import pandas as pd

import dash

from dash import html, dcc

from dash.dependencies import Input, Output

import plotly.express as px

# Read the SpaceX launch data into a 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\_\_)

# Define the app layout

app.layout = html.Div(children=[

    html.H1('SpaceX Launch Records Dashboard',

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

    dcc.Dropdown(

        id='site-dropdown',

        options=[

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

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

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

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

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

        ],

        value='ALL',

        placeholder="Select a Launch Site",

        searchable=True

    ),

    html.Br(),

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

    html.Br(),

    html.P("Payload range (Kg):"),

    dcc.RangeSlider(

        id='payload-slider',

        min=0,

        max=10000,

        step=1000,

        value=[min\_payload, max\_payload]),

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

])

# Callback for updating the pie chart

@app.callback(

    Output(component\_id='success-pie-chart', component\_property='figure'),

    Input(component\_id='site-dropdown', component\_property='value')

)

def get\_pie\_chart(entered\_site):

    if entered\_site == 'ALL':

        success\_df = spacex\_df[spacex\_df['class'] == 1]

        success\_counts = success\_df['Launch Site'].value\_counts().reset\_index()

        success\_counts.columns = ['Launch Site', 'Success Count']

        fig = px.pie(

            success\_counts,

            values='Success Count',

            names='Launch Site',

            title='Success Rates Across All Sites'

        )

        return fig

    else:

        site\_df = spacex\_df[spacex\_df['Launch Site'] == entered\_site]

        fig = px.pie(

            site\_df,

            names='class',

            title=f'Success vs Failure for site {entered\_site}'

        )

        return fig

@app.callback(

    Output(component\_id='success-payload-scatter-chart', component\_property='figure'),

    [

        Input(component\_id='site-dropdown', component\_property='value'),

        Input(component\_id='payload-slider', component\_property='value')

    ]

)

def update\_scatter\_chart(entered\_site, payload\_range):

    low, high = payload\_range

    df\_filtered = spacex\_df[

        (spacex\_df['Payload Mass (kg)'] >= low) &

        (spacex\_df['Payload Mass (kg)'] <= high)

    ]

    if entered\_site == 'ALL':

        fig = px.scatter(

            df\_filtered,

            x='Payload Mass (kg)',

            y='class',

            color='Booster Version',

            title='Payload vs. Launch Outcome for All Sites'

        )

    else:

        site\_df = df\_filtered[df\_filtered['Launch Site'] == entered\_site]

        fig = px.scatter(

            site\_df,

            x='Payload Mass (kg)',

            y='class',

            color='Booster Version',

            title=f'Payload vs. Launch Outcome for {entered\_site}'

        )

    return fig

# Run the app

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

    app.run()