# CS 340 README Template

## About the Project/Project Title

The purpose of the database will allow the customer to work with existing data from animal shelters to identify and categorize available dogs. The Animal Shelter CRUD project will allow the user the ability to create/insert a document into a MongoDB database and collection, read or query documents within the database, update existing documents already inserted within the database, and the ability to remove or delete the documents therein.

## Motivation

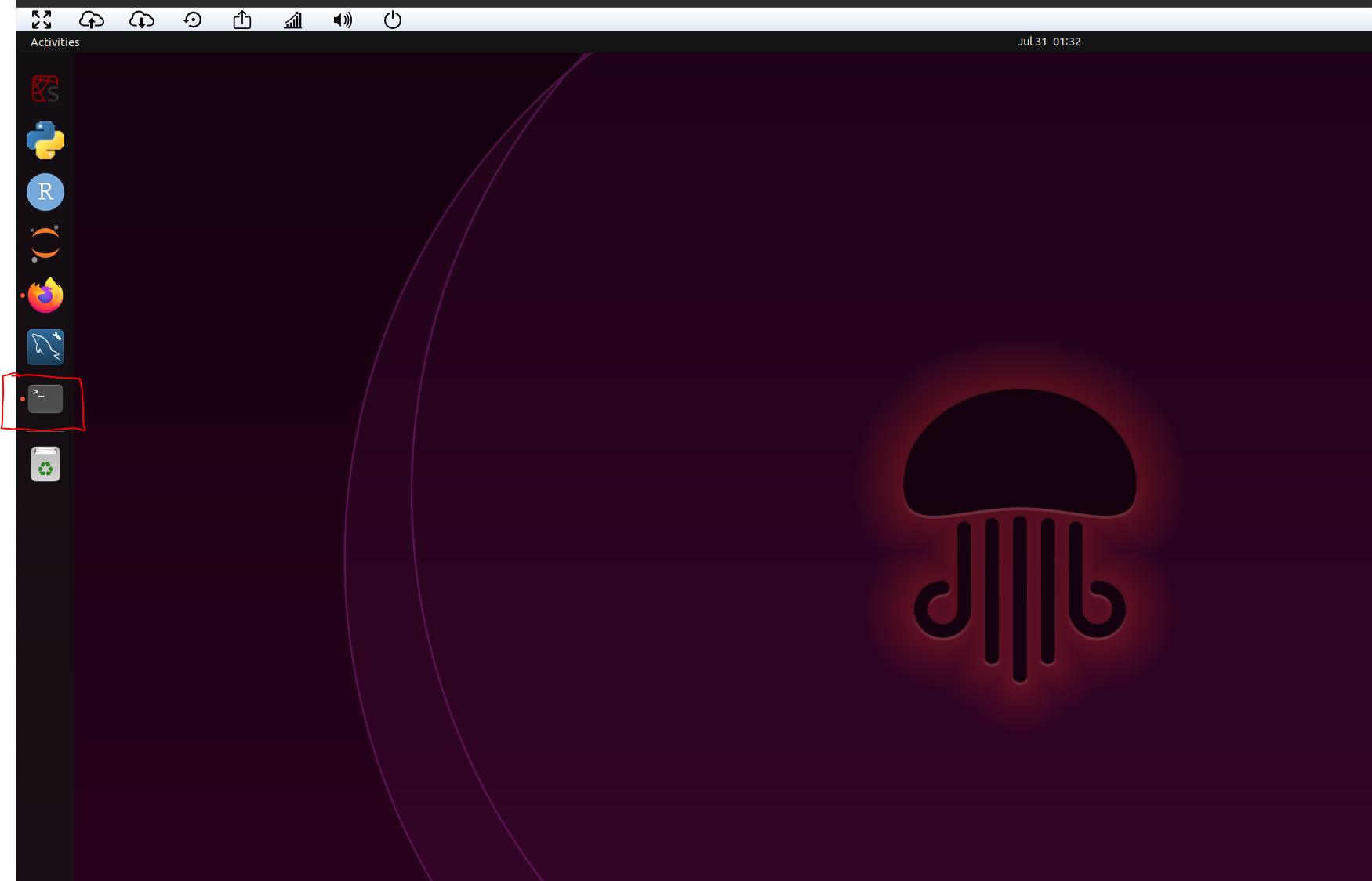
This project exists to give the user quality of life when inserting/creating documents within the database and allowing them to query their findings. These quality-of-life implementations are done so by a python script which runs against the mongo server and will carry out the CRUD operations for the user.

## Getting Started

To begin with you will need to login to your environment by the means provided to you. In this case we will be navigating to the below link to access the VM.

<https://snhu.apporto.com/>

Once you have access to the VM and have desktop access we will navigate to the left side of our screen as seen below and selected the highlighted/circled icon to open the terminal application.

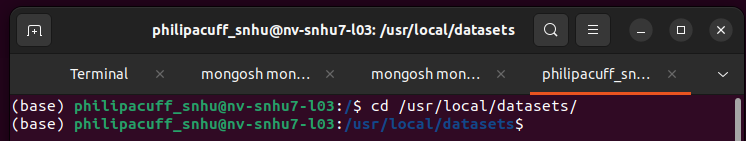


Once we have opened the terminal application we will be provided with the following prompt:

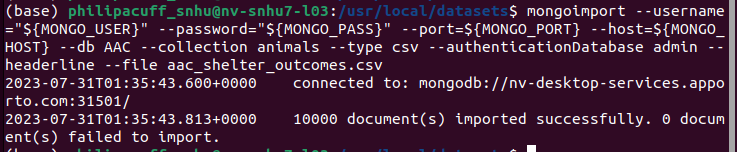


In order to begin we need to navigate to the following directory:

***cd /usr/local/datasets/***

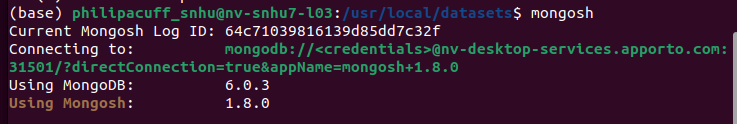


In order to implement this on your local copy you will first need to import the data set using the terminal application. This can be done by using the following command below:***mongoimport --username="${MONGO\_USER}" --password="${MONGO\_PASS}" --port=${MONGO\_PORT} --host=${MONGO\_HOST} --db AAC --collection animals --type csv --authenticationDatabase admin --headerline --file aac\_shelter\_outcomes.csv***



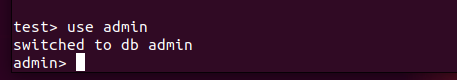
Once you have verified that you imported the data set with no errors move to the next step.

In order to create a user associated with that database you will need to “use admin” after logging in with an admin account into the mongodb (this was already provided in our scenario). You can log in as and admin user by issuing the following command:



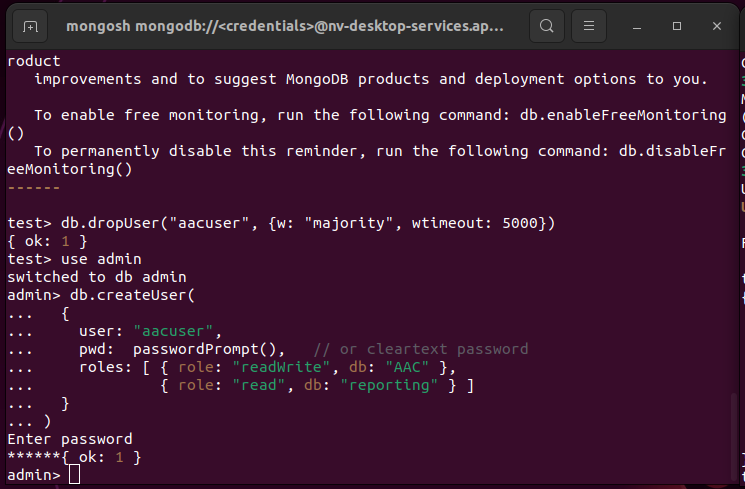
Once you have become admin you can create a new account based on your desired credentials. Our user was created using the following command within mongodb by switching to the admin db.

***use admin***



Insert the following command in the admin database to create an user named aacuser and then insert the password of your choice.

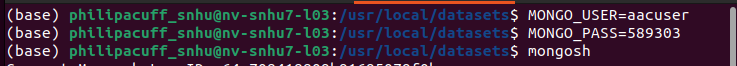
***db.createUser( { user: "aacuser", pwd: passwordPrompt(), /\* or cleartext password\*/ roles: [{ role: "readWrite", db: "AAC" }, { role: "read", db: "reporting" }] })***



With the new user created you should now be able to login as this user to use the create and read functions. You may need to edit your environment in order to login as your created user.

**\*Parsed through terminal\***

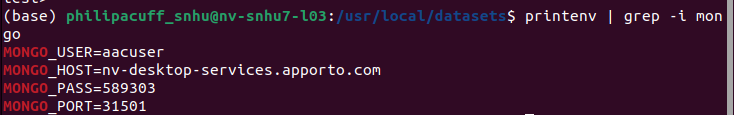
**MONGO\_User=aacuser  
MONGO\_PASS=[password]**



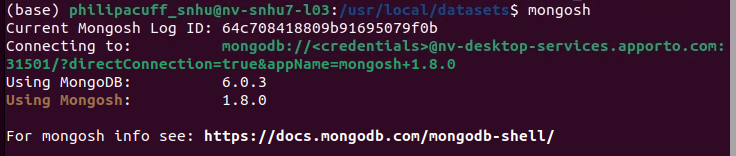
You can test the environment to verify if it has been changed to that user by doing the following:

**\*Parsed through terminal\***

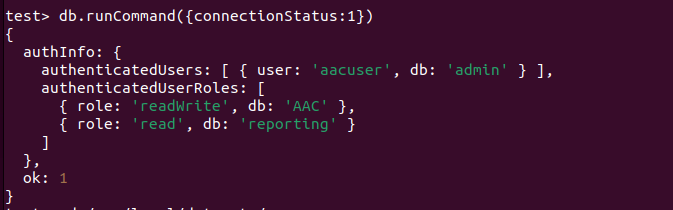
**printenv | grep -i mongo**



If the environment looks correct you may use the “mongosh” command to login to the mongodb.



You can verify the connection status by doing the following command below to verify you are logged in as the correct account:  
  
**db.runCommand({connectionStatus:1})**



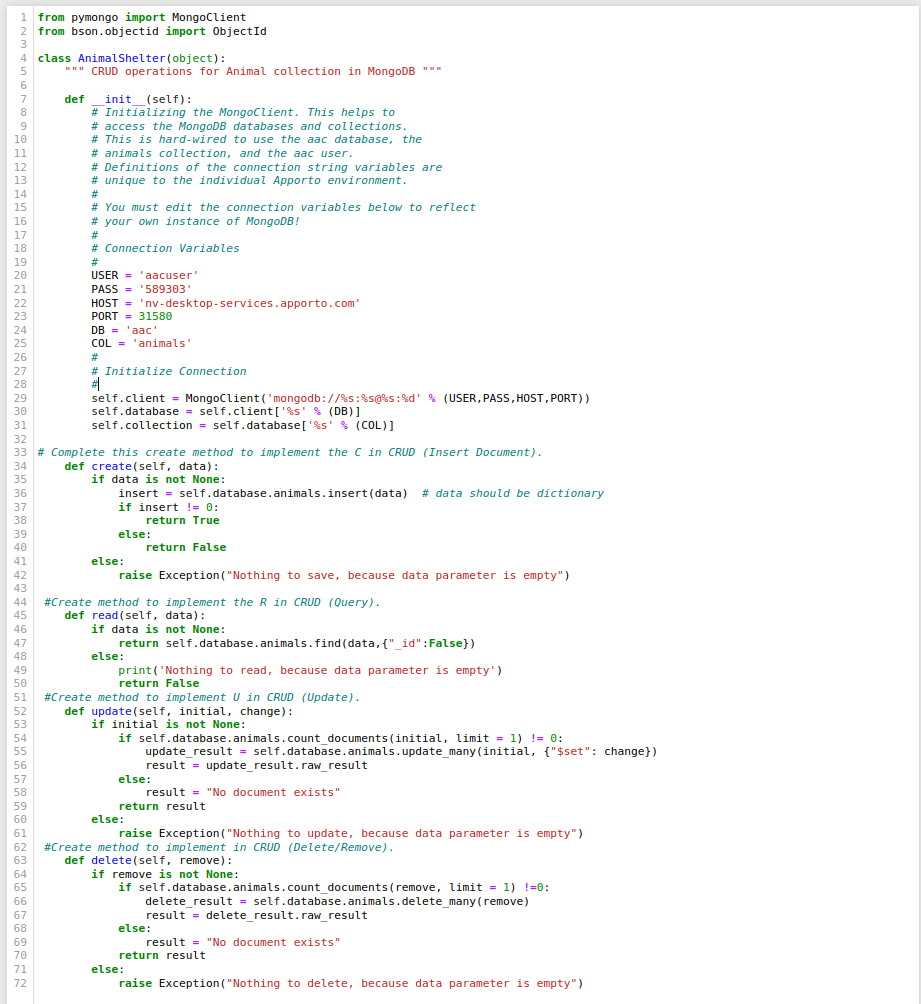
## Installation

* The tools used are listed below and were provided through a VM environment.
  + Terminal Application
  + Jupyter Notebook
  + Mongo

## Usage

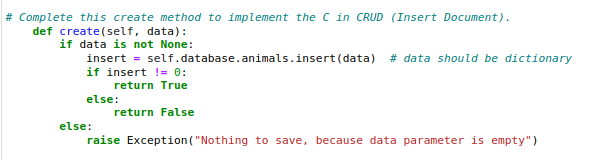
*Use this space to show useful examples of how your project works and how it can be used. Be sure to include examples of your code, tests, and screenshots.*

* CRUD Method This allows the user to insert a document, query/view a document, update existing documents, and remove/delete documents from the database.

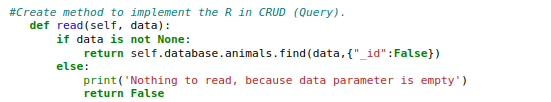


### Code Example

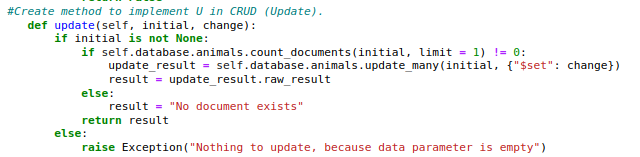
* Create Method
  + The create method allows the user to create/insert a new document with desired parameters into the database



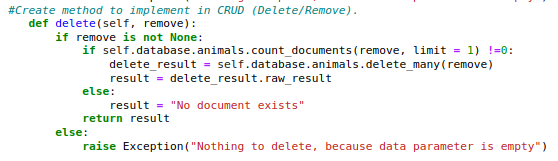
* Read Method
  + The read method allows the user to read/view a document within the database by querying for a specified parameter.



* Update Method
  + The update method allows the user to update an existing document with the new desired parameters for that document



* Delete Method
  + The delete method allows the user to delete a document entirely from the database if desired



**Dashboard Purpose**

The purpose of the dashboard is to allow Grazioso Salvare to identify available dogs they can use in their search and rescue training program. Salvare has contracted for a full stack development of the application, including a database, that will allow its users a dashboard interface that they can interact with directly to the database. This will allow them to located the animals needed for their program using the database information that can provide information about the animals for training as well as their world location. MongoDB was an exceptional choice for development as it allowed us to integrate the database with python. This allowed the use of jupyter notebooks to create test cases and a CRUD method to navigate the database to maintain and manage the database. The Dash framework was a backbone to this idea as it would allow the users a GUI to interact with the database to view the changes and entries in real time. This also allows the user to filter out specific rescue types of animals and their location.

**Dashboard Software & Needed Applications**

* *Jupyter Notebooks*
  + *Notebooks were used to create the code, test files, and implementation of the Dash framework to produce the Dashboard for the users.*
* *Terminal*
  + *Was used to import the aac\_animals csv that the Dashboard will interact with to parse through the data produced by animal shelters, and allow Salvare employees to filter through the data.*
* *Dash Framework*
  + *The backbone of the dashboard as this framework will allow the creation of an interactive GUI for the users to interact with directly to the database.*
* *MongoDB*
  + *The backbone of the project as MongoDB allows us to import the data given by the shelters. With the import of the data we can create, modify, update, and remove the entries, which will be reflected in real time to the dashboard.*
* *Terminal*
  + *The terminal application is the essential gateway on the OS to access the MongoDB.*

**Dashboard Code & Process**

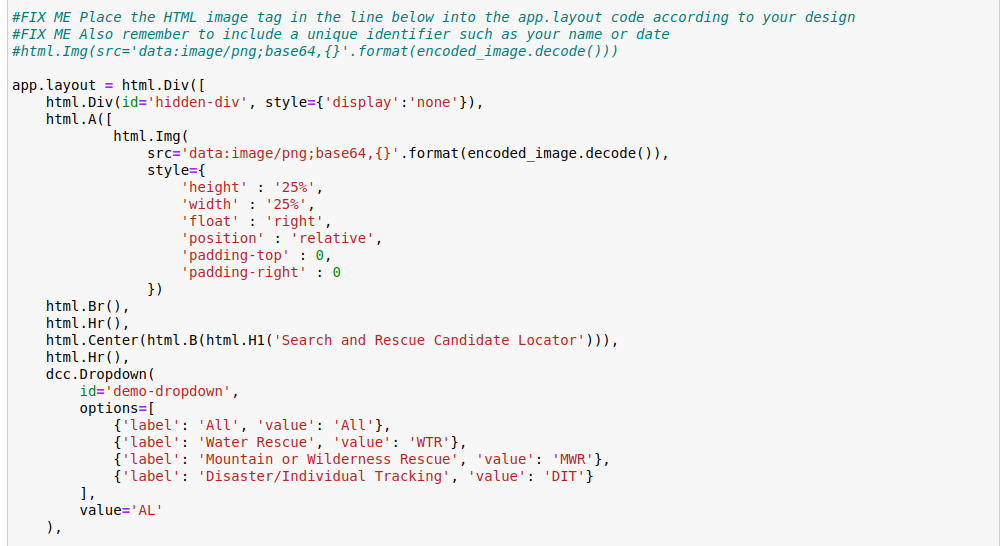
As shown below we have added in functionality that promotes quality of life for the user by giving them more options to interact and filter through the data. The user is able to filter through action parameters and sort items via rows or columns to better understand the data at hand.



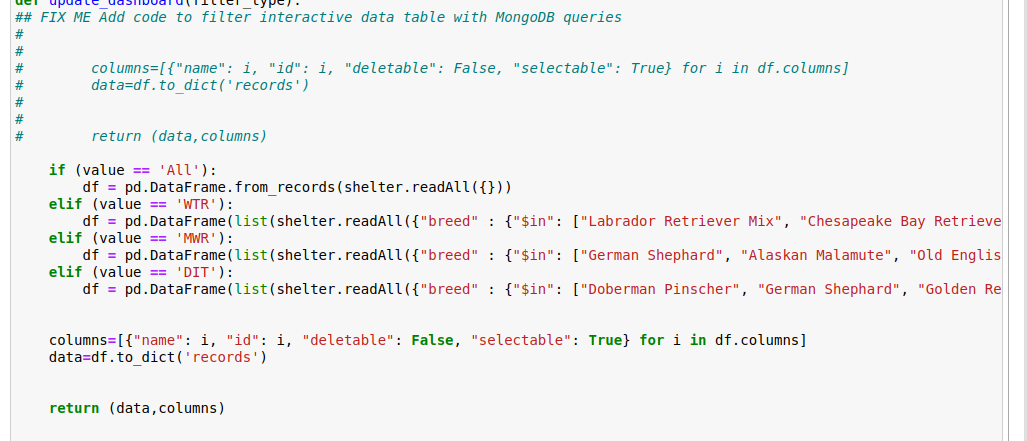
Secondly, we have set up the dashboard so that the data chart and the geolocation chart are side by side which gives the user quality of life as they are able to parse through both data types and cross reference any of the information.



Along with this feature we have also provided a drop-down menu option within the search bar that allows the user to filter between the rescue types provided in Salvar’s specification document. Initially all dogs are displayed on the dashboard and this allows users to specify a specific rescue type so that they can identify the best candidates to recruit for training.



Here the code has been added to filter the interactive data table with the MongoDB Queries. The queries information/data was provided by Salvare’s specifications document as they show through research these types of rescue, breeds, sex, and age, are the ideal rescue animal for each case basis. We have implemented this into the dashboard so that the ideal rescue animal can be filtered for through the rescue data entries from shelters.



Here we have created a pie chart that represents all the top breeds identified by Salvare’s research. This pie graph will display on the dashboard to allow users a visual aid in determining the amount of rescue animals in the shelter data meet their requirements so they can better allocate their resources into training them.



## Roadmap/Features (Optional)

* The backbone of the frontend application is created. The CRUD methods and database end of the project are working as expected. Currently experiencing issues regarding the dashboard testing as errors have not been fleshed out.
* Current errors relate to syntax with regard to the given source code, most notably: html.Br(), and html.Hr(),.
* The design intent of the dashboard has been laid out into the code to meet the requirements for Salvare so that the users may have a GUI dashboard that allows them to interact with the database. Currently troubleshooting the errors to determine why a syntax error is being produced within the interface visual layout.

## Contact

Your name: Tanner Acuff