spacex_dash_app.py

```python
# 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
import numpy as np

# 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
sites = [{'label':'All Sites','value':'All Sites'},
         {'label':'CCAFS LC-40', 'value':'CCAFS LC-40'},
         {'label':'CCAFS SLC-40', 'value':'CCAFS SLC-40'},
         {'label':'KSC LC-39A', 'value':'KSC LC-39A'},
         {'label':'VAFB SLC-4E', 'value':'VAFB SLC-4E'}]
```

```python
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(),
                                html.Div(['Sites: ',dcc.Dropdown(id='site-dropdown',options=sites,value='All Sites', style={'height':'20px','font-size':14}),]),
                                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',...)
                                dcc.RangeSlider(id='payload-slider', min=0, max=10000,step=1000,
                                   marks={0:'0', 2500:'2500', 5000:'5000',7500:'7500',10000:'10000'},
                                    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:
# 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 get_success_pie(sites):

    if sites != 'All Sites':
        df = spacex_df[spacex_df['Launch Site']==sites]
        df['label'] = np.where(df['class'] == 0, 'Failed', 'Success')
        df = df.groupby('label', as_index=False).agg({'class':'count'})
        fig = px.pie(df, values = 'class', names='label', title='Launch Success and Failed count for ' +sites, color='label', color_discrete_map={'Success':'green',

    else:
        fig = px.pie(spacex_df, values='class',names='Launch Site', title='Total Success Launches by 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 payload(site,payload):

    min_payload,  max_payload = payload

    df = spacex_df[(spacex_df['Payload Mass (kg)']>=min_payload) & (spacex_df['Payload Mass (kg)']<=max_payload)]
    t = 'Correlation between Payload and Success all all Sites'

    if site != 'All Sites':
        df = df[df['Launch Site']==site]
        t = f'Payload Mass vs Success Rate for {site}'

    fig = px.scatter(df, x='Payload Mass (kg)', y='class',color= 'Booster Version Category',title=t)

    return fig

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