CURSED SOCIALS · HACK CLUB YSWS · GUIDE PART 1 OF 3

HTML & CSS Basics (without getting a headache)

Never coded a website before? Same, everyone starts here. By the end of Part 1, you'll have your first styled social media post running on your own laptop.



The goal: build a small social feed from scratch, ship it, and get a $5-$10 Hack Club grant.

Learning everything in one giant PDF is miserable, so we split this guide into three short parts:

- **Part 1 (you are here):** Setting up your files, basic HTML tags, colors, fonts, and spacing.
- **Part 2 (next):** Layouts with Flexbox and Grid (navbars, photo grids, and sidebars).
- **Part 3 (last):** Making like buttons actually work with JS and putting your site online.

1. Wait, what even are HTML, CSS, and JS?

Every website you use (Instagram, Discord, Reddit) is basically just three text files working together. You don't need a textbook definition. Just look at the skeleton:



Left: HTML builds the blocks. Middle: CSS paints it and adds style. Right: JS makes buttons actually do stuff.

- **HTML** puts stuff on the screen (a username, a photo, a post caption, a button).
- **CSS** stops it from looking like a boring 1995 Word doc (colors, fonts, round profile pics).
- **JS** makes things react when you click them. Ignore JS for now, we'll get to it in Part 3.

2. Your 2-minute setup

No fancy laptop or crazy downloads needed. Just do these four things:

1. Download **VS Code** (it's free and color-codes your typing so typos stand out).
2. Make a new folder on your Desktop called `my-social` (no spaces in the name).
3. Open that folder in VS Code and create two files inside it: `index.html` and `style.css`.
4. Find `index.html` in your computer's file explorer and double-click it. It opens right in your browser!

```
my-social/
├── index.html   <-- your page (keep this exact name!)
└── style.css    <-- your colors and layout
```

> **Always hit Ctrl+S (or Cmd+S on Mac) before refreshing your browser.** Literally everyone forgets to save at least once and stares at their screen confused for 10 minutes.

3. HTML is just labeling stuff

HTML uses **tags** inside pointy brackets to tell the browser what each piece of text is. Most tags have an opening tag and a closing tag with a slash `/`:

```
<p>my first post!!</p>
▲       ▲          ▲
open    text      close
```

Open `index.html` in VS Code, type a single exclamation mark `!`, and press **Tab**. VS Code writes the starter skeleton for you automatically. Keep it simple:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>My Social Site</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h1>CursedGram</h1>
  <p>hello world or whatever</p>
</body>
</html>
```

Only two parts matter right now: `<head>` holds behind-the-scenes stuff (your tab title and the link to `style.css`), and `<body>` holds everything people actually see on the page.

The only 7 HTML tags you need today

Honestly, don't try to memorize a huge list. Everyone googles tags anyway. These seven cover almost everything in a social feed:

Tag	What it's for on a social site
`<h1>`, `<h2>`, `<h3>`	Big to small headings (app name, username, section title).
`<p>`	Normal text like post captions, bios, and comments.
`<img src="cat.jpg" alt="cat">`	Shows a picture. Put the photo inside your folder! (No closing tag needed.)
`<a href="https://...">`	Clickable link to another site or another page.
`<button>like</button>`	A real clickable button (we'll wire it up in Part 3).
`<ul>` + `<li>`	Bullet list (great for hashtags, menus, or friend lists).

Tag	What it's for on a social site
`<div class="post">`	An invisible box that groups a post's title, caption, and button together.

Try this right now

Replace what's inside your `<body>` tags with one simple post, save, and refresh your browser:

```
<h1>CursedGram</h1>

<div class="post">
  <h3>@you</h3>
  <p>first post on a site i actually coded myself.</p>
  <button class="like-btn">like</button>
</div>
```

Yeah, it looks super plain right now (black Times New Roman text on a white page). That means it's working! Now let's fix the looks with CSS.

4. Adding CSS (making it actually look good)

Make sure this line is inside your `<head>` tag in `index.html` so HTML knows where your CSS lives:

```
<link rel="stylesheet" href="style.css">
```

In `style.css`, you pick something from your HTML, open curly braces `{ }`, and tell it how to look. Always end each line with a semicolon `;`.

```
body {
  background-color: #f4ecdc;
  color: #24211f;
  font-family: Arial, sans-serif;
}
```

How to target the exact thing you want

- **Plain tag name** (like `p` or `button`) styles every single one of those on the page.
- **Dot + class name** (like `.post`) styles anything with `class="post"`. You'll use classes 90% of the time!
- **`:hover`** (like `.like-btn:hover`) changes the look only while your mouse is sitting on top of it.

Quick cheat code: round profile pictures

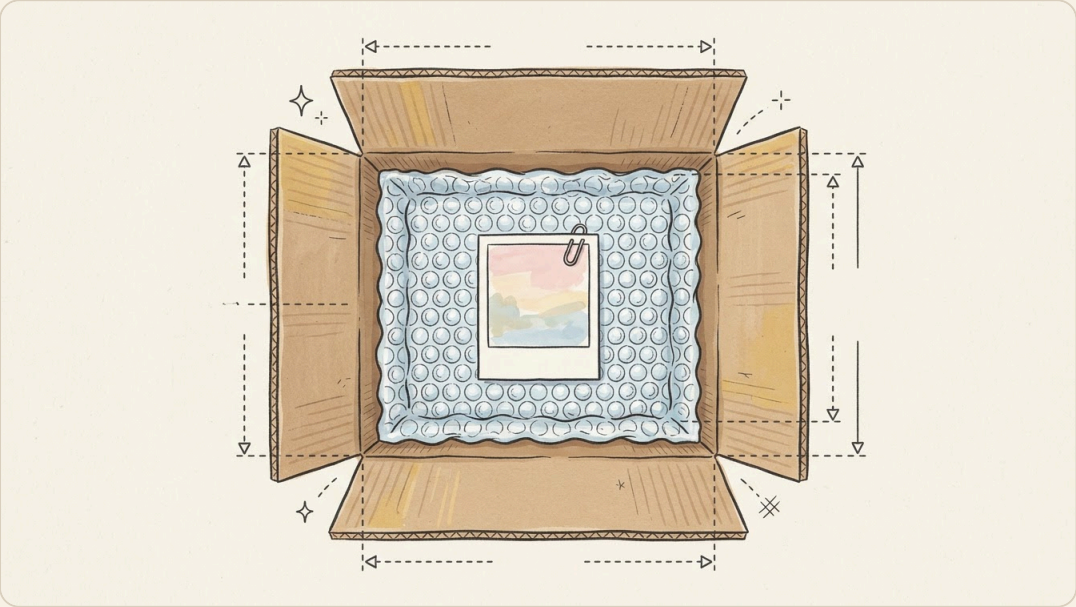
Want an Instagram-style circle avatar instead of a pointy square? Give your image the same width and height, set `border-radius: 50%`, and add `object-fit: cover` so people's faces don't get squished:

```
.avatar {
  width: 48px;
  height: 48px;
  border-radius: 50%;
  object-fit: cover;
}
```

> **Where do you get color codes like `#ec3750`?** Literally just google "color picker", drag the slider to a color you like, and copy the HEX code.

5. The Box Model (the secret to spacing)

Here is the one CSS concept that makes everything click. To a browser, **every single thing on your page is a cardboard box** (even a sentence or a circle).



Polaroid = Content · Bubble wrap = Padding · Cardboard walls = Border · Space outside = Margin.

- **Content (the Polaroid):** Your actual text or photo in the middle.
- **Padding (the bubble wrap):** Soft space *inside* the box so your text isn't squished against the edges.
- **Border (the cardboard walls):** The outline around your post card or button.
- **Margin (the empty space outside):** Pushes other boxes away so two posts aren't glued together.

Let's turn that boring `.post` div into an actual post card using all four:

```
* {
  box-sizing: border-box; /* keeps box widths from acting weird */
}

.post {
  background: #ffffff;
  border: 2px solid #24211f;
  border-radius: 12px;
  padding: 16px;          /* bubble wrap inside the card */
  margin: 20px auto;      /* space outside + auto centers it! */
  max-width: 420px;
}
```

6. Your Part 1 Mini-Project

Paste this into `style.css` to style the post you wrote in Step 3. Save both files and refresh your browser:

```
* {
  box-sizing: border-box;
}

body {
  background: #f4ecdc;
  color: #24211f;
  font-family: Arial, sans-serif;
  padding: 24px;
}

h1 {
  text-align: center;
}

.post {
  background: #ffffff;
  border: 2px solid #24211f;
  border-radius: 14px;
  padding: 18px;
  max-width: 420px;
  margin: 18px auto;
}

.like-btn {
  background: #ec3750;
  color: #ffffff;
  border: 2px solid #24211f;
  border-radius: 999px;
  padding: 6px 16px;
  font-weight: bold;
  cursor: pointer;
}

.like-btn:hover {
  background: #24211f;
}
```

Mess around with it (best way to learn)

- Copy and paste the whole `<div class="post">...</div>` block twice so you have a 3-post feed.
- Add a real photo inside one post using `<img src="yourpic.jpg" style="width: 100%; border-radius: 8px;">`.
- Change the background and button colors to match your favorite app (Discord dark gray, Tumblr navy, or cursed neon green).

Wait, why did mine break? (Quick fixes)

- **Nothing changed when I refreshed:** Hit Ctrl+S in VS Code on *both* files first.
- **Zero CSS is showing up:** Check that `style.css` is in the same folder as `index.html` and spelled the exact same way.
- **One CSS line is being ignored:** Check the line right above it for a missing semicolon `;` or missing closing brace `}`.
- **My image is a broken icon:** Check the file ending (`.jpg` vs `.png`) and keep filenames lowercase with no spaces.

> **You just finished Part 1!**
>
> You officially built and styled your first webpage from scratch. Drop a screenshot in **#cursed-socials** on Slack if you want to show it off or ask for help. Up next in **Part 2**: Flexbox and Grid so you can build navbars, side-by-side avatars, and Instagram-style photo grids.