



FPT POLYTECHNIC



THỰC HỌC – THỰC NGHIỆP



Conceive Design Implement Operate

FRONT-END FRAMEWORK

TỔNG QUAN VỀ VUE.JS

<http://www.poly.edu.vn>

- ⦿ Nắm được các khái niệm cơ bản về Vue.js
- ⦿ Cài đặt môi trường phát triển Vue.js với thư viện Vitejs
- ⦿ Giới thiệu về các cú pháp trong Vue.js
- ⦿ Hiểu được cách sử dụng Bootstrap để tạo giao diện trong Vue



- 📖 Framework là gì?
- 📖 Giới thiệu về Framework VueJS
- 📖 Cài đặt môi trường phát triển
- 📖 Tạo và thực thi Project với Vue.js
- 📖 Cài đặt và sử dụng Bootstrap trong Vuejs
- 📖 Giới thiệu một vài cú pháp trong VueJS





PHẦN 1: TỔNG QUAN VỀ VUEJS

- ❑ Framework là bộ công cụ lập trình.
- ❑ Giúp tổ chức và quản lý mã nguồn tốt hơn
- ❑ Tiết kiệm thời gian và công sức khi phát triển website
- ❑ Giúp cải thiện tốc độ tải và phản hồi của ứng dụng.



- ❑ **Vue** (phát âm /vju:/, như “**view**”) là một framework JavaScript
- ❑ Dùng để xây dựng các giao diện người dùng (UI).
- ❑ Cú pháp đơn giản và thân thiện với người mới bắt đầu.
- ❑ Sử dụng [Virtual DOM](#) để tối ưu hiệu năng ứng dụng



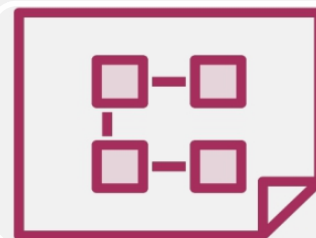
TẠI SAO DÙNG VUEJS?



HTML/CSS/JS
tách biệt



Reactive Data
Binding



Single-File
Components



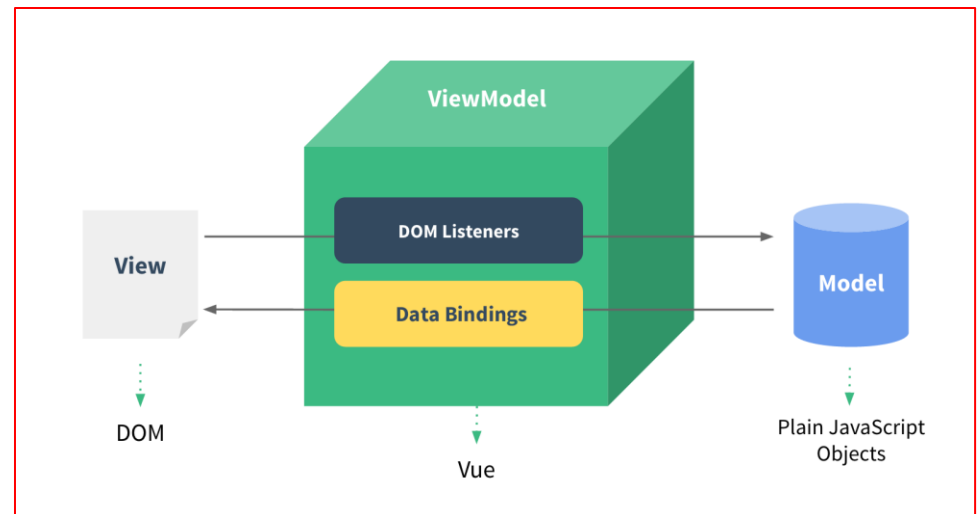
Làm việc tốt với
Back-end

- ❑ **VueJS** được tạo bởi Evan You, cựu nhân viên Google
- ❑ **VueJS** bắt đầu phát triển vào năm 2013
- ❑ Ra mắt phiên bản đầu tiên chính thức năm 2014
- ❑ Phiên bản mới nhất tính đến thời điểm hiện tại (7/2024) là v3.4.33



VUE HOẠT ĐỘNG NHƯ NÀO?

- ❑ Vue.js tự động đồng bộ dữ liệu giữa Model và View thông qua ViewModel, giúp mã nguồn dễ bảo trì và phát triển hơn. Mô hình này được gọi là **MVVM**
- ❑ **View**: Giao diện người dùng
- ❑ **ViewModel**:
 - Data Bindings: Kết nối dữ liệu từ Model với View.
 - DOM Listeners: Theo dõi sự kiện trên DOM và cập nhật Model.
- ❑ **Model**: Chứa dữ liệu.





Visual Studio Code

```
4. vue create hello-world (node)
Vue CLI v3.4.0
? Please pick a preset: Manually select features
? Check the features needed for your project:
  > Babel
    TypeScript
    Progressive Web App (PWA) Support
    Router
    Vuex
    CSS Pre-processors
  > Linter / Formatter
    Unit Testing
    E2E Testing
```



VITE



- ❑ Truy cập: <https://nodejs.org>
- ❑ Tải và cài đặt

Run JavaScript Everywhere

Node.js® is a free, open-source, cross-platform JavaScript runtime environment that lets developers create servers, web apps, command line tools and scripts.

Download Node.js (LTS) 


Downloads Node.js **v20.16.0**¹ with long-term support. Node.js can also be installed via package managers.

Want new features sooner? Get **Node.js v22.5.1**¹ instead.

Create an HTTP Server Write Tests Read and Hash a File Streams Pipeline Work with Threads

```
1 // server.mjs
2 import { createServer } from 'node:http';
3
4 const server = createServer((req, res) => {
5   res.writeHead(200, { 'Content-Type': 'text/plain' });
6   res.end('Hello World!\n');
7 });
8
9 // starts a simple http server locally on port 3000
10 server.listen(3000, '127.0.0.1', () => {
11   console.log('Listening on 127.0.0.1:3000');
12 });
13
14 // run with 'node server.mjs'
```

JavaScript

 Copy to clipboard

Learn more what Node.js is able to offer with our Learning materials.

- ❑ Cách 1: CDN (Content delivery Network)
- ❑ Tạo file index.html và nhúng link sau ở đầu file

```
<script src="https://unpkg.com/vue@3/dist/vue.global.js"></script>
```

```
<script src="https://unpkg.com/vue@3/dist/vue.global.js"></script>

<div id="app">{{ message }}</div>

<script>
  const { createApp, ref } = Vue;

  createApp({
    setup() {
      const message = ref("Hello vue!");
      return {
        message,
      };
    },
  }).mount("#app");
</script>
```

➤ [Xem ví dụ](#)

Cách 2: Sử dụng NPM cài đặt Vitejs

- ❑ Vitejs là công cụ để tạo 1 project **Vuejs** cơ bản và có hiệu suất cao
- ❑ Do chính tác giả Vue tạo ra
- ❑ Sử dụng **npm** để tải **Vitejs**
- ❑ Sử dụng Terminal (hoặc Command Prompt):

```
npm create vue@latest front-end-framework
```



- ❑ Lệnh này sẽ cài đặt và thực thi **create-vue**, công cụ tạo dự án chính thức của **Vue**.
- ❑ Bạn sẽ được yêu cầu chọn các tính năng tùy chọn như hình sau.

```
✓ 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 / Cypress / Nightwatch / Playwright
✓ Add ESLint for code quality? ... No / Yes
✓ Add Prettier for code formatting? ... No / Yes
✓ Add Vue DevTools 7 extension for debugging? (experimental) ... No / Yes

Scaffolding project in ./<front-end-framework>...
Done.
```

CÀI ĐẶT MÔI TRƯỜNG VUEJS

❑ Sử dụng Terminal VSC:

```
cd front-end-framework
npm install
npm run dev
```

Truy cập thư mục

Cài đặt các thư viện

Chạy ứng dụng Vue

❑ Bật trình duyệt

VITE v5.3.5 ready in 659 ms

- **Local:** <http://localhost:5173/>
- **Network:** use **--host** to expose
- press **h + enter** to show help

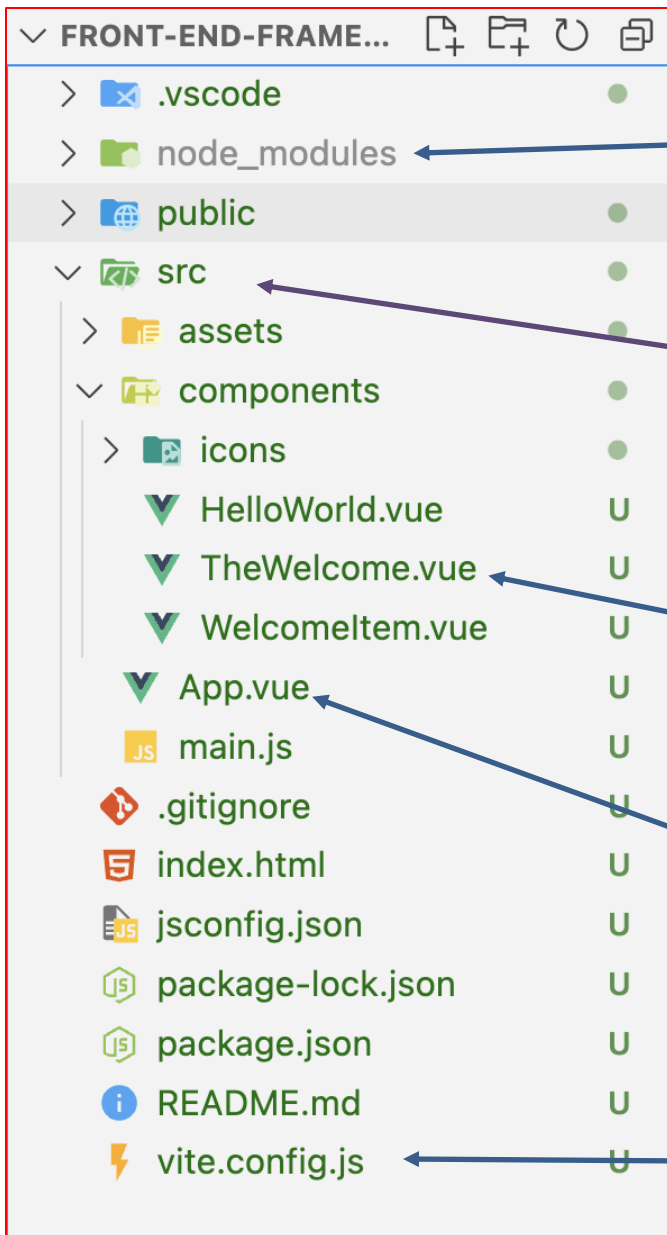


You did it!

You've successfully created a project with
Vite + Vue 3.



TỔ CHỨC CODE TRONG VUEJS



Chứa các module và dependencies

Chứa mã nguồn chính của ứng dụng

Các component con

Root component

File cấu hình vitejs

- ❑ Mọi ứng dụng Vue đều bắt đầu bằng việc tạo một application instance mới với hàm **createApp**:

```
import { createApp } from "vue";  
const app = createApp({  
  /* Các tùy chọn của thành phần gốc */  
});
```

- ❑ Thành phần gốc(root component) là thành phần đầu tiên trong một ứng dụng Vue.
- ❑ Chứa các thành phần con và là điểm khởi đầu của ứng dụng.
- ❑ Được truyền vào hàm **createApp** để tạo ra ứng dụng.
- ❑ Ví dụ: App.vue thường là thành phần gốc.

```
// main.js là file đầu tiên được chạy khi ứng dụng Vue.js được khởi tạo.  
import { createApp } from "vue";  
// Nhập thành phần gốc App từ một single-file component  
import App from "./App.vue";  
createApp(App).mount("#app");
```

- **<script setup>**: Đơn giản hóa khai báo và sử dụng dữ liệu trong Vue 3.
- **<template>**: Định nghĩa cấu trúc giao diện của thành phần bằng HTML.
- **<style scoped>**: Áp dụng CSS chỉ cho thành phần hiện tại.

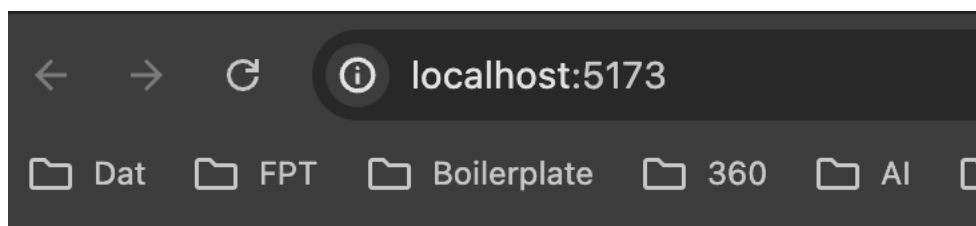
```
<script setup>
// Code js của bạn ở đây
</script>

<template>
  <!-- Code html của bạn ở đây -->
</template>

<style scoped>
/* Code css của bạn ở đây */
</style>
```

- ❑ Truy cập file **App.vue** cập nhật code sau và lưu lại.

```
<template>
  <h1>Hello, VueJs</h1>
</template>
```



Hello, VueJS!

➤ [Xem ví dụ](#)

demo




PHẦN 2: STYLE VÀ TEMPLATE SYNTAX

Sử dụng framework css **Bootstrap** ta thực hiện 2 bước

- Cài đặt Bootstrap: **npm i bootstrap**
- Để nhúng thư viện bootstrapp vào dự án vue:
Mở tệp **main.js** và thêm các dòng sau:

```
import { createApp } from "vue";  
import App from "./App.vue";  
  
import "bootstrap/dist/css/bootstrap.min.css";  
import "bootstrap/dist/js/bootstrap.bundle.min.js";  
  
createApp(App).mount("#app");
```

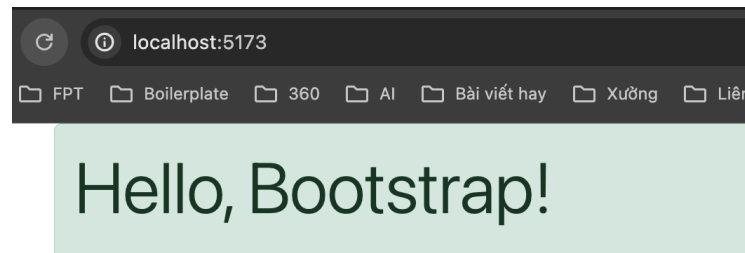


➤ [Xem ví dụ](#)

Kiểm tra bootstrap có hoạt động chưa?

Mở file **App.vue** trong thư mục app và cập nhật code như sau:

```
<template>
  <div id="app">
    <div class="container">
      <div class="alert alert-success">
        <h1 class="display-4">Hello, Bootstrap!</h1>
      </div>
    </div>
  </div>
</template>
```



➤ [Xem ví dụ](#)

- ❑ **Vue** sử dụng cú pháp mẫu dựa trên HTML cho phép bạn liên kết một cách khai báo giữa DOM được render với dữ liệu của của đối tượng Vue bên dưới
- ❑ Hình thức ràng buộc dữ liệu cơ bản nhất là text interpolation
- ❑ Cú pháp “mustache” với hai dấu ngoặc. :
`Message: {{ msg }}`

- **<script setup>**: Đơn giản hóa khai báo và sử dụng dữ liệu trong Vue 3.
- **<template>**: Định nghĩa cấu trúc giao diện của thành phần bằng HTML.
- **<style scoped>**: Áp dụng CSS chỉ cho thành phần hiện tại.

```
<script setup>
// Code js của bạn ở đây
</script>

<template>
  <!-- Phần này là code UI của bạn -->
</template>

<style scoped>
/* Code css của bạn ở đây */
</style>
```

```
<script setup>
const courseName = "Framework Vue 3";
const courseLevel = "Nâng cao";
const courseTime = 30; // giờ
const courselsActive = true;
</script>
<template>
<div class="container my-5">
  <h1 class="display-4 mb-3">Thông tin:</h1>
  <ul class="list-group">
    <li class="list-group-item">
      Tên khóa học: <strong>{{courseName}}</strong>
    </li>
    <li class="list-group-item">Cấp độ: {{courseLevel}}</li>
    <li class="list-group-item">Thời gian: {{courseTime}} giờ</li>
    <li class="list-group-item">
      Trạng thái: {{courselsActive ? "Đang mở" : "Đã đóng"}}
    </li>
  </ul>
</div>
</template>
```

Dữ liệu kiểu chuỗi

Thông tin khóa học:

Tên khóa học: Framework Vue 3

Cấp độ: Nâng cao

Thời gian: 30 giờ

Trạng thái: Đang mở

➤ [Xem Ví dụ](#)

demo

Ngoài ra còn rất nhiều cú pháp khác như:

- ❖ Attribute Binding: **v-bind**
- ❖ Conditional Rendering: **v-if, v-else, v-else-if**
- ❖ List Rendering: **v-for**
- ❖ Two-way Binding: **v-model**
- ❖ Event Handling: **v-on**
- ❖ Conditional Display: **v-show**
- ❖ Class Binding: **v-bind:class**
- ❖ Style Binding: **v-bind:style**

Chúng ta sẽ vào chi tiết ở các bài tiếp theo

- ☑ Framework là gì?
- ☑ Giới thiệu về Framework VueJS
- ☑ Môi trường phát triển
- ☑ Cài đặt
- ☑ Tạo và thực thi Project với Vue.js
- ☑ Kiến trúc tổ chức của Vue.js
- ☑ Cài đặt và sử dụng Bootstrap trong Vue.js
- ☑ Giới thiệu một vài cú pháp trong VueJS



thank
you!