Match

Graphical user interface, text, application, email

Description automatically generated

TODO

html.Div([

html.Div("Dash To-Do List"),

dcc.Input(id="new-item"),

html.Button("Add", id="add"),

html.Button("Clear Done", id="clear-done"),

html.Div(id="list-container"),

html.Div(id="totals"),

]),

style\_todo = {"display": "inline", "margin": "10px"}

style\_done = {"textDecoration": "line-through", "color": "#888"}

style\_done.update(style\_todo)

@app.callback(

[

Output("list-container", "children"),

Output("new-item", "value"),

],

[

Input("add", "n\_clicks"),

Input("new-item", "n\_submit"),

Input("clear-done", "n\_clicks"),

],

[

State("new-item", "value"),

State({"index": ALL}, "children"),

State({"index": ALL, "type": "done"}, "value"),

]

)

def edit\_list(add, add2, clear, new\_item, items, items\_done):

triggered = [t["prop\_id"] for t in dash.callback\_context.triggered]

adding = len([i for i in triggered if i in ("add.n\_clicks", "new-item.n\_submit")])

clearing = len([i for i in triggered if i == "clear-done.n\_clicks"])

new\_spec = [

(text, done) for text, done in zip(items, items\_done)

if not (clearing and done)

]

if adding:

new\_spec.append((new\_item, []))

new\_list = [

html.Div([

dcc.Checklist(

id={"index": i, "type": "done"},

options=[{"label": "", "value": "done"}],

value=done,

style={"display": "inline"},

labelStyle={"display": "inline"}

),

html.Div(text, id={"index": i}, style=style\_done if done else style\_todo)

], style={"clear": "both"})

for i, (text, done) in enumerate(new\_spec)

]

return [new\_list, "" if adding else new\_item]

@app.callback(

Output({"index": MATCH}, "style"),

[Input({"index": MATCH, "type": "done"}, "value")]

)

def mark\_done(done):

return style\_done if done else style\_todo

@app.callback(

Output("totals", "children"),

[Input({"index": ALL, "type": "done"}, "value")]

)

def show\_totals(done):

count\_all = len(done)

count\_done = len([d for d in done if d])

result = "{} of {} items completed".format(count\_done, count\_all)

if count\_all:

result += " - {}%".format(int(100 \* count\_done/count\_all))

return result

Graphical user interface, text, application

Description automatically generatedText

Description automatically generated

Graphical user interface, text, application

Description automatically generated