**SAMSUNG INNOVATION CAMPUS**

**HACKATHON REPORT**

**INDIGO AIRLINE ROUTES**

**Developed by:**

Kavya Antannavar

20231CSE0126

PRESIDENCY UNIVERSITY, BENGALURU

**Submitted on:**

17 July 2025

**Index**

1. **Problem Statement (Abstract)**
2. **Description (Detailed)**
   * **Data within the Problem**
   * **Relevant Information & Assumptions**
   * **Output of the System**
   * **Purpose and Outcome**
   * **Benefits**
3. **Solution Plan**
4. **Design (Diagrams)**
5. **Implementation**
6. **Code & Explanation**
7. **Output Screenshots**
8. **Closure**
9. **Bibliography**

**1.Problem Statement (Abstract)**

In the highly competitive airline industry, strategic route planning is paramount for profitability and growth. Airlines constantly seek to identify new, underserved, and profitable routes while considering operational constraints, market demand, existing competition, and airport capacity. Manually analyzing vast amounts of data for this purpose is inefficient and prone to human error. This project addresses the challenge of providing a data-driven, automated solution to assist airlines, specifically focusing on Indigo, in identifying and evaluating potential new flight routes.

**2.Description (Detailed)**

The airline industry operates on thin margins, making efficient route network expansion critical. Airlines must balance passenger demand with operational feasibility (aircraft range, airport capacity) and market dynamics (competition). Our project develops a prototype tool that simulates this complex decision-making process, offering data-backed recommendations for new routes.

**Data within the Problem**

The core of our analysis relies on two primary data sources:

* **airports.csv**: This publicly available dataset (sourced from https://www.google.com/search?q=OurAirports.com) provides comprehensive information about airports worldwide. For our project, key fields utilized include:
  + name: Full name of the airport.
  + iata\_code: Unique 3-letter IATA code (e.g., DEL, BOM). Crucial for identification.
  + city (originally municipality): The city or municipality the airport serves.
  + iso\_country: The two-letter ISO country code (e.g., IN, AE).
  + latitude\_deg, longitude\_deg: Geographical coordinates for distance calculations and mapping.
  + type: Categorizes the airport (e.g., large\_airport, medium\_airport).
* **CITY\_DEMAND\_SCORES (Simulated Data)**: A dictionary defined within our application that assigns a numerical "demand score" to various cities. This is a hackathon-specific simplification, representing a proxy for market attractiveness or passenger volume. In a real-world scenario, this would be derived from extensive market research, passenger traffic data, economic indicators, and demographic analysis.

**Relevant Information & Assumptions**

To make the simulation realistic within hackathon constraints, several key parameters and assumptions are defined:

* **Aircraft Range (AIRCRAFT\_MAX\_RANGE\_KM)**: Set to 6000 km, representing the typical operational range of Indigo's A320/A321 fleet. Routes exceeding this distance are deemed unfeasible.
* **Minimum Flight Distance (MIN\_FLIGHT\_DISTANCE\_KM)**: Set to 200 km to exclude very short, unprofitable routes for mainline aircraft.
* **Relevant Countries (RELEVANT\_COUNTRIES)**: A curated list of countries where Indigo currently operates or is likely to expand (e.g., India, Southeast Asia, Middle East, and potential long-haul European destinations).
* **Indigo Hubs (INDIGO\_HUBS\_IATA)**: Key airports identified as Indigo's operational hubs (e.g., Delhi, Mumbai, Bengaluru). New routes are primarily considered originating from these strategic locations.
* **Demand Thresholds (DEMAND\_THRESHOLD\_FOR\_RECOMMENDATION)**: A minimum combined demand score required for a route to be considered viable.
* **Profitability Assumptions**: A simplified model (calculate\_profitability\_score) where revenue is linked to demand and distance, and costs are linked to distance.
* **Competition Assessment**: A simplified model (assess\_competition\_level) that categorizes routes as 'High', 'Medium', or 'Low' competition based on predefined competitive routes and hub-to-hub connections.
* **Airport Capacity Check**: A probabilistic model (check\_airport\_capacity) simulating slot availability at known congested airports.
* **Flight Simulation Parameters**: Assumptions for MIN\_FLIGHTS\_PER\_WEEK, MAX\_FLIGHTS\_PER\_WEEK, BASE\_PRICE\_PER\_KM, and PRICE\_VARIANCE\_PERCENT to generate hypothetical flight schedules and prices.

**Output of the System**

The application provides two main outputs:

1. **Terminal Output**:
   * Confirmation of data loading and filtering.
   * A tabular display of the **top 15 recommended new routes**, sorted by simulated demand and profitability. This table includes details like origin/destination cities and IATA codes, distance, demand score, profitability score, and competition level.
   * An interactive **Command-Line Interface (CLI)** that allows the user to select a specific recommended route to view its simulated flight details (day, time, duration, estimated price). The CLI also provides filtering options by price and day.
2. **Interactive HTML Map (indigo\_new\_routes\_map.html)**:
   * A dynamic Folium map centered on India, showing all relevant airports as markers. Indigo hubs are highlighted with a different color and icon.
   * The top 25 recommended new routes are drawn as red lines on the map.
   * Hovering over a route line displays a tooltip with detailed information about the route (cities, IATA codes, distance, demand, profitability, competition, and a summary of simulated flights).

**3.Purpose and Outcome**

* **Purpose**: To provide a rapid prototyping tool for airline route planners, enabling them to quickly identify and evaluate potential new routes based on a combination of strategic, operational, and market factors. It aims to demonstrate a data-driven approach to network expansion.
* **Outcome**: The tool successfully generates a list of feasible and potentially profitable new routes, visualized on an interactive map, and offers simulated flight details for deeper analysis. This outcome provides a tangible starting point for further detailed planning by an airline.

**Benefits**

* **Data-Driven Decisions**: Moves away from intuition-based route planning to a more analytical approach.
* **Efficiency**: Automates the initial screening of thousands of potential routes, saving significant time and resources.
* **Strategic Insight**: Highlights routes that align with Indigo's hub strategy and considers factors like competition and airport capacity.
* **Visualization**: The interactive map provides a clear and intuitive way to understand the geographical implications of new routes.
* **Simulated Details**: The flight simulation offers a glimpse into potential operational aspects and pricing strategies for new routes.
* **Scalability (Conceptual)**: The modular design allows for future integration of more complex data (e.g., real-time competitor data, fuel prices, historical passenger loads) and advanced algorithms.

**3.Solution Plan**

Our solution follows a structured approach:

1. **Data Acquisition & Preparation**: Obtain airport data and define simulated demand scores. Clean and filter the data to include only relevant airports.
2. **Core Analysis Functions Development**: Implement modular functions for:
   * Calculating geographical distances (Haversine formula).
   * Estimating route demand based on city scores.
   * Simulating route profitability.
   * Assessing competition levels.
   * Checking airport capacity constraints.
   * Generating hypothetical flight schedules.
3. **Route Generation & Evaluation Logic**:
   * Iterate through all relevant airport pairs.
   * Apply operational constraints (aircraft range, minimum distance).
   * Apply business logic using the developed analysis functions (demand, profitability, competition, capacity).
   * Filter and store routes that meet all criteria.
4. **Results Visualization**: Utilize the Folium library to create an interactive geographical map displaying airports and recommended routes with informative tooltips.
5. **Interactive User Interface (CLI)**: Develop a command-line interface to allow users to explore simulated flight details for selected recommended routes.

**4.Design (Diagrams)**

**System Architecture Diagram**

**airports.csv**

**|**

**▼**

**airports\_data.py**

**|**

**▼**

**main\_app.py**

**/ | \**

**/ | \**

**▼ ▼ ▼**

**geospatial\_ demand\_ competition\_**

**calcs.py profit\_ capacity\_**

**calcs.py checks.py**

**\ | /**

**\ | /**

**▼ ▼ ▼**

**Potential New Routes Data**

**|**

**▼**

**flight\_simulator.py**

**|**

**▼**

**Top Recommended Routes**

**/ \**

**/ \**

**▼ ▼**

**Terminal Table Folium Map**

**| Generation**

**| |**

**| ▼**

**| indigo\_new\_routes\_map.html**

**|**

**▼**

**CLI Interaction**

**|**

**▼**

**Flight Details**

* **main\_app.py**: The central orchestrator.
* **constants.py**: Stores all static configuration parameters.
* **airports\_data.py**: Handles data input and initial processing.
* **geospatial\_calcs.py**: Contains geographical calculation logic.
* **demand\_profit\_calcs.py**: Encapsulates demand and profitability estimation.
* **competition\_capacity\_checks.py**: Manages competition and airport capacity simulations.
* **flight\_simulator.py**: Generates simulated flight schedules.
* **airports.csv**: The raw data source.
* **indigo\_new\_routes\_map.html**: The interactive map output.
* **Terminal (CLI)**: The command-line interface for user interaction.

**Data Flow Diagram**

**airports.csv**

**|**

**"Loaded by"**

**|**

**▼**

**airports\_data.py**

**|**

**"Cleaned DataFrame"**

**|**

**▼**

**main\_app.py ◆**

**|**

**"Loop through Airport Pairs"**

**|**

**▼**

**geospatial\_calcs.py**

**|**

**"Distance (km)"**

**|**

**▼**

**main\_app.py ◆**

**/ \**

**"Call"/ \"Call"**

**/ \**

**▼ ▼**

**demand\_profit\_calcs.py competition\_capacity\_checks.py**

**| |**

**"Demand & Profit Scores" "Competition & Capacity Status"**

**| |**

**\ /**

**\ /**

**▼ ▼**

**main\_app.py ◆**

**|**

**"Filter & Select Routes"**

**|**

**▼**

**Potential New Routes Data ◆**

**|**

**"Call"**

**|**

**▼**

**flight\_simulator.py**

**|**

**"Simulated Flights"**

**|**

**▼**

**Potential New Routes Data ◆**

**|**

**"DataFrame"**

**|**

**▼**

**Top Recommended Routes**

**/ \**

**"Displayed in" "Visualized by"**

**/ \**

**▼ ▼**

**Terminal Table Folium Map Generation**

**| |**

**"User Selects Route" "Output"**

**| |**

**▼ ▼**

**CLI Interaction indigo\_new\_routes\_map.html**

**|**

**"Displays"**

**|**

**▼**

**Flight Details**

**4.Implementation**

The project is implemented entirely in **Python 3.x**, leveraging several powerful libraries:

* **Pandas**: Used extensively for data loading, cleaning, filtering, and manipulation of tabular airport data. Its DataFrame structure is central to managing the datasets.
* **Folium**: Employed for creating the interactive geographical map. It allows for easy plotting of markers and lines on a Leaflet.js map, providing a rich visualization of the recommended routes.
* **NumPy**: Used indirectly by Pandas for numerical operations, especially in calculations involving geographical coordinates.
* **venv (Python's built-in virtual environment module)**: Utilized to create an isolated Python environment, ensuring that project dependencies are managed separately and do not conflict with other Python projects on the system. This promotes reproducibility and clean development.
* **Modular Design**: The code is structured into several .py files (constants.py, airports\_data.py, geospatial\_calcs.py, demand\_profit\_calcs.py, competition\_capacity\_checks.py, flight\_simulator.py, main\_app.py) to enhance readability, maintainability, and reusability of functions.

**5.Code & Explanation**

The project's code is meticulously commented to explain the logic, purpose of functions, and key decision points.

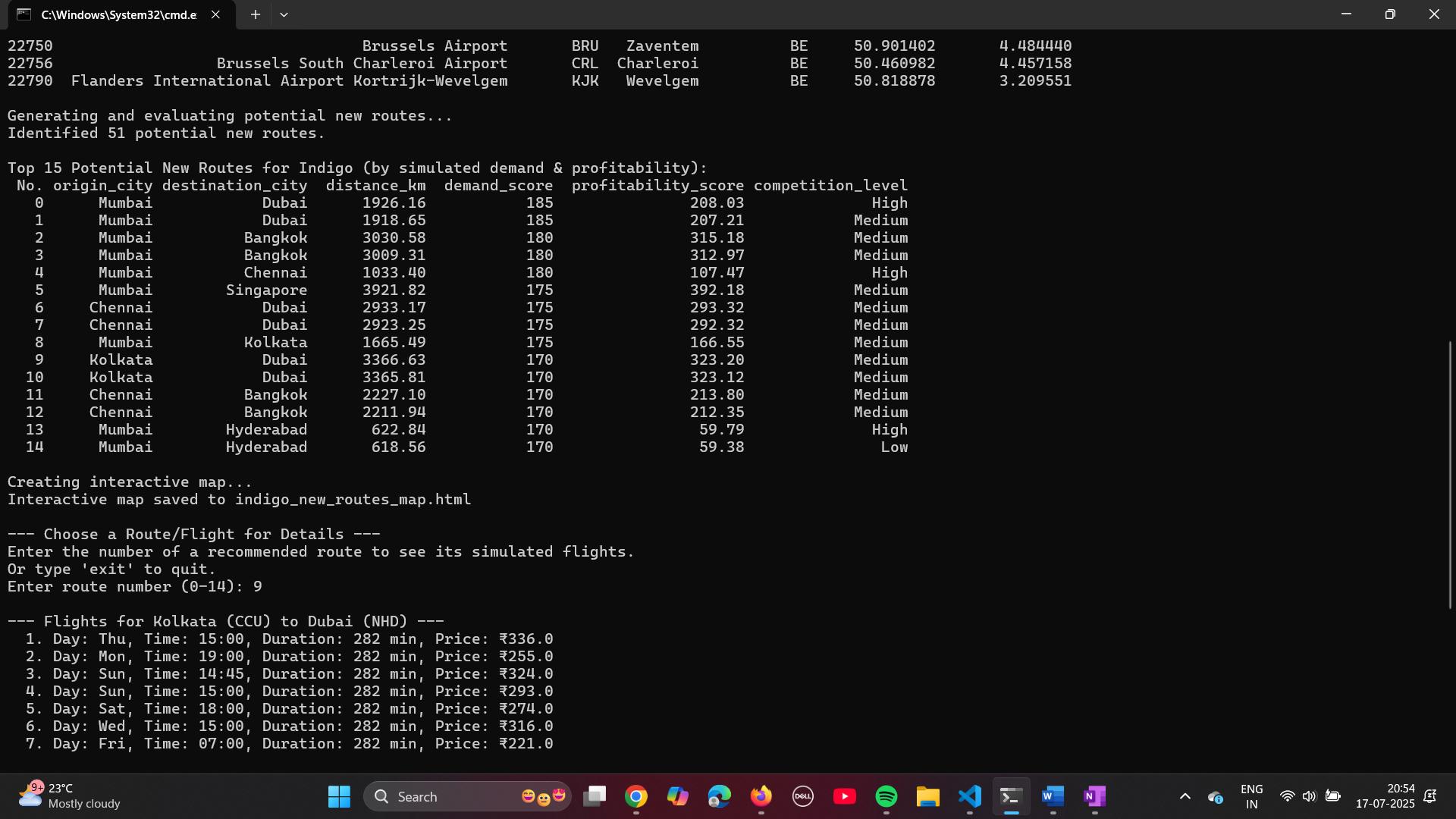
* **main\_app.py**: This is the entry point of the application. It orchestrates the entire workflow:
  + Imports necessary components from other modules.
  + Calls load\_and\_clean\_airports to prepare the data.
  + Contains the main nested loops that iterate through potential origin-destination pairs.
  + Applies all the analysis functions (haversine, get\_route\_demand\_score, calculate\_profitability\_score, assess\_competition\_level, check\_airport\_capacity, simulate\_flights\_for\_route) to each pair.
  + Implements the core business logic for filtering and selecting recommended routes based on combined thresholds (demand, profitability, competition, capacity).
  + Generates the Folium map and saves it.
  + Manages the command-line interface for user interaction.
* **constants.py**: Defines all global variables, configuration parameters, and thresholds used throughout the application. This centralizes configurable values, making it easy to adjust assumptions without modifying core logic.
* **airports\_data.py**: Dedicated to data handling, including reading airports.csv, filtering by country and airport type, and selecting/renaming relevant columns.
* **geospatial\_calcs.py**: Contains the haversine function, a pure mathematical utility for calculating distances.
* **demand\_profit\_calcs.py**: Groups functions related to market analysis, specifically get\_route\_demand\_score (simulated demand) and calculate\_profitability\_score (simulated profitability).
* **competition\_capacity\_checks.py**: Houses functions that simulate external constraints: assess\_competition\_level (market competition) and check\_airport\_capacity (airport operational capacity).
* **flight\_simulator.py**: Contains simulate\_flights\_for\_route, which generates realistic-looking (but simulated) flight schedules, durations, and prices for a given route.

Each function within these modules includes a docstring explaining its purpose, arguments, and return values. Complex logical blocks are broken down with inline comments.

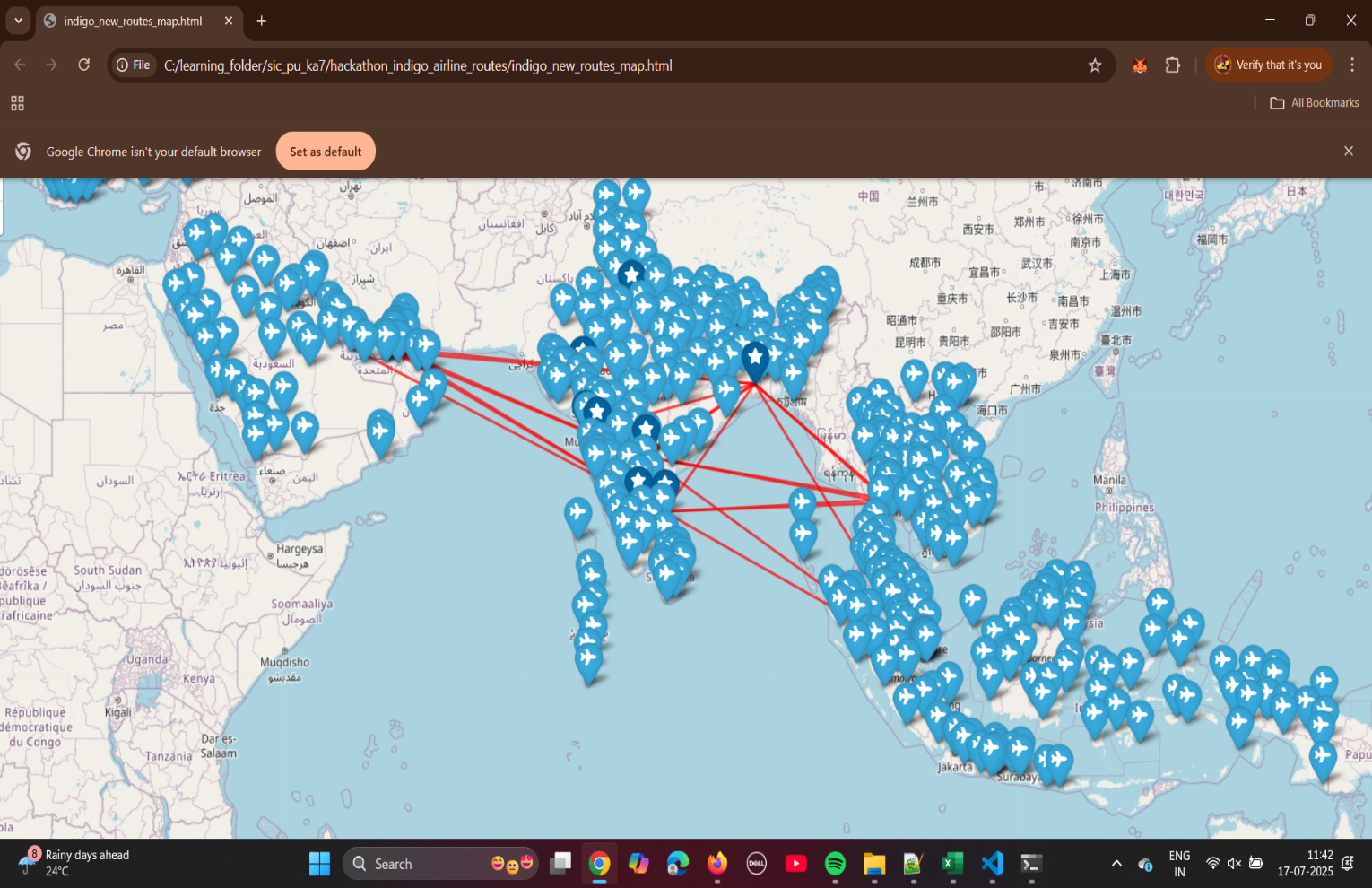
**6.Output Screenshots**

**1. Terminal Output - Top Recommended Routes**

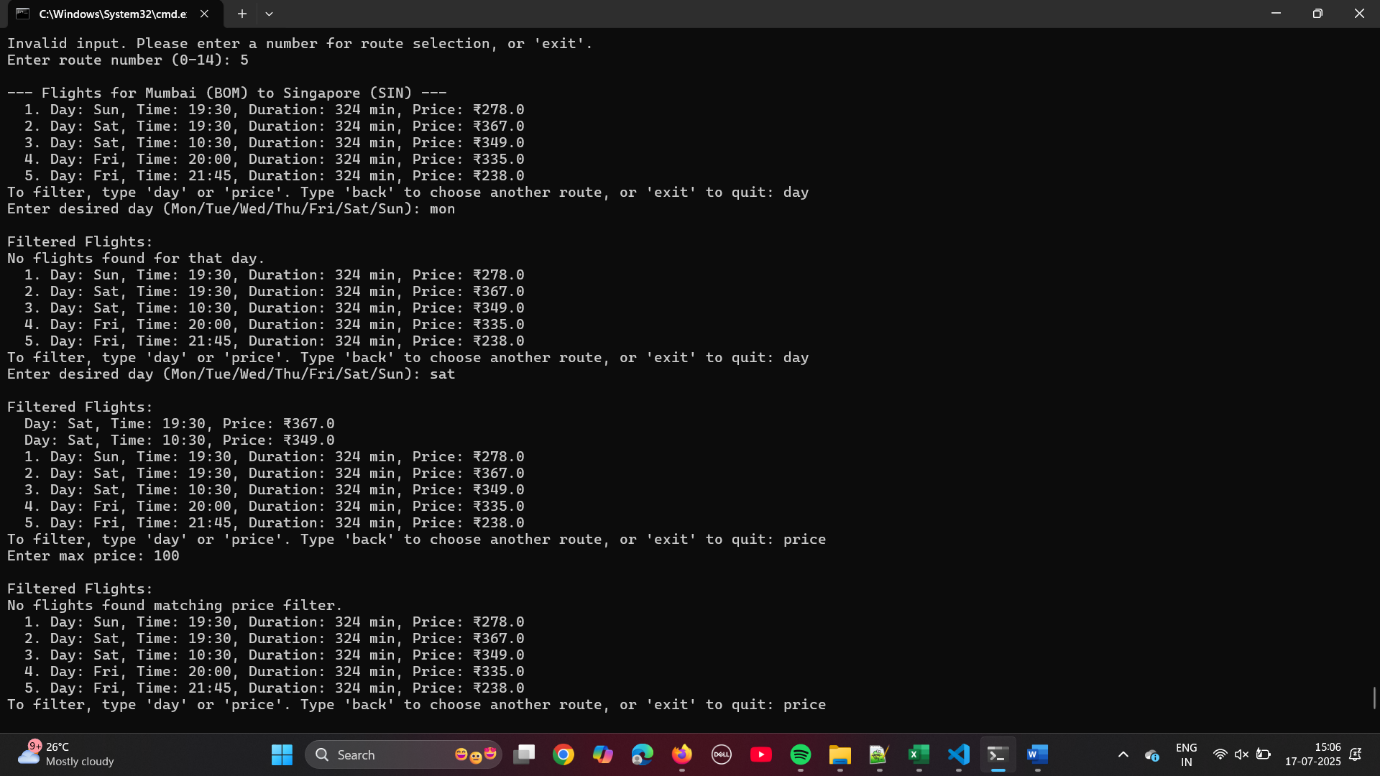
[Insert Screenshot of your terminal showing the list of top 15 routes]



**2. Interactive Map (Browser View)**

****

**3. Terminal Output - Flight Details CLI Interaction**

****

**7.Closure**

This hackathon project successfully demonstrates a conceptual framework for data-driven airline route optimization. By integrating geographical analysis, simulated market demand, profitability, competition, and operational constraints, the tool provides valuable insights for strategic network expansion. The modular Python implementation ensures maintainability, while the interactive map and CLI offer intuitive ways to explore the results. Future enhancements could include integrating real-world data APIs, more sophisticated predictive models, and a full-fledged web-based GUI for broader accessibility.

**8.Bibliography**

* **Airport Data**: https://www.google.com/search?q=OurAirports.com. (Accessed: July 2025). <https://ourairports.com/data/>
* **Python Libraries**:
  + Pandas: <https://pandas.pydata.org/>
  + Folium: <https://python-visualization.github.io/folium/>
  + NumPy: <https://numpy.org/>

####################################################################################################################################################################