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',

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',

min=0, max=10000, step = 1000,

marks={0:'0',10000:'10000'},

value=[0,10000]),

# 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':

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

title='Total successful launches for all sites')

return fig

else:

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

site\_counts = filtered\_df['class'].value\_counts().reset\_index()

site\_counts.columns = ['class', 'count']

fig = px.pie(site\_counts, values='count', names='class',

title=f'Total Successful 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 get\_scatter\_chart(entered\_site, payload\_range):

min\_payload, max\_payload = payload\_range # отримуємо діапазон Payload Mass

# Якщо вибрано "ALL", відображаємо графік для всіх сайтів

if entered\_site == 'ALL':

filtered\_df = spacex\_df[

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

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

]

fig = px.scatter(filtered\_df,

x='Payload Mass (kg)',

y='class',

color='Booster Version Category',

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

labels={'class': 'Launch Outcome'})

return fig

# Якщо вибрано конкретний сайт

else:

filtered\_df = spacex\_df[

(spacex\_df['Launch Site'] == entered\_site) &

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

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

]

fig = px.scatter(filtered\_df,

x='Payload Mass (kg)',

y='class',

color='Booster Version Category',

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

labels={'class': 'Launch Outcome'})

return fig

# Run the app

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

app.run\_server(port=8076)