# CS 340 README

*Use this template to complete your README file. When completing the template, keep the headings as they are so that your document has a clear organization. Remove the italicized prompt text after you have completed each section for a polished final document.*

## About the Project/Project Title

*Provide a little information about your project or an overview that explains* ***what*** *the project is about.*

*CRUD Operations for Animal Collection in MongoDB.*

## Motivation

*This is a short description of the motivation behind the creation and maintenance of the project. This should explain* ***why*** *the project exists.*

*The reason for this project is to learn how to create and develop a working python script with MongoDB.*

## Getting Started

*This is an example of how you may give instructions on setting up your project locally: “To get a local copy up and running, follow these simple example steps.”*

1. *Upload the Austin Animal Center (AAC) Outcomes data set into MongoDB by****importing a CSV file using the appropriate MongoDB import tool****. This file is in the /usr/local/datasets/ directory in Apporto and the filename is “aac\_shelter\_outcomes.csv”. Use the database name “AAC” and collection name “animals”. Complete the import using the mongoimport tool and****take screenshots****of both the import command and its execution.  
     
   Note: If you completed the Module Three milestone, you have already completed this step.*
2. *Next, you must develop a Python module in a PY file, using object-oriented programming methodology, to enable****create****and****read****functionality for the database. To support code reusability, your Python code needs to be importable as a module by other Python scripts.****Develop****a CRUD class that, when instantiated, provides the following functionality:*
   1. ***A method that inserts a document into a specified MongoDB database and collection***
      1. *Input -> argument to function will be set of key/value pairs in the data type acceptable to the MongoDB driver insert API call*
      2. *Return -> “True” if successful insert, else “False”*
   2. ***A method that queries for documents from a specified MongoDB database and specified collection***
      1. *Input -> arguments to function should be the key/value lookup pair to use with the MongoDB driver find API call*
      2. *Return -> result in cursor if successful, else MongoDB returned error message  
           
         Important: Be sure to use****find()****instead of****find\_one()****when developing your method.*

*As you develop your code, be sure to****use industry standard best practices****such as proper naming conventions, exception handling, and in-line comments. This will ensure that your code is easy to read and reusable for future projects.  
  
TIP: Use the following sample code to get started. Note that the authentication to MongoDB is in the initialization method for the CRUD class.*

***Example Python Code to Insert a Document***

*from pymongo import MongoClient*

*from bson.objectid import ObjectId*

*class AnimalShelter(object):*

*""" CRUD operations for Animal collection in MongoDB """*

*def \_\_init\_\_(self):*

*# Initializing the MongoClient. This helps to*

*# access the MongoDB databases and collections.*

*self.client = MongoClient('mongodb://%s:%s@localhost:YOUR\_PORT\_NUMBER' % (username, password))*

*self.database = self.client['project']*

*# Complete this create method to implement the C in CRUD.*

*def create(self, data):*

*if data is not None:*

*self.database.animals.insert(data) # data should be dictionary*

*else:*

*raise Exception("Nothing to save, because data parameter is empty")*

*# Create method to implement the R in CRUD.*

*def read(self, data):*

*if data is not None:*

*self.database.animals.insert(data) # data should be dictionary*

*else:*

*raise Exception("Nothing to save, because data parameter is empty")*

*# Create method to implement the U in CRUD.*

*def update(self, data):*

*if data is not None:*

*self.database.animals.insert(data) # data should be dictionary*

*else:*

*raise Exception("Nothing to save, because data parameter is empty")*

*# Create method to implement the D in CRUD.*

*def delete(self, data):*

*if data is not None:*

*self.database.animals.insert(data) # data should be dictionary*

*else:*

*raise Exception("Nothing to save, because data parameter is empty")*

1. *Finally,****create a Python testing script that imports your CRUD Python module to call and test the create and read instances of CRUD functionality****. Be sure to use the username and password for the “aacuser” account for authentication when instantiating the class. This script should be created in a separate Jupyter Notebook IPYNB file, and should import and instantiate an object from your CRUD library to effect changes in MongoDB. After creating your script, execute it in Jupyter Notebook and take screenshots of the commands and their execution.*

## Installation

*List the tools you need to use the software and how to install them.*

*In order to create and use the program. You need to install pymongo.*

*PyMongo can be installed with*[*pip*](http://pypi.python.org/pypi/pip)*:*

*$ python -m pip install pymongo*

*Or easy\_install from [setuptools](http://pypi.python.org/pypi/setuptools):*

*$ python -m easy\_install pymongo*

*You can also download the project source and do:*

*$ python setup.py install*

## 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

*Show what the library does as concisely as possible. Developers should be able to figure out how your project solves their problem by looking at the code example. Make sure that your code is short and concise.*

import MongoClient

import ObjectId

class AnimalShelter(object):

""" CRUD operations for Animal collection in MongoDB """

def \_\_init\_\_(self):

# Initializing the MongoClient. This helps to

# access the MongoDB databases and collections.

self.client = MongoClient('mongodb://%s:%s@localhost:YOUR\_PORT\_NUMBER' % (username, password))

self.database = self.client['project']

# Complete this create method to implement the C in CRUD.

def create(self, data):

if data is not None:

self.database.animals.insert(data) # data should be dictionary

else:

raise Exception("Nothing to save, because data parameter is empty")

# Create method to implement the R in CRUD.

def read(self, data):

if data is not None:

self.database.animals.insert(data) # data should be dictionary

else:

raise Exception("Nothing to save, because data parameter is empty")

### Tests

*Describe and show how to run the tests with code examples.*

*Testing is done with Jupyter Notebook. You can import your code and run the code to see its execution.*

### Screenshots

*Provide screenshots that demonstrate your work.*

*Graphical user interface, text, application, email

Description automatically generated*

* *An****explanation of the purpose of the CRUD Python module***

*The purpose of CRUD is to create, read, update and delete files within the animals collection.*

* *An****explanation of how the module should be used****, including:*
  + *A description of the Python driver for Mongo that was used and why it was chosen*

*In order to create and execute our python code, I chose to use pymongo. That way I would be able to import the code into Jupyter Notebook.*

* + *An explanation of the attributes and working functionality of the CRUD operations*
* *A****demonstration of the module’s functional operations****, including:*
  + *Screenshots of the MongoDB import execution. You took these screenshots in Step 1.*

*Graphical user interface, text

Description automatically generated*

* + *Screenshots of the user authentication execution. You took these screenshots in Step 2.*

*Text

Description automatically generated*

* + *Screenshots of the CRUD functionality test execution. You took these screenshots in Step 4.*

*Graphical user interface, text, application, email

Description automatically generated*

*Rescue Type and Preferred Dog Breeds Table*

*Grazioso Salvare has provided the following table based on their research on and experience with training rescue dogs. Reference the table provided below to guide you in writing the queries for the interactive option functionality.*

*Rescue Type Preferred Breeds Preferred Sex Training Age\* Water Labrador Retriever Mix, Chesapeake Bay Retriever, Newfoundland Intact Female 26 weeks to 156 weeks Mountain or Wilderness German Shepherd, Alaskan Malamute, Old English Sheepdog, Siberian Husky, Rottweiler Intact Male 26 weeks to 156 weeks Disaster or Individual Tracking Doberman Pinscher, German Shepherd, Golden Retriever, Bloodhound, Rottweiler Intact Male 20 weeks to 300 weeks*

*\*Note: In the data set, there are two variables for age. It will be easier to use the variable age\_upon\_outcome\_in\_weeks to construct your queries. Age calculations are approximations.*

*A picture containing icon

Description automatically generated*

*Graphical user interface, text, application, email

Description automatically generated*

*Graphical user interface, text, application, email

Description automatically generated*

*Graphical user interface, text, email

Description automatically generated*

## Contact

Your name: Alfredo A. Gomez