# CS 340 README Project 2

## About the Project/Project Title

*The AAC Dashboard project aims to create an interactive web-based dashboard for analyzing animal data from an animal shelter. This project utilizes Python, MongoDB, and the Dash framework to provide users with a user-friendly interface for exploring and visualizing animal data.*

## Motivation

*The motivation behind the AAC Dashboard project is to provide Grazioso Salvare with a powerful tool for analyzing and visualizing their animal data. This dashboard will allow them to make informed decisions about animal rescue and adoption strategies based on data-driven insights.*

## Getting Started

*Set up Mongo database with the Grazioso’s csv that they have given you.*

*Authenticate user in our case is ‘accuser’*

*Create a .py for all your create, read, update and delete (CRUD)*

## Installation

*The following tools are required to use the AAC Dashboard:*

*Python*

*MongoDB*

*To install the required Python packages*

*Dash*

*Plotly*

*Pandas*

*JSON*

## Usage

*The AAC Dashboard provides the following functionality:*

*Interactive data filtering using dropdown menus*

*Display of data in a dynamic DataTable*

*Visualization of data using Pie Charts*

*Geolocation mapping of selected animal records*

### Code Example

*# Import necessary libraries*

*import dash*

*import dash\_core\_components as dcc*

*import dash\_html\_components as html*

*# Create Dash app*

*app = dash.Dash(\_\_name\_\_)*

*# Define app layout*

*app.layout = html.Div([*

*dcc.Dropdown(*

*id='my-dropdown',*

*options=[*

*{'label': 'Option 1', 'value': 'option1'},*

*{'label': 'Option 2', 'value': 'option2'}*

*],*

*value='option1'*

*),*

*html.Div(id='output')*

*])*

*# Define callback function*

*@app.callback(*

*dash.dependencies.Output('output', 'children'),*

*[dash.dependencies.Input('my-dropdown', 'value')]*

*)*

*def update\_output(value):*

*return f'You have selected: {value}'*

app.run\_server()

### Tests

*If you look closely at the the def map section, you can see that I have this printing out the index to ensure each time I select an animal row, the map is infact moving.*

def update\_map(viewData, index):

print('index: ' + str(index))

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

# Because we only allow single row selection, the list can

# be converted to a row index here

if index is None:

row = 0

else:

row = index[0]

print('row: ' + str(row))

A screen shot of a computer

Description automatically generated

### Screenshots

A screenshot of a computer

Description automatically generated

A screenshot of a graph

Description automatically generated

A pie chart with different colored circles

Description automatically generated

## Contact

Sarah Schmidt