HOLY ANGEL UNIVERSITY

School of Computing

FLIGHT MANAGEMENT SYSTEM

EMP Airlines

FINAL CASE STUDY

(6INFOMAN)

Submitted by:

Edquiban, Andre Rafael

Meneses, Jashper Troy

Perez, Myco Paul John

Submitted to:

Mr. Rene D. Laguna

Date:

March 29, 2020

Flight Management System – EMP Airlines

The airport needs an application to store data of their customers. In other terms, it helps the airport easily identify flights and departure and customer basic data. It can reduce the manual workload due to digitalized system which will ease the production of data about flights.

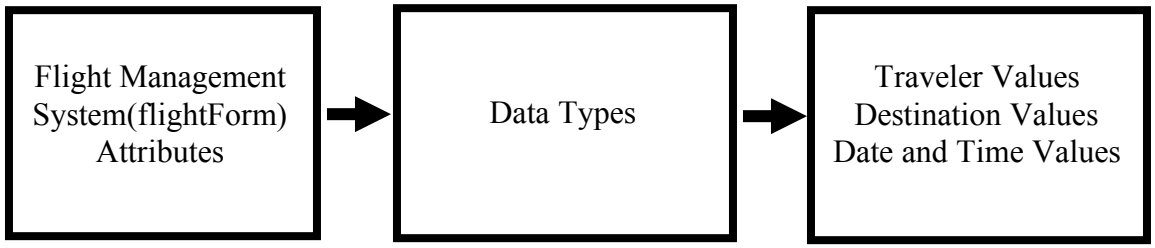
According to Airport Supplier (2019), it is really the front end to an airport management system. It displays information to the public such as flight, time and departure of each plane. Systems like this nowadays are being used to store new data and the database can help the employer have an easy time identifying flight and customer details.

According to Geertjan Wielenga (2010), it is intended to provide an effective way for airports to handle their operations. Airport employees use the system to plan and control all type of flights, airport use and flight schedules, among other things.

According to A-ICE (2019), the program helps create an airport experience that is smooth, keeping passengers updated at all levels is important. Data must be correct and up-to-date, and displayed so passengers can see it instantly to prevent congestion.

In the study, The ACRP (2009) mentioned that the airline is directly responsible for updating, maintain and informing both the airport and the public in real time of its flight operation. Often FIDS are system that are manually modified. The data obtained by Courteney, Hazel (n.d) in the FMS analysis demonstrate that there are several imperfections that are common to line crews working in service. While systems exist that purport to move in service problems back to manufacturers, respondents in the present study indicated that none of the events in this study have been recorded through other avenues.

Conceptual/Theoretical Framework



The conceptual framework above shows the particular process of how the variables in the system would work. The EMP Airline Information System, which is the system itself, along with the entities (flyingFrom, flyingTo, departingOn, returningOn, departureTime, returningTime, fullName, email, phone, nationality, gender, passportNo, passportExpiryDate), and the attributes of those entities represent the independent variable in the study which can be manipulated or controlled by the researchers. The intervening variables are the data type of the attributes, which connects independent to the dependent variable. As the term says itself, it intervenes between the two other variables which will affect the values or the dependent variable. The values given by the user, Traveler Values, Destination Values, and Date and Time values, are the dependent variables. The outcome of the dependent variables, the result of records, will rely on the independent variable with respect to the intervening variables.

Objectives of the Study

The EMP Airline Information System is a proposed system for the reliability of the users related to the program system developed. Therefore, it offers to achieve the following objectives:

1. To be able to create flight
2. To be able to insert flight to database
3. To be able to show flight into list of flights

Scope and Delimitation

This study focused on the data logs recorded of the individuals in the airport by creating an information system that displays flight logs.

This study is intended for airport management employees. It discusses database structures and functions that would be necessary for the system to work. The system contains a registration form process for flight registration, display of information and SQL commands, for the queries, including Select and Insert

The system is designed for the use of management employees and is focused on the flight logs information and other factors that could be related to it, which are: flight time, destination, departure, and return.

Furthermore, the study is only applied to those who are relative to the system, which are flight information management. The case study only attempted to create an information database for gathering the travelers' flight details such as departure and destination.

Significance of the Study

The Flight Management System works to integrate the information of the flights booked by travelers, where they are heading and their time of departure and arrival. The study will also benefit the following:

Traveler

The reason for providing an information database system is to make the available information be easily identified by the management team and have copy of the details for flight cancellation or changing flight date. This could contribute a big development to many airports, when we are living in such a modernized place.

Country/Economy

Not only this could be developed in an airport, but the same idea of database could be used elsewhere. The power of computer can be taken anywhere and database, alone, is an easy way to store and send more than a hundred of information. The management system can be developed with other aspects of the community that uses people's records.

Future Researchers

The study will benefit the future researchers who will conduct similar or related studies. The study will serve as a basis and research material to them for developing a much more advanced system regarding database

Developed Information System

```
-- phpMyAdmin SQL Dump
-- version 4.9.1
-- https://www.phpmyadmin.net/
--
-- Host: localhost
-- Generation Time: Mar 25, 2020 at 04:08 PM
-- Server version: 10.4.8-MariaDB
-- PHP Version: 7.1.32

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET AUTOCOMMIT = 0;
START TRANSACTION;
SET time_zone = "+00:00";



/*!40101 SET
@OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
```

```
/*!40101 SET
@OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET
@OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `Airlines`
--

-- --------------------------------------------------------

--
-- Table structure for table `flightForm`
--

CREATE TABLE `flightForm` (
  `flyingFrom` varchar(50) NOT NULL,
  `flyingTo` varchar(50) NOT NULL,
  `departingOn` date DEFAULT NULL,
  `returningOn` date DEFAULT NULL,
  `departureTime` time DEFAULT NULL,
  `returningTime` time DEFAULT NULL,
  `fullName` varchar(50) NOT NULL,
  `email` varchar(50) NOT NULL,
  `phone` int(20) NOT NULL,
  `nationality` varchar(50) NOT NULL,
  `gender` varchar(10) NOT NULL,
  `passportNo` int(20) NOT NULL,
  `passportExpiryDate` date DEFAULT NULL
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `flightForm`
--

INSERT INTO `flightForm` (`flyingFrom`, `flyingTo`, `departingOn`, `returningOn`,
`departureTime`, `returningTime`, `fullName`, `email`, `phone`, `nationality`,
`gender`, `passportNo`, `passportExpiryDate`) VALUES
('a', 'a', '2020-01-01', '2020-01-01', '01:00:00', '02:00:00', 'a', 'a', 123, 'a', 'a', 222,
'2020-02-02'),
('b', 'b', '2020-02-02', '2020-03-30', '02:00:00', '03:00:00', 'b', 'b', 123, 'b', 'b', 333,
'2025-04-04');
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET
CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION
*/;
```

Database Components, Entity Relationship Diagram and Data Dictionary

The Entity Relationship Diagram represents the model of Flight Management System Entity. The entity-relationship diagram of EMP Airlines shows all the visual instrument of database tables and the relations between airline destination and passenger information. It used structure data and to define the relationships between structured data groups of Flight Management System functionalities. The main entities of EMP Airlines are Destination Entity and Passenger Entity.

EMP Airlines entities and their attributes:

- **Destination Entity:** flyingFrom, flyingTo, departingOn, returningOn, departureTime, returningTime

- **Passenger Entity:** passenger_name, passenger_email, passenger_phone, passenger_nationality, passenger_gender, passenger_passportNo, passenger_passportExpiryDate

References

Courteney, Hazel (n.d). *Assessing error tolerance in Flight Management Systems.*https://www.HeAlert.org/Filemanager/Root/Site_Assets/Standalone_Articles_Not_Linked_To_A_Bulletin/HE00155.Pdf

The ACRP (2009). *Integrating Airport Information Systems.* https://www.nap.edu/read/14234/chapter/7

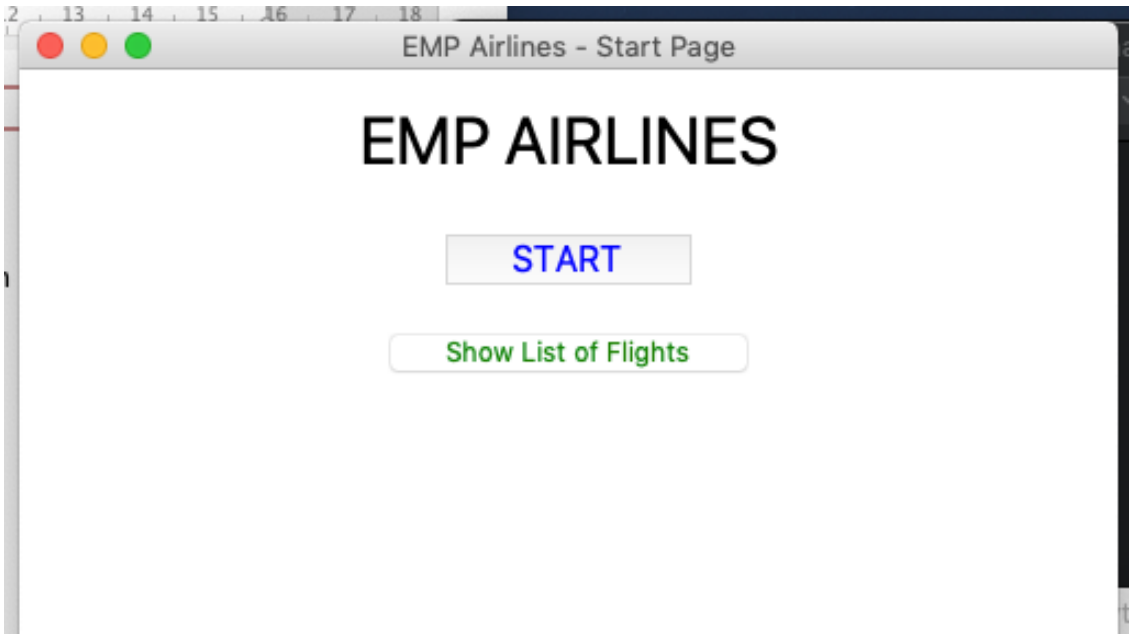
Geertjan Wielenga (2010). *Airport Operation Management on Oracle and the NetBeans Platform*. https://dzone.com/articles/netbeans-airport-operation-management

A-ICE (2019). *How the Flight Information Display System Can Help Airports with Passenger Traffic Increases*. https://www.a-ice.aero/how-flight-information-display-systems-can-help-airports-with-passenger-traffic-increases/

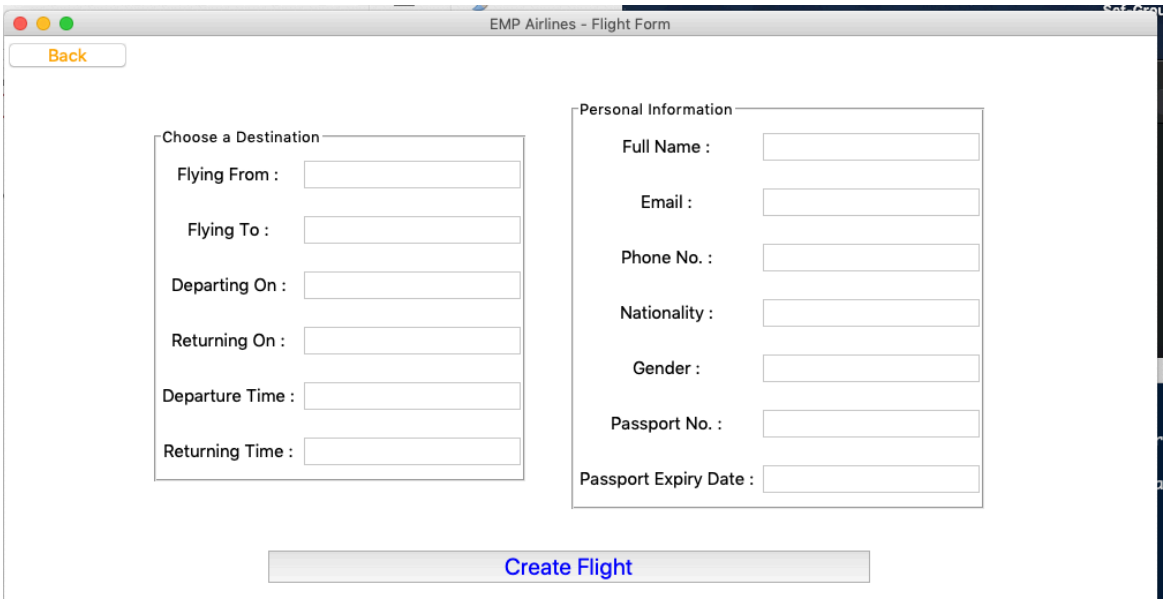
The Airport Supplier (2019). *Flight Information Display System and Aeronautical Billing System.* https://www.airport-suppliers.com/supplier/airport-information-systems-ais/

APPENDIX A

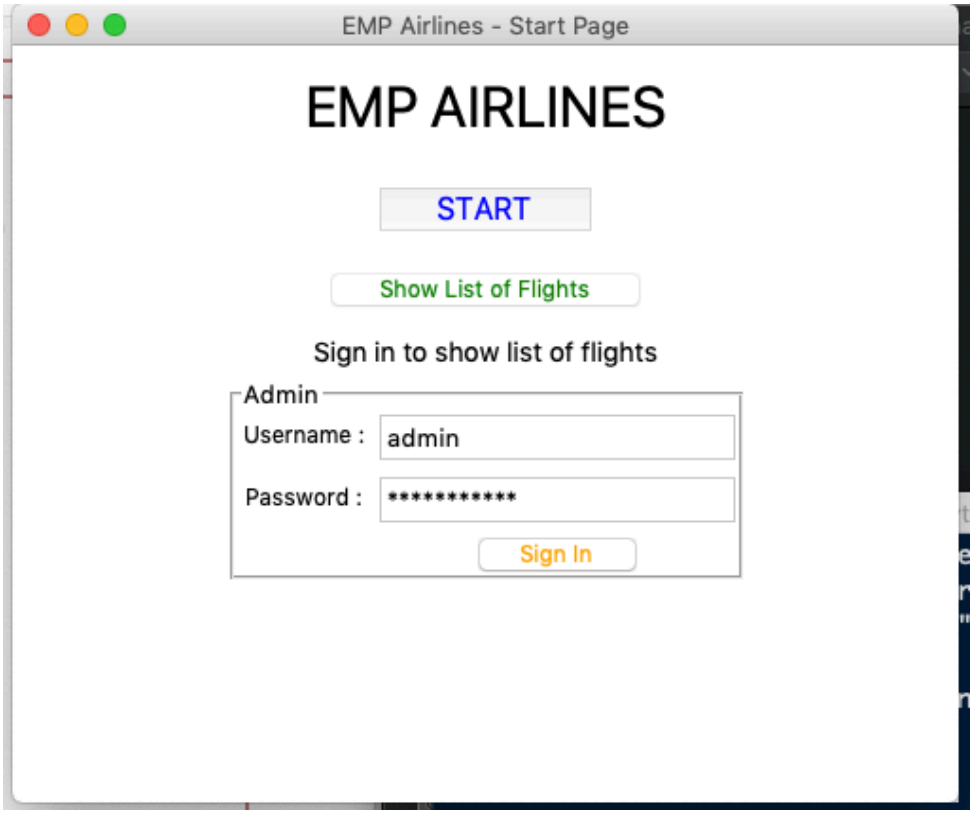
User's Manual



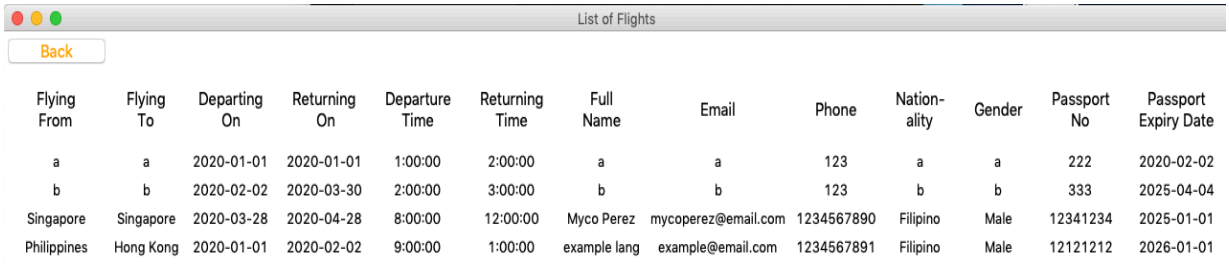
This is the Start Page of the Flight Management System.



This is the flight form of the Flight Management System. This is where the user inputs some data to create a flight.



On the start page you can show the list of flights by clicking the 'Show List of Flights' button, then sign in using a static username and password.



Flying From	Flying To	Departing On	Returning On	Departure Time	Returning Time	Full Name	Email	Phone	Nation-ality	Gender	Passport No	Passport Expiry Date
a	a	2020-01-01	2020-01-01	1:00:00	2:00:00	a	a	123	a	a	222	2020-02-02
b	b	2020-02-02	2020-03-30	2:00:00	3:00:00	b	b	123	b	b	333	2025-04-04
Singapore	Singapore	2020-03-28	2020-04-28	8:00:00	12:00:00	Myco Perez	mycoperez@email.com	1234567890	Filipino	Male	12341234	2025-01-01
Philippines	Hong Kong	2020-01-01	2020-02-02	9:00:00	1:00:00	example lang	example@email.com	1234567891	Filipino	Male	12121212	2026-01-01

This shows the table of flights booked or created by the travelers. It also shows their flight destination, time and information.

APPENDIX B

Sample Reports

```
MariaDB [Airlines]> show tables;
+-------------------+
| Tables_in_Airlines |
+-------------------+
| flightForm        |
+-------------------+
1 row in set (0.128 sec)
```

This shows the tables created in Airlines. ‘flightForm’ table is used in this Flight Management System

```
MariaDB [Airlines]> describe flightForm;
+-------------------+------------+------+-----+---------+-------+
| Field             | Type       | Null | Key | Default | Extra |
+-------------------+------------+------+-----+---------+-------+
| flyingFrom        | varchar(50) | NO   |     | NULL    |       |
| flyingTo          | varchar(50) | NO   |     | NULL    |       |
| departingOn       | date       | YES  |     | NULL    |       |
| returningOn       | date       | YES  |     | NULL    |       |
| departureTime     | time       | YES  |     | NULL    |       |
| returningTime     | time       | YES  |     | NULL    |       |
| fullName          | varchar(50) | NO   |     | NULL    |       |
| email             | varchar(50) | NO   |     | NULL    |       |
| phone             | int(20)    | NO   |     | NULL    |       |
| nationality       | varchar(50) | NO   |     | NULL    |       |
| gender            | varchar(10) | NO   |     | NULL    |       |
| passportNo        | int(20)    | NO   |     | NULL    |       |
| passportExpiryDate | date       | YES  |     | NULL    |       |
+-------------------+------------+------+-----+---------+-------+
13 rows in set (0.541 sec)
```

This shows the data types used in Flight Management System

flyingFrom	flyingTo	departingOn	returningOn	departureTime	returningTime	fullName	email	phone	nationality	gender	passportNo	passportExpiryDate
a	a	2020-01-01	2020-01-01	01:00:00	02:00:00	a	a	123	a	a	222	2020-02-02
b	b	2020-02-02	2020-03-30	02:00:00	03:00:00	b	b	123	b	b	333	2025-04-04
Singapore	Singapore	2020-03-28	2020-04-28	08:00:00	12:00:00	Myco Perez	mycoperez@email.com	1234567890	Filipino	Male	12341234	2025-01-01
Philippines	Hong Kong	2020-01-01	2020-02-02	09:00:00	01:00:00	example lang	example@email.com	1234567891	Filipino	Male	12121212	2026-01-01

This shows the list of flights created by the travelers. It shows their destination, time and personal information.

APPENDIX C

Source Code

```
#Flight Management System
import tkinter as tk
from tkinter import *
import mysql.connector
from tkinter import messagebox

root = tk.Tk()
root.geometry('500x400')
root.title('EMP Airlines - Start Page')

connection = mysql.connector.connect(
    host="localhost",
    user="root",
    passwd="",
    database="Airlines")


def FlightForm():

        def ShowStart():
                flight.withdraw()
                root.update()
                root.deiconify()

        def CreateFlight():
                flyingFrom    = entryFrom.get()
                flyingTo      = entryTo.get()
                departingOn = entryDepOn.get()
```

```
                returningOn = entryRetOn.get()
                departureTime = entryDepTime.get()
                returningTime = entryRetTime.get()

                fullName = entryName.get()
                email = entryEmail.get()
                phone = entryPhone.get()
                nationality = entryNtnlty.get()
                gender = entryGndr.get()
                passportNo = entryPsprt.get()
                passportExpiryDate = entryPsprtExp.get()

                cursor = connection.cursor()
                mysql_insert_query = "INSERT INTO flightForm
VALUES(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"

                recordUser =
(flyingFrom,flyingTo,departingOn,returningOn,departureTime,returningTime,fullNam
e,email,phone,nationality,gender,passportNo,passportExpiryDate)
                cursor.execute(mysql_insert_query, recordUser)
                connection.commit()
                ClearFields()
                messagebox.showinfo('EMP Airlines', 'Flight has been created
successfully!')

        def ClearFields():
                entryFrom.delete(0,END)
                entryTo.delete(0,END)
                entryDepOn.delete(0,END)
                entryRetOn.delete(0,END)
```

```
                entryDepTime.delete(0,END)
                entryRetTime.delete(0,END)
                entryName.delete(0,END)
                entryEmail.delete(0,END)
                entryPhone.delete(0,END)
                entryNtnlty.delete(0,END)
                entryGndr.delete(0,END)
                entryPsprt.delete(0,END)
                entryPsprtExp.delete(0,END)



        root.withdraw()
        flight = tk.Toplevel(root)
        flight.geometry('1000x600')
        flight.title('EMP Airlines - Flight Form')

        bckBtn = Button(flight, text='Back', width=10, fg='orange', font=('bold', 15),
command=ShowStart)
        bckBtn.grid(row=0,column=0, padx=3, pady=5)

        frame = LabelFrame(flight, text='Choose a Destination')
        frame.grid(row=1, column=1, padx=20, pady=20)

        lblFrom = Label(frame, text='Flying From :', font=('bold', 15))
        lblFrom.grid(row=2, column=0)
        entryFrom = Entry(frame)
        entryFrom.grid(row=2, column=1, pady=10)

        lblTo = Label(frame, text='Flying To :', font=('bold', 15))
        lblTo.grid(row=3, column=0)
```

```
entryTo = Entry(frame)
entryTo.grid(row=3, column=1, pady=10)

lblDepOn = Label(frame, text='Departing On :', font=('bold', 15))
lblDepOn.grid(row=4, column=0)
entryDepOn = Entry(frame)
entryDepOn.grid(row=4, column=1, pady=10)

lblRetOn = Label(frame, text='Returning On :', font=('bold', 15))
lblRetOn.grid(row=5, column=0)
entryRetOn = Entry(frame)
entryRetOn.grid(row=5, column=1, pady=10)

lblDepTime = Label(frame, text='Departure Time :', font=('bold', 15))
lblDepTime.grid(row=6, column=0)
entryDepTime = Entry(frame)
entryDepTime.grid(row=6, column=1, pady=10)

lblRetTime = Label(frame, text='Returning Time :', font=('bold', 15))
lblRetTime.grid(row=7, column=0)
entryRetTime = Entry(frame)
entryRetTime.grid(row=7, column=1, pady=10)

frame1 = LabelFrame(flight, text='Personal Information')
frame1.grid(row=1, column=2, padx=20, pady=20)

lblName = Label(frame1, text='Full Name :', font=('bold', 15))
lblName.grid(row=2, column=2)
entryName = Entry(frame1)
entryName.grid(row=2, column=3, pady=10)
```

```
lblEmail = Label(frame1, text='Email :', font=('bold', 15))
lblEmail.grid(row=3, column=2)
entryEmail = Entry(frame1)
entryEmail.grid(row=3, column=3, pady=10)

lblPhone = Label(frame1, text='Phone No. :', font=('bold', 15))
lblPhone.grid(row=4, column=2)
entryPhone = Entry(frame1)
entryPhone.grid(row=4, column=3, pady=10)

lblNtnlty = Label(frame1, text='Nationality :', font=('bold', 15))
lblNtnlty.grid(row=5, column=2)
entryNtnlty = Entry(frame1)
entryNtnlty.grid(row=5, column=3, pady=10)

lblGndr = Label(frame1, text='Gender :', font=('bold', 15))
lblGndr.grid(row=6, column=2)
entryGndr = Entry(frame1)
entryGndr.grid(row=6, column=3, pady=10)

lblPsprt = Label(frame1, text='Passport No. :', font=('bold', 15))
lblPsprt.grid(row=7, column=2)
entryPsprt = Entry(frame1)
entryPsprt.grid(row=7, column=3, pady=10)

lblPsprtExp = Label(frame1, text='Passport Expiry Date :', font=('bold', 15))
lblPsprtExp.grid(row=8, column=2)
entryPsprtExp = Entry(frame1)
entryPsprtExp.grid(row=8, column=3, pady=10)
```

```
        createBtn = Button(flight, text='Create Flight', width=40, font=('bold', 20),
fg='blue', command=CreateFlight)
        createBtn.grid(row=9, column=1, columnspan=5, pady=15)

#Sign In
def Admin():
        lblTitle = Label(root, text='Sign in to show list of flights', font=('bold', 14))
        lblTitle.grid(row=3, column=0)
        frame3 = LabelFrame(root, text='Admin')
        frame3.grid(row=4, column=0)

        lblUname = Label(frame3, text='Username :', font=('bold', 12))
        lblUname.grid(row=4, column=0)
        entryUname = Entry(frame3)
        entryUname.grid(row=4, column=1)
        entryUname.insert(0, 'admin')

        lblPass = Label(frame3, text='Password :', font=('bold', 12))
        lblPass.grid(row=5, column=0)
        entryPass = Entry(frame3, show='*')
        entryPass.grid(row=5, column=1, pady=5)
        entryPass.insert(0, 'admin.admin')

        signInBtn = Button(frame3, width=10,text='Sign In', fg='orange', font=('bold',
12), command=FlightList)
        signInBtn.grid(row=6, column=1)

#Show List of Flights
def FlightList():
```

```
        root.withdraw()
        flightlist = tk.Toplevel(root)
        flightlist.geometry('1500x500')
        flightlist.title('List of Flights')

        cursor = connection.cursor()
        mysql_select_query = "SELECT flyingFrom, flyingTo, departingOn,
returningOn, departureTime, returningTime, fullName, email, phone, nationality,
gender, passportNo, passportExpiryDate FROM flightForm"
        cursor.execute(mysql_select_query)
        result = cursor.fetchall()
        cursor.close()
        space=50

        def ShowStart():
                flightlist.withdraw()
                root.update()
                root.deiconify()

        bckBtn = Button(flightlist, text='Back', width=10, fg='orange', font=('bold', 15),
command=ShowStart)
        bckBtn.grid(row=0,column=0, padx=3, pady=5)

        lblFlyingFrom = Label(flightlist, text='Flying\nFrom', font=('bold', 15))
        lblFlyingFrom.grid(row=1, column=0, pady=10)
        lblFlyingTo = Label(flightlist, text='Flying\nTo', font=('bold', 15))
        lblFlyingTo.grid(row=1, column=1, padx=10, pady=10)
        lblDepartingOn = Label(flightlist, text='Departing\nOn', font=('bold', 15))
        lblDepartingOn.grid(row=1, column=2, padx=10, pady=10)
        lblReturningOn = Label(flightlist, text='Returning\n On', font=('bold', 15))
```

```
        lblReturningOn.grid(row=1, column=3, padx=10, pady=10)
        lblDepartureTime = Label(flightlist, text='Departure\nTime', font=('bold', 15))
        lblDepartureTime.grid(row=1, column=4, padx=10, pady=10)
        lblReturningTime = Label(flightlist, text='Returning\nTime', font=('bold', 15))
        lblReturningTime.grid(row=1, column=5, padx=10, pady=10)
        lblFullName = Label(flightlist, text='Full\nName', font=('bold', 15))
        lblFullName.grid(row=1, column=6, padx=20, pady=10)
        lblEmail = Label(flightlist, text='Email', font=('bold', 15))
        lblEmail.grid(row=1, column=7, padx=10, pady=10)
        lblPhone = Label(flightlist, text='Phone', font=('bold', 15))
        lblPhone.grid(row=1, column=8, padx=20, pady=10)
        lblNationality = Label(flightlist, text='Nation-\nality', font=('bold', 15))
        lblNationality.grid(row=1, column=9, padx=10, pady=10)
        lblGender = Label(flightlist, text='Gender', font=('bold', 15))
        lblGender.grid(row=1, column=10, padx=10)
        lblPassport = Label(flightlist, text='Passport\nNo', font=('bold', 15))
        lblPassport.grid(row=1, column=11, padx=10, pady=10)
        lblPassportExp = Label(flightlist, text='Passport\nExpiry Date', font=('bold',
15))
        lblPassportExp.grid(row=1, column=12, padx=10, pady=10)

        for a in result:
                flight_list = Label(flightlist, text=a[0], font=10)
                flight_list.grid(row=2+space, column=0)

                flight_list = Label(flightlist, text=a[1], font=10)
                flight_list.grid(row=2+space, column=1)

                flight_list = Label(flightlist, text=a[2], font=10)
                flight_list.grid(row=2+space, column=2)
```

```
flight_list = Label(flightlist, text=a[3], font=10)
flight_list.grid(row=2+space, column=3)

flight_list = Label(flightlist, text=a[4], font=10)
flight_list.grid(row=2+space, column=4)

flight_list = Label(flightlist, text=a[5], font=10)
flight_list.grid(row=2+space, column=5)

flight_list = Label(flightlist, text=a[6], font=10)
flight_list.grid(row=2+space, column=6)

flight_list = Label(flightlist, text=a[7], font=10)
flight_list.grid(row=2+space, column=7)

flight_list = Label(flightlist, text=a[8], font=10)
flight_list.grid(row=2+space, column=8)

flight_list = Label(flightlist, text=a[9], font=10)
flight_list.grid(row=2+space, column=9)

flight_list = Label(flightlist, text=a[10], font=10)
flight_list.grid(row=2+space, column=10)

flight_list = Label(flightlist, text=a[11], font=10)
flight_list.grid(row=2+space, column=11)

flight_list = Label(flightlist, text=a[12], font=10)
flight_list.grid(row=2+space, column=12)
```

```
                space = space + 50



#StartPage
lbl = tk.Label(root, text='EMP AIRLINES', font=('bold', 30))
lbl.grid(row=0, column=0, padx=150, pady=10)

startBtn = tk.Button(root, text='START', width=10, font=('bold', 16), fg = 'blue',
command=FlightForm)
startBtn.grid(row=1, column=0, pady=10)

listBtn = tk.Button(root, text='Show List of Flights', width=20, font=('bold', 12), fg =
'green', command=Admin)
listBtn.grid(row=2, column=0, pady=10)
```