|  |  |  |
| --- | --- | --- |
| 南农大  **数据库课程设计报告**  XH2 | | |
|  | 题 目: | 使用JAVA语言搭建购物管理系统 |
|  | 组 长: | 杨亦凡 学号:19421101 |
|  | 组 员 | 冯晨 学号:19421112 |
|  |  | 金科屹 学号:19421105  韩旭林 学号:19421117 |
|  | 学 院: | 人工智能学院 |
|  | 专 业: | 数据科学与大数据技术 |
|  | 班 级: | 数据211 |
|  | 指导教师: | 黄芬 职称: 副教授 |
| 2023年 5月 21日 | | |

目录

**1开发技术、开发工具及技术路线**4

1.1开发技术及开发工具4

**1.1.1开发技术**4

1**.1.2开发工具**4

1.2技术路线5

**2功能分析与设计**7

2.1需求分析7

2**.1.1用户调研**7

2**.1.2初步设计**7

2.2系统设计7

2**.2.1系统流程设计**7

2**.2.2系统功能分析**8

2.3数据库概念结构设计10

2.4数据库逻辑结构设计11

2**.4.1 E-R图向关系模型的转换**11

2**.4.2利用数据依赖的理论对关系模式进行分析利用外码对关系模式进行分析**12

2.5数据库物理结构设计13

2**.5.1确定存储结构**13

2**.5.2评价物理结构**14

2.6模块划分15

**3实现**15

3.1前端界面实现15

3**.1.1最初界面设计**16

3**.1.2客户界面设计**16

3**.1.3供应商界面设计**23

3**.1.4运货商界面设计**24

3.2后端代码实现25

3**.2.1 Java与SQL Server的连接和交互**25

3**.2.2部分功能代码**25

3**.2.3 SQL Server建立数据库**34

**4系统测试与应用**40

4.1系统测试运行环境40

4.2数据库测试与运行40

4**.2.1用户注册**40

4**.2.2用户登录**40

4**.2.3用户搜索商品**40

4**.2.4用户查看商品信息**41

4**.2.5用户添加商品到购物车并查看**41

**4.2.6用户删除购物车商品**41

**4.2.7用户查看个人信息**42

**4.2.8用户查看历史订单**42

**4.2.9用户登出系统**43

**4.2.10供应商接受新订单**43

**4.2.11供应商设置发货日期**44

**4.2.12供应商设置中止量**44

**4.2.13供应商信息显示**44

**4.2.14运货商未完成订单**45

**4.2.15运货商确认签收**45

**4.2.16运货商已完成订单**45

**5总结与展望**47

5.1总结47

5.2不足48

5.3展望48

客户、运货商及供应商的商品订购管理系统

**1 开发技术、开发工具及技术路线**

1.1 开发技术及开发工具

1.1.1 开发技术

本平台是基于java开发的，本次开发的Java版本为JDK17，数据库应用为Microsoft SQL server 19，后台编程采用eclipse。

（1）java

java是一个现代通用的、面向对象的编程语言，它是由Sun微系统公司开发的。它是由James Gosling和他的团队开发的，原名为Oak，经历改良后，后改为Java。

Java是通用的术语，是Java软件及其组件的共同体，包括Java运行时的环境JRE、Java虚拟机局面JVM以及插件API组成。Java 应用编程接口为Java应用提供了一个独立于操作系统的[标准接口](https://baike.baidu.com/item/%E6%A0%87%E5%87%86%E6%8E%A5%E5%8F%A3/7681027?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/Java%20%E7%BC%96%E7%A8%8B%E8%AF%AD%E8%A8%80/_blank)，可分为基本部分和扩展部分。在硬件或操作系统平台上安装一个Java平台之后，Java应用程序就可运行。Java平台已经嵌入了几乎所有的操作系统。这样Java程序可以只编译一次，就可以在各种系统中运行。Java应用编程接口已经从1.1x版发展到1.2版。常用的Java平台基于Java1.8。商业开发一般采用稳定性较好的JDK8.0。

（2）SQL server

SQL Server是美国Microsoft公司推出的关系型数据库管理系统，具有使用方便可伸缩性好与相关软件集成度高等优点，可跨越从运行Microsoft Windows 98的微型计算机到Microsoft Windows 2012的大型多处理器的服务器等多种平台使用。但现在普遍使用MySQL软件，SQL server的普及率较以前降低。

SQL server具备以下特性：

（1）高性能设计，可充分利用WindowsNT的优势。

（2）系统管理先进，支持Windows图形化管理工具，支持本地和远程的系统管理和配置。

（3）强壮的事务处理功能，采用各种方法保证数据的完整性。

（4）支持对称多处理器结构、存储过程、ODBC，并具有自主的SQL语言。 SQLServer以其内置的数据复制功能、强大的管理工具、与Internet的紧密集成和开放的系统结构为广大的用户、开发人员和系统集成商提供了一个出众的数据库平台。

SQL语句可以用来执行各种各样的操作，例如更新数据库中的数据，从数据库中提取数据等。目前，绝大多数流行的关系型数据库管理系统，如Oracle,Sybase,Microsoft SQL Server,Access等都采用了SQL语言标准。虽然很多数据库都对SQL语句进行了再开发和扩展，但是包括Select,Insert,Update,Delete,Create,以及Drop在内的标准的SQL命令仍然可以被用来完成几乎所有的数据库操作。不同数据库的SQL语言不完全一致，在触发器上面，SQL server与MySQL两者的语言并不相同。

1.2.2 开发工具

本系统在Windows 11操作系统上开发，采用JDK环境进行开发，数据库管理系统采用eclipse。

1. Eclipse

Eclipse 是一个开放源代码的、基于[Java](https://baike.baidu.com/item/Java/85979?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)的可扩展[开发平台](https://baike.baidu.com/item/%E5%BC%80%E5%8F%91%E5%B9%B3%E5%8F%B0/8956190?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)。就其本身而言，它只是一个框架和一组服务，用于通过插件组件构建[开发环境](https://baike.baidu.com/item/%E5%BC%80%E5%8F%91%E7%8E%AF%E5%A2%83/10119007?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)。幸运的是，Eclipse 附带了一个标准的插件集，包括Java[开发工具](https://baike.baidu.com/item/%E5%BC%80%E5%8F%91%E5%B7%A5%E5%85%B7/10464557?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)（Java Development Kit，[JDK](https://baike.baidu.com/item/JDK/1011?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)）。

Eclipse是著名的[跨平台](https://baike.baidu.com/item/%E8%B7%A8%E5%B9%B3%E5%8F%B0/8558902?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)的自由[集成开发环境](https://baike.baidu.com/item/%E9%9B%86%E6%88%90%E5%BC%80%E5%8F%91%E7%8E%AF%E5%A2%83?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)（IDE）。最初主要用来Java语言开发，通过安装不同的插件Eclipse可以支持不同的[计算机语言](https://baike.baidu.com/item/%E8%AE%A1%E7%AE%97%E6%9C%BA%E8%AF%AD%E8%A8%80/4456504?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)，比如C++和Python等开发工具。Eclipse的本身只是一个框架平台，但是众多插件的支持使得Eclipse拥有其他功能相对固定的IDE软件很难具有的灵活性。许多软件开发商以Eclipse为框架开发自己的IDE。

Eclipse 最初由OTI和[IBM](https://baike.baidu.com/item/IBM?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)两家公司的IDE[产品开发](https://baike.baidu.com/item/%E4%BA%A7%E5%93%81%E5%BC%80%E5%8F%91/6396644?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)组创建，起始于1999年4月。IBM提供了最初的Eclipse代码基础，包括Platform、JDT 和PDE。Eclipse项目IBM发起，围绕着Eclipse项目已经发展成为了一个庞大的Eclipse联盟，有150多家软件公司参与到Eclipse项目中，其中包括[Borland](https://baike.baidu.com/item/Borland/493987?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)、Rational Software、[Red Hat](https://baike.baidu.com/item/Red Hat/7360805?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)及[Sybase](https://baike.baidu.com/item/Sybase?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)等。Eclipse是一个[开放源码](https://baike.baidu.com/item/%E5%BC%80%E6%94%BE%E6%BA%90%E7%A0%81/7176422?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)项目，它其实是Visual Age for Java的[替代品](https://baike.baidu.com/item/%E6%9B%BF%E4%BB%A3%E5%93%81/8249884?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)，其界面跟先前的Visual Age for Java差不多，但由于其开放源码，任何人都可以免费得到，并可以在此基础上开发各自的插件，因此越来越受人们关注。随后还有包括[Oracle](https://baike.baidu.com/item/Oracle/301207?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/eclipse/_blank)在内的许多大公司也纷纷加入了该项目，Eclipse的目标是成为可进行任何语言开发的IDE集成者，使用者只需下载各种语言的插件即可。

（2）Microsoft SQL Server Management Studio

SQL Server Management Studio 19是一个集成环境，用于访问、配置、管理和开发SQL Server的所有组件。SQL Server Management Studio组合了大量图形工具和丰富的脚本编辑器，使各种技术水平的开发人员和管理员都能访问SQL Server。

SQL Server 是[Microsoft](https://baike.baidu.com/item/Microsoft/125917?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/_blank) 公司推出的[关系型数据库管理系统](https://baike.baidu.com/item/%E5%85%B3%E7%B3%BB%E5%9E%8B%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F/696511?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/_blank)。具有使用方便[可伸缩性](https://baike.baidu.com/item/%E5%8F%AF%E4%BC%B8%E7%BC%A9%E6%80%A7/1414004?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/_blank)好与相关[软件集成](https://baike.baidu.com/item/%E8%BD%AF%E4%BB%B6%E9%9B%86%E6%88%90/895406?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/_blank)程度高等优点，可跨越从运行Microsoft [Windows 98](https://baike.baidu.com/item/Windows 98/758579?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/_blank) 的膝上型电脑到运行[Microsoft Windows](https://baike.baidu.com/item/Microsoft Windows/3304184?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/_blank) 2012 的大型[多处理器](https://baike.baidu.com/item/%E5%A4%9A%E5%A4%84%E7%90%86%E5%99%A8/53566627?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/_blank)的服务器等多种平台使用。

Microsoft SQL Server 是一个全面的数据库平台，使用集成的[商业智能](https://baike.baidu.com/item/%E5%95%86%E4%B8%9A%E6%99%BA%E8%83%BD/406141?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/_blank) (BI)工具提供了企业级的[数据管理](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E7%AE%A1%E7%90%86/295844?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/_blank)。Microsoft SQL Server [数据库引擎](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%BA%93%E5%BC%95%E6%93%8E/1239260?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/_blank)为关系型数据和[结构化数据](https://baike.baidu.com/item/%E7%BB%93%E6%9E%84%E5%8C%96%E6%95%B0%E6%8D%AE/5910594?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/_blank)提供了更安全可靠的存储功能，使您可以构建和管理用于业务的高可用和高性能的数据[应用程序](https://baike.baidu.com/item/%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F/5985445?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/_blank)。

1.2 技术路线

本次课设着重分析客户需求与供货商供货能力的情况，设计了针对客户、运货商以及供货商，设置了三种七层界面，针对不同的登陆者提供不同的服务。

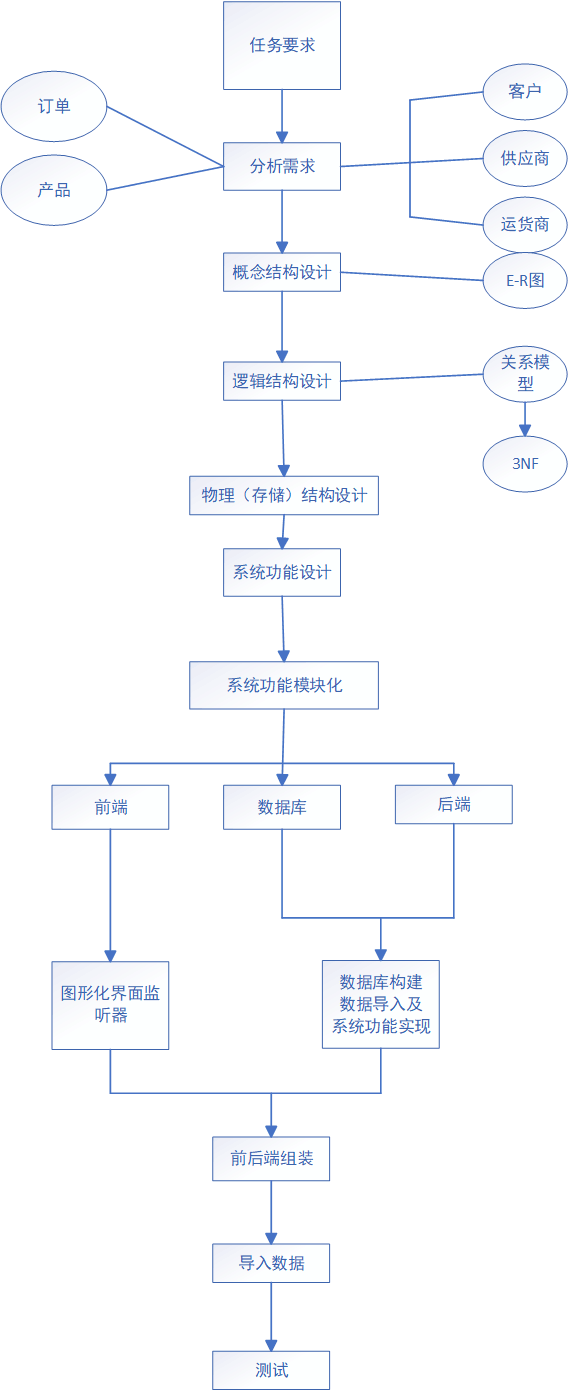
对顾客，完成客户登录与注册，优惠客户的识别，查看订购货物类别及数量，查看购物车，查看购物车，查看剩余库存在库存不足时提醒供货商进货并退回购物车等。并十分人性化的添加了提示顾客添加完成与退回的界面。并设计了菜单，方便顾客返回初始界面等。

供货商收集客户的消费金额及订购数量来选择货物的采购以增加或者删除产品的能力，设置了可以通过订单号或产品名称进行查询的，并设置中止量防止库存亏空等，用来进一步满足客户及潜在消费者，大公司的需求。

运货商对于运货订单进行管理，对于未运货的订单进行特别关注。但并未设置特别详细的界面。

开发技术路线如图1-1所示：

图 1-1 开发路线图

****

**2 功能分析与设计**

2.1 需求分析

**2.1.1 用户调研**

此产品管理系统的用户是客户及公司和供货商，客户必须登录才能进行购买，否则视为非法，若客户无账号，则在登录界面选择注册，完成注册后才有购买功能，进一步保证了客户的隐私性与交易的合法性，产品订购页面的checkbox的设计能够清楚直观地展现产品及数量，便于操作，满足客户实时方便地查看产品的类别及库存，有针对性的订购商品，翻看库存；满足供货商对客户需求的掌握，更有针对性储存产品，可通过设置ID与产品名称来查找货物，查看货物情况，同时设置中止量，防止差错；运货商处有发货订单与未发货订单，防止过长时间不发货导致产品积压，并有到货日期提醒。

**2.1.2 初步设计**

依系统需求，将系统功能分为如下几块：

(1)初始为选择身份界面，有客户，供货商，与运货商三种选择，不同选择进入不同的操作界面。

(2)用户的登录与注册：用户选择进入后是登录及注册界面，同时可以选择密码寻回；若角色未有数据在数据库中，则注册时补充完整信息；登录后，系统判定为优质客户的可以享受优惠，否则无优惠；进入产品查询界面后，可以查询产品ID也可以查询产品名称，容存在产品则直接进入，否则弹窗弹出警告，进入下一个数量选择及查询库存界面，用CheckBox滚轮选择数量，可选择多个产品添加进购物车，点击购物车界面可以进入购物车界面进行查看，同时可以删除多余产品，无法下单时会直接清空购物车。

(3)供应商：供应商分三块面板，分别对应对应基础信息，商品产看界面（从数据库中读出），及查询产品ID与产品名称，查看产品的具体情况，设置出货中止量，截断产品出货，

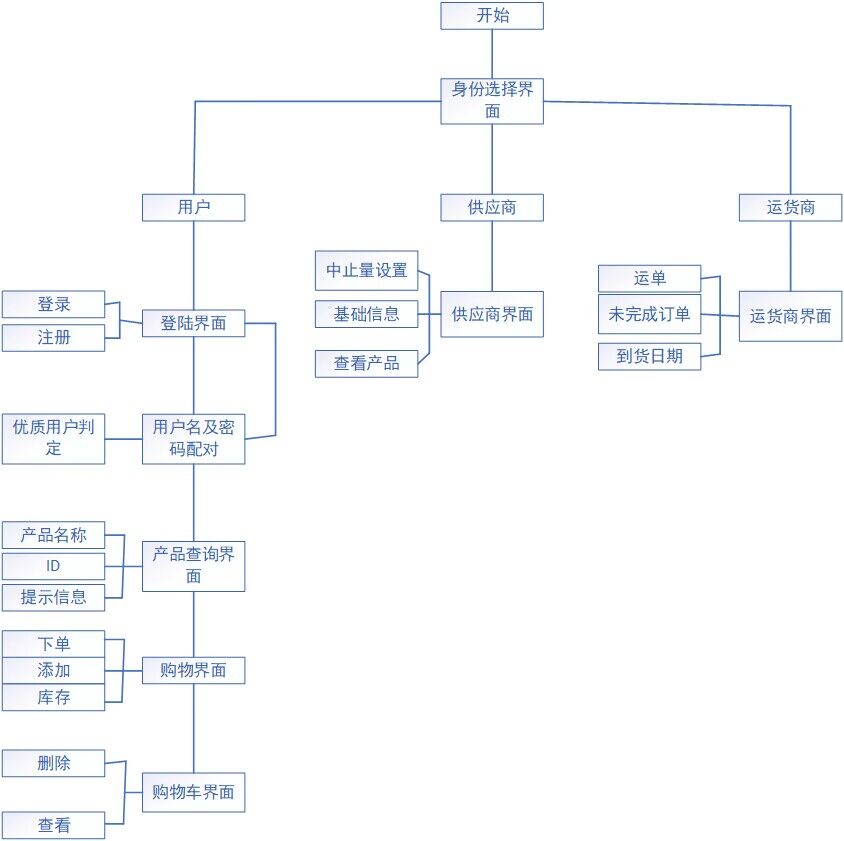
(4)运货商：运货商成员进入运货商界面后，可查看订单的完成的情况及未完成的订单，提醒运货商及时运货。

2.2 系统设计

**2.2.1 系统流程设计**

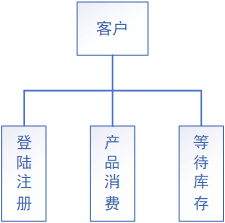
按照各个身份的需求分析所得到的信息后，整个系统流程具体将细化到如图3-1所示：

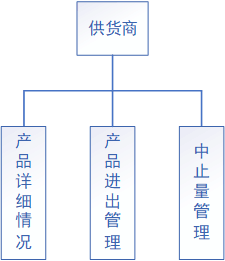
图3-1系统流程图



**2.2.2 系统功能分析**

根据上述系统流程进行分析，将客户，供货商，运货商三份身份分开进行处理，得三份身份的功能概括为图3-2：





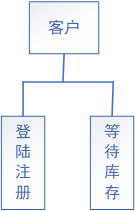


图3-2系统功能图

而系统总功能为承接各个身份，做统一的平台进行处理，下图3-3：

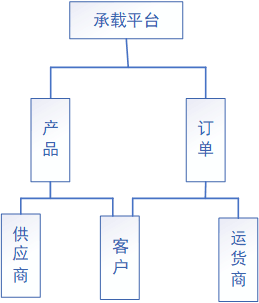


图3-3系统总功能图

2.3 数据库概念结构设计

数据库概念结构设计：

根据需求分析，我们建立如下的E-R模型，如图3-4所示：

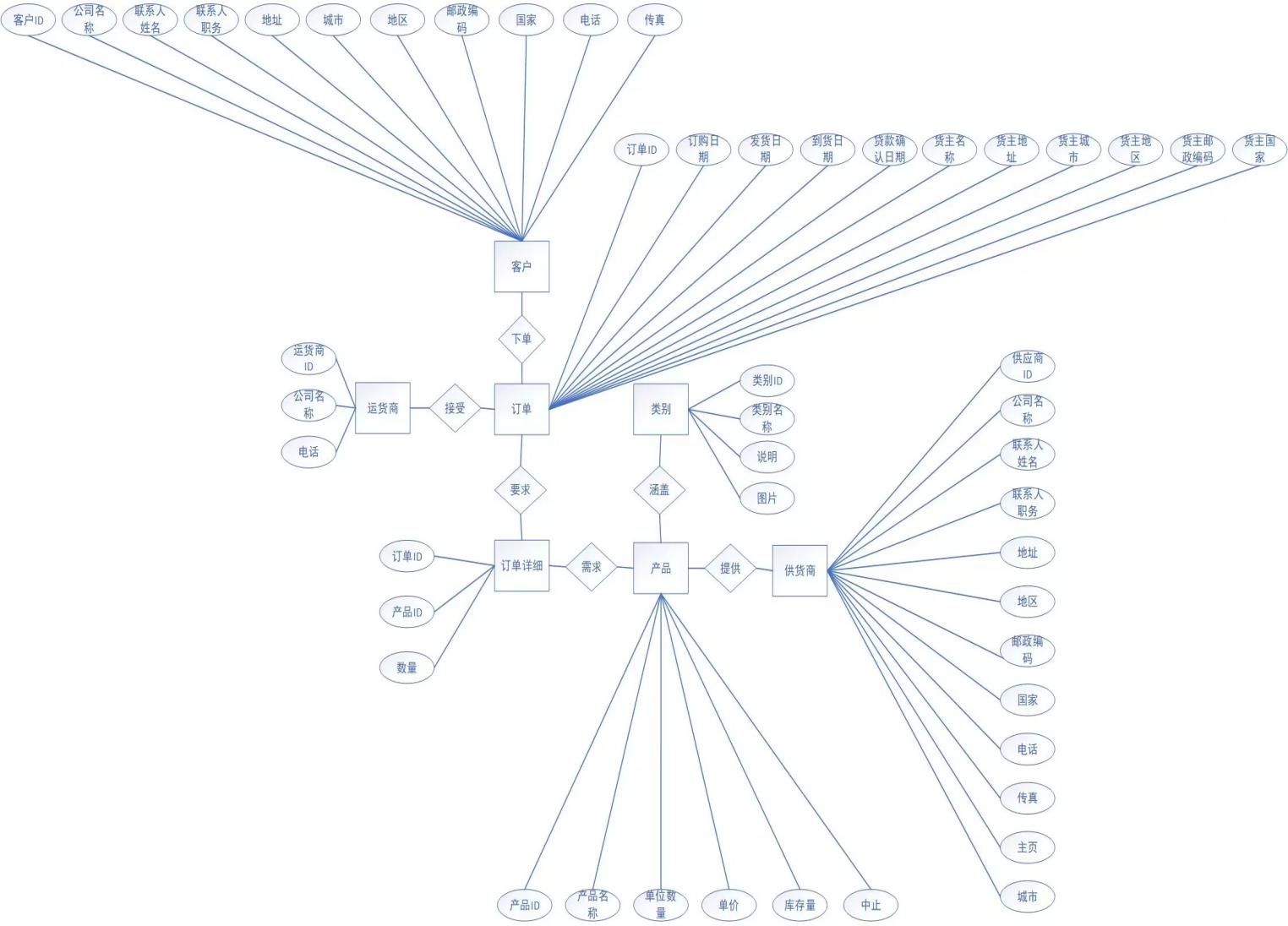
****

图3-4 E-R模型图

2.4 数据库逻辑结构设计

数据库逻辑结构设计即将概念结构转换为某个数据库管理系统所支持的数据模型，并对其进行优化。本课程设计采用关系数据库，需要实现E-R图到关系模型的转换。

将每个成员各个归类，按照成员分类各个拆解出实体，每个实体的表格竖列对应其属性，关系为所属的各个实体的现实联系，一般不表示在表格内，同时各个实体的数量关系不表示在关系模型中。

**2.4.1 E-R图向关系模型的转换**

客户

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 客户ID | 公司名称 | 联系人姓名 | 联系人职务 | 地址 | 城市 | 地区 | 邮政编码 | 国家 | 传真 |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

运货商

|  |  |  |
| --- | --- | --- |
| 运货商ID | 公司名称 | 电话 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

订单

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 订单ID | 订购日期 | 发货日期 | 到货日期 | 货款确认日期 | 货主名称 | 货主地址 | 货主城市 | 货主地区 | 货主邮政编码 | 货主国家 |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |

订单详细

|  |  |  |
| --- | --- | --- |
| 订单ID | 产品ID | 数量 |
|  |  |  |
|  |  |  |
|  |  |  |

类别

|  |  |  |  |
| --- | --- | --- | --- |
| 类别ID | 类别名称 | 说明 | 图片 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

产品

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 产品ID | 产品名称 | 单价数量 | 单价 | 库存量 | 中止量 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

供货商

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 供货商ID | 公司名称 | 联系人姓名 | 联系人职务 | 地址 | 地区 | 邮政编码 | 国家 | 电话 | 传真 | 主页 | 城市 |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |

**2.4.2 利用数据依赖的理论对关系模式进行分析及最终关系模型设计**

首先，我们将原始关系模型中的表转换为符合 1NF 的表，得到以下表：

产品（产品ID，产品名称，单价数量，产品单价，库存量，中止量，供货商ID，类别ID）

供货商（供货商ID，供货商公司名称，供货商联系人职务，供货商联系人姓名，供货商地址，供货商地区，供货商邮政编码，供货商国家，供货商电话，供货商传真，供货商主页，供货商城市）

类别（类别ID，类别名称，说明，图片）

订单详情（订单ID，产品ID，数量）

订单（订单ID，订购日期，发货日期，到货日期，货款确认日期，客户ID，运货商ID）

运货商（运货商ID，运货公司名称，运货商电话）

客户（客户ID，客户公司名称，客户联系人名称，客户联系人职务，客户地址，客户城市，客户地区，客户邮政编码，客户国家，客户传真）

然后，我们需要对每个表进行功能依赖分析，以确保每个表都符合第三范式（3NF）。

在产品表中，存在以下函数依赖：

产品ID → 产品名称，单价数量，产品单价，库存量，中止量，供货商ID，类别ID

供货商ID → 供货商公司名称，供货商联系人职务，供货商联系人姓名，供货商地址，供货商地区，供货商邮政编码，供货商国家，供货商电话，供货商传真，供货商主页，供货商城市

类别ID → 类别名称，说明，图片

因此，我们可以将产品表分解为以下两个表：

产品（产品ID，产品名称，单价数量，产品单价，库存量，中止量，供货商ID，类别ID）

供应商（供货商ID，供货商公司名称，供货商联系人职务，供货商联系人姓名，供货商地址，供货商地区，供货商邮政编码，供货商国家，供货商电话，供货商传真，供货商主页，供货商城市）

类别（类别ID，类别名称，说明，图片）

在订单表中，存在以下函数依赖：

订单ID → 订购日期，发货日期，到货日期，货款确认日期，客户ID，运货商ID

产品ID → 数量

因此，我们可以将订单详情表分解为以下两个表：

订单详情（订单ID，产品ID，数量）

订单（订单ID，订购日期，发货日期，到货日期，货款确认日期，客户ID，运货商ID）

在客户表中，存在以下函数依赖：

客户ID → 客户公司名称，客户联系人名称，客户联系人职务，客户地址，客户城市，客户地区，客户邮政编码，客户国家，客户传真

因此，客户表已经符合 3NF。

在运货商表中，存在以下函数依赖：

运货商ID → 运货公司名称，运货商电话

因此，运货商表已经符合 3NF。

最后，我们需要定义每个表的主键、外键和完整性约束：

产品表的主键是产品ID，外键是供货商ID和类别ID，参照供应表和类别表的主键。产品表中的供货商ID和类别ID必须参照供应表和类别表中已有的值。

供应表的主键是供货商ID，没有外键。

类别表的主键是类别ID，没有外键。

订单详情表的主键是订单ID和产品ID，外键是订单ID和产品ID，参照订单表和产品表的主键。订单详情表中的订单ID和产品ID必须参照订单表和产品表中已有的值。

订单表的主键是订单ID，外键是客户ID和运货商ID，参照客户表和运货商表的主键。 订单表中的客户ID和运货商ID必须参照客户表和运货商表中已有的值。

运货商表的主键是运货商ID，没有外键。

客户表的主键是客户ID，没有外键。

以上是将关系模型转换为3NF约束的关系模型，并说明主键、外键和完整性约束。

2.5 数据库物理结构设计

数据库物理结构设计是指将逻辑数据模型转换成物理存储模型的过程。它涉及到表的设计、索引的设计、分区、数据类型、存储引擎等方面的决策。以下是一些常见的数据库物理结构设计要点：

（1）表的设计：确定表的名称、字段类型、长度、约束及默认值等。需要考虑到数据的完整性、可扩展性、性能等因素。

（2）索引的设计：确定需要创建哪些索引，选择索引类型，以及索引的列。需要综合考虑查询频率、查询条件、数据更新频率等因素。

（3）分区：对于大型数据库系统，可以通过分区来提高查询性能和数据管理效率。可以按照时间、地理位置、业务等因素进行分区。

（4）数据类型：选择合适的数据类型对于数据库性能和数据完整性至关重要。需要根据实际情况选择合适的数据类型。

（5）存储引擎：不同的存储引擎有不同的优缺点，需要根据实际情况选择合适的存储引擎，如InnoDB、MyISAM等。

在进行数据库物理结构设计时，需要充分考虑数据安全、数据完整性、数据可扩展性、查询性能等因素。

**2.5.1确定存储结构**

在数据库物理结构设计中，确定存储结构是一个非常重要的步骤。存储结构的选择将直接影响数据库的性能和可靠性。

以下是一些常见的存储结构：

（1）堆文件：将所有记录都插入到一个堆中，无排序和索引。这种结构对于小型数据库来说可能是最简单的，但是大型数据库如果使用堆文件，性能和查询效率都会受到影响。

（2）顺序文件：将记录按照一个特定的关键字进行排序，可以提高查询效率。但是，如果频繁地插入和删除记录，需要重新排序，对性能影响很大。

（3）索引文件：使用索引结构来提高查询效率。常见的索引结构包括B树、B+树和哈希表。B树和B+树适用于范围查询，而哈希表适用于等值查询。索引文件可以提高查询效率，但是在插入和删除记录时需要更新索引，会影响性能。

（4）散列文件：使用哈希函数将记录散列到不同的桶中，可以提高查询效率。但是，如果哈希函数不好，会导致数据分布不均匀，影响查询效率。

在确定存储结构时，需要根据数据库的大小、数据访问模式、查询需求和系统资源等因素进行考虑。综合考虑，可以采用多种存储结构的组合来提高性能和可靠性。

本次系统采用的数据库是SQL Server2022，数据库以文件形式存储在计算机的磁盘中，主要的文件有主数据文件，次数据文件，事物日志文件：

（1）主数据文件：主数据文件存储了数据库的所有数据，包括表、索引、视图等。这个文件是数据库中最重要的文件，通常也是最大的文件。

（2）次数据文件：次数据文件是主数据文件的补充，当主数据文件无法再存储更多数据时，可以通过增加次数据文件来扩展数据库的存储容量。次数据文件与主数据文件的结构相同。

（3）事务日志文件：事务日志文件记录了所有数据库事务的操作，包括事务开始、结束、回滚等。这个文件是用于数据库恢复和故障恢复的重要文件。

（4）这些文件在数据库的运行中扮演着不同的角色，主数据文件和次数据文件用于存储数据库的实际数据，而事务日志文件用于记录数据库操作的历史信息，以便在需要时进行恢复和修复。

本次数据库采用堆文件方式，而数据由Excal表格储存，是顺序结构，用户身份登入SQL server后，导入数据，查看数据库视图，确认导入数据正常。

**2.5.2评价物理结构**

数据库的物理结构是指数据库在磁盘上的存储方式，包括数据文件、索引文件等。评价数据库的物理结构主要从以下几个方面考虑：

（1）存储效率：即数据库在磁盘上的存储效率，包括存储空间利用率和读写速度等。合理的物理结构能够提高存储空间利用率，减少存储空间的浪费，并且能够提高数据库的读写效率。

（2）维护成本：即维护数据库物理结构所需要的时间和人力成本。合理的物理结构应该能够减少对数据库的维护成本，提高系统的稳定性和可靠性。

（3）数据访问效率：即数据库在查询和修改数据时的效率。合理的物理结构应该能够提高查询和修改数据的效率，使得用户能够更快地获取需要的数据。

（4）安全性：即数据库在物理存储上的安全性。合理的物理结构应该能够保护数据库的数据安全，防止数据泄露和损坏等情况的发生。

数据文件采用堆方式储存，不分先后，在较少文件的储存中，在查找上能同时查找，减少时间浪费，而文件内数据采用顺序方式储存，在文件内数据量较大时，查找节省时间。但是如果频繁的插入删除数据，则需重新排序，对系统性能影响较大。

2.6 模块划分

对于模块划分，是依据不同用户身份开发的，在初始界面的三个模块中，不同的用户身份对应对应不同功能需求，将分成如下模块：

1. 身份选择模块：对应不同身份进入不同处理机制：
2. 用户身份
3. 供应商界面
4. 运货商界面
5. 进入用户界面，首先登录和注册，如果无法登录，则进行注册，填入信息，完成注册，返回登录界面进行登录。

·完成登录后，进入商品查询界面，产品ID与名称都能查询。

·商品查询界面有菜单，菜单可以查看用户的详细信息，历史订单能查询用户之前的购买记录，登出键能快速登出，便于客户切换账号。

·进入产品页面，CheckBox滚轮能快速选择商品数量，添加购物车，并查看库存。

·进入购物车界面，可删除产品。

·购物车界面可下单，下单后弹窗提示。

1. 进入供应商界面：

·显示公司详情

·显示产品详情

·显示顾客下单情况

·提供运货商订单信息

·可及时添加库存，可设置产品中止量

1. 进入运货商界面，显示订单情况，对订单情况分组管理。

·已完成的订单在已完成的界面中。

·未完成的订单进入未完成界面。

·完成订单由客户确认签收

**3实现**

3.1 前端界面实现

这次数据库课设采用分组的形式完成，本小组分配到的是任务2，前端同学的界面开发主要用Java语言在eclipse上完成编程，采用requires java.desktop导入Java的图形化界面包。

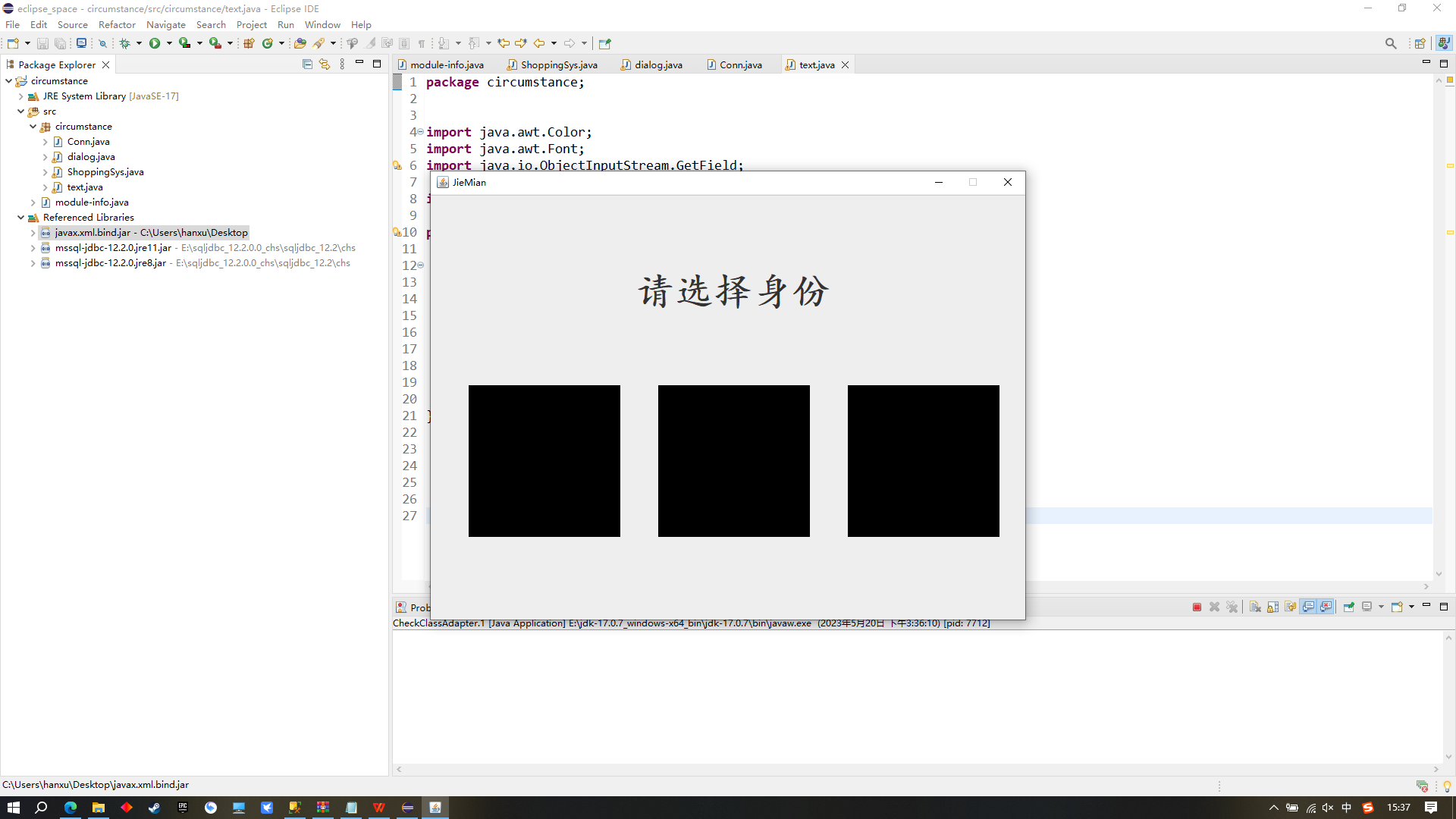
Eclipse

由于之前未开发过前端，一切都是自己摸索的，前端的七个图形可视化界面都承载在一处基底界面上，即jp0,同时界定了所有界面的大小，但不显示任何东西，在演示时不可见。

每个界面的定义与数据设置分类进行，防止混淆。

同时每个界面设置监听器，监听鼠标动作、监听键盘录入与数据库数据反馈。

**3.1.1 最初界面设计**



在同一界面承载了三种身份的进入入口，系统使用方便，美观简洁。

这个界面三个选择框代表了三种身份的登录，最左边是客户，中间是供应商，最右边是运货商。

JPanel jpf=**new** JPanel();

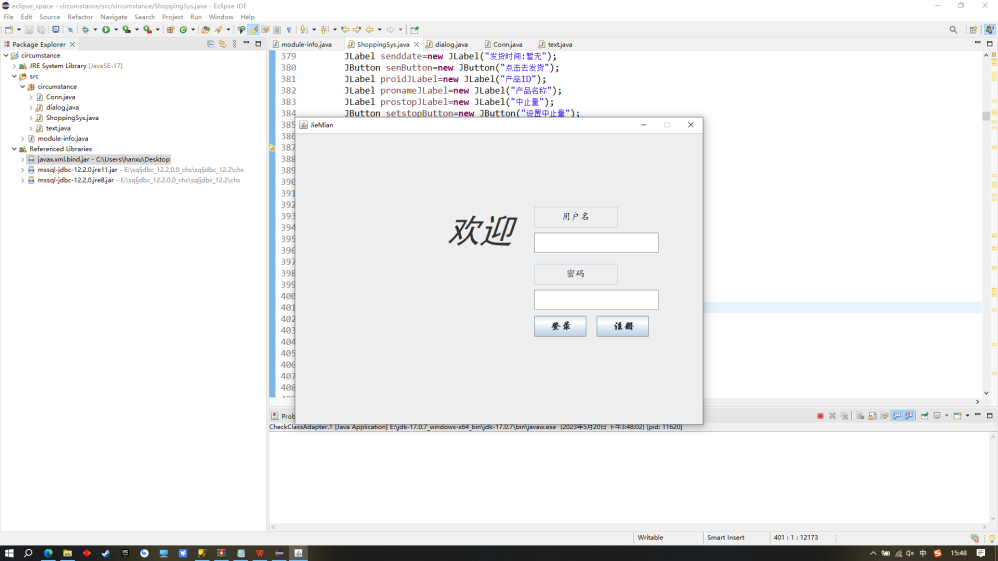
JLabel choiceJLabel=**new** JLabel("请选择身份");

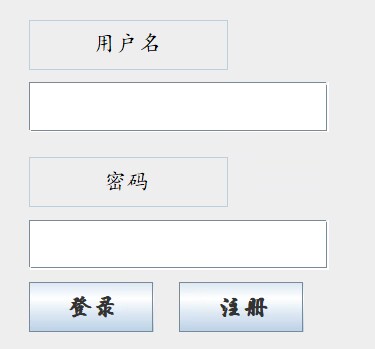
JLabel clientJLabel=**new** JLabel();//客户

JLabel supplyJLabel=**new** JLabel();//供应商

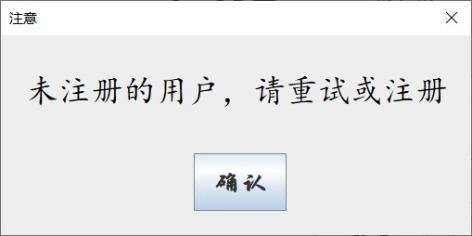
JLabel deliverJLabel=**new** JLabel();//运货商

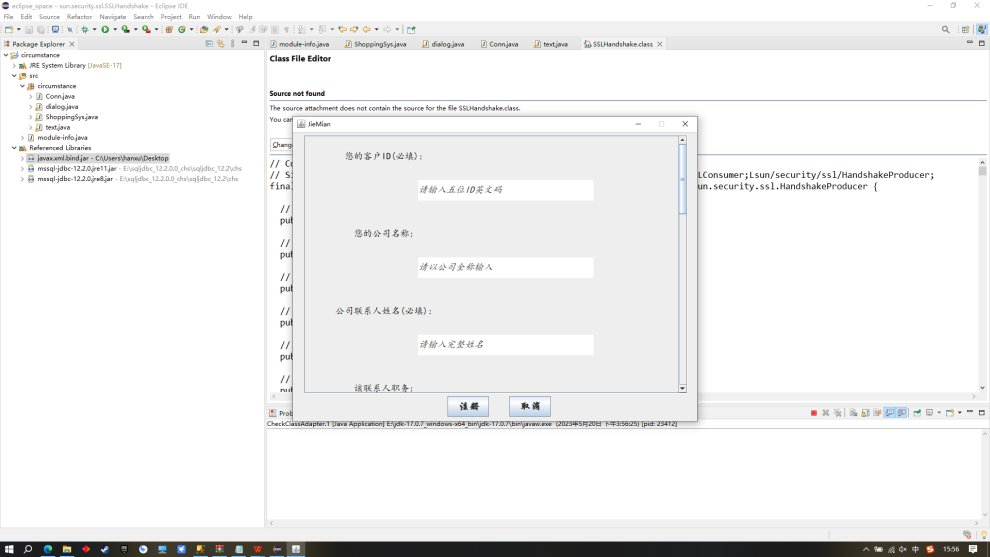
**3.1.2 客户界面设计**

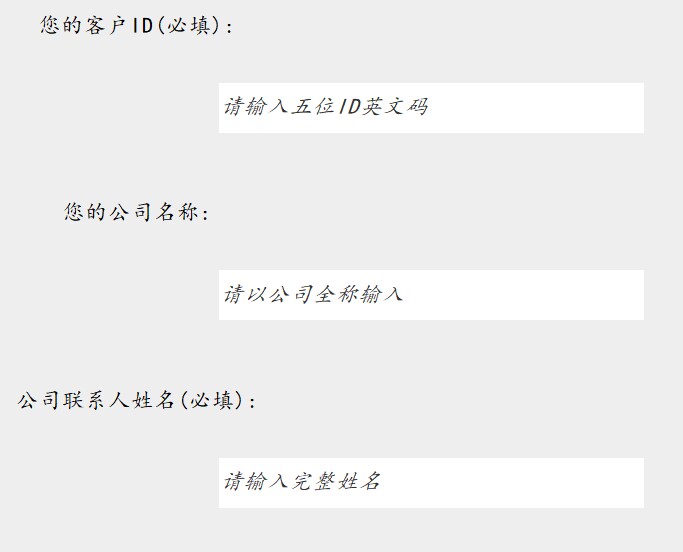


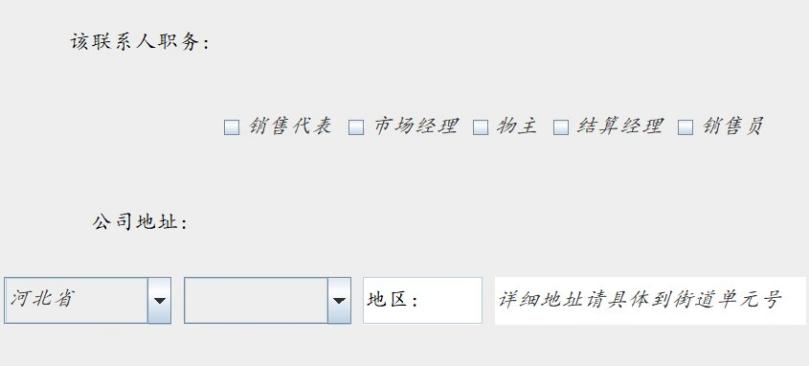


如果有客户记录，直接登陆使用软件，如果显示未登录，则弹窗提醒顾客登录，并转入以下界面进行注册。

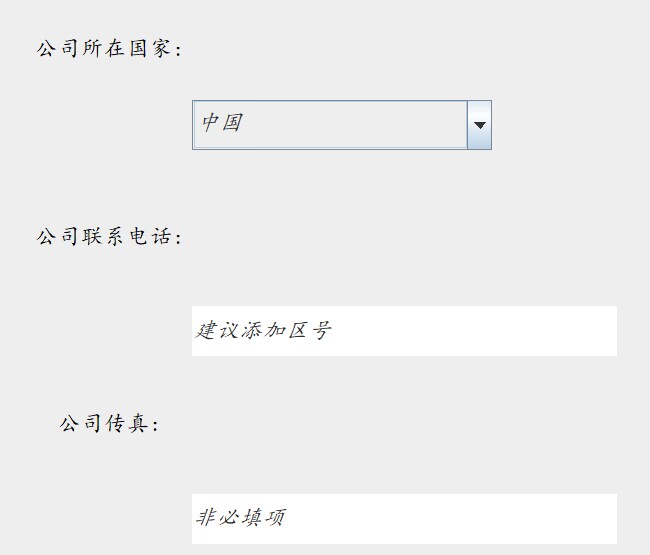


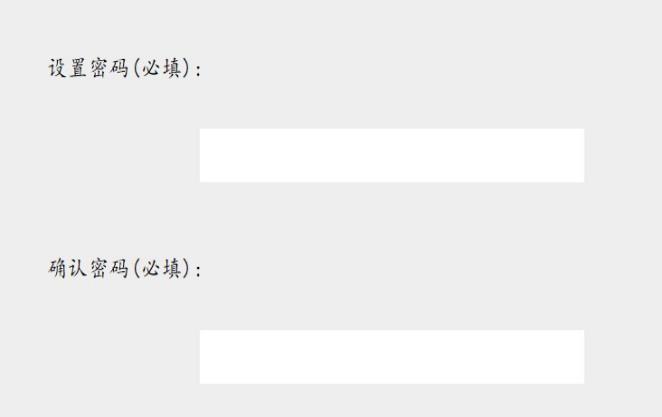




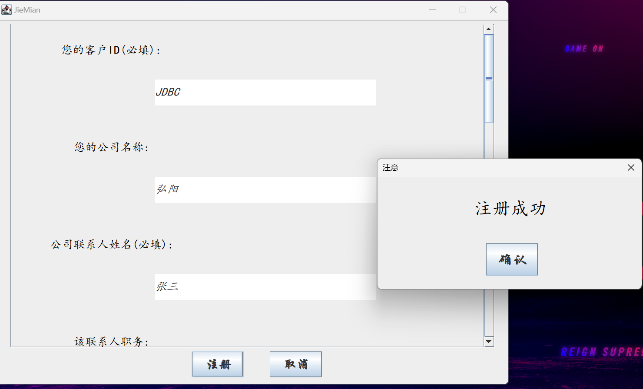


CheckBox的滚轮设计能让顾客免去打字错误，更快捷填入信息。



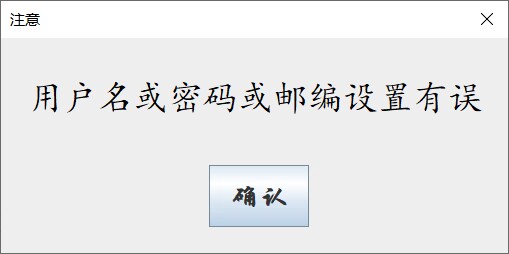


填入信息，对于本系统，所采集的都是string类型的，在后台将string类型转换为daytime类型的。



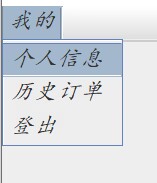
如果完成注册，则提示注册成功。

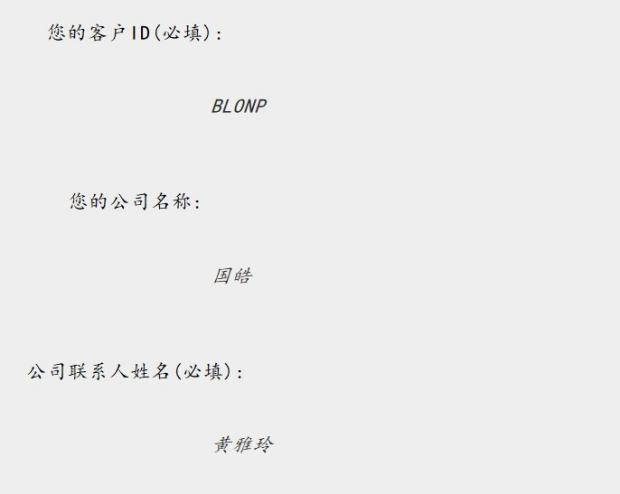
完成注册后，回到登录界面进行登录。



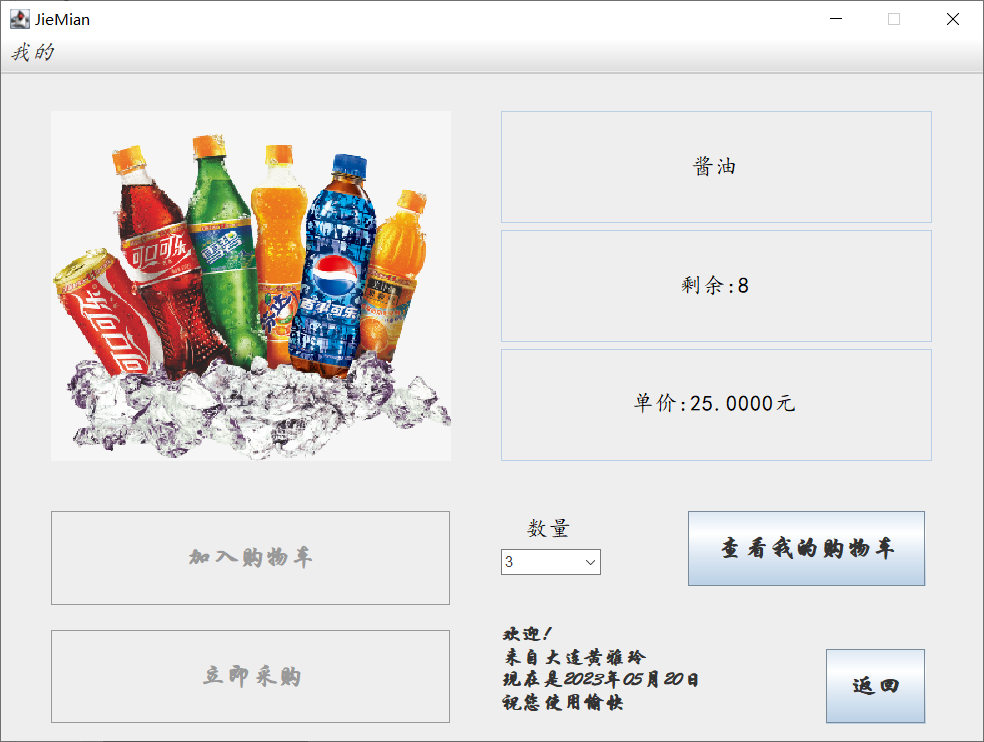
如果注册不成功，则弹窗提示设置有误。



菜单提供了查看个人信息，查看历史订单，登出三种功能。



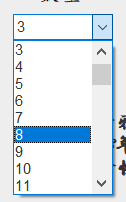
完成登录后，进入产品查询界面。在本界面，既可以通过输入产品ID，也可以通过输入产品名称进行搜索，如果搜索成功，进入以下界面。



进入本界面，客户可以在本界面查看产品的具体信息，并选择添加进入购物车。

数量的选择是本界面的亮点，采用了CheckBox滚动选择，提高便捷性。

本界面的监听器只能在一次操作后进行监听，必须重新选择数量框中的数字才能进行刷新，否则刷新无效。

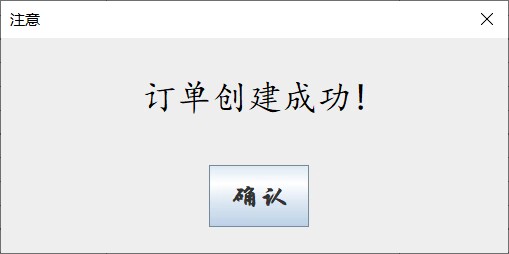




添加成功后出现弹窗提示，提醒顾客。

图形用户界面, 表格

描述已自动生成

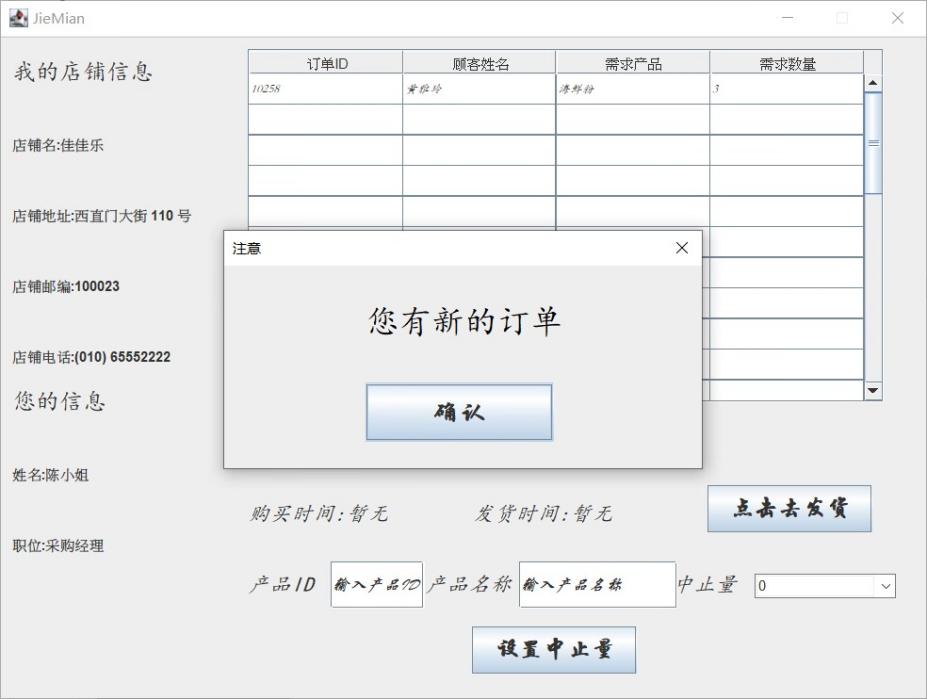


订单创建成功后，弹窗提示客户创建完成，运货公司将根据客户信息提供送货。



进入购物车后，可查看已经选择的商品，确认订购的数量，可以删除多余的商品，本界面的监听器在购物车清空后，使删除框变灰，取消删除功能，防止多余动作。

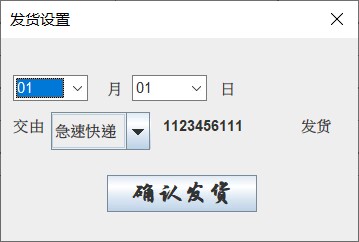
**3.1.3 供应商界面设计**



供应商界面显示供应商信息，同时显示本公司产品详情，监视库存与中止量。

下端输入界面，无论输入产品ID还是输入产品都可以显示对应的信息。

当顾客下单后，系统弹窗提示供应商有新订单。



供应商将订单发送至运货商，设定发货时间，并指定运货公司与编号。



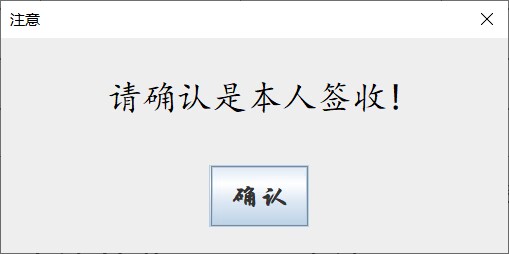
**3.1.4运货商界面设计**



将订单进行分组管理，已完成的订单分一组，未完成的订单分一组，防止混淆。







进入运货商界面后，显示订单信息，查看已完成的订单，如果客户已签收，则点击确认，并弹窗提醒客户本人签收，保护隐私。对未完成的订单进行管理，记录订单详情。

3.2后端代码实现

此次数据库课程设计的代码量较多且具有冗余性，为此仅选取核心和关键部分的代码进行说明和解释。

**3.2.1 java与SQL Server的连接和交互**

本次课程设计中，为了实现java与SQL Server数据库的连接交互功能，先用jdbc配置一个连接环境，再建立了一个类Conn以封装对数据库的连接操作。通过输入URL、数据库用户名和密码与数据库建立连接，再用close方法关闭与数据库的连接，以释放占用的资源**。**

**3.2.2部分功能代码**

**为了更方便展示本次数据库课程设计的具体实现，节选了一些基本功能的代码，以说明java是怎样与数据库进行交互的。**

**（1）登录功能**

psd.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

String tmp=acc.getText();

Connection conn=**null**;

PreparedStatement ps=**null**;

ResultSet resultSet=**null**;

**int** p=1;

String sql = "SELECT \* FROM 客户 where 客户ID=?;";

**try** {

conn=Conn.*getConnection*();

ps=conn.prepareStatement(sql);

ps.setString(1, tmp);

resultSet=ps.executeQuery();

**if**(resultSet.next())

{

String tmp0=acc.getText();

String tmp1=**new** String(psd.getPassword());

Connection conn1=**null**;

PreparedStatement ps1=**null**;

ResultSet resultSet1=**null**;

String sql1 = "SELECT \* FROM 客户 where 客户ID=?;";

String tmp01=acc.getText();

String tmp11=**new** String(psd.getPassword());

Connection conn11=**null**;

PreparedStatement ps11=**null**;

ResultSet resultSet11=**null**;

String sql11 = "SELECT \* FROM 客户 where 客户ID=?;";

**try** {

conn11=Conn.*getConnection*();

ps11=conn11.prepareStatement(sql11);

ps11.setString(1, tmp01);

resultSet11=ps11.executeQuery();

**if**(resultSet11.next())

{

**try**{

{

p=resultSet11.getString("密码").compareTo(tmp11);

}

**if**(p==0)

{

info.setText("欢迎!\n来自"+resultSet.getString("地区")+resultSet.getString("城市")+"的"+resultSet.getString("联系人姓名")+"\n现在是"+formatter.format(calendar.getTime())+"\n祝您使用愉快");

card.show(jp0, "3");

my.setVisible(**true**);

}

**else**

{

dialog tip=**new** dialog("密码错误请重试或注册", 's');

tip.jb1Button.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

tip.dispose();

}

});

tip.setVisible(**true**);

}}**catch**(Exception e1){

info.setText("欢迎!\n来自"+resultSet.getString("地区")+resultSet.getString("城市")+"的"+resultSet.getString("联系人姓名")+"\n现在是"+formatter.format(calendar.getTime())+"\n祝您使用愉快");

card.show(jp0, "3");

my.setVisible(**true**);

}

}

}**catch**(Exception e3){

e3.printStackTrace();

}**finally** {

Conn.*close*(conn11, ps11, resultSet11);

}

}

**else**

{

dialog tip=**new** dialog("未注册的用户，请重试或注册", 's');

tip.jb1Button.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

tip.dispose();

}

});

tip.setVisible(**true**);

}

}**catch**(Exception e3){

e3.printStackTrace();

}**finally** {

Conn.*close*(conn, ps, resultSet);

}

}

});

logup.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

card.show(jp0, "2");

}

});

**（2）注册功能**

out1.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

out1.setVisible(**false**);

in.setVisible(**true**);

out.setVisible(**true**);

idJTextField.setEditable(**true**);companyJTextField.setEditable(**true**);conName.setEditable(**true**);scjl.setVisible(**true**);xsdb.setVisible(**true**);wz.setVisible(**true**);jsjl.setVisible(**true**);xsy.setVisible(**true**);conNameJTextField.setEditable(**true**);guojia.setEnabled(**true**);addressJTextField.setVisible(**true**);

pro.setVisible(**true**);addressJTextField.setEditable(**true**);guojia.setEditable(**true**);phoneJTextField.setEditable(**true**);faxField.setEditable(**true**);posJTextField.setEditable(**true**);passwordField.setVisible(**true**);coPasswordField.setVisible(**true**);city.setVisible(**true**);area.setVisible(**true**);in\_psdText.setVisible(**true**);confirm\_psdText.setVisible(**true**);

degree.setVisible(**false**);de.setVisible(**false**);

card.show(jp0, "3");

my.setVisible(**true**);

}

});

in.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

String s1=idJTextField.getText();

String s2=companyJTextField.getText();

String s3=conNameJTextField.getText();

String s4="";

String s5=addressJTextField.getText();;

String s6="";

String s7="";

String s8=posJTextField.getText();

String s9="";

String s10=phoneJTextField.getText();;

String s11=faxField.getText();

String s12=**new** String(passwordField.getPassword());

String s13=**new** String(coPasswordField.getPassword());

**int** p=1;

**int** p1=0;

**int** p2=0;

**int** p3=0;

**int** p4=0;

**int** p11=0;

**int** p22=0;

**int** p33=0;

p=s12.compareTo(s13);

p1=s1.compareTo("");

p2=s12.compareTo("");

p3=s8.compareTo("");

p4=s3.compareTo("");

p11=s1.compareTo("请输入五位ID英文码");

p22=s3.compareTo("请输入完整姓名");

p33=s8.compareTo("请输入6位邮编");

//s4;

**if**(xsdb.isSelected()) {s4=s4+"销售代表";};

**if**(scjl.isSelected()) {**if**(s4=="") {s4=s4+"市场经理";}**else** {s4=s4+"/"+"市场经理";}};

**if**(wz.isSelected()) {**if**(s4=="") {s4=s4+"物主";}**else** {s4=s4+"/"+"物主";}};

**if**(jsjl.isSelected()) {**if**(s4=="") {s4=s4+"结算经理";}**else** {s4=s4+"/"+"结算经理";}};

**if**(xsy.isSelected()) {**if**(s4=="") {s4=s4+"销售员";}**else** {s4=s4+"/"+"销售员";}};

//s6;

s6=(String)pro.getSelectedItem();

//s7;

s7=area.getText();

s7=s7.replace("地区:", "");

//s9;

s9=(String) guojia.getSelectedItem();

**if**(p==0&&p1!=0&&p2!=0&&p3!=0&&p4!=0&&p11!=0&&p22!=0&&p33!=0)

{

**int** c2=1,c3=1,c5=1,c10=1,c11=1;

c2=s2.compareTo("请以公司全称输入");

c5=s5.compareTo("详细地址请具体到街道单元号");

c10=s10.compareTo("建议添加区号");

c11=s11.compareTo("非必填项");

**int** c22=1,c33=1,c55=1,c1010=1,c1111=1;

c22=s2.compareTo("");

c33=s3.compareTo("");

c55=s5.compareTo("");

c1010=s10.compareTo("");

c1111=s11.compareTo("");

**if**(c2==0||c22==0) {s2=**null**;};

**if**(c5==0||c55==0) {s5=**null**;};

**if**(c10==0||c1010==0) {s10=**null**;};

**if**(c11==0||c1111==0) {s11=**null**;};

Connection conn=**null**;

PreparedStatement ps=**null**;

ResultSet resultSet=**null**;

String sql="insert into 客户(客户ID,公司名称,联系人姓名,联系人职务,地址,城市,地区,邮政编码,国家,电话,传真,密码) values (?,?,?,?,?,?,?,?,?,?,?,?);";

**try** {

conn=Conn.*getConnection*();

ps=conn.prepareStatement(sql);

ps.setString(1,s1);

ps.setString(2,s2);

ps.setString(3,s3);

ps.setString(4,s4);

ps.setString(5,s5);

ps.setString(6,s6);

ps.setString(7,s7);

ps.setString(8,s8);

ps.setString(9,s9);

ps.setString(10,s10);

ps.setString(11,s11);

ps.setString(12,s12);

ps.executeUpdate();

dialog tip=**new** dialog("注册成功", 's');

tip.jb1Button.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

card.show(jp0, "1");

tip.dispose();

}

});

tip.setVisible(**true**);

}**catch**(Exception e3){

dialog tip=**new** dialog("用户名已存在", 's');

tip.jb1Button.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

tip.dispose();

}

});

tip.setVisible(**true**);

}**finally** {

Conn.*close*(conn, ps, resultSet);

}

}

**else**

{

dialog tip=**new** dialog("用户名或密码或邮编设置有误", 's');

tip.jb1Button.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

tip.dispose();

}

});

tip.setVisible(**true**);

}

}

});

**（3）购物车删除功能**

shanchu.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

dialog tip=**new** dialog("确认删除此产品吗?",'c');

tip.jb1Button.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

**int** k=0;

String[] tmp2=**new** String[2];

tmp2[0]=shopStrings[*t*][0];

tmp2[1]=shopStrings[*t*][2];

**for**(k=0;k<*i*;k++)

{

**if**(k>=*t*)

{

shopStrings[k][0]=shopStrings[k+1][0];

shopStrings[k][1]=shopStrings[k+1][1];

shopStrings[k][2]=shopStrings[k+1][2];

}

}

**if**(k>*t*){*i*--;}

Connection conn=**null**;

PreparedStatement ps=**null**;

ResultSet resultSet=**null**;

**try** {

**int** m;

**int** n;

String a="";

conn=Conn.*getConnection*();

String sql="SELECT \* FROM 产品 where 产品名称=?;";

ps=conn.prepareStatement(sql);

ps.setString(1, tmp2[0]);

resultSet=ps.executeQuery();

**if**(resultSet.next())

{

m = Integer.*parseInt*(resultSet.getString("库存量"));

n = Integer.*parseInt*(tmp2[1]);

a=Integer.*toString*(m+n);

}

String sql2= "update 产品 set 库存量=? where 产品名称=?;";

PreparedStatement ps1;

ps1=conn.prepareStatement(sql2);

ps1.setString(1,a);

ps1.setString(2,tmp2[0]);

ps1.executeUpdate();

productNum.setText("剩余:"+a);

dialog tip1=**new** dialog("删除成功!", 's');

tip1.jb1Button.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

tip1.dispose();

tip.dispose();

tableModel.setDataVector(shopStrings, l1);

shanchu.setEnabled(**false**);

**int** k=0;

**double** price=0;

**int** num=0;

**int** sum=0;

**if**(*i*==0)

{

jiesuan.setEnabled(**false**);

}

**for**(k=0;k<*i*;k++)

{

price=Double.*valueOf*(shopStrings[k][1]).intValue();

num=Integer.*parseInt*(shopStrings[k][2]);

sum+=price\*num;

}

String sum1= String.*valueOf*(sum);

priceJTextField.setText(sum1+"元");

}

});

tip1.setVisible(**true**);

}**catch**(Exception e3){

e3.printStackTrace();

}**finally** {

Conn.*close*(conn, ps, resultSet);

}

}

});

**（4）设置发货日期**

senButton.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

JDialog changeDialog =**new** JDialog(**new** JFrame(),"发货设置",**true**);

changeDialog.setSize(300,200);

changeDialog.setLocationRelativeTo(**null**);

changeDialog.setLayout(**null**);

changeDialog.setResizable(**false**);

JLabel mon=**new** JLabel("月");

JLabel da=**new** JLabel("日");

JLabel sJLabel=**new** JLabel("交由");

JLabel s1JLabel=**new** JLabel("发货");

JLabel callJLabel=**new** JLabel("1123456111");

Choice monthChoice=**new** Choice();

Choice dayChoice=**new** Choice();

JComboBox deChoice=**new** JComboBox();

monthChoice.addItem("01");

monthChoice.addItem("02");

monthChoice.addItem("03");

monthChoice.addItem("04");

monthChoice.addItem("05");

monthChoice.addItem("06");

monthChoice.addItem("07");

monthChoice.addItem("08");

monthChoice.addItem("09");

monthChoice.addItem("10");

monthChoice.addItem("11");

monthChoice.addItem("12");

dayChoice.addItem("01");dayChoice.addItem("02");dayChoice.addItem("03");dayChoice.addItem("04");dayChoice.addItem("05");dayChoice.addItem("06"); dayChoice.addItem("07");dayChoice.addItem("08");dayChoice.addItem("09");dayChoice.addItem("10");

**for**(**int** i=10;i<31;i++)

{

dayChoice.addItem(""+i);

}

Connection conn=**null**;

PreparedStatement ps=**null**;

ResultSet rs=**null**;

String sql="select \* from 运货商";

**try**

{

conn=Conn.*getConnection*();

ps=conn.prepareStatement(sql);

rs=ps.executeQuery();

**while**(rs.next())

{

String a=rs.getString("公司名称");

deChoice.addItem(a);

}

}**catch**(Exception e1) {

e1.printStackTrace();

}

JButton confirmchangeButton=**new** JButton("确认发货");

confirmchangeButton.setBounds(85,110,120,30);

monthChoice.setBounds(10,30,60,40);dayChoice.setBounds(105,30,60,40);deChoice.setBounds(40,60,80,30); mon.setBounds(85,20,20,40);da.setBounds(175,20,40,40);sJLabel.setBounds(10,50,30,40);s1JLabel.setBounds(240,50,30,40); confirmchangeButton.setFont(jFont);callJLabel.setBounds(130,50,80,40); changeDialog.add(confirmchangeButton);changeDialog.add(mon);changeDialog.add(da);changeDialog.add(monthChoice); changeDialog.add(dayChoice);changeDialog.add(sJLabel);changeDialog.add(deChoice);changeDialog.add(s1JLabel);changeDialog.add(callJLabel);

**3.2.3SQL Server建立数据库**

建立以下8个基本表（表4-1至表4-8）

表4-1运货商表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 长度 | 字段类型 | 字段说明 |
| 运货商ID | 4 | int | 自动赋予新运货商的编号 |
| 公司名称 | 80 | nvarchar | 运货公司名称 |
| 电话 | 48 | nvarchar | 电话号码包括国家代号或区号 |

表4-2客户表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 长度 | 字段类型 | 字段说明 |
| 客户ID | 10 | nvarchar | 基于客户名称的 5 字符唯一代码 |
| 公司名称 | 80 | nvarchar | NULL |
| 联系人姓名 | 60 | nvarchar | NULL |
| 联系人职务 | 60 | nvarchar | NULL |
| 地址 | 120 | nvarchar | 街道或邮政信箱 |
| 城市 | 30 | nvarchar | NULL |
| 地区 | 30 | nvarchar | 州或省 |
| 邮政编码 | 20 | nvarchar | NULL |
| 国家 | 30 | nvarchar | NULL |
| 电话 | 48 | nvarchar | 电话号码包括国家代号或区号 |
| 传真 | 48 | nvarchar | 电话号码包括国家代号或区号 |

表4-3类别表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 长度 | 字段类型 | 字段说明 |
| 类别ID | 4 | int | 自动赋予新类别的编号 |
| 类别名称 | 30 | nvarchar | 食品类别名称 |
| 说明 | 16 | ntext | NULL |
| 图片 | 16 | image | 描绘食品类别的图片 |

表4-4供应商表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 长度 | 字段类型 | 字段说明 |
| 供应商ID | 4 | int | 自动赋予新供应商的编号。 |
| 公司名称 | 80 | nvarchar | NULL |
| 联系人姓名 | 60 | nvarchar | NULL |
| 联系人职务 | 60 | nvarchar | NULL |
| 地址 | 120 | nvarchar | 街道或邮政信箱 |
| 城市 | 30 | nvarchar | NULL |
| 地区 | 30 | nvarchar | 州或省 |
| 邮政编 码 | 20 | nvarchar | NULL |
| 国家 | 30 | nvarchar | NULL |
| 电话 | 48 | nvarchar | 电话号码包括国家代号或区 号 |
| 传真 | 48 | nvarchar | 电话号码包括国家代号或区 号 |
| 主页 | 16 | ntext | 供应商在 World Wide Web 上 的主页 |

表4-5产品表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 长度 | 字段类型 | 字段说明 |
| 产品ID | 4 | int | 自动赋予新产品的编号 |
| 产品名称 | 80 | nvarchar | NULL |
| 单位数量 | 40 | nvarchar | 自动赋予新产品的编号 |
| 单价 | 8 | money | NULL |
| 库存量 | 2 | smallint | NULL |
| 中止 | 1 | bit | 表示该产品暂停供货 |

表4-6订单表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 长度 | 字段类型 | 字段说明 |
| 订单 ID | 4 | int | 唯一订单编号 |
| 订购日期 | 8 | datetime | NULL |
| 发货日期 | 8 | datetime | NULL |
| 到货日期 | 8 | datetime | NULL |
| 货款确认日期 | 8 | datetime | NULL |
| 货主名称 | 80 | nvarchar | 接收货物的公司或人的名称 |
| 货主地址 | 120 | nvarchar | 仅为街道地址 -- 不允许为邮政信 |
| 货主城市 | 30 | nvarchar | NULL |
| 货主地区 | 30 | nvarchar | 州或省 |
| 货主邮政编码 | 20 | nvarchar | NULL |
| 货主国家 | 30 | nvarchar | NULL |

表4-7订单详情表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 长度 | 字段类型 | 字段说明 |
| 订单 ID | 4 | int | NULL |
| 产品 ID | 4 | int | NULL |
| 数量 | 2 | samllint | NULL |

根据数据库逻辑结构设计，编写SQL语句建立表。

具体实现代码如下：

/\*建立运货商表\*/

create table 运货商

(

运货商ID int,

公司名称 nvarchar(255),

电话 nvarchar(255)

)

/\*建立客户表\*/

create table 客户

(

客户ID nvarchar(255),

公司名称 nvarchar(255)，

联系人姓名 nvarchar(255),

联系人职务 nvarchar(255),

地址 nvarchar(255),

城市 nvarchar(255),

地区 nvarchar(255),

邮政编码 nvarchar(255),

国家 nvarchar(255),

电话 nvarchar(255),

传真 nvarchar(255),

密码 varchar(50)

)

/\*建立类别表\*/

create table 类别

(

类别ID int,

类别名称 nvarchar(255),

说明 ntext,

图片 image

)

/\*建立供应商表\*/

create table 供应商

(

供应商ID int,

公司名称 nvarchar(255),

联系人姓名 nvarchar(255),

联系人职务 nvarchar(255),

地址 nvarchar(255),

城市 nvarchar(255),

地区 nvarchar(255),

邮政编码 nvarchar(255),

国家 nvarchar(255),

电话 nvarchar(255),

传真 nvarchar(255),

主页ntext

)

/\*建立产品表\*/

create table 产品

(

产品ID int,

产品名称 nvarchar(255),

单位数量 nvarchar(255),

单价 money,

库存量 smallint,

中止 int

)

/\*建立订单表\*/

create table 订单

(

订单ID int,

订购日期 datetime,

发货日期 datetime,

到货日期 datetime,

货款确认日期 datetime,

货主名称 nvarchar(255),

货主地址 nvarchar(255),

货主城市 nvarchar(255),

货主地区 nvarchar(255),

货主邮政编码 nvarchar(255),

货主国家 nvarchar(255)

)

/\*建立订单明细表\*/

create table 订单明细

(

订单ID int，

产品ID int，

数量 smallint

)

/\*建立城市表\*/

Create table Sheet1

(

ID int,

省 varchar(50),

城市名 varchar(50),

地区 varchar（50）

）

从excel将测试数据导入到SQL Server数据库

以下列举产品表和客户表导入后的部分截图

产品表：



客户表：



**4 系统测试与应用**

4.1系统测试运行环境

测试系统环境：操作系统为Windows11，已安装SQL Server Management Studio 19。

测试系统硬件配置：处理器为Inter i7，内存16g。

4.2数据库测试与运行

下面展示本次数据库课程设计的主要功能测试。这里我注册客户ID为JDBC，密码为123的账户并登录进入系统，对主要功能进行演示。

4.2.1用户注册

填写客户ID为JDBC，密码为123，并完善相应的个人信息。

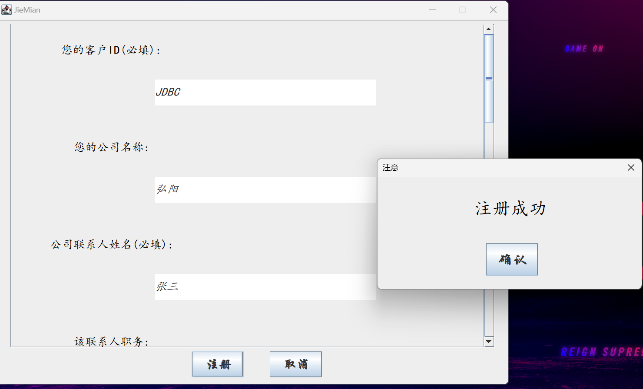


图4-1 用户注册界面、传入数据库的数据

4.2.2 用户登录

输入刚刚注册账户的客户ID和密码，登录进入客户购物系统



图4-2 用户登录界面

4.2.3用户搜索商品

输入产品ID或产品名称对数据库中的产品进行搜索,这里我对产品“牛奶”进行搜索。



图 4-3 产品搜索界面

4.2.4用户查看产品信息

此界面可显示产品“牛奶”的剩余量和单价信息牛奶还剩9箱，单价为19元。



图 4-4 查看数据库中产品数据

4.2.5用户添加商品到购物车并查看

用户可将要购买的一个或多个商品添加到购物车，并能在购物车界面结算。

这里为方便演示，我仅将2箱牛奶和1瓶番茄酱添加至购物车。



图 4-5 添加商品至购物车、购物车界面

4.2.6用户删除购物车商品

用户可对添加至购物车但又不想购买的商品进行删除操作。这里我将添加的1瓶番茄酱从购物车删除。

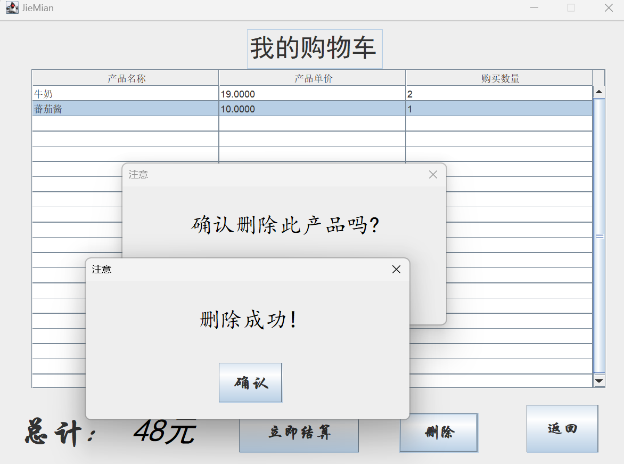


图 4-6 购物车删除商品

4.2.7用户查看个人信息

用户登录进入系统后，可在界面的左上角点击菜单“我的”，并点击子菜单项“个人信息”对自己的客户ID、公司名称、公司联系人姓名和职务、公司地址、公司所在国家、公司联系电话等信息进行查看。

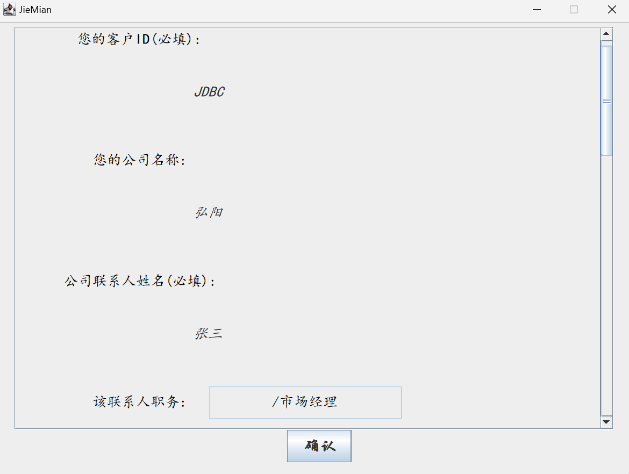


图 4-7 用户个人信息界面

4.2.8用户查看历史订单

用户登录进入系统后，可在界面的左上角点击菜单“我的”，并点击子菜单项“历史订单”对已支付订单的产品名称、产品单价、购买数量、购买日期、总金额进行查看

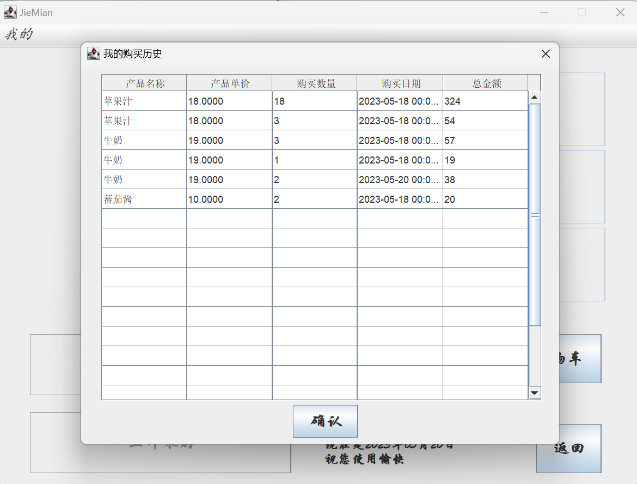


图 4-8 历史订单界面

4.2.9用户登出系统

用户在完成购物操作后，可在界面的左上角点击菜单“我的”，并点击子菜单项“登出”退出系统。

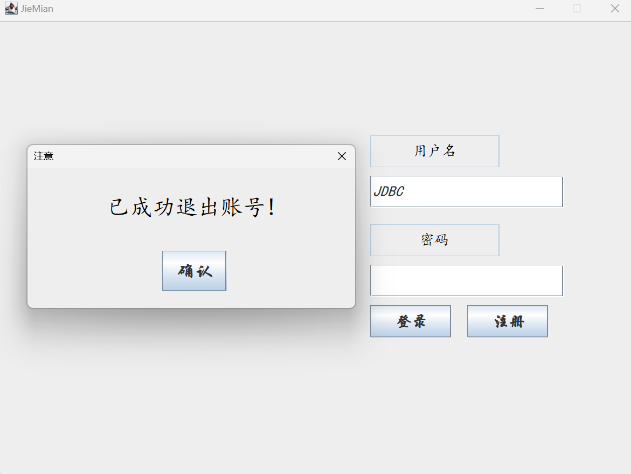


图 4-9 用户登出系统

4.2.10供应商接收新订单

用户下新订单后，会在供应商系统界面弹窗提示“您有新的订单”。



图 4-10 新订单提示弹窗

图形用户界面, 应用程序

描述已自动生成

图 4-11 数据库中发货日期为NULL的新订单

4.2.11供应商设置发货日期

供应商选中一个新订单，可看见下单日期，并对其发货日期进行设置。我这里选中上面客户端在5月20日下的订单“2箱牛奶”，并设置发货日期为5月21日，委托快递公司为急速快递。



图 4-12 发货日期设置界面

4.2.12供应商设置中止量

供应商可通过输入产品ID或产品名称对该产品的中止量进行设置。我这里对产品ID为2的牛奶设置中止量为2

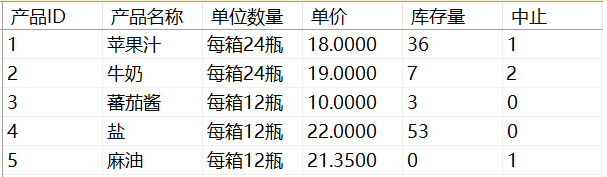
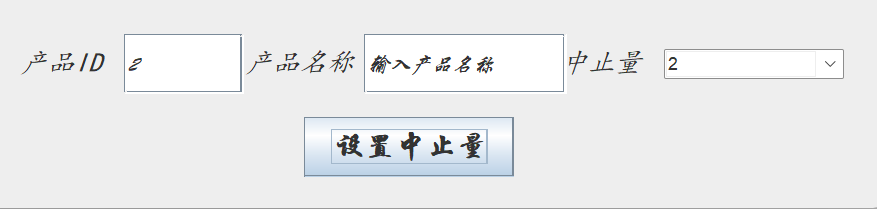


图 4-13 设置产品中止量、数据库产品信息变化

4.2.13供应商信息显示

在供应商系统界面左侧会显示出供应商店铺名、店铺地址、店铺邮编、店铺电话以及店铺联系人的姓名和职务信息。



图 4-14 店铺信息

4.2.14运货商未完成订单

运货商可在系统界面点击“未完成订单”按钮对未完成订单的订单ID、委托公司、运送产品和运送地区信息进行查看。



图4-15 未完成订单信息界面

4.2.15运货商确认签收

运货商在订单配送完成后，可选中已完成配送的订单，并勾选“确认”，完成签收。我这里选中上面客户端下的订单ID为10251的牛奶订单，并确认签收。



图 4-16 订单签收

4.2.16运货商已完成订单

运货商可在系统界面点击“已完成订单”按钮对已完成订单的订单ID、顾客姓名、需求产品名称和数量信息进行查看。



图 4-17 已完成订单信息界面

**5 总结与展望**

5.1总结

本系统实现了以下核心功能：

用户系统：

（1）登陆系统：用户通过输入用户名和密码可访问购物系统。注：只有数据库存在的用户才能进行登录操作，无密码用户输入用户名即可登录。

（2）注册系统：用户按照界面的提升输入必填个人信息后即可完成注册。

（3）搜索查看产品：用户通过输入数据库中已有的产品ID或产品名称，可查看到当前产品的单价、剩余量、产品图例，之后可进行添加产品至购物车或立即购买操作。

（4）加入购物车或立即采购：用户在查看产品界面设置购买数量后，可以将此产品加入购物车，购物车会显示所有商品的总价，也可以立即采购当即支付商品费用。

（5）购物车删除商品：用户在添加商品至购物车后，如果有不想要的商品，可直接选中并点击删除按钮即可从购物车删除。

（6）查看个人信息和历史订单：用户在登录后可点击左上角的“我的”菜单选项，对个人信息和历史订单进行查看。

（7）登出：用户在登录后可点击左上角的“我的”菜单选项，再点击子菜单中的登出即可推出系统。

供应商系统：

（1）店铺信息：在界面左侧可显示店铺名、店铺地址、店铺邮编、店铺电话以及店铺联系人的姓名和职位信息。

（2）发货系统：供应商可选择一项新订单，并对新订单的发货日期进行设置。

（3）设置中止量：供应商可输入产品ID或产品名称对其中止量进行设置。

运货商界面：

（1）签收系统：运货商可查看未完成订单的订单ID、委托公司、运送产品和运送地区，并在完成订单后对订单进行确认签收。

（2）查看已完成订单：运货商可在已完成订单中查看历史订单的订单ID、顾客姓名、需求产品名称和数量。

在这次课程设计中，我们小组分配的任务是设计一个网络购物系统，要面向消费者、供应商和运货商三端使用，这对我们来说是一次尝试与创新的过程，也可以说是一个挑战自己的过程。虽然经过了一个学期的数据库学习，但是还是感觉到实践之难，缺乏经验。现在我们利用上学期学到的知识设计并制作这个网络购物系统。在具体的设计与实践中，我们看到了一个管理系统从无到有的过程，从最开始的按照任务题目信息画出E-R图，到数据库创建表，再到用jdbc连接java和SQL Server数据库，最后到java写前端界面以及监听器，我们查阅了很多资料，这是一个困难的过程，但最后的成果也是令人满意的。

E-R图是本次课程设计的关键点，如果E-R图设计错误会严重影响之后的设计。E-R模型能方便地模拟研究对象的静态过程。E-R图能直观提供表示实体型、属性和联系的方法。在画E-R图的过程中，必须明确题目中的实体、属性和联系，用矩形、椭圆和菱形框出来。一个正确美观的E-R图能为后面的数据库设计建立基础。

创建表包括字段名、数据类型、主键的设计，必须弄清楚表中列的数据类型，是int、nvarchar、datetime、smallint、image型等等，还要设置合适的字符长度，还要就是考虑值是否可以为空。在这个过程中要注意表的列名是不能重复的，具有唯一性。主键的设置则相对容易，重点在于表中每列的数据类型不能设置错了，要小心谨慎。

通过对该网络购物管理系统的设计与开发，我们学习了前后端交互知识，它是原生的jdbc加载驱动连接数据库的，这些基础打牢了再去学习一些框架就会轻松许多，同时也对数据库的理论知识有了更深刻的理解，能更熟练的在java语言中使用SQL语句对数据库进行访问，也学会了如何使用java进行基础的图形化软件设计。在本次数据库课程设计中，我们小组成员团队协作，通过查阅资料，网上搜索等方式共同完成了本次课程设计，最后大家都收获颇丰。

5.2不足

本系统的基本核心功能都已经实现，但是整体上说仍然存在一些不足之处：

（1）用户使用时难以得知产品对应的产品ID，需要提前得知。

（2）数据库设计方面未考虑日志文件，以防止数据库发生故障后能够及时对数据库进行恢复。

（3）在数据库、方面未能完全考虑数据库的安全性与完整性问题。

（4）java swing难以做出美观的界面。

（5）未像淘宝、京东等购物平台一样设计评论功能，用户无法得到对商品的一个客观评价。

5.3 展望

本管理系统面向消费者、供应商和运货商使用，希望本小组设计的网络购物管理系统能为用户的生活带来便利。

本管理系统仅仅是我们对网购管理的第一次尝试，是一个测试版本。希望以后我们能在这次课程设计的基础上，设计出更加完善，更加美观，功能更加丰富的系统。

我们可以开发商品的评论功能，让买家能发表对商品的评价，让打算购物的消费者对商品的质量有一个基本了解。以获得更满意的购物体验的更进一步满足用户需求。数据库设计在未来会更加重视数据隐私和安全性，我们会更加谨慎得处理用户数据，确保其保密性和安全性。另外，随着人工智能和大数据分析技术的发展，数据库设计将会更加注重数据的可扩展性和可操作性。我们需要考虑如何优化数据查询、处理和存储，以便更好地支持数据分析和可视化。最后，数据库设计也将更加注重数据的标准话和一致性。我们需要按照一定的标准和规范来设计数据库，以便数据能更好地被理解和使用。

总之，我们会通过对系统的完善，让用户有一个更加舒适的使用体验，始终秉持面向用户、服务用户的创作理念。