ByteBoard

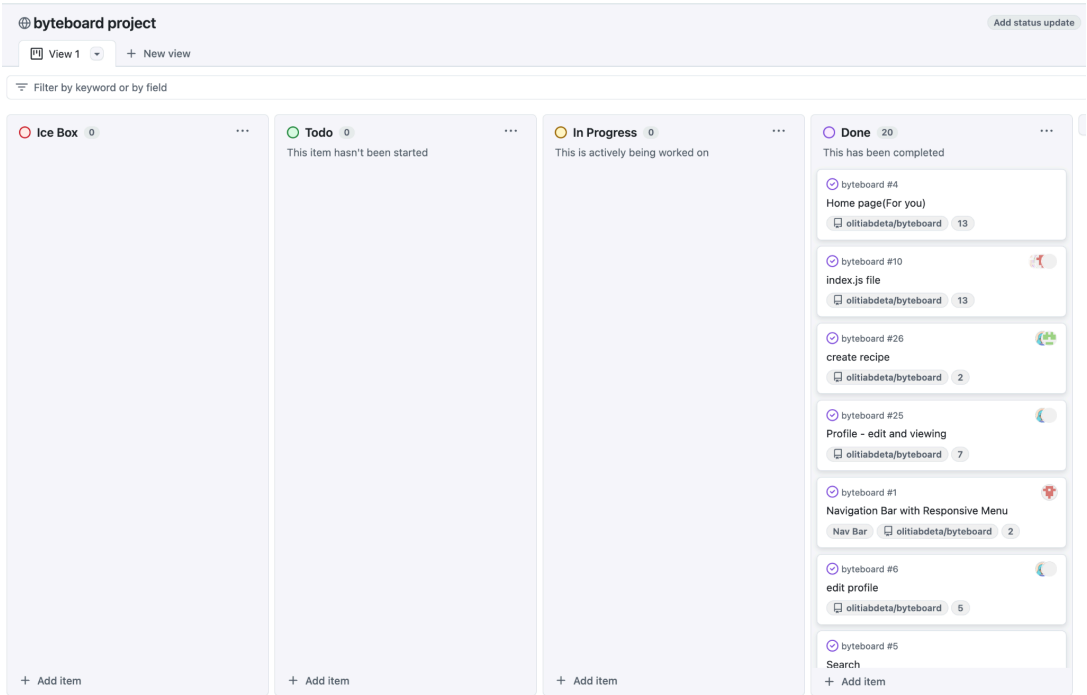
Team Members: Trisha Nookala, Kaylee Dehncke, Oliti Abdeta, Kaitlyn Rapp, Sharon Xiang, Sumeyye Ustunel

Description:

ByteBoard is a recipe-sharing platform designed to make cooking fun and engaging! Navigating recipe sites can be overwhelming, especially for people with busy schedules and new cooks. ByteBoard aims to create a user-friendly space that is easy to browse and allows you to find quick meal ideas. In one place, you can discover new recipes tailored to your preferences and dietary restrictions, create your own recipes, and connect with other cooking enthusiasts. Access all your saved recipe creations displayed in an organized and straightforward manner, making it easy to go back to your favorite meal ideas or reference them for a quick reminder. Searching for more recipes is made simple, you can enter any ingredient or recipe name keyword to explore a variety of recipes that you might enjoy. Inexperience in the kitchen or lack of ability to spend a lot of time curating recipes should not be barriers to incorporating proper nutrition into your lifestyle. With ByteBoard, cooking is accessible for all.

Project Tracker: Here is the link to our project board:
https://github.com/users/olitiabdeta/projects/2

Screenshot if the link does not work:



Project Demo Video: ByteBoard Project Demo

VCS: Link to git repository: https://github.com/olitiabdeta/byteboard
Video demo, project documentation, and project presentation in the milestone submissions folder in the repository.

Contributions of each team member:

Kaitlyn Rapp: As part of the ByteBoard project, I initially focused and contributed on the front end, by creating and refining the Handlebars (hbs) templates, to give us a good base to start. This ensured a cohesive display across the platform to help our ideas stay organized and focused as we progressed. This also included the modification of those GET and POST pathways for these initial files to ensure smooth navigation and proper data handling.

I also developed the Friends page and functionality with the API, allowing users to search for existing users in the database, add them as friends, and view their existing edits to their profiles. I implemented the core logic for those features, ensuring seamless database queries and a user-friendly interface.

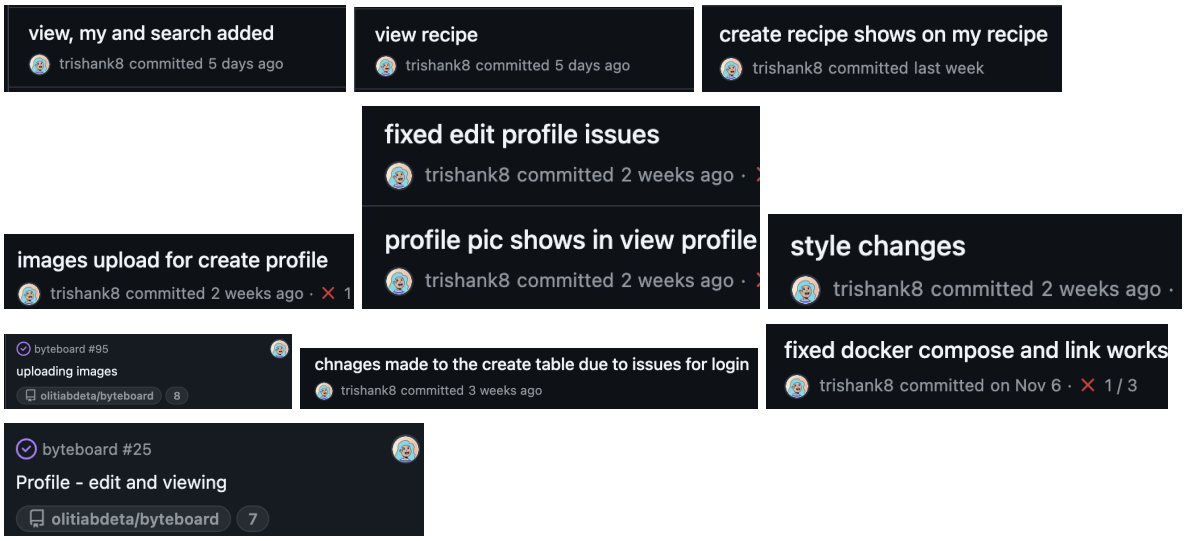
Kaylee Dehncke: For this project, I contributed to multiple aspects of development, including integrating the API key. This involved creating an account to generate the key, configuring it, and implementing all associated API routes. I also focused on debugging and troubleshooting code across various parts of the application, with particular attention to the "Create Recipe" and "My Recipe" pages, where I worked on both the front-end and back-end functionality.

Additionally, I created the navigation bar and structured it as a partial to ensure consistency throughout the application. I also developed and executed the test cases, which were essential for verifying the functionality and reliability of the application. My goal throughout the project was to ensure the team's work came together cohesively and that the application was robust and user-friendly.

Oliti Abdeta: Alongside my team members, I contributed to implementing API routes and templates for our "login", "create recipe", and "friends" pages. I worked on adding functionality, debugging query issues, and addressing error message visibility for users navigating the friends page. Additionally, I ensured a smooth transition from hosting in our local containers to Render deployment, successfully resolving database connectivity and database user authentication issues we faced.

Trisha Nookala: During this project, I contributed to both the front-end and back-end development to improve functionality and user experience. I implemented the "Edit Profile" and "View Profile" features, allowing users to create, view, and edit their profiles. I helped resolve display issues on the "Create Recipe" and "My Recipe" pages to ensure newly created recipes appeared correctly and developed image upload functionality for profiles and recipes. Additionally, I styled the website to provide a cohesive and visually appealing design with

user-friendly navigation. I also addressed Docker-related issues to enable local functionality and resolved redirect problems, ensuring a seamless user experience.



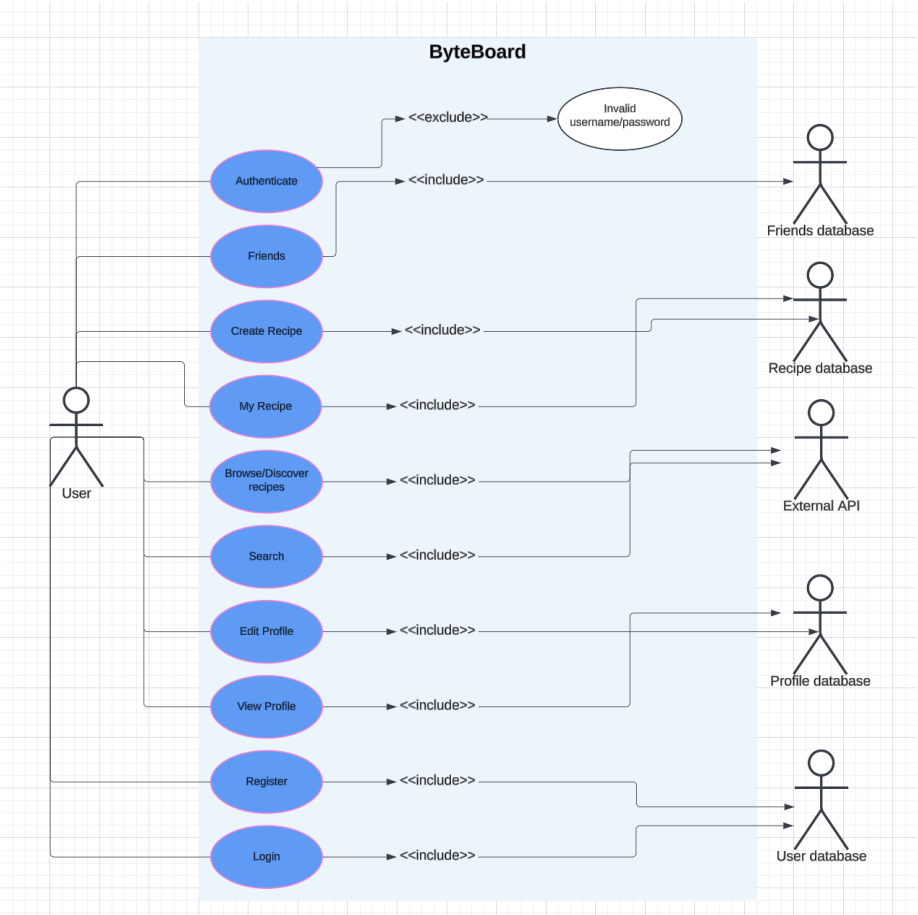
Sharon Xiang:
In this project, I created multiple logo versions and developed SQL tables with various parameters. I helped with research on external APIs for recipe integration. I helped implement picture-uploading functionality and assisted with figuring out Jekyll issues in our GitHub repository. Throughout the website, I made minor edits to maintain consistent layouts and styling. To improve development, I added debugging code and developed routes for several features.

Sumeyye Ustunel: I contributed to the project by implementing API routes for essential functions, including logout functionality and redirecting for login, and registration. I made corrections to errors in the "create table" logic, ensuring smoother functionality. Additionally, I helped to research APIs that best fit the project's requirements, aiding in the integration process. While I attempted to implement the search feature (completed by Kaylee & Trisha I couldn't figure it out), I successfully added new features to the Friends, Login, and Nav sections, enhancing the user experience and the platform's usability.

Use Case Diagram:

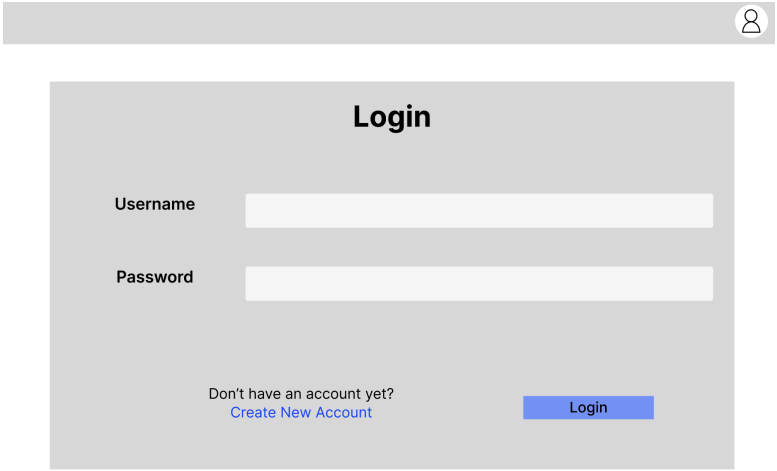
Here is the link to the use case diagram:
https://lucid.app/lucidchart/fef29fb0-8cdd-4c5e-9422-e010a3112b76/edit?viewport_loc=-631%2C-532%2C1668%2C950%2C0_0&invitationId=inv_27e48281-40e8-4b4d-ae52-a5d5214e342e

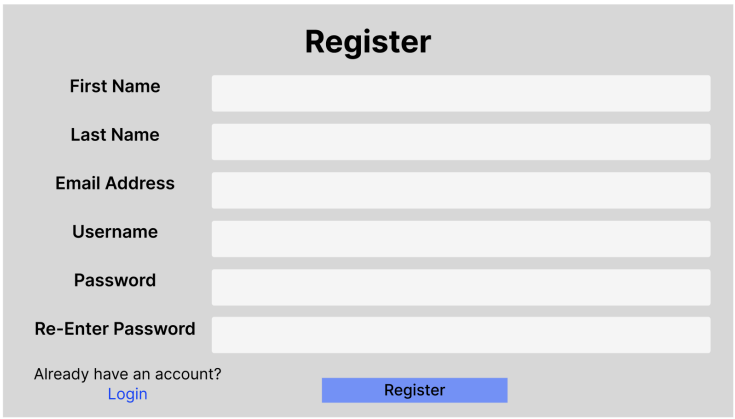
Screenshot of the Use case diagram if the link does not work:



Wireframes: Made the wireframes on Figma. Here is the link https://www.figma.com/design/ez0HPghhIwL9oHsQcLdIBi/ByteBoard?node-id=1-9&t=VlF62PFitQcQKjzg-1

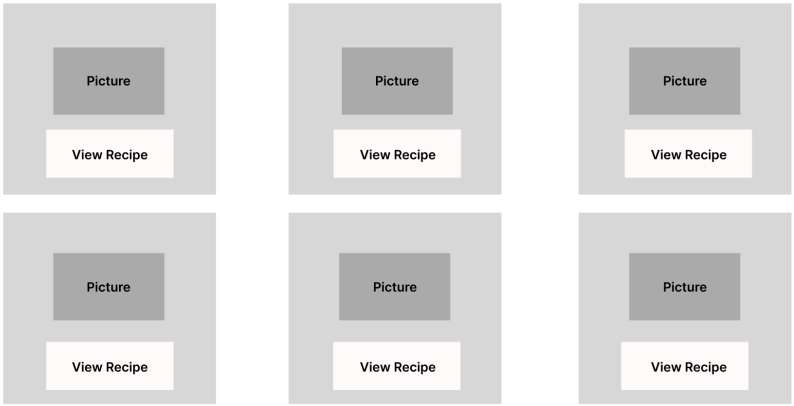
Screenshots of the wireframes below if the link does not work:

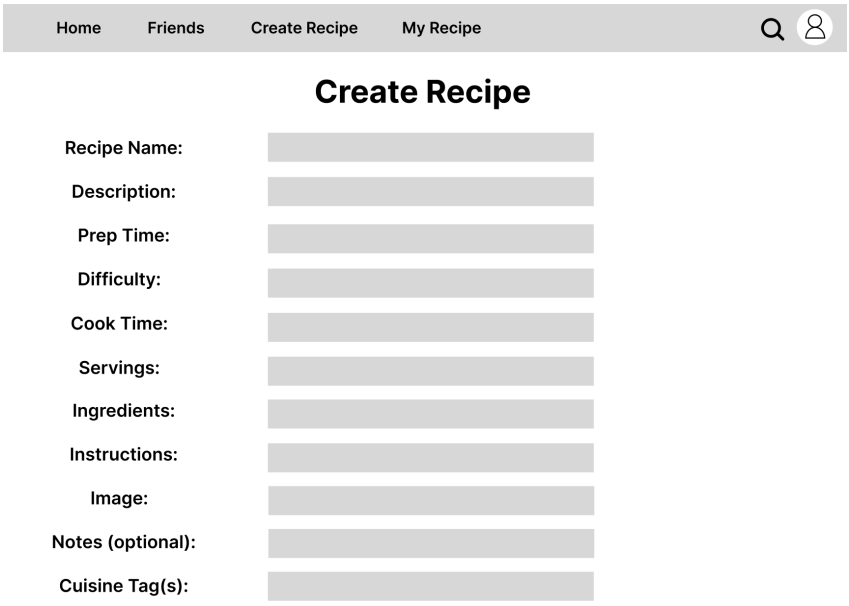




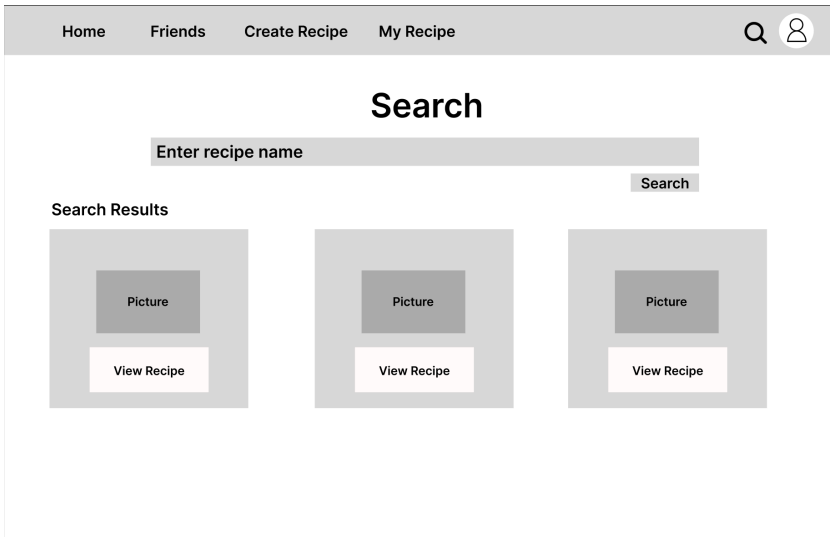
Welcome [user]!

Discover

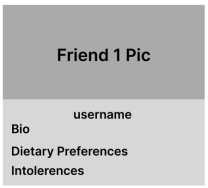



Home
Friends
Create Recipe
My Recipe
Create Recipe
Recipe Name:
Description:
Prep Time:
Difficulty:
Cook Time:
Servings:
Ingredients:
Instructions:
Image:
Notes (optional):
Cuisine Tag(s):

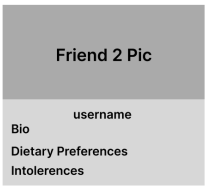

Home
Friends
Create Recipe
My Recipes
My Recipes
Recipe Name
Description
Difficulty
Prep Time
Cook Time
Servings
Notes
Ingredients
Instructions
Picture
Recipe Name
Description
Difficulty
Prep Time
Cook Time
Servings
Notes
Ingredients
Instructions
Picture
Recipe Name
Description
Difficulty
Prep Time
Cook Time
Servings
Notes
Ingredients
Instructions
Picture


Home
Friends
Create Recipe
My Recipe
Search
Enter recipe name
Search
Search Results
Picture
View Recipe
Picture
View Recipe
Picture
View Recipe

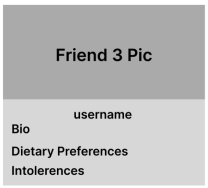
Friends



username
Bio
Dietary Preferences
Intolerences

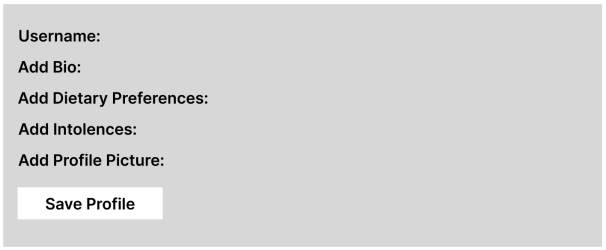


username
Bio
Dietary Preferences
Intolerences

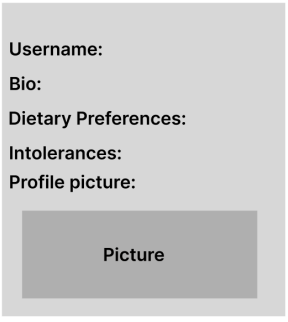


username
Bio
Dietary Preferences
Intolerences

Edit Profile



View Profile



Test Results:
Summary: The testing process for ByteBoard focused on verifying the core functionalities of the application, ensuring reliability, and improving user experience. The test cases covered user registration, login/logout, recipe creation, and error handling. Observations and feedback were used to refine the application and enhance its usability.

Test Cases:

1. Default Welcome Case
    a. Verifies the basic functionality of the server by testing the '/welcome' endpoint. Ensures it responds with the correct status code and a welcome message, confirming the server is running and accessible.
2. User registers correctly
    a. Tests the '/register' endpoint with valid input to ensure new users can successfully register. This case checks that user data is stored securely, including password hashing, and that the server responds with a success status.
3. The user registers with an existing username or email
    a. Tests the '/register' endpoint for proper error handling when a user attempts to register with a duplicate username or email. Confirm the application prevents duplicate accounts and returns a clear error message to the user. The user is then prompted to log in if the username or email exists in the database.
4. The user logs in correctly
    a. Verifies that the '/login' endpoint allows users to log in with valid credentials. This case ensures the server authenticates users and provides access to subsequent application features.
5. The user logs in with an incorrect password
    a. Tests the '/login' endpoint's handling of invalid credentials. Ensures the application denies access, returns a proper error message, and does not compromise security by revealing unnecessary details about the failure.

Test Coverage:

- Core functionality: Covered all critical user actions (registration, login, recipe creation, and logout).
- Error Handling: Verified responses to invalid inputs and prevented security vulnerabilities (e.g., duplicate registrations, incorrect login credentials).
- User Experience: Focused on client-side validation and clear error messages to guide users.

Observations:

Users interacted with the endpoints as intended, testing for expected responses and error handling. For the '/welcome' endpoint, users confirmed server functionality without issues. In registration, some users expected immediate feedback for invalid fields, prompting enhancements like client-side validation and clearer error messaging. Duplicate registration handling worked well, but users suggested improving form behavior by clearing fields after failed attempts. During login, users appreciated the error messages but suggested highlighting failed

attempts more prominently, leading to better alert systems and retry options. These observations helped refine usability and align the application with user expectations.

Deployment:

We utilized Render as our deployment platform to ensure the application was accessible and operational for all users. Render allowed us to host both our Node.js server and PostgreSQL database, creating a seamless environment for deployment.

Live Application:

https://byteboard-egej.onrender.com/

How to Access:

- Simply visit the live link above. The app is fully functional and accessible.

Deployment Details:

- The app is containerized using Docker, ensuring consistency across development and production environments.
- Environment Variables: Managed securely in Render for database credentials and API keys.
- PostgreSQL is configured with connection pooling for optimal performance.