Team Work Log

A) Planning Phase

In this stage, we (team) gathered to brainstorm potential website ideas and agreed to create a recipe website tailored to beginner cooks. The goal was to help people who often find cooking intimidating, like college students, young professionals, or even newlyweds learning to cook for the first time. For example, we have planned to include a “5 - ingredient meals” section to make recipes feel less overwhelming. We also discussed adding tips for common cooking terms (e.g., what it means to "sauté" or "fold in the cheese") to support true beginners.

We defined site goals such as:

i) Providing easy - to - follow instructions.

ii) Offering quick meal ideas.

iii) Including basic kitchen safety tips.

By the end of this phase, we had a shared vision of a helpful, non - intimidating platform that empowers users to try cooking with confidence.

B) Design Choices

Design was where our creative side came into play. We opted for a clean, minimalistic layout using HTML and CSS, ensuring that users wouldn’t get lost in visual clutter. For example, instead of using overly flashy animations, we have gone with a simple navigation bar at the top, clear headings, and structured recipe cards with step - by - step blocks.

Our chosen color scheme would have included soft pastels like light green or beige to evoke calmness in the kitchen, while fonts like "Open Sans" or "Roboto" made the text legible and welcoming. These design choices show an understanding that the website needed to feel approachable, not like a fancy chef’s blog, but more like a friendly cooking coach.

C) Technologies Used

We built the project using HTML for structure (e.g. recipe titles, ingredient lists, and steps) CSS for styling (e.g. adding spacing, colors, and borders) and JavaScript for small interactive elements. One example could be a “Show/Hide Ingredients” toggle button or a dropdown that lets users filter recipes by difficulty level.

Using Replit allowed our team to collaborate remotely and simultaneously - for instance, one person could write the HTML while another adjusted the CSS. It also provided instant previews of our site, which sped up our workflow and allowed real - time corrections.

D) Challenges Faced

No good project comes without hiccups. Team faced issues like:

i) Broken links (e.g., clicking "Breakfast Recipes" leading to a blank page).

ii) Layout misalignment (like recipe cards stacking incorrectly on smaller screens).

iii) Font inconsistencies between pages.

We overcame these by testing regularly and using peer feedback and by sharing the workload evenly. For example, one of us have noticed that the "Contact Us" button disappeared on mobile view and suggested switching to a responsive layout using CSS Flexbox or Grid. These issues, though frustrating, taught valuable lessons in debugging and responsiveness.

E) Testing and Revisions

Testing wasn't just a final step — it was an ongoing process. We tested the site across different browsers (Chrome, Firefox, and even Edge), and on different devices, like checking whether recipe text was readable on a smartphone. For example, we have found that a heading was too large on mobile or that images were slow to load.

After gathering feedback from all teammates or even test users (friend/sibling), we made our revisions on next few points:

i) Fixing slow - loading images by resizing them.

ii) Making buttons larger for mobile users.

iii) Ensuring that all internal links navigated properly.

These tweaks made your site more functional and user - friendly, particularly for the beginner audience we were targeting.

F) Final Reflections

Wow. So, we actually built a website. From scratch. Using actual code. That alone feels like a major win — considering at the start some of us weren't entirely sure if "HTML" was a type of sandwich (spoiler: it's not!).

What we really took away from this project, though, went beyond code. Sure, we learned how to put a website together — structure with HTML, make it look nice with CSS, and add a pinch of JavaScript to make things pop. But more than that, we figured out how to work together without losing our minds. Which, honestly, is impressive.

There were moments when things felt messy — like when a link wouldn't work no matter how many times we stared at it, or when the layout looked perfect on one laptop and like a recipe for disaster on another. But somehow, between the testing, reworking, and mutual pep talks ("Okay, don't panic, it's just CSS") we made it through.

We also figured out how to divide and conquer. One of us took the lead on writing up the recipes, another on the layout, one of us tackled the color scheme (shout - out for picking something that didn't burn our eyes) and another worked the coding magic. And we didn't just work in our own corners — we checked in with each other, gave suggestions, and asked questions (most of the time very panicked ones, but still).

The most surprising part? By the end, we weren't just randomly throwing tags and styles together — we actually started to get it. We understood why things broke, how to fix them, and what to do differently next time. We even got a little excited about tweaking margins and choosing fonts. (Who knew the font drama was real?)

Last words on topic, we didn't just walk away with a website. We walked away with real teamwork skills, a starter pack of coding knowledge, and the bragging rights to say, "Yeah, we built that." And now, if someone ever asks us to build a basic website or help fix a broken link, we won't run — we'll maybe even volunteer. That's Huge Progress!!!