**“云快点”管理服务平台**

**——Web应用项目**

**分析与设计**

项目组成员： 周浩政、余洁仪、刘晴、叶晓敏、郑婉茹

组员学 号 ： 17052058、17052012、17052060、17052074、17052156

项目负责人： 周浩政

联系电 话 ： 13435299883

所在班 级 ： 2017级计算机科学与技术专业B班

二〇二〇年六月

**目 录**

1.需求分析 3

1.1问题陈述 3

1.2用例析取 4

1.3系统用例规约 4

1.4功能结构图 7

2.架构设计 7

2.1架构描述 7

2.2架构图 8

2.3系统关键抽象 8

3.类的设计 8

4.页面设计 11

5.测试报告 15

6.技术难点 16

7.活动顺序图 17

8.安装及使用 17

## 1.需求分析

### 1.1问题陈述

1.1.1项目背景：

5G时代的到来，使其所展现的新产品形态也备受关注。5G流量暴增，很多生活场景即将发生巨大变革。餐饮行业覆盖大量人群，其变革的深度和广度也会出现超人预料的变化。“云快点” 通过内部优化和外部整合来适应这个趋势，针对餐饮行业做一些流量运营及管理工作，并将其他领域导入进来形成一批新的商业模式，这对餐饮管理系统行业来说是一次跳跃式的升级。

小型餐饮企业经营有以下问题：就餐时间集中，顾客排队时间长；经营成本攀升，房租水电费、食材成本难以控制；营销收益难估，无法评估营销活动成本利益；面向顾客比较固定，餐厅孤立生存。此外，商家单独开发系统难、商家耗费成本高，难以实现o2o线上线下联动的经营模式。而“云快点”的诞生，通过相应的运营方式使中小型餐厅的管理智能化的同时，让小餐厅抱团取暖，增加营收，带动当地经济发展。

1.1.2本系统具体功能：

本项目面向中小型餐馆店铺，尝试结合大数据，为商家和用户提供优化云餐服务的互联网管理服务平台，设计实现一个“云快点” web应用，按照角色划分为商家和顾客两个端口，其中商家又细分为前厅和后厨端口。

1.商家前台端口：

（a允许商家对餐厅详情、订单、点餐二维码等一系列信息进行管理；

（b拥有清晰的数据分析和报表，实时获取门店真实经营数据；

（c允许商家发布营销活动，助其吸引新客、沉淀老客；

（d订单智能排队，高峰时段缓解配菜压力；

2．商家后厨端口：

（a建立后厨秩序，配菜和传菜无缝对接，节约时间；

3.用户端口：

（a可浏览附近加入本平台的餐厅简要信息；

（b 扫码等方式点餐或外卖预订，及时查询订单进度详情；

### 1.2用例析取

本项目组通过对上述问题陈述的分析，构建本系统的用例图，如下所示。

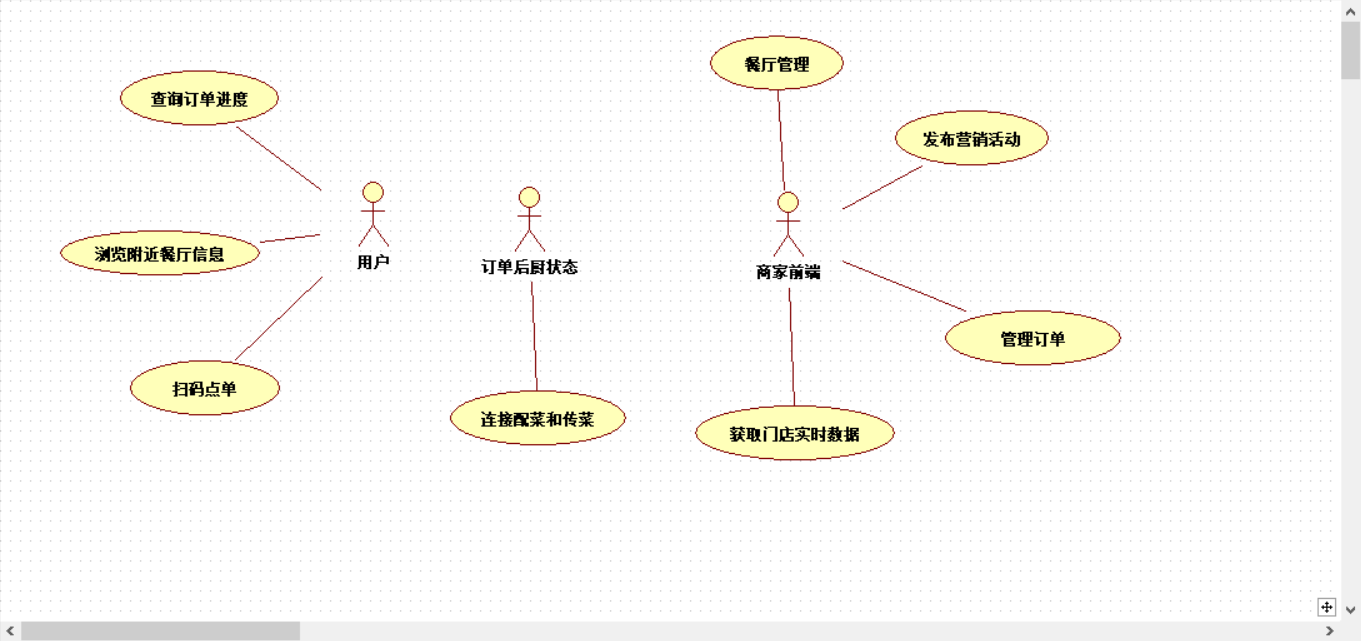


图1-1云快点用例图

### 1.3系统用例规约

* **1.3.1商家端口功能模块：**

1.餐厅管理功能用例规约

|  |  |  |
| --- | --- | --- |
| 用例名称 | 餐厅管理功能用例规约 | |
| 角色 | 商家管理员 | |
| 简要说明 | 该用例允许管理员修改餐厅名称、logo、营业时间、描述、餐厅地址 | |
| 前置条件 | 管理员成功注册登陆系统 | |
| 基本事件流 | 参与者操作 | 系统响应 |
| 1. 启动系统 2. 点击首页编辑店铺按 3. 填写餐厅信息 | 1. 显示系统界面 2. 接收指令，界面跳转 3. 检测信息充分性及合法性 |
| 异常事件流 | 参与者操作 | 系统响应 |
| 1. 必填字段信息为空 | 1. 保存失败，无法更新数据库 |
| 后置条件 | 餐厅信息修改成功，用户端餐厅信息同步更新 | |

2.菜单管理功能用例规约

|  |  |  |
| --- | --- | --- |
| 用例名称 | 菜单管理功能用例规约 | |
| 角色 | 商家管理员 | |
| 简要说明 | 该用例允许管理员对菜品进行分类以及增删改查操作 | |
| 前置条件 | 管理员成功注册登陆系统 | |
| 基本事件流 | 参与者操作 | 系统响应 |
| 1. 点击菜单管理 2. 点击添加类别按钮， 输入菜品具体信息 3. 点击排序按钮键 4. 点击删除菜单按钮 | 1. 显示菜单管理界面 2. 显示类别名，新增添加新菜品按钮 3. 接收指令，存入数据库 4. 根据字母大小自动排序 5. 接收指令，数据库清空该菜品数据 |
| 异常事件流 | 参与者操作 | 系统响应 |
| 1. 必填字段信息为空或超过字符段 | 1. 添加失败，无法更新数据库 |
| 后置条件 | 类别与具体菜品信息修改成功，用户端餐厅信息同步更新 | |

3.订单管理功能用例规约

|  |  |  |
| --- | --- | --- |
| 用例名称 | 订单管理功能用例规约 | |
| 角色 | 商家管理员 | |
| 简要说明 | 该用例允许管理员查看堂食以及外卖订单状态或删除订单 | |
| 前置条件 | 管理员成功注册登陆系统 | |
| 基本事件流 | 参与者操作 | 系统响应 |
| 1. 点击订单管理 2. 搜索查找订单 3. 点击删除菜单按钮 | 1. 显示订单管理界面 2. 通过字符匹配，显示被查找订单 3. 接收指令，数据库清空该订单信息 |
| 后置条件 | 订单状态完成或删除，后厨端订单信息同步更新 | |

4.点餐码功能用例规约

|  |  |  |
| --- | --- | --- |
| 用例名称 | 点餐码功能用例规约 | |
| 角色 | 商家管理员 | |
| 简要说明 | 该用例允许管理员一键增删单独点餐码，并可以下载该二维码图片 | |
| 前置条件 | 管理员成功注册登陆系统 | |
| 基本事件流 | 参与者操作 | 系统响应 |
| 1. 点击二维码管理 2. 输入桌号，点击生成二维码按钮 3. 点击二维码 4. 点击删除二维码按钮 | 1. 显示二维码管理界面 2. 根据桌号自动生成单独二维码显示在界面，二维码连接点餐界面 3. 页面跳转至图片保存界面 4. 接收指令，数据库清空该数据 |
| 异常事件流 | 参与者操作 | 系统响应 |
| 1. 输入桌号框内必填字段信息为空 | 1. 点击生成二维码按钮无效 |

* **1.3.2商家端口功能模块：**

1.后厨状态管理功能用例规约

|  |  |  |
| --- | --- | --- |
| 用例名称 | 后厨状态管理功能用例规约 | |
| 角色 | 商家后厨人员 | |
| 简要说明 | 该用例允许后厨人员查看并更改订单状态 | |
| 前置条件 | 1. 从管理员页面成功注册并绑定后厨人员账号 2. 后厨人员成功登陆系统，显示后厨管理界面 | |
| 基本事件流 | 参与者操作 | 系统响应 |
| 1. 搜索查找订单 2. 点击订单完成按钮 | 1. 通过字符匹配，显示被查找订单 2. 订单状态显示已完成 |
| 异常事件流 | 参与者操作 | 系统响应 |
| 1. 订单完成前，商家端删除订单 | 1. 完成订单指定无效 |
| 后置条件 | 订单状态显示完成，商家管理端订单信息同步更新 | |

* **1.3.3用户端口功能模块**

1.扫码在线点餐功能用例规约

|  |  |  |
| --- | --- | --- |
| 用例名称 | 餐厅管理功能用例规约 | |
| 角色 | 用户 | |
| 简要说明 | 该用例允许系统验证用户、管理员身份合法性后使用系统 | |
| 特殊需求 | 用户注册登录前也可浏览系统界面 | |
| 前置条件 | 用户、管理员进入各自端口界面 | |
| 基本事件流 | 参与者操作 | 系统响应 |
| 1. 扫描桌号二维码进入该店铺点菜界面 2. 选择菜品后点击结算按钮 3. 支付订单 | 1. 进入点菜界面，显示具体菜品 2. 界面跳转，进入确认订单界面 3. 传输订单数据至商家端 |
| 后置条件 | 用户完成订单操作，商家管理界面订单相关数据自动更新 | |

2.预订外卖功能用例规约

|  |  |  |
| --- | --- | --- |
| 用例名称 | 预订外卖功能用例规约 | |
| 角色 | 用户 | |
| 简要说明 | 该用例允许用户进入餐厅具体菜品页面点完餐后，选择预订外卖功能 | |
| 特殊需求 | 用户注册登录前也可浏览系统界面 | |
| 前置条件 | 用户成功注册登陆本系统 | |
| 基本事件流 | 参与者操作 | 系统响应 |
| 1. 点击点菜界面 2. 选择菜品后点击结算按钮 3. 订单类型可修改为外卖 4. 填写配送地址后支付订单 | 1. 界面跳转，显示具体菜品 2. 界面跳转，进入确认订单界面 3. 接受指令，显示地址填写框 4. 传输订单数据至商家端 |
| 异常事件流 | 参与者操作 | 系统响应 |
| 1. 地址填写框内必填字段信息为空 | 1. 点击支付按钮无效 |
| 后置条件 | 用户完成订单操作，商家管理界面订单相关数据自动更新 | |

### 1.4功能结构图

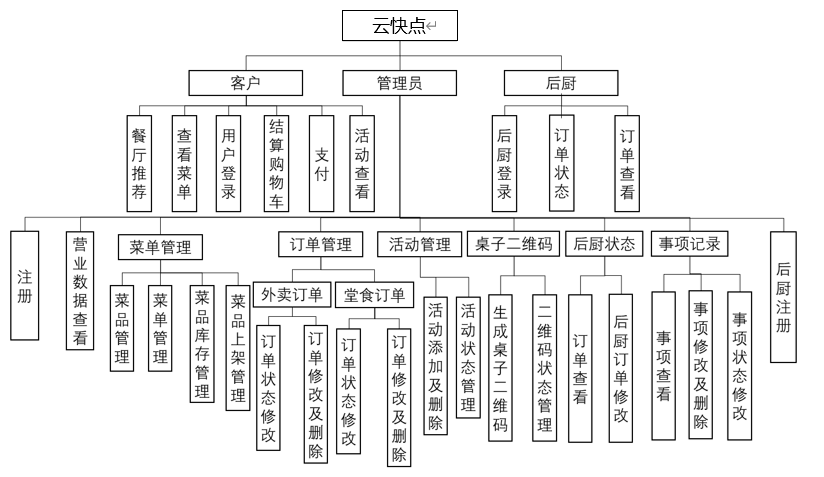


图1-2 微服务集群架构图

### 2.架构设计

### 2.1架构描述

为结合需求，本项目使用MVVM模式进行设计，数据上的交换格式为JSON，后台采取分布式微服务集群架构。

### 2.2架构图

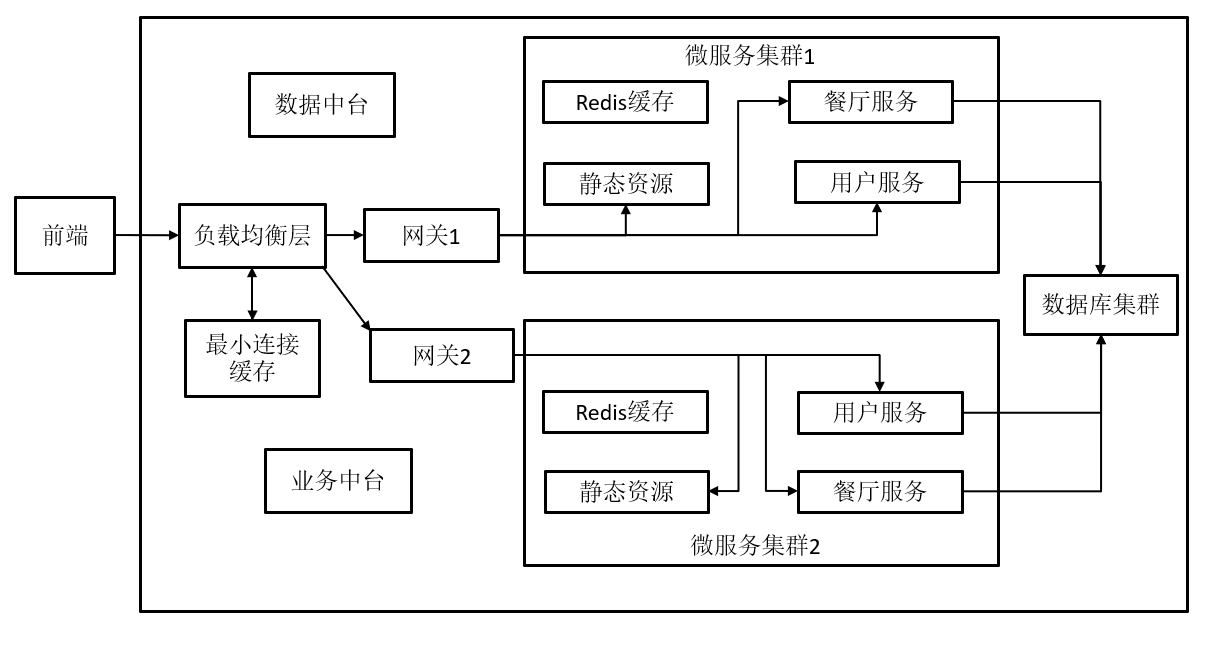


图2-1 微服务集群架构图

### 2.3系统关键抽象

项目组根据架构进行了的析取。本系统架构是MVVM，因此，本系统存在三种类型的类，即为：边界类、控制类、数据类，分布属于MVVM各层。

数据类的析取为关键抽象。

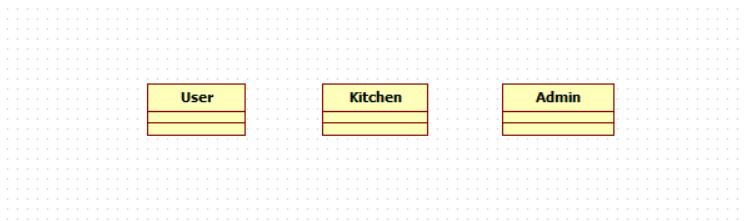
经过本项目组的分析，得到系统关键抽象如图2-2所示。

图2-2 关键抽象图

## 3.类的设计

本项目数据采取开源数据库mysql作为主要存储，使用redis非关系形数据库进行缓存数据存储，架构上支持分布式部署，允许随时扩容处理。

项目主要涉及9个数据表，数据表的定义以及之间的关系如图3-1所示：

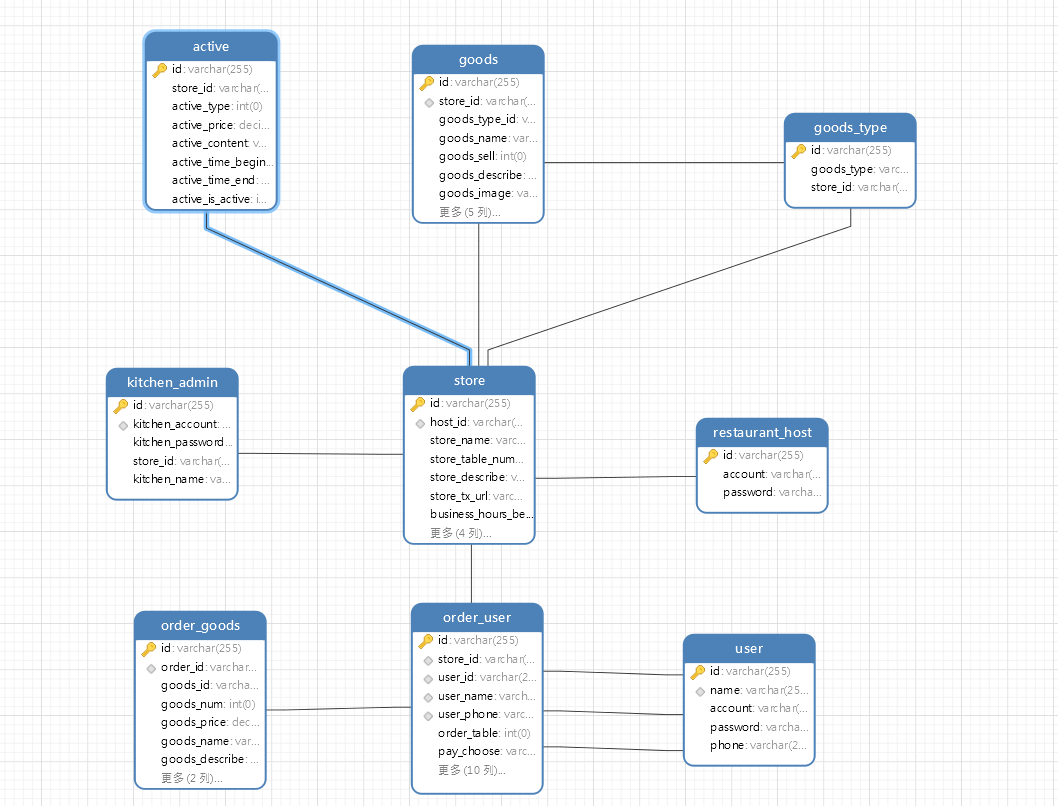


图3-1 类图

以下为各个表之间的关联信息：

表3-1数据库表关联描述

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | 字段 | 关联表 | 关联字段 | 删除时操作 |
| Active | Store\_id | Store | id | RESTRICT |
| Goods | Store\_id | Store | id | RESTRICT |
| Goods | Goods\_type\_id | Goods\_Type | id | RESTRICT |
| Goods\_type | Store\_id | Store | Id | RESTRICT |
| Kitchen\_admin | Store\_id | Store | Id | RESTRICT |
| Store | Host\_id | Restaurant\_host | Id | RESTRICT |
| Order\_goods | Order\_id | Order\_user | Id | RESTRICT |
| Order\_user | Store\_id | Store | Id | RESTRICT |
| Order\_user | User\_id | User | Id | RESTRICT |
| Order\_user | User\_name | User | Name | RESTRICT |
| Order\_user | User\_phone | User | phone | RESTRICT |

以下为数据库中各表的含义：

表3-2数据库表的描述

|  |  |
| --- | --- |
| 表名 | 描述 |
| Active | 活动表，包含活动的各种信息 |
| Goods | 菜品表，包含各个店铺的菜品信息 |
| Goods\_type | 菜品类别表，包含各个店铺的菜品类别信息 |
| Kitchen\_admin | 后厨管理员信息表，包含后厨信息 |
| Order\_goods | 订单内容表，包含各个订单的菜品内容 |
| Order\_user | 订单信息表，包含各个用户的订单信息 |
| Restaurant\_host | 前台管理员信息表，包含前台管理员信息 |
| Store | 店铺信息表，包含店铺信息 |
| User | 用户表，包含各个店铺的用户信息 |

3.1.2 数据访问设计

本项目使用设计模式MVVM进行设计，数据上的交换格式为JSON，后台采取分布式微服务架构，分别分为以下3个微服务：

表3-3微服务信息表

|  |  |  |
| --- | --- | --- |
| 微服务名称 | 微服务描述 | 微服务路径 |
| User | 处理用户逻辑 | ip/user/\* |
| Kitchen | 处理后厨逻辑 | ip/admin/kitchen/\* |
| Admin | 处理管理员逻辑 | ip/admin/\* |

以下为各个微服务的接口信息：

表3-4 Admin服务接口信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 接口名称 | 接口描述 | 接口路径 | 调用方式 |
| 1 | test | 测试接口 | ip/admin/test | Any |
| 2 | addHost | 添加管理员 | ip/admin /addHost | POST |
| 3 | addStore | 添加店铺 | ip/admin /addStore | POST |
| 4 | loginAdmin | 管理员登录 | ip/admin /loginAdmin | POST |
| 5 | getStore | 获取店铺信息 | ip/admin /getStore | GET |
| 6 | editStore | 编辑店铺信息 | ip/admin /editStore | POST |
| 7 | getGoods | 获取菜品信息 | ip/admin /getGoods | GET |
| 8 | addGoodsType | 获取菜单分类信息 | ip/admin /addGoodsType | POST |
| 9 | addGoodsType | 添加菜单分类信息 | ip/admin /addGoods | POST |
| 10 | deleteGoodsType | 深处菜单分类 | ip/admin /deleteGoodsType | GET |
| 11 | deleteGoods | 删除菜品 | ip/admin /deleteGoods | GET |
| 12 | editGoodsType | 编辑菜单分类信息 | ip/admin /editGoodsType | POST |
| 13 | editGoods | 编辑菜品 | ip/admin /editGoods | POST |
| 14 | getOrderAdmin | 获取订单 | ip/admin /getOrderAdmin | GET |
| 15 | deleteOrder | 删除订单 | ip/admin /deleteOrder | GET |
| 16 | editOrderGoodsStatus | 编辑订单菜品状态 | ip/admin /editOrderGoodsStatus | POST |
| 17 | editOrderStatus | 编辑订单状态 | ip/admin /editOrderStatus | POST |
| 18 | editAdminPassword | 更改管理员密码 | ip/admin /editAdminPassword | POST |
| 19 | addActive | 添加活动 | ip/admin /addActive | POST |
| 20 | getActive | 获取活动 | ip/admin /getActive | GET |
| 21 | deleteActive | 删除活动 | ip/admin / /deleteActive | GET |
| 22 | editActive | 编辑活动 | ip/admin /editActive | POST |

表3-5 User服务接口信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 接口名称 | 接口描述 | 接口路径 | 调用方式 |
| 1 | addUser | 添加用户 | /addUser | POST |
| 2 | loginUser | 用户登录 | /loginUser | POST |
| 3 | test | 测试接口 | /test | GET |
| 4 | getStoreList | 获取店铺菜单 | /getStoreList | GET |
| 5 | getStore | 获店铺信息 | /getStore | GET |
| 6 | getGoods | 获取菜单信息 | /getGoods | GET |
| 7 | getActiveUser | 获取活动信息 | /getActiveUser | GET |
| 8 | addOrder | 添加订单 | /order/addOrder | POST |
| 9 | getOrder | 获取订单 | /order/getOrderUser | GET |

表3-6 kitchen服务接口信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 接口名称 | 接口描述 | 接口路径 | 调用方式 |
| 1 | addKitchen | 添加后厨 | /addKitchen | POST |
| 2 | loginAdmin | 后厨登录 | /loginKitchenAdmin | POST |

## 4.页面设计

**4.1 商家端口界面**

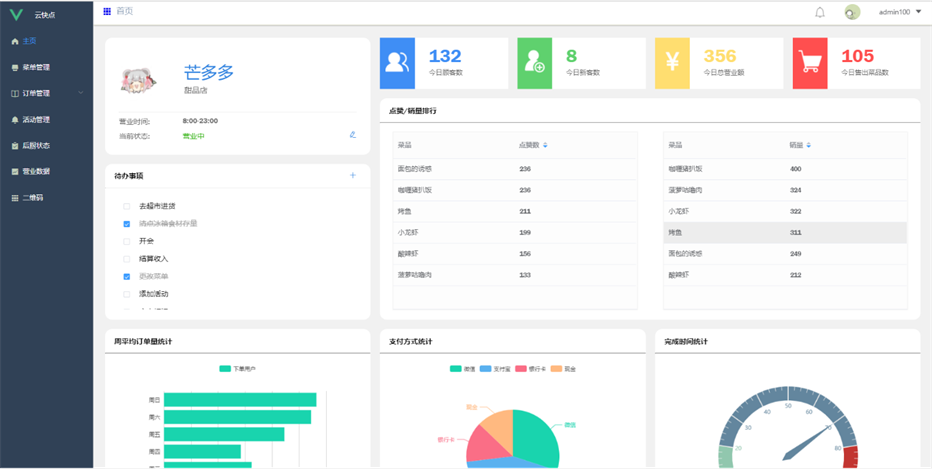
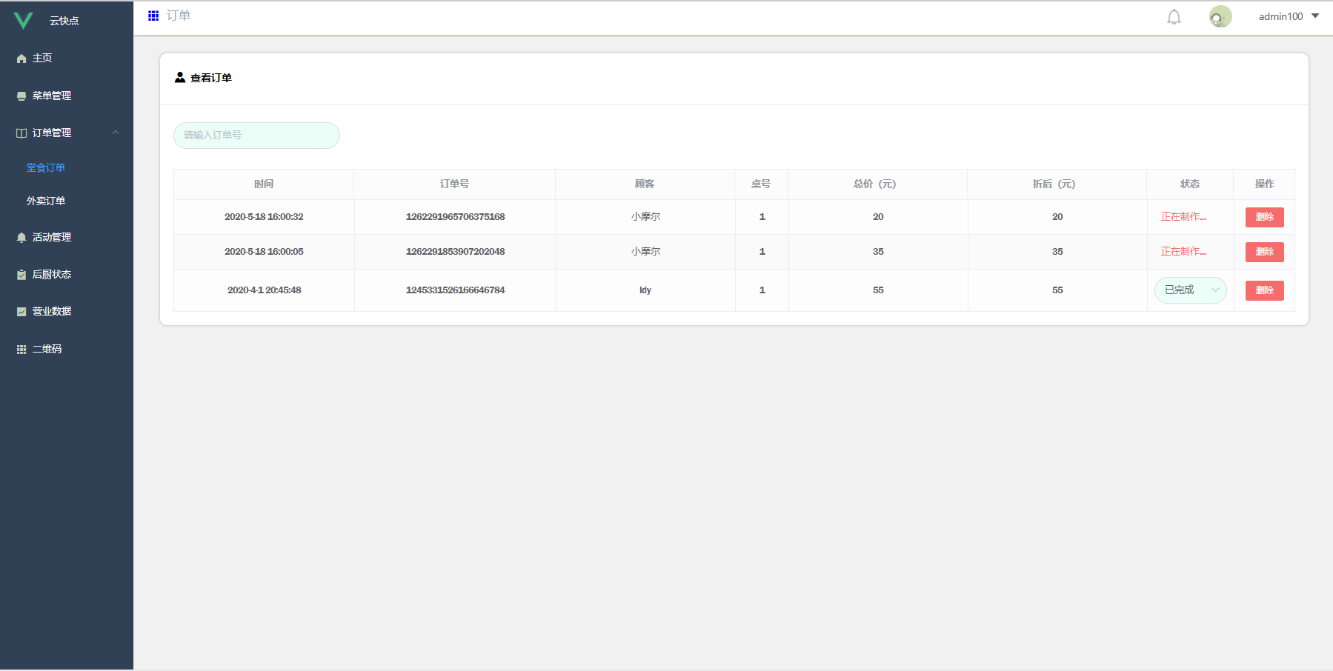
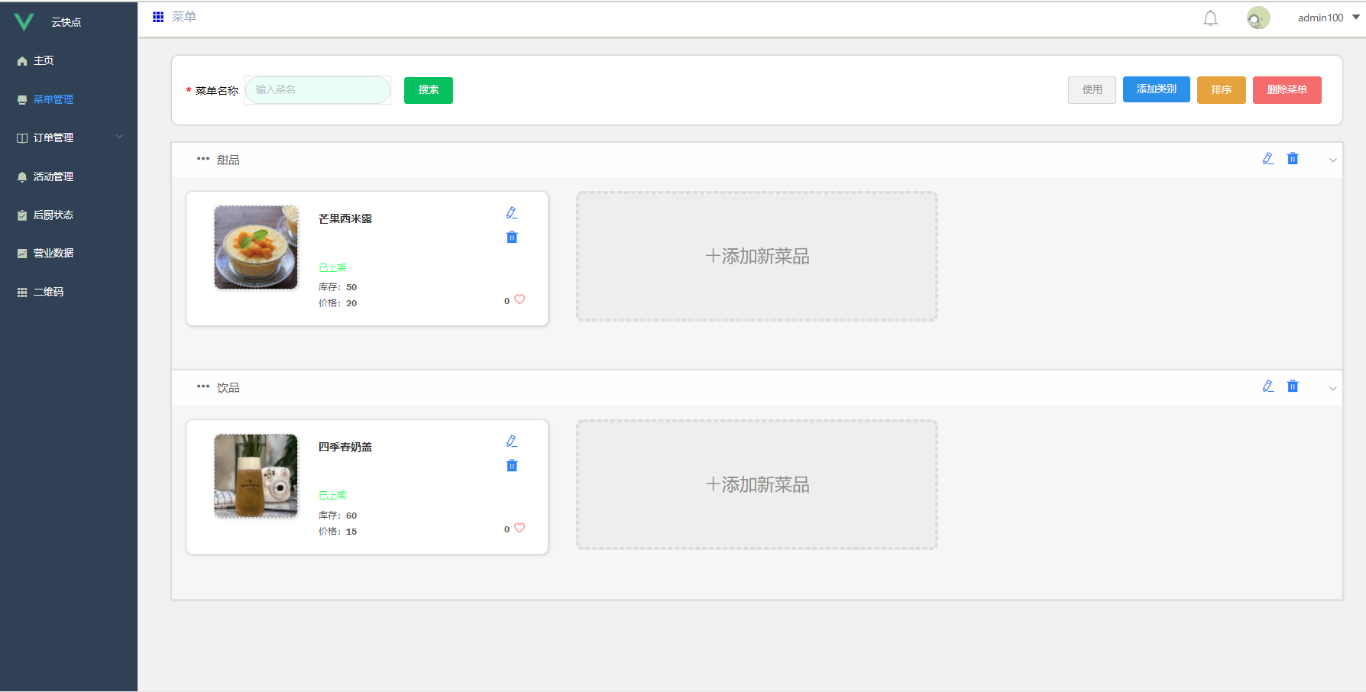


图4-1 首页界面

图4-2 菜单管理界面

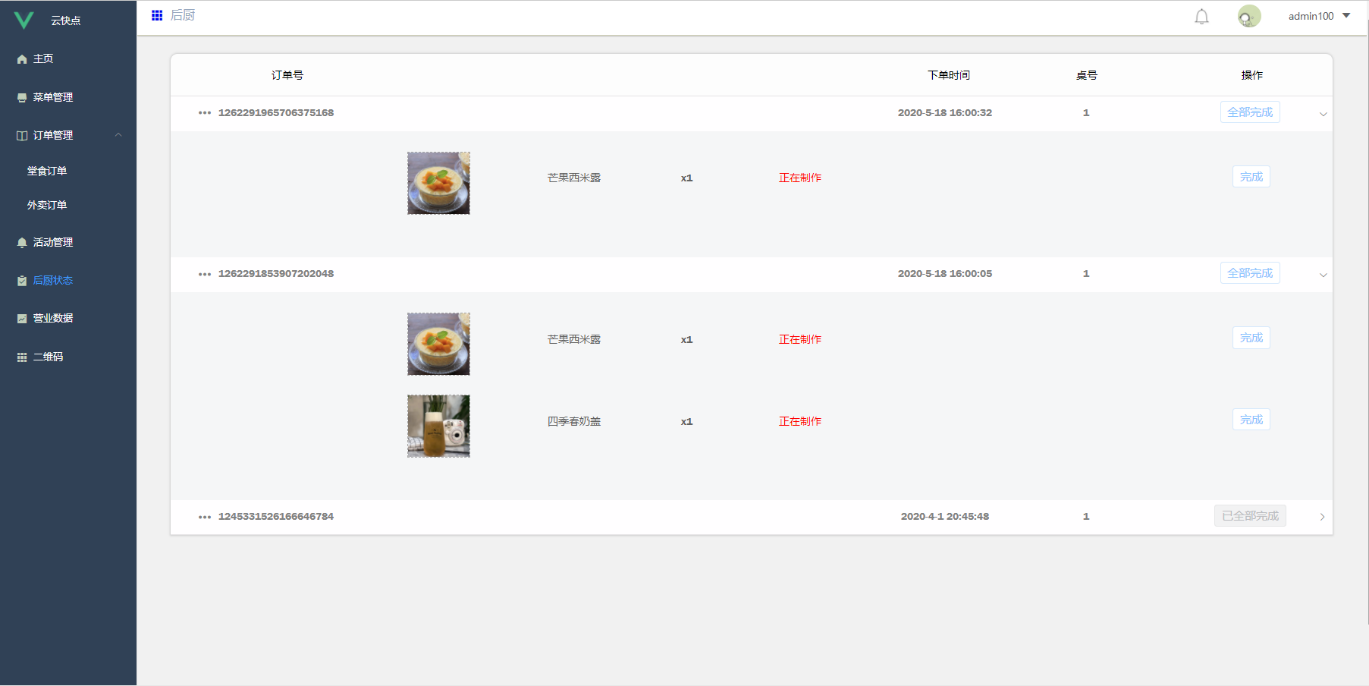
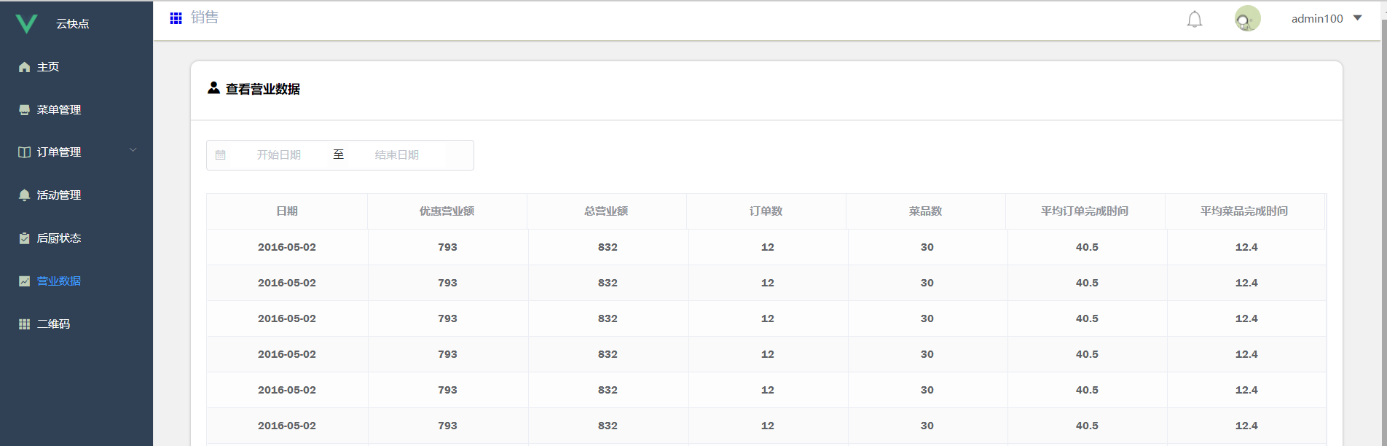
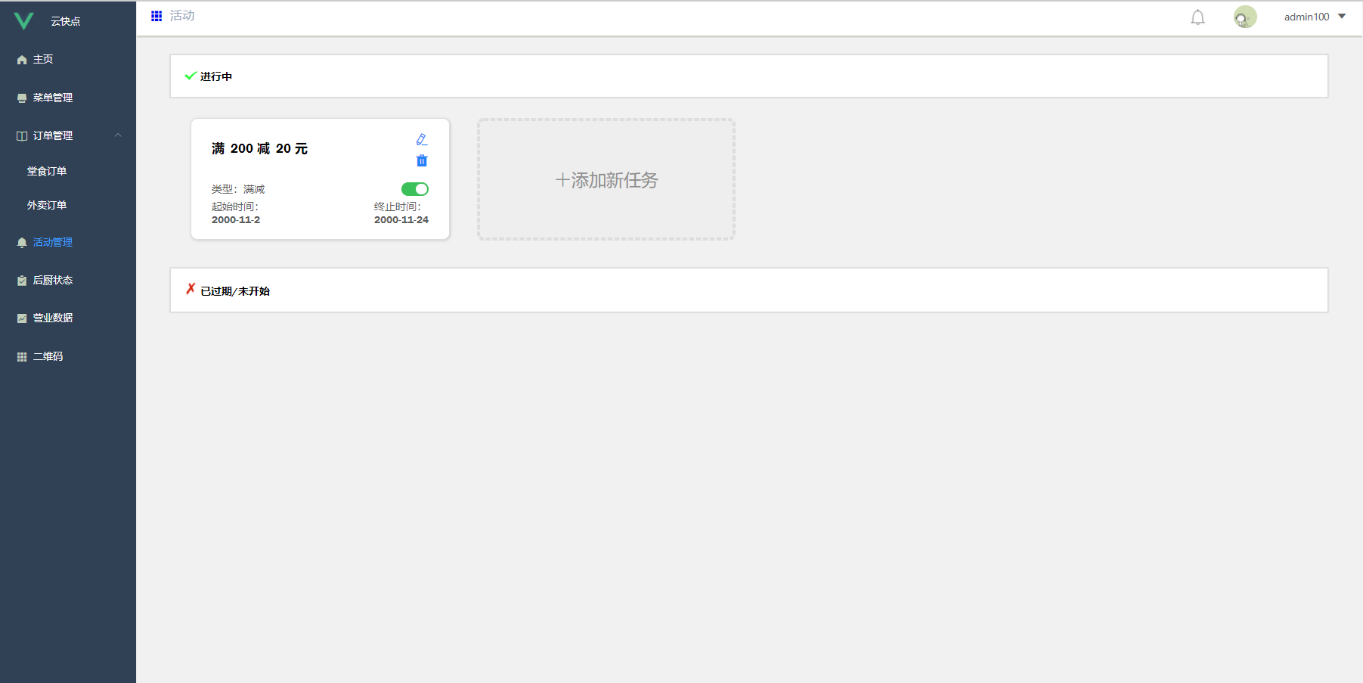
图4-3 订单管理界面

图4-4 后厨状态管理界面

图4-5 活动查看界面

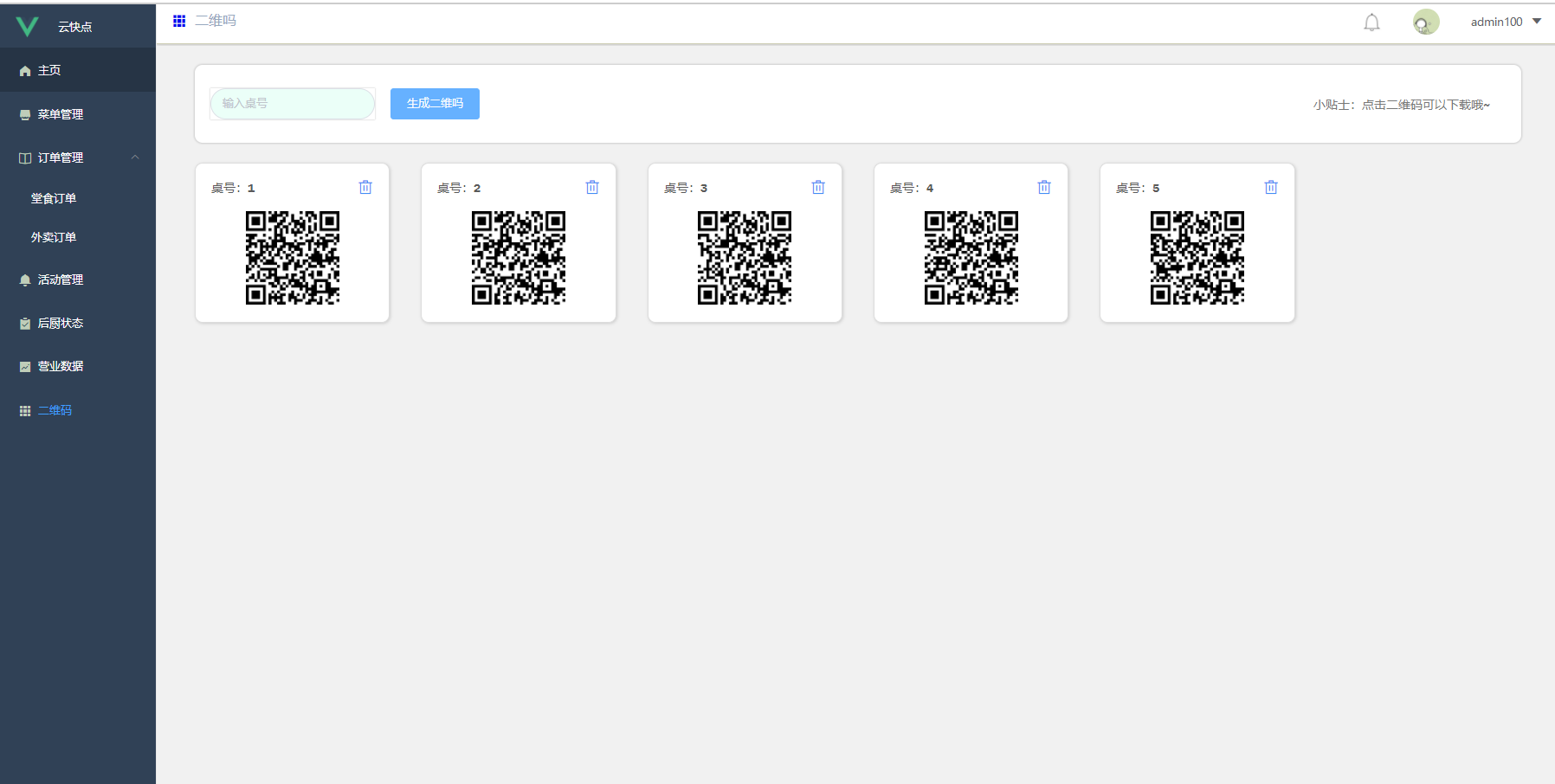
图4-6 营业数据查看界面

图4-7 二维码管理界面

**4.2后厨端口界面**

图4-8 后厨管理界面

**4.3用户端口界面**

　　　　图4-9 用户首页界面　　　　　　　　　　图4-10 用户登陆界面

　　　　图4-11 用户点菜界面　　　　 　　　　　　图4-12 订单结算界面

## 5.测试报告

测试过程中，保证测试的预备工作完成，用户端/商家端模拟点餐进行下单和接单。

测试主要从平均事务响应时间、每秒点击数、业务成功率、系统资源、并发数、Web服务器资源、数据库服务器资源等几个方面分析。

测试要求是验证在20分钟内完成200次用户登录系统，然后进行订单及接单业务，最后退出，在业务操作过程中页面的响应时间不超过3秒，并且服务器的CPU使用率、内存使用率分别不超过75%、70%。结果如下：

1.事务：测试结果无异常。

2.HTTP响应：“HTTP 200”表示请求被正确响应，“HTTP 404”表示文件或者目录未能找到。经过发出的请求大部分都能正确响应，有少部分失败了，但未影响测试结果。

3.并发数：根据Vuser的数量对事务响应时间产生的影响，可以得到系统业务性能测试过程中运行情况，按照预期的设置运行，没有Vuser出现运行错误。

4.登录：系统提交对应Action平均响应时间分别是1.512秒与0.912秒，两个数值均达到要求。

5.每秒点击数：测试结果是“每秒点击数”与“Average Throughput (bytes/second)”不一致，“每秒点击数”不正常。

分析及改进：出现该结果可能是由网络引起的客户端存在问题，未能正确模拟用户的行为。经调整重新测试，测试结果基本一致，服务器能及时接收客户端请求，并能返回结果。

6.业务成功率：事务通过数为252，测试业务成功率为100%。

7.Web服务器资源：配置Tomcat时共分配了100M左右的物理内存，测试初期使用的JVM相对来说较少，测试场景执行期间，报出“Out Of Memory”内存溢出的错误。

分析及改进：原因：Tomcat的内存分配不足。解决方法：为Tomcat分配更多的内存。

8.数据库服务器：测试在页面访问、功能执行方面是没有问题的。

以上测试结果及指标表明，该系统在运行速度、安全性、扩展性、部署方便和可用性方面，都有比较好的性能。

## 6.技术难点

本项目使用了 Sringboot+vue 的微服务结构。开发过程中，主要遇到三个技术难点， 具体及解决方法如下：

1.用户流量爆发相对集中

高峰期用户数量快速增长，可通过部署多个微服务设置负载均衡进行解决，流量低峰 时使用轮询法或随机法的策略进行请求均衡，当数据到达一定程度时可更换为最小连接发，建立最小连接缓存，减少询问所需要的延迟，进一步提升系统吞吐量。还可以将复杂 请求与简单请求通过网关进行分离，设置单独静态文件服务器等解决。除此之外，还可以通过构建数据中台，对数据进行清洗、分析，让我们的系统更加智慧，构建业务中台让我们能够更方便的对数据进行管理，让我们的系统更加敏捷。

2.数据库对上亿数据的兼容问题

由于存在商家超过一定数量以及由于时间积累导致数据库数据过大的可能性，解决方 案为有：使用正确的存储引擎，将经常使用事务的数据表设定引擎InnoDB，不经常使用事 务的设定为MyISAM等，另外还可以通过建立索引，分库分表等方式提升数据库的吞吐量，以下为乐观锁的数据库集群：

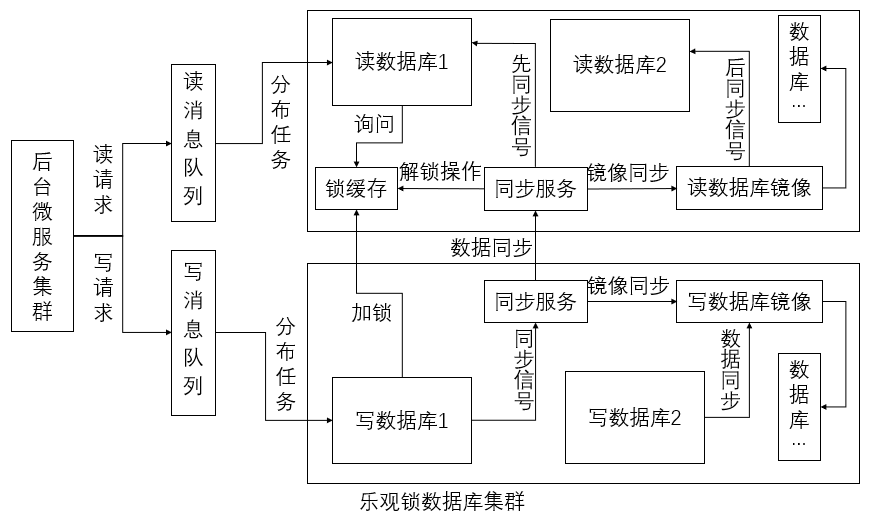


图5-1 乐观锁数据库集群

## 7.活动顺序图

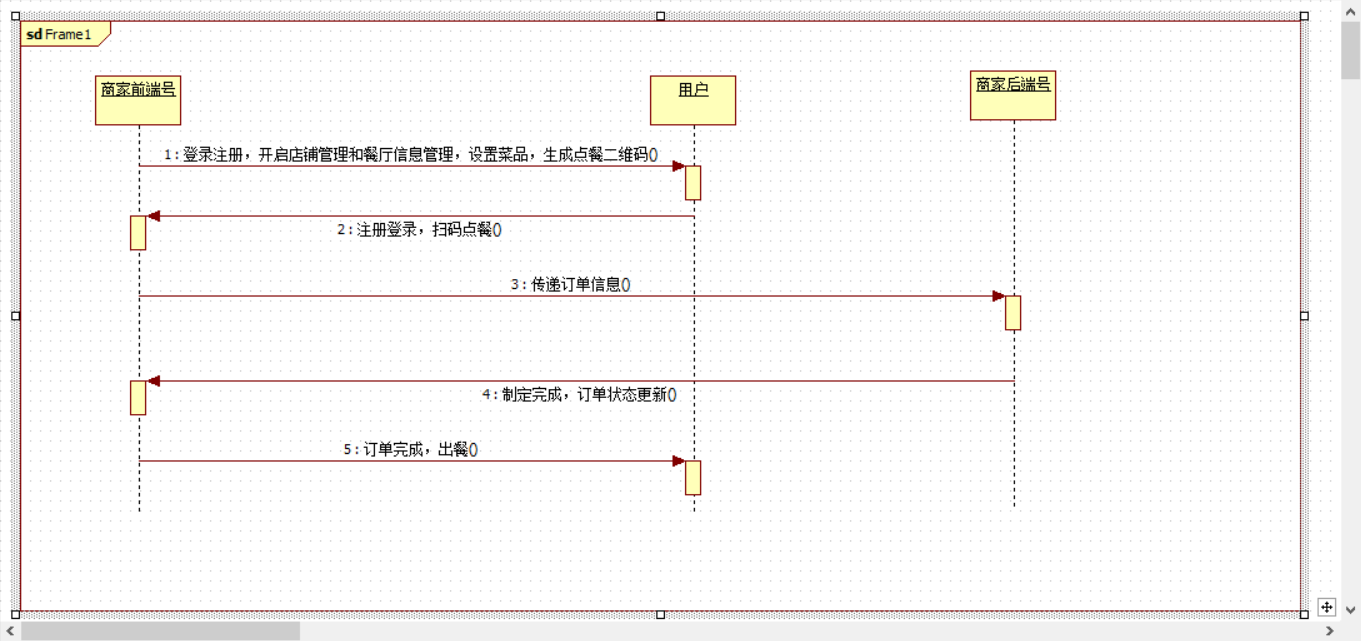


图7-1 活动顺序图

## 8.安装及使用

本项目为web服务，无需安装，只需用户打开浏览器输入网址即可使用，手机端也可通过扫描二维码进行使用，网址如下：

商家前台网址：<http://haozheng.club/restaurant_computer_p>

商家后厨网址：<http://kitchen.haozheng.club/restaurant_computer_k>

手机点餐网址：<http://user.haozheng.club/restaurant_m>

主要功能使用流程图如下：

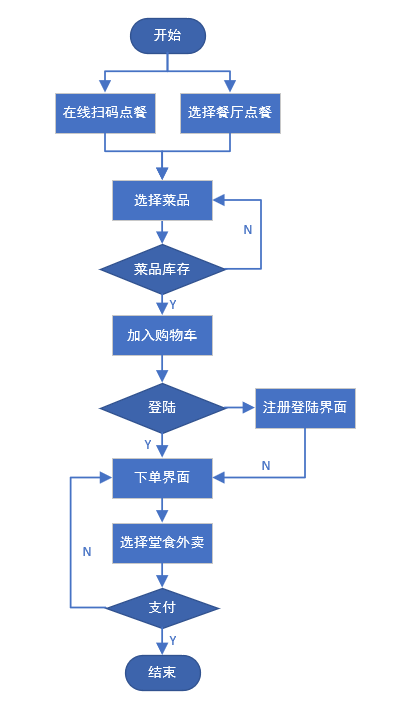
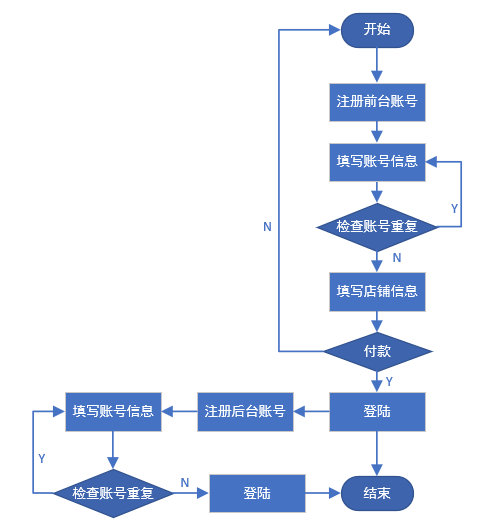
 

图8-1 用户下单流程图 图8-2 商家注册登陆流程图

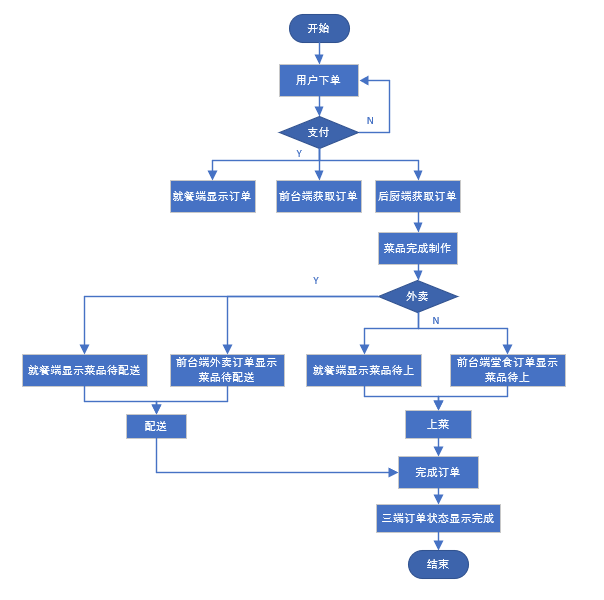


图8-3 订单完成流程图