Prediction of AQI using Airflow: A Real-Time Data Pipeline Approach

Saurabh Pawar, Prashant Warungase, Tanishk Nakhawa, Kartik Shinde, Vishal Pagar, Rutuja Nagdekar

*PG Diploma - Big Data Analytics*

*Centre for Development of Advanced Computing – ACTS*

Pune, India

[{kartikshinde727@gmail.com](mailto:%7bkartikshinde727@gmail.com), [tanishknakhwadrive42@gmail.com](mailto:tanishknakhwadrive42@gmail.com), [rsnagdekar@gmail.com](mailto:rsnagdekar@gmail.com), [pwarungase2000@gmail.com](mailto:pwarungase2000@gmail.com), [vishalpagar154@gmail.com](mailto:vishalpagar154@gmail.com)}

***Abstract*** — **Air quality monitoring has become a crucial aspect of environmental management due to its direct impact on public health and climate change. This project presents a real-time data pipeline for forecasting Air Quality Index (AQI) using Apache Airflow, OpenWeather API, PostgreSQL, and Grafana. The system collects air quality data from the OpenWeather API at regular intervals, processes it using Apache Airflow to ensure efficient data orchestration, and stores it in a PostgreSQL database for structured management. Grafana is then integrated to provide real-time visualization, enabling users to monitor trends and fluctuations in AQI effectively.**

**The implementation of Apache Airflow streamlines data ingestion and processing, ensuring reliability and scalability, while PostgreSQL facilitates efficient storage and retrieval. The integration with Grafana allows for interactive dashboards that enhance data interpretation. This pipeline offers a scalable and automated solution for AQI forecasting, which can be extended to include predictive analytics and machine learning models for more accurate forecasts. Future improvements may involve integrating additional data sources and refining the pipeline for enhanced performance. This project serves as a foundation for real-time environmental monitoring applications, providing valuable insights for researchers, policymakers, and the general public.**

***Keywords* — *Air Quality Index, Real-time data pipeline, Apache Airflow, OpenWeather API, PostgreSQL, Grafana, Data Orchestration, Environmental monitoring, AQI forecasting, Real-time visualization.***

I. INTRODUCTION

Air quality monitoring has emerged as a critical component in environmental management due to its profound implications on public health and climate change. The rapid pace of urbanization and industrialization in recent decades has led to elevated levels of air pollutants, which, in turn, contribute to respiratory and cardiovascular diseases among other health issues. The Air Quality Index (AQI) serves as a standardized metric that quantifies the level of air pollution, offering essential insights that inform policy decisions, public advisories, and environmental interventions.

Traditional AQI monitoring systems often struggle with data latency and fragmented processing, which can delay critical responses to deteriorating air quality. In light of these challenges, this paper presents an integrated, real-time data pipeline designed to enhance AQI forecasting accuracy and responsiveness. Our solution leverages Apache Airflow, OpenWeather API, PostgreSQL, and Grafana to form a robust system capable of seamless data ingestion, processing, storage, and visualization.

Apache Airflow is an open-source workflow orchestration platform that simplifies the creation, scheduling, and monitoring of workflows, often used in data engineering, ETL (Extract, Transform, Load) pipelines, and automation tasks. Workflows in Airflow are defined as Directed Acyclic Graphs (DAGs) written in Python, enabling flexibility and programmatic control over

task execution and dependencies. Each task in a DAG represents a discrete unit of work, and Airflow ensures that tasks are executed in the correct sequence, handles retries in case of failure, and provides detailed logs for debugging.

Airflow integrates seamlessly with a wide range of systems and services, including cloud platforms (like AWS, Google Cloud, and Azure), databases, APIs, and big data tools such as Apache Spark and Hadoop. Its web-based UI offers an intuitive way to visualize DAGs, monitor task statuses in real-time, and interact with running workflows, making it easy to track progress and troubleshoot issues. Airflow also supports advanced features like dynamic DAG generation, parameterization, and extensibility through plugins, allowing users to customize it to meet their specific needs. Its modular and scalable architecture makes it suitable for both small-scale workflows and enterprise-grade data pipelines, ensuring that it grows with the organization’s needs. With its community-driven development and robust feature set, Apache Airflow has become a standard tool for orchestrating and managing complex workflows in data-driven environments. This ensures that data extraction from the OpenWeather API, transformation, and loading into the PostgreSQL database occur in a timely and automated manner. With features like dynamic scheduling, error handling, and scalability, Airflow minimizes manual intervention and enhances the reliability of the data processing cycle.

Grafana is an open-source platform for monitoring, visualizing, and analyzing data from multiple sources in real-time. It supports integration with a wide range of databases and tools, including PostgreSQL, Prometheus, MySQL, InfluxDB, and cloud services like AWS, Azure, and Google Cloud. With its interactive and customizable dashboards, Grafana enables users to visualize metrics through graphs, tables, heatmaps, and more. It also features an alerting system that notifies users based on pre-defined thresholds, making it ideal for proactive monitoring. Grafana is highly extensible, offering plugins for custom visualizations and integrations, as well as API access for automation. Widely used in IT monitoring, business analytics, and DevOps, Grafana is a versatile tool that helps organizations gain insights into their data and improve operational efficiency.

Complementing the robust backend, Grafana plays a pivotal role in transforming stored data into actionable insights. Renowned for its powerful visualization capabilities, Grafana integrates seamlessly with PostgreSQL to create interactive, real-time dashboards. These dashboards enable stakeholders—ranging from environmental researchers to policy makers—to monitor AQI trends and identify anomalies promptly. Grafana’s flexibility in displaying data through various chart types and its support for real-time updates are critical for maintaining situational awareness in dynamic environmental conditions. This paper details the architecture and implementation of our AQI forecasting pipeline, emphasizing the synergistic roles of

Apache Airflow and Grafana. By automating the data pipeline and providing intuitive visualizations, our system offers a scalable solution that improves upon traditional AQI monitoring methods. The integration of these technologies not only accelerates data processing but also enhances the overall decision-making process in environmental management.

1. METHODOLOGY

A. System Architecture

The overall system architecture is designed to streamline the processes of data ingestion, processing, storage, and visualization in a real-time environment. The proposed AQI forecasting pipeline integrates four primary components: data collection from the OpenWeather API, workflow orchestration using Apache Airflow, data storage in PostgreSQL, and real-time visualization via Grafana. Data is first retrieved from the OpenWeather API, which supplies environmental parameters such as pollutant concentrations, temperature, humidity, and wind speed. The collected data is then managed and processed by Apache Airflow, which employs Directed Acyclic Graphs (DAGs) to define task dependencies and ensure a robust execution order. After processing, the structured data is stored in a PostgreSQL database that is optimized for efficient querying and integrity. Finally, Grafana is used to present the data in the form of interactive dashboards, providing near real-time updates and visual insights into AQI trends.

The following block diagram illustrates the system architecture:

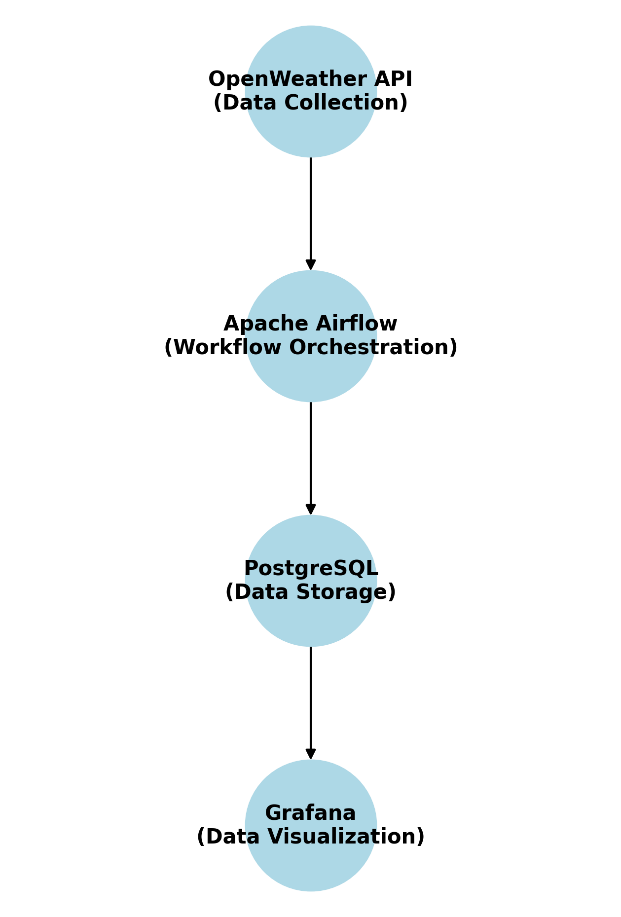


Fig.1. AQI pipeline block diagram

B. Data Collection

Data is collected from the OpenWeather API at regular intervals to capture the most current environmental conditions.

The pipeline is configured to initiate an API call every hour, ensuring near real-time updates. The API returns data in JSON format and provides a comprehensive set of parameters, including pollutant concentrations such as PM2.5, PM10, CO, and NO2, along with meteorological variables like temperature, humidity, and wind speed. API calls are directed to endpoints with necessary query parameters including geographic coordinates, API keys, and units of measurement. Upon receiving the JSON response, the data is parsed and normalized into a structured format that can be efficiently processed by subsequent components in the pipeline.

C. Pipeline Construction

Apache Airflow serves as the backbone for managing the workflow of the data pipeline. Each DAG encapsulates a complete data ingestion cycle, beginning with an Extract Task that initiates an HTTP request to the OpenWeather API. The data retrieved is then passed to a Transform Task, which parses the JSON response and converts it into a structured format suitable for database insertion. The final Load Task is responsible for inserting the processed data into the PostgreSQL database. Airflow’s built-in scheduler triggers these tasks every ten minutes, ensuring that the Transform Task is executed only after a successful extraction and that the Load Task follows a successful transformation. To further enhance the robustness of the pipeline, custom error-handling mechanisms have been implemented. In the event of a failed API call, the system employs a retry mechanism with exponential backoff, attempting up to retries before flagging an error. Detailed logging of all task outputs, errors, and execution details is maintained within a centralized logging system to facilitate continuous monitoring and debugging.

The following diagram shows successful creation of DAG:

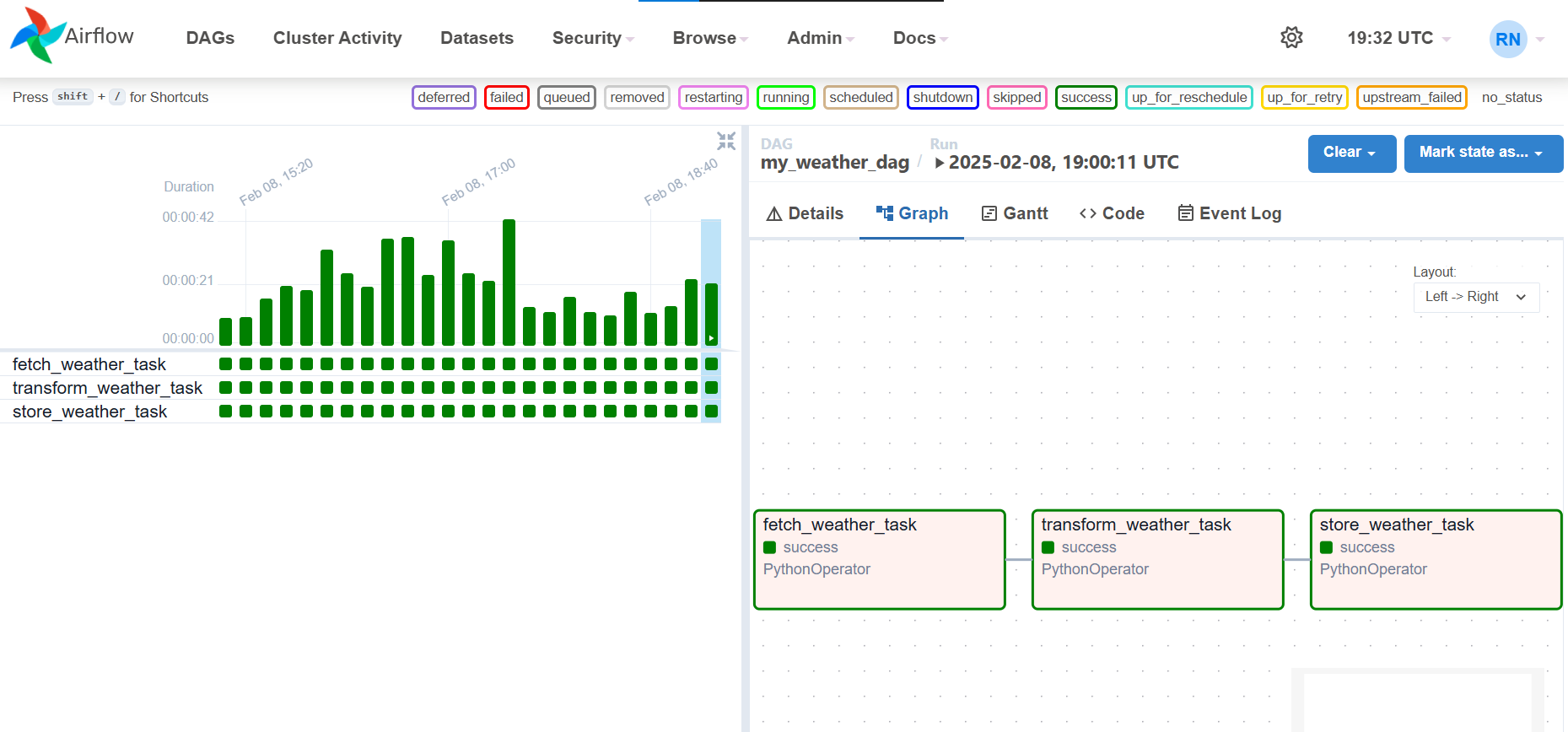


Fig.2. Airflow DAG

D. Data Storage

The PostgreSQL database is designed to store environmental data securely while providing efficient querying capabilities and maintaining data integrity. A dedicated table, named AQI\_data, has been created to capture relevant attributes, including an auto-incremented primary key, a timestamp, a location identifier, pollutant levels (PM2.5, PM10, CO, NO2), and additional environmental parameters such as temperature, humidity, and wind speed. Critical columns such as the timestamp and location identifier are indexed to enhance query performance. Data integrity is preserved through the use of transactions during data insertion,

ensuring atomicity and consistency, while constraints such as NOT NULL and appropriate data type validations are enforced. Furthermore, storage strategies include the partitioning or archiving of older data, thereby facilitating long-term analysis and efficient management of growing datasets.

E. Data Visualization

Grafana is employed to transform the stored data into actionable insights through interactive, real-time dashboards. Grafana is connected to the PostgreSQL database using native data source plugins, enabling seamless querying of the stored data. Custom dashboards have been configured to display various types of visualizations that are most appropriate for environmental data analysis. Time-series graphs, such as line charts, are used to depict AQI trends over time, while gauge panels provide a visual representation of the current AQI levels in relation to established health advisory thresholds. In addition, heatmaps are utilized to illustrate the spatial distribution of pollutants across different geographic regions. The dashboards are configured to refresh every minute, ensuring that users have access to the most current data available. Features such as interactive filters, drill-down options, and real-time alerts are incorporated to enhance user engagement and facilitate deeper analysis.

F. Integration

Seamless integration of all components is achieved through the use of standardized communication protocols and a modular design. Data flows continuously from the OpenWeather API to Apache Airflow, where it is processed and then transferred to PostgreSQL for storage. RESTful API calls ensure reliable data retrieval, while PostgreSQL’s robust connectors facilitate efficient querying by Grafana. The synchronization of tasks is maintained by Airflow’s scheduling and error-handling mechanisms, which guarantee that each stage of the pipeline operates in unison. This modular design also allows for independent scalability of each component, ensuring that the system can accommodate increases in data volume and complexity without compromising performance. Overall, the integration strategy underpins a robust, real-time forecasting system that delivers timely environmental insights for informed decision-making in public health and environmental management.

This comprehensive methodology details the design and implementation of the AQI forecasting pipeline, emphasizing the seamless integration of data collection, processing, storage, and visualization components to achieve a reliable, real-time monitoring system.

1. RESULTS AND DISCUSSION

In this project, we have designed and implemented two Directed Acyclic Graphs (DAGs) to manage data processing and forecasting workflows efficiently.

The first DAG focuses on fetching and processing weather data. It operates in three stages:

1. Fetch\_weather\_task: This task retrieves real-time weather data from the API.
2. Transform\_weather\_task: It cleans and transforms the data into the required format.
3. Store\_weather\_task: The processed data is then stored in the PostgreSQL database for further analysis.

The second DAG is dedicated to performing calculations and forecasting. It consists of the following tasks:

1. Fetch\_data\_task: Fetches stored data for analysis.
2. Transform\_data\_task: Cleans and prepares the data for further operations.
3. Store\_data\_task: Stores the transformed data for reuse.
4. Fetch\_last\_two\_days\_data\_task: Extracts data for the last two days to serve as the input for forecasting models.
5. Apply\_forecasting\_task: Applies ARIMA forecasting techniques to predict future trends.
6. Store\_arima\_forecast\_task: Stores the forecasted results for visualization and reporting.

The key parameters used throughout the workflow include timestamp, longitude, latitude, aqi, pm2\_5, pm10, no2, o3, co, so2, nh3, and no. These parameters ensure a comprehensive understanding of air quality and environmental conditions, which are vital for accurate forecasting. This two-DAG approach modularizes the workflow, enabling better scalability and maintainability. By separating data fetching and processing from calculation and forecasting, we ensure that tasks are decoupled, improving error handling and debugging efficiency. The integration with PostgreSQL provides a reliable backend for managing large datasets, and the ARIMA model's application demonstrates the project's capability to predict air quality indices.

The following diagram shows Airflow AQI Dag:

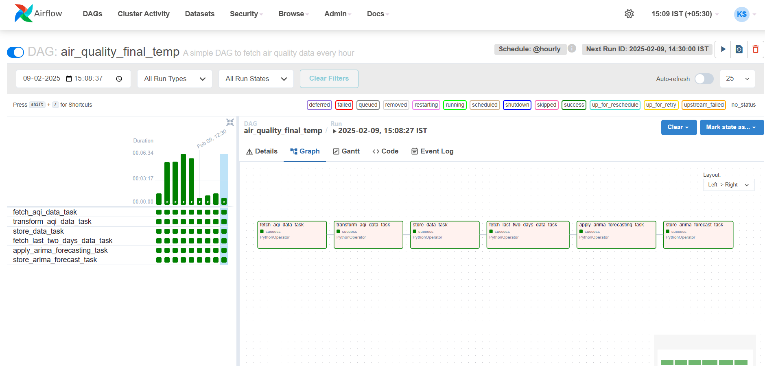


Fig.3. Real time Visualization using Grafana

The following diagram shows real time visualization in Grafana:

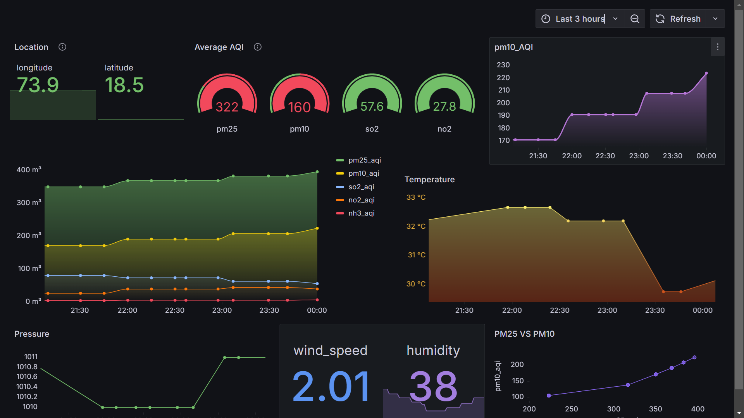


Fig.4. Real time Visualization using Grafana

1. CONCLUSION

This project demonstrates an efficient approach to data processing and forecasting using two DAGs in Apache Airflow. By modularizing workflows for data fetching, transformation, and forecasting, we ensure scalability and maintainability. Key environmental parameters like `aqi`, `pm2\_5`, and `pm10` were successfully utilized for trend prediction using the ARIMA model. The integration of PostgreSQL as a backend and real-time visualization tools enhances the system's reliability