SW Engineering CSC648/8482

DoggyDex Web Application
Section 01, Team 02

Grayson Dew - Team Lead
Orion Culbertson - Front-end Lead
Miguel Galvan - Git Master
Rigo Perez - Back-end Lead
Kristopher Phillips - Scrum Master

Milestone 2
10/21/2021

History Table:

Version	Date
V1	10/21/2021

1. Data Definitions V2

Name	Attributes	Usage
User	-LearningDoggyDex	This is a parent class of RegisteredUser and GuestUser that will encapsulate some attributes and behavior.
RegisteredUser	Parent Class: User -Name -Username -Password -Email -ProfilePicture -Posts -PersonalDoggyDex -Score	This user will start with an empty DoggyDex (empty Found Dog Breeds) and will be able to fill the DoggyDex by taking pictures of dogs they find. They will also be able to make posts to the Lost Dog Community Board.
GuestUser	Parent Class: User	This user will be able to view the DoggyDex in Learning Mode, which means that all dogs are unlocked, and will be able to upload photos of dogs to be identified by our system.
DogBreed	-Breed Name -Photo -About the breed/history -Behavior -Height -Weight -Life Expectancy -Personality	This will be the structure that stores information on a specific dog breed.
<<DogOfDay>>	-DogOfDay	This is a static object that contains the featured dog breed of the day. This will be used to grant points to the user if they take certain actions.
DoggyDexPhotoInterpeter	-UploadedPhoto -IdentifiedDogBreed	This class serves as a facilitator between a user and the ML model.
DogBreedClassifierModel		This class represents the ML model that will actually identify the dog breed.
UploadedPhoto	-Photo	This class will contain the photo a user uploads to our service so that it can be manipulated as needed.

DoggyDex	-DoggyDex (DogBreed[]) -NumBreedsFound -Encounters	This class will represent a DoggyDex. For each User, there will be a learningDoggyDex that is completely filled out that will allow the user to browse all supported breeds. Registered users will also have a personalDoggyDex that will contain an array of all dog breeds they have found so far. This will store all of the statistics of a particular user in regards to different types of dogs. This will show how many times a user has come across a specific type of dog, the dates discovered of breeds, as well as other things.
Post	-Timestamp -Title -BodyText -Media -Location	This class will house all the information that is related to a post on any of the community boards. A user can put anything they want in this post that they want other users to see.
LostDogPost	Parent Class: Post -IsFound	This is a special type of post for lost dogs that will allow us to display
DoggyDate	Parent Class: Post -Date -ProposedTime	This is a special type of post for advertising DoggyDates.
<<Scores>>		This will be a static object that houses all the point values of specific actions. This will ease the maintenance of code so that we can edit scores as needed in just one place.
SessionCookie		Cookie that is used to maintain a user's session as they are traversing across pages. This will only be necessary for a RegisteredUser
ViewModeCookie		This cookie will store whether the user is in "Learning Mode" where they can see all dog breeds or if they are viewing their own, personal DoggyDex with their captured breeds.

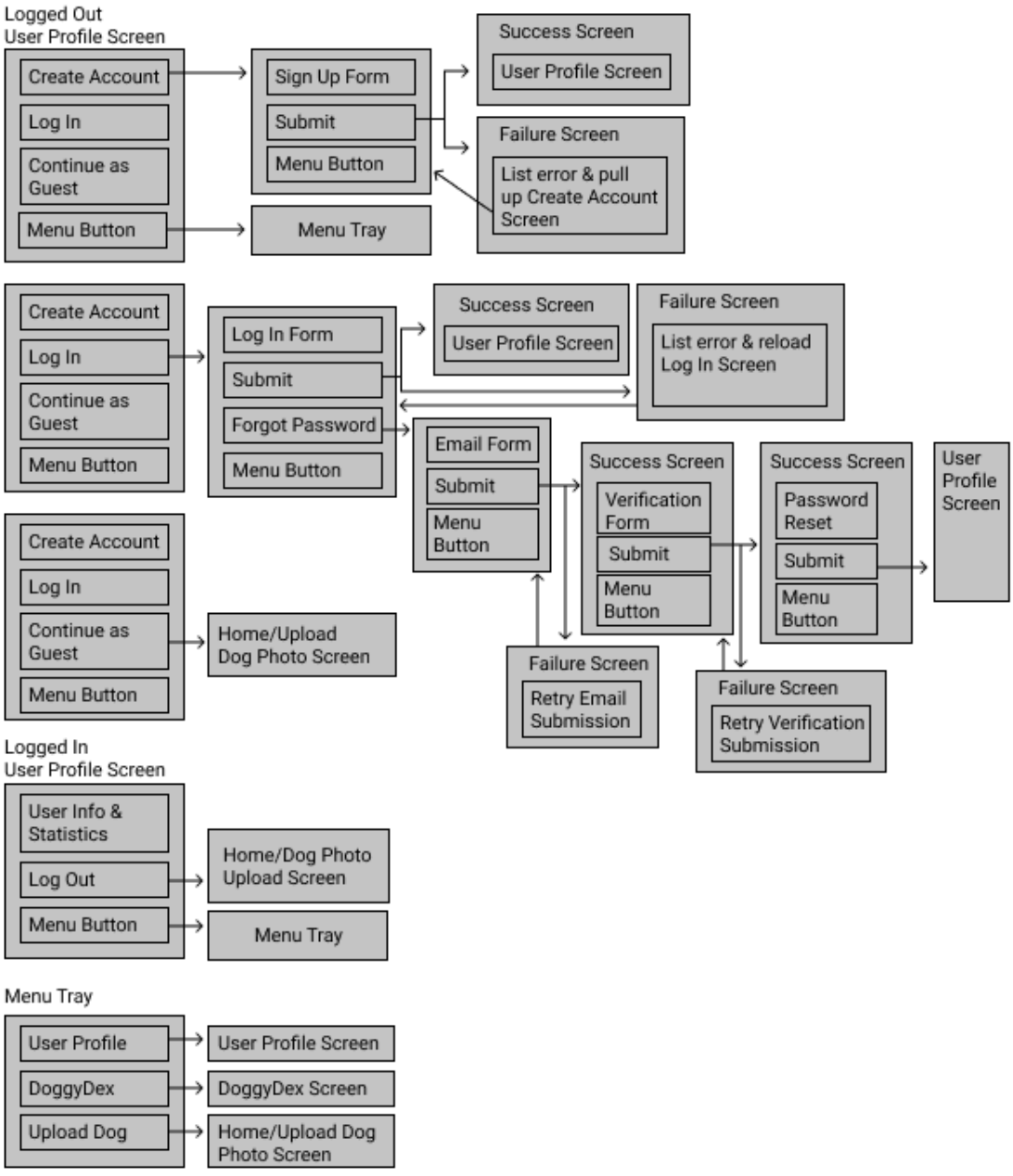
2. Functional Requirements V2

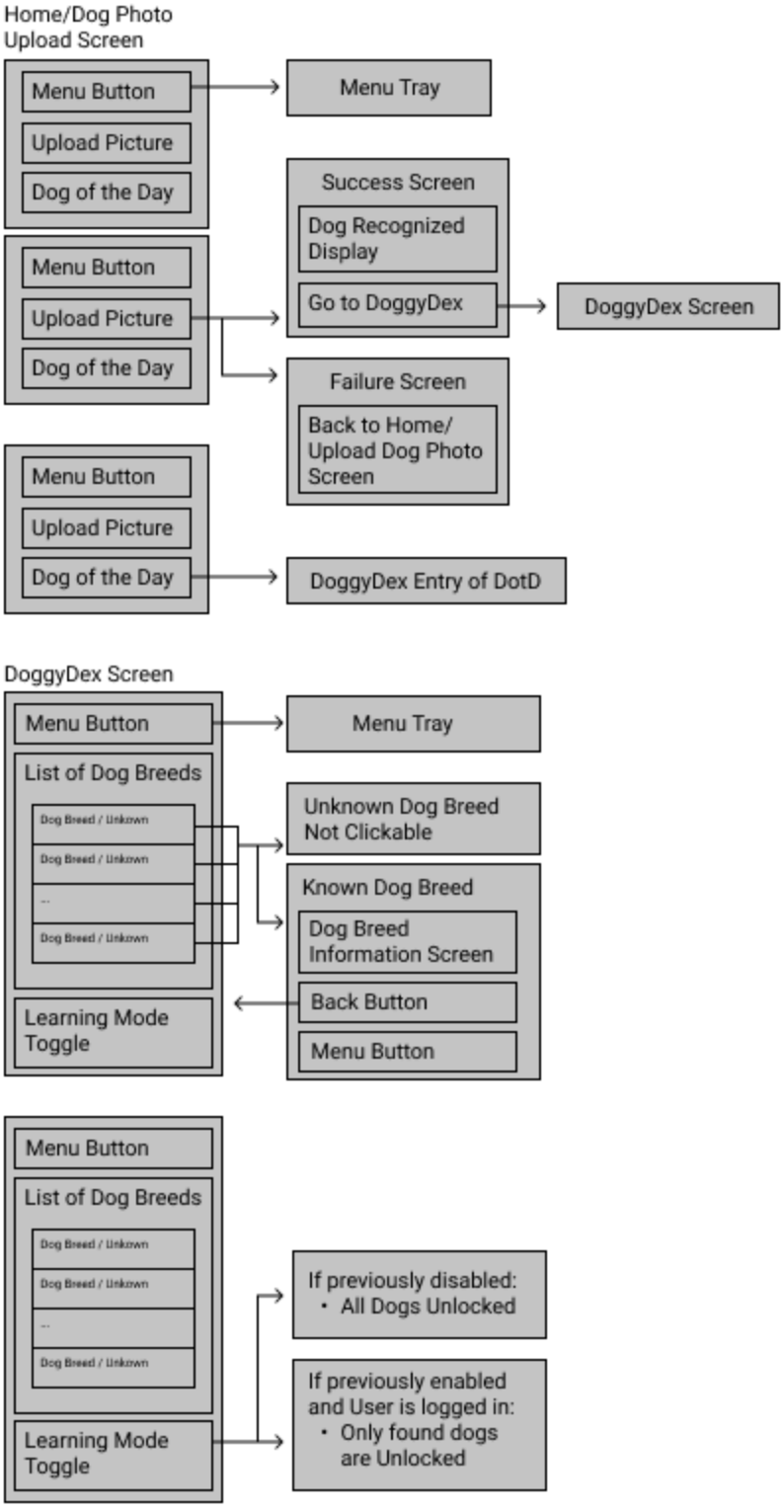
- **User Accounts (1)**
  The application needs functionality to store a variety of data that is specific to each user, such as their own, personal DoggyDex. This functionality will be achieved by requiring users to create an account in order to access that functionality.
    - **Create Account (1)**
      When a user is presented with the login screen, there will need to be an option for them to create an account if they do not already have one. Once created, their username and password (salted and hashed) will be stored in a database so that their identity can be verified after they leave their current user session.
    - **Log In (1)**
      Users will need to log in to gain access to their own DoggyDex, participate in the DoggyDates matching program, or post on the Community Board.
    - **Forgot Password (2)**
      If a user forgets their password, we will need to implement a way for them to recover their account through the website. We will verify their identity by sending them an email that will contain a password reset link for them to follow.
- **Upload Photo (1)**
  The main feature of the application is the classification of photos the user provides. The user will upload the photo of a dog from their device.
    - **Dog Breed gets Classified (1)**
      After users upload a photo of a dog, they will receive back an output of what dog breed our model thinks the dog in their photo belongs to along with a rating of how confident the model is of its prediction. If there is no dog in the photo, the output will warn the user that there is no dog detected. There will be an option for users to still request a classification of dog breed if there is no dog present after this warning.
- **DoggyDex**
    - **Personal DoggyDex (1)**
      This viewing mode of the DoggyDex will only display the dog breeds that a user has already captured in an uploaded photo. Each dog breed that a user has previously captured can be selected by the user to then view a fact sheet that will show relevant information about that particular breed. All other entries will display a question mark and have a blank breed fact sheet.
    - **“Learning Mode” DoggyDex (1)**
      This viewing mode will show a user all the dog breeds that are supported on the platform as well as their fact sheets.

    - **Scoring System (1)**
        - Points for:
            - New dog breed discovered
            - Take picture of dog
            - Take picture of Dog of the Day
            - Open Dog of the Day
    - **Search dog breeds (2)**
      This functionality will allow a user to do a text search of all dog breeds that are currently viewable in whatever viewing mode the user is in.
- **Dog of the Day (2)**
  The goal of this feature is to provide a sense of discovery for the user where they are shown a randomly generated breed of dog that is not already contained within their DoggyDex. For users that are not logged in, the Dog of the Day will be randomly generated from all dog breeds the platform supports.
- **Lost Dog Community Board (2)**
- This feature will be a place for users to post information for other users of the platform to see. In order to post something, a user will be required to be signed in.
- **DoggyDates (3)**
  This feature will allow users that opt into this service to connect with other dog owners and set up playdates for their dogs. Dog owners will be required to advertise what kind of dog they own and broadcast this information to other users. Owners will then input preferences for what kinds of dogs they want their own dog to interact with. The interface for this feature will be akin to a dating app where users can swipe left and right among options for dogs to set up a playdate with.
- **Map features**
  These features will allow the user to input a zip code and receive back a variety of relevant data. The user will be able specify a distance radius they want to look in to show these data.
    - **Dog Park Finder (3)**
      This feature will display all the dog parks within the specified area.
    - **Pet Shelter Finder (3)**
      This feature will display all the pet shelters within the specified area. With the option for pet shelters to supply their own data, this feature can also show what pets are up for adoption at each shown shelter.
    - **Dog-Friendly Areas (3)**
      This feature will display all the restaurants, coffee shops, apartments, and malls (with potential other types of areas) that are dog-friendly.

3. UI Mockups and Storyboards (high level only)

- UX Storyboards for:
    - User Accounts
        - Create Account
        - Log In
        - Forgot Password
    - Upload Dog Photo
    - DoggyDex
        - Personal DoggyDex
        - Learning Mode DoggyDex
        - Scoring System (This will be a part of the User Info & Statistics)
    - Dog of the Day




Home/Dog Photo Upload Screen
Menu Button
Upload Picture
Dog of the Day
Menu Tray
Success Screen
Dog Recognized Display
Go to DoggyDex
DoggyDex Screen
Failure Screen
Back to Home/ Upload Dog Photo Screen
DoggyDex Entry of DotD
DoggyDex Screen
List of Dog Breeds
Dog Breed / Unkown
Learning Mode Toggle
Unknown Dog Breed Not Clickable
Known Dog Breed
Dog Breed Information Screen
Back Button
If previously disabled:
• All Dogs Unlocked
If previously enabled and User is logged in:
• Only found dogs are Unlocked

UX Validation Meeting

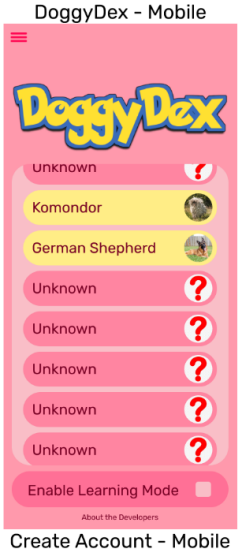
- Useful
    - We're solving the problem of not knowing a dog breed and struggling to figure out what it is.
- Usable
    - The nav bar at the top helps users navigate anywhere they'd like to go with use. Where you are is highlighted on the nav bar so it's easy to know. You are informed whether or not you're in Discovery Mode or Learning Mode on the DoggyDex. We can hope that our model is good enough to recognize dogs well.
- Findable
    - Everything is readable and what functions a user might expect/want are there.
- Credible
    - We'll show our confidence level/percent in our Dog identification, and display a privacy policy stating that we don't store their photos or share personal information.
- Desirable
    - It's a fun gamified way to learn about dogs. Everyone likes dogs and we're going to be a web-app full of dog pictures. The points system will encourage people to keep on taking pictures of dogs and encourage continuous use.
- Accessible
    - Everything is large and readable, which adds itself to ease of use. We're going to make sure everything is encoded properly to ensure users using accessibility software will be able to use our website.

GUI Mockup

Home Screen - Mobile

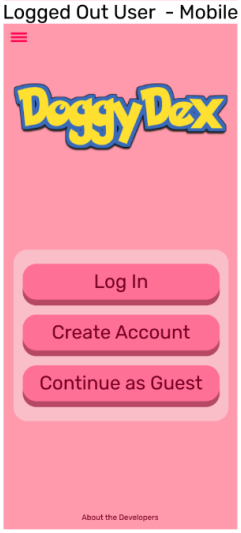
Dog Recognized - Mobile

DoggyDex - Mobile

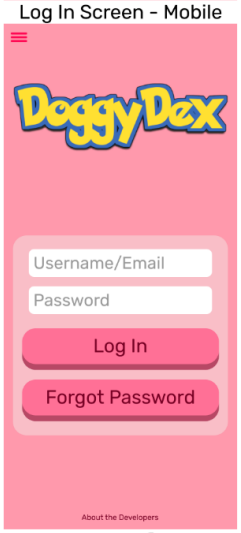


DoggyDex Clicked - Mobile

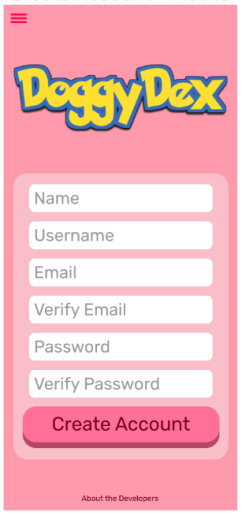
Logged Out User - Mobile



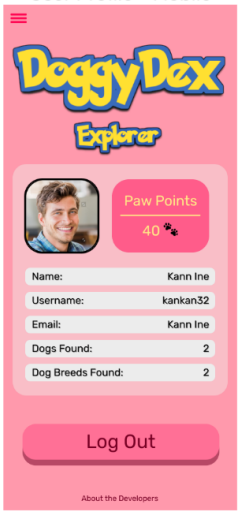
Log In Screen - Mobile



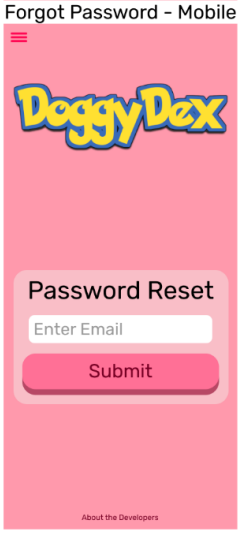
Create Account - Mobile



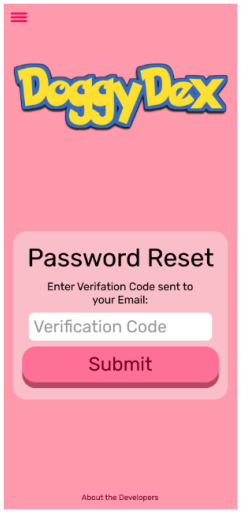
User Profile - Mobile



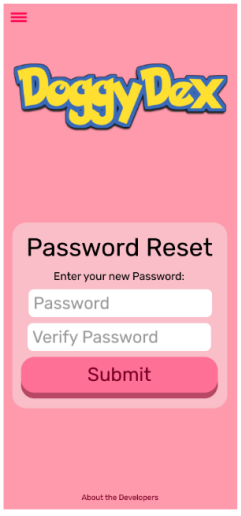
Forgot Password - Mobile



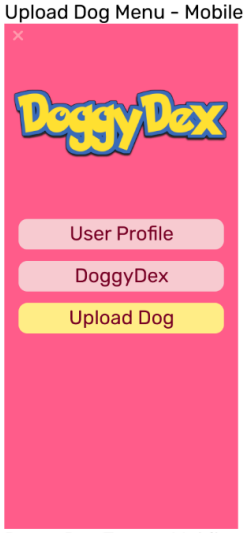
Forgot Password Verification - Mobile



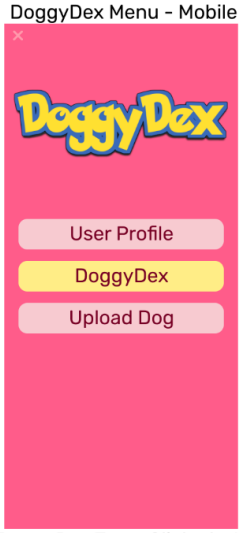
Reset Password - Mobile



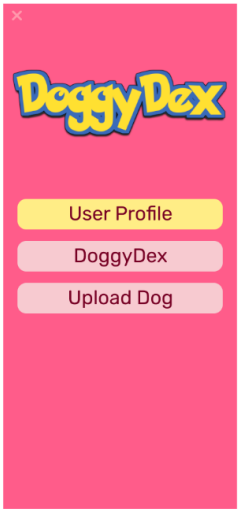
Upload Dog Menu - Mobile



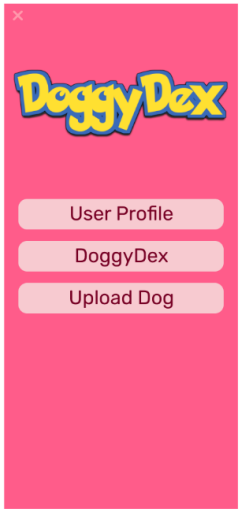
DoggyDex Menu - Mobile



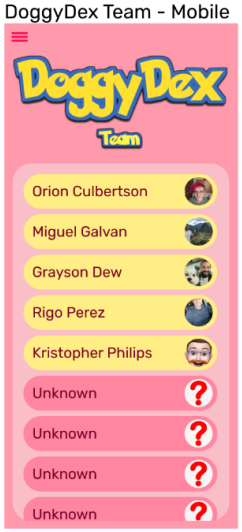
User Profile Menu - Mobile



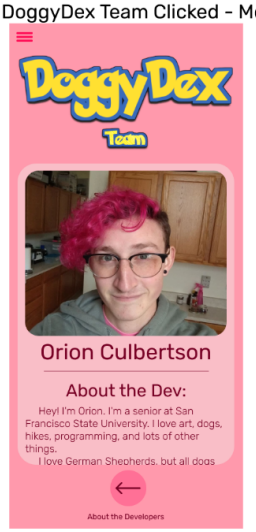
General Menu - Mobile



DoggyDex Team - Mobile



DoggyDex Team Clicked - Mobile



4. High Level Architecture, Database Organization

User Schema

- Name (req)
- Username (req)
- UserID (req) (to key tables with)
- Password (req)
- Email (req)
- ProfilePicture (optional)
- Posts (optional)
- PersonalDoggyDex (req)
- Score (req)

Dog Schema

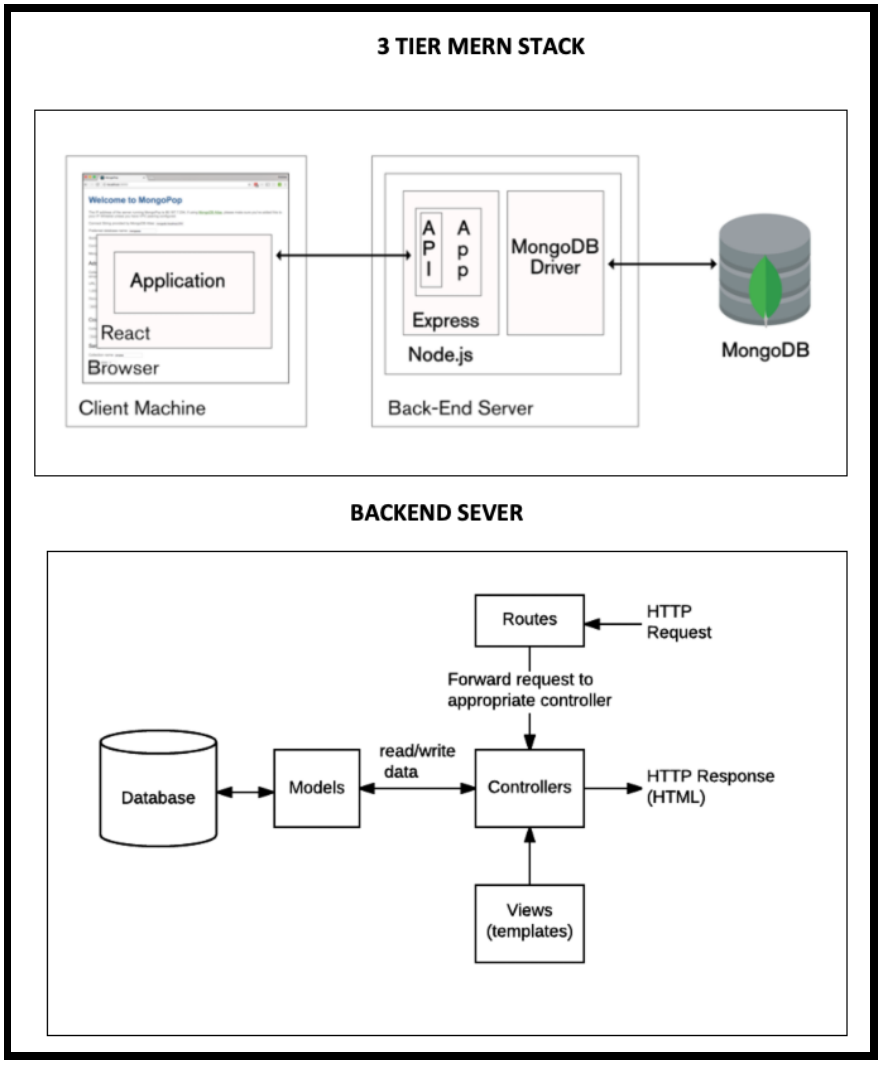
- BreedName (req)
- Photo (link) (req)
- AboutBreed (req)
- Height (req)
- Weight (req)
- LifeExpectancy (req)

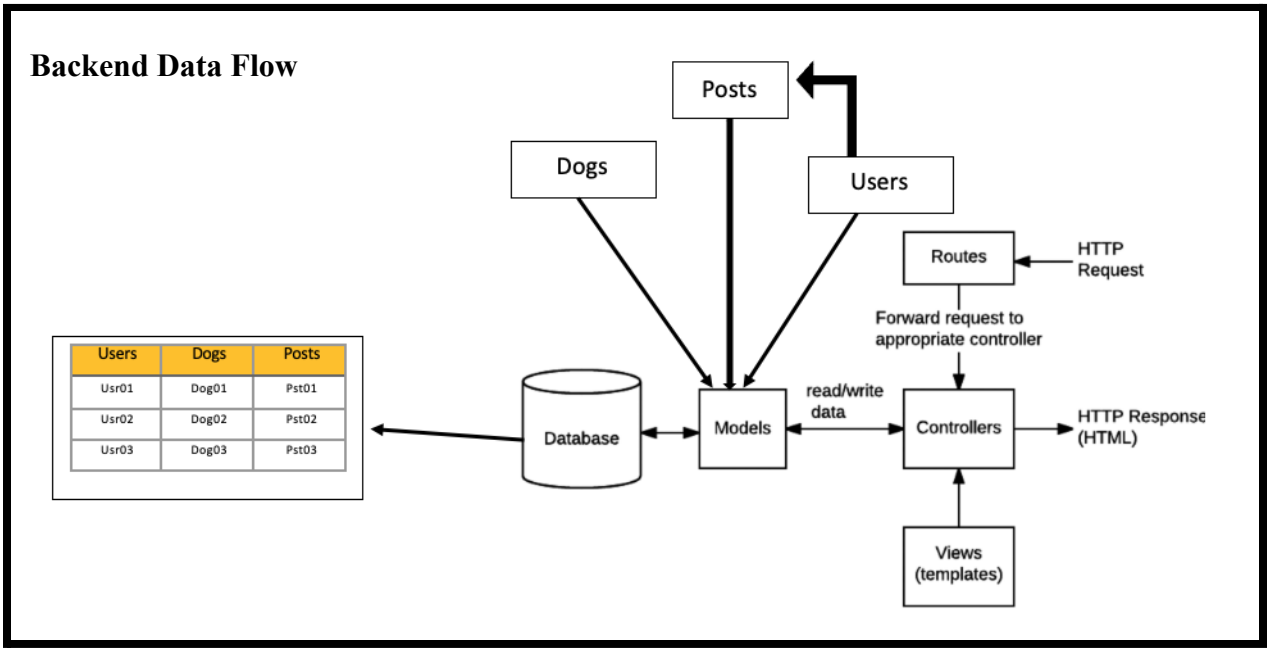
Post Schema

- Timestamp (req)
- Title (req)
- BodyText (req)
- Media (optional)
- Location (optional)

Add/Delete/Search Architecture

Users	**Add/Delete**	User registration and delete account
DoggyDex	**Add/Search/Displayed**	Each user gets an instance of this
Posts	**Add/Delete/Search/Displayed**	Register users can post
Upload Photos	**Add/Displayed**	Uploaded photos will go through dog recognition and and saved in DoggyDex


3 TIER MERN STACK
Application
React
Browser
Client Machine
API
App
Express
Node.js
MongoDB Driver
Back-End Server
MongoDB
BACKEND SEVER
Routes
HTTP Request
Forward request to appropriate controller
Database
Models
read/write data
Controllers
HTTP Response (HTML)
Views (templates)


Backend Data Flow
Posts
Dogs
Users
Routes
HTTP Request
Forward request to appropriate controller
Users
Dogs
Posts
Usr01
Dog01
Pst01
Usr02
Dog02
Pst02
Usr03
Dog03
Pst03
Database
Models
read/write data
Controllers
HTTP Response (HTML)
Views (templates)

Technical Feasibility of Database Operations

User_Table Is a collection of all users that are registered in the DoggyDex app. We will need to **add** users into our database when they register for an account. We will also need to retrieve user information from the database when they are signed in so that we can display information that is user-specific like their profile picture and username. In addition, a user will be allowed to **update** his personal info and credentials. Lastly, a user will be allowed to **delete** his account. This table will also store user-specific information about their DoggyDex. It will store information Dog_Breed_Found (bool), User_Photo_of_Breed, Times_Breed_Encountered. This will need to be both updated and accessed throughout the user's session based on their actions.

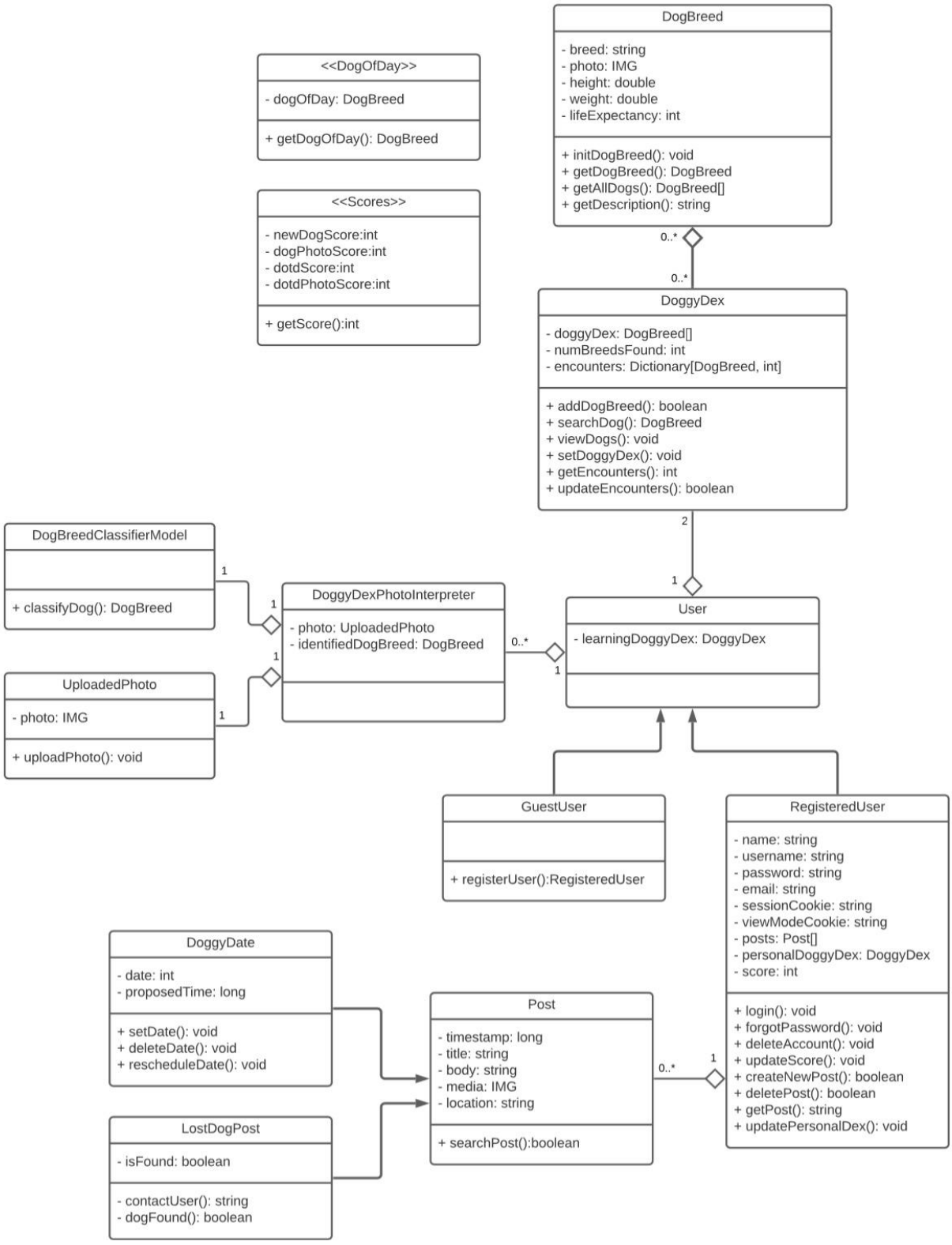
Dog_Breed_Table will be static and will serve as the master list of dog breeds we support. We will only be performing "get" operations on this data.

Post_Table will store all the information users input when creating posts on our Community Boards. This will store a photo, location (zip code), username of the poster, and message body.

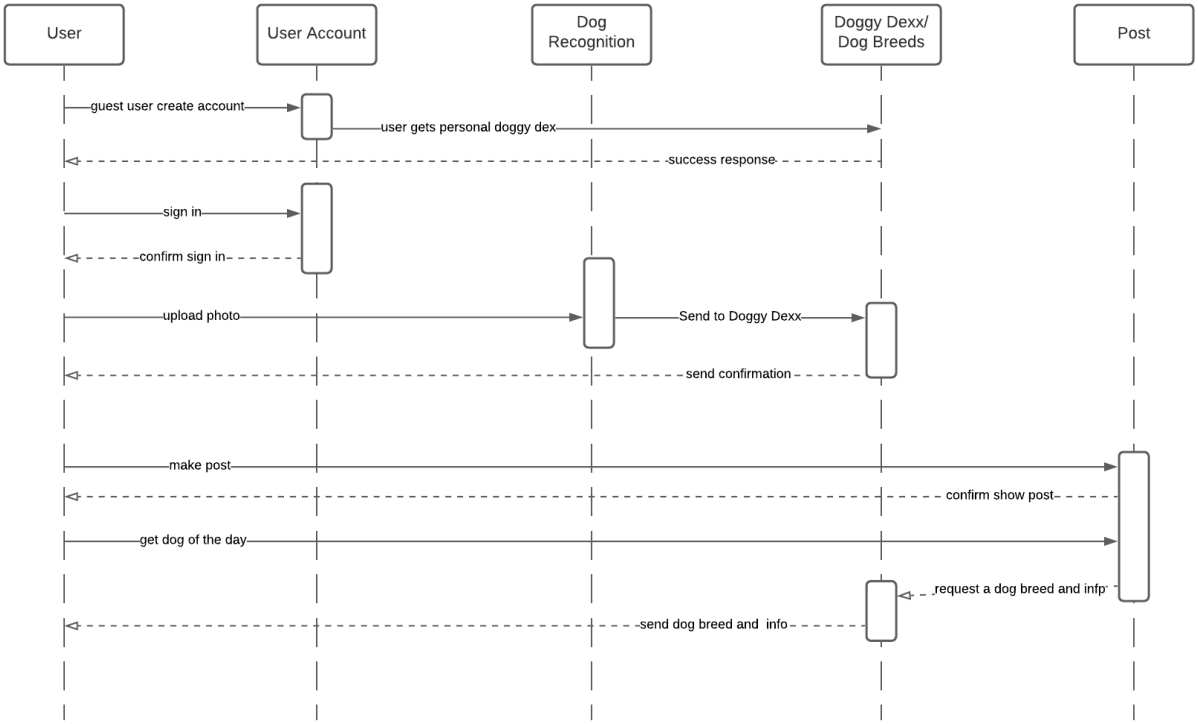
APIs needed for project:

The Dog API for scalable dog breed information: https://docs.thedogapi.com
TensorFlow for image classification: https://www.tensorflow.org/api_docs/python/tf

5. High-Level UML Diagrams



Sequence Diagram



6. Key Risks

a. We have a couple of risks regarding sourcing photos for the dog breed classification model. One is being able to find enough photos of different dog breeds, and one is making sure we use photos we have a right to.
    - We're planning to solve this by mainly using the Stanford Dogs Dataset, which has data on 120 breeds of dogs. This doesn't take care of all the dog breeds, but it might be good enough for our purposes and for getting a start developing a model.

b. We have a legal risk of infringing on the Pokemon brand.
    - We plan to mitigate this by not going overboard on direct references.

c. One risk is that our image model may not be good enough to accurately predict the breed of a dog in a given picture.
    - We hope to learn enough about building an image classification model as well as source enough photos for it to be working fairly well for the dog breeds we can source images of.

d. We also have a budget risk. There's a risk of exceeding the data allowance of our servers, and some APIs that would be really helpful for this project cost money.
    - We plan to overcome the server costs by not overdoing the data we start out with and paying attention to our hosting restrictions. Even though we don't have access to the paid APIs, we hope to build a good enough image classification model to be able to not need these.

7. Project Management

Aside from the classroom sessions, our group has been holding weekly Tuesday Discord meetings to further progress in our project. Both instances of meetings are efficiently used to review what we have covered so far, and to coordinate what short term goals should be achieved by the next meeting, and long term goals for the following week. At the beginning of this milestone, our team had a scrum meeting where we delegated tasks by section number. We tried to be flexible with our expectations of what tasks needed to be completed first and what tasks needed to be done as a team, and we reasonably met this goal. Our team was generally split into a front-end group of three developers and a back-end group of two developers. One back-end developer was responsible for defining the data that will be needed and establishing schemas for said data in our database while the other was tasked with implementing the database in MongoDB. This second developer also created a working demo with a template front-end to simply allow us to add or remove things from the database and have them appear on our site. Next, one front-end developer created the UX storyboard and GUI mockup using Figma, while another created the UML diagram along with input from the entire team. The third developer took the functional demo from our back-end developer and implemented our designed GUI using React.

To communicate all these tasks out, we utilized a group task manager called Asana to show everyone what tasks were assigned to whom and when intermediate deadlines for specific tasks were. As we progressed through this milestone, we found that we were using the meetings to clarify understanding of the entire project and how each of our individual work fit into the whole. This would then cause us to circle back to topics that needed more ironing out upon collective reflection. Despite being plagued by unforeseen circumstances during this milestone, our team's ability to plan and coordinate has allowed us to stay on track and meet our deadlines. Our ability to coordinate and delegate has proven to be a key element of our success as a team. Each team member has a focus within the project, but they are not limiting themselves to that one area. We are all here to learn, so being acquainted with every part of the project is important to our individual success as well. We are fortunate to have figured out an effective means of working from the start, and will continue to utilize these methods to realize project completion.