# PROJECT ONE README

## About the Project/Project Title

This project is the first phase of a bigger project that includes creating a full stack development application that will include a database and a client-facing web application dashboard. For this phase of the project, the focus is on developing a Python module to interact with a MongoDB database that enables the user to insert new data, read and update existing data, and delete data in a specific database with provided authentication.

## Motivation

The project is for an international rescue-animal training company, Grazioso Salvare, that needs a software application that can work with existing data from the animal shelters to identify and categorize available dogs. The first phase will focus on creating a database in MongoDB that can interact with client-side code in Python language.

## Getting Started

In the terminal use the mongoimport command to load the CSV file into the MongoDB database.

*Text

Description automatically generated*

Once the file has loaded, select the directory where the file is located and start the mongo shell with the following command: /mongod\_ctl start

Graphical user interface, text, application

Description automatically generated

This will start the mongo shell requiring authorization. Once mongo is running on the server, start the mongo shell using the proper username and password by entering the following command: mongo –authenticationDatabase “NAME\_OF\_DATABASE” -u “USERNAME” -p

Log-in example with the administrator account “andreasAdmin”:

Text

Description automatically generated

Log-in example with the user account “aacUser”:

Text

Description automatically generated

You can now use a python script to access mongo by importing the CRUD module and typing the following command: from crud\_animal import AnimalShelter



To authenticate the use from Python and access the secured database, create authentication variables



You can then instantiate an assign variable to begin using the CRUD functions: .create(data) and .read(data)



## Installation

The tools needed to run this software include MongoDB and Python. From the python you will need to import the CRUD module that was created, which contains the MongoClient class in the pymongo library.

## Usage

There are four functions written in the CRUD module, to create, read, update, and delete data. The create function is used by using the instantiated variable, in this case “assign” followed by “.create” for the create function, “.read” for the read function, “.update” for the update function, and “.delete” for the delete function. Below are code examples on how to create the CRUD module and perform each of the respective functions.

### Code Example

To create the CRUD module, you must import mongoClient from pymongo, a connection class that acknowledges all writes in MongoDB. Also, import ObjectId from bson.objectid, and dumps from bson.json\_util to create object Id’s and output in JSON format.

Text

Description automatically generated

The next step initializes MongoClient by connecting to the host server and taking username and password parameters

Text

Description automatically generated

You can then write create, read, update, and delete methods to complete the CRUD module.

Create function: The input argument to function will be a set of key/value pairs in the data type acceptable to the MongoDB driver insert API call. The return will be “True” if successful insert, else “False.”

Graphical user interface, text, application, email

Description automatically generated

Read function: The input arguments to function should be the key/value lookup pair to use with the MongoDB driver find API call. The return should result in cursor if successful, else MongoDB returned error message.

*Text

Description automatically generated with medium confidence*Update function: The 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. The return should result in JSON format if successful, else MongoDB returned error message.

Graphical user interface, text

Description automatically generated

Delete function: The input arguments to function should be the key/value lookup pair to use with the MongoDB driver find API call. The return should result in JSON format if successful, else MongoDB returned error message.

Text

Description automatically generated

### Tests

To test if the module is working write some print statements to accompany each command and confirm the appropriate outcome. Below are some test examples from a python script to test each step of the CRUD module.

To test that an object was created after authentication and class instantiation, simply print the instance variable

Graphical user interface, text, application, email

Description automatically generated

To test the insert function, call the create method and print the result, which should return true:

Graphical user interface, text

Description automatically generated

To test the read method, call the read method and print, which should result in a cursor object. When printing the data, the object should also display in JSON format:

Text

Description automatically generated with medium confidence

To test the update function, calling the update method should automatically result in a print in JSON format of the update:

Graphical user interface, text

Description automatically generated with medium confidence

To test the delete function, calling the delete method should automatically result in a print in JSON format of the delete:

Graphical user interface, text, application

Description automatically generated

## Contact

Andreas Galatis