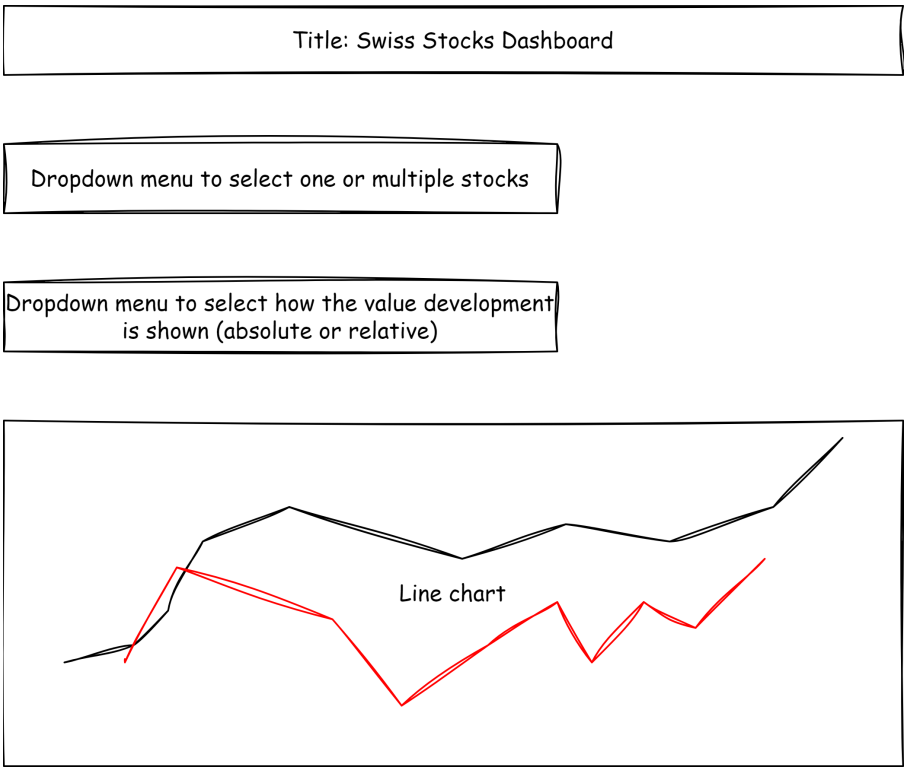
Description of situation:

> A colleague of yours provided you the CSV-file **stock_data_smi.csv**. She asked you to implement a dashboard for analyzing the value development of Swiss stocks since 2008. The dashboard should be organized as follows:



Task:

> Implement this dashboard below:

Solution:

Import packages and data

In [ ]:
```
import pandas as pd
from dash import Dash, html, dcc, callback, Output, Input
import plotly.express as px
```

In [ ]:
```
# Load data
df = pd.read_csv('stock_data_smi.csv', index_col='Date')
```

Create Dash app

In [ ]:
```
# Initialize the app
app = Dash()

# Define app layout
app.layout = html.Div(children=[

    # Add title
    html.H1(children='Swiss Stocks Dashboard', style={'textAlign': 'center'}),

    # Add controls
    html.Div([

        # Add dropdown menu to select stocks
        html.Label('Select stocks to be displayed'),
        dcc.Dropdown(df.columns, value='ABB', id='dropdown_stocks', multi=True),
        html.Br(),

        # Add dropdown menu to select performance type
        html.Label('Choose performance type'),
```

```
        dcc.Dropdown(['Absolute', 'Relative'], value='Absolute', id='dropdown_performance'),

    ], style={'width': 400, 'padding-left': 80}),

    # Add graph next
    dcc.Graph(id='graph'),
    html.Br(),

], style={'fontFamily': 'Arial, sans-serif', 'padding': 40, 'textAlign': 'left'})

@callback(
    Output(component_id='graph', component_property='figure'),
    Input(component_id='dropdown_stocks', component_property='value'),
    Input(component_id='dropdown_performance', component_property='value')
)
def update_graph(dropdown_stocks, dropdown_performance):

    if dropdown_performance == 'Relative':

        # Copy data frame
        df_for_plot = df.copy()

        # Determine relative performance of all stocks
        df_for_plot = df_for_plot / df_for_plot.iloc[-1, :]

        # Create line plot
        fig = px.line(df_for_plot, x=df_for_plot.index, y=dropdown_stocks)

        # Format yaxes
        fig.update_yaxes(tickformat='.0%')
    else:

        # Create line plot
        fig = px.line(df, x=df.index, y=dropdown_stocks)

    # Update legend title
    fig.update_layout(legend_title='Stocks')

    return fig


# Run the app
app.run(debug=True, use_reloader=False)
```