



ENHANCING FLIGHT NAVIGATION MECHANISM FOR OPTIMAL ROUTE PLANNING AND RISK MITIGATION

CREATIVE MINDS

May 25, 2024



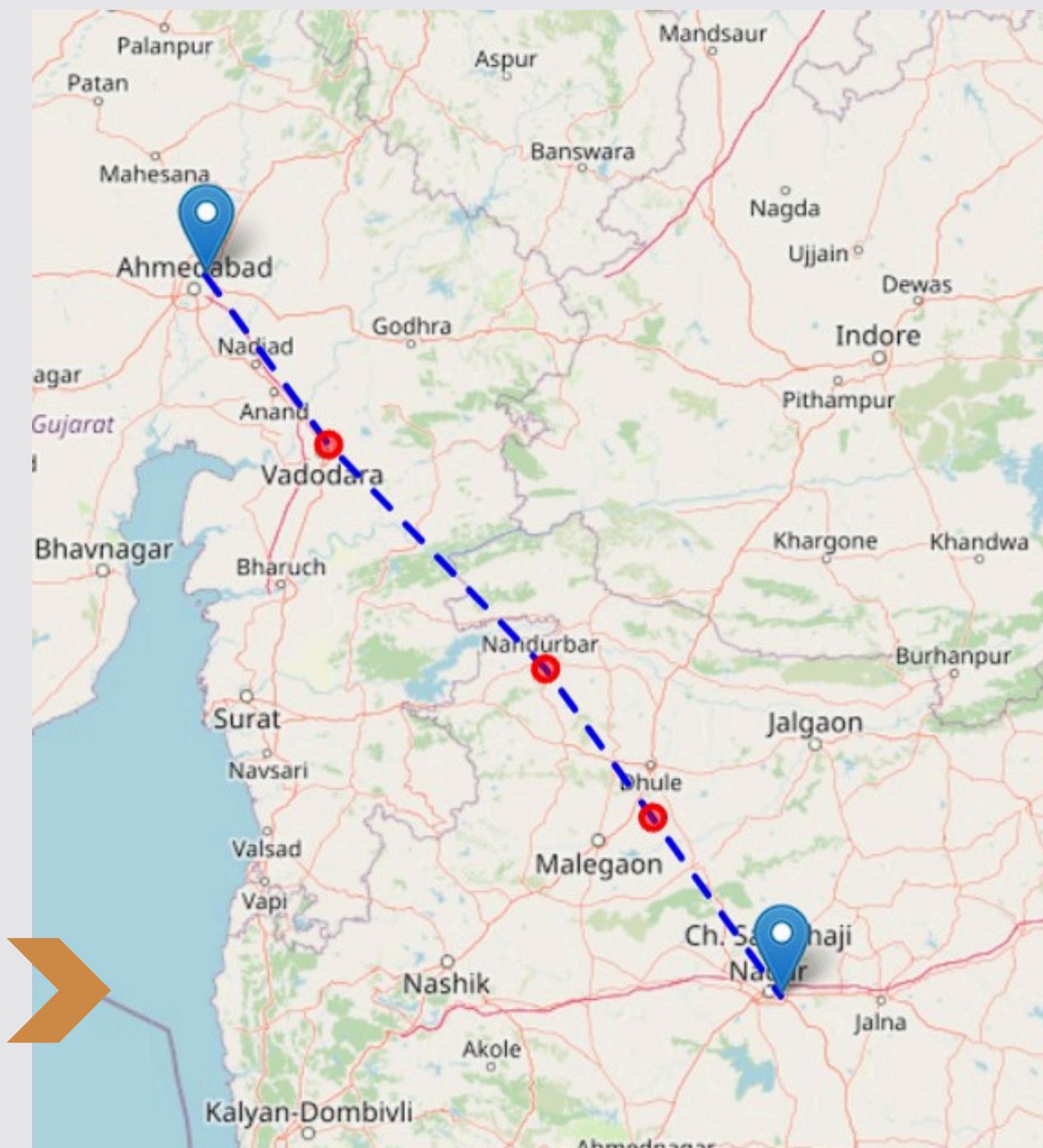
Problem Statement

Goal

- Design, develop, and implement a robust software solution for identifying optimal flight paths.
- Provide real-time risk assessment and alternative route suggestions.
- Integrate real-time health metrics tracking from flight sensor data.

Factors Taken into account

- Shortest Path
- Weather code
- Precipitation (Turbulence)
- Wind Speed (Stability)
- Relative Humidity (Static electricity)
- Temperature (Engine Efficiency)
- Rain
- Snowfall
- Cloud Cover (Visibility)



Tech Stack



Frontend

Languages
HTML ,CSS, JS,Tailwind

Frameworks
React, Node,Apache Kafka

Tools
Vs code, Github



Backend

Languages
Java , SQL

Frameworks
Spring Boot, Spring Data JPA, Junit

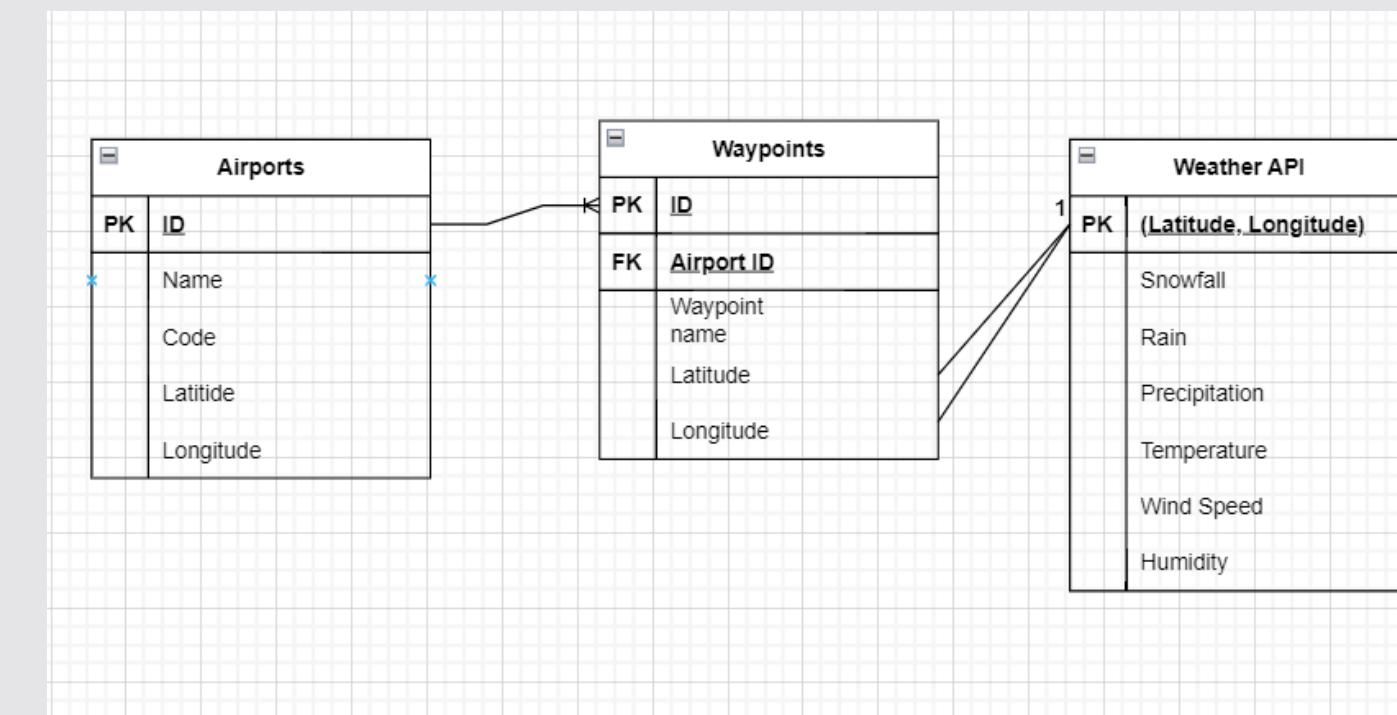
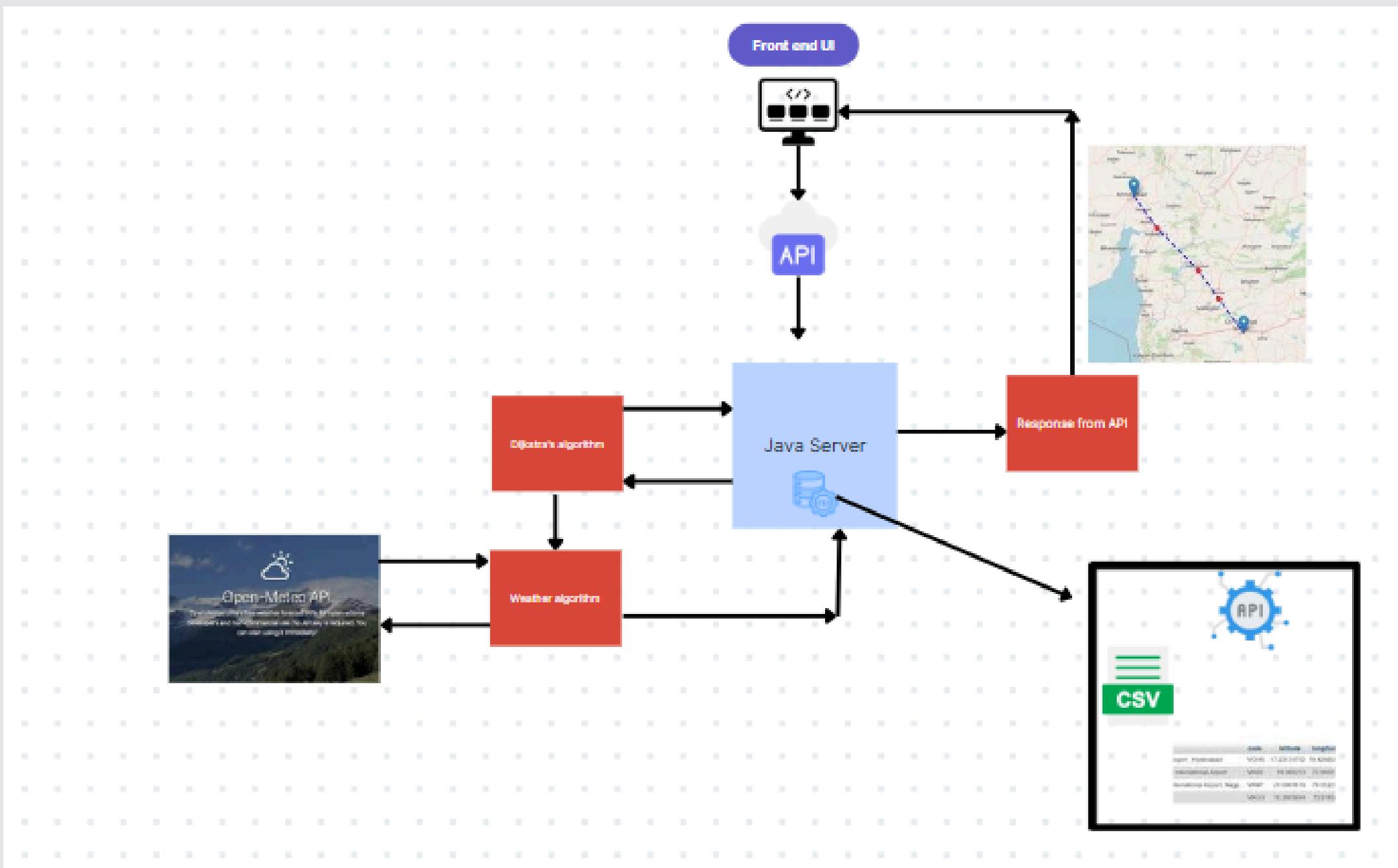
Database
MySQL

Tools
Postman, Eclipse, XAMPP

Other
REST API



Architecture and ERD



Features



- Real Time Route Planner based on weather and three different popular algorithms and logic which includes Dijkstra's algorithm, Haversine algorithm and waypoint logic created by us.2018 : Expansion into new markets, establishing a global presence in Arrowwai.
- Real time weather data from open meteo api

Route Generator

Enter departure and destination ICAO codes to compute a new flight plan. This route generator attempts to find an optimal route between two points, taking into account waypoints and intersections. When crossing the Atlantic or the Pacific, the current oceanic tracks can optionally also be considered. Our route generator is currently limited to 10 waypoints and 10 intersections. Please be patient while the system generates the route.

Departure ICAO: [Input Field] Destination ICAO: [Input Field]

Weather Data

Temperature at 2m: [Graph showing temperature fluctuations over time]

Flight Route Map

Optimal Way / Optimal Way

Flight Route Map

Waypoint 4
Latitude: 17.6744
Longitude: 75.9989
Temperature: 30°C
Humidity: %
Wind Speed: 8.6 km/h
Precipitation: mm

Dashboard Sign Up Sign In

Welcome!

Login or create new account.

Register with

Facebook Apple Google

or

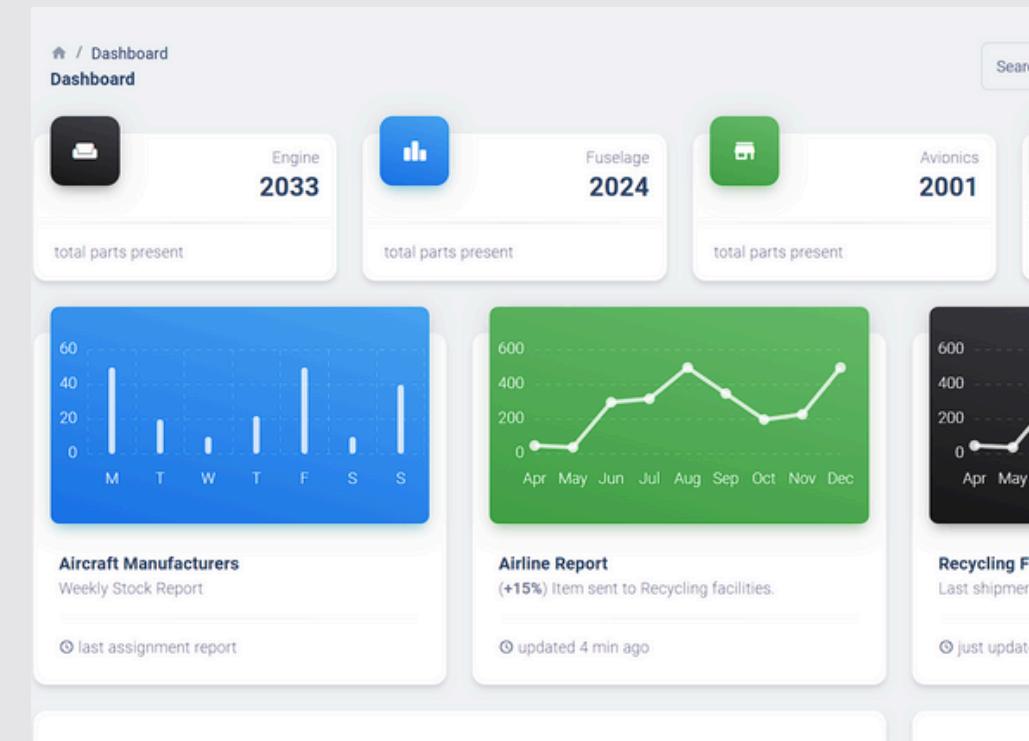
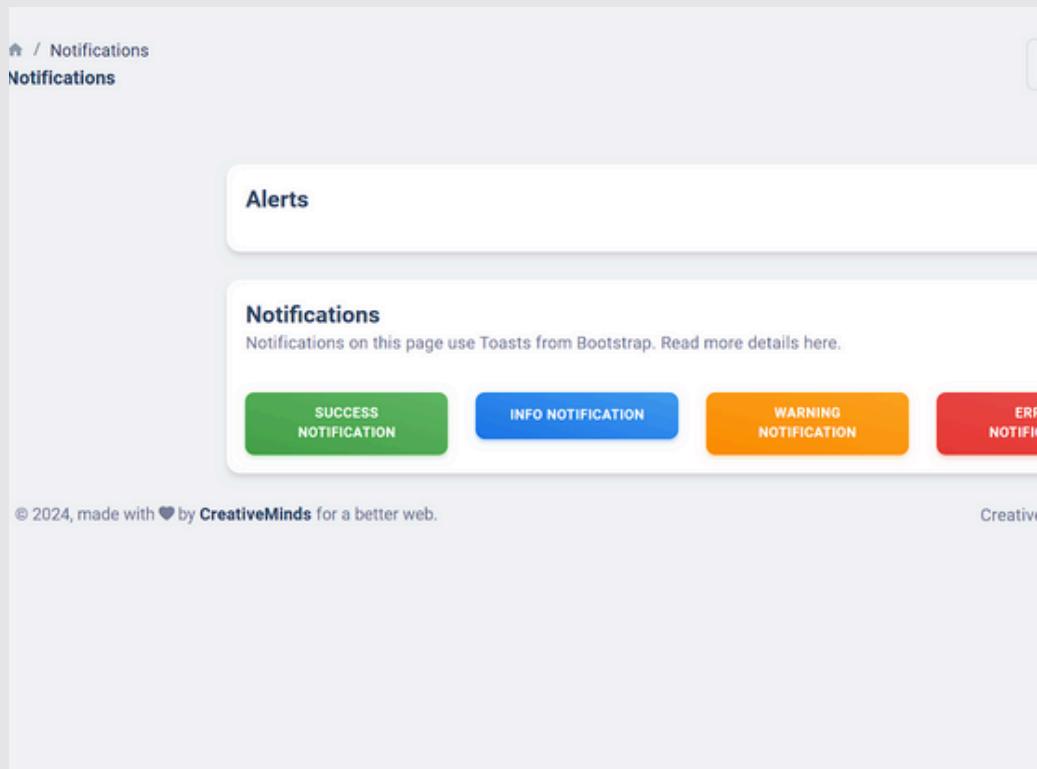
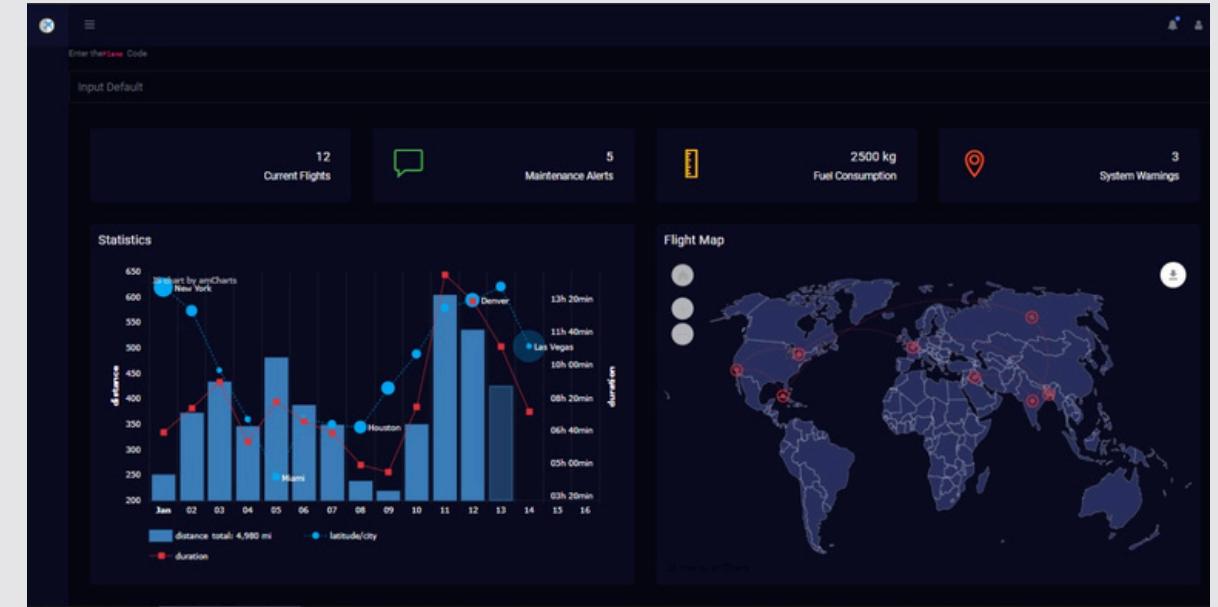
Name: [Input Field]
Email: [Input Field]
Password: [Input Field]

I agree the [Terms and Conditions](#)

Features



- Flight Maintenance Dashboard with notifications to track information about flights .
- Real time data from flight sensor which include fuel consumption, Pressure, ETG readings.
- Information about flight map current flight and summary of flights over a month.





Use Case

There are multiple use case for our project out of which five key use cases that are essential for modern aviation management: real-time weather data, flight optimal route through waypoints, flight optimal route based on weather, flight maintenance dashboard, and flight metrics dashboard

- **Real time weather data**
- **Flight Optimal Route through waypoints**
- **Flight Optimal Route based on weather**
- **Flight Maintenance Dashboard**
- **Flight Metrics Dashboard**



Algorithms

I - Haversine Algorithm

Haversine is an algorithm that can determine the distance between two objects on the surface of a sphere. Haversine has now been developed by using a simple formula, which with computer calculations, can provide a very accurate level of precision between two points.

The Haversine formula is perhaps the first equation to consider when understanding how to calculate distances on a sphere. The word "Haversine" comes from the function:

$$\text{haversine}(\theta) = \sin^2(\theta/2)$$

The following equation where φ is latitude, λ is longitude, R is earth's radius (mean radius = 6,371km) is how we translate the above formula to include latitude and longitude coordinates. Note that angles need to be in radians to pass to trig functions:

$$a = \sin^2(\varphi_B - \varphi_A/2) + \cos \varphi_A * \cos \varphi_B * \sin^2(\lambda_B - \lambda_A/2)$$

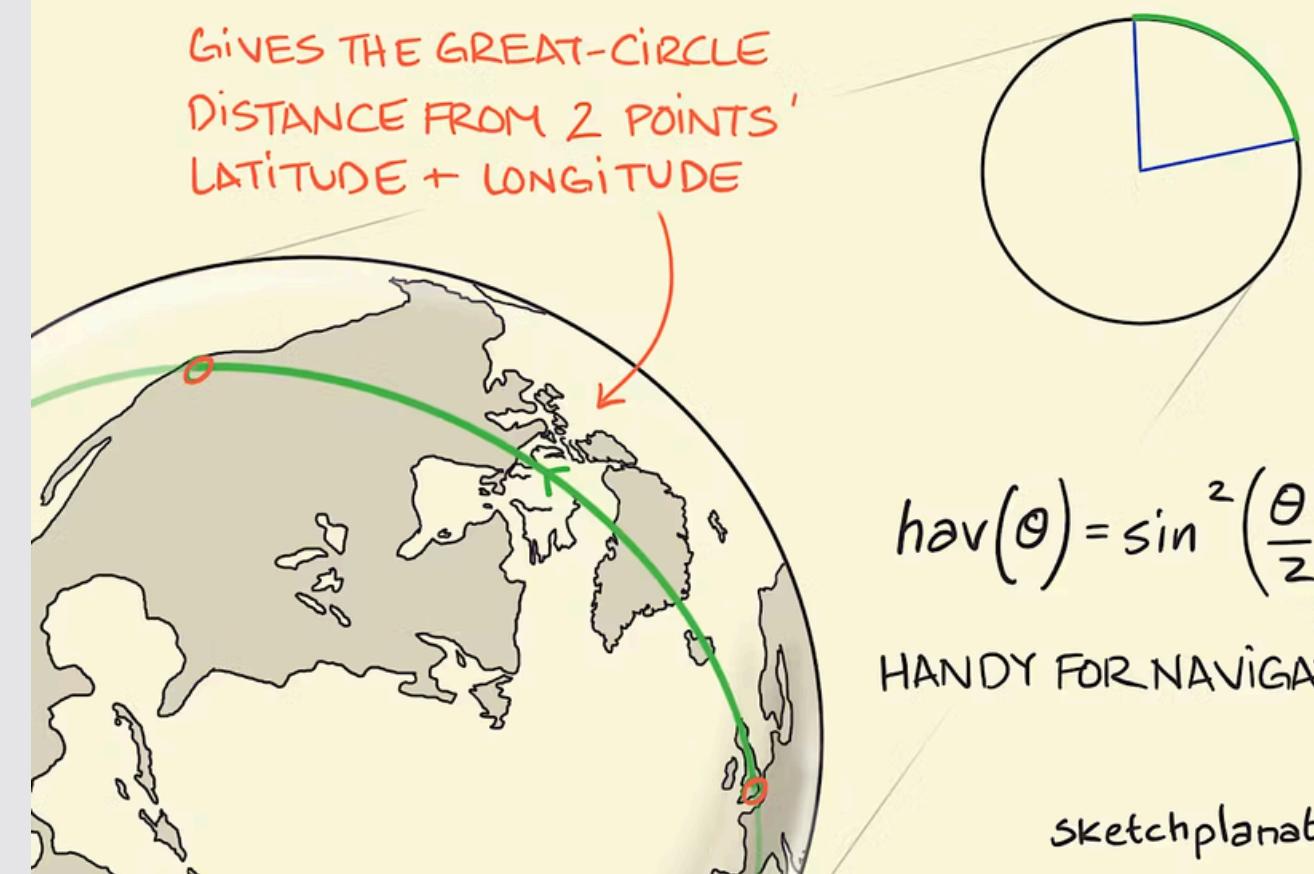
$$c = 2 * \text{atan2}(\sqrt{a}, \sqrt{1-a})$$

$$d = R \cdot c$$

THE HAVERSINE FORMULA

FINDING THE SHORTEST DISTANCE BETWEEN 2 POINTS ON A SPHERE

GIVES THE GREAT-CIRCLE DISTANCE FROM 2 POINTS' LATITUDE + LONGITUDE





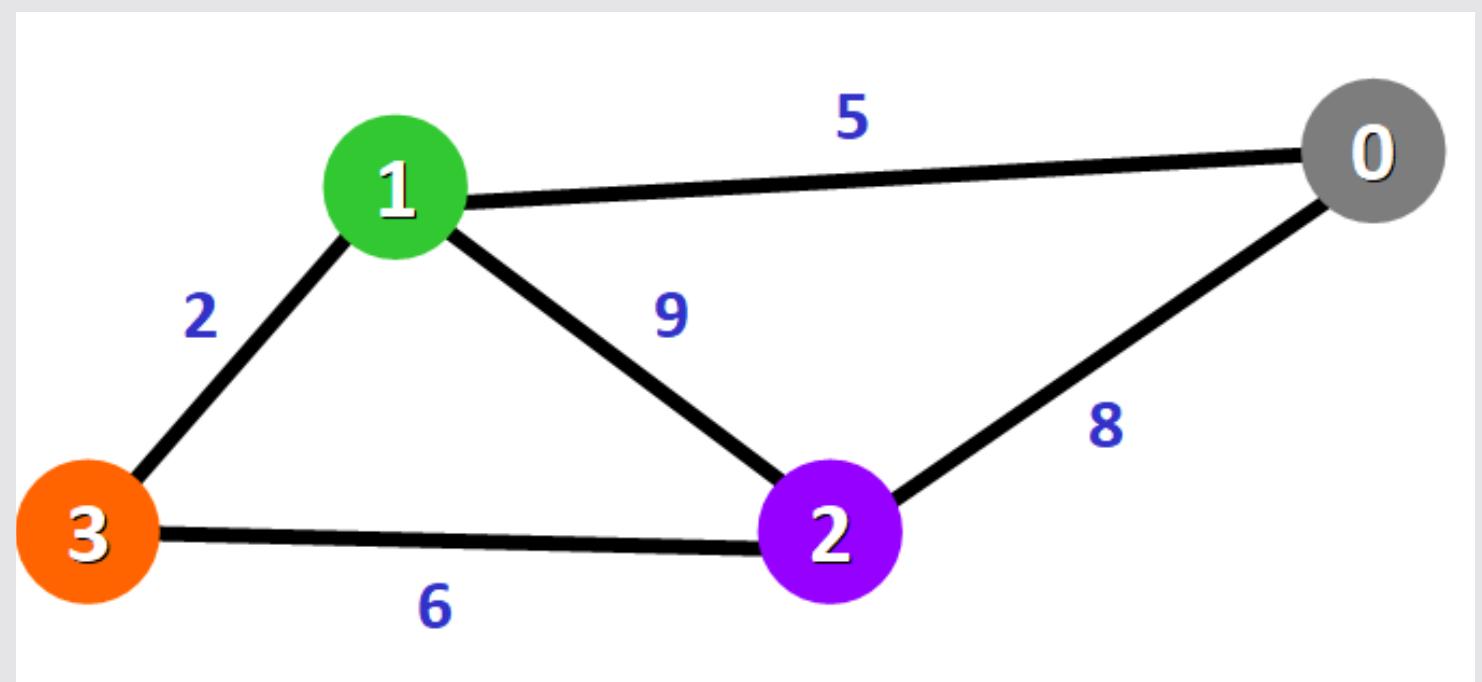
Algorithms

II- Dijkstra's Algorithm

Dijkstra's algorithm is a popular algorithms for solving many single-source shortest path problems having non-negative edge weight in the graphs i.e., it is to find the shortest distance between two vertices on a graph. It was conceived by Dutch computer scientist Edsger W. Dijkstra in 1956.

Algorithm for Dijkstra's Algorithm:

1. Mark the source node with a current distance of 0 and the rest with infinity.
2. Set the non-visited node with the smallest current distance as the current node.
3. For each neighbor, N of the current node adds the current distance of the adjacent node with the weight of the edge connecting 0->1. If it is smaller than the current distance of Node, set it as the new current distance of N.
4. Mark the current node 1 as visited.
5. Go to step 2 if there are any nodes are unvisited.



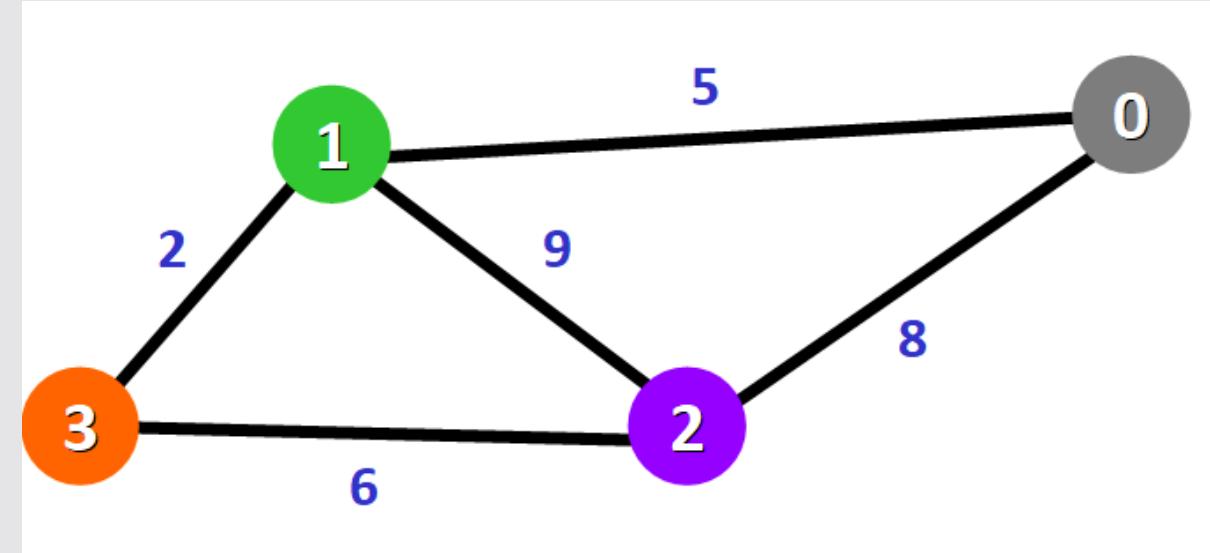
Implementation Using Mentioned Algorithms and Weather Factors By Waypoint Based Approach



For finding the optimal path we are considering Waypoint based path. We can calculate the shortest straight direct path using Haversine formula but in real-time its not possible as straight path may come under unauthorised flight area or any other issue.

So we choose multiple waypoints between source and destination that are authorised and choose the shortest path connecting them .We use Haversine formula to find shortest distance between them and connect them using Dijkstra's shortest path algorithm.

In this way we get result as shortest optimum path(fuel-optimised) of waypoints connected with each other from source to destination.



Considering Weather factor-

We fetch the real-time weather data using API(open-meteo) for waypoints based on their coordinates.We consider factors -temperature(Engine efficiency), cloud cover(visibility), wind speed (flight Stability),Precipitation(turbulence)Snowfall,Humidity(Static Electricity), Rain,Weather Code and disable the waypoints that are not falling under desired range for above factors and feed it to Djikstra-Haversine algorithm.

In this way we get result as shortest optimum path of waypoints connected with each other from source to destination considering weather/external factors.



Future Scope

There are many chaneg that we can include in future scope which inludes

1. Integrating real time APIs for flight maintaince records
2. Implementing Object Detection algorithm like Yolo to idenity defects.
3. Increasing number of waypoints for each airports.
4. Adding more info about flights and adding more factors for optimizaing route.



A complex network graph composed of numerous small, glowing blue spheres connected by thin white lines, forming a dense web of nodes and connections. The graph is set against a dark, almost black, background.

THANK YOU

CREATIVE MINDS