

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
Program Name: B. Tech		Assignment Type: Lab	
Course Coordinator Name		Venkataramana Veeramsetty	
Instructor(s) Name		Dr. V. Venkataramana (Co-ordinator) Dr. T. Sampath Kumar Dr. Pramoda Patro Dr. Brij Kishor Tiwari Dr.J.Ravichander Dr. Mohammand Ali Shaik Dr. Anirodh Kumar Mr. S.Naresh Kumar Dr. RAJESH VELPULA Mr. Kundhan Kumar Ms. Ch.Rajitha Mr. M Prakash Mr. B.Raju Intern 1 (Dharma teja) Intern 2 (Sai Prasad) Intern 3 (Sowmya) NS_2 (Mounika)	
Course Code	24CS002PC215	Course Title	AI Assisted Coding
Year/Sem	II/I	Regulation	R24
Date and Day of Assignment	Week9 - Thursday	Time(s)	
Duration	2 Hours	Applicable to Batches	
AssignmentNumber: 18.4(Present assignment number)/ 24 (Total number of assignments)			
Q.No.	Question		<i>Expected Time to complete</i>
1	Lab 18 – API Integration: Connecting to External Services with Error Handling Lab Objectives: <ul style="list-style-type: none"> • Learn how to integrate Python programs with external REST APIs. • Understand API request/response handling using requests or 		Week9 - Thursday

	<p>similar libraries.</p> <ul style="list-style-type: none"> • Implement proper error handling for failed API calls (timeouts, invalid responses, rate limits). • Practice extracting and displaying meaningful information from API responses. 	
	<p>Task 1: Connect to a Public API</p> <ul style="list-style-type: none"> • Instructions: <ul style="list-style-type: none"> • 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: <ul style="list-style-type: none"> ◦ Program prints API response (e.g., weather data or posts from JSONPlaceholder). ◦ Output should be properly formatted JSON (not raw). <p>Prompt: write a python function that display weather details of a city using weather api without error handling. Display weather details as JSON output</p> <p>To use this code:</p> <ol style="list-style-type: none"> 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: <p>pip install requests</p> <p>You can then run the script and enter any city name to get its current weather details. The output will look something like this:</p>	

```
import requests
import json

def get_weather_details(city_name):
    api_key = 'df8afcace6b1e634119e1ccc754c8e9d'
    base_url = "https://api.openweathermap.org/data/2.5/weather"
    params = {
        'q': city_name,
        'appid': api_key,
        'units': 'metric'
    }
    response = requests.get(base_url, params=params)
    weather_data = response.json()
    print(json.dumps(weather_data, indent=4))

if __name__ == "__main__":
    city = input("Enter city name: ")
    get_weather_details(city)
```

OUTPUT:

```
ive/Desktop/AIAC/Lab-18.4/Task-1.py
Enter city name: warangal
{
  "coord": {
    "lon": 79.5833,
    "lat": 18
  },
  "weather": [
    {
      "id": 801,
      "main": "Clouds",
      "description": "few clouds",
      "icon": "02n"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 23.85,
    "feels_like": 24.3,
    "temp_min": 23.85,
```

```
        "temp_max": 23.85,
        "pressure": 1012,
        "humidity": 77,
        "sea_level": 1012,
        "grnd_level": 981
    },
    "visibility": 10000,
    "wind": {
        "speed": 0.93,
        "deg": 294,
        "gust": 1
    },
    "clouds": {
        "all": 16
    },
    "dt": 1761925058,
    "sys": {
        "country": "IN",
        "sunrise": 1761871218,
        "sunset": 1761912606
    },
    "timezone": 19800,
    "id": 1252948,
    "name": "Warangal",
    "cod": 200
}
```

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

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:

CODE:

```
def display_city_weather(city_name):  
    'appid': api_key,  
    'units': 'metric'  
}  
try:  
    response = requests.get(base_url, params=params, timeout=10)  
    response.raise_for_status()  
    weather_data = response.json()  
    if weather_data.get("cod") != 200:  
        print(f"Error: {weather_data.get('message', 'Unknown Error')})")  
    else:  
        print(json.dumps(weather_data, indent=4))  
except requests.RequestException as e:  
    print(f"An error occurred while connecting to the weather service: {e}")  
  
# Example usage:  
if __name__ == "__main__":  
    city = input("Enter city name: ")  
    display_city_weather(city)
```

OUTPUT:

● Enter city name: warangal

```
{  
    "coord": {  
        "lon": 79.5833,  
        "lat": 18  
    },  
    "weather": [  
        {  
            "id": 801,  
            "main": "Clouds",  
            "description": "few clouds",  
            "icon": "02n"  
        }  
    ],  
    "base": "stations",  
    "main": {  
        "temp": 23.85,  
        "feels_like": 24.3,  
        "temp_min": 23.85,  
        "temp_max": 23.85,  
        "pressure": 1012,  
        "humidity": 77,  
        "sea_level": 1012,  
        "grnd_level": 981  
    },  
    "visibility": 10000,  
    "wind": {  
        "speed": 0.93,  
        "deg": 294,  
        "gust": 1  
    },  
    "clouds": {  
        "all": 16  
    },  
    "dt": 1761925058,  
    "sys": {  
        "country": "IN",  
        "sunrise": 1761871218,  
        "sunset": 1761912606  
    },  
    "timezone": 19800,  
    "id": 1252948,  
    "name": "Warangal",  
    "cod": 200  
}
```

Task 3: Extract and Display Specific Data

- Instructions:

- From the API response (e.g., weather API), extract specific fields (temperature, humidity, description).
- Display them in a user-friendly format (not raw JSON).

- Expected Output:**

- City: London
- Temperature: 18°C
- Humidity: 60%
- Weather: Clear sky

CODE:

```
import requests
def display_weather_info(city_name):
    api_key = 'df8afcace6b1e634119e1ccc754c8e9d'
    base_url = "https://api.openweathermap.org/data/2.5/weather"
    params = {
        'q': city_name,
        'appid': api_key,
        'units': 'metric'
    }
    try:
        response = requests.get(base_url, params=params, timeout=10)
        response.raise_for_status()
        data = response.json()
        if data.get("cod") != 200:
            print(f"Error: {data.get('message', 'Unknown Error')}")
            return
        temp = data.get('main', {}).get('temp')
        humidity = data.get('main', {}).get('humidity')
        desc = data.get('weather', [{}])[0].get('description', '')
        print(f"City: {city_name}")
        if temp is not None:
            print(f"Temperature: {temp}°C")
        else:
            print("Temperature: Data not available")
        if humidity is not None:
            print(f"Humidity: {humidity}%")
        else:
            print("Humidity: Data not available")
        print(f"Weather: {desc.capitalize() if desc else 'Data not available'}")
    except requests.RequestException as e:
        print(f"An error occurred while connecting to the weather service: {e}")

if __name__ == "__main__":
    city = input("Enter city name: ")
    display_weather_info(city)
```

OUTPUT:

	<pre>Enter city name: warangal City: warangal Temperature: 23.85°C Humidity: 77% Weather: Few clouds</pre>	
	<p>Task 4: Build a Function with Parameters</p> <ul style="list-style-type: none"> Instructions: <ul style="list-style-type: none"> 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. Expected Output: <ul style="list-style-type: none"> Input: "New York" Output: City: New York Temperature: 22°C Humidity: 55% Weather: Few clouds Input: "xyz123" Output: <p>Error: City not found. Please enter a valid city.</p> <p>CODE:</p> <pre>import requests def display_city_weather_details(city_name): api_key = 'df8afcace6b1e634119e1ccc754c8e9d' base_url = "https://api.openweathermap.org/data/2.5/weather" params = { 'q': city_name, 'appid': api_key, 'units': 'metric' } try: response = requests.get(base_url, params=params, timeout=10) response.raise_for_status() data = response.json() if data.get("cod") == "404": print("Error: City not found. Please enter a valid city.") return elif data.get("cod") != 200: print(f"Error: {data.get('message', 'Unknown Error')}") return city = data.get("name", "Unknown") country = data.get("sys", {}).get("country", "Unknown") temp = data.get("main", {}).get("temp") humidity = data.get("main", {}).get("humidity") weather_desc = "" if data.get("weather") and isinstance(data["weather"], list) and len(data["weather"]) > 0: weather_desc = data["weather"][0].get("description", "")</pre>	

```

print(f"Weather Details for {city}, {country}:")
if temp is not None:
    print(f" Temperature: {temp}°C")
else:
    print(" Temperature: Data not available")
if humidity is not None:
    print(f" Humidity: {humidity}%")
else:
    print(" Humidity: Data not available")
print(f" Condition: {weather_desc.capitalize()} if weather_desc else 'Data not available'")
except requests.RequestException as e:
    print(f"An error occurred while connecting to the weather service: {e}")

if __name__ == "__main__":
    city = input("Enter city name: ")
    display_city_weather_details(city)

```

OUTPUT:

```

Enter city name: New York
Weather Details for New York, US:
Temperature: 12.53°C
Humidity: 63%
Condition: Overcast clouds

```

Enter city name: xyz123
An error occurred while connecting to the weather service: 404 Client Error: Not Found for url: https://api.openweathermap.org/data/2.5/weather?q=xyz123&appid=df8afcace6b1e634119e1ccc754c8e9d&units=metric

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"}

]

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

CODE:

```
import requests
import json
import os

def display_and_store_weather_details(city_name):
    api_key = 'df8afcace6b1e634119e1ccc754c8e9d'
    base_url = "https://api.openweathermap.org/data/2.5/weather"
    params = {
        'q': city_name,
        'appid': api_key,
        'units': 'metric'
    }
    try:
        response = requests.get(base_url, params=params, timeout=10)
        response.raise_for_status()
        data = response.json()
        if data.get("cod") != 200:
            print(f"Error: {data.get('message', 'Unknown Error')}")
            return
        weather_json = json.dumps(data, indent=4)
        print(weather_json)
        # Store to text file, appending
        filename = "weather_details.txt"
        with open(filename, "a", encoding='utf-8') as f:
            f.write(weather_json + "\n")
        print(f"Weather details appended to {filename}")
    except requests.RequestException as e:
        print(f"An error occurred while connecting to the weather service: {e}")

if __name__ == "__main__":
    city = input("Enter city name: ")
    display_and_store_weather_details(city)
```

OUTPUT:

```
Enter city name: warangal
{
  "coord": {
    "lon": 79.5833,
    "lat": 18
  },
  "weather": [
    {
      "id": 801,
      "main": "Clouds",
      "description": "few clouds",
      "icon": "02n"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 23.85,
    "feels_like": 24.3,
    "temp_min": 23.85,
    "temp_max": 23.85,
    "pressure": 1012,
    "humidity": 77,
    "sea_level": 1012,
    "grnd_level": 981
  },
  "visibility": 10000,
  "wind": {
    "speed": 0.93,
    "deg": 294,
    "gust": 1
  },
  "clouds": {
    "all": 16
  },
  "dt": 1761925967,
  "sys": {
    "country": "IN",
    "sunrise": 1761871218,
    "sunset": 1761912606
  },
  "timezone": 19800,
  "id": 1252948,
  "name": "Warangal",
  "cod": 200
}
Weather details appended to weather_details.txt
```

Deliverables (For All Tasks)

	<ol style="list-style-type: none">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.	
--	---	--