

[电子商务平台开发技术文档]

(密码学原理与实践-实践部分)

[项目名称： 基于互联网的电子商务平台开发]

[姓名： 李尚宇]

[学号： 1180300802]

目录

1. 背景与意义	1
1.1 项目开发意义	1
1.2 国内外现状及技术综述	2
1.2.1 国内外电子商务发展现状:	2
1.2.2 技术综述	3
2. 需求分析	4
2.1 总体需求	4
2.2 功能需求	4
3. 概要设计	5
3.1 总体功能流程图（基本结构框架）	5
3.2 功能结构设计	6
3.2.1 登录注册功能结构设计	6
3.2.2 管理员功能结构设计	7
3.2.3 商城功能结构设计	8
4. 详细设计	9
4.1 数据库设计	9
4.1.1 用户数据表（user 表）	9
4.1.2 商品数据表（goods 表）	9
4.1.3 购物车数据表（shopping_cart 表）	10
4.1.4 订单数据表（completedorder 表）	10
4.2 jsp 开发前端页面	11
4.2.1 项目前端目录	11
4.2.2 以 e_shop.jsp 为例展示前端开发	11
4.3 后端 servlet 实现	12
4.3.1 项目后端目录	12
4.3.2 DBTool 包和 encrypt 包介绍	13
4.3.3 servlet 包介绍	13
4.4 系统安全实现	14
4.4.1 前后端交互加密——RSA 公钥加密:	14
4.4.1.1 对用户输入的信息用 js 实现加密	14
4.4.1.2 前端已有的信息发送到后端时在 jsp 中动态加密	15
4.4.2 防重放攻击——session 用户 id 确认、操作序列号与订单号	16
4.4.2.1 前后端数据传输放重放攻击——用户 id 确认与操作序列号	16
4.4.2.2 交互过程放重放攻击——订单号	16
4.4.3 交互过程安全保障——数字信封与订单号	16
4.5 详细功能结构流程图	18
5. 实现与测试	19
5.1 管理员页面	19
5.1.1 添加商品	19
5.1.2 修改商品信息	20

5.1.3 删除功能.....	21
5.2 普通用户登录和注册页面	21
5.3 商城页面（商城主页面、购物车页面）	22
5.4 支付操作、支付过程的页面跳转以及支付后已购买商品页面：	24
6. 结束语.....	27
参考文献	28

1. 背景与意义

1.1 项目开发意义

互联网给我们的生产生活带来翻天覆地的变化,比如现在以电子商务平台为基础的网上购物已经成为人们生活的一部分。随着人们生活节奏的加快,学习工作日渐繁忙,留给人们上街购物的时间越来越少,越来越多的人参与到网上购物。网络购物是基于浏览器/服务器应用方式,是利用计算机技术、网络技术和远程通信技术,实现电子化、数字化和网络化的整个商务过程。

随着互联网的不断发展和迅速的普及,网上购物这一新的购物形式已逐渐被大部分人所接受并逐渐取代人们传统的购物观念和理念。人们在电脑前就可以在网上浏览到全国及世界各地的任何商品信息,即方便又迅速地搜索到自己所需要的商品,而安全方便的在线支付和送货上门的服务,使人民更加深切地体会到这一购物方式的优越性。与此同时,电商平台这种新的商业运营模式被越来越多的商家用到竞争中,并得到了大多数客户认可,一些电子商务网站的成立,从整体上降低了企业成本,加快了企业对市场的反应速度,提高了商家的服务质量和竞争力。

但是电子商务平台在带来诸多好处的同时也隐藏着一些安全问题。电子商务简单的说就是利用互联网进行的交易活动,电子商务:"电子"+"商务",从电子商务的定义可以了解电子商务的安全也就相应的分为两个方面的安全:一方面是"电子"方面的安全,因为电子商务的开展必须利用互联网来进行,而互联网本身也属于计算机网络,所以电子商务的第一个方面的安全就是计算机网络的安全,它包括计算机网络硬件的安全与计算机网络软件的安全;另一方面是"商务"方面的安全,是把传统的商务活动在互联网上开展时,由于互联网存着很多安全隐患给电子商务带来了安全威胁,简称为"商务交易安全威胁"。这两个方面的安全威胁也就给电子商务带来了很多问题:

1. 计算机网络安全威胁。电子商务包含"三流":信息流、资金流、物流,"三流"中以信息流为核心为最重要,电子商务正是通过信息流为带动资金流、物流的完成。电子商务跟传统商务的最重要的区别就是以计算机网络来传递信息,促进信息流的完成。计算机网络的安全必将影响电子商务中的"信息流"的传递,势必影响电子商务的开展。计算机网络存在以下安全威胁:黑客攻击、计算机病毒的攻击以及拒绝服务攻击等。

2. 商务交易安全威胁。把传统的商务活动在互联网上进行,由于互联网本身的特点,它存在着很多安全威胁,给电子商务带来了安全问题。互联网存在以下安全隐患:开放性、缺乏安全机制的传输协议、软件系统的漏洞和信息电子化。

3. 计算机网络安全威胁与商务交易安全威胁给电子商务带来的安全问题。具体包括信息泄露、信息篡改、身份识别、信息破坏、破坏信息的有效性、泄露个人隐私等安全威胁。

1.2 国内外现状及技术综述

1.2.1 国内外电子商务发展现状:

1995 年, 亚马逊和易贝在美国成立。此后, 这种以互联网为依托进行商品和服务交易的新兴经济活动, 迅速普及全球。新一轮科技革命和产业变革交汇孕育的电子商务, 极大提高了经济运行的质量和效率, 改变了人类的生产生活方式, 成为世界经济的亮点和新增长点。当前, 全球电子商务呈现以下几个特点:

1. 市场规模不断扩大。根据国际知名调查公司 E-marketer 的数据, 2011 年到 2016 年, 全球网络零售交易额从 0.86 万亿美元增长至 1.92 万亿美元, 年平均增长率达 17.4%。随着全球智能手机保有量不断提升、互联网使用率持续提高、新兴市场快速崛起, 全球网络零售仍将保持两位数增长。预计 2020 年, 全球网络零售交易额将超过 4 万亿美元, 占全球零售总额的比例从 2016 年的 7.4% 增长至 14.6%。跨境电子商务尤其是跨境 B2C (企业对个人) 日益活跃。根据埃森哲的研究报告, 2015-2020 年全球跨境 B2C 年均增速约 27%, 2020 年市场规模将达到 9940 亿美元。

2. 地区差距逐渐缩小。欧美地区电子商务起步早、应用广。2016 年美国网络零售交易额达到 3710 亿美元, 占美国零售总额的比例约 8%。英法德西班牙意大利等五国的市场份额最大, 占欧盟电子商务市场总量的 77.5%; 亚洲地区电子商务体量大、发展快。电子商务起源于欧美, 但兴盛于亚洲。亚洲地区网络零售交易额已占全球市场的 46%。中国、印度、马来西亚的网络零售年均增速都超过 20%。中国网络零售交易额自 2013 年起已稳居世界第一; 拉丁美洲、中东及北非地区电子商务规模小、潜力大; 非洲地域广阔, 人口分布不均, 实体店数量少, 居民购物不便, 电子商务发展存在刚性需求。

3. 电子商务市场的发展前景十分诱人。虽然目前还不能预测电子商务交易模式何时能成为主流模式, 但电子商务的市场发展潜力是无穷的, 因为随着互联网信息技术的广泛应用, 全球网民的数量将进一步增长。全球互联网正在快速普及, 这为电子商务的快速发展提供了良好的基础。全球电子商务快速发展, 成为全球居民消费的重要渠道。2019 年全球网民总数达到 41 亿人, 同比增长 5.3%。全球网民渗透率从 2005 年的 16.8% 上升到 2019 年的 53.6%。近年来全球网络零售总额逐年攀升, 2019 年全球网络零售总额为 3.5 万亿美元, 同比增长 20.73%, 预计 2020 年全球网络零售总额将突破 4 万亿美元, 未来几年仍将持续增长; 随着全球电子商务的持续发展, 网络购物已经成为全球居民的一种重要消费方式, 全球数字支付方式逐渐普及。2019 年, 全球电子钱包在电子商务交易中的使用量占比增长到 42%。未来, 随着移动互联网的快速发展, 电子钱包的使用量将进一步增长。

然而, 随着电子商务及网上购物的发展, 国内外电子商务领域内同样涌现出一些问题。

1. 首先是销售者面临的安全风险:

①网络系统安全性:网络安全是网络支付的核心, 要防止入侵者伪装成合法用户来改变真实数据、取消订单或生成虚假订单。

②竞争者窃取物流信息:恶意竞争者通过他人名义, 虚假订购商品, 从而掌握

相关产品的物流配送以及库存分布情况等信息。

③客户资料及机密数据的窃取:销售者在通过网络平台实施交易时,要防止价格、库存、货源、用户信息等商业核心数据被恶意程序攻击和窃取。

④莫名顶替损害企业形象:销售商需要在网络平台建立唯一合法认证的链接、用户名等,防止不法分子通过莫名顶替的形式,以次充好,恶意破坏企业形象和产品质量形象。

2. 其次是消费者所面临的安全威胁:

①虚假订单:不良商家会使用消费者的名义订购劣质产品,并送货, 上门,消费者在收到商品时可能会被要求支付商品或支付退货运费。

②付款后未收到商品:消费者通过网络付款后,销售商的后台工作人员可能不将订单详情和付款金额转发给发货执行部门,从而窃取客户金额,造成消费者财产损失。

③机密性丧失:消费者在消费过程中有可能会泄漏自己的身份信息,如姓名、工作单位、地址、电话、信用卡号码等信息,销售商在得到客户信息后将个人信息进行非法销售,损害消费者利益。

④拒绝服务:恶意竞争者能够通过网络攻击,向销售商的服务器发送大量虚假订单来浪费其网络资源,使得正常消费者无法登录服务器。

1.2.2 技术综述

1. 加密技术（隐私信息保护）

AES 对称密钥加密技术:美国国家标准技术研究所在 2001 年发布了高级加密标准 (AES)。AES 是一个对称分组密码算法,旨在取代 DES 成为广泛使用的标准。根据使用的密码长度, AES 最常见的有 3 种方案,用以适应不同的场景要求,分别是 AES-128、AES-192 和 AES-256。高级加密标准已然成为对称密钥加密中最流行的算法之一。

RSA 公钥加密技术:算法的名字以发明者的名字命名: Ron Rivest, Adi Shamir 和 Leonard Adleman。但 RSA 的安全性一直未能得到理论上的证明。RSA 加密时使用公钥加密,私钥解密。RSA 的安全性依赖于大数分解。公钥和私钥都是两个大素数(大于 100 个十进制位)的函数。据猜测,从一个密钥和密文推断出明文难度等同于分解两个大素数的积。RSA 可用于数字签名,私钥签名公钥验签。具体操作时考虑到安全性和 m 信息量较大等因素,一般是先作 HASH 运算。

2. 支付接口设置（参数传递）

使用 GET 方法将数字信封 (AES-RSA 混合加密) 添加到链接后发送给银行。

3. 密码技术

密码学是一门古老又新兴的学科,网络技术的发展更促进了密码技术的应用和发展。加密技术、数字信封、数字摘要技术、数字签名和数字证书等相关密码技术,能够保障电子商务双方的合法权益免受侵害。

4. 身份认证技术

身份认证是实现网络安全的基石。在实施网络支付时,交易双方必须通过特定的证机制来证明各自身份,验证用户的身份-与所宣称的是否一致, 从而实现针对不同用户的访问控制和记录。

2. 需求分析

2.1 总体需求

总体需求为管理员页面的实现和商城页面的实现以及确保前后端信息传递的安全以及与银行交互时信息的安全。

管理员要实现添加商品、修改商品信息和删除商品的功能。

商城要实现用户注册、登录、添加购物车、移出购物车、购买商品并支付。

2.2 功能需求

1. 管理员：

需要实现三个功能：

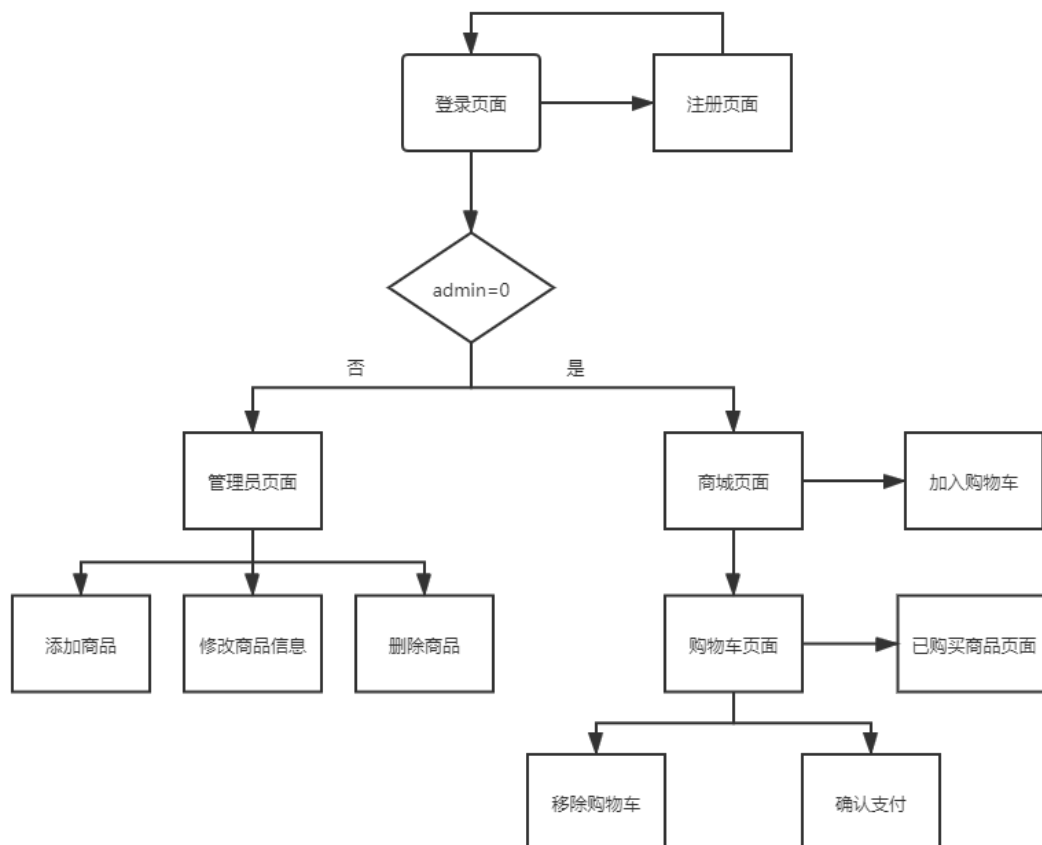
- ① 添加商品（输入商品名称、售价、库存、添加图片）
- ② 修改商品信息（名称、售价、库存）
- ③ 删除不再售卖的商品

2. 商城：

- ① 在商城主页面展示商品信息。
- ② 用户添加购物车并在购物车页面可以查看、移出已添加商品。
- ③ 买家身份认证（加密）。
- ④ 买家帐户注册及维护（加密）。
- ⑤ 订单确认（加密）。
- ⑥ 付款确认（加密），连接调用虚拟支付系统的接口。
- ⑦ 完成交易，接收支付系统发来的付款完成信息后，交易成功，清空购物车，将已购买商品显示在已购买商品页面，并通知买家。

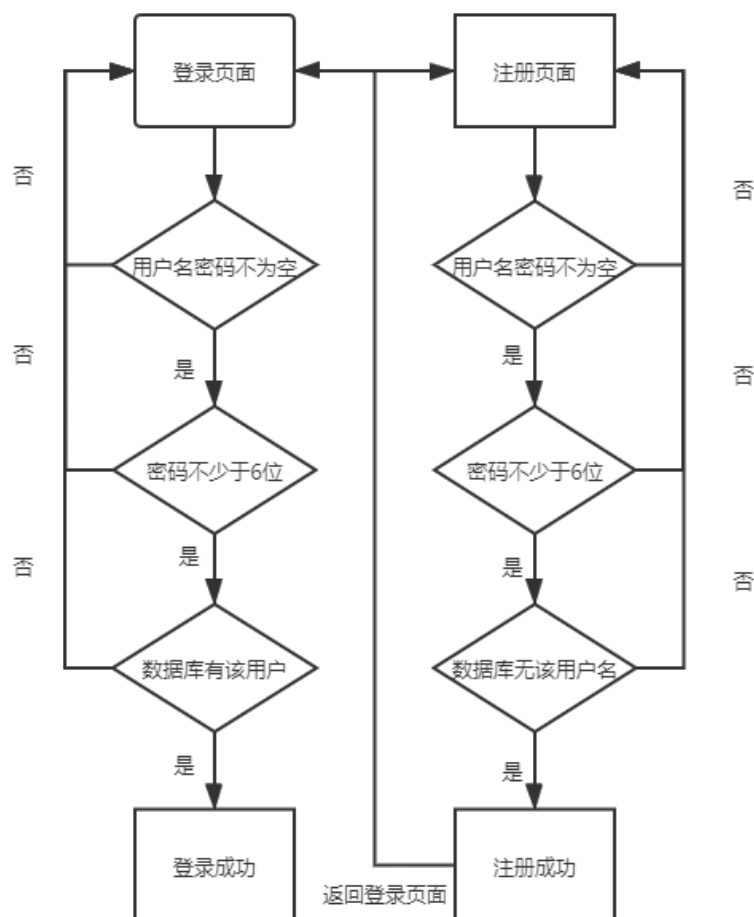
3. 概要设计

3.1 总体功能流程图（基本结构框架）



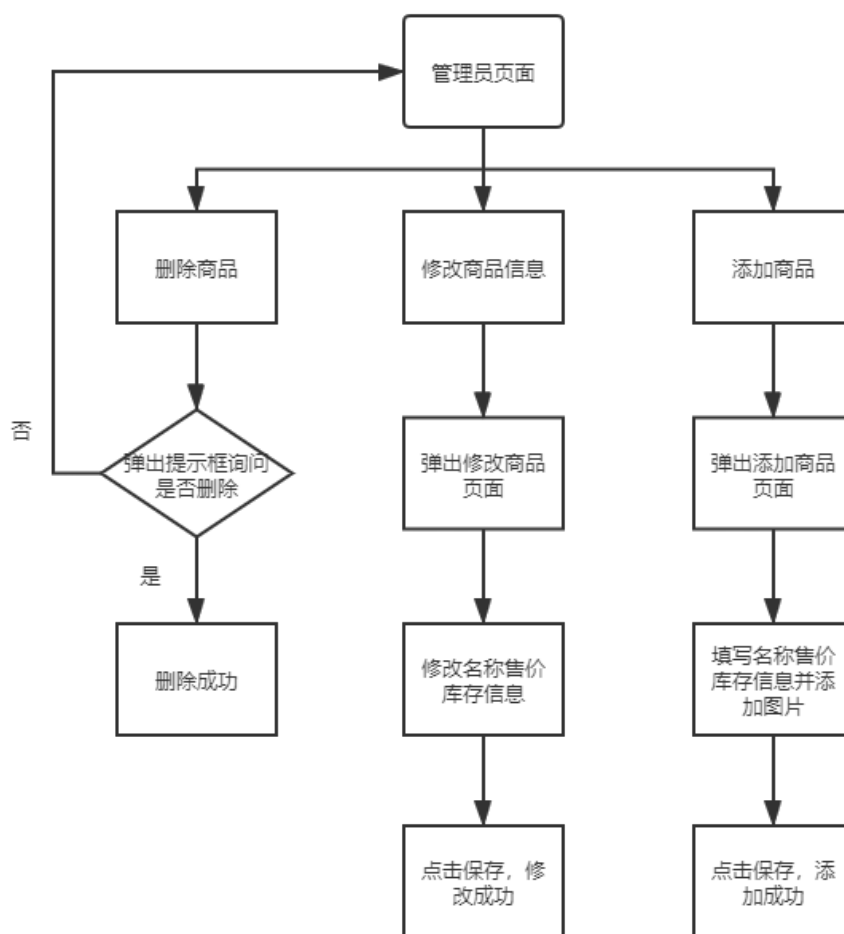
3.2 功能结构设计

3.2.1 登录注册功能结构设计



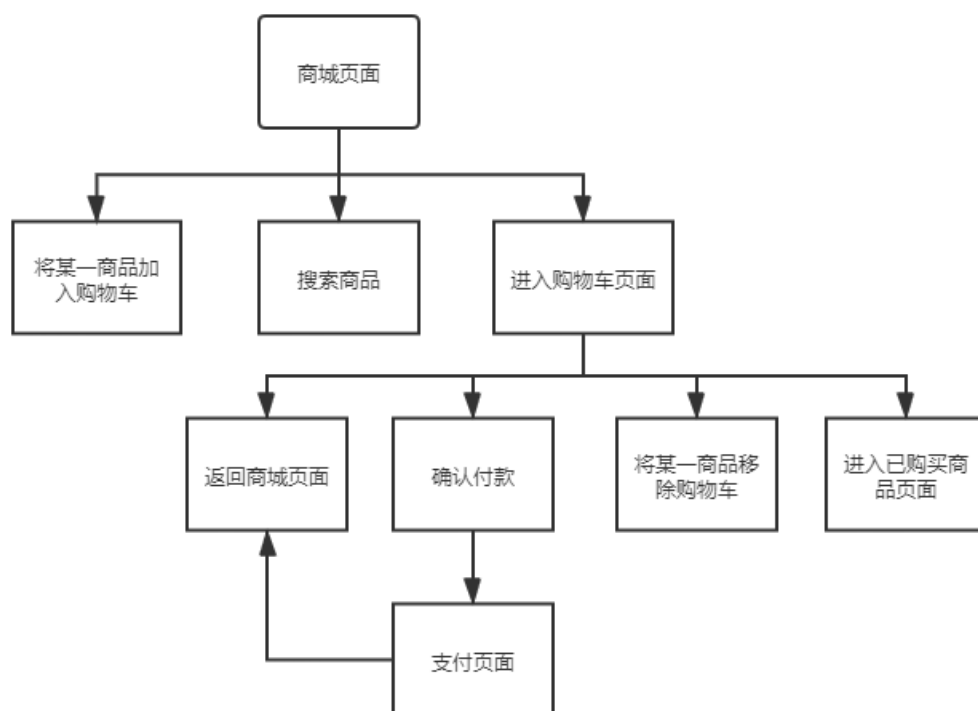
用户登录或注册时都会先经过 login.js 和 register.js 检查格式，格式正确后再将表单信息提交到 LoginServlet 和 RegisterServlet 进行进一步数据库检查，判断是否有该用户。

3.2.2 管理员功能结构设计



管理员页面实现了三个功能，删除商品、添加商品和修改商品信息。添加商品和修改商品信息都用到了新的页面，会以小窗的形式弹出。修改商品时商品名称、金额、库存会默认显示原有信息，修改后会更新数据库。

3.2.3 商城功能结构设计



商城主页面在 `jsp` 实现了动态添加购物车，将添加的商品序列号（`gid`）传到 `EshopServlet` 进行处理，进行数据库操作。商城主页面还有进入购物车按钮，点击即可进入购物车页面。购物车页面显示已添加到购物车的所有商品、购物车中商品的数量以及总金额。用户可在购物车页面点击支付跳转支付页面，也可以点击查看已购买商品按钮前往已购买商品页面，已购买商品页面显示用户历史购买的全部商品及数量。在已购买商品页面可以点击返回购物车按钮返回购物车页面。

4. 详细设计

本系统采用 mysql+jsp+servlet+tomcat 开发

4.1 数据库设计

4.1.1 用户数据表（user 表）

名	类型	长度	小数点	不是 null	
uid	int	0	0	<input checked="" type="checkbox"/>	 1
username	varchar	100	0	<input checked="" type="checkbox"/>	
password	varchar	100	0	<input checked="" type="checkbox"/>	
admin	int	0	0	<input checked="" type="checkbox"/>	

此表一行存储一个用户的信息。

uid 为用户在商城注册后自动分配的用户 id，不会重复，唯一标识用户。

username 为用户名，也不会重复，但一般索引用 uid。

password 为用户密码，注册时用户名和密码都会存到此表。

admin 是管理员标识位，普通用户 admin 为 0，管理员 admin 为 1，登录时后端会根据 admin 的值将用户登录到商城页面或管理员页面。一般注册只能注册普通用户，即 admin 默认为 0。

4.1.2 商品数据表（goods 表）

名	类型	长度	小数点	不是 null	
gid	int	0	0	<input checked="" type="checkbox"/>	 1
name	varchar	255	0	<input checked="" type="checkbox"/>	
price	float	10	2	<input checked="" type="checkbox"/>	
quantity	int	0	0	<input checked="" type="checkbox"/>	
link	varchar	255	0	<input type="checkbox"/>	


此表一行存储一个商品的信息。

gid 为 good id，即商品序列号，管理员添加商品时自动分配，gid 唯一，用户索引商品。

name、price 和 quantity 分别为商品名称、售价和库存，在管理员添加商品时会存入该表。

link 在管理员添加商品图片时会调整 url 为相对路径以方便项目找到图片。

4.1.3 购物车数据表（shopping_cart 表）

名	类型	长度	小数点	不是 null	
▶ id	int	0	0	<input checked="" type="checkbox"/>	 1
uid	int	0	0	<input checked="" type="checkbox"/>	
gid	int	0	0	<input checked="" type="checkbox"/>	
number	int	0	0	<input checked="" type="checkbox"/>	

此表一行存储一个用户购物车的一个商品的信息。即一个用户的购物车所有商品需用多行表示。

id 为此表的 key，无实际作用。

uid 对应该用户，gid 对应某商品，number 为加入购物车的数量。

若用户 1（uid）将商品 23（gid）加入购物车，若该用户购物车没有此商品，则会在此表添加一行（id 自动填充，1,23,1），若购物车已有该商品，则查找 uid=1 且 gid=23 的一行将 number 加一，移出购物车同理，查找该行后 number 减一。

4.1.4 订单数据表（completedorder 表）

名	类型	长度	小数点	不是 null	
▶ uid	int	0	0	<input checked="" type="checkbox"/>	
gid	int	0	0	<input checked="" type="checkbox"/>	
number	int	0	0	<input checked="" type="checkbox"/>	
dealid	varchar	255	0	<input checked="" type="checkbox"/>	
id	int	0	0	<input checked="" type="checkbox"/>	 1

此表相对较复杂，用于记录某用户的某次订单信息（订单是否完成，订单包含哪些商品以及商品数量）

① 例如用户 2（uid）点击支付，会生成此次订单的订单号，将订单号(dealid)和用户 id(uid)存入该表信道一行：

uid	gid	number	dealid	id
2	0	0	JO5IWUE8	86

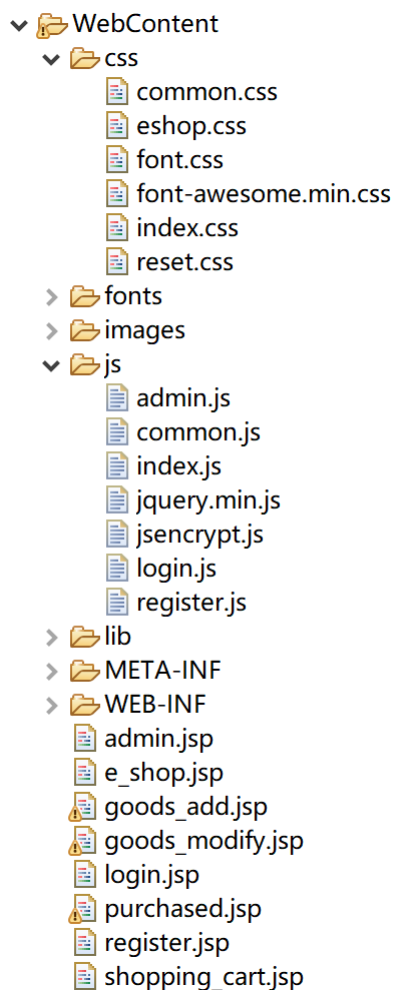
gid 和 number 设为 0 表示订单尚未完成。

② 用户付款后会将该行 gid 和 number 位设为-1（付款后银行会返回 dealid，商城后台会在表中查找是否有该订单且订单未完成，若有，则设为-1）表示订单已完成，然后会在表中继续添加此次订单购买的商品（gid）和数量（number），用 dealid 表示此次订单。

uid	gid	number	dealid	id
2	-1	-1	JO5IWUE8	86
2	5	1	JO5IWUE8	87
▶ 2	6	1	JO5IWUE8	88

4.2 jsp 开发前端页面

4.2.1 项目前端目录



4.2.2 以 e_shop.jsp 为例展示前端开发

e_shop.jsp 分为三块，分别为 head、section 和 footer

head 板块有欢迎信息和搜索框。如果用户没有登录，会将欢迎用户改为提示请登录：

```

<%
    if (session.getAttribute("username") != null) {
%>
<div class="hello">Hello</div>

<div class="username"><%=session.getAttribute("username")%></div>
<div class="hello">! Welcome to Lsy E-shop</div>
<%
    }
%>
<%
    if (session.getAttribute("username") == null) {
%>
<div class="hello">Hello</div>
<div class="hello">
    <a href="login.jsp">请先登录</a>
</div>
<%
    }
%>

```

在 jsp 可以将 Java 代码和 html 语言穿插写来动态实现页面。

section 板块展示所有商品，先从数据库读取所有商品，再循环打印即可：（仅以部分代码进行展示）

```

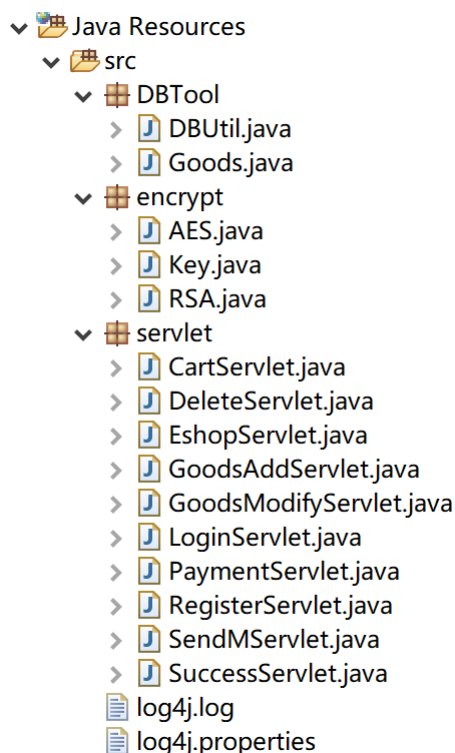
<%
    for (int i = 0; i < goodsList.size() && goodsList.get(i).getQuantity() != 0; i++) {
%>
<li>
    <div class="rec_pic">
        <%
            out.print("<img src='" + goodsList.get(i).getLink() + "'width='220' height='220'>");
        %>
    </div>

```

footer 板块起装饰作用，无实际功能。

4.3 后端 servlet 实现

4.3.1 项目后端目录



4.3.2 DBTool 包和 encrypt 包介绍

DBUtil 类实现连接数据库的功能，在 jsp 和 servlet 中都会调用该类。

Goods 类构造 Goods 类型变量，包括商品的各个属性的 getter 和 setter 方法。

AES 类实现 AES 加密、AES 解密。

RSA 类实现了 RSA 公钥加密、RSA 私钥解密、RSA 私钥签名、RSA 公钥验签。其中 RSA 解密有两个函数，分别为 URLsafe base64 和 base64 编码，因为交互时使用 GET 方法传参，需要 URLsafe 编码加密自然也需要 URLsafe 编码解密。而前后端加密在 js 中实现时用的是 base64 编码，所以后端解密需要 base64 编码格式的 RSA 解密。

Key 类存储商城的公钥和私钥，存储银行的公钥和 ip 地址，实现了随机字符串生成的静态函数，用户生成操作序列号和订单号。

4.3.3 servlet 包介绍

加密解密以在其他部分介绍，在此就不赘述了。

①

LoginServlet 实现获取 login.jsp 中表单提交的信息，访问数据库 user 表判断用户是否存在，若存在是否为管理员。

RegisterServlet 实现获取 register.jsp 中表单提交的信息，访问 user 表判断用户名是否存在，若不存在注册成功，在 user 表添加该用户；若已存在提示后返回注册页面。

EshopServlet 获取 e_shop.jsp 中点击添加购物车时提交的商品信息，从 session

获取 uid, 访问数据库 shoppingcart 表, 若表中该用户已有该商品, 将该行 number 加一, 若没有则添加, 然后将 goods 表中该商品库存减一。

CartServlet 处理 shopping_cart.jsp 中移出购物车按钮提交的商品信息, 类似 EshopServlet。

②

SendMServlet 处理 shopping_cart.jsp 页面确认支付按钮提交的金额信息, 将需要的信息处理后发送给银行, 具体实现在 4.4 中有详细描述。

PaymentServlet 处理银行返回的 payid 并根据 payid 跳转支付页面。

SuccessServlet 处理银行支付成功后返回的订单号及签名, 若验签正确且订单号正确, 订单未完成, 处理数据库: 清空用户购物车, 将订单设置为已完成, 将此次订单商品信息添加到数据库, 具体实现可看 4.1 对数据库的介绍。

③

GoodsAddServlet 处理添加商品页面保存按钮提交的信息, 添加该商品, 更新数据库。

GoodsModifyServlet 处理更改商品页面保存按钮提交的信息, 更改数据库中该商品的信息。

DeleteServlet 处理管理员页面删除按钮提交的商品信息, 删除商品, 更新数据库。

4.4 系统安全实现

4.4.1 前后端交互加密——RSA 公钥加密:

本系统共有两种方式, 一是在 js 函数中实现加密, 二是在 jsp 动态实现加密:

4.4.1.1 对用户输入的信息用 js 实现加密

下载 jsencrypt 库调用 encrypt 函数对 form 表单提交的信息进行加密。

login.js: 对 form 表单提交的用户名和密码用电商的公钥加密:

```
var encryptor = new JSEncrypt() // 创建加密对象实例
var publicKey = "MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDstiYKiePxSEJOiLskFkxwcGPC"
+ "SUIHE4cm00WWzC60zRQYEKUXRauwQoVbe+6wiVgSyDn3yvezfdMja8wTPaUsPh3i"
+ "GYJPWAXZlH5fjv5G02r73LbFJLgveUIjSgykN50smt6S4+fzXg41nrp5Vasg9t9S"
+ "NzjjU7XTYSG+75EuJwIDAQAB";
encryptor.setPublicKey(publicKey); // 设置公钥
form.enpassword.value = encryptor.encrypt(form.password.value); // 对内容进行加密
form.enusername.value = encryptor.encrypt(form.username.value);
var a = form.password.value.length;
form.username.value = '';
form.password.value = '';
for (var i = 0; i < a; i++) {
    form.password.value = form.password.value + '.';
}
```

admin.js: 对修改页面以及添加页面的表单中售价、库存、商品序列号 (gid) 以及操作序列号 (order) 用电商的公钥进行加密:

```
function beforeSubmitModify(form) {
```

```

var encryptor = new JSEncrypt() // 创建加密对象实例
var publicKey = "MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDstiYKiePxseJ0iLskFkxwcGPC"
    + "SUIHE4cm00WwzC60zRQYEKUXRauwQoVbe+6wiVgSyDn3yvezfdMja8wTPaUsPh3i"
    + "GYJPWAXZlH5fjv5G02r73LbFJLgveUIjSgykN50smt6S4+fzXg41nnp5Vasg9t9S"
    + "NzjjU7XTYSG+75EuJwIDAQAB";
encryptor.setPublicKey(publicKey);//设置公钥
form.engid.value = encryptor.encrypt(form.engid.value);
form.enorder.value = encryptor.encrypt(form.enorder.value);
form.enname.value = encryptor.encrypt(form.enname.value);
form.enquantity.value = encryptor.encrypt(form.enquantity.value);
form.enprice.value = encryptor.encrypt(form.enprice.value);
return true;
}

```

function beforeSubmitAdd(form) { } 与上图 beforeSubmitModify() 函数类似。

register.js 中加密与 login.js 类似。

4.4.1.2 前端已有的信息发送到后端时在 jsp 中动态加密

eshop.jsp:

首先从数据库动态获取商品信息:

```

<%
    List<Goods> goodsList = new ArrayList<Goods>();
    //num是String类型的八位序列号
    String num = Key.genString();
    session.setAttribute("num", num);
    try {
        Connection conn = DBUtil.getConnection();
        Statement st = conn.createStatement();
        String sql = "select* from goods";
        ResultSet rs = st.executeQuery(sql);
        while (rs.next()) {
            /* System.out.print("找到!!"); */
            Goods g = new Goods();
            g.setGid(rs.getInt("gid"));
            g.setName(rs.getString("name"));
            g.setPrice(rs.getFloat("price"));
            g.setQuantity(rs.getInt("quantity"));
            g.setLink(rs.getString("link"));
            if(g.getQuantity()!=0){
                goodsList.add(g);
            }
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    } finally {
        DBUtil.close();
    }
%>

```

点击加入购物车时在 form 表单中动态传输 RSA 公钥加密后的商品序列号(gid)、库存以及序列号

```

<%
    out.print("<input type='hidden' name='engid' value='"
        + RSA.encrypt(goodsList.get(i).getGid() + "", Key.getMyPublicKey()) + "'>");
    out.print("<input type='hidden' name='enquantity' value='"
        + RSA.encrypt(goodsList.get(i).getQuantity() + "", Key.getMyPublicKey()) + "'>");
    out.print("<input type='hidden' name='ennum' value='" + RSA.encrypt(num, Key.getMyPublicKey()) + "'>");
%>
<input type="submit" name="submit" value="加入购物车">

```

4.4.2 防重放攻击——session 用户 id 确认、操作序列号与订单号

4.4.2.1 前后端数据传输放重放攻击——用户 id 确认与操作序列号

在前端用户执行某一操作时会发送信息到后端，即使信息已加密，攻击者在截获信息后仍然可以重新发送数据包到相应的后端实现登录、添加购物车、修改商品信息等用户执行的操作。在添加、移除购物车之类的操作发送数据到后端后后端 servlet 会从 session 中获取用户 id（之前登录时存入 session），然而攻击者没有登录过，故无法从 session 获取 id，不会执行操作；

4.4.2.2 交互过程放重放攻击——订单号

用户点击确认支付后商城会向银行发送金额等信息，其中就包括订单号。订单号为随机生成的八位字符串：

```

private static final String[] GENERATE_SOURCE = new String[]{"0", "1", "2", "3", "4", "5", "6", "7",
    "8", "9", "A", "B", "C", "D", "E", "F", "G", "H", "I", "J",
    "K", "L", "M", "N", "O", "P", "Q", "R", "S", "T", "U", "V",
    "W", "X", "Y", "Z"};

public static String genString() {
    StringBuilder sb = new StringBuilder(8);
    for (int i = 0; i < 8; i++) {
        final int index = new Random().nextInt(STR_LEN);
        sb.append(GENERATE_SOURCE[index]);
    }
    return sb.toString();
}

```

订单号重复的概率可以忽略。将订单号（dealid）用 aes 加密后发送给银行，最后支付成功后银行会将订单号用商城公钥加密发送回商城，商城会在 completedorder 表中寻找该订单号，若订单为未完成状态（原理可参考 4.1），则正常进行。若为重放攻击，重新发送订单完成信息给商城，商城解密订单号即可查询到订单已完成，实现了放重放功能。

4.4.3 交互过程安全保障——数字信封与订单号

与银行交互过程会将此次订单的金额、商城的银行卡号生成哈希后用 RSA 算法用商城私钥签名，然后将金额和卡号用 aes 加密，在将 aes 秘钥和订单号用 RSA 算法用银行公钥加密，将这些信息发送给银行。（用银行的公钥加密时银行的公钥是与 CA 交互获得银行的证书从证书中获取的，商城也会定期对银行证书

使用 CA 提供的接口进行证书验证，验证是否有效)

```
String password = rs.getString("password");
//使用用户密码生成aes密钥
String aes_key = AES.genAESKey(password);
//用aes密钥加密金额和商城卡号
String amount_c = AES.encryptNew(amount, aes_key);
String card_c = AES.encryptNew(card, aes_key);
//用md5算法为金额和卡号生成摘要并对摘要签名
MessageDigest md = MessageDigest.getInstance("MD5");
byte[] hash_oi = md.digest((amount_c+card_c).getBytes());
String hash = bytesToHexString(hash_oi);
String signature = RSA.sign(hash,Key.getMyPrivateKey());
//rsa算法加密订单号和aes密钥
String dealid = RSA.encrypt(dealid, Key.getBankPublicKey());
String aes = RSA.encrypt(aes_key,Key.getBankPublicKey());
response.sendRedirect(Key.getIP()+"deal/?amount=" + amount_c
    + "&card=" + card_c + "&aes_key="+aes+"&deal_identify="+dealid
    + "&signature="+signature);
```

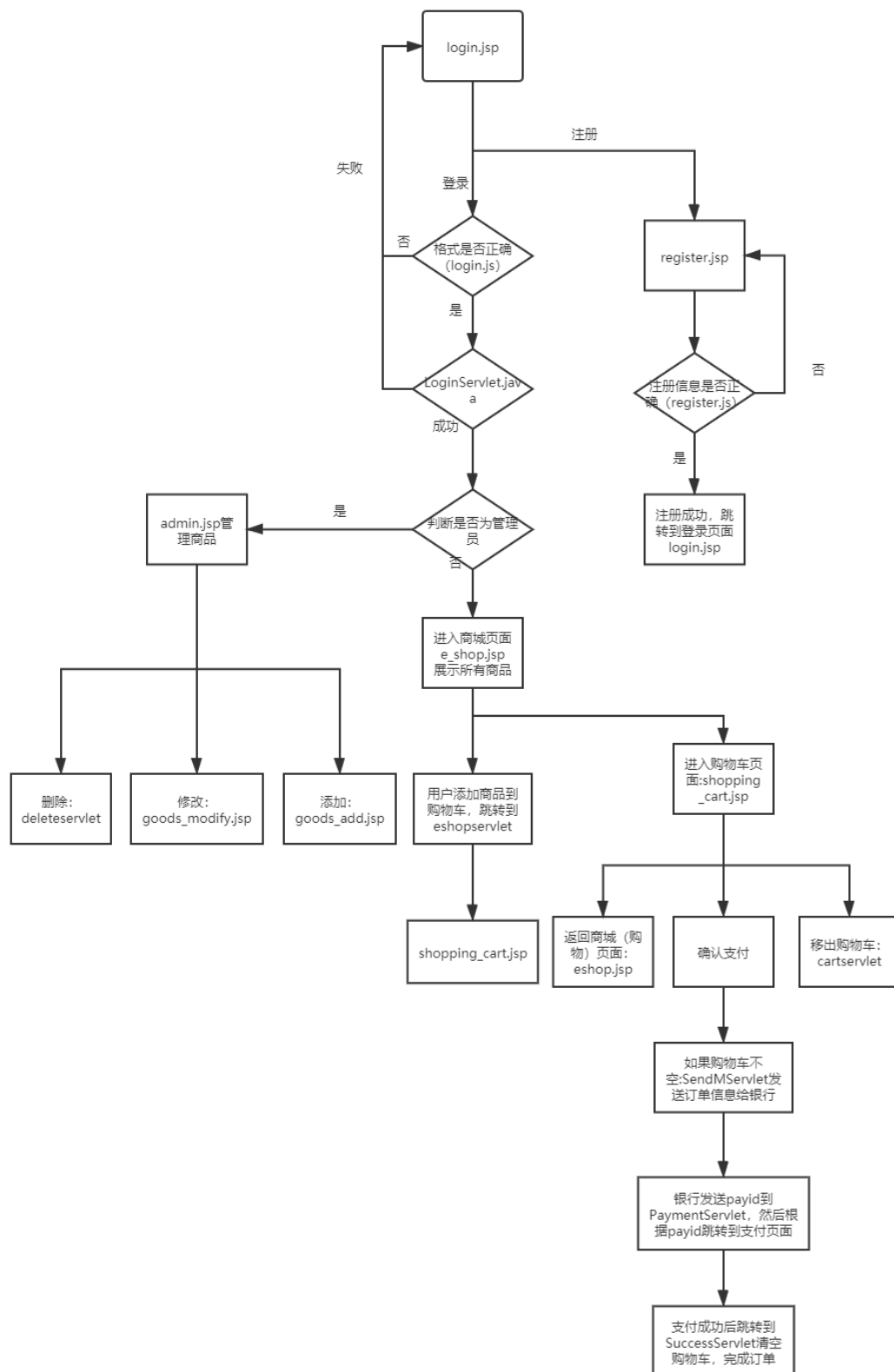
银行需要用商城私钥验签证明商城身份，再用银行私钥解密得到 aes 密钥证明银行身份。敌手没有商城私钥也没有银行私钥，故无法伪造信息攻击交互过程。

银行收到这些信息后进行验签、解密，验签、解密成功后生成支付订单号 payid，并用自己的私钥生成签名一起发送给商城，商城接收后进行验签，检验发送方身份是否为银行。若验签成功则跳转 payid 对应的支付页面：

```
boolean sign = RSA.verify(payId, signature, Key.getBankPublicKey());
if (sign) {
    logger.info("获得payid并对银行验签成功，跳转支付页面。");
    response.sendRedirect(Key.getIP()+"pay/" + pid + "/");
} else {
    String a = URLEncoder.encode("跳转支付页面失败！请重新登录", "UTF-8");
    out.print("<script>alert(decodeURIComponent('" + a
        + "')) ;window.location.href='shopping_cart.jsp'</script>");
    out.flush();
    out.close();
}
```

用户在支付页面付款后银行再发送商城此次订单的订单号 dealid（用商城公钥加密）以及签名，商城验签成功后解密 dealid 进行后续处理（在 4.1 及 4.3 中讲了如何处理）。

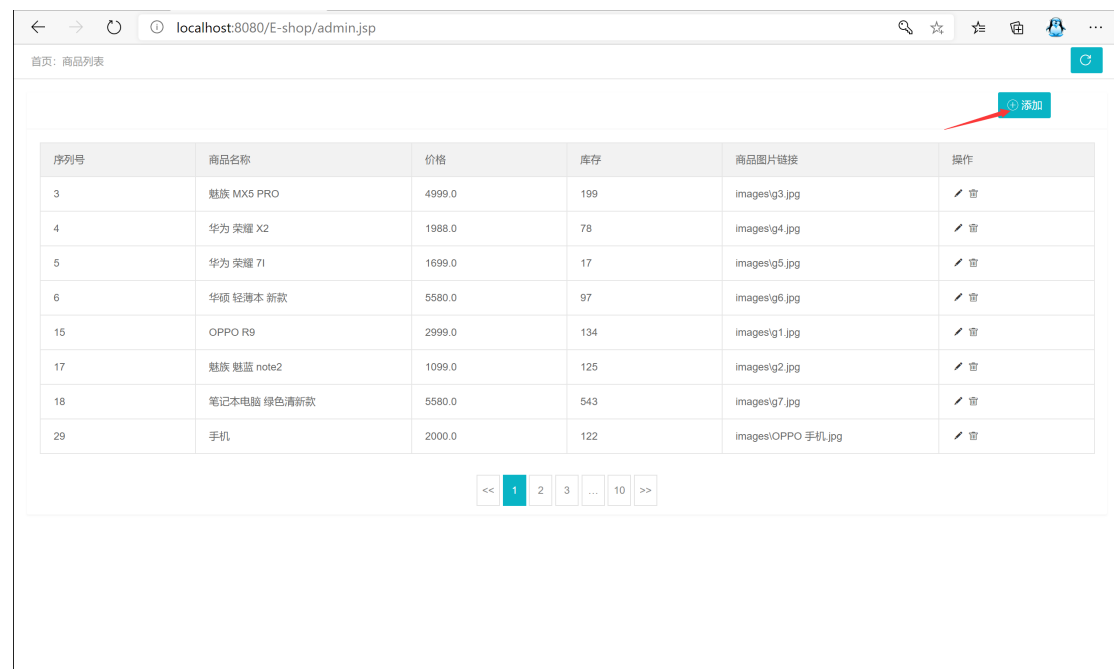
4.5 详细功能结构流程图



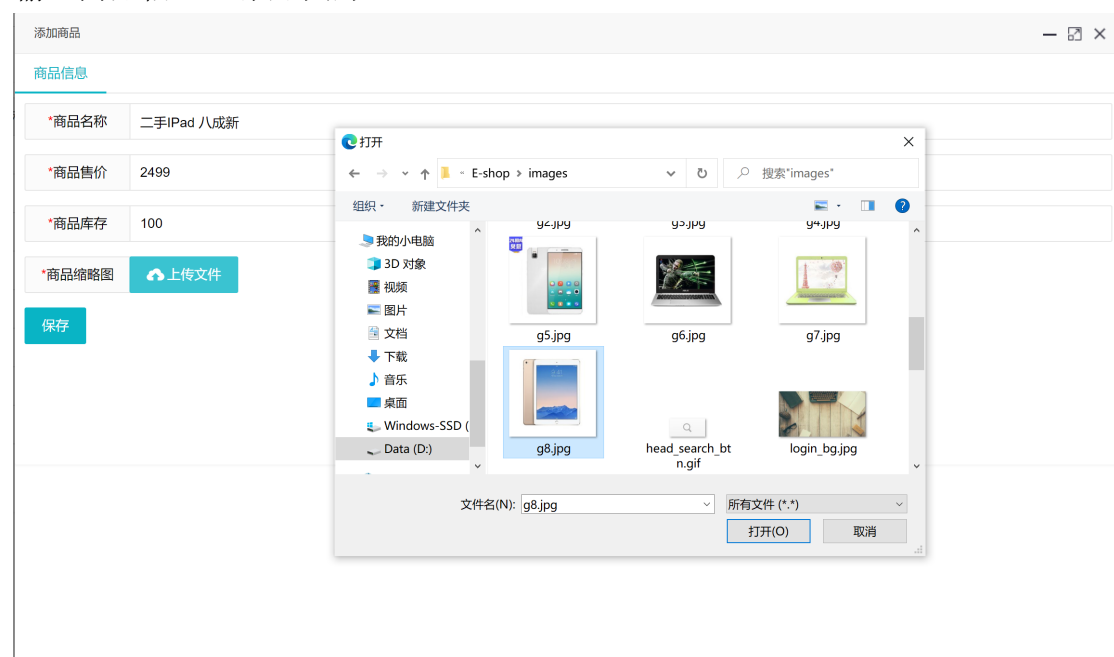
5. 实现与测试

5.1 管理员页面

5.1.1 添加商品



输入商品信息，添加图片：



添加成功：

localhost:8080 显示

添加商品成功!

确定

序列号	商品名称	价格	库存	商品图片链接	操作
3	魅族 MX5 PRO	4999.0	199	images/g3.jpg	✎ 查
4	华为 荣耀 X2	1988.0	78	images/g4.jpg	✎ 查
5	华为 荣耀 7i	1699.0	17	images/g5.jpg	✎ 查
6	华硕 轻薄本 新款	5580.0	97	images/g6.jpg	✎ 查
15	OPPO R9	2999.0	134	images/g1.jpg	✎ 查
17	魅族 魅蓝 note2	1099.0	125	images/g2.jpg	✎ 查
18	笔记本电脑 绿色清新款	5580.0	543	images/g7.jpg	✎ 查
29	手机	2000.0	122	images/OPPO 手机.jpg	✎ 查
30	二手iPad 八成新	2499.0	100	images/g8.jpg	✎ 查

5.1.2 修改商品信息

18	笔记本电脑 绿色清新款	5580.0	543	images/g7.jpg	✎ 查
----	-------------	--------	-----	---------------	-----

编辑

商品信息

*商品名称

笔记本电脑 绿色清新款

*商品售价

5580.0

*商品库存

543

保存

可以修改原来的商品信息（名称、售价、库存）

修改商品售价和库存：

*商品名称

笔记本电脑 绿色清新款

*商品售价

4900.0

*商品库存

200

保存

修改成功：

18	笔记本电脑 绿色清新款	4900.0	200	images\g7.jpg	✎ 查
----	-------------	--------	-----	---------------	-----

5.1.3 删除功能

29	手机	2000.0	122	images\OPPO 手机.jpg	✎ 查
----	----	--------	-----	--------------------	-----

提示要求确认删除操作：

信息

提示：

点击确认后将此商品删除！

您确定继续此操作吗？

确定

取消

删除成功：

localhost:8080 显示

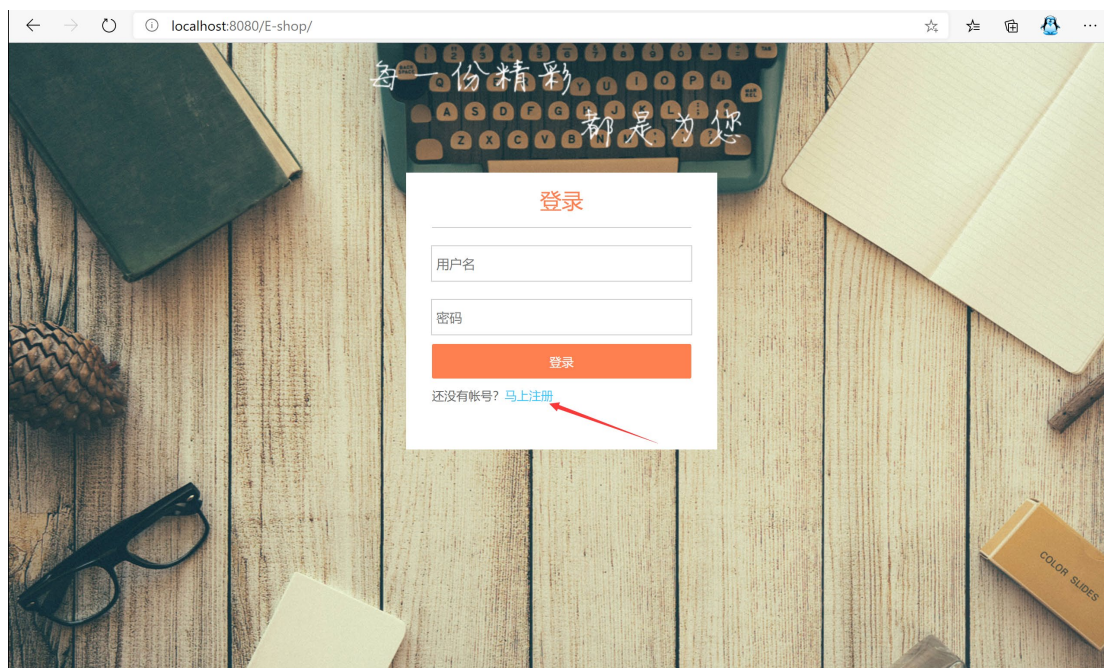
删除成功！

确定

序列号	商品名称	价格	库存	商品图片链接	操作
3	魅族 MX5 PRO	4999.0	199	images\g3.jpg	✎ 查
4	华为 荣耀 X2	1988.0	78	images\g4.jpg	✎ 查
5	华为 荣耀 7i	1699.0	17	images\g5.jpg	✎ 查
6	华硕 轻薄本 新款	5580.0	97	images\g6.jpg	✎ 查
15	OPPO R9	2999.0	134	images\g1.jpg	✎ 查
17	魅族 魅蓝 note2	1099.0	125	images\g2.jpg	✎ 查
18	笔记本电脑 绿色清新款	4900.0	200	images\g7.jpg	✎ 查
30	二手iPad 八成新	2499.0	100	images\g8.jpg	✎ 查

5.2 普通用户登录和注册页面

点击注册：



进入注册页面：



要求用户名尚未注册且两次密码相同。
注册成功后进入商城页面。

5.3 商城页面（商城主页面、购物车页面）

商城主页面：



在商城页面可以看到管理员添加、删除、修改商品信息的操作都成功执行：



笔记本电脑 绿色清新款

加入购物车

售价 ¥4900.0

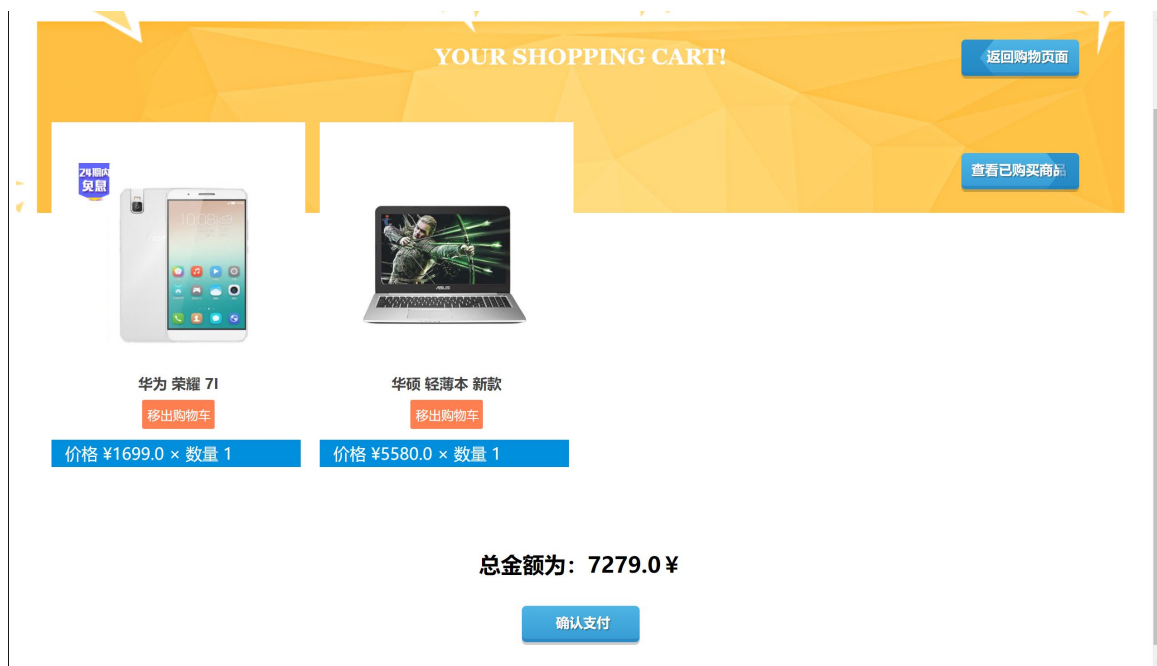


二手iPad 八成新

加入购物车

售价 ¥2499.0

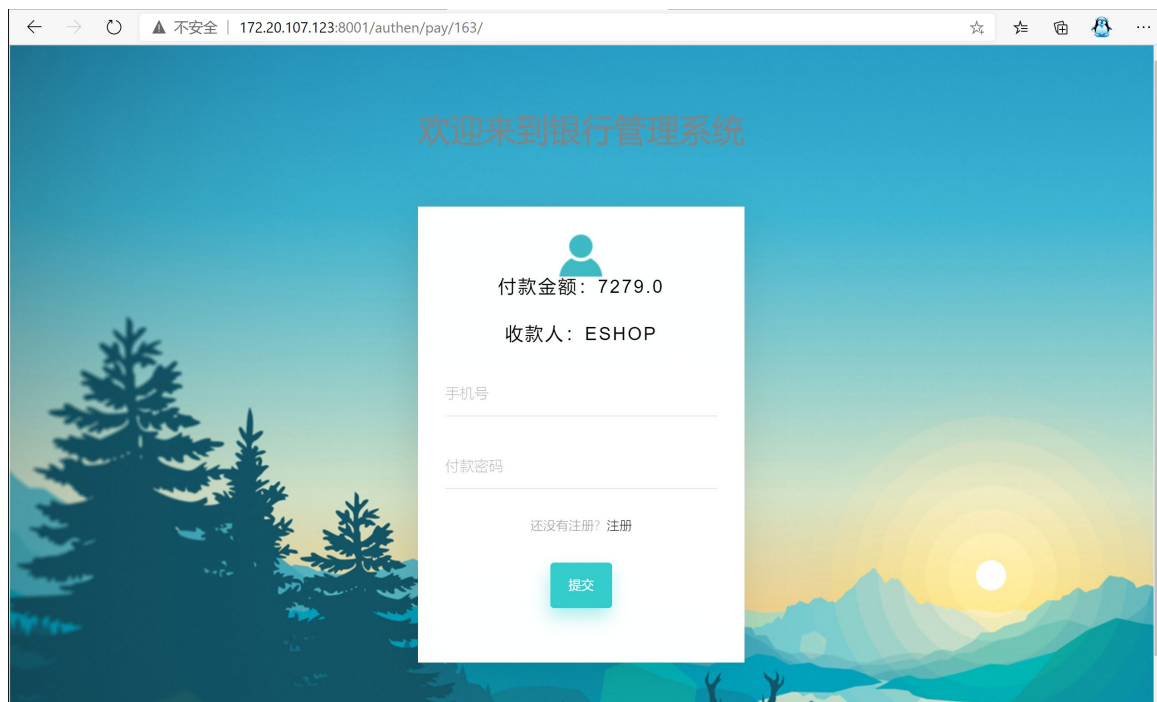
点击查看购物车进入购物车页面：



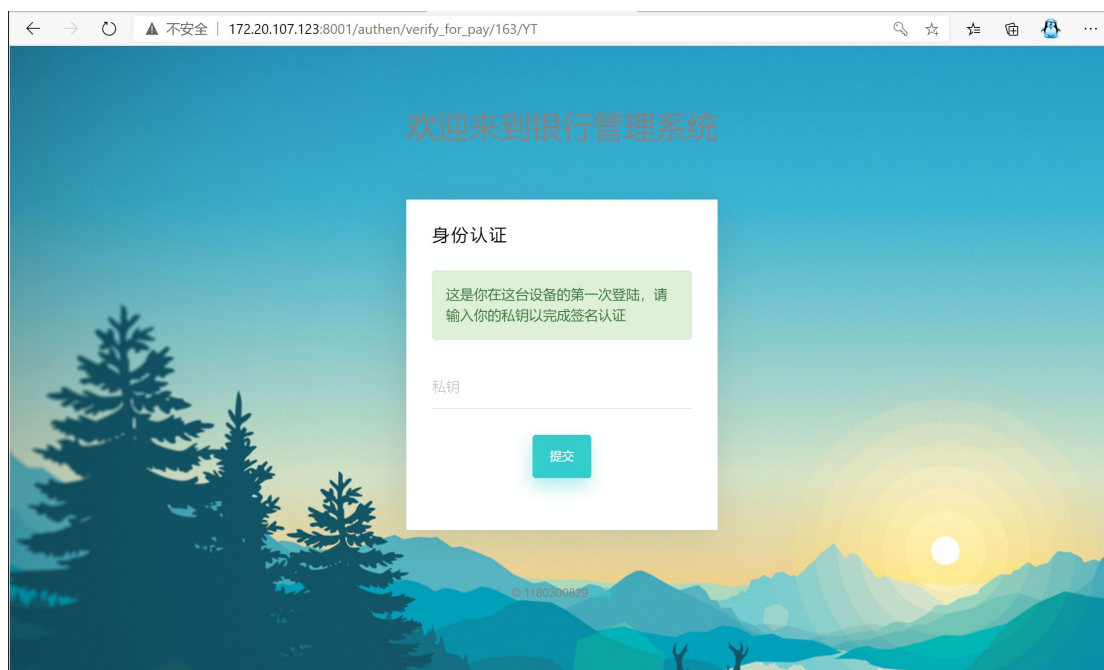
点击确认支付即可跳转到支付页面

5.4 支付操作、支付过程的页面跳转以及支付后已购买商品页面：

点击支付后发送信息到银行后端，银行返回支付订单号，根据支付订单号跳转到支付页面：（可以看到 url 最后跟着支付订单号）



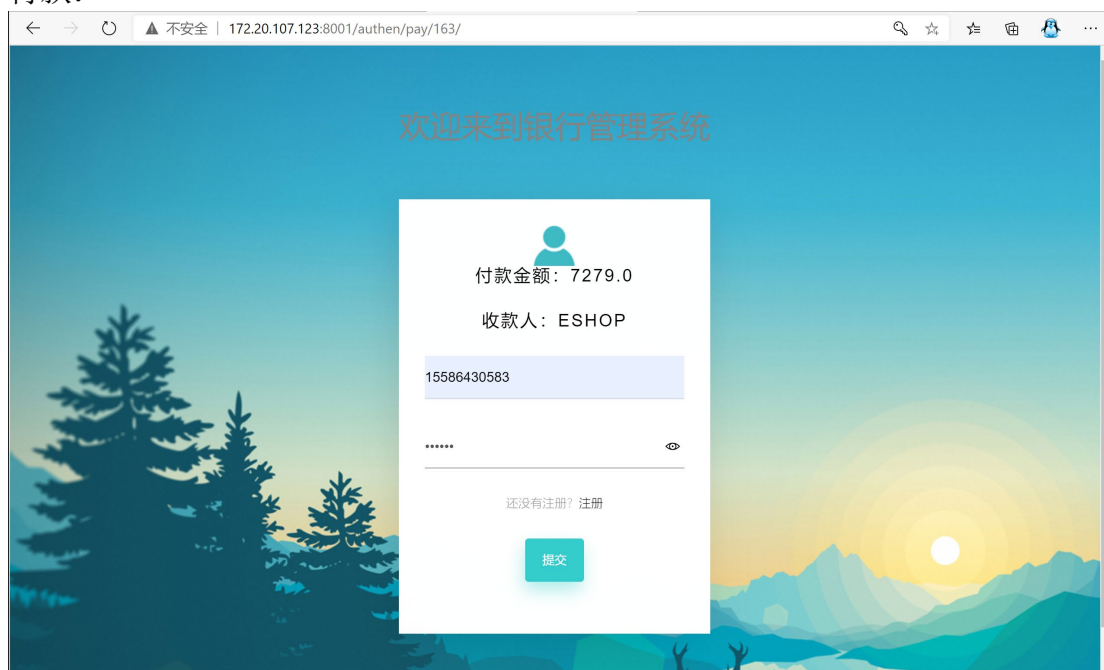
第一次支付为了验证身份会需要输入公钥（具体实现为虚拟银行功能，再次不详述）：



可以看到 completedorder 表多了一行表示该用户的此次订单正在进行:

uid	gid	number	dealid	id
2	0	0	JOSIWUE8	86

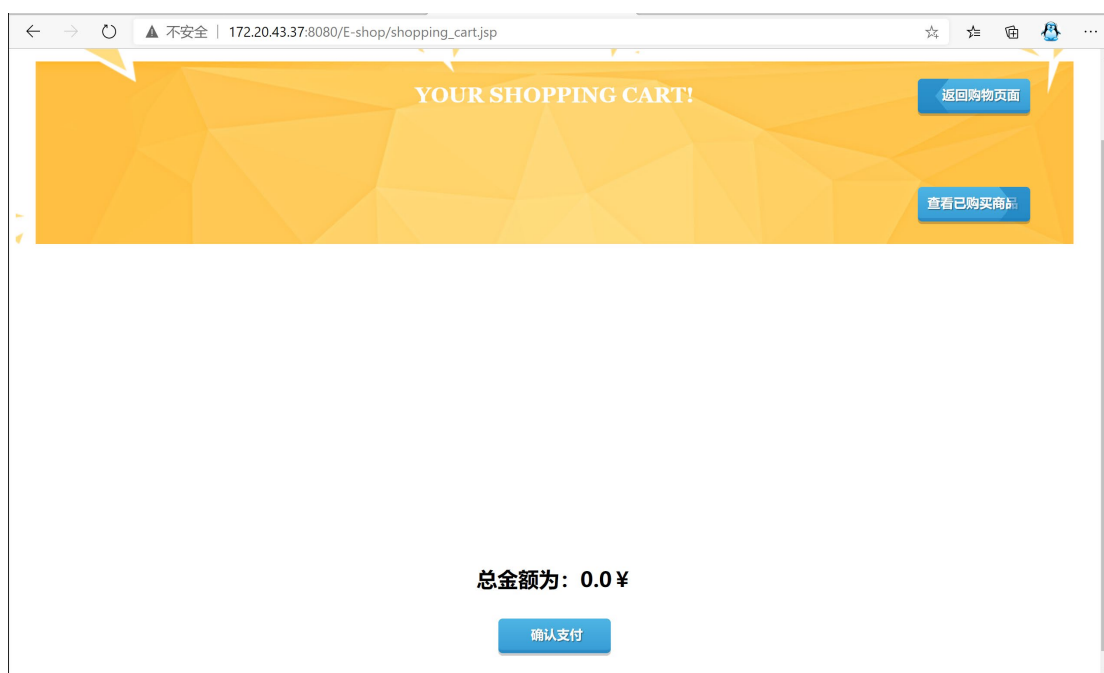
付款:



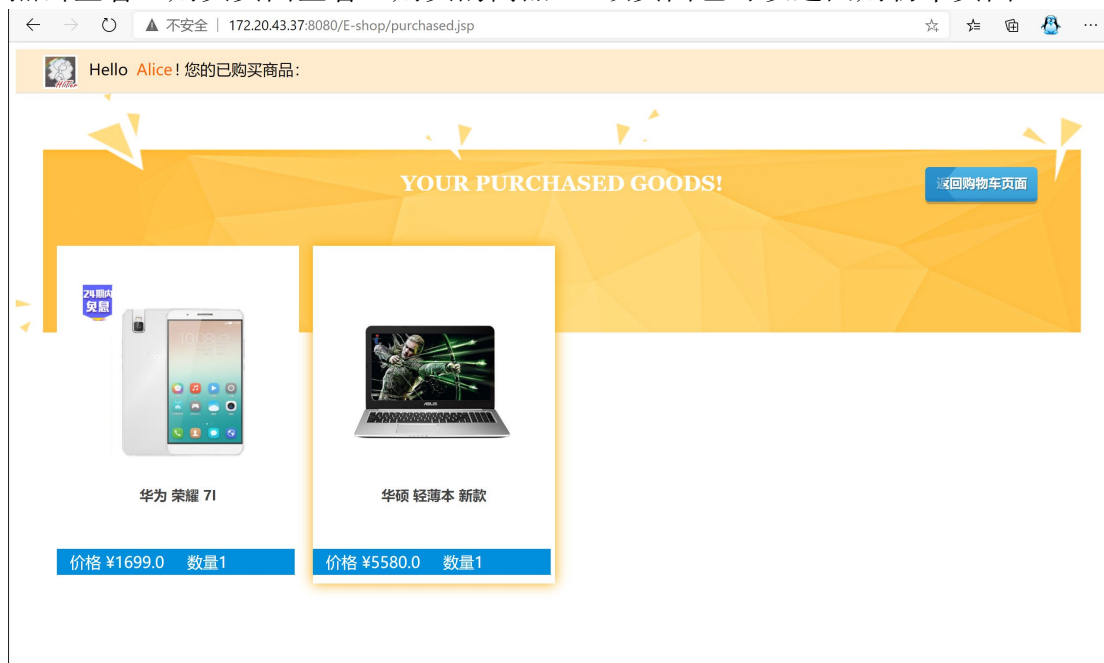
支付成功:



购物车已清空:



点击查看已购买页面查看已购买的商品: (该页面也可以返回购物车页面)



至此购物完成。至于过程中的前后端加密解密、交互过程的加密都在上一部分介绍，此部分就不演示了。

6. 结束语

本次密码学实验让我学到了许多，让我从 web 开发小白变成了已入门者，也通过对密码学原理的实践加深了对密码学的理解。

1. 我学会了基础的 `jsp-servlet` 框架进行前后端及数据库交互。
2. 还学会了一些前端开发的知识，如部分 `html`、`css`、`js` 的知识。
3. 也了解了电子商务平台面临的一些安全问题以及在 `HTTP1.0` 下如何利用密码学知识最大限度的提高平台的安全性。
4. 还知道了要注意交互需要加密和解密的方法相同，即使都用 `AES` 与 `RSA` 加密，具体的加密方式还是可能不同，需要注意具体的算法实现。还有编码格式最好用 `URLsafe` 的 `base64` 编码，因为用 `GET` 方法传参会将参数跟在 `url` 后面，而 `url` 中不能出现 `base64` 中的几个字符，所以需要 `URLsafe` 编码。

参考文献

《电商网站安全问题》:

<https://blog.csdn.net/markkr133/article/details/80545889>

《基于 java 的电子商务平台的设计与开发》:

<https://www.docin.com/p-2179257035.html>

《2020 年全球电子商务市场规模及未来发展趋势分析》:

<https://www.chyxx.com/industry/202010/902336.html>

《国内外电子商务的现状与发展》:

<http://www.npc.gov.cn/npc/c541/201706/408c9b1af8b14630a7f822e1d5c4bb74.shtml>

《电子商务支付存在的安全问题探析论文》:

<http://www.zhuna.cn/zhishi/404683.html>