# Project Two

## About the Project/Project Title

The objective of this project is to code the dashboard and the database interface logic. This will include dashboard attributes. Is user-friendly, intuitive interface and this dashboard reducing user errors and training time.

## Motivation

This project is designed to test our skills at creating a database that is quick to navigate and intuitive. This project helps us maintain app functionality and efficacy for anyone that is using it.

## Getting Started

In the previous README milestone we discuss the process of getting ready, in this Project we don’t have to go in-depth with the process of getting read.

## Installation

In project two there are no installations.

## Usage

Below are code examples of project 2 requirements, what was changed and why. These examples should be clear and understandable to show what has been done to clean the code and create an intuitive interface.

### Code Examples

This code calls the image, it allows the image to display on the page when it is open. With there is my name including the course name

app.layout = html.Div([

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

html.A([

# Add company logo with link to SNHU as well as identifier

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

style={'height': '20%', 'width': '20%'}))]),

html.Center(html.B(html.H1('CS-340 Dashboard - Brandon Porter'))),

html.Hr(),

This code creates an interactive data table, Note this was done in module6 but with listing again

style\_table={'height': '500px', 'overflowY': 'auto'},

style\_cell={'minWidth': '100px', 'width': '100px', 'maxWidth': '100px'},

filter\_action="native",

sort\_action="native",

page\_size=10

),

html.Br(),

html.Hr(),

html.Div(

id='map-id',

className='col s12 m6',

),

html.H1("Code Written and Edited by: Brandon Porter"),

This code sets the geological chart and the pie chart side by side making for a cleaner more user-friendly interface.

html.Div(className='row',

style={'display' : 'flex'},

children=[

html.Div(

id='graph-id',

className='col s12 m6',

style={'width': '50%', 'height': '100%'},

),

html.Div(

id='map-id',

className='col s12 m6',

style={'width': '50%', 'height': '500px'}

)

])

])

This code allows for interacting with filter options on the page.

@app.callback(Output('datatable-id','data'),

Output('datatable-id', 'selected\_rows'),

[Input('filter-type', 'value')])

def update\_dashboard(filter\_type):

## Filter interactive data table with MongoDB queries

#

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

selected\_rows=[0]

if filter\_type == 'default':

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

elif filter\_type == 'Water':

df=pd.DataFrame.from\_records(db.read({

"animal\_type": "Dog",

"sex\_upon\_outcome" : "Intact Female",

"age\_upon\_outcome\_in\_weeks": {"$gte" : 26, "$lte": 156},

"breed": {"$in": ["Labrador Retriever Mix", "Chesapeake Bay Retriever", "Newfoundland"]}

}))

elif filter\_type == 'Mountain':

df=pd.DataFrame.from\_records(db.read({

"animal\_type": "Dog",

"sex\_upon\_outcome" : "Intact Male",

"age\_upon\_outcome\_in\_weeks": {"$gte" : 26, "$lte": 156},

"breed": {"$in": ["German Shepherd", "Alaskan Malamute", "Old English Sheepdog", "Siberian Husky",

"Rottweiler"]}

}))

elif filter\_type == 'Disaster':

df=pd.DataFrame.from\_records(db.read({

"animal\_type": "Dog",

"sex\_upon\_outcome" : "Intact Male",

"age\_upon\_outcome\_in\_weeks": {"$gte" : 20, "$lte": 300},

"breed": {"$in": ["Doberman Pinscher", "German Shepherd", "Golden Retriever", "Bloodhound, Rottweiler"]}

}))

df.drop(columns=['\_id'],inplace=True)

return df.to\_dict('records'), selected\_rows

### ScreenshotsA screenshot of a computer Description automatically generatedA screenshot of a computer Description automatically generatedA screenshot of a computer Description automatically generatedA screenshot of a computer Description automatically generatedA screenshot of a computer Description automatically generatedA screenshot of a computer Description automatically generated

* **Describe the required functionality.**

As seen in the screenshots and in the code, the functions added where a filtering option that allows the user to navigate quicker through what they are looking for in breed, age, sex and so on. The geological map allows for us to know where the location is and the pie chat allows us to know the percentages of the animals in certain categories as well as in general. To make the interface cleaner the map and the pie chart were put side by side. The identifiers where added. To show proof and credit to the individual who completed the project. The logo was also added as a nice touch to the code that was required but brings the interface together.

* **Describe the tools used to achieve this functionality and a rationale for why these tools were used.**

Mongo allows for cleaning editing and guiding through a database. Adding users, deleting users and editing different features. Without mongo the process would be extremely difficult and less than acceptable.

Dash allows you to build web applications entirely in Python, including both the frontend layout and the backend logic. It follows a component-based architecture, where you can define layouts and interactions using Python syntax and integrate charts seamlessly.

* **Explain the steps that were taken to complete the project.**

The main step is plan accordingly. Not everything is going to be quick and easy and for that, having a planned strategy is needed. Going through each step of a fix me and having good resources and support from others is my next main step. After that it is step at a time process.

* **Identify any challenges that were encountered and explain how those challenges**

In this code there where a lot of challenges and reaching out to other students. I was struggling with the image coming through and the pie graph working properly. After some help and support I was able to get that fix and as suspected it was a minor issue with the code. The radio buttons and filter option were not working for a long time, That required help from another student that explained I was missing code entirely and they helped me code through that problem to create a nice, clean and friendly user interface!

## Contact

Your name: Brandon Porter