**CRUD Python Module for MongoDB Website Integration**

I will utilize and improve a previously made README, since I’ve already developed an application that utilizes Python as “glue” between MongoDB and front-end.

## About the Project/Project Title

The creation of a Python module for smooth integration of MongoDB with a website is the goal of this project. I developed this CRUD (Create, Read, Update, and Delete) module as a full-stack developer to offer a straightforward, effective, and user-friendly interface for managing MongoDB databases on the backend of web applications.

## Motivation

The goal of this project is to make it easier to integrate MongoDB with web applications, making database management responsibilities less complicated and freeing developers to work on other elements of their projects. This CRUD Python module intends to make it easier to manage database activities in a more structured and effective manner, which will ultimately help to produce web applications that are more reliable and scalable. In this course I will be creating a Python web application will allow for a smoother transition to any other database implementation.

## Getting Started

The primary goal is to ensure you have the repository installed, to do this you have to:

1. Clone the repository:

git clone https://github.com/(example)/crud-python-mongodb-module.git

1. Change directory to the cloned repository:

cd crud-python-mongodb-module

## Installation

You will need the following tools:

As recommended by MongoDB, it is required that you install Python 3.6 or above, below is the Python website for you to download your version;  
1. Install Python (version 3.6 or higher is recommended):

Download and install Python from the official website (<https://www.python.org/downloads/>).

2. When you open the environment you will need to install the required Python libraries, you can use the requirements.txt to analyze the required libraries.

Open your terminal with the environment activated, and type; pip install -r requirements.txt.

You can also download MongoDB shell but it is not needed.

## Usage

The following examples demonstrate how to use the CRUD Python module;

1. Activate the environment, (IDEs may automatically do this for you).
2. Import the module by typing in; from crud\_module import MongoDB\_CRUD
3. Connect to your database using the parameters on initialization of the object, such as;

db = mongoDB\_CRUD("mongodb://%s:%s@localhost:45652 , username,password)

keep in mind that your username and passwords are in different parameters, not a tuple.

1. You are now able to perform CRUD operations on your database, you can utilize:

db.create(data) to add data into your database, this will return True or False based on success/failure.

You may also use the read\_all and read functionalities to return a cursor of all documents or simply one document(dictionary) with the requested data.

### Code Example

Graphical user interface, text

Description automatically generated with medium confidence

### Tests

You may run tests by using simple conditionals such as the one seeing above;

1. **if** shelter.create(data):
2. print("Success!")
3. **else**:
4. print("Failure!")

Further CRUD Examples:

Graphical user interface, text, application, email

Description automatically generated

## Tools and Rationale of Project

In this project, a web application for a dog rescue dashboard is developed using a variety of tools and technologies. The main applications are Python, MongoDB, and Dash by Plotly.

### MongoDB

MongoDB is a well-known NoSQL database that allows for flexible data storage, allowing for easily and retrieval of data without being restricted rigid schema found in SQL databases. MongoDB scales horizontally, which is very beneficial because as the amount of data grows, MongoDB distributes the load through multiple servers ensuring high performance. Additionally, Python was the chosen language for this project and MongoDB has a native Python driver(PyMongo) that enables seamless interaction between the database and Python code.

### Dash

Dash is a Python web application framework built on Flask, Plotly.js, and React.js. I chose it for this project because it provides a simple and efficient way to create interactive, web-based data visualizations.Dash hasa range of interactive components, such as radio items, sliders, drop-down menus that can be easily integrated into any project, radio items were used in this specific project. Dash provides a seamless interaction and integration with Plotly, Dash is built on top of Plotly which is a popular data visualization library for Python. Plotly allows developers to create sophisticated and interactive visualizations for the dashboard.

### Known issues

I am having issues with the output of the ProjectTwoDashboard in Apporto, this issue began since the new Firefox update profile that Apporto forced me to complete. I am attempting to adapt my code to my desktop computer so that I can run and include the output screenshots in here.

## Contact

Your name: Guilherme Rigaud