AI ASSISSTED CODING

Lab Assignment-18

NAME:NASRIN MD
BATCH:19
2503a51l23

Lab 18:

API Integration: Connecting to external services with
error handling.

Lab Objective

The objective of this lab is to provide students with hands-on experience in integrating external APIs into Python applications using AI-assisted coding tools. Students will:

1. Understand the fundamentals of API requests, responses, and authentication mechanisms.
2. Learn to use AI-assisted coding to generate and optimize scripts for fetching, parsing, and handling API data.
3. Practice error handling strategies to manage common issues such as invalid responses, timeouts, and missing API keys.
4. Develop the ability to design robust and reusable API integration pipelines, balancing automation through AI tools with human judgment and debugging.

Lab Question 1: Weather Forecasting API

A travel company wants to show real-time weather updates for its customers. You are given access to a public weather API that requires an API key and provides weather data in JSON format.

- **Task 1**

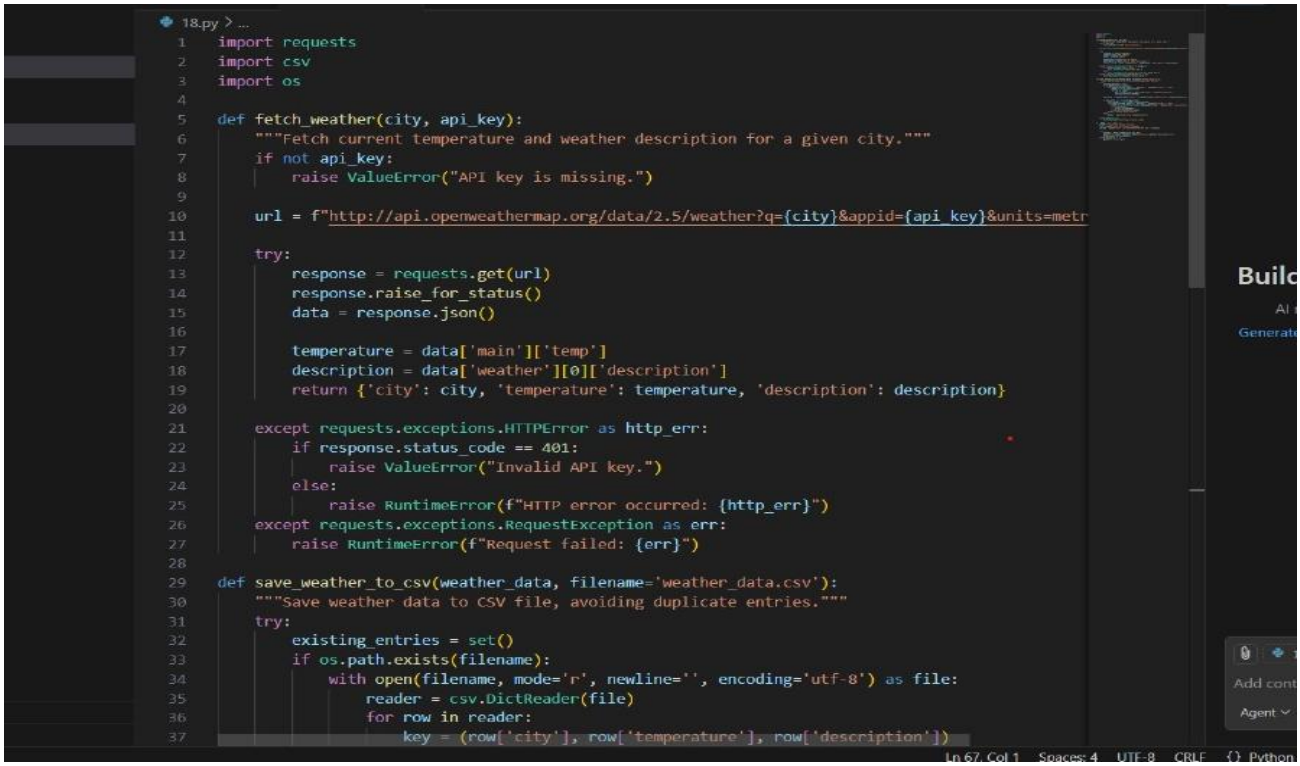
Prompt used:

write a script that fetches the current temperature and weather description for a given city. The script should handle errors if the API key is invalid or missing

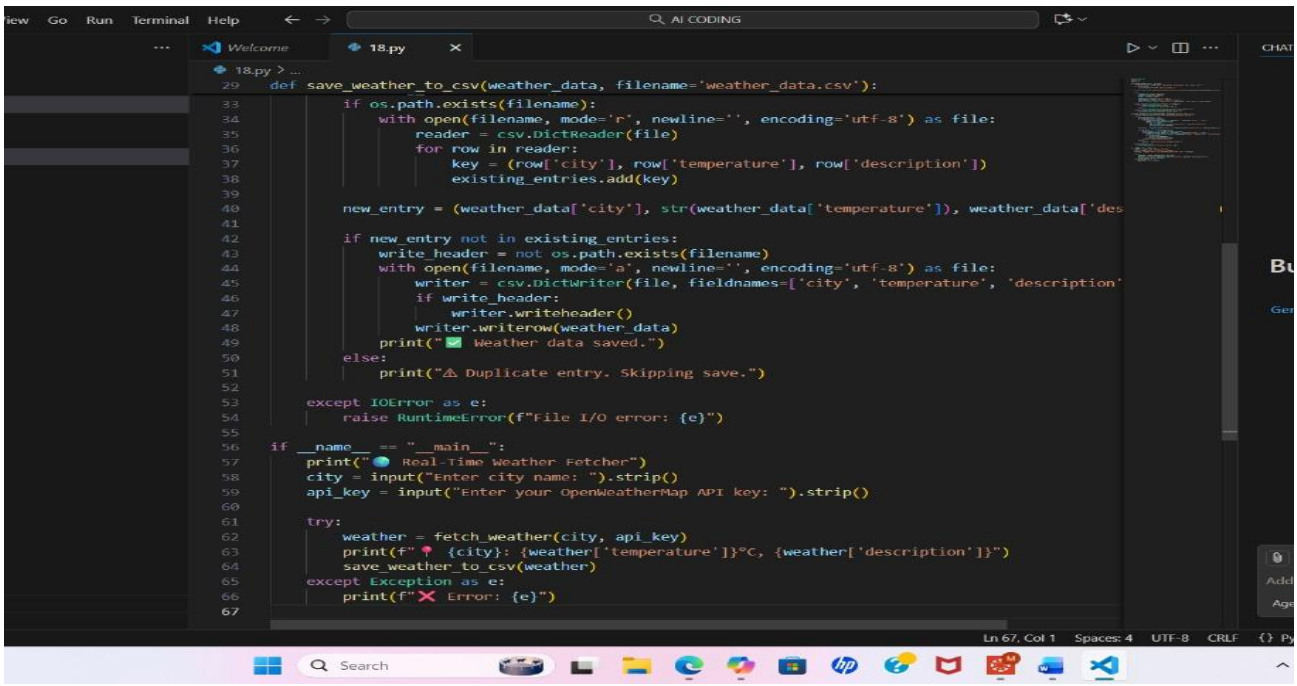
Task 2:

Prompt used:Extend the script to save the weather data into a local CSV file, ensuring that duplicate entries are avoided. Implement error handling for file I/O exceptions.

CODE:



```python
import requests
import csv
import os

def fetch_weather(city, api_key):
    """Fetch current temperature and weather description for a given city."""
    if not api_key:
        raise ValueError("API key is missing.")

    url = f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}&units=metr

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

        temperature = data['main']['temp']
        description = data['weather'][0]['description']
        return {'city': city, 'temperature': temperature, 'description': description}

    except requests.exceptions.HTTPError as http_err:
        if response.status_code == 401:
            raise ValueError("Invalid API key.")
        else:
            raise RuntimeError(f"HTTP error occurred: {http_err}")
    except requests.exceptions.RequestException as err:
        raise RuntimeError(f"Request failed: {err}")

def save_weather_to_csv(weather_data, filename='weather_data.csv'):
    """Save weather data to CSV file, avoiding duplicate entries."""
    try:
        existing_entries = set()
        if os.path.exists(filename):
            with open(filename, mode='r', newline='', encoding='utf-8') as file:
                reader = csv.DictReader(file)
                for row in reader:
                    key = (row['city'], row['temperature'], row['description'])
```



```python
def save_weather_to_csv(weather_data, filename='weather_data.csv'):
            if os.path.exists(filename):
                with open(filename, mode='r', newline='', encoding='utf-8') as file:
                    reader = csv.DictReader(file)
                    for row in reader:
                        key = (row['city'], row['temperature'], row['description'])
                        existing_entries.add(key)

            new_entry = (weather_data['city'], str(weather_data['temperature']), weather_data['des

            if new_entry not in existing_entries:
                write_header = not os.path.exists(filename)
                with open(filename, mode='a', newline='', encoding='utf-8') as file:
                    writer = csv.DictWriter(file, fieldnames=['city', 'temperature', 'description'
                    if write_header:
                        writer.writeheader()
                    writer.writerow(weather_data)
                print("✅ Weather data saved.")
            else:
                print("⚠ Duplicate entry. Skipping save.")

        except IOError as e:
            raise RuntimeError(f"File I/O error: {e}")

    if __name__ == "__main__":
        print("🌍 Real-Time Weather Fetcher")
        city = input("Enter city name: ").strip()
        api_key = input("Enter your OpenWeatherMap API key: ").strip()

        try:
            weather = fetch_weather(city, api_key)
            print(f"📍 {city}: {weather['temperature']}°C, {weather['description']}")
            save_weather_to_csv(weather)
        except Exception as e:
            print(f"❌ Error: {e}")
```

OUTPUT:

TASK1:

```
🌍 Real-Time Weather Fetcher
Enter city name: hyderabad
Enter your OpenWeatherMap API key: f703bc68694edb69901ccafad2b8a084
Enter city name: hyderabad
Enter your OpenWeatherMap API key: f703bc68694edb69901ccafad2b8a084
Enter your OpenWeatherMap API key: f703bc68694edb69901ccafad2b8a084
📍 hyderabad: 27.23°C, mist
✅ Weather data saved.
```

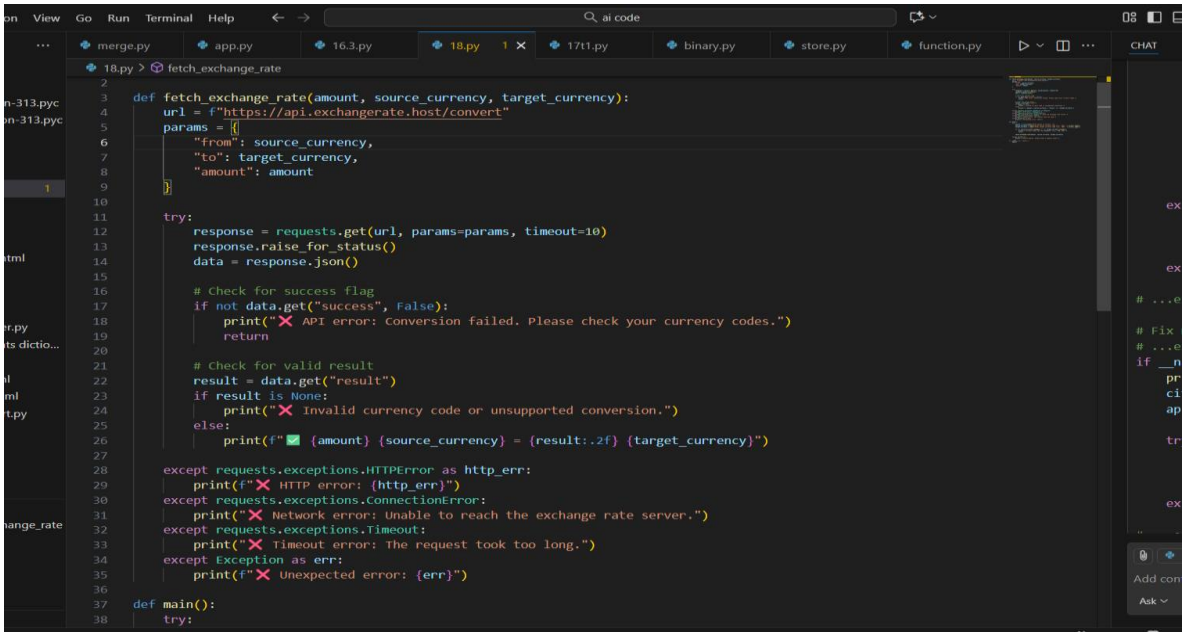
Lab Question 2:

Currency Exchange Rate API
A financial startup needs a tool to convert amounts between currencies using an exchange rate API. However, the API occasionally fails due to server downtime.

• **Task 1:**

Prompt used: Write a script (with AI assistance) that takes user input (amount, source currency, target currency) and fetches the latest exchange rate from the API. Include errors in handling invalid currency codes

CODE:



```python
def main():
    try:
        amount = float(input("Enter amount to convert: "))
        source_currency = input("Enter source currency code (e.g., USD): ").strip().upper()
        target_currency = input("Enter target currency code (e.g., INR): ").strip().upper()

        if not source_currency.isalpha() or not target_currency.isalpha():
            print("❌ Currency codes must be alphabetic (e.g., USD, EUR).")
            return

        fetch_exchange_rate(amount, source_currency, target_currency)

    except ValueError:
        print("❌ Invalid amount. Please enter a numeric value.")

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

OUTPUT:

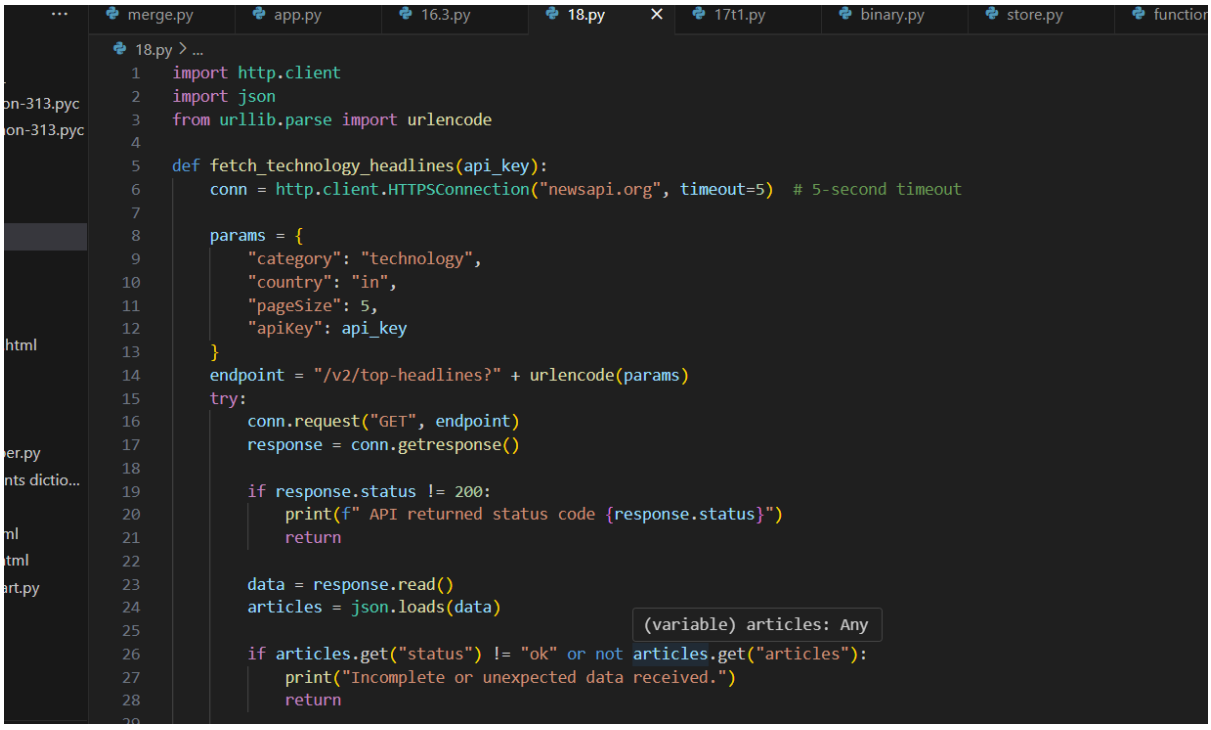
```
PS C:\Users\HASINI\OneDrive\Desktop\ai code> python 18.
Enter amount to convert: 1000
Enter source currency code (e.g., USD): INR
Enter target currency code (e.g., INR): USD
```

Task 2:

Prompt used:

Add logic to retry the request up to three times if the API call fails due to network or server issues and log all failed attempts into a local error log file.

CODE:



```
        print("\n Top 5 Technology Headlines:\n")
        for i, article in enumerate(articles["articles"], 1):
            print(f"{i}. {article['title']}")
            print(f"   Source: {article['source']['name']}")
            print(f"   URL: {article['url']}\n")

    except Exception as e:
        print(f"Request failed or timed out: {e}")
    finally:
        conn.close()

# Example usage
if __name__ == "__main__":
    YOUR_API_KEY = "d8b2ce6f8f2c489894b2625662092291"  # Replace with your actual API key
    fetch_technology_headlines(YOUR_API_KEY)
```

Output:

```
API returned status code 400
API returned status code 400
PS C:\Users\HASINI\OneDrive\Desktop\ai code>
```