# CS 340 Project 2 README

*Lukas Mueller*

## About the Project/Project Title

*The goal s of this project included the creation of a python module that can connect to a MongoDB server and access a particular database using access credentials, and display database information from the database in a data visualization dashboard interface, while providing interactivity options that permit a user to perform dynamic sorting, filtering, selection, and search functions on the data. Interacting with the data table produced in the dashboard interface produces dynamic changes in both a data graph (pie chart) and a geolocation chart (map) that are displayed with the table. The software design pattern used for this multi-tier application is the Model View Controller (MVC) pattern. The model is contained and accessed in MongoDB, the views are dashboard widgets, and the controller uses a custom-built CRUD Python module for queries as part of the interaction between components. Additionally, the RESTful protocol is use for extending the HTTP protocol to provide an application programming interface (API). Interactive options to filter the Austin Animal Center Outcomes data set*

## Motivation

*The reason it is worthwhile to create such a module is that it allows programmatic interaction with a MongoDB database server through python programs or scripts, while reducing the effort of such interaction to the degree that it can be used more ubiquitously. The python CRUD module can also be reused in the deployment of other custom database interface stacks. This can have useful applications in the development of software that requires user friendly database functionality; CRUD operations can be automated and performed in response to actions performed in a dashboard or another graphical user interface. Providing the functionality to sort, filter, select, and search data through an interactive and intuitive user interface permits the utility of the underlying mechanisms to be accessed by a greater number of people. This reduces the barrier of use for a powerful technology and is expected to bring value to both individuals and organizations.*

## Getting Started

*Running a local instance of this program requires a user to first satisfy a few prerequisites. A user must have a local version of the MongoDB installed as a service on their local computer, and this service must be started and running. To start the mongod service from the command line, type:*

*mongod start-noauth*

*A database must be created that is named AAC, which can be optionally populated with documents through an import operation. A MongoDB user “aacuser” must be created, along with a password (in this default case, “password”) and be given access privileges to the AAC database. The user must also ensure that Python 3 has been installed on their local computer, and that a suitable IDE is properly configured and available to run the Python code or that the folder that contains the module has been added to the PATH environment variable, for execution via the command line terminal.*

## Installation

***Python module only*:**

*Download and install an on-premises version of MongoDB from the parent website at www.mongodb.com (the community version is free of charge). Download and install Python 3 at* [*www.python.org*](http://www.python.org/) *(Python comes with its own Python compatible lightweight IDE, “IDLE”). Download the module file and modify the PATH environment variable to include the directory that holds the module; alternatively, you can skip setting the PATH variable and execute the module from withing the parent directory (or specify the entire pathname). Access the python terminal and use the pip command to install pymongo, a native Python driver for MongoDB. After ensuring that the MongoDB service is running, use the terminal to run the module by typing:*

*python crud.py*

*Downloading and installing Jupyter Notebook could also be instrumental to testing, as the test file is in .ipynb format, but other options are also available such as ipython (further detail provided in the testing section, below).*

***(Package Software):***

*Ensure that all files in the downloaded package remain in the same directory, as the removal of some files will disrupt software functionality or prevent it altogether. Run the mongod service, as previously described, and open Jupyter notebook. Access the package directory in the notebook; open and run P2.ipynb. This will initiate the dashboard interface within Jupyter notebook and utilize the crud.py python module as needed to query the MongoDB backend database.*

## Usage

*First, the python module imports the necessary driver and ObjectId class. The AnimalShelter class initializes an object instance that represents a user session; part of the object initialization involves establishing a client/server connection to a MongoDB database, authenticated with a predefined username and password. The class methods use input validation for their parameter, checking that it is both not empty and is of the dictionary data type. The methods that are include in the python module embody the functionality of database create, read, update, and delete functions, while the current application only makes use of the detection of entries matching a filtered query or the complete display of the first document that matches a query. In each case this is accomplished through the manipulation of a cursor that points to a list of matching documents.*

### MongoDB Commands:

***Importing a csv data file into the AAC database:***

Text

Description automatically generated

***Login to MongoDB using a standard user account:***

Text

Description automatically generated

**(The python module performs the equivalent of these login commands when the controller file invokes the AnimalShelter class.)**

### Source Code

### *Module(crud.py):*

Text

Description automatically generated

Text

Description automatically generated

***Controller (P2.ipynb):***

:Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generatedText

Description automatically generated

Text

Description automatically generatedText

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

### Tests (crud\_test.ipynb)

*The test file should be kept in the same directory as the module. The purpose of the test file is only to verify the functionality of the python module component of the software. The default test document was created in Jupyter notebook, so has an. ipynb extension. A user can either open this filetype with Jupyter Notebook, and run the file, or simply convert it to a .py format by navigating to the parent directory and typing the following into the command line (with substitutions):*

*ipython nbconvert --to python [name\_of\_your\_notebook].ipynb*

*After importing the AnimalShelter class from the CRUD module, the test file initializes an AnimalShelter object, specifying the username and password that will be used to login to the AAC database. It then sequentially tests each of the module functions, printing confirmation of each successful function execution.*

### Test Code(crud\_test.ipynb)

Graphical user interface, text, application

Description automatically generatedGraphical user interface, text, application

Description automatically generated with medium confidence

**Test Output**

Text, letter

Description automatically generated

HTTP API:

Graphical user interface, text, email

Description automatically generated

***Jupyter ScreenCast of Running Software***(Double-click on image below to play):



## Roadmap/Features (Optional)

*Future modifications to this project should allow a more diverse backend range of database querying and functionalities such as aggregate, and bulk function. The dashboard interface should also be enriched to include functioning widgets that give easy access to database manipulations such as create, update, and delete functions. Additional dynamic charts can also be added to leverage useful data visualization capabilities. If possible, a custom python distribution package should be created to enable the module to be imported into other code without needing to add it to PATH or traverse to a particular current working directory. The test file should be made more sophisticated and be reformatted into a .py file to reduce the requirements needed to use it across multiple platforms.*

## Contact

Your name: Lukas Mueller

i6r89582@duck.com

SNHU, 2022