**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.Create a model of the data flow that will be used to retrieve and present meteorological data to the user from an external API.   
2. Put in place a Python program that retrieves real-time weather information by integrating with a weather API (such as Open Weather Map).   
3. Present the current meteorological data, encompassing temperature, precipitation, humidity, and wind velocity.   
4. Let users enter the location (coordinates or name of the city) and show the appropriate meteorological** **information.**

**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.**

1. **In response, this is the model data flow:**

**+-------------------------+**

**| User |asssss**

**+-------------------------+**

**|**

**| (1) Input location (city name or coordinates)**

**v**

**+-------------------------+**

**| Weather Monitoring System |**

**+-------------------------+**

**|**

**| (2) Construct API request URL with user input and API key**

**v**

**+-------------------------+**

**| External Weather API |**

**+-------------------------+**

**|**

**| (3) HTTP GET request to API**

**v**

**+-------------------------+**

**| External Weather API |**

**+-------------------------+**

**|**

**| (4) JSON response with weather data**

**v**

**+-------------------------+**

**| Weather Monitoring System |**

**+-------------------------+**

**|**

**| (5) Parse JSON response and extract relevant weather data**

**v**

**+-------------------------+**

**| User |**

**+-------------------------+**

**|**

**| (6) Display weather data to user**

**v**

**Python Code:**

import requests

def fetch\_weather\_data(city\_name):

api\_key = "b11f2fe52244a66eb93ee793f28c2d3b" # Replace with your actual API key

base\_url = "http://api.openweathermap.org/data/2.5/weather?"

request\_url = f"{base\_url}q={city\_name}&appid={api\_key}"

response = requests.get(request\_url)

return response.json()

def show\_weather\_data(data):

if data['cod'] != '404':

main = data['main']

wind = data['wind']

weather\_desc = data['weather'][0]['description']

print(f"Temperature: {main['temp']}K")

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

print (f "Weather Description: {weather\_ desc}")

print (f "Wind Speed: {wind['speed']} m/s")

else:

print ("City Not Found")

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

city\_ name = input ("Enter the city name: ")

weather \_data = fetch \_weather \_ data (city \_name)

show\_ weather\_ data (weather \_data)

**Pseudocode:**

1. Initialize the application

- Import required libraries

- Set up the API key and base URL for the weather API

2. Obtain user input

- Prompt the user to enter a location (city name or coordinates)

3. Retrieve weather data

- Construct the request URL using the user input and API key

- Send an HTTP GET request to the weather API

- Parse the JSON response to extract relevant weather data

4. Display weather information

- Format and present the current weather details: temperature, weather conditions, humidity, and wind speed

5. Handle errors

- Manage any errors that occur during the API request or data parsing