# CS 340 README Template

# About the Project

“Grazioso Salvare Dashboard” is a full-stack Python application designed to help identify dogs suitable for search-and-rescue training. The application connects to existing shelter data (via a MongoDB database) and provides an intuitive, client-facing dashboard.

Key features include:

* Filtering and categorizing dogs based on breed, age, sex, and rescue type.
* Interactive data table with row selection and dynamic highlighting.
* Real-time pie charts visualizing breed distribution.
* Geolocation map showing the location of selected dogs from the shelter data.
* User-friendly interface is designed to reduce errors and minimize training time.

The dashboard is intended for use by Grazioso Salvare, which identifies and trains dogs for water rescue, mountain/wilderness rescue, disaster recovery, and human tracking operations.

# Motivation

This project was developed to support **search-and-rescue operations** by providing a tool that helps staff quickly identify ideal candidates from shelter data. By leveraging a simple, interactive dashboard, staff can make faster, more informed decisions while minimizing the potential for human error.

# Challenges

One of the main challenges encountered during the development of the Grazioso Salvare Dashboard was ensuring that the interactive dashboard components, such as the data table, pie chart, and map updated dynamically and stayed synced when filters were applied. This was overcome by refining the dash callback functions and using a single data source.

# Getting Started

## Prerequisites

* **Python 3.10+**
* **MongoDB** server (self-hosted or cloud)
* Required Python packages (can be installed via pip):
  + pip install pandas numpy matplotlib plotly dash jupyter-dash dash-leaflet

## MongoDB Setup

1. Import your dataset into MongoDB:
2. mongoimport --type=csv --headerline --db <DatabaseName> --collection <CollectionName> --drop <PathToCSVFile>
3. Create a user with read/write access:
4. mongosh
5. use admin
6. db.createUser({
7. user: "<Username>",
8. pwd: passwordPrompt(),
9. roles: [{ role: "readWrite", db: "<DatabaseName>" }]
10. })
11. Update your database connection credentials in CRUD\_Python\_Module.py:
12. username = "<YourUsername>"

password = "<YourPassword>"

# Installation

 Clone the repository:

git clone <repo-url>

cd <repo-directory>

 Install required Python packages:

pip install -r requirements.txt

 Ensure the MongoDB server is running and accessible.

 Launch the dashboard:

python Dashboard.py

or, if using Jupyter Notebook:

from jupyter\_dash import JupyterDash

app.run\_server(mode='inline')

# Usage

## Dashboard Features

* **Filter Dogs by Rescue Type:** Water, Mountain, Disaster, or reset to show all.
* **Interactive Table:** Select a row to highlight and display breed and other details.
* **Pie Chart:** Visualizes breed distribution in the table, hiding small slices unless hovered.
* **Map:** Displays the shelter location of the selected dog, with tooltips and popups for dog information.

## Code Example

Importing the CRUD class:

from CRUD\_Python\_Module import AnimalShelter

Initializing the CRUD class:

shelter = AnimalShelter()

Creating a new record in the database:

animalShelter.create(<data>)

Reading Entries from the database:

results = animalShelter.read(<filter>)

Updating Entries In the database:

count = animalShelter.update(<filter>, <data>)

Deleting Entries from the database:

count = animalShelter.delete(<filter>)

## Tests

You can test functions manually by creating test entries in your database

test\_record = {“animal\_id”:”TEST1234”,…}

shelter.create(test\_record)

print(shelter.read({‘animal\_id’:’TEST1234’})[0])

shelter.update({‘animal\_id’:’TEST1234’}, {“name”:” jake”})

shelter.delete({‘animal\_id’:’TEST1234’})

### Screenshots



A screenshot of a computer

AI-generated content may be incorrect.



A screen shot of a computer code

AI-generated content may be incorrect.



A black text on a white background

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.



A close-up of a computer screen

AI-generated content may be incorrect.



A screenshot of a computer

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

## Contact

Your name: Shamus Cerny