

```
1 import dash
2 import dash_core_components as dcc
3 import dash_html_components as html
4 import pandas as pd
5 import numpy as np
6 from dash.dependencies import Output, Input
7 import os
8 import plotly.express as px
9
10 from assets import whoop
11
12 access_token = whoop.get_access_token("trevor.liggett@gmail.com",
13 os.getenv('WHOOO_PASSWORD'))
14
15 data = whoop.get_user_data_df(access_token,
16                               start_date='2000-01-01T00:00:00.000Z',
17                               end_date='2030-01-01T00:00:00.000Z',
18                               url='https://api-7.whoop.com/users/{}/cycles')
19
20
21
22 data["Date"] = pd.to_datetime(data["date"], format="%Y-%m-%d")
23 data.sort_values("Date", inplace=True)
24
25 # Have to hard code stats at first
26 stats = ['sleep.score', 'sleep.qualityDuration']
27
28
29
30 external_stylesheets = [
31     {
32         "href": "https://fonts.googleapis.com/css2?"
33         "family=Lato:wght@400;700&display=swap",
34         "rel": "stylesheet",
35     },
36 ]
37 app = dash.Dash(__name__, external_stylesheets=external_stylesheets)
38 server = app.server
39 app.title = "Sleep Analysis with WHOOO"
40
41 app.layout = html.Div(
42     children=[
43         html.Div(
44             children=[
45                 html.Img(src="assets/sleep.png", className="header-emoji"),
46                 html.H1(
47                     children="Sleep Analysis: Powered by WHOOO",
48                     className="header-title"
49                 ),
50                 html.P(
51                     children="Monitoring and analyzing"
52                     " sleep patterns between 2018 and 2021"
53                     " using the WHOOO wearable.",
54                     className="header-description",
55                 ),
56             ],
57             className="header",
58         ),
59         html.Div(
```

```

59     children=[
60         html.Div(
61             children=[
62                 html.Div(
63                     children="Date Range", className="menu-title"
64                 ),
65                 dcc.DatePickerRange(
66                     id="date-range",
67                     min_date_allowed=data.Date.min().date(),
68                     max_date_allowed=data.Date.max().date(),
69                     start_date=pd.to_datetime("2021-04-11",
format="%Y-%m-%d"),
70                     end_date=data.Date.max().date(),
71                 ),
72             ],
73         ),
74     ],
75     className="menu",
76 ),
77 html.Div(
78     children=[
79         html.Div(
80             children=dcc.Graph(
81                 id="need-chart",
82                 config={"displayModeBar": False},
83             ),
84             className="card",
85         ),
86         html.Div(
87             children=dcc.Graph(
88                 id="efficiency-chart",
89                 config={"displayModeBar": False},
90             ),
91             className="card",
92         ),
93         html.Div(
94             children=dcc.Graph(
95                 id="consistency-chart",
96                 config={"displayModeBar": False},
97             ),
98             className="card",
99         ),
100        html.Div(
101            children=dcc.Graph(
102                id="pie-chart",
103                config={"displayModeBar": False},
104            ),
105            className="card",
106        ),
107    ],
108    className="wrapper",
109 ),
110 ]
111 )
112
113
114 @app.callback(
115     [Output("need-chart", "figure"), Output("efficiency-chart", "figure"),
Output("consistency-chart", "figure"), Output("pie-chart", "figure")],
116     [

```

```

117     Input("date-range", "start_date"),
118     Input("date-range", "end_date"),
119 ],
120 )
121 def update_charts(start_date, end_date):
122     mask = (data['Date'] > start_date) & (data['Date'] <= end_date)
123     filtered_data = data.loc[mask]
124     # create pie data
125     names = ['sws', 'rem', 'light', 'wake']
126     values = [filtered_data['sleep.sws.duration'].mean() / 3600000,
127               filtered_data['sleep.rem.duration'].mean() /
128               3600000,
129               filtered_data['sleep.light.duration'].mean() /
130               3600000,
131               filtered_data['sleep.wake.duration'].mean() /
132               3600000]
133     pie_data = pd.DataFrame(list(zip(names, values)), columns=['names',
134     'values'])
135
136     need_chart_figure = {
137         "data": [
138             {
139                 "x": filtered_data["Date"],
140                 "y": filtered_data["sleep.qualityDuration"] / 3600000,
141                 "type": "lines",
142                 "name": "Sleep Duration",
143             },
144             {
145                 "x": filtered_data["Date"],
146                 "y": filtered_data["sleep.needBreakdown.total"] / 3600000,
147                 "type": "lines",
148                 "name": "Sleep Need",
149             },
150         ],
151         "layout": {
152             "title": {"text": "Hours of Sleep vs Sleep Need", "x": 0.05,
153     "xanchor": "left"},
154             "xaxis": {"fixedrange": True},
155             "yaxis": {"fixedrange": True},
156             "colorway": ["#52B2BF", "#0A1172"],
157         },
158     }
159
160     efficiency_chart_figure = {
161         "data": [
162             {
163                 "x": filtered_data["Date"],
164                 "y": filtered_data["sleep.efficiency"],
165                 "type": "lines",
166                 "hovertemplate": "%{y:.2f}<extra></extra> percent",
167             },
168         ],
169         "layout": {
170             "title": {
171                 "text": "Sleep Efficiency",
172                 "x": 0.05,
173                 "xanchor": "left",
174             },
175             "xaxis": {"fixedrange": True},
176         },
177     }

```

```

172         "yaxis": {"tickprefix": "", "fixedrange": True},
173         "colorway": ["#7A4988"],
174     },
175 }
176
177 consistency_chart_figure = {
178     "data": [
179         {
180             "x": filtered_data["Date"],
181             "y": filtered_data["sleep.consistency"],
182             "type": "lines",
183             "hovertemplate": "%{y:.2f}<extra></extra> percent",
184         },
185     ],
186     "layout": {
187         "title": {
188             "text": "Sleep Consistency",
189             "x": 0.05,
190             "xanchor": "left",
191         },
192         "xaxis": {"fixedrange": True},
193         "yaxis": {"tickprefix": "", "fixedrange": True},
194         "colorway": ["#7A4988"],
195     },
196 }
197
198 pie_chart_figure = px.pie(pie_data, values=values, names=names,
199
200 color_discrete_sequence=px.colors.sequential.Purp,
201                             hole=.3,
202                             title="Proportion of Sleep Spent in Major
203 Stages")
204
205 return need_chart_figure, efficiency_chart_figure,
206 consistency_chart_figure, pie_chart_figure
207
208 if __name__ == "__main__":
209     app.run_server(debug=True)
210

```