# CS 340 README Template

## About the Project/Project Title

*This project demonstrates how the scripting language Python can be used with MongoDB to create and read documents in an already existing cluster.*

## Motivation

*This project is important as it will help us to automate the manipulation of clusters and collections as well as allow us the ability to take a further look into the different abilities that are associated with different authentications and privileges. Specifically, this will allow us to view and modify various aspects of Grazioso Salvare, an animal training company. This will allow them to better find animals that they can effectively train.*

## Getting Started

*In order to start this project, you want to ensure that you have access to the cluster itself. This means that you must have access granted from a db admin via an authenticated account.*

## Installation

*You will need the following:*

*Jupyter Notebook*

*This will allow you to run the test scripts*

pip install notebook

*Python 3.x*

[*https://www.python.org/downloads/*](https://www.python.org/downloads/)

*The custom classes used to interact with the database is written in Python*

*MongoDB*

*This is what contains/accesses the database as well as the various users*

*PyMongo*

*This is what allows us to use Python to interact with the database*

## Usage

### Code Example

*def create(self, data):*

*if data is not None:*

*if data:*

*self.database.animals.insert\_one(data)*

*return True*

*else:*

*return False*

*def read(self, search): #Checks if seach is null*

*if search is not None:*

*if search:*

*result = self.database.animals.find(search)*

*return result*

*else:*

*e = "Nothing to search, because search parameter is empty"*

*return e*

### Reproduction

*To reproduce, MongoDB must be installed, the CSV must also be installed and imported, the user authentication must be implemented, the CRUD Python module must be implemented.*

### Tests

*The following test should be run in Jupyter Notebook:*

*Text

Description automatically generated*

Steps were also take to ensure that the user of this program was able to interact with it in an intuitive manner. This measns that we needed to incorporate a user interface on the front end to interact with the backend. A Jupyter notebo0ok program was used to include this functionality. Users are able to filter the data as needed. This includes “dash” incorporation, which allows us to also include various “widgets”, such as a map. These were all required functions of the program. MongoDB was used here as we were able to incorporate python code into it through the jupyter notebook. IT is imperative that the user has all required elements required for the program to function properly.

## Contact

Russell Pallas