



NOVEMBER 19, 2021

[DOCUMENT TITLE]
[DOCUMENT SUBTITLE]

ADMIN2
[COMPANY NAME]
[Company address]




Table of Contents

Introduction & Assumptions.....	2
Design of the program	3
Programming Concepts with Explanation	10
Dataflow & Datatypes	10
Functions	10
Control Structures	11
Exception / Error Handling	12
Additional features source code with explanation	12
Admin authentication	12
Screenshot of Sample input-output.....	13
Conclusion	14
Reference	14

Introduction & Assumptions

ARS stands for Airline Reservation System, and it allows an airline to sell its inventory (seats). It has schedules and rates information, as well as a database of reservations (passenger name records) and tickets issued.

Air Malaysia group (AMG) is a luxury low-cost airline based in Malaysia. It is one of Malaysia's fastest-growing airlines in terms of fleet size and destinations. The Air Malaysia Group has scheduled flights to more than 100 locations in 25 countries. AMG decided to improve their web booking by integrating their most recent flight offer, which includes airline schedules, fare tariffs, passenger reservations, and ticket records.

As a result, AMG asks us to design a Python programme for the ARS using suitable python programming logic and principles, with the required features added in the programme.

Some brief of python programming language – Python is a dynamically semantic, interpreted, object-oriented high-level programming language. Its high-level built-in data structures, together with dynamic typing and dynamic binding, making it ideal for Rapid Application Development and as a scripting or glue language for connecting existing components. Python's concise, easy-to-learn syntax prioritises readability, which lowers software maintenance costs. Modules and packages are supported by Python, which fosters programme modularity and code reuse. For all major platforms, the Python interpreter and vast standard library are available for free in source or binary form, and can be downloaded for free. [1]

What we Developed?

We have developed and designed the python airline booking system with the following features:

1. A Menu screen that directs the users to Registered, Not Registered and admin portal.
2. The Registered Members can log in to the system with their usernames.
3. On successful Authentication of user the user can Display , add or modify their profile.
4. The Non Registered members can display all airline schedules, search any airlines they want and make their account ie, get registered by creating their username.
5. The Members can search the flights with the any of the following inputs
 - I. From
 - II. To
 - III. Date of Departure
 - IV. Date of Return
6. The admins have all the rights to add the flights, Modify the flight schedule and display them accordingly.
7. Though, to access the admin portal they are required to add validate their username and password.

Design of the program

Pseudo code:

Function start ()

display ("welcome to AIRLINE RESERVATIONS SYSTEM (ARS)")

display ("1) Registered")

display ("2) Not Registered.")

display ("3) Admin")

display ("4) Exit")

declare option

prompt user for option

read option

if option is equal to 1, then

return function registered()

if option is equal to 2, then

return function not_registered()

if option is equal to 3, then

return function login_admin()

if option is equal to 4, then

return function exit()

else

display("Enter a valid choice")

ENDIF

ENDIF

ENDIF

END function start()

Function registered ()

 Display(Registered Membership)

 Display(1-login)

 Display(r-return)

 Declare option

 prompt user for option

 read option

 if option is equal to 1, then

 return function log_in()

 if option is equal to 2, then

 return function exit()

 else

 Display ("Enter a valid choice")

 ENDIF

 ENDIF

END registered()

Function Log_in()

 Input username

 If input matches the database then

 Display ("Login successful")

 Return Registered_options ()

 Else

 Display ("Invalid Credentials")

 ENDIF

END Log_in()

Function registered_options(user_name,users)

 Display("Registered Membership")

 Display("1) Display")

```

Display("2) Add")
Display("3) Modify")
Display("4) Log out")
declare option
prompt user for option
read option
if option is equal to 1, then
    return function registered_display(user_names, users)
    if option is equal to 2, then
        return function registered_add(user_name)
        if option is equal to 3, then
            return function registered_modify(usr_name,u)
            if option is equal to 4, then
                return function exit()
            else
                display("Enter a valid choice")
        endif
    endif
endif
endif
endif
END function registered_options()

Function registered_display(user_name,user)
    Display ("Display")
    Display ("1) My profiles")
    Display ("2) My bookings")
    Display ("r) return")
    declare option
    prompt user for option
    read option
    if option is equal to 1, then

```

```

        Display (Profiles)
            if option is equal to 2, then
                Display(Bookings)
                    if option is equal to r, then
                        return function Exit()
                    ENDIF
                ENDIF
            ENDIF
        ENDIF
    END function registered_display()

```

```

Function registered_add(user_name)
    Display("Add")
    Display("1) Flight")
    Display("r) Return")
    declare option
    prompt user for option
    read option
    if option is equal to 1, then
        ADD flight details to database
        if option is equal to r, then
            return function Exit()
        ENDIF
    ENDIF
END registered_add()

```

```

Function Registered_modify(user_name,users)
    Display("Modify")
    Display("1) My accounts")
    Display("r) Return")
    declare option

```

```

    prompt user for option
    read option
    if option is equal to 1, then
        Update profile
        if option is equal to r, then
            return function Exit()
        ENDIF
    ENDIF
END registered_modify()

```

```

Funtion not_registered()
    Display("Un Registered Membership")
    Display("(1) Display all airlines schedule")
    Display("(2) Search Airlines")
    Display("(3) Register")
    Display("(4) Return")
    declare option
    prompt user for option
    read option
    if option is equal to 1, then
        Display(schedule)
        if option is equal to 2, then
            search(airline)
            if option is equal to 3, then
                return function registered()
            if option is equal to 4, then
                return function exit()
            else
                display("Enter a valid choice")
            ENDIF
        ENDIF
    ENDIF

```



```

                                ENDIF
                        ENDIF
END function not_registered()

Function admin()
    Display("Admin")
    Display("1) Add flight schedules")
    Display("2) Modify flight schedule")
    Display("3) display")
    Display("4) Return")
    declare option
    prompt user for option
    read option
    if option is equal to 1, then
        Add (flights)
        if option is equal to 2, then
            update(flights)
            if option is equal to 3, then
                display(flights)
                if option is equal to 4, then
                    return function exit()
                else
                    display("Enter a valid choice")
                endif
            endif
        endif
    endif
endif
END function admin()

Function display_flights()

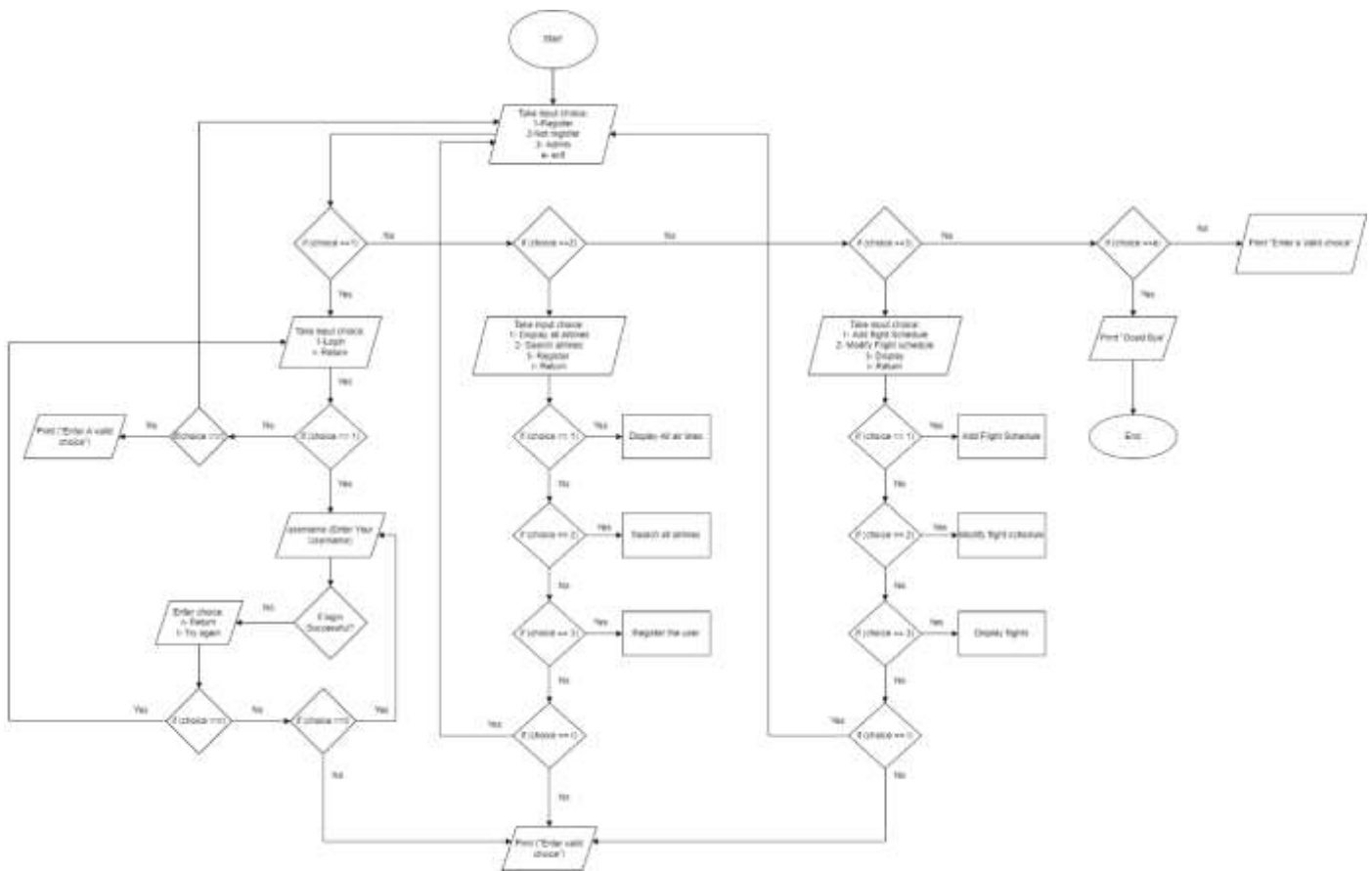
```

```

Display("Admin Display")
Display("1) Flight schedules by flight numebr")
Display("2) flight book by customer")
Display("3) Total tickets")
Display("4) Return")
declare option
prompt user for option
read option
if option is equal to 1, then
    Display(flight number)
        if option is equal to 2, then
            display (customer)
                if option is equal to 3, then
                    display (tickets)
                        if option is equal to 4, then
                            return function exit()
                        else
                            display("Enter a valid choice")
                        endif
                    endif
                endif
            endif
        endif
    endif
endif
endif
END function display_flights()

```

Flow Chart:



Programming Concepts with Explanation

Dataflow & Datatypes

The data type is a property of data that tells the compiler how the programmer intends to utilise that particular piece of data. The operations that can be performed on the data are also defined by the data type. A data type specifies the sort of value that an object can have and the actions that can be performed on it.

In this project we have widely used Integers and strings.

Functions

Functions are very useful for programming. When we wish to perform a particular task repeatedly we use the concept of functions. Instead of defining the task again and again we simply call the function.

For example some of the function we used in are:

Start (), Registered (), login (), admin (), not registered (), etc.

Functions like split (), find (), strip (), append () are also used. These are in built functions.

Control Structures

Sequence – In this type of structure the defined statements are executed one after the other in the same order as they are defined.

Conditional - This type of control is used when we want to take alternative parts based on the conditions given. In other words it is similar to if this then that.

Iterative – This control structure is used to repeat a particular condition until a particular condition is met. In other words this is known as looping.

For example:

```
def start():
    print('welcome to AIRLINE RESERVATIONS SYSTEM (ARS)\n')

    while True:
        print("\nPlease choose your membership")
        print(" [ 1 ] : Registered")
        print(" [ 2 ] : Not Registered")
        print(" [ 3 ] : Admin")
        print(" [ e ] : Exit")
        inp = input("\nChoice: ").replace(' ', '')
        #try:
        if inp in ["e", "r", ""]:
            print("\nGood Bye!")
            break

        elif inp == "1":
            registered()
        elif inp == "2":
            not_registered()
        elif inp == "3":
            admin()
        else:
            print("\nPlease enter a valid choice:")
```

In the above example the code starting from while loop is an example of Iterative, ie it keeps printing the menu until the condition becomes false. Where as in the same code if & else if conditions are examples for the conditional statements. If choice then call function registered () else check for value 2 and call the function not register else check for value 3 and other values and call the functions accordingly.

Exception / Error Handling

Exception and Error handling is done to avoid any unexpected issue/ outcome. For example

```
while True:
    #get numeric value
    inp511 = input("\nplease choose the number of what you want to
modify: ").replace(' ','')
    if inp511 in ['', 'r']:
        #c_var = false
        break
    try:
        inp51 = int(inp511)
    except:
        #if not numeric
        inp51 = 0
        print('\nplease enter a valid number')
```

Here we are taking input from the user for their choice if the choice goes beyond the limit/bound it will throw a message to Enter a valid number within the defined bound In the menu.

Additional features source code with explanation

Admin authentication

Added a feature for the admins to authenticate their Identity before they access the crucial data and make some modifications.

Source code:

```
def login_admin():
    while True:
        loogin_in = False
        admins = {}
        with open('admins.txt','r') as ff:
            for line in ff:
                line = line.strip().split(',')
                admins[line[0]] = line[1]

        user_name = input("\nlogging in\n\nPlease Enter Your Username: (leave
empty to return)\n\n").strip()

        if user_name in ['', 'r']:
            break

        elif admins.get(user_name, False):
            while True:
```

```

        password = input("\nPlease Enter Your password: (leave empty
to return)\n\n").strip()
        if password in ['', 'r']:
            break
        elif admins.get(user_name, False) == password.strip():
            print('\nlogged in successfully')
            loogin_in = True
            break
        else:
            print("\npassword incorrect")

    else:
        print("\nUsername {} Invalid".format(user_name))
    if loogin_in:
        admin()

```

The Data of Username and password is saved in admin.txt file. Admin needs to add their username and password to access the admin portal. On successful authentication the admin can access and modify the flight schedules and other functionalities as needed.

Screenshot of Sample input-output

Sample input	Sample Output
<pre> Choice: 1 Registered Membership [1] : Log In [r] : Return Choice: 1 logging in Please Enter Your Username: (leave empty to return) Hanna </pre>	<pre> logged in successfully Registered Membership [1] : Display [2] : Add [3] : Modify [r] : Log out </pre>
<i>Registered member input</i>	<i>Registered member output</i>
<pre> Choice: 3 logging in Please Enter Your Username: (leave empty to return) Hanna Please Enter Your password: (leave empty to return) Hanna123 </pre>	<pre> logged in successfully ADMIN [1] : Add Flight Schedules [2] : Modify Flight Schedules [3] : Display [r] : Return Choice: █ are </pre>

Admin Input	Admin Output
<pre> ADMIN [1] : Add Flight Schedules [2] : Modify Flight Schedules [3] : Display [r] : Return Choice: 3 ADMIN Display [1] : Flight schedules by flight number [2] : Flight booked by customer [3] : Total tickets [r] : Return Choice: 1 </pre>	<pre> Alberto A345 London Cairo 15/11/2021 21/11/2021 RM100 Alberto A348 London Cairo 20/11/2021 21/11/2021 RM400 Adele A347 Uk Australia 19/11/2021 20/11/2021 RM300 Anna A345 London Cairo 16/11/2021 18/11/2021 RM100 Anna A347 Uk Australia 19/11/2021 20/11/2021 RM300 H123 A345 London Cairo 16/11/2021 18/11/2021 RM100 H123 A347 Uk Australia 19/11/2021 20/11/2021 RM300 H123 A348 London Cairo 20/11/2021 21/11/2021 RM400 </pre>
Input for admin display	Total Tickets

Conclusion

After doing this project released that practical learning is the best way of learning. Learning by doing and improving from mistakes provided me a right path and correct learning about the programming and core concepts including the fundamentals of the programming language.

While implementing the project found major difficulties in nested looping and type casting a data type from one form to another. File I/O was completely an interesting topic for me during this project. Further got some issues like while defining the local variables as we were restricted for using the global variables. Found some more difficulties for reading and writing on the file as the format was need to be maintained and undisturbed. However overall it was fun doing this project and implementing it. Got so many new insights to learn. Thanks for giving this opportunity.

Reference

1. Official Documentation of python, *Definition*,
<https://www.python.org/doc/essays/blurb/>
2. Geeks for Geeks to understand the *fundamentals of the inbuilt functions*.