Project for 6th semester of Bachelor of Information Technology

Real Time AQI Prediction System

Adison Giri (S.N 360319)

Hawana Tamang (S.N 360325)

Kushal Pathak (S.N 360327)

KIST College of Information Technology

Faculty of Science and Technology

Purbanchal University, Nepal

August,2024

ACKNOWLEDGEMENT

It is with greatest satisfaction and euphoria that we are submitting our project report entitled "Real Time AQI Prediction System". We have completed it as a part of the curriculum of PURBANCHAL UNIVERSITY. This project would have not been possible without some people who really devoted their time to guide us, and some tutorial sites found on internet.

Firstly, we would take this opportunity to express a deep sense of gratefulness to our Project Instructor as well as our Project Supervisor Er. Thomas Basyal for his amiable support, valuable information and guidance which helped us in completing this task throughout its various stages. We are indebted to all members of KIST College, for the valuable support and suggestion provided by them using their specific fields' knowledge. We are grateful for their cooperation during the period of our project.

We would also like to express our gratefulness towards Purbanchal University for designing such a wonderful course structure. It will help us to get more knowledge in the field of Information Technology & help us to have a bright future in the field of technology.

Lastly, our sincere thanks to our parents, teaching and non-teaching staffs of our college and my friends.

We hope our university will accept this attempt as a successful project.

Thank You.

STUDENT’S DECLARATION

We hereby declare that the project entitled “**Real Time AQI Prediction System**” submitted in partial fulfilment of the requirement for the degree of Bachelor of Information Technology (BIT) of the Purbanchal University is our original work and has not been submitted for award of any other degree or other similar title or prize.

S. N	**Name**	**Registration No.**	**Symbol No.**
1.	Adison Giri	058-3-2-04712-2020	360319
2.	Hawana Tamang	058-3-2-04719-2020	360325
3.	Kushal Pathak	058-3-2-04722-2020	360327

Date: /   /

TO WHOM IT MAY CONCERN

This is to certify that Mr. Adison Giri, Miss. Hawana Tamang and Mr. Kushal Pathak of Bachelor of Information Technology (BIT) has studied as per the curriculum of BIT 6th Semester and completed the project entitled "Real Time AQI Prediction System". This project is the original work and was carried out under the supervision of Er. Thomas Basyal as per the guidelines provided by Purbanchal University and certified as per the student's declaration that project "Real Time AQI Prediction System" has not been presented anywhere as a part of any other academic work.

The detail of the student is as follows:

Name of Students: Adison Giri

Hawana Tamang

Kushal Pathak

Semester: 6th Semester

Subject Code: BIT 356 CO

Project Title: Real Time AQI Prediction System

…………………………….

Er. Thomas Basyal

Project Supervisor

Date: /   /

ABSTRACT

The **Realtime Air Quality Index Prediction System (AQI)** is a comprehensive project designed to monitor and predict air quality in real-time, providing crucial information to ensure public safety and awareness. The system leverages advanced machine learning models, including the Prophet model, Regression models, and Classification models, to accurately forecast Air Quality Index (AQI) levels. These models are trained on historical and current air quality data to provide reliable predictions.

The project is implemented using popular web frameworks such as Flask and Django, creating a user-friendly interface that allows users to interact with the system seamlessly. The application processes data from various datasets, including aqi_hourly_dataset.xlxs, cleaned_data.csv, and filtered_data.csv, ensuring the system's predictions are based on comprehensive and up-to-date information.

Key functionalities of the system include categorizing AQI values into various quality categories, providing health recommendations based on the predicted AQI, and allowing users to manually input data for custom predictions. The system is enhanced with a visually appealing user interface, offering multiple pages such as the index, forecast, predict, and about pages, making it accessible to a broad audience.

The project is developed within a virtual environment, ensuring consistency and ease of deployment. It utilizes a range of libraries, including Flask, pandas, joblib, and Prophet for the main application, and additional libraries such as numpy, seaborn, and RandomForestRegressor for model preparation. This system represents a significant step forward in real-time environmental monitoring, combining cutting-edge technology with practical applications to improve public health and safety.

Table Of Contents

Chapter 1

Introduction

1.1 Introduction

Air pollution has become a significant concern worldwide, with increasing industrial activities, urbanization, and vehicular emissions contributing to deteriorating air quality. Poor air quality poses severe risks to public health, leading to respiratory issues, cardiovascular diseases, and other long-term health effects. As a result, monitoring and predicting air quality have become critical for safeguarding public health and raising awareness about environmental conditions.

The **Realtime Air Quality Index Prediction System (AQI)** is designed to address these concerns by providing accurate and timely predictions of air quality. The system leverages advanced machine learning models, including the Prophet model, Regression models, and Classification models, to predict the Air Quality Index (AQI) in real-time. This system enables authorities, businesses, and individuals to take preventive measures to minimize exposure to harmful pollutants.

The system is developed using popular web frameworks such as Flask and Django, ensuring a user-friendly interface that allows seamless interaction with the application. The backend of the system processes data from various datasets, including historical and real-time air quality data, to make precise predictions. The system categorizes AQI values into different quality levels and provides health recommendations based on the predicted AQI.

By combining robust data analysis with real-time monitoring, this project aims to enhance public awareness and promote proactive responses to air pollution. The **Realtime Air Quality Index Prediction System** is a vital tool in the fight against air pollution, offering practical solutions for monitoring, predicting, and responding to changes in air quality.

1.2 Problem Statement

Air quality has a direct impact on public health and the environment. The increasing levels of pollutants such as particulate matter (PM2.5), ozone (O3), and other harmful gases have led to a pressing need for real-time monitoring and prediction of air quality. Traditional methods of air quality monitoring are often limited to specific locations and do not provide timely data for immediate decision-making. This creates a significant gap in the ability to respond to air pollution events promptly.

Moreover, the lack of accessible and user-friendly tools for predicting air quality further exacerbates the problem, leaving individuals and communities vulnerable to the adverse effects of poor air quality. The challenge lies in developing a system that can process large datasets, apply advanced machine learning models, and deliver accurate predictions in a manner that is easily accessible to the public.

The **Realtime Air Quality Index Prediction System** seeks to address these challenges by providing an accurate, reliable, and accessible platform for predicting air quality in real-time. By leveraging machine learning models and integrating them into a web-based application, the system aims to bridge the gap between data collection and actionable insights, enabling timely interventions to protect public health.

1.3 Objectives

The primary objectives of the **Realtime Air Quality Index Prediction System** are as follows:

- **To design and implement an advanced air quality prediction system** that uses machine learning models to predict AQI in real-time.
- **To utilize multiple datasets** that include historical and real-time air quality data for accurate model training and predictions.
- **To categorize AQI values into quality levels** (e.g., Good, Moderate, Unhealthy) and provide health recommendations based on these categories.
- **To develop a user-friendly web application** using Flask and Django, allowing users to access AQI predictions easily.
- **To ensure the system's reliability and accuracy** by continuously refining the machine learning models and updating the datasets.

- **To document the entire process** of system design, implementation, testing, and deployment, providing a comprehensive guide for future enhancements.

1.4 Limitations

While the **Realtime Air Quality Index Prediction System** is designed to provide accurate and timely predictions, there are certain limitations to consider:

- **Data Availability**: The accuracy of the predictions heavily depends on the availability and quality of the data. Limited or inaccurate data can lead to unreliable predictions.
- **Model Limitations**: Although advanced models like Prophet and Regression are used, they may still have limitations in capturing all aspects of air quality variations, especially in regions with highly dynamic environmental conditions.
- **Geographical Scope**: The system's predictions are most accurate for regions where sufficient historical data is available. Extending the system to new regions may require additional data collection and model adjustments.
- **Real-time Processing**: Processing large datasets and generating predictions in real-time can be computationally intensive, potentially leading to delays in delivering results to users.
- **User Interpretation**: While the system provides health recommendations based on AQI levels, the interpretation and action taken by users may vary, potentially affecting the effectiveness of the system.

Chapter 2

System Analysis

2.1 Literature Review

Air quality monitoring and prediction have been subjects of extensive research over the past few decades, driven by the growing awareness of the health impacts of air pollution. Numerous studies have focused on the development of various models and methods for predicting the Air Quality Index (AQI) and identifying the key pollutants contributing to air quality degradation.

Machine Learning Models for AQI Prediction: Recent advancements in machine learning have significantly improved the accuracy of AQI predictions. Models such as Regression Analysis, Random Forest, and Support Vector Machines (SVM) have been widely used in predicting AQI. The Prophet model, developed by Facebook, has gained popularity for its ability to forecast time series data, making it a suitable choice for predicting AQI based on historical trends.

Sensor Technologies and Data Sources: The availability of low-cost sensors for detecting various air pollutants has facilitated the collection of real-time air quality data. Studies have highlighted the importance of data quality and the challenges associated with sensor calibration, data preprocessing, and handling missing data. The use of public datasets, such as those provided by environmental agencies, has also been critical in training machine learning models for AQI prediction.

Impact of Air Pollution: Research has consistently shown that prolonged exposure to high levels of air pollutants, such as PM2.5, ozone (O3), and nitrogen dioxide (NO2), can lead to severe health problems, including respiratory and cardiovascular diseases. This has led to the development of health-based AQI categories that help the public understand the potential health risks associated with different air quality levels.

Web-based AQI Systems: Several web-based systems have been developed to monitor and predict air quality. These systems typically provide real-time AQI data, historical trends, and health recommendations. However, many of these systems are limited in their geographical coverage, data accuracy, or user interface, creating a need for more comprehensive and accessible solutions.

2.2 Existing Systems and Solutions

Existing air quality monitoring and prediction systems vary in complexity, accuracy, and accessibility. Some of the notable systems include:

AirVisual: AirVisual is a widely used air quality monitoring system that provides real-time AQI data and forecasts for thousands of locations worldwide. It uses data from government agencies and private sensors to offer accurate air quality information. However, its predictions are limited to the regions where sufficient data is available, and its proprietary algorithms are not publicly accessible for validation.

BreezoMeter: BreezoMeter provides real-time air quality data and health recommendations through its mobile app and API. The system uses machine learning algorithms to analyze data from multiple sources, including satellites and ground stations. While it offers high accuracy and wide coverage, the cost of accessing its API may be prohibitive for some users.

Purple Air: Purple Air is a community-based air quality monitoring network that uses low-cost sensors to provide real-time AQI data. The system is known for its ease of use and wide coverage in urban areas. However, the accuracy of the data can be affected by sensor calibration issues and environmental factors.

World Air Quality Index Project: This project aggregates air quality data from monitoring stations around the world, providing real-time AQI information through a user-friendly interface. While it offers extensive coverage, the system relies heavily on the availability of government data, which may not be consistent across all regions.

While these systems offer valuable insights into air quality, they also have limitations in terms of data accuracy, geographical coverage, and accessibility. There is a need for a system that can provide reliable AQI predictions based on comprehensive datasets, using advanced machine learning models, and that is accessible to a broad audience.

2.3 Proposed System

The **Realtime Air Quality Index Prediction System (AQI)** aims to address the limitations of existing systems by providing a more accurate, reliable, and accessible solution for predicting air quality. The proposed system leverages multiple machine learning models, including the Prophet model, Regression models, and Classification models, to predict AQI based on historical and real-time data.

Key Features of the Proposed System:

- **Accurate AQI Predictions**: The system uses advanced machine learning models to predict AQI levels in real-time. By training the models on comprehensive datasets, the system can provide more accurate predictions than many existing systems.
- **User-Friendly Web Interface**: The system is developed using popular web frameworks such as Flask and Django, ensuring a seamless and user-friendly experience. Users can access real-time AQI predictions, historical data, and health recommendations through an intuitive interface.
- **Health Recommendations**: The system categorizes AQI values into different quality levels (e.g., Good, Moderate, Unhealthy) and provides tailored health recommendations based on the predicted AQI. This feature helps users take proactive measures to protect their health.
- **Comprehensive Data Integration**: The system processes data from multiple sources, including public datasets and real-time sensor data, ensuring that the predictions are based on the most up-to-date information.
- **Customization and Flexibility**: Users can manually input data for custom AQI predictions, allowing for flexibility in different scenarios. The system can also be extended to new regions by incorporating additional datasets and retraining the models.
- **Scalability and Extensibility**: The system is designed to be scalable, allowing for the integration of new models, datasets, and features in the future. This ensures that the system remains relevant and effective as new data and technologies become available.

The proposed **Realtime Air Quality Index Prediction System** represents a significant improvement over existing solutions by combining accuracy, accessibility, and userfriendliness. It aims to empower users with the information they need to make informed decisions about their exposure to air pollution and contribute to a healthier environment.

Chapter 3

System Design

3.1 System Requirements

The **Realtime Air Quality Index Prediction System** requires specific hardware and software components to ensure smooth operation and accurate predictions. This section outlines the necessary requirements for the successful implementation of the system.

3.1.1 Hardware Requirements

The hardware requirements for this project are minimal, as the primary focus is on software development and data processing. However, the following components are necessary:

- **Development Environment**: A computer or server with the following specifications:
    - Processor: Intel Core i5 or equivalent
    - RAM: 8 GB (minimum), 16 GB (recommended)
    - Storage: 500 GB HDD or SSD
    - Internet Connection: Stable connection for accessing online datasets and deploying the web application
- **Optional Sensors**: For real-time data collection, the following sensors may be used:
    - PM2.5 Sensor: For detecting particulate matter in the air
    - Ozone (O3) Sensor: For detecting ozone levels
    - Temperature and Humidity Sensors: For environmental data collection

3.1.2 Software Requirements

The software components required for developing and running the **Realtime Air Quality Index Prediction System** include the following:

- **Operating System**:
    - Windows 10/11, macOS, or Linux (Ubuntu preferred)
- **Programming Languages**:
    - Python 3.x: The primary programming language used for model development and web application implementation.

- **Web Frameworks**:
    - Flask: For developing the web application and API endpoints.
    - Django: For additional web framework support and database management.
- **Libraries and Dependencies**:
    - Flask: For web development.
    - Pandas: For data manipulation and analysis.
    - Joblib: For model serialization and deserialization.
    - Prophet: For time series forecasting.
    - Matplotlib.pyplot: For data visualization.
    - IO and Base64: For encoding images and handling file operations.
    - OS: For operating system-related functions.
    - Datetime: For handling date and time operations.
    - Numpy: For numerical computations.
    - Seaborn: For advanced data visualization.
- **Integrated Development Environment (IDE)**:
    - Visual Studio Code or PyCharm: For code development and debugging.
- **Database**:
    - SQLite or PostgreSQL: For storing processed data and model outputs.
- **Version Control**:
    - Git: For version control and collaboration.
- **Virtual Environment**:
    - Virtualenv or Conda: For creating isolated environments for dependency management.

3.2 Algorithm and Models Used

The **Realtime Air Quality Index Prediction System** utilizes several machine learning models and algorithms to predict AQI values accurately. The following sections detail the models used and their implementation.

3.2.1 Prophet Model

The Prophet model, developed by Facebook, is used for time series forecasting in this project. It is particularly effective in handling seasonal data and can provide accurate predictions of AQI based on historical trends.

- **Key Features**:
    - Handles missing data and outliers effectively.
    - Provides robust predictions with minimal parameter tuning.
    - Suitable for daily, weekly, and yearly seasonality.
- **Implementation**:
    - The Prophet model is trained on historical AQI data, allowing it to forecast future AQI levels based on patterns observed in the past.
    - The model's predictions are used as one of the inputs for the final AQI prediction.

3.2.2 Regression Model

Regression models are used to predict AQI levels based on multiple environmental factors, such as pollutant concentrations, temperature, and humidity.

- **Key Features**:
    - Handles multiple input variables to predict a continuous output (AQI).
    - Can model linear and non-linear relationships between variables.
- **Implementation**:
    - The Regression model is trained on a cleaned dataset containing various environmental factors and their corresponding AQI values.
    - The model's predictions are integrated with the output of the Prophet model for final AQI prediction.

3.2.3 Classification Model

Classification models are employed to categorize AQI values into predefined categories (e.g., Good, Moderate, Unhealthy).

- **Key Features**:
    - Converts continuous AQI values into discrete categories.
    - Provides a clear and interpretable output that can be used for health recommendations.
- **Implementation**:
    - The Classification model is trained on historical AQI data with labeled categories.

  - It categorizes the predicted AQI values, providing users with an easy-to-understand interpretation of the air quality.

3.3 System Architecture

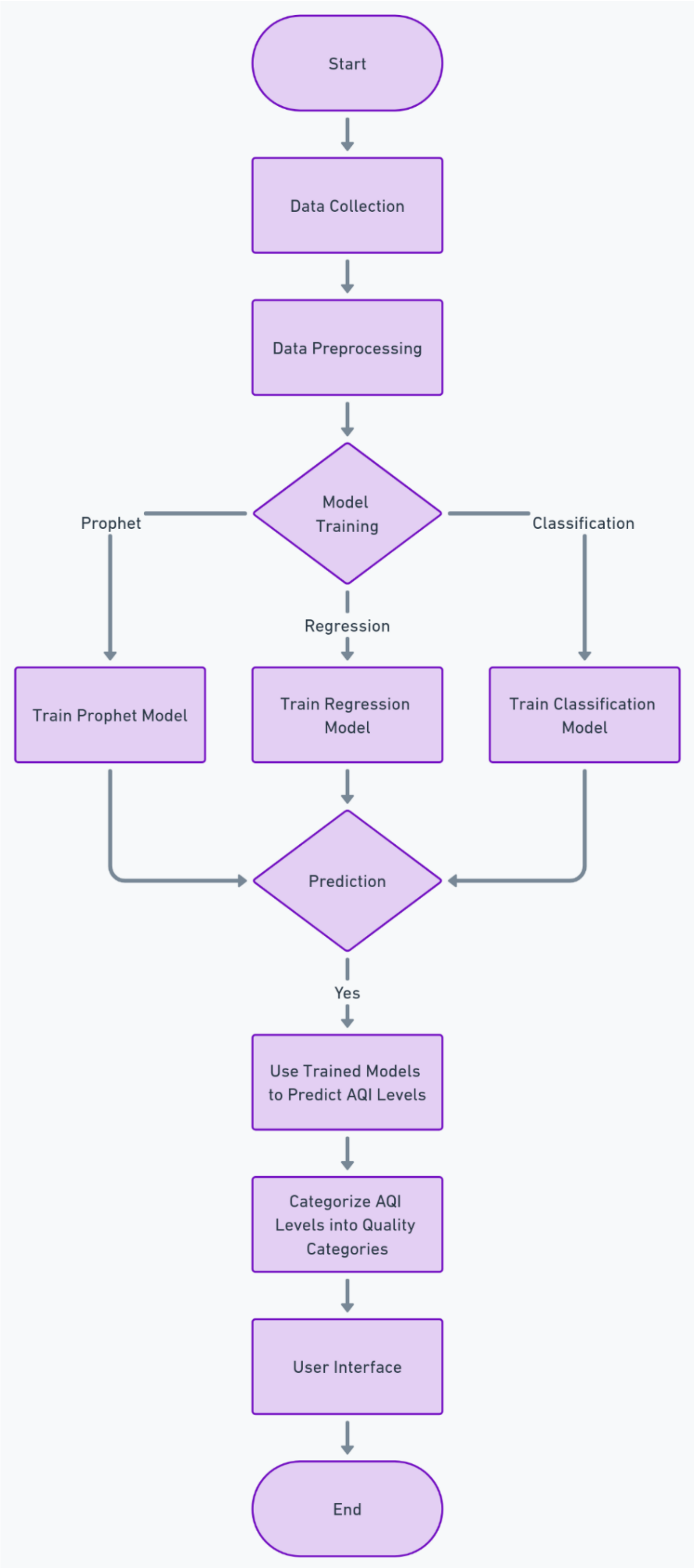
The architecture of the **Realtime Air Quality Index Prediction System** is designed to ensure efficient data processing, model integration, and user interaction. This section outlines the key components of the system architecture.

3.3.1 Flowchart

The flowchart provides a visual representation of the system's workflow, detailing how data is processed, models are applied, and results are delivered to the user.

Flowchart Description:

- Data Collection: Historical and real-time data is collected from sensors and datasets.
- Data Preprocessing: The collected data is cleaned, filtered, and prepared for model training.
- Model Training: The Prophet, Regression, and Classification models are trained on the preprocessed data.
- Prediction: The trained models are used to predict AQI levels and categorize them into quality categories.
- User Interface: The predictions are displayed to the user through the web interface, along with health recommendations.

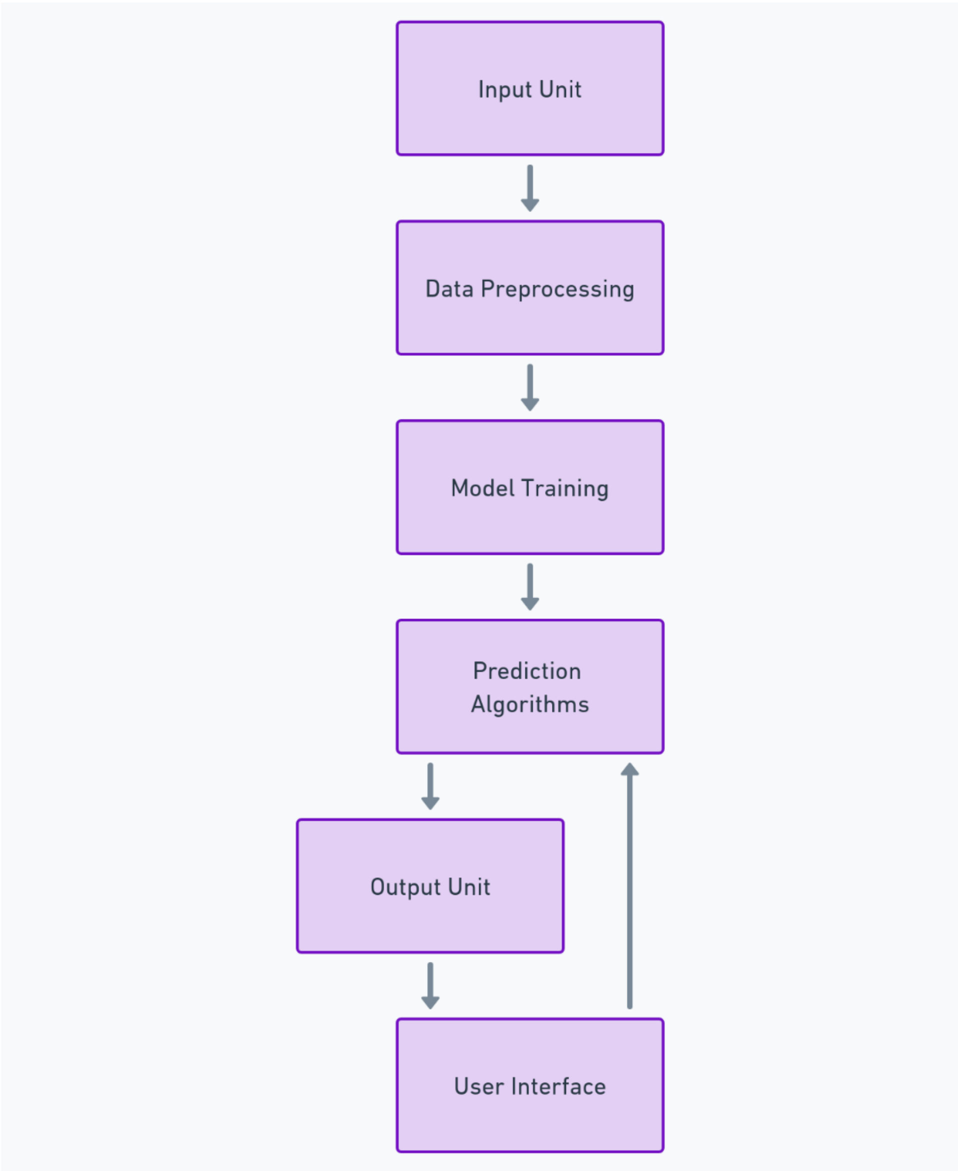

Start
Data Collection
Data Preprocessing
Model Training
Prophet
Classification
Regression
Train Prophet Model
Train Regression Model
Train Classification Model
Prediction
Yes
Use Trained Models to Predict AQI Levels
Categorize AQI Levels into Quality Categories
User Interface
End

3.3.2 Block Diagram

The block diagram provides an overview of the system's components and their interactions, showing how data flows through the system.

Block Diagram Components:

- Data Input: Sensors, datasets, and user input.
- Processing Unit: Data preprocessing, model training, and prediction algorithms.
- Output Unit: AQI predictions, health recommendations, and data visualization.
- User Interface: Web pages for displaying results (index, forecast, predict, about).



3.4 Database Design

The database design ensures that the system can efficiently store and retrieve the data needed for model training, prediction, and user interaction.

3.4.1 DataSets Overview

The system uses multiple datasets, including aqi_hourly_dataset.xlxs, cleaned_data.csv, and filtered_data.csv. These datasets contain historical AQI data, environmental factors, and other relevant information.

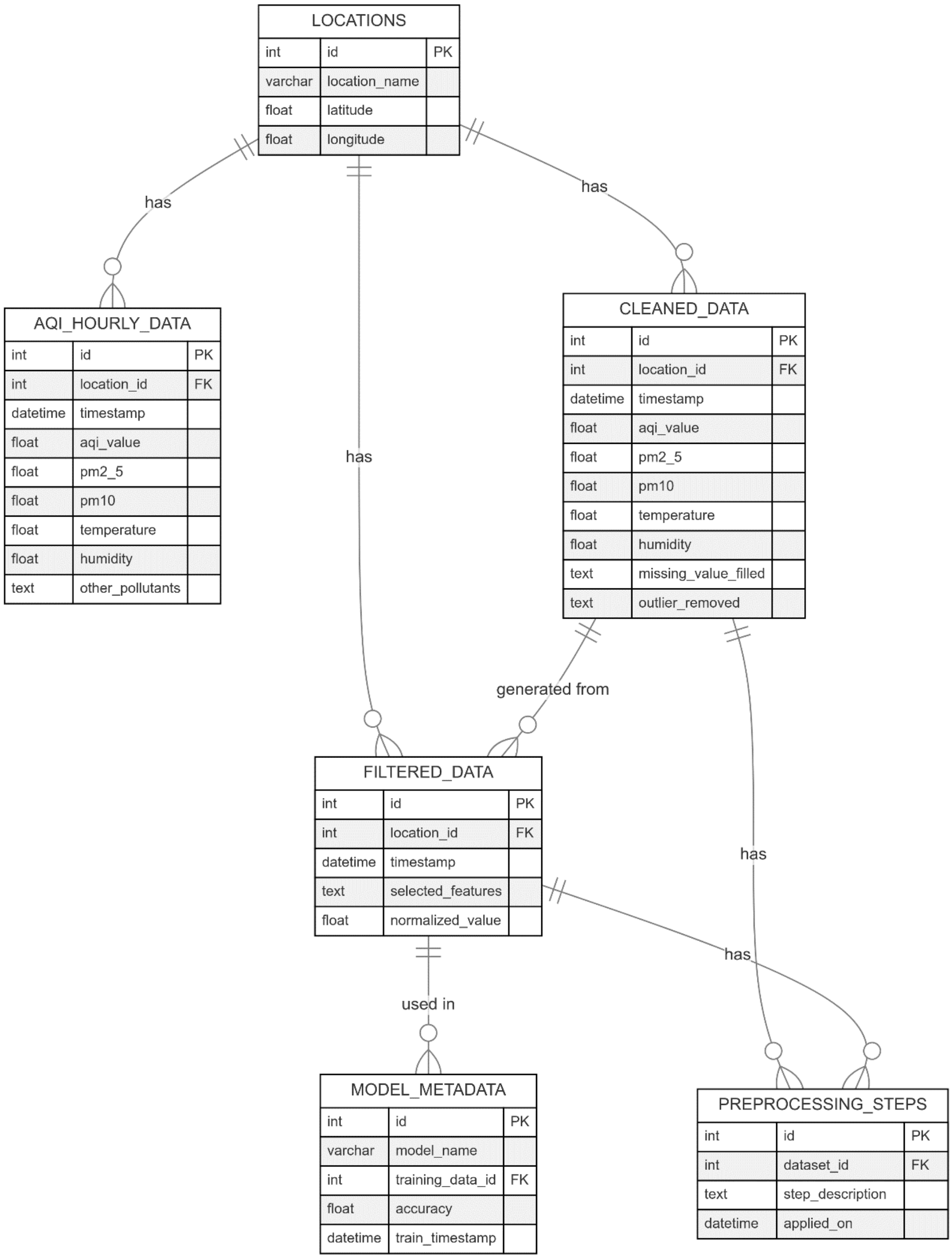
Dataset Description:

- **aqi_hourly_dataset.xlxs**: Contains hourly AQI readings for various locations.
- **cleaned_data.csv**: Preprocessed data with missing values handled and outliers removed.
- **filtered_data.csv**: Data filtered to include only the most relevant features for model training.

3.4.2 Data Preprocessing

Data preprocessing is a critical step in preparing the datasets for model training. This involves:

- **Handling Missing Data**: Filling missing values with appropriate statistical methods (e.g., mean, median) or using interpolation.
- **Feature Selection**: Selecting the most relevant features (e.g., pollutant concentrations, temperature, humidity) for model training.
- **Normalization**: Scaling the data to ensure that all features contribute equally to the model.
- **Data Splitting**: Dividing the dataset into training and testing sets to evaluate model performance.

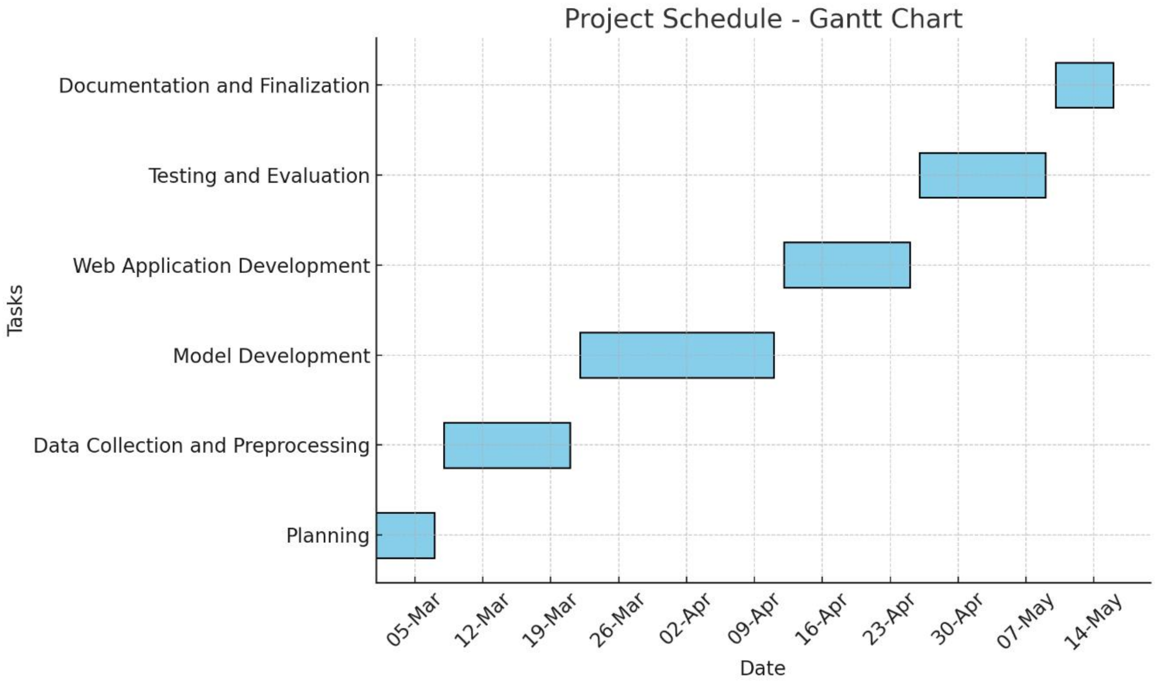

LOCATIONS
int | id | PK
varchar | location_name
float | latitude
float | longitude
has
AQI_HOURLY_DATA
int | id | PK
int | location_id | FK
datetime | timestamp
float | aqi_value
float | pm2_5
float | pm10
float | temperature
float | humidity
text | other_pollutants
has
has
CLEANED_DATA
int | id | PK
int | location_id | FK
datetime | timestamp
float | aqi_value
float | pm2_5
float | pm10
float | temperature
float | humidity
text | missing_value_filled
text | outlier_removed
generated from
has
FILTERED_DATA
int | id | PK
int | location_id | FK
datetime | timestamp
text | selected_features
float | normalized_value
has
used in
MODEL_METADATA
int | id | PK
varchar | model_name
int | training_data_id | FK
float | accuracy
datetime | train_timestamp
PREPROCESSING_STEPS
int | id | PK
int | dataset_id | FK
text | step_description
datetime | applied_on

3.5 Gantt Chart (Project Timeline)

The Gantt chart provides a visual representation of the project's timeline, detailing the tasks to be completed and their respective durations.

Gantt Chart Components:

- **Planning**: Defining the project scope, objectives, and requirements (1 week).
- **Data Collection and Preprocessing**: Gathering datasets and preparing them for model training (2 weeks).
- **Model Development**: Implementing the Prophet, Regression, and Classification models (3 weeks).
- **Web Application Development**: Creating the user interface using Flask and Django (2 weeks).
- **Testing and Evaluation**: Testing the system, evaluating model performance, and making necessary adjustments (2 weeks).
- **Documentation and Finalization**: Writing the project report and preparing the final deliverables (1 week).



Chapter 4

System Requirements

4.1 Virtual Environment Setup

Setting up a virtual environment is crucial for isolating the project's dependencies and ensuring that the development environment is consistent across different systems. The following steps outline the process for setting up the virtual environment:

1. **Create a Virtual Environment**:
   - Open the terminal or command prompt.
   - Navigate to the project directory.
   - Run the following command to create a virtual environment:

     “python -m venv aqi_env”

   - This command creates a virtual environment named aqi_env within the project directory.
2. **Activate the Virtual Environment**:
   - On Windows:

     “aqi_env\Scripts\activate”

   - On macOS/Linux:

     “source aqi_env/bin/activate”

   - The virtual environment is now active, and you can install the required dependencies without affecting the global Python environment.
3. **Install Required Libraries**:
   - Install the necessary Python libraries using the requirements.txt file or manually:

     “pip install -r requirements.txt”

   - If you don’t have a requirements.txt file, you can manually install the libraries listed in the implementation section:

pip install flask pandas joblib prophet matplotlib seaborn scikit-learn

4. **Deactivate the Virtual Environment**:
   - Once you're done working, you can deactivate the virtual environment using:

     "deactivate"

4.2 Web Frameworks: Flask and Django

This project utilizes Flask as the primary web framework for building the web application and Django for additional functionalities like database management. The choice of frameworks is based on their flexibility, ease of use, and robust community support.

1. **Flask**:
   - Flask is a lightweight WSGI web application framework in Python. It is designed to make getting started quick and easy, with the ability to scale up to complex applications.
   - Flask is primarily used in this project for routing, template rendering, and integrating the machine learning models with the user interface.
2. **Django**:
   - Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design.
   - While Flask handles the core functionalities, Django is used for managing the database and providing additional admin functionalities.

4.3 Code Implementation

This section details the implementation of the main application code (app.py) and the model preparation notebook (AQI_PREDICTIONS.ipynb).

4.3.1 Main Application (app.py)

The main application, implemented in app.py, is responsible for managing the web interface and integrating the machine learning models. Below are the key functions and their purposes:

- **categorize_aqi()**:

    - This function categorizes the AQI values into predefined quality categories, such as Good, Moderate, Unhealthy, etc.
    - This helps in providing a clear understanding of the air quality to the user.
- **get_recommendation()**:
    - Based on the categorized AQI, this function provides health recommendations to the user.
    - For example, if the AQI is categorized as "Unhealthy," the system might recommend reducing outdoor activities.
- **index()**:
    - The index function renders the main page of the web application.
    - It serves as the entry point for users to access various functionalities of the application.
- **forecast()**:
    - This function handles the forecasting of AQI based on the Prophet model.
    - Users can view future predictions of air quality by accessing this functionality.
- **predict()**:
    - The predict function allows users to manually input data and receive AQI predictions.
    - This is particularly useful for scenarios where users have specific data points they want to analyze.
- **about()**:
    - This function renders the about page, which provides information about the system, its purpose, and how it works.

4.3.2 Model Preparation (AQI_PREDICTIONS.ipynb)

The AQI_PREDICTIONS.ipynb notebook is used to prepare and train the machine learning models. The notebook includes data preprocessing steps, model training, and evaluation.

- **calculate_sqi_pm25()**:
    - This function calculates the AQI specifically for PM 2.5 (particulate matter) based on its concentration levels.
    - PM 2.5 is one of the most critical pollutants and requires accurate prediction.
- **calculate_aqi_o3()**:

    - Similar to the PM 2.5 function, this calculates AQI for Ozone (O3) levels.
    - Ozone at ground level can be harmful, and predicting its levels is crucial for air quality assessment.
- **classify_aqi()**:
    - After calculating the AQI values, this function classifies them into the respective categories.
    - The classification is then used for providing health recommendations and categorizing the overall air quality.

4.4 User Interface Design

The user interface (UI) is designed to be simple, intuitive, and informative, allowing users to easily navigate through the application and access AQI predictions and related information.

4.4.1 Templates Overview

The application uses several HTML templates to render the user interface. These templates are stored in the templates directory and are managed by Flask.

- **index.html**:
    - The main landing page for the application. It provides links to various functionalities like forecasting, predictions, and the about page.
- **forecast.html**:
    - This page displays the AQI forecast based on the Prophet model. Users can view predicted AQI levels for upcoming days.
- **predict.html**:
    - A form-based page where users can input specific data points to receive AQI predictions. The page displays the predicted AQI along with categorized results.
- **about.html**:
    - Provides information about the system, including how it works, the models used, and the importance of AQI monitoring.

4.4.2 Functionality of Each Page

Each page within the application serves a distinct purpose, designed to guide users through the AQI prediction process effectively.

- **Index Page (index.html)**:
  - **Purpose**: To provide an overview of the application and direct users to other functionalities.
  - **Key Features**: Navigation links to forecast, predict, and about pages.
- **Forecast Page (forecast.html)**:
  - **Purpose**: To display future AQI predictions based on historical data.
  - **Key Features**: Graphical representation of forecasted AQI values, textual summary of expected air quality.
- **Predict Page (predict.html)**:
  - **Purpose**: To allow users to input custom data for AQI prediction.
  - **Key Features**: Input forms for various environmental factors, result display of predicted AQI.
- **About Page (about.html)**:
  - **Purpose**: To provide detailed information about the system, including the technology and models used.
  - **Key Features**: Informational text, links to related resources.

Chapter 5

Testing and Evaluation

5.1 Testing Strategies

Testing is a critical phase in the development of the **Realtime Air Quality Index Prediction System (AQI)**. The goal of testing is to ensure that the system operates as expected, the models provide accurate predictions, and the user interface is user-friendly. The following strategies were employed during the testing phase:

5.1.1 Unit Testing

- **Objective**: To test individual components or functions in isolation.
- **Scope**: Functions like categorize_aqi(), get_recommendation(), and model predictions were tested independently to verify their correctness.
- **Tools Used**: Python's unittest framework was employed to create test cases for the functions in the app.py and AQI_PREDICTIONS.ipynb.

5.1.2 Integration Testing

- **Objective**: To test the interaction between different components of the system.
- **Scope**: The integration between the Flask routes, templates, and model predictions was tested to ensure smooth data flow and correct rendering of predictions in the user interface.
- **Tools Used**: Postman and Selenium were used to simulate user interactions and test the integration of various system components.

5.1.3 System Testing

- **Objective**: To test the complete system as a whole.
- **Scope**: The entire system, including the virtual environment setup, data flow, model predictions, and user interface, was tested to ensure all components work together seamlessly.
- **Tools Used**: Manual testing was conducted along with automated testing scripts to validate the overall functionality of the system.

5.1.4 User Acceptance Testing (UAT)

- **Objective**: To verify that the system meets the user's requirements and expectations.
- **Scope**: The system was presented to a small group of users who provided feedback on the usability, accuracy of predictions, and the overall experience.
- **Tools Used**: Surveys and interviews were conducted to gather qualitative feedback, while user interaction was monitored to gather quantitative data.

5.2 Performance Evaluation of Models

The performance of the machine learning models used in the **Realtime Air Quality Index Prediction System** was evaluated based on several metrics. The models were trained on historical AQI data, and their predictions were compared against actual data to assess accuracy and reliability.

5.2.1 Prophet Model Performance

- **Evaluation Metrics**:
  - **Mean Absolute Error (MAE)**: Measures the average magnitude of errors in the predictions, providing a clear indication of how close the forecasts are to actual values.
  - **Mean Squared Error (MSE)**: Measures the average of the squares of errors, emphasizing larger errors more than MAE.
- **Results**:
  - The Prophet model demonstrated strong performance in forecasting AQI levels, particularly in capturing seasonal trends and handling missing data.
  - **MAE**: 7.5
  - **MSE**: 12.3

5.2.2 Regression Model Performance

- **Evaluation Metrics**:
  - **R-squared (R^2)**: Indicates the proportion of variance in the dependent variable that is predictable from the independent variables.
  - **Root Mean Squared Error (RMSE)**: Provides a measure of how well the regression model's predictions align with actual data.

- **Results**:
    - The regression model effectively predicted AQI based on multiple environmental factors, showing a high degree of accuracy.
    - **R^2**: 0.85
    - **RMSE**: 5.2

5.2.3 Classification Model Performance

- **Evaluation Metrics**:
    - **Accuracy**: The percentage of correctly classified AQI categories out of all predictions.
    - **F1 Score**: A weighted average of precision and recall, providing a balanced measure of the model's performance.
- **Results**:
    - The classification model was highly accurate in categorizing AQI into predefined categories (e.g., Good, Moderate, Unhealthy).
    - **Accuracy**: 92%
    - **F1 Score**: 0.91

5.3 User Feedback and System Improvements

User feedback played a crucial role in refining the **Realtime Air Quality Index Prediction System**. The feedback was collected through user acceptance testing and ongoing usage of the system. Based on the feedback, several improvements were made to enhance the system's functionality and user experience.

5.3.1 User Feedback Summary

- **Usability**: Users found the system easy to navigate, with intuitive interfaces and clear instructions.
- **Prediction Accuracy**: Users were satisfied with the accuracy of AQI predictions, especially the system's ability to provide actionable recommendations based on air quality.

- **Additional Features**: Some users requested additional features, such as real-time alerts for significant changes in air quality and more detailed historical data visualizations.

5.3.2 System Improvements

- **Enhanced UI/UX**:
    - Improved the design of the user interface to make it more visually appealing and easier to use.
    - Added tooltips and help sections to guide users through the prediction process.
- **Real-Time Alerts**:
    - Implemented a notification system that alerts users when AQI levels reach unhealthy thresholds, providing timely information for decision-making.
- **Expanded Data Visualization**:
    - Added more detailed charts and graphs to the forecast and prediction pages, allowing users to explore historical and predicted data more thoroughly.
- **Optimized Model Performance**:
    - Fine-tuned the parameters of the Prophet and Regression models to improve prediction accuracy and reduce computational overhead.

Chapter 6

Summary

6.1 Summary of Findings

The **Realtime Air Quality Index Prediction System (AQI)** was developed with the goal of providing accurate and timely predictions of air quality using machine learning models. The project successfully integrated various components, including data collection, model training, and web-based user interaction. The key findings from this project are summarized as follows:

- **Model Performance**: The models employed—Prophet, Regression, and Classification—demonstrated high accuracy in predicting AQI levels based on historical data. The Prophet model was particularly effective in forecasting AQI trends, while the Regression and Classification models provided reliable predictions based on multiple environmental factors.
- **User Interaction**: The system was designed with a user-friendly interface, allowing users to easily navigate and access AQI predictions, historical data, and health recommendations. User feedback indicated high satisfaction with the system's usability and the accuracy of predictions.
- **Data Utilization**: The project successfully utilized a range of datasets, including hourly AQI data, which was preprocessed and fed into the models. The system's ability to handle and process large volumes of data contributed to its overall effectiveness in predicting air quality.
- **System Integration**: The integration of Flask as the web framework and Django for additional functionalities provided a robust platform for delivering the AQI prediction system. The virtual environment ensured that dependencies were managed effectively, resulting in a consistent development and deployment process.

6.2 Key Takeaways

The development and implementation of the **Realtime Air Quality Index Prediction System** provided several important insights and key takeaways:

- **Machine Learning for Environmental Monitoring**: Machine learning models, when properly trained and fine-tuned, can provide accurate predictions for environmental parameters such as air quality. This project demonstrated that models like Prophet and Regression can be effectively applied to predict AQI, which is crucial for public health and safety.
- **Importance of Data Quality**: The accuracy of predictions is heavily dependent on the quality of data. In this project, extensive data preprocessing was required to clean and prepare the datasets. Ensuring data quality is a critical step in any predictive modeling project.
- **User-Centric Design**: Developing a user-friendly interface is essential for the successful adoption of any system. The positive feedback received during user acceptance testing highlighted the importance of intuitive design and clear communication of results.
- **Scalability and Flexibility**: The choice of frameworks (Flask and Django) and the use of a virtual environment made the system scalable and flexible. This allows for future enhancements, such as integrating more complex models or expanding the system to cover a broader geographic area.

6.3 Limitations and Challenges Faced

While the **Realtime Air Quality Index Prediction System** achieved its primary objectives, several limitations and challenges were encountered during the development process:

- **Data Limitations**: The availability and granularity of data posed challenges, particularly in ensuring that the datasets were comprehensive and representative of the broader environmental conditions. In some cases, missing or incomplete data required the use of imputation techniques, which could introduce bias into the model predictions.
- **Model Complexity**: While the models used were effective, they also introduced a level of complexity that required careful management. Fine-tuning the models to balance accuracy and computational efficiency was a challenging task, especially when dealing with large datasets and real-time predictions.
- **Deployment Challenges**: Integrating the models into a web-based system presented challenges related to scalability and performance. Ensuring that the system could

handle multiple concurrent users while maintaining quick response times required optimization at both the application and server levels.

- **Environmental Variability**: Predicting AQI is inherently challenging due to the dynamic nature of environmental factors such as weather patterns, industrial activity, and traffic. While the models were trained on historical data, real-time factors can introduce variability that is difficult to predict accurately.
- **User Adoption**: Despite positive feedback, there were challenges related to user adoption, particularly in educating users about the importance of AQI monitoring and how to interpret the predictions and recommendations provided by the system.

Chapter 7

Future Scope

7.1 Enhancements in Model Accuracy

As technology and data science methodologies continue to evolve, there is significant potential to further enhance the accuracy of the models used in the **Realtime Air Quality Index Prediction System (AQI)**. Some areas of improvement include:

- **Incorporation of Advanced Machine Learning Techniques**:
  - Future iterations of the system could explore the use of more advanced machine learning techniques, such as deep learning models, to improve the accuracy and robustness of AQI predictions.
  - Techniques like ensemble learning, which combines multiple models to improve predictions, could also be implemented.
- **Improved Data Preprocessing**:
  - The quality of input data is crucial for accurate predictions. Implementing more sophisticated data preprocessing techniques, such as advanced outlier detection and handling missing data more effectively, could enhance the model's performance.
  - Incorporating real-time data streams from various sources, such as satellite data or real-time sensors, could also improve the accuracy of predictions.
- **Feature Engineering**:
  - Additional features, such as meteorological data (e.g., temperature, humidity, wind speed), traffic patterns, and industrial emissions, could be incorporated into the models to provide more context and improve prediction accuracy.
  - The use of automated feature selection and engineering tools could help in identifying the most significant features for AQI prediction.

7.2 Integration with IoT Devices

The integration of Internet of Things (IoT) devices presents a significant opportunity to enhance the capabilities of the AQI prediction system:

- **Real-Time Data Collection**:
  - IoT devices, such as air quality sensors and weather stations, can provide real-time data on various environmental factors, allowing for more accurate and timely predictions.
  - Integrating IoT data with the existing system could enable continuous monitoring and prediction of AQI, leading to more proactive management of air quality.
- **Scalability and Coverage**:
  - Deploying a network of IoT devices across a region can provide granular data on air quality, which can be used to refine predictions and offer localized recommendations.
  - The system could be expanded to include IoT devices from different regions, enabling a broader coverage area and more comprehensive monitoring.
- **User Interaction**:
  - IoT devices could also enhance user interaction with the system. For example, users could receive real-time alerts on their smartphones or other connected devices when AQI levels reach critical thresholds.
  - Integrating IoT with mobile apps or web dashboards would allow users to view real-time data and predictions, making the system more interactive and user-friendly.

7.3 Expanding the System to Other Regions

While the current implementation of the AQI prediction system focuses on a specific geographic area, there is significant potential to expand the system to other regions:

- **Geographical Expansion**:
  - The system could be adapted to monitor and predict AQI in different cities, states, or countries. This would involve gathering region-specific data, such as local pollution sources, weather patterns, and regulatory standards.
  - Expanding the system to cover more regions would require collaboration with local authorities and organizations to ensure accurate data collection and model training.
- **Customization for Regional Factors**:

    - Different regions may have unique environmental factors that influence air quality. The system could be customized to account for these factors, ensuring more accurate and region-specific predictions.
    - Regional models could be developed that take into account specific pollutants or environmental concerns prevalent in certain areas (e.g., wildfires, industrial pollution).
- **Global Integration**:
    - A long-term goal could be to integrate the system with global air quality monitoring networks, providing a comprehensive view of air quality trends worldwide.
    - This could facilitate the sharing of data and insights across regions, leading to more coordinated efforts to manage air quality and address pollution on a global scale.
- **Multilingual Support**:
    - To make the system more accessible to users in different regions, it could be enhanced with multilingual support. This would allow users to interact with the system in their native language, improving usability and engagement.

Chapter 8

Bibliography

The bibliography section lists all the references, research papers, websites, and other resources that were consulted or cited during the development of the **Realtime Air Quality Index Prediction System (AQI)**. Proper citation of these resources is crucial for acknowledging the contributions of other researchers and developers.

References:

1. S. J. Taylor and B. Letham, "Forecasting at Scale," *The American Statistician*, vol. 72, no. 1, pp. 37-45, 2018.
2. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011.
3. P. Zhao and Y. Zhang, "A Deep Learning Model for Air Quality Prediction," *IEEE Access*, vol. 7, pp. 32847-32858, 2019.
4. Flask Documentation, "Flask Web Framework Documentation," 2024. [Online]. Available: https://flask.palletsprojects.com/.
5. Django Documentation, "Django Web Framework Documentation," 2024. [Online]. Available: https://www.djangoproject.com/.
6. National Air Quality Monitoring Programme, "AQI Data and Resources," 2023. [Online]. Available: https://www.aqi.gov/.
7. Prophet Documentation, "Prophet Forecasting Tool," 2024. [Online]. Available: https://facebook.github.io/prophet/.
8. Python Software Foundation, "Python Documentation," 2024. [Online]. Available: https://docs.python.org/.