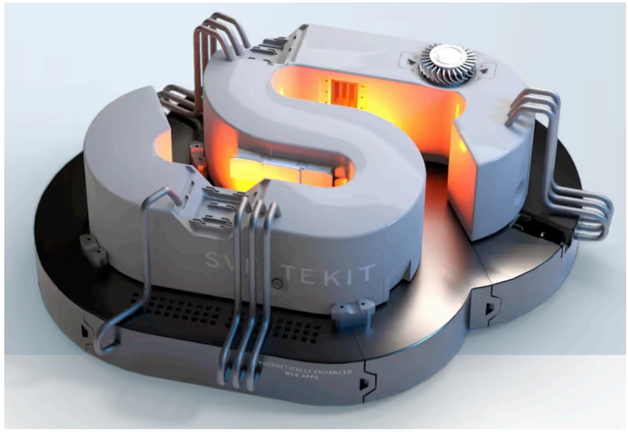
SvelteKit

Introducing SvelteKit - a meta-framework for Svelte

What is Sveltekit?

- SvelteKit is a framework for rapidly developing robust, performant web applications using Svelte.
- If you're coming from React, SvelteKit is similar to Next.
- If you're coming from Vue, SvelteKit is similar to Nuxt.



Remember what Svelte is…

- Svelte is a way of writing user interface components, e.g:
  - navigation bar
  - comment section
  - contact form

- …that users see and interact with in their browsers.
- The Svelte compiler converts your components to JavaScript that can be run to render the HTML for the page and to CSS that styles the page.

Svelte renders UI components - you can compose into a page with just Svelte, however, you need more than just Svelte to write an entire app.

Some Key SvelteKit Features:

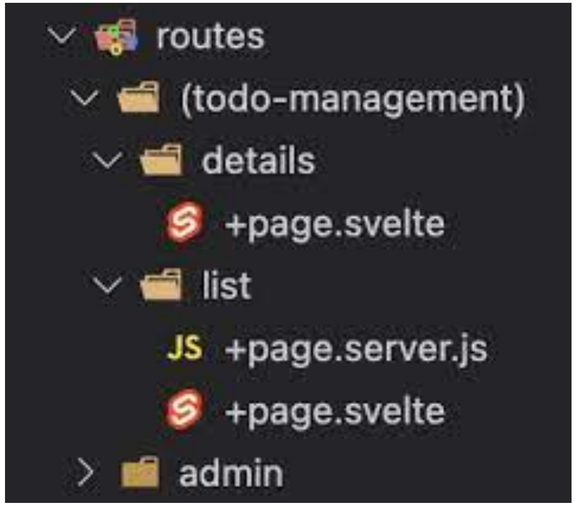
What is Sveltekit?

- file-based page routing
- layouts
- first page visit is server-rendered for performance and remaining pages are rendered in the browser
- file-based endpoints (REST services)
- code splitting
- page visits only load the JS and CSS they need
- hot reloading (HMR)
- static sites and individual static pages
- offline support with ServiceWorkers
- adapters for specific deployment targets

Many of these features can be implemented in Svelte using add on libraries & tools

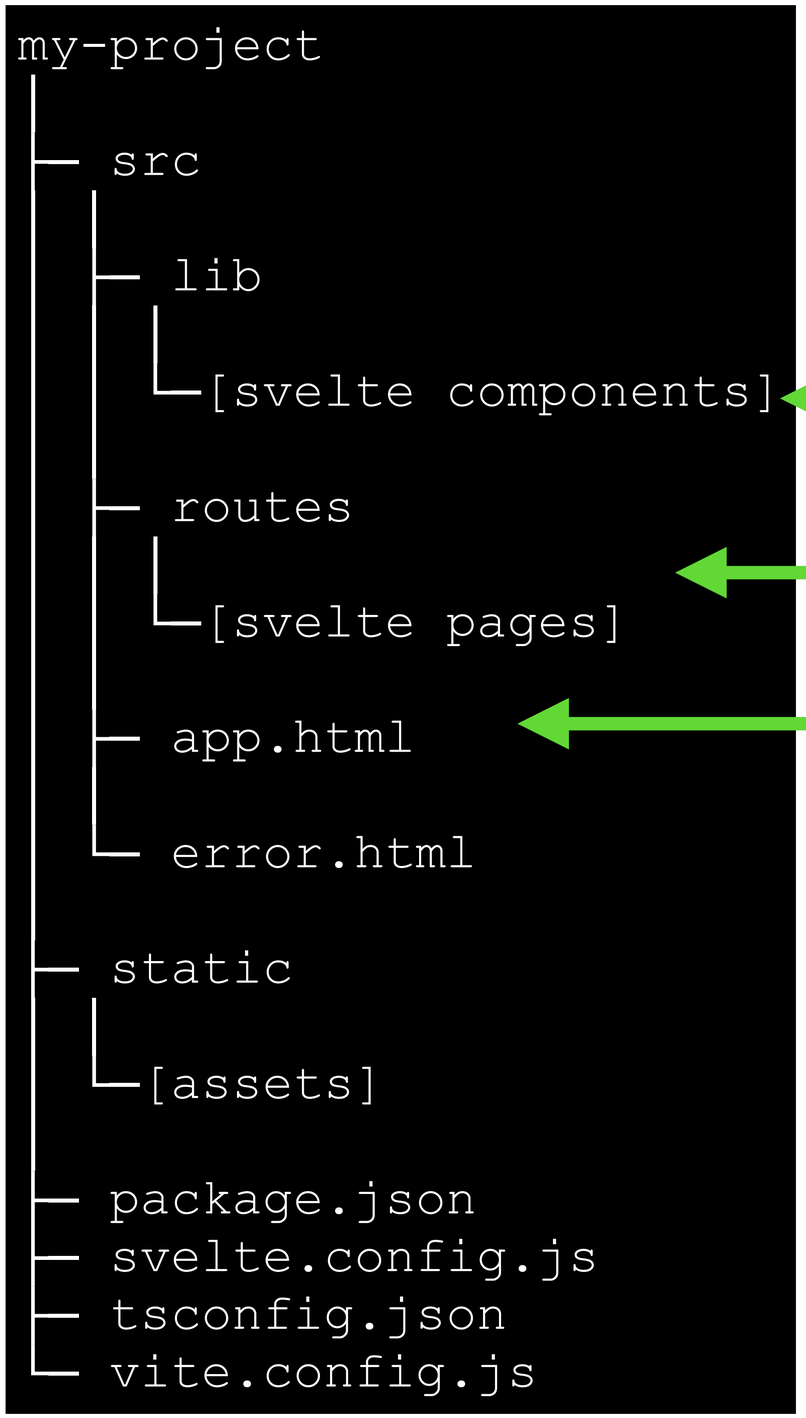
Sveltekit proposes an elegant, simplified implementation of these complex features

Many of these features are available by default, or can be introduced as needed



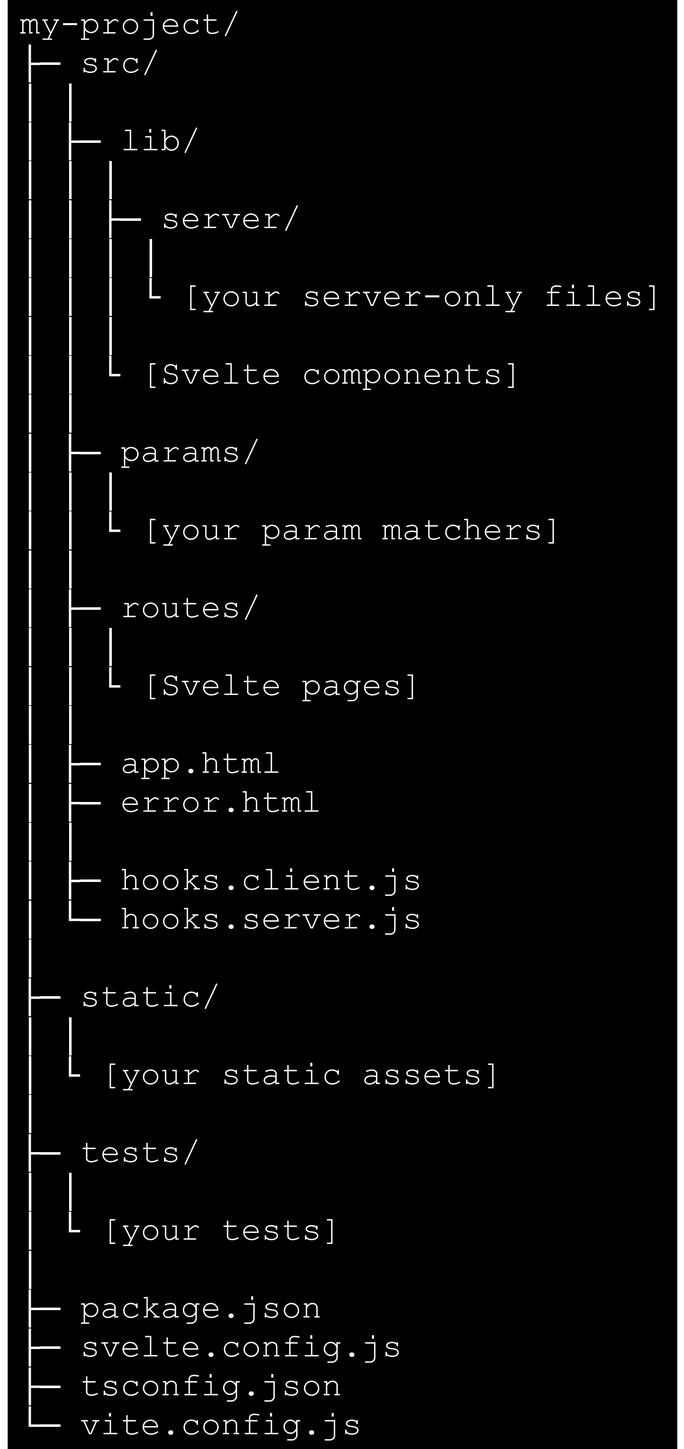
- file-based page routing
- layouts
- first page visit is server-rendered for performance and remaining pages are rendered in the browser
- file-based endpoints (REST services)
- code splitting
- page visits only load the JS and CSS they need
- hot reloading (HMR)
- static sites and individual static pages
- offline support with ServiceWorkers
- adapters for specific deployment targets

SvelteKit Project Structure



- Shared Components
- One Page per ‘View”
- “Main” page

- Prescribed project structure
- Skeleton generated when project created

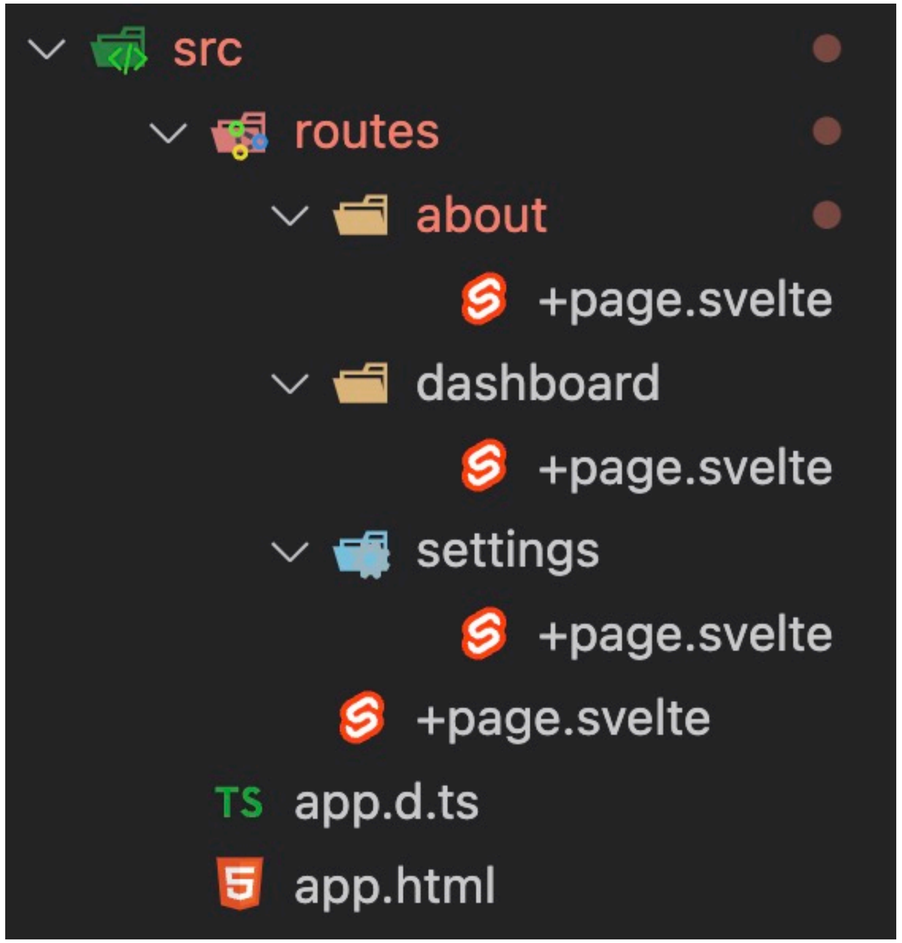


SvelteKit Project Structure

- More advanced features can be introduced as needed:
  - Server side components
  - Advanced url/parameter configuration
  - Hooks
  - Test Infrastructure

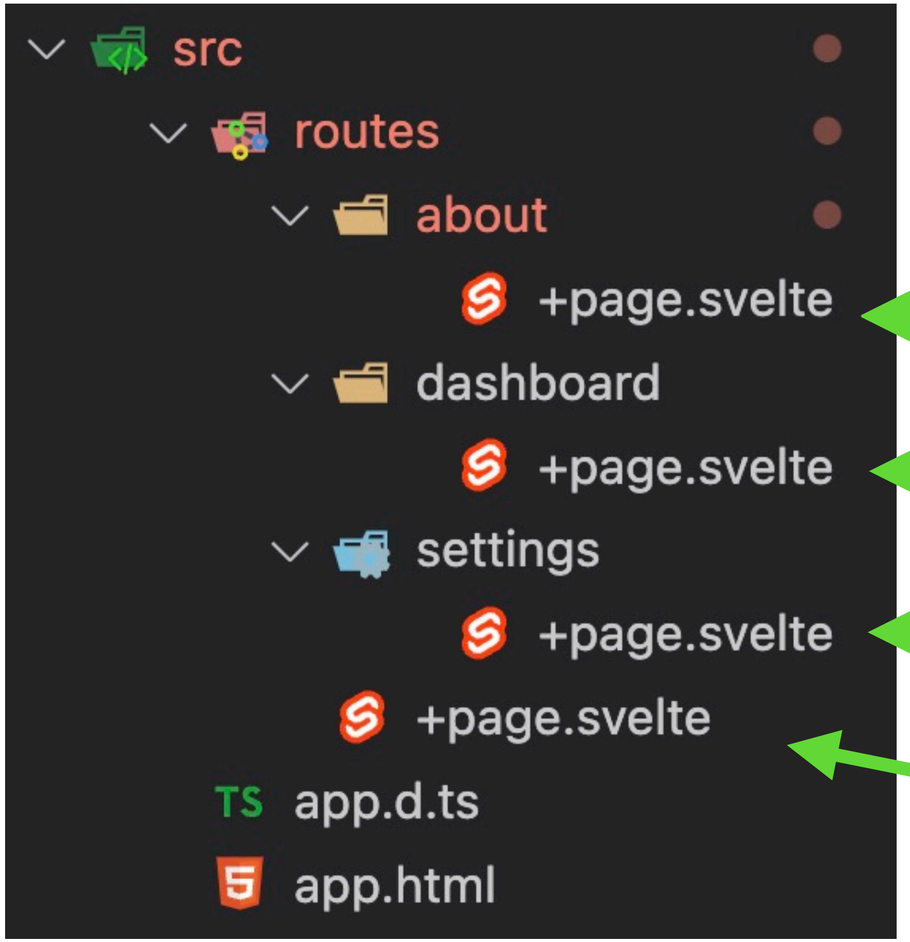
SvelteKit Routing

- At the heart of SvelteKit is a filesystem-based router. The routes of your app — i.e. the URL paths that users can access — are defined by the directories in your codebase:
  - src/routes is the root route
  - src/routes/about creates an /about route
  - src/routes/blog/[slug] creates a route with a parameter, slug, that can be used to load data dynamically when a user requests a page like /blog/hello-world



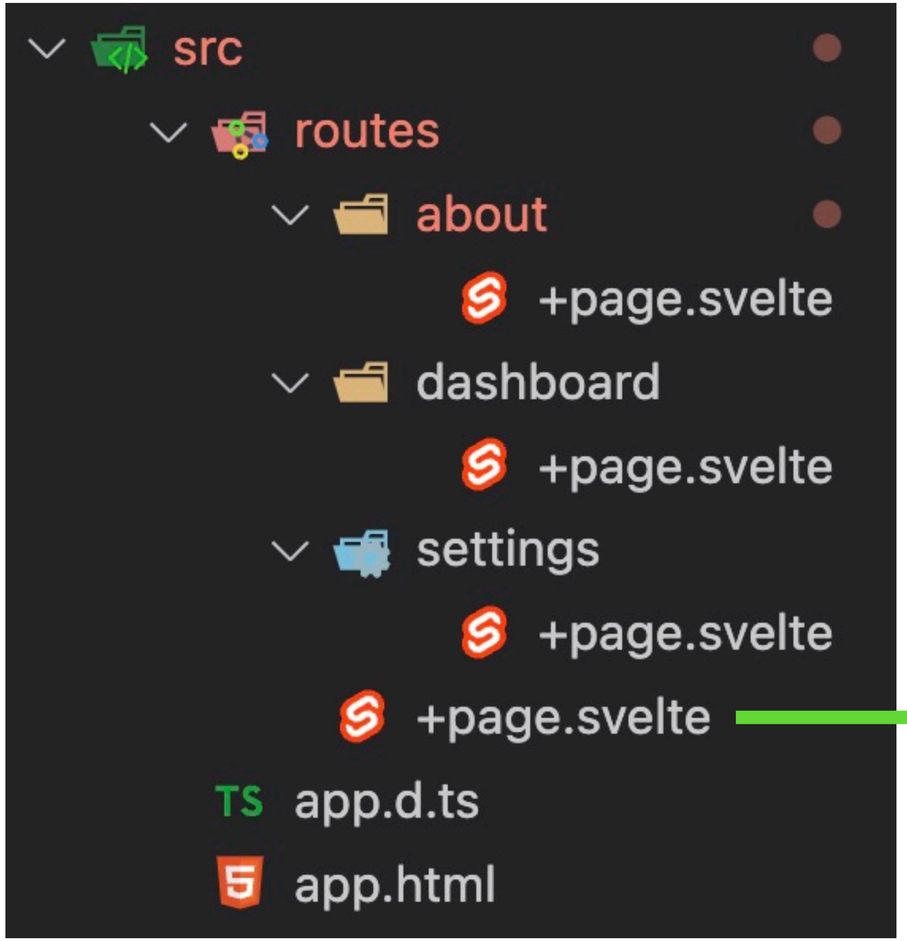
+page

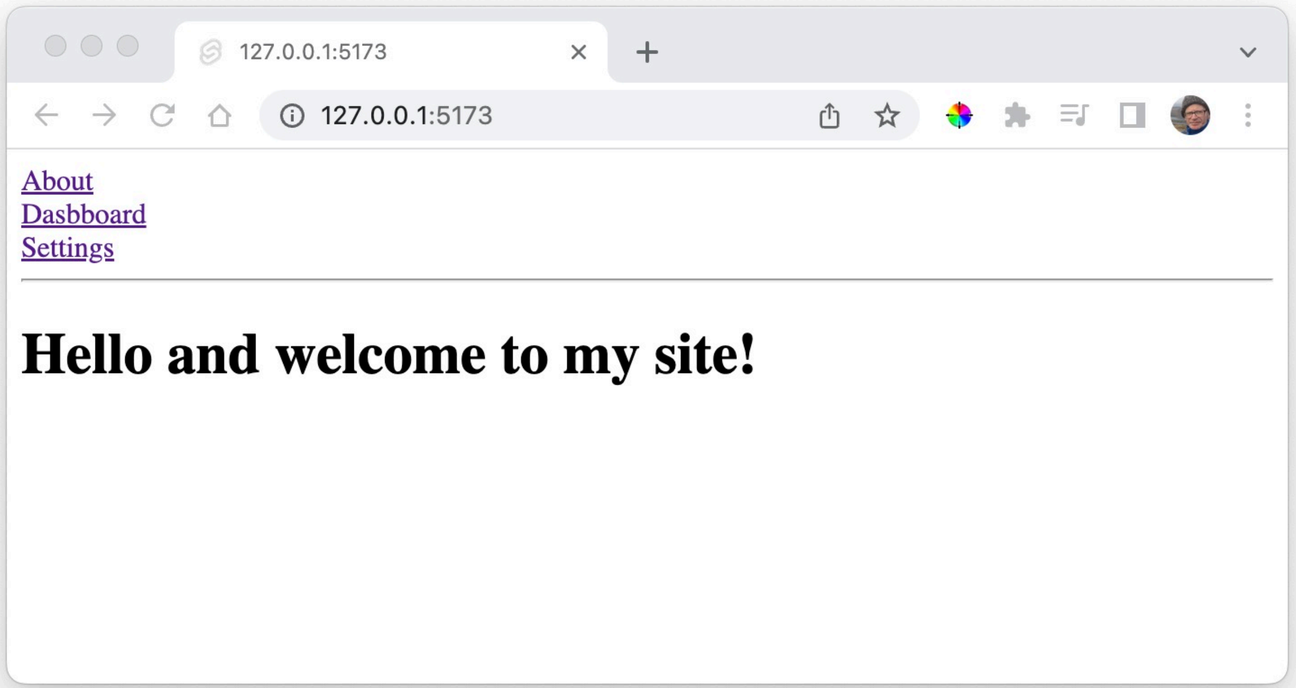
- Each route directory contains one or more route files, which can be identified by their + prefix
- A +page.svelte component defines a page of your app.



- 4 routes
  - /about
  - /dashboard
  - /settings
  - /

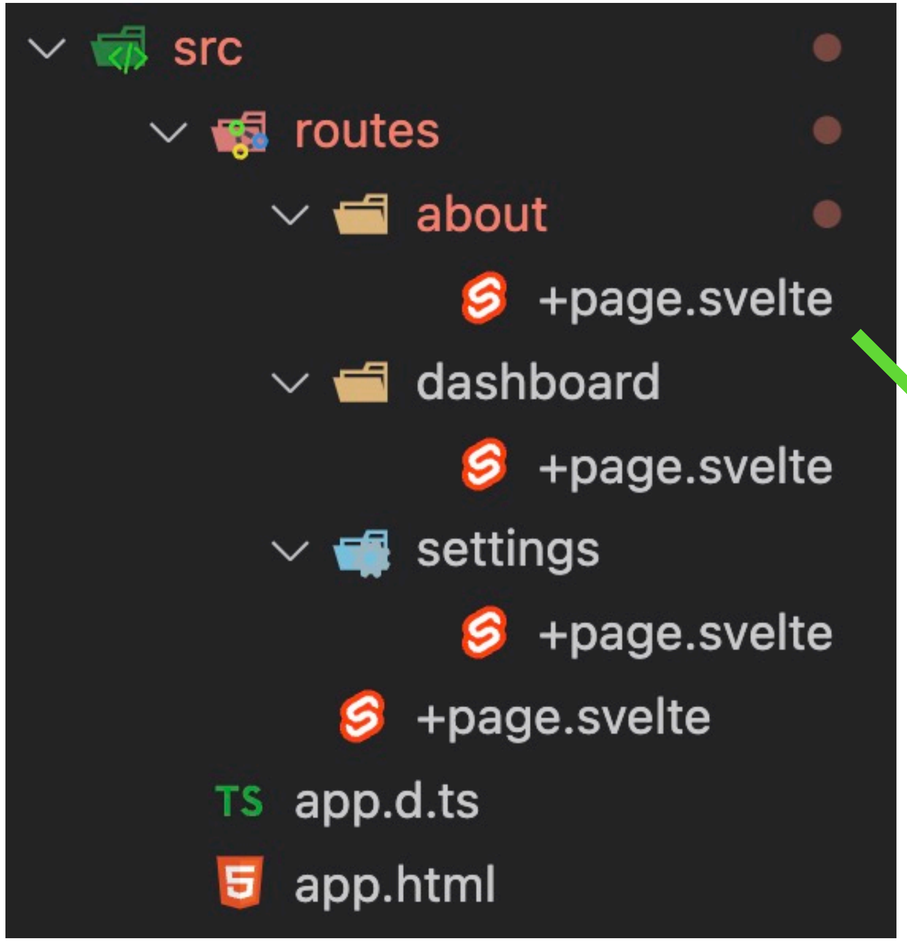
/routes/+page.svelte

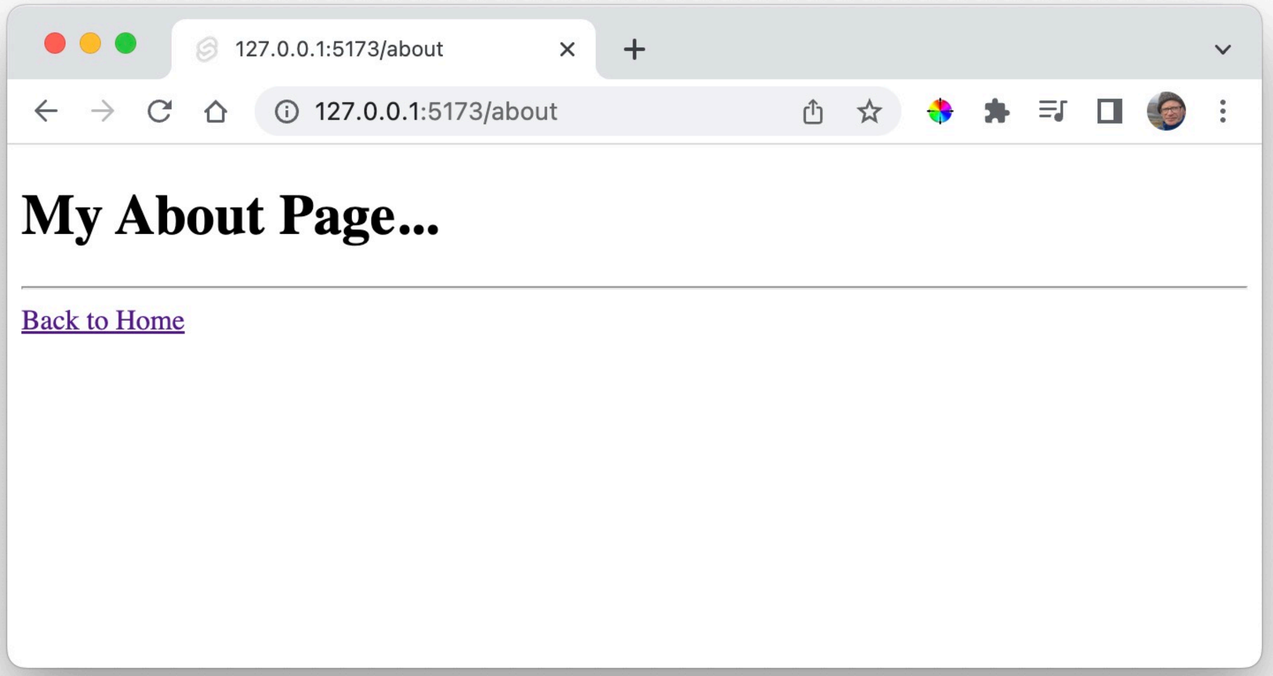





```
<a href="/about">About</a>
<br />
<a href="/dashboard">Dasbboard</a>
<br />
<a href="/settings">Settings</a>
<hr />
<h1>Hello and welcome to my site!</h1>
```

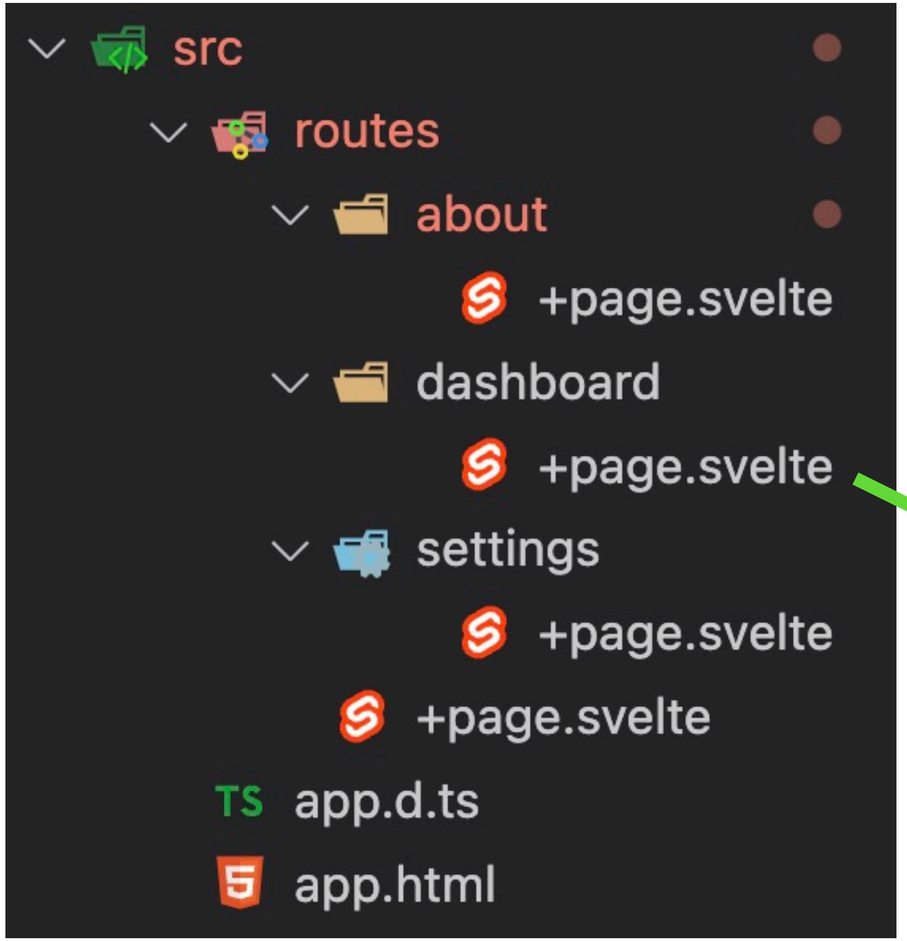
/routes/about/+page.svelte

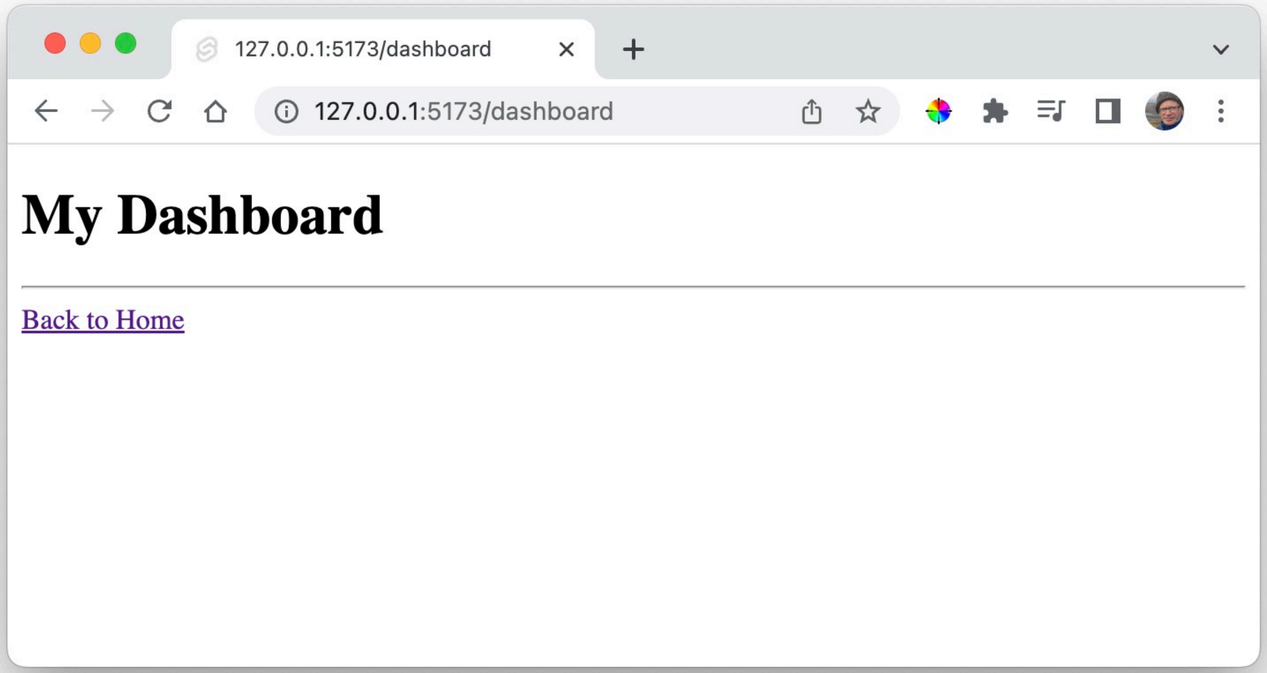





```
<h1>My About Page...</h1>
<hr />
<a href="/">Back to Home</a>
```

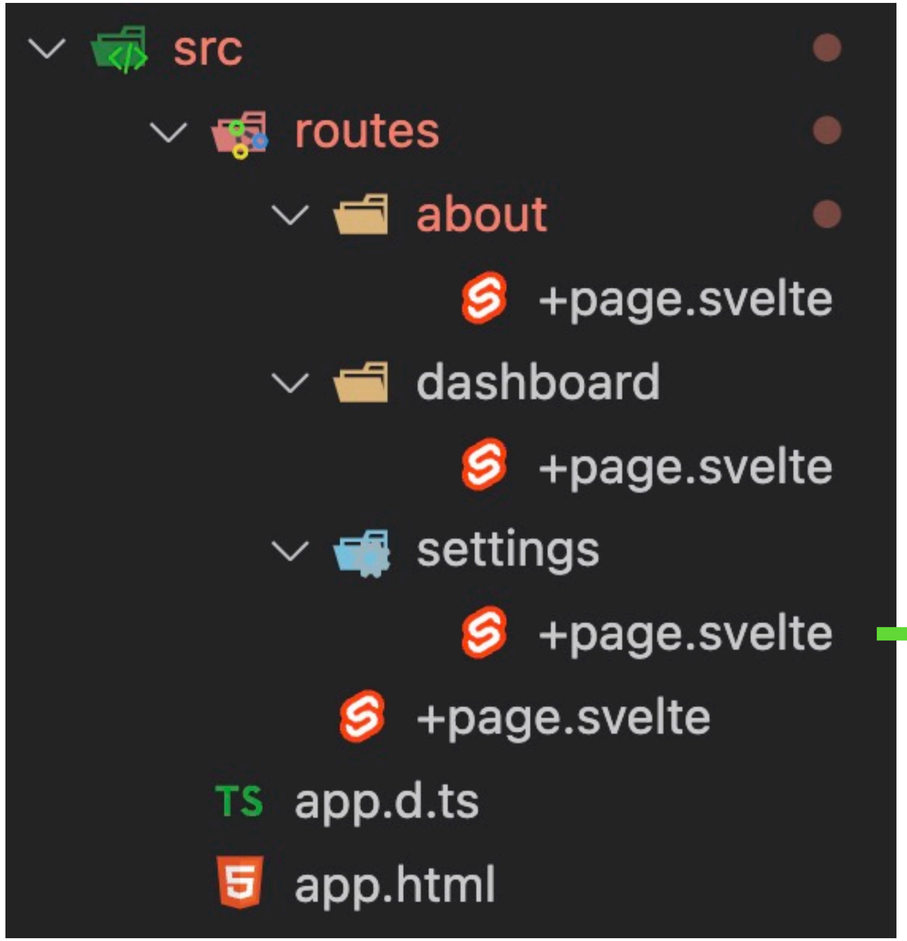
/routes/dashboard/+page.svelte

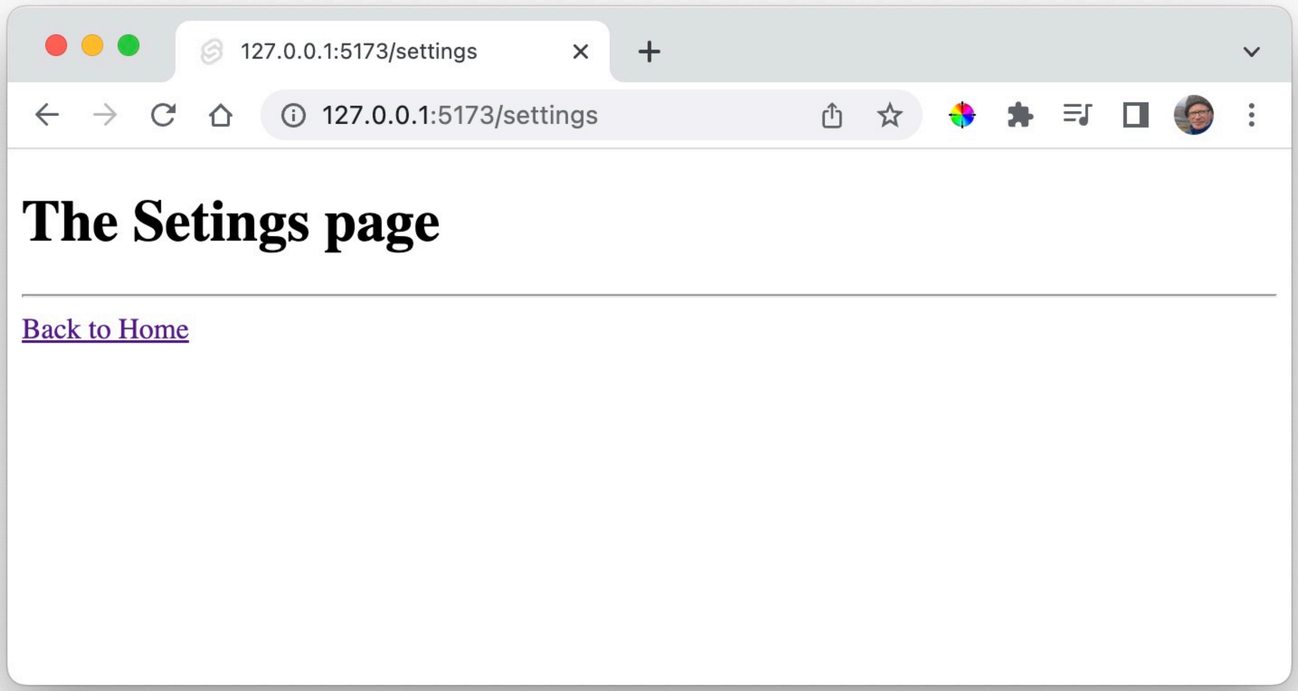





```
<h1>My Dashboard</h1>
<hr />
<a href="/">Back to Home</a>
```

/routes/settings/+page.svelte

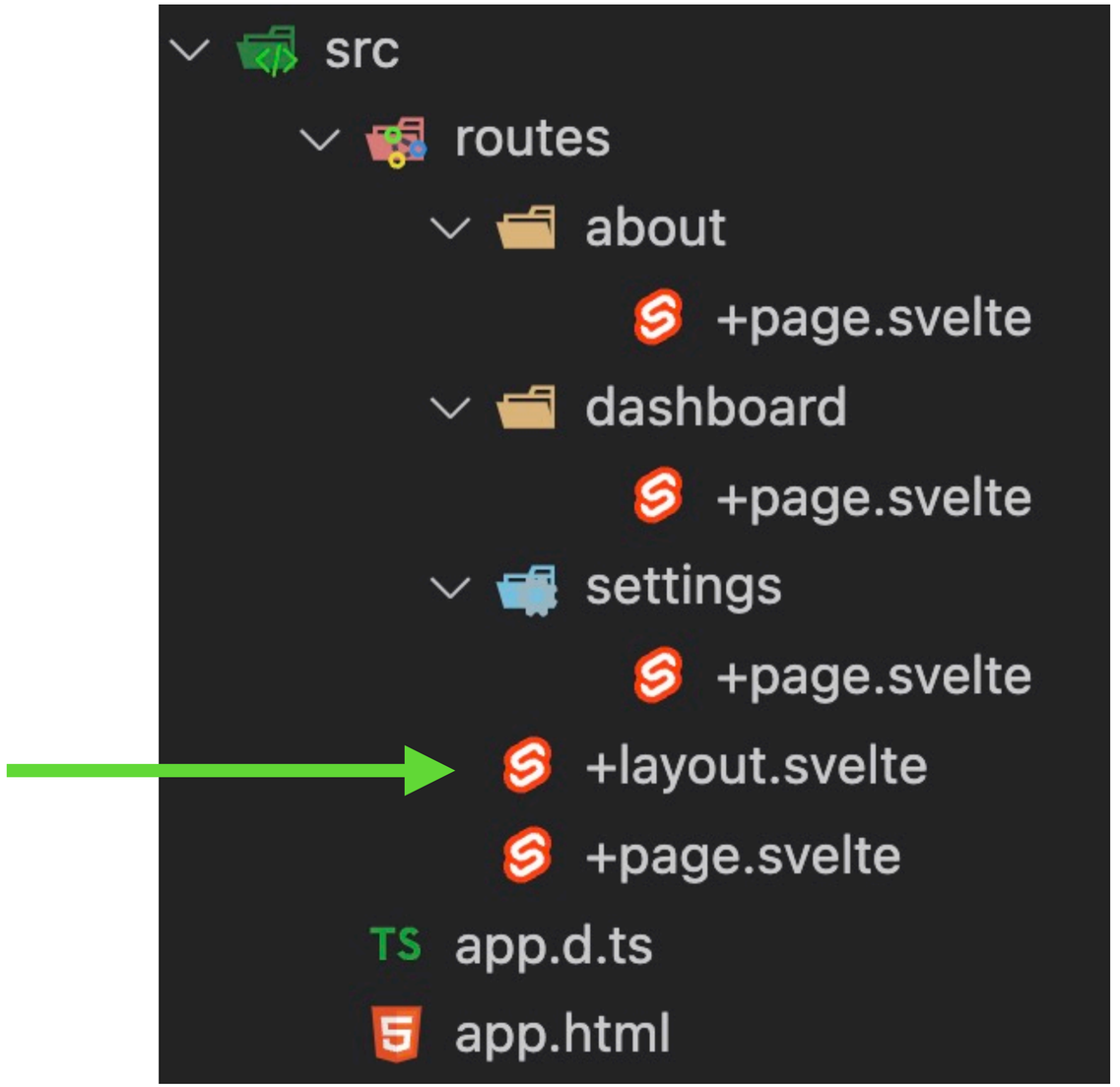




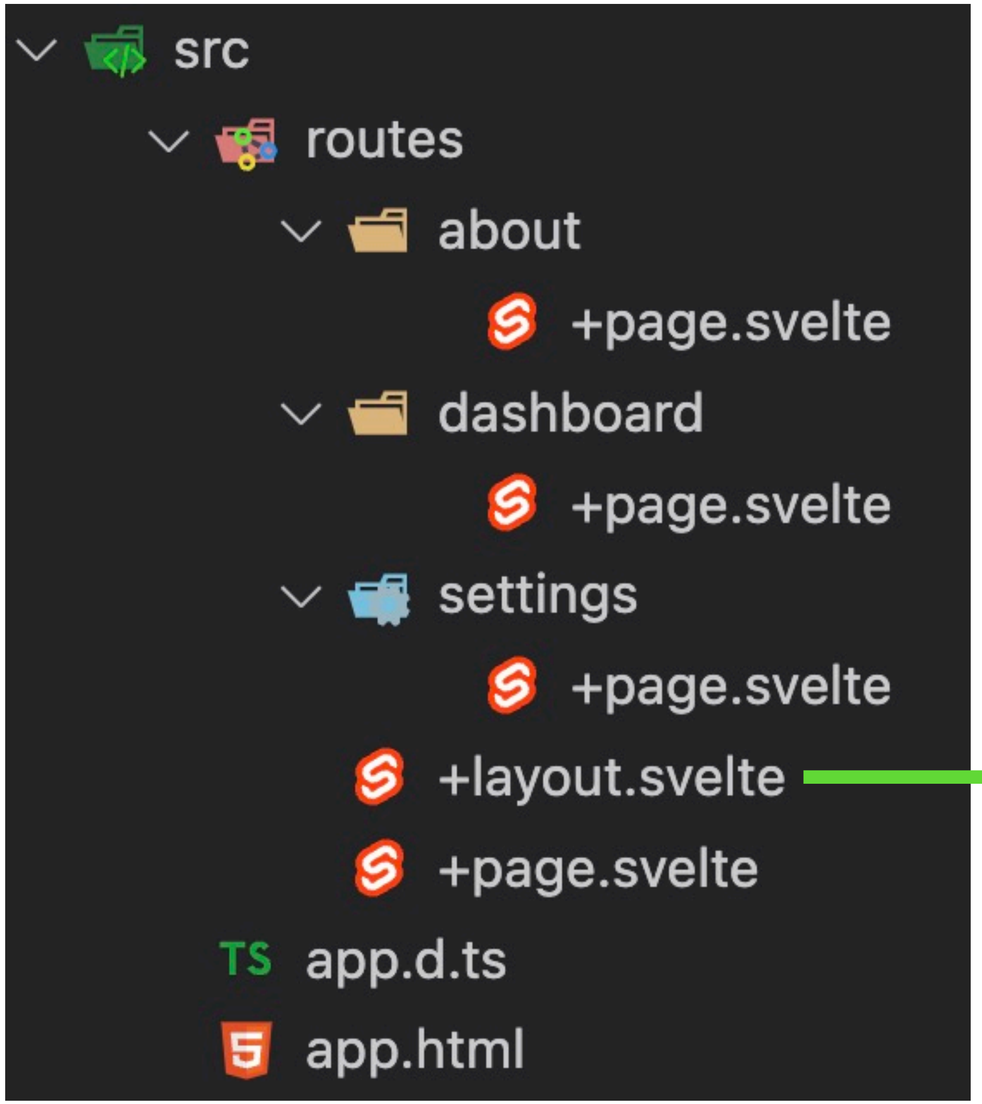

```
<h1>The Setings page</h1>
<hr />
<a href="/">Back to Home</a>
```

+layout.svelte

- By default, pages are entirely standalone components — upon navigation, the existing +page.svelte componen t will be destroyed, and a new one will take its place.
- But in many apps, there are elements that should be visible on every page, such as top-level navigation or a footer.
- Instead of repeating them in every +page.svelte, we can put them in layouts.



+layout.svelte

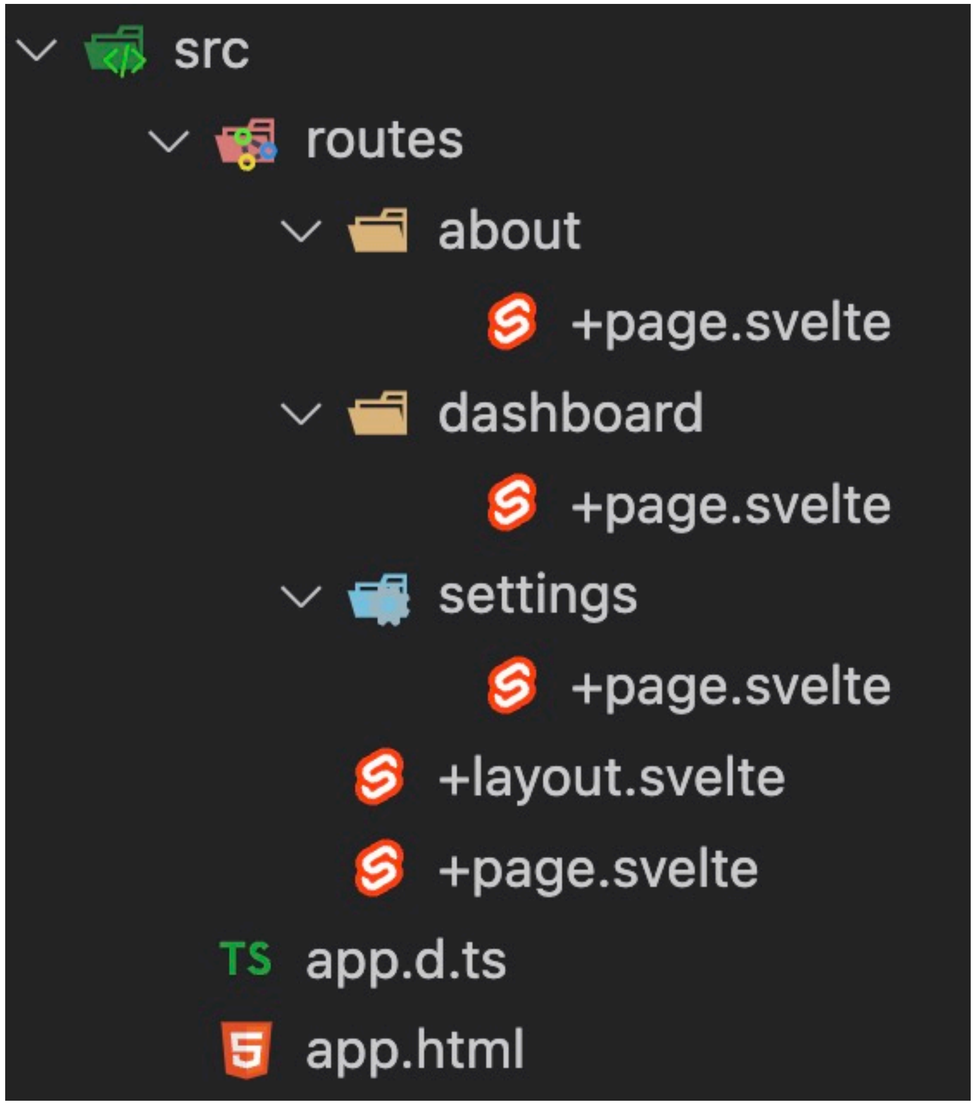


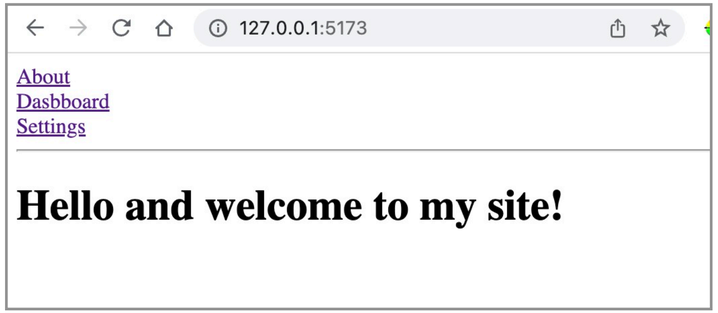
- Includes content that will be part of all pages

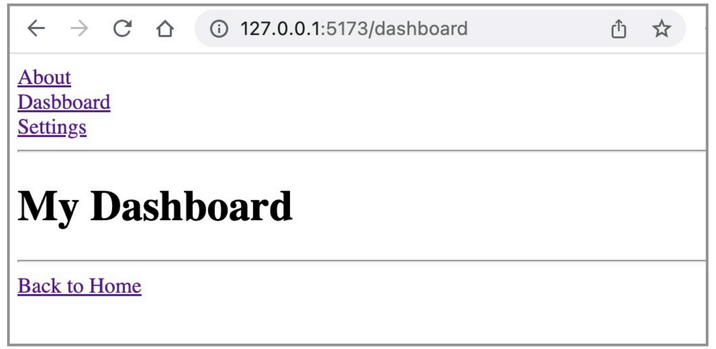
```
<a href="/about">About</a>
<br />
<a href="/dashboard">Dasbboard</a>
<br />
<a href="/settings">Settings</a>
<hr />

<slot />
```

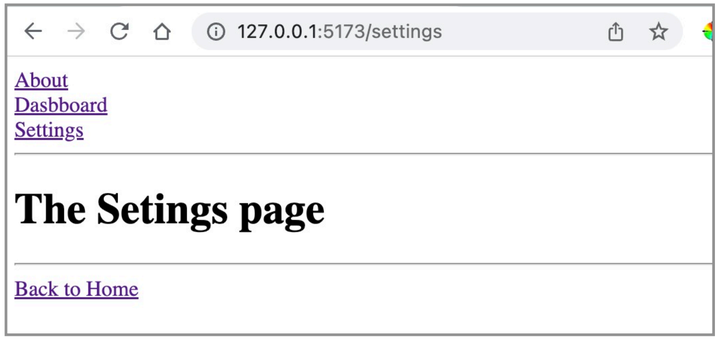
- The <slot> element will be replaced with content from each page

+layout.svelte



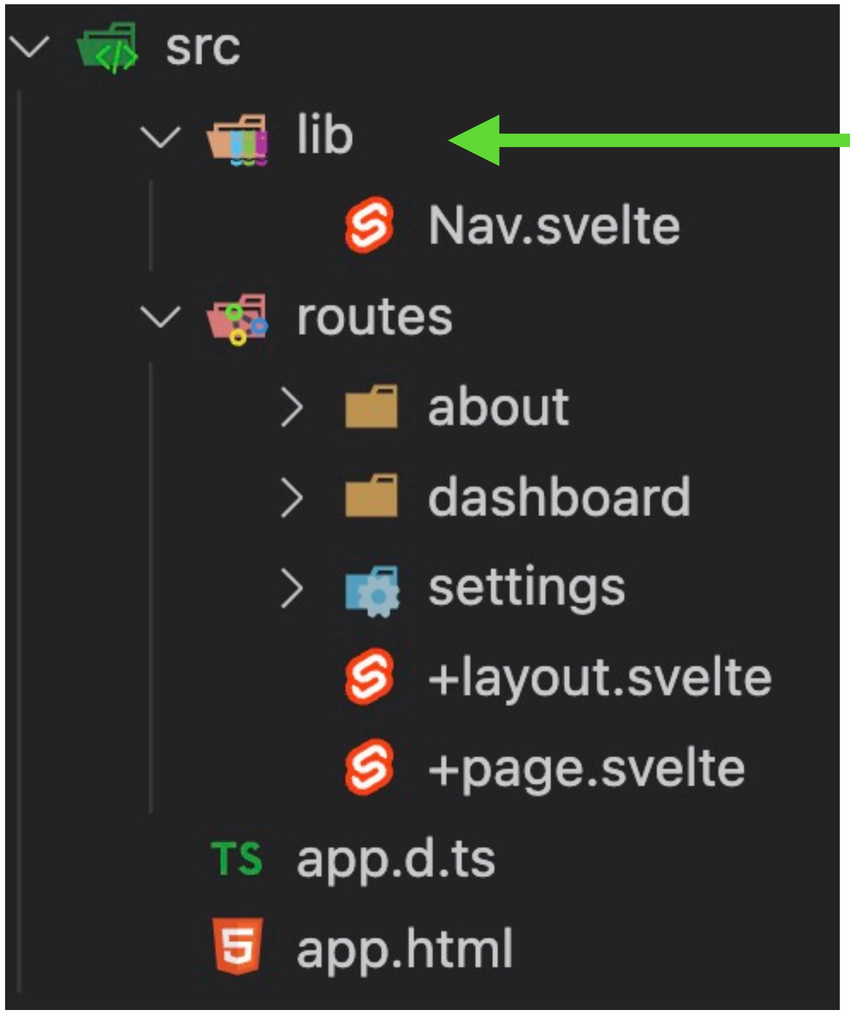






- No changes to the +page.svelte files to have the contents of +layout included

Shared Components

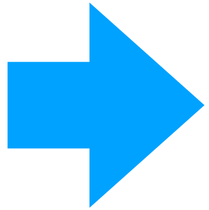


- Shared Components “lib” folder
- Nav Component defines nav bar

```
<a href="/about">About</a>
<br />
<a href="/dashboard">Dasbboard</a>
<br />
<a href="/settings">Settings</a>
<hr />
```

Importing components

```
<a href="/about">About</a>
<br />
<a href="/dashboard">Dasbboard</a>
<br />
<a href="/settings">Settings</a>
<hr />

<slot />
```



+layout.svelte

```
<script>
  import Nav from "$lib/Nav.svelte";
</script>

<Nav />
<slot />
```

- Define a <script> element
- Import the components
- Use the components

- **Built-in server-side rendering (SSR):** SvelteKit makes it easy to create server-rendered applications. You can define server-side routes and APIs using the same Svelte components you use for client-side rendering.

- **Automatic code splitting and lazy loading:** SvelteKit automatically splits your code into smaller chunks and loads them only when needed, reducing initial load times and improving performance.

- **Routing and navigation:** SvelteKit includes a powerful routing system that allows you to define routes using regular expressions, parameters, and wildcards.

- **Zero-config development:** SvelteKit comes with a pre-configured development environment, which means you can start building your application without worrying about setting up a build system or configuring a server.

- **Integration with popular backend systems:** SvelteKit can be integrated with popular backend systems like Node.js, AWS Lambda, and Google Cloud Functions.

So what is SvelteKit Really?

SvelteKit

Introducing SvelteKit - a meta-framework for Svelte