```
# 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__)
uniquelaunchsites = spacex_df['Launch Site'].unique().tolist()
lsites = []
lsites.append({'label': 'All Sites', 'value': 'All Sites'})
for site in uniquelaunchsites:
    lsites.append({'label': site, 'value': site})

# Create an app layout
app.layout = html.Div(children=[html.H1('SpaceX Launch Records Dashboar
d',
                                        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 s
ites
                                # dcc.Dropdown(id='site-dropdown',...)
                                                                    dcc.Dro
pdown(   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': 'CCAFS SLC-
40', 'value': 'CCAFS SLC-40'},
                              {'label': 'KSC LC-39A', 'value': 'KSC LC-
39A'}
        ],
        value='ALL',
                placeholder="Select a Launch Site here",
                searchable=True
```

```
        ),

                                    html.Br(),

                                    # TASK 2: Add a pie chart to show the t
otal successful launches count for all sites
                                    # If a specific launch site was selecte
d, show the Success vs. Failed counts for the site
                                    html.Div(dcc.Graph(id='success-pie-
chart')),
                                    html.Br(),

                                    html.P("Payload range (Kg):"),
                                    dcc.RangeSlider(id='payload_slider', mi
n=0, max=10000,
                                    step=1000, marks = { 0: '0 kg', 1000: '
1000 kg',
                                    2000: '2000 kg',
                                    3000: '3000 kg',
                                    4000: '4000 kg',
                                    5000: '5000 kg',
                                    6000: '6000 kg',
                                    7000: '7000 kg',
                                    8000: '8000 kg',
                                    9000: '9000 kg',
                                    10000: '10000 kg'
                                        },

                                    # TASK 3: Add a slider to select payloa
d range
                                    #dcc.RangeSlider(id='payload-
slider',...)
                                    value=[min_payload,max_payload]
                                    ),

                                    # TASK 4: Add a scatter chart to show t
he 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 update_graph(site_dropdown):
```

```
if (site_dropdown == 'All Sites'):
    df = spacex_df[spacex_df['class'] == 1]
    fig = px.pie(df, names = 'Launch Site',hole=.3,title = 'Total Succe
ss Launches By all sites')
else:
    df = spacex_df.loc[spacex_df['Launch Site'] == site_dropdown]
    fig = px.pie(df, names = 'class',hole=.3,title = 'Total Success Lau
nches for site '+site_dropdown)
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'),Inp
ut(component_id="payload_slider", component_property="value")]
    )
    def update_scattergraph(site_dropdown,payload_slider):
    if site_dropdown == 'All Sites':
        low, high = payload_slider
        df = spacex_df
        mask = (df['Payload Mass (kg)'] > low) & (df['Payload Mass (kg)
'] < high)
        fig = px.scatter(   df[mask], x="Payload Mass (kg)", y="class",
        color="Booster Version",
        size='Payload Mass (kg)',
        hover_data=['Payload Mass (kg)'])
    else:
        low, high = payload_slider
        df = spacex_df.loc[spacex_df['Launch Site'] == site_dropdown]
        mask = (df['Payload Mass (kg)'] > low) & (df['Payload Mass (kg)
'] < high)
        fig = px.scatter(   df[mask], x="Payload Mass (kg)", y="class",
        color="Booster Version",
        size='Payload Mass (kg)',
        hover_data=['Payload Mass (kg)'])
    return fig



# Run the app
if __name__ == '__main__':
    app.run_server()
```