# Import required libraries

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 airline 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

                                # The default select value is for ALL sites

                                # dcc.Dropdown(id='site-dropdown',...)

                                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 here',

                                        searchable=True

                                        ),

                                html.Br(),

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

                                # If a specific launch site was selected, show the Success vs. Failed counts for the site

                                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',...)

                                dcc.RangeSlider(

                                    id='payload-slider',

                                    min=0,

                                    max=10000,

                                    step=1000,

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

                                    marks={

                                        0: '0',

                                        2500: '2500',

                                        5000: '5000',

                                        7500: '7500',

                                        10000: '10000'

                                    }

                                ),

                                # 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:

# Add a callback function for `site-dropdown` as input, `success-pie-chart` as output

@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':

        filtered\_df = spacex\_df

        fig = px.pie(filtered\_df, values='class', names='Launch Site', title='Total Success Launches for All Sites')

    else:

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

        fig = px.pie(filtered\_df, names='class', title=f"Total Success Launches for site {entered\_site}")

    return fig

# TASK 4:

# Add a callback function for `site-dropdown` and `payload-slider` as inputs, `success-payload-scatter-chart` as output

@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):

    if entered\_site == 'ALL':

        filtered\_df = spacex\_df[(spacex\_df['Payload Mass (kg)'] >= payload\_range[0]) &

                                (spacex\_df['Payload Mass (kg)'] <= payload\_range[1])]

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

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

    else:

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

        filtered\_df = filtered\_df[(filtered\_df['Payload Mass (kg)'] >= payload\_range[0]) &

                                  (filtered\_df['Payload Mass (kg)'] <= payload\_range[1])]

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

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

    return fig

# Run the app

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

    app.run\_server()