Fullstack Development

UI Libraries

Why?

> "Normally-expected" UI design and UI functionality take a lot of work to make.

Dropdown

- DIY
- Library
- WAI-ARIA Standard

We want to

- Spend more time on business logic, not UI design and UI logic.
- Establish common framework for collaboration.
- Have maintainable and extensible system.

Types

Library Type	CSS/Theme	Functionality
"Style"	✅	❌
"Component"	✅	✅
"Headless-Component"	❌	✅
"Anti-Component"	✅	✅

Consideration

- Ease of use
  - How much code do I need to write?
- Customizability
  - How much can I change down the line?
- Inclusiveness
  - Do I need to install extra things?
- Look and feel
  - Does it make my app look generic?

Disclaimer

- The choices of UI libraries can be opionated.
- At the end of the day, you must choose your own path.
- Anything marked with 🦀 is my opinion. You don't have to take it.

"Style" libraries

"Style" lirary

- *No library*
- `Styled-Components`
- `Tailwind CSS`

No library

- `globals.css`
  - Not recommended
- Inline CSS
  - Can be used as a quick fix
- CSS Module

CSS Module

- Good
  - No name conflicts
  - Separation between style and content
- Bad
  - 🦀 Separation between style and content

Styled Component

> Styling your way with speed, strong typing, and flexibility.

- Styled components are a CSS-in-JS tool that bridges the gap between components and styling.
- Installation note
  - For NextJS, follow the instructions.

Styled Component

- Good
  - CSS-in-JS
  - Full customizability from pure CSS
- Bad
  - 🦀 CSS-in-JS in separate file (anyway)
  - 🦀 Hard to maintain.

Tailwind CSS

> Rapidly build modern websites without ever leaving your HTML

- A utility-first CSS framework
  - Provides several of opinionated, single-purpose utility classes that you can use directly inside your markup.
- Tailwind founder's keynote

Tailwind CSS

- Good
  - Styling in markup
  - 🦀 Less customizability than pure CSS
  - 🦀 Maintainable: *(3 months from now, I can come back and quickly adjust style.)*
- Bad
  - Need to learn syntax
  - Ugly markup
    - Fix: Use VSCode extension or wrap markup in components.

"Component" lirary

"Component" lirary

- `Bootstrap React`
- `Mantine`
- Others
  - `Material UI`
  - `Ant Design`

Bootstrap React

> The most popular front-end framework, rebuilt for React.

Bootstrap React

- Good
  - Lots of premade components
  - Provided CSS bootrap classes
- Bad
  - Hard to customize
  - 🦀 Bootstrap classes are verbose.
  - 🦀 Look "Bootstrap"

Mantine

> Build fully functional accessible web applications faster than ever – Mantine includes more than 100 customizable components and 50 hooks to cover you in any situation

Mantine

- Good
  - Comprehensive lists of components (seriously)
  - Comes with useful hooks
  - 🦀 No CSS class
    - You can use other CSS style libraries (e.g. `Tailwind CSS`)
- Bad
  - 🦀 some frictions when trying to deeply customize.

"Headless-Component" libraries

"Headless-Component" libraries

- `Radix UI`
- `Headless UI`

Radix UI

> Unstyled, accessible components for building high-quality design systems and web apps in React.

Radix UI

- Good
  - Fully customization (design)
  - Excellent functionality
- Bad
  - Too much code for simple stuffs

"Anti-Component" libraries

"Anti-Component" libraries

> This is NOT a component library. It's a collection of re-usable components that you can copy and paste into your apps.

- `shadcn/ui`
- `Flowbite`

shadcn/ui

> Re-usable components built using Radix UI and Tailwind CSS.

shadcn/ui

- Good
  - Deeply customizable
  - 🦀 Use Tailwind
  - Lot of components
  - 🦀 Integrate with `React Hook Form` and `Tanstack Table`
- Bad
  - Need to learn Tailwind
  - Not as many components as `Mantine`

Pick

- Interactive apps
  - `Mantine`
  - `shadcn/ui`
- Brochure apps
  - `Tailwind CSS`

What is your pick?