1:什么是后台管理系统项目？

注意：前端领域当中，开发后台管理系统项目，并非是java、php等后台语言项目。

在前面课程当中，我们已经开发了一个项目【尚品汇电商平台项目】，这个项目主要针对的是用户（游客），可以让用户在平台当中购买产品。
但是你需要想明白一件事情，用户购买产品信息从何而来呀？

比如：前台项目当中的数据来源于卖家（公司），但是需要注意的时候，卖家它不会数据库操作。对于卖家而言，需要把产品的信息上传于服务器，写入数据库。
卖家并非程序员，不会数据库操作（增删改查）。导致卖家，找了一个程序员，开发一个产品，可以让我进行可视化操作数据库（增伤改查）

卖家（公司）：组成，老板、员工。
老板：开发这个项目，对于老板而言，什么都可以操作。【产品的上架、产品的下架、查看员工的个人业绩、其他等等】
员工：可能就是查看个人业绩

后台管理系统：可以让用户通过一个可视化工具，可以实现对于数据库进行增删改查的操作。
而且需要注意，根据不同的角色（老板、员工），看到的、操作内容是不同的。

对于后台管理系统项目，一般而言，是不需要注册的。

2:模板介绍
简洁版: https://github.com/PanJiaChen/vue-admin-template
加强版: https://github.com/PanJiaChen/vue-element-admin
模板的文件与文件夹认知【简洁版】

build
----index.js webpack配置文件【很少修改这个文件】
mock
----mock数据的文件夹【模拟一些假的数据mockjs实现的】，因为咱们实际开发的时候，利用的是真是接口

node_modules
------项目依赖的模块

public
------ico图标,静态页面，publick文件夹里面经常放置一些静态资源，而且在项目打包的时候webpack不会编译这个文件夹，原封不动的打包到dist文件夹里面

src
-----程序员源代码的地方
------api文件夹:涉及请求相关的
------assets文件夹：里面放置一些静态资源（一般共享的），放在aseets文件夹里面静态资源，在webpack打包的时候，会进行编译
------components文件夹：一般放置非路由组件获取全局组件
------icons这个文件夹的里面放置了一些svg矢量图
------layout文件夹：他里面放置一些组件与混入
------router文件夹：与路由相关的
-----store文件夹：一定是与vuex先关的
-----style文件夹：与样式先关的
------utils文件夹：request.js是axios二次封装文件
------views文件夹：里面放置的是路由组件

App.vue:根组件
main.js：入口文件
permission.js:与导航守卫先关、
settings：项目配置项文件
.env.development
.env.producation

后台管理系统API接口在线文档：
http://39.98.123.211:8170/swagger-ui.html
http://39.98.123.211:8216/swagger-ui.html

3:完成登录业务
-----静态组件完成
-----书写API（换成真实的接口）
-----axios二次封装
-----换成真实接口之后需要解决代理跨域问题(解决代理跨域问题)

4)退出登录业务

5)项目路由的搭建

6)完成品牌管理静态组件

7)完成品牌管理列表的展示
---书写相关的API接口

8)添加品牌与修改品牌的静态组件

9)完成添加品牌功能
----书写先关的API请求接口
----前台需要收集数据，给服务器提交数据（发请求）

10)完成品牌修改功能

11)表单验证功能
elementUI提供表单验证功能（自定义校验规则---重点）

12)删除品牌的操作
elementUI当中组件：有时间的时候多翻看看。

13)平台属性管理的三级联动静态静态组件

14)获取数据动态的展示三级联动

15)完成三级联动业务

16)获取平台属性数据与展示平台属性
属性:颜色
属性值:黑色，红色，粉色，紫色

```
{
    attrName:'颜色',
    attrValueList:['黑色'，紫色]
}
```

17）添加属性与修改属性静态组件

18)收集平台属性的操作

属性名称 属性值列表

属性：颜色
属性值：粉色、红色、蓝色等等

注意1：别再data当中收集三级分类的id
因为对象存储数据无序存储

19)收集平台属性值的操作

20)解决取消按钮数据回显问题

21)修改属性的操作

22)添加属性中的 查看模式与编辑模式 切换

查看模式：显示span
编辑模式：显示input
注意：通过flag标记进行切换查看模式与编辑模式，但是需要注意的时候，一个属性flag没有办法控制全部的属性值的切换

23)查看模式与编辑模式注意事项

24)修改属性中 查看模式与编辑模式操作

25)表单元素自动聚焦的实现

26)删除属性值操作

27)添加属性与修改属性保存的操作

28)完成按钮与三级联动可操作性

29)SPU模块介绍
SPU你可以理解为类

People类【SPU】
实例:【SKU】
小明:小明 18 男 等等
小红：小红 88 女 等等

30）完成SPU管理模块静态

31)完成SPU列表展示

32)完成SPU管理内容切换

----展示SPU列表结构
----添加SPU|修改SPU
----展示添加SKU结构

33)完成SpuForm静态

34)SpuForm业务的分析

---品牌的数据需要发请求的 http://localhost:9529/dev-api/admin/product/baseTrademark/getTrademarkList
---获取平台中全部的销售属性（3个） http://localhost:9529/dev-api/admin/product/baseSaleAttrList
---获取某一个SPU信息 Request URL: http://localhost:9529/dev-api/admin/product/getSpuById/5092
--获取SPU图片 http://localhost:9529/dev-api/admin/product/spuImageList/5092

----SPUFORM子组件发请求地方分析：
不能书写在mounted里面：
因为咱们刚才看了一下已经完成的项目，每一次显示SpuForm子组件的时候，都会发四个请求，
而我们为什么不能放在子组件的mounted里面，因为v-show只是控制SpuForm子组件显示与隐藏，
这个子组件并没有卸载（只是显示或者隐藏），导致mounted只能执行一次。

35)完成SpuForm获取服务器数据的操作

36)完成SpuForm数据的展示与收集

----添加SPU的时候需要给服务器携带的参数

```
{
  "category3Id": 0,
  "tmId": 0,
  "description": "string",
  "spuName": "string",

  "spuImageList": [
    {
      "id": 0,
      "imgName": "string",
      "imgUrl": "string",
      "spuId": 0
    }
  ],

  "spuSaleAttrList": [
    {
      "baseSaleAttrId": 0,
      "id": 0,
      "saleAttrName": "string",
      "spuId": 0,
      "spuSaleAttrValueList": [
        {
          "baseSaleAttrId": 0,
          "id": 0,
          "isChecked": "string",
          "saleAttrName": "string",
          "saleAttrValueName": "string",
          "spuId": 0
        }
      ]
    }
  ],
}
```

37)完成销售属性的展示

整个项目当中销售属性一共三个：颜色、尺码、版本

武侠SPU： 颜色

38)完成SpuForm照片墙图片的收集

----照片墙何时收集数据
---预览照片墙的时候，显示大的图片的时候，需要收集数据吗? ---不需要收集的【数据已经有了】

---照片墙在删除图片的时候，需要收集数据的。
---照片墙在添加图片的时候，需要收集数据的。

39)完成添加属性的操作

------收集哪些数据
baseSaleAttrId
saleAttrName
spuSaleAttrValueList
-----在什么时候收集数据

-----收集到哪里呀？
把数据收集到SPU对象

40)完成收集销售属性值的操作

41)完成销售属性值展示与收集

新增的销售属性值需要收集的字段:
baseSaleAttrId
saleAttrValueName

42)完成删除销售属性与销售属性值操作

43)完成SpuForm组件的保存的操作

44)完成添加Spu的业务
-----点击添加SPU按钮的时候，需要发请求（两个:获取品牌的数据、全部销售属性的数据）

45)完成删除SPU业务

46)完成添加SKU静态组件

47)获取添加SKU的数据
http://localhost:9529/dev-api/admin/product/spuImageList/5704
http://localhost:9529/dev-api/admin/product/spuSaleAttrList/5704
http://localhost:9529/dev-api/admin/product/attrInfoList/1/1/1

48)SkuForm数据的展示与收集


```
{
  "category3Id": 0,
  "createTime": "2021-10-14T00:41:56.934Z",
  "id": 0,
  "isSale": 0,
  "price": 0,
  "skuAttrValueList": [
    {
      "attrId": 0,
      "attrName": "string",
      "id": 0,
      "skuId": 0,
      "valueId": 0,
```

```
      "valueName": "string"
    }
  ],
  "skuDefaultImg": "string",
  "skuDesc": "string",
  "skuImageList": [
    {
      "id": 0,
      "imgName": "string",
      "imgUrl": "string",
      "isDefault": "string",
      "skuId": 0,
      "spuImgId": 0
    }
  ],
  "skuName": "string",
  "skuSaleAttrValueList": [
    {
      "id": 0,
      "saleAttrId": 0,
      "saleAttrName": "string",
      "saleAttrValueId": 0,
      "saleAttrValueName": "string",
      "skuId": 0,
      "spuId": 0
    }
  ],
  "spuId": 0,
  "tmId": 0,
  "weight": "string"
}
```


49)完成图片的展示与收集

50)完成SkuForm保存的操作

51)完成SKU列表的展示

52)完成查看SKU列表的loading效果

----loading效果目前只会展示一次
-----快速切换查看sku会发现上一次的数据会显示

58)完成SKU模块数据的展示

59)完成SKU的上架与下架操作

60)完成SKU查看详情业务

61)深度选择器

> 一般用于原生*CSS*

/deep/ 一般用于less

::v-deep 一般用户scss

62)数据可视化
就是服务器返回的数据，是以视图的形式进行展示【饼图、折线图，K线图】

echarts：vue、react
v-chart：vue
d3.js:vue、react
hightchart：vue、react
echarts：基本使用

63)echarts异步展示数据

1:初始化echarts实例的第二个参数可以设置主题颜色
2：echart如果想异步展示数据，当服务器的数据返回以后echarts实例需要再次调用setOptions方法重新设置配置对象，
将配置对象的数据替换为服务器的数据

64)echarts在Vue中使用

1:准备一个容器【宽度高度】
2：引入echarts核心库
3：初始化echarts实例，与初始化图表展示的数据

65)v-chart使用

66)权限管理的介绍

权限、角色等等业务逻辑

角色:一家企业而言：BOSS、运维、销售、程序员

权限:超级管理员（BOSS），是有权利操作整个项目的所有的模块
　硅谷321（新媒体），只能首页、商品管理者一部分菜单数据
admin：超级管理员-----boss

67:权限管理业务串讲

权限管理：用户管理、角色管理、菜单管理
由于用户管理、角色管理、菜单管理：对于获取数据、展示数据、收集数据相对而言，简单很多，因此进行相应的串讲。
把精力放到如何实现权限业务。

68:菜单权限的业务分析

超级管理:首页、权限模块、商品模块
硅谷321：首页
不同的用户、不同角色的任务，项目当中所能操作的、看见的菜单是不一样的。

如何实现菜单的权限？不同的用户所能操作|查看菜单不一样的？

起始不同的用户（角色），登录的时候会向服务器发请求，服务器会把用户相应的菜单的权限的信息，返回给我们
我们可以根据服务器返回的数据（信息），可以动态的设置路由，可以根据不同的用户展示不同的菜单。

菜单权限:当用户获取用户信息的时候，服务器会把相应的用户拥有菜单的权限信息返回，需要根据用户身份对比出，当前这个用户需要展示哪些菜单

69)完成菜单权限

当用户登录的时候，服务器端会返回相应角色的菜单权限的信息
只不过返回信息是一个数组routes-->['sku','spu',produt']

70)按钮权限

菜单权限：不同的用户（角色），能操作、能观看的菜单是不同的。

按钮的权限：不同的用户（角色），有的用户的是可见按钮、当然有的用户不可见。