Phase-3

Student Name: E Thirupathi

Register Number: 620123106119

Institution: AVS engineering college

Department: Electronics and communication engineering

Date of Submission: 15.05.2025

Github Repository Link: [Update the project source code to
your Github Repository]

---

1. Problem Statement

Rising air pollution affects human health, climate and ecosystems.

Existing monitoring systems focus on real-time data, noton predicting future pollution levels.

There is a need for a predictive system of forecast AQI (Air Quality Index) for early warnings and decision-making.

This project aims to usemachine learning models to accurately predict AQI based on environmental sensor data.

2. Abstract

The project develops an intelligent system that predicts AQI using advanced ML algorithms.

Data includes pollutants (e.g., PM2.5, NO2, CO) and meterological parameters (tempreture, humidity).

The workflow includes data collection, preprocessing, EDA, feature engineering, model building, evaluation, and deployment.

The system provides future air quality insights to assist urban planning, public

health, and policy decisions.

3.System Requirements

Hardware:

64-bit OS, 4GB+ RAM, dual-core processor

Software:

Python 3.8+

Jupyter Notebook / VS Code

Anaconda (optional)

Libraries:

pandas, numpy, matplotlib, seaborn, scikit-learn, xgboost, flask/streamlit (for deployment)

4.Objectives

Collect and preprocess historical air quality and weather data.

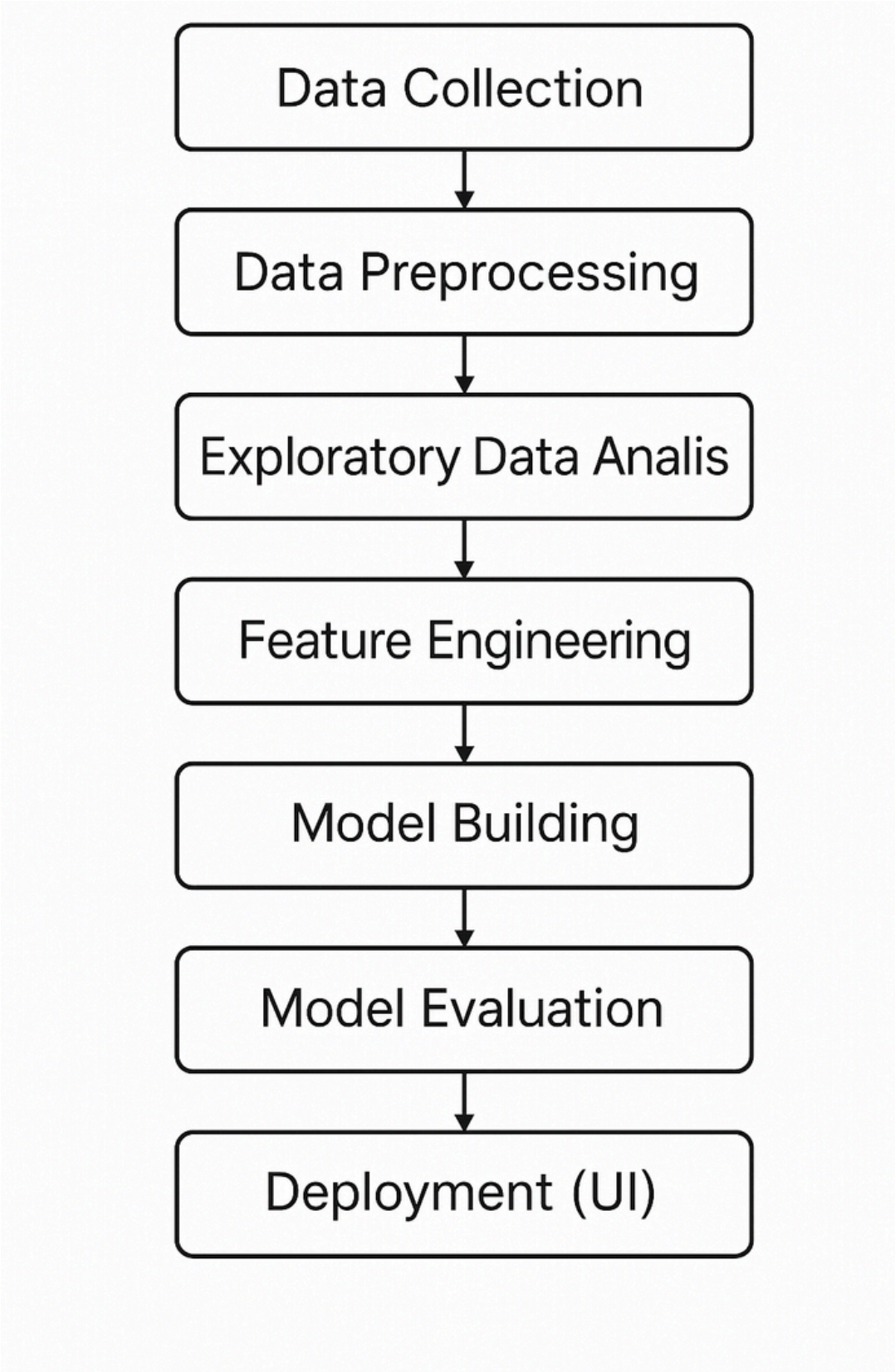
Perform EDA to understand patterns and pollutant impacts.

Engineer meaningful features to improve model accuracy.

Train and evaluate machine learning models for AQI prediction.

Deploy the best-performing model in a user-friendly interface.

5. Flowchart of Project Workflow


Data Collection
Data Preprocessing
Exploratory Data Analis
Feature Engineering
Model Building
Model Evaluation
Deployment (UI)

6. Dataset Description

Source: OpenAQ, Kaggle, UCI ML Repository, CPCB (India)

Key Features:

PM2.5, PM10, CO, NO_2, SO_2, O_3

Temperature, Humidity, Wind Speed

Date & Time

Target: Air Quality Index (AQI)

7. Data Preprocessing

Handling missing and null values. Removing duplicates and outliers. Date/time conversion and decomposition (e.g., extract hour, month) Scaling and normalization of numerical features.

8. Exporatory Data Analysis (EDA)

Histograms and boxplots for pollutant distribution.

Correlation matrix (heatmap) to assess feature relationships.

AQI trends over time (daily, weekly, monthly)

Pollutant concentration vs AQI level plots

9. Feature Engineering

Time features:hour, day, month, weekend.

Lag/rolling average features for pollutants.

Poluynomial and interaction terms.

Encoding categorical features (if present)

10. Model Building

Algorithm Used:

Linear Regression
Random Forest Regressor
XGBoost Regressor

Model Evaluation Matrics:

Mean Absolute Error (MAE).
Eoot Mean Squared Error (RMSE).
R^2 Score

Model Selection:

Cross-Validation.
Hyperparameter tuning using GridSearchCV

11. Model Evaluation

Metrics: MAE, RMSE, R2 Score

Cross-validation to test generalization.

Visualize actual vs predicted AQI.

Select best model for deployment.

12. Deployment

Tool: Flask or Streamlit

Build a web app that allows users to input features and get AQI.

Host locally or on platforms Like Heroku, Render, or Streamlit Cloud.

Optionally integrate with a real-time API or dashband.

13. Source code

```
import pandas as pd
from sklearn.model_selection import train_test_split
from xgboost import XGBRegressor
from sklearn.metrics import mean_squared_error

# Load dataset
df = pd.read_csv("air_quality_data.csv")

# Features and target
X = df.drop("AQI", axis=1)
y = df["AQI"]

# Split dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

# Train model
model = XGBRegressor()
```

```
model.fit(X_train, y_train)

# Predict and evaluate
y_pred = model.predict(X_test)
print("RMSE:", mean_squared_error(y_test, y_pred, squared=False))
```

14.Future scope

Integration with a real-time sensors using IoT.

Predicitive alerts for public health systems.

Mobile app to deliver real-time air quality preadictions.

Inclusion of satellite-based air quality data.

Use of LSTM or other deep learning models for time-series prediction.

13. Team Members and Roles

Saleth Harison J – Project Lead

Defined project scope, managed documentation, supervised the team.

Thirupathi E – Data Analyst

Handled data preprocessing, EDA, and visualization.

Mourish Kanna V – ML Engineer

Built and evaluated models, performed tuning and validation.

Sakthivel D – Web Developer

Deployed the model using Flask/Streamlit and built the user interface.