天津大学

《专业课程设计2》

结课报告

学　　院　智能与计算

专　　业　软件工程

年　　级　2022级

姓　　名　孙佳楠 吉雅 刘文翔

学　　号　**3022244115 6322000326 3022244119**

2024 年 **9** 月 **19** 日

目　录

第一章　　项目概况

1.1　　实践背景

T软公司为了站在互联网的风口起飞，决定快速开发一个互联网APP，模仿著名的互联网点外卖的APP“饿了么”，开发一个竞品APP“饿了吧”。此项目要求必须三周内上线，时间紧任务重，本着小步快跑、快速迭代的原则，“饿了吧”第一期仅开发下列功能：

商家入驻平台，管理待售商品。

用户注册，管理个人账号和个人信息。

用户点餐，管理购物车，管理送货地址，下单支付。

注：支付界面中点击确认后，将调用支付宝等平台完成实际支付，此功能由其他组开发，本项目组不必开发，直接返回支付成功即可。

●项目现状

唯一的程序员小东已跑路，现在项目组只有你们几个实习生。

经检查，小东仅完成了用户点餐功能，包括购物车管理、送餐地址管理、下订单几个功能，其它功能并未完成，像用户、商家、商品等数据，是在数据库中直接添加的。已完成的功能中，也存在不少问题，有功能上的问题，也有设计上的问题。

现有的材料，包括：部分不规范的文档和代码，公司要求小东录制的一些技术讲座视频。

1.2　　实践内容

重新进行项目需求分析，达到预期目标，项目重构，解决原项目中存在的问题。

1、功能方面的问题

功能不完整，只有用户下订单一个流程，缺少其他必要的基本功能。

有的页面无法返回上一步。首页之后的页面缺少返回键，用户只能一路点到底才能退出。

在对订单中涉及的商品名称、价格等信息进行修改时，会引起历史订单信息的修改，应在存储历史信息时进行拷贝存储。修改送货地址也有类似问题。

添加新的收货地址的问题。在新增收获地址时会出现显示的用户信息与新增的不同的情况。原因是前端在取数据时，取得的是登录用户的信息。用户也可以给其他人点餐，根据所选地址进行了显示信息的更改。

前端订单页面的小数显示上存在 bug，可能会出现小数点后很多位数字，没有使用 toFix 函数进行更改限制。

购物车不能展开，无法查看自己已经买的东西的单价和数量。

历史未支付订单应该有支付入口

该项目的CSS很多是写死的，没有考虑页面适配，稍微比例有些问题会出bug。作为一个手机网页端应该尝试满足不同机型的比例适配。

前端版本代码均存在商家列表/食品列表不能拉到底，最后一个信息会被底部菜单栏遮挡，可以理解为下方没有一块空白将列表项上去而底部菜单栏又将其遮挡住。

2、设计和安全方面的问题

前后端接口设计不规范，完全不符合RESTful风格。

用户隐私信息如：电话号码、登录密码等，通过接口对参数进行明文传输使得用户的隐私信息被直接暴露，造成极大的安全隐患。

后端向前端返回的对象为数据库中的该对象的全部信息，例如返回给前端的用户实体包含该用户密码，这是一个不安全的设计，直接将一些不该暴露的信息暴露给前端，存在巨大的信息安全隐患，应当把向前端返回的信息单独封装进一个类，只返回必要的信息。

用户注册时，存入的密码未经任何加密处理，这里应该将密码单向加密后再存入数据库。

后端对前端提交的数据和身份并不检验，这就导致可能有伪造的前端提交恶意数据，可能修改他人送餐地址，或者修改订单金额等。

商家端删除食品时，未曾对该食品是否属于该商家进行验证，导致可能会出现某商家删除其他商家食品的现象。

商品不是无限的，应该有数量，订购时应加互斥锁

考虑到在实际应用场景中可能存在网络异常情况，例如用户进行订单支付时接口超时，重新调用接口会导致重复扣款，严重影响软件质量及用户评价。

后端代码订单业务逻辑缺少对订单和购物车是否为空的验证。

项目四的后端跨域处理方案只允许本地进行访问，完全无法部署上线。

没有部署，仍然是在开发环境中演示。

3、开发路线

基于公司的技术基础和偏好，本项目后续开发将延用之前采用的前后端分离的技术路线。

后端：Spring Boot

前端：VUE 3

数据库：MySQL

1.3 实践目的

了解软件工程实践的基本步骤，掌握面向对象分析（OOA）的基本方法。

1.4 小组成员分工

小组成员：孙佳楠，吉雅，刘文翔

孙佳楠负责后端项目功能的开发，吉雅、刘文翔负责前端项目功能页面的开发，编写项目需求规格说明书。

全体参与项目设计文档、项目测试文档、项目部署文档及本报告的编写。

1.5 本报告的大概内容分布

1.软件需求规格说明书（第二章整体）

2.项目设计文档（第三章整体）

3.项目测试文档（第四章整体）

4.项目部署文档（第五章整体）

5.小组成员分工情况说明（第一章中的1.4小组成员分工）

6.项目开发过程描述，组会、协作、Gitee 上团队成员贡献等，要有依据，如会议记要，Gitee上的上传记录数据等（第四章中的4.2项目开发测试过程）

7.项目特色之处（第三章中的3.1.2特色功能）

8.项目进程过程中的问题点、解决对策、结果反思等（第四章中的4.1测试问题与解决方式）

第二章　　软件需求规格说明书

2.1　　简介

2.1.1　　背景

“饿了吧”是一个在线订餐服务平台，旨在为用户提供便捷的餐饮预订、支付、配送及订单管理等服务。平台同时支持商家入驻，管理菜品、订单及用户评价等功能。目前，“饿 了吧”第一期仅开发以下功能：

1. 用户：可以浏览餐厅、下单、支付、查看订单状态
2. 商家：能够管理菜单、接受订单、更新订单状态
3. 配送员：可以接收配送任务、更新配送状态、查看送餐路线

2.1.2　　约束

插本项目支付界面中点击确认后，将调用支付宝等平台完成实际支付，此功能由其他组开发，本项目组不必开发，直接返回支付成功即可。

2.2　　需求分析

2.2.1　　目标

表 2-1　　目标分类及其内容

目标分类	目标内容
功能性目标	提供用户下单、餐厅管理、配送跟踪、支付功能等；支持实时订单更新和用户反馈。
非功能性目标	系统需要高可用性、响应速度快、数据安全性高、支持高并发。

2.2.2　　涉众

表 2-2　　涉众及其参与的过程表

涉众	涉众涉及的业务范围
用户	下单的顾客、餐厅的消费者。
开发人员	负责系统设计、开发、维护和更新的技术团队。
餐厅	需要管理自己的菜单和订单的餐饮商家。
配送员	负责取餐和送餐的人员。（不一定能完成）
管理层	负责运营、战略规划和决策的管理人员。

2.2.3　　范围

功能范围：包括用户下单、餐厅管理、配送调度、支付系统、用户反馈和评价等。

地域范围：餐厅与用户地址应有恰当的距离

2.2.4 项目成功标准

功能符合性：所有主要功能按预期正常运行。

质量标准：系统稳定性高、性能良好、具备高水平的安全性。

2.3 业务概念

2.3.1 概述

平台主要管理的事物有：用户、餐厅、配送员、订单、评价。

用户与订单：用户可以下多个订单（1对多关系）。

餐厅与订单：餐厅可以接收多个订单（1对多关系）。（未完成）

订单与配送员：每个订单由一个配送员配送（多对1关系）。（不一定能完成）

订单与评价：每个订单可以有一个评价（1对1关系）。

2.3.2 类图描述

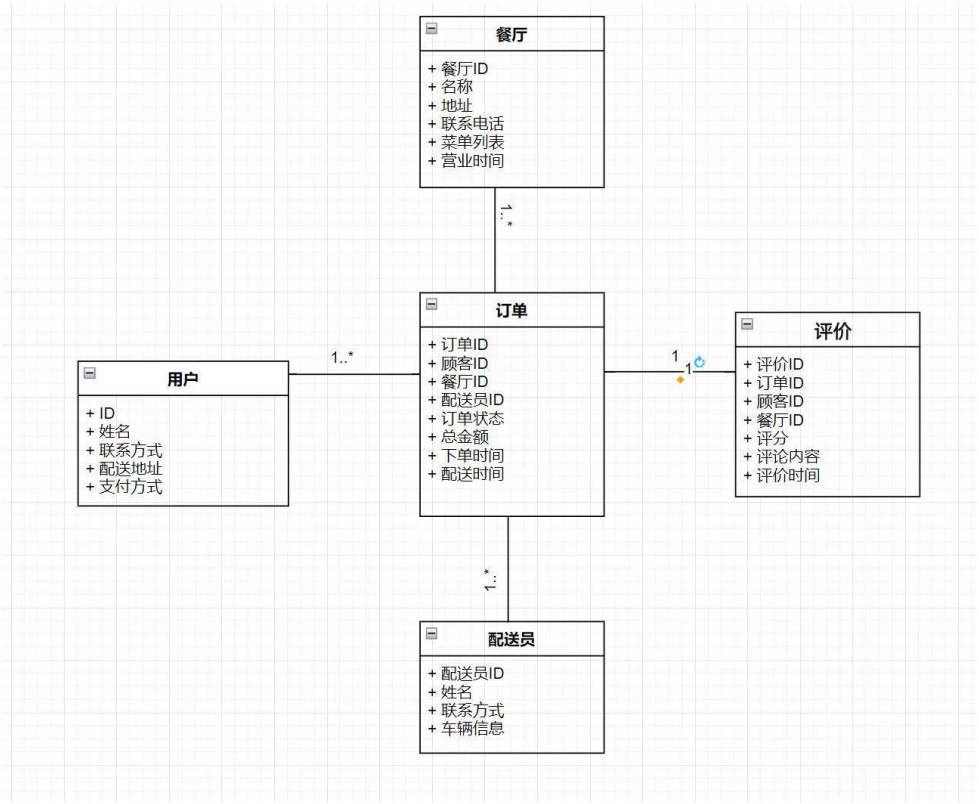


图 2-1 上述业务的类图说明

2.4 业务状态分析

2.4.1 概述

描述一个订单如何完成的业务过程。

2.4.2 状态图描述

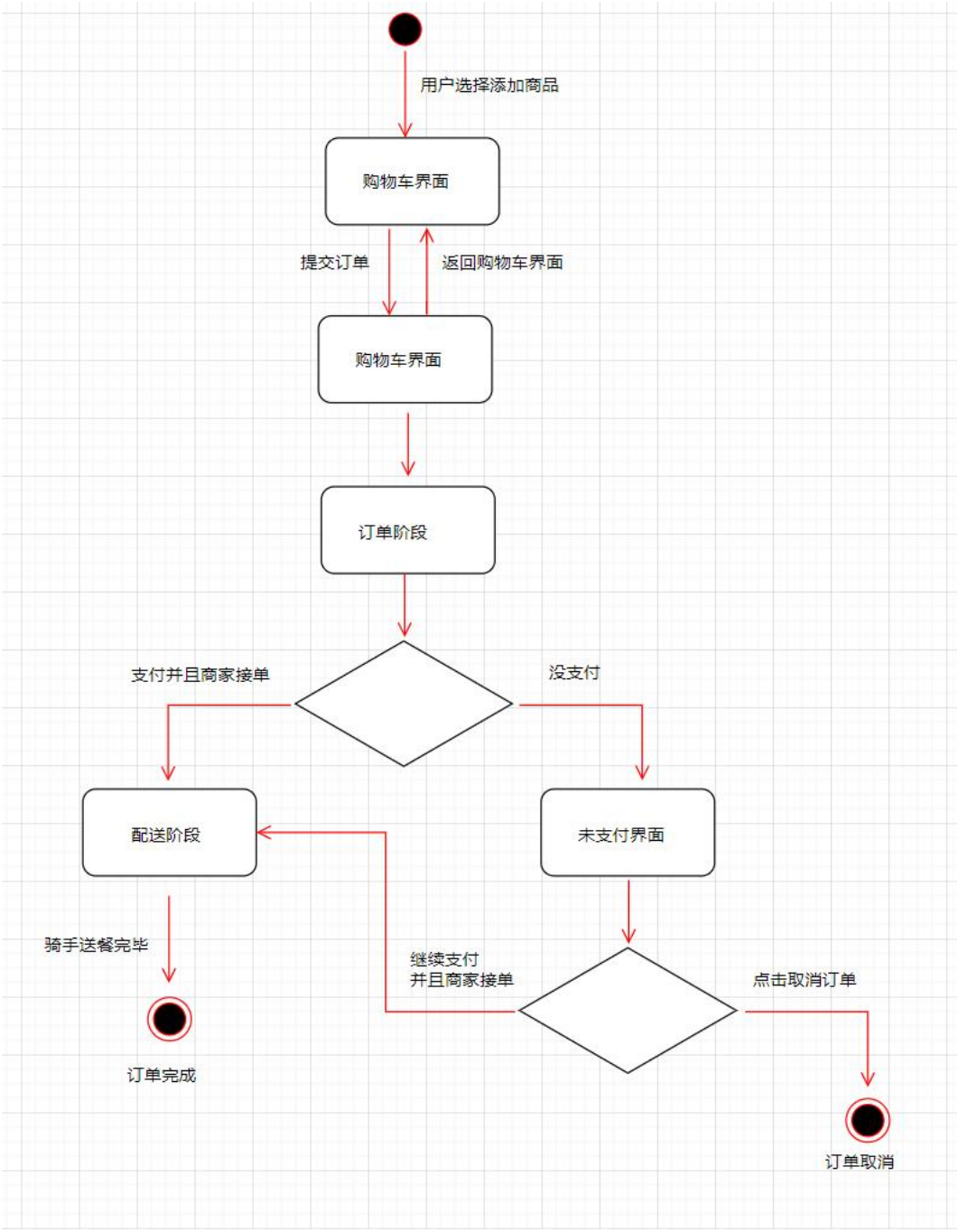


图 2-2 下一个订单的状态图说明

2.5 功能性需求

2.5.1 执行者分析

表 2-3 执行者分析表

执行者	执行任务
平台管理员	饿了吧平台内部的运营和管理人员，负责系统的维护和管理。
商家	入驻饿了吧平台，提供商品的商家。
普通用户（消费者）	使用平台订餐的个人用户。

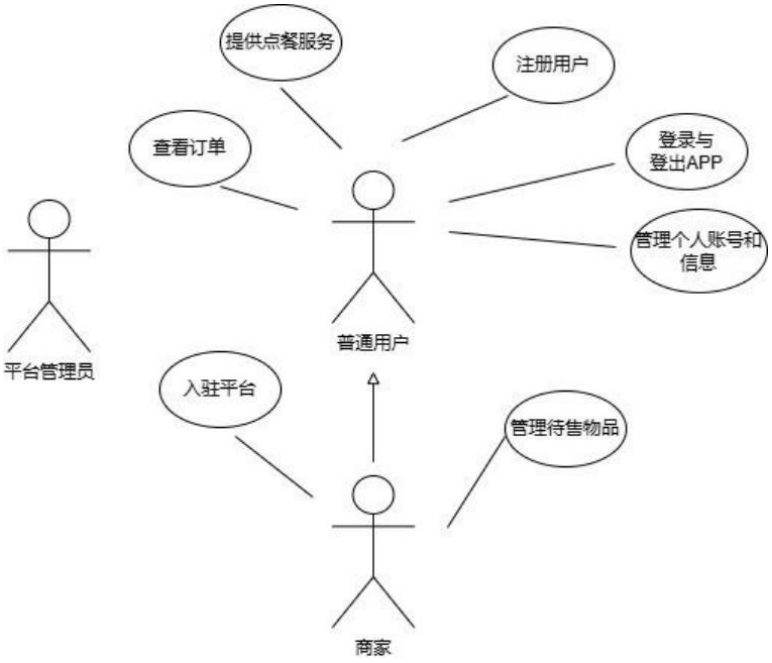


图 2-3 宏观用例图

展示出该APP的大致需求，接下来我们将通过多张分解的用例图以及部分用例的用例表，来详细说明APP的需求。

2.5.2 用例分析

2.5.2.1 普通客户用例分析

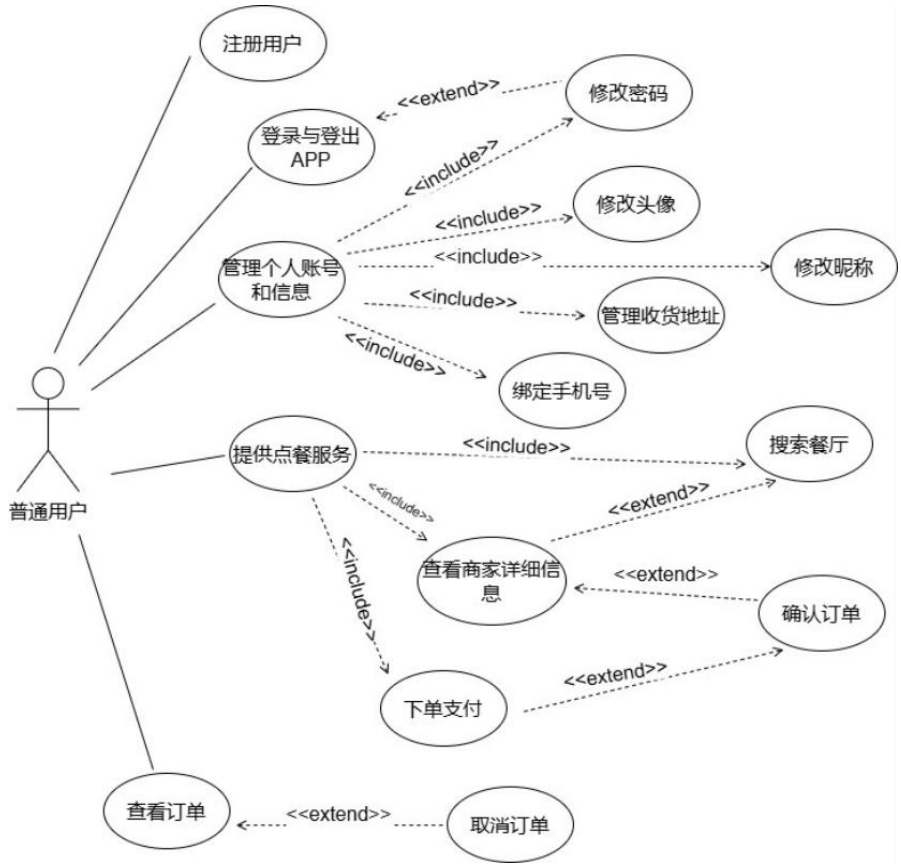


图 2-4 普通客户用例图

表 2-4 UC-01修改密码 用例表

编号	UC-01
名称	修改密码
执行者	普通用户，商家，管理员
优先级	中
描述	通过验证手机号修改密码。
前置条件	1. 忘记登录密码，无法登录。
	2. 已登录。
基本流程	1.提出修改密码。
	2.显示输入已绑定的手机号，发送验证码。
	3.输入验证码，指示提交验证。
	4.显示验证通过。
	5.指示修改密码。
	6.显示修改密码界面。
	7.填写新的密码。
	8.显示修改成功，重新登录。
结束状况	密码修改成功，系统保存用户新数据。

表 2-5　　UC-02修改头像 用例表

编号	UC-02
名称	修改头像
执行者	普通用户，商家，管理员
优先级	低
描述	从相册选择的图片，修改头像。
前置条件	无
基本流程	1.上传图片。 2.显示上传成功。
结束状况	显示图片上传成功。
说明	系统保存用户新数据。

表 2-6　　UC-03修改昵称 用例表

编号	UC-03
名称	修改昵称
执行者	普通用户
优先级	低
描述	修改用户昵称
前置条件	无
基本流程	1.指示修改用户昵称。 2.显示修改成功。
结束状况	显示修改成功
说明	1. 一个月允许更改一次。 2. 系统保存用户数据

表 2-7　　UC-04绑定手机号 用例表

编号	UC-04
名称	绑定手机号
执行者	普通用户、商家、管理员
优先级	高
描述	用户账号绑定手机号，可以修改绑定手机号。
前置条件	无
基本流程	1.注册时指示绑定手机号。
结束状况	显示绑定成功。
可选流程	1.指示验证账号密码。 2.显示密码正确，验证通过。 3.指示修改手机号。 4.显示修改成功。
说明	1.若已绑定的手机号不再使用，可以通过验证密码，从而修改绑定的手机号。 2.系统保存用户数据。

表 2-8　　UC-05管理收货地址 用例表

编号	UC-05
名称	管理收货地址
执行者	普通用户、商家、管理员
优先级	
描述	管理收货地址，如修改，删除，新增地址。
前置条件	无
基本流程	1.用户确认订单后，显示订单信息。 2.选择收货地址。
结束状况	成功选择收获地址，以及收货人信息。
可选流程	无
说明	1.收获地址信息包括地址，收货人姓名，手机号。 2. 修改地址不会影响已完成的订单信息。 3. 收货地址的用户信息可以和登录用户的信息不同。

表 2-9　　UC-06搜索餐厅 用例表（不一定完成）

编号	UC-06
名称	搜索餐厅
执行者	普通用户、管理员
优先级	
描述	通过在搜索框中搜索关键词，找到与之相符的餐厅。
前置条件	无
基本流程	1.搜索关键词。
	2.显示与之相符的商家店铺列表。
结束状况	显示商家店铺列表。
可选流程	1. 指示首页点餐分类信息。
	2.显示商家列表。
说明	在此基础上可以查看商家详细信息。

表 2-10　　UC-07查看商家详细信息 用例表

编号	UC-07
名称	查看商家详细信息
执行者	普通用户、商家、管理员
优先级	
描述	查看商家详细信息及该店铺食品信息。
前置条件	显示商家列表信息。
基本流程	1.指示某个商家。
	2.显示商家详细信息。
结束状况	显示商家详细信息，该店铺食品信息。

表 2-11　　UC-08购物车 用例表

编号	UC-08
名称	购物车
执行者	普通用户
优先级	
描述	展开购物车，查看以选定的物品的物品名、单价、数量以及小计。
前置条件	已经选好要购买的物品。
基本流程	1.指示展开购物车。
	2.显示物品的单价和数量。
结束状况	显示已选定物品的的物品名、单价、数量以及小计。
说明	无

表 2-12　　UC-09确定订单 用例表

编号	UC-09
名称	确定订单
执行者	普通用户
优先级	
描述	确认商品信息、收货地址、以及收货人信息。
前置条件	选择需购买的物品。
基本流程	1.选餐后指示结算。
	2.显示确认订单页面。
	3.确认商品信息，指示支付。
结束状况	确认商品信息，配送费等。
说明	在此基础上可以下单支付

表 2-13　　UC-10下单支付 用例表

编号	UC-10
名称	下单支付
执行者	普通用户
优先级	
描述	完成支付。
前置条件	确保已确认商品新和收货人信息。
结束状况	成功支付并生成订单。
异常流程	支付失败。
说明	1. 历史未支付订单应有支付入口。
	2. 系统保存用户数据。

表 2-14　　UC-11查看订单 用例表

编号	UC-11
名称	查看订单
执行者	普通用户
优先级	
描述	查看历史订单信息。
前置条件	无
基本流程	1.在首页中指示订单。
	2.显示历史订单。
结束状况	显示历史订单。
说明	1. 历史订单中应有已支付的订单和未支付的订单。
	2. 未支付的订单应有支付入口和取消订单入口。

2.5.2.2　商家用例分析

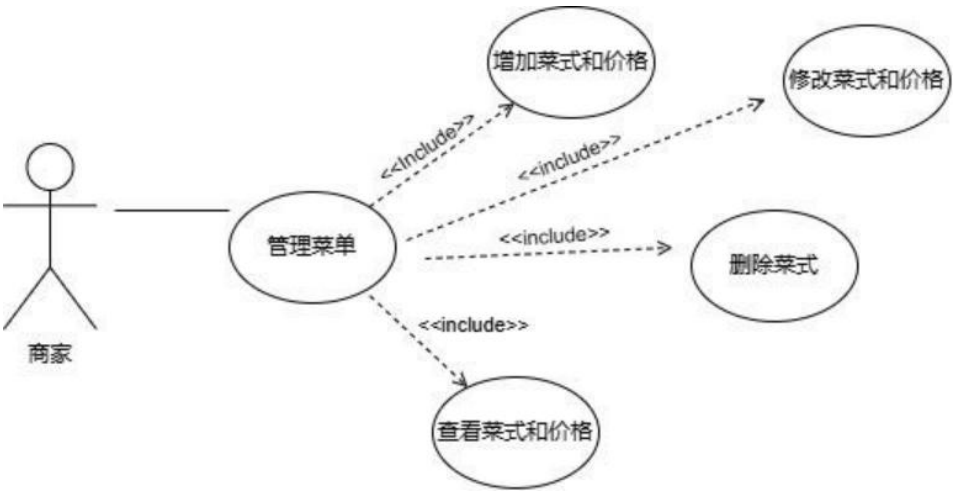


图 2-5　　商家用例图

表 2-15　　UC-12填写店铺信息 用例表

编号	UC-12
名称	填写店铺信息
执行者	商家
优先级	
描述	填写店铺的详细信息，包括店铺名称、地址、营业时间、店铺介绍等。
结束状况	成功填写店铺信息。
说明	完成后店铺后上线。
	系统保存用户数据。

表 2-16　　UC-13管理菜单 用例表

编号	UC-13
名称	管理菜单
执行者	商家
优先级	
描述	上传图片，设置菜单、价格以及库存等。
前置条件	填写店铺协议。
结束状况	对菜单增删改查。
说明	1. 可以在菜单中增加菜式、删除菜式、修改菜单和库存等。 2. 对商品的名称、价格等信息进行修改时，不能引起已支付订单的修改。 3. 商家只能管理自己的店铺，不能修改其他店铺的信息。 4. 系统保存用户数据。

2.5.2.3　管理员用例分析

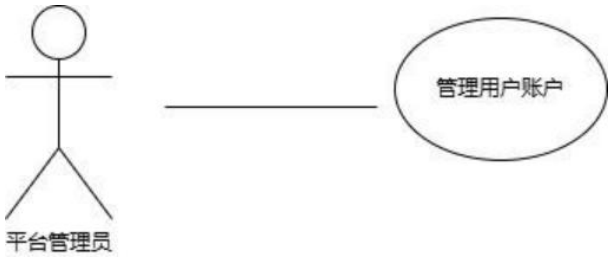


图 2-6　　平台管理员用例图

表 2-17　　UC-14管理用户账户 用例表

编号	UC-14
名称	管理用户账户
执行者	管理员
优先级	
描述	管理平台上的普通用户和商家用户。
前置条件	拥有平台的管理权限。
结束状况	系统保存操作数据，确保数据追溯和系统审计。

2.6 非功能性需求

2.6.1 其他非功能性要求

表 2-18 其他非功能性要求表

非功能性要求	内容
安全性	1.所有用户隐私信息（如电话号码、登录密码等）必须通过加密传输进行，前端和后端之间的数据传输应只包含必要的信息，避免信息泄露。 2.后端需要对前端提交的数据和身份进行严格验证，以防止伪造数据的提交。 3.应有机制防止网络异常导致重复扣款。
易用性	1.用户在浏览过程中应能够方便地返回上一步，所有页面应有明显的返回键或导航选项。 2.修改订单信息时，系统应正确处理历史订单的存储，避免对历史数据的错误修改。 3.系统应确保前端页面小数点准确显示。
性能	1.系统应支持各种手机型号的屏幕比例，避免因比例问题导致的页面显示异常。 2.页面加载和交互应快速响应，避免因加载速度慢影响用户体验。 3.所有接口的响应时间应在合理范围内，确保用户操作的流畅性。
接口	1.前后端接口应遵循 RESTful 风格，确保接口设计规范且易于维护。 2.接口应对敏感信息进行保护，避免直接暴露用户的私人数据。 3.接口应提供详细的错误信息和处理机制，确保在出现异常时能够快速定位和解决问题。
其他	1.系统应在各个功能模块之间建立完整的功能链条，包括未支付订单的支付入口、购物车的功能等，避免遗漏基本功能。 2.系统应在开发完成后进行充分的测试，并部署到正式环境中，确保演示和实际使用中的一致性。

2.6.2 用部署图和构件图设计系统技术框架

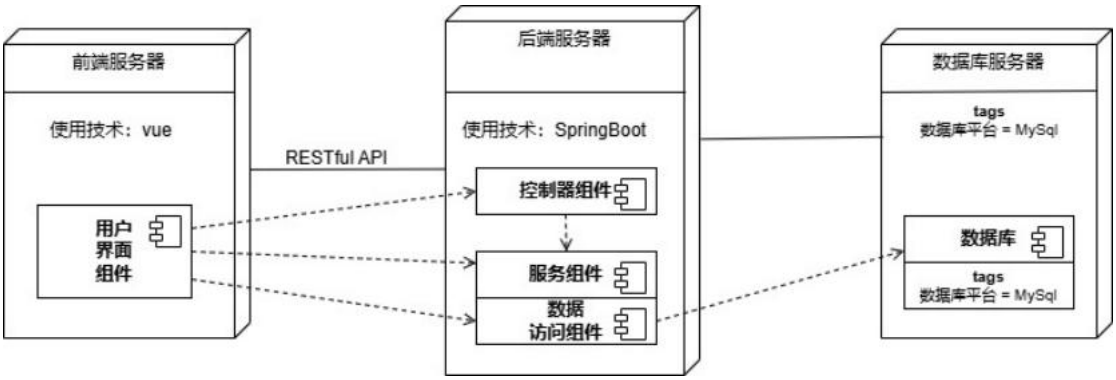


图 2-7 软件架构图

第三章　　项目设计文档

3.1 功能

3.1.1 基本功能

用户可以注册账号，登录账号，退出登录。

用户可以查看商家的详细信息，包括商家名称，菜单，菜品价格。

用户可以将菜品添加到购物车，从购物车移除等。

用户可以在确认订单页面，修改或新增收货地址信息并选择。

用户可以添加多个收货信息，包括姓名，手机号，和地址。

用户可以选择支付或取消支付。

用户可以成功完成订单支付过程。

用户可以查看历史订单信息，包括未支付订单和已支付订单。

用户可以在未支付订单的支付入口进行支付。

3.1.2 特色功能

用户可以查看购物车中商品的名称，单价和数量。

用户可以编辑个人信息包括修改头像，修改密码。

用户可以退出登录。

用户可以选择支付或取消支付。

3.2 前端项目设计

vue2升级到vue3：升级node.js支持vue cli3，升级vue.js并更新相关依赖，安装运行迁移工具，逐一修改迁移中的警告，迁移全局和内部API，逐步替换vue2的语法、组件等等。

在配置index.js文件时，由于使用的初始代码为vue2版本，但是最新下载的vue-router版本为3，故出现报错，因此需要修改代码。如图3-1和3-2。

```
import { createApp } from 'vue';
import App from '../App.vue';
//import router from './router';
import { createRouter, createWebHistory } from 'vue-router';
```

图3-1

```
const router = createRouter({
  history: createWebHistory(process.env.BASE_URL),
  routes
});

createApp(App).use(router).mount('#app');
```

图3-2

添加新的收货地址的问题。修改前在新增收获地址或修改地址时会出现显示的用户信息与新增的不同的情况。

前端订单页面会出现多位小数点的情况，对于该问题使用toFixed函数将数字转换为指定小数位数的字符串。如图3-3所示。

```
<ul class="order-detailet" v-show="item.isShowDetailet">
    <li v-for="odItem in item.list">
        <p>{{ odItem.food.foodName }} x {{ odItem.quantity }}</p>
        <p>&#165;{{ (odItem.food.foodPrice * odItem.quantity).toFixed(2) }}</p
    </li>
    <li>
        <p>配送费</p>
        <p>&#165;{{item.business.deliveryPrice}}</p>
    </li>
</ul>
```

图3-3

对于用户一路点到底才能退出的问题，首页之后的页面增加了返回键。

增加了弹出层的组件(src/components/CartPopup.vue)，以便用户访问购物车并查看已添加的商品信息。

前端版本代码均存在商家列表/食品列表不能拉到底的问题，最底部会被底菜单栏遮挡。为解决该问题修改了商家列表部分的格式，即将margin改成了padding。如图3-4所示。

```
.wrapper .business {
    width: 100%;
    margin-top: 12vw;
    padding-bottom: 14vw;
}
```

图3-4

3.3 后端项目设计

增加了订单业务逻辑对订单和购物车是否为空的验证

com/neusoft/elmboot/service/impl/OrdersServiceImpl.java

用户修改密码

```
@RequestMapping("/modifyPassword")
public int modifyPassword(UserPassword userPassword) throws Exception {
 return userService.modifyPassword(userPassword);
}
```

用户修改头像

```
@RequestMapping("/changeUserPortrait")//base64
public int changeUserPortrait( UserImg userImg) throws Exception {
      return userService.changeUserPortrait(userImg);
}
```

删除用户

```
@RequestMapping("/removeUser")
public int removeUser(User user) throws Exception{
     return userService.removeUser(user);
}
```

更新用户

```
@RequestMapping("/updateUserById")
public int updateUserById(User user) throws Exception{
 return userService.updateUserById(user);
}
```

列出全部商家

```
@RequestMapping("/listAllBusiness")
public List<Business> listAllBusiness() throws Exception {
 return businessService.listAllBusiness();
}
```

订单的支付

```
@RequestMapping("/pay")
public void pay(Integer orderId) throws Exception{
 ordersService.pay(orderId);
}
```

更新支付状态

```
@RequestMapping("/turnOrderState")
public int turnOrderState(Orders orders) throws Exception {
 return ordersService.turnOrderState(orders.getOrderId());
}
```

修改了跨域处理方案，在允许本地进行访问的同时，部署上线。

com/neusoft/elmboot/WebMvcConfig_01.java

src/main/java/com/neusoft/elmboot/WebMvcConfig_01.java

第四章　项目测试文档

4.1 测试问题与解决方式

4.1.1 测试内容

1. 登录和注册用户功能：

测试注册用户功能：验证用户信息是否保存到数据库。

测试登录用户功能：验证用户能否通过手机号和密码成功登录。

测试推出登录功能：验证用户能否推出登录，并回到首页。

2. 管理购物车：

测试访问购物车功能：商家信息页面中，验证能否显示购物车中的信息。

测试增减商品功能：验证是否可以增减需购商品数量。

3. 修改用户信息功能：

测试修改头像功能：验证是否能修改头像，并在用户信息页面中显示新的头像。

测试修改密码功能：验证是否能修改密码，并使用新密码登录。

4. 管理地址功能：

测试修改地址功能：验证是否修改成功，新的地址是否保存到数据库。

测试增加地址功能：验证是否增加成功，新的地址是否保存到数据库。

测试选择地址功能：验证在确认订单页面中，是否显示正确的地址信息。

5. 查看历史订单功能：

测试查询订单功能：验证订单页面是否显示所有的订单信息，订单状态是否正确。

测试生成订单功能：验证在商家界面提交订单后是否生成订单。

6. 支付订单功能：

测试支付订单功能：验证是否可以成功支付，是否可以取消支付，是否可以支付未支付的历史订单。

4.1.2 前端

在IDEA环境中编译，有时遇到无法执行目标文档的报错（如图4-1）。解决方法：在npm install后面加上--legacy-peer-deps就可以解决问题。

图 4-1

在.vue文件中的methods之间记得格式问题，要用英文逗号隔开。

```
SyntaxError: D:\elm\cs\elm3\elmclient\src\views\MyImg.vue: Unexpected token, expected "," (37:3)

  35 |                         }
  36 |                 }
> 37 |                 saveImg(base64String) {
     |                 ^
  38 |                         this.$axios.post('UserController/changeUserPortrait', this.$qs.stringify({
  39 |                                         userId: this.$route.query.userId,
  40 |                                         base64: base64String
```

图 4-2

编译好的功能未及时更新，修改后，刷新一下数据库就可以解决问题。

vue版本的不适配问题，在index.js文中，由于初始代码为vue2版本，但是最新下载的vue-router版本为3，因此需要修改main.js和index.js。

测试项目是有时遇到编译，打包正确但无法与前端连接上的问题，检查发现：

由于前端根目录路由到登陆页面login，从而导致路径请求错误（如图4-3）。应在路由配置文index.js中，对根路径的路由设置为index，而不是login。

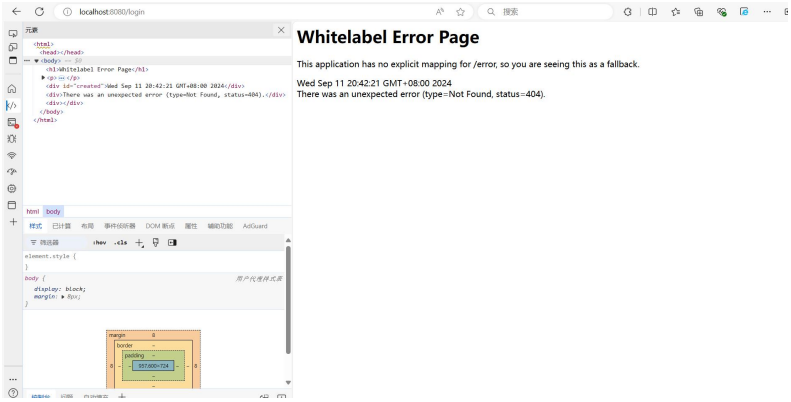


图 4-3

2.删去后端pom文件中的依赖包

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-security</artifactId>
</dependency>
```

订单状态未修改成功，导致已支付成功的订单在历史未支付订单列表中。经检查后发现是前后端接口路径不一致引起的，应该是turnOrderState，而不是updateOrderState。

部分接口测试如下图4-5，图4-6，图4-7，图4-8所示：

测试工具Apifox

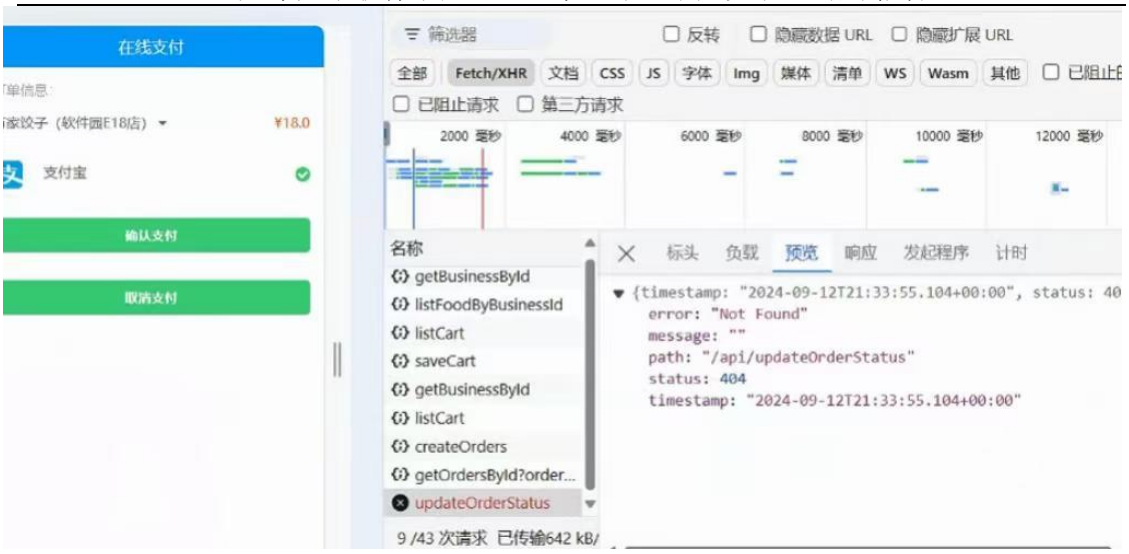


图 4-4

图 4-5 列出所有商家

图 4-6 订单状态改为已支付1

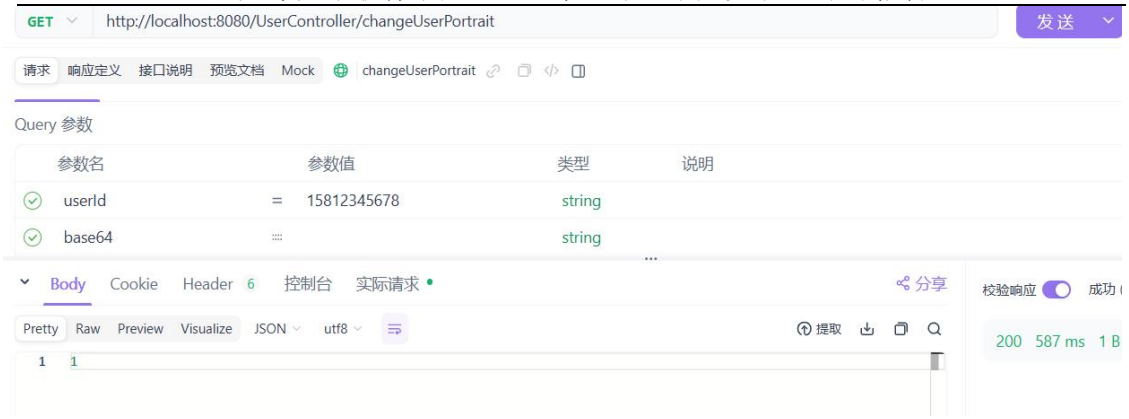


图 4-7 成功修改头像并返1

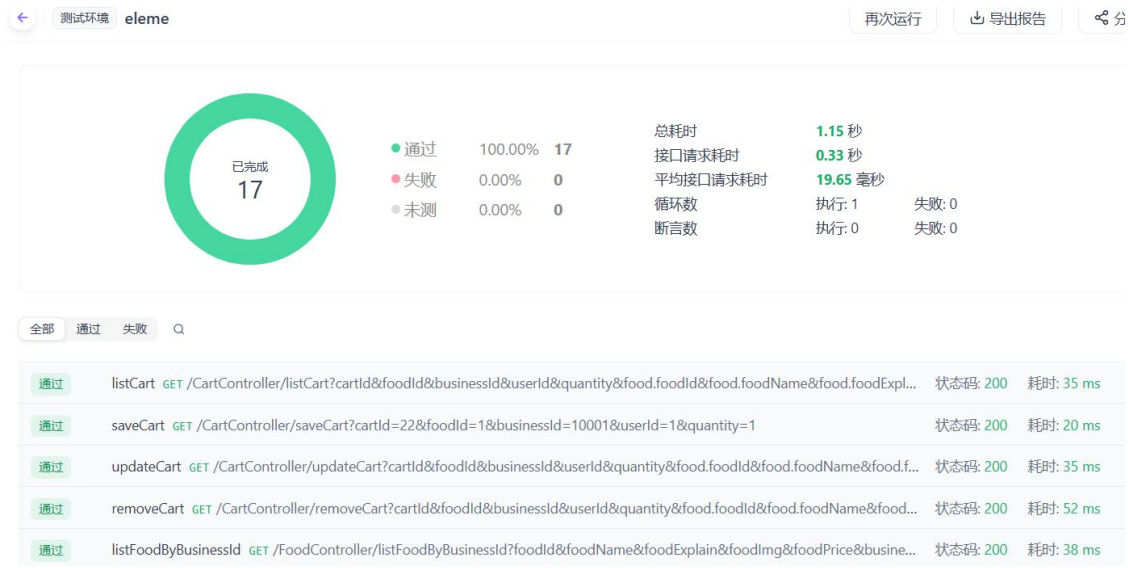


图 4-8 自动化测试

4.1.3 后端

在IDEA环境中编译，有时遇到无法执行目标文档的报错，解决方法：

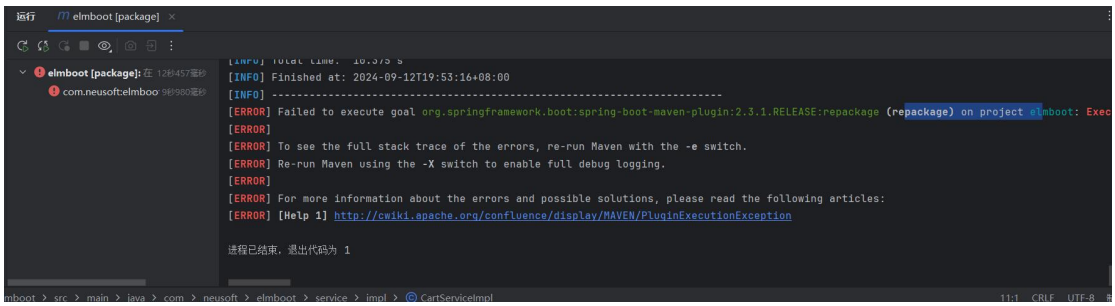


图 4-9

1. 可能是mvn package打包生成的架包还在运行，关闭相应的页面和窗口，停止运行。

2. jdk版本不匹配，检查pom文件，项目结构中的SDK，语言级别以及设置中的java编译器

3. 可能是依赖包没有下载好，手动输入下载指令

4. 可能是maven的仓库，setting路径设置不对

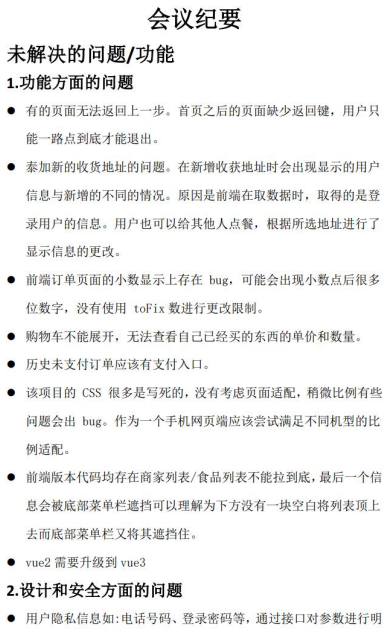
5. 清空缓存重新启动

6. 清空maven仓库中的文件，重新下载依赖包

4.2 项目开发测试过程

我们小组曾开过两次线上会议并做了充分的会议纪要，包括但不限于未解决的问题和新测试出来的问题、已解决问题的测试、任务再分工等。

4.2.1 会议纪要



会议纪要

未解决的问题/功能

1.功能方面的问题

- 有的页面无法返回上一步。首页之后的页面缺少返回键，用户只能一路点到底才能退出。
- 泰加新的收货地址的问题。在新增收获地址时会出现显示的用户信息与新增的不同的情况。原因是前端在取数据时，取得的是登录用户的信息。用户也可以给其他人点餐，根据所选地址进行了显示信息的更改。
- 前端订单页面的小数显示上存在 bug，可能会出现小数点后很多位数字，没有使用 toFix 数进行更改限制。
- 购物车不能展开，无法查看自己已经买的东西的单价和数量。
- 历史未支付订单应该有支付入口。
- 该项目的 CSS 很多是写死的，没有考虑页面适配，稍微比例有些问题会出 bug。作为一个手机网页端应该尝试满足不同机型的比例适配。
- 前端版本代码均存在商家列表/食品列表不能拉到底，最后一个信息会被底部菜单栏遮挡可以理解为下方没有一块空白将列表顶上去而底部菜单栏又将其遮挡住。
- vue2 需要升级到 vue3

2.设计和安全方面的问题

- 用户隐私信息如:电话号码、登录密码等，通过接口对参数进行明文传输，这使得用户的信息被直接暴露，造成极大的安全隐患。
- 用户注册时，存入的密码未经任何加密处理，这里应该将密码单向加密后再存入数据库。
- 商家端删除食品时，未曾对该食品是否属于该商家进行验证，导致可能会出现某商家制除言他商家食品的现象。
- 商品不是无限的，应该有数量，订购时应加互斥锁。
- 后端代码订单业务逻辑缺少对订单和购物车是否为空的验证。
- 项目四的后端跨域处理方案只能允许本地进行访问，完全无法部署上线。
- 没有部署，仍然是在开发环境中演示。

已解决的问题/功能

增加了后端代码订单业务逻辑对订单和购物车是否为空的验证在服务层 OrdersServiceImpl 添加

```
// 验证订单对象是否为空
if (orders == null) {
        throw new IllegalArgumentException("订单对象不能为空");
}

// 验证订单中的必要字段是否有效
if (orders.getUserId() == null || orders.getBusinessId() == null) {
        throw new IllegalArgumentException("用户 ID 和商家 ID 不能为空");
}
```

a）　　b）

```
// 验证购物车是否为空
if (cartList == null || cartList.isEmpty()) {
        throw new IllegalArgumentException("购物车不能为空");
}
```

增加了用户修改密码的功能 `public int modifyPassword(UserPassword userPassword);`

增加了支付功能 `void pay(Integer orderId)`

完成了 vue3 需要的依赖升级和组件更新，但未进行测试

c）

图 4-10　会 议 纪 要 1

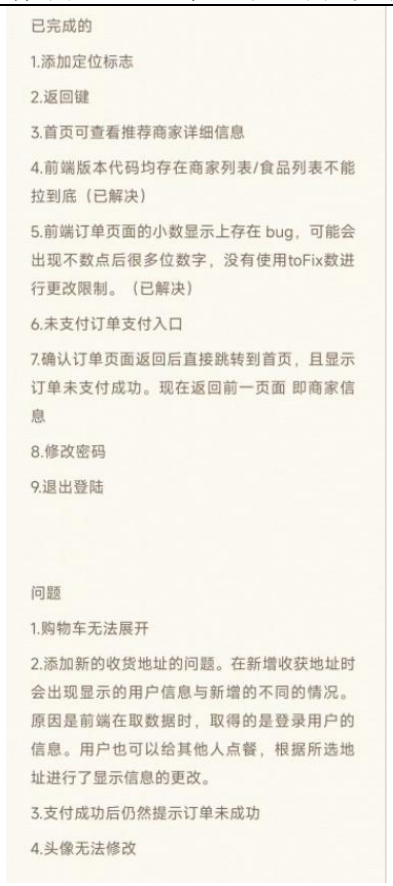
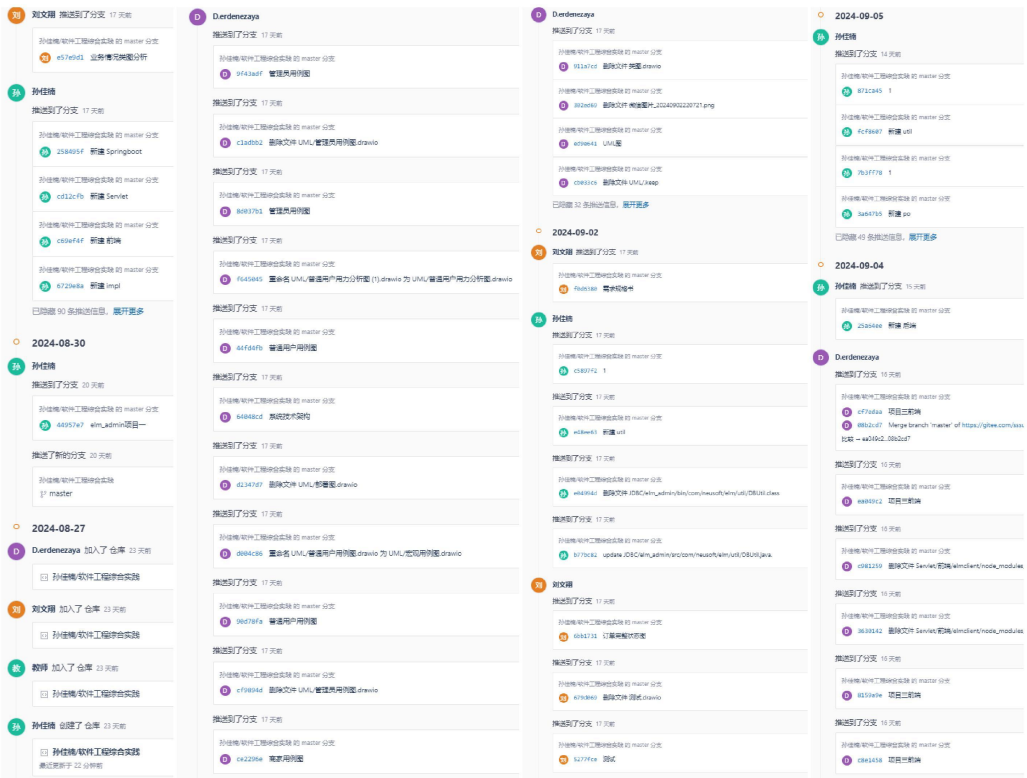
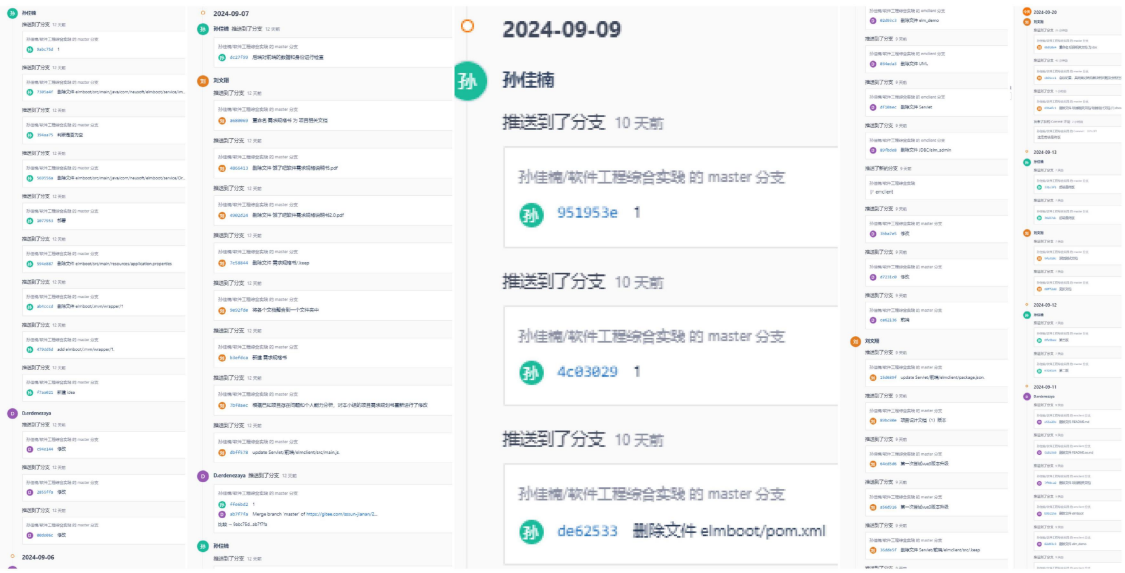


图 4-11 会议纪要 2

4.2.2 gitee上传记录



a）



b)

图 4-12 gitee上传记录

第五章　项目部署文档

5.1　项目具体分布描述

“饿了吧”项目可以分为四个小项目，其具体内容如表5-1。

表 5-1　项目描述表

项目	内容
项目一	纯后端的字符界面的操作数据库的示例，主要介绍JDBC技术
项目二	纯前端的静态页面制作的一些基本知识，主要介绍HTML、CSS、JavaScript等知识
项目三	从项目三开始时完整的“饿了么”项目，前端使用了VUE（但是教程是版本2），后端是servlet技术。这个项目重点学习VUE框架（需要自行补充学习升级到VUE3）
项目四	仅为后端项目，使用Spring Boot完成了后端，这个后端可以直接对接项目三的前端

5.2　依赖关系

列出项目运行所需的所有外部依赖和服务

vue3的依赖：升级vue、vuex、vue-router、@vue/cli-service

Spring Boot的依赖：pom.xml

5.3　部署环境及步骤

5.3.1　部署环境

采用的前后端分离的技术路线。

后端：Spring Boot

前端：VUE 3

数据库：MySQL

软件要求(操作系统、数据库、中间件等)

环境配置(开发、测试、生产环境的差异)：ideal、node.js

5.3.2　部署步骤

详细的部署流程和步骤：正式部署时，需要在前端代码中设置好正式的后端服务的地址，不能再是 localhost 了，localhost 仅调试时使用

5.4 版本历史

部署文档的版本和变更记录有以下：

elm_admin

elm_demo

Sevlet

elmboot

elmboot123

elmboot3

elmclient_final

elmboot_final

图 5-1 上传过的版本历史图

第六章　　结论

在本次“饿了吧”项目的开发过程中，团队成员紧密合作，明确分工，成功完成了基于VUE3和Spring Boot的前后端分离系统的搭建与部署。通过本次实践，团队不仅加深了对前后端技术的理解，还在数据库设计和性能优化方面取得了较理想成果。同时，团队的代码能力、沟通协作和问题解决能力得到了显著提升。未来，我们将继续探索前沿技术，提升软件开发的能力，力求在系统性能和技术创新上取得更大突破。