# CS 340 README Template

Chris Bowles

CS-340-R4859

4/20/24

## About the Project/Project Title

This project is meant to demonstrate learned skills of how to utilize a Python CRUD module to bring in data from MongoDB and then perform queries on the data. This project will also demonstrate how to use Dash to create a dashboard that includes a project title, logo, and data table as well as widgets that help illustrate the data queries.

## Motivation

This project serves as a final project for the SNHU CS-340 Client/Server Development class to demonstrate the skills we have learned so far. This includes the usage of testing tools like Jupyter Notebooks and making queries on datasets that have been imported into MongoDB. We also are meant to demonstrate skills in Python coding for creating the methods used to make the queries.

## Getting Started

1. Initially, we loaded a dataset into MongoDB that we would use for this project. We then created a new user for the database with Read/Write access to only that database. This user authentication will be needed for our application. Then, we moved on to writing the Python application in Spyder.

*Screenshot of dataset loading into Mongo DB:*

A computer screen with white text

Description automatically generated

*Screenshot of new user being created in the Mongo shell:*

A screenshot of a computer screen

Description automatically generated

1. We then wrote our Python code that includes functions for making a connection to MongoDB and CRUD (Create, Read, Update, and Delete) functionality. See below for code:

A screen shot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated

A computer screen shot of a program code

Description automatically generated

A screen shot of a computer error

Description automatically generated

A computer screen shot of a computer error message

Description automatically generated

Our project makes use of the read function mainly to make search queries using the find() function from Mongo.

1. We then transitioned to Jupyter Notebooks to import source code for our testing and dashboard creation through Dash framework. We were given a basic outline but had to fix areas to meet rubric guidelines. For a complete walkthrough of our testing in Jupyter Notebooks, please see the Code Example and Tests section below.
2. One major problem we ran into when setting up the project was forgetting to drop the “\_id” column when updating the data tables in Jupyter Notebooks. The inclusion of this field was causing issues when running the Dash framework as it could not load this data through the dash core component. This was figured out through use of the debugging tool in Dash as we got a Type Error for trying to load the dataset with this column.

## Installation

Installation of MongoDB will be necessary; the server software can be installed for different operating systems such as Linux, MacOS, and Windows. MongoDB drivers for Python are needed since we are using Python to create an application that communicates with the server. One of the reasons for this is the use of Pandas library for Python as well as the Dash framework that we will be creating our dashboard with. Also, an IDE for coding in Python such as Spyder, or PyCharm is needed to develop the application. For testing purposes of the Python application, installation of Jupyter Notebooks is suggested. Through this software, we can use JupyterDash to build the dashboard and test our Python CRUD module. Dash also offers the components we want for this project such as pie charts and radio buttons for our data table filters.

## Usage

### Code Example and Tests

A screenshot of a computer program

Description automatically generated

First snippet of code above – we are importing our libraries and making a connection with the MongoDB through our Python CRUD module. We are also getting a list of all documents from the database and dropping the column for ObjectId as it causes issues with our table.

A screenshot of a computer program

Description automatically generated

Here we are creating our dashboard – the layout includes a title along with a subtitle that includes my name. We are adding an image to the dashboard for the company logo. The logo is placed in the project folder to make the path as simple as possible. We are also constructing a dash core component to include radio buttons that will be used to select our filters for our data table. The value gets passed into another function for updating the table through a MongoDB search query. A screen shot of a computer

Description automatically generated

Here we can see how the data table is constructed. There are several settings for column sorting and filtering as well as pagination (starting at page 0 and showing 10 rows per page). At the bottom of the screenshot we can see how the bottom of the dashboard is coded to display our data chart and geolocation chart.

**A screenshot of a computer program

Description automatically generated**

This function shows how the dashboard update function is coded. It takes in the filter-type variable which is defined when the user selects one of the radio buttons on the dashboard. Depending on the option they choose, there is another call to our read function in the Python module that makes a search query to our Mongo database. The information passed to the function is in the form of a search query used by the Mongo find() method. It is important to note that we need to drop the “\_id” column every time or we will get issues with updating the table.

**A screenshot of a computer code

Description automatically generated**

Here we can see the code for updating the graph when a change has been made to the viewable data on the table. This is what allows the pie chart to reflect the filtered data. The other function shown is for highlighting the columns a different color when selected.

**A screenshot of a computer program

Description automatically generated**

Finally, we have the function for updating the geolocation chart. This is dependent on two things: the data included in the table and out of that data, which row is selected. The function includes settings for dimensions of the geolocation chart, the starting center coordinates, zoom level, and then also includes the tile marker for the currently selected row data.

### Screenshots – Project Functionality

*A screenshot of a computer

Description automatically generated*

A screenshot of the dashboard upon launch. It defaults to the dataset that was initially loaded in before any filters have been selected. We can see the title of the dashboard at the top along with our unique identifier, the company logo, then our filtering options for types of rescue dogs, then our data in table format.

*A screenshot of a computer

Description automatically generated*

Here we see the widgets from our default dataset with the first row selected which is represented on the geolocation chart. Our pie chart reflects every animal breed on the table which is why the representation looks a little off. The geolocation chart gives the coordinates of the selected row in the table. Right now, it is showing the location of the first animal on the table as depicted by the field in bubble in the left most column. Let’s see what happens when we select our first filter option.

*A screenshot of a computer

Description automatically generated*

We can see that the dataset has been updated to only reflect the criteria of our search (refer to the testing file where we wrote our database queries). We see that the pie chart and geolocation charts have been updated as well.

*A screenshot of a computer

Description automatically generated*

The above screenshot shows the next filter for Mountain or Wilderness rescues. Our dataset has reduced drastically to match the new query and our two widgets have again updated.

*A screenshot of a computer

Description automatically generated*

The above screenshot shows the filter for Disaster Rescue or Individual Tracking. The dataset has reduced even further, and our two widgets have again updated.

*A screenshot of a computer

Description automatically generated*

Finally, we can see that when we choose to reset the values by selecting the last filter option, our table goes back to its default state, as do the widgets.

## Contact

Your name: Christopher Bowles

[christopher.bowles@snhu.edu](mailto:christopher.bowles@snhu.edu)

CS-340-R4859