# CS 340 README Template

## About the Project/Project Title

This project is a Python module that establishes an interface to a MongoDB server with authentication. The module implements CRUD operations using the pymongo library. It additionally performs type-checking against inputs to ensure that as CRUD operations are used, the input is first being validated to be of the type that the MongoDB interface expects. This enables better usability as the exceptions provide explicit details about what the operation expects.

## Motivation

This project exists in order to leverage this module as an interface to other Python projects that need to be able to interface with a MongoDB to persist data that is being collected or created and read the data back in an object-oriented approach in a basic interface.

## Getting Started

To get started using this module you may import animal\_shelter.py directly as a local import.

cd mongo

python

>>> from animal\_shelter import AnimalShelter

## Installation

The tools needed to support this module should be installed using the pip python package manager. The usage is of the officially supported MongoDB Python package which provides first-class support for MongoDB and remains a stable offering for a client of MongoDB.

pip install pymongo

## Usage

This package may be used within a Jupyter Notebook as seen in the screenshot below.

Graphical user interface, text, application, email

Description automatically generated

### Code Example

Initialize client:

>>> test\_client = AnimalShelter(username="aacuser", password="superSecret")

Create document:

>>> test\_client.create({"name": "Bob"})

Read documents:

>>> results = test\_client.read()

>>> print(test\_documents.next())

Update documents:

>>> test\_client.update({"name": "Sue"}, {"age\_upon\_outcome": "4.5 years"})

Delete documents:

>>> test\_client.delete({"name": "Sue"})

### Tests

Tests may be run using the referenced examples in the Jupyter notebook seen in this screenshot.

Graphical user interface, text, application, email

Description automatically generated

### Screenshots

This command executes an import of a CSV file to be added to database as documents.Text

Description automatically generated

This screenshot shows the usage of multiple user accounts for security. **Text

Description automatically generated**

This is the class definition with the associated init method noting that when initialized the class expects valid username and password to be provided.*Text

Description automatically generated*

This is the create method used for creating new documents.*A screenshot of a computer

Description automatically generated*

This is the read method used for reading one or many documents.Graphical user interface, text

Description automatically generated

This is the update method used for updating one or many documents.Text

Description automatically generated

This is the delete method used for deleting one or many documents.Text

Description automatically generated

## Demo

## 

## Tools Used

The framework used for developing this application is called Dash and some corresponding modules such as Dash Leaflet and Plotly Express for the frontend and controllers. This allowed for rapid development of dashboards, as these tools are open-source web development frameworks purpose-built for data visualization. As for the middleware used to interact with the persistent storage, pymongo was used to implement CRUD operations for MongoDB. MongoDB is a lightweight NoSQL database that persists the animal shelter data that was required to be filtered and analyzed for this project. pymongo was chosen as the module for interaction with the database due to being supported first-class by MongoDB maintainers, and thus provides the functionality supported by MongoDB.

**Steps to Complete this Project**

Execution of this project was initiated by first analyzing the requirements of the client. After analyzing the needs and selecting pymongo as an interface with MongoDB as a persistent storage, a portable module was developed to provide authentication and CRUD operations on the database collections. After this the Dash app layout was implemented to display the unfiltered data from the collection that was made when importing the animal shelter data. Once the data was able to be displayed in an unfiltered data table, filtering options were added, and then dashboard visualizations that displayed meaningful representations of the data once filtered.

## Challenges

The primary challenge faced was that of originally attempting to run the application server from a Jupyter Notebook. While Jupyter Notebooks are excellent for prototyping and performing data analysis, running an application server from it proved unreliable. To overcome this, the project was developed to run using the Flask server included within the Dash module. This allows for running an HTTP web server from a workstation, and doing so in development-mode to allow for reloading during development and iterating on visualizations.

## References

https://dash.plotly.com

https://pymongo.readthedocs.io/en/stable/

## Contact

Your name: Michael Kennedy - michael.kennedy4@snhu.edu