CS316 Group Project Milestone II
November 6th, 2019
Saloni Bulchandani, Michelle Li, Charlton Lu, Tanisha Nalavadi, Dian Niu

Progress Report

New assumptions since Milestone 1

We are assuming that owners are okay with their phone number and email address being available to all adopters as these are not nullable fields, hence we are taking an owner signing up for the website as implicit permission to share their contact information. In fact, for this privacy reason, we decided not to share address of the owner with adopters and have gotten rid of the "owners near me" functionality. Instead, if the adopter is interested in a dog, they can contact the owner and obtain their address that way. Because of these privacy issues, we have also made it so that adopters also have to create accounts, so that owner details are not shared with the entire public, just with potential adopters.

Brief description of tech stack

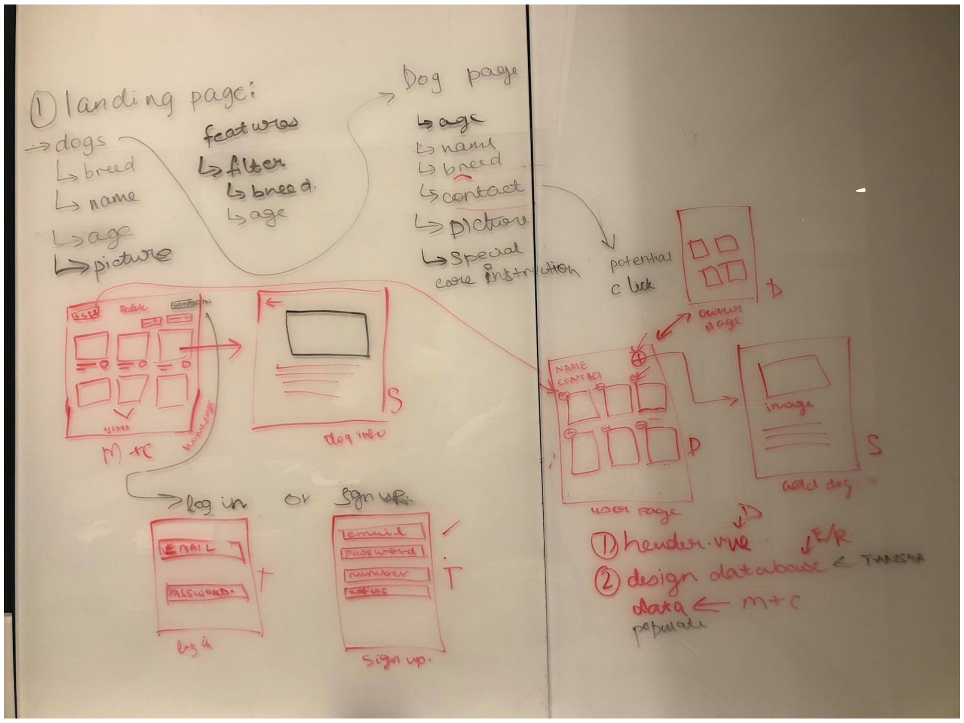
Our database is currently implemented in Firebase, which is built loosely on a NoSQL implementation. Firebase is popular for mobile app development with low query thresholds.
Our frontend uses vue.js, a javascript framework for frontend development with HTML and CSS elements. We decided to use this tech stack because we have team members with more experience in javascript as well as a team member who has worked specifically with this tech stack in a previous project before who was willing to build the initial integrated MVP.
In order to do basic query testing before we have a working web application, we used Firestation, an open source project specifically for testing SQL queries on Firebase data. All of our test queries are included in the txt file and the text output . However, Firestation is a visual output platform, so a screenshot of the result of each query is included in this document.

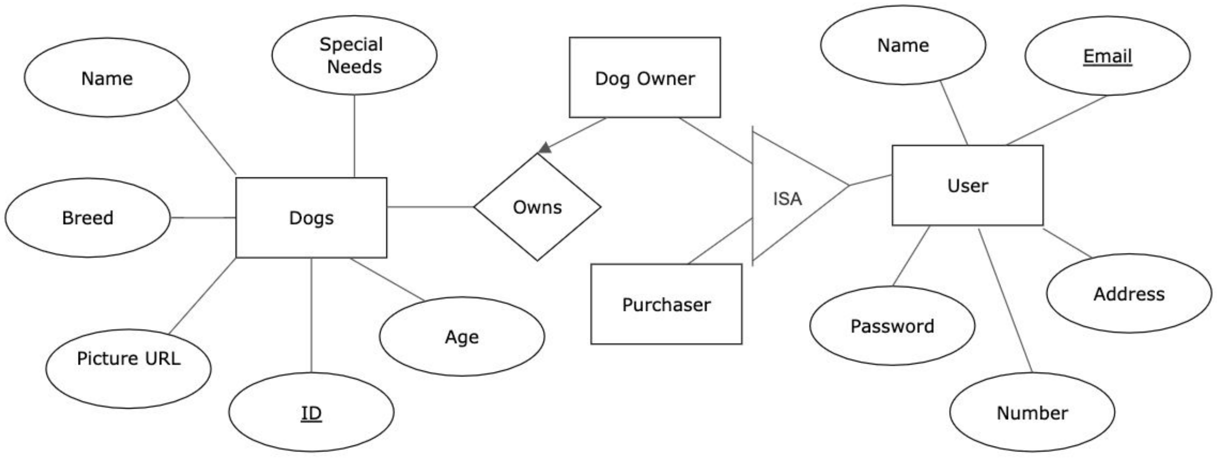
Changes we've made to database

We decided to modify our web application and connect people who wanted to put their dogs up for adoption to potential adopters. Thus adopters can go in and access a list of dogs- they will have the ability to filter the dogs that are being sold according to their preferences (like age, breed etc). They can they access the individual dog profiles that provides additional information like contact details of the seller.
On the other hand, current owners can go in and add dog profiles for dogs that they want to put up for adoption, as well as delete dog profiles for dogs that have been adopted. The profiles added will automatically be added to the main page containing all the dogs that are being put up for adoption.

Below is a rough wireframe of the various pages and the user flow:

CS316 Group Project Milestone II
November 6th, 2019
Saloni Bulchandani, Michelle Li, Charlton Lu, Tanisha Nalavadi, Dian Niu





With regards to the new ER design above, while the user object does store a password attribute in the diagram above, in reality during implementation we plan on using Firebase's authentication service that will get rid of the password being stored. The email will act as the unique identifier for every user and dogs will be uniquely identified with dog IDs. Furthermore, every dog will have exactly one Owner that Owns them, through the Owns relation.

CS316 Group Project Milestone II
November 6th, 2019
Saloni Bulchandani, Michelle Li, Charlton Lu, Tanisha Nalavadi, Dian Niu

New Schema:

<u>ID</u>	Name	Breed	Age	Special_Needs	Picture_URL	Owner_Email
123	Bruno	Pug	4	none	www.pug.com	fh78@duke.edu
321	Mars	Dalmatian	2	Hip dysplasia	www.dalmation.com	sk87@duke.edu

Dog Table with example data

<u>Email</u>	Name	Address	Number	Password	Type
dh78@duke.edu	Sam	Kilgo	2323232323	123	Owner
df93@duke.edu	Holly	Few	232324443	123	Adopter

User Table with example data

Additionally, when we converted the ER diagram to tables- we dissolved the ownable relationship to the unique user id being stored for every dog as a dog must have exactly one owner. Thus, for every owner they can access their dogs that they have put up for selling by querying the dog database with the owner email.Since owners and adopters have the same traits (except owners can own pets), the user table holds a type attribute and we can then introduce a constraint that dogs can only hold the email for users of type owner and not adopter. We will also have a constraint when inserting users that Type can only be Owner or Adopter.

CS316 Group Project Milestone II
November 6th, 2019
Saloni Bulchandani, Michelle Li, Charlton Lu, Tanisha Nalavadi, Dian Niu

OTHER FILES

Text file called members.txt describing what each of us has done [DIAN]

EVERYONE: attended meetings, participated in decision making process, agreed on visual design of website and split work in advance
Dian: Set up documentation workflow and assigned tasks, set up firebase query testing environment, wrote one query, tested and modified all queries, contributed to write up
Saloni: Minimum web application product, wrote four queries, “new assumptions“
Tanisha: ER diagram redesign, contributed to write u (“Changes we’ve made to the database”)
Charles: Generated database and wrote README.txt for database
Michelle: Suggested new tech stack and showed us previous work using said tech stack, worked with charles for generating database + database write up

ReadMe for how to generate the production dataset and load it to the website [CHARLIE]

QUERIES

test-production.sql: containing SQL statements

[DogInfo] Get all of dog’s information after click

SELECT * FROM Dogs; //all dogs
SELECT * FROM Dogs WHERE breed LIKE ‘%Hound%’ AND age= ‘10’;

```
select * from Dogs where
   age > 10
   and breed LIKE "hound";
```

[Userpage] Get all of user’s associated dogs

SELECT * FROM Dogs WHERE owner_email=’example_email@gmail.com’;
SELECT * FROM Dogs WHERE owner_email=’mbrown@live.com’;

CS316 Group Project Milestone II
November 6th, 2019
Saloni Bulchandani, Michelle Li, Charlton Lu, Tanisha Nalavadi, Dian Niu

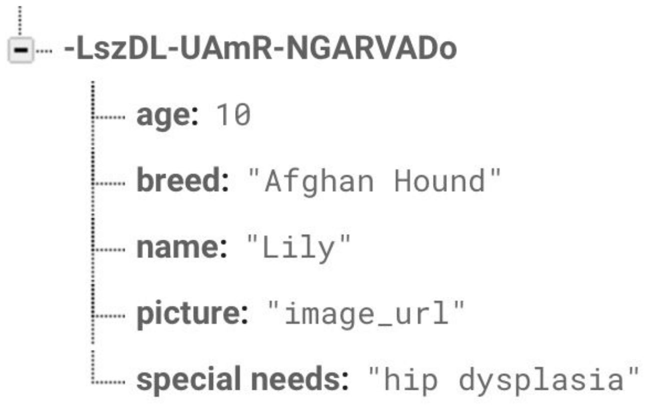
Firestation safe
select * from Dogs where user-email='pierce@msn.com';

[Owner page] Populate with owner's information
SELECT * FROM Users WHERE email='example_email@gmail.com';
select * from User where email = 'pierce@msn.com';

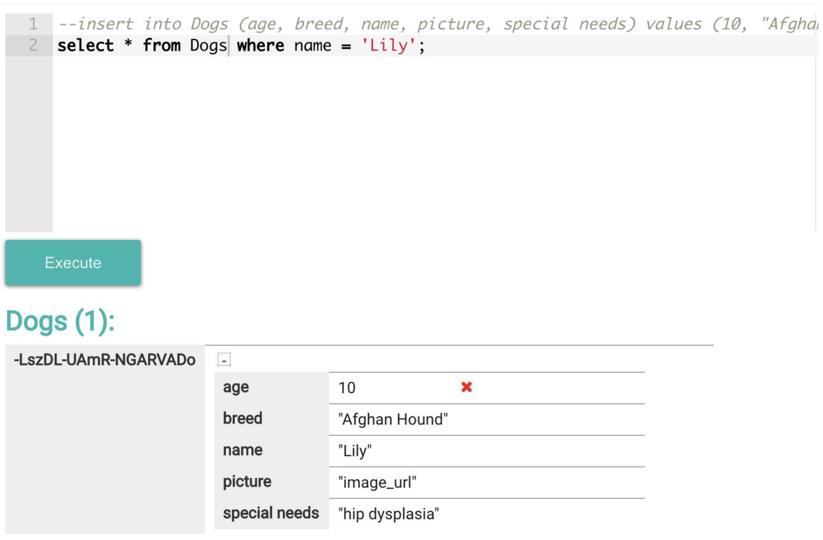
[Signup] Add the new user parameters to the users database
insert into User (address, email, name, number, password) values ("somewehre", "email@domain.org", "name", "9999999999", "badpassword123");

[AddDog] Add a dog to the database ✔️
insert into Dogs (age, breed, name, picture, special needs) values (10, "Afghan Hound", "Lily", "image_url", "hip dysplasia");

CS316 Group Project Milestone II
November 6th, 2019
Saloni Bulchandani, Michelle Li, Charlton Lu, Tanisha Nalavadi, Dian Niu



```
-LszDL-UAmR-NGARVADo
    age: 10
    breed: "Afghan Hound"
    name: "Lily"
    picture: "image_url"
    special needs: "hip dysplasia"
```



```
--insert into Dogs (age, breed, name, picture, special needs) values (10, "Afgha
select * from Dogs where name = 'Lily';
```

Dogs (1):

-LszDL-UAmR-NGARVADo	age	10
	breed	"Afghan Hound"
	name	"Lily"
	picture	"image_url"
	special needs	"hip dysplasia"

[Login] Authenticate in the firebase

Tech/production/out: showing results of the /sql

WEBSITE CODE

Github repo link:

Note: this website is not currently integrated with the backend. Right now this simple but working web app has two working buttons, the home button and the “click for example pet” button.

CS316 Group Project Milestone II
November 6th, 2019
Saloni Bulchandani, Michelle Li, Charlton Lu, Tanisha Nalavadi, Dian Niu

Code for a simple but working web app [dog info]

Pages to build

- Header [Dian]
- Homepage [M+C]
- DogInfo [Saloni]
- Login [Tanisha]
- SignUp [Tanisha]
- OwnerPage [Dian]
- Userpage [Dian]
- AddDog [Saloni]
- Index.html [Dian]