# CS 340 README

## About the Project/Project Title

The purpose of this project for our client Grazioso Salvare is to develop a software application that assists them in identifying and categorizing dogs that are suitable for search-and-rescue training. Grazioso Salvare has partnered with a nonprofit agency that operates five animal shelters around Austin, Texas that will provide the data.

## Motivation

The motivation behind the project is to create an application that can help our client identify and categorize dogs suitable for search-and-rescue training. This application will allow them to access and manage data from partner animal shelters such as age, breed, and proficiency in various rescue scenarios. The project is also open source so that other organizations with similar needs can use the application and contribute to it.

## Getting Started

To get a local copy up and running, follow these steps:

1. Install Python 3.
2. Install your preferred Python IDE.
3. Install MongoDB.
4. Install the PyMongo library.
5. Install Jupyter Notebook

## Installation

To install Python, go to the following site to download and install the latest version. [Download Python | Python.org](https://www.python.org/downloads/)

To install a Python IDE, I’ll use the Spyder IDE as an example. Go to the following website to download and install the exe. [Home — Spyder IDE (spyder-ide.org)](https://www.spyder-ide.org/)

To install MongoDB, go to the following site to download and install the exe. [Download MongoDB Community Server | MongoDB](https://www.mongodb.com/try/download/community-kubernetes-operator)

To install the PyMongo library, open command prompt in Windows and type “py -m pip install pymongo” without the quotes.

To install Juptyer Notebook, open command prompt in Windows and type “py -m pip install notebook” without the quotes.

## Usage

### Code Examples

Import Execution:

A computer screen with white text

Description automatically generated

User Authentication Execution: A screenshot of a computer program

Description automatically generated

Dashboard Functionality:

### Screenshots:

A screenshot of a computer

Description automatically generated

Here you can see the starting state of the dashboard for Grazioso Salvare. The default selection is set to “reset” to display all animals from the beginning. The dashboard table shows all the information for each animal. Below the table, there is a pie chart that gives a breakdown of each breed currently loaded onto the table. There is also a map that displays the location of one selected animal.

A screenshot of a computer

Description automatically generated

This is the dashboard when the “Water Rescue” filter is applied. With the database used in this eaxmple, there are only seventeen animals that meet the criteria for water rescue. The pie chart also shows that all currently available animals are the same breed.

A screenshot of a computer

Description automatically generated

This is the dashboard when the “Mountain Rescue” filter is selected. Here we can see that the database used only has five acceptable dogs and the pie chart shows four different breeds.

A screenshot of a map

Description automatically generated

This is the dashboard when the “Disaster Rescue” filter is selected. Only one valid animal is in the database with the pie chart reflecting that.

A screenshot of a computer

Description automatically generated

## This screenshot demonstrates all data being displayed again when the “Reset” filter is selected.

**Tools Used and Rationale:**

MongoDB was chosen because of its flexibility and ease of integration with Python. It’s also good for storage and retrieval of JSON documents.

The DASH framework was chosen because it provides the necessary structure for the web application’s view and controller components. It helps simplify the creation of interactive web applications with Python, which in turn allows for real-time updates and user-friendly interactions.

Jupyter Notebook was chosen because it allows users to create scripts to test their modules. It also makes it easier to demonstrate functionality and showcase it to other people.

**Steps Taken:**

1. Reviewed Dashboard Specifications Document to understand the required widgets and functionality for the project.
2. Connected to the MongoDB database using the “AnimalShelter” class and hardcoded username and password.
3. Created an unfiltered data table using MongoDB and the CRUD Python module.
4. Implemented interactive radio buttons for rescue type filtering using Dash components.
5. Developed callback functions to dynamically update the data table based on the selected rescue type.
6. Created a dynamic pie chart and geolocation chart using Dash components.
7. Designed the overall dashboard layout using HTML Div elements, including the company logo, filtering options, data table, and charts.
8. Tested components and functionality in Jupyter Notebook.
9. Deployed the dashboard using “app.run\_server()”

**Challenges and Solutions:**

The main challenge I encountered on this project was designing the user interface. I wasn’t sure how to properly implement the components I wanted because I was unfamiliar with the DASH framework. I was able to overcome this challenge by referring to DASH documentation.

## Contact

Matt Smith