

SP-26 — Keyboard Final Report

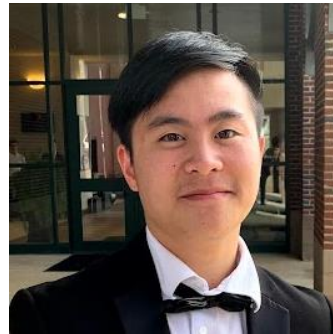
CS 4850 - Section 02 – Fall 2023

Dec 3, 2023

Bazeed



Caleb



Isaac



Roy



Professor Sharon Perry

Website Link: <https://keeplist.github.io/>

GitHub link: <https://github.com/Keeplist/keeplist/> (private as requested)

Contents

Project Plan	3
Software Development Designs	7
Web Page Layout	7
Navbar	7
Home Page	8
Login/Registration	8
Profile	8
Edit/Create Profile	9
Keeb Profile	10
Edit/Create Keeb Profile	11
Version Control	11
Narrative Discussion	12
How the project was created.....	12
Implementation	13
Creating the website.....	14
Navbar	14
Home Page	15
User Profiles and Keeb Profiles.....	17
Challenges	19
Test Plan and Test Report	19
Conclusion.....	20

Project Plan

An updated version of our project plan can be viewed below. Goals were shifted a lot during development so the plan has undergone changes in terms of milestone events and final deliverable.

As a general overview of the product:

Our Keyboard List website will allow users to create their own keyboard on the website. They can link the different parts they want for the build in order to keep track of the building process as well as ensuring that all the parts will function together without issues. The website will also allow users to create accounts and show off their keyboard creations by listing the parts they used to build the keyboard. Each user can display their build (or multiple builds) on their profile and can then share their builds with friends.

As of the version uploaded with this report, the database has not been implemented. Therefore, users cannot create their own accounts, and can only view the design of the website and general layout and structure of the website. Source code will be attached in the zip file that is submitted with this report. This project will likely be expanded outside of the classroom, where we hope to figure out database implementation.

Final Deliverables

1. Final report with descriptions of website development and product.
2. Website that has the designs highlighted in the design documentation of software development, but without login system. Users can click through the example profile created, as this will be what future updates will be based on.
3. Hub website with redirects to website and our GitHub repository with all resources used.

Milestone Events

#1 By 9/15/23

- Determine the tools we will be using to develop the website.
- Front-end and back-end development solidified.
- Website design planned.
- Begin website development.

#2 By 10/30/23

- Website prototype completed.
- Home page completed.
- Website design solidified.

#3 By 11/17/23

- Final website completed with requirements listed below.
- Final report submitted with complete documentation.

Meeting Schedule Date/Time

Every Week: Monday or Wednesday @ 4:45 (stand up meets)

Progress Check for Milestone 1 9/8/23

Progress Check for Milestone 2 10/20/23

Progress Check for Milestone 3 11/9/23

Collaboration and Communication Plan

The major communication channel to be used will be Discord. Response time will be within 24 hours, and everyone will be committed to typing/calling in this channel whenever they need help/need to communicate.

Bi-weekly stand-up meetings to track progress and ensure everyone is on the right track. This will be done on discord, and meeting notes role will be alternated each week to ensure everyone gets a chance to take notes.

Tools for file sharing will be Discord and GitHub.

1.0 Introduction

1.1 Overview

This is a Software Requirements Specification for the KeebList website. It will be organized into separate sections for functional and non-functional requirements. Refer to the table of contents to jump between different sections for in-depth explanations on the requirements listed.

1.2 Project Goals

The goal of this project is to create a website that will be a template to model a keyboard profile site that will incorporate databases at later stages. This will allow users to sign up and create their own keyboard profiles.

1.3 Definitions and Acronyms

Keeb: This is just slang for keyboard, which is in the website's name and various elements within it.

1.4 Assumptions

Some assumptions that arose were centered around the behavior/intention of a user visiting the website. One of them involves the input of a user. It should be noted that all user input should be treated as potentially harmful or dangerous. Furthermore, user-supplied content (i.e., keeb posts) should align with the criteria/guidelines of the website to ensure that unrelated content does not ingress the platform. The instance in which the advent of multiple users simultaneously accessing the website is also an important assumption to consider. A way to mitigate the computational overhead is to scale the resources to combat bottlenecks.

2.0 Functional Requirements

2.1 Home Page

There will be a home page that will allow the user to view different photos, as well as having an information page that will explain to the user how to use the website (this will contribute to usability). There will also be a logo and a login/sign-up button at the top of the screen. This will be the first page most users will see.

2.2 Profile Page

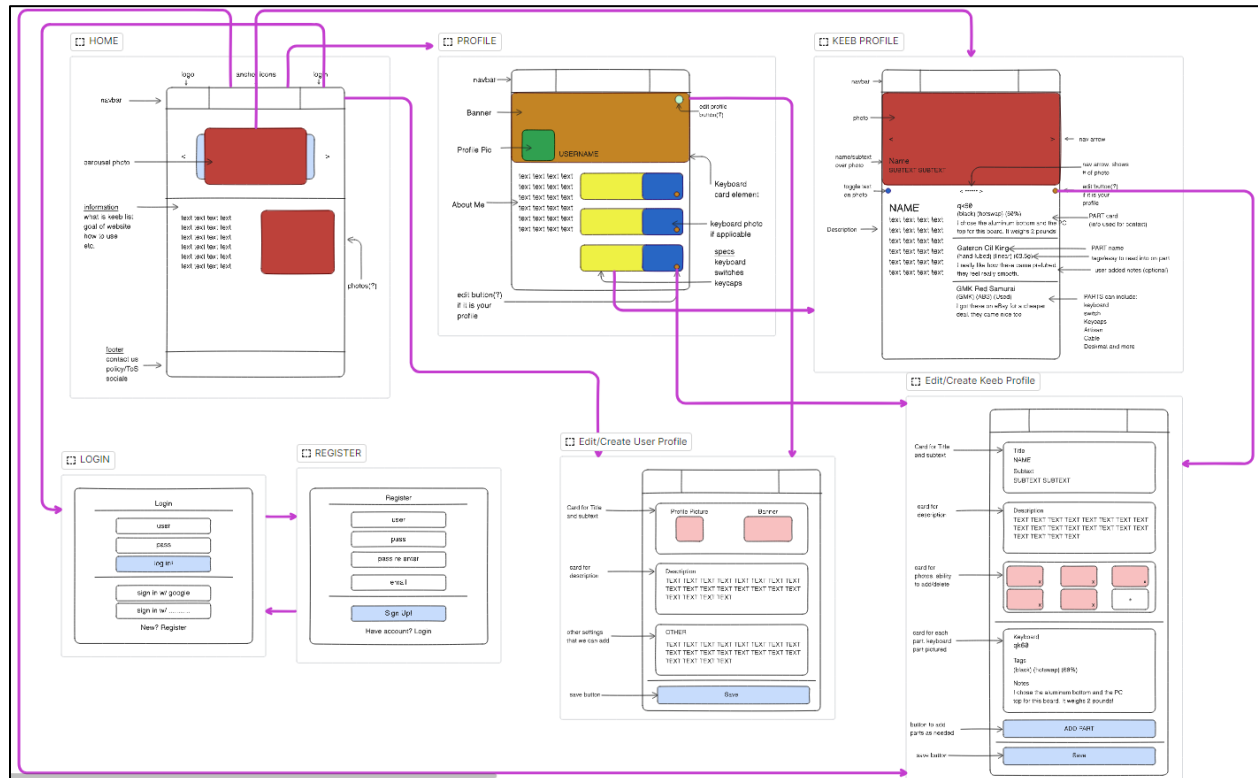
There will be a profile page that will allow users to view theirs or other users' profiles. This page must contain an edit button that when clicked, will take the user to the edit/create profile page. This page will be viewable but not editable with the final product.

2.3 Keeb Profile

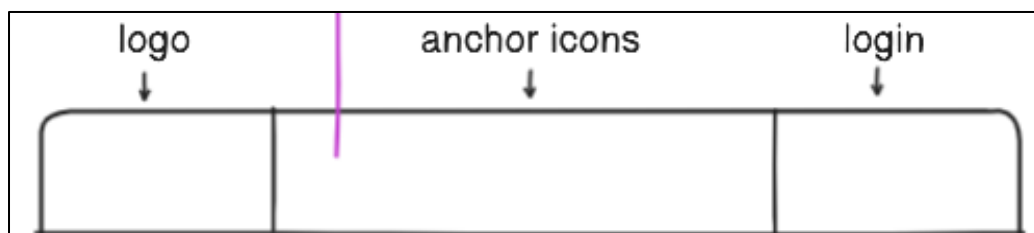
There will be a keeb profile which will be tied to each user. A user can have multiple keeb profiles. A keeb profile will contain information about a specific keyboard that the user owns. The user can click to add more keeb profiles and set up their profiles. The website will contain an example keeb profile page.

Software Development Designs

Web Page Layout



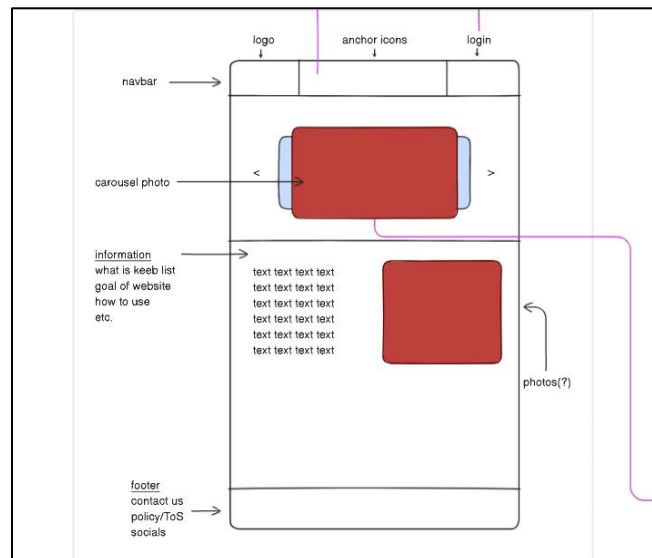
Navbar



All pages will display a navbar at the top. From left to right, we will have our logo, anchor icons, and the login area.

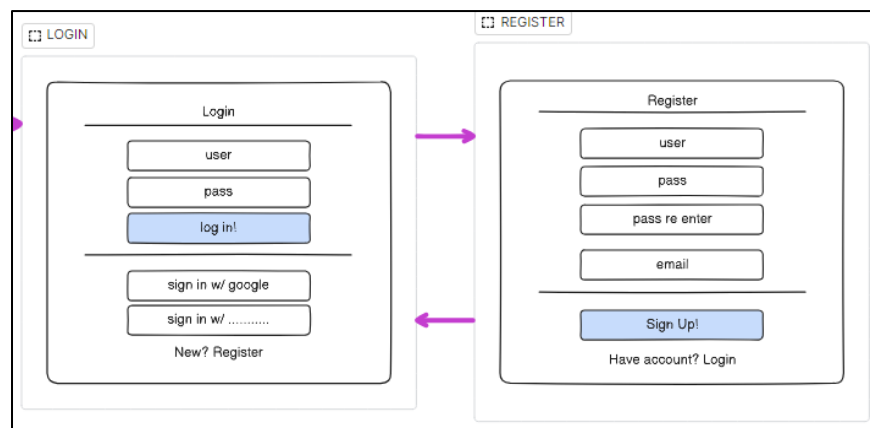
- Our undeclared logo will be created by us. Clicking this logo will send the user back to the home page.
- We will use icons that are anchor elements to take users to different parts of the website. We have no set number of pages yet, so the number of icons is undeclared. If the icons do not match the style of the website, we can revert to text anchors.
- The login area is where users will be able to login and register. When the user logs in, we can add a drop-down menu showing user functions.

Home Page



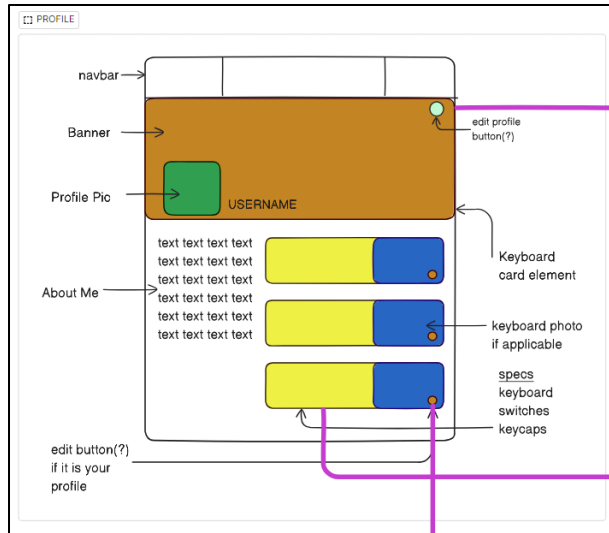
The homepage will show off random builds in the carousel at the top. Below this, we can add information about the website. For a footer, we are unsure if we will be adding this as of now.

Login/Registration



Login and registration are straight forward. We want to allow the users to sign in with Google. For registration, we just require a username, password, and email. **THIS WILL NOT BE INCLUDED IN THE FINAL PRODUCT BUT WILL BE USED FOR FUTURE UPDATES OUTSIDE THE CLASSROOM.**

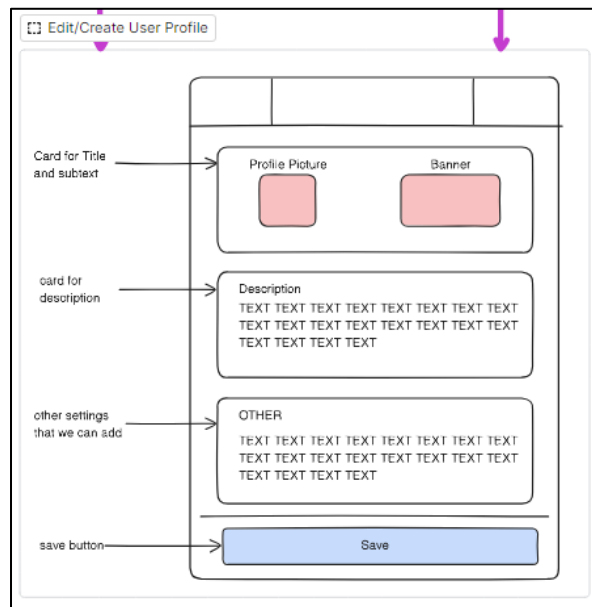
Profile



User profiles will display their profile picture (PFP), a banner photo, about me text, and their keeb builds. There will be an edit profile button somewhere on the page is the user is viewing their profile. For the keeb build display, display a photo and some general information. If it is the user's profile, show an edit button for the keeb.

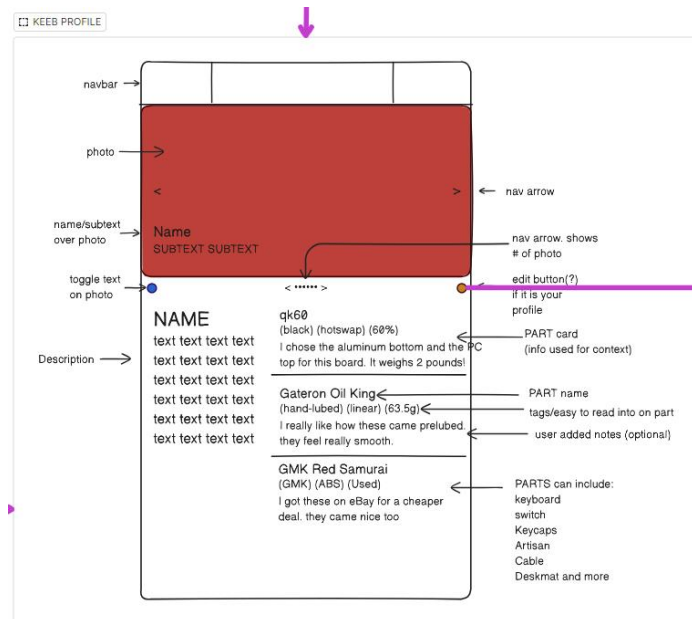
- Potentially display friends here if we add a social aspect to the site.

Edit/Create Profile



In this area, users will be able to interact with these card elements. Each card will have information about the user profile that can be edited. Display a save button at the end. **THIS WILL NOT BE INCLUDED IN THE FINAL PRODUCT BUT WILL BE USED FOR FUTURE UPDATES OUTSIDE THE CLASSROOM.**

Keeb Profile



In the keeb profile, the user is first shown a photo. There will be the name of the board and some subtext (if the user includes it) displayed above the photo. Add some navigation arrows on the side of the photo as well as some navigation on the bottom.

- Potentially add a toggle for the image text. This will allow users to see the photo in its entirety.

On the left column, we will have the name displayed again. Below, user added text that can be about the board.

On the right column, we will have the list of parts. Display the name of the part on the top. Below, tags are displayed as quick information about the part. This can include the color, hardware specs, modifications, build materials, etc.

Edit/Create Keeb Profile

Diagram illustrating the layout of the "Edit/Create Keeb Profile" form, showing various sections and their corresponding labels:

- Card for Title and subtext: Title, NAME, Subtext, SUBTEXT SUBTEXT
- card for description: Description, TEXT TEXT TEXT TEXT TEXT TEXT TEXT TEXT TEXT TEXT TEXT TEXT TEXT TEXT TEXT TEXT TEXT TEXT
- card for photos ability to add/delete: (Placeholder for photos)
- card for each part keyboard part pictured: Keyboard, qk60, Tags (black) (hotswap) (60%), Notes (I chose the aluminum bottom and the PC top for this board. It weighs 2 pounds!)
- button to add parts as needed: ADD PART
- save button: Save

Here, the user will be able to edit all the keeb profile elements. They will be sorted into cards. Each card will have the appropriate information editable so the user can tune it to their liking. At the bottom, we have a button to add another part. This will allow users to add parts where applicable. Some parts are keyboard, switch, keycap, stabilizer, artisan, cable, desk mat, etc. Finally, a save button at the bottom. **THIS WILL NOT BE INCLUDED IN THE FINAL PRODUCT BUT WILL BE USED FOR FUTURE UPDATES OUTSIDE THE CLASSROOM.**

Version Control

For version control, GitHub was used for all management. GitHub is linked at the start of this document. This repository contains all changes that have been made throughout the project's lifespan and will continue to be updated as more changes are made in the future outside of the classroom.

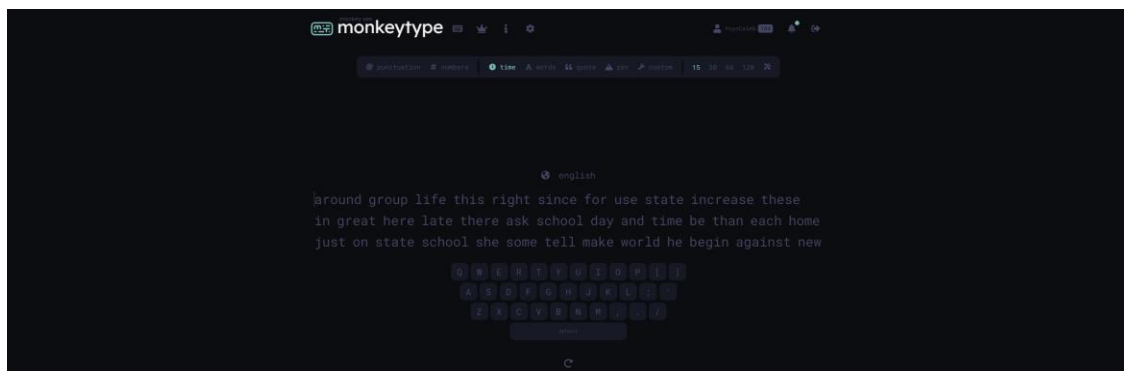
The GitHub will be maintained and be the main source of version control for this website.

Narrative Discussion

[How the project was created](#)

The project first started off as an idea and did not yet mature into anything tangible. The initial step was to sketch up a rough wireframe/mockup of our website's interface and functionality. Our project lead, Caleb, created a rough mockup of the website we wanted to create. The mockup spanned several iterations until everyone came to a general agreement.

To provide more context, we wanted the website to fit a niche of minimality and sleekness. We all made it our goal to do some research on preexisting websites that conform to this niche. We had to take several aspects of this niche into account. This included typography, symmetry, contrast, and ease of use. We quickly planned to center our website's design around a website named Monkeytype. We knew that Monkeytype would lay a great foundation in the process of building our website. Our team completed a wireframe of our website on Eraser.



We made it our goal to list out some of the technologies that we could use in our tech stack for us to have a working proof of concept. We first identified the main components of our tech stack that would satisfy the requirements listed in our project. We knew that since our tech stack would partially be full stack, we had to consider frontend and backend frameworks, tools, and technologies to use.

For our frontend, we considered C#, JavaScript, HTML, Bootstrap, Tailwind, ASP.NET, React, Vue, and Next.js. For our backend we considered Firebase, Node.js., Bun, Deno.js and Vercel. We ended up utilizing the following:

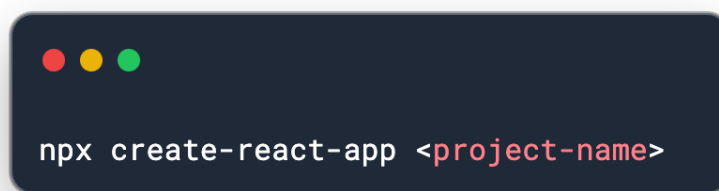
- Node.js for our JavaScript runtime.
- JavaScript as our core language.
- React as their library offers extensive tooling.

- Next.js as our frontend/backend framework as it offers ready to use tools for building pages, components, routing, data processing and more.
- Vercel for deployment of our website
- Firebase's Firestore as a means of storing user data.

We've soon come to realize that Firebase would not be incorporated in the early stages of development since we've agreed that it would be best to dedicate time to it in the evolution stage. We decided to pivot our development time into web development so that we could lay down the foundation of a website that can be built upon after we finish the course and have more time to continue development.

Implementation

In the implementation stage of our project, we built our application using npx's (i.e., Node Package execute) create-react-app command to start off with a simple React app that we could modify to meet our demands.



We quickly noticed that react alone wouldn't be enough to satisfy our demands/requirements, since react is just a library, NOT a framework! Our team weighed our options on using React by itself or introducing Next.js into our development environment. We eventually went with the latter. Complications were bound to show up since Next.js and React development environments don't exactly share the same configuration files and structure. So, we were tasked with creating a new branch from the home test branch. We went ahead and refactored a couple source files to make sure our Next.js app wouldn't break on execution. This meant that we had to change the directory structure of our previous development environment and introduce a conventional style of naming too.

A directory for components allowed us to incorporate components from our React app's development environment with ease. Our routes or pages sat in a directory named routes for our create, profile, and later a login page. The last 3 files are mainly for the root page which is where the home page sits. Layout defines the structure of the webpage using JSX syntax which gives us the ability to write HTML-like code in it whereas Page would contain at least zero or more components and functionality.

Creating the website

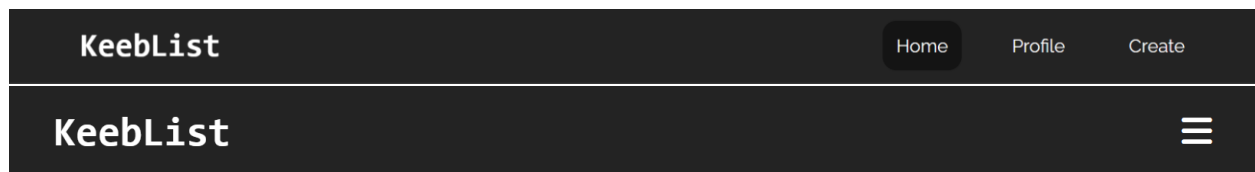
To create the website, we did much research on react.js and CSS development to make sure our website would look. A lot of development was done using codepen.io to see our changes live. This helped us with implementation, as we could make decisions very quickly on how we wanted the website to look. Codepen.io also allowed us to get our website to look as close to our initial design mock-ups as possible.

One of the most important considerations we had to take was the responsiveness of the website. This ensures that the website will look good despite the size of the window or device. Whether the user is full screened, has the browser windowed, or has the website open on their mobile phone, the website should be able to conform its different elements to the size. To review, we will discuss some of the important code we worked on for the website:

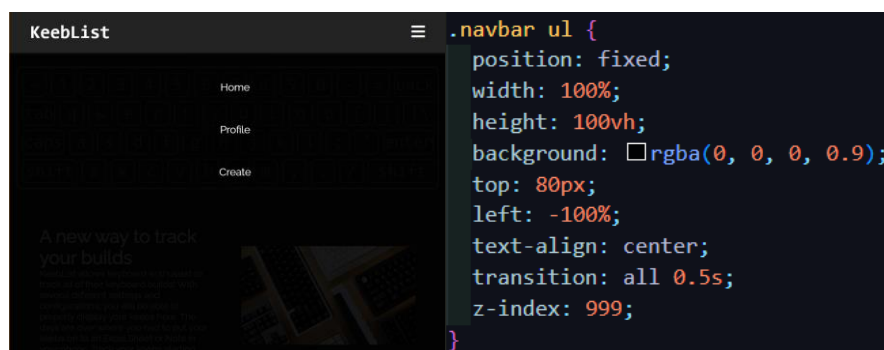
Navbar

The navbar is one of the most important elements on the website. It appears at the very top and is what most people will see first. Our navbar has a simple design with the name of the website/logo on the left side and navigation links on the right. We also added an effect when hovering over these links.

When the width of the website is below 960 pixels, the right-hand navigation links will change to a hamburger menu. Clicking on this icon will reveal the navigation links.



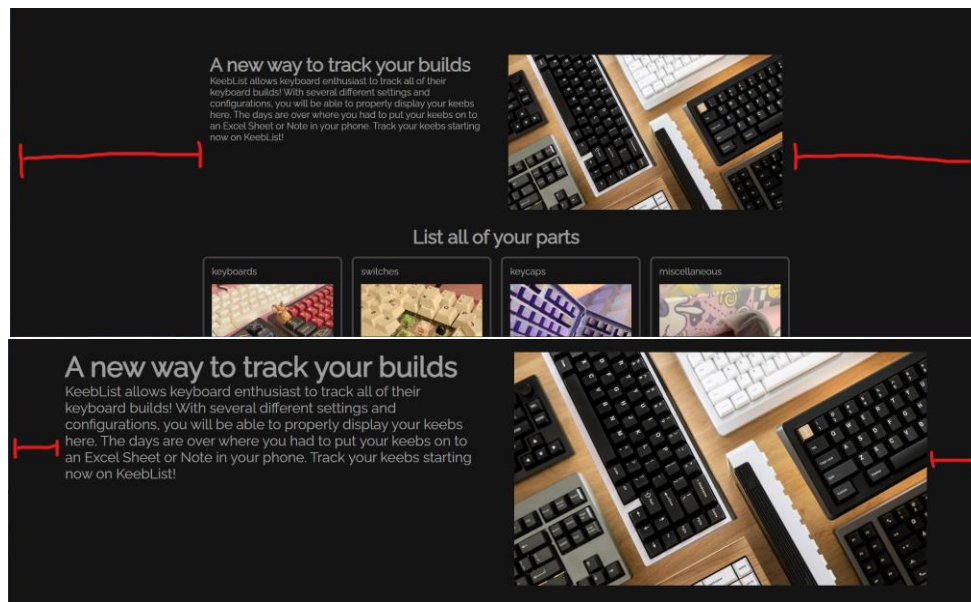
When the screen width is 960 pixels, the size of the navigation links changes to 100%. That way, they stack on top of each other rather than being adjacent. Next, all the navigation links are offset to the left by -100%. This puts them off screen. Once the hamburger menu is pressed, and transition occurs that changes the offset back to 0%. Thus, this will center the navigation links and they will be shown.



Home Page

When creating the homepage, we noticed that a lot of websites had a common way of centering the elements. Depending on the width, extra space would appear on the sides of the content. As the width of the window decreases, this extra space also decreases. When the window is small enough, all the contents will then adjust their size accordingly.

Here, you can notice the different size of space on each side of the contents. The contents stay at the same width of 1350 pixels and only the space around them decreases in size.



This is done by making the contents width 1280 pixels with the display of flex. This allows the context to be the same size and the flex allows the sides to change in size depending on the window.

Once the window size reaches 1350 pixels, the contents change from width 1280 pixels to 100%. This means it will take up the width of the window. This transition is seamless. The display of flex will then alter the size of the elements to fit the window. Finally, once the width of the window is 800 pixels, the elements are changed to display grid. This allows the elements to need to do homework to stack on top of each other.

```

@media (max-width: 1350px) {
  .element-container {
    width: 100%;
  }
  .element-2 {
    flex: 1;
  }
  .container h1 {
    font-size: 35px;
    color: var(--text-color);
  }
  .container p {
    font-size: 17.5px;
    color: var(--text-color);
  }
}

@media (max-width: 800px) {
  .element-row-2 {
    display: grid;
  }
  .element-2 {
    margin-bottom: 10px;
  }
  .element-4 {
    flex-basis: 35%;
    margin-bottom: 10px;
  }
}

```

Below, you can see an example of the page when the width is below 800px. Each element in the row we'll have therewith adjusted to 100%, causing them to stack on top of each other. Notice how the text box is above the picture.



That is how the responsiveness of the website works. This is very important as it is used in other elements on other pages.

The home page is made of two different sections: the keyboard hero section and the cards. The keyboard hero section is comprised of several div elements within each other. There is a div for the hero container, the keyboard outline, each row, each letter's square, and the letter itself. Each row has an array of letters and symbols. We use the arrays to easily fit the letters and symbols into a square for the keyboard row. For some special instances, such as tab, the size is adjusted accordingly.

```


const keebR1 = ["1", "2", "3", "4", "5", "6", "7", "8", "9", "0", "-", "="];
const keebR2 = ["q", "w", "e", "r", "t", "y", "u", "i", "o", "p", "[", "]"];
const keebR3 = ["a", "s", "d", "f", "g", "h", "j", "k", "l", ";", "'"];
const keebR4 = ["z", "x", "c", "v", "b", "n", "m", ",", ".", "/"];

```

For the cards, we have a div to hold all the elements together, a div for each row, and a div for each element.


A new way to track your builds

KeebList allows keyboard enthusiasts to track all of their keyboard builds! With several different settings and configurations, you will be able to properly display your keeb's here. The days are over where you had to put your keeb's on to an Excel Sheet or Note in your phone. Track your keeb's starting now on KeebList!




List all of your parts


keyboards




switches



keycaps



miscellaneous



How do I start?

Simply create an account and then go to the create tab.

In the first row, we have a text box and a picture. In the second row we have four cards displaying different parts. And in the final row, we just have a text box.

User Profiles and Keeb Profiles

For the user and keeb profiles, they have, generally, the same layout. They both have a hero section followed by two elements divided in the middle below. Let's dive into each section of the profiles.

For the hero section, it is like the keyboard hero section on the home page. Instead, we are using a picture instead of an element created in HTML/CSS. Text is provided in the bottom left for the username or title of the keyboard. Some differences include a profile picture for the profile page and subtext for the keeb page. The hero section of the keeb page is also noticeably taller.





For the two sections below the hero section, they are divided in the middle. The sections are divided into the left and right. Once the page becomes small enough in width, they will stack on each other like the elements on the home page.

On the left, there is just a text box. This is meant to describe the user or keeb on the page. On the right, we have some variance between the two pages. On the user profile, we have card elements to display their keyboard builds. For the key profile, we have cards for each part used in the keeb.

Welcome to my Keeb page! I have built 125+ keyboards and do not plan to stop. Please check out some of my builds!

QK60
A Keyboard built for a Samurai

Zoom75
One of the prettiest boards that I have built

WIND X R2
My girlfriends board with neapolitan switches

QK60

This is qwertykey's QK60. The board is Win-keyless and has a polycarbonate top frame, aluminum bottom frame, and stainless steel weight.

This was my first 60% board and I really liked it. I like the small form-factor. I love how the keyboard and keycaps are color matched

QK60
BLACK | RED | HOTSWAP | 60% | FR4 PLATE

I really like how the PC top looked on the Alu bottom. The board weighs around 2lb!

GMK Red Samurai
GMK | ABS | Used

This was my first GMK set and I got a really good deal on them on eBay. I even got the hirigana legends like I had wanted

Gateron Oil King
Pre-Lubed | Linear | 63.5g

The lube job on the switches were great from the factory. I just use them stock because I am too lazy to hand-lube them

Challenges

The biggest challenge for this website was trying to create and link a database for the website to allow for account creation. As we continued to try and develop the site, we realized that creating a functional database that will allow all the features we initially outlined in our deliverables would not be completed by the time of the final project submission. Therefore, we opted to move database linking to a stage of development outside of the classroom moving forward. This is also highlighted in the narrative discussion, where we go more in depth on this specific challenge.

Another challenge was that our group was not very familiar with web development at the start of this project. We decided to continue with this project because each group member is very passionate about the keyboard hobby. This caused a lot of time to be spent on researching web development, which also affected our decision to cut the database for this build.

Test Plan and Test Report

The project's test plan mainly consisted of having friends and family click through the site to ensure that all functional requirements were met. Some examples are stated below:

- To test the Navbar, we ensured that each click on a specific Navbar element would redirect to the appropriate page.
- To test the different cards and pages, we clicked through each page's functionality to ensure it worked as specified in the requirements.

We also used Lighthouse, an automated tool to assess the quality of web pages and achieved a high score for our website. The site works as it should with the requirements mentioned. Further test cases will be developed after properly setting up a database in the future.

Conclusion

Throughout creating the project, our goals and requirements shifted a fair bit. We realized that the original project had a much larger scope than we could create in the time we had, so we needed to scale back to ensure a completed product by the end of the semester.

The requirements of the project shifted over time. The current product contains all specified requirements such as a functioning home page, example profile page, and example keyboard profile pages.

The actual design of the product did not change as much throughout development but needed to be scaled back. The elements that are present in the design documentation but are not yet available on the website are for future planning when a database has been linked and is functional.

For version control, a GitHub repository has been maintained to check all updates on the website as more features are added.

Test results show that the current website functions as intended with the requirements listed in this document.

Finally, because of the challenges we had faced during development, we decided that cutting out the database for now was the best option to ensure the website design and layout was completed by the time the project was due.