SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
Program Name: M.Tech/MSC/MCA		**Assignment Type: Lab**	**Academic Year:**2025-2026
Course Coordinator Name		Venkataramana Veeramsetty	
Course Code		**Course Title**	AI Assisted Problem Solving Using Python
Year/Sem		**Regulation**	R25
Date and Day of Assignment	Week5 - Tuesday	**Time(s)**	
Duration	2 Hours	**Applicable to Batches**	
AssignmentNumber:15.4(Present assignment number)/**24**(Total number of assignments)			

Q.No.	Question	*Expected Time to complete*

1	**Lab 18 – API Integration: Connecting to External Services with Error Handling**
	Lab Objectives:
	• Learn how to integrate Python programs with external REST APIs.
	• Understand API request/response handling using requests or similar libraries.
	• Implement proper error handling for failed API calls (timeouts, invalid responses, rate limits).
	• Practice extracting and displaying meaningful information from API responses.
	
	Task 1: Connect to a Public API
	• **Instructions:**
	  • Use Python (or Node.js/JavaScript) to connect to a public API (e.g., OpenWeatherMap or JSONPlaceholder).
	  • Send a simple GET request to retrieve data.
	  • Display the response in a readable format (pretty JSON).
	• **Expected Output:**
	  o Program prints API response (e.g., weather data or posts from JSONPlaceholder).
	  o Output should be properly formatted JSON (not raw).
		Week5 - Tuesday

Prompt: write a python function that display weather details of a city using weather api without error handling. Display weather details as JSON output

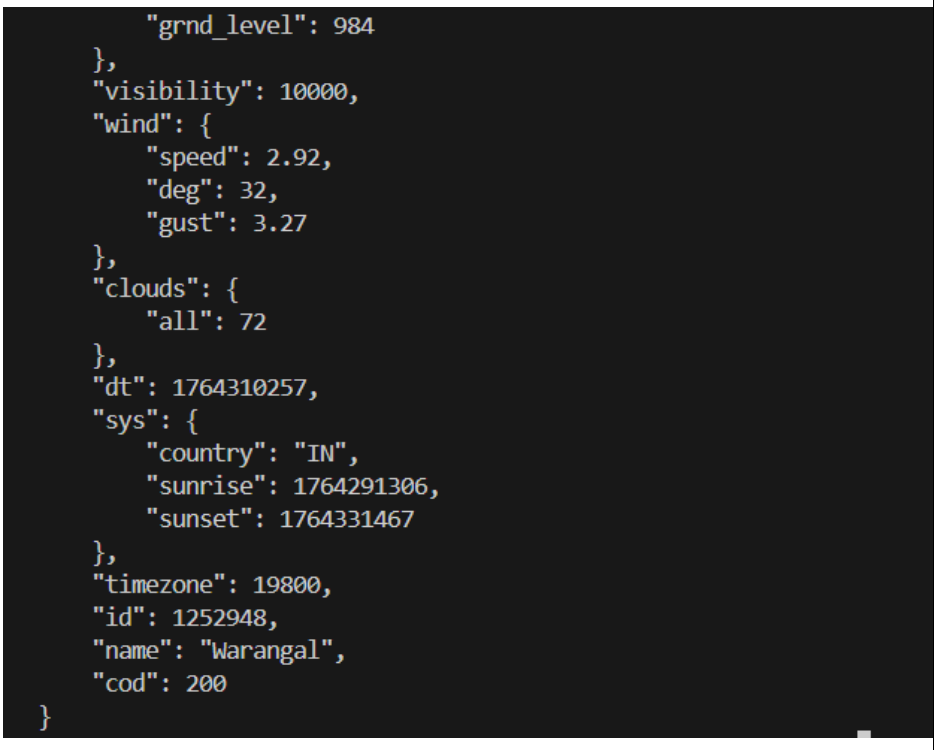
To use this code:

1. **First, you'll need to sign up for a free API key at Open Weather Map: https://openweathermap.org/api**
2. **Replace 'YOUR_API_KEY' in the code with your actual API key**
3. **Install the required requests package by running:**

pip install requests

You can then run the script and enter any city name to get its current weather details. The output will look something like this:

```python
import requests
import json

def get_weather(city):
    api_key = "d0d9c29cd5e0f6aa41e9370ad3874404"  # your working key
    url = f"https://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}"

    try:
        # timeout avoids hanging forever
        response = requests.get(url, timeout=5)

        # If HTTP code is not 200 -> API error
        if response.status_code != 200:
            print("Error: Could not connect to API. Check your API key or city name.")
            return

        weather_data = response.json()

        print("\nWeather Details (JSON):")
        print(json.dumps(weather_data, indent=4))

    except requests.exceptions.Timeout:
        print("Error: Request timed out. Please check your network connection.")

    except requests.exceptions.ConnectionError:
        print("Error: Unable to connect to the internet or API server.")

    except Exception as e:
        print(f"Unexpected Error: {e}")


# --------- MAIN PROGRAM ---------
city = input("Enter city name: ")
get_weather(city)
```



Task 2: Add Error Handling for Invalid API Calls

- **Instructions:**
  - Modify your code from Task 1 to handle errors.
  - Include try/except (Python) or try/catch (JavaScript) blocks.

- Handle cases like:
  - Invalid URL
  - Network timeout
  - Wrong API key (if required)
- Print user-friendly error messages.

- **Expected Output:**
  - If the API works, the result is shown as in Task 1.
  - If there's an error, output:

Error: Could not connect to API. Check your API key or network connection.

Prompt: write a python function that display weather details of a city using weather api with error handling. Display weather details as JSON output

```python
import requests # type: ignore
import json
API_KEY = "7faac7ffaaf6687d935dbc1a59dd5a20"
def get_weather_with_errors(city):
    url = "http://api.openweathermap.org/data/2.5/weather?q=CITY_NAME&appid=7faac7ffaaf6687d935dbc1a59dd5a20&units=metric

    try:
        response = requests.get(url, timeout=5)
        response.raise_for_status()
        data = response.json()

        print(json.dumps(data, indent=4))
        return data
    except requests.exceptions.Timeout:
        print("Error: API request timed out.")
    except requests.exceptions.ConnectionError:
        print("Error: Could not connect to API. Check your internet.")
    except requests.exceptions.HTTPError:
        print("Error: Invalid city or API key.")
    except Exception as e:
        print("Unexpected Error:", str(e))
    return None
get_weather_with_errors("Hyderabad")
```

To use this code:

1. **First, you'll need to sign up for a free API key at Open Weather Map: https://openweathermap.org/api**
2. **Replace 'YOUR_API_KEY' in the code with your actual API key**
3. **Install the required requests package by running:**

```
    "cod": 200
}
PS C:\Users\91832\OneDrive\Documents\Desktop\AI Assignments> cd "c:\Users\91832\OneDrive\Documents\Desktop\AI Assignments"
PS C:\Users\91832\OneDrive\Documents\Desktop\AI Assignments> python -u "c:\Users\91832\OneDrive\Documents\Desktop\AI Assignments\Task-15.2.py
Error: Invalid city or API key.
PS C:\Users\91832\OneDrive\Documents\Desktop\AI Assignments>
```

Task 3: Extract and Display Specific Data

- **Instructions:**
  1. From the API response (e.g., weather API), extract

specific fields (temperature, humidity, description).

2. Display them in a user-friendly format (not raw JSON).

```python
import requests

def get_weather(city_name):
    api_key = 'd0d9c29cd5e0f6aa41e9370ad3874404'  # Your API key
    url = f"http://api.openweathermap.org/data/2.5/weather?q={city_name}&appid={api_key}&units=

    try:
        response = requests.get(url)
        response.raise_for_status()  # Raise error for bad responses

        data = response.json()

        # Extract required fields
        city = data['name']
        temperature = data['main']['temp']
        humidity = data['main']['humidity']
        description = data['weather'][0]['description']

        # Display in user-friendly format
        print(f"City: {city}")
        print(f"Temperature: {temperature}°C")
        print(f"Humidity: {humidity}%")
        print(f"Weather: {description.capitalize()}")

    except requests.exceptions.HTTPError as http_err:
        print(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as err:
        print(f"Other error occurred: {err}")
    except KeyError:
        print("Error: Unexpected response format from API")

# Example usage
city_input = input("Enter city name: ")
```

- **Expected Output:**
- City: London
- Temperature: 18°C
- Humidity: 60%

Weather: Clear sky

```
Error: Invalid city or API key.
PS C:\Users\91832\OneDrive\Documents\Desktop\AI Assignments> cd "c
PS C:\Users\91832\OneDrive\Documents\Desktop\AI Assignments> pytho
.3.py"
Enter city name: Warangal
City: Warangal
Temperature: 24.37°C
Humidity: 52%
Weather: Broken clouds
PS C:\Users\91832\OneDrive\Documents\Desktop\AI Assignments>
```

Task 4: Build a Function with Parameters

- **Instructions:**
  - Write a function that accepts a parameter (e.g., city name

for weather API).

- The function should call the API dynamically based on user input.
- Include error handling if the city is invalid.

```python
def display_weather(city_name):
    url = f"http://api.openweathermap.org/data/2.5/weather?q={city_name}&appid={api_key}&units
    
    try:
        response = requests.get(url)
        data = response.json()

        # Check if city is found
        if response.status_code == 404 or data.get('cod') == '404':
            print("Error: City not found. Please enter a valid city.")
            return

        # Extract required fields
        city = data['name']
        temperature = data['main']['temp']
        humidity = data['main']['humidity']
        description = data['weather'][0]['description']

        # Display in user-friendly format
        print(f"City: {city}")
        print(f"Temperature: {temperature}°C")
        print(f"Humidity: {humidity}%")
        print(f"Weather: {description.capitalize()}")

    except requests.exceptions.RequestException as err:
        print(f"Error: Unable to fetch data. {err}")

# Example usage
display_weather("New York")
display_weather("xyz123")
```

- **Expected Output:**
  - Input: "New York"
  - Output:
  - City: New York
  - Temperature: 22°C
  - Humidity: 55%
  - Weather: Few clouds
  - Input: "xyz123"
  - Output:

Error: City not found. Please enter a valid city.

```
PS C:\Users\91832\OneDrive\Documents\Desktop\AI Assignments> python -u "c:\Use
.4.py"
City: New York
Temperature: 1.87°C
Humidity: 60%
Weather: Scattered clouds
Error: City not found. Please enter a valid city.
PS C:\Users\91832\OneDrive\Documents\Desktop\AI Assignments>
```

Prompt: write a python function that display weather details of a

city using weather api with error handling. Display weather details in user friendly format

---

Task 5: Store API Results Locally

- **Instructions:**
  - Extend your function from Task 4.
  - Save the extracted API results into a local file (results.json or results.txt).
  - Each new request should append results without overwriting old ones**.**
- **Expected Output:**
  - Console still shows formatted output.
  - A local file results.json is created/updated with stored responses like:
  - [
  - {"city": "London", "temp": 18, "humidity": 60, "weather": "Clear sky"},
  - {"city": "New York", "temp": 22, "humidity": 55, "weather": "Few clouds"}

]

```python
import requests
import json
import os

def get_weather_and_store(city_name):
    api_key = input("Enter your OpenWeatherMap API key: ").strip()  # Ask user for API key
    base_url = "http://api.openweathermap.org/data/2.5/weather"

    try:
        # Make the API request
        response = requests.get(base_url, params={"q": city_name, "appid": api_key, "units": "metric"})

        # Raise exception if response code is not 200
        response.raise_for_status()

        data = response.json()

        # Extract necessary details
        weather_info = {
            "city": data["name"],
            "temp": data["main"]["temp"],
            "humidity": data["main"]["humidity"],
            "weather": data["weather"][0]["description"].capitalize()
        }

        # Print formatted JSON output
        print(json.dumps(weather_info, indent=4))

        # Store results in local file (append if file exists)
        file_path = "results.json"
        if os.path.exists(file_path):
            with open(file_path, "r+", encoding="utf-8") as f:
```

Prompt: write a python function that display weather details of a city using weather api with error handling. Display weather details as JSON output. Store the weather details in current directory as text file, every run output will append

✅ Deliverables (For All Tasks)

1. AI-generated prompts for code and test case generation.
2. At least 3 assert test cases for each task.
3. AI-generated initial code and execution screenshots.
4. Analysis of whether code passes all tests.
5. Improved final version with inline comments and explanation.
6. Compiled report (Word/PDF) with prompts, test cases, assertions, code, and output.