

AIRLINE MANAGEMENT SYSTEM



A

[DBMS] Course Project Report

in partial fulfilment of the degree

Bachelor of Technology in Computer Science & Engineering

By

| | |
|-----------|------------|
| P.Riteesh | 2103A51206 |
| K.Sainadh | 2103A51422 |
| D.vikas | 2103A51198 |
| M.Sandeep | 2103A51583 |

Under the guidance of

Mr. Mahendar

Assistant Professor , School of Computer Science and AI

Submitted to

School of Computer Science and Artificial Intelligence





DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the **DATABASE MANAGEMENT SYSTEMS- Course Project** Report entitled “ **AIRLINE MANAGEMENT SYSTEM** ” is a record of bonafide work carried out by the students “P.Riteesh , K.Sainadh , D.Vikas , M.Sandeep” bearing Roll No(s) 2103A51206,2103A51422,2103A51198,2103A51583 during the academic year 2019-20 in partial fulfillment of the award of the degree *Bachelor of Technology* in **Computer Science & Engineering**.

Lab In-charge

Head of the Department

INDEX

1. ABSTRACT
2. OBJECTIVES
3. PROPOSED METHODOLOGY
4. GRAPHICAL USER INTERFACE
5. DATABASE
6. CONCLUSION
7. DATASET
8. CODE

ABSTRACT

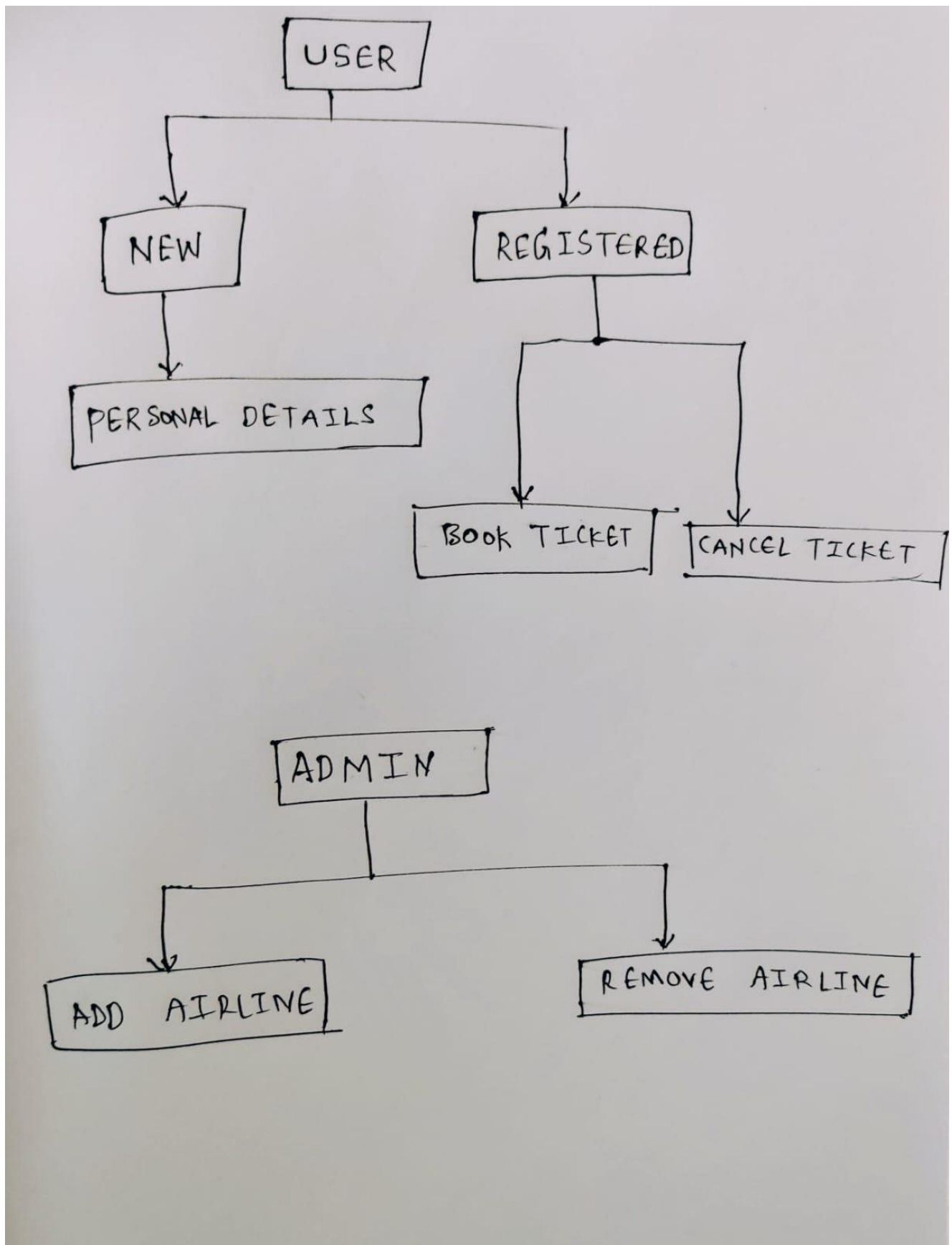
- The airline reservation database project is a software application built using Python and MySQL database, with a graphical user interface (GUI) designed using the Tkinter library. The system allows users to book and cancel flight tickets by providing their personal details and flight preferences, and also enables administrators to manage airline data, such as adding or removing airlines from the database.
- The user interface is simple and intuitive, with easy navigation and input validation to ensure accurate data entry. The system stores and retrieves flight data using MySQL database, ensuring data integrity and security.
- Overall, the airline reservation database project provides a comprehensive solution for managing flight reservations and airline data, with user-friendly features for both users and administrators.

OBJECTIVES

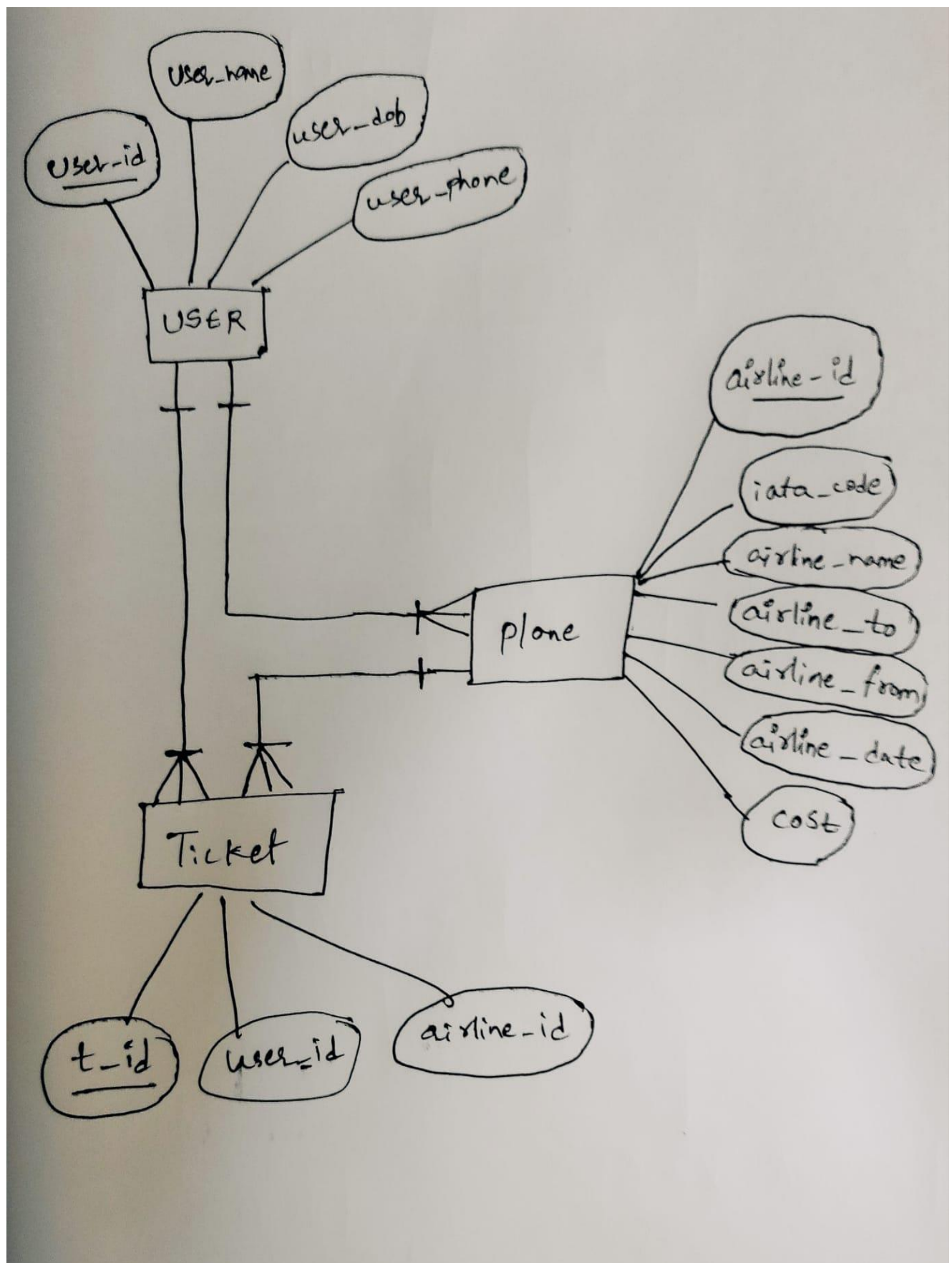
The objectives of the airline reservation database project are:

- To develop a user-friendly software application that allows users to book and cancel flight tickets easily, with minimal input errors and maximum accuracy.
- To provide a graphical user interface (GUI) using the Tkinter library, which is easy to navigate and visually appealing to users.
- To create a MySQL database to store and retrieve flight data, ensuring data integrity and security.
- To enable administrators to manage airline data efficiently, such as adding or removing airlines from the database.
- To incorporate input validation to ensure that the data entered by users is accurate and meets the required format and criteria.
- To provide error handling and exception handling to ensure that the system operates smoothly and does not crash or freeze during use.
- To implement security measures to protect user and airline data from unauthorized access or malicious attacks.
- To optimize the system's performance to ensure that it runs efficiently and without any significant delays or lags.

PROPOSED METHODOLOGY



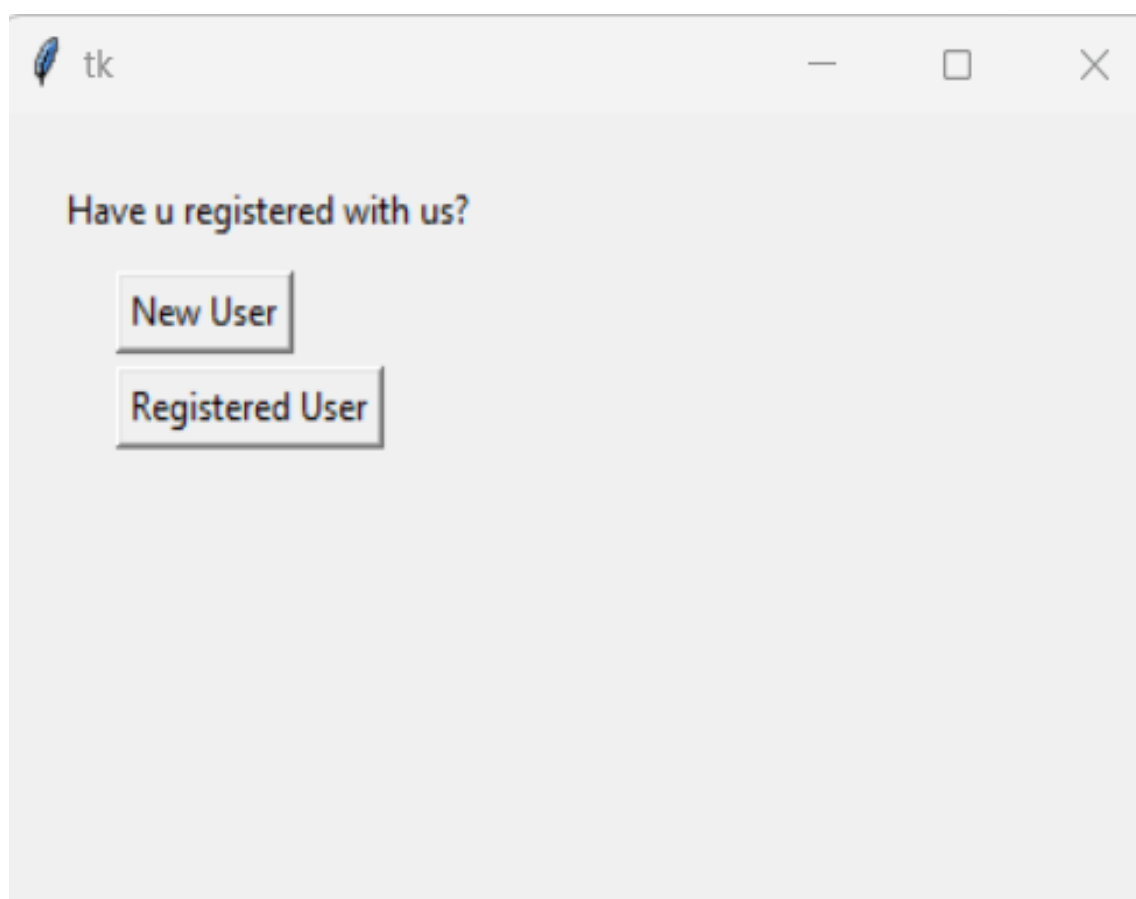
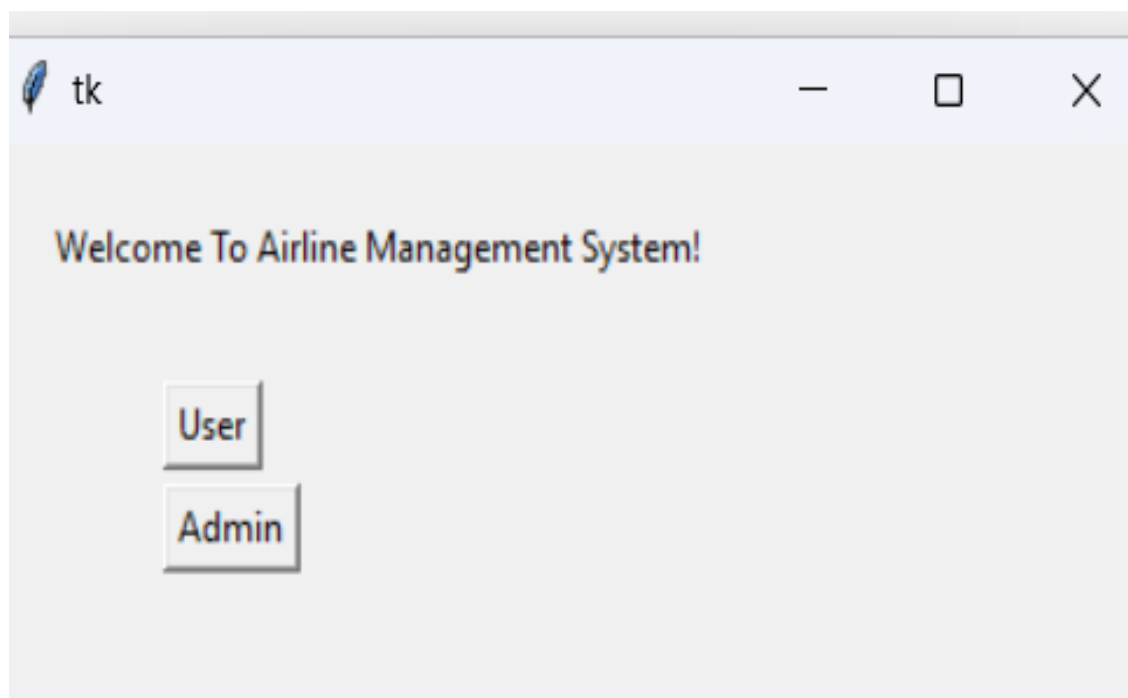
ER DIAGRAM



GRAPHICAL USER INTERFACE

- ✓ The GUI (Graphical User Interface) for this project allows users to book/cancel tickets and the admin to add/remove airlines.
- ✓ The GUI likely includes various widgets such as buttons, labels, entry fields, and drop-down menus, which allow the user to interact with the database. The user interface may also display relevant information such as available flights, costs, and schedules.
 - ✓ The primary objective of the GUI is to provide a user-friendly interface that allows users to interact with the database easily and efficiently. The interface should be intuitive and easy to navigate, with clear instructions and feedback provided to the user at each step.
- ✓ Overall, the GUI plays a critical role in the success of the project, as it determines the user's experience and satisfaction with the application.
- ✓ Tkinter is a built-in Python module used to create Graphical User Interfaces (GUI) for desktop applications. Tkinter provides a set of tools to create windows, buttons, labels, text boxes, and other widgets to build the user interface for desktop applications. It is based on the Tcl/Tk GUI toolkit and is a part of the Python standard library.

- ✓ Tkinter allows developers to create GUI applications for various platforms such as Windows, Mac, and Linux, and it can be used to create a wide range of applications from simple calculator programs to complex enterprise-level software.
- ✓ Some of the key features of Tkinter include its simplicity, ease of use, and ability to integrate with other Python modules and libraries. It also provides various layout managers such as grid, pack, and place, which help to organize the widgets on the window.
- ✓ In addition, Tkinter also allows developers to create custom widgets and graphics, making it a versatile tool for creating sophisticated and visually appealing applications.
- ✓ Overall, Tkinter is a popular choice for building desktop applications due to its ease of use, versatility, and wide availability.



tk

Enter Personal Details!

User Name:

DOB:

Contact number:

tk

Enter Personal Details!

User Name:

DOB:

Contact number:

{{(7,,)}}

Note: This will be your User ID.

tk

Choose an Action

Book Ticket

Cancel ticket

tk

Enter Journey Details!


User ID:

From:

To:

Travel date:

Done

 tk

Enter Journey Details!

User ID:

7

From:

mumbai


To:

delhi

Travel date:

2023-04-07

Done

 tk

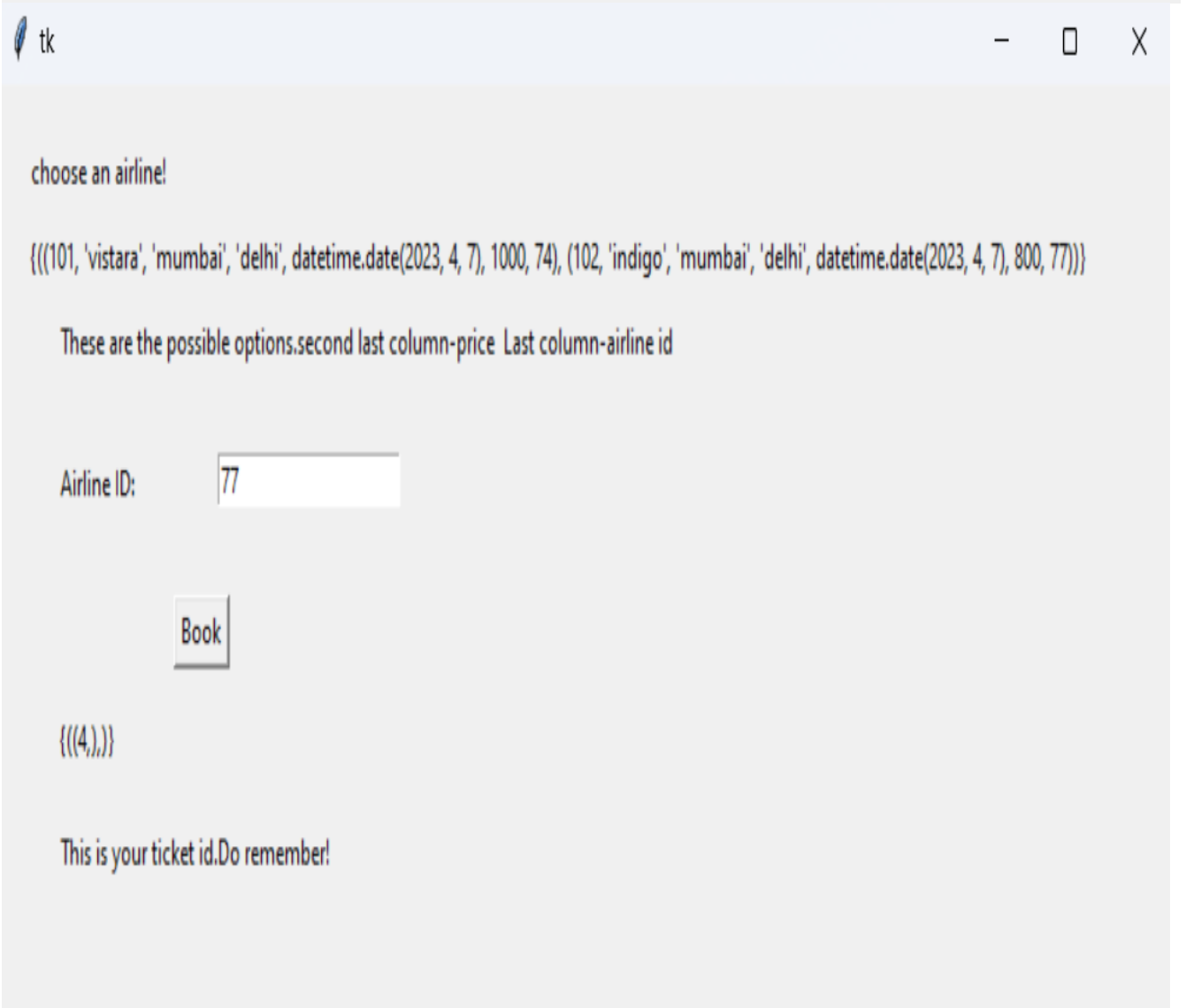
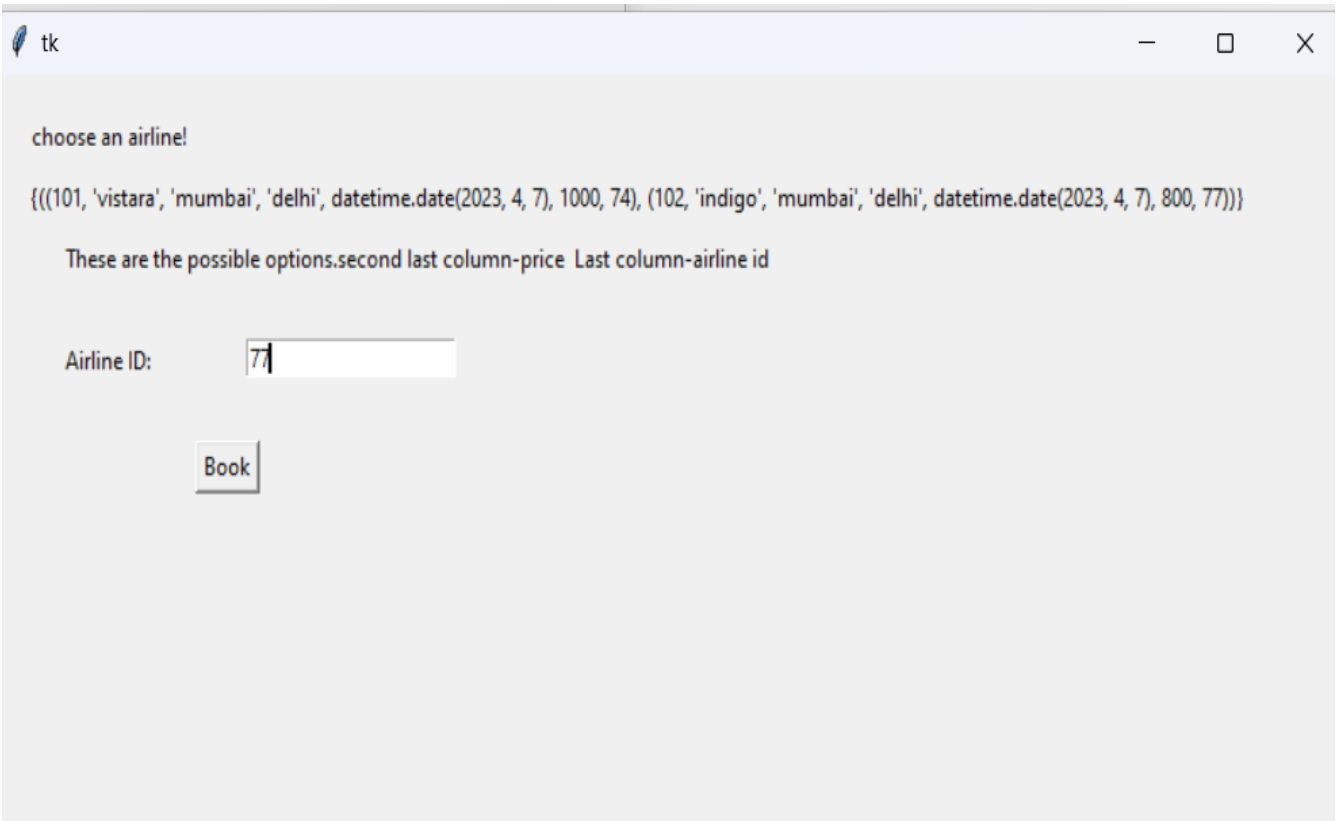
choose an airline!

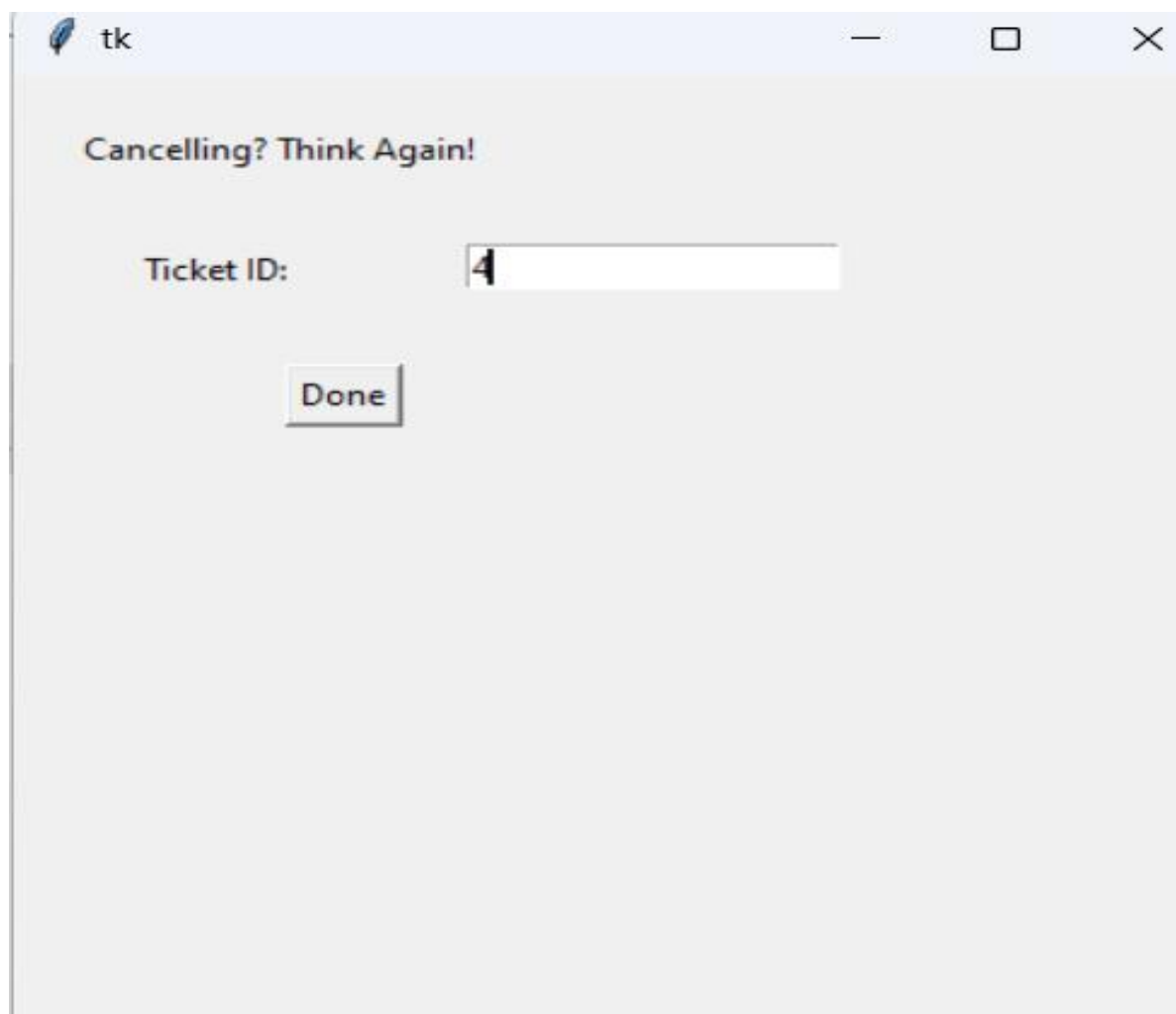
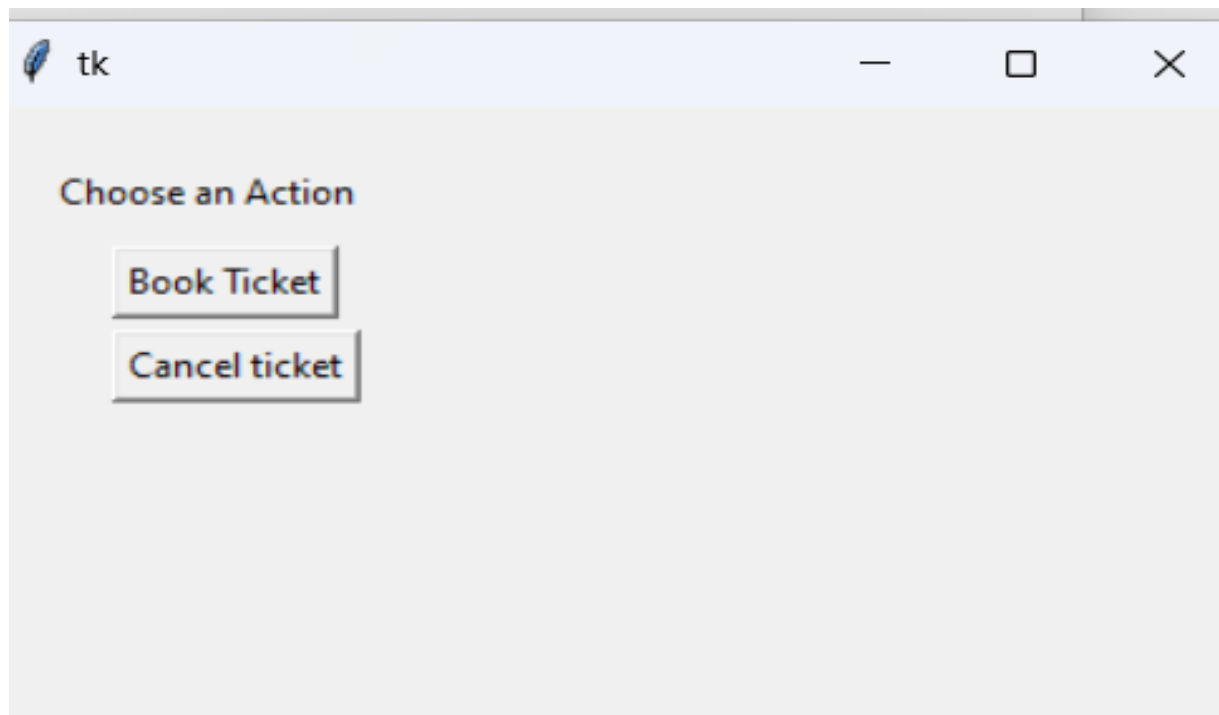
```
{{(101, 'vistara', 'mumbai', 'delhi', datetime.date(2023, 4, 7), 1000, 74), (102, 'indigo', 'mumbai', 'delhi', datetime.date(2023, 4, 7), 800, 77)}}
```

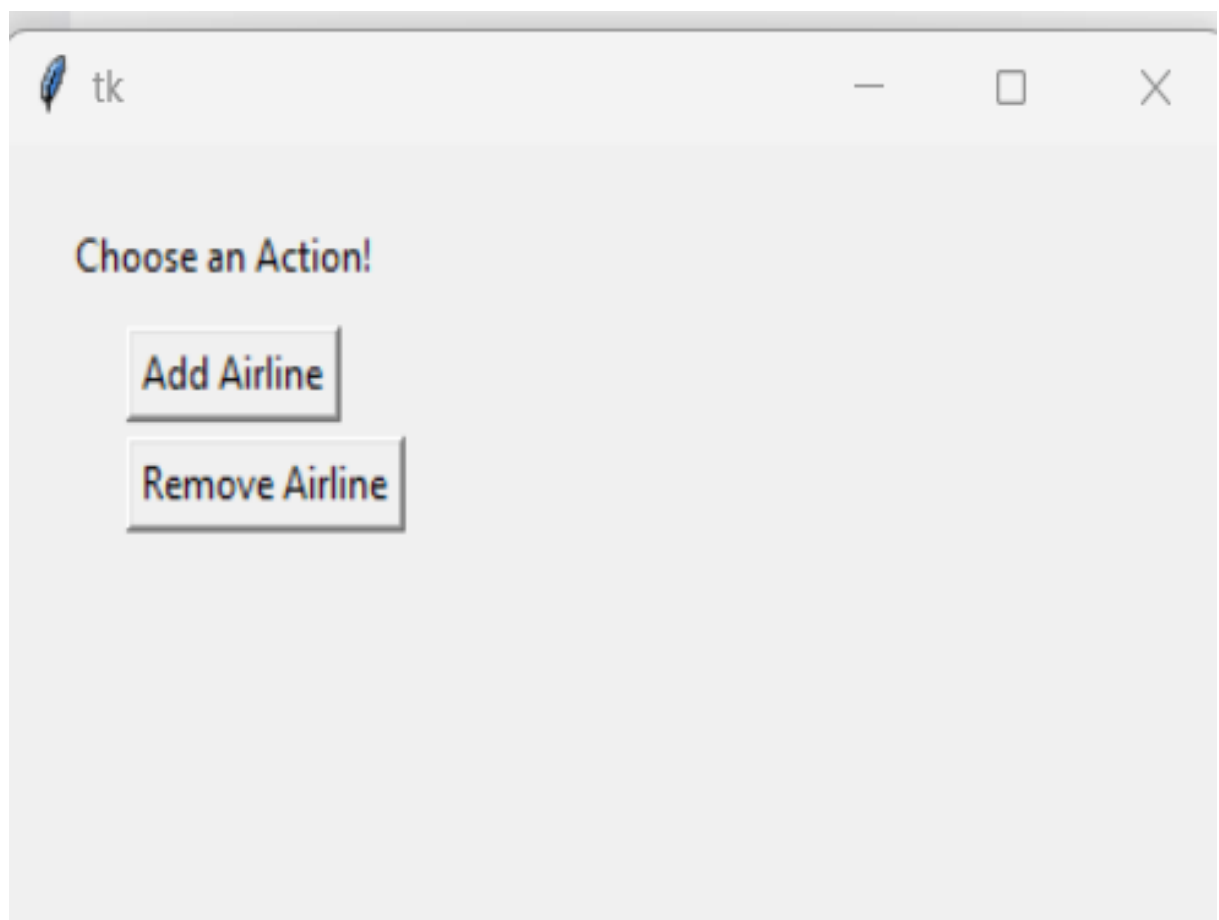
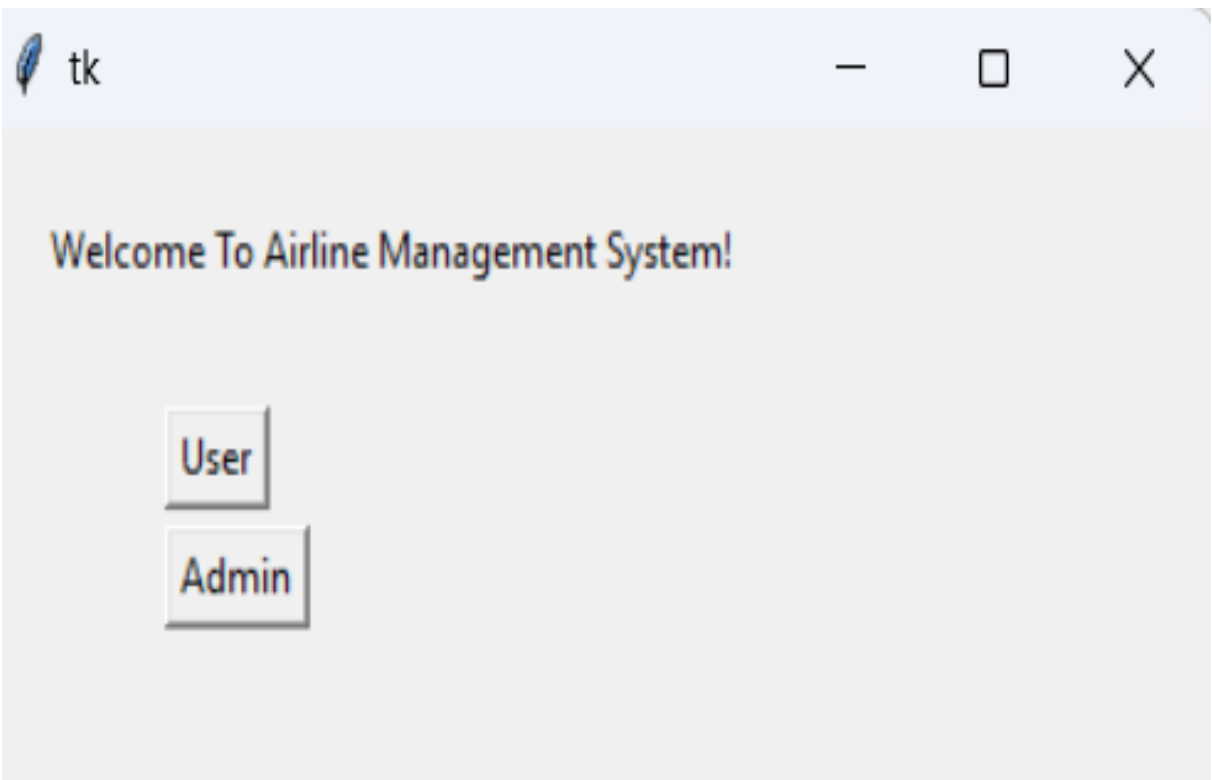
These are the possible options.second last column-price Last column-airline id


Airline ID:

Book







 tk

Enter Airline Details!

Iata code:

Airline Name:


From:

To:

Travel Date:

cost:

Done

 tk

Enter Airline Details!

Iata code:

108

Airline Name:

air india

From:

hyderabad

To:

delhi

Travel Date:

2023-04-20

cost:

1200

Done

tk

Enter Airline Details!

| | |
|---------------|------------|
| Iata code: | 108 |
| Airline Name: | air india |
| From: | hyderabad |
| To: | delhi |
| Travel Date: | 2023-04-20 |
| cost: | 1200 |

Done

{{(126,,,)}}

Note: Remember this ID - Useful to remove airline

tk

Organisational Issues! Let us Know!

| | |
|-------------|-----|
| Airline ID: | 126 |
|-------------|-----|

Done

DATABASE

- ✓ The database includes three tables: "USER", "PLANE", and "TICKET". The "USER" table stores information about users, the "PLANE" table stores information about planes and flights, and the "TICKET" table stores information about the tickets booked by users.
- ✓ The database includes various relationships between the tables, such as a one-to-many relationship between "USER" and "TICKET", a one-to-many relationship between "PLANE" and "TICKET", and a one-to-many relationship between "USER" and "PLANE".
- ✓ The main objective of the database is to store and manage information related to airline reservations, including user information, flight information, and ticket information. The database allows users to book and cancel tickets, and the admin to add and remove airlines.
- ✓ The database design follows the principles of normalization, which helps to ensure data consistency and eliminate redundancy. Each table includes a primary key to uniquely identify each record, and foreign keys are used to establish relationships between the tables.
- ✓ Overall, the airline reservation database you built provides a solid foundation for managing airline reservations and bookings. With proper maintenance and optimization, the database can be scaled up to support larger and more complex applications.

MYSQL

- ✓ MySQL is an open-source relational database management system (RDBMS) that is widely used for building scalable and efficient web applications. MySQL provides a variety of tools and features to store, manage, and access data in an organized and efficient manner.
- ✓ Some of the key features of MySQL include its ability to handle large amounts of data, its scalability, its robustness and reliability, and its support for various programming languages and platforms. MySQL also provides a range of data types, indexes, and constraints to ensure data consistency and integrity.
- ✓ MySQL uses the SQL (Structured Query Language) programming language to interact with the database, allowing users to create, modify, and query data in tables. It also provides various tools for database administration, such as MySQL Workbench, which allows users to manage and monitor databases and perform database operations such as backups and restores.
- ✓ MySQL is widely used by web developers and businesses of all sizes, from small startups to large enterprises, due to its scalability, reliability, and ease of use. It is also compatible with various programming languages and platforms, making it a versatile tool for building web applications.

```

1  CREATE TABLE `plane` (
2      `iata_code` int NOT NULL,
3      `airline_name` varchar(45) NOT NULL,
4      `airline_from` varchar(45) NOT NULL,
5      `airline_to` varchar(45) NOT NULL,
6      `airline_date` date NOT NULL,
7      `cost` int NOT NULL,
8      `airline_id` int NOT NULL AUTO_INCREMENT,
9      PRIMARY KEY (`airline_id`)
10 ) ENGINE=InnoDB AUTO_INCREMENT=127 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci

```

| Column | Type |
|----------------|-------------|
| ✧ iata_code | int |
| ✧ airline_name | varchar(45) |
| ✧ airline_from | varchar(45) |
| ✧ airline_to | varchar(45) |
| ✧ airline_date | date |
| ✧ cost | int |
| ✧ airline_id | int |





```
1 CREATE TABLE `ticket` (  
2   `user_id` int NOT NULL,  
3   `airline_id` int NOT NULL,  
4   `t_id` int NOT NULL AUTO_INCREMENT,  
5   PRIMARY KEY (`t_id`)  
6 ) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

| Column | Type |
|--|------|
|  user_id | int |
|  airline_id | int |
|  t_id | int |

```

1 CREATE TABLE `user` (
2   `user_id` int NOT NULL AUTO_INCREMENT,
3   `user_name` varchar(45) NOT NULL,
4   `user_dob` date NOT NULL,
5   `user_phone` bigint NOT NULL,
6   PRIMARY KEY (`user_id`)
7 ) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci

```

| Column | Type |
|--|-------------|
|  user_id | int |
|  user_name | varchar(45) |
|  user_dob | date |
|  user_phone | bigint |

PYMYSQL

- ✓ PyMySQL is a Python module used to connect to and interact with MySQL databases. It is a pure Python implementation of the MySQL client-server protocol and can be used to connect to MySQL databases on various platforms such as Windows, Mac, and Linux.
- ✓ PyMySQL provides a set of functions and methods to execute SQL queries, fetch data from tables, and perform database operations such as inserting, updating, and deleting data. It also supports transactions and provides error handling mechanisms to handle exceptions and errors that may occur during database operations.
- ✓ Some of the key features of PyMySQL include its simplicity, ease of use, and ability to integrate with other Python modules and libraries. It also supports various versions of MySQL and provides compatibility with various Python versions.
- ✓ Overall, PyMySQL is a popular choice for interacting with MySQL databases in Python applications due to its simplicity, ease of use, and wide availability. It allows developers to build scalable and efficient applications that can handle large amounts of data and provide robust data management capabilities.

CONCLUSION

- ✓ In conclusion, the airline reservation database project you built using Tkinter, Python, and MySQL is a simple yet effective solution for managing airline reservations and bookings. The project includes a user-friendly GUI that allows users to book and cancel tickets, and the admin to add and remove airlines. The database design follows the principles of normalization and includes various relationships between tables to ensure data consistency and eliminate redundancy.
- ✓ The project achieved its objectives of providing a user-friendly interface for managing airline reservations, and the database provides a solid foundation for storing and managing information related to airline reservations. With further development and optimization, the project can be scaled up to support larger and more complex applications.
- ✓ Overall, this project demonstrates the importance of a well-designed database and user interface in building successful and efficient applications.

DATASET

| iata_code | airline_name | airline_from | airline_to | airline_date | cost | airline_id |
|-----------|--------------|--------------|------------|--------------|------|------------|
| 101 | vistara | delhi | mumbai | 01-04-2023 | 1000 | 1 |
| 101 | vistara | mumbai | delhi | 01-04-2023 | 1000 | 2 |
| 101 | vistara | delhi | chennai | 01-04-2023 | 1200 | 3 |
| 101 | vistara | chennai | delhi | 01-04-2023 | 1200 | 4 |
| 102 | indigo | mumbai | delhi | 01-04-2023 | 800 | 5 |
| 102 | indigo | delhi | mumbai | 01-04-2023 | 800 | 6 |
| 102 | indigo | mumbai | chennai | 01-04-2023 | 600 | 7 |
| 102 | indigo | chennai | mumbai | 01-04-2023 | 600 | 8 |
| 103 | spicejet | chennai | mumbai | 01-04-2023 | 500 | 9 |
| 103 | spicejet | mumbai | chennai | 01-04-2023 | 500 | 10 |
| 103 | spicejet | chennai | delhi | 01-04-2023 | 1000 | 11 |
| 103 | spicejet | delhi | chennai | 01-04-2023 | 1000 | 12 |
| 101 | vistara | delhi | mumbai | 02-04-2023 | 1000 | 13 |
| 101 | vistara | mumbai | delhi | 02-04-2023 | 1000 | 14 |
| 101 | vistara | delhi | chennai | 02-04-2023 | 1200 | 15 |
| 101 | vistara | chennai | delhi | 02-04-2023 | 1200 | 16 |
| 102 | indigo | mumbai | delhi | 02-04-2023 | 800 | 17 |
| 102 | indigo | delhi | mumbai | 02-04-2023 | 800 | 18 |
| 102 | indigo | mumbai | chennai | 02-04-2023 | 600 | 19 |
| 102 | indigo | chennai | mumbai | 02-04-2023 | 600 | 20 |
| 103 | spicejet | chennai | mumbai | 02-04-2023 | 500 | 21 |
| 103 | spicejet | mumbai | chennai | 02-04-2023 | 500 | 22 |
| 103 | spicejet | chennai | delhi | 02-04-2023 | 1000 | 23 |
| 103 | spicejet | delhi | chennai | 02-04-2023 | 1000 | 24 |

CODE

```
from tkinter import *
import pymysql.cursors
m=Tk()
m.geometry("400x200")
def bt():
    def dbbt():
        def dbt():
            db=pymysql.connect(host="localhost",user="root",password="tiger",database
="myth")
            cur=db.cursor()
            cur.execute(f"insert into ticket(user_id,airline_id)
values({ui.get()}},{ai.get()}")
            db.commit()
            cur.execute(f"select t_id from ticket where airline_id={ai.get()} and
user_id={ui.get()} ")
            out=cur.fetchall()
            Label(x,text={out}).place(x=40,y=220)
            Label(x,text="This is your ticket id.Do remember!").place(x=40,y=260)
            db.close()
        x=Tk()
        x.geometry("800x600")
        Label(x,text="choose an airline!").place(x=20,y=20)
        db=pymysql.connect(host="localhost",user="root",password="tiger",database="my
th")
        cur=db.cursor()
        cur.execute(f"select * from plane where airline_from='{fr.get()}' and
airline_to='{to.get()}' and airline_date='{td.get()}'")
        out=cur.fetchall()
        Label(x,text={out}).place(x=20,y=50)
        Label(x,text="These are the possible options.second last column-price Last
column-airline id").place(x=40,y=80)
        Label(x,text="Airline ID:").place(x=40,y=130)
        ai=Entry(x)
        ai.place(x=150,y=130)
        Button(x,text="Book",command=dbt).place(x=120,y=180)
        db.close()
    u=Tk()
    u.geometry("400x600")
    Label(u,text="Enter Journey Details!").place(x=20,y=20)
    Label(u,text="User ID:").place(x=40,y=50)
    ui=Entry(u)
    ui.place(x=150,y=50)
    Label(u,text="From:").place(x=40,y=80)
    fr=Entry(u)
    fr.place(x=150,y=80)
    Label(u,text="To:").place(x=40,y=110)
    to=Entry(u)
    to.place(x=150,y=110)
    Label(u,text="Travel date:").place(x=40,y=140)
    td=Entry(u)
    td.place(x=150,y=140)
    Button(u,text="Done",command=dbbt).place(x=120,y=180)
```

```

def ct():
    def dbct():
        db=pymysql.connect(host="localhost",user="root",password="tiger",database="my
th")
        cur=db.cursor()
        cur.execute(f"delete from ticket where t_id={ti.get()}")
        db.commit()
        db.close()
    u=Tk()
    u.geometry("400x600")
    Label(u,text="Cancelling? Think Again!").place(x=20,y=20)
    Label(u,text="Ticket ID:").place(x=40,y=70)
    ti=Entry(u)
    ti.place(x=150,y=70)
    Button(u,text="Done",command=dbct).place(x=90,y=120)
def aa():
    def dbaa():
        db=pymysql.connect(host="localhost",user="root",password="tiger",database="my
th")
        cur=db.cursor()
        cur.execute(f"insert into
plane(iata_code,airline_name,airline_from,airline_to,airline_date,cost) values({ic.g
et()},'{an.get()}', '{fr.get()}', '{to.get()}', '{td.get()}', '{co.get()}')")
        db.commit()
        cur.execute(f"select airline_id from plane where iata_code={ic.get()} and
airline_name='{an.get()}' and airline_from='{fr.get()}' and airline_to='{to.get()}'
and airline_date='{td.get()}' and cost={co.get()}")
        out=cur.fetchall()
        Label(u,text={out}).place(x=40,y=280)
        Label(u,text="Note: Remember this ID - Useful to remove
airline").place(x=40,y=360)
        db.close()
    u=Tk()
    u.geometry("400x600")
    Label(u,text="Enter Airline Details!").place(x=20,y=20)
    Label(u,text="Iata code:").place(x=40,y=50)
    ic=Entry(u)
    ic.place(x=150,y=50)
    Label(u,text="Airline Name:").place(x=40,y=80)
    an=Entry(u)
    an.place(x=150,y=80)
    Label(u,text="From:").place(x=40,y=110)
    fr=Entry(u)
    fr.place(x=150,y=110)
    Label(u,text="To:").place(x=40,y=140)
    to=Entry(u)
    to.place(x=150,y=140)
    Label(u,text="Travel Date:").place(x=40,y=170)
    td=Entry(u)
    td.place(x=150,y=170)
    Label(u,text="cost:").place(x=40,y=200)
    co=Entry(u)
    co.place(x=150,y=200)

```

```

        Button(u, text="Done", command=dbaa).place(x=120, y=250)
def ra():
    def dbra():
        db=pymysql.connect(host="localhost", user="root", password="tiger", database="my
th")
        cur=db.cursor()
        cur.execute(f"delete from plane where airline_id={ai.get()}")
        db.commit()
        db.close()
    u=Tk()
    u.geometry("400x600")
    Label(u, text="Organisational Issues! Let us Know!").place(x=20, y=20)
    Label(u, text="Airline ID:").place(x=40, y=50)
    ai=Entry(u)
    ai.place(x=150, y=50)
    Button(u, text="Done", command=dbra).place(x=90, y=80)
def admin():
    u=Tk()
    u.geometry("400x400")
    Label(u, text="Choose an Action!").place(x=20, y=20)
    Button(u, text="Add Airline", command=aa).place(x=40, y=50)
    Button(u, text="Remove Airline", command=ra).place(x=40, y=80)
def pd():
    def dbpd():
        db=pymysql.connect(host="localhost", user="root", password="tiger", database="my
th")
        cur=db.cursor()
        cur.execute(f"insert into user(user_name,user_dob,user_phone)
values('{un.get()}', '{ud.get()}', '{up.get()}')")
        db.commit()
        cur.execute(f"select user_id from user where user_name='{un.get()}' and
user_dob='{ud.get()}' and user_phone={up.get()}")
        out=cur.fetchall()
        Label(u, text={out}).place(x=40, y=330)
        Label(u, text="Note: This will be your User ID.").place(x=40, y=360)
    u=Tk()
    u.geometry("400x600")
    Label(u, text="Enter Personal Details!").place(x=20, y=20)
    Label(u, text="User Name:").place(x=40, y=80)
    un=Entry(u)
    un.place(x=150, y=80)
    Label(u, text="DOB:").place(x=40, y=110)
    ud=Entry(u)
    ud.place(x=150, y=110)
    Label(u, text="Contact number:").place(x=40, y=140)
    up=Entry(u)
    up.place(x=150, y=140)
    Button(u, text="Done", command=dbpd).place(x=120, y=300)
def zen():
    u=Tk()
    u.geometry("400x400")
    Label(u, text="Choose an Action").place(x=20, y=20)
    Button(u, text="Book Ticket", command=bt).place(x=40, y=50)

```

```
    Button(u,text="Cancel ticket",command=ct).place(x=40,y=80)
def user():
    u=Tk()
    u.geometry("400x400")
    Label(u,text="Have u registered with us?").place(x=20,y=20)
    Button(u,text="New User",command=pd).place(x=40,y=50)
    Button(u,text="Registered User",command=zen).place(x=40,y=80)

Label(m,text="Welcome To Airline Management System!").place(x=20,y=20)
Button(m,text="User",command=user).place(x=60,y=70)
Button(m,text="Admin",command=admin).place(x=60,y=100)
m.mainloop()
```