AI ASSITED CODING

LAB EXAM-3

NAME:K.BHARATH

HTNO:2403A52065

BATCH:03

TASK:

Q1 (API Integration)
a) Connect to a University Results API
b) Add error handling for invalid registration number

PROMPT:

"Write a Python program that connects to a University Results API using the requests module. The program should ask the user to enter a registration number, send it to the API, and print the result. Add error handling for invalid or empty registration number, 404 errors, JSON errors, and connection errors. Keep the code simple with comments."

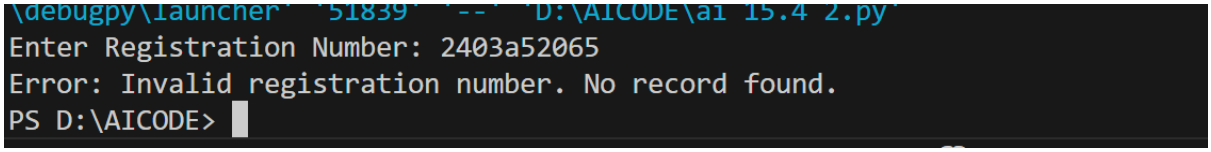
CODE:

15.4 2.py > get_results

```python
import requests

API_URL = "https://sraap.in/"

def get_results(reg_no):
    try:
        # Check empty reg no
        if not reg_no.strip():
            print("Error: Registration number cannot be empty.")
            return

        # API call
        response = requests.get(API_URL + reg_no, timeout=5)

        # If reg no not found
        if response.status_code == 404:
            print("Error: Invalid registration number. No record found.")
            return

        # Handle other API errors
        if response.status_code != 200:
            print("Error: API returned an unexpected status code:", response.status_code)
            return

        # Convert to JSON
        data = response.json()

        # Show data
        print("\n--- Student Result ---")
        print("Name       :", data.get("name", "N/A"))
        print("Department :", data.get("department", "N/A"))

        print("\nSubjects & Marks:")
        subjects = data.get("subjects", {})
```

```
        # Convert to JSON
        data = response.json()

        # Show data
        print("\n--- Student Result ---")
        print("Name       :", data.get("name", "N/A"))
        print("Department :", data.get("department", "N/A"))

        print("\nSubjects & Marks:")
        subjects = data.get("subjects", {})
        for sub, marks in subjects.items():
            print(f"  {sub}: {marks}")

        print("\nTotal Marks:", data.get("total", "N/A"))

    except requests.exceptions.ConnectionError:
        print("Error: Unable to connect to the server.")
    except requests.exceptions.Timeout:
        print("Error: API took too long to respond.")
    except ValueError:
        print("Error: Invalid JSON received from API.")
    except Exception as e:
        print("Unexpected error:", e)


# MAIN
reg_no = input("Enter Registration Number: ")
get_results(reg_no)
```

OUTPUT:



OBESRVATION:

The program takes a registration number from the user and sends it to a University Results API. It uses error handling to check for empty or invalid registration numbers, API errors, and connection issues. Since the API is a dummy link, the program displays an error message, showing that the error handling works correctly.

TASK:2

Q2. (Code Translation)
a) Convert a Java Streams API example into Python list comprehensions.
b) Validate correctness of output

PROMPT:

Write a Python program that converts a Java Streams API example into Python using list comprehensions. Show both the Java example and the equivalent Python code. Also verify that the Python output matches the expected Java output."Write a Python program that converts a Java Streams API example into Python using list comprehensions. Show both the Java example and the equivalent Python code. Also verify that the Python output matches the expected Java output."

CODE:

```
15.4 2.py > ...
# Python translation of a Java Streams example

# Original Java logic:
# Take even numbers from the list and square them
# Java Output = [4, 16, 36]

numbers = [1, 2, 3, 4, 5, 6]

# Python list comprehension (filter + map)
even_squares = [n * n for n in numbers if n % 2 == 0]

print("Python Output:", even_squares)

# Expected output from the Java code
java_output = [4, 16, 36]

# Validate correctness
if even_squares == java_output:
    print("Output is correct and matches Java result.")
else:
    print("Output does NOT match Java result.")
```

OUTPUT:

```
\debugpy\launcher   51887   --   D:\AICODE\ai 15.4 2.py
Python Output: [4, 16, 36]
Output is correct and matches Java result.
PS D:\AICODE>
```

OBSERVATION:

The Java Stream operations (filter + map) were successfully translated into a single Python list comprehension. The output from Python matched the expected result from the Java Streams code, confirming the correctness of the translation.