Covid_19\First.py

```python
import numpy as np
import pandas as pd
import plotly.graph_objs as go
import dash
from dash import dcc
from dash import html
from dash.dependencies import Input, Output
import plotly.express as px

external_stylesheet = [
    {
        'href': 'https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css',
        'rel': 'stylesheet',
        'integrity': 'sha384 Vkoo8x4CGsO3 + Hhxv8T / Q5PaXtkKtu6ug5TOeNV6-gBiFeWPGFN9MuhOf23Q9Ifjh',
        'crossorigin': 'anonymous'
    }
]

data = pd.read_csv("state_wise_daily data file IHHPET.csv")
Total = data.shape[0]
Active = data[data['Status']=='Confirmed'].shape[0]
recovered = data[data['Status']=='Recovered'].shape[0]
deaths = data[data['Status']=='Deceased'].shape[0]

options = [
    {'label': 'All', 'value': 'All'},
    {'label': 'Hospitalized', 'value': 'Hospitalized'},
    {'label': 'Recovered', 'value': 'Recovered'},
    {'label': 'Deceased', 'value': 'Deceased'}
]

options1 = [
    {'label': 'All', 'value': 'All'},
    {'label': 'Mask', 'value': 'Mask'},
    {'label': 'Sanitizer', 'value': 'Sanitizer'},
    {'label': 'Oxygen', 'value': 'Oxygen'}
]

options2 = [
    {'label': 'All', 'value': 'Status'},
    {'label': 'Red Zone', 'value': 'Red Zone'},
    {'label': 'Blue Zone', 'value': 'Blue Zone'},
    {'label': 'Green Zone', 'value': 'Green Zone'}
    # {'label': 'Orange Zone', 'value': 'Orange Zone'}
]

app = dash.Dash(__name__, external_stylesheets=external_stylesheet)
```

```

app.layout = html.Div([

    html.H1("cororna virus pandemic year 2020"),
    html.Div([
        html.Div([
            html.Div([
                html.Div([
                    html.H3("Total Cases", className='text-light'),
                    html.H4(" = 1791", 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(" = 597", className="text-light")
                ], className='card-body')
            ], className='card bg-info')
        ], className='col-md-3'),

        html.Div([
            html.Div([
                html.Div([
                    html.H3("Recovered Cases", className='text-light'),
                    html.H4(" = 597", className="text-light")
                ], className='card-body')
            ], className='card bg-success')
        ], className='col-md-3'),

        html.Div([
            html.Div([
                html.Div([
                    html.H3("Total Deaths", className='text-light'),
                    html.H4(" = 597", className="text-light")
                ], className='card-body')
            ], className='card bg-warning')
        ], className='col-md-3')
    ], className='row'),

    html.Div([
        html.Div([
            html.Div([
                html.Div([
                    dcc.Dropdown(id='plot-graph', options=options1, value='All'),
                    dcc.Graph(id='graph')
                ], className='card-body')
            ], className='card bg-info')
```

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

        html.Div([
            html.Div([
                html.Div([
                    dcc.Dropdown(id='my_dropdown', options=options2, value='Status',style={"width": "100%","fontSize": "25px"}),
                    dcc.Graph(id='the_graph')
                ], className='card-body')
            ], className='card bg-danger')
        ], className='col-md-6')
    ], className='row'),

    html.Div([
        html.Div([
            html.Div([
                html.Div([
                    dcc.Dropdown(id='picker', options=options, value='All'),
                    dcc.Graph(id='bar')
                ], className='card-body')
            ], className='card bg-warning')
        ], className='col-md-12')
    ], className='row')
], className='container')


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

    if type == 'All':
        return {'data': [go.Bar(x=data['State'], y=data['Total'])],'layout': go.Layout(title="State Total Count : ", plot_bgcolor='orange')}

    if type == 'Hospitalized':
        return {'data': [go.Bar(x=data['State'], y=data['Total'])],'layout': go.Layout(title="Hospitalized", plot_bgcolor='orange')}

    if type == 'Recovered':
        return {'data': [go.Bar(x=data['State'], y=data['Total'])],'layout': go.Layout(title="Recovered", plot_bgcolor='orange')}

    if type == 'Deceased':
        return {'data': [go.Bar(x=data['State'], y=data['Total'])],'layout': go.Layout(title="Deceased", plot_bgcolor='orange')}



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

    if type == 'All':
        return {'data': [go.Line(x=data['Status'], y=data['Total'])],'layout': go.Layout(title="Commodities Total Count", plot_bgcolor='pink')}

    if type == 'Mask':
        return {'data': [go.Line(x=data['Status'], y=data['Total'])], 'layout': go.Layout(title="Mask", plot_bgcolor='pink')}

    if type == 'Sanitizer':
        return {'data': [go.Line(x=data['Status'], y=data['Total'])], 'layout': go.Layout(title="Sanitizer", plot_bgcolor='pink')}

    if type == 'Oxygen':
        return {'data': [go.Line(x=data['Status'], y=data['Total'])],'layout': go.Layout(title="Oxygen", plot_bgcolor='pink')}



@app.callback(Output('the_graph','figure'),[Input('my_dropdown','value')])
def generate_graph(my_dropdown):

    chart = px.pie(data_frame=data,names= my_dropdown, hole=0.4)
    return(chart)

if __name__ == '__main__':
    app.run(debug=True)
```