# 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 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',...)

                                html.Br(),

                                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

                                # 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(),

                                ])

                                # Function decorator to specify function input and output

@app.callback(Output(component\_id='success-pie-chart', component\_property='figure'),

             Input(component\_id='site-dropdown', component\_property='value'))

def get\_pie\_chart(value):

    filtered\_df = spacex\_df[spacex\_df['Launch Site']==value].groupby('class')

    if value == 'ALL':

        fig = px.pie(spacex\_df, values='class', names='Launch Site', title='Success vs. Failed Launches for All Sites')

        return fig

    else:

        fig = px.pie(filtered\_df, values='class', names='class', title=f'Success vs. Failed Launches for {value}')

        return fig

                                # TASK 3: Add a slider to select payload range

                                #dcc.RangeSlider(id='payload-slider',...)

                                # 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

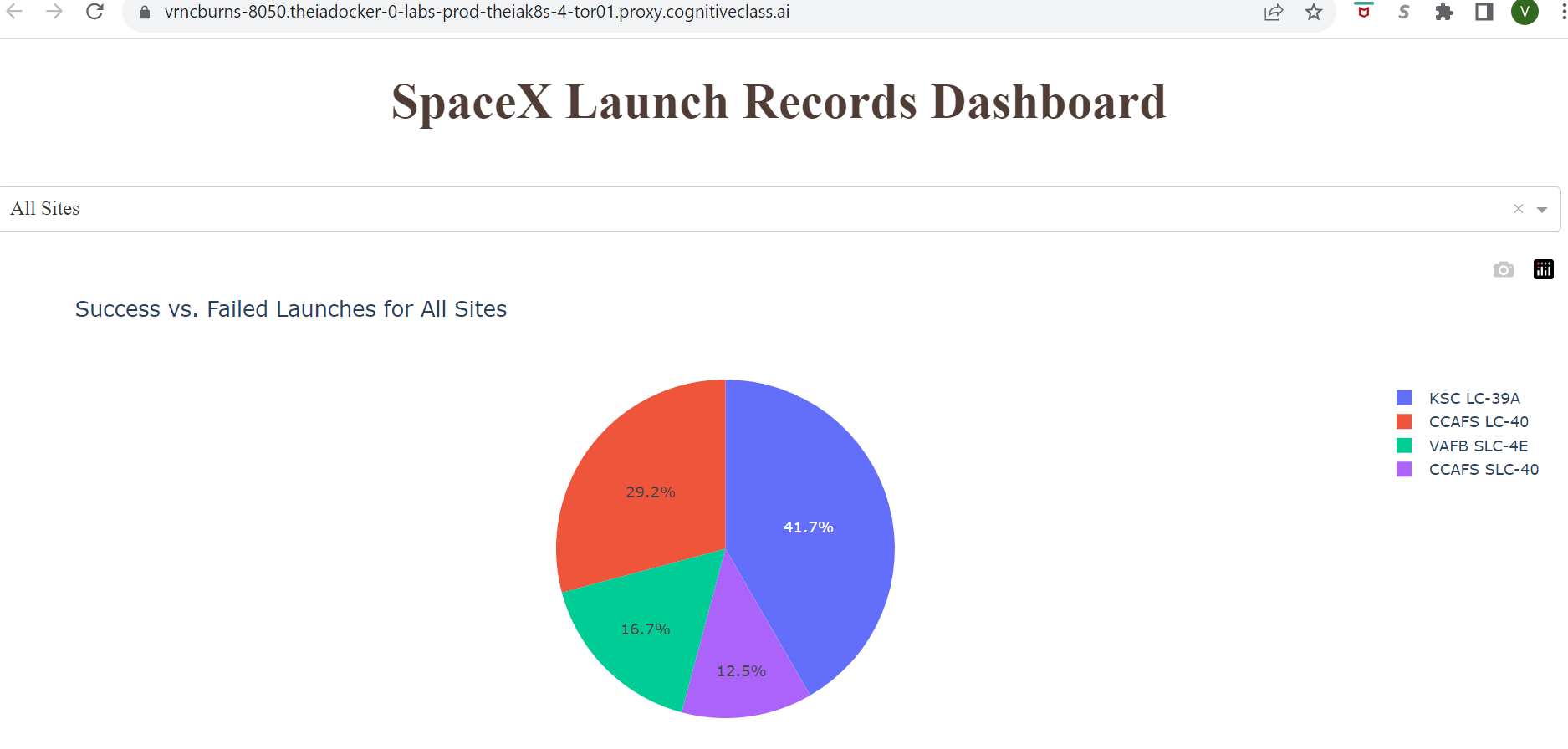
# TASK 4:

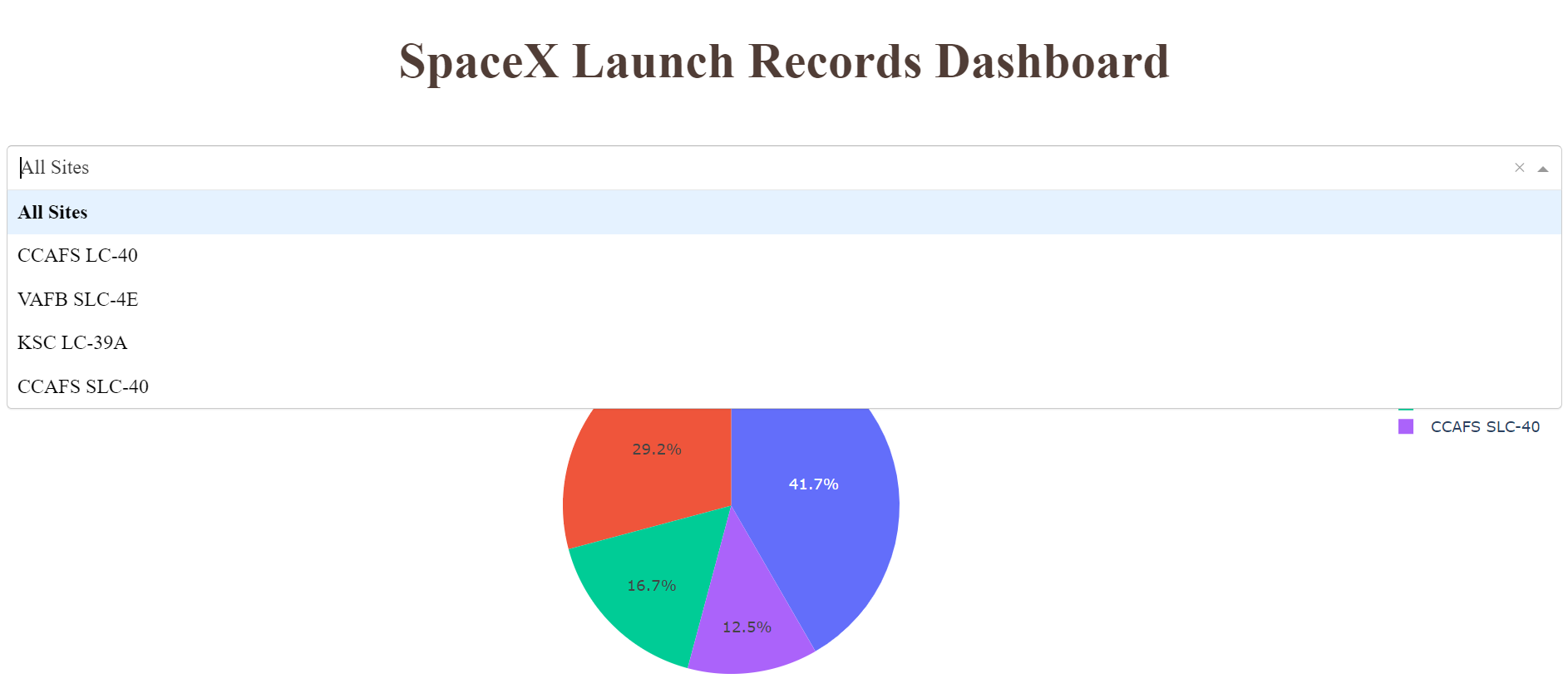
# Add a callback function for `site-dropdown` and `payload-slider` as inputs, `success-payload-scatter-chart` as output

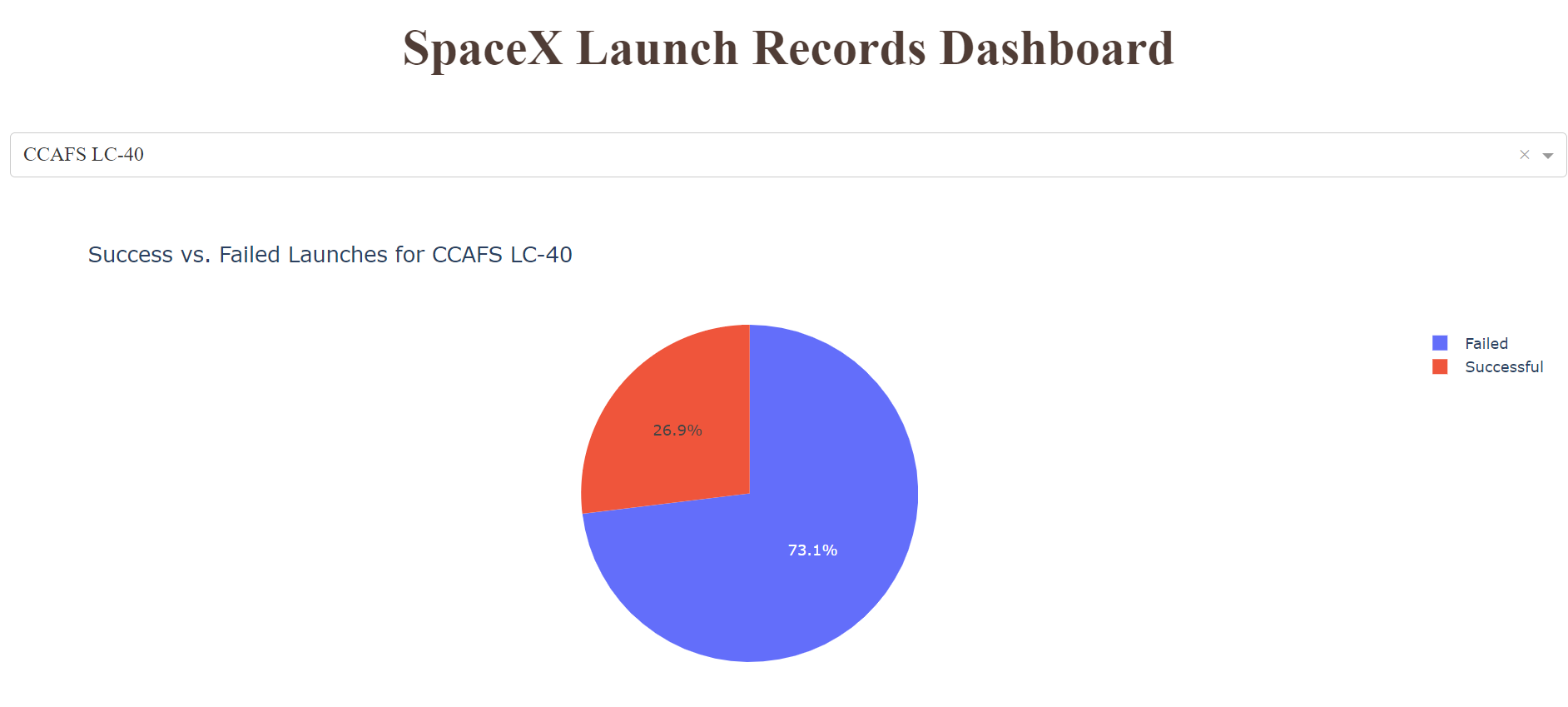
# Run the app

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

    app.run\_server()







# # 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 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',...)

                                html.Br(),

                                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

                                # 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(),

                                ])

                                # Function decorator to specify function input and output

@app.callback(Output(component\_id='success-pie-chart', component\_property='figure'),

             Input(component\_id='site-dropdown', component\_property='value'))

def get\_pie\_chart(value):

    filtered\_df = spacex\_df[spacex\_df['Launch Site']==value]

    if value == 'ALL':

        fig = px.pie(spacex\_df, values='class', names='Launch Site', title='Success vs. Failed Launches for All Sites')

        return fig

    else:

        Failed = len(filtered\_df[filtered\_df['class'] == 0])

        Successful = len(filtered\_df[filtered\_df['class'] == 1])

        labels = ['Failed', 'Successful']

        values = [Failed, Successful]

        fig = px.pie(filtered\_df, values=values, names=labels, title=f'Success vs. Failed Launches for {value}')

        return fig

                                # TASK 3: Add a slider to select payload range

                                #dcc.RangeSlider(id='payload-slider',...)

                                # 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

# TASK 4:

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

# Run the app

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

    app.run\_server()

"password": "FS1MCfRS6kYE1slg",

"username": "nxv72467"

1. Which site has the largest successful launches?

VAFB SLC-4E

1. Which site has the highest launch success rate?

KSC LC-39A

1. Which payload range(s) has the highest launch success rate?
2. 3,000-4,000
3. Which payload range(s) has the lowest launch success rate?
4. 5,500-6,500
5. Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest  
   launch success rate? FT, B5

# 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 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',...)

                                html.Br(),

                                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

                                # 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(),

                                # TASK 3: Add a slider to select payload range

                                #dcc.RangeSlider(id='payload-slider',...)

                                dcc.RangeSlider(id='payload-slider',

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

                                                marks={0: '0',

                                                    100: '100'},

                                                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'))

                                ])

 # Function decorator to specify function input and output

@app.callback(Output(component\_id='success-pie-chart', component\_property='figure'),

             Input(component\_id='site-dropdown', component\_property='value'))

# TASK 2:

# Add a callback function for `site-dropdown` as input, `success-pie-chart` as output

def get\_pie\_chart(value):

    filtered\_df = spacex\_df[spacex\_df['Launch Site']==value]

    if value == 'ALL':

        fig = px.pie(spacex\_df, values='class', names='Launch Site', title='Success vs. Failed Launches for All Sites')

        return fig

    else:

        Failed = len(filtered\_df[filtered\_df['class'] == 0])

        Successful = len(filtered\_df[filtered\_df['class'] == 1])

        labels = ['Failed', 'Successful']

        values = [Failed, Successful]

        fig = px.pie(filtered\_df, values=values, names=labels, title=f'Success vs. Failed Launches for {value}')

        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(value, value1):

    if value == 'ALL':

        filtered\_payload = spacex\_df[(spacex\_df['Payload Mass (kg)'] >= value1[0]) & (spacex\_df['Payload Mass (kg)'] <= value1[1])]

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

        return fig

    else:

        filtered\_df = spacex\_df[spacex\_df['Launch Site']==value]

        filtered\_payload = filtered\_df[(filtered\_df['Payload Mass (kg)'] >= value1[0]) & (filtered\_df['Payload Mass (kg)'] <= value1[1])]

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

        return fig

# Run the app

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

    app.run\_server()