**Routing**

Next.js has a file-system based routing system

URLs you can access in your browser are determined by how you organize you files and folders in your code.

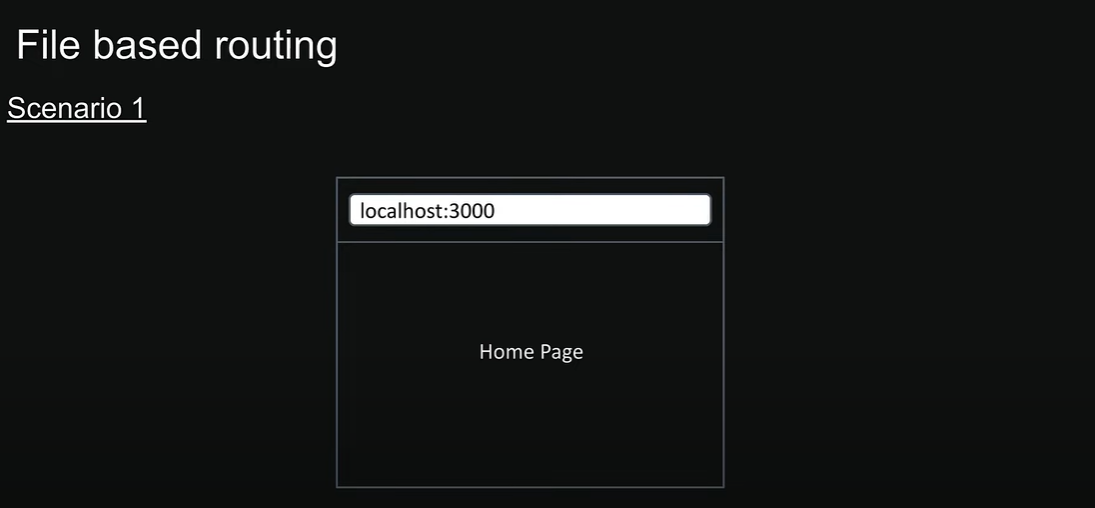
Routing conventions

1. All routes must live inside the app folder
2. Route files must be named either page.js or page.tsx
3. Each folder represents a segment of the URL path

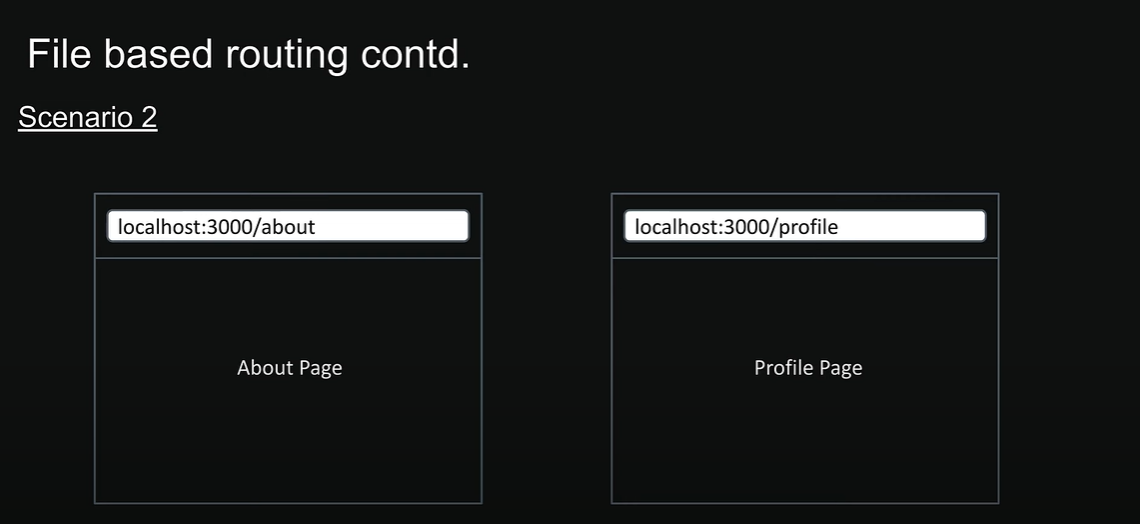
When these conventions are followed, the file automatically becomes available as a route.

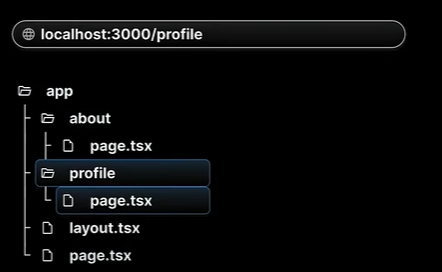
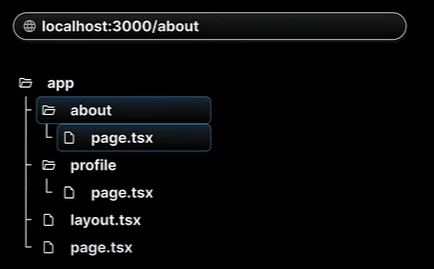
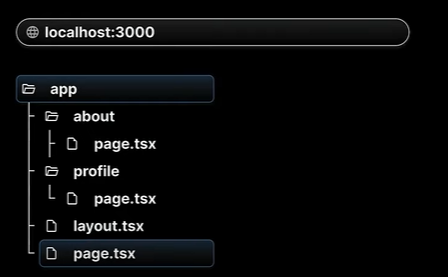
**File based routing ( Định tuyến dựa trên tập tin )**

Scenario 1:

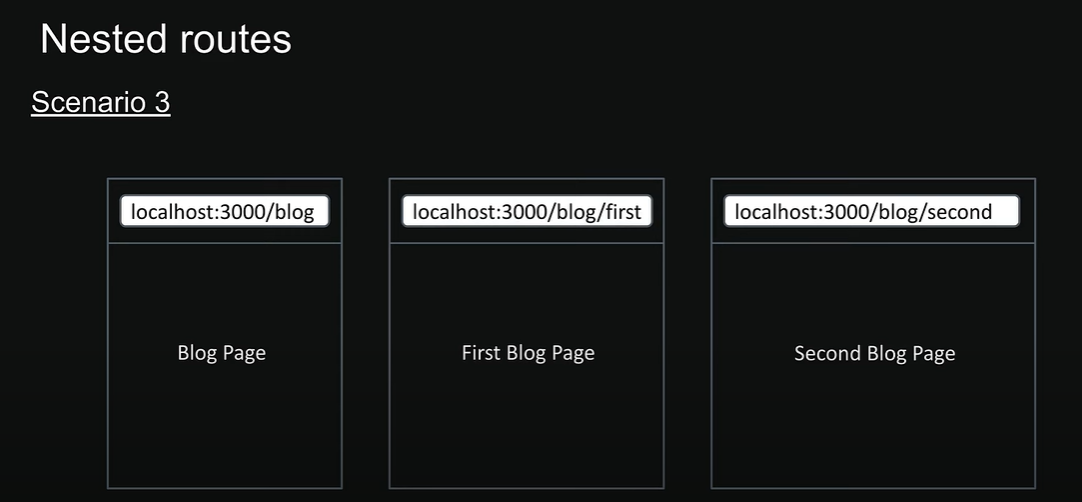


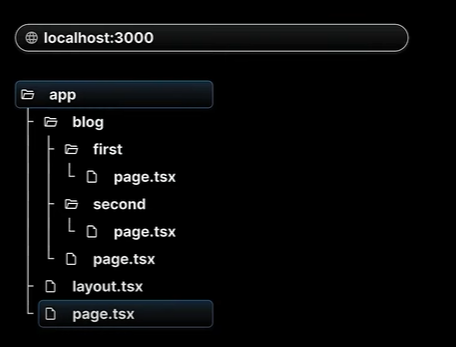
Scenario 2:



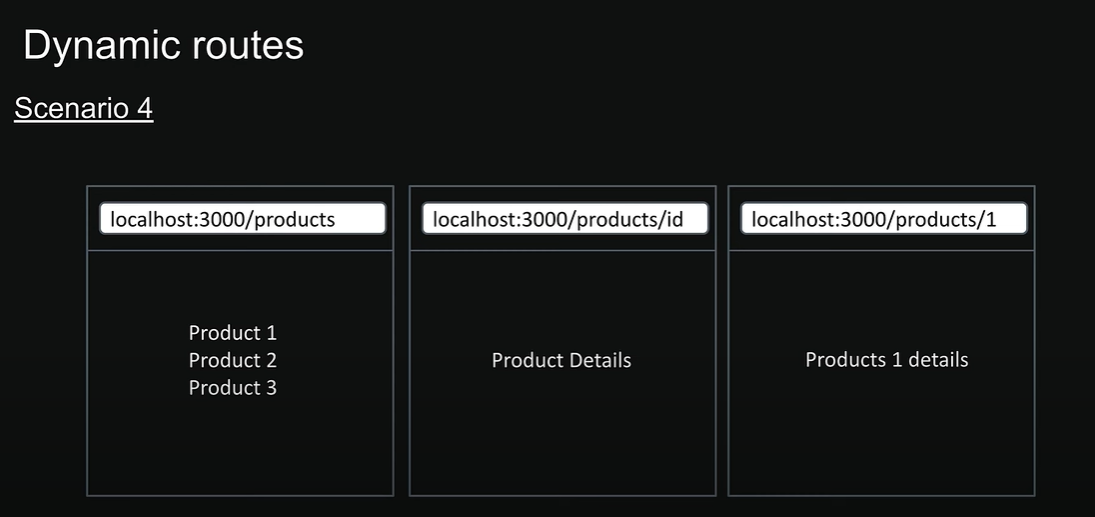


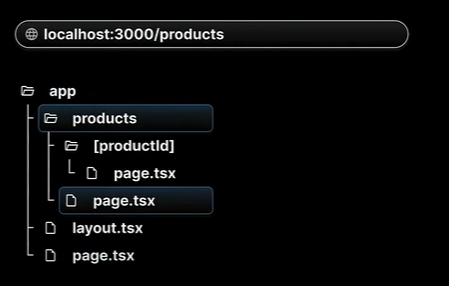
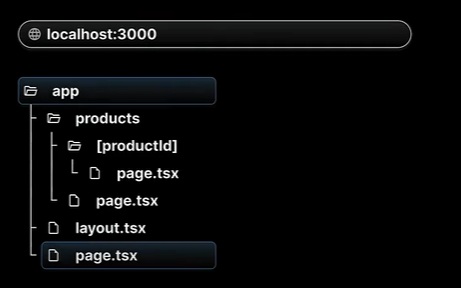
**Nested Routes ( Định tuyến lồng nhau )**

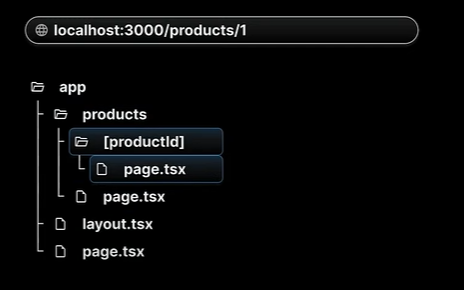




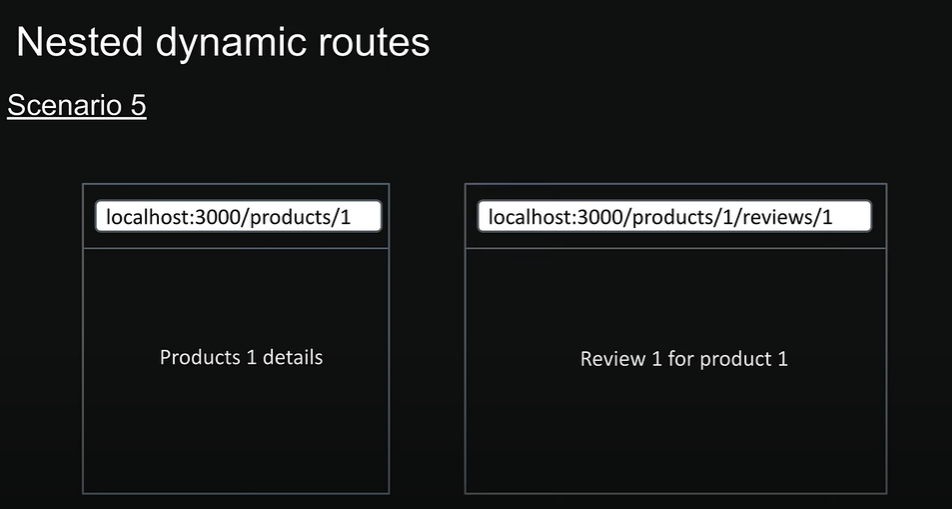
**Dynamic Routes ( Định tuyến động )**

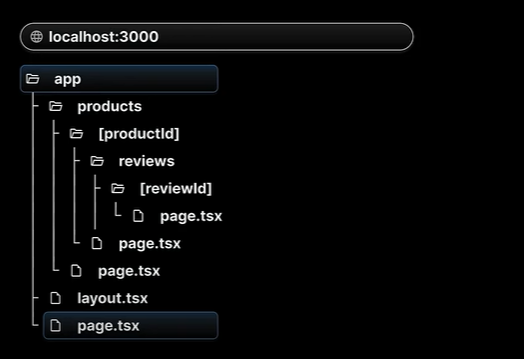


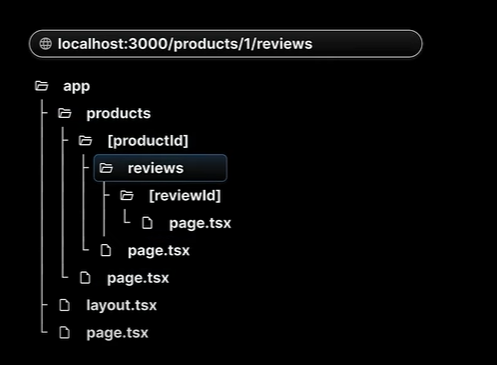


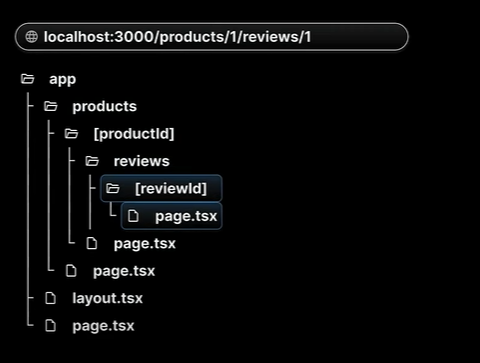


**Nested Dynamic Routes ( Định tuyến động lồng nhau)**

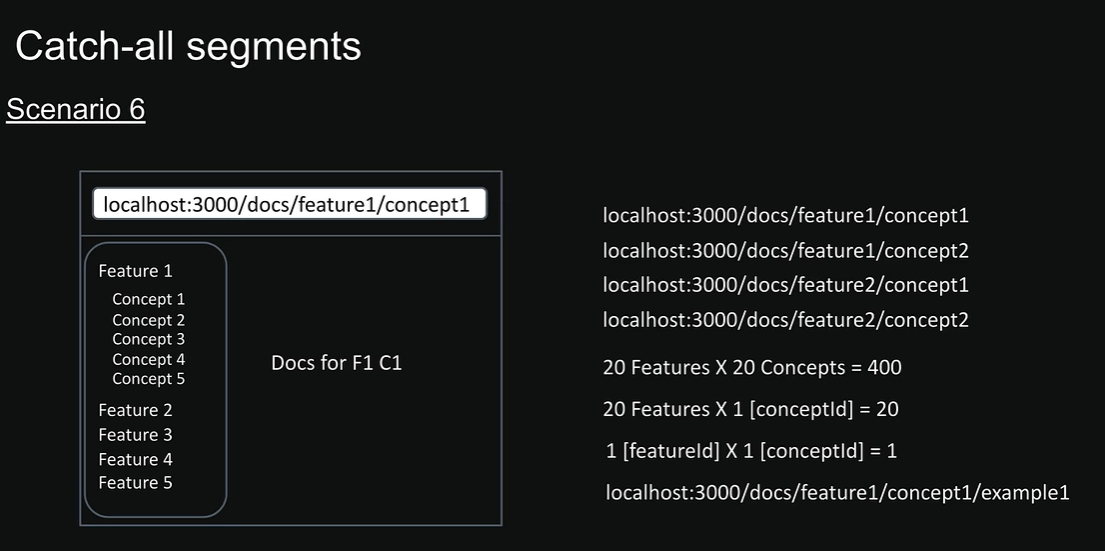


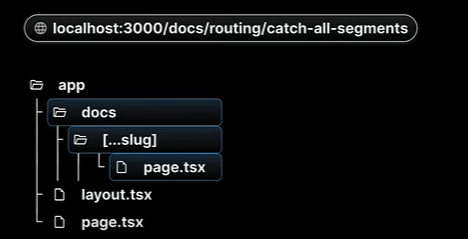
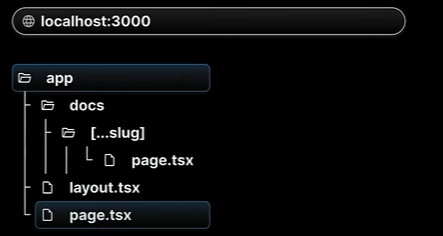






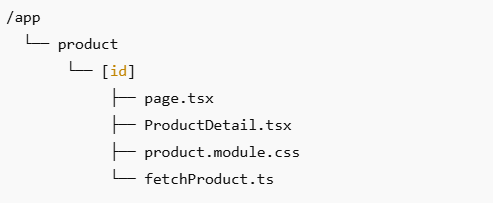
**Catch-all segments ( Bắt nhiều cấp định tuyến thành một )**





**File colocation**

* đặt các file logic, UI, CSS,... liên quan đến một page/route/component vào cùng một folder.
* project sạch, dễ quản lý, và phù hợp với cấu trúc module hóa hiện đại.



**Private Folders**

* A way to tell Next.js, “Hey, this folder is just for internal stuff – don’t include it in the routing system.”
* The folder and all its subfolders are excluded from routing.
* Add an underscore at the start of the folder name ( \_folderName )

Private folders are super useful for a bunch of things:

* Keeping you UI logic separate from routing logic
* Having a consistent way to organize internal files in your project
* Making it easier to group related files in your code editor
* Avoiding potential naming conflicts with future Next.js file naming conventions

If you actually want an underscope in your URL, use “%5F” instead. That’s just the URL – encoded version of an underscore

**Route Groups**

Lets us logically organize our routes and project files without impacting the URL structure

Let’s implement authentication routes:

* Register
* Login
* Forgot password

**Routing Metadata**

**Routing Metadata** trong Next.js là các thông tin bổ sung mà bạn có thể khai báo để mô tả route đó — ví dụ như **tiêu đề trang (title), mô tả (description), favicon, thẻ Open Graph, robots.txt,...** dùng cho SEO, chia sẻ mạng xã hội, và trải nghiệm người dùng tốt hơn.

The Metadata API in Next.js is a powerfull feature that lets us define metadata for each page

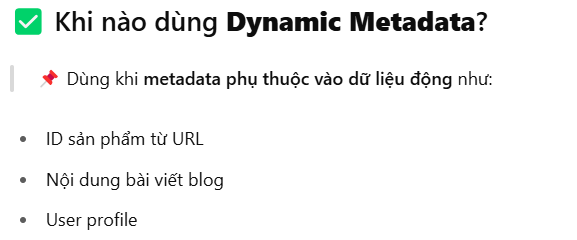
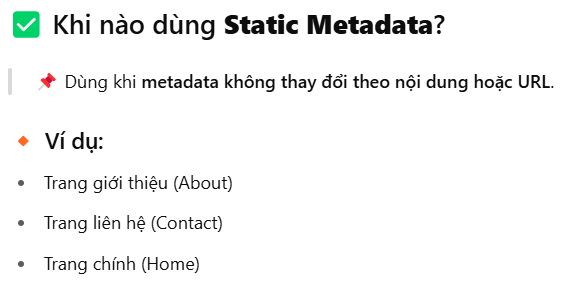
Metadata ensures our content looks great when it’s shared or indexed by search engines

Two ways to handle metadata in layout.tsx or page.tsx files:

1. Export a static ***metadata*** object
2. Export a dynamic ***generateMetadata*** function

**Configuring metadata**

* Both layout.tsx and page.tsx can export metadata. Layout metadata applies to all its page, while page metadata is specific to that page
* Metadata follows a top-down order, starting from the root level
* When metadata exists in mutiple places along a route, they merge together, with page metadata overriding layout metadata for matching properties.



| **Đặc điểm** | **Static Metadata** | **Dynamic Metadata** |
| --- | --- | --- |
| Khai báo ở đâu | export const metadata = {} | export async function generateMetadata() |
| Dữ liệu đầu vào | Không cần | Nhận params, searchParams, v.v. |
| Tối ưu SEO | Rất tốt | Tốt nếu fetch nhanh và đúng cách |
| Phù hợp với | Trang cố định | Trang chi tiết, URL động |