****

**实**

**验**

**报**

**告**

**课程名称： J2EE程序设计**

**项目名称： 网上购物系统**

**学 号： 0174102**

**姓 名： 涂海霞**

**时间：2019年6月**

# 1 系统介绍

随着互联网时代信息技术的高速发展和4G,4G+乃至5G技术的迅速普及，企业的各式网站在人们的衣食住行中发挥着越来越大的作用。在如今，一个功能强大、界面美观的电子商务网站，对用户的吸引力与作用是足够大的，而且也能为企业带来足够的经济效益。

本购物网站系统模拟实现了用户注册、用户登录、个人信息修改、商品浏览、删除订单、支付订单、购买历史浏览等多项功能，满足了基本的购物流程需求。

## 功能：

1.用户注册

2.用户登录

3.查看用户本人详细信息，包括用户名、邮箱地址、上次登录时间等

4.用户信息修改，包括邮箱地址与密码等

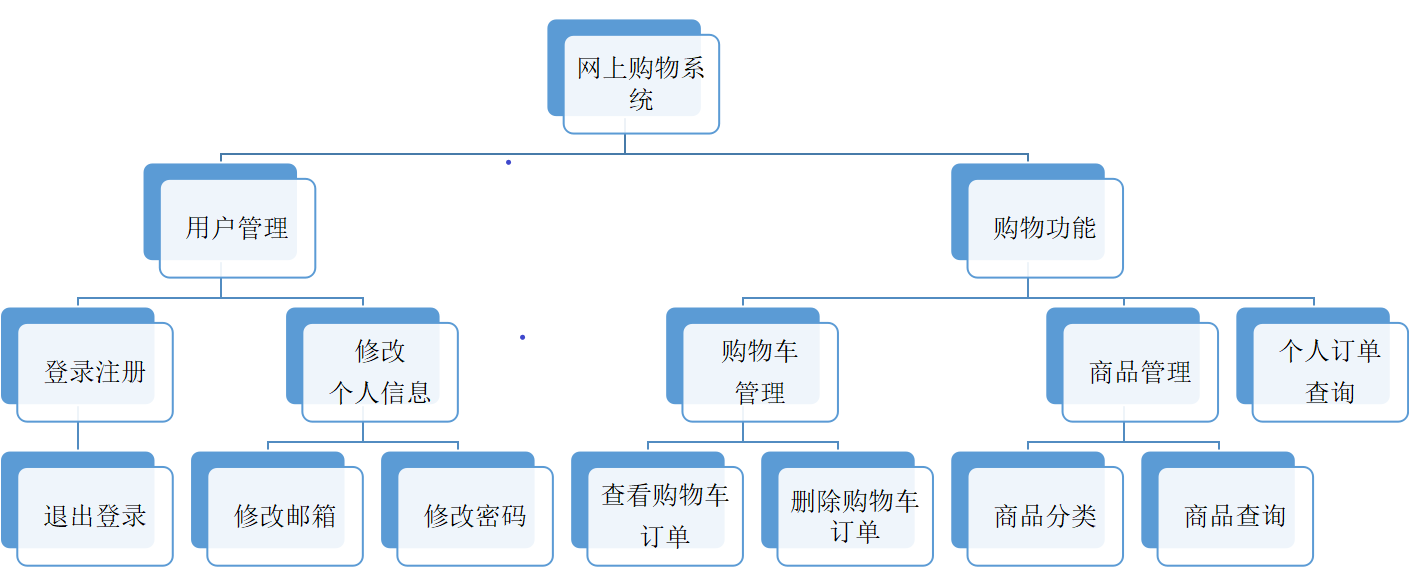
5.浏览商品详细信息

6.将商品加入购物车

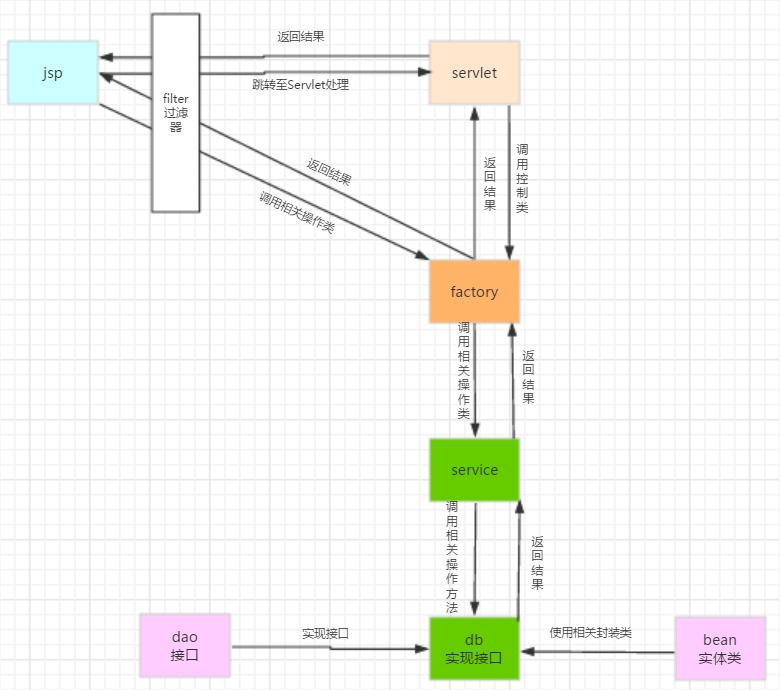
7.删除购物车中的订单

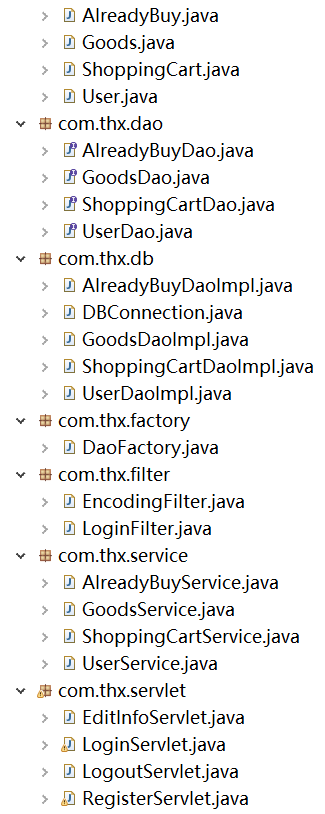
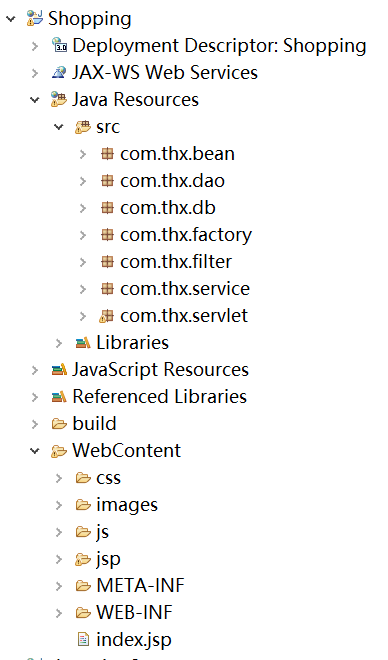
8.提交订单并支付

9.查看购买记录



# 2 业务逻辑设计





# 3 功能设计

1. 用户注册：
2. 用户通过register.jsp页面填写用户名、密码等信息进行注册，提交表单跳转到registerServlet进行注册操作；
3. registerServlet中定义了register函数对注册信息进行封装，并调用DaoFactory.getUserServiceInstance().addUser(user)方法；
4. 以上方法是DaoFactory控制类调用getUserServiceInstance()方法获取用户操作类UserService，再通过UserService调用addUser(user)方法将用户注册信息加入数据库中；
5. addUser(user)方法对注册信息中的用户名和邮箱进行检验，若数据库中已含有相同的用户名或邮箱则注册失败，重新返回注册页面；若无重复的用户名和密码则调用com.thx.db包中UserDaoImpl类中的addUser(user)方法对数据库进行用户增加操作。
6. 用户登录：
7. 用户通过login.jsp页面填写用户名及密码，提交表单至LoginServlet进行登录操作；
8. 在LoginServlet中通过DaoFactory.getUserServiceInstance()获取用户操作类UserService；
9. (1)判断登录信息是否已登录：通过UserService类调用 queryByName(uname)方法来获取user对象，若user不为null则已注册，反之 未注册，登录失败；

(2)判断密码是否正确：通上面获取的user对象中的getPasswd()方法来获取数据库中的密码并同登录信息中的密码进行比对，若密码一致则创建session并更新上一次登录的时间后跳转至首页index.jsp，反之密码错误，登录失败，跳转至login.jsp。

1. 用户个人信息浏览：

用户注册后方可充index.jsp页面进入个人信息页面showMessage.jsp浏览本人信息。

1. 用户个人信息修改：
2. 用户进入showMessage.jsp中点击编辑个人信息进入EditInfoServlet中通过DaoFactory.getUserServiceInstance()获取用户操作类UserService；
3. UserService类的userDao对象调用editEmail和editPasswd方法对邮箱和密码进行相关更改；
4. editEmail方法中先调用queryByEmail方法获取user对象，判断user是否为空并且该user的id是否为当前用户的id，若user不为空并且id不一致，即已存在其他用户含相同的email，不更新，反之则通过当前用户的id更新用户的email；
5. editPasswd直接通过当前用户id更新用户密码即可。
6. 商品信息浏览：
7. 用户可在首页index.jsp浏览所有统一样式的商品，直接在首页通过DaoFactory.getGoodsServiceInstance()获取商品操作类GoodsServices，再调用getAllGoods来获取数据库中所有的商品，通过for循环展示商品的信息；
8. 用户可点击相应商品链接进入goodsDescribed.jsp页面，浏览当前商品的详细信息，商品信息是通过商品操作类GoodsServices调用queryById方法通过request获取的gid来获取相应的商品信息。
9. 商品加入购物车：
10. 用户可点击首页index.jsp或商品详情页goodsDescribed.jsp中的“加入购物车”，跳转至addToCart.jsp页面；
11. 在addToCart.jsp中通过request.getgetParameter获取当前商品的数量number和id；
12. 通过DaoFactory.getShoppingCartServiceInstance()获取购物车操作类ShoppingCartService，进而调用addGoods()方法将商品加入购物车数据库中，商品加入成功跳转至购物车页面ShoppingCart.jsp，反之则提示加入购物车失败；
13. 在addGoods()方法中首先通过用户id和商品id调用getDesignateGoodsMs来判断购物车中是否已含有该商品，若已含有该商品则更新该商品的数量，反之则直接插入该商品的信息；
14. 删除购物车中订单：
15. 在购物车页面shoppingCart.jsp中，用户可点击“删除”,跳转至deleteGoods.jsp;
16. 通过DaoFactory.getShoppingCartServiceInstance()，获取购物车操作类ShoppingCartService，调用deleteGoods()方法，对购物车中的商品进行删除操作；
17. 在deleteGoods()方法中，仍应先调用getDesignateGoodsMs判断购物车中是否含有该商品，其次应对数据库购物车中该商品的数量与当前需删除的商品的数量进行比较，若购物车中商品已有的数量等于当前需删除的数量，则直接delete删除购物车中的商品，若已有的数量大于当前需删除的数量，则更新数据库购物车中当前商品的数量，若小于则提示删除订单失败；
18. 提交订单并支付：
19. 进入购物车页面shoppingCart.jsp中，点击支付即可支付购物车中所有的商品，但此处并未实现真正的支付，只是删除购物车中的商品，并且将当前删除的商品加入到已购买的数据库中，并显示到alreadyBuy.jsp上；
20. 用户点击支付，跳转至buyGoods.jsp,通过DaoFactory.getShoppingCartServiceInstance()，获取购物车操作类ShoppingCartService，调用payAllGoods()方法获取所有购物车中的商品，并循环调用payGoods()方法逐一处理；
21. 在payGoods()方法中需先判断所支付的商品的数量是否小于等于该商品的库存，通过商品id获取该商品库存通需支付的数量进行比较，若库存大于等于所需支付的数量，则在商品表中对该商品进行删除操作，同时若已购买的订单中已含该商品则直接更新数量，若无该商品则增加该商品的记录，若库存小于所需支付的数量，则提示删除失败；
22. 查看购买记录：

已登录的用户可直接在alreadyGoods.jsp中浏览已购买的商品，通过DaoFactory.getAlreadyBuyServiceInstance()，获取AlreadyBuyDao已购买订单的操作类，调用getAllBuyGoods()来获取所有已购买的商品。

1. 过滤器：
2. 登录过滤器：如果未登录则无法进入相应页面（alreadyBuy.jsp、shoppingCart.jsp、addToCart.jsp、showMessage.jsp）；
3. 字符编码过滤器：将请求响应的编码一律初始化为utf-8；

# 4 系统展示

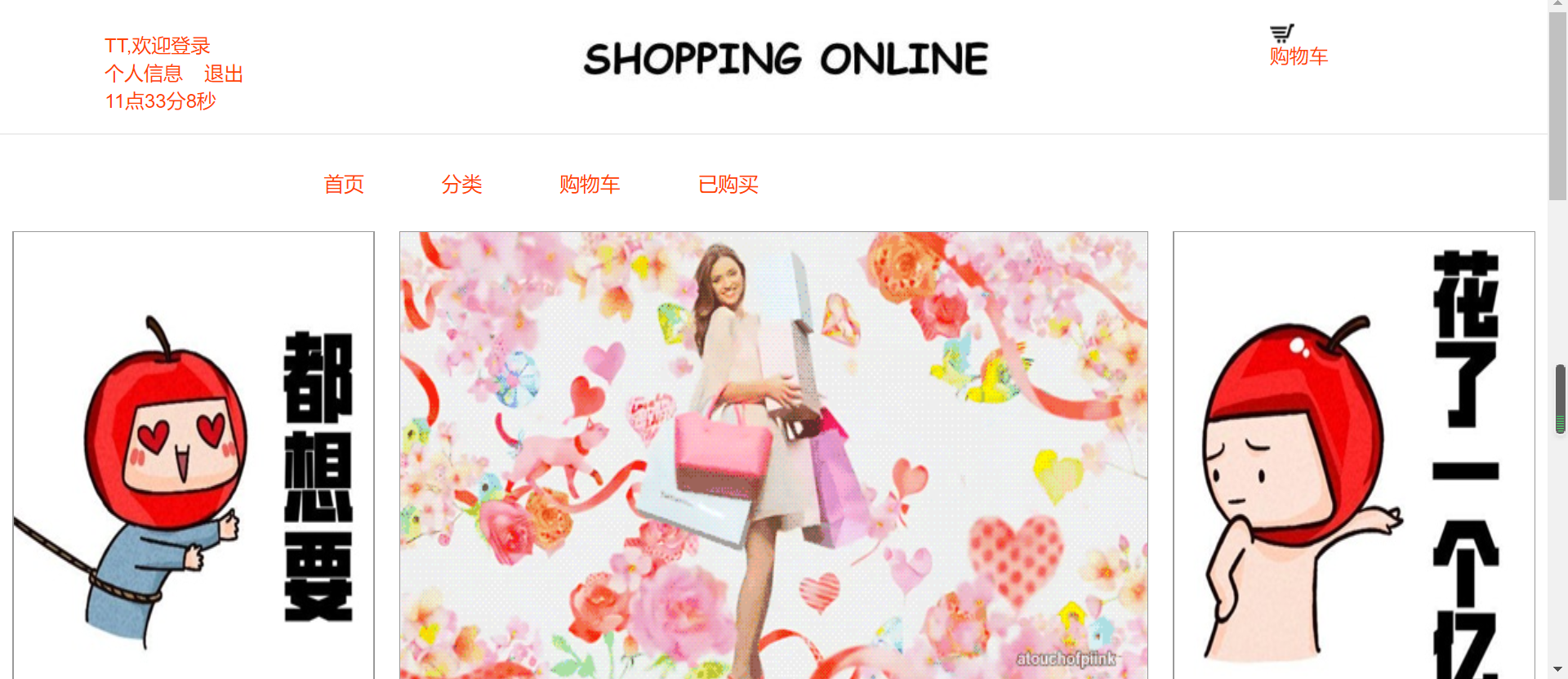
1. 注册页面



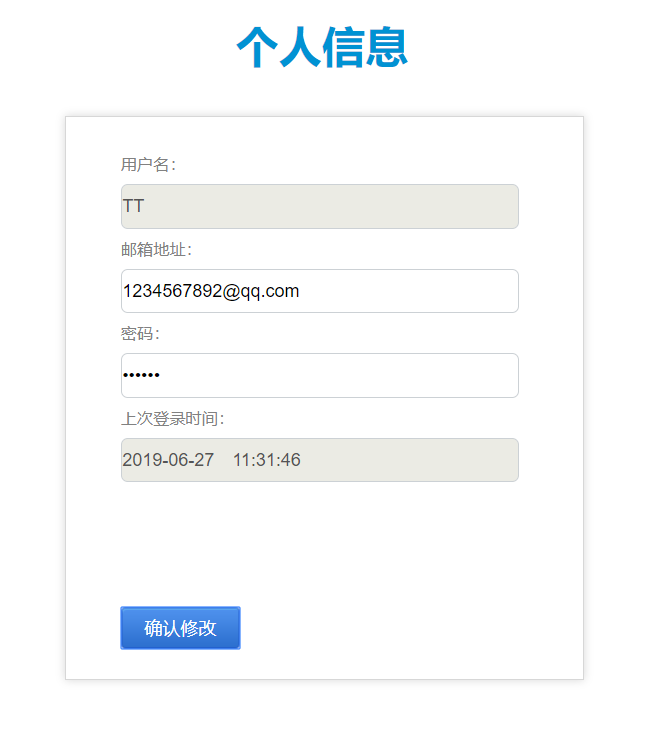
1. 登录页面：



1. 首页：



1. 个人信息：



1. 商品分类：



1. 购物车：



1. 删除订单：



8、支付订单：



1. 商品详情：



9、已购买订单：



1. 数据库设计：

