Week 8

Goals

Here are the goals for this week:

- [ ] Build a static `<Sidebar />` component.

Recap and a Quick Note...

I know a lot of people couldn't finish implementing state during last week's session, and that is ok! The concept of state is non-trivial and much harder to implement on your own than tasks from prior weeks.

Which brings me to my note...

If you have free time, go watch a YouTube video and go back in to try again! Even better, read the documentation!

I hate to admit it, but reading documentation is something I only really started doing a few months ago, and I realize now that it is extremely useful and a high ROI for your skills as a programmer/student/human imo.

Try it out!

Now for the objective...

Your Objective

Right now, our Kudos Board is just a form and some cards on a blank page. "Real" apps have navigation, a persistent header, and a consistent layout. When you click a menu button on a mobile site, you naturally expect a sidebar to behave and feel a certain way.

Your goal this week is to build a component of that static app layout. You'll create a static `<Sidebar />` (side menu) using TypeScript, and style it using CSS.

Motivation

You might be thinking, "Why are we making an individual `Sidebar.tsx` file? Couldn't I just dump all the HTML into `App.tsx`?"

Wellllll yes, you coulddddd, BUT you'd be setting yourself up for a massive headache later.

This idea of breaking your UI into small, self-contained pieces is one of the most important concepts in React (and web dev).

Here's some reasons why:

- DRY (Don't Repeat Yourself): This is a core programming principle. If you find yourself copying and pasting the same chunk of HTML/CSS/JS, you should probably make it a component. Why? Because if you need to change it, you only have to change it in one file, not 10.
- It's just easier □□...
  - **Easier to Fix (Maintainability):** If your sidebar has a bug, you know *exactly* where to look: `Sidebar.tsx`. You don't have to hunt through a giant 1000-line `App.tsx` file to find the broken `<div>`.
  - **Easier to Read (Readability):** When someone else (or you, 6 months from now) looks at your `App.tsx`, they won't see a wall of code. They'll see `<Sidebar />`, `<KudosForm />` and `<KudosCards />`. You instantly know what the page is made of.
  - **Easier to Grow (Scalability):** Need another sidebar on a different page down the line? Just import `<Sidebar />`. You can pass it a prop to change its behavior. This makes building new features way faster.

By building a clean component today, you're building a something that you can trust, understand, and reuse anywhere, which is key to building large, complex projects.

Task Outline

Implement a responsive navigation layout. This will involve creating a `<Sidebar />` component.

Your Checklist

To get this done, you'll need to:

1. Create a new component file `Sidebar.tsx` (and a CSS file, `Sidebar.css`).
2. Build the `<Sidebar />`.
   - It should be a vertical bar positioned on the left side of the screen.
   - Add a few placeholder links inside like "Home," "Give Kudos," and "Profile." (don't worry about them not going anywhere yet!)
3. Now, go into your `.css` files and make it look good! In `Sidebar.css`, style the `.sidebar` class to be a vertical bar that's always visible on the left.
4. In your main `App.tsx`, implement your component.

Some Hints...

Remember prior weeks! Our past sessions have covered the skills you need to do this, so feel free to reference them

Here's some icons for your sidebar. Simply copy this code and paste (define) at the top of your `Sidebar.tsx` file:

```
import './sidebar.css';

// --- Helper Icon Components (Replaces lucide-react) ---
// We define simple SVGs here to keep the file self-contained.

// Icon for "Home"
const HomeIcon = ({ className }: { className: string }) => (
  <svg
    className={className}
    xmlns="http://www.w3.org/2000/svg"
    viewBox="0 0 24 24"
    fill="none"
    stroke="currentColor"
    strokeWidth="2"
    strokeLinecap="round"
    strokeLinejoin="round"
  >
    <path d="m3 9 9-7 9 7v11a2 2 0 0 1-2 2H5a2 2 0 0 1-2-2z" />
    <polyline points="9 22 9 12 15 12 15 22" />
  </svg>
);

// Icon for "Give Kudos" (using a simple "gift" icon)
```

```
const KudosIcon = ({ className }: { className: string }) => (
  <svg
    className={className}
    xmlns="http://www.w3.org/2000/svg"
    viewBox="0 0 24 24"
    fill="none"
    stroke="currentColor"
    strokeWidth="2"
    strokeLinecap="round"
    strokeLinejoin="round"
  >
    <polyline points="20 12 20 22 4 22 4 12" />
    <rect x="2" y="7" width="20" height="5" />
    <line x1="12" y1="22" x2="12" y2="7" />
    <path d="M12 7H7.5a2.5 2.5 0 0 1 0-5C11 2 12 7 12 7z" />
    <path d="M12 7h4.5a2.5 2.5 0 0 0 0-5C13 2 12 7 12 7z" />
  </svg>
);

// Icon for "Profile" or "User"
const UserIcon = ({ className }: { className: string }) => (
  <svg
    className={className}
    xmlns="http://www.w3.org/2000/svg"
    viewBox="0 0 24 24"
    fill="none"
    stroke="currentColor"
    strokeWidth="2"
    strokeLinecap="round"
    strokeLinejoin="round"
  >
    <path d="M20 21v-2a4 4 0 0 0-4-4H8a4 4 0 0 0-4 4v2" />
    <circle cx="12" cy="7" r="4" />
  </svg>
);
```

I'm done... what's next!?

Nice! You've got a clean and modern sidebar!

Next week, we'll look into how to make those sidebar links actually go to different pages by introducing React Router!

Revision #7
Created 11 November 2025 06:34:57 by Emilio Cardillo Schrader
Updated 19 November 2025 19:14:38 by Emilio Cardillo Schrader