A screenshot of a graph

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Which site has the largest successful launches? KSC LC-39A
2. Which site has the highest launch success rate?
3. Which payload range(s) has the highest launch success rate? 6000-9000 kg
4. Which payload range(s) has the lowest launch success rate?
5. Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest  
   launch success rate? FT

import pandas as pd

import plotly.express as px

from dash import Dash, dcc, html, Input, Output

# Load the dataset into spacex\_df

spacex\_df = pd.read\_csv('spacex\_launch\_dash.csv')

# Get the unique launch sites from the spacex\_df DataFrame

launch\_sites = spacex\_df['Launch Site'].unique().tolist()

# Calculate min and max payload values for the range slider

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

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

# Create options for the dropdown including the default 'All Sites' option

options = [{'label': 'All Sites', 'value': 'ALL'}] + [{'label': site, 'value': site} for site in launch\_sites]

# Initialize the Dash app

app = Dash(\_\_name\_\_)

# Define the layout of the app

app.layout = html.Div([

    html.H1('SpaceX Launch Dashboard'),

    html.Label('Select a Launch Site:'),

    dcc.Dropdown(

        id='site-dropdown',

        options=options,

        value='ALL',  # Default value is 'All Sites'

        placeholder="Select a Launch Site here",

        searchable=True

    ),

    html.Div(id='site-dropdown-output'),

    dcc.Graph(id='success-pie-chart'),

    html.Label('Select Payload Range (kg):'),

    dcc.RangeSlider(

        id='payload-slider',

        min=min\_payload,

        max=max\_payload,

        step=1000,

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

        marks={min\_payload: str(min\_payload), max\_payload: str(max\_payload)}

    ),

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

])

# Define callback to update pie chart based on dropdown and slider selection

@app.callback(

    Output('success-pie-chart', 'figure'),

    [Input('site-dropdown', 'value'),

     Input('payload-slider', 'value')]

)

def update\_pie\_chart(selected\_site, payload\_range):

    min\_payload, max\_payload = payload\_range

    if selected\_site == 'ALL':

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

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

        title = f'Success Launches by Payload Range ({min\_payload}-{max\_payload} kg)'

    else:

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

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

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

        title = f'Success Launches at {selected\_site} by Payload Range ({min\_payload}-{max\_payload} kg)'

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

    return fig

# Define callback to update scatter chart based on dropdown and slider selection

@app.callback(

    Output('success-payload-scatter-chart', 'figure'),

    [Input('site-dropdown', 'value'),

     Input('payload-slider', 'value')]

)

def update\_scatter\_chart(selected\_site, payload\_range):

    min\_payload, max\_payload = payload\_range

    if selected\_site == 'ALL':

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

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

        title = f'Success Launches by Payload Range ({min\_payload}-{max\_payload} kg)'

    else:

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

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

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

        title = f'Success Launches at {selected\_site} by Payload Range ({min\_payload}-{max\_payload} kg)'

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

    return fig

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

    app.run\_server(debug=True, port=8052)