

课程设计最终报告

姓名： 陈星棋

学号： 55180422

目录

| | |
|-------------------------------|----|
| 一. 开发环境..... | 3 |
| 二. 需求分析..... | 3 |
| 三. 设计..... | 15 |
| 四. 实现..... | 27 |
| 五. 主要难点重点问题讨论..... | 31 |
| 六. 与商用微商系统的差距..... | 35 |
| 七. 重新开发会采取的措施..... | 35 |
| 八. 获得的提高..... | 36 |
| 九. 缺憾..... | 36 |
| 十. C/S 开发模式与 B/S 开发模式的比较..... | 37 |

一. 开发环境

(1) C/S 模式

QT 5.12.0 MinGW 64-bit

使用语言:

C++

(2) B/S 模式

Apache + Thinkphp5 框架

使用语言:

前端: css + javascript (jquery)+ html

后端: php(TP5)+workerman 框架

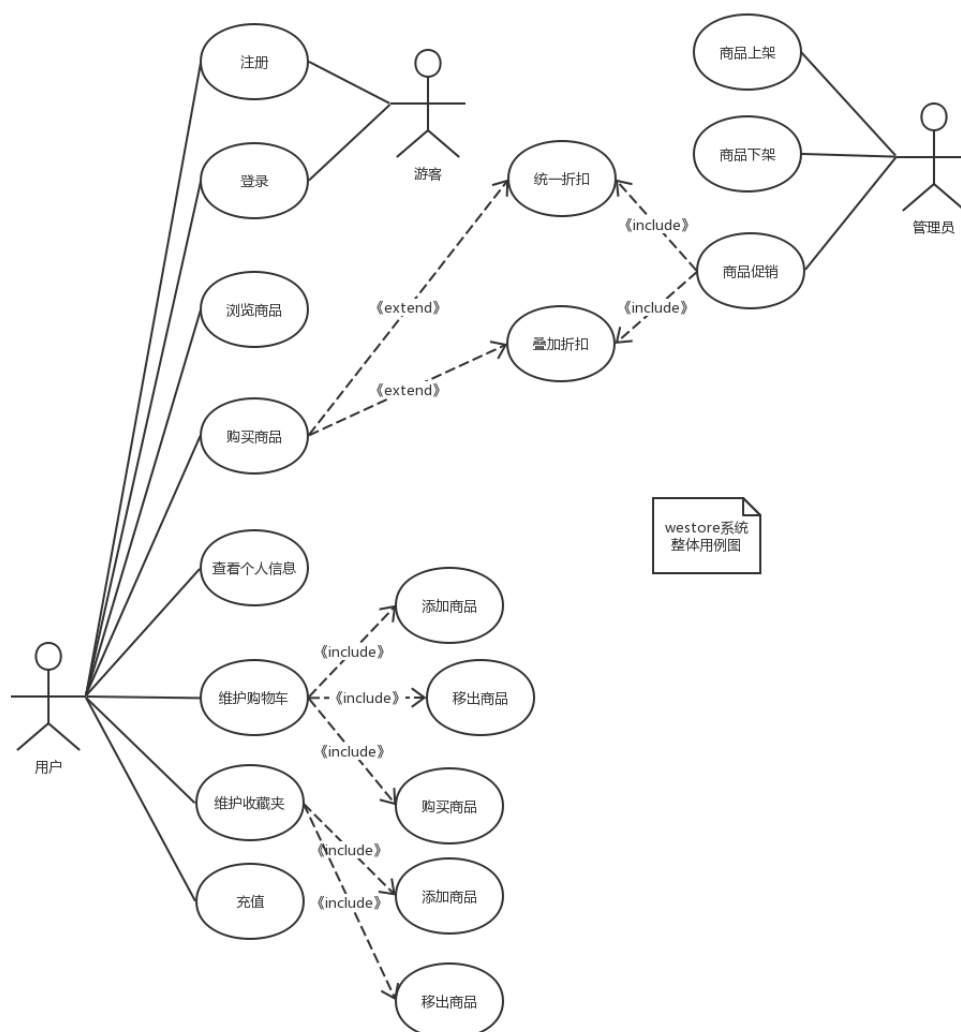
数据库: mysql

缓存: redis

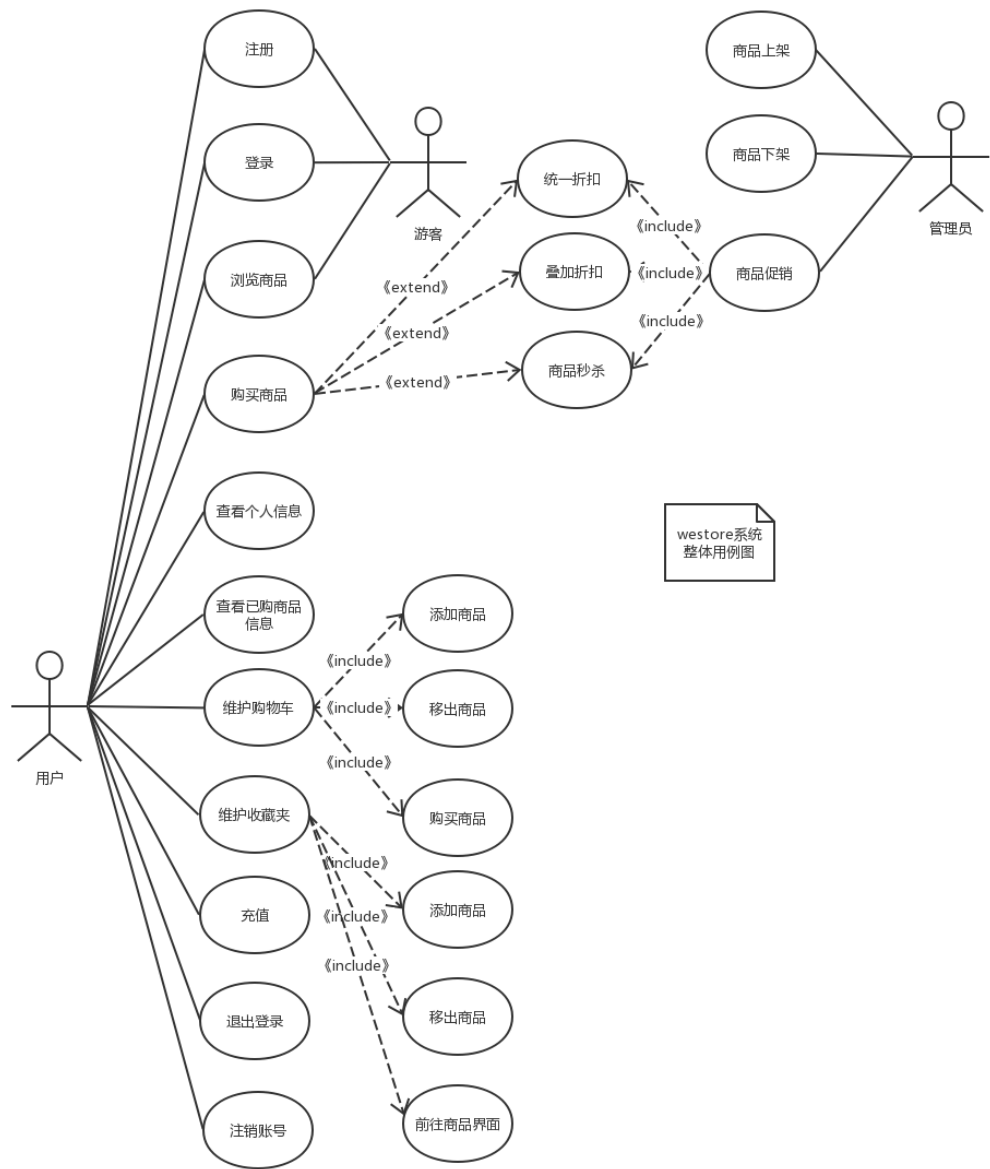
二. 需求分析

1. 系统整体用例图

(1) C/S 模式



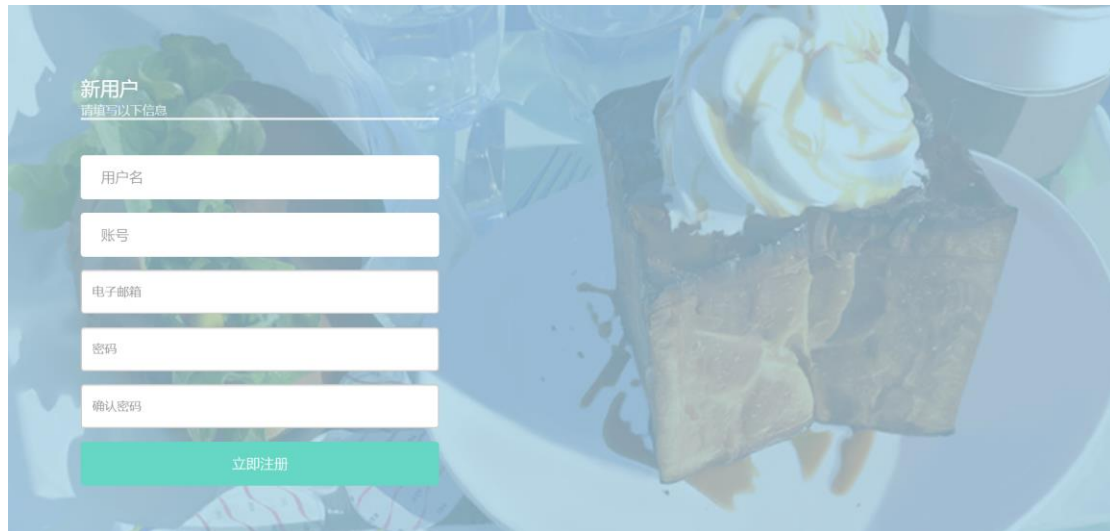
(2) B/S 模式



2. 功能描述(基于 B/S 模式)

用户端：

(1) 注册

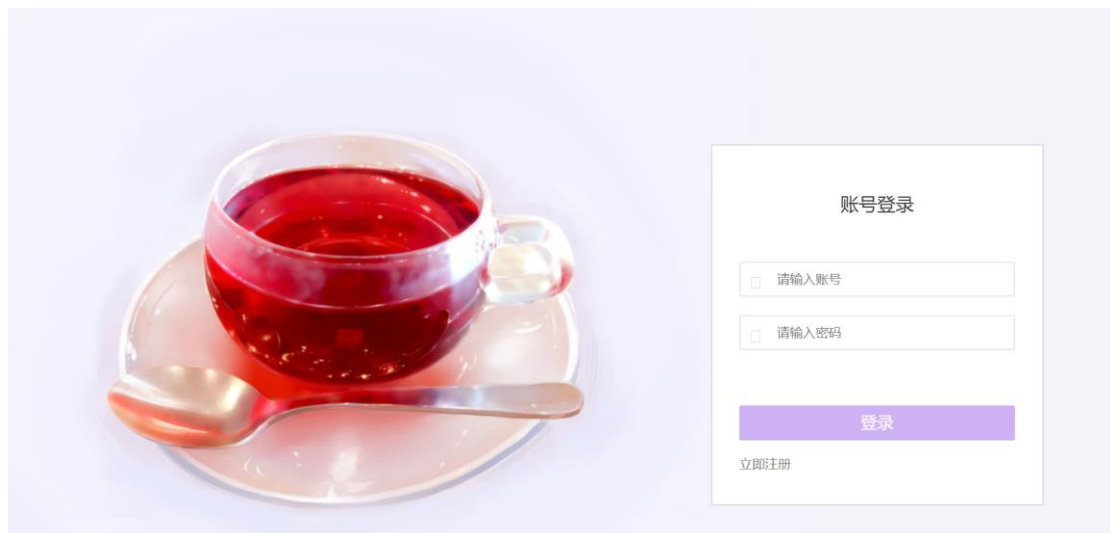
A screenshot of a web registration form titled "新用户" (New User) with the subtitle "请填写以下信息" (Please fill in the following information). The form is overlaid on a background image of a chocolate cake with white cream. It contains five input fields: "用户名" (Username), "账号" (Account), "电子邮箱" (Email), "密码" (Password), and "确认密码" (Confirm Password). Below the fields is a green button labeled "立即注册" (Register Now).

1. 表单限制：

- a. 账号：全数字且长度不少于 6 位
- b. 用户名：不包含空格
- c. 电子邮箱：符合电子邮箱格式
- e. 密码：长度不少于 7 位
- f. 确认密码：与密码一致

2. 立即注册：检测表单填写是否符合规范，若符合规范则发送至服务器对用户名与账号的唯一性进行审核，若全部符合规则则显示注册成功并自动跳转至主界面，否则显示错误信息。

(2) 登录

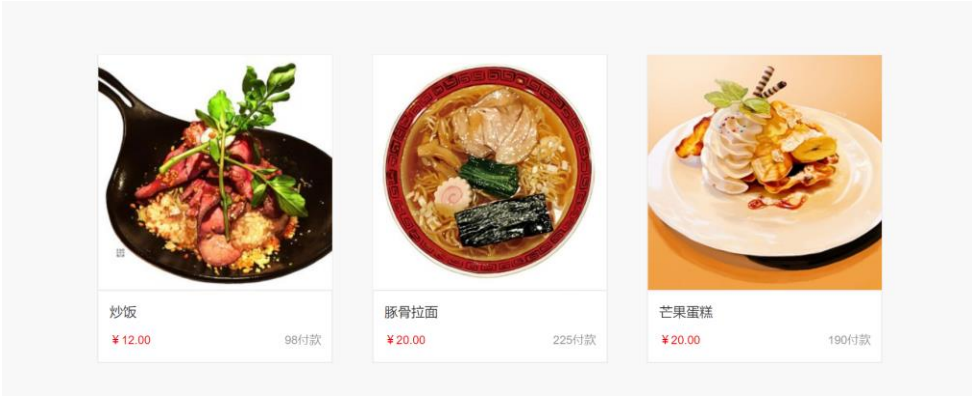
A screenshot of a web login form titled "账号登录" (Account Login). The form is overlaid on a background image of a cup of red liquid. It contains two input fields: "请输入账号" (Please enter account) and "请输入密码" (Please enter password). Below the fields is a purple button labeled "登录" (Login). At the bottom left of the form is a link labeled "立即注册" (Register Now).

1. 账号：输入账号

2. 密码：输入密码

3. 登录：将账号与密码发送至服务器端进行审核，正确则进入主界面，否则发送错误信息。

(3) 浏览商品



图片：游客点击图片跳转至登录界面，用户点击图片跳转至商品购买界面

(4) 购买商品



1. 自动弹出：若商品在叠加折扣列表或统一折扣列表中则进入商品购买界面时自动弹出商品折扣信息。
2. 加减号：增减购买商品数量，当数量达到剩余量时则无法继续增加。
3. 立即购买：弹出密码输入框，若密码输入正确且商品数量足够则显示购买成功，否则显示密码错误信息。
4. 加入购物车：商品加入购物车，若已存在则显示已存在
5. 收藏：商品加入收藏夹，若已存在则显示已存在

(5) 查看个人信息







1. 图片：点击图片则进入文件选择，文件限制：必须为 jpg/png 且文件大小小于 1MB 的文件格式，符合规则则实时更换头像，否则提示错误信息。
2. 已购商品：进入已购商品界面
3. 余额：进入充值界面
4. 购物车：进入购物车界面
5. 收藏夹：进入收藏夹界面
6. 退出登录：清除用户存储的 cookie 并跳转至登录界面
7. 注销账号：用户在弹出的输入框中输入密码，若密码正确则清除用户所有信息，否则提示错误信息。

(6) 查看已购商品信息



| 商品 | 购买时间 |
|---|---------------------|
|  烧鱼 | 2019-11-08 09:09:12 |
|  鲷鱼烧 | 2019-11-08 09:08:52 |

(7) 维护购物车

| | | | | | | | |
|---|---|------|-------|-------|------------------|-------|----|
|  |  | 黄油蛋糕 | 23.00 | 20.70 | <div>- 1 +</div> | 20.70 | 删除 |
|  |  | 炒饭 | 12.00 | 10.80 | <div>- 1 +</div> | 10.80 | 删除 |
| | | | | | | 应付: 0 | 结算 |

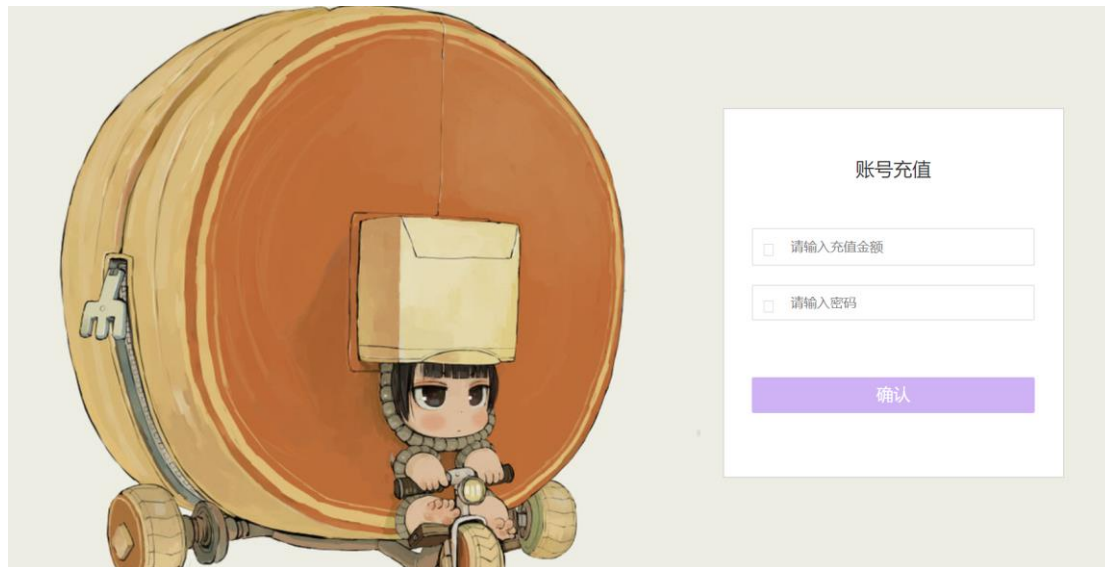
1. 选择框：代表已选择购买此商品
2. 加减号：增加购买商品数量，且会更加商品折扣信息实时更改商品折后价
3. 删除：将商品移出购物车
4. 结算：若在弹出的输入框中输入的密码与保存在 cookie 内的密码一致则购买已选择商品，并在成功购买后将商品移出购物车

(8) 维护收藏夹

| 商品 | 单价 | 操作 |
|---|---------------|----|
|  | 芒果蛋糕 18.00 | 删除 |
|  | 黄油蛋糕 20.70 | 删除 |

1. 图片：点击图片，若此商品未下架则跳转至商品购买界面，否则提示商品已下架
2. 删除：将商品移出收藏夹

(9) 充值：



1. 表单限制：

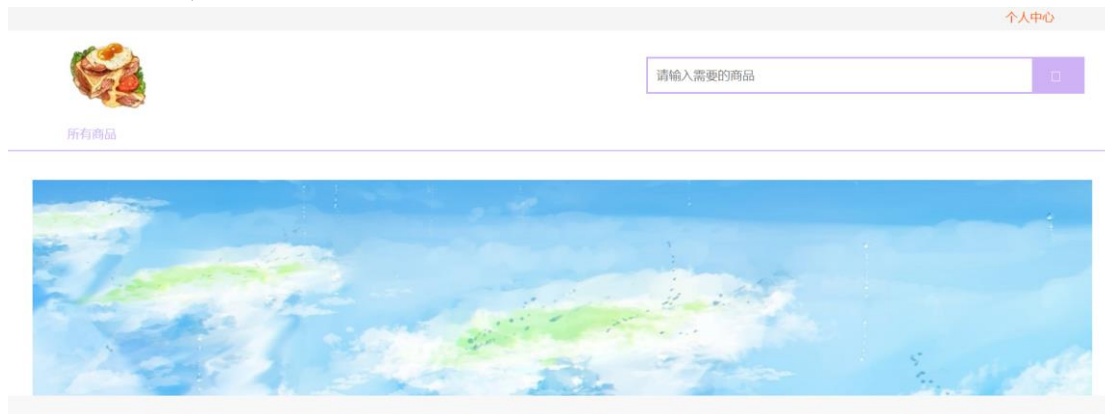
- a. 充值金额：只可输入浮点数或整数且总金额不能超过 50 万
- b. 密码：输入登录密码

2. 确认：

若表单填写符合要求则显示充值成功并自动跳转至个人信息界面，否则显示错误信息并自动刷新此页面。

(10) 商品秒杀

1. 当天无秒杀商品



图片：点击图片无任何弹出

2. 当天存在秒杀商品



图片：点击图片进入商品秒杀界面



3. 秒杀界面



立即秒杀：若商品数量足够则显示秒杀成功，否则显示秒杀已结束并自动跳转至主界面

管理员端：

(1) 商品界面

| | | | | | |
|---|------|-------|---|-----|----|
|  | 黄油蛋糕 | 23.00 | 0 | 109 | 下架 |
|  | 炒饭 | 12.00 | 0 | 98 | 下架 |

上架新商品

1. 图片：弹出增加商品的数量信息
2. 下架：将商品下架
3. 上架新商品：进入商品上架界面

(2) 商品上架

上架商品
请填写以下信息

商品名称

 点击上传商品图片

销售单价

上架数量

确认上架



1. 表单限制：
 - a. 商品名称：输入商品名称
 - b. 图片：点击图片，若商品名称已填写则进入文件选择，否则提示输入商品名称，文件限制：必须为 jpg/png 且文件大小小于 1MB。在上传图片后立即对商品名称进行检测，若重名则通知客户更改商品名称。
 - c. 销售单价：只能输入浮点数或整数
 - d. 上架数量：只能输入整数且数量必须大于 0
2. 确认上架：若表单信息填写符合规则则将表单提交至服务器并提示商品上架成功。

(3) 商品促销

| 促销类别 | 具体方案 | 操作 |
|------|------|----|
| 折扣 | 统一折扣 | 管理 |
| 折扣 | 叠加折扣 | 管理 |
| 秒杀 | 定点秒杀 | 管理 |



- 1. 统一折扣管理：进入统一折扣管理界面
- 2. 叠加折扣管理：进入叠加折扣管理界面
- 3. 定点秒杀管理：进入定点秒杀管理界面

(4) 统一折扣管理：

| | | | | | |
|---|-----|-------|------|------|----|
|  | 炒饭 | 12.00 | 10.8 | 0.90 | 修改 |
|  | 鸡蛋糕 | 20.00 | 18 | 0.90 | 修改 |
| | | | | | 增加 |

- 1. 修改：弹出折扣输入框，只可输入 $0 < x \leq 1$ 内的浮点数，若输入 1 则将商品移出统一折扣
- 2. 添加：弹出商品名称输入框，管理员输入商品名称，若商品存在且不在统一折扣列表内则显示添加成功，否则显示错误信息。

(5) 叠加折扣管理：

| | | | | | | |
|---|------|-------|------------|------------|------------|----|
|  | 芒果蛋糕 | 20.00 | 0.90 18 | 1.00 20 | 1.00 20 | 管理 |
|  | 牛排 | 40.00 | 0.90 36 | 0.50 20 | 0.20 8 | 管理 |
| | | | | | | 增加 |

1. 管理：弹出折扣信息输入框，用户输入一、二、三件折扣信息，若全为 1 则将商品移出叠加折扣
2. 添加：弹出商品名称输入框，管理员输入商品名称，若商品存在且不在叠加折扣列表内则显示添加成功，否则显示错误信息。

(6) 秒杀管理：

| 商品 | 秒杀价 | 数量 | 秒杀时间 | 操作 |
|--|-----|----|---------------------|----|
|  烧鱼 | 1 | 1 | 2019-11-09 10:26:01 | 管理 |
|  牛排 | 1 | 1 | 2019-11-10 10:27:56 | 管理 |
| | | | | 增加 |

1. 管理：弹出如下表单

请填写以下信息

秒杀价

数量

年 / 月 / 日 --:--

确定

取消

表单限制：

a. 秒杀价：只允许输入整数

b. 数量：只允许输入整数且数量不小于当前商品数量，若数量为 0 且当日不为此商品的秒杀时间则将商品移出秒杀列表。

c. 秒杀开始时间：

(1) 一天内只能进行一次秒杀，每次秒杀持续五分钟。

(2) 在一场秒杀结束前不能开始下一场秒杀，且两场秒杀时间间隔不小于十分钟
确定：若修改的商品未在秒杀当日且修改信息符合表单规则则提示修改完成，否则提示错误信息。

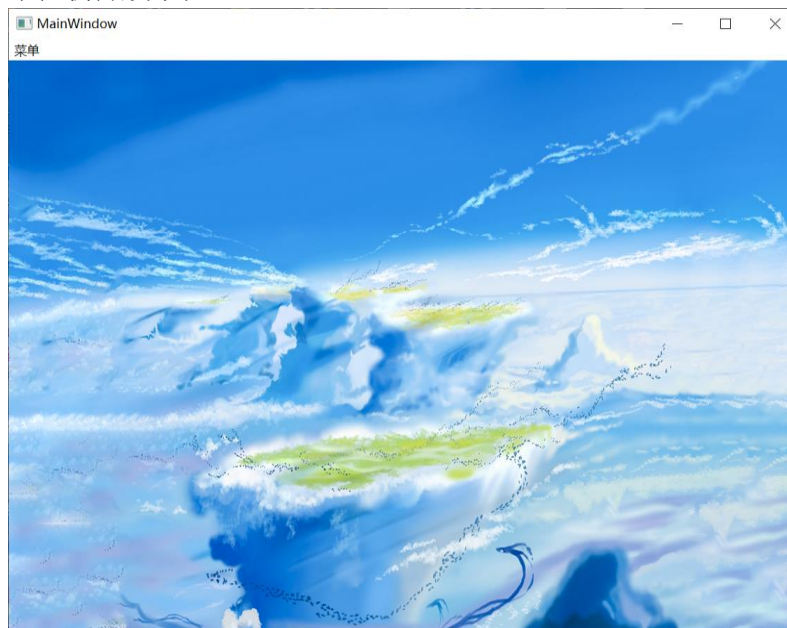
3. 添加：弹出商品名称输入框，管理员输入商品名称，若商品存在且不在秒杀列表内则显示添加成功，否则显示错误信息。

三. 设计

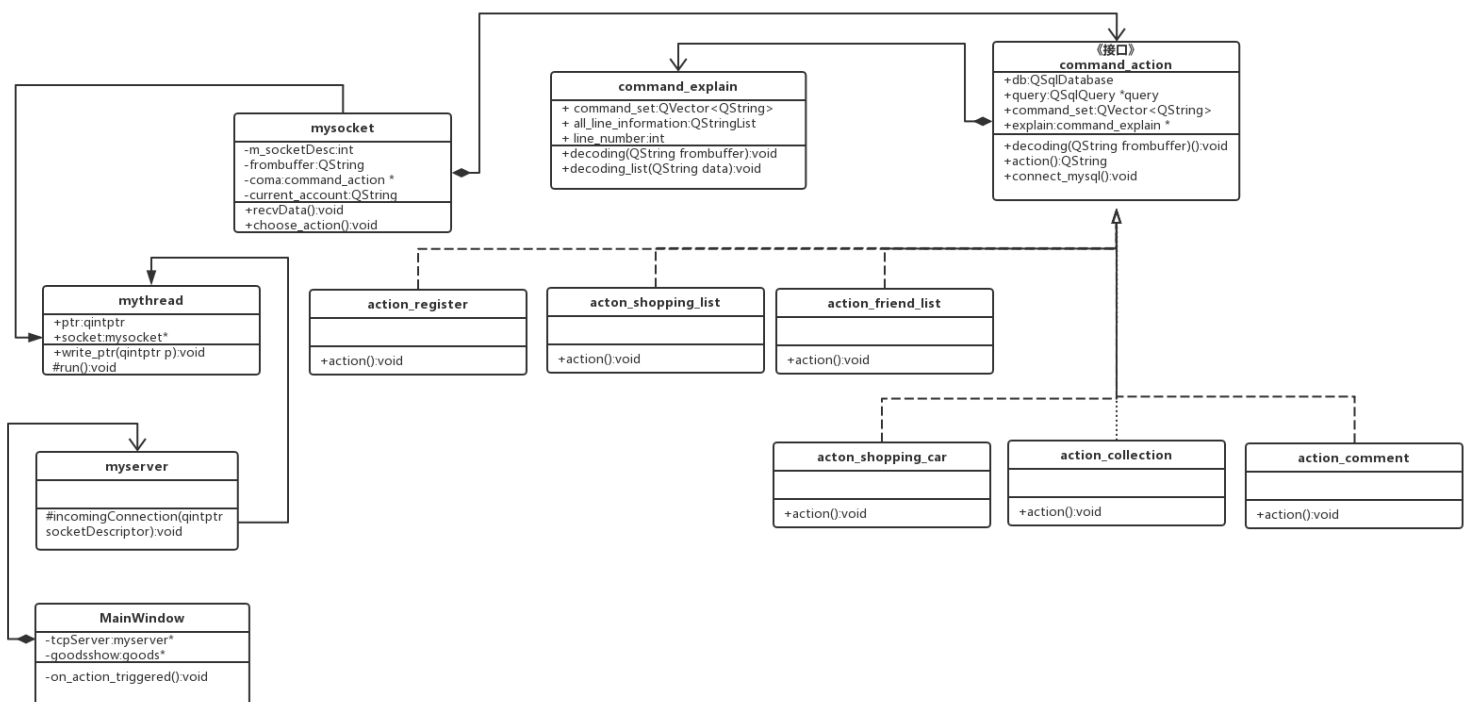
1. 类关系图(基于 C/S 模式)

管理员端:

(1) 初始界面



类图:



(2) 商品查看界面/商品上架窗口

Dialog

刷新

上架商品信息

上架商品

| | 名称 | 单价 | 数量 | 评分 | 销售量 | 下架 | 更改价格 |
|----|------|-----|----|----|-----|----|------|
| 1 | 乌冬 | 10 | 11 | 3 | 29 | 下架 | 价格 |
| 2 | 鸡蛋糕 | 20 | 10 | 0 | 20 | 下架 | 价格 |
| 3 | 炒饭 | 12 | 30 | 0 | 0 | 下架 | 价格 |
| 4 | 戚风蛋糕 | 30 | 6 | 0 | 14 | 下架 | 价格 |
| 5 | 水果沙拉 | 19 | 20 | 0 | 0 | 下架 | 价格 |
| 6 | 果汁 | 3 | 28 | 2 | 2 | 下架 | 价格 |
| 7 | 可乐 | 2.5 | 22 | 3 | 28 | 下架 | 价格 |
| 8 | 芒果蛋糕 | 35 | 17 | 0 | 23 | 下架 | 价格 |
| 9 | 蛋挞 | 12 | 30 | 0 | 0 | 下架 | 价格 |
| 10 | 雪碧 | 2.5 | 40 | 0 | 40 | 下架 | 价格 |
| 11 | 橙汁 | 2.8 | 20 | 0 | 0 | 下架 | 价格 |
| 12 | 苹果派 | 13 | 25 | 0 | 0 | 下架 | 价格 |
| 13 | 冰淇淋 | 3 | 4 | 0 | 0 | 下架 | 价格 |

促销系统

Dialog

上架商品信息表

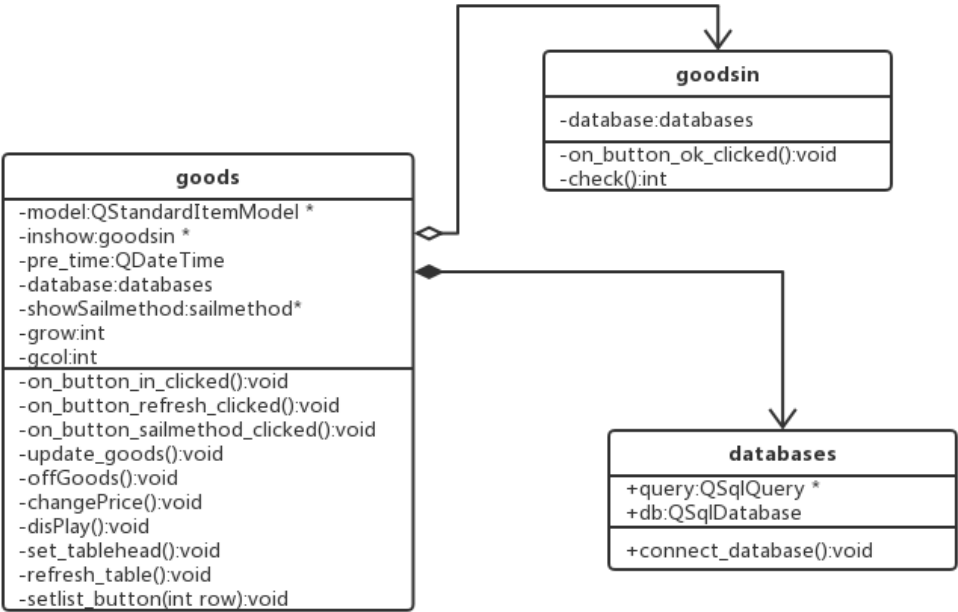
商品名称

销售单价

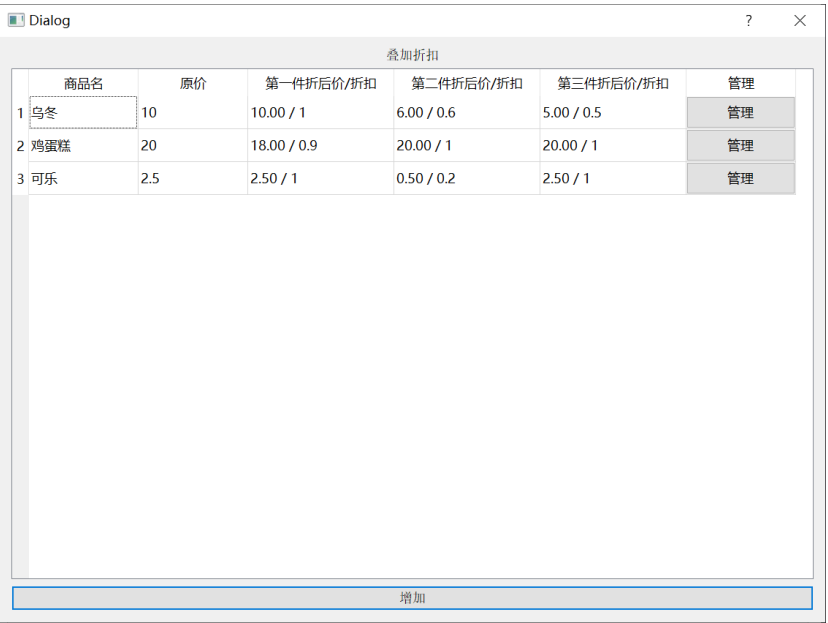
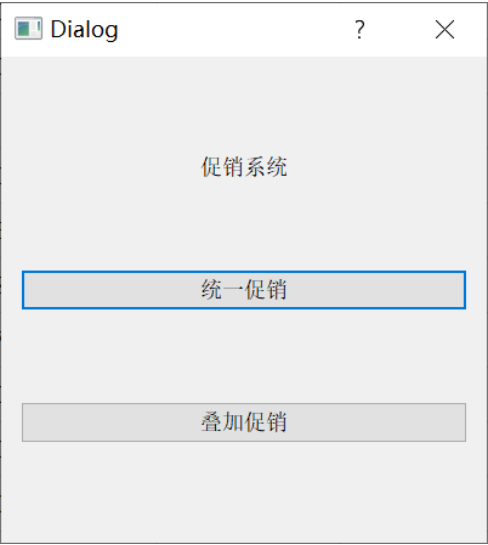
上架数量

确认

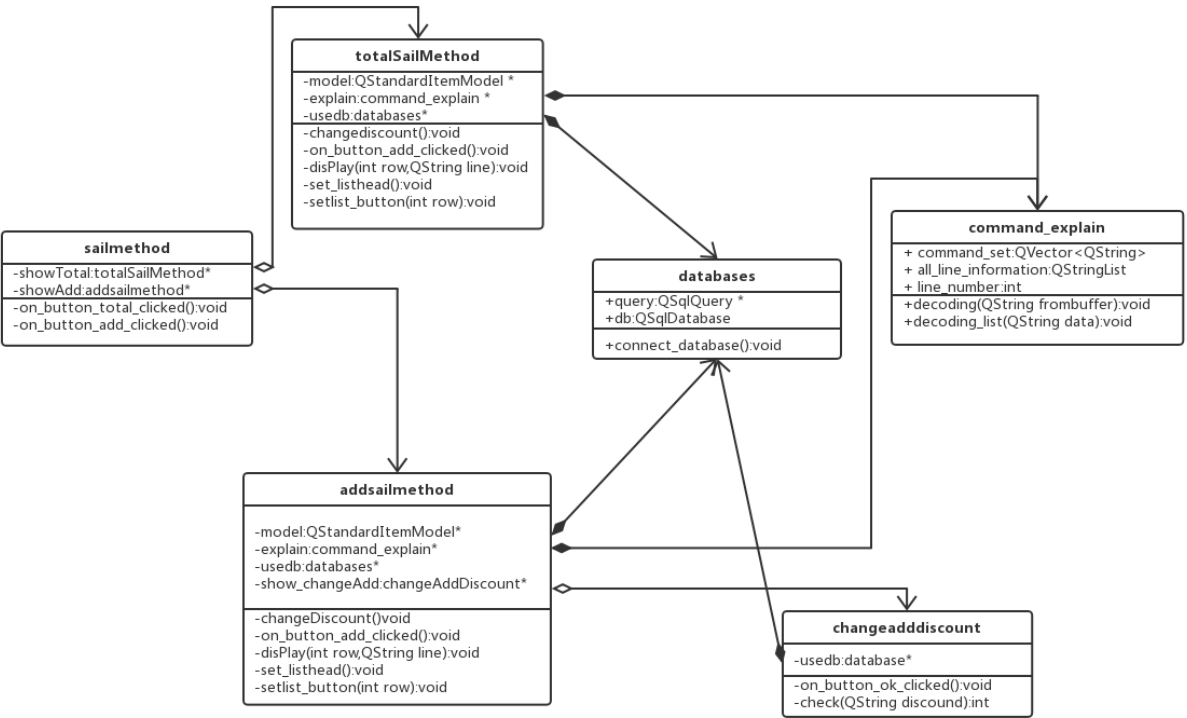
类图：



(3) 商品促销界面



类图:



客户端：
(1) 登录界面

Dialog

?

×

登录界面

账号

密码

登录

忘记密码

注册

Dialog

?

×

注册界面

账号

密码

邮箱

获取验证码

验证码

确定

Dialog

?

×

忘记密码

邮箱

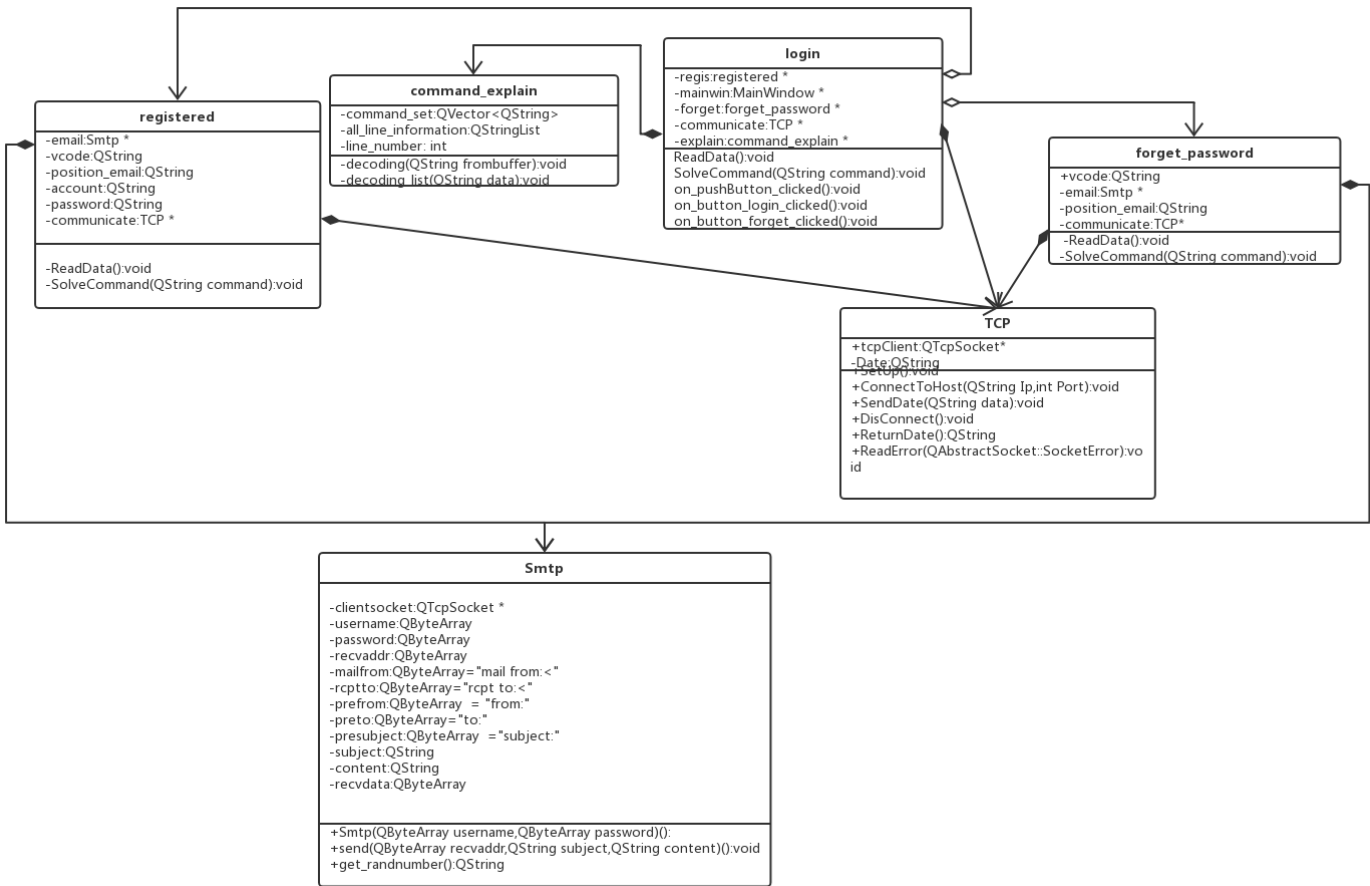
获取验证码

验证码

新密码

确认

类图：



(2) 查看商品界面

MainWindow

菜单

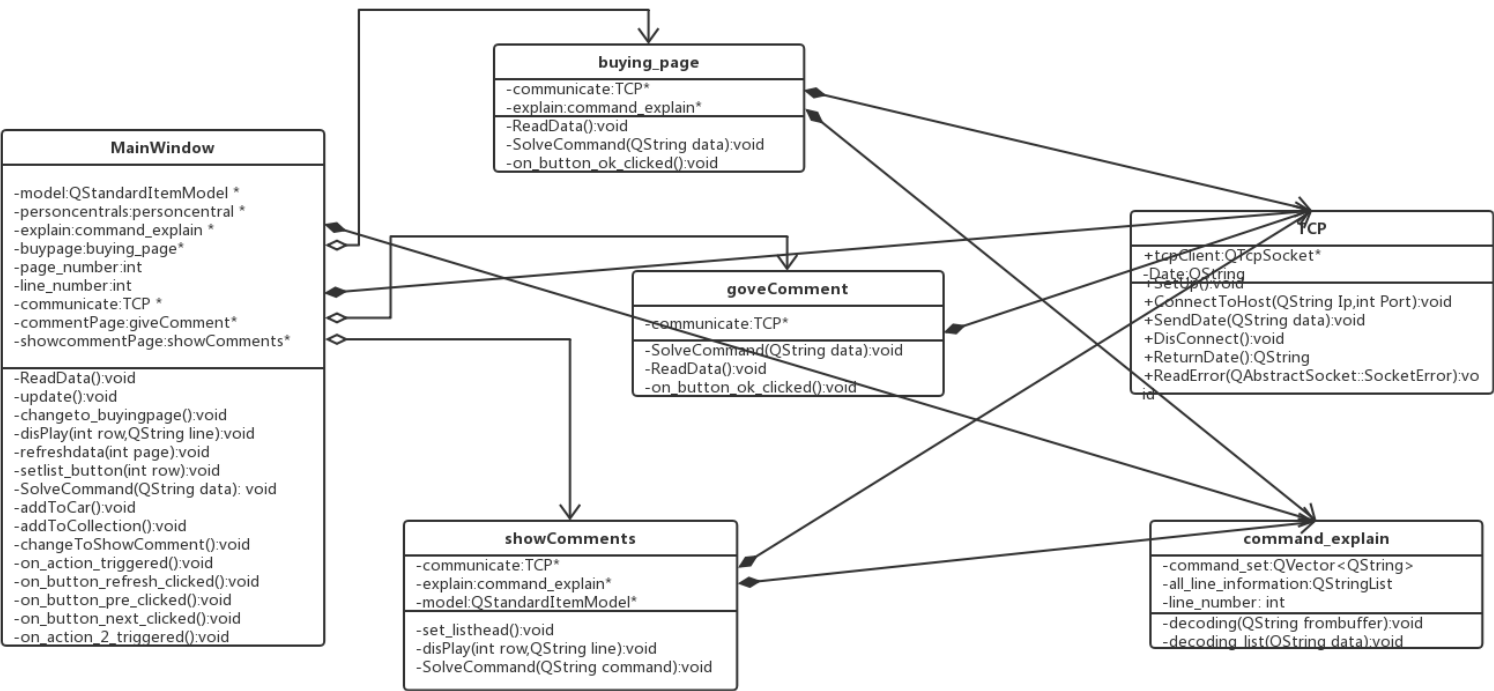
刷新

| | 名称 | 单价 | 数量 | 评分 | 销量 | 评价 | 购买 | 购物车 | 收藏 | 查看评价 |
|----|------|-----|----|----|----|----|----|-----|----|------|
| 1 | 乌冬 | 10 | 11 | 3 | 29 | 评价 | 购买 | 购物车 | 收藏 | 查看评价 |
| 2 | 鸡蛋糕 | 20 | 10 | 0 | 20 | 评价 | 购买 | 购物车 | 收藏 | 查看评价 |
| 3 | 炒饭 | 12 | 30 | 0 | 0 | 评价 | 购买 | 购物车 | 收藏 | 查看评价 |
| 4 | 戚风蛋糕 | 30 | 6 | 0 | 14 | 评价 | 购买 | 购物车 | 收藏 | 查看评价 |
| 5 | 水果沙拉 | 19 | 20 | 0 | 0 | 评价 | 购买 | 购物车 | 收藏 | 查看评价 |
| 6 | 果汁 | 3 | 28 | 2 | 2 | 评价 | 购买 | 购物车 | 收藏 | 查看评价 |
| 7 | 可乐 | 2.5 | 22 | 3 | 28 | 评价 | 购买 | 购物车 | 收藏 | 查看评价 |
| 8 | 芒果蛋糕 | 35 | 17 | 0 | 23 | 评价 | 购买 | 购物车 | 收藏 | 查看评价 |
| 9 | 蛋挞 | 12 | 30 | 0 | 0 | 评价 | 购买 | 购物车 | 收藏 | 查看评价 |
| 10 | 雪碧 | 2.5 | 40 | 0 | 40 | 评价 | 购买 | 购物车 | 收藏 | 查看评价 |

上一页

下一页

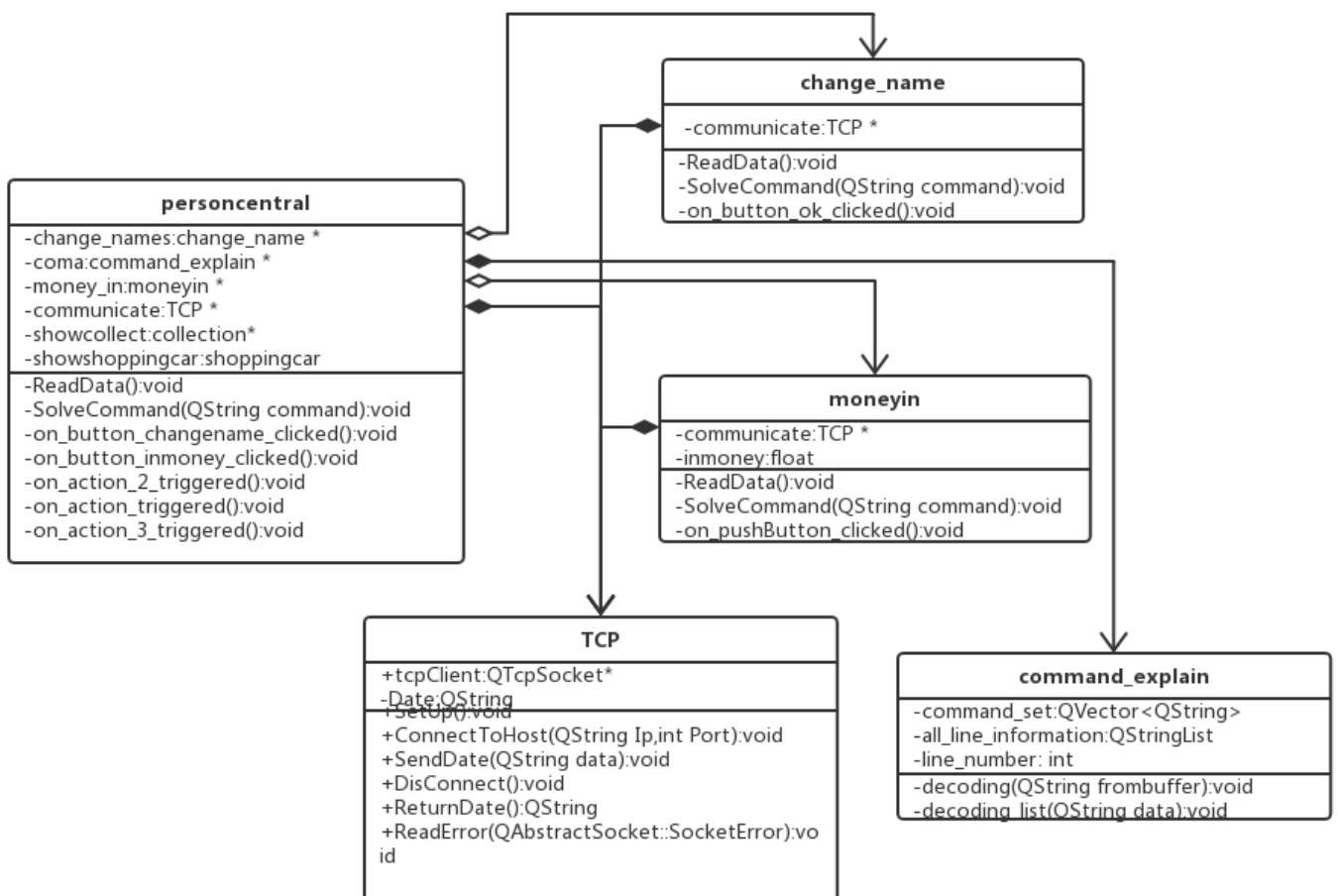
类图：



(3) 个人中心界面



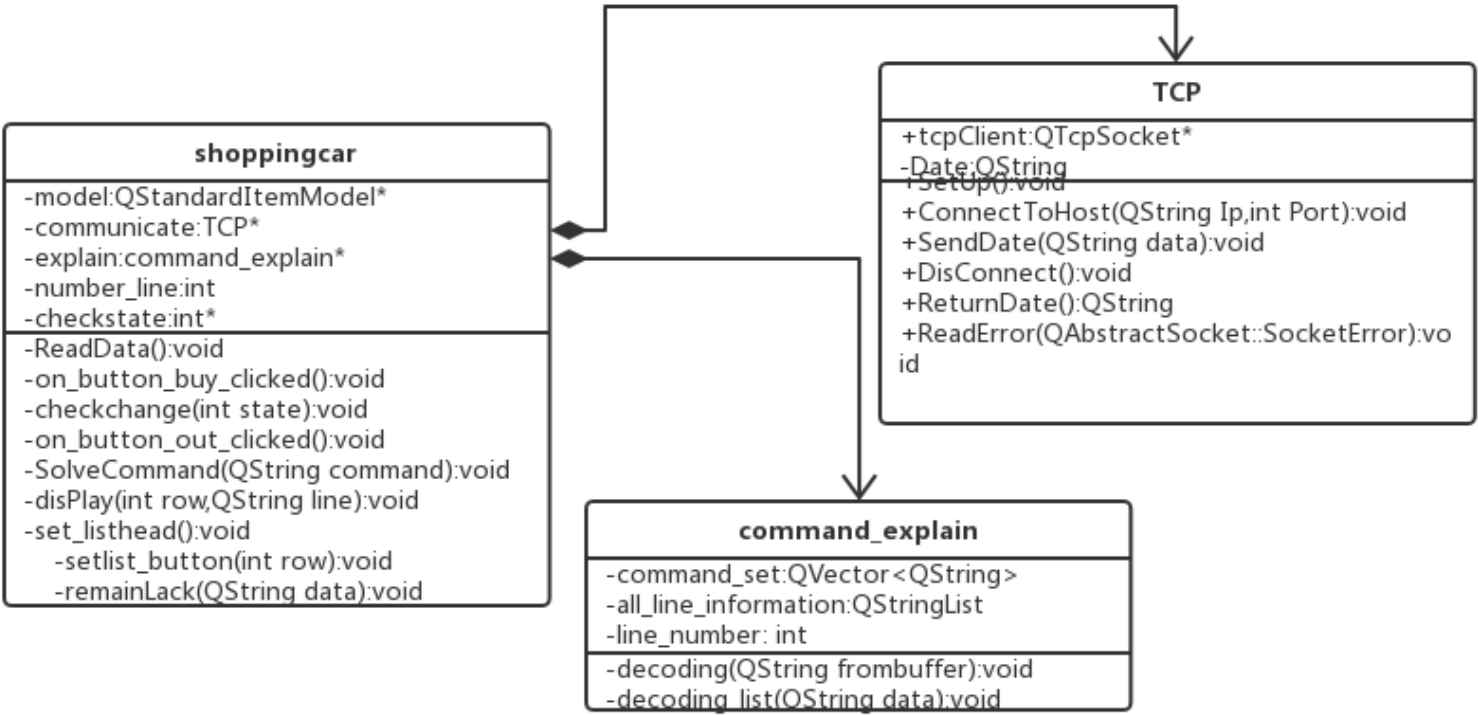
类图:



(4)购物车界面



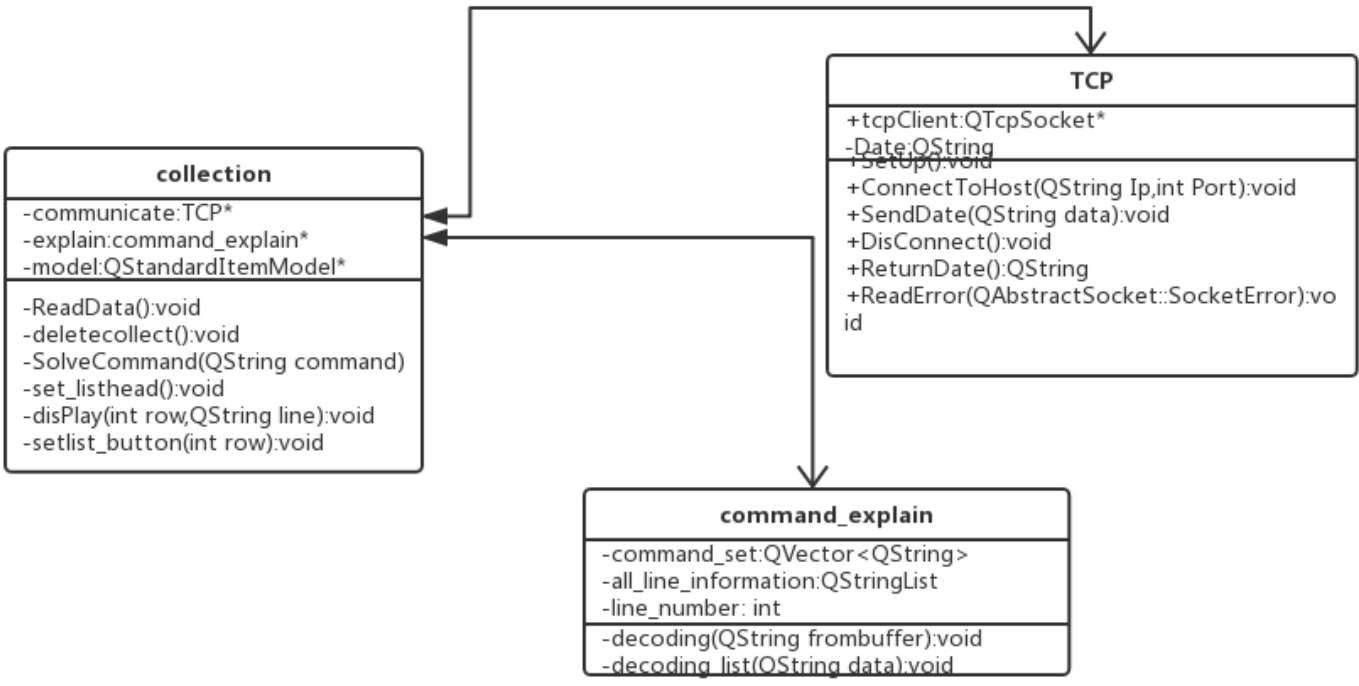
类图：



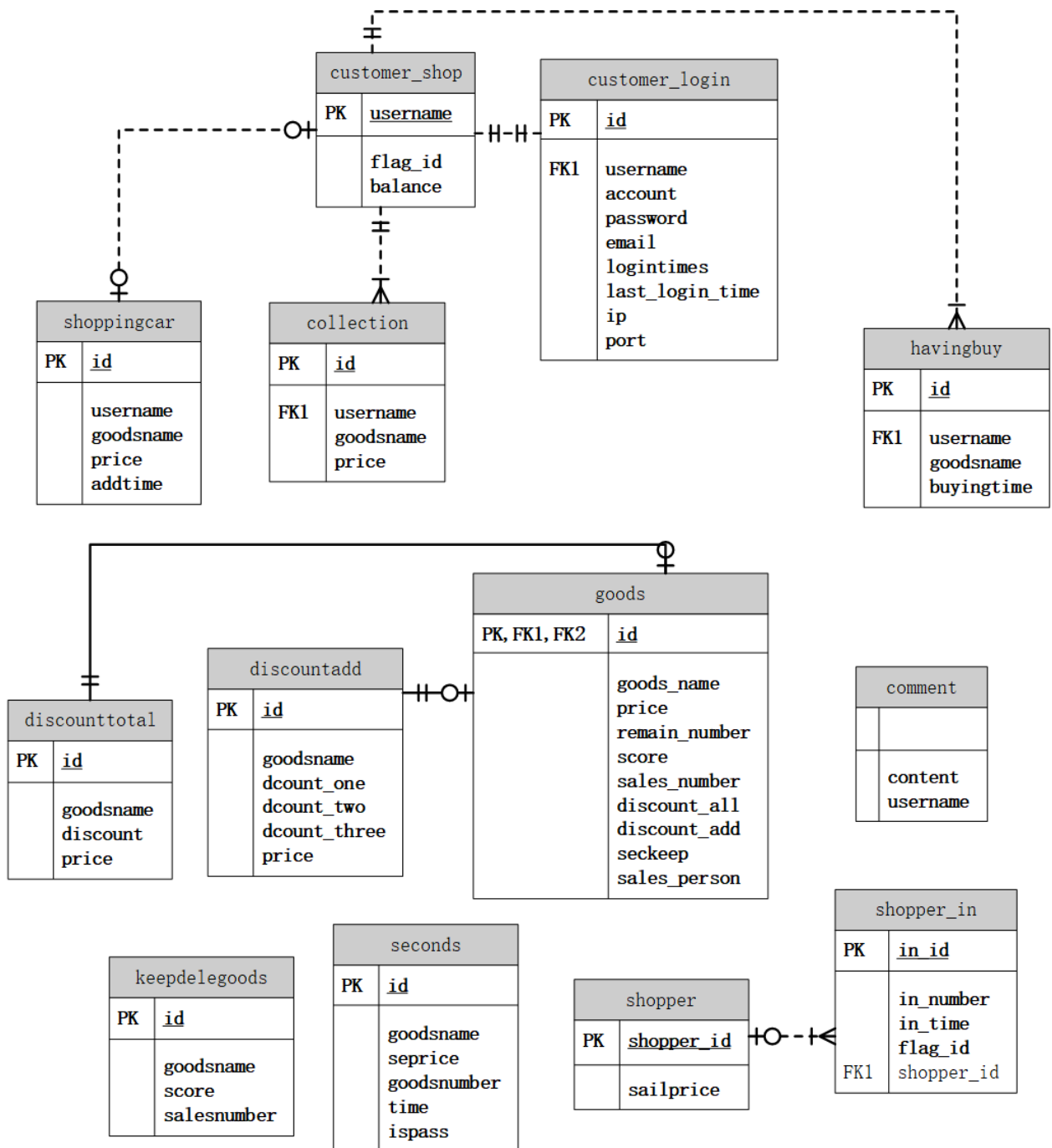
(5)收藏夹界面



类图：



2. 数据表结构图

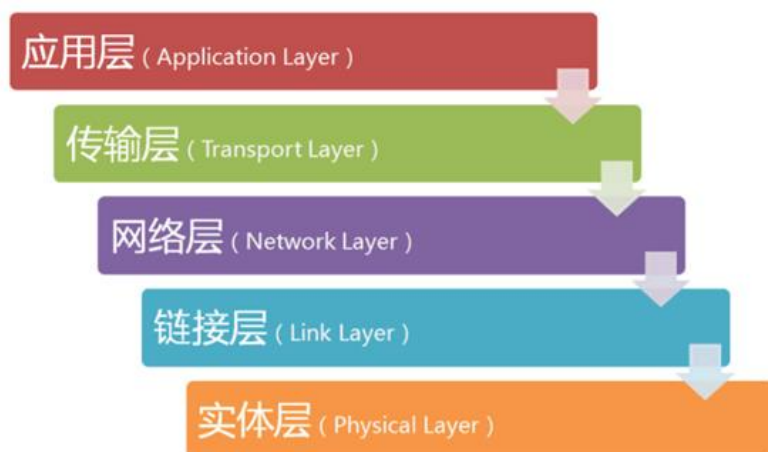


| 数据表名称 | 解释 | 数据表名称 | 解释 |
|----------------|------------|---------------|------------|
| customer_login | 存储用户个人信息 | customer_shop | 存储用户账户信息 |
| shoppingcar | 存储购物车信息 | collection | 存储收藏夹信息 |
| havingbuy | 存储已购商品信息 | discounttoal | 存储商品统一折扣信息 |
| discountadd | 存储商品叠加折扣信息 | goods | 存储商品信息 |

| 数据表名称 | 解释 | 数据表名称 | 解释 |
|---------|----------|---------------|-------------|
| comment | 存储商品评价信息 | shopper_in | 存储上架记录 |
| seconds | 存储秒杀商品信息 | keepdelegoods | 存储曾经上架的商品信息 |
| shopper | 存储管理员信息 | | |

3. Web 原理

1. TCP/IP 五层网络模型



2. 各层分析

(1) 实体层(物理层)

实体层是把电脑连接起来的物理手段。它主要规定了网络的一些电气特性，作用是负责传送 0 和 1 的电信号。

(2) 链接层

a. 作用：同一子网内使用广播的方式发送数据

b. 协议：

1) 以太网协议：一组电信号构成一个数据包，叫做“帧”。每一帧分成两个部分：标头和数据，head 中包含 MAC 地址，标头包含数据包的一些说明项，比如发送者、接受者、数据类型等等；数据则是数据包的具体内容。

c. 说明：

1) MAC 地址：数据包是由一个一个网卡传输到另一个网卡的，每一个网卡在生产时即具有了独一无二的 MAC 地址

2) 广播传输数据：某计算机发送数据，同一个子网络的计算机都会收到发送的包。它们读取这个包的“标头”，找到接收方的 MAC 地址，然后与自身的 MAC 地址相比较，如果两者相同，就接受这个包，做进一步处理，否则就丢弃这个包。

(3) 网络层

a. 作用：不同子网间传输数据

b. 协议：

1) IP 协议：规定网络地址的协议，叫做 IP 协议。它所定义的地址，就被称为 IP 地址。IP 地址分成两个部分，前一部分代表网络，后一部分代表主机，处于同一个子网络的电脑，它们 IP 地址的网络部分必定是相同的，因此通过识别

IP 地址来识别是否处于同一子网内。

IP 数据包：分为“标头”和“数据”两个部分，直接放进以太网数据包的“数据”部分。标头部分主要包括版本、长度、IP 地址等信息，“数据”部分则是 IP 数据包的具体内容。

2) ARP 协议：

用途：获取同一子网中的目标 ip 地址的 mac 地址，若不为同一子网则通过 ARP 获取网关 mac，再由网关处理。

c. 说明：

1) 子网掩码：用于判断两台计算机是否属于同一个子网络，它的网络部分全部为 1，主机部分全部为 0，用于与 IP 地址相与来获得 IP 地址的网络部分。

2) 通过 IP 地址识别网络的两台计算机若位于同一子网内则通过广播的方式完成通信，若不位于同一子网内则将数据发送至网关，由网关将数据发往目标 IP 地址网络，再通过 MAC 地址进行数据传输。

(4) 传输层

a. 作用：建立端到端通信，不同的程序使用不同的端口，需要将数据传输到对应程序的端口位置

b. 协议：

1) UDP 协议：

特点：格式几乎就是在数据前面加上端口号。

UDP 数据包：也是由“标头”和“数据”两部分组成，标头部分主要定义了发出端口和接收端口，“数据”部分就是具体的内容。然后，把整个 UDP 数据包放入 IP 数据包的“数据”部分。

2) TCP 协议：

1. 特点：三次握手与四次挥手

2. TCP 数据包和 UDP 数据包一样，都是内嵌在 IP 数据包的“数据”部分

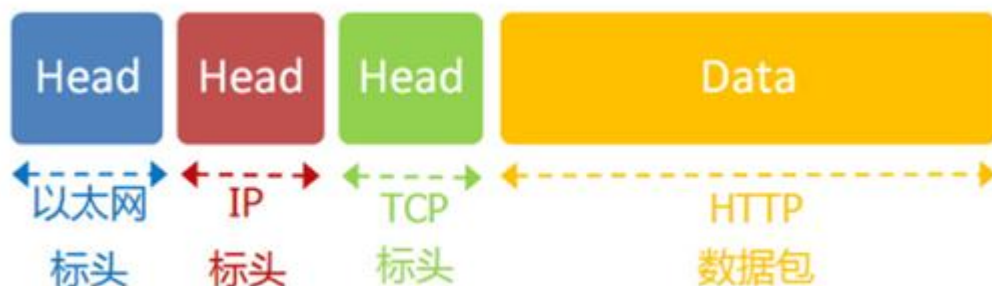
(5) 应用层

a. 作用：将接收的数据进行解读

b. 协议：HTTP，FTP，DNS

应用层数据放在 TCP 数据包的“数据”部分

当使用 http 协议时以太网数据包格式如下：



3. Web 通信原理

1) DNS 协议

用途：假设输入网站 `www.baidu.com`，为建立通信必须获得对方的 IP 地址，则 DNS 获取域名并将其解析为目标 IP 地址并发送给访问者

2) 通信流程

1. 输入网站，本机 IP 地址与选择的域名服务器 IP 地址利用子网掩码进行比对，发现本机 IP 地址与域名服务器 IP 地址不处于同一子网内，则通过网关将数据传输至目标 IP 网络，再将数据传输至域名服务器 MAC 地址处。

2. 获取到域名服务器传输的解析 IP 地址后，本机利用同样的方式确认本机 IP 地址与目标 IP 地址是否处于同一子网，并将数据传输至目标服务器处，服务器解析 HTTP 协议获取请求数据，由于 HTTP 协议位于应用层且传输层使用 TCP 协议，因此每一次 HTTP 请求就需要建立一次 TCP 连接和释放 TCP 连接。

3. 服务器读出传输数据的“HTTP 请求”，接着做出“HTTP 响应”，再用 TCP 协议发回来。本机收到 HTTP 响应以后，就可以将网页显示出来，完成一次网络通信。

3) 各层协议嵌套分析

1. HTTP 协议

HTTP 数据包嵌在 TCP 数据包内，包含一个完整的 HTTP 请求消息，含有一个请求行，若干个请求头与实体内容。

请求行：描述客户端的请求方式、请求资源的名称、http 协议的版本号。

请求头：包含客户机请求的服务器主机名，客户机的环境信息等。

实体内容：包含浏览器端通过 HTTP 协议发送给服务器的实体数据，例如登陆时的账号信息等。

2. TCP 协议

TCP 数据包需要设置端口，接收方的 HTTP 端口默认是 80，发送方（本机）的端口是一个随机生成的 1024-65535 之间的整数，TCP 数据包再嵌入 IP 数据包。

3. IP 协议

IP 数据包需要设置双方的 IP 地址，发送方是本机 IP 地址，接收方是目标 IP 地址。IP 数据包嵌入以太网数据包。

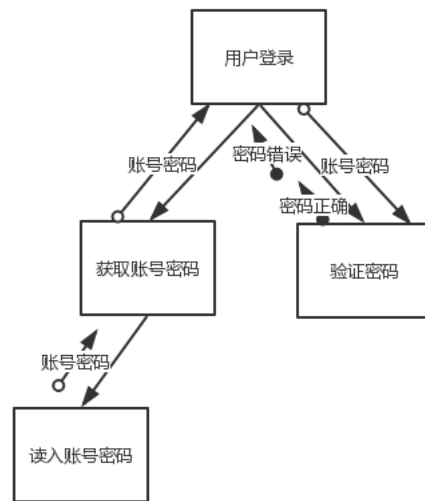
4. 以太网协议

以太网数据包需要设置双方的 MAC 地址，发送方为本机的网卡 MAC 地址，接收方为网关的 MAC 地址（通过 ARP 协议得到）。

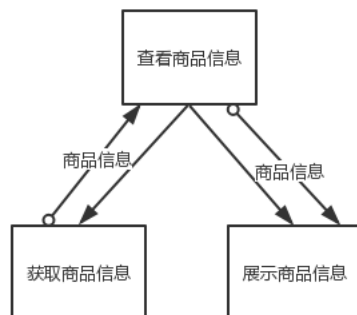
四. 实现

1. 用户模块结构图

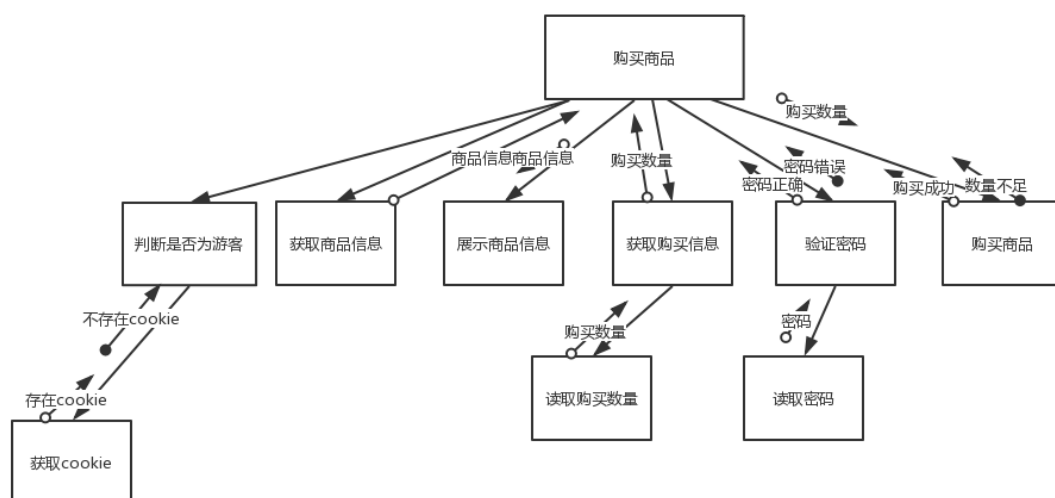
(1) 用户登录



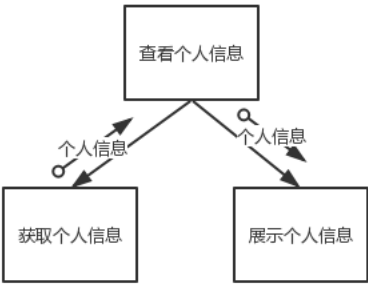
(2) 查看商品信息



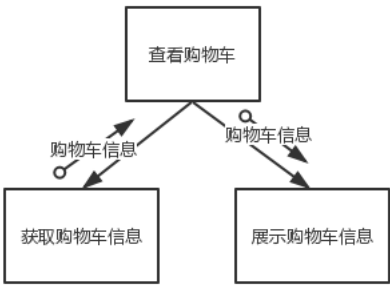
(3) 购买商品



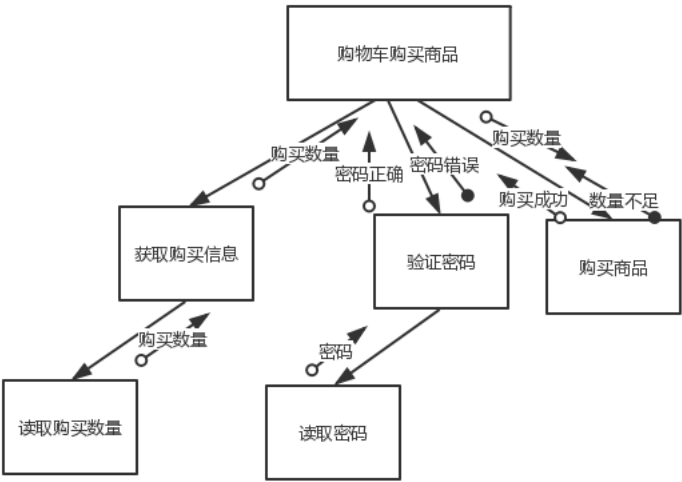
(4) 查看个人中心



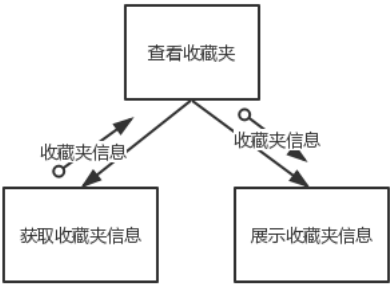
(5) 查看购物车



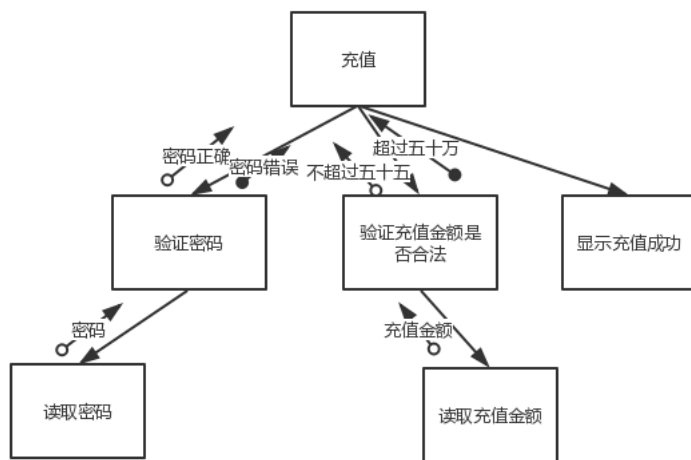
(6) 购物车内购买商品



(7) 查看收藏夹



(8) 充值



2. 测试用例

B/S 模式:

(1) 更换头像

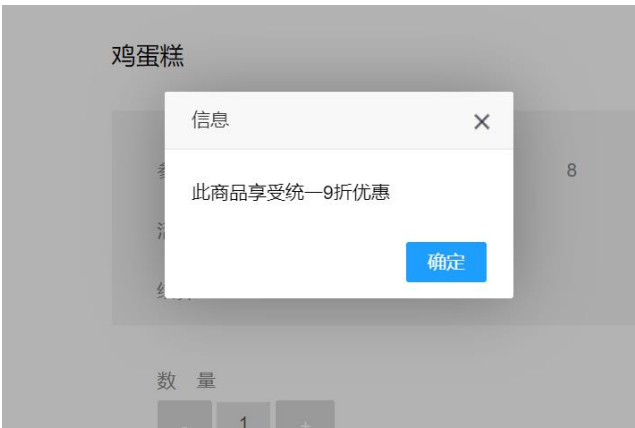
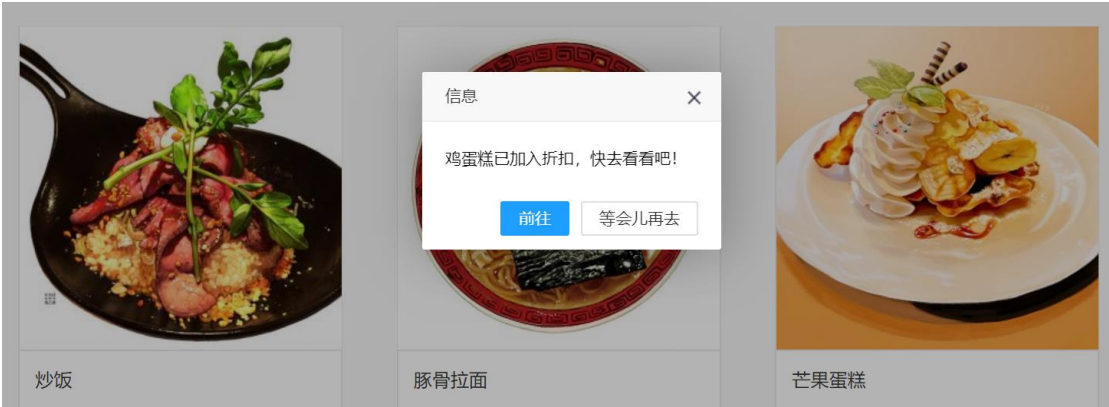
a) 上传超过 1MB 的图片



b) 上传非 jpg 与 png 文件



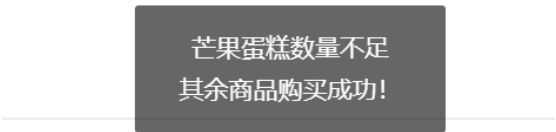
(2) 某商品加入折扣列表内通知客户端



(3) 同时购买商品导致数量不足



(4) 购物车购买商品某商品数量不足



C/S 模式:

1. 邮箱注册

WeShop

星罗棋布

[详情](#)

您正在进行WeShop的账号注册操作，本次
请求的邮件验证码是：k5Cadi

五. 主要难点重点问题讨论

C/S 模式:

1. 使用多线程对多用户操作进行处理

a) 项目环境特点:

(1)QT 中的 QTcpServer 在使用 listen() 进行端口监听时，若接收到连接将调用 incomingConnection(qintptr socketDescriptor) 函数，其中 socketDescriptor 为客户端标识，QTcpSocket 可获取此标识与客户端交互。当客户端有消息到达时 QTcpSocket 将触发 readyRead 信号。

(2)QT 使用信号与槽机制将信号与槽函数关联，当发出信号时将自动触发槽函数

(3)QT 创建线程的方式之一为继承 QThread，重写 run() 函数，则 run() 函数即为线程运行函数

b) 设计流程:

(1)新建一个类继承自 QTcpServer，重写 protected: void incomingConnection(qintptr socketDescriptor); 在此函数中创建线程，则实现了每有一个新连接则创建一个独立的新线程对其进行处理

(2)新建一个类继承自 QThread，重写 protected: void run(); 将此类组合进 1 中的类，则当 1 中的类使用 start() 后将自动调用 run() 函数，run() 函数维持 QTcpSocket 的信号与槽函数的绑定。

(3)新建一个类继承自 QTcpSocket，加入自定义函数 recvData，将 readyRead 信号与 recvData 绑定，则当有消息到达时在此类的内部即可完成消息处理。

c) 核心思想:

利用 QT 自身函数特性为每一个新连接创建一个新线程，新线程维持 readyRead 信号与槽函数的绑定，此槽函数即为信息处理函数。

2. 处理并发问题

a) 主要思路:

使用 mysql 自带的事务操作对并发问题进行处理，其中事务隔离级别选用默认的隔离级别：repeatable-read

b) repeatable-read 隔离级别特点:

当多线程同时访问数据库时, 若同时进行 select 操作则将触发共享锁, 即各线程可同时读取数据, 不存在阻塞; 若同时进行 update 操作则将触发排他锁, 即各操作将进行排序一个一个对数据库进行操作, 在一个线程 commit 后其他线程的操作才会执行, 否则处于阻塞状态。

c) 并发处理具体方式:

使用如下语句对商品数量进行修改:

```
update goods set remain_number=remain_number-1 where id=1 and  
remain_number>0;
```

则假设用户 A 与用户 B 同时购买商品 a 时流程如下:

两条 update 指令同时到达 Mysql 处从而触发排它锁, 数据库对两条指令进行排队, 假设用户 A 在前, 则用户 A 正确执行指令, 此时用户 B 执行相同指令, 因为此时 remain_number=0 所以无法找到该商品, 从而无法再次对相同商品执行 update 操作, 为通知客户端购买是否成功, 使用 SELECT ROW_COUNT(); 指令查看上一条 update 指令所影响数据条目。

B/S 模式:

1. 支持上传商品时上传商品图片, 用户使用头像并更换头像, 所有商品都以图片的方式进行展示。

实现方式:

a) 使用 html 的 input 标签作为图片获取载体, 且由于原始标签影响美观, 因此使用 style="display:none" 将其隐藏, 使用自定义图标间接获取图片, 并使用 .change() 对图片是否已上传进行监视。

b) 当图片上传后使用 jquery.form.min.js 的 ajaxSubmit 方法发送图片至服务器。

c) 服务器删除原始图片, 将图片转换为缩略图, 存储图片。

d) 前端的 jquery 更新 img 的 src 属性。

2. 部分页面采用 ajax 技术实现页面的动态更新

流程:

a) 在目标页面预先设置容器

b) 使用 jquery 的 .load() 向服务器请求元素并将元素放入指定容器

c) 服务器在接收请求后将收到的信息传入所请求的 html 文件中并将其返回

优点: 当用户发起请求时无需刷新整个页面就可以对页面进行局部更新, 增加用户的体验度。

3. 当商家将某商品加入商品折扣列表中时服务器将主动向当前在线的客户端发送消息推送, 提示客户端某商品已加入折扣。

实现方式:

使用基于 workerman 框架的 websocket 技术, 服务器监听自定义的 2346 端口, 由于此项目为 1 对多电商系统, 因此使用一个变量存储商家连接, 使用数组存储客户端连接, 当商家将商品加入折扣列表时将信息发送至服务器, 服务器再将信

息循环发送至所有客户端，则完成了服务器反向推送的功能。

4. 使用模型对数据表进行操作

TP5 框架中的每一个 model 都对应了数据库中的一张表，当示例化了一个 model 时其属性即为数据库中的一个字段，通过此方式脱离了数据库的原生语句操作，完成了“面向关系”到“面向对象”的转换。

5. 秒杀功能架构

(一)秒杀时间分为三个阶段：

1. 秒杀开始前：

时间段：当前日期等于秒杀日期且当前时间小于秒杀开始时间

服务器端任务：初始化 redis 数据，将当前商品秒杀信息由 mysql 存入 redis

2. 秒杀进行中：

时间段：当前时间大于等于秒杀开始时间且小于秒杀开始时间五分钟后

服务器端任务：处理用户购买信息

3. 秒杀结束：

时间段：

(1) 当前时间大于等于秒杀时间开始五分钟后且持续时间为五秒

(2) 秒杀商品已全部售出

服务器端任务：处理 redis 中存储的用户秒杀信息，对商品购买进行处理

(二)秒杀限制设计：

1. 秒杀时间限制：

(1) 一天内只能进行一次秒杀，每次秒杀持续五分钟。

(2) 在一场秒杀结束前不能开始下一场秒杀，且两场秒杀时间间隔不小于十分钟

2. 秒杀商品限制：

在将商品加入秒杀列表时此商品数量必须>0

3. 秒杀购买限制：

用户点击秒杀按钮一次只能购买一件商品

4. 修改秒杀信息限制：

(1) 商家无法在秒杀当天对相应的秒杀商品进行修改

(2) 秒杀进行时无法对秒杀信息进行修改

(3) 商家若在商品列表下架秒杀商品则提示商家必须先将商品移出秒杀列表

(4) 商家修改秒杀销售数量时秒杀数量必须大于或等于当前商品数量

5. 删除 mysql 中已结束秒杀信息时间限制：

每天的 00:05:05 删除数据库中昨天的秒杀商品信息

(三) 难点:

1. 一天内只能进行一次秒杀, 若秒杀一结束就将秒杀商品信息删除则商家可通过秒杀结束后再增加当日秒杀的方式实现一天执行多次秒杀。

解决方式: 秒杀结束时不立即删除秒杀信息, 而选择每天的 00:05:05 进行删除, 设置此时间的意义在于:

即便秒杀时间定为 23:59:59, 信息的删除也不会影响正在进行的秒杀。秒杀时长为 5 分钟, 则结束时间最迟为 00:04:59, 同时为结束后信息的处理预留了 5 秒的时长。

2. 将商品加入秒杀后可能出现商品数量不足的情况。

解决方式: 增加一个字段用来存储商品的秒杀数量, 当商品加入秒杀时直接将原商品数量以 mysql 事务操作的方式减少, 则秒杀购买结算时先从新字段减少数量, 若有剩余则增加原商品数量, 使用 mysql 事务操作使得商品的数量避免了并发的影响。

3. 当秒杀活动开始时将出现高并发现象, 若只使用 mysql 的事务操作, 大量数据同时访问数据库可能造成性能下降, 互相等待甚至死锁问题。

解决方式: 采用 redis 将秒杀商品信息暂时进行存储, 因为 redis 将数据存于内存, 因此性能极高, 为解决并发问题, 在商品加入秒杀时即使用 list 数据结构存储数量, 当用户购买请求到达时使用 lpop 读出数据, 若有数据读出则将用户名使用 redis 存储, 否则表示已经卖完商品。此方法利用了 redis 对 list 操作的原子性解决了购买的并发问题。

4. 客户可通过修改系统时间提前发起秒杀购买请求

解决方式: 秒杀的所有时间皆以后端时间为准, 前端时间仅提供客户参考作用, 为同步时间, 客户的每次刷新服务器都将返回服务器系统时间, 客户端以此时间为据通过 jquery 更改倒计时显示时间。

(四) 技术特点:

1. 使用 workerman 框架的定时器进行计时

服务器每一秒都获取当前的时间信息, 通过与秒杀三阶段所设定的时间进行对比以进行相应的业务, 通过此方式实现了后端主动控制秒杀的所有进程而不是由客户端的点击来进行相应业务的触发。

2. 使用消息队列异步处理用户购买信息

由于秒杀开始时将有大量数据访问, 若同时操作 mysql 数据库将使用户无法立即得到秒杀操作所返回的信息, 因此采用消息队列方式异步处理, 信息到达服务器端时立即判断是否秒杀成功, 成功则存储用户名并立即返回成功秒杀的信息, 否则立即返回秒杀失败的信息, 而购买信息的处理则等到秒杀进入第三阶段(秒杀结束)时才取出秒杀成功的用户名进行一一处理。

六. 与商用微商系统的差距

1. 商业微商系统使用的支付方式通常是与支付宝与微信支付绑定，而 Westore 系统只能使用充值模拟支付操作。且通常可以直接由微信或 QQ 授权登录，而不需要重新创建账号。
2. 商业微商系统对数据库的应用采取分布式与集群的方式，分布式使得数据库职能分解，集群在几个服务器上部署相同的应用程序，来分担客户端请求。分布式与集群架构使得网站在应对大量数据时得到更大的抗压能力，而 Westore 系统的数据库存在一台电脑内使得存储性能远不及商业微商系统。
3. 商业微商系统对缓存的应用非常广，如从浏览器缓存，反向代理缓存，页面缓存，局部页面缓存，对象缓存等，对缓存的大量使用使得访问数据库的次数大大降低，从而提高数据库的承载力。而 Westore 对缓存的主动使用仅限于秒杀系统。
4. 商业微商系统通常是大型系统，而为使大型系统更易维护与扩展需要对其进行拆分。以淘宝网为例，经过搜索，淘宝网对整个系统进行了水平和垂直两个方向的拆分，水平方向上，按照功能分为交易，评价，用户，商品等系统，同样垂直方向上，划分为业务系统，核心业务系统以及基础服务，这样使得各个系统都可以独立维护和独立的进行水平伸缩，比如交易系统可以在不影响其它系统的情况下独立的进行水平伸缩以及功能扩展。而 Westore 的系统架构还相当脆弱，若想要对功能进行扩展需要修改多处代码位置。
5. 商业微商系统还具有强大的监控与预警系统，比如当某个页面访问量增多的时候，系统能自动预警，某台服务器的 CPU 和内存占用率突然变大的时候，系统也能自动预警，拥有强大的监控与预警系统使得系统变得更易维护，且安全性的增加也使得用户对系统的信任得以提升。

七. 重新开发会采取的措施

1. 将 C/S 模式与 B/S 模式进行结合。在此系统中由于一开始未考虑使用 B/S 模式因此在开发初期的架构上并未考虑二者的结合问题，导致后期加入 B/S 模式后无法再对 C/S 架构进行更改，使得二者成为了两个毫无关联的系统。
2. C/S 模式的 UI 设计采用的是多页面布局，降低了代码的复用，且多个界面使得任务栏有多个窗口。若重新开发将使用统一界面布局，以一个主界面为主，用点击切换的方式代替窗口弹出。
3. 增加好友功能，实现与客服和与好友间的通信功能，且能够在此基础上增加“帮一帮”功能。
4. 重构 B/S 网站架构。目前的网站架构相对零散，可扩展性与可维护性相对较差，可通过学习其他网站的架构模式以提高对架构方式的理解。
5. 将网站加载到阿里云服务器，使得 Westore 成为真正的网站。

八. 获得的提高

1. 更深刻得理解了网络通信原理，明白了 TCP/IP 五层网络模型的真正意义。
2. 理解了网站的构站思路，在实践中理解了网站的前端与后端的区别，前端负责将数据展示给客户，后端对于用户不可见，因此数据的处理放于后端更安全。
3. 明白了多线程的意义，多线程的使用使得在单进程内能同时执行多个操作，从而避免了排队处理的产生，同时多线程也引发了并发问题，由此产生了锁机制，通过加锁避免了多线程对同一数据的更改产生错误。
4. 理解了持久层的意义，数据库的产生使得程序员摆脱了繁杂的文件操作，而持久层的产生又使得程序员摆脱了重复的 mysql 语句，可以使用面向对象思想对数据库进行操作，从而大大提高了写程序效率，而数据库作为工具自带的事务操作使得多线程访问数据不会产生并发问题，从而保证了数据的正确性与安全性。
5. 学习的方法上意识到编程最重要的是动手实践，光是看书或是看视频而没有上手实践在效率上较低，可以在编写总系统前先写几个小例子验证方法是否能够成功，这样既不会由于方法不成功对主系统造成影响也能够对方法的正确与否进行检验。且在想要实现一种功能时应先自行对功能的实现方式进行思考，如 Westore 项目中的下拉动态加载更多数据，一开始直接在网上进行搜索，而搜索到的内容无法理解，于是自己进行思考并发现利用现有的技术已经可以实现此功能，于是利用自己的理解实现后再与网上的实现方式进行对照发现大同小异。
6. 在进行一个大功能的架构时可以先查看主流架构的实现方式，如秒杀系统，在经过搜索后得知秒杀系统的难点在于高并发的处理并得知了 redis 缓存技术，在接下来的架构中便围绕 redis 缓存技术进行构造，从而达到事半功倍的效果。

九. 缺憾

1. 没有将 C/S 模式与 B/S 模式结合起来。
方法：二者对同一个数据库进行操作且功能应完全对应，在架构时便对二者的结合进行思考。
2. 没有加入售后系统与好友功能。
方法：
 1. 售后系统：增加客服交流功能，当用户对客服发起交流请求时服务器查得存储的用户 IP 地址与端口号，将回复的消息只传递给此用户，则通过此方式实现了交流。
 2. 好友功能：用户输入好友账号后将好友请求进行存储，若好友在线则将请求直接进行发送，否则每当用户登录时便对数据进行访问，查看是否有属于自己的好友请求，若请求账号是自己则将数据发送至此账号，从而实现异步好友添加。

十. C/S 开发模式与 B/S 开发模式的比较

1. 二者的通信方式不同。C/S 通信需要手动建立 TCP 连接，而 B/S 开发模式若仅采用请求响应的方式进行开发则无需开发人员手动进行 TCP 连接，Web 开发利用的是基于 TCP 的 HTTP 应用层协议，当用户输入 URL 进行访问时数据将自动进行封装嵌套并发送至服务器，服务器也将数据包按照协议进行解包，从而将得到的数据传给子进程内的 php 程序进行处理并将处理结果重新传给主进程再封装并发送给用户浏览器。

2. 二者处理数据排队的方式不同。C/S 开发需手动建立多线程并对多线程进行管理，在此项目中使用的 Web 服务器为 Apache，而 Apache 默认保持多个子进程等待处理数据，当数据到来时子进程又会根据不同的信息处理模式生成不同数量子线程对数据进行处理。因此 B/S 开发若为简单的请求响应模式则也无需手动对线程进行处理。

3. 二者处理服务器主动通信的方式不同。C/S 模式采用的同样的 TCP 连接来进行消息传递，而 B/S 模式进行服务器主动通信需使用 websocket 协议，websocket 协议在建立时需从 HTTP 协议升级协议为 websocket，因此协议仍然是基于 TCP 传输层之上。

4. 二者的开发模式相似。B/S 开发模式将前端界面与后端服务器程序分离，界面与服务器程序之间的关联仅在于数据传输上，同样 C/S 模式由于采用的是 QT 开发环境，QT 将 UI 界面与业务实现自动分离，因此业务的实现同样也不需要考虑界面。且 QT 的界面与业务代码也是基于请求响应模式，与 web 相似。

5. 二者的更新方式不同。当在 C/S 模式上增加功能时需要在客户端程序上进行修改，因此若发布后更新软件需要重新下载新的程序，而 B/S 模式增加功能只需在客户端控制器进行实现，当用户输入 URL 时便自动加载新的界面。