



Nuxt Fundamentals Navigation

Peter Kassenaar —
info@kassenaar.com



Creating navigation

Global Navigation Bars and/or Main Menu for your site

Creating global navigation



- Drop global navigation in your `index.vue` or a [layout file](#)
- Best practices:
 - Create a [separate component](#) for navigation, to keep files clean and simple
 - Name a global component something like `TheNavigation.vue`, `MainNavigation.vue` or `MainSidebar.vue`
- Add your global navigation to `./pages/index.vue`, or `layouts/default.vue`
 - For instance like:

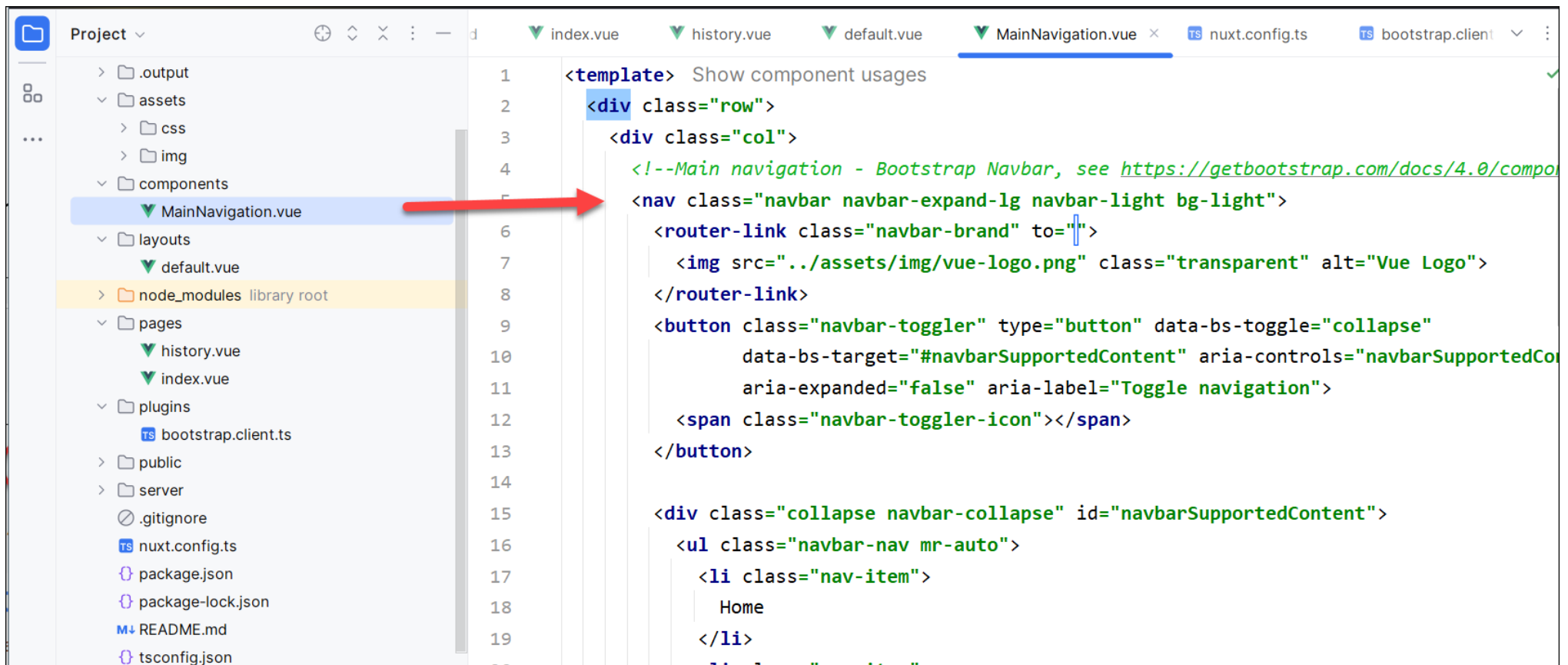
```
<template>
  <div class="container">
    <MainNavigation/> ←
    <slot/>
  </div>
</template>

<script setup lang="ts">
import MainNavigation from "@components/MainNavigation.vue";
</script>
```

MainNavigation.vue



- Contains – in this example – a default Bootstrap Navigation Bar

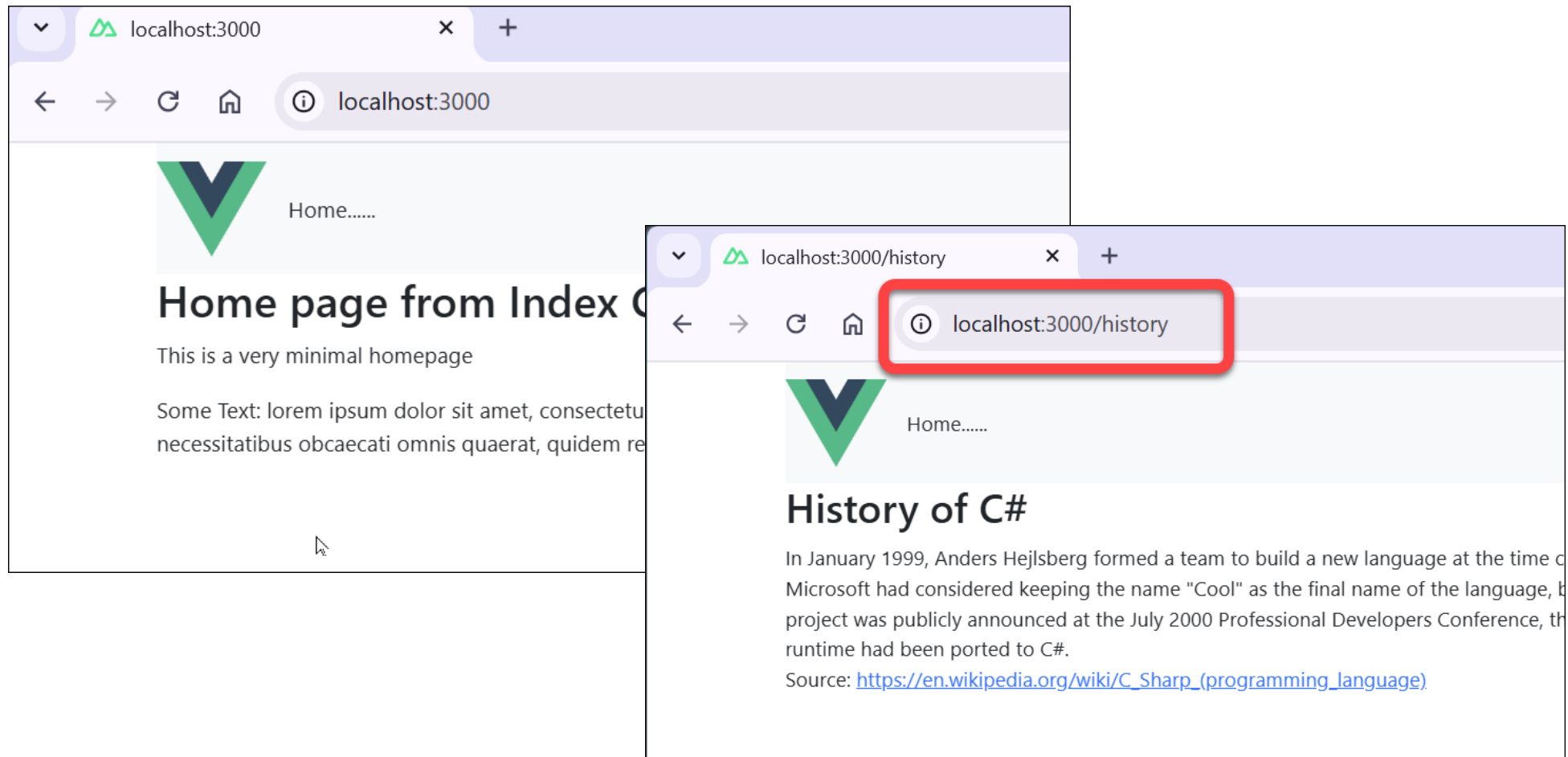


```
1 <template> Show component usages
2 <div class="row">
3   <div class="col">
4     <!--Main navigation - Bootstrap Navbar, see https://getbootstrap.com/docs/4.0/comp
5     <nav class="navbar navbar-expand-lg navbar-light bg-light">
6       <router-link class="navbar-brand" to="/">
7         
8       </router-link>
9       <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
10         data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedCo
11         aria-expanded="false" aria-label="Toggle navigation">
12         <span class="navbar-toggler-icon"></span>
13       </button>
14
15       <div class="collapse navbar-collapse" id="navbarSupportedContent">
16         <ul class="navbar-nav mr-auto">
17           <li class="nav-item">
18             Home
19           </li>
```

Browser



- This works, but no global navigation yet (= no links)
- Links are activated/filled in next...





Linking between pages

Using the `<nuxt-link>` element



Linking between pages

- In nuxt, you create links between pages using the `<nuxt-link>` element
 - They will be **compiled** to `<a href>` links.
- It looks like the standard `<router-link>` of Vue.
- If you use regular `` tags yourself, the page will refresh!
 - It **reloads the complete application**, not what you want!

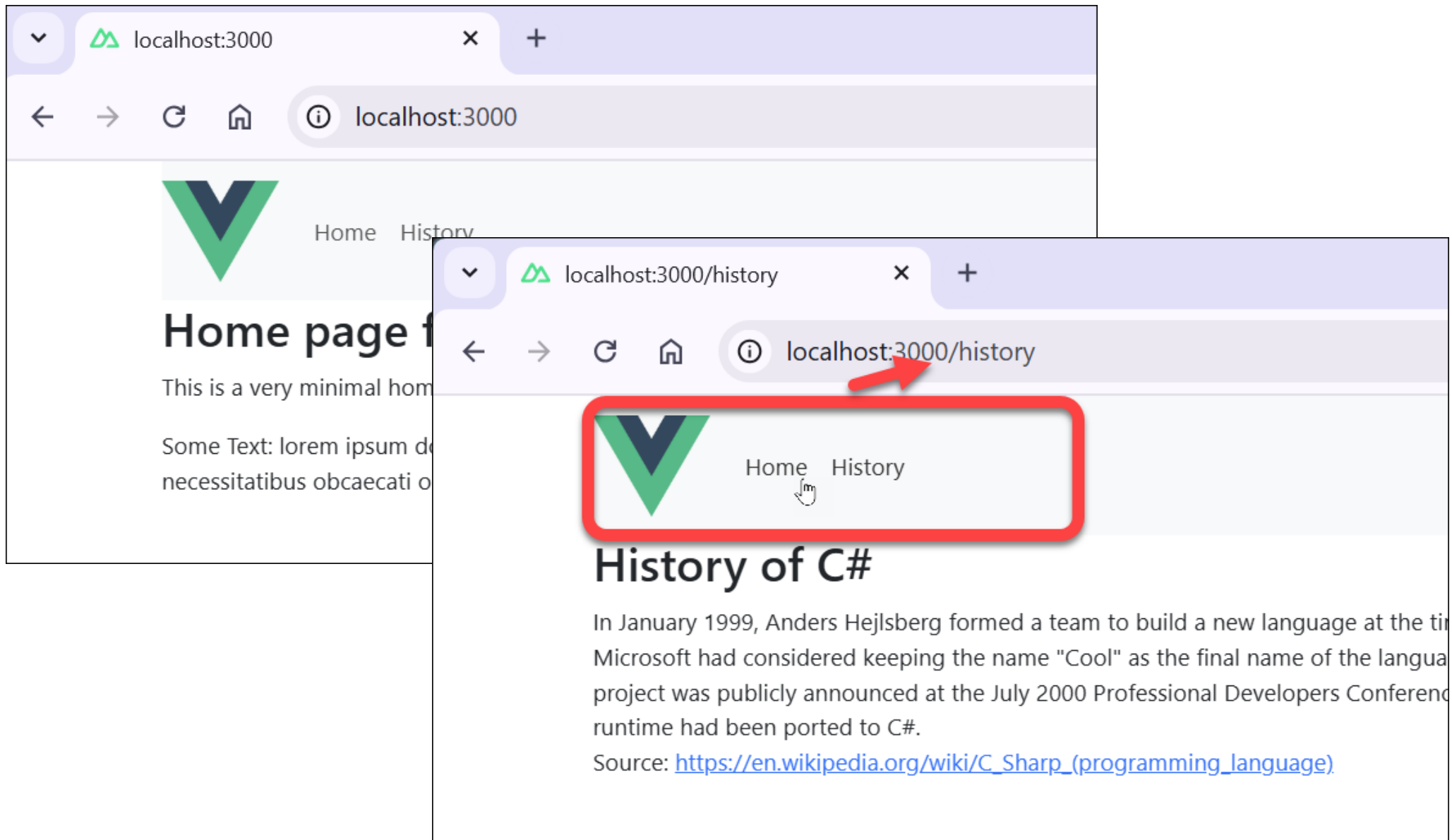
Using <nuxt-link>



```
<ul class="navbar-nav mr-auto">
  <li class="nav-item">
    <NuxtLink class="nav-link" to="/">Home</NuxtLink>
  </li>
  <li class="nav-item">
    <nuxt-link class="nav-link" to="/history">History</nuxt-link>
  </li>
</ul>
```

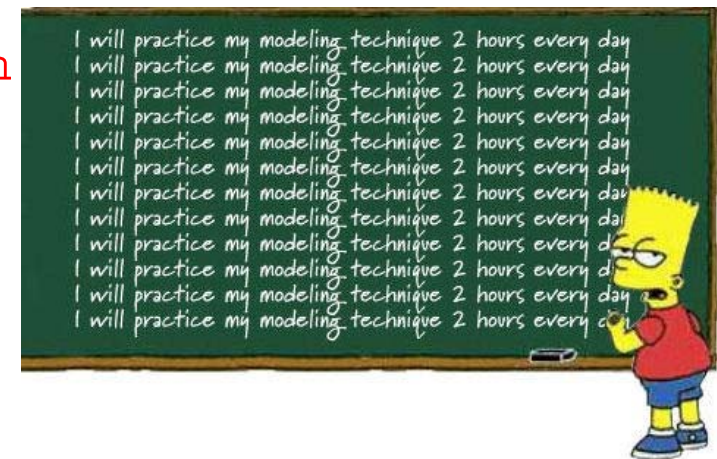
We can use both <NuxtLink> and <nuxt-link>
syntax (as with all Vue components)

Browser



Workshop

- Create a **Main Navigation** system for your app
- It can be a main menu header, sidebar or whatever.
- Be sure to create a component of it, and use it in `layouts/default.vue` or in your `index.vue` page
- Add links to the main navigation of your app
 - Use `<NuxtLink>` component for that – no need to import
- Test in the browser if they work
- Example: [../examples/120-basic-navigation](https://nuxt.com/docs/examples/120-basic-navigation)





Dynamic routing and route parameters

Sending parameters in your route

Dynamic Routing



You create a dynamic route by using the filename in square brackets

- First, create a component `<PostList/>`, containing an overview of some dummy posts
- <https://jsonplaceholder.typicode.com/posts>
- For details, create for instance a `./posts/[id].vue` file

Dynamic Routing, . /posts/index.vue



```
<!-- Posts list -->
<ul class="list-group">
  <li class="list-group-item"
    v-for="post in posts" :key="post.id">
    <nuxt-link :to="`/posts/${post.id}`" class="nav-link">
      {{ post.id }} - {{ post.title }}
    </nuxt-link>
  </li>
</ul>
```

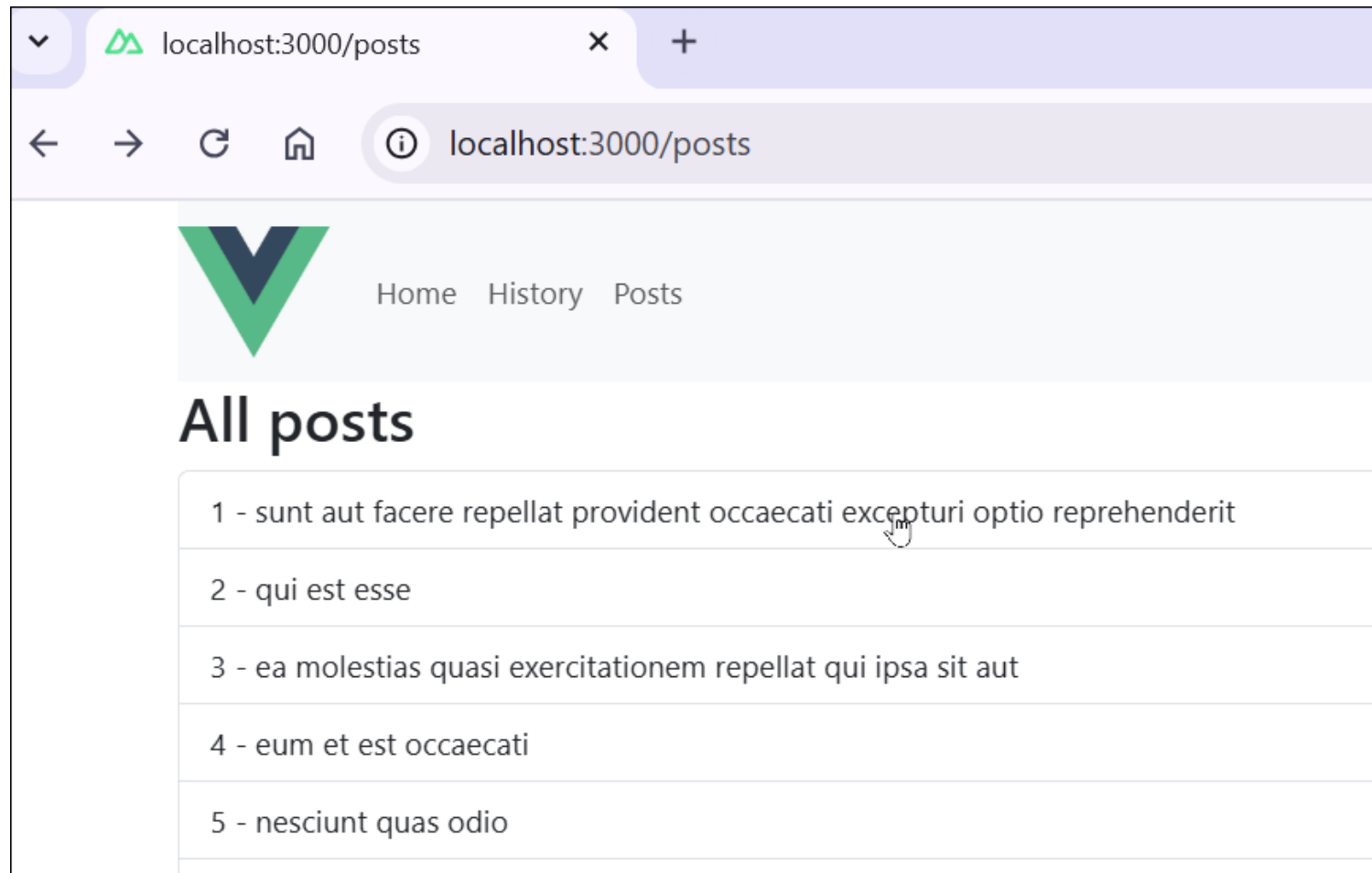
```
// URL to fetch data from.
const url = 'https://jsonplaceholder.typicode.com/posts';

// Reactive variables
const posts = ref<Post[]>([]); // store the posts

// Fetch posts when the component mounts
const fetchPosts = async () => {
  const response = await fetch(url);
  if (!response.ok) {
    throw new Error(`Error : ${response.statusText}`);
  }
  posts.value = await response.json();
};
```

Fetching dummy data from a
backend, using `fetch()`

Dynamic Routing



Dynamic Routing, details in [id].vue



Now, create a directory with a dynamic file

```
<template>
  <h2>Post number : {{ id }}</h2>
  <ul v-if="!isLoading && post" class="list-group">
    <li class="list-group-item">
      {{ post.id }}
    </li>
    ...
  </ul>
</template>
```



```
<script setup lang="ts">
import type {Post} from "~/types/Post";

const route = useRoute()
const id = route.params.id

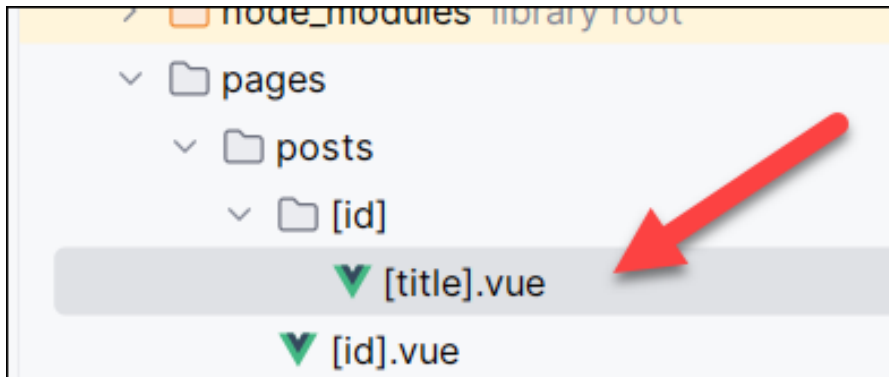
// URL to fetch data from.
const postDetail = `https://jsonplaceholder.typicode.com/posts/${id}`;

// Reactive variables
const post = ref<Post | null>(null); // store the post

// Fetch posts when the component mounts
const fetchPost = async () => {
  try {
    const response = await fetch(postDetail);
    post.value = await response.json();
  }
  ...
};
onMounted(fetchPost);
</script>
```

Multiple parameters?

- Sometimes you want to send **multiple parameters** in a URL
 - Like `http:.../posts/1/title`, where `1` and `title` are the dynamic parameters.
- **Solution**: create a directory structure that matches the parameters.
 - Your route can be like: `/posts/:id/:title`
 - Then your folder structure must be like:



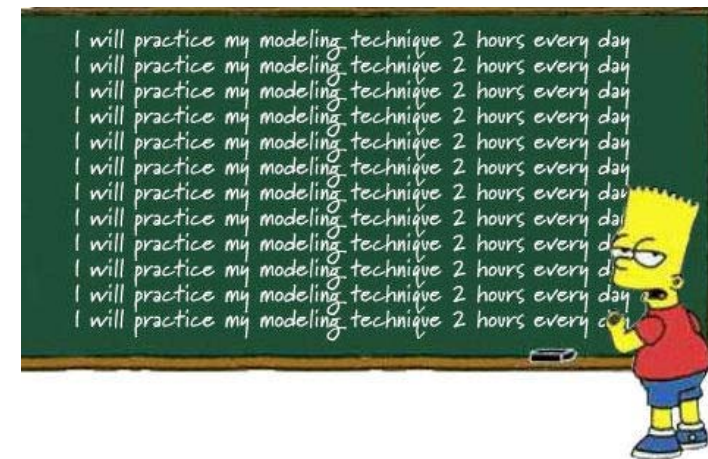
- Also: make sure to correctly **encode/decode** possible special characters in the url:

```
<nuxt-link :to="`/posts/${post.id}/${encodeURIComponent(post.title)}" ...>
```

```
const postTitle = decodeURIComponent(route.params.title);
```


Workshop

- Add dynamic links to the Nuxt application
- You can use the address jsonplaceholder.typicode.com/posts for some random data (or `/users` for dummy user data)
- Or maybe you have your own RESTful API.
- Test in the browser if they work
- Example [../130-navigation-parameters](#)



Checkpoint

- You know that the router is **automatically bundled** and configured once you add a `./pages` directory in your Nuxt-app.
- All pages in the directory become a **route in your application**
- Subdirectories like `./pages/myDir` are subfolders on your route.
- You can create **dynamic routes** using `[param].vue` as the filename.
- Create a directory structure if you need **multiple params**.