**AIR QUALITY ANALYSIS**

**OBJECTIVE**

To transform the design and ideas from the Problem Definition and Design Thinking phase into practical solutions for analyzing Air Quality Analysis using Jupyter notebooks and JupyterBook.

**Step 1: Data Preprocessing**

* **Data Transformation:** The conversion of raw data into a structured format that is suitable for analysis and decision-making. This includes cleaning, formatting, and standardizing data.
* **Data Analysis:** Utilizing statistical, computational, or analytical methods to derive insights, trends, and patterns from the processed data. This step often involves data mining, machine learning, and visualization.
* **Information Extraction:** Extracting relevant information and knowledge from the data to support informed decision-making, research, and problem-solving.
* **Decision Support:** Providing valuable insights to aid in making informed decisions, whether in business, science, or other domains

**Step 2: Jupyter Environment Setup**

* **Environment Setup**: Ensure that your Jupyter environment is properly configured and ready for data analysis.
* **Library Installation**: Install the necessary Python libraries like Pandas, Matplotlib, Seaborn, and NumPy for data analysis and visualization.

**Step 3: Data Analysis**

* **Data Preparation:** This involves cleaning and transforming raw data into a suitable format for analysis. Data may need to be structured, filtered, and organized.
* **Exploratory Data Analysis (EDA):** EDA involves visualizing data and generating summary statistics to understand its key characteristics, trends, and patterns.
* **Statistical Analysis:** Using statistical techniques to quantify and validate findings. This can involve hypothesis testing, regression analysis, and more.
* **Interpretation**: Drawing meaningful insights and conclusions from the analysis results and communicating these findings to stakeholders.

**Data Visualization**

* **Charts** **and** **Graphs**: Creating visual representations such as bar charts, line graphs, scatter plots, and heatmaps to display data in an easily digestible manner.
* **Dashboards:** Building interactive dashboards that allow users to explore data and interact with visualizations to answer specific questions or gain insights.
* **Infographics**: Designing concise and visually appealing graphics to convey key data-driven messages or stories.

**Step 4: Insights Generation**

* **Data Analysis:** Insight generation typically begins with data analysis. This involves techniques such as statistical analysis, data mining, machine learning, and qualitative analysis, depending on the nature of the data.
* **Pattern Recognition:** The identification of recurring patterns or anomalies within the data is a fundamental aspect of insight generation. These patterns may not be immediately apparent and often require in-depth analysis.
* **Data Visualization:** Data visualization is often used to represent insights in a more understandable and compelling manner. Visualizations like charts, graphs, and maps can make complex findings more accessible

**Step 5: JupyterBook Creation**

* **Installation**: Install JupyterBook, a tool to create interactive, shareable, and documentation-ready books from Jupyter notebooks.
* **Book Structure**: Define the structure of your JupyterBook, including chapters and sections.
* **Notebook Integration**: Include your Jupyter notebooks, code, visualizations, and insights into the JupyterBook

**JUPYTERBOOK**

**VISUALYISING TOP 10 STOPS BY BOARDING POINTS USING JUPYTERNOTE BOOK**

import pandas as pd

import matplotlib.pyplot as plt

data = pd.read\_excel('Air Quality Analysis dataset.xlsx')

data.replace('NA', float('nan'), inplace=True)

pollutants = ['SO2', 'NO2', 'RSPM/PM10', 'PM 2.5']

location\_types = data['Type of Location'].unique()

average\_pollutant\_levels = []

for location\_type in location\_types:

avg\_levels = [data[data['Type of Location'] == location\_type][pollutant].mean() for pollutant in pollutants]

average\_pollutant\_levels.append(avg\_levels)

plt.figure(figsize=(12, 6))

width = 0.2

x = range(len(location\_types))

for i, pollutant in enumerate(pollutants):

plt.bar([pos + width \* i for pos in x], [level[i] for level in average\_pollutant\_levels], width=width, label=pollutant)

plt.xlabel('Type of Location')

plt.ylabel('Average Concentration')

plt.title('Average Pollutant Levels by Type of Location')

plt.xticks([pos + width for pos in x], location\_types)

plt.legend()

plt.grid(axis='y')

plt.show()

plt.figure(figsize=(12, 8))

for i, pollutant in enumerate(pollutants):

plt.subplot(2, 2, i+1)

data[pollutant].plot(kind='hist', bins=20, edgecolor='k', alpha=0.7)

plt.xlabel(pollutant + ' Concentration')

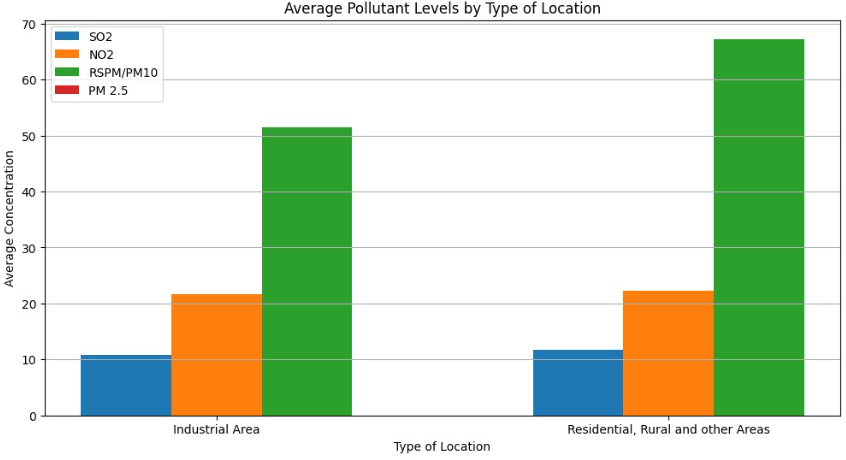
plt.ylabel('Frequency')

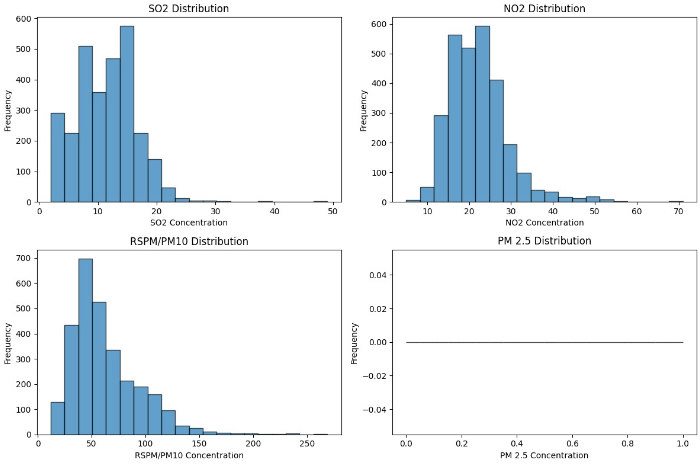
plt.title(f'{pollutant} Distribution')

plt.tight\_layout()

plt.show()

**OUTPUT**

****

****