

Your Perfect Pet: Pet Adoption Website

Website URL: <http://flip2.engr.oregonstate.edu:9900/>

The idea for our project came out of both of our love for animals; it was one of the things we immediately clicked on in our first meeting and seemed like an obvious starting point for this project. The main goal of our project, to create a website that lists people and animals and tries to match them, has stayed consistent, but the breadth of the endeavour has been paired down. Most of the edits we made over the course of development served to make the organization of the project and the implementation of the database backend more straightforward, both for our development needs and timeline, and to ease the experience of the hypothetical end user. Around week three we had a “Eureka” moment when we realized we could streamline the logical system by which we paired potential adopters with animals, which really enabled us to get the ball rolling on the development.

There were a number of UI changes we made along the way. Initially, we had intended for our website to be one page with conditional formatting dictating which forms appeared and disappeared for the user, as needed. We decided that, while such an experience would be sleek it could be confusing for the user, as well as being a source of complication for the project that was unnecessary for completing the delineated tasks in the project document. Also, after a number of peer reviews some students were finding this layout to be confusing. To feed two birds with one scone, we decided to break all of the modules out into their own pages, with links on each page to make the UX more straightforward and simple. In the same vein, we decided to restrict the domain of the project to zip codes related to the city of Portland, OR because trying to map the whole country was, frankly, a silly thing to waste time on, and not something that added a lot of value to the project itself. This stage of development consisted of a lot of reality checks, and having to take a step back to understand that what we originally sought out to accomplish wasn't impossible, but was very ambitious, and we could develop a proof of concept with a more narrow set of data points.

With regards to developing the backend, once we began to plan the particular implementation of the database, we realized we needed to change around some primary keys and also implement two relationship tables to manage the many-to-many relationships between people and locations and people and species. This wasn't necessarily a hurdle we had to overcome, but was something that became clear as we learned more about the relationships between different tables in a database and the ways in which they are managed by the backend system of choice.

Project Outline

Introduction: Our website will serve as a matchmaking service between people looking for a new pet and animals in need of a home in the Portland, Oregon. Portland will be the Pilot test city before the website is expanded to other cities across the USA. The entities we will define are the animals seeking homes, the people seeking pets, the locations of the people and animals. People seeking pets will choose what types of pets they are looking for (dog or cat, for example), and the animals will be classified by what type of animal they are. The relationships between all four entities will generate a list of perfect matches between people in need of pets and animals in need of homes.

Entities:

Entity I: Animals

Attributes:

- ANIMAL_ID
 - Data type: auto-incrementing integer
 - Constraint: Primary Key
- Name
 - The name of the animal that is up for adoption, assigned by the rescue group
 - Data type: VARCHAR(255)
- Age
 - Approximate age, in years only
 - Data type: integer
- Weight (in lbs)
 - Data type: int(11), zero value indicates less than 1 lb of weight
- Location_ID
 - Data type: int(11)
 - Nullable
 - Constraint: Foreign Key
- SPECIES_ID
 - Data type: integer
 - Constraint: Foreign Key

Entity II: People

Attributes:

November 11: We have chosen to split up the name attribute into first_name and last_name

- People_id
 - Data type: auto-incrementing integer
 - Constraint: Primary Key
- First name
 - Data type: VARCHAR(255)
- Last name
 - Data type: VARCHAR(255)
- Age
 - Data type: integer
 - Unit: years
- Location_ID (See update note under location)
 - Data type: int(11)
 - Constraint: Foreign Key
- SPECIES_ID
 - Data type: integer
 - Constraint: Foreign Key

Entity III Location

Attributes:

November 11: We have chosen to restrict the domain of our database to that of Portland, Oregon. We have also chosen to restrict ourselves to five digit zip codes as opposed to the full nine digit versions. Portland, OR has 34 five digit zip codes assigned to it which we decided was sufficiently large for the purposes of this project.

- Location_id
 - Data type: int(11), auto-incrementing
 - Constraint: Primary key
- Zip code
 - Five digit series of numbers that in the United States corresponds to a specific area within a specific City, State pair (for larger areas of the country a city, state pair may have more than one zip code assigned to it, however zip codes never overlap).
 - Data type: VARCHAR(5), text in the format #####
- City
 - Data type: VARCHAR(255)
- State
 - Two letter abbreviation corresponding to US state.

- Data type: CHAR(2) format “XY”

Entity IV: Species

Attributes:

- SPECIES_ID
 - Data type: auto-incrementing integer
 - Constraint: Primary Key
- SPECIES_NAME
 - Data type: VARCHAR(255)
 - Name of the animal type (for example, dog or cat)

Relationships:

1) **Many-to-many:** Species to People

Each person could be looking for a different type of animal. For example, Person X could want a dog or a cat. Each type of animal could be paired with multiple people. For example, Person X wants a cat but so does Person Y and Person Z. Therefore, this relationship is many-to-many.

2) **One-to-many:** Species to Animal

Each animal will only be classified as one species. For example, Animal A is a dog. Therefore, it will only have one species ID associated with it, namely, the species ID for dog. However, there could be many distinct animals in the database which are dogs. Therefore, the relationship between species and animal is one-to-many.

3) **Many-to-many:** Location to People

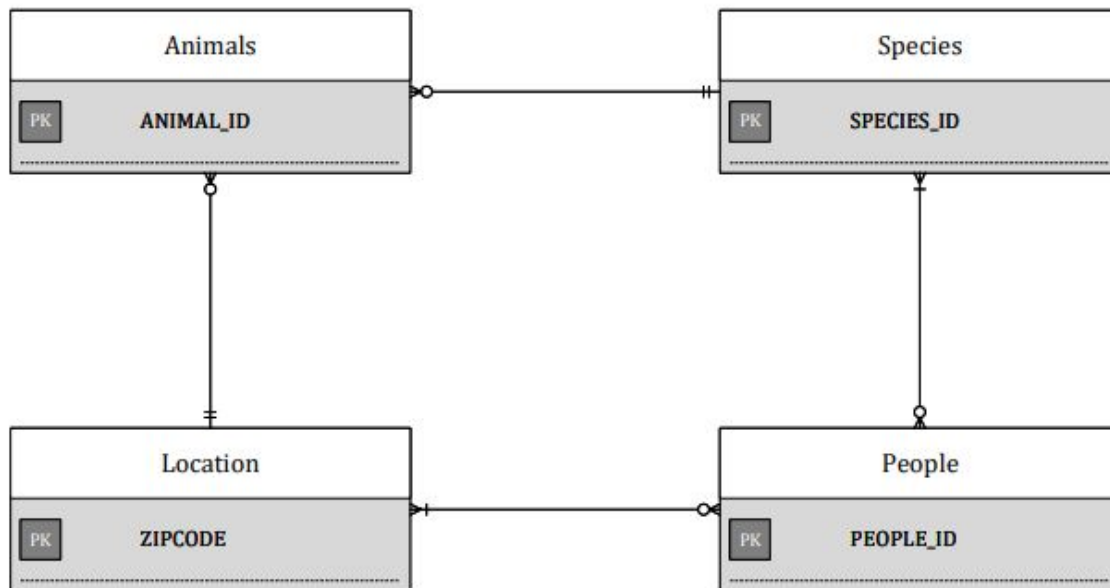
In regards to a person, the zip codes they are searching for correspond to the regions in which they are searching for an animal. Even though they might live in zip code A, they could be searching for an animal in zip codes A, B, C, and D, because they are willing to go outside of their immediate region to find an animal. On the other hand, zip code A could have multiple people associated with it. Therefore, Location to People is a many to many relationship.

4) **One-to-many:** Location to Animal

An animal can only physically be in one location. However, a single location can be tied to many different animals. Animal X could be in Houston but so could Animal Y and Animal Z. Therefore, this defines a one-to-many relationship.

ER Diagram:

ER Diagram
Your Perfect Pet: Pet Adoption Website



Each animal is exactly one species.
Each species can apply to many animals.

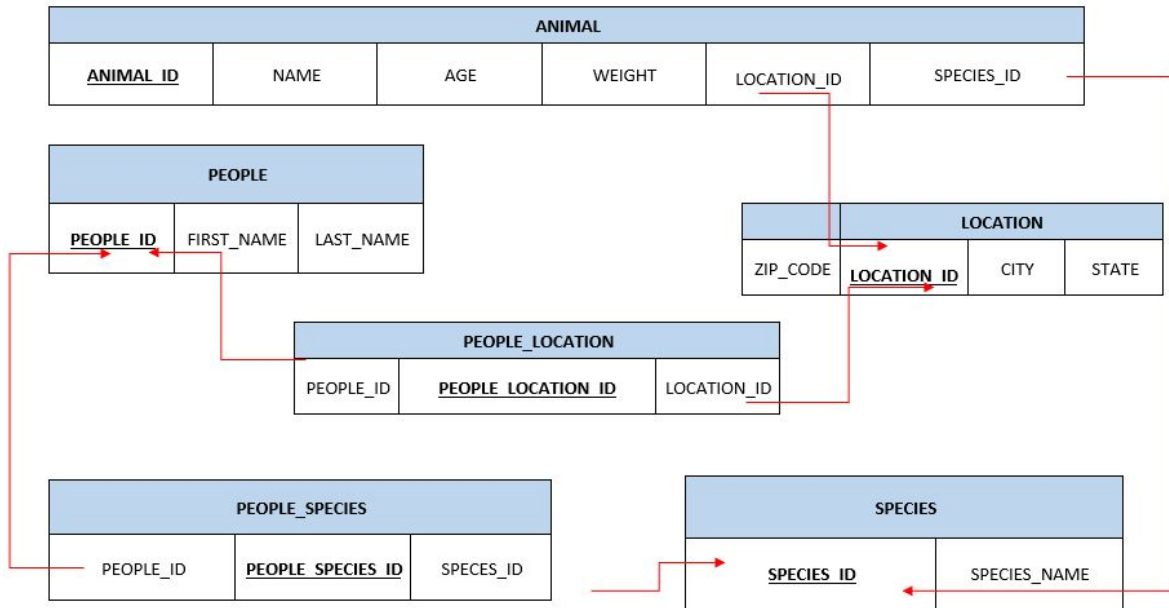
Each person is looking for at least one species.
Each species can be matched with many people.

Each Person has at least one location
Each animal has exactly one location

Each Location can have many people
Each Location can have many animals

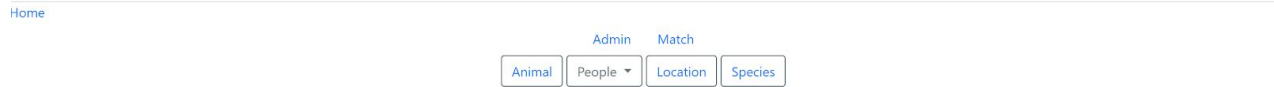
Schema:

CS340 Project Schema

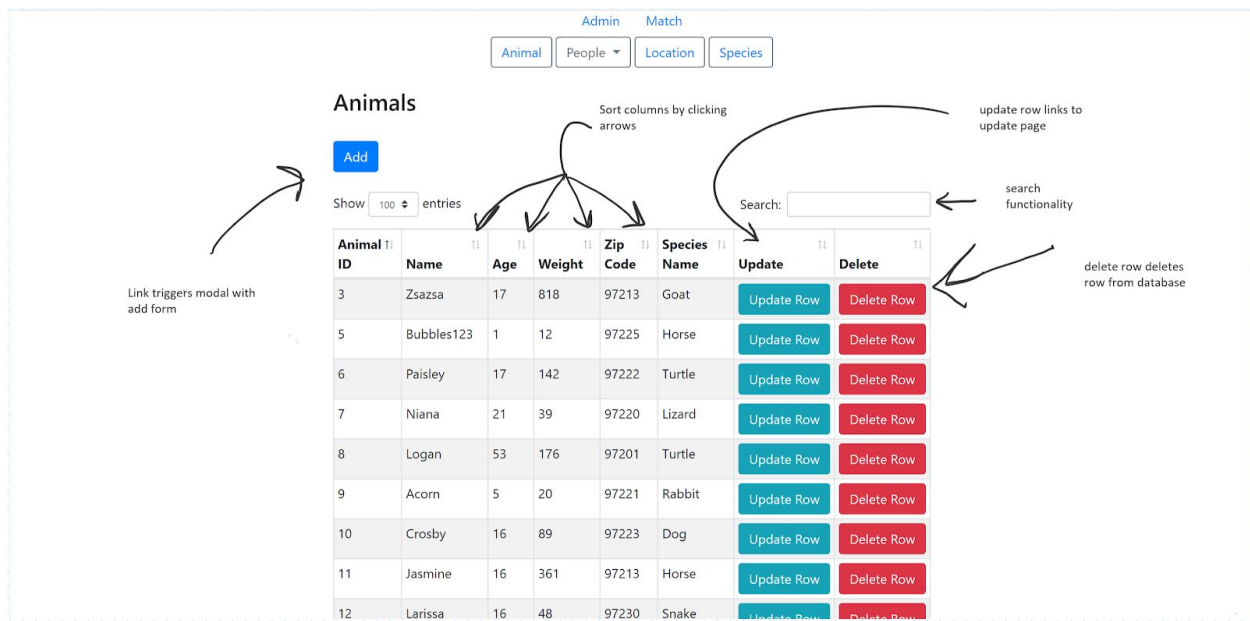


UI Screen Captures

Admin Page with Links to pages related to each entity



Animal Page with add, delete, update, sort, filter (search) functionalities



Add Modal detail of add functionality

Animal Add

Name
Boris

Age (years)
0-99

Weight (lbs)
200

Zip Code (5 Digit)
97201

Species
DOG

Add Animal

add UI is the same for all entities with the only differences being the relevant fields

Animal ID	Name	Age (years)	Weight (lbs)	Zip Code	Species	Update Row	Delete Row
3	Zsaz	5	20	97221	Rabbit	Update Row	Delete Row
5	Bub	16	89	97223	Dog	Update Row	Delete Row
6	Pais	16	361	97213	Horse	Update Row	Delete Row
7	Nia	16	48	97230	Snake	Update Row	Delete Row

Update Animal Page

Update Animal

Name
Bubbles123

Age (years)
1

Weight (lbs)
12

Zip Code
97225

Species
HORSE

Update Animal

update page pre-populates will data from relevant row

People Page with add, delete, sort, filter (search) functionalities for M:M relationship

Home

Admin Match

Animal People Location Species

Person
People_Location
People_Species

drop down menu for main people page and both people relationship tables

search

People:

Add

Triggers add modal, similar to animal page

Show 100 entries

Search:

People ID	First Name	Last Name	Delete
122	HAZEL	ADAMS	Delete
143	JOSHUA	ADAMS	Delete
180	DANIELLE	ALLEN	Delete
111	MELVIN	ANDERSON	Delete
171	CARL	BAILEY	Delete
188	CHRISTI	BAILEY	Delete
20	JANA	BAILEY	Delete
16	ANDRE	BAKER	Delete

Deletes all rows from relevant tables

People_Location Relationship Page with add, sort, filter (search) and delete functionalities

Home

Admin Match

Animal People Location Species

People and Their Selected Locations:

Add

Show 100 entries

Search:

PEOPLE_LOCATION_ID	PEOPLE_ID	First Name	Last Name	LOCATION_ID	Zip Code	Delete
1	1	NANETTE	BROWN	13	97214	Delete
2	1	NANETTE	BROWN	34	97267	Delete
3	1	NANETTE	BROWN	3	97203	Delete
4	2	TRACEY	HALL	30	97233	Delete
5	2	TRACEY	HALL	11	97212	Delete
6	3	TONY	TORRES	25	97227	Delete
7	3	TONY	TORRES	13	97214	Delete
8	3	TONY	TORRES	30	97233	Delete

Deletes only this row from the people table

People_Species Relationship Page with add, delete, sort, filter (search) functionalities

[Home](#)

[Admin](#)[Match](#)

[Animal](#)[People](#)[Location](#)[Species](#)

People and Their Selected Species:

Add

Show 100 entries

Search:

PEOPLE_SPECIES_ID	PEOPLE_ID	First Name	Last Name	SPECIES_ID	Species Name	Delete
1	1	NANETTE	BROWN	7	SNAKE	Delete
2	2	TRACEY	HALL	8	LIZARD	Delete
3	3	TONY	TORRES	2	CAT	Delete
4	4	MARA	ROBERTS	5	BIRD	Delete
5	4	MARA	ROBERTS	1	DOG	Delete
6	5	MADELEINE	EDWARDS	10	GOAT	Delete
7	6	LATONYA	THOMPSON	4	TURTLE	Delete
8	6	LATONYA	THOMPSON	10	GOAT	Delete

Deletes only this row from the people table

Locations Page with add, delete, sort, filter (search) functionalities

[Home](#)

[Admin](#)[Match](#)

[Animal](#)[People](#)[Location](#)[Species](#)

Locations

Add

Show 100 entries

Search:

Location ID	Zip Code	City	State	Delete
1	97201	PORTLAND	OR	Delete
2	97202	PORTLAND	OR	Delete
3	97203	PORTLAND	OR	Delete
4	97204	PORTLAND	OR	Delete
5	97205	PORTLAND	OR	Delete
6	97206	PORTLAND	OR	Delete
7	97208	PORTLAND	OR	Delete
8	97209	PORTLAND	OR	Delete
9	97310	PORTLAND	OR	Delete

Species Page with add, delete, sort, filter (search) functionalities

[Admin](#) [Match](#)

[Animal](#) [People](#) [Location](#) [Species](#)

Species

[Add](#)

Show entries Search:

Species ID	Species Name	Delete
1	DOG	Delete
2	CAT	Delete
3	RABBIT	Delete
4	TURTLE	Delete
5	BIRD	Delete
6	HORSE	Delete
7	SNAKE	Delete
8	LIZARD	Delete
9	PIG	Delete

Match page, links to admin page are now removed (because of operation in a different mode)

[Home](#) [Admin](#) [Match](#)

Matching Pets with People

Select Zip Code and Species:

Zip Code:

Species:

[Find Matches!](#)

If no rows return in either table, then there were no matches for this particular species and zip code combination.

Matched People

First Name	Last Name
CAROLINA	HOWARD
BECKY	DAVIS

Matched Animals

Name	Age	Weight
Nanda	2	1988

sortable

Two fields by which to search for matches

Matches returned from given inputs

Match page detail: two drop down menus pre-populated with valid values by which to find matches

[Home](#)

[Admin](#) [Match](#)

Matching Pets with People

Select Zip Code and Species:

Zip Code:

97204

Choose Zip Code Here:

97201

97202

97203

97204

97205

97206

97208

97209

97210

97211

97212

97213

97214

97215

97216

[Home](#)

[Admin](#) [Match](#)

Matching Pets with People

Select Zip Code and Species:

Zip Code:

97204

Species:

HORSE

Choose Species Here:

DOG

CAT

RABBIT

TURTLE

BIRD

HORSE

SNAKE

LIZARD

PIG

GOAT

PANDA