编号_____

南京航空航天大学

毕业设计

题　目　校园易物系统的分析与设计

学生姓名　尼玛扎西

学　　号　091300422

学　　院　经济与管理学院

专　　业　信息管理与信息系统

班　　级　0913201

指导教师　胡正华　副教授

二〇一七年六月

南京航空航天大学
本科毕业设计诚信承诺书

本人郑重声明：所呈交的毕业设计（题目：校园易物系统的分析与设计）是本人在导师的指导下独立进行研究所取得的成果。尽本人所知，除了毕业设计中特别加以标注引用的内容外，本毕业设计不包含任何其他个人或集体已经发表或撰写的成果作品。

作者签名：　　年　月　日

（学号）：

校园易物系统的分析与设计

摘 要

校园易物系统是以以物易物为原始构想展开的，在个别大学校园里已经存在类似的平台，该系统的作用就是使得物尽其用，还可以在一定程度上节省花销。

论文研究校园易物系统的分析与设计，论文首先研究校园易物系统的分析，其次研究校园易物系统的设计，最后根据系统分析与设计的成果对系统进行实现,论文的成功对校园内闲置物品价值的发挥有提升的意义。系统的分析与设计使用的是 UML 面向对象的的分析与设计方法。系统分析阶段分别使用用例图确定需求、活动图确定业务流程、时序图确定使用方法及相关类。系统设计阶段分别通过类模型描述系统框架、数据库物理模型描述表间关系以及各表的字段、部署构件图描述项目实施部署环境及运行方式。系统使用的架构是 B/S 架构，客户端是浏览器，用户通过浏览器就可以访问服务。系统主要使用的开发语言是 java，框架选择的是 Springboot+mybatitis，前端框架选用的是 bootstrap。

关键词：面向对象模型，B/S 架构，浏览器，时序图

Analysis and Design of University Campus Barter System

Abstract

The college swop system is based on the original concept of the scheme that barter, and there is already a similar platform on the individual university campus. The role of the system is to make the best use of goods, and it can save the cost to a certain extent.

This paper studies the design and design of campus easy system; firstly, the paper analyzes the campus barter system; second, the paper studies the design of the campus barter system; finally, implement the system based on the results of System Analysis and Design; the success of the paper play a improvement role on the value of idle items. System analysis and design using UML object-oriented's analysis and design methods. System analysis phase use the use case diagram to determine the needs ,use the activities to determine the business process, use the sequence diagram to determine the use of methods and related classes. The system design phase describes the system framework through the class diagram, use the database physical model describes the relationship between the table, the deployment diagram of components to describe the project implementation of the deployment environment and mode of operation. The architecture used by the system is the B / S architecture, the client is the browser, the user can access the service through the browser. System is mainly used in the development language is java, the framework is selected Springboot + Mybatitis, front-end framework is selected bootstrap.

Key Words：Object-oriented model; B/S architecture; Browser; Sequence diagram

目 录

图目录

表目录

第一章 引 言

1.1 系统开发背景

首先是一个现实生活中关于易物的真实案例：[1]在德国，有一位叫做海德马里·施韦尔默的69岁的妇女，她本想着如何“免费”活一年，但是他却做到了“免费”活15年，15年间不使用任何货币交易。在那一年，也就是1996年，她辞了工作 ，停了医疗保险，关闭了银行账户，搬出了租来的住房，舍弃了财产，在多特蒙德市郊区开了一家以物易物的小店，过起“免费”的生活，患病以后也是以交换的方式换取治疗。她除了向当地人提供物品交换平台之外和还提供技能交换的平台，她是提供者也是受益人。例如，修车服务可以换修管道工服务，旧衣服可以换厨房用具。这位老妇人通过这种方式不仅养活了自己，还帮助一些失业和退休人员，让他们的旧物件和技术有了用武之地。通过这个简短的事例，可以从中看出易物的价值是不可估量的，不过就当前条件而言我只能以校园环境为主要”易物“实施场所。以下就以学生为易物平台的主要受众展开讨论。

[2]以物易物，随着现代信息技术的快速发展，国内外的以物易物平台越来越多，其中有一个普遍现象，其主要用户是学生、白领和准白领，年龄主要分布在 16～40 岁之间。这种新型的电子商务交易模式就是基于 Web2.0 模式建立一个物品交换的虚拟空间，换客们可以在这个平台上进行免费搜索、交流以及交换。本系统设计主要针对在校大学生群体，能够实现旧物发布、旧物展示、物品交换等的功能，是一个功能全面、操作简单的以物易物平台，可以方便大学生将闲置资源充分利用，并一定程度加强在校大学生之间的交流。

在对在校大学生进行的调查问卷中， 在以校园以主要区域考察了建立易物系统的需求，可行性以及必要性，在问卷显示结果看来在大学校园急需建立这样一个易物平台。问卷显示：大学老师和学生的课本以及杂志之类的书籍属量最大的闲置品；据问卷显示目前在校的大学生有或多或少的闲置的物品，然而使用易物相关平台人的却很少；大多数的学生使用易物类的网站是为了省钱以及减少自己的闲置物品。在校的大学生就我自身而言就有很多闲置物品，经常因为不知道怎么处理而苦恼，如果校园内出现一个能帮助他们增进相互之间交流的以物易物平台， 将会使得在校大学生的闲置物品在一定程度上物尽其用，并且享受这种原始易物的乐趣。

1.2 系统开发目的及意义

在校内，每个学生都有一定的搁置物品。这些物品最终的结果，或丢弃、或赠人，没办法使得物品发挥其应有的价值。该系统的目的就是在本校园内，起到深层次挖掘物品价值的作用，尽量使每个物品的价值得到尽可能大的发挥。在另一方面，也能为校内学生降低一定的花销；也响应了以环保为主的生活方式。

第二章　系统分析

本章内容是以本系统的分析为主的，分析主要包括三个方面，需求分析、业务分析、软件分析。这里的分析是以业务逻辑为主导的分析。其中需求分析是以用例图来分析表示的、业务分析图是以活动图分析表示的、软件分析是以时序图分析表示的。

系统的分析是使用的一款专门的软件来完成的，下一章系统的设计也是使用这款软件完成的。这款软件叫 Powerdesigner，版本号是 16。PowerDesigner 最初由 Xiao-Yun Wang（王晓昀）在 SDP Technologies 公司开发完成。PowerDesigner 是一个企业（Sybase）的建模和设计解决方案，采用模型驱动的方法是对业务进行整合，它可以帮助企业有效架构的发展，发展的生命周期管理它提供了一个功能强大的分析和设计技术。其结果是，标准的数据建模技术（UML，数据建模引领业务流程建模和市场）的全集成，并集成了.NET，工作区，PowerBuilder 中，的 Java™，Eclipse 和其他主流并提供业务分析和开发平台的各种 PowerDesigner 中的创意传统软件开发的生命周期管理数据库的设计方案规范。它还支持 60 种关系数据库管理系统（RDBMS）/版本。

2.1　需求分析

以下主要以系统的大体功能为为主，以用户为主导方向来分析该需求。

1. 注册

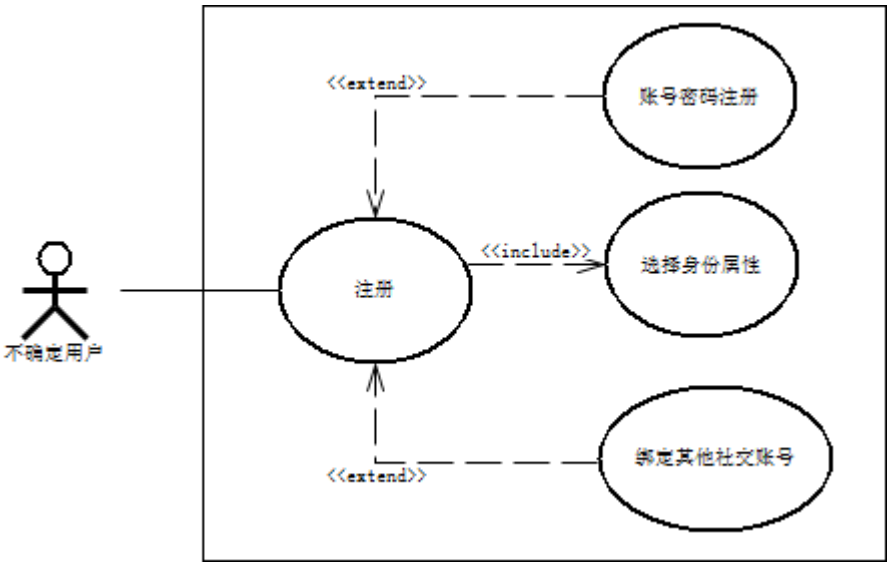

图 2. 1 用户注册用例

图上是关于用户注册的用例模型。系统设定，为了统一确定每一个人在系统中是唯一并且真实存在的。所以每一个使用该系统的人都需要经过注册才能进入系统。在注册以前暂且认为是不确定用户 X。注册时系统会提供两种注册方式：账号密码、绑定其他社交账号；其中这两者注册都必须包含一个功能——确定身份属性，是已易物为目的而来还是以公益筹资而来。由于时间有限，在系统实现时并不能保证完全按照设计的模样做出来，实际会有一些取舍，比如就注册功能而言，在刚入手时我只会实现通过账号密码来实现注册；如果，后期时间充裕我会选择性完成之前舍去的功能。

2. 登录

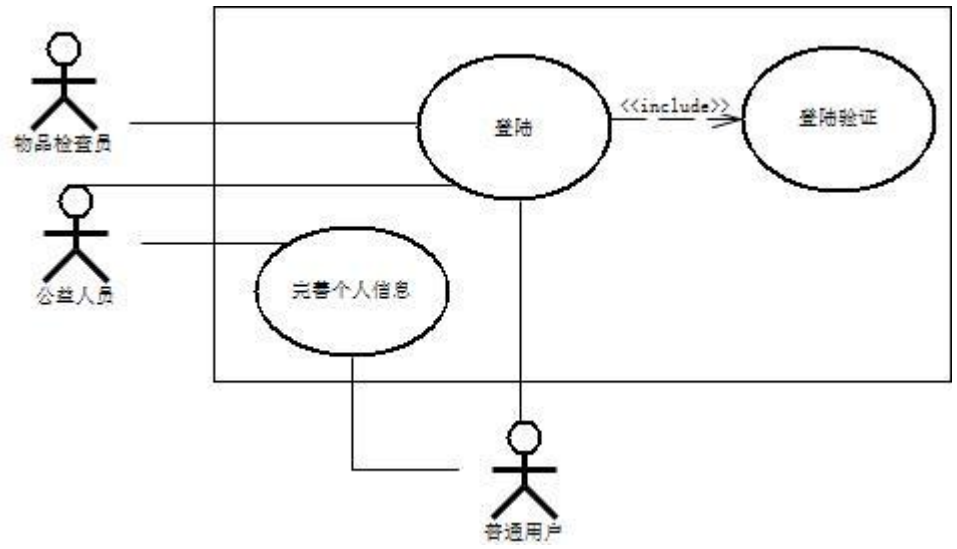

图 2. 2 不同用户的登录

上图描述的不同用户登陆的用例模型。在系统的真实用户中只有三种用户；一、是以易物为主要目的的普通用户。二、是以公益筹资为主要目的的公益用户。三、是以管理系统的管理员用户。这三种用户如果要使用系统功能，他们都必须登录。登陆之后，普通用户和公益用户可以继续完善自己的个人信息。公益用户必须进行个人信息完善并通过管理审核才会有筹集公益物资的权限。管理员用户就没有这个功能，因为没有任何必要性。

3. 个人物品管理

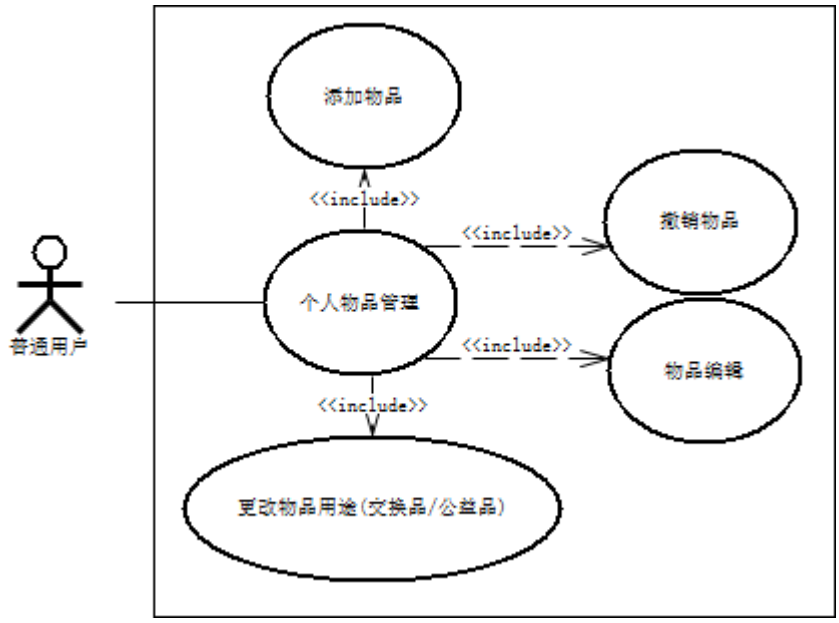

图 2. 3 个人物品管理

图上是关于普通用户登录后所有的功能之一——个人物品管理，也是本系统最主要的功能之一。当普通用户登陆后会有一个个人物品管理的模块，该模块具体包含四个功能：新增一个物品；撤销（删除）物品；物品编辑，修改物品信息；更改物品用途，更改可选项有交换品/公益品。以下会一一展开分析。

3.1 个人物品管理-添加物品

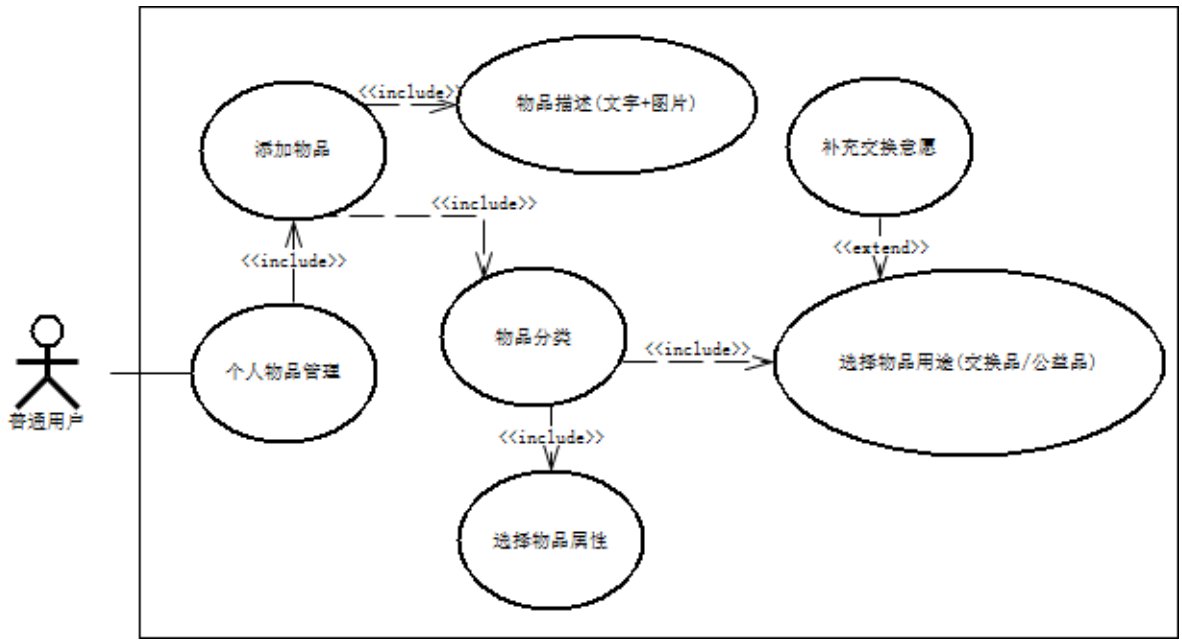

图 2. 4 添加物品

图上描述的是关于普通用户登录后选择了物品管理模块后的其中一个功能模块——新增

一件物品。如果要新增一件物品的，首先应该包含物品描述+图片，然后还需要为物品选一个分类，而这个分类可以当做一个属性来执行；到这里就算是有了一个物品，不过还差一件事，就是决定这个物品到底是用来交换的还是直接捐赠为公益品？如果是用来交换的，这就又拓展出一个新功能——补充个人交换意愿，这样就有可能得到自己想要的物品，这是一个拓展功能，可有可无，到底执行与否还是看个人。

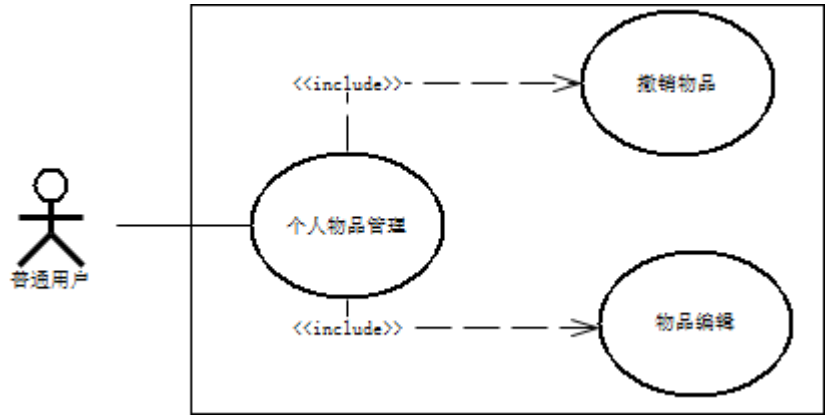

图 2. 5 删除/更新物品

3.2 个人物品管理-撤销物品

删除一件物品不会被拆分成几个功能,也不能拓展出其他用例—功能。该功能的详细过程会在业务分析中详细展开。

3.3 个人物品管理-物品编辑

编辑一件物品这一功能也不会被拆分成几个功能,也不能拓展出其他用例（功能）。该功能的详细过程也将会在业务分析中详细展开。

3.2 个人物品管理-更改物品用途（易物/公益）

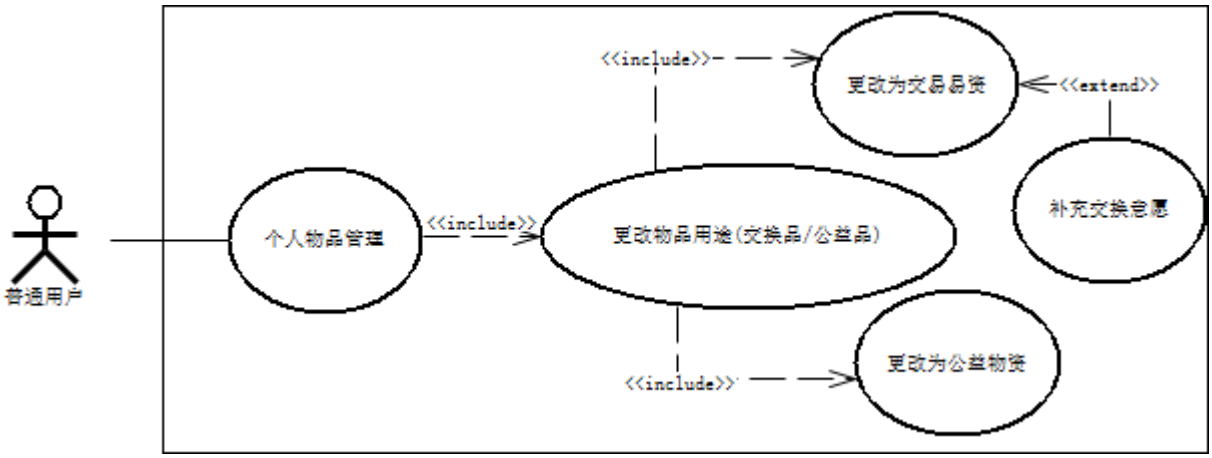

图 2. 6 更改物品用途

图上描述的是关于普通用户登录后选择了物品管理模块下更改物品用途的功能的用例模

型。更改物品用途用例包含了更改为交易易资、更改为公益物资两个用例。就本系统而言，物资流转的方式也就只有这两种。其中更改为交易物资这一用例又拓展出一个用例——补充交换意愿，物品从公益物资转换为交易易资，就多了一个属性——交换意愿；这个拓展功能也是可选项，可执行可不执行；这个功能存在的意义：可能使得用户交换到自己想要的物品，从而提高用户交换物品时的满意度。

4. 交易管理

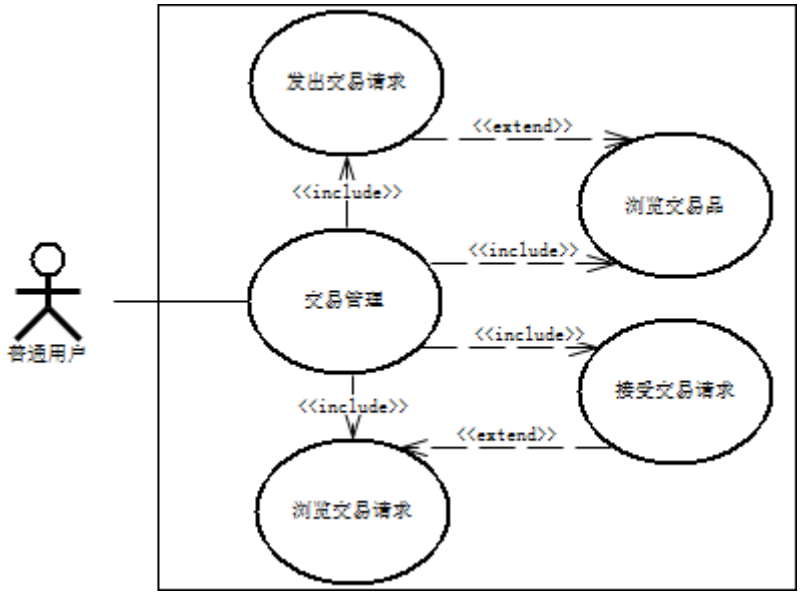

图 2. 7 交易管理

图上描述的是普通用户登陆后会用到的主要模块之一——交易管理的用例模型。交易管理用例包含了四个用例：浏览交易品、发出交易请求、浏览交易请求、接受交易请求；这四个用例是实现交易的最主要的用例。其中发出交易请求用例既是必须存在的用例，又是浏览交易品用例的拓展用例；同样，接受交易请求用例既是包含于交易管理用例的，又是浏览交易请求的拓展用例。所以，发出交易请求与接受交易请求用例于系统而言是必要的组成部分，于用户而言是可执行可不执行的。

4.1 接受交易请求

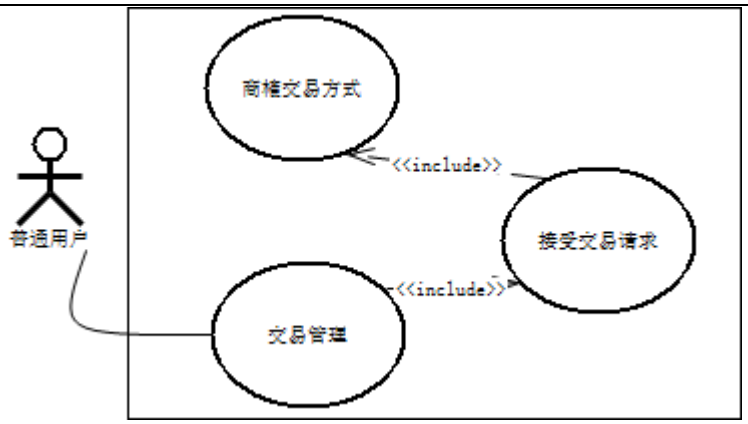

图 2. 8 接受交易请求

图上描述的是关于用户登录后，交易模块下的接受交易请求的用例模型。接受交易请求的用例包含了一个用例——商榷交易方式用例。商榷交易方式是接受交易请求之后所需要进行的步骤，也是整个交易的倒数第二步，其详细过程将在下一节业务分析中展开分析。

4.2 发出交易请求

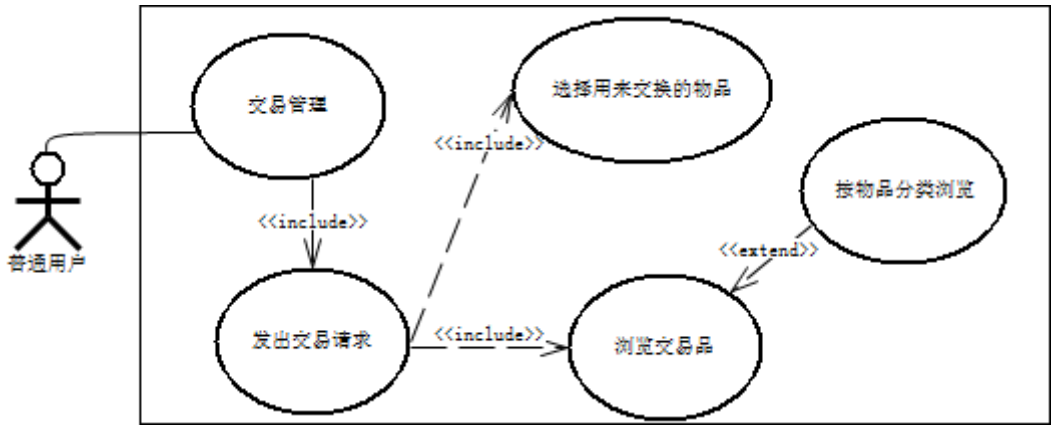

图 2. 9 发出交易请求

图上描述的是关于用户登陆后，交易模块下的发出交易请求的用例模型。发出交易请求用例分别包含了浏览交易品用例和选择交易品用例；浏览交易品用例又扩展出按物品分类浏览用例；以上就是发出交易请求所需要的所有用例，以及用例之间的关系。大致过程就是用户通过浏览交易品（可分类浏览）选定要交易的物品，然后选出自己的物品做筹码来交换目标物品；详细业务过程将在下一节业务分析中展开分析。

5. 公益资源管理

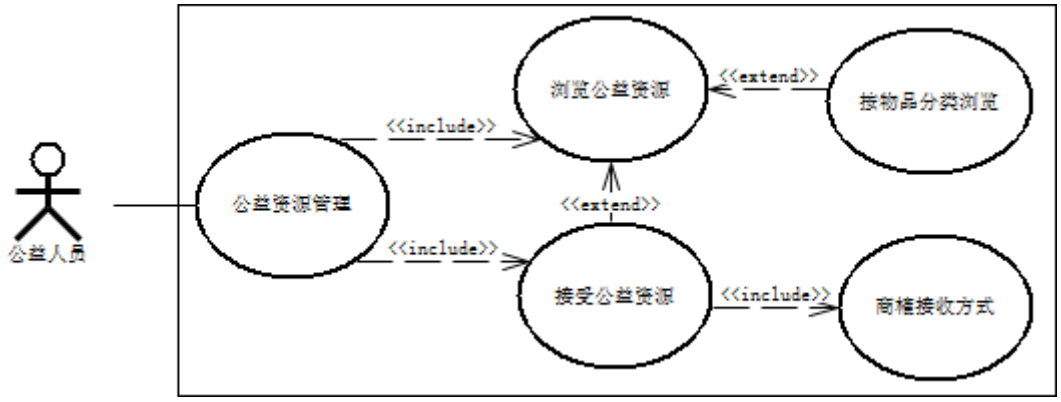

图 2. 10 公益资源管理

图上描述的是关于公益用户登陆后，关于公益资源管理的用例模型。公益资源管理用例分别包含了浏览公益资源用例和接受公益资源用例；浏览公益资源用例扩展出了按物品分类浏览用例（用浏览交易品）；其中浏览公益资源用例同是接受公益资源用例的额扩展用例；接受公益资源用例还包含了商榷交易方式用例。其大致过程就是公益用户登录后根据物品分类浏览公益资源，从而选择接受与否，如果接收的话则进入商榷接收方式阶段；详细业务过程将在下一节业务分析中展开分析。

6．后台管理

物品管理将分为两个图例说明：物品管理、用户管理。

6.1 物品管理

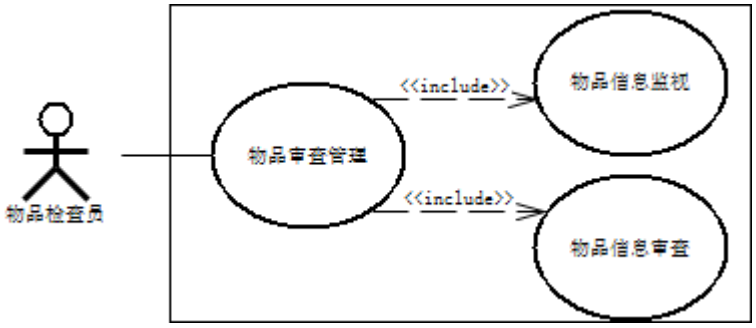

图 2. 11 物品管理

图上描述的是关于后台管理审查物品的用例。物品审查管理用例分别包含了物品信息监视用例和物品信息审查用例；物品信息审查用例作用在物品首次添加上传到公共主页时（包含了交易物品和公益物品）；物品信息监视用例作用在所有物品的“一生”，在物品未下架之前，有关于物品信息的所有的变动都会被后台监视，以防止出现违背规定的交易；详细业务过程将在下一节业务分析中展开分析。

6.2 用户管理

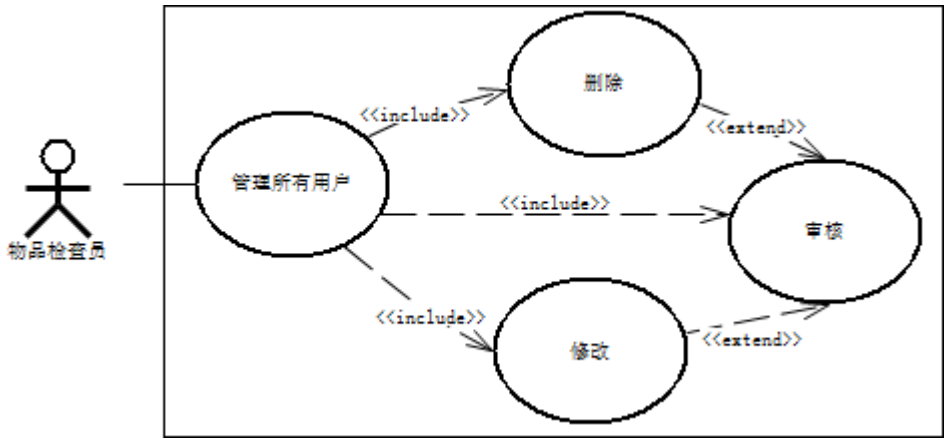

图 2. 12 用户管理

图上描述的是关于后台管理管理用户的用例模型。管理所有用户用例分别包含了三个用例：删除、修改、审核。其中审核用例还是删除用例于修改用例的扩展用例，即后台管理在对一个用户进行信息变更甚至删除时应当再审核一下该用户的信息；关于审核用例的作用阶段：在用户注册为校园学生、公益用户时，需要后台管理审核；详细业务过程将在下一节业务分析中展开分析。

2.2 业务分析

业务分析是展开的需求分析，是将需求分析中的用例具体化。本节由活动图来完成。

1. 注册

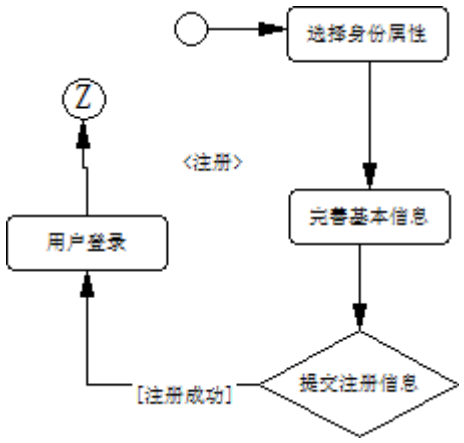

图 2. 13 注册

图上描述的是关于用户注册的活动图。不确定用户首先选择自己的身份属性，确定自己是公益用户还是普通用户。之后便是完善基本的信息，例如学号，姓名，性别。然后就是提交注册信息，在注册成功之后系统追自动向导至登陆界面。

2. 登录

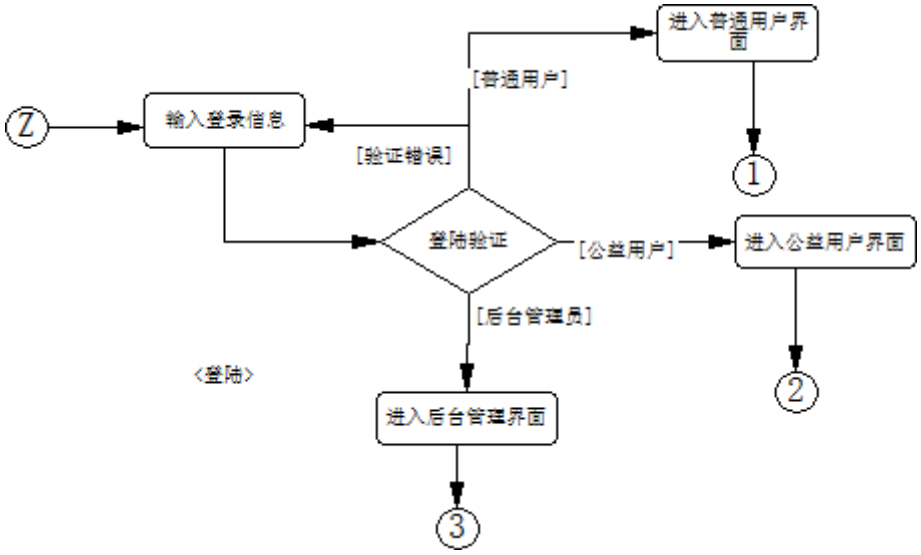

图 2. 14 登录

图上描述的是关于三种身份的用户登录的活动图。用户首先来到登陆页面，输入登录信息，系统会验证合法性，如果验证出错请重新登陆，反之验证成功，系统还需要做进一步的判断，判断登陆者的身份属性，之前提到过，系统的使用者分别是普通用户、公益用户、后台管理三种以后。在系统进一步判断之后，会自动导航到各自对应的页面。

3．普通用户所具有的所有功能

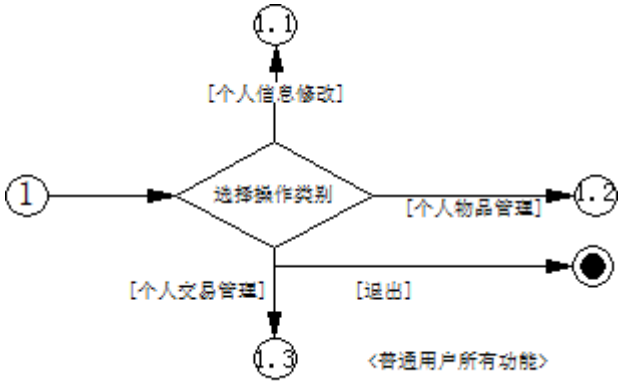

图 2. 15 普通用户功能概览

图上描述的是关于普通用户登陆后所能使用到的所有功能。由图可见一共三个功能入口：个人信息修改、个人物品管理、个人交易管理；以上是最主要的三个功能，还有常规需要设定的功能——退出系统。下面将一一详细分析各个功能。

3.1 个人信息修改

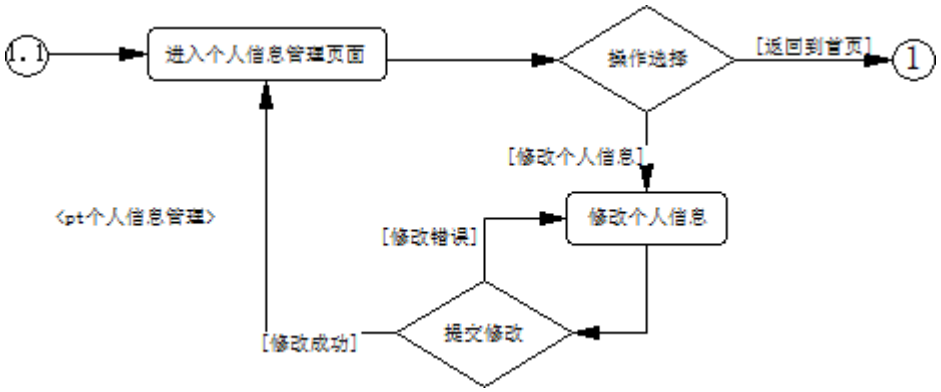

图 2. 16 普通用户个人信息修改

图上描述的是关于普通用户个人信息管理的活动图。用户进入个人信息管理页面之后有两个选择，退回到首页或者选择修改或完善个人信息。如果选择修改个人信息，会出现一个信息可编辑的界面，用户可以通过此界面完善或修改信息，信息修改完成后需要提交到后台验证（平台会定制一些约定来保证用户信息不会影响到平台的正确运行），后台会返回一个结果，如果信息没有问题则返回成功进入初始页面反之返回修改错误进入重新修改页面。

注：由于时间紧迫，有关公益模块的设计后续将暂停，主要以实现易物为主。

3.2.1 个人物品管理——添加物品

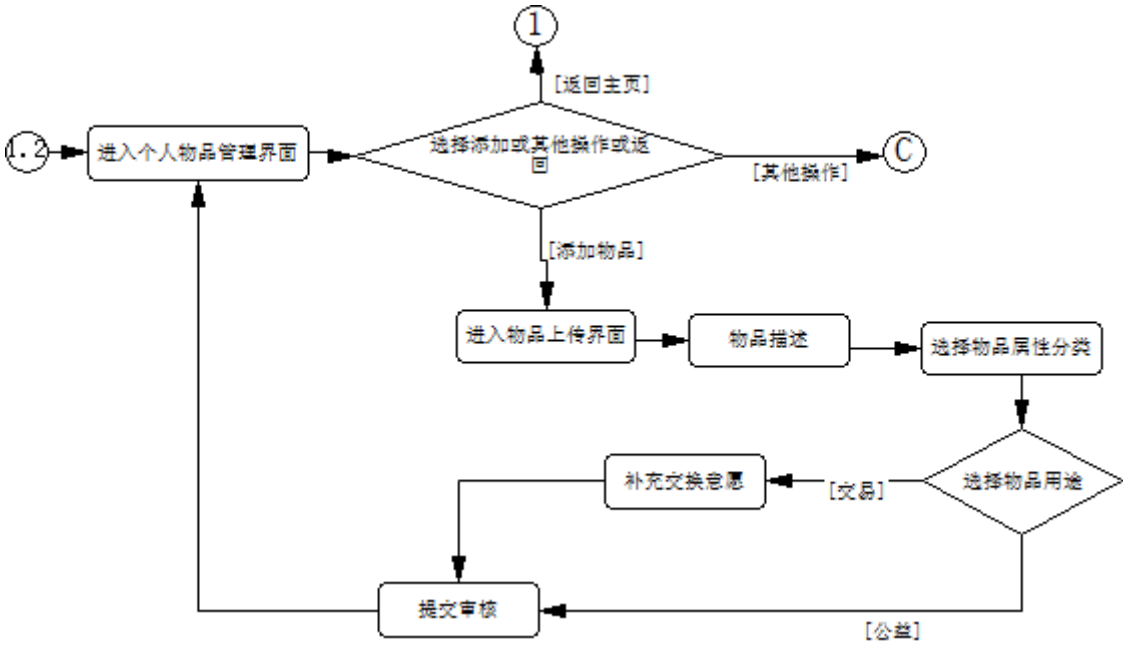

图 2. 17 物品管理—添加

图上主要描述的是关于普通用户添加一件物品的操作。首先用户进入个人物品管理界面后会有三个操作选项：返回主页、其他操作、添加物品；以下将详细描述添加物品的流程。用户选择添加物品选项后进入了物品上传界面，在物品上传界面会要求用户描述所要上传的

物品信息（实例一个物品），通过物品的名称、图片、描述等信息来描述。为了进一步方便用户的使用，还要用户确定物品的分类，例如：文具、电子产品等等之类的。下一步是选择物品的用途，此项是以交易与公益为选项的。如果是公益用品直接提交审核；如果是交易需要进一步补充交换意愿，补充后再提交审核。以上便是普通用户添加一件物品的流程。

3.2.2 个人物品管理——物品编辑

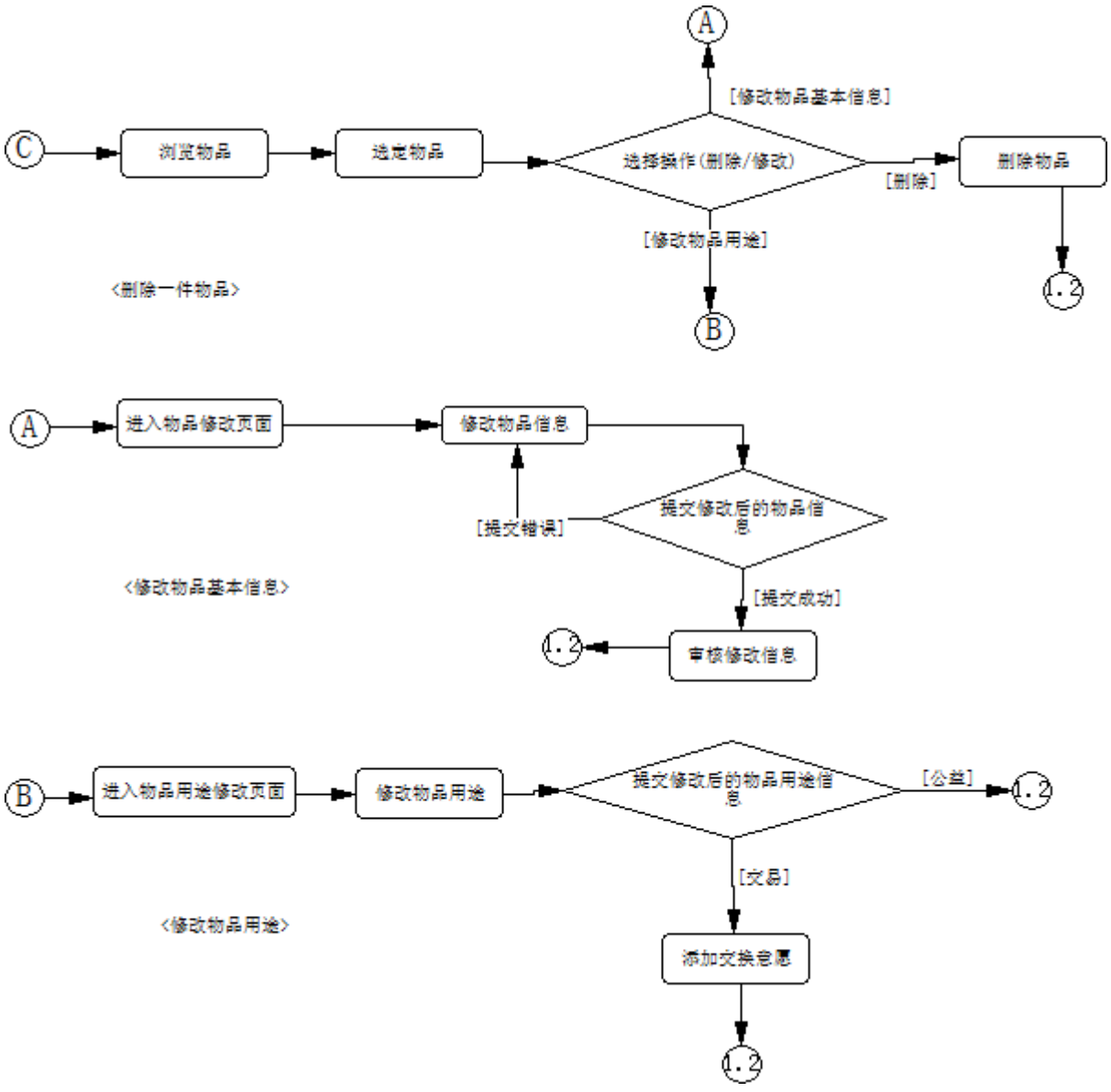

图 2. 18 物品管理—编辑

图上描述的是关于普通用户的物品编辑管理：A、B、C。A 代表修改物品信息；B 代表修改物品用途，没有跟 A 功能合在一块是因为 B 功能除了修改用途这一属性之外还需要判别用途是什么并作相应的调整；C 代表删除一件物品。以下详细描述各个模块的功能，修改物

品信息：首先用户通过选定特定物品进入详情页面，在点击修改，就会进入对应物品信息编辑页面，用户完成编辑后需要提交到后台，提交的过程中可能因网络差错或其他原因导致提交不成功，如果真的发生了，则需要重新提交，提交成功后再由后台判断修改的信息是否否和约定；修改物品用途：同修改物品信息，首先经由选定特定的物品再选择要执行的功能——修改物品用途，然后选择修改物品用途，可供修改的选项有公益与交易，如果选择修改为公益，确定修改后直接进入 1.2 步骤，如果选择修改为交易，则需要补充交换意愿再提交再返回至 1.2 步骤；删除一件物品：首先经由浏览物品再选定想要删除的物品然后执行删除操作，系统便会执行删除物品功能再返回至 1.2 步骤。

3.3.1 个人交易管理—主动交易

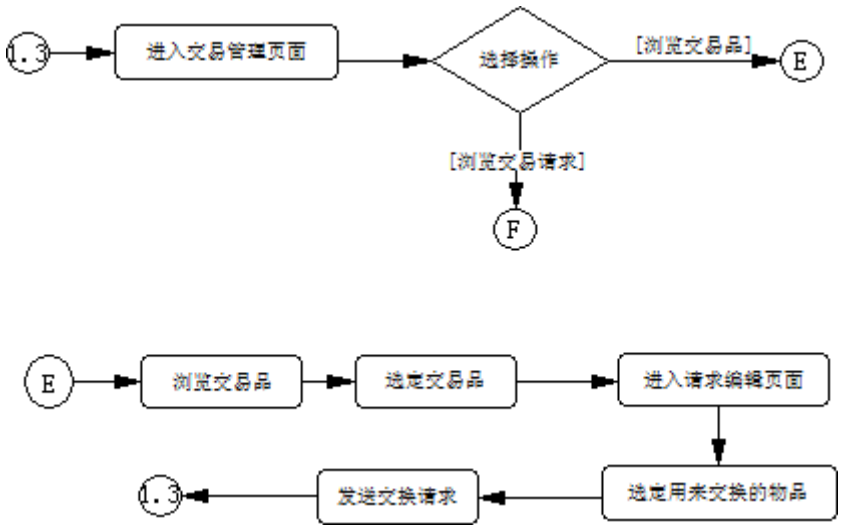

图 2. 19 主动交易

图上描述的是关于用户交易的活动图之一——主动交易，除此外对应的还有被动交易。当用户进入交易管理页面（或称交易广场）后，用户会有两个操作选择：浏览交易品进行交易或者浏览交易请求进行交易。这里浏览交易品进行的交易可以视作主动交易，通过浏览交易请求进行的交易视作被动交易，图上 E 开头的图对应的活动图描述的就是通过浏览交易品进行的主动交易。

主动交易过程：首先用户需要浏览交易品，然后选定用户中意的物品，然后执行交易操作，然后进入到交易请求编辑界面，到了交易请求编辑界面后主要的任务就是请求方选定自己的一样物品作为交换的物品，到这一步交易请求就编辑好了，然后由请求方用户执行发送交易请求。

3.3.2 个人交易管理—被动交易

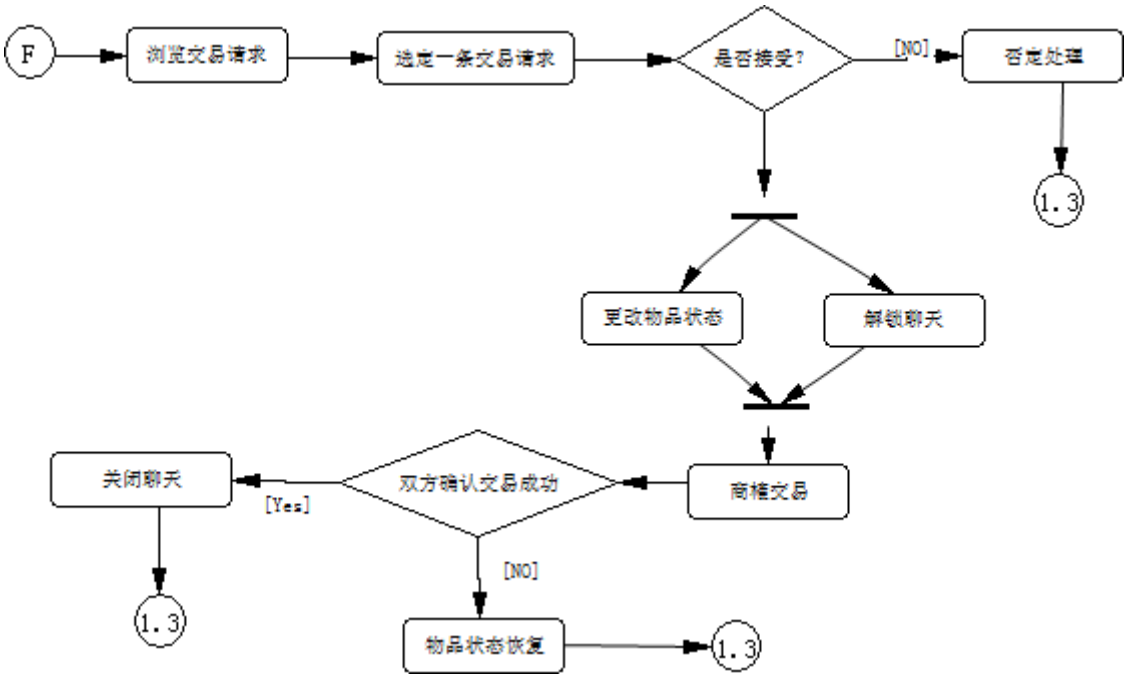

图 2. 20 被动交易

图上描述的是关于用户进行被动交易的活动图。

被动交易过程：首先接受方用户（当前用户）浏览收到的交易请求，然后根据请求的摘要选定一条请求查看性情，看到请求的详细信息后，接受方用户根据个人倾向选择接受或者不接受，如果选择不接受，就标记请求未通过，发送请求的一方也会看到请求的处理结果，当前这条交易到此结束。如果接受方用户选择了接受，系统会在双方之间开通一条聊天通道，同时会把当前正处在交易的物品的状态标记为交易中（此种状态下，物品无法再进行主动交易和被动交易），然后由发送方与请求方协商交易，最终在系统上需要双方确认交易成功，如果有一方未确认或者选择未成功，则交易结束并关闭聊天通道，原物品状态恢复；如果双方都确认交易成功则物品下架并关闭聊天通道，交易结束。

4.1 后台管理

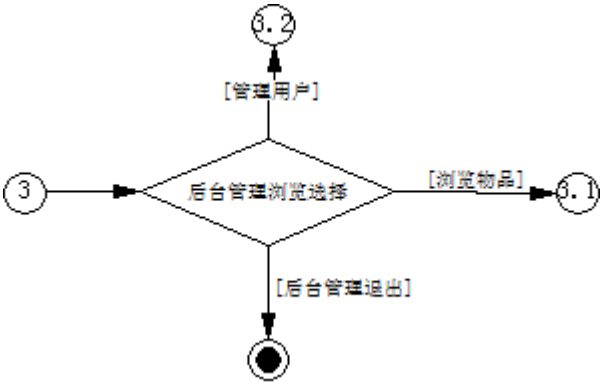

图 2. 21 后台管理总览

图上描述的是关于后台管理用户的功能总览。后台管理登入首页后总共有三个操作选择：管理用户、浏览物品、退出。

4.2 后台管理—物品管理

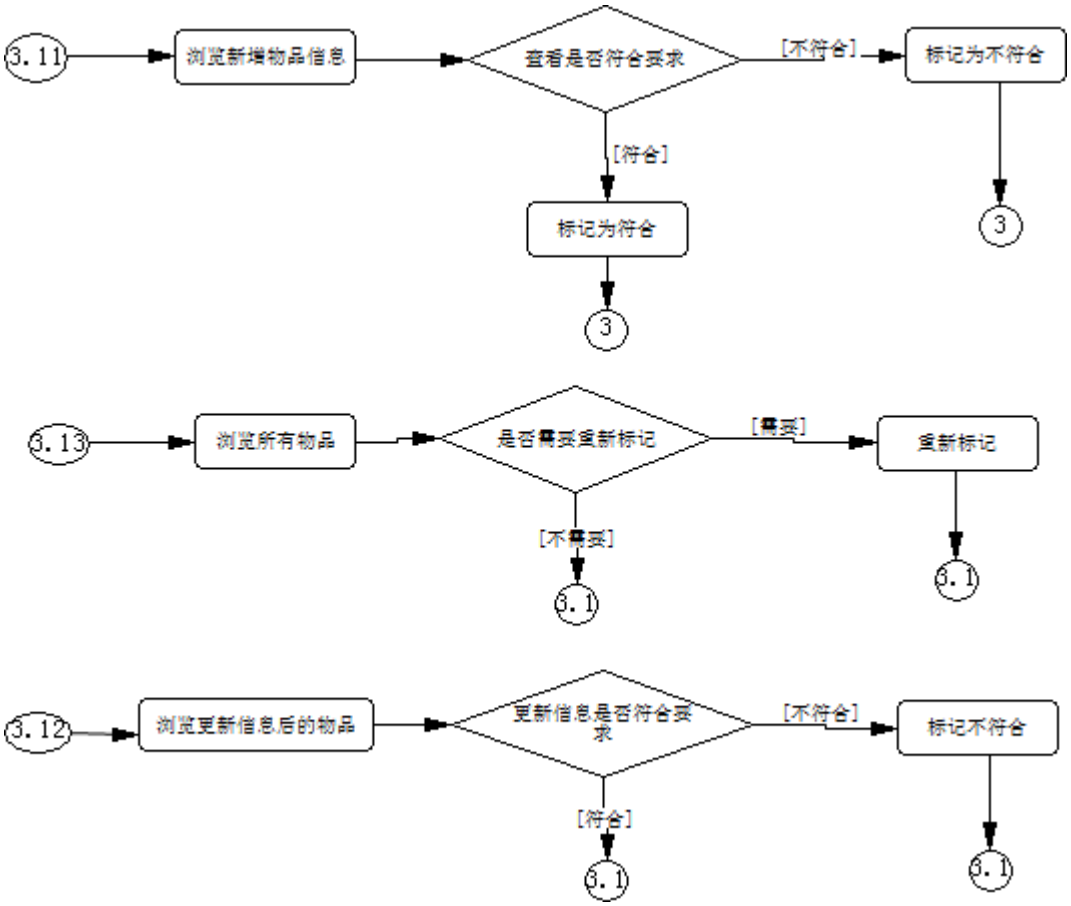

图 2. 22 后台—物品管理

图上描述的是后台管理对应的所有物品的操作。

审查新增物品：每当有新增物品时，都需要通过后台管理审查才能进入到交易广场（被其他用户看到），后台管理通过物品的描述信息（包括图片）决定物品是否符合约定，符合标

记为符合（改变物品状态——上架），不符合标记为不符合，该物品所有者会看到不符合，从而修改物品描述或是其他操作。

物品监听：即图上 3.12 开始的活动图部分。监听只是一个比喻，系统会察觉到所有物品信息的变动操作，如有变动，物品状态将会进入到审查阶段，即需要管理重新审核物品描述信息是否符合约定并做出相应的操作。

重新标记：即图上 3.13 开始的活动图部分。重新标记模块的存在意义在于以防管理有处理错误的时候，所以需要重新标记。其过程是根据物品描述信息来判断是否需要进一步操作。

4.3 后台管理—普通用户

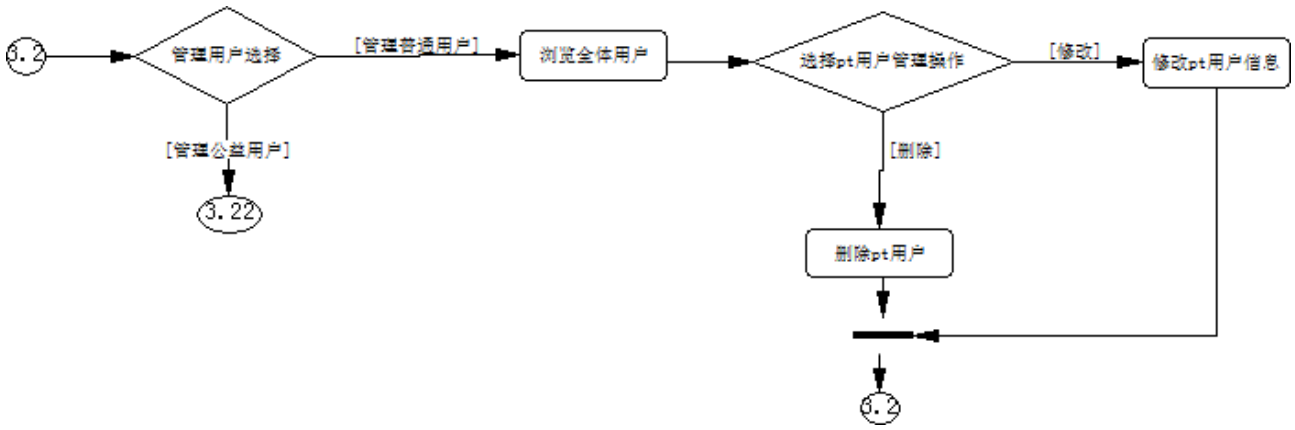

图 2. 23 后台—普通用户管理

图上描述的是关于后台管理普通用户的活动图。

管理普通用户：首先管理选择要管理普通用户，系统会返回所有普通用户的列表以供管理，管理对于普通用户有两个操作选项；一、根据用户列表选定特定用户修改用户信息，二、根据列表单个删除用户。

注：关于以上的描述中，后台管理需要审查的步骤比较繁琐，会导致系统运行效率低下，所以部分审查功能会通过系统自动根据词语匹配的方式执行（实现前期，部分审查功能将不会实现）。

2.3 时间顺序分析

时间顺序分析采用 UML—时序图来描述。时序图——根据时间的先后顺序描述程序的执行步骤。时序图主要由对象、参与者、消息、生命线、控制焦点等元素组成，对象是在时序图中主要的交互角色，参与者可以是人或者系统，消息就是对象之间相互交互时的信息，生命线是对象存在的时间长度，控制焦点代表对象在焦点处被激活并执行相关的操作。以下将

一一细化上面所描述的业务到具体程序执行步骤。

1. 注册

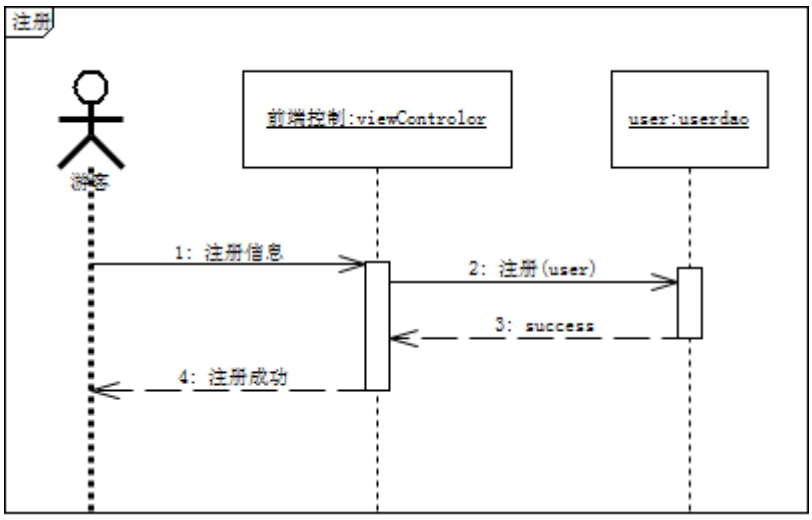

图 2. 24 注册—时序图

图上描述的是关于注册的时序图。图上的参与者是游客（人或者某系统），主要的对象有前端控制对象和 user 对象；参与者游客通过前端（浏览器）给系统发送消息——注册，并且填写好了注册表单发送给了前端控制对象；前端控制对象主要处理用户对前端的请求并交给相应的对象进行处理并得到反馈，然后反馈给用户前端控制收到游客的注册消息后随即就给 user 对象发送消息——执行注册功能，并把游客的注册信息一并发送给了 user 对象；user 对象就是实际执行对象，它收到前端控制对象的消息——执行注册功能后，立马执行了注册功能，并给前端控制对象发送反馈——success（注册成功）；前端控制对象收到由 user 对象发来的反馈后立即给游客发消息——注册成功。以上游客就是注册时具体的程序执行顺序。

2. 登录

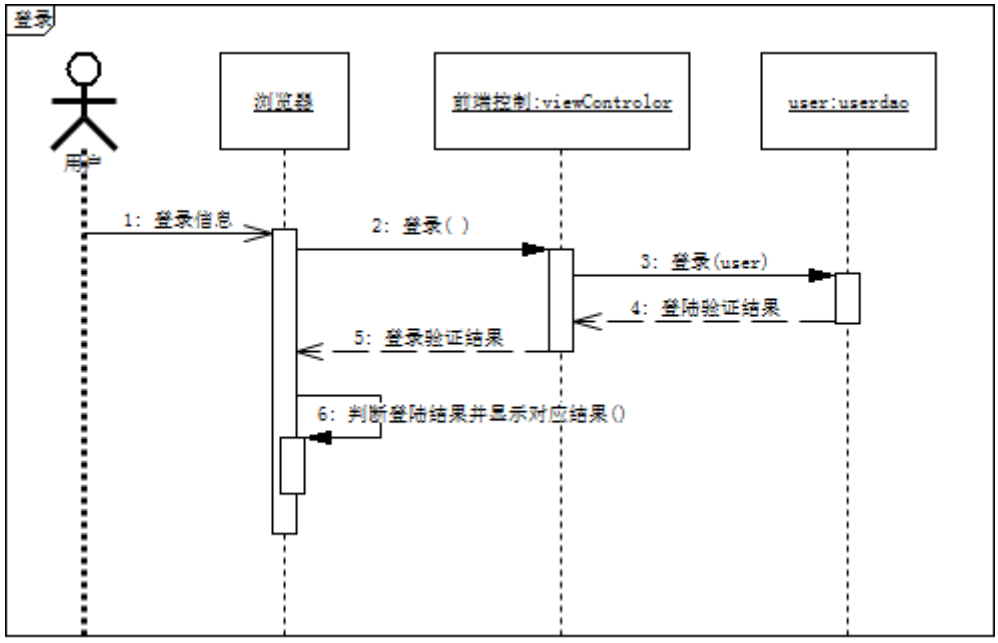

图 2. 25 登录—时序图

图上描述的是登录时序图。参与者是通过注册的用户， 能与用户直接互动的对象就是浏览器，用户把登录信息给浏览器，浏览器在通过 http 协议把数据发送给后台；与登录相关的后台对象有前端控制对象和 user 对象；当前端控制对象收到浏览器发送的登录请求后，随即调用 user 对象的登录方法，并把当前登录用户的信息发送给 user 对象；user 对象接收到前端控制对象的调用后立即执行登录方法，执行完后返回执行结果给前端控制对象，再由 前端控制对象发送登录结果给当前用户所使用的浏览器，浏览器接收到结果后，再自行判断登录结果并根据结果显示相应的页面，已告知用户登录成功与否。以上便是用户登录的时序图。

3.编辑用户信息

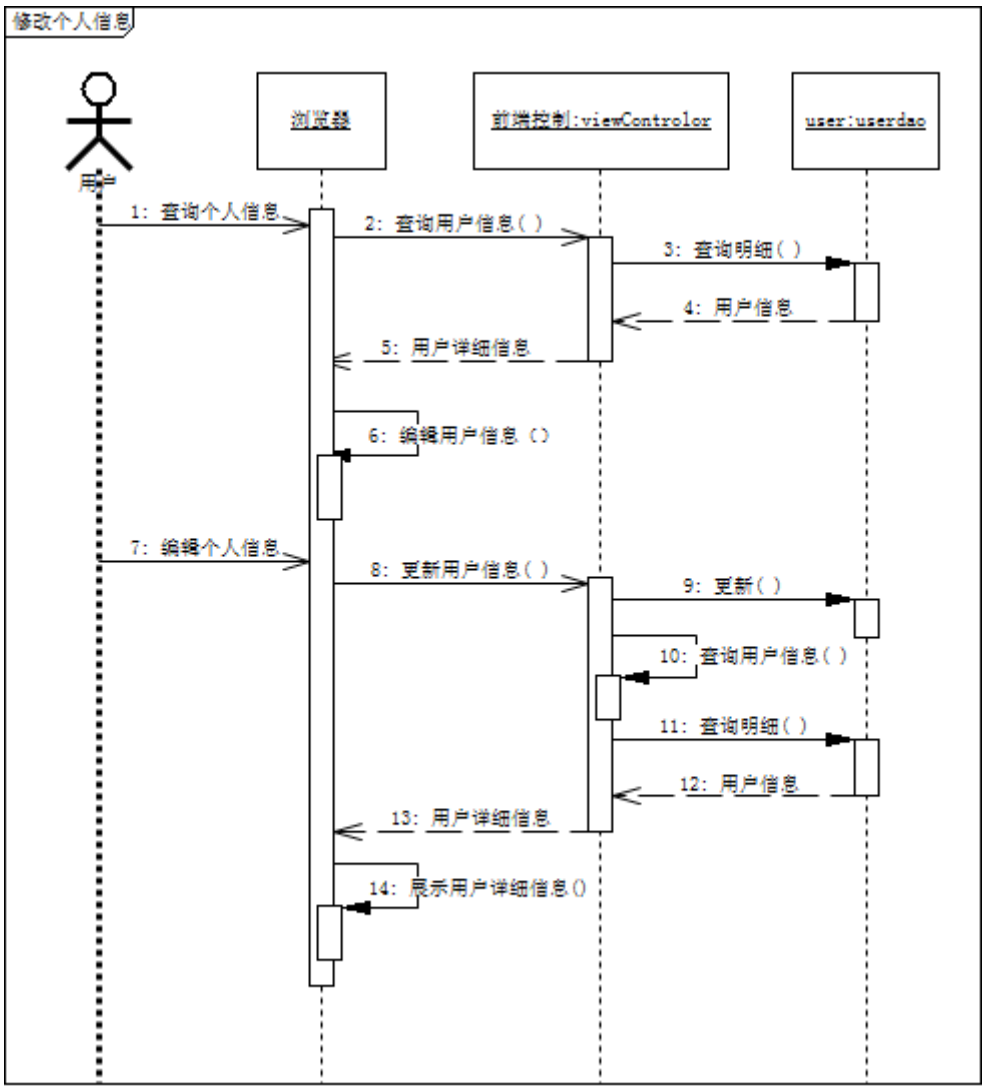

图 2. 26 编辑用户信息—时序图

图上描述的是用户编辑个人信息时序图。关于用户编辑个人信息的一个整体过程：首先用户先要看到自己的所有信息然后编辑，再提交修改，最后还需要看到修改后的数据。参与者是通过注册并且已登录的用户，如果要修改个人信息，用户状态必须是已登录；主要参与的对象有浏览器（与用户的直接交互的对象）、前端控制、user(用户实体类数据操作对象)。用户首先通过浏览器向服务器发送查询个人信息的请求，浏览器向后台对象前端控制发送请求获取用户数据，前端控制对象再调用 user 对象以获取用户数据，user 对象接收到调用指令后执行查询方法，将得到的用户信息返回给前端控制对象，再由前端控制对象将用户信息发送至浏览器，再由浏览器显示用户详细信息。

以上完成了用户查询个人信息的步骤，接下来将描述用户修改个人信息详细步骤。用户在浏览器看到个人信息后，选择编辑并且提交编辑后的个人信息。浏览提交修改后的个人信

息到后台对象前端控制，前端控制对象再调用 user 对象以执行更新用户信息的功能，user 接收到调用后执行。由于前端控制对 user 对象发出的同步消息，所以要等 user 对象执行完之前的指令才能继续执行之后的步骤，当 user 执行完更新后，前端控制继续执行查询用户信息这一步骤，再将通过 user 对象获得的用户信息返回给浏览器，再由浏览器展示给用户。

由此一来，用户便可以看到个人信息前后的变动了。

4. 添加商品

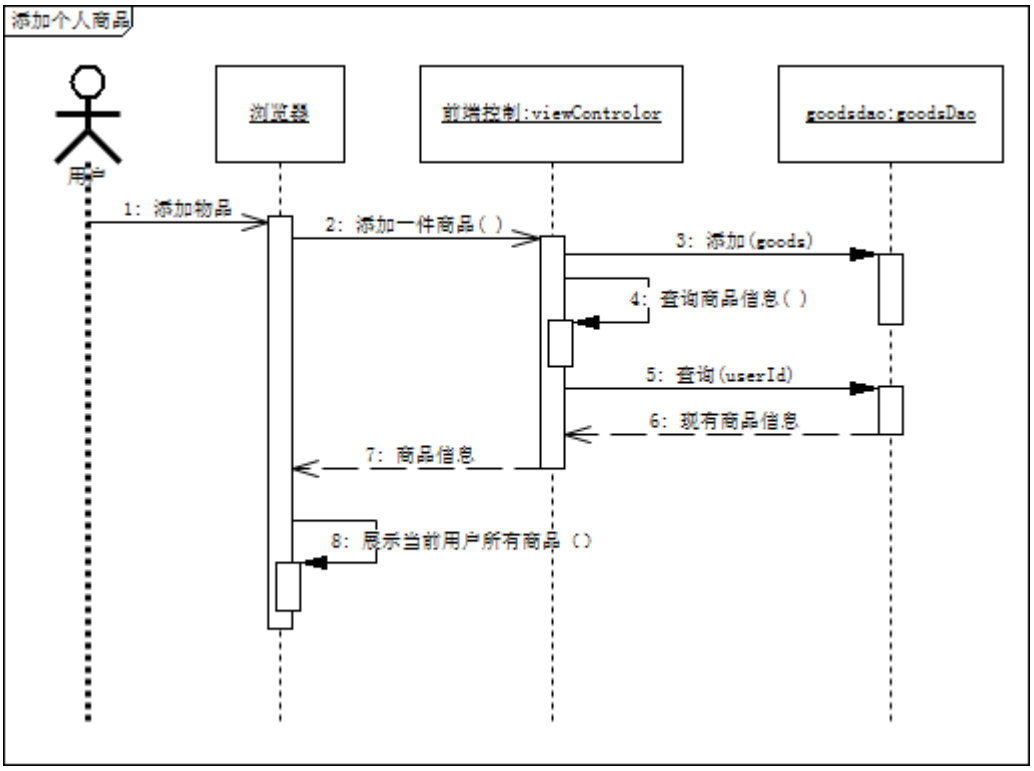

图 2. 27 添加商品—时序图

图上描述的是添加商品时序图。参与者是已经登陆的用户，主要的交互对象有：浏览器、前端控制、goodsdao 等对象。大致过程是用户通过浏览器添加一件物品，之后系统返回增加后的所有物品信息。首先首先用户通过浏览器提交新商品信息表单到后台对象，后台对象前端控制收到浏览器的请求——添加一件商品，前端控制随即调用 goodsdao 对象，前端控制发送消息给 goodsdao 添加商品 goods，goodsdao 收到消息后立即执行添加方法。当 goodsdao 执行完成后，前端控制对象继续执行后续步骤，前端控制对象对自己发送消息——查询当前用户所有的商品信息，需要调用 goodsdao 对象来执行这个功能，goodsdao 接收到消息随即查询了对应用户所有的商品信息并返回给前端控制对象，前端控制对象收到返回值后立即将获得的商品列表发送给浏览器，再由浏览器展示给当前用户，他所有的商品。

以上便是用户添加一件商品的时序图，其中需要说明一点。当用户完成添加步骤后，用户会看到他所有的商品信息，来确定他是否有添加成功。如果未成功，用户可以选择再次添加或者联系管理来解决问题。

5. 编辑商品信息

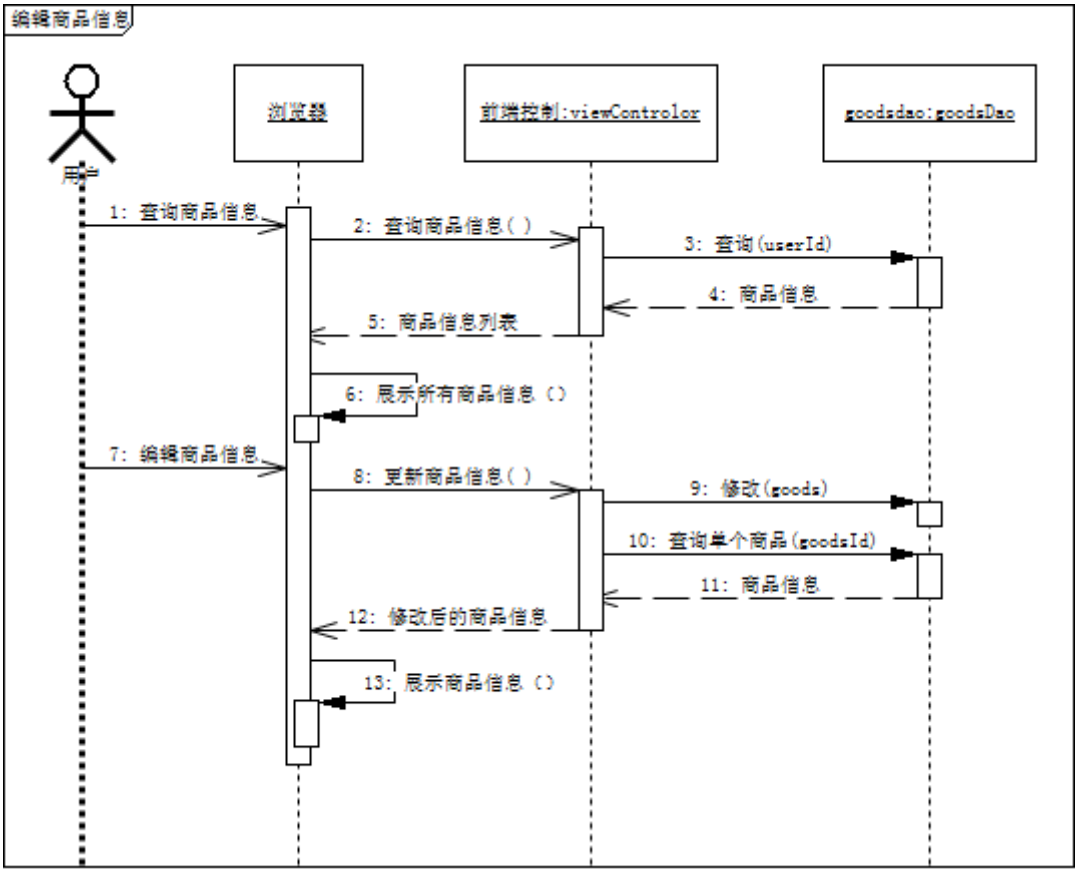

图 2. 28 编辑商品—时序图

图上描述的是关于用户编辑个人商品信息的时序图，大致过程描述：用户首先要看到自己所有的商品信息，然后再选择其中一件商品进行编辑，编辑完成后还需要看到修改后的数据。参与者是已登录的用户，参与的对象有浏览器、前端控制、goodsdao（商品数据处理对象）。以下将详细描述。

用户首先通过浏览器向后台请求获得所有商品信息。浏览器收到用户的请求后向后台对象发送消息查询当前用户所有商品信息，前端控制对象收到消息后随即发送消息给 goodsdao 对象请求获得制定用户的所有商品信息，goodsdao 收到消息后立即执行查询方法将获得的数据传给前端控制对象，前端控制对象收到返回值后立即发送给浏览器，浏览器收到请求的数据后，随即展示给用户。

完成第一步之后，用户根据需求选定特定商品并修改，然后通过浏览器提交保存。浏览

器收到请求后，立即向后台前端控制对象发送消息——更新当前商品信息，前端控制收到消息后随即向 goodsdao 发送消息并传值（修改后的商品）——保存当前修改。goodsdao 收到消息后立即执行更新商品信息功能。待 goodsdao 执行完更新之后，前端控制继续执行查询当前修改的商品信息，goodsdao 收到消息后执行查询并返回商品信息到前端控制对象。前端控制收到返回值后发送给浏览器，浏览器获得请求值后立即展示给用户。用户此时便可以看到修改后的商品的详细信息。

同上添加商品，如果操作不成功，用户可以选择重新操作或者联系管理来解决问题。

6. 删除商品

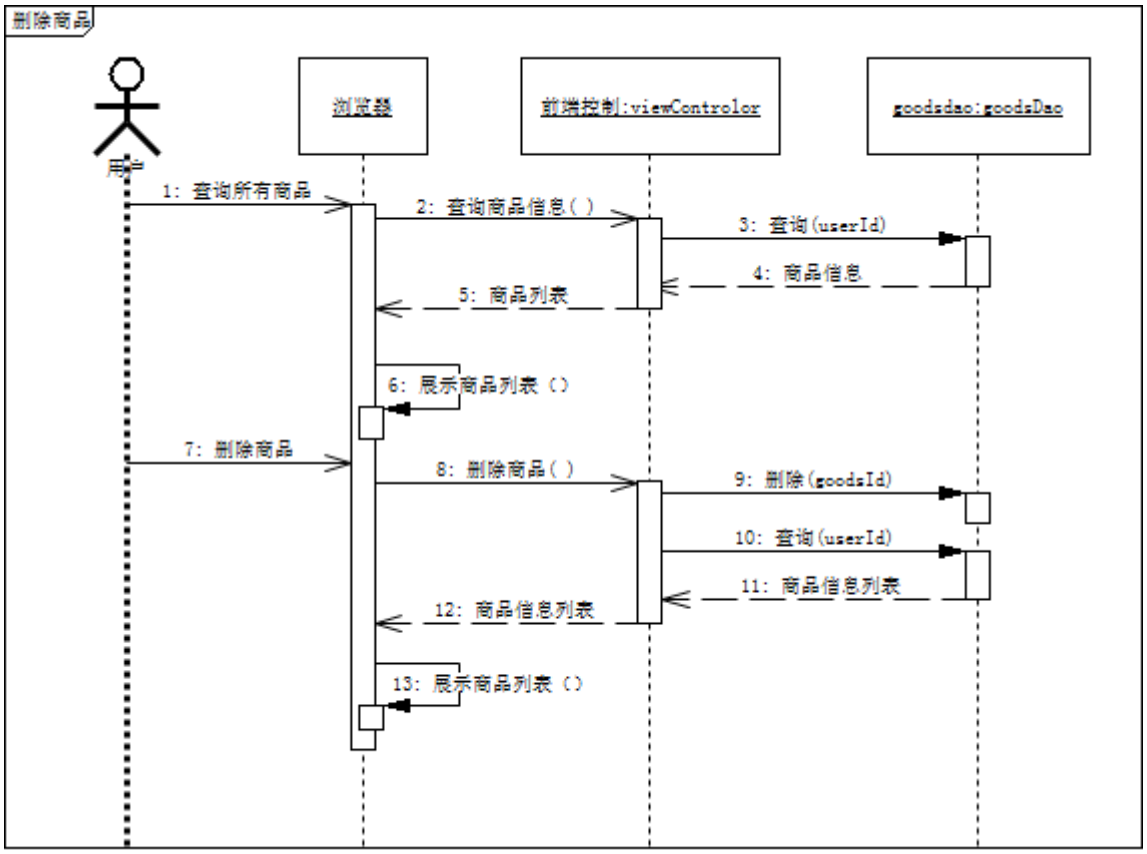

图 2. 29 删除商品—时序图

图上描述的是关于用户删除商品的时序图，大致过程：用户首先要看到商品列表，再根据个人需要删除指定的商品。参与者是已登录的用户，参与对象有浏览器、前端控制、goodsdao；首先用户通过浏览器向后台对象请求获得个人所有商品信息，浏览器收到消息后，立即向后台对象前端控制发送请求获得当前用户所有商品信息，前端控制对象收到请求后立即调用 goodsdao 对象获取浏览器请求的数据，goodsdao 受到调用后立即根据用户 Id 执行查询方，然后将查到的信息返回给前端控制对象，前端控制对象收到数据后立刻发送给浏览器。

浏览器收到请求的数据后立即展示给用户以备操作。

以上完成了整体过程的第一步，随后用户通过浏览器展示的数据，再根据个人需求选定要删除的商品并提交。浏览器受到用户的请求后，立即向后台对象前端控制发送删除当前商品的请求，前端控制对象收到请求后立即调用 goodsdao 根据当前商品 ID 执行删除。随后前端控制对象再调用 goodsdao 对象查询删除后当前用户对应的商品列表，前端控制对象获得 goodsdao 返回的数据后，立即发送给浏览器，浏览器收到请求的结果后立即展示给用户。

关于删除操作，有一点需要说明，暂时未设计批量删除的功能与时序图。

同上，如果用户操作不成功，可以选择重新操作或者联系管理以解决问题。

7. 主动交易

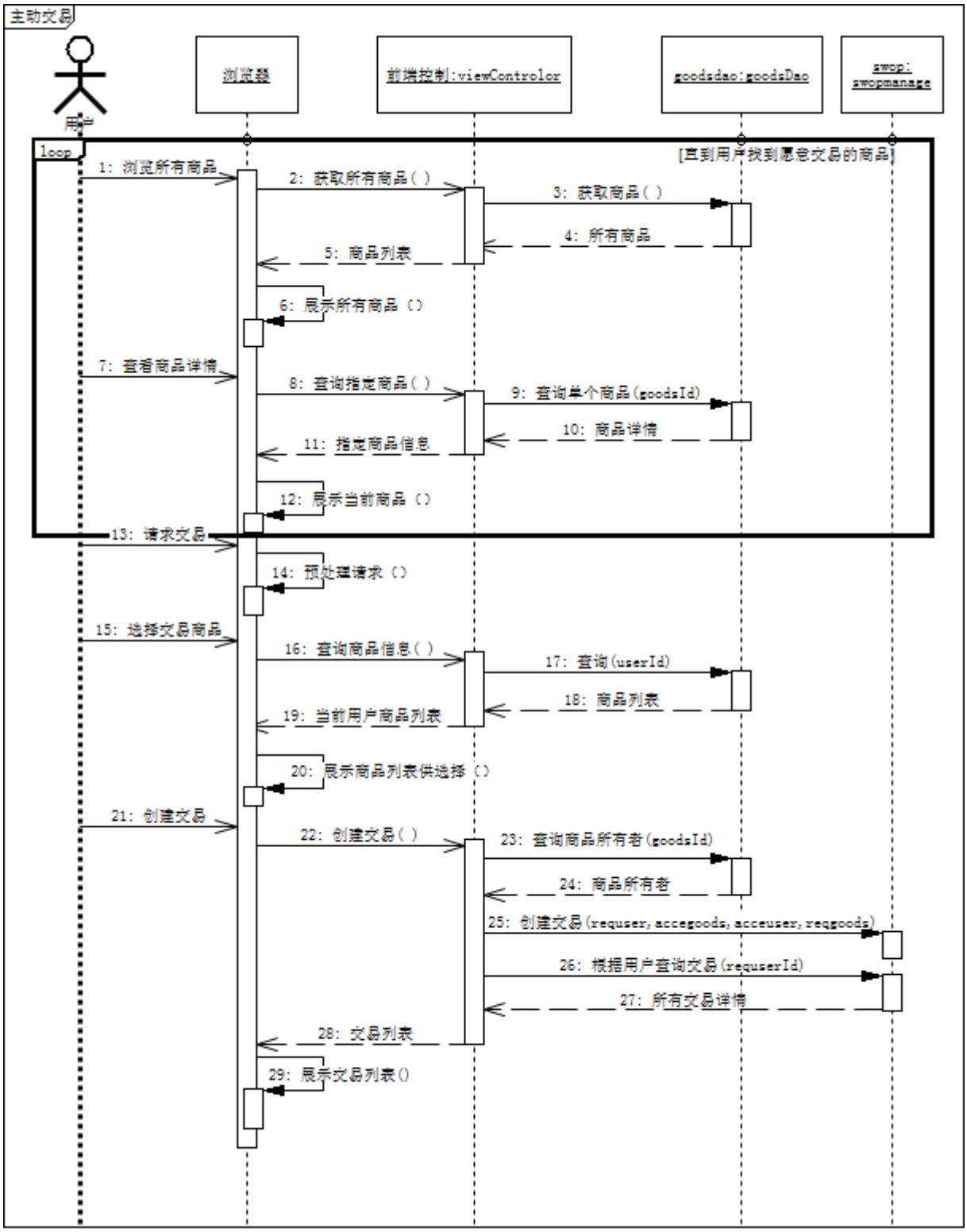

图 2. 30 主动交易—时序图

图上描述的是关于用户主动进行交易的时序图，大致过程：用户首先再所有的可交易的商品中选出愿意交易的商品，然后再从个人商品中选取出用来交易的物品。参与者是已登录

的用户，参与的对象有浏览器、前端控制、goodsdao、swop（交易数据处理对象）等。首先用户通过浏览器发送请求——浏览所有可交易商品，浏览器受到请求后向后台对象前端控制发送请求获取所有商品，前端控制对象受到请求后立即调用 goodsdao 对象的方法获取所有可用商品，前端控制收到返回值后就把数据发送给浏览器，浏览器收到请求的数据后，立即展示给用户。

以上完成了整体进程的第一步，当用户看到所有商品后，根据个人需求选定商品要求查看详情以获得更多信息。浏览器收到用户的请求后，立即向后台对象前端控制发送请求以获得当前商品的详细信息，前端控制对象收到浏览器的请求后立即调用 goodsdao 根据当前商品 Id 查询商品详情，前端控制收到 goodsdao 的返回值后立即发送给浏览器，浏览器获得请求的数据后立即展示给用户。

以上完成了整体过程的第二步。以上两步是一个循环，循环终止的条件是直到用户找到愿意交易的商品，否则就要不断重复上述步骤。

当用户找到想交易的商品后就可以继续向下执行。用户找到心悦的商品后发出交易请求，浏览器受到请求后，没有直接向后台对象反馈，而是预处理请求，预处理方式就是让用户在发送请求之前选择要用来交易的物品。用户看到浏览器的反馈后立即请求要看自己有哪些商品可以用来交易，浏览器接收到请求后立即向后台对象前端控制发送请求，请求获得当前用户所有的可交易的商品。前端控制收到请求后立即调用 goodsdao 通过当前用户 Id 获取对应的所有商品。当前端控制收到 goodsdao 的执行后的返回值时，立即将所得数据（商品列表）发送给浏览器。浏览器收到数据后立即以单选的方式展示给用户。这样用户就可以选择自己用来交易的物品了，这是第三步。

第四步，用户选择好交易商品后立即通过浏览器向服务器发送创建交易的请求。浏览器受到用户的操作后，立即向后台对象前端控制发出请求创建一条交易。前端控制收到请求后为了创建这条交易还需要一个元素就是被请求的用户，所以还需要通过 goodsdao 根据请求的商品 Id 来获得被请求的用户，即请求用户——acceuser。前端控制对象获得该用户值后随即激活 swop 对象（交易数据操作对象）通过 requser（请求的用户）、accegoods（被请求的商品）、acceuser(被请求的用户)、reqgoods（请求的用户的商品）执行创建交易的方法。当 swop 执行完上述步骤之后，前端控制又要求 swop 根据 requserId（请求的用户的 Id）查询所有正在进行的交易。前端控制获得要求的数据后立即发送给浏览器，当浏览器收到请求的

数据后立即展示给用户。以便用户查看当前交易的进度。

同上，如果用户操作无法成功，用户可以选择重新上述操作或者联系管理以解决问题。

8. 被动交易

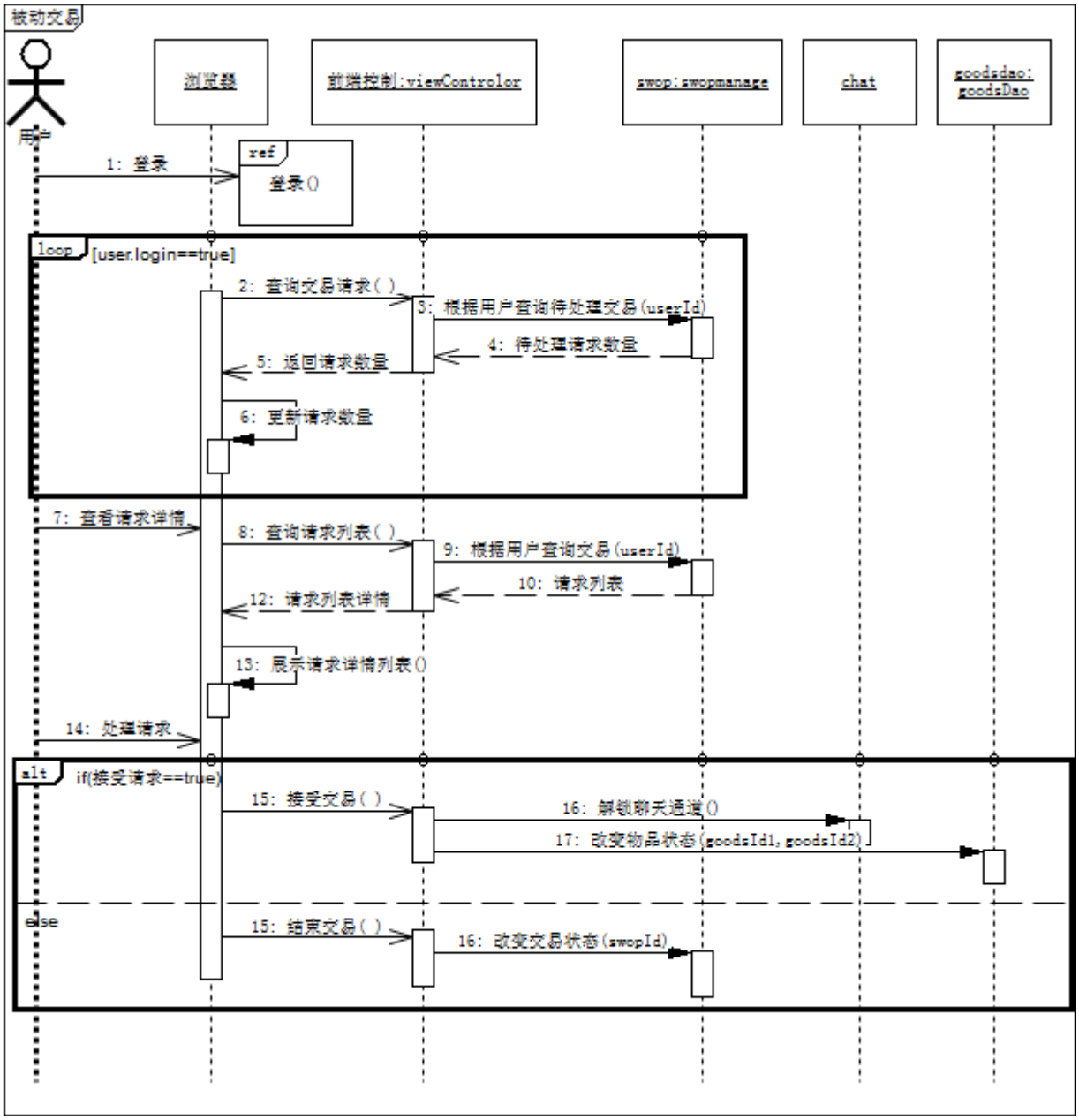

图 2. 31 被动交易—时序图

图上描述的是关于用户被动交易的时序图,大致过程,首先用户根据浏览器显示的请求查看请求详情,再定夺是否接受交易，如果接受就解锁聊天通道否则结束交易。参与者是已登录的用户，参与的对象有浏览、前端控制、swop、goodsdao、chat（网页聊天系统）。

第一步，用户先登录，登录后不论用户在做什么。浏览器都会执行下述循环。在没有用

户的请求下，浏览器主动向后台发送请求。请求获得关于当前用户的所有待处理的交易，前端控制收到请求后立即调用 swop 通过当前用户身份信息查询相关待处理的交易数量。前端控制收到 swop 返回的数据后立即发送给浏览器，浏览器接收到请求的数据后立即更新页面上关于待处理交易请求的数量。以上就是循环部分，循环的条件《user.login==true》,就是只要用户的状态是登录的浏览器就会不断地执行上述主动行为。

第二步，当用户看到更新的数量后，根据个人意愿选择查看待处理交易请求详情。如果待处理请求数为零，系统默认为用户不会请求查看详情。浏览器受到用户的请求后立即向后台对象前端控制请求获得当前用户待处理请求的详细信息。前端控制收到请求后立即激活 swop 对象并使其根据当前用户信息查询相关待处理的交易详情。前端控制收到 swop 对象返回的数据后立即发送给浏览器，浏览器立即展示给用户以供处理。

第三步，用户看到待处理的请求后详情后，会有两个选择：接受请求、不接受请求。如果用户选择了接受请求，浏览器会向后台请求接受请求的程序，前端控制收到接受请求并请求后续程序的消息后，立即向 chat 对象（网页聊天系统）发送消息，请求解锁当前交易涉及的用户之间的聊天通道。当 chat 对象执行完后，前端控制继续执行后续步骤，调用 goodsdao 根据当前交易的物品信息锁定他们的状态，使得无法被其他用户主动或被动的交易。如果用户不接受请求，浏览器接收到用户的回馈后，向后台发送消息请求不接受请求的后续步骤，前端控制对象收到请求后立即激活 swop 对象根据当前交易 Id 执行结束交易操作。

同上，如果操作不成功，用户可以通过重新操作或者联系管理以解决问题。

9. 管理删除用户

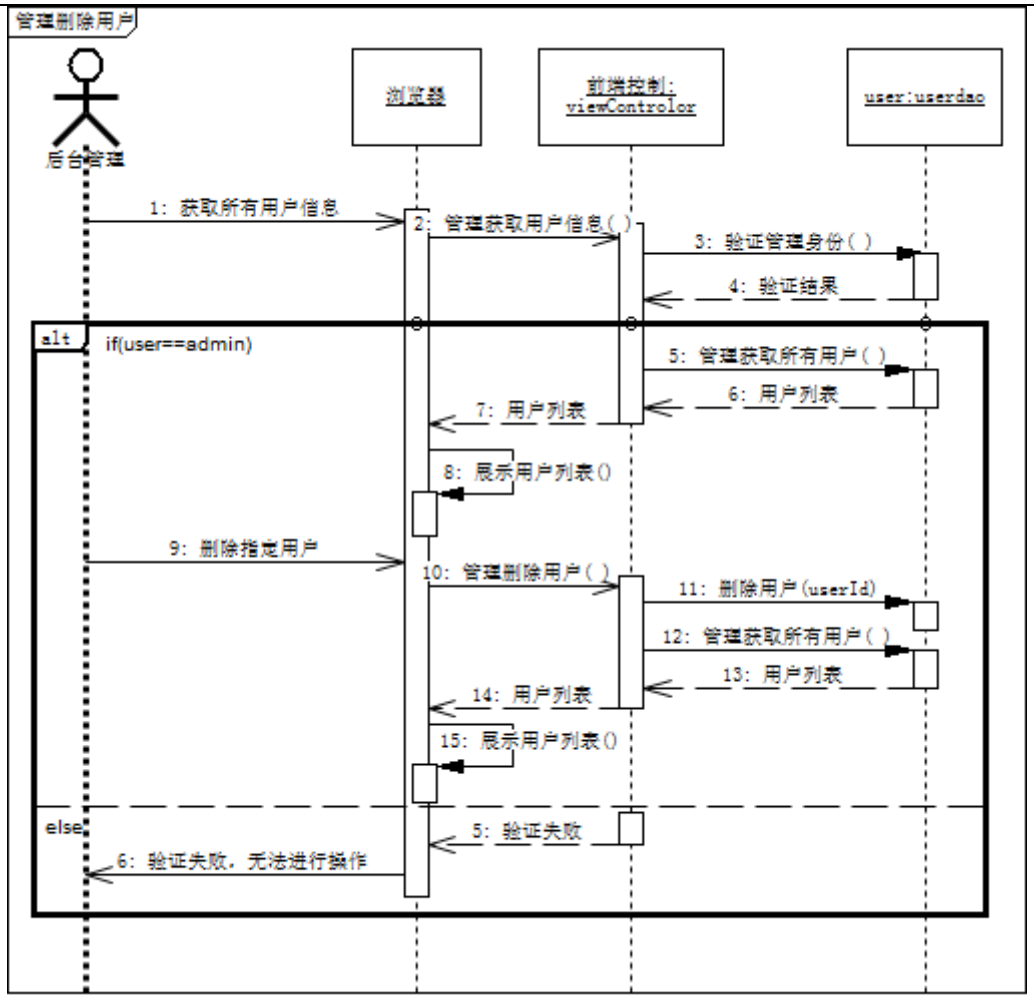

图 2. 32 管理删除用户—时序图

图上描述的是关于后台管理删除用户的时序图，大致过程：管理首先要获取所有可用用户数据，然后根据用户具体情况决定删除与否。参与对象是已登录的管理，参与的对象有浏览器、前端控制、user 等。

第一步，管理通过浏览器请求所有可用用户数据，浏览器收到请求后立即向后台对象前端控制发送请求获得用户数据。前端控制收到请求后并没有直接相应当前请求，因为当前请求的数据是很隐私的，所以前端控制对象手续必须要确认一下当前用户的身份信息。因此，前端控制对象首先通过 user 对象根据当前用户的 Id 查询身份是否是管理员。

第二步，前端控制对象得到身份验证结果后，才能继续向下执行，为保证程序不会出现逻辑漏洞，所以，必须假设当前的“管理”身份的验证结果可能不再是管理，也可能还是管理，以下将分类阐述。

如果当前管理任然是管理，前端控制确认当前用户的身份是管理后，才执行获取当前管理请求的数据——获取所有可用用户信息。前端控制再次激活 user 对象执行获取当前所有可

用用户信息，前端控制得到 user 返回的数据后，立即发送给浏览器，浏览器接收到数据后立即展示给用户。管理看到用户列表后，再根据用户信息判断需要删除的用户。管理制定要删除额度用户后，浏览器接收到请求，立即向后台对象前端控制发送请求删除当前指定用户。前端控制收到消息后，立即激活 user 对象根据当前指定的用户 Id 执行删除。当 user 对象执行完删除步骤之后，前端控制继续激活 user 对象执行获取所有当前可用用户信息，这次查询不用验证身份信息，因为在执行删除功能时管理还是管理，程序执行速度可以不计，所以可以不用再这里再次验证当前管理的身份信息。

回到第二步开始处，如果当前管理已经不再是管理，则前端控制会向浏览器发送一条消息“验证失败”，浏览器接收到前端控制对象的消息后立刻明白了，立即向当前用户展示消息“身份验证失败，无法操作”。

同上，如果操作不成功可以重新操作或者联系系统维护人员（如果有的话）以解决问题，除了身份验证失败以外。

10．管理编辑用户

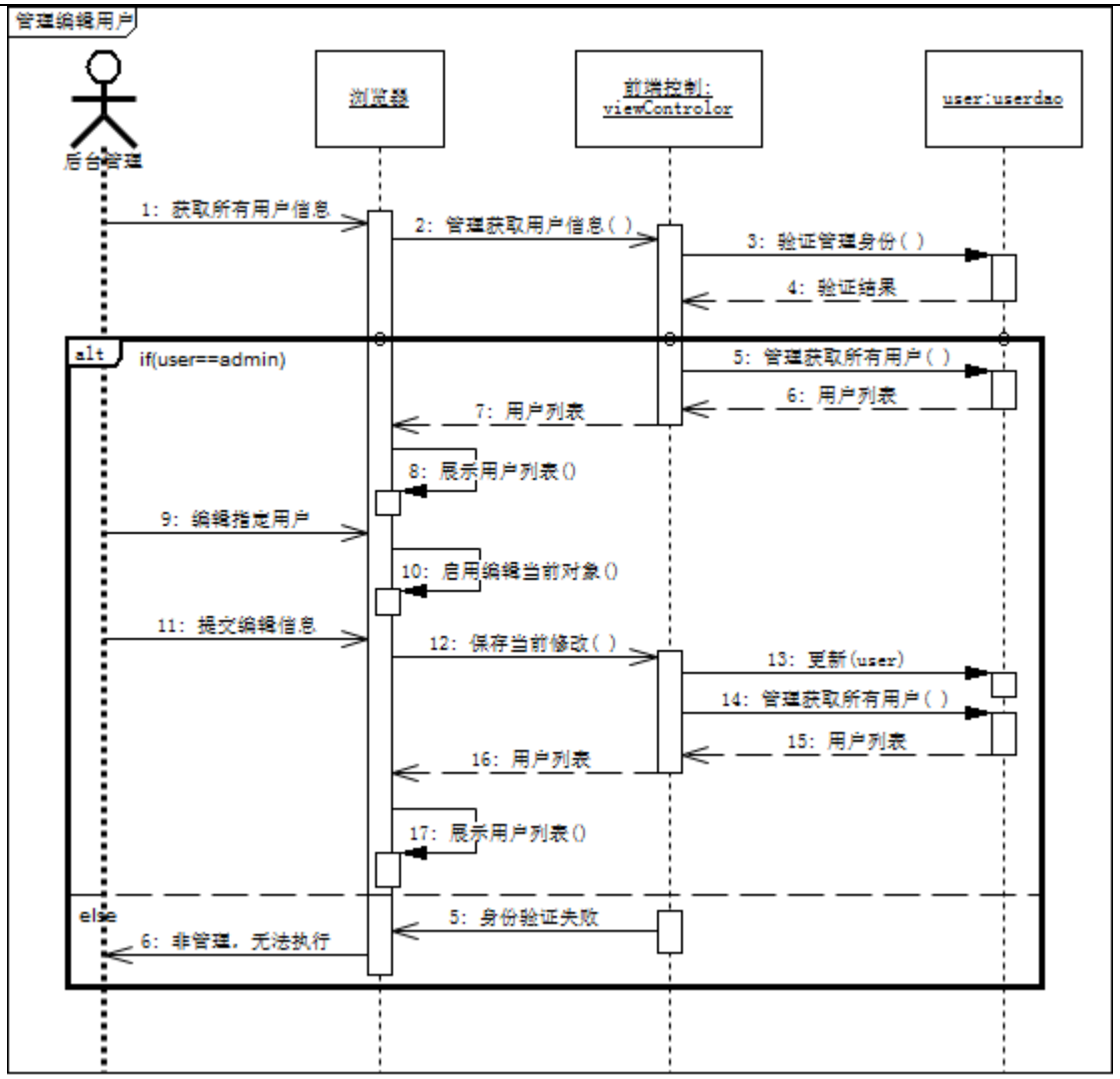

图 2. 33 管理编辑用户—时序图

图上描述的是关于管理编辑用户信息的时序图，大致过程：管理首先获取当前所有可用用户，然后根据用户情况编辑用户信息类似于管理删除用户的过程。参与者是已登录的管理员，参与对象有浏览器、前端控制、user。

第一步同管理删除用户的第一步，验证管理身份信息。

第二步直到第八条消息处相同，之后的步骤就不相同了。到第八条消息，如果当前用户时管理，浏览器展示用户列表给管理。管理根据用户信息判别需要修改的用户并请求修改，浏览器收到用户的请求后，立即启用当前管理指定用户信息的编辑功能。管理编辑完成后提交保存，浏览器收到请求后立即向后台对象前端控制发送请求保存当前修改用户信息。前端控制收到信息后立即激活 user 对象根据当前用户更新信息，当 user 执行完更新后，前端控制继续要求 user 对象查询所有用户详细信息列表。前端控制收到 user 的返回值后立即发送

给浏览器，浏览器接收到请求的结果后立即展示给用户。

回到第二步，如果当前用户不是管理，则前端控制会向浏览器发送一条消息“验证失败”，浏览器接收到前端控制对象的消息后立刻明白了，立即向当前用户展示消息“身份验证失败，无法操作”。

同上，如果操作不成功可以重新操作或者联系系统维护人员（如果有的话）以解决问题，除了身份验证失败以外。

11. 管理编辑商品

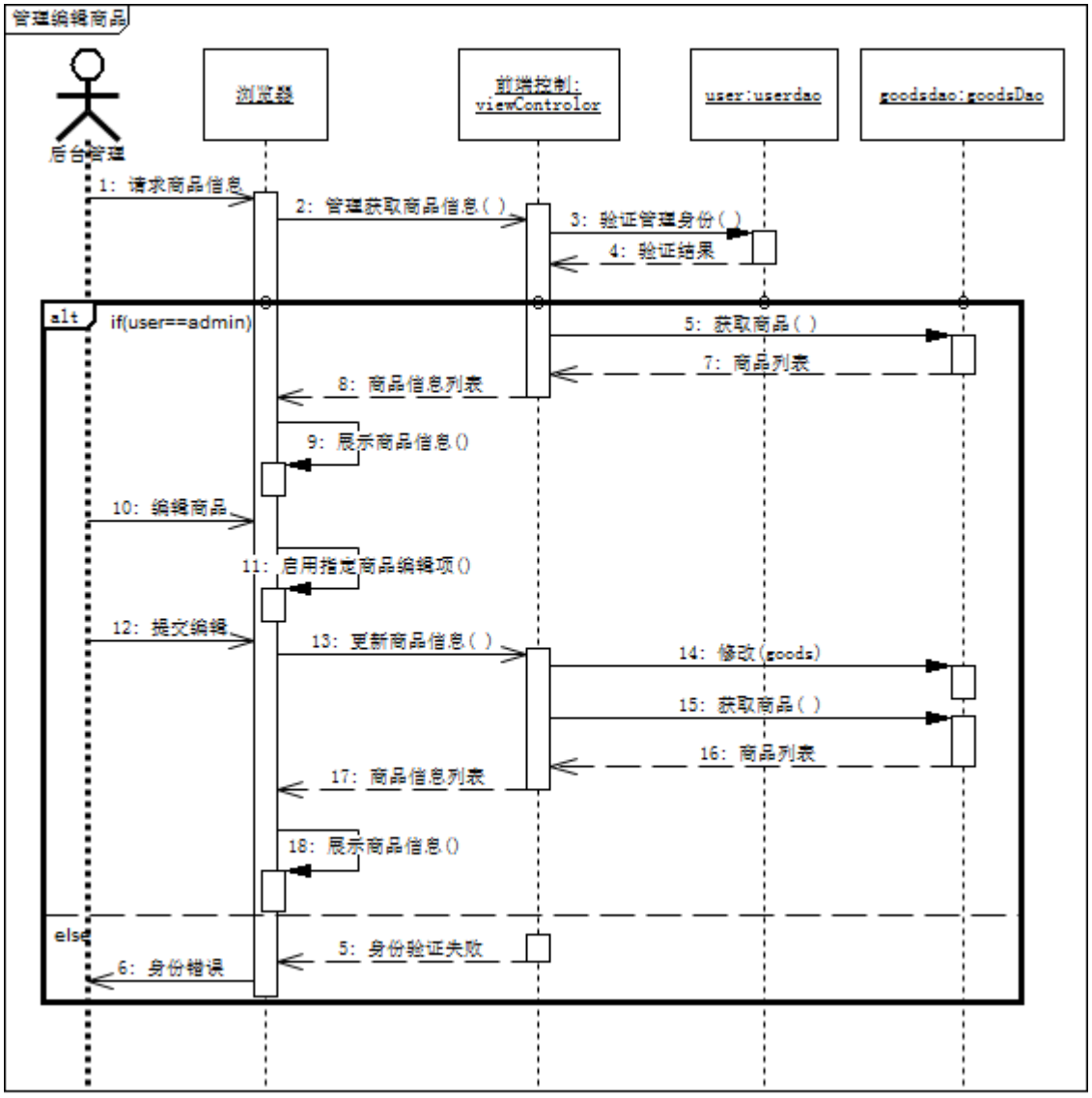

图 2. 34 管理编辑商品—时序图

图上描述的是关于管理编辑商品信息的时序图，大致过程：当前用户(管理)请求获得所有商品信息，然后再根据具体情况修改商品信息。参与者是已登录的管理员，参与对象有浏

览器、前端控制、user、goodsdao。

第一步，用户首先通过浏览器商品信息，浏览器收到请求后立即向后台对象发送请求，前端控制收到请求后并没有直接回应当前请求，而是首先验证用户身份信息，如果是管理再做动作，如果不是另做其他操作。前端控制根据当前用户 Id 激活 user 对象查询验证当前用户的身份信息。

以下是第二步，验证结果要么当前用户是管理，要么不是。第一种情况，如果是管理的。前端控制立即激活 goodsdao 对象并执行获取所有用户的操作。前端控制对象获得 goodsdao 的返回值之后立即发送给浏览器，浏览器收到数据后立即真实给用户。用户根据具体情况决定需要编辑的商品，浏览器相应用户操作，启用对应的商品的编辑功能。用户完成编辑后要提交修改，浏览器立即向后台对象发送请求保存当前修改。前端控制对象收到请求后立即激活 goodsdao 对象根据当前商品信息执行保存修改功能。当 goodsdao 执行完更新后，前端控制对象继续调用 goodsdao 对象获取当前所有可用商品信息。前端控制接收到 goodsdao 的返回数据后，立即发送给浏览器，浏览器接收到请求的数据后立即展示给用户。

另一种可能，如果验证结果表明当前用户不是管理。前端控制想浏览器发送消息“验证失败，拒绝执行请求”，浏览器收到信息后，向用户展示消息“用户身份非管理，非法操作”。

同上，如果操作无法完成，可以通过重新操作或联系系统维护人员（如果有）以解决问题。

12. 管理删除商品

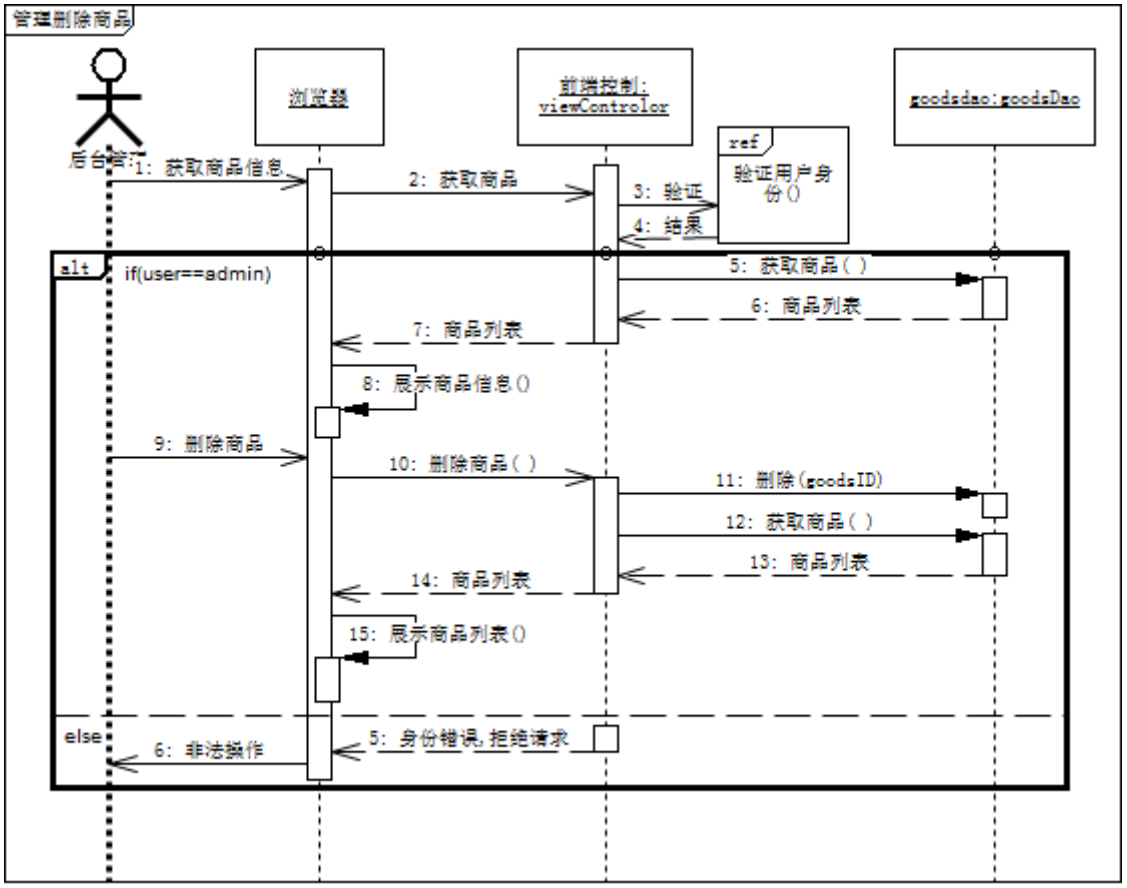

图 2. 35 管理删除商品—时序图

图上描述的是关于管理删除商品的时序图，大致过程：用户首先请求所有商品信息，获得请求数据后，再根据具体情况删除部分商品。参与者是管理员，参与对象有浏览器、前端控制、goodsdao。

第一步，用户以管理员身份请求所有商品信息，浏览器接收到请求后首先验证当前用户身份是否管理员。通过引入封装好的静态方法——验证用户身份，返回值是 Boolean 类型的。

如果返回值为 TRUE，前端控制激活对象 goodsdao 并执行查询所有事商品信息的功能。前端控制收到 goodsdao 返回的数据后立即发送给浏览器，浏览器接收到请求的数据立即展示给管理，管理再根据具体情况删除商品并提交删除。前端控制收到删除的请求后立即激活 goodsdao 根据 goodsId 执行删除功能，当 goodsdao 执行完删除后，前端控制继续要求 goodsdao 执行获取当前所有可用商品信息。前端控制收到 goodsdao 的数据后立即发送予浏览器。浏览器接收到数据后立即展示给用户，一对比操作前后的效果。

如果返回值 FALSE，前端控制向浏览器发送请求“身份错误，拒绝请求”，浏览器收到消息后立即向用户展示消息“身份错误，非法操作”。

同上，如果操作无法成功，可以重新操作或者联系系统维护人员以解决问题。

13. 强制终结交易

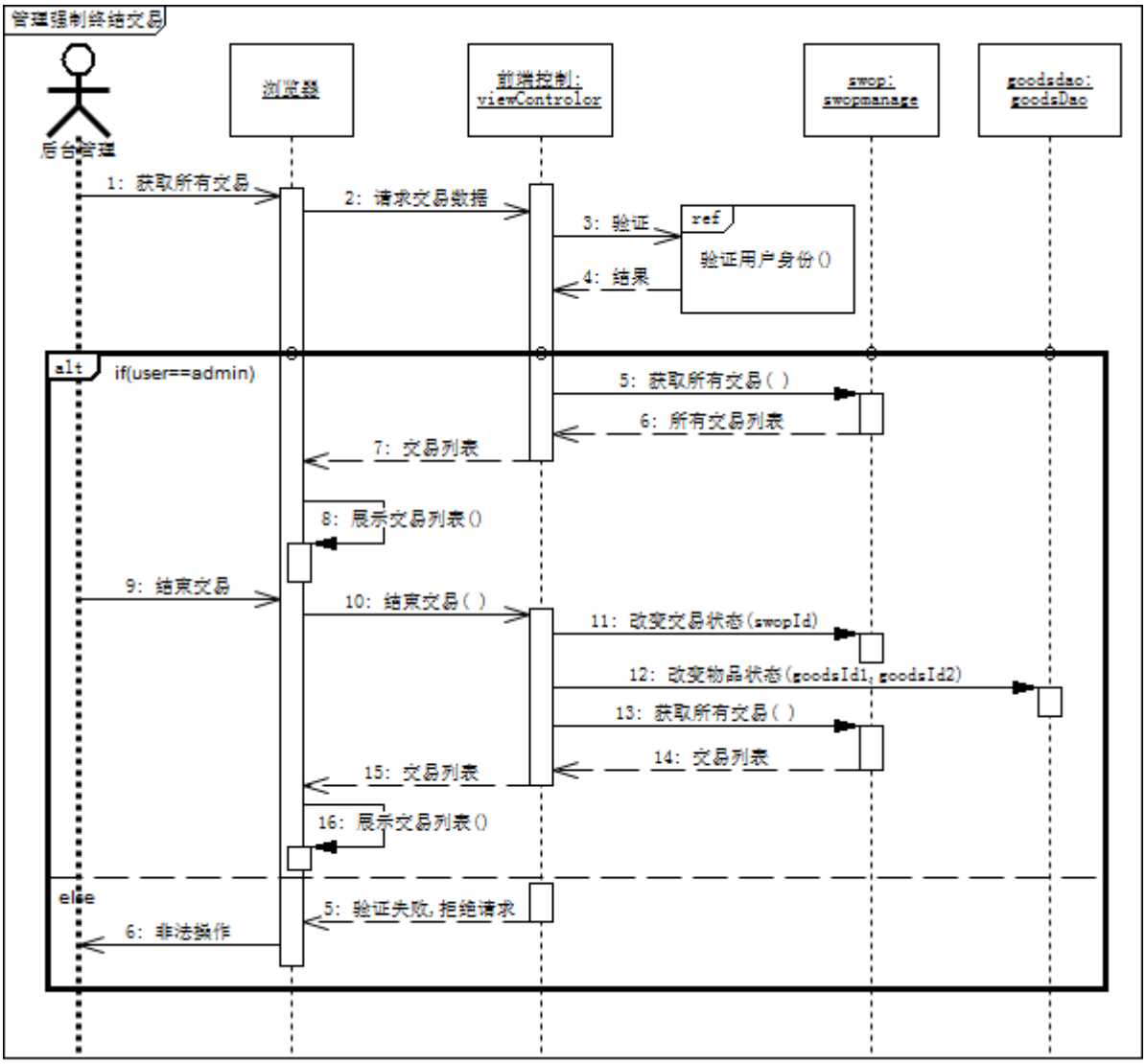

图 2. 36 管理强制终结交易—时序图

图上描述的是关于管理强制结束交易的时序图，大致过程：管理首先获取当前所有正在进行的交易，然后根据具体情况强制结束部分不否和约定的交易。参与者是已登录的管理员，参与的对象有浏览器、前端控制、swop、goodsdao。

说明：关于管理对交易的操作。由于交易的过程比较单调，所以管理对交易的操作暂时只有强制结束交易，以防不合约定的交易达成。

第一步，用户以管理员的身份请求获得所有交易信息，前端控制收到请求后首先验证用户的身份是否是管理员。前端控制通过调用引入的方法——验证用户身份，来确定用户是否是管理，返回值时 Boolean 类型。

如果返回值为 TRUE，前端控制激活 swop 对象执行查询所有交易的方法。前端控制获得 swop 对象返回的交易列表后立即发送给浏览器，浏览器收到请求的数据后立即展示给管理以供操作。管理看到交易列表后根据具体情况删除部分不和约定的交易，浏览器收到删除操作后提交删除到后台对象前端控制。前端控制收到请求后，立即激活 swop 对象执行删除操作。Swop 执行完删除操作后，前端对象继续激活 goodsdao 对象更新涉及交易的商品的状态。Goodsdao 执行完更新后，前端控制继续激活 swop 对象查询当前所有正在进行的交易。前端控制收到 swop 返回的数据后立即发送给浏览器，浏览器接收到请求的结果后，立即展示给用户，以对比操作前后的变化。

如果返回值为 FALSE，前端控制向浏览器发送请求“身份错误，拒绝请求”，浏览器收到消息后立即向用户展示消息“非法操作”。

同上，如果操作无法成功，可以重新操作或者联系系统维护人员以解决问题。

第三章　系统设计

在完成了系统的分析之后，由此进入设计阶段。系统的设计所涉及的内容有类模型的设计、数据库的设计、部署图的设计、构件图的设计等。类模型是研发时系统后台所使用的主体结构和设计思路，类模型就是一个开发框架，在真正研发过程中，程序员所要做的就是在框架中完成对应框架要求的功能；数据库是系统使用的唯一数据存储的工具；部署图是系统开发完成后部署时的框架；构件图是整体业务的一个划分。

3.1　类模型设计

关于类模型的设计需要设计类的细节，以及类与类之间的关系（类图模型）。而类与类之间的关系需要参考之前时序图中各对象之间的关系。类图的所用到的元素有类（class）、依赖关系(dependency)、接口(interface)、关联(association)、泛化(继承:generalization)、组合（composition）、聚合（aggregation）、实现（realization）等。以下将详细描述关于本系统类图的设计。

3.1.1 类的识别

表 3.1 类清单

名称	代码	可见性	抽象	类的类型
User	user	public	FALSE	Class
Userdao	userdao	public	FALSE	Class
goodsDao	goodsDao	public	FALSE	Class
Goods	Goods	public	FALSE	Class
Swop	swop	public	FALSE	Class
Swopmanage	swopmanage	public	FALSE	Class
viewControlor	viewControlor	public	FALSE	Class

3.1.2 类成员的识别

关于类明细，分两种类实体类和方法类。实体类以类属性为主，方法类没有设计任何属性，仅有方法。所以一下将分两类详细描述各个类的明细。

实体类明细

表 3.2 实体类 user 属性表

Name	Code	Data Type	Visibility	Initial Value	Classifier
userID	userID	String	private		user
user_name	userName	String	private		user
password	password	String	private		user
sex	sex	String	private		user
birthdate	birthdate	String	private		user
nick_name	nickName	String	private		user
email	email	String	private		user
userstatu	userstatu	String	private		user
exist	exist	String	private		user
level	level	String	private		user

表 3.3 实体类 swop 属性表

Name	Code	Data Type	Visibility	Initial Value	Classifier
swopID	swopID	String	private		swop
requerId	requerId	String	private		swop
reqgoogsId	reqgoogsId	String	private		swop
accessuserID	accessuserID	String	private		swop
accessgoodsID	accessgoodsID	String	private		swop
swopStatu	swopStatu	String	private		swop
requestdate	requestdate	String	private		swop
finalTime	finalTime	String	private		swop
exist	exist	String	private		swop

表 3.4 实体类 goods 的属性表

Name	Code	Data Type	Visibility	Initial Value	Classifier
goodsId	goodsID	String	private		Goods
goodsName	goodsName	String	private		Goods
userID	userID	String	private		Goods
goodsDescribe	goodsDescribe	String	private		Goods
goodsPic	goodsPic	String	private		Goods
goodsType	goodsType	String	private		Goods
swopWill	swopWill	String	private		Goods
goodsStatu	goodsStatu	String	private		Goods
addDate	addDate	String	private		Goods
exist	exist	String	private		Goods

以上实体类阐述结束，以下是方法类。

表 3.5 方法类 goodsdao 的方法成员

Name	Code	Return Type	Visibility	Classifier
修改	update	void	public	goodsDao
添加	add	void	public	goodsDao
查询	search	List<Goods>	public	goodsDao
查询单个商品	searchById	Goods	public	goodsDao
删除	delete	void	public	goodsDao
获取商品	getallgoods	List<Goods>	public	goodsDao
查询商品所有者	searchuserBygoods	user	public	goodsDao
改变物品状态	updategoodsstat	void	public	goodsDao

表 3.6 方法类 swopmanage 的方法成员

Name	Code	Return Type	Visibility	Classifier
创建交易	creteswop	void	public	swopmanage
根据用户查询交易	searchswopByuserId	List<swop>	public	swopmanage
根据用户查询待处理交易	searchundealswopByuser	int	public	swopmanage
改变交易状态	updateswopstat	void	public	swopmanage
获取所有交易	getallswops	List<swop>	public	swopmanage

表 3.7 方法类 userdao 的方法成员

Name	Code	Return Type	Visibility	Classifier
注册	register	void	public	userdao
登录	login	boolean	public	userdao
查询明细	searchuserinfo	user	public	userdao
更新	update	void	public	userdao
验证管理身份	checkadmin	boolean	public	userdao
管理获取所有用户	admingetalluser	List<user>	public	userdao
删除用户	deleteuser	void	public	userdao

表 3.8 方法类 viewControlor 的方法成员

Name	Code	Return Type	Visibility	Classifier
登录	login	void	public	viewControlor
查询用户信息	searchuserinfo	void	public	viewControlor
更新用户信息	updateuserinfo	void	public	viewControlor
添加一件商品	addgoods	void	public	viewControlor
查询商品信息	searchgoodsinfo	void	public	viewControlor
更新商品信息	updategoodsinfo	void	public	viewControlor
删除商品	deletegoods	void	public	viewControlor
获取所有商品	getallgoods	void	public	viewControlor
查询指定商品	searchgoodsById	void	public	viewControlor

创建交易	createswop	void	public	viewControlor
查询交易请求	searchswops	void	public	viewControlor
查询请求列表	searchswoplist	void	public	viewControlor
接受交易	acceptreqswop	void	public	viewControlor
结束交易	stopswop	void	public	viewControlor
管理获取用户信息	admingetallusers	void	public	viewControlor
管理删除用户	admindeleteuser	void	public	viewControlor
保存当前修改	adminupdateuser	void	public	viewControlor
管理获取商品信息	admingetallgoods	void	public	viewControlor

以上所有方法类中所有方法都是时序图用到的方法，在此就不再重复解释了。实体类是方法类的依赖品，方法需要依赖实体的属性来执行数据的处理。当说明所有的类之后下面开始描述类与类之间的关系。

3.1.3 类关系的识别

1. 商品类的依赖关系

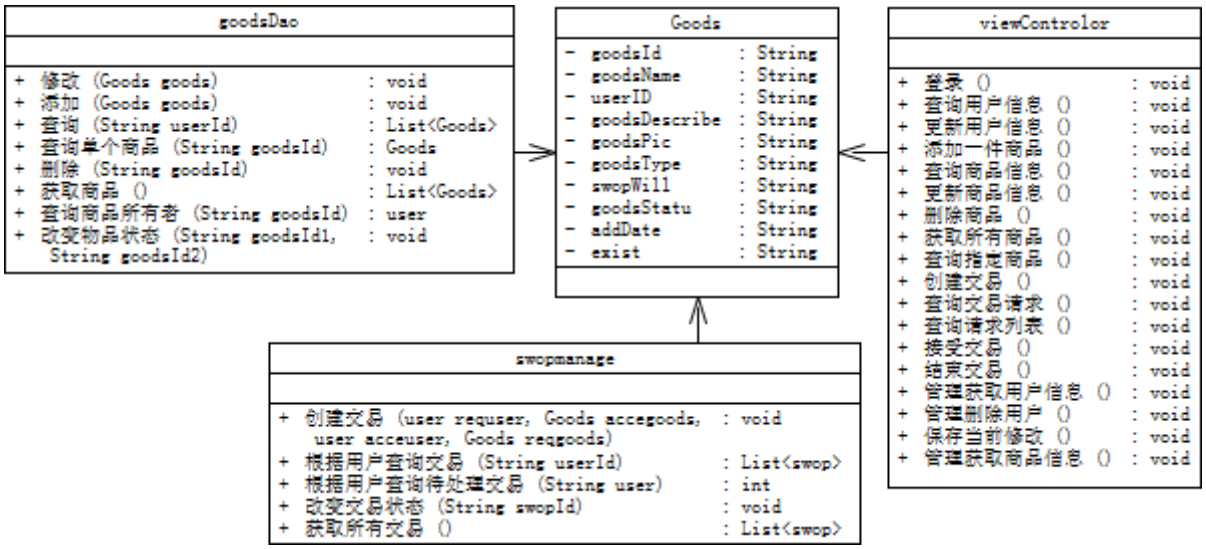

图 3. 1 商品依赖—类图

图上描述的是关于商品依赖的类图。从图中可以看到 goods 类有三个依赖类 goodsdao、viewControlor、swopmanage，这三个依赖类对于 goods 类来说都属于使用依赖。使用依赖，顾名思义就是这三个依赖类之所以依赖 goods 类是因为它们都要使用 goods 类。Goodsdao 类的所有方法都是依赖 goods 类的，所以可以说是“重”依赖关系；swopmanage 的部分方法依赖 goods 类的，所以可以说是“轻”依赖关系；viewControlor 类并没有直接依赖 goods 类，而是间接通过其他类对象来使用 goods 类，所以可以说是“间接”依赖关系。

2. 用户类的依赖关系

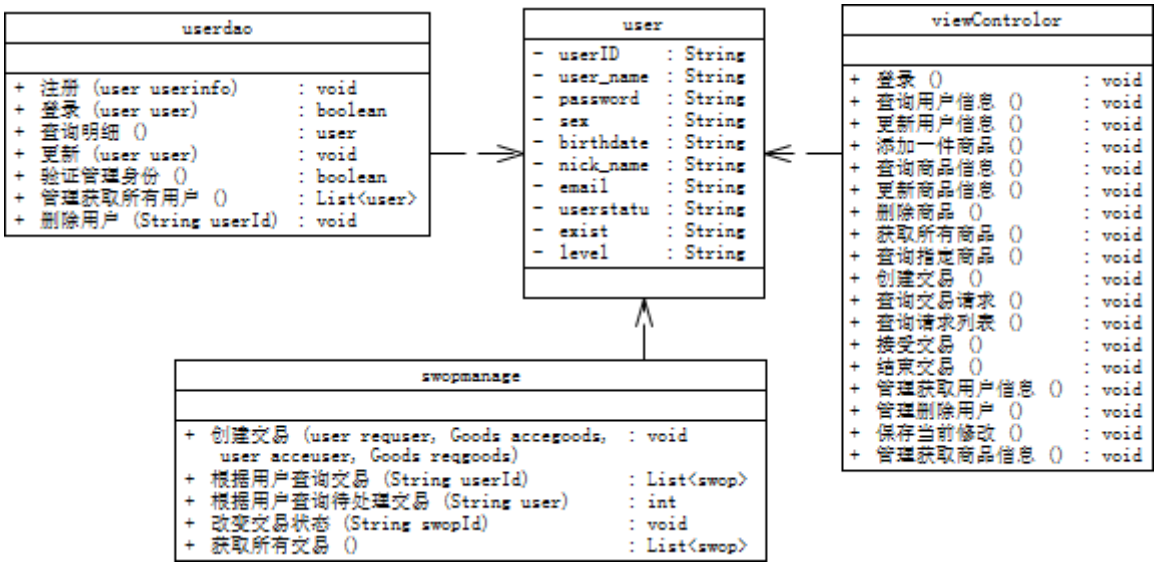

图 3. 2 用户依赖—类图

图上描述的是部分类依赖 user 类的关系图。涉及到的类有 user、userdao、viewControlor、swopmanage。其中 userdao、swopmanage 类在方法上直接依赖 user 类，viewControlor 是间接地通过其他类对象依赖 user 类。Userdao 类是完全依赖，swopmanage 是轻度依赖、viewControlor 是间接依赖。

3. Swop 类的依赖关系

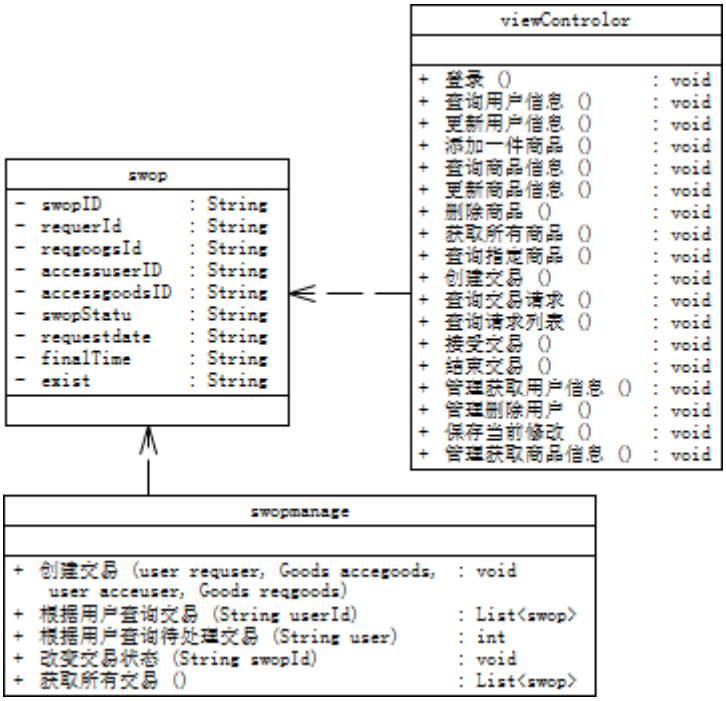

图 3. 3 交易依赖—类图

图上描述的是部分类依赖 swop 类的关系图。涉及到的类有 swop、swopmanage、

viewControlor。其中 swopmanage 在方法上是完全依赖于 swop 类，viewControlor 类在方法上是间接通过其他类对象依赖 swop 类的。

4. viewControlor 类的聚合图

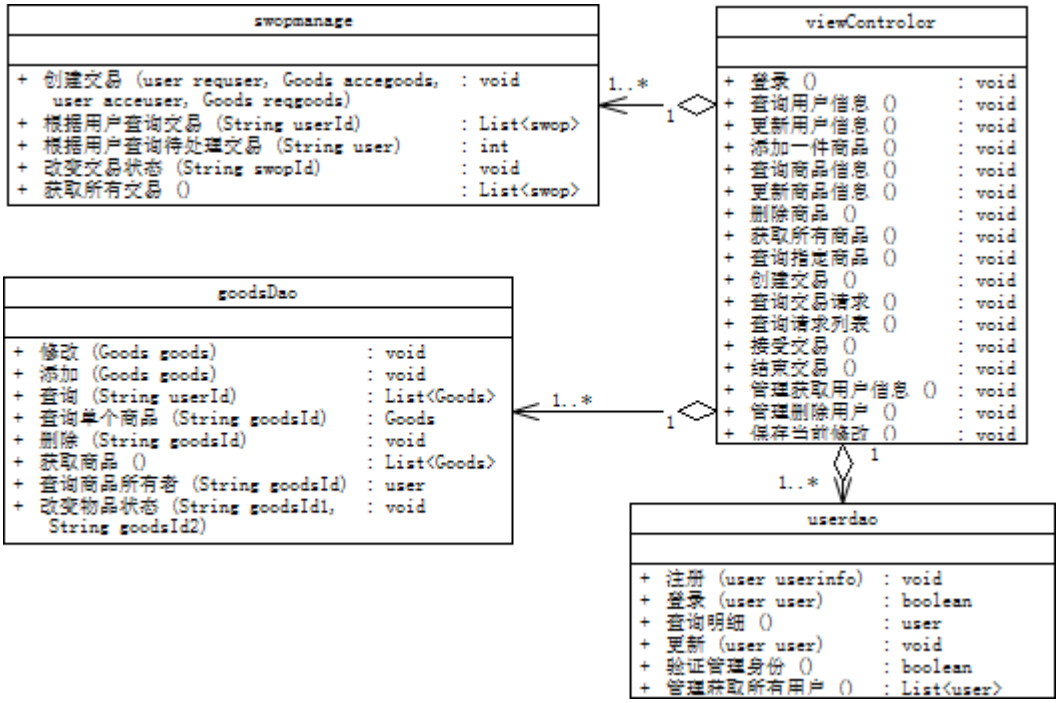

图 3. 4 交易聚合—类图

图上描述的是 viewControlor 与 swopmanage、userdao、goodsDao 之间的关系。swopmanage、userdao、goodsDao 是 viewControlor 的聚合成员，它们之间是聚合关系。VviewControlor 就像是一片森林，而 swopmanage、userdao、goodsDao 就是其中的树木，它们可以存在亦可以不存在。但根据之前时序图的设计，它们必然是 viewControlor 中的树木，而且它们会被创造出很多为 viewControlor 所用。

3.2 数据库的设计

数据是程序的核心，程序的每一步运行都是为了更好地处理数据。而程序不直接与数据打交道，数据存在数据库里，所以直接与程序打交道的是数据库，然后再是程序。数据库的主要组成部分是表、视图、存储过程，这里只涉及到视图，存储和过程暂且不论。数据有序的存储在表里，并且表之间存在相互的依赖关系，形成一个完整的逻辑约束，使得数据有序且能够提高程序开发效率。

3.2.1 基表的识别

表 3.9 数据库表清单

Name	Code

交易	Swop
公益用户	pfUser
公益资源	publicFareGoods
用户	gUser
物品	Goods

从表上可以看出数据库一共五张表，分别是交易表、公益用户表、公益资源表、用户表、物品表。

3.2.2 基表属性的识别

表 3.10 交易表属性明细

Name	Code	Data Type	Primary	Foreign Key
swopID	swopID	varchar(20)	X	
请求用户	requestId	varchar(9)		X
请求物品	reqgoogsId	varchar(20)		X
被请求用户	accessuserID	varchar(9)		X
被请求物品	accessgoodsID	varchar(20)		X
交易状态	swopStatu	varchar(300)		
请求时间	requestdate	varchar(200)		
最终时间	finalTime	varchar(200)		
exist	exist	varchar(10)		

表 3.11 公益用户表属性明细

Name	Code	Data Type	Primary	Foreign Key
userID	userID	varchar(20)	X	
密码	password	varchar(200)		
姓名	userName	varchar(200)		
身份证号码	identityID	varchar(18)		
性别	sex	varchar(18)		
出生日期	birthdate	varchar(100)		
工作单位	work	varchar(300)		
地址	address	varchar(300)		
证明图 2	prooftwoPic	text		
证明图 1	proofonePic	text		
证明图 3	proofthreePic	text		
用户状态	userStatu	varchar(300)		
exist	exist	varchar(10)		

表 3.12 公益资源属性明细表

Name	Code	Data Type	Primary	Foreign Key
goodsID	goodsID	varchar(20)		X
pfID	pfID	varchar(20)	X	

userID	userID	varchar(20)		X
exist	exist	varchar(10)		

表 3.13 用户属性明细表

Name	Code	Data Type	Primary	Foreign Key
userID	userID	varchar(9)	X	
姓名	user_name	varchar(200)		
密码	password	varchar(200)		
性别	sex	varchar(10)		
出生日期	birthdate	varchar(30)		
昵称	nick_name	varchar(300)		
email	email	varchar(300)		
用户状态	userstatu	varchar(300)		
exist	exist	varchar(10)		
用户等级	level	varchar(50)		

表 3.14 商品信息明细表

Name	Code	Data Type	Primary	Foreign Key
goodsID	goodsID	varchar(20)	X	
userID	userID	varchar(9)		X
物名	goodsName	varchar(300)		
物品描述	goodsDescribe	text		
物品图片	goodsPic	text		
物品分类	goodsType	varchar(300)		
交易意愿	swopWill	varchar(300)		
物品状态	goodsStatu	varchar(200)		
添加日期	addDate	varchar(300)		
exist	exist	varchar(10)		

以上分别通过字段名、代码、数据类型、主键？、外键？等说明了数据库中每一张表。以下将详细阐述表与表之间的关系以及设计思路。

3.2.3 表关系的识别

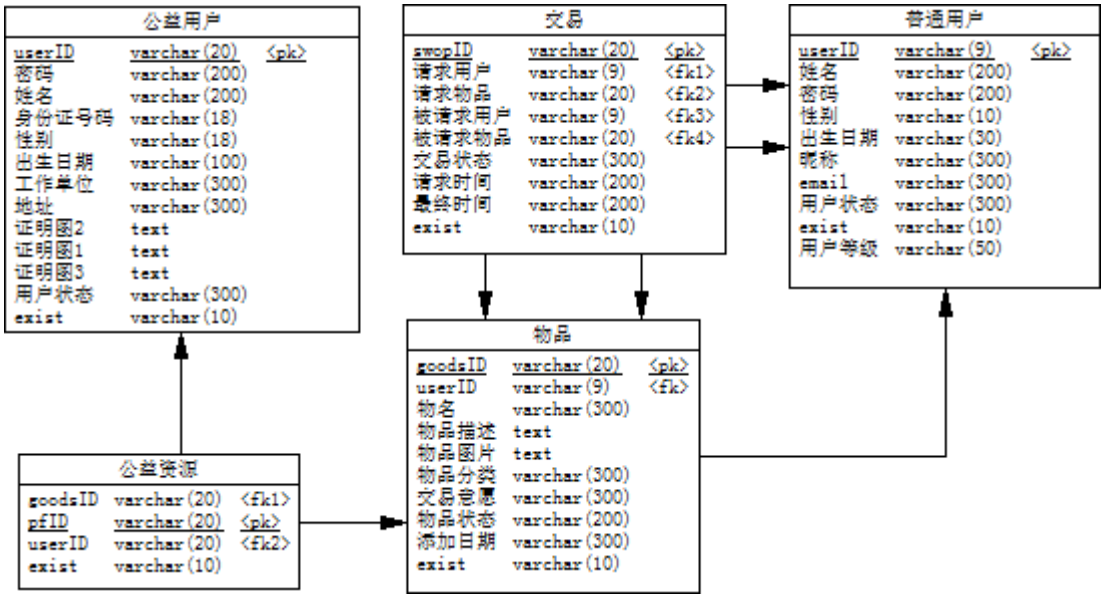

图 3. 5 数据库物理图

图上描述的表之间的关系，接下来将以从左到右从上到下的顺序说明它们之间的关系。首先是公益用户表，与公益用户表相关的表是公益资源表，由于公益用户对公益资源是 1:*的关系，公益资源对公益用户是 1:1，所以公益资源纳入公益用户的主键作为自身的外键属性。与交易表相关的表有物品表和用户表，物品表与交易的关系是 1:2，交易表与物品表的关系是 1:*，所以交易表需要两次引入物品表的主键，作为自身的被请求商品字段和请求商品字段。被引入的这两个字段都是联合作为交易表的一条外键字段。用户表同物品表，交易表与用户表的关系是 1:2，用户表与交易表的关系是 1:*，所以同样是交易表两次引入用户表的主键联合作为自身的外键。接下来是用户表，与用户表相关的表有交易表与物品表，由于与交易表的关系之前已经说过了，这里就不重复描述了。用户表与物品表的关系是 1:*，物品表与用户表的关系是 1:1，所以用户表需要引入物品表的主键作为自身的外键字段以保持数据的完整约束。接下来是公益资源表，同上不阐述重复的内容，公益资源表与物品表的关系是 1:1，物品表与公益资源的关系是 1:*的关系，所以公益资源表当引入物品表的主键作为外键约束条件。

3.3 开发语言和环境

3.3.1 后台

后台即服务器的服务程序。后台以 Java 为主要开发语言，数据库选用 MySql。这里主要

阐释一下，数据库选用 MySql 的原因。MySql 同 MS-SqlServer 一样是一种关系型数据库，两者之间，更多的中小型企业通常都喜欢使用 MySql。

其优势主要有以下几点：

1） 体积小

2） 速度快

3） 总体拥有成本低

4） 开源（源代码开放）

我选择它的主要原因是前三项，不过我使用的还是社区版（免费）。虽然是社区版，但病不影响功能的开发。

3.3.1.1 Java

Java 是一门面向对象编程语言，不仅吸收了 C++语言的各种优点，还摒弃了 C++里难以理解的多继承、指针等概念，因此 Java 语言具有功能强大和简单易用两个特征。Java 语言作为静态面向对象编程语言的代表，极好地实现了面向对象理论，允许程序员以优雅的思维方式进行复杂的编程。

Java 的三个基本特征：

1） 封装：封装就是把属于同一类事物的共性（包括属性与方法）归到一个类中，以方便使用。

2） 继承：就是个性对共性的属性与方法的接受，并加入个性特有的属性与方法。

3） 多态：多态就是在抽象的层面上实施一个统一的行为，到个体（具体）的层面上时，这个统一的行为会因为个体（具体）的形态特征而实施自己的特征行为。

3.3.1.2 框架选择

后台框架的选用主要考虑数据处理的便捷性、安全性、以及其效率。因此数据关于数据库操作部分的框架选择 MyBatis。MyBatis 的前身叫 iBatis，本是 apache 的一个开源项目，2010 年这个项目由 apache software foundation 迁移到了 google code，并且改名为 MyBatis。MyBatis 是支持普通 SQL 查询，存储过程和高级映射的优秀持久层框架。MyBatis 消除了几乎所有的 JDBC 代码和参数的手工设置以及结果集的检索。MyBatis 使用简单的 XML 或注解用于配置和原始映射，将接口和 Java 的 POJOs（Plan Old Java Objects，普通的 Java 对象）映射成数据库中的记录。选用该框架的主要原因是，该框架操作简单，对数据的控制

比较灵活，大部分数据的操作都是同 Sql 语句完成，即使功能变动比较大，也不要考虑更改框架。

关于后台逻辑处理的与前端交互的框架，我打算选用 SpringBoot。暂时还未曾接触过该框架。如果后期实现时真的无法适应的话，我会选用 Struts2。相对而言我比较熟悉 Struts2。虽然是一个比较老的框架，但是大部分后台框架原理实现类似。都是通过依赖注入、控制反转的概念提高工作效率。以下简单介绍以下 Springboots 与 Struts2。

Spring Boot 是由 Pivotal 团队提供的全新框架，其设计目的是用来简化新 Spring 应用的初始搭建以及开发过程。 该框架使用了特定的方式(继承 starter，约定优先于配置)来进行配置，从而使开发人员不再需要定义样板化的配置。通过这种方式，Boot 致力于在蓬勃发展的快速应用开发领域（rapid application development）成为领导者。 Spring Boot 并不是一个框架，从根本上将，它就是一些库的集合，maven 或者 gradle 项目导入相应依赖即可使用 Spring Boot，而且无需自行管理这些库的版本。

Struts2 是 java web 的框架，在 Java Web 开发中，表示层框架，其核心是通过扩展 Servlet 来帮助处理 http 请求。Struct2 的框架由 3 个部分组成：核心控制器 FilterDispatcher、业务控制器、和用户实现的业务逻辑组件，其基本流程为：FilterDispatcher->Action->业务逻辑组件。核心控制器负责拦截所有的用户请求，当请求是*.action 结尾会被转入 Struts2 框架处理，Struts2 再决定调用哪个业务逻辑组件。业务控制器就是实现 Action 类的实例，Action 类通常包含一个 execute 方法（也可在配置文件中指定方法执行），该方法返回一个逻辑视图名的字符串。

3.3.2 前端

前端开发就是客户端的开发，即浏览器端的应用开发。

3.3.2.1 语言选用

前端语言无非就是：html、css、javascript、jQuery;在前端开发中我所使用到的也就是这几种语言。关于这几种语言我就不一一介绍了。全部都是以浏览为主的解释性脚本语言，整个工作过程就是浏览器从服务器获得网页数据，即由以上几种语言组成的应用程序，获得这些数据后，浏览器就开始一行一行翻译，就像画画一样在浏览器面板中画出或者说是解释出网页的样子。

3.3.2.2 Bootstrap

Bootstrap 是一种前端框架，使用它可以大大提高前端的开发效率，其中集成了很多前端样式、插件，而且都很美观。以下就简单介绍以下 Bootstrap。

BootStrap 是基于 HTML、CSS 和 JavaScript 的框架，使你只需要写简单的代码就可以很快的搭建一个还不错的前端框架，他是后端程序员的福音，使他们只需要专注业务逻辑，而无须浪费太多的精力在界面设计上。它可以开发*全响应式网页*——不论你使用手机、平板电脑、普通个人电脑浏览网站内容，所有的元素都可以很优雅的呈现。所以，可以用他来开发适合各种设备浏览的页面，避免了大量的因为兼容性而导致的重复劳动。

3.4 部署图

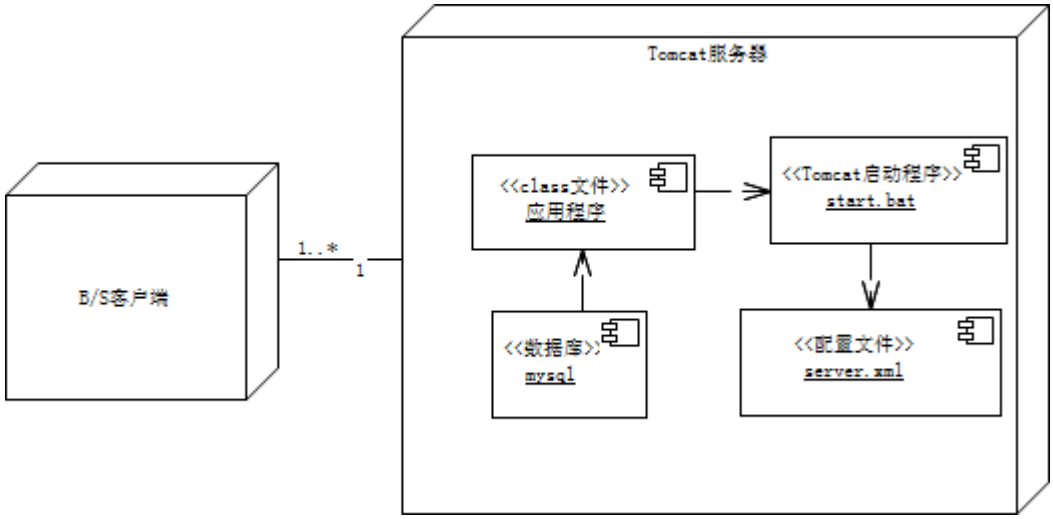

图 3. 6 构件部署图

图上描述的是构件部署图。原本 Tomcat 服务器所在节点应当是带有阴影的，但是所使用的作图软件没有对应的模型。图上主要有两个节点组成，Tomcat 服务器和 B/S 客户端，其中 Tomcat 服务器是处理器，B/S 客户端是用户的操作设备。在处理器节点 Tomcat 服务器中是构件图，较为详细的描述了，系统的部署情况。首先说明服务器操作系统是 WindowsR2，Tomcat 版本是 1.8。由于个人条件有限所以数据库和 Tomcat 服务器在同一台处理器上。构件图中有开发好的 web 应用程序，数据库（Mysql），Tomcat 配置文件，和 Tomcat 启动脚本。它们之间关系是，数据库依赖 web 应用程序，web 应用程序依赖 Tomcat 启动程序，Tomcat 启动脚本依赖 Tomcat 配置文件。所以在部署文件的时候，只需要在配置文件中稍作修改，再启动脚本文件，服务就会启动了。

与之相对应的是 B/S 客户端节点，由于是 b/s 项目，所以只要用户拥有一个浏览器就可

以接受到服务器的服务。客户端和服务器之间的关系是多对一的关系。由于前端程序的要求限制，浏览器必须使用谷歌或者火狐才能保证用户在使用的过程中不会碰到不必要的困扰。

第四章 系统测试

4.1 系统实现

1. 系统整体外观实现

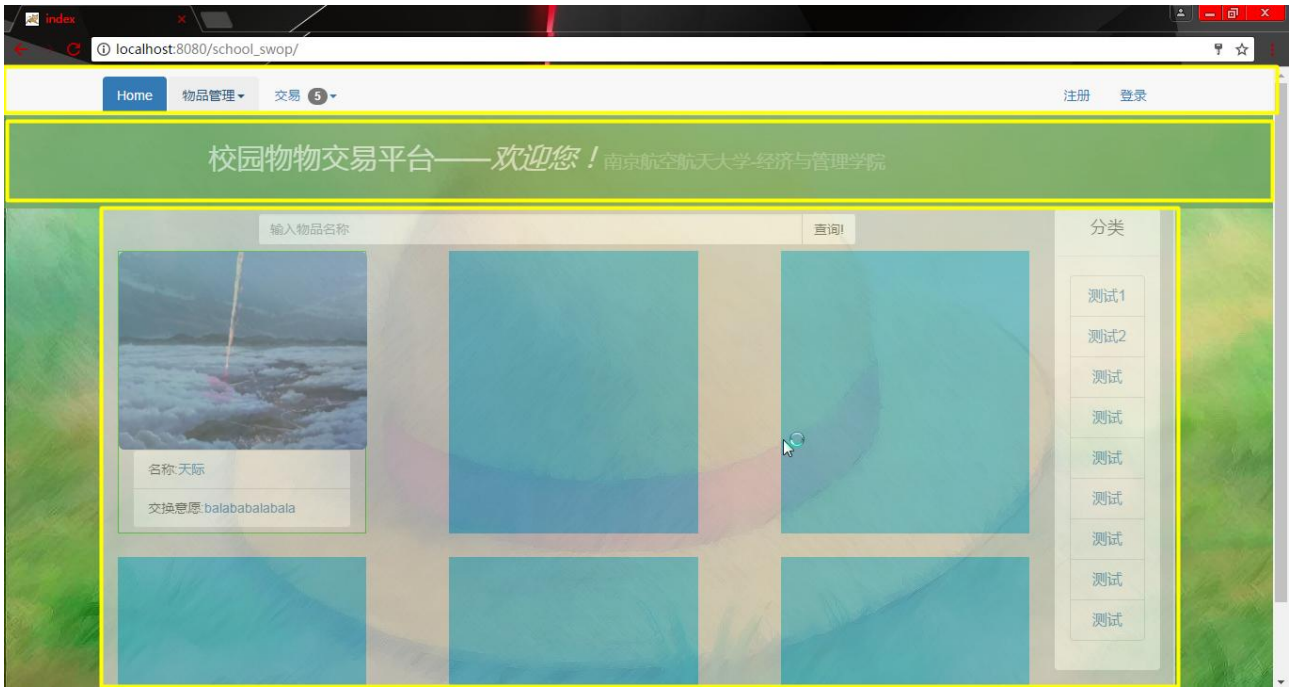

图 4. 1 客户端首页-实现

图上描述的是系统客户端(浏览器)设计的整体界面。主界面主要由三个部分构成：第一部分就是图上从上至下顺序第一个黄色包围的框，系统的导航栏（导航栏包括主页、物品管理（物品管理包括新增物品、编辑物品）、交易（交易包括交易记录、收到的请求、发出的请求））；第二部分是系统标题栏“校园物物交易平台——欢迎您！”；第三部分是系统功能主要实现的区域（是一个嵌套的网页通过 Ajax 嵌套），当前显示的是所有物品（空白的蓝色框图是伪装的物品）。

2. 注册

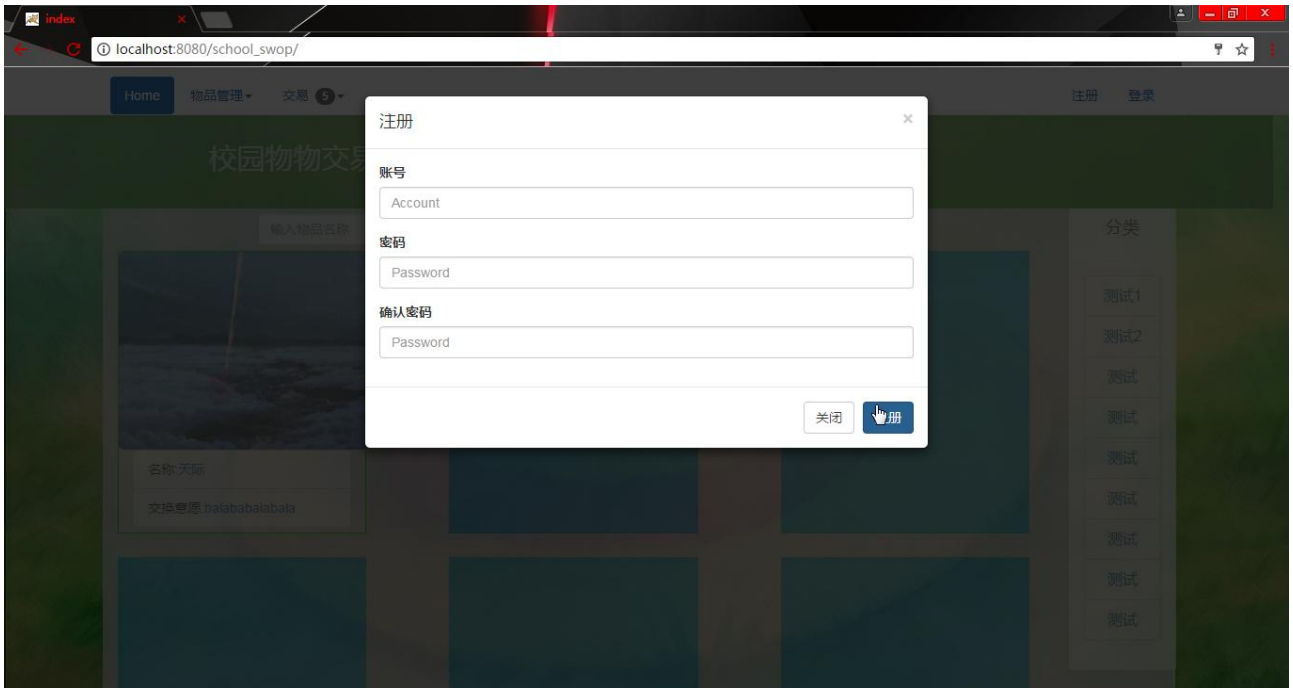

图 4. 2 注册-实现

图上描述的是用户注册实现，用户在当前界面提交注册信息。

3. 登录

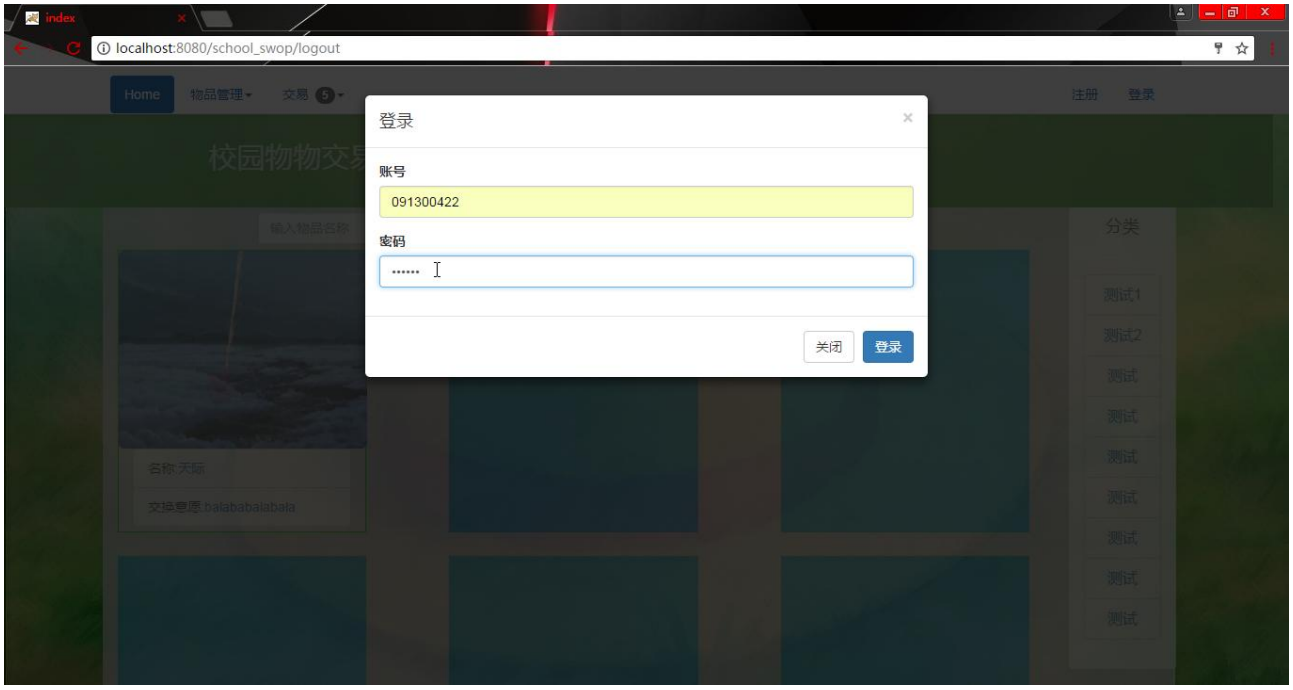

图 4. 3 登录-实现

图上描述的是用户登录实现，用户可以在当前界面登入系统。实现原理——Bootstra 模态框。

4. 用户添加物品实现

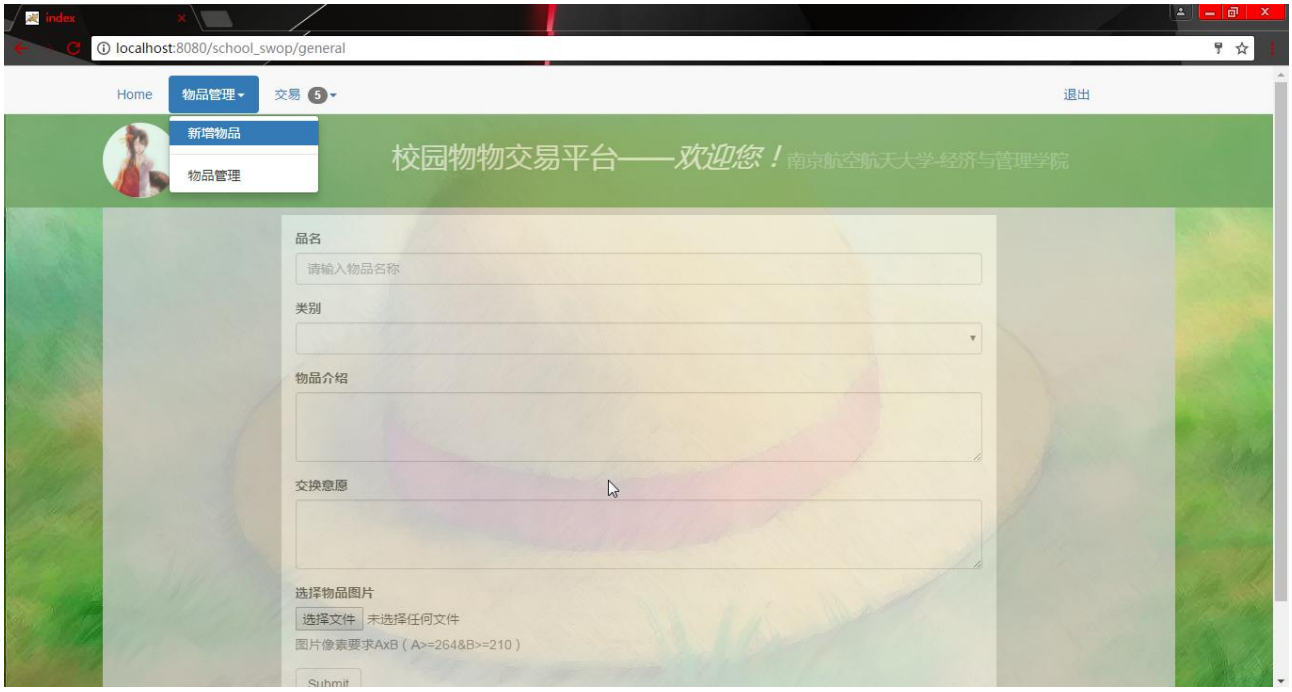

图 4. 4 用户添加物品-实现

图上描述的是用户添加物品的实现。用户在当前界面完善物品信息后便可以提交审核。

5. 用户编辑或删除物品

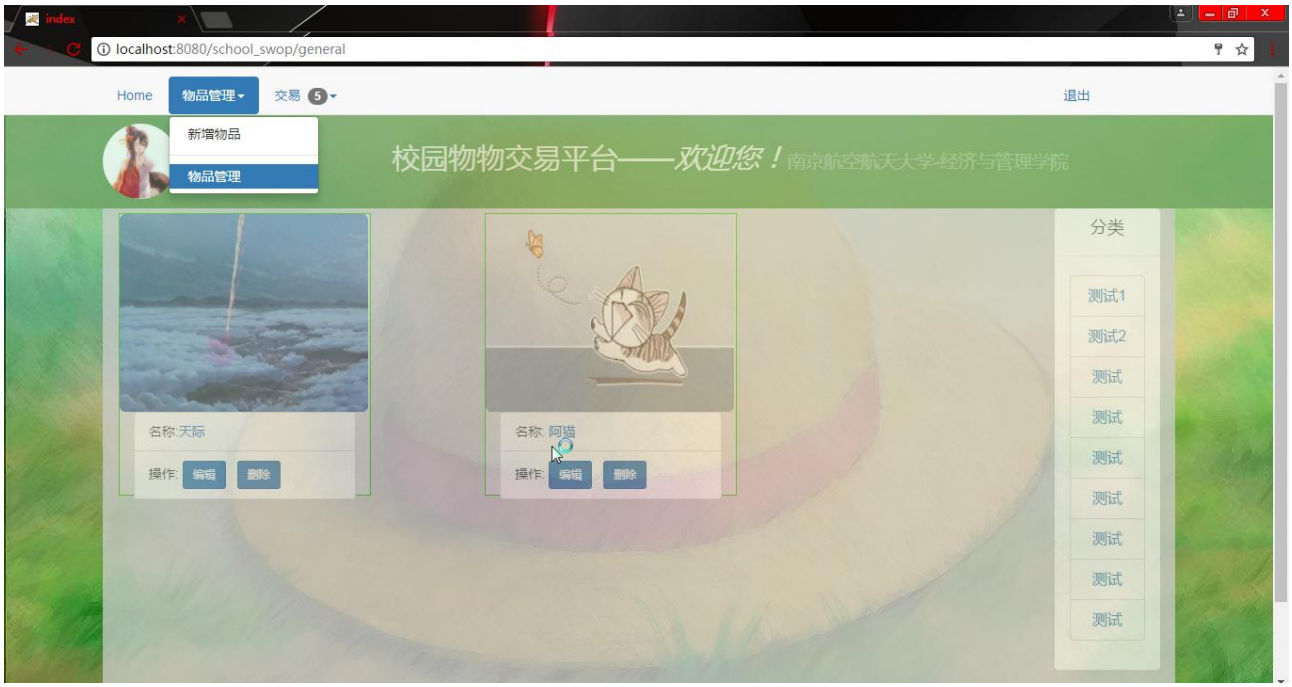

图 4. 5 物品管理-实现

5.1 编辑物品

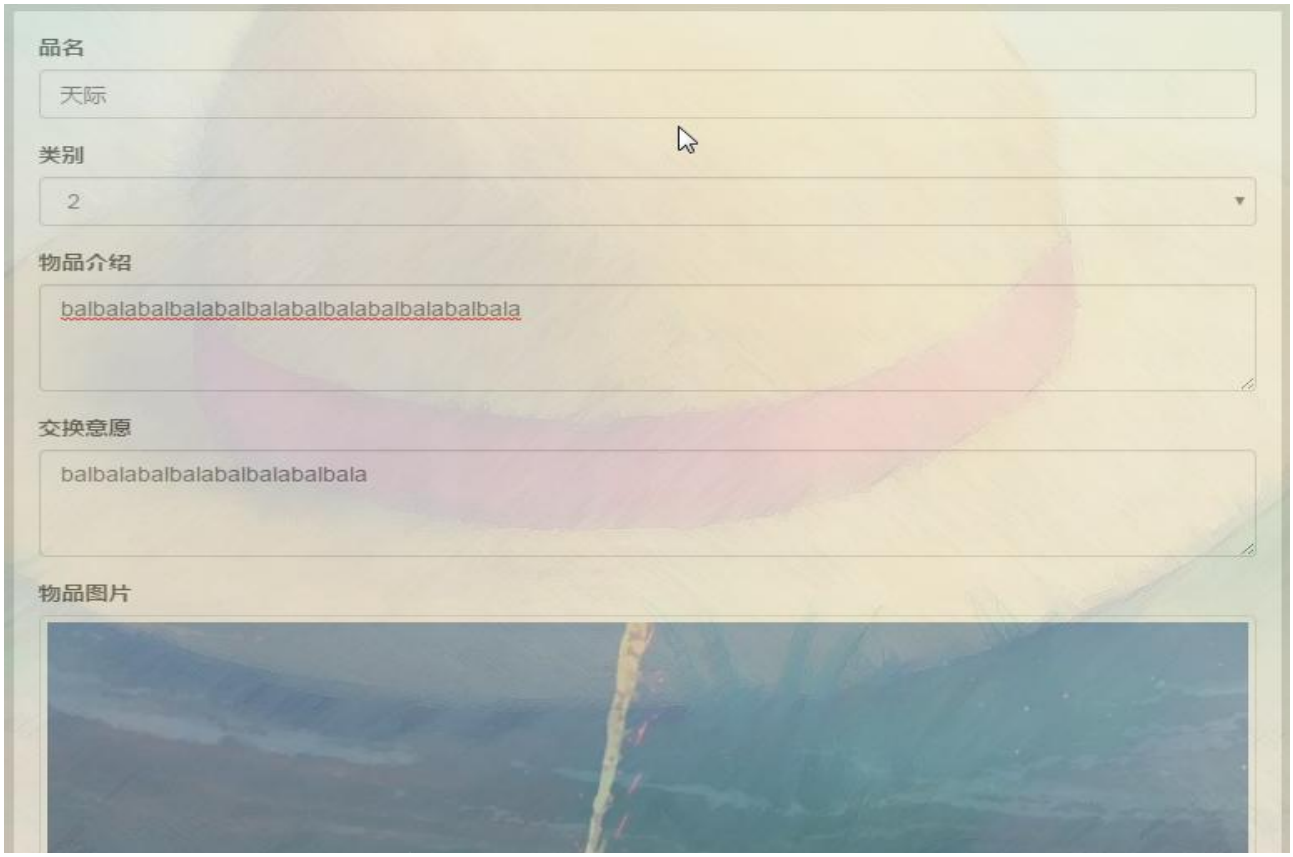

图 4. 6 编辑物品-实现

5.2 删除物品

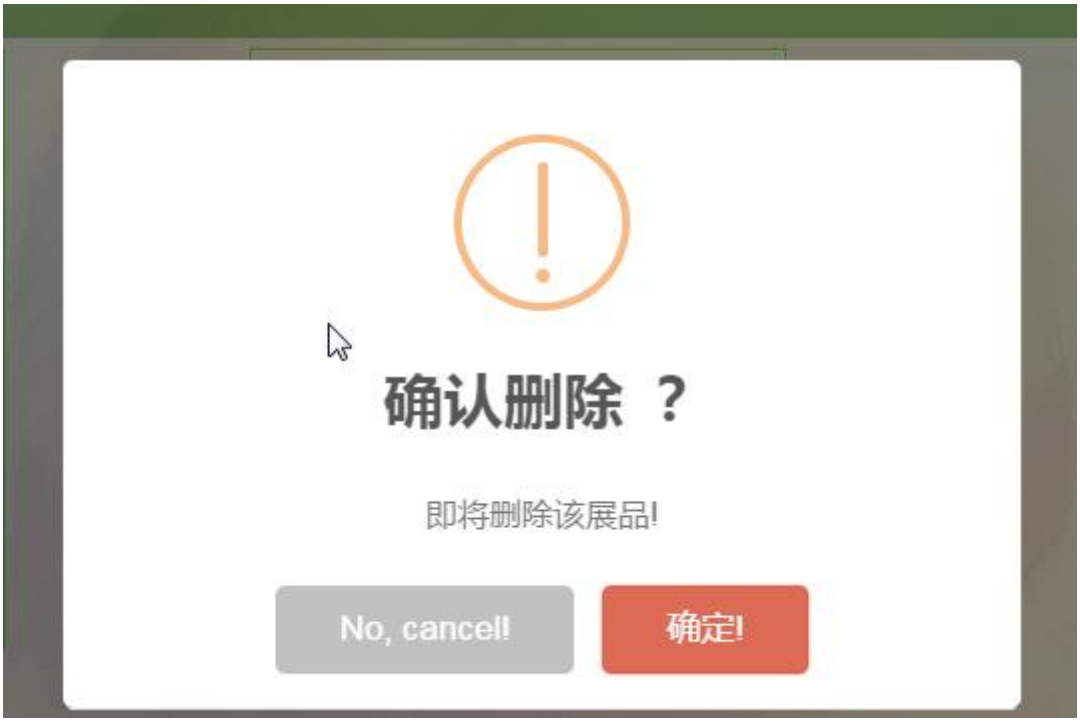

图 4. 7 删除物品-实现

5. 交易管理实现

图 4. 8 交易管理-实现

图上描述的交易管理的实现。在导航栏中交易下拉框选项旁附有数字表示的是待处理交易请求数总和，用户可访问下拉菜单中任意带数字项获取相应的交易详情。用户还可以根据此下拉菜单查询历史交易记录。

4.2 系统登录用例测试

当前用例采用白盒测试。

1. 前端请求登录代码片段

```
1.  $('#login').click(function() {
2.          if($('#logid').get(0).value.trim()!=""&$('#logpwd').get(0).value.trim()!=""){
3.              $.post("userlogin_login",
4.                      {"user.userId":$('#logid').val().trim(),
5.                       "user.password":$('#logpwd').val().trim()},
6.                      function(data,status){
7.                           if(status=="success"){
8.                                  var strs = data.trim().split(",");
9.                                  if(strs[0].trim()=="1"){
10.                                      window.location.replace(strs[1]);
11.                                  }else{
12.                                      $('#logpwd').prop("value","");
```

```
                                    swal("密码或账号错误","请重新登录！","warning");
                                }
                            }else{
                                swal("请求错误","请重新登录！","error");
                            }  });
        }else{
            $('#logid').prop("value","");
            $('#logpwd').prop("value","");
            alert("完善登录信息先~");
        }  });
```

2. 后台处理登录请求代码片段

```
public void login() throws IOException{
    Guser users = guserdao.loginCheck(user.getUserId(), user.getPassword());
    String results = "general";
    String userLevel = "GU";
    if(users!=null){
        session.put("user", users);
        results = "1".equals(users.getUserstatu().trim())?"general":("3".equals(users.getUser
statu().trim())?"publicfare":"admin");
        userLevel = "1".equals(users.getUserstatu().trim())?"GU":("3".equals(users.getUsersta
tu().trim())?"PU":"AU");
        session.put("userLevel", userLevel);
        ServletActionContext.getResponse().setCharacterEncoding("utf-8");
        ServletActionContext.getResponse().getWriter().write("1,"+results);
    }else{
        ServletActionContext.getResponse().getWriter().write("0,0");
    }
}
```

3. struts.xml配置文件配置代码片段

```
<package name="default" extends="struts-default" namespace="/">
    <action name="userlogin_*" class="com.school_swop.action.Useraction" method="{1}">
    </action>
    <action name="general"><result>/index.jsp</result></action>
    <action name="publicfare"><result>/index.jsp</result></action>
    <action name="admin"><result>/index.jsp</result></action>
</package>
```

当用户提交登录信息时，前端通过jQuery封装的Ajax的post方法提交数据。url=

“userlogin_login”，提交的数据：user.userId: 091300422, user.password:123456。首先struts拦截器拦截到请求“userlogin_login”然后交给用户程序，再解析url，执行用户程序login方法。Login方法根据请求的数据，实例一个用户对象user，然后再查询数据库中是否存在当前对象user。如果存在返回完整的当前user对象，如果没有返回null。进一步，程序判断返回的用户对象，如果存在当前用户，则将当前用户信息存入HttpSession对象，这样在服务器内用户登录信息可以保存>=20分钟。然后将当前用户等级也存入session以便后续功能的实现。最后通过httpServletRepsponse向前端发送消息“1,[general/publicfare/admin]”当前应当发送给前端的消息是“1,general”（当前用户Id:091300422存在并是普通用户所以results==”general”）。假如，user==null，则向前端发送“0,0”。

服务器端程序执行完毕。客户端收到服务器的返回值字符串data==”1,general”,data通过“,”被转换成字符串数组strs=={”1”,”general”}。程序判断strs[0]==”1”成立，于是就重新定位页面url=strs[1]，客户端请求访问strs[1](当前返回的值是general)；如果strs[0]!=”1”,就是登录失败，用户名或密码错误。Struts拦截器根据请求url返回页面。登录结束。

1. 登录前

图 4. 9 登录前-登录测试

2. 登陆错误

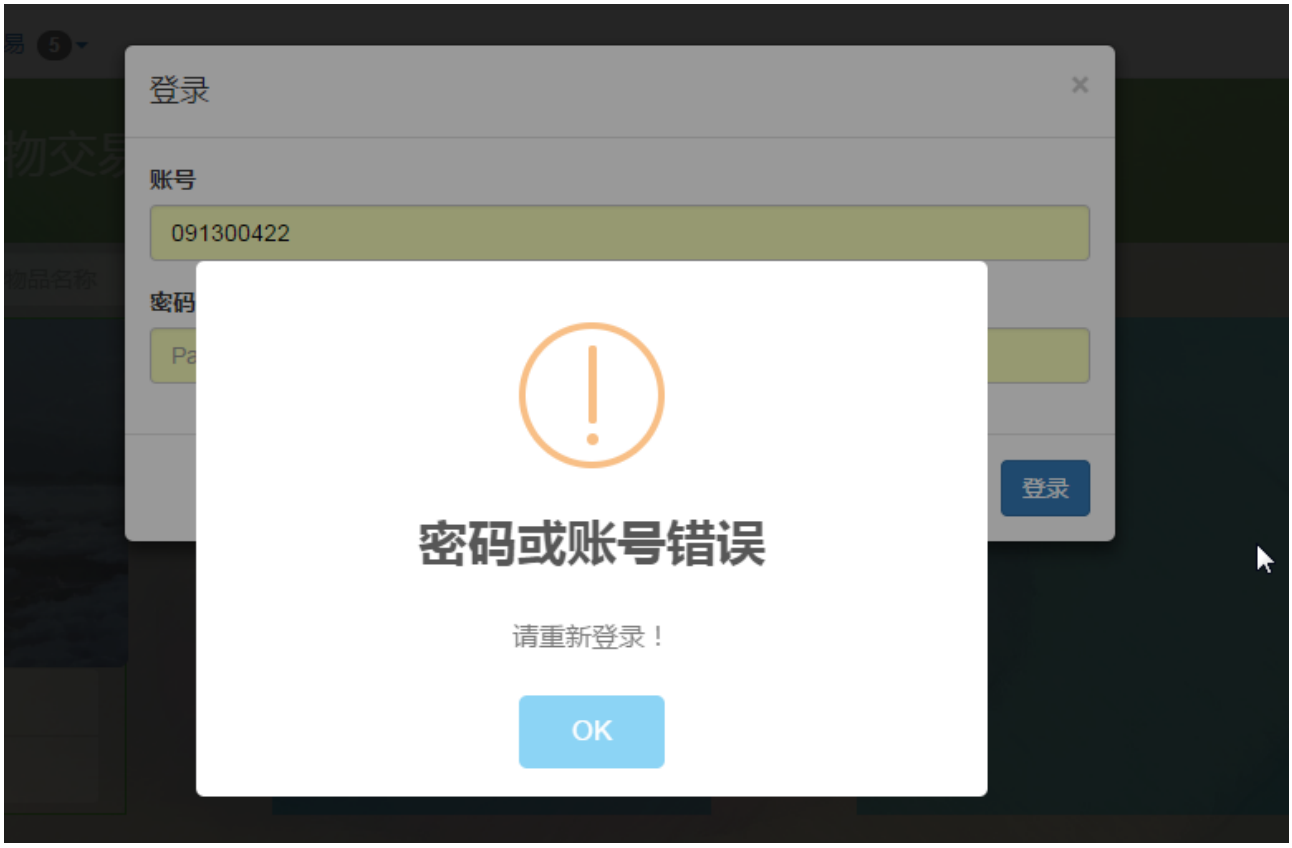

图 4. 10 登录错误-登录测试

3. 登陆后

图 4. 11 登录后-登录测试

4.3 系统登录拦截器用例测试

当前用例同采用白盒测试。

1. 拦截器代码

```
@Override
    public String intercept(ActionInvocation ai) throws Exception {
        String actionName = ai.getProxy().getActionName();
        Map<String, Object>session = ai.getInvocationContext().getSession();
```

```
6.          Object user = session.get("user");
7.          String userLevel = (String)session.get("userLevel");
8.          String str  = "relogin";
9.          if("userlogin_login".equals(actionName)||"logout".equals(actionName)||(user!=null&&ac
    tionName.substring(0, 2).equals(userLevel))){
10.             str = ai.invoke();
11.         }
12.         else{
13.             HttpServletRequest req = ServletActionContext.getRequest();
14.             HttpSession sessions = req.getSession();
15.             sessions.invalidate();
16.             req.setAttribute("loginstatus","登陆已失效，请重新登陆！");
17.         }
18.         return str;
19.     }
```

2. 配置文件代码

```
1.  <package name="goodsMana" extends="struts-default" namespace="/">
2.          <interceptors>
3.              <interceptor name="logincheckeInterceptor" class="com.school_swop.util.LoginInter
    ceptor"></interceptor>
4.              <interceptor-stack name="myDefault">
5.                  <interceptor-ref name="logincheckeInterceptor"></interceptor-ref>
6.                  <interceptor-ref name="defaultStack"></interceptor-ref>
7.              </interceptor-stack>
8.          </interceptors>
9.          <default-interceptor-ref name="myDefault"></default-interceptor-ref>
10.         <global-results>
11.             <result name="relogin">/index.jsp</result>
12.         </global-results>
13.         <action name="goodsmana_*" class="com.school_swop.action.Goodsmanageaction" method="{
    1}">
14.             <result>/index.jsp</result>
15.         </action>
16.     </package>
```

当前拦截器可拦截所有客户端请求并过滤请求是否符合本身约定，拦截器拦截一个请求后首先获取该请求的 action_name，再从 session 中提取是否有当前用户登录的信息记录，再获取当前用户的等级，如果当前请求操作不符合当前用户的身份，或者 session 中不存在

当前用户登录信息，或者当前请求不是登录，都会被拦截并清空当前用户 session 中存储的所有信息，并返回系统首页页面——提醒用户登录已失效。

1. 登录拦截

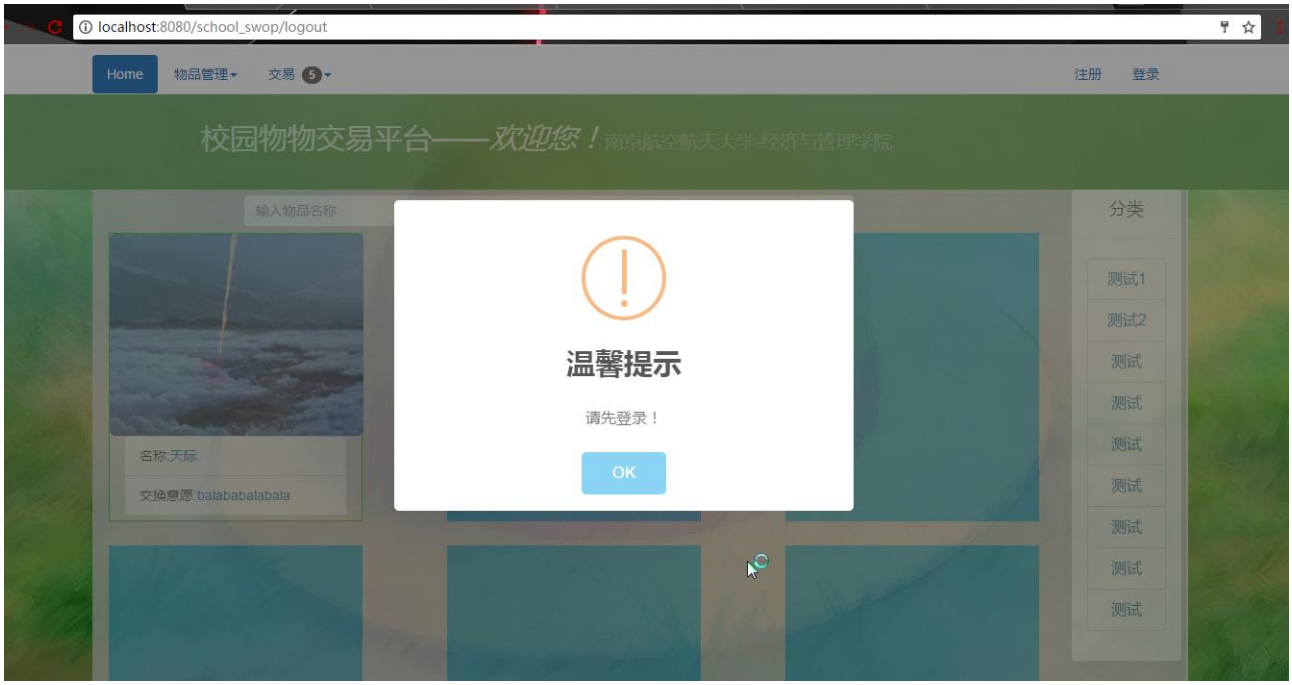

图 4. 12 登录拦截-拦截器测试

图上描述的是关于用户未登录时请求获取物品详细信息，之后被登录拦截器所拦截并返回提示消息“登陆后才能进一步操作”。

第五章　总结与展望

校园易物系统的分析与设计主要分两个阶段,系统的分析与系统的设计。系统的分析又分为需求分析、业务分析、软件分析三个部分，系统的设计又分为类模型设计、数据库设计、软硬件部署设计三个部分。当系统完成以上步骤之后，剩下的步骤就是系统的实现了。系统的分析与设计完全采用的是 UML 面向对象的分析与设计工具，使用的软件是 Powerdesigner16.5。

在系统的需求分析阶段，主要的工作内容是以源点出发分析用户的需求。需求分析是系统研发准备工作的第一步，是所有后续步骤的基础。系统的源点就是最原始最核心的系统构想——校园内以物易物的构想。在需求分析阶段使用的面向对象模型是用例模型，以用户为主体，以源点构想为思想核心，分析系统的大致的功能需求。用例图初步描绘出了系统的框架模型，还需要需要进一步分析细化。

系统分析的第二阶段业务分析，再这一阶段主要的工作内容是根据上一阶段的结果细化需求分析，细化到每一步过程在逻辑上清晰可见。在业务分析阶段使用的是活动图，也可用业务流程图代替，在本质上它们起到相同的作用——将需求用过程“实现”出来。通过活动图，开发人员可以清楚的直到如何才能达成用户的某一个需求。

系统分析的第三阶段软件分析，同上这一阶段的主要工作同是细化上一阶段的模型。软件分析阶段可以确定系统中会用到的类，以及各个类对象之间的互动。这一阶段使用的是时序图，严格要求程序在时间顺序上的执行步骤。在这一阶段是将上一阶段的活动图——逻辑流程图细化成物理流程模型，将所有步骤用类对象以及类方法进一步细化到时间执行步骤。由于程序是严格按照执行指令执行的，所以以时间为主要的要素之一。这一阶段确定了类，以及类方法。

在系统设计的类模型设计阶段，是根据之前分析的结果，作进一步设计，完成系统的主体框架。类模型设计阶段使用的是 UML 的类图，根据分析阶段的成品，补充完整类的方法、补充类的属性、明确类之间的关系。

在系统设计的数据库设计阶段，主要工作内容是设计系统所用到的数据库。根据分析阶段的结果分析出需要的表，表的字段及数据类型、以及表间关系。

在系统的部署构件图设计阶段，主要的工作内容是交代如何部署已经完成的应用程序，由于是 B/S 架构，所以系统并不需要准备客户端设备程序。

对于当前系统的展望。系统当前还在开发中，如果能够上线，我希望系统真的可以像预想的一样，起到它本该有的作用。系统分析时最初的构想就是实现校园内以物易物的功能，这样一来至少在校园内可以使得物尽其用。在一定程度上还可以减少不必要的支出。或许还会有以外的收获。此外由于时间限制，系统原本构想设计公益捐助功能并没有完整的实现设计。所以如果以后系统能够排上用场的话，它的第一个更新就是公益捐助的功能补充。这样一来，一方面可以使得物品价值得到了尽量完全的发挥，另一方面也能为那些贫困的人资助一点。如果还有余力，还可以完善系统的安全保障，由于当部分审核功能都是人工完成，效率低下，并且耗费资源大。所以可以结合大数据，将后台部分手工功能全部以自动化。如果再有余力可以完善前端界面的设计，由于前端设计中即时消息设计耗费服务器很严重，所以需要进一步优化，减轻服务器的负担。

参 考 文 献

[1] 辛娅. 优雅易物 15 年“不花钱”[J]. 新天地, 2011 年第 10 期:52.
[2] 相丽玲 吴 泽. 大学生校园以物易物平台的设计与实现[J]. 晋中学院学报, 2015, 32(3):68.
[3] 杨冬春.以物易物网络交换平台搭建背景研究［J］.电子商务，2012（7）：86～87.
[4] 王嘉琦.浅谈我国 C2C 换物网站的发展现状［J］.BSERVATION，2011（6）：15～17.
[5] 付长青 庄程. 校园电子商务一一二手交易网站 [J]. 商场现代化，2008(10)：32～33.
[6] 董勇 郎永祥 翁代云.基于面向对象技术的高校教师绩效考核系统的研究与实践[J]. 科技信息,2008(24):32.
[7] 王桂芳 宋丽花. 基于 Agent 的远程教学系统[J]. 科技情报开发与经济,2003 年 11 期:43.
[8] 尹夏璐. 市委党校图书管理系统的分析与设计[D].云南：云南大学，2013.
[9] 吴峰. 新疆师范大学教育工会管理系统设计与实现[D]. 吉林：吉林大学，2014.
[10] 姜明. 住房担保信息管理系统的设计与实现[D]. 东北：东北大学，2011.
[11] 李彤. 统一认证授权平台的研究及在电动汽车领域的应用[D]. 北京：北京邮电大学，2014.
[12] 叶长春. 基于 MVC 的 Struts 框架的应用研究[D]. 武汉：武汉理工大学，2008.
[13] 谢宗旺 方旭升. 基于 Struts2 和 Spring 框架的 Web 整合开发研究[J]. 价值工程,2011,30(7):32.
[14] 李硕. 基于 WEB 的在线学习系统[D].互联网：互联网文库,2015.
[15] 孙蓓蓓. 基于 J2EE 的网络信息资产管理系统设计与实现[D]. 西安：西安电子科技大学，2012.
[16] 李志强. 基于 Struts 框架和 Ajax 技术的汽车租赁管理系统设计与实现[D]. 郑州: 解放军信息工程大学,2008.
[17] 杨辰. 车辆尾气自动检测方法研究及检测系统开发[D]. 武汉：武汉理工大学，2013.
[18] 宋森. ASP 库存管理系统设计与实现[D]. 成都: 电子科技大,2013.
[19] 张姝. 基于技能库的软件培训管理系统的开发研究[D]. 上海：复旦大学,2008.
[20] 于波. 电能计量设备管理信息子系统的研究[D]. 河北：华北电力大学,2005.
[21] 孙春艳. 企业管理软件中用户管理的.net 组件化实现[D]. 吉林：吉林大学，2012.
[22] 魏涛. RoboCup 仿真球队的研究与实现[D]. 南京：南京理工大学，2007.
[23] 陈阿梅. 峨眉电力宾馆管理信息系统设计实现[D]. 成都：电子科技大学,2012.
[24] 杨欣. 种质资源库圃绩效考核系统的设计与构建[D]. 南京：南京理工大学，2011.
[25] 杨梅 杨建平. 基于 XML 的三层体系结构的 B2C 电子商务网站的架构与实现[J]. 电子技术与软件工程,2013(12)：11
[26] 程光磊. 基于 Flex 与 J2EE 架构的 CMS 系统的设计与实现[D]. 安徽:安徽理工大学,2012.
[27] 李晶. 基于 SNMP 的网络监测系统的研究与设计[D]. 河北：河北工程大学,2012.
[28] 杨新庆. 基于 Android 平台的无线旅游导航系统的设计与实现[D]. 吉林:吉林大学,2011.
[29] 马跃. 基于三维动画角色建模的资源库设计与应用[D]. 上海：复旦大学,2010.
[30] 张波. 基于 AutoCAD 的客船撤离计算辅助工具的设计与实现[D]. 上海: 上海交通大学,2014.

致　　谢

这次毕业设计可以圆满的完成，首先要感谢我的指导老师。他学识渊博、治学严谨，平易近人，在他的悉心指导中，我不仅学到了扎实的专业知识，也为以后的工作奠定了坚实的基础；他严谨的治学态度，一丝不苟的工作作风和诲人不倦的育人精神让我受益菲浅。在设计的整个过程中，给予我精心的指导与帮助，为我们的设计付出了辛勤的劳动，倾注了大量时间和精力，在此向他表示诚挚的敬意和衷心的感谢。

感谢南京航空航航天大学大学经济与管理学院所有曾经帮助过我的老师和同学，他们的教授与帮助，使我获得了大量的知识，圆满完成了学业，在此我深深地表示敬意和由衷的感激之情。

最后，我要感谢所有在论文撰写过程中给予我支持和帮助的老师、同学和朋友们。