# 自动售货机系统

本系统是一个售货机系统。无人售货机凭借着没有收银员、结账无需排队、全天候售货等特点，已经成为了大众关注的焦点。本系统就是这样一个方便用户生活的系统。

无人售货机本质上就是超微型便利店，它的出现，最重要的目的还是方便人们的生活。因此它的目标和定位是，结账无需排队、全天候、快速获取商品。与之对应的，技术上面采用基于事务的MySQL数据库来存储各种交易信息，使用Java语言来编写事务相关的代码，最终形成售货机系统。

## 1.研究背景

在第五次工业革命中，智能化概念正在悄无声息的出现并蔓延在人们的生活中，无人售货机正慢慢地变革着消费形式，不论是从销量还是从售卖方式上都发生了改变。无人售货机本质上就是超微型便利店。如同零售渠道发展史，从商场-大卖场-超市-便利店，零售渠道越来越小，品类越来越少，离用户越来越近，而无人售货机未来将是比便利店更小，离用户更近的超微型零售渠道。无人贩卖在未来也是一大发展方向。

## 2.市场调查

无人零售方面，出现最早的是自动售货机。在移动支付尚未普及前，货币识别类的自动售货机就已在人流较大的地铁站等公共场所出现，而支付宝、微信支付火爆之后，自动售货机开始增多，凭借没有收银员、结账无需排队、全天候售货这些特点，成为了大众关注的焦点。

如果是货币识别类的自动售货机，那么它是无法分辨正在购物的顾客是第一次购物，还是已经在此处购物过的。而使用线上支付则可以通过支付宝账号或者微信账号等进行分辨，或许可以基于此设计更智能的推荐算法，针对每个客户的往期购买记录，推荐更合适的产品；或者客户主动记录自己的偏好，实现自定义的售货机。

在已经使用的自动售货机中，大部分人会选购其中的饮品和方便面类的食品。

## 3.功能设计

### 3.1 实体

首先是货物。

货物需要具有id属性、name属性、class属性、cost属性、number属性和description属性。

其次是货物供应商。

供应商需要具有id属性、name属性、联系人属性、电话属性、地址属性和description属性等。

然后是人物。

人物需要具有id属性、userCode属性、密码、性别、生日、联系方式、地址、角色属性等。

根据角色属性不同可以分为管理员和用户。

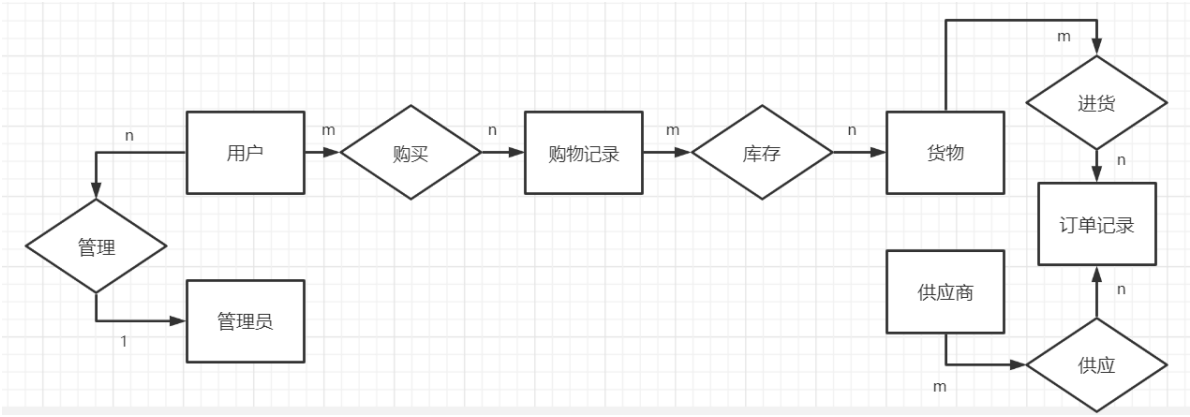
然后是购物记录。

记录需要有id属性、用户id、货物id、货物class、cost、订单号和Date属性.

然后是进货订单记录。

记录需要有id属性、订单号、货物名、货物类别、货物价格等属性。

## ER图




## 3.2 功能

**1.添加用户**

这是本系统后台的核心功能。

为了方便管理，本系统面向的是学校内 / 小区内的用户，用户需要提前将信息录入系统（通过管理员），才能够使用本系统。这样，管理员便可以对用户进行筛选。同时，用户的部分信息被记录下来，如果进行数据分析或者进行更多功能的开发时，更为方便。

将用户信息录入系统分为两种方式，一种是通过.CSV文件导入批量的用户，另一种则是通过在线编辑的方式。

通过CSV文件导入用户的方式，适合于前期系统即将投入使用时，批量导入用户。可以通过线上请用户填写问卷或者线下到营业厅统计信息等方式，将绝大部分潜在用户提前录入系统，方便用户使用。

通过在线编辑的方式，适合于系统已经投入使用时，导入个别新用户。新用户需要联系系统管理员来将信息录入系统，然后使用系统。

管理员登录系统，可以进入系统后台，进行添加用户 / 通过CSV文件批量导入用户的操作，将新用户加入系统，便于保存他的信息。加入系统的用户才可以进行登录、购物等操作。

总结一下：

在设计之初，系统需要确定面向哪些对象进行使用。例如，如果将系统投放在校园内进行使用，要求只有在本校内的学生和老师才能使用；如果将系统投放在学员队内进行使用，要求只有本队的人员才能使用，以此来保证对学员队的供货能够充足，满足学员的需求，提高学员们的使用体验。

基于此设计，在系统前期即将投入使用时，需要批量导入可使用该系统的用户。因此，本系统实现了基于CSV文件的导入大批量用户的功能。可以通过线上请用户填写问卷或者线下到营业厅统计信息等方式，将绝大部分潜在用户提前录入系统，方便系统投放后用户的使用。系统已经投入使用时，再导入个别新用户则不需要通过此功能。

本功能会在后台用户管理页面提供一个按钮以供管理员使用，生成.csv文件后，管理员点击“批量导入用户”按钮，即可将.csv文件中的所有数据统一导入数据库中。即管理员只需要将.csv文件准备好即可。

**2.修改用户信息**

用户的信息可能录入错了，或者联系方式 / 地址等改变了，那么就需要对用户的信息进行修改。后台管理系统不能缺少修改用户信息的功能。

这个功能的实现，是基于事务型关系数据库的update操作的，不能保留用户留下的“历史记录”，可以尝试进行修改。

**3.删除用户**

用户可能不需要在本系统进行购物了，那么就需要将其数据进行删除。后台管理系统不能缺少删除用户的功能。

**4.对部分用户信息进行可视化**

- 查看用户列表
    - 默认查看方式是将所有用户进行显示，使用分页。
    - 可以在顶部搜索框进行搜索
        - 对用户的姓名进行匹配
    - 可以区分管理人员和客户
        - 通过userCode属性进行区分
- 可视化

  点击顶部导航栏的Charts，进入可视化界面。

  后台管理员可以对用户登记在系统的信息进行可视化操作。例如，管理员可以对用户的性别进行可视化，分析用户的性别组成；也可以对用户数量的变化进行可视化，分析用户量在系统投入使用的增速。

  后台管理员可以对用户在系统进行购物后，生成的订单记录进行可视化，分析日订单数目的变化、总订单数目的增速等。

  用户也可以对自己在此系统的消费记录进行可视化操作。例如，某用户可以对自己的购物数目进行可视化。

  这些功能的实现，需要借助JFreeChart和Echarts。并且在后台管理页面中提供了可点击的按钮，直接让管理员进行使用，非常方便。

    - 分析用户的性别组成
        - 录入用户信息时已经提到，会记录性别等信息，可以借此进行数据分析，绘制饼图
    - 分析订单数的变化
        - 可以分析自系统投放使用以来，总订单数的变化，绘制折线图
    - 分析日销量的变化
        - 绘制时序图

**5.购物**

这是系统前台的核心功能。

用户可以在登录后，进入系统前台，根据自己的偏好进行商品的选购。例如，商品是按照种类进行划分的，用户可以通过对不同种类商品的选择，进行有目的性地选购。或许可以基于用户的多次筛选进行商品的展示。用户选择了想要购买的商品后，可以提交订单，结束本次购物。购物记录会存入数据库，管理员有权限查看。

在使用一般售货机时，面对繁多的货品，一时间用户可能忘记了自己需要购买的货品，或是一时半会儿找不到自己需要的货品。因此，为了提高用户的使用体验、方便用户对需要的物品进行查找，系统投入使用时，提供了货品筛选功能。

首先是基于货品的种类进行筛选，实现了分类购物的功能，用户可以缩小自己需要查找的范围，节省了大量的搜索时间；其次是基于货品名的精确筛选，采取了模糊匹配，待用户输入关键词后，将与之匹配的货品全部搜索出，让用户能够及时找到自己需要的货品。

本功能会在前台用户使用界面提供一个导航栏给用户，方便其基于货品的种类进行筛选；也会在货品展示的上方显示一个搜索框，让用户进行基于关键字的搜索。

# 4.页面设计

## 用户选购页面


用户进入欢迎页面后，中间的商品展示区域会展示默认的商品或需要促销的商品，左侧侧边栏可以让用户做出选择，选择自己的偏好。

用户选择一个类别之后，可以再根据自己的偏好进行二次筛选，例如，饮品类，想要购买水还是饮料，食品类，对口味有什么要求等等。需要一个物品描述属性，实现此功能。

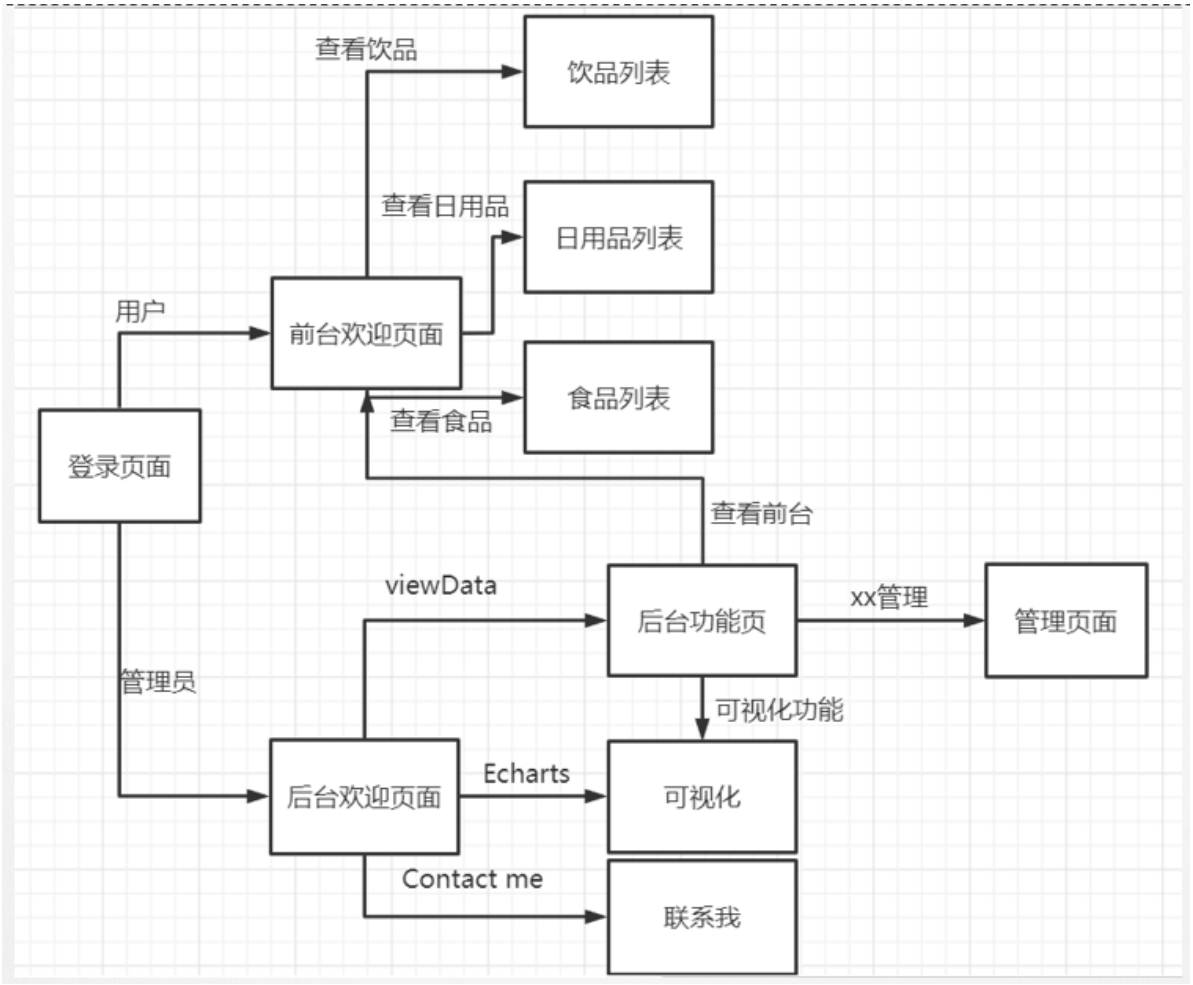
## 系统登录页面

为了方便管理，增加了系统管理员的角色，管理员可以登录到系统的后台。点击登录后，用户名和密码会传入后台程序，程序对用户名和密码进行验证，如果与数据库中的内容匹配，则进入管理界面。只有管理员可以登录到系统后台，其他用户只能进入前台界面。

## 管理员主界面


点击Home，可以进入用户选购界面，测试面向用户的功能。点击Data，管理员（开发者）可以查看各种商品的库存情况，左侧侧边栏提供选择商品种类的功能，即根据物品类别筛选出物品，将其展示在右侧。点击Order，可以查看订单，同样可以加入对订单的筛选功能，即在左侧侧边栏提供选择商品种类的控件，点击后，在右侧可以展示出对应的订单。

## 页面流转图




# 5.技术应用

## 1.基于Maven的外部依赖管理

在JavaWeb开发中，需要使用大量的jar包，手动导入过于麻烦，且有的jar包之间会有依赖，只导入一个，同样无法运行。

因此本系统使用了基于Maven的外部依赖管理，导入jar包时，只需要在.pom文件中进行配置即可。而且，导入一个jar包后，MAVEN会自动导入其相关的依赖，极其方便。

IDEA内置了Maven插件，可以直接使用其进行开发，也可以在官网下载后，将jar包下载到本地的maven-repo进行开发。

## 2.基于DBCP数据源的数据库连接池

在获取数据库连接时，封装了基于DBCP数据源的数据库连接池技术。

DBCP数据库连接池是 apache 上的一个Java连接池项目。

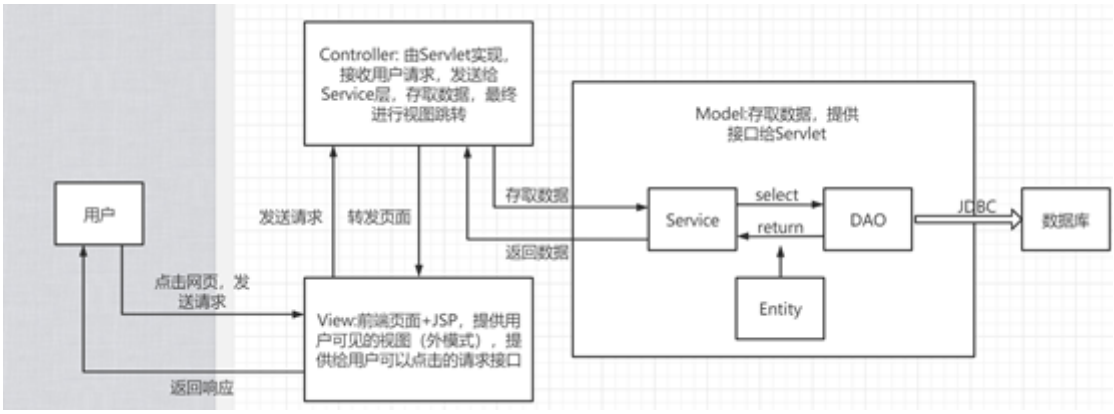
通过连接池预先同数据库建立一些连接放在内存中(即连接池中)，应用程序需要建立数据库连接时直接到从接池中申请一个连接使用，用完后由连接池回收该连接，从而达到连接复用，减少资源消耗的目的，使系统更加稳定。

具体开发时，导入jar包就用到了Maven的包管理技术。

```
<!-- https://mvnrepository.com/artifact/commons-dbcp/commons-dbcp -->
 <dependency>
   <groupId>commons-dbcp</groupId>
   <artifactId>commons-dbcp</artifactId>
   <version>1.4</version>
 </dependency>
```

使用DBCP连接池时，需要配置dbcpconfig.properties文件，在DbUtil.java工具类中进行导入并封装进getConnection方法中。

### 3.基于MVC框架的后端实现技术




Model是业务模型，完成了存取数据、提供接口给Servlet层的作用。具体实现中，DAO部分负责通过JDBC与数据库进行交互，每个数据库表都对应一个实体类，DAO通过DHCP数据库连接池获取连接后，使用嵌入式sql语句读取数据库中的数据，将实体类返回。Service部分负责接收DAO返回的实体类，并根据自己需要的服务，提供接口给上层，以供上层进行调用。

View是用户界面，即用户可见的外模式。前端页面是由JSP+HTML+CSS+JS进行编写的，页面中会提供给便于用户操作的可点击的按钮，这些按钮绑定了对应的请求，可以发送给控制层，控制层再调用业务层进行处理，最终返回给用户响应，使得用户获得自己需要的页面。

Controller是控制层，主要负责将用户请求与业务模型进行对接。这一层通过编写对应的Servlet进行实现。每个Servlet都是继承自Java的HttpServlet类，只需重写doGet和doPost函数即可。用户提交的请求会作为参数HttpServletRequest传入doGet或doPost函数，业务模型返回的响应会作为参数HttpServletResponse传回，最终进行页面重定向或转发后，用户所得的新页面即是所请求的页面，一个功能也就此实现。

## 6.开发中的问题

### 1.

修改了css样式 / js 文件，前端无变化！

这是**浏览器的自动缓存机制导致的**！

要ctrl F5，删掉所有本地临时文件的缓存，全部从服务器端下载，才可以！

### 2.

js文件的中文不能正常显示

设置utf-8编码+保存文件

### 3.配置Servlet注解

@WebServlet("/servlet/4")

@WebServlet("servlet/4")会导致服务器应用deploy错误