🌍AirAware – Air Quality Monitoring & Prediction System

AirAware is a full-stack Air Quality Monitoring and Prediction system designed to analyze air pollution data, visualize trends, predict AQI levels, and provide health advisories. The project integrates **backend APIs**, **database handling**, and **machine learning logic** to deliver a complete data-driven solution.

This project was developed with a strong focus on **backend development, API handling, and ML integration**, following a milestone-based approach.

🙊Project Objectives

- Monitor air quality data across cities (Maharashtra dataset)
- Visualize PM2.5 and PM10 pollution trends
- Predict AQI levels based on pollutant values
- Provide health advisories based on AQI
- Enable data download and user interaction

🍂Project Architecture (Flow)

1. **Dataset Ingestion**
2. Maharashtra AQI dataset is stored in the database

3. Data includes PM2.5, PM10, city, date, and other attributes

4. **Backend Processing (Flask)**

5. Flask handles routing, authentication, APIs, and ML logic

6. Environment variables are loaded securely using `.env`

7. **Database Layer (MySQL)**

8. Stores air quality data, users, and feedback

9. Queries aggregate pollution values for dashboards

10. **Machine Learning & AQI Logic**

11. AQI calculated using PM2.5 and PM10 weighted formula

12. AQI category mapping based on Indian AQI standards

13. **Frontend Rendering**

14. HTML templates render dashboards, charts, prediction pages
15. JavaScript fetches API data dynamically

Tech Stack Used

Backend

- Python (Flask)
- MySQL
- dotenv (Environment configuration)

Machine Learning

- NumPy
- Scikit-learn (Linear Regression – basic AQI modeling)

Frontend

- HTML, CSS, JavaScript
- Chart-based visualizations

Important Files & Their Role

`.env`

Stores sensitive database credentials securely: - DB_HOST - DB_USER - DB_PASSWORD - DB_NAME

`app.py`

Main backend application file: - User authentication (login/logout) - Dashboard data aggregation - AQI prediction logic - Health advisory system - Chatbot logic - Language translation support - Data download APIs

Dataset File

- Maharashtra AQI dataset used for analysis and predictions

🦣Key Features Explained

🐓Dashboard

- Displays city-wise average PM2.5 and PM10
- Month-wise pollution trends via charts

🐕 AQI Prediction

- Takes PM2.5 and PM10 inputs
- Computes AQI using weighted formula
- Categorizes AQI into Good, Moderate, Poor, etc.

🐗Health Advisor

- Provides health tips based on AQI range
- Suggests mask usage and risk level

🐚Chatbot

- Rule-based chatbot for AQI-related queries
- Answers about pollution, masks, health risks

🐜Data Download

- Download full dataset as CSV
- Download city-wise air quality data

🐝Language Support

- English & Hindi language switching

🌳How to Run the Project

Step 1: Clone the Repository

```
git clone <repository-url>
cd airaware
```

Step 2: Create Virtual Environment (Optional)

```
python -m venv venv
venv\Scripts\activate
```

Step 3: Install Dependencies

```
pip install -r requirements.txt
```

Step 4: Configure Environment Variables

Create a `.env` file:

```
DB_HOST=localhost
DB_USER=root
DB_PASSWORD=your_password
DB_NAME=air_quality_db
```

Step 5: Setup Database

- Create MySQL database `air_quality_db`
- Import AQI dataset into `air_quality` table
- Create required tables (`users`, `feedback`)

Step 6: Run the Application

```
python app.py
```

Open browser and visit:

```
http://127.0.0.1:5000
```

🍁Testing Flow

- Login with valid credentials
- View dashboard pollution trends
- Predict AQI using PM values
- Check health advisory
- Download datasets

🌀Milestone Contribution Summary

- **Backend Development:** Complete Flask API design
- **API Handling:** Dashboard, prediction, download, chatbot APIs

- **Database Integration:** MySQL queries & aggregation
- **ML Integration:** AQI calculation & prediction logic

🏖️Conclusion

AirAware successfully demonstrates how air quality data can be transformed into meaningful insights using backend APIs, database systems, and machine learning logic. The project emphasizes real-world problem-solving by combining data analysis, prediction, and health awareness into a single platform. This system can be further extended with real-time sensors, advanced ML models, and modern frontend frameworks, making it a strong foundation for scalable environmental monitoring solutions.

✨*Project developed as part of an academic major project focusing on Backend, APIs, and ML Integration.*