{

"cells": [

{

"cell\_type": "code",

"execution\_count": 41,

"metadata": {

"scrolled": true

},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

" <iframe src=\"/app/endpoints/7cbe7d98ecfe4fe69aac031cee3b5070/\" width=800 height=600 frameborder=\"0\"></iframe>\n",

" <hr/><a href=\"/app/endpoints/7cbe7d98ecfe4fe69aac031cee3b5070/\" target=\"\_new\">Open in new window</a> for /app/endpoints/7cbe7d98ecfe4fe69aac031cee3b5070/\n",

"</div>"

],

"text/plain": [

"<jupyter\_plotly\_dash.dash\_wrapper.JupyterDash at 0x121ab7510>"

]

},

"execution\_count": 41,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"from jupyter\_plotly\_dash import JupyterDash\n",

"\n",

"import dash\n",

"import dash\_leaflet as dl\n",

"import dash\_core\_components as dcc\n",

"import dash\_html\_components as html\n",

"import plotly.express as px\n",

"import dash\_table as dt\n",

"from dash.dependencies import Input, Output, State\n",

"\n",

"import os\n",

"import numpy as np\n",

"import pandas as pd\n",

"from pymongo import MongoClient\n",

"from bson.json\_util import dumps\n",

"\n",

"#### FIX ME #####\n",

"Project\_One.ipynb\n",

"\n",

"\n",

"\n",

"\n",

"\n",

"###########################\n",

"# Data Manipulation / Model\n",

"###########################\n",

"# FIX ME change for your username and password and CRUD Python module name\n",

"username = \"myTestAdmin\"\n",

"password = \"test\"\n",

"shelter = AnimalShelter(username, password)\n",

"\n",

"\n",

"# class read method must support return of cursor object \n",

"df = pd.DataFrame.from\_records(shelter.read({})\n",

"\n",

"\n",

"\n",

"#########################\n",

"# Dashboard Layout / View\n",

"#########################\n",

"app = JupyterDash('SimpleExample')\n",

"\n",

"#FIX ME Add in Grazioso Salvare’s logo\n",

"image\_filename = 'Grazioso Salvare Logo.png' # replace with your own image\n",

"encoded\_image = base64.b64encode(open(image\_filename, 'rb').read())\n",

"\n",

"#FIX ME Place the HTML image tag in the line below into the app.layout code according to your design\n",

"#FIX ME Also remember to include a unique identifier such as your name or date\n",

"#html.Img(src='data:image/png;base64,{}'.format(encoded\_image.decode()))\n",

"\n",

"app.layout = html.Div([\n",

"# html.Div(id='hidden-div', style={'display':'none'}),\n",

" html.Center(html.B(html.H1('SNHU CS-340 Dashboard'))),\n",

" html.Hr(),\n",

" html.Div(\n",

" html.H1(\"Jamilyn's Dashboard\"),

")\n"

"#FIXME Add in code for the interactive filtering options. For example, Radio buttons, drop down, checkboxes, etc.\n",

"\n",

"\n",

" ),\n",

" html.Hr(),\n",

" dt.DataTable(\n",

" id='datatable-id',\n",

" columns=[\n",

" {\"name\": i, \"id\": i, \"deletable\": False, \"selectable\": True} for i in df.columns\n",

" ],\n",

" data=df.to\_dict('records'),\n",

"#FIXME: Set up the features for your interactive data table to make it user-friendly for your client\n",

"#If you completed the Module Six Assignment, you can copy in the code you created here \n",

"@app.callback(\n",

" Output('datatable-id', 'style\_data\_conditional'),\n",

" [Input('datatable-id', 'selected\_columns')]\n",

")\n",

"def update\_styles(selected\_columns):\n",

" return [{\n",

" 'if': { 'column\_id': i },\n",

" 'background\_color': '#D2F3FF'\n",

" } for i in selected\_columns]\n",

"\n",

"\n",

"@app.callback(\n",

" Output('map-id', \"children\"),\n",

" [Input('datatable-id', \"derived\_viewport\_data\")])\n",

"def update\_map(viewData):\n",

"#This sets up the dashboard so that your chart and your geolocation chart are side-by-side\n",

" html.Div(className='row',\n",

" style={'display' : 'flex'},\n",

" children=[\n",

" html.Div(\n",

" id='graph-id',\n",

" className='col s12 m6',\n",

"\n",

" ),\n",

" html.Div(\n",

" id='map-id',\n",

" className='col s12 m6',\n",

" )\n",

" ])\n",

"])\n",

"\n",

"#############################################\n",

"# Interaction Between Components / Controller\n",

"#############################################\n",

"\n",

"\n",

"\n",

" \n",

"@app.callback([Output('datatable-id','data'),\n",

" Output('datatable-id','columns')],\n",

" [Input('filter-type', 'value')])\n",

"def update\_dashboard(filter\_type):\n",

"### FIX ME Add code to filter interactive data table with MongoDB queries\n",

"\n",

" \n",

" columns=[{\"name\": i, \"id\": i, \"deletable\": False, \"selectable\": True} for i in df.columns]\n",

" data=df.to\_dict('records')\n",

" \n",

" \n",

" return (data,columns)\n",

"\n",

"\n",

"\n",

"\n",

"@app.callback(\n",

" Output('datatable-id', 'style\_data\_conditional'),\n",

" [Input('datatable-id', 'selected\_columns')]\n",

")\n",

"def update\_styles(selected\_columns):\n",

" return [{\n",

" 'if': { 'column\_id': i },\n",

" 'background\_color': '#D2F3FF'\n",

" } for i in selected\_columns]\n",

"\n",

"@app.callback(\n",

" Output('graph-id', \"children\"),\n",

" [Input('datatable-id', \"derived\_viewport\_data\")])\n",

"def update\_graphs(viewData):\n",

" ###FIX ME ####\n",

" # add code for chart of your choice (e.g. pie chart) #\n",

" #return [\n",

" # dcc.Graph( \n",

" # figure = ###\n",

" # ) \n",

" #]\n",

"\n",

"@app.callback(\n",

" Output('map-id', \"children\"),\n",

" [Input('datatable-id', \"derived\_viewport\_data\")])\n",

"def update\_map(viewData):\n",

"#FIXME: Add in the code for your geolocation chart\n",

"#If you completed the Module Six Assignment, you can copy in the code you created here.\n",

"def update\_map(viewData):

"dff = pd.DataFrame.from\_dict(viewData)

"# Austin TX is at [30.75,-97.48]

"return [

" dl.Map(style={'width': '1000px', 'height': '500px'}, center=[30.75,-97.48], zoom=10, children=[

" dl.TileLayer(id="base-layer-id"),

" # Marker with tool tip and popup

" dl.Marker(position=[30.75,-97.48], children=[

" dl.Tooltip(dff.iloc[0,4]),

" dl.Popup([

" html.H1("Animal Name"),

" html.P(dff.iloc[1,9])

" ])

" ])

" ])

"] \n",

"]

"},

"{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

},

{

"cell\_type": "code",

"execution\_count": null,

"metadata": {},

"outputs": [],

"source": []

}

],

"metadata": {

"kernelspec": {

"display\_name": "Python 3",

"language": "python",

"name": "python3"

},

"language\_info": {

"codemirror\_mode": {

"name": "ipython",

"version": 3

},

"file\_extension": ".py",

"mimetype": "text/x-python",

"name": "python",

"nbconvert\_exporter": "python",

"pygments\_lexer": "ipython3",

"version": "3.7.7"

}

},

"nbformat": 4,

"nbformat\_minor": 2

}