# Project 2 README

## About the Project/Project Title

*This is an application that creates a dashboard, reading and manipulating data that is read in with python for our client Grazioso Salvare. We are using MongoDB as the database for this application in which we created a user that allows us to manipulate the data with Python code to create, read, update, and delete the data if needed in our application we are mainly using the read function to read the data and filter it to our clients specifications.*

## Motivation

*This is a class project where we are being tested with how we can manipulate data by using a python file and by manipulating that data with the dashboard that we are creating. We are reading our data from the MongoDB file we have by connecting to and then manipulating that with our python code to filter the data we read in.*

## Getting Started

*To get this program started we will need to*

1. *Start up MongoDB*
2. *Start Jupyter*
3. *Open file in Jupyter with our .ipynb file, our python file, and our logo we are using.*
4. *Go into the .ipynb file and hit run to start the dashboard.*

## Installation

*We will need a few tools to run this*

1. *Current version of Python to run .py files*
2. *Jupyter to run the .ipynb file*
3. *MongoDB to have and access the database*

## 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.*

### Code Example:

**Animal Creation**

*This screenshot below is how we will be able to access the database by calling in our user account with the password for them, creating the data, and then pushing the data through with that for loop. We also show how to query information to find certain animals in this case we are looking for an animal with the name of Ahmad (me).*

*To create the data we will import our python data called CRUD which we are using the AnimalShelter module which will give us access to the create and read functions. To create new data we will need to create a data variable in which we will store the information and then push the data with the for loop in which we take the data in and then call animalShelter.create().*

***Animal Query***

*To query data from our database we will create a dataQuery variable and call in our animalShelter.Read() function in which we will input what information we are looking for. In our example below we are looking for an animal name called Ahmad.*

*Animal Update*

*To update data from our database will create a updateQuery variable in which we will use the same example as before where we search for my name and then we create a data variable in which we tell it what we want to update that information to in. We then call our animalShelter function and tell it to use the update function and give it the query information and the data it’s changing.*

*Animal Delete*

*To delete data from our database we will create a deleteQuery in which we specify the data we are looking to delete where it will delete the first instance of it and in this case I am looking for my name and animal type of dog. We then call in our animalShelter function in which we tell it to delete our deleteQuery, if you look in the image you will see that there is 2 missing {‘name’ : ‘ Ahmad’, ‘animal\_type’ : ‘Dog’} that is because on got updated to have a different name and another got deleted.*

*Connection To Dashboard*

*You can see in this screenshot that we are importing our python code from our CRUD file and then importing the AnimalShelter function. We also need to be able to use our username and password that we created earlier for the user in MongoDB which you can see has the username and password and then connects to AnimalShelter as shelter. We then read in our data by connecting to the database and using our python read function to read all the data.*

*Dashboard Logo*

*In the dashboard layout we were asked to include an image with the ability to click on it and it redirects to the website we were given, we first bring in our image file and then encode the image, after that we create a html.A where we call in html.Img and inside of that is where we display the image and after that we create a href to the website. This allows you to click on the image and takes you to the targeted website.*

*Ease Of Use*

*We were asked to create a dashboard that is easy to use which we accomplished by turning on and off options that the users can use to read this data easily*

*Filter Options*

*In the dashboard we created 4 radio buttons to filter our data showing on the dashboard to meet our clients’ specs, these filter options are Water Rescue, Mountain Rescue, Disaster Rescue, and a Reset button to reset the data back to normal with no filter options. In our code we read from our database but make sure to have it filter to the options that we were given where we start by the breeds, then the wanted sex of the animal, and then finally the age range of the animal.*

*Geolocation*

*In the dashboard we also can show the location of the dog on our map by using the geolocation of it. You can also click on the button on the map to show the name of the animal, this was helped to show where the animals are located for our client if they decide to pick up the animal for their operations.*

*Data Filtering*

*In the dashboard we created a pie chart that shows data change as we filter the options this option will help show the types of dogs in each filter option and the age they are since we are looking for certain breeds and a certain age range is why we picked the data to show this information.*

### Tests

*We tested to ensure that only strings can get parsed into the database which you can see with the testing screenshot in which we tried to create a document with data of {0:0} but you get an invalid document type due to the fact of the data is not allowed.*

### Screenshots

*Upload of Database*

*Graphical user interface, text

Description automatically generated*

*User Creation In Mongo*

*Text

Description automatically generated*

*Animal Creation Code Example*

**

*Animal Query*

*Text

Description automatically generated*

*Animal Update*

*Graphical user interface, text, application

Description automatically generated*

*Animal Delete*

*Graphical user interface, text, application

Description automatically generated*

*Testing Code*

*Text

Description automatically generated*

*Connection To Database*

*Graphical user interface, text, application, email

Description automatically generated*

*Dashboard Logo*

*Graphical user interface, text, application, email

Description automatically generated*

*Graphical user interface, text, application, email

Description automatically generated*

*Graphical user interface, application, Teams

Description automatically generated*

*Ease Of Use*

*Graphical user interface, text, application, email

Description automatically generated*

*Table

Description automatically generated*

*Filter Options*

*A picture containing text

Description automatically generated*

*Text

Description automatically generated with medium confidence*

*Table

Description automatically generated*

*Filter Pie Data*

*Graphical user interface, text

Description automatically generated with medium confidence*

*Reset Data Pie Chart*

*Chart, pie chart

Description automatically generated*

*Water Rescue Data*

*Chart, bubble chart

Description automatically generated*

*Mountain Rescue*

*Chart, pie chart

Description automatically generated*

*Disaster Rescue*

*Chart, pie chart

Description automatically generated*

*Geolocation Data*

*Graphical user interface, text, application

Description automatically generated*

*Water Rescue Filter Geolocation*

*Map

Description automatically generated*

## Steps To Complete Project

The steps I took to complete this project was a gradual one in which with each week of work we did was helping me get ready for this project submission and the stuff we learned allowed me to learn how to connect my python code and make it interact with the dataset in MongoDB, and then to use all that I have learned to create the dashboard for the client in our project.

## Difficulties & Challenges

The main thing that was difficult for me was connecting python to MongoDB in which I was running into issues but with the announcement that my professor put out I was able to change part of the code to make that work and the connection issues finally went away.   
  
 Another issue that I ran into was with the development of the filter options in the dashboard code in which I thought I had everything setup correctly with my if-elif statements, but the project would only allow one of them to work and the others would not work. I finally figured out the issue was that I had some indentations in that section of code that was making it hard to read the return data statement but once I got those indentation issues fixed, I was able to finally get everything working.   
  
 Everything else seemed to fall in line after that in which I was able to use the resources that were provided to create the pie-chart and I used the code from module 6 for my geolocation which helped save a bunch of time there.

## Roadmap/Features (Optional)

*We have currently reached our goal of having a fully functional CRUD python code in which we are able to do every instance of creation, reading, updating, and deleting. We are currently assessing into what else we should add to our code base that will make the lives of our users easier, if you have any ideas or requests, please feel free to reach out to our development team so that we can get a better understanding of what our users would like for us to add next.*

## Contact

Ahmad Omar