**CS 340 README**

**Required Functionality:**

The functionality requirements for this project were to create a dashboard, controllers, and database to track and store animals. The database tracks animals and information that can be parsed and searched through using a dashboard. The goal of the dashboard is to locate certain animals that meet rescue parameters. The quick search buttons on the top show all the respective animals that fit that category.

Initial View:

A screenshot of a computer

Description automatically generated

Water Rescue:

A screenshot of a computer

Description automatically generated

Mountain or Wilderness Rescue:

A screenshot of a computer

Description automatically generated

Disaster or Individual Tracking:  
  
A map on a white background

Description automatically generated

Reset:

A screenshot of a computer

Description automatically generated

**Tools:**

The tools used for this project were MongoDB as the database, dash as the front end dashboard, and leaflet and plotly python libraries to add charts and geomapping to the application. MongoDB was chosen as the database due to it being NoSQL which allows for easier queries against long documents and extensive collections. Dash was used for its simplistic and lightweight front-end capabilities. Dash allowed for a easy set up, but clean looking UI/UX that is intuitive and easy to alter.

<https://www.dash-leaflet.com/>

<https://plotly.com/python/plotly-express/>

The project started in Mongo shell where I created users and started manipulating the data and collections. After this, I began working on a python class that would act as the controller and give CRUD functionality to the user to use on the MongoDB animals database. The controller was usable on its own, but not user friendly. It was then the last step to create a user interface that was simple and clear. The dashboard that was developed had the important information a user would need and was not overly complicated.

There were some challenges I faced with getting callbacks to trigger the correct buttons and query the correct data, however after going back and adding log output data I was able to figure out where the misfires were occurring and fix the issue.