The completed code -

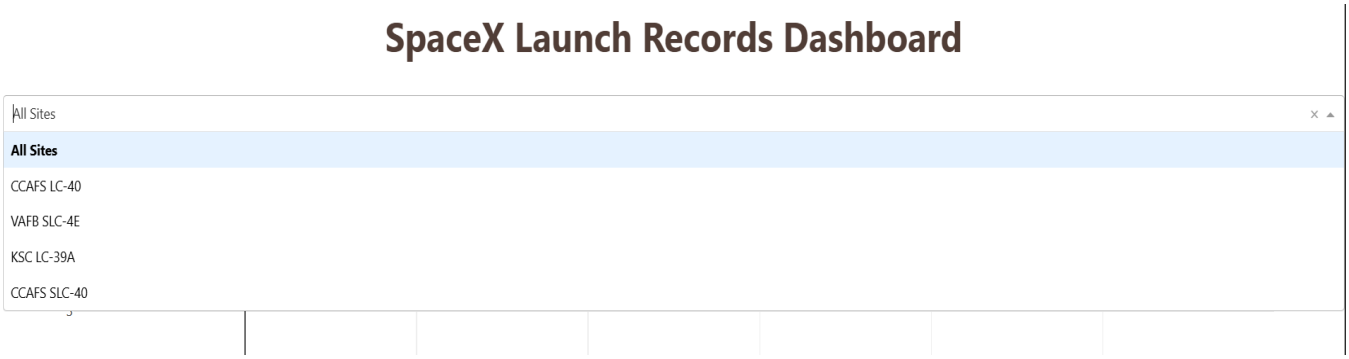
```
# Import required libraries
import pandas as pd
import dash
from dash import html
from dash import dcc
from dash.dependencies import Input, Output
import plotly.express as px

# 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}),
                                dcc.Dropdown(
                                    id='site-dropdown',
                                    options=[{'label': 'All Sites', 'value':
'ALL'}] +
                                            [{'label': site, 'value': site} for
site in spacex_df['Launch Site'].unique()],
                                    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(),

                                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={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('success-pie-chart', 'figure'),
    Input('site-dropdown', 'value')
)
def update_success_pie(selected_site):
    if selected_site == 'ALL':
        # Pie chart for total success launches by site
        fig = px.pie(
            spacex_df[spacex_df['class'] == 1],
            names='Launch Site',
            title='Total Successful Launches by Site'
        )
    else:
        # Pie chart for success/failure for a single site
        site_df = spacex_df[spacex_df['Launch Site'] == selected_site]
        fig = px.pie(
            site_df,
            names='class',
            title=f'Success vs. Failure for {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('success-payload-scatter-chart', 'figure'),
    [
        Input('site-dropdown', 'value'),
```

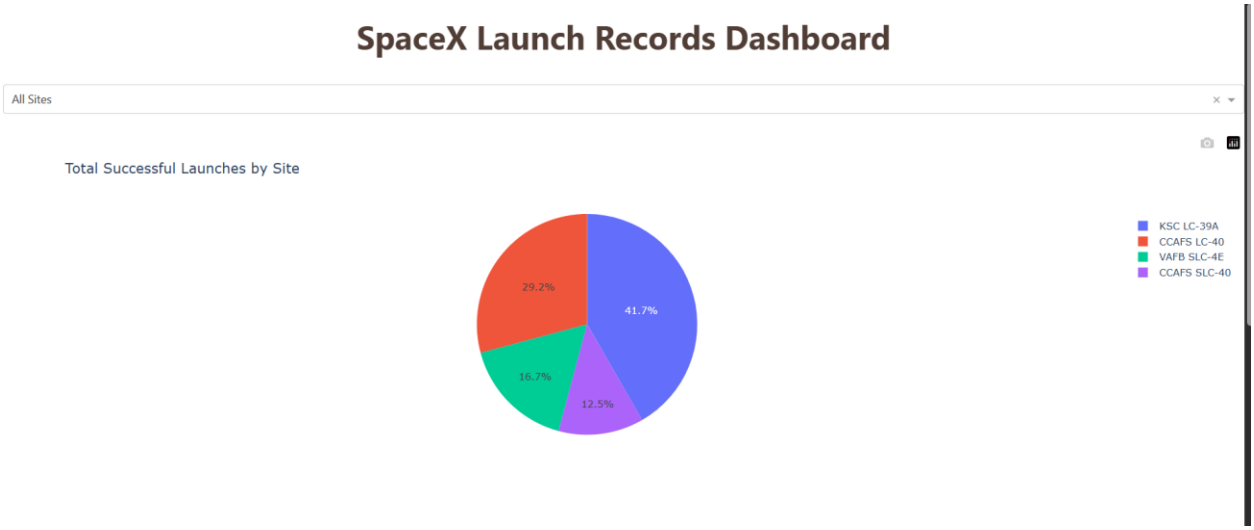
```
        Input('payload-slider', 'value')
    ]
)
def update_scatter(selected_site, payload_range):
    low, high = payload_range
    filtered_df = spacex_df[(spacex_df['Payload Mass (kg)'] >= low) &
(spacex_df['Payload Mass (kg)'] <= high)]
    if selected_site == 'ALL':
        fig = px.scatter(
            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 {selected_site}'
        )
    return fig

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

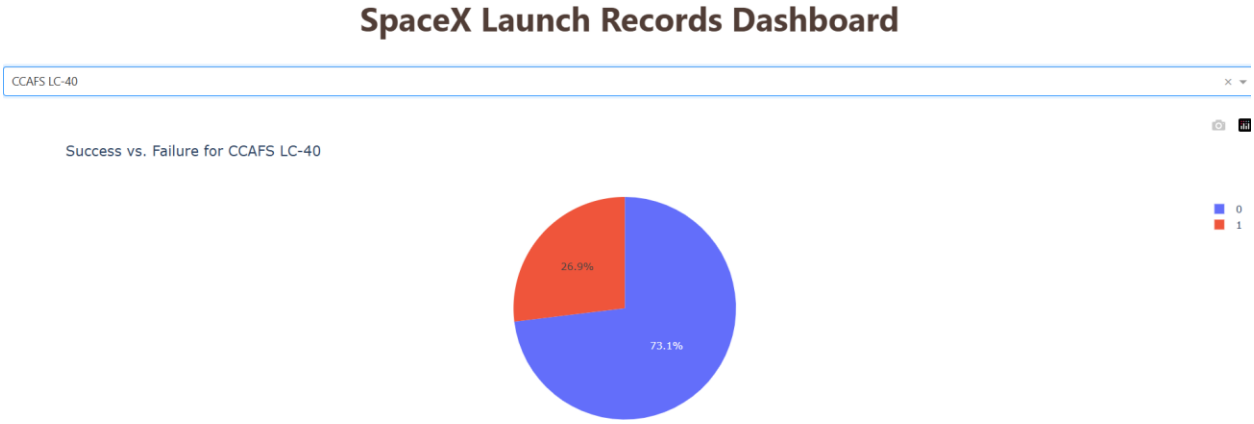
The Output I got by running the above code –

Task 1



Task 2

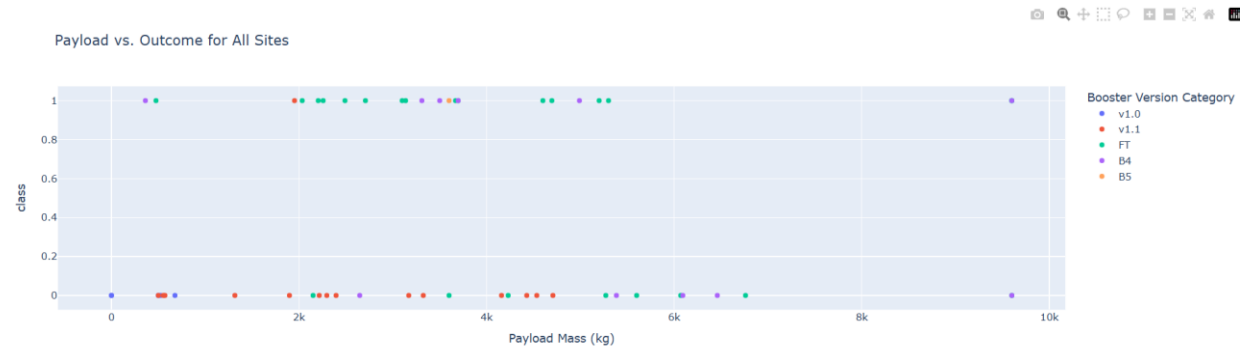

SpaceX Launch Records Dashboard
All Sites
Total Successful Launches by Site
29.2%
41.7%
16.7%
12.5%
KSC LC-39A
CCAFS LC-40
VAFB SLC-4E
CCAFS SLC-40

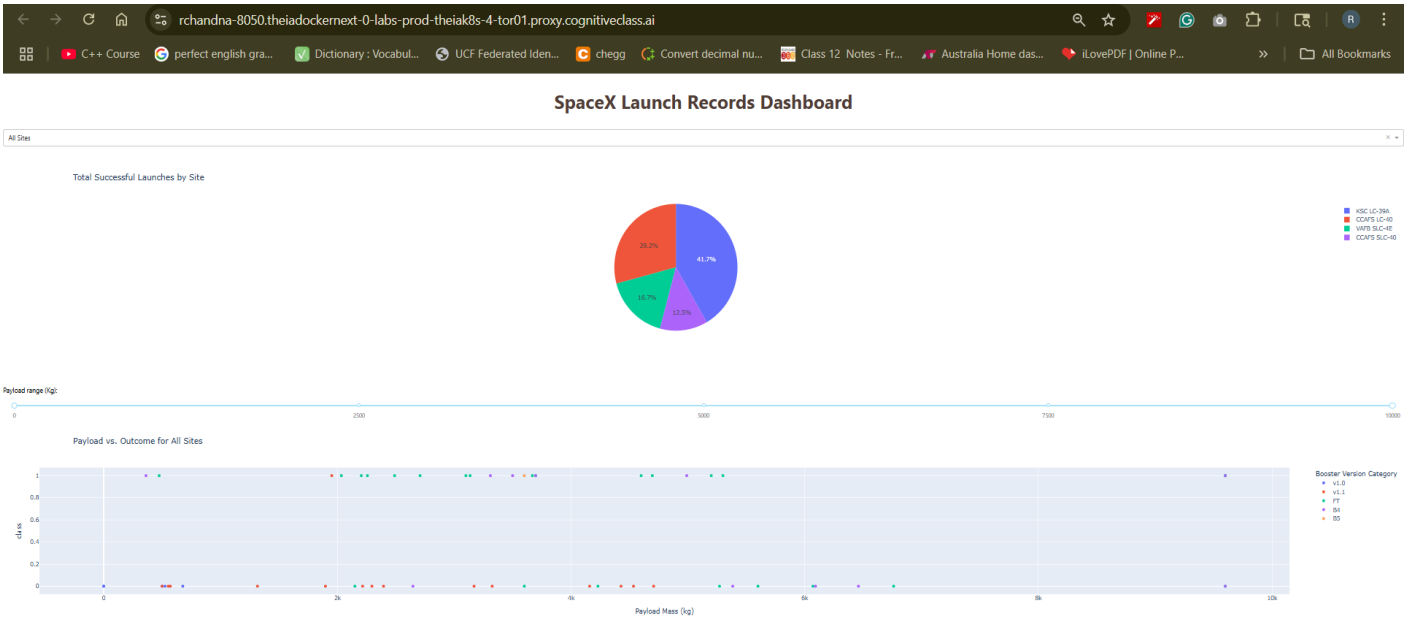

SpaceX Launch Records Dashboard
CCAFS LC-40
Success vs. Failure for CCAFS LC-40
26.9%
73.1%
0
1

Task 3


Payload range (Kg):
0
2500
5000
7500

Task 4


Payload vs. Outcome for All Sites
class
Payload Mass (kg)
Booster Version Category
v1.0
v1.1
FT
B4
B5


rchandna-8050.theiadockernext-0-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai
SpaceX Launch Records Dashboard
All Sites
Total Successful Launches by Site
Payload range (Kg):
Payload vs. Outcome for All Sites
Payload Mass (kg)
Booster Version Category