App for Animal Adoption

Final Report

CS 467 - Capstone Project - Summer 2020

Kevin Graebel, Maria Taborda, Robert Pino

Introduction

Over the course of this semester, our group developed a web application that was intended to design a user experience for animal adoption web-applications, both from the point of view of the guest (potential animal adopter) and administrator (shelter employee). It can be found hosted at https://tgp-pet-cs467.uc.r.appspot.com.

There has been a marked increase in the number of animal adoptions worldwide as people desire a new partner in their life to help them through months of isolated quarantine. However, there are still a number of animal shelters that, despite a greater number of people seeking to adopt animals, are not able to take advantage of this situation given their lack of a web portal for potential adopters to engage with. We entered this project with an existing passion for animal adoption, as all members of our team have pets who were adopted, and one used to work at an animal shelter. Our goal was to build something that would be useful if deployed in a real world application to solve this issue.

In addition to building something that would have a purpose, we aimed to refine our skills with the various development tools we chose for this project. Based on previous work in OSU courses, the team had experience with development in HTML, JavaScript, Node.js, and Bootstrap, so we used this as the foundation of the web application. We brought in other useful additions via API, such as Google OAuth for user management, MailJet for email integration, Google Cloud Datastore for NoSQL database management, that provided a more robust set of functionality.

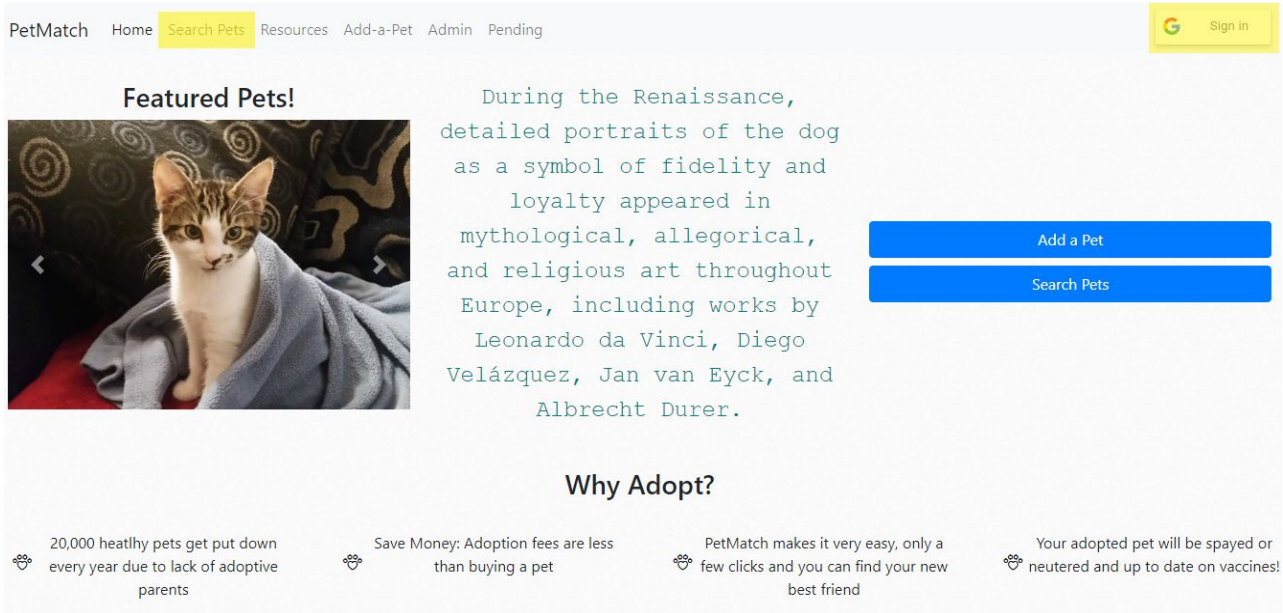
Setup and Usage

Our pet adoption app can be accessed at https://tgp-pet-cs467.uc.r.appspot.com/**.** Our app's use can be broken into two cases: that of a guest (potential animal adopter), and that of an administrator.

Guest Use Case

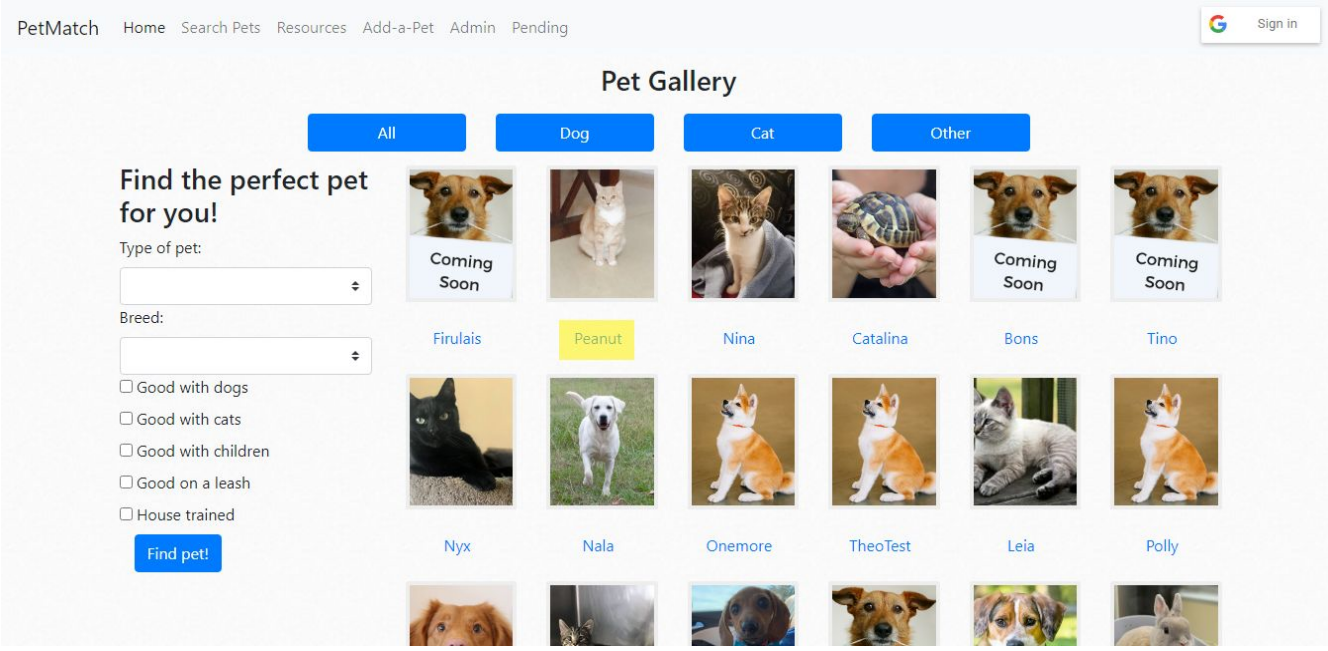
Home Page

1. The guest's user experience starts at the landing page**,** "PetMatch Home". This page features a rotating carousel of pictures of recently added pets that will take you directly to the animal adoption page if you wish.
2. Click on the "Sign In" button at the top, and use your desired credentials. This will not make you an administrator, but the site now will have access to your username and email address necessary to interact with certain pages.
3. The "Add a Pet" button is intended only for site administrators. As a "guest", you will be able to fill out this form, but you will receive a warning when you submit that you must be an administrator and the registration will fail.



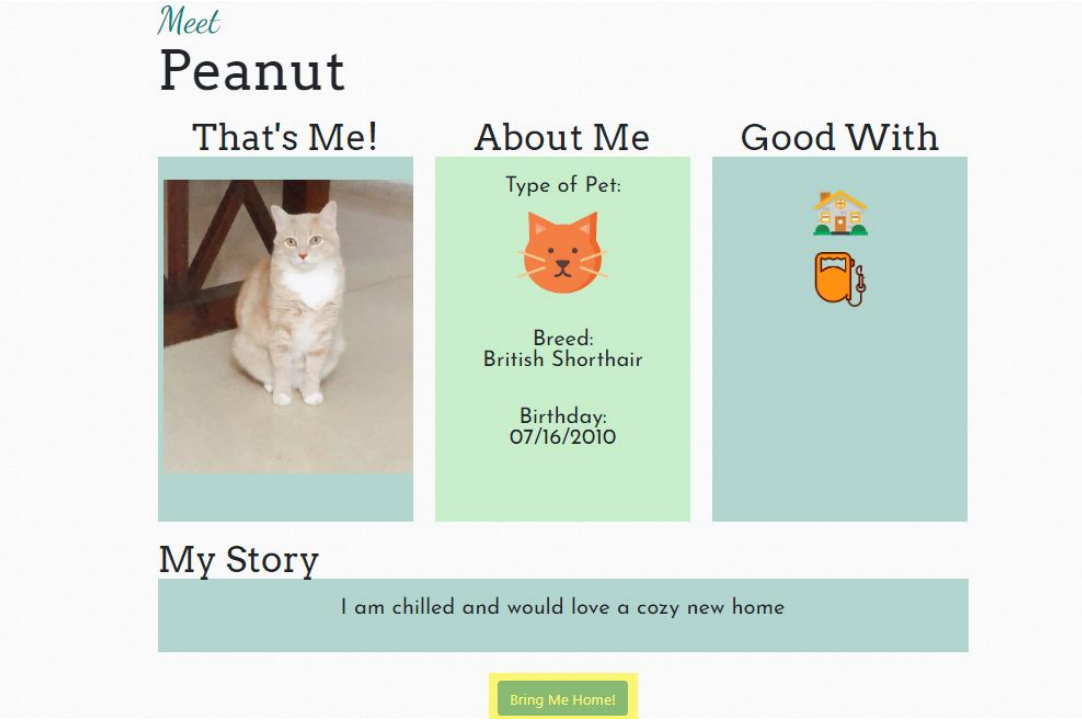
Search Pets

4. Click on “Search Pets” in the navigation bar or on the home page. The PetMatch Search page, by default, will display all pets that are currently in the database and available to be adopted.
5. To filter results, under “Find the perfect pet for you!”, use a dropdown to select the type of pet (dog, cat, or other) and optionally use a second dropdown to specify the specific breed of pet they are looking for. You can also select the “disposition” of the pet they wish to adopt, such as “Good with dogs”.
6. Once the filter is complete, click “Find pet!”. The images in the center of the page will update, and only the pets that meet the given criteria will appear. Alternatively, a quick filter can be done using the “All”, “Dog”, “Cat”, and “Other” buttons on the top of the page.
7. If a user did not find a pet that matched their particular criteria, they can click the “Subscribe” button at the bottom of the page. Click on this button to use your signed-in account to register for a mailing list that will inform you every time a new pet is added to the database.
8. Clicking on the name of any of the pets will direct you to their particular Pet Profile



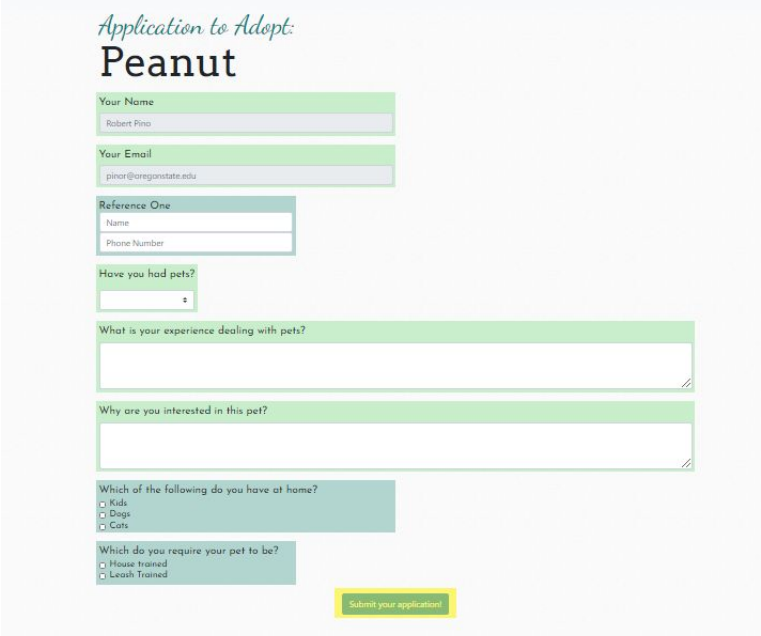
Pet Profile

9. Each pet has its own “Pet Profile”, akin to a user profile on a social media site. The “About Me” section displays a visual image to represent the species of pet, as well as the pet’s breed and approximate birthday. The “Good With” section displays images reflecting the pet’s disposition: if the image of the baby is present, that pet is good with children, etc. If an image is not present, then the pet has not been listed as having a positive disposition towards that subject.
10. Click on the “Bring Me Home!” button, which will direct you to the application to adopt that pet. You must be logged in to complete this step.



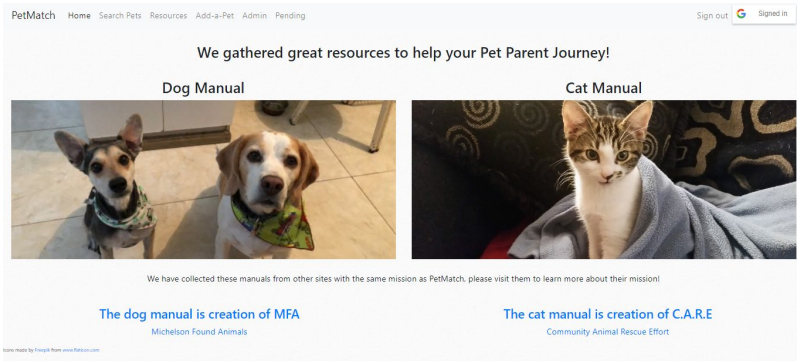
Adoption Application

11. Because you signed in at the beginning of their time at the site, the application is able to automatically populate your name and email address. You are required to provide a personal reference, as well as background information about your experience with pets, your reason for wanting a new pet, and what your current living situation is, including any kids, dogs, or cats already.
12. The “Submit your application!” button removes the pet in question from the searchable database. This pet is now moved to “Pending Adoption” and will be removed from the main search page.



Resources

13. You can visit the “Resources” page to learn more about the care of said pet. There are two links on this page- links to outside pages on taking care of newly adopted pets.

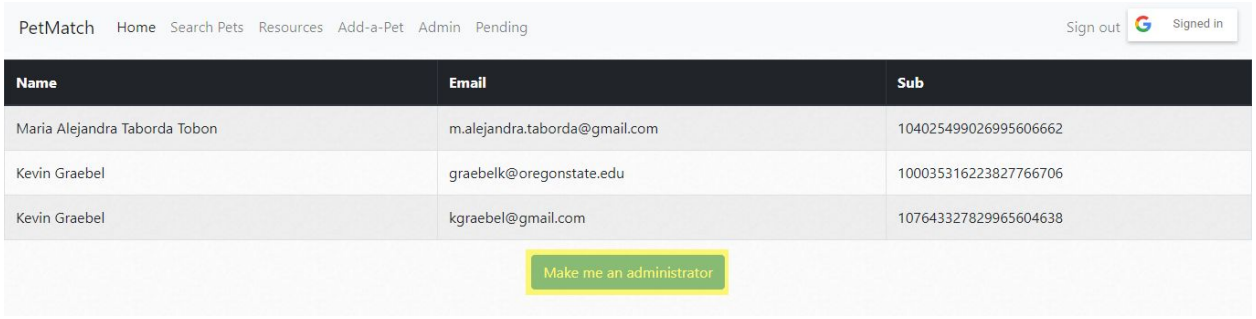


Administrator Use Case

An administrator can engage in all of the same tasks as a potential adopter, but with the additional ability to add new pets to the database.

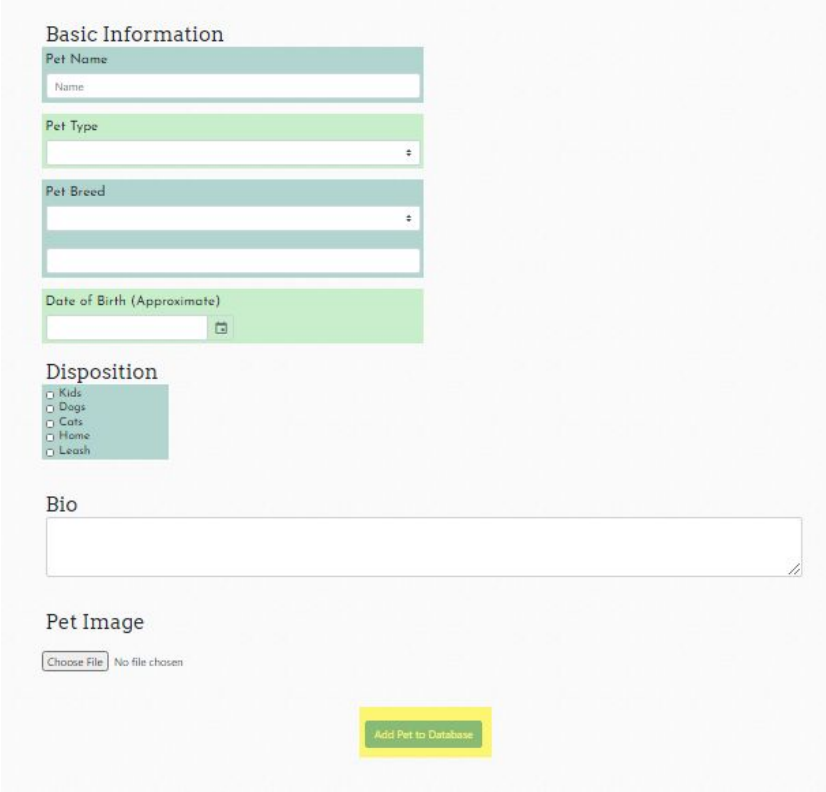
Admin

14. Click on the “Admin” link at the top of the navigation bar. This page lists all of the current administrators on the site. In a real world application, this would be hidden for non-administrators, but is public for demonstration purposes.
15. Clicking on the “Make me an administrator” button turns the account associated with the user’s Google account into an administrator account on the site, enabling you to access and use the “Add-a-Pet” page.

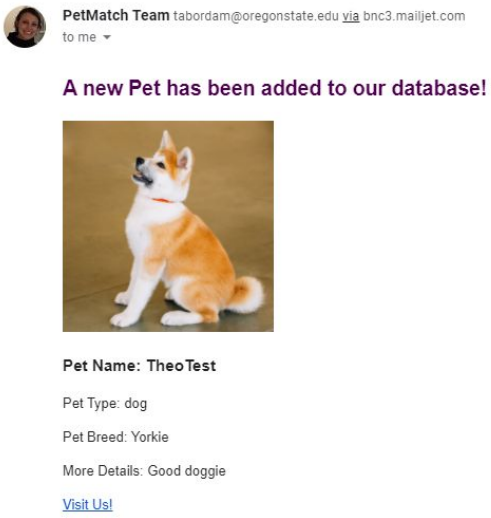


Add-A-Pet

16. Once admin rights are assigned, click on the “Add-a-Pet” link on the navigation bar.
17. The form on this page requires the pet name, type, breed, approximate date of birth, disposition, and biographical information. It also requires the user to upload an image of the pet. Unless every piece of information has been included when you click on “Add Pet to Database”, the form will prompt you to go back and complete the missing sections.
18. The “breed” section of the form changes depending on what “type” of pet selected. When “Dog” is selected, the breed section becomes a drop-down menu that only displays various breeds of dogs. When “Cat” is selected, the breed section becomes a drop-down menu that only displays various breeds of cats. If you select “Other” for pet type, the breed section becomes a text box, and you are required to enter in the specific type of pet.
19. Once the pet has been added, click on the “Search Pets” link and the new pet will be available.

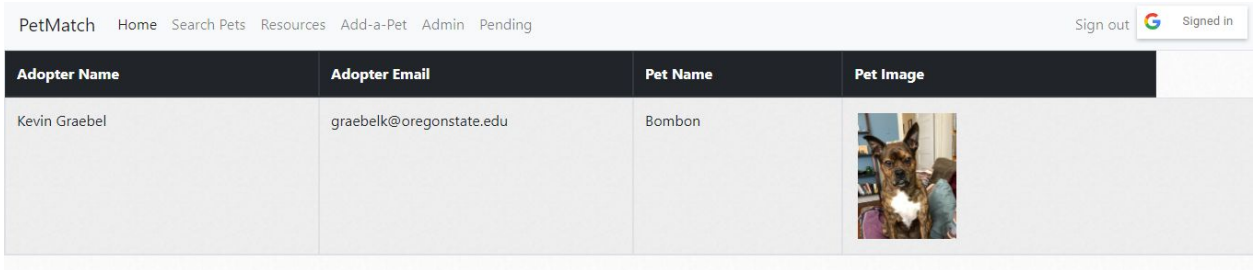


20. If you have subscribed to the site, you will receive an email notification including some details about the new pet that has been added. You can click on the image in this email to take you directly to the profile page for this pet.



Pending

21. This page populates a table with the name and email address of users who have submitted adoption applications for the pet that is being applied for, as well as the pet's picture. In a real world application, an administrator would review the user's application to approve or deny the request. The workflow for doing this final stage of adoption was considered out of scope for this project.

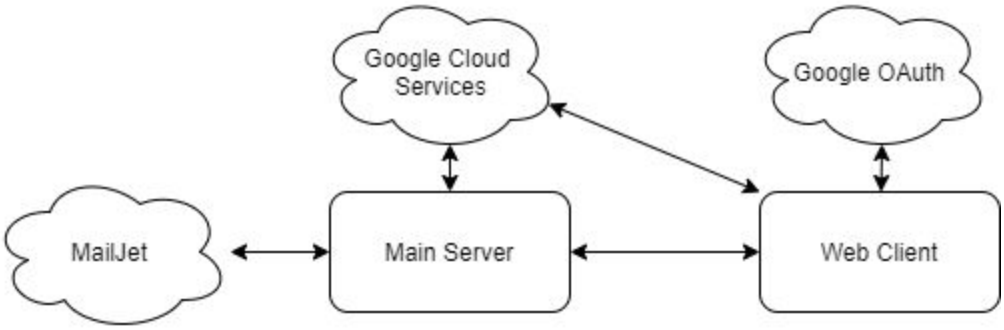


Completion

22. Once you have done exploring the page, click on the "Sign out" button at the top right hand corner of the page. This will remove the JWT stored locally on your computer that was used to manage the administrative flow of the page.

Software & System Integration

Our project was designed using a client-server architecture. The dynamic interactions are defined in the diagram shown below.



Database Interactions

The data for the pages was stored in Google's Cloud Datastore (data) and Storage (pictures). It was accessed through functions and filters defined in the server code.

- For basic search pages, the data was extracted based on a filter of "available", and returned to the page in a JSON format, sometimes as a single value and other times as an array of JSON objects. Handlebars express was used to dynamically populate the pages based on this array.
- For filtered searches, the web site generated a new GET request with the query in the URL, for example as search?type=dog. This was extracted from the URL and passed as a query filter to the database before returning the matching values

Administrative Flow

When a user clicks on the Sign In button, the google OAuth plug-ins store a JSON Web Token (JWT) on the client computer. This JWT is authenticated with the server and sent with POST requests to pages that require either a login for a guest or an administrative account.

Profile Pages

Pet profile pages used the JSON data from the server to build a more graphical experience. Attributes like "Good with dogs" were instead replaced with an icon by using javascript logic.

Languages, Libraries, and other systems used

HTML & JavaScript

As this project was scoped as a web application, HTML and JavaScript were used as a baseline for the code development.

Bootstrap

Bootstrap was used to define the layout for the pages to ensure a professional appearance and consistency in design.

Node.JS

The server backend was written in javascript using the Node.JS runtime environment.

Handlebars-Express

Handlebars was used to dynamically populate the necessary information on each of the web pages. This enabled the server to be able to pull data from the database and send a JSON object to the client and pull components from it as needed.

Google Cloud Datastore

Google Cloud Datastore (a NoSQL database) was used to host the dynamic information necessary to manage the site. This was selected because it is free, integrates well with Node.JS, and can also be used easily when the page is hosted on google.

Google Cloud Storage & Multer-google-storage

Google Cloud storage was used to upload and display images of the pets. The Multer-google-storage library was necessary to pull the image from the add-a-pet input form, extract it, and populate it in the database.

Google Auth library

The Google Auth Library was used to allow a secure login to the site, and eliminated the need for a separate, secure storage of passwords or custom user management. Instead, a user's

google credentials were used for pre-filling information and for validating access to protected areas.

Express-JWT & JWT-Decode

A JSON Web Token (JWT) was stored on the client's browser after a successful login. This is sent with certain requests to the server to be able to do protected actions, like adding a pet to the database, if the user is authenticated as an administrator.

GitHub

GitHub was used for version control of our code. All 3 members of the team contributed to building out the code to run the application.

Node-MailJet

To enable email functionality Mailjet was used, the API allows to create a contact and add them to a distribution list, then from that list it will subscribe the contact to a campaign, which is triggered every time a new pet is added to the database

Other APIs

For the daily feed, the project has a call to a dog (https://github.com/kinduff/dog-api) and a cat (https://documenter.getpostman.com/view/1946054/S11HvKSz?version=latest) api integrated into the main page.

Contributions from each team member

Kevin Graebel was responsible for the back-end, database, and administrative workflow. He built the structure for the server code and ensured that each of the site pages was served with the information necessary to display properly, including data from the google database. Kevin also was responsible for designing and integrating the administrative flow, including the google sign in and using this data for accessing protected parts of the site. Finally, Kevin supported the other team members with a variety of HTML related tasks, including integrating handlebars into their pages, by providing sample code and troubleshooting various issues.

Maria Taborda was responsible for the Landing page, Search page, Resources page and email functionality. The landing page includes integration with apis for fun facts/daily feed. The search page includes functionality to pass the right parameters to the backend for querying the pets, the resource page includes download of files functionality, lastly the email functionality is an api integration which allows the user to subscribe to the site and get notifications over email every time there is a new pet added to the database. Maria also created the prototypes for the site, which were the basis for the design and helped make all the pages responsive by handling image resizing across the site.

Robert Pino was responsible for the “Add-a-Pet”, “Pet Profile”, and “Adopt me!” pages. All three pages were responsive. The “Pet Profile” and “Adopt me!” pages represent the conclusion of the ‘guest’/‘potential pet adopter’ usage case, and the “Add-a-Pet” page represents the majority of the ‘administrator’ usage case. This included creating a consistent visual aesthetic, implemented via CSS, and determining a consistent and logical ordering of the pet information to be shown on all forms site-wide. He created the forms that were used on the “Add-a-Pet” and “Adopt me!” pages to pass the necessary pet information variables into the database, and he devised an intricate system of conditional code that was used on the “Pet Profile” pages, among other places, to dynamically display the relevant information from the database back to the user. He also devised the method that would be used site-wide to dynamically alter the “breed” input form to reflect which pet “type” had been selected in the previous form. He helped with various other UI/UX issues during development, including contributing the site-wide background colors texture. He wrote the introduction, conclusion, and usage flow sections of the final report.

Deviations from Original Plan

The team has held to the project plan, with one exception. The team decided to change from angular to node.js with express-handlebars to handle the dynamic page creation. This was decided as it was more broadly familiar to the team, and seemed like it would support the necessary functionality. The team also decided to undertake all of the “stretch” goals that were defined in the project plan, so we added some additional APIs to the original plan. Otherwise, the page design, flow, and functionality are based off of the initial project plan.

Conclusion

Overall, this project has provided us with a productive look into some of the challenges- and solutions- of real-world app development.

Our first major takeaway relates to focus. We were able to select this project because we were already passionate about the topic, and our preexisting knowledge allowed us to avoid many of the pitfalls and hurdles that we might have otherwise encountered. For example, because members of our group had, in the past, adopted pets through web portals, we entered the project with a clear understanding of what information/variables our app would need to manage, what methods it should use to collect that information from the user, and what format the resulting information should be displayed back out. This alignment, while not possible in every real-world development project, is an ideal development situation and a unique opportunity to expand our exploration and creativity.

Our second major takeaway was the importance of understanding the tools available to you as a developer, both in terms of the relationship between a tool and your skill with it, as well as in terms of the relationship between a tool and it's appropriateness for a particular job. Our initial plan with the project was to implement the majority of the front-end in Angular, a powerful platform which some of our team members had experience with and others had never used before. As we worked through the first week, we came to the conclusion that it was not an appropriate fit for the task we were trying to accomplish. As a result, despite us having worked to develop institutional knowledge on that particular tool, we exchanged it for another that was more appropriate for our needs. When we made our shift to Bootstrap, there was a steeper learning curve that had to take place, but because the tool more adequately fit the needs of the final product, this change was necessary. On many occasions, our team would share unique pieces of code that we had found to implement in one granular instance, and that code would end up influencing code somewhere completely different.

Collaborating on this project over the last semester, our team has created an application that we are proud to display to you and to others who might view our portfolios. Our project builds on something that we are personally passionate about, and that is globally relevant, creating for us a best-case-scenario within which to develop and learn new development skills. We sincerely hope that you enjoy exploring our web application.

Appendix - Requirements Traceability Matrix

Core Requirements

Use Case/Requirement	Steps/Highlights	Status
Accounts and login. Admin and public accounts.		Complete
Shelter needs to be able to create new profiles, with type of animal, breed, disposition, pic, availability, status, and description.	1- Navigate to the webpage URL https://tgp-pet-cs467.uc.r.appspot.com/ 2- Navigate to the add a pet either via the nav bar or the button in the landing page 3- Enter the information required 4- The pet will be added to the database and visible in the search, the picture functionality is still to be implemented, a placeholder image will appear	Complete
Users can browse profiles to find a pet, or search by type, breed, disposition, or date created.	1- Navigate to the webpage URL https://tgp-pet-cs467.uc.r.appspot.com/ 2- Navigate to the search a pet either via the nav bar or the button in the landing page 3- Select some of the filters to test	Complete

Animal types include dog and cat plus "other".	This has been included in the front and backend	Complete
Breed shall be the most common ones plus "other"	Open the search app and search the pets by breed, enter a new pet specifying the breed	Complete
Dispositions are checkboxes that include "Good with other animals", "Good with children", "Animal must be leashed at all times". Availability includes "Not Available", "Available", "Pending", "Adopted"	1- Navigate to the webpage URL https://tgp-pet-cs467.uc.r.appspot.com/ 2- Navigate to the search a pet either via the nav bar or the button in the landing page In the search page itself we can find some of the filters which are the dispositions 3- Pick a pet to navigate to the pet profile, the dispositions of the pet can be seen in the profile	Complete
App/website should have a landing page.	1- Navigate to the webpage URL https://tgp-pet-cs467.uc.r.appspot.com/ the home page will be loaded, user can navigate to other pages in the site from here as well as see featured pets and some general information	Complete

Status is just a news/PR blurb. Update it to send the status and image of animal to the Daily feed queue (stretch requirement)	Can be displayed in home page	To meet this stretch requirement we used a daily feed quote from a cat-fact and dog fact api, we included some featured pets in a carousel image as well
Profiles can be removed when the animal is picked up.		Complete

Stretch Requirements

Use Case/Requirement	Steps/Highlights	Status
Website needs to be responsive (viewable on desktop or mobile).	All pages are designed to be responsive. Shrink the width and columns will re-order	Complete
Email notifications for newly added profiles	Go to the search page and subscribe to get information on new pets Add a new pet (you need to be an admin first) Check your email - you should get an email notifying that there was a new pet added to the database	Complete
Pics can rotate through a set of images	Home page rotates through a carousel of pet images.	Complete