**Assignment: Python Programming for GUI**

**Development**

Name: Tharunika Reddy .CH

Register Number: 192324307

Department: AI & DS

Date of Submission: 26.08.2024

**Problem: Air Pollution API**

1. Current, forecast and historical air pollution data
2. Forecast for 4 days ahead with 1-hour step
3. Air Pollution API includes both Air Quality Index and indices for CO, NO, NO2, O3, SO2, NH3,PM2.5, PM10 .

**Deliverables:**

1. Data flow diagram illustrating the interaction between the application and the API .
2. Pseudo code and implementation
3. Documentation of the API integration and the methods used to fetch and display the data.
4. Explanation of any assumptions made and potential improvements

**Solution:**

## Air Pollution API

**1. Data Flow Diagram**

Start

**Define API Key and City**

**Construct API Request URL**

**Make API Request**

**Check Response Status**

**Analyze Response Status**

**Failure**

**success**

**Log Error/Display Error Message**

**Parse JSON Data**

**Extract AQI & Pollutant Data**

**Display Data to User**

**end**

**2. Implementation**

import requests

def get\_air\_pollution\_data(api\_key,lat,lon):

url =

f"http://api.openweathermap.org/data/2.5/air\_pollution?lat={lat}&lon={lon}&appid

{api \_ key}

response=requests.get(url)

if response.status\_code==200:

data=response.json()

aqi=data['list'][0]['main']['aqi']

components=data['list'][0]['components']

print("Air Quality Index (AQI):", aqi)

print("Concentration of pollutants in µg/m³:")

print("CO:", components['co'])

print("NO:", components['no'])

print("NO2:",components['no2'])

print("O3:",components['o3'])

print("SO2:",components['so2'])

print("PM2.5:",components['pm2\_5'])

print("PM10:",components['pm10'])

print("NH3:",components['nh3'])

else:

print("Failed to retrieve data. HTTP Status code:", response .status \_code)

if \_name\_ == "\_main\_":

api \_ key = "7388dca5f7653c38ee1f85baa3f7a900"

lat = "37.7749"

lon = "-122.4194"

get \_ air \_ pollution \_data( api \_key, lat, lon)

**3. Display the air pollution information**

Air Quality Index (AQI): 1

Concentration of pollutants in µg/m³:

CO: 220.3

NO: 0.5

NO2: 3.26

O3: 49.35

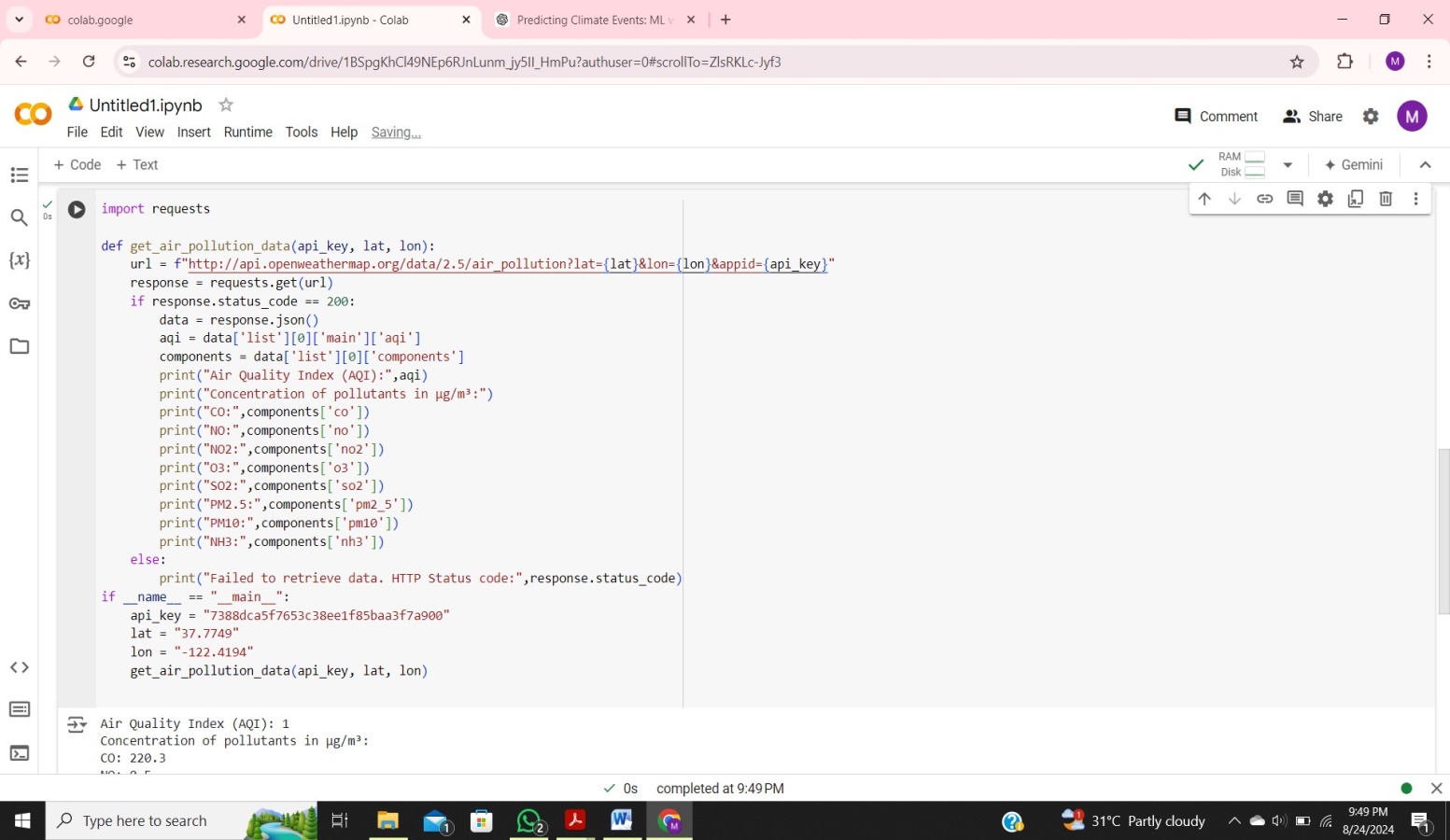
SO2: 0.54

PM2.5: 1.79

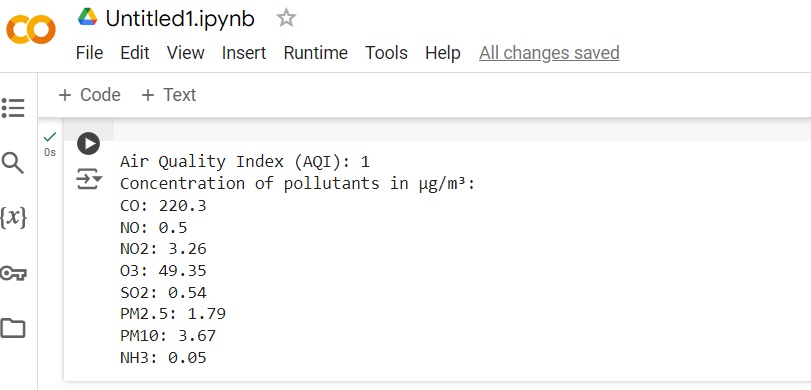
PM10: 3.67

NH3: 0.05

**4. User Input**



**5. user output:**

****

**DOCUMENTATION:**

CODE EXPLANATION:

1. **Imports:**
   * The **requests** library is imported to facilitate HTTP requests to external APIs.
2. **Function Definition:**
   * **Function Name:** **get\_air\_pollution\_data(api\_key, lat, lon)**
   * **Purpose:** The function fetches air pollution data from the OpenWeatherMap API using the provided latitude (**lat**) and longitude (**lon**) coordinates.
3. **API Request:**
   * The function builds a URL using the API key and geographical coordinates.
   * A GET request is then sent to the OpenWeatherMap API endpoint using the constructed URL.
4. **Response Handling:**
   * The function checks if the HTTP response status code is 200 (indicating a successful request).
   * If the request is successful, the function decodes the JSON response into a usable format.
5. **Data Extraction:**
   * The function extracts the following from the JSON response:
     + **Air Quality Index (AQI)**
     + **Pollutant concentrations** (e.g., PM2.5, PM10) in micrograms per cubic meter (µg/m³).
6. **Output:**
   * The function prints the Air Quality Index (AQI) and the concentrations of the extracted pollutants.
7. **Error Handling:**
   * If the API request fails (i.e., the status code is not 200), the function prints an error message to inform the user of the issue.
8. **Usage Example:**
   * Replace **"YOUR\_API\_KEY"** in the function call with your actual OpenWeatherMap API key.
   * Provide the appropriate **lat** and **lon** values to retrieve and display the air quality data for a specific location

ASSUMPTIONS MADE:

1. **API Key Validity:** The API key is valid, active, and has the necessary permissions.
2. **Endpoint Availability:** The OpenWeatherMap API endpoint is operational and accessible.
3. **Response Format:** The API response follows the expected structure with the correct keys (**'list'**, **'main'**, **'components'**).
4. **Geographical Coordinates:** Latitude and longitude inputs are valid and correctly formatted.
5. **Data Completeness:** The API response includes all necessary data fields, such as AQI and pollutant concentrations.
6. **Rate Limits:** API requests adhere to OpenWeatherMap's rate limits and usage policies.
7. **Localization:** Data is returned in a standardized format, such as µg/m³ for pollutant concentrations.
8. **Error Handling:** The code manages errors for unexpected API responses, such as outages or incorrect inputs.

**LIMITATIONS:**

**Data Accuracy**: Variable accuracy and coverage, especially in less monitored areas.

**API Limits**: Rate and quota limits restrict the number of requests and data volume.

**Granularity**: Limited parameters and data resolution may affect detail and precision.

**Cost**: Higher-quality or more frequent data often requires a paid subscription.

**Reliability:** Dependence on third-party service availability and potential API errors.