COVID-19 IMPACT ANALYSIS

app.py file

import numpy as np

import pandas as pd

import plotly.graph\_objs as go

import dash

import dash\_core\_components as dcc

import dash\_html\_components as html

from dash.dependencies import Input, Output

import plotly.express as px

external\_stylesheet= [

    {

        'href': 'https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min.css',

        'rel': 'stylesheet',

        'integrity': 'sha384Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh',

        'crossorigin': 'anonymous'

     }

]

#patients=pd.read\_csv('state\_wise\_daily.csv')

patients=pd.read\_csv('C:/Users/Afifa/Desktop/IHHPET notes/Covid projrct/corona virus pandemic/state\_wise\_daily.csv')

total=patients.shape[0]

active=patients[patients['Status']=='Confirmed'].shape[0]

recovered=patients[patients['Status']=='Recovered'].shape[0]

deceased=patients[patients['Status']=='Deceased'].shape[0]

options1=[

    {'label':'All','value':'All'},

    {'label':'Hospitalized','value':'Hospitalized'},

    {'label':'Recovered','value':'Recovered'},

    {'label':'Deceased','value':'Deceased'}

    ]

options2=[

    {'label':'All','value':'All'},

    {'label':'Mask','value':'Mask'},

    {'label':'Sanitizer','value':'Sanitizer'},

    {'label':'Oxygen','value':'Oxygen'},

    ]

options3=[

      {'label':'Red Zone', 'value':'Red Zone'},

      {'label':'Blue Zone','value':'Blue Zone'},

      {'label':'Green Zone','value':'Green Zone'},

      {'label':'Orange Zone','value':'Orange Zone'}

    ]

 # to host the website in local server

app = dash.Dash(\_\_name\_\_,external\_stylesheets=external\_stylesheet)

# to make app layout: container class

app.layout=html.Div([

        html.H1('Corona Virus Pandamic',style={'color':'#33FFD4 ', 'text-align':'center'}),

        # Row 1- 4 columns(cards)

        html.Div([

            html.Div([

                html.Div([

                    html.Div([

                        html.H3("Total Cases",className='text-light'),

                        html.H4(total,className='text-light')

                        ],className='card-body')

                    ],className='card bg-danger')

                ],className='col-md-3'),

            html.Div([

                html.Div([

                    html.Div([

                        html.H3("Active Cases",className='text-light'),

                        html.H4(active,className='text-light')

                        ],className='card-body')    #card-body

                    ],className='card bg-info')     #card with bg colour

                ],className='col-md-3'),

            html.Div([

                html.Div([

                    html.Div([

                        html.H3("Recovered Cases",className='text-light'),

                        html.H4(recovered,className='text-light')

                        ],className='card-body')         #card-body

                    ],className='card bg-warning')       #card with bg colour

                ],className='col-md-3'),

            html.Div([

                html.Div([

                    html.Div([

                        html.H3("Total Deaths",className='text-light'),

                        html.H4(deceased,className='text-light')

                        ],className='card-body')

                    ],className='card bg-success')

                ],className='col-md-3'),

            ],className='row'),

        # Row 2

        html.Div([

            html.Div([

                html.Div([

                    html.Div([

                        dcc.Dropdown(id='plot-graph', options=options2, value='All'),

                        dcc.Graph(id='graph')

                        ],className='card-body')

                    ],className='card bg-success')

                ],className='col-md-6'),

            html.Div([

                html.Div([

                    html.Div([

                        dcc.Dropdown(id='my\_dropdown', options=options3, value='Status'),

                        dcc.Graph(id='the\_graph')

                        ],className='card-body')

                    ],className='card bg-info')

                ],className='col-md-6'),

            ],className='row'),

        # Row 3

        html.Div([

            html.Div([

                html.Div([

                    html.Div([

                        # for graphs we use dcc(dash core components)

                        dcc.Dropdown(id='picker',options=options1, value='All'),

                        dcc.Graph(id='bar')

                        ],className='card-body')

                    ],className='card bg-warning')

                ],className='col-md-12')

            ],className='row'),

    ],className='container',style={'background-color': '#000000 '})

# for row 3

@app.callback(Output('bar','figure'),[Input('picker','value')])

def update\_graph(type):

    if type=='All':

        return {'data':[go.Bar(x=patients['State'],y=patients['Total'])],

                'layout':go.Layout(title='State Total Count',plot\_bgcolor='orange')

                }

    if type=='Hospitalized':

        return{'data':[go.Bar(x=patients['State'],y=patients['Hospitalized'])],'layout':go.Layout(title='State Total Count',plot\_bgcolor='orange')

                }

    if type=='Recovered':

        return {'data':[go.Bar(x=patients['State'],y=patients['Recovered'])],

                'layout':go.Layout(title='State Total Count',plot\_bgcolor='orange')

                }

    if type=='Deceased':

        return {'data':[go.Bar(x=patients['State'],y=patients['Deceased'])],

                'layout':go.Layout(title='State Total Count',plot\_bgcolor='orange')

                }

# For Row 2(1)

@app.callback(Output('graph','figure'),[Input('plot-graph','value')])

def generate\_graph(type):

    if type=='All':

        return {'data':[go.Line(x=patients['Status'], y=patients['Total'])],

                'layout':go.Layout(title= "Commodities Total Count",plot\_bgcolor='pink')}

    if type=='Mask':

        return {'data':[go.Line(x=patients['Status'],  y=patients['Mask'])],

                'layout':go.Layout(title= "Commodities Total Count",plot\_bgcolor='pink')}

    if type=='Sanitizer':

        return {'data':[go.Line(x=patients['Status'],  y=patients['Sanitizer'])],

                'layout':go.Layout(title= "Commodities Total Count",plot\_bgcolor='pink')}

    if type=='Oxygen':

        return {'data':[go.Line(x=patients['Status'], y=patients['Oxygen'])],

                'layout':go.Layout(title= "Commodities Total Count",plot\_bgcolor='pink')}

# For row 2(2)

@app.callback(Output('the\_graph','figure'),[Input('my\_dropdown','value')])

def generate\_graph(my\_dropdown):

    piechart = px.pie(data\_frame=patients,names= my\_dropdown, hole=0.3)

    return (piechart)

if \_\_name\_\_ == '\_\_main\_\_':

    # to get a local host from the class

    app.run\_server(debug = True)

dataset.py file

import pandas as pd

import numpy as np

import dash

import dash\_core\_components as dcc

import dash\_html\_components as html

from dash.dependencies import Input, Output

from dash import Dash, html

app = Dash(\_\_name\_\_)

app.layout = html.Div([

    html.Div(children='Hello World')

])

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)