# Python project running in Theia IDE - IBM capstone project

# Import required libraries

import pandas as pd

import dash

from dash import html

from dash import 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':'VAFB SLC-4E', 'value':'VAFB SLC-4E'},

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

{'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

# 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', 100: '100'},

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):

filtered\_df = spacex\_df

success\_failure\_counts = []

if entered\_site == 'ALL':

fig = px.pie(spacex\_df,

values='class',

names='Launch Site',

title='Total success launches for all sites')

return fig

else:

# return the outcomes piechart for a selected site

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

#print(filtered\_df)

dummy\_df = filtered\_df.loc[filtered\_df['class'] == 1]

#print(dummy\_df)

success\_count = len(dummy\_df)

dummy\_df = filtered\_df.loc[filtered\_df['class'] == 0].value\_counts()

failure\_count = len(dummy\_df)

#print(success\_count)

#print(failure\_count)

success\_failure\_counts = [failure\_count, success\_count]

#success\_faliure\_counts = filtered\_df['class'].value\_counts()

#print(success\_failure\_counts)

fig = px.pie(filtered\_df,

values= success\_failure\_counts,

names= ['Failure', 'Success'],

#color= ['Failure', 'Success'],

title='Success versus failure for launch site ' + entered\_site)

return fig

# TASK 4:

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

# and `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\_plot(entered\_site, payload\_range):

if entered\_site == 'ALL':

# filter only the payload within the payload\_range

filtered\_df=spacex\_df.loc[(spacex\_df['Payload Mass (kg)'] >= payload\_range[0]) & (spacex\_df['Payload Mass (kg)'] <= payload\_range[1])]

#print(payload\_range)

#print(filtered\_df)

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

color = "Booster Version Category", title = "Mission outcome versus payload - all sites")

return fig

else:

# retain only the missions launched from site "entered\_site"

#print(payload\_range)

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

# retain only missions within payload range

filtered\_df=filtered\_df.loc[(filtered\_df['Payload Mass (kg)'] >= payload\_range[0]) & (filtered\_df['Payload Mass (kg)'] <= payload\_range[1])]

#print(filtered\_df)

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

color = "Booster Version Category", title = "Mission outcome versus payload for site " + entered\_site)

return fig

# Run the app

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

app.run\_server()