# CS 340 README Antonio Ocasio

## About the Project/Project Title

The purpose of this application is to provide a web interface that allows the u ser to view data that is stored in a Mongo database. The data includes dog breeds, sex, age, and type of rescue the dog is. The data is provided through a dashboard to make it easy for the user to navigate.

## Motivation

The motivation behind this project was to provide an easy way for the user to access the database. By using python and MongoDB, this made it possible to create a webpage that uses a dashboard which allows the user to easily view and sort the data that is stored in the database. Since python uses SQL tools, we were able to use the syntax from SQL that I learned from before. We also were able to incorporate the CRUD methods from the AnimalShelter.py file.

## Getting Started

The first step to installing this program is to install and create a Mongo Database. You will then need to program the python CRUD file. After this, you will need to create a dashboard web application using the python CRUD file that you created.

1. Create database with collection called aac
2. Create user with admin privileges for aac database
3. Import the aac\_shelter\_outcomes.csv file to the database
4. Login as admin
5. Create a new dash web application with the proper html layout. Then create the ids, data data frame and map
6. Create an app callback to add the data frame with the data from the database
7. Create radio buttons for the user to sort the database.
8. Create the callback for the application to update the map with the item in the category that the user chooses.
9. Create the pie chart from the data that is on the screen. Create a callback that updates the pie chart with the data that is chosen from the filters.

## Installation

The tools that you would need for this application are Jupyter Notebook, Python, and MongoDB.

To install Jupyter Notebooks you can get them from <https://jupyter.org/install>

To install Python you can get them from <https://www.python.org/downloads/>

Finally to install MongoDB you can install it from https://docs.mongodb.com/manual/installation/

## Usage

### Code Example

*import base64*

*import logging*

*import dash*

*import dash\_core\_components as dcc*

*import dash\_html\_components as html*

*import dash\_leaflet as dl*

*import plotly.express as px*

*import dash\_table*

*import pandas as pd*

*from dash.dependencies import Input, Output*

*from dash.exceptions import PreventUpdate*

*from flask import Flask*

*from AnimalShelter import AnimalShelter*

*# Create the dash application*

*server = Flask(\_\_name\_\_)*

*logger = logging.getLogger(\_\_name\_\_)*

*app = dash.Dash(*

*\_\_name\_\_,*

*url\_base\_pathname="/animal-shelter/",*

*server=server,*

*)*

*# username and password and CRUD Python module name*

*username = "accuser"*

*password = "aacuser"*

*aac = AnimalShelter(username, password)*

*logger.info("Connected to {aac.database.name} Database")*

*# Add in Grazioso Salvare’s logo*

*image\_filename = 'Grazioso\_Salvare\_Logo.png' # replace with your own image*

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

*df = pd.DataFrame.from\_records(aac.read({}))*

*# Declare the application interfaces*

*app.layout = html.Div([*

*html.Hr(),*

*html.Div(id="query\_out"),*

*html.Div(id="hidden\_div", style={"display": "none"}),*

*# Use row and col to control vertical alignment of logo / brand*

*html.Div(*

*[*

*dbc.Col(*

*[*

*html.A(*

*[*

*html.Img(*

*src="data:image/png;base64,{}".format(*

*encoded\_image\_logo.decode()*

*),*

*style={"height": "2" "00px"},*

*)*

*],*

*href="https://www.snhu.edu",*

*),*

*html.Img(*

*src="data:image/png;base64,{}".format(encoded\_image\_dog.decode()),*

*style={"height": "2" "00px"},*

*className="align-right",*

*),*

*html.H4(*

*children="Created by Arys Pena",*

*style={"textAlign": "right", "color": "white"},*

*),*

*html.B(*

*html.Center(*

*[*

*html.H1(*

*"Grazioso Salvare Animal Shelter Web Application Dashboard"*

*),*

*html.H3("Web Application Dashboard"),*

*]*

*),*

*style={"color": "white"},*

*),*

*],*

*className="col-6",*

*),*

*],*

*style={"height": "auto", "width": "auto", "backgroundColor": "#0067b9",},*

*),*

*html.Hr(),*

*html.Div(*

*[*

*html.B("Step 1: "),*

*"Select a rescue type from the options below:",*

*html.Br(),*

*dcc.RadioItems(*

*id="radio\_items\_id",*

*options=[*

*{"label": "Water Rescue", "value": "WR"},*

*{"label": "Mountain Rescue", "value": "MR"},*

*{"label": "Disaster Rescue", "value": "DR"},*

*{"label": "Reset", "value": "R"},*

*],*

*# value="R",*

*labelStyle={"display": "inline-block"},*

*),*

*html.Br(),*

*html.B("Step 2: "),*

*"Click on the circle on the left of the row within the table to filter the plots below. Clicking on a row highlights the dog's name in the bar chart.",*

*html.Br(),*

*]*

*),*

*html.Div(*

*[*

*dash\_table.DataTable(*

*id="datatable\_id",*

*columns=[*

*{"name": i, "id": i, "deletable": False, "selectable": True}*

*for i in df.columns*

*],*

*editable=False,*

*filter\_action="native",*

*row\_selectable="single",*

*selected\_columns=[],*

*),*

*html.Br(),*

*html.B("Step 3: "),*

*"Click 'Reset' to display all results (limited to 40 for performance).",*

*]*

*),*

*html.Br(),*

*html.Hr(),*

*html.Div(*

*dbc.Row(*

*[*

*dbc.Col(html.Div(id="datatable\_id\_container"), width=6),*

*dbc.Col(html.Div(id="map\_id"), width=6),*

*],*

*),*

*),*

*]*

*)*

*#############################################*

*# Interaction Between Components / Controller*

*#############################################*

*# This callback will highlight a row on the data table when the user selects it*

*@app.callback(*

*Output("datatable\_id", "style\_data\_conditional"), [Input("datatable\_id", "selected\_columns")]*

*)*

*def update\_styles(selected\_columns):*

*return [{"if": {"column\_id": i}, "background\_color": "#D2F3FF"} for i in selected\_columns]*

*@app.callback(*

*Output("datatable\_id\_container", "children"),*

*[*

*Input("datatable\_id", "derived\_virtual\_data"),*

*Input("datatable\_id", "derived\_virtual\_selected\_rows"),*

*],*

*)*

*def update\_graphs(derived\_virtual\_data, derived\_virtual\_selected\_rows):*

*if derived\_virtual\_selected\_rows is None:*

*derived\_virtual\_selected\_rows = []*

*dff = df if derived\_virtual\_data is None else pd.DataFrame(derived\_virtual\_data)*

*colors = [*

*"#7FDBFF" if i in derived\_virtual\_selected\_rows else "#0074D9" for i in range(len(dff))*

*]*

*return [*

*dcc.Graph(*

*id=column,*

*figure={*

*"data": [{"x": dff["name"], "type": "bar", "marker": {"color": colors},}],*

*"layout": {*

*"xaxis": {"automargin": True},*

*"yaxis": {"automargin": True, "title": {"text": column}},*

*"height": 250,*

*"margin": {"t": 10, "l": 10, "r": 10},*

*},*

*},*

*)*

*# check if column exists*

*for column in ["age\_upon\_outcome\_in\_weeks"]*

*if column in dff*

*]*

*@app.callback(Output("datatable\_id", "data"), [Input("radio\_items\_id", "value")])*

*def update\_datatable(value):*

*if value == "R":*

*df = pd.DataFrame.from\_records(aac.read\_all()).to\_dict("records")*

*print("Reset button pressed")*

*return df*

*if value == "WR":*

*df = pd.DataFrame.from\_records(aac.filter\_water\_rescue())*

*print(f"Filtered to Water Rescue \n {df.head(5)}")*

*return df.to\_dict("records")*

*if value == "MR":*

*df = pd.DataFrame.from\_records(aac.filter\_mountain\_wilderness())*

*print(f"Filtered to Mountain \n {df.head(5)}")*

*return df.to\_dict("records")*

*if value == "DR":*

*df = pd.DataFrame.from\_records(aac.filter\_disaster\_rescue\_tracking())*

*print(f"Filtered to Disaster Rescue \n {df.head(5)}")*

*return df.to\_dict("records")*

*@app.callback(*

*Output("map\_id", "children"),*

*[*

*Input("datatable\_id", "derived\_virtual\_data"),*

*Input("datatable\_id", "derived\_virtual\_selected\_rows"),*

*],*

*)*

*def update\_map(derived\_virtual\_data, selected\_row\_index):*

*dff = df if selected\_row\_index is None else pd.DataFrame(derived\_virtual\_data)*

*if selected\_row\_index is None or selected\_row\_index is None or len(selected\_row\_index) == 0:*

*raise PreventUpdate*

*return [*

*dl.Map(*

*style={"width": "1000px", "height": "500px"},*

*center=[*

*float(dff.iloc[selected\_row\_index, 13].values[0]),*

*float(dff.iloc[selected\_row\_index, 14].values[0]),*

*], # [30.75, -97.48],*

*zoom=10,*

*children=[*

*dl.TileLayer(id=f"base\_layer\_id"), # \_{index}*

*# Marker with tool tip and popup*

*dl.Marker(*

*position=[*

*float(dff.iloc[selected\_row\_index, 13].values[0]),*

*float(dff.iloc[selected\_row\_index, 14].values[0]),*

*],*

*children=[*

*dl.Tooltip(dff.iloc[selected\_row\_index, 4]), # dff.iloc[0, 4]),*

*dl.Popup(*

*[html.H2("Animal Name"), html.P(dff.iloc[selected\_row\_index, 9])]*

*),*

*],*

*),*

*],*

*)*

*]*

*if \_\_name\_\_ == "\_\_main\_\_":*

*app.run\_server(host="0.0.0.0", debug=True)*

**Usage**

The application has filters by using radio buttons to filter the animals by breed, sex, and type of rescue. There is a reset button so if the user clicks this button, this resets all of the filters back to as if the application first started for the first time.

NOTE: I could not get my program to run properly, I have got help from my older cousin over the weekend but I still could not get it fixed. I did it as best as possible. I am lost on how I could fix it.

## Contact

Your name: Antonio Ocasio, Antonio.ocasio@snhu.edu