# CS 340 README

## About the Project/Project Title

The purpose of this project is to develop software tools for an animal rescue organization so that they can connect to partnered animal shelters to find suitable dogs to be trained to become rescue dogs.

## Motivation

This project exists to be an open-source software application to connect with the data of existing animal shelters and find suitable dogs to be trained as rescue dogs*.* MongoDB was chosen to be used when building the database because of the easy-to-use integration tools with Python.

## Getting Started

Ensure that a MongoDB instance exists on the animal shelter’s server, and that you have the proper user credentials to access the desired collection.

Ensure that crud.py is in your PYTHONPATH and that the MongoDB connection port is correct in the file.

Ensure that the user login credentials are correct under the Data Manipulation/Model.

## Installation

To start, make sure that python 3.6 or newer is installed. If not, it can be downloaded and installed from <https://python.org/downloads>. Once that step is complete, [install Jupyter Notebook.](https://jupyter.org/install) Install the [pymongo](https://pymongo.readthedocs.io/en/stable/installation.html) package. Once Python is installed, multiple packages will need to be installed through the Command Line. Each command is formatted like pip install [package].

[package] will be replaced with:

* dash
* dash\_leaflet
* dash\_core\_components
* dash\_html\_components
* dash\_table
* dash.dependencies
* plotly.express
* jupyter\_plotly\_dash
* numpy
* panda

## Usage

### Dashboard Components

Header and logo: The unique header and logo can be changed depending on need through the Jupyter Notebook file.

Data Table: Currently the data table contains unfiltered data from the ‘animals’ collection, including all fields which are shown as the columns of the table.

Radio Buttons: The four radio buttons above the data table are used for filtering the animals shown on the table and the lower graph.

Chart: An interactive chart that shows a histogram of the number of each breed on the current page of the data table. This chart will update as the shown columns of the data table change.

Map: An interactive map that shows information on the animal on the first row of the data table.

**Filtering**

The four current filter options and parameters are:

Water Rescue: Dogs, Intact Female, Training Age between 26 and 156 weeks, and with a preferred breed of Labrador Retriever Mix, Chesapeake Bay Retriever, or Newfoundland.

Mountain or Wilderness Rescue: Dogs, Intact Male, Training Age between 26 and 156 weeks, and with a preferred breed of German Shepherd, Alaskan Malamute, Old English Sheepdog, Siberian Husky, or Rottweiler

Disaster or Individual Tracking: Dogs, Intact Male, Training Age between 20 and 300 weeks, and with a preferred breed of Doberman Pinscher, German Shepherd, Golden Retriever, Bloodhound, or Rottweiler

Reset: Shows the unfiltered data from the database

### Methods

create: enters a new document into the database. Returns True on success, False otherwise

read: queries the database for documents matching the input. Returns a cursor

update: modifies matching documents in the database. Returns MongoDB JASON response

delete: removes matching documents from the database. Returns MongoDB JASON response

### Code Example

**To create instance of a database client:**

client = AnimalShelter('aacuser')

**To insert a new entity:**

client.create({"animal\_type": "something"})

**To query the animals collection:**

client.read({"animal\_type": "something"})

**To update the animals collection:**

client.update({"animal\_type":"something"},{"animal\_type":"dog"})

**To delete entities from the animals collection:**

client.delete({"animal\_type":"dog"})

### Tests

>>> client = AnimalShelter('aacuser')

>>> client.create({"animal\_type": "something"})

True

>>> [\*client.read({"animal\_type": "something"})]

[{'\_id': ObjectId('62992f7a4a900a9fdbf7921a'), 'animal\_type': 'something'}]

>>> client.update({"animal\_type":"something"},{"animal\_type":"dog"})

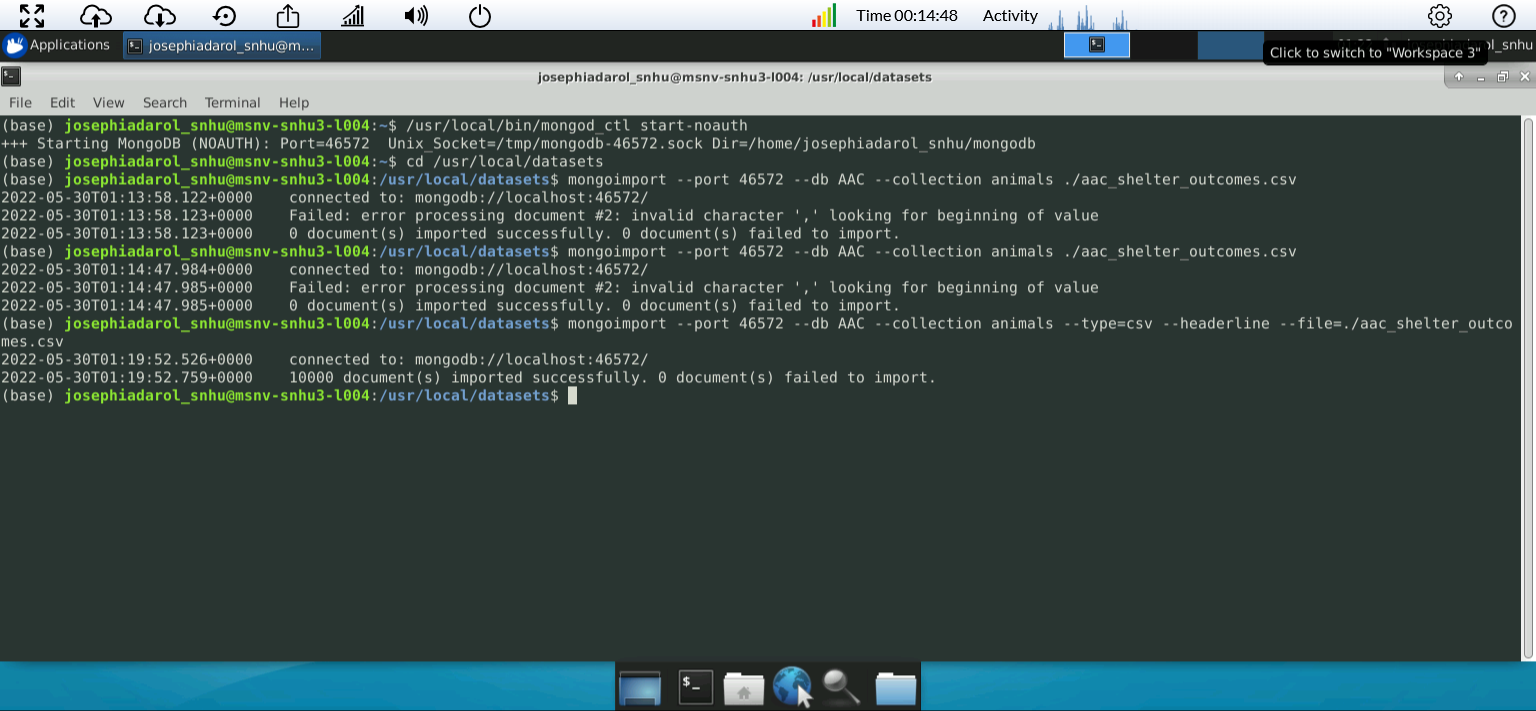
{'n': 1, 'nModified': 1, 'ok': 1.0, 'updatedExisting': True}

>>> client.delete({"animal\_type":"dog"})

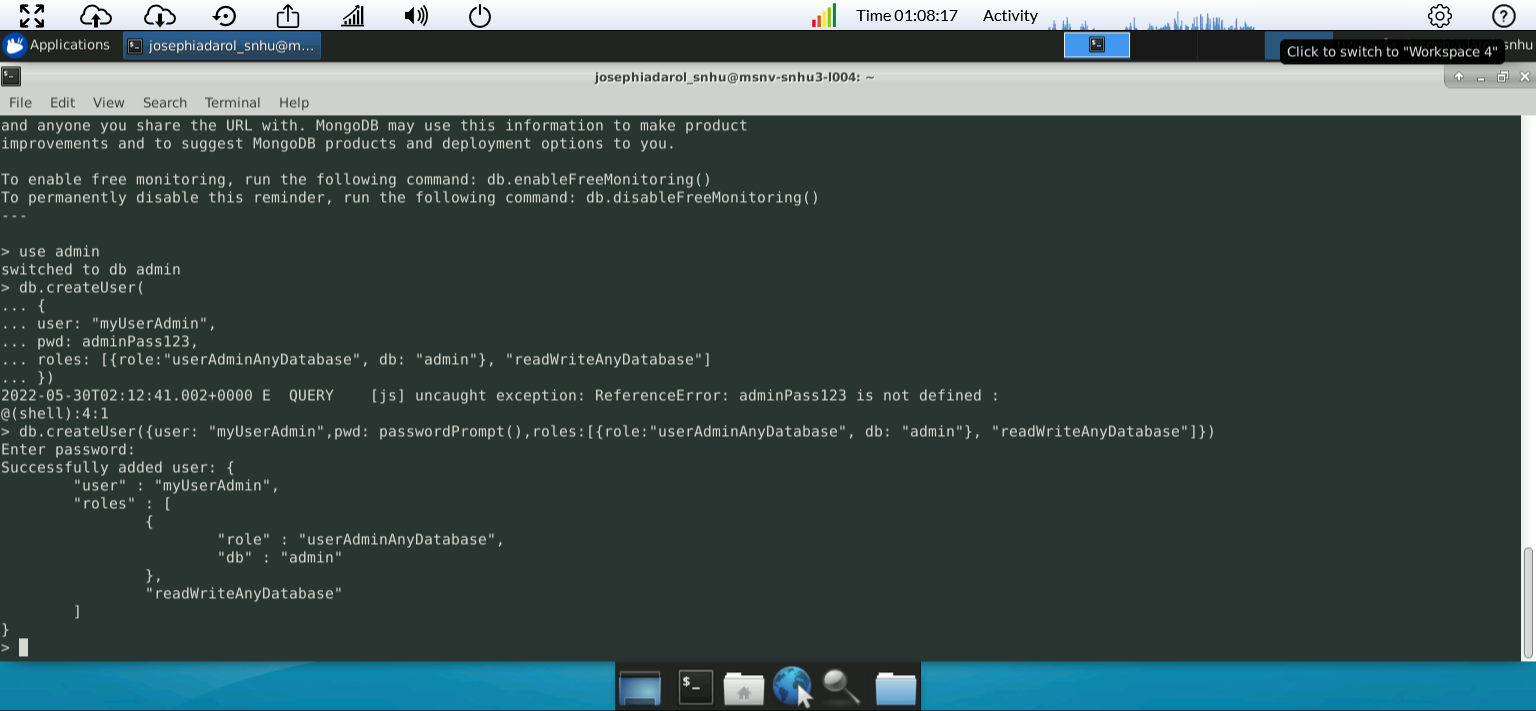
{'n': 1, 'ok': 1.0}

### Screenshots

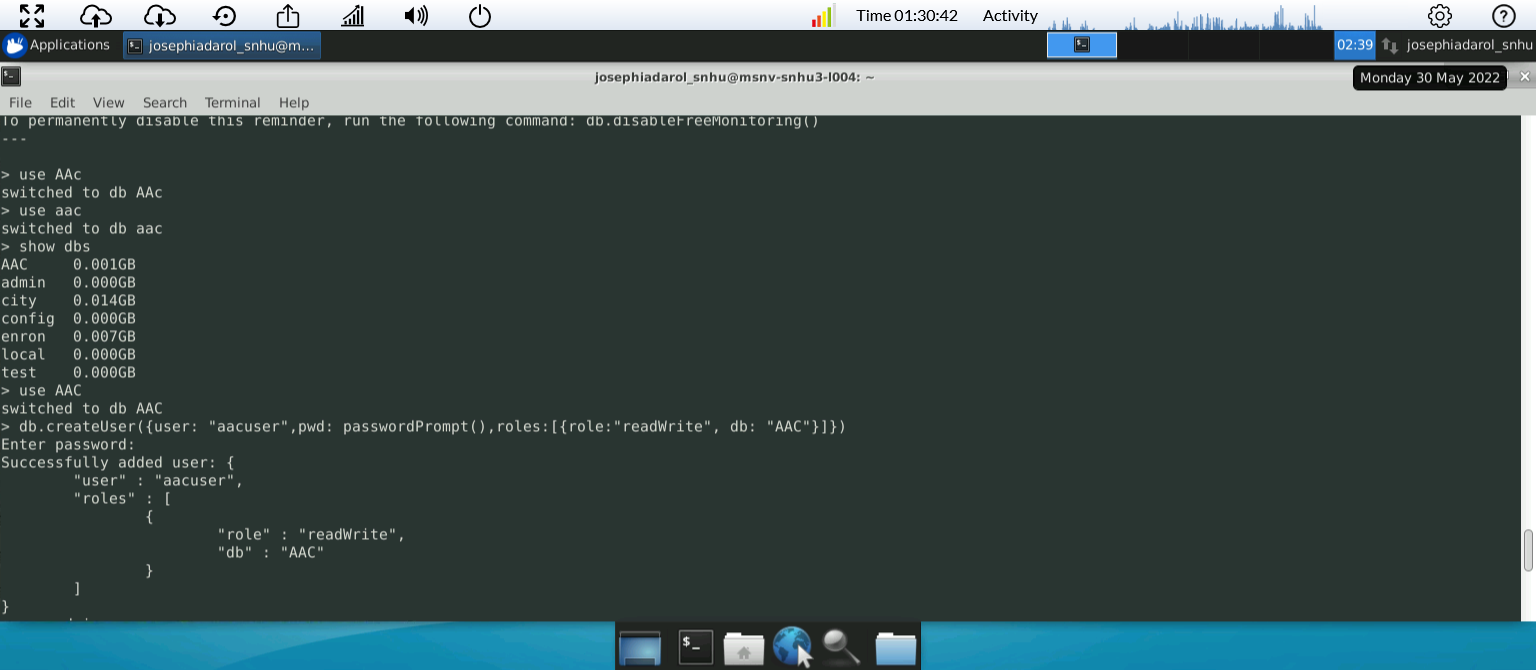
MongoDB import execution



Creating admin account

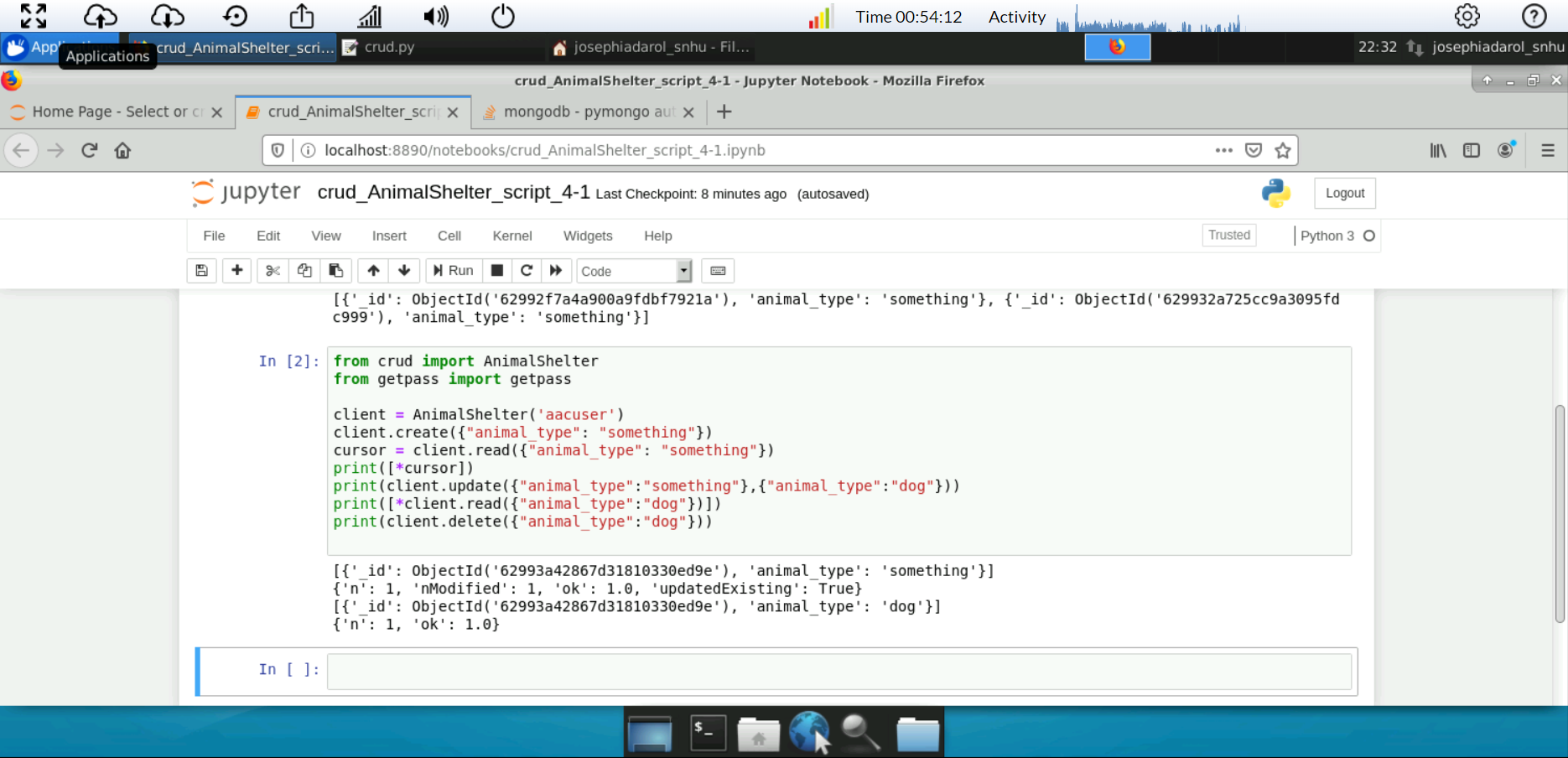


Creating aacuser account

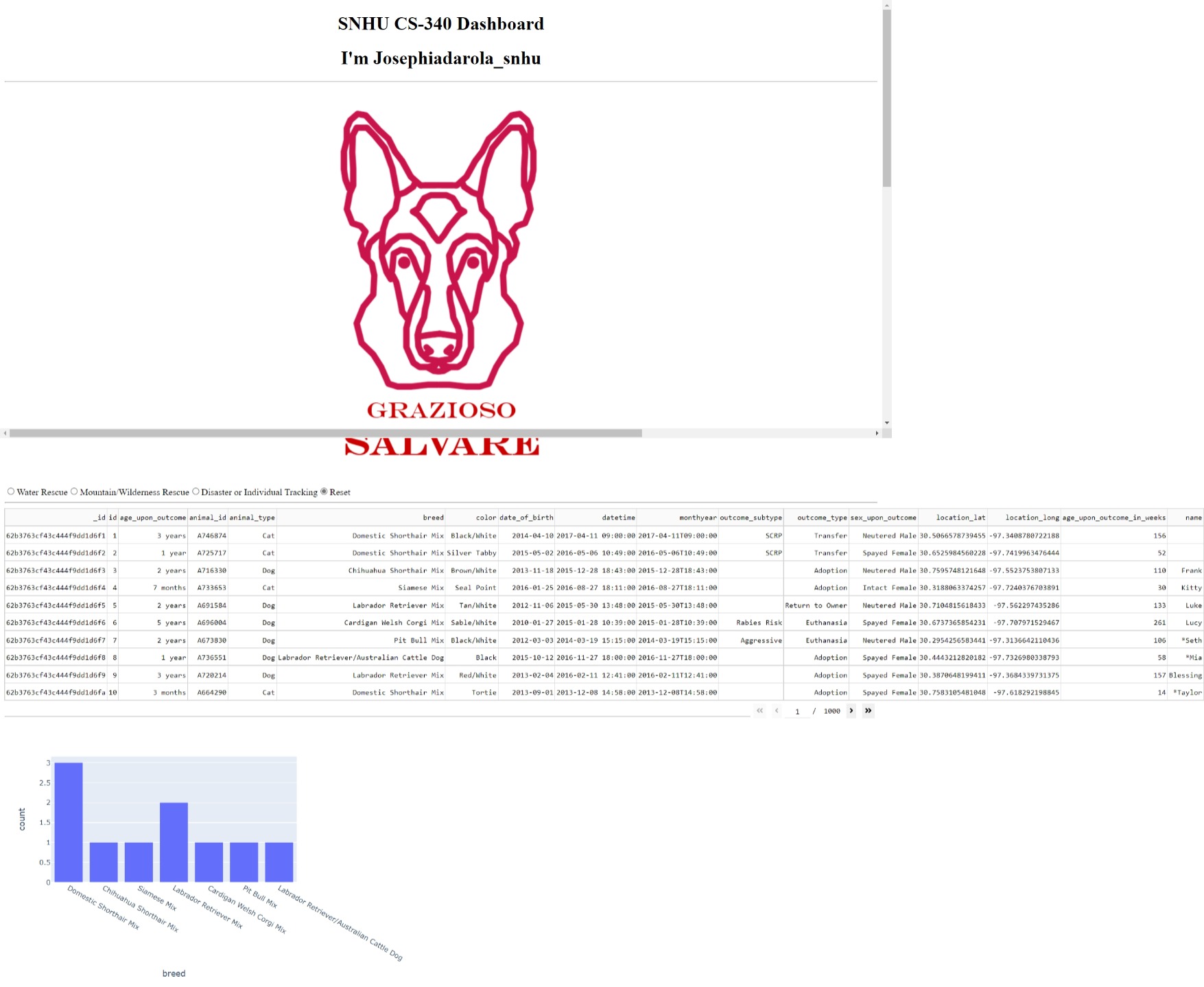


**Example Code Execution**

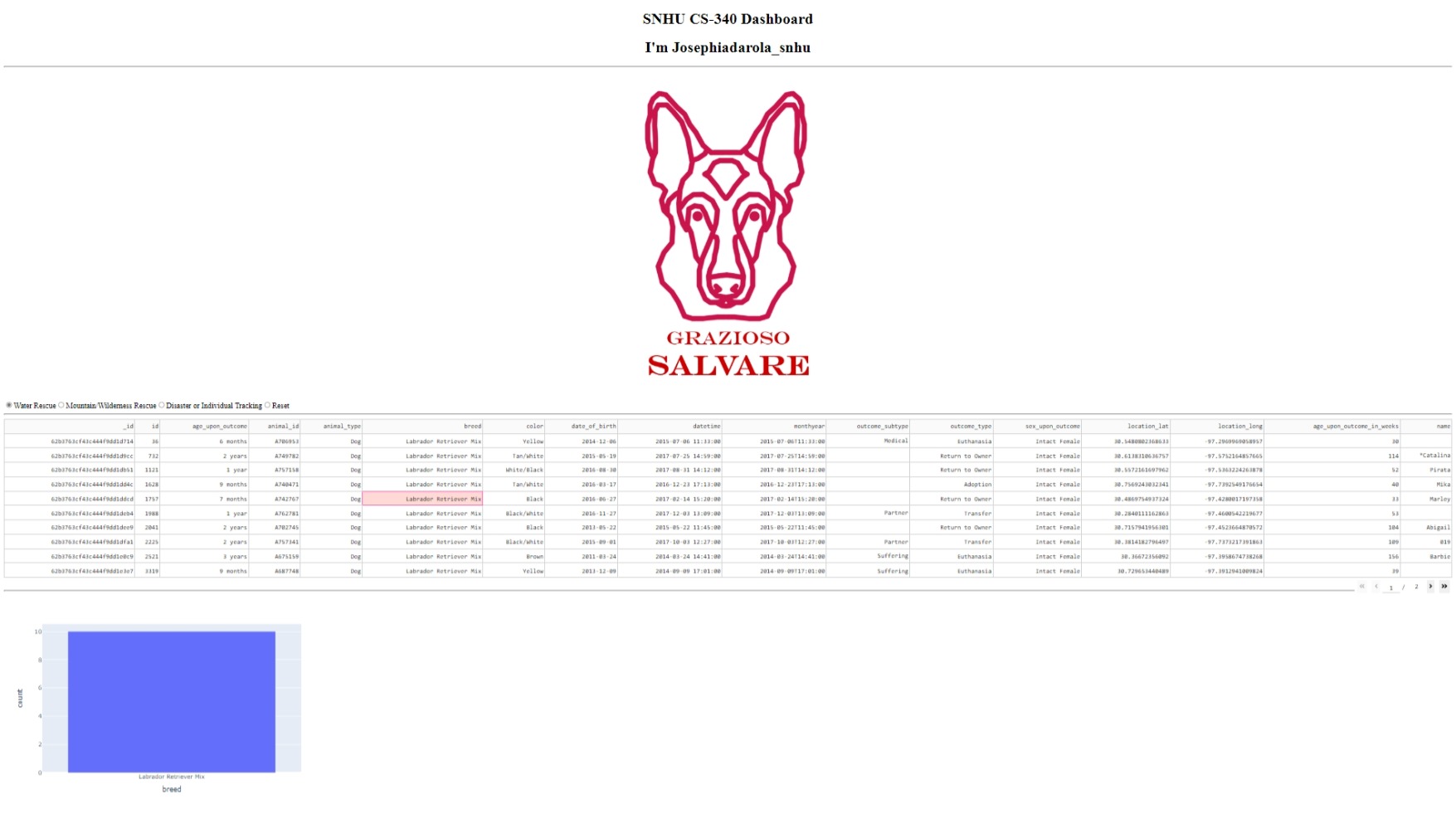
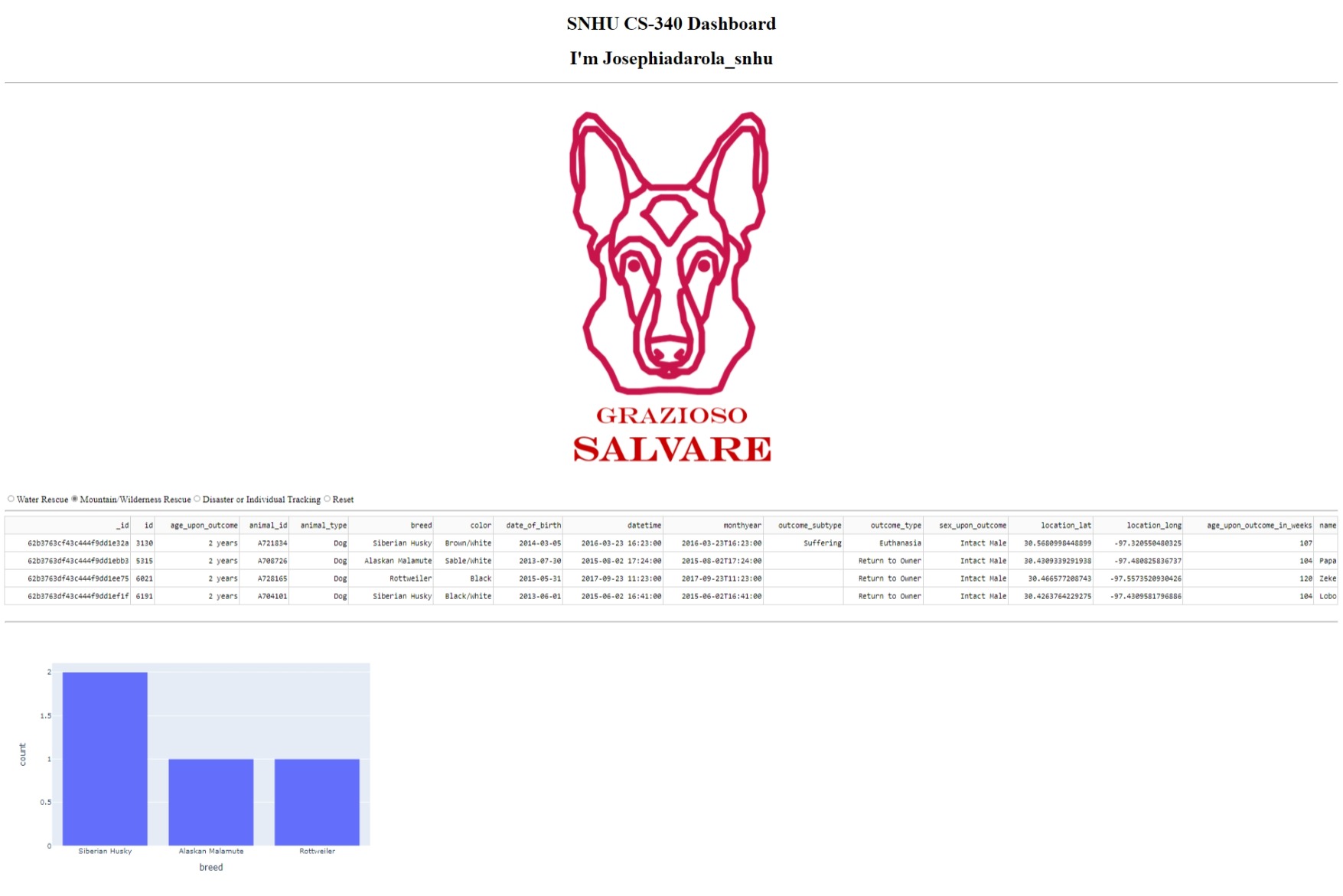
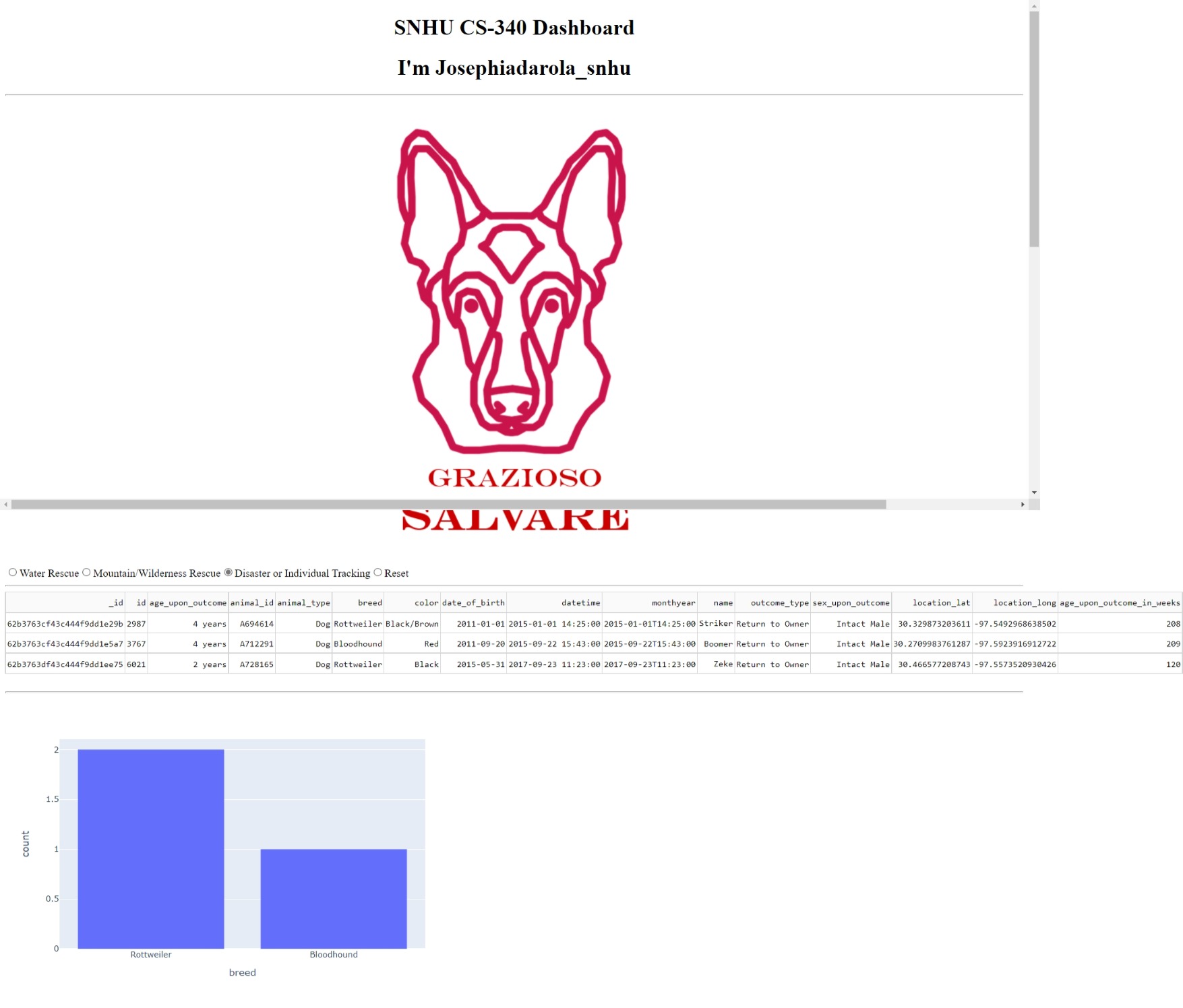
These are test queries with no dashboard application



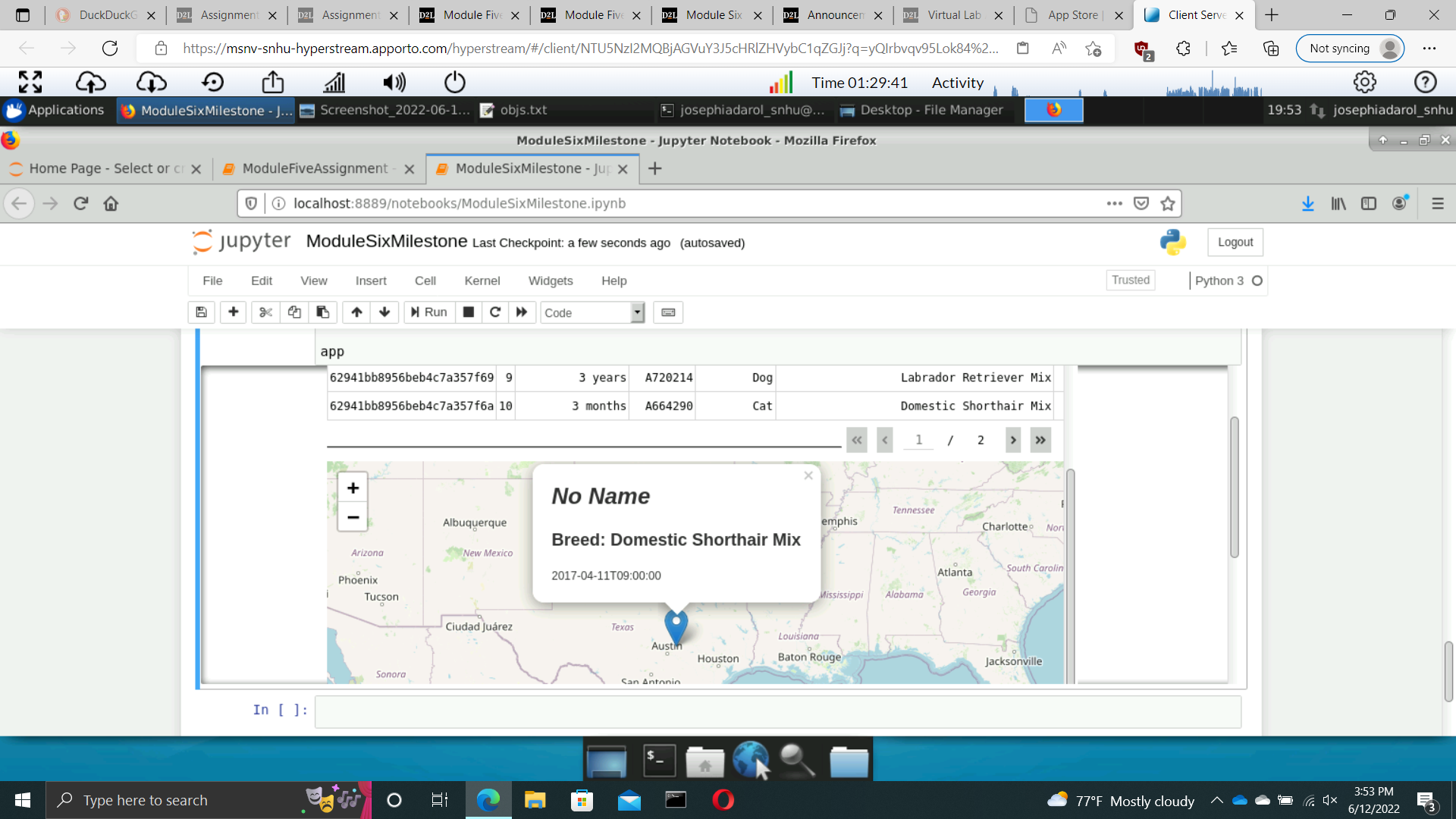
Starting dashboard application (minus interactive map)



Dashboard when filtered



**When using the dashboard on Windows there is a possibility that the interactive map will not load.**



## Contact

Joseph Iadarola