2

NUXT ET LES AGENTS · CHAPITRE 2

Du routeur à Nuxt

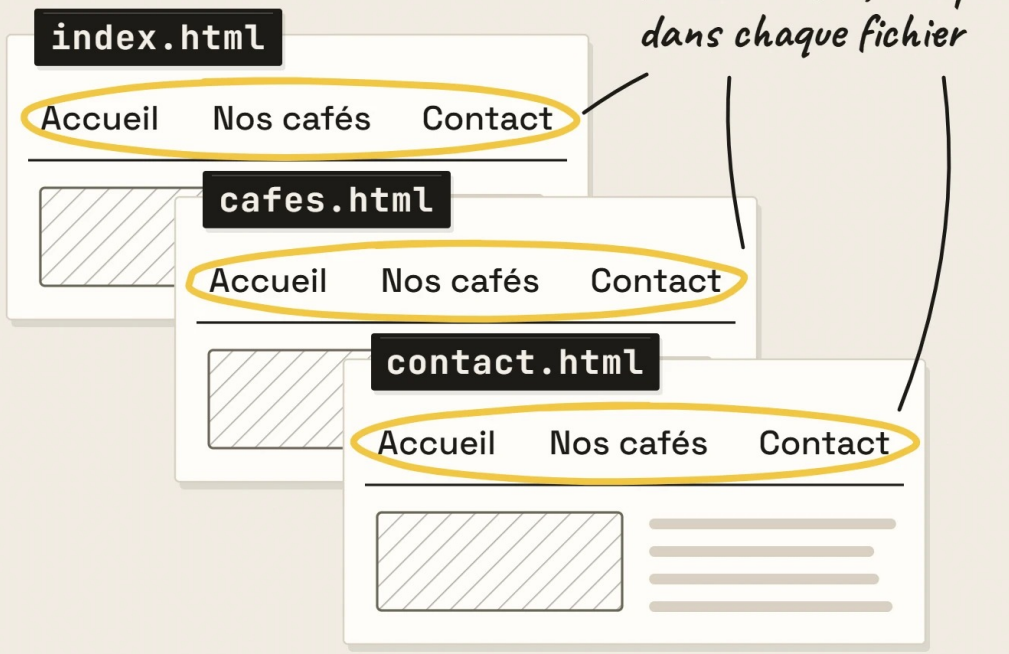
Thibault Six · IFOSUP Wavre · 2026-2027



? Dans un projet Vue, combien y a-t-il de fichiers HTML ?

Un site HTML a un fichier par page, une application Vue n'en a qu'un.

avant · le site en HTML



après · l'application Vue

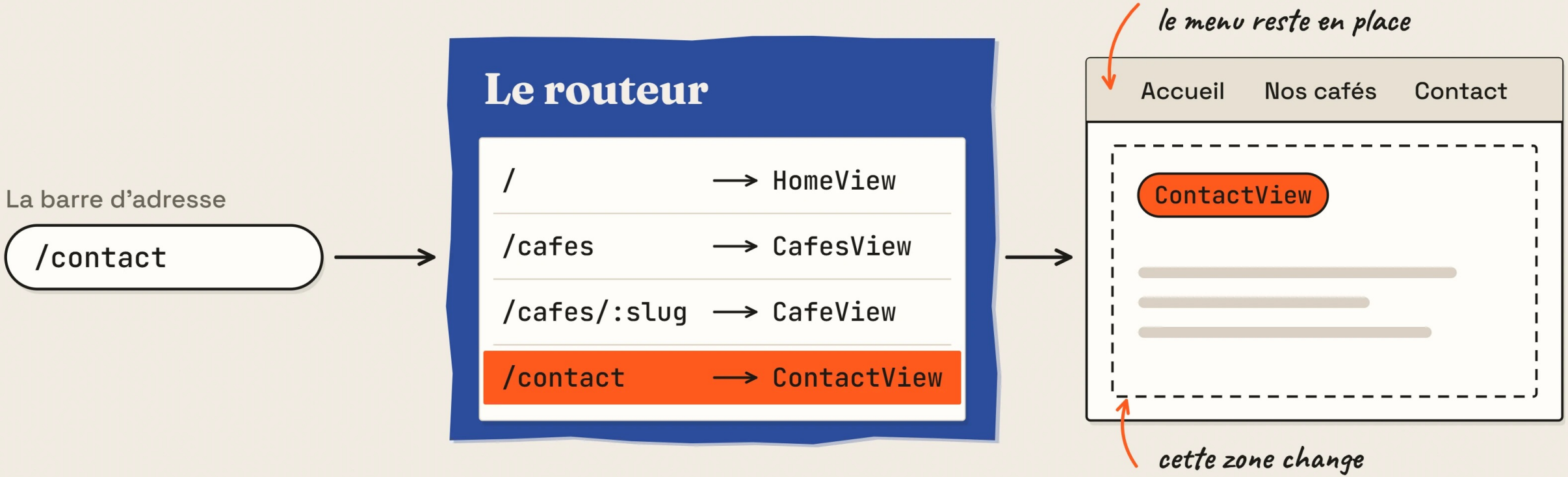
dist/ : le projet construit avec npm run build



Kawa, un torréfacteur de quartier à Wavre, nous sert d'exemple. Son application Vue n'a qu'un fichier, index.html, et le serveur renvoie le même pour toutes les adresses. C'est donc à l'application de choisir quoi afficher.

? Un seul fichier HTML pour toutes les adresses. Comment l'application sait-elle quelle page afficher ?

Le routeur lit l'adresse et affiche le composant qui lui correspond.



Le routeur fait ce tri avec sa table des routes. Chaque ligne, une route, relie une adresse à un composant. Au clic sur un lien, il change l'adresse et le composant, sans redemander de page au serveur.

? Avec Vue Router, où ranges-tu les pages de Kawa ?

Avec Vue Router, chaque page est un composant, et la table des routes doit l'importer.

```
src/
├── assets/
│   └── main.css
├── data/
│   └── cafes.json
├── router/
│   └── index.js
├── views/
│   ├── HomeView.vue
│   ├── CafesView.vue
│   ├── CafeView.vue
│   └── ContactView.vue
├── App.vue
└── main.js
```

terminal

```
npm install vue-router
```

src/router/index.js · 1/2

```
import { createRouter, createWebHistory } from 'vue-router'
import HomeView from '../views/HomeView.vue'
import CafesView from '../views/CafesView.vue'
import CafeView from '../views/CafeView.vue'
import ContactView from '../views/ContactView.vue'
```

import : la table va chercher chaque composant dans son fichier

Vue Router est le routeur officiel de Vue. C'est un paquet, du code prêt à l'emploi qu'on ajoute au projet avec la commande de la fiche. Ce code, tu ne l'écriras pas : on le regarde pour voir le travail à faire.

? Dans la suite du fichier, quelle ligne relie l'adresse /contact à sa page ?

Dans la table des routes, chaque ligne relie une adresse à un composant.

src/router/index.js · 2/2

```
const router = createRouter({
  history: createWebHistory(),
  routes: [
    { path: '/', component: HomeView },
    { path: '/cafes', component: CafesView },
    { path: '/cafes/:slug', component: CafeView },
    { path: '/contact', component: ContactView },
  ],
})

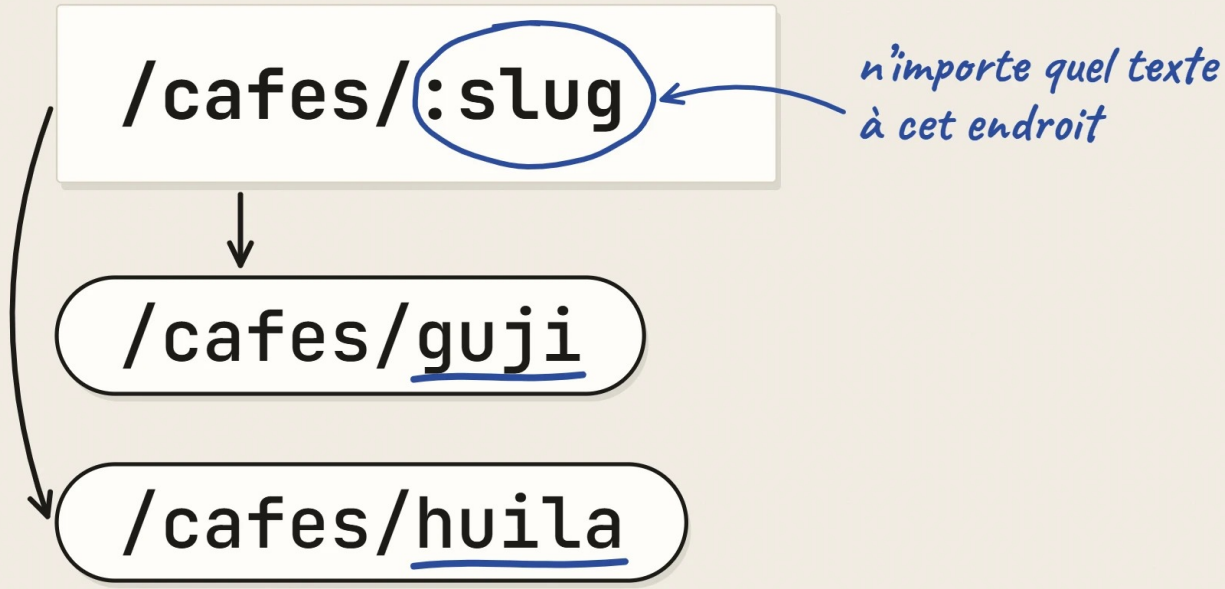
export default router
```



C'est la table de la slide 2, écrite à la main, avec une ligne par adresse. Le reste du fichier crée le routeur, tu n'as pas à le lire.

? Kawa vend quatre cafés. Faut-il quatre lignes dans la table, une par café ?

Une seule route, /cafes/:slug, sert pour tous les cafés de Kawa.



le slug, c'est le nom du café écrit pour une adresse

Guji ⟶ `guji`

en minuscules, sans accent, avec des tirets

route : l'adresse en cours

`route.params.slug vaut 'guji'`

params : les morceaux variables de l'adresse

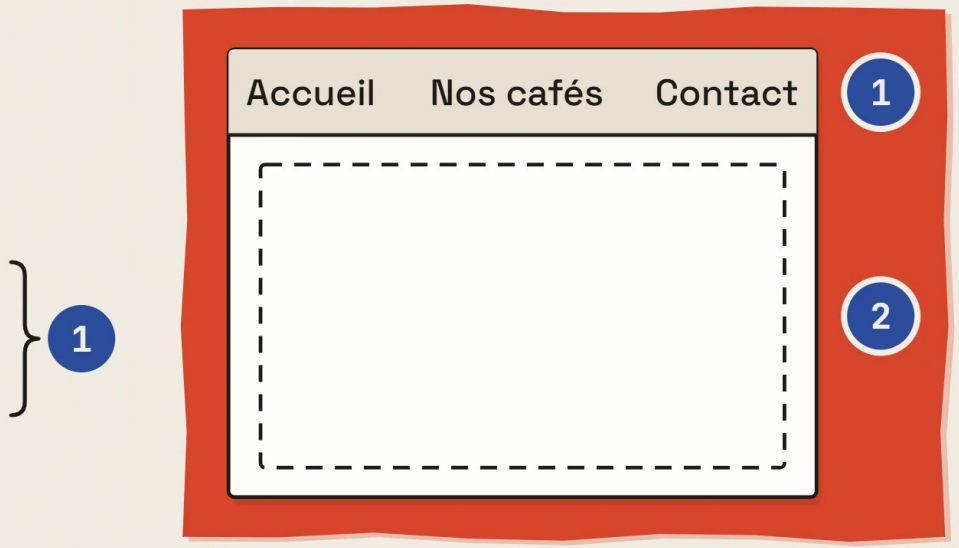
Ce que tu tapes à cet endroit de l'adresse, le composant CafeView le retrouve sous le nom slug. Puis il cherche le café qui porte ce slug dans le fichier cafes.json.

? En HTML, le menu de Kawa s'écrivait dans chaque fichier. Avec Vue Router, combien de fois ?

Dans src/App.vue, le menu s'écrit une fois, et RouterView marque où la page s'affiche.

src/App.vue

```
<template>
  <header>
    <nav>
      <RouterLink to="/">Accueil</RouterLink>
      <RouterLink to="/cafes">Nos cafés</RouterLink>
      <RouterLink to="/contact">Contact</RouterLink>
    </nav>
  </header>
  <main>
    <RouterView />
  </main>
</template>
```



1 RouterLink remplace la balise <a>, sans recharger la page

2 ici s'affiche le composant choisi

src/App.vue est le composant qui entoure toutes les pages. Au clic sur un lien du menu, le routeur change le composant de la zone, et le menu reste en place.

? Kawa demande une page « Les ateliers ». Avec Vue Router, que dois-tu toucher dans le projet ?

Pour la page « Les ateliers », tu cherches avec ton voisin chaque endroit à toucher.

```
src/
├── assets/
│   └── main.css
├── data/
│   └── cafes.json
├── router/
│   └── index.js
├── views/
│   ├── HomeView.vue
│   ├── CafesView.vue
│   ├── CafeView.vue
│   └── ContactView.vue
├── App.vue
└── main.js
```

Kawa demande une page « Les ateliers »

`/ateliers`

ta réponse : quel fichier, quoi y ajouter

?

Reprends les trois slides précédentes. Pour chaque endroit, dis le fichier et ce que tu y ajoutes.

? Qu'as-tu trouvé avec ton voisin, et que se passe-t-il si tu oublies une étape ?

Avec Vue Router, une page de plus demande quatre étapes, dans trois fichiers.

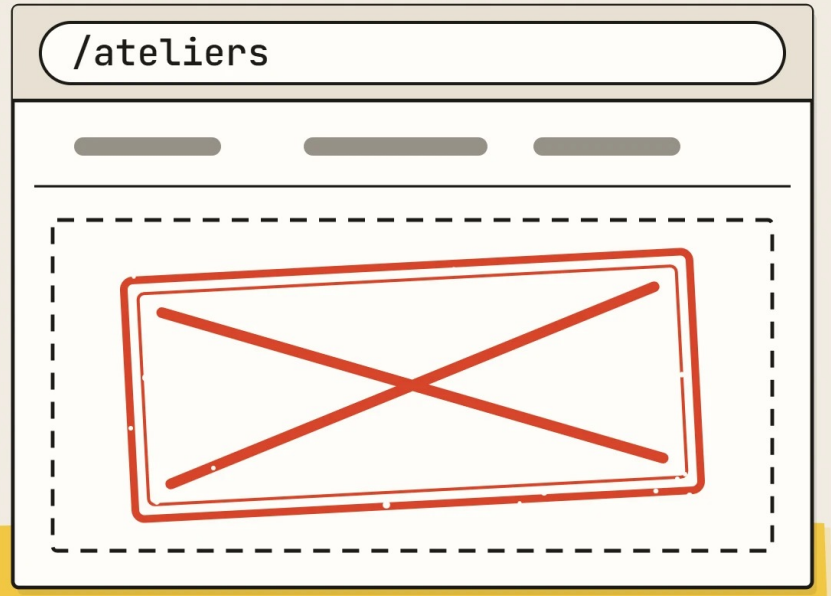
```
src/
├── assets/
│   └── main.css
├── data/
│   └── cafes.json
├── router/
│   └── index.js
├── views/
│   ├── HomeView.vue
│   ├── CafesView.vue
│   ├── CafeView.vue
│   ├── ContactView.vue
│   └── AteliersView.vue
├── App.vue
└── main.js
```

2 l'importer

3 ajouter sa ligne dans la table

1 **créer le composant**

4 ajouter son RouterLink dans le menu de src/App.vue



si tu importes la page sans ajouter sa ligne, /ateliers n'affiche que le menu

Ces quatre étapes reviennent à chaque nouvelle page. Si tu en oublies une, il manque quelque chose, comme ici.

? Voici le même site de Kawa, refait avec Nuxt. Quelle adresse donne chaque fichier du dossier pages ?

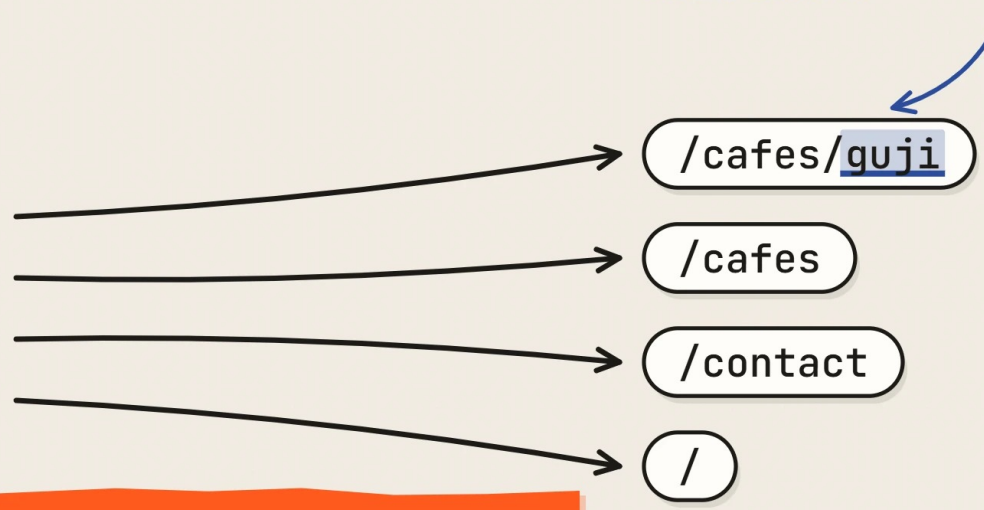
Dans Nuxt, chaque fichier du dossier pages est une page, et son chemin donne son adresse.

dans Nuxt, le code du site est dans le dossier app/

```
app/
└── pages/
    ├── cafes/
    │   ├── [slug].vue
    │   └── index.vue
    ├── contact.vue
    └── index.vue
```

index.vue donne l'adresse de son dossier

[slug] : n'importe quel texte à cet endroit



Créer un fichier dans pages, c'est créer une page.

```
route.params.slug vaut 'guji',
comme avec Vue Router
```

On refait le même site de Kawa avec Nuxt. Nuxt est un framework, c'est-à-dire des outils et des règles de rangement, construit sur Vue. Tu y écris les mêmes composants .vue.

? Dans Nuxt, où s'écrit le menu commun à toutes les pages de Kawa ?

Le layout entoure toutes les pages : il porte le menu et montre où la page s'affiche.

app/layouts/default.vue

default : le layout des pages qui n'en demandent pas un autre

```
<template>
  <header>
    <nav>
      <NuxtLink to="/">Accueil</NuxtLink>
      <NuxtLink to="/cafes">Nos cafés</NuxtLink>
      <NuxtLink to="/contact">Contact</NuxtLink>
    </nav>
  </header>
  <main>
    <slot />
  </main>
</template>
```

navigateur · /cafes/guji



1 NuxtLink joue le rôle de RouterLink

2 `<slot />` : la place laissée libre, où s'affiche la page, comme RouterView

Le menu vient du layout, et la fiche de Guji vient du fichier app/pages/cafes/[slug].vue. La version Vue Router du site affiche exactement la même page.

? Kawa demande la même page « Les ateliers ». Dans Nuxt, combien d'étapes, et dans combien de fichiers ?

Dans Nuxt, une page de plus demande deux étapes, dans deux fichiers.

Pour ajouter une page	Vue Router	Nuxt
Créer le composant de la page	`src/views/AteliersView.vue`	`app/pages/ateliers.vue`
L'importer	une ligne dans `src/router/index.js`	rien
Déclarer son adresse	une ligne dans `src/router/index.js`	rien : le nom du fichier suffit
Ajouter le lien au menu	une ligne dans `src/App.vue`	une ligne dans `app/layouts/default.vue`
Bilan	**1 fichier créé, 2 fichiers modifiés, 3 lignes ajoutées en plus du composant**	**1 fichier créé, 1 fichier modifié, 1 ligne ajoutée en plus du composant**

`navigateur · /ateliers`



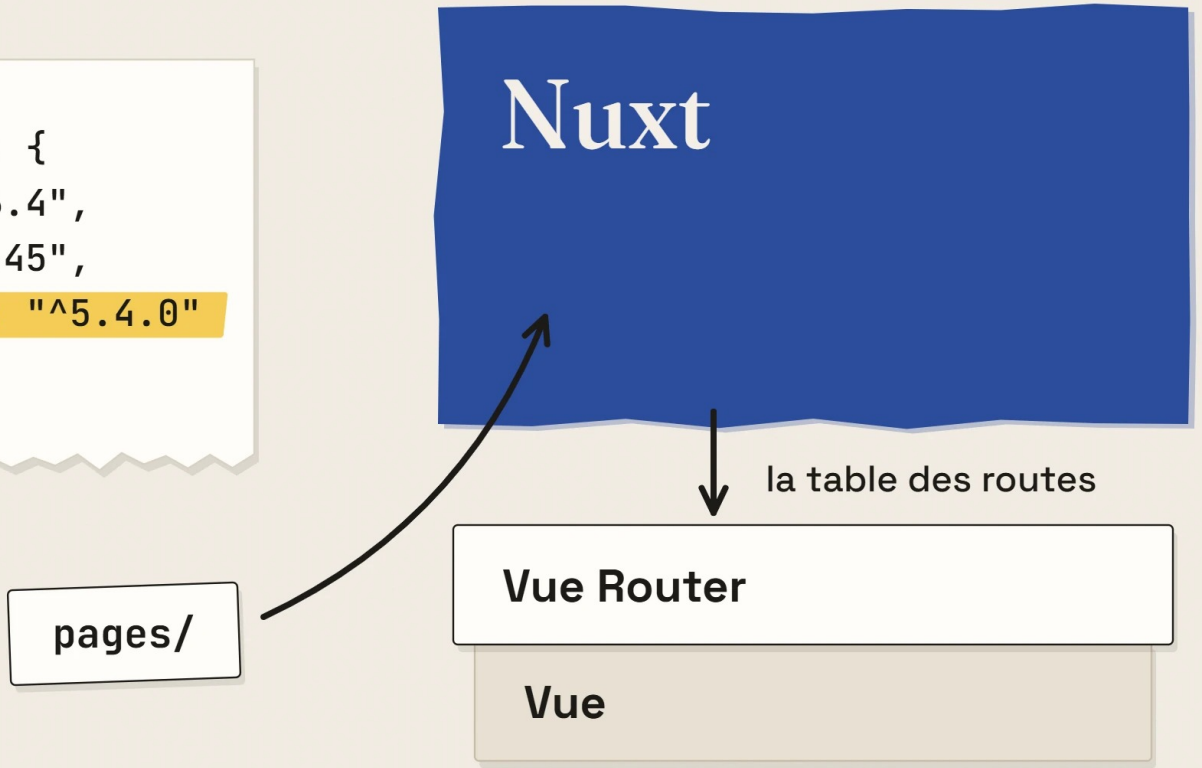
Dans Nuxt, l'adresse /ateliers répond dès que le fichier existe, même pendant que le site tourne avec npm run dev. Le lien du menu sert au visiteur, pas au routeur.

? Nuxt a-t-il remplacé Vue Router ?

Nuxt se sert toujours de Vue Router : il lit le dossier pages et écrit la table pour toi.

package.json

```
"dependencies": {
  "nuxt": "^4.6.4",
  "vue": "^3.5.45",
  "vue-router": "^5.4.0"
}
```



le routeur par fichiers

Nuxt ajoute aussi

- les images
- le contenu en Markdown
- le référencement
- un petit backend, du code sur le serveur

Les dépendances, ce sont les paquets dont le projet a besoin, et package.json les liste. Nuxt part de Vue et ajoute ce qu'il faut pour un site complet. On dit que c'est un meta-framework, un framework construit au-dessus d'un autre.

? Voici le dossier pages d'un fleuriste. Quelle adresse donne chaque fichier ?

À toi : écris l'adresse de chaque fichier du fleuriste, sans ordinateur, en cinq minutes.

le fleuriste

```
pages/
├── bouquets/
│   ├── [slug].vue      →  _____
│   └── index.vue       →  _____
├── infos/
│   ├── contact.vue     →  _____
│   └── livraison.vue   →  _____
├── equipe.vue          →  _____
├── index.vue           →  _____
└── mariages.vue        →  _____
```

1. quelle adresse affiche la fiche du bouquet « Pivoines blanches » ?
2. que vaut alors `route.params.slug` ?
3. que donne l'adresse `/infos` ?

indice : regarde ce que contient le dossier infos

Écris tes réponses sur papier, puis compare-les avec ton voisin.

? Quelles adresses as-tu trouvées, et que donne /infos ?

Chaque dossier a besoin de son index.vue, sinon son adresse répond par une erreur 404.

le fleuriste

```
pages/
├── bouquets/
│   ├── [slug].vue   ──→  /bouquets/…, une adresse par bouquet
│   └── index.vue    ──→  /bouquets
├── infos/           ┆    ici manque index.vue
│   ├── contact.vue  ──→  /infos/contact
│   └── livraison.vue ──→  /infos/livraison
├── equipe.vue       ──→  /equipe
├── index.vue        ──→  /
└── mariages.vue     ──→  /mariages
```

1. `/bouquets/pivoines-blanches`
2. `route.params.slug` vaut `'pivoines-blanches'`
3. `/infos` : erreur 404, page introuvable

1. **un `index.vue` par liste**
2. **un `[slug].vue` par fiche**
3. **des noms sans accent ni espace**

une page simple : un seul fichier, comme equipe.vue

Le dossier pages du site de ton client marche de la même façon. Pendant l'atelier, vérifie avec ces trois règles les pages que ton agent crée.