# CS 340 README – Grazioso Salvare Dashboard

## About the Project/Project Title

This project is about creating a dashboard for the client, Grazioso Salvare. The purpose of this dashboard is to work with existing data from animal shelters to identify and categorize available dogs. The dashboard is web-based and will use the tools Mongo DB, Python, and a Dashboard Framework while implementing our CRUD operations.

## Motivation

The motivation behind this project is to create an interactive dashboard that the users in the company can utilize to allow them to access the database. This dashboard will allow users to filter results, provide a type of graph that displays good candidate breeds for said rescue type, and see the geolocation of said rescue animal.

## Getting Started

Assuming MongoDB is already installed, you will need to enter MongoDB and import a csv file. For this project, we used the file “aac\_shleter\_outcomes.csv. After a successful import, we can move to a notebook, like Jupyter, to run and test our program with the appropriate connection variables. These connection variables include, USER, PASS, HOST, PORT, DB, and COL. These should reflect your connection variables that you can access from MongoDB and will be in your program code.

I suggest you create a folder to maintain your .py file, an .ipynb file and an optional image that can represent your company (i.e. a logo).

## Installation

MongoDB shell like mongosh to connect to the MongoDB

Current Version of Python installed

A python framework that can build our interactive dashboard. For this project, we are the framework, Dash. To install, you can access your command prompt or terminal and type in pip install dash. In addition, you need to install two more components for Dash, which is pip install pandas and pip install dash-daq. Once these are installed, you will also have access to the Plotly library that can assist you with graphing the data.

A screenshot of a computer

AI-generated content may be incorrect.

An interactive notebook that has the current version of python installed and can run a .py file and a .ipynb file. For example, to install Jupyter, you can access your command prompt or terminal and type in pip install notebook, press enter. To launch the notebook, you will type jupyter notebook. We will also test and launch our dashboard from the notebook as well.

*A screenshot of a computer code

AI-generated content may be incorrect.*

## Usage

*Use this space to show useful examples of how your project works and how it can be used. Be sure to include examples of your code, tests, and screenshots.*

### Code Example

*Show what the library does as concisely as possible. Developers should be able to figure out how your project solves their problem by looking at the code example. Make sure that your code is short and concise.*    elif filter\_type =='Disaster':

         df = pd.DataFrame.from\_records(shelter.read( #$and performs and, $in performs in, $or for or, $lte less than or equal, $gte viceversa

                   { "$and":[

                       {"$or": [

                           {'breed':{"$regex" : 'Doberman Pinscher'}}, #$regex for matching strings in searches

                           {'breed':{"$regex" : 'German Shepard'}},

                           {'breed':{"$regex" : 'Golden Retriever'}},

                           {'breed':{"$regex" : 'Bloodhound'}},

                           {'breed':{"$regex" : 'Rottweiler'}}]},

                       {'sex\_upon\_outcome' : 'Intact Male'},

                       { "$and" :[

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

                           {'age\_upon\_outcome\_in\_weeks': {"$lte" : 300}}

                       ]}

                   ]}))

This is an excerpt of the program that is in the. ipynb file. This shows that if a user chooses to filter the data table by Mountain or Wilderness Rescue, then the following breeds if available in that location will appear. This will interact with the chart and map as well.

### Tests

To run a test, you can create a code layout in the .ipynb file. You should import the correct modules, libraries and your .py file that has your CRUD module. You should include your connection variables, access to connect and ability to read your documents. Right here you can debug to ensure that you have connected to your database. Next, you can create a dashboard layout and datatable that will appear on the dashboard. From here you can add features and filters that you would like to be implemented. Afterwards, you can create interactions for the features and filters with app.callbacks. Once you have everything together, you can end your program with app.run\_server(debug=True), this will allow you to start a local server to your application. You can check if your filters are interacting with the appropriate callbacks and are updating your table, chart, and graph.

### Screenshots

Water Rescue

A screenshot of a computer screen

AI-generated content may be incorrect.

Mountain Rescue

A screenshot of a map

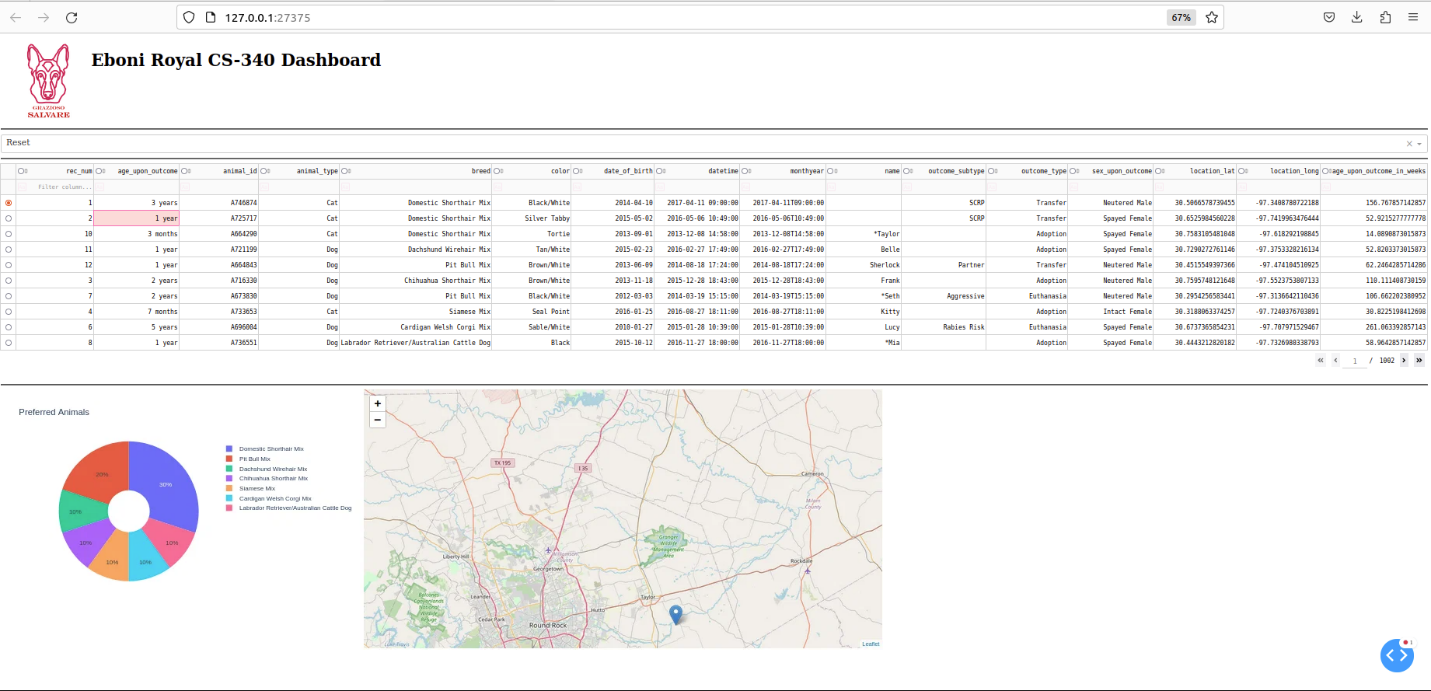
AI-generated content may be incorrect.

Disaster or Individual Rescue

A screenshot of a computer screen

AI-generated content may be incorrect.

Reset



## Roadmap/Features (Optional)

No known issues as of now. Keep in mind to troubleshoot and check clear syntax and correct indentation. If a variable is off, even for just a tad bit, it can cause your application not to function properly.

## Contact

Your name: Eboni Royal