Vue2Project DAY03

百慕大影城后台管理项目实践

1. 项目立项。

   项目背景：有一个小老板开了一家私人影院，最后越做越强，开成了连锁店。想要写一套系统管理电影院的运营。主要功能包括：普通用户通过**前台项目**来查看影院及上映的电影信息、电影院检索、选择影院、选座、买票等功能。管理员可以通过**后台管理项目**进行：影院管理、演职人员管理、放映厅管理、排片管理等。

2. 需求分析。
3. 原型设计。axureshop.com
4. 数据库设计。
5. 编码、测试。
6. 部署上线。

在本机搭建后端服务运行环境

1. 配置数据库。
2. 下载bmd_server.zip，找一个干净地方解压缩：/day03/demo下。

   进入解压缩后的目录中，通过pm2命令即可启动两个后端服务：

   index.js：数据接口服务。端口：3010

   uploadserver.js：上传文件服务。端口：9000

   1. 安装pm2进程管理工具：

      ```
      npm install -g  pm2
      ```

   2. 利用pm2的命令，启动这两个服务：

      ```
      # 执行这两个命令的时候需要当前目录在项目内部
      pm2  start  index.js
      pm2  start  uploadserver.js
      ```

      查看pm2的错误消息：

      ```
      pm2  logs
      ```

   3. 测试本地接口：

      ```
      http://localhost:3010/movie-actors?page=1&pagesize=100
      ```

4. 普及一些pm2的常用命令：

```
pm2 restart index  # 重新启动index服务
pm2 logs   # 查看日志信息
pm2 stop  index  # 停止index服务
pm2 status  # 查看服务状态
pm2 list    # 列出当前服务列表
```

接口相关

当前服务端提供了40多个接口，设计项目的各个模块：

```
https://docs.apipost.cn/preview/b8d0be111e068a7c/09ff5096269510aa
```

可以将接口导入到apipost中，方便测试接口。

在本机搭建前端脚手架环境

1. 新建项目：`bmdstudios-ms-client`，找一个干净目录：day03/demo

   ```
   vue  create bmdstudios-ms-client
   ```

   选择合适的配置，使用vue2.x生成脚手架项目包。

2. **进入项目目录中**，安装所需模块：

   ```
   # 通过cd命令，进入项目目录后安装
   # cd bmdstudios-ms-client
   npm i axios -S
   npm i element-ui -S
   ```

3. 在main.js中，配置ElementUI，引入所有组件：

   ```
   import ElementUI from 'element-ui';
   import 'element-ui/lib/theme-chalk/index.css';
   Vue.use(ElementUI);
   ```

4. 启动脚手架：

   ```
   npm run serve
   ```

功能实现

搭建项目的主体页面结构

1. 提供Header、Aside、Main部分结构。

   下载HomeView.vue、logo.svg。

   将HomeView.vue替换掉views/HomveView.vue。

   将logo.svg放入assets目录下即可。

2. 修改App.vue。

```
<template>
  <div id="app">
    <router-view/>
  </div>
</template>

<style lang="scss">
* {
  margin: 0;
  padding: 0;
}
</style>
```

为后台管理项目设计嵌套路由并实现

1. 明确页面需求，了解到底都有哪些页面，根据页面的结构，设计嵌套路由。

```
http://localhost:8080/user/login  没有顶栏与侧栏

http://localhost:8080/home/index        看到main部分显示Index
http://localhost:8080/home/actor-list   看到main部分显示演员列表
http://localhost:8080/home/actor-add    看到main部分显示新增演员表单
```

2. 准备所需要的组件：

```
src/views/Index.vue
src/views/actor/ActorList.vue
src/views/actor/ActorAdd.vue
```

3. 配置嵌套路由：children：

```
配置:  /                    重定向到 /home/index
配置:  /home/index
配置:  /home/actor-list
配置:  /home/actor-add
```

4. 在el-main中，添加二级路由占位符：`<router-view/>`

5. 测试即可。

```
http://localhost:8080/home/index        看到main部分显示Index
http://localhost:8080/home/actor-list   看到main部分显示演员列表
http://localhost:8080/home/actor-add    看到main部分显示新增演员表单
```

6. 实现点击左侧边栏菜单，实现跳转到相应页面。（刷新后默认选中是正确的）

在首页显示echarts图表

1. 在脚手架项目的根目录下，执行命令，安装echarts：

```
npm install echarts --save
```

2. 当某一个组件，需要显示图表时，需要提供一个容器div（设置好id、style）：

```
<div id="main"></div>
```

```
import * as echarts from 'echarts';

// 基于准备好的dom，初始化echarts实例
var myChart = echarts.init(document.getElementById('main'));
// 绘制图表
myChart.setOption({
  title: {
    text: 'ECharts 入门示例'
  },
  tooltip: {},
  xAxis: {
    data: ['衬衫', '羊毛衫', '雪纺衫', '裤子', '高跟鞋', '袜子']
  },
  yAxis: {},
  series: [
    {
      name: '销量',
      type: 'bar',
      data: [5, 20, 36, 10, 10, 20]
    }
  ]
});
```

实现演员列表页面相关业务

需求分析： 用户点击演员列表菜单项后，看到列表页，在页面中展示演员列表数据。提供面包屑导航，提供模糊查询的输入框，输入关键字后，可以模糊查询相关的演员信息。提供演员的删除操作。

实现思路：

1. 准备一个静态页面：ActorList.vue。提供基本的页面内容。

2. 当页面挂载完毕后，发送http请求，访问演员列表数据。将数据显示在页面中。
    1. mounted
    2. axios myaxios 接口
    3. Person.vue `<person name="" avatar=""></person>`
    4. v-for

实现演员列表页面模糊查询演员列表业务

1. 给查询按钮绑定事件。
2. 为搜索框做双向数据绑定。
3. 获取关键字后，发送post请求，加载演员列表。
4. 更新列表即可。

晚上想要把白天的代码再写一遍如何做?

1. 找一个干净目录，从git把我的代码下载下来。 `git clone` 。
2. 按照所需要的模块： `npm install` 。
3. 将git版本回滚到早上的版本： `git reset 版本hash值` 。

    ```
    # 在git项目的根目录下执行（.git文件夹所在的目录）
    git  reset  b43fd287581b6f5ef1195afa8b2afcfcc8cacb28  --hard
    ```

4. 在该基础之上往后写就行了。