Infosys Springboard Virtual Internship 6.0 Completion Report

Team Details

Team Number: 2
Start date: 22 Sep 2025
Intern Names - Shruti Singh, R Doondi Gangadhar, Rahul Vangari, Shiven Poojary, Payal Rawat
Internship Duration: 8 Weeks

1.Project Title: Forecasting Air Quality Using Historical Pollution Data

2.Project Objective: The main objective of this project is to design and develop an air quality forecasting system that predicts the Air Quality Index (AQI) and concentrations of major pollutants such as PM2.5, PM10, NO_2, SO_2, and O_3 using historical pollution and environmental data.

This project aims to empower environmental agencies, city planners, and citizens with timely and accurate insights to anticipate pollution levels, issue early warnings, and support proactive environmental decision-making. By leveraging time series forecasting models and interactive visualization tools, the system provides both predictive analytics and intuitive interfaces for exploring pollution trends and seasonal variations.

3. Project Description in Detail: Air Quality Forecasting System is a Python-based application integrated with machine learning frameworks and a Streamlit-based web dashboard. It is designed to streamline the collection, analysis, and prediction of air quality metrics, combining data science techniques with real-time visualization and alerting capabilities.

Approach Taken:

1.Requirement Analysis: The project began with an analysis of the challenges faced by environmental monitoring bodies and urban planners—such as the need for real-time air quality insights, difficulty in interpreting historical pollution data, and the lack of automated forecasting tools.

Requirements were gathered to ensure the system supports multi-city data, handles missing timestamps, and provides interpretable visual forecasts.

2.System Design and Architecture: A modular and scalable architecture was designed to separate data preprocessing, model training, forecasting, and visualization components.

The architecture includes:

- A data ingestion module for importing pollution datasets from sources like Kaggle.
- A modeling layer integrating time series models such as ARIMA, Prophet, LSTM, and XGBoost.
- A Streamlit web interface for user interaction and result visualization.
- Admin controls for dataset uploads and model retraining.

3.Development: Implemented data preprocessing pipelines to clean, resample, and normalize time series pollution data.

- Trained and compared multiple forecasting models based on metrics like RMSE and MAE to identify the best-performing one for AQI prediction.
- Developed an alert mechanism that classifies forecasted AQI values into categories (Good, Moderate, Unhealthy, etc.) and highlights days likely to exceed safe thresholds.
- Designed an interactive Streamlit dashboard featuring:
  - AQI and pollutant trend visualization
  - Forecast charts and gauge meters
  - Alert banners for hazardous levels
  - Admin panel for uploading new data and retraining models.

4.Testing and Deployment: Comprehensive testing was conducted, including:
Data validation tests to ensure clean, consistent time series inputs.
Model evaluation tests for accuracy and robustness across different cities and seasons.
Interface usability testing to confirm a smooth and intuitive dashboard experience.

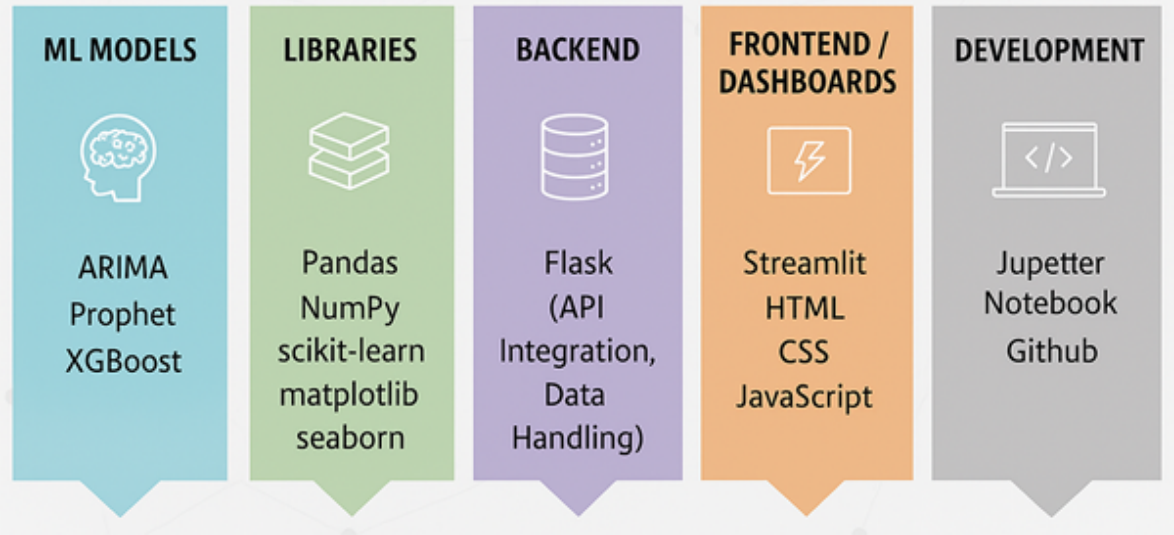
Impact and Real-World Implementation

● **Public Health & Awareness:**
Helps citizens make informed decisions by predicting poor air quality and issuing timely alerts.
● **Support for Authorities:**
Assists policymakers and planners with data-driven insights for pollution control and traffic regulation.
● **Research & Analysis:**
Provides valuable data for researchers to study pollution trends and environmental factors.
● **Scalability:**
Can be expanded to multiple cities, integrated with IoT sensors, and enhanced with advanced AI models.

4.Timeline Overview

Week	Activities Planned	Activities Completed
Week 1	Project kickoff, requirement gathering, and data source identification (Kaggle).	Project initialized, datasets collected, and requirements finalized.
Week 2	Data preprocessing, cleaning missing values, and exploratory data analysis (EDA).	Data cleaned, missing timestamps handled, and EDA completed with pollutant trend visualizations.
Week 3	Feature engineering (temporal and weather-based), and preparing data for modeling.	Feature set finalized, data normalized, and ready for model training.
Week 4	Model development and training using ARIMA, Prophet, and LSTM.	Models trained and evaluated; best model selected based on RMSE and MAE metrics.
Week 5	Implement AQI calculation logic and define alert thresholds.	AQI categories implemented; alert mechanism functional for unsafe pollution levels.
Week 6	Create visual trend analysis and pollutant contribution charts.	Trend visualization module completed with clear seasonal and pollutant patterns.
Week 7	Develop Streamlit dashboard for forecast visualization and admin panel.	Streamlit dashboard built with interactive plots, AQI gauge, and admin upload feature.
Week 8	Conduct final testing, report preparation, and project presentation.	Testing completed, report documented, and final demo successfully delivered.

5a.Tech Stack



5b. Project execution details

The AI-Driven Air Quality Forecasting System uses machine learning to predict AQI and pollutant levels from historical data. It provides an interactive dashboard for visualization and alerts.

1. **Planning & Design:** Collected datasets, defined objectives, designed architecture, and selected Python with Streamlit for development.
2. **Development & Integration:** Preprocessed data, engineered features, and trained multiple forecasting models (ARIMA, Prophet, LSTM, XG Boost). Implemented AQI logic and alert thresholds.
3. **Dashboard & Visualization:** Built a Streamlit dashboard for AQI forecasts, pollutant trends, and admin data upload.
4. **Testing & Deployment:** Conducted testing for accuracy and performance; deployed final dashboard for demonstration.
5. **Enhancements:** Added trend comparison, alert notifications, and documentation for final submission.

6. Snapshots / Screenshots

Fig-1: Air Quality Data Explorer

Frontend & Visualization (Web)

- HTML: Structures the dashboard layout and allows for dynamic insertion of cities into the dropdown.
- CSS/Bootstrap 5: Provides the clean, responsive design and styling.
- JavaScript: Handles all client-side logic, user interaction, and data fetching.
- Chart.js: A Powerful library used for all visualizations:
- Line Chart (Time Series)
- Bar Chart (PM2.5 Distribution)
- Bubble Chart (Pollutant Correlations)

Backend & Data Processing (Python)

- Flask: Lightweight web framework for building the API and serving the dashboard.
- Pandas & NumPy: Core tools for data loading, cleaning, and preparation.
- Scikit-learn (sklearn): Used specifically for Min-Max Scaling of the air quality data (0-1 range).
- API Logic: Handles dynamic data retrieval based on City and Pollutant filters.

Steps to Run the Air Quality Dashboard (AirAware Dashboard 1)

- File Setup: Ensure you have app.py, data_handler.py, the required templates/ and static/ folders, and the Dataset_AQI4-5 - Copy.csv file in your project root
- Install: Open your terminal and run :
- pip install Flask pandas numpy scikit-learn flask-cors
- 3. Run Server: Execute the main application file:
- python app.py
- 4. Access: Open your web browser to: http://127.0.0.1:5000

Fig-2: Air Quality Forecast Engine

Frontend & Visualization (Web)

1. UI/Style (index.html, Tailwind)

- Structure: HTML defines the layout, including controls (buttons, selectors) and chart areas.
- Styling: Tailwind CSS provides a fast, responsive, and utility-first design.

2. Logic (app.js)

- Data Fetching: Initiates AJAX calls to the Flask API (/api/data) upon user input.
- Interactivity: Handles clicks on Pollutant buttons and changes in Model/Horizon selectors to refresh data dynamically.

3. Charts (Chart.js)

- Tool: Uses the Chart.js library for all graphics.
- Visuals: Renders Grouped Bar Charts (Model Performance), Line Charts (Forecast), and updates them instantly with new JSON data.

Backend & Data Processing (Python)

1. Flask Backend (API)

- Role: Web server; defines the API route /api/data to serve dashboard data.
- Key Action: Safely defaults the city for data requests.

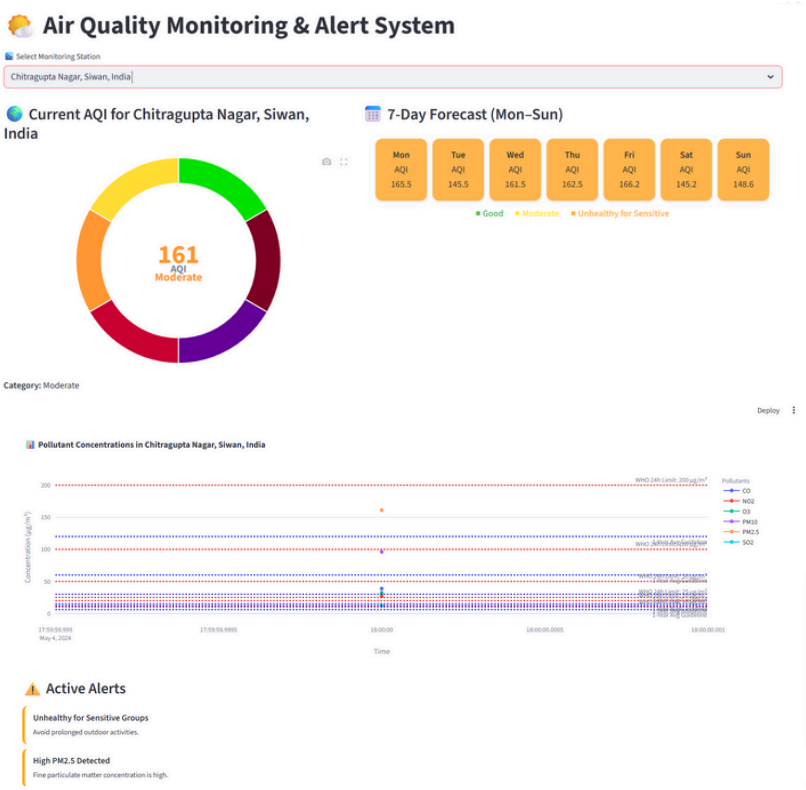
2. Data Pipeline (data_handler.py)

- Load & Clean
- Imputation: Fills missing values (NaN) using the median.
- ML Prep (Scaling): Applies MinMaxScaler to normalize pollutant data to a 0-to-1 range.
- Visualization

Steps to Run the Air Quality Forecast Engine(AirAware Dashboard 2)

- Install Python: Ensure Python 3.8+ is ready.
- Install Dependencies: Run pip install Flask pandas numpy scikit-learn flask-cors.
- Place Data: Put Dataset_AQI4-5 - Copy.csv in the root folder.
- Run Server: Execute python app.py.
- Access: Open http://127.0.0.1:5001/ in your browser.

Fig-3: Alert Logic & Trend Visualization



Flask (Backend API)

- Role: Server-Side data engine.
- Functions: Data Processing (Cleaning, Imputation, Scaling), Modeling Logic, and serving JSON data via API.

Streamlit

- Role: Client-Side interface.
- Functions: Provides the Interactive UI, requests data from Flask, and renders dynamic charts (Chart.js/Plotly).

Steps to Run the Alert Logic & Trend Visualization(AirAware Dashboard 3)

1.Setup

- Install Dependencies: Run pip install -r air_quality_api/requirements.txt and pip install -r air_quality_dashboard/requirements.txt.

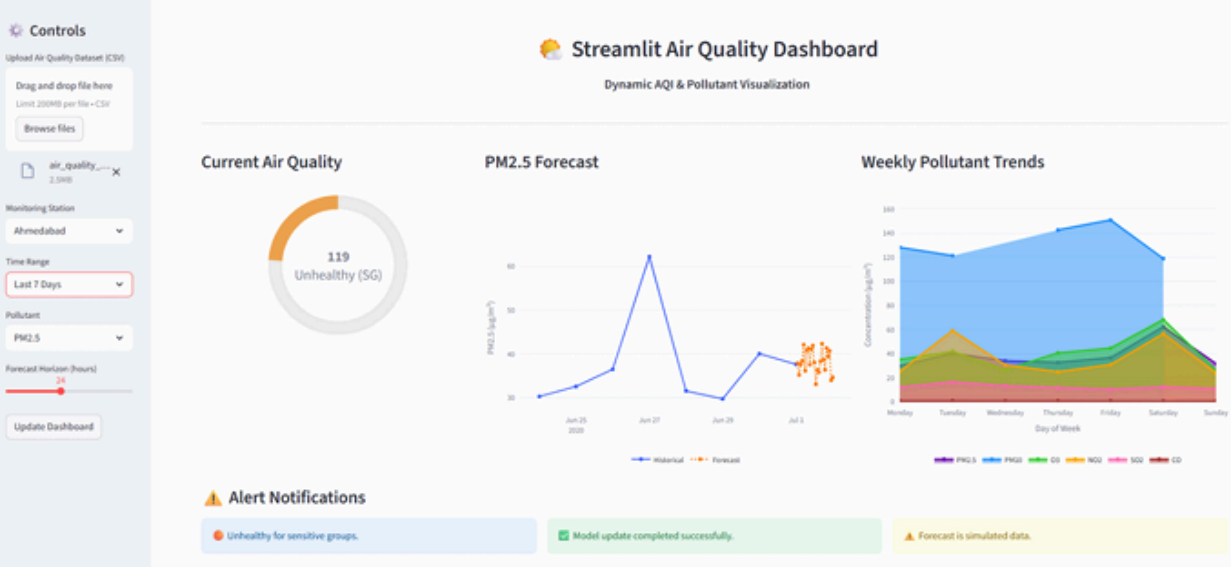
2. Start the Flask Backend (API)

- Navigate: cd air_quality_api
- Execute: python api.py
- Status: Must remain running (usually on port 5002).

3. Start the Streamlit Frontend (Dashboard)

- Navigate (New Terminal): cd air_quality_dashboard
- Execute: streamlit run dashboard_app.py
- Access: Open the browser link provided by Streamlit (usually http://localhost:8501).

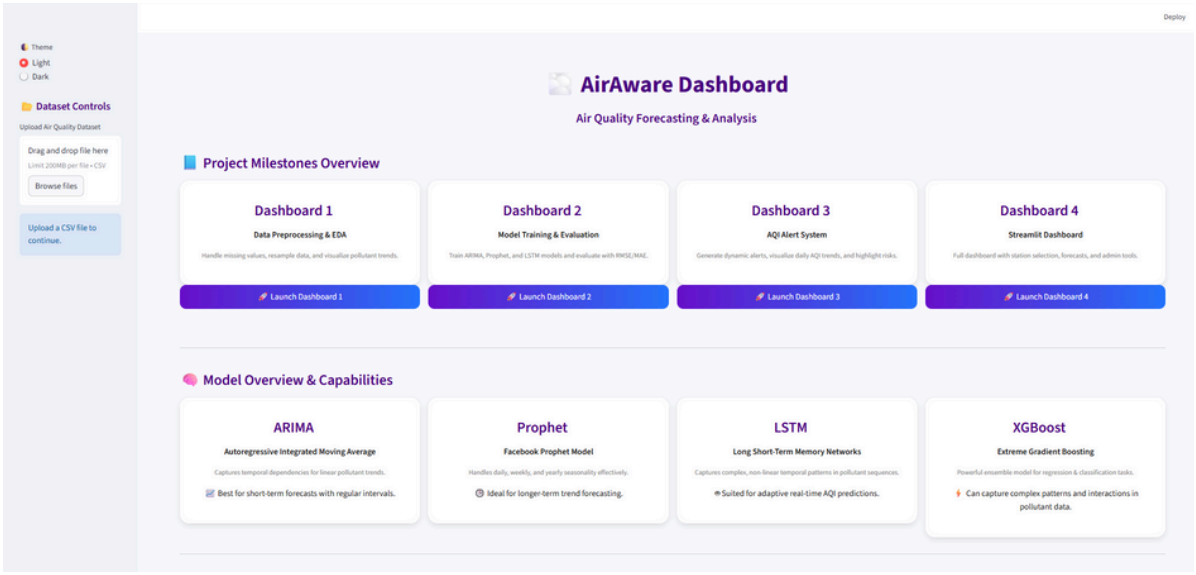
Fig-4: Streamlit Web Dashboard



Steps to Streamlit Web Dashboard(AirAware Dashboard 4)

- Open your project folder in VS Code or any IDE.
- Ensure you have Python 3.8 or higher installed.
- Open the terminal inside the project directory.
- Install required libraries: pip install streamlit pandas numpy plotly.
- Run the dashboard using: streamlit run app.py.
- Wait for the local URL to appear and click to open in your browser.
- Explore interactive AQI trends, pollutant data, and model insights in real time.

Fig-5: Main Dashboard



7. Challenges Faced

During the internship, several challenges were encountered in technical and operational areas. Each obstacle offered valuable learning opportunities and was overcome through systematic approaches.

1. Technical Challenges

- **Data Quality Issues:** Inconsistent and missing data in pollution datasets affected initial model accuracy.
- **Resolution:** Applied data cleaning, interpolation, and normalization techniques to ensure consistency.
- **Model Optimization:** Tuning parameters for LSTM and Prophet models to reduce forecast errors was challenging.
- **Dashboard Integration:** Integrating dynamic plots into the Streamlit dashboard initially caused performance lags.
- **Resolution:** Optimized rendering and used caching to improve dashboard responsiveness.

2. Operational Challenges

- **Time Management:** Balancing data preparation, model development, and dashboard design within weekly milestones.
- **Resolution:** Created a structured weekly task plan and prioritized key modules first.
- **Testing & Debugging:** Ensuring accurate forecasts and functional alerts required extensive testing.
- **Resolution:** Used systematic testing methods and visual validation with sample forecasts.

3. Communication Challenges

- **Scope Clarity:** Early confusion about feature depth and model type selection.
- **Resolution:** Regular mentor discussions helped refine project objectives and deliverables.

8. Learnings & Skills Acquired

- **Technical Skills:** Hands-on experience with Python, Streamlit, Pandas, Prophet, ARIMA, and LSTM models for time series forecasting.
- **Data Science & AI:** Improved understanding of preprocessing, feature engineering, and evaluation metrics (MAE, RMSE).
- **Project Management:** Strengthened planning, scheduling, and milestone tracking skills.
- **Soft Skills:** Enhanced problem-solving, communication, and teamwork through regular reviews and feedback sessions.

9. Testimonials from Experience

Working on this project was a highly valuable learning experience. It allowed us to apply data science and AI techniques to real-world environmental challenges.

Key achievements include:

- Successfully developing a functional AQI forecasting system with interactive visualization and alerts.
- Gaining practical exposure to time series modeling and dashboard development.
- Improving analytical and technical problem-solving abilities.
- Learning the importance of structured communication and feedback during project development.

Overall, this internship strengthened our confidence in applying machine learning to impactful, real-world problems.

10. Conclusion

The internship was an enriching experience that bridged the gap between theory and practice. Developing the AI-based Air Quality Forecasting System enhanced our technical expertise in data analysis, forecasting models, and web deployment.

It improved our project management and teamwork skills while deepening our understanding of how AI can contribute to sustainable environmental solutions. This project has been a key step toward our goal of becoming a proficient data scientist and AI practitioner.

11. Acknowledgements

We sincerely thank **Infosys Springboard** for providing the opportunity to work on this project.

We are deeply grateful to our mentor **Ms. Umme Asma S** for her constant guidance, technical insights, and constructive feedback throughout the internship.

We also thank our peers and collaborators for their support, teamwork, and encouragement, which made this project a rewarding experience.