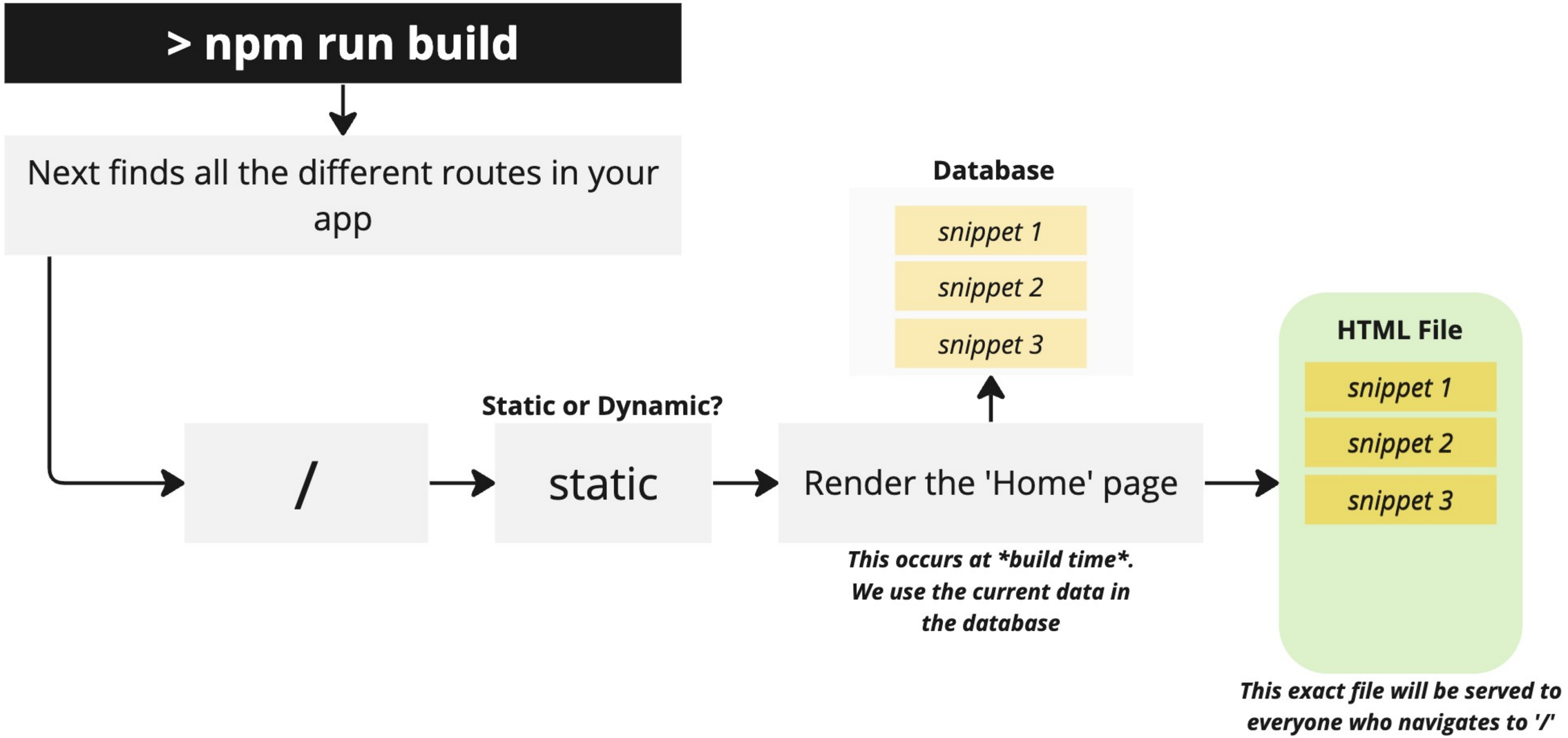
Caching

Next implements caching in several locations.

Can lead to unexpected behavior

Data Cache →	Responses from requests made with '**fetch**' are stored and used across requests.
Router Cache →	'Soft' navigation between routes are cached in the browser and reused when a user revisits a page.
Request Memoization →	Make two or more 'GET' requests with 'fetch' during a user's request to your server? Only one 'GET' is actually executed.
Full Route Cache →	**At build time**, Next decides if your route is ***static*** or ***dynamic***. If it is static, the page is rendered and the result is stored. In production, users are given this pre-rendered result.

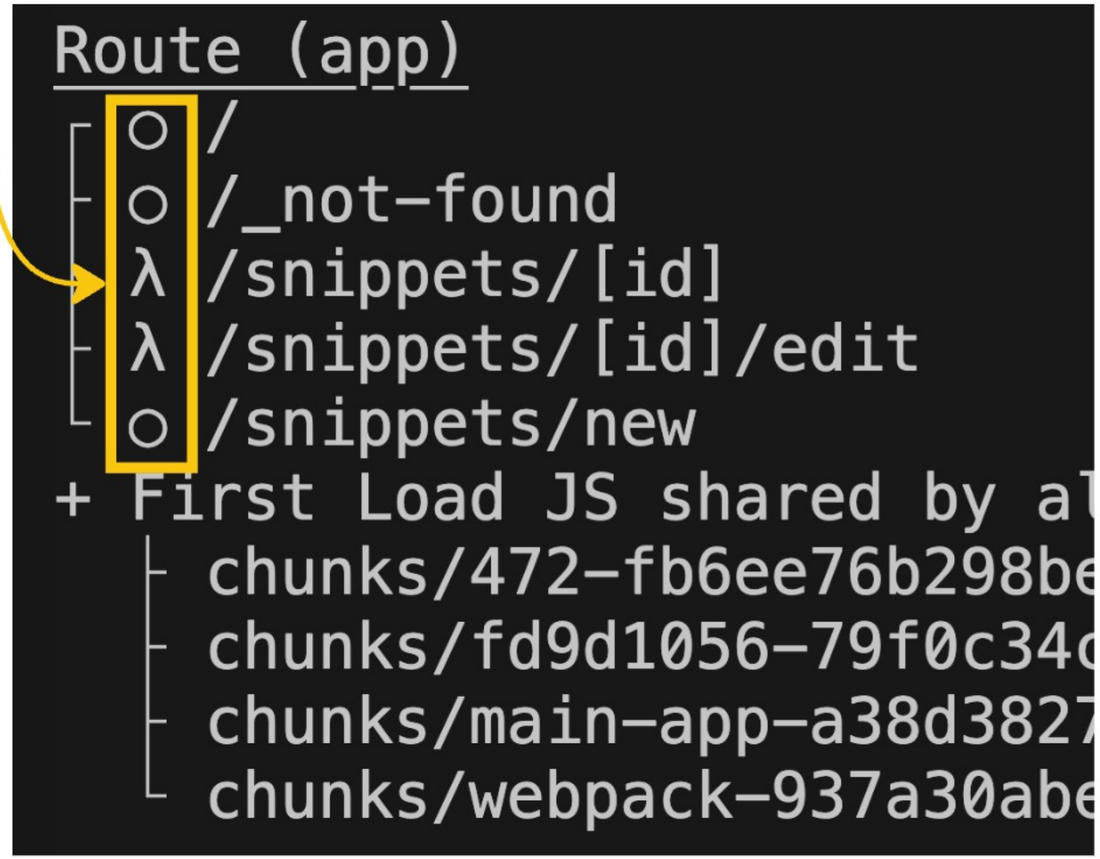

> npm run build
Next finds all the different routes in your app
/
Static or Dynamic?
static
Render the 'Home' page
Database
snippet 1
snippet 2
snippet 3
This occurs at *build time*. We use the current data in the database
HTML File
snippet 1
snippet 2
snippet 3
This exact file will be served to everyone who navigates to '/'

Not just a pretty bullet list!

These indicate what Next thinks about your different routes

○ → Next thinks this route contains only static data. Next will render the page **NOW, ONE TIME** and give that version to everyone who visits your app

λ → Next thinks this route contains dynamic data. Next will render the page **whenever someone visits it**.



What makes a page "dynamic"?

Calling a 'dynamic function' or referencing a 'dynamic variable' when your route renders	cookies.set() cookies.delete() useSearchParams() searchParams prop
Assigning specific 'route segment config' options	export const dynamic = 'force-dynamic' export const revalidate = 0
Calling 'fetch' and opting out of caching of the response	fetch('...', { next: { revalidate: 0 } });
Using a dynamic route	/ snippets / **[id]** / page.tsx / snippets / **[id]** / edit / page.tsx

Help! My next page is rendering with out-of-date data!

There are several ways to control caching

Time-Based	→	Every X seconds, ignore the cached response and fetch new data
On-Demand	→	Forcibly purge a cached response
Disable Caching	→	Don't do any caching at all

Time-Based → Every X seconds, ignore the cached response and fetch new data

Every 3 seconds the next request to this route will trigger a rerender

```
export const revalidate = 3;

export default async function Page() {
    const snippets = await db
        .snippets.findMany();

    return <div>{snippets.map(..)}</div>
}
```

On-Demand → Forcibly purge a cached response

Dump cache for everything in a page

```
import { revalidatePath } from "next/cache";

// When we think data that the '/snippets'
// route uses has changed...
revalidatePath('/snippets');
```

Disable Caching → Don't do any caching at all

Option #1

Disable all caching for a route

```
export const revalidate = 0;

export default async function Page() {
    // Never caches!
}
```

Option #2

Disable all caching for a route

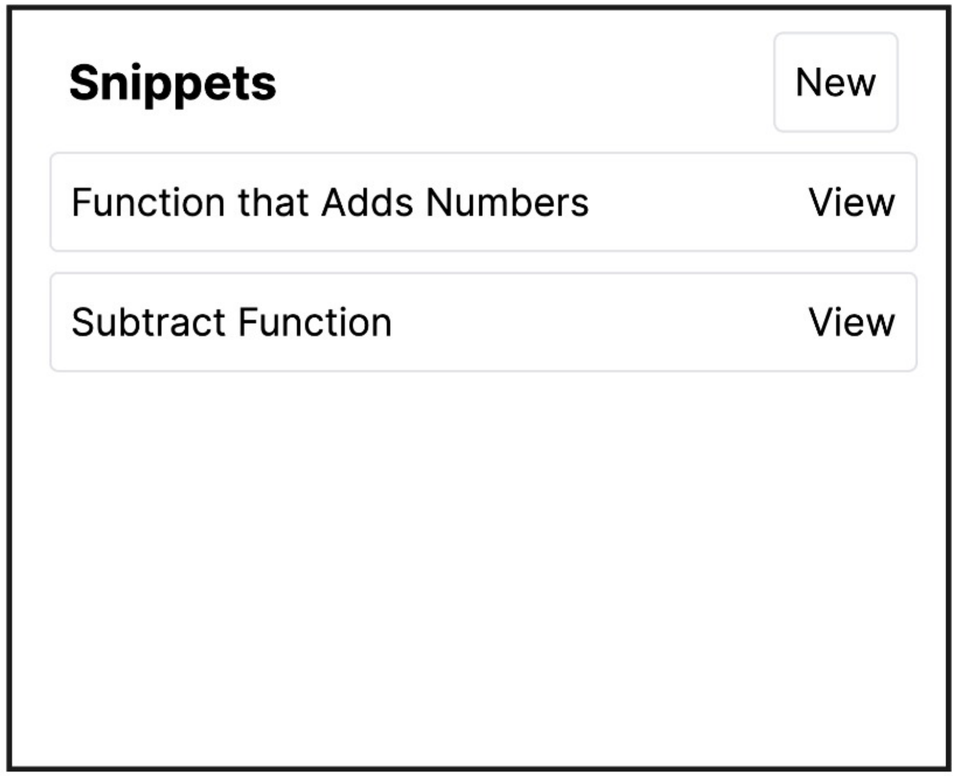
```
export const dynamic = "force-dynamic";

export default async function Page() {
    // Never caches!
}
```

Time-Based → Every X seconds, ignore the cached response and fetch new data

Front page of a social media site

Data is changing all the time - only get the top posts every 10-30 seconds

On-Demand → Forcibly purge a cached response



Any app where we know **when** data changes **and** the user expects to see up-to-date data

For our app, we know *exactly* when data changes. It changes when our create/edit/delete Server Actions run!

Disable Caching → Don't do any caching at all

NextJS → Outside API

Any app where we
(don't know when data changes
or
when we expect the data to be different with every request)
and
the user still expects to see up to date data

> npm run build

Next finds all the different routes in your app

/snipppets/[id] → **Static or Dynamic?** → dynamic → Run 'generateStaticParams'

Database

- *snippet 1*
- *snippet 2*
- *snippet 3*

This occurs at *build time*. We use the current data in the database

- {id: 1}
- {id: 2}
- {id: 3}

<SnippetShowPage params={{id : **1**}} />

<SnippetShowPage params={{id : **2**}} />

<SnippetShowPage params={{id : **3**}} />

Next caches the result

Next caches the result

Next caches the result