React folders structures

When organizing a React project, the folder structure plays a crucial role in maintainability, scalability, and developer experience. React doesn't enforce a specific structure, allowing flexibility to choose one that best fits your project's needs. Here's an overview of common folder structures

1. Flat Structure (Ideal for Small Projects)

Start with a simple structure where all files reside directly under the src folder.

2. Feature-Based Structure (Recommended for Medium to Large Projects)

Organize files by feature or domain, grouping related components, hooks, styles, and tests together.

3. Type-Based Structure (Traditional Approach)

Group files by their type, such as components, hooks, and utilities.

What is folder?

A **folder** (also called a **directory**) is like a container that holds **files** and/or **other folders**.

Think of it like a **real-world folder** in a filing cabinet. Just like you use a folder to organize paper documents, you use folders in a computer to organize code, images, or any kind of files.

In coding / React projects:

Folders help keep your project **organized and manageable**.

For example, in a React app you might have a folder called components/ where you keep all your React components.

Why use folders?

- To keep related files together
- To make your project easier to navigate
- To avoid one giant messy list of files

Types of react folder structures

1.basic structure

2. Type-Based Structure

3. Feature-Based / Domain-Driven Structure

4. Atomic Design (Optional Layer)