



1. Which site has the largest successful launches? KSC LC-39A
2. Which site has the highest launch success rate? CCAFS LC-40 o KSC LC-39A
3. Which payload range(s) has the highest launch success rate? In general, intermediate payload ranges (2000-4000 kg) tend to show a higher success rate.
4. Which payload range(s) has the lowest launch success rate? Extreme ranges, especially very low (<1000 kg) or very high (>8000 kg) loads, tend to have a lower success rate.
5. Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest  
   launch success rate? The FT version of the F9 Booster generally has the highest success rate.

**spacex\_dash\_app.py**

# 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 SpaceX 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

    dcc.Dropdown(

        id='site-dropdown',

        options=[

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

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

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

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

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

        ],

        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

    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: f'{i}' for i in range(0, 10001, 2500)},

        value=[min\_payload, max\_payload]

    ),

    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 update\_pie\_chart(selected\_site):

    if selected\_site == 'ALL':

        fig = px.pie(

            spacex\_df,

            values='class',

            names='Launch Site',

            title='Total Success Launches by Site'

        )

    else:

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

        fig = px.pie(

            filtered\_df,

            names='class',

            title=f'Total Success and Failure for site {selected\_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 update\_scatter\_chart(selected\_site, 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'

        )

    else:

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

        fig = px.scatter(

            filtered\_df,

            x='Payload Mass (kg)',

            y='class',

            color='Booster Version Category',

            title=f'Payload vs. Outcome for site {selected\_site}'

        )

    return fig

# Run the app

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

    app.run\_server(port=8060)