Let's walk through **Next.js basics using the App Router**, which is the **newer routing system** introduced in **Next.js 13+** (recommended as of Next.js 14+). This replaces the older **Pages Router** with a **file-based nested routing system** using a special `app/` directory.

□ 1. Project Setup

First, create a new Next.js project with the App Router:

```
npx create-next-app@latest my-next-app
```

When prompted:

- □ Enable **App Router** (important)
- □ You can skip Tailwind, etc., for now unless you want them

```
cd my-next-app
npm run dev
```

□ 2. Project Structure (App Router)

The `app/` directory drives the routing:

```
my-next-app/
├── app/
│   ├── page.tsx          ← Homepage (route: "/")
│   ├── about/
│   │   └── page.tsx      ← About page (route: "/about")
│   ├── layout.tsx        ← Shared layout (applies to all pages)
├── public/
├── styles/
├── next.config.js
└── package.json
```

□ 3. Creating Pages

□ `app/page.tsx`

```
export default function HomePage() {
  return <h1>Welcome to the Home Page</h1>;
}
```

□ `app/about/page.tsx`

```
export default function AboutPage() {
  return <h1>About Us</h1>;
```

```
}
```

Now you have:

- `/` → HomePage
- `/about` → AboutPage

□ 4. Layouts

Create a shared layout for your pages.

□ **app/layout.tsx**

```
export const metadata = {
  title: 'My App',
  description: 'Using App Router in Next.js',
};

export default function RootLayout({ children }: { children:
React.ReactNode }) {
  return (
    <html lang="en">
      <body>
        <nav>
          <a href="/">Home</a> | <a href="/about">About</a>
        </nav>
        <main>{children}</main>
      </body>
    </html>
  );
}
```

This layout wraps **all pages** inside the `app/` directory.

□ 5. Routing Concepts in App Router

□ Nested Routes

Just create folders inside `app/`:

```
app/
└── blog/
    ├── page.tsx          ← `/blog`
    └── post/
        └── page.tsx      ← `/blog/post`
```

□ Dynamic Routes

Use `[param]` syntax:

```
app/
└── blog/
    └── [slug]/
        └── page.tsx      ← `/blog/hello-world`
// app/blog/[slug]/page.tsx
type Props = {
  params: { slug: string }
};

export default function BlogPost({ params }: Props) {
  return <h1>Post: {params.slug}</h1>;
}
```

□ 6. Client vs Server Components

By default, components in `app/` are **Server Components**.

To make a component run on the client (for interactivity), add `"use client"` at the top:

```
"use client";

import { useState } from "react";

export default function Counter() {
  const [count, setCount] = useState(0);
  return <button onClick={() => setCount(count + 1)}>Clicked {count}
times</button>;
}
```

□ 7. Fetching Data (Server-Side)

□ Static (default):

```
export async function generateStaticParams() {
  return [{ slug: "post-1" }, { slug: "post-2" }];
}

export async function getData(slug: string) {
  const res = await fetch(`https://api.example.com/posts/${slug}`);
  return res.json();
}

export default async function PostPage({ params }: { params: { slug: string
} }) {
  const post = await getData(params.slug);
  return <div>{post.title}</div>;
}
```

□ Dynamic with revalidation:

```
export const revalidate = 60; // revalidate every 60 seconds
```

□ 8. Summary of Key Files

File/Folder	Purpose
`app/page.tsx`	Home page (`/`)
`app/about/page.tsx`	About page (`/about`)
`app/layout.tsx`	Root layout for all pages
`app/[slug]/page.tsx`	Dynamic route for `/[slug]`
`"use client"`	Enables client-side interactivity

□ Bonus: Add Navigation with `next/link`

```
import Link from "next/link";

<Link href="/">Home</Link>
<Link href="/about">About</Link>
```

Here's the **full working Next.js App Router example in plain JavaScript** — no TypeScript, no extras, just **JavaScript + App Router + basic features**.

□ What You'll Get

- □ JavaScript version (no TypeScript)
- □ App Router structure
- □ Pages: `/`, `/about`, `/blog`, `/blog/[slug]`
- □ Shared layout
- □ Navigation using `next/link`
- □ Client-side counter
- □ Simple mock data fetching
- □ Minimal styling

□ 1. Create the App (with App Router)

```
npx create-next-app@latest my-next-js-app
```

When prompted:

- □ Enable **App Router**
- □ Disable TypeScript
- □ Skip Tailwind / ESLint unless you want it
- □ Enable `src/` directory (optional but recommended)

□ 2. Project Folder Structure

Here's what the final structure will look like:

```
my-next-js-app/
└── app/
    ├── layout.js
    ├── page.js                 ← Home page
    ├── about/
    │   └── page.js             ← /about
    ├── blog/
    │   ├── page.js             ← /blog
    │   └── [slug]/
    │       └── page.js         ← /blog/[slug]
    └── components/
        └── Counter.js          ← client component
└── public/
└── app/globals.css            ← optional global styles
```

✍ 3. Actual Code Files

app/layout.js

```
import Link from 'next/link';
import './globals.css'; // Optional styling

export const metadata = {
  title: 'Next.js JS App',
  description: 'Built with App Router in JavaScript',
};

export default function RootLayout({ children }) {
  return (
    <html lang="en">
      <body>
        <nav style={{ marginBottom: '20px' }}>
          <Link href="/">Home</Link> |{' '}
          <Link href="/about">About</Link> |{' '}
          <Link href="/blog">Blog</Link>
        </nav>
        <main>{children}</main>
      </body>
    </html>
  );
}
```

---

app/page.js – Home Page

```
import Counter from './components/Counter';

export default function HomePage() {
  return (
    <div>
      <h1>Welcome to the Home Page</h1>
      <p>This is built using the App Router in JavaScript.</p>
      <Counter />
    </div>
  );
}
```

---

app/about/page.js – About Page

```
export default function AboutPage() {
  return (
    <div>
      <h1>About Page</h1>
      <p>This is the about page of our app.</p>
    </div>
  );
}
```

📄 `app/blog/page.js` – Blog List

```
import Link from 'next/link';

const posts = [
  { slug: 'hello-world', title: 'Hello World' },
  { slug: 'nextjs-routing', title: 'Next.js App Router' },
];

export default function BlogPage() {
  return (
    <div>
      <h1>Blog</h1>
      <ul>
        {posts.map((post) => (
          <li key={post.slug}>
            <Link href={`/blog/${post.slug}`}>{post.title}</Link>
          </li>
        ))}
      </ul>
    </div>
  );
}
```

📄 `app/blog/[slug]/page.js` – Dynamic Blog Post

```
const posts = {
  'hello-world': {
    title: 'Hello World',
    content: 'This is the first blog post.',
  },
  'nextjs-routing': {
    title: 'Next.js App Router',
    content: 'This explains how to use the new routing system in Next.js.',
  },
};

export default function BlogPost({ params }) {
  const post = posts[params.slug];

  if (!post) {
    return <h1>Post not found</h1>;
  }

  return (
    <div>
      <h1>{post.title}</h1>
      <p>{post.content}</p>
    </div>
  );
}
```

✅ `app/components/Counter.js` – Client-Side Counter

```
'use client';

import { useState } from 'react';

export default function Counter() {
  const [count, setCount] = useState(0);

  return (
    <div style={{ marginTop: '20px' }}>
      <button onClick={() => setCount(count + 1)}>
        You clicked {count} times
      </button>
    </div>
  );
}
```

✅ (Optional) `app/globals.css`

```
body {
  font-family: sans-serif;
  padding: 20px;
}
nav a {
  margin-right: 10px;
  text-decoration: none;
  color: blue;
}
```

Make sure it's imported in `layout.js`:

```
import './globals.css';
```

✅ 4. Run the App

```
npm run dev
```

Open your browser at: http://localhost:3000

Try:

- `/`
- `/about`
- `/blog`
- `/blog/hello-world`
- `/blog/nextjs-routing`

□ Done! You Now Have:

- □ Working Next.js App Router project (JS only)
- □ Static & dynamic pages
- □ Layout + navigation
- □ Client component with state
- □ No TypeScript