

A DISSERTATION ON
AIR QUALITY INDEX FOR URBAN AREAS

Submitted in partial fulfilment of the requirement for the
award of the degree of

MASTERS OF SCIENCE

in

Computer Science (Big Data Analytics)

Submitted by

Panchajanya Sarkar

2021MSBDA028

Under the Guidance of

Nitesh Yadav

Jr. Data Scientist, Celebal Technologies

&

Dr. Pritpal Singh

Assistant Professor

Department of Data Science and Analytics



Department of Data Science and Analytics
School of Mathematics, Statistics and Computational Science

CENTRAL UNIVERSITY OF RAJASTHAN

August 2023

DECLARATION

I certify that

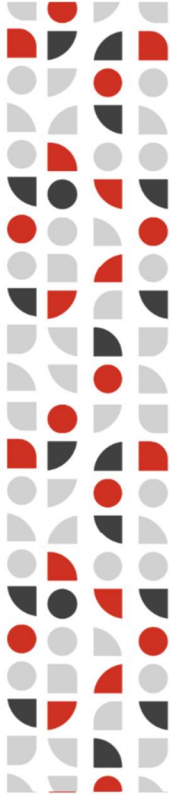
- a. The work contained in the dissertation has been done by myself under the supervision of my supervisor.
- b. The work has not been submitted to any other Institute for any degree or diploma.
- c. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.
- d. Whenever I have used materials (data, theoretical analysis, and text) from other sources, I have given due credit to them by citing them in the text of the dissertation and giving their details in the references.
- e. Whenever I have quoted written materials from other sources and due credit is given to the sources by citing them.

Date: August 8, 2023

Place: Ajmer

Panchajanya Sarkar

2021MSBDA028



CERTIFICATE OF INTERNSHIP

This Certificate is Proudly awarded to :

Panchajanya Sarkar

in recognition of your outstanding completion of **Celebal Summer Internship Program** at Celebal Technologies under **Data Science** domain for the duration of **27th May 2023 to 25th July 2023**.

You are found to be hardworking, enthusiastic and diligent during your internship. We highly appreciate your accomplishments and wish you the best for the future.

Sharthak Acharjee
Senior Manager - HR



08-08-2023

CERTIFICATE

This is to certify that the project entitled “**Air Quality Index for Urban Areas**” was carried out by **Panchajanya Sarkar (2021MSBDA028)** at **Celebal Technologies, Jaipur** under my guidance during **May, 2023** to **July, 2023**.

.....

Nitesh Yadav

Jr. Data Scientist
Celebal Technologies, Jaipur

DEPARTMENT OF DATA SCIENCE AND ANALYTICS
CENTRAL UNIVERSITY OF RAJASTHAN, INDIA

08-08-2023

CERTIFICATE

This is to certify that the project titled “**Air Quality Index for Urban Areas**” is a record of bonafide work done by **Panchajanya Sarkar** (2021MSBDA028) submitted in partial fulfilment of the requirements for the award of the Degree of Masters of Science (M.Sc.) in Data Science and Analytics of Central University of Rajasthan, during the academic year 2022-23.

Dr. Pritpal Singh

Supervisor,

DEPARTMENT OF DATA SCIENCE AND ANALYTICS

Central University of Rajasthan

Dr. Vidyottama jain

HOD,

DEPARTMENT OF DATA SCIENCE AND ANALYTICS

Central University of Rajasthan

ACKNOWLEDGEMENT

I would like to thank **Dr. Pritpal Singh** (Assistant Professor) and all faculty members at the Department of Data Science and Analytics, Central University of Rajasthan for providing me an opportunity to work on the topic. I will forever be indebted to **Dr. Vidyottama Jain** (Head of the Department), who was always there to help me and provide me the little advices on the minute aspects of the project. To all my batch mates of the Department of Computer Science, thank you all for being there for me always! Finally, I would like to thank my parents for their constant support and guidance.

Panchajanya Sarkar

Semester IV
Department of Data Science and Analytics
Central University of Rajasthan

ABSTRACT

Job training/Internships are the intrinsic part of the curriculum of a Master of Science Computer Science (BDA). Students who are pursuing this course are required to undergo training of 4-6 months in a corporate firm. It is outlined to make students familiar with the business environment and situations. It helps students in applying their theoretical knowledge of this specific course to real life problems and helps them to learn about the corporate culture and system.

As for the curriculum of my degree, I undertook my internship in Celebal Technologies as a **Data Science Intern**. Duration of my internship was from May 2023 to July 2023.

During this internship, my major time was spent on working on Feature Engineering, Model Testing and UI creation part of the product. For the documentation part, I chose the same topic to make my report. This project report is a synopsis of all the work and knowledge I accumulated during my internship period.

During this internship period, I came across various tools like -

- git
- gradio
- docker
- vscode, etc..

These tools (not limited to) helped me to work on my project and they served fruitful.

ABOUT THE ORGANISATION

Celebal Technologies is a premier software services company in the field of Data Science, Big Data, and Enterprise Cloud. We help you achieve a competitive advantage with intelligent data solutions, built using cutting-edge technology. Our core offerings are around Cloud Innovation, Supply Chain Analytics, Chat Bots, Power Platforms, and Data Analytics. Our solutions can help you accelerate decision-making and take giants leaps in your digital transformation journeys.

The mission of the company is to make data simple and easy to understand for all organizations. We are committed to providing solutions powered by modern cloud and artificial intelligence that integrate with traditional enterprise software. Our tailor-made solutions help enterprises maximize productivity, improve speed and accuracy.

It believes that the ever-evolving technology and analytics landscape will fundamentally transform service and solutions delivery across all sectors and industries, with domain expertise driving this change.

The senior partners come with 20+ years of experience in the industry and have worked for most of the Fortune 500 companies across key analytical areas in the US, Europe, Middle East, Asia, and Africa. It is here to be your partners throughout your transformational journey in **Celebal Technologies**.

It is currently leveraging Machine Learning, Data Science and Cloud Infrastructure to transform existing process and services delivery.

Website – <https://celebaltech.com>

Industry: IT Services and IT Consulting

Company Size: 2000+

Founded: 2016

Specialties:

Risk, Compliance, Data Analytics, Machine Learning, Data Engineering, Credit Risk, AML, FCC, Artificial Intelligence, Transformation, Regulatory Compliance, Model Risk, Quantitative Risk, Market Risk, Technology, financial Crime, Digitization, Algorithms, Model Validation, Modeling, and Performance Monitoring

CONTENTS

Chapter 1 – Introduction

- 1.1 – Introduction and Motivation 11
- 1.2 – Objectives of the Product13

Chapter 2 – Background Material

- 2.1 – Conceptual Overview15
- 2.2 – Technologies Involved17

Chapter 3 – Methodology

- 3.1 – Detailed Methodology22
- 3.2 – Circuit Layouts / Block Diagram30

Chapter 4 – Implementation

- 4.1 – Modules and Basic Implementation33
 - 4.1.1 – Environment Setup33
 - 4.1.2 – Data Import from Source34
 - 4.1.3 – Exploratory Data Analysis34
 - 4.1.4 – Train and Test Split39
 - 4.1.5 – Lazy Predictor on Regression Models40
 - 4.1.6 – Take User Input and Display predictions42
 - 4.1.7 – User Interface over the Web43
- 4.2 – Prototypes of the Project44**
 - 4.2.1 – Home Page Prototype44
 - 4.2.2 – Map View Prototype45
 - 4.2.3 – AQI Trends46

Chapter 5 – Result and Analysis	
5.1 – Result Presentation	47
5.2 – Data Analysis and Interpretation	47
5.3 – Interpretation and Insights	48
5.4 – Future Recommendation	49
Chapter 6 – Conclusion and Future Scope	
6.1 – Conclusion and Future Scope	50
Chapter 7 – References	52

Chapter 1

INTRODUCTION

1.1 : Introduction and Motivation

Air quality is a critical concern in urban environments, directly affecting the health and well-being of residents. The "Air Quality Prediction for Urban Areas" project aims to develop an accurate and efficient predictive model for assessing air quality levels in densely populated cities. By leveraging advanced machine learning techniques and real-time data, this project seeks to empower city planners, residents, and policymakers with actionable insights to mitigate air pollution's adverse effects.

Regardless of what problem we are solving, an interpretable model will always be preferred because both the end-user and your boss/co-workers can understand what our model is doing. Model Interpretability also helps us debug our model by giving us a chance to see what the model thinks is essential.

Furthermore, we can use interpretable models to combat the common belief that Machine Learning algorithms are black boxes and that we humans aren't capable of gaining any insights into how they work.

Even today, data science and machine learning applications are still perceived as black boxes capable of magically solving a task that couldn't be solved without it. However, this isn't at all true; then, for a data science project to succeed, the development team must understand the problem they are trying to solve and know what kind of model they need for the problem at hand.

However, considering that most business people won't have an intuitive understanding of the machine learning pipeline and therefore won't understand our fancy accuracy metrics or loss functions, we need another way to show them our model's performance.

Furthermore, good performance doesn't always mean that our model is doing the right thing, and therefore often, the following questions emerge:

Why should I trust the model?

How does the model make predictions?

Data scientists and researchers have been trying to answer these questions for several years now and have come up with multiple approaches to extract information about the decision process of machine learning models.

Some of these methods are model-specific, while others work for every model no matter the complexity. Speaking of complexity, every data scientist will know of the model interpretability vs. model performance trade-off, which says if we increase the complexity of the model, it will get harder to interpret it correctly.

In a subsequent sections of this report, we will delve into the methodologies used, data analysis, predictive modelling, web dashboard development, results, and future directions of the "Air Quality Index for Urban Areas" project. Through these efforts, we aim to realize a more informed, healthier, and environmentally conscious urban community.

1.2 : Objectives of the Product

The project “**Air Quality Analysis for Urban Areas**” has the objective to develop a model that can predict air quality in urban areas. The project involves in various stages, including collecting a proper dataset from a reliable source, Exploratory Data Analysis (EDA), training a machine learning model and saving the model for future use.

In brief, the objectives are discussed below, indexed -

- Identify a reliable source of Air Quality Data for urban areas. Scraped data over a longer period of time or from a reliable source is accepted. The data shall include attributes such as pollutant concentrates (PM2.5, PM10, CO, NO2 etc.), meteorological conditions and timestamp
- Exploratory Data Analysis, abbreviated as EDA is performed on the data for performing data cleaning and preprocessing to handle missing values, outliers and inconsistent data across the dataset.

Visualizations are an important part of representing data. Visualize the data using appropriate graphs and plots to understand the relationship between different features and air quality.

- Feature Engineering is useful for selecting relevant features that are likely to influence air quality. Feature engineering is performed to encode categorical values, handling numerical features and creating new derived features if necessary.
- Machine Learning Model training is an important part of the project. The following steps are performed to incorporate the machine learning process -
 - Split the dataset into training and testing sets
 - Choose an appropriate Machine Learning algorithm (regression) for prediction.

- Train the model using the testing and training data
- Fine-tune the model by adjusting hyper-parameters to improve its performance.
- Model Evaluation and Deployment
 - Evaluate the trained model's performance using suitable metrics like Mean Squared Error or R-Squared
 - Develop a simple web interface or CLI based application where users can give inputs and get predictions based on model

Chapter 2

BACKGROUND MATERIAL

2.1 : Conceptual Overview (Concepts / Theory used)

A conceptual overview provides a high-level understanding of the key concepts, ideas, and relationships within a particular topic, system, or project. It serves as an introductory guide that helps readers or stakeholders grasp the fundamental elements without diving into intricate details.

The "Air Quality Index for Urban Areas" project is centered on addressing the critical concern of air quality in densely populated cities. This conceptual overview provides a high-level understanding of the project's objectives, scope, and significance.

A briefed and indexed conceptual overview of this project -

- **Key Concepts**
 - **Air Quality Index (AQI):** A composite measure that quantifies the levels of common air pollutants, such as PM2.5, PM10, NO2, SO2, and O3, into a single numerical value. The AQI provides a standardized way to communicate the potential health risks associated with different air quality levels.
 - **Urban Areas:** High-density urban environments characterized by complex interactions between human activities, emissions, weather conditions, and pollution sources.
 - **Prediction and Monitoring:** Utilizing advanced machine learning techniques to predict air quality levels and real-time monitoring of air pollutants.

- **Relationships**

The project aims to develop accurate predictive models that utilize historical air quality data, meteorological conditions, and various influencing factors to estimate future air quality levels in urban areas. These models will enable the creation of real-time Air Quality Index (AQI) values, allowing city residents, planners, and policymakers to make informed decisions about outdoor activities, health protection measures, and pollution control strategies.

- **Purposes**

The primary purpose of this project is to provide urban residents and stakeholders with timely and reliable air quality information. By predicting air quality levels and presenting them through a simplified Air Quality Index, the project seeks to raise awareness about pollution hazards, promote healthier lifestyle choices, and facilitate data-driven urban planning decisions.

- **Scope**

The project encompasses data collection, model development, and user interface design. It focuses on urban areas, recognizing the complex interplay between pollution sources, meteorological conditions, and human behavior. The scope extends to creating a user-friendly web-based dashboard that displays real-time AQI information, making it easily accessible to residents.

- **Significance**

Urban air quality has a direct impact on public health and quality of life. The project's significance lies in its potential to empower individuals to make informed choices regarding outdoor activities and exposure to pollutants. Additionally, it provides city planners and policymakers with reliable data to implement effective pollution control measures, ultimately contributing to improved air quality, healthier communities, and sustainable urban development.

In conclusion, the "Air Quality Index for Urban Areas" project addresses the pressing need for accessible and accurate air quality information in urban environments. By developing predictive models and a user-friendly interface, the project endeavors to

enhance awareness, health, and environmental well-being in cities.

2.2 : Technologies involved

The "Air Quality Index for Urban Areas" project involves the integration of several technologies to achieve its objectives. Here are some of the key technologies that could be involved in the project:

- **Development Environment**

- *Jupyter Notebook*

- An interactive web application called Jupyter Notebook is used for things like research, coding, and data analysis. It enables us to create documents that include text explanations, live code, and visualizations. We can add formatted text to “markdown cells” and write and execute code in various programming languages inside of “code cells.” The platform supports interactive reports and tutorials and makes it simple to share dynamic visualizations. Data scientists, researchers, and teachers like Jupyter Notebook because of its adaptability and collaborative features.

- *Visual Studio Code (IDE)*

- The user-friendly interface of Visual Studio Code comes with a variety of tools like syntax highlighting, code completion, and built-in Git support. Extensions that improve functionality for different programming languages, frameworks, and tools make it highly customizable. Because of its speed, flexibility, and ability to work in various coding environments, VS Code is a favorite among programmers and developers. It also offers debugging features, an integrated terminal, and a rich ecosystem of extensions.

- *IPython (Kernel)*

- IPython (Interactive Python) is a command shell for interactive computing in multiple programming languages, originally developed for the Python programming language, that offers introspection, rich

media, shell syntax, tab completion, and history. *IPython provides the following features:*

- Interactive shells (terminal and Qt-based).
- A browser-based notebook interface with support for code, text, mathematical expressions, inline plots and other media.
- Support for interactive data visualization and use of GUI toolkits.
- Flexible, embeddable interpreters to load into one's own projects.
- Tools for parallel computing.

IPython is a NumFOCUS fiscally sponsored project

Next page, we will see some logos of the following products.



Fig 2.2.1 Logos of Jupyter, VSCode and IPython

- **Machine Learning**

- Machine Learning algorithms for predicting air quality levels based on historical data, meteorological data, meteorological conditions and other relevant factors.

- *Linear Regression*

In machine learning and statistics, linear regression is a fundamental supervised learning algorithm. Using one or more input features, it predicts a continuous numerical output. Finding the best-fitting linear relationship between the input features and the output variable is the main goal of linear regression.

$$Y = \beta_0 + \beta_1 * X_1 + \beta_2 * X_2 + \dots + \beta_n * X_n + \epsilon$$

$$Cost\ Function(MSE) = \frac{1}{n} \sum_{i=0}^n (y_i - y_{i\ pred})^2$$

Replace $y_{i\ pred}$ with $mx_i + c$

$$Cost\ Function(MSE) = \frac{1}{n} \sum_{i=0}^n (y_i - (mx_i + c))^2$$

Fig 2.2.2 – Cost Functions and Equation of Linear Regression

- Data Pre-processing techniques to clean and transform raw data into usable formats for modelling.
- **Environmental Data collection**
 - Environmental sensors for collecting real-time air quality and meteorological factors
 - IoT devices to gather various data from various locations within the urban area.
- **Geographical Information System**
 - GIS technology to map and visualize air quality data spatially, showing variations across different areas of the city.
- **Web Development**
 - Web development frameworks (such as Django, Flask, or Node.js) to create a user-friendly web-based dashboard for displaying real-time air quality information and AQI values.



Fig 2.2.3 Logos of few frameworks

- Front-end technologies (Gradio) for designing the dashboard's user interface.



Fig 2.2.4 Logo of Gradio

- **Database Management**

- Database management for storing and managing large volumes of historical and real-time air quality data. e.g. - MongoDB, SQL



Fig 2.2.5 Logos of MongoDB and SQL

- **Cloud Computing**

- Cloud Platforms like Google Cloud, AWS, Microsoft Azure, Huggingface for scalable storage, processing and hosting the predictive models.



Fig 2.2.6 Logos of GCP, AWS, Azure and Huggingface

- **Data Visualization libraries**
 - Data visualization libraries like Matplotlib for creating interactive graphs, charts, maps to present air quality trends and predictions.
- **API Integration**
 - Integration of APIs from weather services, pollution monitoring agencies and other relevant sources to gather external models for model inputs.
- **Model Deployments**
 - Deployment frameworks like Huggingface actions, Docker etc. to deploy and manage machine learning models to production environments.



Fig 2.2.7 Logo of Docker

- **Version Control**
 - Version control system like Git to manage code changes and collaborations among project team members.



Fig 2.2.8 Logo of git

The combination of these technologies enables the creation of an integrated system that predicts, visualizes, and communicates air quality information to urban residents and stakeholders. The technologies work in tandem to provide actionable insights, promote public awareness, and support informed decision-making for a healthier urban environment.

Chapter 3

METHODOLOGY

3.1 : Detailed Methodology with diagrammatic representation

Methodology refers to the systematic set of principles, practices and procedures that guide a particular area of study, research or problem solving. It outlines how tasks should be carried out, including the techniques, tools and steps involved. Essentially, it is the framework that ensures consistency, reliability and validity in achieving the desired outcomes and results.

A flowchart of the steps involved in the methodology:

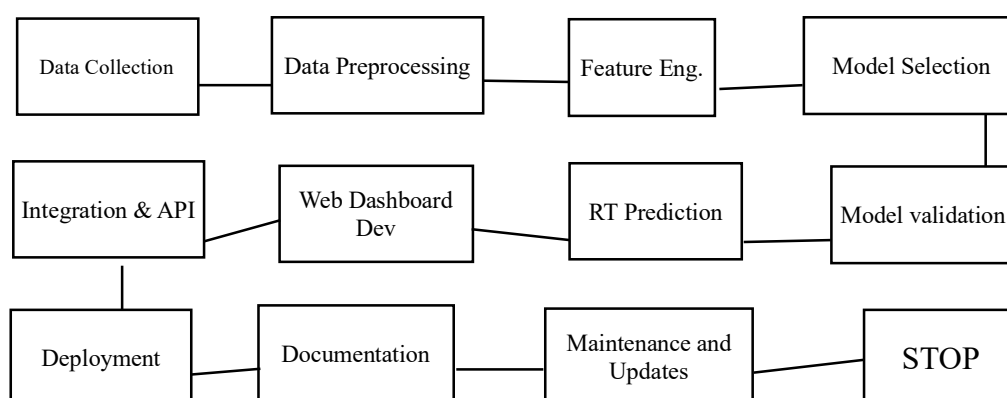


Fig 3.1.1 Flowchart of Methodology

A detailed breakdown of the methods that will be adopted for the “Air Quality Index for Urban Areas” project:

- **Data Collection**

Data Collection refers to collecting data from various sources, like via web scraping or from trusted sources like meteorological websites or from prebuilt datasets available across the internet. Data are of two types :

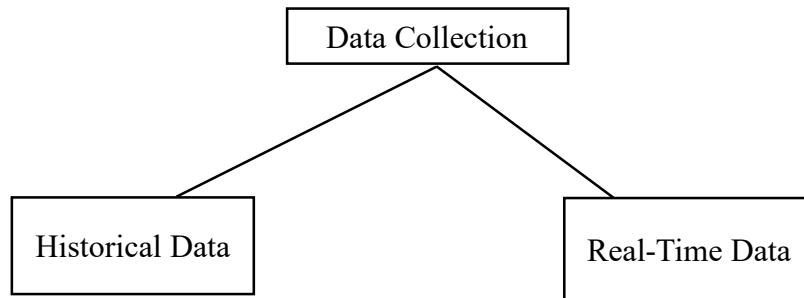


Fig 3.1.2 Diagram of Data Collection

- **Historical Data:** Obtain historical air quality from government agencies or research organizations, covering a significant period to capture seasonal and annual variations
- **Real-time data:** Deploy a network of environmental sensors across the urban area to continuously collect real-time air quality measurements. These sensors will cover various parts of the city to capture spatial variations.

- **Data Pre-processing**

Data Pre-processing refers to processing the data before working on it. It refers to various methods of handling data mentioned below in a diagram -

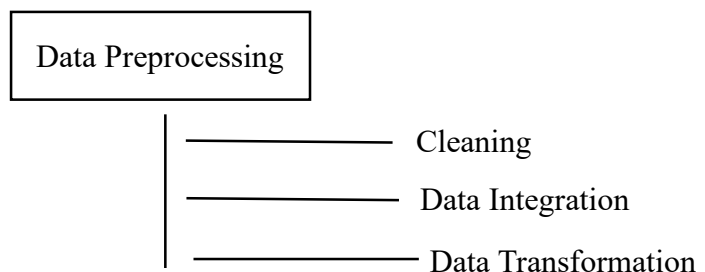


Fig 3.1.3 Diagram of Data Pre-processing

- **Cleaning** : Handle missing values, outliers and inconsistencies using techniques like interpolation of data or data imputation.
- **Data Integration** : Combine Air Quality data with meteorological data (temperature, humidity and wind speed) obtained from various weather stations.
- **Data Transformation** : Normalize or standardize the data to ensure that different variables are on similar scales.

- **Feature Engineering**

Feature Engineering refers to the working on various features and extracting various informations from it.

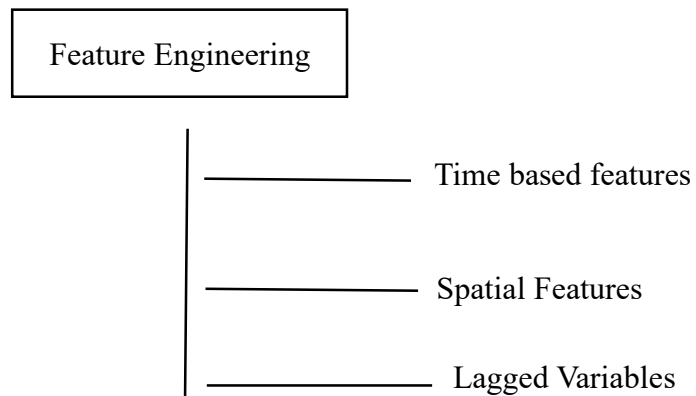


Fig 3.1.4 Diagram of Feature Engineering

- **Time Based Features** : Create features that capture temporal patterns such as time of the day, day of the week and season.
- **Spatial Features** : .Integrate geographical features like distance from major roads, land usage and population density.
- **Lagged variables** : Include lagged values of air quality parameters to account for past influences.

- **Model Selection and Development**

Model selection refers to selection of models and figuring out whether to use Classification, Regression or Time-Series Models for forecasting or predicting.

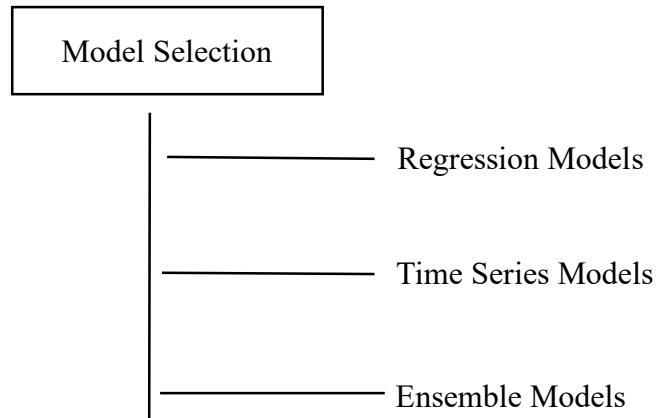


Fig 3.1.5 Diagram of Model Selection

- **Regression Models** : Employ regression techniques like Linear Regression, Lasso or Ridge Regression to predict Air Quality Parameters
- **Time Series Models** : Utilize time series forecasting models such as ARIMA or LSTM to capture temporal dependencies in the data.
- **Ensemble techniques** : Combine the outputs of multiple models for improved prediction and accuracy.

- **Model Validation**

Model validation refers to splitting the models into training and testing datasets to perform the machine learning algorithms on it.

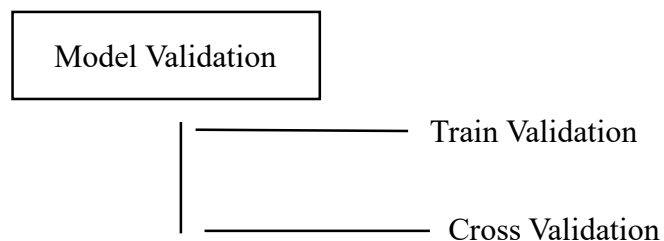


Fig 3.1.6 Diagram of Model Validation

- **Train Validation** : Divide the historical data into training and validation sets for model training and performance evaluation, respectively.
- **Cross Validation** : Apply k-Fold cross validation to ensure robustness of the models and avoid overfitting.

- **Real Time Prediction**

Real Time prediction is predicting the air quality over a certain period of time to check whether the output is correct and based or not.

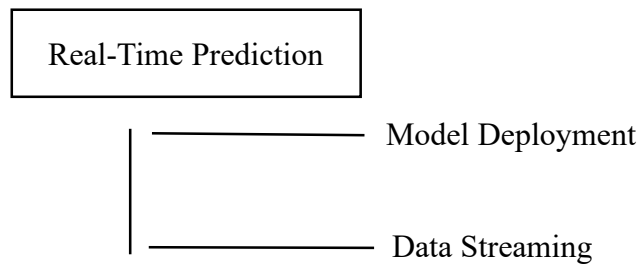


Fig 3.1.7 Diagram of RT Prediction

- **Model Deployment** : Deploy the trained models on a cloud platform using technologies like Flask to facilitate real-time prediction.
- **Data Streaming** : Set up a data pipeline that streams real-time air quality and meteorological data to the deployed models.

- **Web Dashboard Development**

To display the results or what we call predictions we need to display the results into a user-friendly dashboard. A web dashboard is the best choice.

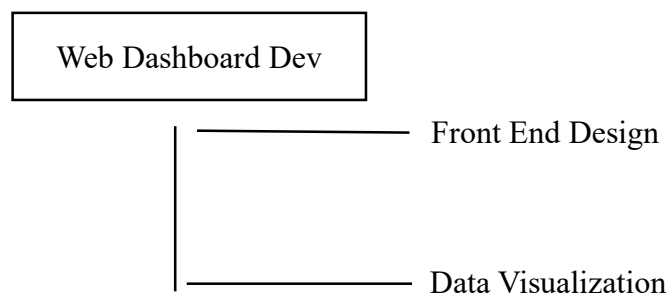


Fig 3.1.8 Diagram of Web Dashboard

- **Front-End Design** : Create an interactive and intuitive web-based dashboard using HTML and CSS, Javascript, React
- **Data Visualization** : Integrate data visualization libraries like Plotly, Leaflet to display real-time AQI values, trends and geographical distributions.

- **API Integration**

API stands for Application Platform Interface which is basically used for fetching weather details from various sources. API integration is a must for AQI since it is not possible for an end user to collect all the data in a standalone environment.

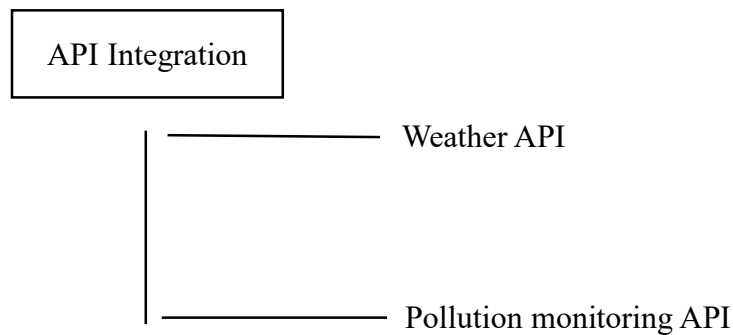


Fig 3.1.9 Diagram of API Integration

- **Weather API** : Integrate weather APIs to fetch real-time meteorological data and enhance the accuracy of predictions.
- **Pollution monitoring APIs** : Incorporate data from pollution monitoring agencies to validate model predictions and ensure data accuracy.

- **Deployment**

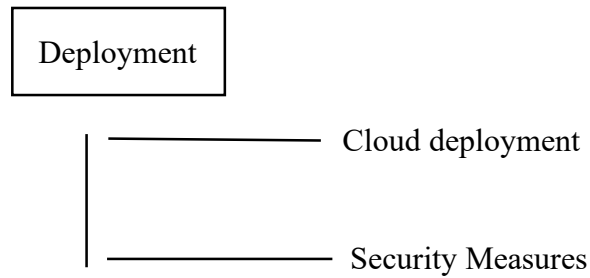


Fig 3.1.10 Diagram of Deployment

- **Cloud Deployment:** Deployment is widely used for cloud deployments, where the web dashboard is deployed along with predictive models on a cloud platform e.g. AWS, Azure, GCP etc. to ensure scalability
- **Security Measures :** Implement encryption, secure authentication and authorization

- **Documentation**

Documentation refers to writing a detailed guide on the product. It is a very important part of development.

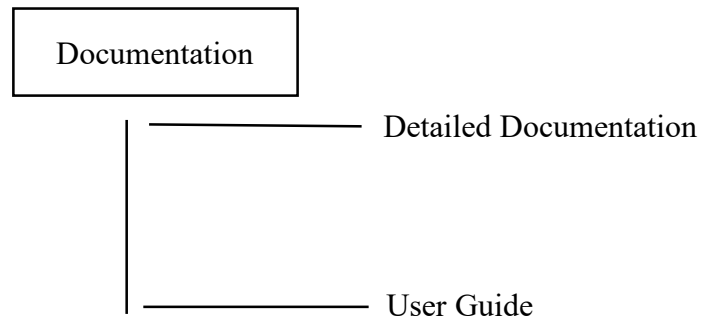


Fig 3.1.11 Diagram of Documentation

- **Detailed Documentation :** Prepare comprehensive documentation detailing the projects methods, algorithms, data sources, pre-processing steps, model training and deployment process.

➤ **User guide** : Create a user guide that explains how to use the web dashboard and interpret AQI values.

- **Maintenance and Updates**

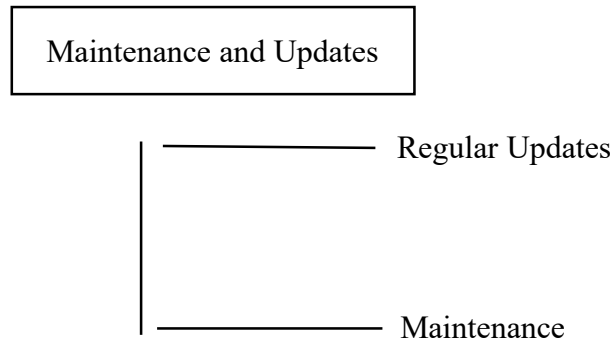


Fig 3.1.12 Diagram of maintenance and Updates

- Regularly updates the predictive models to ensure they reflect changing air quality patterns.
- Maintain the dashboard, addressing any issues, bugs or necessary updates.

Through the adaptation of these detailed methods, the **Air Quality Index for Urban Areas** project aims to develop a robust and user-friendly system that predicts air quality levels, calculates AQI values and empowers residents and stakeholders with actionable insights for healthier urban livings.

3.2 : Circuit Layouts / Block Diagrams

- **Data Collection and Prediction**

Here is a Block diagram of **Data Collection and Prediction** and the diagram has the following graphical characteristics -

- **Big Square (centre)** for the main feature
- **Small Square(s)** for sub-features

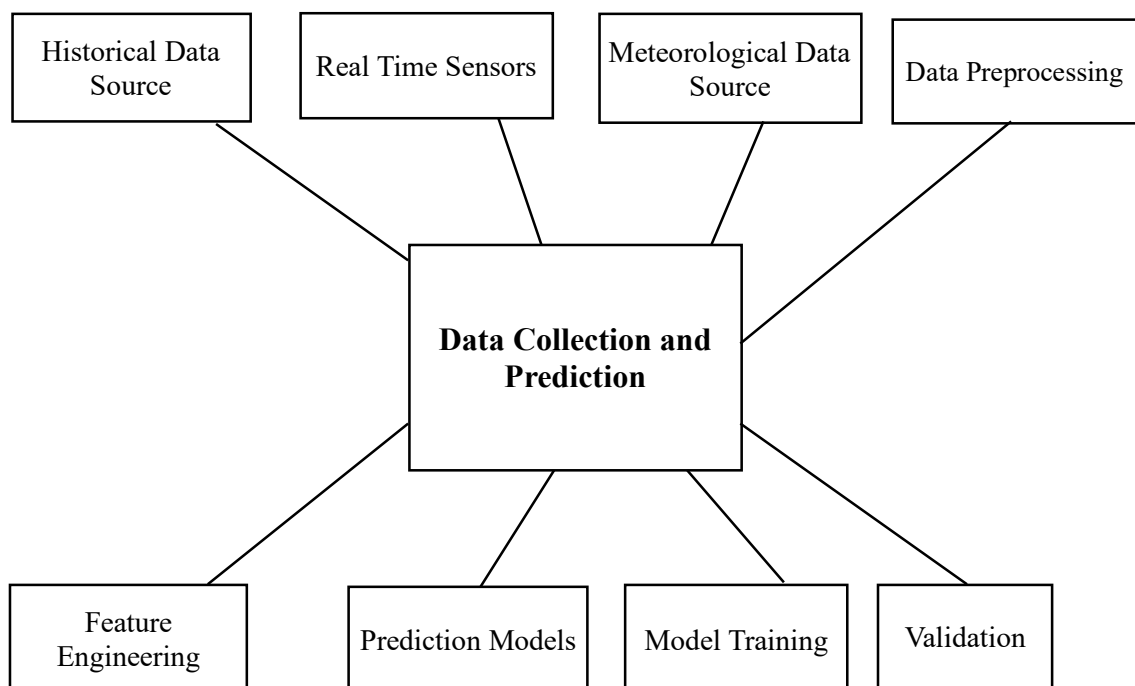


Fig 3.2.1 Block Diagram of Data Collection and Prediction

- **Real-Time Prediction and Dashboard**

Here is a Block diagram of **Real-Time Prediction and Dashboard** and the diagram has the following graphical characteristics -

- **Big Square (centre)** for the main feature
- **Small Square(s)** for sub-features

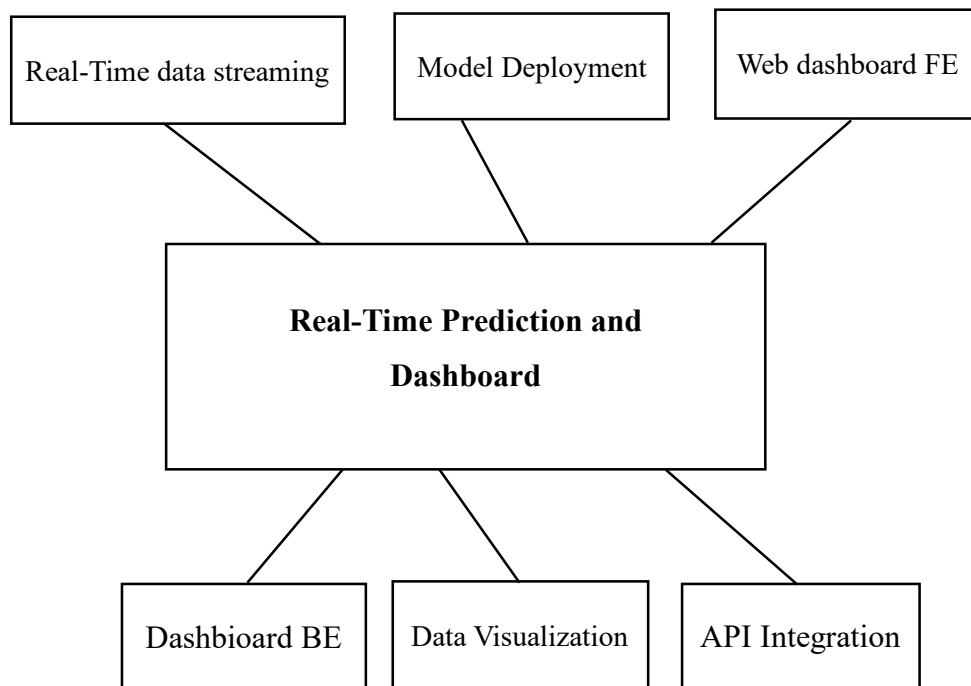


Fig 3.2.2 Block Diagram of RealTime Prediction and Dashboard

- **User Testing and Deployment**

Here is a Block diagram of **User testing and deployment** and the diagram has the following graphical characteristics -

- **Big Square (centre)** for the main feature
- **Small Square(s)** for sub-features

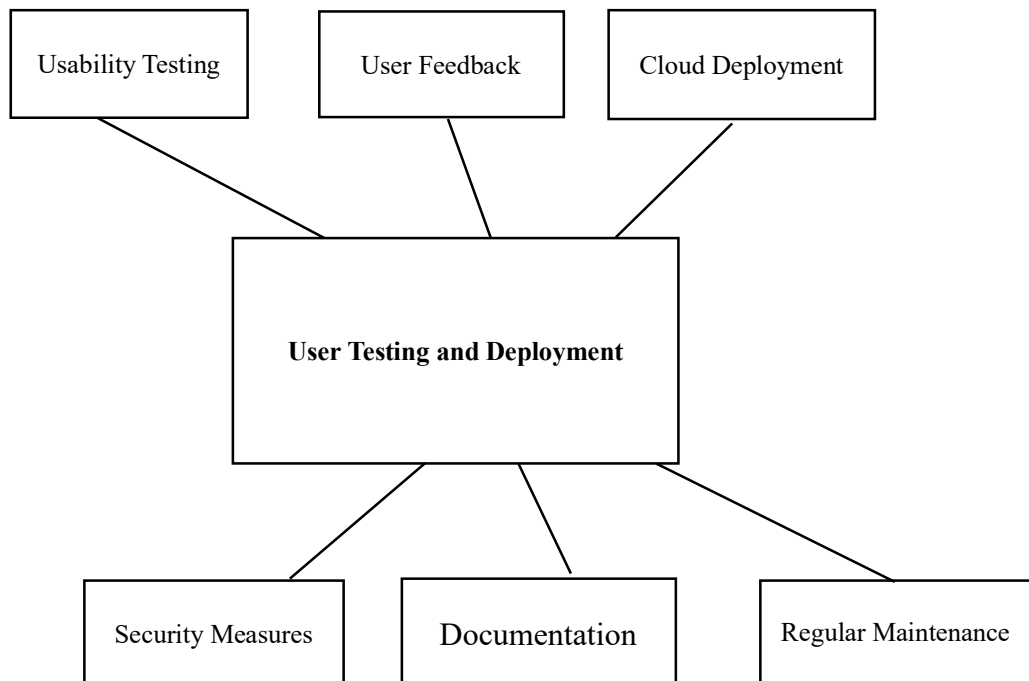


Fig 3.2.3 Block Diagram of User Testing and Deployment

Chapter 4

IMPLEMENTATION

4.1 : Modules and Basic Implementation

This project's implementation requires a number of different steps and pipelines along with components. This consists of the following steps -

- Environment setup
- Data Import from source
- Exploratory Data Analysis
- Split Train and Test data
- Implement Lazy Predict to run various regression models
- Take user input and display predictions
- Web Output

◦ 4.1.1 : *Environment Setup*

Environment setup is the first step of creating a supported environment for developing and working on it. There is a requirement file in text format which holds the libraries required for the project. A sample screenshot is given below.

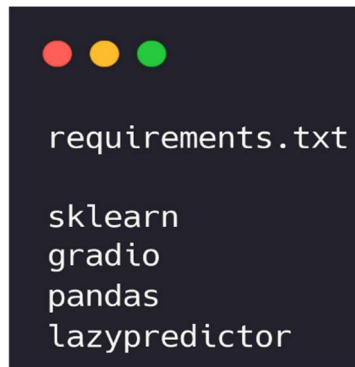


Fig 4.1.1 Requirements

The following requirements are installed by **pip**, running the following command

```
pip install -r requirements.txt
```

This command will install the libraries and the environment shall be ready.

◦ 4.1.2 : Data Import from the source

Data Import is done by pandas, using a CSV file which holds the collected data and it is read in the environment and saved in a given variable.

The command for reading the data is

```
pandas.read_csv(<path_to_data>, encoding = <encoding>,  
low_memory=<bool>)
```

```
# read the data from the csv file located inside datasets folder  
df = pd.read_csv('datasets/data.csv', encoding = "ISO-8859-1", low_memory=False)
```

Fig 4.1.2 Reading the data from source

◦ 4.1.3 : Exploratory Data Analysis

EDA is crucial step in understanding and gaining insights from your data before diving into modelling and analysis. For the **Air Quality Index** project, EDA helps us to understand the characteristic of the air quality and meteorological data. The basic steps of conducting EDA on the dataset are -

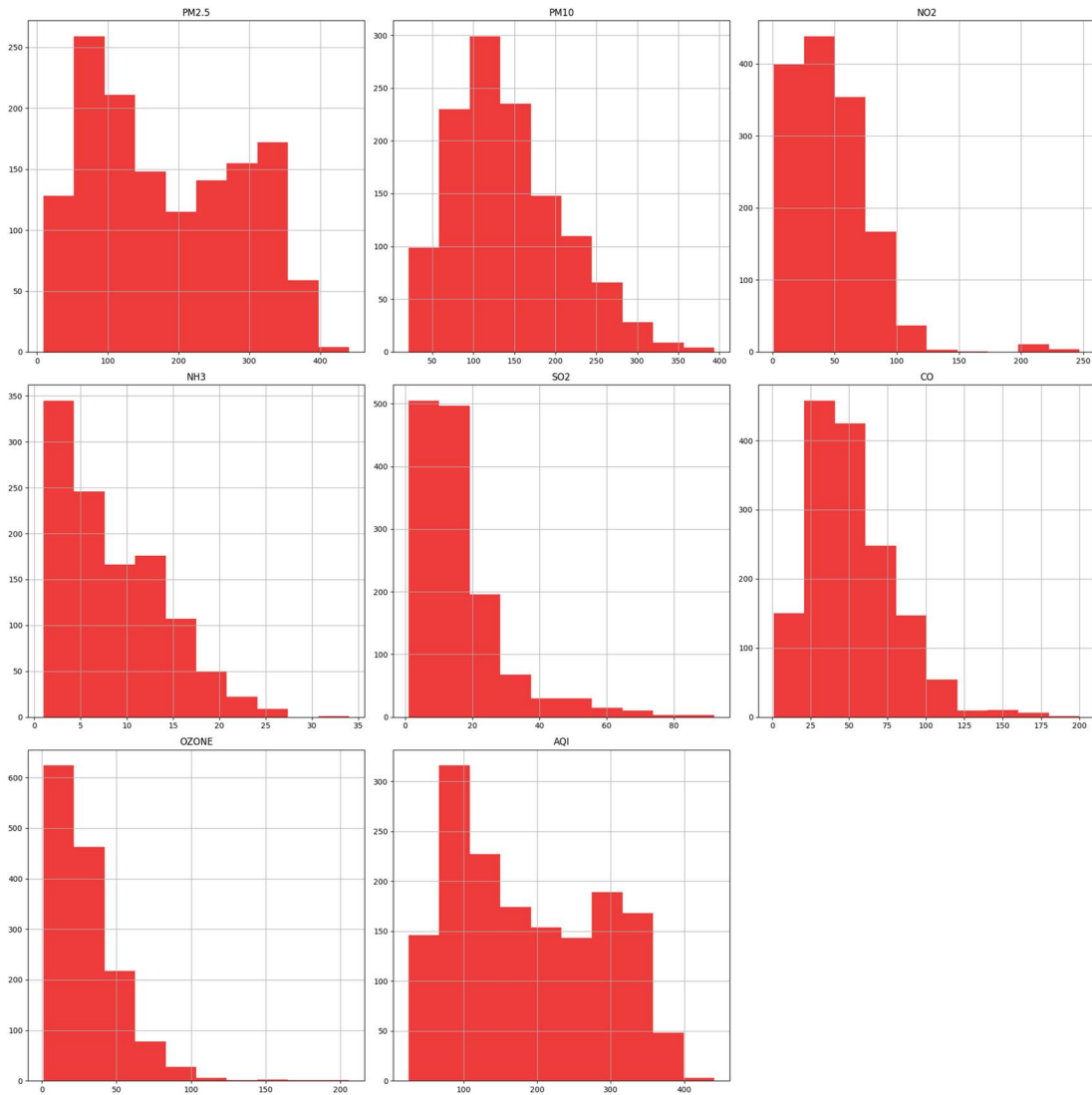
- *Data Summary*
 - Use summary stats like mean, median SD to get a sense of central tendency and detect outliers, spread of the data
 - Summarize the outliers and fix if any

```
In [97]: # check the shape of the dataframe  
print("There are (df.shape[0]) rows and (df.shape[1]) columns in the dataframe")  
  
There are 435742 rows and 13 columns in the dataframe  
  
In [98]: # check the basic info of the dataframe  
df.info()  
  
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 435742 entries, 0 to 435741  
Data columns (total 13 columns):  
#   Column                Non-Null Count  Dtype    
---  ---                  
0   stn_code                291655 non-null object    
1   sampling_date           435739 non-null object    
2   state                   435742 non-null object    
3   location                435739 non-null object    
4   agency                  286261 non-null object    
5   type                   438369 non-null object    
6   so2                     481896 non-null float64   
7   no2                     419989 non-null float64   
8   rspm                    385558 non-null float64   
9   spm                     198355 non-null float64   
10  location_monitoring_station 488251 non-null object    
11  pm2_5                   9314 non-null  float64   
12  date                   435735 non-null object    
dtypes: float64(5), object(8)  
memory usage: 43.2+ MB  
  
In [99]: # check the type of data for each column  
df.dtypes  
  
out[99]: stn_code                object  
sampling_date           object  
state                   object  
location                object  
agency                  object  
type                   object  
so2                     float64  
no2                     float64  
rspm                    float64  
spm                     float64  
location_monitoring_station object  
pm2_5                   float64  
date                   object  
dtype: object
```

Fig 4.1.3 Data Summary View

- *Data Visualization*

- Create visualization to better understand the distributions of air quality parameters.
- Few images of the visualization are presented below -

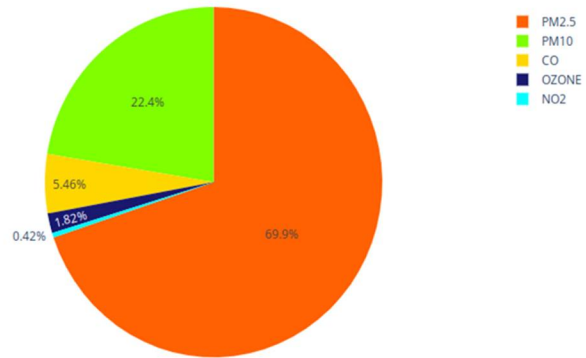


```
1 import matplotlib.pyplot as plt
2 df.hist(figsize = (20,20),color='#EE3B3B')
3 plt.tight_layout()
4 plt.show()
```

Fig 4.1.4 Histogram of the dominant columns and the code

- Predominant character means the columns which are highly correlated to the Air Index Quality
- The next image shows a Pie Chart of the dominance of the predominant characters alongwith the code

Predominant_Parameter Count



```

1 import plotly.express as px
2 Predominant_Parameter = ['PM2.5', 'PM10', 'CO', 'OZONE', 'NO2']
3 val_counts = [998, 320, 78, 26, 6]
4
5 fig = px.pie(values=val_counts, names=Predominant_Parameter,
6             color=Predominant_Parameter,
7             color_discrete_map={'PM2.5': '#FF6103',
8                               'PM10': '#7FFF00', 'CO': '#FFD700', 'OZONE': '#191970', 'NO2': 'aqua'},
9             title='Predominant_Parameter Count')
10
11 fig.show()

```

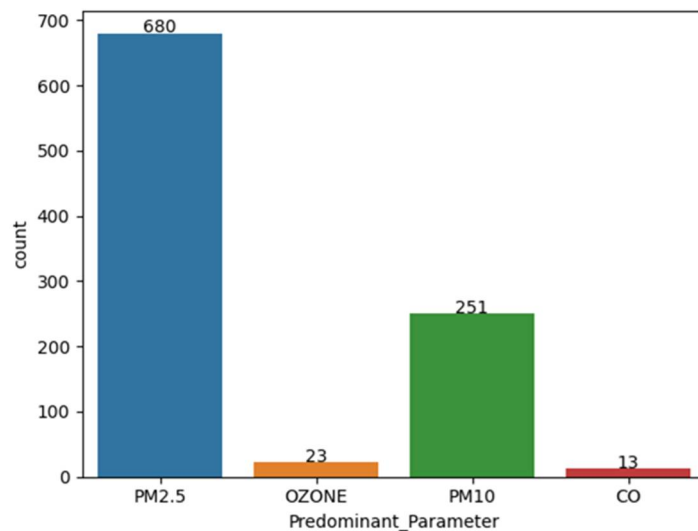
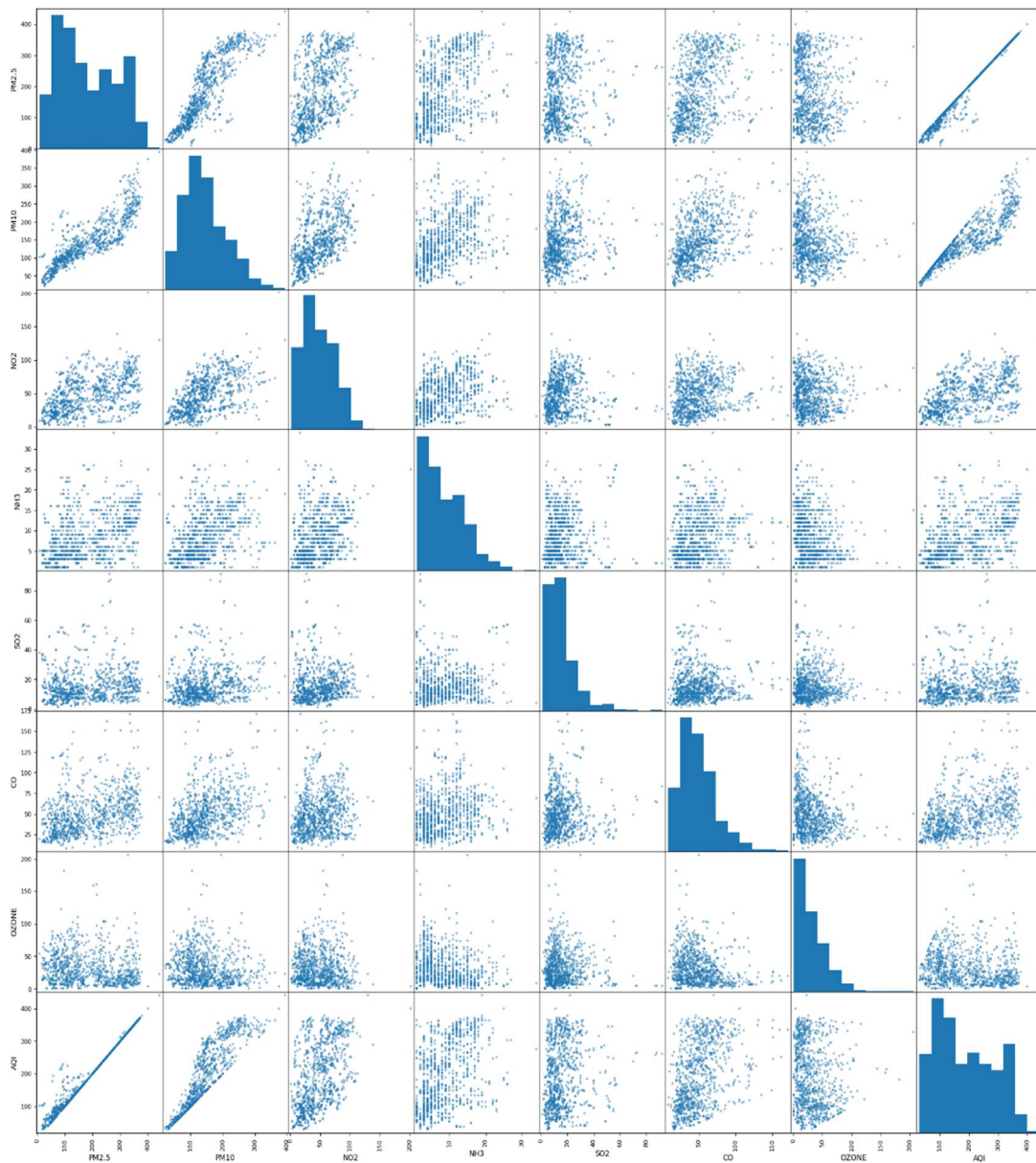


Fig 4.1.5 Pie Chart of the dominance of the predominant characters along with the code

- Scatter Plot along a 2-D plane shows how the data are scattered across the axes.
- A scatter plot across the total columns are shown next.



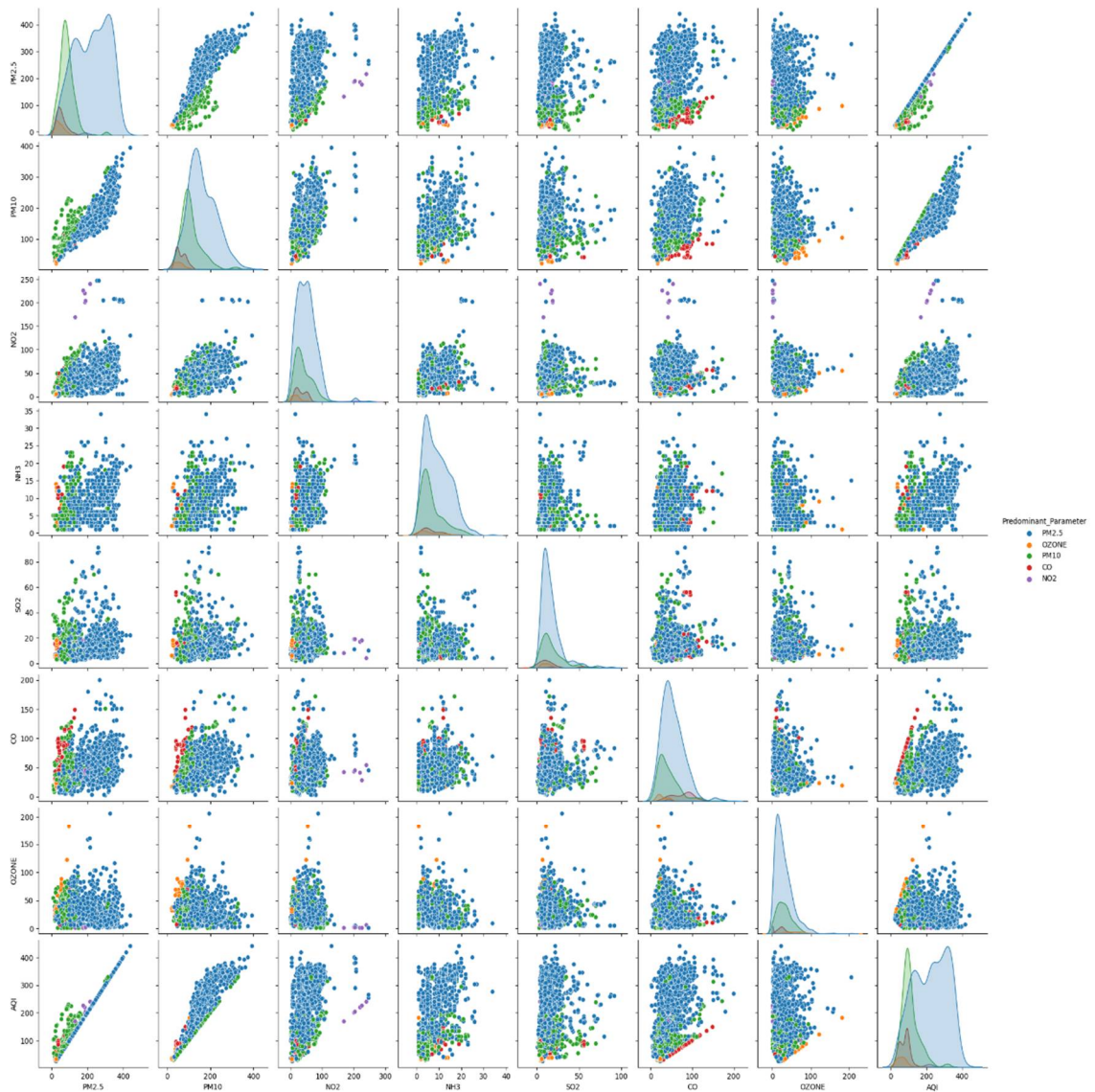
```

1 plt.figure(figsize=(20,20))
2 from pandas.plotting import scatter_matrix
3 p=scatter_matrix(df_cleaned,figsize=(25, 25))

```

Fig 4.1.6 Scatter Plot along with the code

- Scatter Plot of the Predominant Column shows how much the data are scattered across the columns



```
1 p=sns.pairplot(df, hue = 'Predominant_Parameter')
```

Fig 4.1.7 Scatter Plot of Predominant Parameters

- **4.1.4 : Train and Test Split**

- **What is train and test splitting?**

Train and test splitting is a technique used in machine learning to evaluate the performance of a model. It involves splitting the data into two sets: the training set and the test set. The training set is used to train the model, and the test set is used to evaluate the model's performance on unseen data.

- **Why is train and test splitting important?**

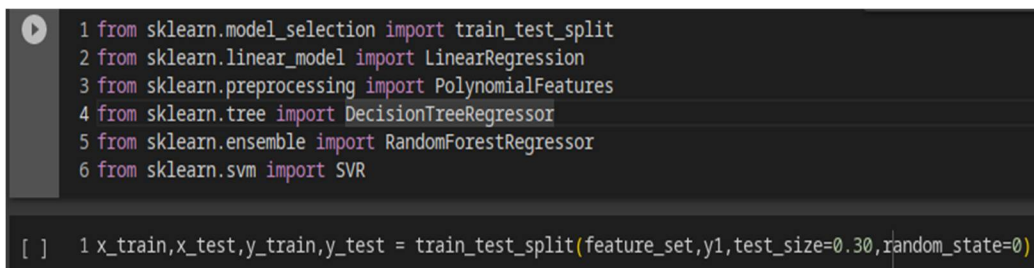
Train and test splitting is important because it allows us to assess how well the model will perform on new data. If we only train the model on the training set, we may be overfitting the model to the training data. This means that the model will perform well on the training data, but it will not generalize well to new data. By using a test set, we can evaluate the model's performance on unseen data and ensure that it is not overfitting.

- **How to do train and test splitting?**

There are a few different ways to do train and test splitting. One common approach is to use a 70/30 split. This means that 70% of the data is used for training and 30% of the data is used for testing. Another approach is to use a stratified split. This means that the data is split so that the proportions of the different classes in the training set are the same as the proportions of the different classes in the original dataset.

- **What is the 70/30 split?**

The 70/30 split is a common split ratio used for train and test splitting. It means that 70% of the data is used for training and 30% of the data is used for testing. This split ratio is a good compromise between having enough data to train the model and having enough data to evaluate the model's performance on unseen data.



```
1 from sklearn.model_selection import train_test_split
2 from sklearn.linear_model import LinearRegression
3 from sklearn.preprocessing import PolynomialFeatures
4 from sklearn.tree import DecisionTreeRegressor
5 from sklearn.ensemble import RandomForestRegressor
6 from sklearn.svm import SVR

[ ] 1 x_train,x_test,y_train,y_test = train_test_split(feature_set,y1,test_size=0.30,random_state=0)
```

Fig 4.1.8 Splitting Data into Train and Test

- **4.1.5 : LazyPredictor to run all regression models**

LazyPredict is a Python library that automates the process of model selection and evaluation in machine learning. It provides a simple and efficient way to build and evaluate a variety of machine learning models, including decision trees, random forests, support vector machines, and neural networks. LazyPredict also supports hyperparameter tuning, which allows you to optimize the performance of your models.

One of the key features of LazyPredict is its ability to compare the performance of different models on a single dataset. This makes it easy to identify the best model for your specific problem. LazyPredict also provides a number of other features that make it a valuable tool for machine learning practitioners, including:

Automatic data preprocessing: LazyPredict can automatically preprocess your data, which can save you a significant amount of time.

Model selection: LazyPredict can automatically select the best model for your problem, based on a variety of criteria.

Hyperparameter tuning: LazyPredict can automatically tune the hyperparameters of your models, to improve their performance.

Evaluation metrics: LazyPredict provides a variety of evaluation metrics, so you can assess the performance of your models.

Model explainability: LazyPredict can provide explanations for the predictions made by your models, which can help you to understand how the models work.

```
# use lazypredict to find the best model for this dataset
# import LazyRegressor

from lazypredict.Supervised import LazyRegressor

# create the model
reg = LazyRegressor(verbose=0, ignore_warnings=False, custom_metric=None)

# fit the model
models, predictions = reg.fit(train.drop(columns=['AQI']), test.drop(columns=['AQI']), train['AQI'], test['AQI'])

# print the models
print(models)
```

Fig 4.1.9 LazyPredict codes

Model	Adjusted R-Squared	R-Squared	RMSE
ExtraTreeRegressor	1.00	1.00	2.48
XGBRegressor	1.00	1.00	4.44
GradientBoostingRegressor	1.00	1.00	5.51
ExtraTreesRegressor	1.00	1.00	5.67
RandomForestRegressor	1.00	1.00	7.94
LGBMRegressor	1.00	1.00	8.20
DecisionTreeRegressor	1.00	1.00	8.66
HistGradientBoostingRegressor	1.00	1.00	8.66
BaggingRegressor	0.99	0.99	8.87
MLPRegressor	0.99	0.99	9.37
KNeighborsRegressor	0.99	0.99	9.49
Ridge	0.96	0.96	24.33
SGDRegressor	0.96	0.96	24.33
BayesianRidge	0.96	0.96	24.34
RidgeCV	0.96	0.96	24.34
Lars	0.96	0.96	24.34
LarsCV	0.96	0.96	24.34
LassoLarsCV	0.96	0.96	24.34
LassoLarsIC	0.96	0.96	24.34
TransformedTargetRegressor	0.96	0.96	24.34
LinearRegression	0.96	0.96	24.34
LassoCV	0.96	0.96	24.35
Lasso	0.96	0.96	24.71
LassoLars	0.96	0.96	24.71
OrthogonalMatchingPursuitCV	0.96	0.96	24.90
RANSACRegressor	0.96	0.96	24.94
ElasticNetCV	0.96	0.96	25.75
HuberRegressor	0.96	0.96	26.01
PoissonRegressor	0.95	0.95	27.54
LinearSVR	0.95	0.95	27.64
PassiveAggressiveRegressor	0.93	0.93	32.48
OrthogonalMatchingPursuit	0.93	0.93	33.17
ElasticNet	0.93	0.93	33.64
SVR	0.92	0.92	34.54
NuSVR	0.92	0.92	34.93
AdaBoostRegressor	0.91	0.91	37.04
TweedieRegressor	0.88	0.88	42.90
GammaRegressor	0.81	0.81	54.41
DummyRegressor	-0.01	-0.00	123.83
GaussianProcessRegressor	-0.19	-0.18	134.64
KernelRidge	-2.87	-2.85	242.73

Model	Time Taken
ExtraTreeRegressor	0.04
XGBRegressor	0.52
GradientBoostingRegressor	1.87
ExtraTreesRegressor	2.14
RandomForestRegressor	7.38
LGBMRegressor	0.21
DecisionTreeRegressor	0.20
HistGradientBoostingRegressor	0.79
BaggingRegressor	0.98
MLPRegressor	6.78
KNeighborsRegressor	0.10
Ridge	0.02
SGDRegressor	0.09
BayesianRidge	0.06
RidgeCV	0.05
Lars	0.03
LarsCV	0.06
LassoLarsCV	0.05
LassoLarsIC	0.02
TransformedTargetRegressor	0.02
LinearRegression	0.02
LassoCV	0.18
Lasso	0.02
LassoLars	0.01
OrthogonalMatchingPursuitCV	0.06
RANSACRegressor	0.04
ElasticNetCV	0.18
HuberRegressor	0.13
PoissonRegressor	0.04
LinearSVR	0.03
PassiveAggressiveRegressor	0.02
OrthogonalMatchingPursuit	0.02
ElasticNet	0.06
SVR	3.74
NuSVR	2.45
AdaBoostRegressor	1.05
TweedieRegressor	0.03
GammaRegressor	0.03
DummyRegressor	0.04
GaussianProcessRegressor	10.36
KernelRidge	4.32

Fig 4.1.10 Results of LazyPredict

◦ **4.1.6 : Take user input and display predictions**

- User input is required to perform the prediction of AQI over a certain place.
- The code and the result is shown below

```
# predict AQI for Kolkata using the model based on user input

# create a function to take user input
def take_input():
    print("Enter the following details:")
    print("so2: ", end="")
    so2 = float(input())
    print(so2, "\n")
    print("no2: ", end="")
    no2 = float(input())
    print(no2, "\n")
    print("rspm: ", end="")
    rspm = float(input())
    print(rspm, "\n")
    print("spm: ", end="")
    spm = float(input())
    print(spm, "\n")
    print("pm2_5: ", end="")
    pm2_5 = float(input())
    print(pm2_5, "\n")
    print("si: ", end="")
    si = float(input())
    print(si, "\n")
    print("ni: ", end="")
    ni = float(input())
    print(ni, "\n")
    print("rpi: ", end="")
    rpi = float(input())
    print(rpi, "\n")
    print("spi: ", end="")
    spi = float(input())
    print(spi, "\n")
    print("AQI: ", end="")
    aqi = float(input())
    print(aqi, "\n")
    return so2, no2, rspm, spm, pm2_5, si, ni, rpi, spi, aqi

# take user input
so2, no2, rspm, spm, pm2_5, si, ni, rpi, spi, aqi = take_input()

# create a dataframe with the user input
user_input = pd.DataFrame({'so2': [so2], 'no2': [no2], 'rspm': [rspm], 'spm': [spm], 'pm2_5': [pm2_5], 'si': [si], 'ni': [ni], 'rpi': [rpi], 'spi': [spi], 'aqi': [aqi]})

# predict the aqi for the user input
pred = model.predict(user_input.drop(columns=['AQI']))

# print the predicted aqi
print("The predicted AQI is: ", pred[0])
```

```
Enter the following details:
so2: 22.2
no2: 104.8
rspm: 476.0
spm: 255.0
pm2_5: 32.0
si: 27.75
ni: 124.8
rpi: 573.85
spi: 205.0
AQI: 573.85
The predicted AQI is: 596.5795752415531
```

Fig 4.1.11 Taking User Inputs from CLI and display the results

- **4.1.7 : User Interface over the web**

- Using Gradio to have a user interface for users to interact
- Gradio is a free open-source python library that allows users to create interactive web demos for ML models
- Gradio supports a variety of input and output components, including images, text, audio, and video.
- Gradio is easy to use, even for beginners. You can create a Gradio demo in just a few lines of code.
- Gradio demos are portable and can be shared with others through a URL.
- Gradio is actively maintained and developed. There are a number of community-contributed components and features available.

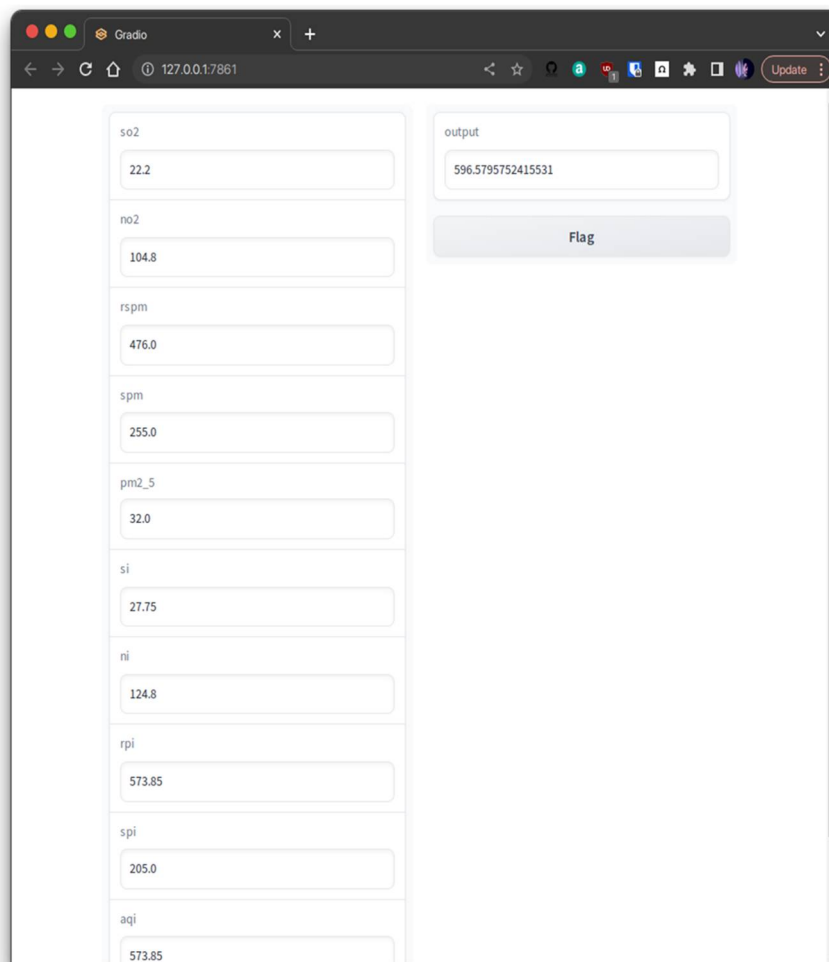


Fig 4.1.12 Gradio Interface

4.2 : Prototypes of Project

Creating prototypes for the **Air Quality Index for Urban Areas** project involves creating mock-ups or interactive representation of the web dashboard and other components.

Here is an example of how the main sFig x.xx Map View design Prototype sections of the web dashboard might be represented.

- **4.2.1 : Home Page**

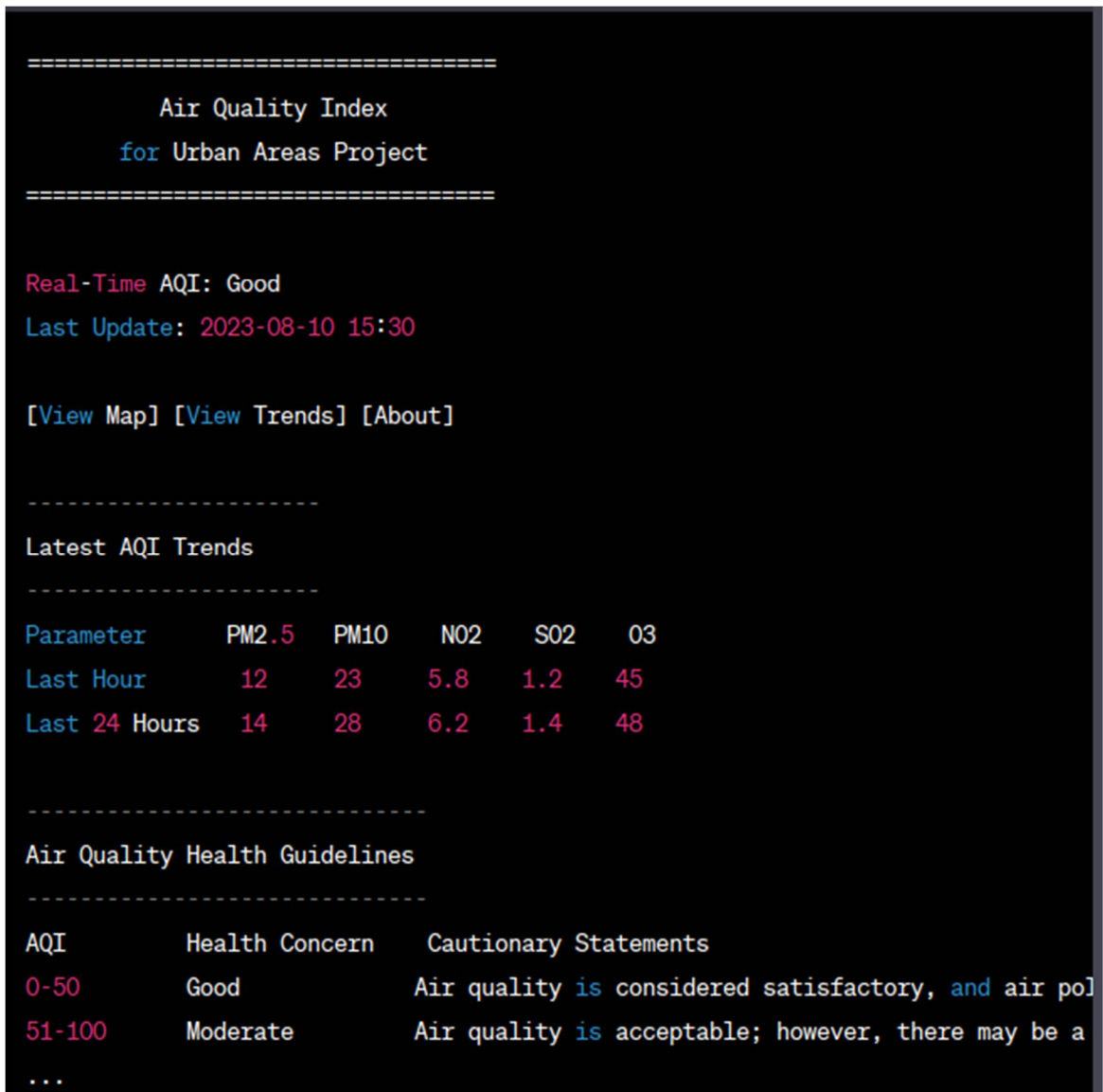


Fig 4.1.13 Home Page design Prototype

- **4.2.2 : Map View prototype**

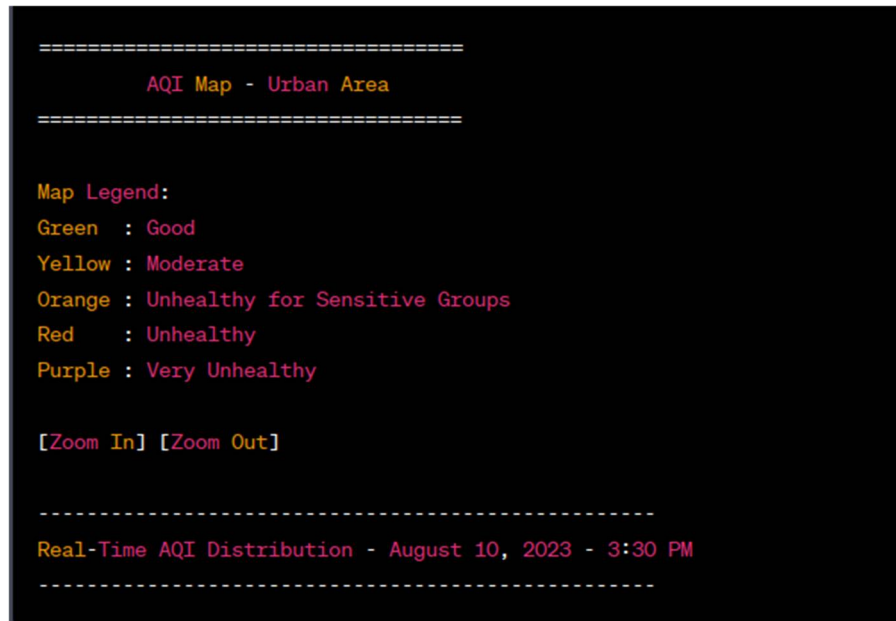


Fig 4.1.14 Map View design Prototype

Map View Prototype is the prototype model of the webpage for displaying the Map view of an area. This contains a color group which signifies the AQI of a certain area.

Green signifies Good AQI

Yellow signifies Moderate AQI

Orange to Red signifies unhealthy at intense levels

◦ 4.2.3 : *AQI Trends*

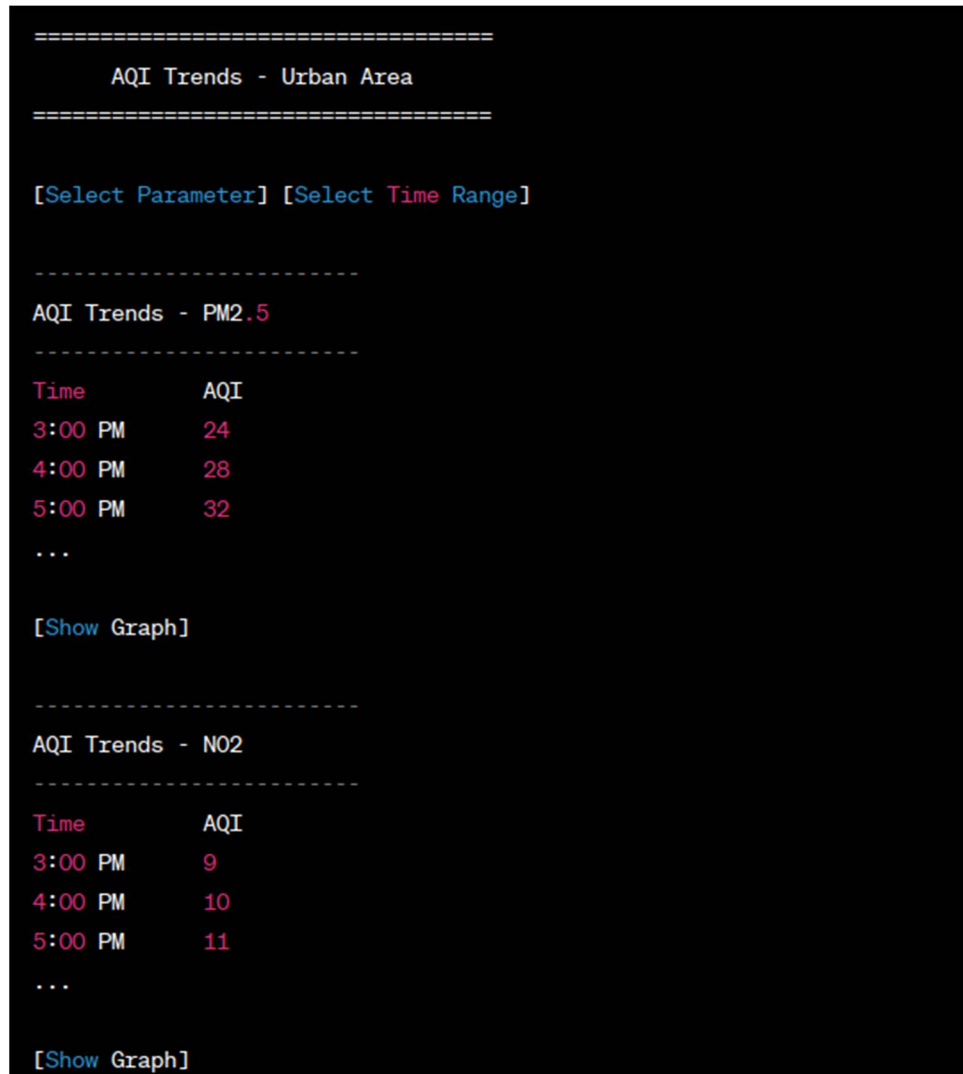


Fig 4.1.15 Hourly Trends in Air Quality Index

The hourly trend shows the AQI change in hourly periods. This is very important to work with since it is used by many users and public to plan and execute, related to health.

Chapter 5

RESULT AND ANALYSIS

The Result and Analysis of the **Air Quality Index for Urban Areas** project are essential for understanding the project's effectiveness, validating its predictions and deriving actionable insights.

The approaches to presenting the results and conducting analysis are -

5.1 Result Presentation:

- *An AQI prediction dashboard*
 - Showcase the developed dashboard where user can actually look into the predicted values
 - Highlight the user-friendly interface
- *Predictive Model Performance*
 - Present performance metrics, like RMSE, R-Squared for each air quality parameter's predictive models.
 - Display the metrics along with the data
- *Spatial Analysis*
 - Examine spatial trends like AQI near industrial areas, pollution hotspots etc.

5.2 Data Analysis and Interpretation:

- *Correlation Analysis:*
 - Analyze the correlation between air quality parameters and meteorological factors
 - Identify which meteorological factors have the most influence on air quality levels.

- ***Temporal Patterns:***
 - manually investigate daily, weekly and seasonal patterns in air quality fluctuations
 - Identify peak pollution hours or days and potential contributing factors
- ***Impact on Health and Environment:***
 - Discuss and point out the health risks associated with different Aqi ranges
 - Highlight how the project's real-time information empowers residents to protect their health.

5.3 Interpretation and Insights:

- ***Public health implications***
 - Interpret the AQI values manually in terms of health risks for different populations, e.g. sensitive groups
 - Discuss how the project contributes to reducing health risks by providing timely information
- ***Urban Planning Insights***
 - Derive insights from urban planning decisions based on the relationship between air quality and influencing factors
 - Discuss how the data can guide policies for zoning, transportation and industrial regulations.
- ***Comparative Analysis***
 - Compare model predictions with actual air quality data to assess the accuracy of predictions.
 - Identify potential areas for model improvements or refinement.

5.4 Future Recommendations:

- Suggest avenues for further researches and improvements in predictive models.
- Recommend expanding the sensor network to enhance spatial coverage and accuracy.
- Propose ways to integrate additional influencing variables for more accurate predictions.

Overall, incorporating visual aids such as graphs, charts and maps to make the analysis more accessible and impactful. The goal is to provide a comprehensive understanding of the project's outcomes. Demonstrate the values of the real-time AQI dashboard and offer insights that can guide future actions for air quality management and urban planning.

Chapter 6

CONCLUSION AND FUTURE SCOPE

The **Air Quality Index for Urban Areas** project has successfully addressed the critical concern of air quality in densely populated cities by providing accurate real-time air quality information through an intuitive web dashboard. This project aimed to enhance public awareness, enable informed decision-making and contribute to healthier environments. Through the integration of historical and real-time data, predictive modelling and user-friendly visualization, the project has achieved its objective and delivered tangible benefits to both residents and urban planners.

The developed web dashboard has empowered to make informed choices about outdoor activities based on real-time Air Quality Index values. Additionally, the project's insights have enabled urban planners and policymakers to implement data driven pollution control measures and make informed decisions regarding urban development, transportation and industrial regulations.

Future Scope:

While the project has achieved significant milestones, there are several avenues for future expansions and improvement:

1. Sensor Network Expansions

- Increase the coverage of environmental sensors across the urban area to capture a more comprehensive spatial representation of air quality.

2. Inclusion of more influencing factors

- Incorporate additional influencing variables such as traffic density, land use, and emissions from various sources to enhance prediction accuracy.

3. Machine Learning Advancements

- Explore advanced machine learning techniques, such as deep learning, to further improve the accuracy of predictive models.

4. User Engagement Features

- Introduce personalized alerts and notifications for residents, informing them of critical air quality changes.

5. Integration with Health Services

- Collaborate with healthcare providers to integrate the AQI data into health advisories and services for vulnerable groups.

6. Long-Term Trend Analysis

- Extend the analysis to identify long-term trends in air quality and assess the effectiveness of pollution control measures over time

7. Collaboration with Industries

- Collaborate with industries and factories to gather real-time emission data, enabling more accurate prediction modeling.

8. Mobile Application development

- Develop a mobile application for easier access to real-time AQI information, making it more convenient for residents.

9. Research on Health Impacts

- Collaborate with health experts and researchers to conduct studies on the health impact of improved air quality due to the project's interventions.

10. Expansion to other cities

- Replicate the project in other cities, customizing it based on local conditions and needs.

In conclusion, the "Air Quality Index for Urban Areas" project has laid the foundation for a healthier and more informed urban population. By continuously improving and expanding the project's scope, we can contribute to sustained environmental well-being, informed public decisions, and data-driven urban development for a better future.

Chapter 7

REFERENCES

- [1] Simple Linear Regression – “Machine Learning”, Oxford UP, 2021, ISBN-13: 978-0-19-012727-5
- [2] Exploratory Data Analysis – Kaggle, 15-08-2023
- [3] training and splitting test/train data – “Statistical Analysis and Data Mining”, 2022, Volume 15, Issue 4
- [4] LazyPredict – Pypi, 15-08-2023
- [5] Gradio – Gradio, 15-08-2023
- [6] git – git-scm, 15-08-2023
- [7] data visualization – youTube, 15-08-2023