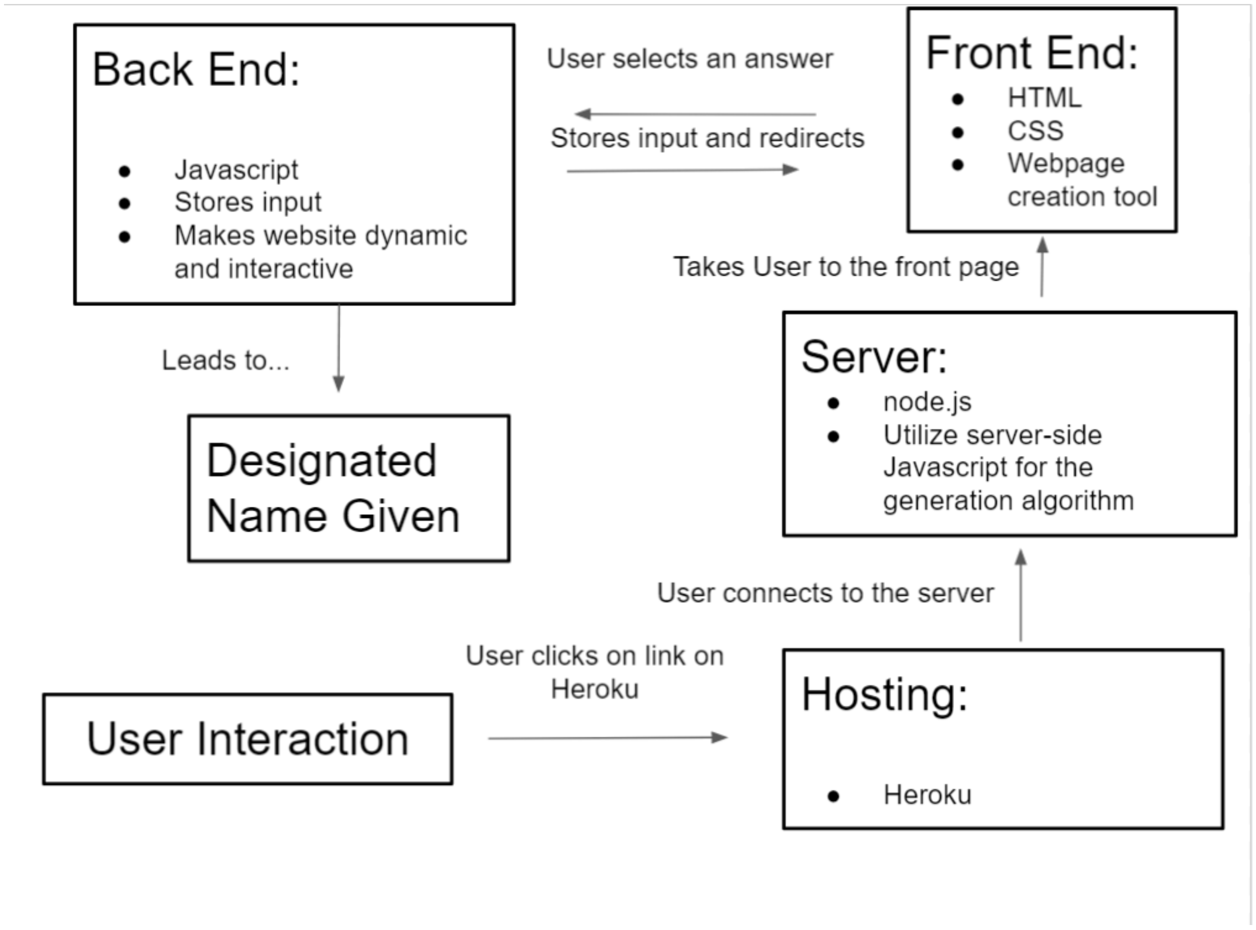
Team Members:

- Owen Smith: osmith-CU
- Jingqi Liu(JanesyLiu): jili3523
- Mathew Kim: maki9472
- Patrick Taylor: pata0278
- John Hodgen: joho8797

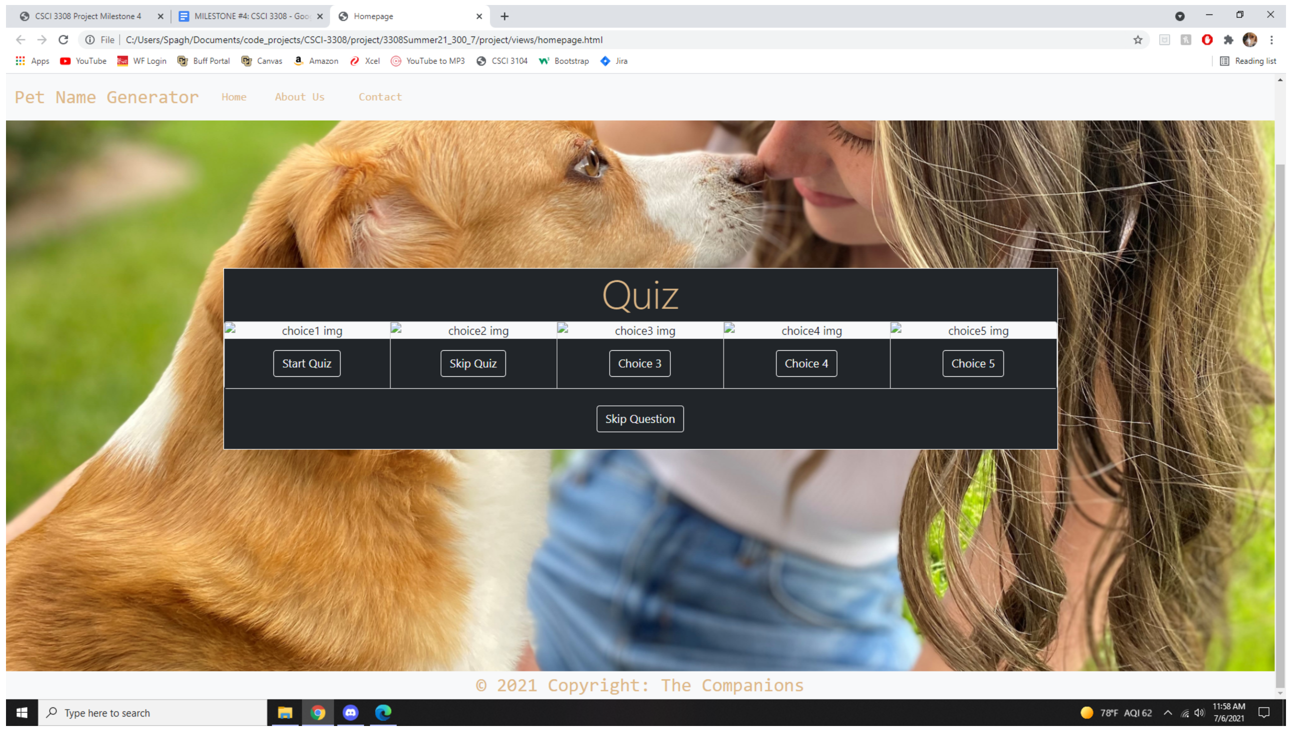
Revised List of Features:

- Added “About Us” and “Contact” page
- Further revised the home page and made the quiz portions and options look more concise
- Cleaned up HTML and CSS Code
- Created mock txt files for randomly generated names
- Created javascript function to return 5 random names from SQL table
- Created SQL script to do BULK Insert of txt files to SQL server
- Created SQL script to create a table where none exists for testing javascript

Architecture Diagram:


Back End:
Javascript
Stores input
Makes website dynamic and interactive
User selects an answer
Stores input and redirects
Front End:
HTML
CSS
Webpage creation tool
Takes User to the front page
Leads to...
Designated Name Given
Server:
node.js
Utilize server-side Javascript for the generation algorithm
User connects to the server
User clicks on link on Heroku
User Interaction
Hosting:
Heroku

Front End Design:



(not including diagram because front-end is basically complete)

Design:

- Copyright banner indicating ownership of our property
- Dynamic quiz box composed of bootstrap cards; dynamically grows and shrinks depending on the number of answers for given question (see in git repo for demo)
    - At most five answer choices for any given question
    - Replaceable question text (replaces "Quiz" in actual questions)
    - Images associated with each answer (pending implementation for copyright-free images)
    - Button to skip the question for any given answer, blacklists names for that answer being polled for final namedraw
- Dynamic, readable sticky navbar
    - Includes link to about us page explaining website and team
    - Includes link to contact information for questions or concerns about the site
    - Includes link to homepage such that the quiz can be restarted and the page refreshed at any time

Still needs implementation:

- Login/email-access service to access acquired names so that the user can keep a list of name options for later, can be a separate page

Web Service Design:

Setup Heroku app for mock implementation and testing

Setup Heroku database with PostgreSQL add on

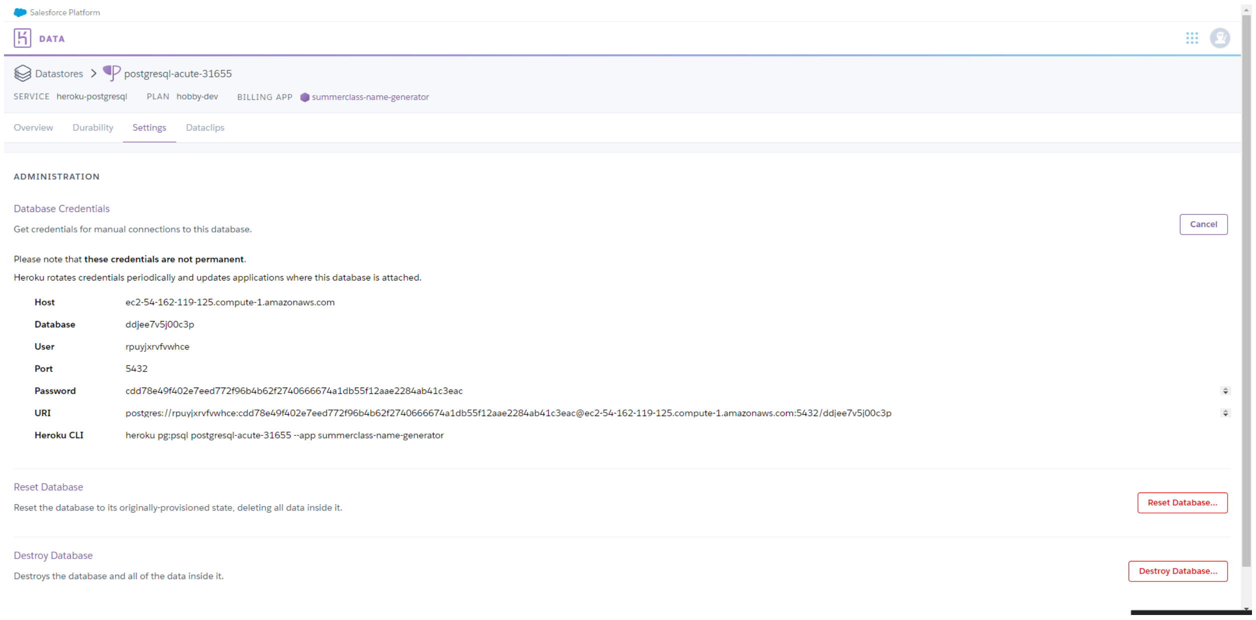
Database is ready to receive deployments for tables and files

-JS file will query the database with selected traits matching each column

-database will return all names of these traits to the JS file

-JS file will query the database with userEmail and passkey

-database will return all past generated names up to ten most previous to JS file



SERVICE heroku-postgresql PLAN hobby-dev BILLING APP summerclass-name-generator

ADMINISTRATION

Database Credentials

Get credentials for manual connections to this database.

Please note that **these credentials are not permanent**.

Heroku rotates credentials periodically and updates applications where this database is attached.

Host	ec2-54-162-119-125.compute-1.amazonaws.com
Database	ddjee7v5j00c3p
User	rpuyjxrvfvwhce
Port	5432
Password	cdd78e49f402e7eed772f96b4b62f2740666674a1db55f12aae2284ab41c3eac
URI	postgres://rpuyjxrvfvwhce:cdd78e49f402e7eed772f96b4b62f2740666674a1db55f12aae2284ab41c3eac@ec2-54-162-119-125.compute-1.amazonaws.com:5432/ddjee7v5j00c3p
Heroku CLI	heroku pg:psql postgresql-acute-31655 --app summerclass-name-generator

Reset Database

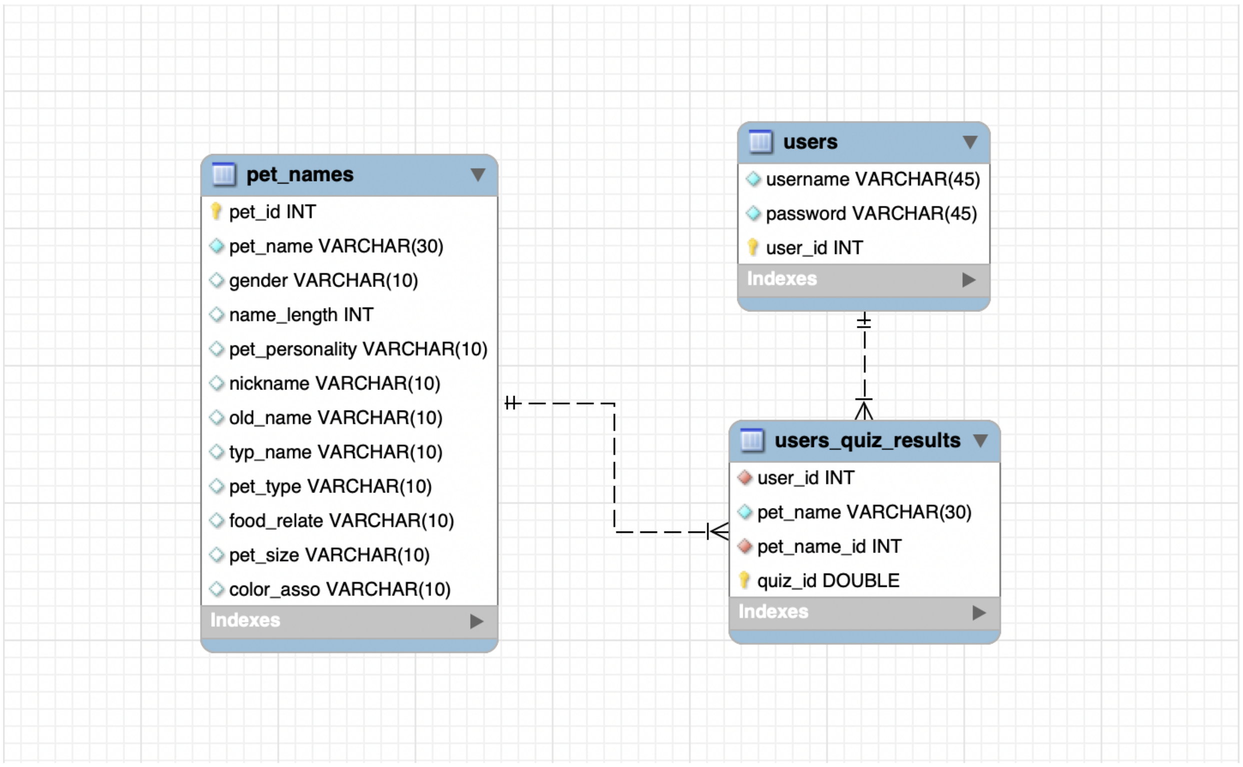
Reset the database to its originally-provisioned state, deleting all data inside it.

Destroy Database

Destroys the database and all of the data inside it.

Database Design:

Using PostgreSQL for databases stored on Heroku.



-Create table to store via Heroku server

```
DROP TABLE IF EXISTS pet_names;
CREATE TABLE IF NOT EXISTS pet_names (
  pet_id INT NOT NULL, /* server-side generated id to use as key /
  pet_name VARCHAR(30) NOT NULL,
  gender SMALLINT,   / 0- female, 1-male, 2-both /
  name_length INT,
  pet_personality VARCHAR(30), / crazy, lazy, sweet, smart /
  nickname VARCHAR(30), / 'nickname' OR NULL /
  old_name VARCHAR(30), / 'old' or 'new' /
  typ_name INT, / 'unique' or NULL /
  pet_type VARCHAR(30), / dog, cat, rodent, or bird /
  food_relate VARCHAR(30), / 'food' or NULL /
  pet_size VARCHAR(30), / small, medium, large /
  color_ass INT, / 'color' or NULL /
  PRIMARY KEY(pet_id, pet_name)
);



/ Insert Dog Values */
INSERT INTO pet_names(pet_id, pet_name, gender, name_length, pet_personality,
      nickname, old_name, typ_name, pet_type, food_relate, pet_size, color_ass)
VALUES(1,'Bernard', 1, 7, 'sweet', NULL, 'old', 'unique', 'dog', NULL, 'large', NULL),
VALUES(2,'Bernard', 1, 7, 'sweet', NULL, 'old', 'unique', 'cat', NULL, 'large', NULL),
VALUES(3,'Bernard', 1, 7, 'sweet', NULL, 'old', 'unique', 'rodent', NULL, 'medium', NULL),
VALUES(4,'Elizabeth', 0, 6, 'smart', 'nickname', 'old', NULL, 'dog', NULL, 'medium', NULL),
VALUES(5,'Bailey', 0, 6, 'sweet', NULL, 'new', NULL, 'dog', NULL, 'medium', NULL),
VALUES(6,'Tiger', 1, 5, 'smart', NULL, 'new', 'unique', 'cat', NULL, 'large', NULL),
VALUES(7,'Blue', 1, 4, 'lazy', NULL, 'old', 'unique', 'dog', NULL, 'large', 'color'),
VALUES(8,'Coco', 0, 4, 'crazy', NULL, 'new', NULL, 'dog', NULL, 'small', NULL),
VALUES(9,'Rocky', 1, 5, 'sweet', NULL, 'new', NULL, 'dog', NULL, 'medium', NULL),
VALUES(10,'Chunk', 1, 5, 'lazy', NULL, 'new', 'unique', 'dog', 'food', 'large', NULL),
VALUES(11,'Peanut', 1, 6, 'crazy', NULL, 'new', NULL, 'dog', 'food', 'small', NULL),
VALUES(12,'Arizona', 1, 7, 'smart', NULL, 'old', 'unique', 'dog', NULL, 'medium', NULL)
VALUES(13,'Binx', 1, 4, 'smart', NULL, 'new', 'unique', 'cat', NULL, 'medium', NULL);
```

-Pathway for SQL and JS queries

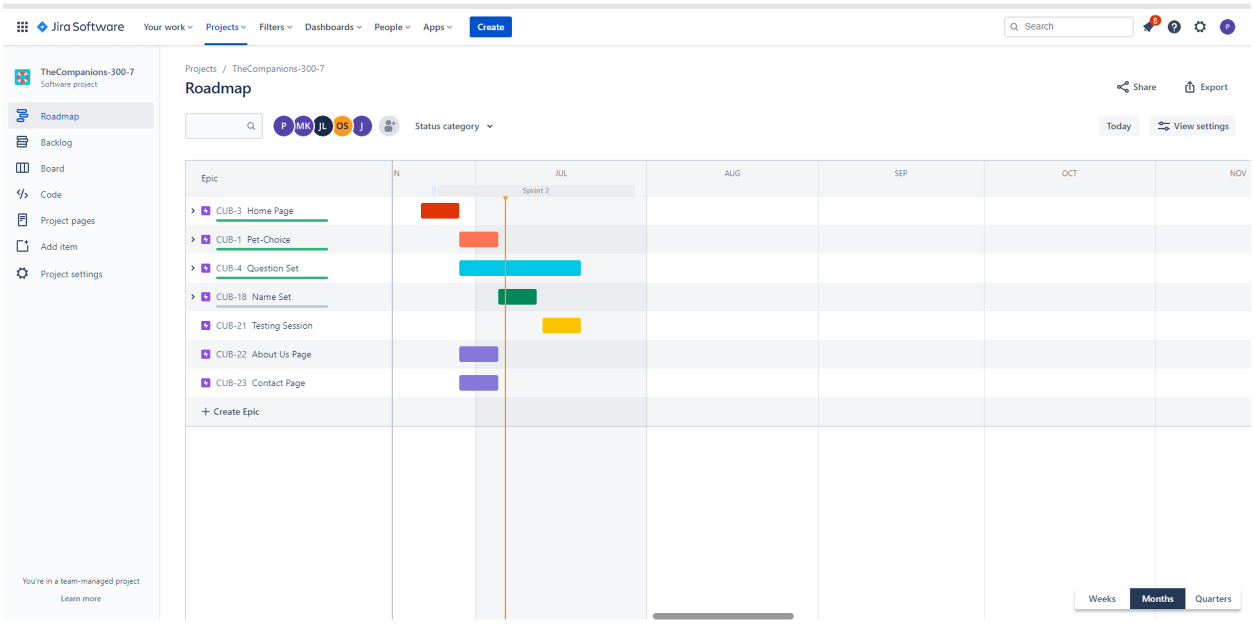
Create a user struct in JS with 10 correct_properties;
When question is answered, fill out and complete field in user struct;

Do after answering final question:

```
SELECT * WITH (correct_properties given the temp user struct);
Fill temp user array of strings with the SELECT;
generate random (int) via JS function, pass in array length for maxval();
Choose from temp user array via random int;
Send name to HTML to display on final screen;
```

Individual Contributions:

- Owen Smith:
    - Finished CSS for the homepage, did full homepage design and implementation
    - Scripts for question box mechanics
    - https://github.com/CSCI-3308-CU-Boulder/3308Summer21_300_7/commit/9cef66e9999a266ab1d8b9e7fc0e6494c60fbf10
- Jingqi Liu:
    - Clarified the scope of the test task, including the test environment, test methods, test cases, test tools.
    - Set up the software and hardware environment required for the test, including the operating system.
    - Finished the test plan: Outlines the testing strategy.
- Mathew Kim:
    - Developed the front end (HTML/CSS) for about_us.html and contact.html
    - Will further develop the two pages soon to finish it
    - Planning to make the contact page actually functional by utilizing a dummy email account to connect it to
    - https://github.com/CSCI-3308-CU-Boulder/3308Summer21_300_7/commit/69619cbac0e5eda48d990cf6a789db07143d9cd3
- Patrick Taylor:
    - Designed back end layout of project to be able to query names from a javascript function via main front end
    - Created a javascript function to query mock tables
    - Created SQL file to create tables
    - Created mock folder/files for names via GitHub
    - Created Heroku dataStore for pushing SQL data to the cloud
- John Hodgen:
    - Created back end tables formatting for SQL queries
    - Created temp names with properties to test table queries
    - Created Entity Relationship Diagram for website's database



Some Small Challenges:

-figuring out the pathways for server hosting took longer to finalize the plan than it most likely should have

-another challenge is getting work constantly updated in our team repositories, this can easily be fixed by us addressing to upload things when they are updated during in-person meetings

-another challenge has been to find decent work for every group member to complete, as we are locked to the online format we usually work alone, this can easily be resolved by having more discord or zoom time spent working together on the project instead of as individuals