Tailwind vs Bootstrap vs CSS-in-JS: A Comparative Study

Modern frontend development involves several choices when it comes to styling: utility-first frameworks, component-based styles, and traditional class-based approaches. This writeup compares **Tailwind CSS**, **Bootstrap**, and **CSS-in-JS** across several dimensions: philosophy, developer experience, performance, customization, and use cases.

🛠️ 2. Developer Experience

✅ Tailwind CSS

- No need to leave HTML/JSX; style using utility classes.
- Autocompletion support in IDEs (with plugins).
- Easy to apply conditional styling (with variants).

✅ Bootstrap

- Quick to prototype using pre-built components.
- Great documentation and consistent design patterns.
- Requires custom CSS/overrides for deep customization.

✅ CSS-in-JS

- Full dynamic styling capabilities using JS/TS.
- Can access props, themes, and runtime logic in styles.
- Requires Babel or build tool support (emotion, styled-components).

🎨 4. Customization and Theming

✅ Tailwind CSS

- Uses `tailwind.config.js` for central configuration.
- Supports custom themes, color palettes, breakpoints.

✅ Bootstrap

- Customizable using SCSS variables.
- Theming system is comprehensive but can be verbose.

✅ CSS-in-JS

- Ultimate flexibility: you can define themes in JS and use logic to change styles dynamically.

📚 Use Cases

Use Case	Recommended Approach
Design systems or UI libraries	Tailwind or CSS-in-JS
Prototyping or admin dashboards	Bootstrap
Themed apps with dynamic styles	CSS-in-JS
Performance-critical SPAs	Tailwind