

# CS-340 Project Two (Grazioso Salvare) Dashboard's README File

## Project Description

The aim of this project is to create a web-based dashboard that displays data visualizations and allows for interactive exploration of a dataset. The dashboard is built using the Dash framework in Python, and the data is stored in a MongoDB database.

## Required Functionality

The required functionality of the project includes:

**Data loading:** The dashboard should load data from a MongoDB database and perform any necessary pre-processing steps.

**Data visualization:** The dashboard should display data visualizations, such as scatter plots, line charts, and bar charts, that allow users to explore the data.

**Interactivity:** The dashboard should allow users to interact with the data by selecting different variables, filtering data, and zooming in on specific parts of the visualization.


**Deployment:** The dashboard should be deployed to a web server so that it can be accessed by users.

## Screenshots/Screencast

Screenshots of the dashboard in action to demonstrate that the required functionality has been achieved.

Interactive Filter Options and Interactive Data Table

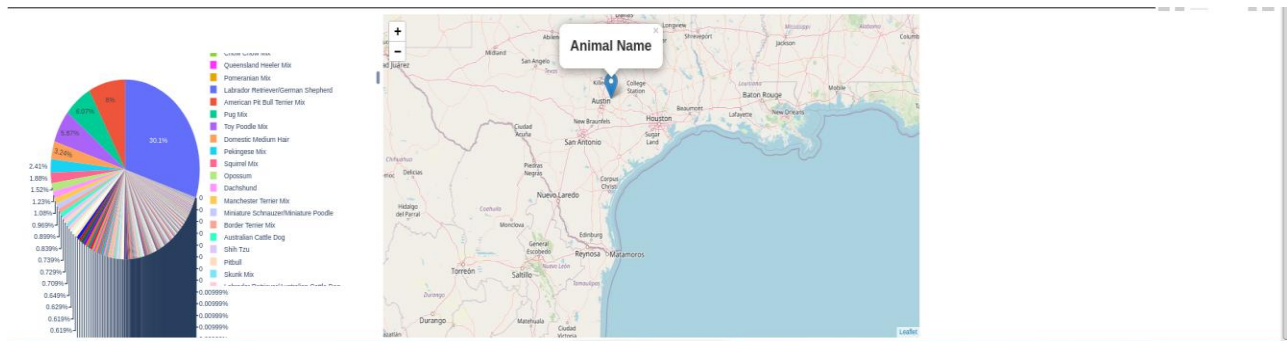
**SNHU CS-340 Dashboard**  
Created by Kennedy Uzoho on 02/23/2023



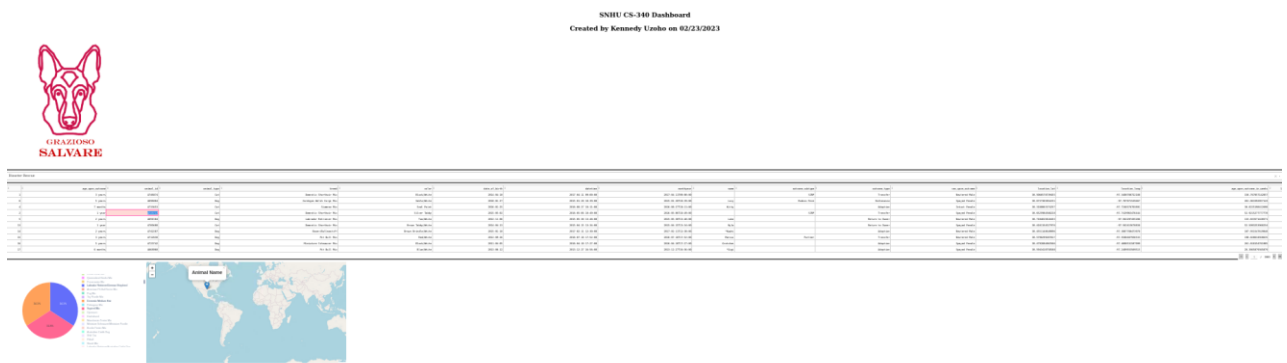
| Select...       |          |         |     |                         |                     |            |                     |                     |          |         |                 |               |                  |                   |                  |
|-----------------|----------|---------|-----|-------------------------|---------------------|------------|---------------------|---------------------|----------|---------|-----------------|---------------|------------------|-------------------|------------------|
| Water Rescue    |          |         |     |                         |                     |            |                     |                     |          |         |                 |               |                  |                   |                  |
| Mountain Rescue |          |         |     |                         |                     |            |                     |                     |          |         |                 |               |                  |                   |                  |
| Disaster Rescue |          |         |     |                         |                     |            |                     |                     |          |         |                 |               |                  |                   |                  |
| Reset           |          |         |     |                         |                     |            |                     |                     |          |         |                 |               |                  |                   |                  |
| 2               | 1 year   | A725717 | Cat | Domestic Shorthair Mix  | Silver Tabby        | 2015-05-02 | 2016-05-06 10:49:00 | 2016-05-06 10:49:00 |          | SCP     | Transfer        | Spayed Female | 30.4525984568228 | -97.7413993476444 | 52.9215277777778 |
| 5               | 2 years  | A691584 | Dog | Labrador Retriever Mix  | Tan/White           | 2012-11-06 | 2015-05-30 13:48:00 | 2015-05-30 13:48:00 | Luke     |         | Return to Owner | Neutered Male | 30.7184015618433 | -97.562297435286  | 133.653571428571 |
| 13              | 1 year   | A708488 | Cat | Domestic Shorthair Mix  | Brown Tabby/White   | 2014-04-13 | 2015-04-15 13:34:00 | 2015-04-15 13:34:00 | Nyla     |         | Return to Owner | Spayed Female | 30.4181154527976 | -97.562415678838  | 52.5093253968254 |
| 14              | 2 years  | A742287 | Dog | Boxer/Bulldozer         | Brown Brindle/White | 2015-01-18 | 2017-02-11 12:30:00 | 2017-02-11 12:30:00 | *Kawhi   |         | Adoption        | Neutered Male | 30.4551148648996 | -97.3887788473978 | 107.931547619848 |
| 15              | 3 years  | A712638 | Dog | Pit Bull Mix            | Red/White           | 2012-09-26 | 2016-07-10 17:52:00 | 2016-07-10 17:52:00 | Marcus   | Partner | Transfer        | Neutered Male | 30.578229287017  | -97.5588487938533 | 198.820634928635 |
| 16              | 5 years  | A723742 | Dog | Miniature Schnauzer Mix | Black/White         | 2011-04-05 | 2016-04-10 17:27:00 | 2016-04-10 17:27:00 | Gretchen |         | Adoption        | Spayed Female | 30.479284863566  | -97.4888531587999 | 261.818154761985 |
| 17              | 6 months | A668988 | Dog | Pit Bull Mix            | Blue/White          | 2013-06-12 | 2013-12-27 16:56:00 | 2013-12-27 16:56:00 | *Gigi    |         | Adoption        | Spayed Female | 30.5943418758588 | -97.2489933569515 | 28.3865879365879 |

1 / 2005

## Charts (Pie Chart and Geolocation Chart Next to it)



## The entire Web Application Dashboard with Interactive Data Table



## Tools Used

**Python:** We use Python as the main programming language for this project, including data pre-processing, data visualization, and web application development.

**Dash framework:** We use the Dash framework to build the web application, which provides a simple and flexible way to create web-based data visualizations and dashboards.

**Plotly:** We use the Plotly library to create interactive data visualizations, which provides a powerful and flexible way to create customized data visualizations.

## MongoDB

MongoDB is a popular NoSQL database that is known for its flexibility, scalability, and ease of use. It was chosen as the model component for the project for several reasons:

*Flexibility:* MongoDB's document-based data model makes it easy to store and retrieve complex data structures. This is particularly useful for web applications that require a high degree of flexibility in their data models.

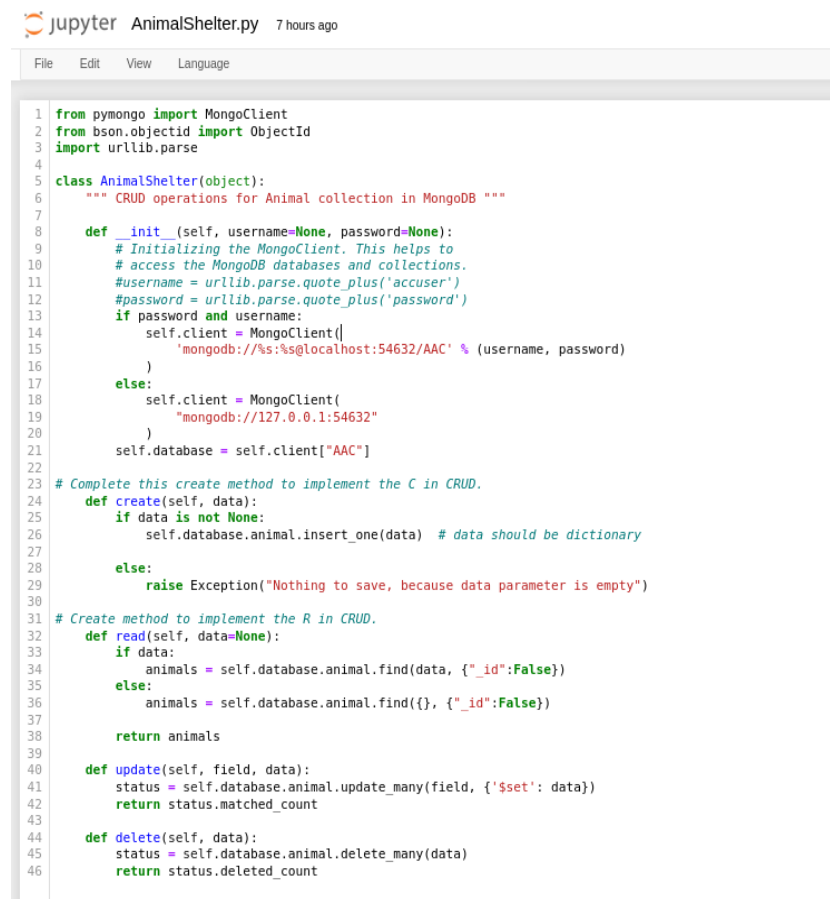
*Scalability:* MongoDB is designed to scale horizontally, which means that it can handle large volumes of data and high levels of traffic without sacrificing performance.

*Ease of use:* MongoDB's simple query language and intuitive API make it easy to work with, even for developers who are new to NoSQL databases.

In terms of interfacing with Python, MongoDB provides a number of useful features and capabilities:

**Python driver:** MongoDB provides an official Python driver that makes it easy to connect to a MongoDB database and perform CRUD (create, read, update, delete) operations.

Here is a snippet of the CRUD module:



The screenshot shows a Jupyter Notebook interface with a file named 'AnimalShelter.py' opened 7 hours ago. The code defines a class 'AnimalShelter' that interacts with a MongoDB database. It includes methods for creating, reading, updating, and deleting data. The code is as follows:

```

1 from pymongo import MongoClient
2 from bson.objectid import ObjectId
3 import urllib.parse
4
5 class AnimalShelter(object):
6     """ CRUD operations for Animal collection in MongoDB """
7
8     def __init__(self, username=None, password=None):
9         # Initializing the MongoClient. This helps to
10        # access the MongoDB databases and collections.
11        #username = urllib.parse.quote_plus('accuser')
12        #password = urllib.parse.quote_plus('password')
13        if password and username:
14            self.client = MongoClient(
15                "mongodb://%s:%s@localhost:54632/AAC" % (username, password)
16            )
17        else:
18            self.client = MongoClient(
19                "mongodb://127.0.0.1:54632"
20            )
21        self.database = self.client["AAC"]
22
23    # Complete this create method to implement the C in CRUD.
24    def create(self, data):
25        if data is not None:
26            self.database.animal.insert_one(data) # data should be dictionary
27
28        else:
29            raise Exception("Nothing to save, because data parameter is empty")
30
31    # Create method to implement the R in CRUD.
32    def read(self, data=None):
33        if data:
34            animals = self.database.animal.find(data, {"_id": False})
35        else:
36            animals = self.database.animal.find({}, {"_id": False})
37
38        return animals
39
40    def update(self, field, data):
41        status = self.database.animal.update_many(field, {'$set': data})
42        return status.matched_count
43
44    def delete(self, data):
45        status = self.database.animal.delete_many(data)
46        return status.deleted_count

```

**BSON:** MongoDB uses BSON (Binary JSON) to represent data in a binary format, which is more efficient than using traditional JSON. This makes it faster and more efficient to read and write data from Python.

Aggregation framework: MongoDB's aggregation framework provides a powerful set of tools for querying and analyzing data. This is particularly useful for applications that require complex data analysis or reporting.

Overall, MongoDB's flexibility, scalability, and ease of use, combined with its strong support for Python, make it an excellent choice for the model component of web applications.

### **Steps Taken**

To complete the project, we followed the following steps:

We collected and pre-processed the data.

We designed and built the dashboard using the Dash framework, including data visualization and interactivity features.

We connected the dashboard to the MongoDB database, allowing for data retrieval and storage.

### **Challenges**

During the development process, we encountered several challenges, such as data pre-processing and creating custom data visualizations. To overcome these challenges, we used online resources, including documentation and forums, and consulted with other developers.

### **Resources and Software Applications:**

#### **Software**

Anaconda: Anaconda is a distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment.

Jupyter Notebook: The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience.

#### **Resources**

<https://docs.mongodb.com/>

<https://dash.plotly.com/introduction>

<https://pandas.pydata.org/docs/>

<https://docs.python.org/3/>

<https://www.w3schools.com/css/default.asp>

<https://www.w3schools.com/python/pandas/default.asp>

Austin Animal Center Outcomes Spreadsheet csv dataset – Reference: Austin Animal Center. (2020). Austin Animal Center Outcomes [Data set]. City of Austin, Texas Open Data Portal. <https://doi.org/10.26000/025.000001>

**Contact**

Kennedy Uzoho

[Kennedy.uzoho@snhu.edu](mailto:Kennedy.uzoho@snhu.edu)