

AP LAB_3

~ Prof. Ishan Maheta

✓ Title 3: Railway Ticket Reservation System

Objective:

- To implement a Python program that simulates a basic railway ticket reservation system.
 - To load train and passenger data from CSV files, check seat availability, book tickets, update train data, calculate fare, and generate reports.
-

Task Description:

You are tasked with developing a **railway ticket reservation system** for a busy rail network. The system must:

1. Load Train Data from `trains.csv` (Train ID, Train Name, Source, Destination, Total Seats, Distance).
 2. Load Passenger Data from `passengers.csv` (Passenger Name, Train ID, Number of Tickets).
 3. Check Seat Availability and confirm/reject bookings.
 4. Calculate Fare based on distance:
 - ≤ 500 km \rightarrow ₹1.5 per km
 - 501–1000 km \rightarrow ₹1.2 per km
 - 1000 km \rightarrow ₹1.0 per km
 5. Update seat availability after successful bookings.
 6. Generate Reports:
 - **Report 1:** Train details (name, route, available seats).
 - **Report 2:** Revenue summary (successful/failed bookings, fare collected).
 7. Handle errors (invalid Train IDs, insufficient seats, invalid ticket numbers).
-

Python Code:

```
import pandas as pd

TRAINS_FILE = r"D:\Ghanu Study\Sem - 5\Advance
Python\Lab\Code\files\trains.csv"
PASSENGERS_FILE = r"D:\Ghanu Study\Sem - 5\Advance
Python\Lab\Code\files\passengers.csv"
```

```

def fare(distance):
    if distance <= 0:
        return 0
    elif distance <= 500:
        return distance * 1.5
    elif distance <= 1000:
        return distance * 1.2
    else:
        return distance * 1.0

def book(passenger, trains):
    train_id = passenger["Train ID"]
    num_tickets = passenger["Number of Tickets"]
    if num_tickets <= 0:
        return "Failed", 0
    if train_id not in trains["Train ID"].values:
        return "Failed", 0
    train_index = trains[trains["Train ID"] == train_id].index[0]
    available = trains.loc[train_index, "Total Seats"]
    if num_tickets > available:
        return "Failed", 0
    trains.loc[train_index, "Total Seats"] -= num_tickets
    total_fare = num_tickets * trains.loc[train_index, "Single Fare"]
    return "Success", total_fare

def write_reports(trains, passengers):
    with open("report1.txt", "w") as f:
        f.write("=== Train Details ===\n\n")
        for i, train in trains.iterrows():
            f.write(f"{i+1}) {train['Train Name']}\n")
            f.write(f"    Source: {train['Source Station']}\n")
            f.write(f"    Destination: {train['Destination Station']}\n")
            f.write(f"    Seats Available: {train['Total Seats']}\n\n")
    with open("report2.txt", "w") as f:
        f.write("=== Booking & Revenue Report ===\n\n")
        for i, train in trains.iterrows():
            train_id = train["Train ID"]
            success = passengers[(passengers["Train ID"] == train_id) &
(passengers["Status"] == "Success")]
            failed = passengers[(passengers["Train ID"] == train_id) &
(passengers["Status"] == "Failed")]
            f.write(f"{i+1}) {train['Train Name']}\n")
            f.write(f"    Successful Bookings: {len(success)}\n")
            f.write(f"    Failed Bookings: {len(failed)}\n")
            f.write(f"    Total Fare Collected: {success['Fare'].sum()}\n\n")

def main():
    trains = pd.read_csv(TRAINS_FILE)
    passengers = pd.read_csv(PASSENGERS_FILE)
    passengers["Status"] = "Pending"
    passengers["Fare"] = 0

```


```

trains["Single Fare"] = trains["Distance"].apply(fare)
for idx, passenger in passengers.iterrows():
    status, total_fare = book(passenger, trains)
    passengers.at[idx, "Status"] = status
    passengers.at[idx, "Fare"] = total_fare
write_reports(trains, passengers)
print("✅ Reports generated successfully! (report1.txt & report2.txt)")

if __name__ == "__main__":
    main()

```

Sample Input Files:

 trains.csv

```

Train ID,Train Name,Source Station,Destination Station,Total Seats,Distance
T001,Shatabdi Express,Mumbai,Delhi,300,1384
T002,Rajdhani Express,Delhi,Kolkata,350,1530
T003,Duronto Express,Chennai,Bangalore,200,350
T004,Garib Rath,Ahmedabad,Jaipur,400,650
T005,Jan Shatabdi,Pune,Nagpur,250,820

```

 passengers.csv

```

Passenger Name,Train ID,Number of Tickets
Aarav,T001,2
Isha,T002,4
Rohan,T003,1
Meera,T004,3
Vikram,T005,2000
Priya,T001,1
Ananya,T003,4
Siddharth,T004,1
Pooja,T005,3

```

Sample Output (Reports):

```
-----  
Error -> Not enough seats in train T005  
File -> d:\Ghanu Study\Sem - 5\Advance Python\Lab\Code\Chetan Final\Lab-3.py  
Function -> book, Line -> 53  
Code -> raise ValueError(f"Not enough seats in train {train_id}")  
-----
```

```
✓ Reports generated successfully!
```

✓ Conclusion:

This experiment successfully simulates a **railway reservation system** using Python. It demonstrates reading structured data from CSV files, handling bookings with validations, calculating fares dynamically, updating seat availability, and generating professional PDF reports. With error handling and modular design, it efficiently automates essential parts of railway operations.
