# CS 340 README

## About the Project/Project Title

This project was made to use Python to access and manipulate a database of animals for Grazioso Salvare, which is an innovative rescue-animal training company. The main functionality would be the essential CRUD (create, read, update, and delete) documents. The documents in this case are about animals and their details, which are all stored in a database.

## Motivation

The main motivation is to improve Grazioso Salvare’s ability to find dogs that are available for adoption.

## Getting Started

If you wanted to use this project locally you would have to:

1, Using MongoDB, import the csv file aac\_shelter\_outcome.csv.

2. Make or edit existing indexes to make querying easier.

3. Create a user account with read and write access (more or less an admin account).

4. Finally, access the program through Python.

## Installation

To use this locally, you’d need MongoDB in order to access the database, Python 3+/Jupyter to access the Python files and PyMongo driver to interface MongoDB through Python.

## Usage

### Code Example

For the time being, the code can only write and read the database. Later on, users will also be able to update animals already in the database, and then remove animals from the database.

Creating an animal entry:

*A white background with black text

Description automatically generated*

Reading an animal entry:

*A computer code with red text

Description automatically generated*

Updating an animal entry:

*A screen shot of a computer

Description automatically generated*

Removing an animal entry:

*A screen shot of a computer

Description automatically generated*

### Tests

As a simple test, a user could try to create an animal with the command:

AnimalShelter().create({ ‘animal\_id’: “test”})

Afterwards the user can then find the animal with the command:

AnimalShelter().read({ ‘animal\_id’: “test”})

Screenshots below offer more examples:

### Screenshots

Creating and finding an animal’s entry:

A screenshot of a computer

Description automatically generated

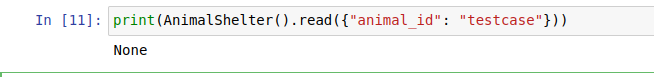
Updating an animal's entry:

*A screenshot of a computer

Description automatically generated*

Removing an animal's entry and verifying the removal:

**

**

Following more work, more functionality has been added to achieve a semi-functional dashboard:

A screenshot of a computer

Description automatically generated

A white rectangular object with text

Description automatically generated with medium confidence

A screenshot of a map

Description automatically generated

A screenshot of a computer

Description automatically generated

As it stands, the dashboard can sort via a specific column (by animal type, animal age, ascending or descending). While the filter for certain rescue animal types has been coded, it is not implemented in a fully functional capacity and will need to be further iterated upon.

MongoDB was quite beginner friendly and had a lot of resources to help. As well as a fairly flexible and straightforward control scheme, MongoDB’s ability to interface with Python made it the right choice for this project.

Challenges:

When trying to filter the database, the filter cannot return the necessary data type. It can find the data, but it can’t complete the return to the dashboard.

## Contact

Similoluwa Adeyemi