Routing

One core feature of Nuxt is the file system router. Every Vue file inside the `pages/` directory creates a corresponding URL (or route) that displays the contents of the file. By using dynamic imports for each page, Nuxt leverages code-splitting to ship the minimum amount of JavaScript for the requested route.

Pages

Nuxt routing is based on vue-router and generates the routes from every component created in the `pages/` directory, based on their filename.

This file system routing uses naming conventions to create dynamic and nested routes:

pages/ directory   Generated Router file

pages/ directory

```
pages/
--| about.vue
--| index.vue
--| posts/
----| [id].vue
```

👉 Read more in Docs > Guide > Directory Structure > Pages.

Navigation

The `<NuxtLink>` component links pages between them. It renders an `<a>` tag with the `href` attribute set to the route of the page. Once the application is hydrated, page transitions are performed in JavaScript by updating the browser URL. This prevents full-page refreshes and allows for animated transitions.

When a `<NuxtLink>` enters the viewport on the client side, Nuxt will automatically prefetch components and payload (generated pages) of the linked pages ahead of time, resulting in faster navigation.

pages/app.vue

```
<template>
  <header>
    <nav>
      <ul>
        <li><NuxtLink to="/about">About</NuxtLink></li>
        <li><NuxtLink to="/posts/1">Post 1</NuxtLink></li>
        <li><NuxtLink to="/posts/2">Post 2</NuxtLink></li>
      </ul>
    </nav>
  </header>
</template>
```

👉 Read more in Docs > API > Components > Nuxt Link.

Route Parameters

The `useRoute()` composable can be used in a `<script setup>` block or a `setup()` method of a Vue component to access the current route details.

pages/posts/[id].vue

```
<script setup lang="ts">
const route = useRoute()

// When accessing /posts/1, route.params.id will be 1
console.log(route.params.id)
</script>
```

👉 Read more in Docs > API > Composables > Use Route.

Route Middleware

Nuxt provides a customizable route middleware framework you can use throughout your application, ideal for extracting code that you want to run before navigating to a particular route.

Route middleware runs within the Vue part of your Nuxt app. Despite the similar name, they are completely different from server middleware, which are run in the Nitro server part of your app.

There are three kinds of route middleware:

1. Anonymous (or inline) route middleware, which are defined directly in the pages where they are used.
2. Named route middleware, which are placed in the `middleware/` directory and will be automatically loaded via asynchronous import when used on a page. (**Note**: The route middleware name is normalized to kebab-case, so `someMiddleware` becomes `some-middleware`.)
3. Global route middleware, which are placed in the `middleware/` directory (with a `.global` suffix) and will be automatically run on every route change.

Example of an `auth` middleware protecting the `/dashboard` page:

middleware/auth.ts pages/dashboard.vue

middleware/auth.ts

```
export default defineNuxtRouteMiddleware((to, from) => {
  // isAuthenticated() is an example method verifying if a user is authenticated
  if (isAuthenticated() === false) {
    return navigateTo('/login')
  }
})
```

👉 Read more in Docs > Guide > Directory Structure > Middleware.

Route Validation

Nuxt offers route validation via the `validate` property in `definePageMeta` in each page you wish to validate.

The `validate` property accepts the `route` as an argument. You can return a boolean value to determine whether or not this is a valid route to be rendered with this page. If you return `false`, and another match can't be found, this will cause a 404 error. You can also directly return an object with `statusCode` / `statusMessage` to respond immediately with an error (other matches will not be checked).

If you have a more complex use case, then you can use anonymous route middleware instead.

pages/posts/[id].vue

```
<script setup lang="ts">
definePageMeta({
  validate: async (route) => {
    // Check if the id is made up of digits
    return /^\d+$/.test(route.params.id)
  }
})
</script>
```

Edit on Github

Enterprise Design Kit NuxtLabs

Nuxt Studio