# CS 340 README

## About the Project/Project Title

The Grazioso Salvare company has been assigned to work on a project for an innovative international rescue-animal training company. Grazioso Salvare wants a database that aggregates animal shelter data from across many states so they can search for dogs that make good training candidates. Grazioso Salvare trains animals for multiple training scenarios, and the company needs an easy way to identify potential candidates quickly and efficiently. The client also wants a dashbard to be able to visualize the data and make it easier to understand

## Motivation

The motivation for this project was to provide a user with a way to search for and access profile information on potential candidates and metadata from those potential trainings. This process prior to project was manual so it required contacting animal shelters individual to check for candidates, and no common location was kept for any notes. The dashboard that will be created will help the client visualize their data with the use of different pieces of visual information such as charts and geolocation maps.

## Getting Started

## Installation

There were several tools used for this project. These include Python and mongoDB. These can both be downloaded online. Python is the programming language used to develop your code and mongoDB is the database that is used. When both are installed, the python interpreter can be launched by typing in python in the terminal. You also need to import pymongo at the top of your python files so you can mongoDB along with python. The pymongo driver is an intuitive API that allows you to access mongo databases with python code. Jupyter Notebook was also used which is an IDE used to develop your code. The dash framework was used to create the dashboard for this project. The dash framework is open-source Python framework used for building analytical web applications. This was used so large amounts of data can be visualized and analyzed by the client. This framework along with the other frameworks to build a web based application can be used by importing them at the top of your source code file. This is demonstrated in the screenshots below. The numpy and padas libraries were also imported into this dashboard project. These libraries serve to help us create data visualization. The pandas l library will help us create the data fames for this dashboard.

There are a few resources that helped me complete this that will be listed below.

[*https://dash.plotly.com/dash-core-components*](https://dash.plotly.com/dash-core-components)

[*https://dash.plotly.com/datatable*](https://dash.plotly.com/datatable)

[*https://dash-leaflet.herokuapp.com/*](https://dash-leaflet.herokuapp.com/)

## Usage

For the user to get started, they would need to sign in as the aacuser with the aacpassword that is used as user authentication. When create the create and read portions of the code, I had to make sure I was using the write method for the job so I had to look back to remember what they were. For the create function, I used the insert function on the animals database to create a new entry and I used the find function on the animals database to find specific files. The update and delete functions of the module were developed next. The update function takes search criteria and updates it in the database if that data is found. The delete function finds data in the mongo database and deletes it if it is able to find it.

When looking at the dashboard code, we can see different functionality such as the layout with the app that uses html. The dashboard can also be filtered based on the breed of the particular dog. It also shows geolocation data and a chart. This code can be enhanced to meet any future client needs by adding different data visualization elements.

### Code Example

db.createUser(

{

user: "aacuser",

pwd: "aac123",

roles: [ { role: "readWrite", db: "AAC" } ]

}

)

This code was used to create a user for the AAC database. This user has the ability to write and read files to and from this database. This application uses a mongo database. This crud module currently has the create and read features. The create feature is used to add new data to the database. The read function is used to read entries from the database.

### Tests

These tests are used to read and write to the mongo database.

def create(self, data):

if data is not None:

insertSuccess = self.database.animals.insert(data) # data should be dictionary

# Check insertSuccess for operation

if insertSuccess != 0:

return False

# default return

return True

else:

raise Exception("Nothing to save, because data parameter is empty")

# Create method to implement the R in CRUD.

def read(self, searchData):

if searchData:

data = self.database.animals.find(searchData, {"\_id": False})

else:

data = self.database.animals.find( {}, {"\_id": False})

# Return the dataset else let the error flow up

return data

### Screenshots

Text

Description automatically generatedIMPOTY Text

Description automatically generated

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Project 2 Screenshots

Graphical user interface, text, application

Description automatically generatedGraphical user interface, text, application

Description automatically generatedGraphical user interface, text, application, email

Description automatically generated

## Roadmap/Features (Optional)

## Contact

Your name: Tom Czubat