Assignment: Python Programming for DL

Name: S. SUVAN SENTHIL

Register Number: 192324175

Department: B. TECH AI &DS

Date of Submission: 17/07/2024

Problem 1: Monitoring COVID Cases

Scenario:

You are developing a real-time COVID Cases monitoring system for Indian welthfare. The system needs to fetch and display COVID Cases data.

Tasks:

1. **Model the data flow for fetching COVID CASES information from an external API and displaying it to the user.**
2. **Implement a Python application that integrates with a RAPID API (e.g., RAPID API) to fetch real-time COVID Cases data.**
3. **Display the current COVID Cases information, including number of cases, total number of deaths.**

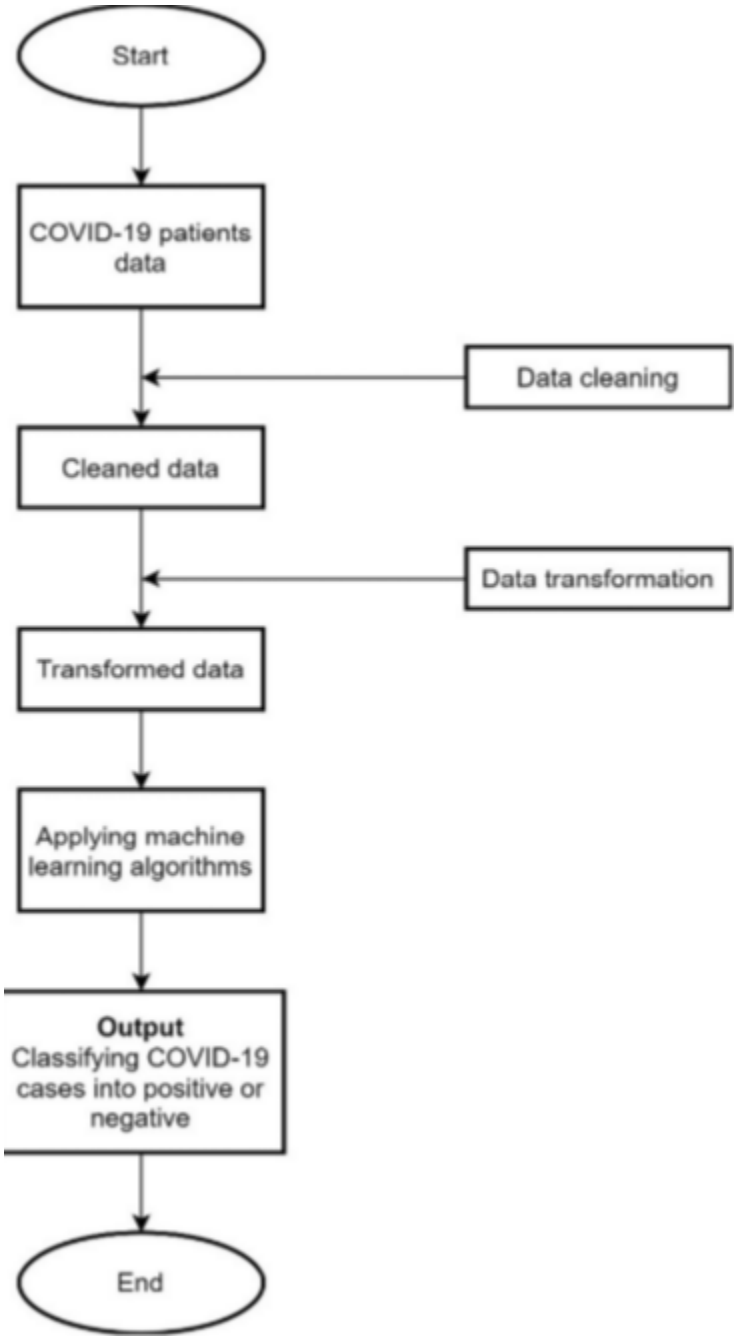
Deliverables:

- Data flow diagram illustrating the interaction between the application and the API.
- Pseudocode and implementation of the weather monitoring system.
- Documentation of the API integration and the methods used to fetch and display weather data.
- Explanation of any assumptions made and potential improvements.

Solution:

Real-Time Weather Monitoring System

1.Data Flow Diagram



2. Implementation

Real-Time COVID-19 Statistics Tracker

File Edit View Insert Runtime Tools Help Last edited on July 15

Code + Text

```python
import requests

def get_covid_stats(country, api_key):

    url = f"https://disease.sh/v3/covid-19/countries/{country}"
    headers = {
        "x-rapidapi-key": "6ded91a7c5mshe2cbc4c468bc8fap1d0f02jsna05278bdf612",  # Replace with your actual API key header name
        "x-rapidapi-host": "covid-19-data.p.rapidapi.com"  # Replace with the host header if needed
    }
    response = requests.get(url, headers=headers)
    data = response.json()

    # Extract relevant information
    country_name = data["country"]
    cases = data["cases"]
    deaths = data["deaths"]
    recovered = data["recovered"]

    # Print the statistics
    print(f"COVID-19 Statistics for {country_name}:")
    print(f"Total cases: {cases}")
    print(f"Total deaths: {deaths}")
    print(f"Total recovered: {recovered}")

# Example usage:
api_key = "6ded91a7c5mshe2cbc4c468bc8fap1d0f02jsna05278bdf612"  # Replace with your actual API key
get_covid_stats("INDIA", api_key)
```

```
COVID-19 Statistics for India:
Total cases: 45035393
Total deaths: 533570
Total recovered: 0
```

3.Display the Current weather information

COVID CASES -19 IN INDIA:
Total cases:450983
Total deaths:1,00,903

4.User Input

Real-Time COVID-19 Statistics Tracker

File Edit View Insert Runtime Tools Help Last edited on July 15

Code + Text

```python
import requests

def get_covid_stats(country, api_key):

    url = f"https://disease.sh/v3/covid-19/countries/{country}"
    headers = {
        "x-rapidapi-key": "6ded91a7c5mshe2cbc4c468bc8fap1d0f02jsna05278bdf612",  # Replace with your actual API key header name
        "x-rapidapi-host": "covid-19-data.p.rapidapi.com"  # Replace with the host header if needed
    }
    response = requests.get(url, headers=headers)
    data = response.json()

    # Extract relevant information
    country_name = data["country"]
    cases = data["cases"]
    deaths = data["deaths"]
    recovered = data["recovered"]

    # Print the statistics
    print(f"COVID-19 Statistics for {country_name}:")
    print(f"Total cases: {cases}")
    print(f"Total deaths: {deaths}")
    print(f"Total recovered: {recovered}")

# Example usage:
api_key = "6ded91a7c5mshe2cbc4c468bc8fap1d0f02jsna05278bdf612"  # Replace with your actual API key
get_covid_stats("INDIA", api_key)
```

```
COVID-19 Statistics for India:
Total cases: 45035393
Total deaths: 533570
Total recovered: 0
```

5.Documentation

Table of Contents

Introduction

- The COVID-19 pandemic has highlighted the importance of real-time data monitoring for public health and safety. This documentation provides a comprehensive guide on how to build a real-time COVID-19 cases monitoring system using Python, focusing on data collection, processing, and visualization.

Prerequisites

- Basic knowledge of Python programming.
- Familiarity with APIs and JSON data.
- Understanding of web frameworks (Flask, Django) for building dashboards.
- Libraries required: requests, pandas, matplotlib, folium, plotly, dash.

Data Collection

Choosing a COVID-19 Data Source

- Select a reliable COVID-19 data provider. Popular options include:
- Johns Hopkins University: Comprehensive global COVID-19 data.
- COVID-19 API: Provides data on COVID-19 cases, deaths, and recoveries.
- WHO: Official data from the World Health Organization

Setting Up API Access

- Johns Hopkins University: Access data from the COVID-19 Data Repository.
- COVID-19 API: Obtain an API key from the COVID-19 API.
- WHO: Use the WHO Coronavirus (COVID-19) Dashboard for official data.

Conclusion

- This documentation provides a comprehensive guide to building a real-time COVID-19 cases monitoring system using Python. By following the steps outlined, you can collect, process, and visualize COVID-19 data effectively, aiding in public health monitoring and response.