# Project Two Read Me

## About the Project/Project Title

This project is called “Search-And-Rescue Training.” We designed this application for Grazioso Salvare, where we help identifity dogs who are good candidates for search and rescue training. Our app can help identify specific dogs who would be more equipped for certain types of search and rescue, as well as filters the results by different specifications. The user should have a better idea of potential search and rescue candidate dogs after using our application.

## Motivation

The intention of this project was to help users easily identify dogs that are search and rescue candidates.

## Getting Started

To use our application, follow these steps –

1. Import Austin Animal Center Database
2. Create admin user with Read and Write privelages for database
3. Run the “main.ipynb” file
4. Edit the file to insert your desired pets to the database.
5. Run the “main.ipynb” script
6. User will be displayed the dashboard where they can filter animals, see interactive chart, and geolocate animals.

## Installation

The user needs access to MongoDB and Python, and the user needs to import the Austin Animal Center Database prior to running the application, as well as creating admins with efficient read and write rights.

Graphical user interface, text, application, email

Description automatically generated

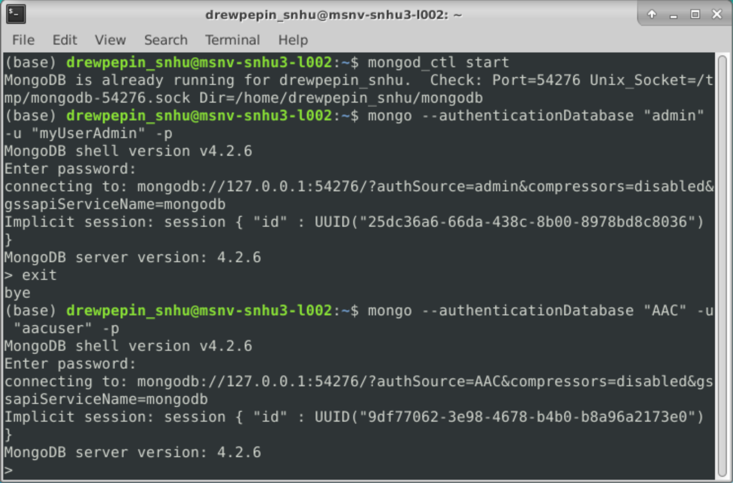
## Usage

First, you must import the database.

Graphical user interface, text

Description automatically generated

Then, you must create a user “aacuser” with proper read and write privelages for AAC database.



**Running Application**

Upon running the application, the user will be presented with this dashboard.

Graphical user interface, application, table

Description automatically generated

### Code Example

Animal\_shelter.py is where the Create and Read objects are held.

#create

def create(self, data):

if data is not None:

insert = self.database.animals.insert(data)

if insert!=0:

return True

else:

return False

else:

raise Exception("Nothing to save, paramter empty."

#read

def read(self, criteria=None):

if criteria:

data = self.database.animals.find(criteria,{"\_id":False})

else:

data = self.database.animals.find({}, {"\_id":False})

return data

#update

def update(self, query, data):

if data is not None:

data\_update = self.database.animals.update\_one(query, data)

else:

raise exception(“Nothing to save”)

#delete

def delete(self, data):

if data is not None:

data\_delete = self.database.animals.delete\_one(data)

else:

raise exception(“Nothing to delete”)

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

### Tests

Main.ipynb is where the tests are held.

for i in animal\_data:

a.create(i)

dogs = a.read({"type": "dog"})

for dog in dogs:

print(dog)

a.update(query, newname)

a.delete({“name”: “pup”})

### Screenshots

Create and Read

Graphical user interface, text, application

Description automatically generated

Text

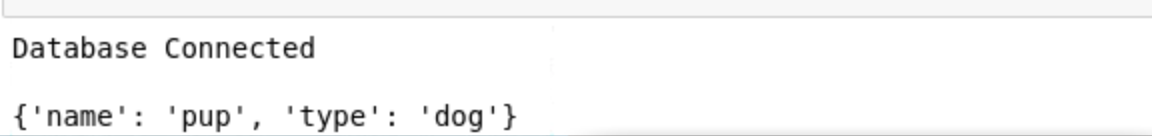
Description automatically generated

Update and Delete

We updated a dog’s name to ‘pup’, and then deleted that dog.

Shape

Description automatically generated



Graphical user interface, text, application, email

Description automatically generated

## Roadmap/Features

Drop Down Filter

Graphical user interface, application

Description automatically generated

Graphical user interface, application, table

Description automatically generated

Breed Chart

## 

Geolocator

## 

## Contact

Your name: Drew Pepin