# CS 340 README: Grazioso Salvare

## About the Project/Project Title

Grazioso Salvare identifies dogs that are good candidates for search-and-rescue training. When trained, these dogs are able to find and help rescue humans or other animals, often in life-threatening conditions. To facilitate the needs of the company this project implements interaction and functionality for client and server interactions. Specifically, identify and filter through existing animal shelter data.

**Tools**

**Database -> MongoDB (**[**https://www.mongodb.com/**](https://www.mongodb.com/)**)**

MongoDB is built on a scale-out architecture for developing scalable applications with evolving data schemas.

**Middleware -> Python (**[**https://www.python.org/**](https://www.python.org/)**)**

**Client Dashboard -> Jupyter Notebook (**[**https://jupyter.org/**](https://jupyter.org/)**)**

Jupyter Notebook is used to create interactive notebook documents that can contain live code, equations, visualizations, media, and other computational outputs.

**Interface Table, Map, Pie Chart -> Dash & Plotly (**[**https://dash.plotly.com/**](https://dash.plotly.com/)**)**

Plotly is a graphing and analytics library that is functional with Python, among many other languages. Dash is a Python framework that is used for building web-based applications. Combined, these tools offer graphing and widgets within a web-based dashboard.

**Project Setup**

**Database**

The first step is to set up users for authentication:

*db.createUser ({*

*user: "aacuser",*

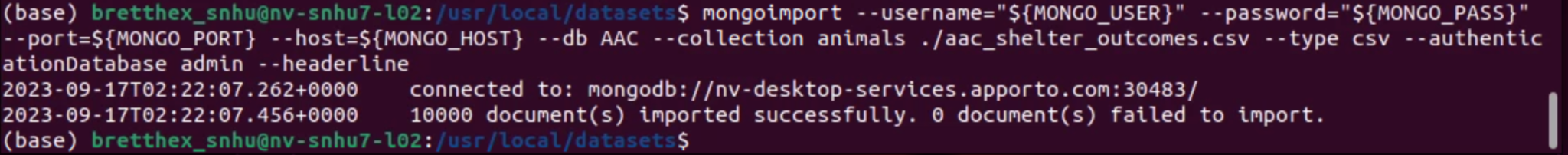
*pwd: passwordPrompt(), // This will create a prompt for entering the password*

*roles: [{ role: "readWrite", db: "AAC" }]*

*})*

The next step is to import the .csv file into the database:

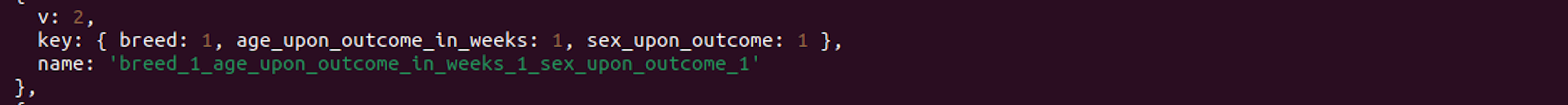
*mongoimport --port “port number” --db AAC --collection animals ./aac\_shelter\_outcomes.csv*



After importing the .csv file we have to set up the indexes that are used by the dashboard:

*db.animals.createIndex( {breed:1, age\_upon\_outcome\_in\_weeks: 1, sex*

*-upon\_outcome1} )*



**CRUD Functionality**

Object used to connect to MongoDB:

| **class** **AnimalShelter**(object):  **def** **\_\_init\_\_**(self, username, password):  **try**:  self.client = MongoClient('mongodb://%s:%s@nv-desktop-services.apporto.com:30483' % (username, password))  self.database = self.client['AAC']  self.collection = self.database['animals']    print("Initialization successful")  **except** Exception **as** e:  print("An error occurred during initialization: ", e) |
| --- |

Create:

@param - newDoc User inputs new animal information in dict form

| # This method is used to create and add documents  **def** **create**(self, newDoc = None):  # Check for document information  **if** Newdata **is** **None**:  **raise** Exception("Nothing to save, because data parameter is empty")    # Attempt to create new document   newDocAdded = self.database.animals.insert(data) # data should be dictionary  **if** newDocAdded **is** **None**:  **raise** Exception('Failed to add new document check data.') |
| --- |

Read:

@param - findData receives index {‘animal\_type”: “Dog”}

| # This method is used to create and add documents  **def** **create**(self, newDoc = None):  # Check for document information  **if** Newdata **is** **None**:  **raise** Exception("Nothing to save, because data parameter is empty")    # Attempt to create new document   newDocAdded = self.database.animals.insert(data) # data should be dictionary  **if** newDocAdded **is** **None**:  **raise** Exception('Failed to add new document check data.') |
| --- |

Update:

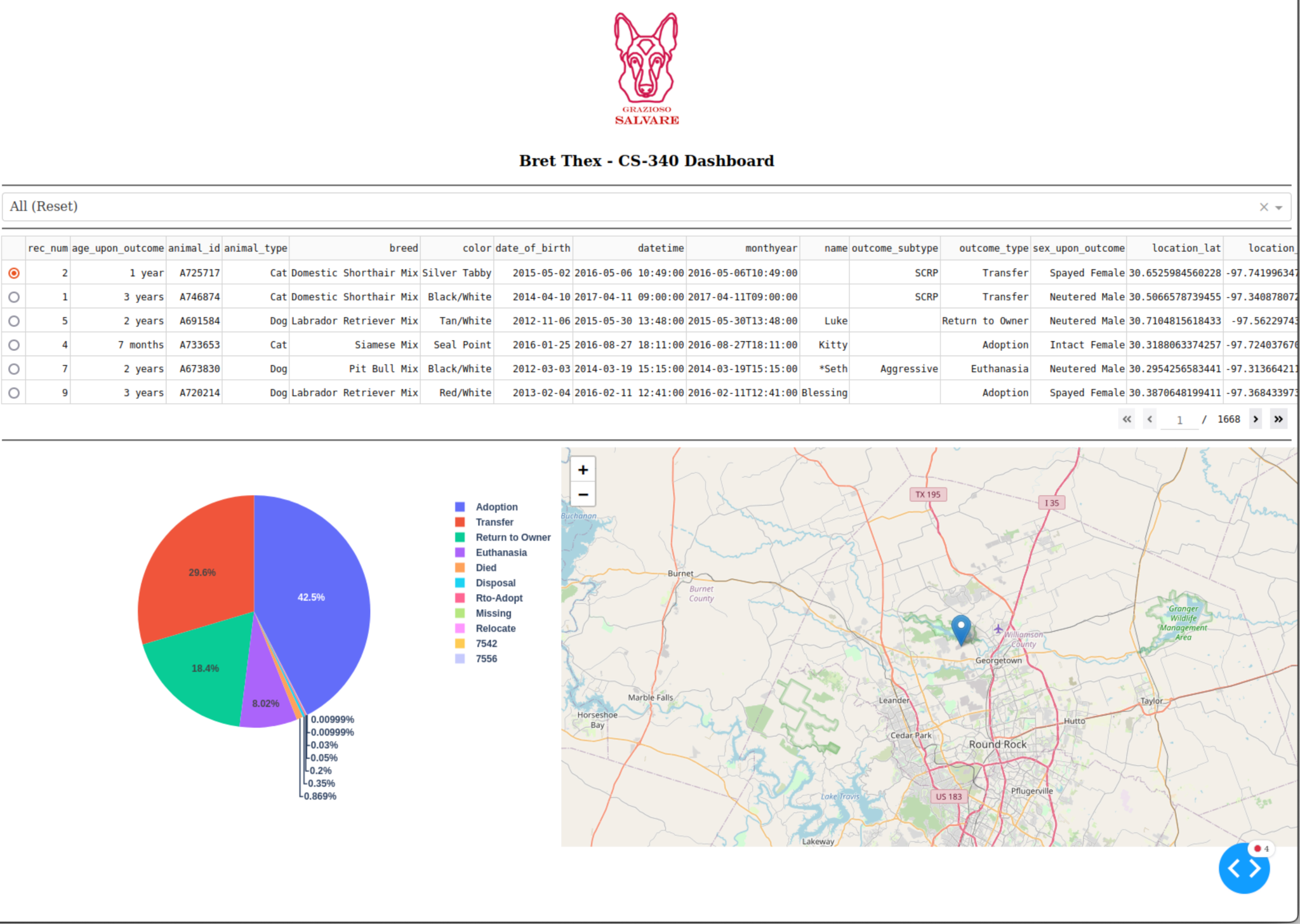
@param - filter receives index that you want to update {“breed”: “”}

@param - update receives index {“breed”: “”}

| # This method is used to delete documents  **def** **delete**(self, removeFilter = None):  # Check for a valid filter  **if** removeFilter **is** **None**:  **raise** Exception('\nERROR: No valid filter was specified.')    #Delete records basd on filter  deleted = this.db.animals.delete\_many(removeFilter)  **if** deleted **is** **None**:  **raise** Exception('\nERROR: Failed to delete records based on filter.') |
| --- |

**Functionality**

**No Filter**

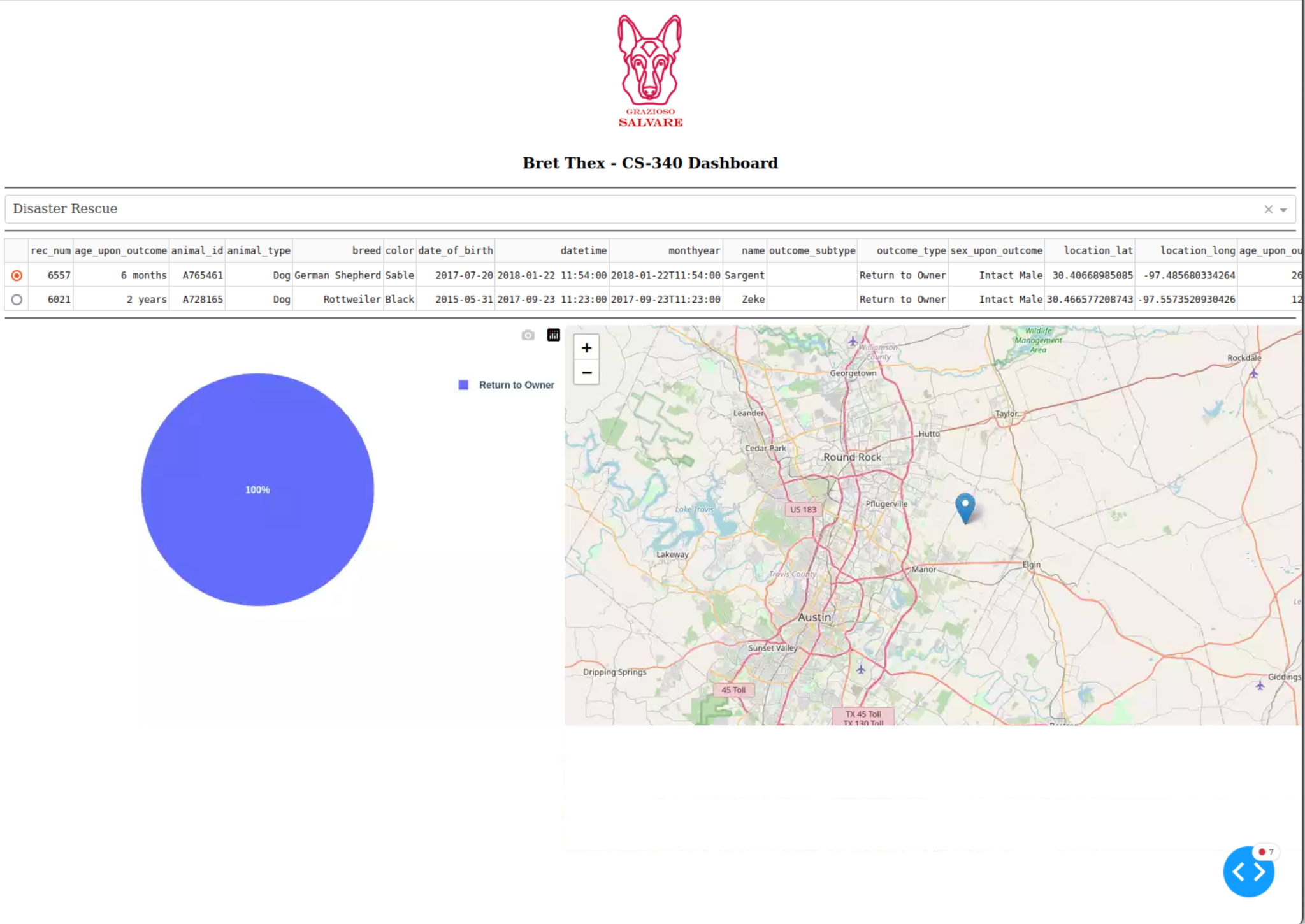
****

**Water Rescue**

## 

## Mountain Rescue

**Disaster Rescue**



## Contact

Your name: Bret Thex

bret.thex@snhu.edu