HACK CLUB ZOO

MAKE AN ANIMAL THEMED SVELTE SITE

GET IKEA PLUSHIES AND STICKERS

GUIDE BY YOURS TRULY,

lynn

A Hack Clubber and Svelte
Developer from New York State

November 2025

BEFORE WE START

This guide will not go into every requirement you need to submit to the program. You should have a decent understanding of HTML, CSS, and JS before participating.

View all requirements at

ZOO.HACKCLUB.COM/REQUIREMENTS

View ways to get bonus points on your project at

ZOO.HACKCLUB.COM/MYNTS

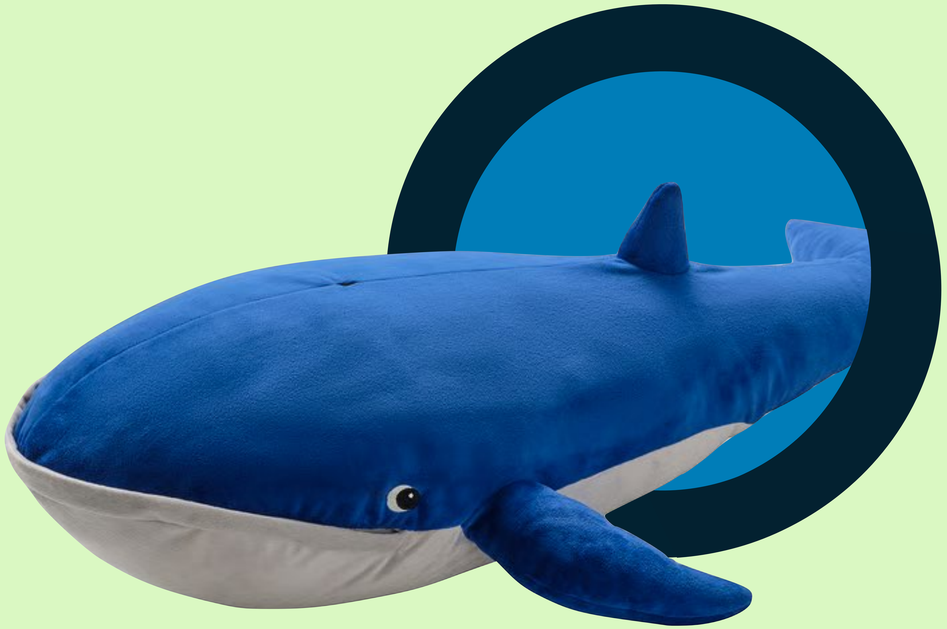
BEFORE WE START

Make sure Hackatime is setup on your computer

Hack Club cannot provide plushies or stickers for your efforts unless you track your time!

Setup Hackatime at

HACKATIME.HACKCLUB.COM



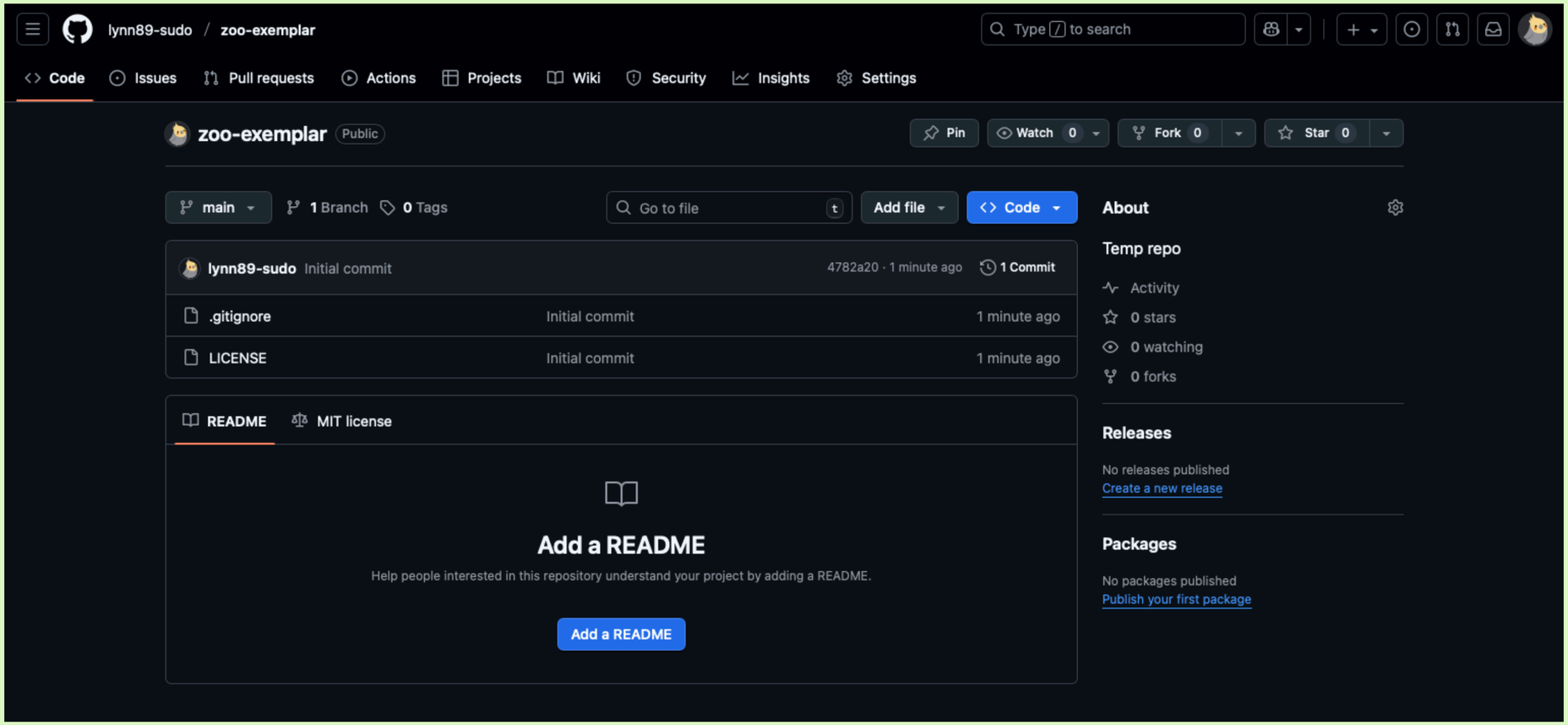
1

Setting up SvelteKit

1

MAKE A GITHUB REPOSITORY

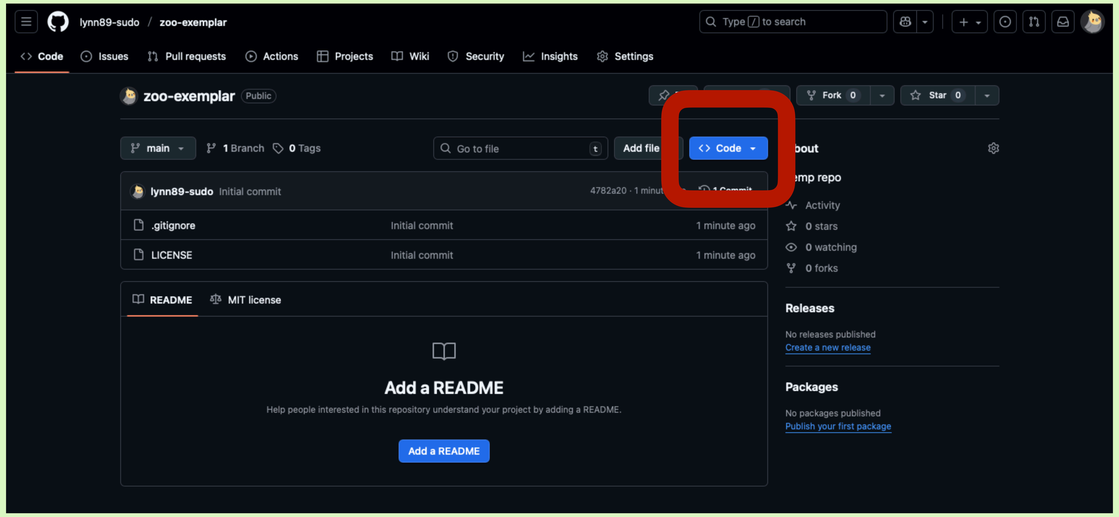
Make a GitHub repository and set the .gitignore to “Node”. Once done, your repository should look something like this.



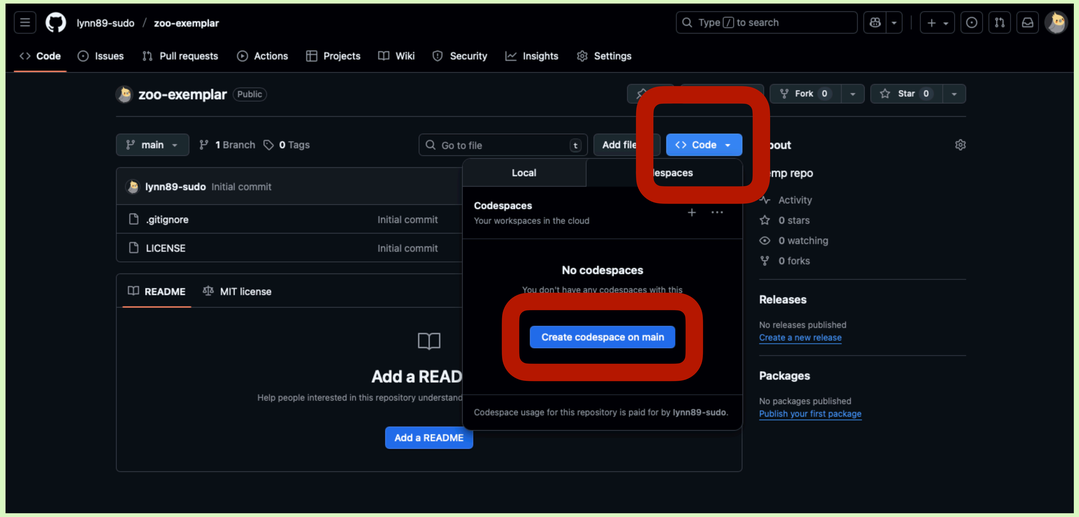
1 OPEN UP YOUR IDE (INTERNAL DEVELOPMENT ENVIRONMENT)

It really doesn't matter what IDE you use, so long as whatever IDE you use supports Node.js and has it currently installed

If you aren't sure, just follow the steps below to create a Codespace (Codespaces have Node.js -and all that jazz- pre-installed)



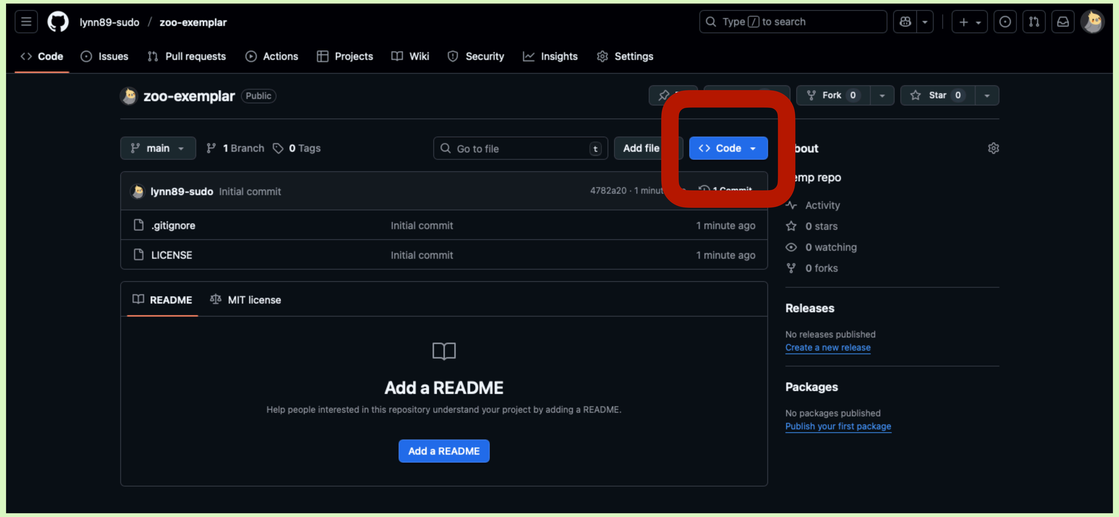
Click the "Code" button



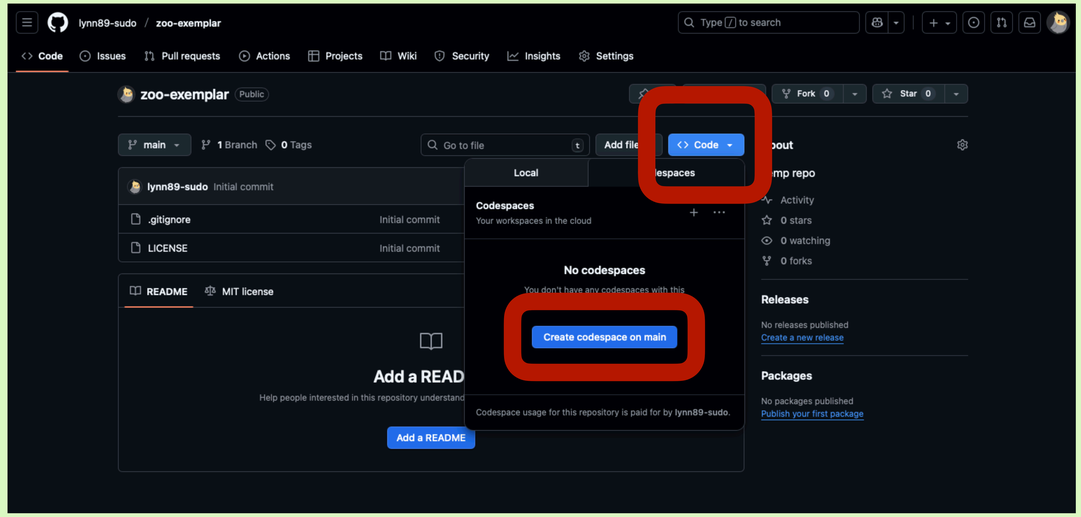
Go to "Codespaces" and "Create codespace on main"

1 OPEN UP YOUR IDE (INTERNAL DEVELOPMENT ENVIRONMENT)

From here, follow instructions on hackatime.hackclub.com to setup Hackatime code tracking. This is required to earn plushies and stickers.



Click the “Code” button



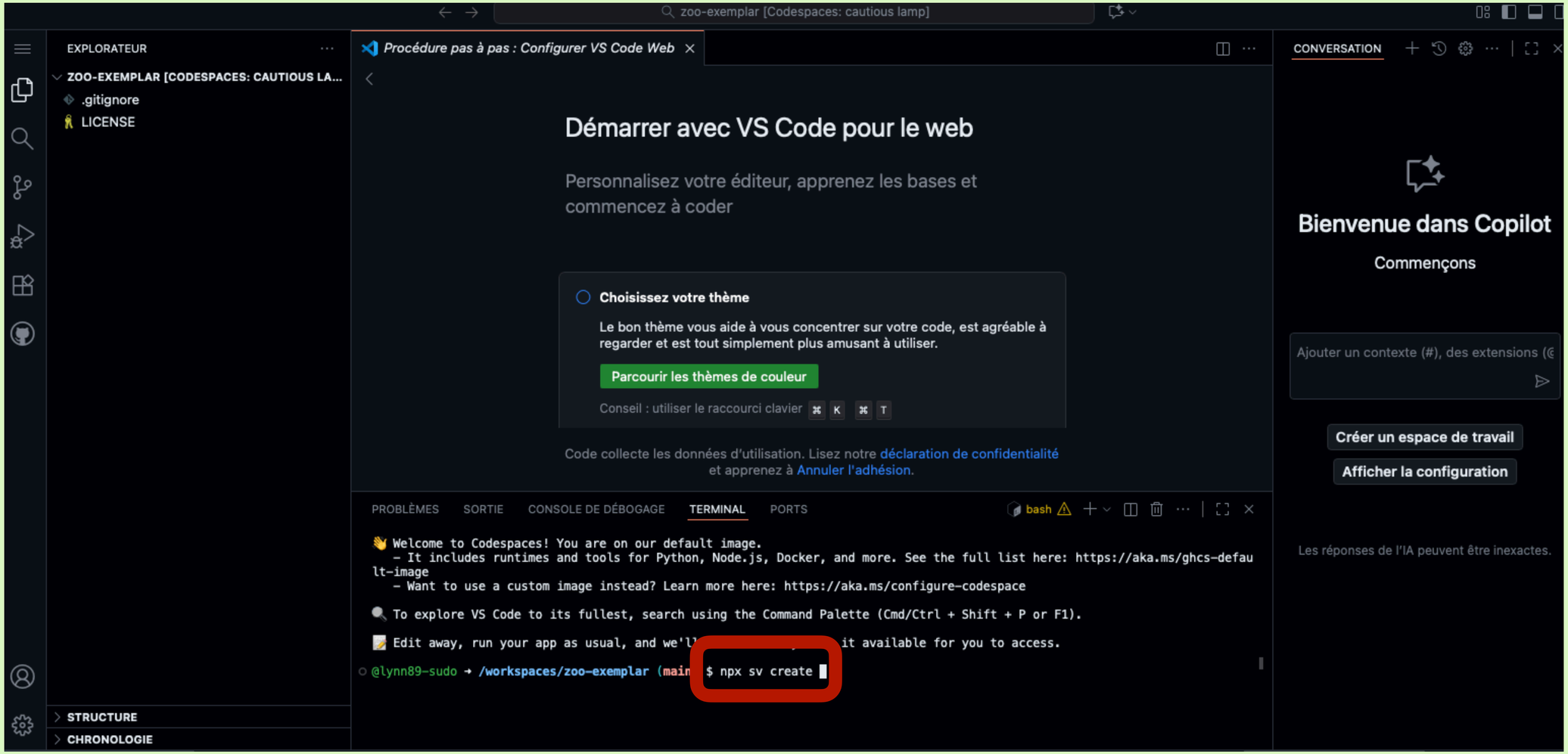
Go to “Codespaces” and “Create codespace on main”

1

SETUP SVELTEKIT WITH NPX

In the terminal, enter “npx sv create”

Yes, my Codespace is in French. Yours, though, should be in English or whatever language that your computer uses.



1

SETUP SVELTEKIT WITH NPX

Yes, we want to install the Svelte packages

Click “y” on your keyboard

```
PROBLÈMES   SORTIE   CONSOLE DE DÉBOGAGE   TERMINAL   PORTS                npm

👋 Welcome to Codespaces! You are on our default image.
   - It includes runtimes and tools for Python, Node.js, Docker, and more. See the full list here: https://aka.ms/ghcs-defau
lt-image
   - Want to use a custom image instead? Learn more here: https://aka.ms/configure-codespace

🔍 To explore VS Code to its fullest, search using the Command Palette (Cmd/Ctrl + Shift + P or F1).

📝 Edit away, run your app as usual, and we'll automatically make it available for you to access.

@lynn89-sudo → /workspaces/zoo-exemplar (main) $ npx sv create
Need to install the following packages:
sv@0.9.9
Ok to proceed? (y)
```

1

SETUP SVELTEKIT WITH NPX

Just click “enter” on your keyboard

If you know what you are doing, you can create a filename for your project; however, given you are using a guide, I suggest you maybe don’t and just follow the instructions)

```
PROBLÈMES   SORTIE   CONSOLE DE DÉBOGAGE   TERMINAL   PORTS                    npm

🔍 To explore VS Code to its fullest, search using the Command Palette (Cmd/Ctrl + Shift + P or F1).

📝 Edit away, run your app as usual, and we'll automatically make it available for you to access.

@lynn89-sudo → /workspaces/zoo-exemplar (main) $ npx sv create
Need to install the following packages:
sv@0.9.9
Ok to proceed? (y) y

┌  Welcome to the Svelte CLI! (v0.9.9)
│
◆  Where would you like your project to be created?
│  (hit Enter to use './')
└
```

1

SETUP SVELTEKIT WITH NPX

You can use the left and right arrow keys to select.
Make sure you choose “yes” and then click enter

```
PROBLÈMES   SORTIE   CONSOLE DE DÉBOGAGE   TERMINAL   PORTS                npm

@lynn89-sudo → /workspaces/zoo-exemplar (main) $ npx sv create
Need to install the following packages:
sv@0.9.9
Ok to proceed? (y) y

  Welcome to the Svelte CLI! (v0.9.9)

◇ Where would you like your project to be created?
  ./

◆ Directory not empty. Continue?
  ● Yes / ○ No
```

1

SETUP SVELTEKIT WITH NPX

You can use the up and down arrow keys to go through the selections; however, the default -minimal- is all we need.

```
PROBLÈMES   SORTIE   CONSOLE DE DÉBOGAGE   TERMINAL   PORTS                npm

@lynn89-sudo → /workspaces/zoo-exemplar (main) $ npx sv create

┌  Welcome to the Svelte CLI! (v0.9.9)
│
◇  Where would you like your project to be created?
│  ./
│
◇  Directory not empty. Continue?
│  Yes
│
◆  Which template would you like?
│  ● SvelteKit minimal (barebones scaffolding for your new app)
│  ○ SvelteKit demo
│  ○ Svelte library
└
```

1

SETUP SVELTEKIT WITH NPX

We aren't using any Typescript

Once you click enter you may see some files appear in the hierarchy; don't worry, that's expected!

```
PROBLÈMES   SORTIE   CONSOLE DE DÉBOGAGE   TERMINAL   PORTS                npm

@lynn89-sudo → /workspaces/zoo-exemplar (main) $ npx sv create
│  ./
│
◇  Directory not empty. Continue?
│  Yes
│
◇  Which template would you like?
│  SvelteKit minimal
│
◆  Add type checking with TypeScript?
│  ○ Yes, using TypeScript syntax
│  ○ Yes, using JavaScript with JSDoc comments
│  ● No
└
```

1

SETUP SVELTEKIT WITH NPX

We can use up and down keys normally; however, this is multi-select, so we click “space” for the options we want. All we need is “sveltekit-adapter”, so select that and then enter.

```
PROBLÈMES   SORTIE   CONSOLE DE DÉBOGAGE   TERMINAL   PORTS                npm

@lynn89-sudo → /workspaces/zoo-exemplar (main) $ npx sv create
◆  What would you like to add to your project? (use arrow keys / space bar)
│  ◻ prettier
│  ◻ eslint
│  ◻ vitest
│  ◻ playwright
│  ◻ tailwindcss
│  ■ sveltekit-adapter (deployment - https://svelte.dev/docs/kit/adapters)
│  ◻ devtools-json
│  ◻ drizzle
│  ◻ lucia
│  ◻ mdsvex
│  ...
└
```

1

SETUP SVELTEKIT WITH NPX

We are using static adapter

```
PROBLÈMES   SORTIE   CONSOLE DE DÉBOGAGE   TERMINAL   PORTS                npm

@lynn89-sudo → /workspaces/zoo-exemplar (main) $ npx sv create
◇  What would you like to add to your project? (use arrow keys / space bar)
│  sveltekit-adapter
│
◆  Which SvelteKit adapter would you like to use?
│  ○ auto
│  ○ node
│  ● static (@sveltejs/adapter-static)
│  ○ vercel
│  ○ cloudflare
│  ○ netlify
└
```

1

SETUP SVELTEKIT WITH NPX

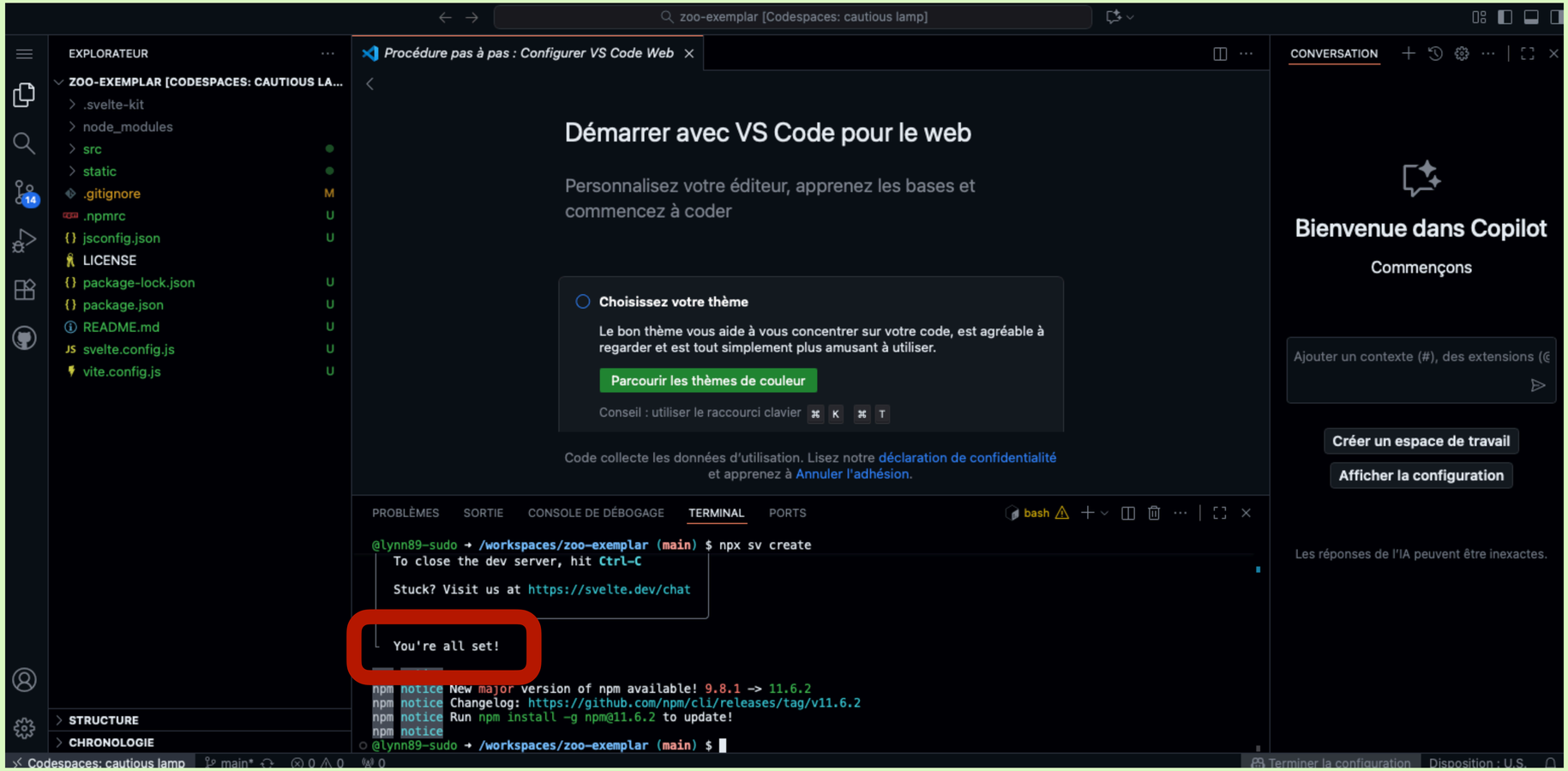
We vibe with npm, so select that.

```
PROBLÈMES   SORTIE   CONSOLE DE DÉBOGAGE   TERMINAL   PORTS                npm

@lynn89-sudo → /workspaces/zoo-exemplar (main) $ npx sv create
◇  Which SvelteKit adapter would you like to use?
│  static
│
◆  Successfully setup add-ons
│
◆  Which package manager do you want to install dependencies with?
│  ○ None
│  ● npm
│  ○ yarn
│  ○ pnpm
│  ○ bun
│  ○ deno
└
```

1

SETUP SVELTEKIT WITH NPX

It should load for a second, then tell you that you're all set!



1

COMMIT OUR INITIAL REPOSITORY SETUP

While everything is setup locally, lets just sync our repository that we made with all the new changes.

Run “git add .” in the terminal

```
PROBLÈMES   SORTIE   CONSOLE DE DÉBOGAGE   TERMINAL   PORTS                bash

@lynn89-sudo → /workspaces/zoo-exemplar (main) $ npx sv create
│  To close the dev server, hit Ctrl-C
│
│  Stuck? Visit us at https://svelte.dev/chat
│
└  You're all set!

npm notice
npm notice New major version of npm available! 9.8.1 -> 11.6.2
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.6.2
npm notice Run npm install -g npm@11.6.2 to update!
npm notice
@lynn89-sudo → /workspaces/zoo-exemplar (main) $ git add .
```

1

COMMIT OUR INITIAL REPOSITORY SETUP

Then, run this: git commit -m “Initial Commit”

```
PROBLÈMES   SORTIE   CONSOLE DE DÉBOGAGE   TERMINAL   PORTS                bash

@lynn89-sudo → /workspaces/zoo-exemplar (main) $ npx sv create
│  Stuck? Visit us at https://svelte.dev/chat
│
└  You're all set!

npm notice
npm notice New major version of npm available! 9.8.1 -> 11.6.2
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.6.2
npm notice Run npm install -g npm@11.6.2 to update!
npm notice
● @lynn89-sudo → /workspaces/zoo-exemplar (main) $ git add .
● @lynn89-sudo → /workspaces/zoo-exemplar (main) $ git commit -m "Initial commit"
[main e2de0del Initial commit
```

1

COMMIT OUR INITIAL REPOSITORY SETUP

The terminal should spit out a bunch of text like this (dodge out of the way so it doesn't spit in your face!)

PROBLÈMES SORTIE CONSOLE DE DÉBOGAGE TERMINAL PORTS bash

```
@lynn89-sudo → /workspaces/zoo-exemplar (main) $ git commit -m "Initial commit"
 create mode 100644 .npmrc
 create mode 100644 README.md
 create mode 100644 jsconfig.json
 create mode 100644 package-lock.json
 create mode 100644 package.json
 create mode 100644 src/app.html
 create mode 100644 src/lib/assets/favicon.svg
 create mode 100644 src/lib/index.js
 create mode 100644 src/routes/+layout.svelte
 create mode 100644 src/routes/+page.svelte
 create mode 100644 static/robots.txt
 create mode 100644 svelte.config.js
 create mode 100644 vite.config.js
@lynn89-sudo → /workspaces/zoo-exemplar (main) $
```

1

COMMIT OUR INITIAL REPOSITORY SETUP

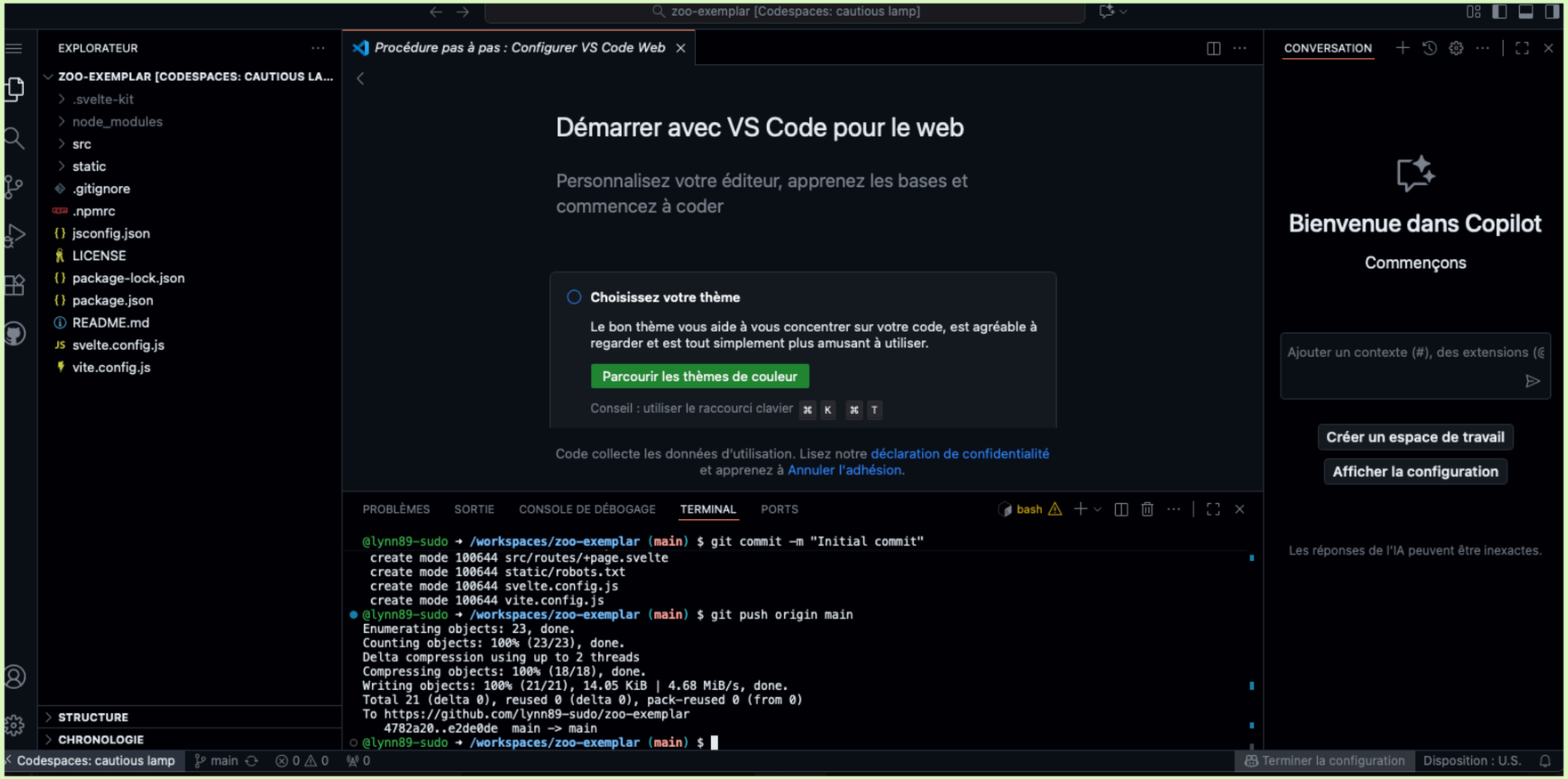
Finally, run this: git push origin main

```
PROBLÈMES   SORTIE   CONSOLE DE DÉBOGAGE   TERMINAL   PORTS                bash

@lynn89-sudo → /workspaces/zoo-exemplar (main) $ git commit -m "Initial commit"
 create mode 100644 .npmrc
 create mode 100644 README.md
 create mode 100644 jsconfig.json
 create mode 100644 package-lock.json
 create mode 100644 package.json
 create mode 100644 src/app.html
 create mode 100644 src/lib/assets/favicon.svg
 create mode 100644 src/lib/index.js
 create mode 100644 src/routes/+layout.svelte
 create mode 100644 src/routes/+page.svelte
 create mode 100644 static/robots.txt
 create mode 100644 svelte.config.js
 create mode 100644 vite.config.js
@lynn89-sudo → /workspaces/zoo-exemplar (main) $ git push origin main
```

1

COMMIT OUR INITIAL REPOSITORY SETUP

Text in the terminal after you run the command should look like this, and all the green and yellow where your files are should just turn to white.



ALL DONE!

Take a break before we jump into the next part ^w^

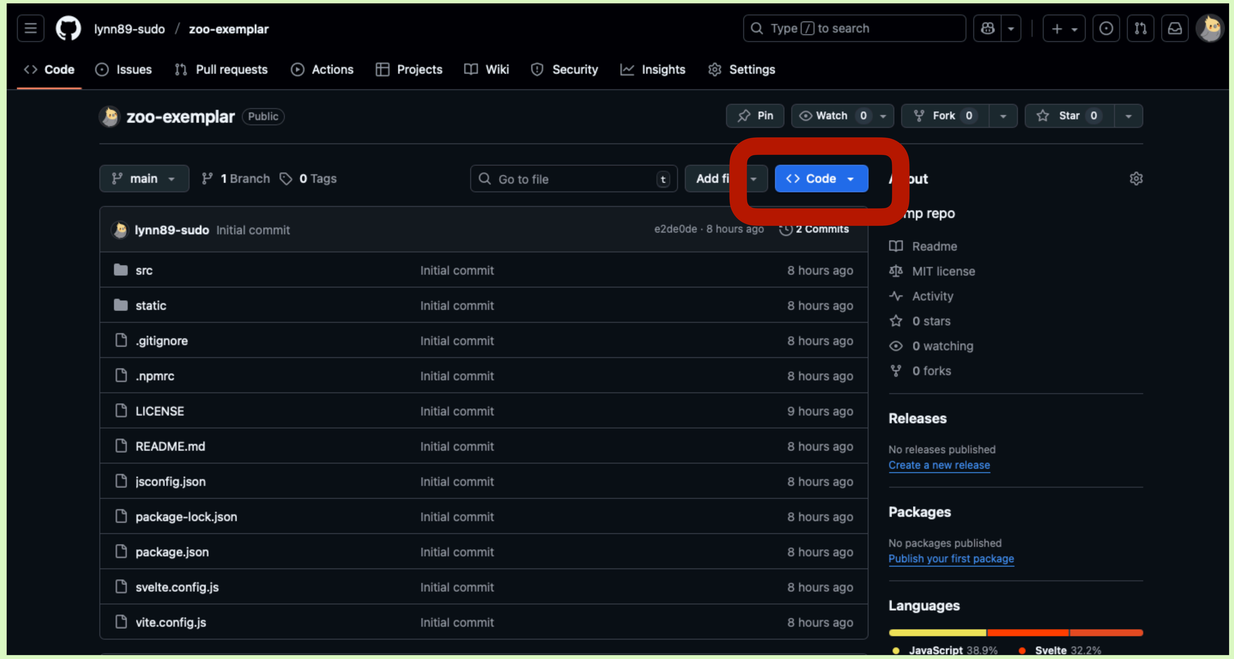
2

Getting around SvelteKit File Structure

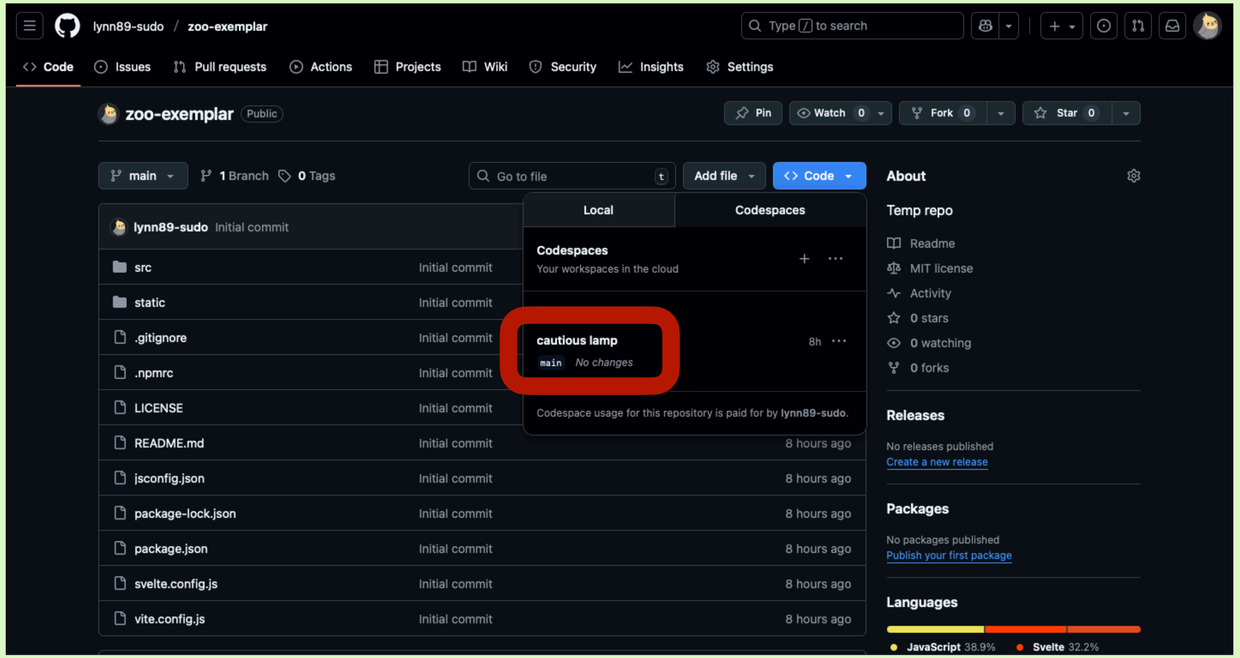
2

OPEN UP YOUR IDE

Let's reopen our IDE. If you were using Codespaces, head back to the repository and follow the steps below:



Click on Code

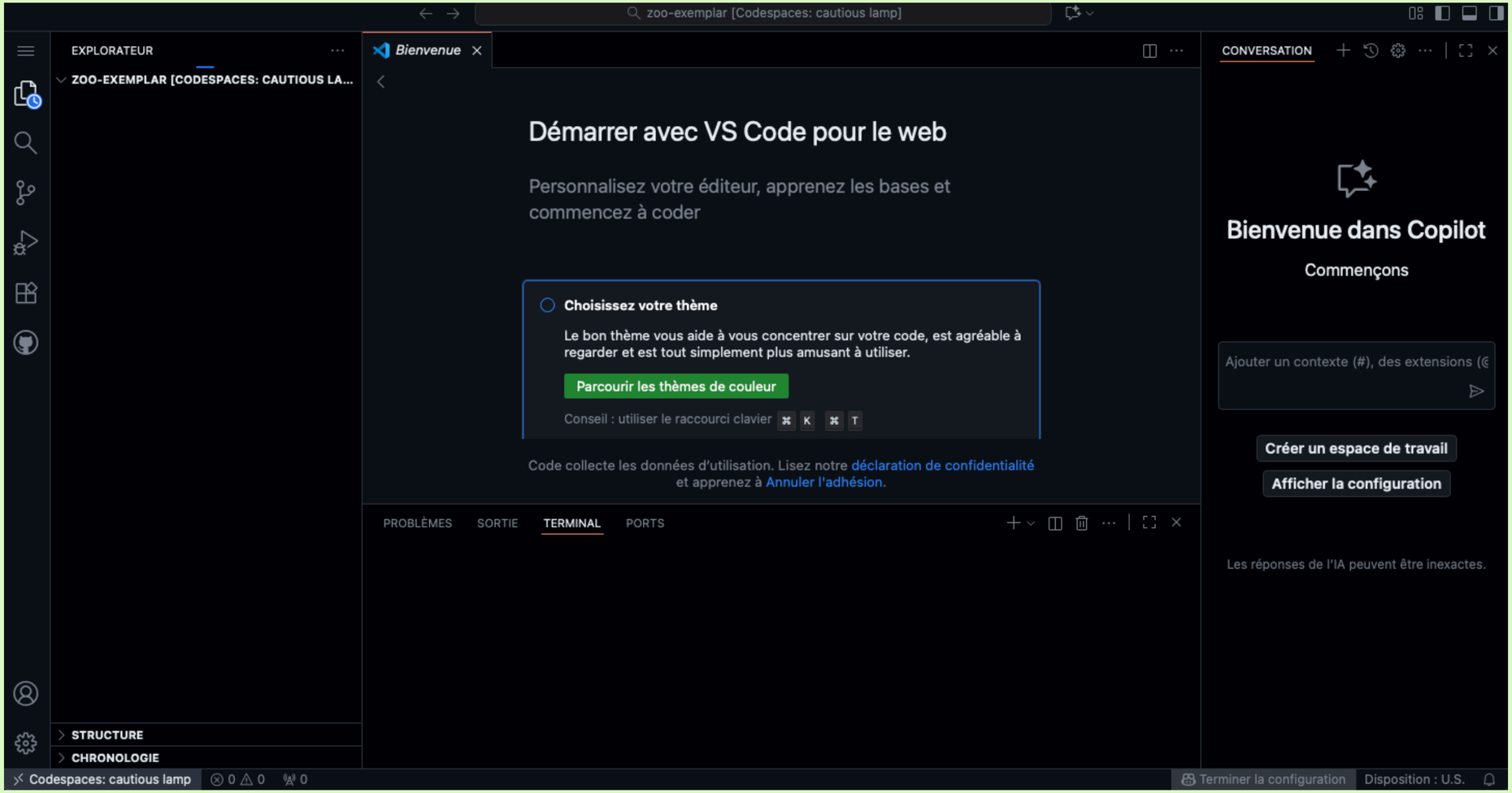


Click on the name of the IDE. Codespaces gives 2 random words to form the name.

2

OPEN UP YOUR IDE

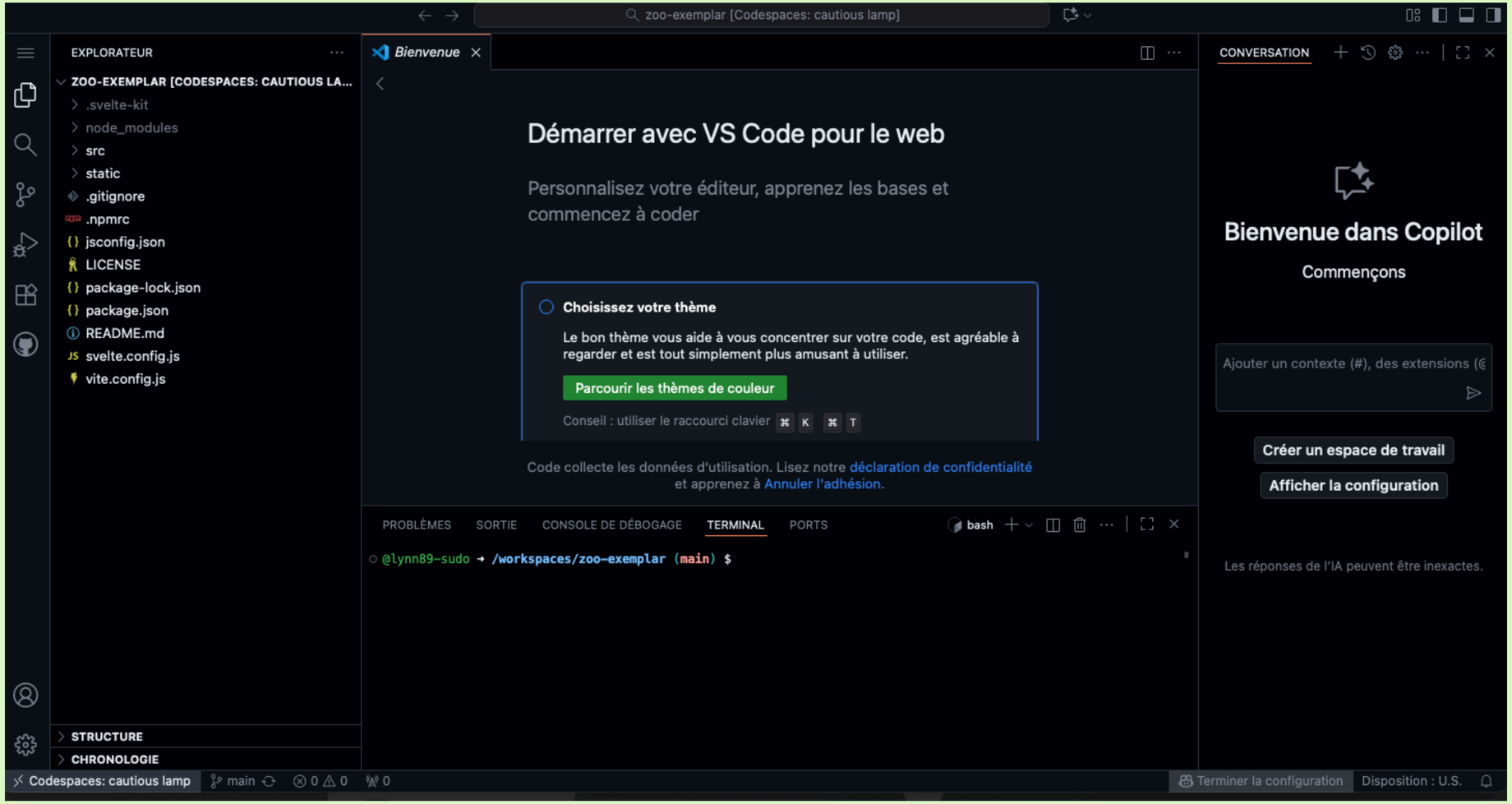
Give everything a second to load; patience is required for Codespaces ~~(see Dad, coding can teach you life skills)~~



2

OPEN UP YOUR IDE

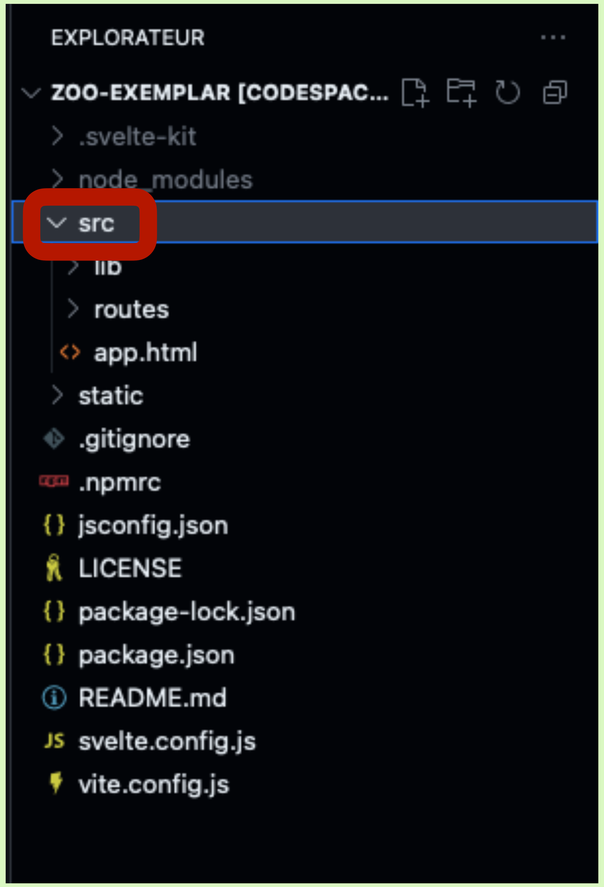
Okey dokey, everything is loaded now. Your IDE should look like this once loaded:



2

SRC FOLDER

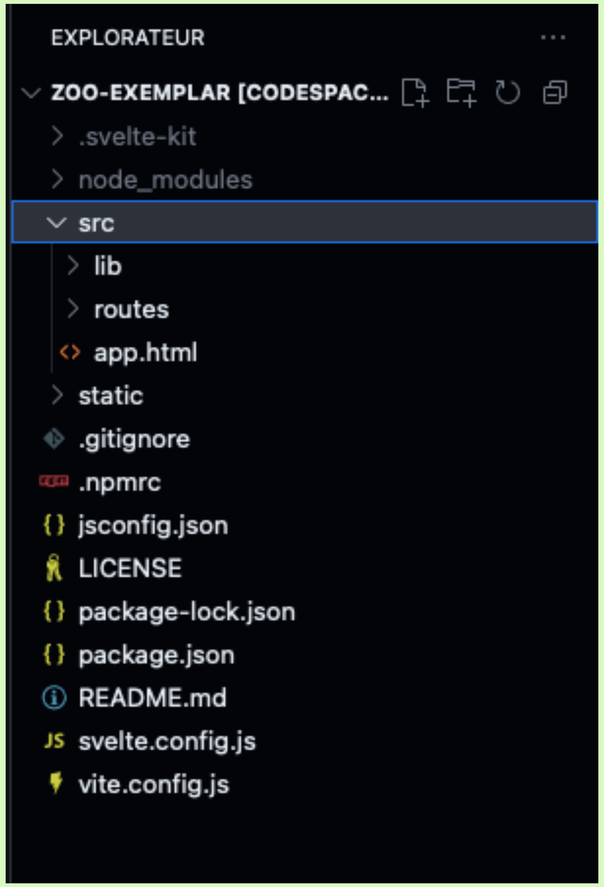
Open up the src folder



2

SRC FOLDER

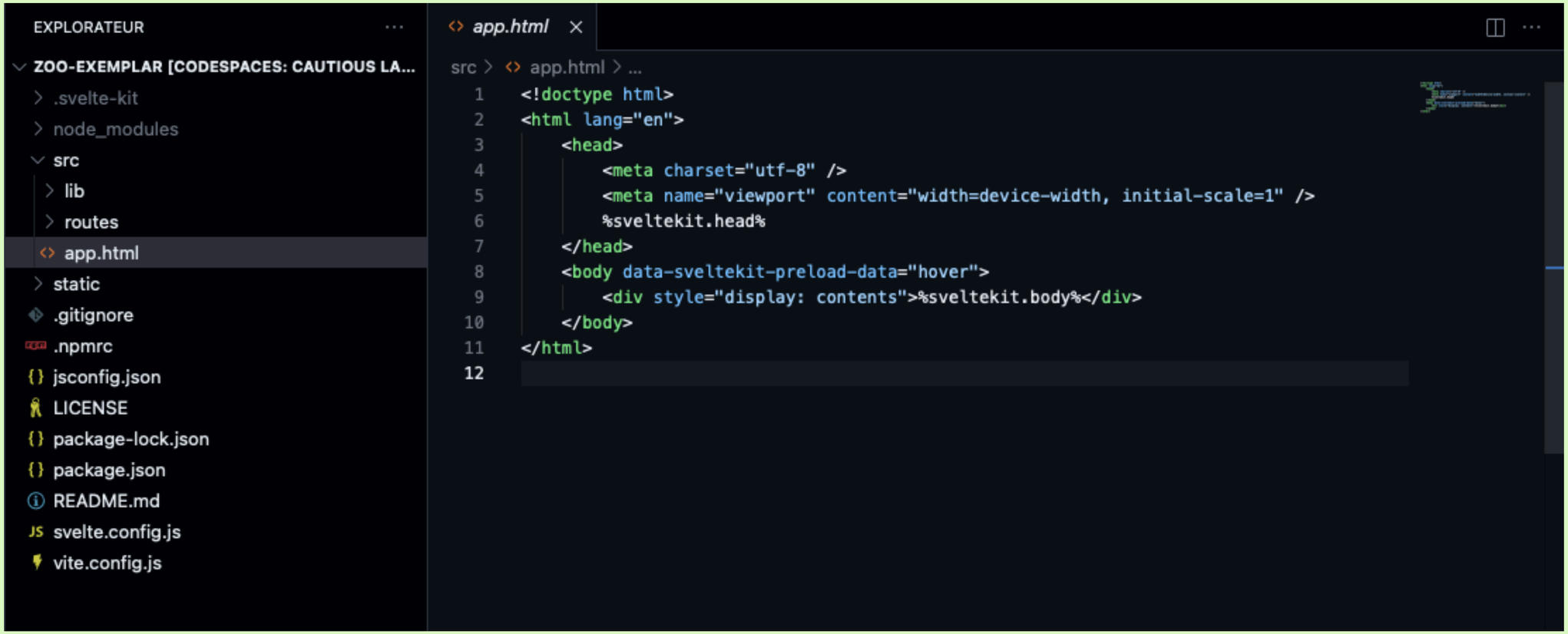
Below src, you should see lib, routes, and app.html



2

SRC FOLDER

app.html is what SvelteKit uses to render everything into HTML format. We won't need to touch this file for the most part.

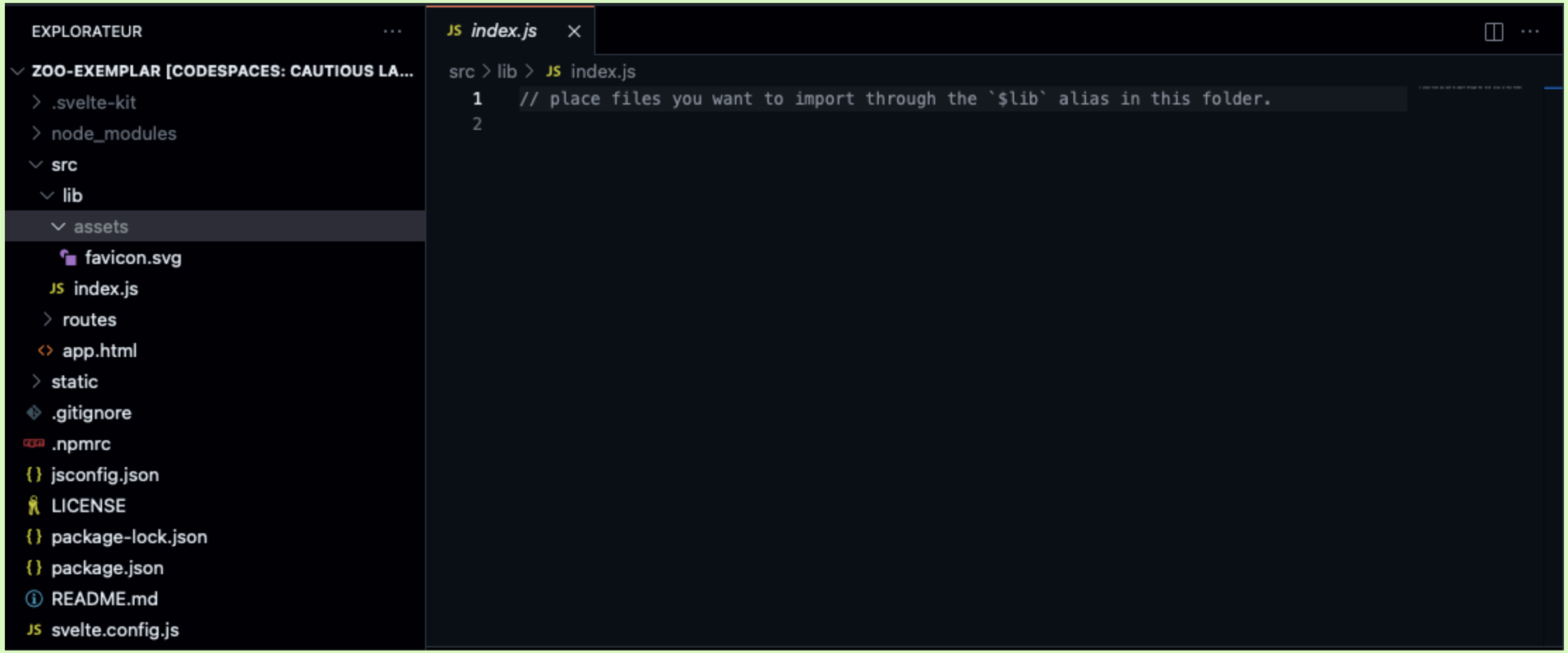


2

SRC FOLDER

Under the lib folder is assets, assets/favicon.svg, and index.js

Lib can only store text documents, such as JS, SVGs, or Svelte Components (chunks of reusable Svelte code, which we'll visit later) that we can import later



2

SRC FOLDER

Under routes is where the actual magic (yes, real magic ^w^) happens.

Routes contains all the content in our web pages - besides components - and also dictates the URL structure for the different pages. Inside routes is +page.svelte, which is the code for the homepage of our website.

(Your IDE may ask to download the Svelte extension if it's VSCode or Codespaces; choose yes)

2

SRC FOLDER

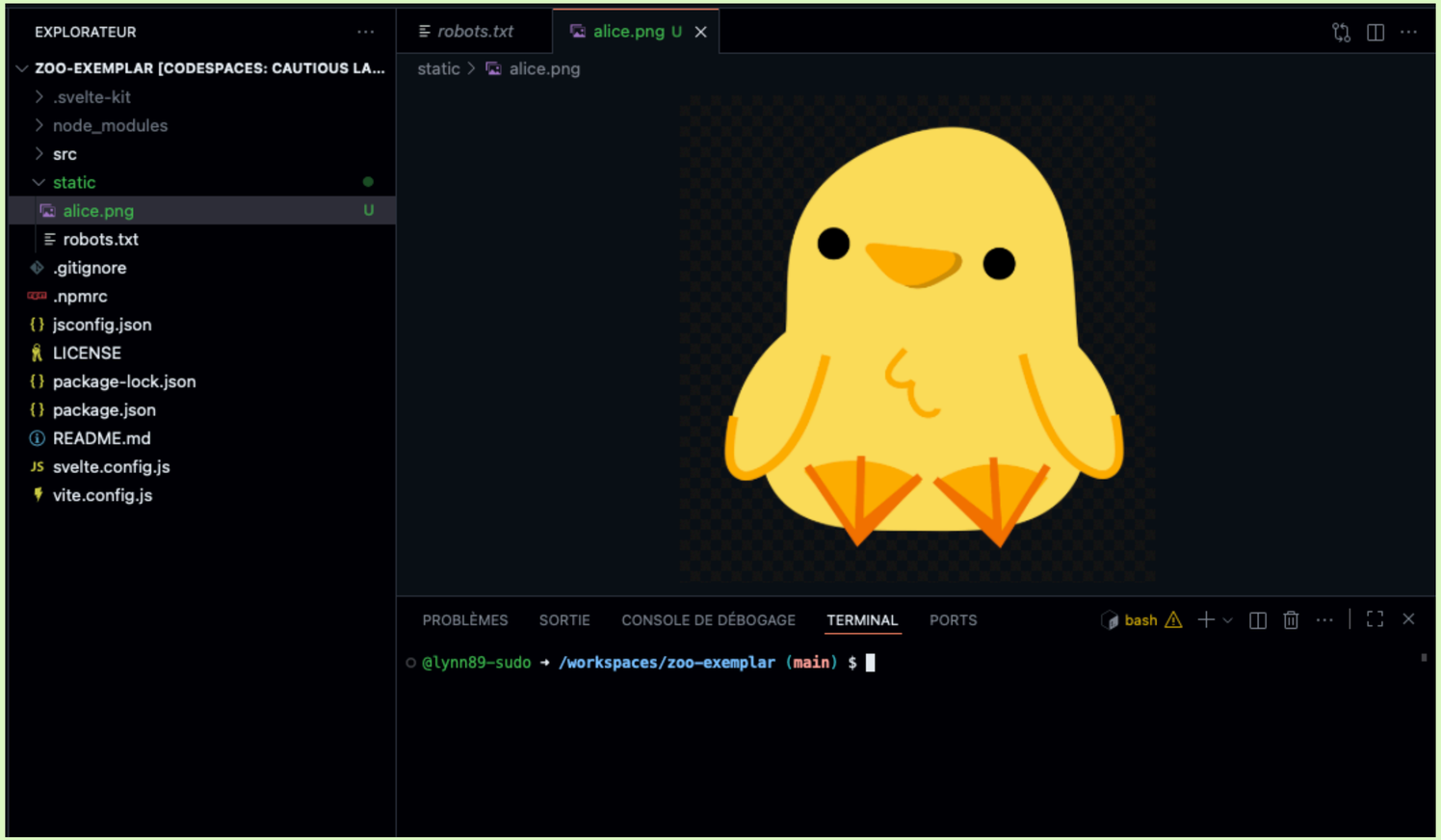
+layout.svelte contains code that gets applied to every page on our site. From favicon to <style> rulesets and Navbars, whatever is put here will render with the page's content. The page's content (stuff in +page.svelte) is defined by {@render children?.()}

(I use this very often to put my Google fonts and icons import links)

2

STATIC FOLDER

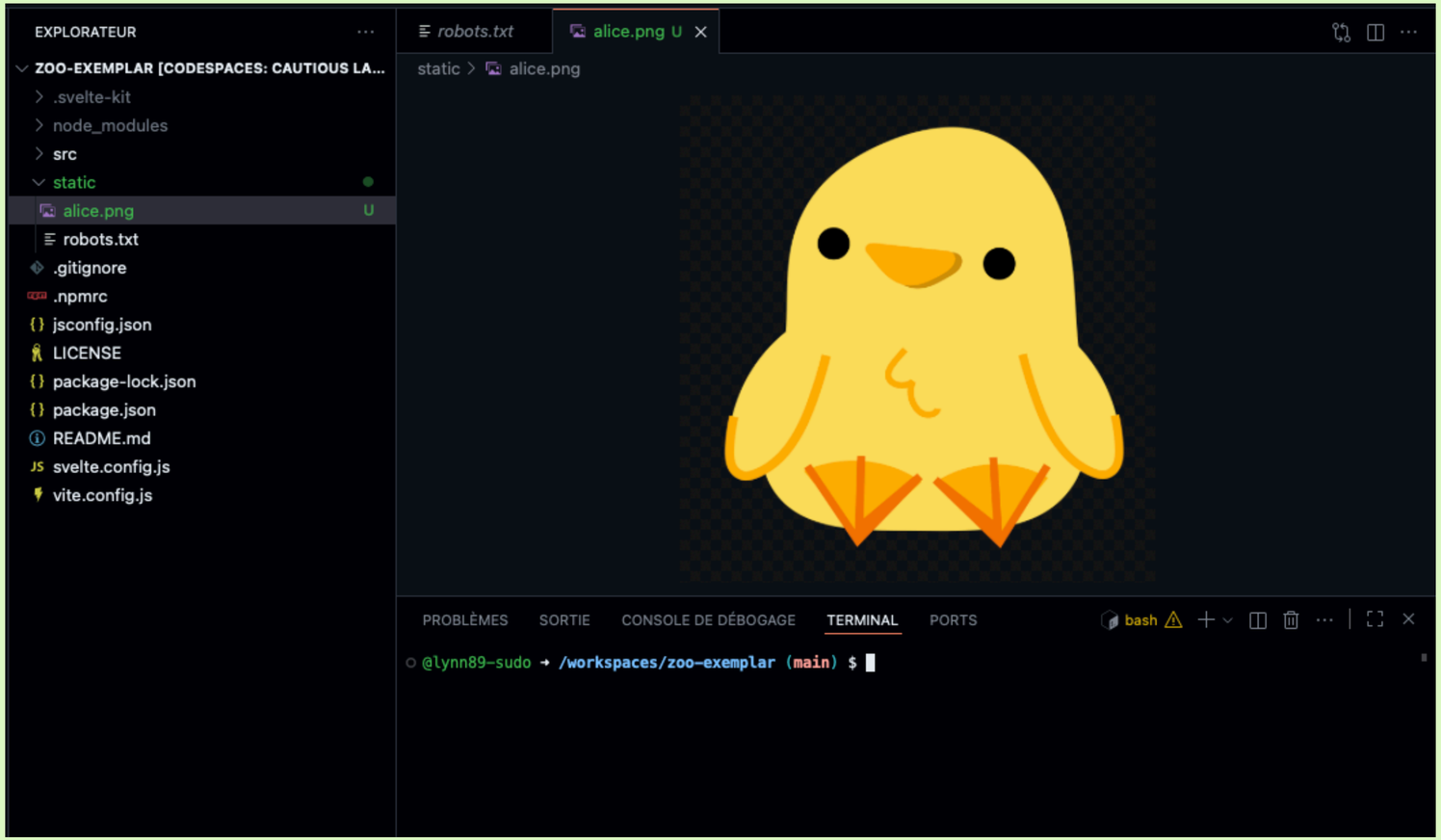
Close out of src and open up static. Static works similarly to lib; however, this is meant for all the images on your site. Ignore the robots.txt for now, just don't touch it and let it do its thing.



2

STATIC FOLDER

Go ahead and upload an image to the static folder! I've uploaded a picture of a duck I made, which I called Alice

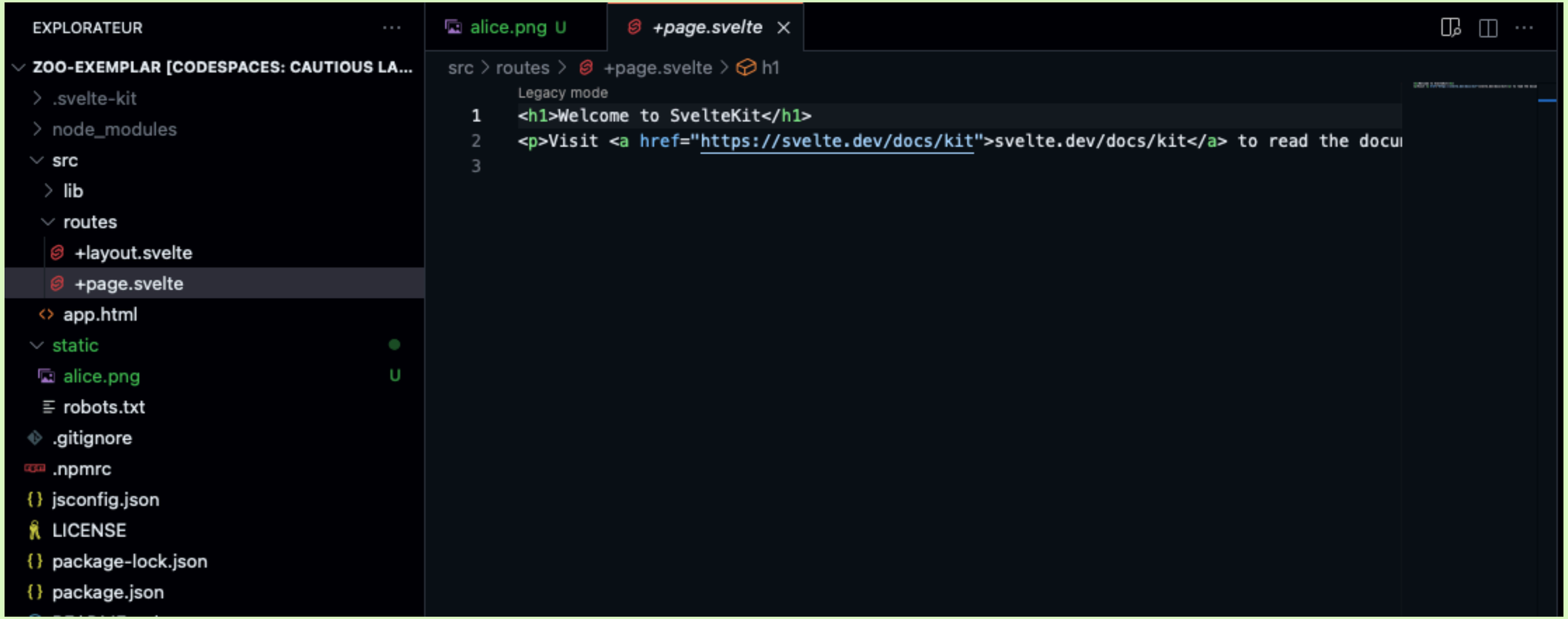


2

EDITING +PAGE.SVELTE

Go back to src/routes/+page.svelte now

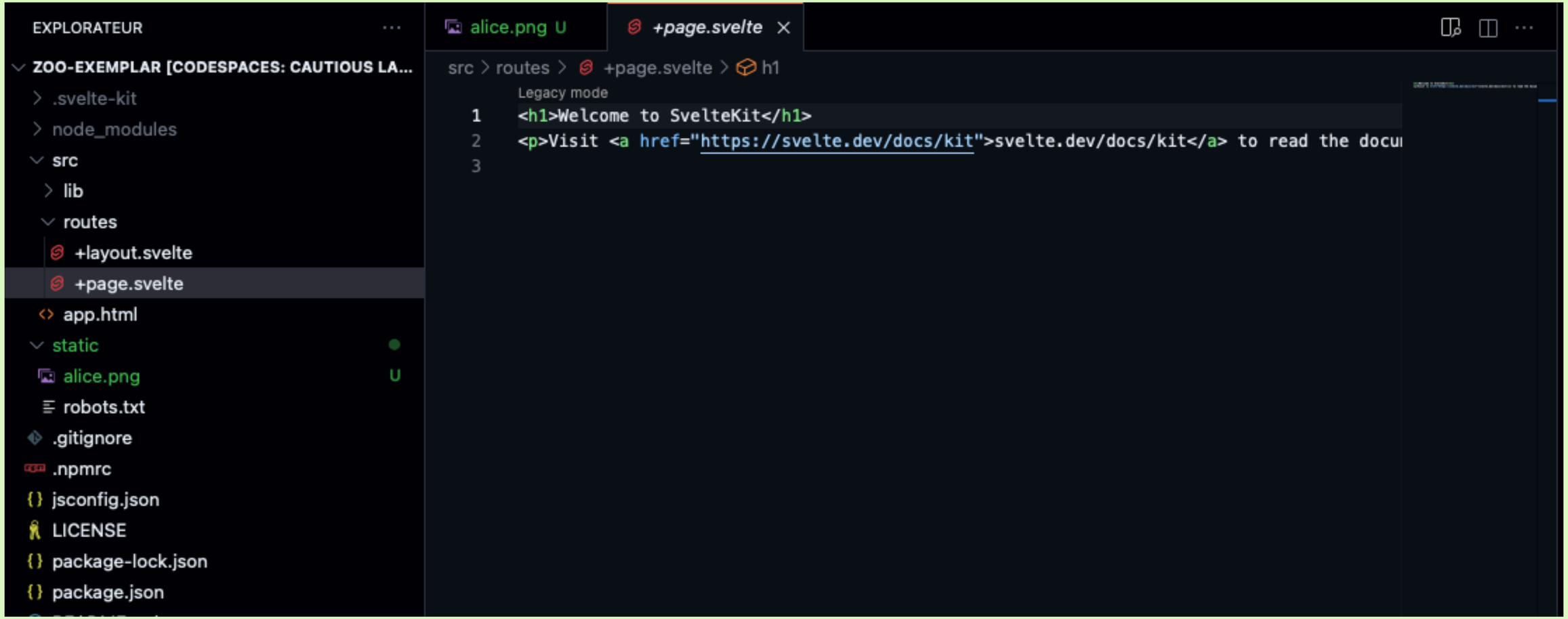
No need to close static



2

EDITING +PAGE.SVELTE

The format for a Svelte page is very similar to that of normal HTML. Let's start editing the page!



2

EDITING +PAGE.SVELTE

Now, below those tags, start writing your HTML body tags (but without <body>).
Mine looks something like this.

alice.png U +page.svelte M

src > routes > +page.svelte > h3

Legacy mode

```
1  <script>
2
3  </script>
4  <style>
5
6  </style>
7
8  <h1>Hello world!</h1>
9  <h3>This is Alice the ducky!</h3>
10
```

2

EDITING +PAGE.SVELTE

Time to import our picture. With static, all we have to do to import is call it as such:

alice.png U +page.svelte 1, M

src > routes > +page.svelte > img

Legacy mode

```
<script>

</script>
<style>

</style>

<h1>Hello world!</h1>
<h3>This is Alice the ducky!</h3>
<img src = "alice.png"/>
```

2

EDITING +PAGE.SVELTE

Unless we create folders inside static to store our images, all we have to do for the src is the filename itself. With a folder, it would look like this:

[folder-name]/alice.png

alice.png U +page.svelte 1, M

src > routes > +page.svelte > img

Legacy mode

```
1  <script>
2
3  </script>
4  <style>
5
6  </style>
7
8  <h1>Hello world!</h1>
9  <h3>This is Alice the ducky!</h3>
10 <img src = "alice.png"/>
11
```

2

EDITING +PAGE.SVELTE

Let's fix this yellow squiggly line now ~~(cause we don't like things that ain't straight)~~

Svelte enforces higher standards of accessibility practices, which allows sites to integrate with assistive technology, so we have to add an alt attribute to our image

2

VIEWING OUR WEBSITE

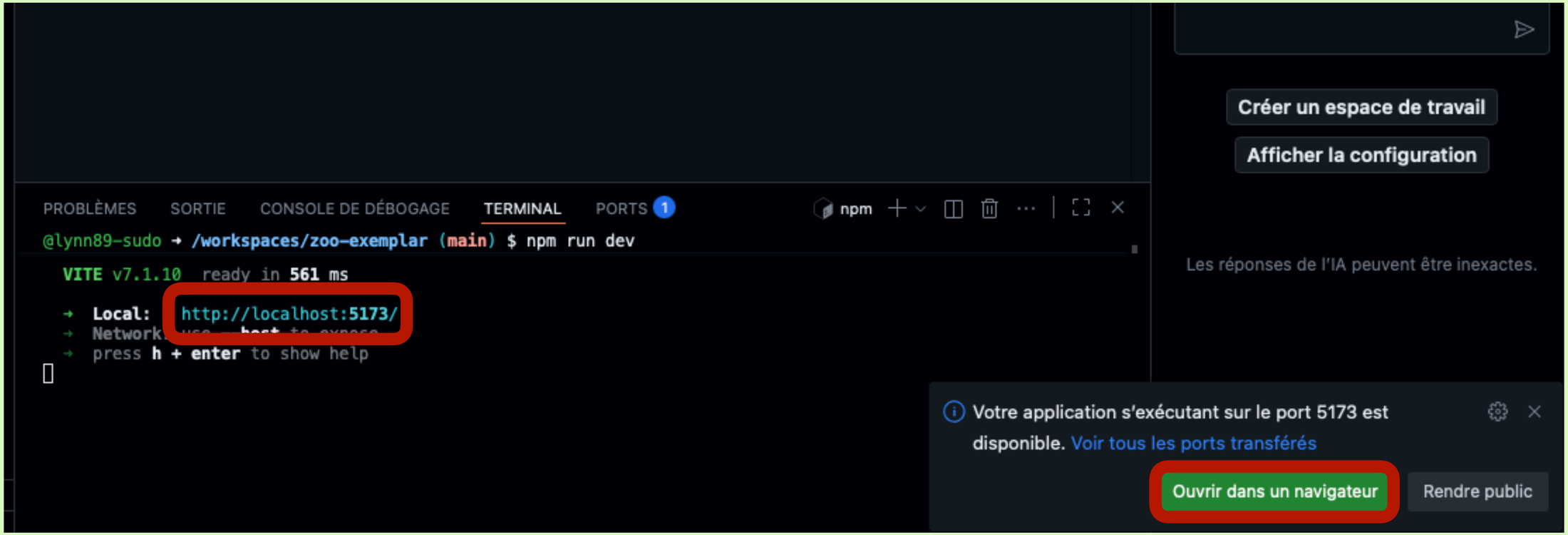
Alright, we've edited +page.svelte; however, how do we really know what this looks like? Unless you're omniscient (no, you're not), we'll need to preview the website.

In the terminal, run "npm run dev"

2

VIEWING OUR WEBSITE

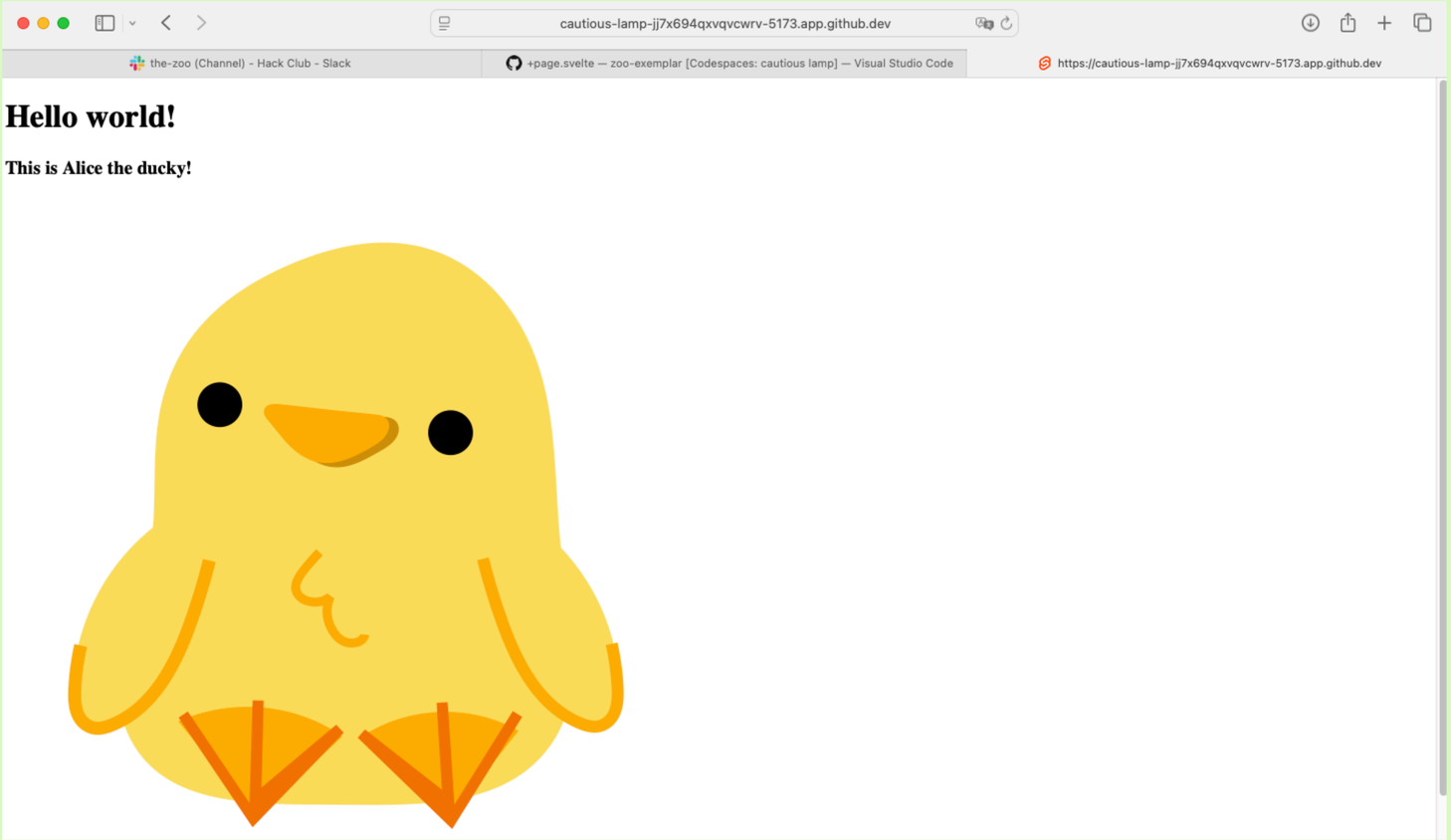
Once you run it, your screen should look something like this. If you're on VSCode, just hover over that http://localhost link and then cmd + click on it. If on Codespaces, click on that "Open in Browser" button.



2

VIEWING OUR WEBSITE

We've done it! We've completed our first Svelte page! Give yourself a pat on the back, unless you don't have a back that is. Then maybe just pat yourself on the head (I hope you have a head)!



2

VIEWING OUR WEBSITE

From here, you can go back and add more to your page, or add in some <style> rules! I don't advise adding to the <script> tag just yet; however, if you know what you are doing, you should be fine!

```
<script>

</script>
<style>
    * {
        font-family: Futura, Montserrat;
        text-align: center;
    }
    img {
        width: 40%;
        display: block;
        margin: 0 auto;
    }
</style>

<h1>Hello world!</h1>
<h3>This is Alice the ducky!</h3>
<img src = "alice.png" alt="Alice the duck"/>
```

2

COMMIT CHANGES

Before you finish, make sure you commit all your changes like we did in the first section! You may need to create a new terminal with the + button if you used npm run dev (which you did...)

```
● @lynn89-sudo → /workspaces/zoo-exemplar (main) $ git add .
● @lynn89-sudo → /workspaces/zoo-exemplar (main) $ git commit -m "Added in Alice"
  [main 93804dd] Added in Alice
   2 files changed, 18 insertions(+), 2 deletions(-)
   create mode 100644 static/alice.png
● @lynn89-sudo → /workspaces/zoo-exemplar (main) $ git push origin main
  Enumerating objects: 12, done.
  Counting objects: 100% (12/12), done.
  Delta compression using up to 2 threads
  Compressing objects: 100% (7/7), done.
  Writing objects: 100% (7/7), 47.02 KiB | 23.51 MiB/s, done.
  Total 7 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
  remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
  To https://github.com/lynn89-sudo/zoo-exemplar
     e2de0de..93804dd  main -> main
```

ALL DONE!

Seriously, take a break before we jump into the next part ^w^

3

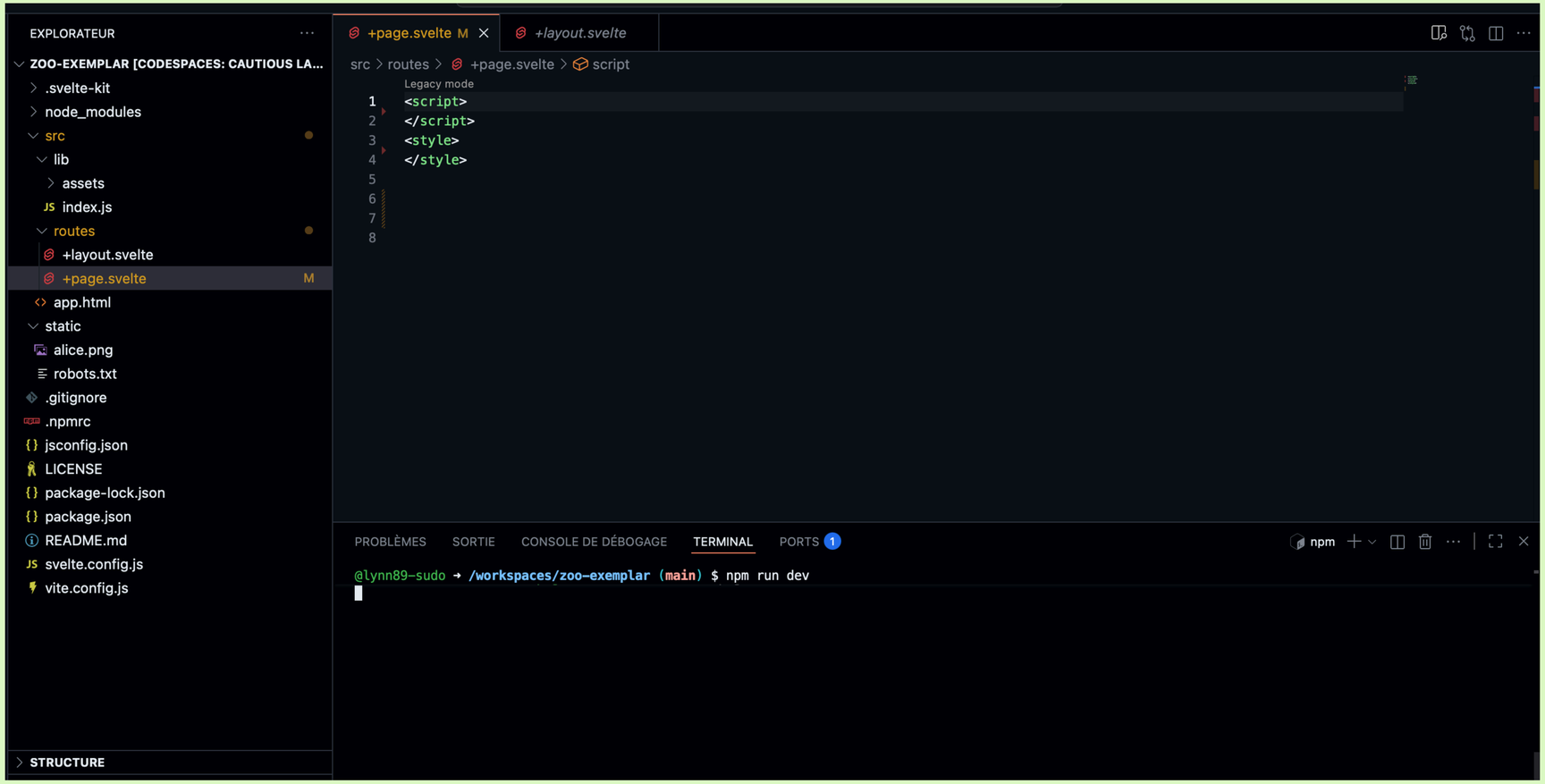
Svelte Components and Runes

3

STARTING OUR SUBMISSION

What I've showed you in the previous sections of the tutorial were just to showcase how Svelte works at a base level. Now, let's wipe that clean and start on our actual submission!

If you want, you can keep the imported image and the <style> rules that you made ^w^



3

IMPORTING A SVELTE COMPONENT

Components are a very useful feature in Svelte, and you're required to use them for this workshop!

Let's import one in! After that, we'll make one ourselves!

raw.githubusercontent.com

```
<style>
    .bannerIcon {
        vertical-align: -0.07em;
        font-size: 24px !important;
    }
    @font-face {
        font-family: 'Phantom Sans';
        src: url('https://assets.hackclub.com/fonts/Phantom_Sans_0.7/Bold.woff')
            format('woff'),
            url('https://assets.hackclub.com/fonts/Phantom_Sans_0.7/Bold.woff2')
            format('woff2');
        font-weight: bold;
        font-style: normal;
        font-display: swap;
    }

    #zoo_linkBanner {
        background-color: rgb(29, 65, 29);
        padding: 10px;
        text-align: center;
        vertical-align: center;
        position: fixed;
        left: 0;
        right: 0;
        top: 0;
        height: 65px;
        z-index: 1000;
        font-size: 15px;
        transition: box-shadow 0.3s ease-in-out;
        font-family: Phantom Sans, sans-serif;
        text-decoration: none;
        color: white;
    }
    #zoo_linkBanner:hover {
        cursor: pointer;
        box-shadow: 0px 0px 50px 10px rgba(0, 0, 0, 0.551);
    }
</style>

<a href = "https://zoo.hackclub.com"><div id="zoo_linkBanner">
    <h1>CHECK OUT MORE AT THE ZOO <span class="material-symbols-outlined bannerIcon">pets</span> </h1>
</div></a>
<div style="height: 100px"></div>
```

3

IMPORTING A SVELTE COMPONENT

Copy all the code in this file:

https://raw.githubusercontent.com/lynn89-sudo/red-panda/refs/heads/main/src/lib/Zoo.svelte

raw.githubusercontent.com

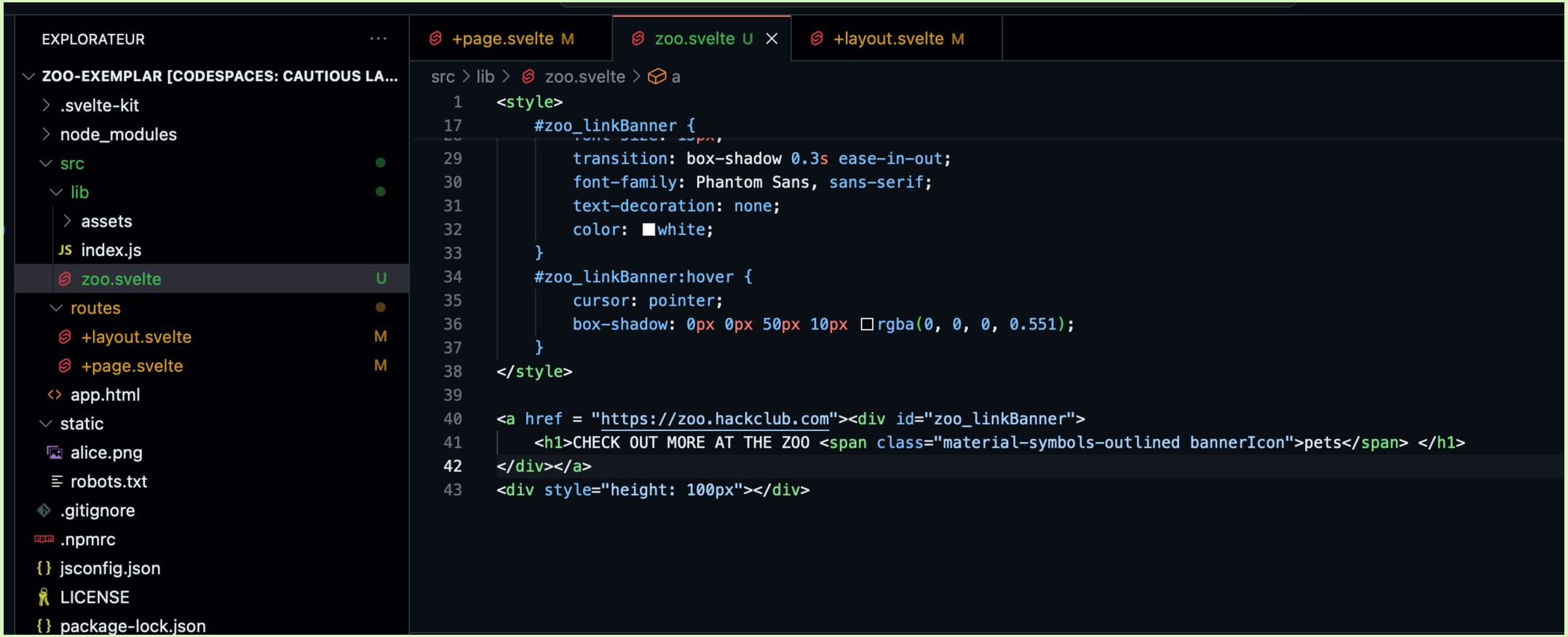
```
<style>
    .bannerIcon {
        vertical-align: -0.07em;
        font-size: 24px !important;
    }
    @font-face {
        font-family: 'Phantom Sans';
        src: url('https://assets.hackclub.com/fonts/Phantom_Sans_0.7/Bold.woff')
            format('woff'),
            url('https://assets.hackclub.com/fonts/Phantom_Sans_0.7/Bold.woff2')
            format('woff2');
        font-weight: bold;
        font-style: normal;
        font-display: swap;
    }

    #zoo_linkBanner {
        background-color: rgb(29, 65, 29);
        padding: 10px;
        text-align: center;
        vertical-align: center;
        position: fixed;
        left: 0;
        right: 0;
        top: 0;
        height: 65px;
        z-index: 1000;
        font-size: 15px;
        transition: box-shadow 0.3s ease-in-out;
        font-family: Phantom Sans, sans-serif;
        text-decoration: none;
        color: white;
    }
    #zoo_linkBanner:hover {
        cursor: pointer;
        box-shadow: 0px 0px 50px 10px rgba(0, 0, 0, 0.551);
    }
</style>

<a href = "https://zoo.hackclub.com"><div id="zoo_linkBanner">
    <h1>CHECK OUT MORE AT THE ZOO <span class="material-symbols-outlined bannerIcon">pets</span> </h1>
</div></a>
<div style="height: 100px"></div>
```

3

IMPORTING A SVELTE COMPONENT

Go into the lib folder

Create a svelte file; I'm calling it zoo.svelte, but that doesn't matter. Paste everything into there!



3

IMPORTING A SVELTE COMPONENT

Since this component uses Google Fonts icons, we'll need to import that in. The best way to do this is inside +layout.svelte

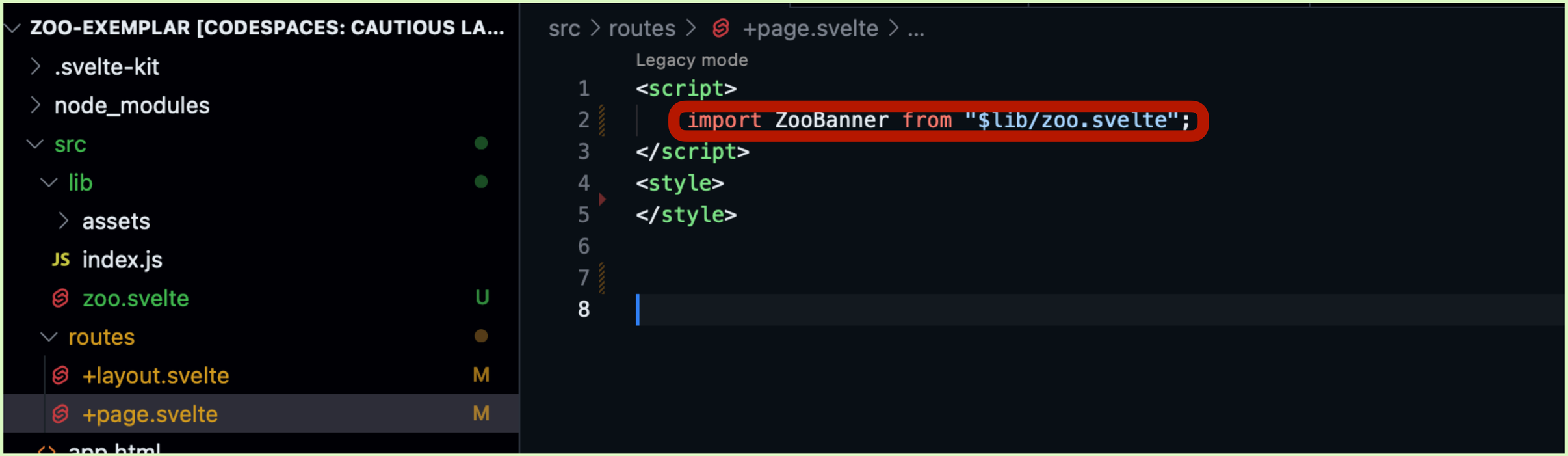
If your confused about what link you can use to statically import, go to fonts.google.com and search for "pets" in the icons

3 IMPORTING A SVELTE COMPONENT

Time to use the component! Head on over to your +page.svelte and we'll set it up.

What you need to do first is import it. Follow the model below, replacing the name I gave the file with your own.

While it doesn't matter what you put in place of "ZooBanner", it is critical that you have the first letter of the name be uppercase.



3

IMPORTING A SVELTE COMPONENT

Finally, we can use the component. To use it, you can call it by whatever you named it in place of “ZooBanner” and as a self-closing tag.

+page.svelte M | zoo.svelte U | +layout.svelte M

src > routes > +page.svelte > ...

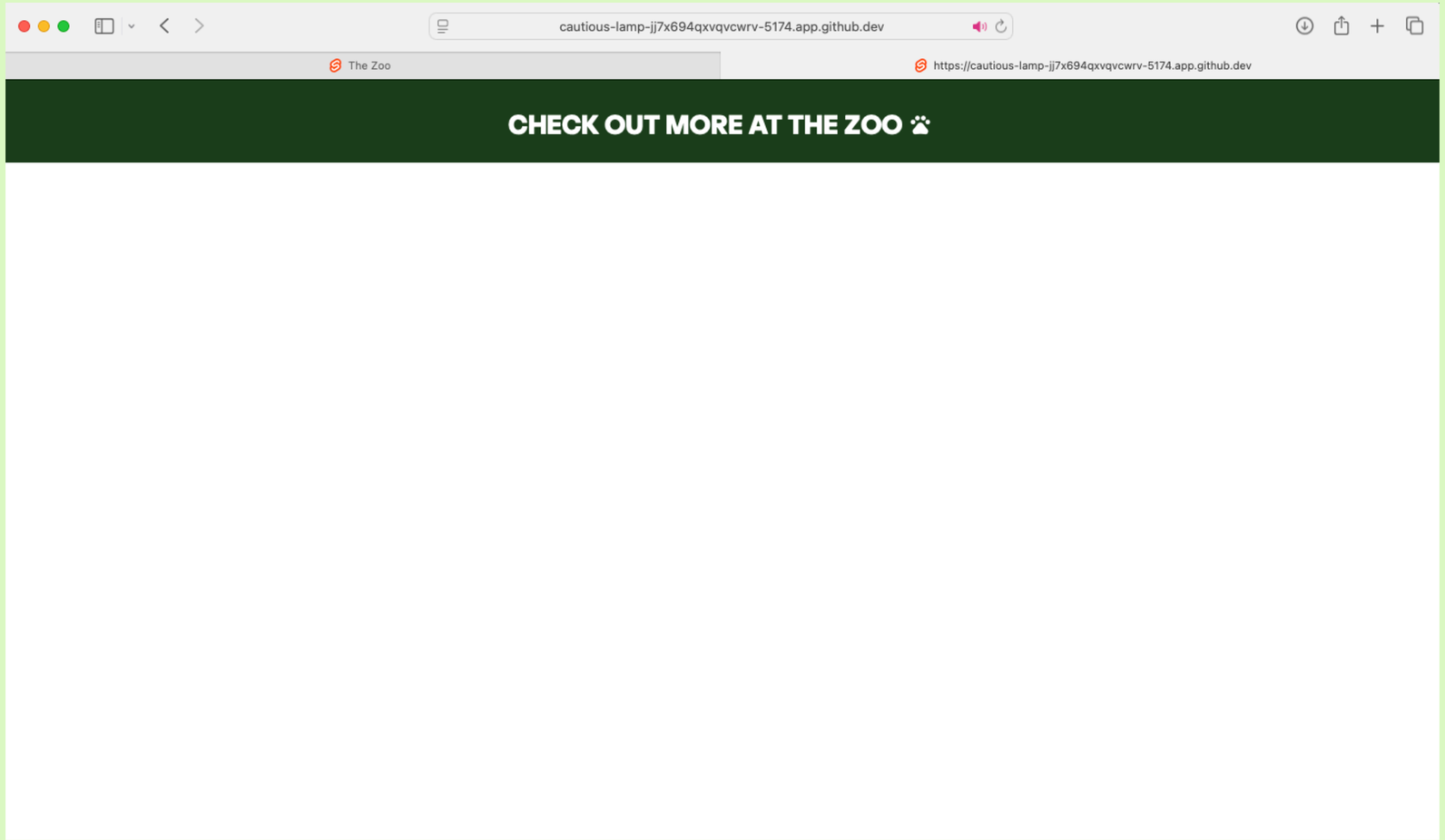
Legacy mode

```
1  <script>
2      import ZooBanner from "$lib/zoo.svelte";
3  </script>
4  <style>
5  </style>
6  <ZooBanner />
7
8
```

3

IMPORTING A SVELTE COMPONENT

If you see this in your preview, you've done everything right!



3

IMPORTING A SVELTE COMPONENT

Since the requirement is to include this on every single page of the site, I'm actually going to remove the instance of this component and put it in +layout.svelte

```
Runes mode
<script>
    import favicon from '$lib/assets/favicon.svg';

    let { children } = $props();

    import ZooBanner from "$lib/zoo.svelte";
</script>

<svelte:head>
    <link rel="icon" href={favicon} />
    <link rel="stylesheet" href="https://fonts.googleapis.com/css2?family=Material+Symbols+Outlined:opsz,wght,FILL,GRAD@24,400,0
</svelte:head>

<ZooBanner />
{@render children?.()}
```

3

MAKING A SVELTE COMPONENT

Time to make your own component! Some ideas you could think of are navbars and footers. Don't worry about complicated ideas right now; if you want, you can make plenty of components afterwards!

Once you finish your svelte file in lib, follow the same steps we took before. Consider putting it in +layout.svelte vs. +page.svelte! You should put it in layout if your component should be shown on every page.

TAKE

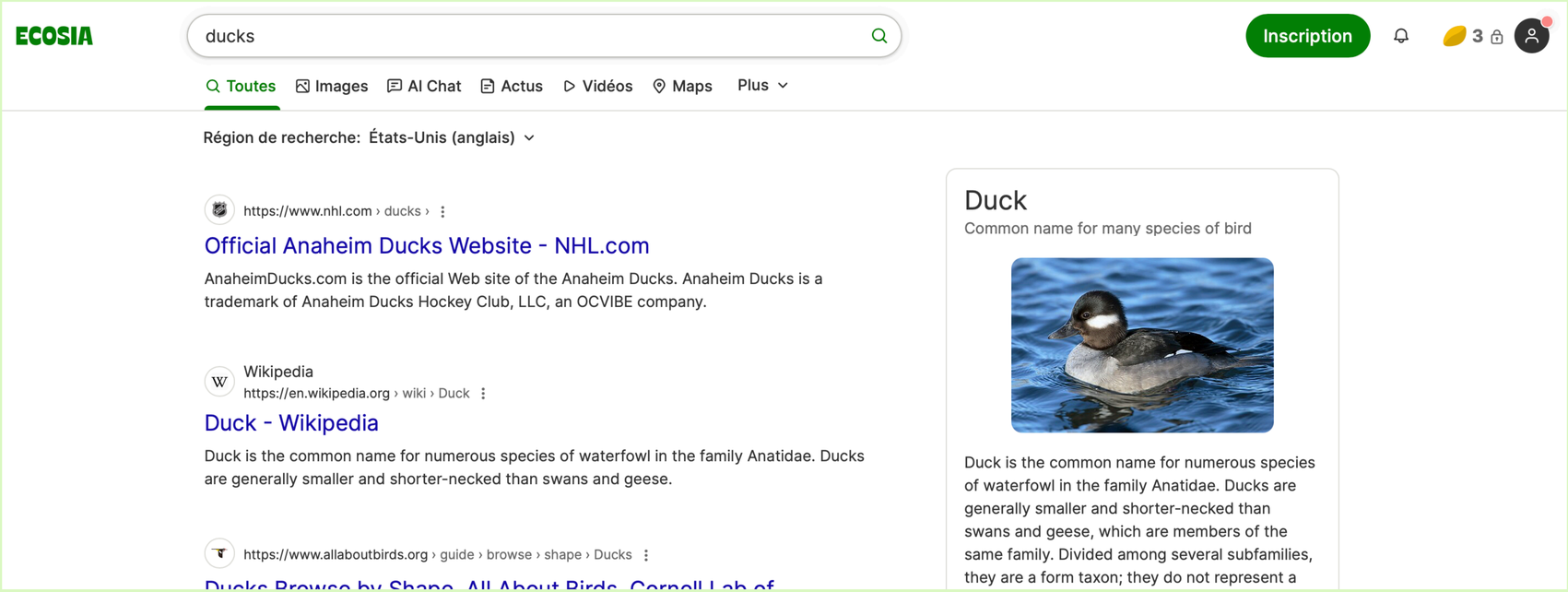
3-7 MINS

3

MAKING A FACTS PANEL

Good job making your first component! Now, let's move onto Svelte Runes (the 2nd core part of Svelte), and we'll also use this opportunity to make our dynamic facts panel.

Hopefully, by this point, you've chosen your animal for your site. If not, do that now. After you've done that, do a bit of research and get at least 3 facts on your animal! Include some interesting facts, don't just make it boring!



3

MAKING A FACTS PANEL

Once you have your 3 or more facts, put it into a list in your script!

If you're good with JSON, you could add onto this by adding arrays instead of strings to contain more information: title, icons, etc. By the end, you need to do at least one thing to make this unique (but I don't ^w^)!

```
<script>
    let facts = [
        "Ducks are omnivores – they eat plants, insects, worms, snails, and even small fish.",
        "Ducks have webbed feet, which act like paddles, making them excellent swimmers.",
        "Ducks can sleep with one eye open – half their brain stays awake to watch for predators.",
        "Ducks are omnivores – they eat plants, insects, worms, snails, and even small fish.",
        "Ducklings can swim and feed themselves within hours after hatching!"
    ]
</script>
<style>
</style>
```

3

MAKING A FACTS PANEL

Now, let's make the for our facts panel and set the variable "fact".

We'll use a $state rune to declare fact. When we use state with a variable, this allows us to easily change aspects of the UI that contain the information in the variable. For example, if fact is assigned to a certain fact in our list, every time we change fact it will be replaced on the user screen! This isn't exactly what we'll do in the end, but we'll be doing something similar!

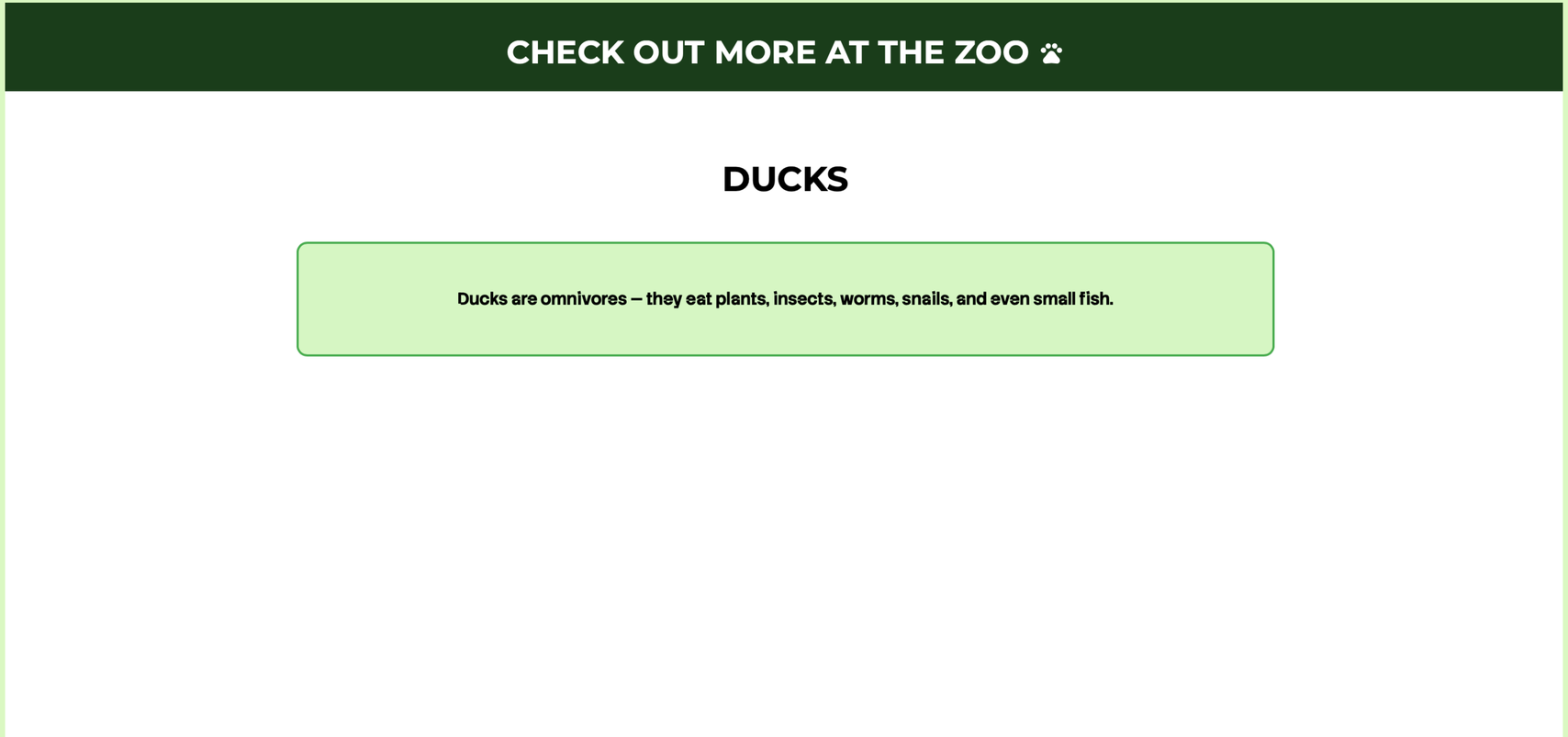
We don't need to set fact to anything right now; however, when we do the value will go inside the parentheses of $state

```
<script>
    import "$lib/styles.css";

    let facts = [
        "Ducks are omnivores – they eat plants, insects, worms, snails, and even small fish.",
        "Ducks have webbed feet, which act like paddles, making them excellent swimmers.",
        "Ducks can sleep with one eye open – half their brain stays awake to watch for predators.",
        "Ducks are omnivores – they eat plants, insects, worms, snails, and even small fish.",
        "Ducklings can swim and feed themselves within hours after hatching!"
    ];
    let fact = $state();
</script>
<style>
</style>
<br>
<h1>DUCKS</h1>
<br>
<div id="factContainer">

</div>
```

3

MAKING A FACTS PANEL

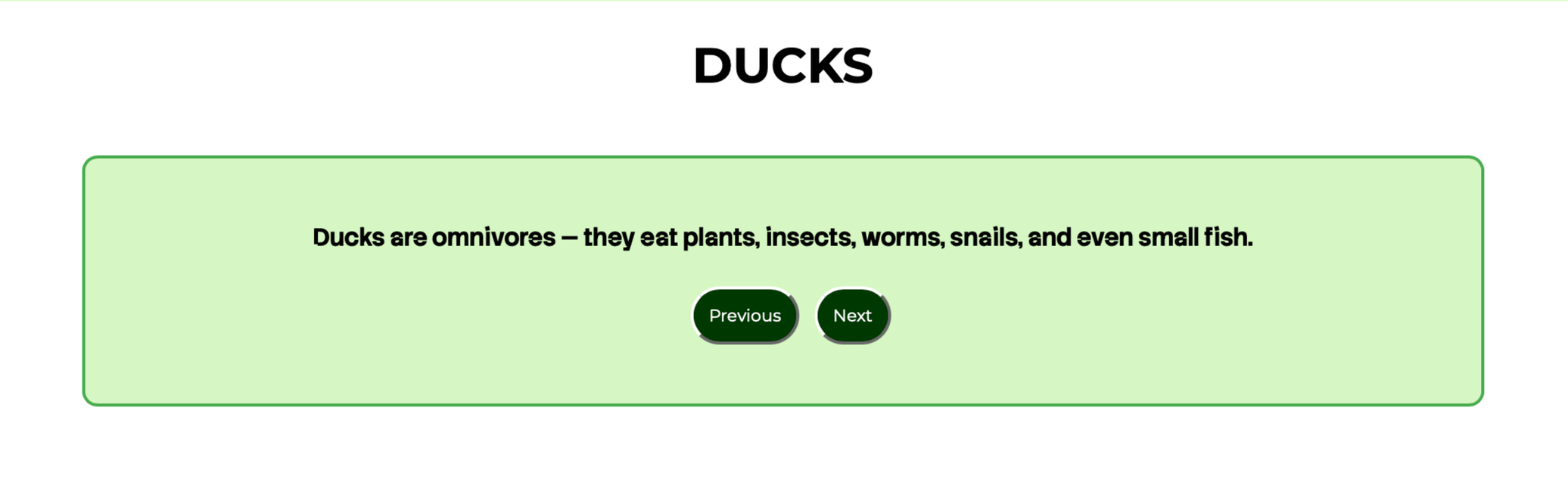
When you load the preview, you should see something like this! I added in some CSS; however, if you see a fact, you're on track! (amazing rhyme, I know)



3

MAKING A FACTS PANEL

Time to add in those buttons so we can cycle through. From here, the tutorial becomes more independent! Use your JS knowledge and put your two (or three) brain cells together! Remember, all you have to do to change the factText is change the index we set in the fact variable!



ALL DONE!

Once you're done with that facts panel, commit your changes and take a nap!

4

Multiple Pages

4

MAKING A NEW PAGE

Like I said before, I'm making you put your brain cells to use on how to meet the rest of the requirements (zoo.hackclub.com/requirements); however, I'll show you how you setup multiple pages for Svelte

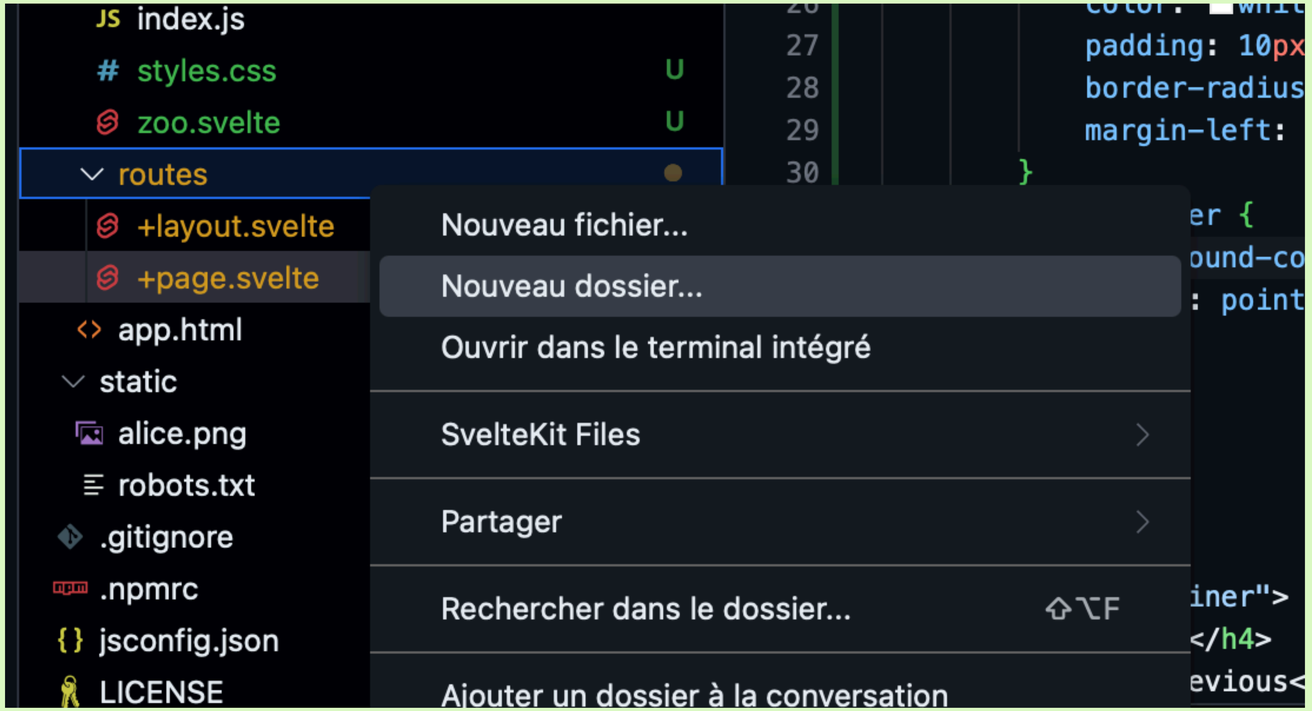
Make sure you're in the routes folder

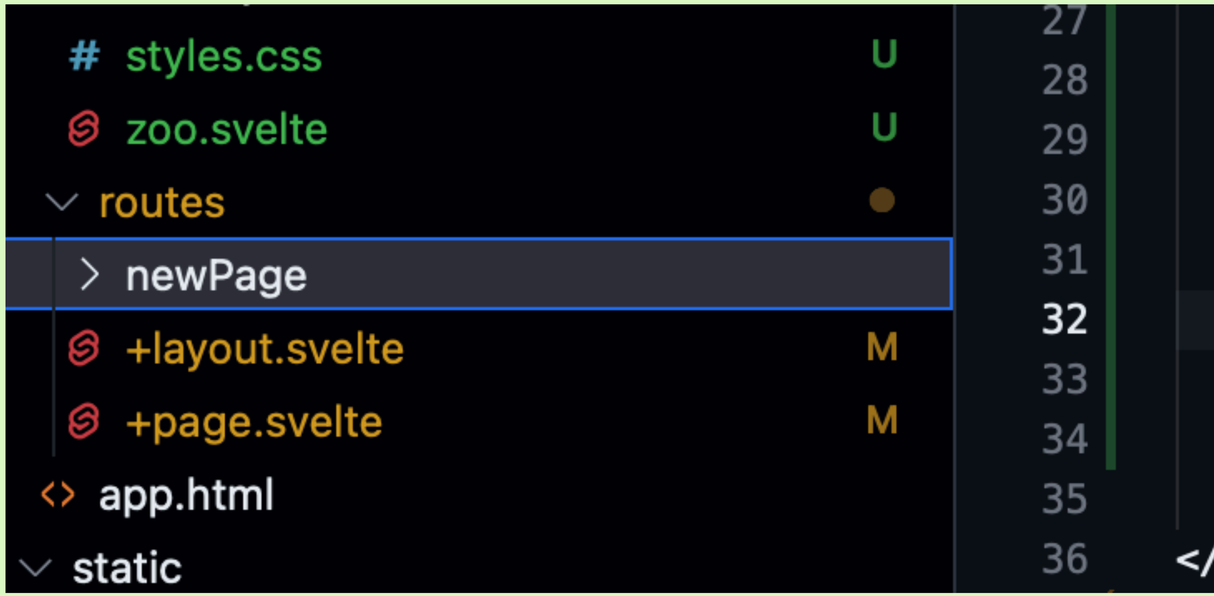
4

MAKING A NEW PAGE

Under routes, make a new folder. You should call this folder whatever you want to name the subpage

For example: zoo.hackclub.com/requirements would be configured by setting the folder name to “requirements”



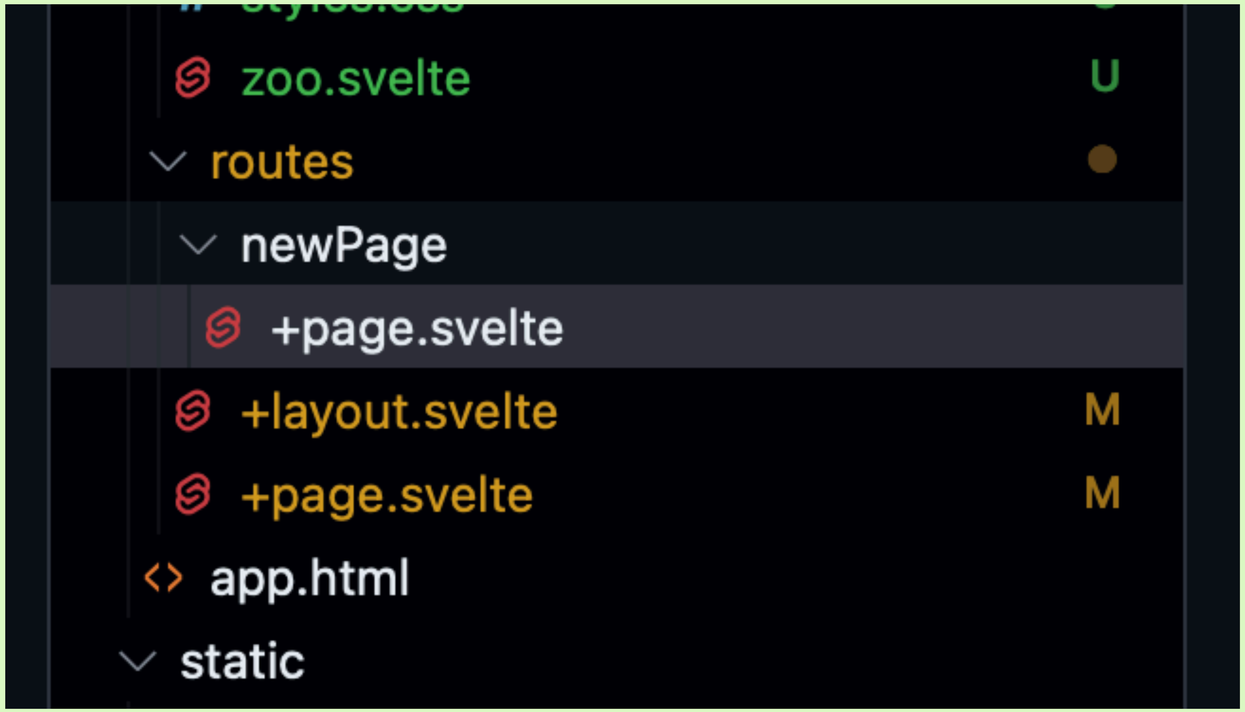


4

MAKING A NEW PAGE

Inside this folder, make a +page.svelte

For my purposes, I'm going to add some filler text. Just add something into here for your page

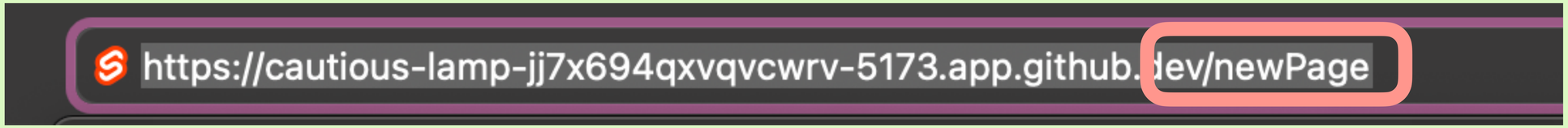



```
<script>
    import "$lib/styles.css";
</script>

<h1>Hello world</h1>
<h3>It is I, Me</h3>
```

4

PREVIEWING A NEW PAGE

To see our page, just navigate to the subpage in the browser a you would normally!



ALL DONE!

Keep up the good work!

5

Deployment

CONTINUE AFTER FINISHING YOUR SUBMISSION. MAKE SURE TO REGULARLY COMMIT YOUR CHANGES, EVEN BEFORE DEPLOYING

5

GREAT JOB

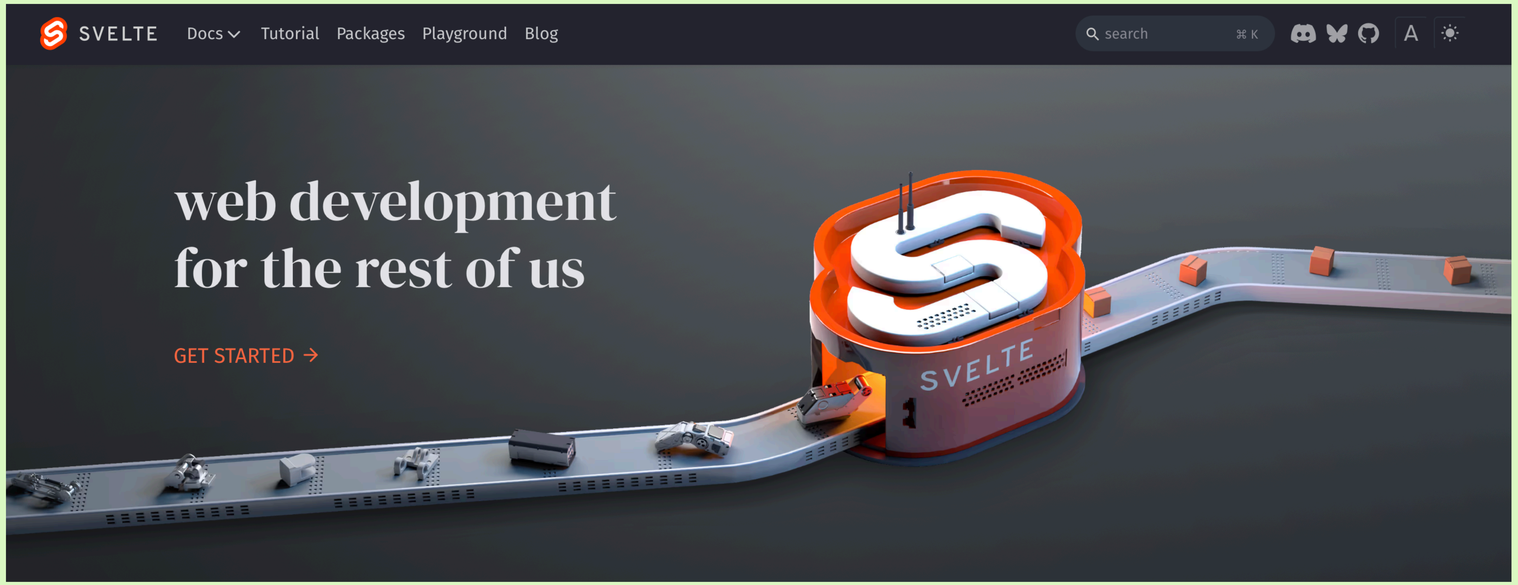
Give yourself a pat on the back for completing your submission.
Double (or triple) check that your project meets all the requirements on zoo.hackclub.com/requirements

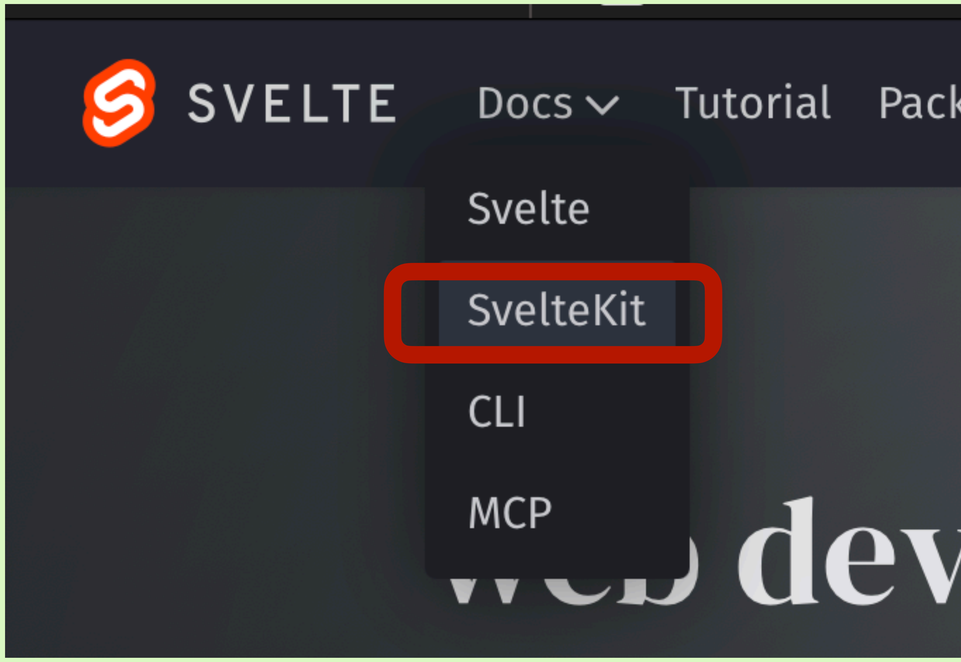
5

STATIC ADAPTER

Alright, now that that's out of the way, it's time to deploy ~~(to the military)~~!!! YAY!

If you haven't already been using the Svelte documentation, you're missing out. Let's use it now. Go to svelte.dev and under docs, go to SvelteKit





5

STATIC ADAPTER

Under Build and Deploy, go to Adapters. Then, go to adapter-static

Official adapters exist for a variety of platforms — these are documented on the following pages:

- `@sveltejs/adapter-cloudflare` for Cloudflare Workers and Cloudflare Pages
- `@sveltejs/adapter-netlify` for Netlify
- `@sveltejs/adapter-node` for Node servers
- `@sveltejs/adapter-static` for static site generation (SSG)
- `@sveltejs/adapter-vercel` for Vercel

Additional community-provided adapters exist for other platforms.

5

STATIC ADAPTER

Scroll down to (or click on) GitHub Pages

Copy the code that is provided for the config

GitHub Pages

When building for GitHub Pages, if your repo name is not equivalent to `your-username.github.io`, make sure to update `config.kit.paths.base` to match your repo name. This is because the site will be served from `https://your-username.github.io/your-repo-name` rather than from the root.

You'll also want to generate a fallback `404.html` page to replace the default 404 page shown by GitHub Pages.

A config for GitHub Pages might look like the following:

svelte.config.js

```
import adapter from '@sveltejs/adapter-static';

/** @type {import('@sveltejs/kit').Config} */
const config = {
  kit: {
    adapter: adapter({
      fallback: '404.html'
    }),
    paths: {
      base: process.argv.includes('dev') ? '' : process.env.BASE_PATH
    }
  }
};
```

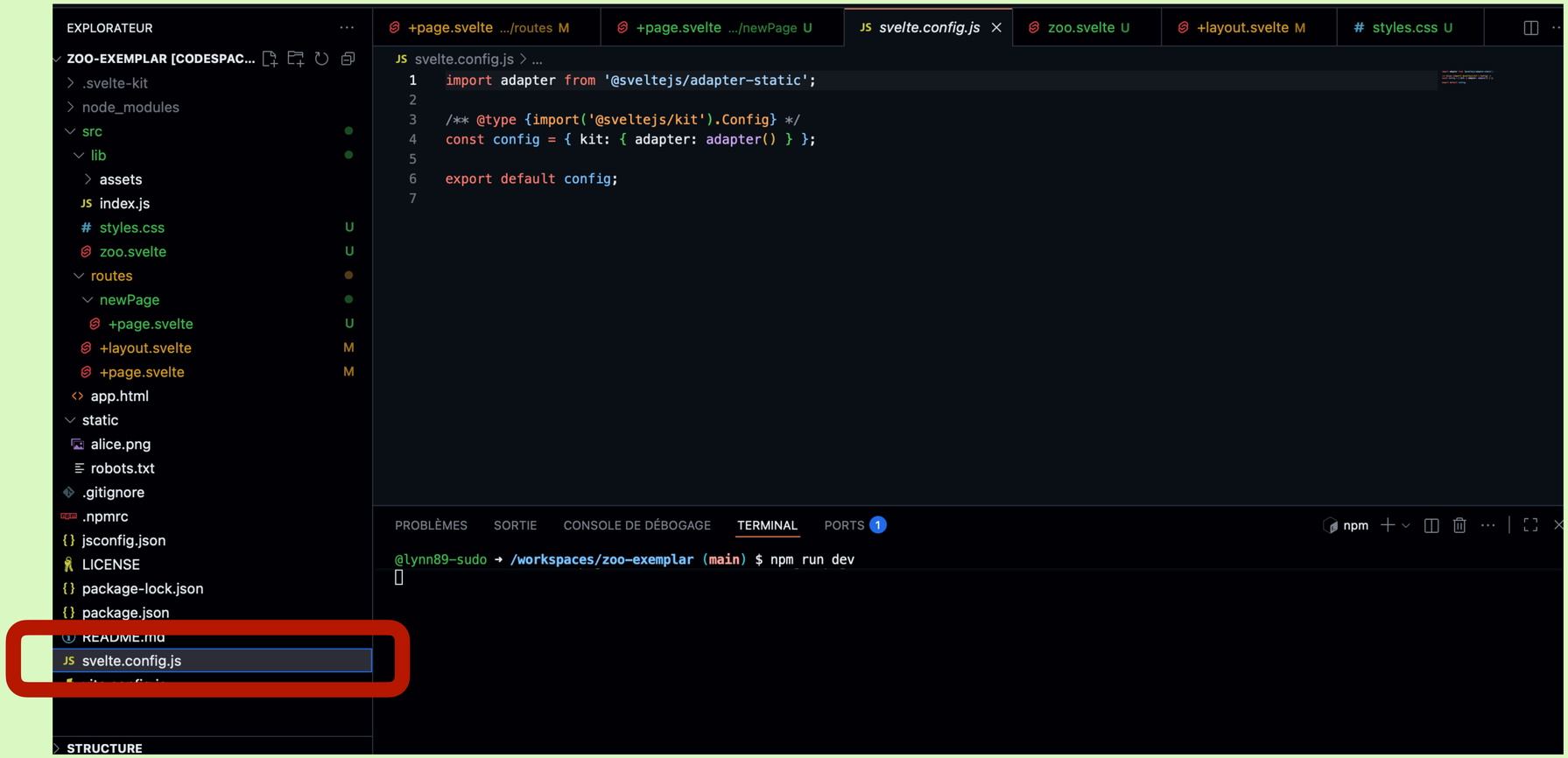
On this page

Static site generation
Usage
Zero-config support
Options
GitHub Pages

5

STATIC ADAPTER

Go back to your code and open up svelte.config.js (in the root of the folder)



5

STATIC ADAPTER

Replace everything in here with what you copied over. Then, replace base with this: “/“ + repo-name

Repo Name is whatever you named your repository, case-sensitive.

Put that in single quotations

```
import adapter from '@sveltejs/adapter-static';

/** @type {import('@sveltejs/kit').Config} */
const config = {
    kit: {
        adapter: adapter({
            fallback: '404.html'
        }),
        paths: {
            base: process.argv.includes('dev') ? '' : process.env.BASE_PATH
        }
    }
};

export default config;
```

```
import adapter from '@sveltejs/adapter-static';

/** @type {import('@sveltejs/kit').Config} */
const config = {
    kit: {
        adapter: adapter({
            fallback: '404.html'
        }),
        paths: {
            base: '/zoo-exemplar'
        }
    }
};

export default config;
```

5

.NOJEKYLL

In your static folder, create a blank document called ".nojekyll"

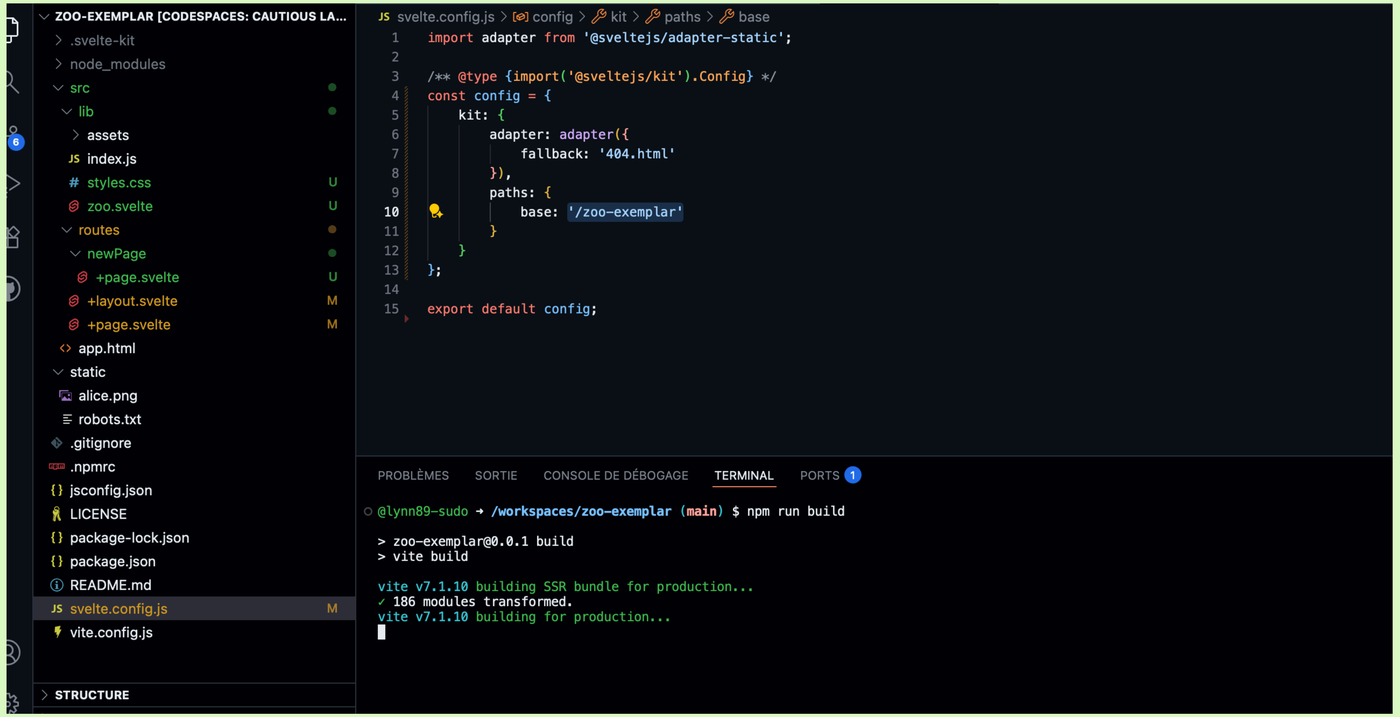
Don't put anything inside the document

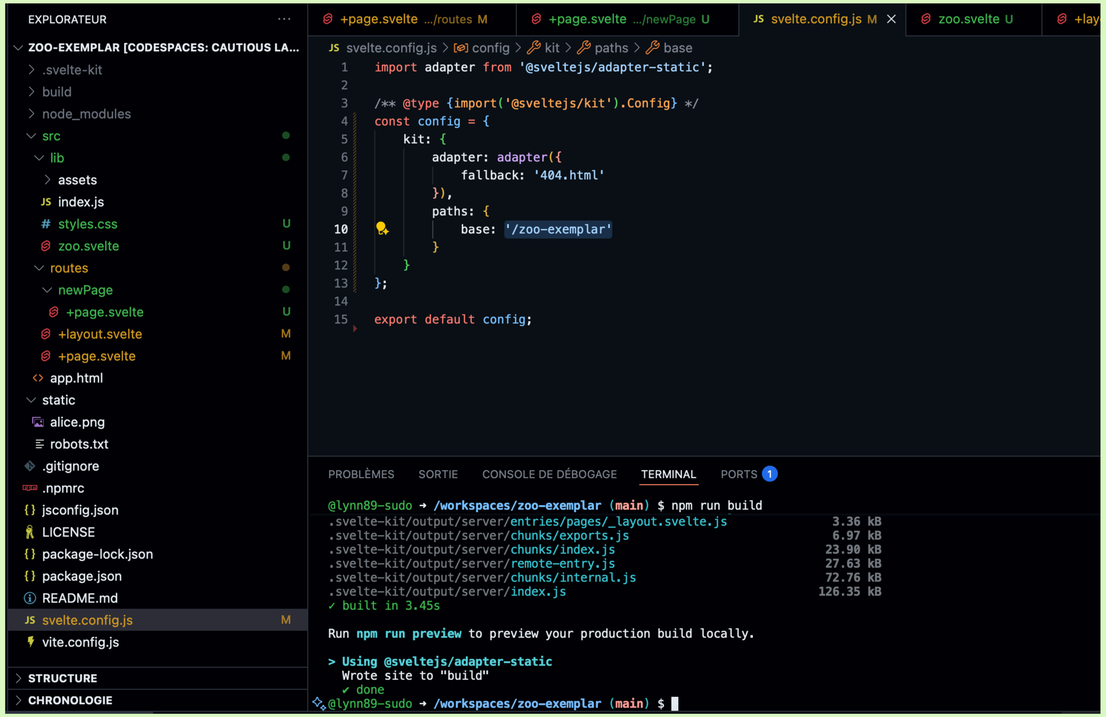
5

BUILDING WITH NPM

In the terminal, run “npm run build” and let it load

Once finished, it should create a new folder called “build”



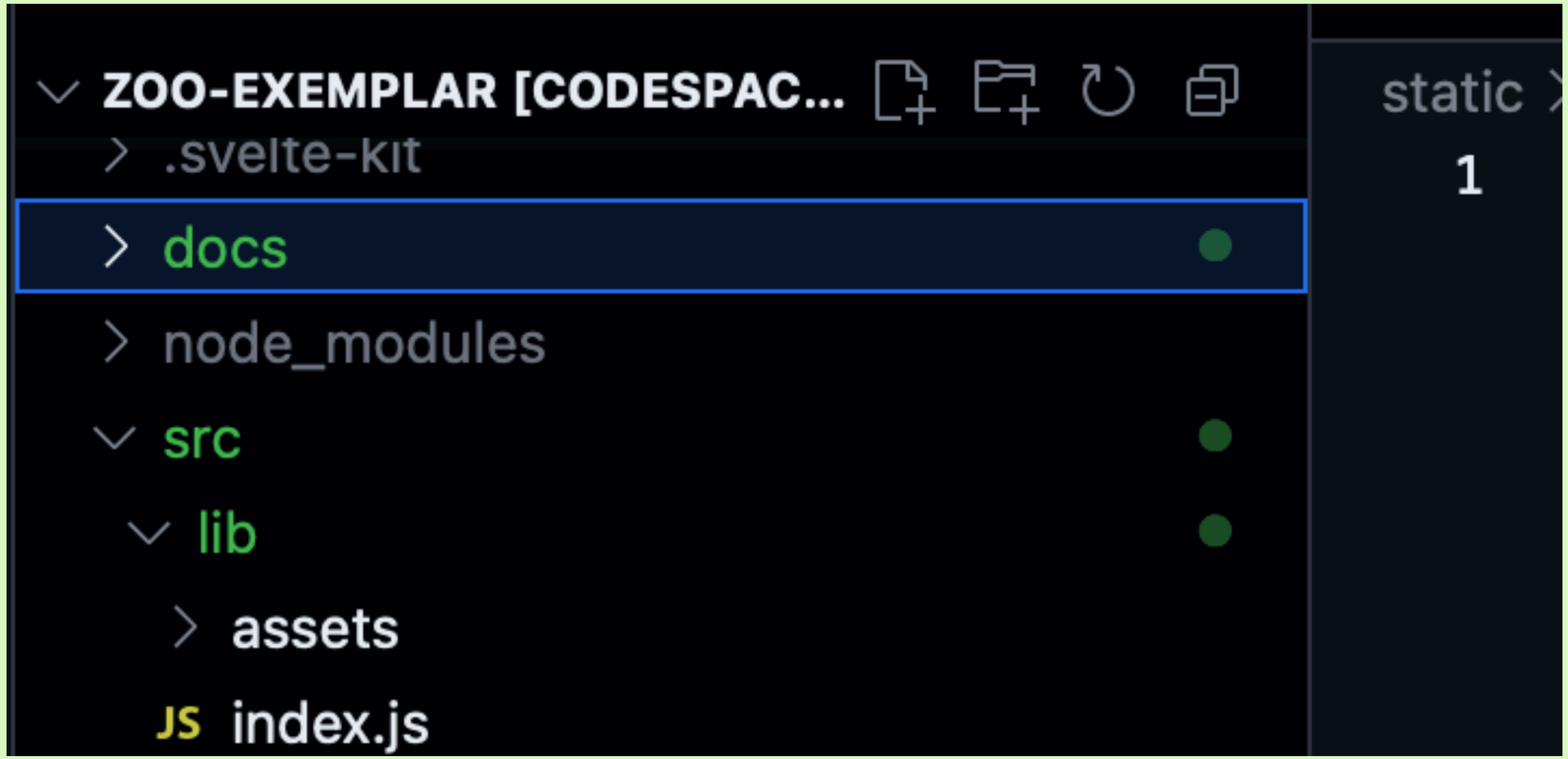


5

BUILDING WITH NPM

Rename the “build” folder to “docs”

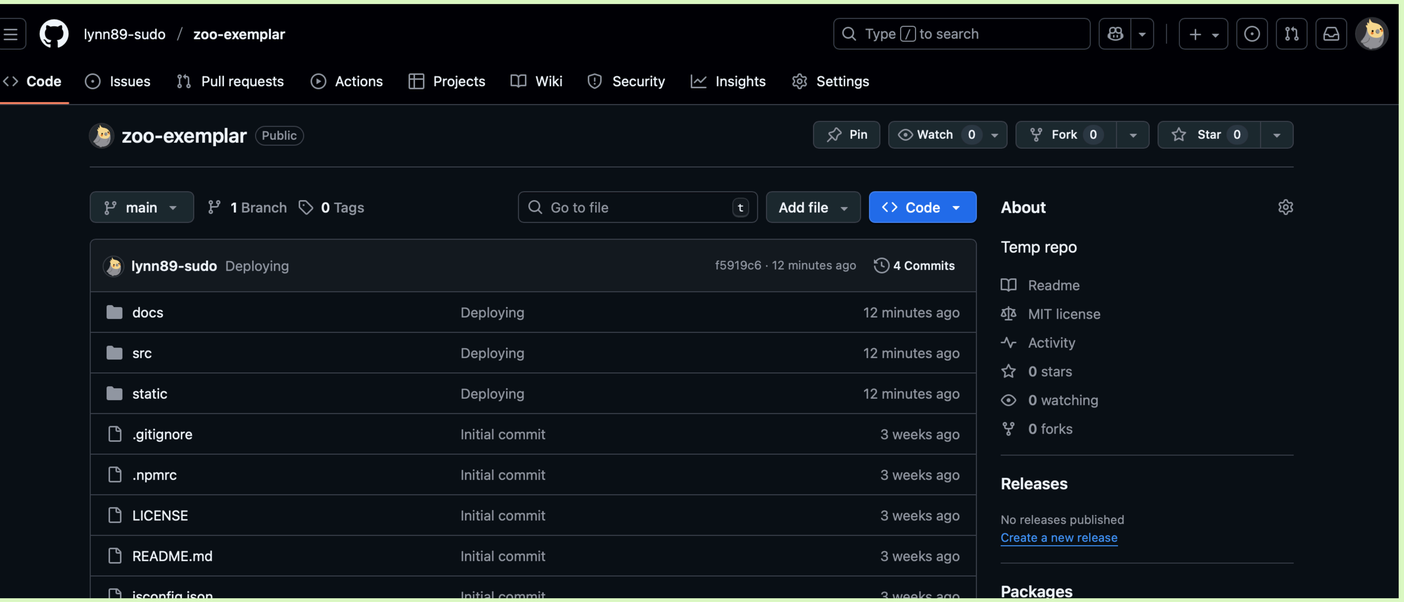
Once you do this, commit and push your changes!

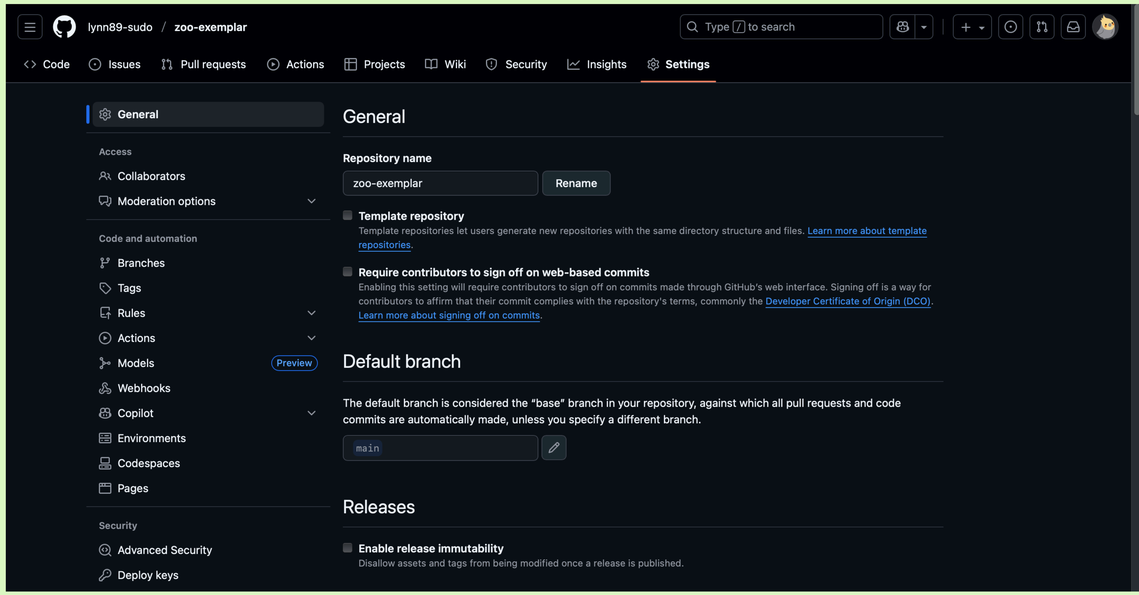


5

DEPLOYING ON GITHUB

Now, head over to your repo's settings on github.com





5

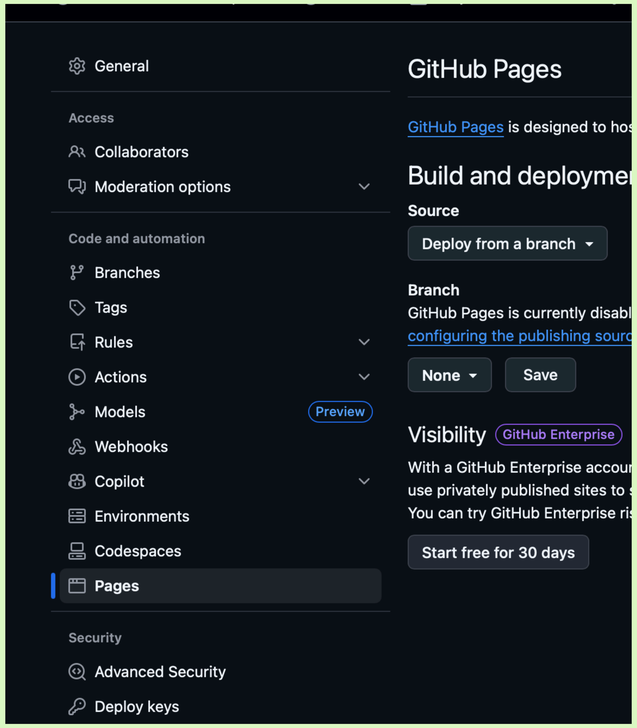
DEPLOYING ON GITHUB

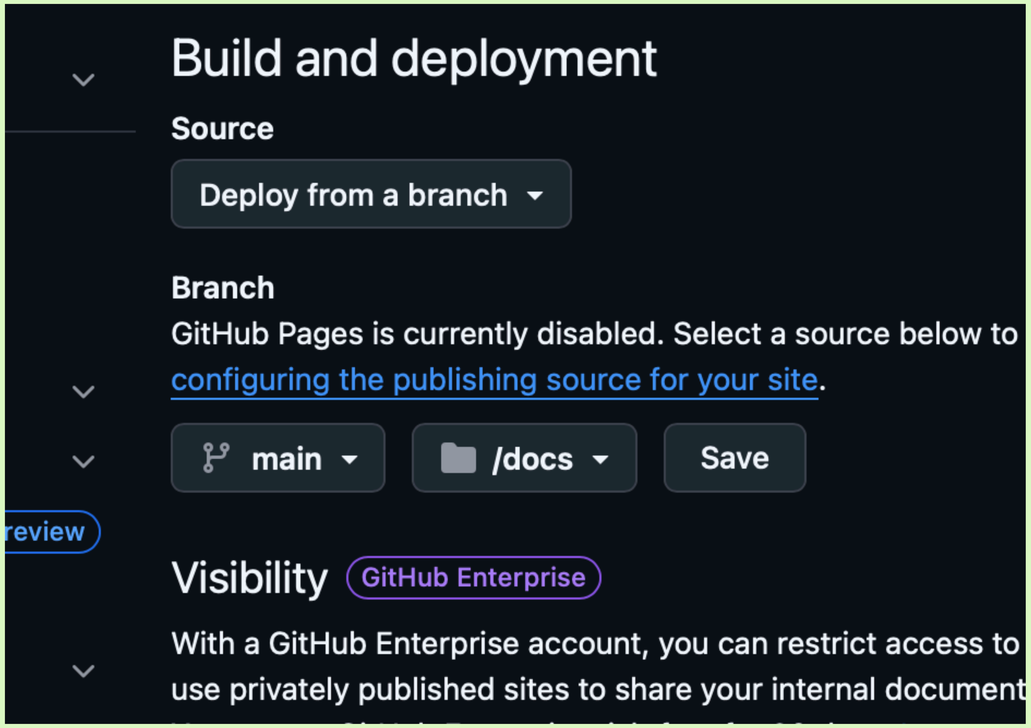
Go to the pages section

Select main branch

Choose docs, not root

Save

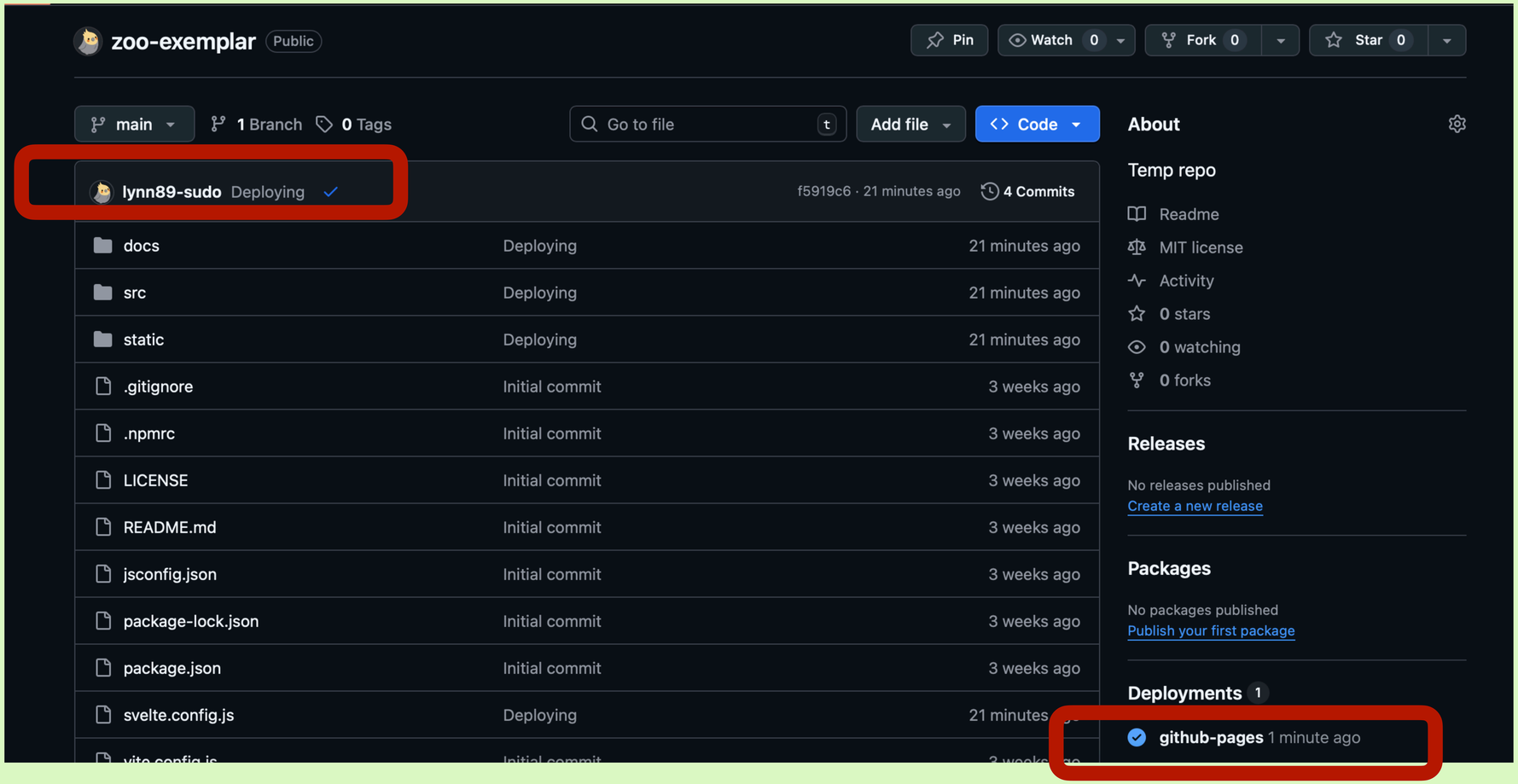




5

DEPLOYING ON GITHUB

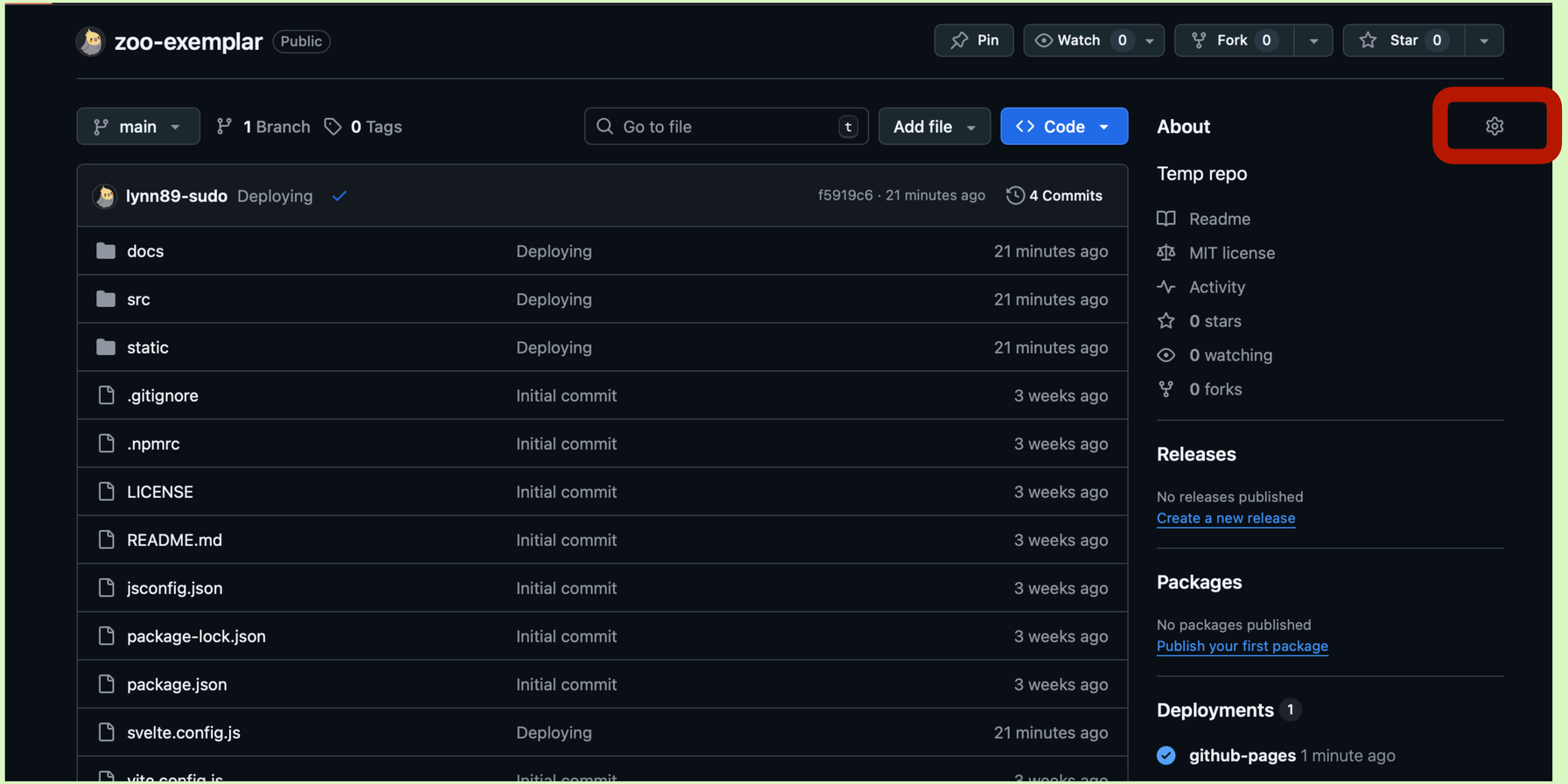
Go back to the main view. If you see the check mark, it deployed!



5

DEPLOYING ON GITHUB

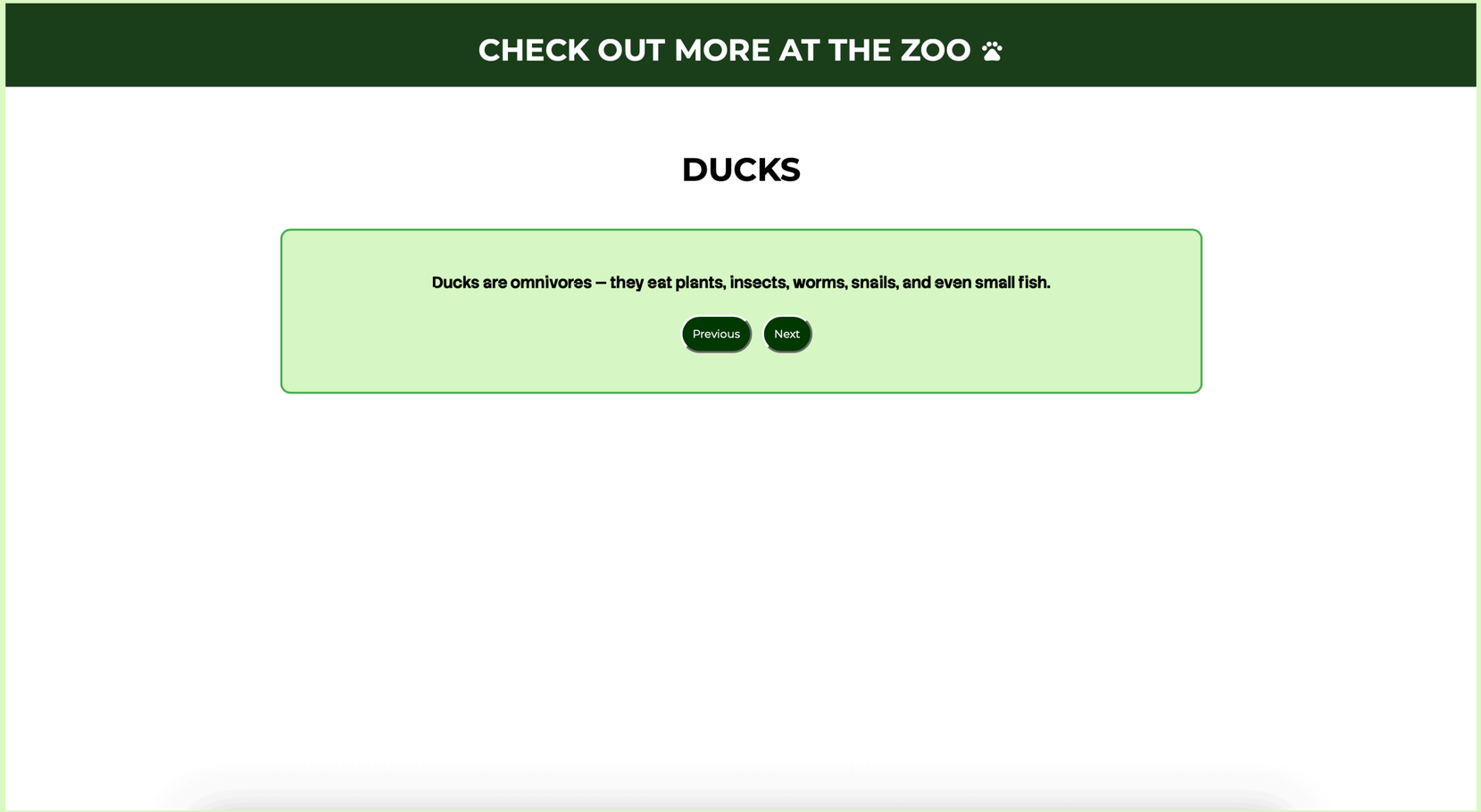
Click on the settings icon (different from before) and select “Use your GitHub Pages Website. Then save



5

ENJOY THE FRUITS OF YOUR LABOR (AND MAYBE EVEN SOME ACTUAL FRUIT?)

Click on the link to see your website live!



HACK CLUB ZOO

YOU'RE DONE! CONGRATULATIONS!

Submit at submit.hackclub.com/zoo