zaddy18 Create Build an Interactive Dashboard with Ploty Dash.py 9e90711 · 2 years ago

95 lines (88 loc) · 5 KB

Code Blame Raw

```
# 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':
                                               'font-size': 40}),
                                # TASK 1: Add a dropdown list to enable Launch
                                # The default select value is for ALL sites
```

```
                                dcc.Dropdown(id='site-dropdown',
                                            options=[
                                                {'label': 'All Sites', 'value':
                                                {'label': 'CCAFS LC-40', 'value
                                                {'label': 'VAFB SLC-4E', 'value
                                                {'label': 'KSC LC-39A', 'value'
                                                {'label': 'CCAFS SLC-40', 'valu
                                            ],
                                            value='ALL',
                                            placeholder="Select a Launch Site h
                                            searchable=True
                                            ),
                                html.Br(),

                                # TASK 2: Add a pie chart to show the total suc
                                # If a specific launch site was selected, show
                                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', 50
                                            value=[min_payload, max_payload
                                # TASK 4: Add a scatter chart to show the corre
                                html.Div(dcc.Graph(id='success-payload-scatter-
                                ])

# TASK 2:
# Add a callback function for `site-dropdown` as input, `success-pie-chart` as
# Function decorator to specify function input and output
@app.callback(Output(component_id='success-pie-chart', component_property='figu
              Input(component_id='site-dropdown', component_property='value'))
def get_pie_chart(entered_site):
    filtered_df = spacex_df
    if entered_site == 'ALL':
        fig = px.pie(filtered_df, values='class',
```

```
            names='Launch Site',
            title='Total Success Launches By Site')
            return fig
        else:
            # return the outcomes piechart for a selected site
            filtered_df = spacex_df[spacex_df['Launch Site'] == entered_site]
            filtered_df = filtered_df.groupby(['Launch Site', 'class']).size().rese
            fig = px.pie(filtered_df, values='class count',
            names='class',
            title=f'Total Success Launched for site {entered_site}')
            return fig
    # TASK 4:
    # Add a callback function for `site-dropdown` and `payload-slider` as inputs, `
    @app.callback(Output(component_id='success-payload-scatter-chart', component_pr
                  [Input(component_id='site-dropdown', component_property='value'),
                  Input(component_id='payload-slider', component_property='value')]
    def get_scatter_chart(entered_site, payload):
        low, high = payload
        filtered_df = spacex_df[(spacex_df['Payload Mass (kg)'] > low) & (spacex_df
        if entered_site == 'ALL':
            fig = px.scatter(filtered_df, x='Payload Mass (kg)', y='class',
            color='Booster Version Category',
            title='Correlation between Payload and Success for all Sites')
            return fig
        else:
            fig = px.scatter(filtered_df[filtered_df['Launch Site'] == entered_site
            x='Payload Mass (kg)', y='class',
            color='Booster Version Category',
            title=f'Correlation between Payload and Success for site {entered_site}
            return fig

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