

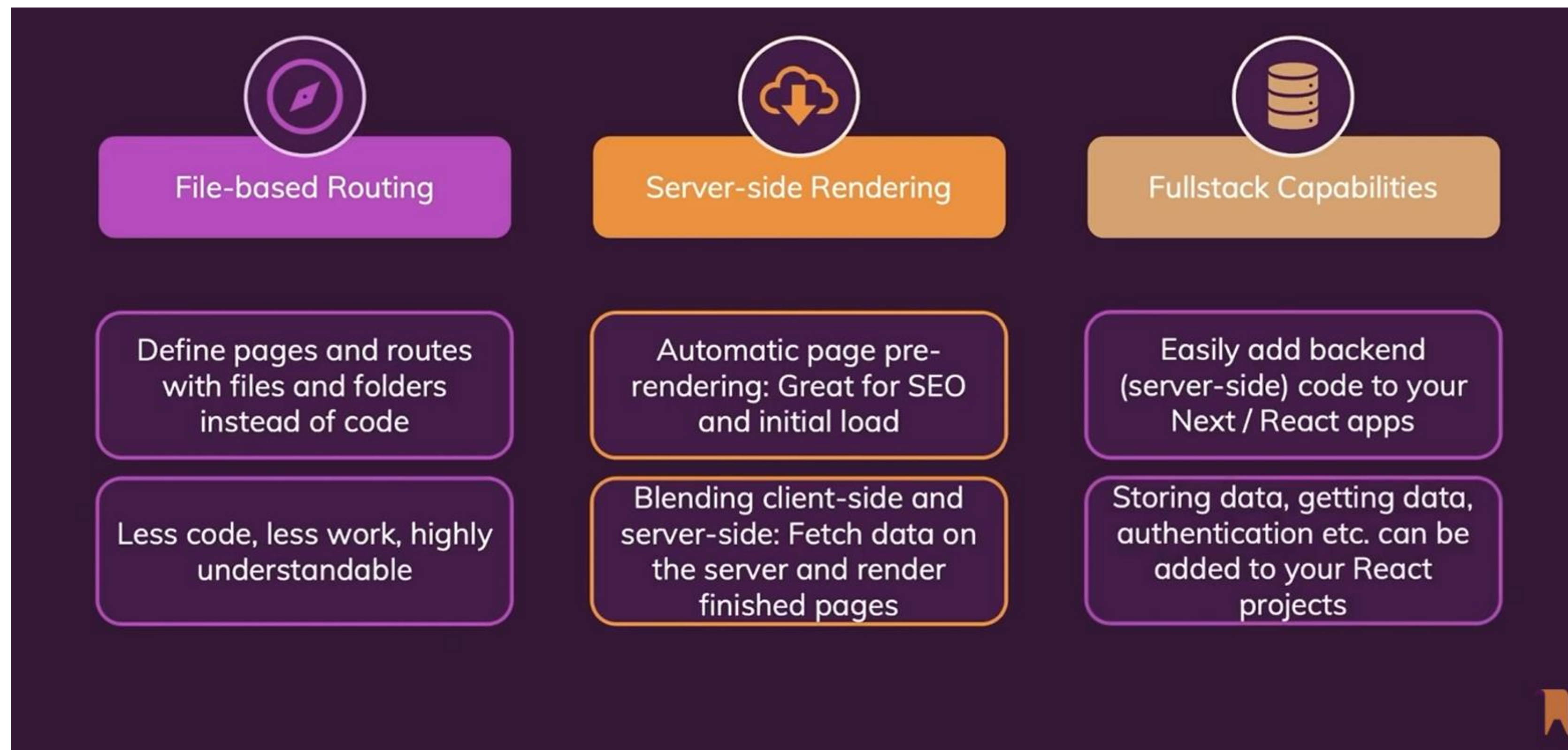
Next JS

What is Next.js?

- React Framework that allows us to build web applications using JavaScript and React more quickly
- Better to use to create static and server-rendered applications
- Benefits of server-rendered apps:
 - Faster initial page rendering
 - Better search engine optimisation

NEXT.js

Next js Key features and benefits



Create an App with Next.js

- Open your command line interface and create a project with a meaningful name.

E.g. `npx create-next-app my-first-next-app`

- Navigate to the directory you have just created and open it by your code editor . E.g. `cd my-first-next-app`
- Run the development server by typing `npm run dev`

Adding Links to Pages

- To support client-side navigation we use Next.js' Link API

```
import Link from 'next/link'

export default () => (
  <div>
    <Link href="/index">
      <a>Home<a/>
    </Link>
    <p>This is the About page</p>
  </div>
)
```

Adding Images and Other Static Resources

- In folder called “static” in your project root directory
- From your code, you can then reference those files with /static/ URLs

```
export default () => 
```


Styling Next.js Apps

- You can customize for each file or in globals.css

```
Home.module.css > .btn
.title {
  color: #333;
  padding-bottom: 20px;
  text-align: center;
}
.text {
  color: #777;
}
.btn {
  display: block;
  width: 150px;
  padding: 8px 0;
  margin: 20px auto;
  background: #4979ff;
  border-radius: 4px;
  color: white;
  text-align: center;
}
```

```
index.js > Home
import Head from 'next/head'
import Navbar from '../comps/Navbar'
import Footer from '../comps/Footer'
import styles from '../styles/Home.module.css'
import Link from 'next/link'

export default function Home() {
  return (
    <div>
      <h1 className={styles.title}>Homepage</h1>
      <p className={styles.text}>Lorem, ipsum dolor sit amet consectetur adipisicing elit. Provident
      <p className={styles.text}>Lorem, ipsum dolor sit amet consectetur adipisicing elit. Provident
      <Link href="/ninjas">
        <a className={styles.btn}>See Ninja Listing</a>
      </Link>
    </div>
  )
}
```

404 page with redirecting the user

```
import Link from 'next/link'
import { useEffect } from 'react'
import { useRouter } from 'next/router'

const NotFound = () => {
  const router = useRouter();

  useEffect(() => {
    setTimeout(() => {
      // router.go(1)
      router.push('/');
    }, 3000)
  }, [])

  return (
    <div className="not-found">
      <h1>Oooops...</h1>
      <h2>That page cannot be found.</h2>
      <p>Go back to the <Link href="/"><a>Homepage</a></Link></p>
    </div>
  );
}

export default NotFound;
```

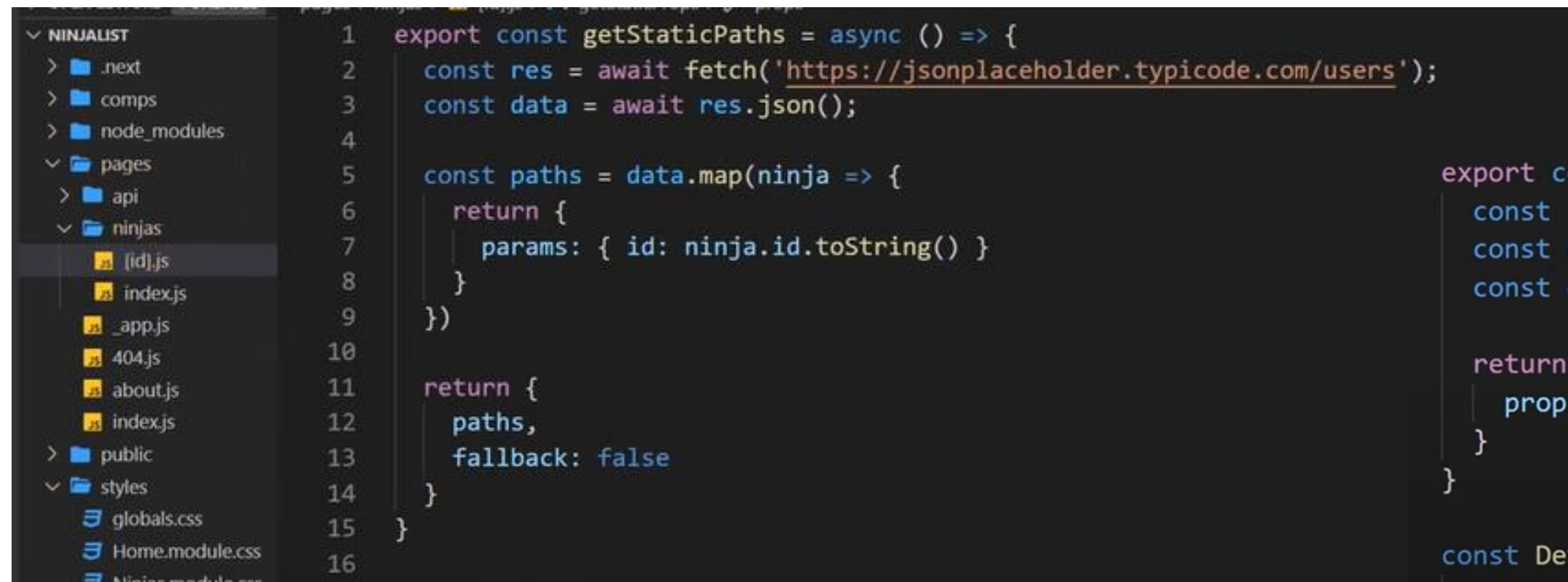

Fetching Data (getStaticProps)

- <https://nextjs.org/docs/basic-features/data-fetching>
- If you export an async function called getStaticProps from a page, Next.js will pre-render this page at build time using the props returned by getStaticProps.

```
export const getStaticProps = async () => {  
  
  const res = await fetch('https://jsonplaceholder.typicode.com/users');  
  const data = await res.json();  
  
  return {  
    props: { ninjas: data }  
  }  
}  
  
const Ninjas = ({ ninjas }) => {  
  return (  
    <div>  
      <h1>All Ninjas</h1>  
      {ninjas.map(ninja => (  
        <div key={ninja.id}>  
          <a className={styles.single}>  
            <h3>{ninja.name}</h3>  
          </a>  
        </div>  
      ))}  
    </div>  
  )  
}
```

Fetching Data Dynamic Routes Fetching a Single Item(getStaticProps)

- If a page has dynamic routes (documentation) and uses getStaticProps it needs to define a list of paths that have to be rendered to HTML at build time.
- If you export an async function called getStaticPaths from a page that uses dynamic routes, Next.js will statically pre-render all the paths specified by getStaticPaths.



```
1 export const getStaticPaths = async () => {
2   const res = await fetch('https://jsonplaceholder.typicode.com/users');
3   const data = await res.json();
4
5   const paths = data.map(ninja => {
6     return {
7       params: { id: ninja.id.toString() }
8     }
9   })
10
11   return {
12     paths,
13     fallback: false
14   }
15 }
16
```

```
export const getStaticProps = async (context) => {
  const id = context.params.id;
  const res = await fetch('https://jsonplaceholder.typicode.com/users/' + id);
  const data = await res.json();

  return {
    props: { ninja: data }
  }
}

const Details = ({ ninja }) => {
  return (
    <div>
      <h1>{ ninja.name }</h1>
      <p>{ ninja.email }</p>
    </div>
  )
}
```



Resources

- <https://www.youtube.com/watch?v=A63UxsQsEbU&list=PL4cUxeGkcC9g9gP2onazU5-2M-AzA8eBw&index=1>
- <https://www.youtube.com/watch?v=MFuwkrseXVE>