Group 7
Full Stack

"Collie Meets Beagle"

A React App

CodeFirstGirls

Tara Patterson

Cecilia Chang

Emily Tsang

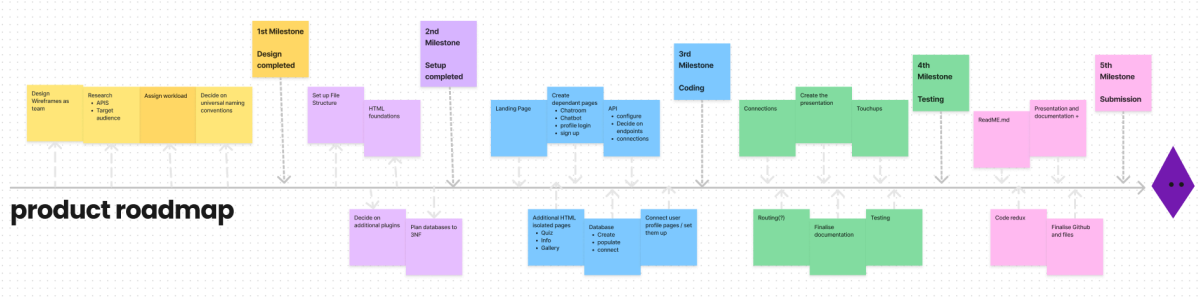
Grace Tashie-Lewis

Christel Gomez

Introduction

The primary aim of this project is to create a React application designed to bring dog owners together, fostering a vibrant and supportive community. Targeting dog owners of all skill levels, our app seeks to enhance their understanding of their dogs' breed, introduce them to ideal local walking trails, and provide a platform for knowledge sharing. By leveraging various React libraries and expanding our expertise in React, Redux, APIs, and design, we aim to deliver a comprehensive and user-friendly experience.

This report will detail the development process and the key milestones achieved throughout the project. Below is a visual representation of our project roadmap that displays the project's timeline and objectives, ensuring a clear understanding of the steps taken to achieve our goals.



Background

This project focuses on developing a React application aimed at dog owners, with the intent to create a supportive and informative community. Understanding the requirements and specific details of this project is crucial to appreciating its scope and functionality. Here are the key components and features that define our app:

1. **User Profiles**: Each dog owner will have a personalised profile where they can share information about themselves and their dogs. This includes details such as dog breed, age, favourite activities, and photos.
2. **Breed Information**: The app will provide comprehensive information about various dog breeds. This includes breed characteristics, care tips, and common health issues, sourced from reliable veterinary and canine organisations online resources.
3. **Local Walking Trails**: Utilising user preferences and mapping APIs, the app will suggest the best local walking trails tailored to the user's answers and their dog's needs.
4. **Technical Framework**: The app is built using React, leveraging libraries such as Redux for state management and various APIs for data integration from a MongoDB database. The design focuses on a responsive and intuitive user interface, ensuring accessibility across different devices.
5. **Security and Privacy**: Ensuring the security and privacy of user data is paramount. The app will implement robust authentication and data protection measures, complying with relevant privacy laws and best practices.

By integrating these features, the project aims to provide a holistic platform that not only connects dog owners but also enriches their experience and knowledge, ultimately fostering a stronger community of informed and engaged pet owners.

Specifications and Design

This project requires both technical and non-technical specifications to achieve its goal of creating a React application that connects dog owners. On the technical side, the application will be built using React for the user interface and Redux for state management. It will integrate various APIs to retrieve and update data, such as dog facts and mapping services, and ensure a responsive design for compatibility across different devices. The backend will utilise Node.js, with database tool MongoDB for storing user profiles,user details, and dog information. Security measures for the backend include user authentication, data encryption, and secure data transmission.

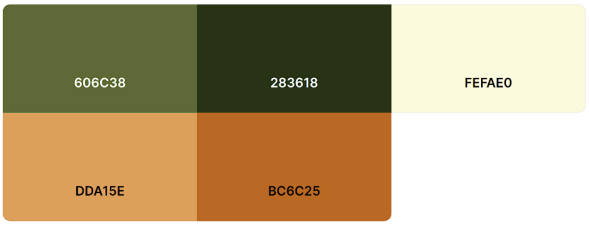
The non-technical requirements focus on providing an intuitive and engaging user experience. The user interface will feature a clean, modern design with high contrast and readable fonts to enhance accessibility. High-quality educational resources on dog breeds, training, and health will be curated and presented in a user-friendly format. The architecture will consist of modular React components, a RESTful API using Node.js, and a non-relational database for data flexibility. Deployment will be managed through Github. This comprehensive approach ensures the app is robust, scalable, and user-friendly, effectively meeting the needs of dog owners and fostering a strong community.

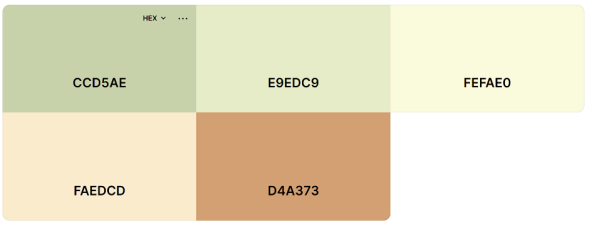
Maintaining user accessibility and usability is central to our design philosophy. We will rigorously test the application for adherence to the law of contrast to ensure that all text is readable against its background, benefiting users with visual impairments. Alt text will be provided for all images, enhancing the experience for users relying on screen readers. Our colour choices will follow colour theory principles to create a visually appealing and functional interface, avoiding colour combinations that are difficult for colorblind users to distinguish. Font discussions will focus on selecting readable, accessible fonts, and we will use appropriate text sizes to ensure clarity and legibility. By incorporating these elements and conducting thorough usability testing, we aim to create an inclusive application that is easy to navigate and use for all dog owners, regardless of their abilities.

Evaluation/Justification of Design:

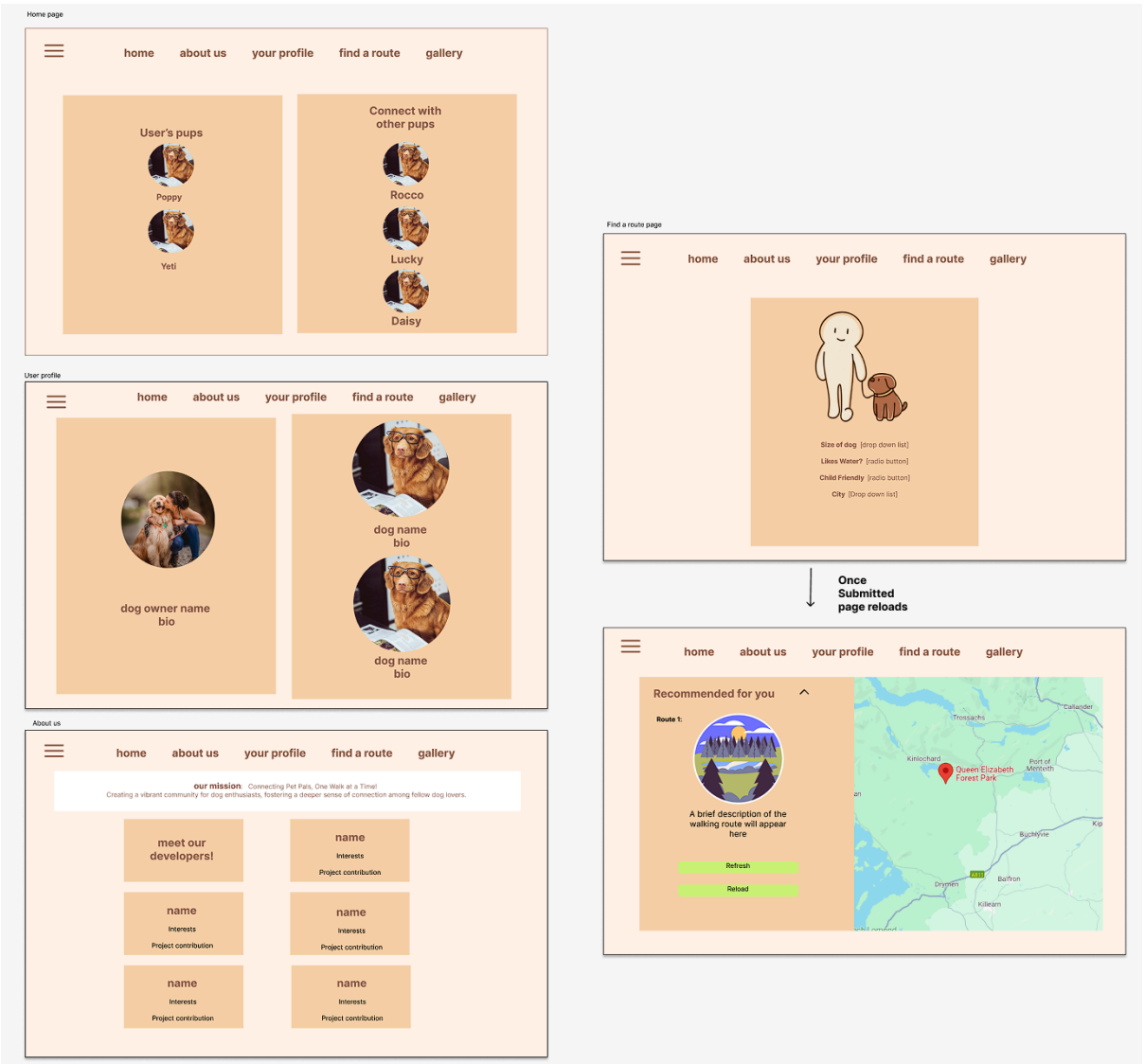
We chose a dog theme as our app is based on dogs and dog walking routes so the colours were shades of beige/brown/chocolate similar to a dog's fur colour, green like the colour of parks and nature. We discovered a UI concept called 'The Law of Contrast'. It describes how colour contrast plays an important role in effective and eye-catching UI. The dark and light contrast used for different elements on the web page enhances readability.
We abided by the web content accessibility guidelines' minimum contrast ratio and used "webaim contrast checker" to check the contrast ratio for improved readability of text and for the user to be able to differentiate different elements on the webpage, inclusive of users with visual impairments such as colour blindness. Font 'Varela Round' was chosen to add to the

clean, minimalistic and modern design. The font’s roundness creates a playful, warm and approachable tone similar to that of a dog.





The initial mock-up created in Figma:



Implementation and execution

Our development approach was rooted in Agile methodologies, specifically leveraging the Scrum framework to manage and execute the project efficiently. We began by conducting a SWOT analysis in our first team meeting to highlight each team member's strengths and weaknesses. This allowed us to assign roles that best matched individual skills and

interests. Emily took charge of designing the mockups and homepage, ensuring the user interface was both intuitive and visually appealing. Chrissie created the database connection using Node.js, created endpoint routers using Express and built object datasets using MongoDB and a Node.js based library Mongoose. Cecilia focused on user authentication, managing the sign-up process, the implementation of Redux for state management and password security. During this project the entire team made strong contributions to the creation of a highly professional React Application. As everyone contributed huge efforts between different sections, a clear depiction has been created in the table below.

	Tara	Grace	Cecilia	Emily	Chrissie
Research	✓	✓	✓	✓	✓
Design	✓			✓	
Wireframes	✓			✓	
Components	✓		✓	✓	✓
Pages	✓	✓	✓	✓	✓
Hooks	✓	✓	✓	✓	✓
Styled components	✓		✓	✓	
CSS	✓	✓	✓	✓	✓
Routing			✓		✓
Testing	✓		✓	✓	✓
APIs	✓		✓	✓	✓
Databases			✓		✓
Authentication			✓		
Redux			✓		✓

Tools

We utilised a range of tools and libraries to support our development process. For communication and task management, we relied on an Agile board from Jira for tracking progress, Zoom and Google Meets for virtual meetings, and Slack for instant messaging and collaboration. Our development environment included VSCode for code editing, with React and various React libraries forming the backbone of our front-end development. Redux was used for state management, MongoDB served as our database solution and Node.js

provided the server environment for our app. The Google Maps API was integrated to provide detailed and interactive maps for local walking trails.

Our Approach

Adopting a Scrum Agile approach, we broke down tasks into 'must-have' and 'could-have' categories to prioritise essential features and manage our workload effectively. This iterative approach involved regular sprints, during which we revisited and refined aspects of our initial design. For instance, we had to abandon the user chat room feature due to GDPR compliance issues. Where appropriate, our team organised paired programming to enhance code quality and foster collaboration. Each branch merge was preceded by thorough team analysis and discussion, ensuring that all changes were scrutinised and aligned with our project goals. This methodical and collaborative approach ensured that we remained agile and responsive to challenges, ultimately delivering a robust and user-friendly application.

MongoDB

The decision to integrate NoSQL, specifically MongoDB, into our backend architecture provided unparalleled flexibility and scalability. This choice facilitated seamless data management, allowing for efficient storage and retrieval of user and pet information. Embracing NoSQL principles empowered us to adapt to evolving project requirements with ease. Diving deeper into MongoDB functionality provided invaluable insights into document-oriented database management. Leveraging MongoDB's rich query language and flexible schema design, we optimised data storage and retrieval processes, ensuring efficient data handling throughout the application.

React

Utilising React hooks and components revolutionised our frontend development approach. React hooks, such as useState and useEffect, streamlined state management and lifecycle handling, enhancing code readability and maintainability. Additionally, modularizing our UI into reusable components fostered code reusability and facilitated rapid prototyping.

Styled Components

Incorporating Styled Components elevated our styling capabilities, offering a dynamic approach to CSS management within our React application. This CSS-in-JS solution allowed for the creation of encapsulated styling components, promoting better organisation and modularity in our codebase.

Enhancing Security with Hashed Passwords

In our application, we implement bcrypt, a powerful hashing function, to secure user passwords during both registration and login processes.

Upon registration, passwords are hashed using bcrypt's bcrypt.hash function, which incorporates a salt and iterates the hashing process for enhanced security. This hashed password, along with other user details, is then stored in the database. During login attempts, the entered password is compared to the hashed password in the database using bcrypt's bcrypt.compare function. This comparison, facilitated by the same salt used during registration, ensures that passwords remain secure, as plaintext passwords are neither stored or transmitted.

This robust hashing mechanism significantly bolsters the security of user authentication within our application, mitigating the risk of unauthorised access to sensitive user information.

Redux Implementation

Redux, known for its powerful state management capabilities, was chosen as the central tool for managing application state. As Redux provides a robust framework for handling state in large-scale applications, we opted to implement it in our login feature to ensure efficient management of user authentication state.

Our Redux setup involved creating a userSlice within the Redux folder using createSlice from @reduxjs/toolkit, with the initial state for the user object set to null. This slice included reducers for login, register, and logout, each updating the user state with the provided payload. In the useLogin hook, a successful login request to the backend (axios.post) triggered the dispatch of the login action with user information obtained from the login form. Similarly, in the useRegister hook, a successful registration request (axios.post) dispatched the register action. Reducers in the userSlice updated the user state accordingly. The selectUser selector function allowed access to the user state.

This implementation ensured centralised state management for user authentication across the application, enhancing maintainability and scalability.

Testing and Evaluation

Our testing and evaluation process was crucial to ensuring the reliability and usability of our React application. Each team member contributed their unique perspective and expertise to the testing phase, providing a comprehensive evaluation of the system's functionality and user experience. Our testing strategy incorporated a combination of unit tests, user tests, and functional tests to cover all aspects of the application. Each member was responsible for writing the unit tests for their own components.

We implemented unit tests using JEST, a powerful testing framework for JavaScript, to validate individual components of our React application. These tests allowed us to identify and fix bugs, ensuring that each component performed as expected in isolation. User testing was conducted on the overall user experience, including the ease of navigation, visual design, and the effectiveness of key features like user profiles, trail reviews, and other

interactions. Functional testing ensured that the application worked as a cohesive whole, with all components interacting seamlessly.

Despite our thorough testing approach, we encountered some system limitations. For example, we had to omit the user chat room feature due to GDPR compliance issues, which affected our initial design plans. Additionally, integrating third-party APIs presented challenges, such as handling rate limits and ensuring data consistency. These limitations required us to adapt and refine our approach, demonstrating the importance of flexibility and iterative development in our project.

During the testing phase, Tara encountered significant challenges with testing the Google Maps component due to dependency conflicts between React and JEST. These conflicts prevented them from effectively running unit tests on the map features, which was critical for ensuring accurate location services and trail displays. Despite these obstacles, they focused on manual testing and utilised user feedback to verify the functionality and usability of the Google Maps integration, ensuring it met the project requirements and provided a seamless experience for users.

Conclusion

In conclusion, our journey to develop a React application connecting dog owners has been both challenging and rewarding. Through meticulous planning, collaborative teamwork, and adherence to Agile methodologies, we successfully brought our vision to life. From the initial design phase to the implementation and testing stages, each team member played a crucial role in shaping the application's functionality and user experience. While we encountered some obstacles and system limitations along the way, our collective efforts and problem-solving skills enabled us to overcome these hurdles and deliver a robust and user-friendly product. Moving forward, we remain committed to ongoing refinement and improvement, ensuring our application continues to meet the needs of dog owners and fosters a vibrant and supportive community. Overall, this project has been a testament to our dedication, creativity, and resilience as a team, and we look forward to seeing the positive impact our application will have on dog owners worldwide.