# 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',...)

html.Br(),

dcc.Dropdown(

id='site-dropdown',

options=[

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

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

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

{'label': 'CCAFS SLC-40','value': 'CCAFS SLC-40'}, {'label': 'KSC LC-39A', 'value': 'KSC LC-39A'},

],

value='ALL',

placeholder="Select a Launch Site here",

searchable=True

),

# 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

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

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], # Set the initial value to the minimum and maximum payload values

marks={

0: {'label': '0 kg'},

2500: {'label': '2500 kg'},

5000: {'label': '5000 kg'},

7500: {'label': '7500 kg'},

10000: {'label': '10000 kg'}

}

),

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

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

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

])

@app.callback(

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

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

)

def get\_pie(value):

filtered\_df = spacex\_df

if value == 'ALL':

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

return fig

else:

filtered\_df = spacex\_df[spacex\_df['Launch Site'] == value].groupby(['Launch Site', 'class']). \

size().reset\_index(name='class count')

title = f"Total Success Launches for site {value}"

fig = px.pie(filtered\_df,values='class count', names='class', title=title)

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(selected\_site, selected\_payload):

low, high =selected\_payload

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

filtered\_df1 = spacex\_df[mask]

if selected\_site =='ALL':

fig = px.scatter(filtered\_df1, x='Payload Mass (kg)', y='class', color='Booster Version Category',

title='Correlation of Payload and Successful Missions for All Sites')

return fig

else:

filtered\_df2= filtered\_df1[filtered\_df1['Launch Site'] == selected\_site]

fig = px.scatter(filtered\_df2, x='Payload Mass (kg)', y='class', color='Booster Version Category',

title=f'Correlation of Payload and Successful Missions for site {selected\_site}')

return fig

# Run the app

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

app.run\_server()