# CS 340 README Template

## About the Project/Project Title

This project is a client-facing web application that interfaces with a MongoDB database. It includes a Python module to allow for the client-side code to utilize MongoDB CRUD operations. The client-side code is a Dash web-application that uses a pandas dataframe to store the database.

## Motivation

International rescue-animal training company, Grazioso Salvare requested a software application to identify and categorize dogs. They want to work with existing data from animal shelters in order to find suitable candidates for rescue training. They also requested that the code from this project be open source so other organizations can adapt it for their needs.

## Getting Started

In order to get started with your own instance of the project:

1. Download the required software listed in the installation section
2. Connect to the MongoShell using an admin user by typing mongosh in the terminal
3. Create a new user with read-write permission for your database
4. Edit the variables in the AnimalShelter class located in animal\_shelter.py to reflect your environment
5. Open the Jupyter Notebook file and Import AnimalShelter from animal\_shelter.py
6. Edit the username and password to reflect the created username and password
7. Create a new instance of the AnimalShelter class by using:
   1. variable\_name = AnimalShelter(username, password)
8. Now you can utilize CRUD operations in the Python Shell using the AnimalShelter class
   1. Use the create method to add documents to the database
   2. Use the read method to search the database for documents
   3. Use the update method to modify documents in the database
   4. Use the delete method to remove documents from the database
9. Edit the Dash layout and callbacks to achieve the desired functionality
   1. HTML and Dash components can be added in the app.layout section
   2. Callbacks can be added to interact with the components and utilize CRUD operations

## Installation

The database side of the application requires MongoDB. Installation instructions can be found [here](https://www.mongodb.com/docs/manual/installation/).

The Python module requires Python, which can be downloaded [here.](https://www.python.org/downloads/)

The Python libraries used in the module are:

* [Pymongo](https://pypi.org/project/pymongo/)
* Bson(included in Pymongo)
* [Pprint](https://pypi.org/project/pprintpp/)

The client side dashboard is a Jupyter Notebook. Instructions for installing Jupyter can be found [here.](https://jupyter.org/install)

Instructions for installing and utilizing Dash can be found [here.](https://dash.plotly.com/installation)

## Usage

### Code Example

* AnimalShelterClass:
  + Create a new instance of AnimalShelter class:
    - animalShelter = AnimalShelter()
  + Add a document to the database:
    - animalShelter.create(document)
  + Read the database to find a matching document:
    - animalShelter.read(document)
  + Display every document in the database:
    - animalShelter.read(None)
  + Update documents that match a query in the database:
    - animalShelter.update(query, modification)
  + Delete documents that match a query from the database:
    - animalShelter.delete(query)
* Dash:
  + Update dataframe based on query
    - df = pd.DataFrame.from\_records(db.read({query}))
    - df.drop(columns=['\_id'],inplace=True)
    - return df.to\_dict(‘records’)

### Screenshots

Create:

A screenshot of a computer

Description automatically generated

Read:

*.* *A screenshot of a computer

Description automatically generated*

Update:

*A screenshot of a computer

Description automatically generated*

Delete:

*A screen shot of a computer

Description automatically generated*

Dashboard without any filters:*A white and black dashboard with black text

Description automatically generated with medium confidence*

After Selecting Water Rescue:

A white background with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

Mountain or Wilderness Rescue:

A white sheet with black text

Description automatically generated

A screenshot of a map

Description automatically generated

Disaster or Individual Tracking:

A white sheet with black text

Description automatically generated

A screenshot of a map

Description automatically generated

Resetting the dataframe:

A screenshot of a web application

Description automatically generated

A screenshot of a map

Description automatically generated

## Contact

Adam Koenig