**COVID CASES TRACKING APPLICATION**

*A*

*Mini Project Report*

*Submitted in partial fulfilment of the*

*Requirements for the award of the Degree of*

**BACHELOR OF ENGINEERING**

IN

**INFORMATION TECHNOLOGY**

By

**VANNOJ ABHINAV 1602-20-737-062**

**BITTU SREE VARSHITHA 1602-20-737-109**

**NANDELLI AKIRA NANDAN RAO 1602020-737-063**

****

**Department of Information Technology**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University)**

**Ibrahim Bagh, Hyderabad-31**

**2022**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University)**

**Hyderabad-500 031**

**Department of Information Technology**

****

**DECLARATION BY THE CANDIDATE**

We, **VANNOJ ABHINAV, BITTU SREE VARSHITHA** and **NANDELLI AKIRA NANDAN RAO,** bearing hall ticket numbers, **1602-20-737-062, 1602-20-737-109** and **1602-20-737-063,** hereby declare that the project report entitled “**COVID CASES TRACKING APPLICATION”** is submitted in partial fulfilment of the requirement for the award of the degree of **Bachelor of Engineering** in **Information Technology**

This is a record of bonafide work carried out by us and the results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

**VANNOJ ABHINAV**

**1602-20-737-062**

**BITTU SREE VARSHITHA**

**1602-20-737-109**

**NANDELLI AKIRA NANDAN RAO**

**1602020-737-063**

(Faculty In-Charge) (Head,Dept of IT)

**ACKNOWLEDGEMENT**

We feel very delighted to get this rare opportunity to show our profound senses of reverences and indebtedness to our esteemed teacher DRL Prasanna ma’am and our class coordinator Radha ma’am for advising us and introducing the project in a easy to understand way which has helped us complete our project easily and effectively on time.

We are dearly obliged for giving us an opportunity to work on this project which will surely prove to be a great asset in the future and has a greater scope and updates in coming dates.

Thank you.

**ABSTRACT**

This is a simple GUI based application which automatically generates the total number of covid cases either in any state of India or in any country both graphically and manual data. This application is user friendly, and the instructions would be clear in the application, the user must select the option to either view covid cases in India or in any other country. Then he would be given an option to view data in the form of graphs or just the manual data.

In this application we have implemented GUI using tkinter library and used other libraries like matplotlib to depict data graphically, BeautifulSoup to scrape the data from the API’s, requests library to request data from the URL’s and pandas library to manipulate data and to create data frames. This application is easy to understand and user friendly.

The main aim of this project is to provide information regarding the rising covid cases across the globe, so that the people will be aware of how the situation outside is. So, with this project we would be gaining knowledge regarding the implementation of different libraries such as tkinter, BeautifulSoap, pandas, requests and matplotlib.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **CONTENTS** | **Page No.** |
| **1** | **Introduction** | **6** |
| **2** | **Technology** | **8** |
| **3** | **Design** | **9** |
| **4** | **Implementation** | **13** |
| **5** | **Testing** | **19** |
| **6** | **Additional Knowledge acquired** | **23** |
| **7** | **Conclusion and Future work** | **24** |
| **8** | **References** | **24** |

**INTRODUCTION**

Covid Cases Tracking Application’s main aim is to provide information regarding the rising Covid cases across the globe. Stay-Up-To-Date with valuable coronavirus (Covid-19) data Summarized by choosing any country. This is a GUI application based on the concept of Web Scraping. This application is easily understandable and has an interactive user interface.

This application’s main purpose is to provide the number of covid cases, deaths, and many others in a particular state in India and any country across the world.

Our application implements Tkinter, BeautifulSoup, pandas, requests, Matplotlib, and many more modules. In our application, there are mainly two activities that can be majorly classified into User actions and admin actions. These can be further sub-classified as their specific actions that are done and are specific to each.

We have collected data by scraping the required data from the websites (Provided in the Reference section) by using the library BeautifulSoup and extracting the required information to output to the user. We have requested the data from the URL using the requests library.

The user has to choose the location at which he/she wants to view the covid-19 details. By just providing these details admin will display the Total number of cases, deaths, Recovered, and Active cases. We also provide details of several other data such as serious/critical cases, Total tests, population.

When the user opens our application, they have to choose any one option out of two options. The two options are India and International, that is admin will provide coronavirus updates according to the country the user has chosen.

If the user chooses India, then the user has to enter the state name in which he/she wants the coronavirus updates. Then the user has to choose one more option that is whether he/she wants the data in a graphical form or normal text form. If the user chooses graphical form then for the country India and the particular state user entered our application will show the coronavirus (Covid-19) details in a graphical form(Bar graph) Using matplotlib from the website worldometers and covid-19india.

If the user wants the data in normal text form, then a window will pop on the user’s display containing the Total number of active cases, confirmed cases, deaths, and recovered.

If the User chooses the International option, then the application asks the user to enter the country name other than India to show the updates of covid-19 across the world. After entering the country name again, the user has to choose any one option that is either the data has to be viewed in a graphical way or a normal text form.

If the User chooses the “Get data Graphically” option then the user will get the bar graph showing the covid-19 details which include the Total number of cases, deaths, Recovered, Active cases number of serious/critical cases, Total tests, population, the total number of cases per 1M Population, Tests per 1M Population.

If the user chooses the “Show data” option then a window will be opened on the Users screen showing the Covid-19 details of that particular country which includes the Total number of cases, deaths, Recovered, Active cases number of serious/critical cases, Total tests, population, the total number of cases per 1M Population, Tests per 1M Population.

If the user enters an invalid Country name or does not enter the country or state name with first letter capital then a window with no details will be displayed on the user’s screen.

We have implemented the GUI part of this application using the Tkinter library. Designing the windows was made easier using the canvas widgets of Tkinter. We have also attached the relevant background images to the widows to make the application more interactive and make the application look more attractive.

**TECHNOLOGY**

**Hardware Requirements:**

* 512 MB RAM
* 2GB HDD
* CORE i5

**Software Requirements:**

* Windows XP/Windows 2000
* PYTHON Interpreter

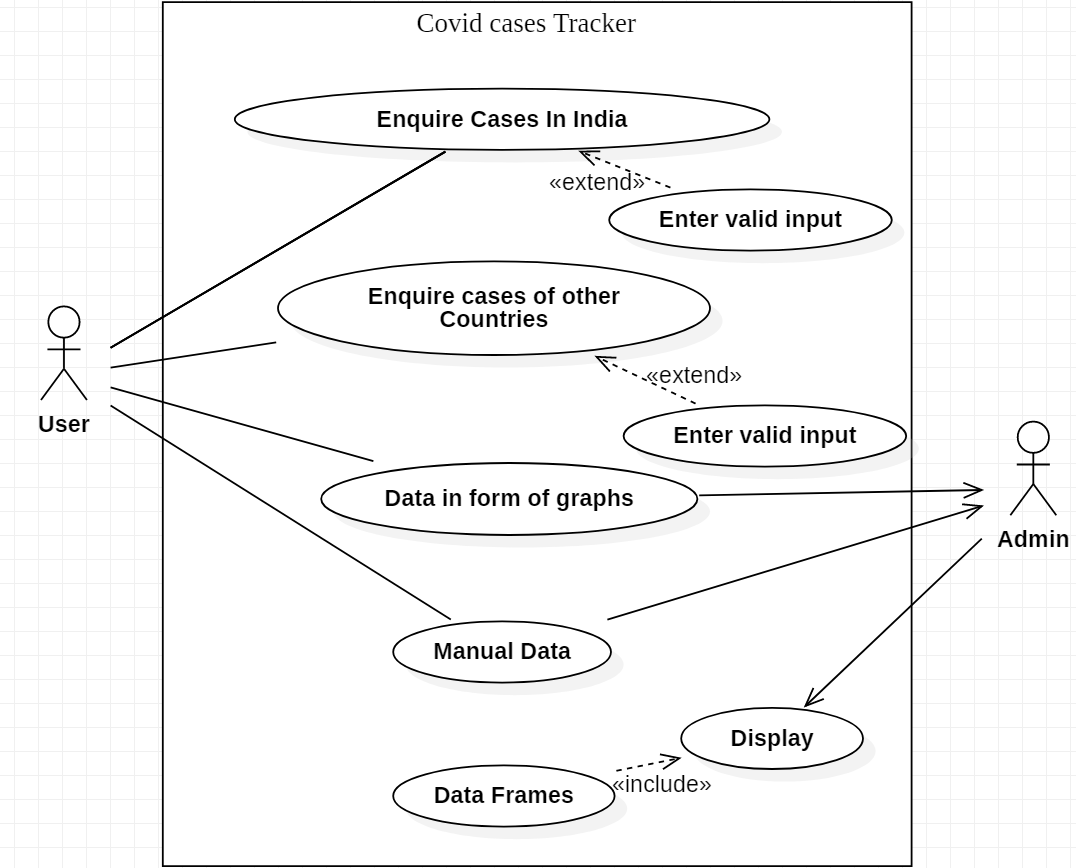
**Packages used:**

* tkinter
* BeautifulSoup
* matplotlib
* pandas
* requests

**PROPOSED WORK**

**DESIGN**: -

USE CASE DIAGRAM: -



|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID**:TC01 | | **Use Case ID**:UC01 | |
| **Test Case Title**: Enquire cases | |  | |
| **Test Case Description:** Gives user two options to choose, either to view cases in India or any other country. | |
| **Test Steps** | **Expected Result** | | **Actual Result** |
| Choose one option | A new window is popped. | | A new window regarding the user’s choice is popped. |

USE CASE DESCRIPTIONS**:**

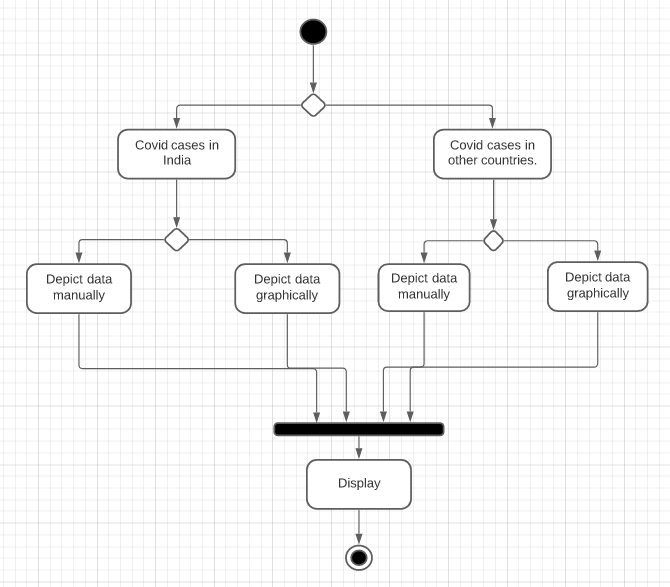
|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID**:TC02 | | **Use Case ID**:UC02 | |
| **Test Case Title**: Enquire cases in a particular state manually | |  | |
| **Test Case Description:** Gives user 2 options to display data either manually or graphically. | |
| **Test Steps** | **Expected Result** | | **Actual Result** |
| Choose depict data manually | Data will be depicted manually. | | Data is depicted manually. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID**:TC03 | | **Use Case ID**:UC03 | |
| **Test Case Title**: Enquire cases in a particular state graphically | |  | |
| **Test Case Description:** Gives user 2 options to display data either graphically. | |
| **Test Steps** | **Expected Result** | | **Actual Result** |
| Choose depict data graphically | Data will be depicted. | | Data is depicted graphically. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID**:TC04 | | **Use Case ID**:UC04 | |
| **Test Case Title**: Enquire cases in a particular country manually | |  | |
| **Test Case Description:** Gives user 2 options to display data either manually or graphically. | |
| **Test Steps** | **Expected Result** | | **Actual Result** |
| Choose depict data manually | Data will be depicted. | | Data is depicted manually. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID**:TC05 | | **Use Case ID**:UC05 | |
| **Test Case Title**: Enquire cases in a particular country graphically. | |  | |
| **Test Case Description:** Gives user 2 options to display data either manually or graphically. | |
| **Test Steps** | **Expected Result** | | **Actual Result** |
| Choose depict data graphically | Data will be depicted. | | Data is depicted graphically. |

ACTIVITY DIAGRAM:



**IMPLEMENTATION**

**Source Code**: -

**covid\_tracker.py:**

# importing modules

from bs4 import BeautifulSoup

import pandas as pd

import matplotlib.pyplot as plt

from tkinter import \*

import requests

from tkinter import ttk

# storing the url in the form of string

root = Tk()

# setting geometry

root.geometry("700x400")

# setting title

root.title("Get Covid-19 Data Country Wise")

url = "https://api.covid19india.org/state\_district\_wise.json"

url1 = 'https://www.worldometers.info/coronavirus/'

# Create object page

page = requests.get(url1)

soup = BeautifulSoup(page.text, 'lxml')

table1 = soup.find('table', id='main\_table\_countries\_today')

headers = []

for i in table1.find\_all('th'):

title = i.text

headers.append(title)

headers[13] = 'Tests/1M pop'

mydata = pd.DataFrame(columns=headers)

for j in table1.find\_all('tr')[1:]:

row\_data = j.find\_all('td')

row = [i.text for i in row\_data]

length = len(mydata)

mydata.loc[length] = row

# Drop and clearing unnecessary rows

mydata.drop(mydata.index[0:7], inplace=True)

mydata.drop(mydata.index[222:229], inplace=True)

mydata.reset\_index(inplace=True, drop=True)

# Drop “#” column

mydata.drop('#', inplace=True, axis=1)

# Export to csv

mydata.to\_csv('covid\_data.csv', index=False)

# Try to read csv

mydata2 = pd.read\_csv('covid\_data.csv')

def case1():

global case\_1

case\_1 = Toplevel(root)

case\_1.title("India")

case\_1.geometry("612x408")

bg1 = PhotoImage(file="istockphoto-1209432716-612x612.png")

my\_canvas = Canvas(case\_1, width=612, height=408)

my\_canvas.pack()

my\_canvas.create\_image(0, 0, image=bg1, anchor="nw")

state = Label(my\_canvas, text="Covid cases in India", fg="black", bg="LightBlue1")

state.config(font='calibri 30')

state.place(relx=0.22, rely=0.1)

ulabel = Label(my\_canvas, text="Enter State name:", bg="LightBlue1")

ulabel.place(relx=0.4, rely=0.27)

ulabel.config(font="calibri 12")

user = Entry(my\_canvas, bg='#d3d3d3', fg='black', textvariable=data)

user.config(width=42)

user.place(relx=0.3, rely=0.35)

ttk.Button(my\_canvas, text="Get Data Graphically", command=casesDataGraph).place(x=250, y=170)

ttk.Button(my\_canvas, text="Show Data", command=casesData).place(x=270, y=200)

my\_canvas.pack()

my\_canvas.mainloop()

def case2():

case\_2 = Toplevel(root)

case\_2.title("Internation")

case\_2.geometry("600x400")

bg1 = PhotoImage(file="shutterstock\_1647268288 (2).png")

my\_canvas1 = Canvas(case\_2, width=600, height=400)

my\_canvas1.pack()

my\_canvas1.create\_image(0, 0, image=bg1, anchor="nw")

country = Label(my\_canvas1, text="Covid-19 cases across the world", fg="black", bg="gold")

country.config(font='calibri 20')

country.place(relx=0.2, rely=0.15)

ulabel = Label(my\_canvas1, text="Enter Country name:", bg="gold")

ulabel.config(font='calibri 12')

ulabel.place(relx=0.39, rely=0.27)

user = Entry(my\_canvas1, bg='#d3d3d3', fg='black', textvariable=data1)

user.config(width=42)

user.place(relx=0.28, rely=0.35)

ttk.Button(my\_canvas1, text="Get Data Graphically", command=casesDataInternationGraph).place(x=230, y=180)

ttk.Button(my\_canvas1, text="Show Data", command=casesDataInternation).place(x=250, y=220)

my\_canvas1.pack()

my\_canvas1.mainloop()

def casesDataInternation():

y = data1.get()

lst = []

flag = False

for index, row in mydata2.iterrows():

if row[0] == y:

flag = True

lst = row.tolist()

if flag:

newWindow = Toplevel(root)

newWindow.title("Show Data")

newWindow.geometry("400x400")

login\_canvas = Canvas(newWindow, width=720, height=440, bg="PeachPuff2")

login\_canvas.pack()

login\_frame = Frame(login\_canvas, bg="LightBlue1")

login\_frame.place(relwidth=0.8, relheight=0.8, relx=0.1, rely=0.1)

print(lst)

lst2 = list(range(10))

lst1 = [int(x.replace(',', '')) for x in lst[1:14] if str(x) != 'nan']

Label(login\_frame, text="TotalCases: {0}\nTotalDeaths: {1}\nTotalRecovered: {2}\nActiveCases: {3}\nSerious/Critical: {4}\nTotCases/1M pop: {5}\n Deaths/1M pop: {6}\nTotalTests: {7}\nTests/1Mpop: {8}\nPopulation: {9}\n".format(lst1[0], lst1[1], lst1[2], lst1[3], lst1[4], lst1[5], lst1[6], lst1[7], lst1[8], lst1[9]), font="TimesNewRoman 15", fg='black', bg='LightBlue1').pack(side=LEFT)

def casesDataInternationGraph():

# getting the json data by calling api

y = data1.get()

lst = []

flag = False

for index, row in mydata2.iterrows():

if row[0] == y:

flag = True

lst = row.tolist()

if flag:

lst2 = list(range(10))

lst1 = [int(x.replace(',', '')) for x in lst[1:14] if str(x) != 'nan']

tick\_label = ['TotalCases', 'TotalDeaths', 'TotalRecovered', 'ActiveCases', "Serious,Critical", 'TotCases/1M pop',

'Deaths/1M pop', 'TotalTests', 'Tests/1Mpop', 'Population']

plt.bar(lst2, lst1, color='green', tick\_label=tick\_label, width=0.7)

plt.xlabel('x - axis')

# naming the y-axis

plt.ylim(0, 7 \*\* 10)

plt.ylabel('y - axis')

# plot title

plt.title('Covid Data')

plt.show()

def casesData():

dataa = ((requests.get(url)).json())

states = []

x = data.get()

# getting states

for key in dataa.items():

states.append(key[0])

# getting statewise data

for state in states:

if state == x :

active, confirmed, deaths, recovered = 0, 0, 0, 0

newWindow = Toplevel(root)

newWindow.title("Show Data")

newWindow.geometry("280x280")

my\_canvas = Canvas(newWindow, width=520, height=440, bg="PeachPuff2")

my\_canvas.pack()

login\_frame = Frame(my\_canvas, bg="LightBlue1")

login\_frame.place(relwidth=0.8, relheight=0.8, relx=0.1, rely=0.1)

f = (dataa[state]['districtData'])

tc = []

dis = []

act, con, dea, rec = 0, 0, 0, 0

# getting districtwise data

for key in (dataa[state]['districtData']).items():

district = key[0]

dis.append(district)

active = dataa[state]['districtData'][district]['active']

confirmed = dataa[state]['districtData'][district]['confirmed']

deaths = dataa[state]['districtData'][district]['deceased']

recovered = dataa[state]['districtData'][district]['recovered']

if district == 'Unknown':

active, confirmed, deaths, recovered = 0, 0, 0, 0

tc.append([active, confirmed, deaths, recovered])

act = act + active

con = con + confirmed

dea = dea + deaths

rec = rec + recovered

tc.append([act, con, dea, rec])

dis.append('Total')

Label(login\_frame, text="Active: {}\nConfirmed: {}\nDeaths: {}\nRecovered{}\n".format(act, con, dea, rec), font="TimesNewRoman 15", fg='black', bg='LightBlue1').pack(side=LEFT)

def casesDataGraph():

dataa = ((requests.get(url)).json())

states = []

x = data.get()

root.update()

# getting states

for key in dataa.items():

states.append(key[0])

# getting statewise data

for state in states:

if state == x:

f = (dataa[state]['districtData'])

tc = []

dis = []

act, con, dea, rec = 0, 0, 0, 0

# getting districtwise data

for key in (dataa[state]['districtData']).items():

district = key[0]

dis.append(district)

active = dataa[state]['districtData'][district]['active']

confirmed = dataa[state]['districtData'][district]['confirmed']

deaths = dataa[state]['districtData'][district]['deceased']

recovered = dataa[state]['districtData'][district]['recovered']

if district == 'Unknown':

active, confirmed, deaths, recovered = 0, 0, 0, 0

tc.append([active, confirmed, deaths, recovered])

act = act + active

con = con + confirmed

dea = dea + deaths

rec = rec + recovered

tc.append([act, con, dea, rec])

dis.append('Total')

parameters = ['Active', 'Confirmed', 'Deaths', 'Recovered']

# creating a dataframe

df = pd.DataFrame(tc, dis, parameters)

print('COVID - 19', state, 'District Wise Data')

print(df)

# plotting of data

plt.bar(dis, df['Active'], width=0.5, align='center')

fig = plt.gcf()

fig.set\_size\_inches(18.5, 10.5)

plt.xticks(rotation=75)

plt.show()

# Driver Code

data = StringVar()

data1 = StringVar()

bg = PhotoImage(file="istockphoto-1213090148-170667a (.png")

my\_canvas = Canvas(root, width=700, height=400)

my\_canvas.pack(fill='both', expand=True)

my\_canvas.create\_image(0, 0, image=bg, anchor="nw")

heading = Label(my\_canvas, text="Covid-19 Cases Tracker", fg="black", bg="khaki1")

heading.config(font='TimesNewRoman 27')

heading.place(relx=0.22, rely=0.1)

ulabel = Label(my\_canvas, text="Select one option", fg='black', bg='khaki1')

ulabel.config(font='TimesNewRoman 18')

ulabel.place(relx=0.36, rely=0.3)

ttk.Button(my\_canvas, text="India", command=case1, width=20).place(x=285, y=200)

ttk.Button(my\_canvas, text="International", command=case2, width=20).place(x=285, y=290)

my\_canvas.pack()

root.mainloop()

**Additional files:**

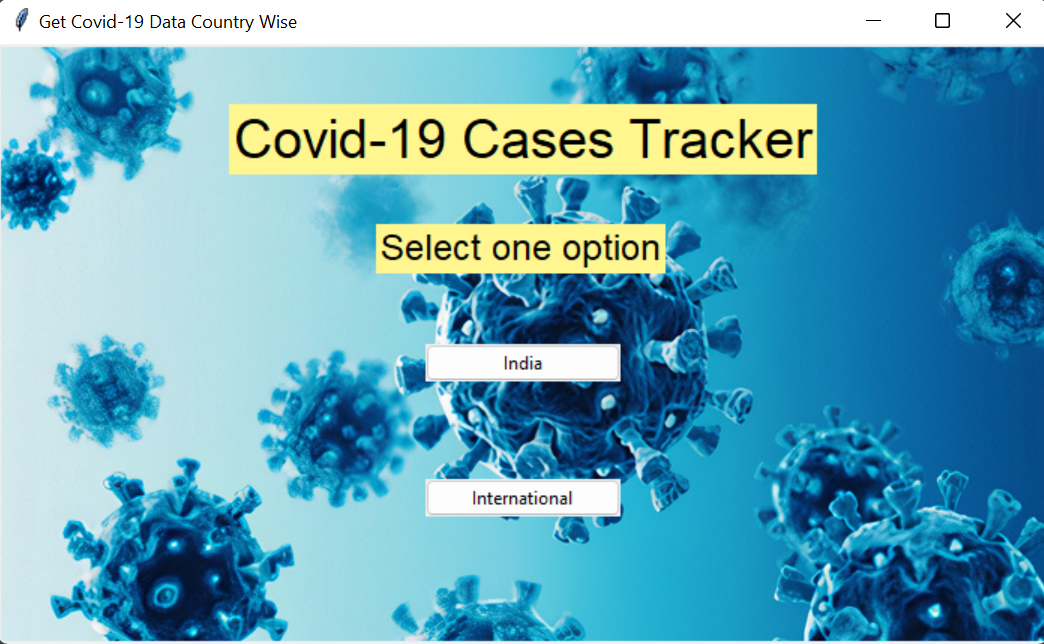
**GitHub Link: -**

<https://github.com/Abhinav-1903/Covid-19-Cases-tracking-Application>

**TESTING**

**Screen Shots of Application Test Cases:**

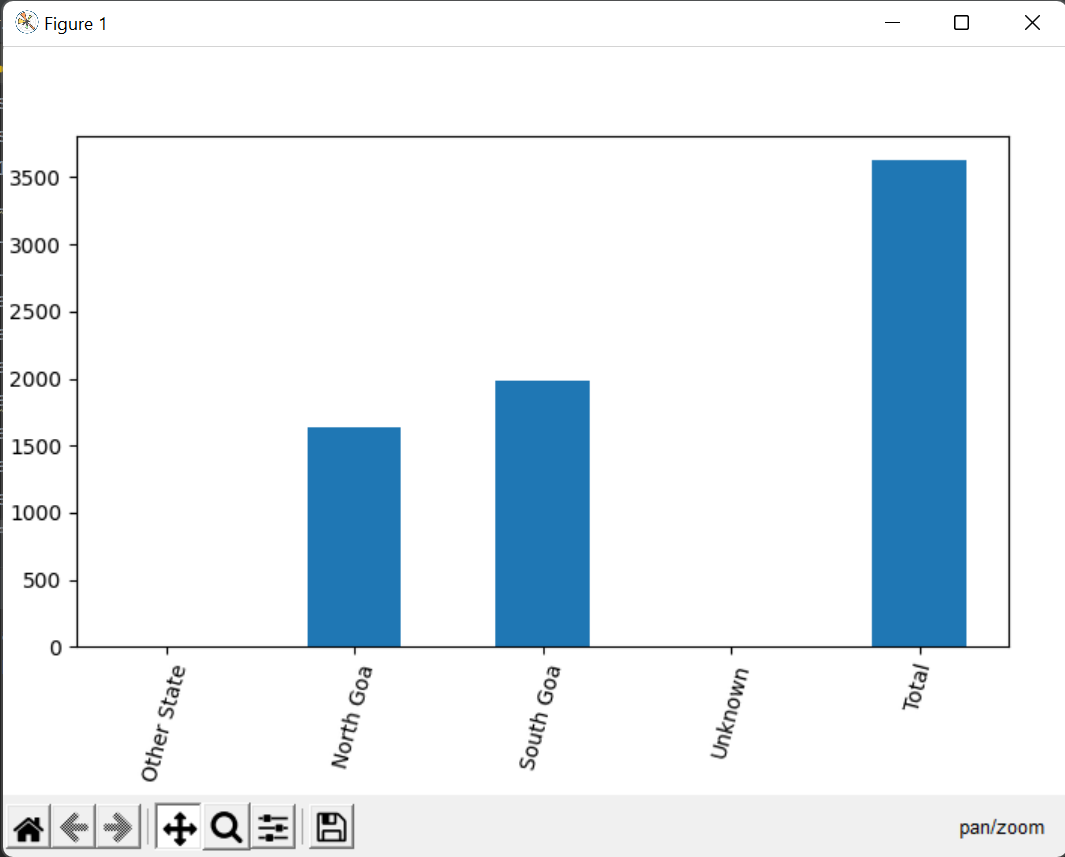
Homepage:

****

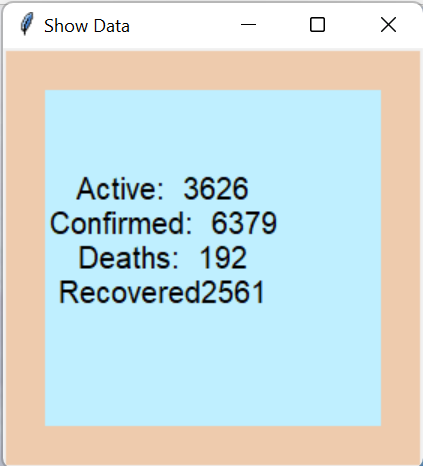
Upon choosing “India” option:

****

Upon choosing “Get Data Graphically” option for the state “Goa”:

**f 7**

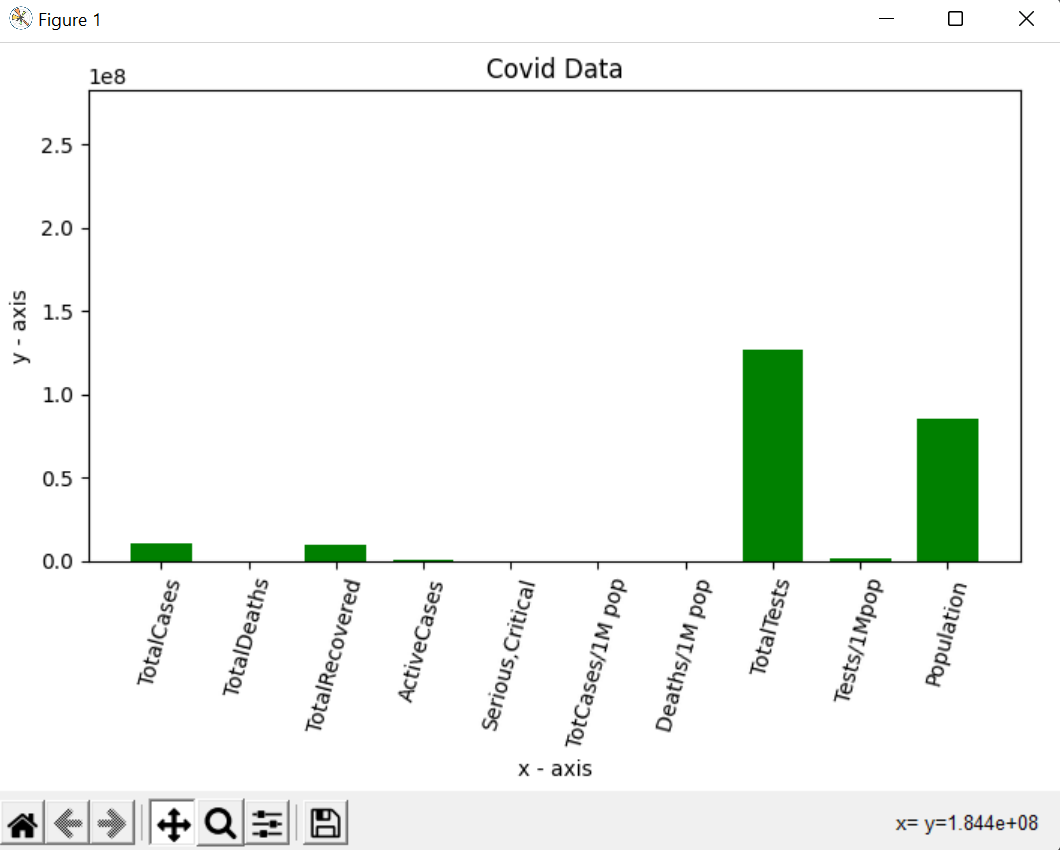
Upon choosing “Show Data” option for the same state “Goa”:

****

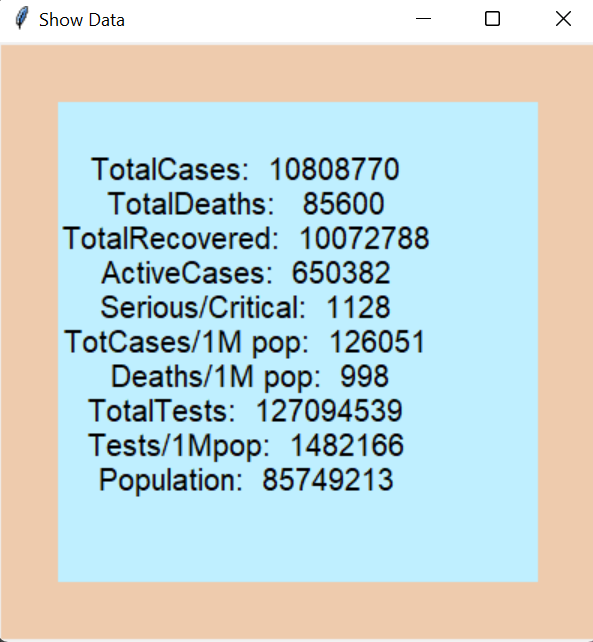
Upon choosing “International” option from the Home page:

****

Upon choosing “Get Data Graphically” for the country “Turkey”:

****

Upon choosing “Show Data” option for the same country “Turkey”:

****

**ADDITIONAL KNOWLEDGE ACQUIRED**

We have gained knowledge regarding many topics which weren’t the part of the curriculum until B.E - IInd Semester such as:

* Implementing GUI using tkinter library.
* Web Scraping using Beautiful Soup.
* Creating Data Frames using pandas.
* Requesting data from an URL using requests.
* Designing Use Case Diagram and Activity Diagram

**CONCLUSION & FUTURE WORK**

Covid-19 cases tracker is an informative GUI application which is based on Web Scraping. In future, we would like to make the application livelier by providing the required data using Django and Machine Learning, we would also try to make the application available for all platforms and depict data in more graphical manner using other plotting methods.

**REFERENCES**

* Python Programming - Using Problem Solving Approach First Edition by Reema Thareja
* <https://www.geeksforgeeks.org/python-tkinter-canvas-widget/>
* <https://www.geeksforgeeks.org/implementing-web-scraping-python-beautiful-soup/>
* <https://www.geeksforgeeks.org/pythonpandasdataframe/#:~:text=Pandas%20DataFrame%20is%20two%2Ddimensional,fashion%20in%20rows%20and%20columns.>
* <https://www.worldometers.info/coronavirus/#countries>