Meagan Holub

Project 2

CS 340

**Read Me**

**About this project**

The purpose of this project was to create a software application that can work with existing data from the animal shelters to identify and categorize available dogs. Grazioso Salvare is seeking a software application that can work with existing data from the animal shelters to identify and categorize available dogs. Global Rain has contracted for a full stack development of this application, including a database and a client-facing web application dashboard through which users at Grazioso Salvare will access the database. In the initial phases of this development, you developed a database and a Python module enabling CRUD functionality for MongoDB. Additionally, Grazioso Salvare has requested that the code for this project be open source and accessible on GitHub, so that it may be used and adapted by similar organizations.

**Using Mongo DB**

Mongo was the database used to access and collect documents needed to complete this project. It is a Python friendly interface that makes it easy to access csv documents. Python can use database tools like SQL, but the syntax is so different that it can be cumbersome to switch back and forth. The selection tools when running the basic CRUD functions of a database are significantly more complex in a SQL database than they are with a Mongo DB, using Python.

**Getting Started**

1. Create a Mongo Database and create a database called AAC.
2. Create a user with read/write privileges to that AAC database.
3. Import the data from aac\_shelter\_outcomes.csv file

**Purpose of the CRUD Python module**

The purpose of CRUD in MongoDB is based off the fundamental operations of read, update, and delete (CRUD). Creating databases of document collections (AAC) and performing operations to retrieve specific documents from the database system. Using the mongo import tool I was able to create method that retrieved the needed documentation for this project. The class that is created has the purpose of

* Input arguments to function a set of key/value pairs in the data type to the MongoDB API call
* Read methods that queries for documents from a specified MongoDB database and collection
* Update methods that change documents from the specified MongoDB database
* Delete methods that removes, and updates specified documents from MongoDB database

**MongoDB Import**

Begin with /usr/local/bin/mongod\_ctl start-noauth. I then accessed the data base by typing MONGO.

Text

Description automatically generated

Graphical user interface, text, website

Description automatically generated

Text

Description automatically generated

**User Authentication**

After mongo is enabled and running I typed use admin which accessed the admin database and used the “db.createUser()” to create my admin account and AAC accuser account

While using admin database, create admin account and give the admin the ability to readWrite using a code like this example: db.createUser( {user: “J.Silva”, pwd: “password”, roles: [ { role: “userAdminAnyDatabase”, db: “admin”}, “readWriteAnyDatabase” ]}). Once this admin is created, an output of a successful creation should be displayed to acknowledge the admin was successful with the information pertaining to the admin. The same type of code can be used to create a user account for the AAC database, but the code will have to be modified in certain areas.

mongod\_ctl start-noauth

mongod\_ctl stop

to start and stop the database

## Installation

The tools you will need to run this include Jupyter Notebooks, Python for command line, and MongoDB. The installation of each is detailed in labeled sections right below this line.

**Jupyter Notebooks**: Jupyter can be installed from the command line in any major operating system using the simple instructions here: <https://jupyter.org/install>. For detailed instructions like Proxy servers for Windows, Mac, and Linux, follow the instructions here: <https://jupyterlab.readthedocs.io/en/stable/getting_started/installation.html>

**Python**: Detailed installation instructions for Python are available here: <https://realpython.com/installing-python/>. Once you have Python installed, you should be able to use this program from the Terminal on Mac or Linux or from the Command Prompt for Windows.

**MongoDB**: MongoDB comes in Community or Enterprise editions. Detailed instructions for the installation and downloading of MongoDB are available here: <https://docs.mongodb.com/manual/installation/>.

**Plotly**

Plotly must be imported in order to generate the proper charts. Plotly is a charting tool for Python applications and can be imported directly into your Python module from your Jupyter notebook. If you need a local copy of Plotly installed, see the documentation here: <https://www.journaldev.com/19692/python-plotly-tutorial#:~:text=Installation.%20To%20install%20plotly%2C%20open%20a%20terminal%20window,to%20install%20to%20collect%20dependencies%20and%20download%20them%3A>

**Dash**

Dash is a framework used to build web applications. You can import the Dash Core Components into your Jupyter notebook and you can install Dash using the following information: <https://pypi.org/project/dash/>

**Pandas**

Pandas is used in this web application as well. Pandas is a tool for Python that creates the data frames. Pandas has other dependencies and information that should be reviewed before use here: <https://pandas.pydata.org/pandas-docs/stable/getting_started/install.html>

**Reset (Show all Data)**



Graphical user interface, application

Description automatically generated

**Code Samples from Dashboard**

*Complex Query*

*![Text

Description automatically generated]()*

*Map Markers*

*![Text

Description automatically generated]()*

*Pie Chart*

*![Graphical user interface, text, application

Description automatically generated]()*

*Pie Chart*

Import view data

Names = data(breed) – obtain and transfer to list

Values = data(breed) – obtain occurrence counts and transfer to list

Return graph using the view data, names as the search key, and values as the pie slice values

**Example of CRUD code**

Graphical user interface, text, application, email

Description automatically generated

After successfully running all charts and maps I knew that my application was successful for the company. I have attached a zip file to submit of both my project two completed code and my original CRUD.py file made in the beginning of this process.