FLIGHT MANAGEMENT SYSTEM

BY - KEERTHIVASAN . R

OF CLASS 12 E

Chettinad Vidyashram

(Affiliated to Central Board of Secondary Education, New Delhi)
(Chettinad House, R.A.Puram, Chennai – 600 028)

COMPUTER SCIENCE

Certified to be the Bonafide Record of work done by _____ of Std XII Sec ____ in the Computer Science Lab of the CHETTINAD VIDYASHRAM, CHENNAI, during the year 2021 – 2022.

Date: Teacher-in-charge

REGISTER NO. _____

Submitted for All India Senior Secondary Practical Examination in Computer Science held on _____at Chettinad Vidyashram, Chennai – 600 028.

Principal Internal Examiner External Examiner

ACKNOWLEDGEMENT

I would like to express my sincere thanks to Meena Aunty, Principal Mrs. S. Amudha Lakshmi for their encouragement and support to work on this Project. I am grateful to my computer science teacher Mrs. Kalpana G and to the Computer Science department for the constant guidance and support to complete the project.

TABLE OF CONTENTS

INTRODUCTION TO PYTHON

Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming .Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management.It also features dynamic name resolution (late binding), which binds method and variable names during program execution.

PROJECT DESCRIPTION

The Project developed is a flight management system, which is a useful tool that can be used to display information about the details of a flight in a public display monitor. This program can be remotely controlled by the Supervisor to change status, estimated arrival time of different flights hassle free. This project has been developed keeping in mind the systems used in airports and other transportation stations and the challenges faced by the people.

FUNCTIONS USED

Def reminder()
Def delete_users_account()
Def delete_users_back()
Def user_management()
Def adding_user()
Def adding_admin()
Def admin_username()
Def display_users()
Def update()
Def cancel()
Def viewing_flights()

SOURCE CODE

```
import tkinter as tk
import datetime

admin = {'admin': 'admin'}
manager = {'manager': 'manager'}
standard = {'standard': 'standard'}
scheduled = {'PH104': ["10:10", 'Manila', "Delayed"], 'NH874': ['5:30',
'Tokyo', 'Delayed'],
            'LH8472': ['1:49', 'Frankfurt', 'Scheduled'], 'AI672': ['7:40',
'Dehli', 'Scheduled'],
            '6E783': ['2:50', 'Bengaluru', 'Delayed'], 'BA456': ['9:320',
'London', 'Scheduled'],
            'AF217': ['8:40', 'New York', 'Scheduled']}
cancelled = {'SQ789': ["3:00", "Singapore", "Cancelled"]}


def reminder():
    time_clock = datetime.datetime.now()
    for x in scheduled:
        c = scheduled[x][0].split(":")
        time_hrs = time_clock.replace(hour=int(c[0]), minute=int(c[1]),
second=0, microsecond=0)
        if time_clock > time_hrs:
            scheduled_reminders = tk.Tk()
            scheduled_reminders.title("Reminder!")
            tk.Label(master=scheduled_reminders, text="Flight Number " +
x + " Not Updated").grid(row=0, column=0)
            scheduled_reminders.mainloop()
```

```
def delete_users_account():
    def delete_users_back():
        user = username.get()
        if not (user in admin or user in manager or user in cancelled):
            root = tk.Tk()
            root.title("Username Not Found!")
            tk.Label(master=root, text="Username not found! Please try
again!").grid(row=1, column=1)

        else:
            if username.get() == 'admin':
                a = tk.Tk()
                a.title("Can't Delete user!")
                tk.Label(master=a, text="Cannot Delete user!").grid(row=1,
column=1)
                a.mainloop()
            elif user in admin:
                if user != 'admin':
                    del admin[user]
                    ad = tk.Tk()
                    ad.title("Success!")
                    tk.Label(master=ad, text="Admin User Successfully
Deleted!").grid(row=1, column=1)

            elif user in manager:
                del manager[user]
                ad = tk.Tk()
                ad.title("Success!")
                tk.Label(master=ad, text=" Manager Successfully
Deleted!").grid(row=1, column=1)
            if user in standard:
                del standard[user]
```

```
            ad = tk.Tk()
            ad.title("Success!")
            tk.Label(master=ad, text="Standard User Successfully
Deleted!").grid(row=1, column=1)

    users_delete_acc = tk.Tk()
    users_delete_acc.title("Delete A User")
    tk.Label(master=users_delete_acc, text="Enter the
Username").grid(row=1, column=0)
    username = tk.Entry(master=users_delete_acc)
    username.grid(row=1, column=1)
    admin2 = tk.Button(master=users_delete_acc, text="Confirm
Deletion", width=25, command=delete_users_back).grid(
        row=1, column=2)


def user_management():
    def adding_user():
        def adding_admin():
            def admin_username():
                def admin_username_back():
                    password = passwrd.get()
                    admin[user] = password
                    root = tk.Tk()
                    root.title("Success")
                    tk.Label(master=root, text="Admin Successfully
Added!").grid(row=1, column=1)

                user = username.get()
                if user in admin or user in manager or user in standard:
                    root = tk.Tk()
                    root.title("Username Already Exists!")
```

```
                tk.Label(master=root, text="Username Already Exists!
Please Try Again").grid(row=1, column=1)
                manage_user_scheduled.destroy()
                system.destroy()
            else:
                tk.Label(master=system, text="Enter the
Password").grid(row=2, column=0)
                passwrd = tk.Entry(master=system, show='*')
                passwrd.grid(row=2, column=1)
                tk.Button(master=system, text="Confirm Password",
command=admin_username_back, width=25).grid(row=2,

column=3)

        system = tk.Tk()
        system.title("Add An Admin")
        tk.Label(master=system, text="Enter the
Username").grid(row=1, column=0)
        username = tk.Entry(master=system)
        username.grid(row=1, column=1)
        admin3 = tk.Button(master=system, text="Confirm Username",
width=25, command=admin_username).grid(row=1,

column=3)

    def add_manager():
        def manager_username():
            def manager_users_back():
                password = passwrd.get()
                manager[user] = password
                root = tk.Tk()
                root.title("Success")
```

```
                tk.Label(master=root, text="Manager Successfully
Added!").grid(row=1, column=1)

            user = username.get()
            if user in admin3 or user in manager or user in standard:
                root = tk.Tk()
                root.title("Username Already Exists!")
                tk.Label(master=root, text="Username Already Exists!
Please Try Again").grid(row=1, column=1)
                manage_user_scheduled.destroy()
                system.destroy()
            else:
                tk.Label(master=system, text="Enter the
Password").grid(row=2, column=0)
                passwrd = tk.Entry(master=system, show='*')
                passwrd.grid(row=2, column=1)
                tk.Button(master=system, text="Confirm Password",
command=manager_users_back, width=25).grid(row=2,

column=3)

        system = tk.Tk()
        system.title("Add A Manager")
        tk.Label(master=system, text="Enter the
Username").grid(row=1, column=0)
        username = tk.Entry(master=system)
        username.grid(row=1, column=1)
        admin3 = tk.Button(master=system, text="Confirm Username",
width=25, command=manager_username).grid(row=1,

column=3)

    def adding_standard():
```

```
        def standard_username():
            def standard_back_users():
                password = passwrd.get()
                standard[user] = password
                root = tk.Tk()
                root.title("Success")
                tk.Label(master=root, text="Standard User Successfully
Added!").grid(row=1, column=1)

            user = username.get()
            if user in admin or user in manager or user in standard:
                root = tk.Tk()
                root.title("Username Already Exists!")
                tk.Label(master=root, text="Username Already Exists!
Please Try Again").grid(row=1, column=1)

                system.destroy()
            else:
                tk.Label(master=system, text="Enter the
Password").grid(row=2, column=0)
                passwrd = tk.Entry(master=system, show='*')
                passwrd.grid(row=2, column=1)
                tk.Button(master=system, text="Confirm Password",
command=standard_back_users, width=25).grid(row=2,

column=3)

        system = tk.Tk()
        system.title("Add A Standard User")
        tk.Label(master=system, text="Enter the
Username").grid(row=1, column=0)
        username = tk.Entry(master=system)
        username.grid(row=1, column=1)
```

```
        admin3 = tk.Button(master=system, text="Confirm Username",
width=25, command=standard_username).grid(row=1,

column=3)

    adding_users_scheduled = tk.Tk()
    adding_users_scheduled.title("Add A User")
    tk.Button(master=adding_users_scheduled, width=25, text="Add
An Admin", command=adding_admin).grid(row=1,

column=1)
    tk.Button(master=adding_users_scheduled, width=25, text="Add
A Supervisor", command=add_manager).grid(row=2,

column=1)
    tk.Button(master=adding_users_scheduled, width=25, text="Add
A Standard User", command=adding_standard).grid(
        row=3, column=1)

  def display_users():
    display_scheduled = tk.Tk()
    display_scheduled.title("View Users")
    tk.Label(master=display_scheduled, text="Admin
Are:").grid(row=0, column=0)
    e = 0
    for a in admin:
        e += 1
        tk.Label(master=display_scheduled, text=("------",
a)).grid(row=e, column=0)
    e += 1
    tk.Label(master=display_scheduled, text="Manager
Are:").grid(row=e, column=0)
    for a in manager:
```

```
            e += 1
            tk.Label(master=display_scheduled, text=("-----",
a)).grid(row=e, column=0)
        e += 1
        tk.Label(master=display_scheduled, text="Standard Users
Are:").grid(row=e, column=0)
        for a in standard:
            e += 1
            tk.Label(master=display_scheduled, text=("-----",
a)).grid(row=e, column=0)

    def deleting_users():
        def deleting_back_user():
            user = username.get()
            if not (user in admin or user in manager or user in standard):
                root = tk.Tk()
                root.title("Username Not Found!")
                tk.Label(master=root, text="Username not found! Please try
again!").grid(row=1, column=1)
            else:
                if user in admin:
                    del admin[user]
                    admin1 = tk.Tk()
                    admin1.title("Success!")
                    tk.Label(master=admin1, text="Admin User Successfully
Deleted!").grid(row=1, column=1)
                    delete_users_scheduled.destroy()
                elif user in manager:
                    del manager[user]
                    admin1 = tk.Tk()
                    admin1.title("Success!")
                    tk.Label(master=admin1, text=" Manager Successfully
Deleted!").grid(row=1, column=1)
```

```
                delete_users_scheduled.destroy()
            if user in standard:
                del standard[user]
                admin1 = tk.Tk()
                admin1.title("Success!")
                tk.Label(master=admin1, text="Standard User
Successfully Deleted!").grid(row=1, column=1)
                delete_users_scheduled.destroy()

        delete_users_scheduled = tk.Tk()
        delete_users_scheduled.title("Delete A User")
        tk.Label(master=delete_users_scheduled, text="Enter the
Username").grid(row=1, column=0)
        username = tk.Entry(master=delete_users_scheduled)
        username.grid(row=1, column=1)
        admin2 = tk.Button(master=delete_users_scheduled,
text="Confirm Deletion", width=25,
                           command=deleting_back_user).grid(row=1,
column=2)

    manage_user_scheduled = tk.Tk()
    manage_user_scheduled.title("Manage Users")
    tk.Button(master=manage_user_scheduled, text="Add Users",
width=25, command=adding_user).grid(row=1, column=0)
    tk.Button(master=manage_user_scheduled, text="Delete A User",
width=25, command=deleting_users).grid(row=2,

column=0)
    tk.Button(master=manage_user_scheduled, text="View Current
Users", width=25, command=display_users).grid(row=3,

column=0)
```

```
def update():
    scheduled_update = tk.Tk()
    scheduled_update.title("Update/Add A Flight")
    tk.Label(master=scheduled_update, text="Enter The Flight
Number").grid(row=1, column=0)
    number_of_flight = tk.Entry(master=scheduled_update)
    number_of_flight.grid(row=1, column=1)

    def update_btn1():
        if number_of_flight.get() not in scheduled:
            scheduled[number_of_flight.get()] = ["", "", ""]
        tk.Label(master=scheduled_update, text="Enter Departure
Time").grid(row=3, column=0)
        departure_time = tk.Entry(master=scheduled_update)
        departure_time.grid(row=3, column=1)
        tk.Label(master=scheduled_update, text="Enter
Status").grid(row=4, column=0)
        stat = tk.Entry(master=scheduled_update)
        stat.grid(row=4, column=1)
        tk.Label(master=scheduled_update, text="Enter
Destination").grid(row=5, column=0)
        destination_place = tk.Entry(master=scheduled_update)
        destination_place.grid(row=5, column=1)

        def update_btn2():
            if departure_time.get() != "":
                scheduled[number_of_flight.get()][0] = departure_time.get()
            if stat.get() != "":
                scheduled[number_of_flight.get()][2] = stat.get()
            if destination_place.get() != "":
                scheduled[number_of_flight.get()][1] =
destination_place.get()
```

```
            update_root = tk.Tk()
            update_root.title("Successfully Updated!")
            tk.Label(master=update_root, text="Successfully
Updated!!").grid(row=0, column=0)
            scheduled_update.destroy()

        tk.Button(master=scheduled_update, text="Confirm",
command=update_btn2).grid(row=6, column=1)

    tk.Button(master=scheduled_update, text="Confirm",
command=update_btn1).grid(row=2, column=1)
    scheduled_update.mainloop()


def cancel():
    def back_cancelled():
        flight = flight_number.get()
        cancelled_scheduled.destroy()
        if flight in cancelled:
            root = tk.Tk()
            root.title("Already Cancelled!")
            tk.Label(master=root, text="Flight Already Cancelled. Please
try again!").grid(row=1, column=1)
            cancel()
        elif not (flight in scheduled):
            root = tk.Tk()
            root.title("Flight Not Found!")
            tk.Label(master=root, text="Flight Not Found. Please try
again!").grid(row=1, column=1)
            cancel()
        else:
            flight1 = scheduled.pop(flight)
            del flight1[2]
```

```
            flight1.append("Cancelled")
            cancelled[flight] = flight1
            root = tk.Tk()
            root.title("Success!")
            tk.Label(master=root, text="Flight Successfully
Cancelled!").grid(row=1, column=1)

    cancelled_scheduled = tk.Tk()
    cancelled_scheduled.title("Cancel a Flight")
    tk.Label(master=cancelled_scheduled, text="Enter the flight
number").grid(row=1, column=0)
    flight_number = tk.Entry(master=cancelled_scheduled)
    flight_number.grid(row=1, column=1)
    admin7 = tk.Button(master=cancelled_scheduled, width=25,
text="Confirm", command=back_cancelled).grid(row=2,

column=1)



def admin_main_features():
    def switch_users_admin():
        admin_main_scheduled.destroy()
        login()

    admin_main_scheduled = tk.Tk()
    admin_main_scheduled.title("Admin Control Panel")
    tk.Button(master=admin_main_scheduled, text="View The Details
Of Flights", command=viewing_flights).grid(row=1,

column=1)
    tk.Button(master=admin_main_scheduled, text="Switch User",
command=switch_users_admin).grid(row=2, column=1)
```

```
    tk.Button(master=admin_main_scheduled, text="Cancel A Flight",
command=cancel).grid(row=4, column=1)
    tk.Button(master=admin_main_scheduled, text="Manage Users",
command=user_management).grid(row=5, column=1)
    tk.Button(master=admin_main_scheduled, text="Exit The
Program", command=exit).grid(row=6, column=1)
    tk.Button(master=admin_main_scheduled, text="Update/Add A
Flight", command=update).grid(row=3, column=1)


def manager_main():
    def switch_users_manager():
        manager_main_scheduled.destroy()
        login()

    manager_main_scheduled = tk.Tk()
    image = tk.PhotoImage(file="icon.png")
    ask = tk.Label(master=manager_main_scheduled,
image=image).grid(row=0, column=0)
    manager_main_scheduled.title("Supervisor Control Panel")
    tk.Label(master=manager_main_scheduled, text="").grid(row=6,
column=0)
    tk.Button(master=manager_main_scheduled, text="View The
Details Of Flights", command=viewing_flights).grid(row=1,

column=1)
    tk.Button(master=manager_main_scheduled, text="Switch User",
command=switch_users_manager).grid(row=2, column=1)
    tk.Button(master=manager_main_scheduled, text="Cancel A
Flight", command=cancel).grid(row=4, column=1)
    tk.Button(master=manager_main_scheduled, text="Exit The
Program", command=exit).grid(row=5, column=1)
```

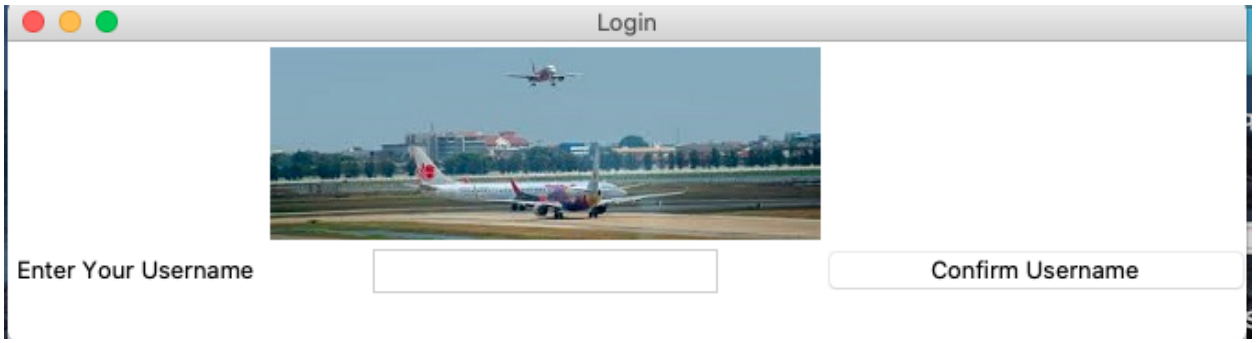
```
    tk.Button(master=manager_main_scheduled, text="Update/Add A
Flight", command=update).grid(row=3, column=1)
    manager_main_scheduled.mainloop()


def viewing_flights():
    can = 0
    ret = 1
    display = tk.Tk()
    display.title("View Details Of Flights")
    tk.Label(master=display, text="Flight
Number--------ETA--------Destination-------Status").grid(row=1,
column=0)
    for i in scheduled:
        can += 1
        ret += 1
        tk.Label(master=display,
                 text=(i, "-------", scheduled[i][0], "--------", scheduled[i][1],
"--------", scheduled[i][2])).grid(
            row=ret, column=0)
    for i in cancelled:
        can += 1
        ret += 1
        tk.Label(master=display,
                 text=(i, "-------", cancelled[i][0], "--------", cancelled[i][1],
"--------", cancelled[i][2])).grid(
            row=ret, column=0)


def main_standard():
    def switch_user_standard():
        main_standard_scheduled.destroy()
        login()
```

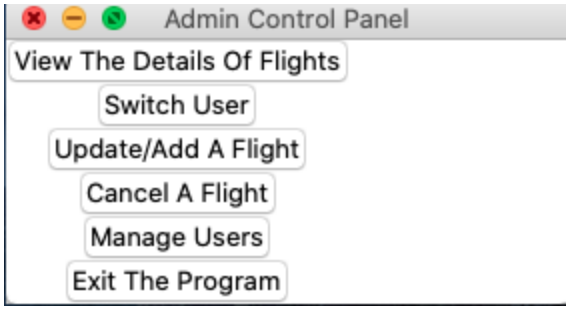
```
    main_standard_scheduled = tk.Tk()
    main_standard_scheduled.title("Standard User Control Panel")
    tk.Button(master=main_standard_scheduled, text="View The
Details Of Flights", command=viewing_flights).grid(row=2,

column=0)
    tk.Button(master=main_standard_scheduled, text="Switch User",
command=switch_user_standard).grid(row=3, column=0)
    tk.Button(master=main_standard_scheduled, text="Exit The
Program", command=exit).grid(row=4, column=0)
    tk.Label(master=main_standard_scheduled, text="").grid(row=5,
column=0)
    img = tk.PhotoImage(file="icon.png")
    no = tk.Label(master=main_standard_scheduled,
image=img).grid(row=1, column=0)
    main_standard_scheduled.mainloop()


def login():
    def user_verification():
        def password_verification():
            pas = password.get()
            if check == 1:
                if admin[a] == pas:
                    login_sched.destroy()
                    admin_main_features()
                    reminder()
                else:
                    admin3 = tk.Tk()
                    admin3.title("Wrong Password!")
                    tk.Label(master=admin3, text="Wrong Password! Please
Try Again!").grid(row=1, column=1)
```

```
        elif check == 2:
            if manager[a] == pas:
                login_sched.destroy()
                manager_main()
                reminder()
            else:
                admin3 = tk.Tk()
                admin3.title("Wrong Password!")
                tk.Label(master=admin3, text="Wrong Password! Please
Try Again!").grid(row=1, column=1)
        elif check == 3:
            if standard[a] == pas:
                login_sched.destroy()
                main_standard()

            else:
                admin3 = tk.Tk()
                admin3.title("Wrong Password!")
                tk.Label(master=admin3, text="Wrong Password! Please
Try Again!").grid(row=1, column=1)

    a = username.get()

    if not (a in admin or a in manager or a in standard):
        admin1 = tk.Tk()
        admin1.title("Wrong Username!")
        tk.Label(master=admin1, text="Username Not Found. Please
Try Again!").grid(row=1, column=1)
    else:
        if a in manager:
            check = 2
        elif a in admin:
```

```
            check = 1
        elif a in standard:
            check = 3

        tk.Label(master=login_sched, text="Enter Your
Password").grid(row=2, column=0)
        password = tk.Entry(master=login_sched, show='*')
        password.grid(row=2, column=1)
        admin3 = tk.Button(master=login_sched, text="Confirm
Password", width=25,
                           command=password_verification).grid(row=2,
column=2)

    login_sched = tk.Tk()
    login_sched.title("Login")
    tk.Label(master=login_sched, text="").grid(row=5, column=1)
    image = tk.PhotoImage(file="icon.png")
    no = tk.Label(master=login_sched, image=image).grid(row=0,
column=1)
    tk.Label(master=login_sched, text="Enter Your
Username").grid(row=1, column=0)
    username = tk.Entry(master=login_sched)
    username.grid(row=1, column=1)
    asking = tk.Button(master=login_sched, text="Confirm Username",
width=25, command=user_verification).grid(row=1,

column=2)
    login_sched.mainloop()


login()
```

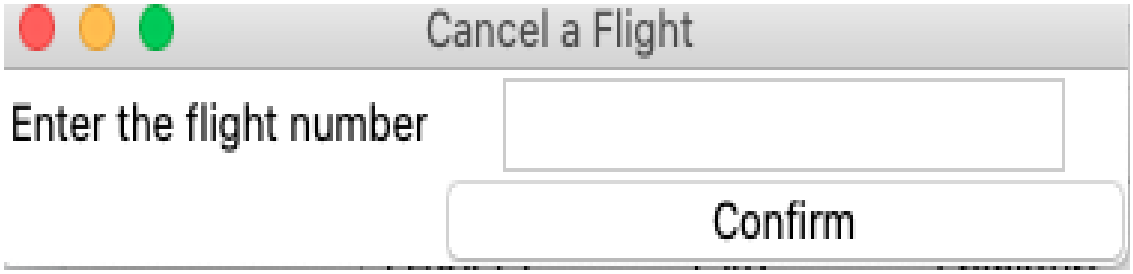
OUTPUT


Login
Enter Your Username
Confirm Username


Admin Control Panel
View The Details Of Flights
Switch User
Update/Add A Flight
Cancel A Flight
Manage Users
Exit The Program


View Details Of Flights
Flight Number--------ETA--------Destination-------Status
PH104 ------- 10:10 -------- Manila -------- Delayed
NH874 ------- 5:30 -------- Tokyo -------- Delayed
LH8472 ------- 1:49 -------- Frankfurt -------- Scheduled
AI672 ------- 7:40 -------- Dehli -------- Scheduled
6E783 ------- 2:50 -------- Bengaluru -------- Delayed
BA456 ------- 9:320 -------- London -------- Scheduled
AF217 ------- 8:40 -------- {New York} -------- Scheduled
SQ789 ------- 3:00 -------- Singapore -------- Cancelled


Update/Add A Flight
Enter The Flight Number
Confirm


Cancel a Flight
Enter the flight number
Confirm


Manage Users
Add Users
Delete A User
View Current Users

CONCLUSION

In conclusion, the above program makes it easier for departing and arriving passengers to easily check information regarding their flights and saves time and effort. It also is a great tool for airports to display information in the most effective manner.

BIBLIOGRAPHY

https://realpython.com/python-gui-tkinter/

https://www.tutorialspoint.com/python/python_gui_programming.htm

https://docs.python.org/3/library/tkinter.html

https://www.google.co.uk/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjPyNfny8fzAhVSmuYKHQgqCbsQwqsBegQIBBAB&url=https%3A%2F%2Fwww.youtube.com%2Fwatch%3Fv%3DftPoGO5AvyM&usg=AOvVaw0R4Waylv_RXUqQ2yPiXO6r

https://www.google.co.uk/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjAhNWXzMfzAhVMWX0KHdu7C7MQtwJ6BAgCEAM&url=https%3A%2F%2Fitsourcecode.com%2Ffree-projects%2Fpython-projects%2Fflight-management-system-in-python-with-source-code%2F&usg=AOvVaw0tdjcAWqtgPhkwYUahsxwZ

----------x----------x----------x----------