# CS 340 README Project Two Submission

**About the Project/Project Title**

*The Project’s Title is named, “7-2 Project Two Submission”. This project is done by building a Create(C), Read (R), Update (U), and a Delete (D) functionality in the programming language of choice here is Python to be implemented into the MongoDB Server where its interaction belongs to the CRUD operation from the backend development. Along with this I have created and developed a python app with some aid of libraries like Dash which helps to fetch data from Mongo’s Database that is Rest API where I had completed them within the course.*

## Motivation

*From this project the databases are given where it is very simple for a developer to gain access and query within the MongoDB. The specification to this project is to build a Create, Read, Update, and Delete functionalities in the chosen programming language Python which its execution would translate or fetch to retrieve back from the Mongo’s Database developer companion which the Jupyter Notebook Middleware gets involved. The motivation of keeping myself on track to continue developing this project was to organize it in the right meaningful way, which is user friendly where precisely my application comes in to play for Grazioso Salvare to take over and be useful for business needs.*

## Getting Started

*Well, I first started when used the MongoDB import tool which is called, “Austin Animal Shelter Center Outcomes” Datasets by first creating an Index as shown below from my screenshot. Where I also needed to authenticate the Database with the prompt username “AAC” and password “aacuser” which it creates granted access for a user to login where I had to now log into the Mongo using the new user identification which was created as shown below from the screenshots. Then I got access to the cd /usr/local/datasets folder to use the database mongo import tool as show below in the screenshots, which imports all files from all the animals’ data to be used for the project.*

*Graphical user interface, text

Description automatically generated*

*Graphical user interface, text

Description automatically generated*

*A screenshot of a computer

Description automatically generated*

*Graphical user interface, text, application

Description automatically generated*

## Installation

*I had to use libraries drivers as my tools for Jupyter notebook and the command line to interact to each other. Next, I had to work in Python language from the Jupyter notebook which I first had to create a File.py to create, read, update, and delete functionalities of the CRUD operations. Then, I had to test my File.py which is called, “AnimalShelter.py” in a different file where it’s under IPYNB file. As shown below from the screenshot, I learned that I had to use AnimalShelter.py file that has the CRUD operation to be imported into the IPYNB test file.*

**

## Usage

*The CRUD functionalities Project is self-explanatory which helps the Mongo Database to work with the infrastructure built is to make the Mongo Database know how to create, read, update, and delete any file or data in the datasets given.*

### Code Example

*Below I have inserted some attached screenshots to demonstrate how the Python code is used by the libraries and the drivers like PYMongo for instance. As I had to research to find what extensions, drivers, and libraries are used to be shared in this project. Below are some screenshots demonstrating how I made the Python Module file where the functions of CRUD operation will be integrated.*

* **A Create method that inserts a document into a specified MongoDB database and collection**
  + Input -> argument to function will be a set of key/value pairs in the data type acceptable to the MongoDB driver insert API call.
  + *Graphical user interface, text, application

    Description automatically generated*Return -> “True” if successful insert, else “False”.
* **A Read method that queries for document(s) from a specified MongoDB database and specified collection**
  + Input -> arguments to function should be the key/value lookup pair to use with the MongoDB driver find API call.
  + Return -> result in cursor if successful, else MongoDB returned error message.

*Graphical user interface, text, application, email

Description automatically generated*

* **An Update method that queries for and changes document(s) from a specified MongoDB database and specified collection**
  + Input -> arguments to function should be the key/value lookup pair to use with the MongoDB driver find API call. Last argument to function will be a set of key/value pairs in the data type acceptable to the MongoDB driver insert API call.
  + Return -> result in JSON format if successful, else MongoDB returned error message.

*Graphical user interface, text, application, email

Description automatically generated*

* **A Delete method that queries for and removes document(s) from a specified MongoDB database and specified collection**
  + Input -> arguments to function should be the key/value lookup pair to use with the MongoDB driver find API call.
  + Graphical user interface, text, application, email

    Description automatically generatedReturn -> result in JSON format if successful, else MongoDB returned error message.

### Tests

*For testing, I had used the Jupyter notebook software tool to import the CRUD anticipated operation by providing some sample data that may be used when using MongoDB could be when attempting to create, read, update, or delete any given data as it is shown below from the screenshots. I had to create a python script to test the CRUD operations which it shows that it’s working properly from the original python module of Animal Shetler.py file that was imported for Mongo database integration as followed:*

*from AnimalShelter import AnimalShelter*

*def createAnimals():*

*animals = AnimalShelter()*

*data = {}*

*print("Enter Data, [Key followed by Value]")*

*while (True):*

*key = input("Enter Key or 'exit' to quit ")*

*if key == "exit":*

*break*

*value = input("Enter Value: ")*

*data[key] = value*

*animals.create(data)*

*def searchAnimals():*

*animals = AnimalShelter()*

*data = {}*

*print("Enter Data, [Key followed by Value]")*

*while (True):*

*key = input("Enter Key 'exit' to quit.")*

*if key == "exit":*

*break*

*value = input("Enter Value:\n")*

*data[key] = value*

*animals.read(data)*

*def updateAnimals():*

*animals = AnimalShelter()*

*specifications = {}*

*print("Enter Specifications of animal to update: ")*

*while (True):*

*key = input("Enter Key: ")*

*if key == "exit":*

*break*

*value = input("Enter Value: ")*

*specifications[key] = value*

*print("Enter New Values")*

*new\_data = {} #empty dictionary*

*while(True):*

*key = input("Enter Key: ")*

*if key == "exit":*

*break*

*value = input("Enter Value: ")*

*new\_data[key] = value*

*animals.update(specifications, new\_data)*

*def deleteAnimals():*

*animals = AnimalShelter()*

*todelete = {}*

*print("Enter Specifications for animal to be deleted")*

*while (True):*

*key = input("Enter Key [exit to quit]:")*

*if key == "exit":*

*break*

*value = input("Enter Value:\n")*

*todelete[key] = value*

*animals.delete(todelete)*

*read\_data = animals.read(todelete)*

*#Confirm Deletion of Data*

*if read\_data is not None:*

*print("Animal Deleted")*

*else:*

*print("Animal was not deleted")*

*def displayMenu():*

*print("1. Add New Animal")*

*print("2. Remove Animal(Delete)")*

*print("3. Update Animal (Modify)")*

*print("4. Search For An Animal")*

*print("5. Exit app")*

*def main():*

*displayMenu()*

*user = input("Choice")*

*while user != "5":*

*if user =="1":*

*createAnimals()*

*elif user == "2":*

*deleteAnimals()*

*elif user == "3":*

*updateAnimals()*

*elif user == "4":*

*searchAnimals()*

*else:*

*print("Invalid data")*

*displayMenu()*

*user = input("Choice ")*

*main()*

### Screenshots

***Output:***

*Graphical user interface, text, application, email

Description automatically generated*

*Graphical user interface, text, application, email

Description automatically generated*

## Roadmap/Features (Optional)

*Provide an open issues list of proposed features (and known issues). If you have ideas for releases in the future, it is a good idea to list them in the README. What makes your project stand out?  
  
Note: This section is optional for the purposes of this assignment. If you choose not to fill out this section, remove it from your final README file.*

## Contact

Your name: Joel Meza