# MongoCRUD

Table of Contents

* [About the Project](#_About_the_Project)
* [Motivation](#_Motivation)
* [Getting Started](#_Getting_Started)
* [Installation](#_Installation)
* [Usage](#_Usage)
* [Roadmap/Features](#_Roadmap/Features)
* [Contact](#_Contact)

## About the Project

*The mongo\_crud.py module consists of the MongoCrud class and allows basic CREATE, READ, UPDATE, and DELETE operations to be performed from a web application or API, by connecting to a* [*MongoDB*](https://www.mongodb.com/) *database. MongoDB is a* [*NoSQL*](https://en.wikipedia.org/wiki/NoSQL) *(Not only SQL) document-oriented database system that specializes in horizontal scaling and fast transactions. This module allows web apps and other software to access MongoDB databases, using the four* [*CRUD*](https://en.wikipedia.org/wiki/Create,_read,_update_and_delete#:~:text=In%20computer%20programming%2C%20create%2C%20read,basic%20operations%20of%20persistent%20storage.) *operations, which are as follows:*

* *CREATE*
* *READ*
* *UPDATE*
* *DELETE*

*The user can create new documents to be inserted into the ‘animals’ collection of the’ AAC’ database, view (read) documents in the database by using custom query filters, update documents, and delete documents based on matching queries.*

*The ‘AAC’ database has already been verified and created:*

A screen shot of a computer

Description automatically generated





A computer screen with white text

Description automatically generated

## Motivation

*Databases are used in almost every type of software application, and MongoDB is among the most popular. Working with MongoDB can prove to be quite tedious, and it is common for users to make mistakes when working with a command-line interface (CLI). The mongo\_crud.py module abstracts away the tedious nature of working with databases in a command-line environment and provides a safer, more secure user experience (UX). Future updates will introduce more features and more robust security!*

## Getting Started

*The MongoCrud module can be thought of as the middle “glue” layer between the MongoDB database (base level) and web app client, such as a “DASH” dashboard or an API (application programming interface). The mongo\_crud.py module should be moved to the site-packages location that contains other 3rd party modules:*

* ***Windows****:* 
* ***macOS****/****Linux****:*

* OR *

* ***virtual environment*** *(like* [*Anaconda*](https://www.anaconda.com/)*):*

**

*Be sure to replace <username> with your username and ‘XX’ (or X.X) with your Python version.*

*PyMongo is a library with tools for working with MongoDB and is the official MongoDB driver for Python. PyMongo allows for server administration and even indexing to speed up database operations.*

*The mongo\_crud module’s class (MongoCrud), as well as MongoClient from the PyMongo library, needs to be imported into the API being developed, as seen here:*

**

**

*You will need to install* [*PyMongo*](https://pymongo.readthedocs.io/en/stable/) *on your system, using* [*pip*](https://pypi.org/project/pip/) *or another installation method:*

**

*Now you are ready to programmatically call the class methods from the module, based on the specific triggering mechanism(s) used in your given API or web application. Just create a MongoCrud instance:*

**

*To create a new document and insert the document into the database collection:*

**

*To access a document or group of documents by key-value pair(s):*

**

*To access the database, the user must be authenticated using a user account found in the ‘admin’ database. The ‘admin’ database uses* [*SCRAM*](https://en.wikipedia.org/wiki/Salted_Challenge_Response_Authentication_Mechanism)*-*[*SHA-256*](https://www.simplilearn.com/tutorials/cyber-security-tutorial/sha-256-algorithm) *encryption for password authentication. Currently, the user credentials are hard-coded into the module; however, these strings containing credentials will be removed before deployment of the module.*

*The create and read functions start by ensuring the input value is not empty and is of the dict data type; otherwise, an exception is thrown:*

**

**

*In the create function, if the data is indeed of type* [*dict*](https://www.w3schools.com/python/python_dictionaries.asp) *(dictionary), then the new document is inserted into the collection by using the MongoDB ‘insert\_one’ function.*

*The read function uses a dictionary as a query filter by using the MongoDB ‘find’ function to retrieve the document as a Cursor object. I spent a bit of time struggling to find an efficient way to iterate through the Cursor object so that the queried results could be returned. To solve this problem, I ended up using a list comprehension that only took a single line of code:*



## Installation

*To implement the MongoCrud class methods, the* [*Python*](https://www.python.org/) *programming language and an appropriate IDE (integrated development environment) are needed to develop an API or web application to interact with the mongo\_crud.py module.*

*To install Python, you must first download the installer from* [*https://www.python.org/downloads/*](https://www.python.org/downloads/)*, and install it using the settings appropriate for your development environment:*

A screenshot of a computer

Description automatically generated

To install [PyCharm](https://www.jetbrains.com/pycharm/), just download it from <https://www.jetbrains.com/pycharm/>, and install it:



## Usage

*The mongo\_crud module abstracts away the tedious nature of working with a MongoDB database. Instead of memorizing a bunch of commands, the user may focus on providing the key-value pairs needed for proper creation and querying of the documents found in the specified collection.*

*Typically, a MongoDB user would have to manually start the shell using the ‘mongosh’ command in a Linux (or other) shell environment.*

A computer screen with text and numbers

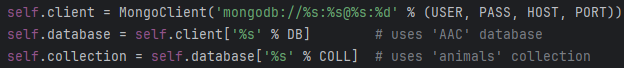
Description automatically generated

*Once in production, the user will be able to log in to the appropriate user account without resorting to using the MongoDB syntax used here:*

A black background with green numbers

Description automatically generated

*These steps are taken care of by the module itself upon creating a MongoCrud class instance:*

**

*It is recommended to develop your API in a way that further abstracts away the dictionary format that is needed to call these functions, so that the user may just focus on providing the key and value strings, omitting the parentheses (), curly braces {}, and commas.*

*Using the module’s ‘create’ function performs a MongoDB ‘insert\_one’ method, using the user-provided key-value pair(s). This abstracts away the following function (example shows recommended ‘insertOne’ function), which is normally needed for this task:*

*A computer screen with numbers and letters

Description automatically generated*

*This is done by passing the key-value pairs (‘doc’) using the following class function:*

A screen shot of a computer code

Description automatically generated

*The API may also be used to query the database to find any documents matching the user-provided filter, by using the module’s ‘read’ function. This will call the MongoDB ‘find’ function, as seen in the following:*

A screenshot of a computer program

Description automatically generated

A computer screen with numbers and letters

Description automatically generated

*The following method provides this functionality upon calling and passing a key-value pair ‘query\_filter’:*

A screen shot of a computer code

Description automatically generated

*The ‘update’ function can also be called to update any of the fields (except ‘\_id’) by executing the ‘update\_many’ function with a ‘query\_filter’ and ‘update\_dict’ being passed, as seen below:*

*A computer screen with numbers and symbols

Description automatically generated*

*This is the Python code that performs this action using parameters ‘query\_filter’ and ‘update\_dict’:*

*A screen shot of a computer program

Description automatically generated*

*The ‘delete’ function calls the MongoDB ‘delete\_many’ that is used here:*

**

*Here is the code, which utilized a key-value pair ‘query\_filter’:*

*A screen shot of a computer code

Description automatically generated*

*In addition to checking for empty inputs and implementing standard error-catching strategies, thorough testing was done on each method in the module to ensure a robust and secure product.*

*If you plan on running these tests yourself (in the test\_mongo\_crud.py file), just make sure the mongo\_crud module is installed, as well as the Python* [*unittest*](https://docs.python.org/3/library/unittest.html) *library:*

**

*To run the included tests (assuming test\_mongo\_crud.py is located in site-packages), open your OS’s CLI, and follow the steps listed below:*

1. *Navigate to the site-packages folder where your third-party libraries should be installed.*
2. *Enter the following command in your CLI:*

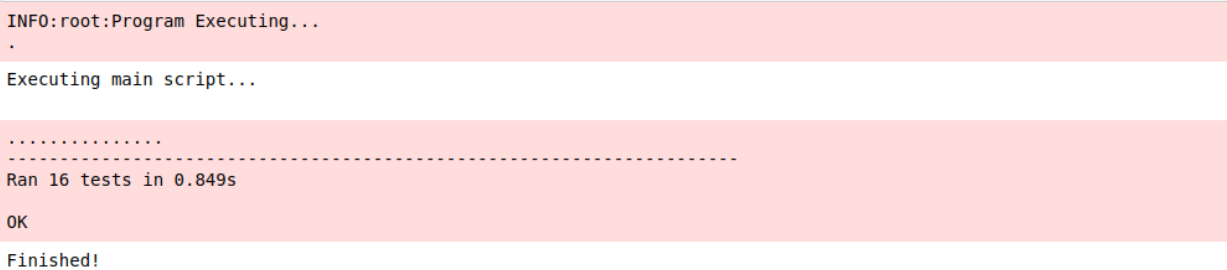


*If you would like to run the tests programmatically from your API, you can simply import the test script and unittest, and execute it with code:*

**

**

*All tests should pass without error:*

**

## Roadmap/Features

*As previously stated, MongoDB is a NoSQL database that specializes in efficient horizontal scaling and super-fast transactions. The mongo\_crud.py module makes working with MongoDB a simple and safe process. Future changes include the following:*

* *change MongoCrud ‘create’ function from the deprecated ‘insert\_one’ function to the recommended ‘insertOne’ function*
* *implement MongoCrud ‘update’ and ‘delete’ functions*
* *develop proprietary, modular API to interact with mongo\_crud.py module*
* *change connection and login details from hard-coded strings to a more secure methodology*
* *create an installer package for easy installation*

## Contact

*Matthew Pool*

*Computer Science, Southern New Hampshire University*

[*matthew.pool@snhu.edu*](mailto:matthew.pool@snhu.edu)

*April 7, 2024*