Vue3 大事件管理系统

黑马程序员
www.itheima.com

传智教育旗下
高端IT教育品牌

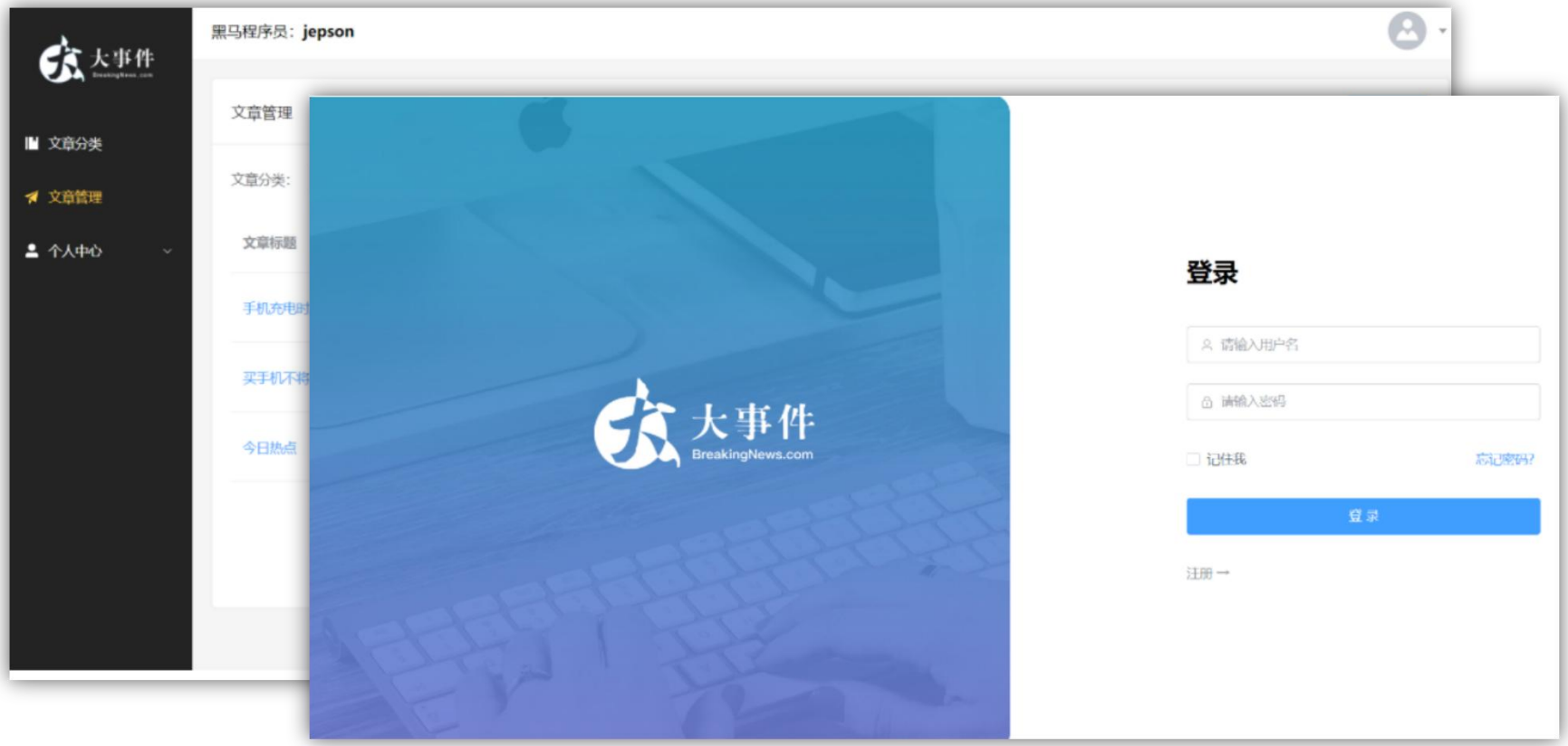
01 大事件项目介绍 和 创建

Vue3 大事件管理系统

在线演示： https://fe-bigevent-web.itheima.net/login

接口文档： https://apifox.com/apidoc/shared-26c67aee-0233-4d23-aab7-08448fdf95ff/api-93850835

基地址： http://big-event-vue-api-t.itheima.net



pnpm 包管理器 - 创建项目

一些优势：比同类工具快2倍左右、节省磁盘空间... https://www.pnpm.cn/

安装方式：npm install -g pnpm

创建项目：pnpm create vue

npm	yarn	pnpm
npm install	yarn	pnpm install
npm install axios	yarn add axios	pnpm add axios
npm install axios -D	yarn add axios -D	pnpm add axios -D
npm uninstall axios	yarn remove axios	pnpm remove axios
npm run dev	yarn dev	pnpm dev

创建项目

```
pnpm create vue
```

```
Vue.js - The Progressive JavaScript Framework

√ Project name: ... Vue3-big-event-admin
√ Package name: ... vue3-big-event-admin
√ 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
√ Add Prettier for code formatting? ... No / Yes

Scaffolding project in E:\CODE\Vue3-big-event-admin...

Done. Now run:

  cd Vue3-big-event-admin
  pnpm install
  pnpm format
  pnpm dev
```

1. 进入项目目录
2. 安装依赖
3. 启动项目

02 Eslint 配置代码风格

Eslint 配置代码风格

配置文件 .eslintrc.cjs

1. prettier 风格配置 https://prettier.io
   1. 单引号
   2. 不使用分号
   3. 宽度80字符
   4. 不加对象|数组最后逗号
   5. 换行符号不限制（win mac 不一致）

2. vue组件名称多单词组成（忽略index.vue）

3. props解构（关闭）

提示：安装Eslint且配置保存修复，不要开启默认的自动保存格式化

```
4  module.exports = {
5    root: true,
6  › extends: […
11   ],
12 › parserOptions: {…
14   },
15   rules: {
16     'prettier/prettier': [  1
17       'warn',
18       {
19         singleQuote: true,
20         semi: false,
21         printWidth: 80,
22         trailingComma: 'none',
23         endOfLine: 'auto'
24       }
25     ],
26     'vue/multi-word-component-names': [ 2
27       'warn',
28       {
29         ignores: ['index']
30       }
31     ],
32     'vue/no-setup-props-destructure': ['off'] 3
33   }
34 }
35
```

03 配置代码检查工作流

提交前做代码检查

1. 初始化 git 仓库，执行 git init 即可
2. 初始化 husky 工具配置，执行 pnpm dlx husky-init && pnpm install 即可

   https://typicode.github.io/husky/
3. 修改 .husky/pre-commit 文件

问题：pnpm lint 是全量检查，耗时问题，历史问题。

暂存区 eslint 校验

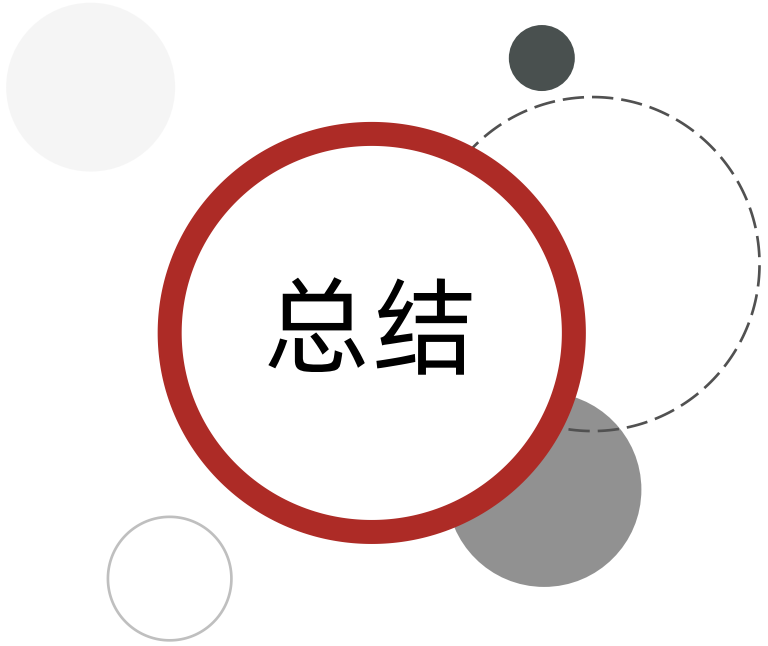
1. 安装 lint-staged 包 pnpm i lint-staged -D

2. package.json 配置 lint-staged 命令

3. .husky/pre-commit 文件修改

```
-pnpm lint
+pnpm lint-staged
```

```
4   "scripts": {
5     "dev": "vite",
6     "build": "run-p type-check build-only",
7     "preview": "vite preview --port 4173",
8     "build-only": "vite build",
9     "type-check": "vue-tsc --noEmit",
10    "lint": "eslint . --ext .vue,.js,.jsx,.
11    "prepare": "husky install",
12  1 "lint-staged": "lint-staged"
13  },
14  "lint-staged": { 2
15    "*.{js,ts,vue}": [
16      "eslint --fix"
17    ]
18  },
```



1. 如何在 git commit 前执行 eslint 检查？

   使用 husky 这个 git hooks 工具

2. 如何只检查暂存区代码？

   使用 lint-staged 工具

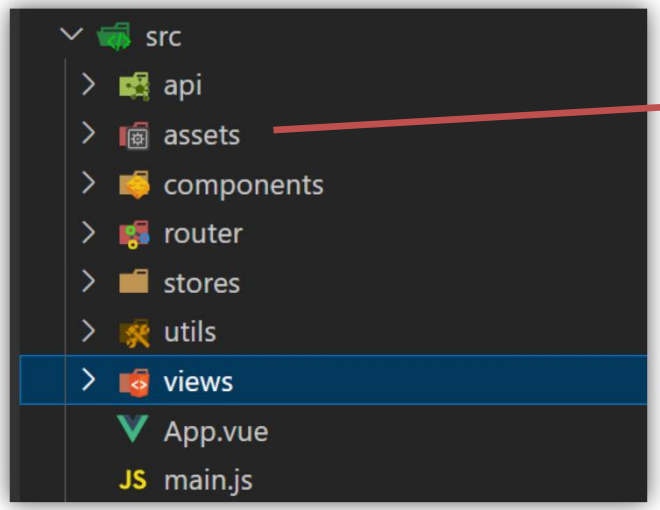
04 目录调整

目录调整

默认生成的目录结构不满足我们的开发需求，所以这里需要做一些自定义改动。

主要是以下工作：

1. 删除一些初始化的默认文件
2. 修改剩余代码内容
3. 新增调整我们需要的目录结构
4. 拷贝全局样式和图片，安装预处理器支持



安装 sass 预处理器

```
pnpm add sass -D
```

导入 main.js 作为全局样式

```
import  './assets/main.scss'
```

05 vue-router4 路由代码解析

路由初始化

```vue-router3.x
import VueRouter from 'vue-router'
// 初始化 vue-router3.x(Vue2)
const router = new VueRouter({
  mode: 'history',
  routes: [],
})

export default router
```

```vue-router4.x
import { createRouter, createWebHistory } from 'vue-router'
// 初始化 vue-router4.x(Vue3)
const router = createRouter({
  history: createWebHistory(import.meta.env.BASE_URL),
  routes: []
})

export default router
```

1. 创建路由实例由 createRouter 实现
2. 路由模式
    1. history 模式使用 createWebHistory()
    2. hash 模式使用 createWebHashHistory()
    3. 参数是基础路径，默认/

总结

```
import { createRouter, createWebHistory } from 'vue-router'

const router = createRouter({
  history: createWebHistory(import.meta.env.BASE_URL),
  routes: []
})

export default router
```

```
// 在 Vue3 CompositionAPI 中
// 1. 获取路由对象 router  useRouter
//    const router = useRouter()
// 2. 获取路由参数 route   useRoute
//    const route = useRoute()
import { useRoute, useRouter } from 'vue-router'
const router = useRouter()
const route = useRoute()

const goList = () => {
  router.push('/list')
  console.log(router, route)
}
```

创建一个路由实例，路由模式是history模式，路由的基础地址是 vite.config.js中的 base 配置的值，

默认是 / ，　　环境变量地址：https://cn.vitejs.dev/guide/env-and-mode.html

06 引入 Element Plus 组件库

按需引入 Element Plus

1. 安装： pnpm add element-plus
2. 配置按需导入：

   官方文档： https://element-plus.org/zh-CN/guide/quickstart.html
3. 直接使用组件
4. 彩蛋：默认 components 下的文件也会被 自动注册~

07 Pinia 构建仓库 和 持久化

Pinia 构建用户仓库 和 持久化

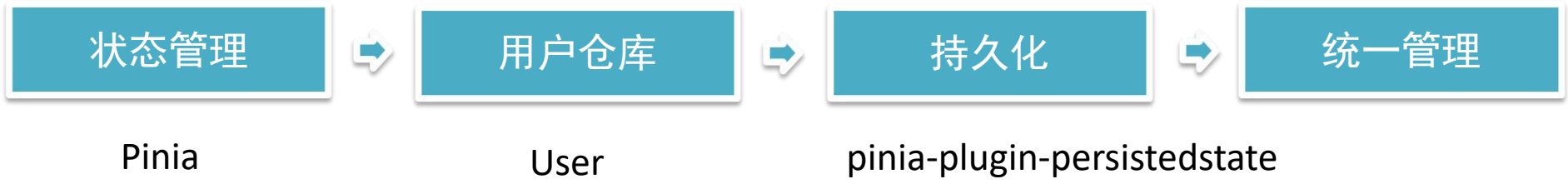
Pinia

User

pinia-plugin-persistedstate

08 Pinia 仓库统一管理

Pinia 仓库统一管理



pinia 独立维护

- 现在：初始化代码在 main.js 中，仓库代码在 stores 中，代码分散职能不单一
- 优化：由 stores 统一维护，在 stores/index.js 中完成 pinia 初始化，交付 main.js 使用

仓库 统一导出

- 现在：使用一个仓库 import { useUserStore } from `./stores/user.js` 不同仓库路径不一致
- 优化：由 stores/index.js 统一导出，导入路径统一 `./stores`，而且仓库维护在 stores/modules 中

09

数据交互 - 请求工具设计

数据交互 - 请求工具设计


axios配置
创建axios实例
基准地址，超时时间
请求拦截器
携带 token
响应拦截器
业务失败处理，摘取核心响应数据，401处理

10 整体路由设计

首页整体路由设计

path	文件	功能	组件名	路由级别
/login	views/login/LoginPage.vue	登录&注册	LoginPage	一级路由
/	views/layout/LayoutContainer.vue	布局架子	LayoutContainer	一级路由
├── /article/manage	views/article/ArticleManage.vue	文章管理	ArticleManage	二级路由
├── /article/channel	views/article/ArticleChannel.vue	频道管理	ArticleChannel	二级路由
├── /user/profile	views/user/UserProfile.vue	个人详情	UserProfile	二级路由
├── /user/avatar	views/user/UserAvatar.vue	更换头像	UserAvatar	二级路由
├── /user/password	views/user/UserPassword.vue	重置密码	UserPassword	二级路由

传智教育旗下高端IT教育品牌