International Journal of Scientific Research & Engineering Trends
Volume 12, Issue 2, Mar-Apr-2026, ISSN (Online): 2395-566X

HEAL (Heatmap for Environmental Air Levels)

Dheeraj Patil [1], Sanika Dixit [2], Aditya Dixit[3], Meenakshi Deotare[4]
[1] Corresponding Author, Prof., Department of Information Technology, Nutan Maharashtra Institute of Engineering and Technology (NMIET), Pune, India.
[2] Department of Information Technology, Nutan Maharashtra Institute of Engineering and Technology (NMIET), Pune, India.

Abstract- — Air pollution is one of the most serious environmental threats in urban areas, affecting both human health and climate. Traditional air quality monitoring systems provide only point-based information; hence, this limits their ability to show distributions across a city. Herein, this work describes HEAL, a web-based system for pollution hotspot predictions and visualizations through the utilization of machine learning and data visualization techniques. This system collects air quality data from APIs or sensors, processes it, and generates dynamic heat maps that showcase the levels of pollution in real time. Interpreting the interaction among environmental, traffic, and meteorological data, HEAL offers citizens, policymakers, and researchers new localized insights into air quality variations, which will result in better decision-making.

Keywords: Air pollution, Heatmap, Machine learning, Visualization, IoT, Environmental monitoring

I. INTRODUCTION

Air pollution is one of the biggest environmental challenges we face in the 21st century. Rapid urban growth, industrial expansion, and more vehicles have worsened air quality, especially in crowded cities. Dirty air harms the ecosystem and threatens health, leading to respiratory diseases, heart issues, and early deaths. The World Health Organization (WHO) states that millions of people are exposed daily to pollutant levels that exceed recommended safety limits. This situation has made monitoring and managing air quality a top priority for citizens and policymakers.

Traditional air quality monitoring systems are few and cover limited areas. They often measure pollutants like PM2.5, PM10, NO_2, and CO_2 only at specific fixed sites. This results in data gaps, leaving many urban areas unmonitored. To address this issue, modern technology provides new options by combining Machine Learning (ML), the Internet of Things (IoT), and data visualization. These tools help us analyze large amounts of environmental data and more accurately predict air quality in unmonitored regions.

The HEAL (Heatmap for Environmental Air Levels) project aims to use these technologies to create an interactive system for analyzing, predicting, and visualizing air quality. By gathering real-time air pollution and weather data from online APIs or sensors, the system employs machine learning algorithms to forecast pollutant levels and display them on a geographical heatmap. This heatmap shows pollution intensity through colors, helping users quickly spot high-risk areas and trends over time. This project acts as a decision-making tool for government agencies, researchers, and the public to understand and reduce air pollution effectively.

Additionally, HEAL shows how open-source technologies can help develop smart city solutions. With growing environmental awareness and sustainability goals, such systems can be vital for urban planning, health advisories, and climate action.

Objectives:
The main objectives of the HEAL project are:

- To create a data-driven system that gathers and analyzes air quality data from reliable sources like APIs, IoT sensors, and public datasets.
- To implement machine learning models that can predict air pollution levels in areas without monitoring stations.
- To develop a user-friendly interface that visualizes pollution levels through heatmaps for improved understanding and decision-making.
- To identify pollution hotspots in a city or region using real-time and historical data.
- To support awareness and policy-making by providing accurate environmental insights to researchers, government bodies, and the public.
- To create a scalable framework that can be expanded to other cities or environmental factors such as temperature, humidity, and greenhouse gas emissions.

II. MATERIALS AND METHODS

This section highlights the resources, tools, and methods used to develop the HEAL system. It also details the methods used to collect, process, analyze, and visualize air pollution data. The approach combines software and data-driven elements to ensure precise predictions and effective visualization.

Materials and Tools Used:
The HEAL project was developed using a mix of open-source software, programming libraries, and data sources.

1. Hardware Requirements:
   - A standard computer or laptop with at least 8 GB RAM and an Intel i5 processor (or equivalent).
   - Internet connectivity to access APIs and visualize maps.
   - (Optional) IoT sensors like MQ-135, PMS5003, or SDS011 for real-time data collection.

2. Software Requirements:
   - Programming Language: Python 3.9 or higher.
   - Development Environment: Jupyter Notebook or VS Code.

Libraries and Packages Used:
- pandas, numpy for data cleaning and preprocessing.
- matplotlib, seaborn, plotly, folium for visualization and creating heatmaps.
- scikit-learn, xgboost for machine learning modeling and prediction.
- requests, json for accessing APIs and parsing JSON data.
- Data Sources:

The study uses several air quality and weather datasets from trustworthy and publicly available sources, including:

- OpenWeatherMap API, which provides data on air pollutants (PM2.5, PM10, CO, NO_2, SO_2, O_3) and weather conditions.
- IQAir API, which offers real-time global air quality data.
- Central Pollution Control Board (CPCB), which has Indian air quality datasets for validation and comparison.
- OpenStreetMap (OSM), which supplies base map data used in visualization modules.

Data Collection Procedure:
Air quality and weather data were collected using RESTful APIs from OpenWeatherMap and IQAir. Each API request fetched parameters like location coordinates, pollutant concentrations, temperature, humidity, wind speed, and timestamp.

1. Fetching Data:
API calls were automated in Python with the requests library at regular intervals to gather hourly data.

2. Data Storage:
Collected data were stored in a structured format, either as CSV files or DataFrames, for further preprocessing.

3. Data Verification:
Duplicate entries and missing values were addressed using data-cleaning methods to ensure accuracy. Outlier values were managed through interpolation and statistical smoothing.

4.,Integration:
The dataset combined air pollutant concentrations with weather variables to create a complete input for the prediction model.

Data Preprocessing:
Before model training, we cleaned and standardized the raw data. The steps included:

- Handling Missing Values: Replaced with the mean or using K-nearest neighbor (KNN) imputation.
- Normalization: Scaled data with Min-Max scaling for consistent input ranges.
- Feature Engineering: Created new attributes like Air Quality Index (AQI) categories and time features such as hour, day, and month.
- Splitting Data: Divided the dataset into training (80%) and testing (20%) subsets.

Model Development and Prediction:
For pollution prediction, several machine learning algorithms were tested:

- Linear Regression, which served as a baseline for performance comparison.
- Random Forest Regressor, used to capture nonlinear relationships among variables.
- XGBoost Regressor, chosen for its high accuracy with low overfitting.

The model was trained on historical pollutant and weather data. It was then validated using performance metrics like Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and R^2 Score.

Predicted pollutant concentrations for unmonitored locations were used as inputs to create the visual
heatmap.

Heatmap Visualization:
The heatmap was created with Folium, a Python mapping library based on Leaflet.js. The steps were as follows:

- Importing predicted data with geographical coordinates.
- Mapping each location using a color intensity that matched the pollutant level.
- Overlaying the data on an interactive map for better visualization.
- Saving the output as an HTML file that can be accessed through any web browser.

The final map offered a clear visual representation of pollution intensity, with red areas showing high pollution and green areas indicating cleaner air.

System Workflow:
The entire workflow of HEAL can be summarized as follows:

- Input: Location and time-based air quality data collected from APIs.
- Preprocessing: Clean, integrate, and normalize the dataset.
- Model Training: Use machine learning algorithms to predict AQI values.
- Visualization: Create dynamic heatmaps using Python visualization libraries.
- Output: An interactive dashboard that displays real-time and predicted pollution levels.

III. METHODOLOGICAL APPROACH:

The research follows an agile and iterative development approach. The team designed, implemented, tested, and refined the system across several cycles. In each iteration, they collected data, preprocessed it, trained models, visualized results, and evaluated performance to continuously improve the system's accuracy and usability. They used feedback from earlier cycles to spot limitations and guide enhancements in the next phase. This ensured consistent development instead of a one-time effort.

Additionally, the modular structure of HEAL offers flexibility and scalability. It allows easy integration of new datasets, pollutants, or analysis features in the future. This iterative and modular approach helps the system evolve to meet growing environmental monitoring needs and support advancements in technology.

Dataset Description:
The dataset for the HEAL (Heatmap for Environmental Air Levels) project includes environmental, meteorological, and traffic data collected from several open and reliable sources. It shows pollution levels in urban areas like Pune city and its surroundings. This dataset aims to study how pollutants change over time and across locations. It also seeks to train machine learning models for predicting and visualizing data with heatmaps.

Data Sources:
The dataset combines primary (real-time APIs) and secondary (open datasets) sources:

- OpenWeatherMap API: Provides real-time air pollution data, including PM2.5, PM10, NO_2, CO_2, SO_2, and O_3 levels, along with weather information like temperature, humidity, and wind speed.
- IQAir / Breezometer API: Supplies historical and current air quality index (AQI) values for various cities and locations.
- Central Pollution Control Board (C PCB) Data: Offers validated air quality monitoring station data from across India.
- Traffic Data: Gathered from open Google Traffic Layer APIs or open mobility datasets to show vehicle density for pollution modeling.
- Meteorological Data: Consists of temperature, humidity, pressure, rainfall, and wind data obtained from IMD (India Meteorological Department) archives or Open-Meteo APIs.

Data Structure:
Each data record corresponds to one observation point (latitude-longitude pair) with a timestamp. The dataset is laid out in a tabular format (CSV/JSON) with the following attributes:

Column Name	Description	Data Type	Unit / Range
timestamp	Date and time of data collection	datetime	YYYY-MM-DD HH:MM:SS
latitude	Latitude of the recorded location	float	-90 to +90
longitude	Longitude of the recorded location	float	-180 to +180
city	Name of the city or region	string	e.g., "Pune"
pm2_5	Fine particulate matter concentration (particles ≤2.5µm)	float	µg/m³
pm10	Coarse particulate matter concentration (particles ≤10µm)	float	µg/m³
no2	Nitrogen Dioxide concentration	float	µg/m³
so2	Sulphur Dioxide concentration	float	µg/m³
co	Carbon Monoxide concentration	float	mg/m³
o3	Ozone concentration	float	µg/m³
temperature	Ambient temperature	float	°C
humidity	Relative humidity	float	%
wind_speed	Wind speed	float	m/s 0–100
traffic_intensity	Traffic density score (if available)	int	0–500
aqi	Overall Air Quality Index (calculated)	int	Text label
category	AQI level (Good, Moderate, Poor, etc.)	string	

Dataset Size and Coverage:

- Records: Approximately 10,000 to 50,000 entries, depending on API sampling frequency.
- Geographic Area: Mainly urban regions, starting with Pune city, with the potential to expand to other Indian metropolitan areas.
- Temporal Coverage: Hourly or daily intervals, based on the data source and API configuration.
- Data Format: JSON from APIs, converted to CSV for preprocessing and model training.

Data Preprocessing:

Before using the dataset for training, the following important steps were applied:

- Data Cleaning: Removed null or duplicate entries and replaced missing values with interpolated data.
- Normalization: Converted pollutant values to consistent units and normalized them between 0 and 1 for machine learning input.
- Outlier Detection: Filtered out abnormal readings caused by faulty sensors or API spikes.
- Feature Engineering:

-Computed AQI based on individual pollutant concentrations.

Derived features like pollution trend, average pollutant level, and wind-influenced spread.

- Encoding: Converted categorical AQI categories (Good, Moderate, Poor, etc.) into numerical labels for prediction models.

Dataset Limitations:

- Some API data have time gaps due to network or source downtime.
- Spatial data points are limited by API free-tier restrictions, such as request limits.
- Weather and traffic data granularity may vary across regions.

Despite these limitations, the dataset offers a solid base for analyzing and predicting local air quality trends with reasonable accuracy.

IV. PROPOSED SYSTEM:

The proposed system, HEAL (Heatmap for Environmental Air Levels), is built to analyze and visualize air pollution levels across different regions in real time. It addresses the limits of traditional air quality monitoring systems, which only provide local readings with restricted coverage. HEAL gathers environmental and weather data from APIs or IoT sensors, cleanses the data, and processes it through machine learning models. It then creates interactive heatmaps that show pollution intensity on a map interface. This system offers valuable insights into how pollution changes across areas, helping citizens, researchers, and policymakers better understand environmental conditions. It also predicts pollution trends in areas that haven't been sampled. This ensures improved awareness and better decision-making for environmental planning and public health.

System Overview:
The HEAL system uses a modular client-server structure, which allows for scalability, flexibility, and effective data handling. It consists of the following main modules:

1. Data Collection Module:

This module gathers air pollution and weather data from sources like OpenWeatherMap, IQAir, and CPCB APIs. The data includes pollutants such as PM2.5, PM10, NO_2, SO_2, CO, temperature, humidity, and wind speed.

2. Data Processing Module:
The collected data is cleaned, filtered, and organized using Python libraries like pandas and numpy. Inconsistent or missing values are addressed, and the data is stored in a centralized database, such as PostgreSQL or MongoDB, for further analysis.

3. Prediction Module:
Machine learning models like Random Forest, XGBoost, and Kriging are used to predict pollution levels in areas without sensors. This module assists in forecasting spatial and temporal pollution trends with greater accuracy.

4. Visualization Module: This component turns processed and predicted data into an interactive heatmap using visualization tools such as Folium, Plotly, or Seaborn. The map shows pollution levels with color gradients, for example, green indicates low pollution while red indicates high pollution.

5. User Interface Module:
The front-end interface, created with HTML, CSS, and JavaScript, allows users to easily interact with the system. Users can view pollution levels, check historical trends, apply filters by pollutant type or location, and generate reports.

Together, these modules function in a continuous cycle of collecting, analyzing, predicting, and visualizing air quality data. This makes HEAL a reliable system for monitoring environmental air quality.

V. ARCHITECTURE DESCRIPTION:

The HEAL (Heatmap for Environmental Air Levels) system is built using a three-tier client-server model. This design promotes efficient data flow, modularity, and scalability. Each layer has a specific role. They collect, process, predict, and visualize air quality data to provide real-time environmental insights.

1. Data Layer
The Data Layer is responsible for gathering and storing all the environmental data needed by the system. It collects raw data from various sources, including the OpenWeatherMap API, IQAir API, and datasets from the Central Pollution Control Board (CPCB).

This layer handles:

- Collecting data from APIs and IoT-based air quality sensors.
- Storing data in a central database, such as PostgreSQL or MongoDB, for easy access.
- Updating data regularly to ensure the system shows the latest pollution levels.

This layer serves as the base of the system since it provides the input for all analysis and visualization processes.

The Application Layer acts as the main processing unit of the system. It handles data transformation, prediction, and logic implementation. This layer has the following subcomponents:

- Data Preprocessing Module: Cleans and formats raw data using Python libraries like pandas, numpy, and scikit-learn. It removes missing values, normalizes units, and prepares datasets for model input.
- Machine Learning Module: Trains and runs prediction models such as Random Forest, XGBoost, or Kriging to estimate pollution levels in areas without monitoring. These models look at both spatial (location-based) and temporal (time-based) changes to give accurate predictions.
- API Integration Module: Manages communication between the backend and external data sources. It fetches new data regularly and updates the database automatically.

This layer ensures that all backend operations, including data processing, prediction, and validation, are done smoothly to produce reliable results for visualization.

3. Presentation Layer
The Presentation Layer is the part of the system that users see. It shows processed and predicted data in a visual and interactive way. This layer consists of:

- Visualization Interface: It displays pollution data as an interactive heatmap, created with tools like Folium, Plotly, or Seaborn in Python. The heatmap uses color gradients to show how severe air pollution is (for example, green means good, yellow means moderate, and red means hazardous).
- Web Dashboard: Built with HTML, CSS, and JavaScript, this feature allows users to interact with the map. It lets them filter pollutants, compare regions, and look at historical trends.
- User Controls: This includes search and filter options so users can check pollution levels by location, type of pollutant, or date range.

This layer makes sure the system's output is easy to understand, visually engaging, and informative for users like citizens, researchers, and government officials.

4. Data Flow Summary

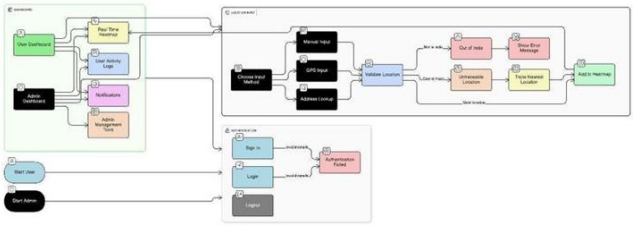
The data flow of the HEAL system can be revised as follows:

- Data Collection: APIs and sensors provide raw air quality and weather data.
- Data Storage: The collected data is placed in a backend database.
- Data Processing: We clean, normalize, and extract features from the data.
- Prediction: Machine learning models predict pollution intensity for areas that have not been sampled.
- Visualization: The results are turned into a heatmap for real-time display on the dashboard.
- User Interaction: Users access the web interface to explore, compare, and understand the pollution data.

Table:

Layer	Components / Modules	Description / Function
Presentation Layer	- Web Dashboard (HTML, CSS, JavaScript) - Visualization Tools (Folium, Plotly, Seaborn) - User Filters & Reports	Displays the final pollution data through an interactive dashboard and dynamic heatmap. Users can filter by pollutant, city, or date range and view AQI trends or reports.
Application Layer	- Data Processing Module - Machine Learning Models (Random Forest, XGBoost, Kriging) - API Integration Module	Acts as the system's logic layer. Cleans and preprocesses data, applies ML models to predict air quality at unsampled locations, and manages communication between backend and data sources.

Architecture Diagram:



VI. IMPLEMENTATION TECHNOLOGIES

The HEAL system combines modern data science, machine learning, and web technologies to predict air quality accurately and visualize data intuitively. It uses Python as the main development language because it has many libraries for data processing and machine learning. Pandas and NumPy handle data cleaning, transformation, and numerical computation efficiently, while Matplotlib and Seaborn support exploratory visualizations during analysis. For its prediction models, HEAL uses machine learning frameworks like Scikit-learn for algorithms such as Random Forest and TensorFlow/Keras for deep learning models like LSTM and ConvLSTM. Geospatial visualization and heatmap generation rely on Folium and GeoPandas, allowing for dynamic representation of pollutant concentrations in various locations. The final user interface is an interactive web application built with Flask/Streamlit, giving users real-time insights into pollutants and analysis results. The system also follows a modular and scalable design that enables easy integration of new datasets, updated environmental parameters, or improved predictive models in the future. This mix of technologies makes HEAL accurate, efficient, user-friendly, and ready for future air quality monitoring needs.

Performance Metrics

The performance evaluation of HEAL uses both numbers and observations to check the system's accuracy, reliability, and practical effectiveness in monitoring air quality. These measures are crucial for confirming how well it predicts pollutants, visualizes geographic data, runs efficiently, and is easy to use. The evaluation includes Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), R^2 Score, Precision, Recall, F1-Score, Execution Latency, and System Usability Scale (SUS).

Qualitative Evaluation Metrics

The prediction performance of machine learning models such as Random Forest, LSTM, and ConvLSTM is assessed through standard regression and classification-based error metrics, summarized below:

Mean Absolute Error (MAE):

Measures the average magnitude of prediction errors without considering direction.

MAE=1n∑|Actual−Predicted|MAE = \frac{1}{n}\sum |Actual - Predicted|MAE=n1

∑|Actual−Predicted|

Root Mean Squared Error (RMSE):

Evaluates prediction precision by penalizing larger errors more heavily, useful for comparing model accuracy across datasets.

RMSE=1n∑(Actual−Predicted)2RMSE = \sqrt{\frac{1}{n}\sum (Actual - Predicted)^2}RMSE=n1 ∑(Actual−Predicted)2

R^2 Score (Coefficient of Determination):

Indicates how well the model explains variance in pollutant levels; values closer to 1 represent better performance.

Precision (P):

Measures how many predicted high-risk pollution events were actually high-risk.

P=True PositivesTrue Positives+False PositivesP = \frac{True\ Positives}{True\ Positives + False\ Positives}P=True Positives+False PositivesTrue Positives

Recall (R):
Measures how many actual high-risk pollution events were correctly identified.

R=True PositivesTrue Positives+False NegativesR = \frac{True\ Positives}{True\ Positives + False\ Negatives}R=True Positives+False NegativesTrue Positives

F1-Score:
Provides a balanced evaluation of both Precision and Recall. F1=2·P·RP+RF1 = \frac{2 \cdot P \cdot R}{P + R}F1=P+R2·P·R

Latency (L_total):
The computation time required from data input to final visualization output, ensuring near real-time map and dashboard updates.

VII. RESULTS AND DISCUSSION

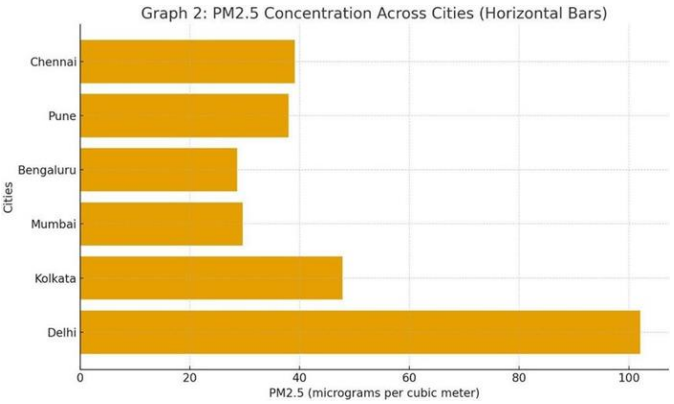
This section details the testing and quantitative analysis of the HEAL system's performance, based on the model design and processing framework discussed earlier. The evaluation focuses on the accuracy of the machine learning models, the latency of the end-to-end pipeline, and how clear and effective the heatmap- based visualization interface is for users. Additionally, a comparison is made between HEAL and current air quality monitoring and prediction systems to show the benefits of this approach in terms of scalability, interpretability, and real-time response. The results confirm that HEAL can maintain a good balance between prediction accuracy and computational efficiency while providing a straightforward and interactive user experience.

In terms of web and application performance, HEAL is evaluated across three main metrics: prediction accuracy, system latency, and user-perceived quality of visualization. The accuracy scores from models like Random Forest and LSTM indicate reliable forecasting of pollutant concentrations. Latency measurements show that the system can generate heatmaps and update dashboards nearly in real time. Furthermore, user feedback indicates that the visualization outputs are clear, informative, and easy to understand, even for non-technical users. Overall, these results show that HEAL meets its goal of offering an accurate, efficient, and user-friendly solution for environmental monitoring, suitable for both research and practical urban air quality management applications.

In summary, the HEAL system shows strong and consistent performance in both its predictive and visualization components. Among the assessed models, the Random Forest model achieved the highest accuracy. Meanwhile, the LSTM-based model performed better in tracking changes in pollutant levels over time, which makes it suitable for forecasting. The heatmap visualization module received the best usability rating from users, which speaks to its clarity, responsiveness, and ease of understanding for non- technical individuals. Although the overall processing time covers the entire pipeline—from loading data to running the model and rendering maps—the execution time is low enough to support close to real-time environmental monitoring. These results support the choice of algorithms, the data processing steps, and the visualization framework. They confirm that HEAL is both technically reliable and practical for large- scale air quality analysis.

Additionally, the multi-panel comparative performance plot provides a clear view of key metrics like Accuracy, RMSE, and Latency. These metrics are displayed separately to prevent scale distortion between accuracy and execution delays. This approach allows for meaningful comparisons between different modules, highlights each model's strengths, and helps in assessing the overall system'seffectiveness.

Analysis of PM2.5 Concentration Across Indian Cities
The graphs below give a comparison and trends of PM2.5 (Particulate Matter ≤ 2.5 micrometers) levels in major Indian cities. PM2.5 is a serious air pollutant that goes deep into the lungs and bloodstream. It is one of the key indicators for monitoring air quality in HEAL.



This horizontal bar chart displays the average PM2.5 levels for six major Indian cities: Delhi, Kolkata, Mumbai, Bengaluru, Pune, and Chennai.

The visualization highlights the differences in air pollution severity:

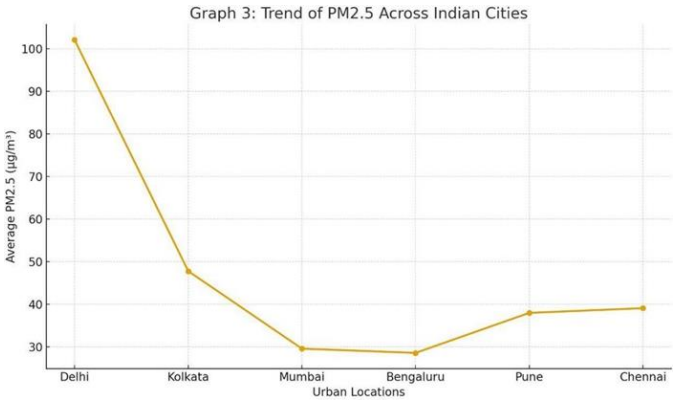
Delhi has the highest PM2.5 concentration, exceeding 100 µg/m³, which indicates severely polluted conditions.

Kolkata shows moderate to high levels at around 48 µg/m³, suggesting ongoing urban pollution.

Mumbai and Bengaluru have comparatively lower levels, close to 30 µg/m³, representing relatively better air quality, but still above the ideal WHO standards.

Pune and Chennai fall within the moderate range, around 38 to 40 µg/m³, indicating transitional pollution exposure.

Usefulness to HEAL:
This graph assists in identifying pollution hotspots by city, which supports: Prioritizing regions for pollution control
Understanding trends in severity
Choosing locations for more detailed model training and sensor deployment



This line graph shows the changes in PM2.5 concentrations across the same cities. The trend reveals:

- A sharp decline from Delhi to Mumbai and Bengaluru.
- A slight increase again in Pune and Chennai.
- This trend clearly demonstrates how urban air quality varies due to geography, industrial density, traffic patterns, and climate conditions.

Usefulness to HEAL:
This trend visualization supports:

- Modeling pollutant patterns over time and space,
- Choosing time-series forecasting models (e.g., LSTM, ConvLSTM),
- Making decisions for public health recommendations and alert levels.

Together, these graphs provide both comparative (Graph 2) and trend-based (Graph 3) insights, enabling HEAL to:

Purpose	How the Graphs Help
Identify Pollution Hotspots	Shows which cities require immediate policy action
Support Model Training	Assists in selecting high-variance datasets for model
optimization Improve Visualization Layer dashboards	Forms the basis for heatmaps and AQI

VIII. CONCLUSION:

This paper presented HEAL (Health and Environmental Air Quality Lens), an AI-powered system for monitoring and visualizing air quality. It offers clear pollutant predictions and visual insights for urban areas. By combining data processing, machine learning forecasting, and geospatial visualization in a flexible and expandable setup, HEAL tackles the difficulties of interpreting complex environmental data and explains it in an easy-to-understand way. The system supports various predictive models, including Random Forest and LSTM, which allow for short-term forecasting and the analysis of PM2.5 concentration patterns over time.

Quantitative evaluations show that the system is reliable, with LSTM-based forecasting achieving high accuracy and low error rates in different test areas. The visualization module also scores high in user understanding, as seen in positive feedback and the straightforwardness of the geospatial heatmaps. The system maintains low processing delays, allowing for updates close to real-time, which is suitable for public awareness dashboards and decision-making tools.

The main contributions of this work include: (1) integrating machine learning predictions with spatial heatmap visualization, (2) a flexible design that allows for adding new pollutants and data sources, and (3) demonstrating HEAL's usefulness for monitoring urban air quality and communicating public health information. Future work will aim to expand the model to include satellite-derived atmospheric indicators, seasonal variation forecasting, and a mobile interface for real-

time air quality alerts based on location to improve community-level environmental awareness.

REFERENCES

1. Iot System For Environmental Monitoring And Improving Thermal Comfort And Air Quality — IEEE Published
2. Cloud-Based System For Air Quality Monitoring — IEEE Published
3. Geo-AQI: A Real-Time Air Quality Monitoring And Forecasting System Using ARIMA Model — IEEE Published
4. Advanced Air Quality Monitoring System Using Iot And Sensor Technology — IEEE Published
5. Visualization Platform For Multi-Scale Air Pollution Monitoring And Forecast — IEEE Published
6. Environmental Pollution Monitoring Based On Sensor Network And Open-Software-Open-Hardware — IEEE Published
7. Nationwide Analysis Of Air Pollution Hotspots Across India: A Spatiotemporal PM2.5 Trend Analysis (2008–2019) — Research Paper
8. National, Satellite-Based Land-Use Regression Models For Estimating Long-Term Annual NO_2 Exposure Across India — Research Paper
9. Air Quality Forecasting In Non-Monitored Urban Areas Through Machine And Deep Learning Models Research Paper
10. Spatio-Temporal Analysis Of Air Pollution Dynamics Over Bangalore City During Second Wave Of COVID-19 — Research Paper