**Low Level Design**

**Air Quality Index Prediction**

|  |  |
| --- | --- |
| **Written by** | **Gunaseelan V** |
| **Documented Version** | **1.0** |
| **Last Revised Date** | **18 December 2024** |

****Contents****

1. **Abstract**
2. **Introduction  
   2.1 Why this Low-Level Design Document?  
   2.2 Scope  
   2.3 Constraints  
   2.4 Risk**
3. **Technology Stack**
4. **Proposed Solution**
5. **Model Training/Validation Workflow**
6. **User I/O Workflow**
7. **Exceptional Scenarios**
8. **Key Performance Indicators (KPI)**

### **1. Abstract**

**The Air Quality Index (AQI) Prediction System is designed to predict AQI values based on key pollutants such as O₃, NO₂, PM₁₀, and CO. By utilizing machine learning techniques like XGBoost and Gradient Boosting, the system provides accurate predictions that can assist in environmental monitoring, public health decisions, and pollution control measures. It ensures scalability and real-time usability for end users, with a cloud-based deployment ensuring accessibility for wide-scale monitoring.**

### **2. Introduction**

#### **2.1 Why this Low-Level Design Document?**

**The purpose of this Low-Level Design Document (LLD) is to provide detailed planning for the implementation of the AQI Prediction System. It acts as a guide for developers, testers, and stakeholders, breaking down the system into components such as data collection, model training, user workflows, and database design. This document ensures a clear understanding of how the system will be built and helps streamline the development process.**

#### **2.2 Scope**

**This LLD focuses on the following components:**

* ****Data Collection**: Historical AQI data and pollutant levels will be collected, cleaned, and preprocessed.**
* ****Machine Learning Models**: XGBoost, Gradient Boosting, and other relevant models will be used for AQI prediction.**
* ****Model Deployment**: The models will be deployed on a cloud platform (e.g., AWS) for accessibility and scalability.**
* ****User Interaction**: The system will accept real-time data for prediction and provide results through a user-friendly interface (e.g., Streamlit).**
* ****Storage**: Predicted AQI values and user data will be stored for future analysis.**

#### **2.3 Constraints**

* ****Data Availability**: Reliable and complete pollutant data is required for accurate predictions.**
* ****Resource Constraints**: Limited computational power may affect model training time.**
* ****Latency**: There might be delays in generating predictions when processing large volumes of input data.**
* ****Data Format**: Variations in data formats, missing values, or inconsistent pollutant data may affect the accuracy of predictions.**

#### **2.4 Risks**

* ****Model Overfitting**: Risk of overfitting with complex models and imbalanced data.**
* ****External Changes**: Sudden changes in environmental conditions (e.g., weather patterns) may affect model accuracy.**
* ****Cloud Downtime**: Unavailability of cloud services can impact the availability of the system.**
* ****Security Risks**: User and environmental data stored in the database should be protected against unauthorized access.**

### **3. Technical Specifications**

#### **3.1 Dataset**

**The system uses structured datasets containing historical AQI values, pollutant concentrations, and other environmental attributes to predict AQI. Key features include:**

****Key Attributes****

****Pollutant Levels**:**

* + ****O₃ (Ozone) AQI****
  + ****NO₂ (Nitrogen Dioxide) AQI****
  + ****PM₂.₅ (Particulate Matter) AQI****
  + ****CO (Carbon Monoxide) AQI****

****Date and Time**: Timestamp of when AQI data was recorded.**

****Weather Data** (optional): External factors like temperature, humidity, wind speed that can influence AQI.**

****Preprocessing Steps****

* ****Dimensionality Reduction**: Initial features are reduced based on correlation to focus on key variables influencing AQI prediction.**
* ****Missing Value Handling**: Missing pollutant levels are handled by imputation (e.g., mean imputation for continuous values).**
* ****Encoding**: Categorical variables, if any (e.g., city type), are numerically encoded.**
* ****Data Transformation**: Logarithmic transformations are applied to skewed variables (e.g., pollutant concentrations).**

#### **3.2 Input Schema**

* ****Input Columns**:**
  + **O₃ AQI**
  + **NO₂ AQI**
  + **PM₂.₅ AQI**
  + **CO AQI**

****Output**:**

* **Predicted AQI (target variable)**

#### **3.3 Prediction Workflow**

****3.3.1 Pretrained Model Setup**  
The system utilizes a pretrained model developed on structured datasets containing pollutant levels. The model has been trained using advanced machine learning algorithms such as XGBoost or Gradient Boosting.**

****3.3.2 Model Training****

* ****Data Splitting**: The dataset is divided into training (80%) and testing (20%) sets.**
* ****Transformation**: Features are transformed (logarithmic scaling for highly skewed variables).**
* ****Model Training**: The Gradient Boosting Regressor model is used due to its ability to handle both numerical and categorical data.**

****3.3.3 User Input****

* **Real-time data, such as O₃, NO₂, PM₂.₅, and CO concentrations, are collected via a user-friendly Streamlit interface.**
* **The system validates the input and processes it through the same preprocessing steps used during model development.**

****3.3.4 Prediction Workflow****

* ****Model Prediction**: The processed user input is passed to the trained model to generate the predicted AQI values.**
* ****Transformation Reversal**: If necessary, log-transformed predictions are converted back to their original scale using inverse transformations (e.g., exponentiation).**

****3.3.5 Output Delivery**  
The predicted AQI values are displayed in a user-friendly interface, along with evaluation metrics:**

* ****R² Score****
* ****Mean Absolute Error (MAE)****
* ****Mean Squared Error (MSE)****

### **4. Technology Stack**

* ****Programming Language**: Python (for model building and deployment)**
* ****Libraries/Frameworks**:**
  + **Pandas (data manipulation)**
  + **NumPy (numerical operations)**
  + **Scikit-learn (model training and evaluation)**
  + **XGBoost (model training)**
  + **Streamlit (UI and user interaction)**
  + **Pickle (model saving/loading)**

### **5. Proposed Solution**

1. ****Data Collection**: Historical AQI data along with pollutant concentrations are collected and merged into a structured dataset.**
2. ****Data Preprocessing**: Missing values, categorical encoding, and scaling are performed to ensure compatibility with the model.**
3. ****Model Training**: Models like Gradient Boosting or XGBoost are trained using historical AQI data to predict future AQI levels.**
4. ****Real-Time Input**: The system collects real-time AQI data through a Streamlit-based UI for prediction.**
5. ****Prediction Output**: The system processes inputs and generates AQI predictions, which are then displayed in an easy-to-understand format.**

### **6. Model Training/Validation Workflow**

1. **Data collection and preprocessing.**
2. **Model training using XGBoost or Gradient Boosting.**
3. **Hyperparameter tuning with GridSearchCV.**
4. **Model evaluation using metrics like R², MSE, and MAE.**
5. **Model saving and deployment.**

### **7. User I/O Workflow**

1. ****Input Collection**: Users enter pollutant concentrations (O₃, NO₂, PM₂.₅, CO) and any optional weather data.**
2. ****Input Validation**: Check that all fields are filled in and inputs are correct.**
3. ****Prediction**: Generate AQI prediction using the trained model.**
4. ****Output Display**: Display AQI value along with the accuracy metrics.**
5. ****Data Storage**: Store input and prediction data in a database for future analysis.**
6. ****Visualizations**: Show trends, e.g., pollutant levels vs AQI over time.**

### **8. Exceptional Scenarios**

1. ****Missing Inputs**: Prompt the user to provide all required fields.**
2. ****Database Failure**: Log and retry the database connection.**
3. ****Invalid Input Format**: Notify the user of the incorrect input format.**
4. ****Model Not Found**: Display a default prediction and show a warning message.**

### **9. Key Performance Indicators (KPIs)**

1. ****Prediction Accuracy**: R² score, MSE, MAE.**
2. ****Latency**: Time taken to generate predictions for real-time inputs.**
3. ****Scalability**: The system’s performance under high data load.**
4. ****User Satisfaction**: Measured via user feedback.**
5. ****Database Efficiency**: Time taken for storing and retrieving predictions.**