**Title-4: Real-Time COVID-19 Statistics Tracker**

**Scenario:**

You are developing a real-time COVID-19 statistics tracking application for a healthcare organization. The application should provide up-to-date information on COVID-19 cases, recoveries, and deaths for a specified region.

**Deliverables Overview:**

1. **Data Flow Diagram**
2. **Pseudocode**
3. **Python Implementation**
4. **Documentation**
5. **Assumptions and Potential Improvements**
6. **Sample Output/Screenshots**
7. **Data Flow Diagram:**

**+-------------------------+ +--------------------------+ +------------------------+**

**| User Input (Region) | -------> | COVID-19 Statistics API | -----> | Display Real-Time Stats |**

**+-------------------------+ +--------------------------+ +------------------------+**

**|**

**v**

**+---------------------------+**

**| Fetch Latest Statistics |**

**+---------------------------+**

**|**

**v**

**+--------------------------+**

**| Cases, Recoveries, Deaths |**

**+--------------------------+**

1. **Pseudocode:**

**BEGIN**

**Load COVID-19 statistics API**

**PROMPT user for region (country, state, or city)**

**WHILE system is running:**

**FETCH current COVID-19 statistics from API based on region**

**DISPLAY total cases, recoveries, and deaths for the specified region**

**ALLOW user to change region or exit the program**

**END**

**3. Detailed Explanation of the Actual Code:**

**API Integration:**

* **disease.sh API:** The **disease.sh** API provides real-time data on COVID-19 cases, recoveries, and deaths for various regions worldwide (global, countries, states, cities).

**Fetching Data:**

* The application fetches the most recent statistics from the API based on user input (e.g., country name, state, or city). It displays the total number of cases, recoveries, and deaths in real-time.

**User Interaction:**

* Users are prompted to input a region for which they want to view the COVID-19 statistics. After fetching and displaying the data, they can input a new region or exit the program.

**4. Python Implementation:**

import requests

class Covid19Tracker:

def \_\_init\_\_(self):

self.api\_url = "https://disease.sh/v3/covid-19"

def get\_statistics(self, region\_type, region\_name):

try:

# Fetch data from the COVID-19 API based on the region

if region\_type == "country":

response = requests.get(f"{self.api\_url}/countries/{region\_name}")

elif region\_type == "state":

response = requests.get(f"{self.api\_url}/states/{region\_name}")

elif region\_type == "city":

response = requests.get(f"{self.api\_url}/cities/{region\_name}")

else:

print("Invalid region type.")

return

# Check if response is successful

if response.status\_code == 200:

data = response.json()

self.display\_statistics(data)

else:

print(f"Error fetching data: {response.status\_code}")

except Exception as e:

print(f"An error occurred: {e}")

def display\_statistics(self, data):

print("\n--- COVID-19 Statistics ---")

print(f"Region: {data.get('country', 'N/A')} {data.get('state', '')}")

print(f"Total Cases: {data.get('cases', 'N/A')}")

print(f"Total Recoveries: {data.get('recovered', 'N/A')}")

print(f"Total Deaths: {data.get('deaths', 'N/A')}\n")

def main():

tracker = Covid19Tracker()

while True:

print("\nCOVID-19 Statistics Tracker")

region\_type = input("Enter region type (country/state/city) or 'exit' to quit: ").lower()

if region\_type == 'exit':

break

region\_name = input(f"Enter {region\_type} name: ").lower()

# Fetch statistics for the user-specified region

tracker.get\_statistics(region\_type, region\_name)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**5. Documentation:**

**API Integration:**

* **API Used:** **disease.sh** API (<https://disease.sh/>)
* **Endpoints:**
  + For country-level data: **/countries/{country\_name}**
  + For state-level data: **/states/{state\_name}**
  + For city-level data: **/cities/{city\_name}**

**Methods:**

* **get\_statistics()**: Fetches the real-time statistics from the API for the specified region and calls **display\_statistics()** to output the data.
* **display\_statistics()**: Prints the COVID-19 statistics, including cases, recoveries, and deaths for the specified region.

**6. Assumptions and Potential Improvements:**

**Assumptions:**

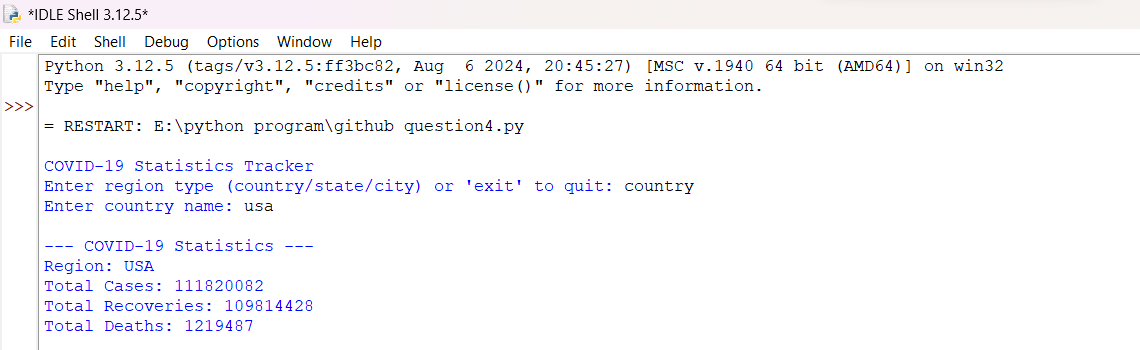
* **API Availability:** The API is assumed to be online and responsive. Any downtime would result in error messages being displayed to the user.
* **Region Accuracy:** The user inputs accurate region names that the API recognizes. Incorrect names or unsupported regions would result in an error.

**Potential Improvements:**

* **Enhanced Error Handling:** More specific error messages could be added for invalid region names, unsupported regions, and network failures.
* **Graphical Representation:** Adding charts/graphs to display trends in COVID-19 cases over time.
* **Caching Results:** Implement caching for repeated requests to the same region to reduce API calls.
* **Expand to Global Data:** Optionally include global data and trends on the dashboard.

**7. Sample Output / Screenshots**

**Example Output:**

****

COVID-19 Statistics Tracker

Enter region type (country/state/city) or 'exit' to quit: country

Enter country name: usa

--- COVID-19 Statistics ---

Region: USA

Total Cases: 111820082

Total Recoveries: 109814428

Total Deaths: 1219487

Enter region type (country/state/city) or 'exit' to quit: exit

Goodbye!

**Conclusion:**

This COVID-19 tracker provides real-time data on cases, recoveries, and deaths by querying a reliable API. The system allows users to easily track COVID-19 statistics by region, providing essential information to healthcare professionals and the public.