# CS 340 README

## About the Project/Project Title

*This software application is used to search for dogs from a database that can be used for search-and-rescue training. The user will be able to search for a specific characteristic such as age or breed. This application will work with an existing database with the options to add, delete, and update profiles.*

## Motivation

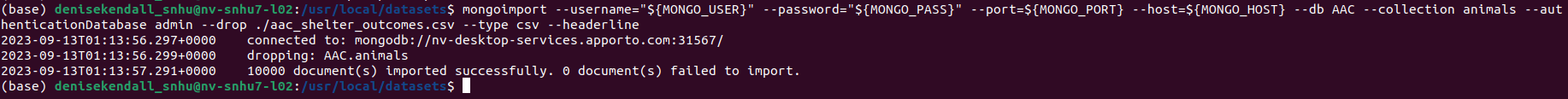
*Grazioso Salvare, a rescue-animal training company, and a nonprofit agency that operates five animal shelters have teamed up to find potential dogs for search-and-rescue training. This will give a dog a new purpose and job that is for the greater good. These dogs will be trained for different types of rescues such as water rescue and locating humans after a disaster.*

## Getting Started

*To set up a project locally, the user will need to import the database by using the Mongo import tool. Then the user will need to log in with a username and password. Depending on the users’ permissions, they will be able to read, create, update, and delete documents. The create method takes the data that was submitted and inserts a document into the specified collection. The method will return true if the document was inserted successfully and false if it was not. The read method searches through the specified collection and will return the results. If the command was unsuccessful, it will return an empty list.*

## Installation

*To import your database, a user will use the mongoimport tool in the terminal. The default datatype is JSON and the user will need to specify that they are importing a CSV file. To create a new user, a user will need to use the admin database. The createUser command can be used to assign a username, password, and role. After importing the database, the user will need to log in using their username and password. Jupyter Notebooks was used to test our CRUD Python module. A IPYNB file was created in Jupyter Notebooks which called and tested the CRUD functionality. For the dashboard layout, we used the Dash framework. Dash is easy to use with pre-build components.*



A computer screen shot of text

Description automatically generated

A screenshot of a computer screen

Description automatically generated

**Steps**

*First, you will need to create a user that can access the database. Then the user can import the database and log into the database. Then you will need to create python module that will hold the functions to interact with the database such as read or create. Then you will need to create a IPYNB file in Jupyter Notebooks to create a dashboard.*

## Usage

### Code Example

*Python has great support for data manipulation and processing tasks which is why it was used for the CRUD module. First, I created the python file. The first function in the module is used to create a document. A user will set key/value pairs and if the creation is successful the function will return True. If there is an error, the function will return an error message and print False. The next function will query for documents from a specific MongoDB database and collection. A user will specify the key/value lookup pair and if the function was successful, it will return a list. If the function was unsuccessful, it will return an empty list. The update function will request a key/value lookup pair that the user query for documents from the specified database. Then the function will request what data the user wants to update. If it was successfully updated the function will return the number of objects that were modified. If not, it will return an error message. The last function is the delete method. The user will input a key/value lookup pair and the function will search and remove the documents from the specified database. The function will return the number of objects removed if it was successfully and it will return an error message if it failed.*

*A computer code with many letters

Description automatically generated with medium confidence*

*A computer code with text

Description automatically generated with medium confidence*

*A computer code with black text

Description automatically generated*

*A computer code with text

Description automatically generated with medium confidence*

### Tests

*To test the methods, I used the aacuser account. Then I created a class named animal\_shelter and input the connection variables. Then I created a document using the create function. I set the type to dog and the name to Bob. The document was successfully created, and the results returned True. I tested the read function by using the type dog as my key/value lookup pair. A list was returned that included the correct key/value pair. After that, I tested the update function by searching for the type dog and updating the type to cat. The test was successful, and the method return 1 as the number of documents updated. The last test was the delete function. I used the key/value look up pair, type cat. Then the function searched the database using that key/value pair and deleted the documents. Then it returned how many objects were deleted.*

**

*A close up of a text

Description automatically generated*

**

*A close up of text

Description automatically generated*

**

*A computer code with red text

Description automatically generated*

**

*A computer screen shot of a cat

Description automatically generated*

**

### Dashboard

*The dashboard has different features such as a list of the profiles of the dogs within the database. When a user logs in they will see an unfiltered list of dogs. At the top of the page, a user can filter the list by selecting a button or reset the list. Towards the bottom of the page, there is a map of the location of the dog. The user can select the button on the left side of the list to find the location of the dog. The dashboard also includes a pie graph that displays the percentage of available dogs based on the list. If a filter is selected, then the pie graph will update based on the filtered list.*

*After the user logs in, the IPYNB file will connect to the database using the python module that was previously created. Then I designed how the dashboard will look. I created the buttons for the filters, the design of the database list, and the placement for the graphs. Then we created a function that will update the list based on the button that the user pushes. The next function will update the pie chart. The function update\_styles will add a background color to a column that is selected. The last function will update the map when a user selects a certain profile.*

*A close-up of a computer code

Description automatically generated*

*A screen shot of a computer code

Description automatically generated*

*A red line drawing of a dog

Description automatically generated*

*A black and white screen with white text

Description automatically generated*

*A computer code with text

Description automatically generated*

*A map with a location on it

Description automatically generated*

*A screenshot of a computer code

Description automatically generatedA screen shot of a computer code

Description automatically generated*

*A screen shot of a computer

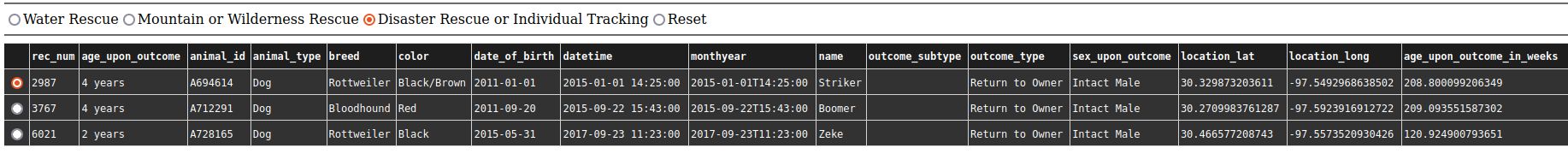
Description automatically generated*

*A screen shot of a computer

Description automatically generated*

*A black screen with many small colored icons

Description automatically generated with medium confidence*

**

*A computer screen shot of text

Description automatically generated*

*A pie chart with numbers and text

Description automatically generated*

*A blue circle with white text

Description automatically generated*

*A colorful pie chart with numbers

Description automatically generated*

*A pie chart with numbers and a red circle

Description automatically generated*

*A screenshot of a computer code

Description automatically generated*

*A black and pink rectangular sign with white text

Description automatically generated*

*A screenshot of a computer program

Description automatically generated*

*A screenshot of a map

Description automatically generated*

## Contact

Denise Kendall