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

**4Scenario:**

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.

**Tasks:**

1. Model the data flow for fetching COVID-19 statistics from an external API and displaying it to the user.
2. Implement a Python application that integrates with a COVID-19 statistics API (e.g., disease.sh) to fetch real-time data.
3. Display the current number of cases, recoveries, and deaths for a specified region.
4. Allow users to input a region (country, state, or city) and display the corresponding COVID-19 statistics.

**Deliverables:**

* Data flow diagram illustrating the interaction between the application and the API.
* Pseudocode and implementation of the COVID-19 statistics tracking application.
* Documentation of the API integration and the methods used to fetch and display COVID-19 data.
* Explanation of any assumptions made and potential improvements.

**Approach:**

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

**| User Input |**

**| (Region Query) |**

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

**|**

**v**

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

**| Application Logic |**

**| (Fetch Data from API)|**

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

**|**

**v**

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

**| External COVID-19 |**

**| Statistics API |**

**| (e.g., disease.sh) |**

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

**|**

**v**

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

**| Display Results |**

**| (Cases, Recoveries, |**

**| Deaths for Region) |**

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

**The key steps are:**

1. The user inputs a region (country, state, or city) into the application.
2. The application sends a request to the COVID-19 API to fetch the real-time statistics for the specified region.
3. The COVID-19 API processes the request and returns the current case, recovery, and death data.
4. The application receives the COVID-19 statistics and displays the information to the user.

**Pseudocode:**

import requests

def get\_covid\_stats(region):

"""

Fetch COVID-19 statistics for the specified region from the API.

"""

api\_url = f"https://disease.sh/v3/covid-19/countries/{region}"

response = requests.get(api\_url)

data = response.json()

cases = data["cases"]

recoveries = data["recovered"]

deaths = data["deaths"]

return cases, recoveries, deaths

def display\_covid\_stats(cases, recoveries, deaths):

"""

Display the COVID-19 statistics in a user-friendly format.

"""

print(f"Current COVID-19 Statistics:")

print(f"- Total Cases: {cases:,}")

print(f"- Total Recoveries: {recoveries:,}")

print(f"- Total Deaths: {deaths:,}")

def main():

"""

Main function to handle user input and display COVID-19 statistics.

"""

region = input("Enter a country, state, or city: ")

cases, recoveries, deaths = get\_covid\_stats(region)

display\_covid\_stats(cases, recoveries, deaths)

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

main()

**Detailed explanation of the actual code:**

* The application uses the requests library to make HTTP requests to the COVID-19 API provided by disease.sh. The get\_covid\_stats function takes a region (country, state, or city) as input and returns the current number of cases, recoveries, and deaths for that region.
* The display\_covid\_stats function is responsible for formatting and printing the COVID-19 statistics in a user-friendly way. It takes the cases, recoveries, and deaths data as input and displays them with appropriate formatting (e.g., adding commas to large numbers).
* The main function is the entry point of the application. It prompts the user to enter a region, calls the get\_covid\_stats function to fetch the data, and then passes the results to the display\_covid\_stats function to display the information.

**Assumptions made (if any):**

* The application assumes that the disease.sh API is available and providing accurate real-time COVID-19 data.
* The application assumes that the user will input a valid region (country, state, or city) that the API can recognize.
* Potential Improvements:
* Add error handling to the application to gracefully handle API errors or invalid user input.
* Provide additional features, such as the ability to display historical COVID-19 data, trends, or visualizations.
* Integrate the application with a user interface (e.g., a web application or a mobile app) to improve the user experience.
* Allow users to select multiple regions and compare the COVID-19 statistics side-by-side.
* Provide the ability to set alerts or notifications for significant changes in COVID-19 statistics.

**Limitations:**

1. The API may have rate limits that restrict the number of requests.
2. The data may not always be up-to-date due to delays in reporting.
3. The application currently only handles countries; state and city-level queries may require additional endpoints.

**Code:**

import urllib.request

import json

# Function to get COVID-19 statistics from disease.sh API

def get\_covid\_data(location):

# Set the API endpoint and parameters

url = f"https://disease.sh/v3/covid-19/countries/{location}"

# Send a GET request to the API

with urllib.request.urlopen(url) as response:

# Load the JSON response

data = json.load(response)

# Extract the relevant COVID-19 data

cases = data["cases"]

recoveries = data["recovered"]

deaths = data["deaths"]

# Return the COVID-19 data as a dictionary

return {"cases": cases, "recoveries": recoveries, "deaths": deaths}

# Function to display the COVID-19 statistics

def display\_covid\_data(covid\_data):

# Print the COVID-19 data in a readable format

print("Current COVID-19 Statistics:")

print(f"Cases: {covid\_data['cases']}")

print(f"Recoveries: {covid\_data['recoveries']}")

print(f"Deaths: {covid\_data['deaths']}")

# Main function to run the program

def main():

# Get the location from the user

location = input("Enter the country, state, or city: ")

# Get the COVID-19 data

try:

covid\_data = get\_covid\_data(location)

except urllib.error.HTTPError:

print("Failed to retrieve COVID-19 data. Please check the location and try again.")

return

# Display the COVID-19 data

display\_covid\_data(covid\_data)

# Run the main function

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

main()

**Sample Output / Screen Shots**

