## Grazioso Salvare Rescue-Animal Dashboard

**A screenshot of a map

Description automatically generated**

## About the Project

This project is centered around Grazioso Salvare, a rescue-animal training company. It is designed to find and categorize dogs to from animal shelters across Austin, Texas. This software uses existing data from the animal shelters and allows the user to create, read, update, and delete from the database. It will then allow the user to filter through the information and find animal specific information.

## Motivation

Grazioso Salvare trains animals to save people and other animals’ lives, and this project aims to support that process by making it easier and faster to find trainee candidates.

## Getting Started -

**Database with User Authentication:**

I have set up a MongoDB database that holds existing information from 5 shelters in the Austin, Texas area. User authentication is implemented to only allow secure access to specific databases. This was implemented using basic authentication by creating a user within MongoDB and granting them specific access permissions to a database.

**Why use a CRUD Module?**

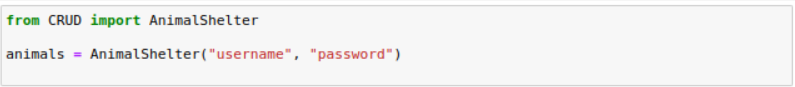
The goal is to give authorized users thorough interaction with the database. The CRUD module allows that functionality by allows full modification and accessibility to the database through various functions. The PyMongo driver for MongoDB and Python is used for this project. It is chosen because it is versatile and easy to use.

**CRUD Operations:**

All CRUD functionalities are built into the ‘AnimalShelter’ Python class.

**Import**

First, Import the ‘AnimalShelter’ class and initialize an instance using your MongoDB username and password.



**Create Method (C)**

The create method allows users to add new entries into the MongoDB database, with details in the form of dictionary entries. It requires a data parameter which is a dictionary representing the data to be inserted. It returns *True* if successful and *False* if not.



**Read Method (R)**

The read method allows users to submit a query to the database to find animals that meet specific criteria. It requires a search parameter which is a dictionary representing the data to be found. It returns a list of matching records.



**Update Method (U)**

The update method allows users to submit a query to the database to find a record of an animal and update. It requires a search parameter which is a dictionary representing the data to be found, and an update query, specifying the information to be updated. It returns a string representing the number of updated records.



**Delete Method (D)**

The delete method allows removal of records. It requires a dictionary parameter representing what is to be deleted. It returns a string representing the number of deleted records.



All methods implement data validation and error handling to ensure secure functionality.

**Dashboard Functionality**

A dashboard is incorporated to show a graphical interface for the users. It utilizes Dash to provide filtering based on rescue type and visualized data. The dashboard is designed following a Model-View-Controller pattern to ensure it is modular and user-focused.

**Data Manipulation – Model:**

The data of the application uses MongoDB to store records of different animals in shelters in the vicinity. To prepare the data for visualizations, we connect to the MongoDB database and transform it into a Pandas data frame. The data frame is the backbone of the data table and two graphs used in the dashboard.

**Dashboard Layout – View:**

The dashboard layout is created using Dash components and HTML, to make sure the data is visually appealing. Users can filter information based on the type of rescue animal they are trying to find.

**Component Interaction – Controller:**

The controller is where Pandas and Dash’s callback functions operate. These components are the *glue* that allows interaction between the model (MongoDB database) and the view (Dash Dashboard). Callback functions are used to update the dashboard in real-time.

**Challenges and Solutions:**

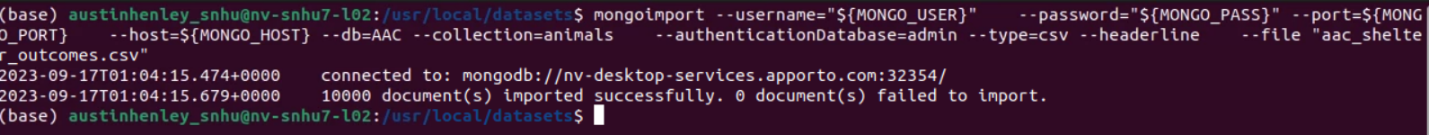
While developing the CRUD functionality, the biggest challenges are data validation and error reporting. Checks were implemented to ensure only valid data is stored and entered by the user. There is also ample exception reporting within the class, focused on giving the user information on what went wrong.

Another challenge was in the creation of the dashboard. Using appropriate queries are extremely important, and it took several bouts of testing to ensure that user requirements were met.

**Steps to reproduce:**

1. Import database into MongoDB: This project used existing information from animal shelters in Austin, Texas.

Example of importing database in MongoDB:



1. Create User Account – Create a new user for the database and grant them access to the required database.

Testing authorization of a new user account in MongoDB:

A computer screen shot of a program

Description automatically generated

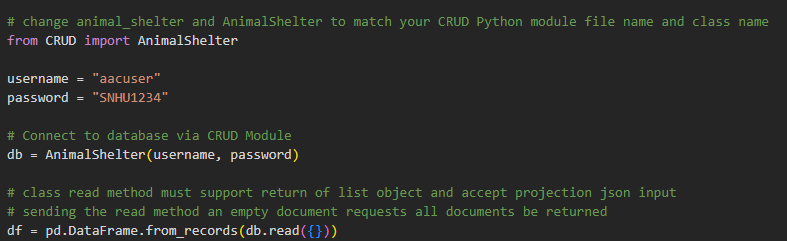
1. Create CRUD module –
   1. Create method – Allow user to add new entries into database using dictionary variables.
   2. Read method – Allow users to find entries using dictionary variables.
   3. Update method – Allow users to find and update a record using dictionary variables.
   4. Delete method – Allow users to find and delete a record using a dictionary variable.
2. Setup a Jupyter Dashboard using Dash –
   1. Import necessary libraries such as *dash*, *dash\_leaflet*, and *plotly*

Example imports:

Example import commands

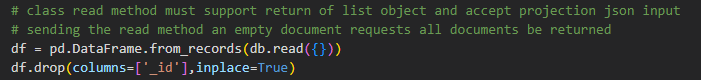

1. Connect to database using CRUD module.

Example of connecting to CRUD module:



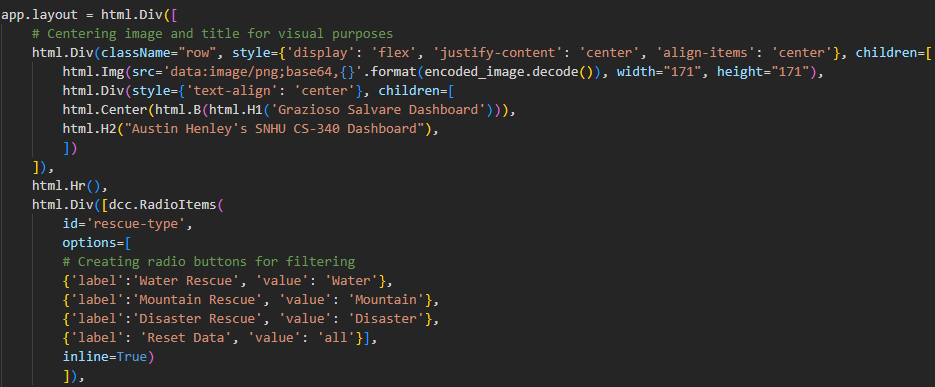
1. Prepare the data and convert it into a Pandas data frame

Example of reading and converting data to data frame:



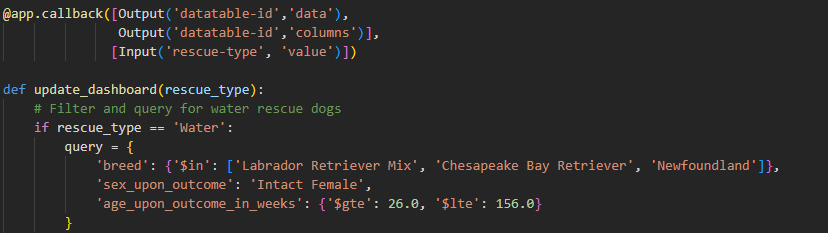
1. Define a layout for the data and create a dashboard

Example of dash layout :

**

1. Use Dash’s callback functionality to make the dashboard interactive for users.

Example of callback functionality using a filter



1. Run dashboard to make it accessible.

Example code to run dashboard:



## Installation

* Python
* PyMongo library
* MongoDB
* Python IDE
* Jupyter Notebook – For testing of Python Module
* Dash
* Pandas
* Plotly
* Dash-leaflet

## Usage

### Code Example

*Create method used in this module:*

*A screen shot of a computer code

Description automatically generated*

Read method used in this module:

A computer screen shot of a code

Description automatically generated

### Tests

*Testing method for Create:*

*A screen shot of a computer code

Description automatically generated*

*Testing method for Read:*

A screenshot of a computer code

Description automatically generated

*Testing method for Update:*

A screenshot of a computer program

Description automatically generated

*Testing method for Delete:*

A screenshot of a computer code

Description automatically generated

*Dashboard functionality:*

**No Filter selected / Reset Filter:**

**A screenshot of a computer screen

Description automatically generated**

**Water Rescue Filter:**

A screenshot of a map

Description automatically generated

**Mountain Rescue Filter:**A screenshot of a map

Description automatically generated

**Disaster Rescue Filter:**

**A screenshot of a map

Description automatically generated**

## Contact

Austin Henley – Austin.Henley@snhu.edu