# **Assignment: Python Programming for GUI Development**

Name : G. Mohammad Shareef

Registration Number : 192311288

Department : CSE

Date Of Submission : 26-08-2024

**Problem 1: Real-Time Weather Monitoring System**

**Scenario:**

You are developing a real-time weather monitoring system for a weather forecasting  company. The system needs to fetch and display weather data for a specified location.

**Tasks:**

1. Model the data flow for fetching weather information from an external API and  displaying it to the user.

2. Implement a Python application that integrates with a weather API (e.g.,  Open Weather Map) to fetch real-time weather data.

3. Display the current weather information, including temperature, weather  conditions, humidity, and wind speed.

4. Allow users to input the location (city name or coordinates) and display the  corresponding weather data.

**Deliverables:**

• Data flow diagram illustrating the interaction between the application and the API.

• Pseudocode and implementation of the weather monitoring system.

• Documentation of the API integration and the methods used to fetch and display  weather data.

• Explanation of any assumptions made and potential improvements.

**Solution:**

**Real-Time Weather Monitoring System**

**1.Data Flow Diagram:**

Display Weather Data

Weather API

Weather Monitoring System

User Input

**2.Implementation:**

**Pseudocode:**

BEGIN

DISPLAY "Enter location (city name or coordinates):"

INPUT location

API\_KEY = "your\_api\_key"

BASE\_URL = "http://api.openweathermap.org/data/2.5/weather"

REQUEST\_URL = BASE\_URL + "?q=" + location + "&appid=" + API\_KEY

RESPONSE = SEND\_REQUEST(REQUEST\_URL)

IF RESPONSE IS SUCCESSFUL THEN

WEATHER\_DATA = PARSE\_JSON(RESPONSE)

TEMPERATURE = WEATHER\_DATA["main"]["temp"]

WEATHER\_CONDITIONS = WEATHER\_DATA["weather"][0]["description"]

HUMIDITY = WEATHER\_DATA["main"]["humidity"]

WIND\_SPEED = WEATHER\_DATA["wind"]["speed"]

DISPLAY "Temperature: " + TEMPERATURE

DISPLAY "Weather Conditions: " + WEATHER\_CONDITIONS

DISPLAY "Humidity: " + HUMIDITY

DISPLAY "Wind Speed: " + WIND\_SPEED

ELSE

DISPLAY "Error fetching weather data."

ENDIF

END

**Code:**

import requests

# Define the API endpoint and parameters

api\_key = ' 27dd6a2031834c2094b80536242508'  # Replace with your actual API key

query = 'vijayawada'

url = f'http://api.weatherapi.com/v1/current.json?key={api\_key}&q={query}&aqi=yes'

def fetch\_current\_weather(api\_key, query):

    url = f'http://api.weatherapi.com/v1/current.json?key={api\_key}&q={query}&aqi=yes'

    response = requests.get(url)

    if response.status\_code == 200:

        data = response.json()

        # Access and print relevant data

        location = data['location']

        current = data['current']

        print(f"Location: {location['name']}, {location['region']}, {location['country']}")

        print(f"Temperature: {current['temp\_c']}°C")

        print(f"Condition: {current['condition']['text']}")

        print(f"Humidity: {current['humidity']}%")

        print(f"Air Quality (US EPA Index): {current['air\_quality']['us-epa-index']}")

        print(f"Wind Speed: {current['wind\_kph']} kph")

        print(f"Last Updated: {current['last\_updated']}")

    else:

        print(f"Failed to retrieve data: {response.status\_code}")

# Fetch and print the current weather data for Vijayawada

fetch\_current\_weather(api\_key, query)

**3.Display the Current weather information:**

Location: Kadapa, Andhra Pradesh, India

Temperature: 32.2°C

Condition: Partly cloudy

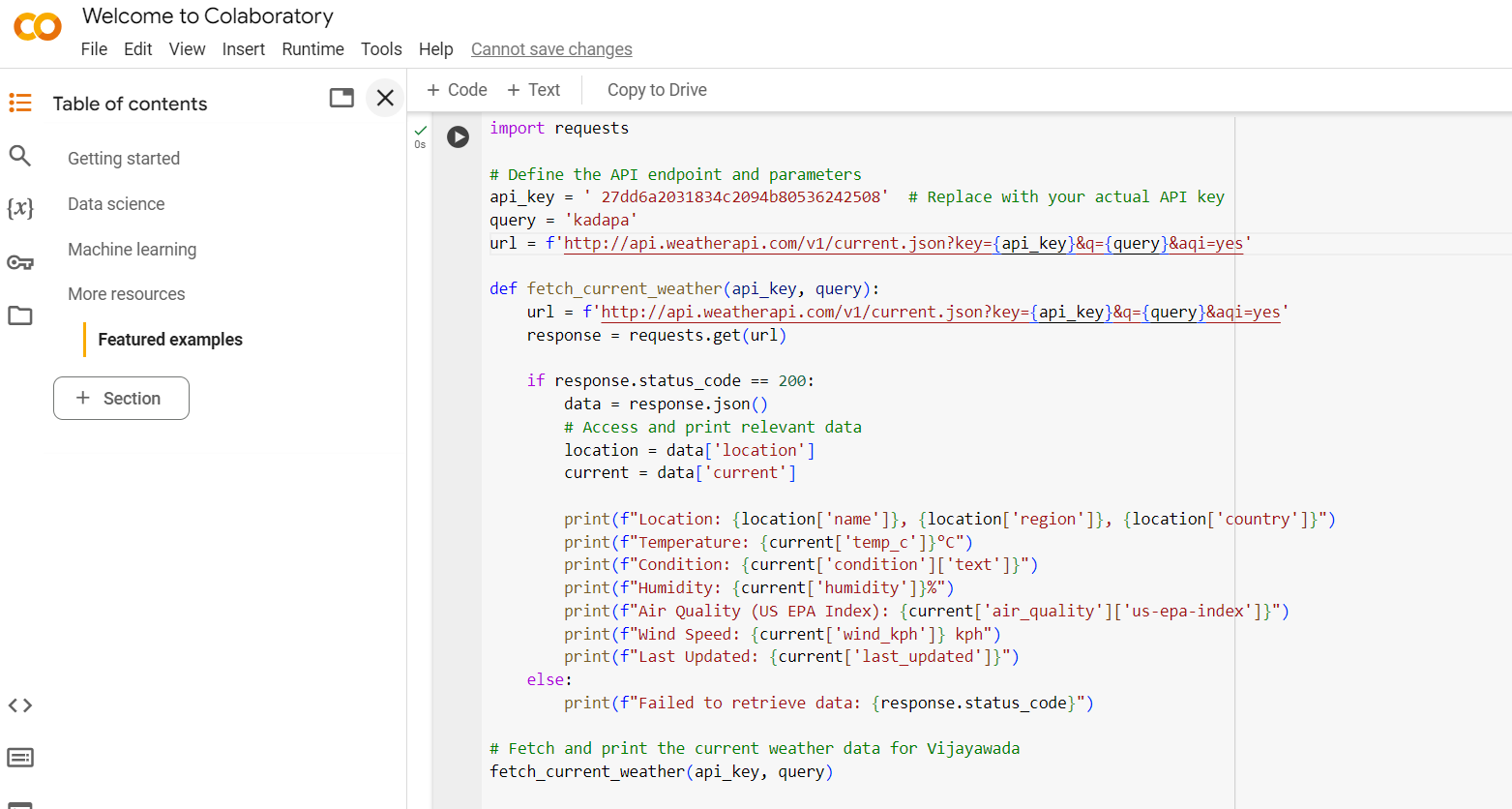
Humidity: 46%

Air Quality (US EPA Index): 1

Wind Speed: 28.1 kph

Last Updated: 2024-08-26 10:15

**4.User Input:**

****

A screenshot of a computer

Description automatically generated

**5.Documentation :**

**Documentation of API Integration**

1. **API Endpoint**: The weather data is fetched from **http://api.openweathermap.org/data/2.5/weather**.
2. **Parameters**:
   * **q**: Location (city name or coordinates).
   * **appid**: Your API key from OpenWeatherMap.
   * **units**: Metric for temperature in Celsius.
3. **Response**: The API returns a JSON object containing various weather attributes. Important fields include:
   * **main.temp**: Current temperature.
   * **weather[0].description**: Weather conditions.
   * **main.humidity**: Humidity percentage.
   * **wind.speed**: Wind speed.
4. **Error Handling**: The application checks the HTTP response status code. If it’s not 200, it indicates an error in fetching data.

**Assumptions and Potential Improvements**

**Assumptions**:

* User provides valid location input.
* The API key is correct and has not exceeded usage limits.

**Potential Improvements**:

* Add more robust error handling to manage various API errors.
* Support for additional units of measurement (e.g., Fahrenheit).
* Include a graphical user interface (GUI) for a more user-friendly experience.
* Extend support for multiple locations or historical weather data.