VIT BHOPAL

PYTHON PROJECT

Title: Railway Ticket Booking System

Semester 1

Submitted By : Palak Gupta
Registration No.: 25BAI10870

INTRODUCTION

This is a simple Python based program on the topic -- Railway Ticket Booking System. The program ask the user to enter passenger and journey details, starting and destination stations. And to select a coach from the available coachs. According to the provided details it will give you a formatted ticket.

PROBLEM STATEMENT

Booking a railway ticket usually involves sharing your details, choosing desired coach accordingto your convenience, and knowing how much the ticket will cost.
When done manually, this process can be slow and their are high chances of mistakes
Therefore this project is degined to avoid any mistakes and calculate the fare correctly and give you the formatted ticket.

Functional Requirements

- Inputs by user:

Passenger Name -
Age -
Starting Station -
Destination Station -
Train Number -
Train Name -
Coach Selection from
the options available-

- System Functions:

Displays list of available coach types
Validate coach selection
Calculate fare based on selected coach
Display booking details
Show error message if invalid coach is input

- Outputs:

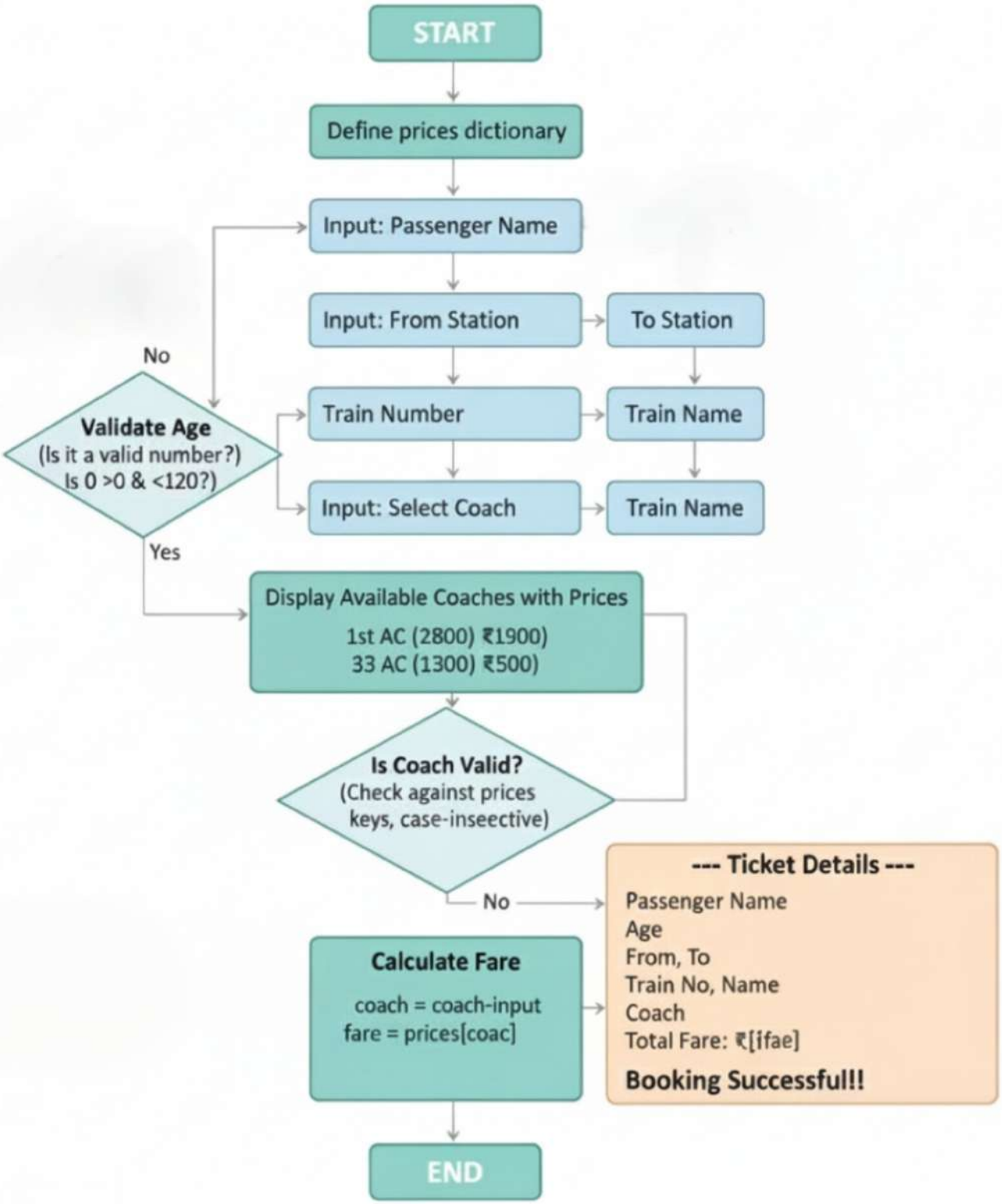
Ticket details
Successful Booking Message

Non-functional Requirements

- Performance
- Scalability
- Security
- Usability
- Reliability
- Maintainability
- Availability
- Compatibility

System Architecture

1. Display welcome message.
2. Collect passenger and travel details.
3. Display list of coach options with prices.
4. Validate the user's selected coach.
5. Calculate the fare.
6. Display the ticket summary.
7. End the process or show an error.


START
Define prices dictionary
Input: Passenger Name
Input: From Station
To Station
No
Validate Age
(Is it a valid number?)
Is 0 >0 & <120?)
Train Number
Train Name
Input: Select Coach
Train Name
Yes
Display Available Coaches with Prices
1st AC (2800) ₹1900)
33 AC (1300) ₹500)
Is Coach Valid?
(Check against prices
keys, case-inseective)
No
--- Ticket Details ---
Passenger Name
Age
From, To
Train No, Name
Coach
Total Fare: ₹[ifae]
Booking Successful!!
Calculate Fare
coach = coach-input
fare = prices[coac]
END

Design Decision Rationale

- Python Dictionary for Coach Pricing:
  A dictionary makes it easy to map coach names to their prices, ensuring quick lookup and clean structure.

- Console-Based Input:
  It keeps the project easy for beginners without involving any GUIs or databases.

- Simple Conditional Validation:
  Ensures learning of basic decision-making in Python.

- Linear Flow:
  Keeps the process easy to follow for new programmers.

Implementation Details

This program is implemented entirely in Python language using:

input() for user interactions

print() for displaying messages and ticket output

A dictionary for storing coach types and fares

A loop to list available coaches

An if condition to validate the coach and calculate fare
The code follows a simple structure, ensuring readability and clarity for beginners.

[1]:

```
# Railway Ticket Booking System

prices = {
    "1st AC": 2800,
    "2nd AC": 1900,
    "3rd AC": 1300,
    "Sleeper": 500
}

print("____Railway Ticket Booking System____\n")

name = input("Enter Passenger Name: ")
age = input("Enter Age: ")
from_station = input("From Station: ")
to_station = input("To Station: ")
train_no = input("Train Number: ")
train_name = input("Train Name: ")

print("\nAvailable Coaches:")
for c in prices:
    print("-", c)

coach = input("\nSelect Coach: ")

if coach in prices:
    fare = prices[coach]

    print("\n----- Ticket Details -----")
    print("Passenger Name:", name)
    print("Age:", age)
```

```python
if coach in prices:
    fare = prices[coach]

    print("\n----- Ticket Details -----")
    print("Passenger Name:", name)
    print("Age:", age)
    print("From:", from_station)
    print("To:", to_station)
    print("Train No:", train_no)
    print("Train Name:", train_name)
    print("Coach:", coach)
    print("Total Fare: ₹", fare)
    print("\nBooking Successful!! ")

else:
    print("\n#Oops Sorry#\nInvalid coach selected, Please try again.")
```

```
____Railway Ticket Booking System____

Enter Passenger Name:  PALAK GUPTA
Enter Age:  19
From Station:  BHOPAL
To Station:  JHANSI
Train Number:  12172
Train Name:  express

Available Coaches:
- 1st AC
- 2nd AC
- 3rd AC
- Sleeper
```

```
else:
    print("\n#Oops Sorry#\nInvalid coach selected, Please try again.")
```

```
____Railway Ticket Booking System____

Enter Passenger Name:  PALAK GUPTA
Enter Age:  19
From Station:  BHOPAL
To Station:  JHANSI
Train Number:  12172
Train Name:  express

Available Coaches:
- 1st AC
- 2nd AC
- 3rd AC
- Sleeper

Select Coach:  1st AC

----- Ticket Details -----
Passenger Name: PALAK GUPTA
Age: 19
From: BHOPAL
To: JHANSI
Train No: 12172
Train Name: express
Coach: 1st AC
Total Fare: ₹ 2800

Booking Successful!!
```

```
    print("Coach: ", coach)
    print("Total Fare: ₹", fare)
    print("\nBooking Successful!! ")

else:
    print("\n#Oops Sorry#\nInvalid coach selected, Please try again.")
```

```
____Railway Ticket Booking System____

Enter Passenger Name:  Ambar Gupta
Enter Age:  18
From Station:  Pune
To Station:  Bhopal
Train Number:  34281
Train Name:  Taj Express

Available Coaches:
- 1st AC
- 2nd AC
- 3rd AC
- Sleeper

Select Coach:  Premium

#Oops Sorry#
Invalid coach selected, Please try again.
```

[ ]:

```
    print("To:", to_station)
    print("Train No:", train_no)
    print("Train Name:", train_name)
    print("Coach:", coach)
    print("Total Fare: ₹", fare)
    print("\nBooking Successful!! ")

else:
    print("\n#Oops Sorry#\nInvalid coach selected, Please try again.")
```

```
____Railway Ticket Booking System____

Enter Passenger Name:  PALAK GUPTA
Enter Age:  19
From Station:  Bhopal
To Station:  Jhansi
Train Number:  12712
Train Name:  Express

Available Coaches:
- 1st AC
- 2nd AC
- 3rd AC
- Sleeper

Select Coach:  1st ac

#Oops Sorry#
Invalid coach selected, Please try again.
```

```
else:
    print("\n#Oops Sorry#\nInvalid coach selected, Please try again.")
```

```
____Railway Ticket Booking System____

Enter Passenger Name:  Palak Gupta
Enter Age:  19
From Station:  Jhansi
To Station:  Bhopal
Train Number:  12178
Train Name:  Punjab Mail

Available Coaches:
- 1st AC
- 2nd AC
- 3rd AC
- Sleeper

Select Coach:  Sleeper

----- Ticket Details -----
Passenger Name: Palak Gupta
Age: 19
From: Jhansi
To: Bhopal
Train No: 12178
Train Name: Punjab Mail
Coach: Sleeper
Total Fare: ₹ 500

Booking Successful!!
```

```
    print("\nOops Sorry!\nInvalid Coach selected, Please try again.")
```

```
____Railway Ticket Booking System____

Enter Passenger Name:  Ambar Gupta
Enter Age:  18
From Station:  Jhansi
To Station:  Mumbai
Train Number:  34187
Train Name:  Pune Express

Available Coaches:
- 1st AC
- 2nd AC
- 3rd AC
- Sleeper

Select Coach:  2nd AC

----- Ticket Details -----
Passenger Name: Ambar Gupta
Age: 18
From: Jhansi
To: Mumbai
Train No: 34187
Train Name: Pune Express
Coach: 2nd AC
Total Fare: ₹ 1900

Booking Successful!!
```

Testing Approach

Follow the following steps
to test the program:

1. Run the script:
The program displays the heading
"Railway Ticket Booking System".

2. Enter valid user inputs, For example:
Passenger name: PALAK GUPTA
Age: 19
From station: Bhopal
To station: Jhansi
Train number: 12712
Train name: Dakshin Express

3. View available coaches:
The program displays on terminal window:
1st AC
2nd AC
3rd AC

4. Select a valid coach
Enter exactly one from the
provided list(case-sensitive).
for example: 3rd AC
(AC should be in capital as
mentioned in the list)

5. Check whether ticket is generated:
If you filled the valid coach,
the trailling options will appear
which will ask the user to enter.

Passenger details
Journey details
Coach selected
Fare
"Booking Successful!!" message

6. Test invalid input
Enter something like ac or Premium.

The program will print:
#Oops Sorry#
Invalid coach selected, Please try again.

Challenges Faced

- ☐ Handling User Input Properly.
- ☐ Validating Coach Selection.
- ☐ Designing a Simple Yet Clear Flow.
- ☐ Managing Data Without a Database.
- ☐ Ensuring Error-Free Execution.
- ☐ Making the Output User-Friendly.
- ☐ Keeping the Code Beginner-Friendly.

Learnings & Key Takeaways

Working on this Railway Ticket Booking System helped me strengthen my Python basics, especially handling user inputs, using dictionaries for data storage, and implementing simple validation checks.
I learned how to structure a small console-based application and present output in a clear, readable format. This project also improved my debugging skills and gave me insight into how real-world booking systems work at a fundamental level.

Future Enhancements

- ☐Add a Graphical User Interface (GUI)
- ☐Include Multiple Passenger Support
- ☐Add Train Database or File Storage
- ☐Implement Seat Availability & Quotas
- ☐Dynamic Fare Calculation
- ☐Add Payment Simulation
- ☐Ticket Cancellation & Modification
- ☐Generate Printable Ticket or PDF
- ☐Error Handling & Input Validation Improvements

References

- Vityatri.
- My Python faculty-
  Dr Mayuri A V R
- Python Official
  Documentation.
- Stackflow.
- General knowledge of
  Railway Booking System.