

Lab 18– API Integration: Connecting to external services with error handling

Name: YASHWANTH

Enrollment Number: 2503A51L42

Assignment Number:18.2

Lab Question 1 : Weather Forecasting API

Task 1

Prompt:

Write a Python script using the OpenWeatherMap API to fetch and display current temperature and weather description for a given city with error handling for missing or invalid API key.

Code:

```
assignment 18.2 > lab q1 > task.py > ...
1 """
2 Simple Python Weather App
3 Fetches and displays the current temperature and weather description
4 for any city using the OpenWeatherMap API.
5 """
6
7 import requests
8
9 # 🔑 Your personal API key (replace this with your actual key)
10 API_KEY = "ee15900b8a955f7c9c7d2f83f02ae831"
11 API_URL = "https://api.openweathermap.org/data/2.5/weather"
12
13
14 def fetch_weather(city: str) -> dict:
15     """Fetch weather data from OpenWeatherMap API for the given city."""
16     params = {"q": city, "appid": API_KEY, "units": "metric"}
17
18     try:
19         response = requests.get(API_URL, params=params, timeout=10)
20         response.raise_for_status()
21     except requests.exceptions.RequestException as e:
22         print(f"⚠ Network error: {e}")
23         return None
24
25     data = response.json()
26
27     if data.get("cod") != 200:
28         print(f"✗ Error: {data.get('message', 'Unknown error')}")
29         return None
30
31     return data
32
33
34 def main():
35     print("🌤 Welcome to the Simple Weather App 🌩")
36     city = input("Enter city name: ").strip()
37
38     if not city:
39         print("⚠ Please enter a valid city name.")
40         return
41
42     weather_data = fetch_weather(city)
43
44     if weather_data:
45         temp = weather_data["main"]["temp"]
46         description = weather_data["weather"][0]["description"].capitalize()
47         print(f"\n📍 {city}")
48         print(f"🌡 Temperature: {temp}°C")
49         print(f"☁️ Weather: {description}")
50     else:
51         print("Failed to fetch weather data.")
52
53
54 if __name__ == "__main__":
55     main()
```

Output:

```
PS C:\Users\Suhana_Rehan\OneDrive\Desktop\A  
ams\Python\Python312\python.exe" "c:/Users/  
2/lab q1/task.py"  
☁️ Welcome to the Simple Weather App ☁️  
Enter city name: warangal  
  
📍 warangal  
🌡️ Temperature: 24.61°C  
☁️ Weather: Overcast clouds  
PS C:\Users\Suhana_Rehan\OneDrive\Desktop\A
```

Observations:

- I used AI to help me understand how to send API requests using the `requests` module.
 - It also showed me how to check if my API key is missing or wrong.
 - I learned that JSON responses can be accessed like dictionaries in Python.

Task 2

Prompt:

Extend the weather script to save weather data into a CSV file without duplicates and handle file read/write errors.

Code:

Output:

```
👉 Welcome to the Simple Weather App 🌈
Enter city name: warangal

📍 warangal
🌡️ Temperature: 24.61°C
☁️ Weather: Overcast clouds
✅ Saved weather data for 'warangal' to weather_data.csv.
PS C:\Users\Suhana Rehan\OneDrive\Desktop\AI assistant coding & "C:/Users/Suhana Rehan.exe" "c:/Users/Suhana Rehan/OneDrive/Desktop/AI assistant coding/assignment 18.2/lab q1
👉 Welcome to the Simple Weather App 🌈
Enter city name: warangal

📍 warangal
🌡️ Temperature: 24.61°C
☁️ Weather: Overcast clouds
⚠️ Weather for 'warangal' already exists in weather_data.csv. Skipping duplicate entry.

📄 weather_data.csv > 📁 data
1   City,Temperature (°C),Weather Description
2   warangal,24.61,Overcast clouds
3
```

Observations:

- I learned how to use csv.DictWriter to store data safely.
- The AI helped me avoid duplicate city entries by checking before writing.
- I also handled file errors using try and except blocks.

Lab Question 2: Currency Exchange Rate API

Task 1

Prompt:

Write a Python script that takes user input (amount, source, target currency) and fetches the latest exchange rate with error handling for invalid currency codes.

Code:

```
assignment 18.2 > lab2 > task1.py > fetch_exchange_rate
1 import requests
2
3 API_KEY = "fdded72fb50ef5a4d05aaa13c1d95692e"
4 BASE_URL = "https://openexchangerates.org/api/v6/latest.json" # reliable free exchange rate API
5
6 def fetch_exchange_rate(source_currency, target_currency):
7     """Fetch exchange rate from source to target currency."""
8     try:
9         response = requests.get(f"{BASE_URL}{source_currency.upper()}")
10        response.raise_for_status() # raise error for bad responses
11        data = response.json()
12
13        if data.get("result") != "success":
14            raise ValueError("Invalid source currency or API error.")
15
16        rates = data.get("rates", {})
17        if target_currency.upper() not in rates:
18            raise ValueError("Invalid target currency code.")
19
20        return rates[target_currency.upper()]
21
22    except requests.exceptions.RequestException as e:
23        print("⚠️ Network or connection error:", e)
24    except ValueError as e:
25        print("⚠️", e)
26    except Exception as e:
27        print("⚠️ Unexpected error:", e)
28
29    return None
30
31
32 def main():
33     try:
34         amount = float(input("Enter amount: "))
35         source = input("Enter source currency (e.g., USD, EUR, INR): ").strip()
36         target = input("Enter target currency (e.g., USD, EUR, INR): ").strip()
37
38         rate = fetch_exchange_rate(source, target)
39         if rate:
40             converted = amount * rate
41             print(f"\nconverted {amount:.2f} {source.upper()} = {converted:.2f} {target.upper()}")
42         else:
43             print("Conversion failed. Please check your currency codes.")
44     except ValueError:
45         print("⚠️ Please enter a valid numeric amount.")
46
47
48 if __name__ == "__main__":
49     main()
```

Output:

```
/Users/Suhana Rehan/OneDrive/Desktop/AI assistant coding/as
1 Enter amount: 1000
Enter source currency (e.g., USD, EUR, INR): usd
Enter target currency (e.g., USD, EUR, INR): inr
2
3 1000.00 USD = 88250.21 INR
```

Observations:

- I used AI to help create input prompts and handle wrong currency codes.
 - I understood how to check if the API response was valid before using it.
 - The script now gives a clear error message when something goes wrong.
-

Task 2

Prompt:

Add retry logic to the currency script to attempt the API call up to three times if it fails, and log all errors into a local file.

Code:

```
assignment18.2 > lab2 > task2.py > ...
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75

# --- Configuration ---
API_KEY = "fddad72f5b0ef5a4d05aa13c1d95692e"
BASE_URL = "https://open.er-api.com/v6/latest"
MAX_RETRIES = 3
LOG_FILE = "error_log.txt"

# --- Logging Setup ---
logging.basicConfig(
    filename=LOG_FILE,
    level=logging.ERROR,
    format="%(asctime)s - %(levelname)s - %(message)s"
)

def fetch_exchange_rate(source_currency, target_currency):
    """Fetch exchange rate with retry logic and error logging."""
    for attempt in range(1, MAX_RETRIES + 1):
        try:
            print(f"\n  Attempt {attempt} to fetch exchange rate...")
            response = requests.get(f"{BASE_URL}{source_currency.upper()}", timeout=10)
            response.raise_for_status()

            data = response.json()
            if data.get("result") != "success":
                raise ValueError("Invalid source currency or API returned an error.")

            rates = data.get("rates", {})
            if target_currency.upper() not in rates:
                raise ValueError("Invalid target currency code.")

            return rates[target_currency.upper()]

        except requests.exceptions.RequestException as e:
            logging.error("Network error (Attempt {attempt}): {e}")
            print("⚠ Network issue on attempt {attempt}. Retrying...")
            time.sleep(2)

        except ValueError as e:
            logging.error("Value error: {e}")
            print("⚠ {e}")
            break # no need to retry invalid currency codes

        except Exception as e:
            logging.error("Unexpected error: {e}")
            print("⚠ Unexpected error occurred.")
            break

    print("❌ Failed to fetch exchange rate after multiple attempts.")
    return None

def main():
    try:
        amount = float(input("Enter amount: "))
        source = input("Enter source currency (e.g., USD, EUR, INR): ").strip()
        target = input("Enter target currency (e.g., USD, EUR, INR): ").strip()

        rate = fetch_exchange_rate(source, target)
        converted = amount * rate
        print(f"\n  {amount:.2f} ({source.upper()}) - {converted:.2f} ({target.upper()})")

    except ValueError:
        print("⚠ Please enter a valid numeric amount.")
        logging.error("Invalid numeric amount entered by user.")

    if __name__ == "__main__":
        main()
```

Output:

The terminal window shows the execution of a Python script named task2.py. It prompts for source and target currencies and an amount. It then attempts to fetch exchange rates from an API, handling errors like invalid input and API failures. The output ends with a message about conversion failing due to multiple attempts. To the right of the terminal, a file named error.log.txt is shown with two entries: one for an invalid numeric amount and another for an invalid source currency or API error.

```
Enter amount: 3000
Enter source currency (e.g., USD, EUR, INR): inr
Enter target currency (e.g., USD, EUR, INR): eur
⌚ Attempt 1 to fetch exchange rate...

⌚ 3000.00 INR = 29.21 EUR
PS C:\Users\Suhana Rehan\OneDrive\Desktop\AI assistant coding>
1/Programs/Python312/python.exe "c:/Users/Suhana Rehan/assignment 18.2/lab2/task2.py"
Enter amount: jnlkj
⚠️Please enter a valid numeric amount.
PS C:\Users\Suhana Rehan\OneDrive\Desktop\AI assistant coding>
1/Programs/Python312/python.exe "c:/Users/Suhana Rehan/assignment 18.2/lab2/task2.py"
Enter amount: 49999
Enter source currency (e.g., USD, EUR, INR): kin
Enter target currency (e.g., USD, EUR, INR): rur
⌚ Attempt 1 to fetch exchange rate...
⚠️Invalid source currency or API returned an error.
✖️Failed to fetch exchange rate after multiple attempts.
Conversion failed. Please check your inputs or try again later.

error.log.txt
1 2025-10-28 13:08:28,916 - ERROR - Invalid numeric amount entered by user.
2 2025-10-28 13:08:45,954 - ERROR - Value error: Invalid source currency or API returned an error.
```

Observations:

- AI showed me how to use a simple retry loop with for and try.
- I created an error log file using basic file handling.
- I learned that logging helps when APIs fail due to server issues.

Lab Question 3 – News Headlines API

Task 1

Prompt:

Write a Python script to fetch and print the top 5 technology headlines from a news API, with timeout error handling.

Code:

```
assignment 18.2 > lab3 > ⚡ task1.py > ⚡ fetch_tech_news
 1  import requests
 2
 3  API_KEY = "9cac0a9af334958987e781c4a196c75"
 4  URL = "https://newsapi.org/v2/top-headlines"
 5
 6  def fetch_tech_news():
 7      params = {
 8          "category": "technology",
 9          "language": "en",
10          "pageSize": 5, # fetch only top 5 headlines
11          "apiKey": API_KEY
12      }
13
14      try:
15          response = requests.get(URL, params=params, timeout=5)
16          response.raise_for_status() # raise error for bad status codes
17          data = response.json()
18
19          if data.get("status") != "ok":
20              print("⚠️ Error fetching news:", data.get("message", "Unknown error"))
21              return
22
23          print("\n🕒 Top 5 Technology Headlines:\n")
24          for i, article in enumerate(data.get("articles", []), 1):
25              print(f"[{i}]. {article['title']}")
26              print(f"  Source: {article['source']['name']}")
27              print(f"  URL: {article['url']}\n")
28
29      except requests.Timeout:
30          print("✖️ Request timed out. Please check your internet connection and try again.")
31      except requests.RequestException as e:
32          print(f"✖️ Network error: {e}")
33      except Exception as e:
34          print(f"⚠️ Unexpected error: {e}")
35
36  if __name__ == "__main__":
37      fetch_tech_news()
```

Output:

```
PS C:\Users\Suhana Rehan\OneDrive\Desktop\AI assistant coding & "C:/Users/Suhana Rehan/AppData/Local/Programs/Python/exe" "c:/Users/Suhana Rehan/OneDrive/Desktop/AI assistant coding/assignment_18.2/lab3/task1.py"
[+] Top 5 Technology Headlines:

1. Retroid Pocket 6 and Pocket G2 Officially Unveiled - Retro Handhelds
Source: Retrohandhelds.gg
URL: https://retrohandhelds.gg/retroid-pocket-6-and-pocket-g2-officially-unveiled/

2. This Week's Japanese Game Releases: Dragon Quest I & II HD-2D Remake, Tales of Xillia Remastered, more - Gematsu
Source: Gematsu
URL: https://www.gematsu.com/2025/10/this-weeks-japanese-game-releases-dragon-quest-i-ii-hd-2d-remake-tales-of-xillia

3. Cancelled God of War PS5 Game Leaks in Screenshots - Push Square
Source: Push Square
URL: https://www.pushsquare.com/news/2025/10/cancelled-god-of-war-ps5-game-leaks-in-screenshots

4. Xbox take massive 200+ foot TV into the air over Miami to break bizarre gaming world records - supercarblondie.com
Source: Supercarblondie.com
URL: https://supercarblondie.com/xbox-ninja-gaiden-4-helicopter-record-swae-lee/

5. The Neuroscience-Based Nike Mind 001 Appears in "Black" - hypebeast.com
Source: HYPEBEAST
URL: https://hypebeast.com/2025/10/nike-mind-001-black-hq4307-001-release-info
```

Observations:

- AI helped me understand how to use the timeout parameter in requests.
- The script now prints headlines clearly in the console.
- I learned how to handle slow or failed responses without crashing.

Task 2

Prompt:

Clean and preprocess the headlines by removing special characters and converting them to title case, handling empty or null values.

Code:

```
assignment 18.2 > lab3 > task2.py > fetch_tech_news
  1  import requests
  2  import re
  3  API_KEY = "9cac00a9af334958987e781c4a196c75"
  4  URL = "https://newsapi.org/v2/top-headlines"
  5  def clean_headline(text):
  6      """Remove special characters and convert to title case."""
  7      if not text or not isinstance(text, str):
  8          return "No Title Available"
  9      # Remove special characters except letters, numbers, spaces, and basic punctuation
 10      cleaned = re.sub(r"[^A-Za-z0-9 ,!?'-]", "", text)
 11      return cleaned.title()
 12  def fetch_tech_news():
 13      params = {
 14          "category": "technology",
 15          "language": "en",
 16          "pageSize": 5,
 17          "apiKey": API_KEY
 18      }
 19      try:
 20          response = requests.get(URL, params=params, timeout=5)
 21          response.raise_for_status()
 22          data = response.json()
 23          if data.get("status") != "ok":
 24              print("⚠ Error fetching news:", data.get("message", "Unknown error"))
 25              return
 26          print("\n[+] Top 5 Cleaned Technology Headlines:\n")
 27          for i, article in enumerate(data.get("articles", []), 1):
 28              raw_title = article.get("title", "")
 29              clean_title = clean_headline(raw_title)
 30              source = article.get("source", {}).get("name", "Unknown Source")
 31              url = article.get("url", "No URL Provided")
 32              print(f"{i}. {clean_title}")
 33              print(f"  Source: {source}")
 34              print(f"  URL: {url}\n")
 35      except requests.Timeout:
 36          print("✖ Request timed out. Please check your internet connection and try again.")
 37      except requests.RequestException as e:
 38          print(f"✖ Network error: {e}")
 39      except Exception as e:
 40          print(f"⚠ Unexpected error: {e}")
 41  if __name__ == "__main__":
 42      fetch_tech_news()
```

Output:

Top 5 Cleaned Technology Headlines:

1. Retroid Pocket 6 And Pocket G2 Officially Unveiled - Retro Handhelds
Source: Retrohandhelds.gg
URL: <https://retrohandhelds.gg/retroid-pocket-6-and-pocket-g2-officially-unveiled>
2. This Weeks Japanese Game Releases Dragon Quest I Ii Hd-2D Remake, Tales Of Xillia 2, Final Fantasy VII Remake Part 2, And More
Source: Gematsu
URL: <https://www.gematsu.com/2025/10/this-weeks-japanese-game-releases-dragon-quest-ii-hd-2d-remake-tales-of-xillia-2-final-fantasy-vii-remake-part-2-and-more>
3. Cancelled God Of War Ps5 Game Leaks In Screenshots - Push Square
Source: Push Square
URL: <https://www.pushsquare.com/news/2025/10/cancelled-god-of-war-ps5-game-leaks-in-screenshots>
4. Xbox Take Massive 200 Foot Tv Into The Air Over Miami To Break Bizarre Gaming World Record
Source: Supercarblondie.com
URL: <https://supercarblondie.com/xbox-ninja-gaiden-4-helicopter-record-swee-leet>
5. The Neuroscience-Based Nike Mind 001 Appears In Black - Hypebeast.Com
Source: HYPEBEAST
URL: <https://hypebeast.com/2025/10/nike-mind-001-black-hq4307-001-release-info>

Observations:

- I used AI to clean text using regular expressions and title() function.
 - It helped me skip empty or None headlines safely.
 - The cleaned headlines look much more readable now.
-