# Grazioso Salvare Application README

## About the Application

The purpose of this project is to meet the specified requirements outlined by the client. To enable Grazioso Salvare users to access their database effectively, we developed a comprehensive full-stack application. This application comprises two fundamental components: a Python-based backend that interfaces with a MongoDB database. The interaction with the database is facilitated through the utilization of the pymongo package. To handle data retrieval and modification within the database, the application relies on a CRUD capability script.

The dashboard itself serves as a platform offering diverse functionalities, including statistical insights, geographical data representation, and both filtered and unfiltered data views extracted from the animal shelter's database. Beyond this, the program has been constructed with a broader purpose: aiding Grazioso Salvare in identifying suitable candidates for search-and-rescue training. Additionally, the program has been designed to seamlessly integrate data from various shelter and rescue sources, further enhancing its utility for Grazioso Salvare.

## Getting Started

* Navigate to the MongoDB terminal.
* Import the CSV file named “aac\_shelter\_outcome.csv”.
* Type in the user credentials for authentication.
* Obtain access to Python to run the program.
* The supplementary files “CRUD.py” (backend) along with “ProjectTwo.ipynb” (frontend) will be needed.
* **NOTE:** All necessary libraries must be imported.

Besides the above, there may be times where the python shell, terminal, and/or Jupyter Notebook does not update or run, in that case, it will be imperative to refresh and/or restart the terminal or program.

## Installation

Access to the database being utilized requires MongoDB. Chosen for its versatility and flexibility, MongoDB serves as the model in development, adept at storing multiple objects and handling real-time data efficiently. The Dash framework structures the view and controls for the web app, serving as the user interface and enabling interaction with all project features.

For Python, a current version of Python shell and Jupyter Notebook is needed to run the “CRUD.py” and “ProjectTwo.ipynb” files. See the following for the necessary code lines:

## Usage

The code line "AnimalShelter(username, password).create(data)" is employed for the creation and addition of an animal record (data) into the database. Utilizing both the CSV file within MongoDB and the program aids in the creation and retrieval of animals from the database. On the server side or the back-end, the "create" function utilizes the "insert\_one" command within MongoDB to insert a new animal into the database. Similarly, the "read" function employs the "find" operator in MongoDB for retrieval of all animal records from the database that match the query entered by the user. Furthermore, the "update" function employs the "find" operator in conjunction with the “update\_one” in MongoDB for updating animal records within the database based on a query entered by the user. Lastly, the "delete" function employs the "delete" operator in MongoDB for deleting animal records within the database based on a query entered by the user. Supplementarily on the client-side or frontend, the “update\_dashboard” method was used to use the different filter options on the top of the dashboard in the data table based on the client’s, Grazioso Salvare, preferences in dogs. Next, the “update\_graphs” function was used to update the pie chart displaying the percentage of each breed of dog in the data table. Moreover, the “update\_styles” method was used to highlight the selected row by the user. Lastly, the “update\_map” function was utilized to generate the map on the map on the dashboard under the data table on the right of the pie chart. The map showed a location near Austin, TX as default, but updated based on the data table filtering. That being said, all of the frontend code or the dashboard was created using HTML and various Python libraries and frameworks such as Dash, JupyterDash, pandas, base64 (for the client logo), os, numpy, and matplotlib.

### Tests

As an example, a test is “AnimalShelter(username, password).read({"name": "AnimalName"})” which is used to test whether animal(s) with “AnimalName” exist and to display them all in a list of records or an empty list. See the screenshots below for an extensive list of tests.

### Screenshots

A screenshot of a computer screen

Description automatically generated

**Figure 1.** Imports a CSV file using the appropriate MongoDB import tool.

A screenshot of a computer screen

Description automatically generated

**Figure 2.** Ensures user authentication to the database and collection by creating user account.

## A screenshot of a computer code Description automatically generated

**Figure 3.** Testing all the CRUD methods of the application for validity.

A screenshot of a computer program

Description automatically generated

**Figure 4.** Testing all the CRUD methods of the application for invalidity.

****

**Figure 5.** This screencast shows the frontend or the dashboard of the project.

**Challenges**

One of the challenges that was faced during this project was not being able to create the CRUD.py file though the Python shell. Due to the lack of experience of using a terminal or Python shell it took some time to locate and use the Python shell effectively. Another challenge was with testing and loading in the CRUD.py file or the Python CRUD module within Jupyter Notebook. However, it turned out to be a quick fix when the import statement “from CRUD import AnimalShelter” was re-adjusted by adding the “()” after “AnimalShelter” and passing the username and password of the MongoDB user in the test statement “AnimalShelter(username, password).read({‘name’: ’Rex’})”, for example. Another major challenge was, perhaps, resolving errors during the client-side development and having the results show up visually on the frontend/dashboard. Due to the internet speed and Jupyter Notebook not updating, it took some time and thorough debugging to resolve all the errors.

## Contact

Your name: Rayyan Abdulmunib