Janice Ainembabazi

Southern New Hampshire University

CS-340-18683-M01 Client/Server Development

Prof. Jeff Sanford

November 24th 2024

# CS 340 README Template

## About the Project/Project Title

This project is a Python module designed to interact with a MongoDB database, allowing for basic CRUD (Create, Read, Update, Delete) operations. For now, the module focuses on the "Create" and "Read" functionalities, which allow users to insert and retrieve records from a MongoDB database. The main goal is to create an easy-to-use Python tool for managing data in MongoDB, with plans to expand it to include update and delete functionalities in the future.

## Motivation

The motivation behind this project is to build a reusable Python module that can interact with MongoDB databases and perform essential CRUD operations. MongoDB is a flexible, NoSQL database that does not require a rigid structure, making it ideal for a wide range of use cases. By developing this module, I have practiced working with databases in Python and started creating a solid foundation for future enhancements, such as adding update and delete capabilities.

## Getting Started

To get this project running on your local machine, you will need:

* MongoDB installed and running. You can download MongoDB from its official website.
* Install the required Python dependencies. The two main libraries used in this project are PyMongo and BSON.
* Install these libraries using pip by running pip install pymongo and pip install bson. After that,
* Then you will need to set up MongoDB with user authentication using the username aacuser and the password (your chosen password).

With MongoDB up and running and the required libraries installed, you will be able to use the module to interact with the database.

## Installation

To use the software, you need to install MongoDB, PyMongo, and BSON. MongoDB can be downloaded from the official website, and the Python libraries can be installed via pip. PyMongo is the official Python library for interacting with MongoDB, while BSON is used for handling the data format that MongoDB uses. These tools were chosen because MongoDB is a flexible NoSQL database that fits the project’s needs, PyMongo simplifies the process of interacting with MongoDB, and BSON ensures compatibility between Python and MongoDB's data format.

## Usage

### Code Example

Here is a simple example of how the module works. First, you create a new instance of the AnimalShelter class. Then, you create a dictionary with the data you want to insert. After that, you call the create method to add the new record to the database. To retrieve records, you pass a query to the read method, which returns the matching records from the database.

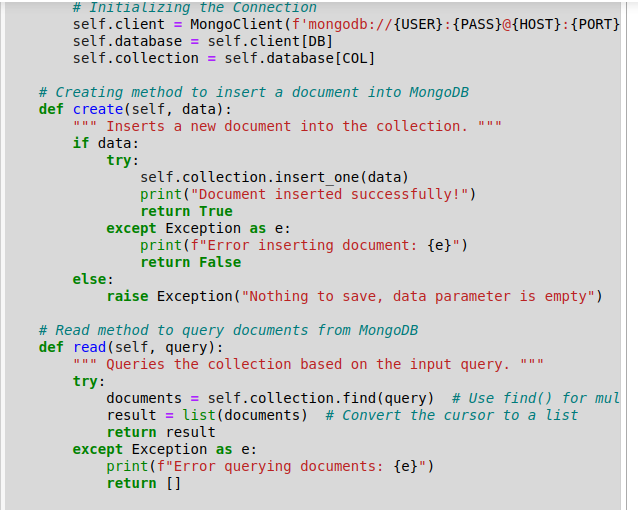
In my code, a new Goat named Billy is added to the database, and then all Goats are retrieved from the database using a query that searches for animals with the species "Goat."

### Tests

To test the create and read functionalities, you can run the following code. First, the create method inserts a new animal record. Then, the read method fetches records that match the query.

### Screenshots







## Roadmap/Features (Optional)

Looking ahead, I plan to add the following features to the module:

* **Update**: The ability to modify existing records in the database.
* **Delete**: The ability to remove records from the database.

There are currently no major known issues, but I plan to improve error handling and validation of user input in future updates.

## Contact

Janice Ainembabazi