# CS 340 README Template

## About the Project/Project Title

The Crud class is a Python utility designed to simplify interaction with MongoDB databases. It enables developers to perform basic CRUD (Create, Read, Update, Delete) operations efficiently while adding features to ensure data integrity and prevent schema violations.

## Motivation

Working with MongoDB often involves repetitive tasks such as validating connections, ensuring schema integrity, and handling data queries. This project was motivated by the need to streamline these tasks into a reusable, robust class while reducing the likelihood of common errors.

**Why Python And pymongo**

There are several reasons why Python Driver was chosen for this program.

1. Python and its driver have wide support and easy documentation, and it has the largest developer community support among open source programming languages
2. Pymongo is officially supported by the community and is stable
3. Python offers continuous update support and it is very unlikely that this support will be cut in the future due to its popularity
4. It has high backward and forward compatibility

## Getting Started

To use the Crud class in your project, follow these steps:

1. Clone or download the script file (connection.py) into your project directory.
2. Ensure you have Python installed on your system.
3. Install the required Python library: pymongo.

## Installation

***Prerequisites***

Python 3+

MongoDB server running locally or accessible remotely

pymongo library installed via:

Plotly Dash

Dash Leaflet

pip install pymongo

## *Importing Data*

## A screenshot of a computer program Description automatically generated

**Create A User**

**A screenshot of a computer program

Description automatically generated**

## Usage

*Use this space to show useful examples of how your project works and how it can be used. Be sure to include examples of your code, tests, and screenshots.*

* *Connecting to MongoDB*
* *Initialize the Crud object.*

*A close up of a word

Description automatically generated*

* *Set the required attributes (user, password, host, port, and database).  
  A white card with red text

  Description automatically generated*
* *Call the connect method to establish a connection*

*A screen shot of a computer code

Description automatically generated*

### Code Example

* **Creating a Record :** Use the create method to insert data into a collection. Ensure protectSheme is set appropriately.

**data = {"name": "Alice", "status": “Adopted”}**

**crud.create("animals", data)**

* **Reading Records :** Retrieve documents from a collection using the read method:

**query = {"age": {"$gte": 18}}**

**results = crud.read("users", query)**

**for document in results:**

**print(document)**

* **Update Record:** This function will update the records. 1st argument: Collection Name, 2nd argument: Data 3rd argument Condition, The function will throw an error when it cannot find the Collection name in the database. Also, condition and data must be of Dictionary type and must not be empty.

**data = {“age”:”old”}**

**condition = {"age": {"$gte": 10}}**

**result = crud.update("animals” , data,condition)**

**if result:**

**print(“Success’)**

**else:**

**print("Failed”)**

* **Delete Record :** This function is used to delete records. It takes 3 arguments. 1st argument: Collection Name, 2nd argument: Condition, 3rd Delete Many, The function will throw an error if it cannot find the Collection name in the database. Also, the condition is of the Dictionary type and should not be empty. The deleteMany argument is True by default, meaning it can delete many records, but if only one record is updated, setting this argument to False will delete a single record. Also, this method has a structure that prevents all records from being accidentally deleted. The safeDelete property is True by default, we can make it False manually, in this case you can leave the condition blank, but since all your records can be deleted, it is recommended to use it carefully.

**condition = {"age": {"$gte": 10}}**

**result = crud.update("animals” , condition,True)**

**if result:**

**print(“Success’)**

**else:**

**print("Failed”)**

### Tests

* ***Schema Protection*** *The protectSheme feature ensures data integrity by validating keys before inserting data.*

*crud.protectSheme = True*

*data = {"name": "Bob", "non\_existing\_field": "value"}*

*try:*

*crud.create("users", data)*

*except Exception as e: print(e) # Output: Key not matching column names.*

* ***Safe Delete*** *This feature prevents accidental deletion of all records, caution is advised when removing*

***crud.*** ***safeDelete= False***

**condition = {}**

**result = crud.update("animals” , condition,True)**

**if result:**

**print(“Success’)**

**else:**

**print("Failed”)**

### Screenshots

*A white background with black text

Description automatically generated*

*A screenshot of a computer code

Description automatically generated*

*A screenshot of a computer code

Description automatically generated*

*A screen shot of a computer code

Description automatically generated*

*A screen shot of a computer code

Description automatically generated*

*A screen shot of a computer code

Description automatically generated*

## A screenshot of a computer program Description automatically generated

## A screenshot of a computer code Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer program

Description automatically generatedA screenshot of a computer code

Description automatically generatedA screen shot of a computer code

Description automatically generatedA white background with black text

Description automatically generatedA screenshot of a computer code

Description automatically generatedA screen shot of a computer code

Description automatically generatedA screen shot of a computer code

Description automatically generatedA screen shot of a computer code

Description automatically generated

## Roadmap/Features (Optional)

* Add compatibility with additional MongoDB features (e.g., aggregation pipelines).
* Enhance error messages for better debugging.
* Implement connection pooling for improved performance.

**Web InterFace Integration**

Find the lines shown in the image below and change the username and password.

A white screen with black text

Description automatically generated

If there is a change in the standard information, check the information in the picture below.

A screenshot of a computer program

Description automatically generated

Run the code with shift enter, you will get a result similar to the image below, click the link there

A screenshot of a computer code

Description automatically generated

The code will work as follows

To see the location of the animal, select one of the radio buttons on the side, the map will appear below and you can filter the rescue animals from the section above

A screenshot of a dashboard

Description automatically generatedA screenshot of a map

Description automatically generated

**Implementation of the Program:**

1. A class was created to perform operations on MongoDB based on the client's requirements.
2. While designing the interface, the client's logo and a link to their website were added at the top.
3. The developer's signature was included.
4. A table was created to display the data using the created class.
5. A filtering section was added to filter the displayed data for suitable animals based on criteria provided by the client.
6. A pie chart was added to analyze the data in the table visually.
7. A map was integrated to display the location of the selected animal

A screenshot of a computer

Description automatically generated

**Usage:**

* When the page first loads, all data is listed. If a feature is selected from the filtering section, the data table and pie chart are updated according to this filter.
* If the radio button next to an animal is clicked, a map showing the animal's location appears below, next to the pie chart.

From the image below, it can be seen how the table and pie chart change when a filter is applied.

A screenshot of a computer

Description automatically generated

The functionality of the map that appears when an animal is selected can also be observed from the image below.

A screenshot of a computer

Description automatically generated

## Contact

Your name: Ali ARSLAN

ali.arslan@snhu.edu