

1. Application name: Dog encyclopedia.

2. Problem area:

The website's primary serves educational, informational, academic, entertainment and charitable purposes.

It can be appropriate for:

- pet stores,
- people that already are or are potentially interested in becoming dog owners and/or breeders,
- pet competitions organizers,
- educational establishments (especially schools),
- other developers,
- people, who like dogs,
- charities for further cooperation.

3. Project goals:

The system helps to generalize information about different breeds of dogs, including their health, behaviour, and lifestyle. It simultaneously helps to spread awareness about homeless animals and find them new owners.

4. System responsibilities:

The system stores information about breeds and adoption options, as well as user data (personal information and favourite breeds). It allows communication between user and shelter. The system should be managed by administrators.

5. System users:

- website administrator,
- user,
- registered user.

6. Functionalities:

System allows unregistered users:

- registration,
- search and sort breeds via different criteria.

System allows users search (as unregistered users) and:

- login and profile modification,
- adding favourite breeds to the profile,

- filling forms on the adoption page.

System allows (administrators):

- managing (adding, updating, deleting) breed data,
- managing shelter data.

7. User requirements:

Entities:

Dog breed entity represents class Breed and presents general information about the dog (its brief description, physical and visual characteristics).

Dog breed:

- Id,
- Name (e.g. German Shepherd, Bulldog),
- Description (general description),
- Origin (country of origin),
- Lifespan (expected life duration),
- Weight (average weight for adults),
- Height (average height for adults),
- Colour (possible fur colours, foreign key from Colour entity),
- Size (e.g. small, medium, large),
- Dog type (foreign key from Dog type entity),
- Appearance (photo link).

Dog type entity represents main features, characteristic of certain larger dog groups, that can be same between a couple of breeds.

Dog type:

- Id,
- Dog group (e.g. terrier, herding, hound etc.),
- Purpose (if group is working (hunting, protection etc.)),
- Family friendly (if are recommended for families with children by experts or not).

Colour entity represents fur colours, most popular among certain breeds. RGB codes are needed for further visualization of the on the website.

Colour:

- Id,
- Code (RBG colour code),
- Name.

Adoption option

- Id,
- Name (dog name),

- Sex (male/female),
- Age (if known),
- Breed (if known, foreign key from Breed entity)
- Weight (in kgs),
- Description (general description of the dog, its lifestyle and character),
- Appearance (link to the photo if available),
- Availability (boolean).

Relationships between tables:

- Dog type and Dog breed: one-to-many,
- Dog breed and Colour: many-to-many,
- Breed and Adoption: one-to-many.

Functionalities:

Managing dog types:

- Add, update (as well as partially), and delete dog types.
- Retrieve full information about available types.

Managing dog breeds:

- Add, update (or partially update), and delete dog breeds.
- Retrieve full information about a breed. Sort retrieved information according to all parameters. Retrieve appearance as image, located by the link and colours as colours for elements on the screen for proper visual display.
- Assign multiple fur colours to a breed.

Managing colours:

- Add, update (and partially update), and delete colours.
- Retrieve information about existing colours.

Manage Adoption Options:

- Add, update, and delete adoption options.
- Retrieve detailed information about dogs available for adoption (appearance should be an image located by the address specified by the field).
- Update availability status (if dog has already been adopted).

Managing user account:

- Add breeds to favourites.
- Fill adoption forms.

Class methods

TypeService class:

- AddType(): Method to add a new dog type.
- UpdateType(): Method to update an existing dog type.

- DeleteType(): Method to delete a dog type.
- GetType(): Method to retrieve information about specific dog type.
- GetAllTypes(): Method to retrieve information about all types.

BreedService Class:

- AddBreed(): Method to add a new dog breed.
- UpdateBreed(): Method to update an existing dog breed.
- DeleteBreed(): Method to delete a dog breed.
- GetBreed(): Method to retrieve details of a specific dog breed.
- GetAllBreeds(): Method to retrieve information about all breeds.

ColourService Class:

- AddColour(): Method to add a new colour.
- UpdateColour(): Method to update an existing colour.
- DeleteColour(): Method to delete a colour.
- GetColour(): Method to retrieve information about a specific colour.
- GetAllColours(): Method to retrieve information about all colours.

AdoptionService Class:

- AddAdoption(): Method to add a new adoption option.
- UpdateAdoption(): Method to update an existing adoption option.
- DeleteAdoption(): Method to delete an adoption option.
- GetAdoption(): Method to retrieve information about a specific adoption option.
- GetAllAdoptions(): Method to retrieve information about all adoption options.
- Adopt(): Method to mark a dog as adopted (update its availability status).

UserService:

- Login(): Give user access to one's profile.
- Register(): Create account for new user.
- AddFavourite(): Add breed to list of favourites.
- SendAdoptionForm(): Send adoption form with your information.

Actor inheritance

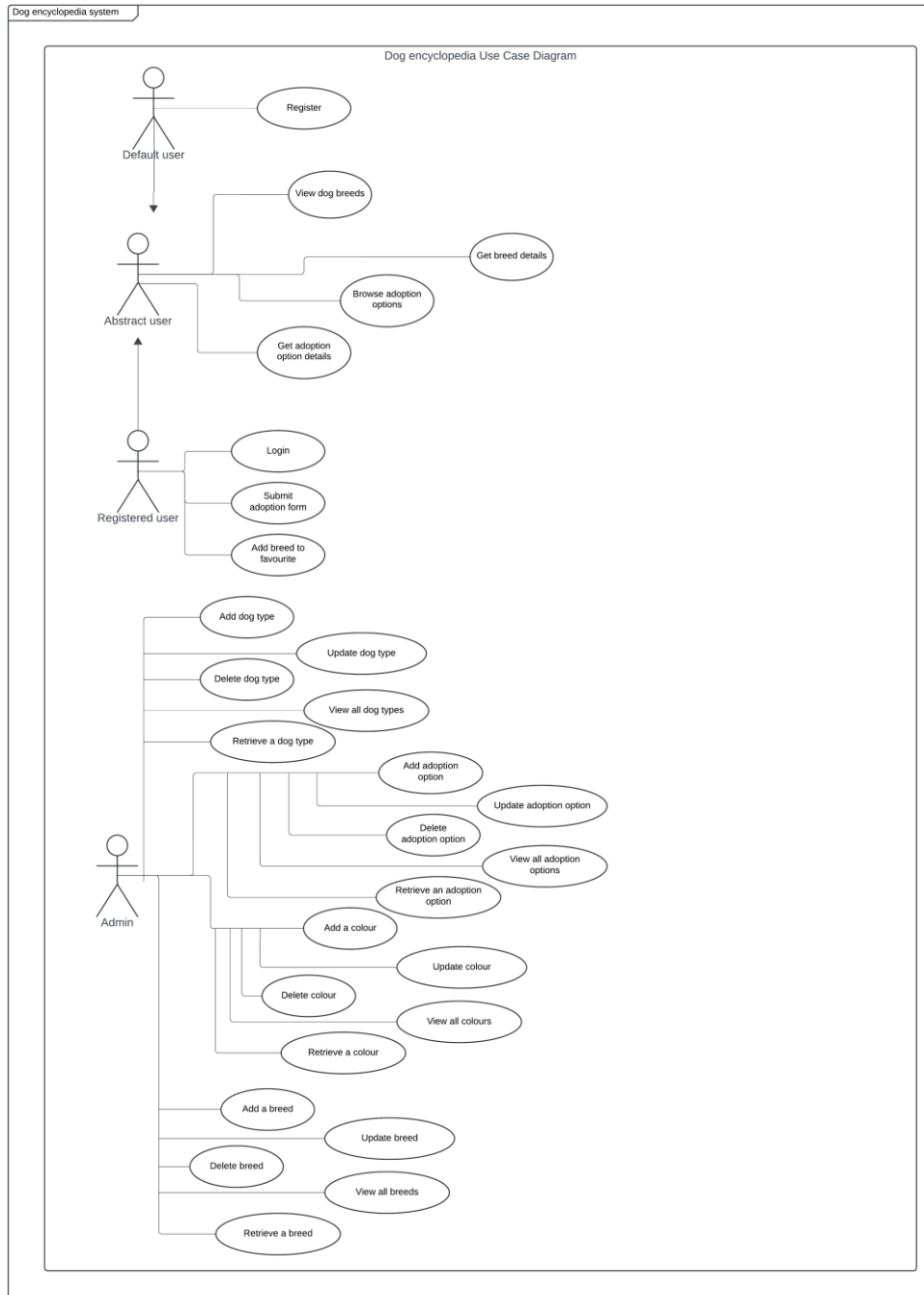
Registered users have same access as usual users, as well as are able to fill out adoption forms and add breeds to favourites.

Constraints:

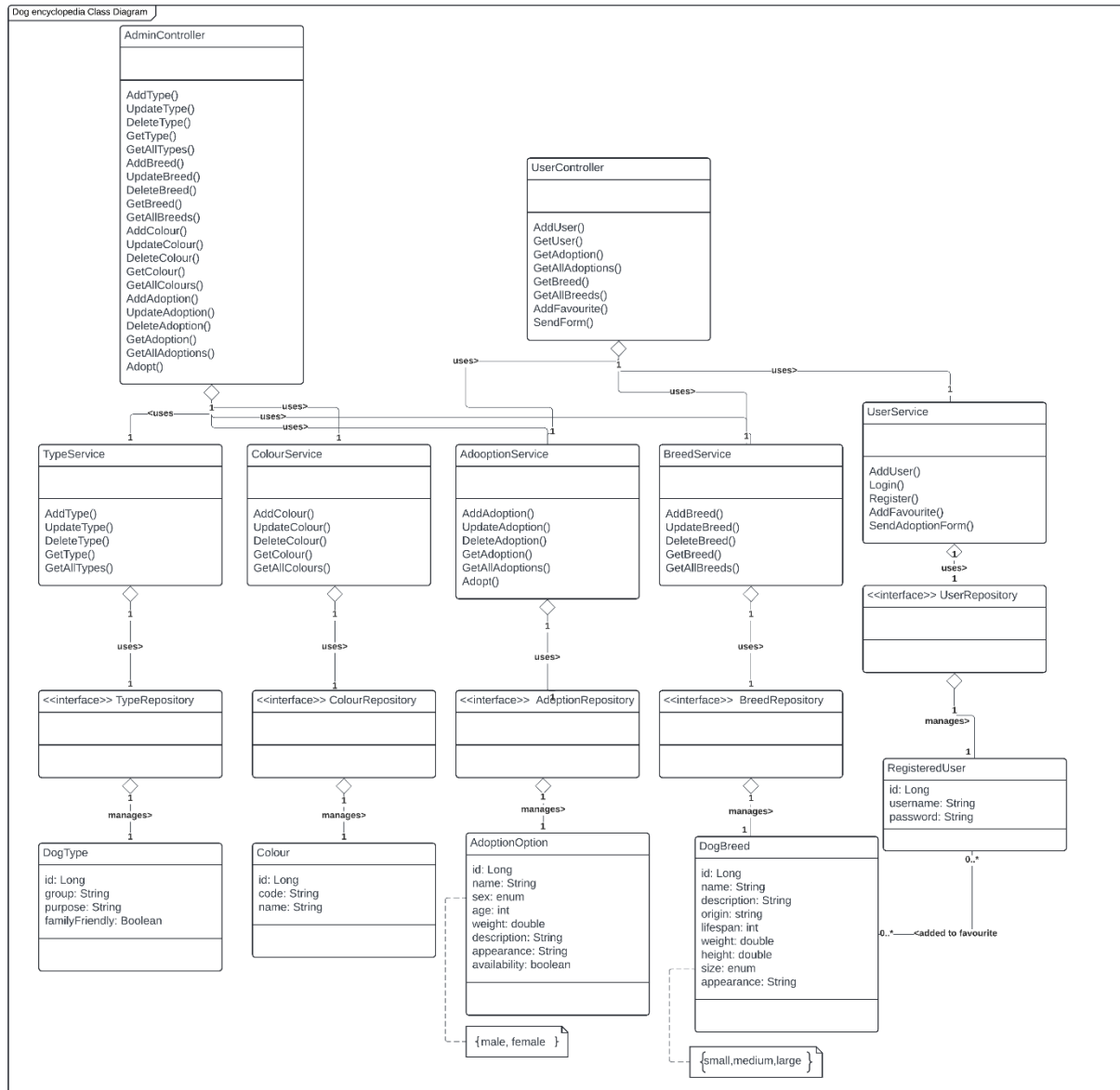
- System should be based on Java with Spring Framework.
- All dependencies should be managed via Gradle.
- System must use H2 database for managing all entities.
- System should perform without failure and handle incorrect interactions properly.

- System must use secure authentication and authorization mechanisms.
- UI should be user-friendly and intuitive with simple menu.
- All essential and most frequently used features should be always displayed on the user's screen.
- System should perform consistently across different devices and operating systems.

8. Functional requirements: Use case diagram



9. Description of the system structure: Class diagram



10. Non-functional requirements:

- System should be based on Java with Spring Framework.

Metric: Adherence to Java and Spring Framework version compatibility rules.

- All dependencies should be managed via Gradle.

Metric: Successful Gradle build.

- System must use H2 database for managing all entities.

Metric: Successful establishment of connection and data management using H2 database.

- System should perform without failure and handle different incorrect interactions properly.

Metric: Track error rate. Handle different scenarios for user interactions via testing and user feedback.

- System must use secure authentication and authorization mechanisms.

Metric: Compliance with standard authentication and authorization protocols.

- UI should be user-friendly and intuitive with a simple menu.

Metric: User reviews and satisfaction surveys, as well as testing of interface usability and accessibility.

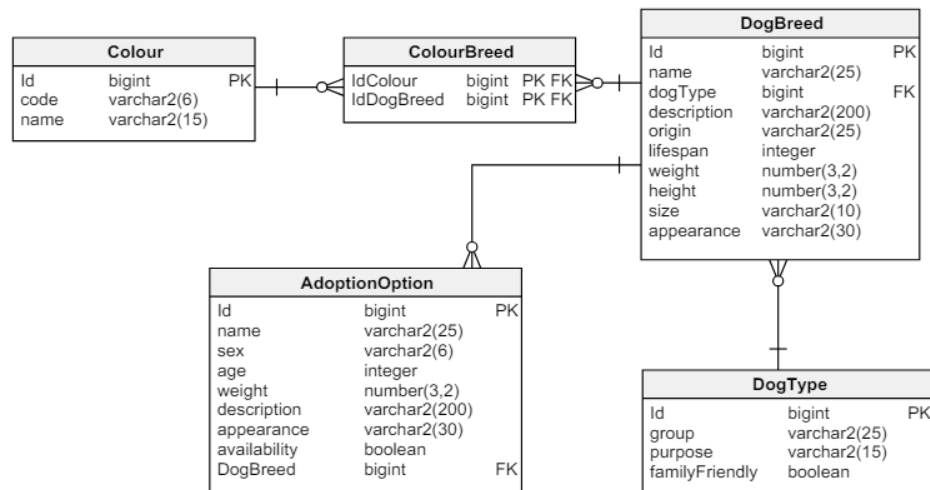
- All essential and most frequently used features should always be displayed on the user's screen.

Metric: Perform user testing to identify vital features and ensure their visibility on the screen.

- System should perform consistently across different devices and operating systems.

Metric: Measure response time and design compatibility on different devices and operating systems.

11. Database schema



12. API endpoints

UserController:

1. POST /users: Add a new user.
2. GET /users/{id}: Get user by ID.

3. GET /breeds/{id}: Get dog breed by ID.
4. GET /breeds: Get all dog breeds.
5. GET /adoptions/{id}: Get adoption option by ID.
6. GET /adoptions: Get all adoption options.
7. POST /users/{userId}/favourites: Add a new favourite breed to the user's profile.
8. POST /adoptionforms: Submit a new adoption form (personal information is in the body).

AdminController:

1. POST /admin/types: Add a new dog type.
2. PUT /admin/types/{id}: Update an existing dog type.
3. DELETE /admin/types/{id}: Delete a dog type.
4. GET /admin/types/{id}: Get dog type by ID.
5. GET /admin/types: Get all dog types.
6. POST /admin/colours: Add a new colour.
7. PUT /admin/colours/{id}: Update an existing colour.
8. DELETE /admin/colours/{id}: Delete a colour.
9. GET /admin/colours/{id}: Get colour by ID.
10. GET /admin/colours: Get all colours.
11. POST /admin/breeds: Add a new dog breed.
12. PATCH /admin/breeds/{id}: Update an existing dog breed.
13. DELETE /admin/breeds/{id}: Delete a dog breed.
14. GET /admin/breeds/{id}: Get dog breed by ID.
15. GET /admin/breeds: Get all dog breeds.
16. POST /admin/adoptions: Add a new adoption option.
17. PATCH /admin/adoptions/{id}: Update an existing adoption option (includes adopting a dog).
18. DELETE /admin/adoptions/{id}: Delete an adoption option.
19. GET /admin/adoptions/{id}: Get adoption option by ID.
20. GET /admin/adoptions: Get all adoption options.

13. List of used technologies

Java with Spring Framework and Gradle – main programming language, data management, web application core.

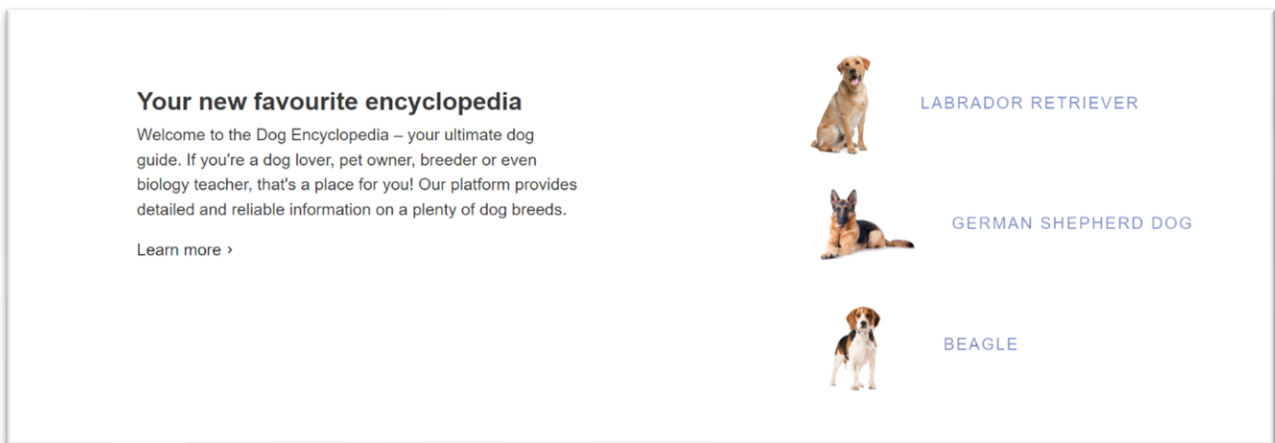
HTML – application page structure and content display.

CSS – web pages stylization.

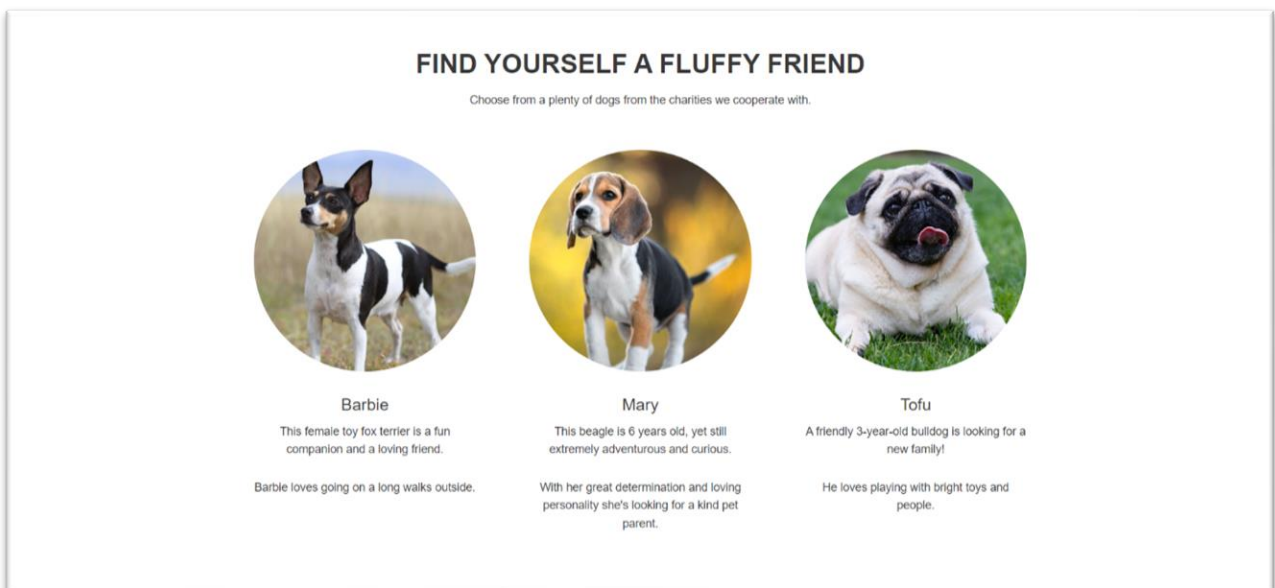
H2 database – managing breed and adoption data.

Github – tracking and managing application development.

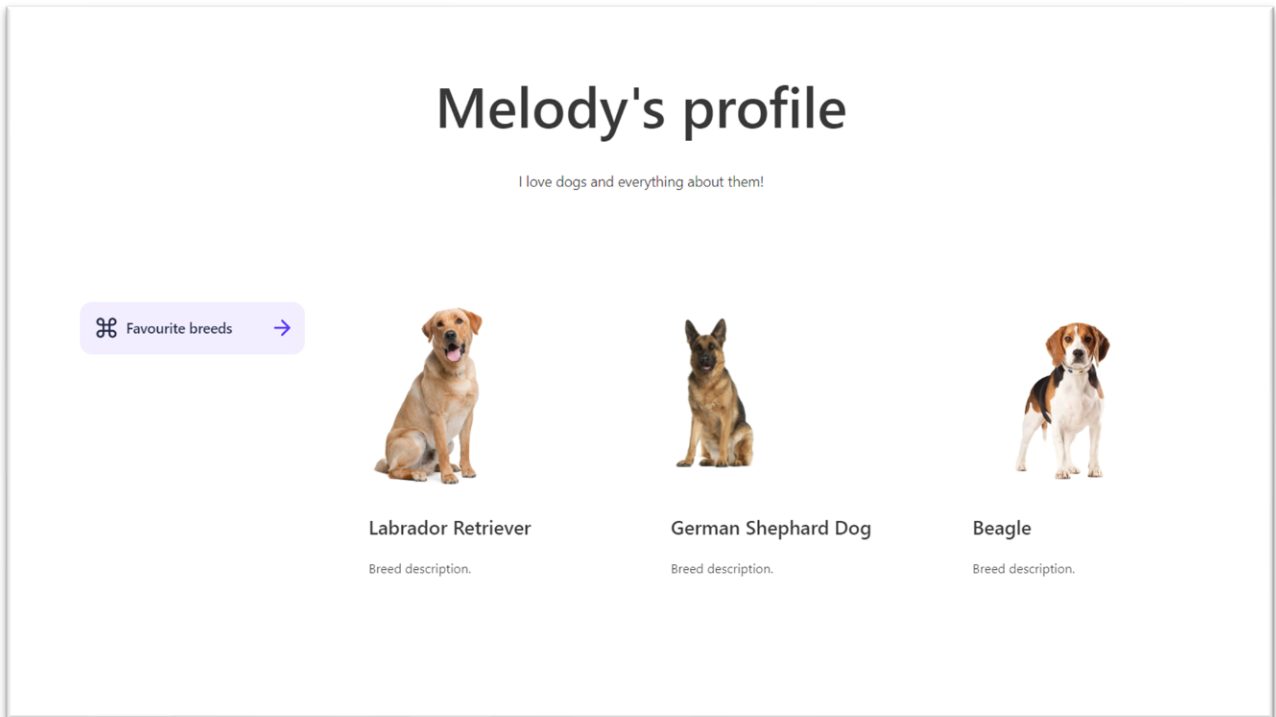
14. Visualization of the appearance of applications/views for each functionality



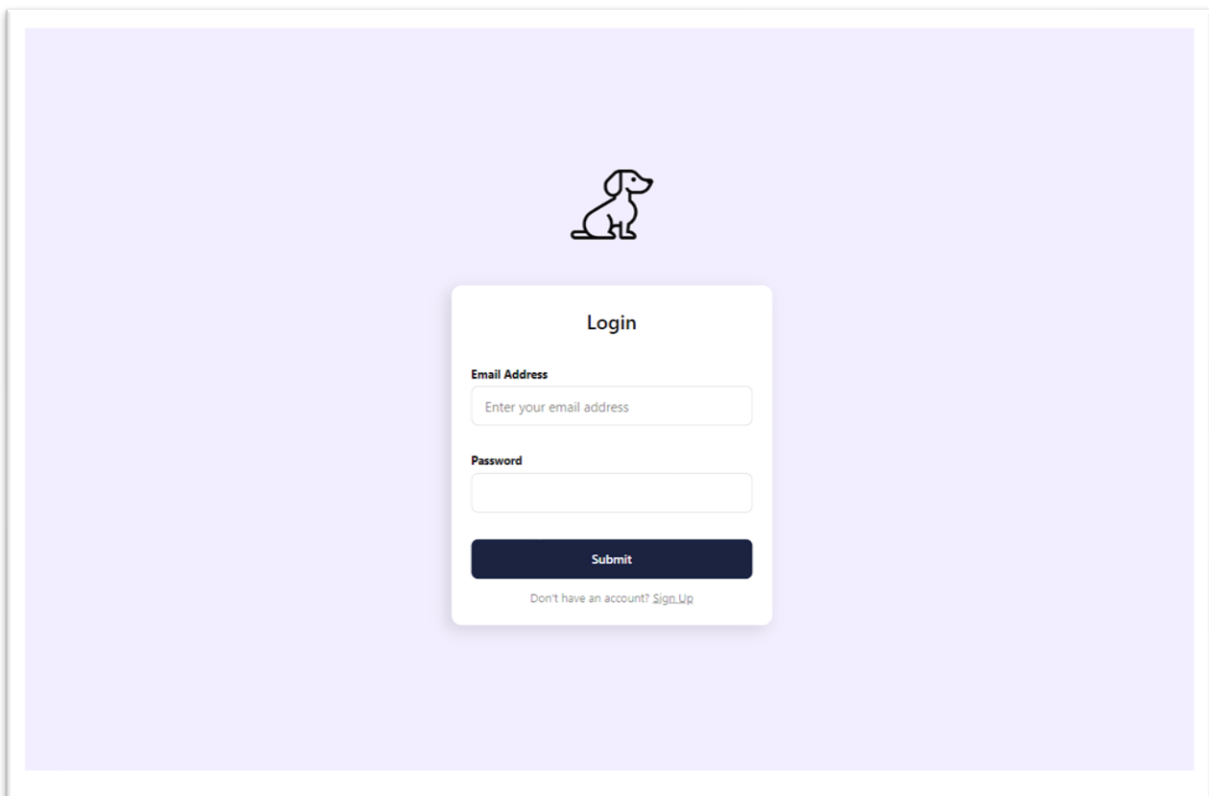
Index page, redirecting user to breed page and displaying links to some of the most popular breeds.




Adoption page, redirecting user to each adoption option.



User's profile displaying list of one's favourite breeds.



Login page (also with redirection to the registration page).



Sign up

Username

Email Address

Password

☐ I agree to this website's [Privacy Policy](#) and [Terms of Service](#).

Submit

[Already have an account? Login](#)

Registration page.

Menu ▾

Dog Encyclopedia

GET STARTED

Filter ▾

Search

Breed search with sorting options on main page.

Dog breeds

Filter



German Shepherd Dog

Generally considered dogkind's finest all-purpose worker, the German Shepherd Dog is a large, agile, muscular dog of noble character and high intelligence. Loyal, confident, courageous, and steady, the German Shepherd is truly a dog lover's delight.

[Learn more.](#)

Beagle

Not only is the Beagle an excellent hunting dog and loyal companion, it is also happy-go-lucky, funny, and extremely cute. They were bred to hunt in packs, so they enjoy company and are generally easygoing. The Beagle's fortune is in his adorable face.

Breed page with all breeds.

Dog breeds

Filter



German Shepherd Dog

Generally considered dogkind's finest all-purpose worker, the German Shepherd Dog is a large, agile, muscular dog of noble character and high intelligence. Loyal, confident, courageous, and steady, the German Shepherd is truly a dog lover's delight.

[Learn more.](#)

Sample search result.

Administrator

GET

POST


PUT

PATCH

DELETE

Send

Administrator page with available commands.



Adoption form

Phone number

Description

Submit

Adoption form a registered user can submit.