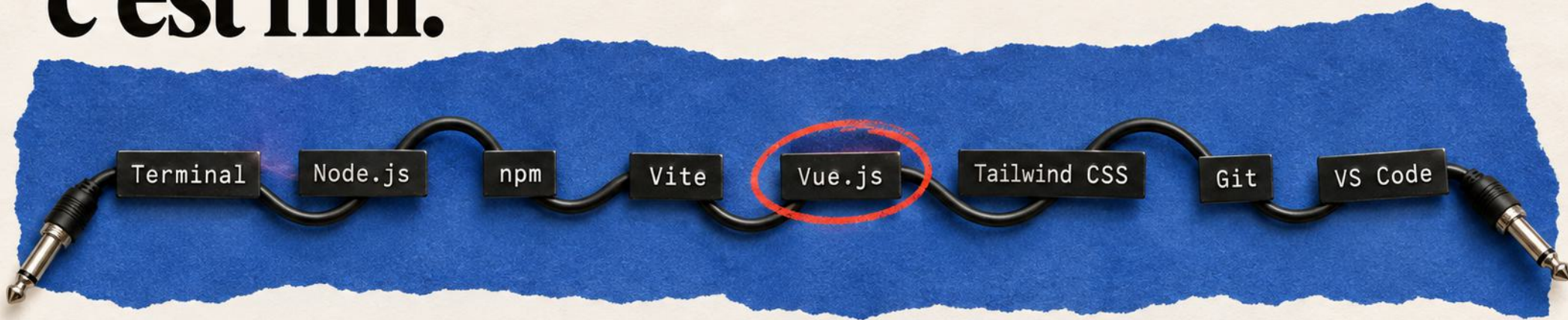


LET'S
BUILD

Ouvrez le terminal. On code votre portfolio.

Vue.js, Tailwind, et la boîte à outils
d'un studio web en 2026.

Un fichier `index.html` ouvert dans le navigateur : c'est fini.



En agence, en freelance, dans n'importe quelle équipe produit, le travail passe par une **toolchain** : une chaîne d'outils qui s'enchaînent.

Tu dois penser à mettre à jour l'affichage toi-même.

Sur une page avec trente éléments liés, tu vas en oublier un. C'est le bug numéro un des sites codés à la main.

```
<script>
  let count = 0;
  const btn = document.getElementById('btn');
  const countSpan = document.getElementById('count');

  btn.addEventListener('click', () => {
    count++;
    countSpan.textContent = count;
  });
</script>
```



Tu décris ce que la page doit afficher.

App.vue

```
<script setup>
  import { ref } from 'vue'

  const count = ref(0)
</script>
```

```
<template>
  <button @click="count++">
    Compteur : {{ count }}
  </button>
</template>
```

Aucun getElementById nulle part. Vue s'occupe du DOM.

SFC

Réactivité et composants réutilisables.

1. La réactivité.

Tu declares une donnée, tu l'affiches, et quand la donnée change, l'affichage suit tout seul.

2. Les composants réutilisables.

Un bouton, une carte projet, une navbar : tu les écris une fois, tu les réutilises partout. Tu corriges à un seul endroit.



Un composant tient en **trois** blocs.

`<script setup>` pour la logique ·

`<template>` pour le HTML ·

`<style scoped>` pour le CSS

Ce fichier s'appelle un Single-File
Component (SFC), extension `.vue`.

`scoped` signifie que ce CSS ne s'applique
qu'à ce composant : pas de style qui déborde.



Les agents IA manipulent exactement ces outils.

Ils lancent des commandes, créent des fichiers, installent des paquets.

Si tu ne sais pas ce qu'est un package.json ou pourquoi le serveur tourne sur le port 5173, tu ne peux pas relire ce qu'un agent a fait, ni lui dire qu'il s'est trompé.

On apprend la chaîne à la main
pour pouvoir la diriger.

à la main

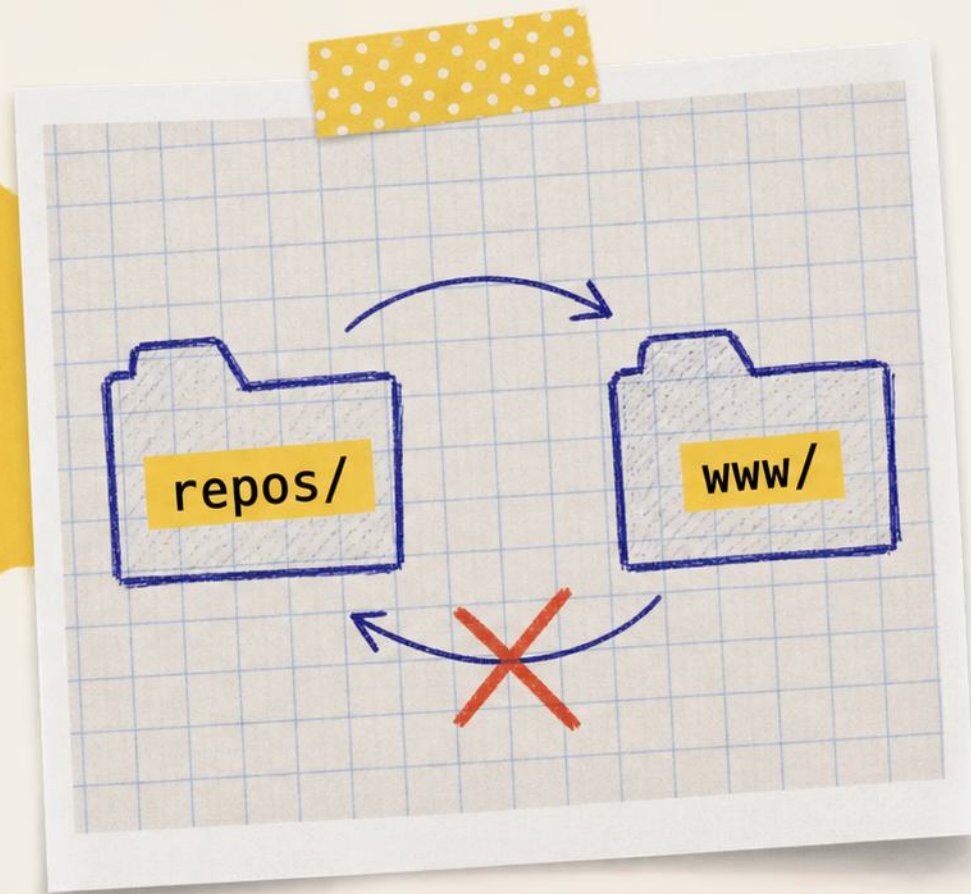
```
{  
  "name": "mon-projet",  
  "version": "1.0.0",  
  "scripts": {  
    "dev": "vite",  
    "build": "vite build"  
  },  
  "dependencies": {  
    "vite": "^5.0.0"  
  }  
}
```

```
npm run dev  
touch fichier.txt  
npm install
```

```
$ npm install  
  
added 124 packages, and audited 125 packages  
  
found 0 vulnerabilities  
  
$ npm run dev  
  
> mon-projet@1.0.0 dev  
> vite  
  
VITE v5.0.0 ready in 300 ms  
→ Local: http://localhost:5173
```

repos pour le JavaScript, www pour le PHP.

- `~/repos/` → les projets JavaScript de ce cours (Vue, Vite).
Un dossier par projet.
- `C:\laragon\www\` (Windows) ou `/Applications/MAMP/htdocs/` (Mac)
→ les projets PHP.



*On ne mélange jamais les deux.
Un projet Vue ne va pas dans www.*

Trois objectifs, pas un de plus.

On y va !

1.

Te servir du terminal
sans avoir peur
de l'écran noir.



2.

Installer ton premier
projet Vue et le voir
tourner dans le navigateur.



3.

Configurer VS Code
avec les bonnes
extensions.



Tout ça sert votre portfolio.

Le logo et les maquettes viennent du volet design ;
ici, tu les transformes en site réel, en ligne, à ton nom.
Vue + Tailwind, puis Vercel.

