# CS 340 README Template

About the Project/Project Title

Global rain is developing a full-stack application for Grazioso Salvare, complete with a web-based client and backend database, to access information about animals from shelters in Austin, Texas.

## Motivation

This application will aid in selecting animals to be used for search-and-rescue. There are certain requirements that an animal must meet before being considered for this specialized training, and this app will allow Grazioso Salvare to interpret the data from shelters around the Austin, Texas area.

## Getting Started

In order to run get this application up and running locally, follow these steps:

* Install the necessary packages in the *Installation* section.
* Login to MongoDB as an Admin through mongosh
* Set up a local user with read-write access for the “aac” database.
* Instantiate the AnimalShelter object in your testing or client code and use the object methods to create and read documents.
* The dashboard will now be usable via the local browser link at the bottom of the page.

## Installation

Hosting the backend of this application will require an Ubuntu Linux server running MongoDB v6.0.5, and it will need Python v3.9. The front-end will be run from the browser, so it will be necessary to have a compatible browser to view the client. MongoDB was used to make database querying more efficient for web-based access. Python was used to make middleware to connect the database to the client.

## Usage

Mongo DB was used in this project due to it being simple, lightweight, efficient, and easily scalable. Mongo DB also provides an easy-to-use API for Python to ease development and maintenance. Dash was also used to build the web page dashboard. This Python accessible API allows front-end browsers to be developed without the need for HTML, CSS, and JavaScript. It integrates well with the Mongo DB API as well with its event callbacks, which can be used for interactive elements on the client.

**Links**

Jupyter Notebooks: <https://jupyter.org>

Mongo DB: <https://www.mongodb.com>

Plotly Dash: <https://plotly.com/dash/>

Spyder IDE: <https://www.spyder-ide.org/>

Ubuntu Linux: <https://ubuntu.com/desktop>

### Code Example

Figure 1 and 2 show the middleware code. This program builds the CRUD style app that communicates with the database and the client. You can see depicted the code for creating, reading, updating, and deleting documents.

### Tests

Jupyter Notebooks was used to test this application. Figures 3 to 6 show some of the code used to build the dashboard, use the CRUD app, and display an interactive front-end. Figures 7 to 11 show the dashboard and the attempted interactivity.

### Screenshots

Figure 1: Create and Read Methods

A screenshot of a computer program

Description automatically generated with medium confidence

Figure 2: Update and Delete Methods

A screen shot of a computer program

Description automatically generated with low confidence

Figure 3: Imports, Crud Object, Logo

A screenshot of a computer

Description automatically generated

Figure 4: HTML Page Structure

A screenshot of a computer program

Description automatically generated with medium confidence

Figure 5: Table Update Logic

A screenshot of a computer program

Description automatically generated with medium confidence

Figure 6: Graph, Style, and Map Update Logic

A screenshot of a computer program

Description automatically generated with medium confidence

Figure 7: Dashboard at Start

A picture containing text, screenshot, line

Description automatically generated

Figure 8: Select Disaster Rescue/Individual Tracking

A picture containing text, screenshot

Description automatically generated

Figure 9: Select Mountain/Wilderness Rescue

A picture containing text, screenshot

Description automatically generated

Figure 10: Select Water Rescue

A picture containing text, screenshot

Description automatically generated

Figure 11: Select New Row

A picture containing text, screenshot

Description automatically generated

**Challenges**

I had a lot of challenges along the way with this project. I overcame many of them, but I was not successful in jumping every hurdle. For those that I did not manage, like making the page fully interactive, I would really like some help so that I understand these concepts for the future. I had a lot of trouble getting each of the elements to display on the page, like the pie graph and the radio buttons. I had to do a fair bit of research to figure out how to make the radio buttons work, but I never managed to get them to change the table. They seemed to be working from the back end, but nothing I did would get them to be interactive in the correct way. Unfortunately, due to the impossible workload I had from my other class (Linear Algebra), my full-time job that I’ve had to work late several days, and a doctor’s appointment I could not miss, I did not get the chance to start this project until Sunday. This made it so that I had no chance to reach out for help in a timely manner. I’ve truly put my best out there to get this app working, but I’m just not doing something right. Please let me know what I did wrong. I had to submit it now because this is the end of the time I have this week.

## Contact

Your name: Shawn Way