Module 4 ReadMe Assignment

William Neal

SNHU

CS340: Client/Server Development

Professor Kellogg

9/24/2023

* How clearly did each README file describe the **purpose** of their project? – Zircon has a break down of everything with a hyperlinked table of contents, screen shots of working code and a strong overview of features. BigFoot-dash-app looks like one of my ReadMe’s. Plotly Dash is somewhere in between. It has solid media showing functionality but not really enough to describe the code base.
* What information did the README files include about the **functionality** of their project? What was helpful about this information? What could be improved? – Zircon was fine and provided detailed screen shots of the code base. BigFoot provided no information at all, there was nothing there to even begin to understand functionality. Lastly Plotly, they showed nice image clips of the functionality and all the hyperlinks too me to outside sources but nothing really told me about the code base.
* What information did the README files include about the **tools** (software and libraries) that they used or how the user could **reproduce** the project? – BigFoot had add the stuff in requirements.txt which was super helpful. That is sarcasm if you can’t tell. Plotly talks about functionality with connecting to python’s back end but nothing really about how the technology works. Zircon is the only one who shows basic usage, dependencies, and how the API works. I think Zircon has the best read me of the three, Plotly has great media and outside sources but lacks code base knowledge. And lastly my spirit animal BigFoot, well they made a ReadMe like I would make a ReadMe, which is nothing at all.

# CS 340 README Template

*Use this template to complete your README file. When completing the template, keep the headings as they are so that your document has a clear organization. Remove the italicized prompt text after you have completed each section for a polished final document.*

## About the Project/Project Title

*Provide a little information about your project or an overview that explains* ***what*** *the project is about.*

## Motivation

*The objective of this project is to develop a CRUD class that inserts documents into a database.*

## Getting Started

*In order to use this you must first authenticate, connect to the database and then run the python script.*

## Installation

*First fight your way through snhu.com to Brightspace from there you need to find virtual lab in course section. Once there connect to apporto. Now connect to the DB from the terminal and enter the necessary credentials in order to authenticate. Once there run whatever scripts you can.*

## Usage

*I don’t actually understand why this would be useful at all, it’s also probably why my code didn’t work. I was getting an indentation error but I saw nothing wrong with the code. I used spyder to write the code, but I didn’t understand how to import it to the shell program. I also don’t understand why it would be very useful. Seeing as I did not get the code working, and I do not fully understand the importance of the assignment I can not truthfully answer the next couple of questions. If I could get some feedback on what I am doing wrong I would greatly appreciate it. I feel like I am back to week 1 with understanding, not a good feeling.*

### Code Example

*Show what the library does as concisely as possible. Developers should be able to figure out how your project solves their problem by looking at the code example. Make sure that your code is short and concise.*

### Tests

*Describe and show how to run the tests with code examples.*

### Screenshots

*Provide screenshots that demonstrate your work.*

## Contact

Your name: William Neal

* An **explanation of the purpose of the CRUD Python module –**

CRUD stands for create, read, update and delete. So currently our create should allow us to insert documents into our database and collection. I used create\_document for this operation. The read or read\_documents method allows a user to search the database very easily. Update is being able to query the database, find data based off of key value pairs and then update that data. Then of course of delete is exactly what it says. We find data in our database based off our query and remove its existence.

* An **explanation of how the module should be used**, including:
  + A description of the Python driver for Mongo that was used and why it was chosen
  + An explanation of the attributes and working functionality of the CRUD operations

So I used pymongo library I had tried a couple of different ones but everything kept failing and I could not get my module 4 assignment to actually run which was not much different here. Not sure why the VC is so clunky and buggy but alas. I chose pymongo because it’s the official mongoDB driver for python and felt like it should work. I started with an initialization of MongoDBCRUD, took in username, password, host, port, db, and collection. I then built create, read, update and delete. The connection set up and break down is handled automatically. The last thing I managed to do was build out my error handling which I used exception e and print statements for what was wrong.

* A **demonstration of the module’s functional operations**, including:
  + Screenshots of the MongoDB import execution. You took these screenshots in Step 1.
  + Screenshots of the user authentication execution. You took these screenshots in Step 2.
  + Screenshots of the CRUD functionality test execution. You took these screenshots in Step 4.

A computer screen shot of a computer screen

Description automatically generated

Connecting

A screenshot of a computer

Description automatically generated

Both files side by side

A computer screen with a message box

Description automatically generated

Refused to run

A screenshot of a computer

Description automatically generated

Tried in different file

A computer screen shot of a computer screen

Description automatically generated

Tried different type of file as well.

Spyder code:

import pymongo

class MongoDBCrud:

def \_\_init\_\_(self, username, password, host, port, database, collection):

self.username = 'aacuser'

self.password = 'iamterribleatcoding'

self.host = 'nv-desktop-services.apporto.com'

self.port = 31580

self.database\_name = 'aac'

self.collection\_name = 'animals'

self.client = pymongo.MongoClient(host, port, username=username, password=password)

self.db = self.client[database]

self.collection = self.db[collection]

def insert\_document(self, document):

try:

result = self.collection.insert\_one(document)

return True if result.inserted\_id else False

except Exception as e:

print(f"cant insert it mane: {str: (e)}")

return False

def find\_documents(self, query={}):

try:

return list(self.collection.find(query))

except Exception as e:

print("didnt find")

return []

def update\_documents(self, query, update\_data):

try:

result = self.collection.update\_many(query, {"$set": update\_data})

return result.modified\_count

except Exception as e:

print("couldnt update")

return 0

def delete\_document(self, query):

try:

result = self.collection.delete\_many(query)

return result.deleted\_count

except Exception as e:

print("couldnt delete")

return 0

def close\_connection(self):

self.client.close()

jupyter code:

from mongo\_crud import MongoDBCRUD

username = "aacuser"

password = "iamterribleatcoding"

host = "nv-desktop-services.apporto.com"

port = 31580

database = "aac"

collection = "animals"

crud = MongoDBCRUD(username, password, host, port, database, collection)

document\_to\_insert = {"keys": "values", "keys2": "values2"}

create\_result = crud.create\_document(document\_to\_insert)

print("Create result:", create\_result)

query = {"keys": "values"}

read\_result = crud.read\_documents(query)

print("Read result:", read\_result)

update\_query = {"keys": "values"}

update\_data = {"keys2": "value\_new"}

modified\_count = crud.update\_documents(update\_query, update\_data)

print("Modified count:", modified\_count)

delete\_query = {"keys": "values"}

deleted\_count = crud.delete\_documents(delete\_query)

print("Deleted counting:", deleted\_count)

crud.close\_connection()

A computer screen with many colorful text

Description automatically generated

A computer screen with many colorful text

Description automatically generated

A screenshot of a computer

Description automatically generated