

**Title:** WalkBoulder

**Members:**

Name:	Github:	Email:
John Mathews	jmmathewsiii	john.mathewsiii@colorado.edu
Luc Iñaki Palacios	Lipikan-0	lupa6264@colorado.edu
Isabelle Lindfors	EpochAncientEnigma	isli2136@colorado.edu
Eddie Strand	eddie410	edst4577@colorado.edu
Parth Chudappa	ParthChudappa	pach7325@colorado.edu
Josh Kim	joshmkim	joshmyeongkim@gmail.com

**Summary:**

WalkBoulder is an app that gives users important information about walking/hiking trails in and around the city of Boulder. It uses the Google Maps API to display a map of Boulder and allows users to create a trail between two places in Boulder. These locations are validated by the Google Maps Autocomplete feature. The website also saves these trails so that they can be displayed on the home page. Users are then able to search for trails, review them, and add them to their profiles.

To allow for personalization, we provide users with the ability to change their profile with updated information, such as name and email, and a profile picture. Additionally, users can engage with others through the Posts feature, where they can upload images with captions that are displayed for all users on the app.

Furthermore, users can connect with one another by sending friend requests. Upon acceptance, the two users are connected in the database. This was not a core feature of the website, but it could be improved by allowing users to share walk information with their friends and providing them with the ability to save walks as group walks among friends.

## Project Tracker:

Link: <https://github.com/users/joshmkim/projects/2/views/1>

## Screen Shots:

The screenshot displays a Project Tracker interface with four columns: **Ice box** (5 items), **Todo** (0 items), **In Progress** (0 items), and **Done** (21 items). The **Done** column is expanded to show a list of completed tasks.

**Ice box** (5 items):

- walkboulder #21: Integrate WalkScore data
- walkboulder #32: Display the Walk Score in the UI
- walkboulder #45: Transfer Walk Data to A Walk Buddy
- walkboulder #33: Fetching WalkScore data and figuring out how to implement it into our database
- walkboulder #18: Make an footer for website


**Todo** (0 items):

**In Progress** (0 items):

**Done** (21 items):

- walkboulder #46: Get Start and End Locations Using API
- walkboulder #35: Write a review of a trail
- walkboulder #36: Access Reviews of trails
- walkboulder #16: UI to display map on page
- walkboulder #22: Create a route
- walkboulder #29: Display the route on the map
- walkboulder #23: Display my route on the map
- walkboulder #24: Add Friends
- walkboulder #25: Add info to main page
- walkboulder #27: Make a Navigation Bar
- walkboulder #19: Integrating navigation bar
- walkboulder #31: Add Sign in and Account functionality
- walkboulder #20: Draft User Profile Page
- walkboulder #26: Recent Walks
- walkboulder #53: Set up Achievement System
- walkboulder #28: Style Homepage
- walkboulder #39: Adding trail and user tables to creat.sql
- walkboulder #40: create insert.sql
- walkboulder #76: Posts page
- walkboulder #77: Login/logout/register pages
- walkboulder #78: Settings page

Each task card includes a status icon (green circle for 'Ice box', blue circle for 'Todo', yellow circle for 'In Progress', and purple circle for 'Done'), a title, and a description. The 'Done' column also features a vertical scrollbar on the right side.

**Video:**  screen-capture (1).webm

**VCS:** <https://github.com/joshmkim/walkboulder>

## **Contributions:**

### Parth Chudappa:

**Home Page:** Created home page, with all relevant information and endpoints, implemented search bar, implemented database functionality for trails, created trails page with all trails in home page, implemented save features for trails to save them to profile page

**Trails Page:** Created uniquely auto generated pages for each trail that's added to db, added functionality for reviews and images for these trails in the db to be linked, implemented all these features to appear on trails page

**Profile and Settings:** Created friend functionality for these pages, and improved on existing structures to link functionalities to add saved trails appear on profile page

**Nav Bar:** Made Navbar, linked login functionality, created conditional buttons

**Styling:** Styled Home, Login, Logout, Register, Profile, Posts, Trials, Settings

**Bugs:** Fixed bugs with search bar, DB trials linking, rating functionality, review functionality linking, Review Page buttons, login/logout not working, navbar buttons not appearing, modal bugs, fixed some functionality with profile not appearing properly

### Josh Kim:

**Register, login, logout pages:** I created the endpoints, SQL tables, and initial styling for these pages.

**Settings page:** I created a settings page to allow users to edit their password and usernames, and I also added a few more columns in the users table so that users could add things like their email, names, and profile picture.

**Multer + Bytea image integration:** I set up endpoints that work through multiple pages so that we could save images in our database for things like profile pictures and trail images.

**Profile page:** I created an initial profile page to display user information like name, username, and profile picture

**Posts page:** I created endpoints, styling, and SQL tables for a social media page where users can upload images for others to see with a caption.

Isabelle Lindfors: I set up the initial database and its layout - as well as run some unit and integration tests to ensure that the database was properly connected and figure out how to establish the query needed later in the project. I also worked on - with the help of ChatGPT - getting a SQL function with a trigger set up so that add friends would automatically be bidirectional. After that, I worked on stuff for the profile page: getting the achievements, walk history, and showing friends displayed and established; and tried to, but failed to get the add friends functionality working.

Jack Mathews: I created the Maps page that allows the user to create a walking route. This process involved developing an understanding of the Google Maps JavaScript API and two of its key features. While Google provided much of the code, I made significant modifications so that the features of our website could be included. I also made modifications to the database, the Profile page, and the Reviews page so that the API calls would work properly. Finally, I completed the POST route that allows the user to save a route from the Maps page.

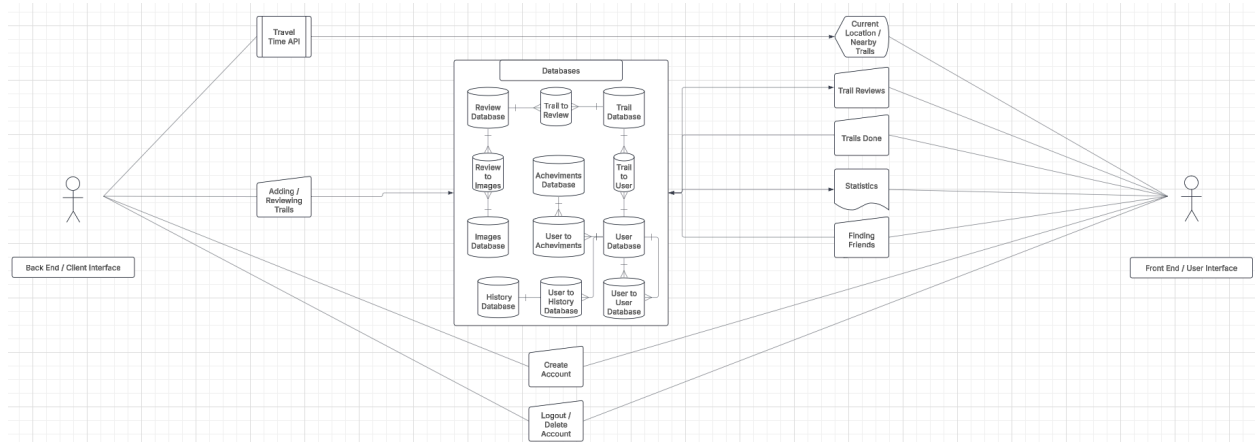
Luc Iñaki Palacios: I worked extensively on the Google Maps API. Most of the front-end Google code provided to implement the API had to be updated to fit our requirements; the original code provided was around 2000 lines, so understanding and sifting through it was very time-consuming. The modifications included fixing the starting position of routes instead of having a set starting location, modifying all the modals to receive the correct data, working with and adding to the Google location objects to extract the correct data, and finally assisting in the POST API request to insert the route data into the database.

Eddie Strand:

I implemented everything seen in the reviews page including the review modal and list of all reviews based on highest rated and most reviewed. I modified the review sql table, set all of the sql queries for reviews and submit-reviews, added java functionalities to sort reviews and calculate ratings, and html/css for reviews page. (All of the reviews.hbs, post/get reviews, post submit-reviews, reviews table).

**Link:**

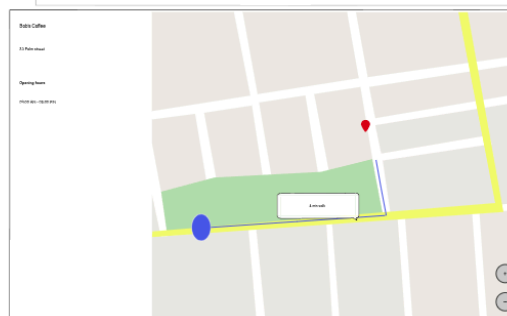
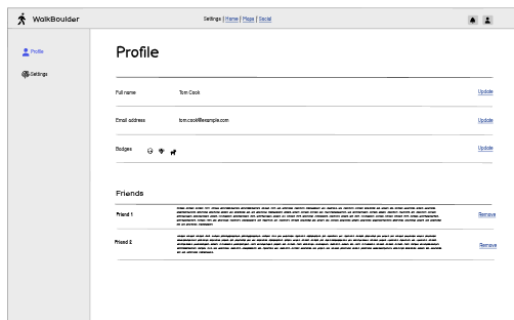
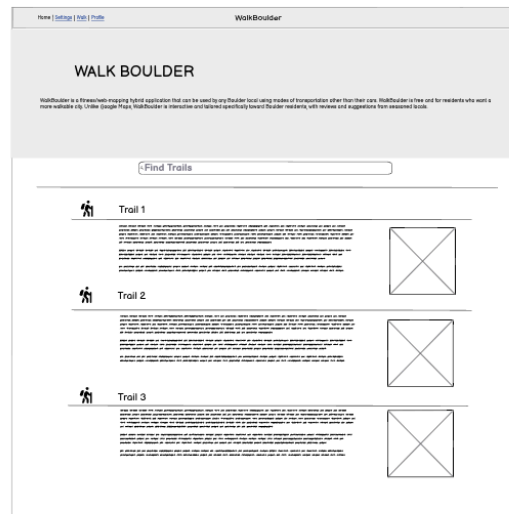
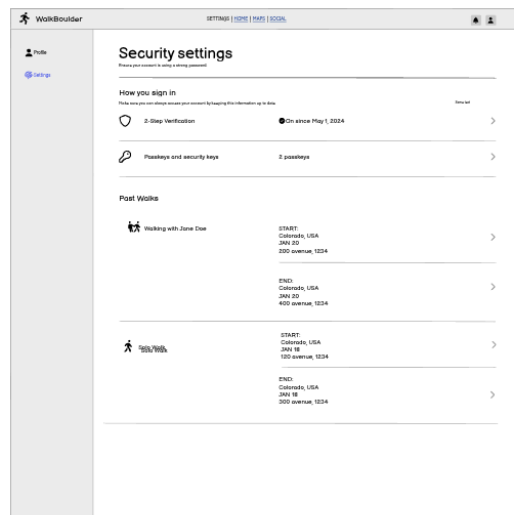
[https://lucid.app/lucidchart/f5b39da3-557b-45c1-8c9b-dcb5779e0682/edit?invitationId=inv\\_8d07f909-d363-412b-8c79-2b4ec21efc22&page=0\\_0#](https://lucid.app/lucidchart/f5b39da3-557b-45c1-8c9b-dcb5779e0682/edit?invitationId=inv_8d07f909-d363-412b-8c79-2b4ec21efc22&page=0_0#)



## Wireframes:

**Link:** <https://balsamiq.cloud/seuhii7/pkt7uw0/rE49B>

**Image:**



## Test Results:

### Test 1: *Adding friends*

This test included sending and receiving a friend request from a user's randomly generated ID number. Once requested, the friend can be accepted. The test was successful and both users are updated with their new friend. Both friends' lists are updated with one another's information.

### Test 2: *Transfer walk data to a friend*

We had hoped to implement functionality to send information about a walk to a friend, but we did not prioritize it and were unable to include it in time.

### Test 3: *Writing a review of a trail*

When a user presses "Write Review" on the Reviews page, they are prompted to choose a trail, select a rating out of five stars, and write a comment about the trail. After submitting the review, we can see that the new review is shown under the list of reviews for that trail. Additionally, the average rating is updated. We did not include difficulty information as difficulty is neither an objective or subjective metric, so it would not make sense to allow users to judge a trail by its difficulty. We do include the distance of the trail so that users can better understand how much time it may take to complete the trail.

## Deployment:

Render Link: <https://walkboulderwebservice.onrender.com/>

If Render fails, to run the app locally, follow these steps:

1. Pull the repository from GitHub
2. Create a .env inside ProjectSourceCode containing the following:

```
# database credentials
POSTGRES_USER=<any username>
POSTGRES_PASSWORD=<any password>
POSTGRES_DB="users_db"

# Node vars
SESSION_SECRET=<any secret>
```

3. Navigate (cd) to ProjectSourceCode
4. Run "docker compose up"