# 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': site, 'value': site} for site in spacex\_df['Launch Site'].unique()],

                                    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={i: str(i) for i in range(0, 10001, 2000)},

                                    value=[min\_payload, max\_payload]

                                ),

                                # 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: Callback for 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':

        fig = px.pie(spacex\_df,

                     values='class',

                     names='Launch Site',

                     title='Total Success Launches by Site')

        return fig

    else:

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

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

        counts.columns = ['class\_outcome', 'count']

        fig = px.pie(counts,

                     values='count',

                     names='class\_outcome',

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

        return fig

# TASK 4: Callback for scatter chart

@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\_plot(selected\_site, payload\_range):

    # Filter by payload range

    low, high = payload\_range

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

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

    if selected\_site == 'ALL':

        fig = px.scatter(filtered\_df,

                         x="Payload Mass (kg)",

                         y="class",

                         color="Booster Version Category",

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

                         hover\_data=['Launch Site'])

    else:

        site\_df = filtered\_df[filtered\_df['Launch Site'] == selected\_site]

        fig = px.scatter(site\_df,

                         x="Payload Mass (kg)",

                         y="class",

                         color="Booster Version Category",

                         title=f"Payload vs. Outcome for {selected\_site}",

                         hover\_data=['Launch Site'])

    return fig

# Run the app

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

    app.run()

1. Which site has the largest successful launches?

KSC LC-39A has the largest successful launches.

1. Which site has the highest launch success rate?

KSC LC-39A

1. Which payload range(s) has the highest launch success rate?

From 2K to 4K

1. Which payload range(s) has the lowest launch success rate?

From 6K to 9K

1. Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest  
   launch success rate?

FT