# DAY01:

**今日进度：（5条）:**

1. 完成登录界面及交互

2. 完成后台首页组件

3. 完成修改密码组件

4. 完成账户列表组件的表格

5. 完成添加账户组件

**今日总结和问题（遇到了什么问题？如何解决的 3条）？：**

1. 登录界面中表单验证的实现遇到了困难

解决1：利用表单的validate方法，验证成功返回true，不成功返回false，

另外还使用了vue的选择器来选中需要验证的表单

<form ref=”loginForm”>

this.$refs.loginForm.validate(val=>{})

解决2：利用事件对象e，通过类名查找到小眼睛，点击切换眼睛

changeEye(e) {

if (e.target.className.includes("yanjing")) {

this.isOpen = !this.isOpen;

}

}

1. 后台首页遇到的问题是echarts的使用忘记了

解决：因为要操作dom，所有生成echarts的代码要写在mounted()中

1. 下载 yarn add echarts
2. 引入import echarts from "echarts"
3. 挂载Vue.prototype.$echarts = echarts
4. 创建一个有宽高的盒子，并设置ref属性，方便查找
5. 初始化echarts var myChart = this.$echarts.init(this.$refs.dataEcharts);
6. 去官网找图，复制 Var options={…}
7. myChart.setOption(option);
8. 封装组件遇到的小问题：当需要复用iconfont字体图标时，如何同时自定义字体图标的类型和颜色。

解决：类型使用动态类名控制，颜色用动态id控制，缺陷是，不能动态设置相同颜色的字体图标。

# DAY02:

**今日进度：（5条）:**

1. 完成头部面包屑和用户头像、信息
2. 实现基于element-ui的二次封装组件Panel.vue，并广泛使用于项目
3. 完成账户管理模块所有组件，并实现了表单自定义验证功能
4. 完成销售统计模块组件
5. 完成商品分类和商品添加组件

**今日总结和问题（遇到了什么问题？如何解决的）？：**

1. 表单自定义验证中的反向一致验证无效果

解决1：自定义验证函数参数写错了，导致无法进行正则验证

解决2: validateField 是对部分表单字段进行校验的方法，我将参数传错，传成了函数

1. 面包屑功能只有刷新和进入才正常切换

解决：watch: {

"$route.path"() {

this.calcBread();//忘记写this，导致函数无法正常调用

}

1. 设置表单宽度失败

解决：style=”width=200px”，头有点儿晕，写错了行内样式，应该是style=”width:200px”;

1. 表格中的switch开关无法相互独立切换

解决：问题未解决

# DAY03:

**今日进度：（5条）:**

1. 完成所有模块组件
2. 使用媒体查询和Flex布局优化首页卡片
3. 完成账号列表组件接口联调
4. 完成登录权限，保持登录状态和退出登录功能
5. 完成添加账号组件及账号列表联动

**今日总结和问题（遇到了什么问题？如何解决的）？：**

1. 登录组件中，点击登录没有触发http请求

问题：没有正确暴露登录ajax函数

/\* 登录请求 \*/

**export** **function** checkLogin(params) {

**return** req.post("/users/checkLogin", params)

}

再次强化ES的模块化

导出变量:

Export var firstName = “Jack”

Export var year = 1950

或者

Export {firstName, year}

导出函数：

Export function multiply(x,y){

Return x\*y

}

或者

function multiply(x,y){

Return x\*y

}

Export {multiply}

导入变量

Import {firstName, year} from “./profile”

导入函数

Import{multiply} from “xxx.js”

Case2:export default

//export-default.js

导出

Export default function(){

Console.log(“foo”)

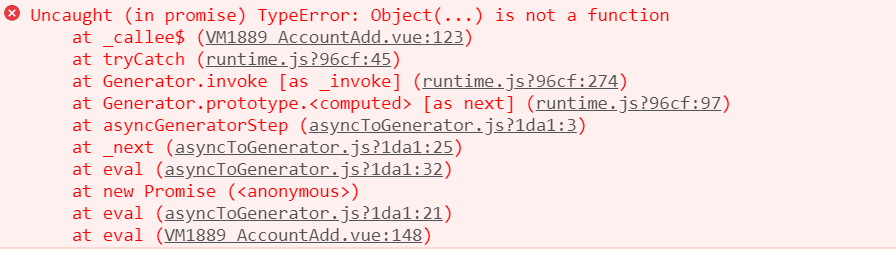
}//同上，这里也可以单独把函数提出去，只暴露函数名

导入

Import ‘任意变量名’ from ‘/export-default.js’

1. 调式获取用户列表接口时，把添加账号接口给改了，导致添加账号接口无法调用，报错。

正确的做法是在新加获取账号列表接口函数，粗心导致的。



1. 用户列表组件，批量删除操作无法获取到被选中的行

解决：这里运用了表格的change-selection事件，主要分清事件和方法

@selection-change="handleSelectionChange"

 // 当选择项发生变化时会触发该事件

    handleSelectionChange(rows) {

      this.ids = rows.map(v => v.id);

      console.log(this.ids);

    },

# DAY04:

**今日进度：（5条）:**

1. 完成个人中心用户数据接口和用户头像联调
2. 完成商品分类和添加商品分类功能模块
3. 实现商品分类和商品添加联调
4. 完成添加商品功能模块
5. 实现添加商品和商品列表功能及接口联调

**今日总结和问题（遇到了什么问题？如何解决的）？：**

1. 引入接口调用函数时，忘记加{}，导致程序报错
2. 在自定义验证中调用接口函数，写错参数，报错，应该直接使用回调函数参数
3. 调用函数没写this，报错
4. 图片渲染:src=”imgBaseUrl+imgUrl”加号前后写反
5. 动态属性忘记加冒号。

# DAY05:

**今日进度：（5条）:**

1. 添加了商品列表组件的查询功能，优化编辑和删除体验
2. 完成订单列表组件
3. 完成订单变量组件与订单列表联动
4. 完成店铺管理组件
5. 优化个人中心头像预览与更新功能

**今日总结和问题（遇到了什么问题？如何解决的）？：**

1. 获取订单详情失败

原因：请求接口的参数直接传的id，应该是{id}才对

1. 头像或单张图片上传/更新问题
2. 头像/图片上传成功回调返回的imgUrl为xxxx.jpg，图片直接回填到xxxx.imgUrl
3. 修改头像/图片请求参数的imgUrl为xxxx.jpg，上传时，直接使用xxxx.imgUrl
4. 显示头像/图片的方式为，显示上传需要设置src为imgBaseUrl+xxxx.imgUrl

 <img v-if="xxxx.imgUrl" :src="imgBaseU**rl+xxxx.imgUrl" class="avatar" />**

**说明：**

**如果响应的图片数据为xxx.jpg，那么上传回填就用xxx.jpg**

**如果响应的图片数据为**[**http://......jpg**](http://......jpg)**，那么上传回填就用imgBaseUrl+xxx.jpg**

1. 店铺管理的照片墙问题
2. 获取店铺信息照片墙数据格式:

pics:[xxx.jpg,xxx.jpg,xxx.jpg]

1. 照片墙显示方式:

:file-list//属性值为[{name: 'food.jpg', url: 'https://xxx.cdn.com/xxx.jpg'}]

因此在1)中获取到pics，需要进行数据处理

**let** { data } = **await** getShopInfo();

    this.shopForm = data;

    this.shopForm.pics = this.shopForm.pics.map(v **=>** ({

      name: v,

      url: this.imgBaseUrl + v

    }));

1. 上传照片到照片墙===>对接修改店铺信息的照片墙请求参数

handleImgSuccess(res) {

      let { code, msg, imgUrl } = res;

      if (code === 0) {

        this.$message({ type: "success", message: msg });

        this.shopForm.pics.push({

          name: imgUrl,

          url: this.imgBaseUrl + imgUrl

        });

      }

    }

1. 从照片墙删除照片===>对接修改店铺信息的照片墙请求参数

handleRemove(file) {

      this.shopForm.pics.forEach((v, i) => {

        if (v.uid === file.uid) {

          this.shopForm.pics.splice(i, 1);

        }

      });

    },

1. 发送修改店铺信息请求时，还需要将pics进行处理

因为当前pics的格式为{name:”xxx.jpg”,url:<http://.....>,uid:"153365”}

**let** arr = this.shopForm.pics.map(v **=>** v.name);

        pics: JSON.stringify(arr)

注意：请求参数如果时数组，还需要用JSON转成字符串，才能成功发送请求。

# DAY06:

**今日进度：（5条）:**

**今日总结和问题（遇到了什么问题？如何解决的）？：**

# DAY07:

**今日进度：（5条）:**

**今日总结和问题（遇到了什么问题？如何解决的）？：**