**A screenshot of a computer program

AI-generated content may be incorrect.**

**A screen shot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**🔹 1. What is Tailwind CSS?**

* Tailwind CSS is a **utility-first CSS framework**.
* Instead of writing custom CSS classes, you directly apply **predefined utility classes** (like p-4, text-center, bg-blue-500) to HTML/JSX elements.
* This makes styling faster, more consistent, and removes the need to switch between CSS files and components.

**🔹 2. If we already have CSS, why do we need Tailwind?**

* Plain CSS → requires writing styles in .css files, managing class names, and keeping things DRY.
* Tailwind → gives you **ready-to-use utility classes**, avoids naming confusion, and reduces time.
* Benefits:
  + Faster development.
  + Consistent design system.
  + Responsive and theming support out of the box.

**🔹 3. What is Bootstrap? How is it different from Tailwind?**

**Bootstrap**:

* A **component-based CSS framework** → comes with pre-designed components (navbar, cards, modals).
* You mostly use **ready templates**.

**Tailwind**:

* A **utility-first framework** → gives small utility classes to build **your own design**, no fixed look.

👉 **Key difference**:

* Bootstrap → "Here’s a prebuilt button, use it."
* Tailwind → "Here are classes, build your button exactly how you want."

**🔹 4. Explore all the ways of writing CSS**

1. **Inline CSS** → <div style={{ color: "red" }}>Hello</div>
2. **Internal CSS** → writing <style> tag inside HTML.
3. **External CSS** → separate .css files imported into the project.
4. **CSS Modules** → scoped CSS (e.g., Button.module.css).
5. **Styled Components** → CSS-in-JS (using libraries).
6. **Utility-first frameworks** → Tailwind CSS.
7. **Framework CSS** → Bootstrap, Bulma, etc.

**🔹 5. How do we configure Tailwind?**

* After installing Tailwind (npm install tailwindcss postcss autoprefixer), you initialize it:

A black screen with white text

AI-generated content may be incorrect.

* This creates tailwind.config.js → where you configure paths, themes, extensions, plugins.
* Then you add Tailwind directives in your CSS:

A black background with white text

AI-generated content may be incorrect.

**🔹 6. Keys in tailwind.config.js**

* **content** → tells Tailwind where to scan files for class names (like ["./src/\*\*/\*.{js,jsx,ts,tsx}"]).
* **theme** → contains default styling (colors, fonts, spacing).
* **extend** → used to add custom values without overriding defaults (e.g., add a new color).
* **plugins** → to add extra Tailwind features (like forms, typography).

**🔹 7. Why do we have .postcssrc file?**

* Tailwind is built on **PostCSS** (a CSS processing tool).
* .postcssrc configures PostCSS to tell it which plugins to use → e.g., Tailwind and Autoprefixer.
* Autoprefixer ensures CSS works across browsers by automatically adding vendor prefixes (-webkit-, -moz-).

✅ **Interview-ready short answer for “Tailwind CSS”**:

Tailwind is a utility-first CSS framework that provides small, reusable classes to style components directly in markup. Unlike Bootstrap, which gives prebuilt UI components, Tailwind gives flexibility to design custom UIs faster. It integrates with PostCSS, configured through tailwind.config.js and .postcssrc, making it powerful for modern responsive design.