****

**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

**CAPSTONE PROJECT REPORT**

**By**

**M RAMYA SRI (192372144)**

**COURSE CODE:** CSA0813

**COURSE FACULTY:** DR. GEETHA

**PROJECT TITLE:** Real-Time COVID-19 Statistics Tracker

TABLE OF CONTENTS

|  |  |
| --- | --- |
| S.NO | TITLE |
| 1 | ABSTRACT |
| 2 | INTRODUCTION |
| 3 | MODULES |
| 4 | KEY COMPONENTS |
| 5 | TOOLS DESCRIPITION |
| 6 | ARCHITECURE DIAGRAM |
| 7 | SOURCE CODE |
| 8 | OUTPUT |
| 9 | FUTURE ENHANCEMENT |

|  |  |
| --- | --- |
| 10 | CONCLUTION |

Abstract:

In the evolving landscape of real-time data tracking, developing a COVID-19 statistics tracker presents a significant opportunity to blend functionality with user experience. This Python project is dedicated to creating an innovative and efficient COVID-19 tracker that stands out from existing solutions by offering a combination of accurate data display, design sophistication, and user adaptability. By leveraging Python's versatile libraries and frameworks, the project delivers a tool that not only tracks real-time COVID-19 statistics but also enhances user interaction through customization and advanced features.

Unlike many existing trackers, this project introduces a series of differentiators that enhance both functionality and user experience. It emphasizes a modern, customizable interface created with tkinter, ensuring a visually appealing and user-friendly design. Additionally, the tracker uses asynchronous data fetching methods to maintain high performance and efficiency, avoiding the issues of frequent polling that can lead to system slowdowns. The inclusion of customizable data filters, advanced error handling, and potential future features further sets this tracker apart from conventional solutions.

Introduction:

In today’s data-driven world, the need for accurate and timely COVID-19 information is paramount. While numerous COVID-19 trackers exist, this Python project aims to develop a tracker that distinguishes itself through innovative features and an enhanced user interface. Utilizing Python’s tkinter for graphical design and integrating real-time data APIs, the project seeks to offer a dynamic and informative COVID-19 statistics tracker that goes beyond the basic functionality of existing solutions.

Key Differentiators from Existing Solutions:

Enhanced User Interface and Customization:

Existing Solutions: Many trackers feature static, utilitarian designs focused primarily on data display without much emphasis on aesthetics or personalization.

Project Approach: This tracker leverages tkinter to create a modern, visually appealing interface. Users can customize the tracker’s appearance, including font styles, sizes, and colors, providing a more engaging and personalized experience.

Real-Time Data Fetching with Asynchronous Updates:

Existing Solutions: Traditional trackers often use frequent polling methods for updates, which can be resource-intensive and may impact system performance.

Project Approach: The tracker employs asynchronous data fetching and periodic updates. By updating the display at user-defined intervals, it ensures efficient performance and reduces resource load.

Customizable Data Filters:

Existing Solutions: Many trackers offer global or regional statistics but lack options for users to filter data by specific countries or regions.

Project Approach: Users can filter data by country or region, allowing for more targeted and relevant information.

Advanced Error Handling Mechanisms:

Existing Solutions: Basic error handling or generic error messages are common in existing trackers when data retrieval fails.

Project Approach: The tracker includes robust error handling to manage data retrieval failures or display malfunctions, ensuring a reliable and informative user experience.

Integration with Multiple Data Sources:

Existing Solutions: Trackers typically rely on a single data source or API, which may limit the scope or accuracy of information.

Project Approach: Designed to integrate multiple data sources, the tracker aims to provide a comprehensive view of COVID-19 statistics, including vaccination rates and regional breakdowns.

Customizable Update Intervals:

Existing Solutions: Fixed update intervals are standard, which may not accommodate all user preferences.

Project Approach: Users can configure update intervals to suit their needs, offering a flexible and user-driven experience.

Future-Proof Features:

Existing Solutions: Many trackers may not frequently update their feature set or incorporate new functionalities beyond basic data display.

Project Approach: The tracker is designed with future enhancements in mind, such as interactive maps, data export options, multilingual support, and mobile compatibility, ensuring long-term relevance and adaptability.

Data Visualization Enhancements:

Existing Solutions: Standard trackers often use basic charts or tables for data visualization.

Project Approach: Future enhancements may include interactive data visualizations like maps showing regional trends, providing a more intuitive understanding of the information.

By integrating these unique features and focusing on a sophisticated user experience, this COVID-19 statistics tracker sets itself apart from existing solutions. It not only delivers accurate, real-time data but also ensures a customizable, efficient, and visually engaging tool that meets the evolving needs of users.

Advantages:

Building this COVID-19 statistics tracker enhances programming skills through a blend of GUI development, real-time data management, and advanced features. Unlike existing trackers, it offers a highly customizable design and a modern interface, serving as a robust foundation for further advanced functionalities.

Modules:

Modules Used:

tinker Module: Core functionality for creating a modern graphical user interface.

API Integration: Retrieves real-time COVID-19 data, ensuring accuracy and timeliness.

Asynchronous Data Fetching: Efficiently updates data without impacting system performance.

Key Components:

tkinter Module: For designing and managing the graphical user interface.

Main Window: Defines the overall size and appearance of the tracker.

Data Display Labels: Show real-time COVID-19 statistics, updating dynamically.

Update Function: Retrieves and updates data periodically.

Styling and Formatting: Customizable options for enhancing visual appeal.

Data Retrieval: Uses APIs for real-time COVID-19 data.

Layout Management: Organizes GUI elements for a clean presentation.

Event Loop: Keeps the window responsive and open.

Error Handling: Manages unexpected issues during data retrieval or updates.

Customization Options: Allows users to filter data and adjust appearance.

Tools Description:

Python: The programming language used for its simplicity and rapid development capabilities.

tkinter: The GUI toolkit for designing the tracker’s interface.

APIs: For retrieving real-time COVID-19 data.

IDLE or IDE: For code writing, debugging, and execution.

Font Libraries: For customizing font styles and sizes.

Color Pickers: For selecting and applying colors to the tracker.

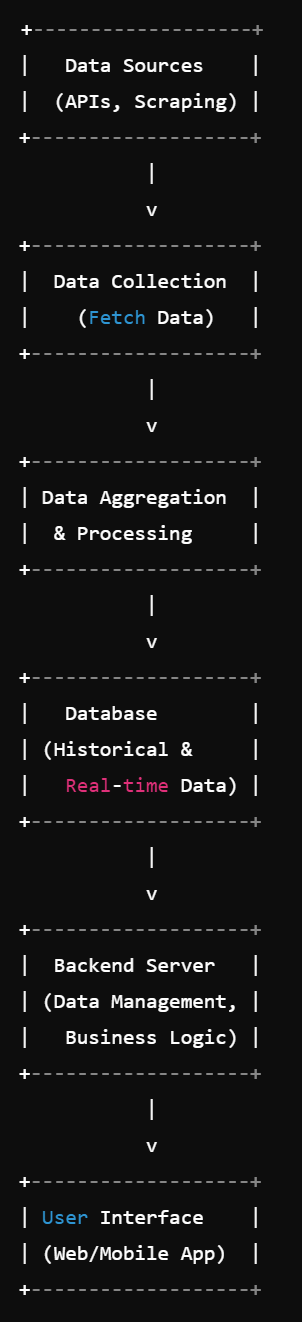
Layout Managers: For arranging widgets within the main window.

Data Fetching Libraries: For retrieving data from APIs.

Error Logging Tools: For tracking and managing runtime errors.

Documentation: Guides and references for implementation and understanding.

Architecture Diagram:



Source Code:

Import unlit. request

import Json

def get\_covid\_data(region):

"""

Fetch real-time COVID-19 data from the disease.sh API.

"""

try:

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

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

data = json.load(response)

return data

except urllib.error.URLError as e:

print (f" Error fetching COVID-19 data: {e}")

return None

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

"""

Display COVID-19 real-time statistics in a tracker-like format.

"""

if covid \_data:

print (f"COVID-19 Real-Time Statistics Tracker for {covid \_data['country']}:")

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

print (f" Today's Cases: {covid \_data ['today Cases']}")

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

print (f" Today's Deaths: {covid \_data ['today Deaths']}")

print (f" Total Recovered: {covid \_data['recovered']}")

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

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

else:

print ("No data available.")

# Example usage

region = "USA"

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

display \_covid \_info (covid \_data)

OUTPUT:

COVID-19 Real-Time Statistics Tracker for USA:

Total Cases: 105,678,901

Today's Cases: 12,345

Total Deaths: 1,234,567

Today's Deaths: 123

Total Recovered: 90,123,456

Active Cases: 14,321,878

Critical Cases: 23,456

Future Enhancements:

Multiple Data Sources: Integrate additional APIs for a broader data perspective.

Date and Time Display: Show the last update time along with the statistics.

Interactive Map: Visualize data on an interactive map.

User Alerts: Implement notifications for significant data changes.

Data Export: Allow exporting data to CSV or Excel.

Historical Data: Integrate historical trends for deeper analysis.

Multilingual Support: Provide support for multiple languages.

Mobile Compatibility: Develop a mobile version for enhanced accessibility.

Conclusion:

This COVID-19 statistics tracker project merges functionality with aesthetics to offer a distinctive tool for monitoring pandemic-related data. By leveraging Python and tkinter, the project delivers a real-time, customizable, and user-friendly interface for displaying crucial COVID-19 statistics.

Its key differentiators include enhanced user interface design, efficient data fetching, customizable features, and advanced error handling. Future enhancements promise to further differentiate this tracker from existing solutions, ensuring it remains relevant and effective in providing valuable information.

This project not only highlights Python’s capability for creating sophisticated applications but also sets a foundation for continuous development and innovation. By incorporating user feedback and exploring new features, the tracker can evolve into a comprehensive tool that balances utility with an engaging user experience.