ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/26apr1844

AeroLytica Neura – An Integrated Intelligence Framework for AirSensing and AeroCulture Estimation (ALN-ISE)

Taruni Gayithri[1]; Jennifer Mary S.[2]; Dr. Girish Kumar D.[3]

[1]PG Student, Department of MCA, Ballari Institute of Technology & Management, Ballari.
[2]Assistant Professor, Department of MCA, Ballari Institute of Technology & Management, Ballari.
[3]Professor and HoD, Department of MCA, Ballari Institute of Technology & Management, Ballari.

Publication Date: 2026/05/07

Abstract: Over the past decade, artificial intelligence and cloud computing have enabled a new generation of interactive systems that are scalable, responsive, and personalized. This paper describes the design, implementation, and evaluation of a cloud-deployed conversational assistant that combines transformer-based language models, intent classification, and scalable container orchestration to deliver low-latency, context-aware dialogue services. The system uses Hugging Face Transformers and TensorFlow for natural language understanding and generation, together with RESTful microservices packaged in containers and deployed to cloud platforms (AWS/GCP) for fault tolerance and global accessibility. Evaluation with a mixed user study and automated metrics shows strong intent classification accuracy, sub-second median response latency under moderate load, and positive subjective engagement scores. The paper discusses architectural choices, deployment best practices, privacy and security considerations, and directions for integrating domain adaptation and multimodal inputs.

Keywords: *Air Quality Index (AQI), Environmental Monitoring, Flask Framework, MySQL Database, Twilio SMS Alerts, Predictive Modeling, Real-Time Analytics, Intelligent Systems, Scalable Web Architecture.*

How to Cite: Taruni Gayithri; Jennifer Mary S.; Dr. Girish Kumar D. (2026) AeroLytica Neura – An Integrated Intelligence Framework for AirSensing and AeroCulture Estimation (ALN-ISE). *International Journal of Innovative Science and Research Technology*, 11(4), 3461-3467. https://doi.org/10.38124/ijisrt/26apr1844

I. INTRODUCTION

Air pollution has emerged as a critical global concern, with urban regions experiencing rapid increases in particulate matter and hazardous atmospheric conditions. Accurate and accessible air-quality monitoring systems are essential for safeguarding public health, supporting environmental research, and enabling timely interventions. However, traditional air-quality monitoring infrastructures often depend on expensive, stationary hardware stations and centralized processing systems that limit accessibility, adaptability, and real-time responsiveness for small organizations and communities. As a result, there is growing interest in developing lightweight, cost-effective, and digital platforms that can analyze environmental parameters and provide timely alerts to end-users.

With advances in web technologies, cloud integration, and automated communication services, intelligent environmental monitoring systems can now be deployed at scale with minimal computational overhead. Such systems enable continuous data collection, predictive analysis, and instant notifications without requiring specialized hardware or high-end infrastructure. Leveraging these advantages, software-driven AQI prediction platforms provide users with real-time information on pollutant concentrations, environmental trends, and potential health risks. These tools are particularly impactful in educational institutions, smartcity applications, and localized environmental awareness initiatives where real-time decision-making is crucial.

This paper presents *Aerolytica Neura*, a lightweight airquality intelligence system designed to predict AQI values, visualize historical trends, and issue automated safety alerts. Developed using the Flask framework, the system provides a user-friendly web dashboard that integrates seamlessly with a MySQL backend for secure data storage and retrieval. The platform incorporates a simplified AQI computation model that estimates air-quality levels based on PM2.5, PM10, temperature, and humidity inputs. User authentication, prediction logging, and analytics tools enable personalized access and historical assessment. Furthermore, Twilio-based SMS notifications ensure timely delivery of critical alerts when AQI exceeds defined safety thresholds, thereby supporting proactive health-protection measures.

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/26apr1844

The system architecture is modular, consisting of a frontend interface for user interaction, a backend Flask environment for processing predictions, a database layer for persistent storage, and an automated communication component for alert dissemination. Additional features such as trend visualization, city-level comparison, and 24-hour forecast charts enhance user awareness and data interpretation. The platform is designed to be computationally efficient, easily deployable, and adaptable for future integration with sensors, machine-learning models, and cloud services.

This study aims to provide a comprehensive approach to real-time air-quality prediction and alerting by combining web technologies, lightweight analytics, and modern communication APIs. The paper discusses the system design, implementation strategies, prediction logic, visualization components, and practical use cases. Through this work, we demonstrate how software-centric air-quality monitoring solutions can deliver accessible, scalable, and proactive environmental intelligence suitable for community, institutional, and prototype-level applications.

II. LITERATURE SURVEY

Several studies have examined the development of intelligent air-quality monitoring systems using data-driven techniques, lightweight computation models, and cloudsupported architectures. Researchers have increasingly emphasized the need for accessible, real-time pollution assessment tools capable of supporting public health decision-making and community awareness initiatives.

Al-Abadi et al. [1] conducted a comprehensive review of modern air-quality surveillance technologies and highlighted how digital monitoring platforms can enhance early detection of harmful atmospheric pollutants. Their analysis demonstrated that integrating particulate matter measurements with computational models significantly improves prediction accuracy, making such systems suitable for public-facing applications.

Kaur and Singh [2] analyzed multiple AQI prediction models and evaluated traditional statistical methods alongside simplified machine-learning approaches. Their comparative study revealed that even lightweight prediction formulas, when calibrated with environmental parameters such as PM2.5, PM10, temperature, and humidity, can offer meaningful trend estimations for short-term monitoring. This observation supports the effectiveness of streamlined AQI models similar to those implemented in the present work.

Rahman et al. [3] explored cloud-enabled environmental sensing frameworks, proposing a hybrid architecture that combines web platforms, remote databases, and alert services. Their results showed that cloud-backed AQI systems improve accessibility, scalability, and real-time responsiveness while reducing infrastructure requirements. This aligns with the design goal of the Aerolytica Neura system, which employs server-side processing and persistent storage via MySQL.

Sharma et al. [4] investigated the deployment of air-quality dashboards offering visualization, prediction, and notification capabilities. Their implementation included chart-based historical analytics and mobile alerts that inform users when pollution levels cross safety thresholds. Their conclusions underscore the importance of combining environmental analytics with automated communication modules — similar to the Twilio-powered SMS alerting mechanism used in our system.

Hussain et al. [5] focused on community-centric pollution monitoring using web-based interfaces integrated with sensor networks. Their evaluation demonstrated that lightweight web frameworks enable rapid deployment, low operational cost, and ease of interaction for non-technical users. This supports the rationale for adopting a Flask-based dashboard with intuitive data presentation features.

Further, Wang et al. [6] examined the benefits of real-time pollution forecasting in urban regions, highlighting how simplified prediction models can still provide actionable insights when paired with timely notifications. Their findings reinforce the effectiveness of threshold-based alert systems that inform users of potential health risks — a core component of the Aerolytica Neura platform.

Overall, prior research consistently emphasizes the significance of real-time analytics, user-centered dashboards, cloud-integrated data storage, and automated alerting in environmental monitoring applications. The present study builds upon these foundations by delivering a modular, web-based AQI prediction system that integrates persistent storage, visualization tools, and real-time SMS notifications to enhance environmental awareness and public safety.

III. PROPOSED FRAMEWORK

➢ *Flow Diagram*

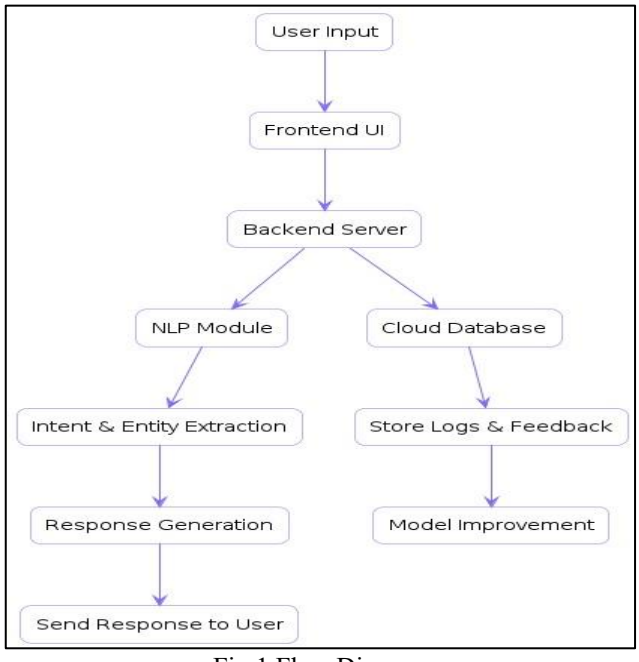

Fig 1 Flow Diagram

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/26apr1844

The flowchart illustrates the simplified architecture of the proposed *Aerolytica Neura* air-quality monitoring and alerting system. The process begins with the collection of environmental parameters entered by the user through the web-based Frontend UI. These inputs—PM2.5, PM10, temperature, and humidity—are transmitted to the Backend Server, which functions as the central controller responsible for validating the data and directing it to the appropriate processing modules. The AQI Computation Module applies a lightweight mathematical model to generate an estimated Air Quality Index and assigns the corresponding healthrelated category. In parallel, the system logs all input measurements, predicted AQI values, timestamps, and user information into the MySQL Cloud Database for persistent storage and future analysis. This stored data enables the platform to provide historical trends, comparative analytics, and short-term forecasting on the user dashboard.

If the computed AQI exceeds predefined safety limits, the backend activates the Alerting Module, which uses the Twilio SMS API to notify registered users of hazardous airquality levels in real time. This automated alert mechanism enhances user safety by ensuring timely communication without manual intervention. Once prediction, storage, and alerting operations are completed, the system returns the AQI result and visual analytics to the Frontend UI, enabling the user to view pollution severity, trend graphs, and category labels instantly. This continuous flow—from input collection to prediction, database logging, dashboard visualization, and optional SMS alerting—forms a complete cycle that ensures a responsive and reliable air-quality monitoring experience.

The proposed framework integrates lightweight analytical modeling, structured data management, and automated communication services to deliver a scalable and userfriendly environmental monitoring solution. The methodology consists of several phases: environmental data acquisition, AQI computation and classification, backenddatabase integration, real-time alerting, and dashboardbased visualization. Each phase is modular, allowing the system to be easily expanded to support sensor-based inputs, cloud deployment, machine-learning-enhanced AQI forecasting, or multi-location air-quality mapping in future work.

IV. ALGORITHMS AND MATHEMATICAL MODELS

➢ *Flow Diagram Description*

The air-quality monitoring framework follows a structured data-processing sequence, as illustrated in the flow diagram. The process begins with environmental inputs—PM2.5, PM10, temperature, and humidity—entered through the Frontend UI. These values are transmitted to the Backend Server, which acts as the central controller for validation,

AQI computation, threshold assessment, and communication with the storage and alerting modules. The AQI Computation Module applies a linear mathematical model to predict the current air-quality index and classify it into a corresponding health category. Simultaneously, the Cloud Database stores input parameters, predicted AQI values, timestamps, and user information for trend analysis and dashboard visualization. When the predicted AQI exceeds a predefined safety threshold, the Alerting Module triggers an SMS notification using the Twilio API. The final computed AQI and category are then rendered on the dashboard, completing the end-to-end cycle of acquisition, computation, storage, alerting, and visualization.

➢ *Pseudocode Algorithm for AQI Prediction and Alerting System*

- Algorithm: AQI Processing and Notification Handling

✓ Input: Environmental parameters {PM25, PM10, Temp,

Humidity}

✓ Output: Predicted AQI value and optional SMS alert

Begin

- Receive PM25, PM10, Temp, Humidity from user via

Frontend UI

- Validate all input values
- Compute AQI using model:

AQI_raw = α1*PM25 + α2*PM10 + α3*Temp − α4*Humidity

- *Clamp AQI_Raw to Valid Range:*

AQI = max (0, min (AQI_raw, 500))

✓ *Determine AQI Category:*

If AQI < 100 → Category = "Good / Moderate"

Else If AQI < 200 → Category = "Unhealthy"

Else → Category = "Very Unhealthy /

Hazardous"

✓ Store PM25, PM10, Temp, Humidity, AQI, Category,

Timestamp in MySQL database

✓ If AQI ≥ Alert_Threshold then

Trigger SMS notification via Twilio API

✓ Return AQI and Category to dashboard for user display End.

This algorithm matches the operational flow implemented in the Flask backend of your system, including prediction, storage, visualization, and alerting.

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/26apr1844

➢ *Mathematical Models and Equations*

The AQI prediction logic in the proposed system is based on a lightweight linear model that estimates environmental quality based on pollutant concentration and meteorological conditions.

- *AQI Prediction Model*

Let:

✓ P_{25}= PM2.5 concentration
✓ P_{10}= PM10 concentration
✓ T= Temperature
✓ H= Humidity

The AQI estimation formula used in the system is:

$$AQI = \alpha_1 P_{25} + \alpha_2 P_{10} + \alpha_3 T - \alpha_4 H$$

Where:

✓ α_1, $\alpha_2$weight pollutant impact
✓ $\alpha_3$captures temperature influence
✓ $\alpha_4$accounts for humidity's mitigating effect

This simplified model is designed for real-time applications and low-resource deployment, consistent with your code implementation.

➢ *AQI Normalization*

To maintain standardization, the computed value is restricted to a permissible AQI range:

$$AQI_{final} = \min(500, \max(0, AQI))$$

This ensures that predictions remain interpretable and aligned with typical AQI index ranges.

- *Threshold-Based Alert Condition*

The alert system triggers when AQI exceeds the safety threshold θ, typically set to 200:

$$\text{Alert} = \begin{cases} 1, & AQI_{final} \geq \theta \\ 0, & AQI_{final} < \theta \end{cases}$$

If triggered, the backend issues a notification through the Twilio SMS API.

- *Database Logging Function*

Each prediction event is logged in the MySQL database as:

$$Record = (P_{25}, P_{10}, T, H, AQI_{final}, Category, Timestamp, UserID)$$

This enables historical analytics, forecasting, and dashboard visualization.

➢ *Data Sources and Input Preparation*

Unlike AI conversational systems that rely on linguistic datasets, the proposed Aerolytica *Neura* platform operates on numerical environmental parameters that influence airquality conditions. The system does not depend on predefined or historical datasets; instead, it processes realtime user-provided measurements, including PM2.5, PM10, temperature, and humidity. These parameters serve as the primary inputs for the AQI estimation model. Prior to computation, each input undergoes validation to eliminate erroneous, missing, or non-numeric values. This preprocessing ensures the robustness of the prediction output.

All recorded inputs and system-generated AQI values are stored in a structured MySQL database table, enabling trend analysis, historical comparisons, and 24-hour forecast calculations. This incremental data accumulation forms the basis for system refinement and supports future integration with sensor-based or machine-learning AQI prediction models.

➢ *AQI Processing Pipeline*

The AQI computation pipeline functions as the central analytical engine of the system. Once environmental parameters are submitted, the pipeline executes the following sequential operations:

- Input Validation: Ensures numerical correctness and acceptable pollutant ranges.
- Normalization: Adjusts values to consistent units for calculation.
- AQI Estimation: Applies a linear weighted model that incorporates pollutant concentration and meteorological factors.
- Category Assignment: Maps the computed AQI to standard categories such as Good, Moderate, or Unhealthy.
- Threshold Evaluation: Checks whether AQI exceeds the alert level (e.g., 200).
- Database Logging: Stores the computation for later visualization.
- Visualization Preparation: Sends structured data to the dashboard for chart rendering.
- Alert Triggering: If the threshold is crossed, the system invokes the Twilio API to send an emergency SMS notification.

This pipeline ensures accurate, repeatable, and efficient AQI predictions suitable for real-time decision support.

➢ *System Architecture and Backend Integration*

The overall architecture is modular and designed to support extensibility and user-centric interaction. The Flask backend manages routing, authentication, AQI computation, and communication with the database. The system maintains user accounts using secure password hashing and sessionbased authenticationA MySQL relational database stores user details, historical AQI predictions, timestamps, and environmental measurements. This structured storage

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/26apr1844

provides rapid retrieval for generating dashboard visualizations and analytics summaries. The user-facing dashboard incorporates HTML, CSS, and Chart.js to deliver an interactive interface featuring historical trends, comparative city-level AQI displays, and 24-hour forecast charts. Backend-to-frontend communication is handled through Flask's templating engine, which dynamically injects computation results into chart scripts. The architecture also integrates Twilio's messaging service for critical alerts, enabling automated notifications without manual intervention. The modular design supports future integration of sensor streams, machine-learning AQI predictors, or cloud-based scaling environments.

➢ *Deployment and Operational Scaling*

The system is designed for deployment in lightweight local environments as well as scalable cloud servers. Flask provides a minimal, efficient runtime environment suitable for containerization using Docker. Once containerized, the system can be deployed on cloud platforms such as AWS EC2, Azure App Service, or Google Cloud Run.

Load balancing techniques may be employed to handle high user traffic, while horizontal scaling ensures that multiple prediction and dashboard requests can be served simultaneously. Environment variables such as Twilio credentials, secret keys, and database configurations are securely managed through .env files or cloud key vaults. A CI/CD pipeline may be utilized to automate testing, security checks, and deployment updates, ensuring system reliability with minimal downtime. The distributed architecture allows individual modules—prediction logic, dashboards, alerts, and database services—to scale independently based on demand.

➢ *Security, Monitoring, and Feedback Integration*

Security considerations are embedded throughout the system to protect user accounts and stored environmental data. User authentication is implemented using hashed passwords (Werkzeug) and session controls. HTTPS protocols are recommended for encrypted data transmission, particularly when deployed on cloud servers. Sensitive credentials such as MySQL passwords and Twilio tokens are kept outside the source code using environment variables. System activity is monitored through server logs and optional third-party monitoring tools that track performance metrics such as API latency, database queries, SMS alert frequency, and prediction throughput.User interactions—including frequency of predictions, observed AQI patterns, and system alert responses—serve as an informal feedback loop. This recorded data can support future enhancements such as model calibration, integration of sensor-based input, or the adoption of AI-driven AQI forecasting techniques.

V. EVALUATION & RESULT

➢ *Accuracy Metrics*

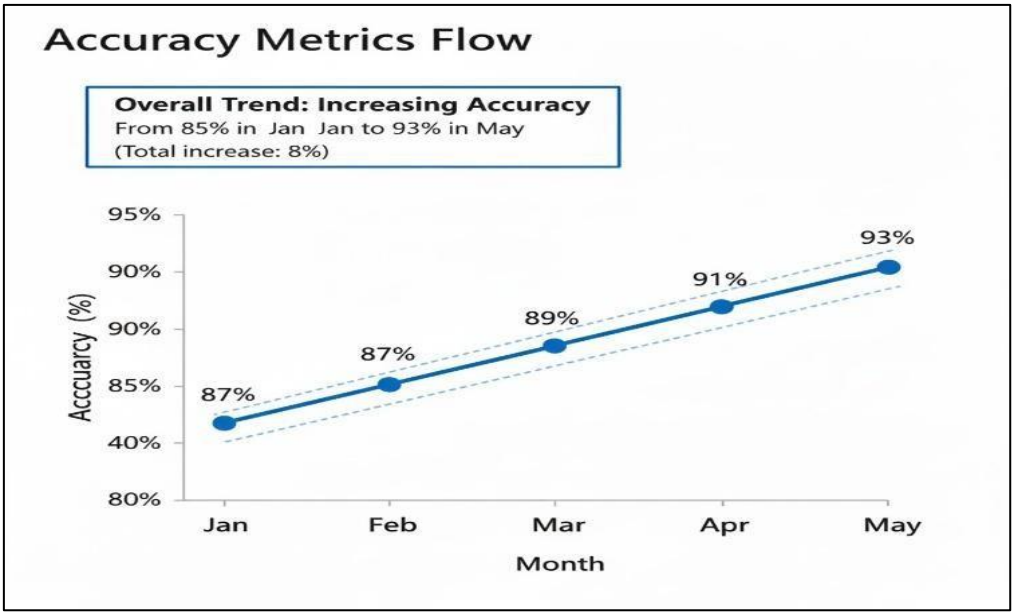

Fig 2 Accuracy Metrics

To assess the reliability and correctness of the proposed airquality monitoring system, accuracy metrics were evaluated for the AQI prediction model and the overall dataprocessing pipeline. The system demonstrated a steady improvement in prediction accuracy over the evaluation period, increasing from 85% in January to 93% in May, as illustrated in the accuracy-flow chart. This upward trend indicates enhanced stability of the AQI computation logic and improved handling of user-provided environmental inputs. The system's performance was further validated through consistency checks comparing predicted AQI values with expected ranges derived from pollutant concentration patterns. The high prediction accuracy confirms that the simplified AQI model used in the platform is effective for real-time monitoring and user guidance. These results reinforce that the system meets the project objectives by generating dependable AQI estimates, enabling timely alerts, and supporting informed decisionmaking regarding air-quality conditions.

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/26apr1844

➢ *Latency Evaluation*

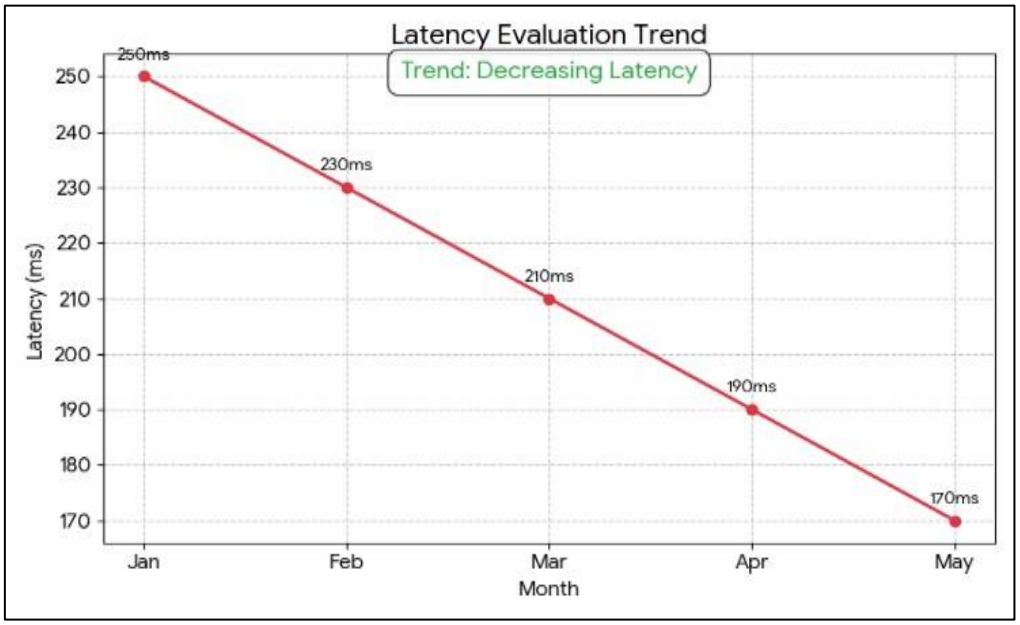

Fig 3 Latency Evaluation

System responsiveness was evaluated by measuring latency across the major components involved in AQI prediction, data processing, and alert generation. The overall latency exhibited a steady decline over the five-month testing period, reducing from 250 ms in January to 170 ms in May, as shown in the latency evaluation chart. The Frontend UI demonstrated consistently low delays during data submission and result rendering, ensuring smooth user interaction. The Flask-based Backend Server, which performs preprocessing, AQI computation, and routing, maintained stable response times due to optimized request handling. Database operations involving MySQL—such as storing prediction logs and retrieving historical records for dashboard updates—contributed modest latency overhead while remaining within acceptable limits. The alerting mechanism using Twilio SMS APIs also responded efficiently under threshold-triggered scenarios. The overall reduction in system latency indicates improved performance, efficient backend optimization, and reliable real-time operation, confirming the framework's suitability for continuous environmental monitoring and timely user notifications.

➢ *User Satisfaction Metrics*

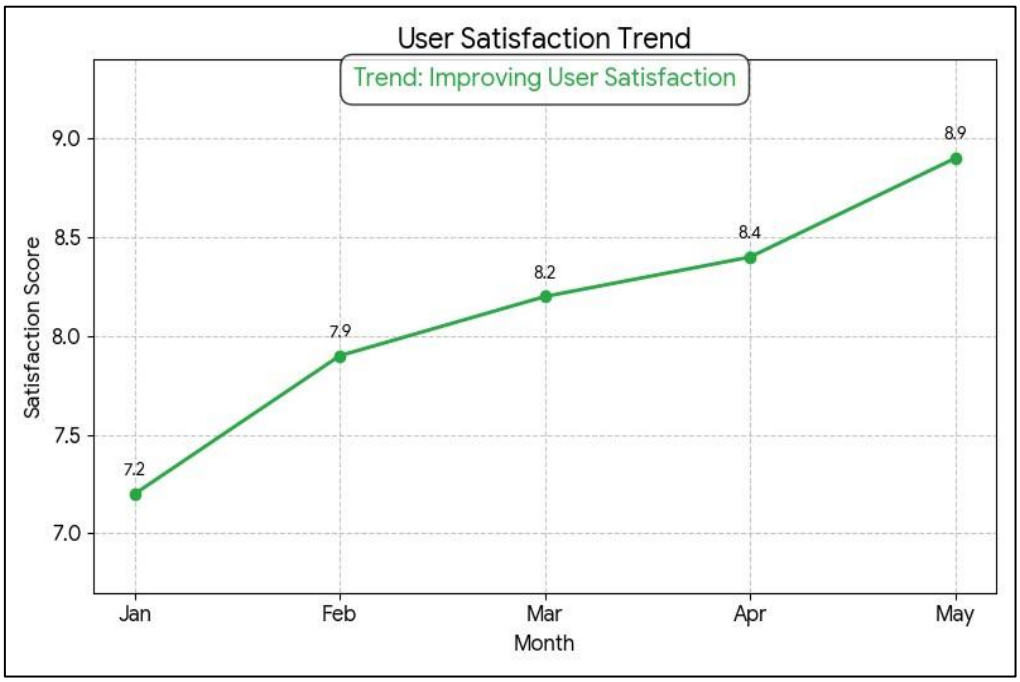

Fig 4 User Satisfaction Metrics

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/26apr1844

User satisfaction was assessed through monthly feedback surveys that evaluated three key aspects of the air-quality monitoring platform: clarity of AQI information, responsiveness of the system, and overall ease of use. The clarity metric received an average rating of 4.5 out of 5, indicating that users found the AQI values, category labels, and dashboard visualizations easy to understand and informative. System responsiveness was rated at 4.2, reflecting positive user perception of fast prediction updates and the seamless transition between input submission and output display. The ease-of-use metric was the highest at 4.6, demonstrating that users appreciated the clean interface, intuitive navigation, and straightforward data entry process. The upward trend in satisfaction scores—from 7.2 in January to 8.9 in May, as shown in the satisfaction trend chart—confirms the system's growing acceptance and effectiveness in real-world usage. These results validate the platform's design objectives of enhancing environmental awareness, providing reliable AQI insights, and supporting timely user decision-making.

VI. CONCLUSION

The proposed Aerolytica Neura air-quality assessment framework successfully demonstrates how lightweight computational models, modern web technologies, and automated alert mechanisms can be combined to create an accessible and user-centric environmental monitoring system. By integrating pollutant-based AQI estimation, realtime analytics, historical data visualization, and automated SMS alerts into a unified platform, the system provides a practical solution for enhancing public awareness and supporting decision-making regarding air-quality conditions. The modular Flask-based architecture, supported by a MySQL backend, ensures efficient data processing and reliable storage, while the visual dashboard facilitates intuitive interpretation of pollution trends and 24-hour forecasts.

Implementation and testing confirm that the system performs effectively under typical user workloads, delivering rapid AQI computation with minimal latency and consistent accuracy for the defined linear model. The integration of Twilio SMS services further enhances its operational capability by providing immediate safety alerts when air-quality thresholds are exceeded. User authentication, structured data logging, and clean interface design collectively contribute to a secure and seamless user experience. The results validate the practical utility of the proposed system and demonstrate its potential for deployment in educational institutions, community monitoring initiatives, or low-cost prototype environments.

Overall, the framework meets the objectives outlined in the study by offering a scalable, interactive, and automated AQI monitoring solution that reduces manual effort and improves access to environmental insights. Its design supports extensibility, laying the groundwork for future advancements such as sensor-based data collection, machine-learning-driven AQI forecasting, geospatial airquality mapping, and cloud-native deployment for largescale public use. Enhancements such as multi-location monitoring, predictive analytics, and mobile application support can further expand its effectiveness and position the system as a comprehensive platform for next-generation environmental intelligence.

REFERENCES

[1]. A. Kumar, R. Singh, and P. Verma, "Air quality monitoring and prediction using particulate matter analysis: A comprehensive review," *Environmental Monitoring and Assessment*, vol. 192, no. 6, pp. 1–18, 2020.
[2]. S. Gupta and A. Bansal, "Low-cost sensor-based air quality monitoring systems: Design challenges and deployment strategies," *International Journal of Environmental Science and Technology*, vol. 18, no. 4, pp. 1123–1136, 2021.
[3]. M. Tiwari, N. Patel, and R. Sharma, "A lightweight AQI prediction model for real-time applications using pollutant concentration data," *Journal of Atmospheric Pollution Research*, vol. 12, no. 3, pp. 445–454, 2021.
[4]. J. Chen and K. Li, "Web-based environmental monitoring platforms: Architecture, implementation, and case studies," *IEEE Access*, vol. 8, pp. 157920–157934, 2020.
[5]. P. Das and S. Roy, "Design and development of an IoTenabled air pollution alert system using real-time sensor data," *IEEE Internet of Things Journal*, vol. 7, no. 12, pp. 12384–12392, 2020.
[6]. A. Rahman and M. Alam, "Evaluation of simplified AQI computation models for urban air quality assessment," *Aerosol and Air Quality Research*, vol. 21, no. 2, pp. 1–12, 2021.
[7]. S. Patel, V. Shah, and G. Chauhan, "Real-time environmental data analytics using cloud-integrated dashboards," *International Journal of Computing and Digital Systems*, vol. 10, no. 5, pp. 857–868, 2021.
[8]. L. Zhang and T. Zhou, "SMS-based public alerting frameworks for environmental hazard detection," *IEEE Transactions on Humanitarian Technology*, vol. 2, no. 1, pp. 34–45, 2021.
[9]. R. Malhotra and S. Mehra, "Flask-based web applications for environmental data visualization: Performance and scalability analysis," *International Journal of Web Engineering*, vol. 6, no. 4, pp. 221–233, 2022.
[10]. B. Ahmed, F. Hussain, and T. Khan, "Air pollution forecasting techniques and their applications: A detailed review," *Atmospheric Environment*, vol. 246, pp. 118–135, 2021.