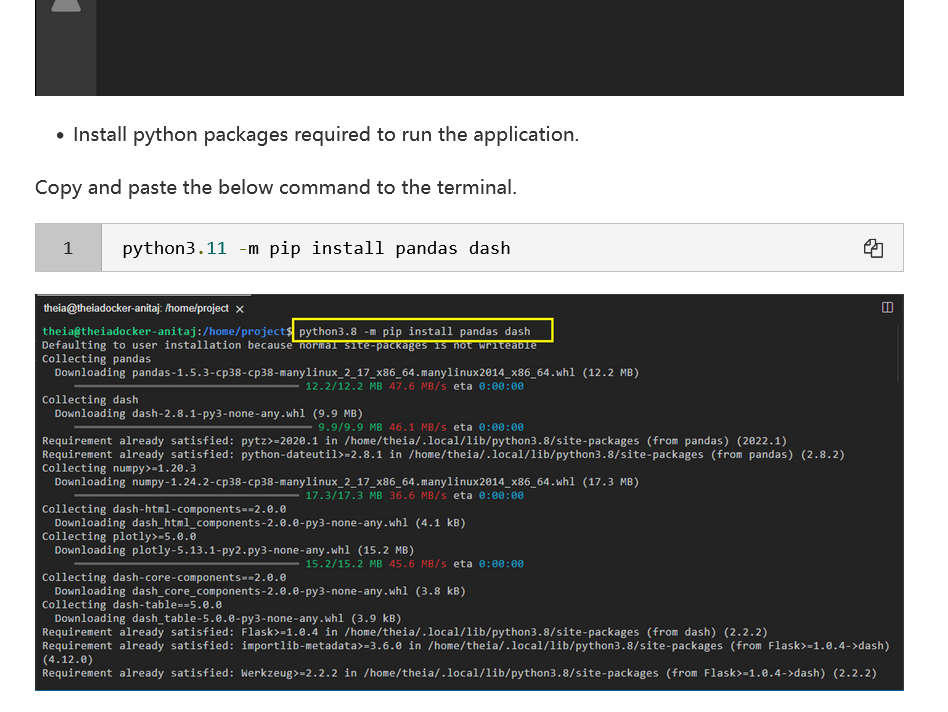
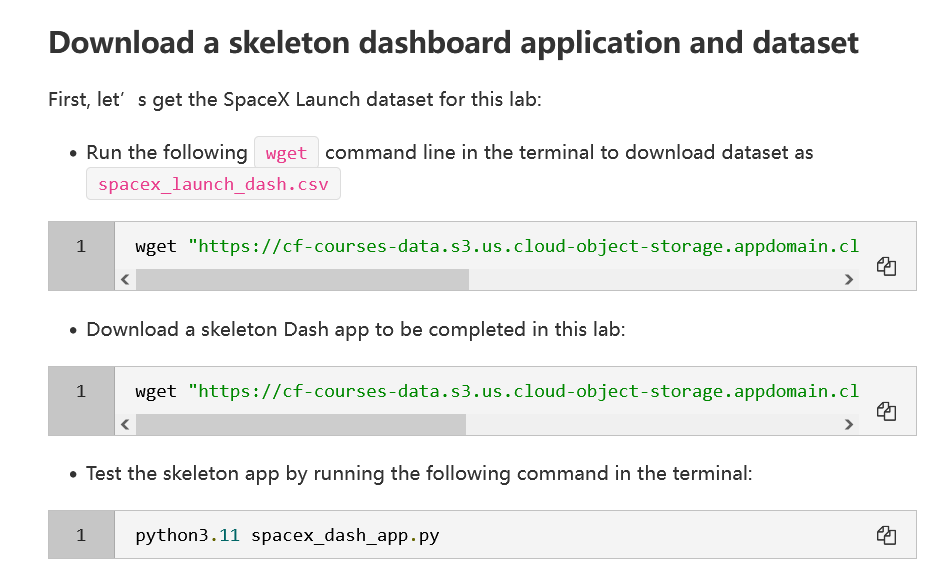
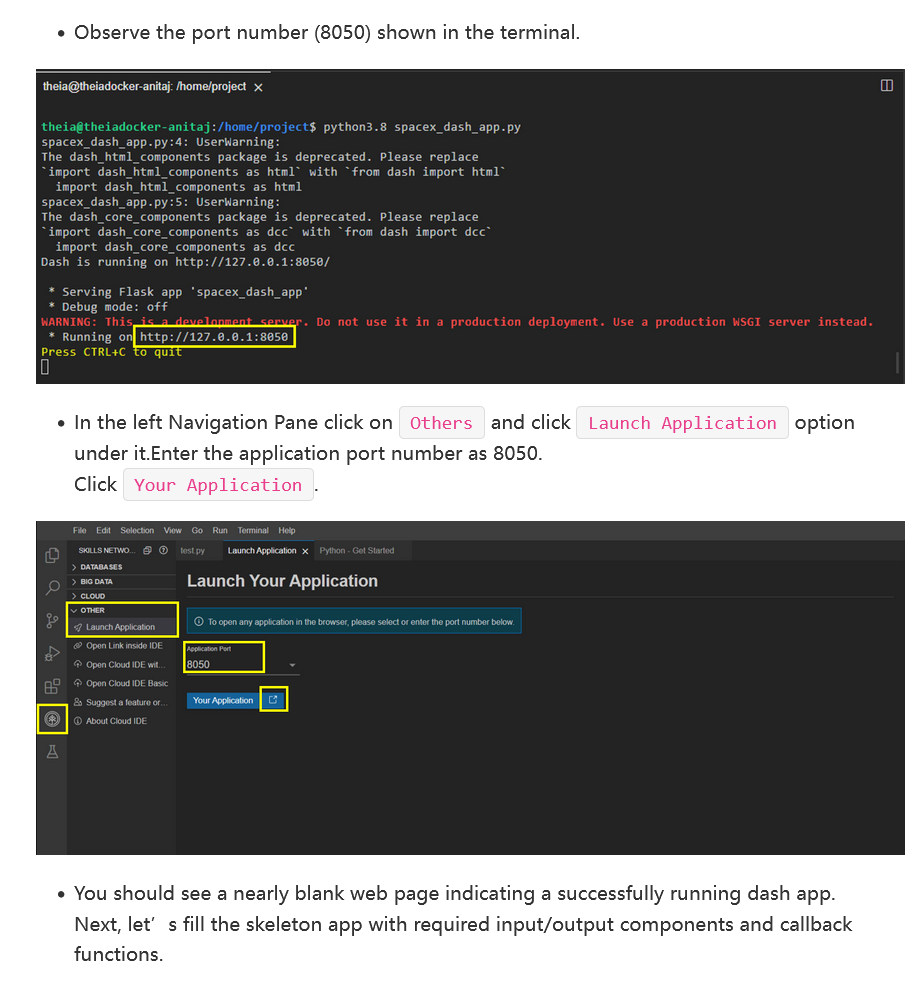
**Build a Dashboard Application with Plotly Dash**

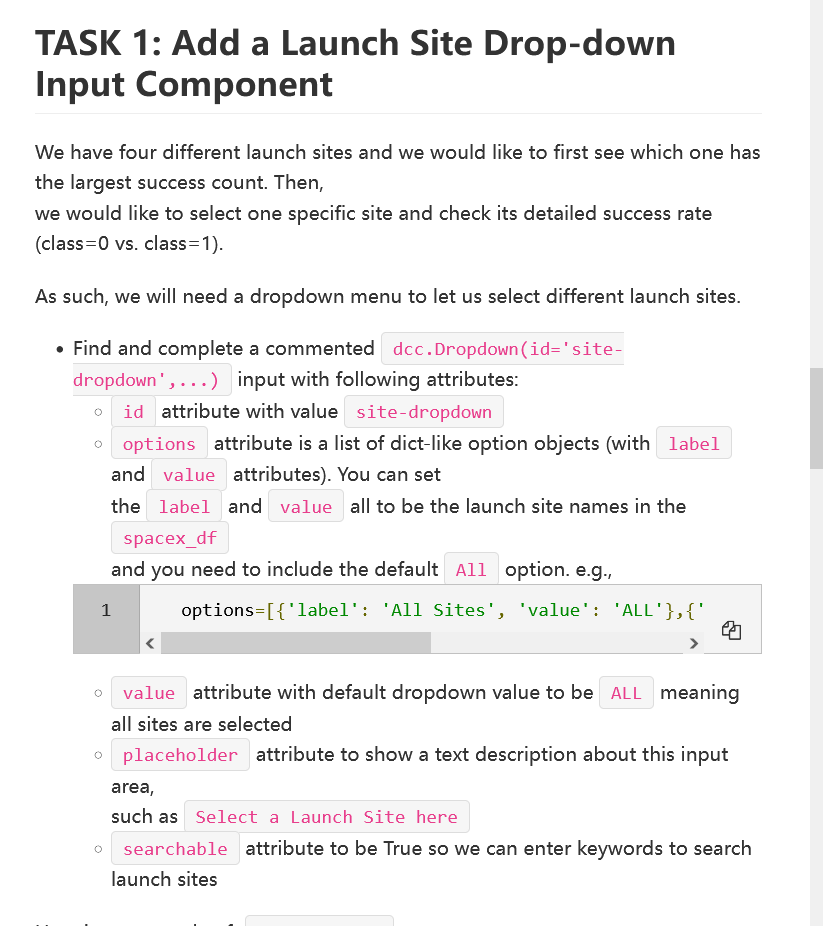
New terminal <-

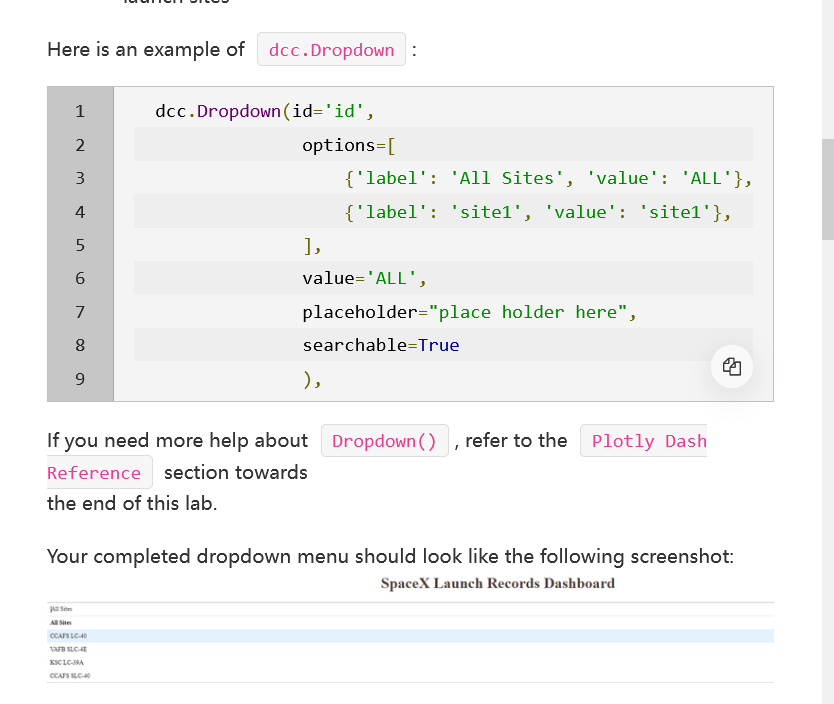
python3.11 -m pip install pandas dash











dcc.Dropdown(id='site-dropdown',

                                             options=[

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

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

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

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

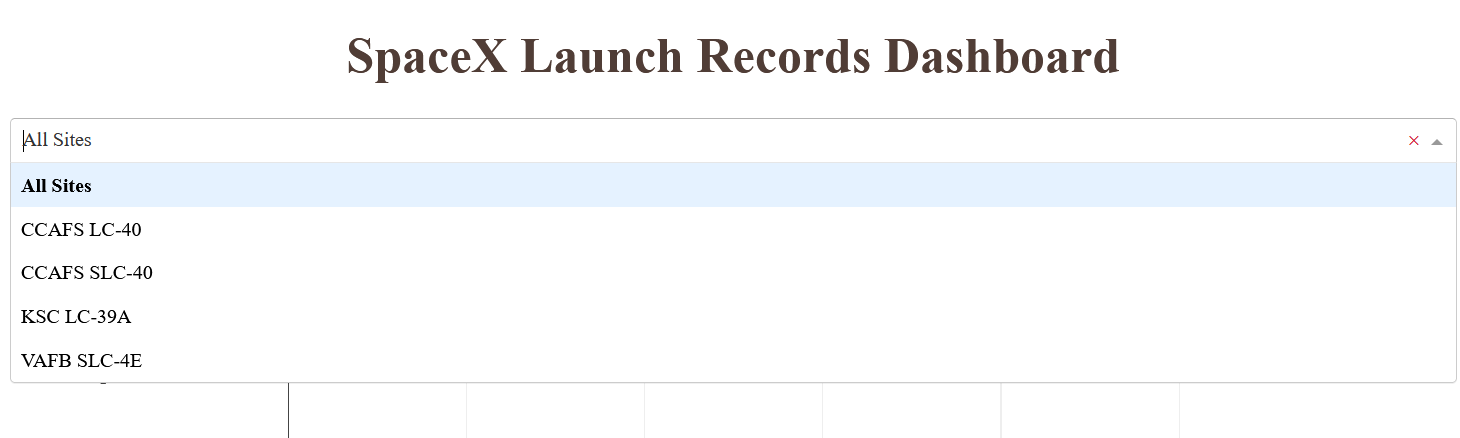
                                                {'label':'VAFB SLC-4E', 'value':'VAFB SLC-4E'},

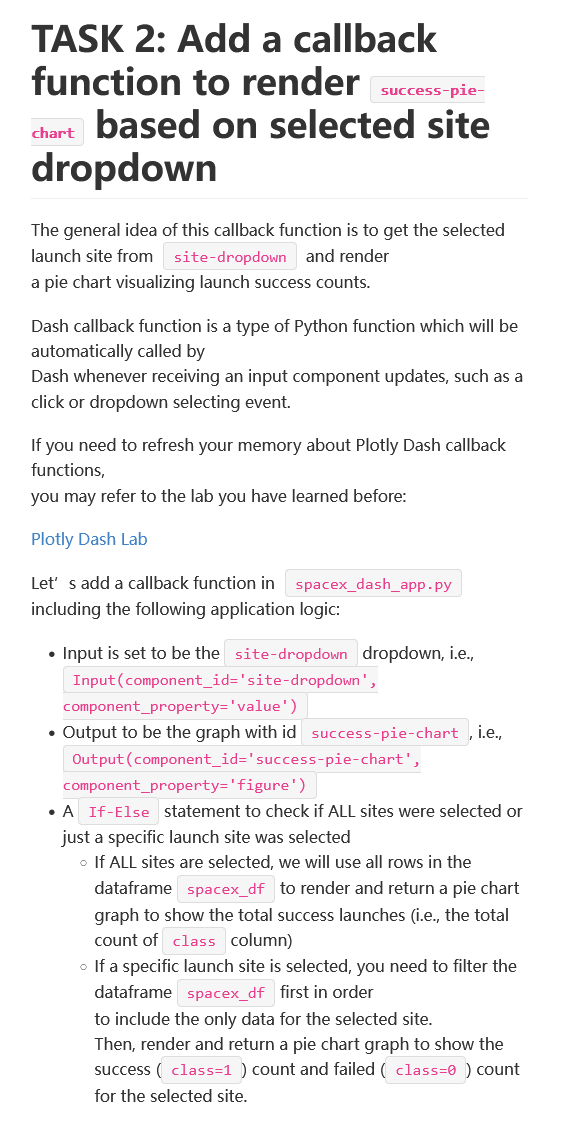
                                             ],

                                              value='ALL',

                                              placeholder="Select a Launch Site here",

                                              searchable=True),





# TASK 2:

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

# 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(entered\_site):

    filtered\_df = spacex\_df

    if entered\_site == 'ALL':

        filtered\_df=filtered\_df[filtered\_df['class']==1].groupby(['Launch Site'],as\_index=False)['class'].count()

        fig = px.pie(filtered\_df, values='class',

        names='Launch Site',

        title=' Total success launches')

        return fig

    else:

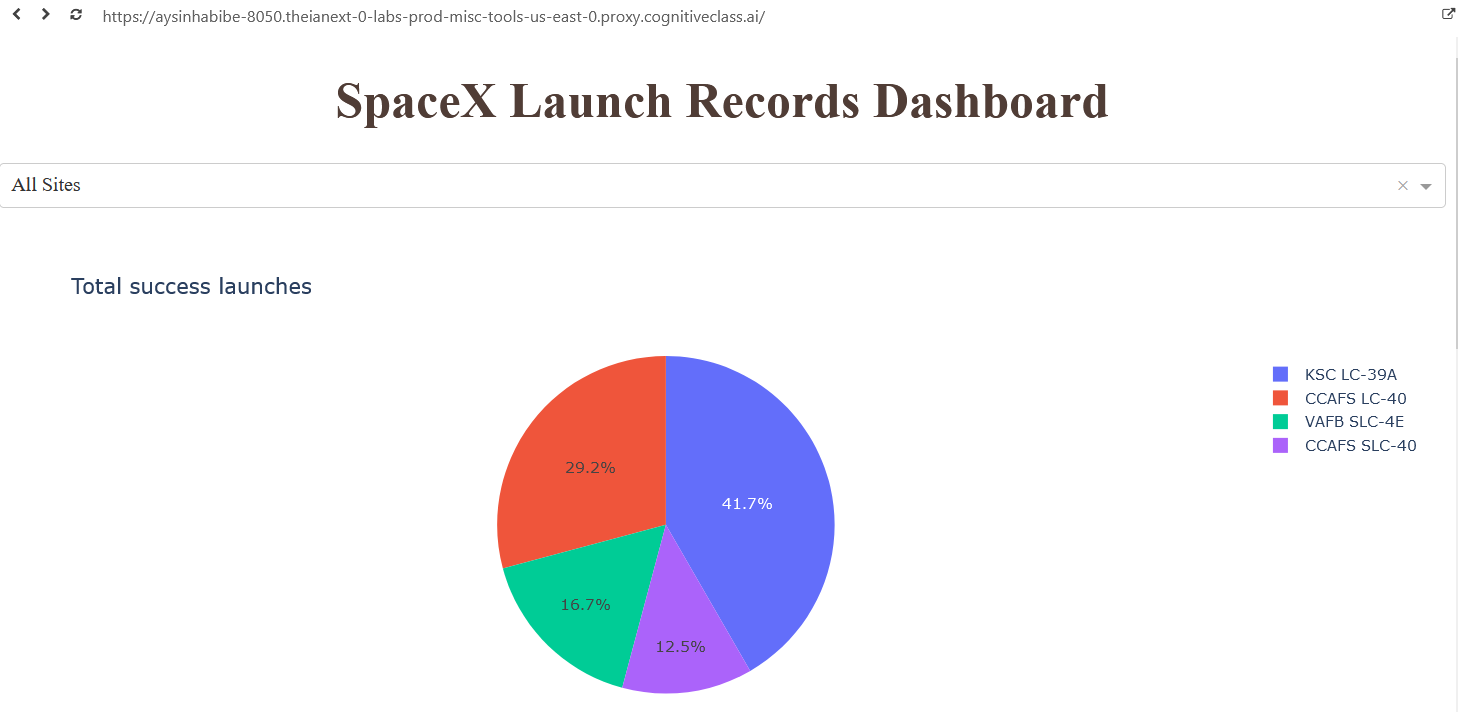
        filtered\_df=filtered\_df[filtered\_df['Launch Site']==entered\_site].groupby(['class'],as\_index=False).count()

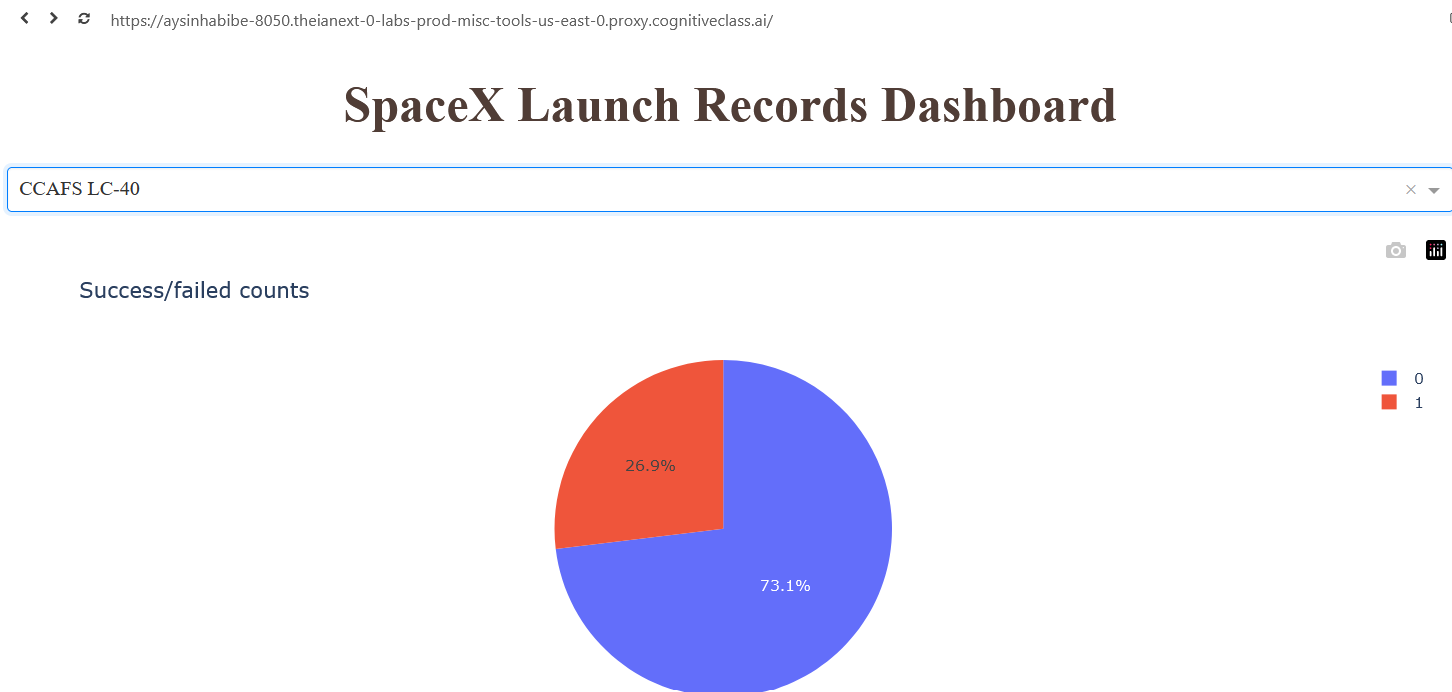
        fig = px.pie(filtered\_df, values='Launch Site',

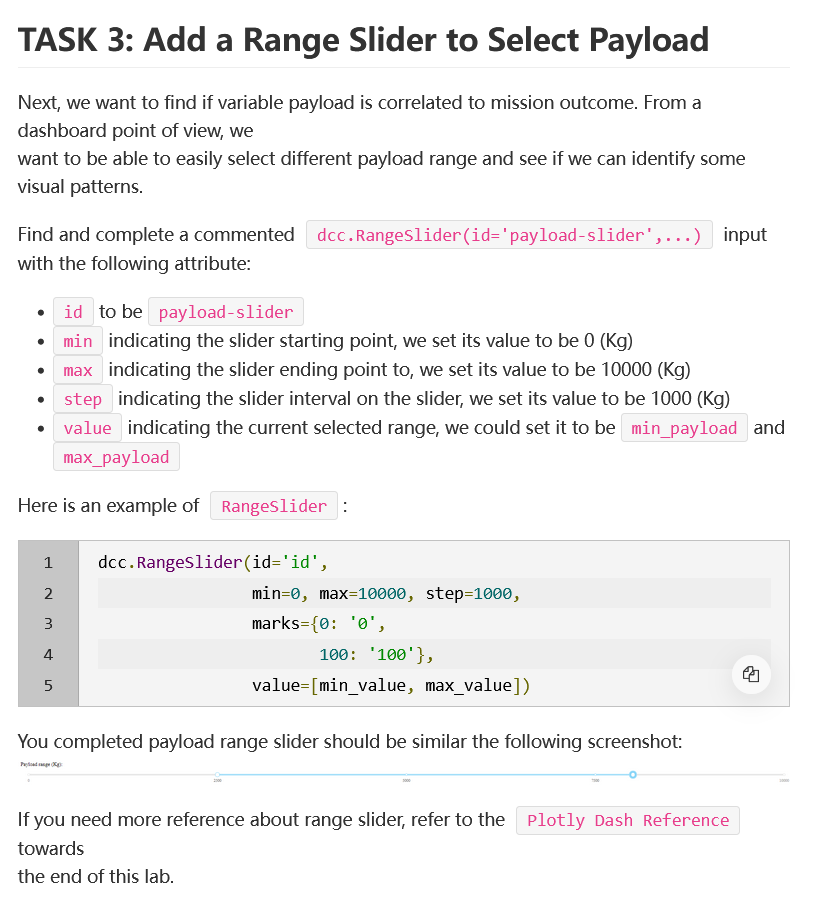
        names='class',

        title=' Success/failed counts')

        return fig







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

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

                                               marks={0:'0',

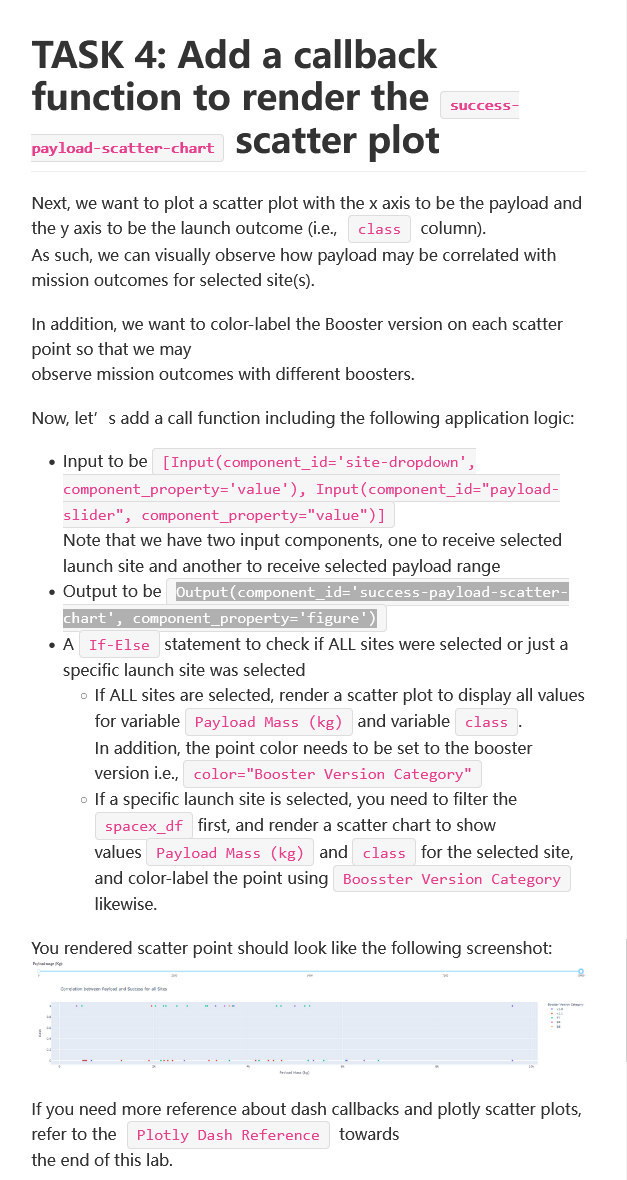
                                                      100:'100'},

                                            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')),

                                ])



@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\_plot(entered\_site,entered\_slider):

    filtered\_df = spacex\_df

    if entered\_site == 'ALL':

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

        return fig

    else:

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

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

        return fig

===all===

# 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',

                                             options=[

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

                                                {'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'},

                                             ],

                                              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',

                                                      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')),

                                ])

# TASK 2:

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

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

def get\_pie\_chart(entered\_site):

    filtered\_df = spacex\_df

    if entered\_site == 'ALL':

        filtered\_df=filtered\_df[filtered\_df['class']==1].groupby(['Launch Site'],as\_index=False)['class'].count()

        fig = px.pie(filtered\_df, values='class',

        names='Launch Site',

        title=' Total success launches')

        return fig

    else:

        filtered\_df=filtered\_df[filtered\_df['Launch Site']==entered\_site].groupby(['class'],as\_index=False).count()

        fig = px.pie(filtered\_df, values='Launch Site',

        names='class',

        title=' Success/failed counts')

        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\_plot(entered\_site,entered\_slider):

    filtered\_df = spacex\_df

    if entered\_site == 'ALL':

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

        return fig

    else:

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

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

        return fig

# Run the app

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

    app.run\_server()