Enter password: ")

    # Execute a DELETE query to delete the admin with the given email and password

    query = f"DELETE FROM ADMIN WHERE EMAIL='{email}' AND PASSWORD='{password}'"

    cursor.execute(query)

    cnx.commit()

    print("Account deleted successfully!")

    print("\*"\*50)

def delete\_user\_account():

    # Get the email and password to delete the account for

    email = input("Enter email: ")

    password = input("Enter password: ")

    # Execute a DELETE query to delete the user with the given email and password

    query = f"DELETE FROM USER WHERE EMAIL='{email}' AND PASSWORD='{password}'"

    cursor.execute(query)

    cnx.commit()

    print("Account deleted successfully!")

    print("\*"\*50)

def exit():

    # Close the cursor and connection

    cursor.close()

    cnx.close()

    sys.exit()

def admin\_menu():

    options = {

        1: airport\_menu,

        2: airline\_company\_menu,

        3: exit

    }

    while True:

        print("Menu:")

        print("1. Airport operations")

        print("2. Airline company operations")

        print("3. Exit")

        choice = int(input("Enter your choice: "))

        options[choice]()

def airport\_menu():

    options = {

        1: Add\_airport,

        2: read\_airport,

        3: update\_airport,

        4: delete\_airport

    }

    while True:

        print("Airport menu:")

        print("1. Add airport")

        print("2. Read airport")

        print("3. Update airport")

        print("4. Delete airport")

        print("5. Go back")

        choice = int(input("Enter your choice: "))

        if choice == 5:

            break

        options[choice]()

def Add\_airport():

    # Get the airport code, name, location, city, state, country, and zip code to Add a new airport

    airport\_code = input("Enter airport code: ")

    airport\_name = input("Enter airport name: ")

    location = input("Enter location: ")

    city = input("Enter city: ")

    state = input("Enter state: ")

    country = input("Enter country: ")

    zip\_code = input("Enter zip code: ")

    # Execute an INSERT query to add a new airport to the AIRPORT table

    query = f"INSERT INTO AIRPORT (AIRPORT\_CODE, AIRPORT\_NAME, LOCATION, CITY, STATE, COUNTRY, ZIP) VALUES ('{airport\_code}', '{airport\_name}', '{location}', '{city}', '{state}', '{country}', '{zip\_code}')"

    cursor.execute(query)

    cnx.commit()

    print("Airport Addd successfully!")

    print("\*"\*50)

def read\_airport():

    # Execute a SELECT query to get the airport with the given airport code

    query = f"SELECT \* FROM AIRPORT"

    cursor.execute(query)

    airport = cursor.fetchall()

    for row in airport:

        print(f"Airport code: {row[0]}")

        print(f"Airport name: {row[1]}")

        print(f"Location: {row[2]}")

        print(f"City: {row[3]}")

        print(f"State: {row[4]}")

        print(f"Country: {row[5]}")

        print(f"Zip code: {row[6]}")

        print("\*"\*50)

def update\_airport():

    # Get the airport code to update

    airport\_code = input("Enter airport code: ")

    # Execute a SELECT query to get the airport with the given airport code

    query = f"SELECT \* FROM AIRPORT WHERE AIRPORT\_CODE='{airport\_code}'"

    cursor.execute(query)

    airport = cursor.fetchone()

    if airport is None:

        print("Airport not found.")

        print("\*"\*50)

    else:

        # Get the updated airport code, name, location, city, state, country, and zip code

        airport\_name = input("Enter updated airport name: ")

        location = input("Enter updated location: ")

        city = input("Enter updated city: ")

        state = input("Enter updated state: ")

        country = input("Enter updated country: ")

        zip\_code = input("Enter updated zip code: ")

        # Execute an UPDATE query to update the airport with the given airport code

        query = f"UPDATE AIRPORT SET AIRPORT\_NAME='{airport\_name}', LOCATION='{location}', CITY='{city}', STATE='{state}', COUNTRY='{country}', ZIP='{zip\_code}' WHERE AIRPORT\_CODE='{airport\_code}'"

        cursor.execute(query)

        cnx.commit()

        print("Airport updated successfully!")

        print("\*"\*50)

def delete\_airport():

    # Get the airport code to delete

    airport\_code = input("Enter airport code: ")

    # Execute a DELETE query to delete the airport with the given airport code

    query = f"DELETE FROM AIRPORT WHERE AIRPORT\_CODE='{airport\_code}'"

    cursor.execute(query)

    cnx.commit()

    print("Airport deleted successfully!")

    print("\*"\*50)

def airline\_company\_menu():

    options = {

        1: Add\_airline\_company,

        2: read\_airline\_company,

        3: update\_airline\_company,

        4: delete\_airline\_company

    }

    while True:

        print("Airline company menu:")

        print("1. Add airline company")

        print("2. Read airline company")

        print("3. Update airline company")

        print("4. Delete airline company")

        print("5. Go back")

        choice = int(input("Enter your choice: "))

        if choice == 5:

            break

        options[choice]()

def Add\_airline\_company():

    # Get the company ID and name to Add a new airline company

    company\_id = input("Enter company ID: ")

    company\_name = input("Enter company name: ")

    # Execute an INSERT query to add a new airline company to the AIRLINE\_COMPANY table

    query = f"INSERT INTO AIRLINE\_COMPANY (COMPANY\_ID, COMPANY\_NAME) VALUES ('{company\_id}', '{company\_name}')"

    cursor.execute(query)

    cnx.commit()

    print("Airline company Addd successfully!")

    print("\*"\*50)

def read\_airline\_company():

    # Execute a SELECT query to get the airline company with the given company ID

    query = f"SELECT \* FROM AIRLINE\_COMPANY"

    cursor.execute(query)

    airline\_company = cursor.fetchall()

    for row in airline\_company:

        print(f"Company ID: {row[0]}")

        print(f"Company name: {row[1]}")

        print("\*"\*50)

def update\_airline\_company():

    # Get the company ID to update

    company\_id = input("Enter company ID: ")

    # Execute a SELECT query to get the airline company with the given company ID

    query = f"SELECT \* FROM AIRLINE\_COMPANY WHERE COMPANY\_ID='{company\_id}'"

    cursor.execute(query)

    airline\_company = cursor.fetchone()

    if airline\_company is None:

        print("Airline company not found.")

        print("\*"\*50)

    else:

        # Get the updated company name

        company\_name = input("Enter updated company name: ")

        # Execute an UPDATE query to update the airline company with the given company ID

        query = f"UPDATE AIRLINE\_COMPANY SET COMPANY\_NAME='{company\_name}' WHERE COMPANY\_ID='{company\_id}'"

        cursor.execute(query)

        cnx.commit()

        print("Airline company updated successfully!")

        print("\*"\*50)

def delete\_airline\_company():

    # Get the company ID to delete

    company\_id = input("Enter company ID: ")

    # Execute a DELETE query to delete the airline company with the given company ID

    query = f"DELETE FROM AIRLINE\_COMPANY WHERE COMPANY\_ID='{company\_id}'"

    cursor.execute(query)

    cnx.commit()

    print("Airline company deleted successfully!")

    print("\*"\*50)

def user\_menu():

    options = {

        1: RESERVATION\_menu,

        2: pie\_chart,

        3: cancel\_flight,

        4: ticket\_validation,

    }

    while True:

        print("Airline company menu:")

        print("1. Book Flight")

        print("2. See a Pie chart on which airline is used the most")

        print("3. Cancel Flight")

        print("4. Ticket Validation")

        print("5. Go Back")

        choice = int(input("Enter your choice: "))

        if choice == 5:

            break

        options[choice]()

def RESERVATION\_menu():

     # Get the email and password of the user

    email = input("Enter email: ")

    password = input("Enter password: ")

    # Execute a SELECT query to get the user with the given email and password

    query = f"SELECT \* FROM USER WHERE EMAIL='{email}' AND PASSWORD='{password}'"

    cursor.execute(query)

    user = cursor.fetchone()

    if user is None:

        # Get the email and password of the user

        print("Incorrect Password! Try again")

    else:

        # Get the list of airports

        query = "SELECT AIRPORT\_CODE, AIRPORT\_NAME FROM AIRPORT"

        cursor.execute(query)

        airports = cursor.fetchall()

        print("Airports:")

        for airport in airports:

            print(f"{airport[0]}: {airport[1]}")

        # Get the from airport code and to airport code

        from\_airport = input("Enter from airport code: ")

        to\_airport = input("Enter to airport code: ")

        # Get the list of airline companies

        query = "SELECT COMPANY\_ID, COMPANY\_NAME FROM AIRLINE\_COMPANY"

        cursor.execute(query)

        companies = cursor.fetchall()

        print("Airline companies:")

        for company in companies:

            print(f"{company[0]}: {company[1]}")

        # Get the airline company ID

        company\_id = input("Enter airline company ID: ")

        query = f"SELECT COMPANY\_NAME FROM AIRLINE\_COMPANY WHERE COMPANY\_ID = '{company\_id}'"

        cursor.execute(query)

        temp = cursor.fetchone()

        airline  = temp[0]

        # Get the seat class

        seat\_class = input("Enter seat class (economy, business, first): ")

        # Generate a random PNR

        number = str(random.randint(1000, 9999))

        char = f'{company\_id}'.join(random.choices(string.ascii\_uppercase, k=4))

        pnr = char + number

        # Execute an INSERT query to add a new RESERVATION to the RESERVATION table

        query = f"INSERT INTO RESERVATION (PNR, EMAIL, FROM\_AIRPORT, TO\_AIRPORT, AIRLINE, SEAT\_CLASS) VALUES ('{pnr}', '{email}', '{from\_airport}', '{to\_airport}', '{airline}', '{seat\_class}')"

        cursor.execute(query)

        cnx.commit()

        print(f"Flight booked successfully! PNR: {pnr}")

        print("\*"\*50)

        print("\*"\*50)

        print("Ticket overview:")

        print(f"PNR: {pnr}")

        print(f"From: {from\_airport}")

        print(f"To: {to\_airport}")

        print(f"Airline: {company\_id}")

        print(f"Seat class: {seat\_class}")

        print(f"Email: {email}")

        print("\*"\*50)

def pie\_chart():

    # Execute a query to get the number of reservations for each airline

    query = '''

    SELECT AIRLINE, COUNT(\*) as count

    FROM RESERVATION

    GROUP BY AIRLINE

    '''

    cursor.execute(query)

    # Fetch the results

    results = cursor.fetchall()

    # Extract the data you want to plot

    airline\_ids = [row[0] for row in results]

    reservation\_counts = [row[1] for row in results]

    # Create the pie chart

    plt.pie(reservation\_counts, labels=airline\_ids, autopct='%1.1f%%')

    # Add a title

    plt.title('Percentage of Airlines choosed')

    # Show the plot

    plt.show()

def cancel\_flight():

    # Get the PNR, email, and password of the user

    pnr = input("Enter PNR: ")

    email = input("Enter email: ")

    password = input("Enter password: ")

    # Execute a SELECT query to get the user with the given email and password

    query = f"SELECT \* FROM USER WHERE EMAIL='{email}' AND PASSWORD='{password}'"

    cursor.execute(query)

    user = cursor.fetchone()

    if user is None:

        print("Invalid email or password.")

        print("\*"\*50)

    else:

        # Execute a SELECT query to get the RESERVATION with the given PNR and email

        query = f"SELECT \* FROM RESERVATION WHERE PNR='{pnr}' AND EMAIL='{email}'"

        cursor.execute(query)

        RESERVATION = cursor.fetchone()

        if RESERVATION is None:

            print("RESERVATION not found.")

        else:

            # Execute a DELETE query to delete the RESERVATION with the given PNR

            query = f"DELETE FROM RESERVATION WHERE PNR='{pnr}'"

            cursor.execute(query)

            cnx.commit()

            print("Flight cancelled successfully!")

            print("\*"\*50)

def ticket\_validation():

    # Get the PNR and email of the user

    pnr = input("Enter PNR: ")

    email = input("Enter email: ")

    # Execute a SELECT query to get the RESERVATION with the given PNR and email

    query = f"SELECT \* FROM RESERVATION WHERE PNR='{pnr}' AND EMAIL='{email}'"

    cursor.execute(query)

    RESERVATION = cursor.fetchone()

    if RESERVATION is None:

        print("Ticket not valid.")

        print("\*"\*50)

    else:

        # Execute a SELECT query to get the from airport and to airport with the given airport codes

        from\_query = f"SELECT \* FROM AIRPORT WHERE AIRPORT\_CODE='{RESERVATION[2]}'"

        to\_query = f"SELECT \* FROM AIRPORT WHERE AIRPORT\_CODE='{RESERVATION[3]}'"

        cursor.execute(from\_query)

        from\_airport = cursor.fetchone()

        cursor.execute(to\_query)

        to\_airport = cursor.fetchone()

        # Execute a SELECT query to get the airline company with the given company name

        query = f"SELECT \* FROM AIRLINE\_COMPANY WHERE COMPANY\_NAME = '{RESERVATION[4]}'"

        cursor.execute(query)

        company = cursor.fetchone()

        print("\*"\*50)

        print("Ticket overview:")

        print(f"PNR: {RESERVATION[0]}")

        print(f"From: {from\_airport[1]} ({RESERVATION[2]})")

        print(f"To: {to\_airport[1]} ({RESERVATION[3]})")

        print(f"Airline: {company[1]} ({RESERVATION[4]})")

        print(f"Seat class: {RESERVATION[5]}")

        print(f"Email: {RESERVATION[6]}")

        print("\*"\*50)

main()

**MySQL CODE:**

-- Creating the database

CREATE DATABASE AIRPLANE;

-- Use database

USE AIRPLANE;

-- Creating a Admin table

CREATE TABLE ADMIN (

Admin\_ID INT AUTO\_INCREMENT,

EMAIL VARCHAR(25) ,

FNAME VARCHAR(25) NOT NULL,

LNAME VARCHAR(25) NOT NULL,

PHONE BIGINT ,

PASSWORD VARCHAR(25) NOT NULL,

CONSTRAINT USER\_PK PRIMARY KEY (ADMIN\_ID,EMAIL)

);

-- Insert data into Admin table

INSERT INTO ADMIN(ADMIN\_ID, EMAIL, FNAME, LNAME, PHONE, PASSWORD) VALUES

(NULL, 'ayush.kadali@gmail.com', 'Ayush', 'Kadali', 5789546202, '123456789');

-- Creating a User table

CREATE TABLE USER (

USER\_ID INT AUTO\_INCREMENT,

EMAIL VARCHAR(25) ,

FNAME VARCHAR(25) NOT NULL,

LNAME VARCHAR(25) NOT NULL,

PHONE BIGINT ,

TYPE\_OF\_USER VARCHAR(25) NOT NULL,

PASSWORD VARCHAR(25) NOT NULL,

CONSTRAINT USER\_PK PRIMARY KEY (USER\_ID,EMAIL)

);

-- Insert data into USER table

INSERT INTO USER(USER\_ID, EMAIL, FNAME, LNAME, PHONE, TYPE\_OF\_USER, PASSWORD) VALUES

(NULL, 'priya.sharma@gmail.com', 'Priya', 'Sharma', 9876543212, 'Agent', '123456789'),

(NULL, 'krishna.kumar@gmail.com', 'Krishna', 'Kumar', 9876543213, 'Traveller', 'TIGER123'),

(NULL, 'ravi.shankar@gmail.com', 'Ravi', 'Shankar', 9876543214, 'Agent', 'RAVIAGENT'),

(NULL, 'sana.khan@gmail.com', 'Sana', 'Khan', 9876543215, 'Traveller','SANA@TRAVELLER'),

(NULL, 'ankit.jain@gmail.com', 'Ankit', 'Jain', 9876543216, 'Airline agent','123');

-- Creating a Airport table

CREATE TABLE AIRPORT(

AIRPORT\_CODE VARCHAR(25),

AIRPORT\_NAME VARCHAR(100) NOT NULL,

LOCATION VARCHAR(50) NOT NULL,

CITY VARCHAR(50) NOT NULL,

STATE VARCHAR(50) NOT NULL,

COUNTRY VARCHAR(50) NOT NULL,

ZIP INT NOT NULL,

CONSTRAINT AIRPORT\_PK PRIMARY KEY (AIRPORT\_CODE)

);

-- Insert data into Airport table

INSERT INTO AIRPORT(AIRPORT\_CODE, AIRPORT\_NAME, LOCATION, CITY, STATE, COUNTRY, ZIP) VALUES

('DEL', 'Indira Gandhi Airport', 'New Delhi', 'New Delhi', 'Delhi', 'India', 110037),

('BOM', 'Chhatrapati Shivaji', 'Mumbai', 'Mumbai', 'Maharashtra', 'India', 400099),

('BLR', 'Kempegowda Airport', 'Bengaluru', 'Bengaluru', 'Karnataka', 'India', 560008),

('CCU', 'Netaji Subhas Chandra', 'Kolkata', 'Kolkata', 'West Bengal', 'India', 700052),

('AMD', 'Sardar Vallabhbhai', 'Ahmedabad', 'Ahmedabad', 'Gujarat', 'India', 380015);

-- Create Airline Company table

CREATE TABLE AIRLINE\_COMPANY(

COMPANY\_ID VARCHAR(50) ,

COMPANY\_NAME VARCHAR(100) NOT NULL,

CONSTRAINT COMPANY\_PK PRIMARY KEY (COMPANY\_ID)

);

-- Insert data into Airline Company table

INSERT INTO AIRLINE\_COMPANY(COMPANY\_ID, COMPANY\_NAME) VALUES

('AI', 'Air India'),

('G8', 'GoAir'),

('SG', 'SpiceJet'),

('6E', 'IndiGo'),

('UK', 'Vistara');

CREATE INDEX email\_index ON USER (EMAIL);

-- Create a RESERVATION TABLE

CREATE TABLE RESERVATION(

PNR VARCHAR(20),

USER\_ID INT,

FROM\_AIRPORT VARCHAR(25) NOT NULL,

TO\_AIRPORT VARCHAR(25) NOT NULL,

AIRLINE VARCHAR(100) NOT NULL,

SEAT\_CLASS VARCHAR(25) NOT NULL,

EMAIL VARCHAR(25),

CONSTRAINT PNR\_PK PRIMARY KEY(PNR),

CONSTRAINT USER\_ID\_FK FOREIGN KEY (USER\_ID) REFERENCES USER(USER\_ID)

ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT FROM\_FK FOREIGN KEY (FROM\_AIRPORT) REFERENCES AIRPORT(AIRPORT\_CODE)

ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT TO\_FK FOREIGN KEY (TO\_AIRPORT) REFERENCES AIRPORT(AIRPORT\_CODE)

ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT AIRLINE\_FK FOREIGN KEY (AIRLINE) REFERENCES AIRLINE\_COMPANY(COMPANY\_NAME)

ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT EMAIL\_FK FOREIGN KEY (EMAIL) REFERENCES USER(EMAIL)

ON DELETE CASCADE ON UPDATE CASCADE

);

-- INSERT data into Reservation table

INSERT INTO RESERVATION (PNR, USER\_ID, FROM\_AIRPORT, TO\_AIRPORT, AIRLINE, SEAT\_CLASS, EMAIL) VALUES

('AI-GTVF8679', 1, 'DEL', 'BOM', 'Air India', 'Business','priya.sharma@gmail.com' ),

('6E-UVFP1061', 2, 'CCU', 'BLR', 'Indigo', 'Economy','krishna.kumar@gmail.com' ),

('UK-IQBQ6335', 3, 'BOM', 'AMD', 'Vistara', 'Business','ravi.shankar@gmail.com' ),

('G8-DPNI7220', 4, 'BLR', 'DEL', 'GoAir','Economy','sana.khan@gmail.com' ),

('AI-XMDF4536', 5, 'AMD', 'CCU', 'Air India', 'Business','ankit.jain@gmail.com' );

**Introduction**

The advancement of technology has greatly impacted the way we travel and make reservations. In the past, booking a flight was a time-consuming and tedious process that involved visiting a travel agent or calling the airline directly. However, with the introduction of computerized reservation systems, the process has become much simpler and more efficient.

A computerized airplane reservation system is a software program that allows users to search for and book flights online. This system is beneficial for both travellers and airlines as it allows for quick and convenient booking, as well as efficient management of reservations and ticket sales.

With a computerized reservation system, travellers can easily search for flights based on their preferred destination, dates, and budget. They can also view flight schedules, fares, and availability, and make reservations in just a few clicks. Additionally, the system often offers additional features such as the ability to select seats, add extras such as luggage or in-flight meals, and make payments securely.

For airlines, a computerized reservation system allows for efficient management of reservations and ticket sales, as well as the ability to track and analyse data on customer travel patterns and preferences. This information can be used to optimize routes, pricing, and marketing strategies.

In conclusion, a computerized airplane reservation system is a valuable tool that makes the process of booking flights easier and more convenient for travellers, while also providing benefits to airlines in terms of efficiency and data management.

**Bibliography**

* Computer science With Python - Class XI By : Sumita Arora
* Subject Teacher
* Internet
  + <https://www.geeksforgeeks.org/>
  + <https://stackoverflow.com/>
  + <https://www.w3schools.com/>
  + <https://www.codecademy.com/>