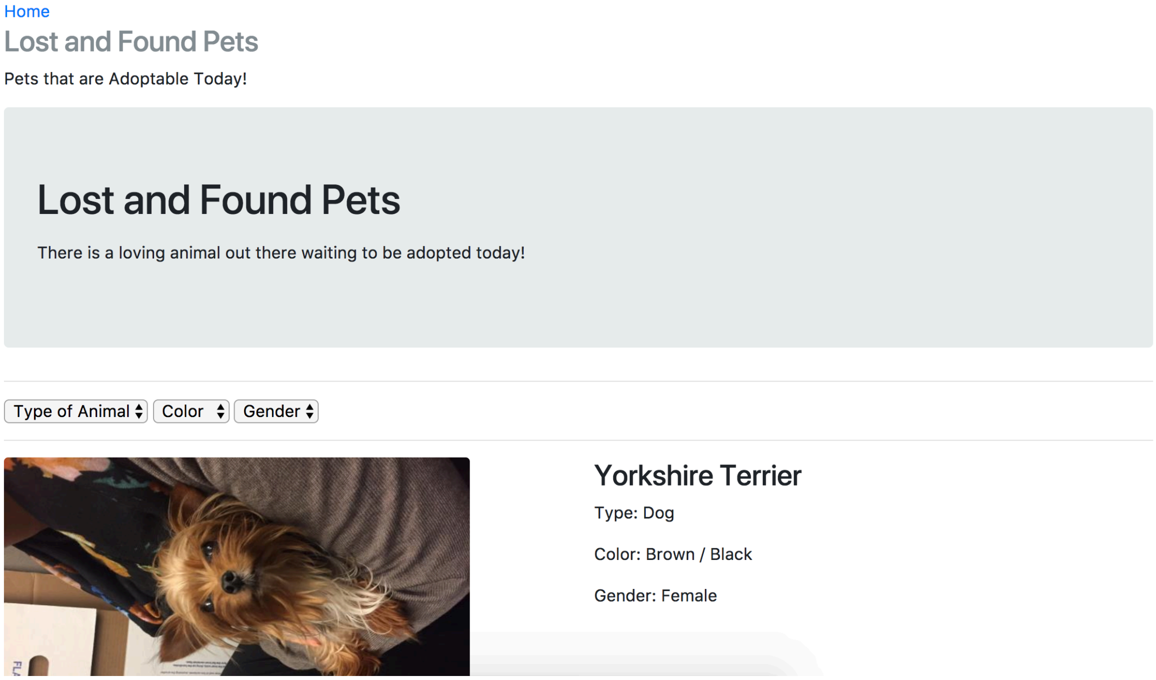
Abigail Graese, David Horne, Luna Amabaye
CS 4420 Project Status Report

Files: *Can be found at https://github.com/agraese/CS_4420 with instructions on how to run the app.*

Progress Description:



Home

Lost and Found Pets

Pets that are Adoptable Today!

Lost and Found Pets

There is a loving animal out there waiting to be adopted today!

Yorkshire Terrier

Type: Dog

Color: Brown / Black

Gender: Female

The above picture is a demonstration of the progress we have made with the web application. It runs on your local host using flask. We have also gathered and sanitized the data that we are using in the database, Lost, Found, and Adoptable pets from King County Washington. Our database has been created, utilizing sqlAlchemy to help us connect the front end to the back end.

For the Final Report:

We will finish connecting the database to the web application and ensure that queries can be run against the database using user input from the dropdowns that have all the options for finding pets. In addition to this, we need to implement the rest of the tables and ensure the relationships are set up.

We will also include a section about all that we learned in setting up the database and web application, as the we had not worked with the tools we chose to work with before this project.

SQLAlchemy

For the backend database we used SqlAlchemy. It is python sql toolkit extension that allows you to make a separate database for flask and integrate it. It is Sql based database, this will be able to make relations and it will be easier for CRUD.

The database is stored in SQLAlchemy directory rather than our app directory. When integrating sqlalchamy we ran into some issues. The first issue was it wasn't clear where exactly the database was going to be placed. We were not sure if the database had to be in the file where the app was or in SQLalchemy file. It had to be in SQLAlchemy folder.

Once we made the test database in SQLAlchemy database it was hard to configure our app.py for flask-SQLAlchemy.
The various problems arose partly because it was difficult to configure it specifically to our app.py code we had. Once that was done, the program into problems which were fixed by installing flask-SQLAlchemy, because flask-SQLAlchemy does not automatically come with flask as we originally thought.

What we learned about SQLAlchemy:
SQLAlchemy gives your flask application the full functionality of a sql database and makes it easy to work with databases in general. flask-SQLAlchemy is a lot more flexible, and adaptable than other SQL databases because it sees the database as a relational algebra engine, not just a collection of tables. On the down side, if a person is not already familiar with it, it could be a huge learning curve. Configuring and making the actual data is a lot different than a normal SQL database. But it makes up for that by being very flexible and easy once it works.