

Vue.js

Criar um projeto Vue do zero pela linha de comando:

- Utilizamos o comando `npm create vue@3.7.3` (ou o `npm create vue@latest`)
 - `npm create vue@3.7.3`
 - **Versão Específica:** Este comando cria um novo projeto Vue utilizando exatamente a versão 3.7.3 do Vue.
 - **Quando usar:** Você utilizaria este comando se quiser garantir que o projeto use uma versão específica do Vue, por exemplo, para garantir compatibilidade com bibliotecas ou ferramentas que você já esteja utilizando ou para seguir uma versão recomendada por um tutorial ou projeto existente.
 - `npm create vue@latest`
 - **Versão Mais Recente:** Este comando cria um novo projeto Vue utilizando a versão mais recente disponível no momento.
 - **Quando usar:** Este comando é ideal quando você deseja iniciar um novo projeto com a versão mais recente do Vue, garantindo que você tenha acesso às últimas funcionalidades, melhorias e correções de bugs.

Após utilizar o comando e criar o projeto Vue, no terminal irá mostrar algumas perguntas (lembrando que as respostas marcadas abaixo são apenas exemplos, depois da imagem tem a tradução das perguntas o que elas significam)

```
PS C:\Users\Antonio> npm create vue@3.7.3

Vue.js - The Progressive JavaScript Framework

✓ Project name: ... cookin-up
✓ Add TypeScript? ... No / Yes
✓ Add JSX Support? ... No / Yes
✓ Add Vue Router for Single Page Application development? ... No / Yes
✓ Add Pinia for state management? ... No / Yes
✓ Add Vitest for Unit Testing? ... No / Yes
✓ Add an End-to-End Testing Solution? » No
✓ Add ESLint for code quality? ... No / Yes
```

As próximas perguntas estão em inglês. Para alternar as opções de sim (yes) e não (no), utilizamos as setas de navegação do teclado e confirmamos a escolha com o "Enter".

Nome do projeto: Aqui escreva o nome do nosso projeto, escolhido pelo grupo

1. Adicionar TypeScript ao projeto?

- **O que significa:** Pergunta se você deseja configurar o projeto para utilizar TypeScript, um superconjunto do JavaScript que adiciona tipagem estática e outras funcionalidades avançadas.
- **Quando dizer sim:** Se você quer benefícios como checagem de tipos, melhor suporte a IDEs, e pretende usar TypeScript na sua base de código.
- **Quando dizer não:** Se você prefere trabalhar apenas com JavaScript ou está criando um projeto simples que não precisa de tipagem estática.

2. Adicionar suporte JSX?

- **O que significa:** Pergunta se você deseja habilitar o suporte a JSX, uma sintaxe semelhante ao XML/HTML usada para escrever componentes em JavaScript. JSX é comumente usado com React, mas também pode ser usado no Vue.
- **Quando dizer sim:** Se você está familiarizado com JSX e prefere utilizá-lo para criar seus componentes, ou se você pretende portar código ou componentes de React para Vue.
- **Quando dizer não:** Se você planeja escrever seus templates de componentes em HTML, que é o padrão no Vue.

3. Adicionar Vue Router para desenvolvimento de aplicação de página única?

- **O que significa:** Pergunta se você deseja adicionar o Vue Router, uma biblioteca oficial do Vue.js para gerenciar rotas e navegação em aplicações de página única (SPA).
- **Quando dizer sim:** Se você está criando uma aplicação com múltiplas páginas ou seções que precisam ser navegadas sem recarregar a página inteira (SPA).
- **Quando dizer não:** Se você está construindo um projeto simples, como um componente único ou uma aplicação que não precisa de navegação entre páginas.

4. Adicionar Pinia para gerenciamento de estado?

- **O que significa:** Pergunta se você deseja adicionar o Pinia, uma biblioteca moderna e leve para gerenciamento de estado global no Vue, substituta do Vuex.
- **Quando dizer sim:** Se você precisa compartilhar e gerenciar estados (dados) entre diferentes componentes da sua aplicação.
- **Quando dizer não:** Se sua aplicação é pequena e o gerenciamento de estado global não é necessário, ou se você pretende gerenciar estados apenas dentro dos componentes individuais.

5. Adicionar Vitest para teste de unidade?

- **O que significa:** Pergunta se você deseja adicionar o Vitest, uma ferramenta moderna de testes de unidade, desenvolvida para trabalhar bem com Vue.

- **Quando dizer sim:** Se você planeja escrever e executar testes unitários para garantir que as partes individuais da sua aplicação funcionem como esperado.
- **Quando dizer não:** Se você não planeja adicionar testes unitários ao seu projeto, pelo menos inicialmente.

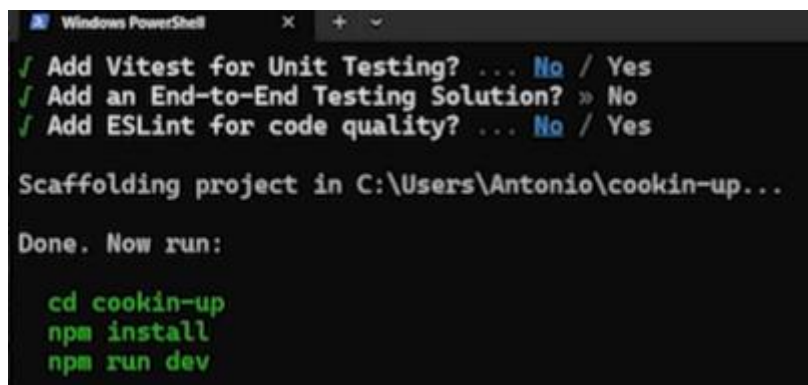
6. Adicionar solução para teste de ponta a ponta?

- **O que significa:** Pergunta se você deseja configurar testes de ponta a ponta (E2E), que simulam o comportamento do usuário para testar o fluxo completo da aplicação.
- **Quando dizer sim:** Se você precisa garantir que a aplicação funcione corretamente em cenários reais de uso, simulando interações completas de usuários.
- **Quando dizer não:** Se você não pretende realizar testes de ponta a ponta ou se o projeto é muito simples para justificar essa configuração.

7. Adicionar ESLint para qualidade de código?

- **O que significa:** Pergunta se você deseja adicionar ESLint, uma ferramenta para analisar e encontrar problemas no código, aplicando regras de estilo e boas práticas.
- **Quando dizer sim:** Se você deseja manter um código consistente, evitar erros comuns e seguir boas práticas recomendadas pelo ESLint.
- **Quando dizer não:** Se você está criando um projeto experimental, onde o estilo de código e boas práticas não são uma prioridade.

Depois de respondido as perguntas, o terminal irá mostrar as informações da imagem abaixo:



```
Windows PowerShell
✓ Add Vitest for Unit Testing? ... No / Yes
✓ Add an End-to-End Testing Solution? » No
✓ Add ESLint for code quality? ... No / Yes

Scaffolding project in C:\Users\Antonio\cookin-up...

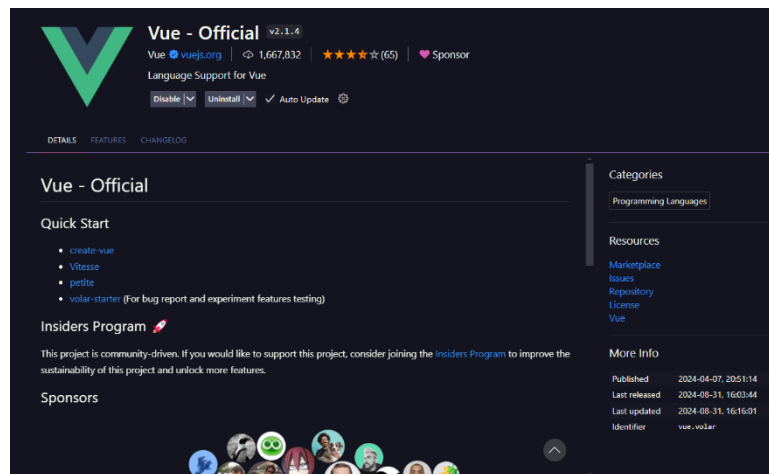
Done. Now run:

cd cookin-up
npm install
npm run dev
```

Navegue até a pasta do projeto criado **cd @nome-projeto**, depois abra o Visual Studio Code, utilize o comando **code**.

Após criado e configurado o projeto, dentro do VSCode, instale a extensão da imagem abaixo:

A que você precisará é a [Vue - Official](#). A que você **não** precisa instalar é a [TypeScript Vue Plugin \(Volar\)](#).



Com a extensão instalada, vamos voltar ao explorador de arquivos do projeto e abrir o arquivo que está na raiz, chamado **package.json**. Este arquivo é padrão para qualquer projeto Node.

Nesse arquivo, a partir da linha 12, temos uma lista de dependências na qual a única listada nessa parte é a vue, um pacote da própria Vue que executa todas as configurações necessárias.

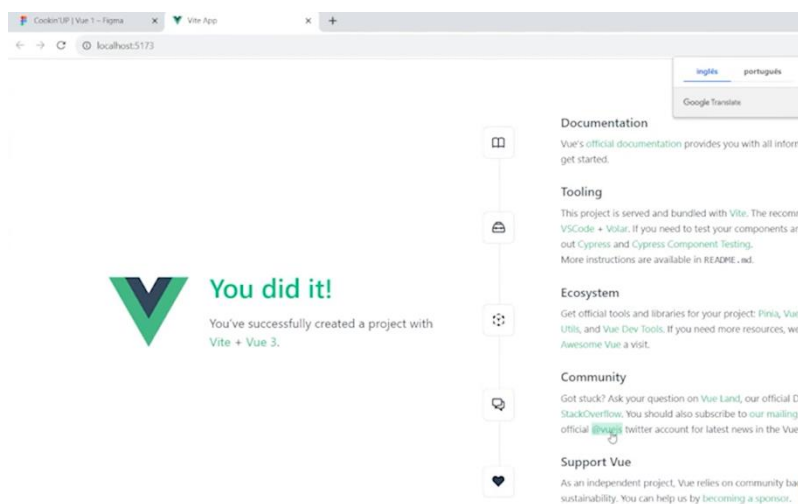
A partir da linha 15, temos as devDependencies, que listam outras ferramentas de terceiros que Vue também utiliza para conseguir executar o projeto.

Vamos retornar ao VSCode e abrir o terminal integrado, o qual vamos usar a partir de agora. Vamos digitar **npm install**.

Ao pressionar "Enter", ele instalará todas as dependências do package.json diretamente da internet. Esse processo pode levar algum tempo.

Agora, o próximo passo é executar o outro comando que o terminal nos forneceu: **npm run dev**

Após isso, irá aparecer no terminal um link <http://localhost:5173> clique nele e pronto, a tela que aparece no google é essa:



Para saber mais: O arquivo package.json

Abaixo está como o arquivo package.json do projeto exemplo é:

```
{
  "name": "cookin-up",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "dev": "vite",
    "build": "run-p type-check build-only",
    "preview": "vite preview",
    "build-only": "vite build",
    "type-check": "vue-tsc --noEmit -p tsconfig.app.json --composite false"
  },
  "dependencies": {
    "vue": "^3.3.4"
  },
  "devDependencies": {
    "@tsconfig/node18": "^18.2.0",
    "@types/node": "^18.17.5",
    "@vitejs/plugin-vue": "^4.3.1",
    "@vue/tsconfig": "^0.4.0",
    "npm-run-all": "^4.1.5",
    "typescript": "~5.1.6",
    "vite": "^4.4.9",
    "vue-tsc": "^2.0.29"
  }
}
```

=====

Entendendo os atributos desse JSON:

[Para saber mais sobre JSON](#)

"name" (nome): o nome do pacote (ou projeto) Node.js.

"version" (versão): a versão atual do pacote. O NPM utiliza o chamado versionamento semântico (SemVer). Você pode ler mais sobre versionamento e como ele é utilizado neste artigo: <https://www.alura.com.br/artigos/versionamento-semantic-breve-introducao>

"private" (privado): define se o seu pacote será privado ou se ele estará disponível para que outras pessoas da comunidade possam utilizá-lo.

"scripts": essa sessão tem alguns scripts pré-definidos, mas você também pode definir os seus personalizados. Leia sobre scripts na documentação do npm para mais informações: <https://docs.npmjs.com/cli/v8/using-npm/scripts>

"dependencies" (dependências): define a lista de pacotes necessários para executar seu projeto num ambiente de produção.

"devDependencies" (dependências de desenvolvimento): define a lista de pacotes necessários para executar o projeto em um ambiente de desenvolvimento e de testes.

=====

Existem outras configurações que podem existir nesse arquivo, como o repositório do Git, homepage, peerDependencies, entre outras que podem ser visualizadas na [documentação oficial do NPM](#)