**一．用户退出**

**在登录时已经把用户的信息保存到本地了，所以点击退出时判断是不是确认退出，如果确认退出，把本地存储删除，window.localStarge.removeItem('保存的信息名称'),然后路径跳转到登录界面， this.$router.push({ name: 'login' })，点击的时候给click加入一个.native属性，.native是事件修饰符，将原始的html页面注册到组件的根元素，**

**二.用async封装axios，和try{}catch(){}配合使用**

**任何函数都可以标记为async函数，async函数中可以使用await关键字，**

**await后面要跟一个promise才有意义**

**await要等待promise中的resolve调用了才可以往后执行，**

**如果resolve有参数，可以再await前面定义一个变量或常量接收**

**async函数的执行顺序**

**函数中的await只是在函数内部等待，不会阻塞函数之后的代码执行**

**Async函数的返回值**

**Async如果标记的是一个普通函数，内部没有跟await配合使用，那么会被包装为一个立即被resolve的promise对象，return的值就是resolve（）里面的参数**

**如果是promise对象，则不做任何处理，可以在另一个async函数中用await接收它的结果**

**try-catch和async没什么关系，只是能捕获async中的异常，不是es6的语法，是js自带的**

**三.异常处理**

**try-catch是js中的异常捕获代码，我们可以把有可能出错的代码放到try中，如果try出错，停止try中后续代码执行，进入catch中，catch默认接收一个参数：是错误的信息数据**

**try-catch默认只能捕获同步代码异常，不能捕获普通的异步函数调用的错误，也不能捕获promise中的reject异常，这几种异常都捕获不到，但是可以捕获async函数中（await promise对象）中的异常，**

**try-catch代码的执行顺序是从上到下，先进入try，如果try中代码没有错误，就继续往后执行，如果try中代码有错误，就进入catchcatch执行完毕，继续往后执行。**

**手动抛出异常，throw new Error（），相当于new Promise((reslove,reject) => {reject('错误信息')})**

**四.封装auth模块，**

**按需导出，按需导入，把设置，调用，和移除用户信息封装起来，用export导出，看哪个位置需要，就调用，**

**导出： export XXX 导入： import { 名字 } from “模块”**

**五． 配置axios的基础路径和简写方法**

**在入口模块中导入axios，从axios github上找到设置基础路径的代码，**

**将axios配置到vue的prototype中，因为组件都是vue实例，所以vue的原型对象都可以运用，但在命名中尽量加一个$符号，因为有可能跟组件中的数据命名冲突****。**