CS 340 README Template

About the Project/Project Title
This project involves the development of a Python module that enables basic CRUD (Create, Read, Update, Delete) operations for interacting with a MongoDB database. This module will be utilized as the backend for a web application

Motivation
The motivation behind this project is to provide a practical learning experience in connecting a web application to a database using Python. The CRUD module focuses on the essential operations needed to manage database records, providing a foundational understanding of backend development and database interaction.

Getting Started
To get a local copy up and running, follow these simple steps:

1. Ensure MongoDB is installed and running on your local machine.
2. Create a database named AAC and a collection named animals.
3. Set up a user account for MongoDB with the appropriate permissions.
4. Import the provided dataset into the MongoDB collection.
5. Clone the repository containing the CRUD module.
6. Install the necessary Python libraries.

Installation

The following tools and libraries are required to use the software:

- **Python**: The primary programming language used for the project.
- **PyMongo**: A Python library for MongoDB operations.
- **Jupyter Notebook**: For testing and demonstrating the module.

Steps to Install

1. **MongoDB Installation**: Follow the instructions from the MongoDB official website to install MongoDB on your system.
2. **Clone the Repository**:

   bash
   git clone <repository-url>

3. **Install Dependencies**:

   bash
   pip install pymongo dash dash-leaflet plotly

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

Usage
Below are examples of how to use the CRUD module.

Create Operation

python
```
from crud_module import AnimalShelter

# MongoDB connection variables
USER = 'aacuser'
PASS = 'SNHU1234'
HOST = 'nv-desktop-services.apporto.com'
PORT = 31580
DB = 'aac'
COL = 'animals'

# Instantiate the CRUD class
shelter = AnimalShelter(USER, PASS, HOST, PORT, DB, COL)

# Test data for insertion
test_data = {
    "animal_id": "A123456",
    "name": "Buddy",
    "species": "Dog",
    "age": 5,
    "adopted": False
}

# Test the create method
create_result = shelter.create(test_data)
print("Create Result:", create_result)
```

Read Operation

python
```
# Test the read method
query = {"species": "Dog"}
read_result = shelter.read(query)
print("Read Result:", read_result)
```

Update Operation

```
# Test the update method
update_data = {"name": "Buddy Jr."}
update_result = shelter.update({"animal_id": "A123456"}, update_data)
print("Update Result:", update_result)

# Verify the update
updated_read_result = shelter.read({"animal_id": "A123456"})
```

Note: This template has been adapted from the following sample templates: [Make a README](), [Best README Template](), and [A Beginners Guide to Writing a Kickass README]().

print("Updated Read Result:", updated_read_result)

Delete Operation

```
# Test the delete method
delete_result = shelter.delete({"animal_id": "A123456"})
print("Delete Result:", delete_result)

# Verify the deletion
deleted_read_result = shelter.read({"animal_id": "A123456"})
print("Deleted Read Result:", deleted_read_result)
```

Dashboard Usage

To run the dashboard, execute the Jupyter Notebook file ProjectTwoDashboard.ipynb. The dashboard includes the following components:

1. **Interactive Filtering Options**: Radio buttons to filter data based on the type of rescue.
2. **Data Table**: Displays the filtered data dynamically.
3. **Geolocation Chart**: Shows the location of selected animals on a map.
4. **Pie Chart**: Displays the distribution of breeds for the selected rescue type.

Running the Dashboard

```
# Ensure MongoDB is running and execute the notebook
jupyter notebook ProjectTwoDashboard.ipynb
```

Code Example

```
# Example usage of the AnimalShelter class
from crud_module import AnimalShelter

shelter = AnimalShelter(USER, PASS, HOST, PORT, DB, COL)

# Insert a document
shelter.create({"animal_id": "A123457", "name": "Charlie", "species": "Cat", "age": 3, "adopted":
True})

# Query documents
results = shelter.read({"species": "Cat"})
for result in results:
    print(result)

# Update a document
shelter.update({"animal_id": "A123457"}, {"name": "Charles"})

# Delete a document
shelter.delete({"animal_id": "A123457"})
```

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

Tests

To run tests, use the following steps:

1. ***Create Test Data**: Prepare test data for the create operation.*
2. ***Run Create and Read Methods**: Execute the CRUD operations and verify the results.*

Example test script:

```
import unittest
from crud_module import AnimalShelter
from bson.objectid import ObjectId

class TestAnimalShelter(unittest.TestCase):

    @classmethod
    def setUpClass(cls):
        cls.shelter = AnimalShelter(
            user='aacuser',
            password='SNHU1234',
            host='nv-desktop-services.apporto.com',
            port=31992,
            db='AAC',
            collection='animals'
        )
        cls.test_data = {"animal_id": "A123458", "name": "Max", "species": "Dog", "age": 2, "adopted":
False}

    def setUp(self):
        # Ensure the test data has a unique _id
        self.test_data["_id"] = ObjectId()
        # Clean up the collection before each test
        self.shelter.collection.delete_many({"animal_id": "A123458"})

    def test_create(self):
        result = self.shelter.create(self.test_data)
        self.assertTrue(result)

    def test_read(self):
        self.shelter.create(self.test_data)
        result = self.shelter.read({"animal_id": "A123458"})
        self.assertEqual(len(result), 1)
        self.assertEqual(result[0]["name"], "Max")

    def test_update(self):
        self.shelter.create(self.test_data)
        update_data = {"name": "Maxwell"}
        result = self.shelter.update({"animal_id": "A123458"}, update_data)
        self.assertTrue(result)
```

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

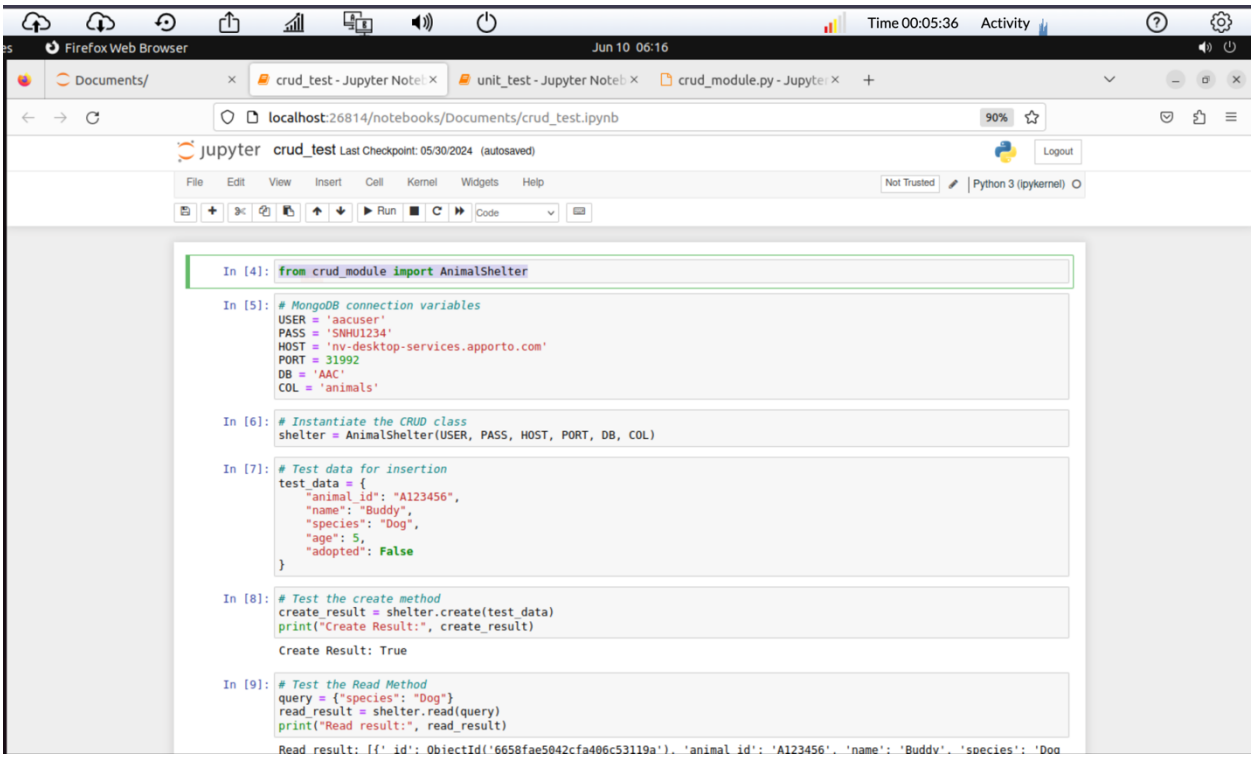
```
        updated_doc = self.shelter.read({"animal_id": "A123458"})
        self.assertEqual(updated_doc[0]["name"], "Maxwell")

    def test_delete(self):
        self.shelter.create(self.test_data)
        result = self.shelter.delete({"animal_id": "A123458"})
        self.assertTrue(result)
        deleted_doc = self.shelter.read({"animal_id": "A123458"})
        self.assertEqual(len(deleted_doc), 0)

if __name__ == '__main__':
    unittest.main(argv=[''], verbosity=2, exit=False)
```

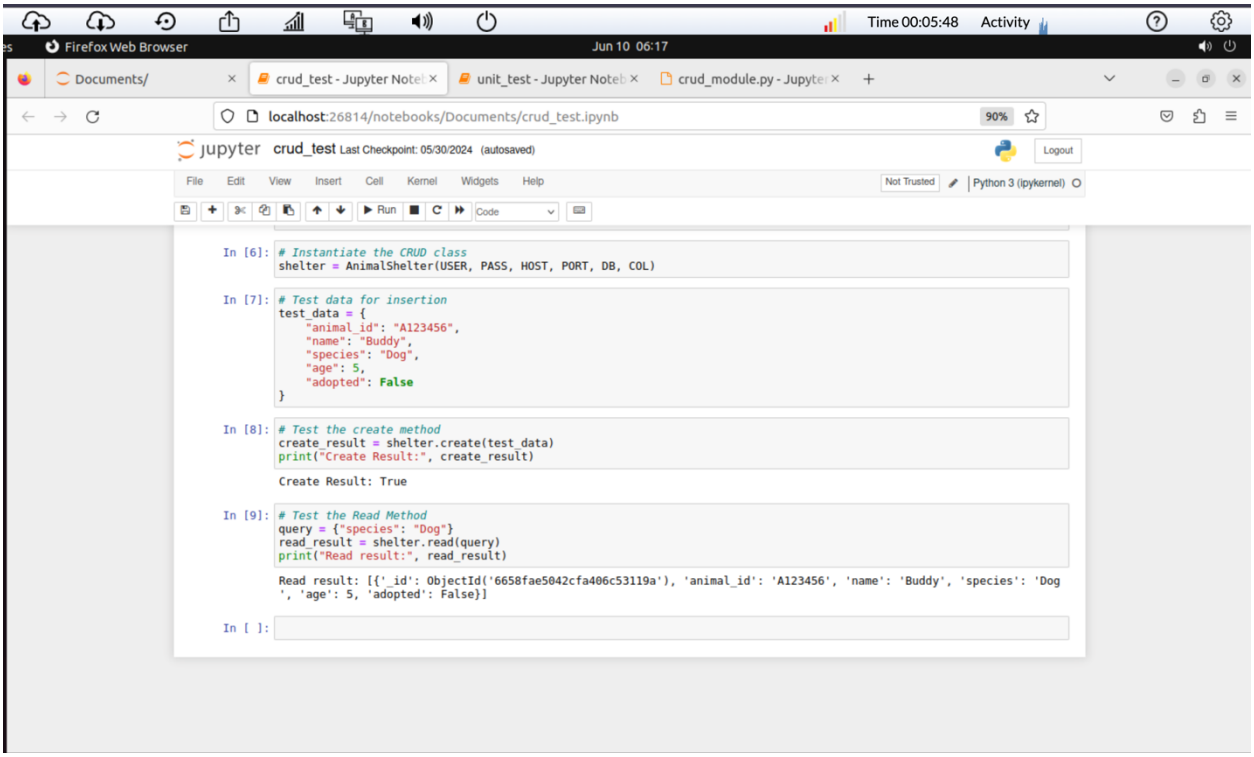
Screenshots

Crud Test script

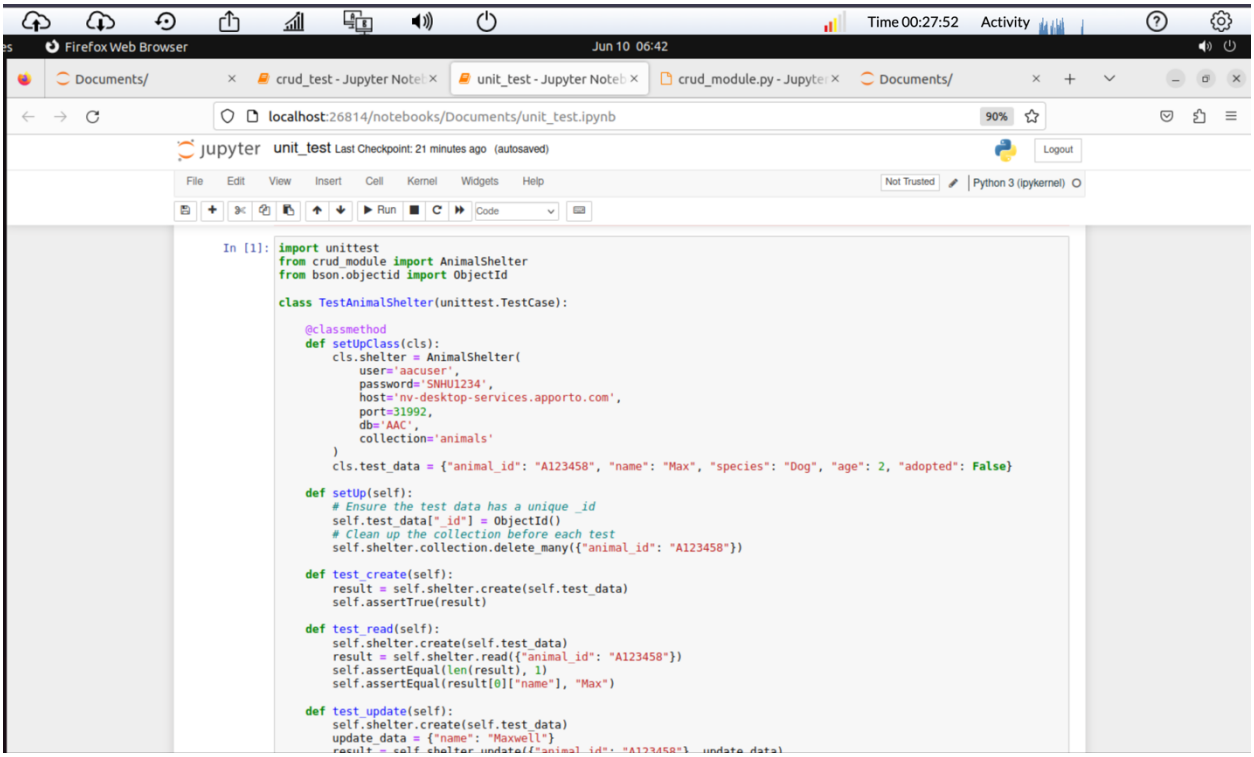


Note: This template has been adapted from the following sample templates: [Make a README](), [Best README Template](), and [A Beginners Guide to Writing a Kickass README]().

Crud Test Results

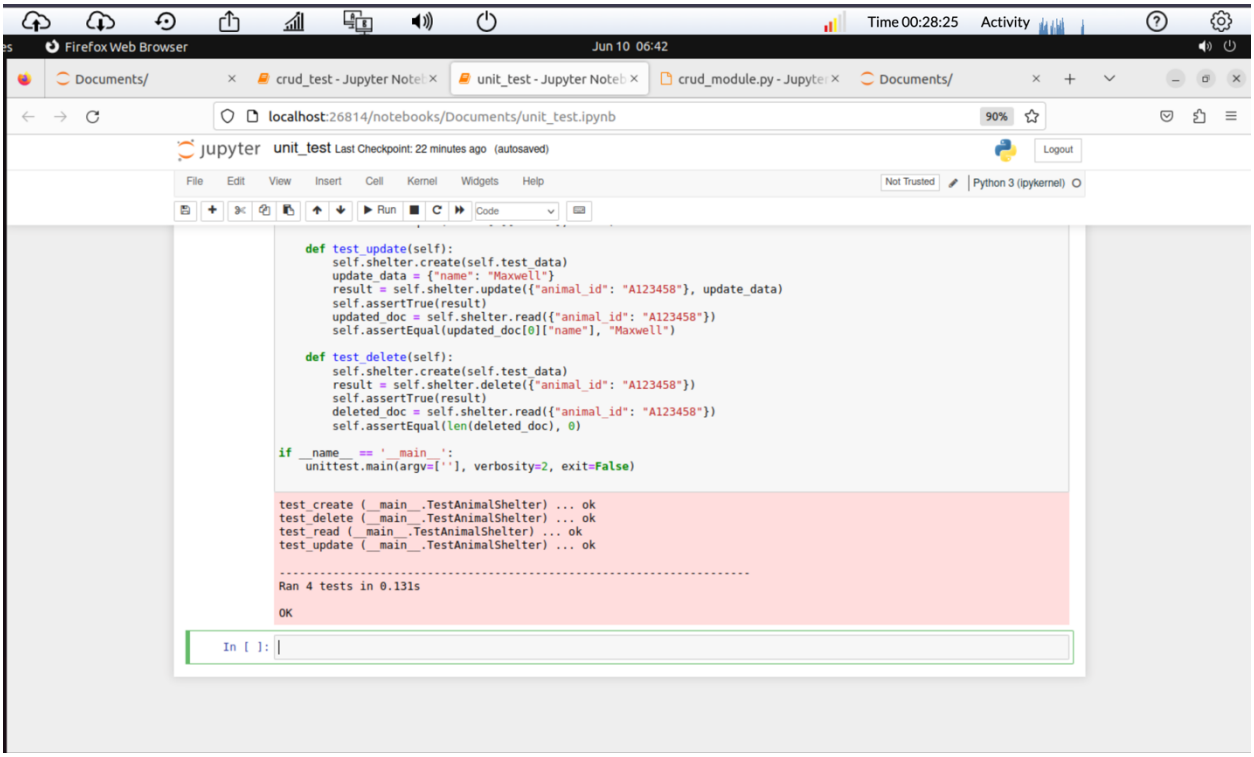


Unit Tests

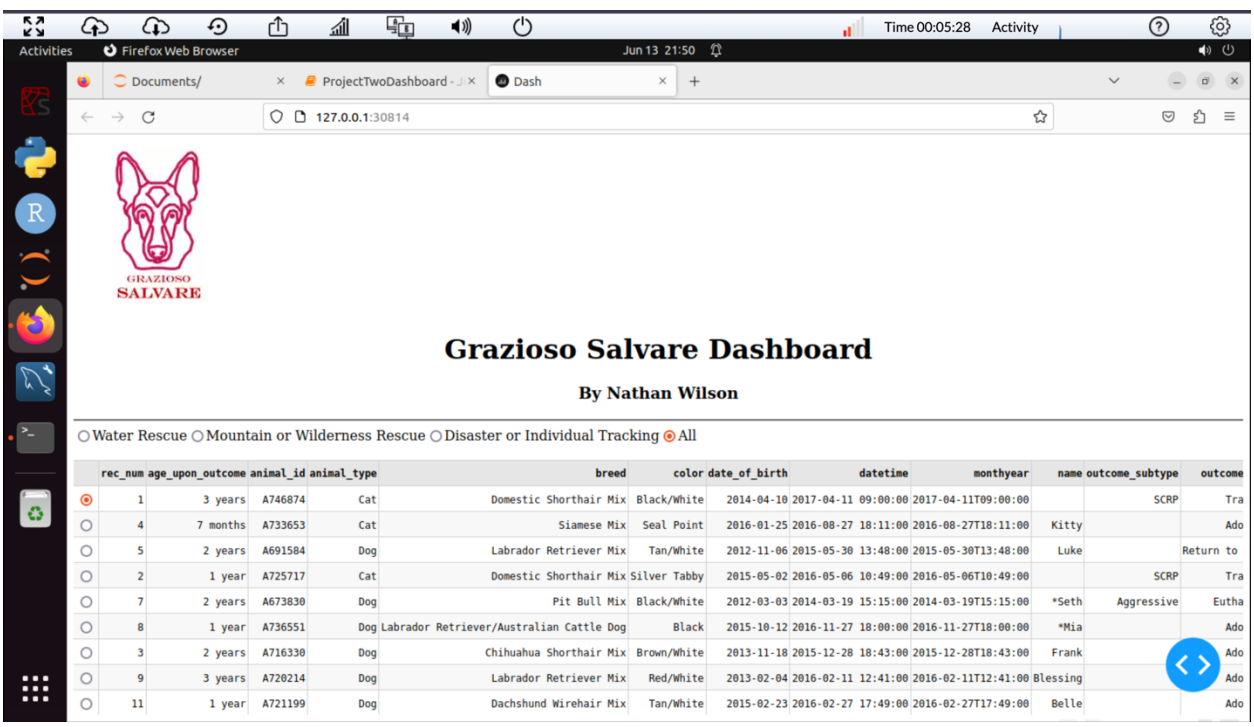


Note: This template has been adapted from the following sample templates: [Make a README](), [Best README Template](), and [A Beginners Guide to Writing a Kickass README]().

Unit Test results



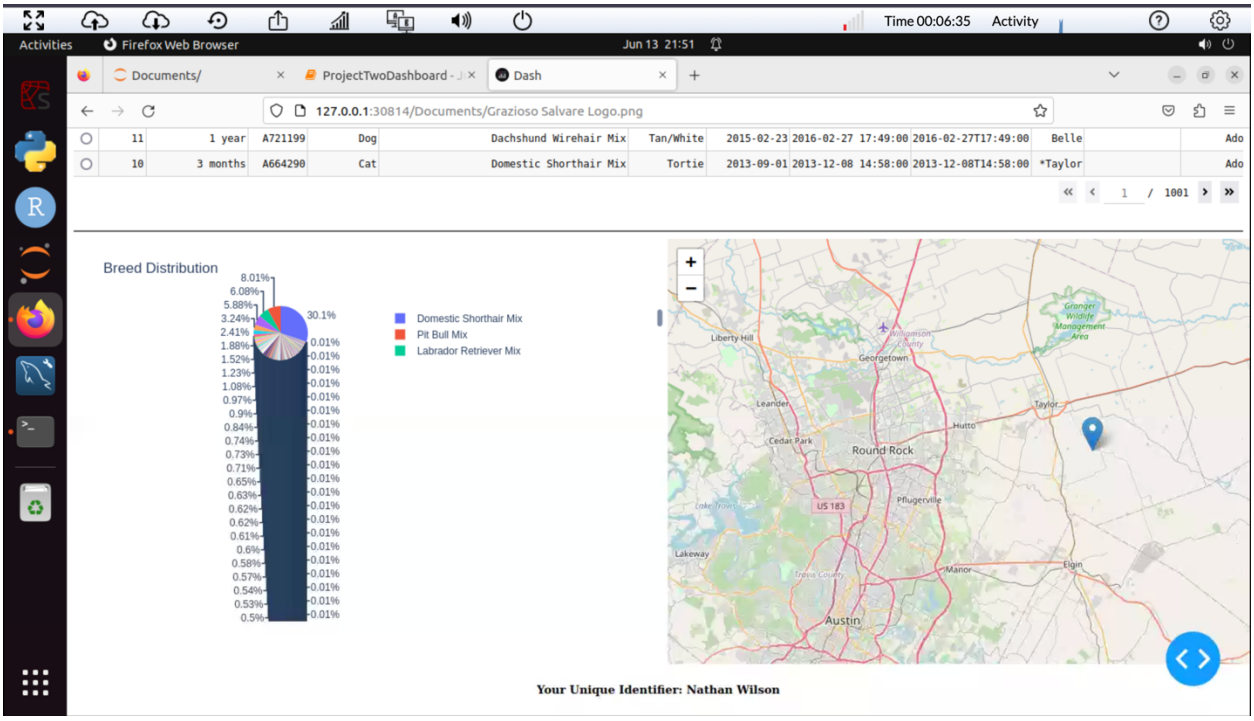
Dashboard unfiltered outcomes



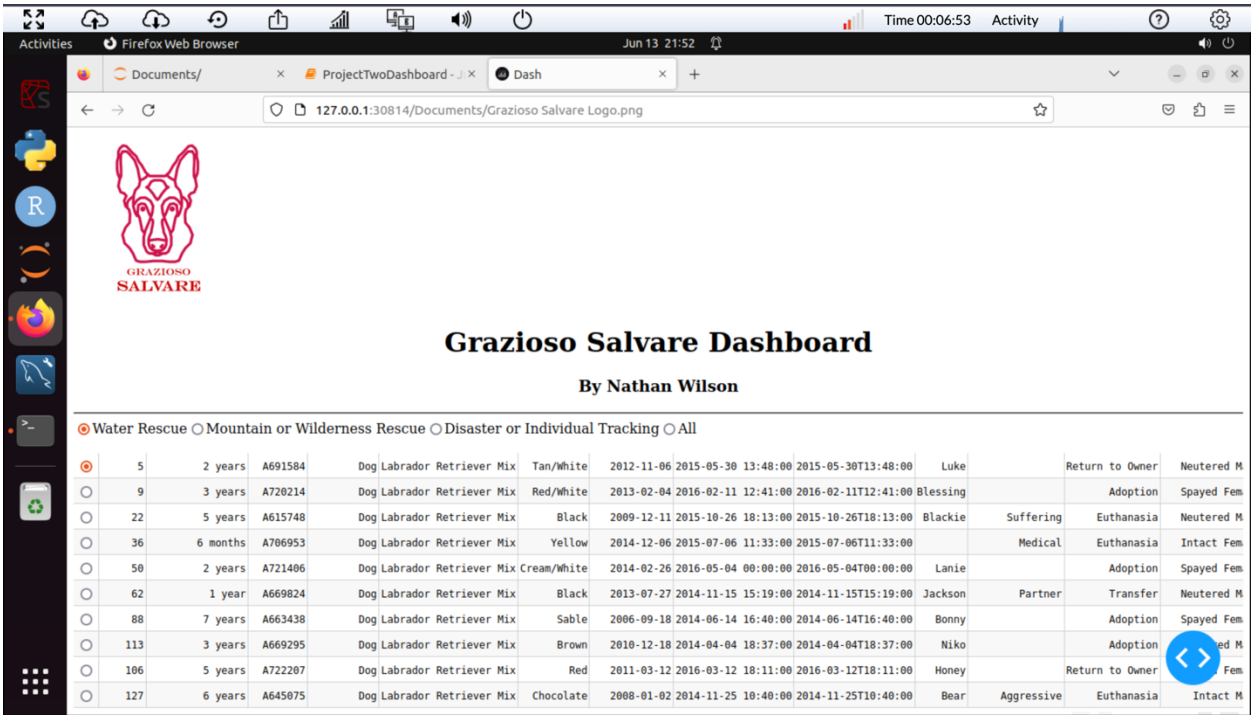
	rec_num	age_upon_outcome	animal_id	animal_type	breed	color	date_of_birth	datetime	monthyear	name	outcome_subtype	outcome
◉	1	3 years	A746874	Cat	Domestic Shorthair Mix	Black/White	2014-04-10	2017-04-11 09:00:00	2017-04-11T09:00:00		SCRP	Tra
○	4	7 months	A733653	Cat	Siamese Mix	Seal Point	2016-01-25	2016-08-27 18:11:00	2016-08-27T18:11:00	Kitty		Ado
○	5	2 years	A691584	Dog	Labrador Retriever Mix	Tan/White	2012-11-06	2015-05-30 13:48:00	2015-05-30T13:48:00	Luke		Return to
○	2	1 year	A725717	Cat	Domestic Shorthair Mix	Silver Tabby	2015-05-02	2016-05-06 10:49:00	2016-05-06T10:49:00		SCRP	Tra
○	7	2 years	A673830	Dog	Pit Bull Mix	Black/White	2012-03-03	2014-03-19 15:15:00	2014-03-19T15:15:00	*Seth	Aggressive	Eutha
○	8	1 year	A736551	Dog	Labrador Retriever/Australian Cattle Dog	Black	2015-10-12	2016-11-27 18:00:00	2016-11-27T18:00:00	*Mia		Ado
○	3	2 years	A716330	Dog	Chihuahua Shorthair Mix	Brown/White	2013-11-18	2015-12-28 18:43:00	2015-12-28T18:43:00	Frank		Ado
○	9	3 years	A720214	Dog	Labrador Retriever Mix	Red/White	2013-02-04	2016-02-11 12:41:00	2016-02-11T12:41:00	Blessing		Ado
○	11	1 year	A721199	Dog	Dachshund Wirehair Mix	Tan/White	2015-02-23	2016-02-27 17:49:00	2016-02-27T17:49:00	Belle		Ado

Note: This template has been adapted from the following sample templates: [Make a README](), [Best README Template](), and [A Beginners Guide to Writing a Kickass README]().

Dashboard Unfiltered Breed distribution

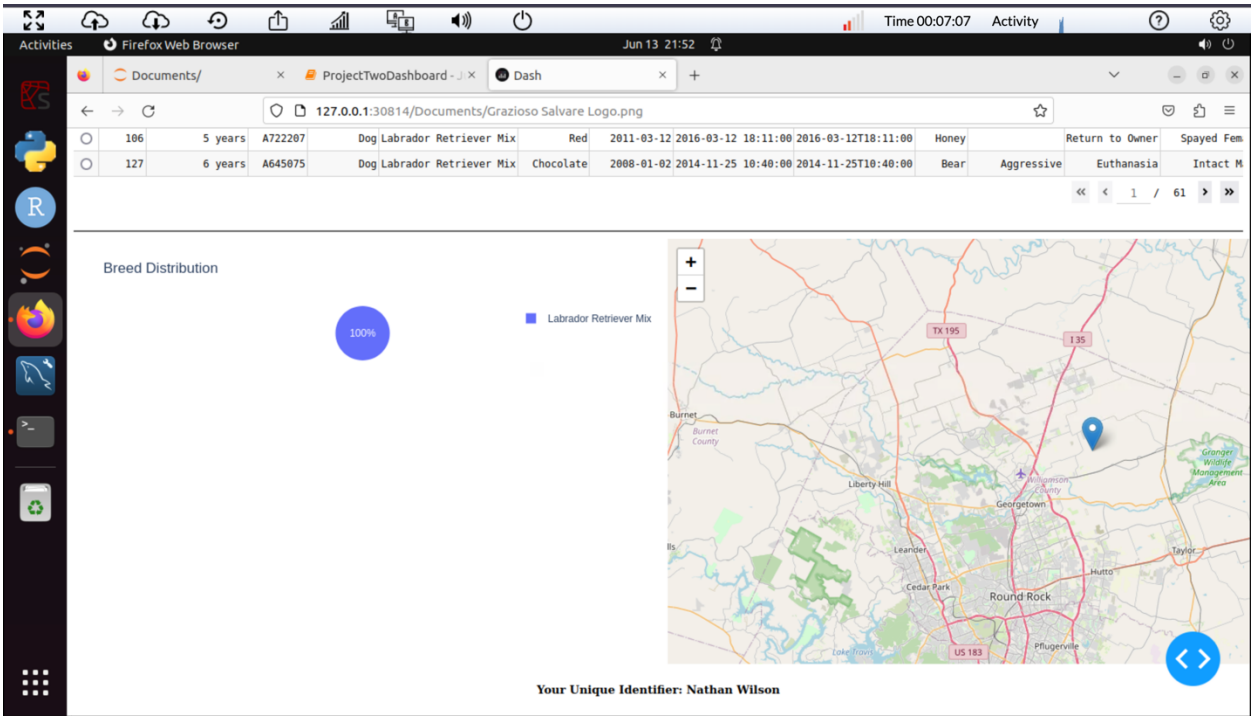


Dashboard – Water Rescue outcomes

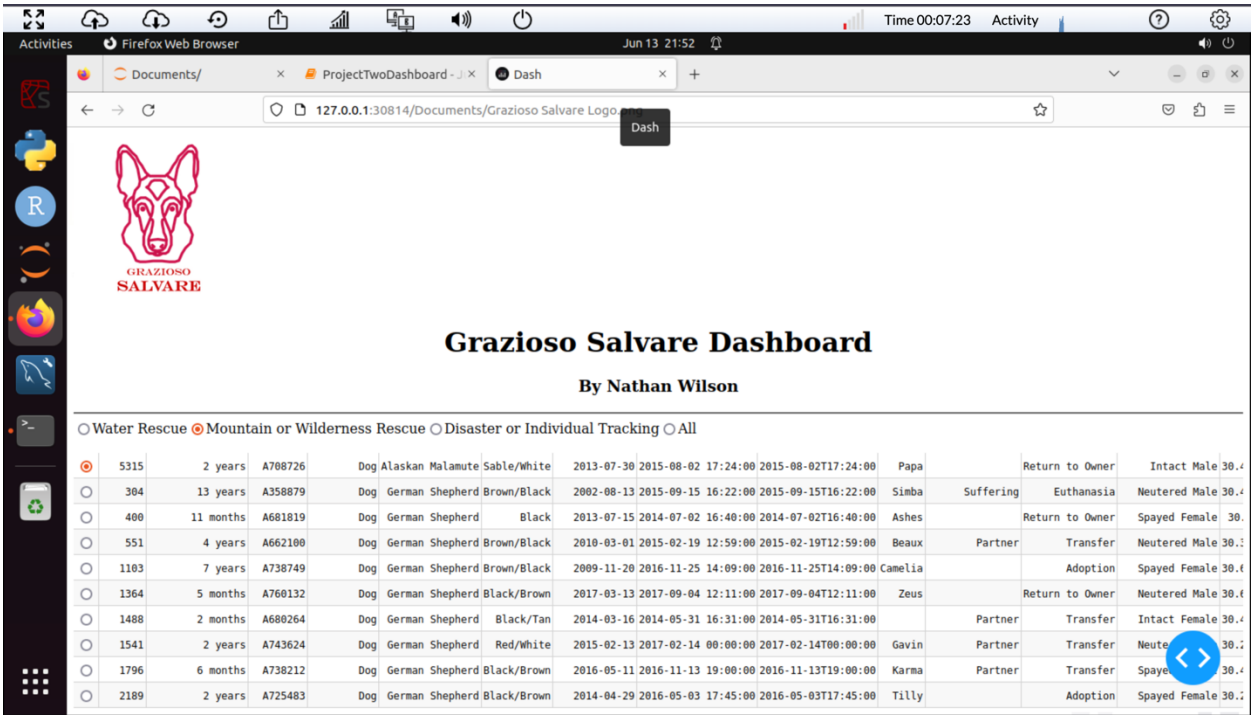


Note: This template has been adapted from the following sample templates: [Make a README](), [Best README Template](), and [A Beginners Guide to Writing a Kickass README]().

Water Rescue Breed distribution

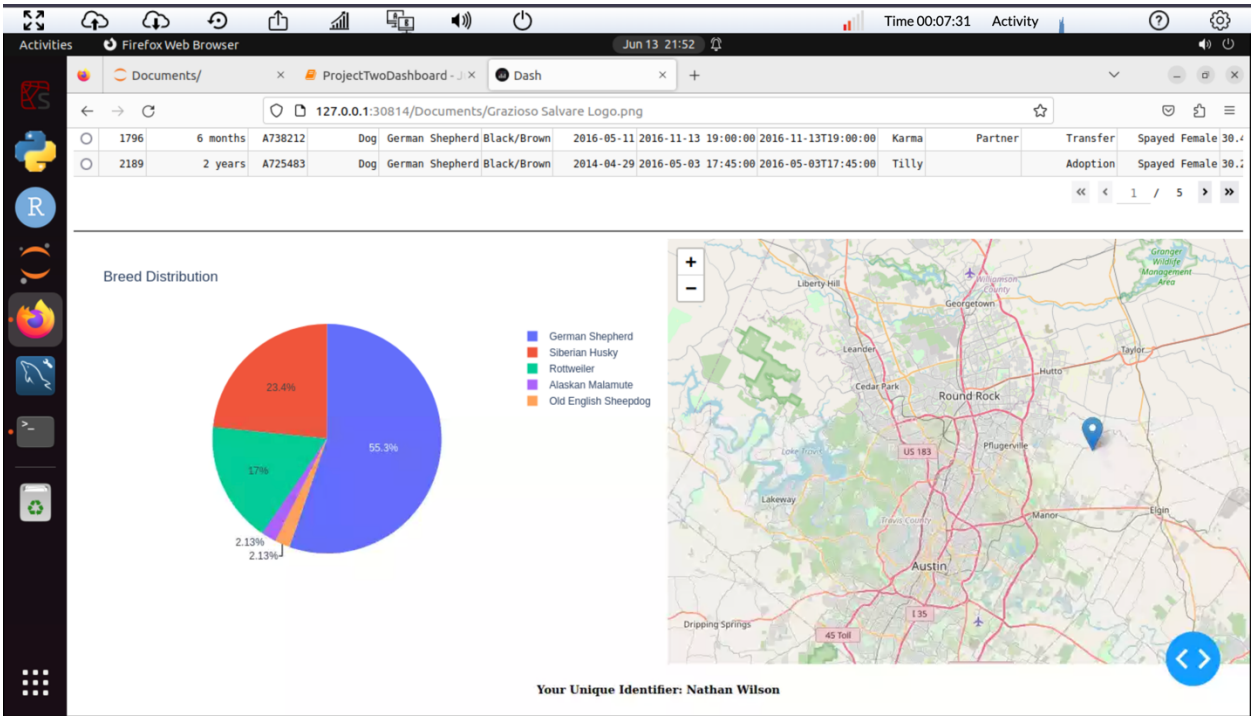


Dashboard – Mountain or Wilderness Rescue Outcomes

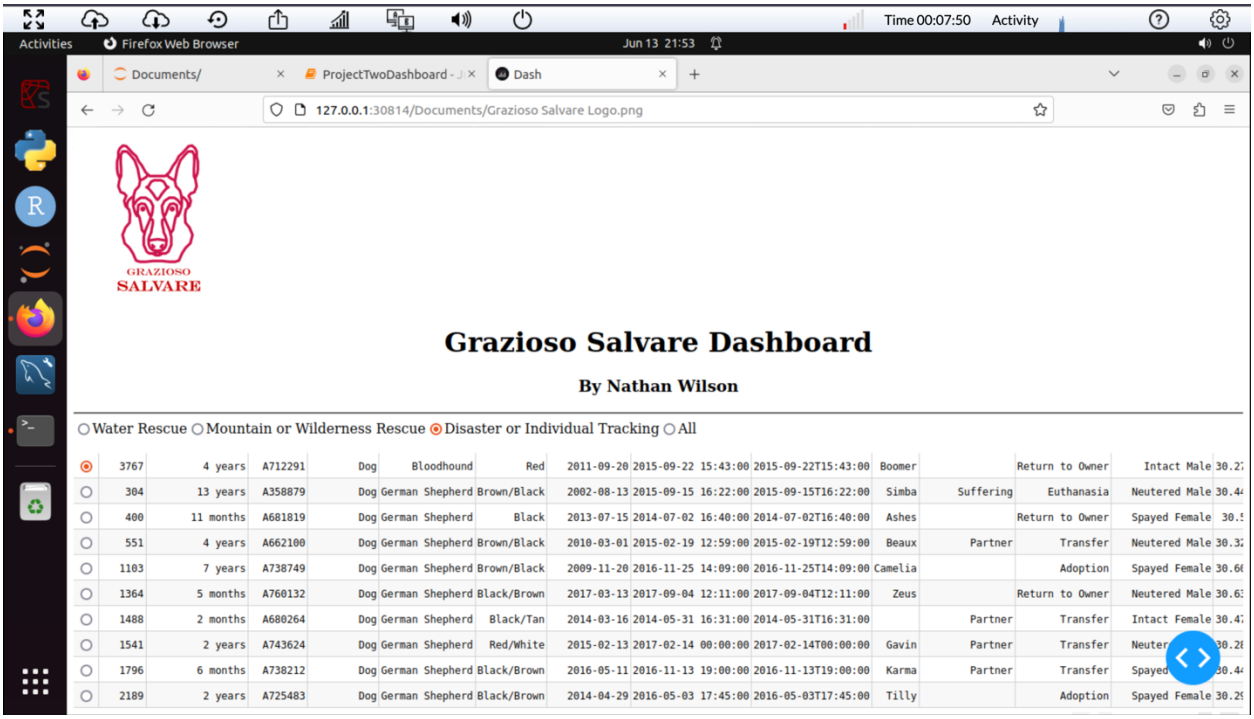


Note: This template has been adapted from the following sample templates: [Make a README](), [Best README Template](), and [A Beginners Guide to Writing a Kickass README]().

Mountain or Wilderness Rescue Breed distribution

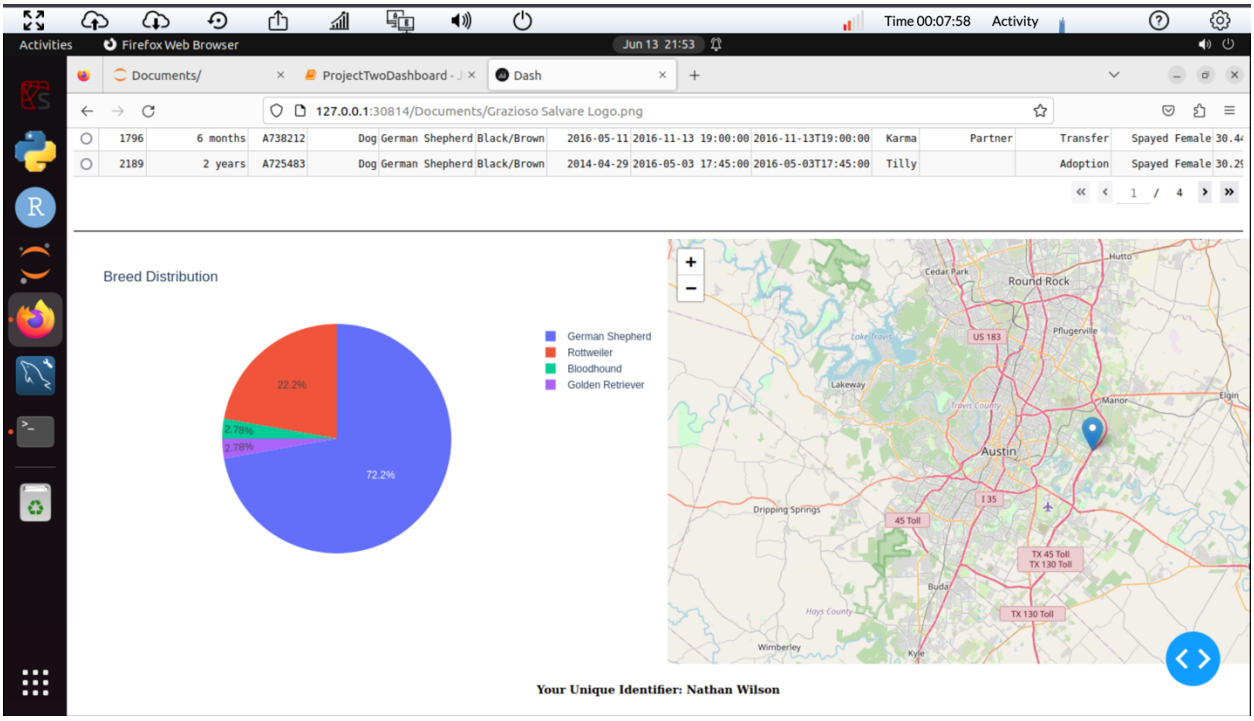


Disaster or Individual Tracking outcomes



Note: This template has been adapted from the following sample templates: [Make a README](), [Best README Template](), and [A Beginners Guide to Writing a Kickass README]().

Disaster or Individual Tracking breed distribution



Roadmap/Features (Optional)

Provide an open issues list of proposed features (and known issues). If you have ideas for releases in the future, it is a good idea to list them in the README. What makes your project stand out?

Note: This section is optional for the purposes of this assignment. If you choose not to fill out this section, remove it from your final README file.

Contact

Your name: Nathan Wilson

Note: This template has been adapted from the following sample templates: [Make a README](), [Best README Template](), and [A Beginners Guide to Writing a Kickass README]().