import pandas as pd

import dash

import dash\_html\_components as html

import dash\_core\_components as dcc

from dash.dependencies import Input, Output

import plotly.express as px

# Read the SpaceX data into pandas dataframe

spacex\_df = pd.read\_csv("spacex\_launch\_dash.csv")

max\_payload = spacex\_df['Payload Mass (kg)'].max()

min\_payload = spacex\_df['Payload Mass (kg)'].min()

# Create a Dash application

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

# Create an app layout

app.layout = html.Div(children=[

    html.H1('SpaceX Launch Records Dashboard',

            style={'textAlign': 'center', 'color': '#503D36', 'font-size': 40}),

    # TASK 1: Add a dropdown list to enable Launch Site selection

    dcc.Dropdown(

        id='site-dropdown',

        options=[

            {'label': 'All Sites', 'value': 'ALL'},

            {'label': 'CCAFS LC-40', 'value': 'CCAFS LC-40'},

            {'label': 'VAFB SLC-4E', 'value': 'VAFB SLC-4E'},

            {'label': 'KSC LC-39A', 'value': 'KSC LC-39A'},

            {'label': 'CCAFS SLC-40', 'value': 'CCAFS SLC-40'}

        ],

        value='ALL',

        placeholder='Select a Launch Site',

        searchable=True

    ),

    html.Br(),

    # TASK 2: Add a pie chart to show the total successful launches count for all sites

    html.Div(dcc.Graph(id='success-pie-chart')),

    html.Br(),

    html.P("Payload range (Kg):"),

    # TASK 3: Add a slider to select payload range

    dcc.RangeSlider(

        id='payload-slider',

        min=min\_payload,

        max=max\_payload,

        step=1000,

        marks={int(i): str(i) for i in range(int(min\_payload), int(max\_payload) + 1, 1000)},

        value=[min\_payload, max\_payload]

    ),

    html.Br(),

    # TASK 4: Add a scatter chart to show the correlation between payload and launch success

    html.Div(dcc.Graph(id='success-payload-scatter-chart')),

])

# TASK 2: Callback for `site-dropdown` as input, `success-pie-chart` as output

@app.callback(

    Output('success-pie-chart', 'figure'),

    [Input('site-dropdown', 'value')]

)

def update\_pie\_chart(selected\_site):

    if selected\_site == 'ALL':

        fig = px.pie(spacex\_df, values='class',

                     names='Launch Site',

                     title='Total Success Launches By Site')

    else:

        filtered\_df = spacex\_df[spacex\_df['Launch Site'] == selected\_site]

        fig = px.pie(filtered\_df, names='class',

                     title=f'Total Success Launches for site {selected\_site}')

    return fig

# TASK 4: Callback for `site-dropdown` and `payload-slider` as inputs, `success-payload-scatter-chart` as output

@app.callback(

    Output('success-payload-scatter-chart', 'figure'),

    [Input('site-dropdown', 'value'),

     Input('payload-slider', 'value')]

)

def update\_scatter\_chart(selected\_site, selected\_payload):

    filtered\_df = spacex\_df[

        (spacex\_df['Payload Mass (kg)'] >= selected\_payload[0]) &

        (spacex\_df['Payload Mass (kg)'] <= selected\_payload[1])

    ]

    if selected\_site != 'ALL':

        filtered\_df = filtered\_df[filtered\_df['Launch Site'] == selected\_site]

    fig = px.scatter(filtered\_df, x='Payload Mass (kg)', y='class',

                     color='Booster Version Category',

                     title='Correlation between Payload and Success')

    return fig

# Run the app

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

    app.run\_server()

