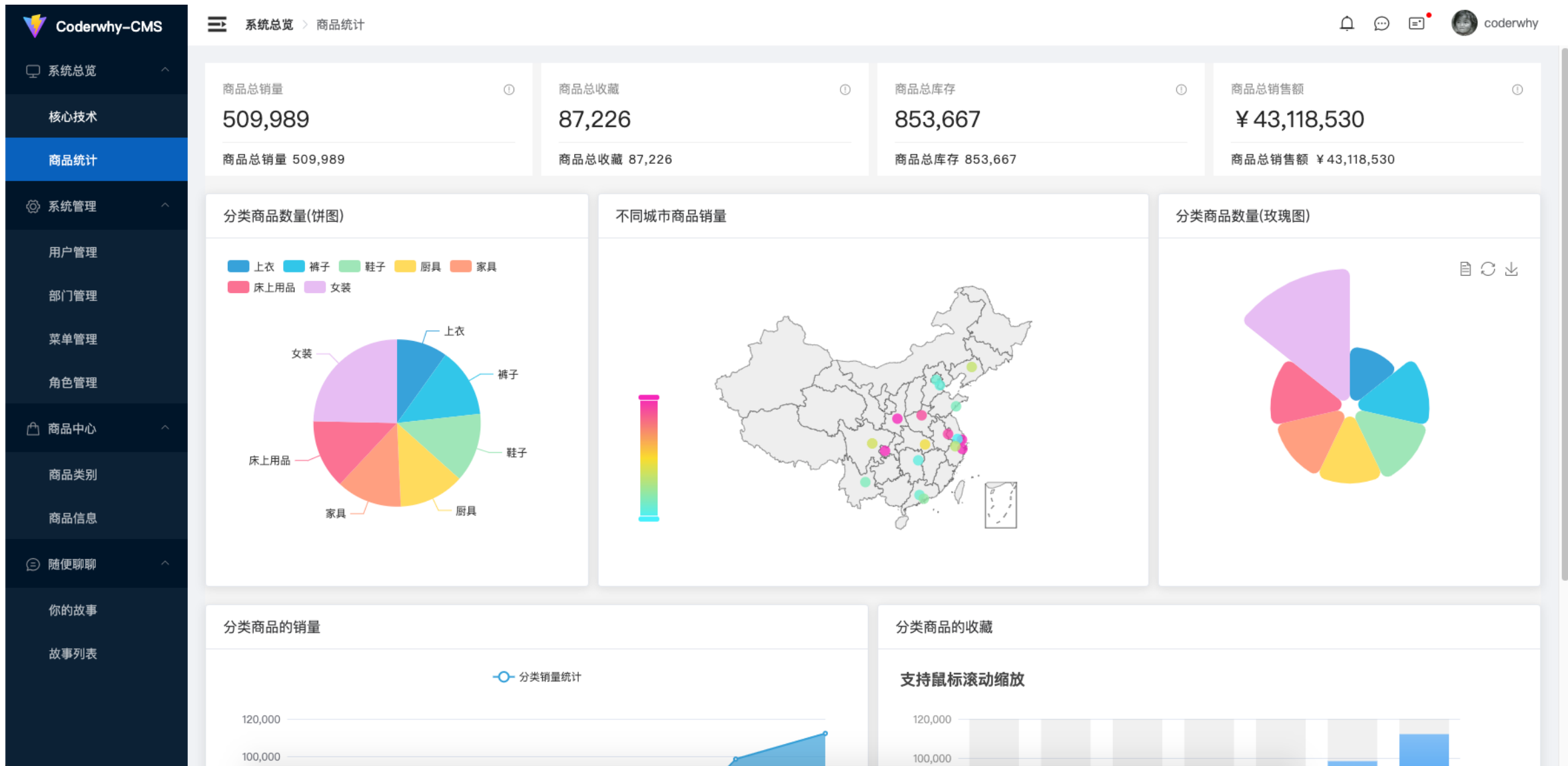


Vue3 + TypeScript项目实战

王红元 coderwhy

项目介绍 - 后台管理系统

- 项目预览地址: <http://152.136.185.210>
 - 账号1: coderwhy 密码: 123456 账号2: coderdemo 密码: 123456
- 技术栈介绍:
 - 开发工具 :Visual Studio Code
 - 编程语言 :TypeScript 4.x + JavaScript
 - 构建工具 :Vite 3.x / Webpack5.x
 - 前端框架 :Vue 3.x + setup
 - 路由工具 :Vue Router 4.x
 - 状态管理 :Vuex 4.x / Pinia
 - UI 框架 :Element Plus
 - 可视化 :Echart5.x
 - 工具库 :@vueuse/core + dayjs + countup.js等等
 - CSS 预编译 :Sass / Less
 - HTTP 工具 : Axios
 - Git Hook 工具 :husky
 - 代码规范 :EditorConfig + Prettier + ESLint
 - 提交规范 :Commitizen + Commitlint
 - 自动部署 :Centos + Jenkins + Nginx



用户名

请输入用户名

真实姓名

请输入真实姓名

手机号码

请输入手机号码

状态

请选择状态

创建时间



开始时间



结束时间

重置

查询

用户列表

新建数据

<input type="checkbox"/>	序号	用户名	真实名	手机号码	状态	创建时间	更新时间	操作
<input type="checkbox"/>	1	coreCoder	coreCoder	13812345678	启用	2022-10-13 04:32:14	2022-10-13 04:35:30	编辑 删除
<input type="checkbox"/>	2	james	詹姆斯1号	13322223338	启用	2022-10-13 00:03:05	2022-10-13 02:20:41	编辑 删除
<input type="checkbox"/>	3	zxx	周星星1号	18892321212	启用	2022-10-12 15:06:10	2022-10-12 21:35:57	编辑 删除
<input type="checkbox"/>	4	curry	库里	17388655561111	启用	2022-10-12 06:51:14	2022-10-12 15:25:32	编辑 删除
<input type="checkbox"/>	5	lyh	李银河	17754456666	启用	2021-05-02 15:24:12	2021-08-20 12:07:23	编辑 删除
<input type="checkbox"/>	6	wxb	王小波	18855556666	启用	2021-05-02 15:24:12	2021-05-02 15:26:20	编辑 删除
<input type="checkbox"/>	7	kobe	kobe	16655556666	启用	2021-05-02 15:24:12	2021-05-02 15:26:20	编辑 删除
<input type="checkbox"/>	8	lily	lily	13355556666	启用	2021-05-02 15:24:12	2021-05-02 15:26:20	编辑 删除
<input type="checkbox"/>	9	coderdemo	demo	17766665555	启用	2021-08-23 15:24:12	2021-08-23 15:24:21	编辑 删除



创建Vue项目

■ 方式一：Vue CLI

- 基于webpack工具;
- 命令: vue create

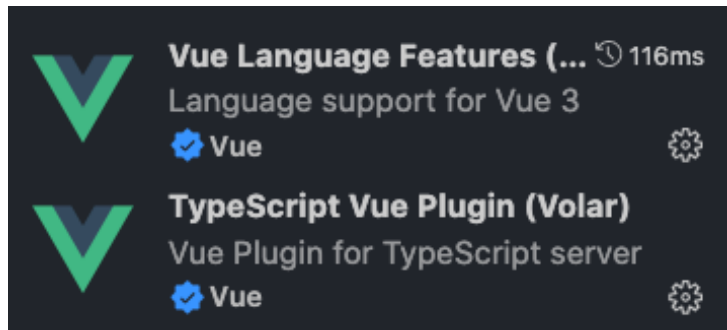
■ 方式二：create-vue

- 基于vite工具;
- 命令: npm init vue@latest

■ 项目配置:

- 配置项目的icon
- 配置项目的标题
- 配置项目别名等 (vite默认配置)
- 配置tsconfig.json

- 安装volar和volar+TS插件:



- 配置vue文件模块:

```
declare module '*.vue' {  
  import { DefineComponent } from 'vue'  
  const component: DefineComponent  
  export default component  
}
```

配置项目代码规范 – 见markdown

大纲

项目搭建规范

一. 代码规范

1.1. 集成editorconfig配置

1.2. 使用prettier工具

1.3. 使用ESLint检测

1.4. git Husky和eslint (后续)

1.5. git commit规范 (后续)

1.5.1. 代码提交风格

1.5.2. 代码提交验证

二. 接口文档

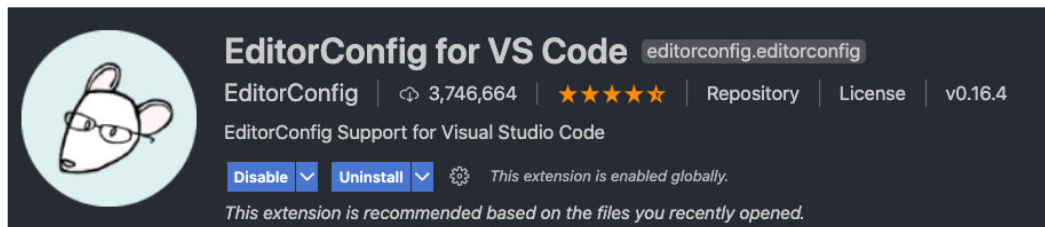
项目搭建和接口文档

1.1. 集成editorconfig配置

EditorConfig 有助于为不同 IDE 编辑器上处理同一项目的多个开发人员维护一致的编码风格。

```
1 # http://editorconfig.org
2
3 root = true
4
5 [*] # 表示所有文件适用
6 charset = utf-8 # 设置文件字符集为 utf-8
7 indent_style = space # 缩进风格 (tab | space)
8 indent_size = 2 # 缩进大小
9 end_of_line = lf # 控制换行类型(lf | cr | crlf)
10 trim_trailing_whitespace = true # 去除行尾的任意空白字符
11 insert_final_newline = true # 始终在文件末尾插入一个新行
12
13 [* .md] # 表示仅 md 文件适用以下规则
14 max_line_length = off
15 trim_trailing_whitespace = false
```

VSCode需要安装一个插件：EditorConfig for VS Code



1.2. 使用prettier工具

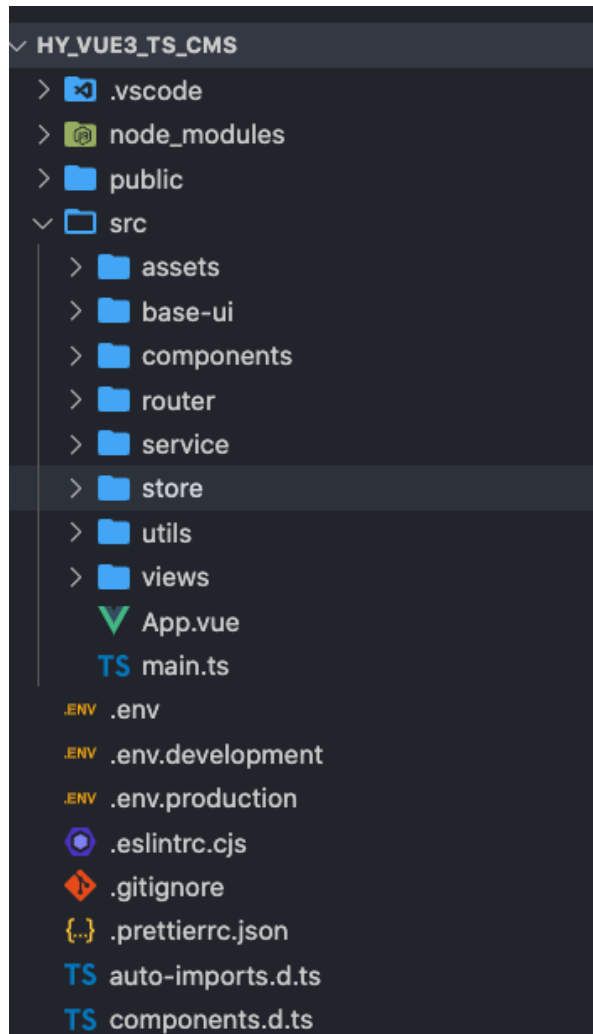
Prettier 是一款强大的代码格式化工具，支持 JavaScript、TypeScript、CSS、SCSS、Less、JSX、Angular、Vue、GraphQL、JSON、Markdown 等语言，基本上前端能用到的文件格式它都可以搞定，是当下最流行的代码格式化工具。

1. 安装prettier

```
1 npm install prettier -D
```

项目目录结构划分

- 对项目进行目录结构的划分:

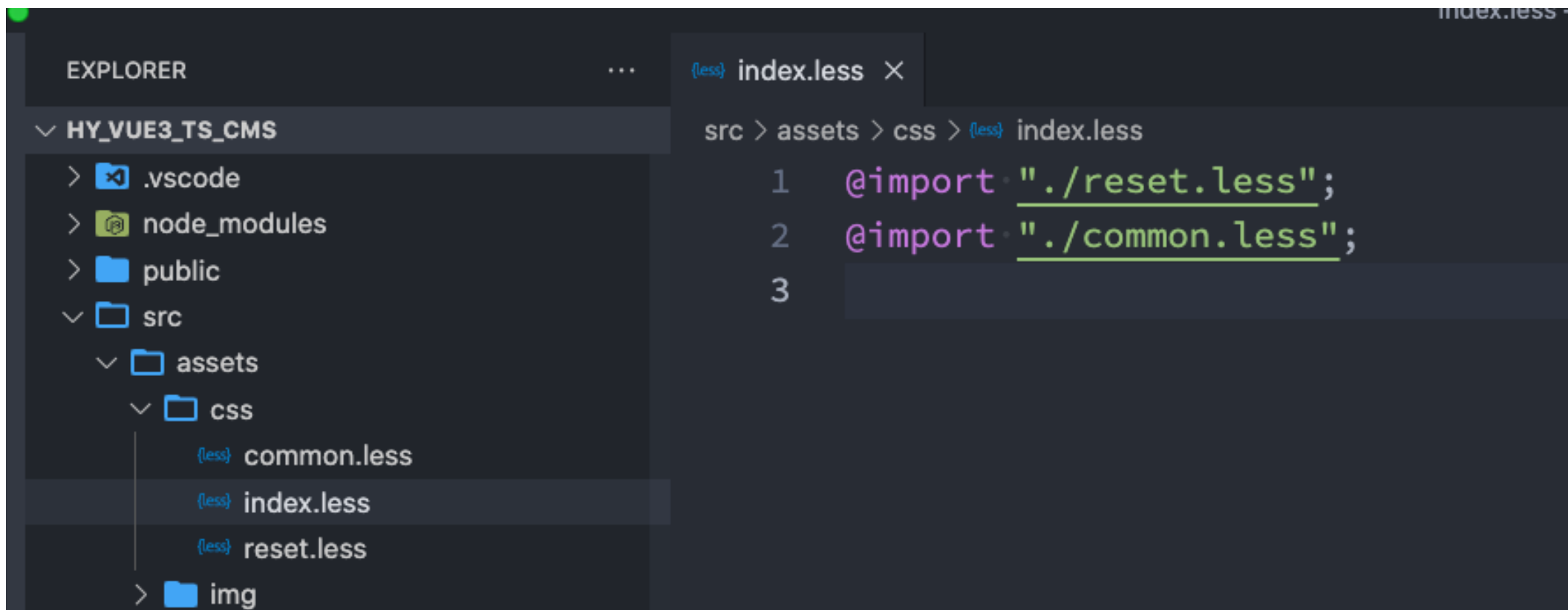


CSS样式的重置

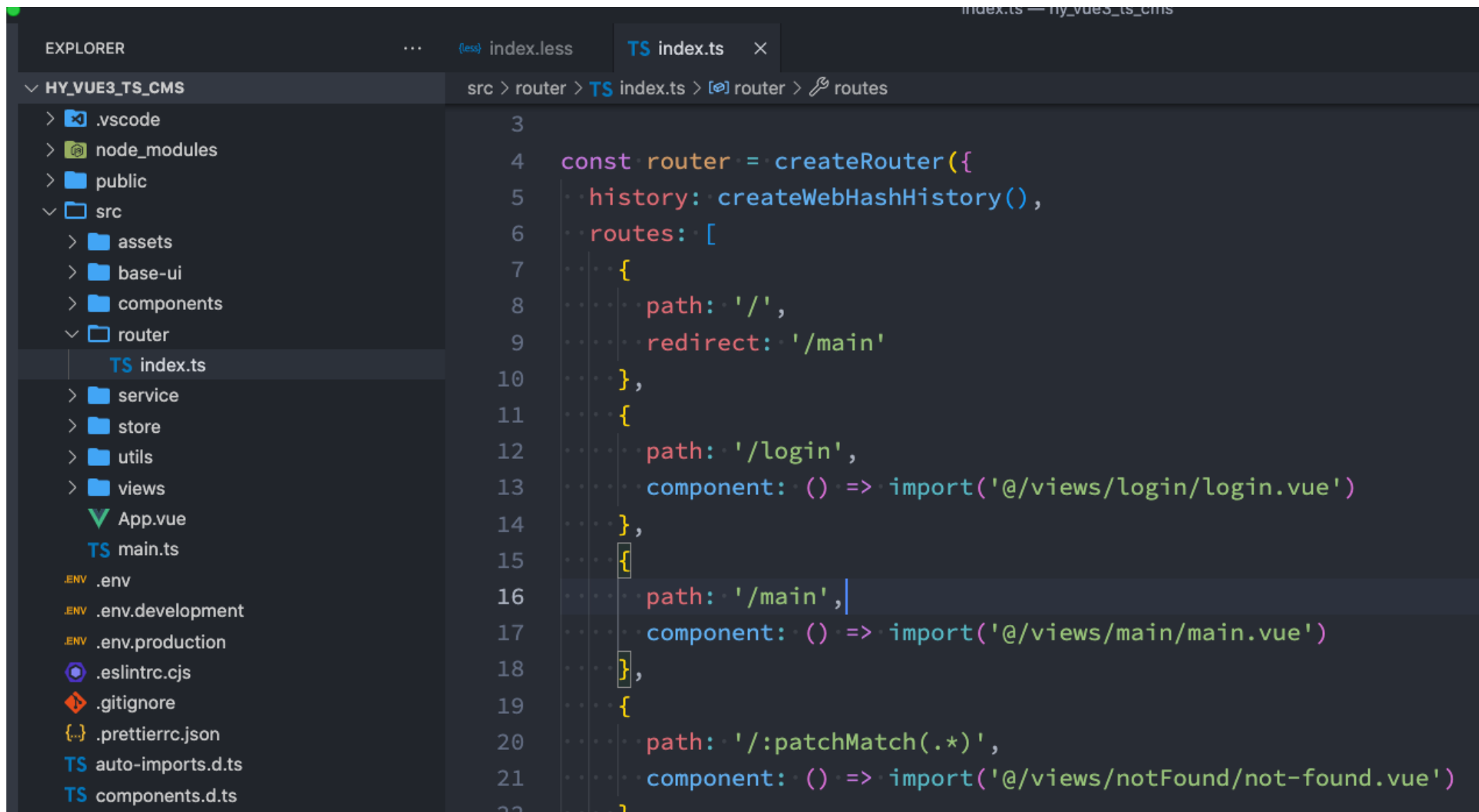
■ 对默认CSS样式进行重置:

▣ normalize.css

▣ reset.css



全家桶 - 路由配置

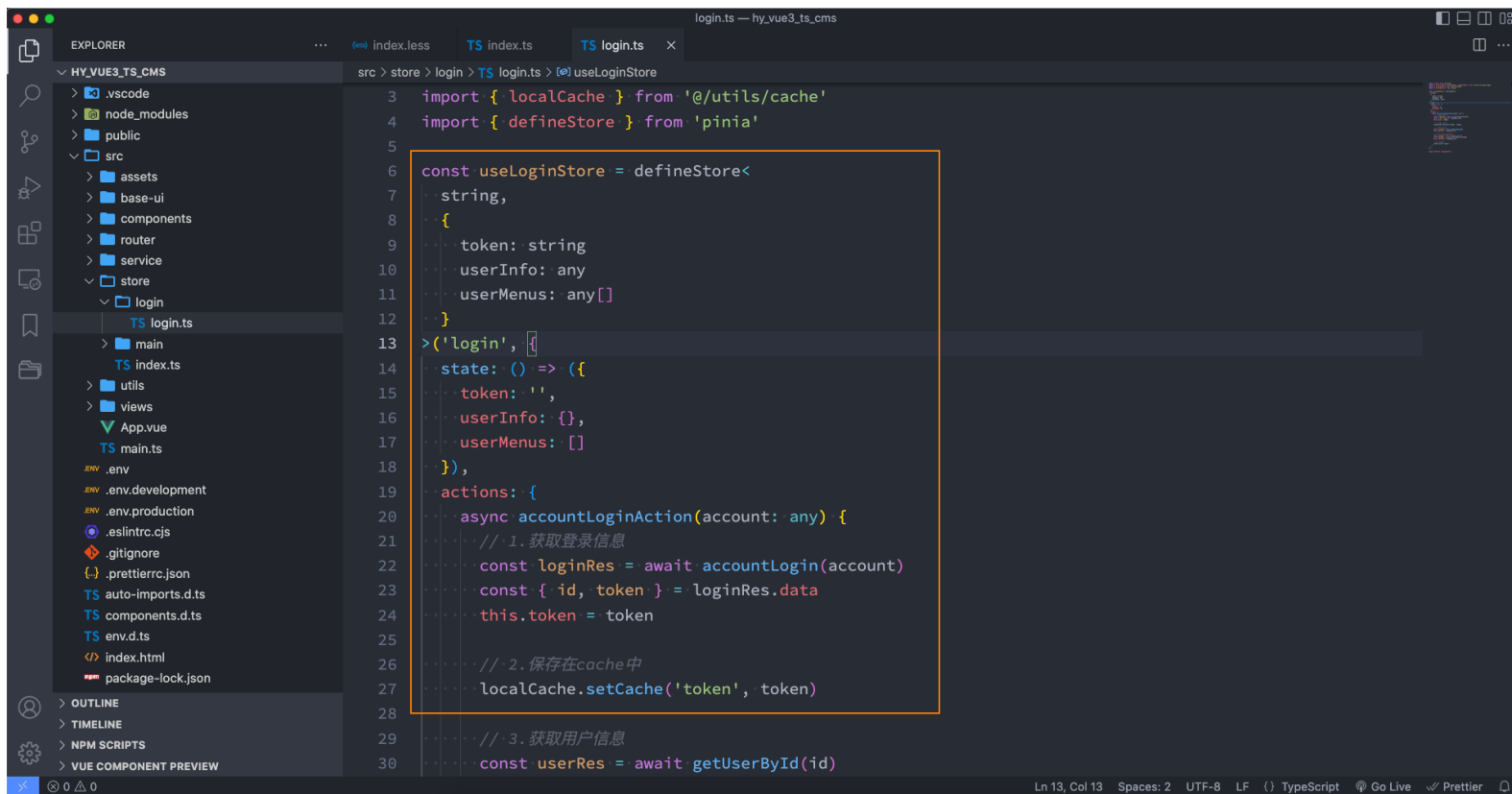


The screenshot shows a VS Code editor with a project named 'HY_VUE3_TS_CMS'. The Explorer sidebar on the left shows the file structure, with the 'router' directory expanded. The 'index.ts' file in the 'router' directory is open in the editor. The breadcrumb navigation at the top indicates the path: 'src > router > TS index.ts > router > routes'. The code in 'index.ts' defines a Vue Router instance with the following configuration:

```
3
4  const router = createRouter({
5    history: createWebHashHistory(),
6    routes: [
7      {
8        path: '/',
9        redirect: '/main'
10     },
11     {
12       path: '/login',
13       component: () => import('@views/login/login.vue')
14     },
15     {
16       path: '/main',
17       component: () => import('@views/main/main.vue')
18     },
19     {
20       path: '/*:patchMatch(.*)',
21       component: () => import('@views/notFound/not-found.vue')
22     }
23   ]
24 })
```

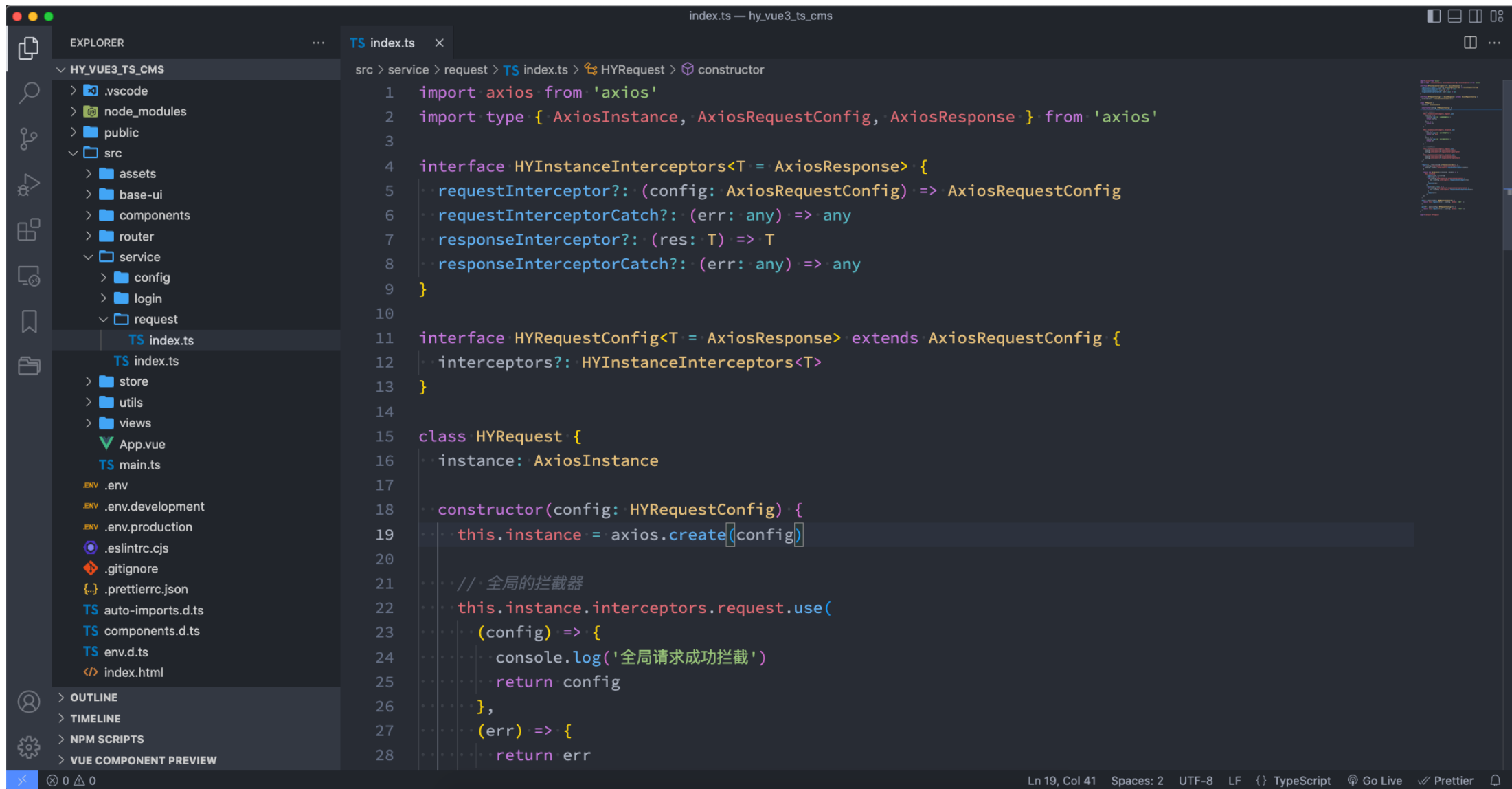
■ 状态管理的选择:

- ❑ vuex: 目前依然使用较多的状态管理库;
- ❑ pinia: 强烈推荐, 未来趋势的状态管理库;



```
3 import { localCache } from '@/utils/cache'
4 import { defineStore } from 'pinia'
5
6 const useLoginStore = defineStore<
7   string,
8   {
9     token: string
10    userInfo: any
11    userMenus: any[]
12  },
13  >('login', {
14    state: () => ({
15      token: '',
16      userInfo: {},
17      userMenus: []
18    }),
19    actions: {
20      async accountLoginAction(account: any) {
21        // 1. 获取登录信息
22        const loginRes = await accountLogin(account)
23        const { id, token } = loginRes.data
24        this.token = token
25
26        // 2. 保存在cache中
27        localCache.setCache('token', token)
28
29        // 3. 获取用户信息
30        const userRes = await getUserById(id)
```

网络请求封装axios



```
1 import axios from 'axios'
2 import type { AxiosInstance, AxiosRequestConfig, AxiosResponse } from 'axios'
3
4 interface HYInstanceInterceptors<T = AxiosResponse> {
5   requestInterceptor?: (config: AxiosRequestConfig) => AxiosRequestConfig
6   requestInterceptorCatch?: (err: any) => any
7   responseInterceptor?: (res: T) => T
8   responseInterceptorCatch?: (err: any) => any
9 }
10
11 interface HYRequestConfig<T = AxiosResponse> extends AxiosRequestConfig {
12   interceptors?: HYInstanceInterceptors<T>
13 }
14
15 class HYRequest {
16   instance: AxiosInstance
17
18   constructor(config: HYRequestConfig) {
19     this.instance = axios.create(config)
20
21     // 全局的拦截器
22     this.instance.interceptors.request.use(
23       (config) => {
24         console.log('全局请求成功拦截')
25         return config
26       },
27       (err) => {
28         return err
```

区分 development和production 环境

■ Vite的环境变量：

■ Vite 在一个特殊的 `import.meta.env` 对象上暴露环境变量。这里有一些在所有情况下都可以使用的内建变量：

❑ `import.meta.env.MODE`: {string} 应用运行的模式。

❑ `import.meta.env.PROD`: {boolean} 应用是否运行在生产环境。

❑ `import.meta.env.DEV`: {boolean} 应用是否运行在开发环境 (永远与 `import.meta.env.PROD`相反)。

❑ `import.meta.env.SSR`: {boolean} 应用是否运行在 server 上。

■ Vite 使用 `dotenv` 从你的 环境目录 中的下列文件加载额外的环境变量：

```
.env           # 所有情况下都会加载
.env.local     # 所有情况下都会加载，但会被 git 忽略
.env.[mode]    # 只在指定模式下加载
.env.[mode].local # 只在指定模式下加载，但会被 git 忽略
```

```
index.ts  .ENV .env.development ×
.ENV .env.development
1  VITE_BASE_URL=http://codercba.com/dev:1888
2  VITE_TIME_OUT=10000
```

■ 只有以 `VITE_` 为前缀的变量才会暴露给经过 vite 处理的代码。



Element-Plus集成



■ Element Plus UI组件库

Element Plus

基于 Vue 3，面向设计师和开发者的组件库



■ 集成方案：

- 安装：<https://element-plus.gitee.io/zh-CN/guide/installation.html>
- 导入：<https://element-plus.gitee.io/zh-CN/guide/quickstart.html>