CS-340 7-2 Project Two Submission: Read Me

Cade Bray

Cade.Bray@snhu.edu

Southern New Hampshire University

Chao Ling, M.S

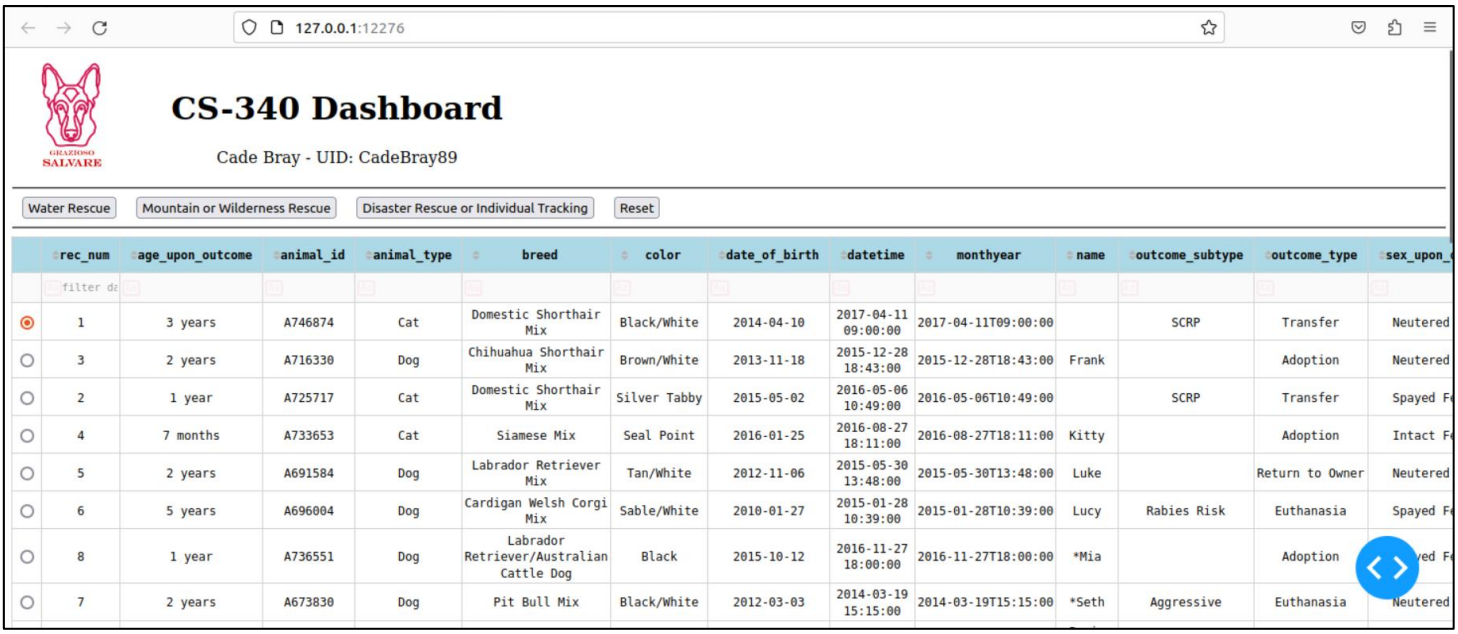
February 25th, 2025

During this project my client Grazio Salvare has requested documentation demonstrating the requested functionality, descriptions of the tools used to achieve functionality, and rationale for use of these tools. In addition, I will include an explanation of all the steps taken to complete the specifications in this project and a report as to the challenges I encountered during the implementation along with how I had overcome those challenges. Please refer to the CS-340 Dashboard Specifications Document provided by the client linked found in the reference material.

To start, I'll speak about the functionality of the project. The client requested their specific branding to be found on the dashboard. I've implemented this by adding the clients provided logo (Grazioso Salvare Logo) in the top left corner of the dashboard along with a header that shows "CS-340 Dashboard" as seen in figure 1 below. The client insisted that I include a credit for myself as the developer in two locations found below the header as "Cade Bray – UID: CadeBray89" and above the charts section of the dashboard seen in figures one and FIXME. The rationale for these locations conforms with industry competition conventions and with the absence of specific directives I've chosen to conform to those conventions. As requested by the client I've also made the image linkable to the https://www.snhu.edu homepage.

Grazioso Salvare required interactive filtering options for the Austin Animal Center Outcomes data set. The client did not specify how these filtering options should be implemented but did give examples such as buttons, radio buttons, or a drop-down menu. I've chosen to implement simple buttons for the filtering ability. The rationale behind this design choice was to create a sleek and user-friendly design that was minimal in nature compared to a more cluttered approach such as radio buttons. The data sets the client has requested to be predefined filtering buttons to include 'Water Rescue', 'Mountain or Wilderness Rescue', 'Disaster Rescue or

Individual Tracking', and a filter 'Reset' button. The specifications for these predefined queries will be explored later in this document. The implemented buttons can be seen as described in figure one.



	rec_num	age_upon_outcome	animal_id	animal_type	breed	color	date_of_birth	datetime	monthyear	name	outcome_subtype	outcome_type	sex_upon_
	filter da												
◉	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	Transfer	Neutered
○	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		Adoption	Neutered
○	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	Transfer	Spayed F
○	4	7 months	A733653	Cat	Siamese Mix	Seal Point	2016-01-25	2016-08-27 18:11:00	2016-08-27T18:11:00	Kitty		Adoption	Intact F
○	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 Owner	Neutered
○	6	5 years	A696004	Dog	Cardigan Welsh Corgi Mix	Sable/White	2010-01-27	2015-01-28 10:39:00	2015-01-28T10:39:00	Lucy	Rabies Risk	Euthanasia	Spayed F
○	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		Adoption	ved F
○	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	Euthanasia	Neutered

Figure 1, Dashboard untouched by filtering options

The client also required the data table displayed to dynamically respond to filtering options. To implement this requirement two trains of thought were formed. The dashboard needed to respond to filter requests without reloading the page and could take in user designed filter criteria. To update the dashboard without reloading the page I've used a site framework that meets that requirement from the company plotly. This framework is called dash, and its documentation guide is linked in this document's references (Plotly, n.d.). Secondly, the user can provide their own unique queries by using the 'filter data' row located at the top of dashboard data set.

Grazioso Salvare has requested a geolocation and a second chart visualization of my choice to be added below the data set. For the convenience of the user, I've enabled pagination,

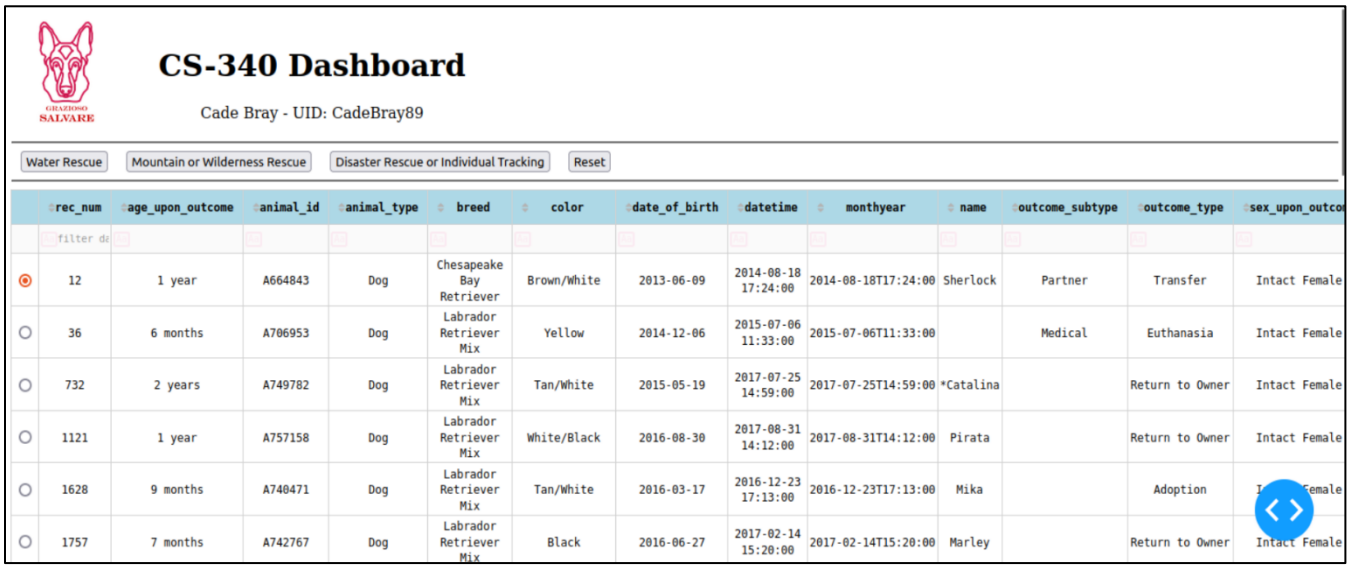
so the user doesn't need to scroll through the entire data set to view the geolocation and secondary chart. I've set the page limit to 20 entries so the user could see considerable data with limited scrolling. The second chart for visualization was implemented as a pie chart. The rationale behind choosing a pie chart was that the purpose for this visualization was to see the percentage of available breeds in the database that met the query requirements. These charts can be seen in figures seven through eleven below.

To fulfill the requirement of interactive filtering options described above that were implemented as buttons I've created a mongo database middleware built on top of the PyMongo driver. This middleware simplifies the complex driver to this project's core functional requirements of providing an inclusive list of entries based on a query. For future additions and to conform with best practices the create, update, and delete methods of the CRUD API structure were created in this middleware as well. The requirements for the specific predefined queries that are associated with the buttons can be seen in figure two below.

Rescue Type	Preferred Breeds	Preferred Sex	Training Age*
Water	Labrador Retriever Mix, Chesapeake Bay Retriever, Newfoundland	Intact Female	26 weeks to 156 weeks
Mountain or Wilderness	German Shepherd, Alaskan Malamute, Old English Sheepdog, Siberian Husky, Rottweiler	Intact Male	26 weeks to 156 weeks
Disaster or Individual Tracking	Doberman Pinscher, German Shepherd, Golden Retriever, Bloodhound, Rottweiler	Intact Male	20 weeks to 300 weeks

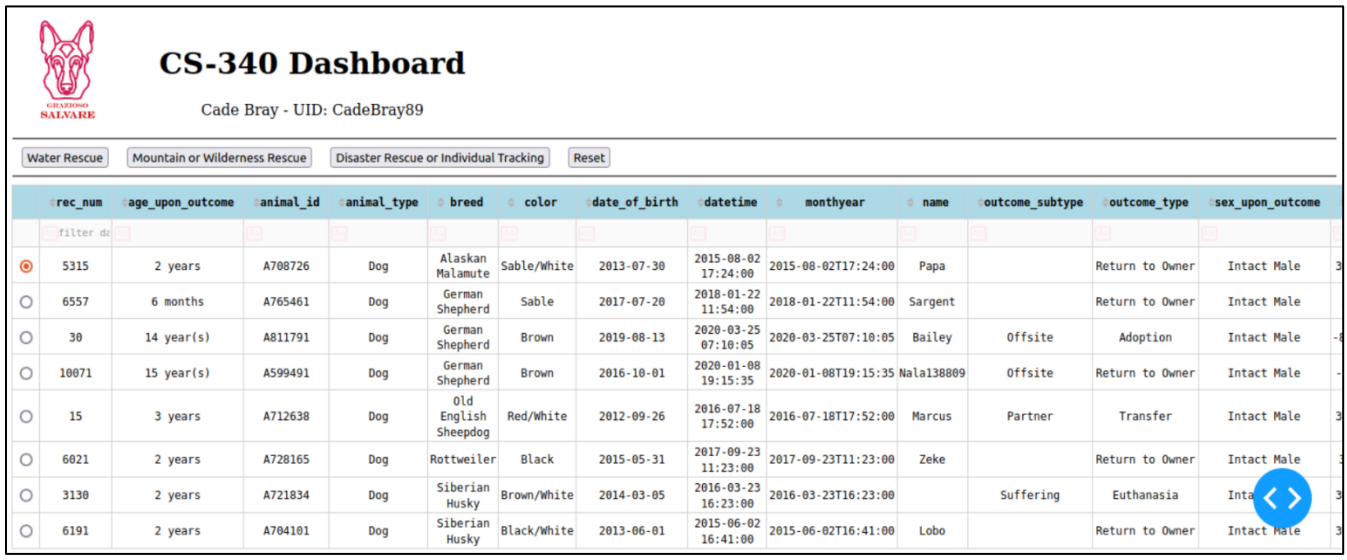
Figure 2, Query requirements table provided by client in specification document

These queries were built and implemented as custom MongoDB queries that update the data set object upon the buttons click as seen in figures three through six.



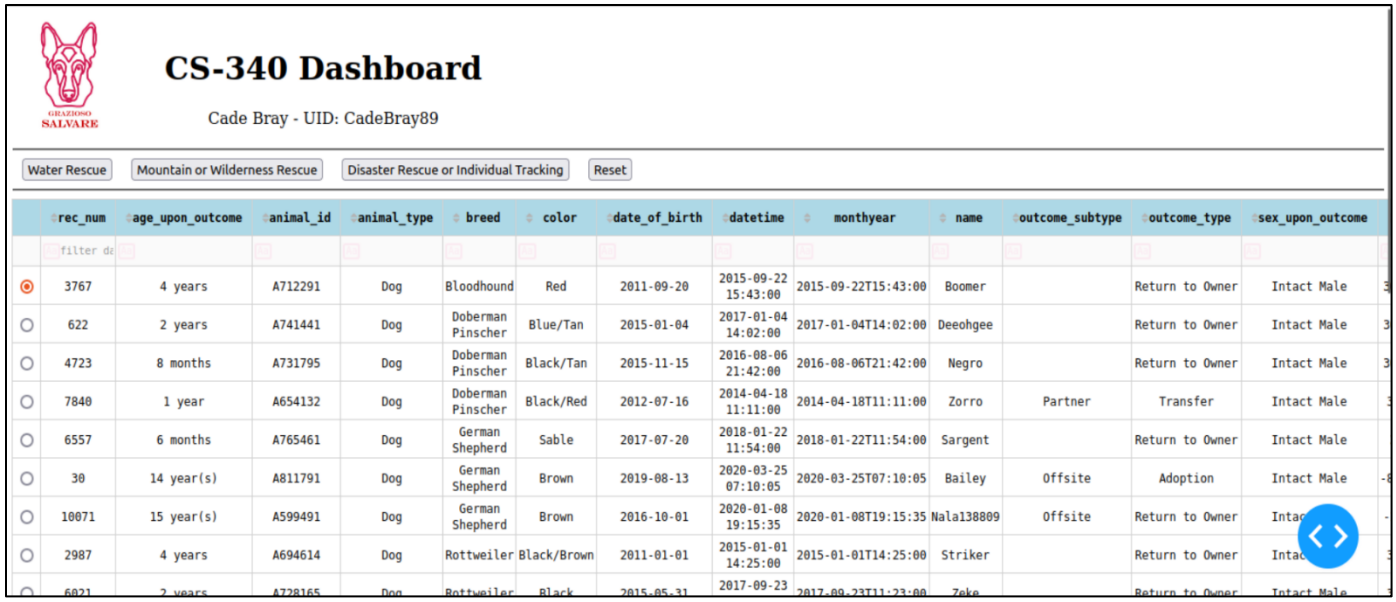
rec_num	age_upon_outcome	animal_id	animal_type	breed	color	date_of_birth	datetime	monthyear	name	outcome_subtype	outcome_type	sex_upon_outcome
12	1 year	A664843	Dog	Chesapeake Bay Retriever	Brown/White	2013-06-09	2014-08-18 17:24:00	2014-08-18T17:24:00	Sherlock	Partner	Transfer	Intact Female
36	6 months	A706953	Dog	Labrador Retriever Mix	Yellow	2014-12-06	2015-07-06 11:33:00	2015-07-06T11:33:00		Medical	Euthanasia	Intact Female
732	2 years	A749782	Dog	Labrador Retriever Mix	Tan/White	2015-05-19	2017-07-25 14:59:00	2017-07-25T14:59:00	*Catalina		Return to Owner	Intact Female
1121	1 year	A757158	Dog	Labrador Retriever Mix	White/Black	2016-08-30	2017-08-31 14:12:00	2017-08-31T14:12:00	Pirata		Return to Owner	Intact Female
1628	9 months	A740471	Dog	Labrador Retriever Mix	Tan/White	2016-03-17	2016-12-23 17:13:00	2016-12-23T17:13:00	Mika		Adoption	
1757	7 months	A742767	Dog	Labrador Retriever Mix	Black	2016-06-27	2017-02-14 15:20:00	2017-02-14T15:20:00	Marley		Return to Owner	Intact Female

Figure 3, Dashboard filtered by Water Rescue



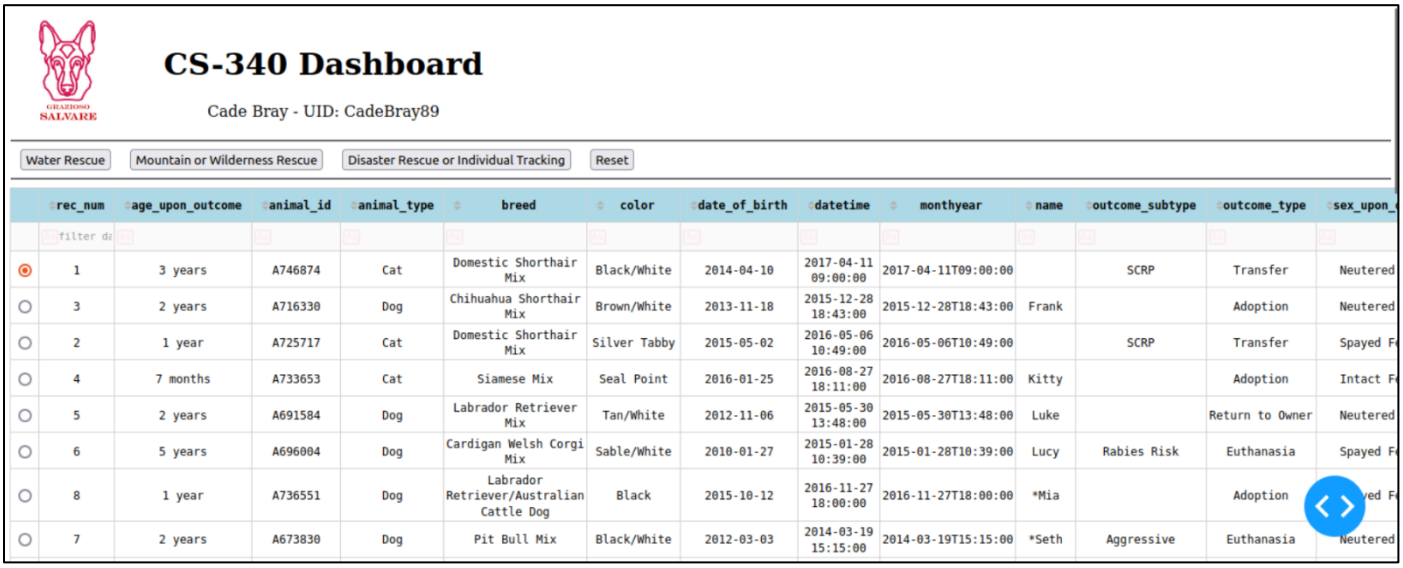
rec_num	age_upon_outcome	animal_id	animal_type	breed	color	date_of_birth	datetime	monthyear	name	outcome_subtype	outcome_type	sex_upon_outcome
5315	2 years	A708726	Dog	Alaskan Malamute	Sable/White	2013-07-30	2015-08-02 17:24:00	2015-08-02T17:24:00	Papa		Return to Owner	Intact Male
6557	6 months	A765461	Dog	German Shepherd	Sable	2017-07-20	2018-01-22 11:54:00	2018-01-22T11:54:00	Sargent		Return to Owner	Intact Male
30	14 year(s)	A811791	Dog	German Shepherd	Brown	2019-08-13	2020-03-25 07:10:05	2020-03-25T07:10:05	Bailey	Offsite	Adoption	Intact Male
10071	15 year(s)	A599491	Dog	German Shepherd	Brown	2016-10-01	2020-01-08 19:15:35	2020-01-08T19:15:35	Nala138809	Offsite	Return to Owner	Intact Male
15	3 years	A712638	Dog	Old English Sheepdog	Red/White	2012-09-26	2016-07-18 17:52:00	2016-07-18T17:52:00	Marcus	Partner	Transfer	Intact Male
6021	2 years	A728165	Dog	Rottweiler	Black	2015-05-31	2017-09-23 11:23:00	2017-09-23T11:23:00	Zeke		Return to Owner	Intact Male
3130	2 years	A721834	Dog	Siberian Husky	Brown/White	2014-03-05	2016-03-23 16:23:00	2016-03-23T16:23:00		Suffering	Euthanasia	
6191	2 years	A704101	Dog	Siberian Husky	Black/White	2013-06-01	2015-06-02 16:41:00	2015-06-02T16:41:00	Lobo		Return to Owner	Intact Male

Figure 4, Dashboard filtered by Mountain or Wilderness Rescue



	rec_num	age_upon_outcome	animal_id	animal_type	breed	color	date_of_birth	datetime	monthyear	name	outcome_subtype	outcome_type	sex_upon_outcome
	filter da												
◉	3767	4 years	A712291	Dog	Bloodhound	Red	2011-09-20	2015-09-22 15:43:00	2015-09-22T15:43:00	Boomer		Return to Owner	Intact Male
○	622	2 years	A741441	Dog	Doberman Pinscher	Blue/Tan	2015-01-04	2017-01-04 14:02:00	2017-01-04T14:02:00	Deeohgee		Return to Owner	Intact Male
○	4723	8 months	A731795	Dog	Doberman Pinscher	Black/Tan	2015-11-15	2016-08-06 21:42:00	2016-08-06T21:42:00	Negro		Return to Owner	Intact Male
○	7840	1 year	A654132	Dog	Doberman Pinscher	Black/Red	2012-07-16	2014-04-18 11:11:00	2014-04-18T11:11:00	Zorro	Partner	Transfer	Intact Male
○	6557	6 months	A765461	Dog	German Shepherd	Sable	2017-07-20	2018-01-22 11:54:00	2018-01-22T11:54:00	Sargent		Return to Owner	Intact Male
○	30	14 year(s)	A811791	Dog	German Shepherd	Brown	2019-08-13	2020-03-25 07:10:05	2020-03-25T07:10:05	Bailey	Offsite	Adoption	Intact Male
○	10071	15 year(s)	A599491	Dog	German Shepherd	Brown	2016-10-01	2020-01-08 19:15:35	2020-01-08T19:15:35	Nala138809	Offsite	Return to Owner	Intac
○	2987	4 years	A694614	Dog	Rottweiler	Black/Brown	2011-01-01	2015-01-01 14:25:00	2015-01-01T14:25:00	Striker		Return to Owner	Intac
○	6021	2 years	A728165	Dog	Rottweiler	Black	2015-05-31	2017-09-23	2017-09-23T11:23:00	Zeke		Return to Owner	Intact Male

Figure 6, Dashboard filtered by Disaster Rescue or Individual Tracking



	rec_num	age_upon_outcome	animal_id	animal_type	breed	color	date_of_birth	datetime	monthyear	name	outcome_subtype	outcome_type	sex_upon_
	filter da												
◉	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	Transfer	Neutered
○	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		Adoption	Neutered
○	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	Transfer	Spayed F
○	4	7 months	A733653	Cat	Siamese Mix	Seal Point	2016-01-25	2016-08-27 18:11:00	2016-08-27T18:11:00	Kitty		Adoption	Intact F
○	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 Owner	Neutered
○	6	5 years	A696004	Dog	Cardigan Welsh Corgi Mix	Sable/White	2010-01-27	2015-01-28 10:39:00	2015-01-28T10:39:00	Lucy	Rabies Risk	Euthanasia	Spayed F
○	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		Adoption	ved F
○	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	Euthanasia	Neutered

Figure 5, Dashboard after clicking the Reset button to revert to an unfiltered state

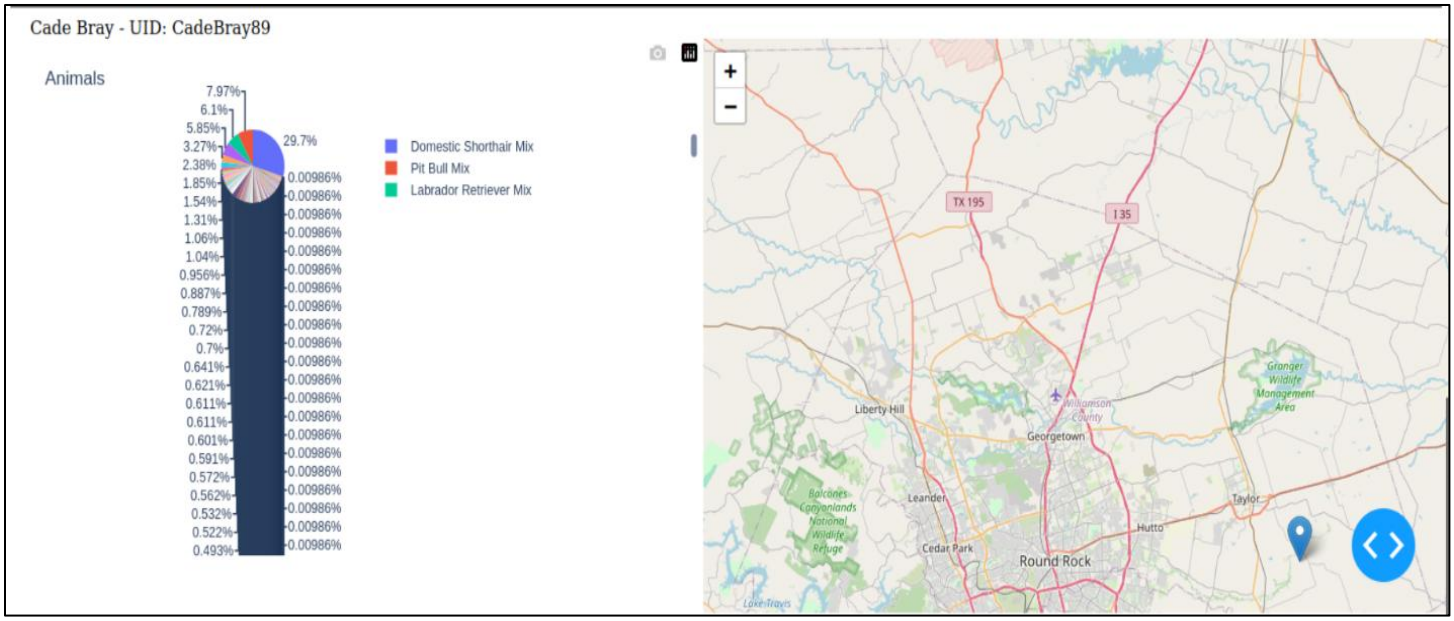


Figure 7, Untouched geolocation and pie chart

The pie chart has many values associated with it and as such is scaled but becomes more user friendly and operable as queries are defined.

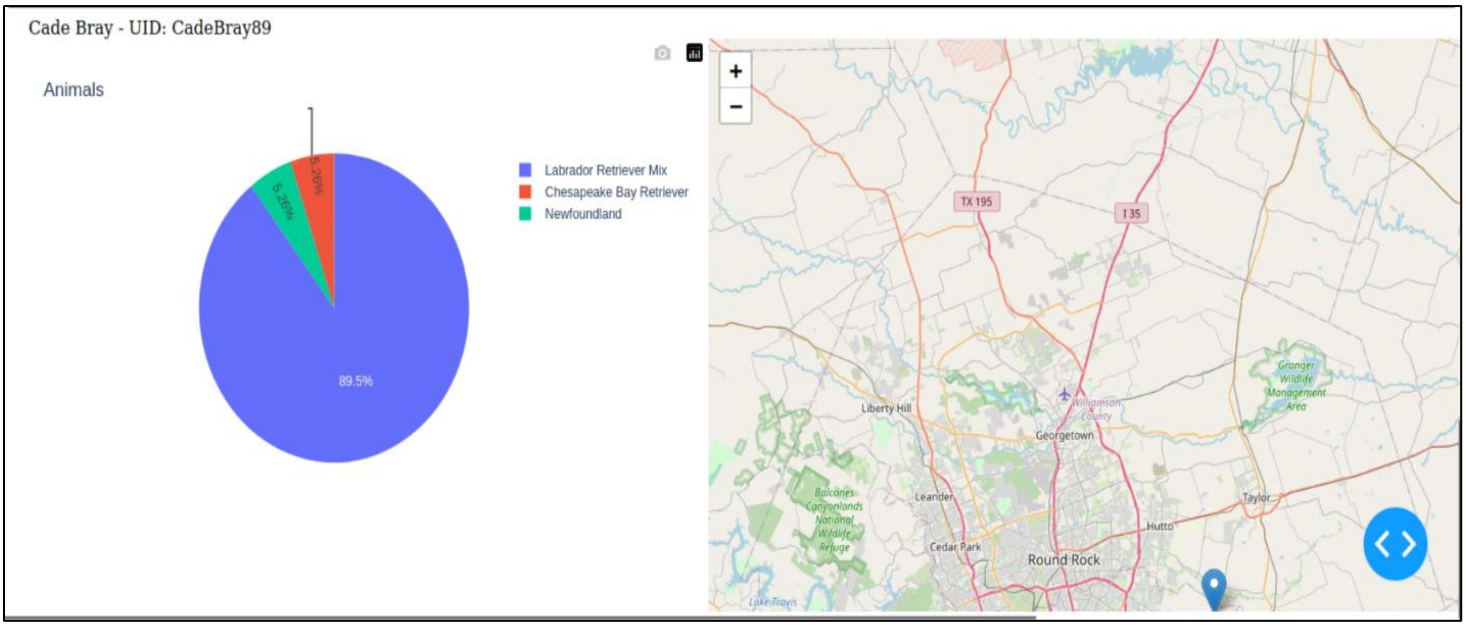


Figure 8, Geolocation and pie chart filtered by water rescue query

Figure 9, Geolocation and pie chart filtered by Mountain or Wilderness Rescue query

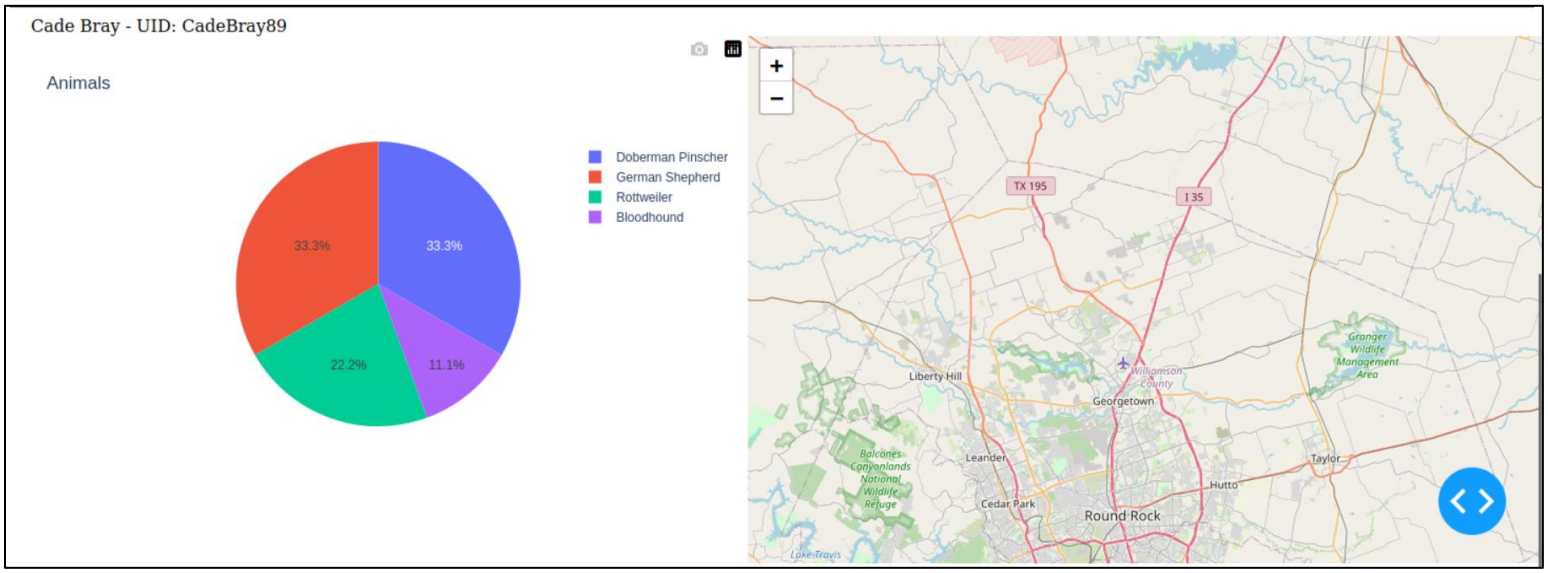


Figure 10, Geolocation and pie chart filtered by Disaster or Individual Tracking query

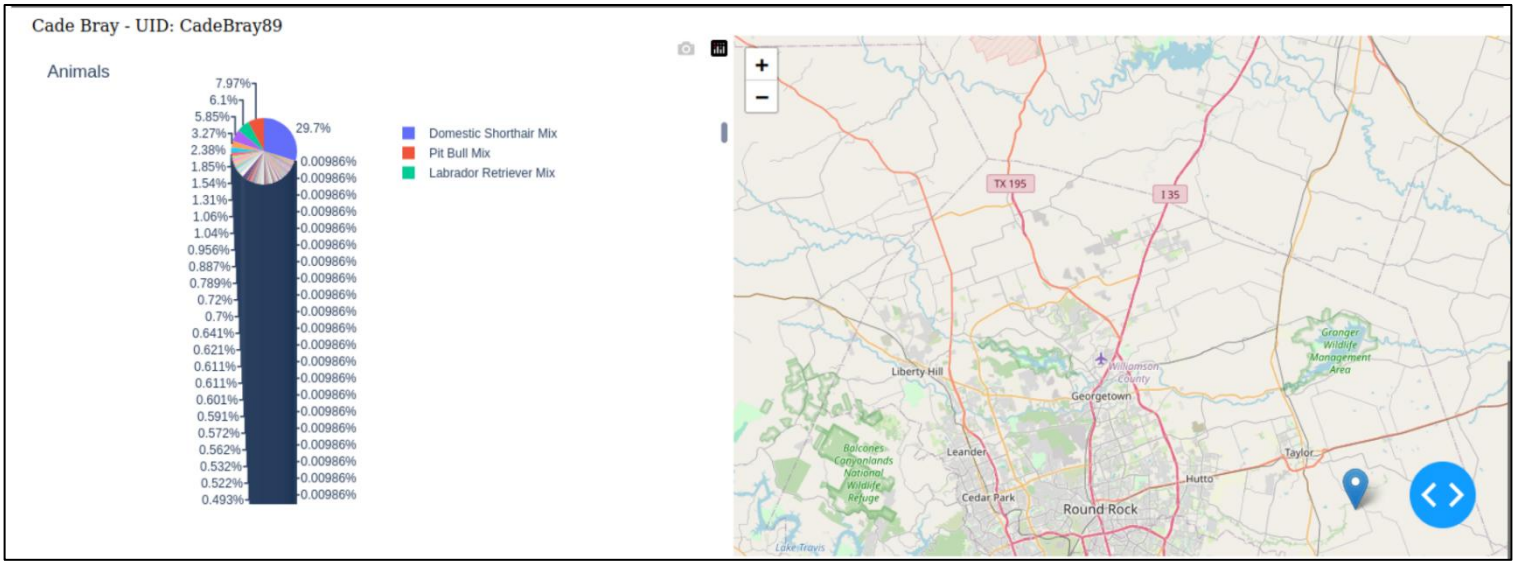


Figure 11, Geolocation and pie chart filtered by Reset query

The steps taken to complete this project can be summarized as follows. First, I studied the PyMongo driver documentation before implementing CRUD API middleware. This middleware was documented and tested using industry standard best practices. This document and Jupyter notebook can be found in my previous updates to the client as ‘test_book.ipynb’ and the associated read me file. Secondly, I studied the dash framework provided plotly before creating a Jupyter notebook that contains the dashboard implementation. This implementation contains industry best practices including ample documentation for other developers to understand the rationale behind its work.

Challenges that I faced during the dashboard implementation included two large hurdles. The first challenge was the site styling which involved studying the dash documentation to understand how html styling was performed through their dash framework. Doing so allowed me to adjust the size of branding elements and move elements in column and inline positions. Additionally, I was able to create a modern design with buttons that involved creating padding to

become user friendly. The second challenge I faced was in handling the button functionality. I started by using the ‘filter-type’ input for the app callback and assigned that parameter id to each button with an associated value that would be returned. Little did I know that the dash framework does not support multiple elements with the same object ID. This posed an issue when debugging because it did not produce a dash site error nor an IDE linter error. I experimented with implementations until I found that assigning each button its own ID allowed the buttons to call the app call backs function.

In addition to this challenge, I was unable to pass a value associated with the buttons and the only value I was receiving was ‘n_clicks’ which was a simple integer of how many times the user hand clicked the button. I explored the documentation once again to find that I could instead pass ‘n_clicks_timestamp’. This allowed me to compare timestamps to find the most recently clicked button. I proceeded to implement this logic into the dashboard but faced another issue. If the button had never been clicked before it would return a call back error because a None type couldn’t be evaluated against the other timestamps. To avoid this issue the dictionary object that was initialized on the app call back asserted a default zero integer to compare against the other timestamps.

To release this project into production the client needs only open the provided Jupyter notebook, change final line ‘app.run_server(debug=True)’ to contain ‘debug=False’ and run the notebook. In the output of the Jupyter notebook the client should see the middleware initialized by pinging the database and returning a successful statement and the local IP address that the site is hosted on. When a button is clicked on the site by a user the notebook output will contain an updating dashboard statement to assist in debugging if needed.

References

Plotly. (n.d.). *Dash Python User Guide*. Retrieved from Plotly: https://dash.plotly.com/

Southern New Hampshire University. (n.d.). *CS-340 Dashboard Specifications Document*. Retrieved from Southern New Hampshire University: https://learn.snhu.edu/content/enforced/1831818-CS-340-13191.202511-1/course_documents/CS%20340%20Dashboard%20Specifications%20Document.pdf?_&d2lSessionVal=ScQoRYLN9OTGv4T9RbCyYwie6&ou=1332057&ou=1831818

Southern New Hampshire University. (n.d.). Grazioso Salvare Logo. Retrieved from https://learn.snhu.edu/content/enforced/1831818-CS-340-13191.202511-1/course_documents/Grazioso%20Salvare%20Logo.png?_&d2lSessionVal=ScQoRYLN9OTGv4T9RbCyYwie6&ou=1332057&ou=1831818