华南理工大学

《网络应用开发》课程实验报告

实验题目： 设计并实现一个电子商务网站的开发和在线部署 \_\_

姓名： 陈映松 学号： 202130442563

班级： 21计科2班 \_

提交日期： 2023-12-19

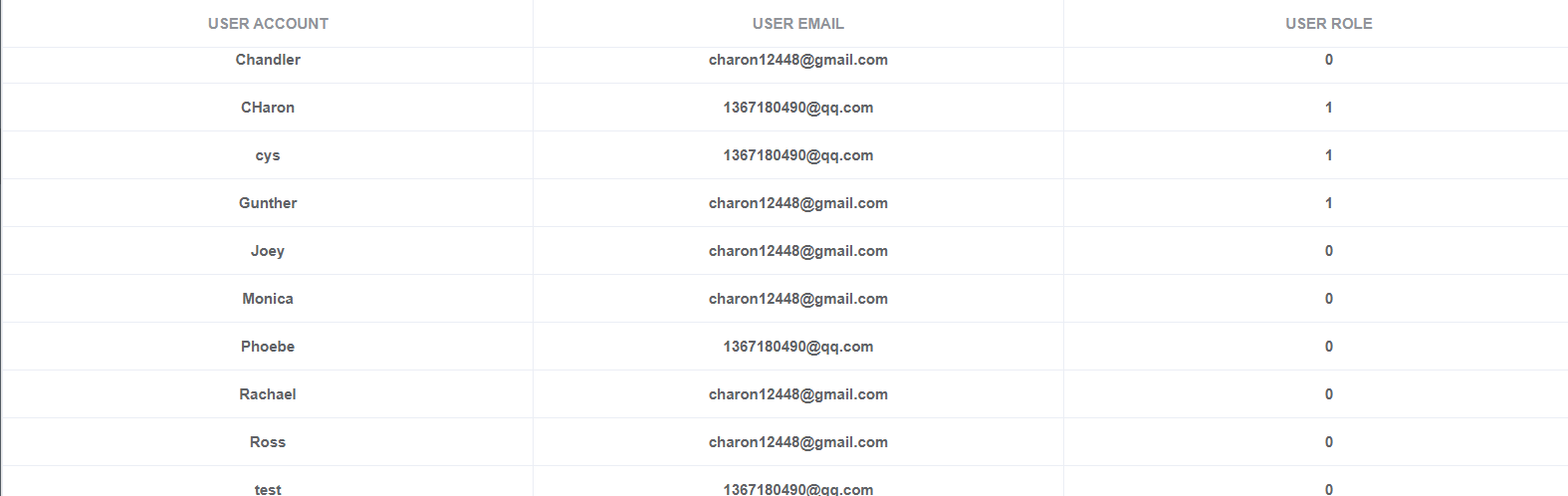
代码托管地址：<https://github.com/CHaron1244/web_application_development>

部署在线网站： 服务器ip地址: 114.132.219.217

网站部署超链接地址: <http://114.132.219.217/>

测试账号口令：测试管理员账号：CHaron 密码：123

测试顾客账号：Monica 密码：123

注：可供测试的账号如下图（所有账号密码都是123，role为0表示顾客，为1表示管理员）：

|  |
| --- |
| **实验概述** |
| 【实验题目】  设计并实现一个电子商务网站的开发和在线部署.  【基本功能要求】  顾客：  用户的注册、登录、注销  购买流程（浏览/查询->添加至购物车->付款->发送电子邮件确认收货）  销售：  商品目录的管理（包括最基本的增删改等操作）  后台销售统计报表，销售状态  客户的浏览/购买日志记录  【实验环境和工具】  Windows 11  Mysql 8.0  Nginx 1.24.0  Apache Tomcat/10.1.15  Jdk 17.0.8+9-LTS-211 |
| **实验内容**   1. 系统设计：   给出系统整体架构，并分别介绍各个模块。  1.1数据库设计  根据任务要求我设计了6张表来管理，表结构如下展示： 用户 User  | **属性名称** | **属性类型** | **is null** | **primary key** | | --- | --- | --- | --- | | user\_account | varchar(30) | no | yes | | user\_password | varchar(30) | no | no | | user\_email | varchar(30) | no | no | | user\_role(用户角色-customer0，admin-1) | int | no | no |  商品 Good  | **属性名称** | **属性类型** | **is null** | **primary key** | | --- | --- | --- | --- | | good\_name | varchar(30) | no | yes | | good\_price | double | no | no | | good\_img | varchar(100) | no | no | | good\_sale | int | no | no | | good\_info | varchar(100) | no | no | | good\_type | varchar(30) | no | no |  购物车Cart  | **属性名称** | **属性类型** | **is null** | **primary key** | | --- | --- | --- | --- | | user\_account(关联外键) | varchar(30) | no | no | | good\_name(关联外键) | varchar(30) | no | no | | good\_price | double | no | no | | good\_number | int | no | no |  订单 Bill  | **属性名称** | **属性类型** | **is null** | **primary key** | | --- | --- | --- | --- | | user\_account(关联外键) | varchar(30) | no | no | | good\_name(关联外键) | varchar(30) | no | no | | good\_price | double | no | no | | order\_time | varchar(30) | no | no |  顾客购买日志 BuyLog  | **属性名称** | **属性类型** | **is null** | **primary key** | | --- | --- | --- | --- | | user\_account(关联外键) | varchar(30) | no | no | | good\_name(关联外键) | varchar(30) | no | no | | good\_number | int | no | no | | buy\_time | varchar(30) | no | no |  顾客浏览日志 BrowseLog  | **属性名称** | **属性类型** | **is null** | **prmairy key** | | --- | --- | --- | --- | | user\_account(关联外键) | varchar(30) | no | no | | good\_name(关联外键) | varchar(30) | no | no | | browse\_time | varchar(30) | no | no |   1.2 网站文件架构  网站文件的基本架构如下：    1.3 网站网页架构    1.4 基础功能实现  对于顾客要求能够完成注册，登录，注销，浏览商品，添加商品至购物车，购买商品，支付账单，给顾客发送电子邮件等功能。  对于管理员要求能够完成管理商品（包括基本的增加商品，删除商品，修改商品信息等），  统计商品销售数据，查看顾客浏览或者购买商品的日志记录。  对于这些功能，我采用了vue+springboot框架来设计网站。基本流程如下：  一．网页点击按钮触发事件，发送请求  二．axios代理前端发送指定请求到后端springboot工程指定的controller  三．controller接受前端发送的请求，进行处理打包后转发给对应的service  四．service接受controller转发的请求，并转发给对应的mapper  五．Mapper生成相应的mysql语句与数据库进行交互并返回查询结果  六．vue前端接收到后端结果后渲染展示相关信息。  下面以注册一位新的用户功能为例讲解这几个步骤：  一．网页点击注册按钮触发注册事件，前端发送注册请求    二．axios代理前端发送指定请求到后端springboot工程指定的controller    Register.vue文件中vue前端的register代码段如上。可以看到vue先对表单内容进行valid验证，通过验证之后axios就会携带信息代理发送请求到/user/register地址，后端处理后返回数据res。如果后端返回结果的标识码（res.data.code）为0，说明此用户  已经成功注册，否则未能成功注册。  三．controller接受前端发送的请求，进行处理打包后转发给对应的service  userController.java文件中controller段代码如下：    首先进行的for循环是为了确保没有重复的用户（保证用户账号唯一），若已经存在相同的用户账号则禁止注册并返回错误提示信息。倒数第二行userservice.addUser(user)则是转发请求给userservice进行处理。如果能够成功执行则返回成功信息（Result.success（））。  四．service接受controller转发的请求，并转发给对应的mapper  userService.java文件中处理register代码段如下：    这里函数中service只是简单再次转发了请求到usermapper进行添加用户。  五．mapper生成相应的mysql语句与数据库进行交互并返回查询结果  userMapper.java文件中关于添加用户信息代码段如下：    userMapper接口定义的方法可以直接调用mysql语句执行，这里是将前端传递过来进行整合成用户类对象的参数user进行添加。  六．vue前端接收到后端结果后渲染展示相关信息  当后端成功添加一位用户时，前端会有弹窗提示，告知用户已经注册成功，并且还温馨提示可以点击下方的按钮回到登录界面进行登录，界面如下：  先填好相应的信息：    然后点击注册按钮，界面上方会显示弹窗如下：    之后测试登录可以成功登录，说明用户注册成功。     1. 代码实现   具体的代码实现过程介绍，并给出关键代码（不要忘记注释）；  **给出github或gitee代码地址**，不要忘记在“readme.md”之中著名自己的学号和姓名，以及代码文件的简单说明。  显然，项目代码量过于庞大，想要把每一个地方的构思都展示在报告中未免不现实，故在此我只选取有代表性思想的地方进行代码讲解与展示，在源代码中也配有相应的注释，可以参考来看。  2.1代码实现构思  此部分主要讲解我是如果构思整个项目的布局以及一些小的细节思考。  vue项目采用了单网页架构，整个项目中只有一个index.html作为页面进行页面展示，而所有的内容都是以不同的url作为信息来源渲染数据到屏幕上。为了美观与工整，我想到了给所有页面添加固定的头部与侧边栏，这些部分也可以作为导航按钮前往不同的页面，同时我也整合了element-ui库来布局组件，让组件更加美观。例如我就在头部右侧名字的位置设置了一个下拉菜单（鼠标只要悬浮在名字上就会触发下拉，不必点击），里面可以选择退出登录或者展示个人信息。这样设计既保留了美感，又把所有的功能都一并实现，可谓一举两得。  springboot工程我采用了传统的controller—service—mapper三层结构，这种三层结构的设计模式有助于代码的模块化、可维护性和可扩展性。它将不同层次的责任清晰分离，使得代码更易于管理和测试。控制器负责处理请求和响应，服务层处理业务逻辑，而数据访问层则负责与数据库或其他数据源交互。这种结构有助于降低耦合度，提高代码的可读性和可维护性。  然后是一些关键性代码的展示，这里我就以比较关键的顾客界面的实现把前后端代码进行演示。  前端页面Customer.vue代码如下：<template>    <div>      <div style="margin:10px 0">        <el-input v-model="search" placeholder="Please enter the keyword to find what you like!"          style="width: 50%;border-bottom-color: black" clearable></el-input>        <el-button type="primary" style="margin-left:  5px ;          background-color: black; color: white;border-color: white;" @click="load">search</el-button>      </div>      <!-- 在el-table上加上height属性就可以固定表头，先用着再看要不要不需要这个功能 -->      <el-table :data="readTableData" style="width: 100%;font-size: 15px;font-weight: bold" border height="1000">        <el-table-column prop="goodName" label="GOOD NAME" align="center"></el-table-column>        <el-table-column prop="goodPrice" label="GOOD PRICE" align="center">          <template slot-scope="scope">            $&nbsp;{{ scope.row.goodPrice }}          </template>        </el-table-column>        <el-table-column prop="" label="GOOD PICTURE" align="center">          <template slot-scope="scope">            <!-- <img src="../../public/imgs/Americano.png" alt="hello" width="100%"></img> TODO,图片路径未处理好 -->            <img :src="require(`@/assets/imgs/${scope.row.goodName}.png`)" alt="hello" width="100%"></img>          </template>        </el-table-column>        <el-table-column prop="goodSale" label="GOOD SALE" align="center"></el-table-column>        <el-table-column prop="goodInfo" label="GOOD INFORMATION" align="center"></el-table-column>        </el-table-column>        <el-table-column prop="goodType" label="GOOD TYPE" align="center"></el-table-column>        <el-table-column label="OPERATION" align="center">          <template slot-scope="scope">            <el-button size="mini" type="primary" @click="showAddToCartDialog(scope.$index, scope.row)">Add to              Cart</el-button>            <el-dialog title="Now Add this refreshment to your cart!" :visible="addToCartDialogVisible"              @close="addToCartDialogVisible = false">              <el-form :model="addToCartForm" ref="addToCartForm" label-width="80px">                <el-form-item label="Quantity">                  <el-input v-model="addToCartForm.quantity" type="number" autocomplete="off"></el-input>                </el-form-item>              </el-form>              <div slot="footer" class="dialog-footer">                <el-button @click="addToCartDialogVisible = false">Cancel</el-button>                <el-button type="primary" @click="AddToCart">Confirm</el-button>              </div>            </el-dialog>          </template>        </el-table-column>        <el-table-column label="FURTHER" align="center">          <template slot-scope="scope">            <el-button size="mini" type="info" @click="generateBrowseLog(scope.$index, scope.row)">WANT TO KNOW              MORE?</el-button>            <el-dialog title="NOTICE" :visible.sync="dialogVisible" width="40%">              <span>We will notify you if there's more information, and this will produce a browse log. Good luck!</span>              <span slot="footer" class="dialog-footer">                <!-- <el-button @click="dialogVisible = false">Cancel</el-button> -->                <el-button type="primary" @click="dialogVisible = false">Confirm</el-button>              </span>            </el-dialog>          </template>        </el-table-column>      </el-table>    </div>  </template>  <script>    import axios from 'axios'    import dayjs from 'dayjs'    export default {      name: 'Customer',      data() {        return {          dialogVisible: false,          index: 0,          row: {},          addToCartDialogVisible: false, // 弹出框默认不出现          addToCartForm: {            quantity: 1, // 默认数量为1          },          dialogFormVisible: false,          form: {},          tableData: [], // 所有商品信息          search: '',          user: {},          currentTime: ''        };      },      created() {        let str = sessionStorage.getItem("user"); // str is a JSON object        this.user = JSON.parse(str); // user is a js object (this.user.userAccount)访问        this.load(); // 加载所有商品信息      },      methods: {        showAddToCartDialog(index, row) {          this.addToCartDialogVisible = true          this.index = index          this.row = row        },        load() {          axios.get("/good/selectAllGoods").then((res) => {            this.tableData = res.data.data;          })        },        AddToCart() { // 自己想的方法：用data里面的index row来存点击按钮返回过来的index row          const role = this.user.userRole          if (role == '0') { // 规定只有customer才能购物，管理员只能管理商品            // let str = sessionStorage.getItem('user') // str is a JSON object            const userAccount = this.user.userAccount            const goodName = this.row.goodName            const goodPrice = this.row.goodPrice            const goodNumber = this.addToCartForm.quantity            const form = {              userAccount,              goodName,              goodPrice,              goodNumber            }            axios.post('/cart/addItem', form).then(res => {              const ans = res.data              if (ans.code == '0') {                this.$message({                  message: `Successfully add your commodity to your cart!`,                  type: 'success'                })              } else {                this.$message({                  type: "error",                  message: 'Sorry,fail to add this item failed!!'                })              }            }).catch(error => {              this.$message({                type: "error",                message: 'Sorry,fail to add this item!!'              })            })            this.load() // 数据更新就要加载            this.addToCartForm.quantity = 1 // 重置为1每次            this.addToCartDialogVisible = false          }        },        handleDelete(index, row) {// console.log(index, row); index是这一行数据索引(from 0)，row是这一行数据，可以直接获取元素          const goodName = row.goodName          axios.post(`/good/deleteGoodByName/${goodName}`).then(res => {            if (res.data.code == '0') {              this.$message({                message: `Successfully delete ${goodName}`,                type: 'success'              })            } else {              this.$message({                message: 'Something wrong happened!',                type: 'error'              })            }          })        },        generateBrowseLog(index, row) {          const account = this.user.userAccount          const goodName = row.goodName          this.currentTime = dayjs().format("YYYY-MM-DD HH:mm:ss")          const form = {            userAccount: account,            goodName: goodName,            browseTime: this.currentTime          }          axios.post('/browseLog/addBrowseLog', form).then(res => {            if (res.data.code == '0') {              this.$message({                type: 'success',                message: 'Successfully add this browse log!'              })            } else {              this.$message({                type: 'error',                message: 'Fail to add this browse log!'              })            }          }).catch(error => {            this.$message({              type: 'error',              message: 'Oops! Something wrong happened!'            })          })          setTimeout(() => {            this.dialogVisible = true // 提醒用户会产生browse log          }, 1000)        }      },      watch: {        tableData() {          deep: true,            this.load()        }      },      computed: {        readTableData() {          return this.tableData.filter(item => {            if (this.search === '')              return item.goodName.toLowerCase().includes('');            else              return item.goodName.toLowerCase().includes(this.search.toLowerCase());          });        }      }    };  </script>  <style scoped>    .el-row {      margin-bottom: 20px;    }    .el-col {      border-radius: 4px;    }    .bottom {      margin-top: 13px;      line-height: 12px;      display: flex;      justify-content: space-between;      align-items: center;    }    .button {      padding: 0;      min-height: auto;    }    .image {      width: 100%;      display: block;    }  </style>  可以看到customer的template的div标签中主要就2个div标签，一个用来维护最上方搜索框，一个用来维护商品信息表。搜索框也配有搜索按钮（虽然输入文本时已经会有及时相应数据展示不同信息了），搜索框输入文本绑定遍历search。下面的表格我借助了element-ui的el-table结构进行实现，展示的数据为readTableData（不是完全的商品数据是因为需要根据上方搜索框内文本继进行数据展示的筛选），readTableData与tableData的关系在computed的函数中有写到。然后就是一列一列数据绑定到商品名字，价格之类的信息。script段的create函数钩子在网页创建时从sessionStorage中获取当前登录的user信息并保存在this.user中。methods方法里面的load函数从后端获取所有商品信息，AddToCart绑定添加商品的按钮，可以添加一件商品到购物车中。generateBrowseLog向后端写入一条浏览日志记录，其他函数比较简单，在此不进行说明。  再来看看springboot后端程序：  首先是user用户实体类和商品good实体类（这两个都用到了）：  这里的注释是springboot工程自动补全setter，getter，构造函数的简写形式。  然后看下添加购物车AddToCart绑定的后端处理代码：  可以看到后端处理过程并不复杂（主要要求实现的功能不是很难，每一层没有做过多额外的工作，基本都是简单得向下一层转发数据即可）。函数绑定一个RequestMapping注解来指定当访问到url后缀为此时调用addItem函数，controller直接转发请求给相应的service，service也直接转发给相应的mapper，mapper就直接调用相应的数据库语句执行即可。  2.2 代码功能说明  下面分别介绍比较重要的文件或文件夹的作用（个别文件/文件夹的作用已在第一部分导图中标注）：  **前端：**  .vscode文件夹：概要存储项目特定的 VSCode 配置  dist文件夹：用于生产环境部署的最终版本  node\_modules文件夹：存放当前项目所依赖的各种第三方库，模块和包  public/imgs 和 src/assets/imgs文件夹：存放商品图片  public/static/config.js: 指定文件上传的 URL 地址被指定为当前主机的本地地址  public/favicon.ico：vue图标  public/index.html:vue.js 单页应用的主 HTML 文件，负责引入 Vue.js 库、打包后的应用程序文件以及定义应用的基本结构  src/assets/css/global.css: 设置所有 HTML 元素的默认样式  src/components文件夹：存放vue基本布局（头部和侧边栏）  src/components/Aside.vue：设置界面侧边栏布局  src/components/Header.vue：设置界面头部布局  src/layout文件夹：运用components里面组件并且展示组件  src/layout/layout.vue：运用2个布局组件进行格式布局  src/router/index.js：存放路由配置信息  src/store/index.js：存放 vuex 应用状态管理模式的相关文件  src/views文件夹：存放各种视图组件  src/views/AllPersonInfo.vue：存放并展示所有人员信息的组件  src/views/Bill.vue：存放并展示所有账单的组件  src/views/BrowseLog.vue：存放并展示所有浏览日志记录的组件  src/views/BuyLog.vue：：存放并展示所有购买日志的组件  src/views/Cart.vue：存放并展示所有购物车商品信息的组件  src/views/Customer.vue：顾客界面组件  src/views/Good.vue：存放并展示所有商品组件（管理员专用）  src/views/Login.vue：登录界面组件  src/views/Person.vue：存放并展示个人信息的组件  src/views/Register.vue：注册组件  src/.env.production：定义生产环境下的特定配置和变量  src/App.vue：Vue.js 应用程序的根组件  src/main.js：Vue.js 应用的入口文件，通常用于创建 Vue 实例，并且在其中引入和配置一些全局的插件、组件、路由等  其余就是一些vue项目的配置文件......  **后端：**  ├─main  │ ├─java  │ │ └─com  │ │ └─friend  │ │ │ PerkApplication.java （springboot应用程序的入口）  │ │ │  │ │ ├─config  │ │ │ AliPayConfig.java （阿里云支付的配置文件）  │ │ │ AppConfig.java （创建一个RestTemplate的实例）  │ │ │ CorsConfig.java （配置跨域资源共享（CORS））  │ │ │ Result.java （封装后端传递结果）  │ │ │  │ │ ├─controller  │ │ │ AlipayController.java （阿里云支付控制文件）  │ │ │ billController.java （账单支付控制文件）  │ │ │ browseLogController.java （浏览日志记录控制文件）  │ │ │ buyLogController.java （购买日志记录控制文件）  │ │ │ cartController.java （购物车控制文件）  │ │ │ emailController.java （电子邮箱发送控制文件）  │ │ │ goodController.java （商品管理控制文件）  │ │ │ userController.java （用户管理控制文件）  │ │ │  │ │ ├─entity  │ │ │ AlipayBean.java （阿里云支付实体）  │ │ │ Bill.java （账单实体）  │ │ │ BrowseLog.java （浏览日志记录实体）  │ │ │ BuyLog.java （购买日志记录实体）  │ │ │ Cart.java （购物车实体）  │ │ │ Good.java （商品实体）  │ │ │ User.java （用户实体）  │ │ │  │ │ ├─mapper  │ │ │ billMapper.java （账单数据库操作接口）  │ │ │ browseLogMapper.java （浏览日志记录数据库操作接口）  │ │ │ buyLogMapper.java （购买日志记录数据库操作接口）  │ │ │ cartMapper.java （购物车数据库操作接口）  │ │ │ goodMapper.java （商品数据库操作接口）  │ │ │ userMapper.java （用户数据库操作接口）  │ │ │  │ │ └─service  │ │ billService.java （账单调用数据访问层文件）  │ │ browseLogService.java （浏览日志调用数据访问层文件）  │ │ buyLogService.java （购买日志数据访问层文件）  │ │ cartService.java （购物车调用数据访问层文件）  │ │ goodService.java （商品调用数据访问层文件）  │ │ userService.java （用户调用数据访问层文件）  │ │  │ └─resources  │ application.yml （springboot 项目中用于配置应用程序属性的文件）  │  └─test  └─java  └─com  └─friend  PerkApplicationTests.java  target文件夹：maven 构建过程的输出目录  pom.xml：用于定义和配置 maven 项目的元数据和构建配置。它包含了项目的依赖关系、插件、构建配置等信息  2.3 代码开源  2.3.1在GitHub上注册一个账号用于管理代码    2.3.2 上传相应的代码以及说明  在此给出github仓库地址：<https://github.com/CHaron1244/web_application_development>     1. 功能测试：   给出测试方法以及测试过程的详细说明；  给出测试结果的说明（例如截图）。  3.1登录与注册  在浏览器输入<http://114.132.219.217/>，自动重定向到/login登录界面，如下：    账号密码为空时提示用户输入密码（红字）    账号或者密码错误时弹窗提醒登录失败    登录login成功进行弹窗并且跳转顾客界面customer    3.2 顾客基本界面（选购商品）  顾客界面主要内容展示    右侧有滚动条，滚动可以浏览更多的商品    最上面的搜索框可以根据商品名称进行筛选想要的商品信息（下面是四组测试）          点击want按钮弹出提示用户此条浏览记录到浏览记录日志中    点击添加商品按钮，弹出提示用户添加多少件商品    商品添加到购物车成功弹出提示    此时看到购物车界面有刚刚添加的用户和商品信息    3.3 管理员基本界面（管理商品）  管理员进入商品管理界面弹窗提示成功进入  鼠标悬浮在哪一行，哪一行背景就会变灰  上方搜索框可以根据商品名称进行搜索显示信息  右侧配有滚动条可以滚动浏览所有商品的信息  点击每一栏商品右侧的编辑按钮，弹窗提示用户输入想要修改的内容  这是我修改Americano商品的信息，可以根据type栏判断是否修改成功  注意到Price一栏右侧有箭头，规定只能输入数字  点击确认按钮可以看到修改过的Americano商品信息已经更新成我们输入的信息（销售量不会改变）  点击每一栏商品右侧的删除按钮，弹窗提示用户是否想要删除这件商品  点击取消按钮弹窗提示取消删除这件商品  点击确认按钮可以看到删除的Iced Coffee信息已经不出现在商品列表中了  点击上方添加商品按钮，弹窗提示用户输入想要添加商品的信息  这是我填写的添加商品的信息  点击确认按钮弹窗提示成功添加一件商品  可以看到我们添加的商品已经超过展示在界面上了  3.4 权限测试（顾客和管理员各自有无法进入的界面）  管理员无法进入顾客界面    顾客无法进入2个log界面，弹窗提示    顾客就算点击log，good界面，浏览器url也不改变，依然是顾客界面    顾客无法进入good界面，弹窗提示    顾客无法进行ALLPERSONINFO界面，弹窗提示    3.5 头部右侧下拉菜单退出登录  点击退出登录弹窗提示用户并且等待2秒才跳转到登录界面    3.6 头部右侧下拉菜单展示个人信息  头部右侧显示顾客账号，鼠标悬浮有下拉菜单可以推出登录或者展示个人信息    查看个人信息跳转个人页面person    edit按钮开启编辑，close edit按钮关闭编辑    点击save按钮弹窗提示是否保存修改过的个人信息    点击注销cancel按钮弹窗提示用户是否注销    取消修改个人信息，弹窗提示用户取消修改    取消注销弹窗提示用户注销失败    修改个人信息成功之后弹窗提示用户修改成功    注销成功弹窗弹窗提示用户成功注销并欢迎下次再来（包含姓名）    注销之后再次登录已经不存在此用户    3.7 所有人员信息展示  管理员点击展示所有人信息的按钮弹窗提示成功进入并显示所有人除了密码以外的信息    上方搜索框可以根据用户账号内容展现相关信息，例如输入r，账号中包含r的用户信息才会展示    3.8 购物车  购物车界面cart  点击添加到购物车按钮，弹窗提示是否添加到购物车  取消添加到购物车弹出提示取消  点击确认按钮弹窗提示成功添加  看一下邮箱是否有收到相应的邮件，确实可以（这里收件人跟上面图片不一样，因为是不同的测试组）    购买日志中也能看到刚刚（11月18日）的操作已经被成功记录  承接支付宝付款，此时Latte销量为0  再次来到商品管理页面good，可以看到Latte的销量已经变成2了  3.9支付宝支付功能  购物车中这里有一条CHaron的商品记录    点击支付按钮等待5-10秒跳转到支付界面，输入账号和密码（默认111111）    再等待5-10秒进入输入密码界面（默认111111）    订单正在处理显示界面    提示用户成功付款界面    3.10 订单  订单界面弹窗提示进行此界面    上方搜索框可以根据用户账号搜索对应的日志，此时输入ca，CHaron的日志不会进行展示    3.11 浏览日志与购买日志  点击购买日志记录界面弹窗提示用户    点击左侧浏览浏览日志弹出提示进入浏览日志界面进行查看    上方搜索框可以根据用户账号搜索对应的日志，此时输入ar，CHaron的日志会进行展示  上方搜索框可以根据用户账号搜索对应的日志，此时输入chari，CHaron的日志不会进行展示  上方搜索框可以根据用户账号搜索对应的日志，此时输入s，CHaron的日志不会进行展示   1. 应用部署   给出部署的详细过程；  **给出部署的地址，以及相关的测试ID和口令。**  4.1前端vue导出dist文件夹  在vue文件夹中打开终端执行npm run build即可生成dist文件夹，其中包含了打包好的前端项目。    4.2 后端springboot导出jar包  IDEA中有生成springboot工程jar包的图形化界面工具，可以直接使用：    依次点击clean，compile和package，即可生成jar包，存放在target目录下。效果如下：  先clean：    再compile：    最后package：    如果package命令可以看到BUILD SUCCESS字段说明jar已经生成成功，可以点击target目录进行查看，可以看到已经生成了名为perk的jar包：    4.3 mysql导入备份sql文件  4.3.1 在Windows终端使用mysqldump备份本地数据库：    运行完命令后可以看到桌面上多了一个shop.sql文件，此文件可以完全恢复shop数据库的信息。  4.3.2 将shop.sql备份文件导入云端数据库：  先打开服务器的宝塔面板操作界面，看到左边的数据库点击    点击添加数据库，填写相关信息    创建一个与本地数据库相同的数据库（shop），创建完毕后效果如下：    点击备份列的导入按钮，导入本地的数据库备份文件，导入之后的效果如下    4.4 腾讯云账号注册与云服务器申请  首先注册一个腾讯云账号并且登录  然后就是购买一个轻量级服务器，如下：  4.5 服务器环境部署  4.5.1 编辑nginx配置文件  切换到nginx程序所在目录/usr/local/nginx，打开其配置文件/conf/nginx.conf,内容如下：    编辑server块信息：    参数说明：server\_name是服务器ip地址，root是vue前端dist文件路径，proxy\_pass是访问到/api/路径之后跳转到的url地址，这里为本地的9090端口，与springboot工程端口保持一致，这样后端才可以收到前端的请求进行处理。  4.5.2 启动springboot后端jar包  来到jar所在目录，输入nohup命令，表示不关闭jar包运行（即使在本地页面关闭后也保持运行），效果如下：    这样说明jar包已经成功启动，并且永远不能关闭（除非手动运行kill命令）   1. 参考文献   [1]王龙军,王晶,许靖唯.基于Spring Boot与Vue.js的问卷模块在数字阅读推广系统中的设计与实现[J].内蒙古科技与经济,2023(18):115-118.  [2]潘涛,王柳,董冉冉.基于Vue.js框架的网上商城管理系统的设计与实现[J].科技与创新,2023(13):8-10.DOI:10.15913/j.cnki.kjycx.2023.13.003.  [3]银莉,杜啸楠.基于SpringBoot和Vue的校园积分系统的设计与实现[J].集成电路应用,2023,40(07):414-415.DOI:10.19339/j.issn.1674-2583.2023.07.187.  [4]庞嵩昊,李盈,赵艺等.基于Vue和SpringBoot前后端分离的宠物服务系统的设计与实现[J].电脑知识与技术,2023,19(21):42-45.DOI:10.14004/j.cnki.ckt.2023.1068.  [5]王科翔,刘星含,娄志城等.基于Spring Boot+Vue框架的智慧学习平台的设计与实现[J].无线互联科技,2023,20(14):65-68. |
| **小结** |
| 在本次《网络应用开发》课程实验中，我选择采用了Vue+SpringBoot框架进行开发，而没有选择传统的JSP网页搭建。这一决定基于对现代前后端分离开发模式的认识和对Vue+SpringBoot框架优势的了解。通过使用Vue+SpringBoot，我能够更好地实现前后端的分离，提高开发效率，并且能够更好地应对复杂的前端交互和数据展示需求。然而，在学习和应用Vue+Spring Boot框架的过程中，我遇到了许多困难。首先是对全新的开发模式和技术栈的不熟悉，需要花费一定的时间去理解和掌握。其次是在实际开发中遇到了各种各样的问题，比如跨域访问、数据传输格式、前后端接口对接等等。为了解决这些问题，我不断地搜索资料、阅读文档，并且尝试了各种方法，最终逐渐克服了这些困难，积累了宝贵的经验。除此之外，在实验过程中，我还学会了如何将项目部署到服务器端，以实现公网访问网页的功能。这涉及到服务器环境的搭建、项目的打包部署、域名解析等一系列操作，通过自己的实践和摸索，我成功地将项目部署到了服务器上，并实现了公网访问网页的效果。这为我提供了宝贵的实际操作经验，也增强了我对项目部署和运维的理解和能力。通过本次实验，我不仅掌握了现代化的前后端分离开发模式和流行的Vue+Spring Boot框架，还积累了大量的实际开发经验和解决问题的能力。这些经验和能力将对我的未来学习和工作产生积极的影响，使我能够更好地应对复杂的网络应用开发任务和挑战 |
| **指导教师评语及成绩** |
| 评语：  成绩：    指导教师签名：                                                  批阅日期： |