长 沙 学 院

软件工程基础实训Ⅱ 实训

|  |  |
| --- | --- |
| **题目** | **基于C/S架构的餐饮管理系统** |
| **学院** | **计算机科学与工程** |
| **专业(班级)** | **软件工程（21软件01）** |
| **姓名** | **唐伟杰** |
| **学号** | **B20210304102** |
| **指导教师** | **杨刚、潘怡** |
| **起止日期** | **2023.06.05～2023.06.16** |

实训任务书

**课程名称：软件工程基础实训II**

**实训题目：基于C/S架构的餐饮管理系统**

### 已知技术参数和设计要求：

1. 问题描述（功能要求）：

本次案例将完成一个基于C/S架构的餐饮管理系统。

（1）.进行餐饮管理系统的功能设计。

（2）.设计餐饮管理系统的数据库。

（3）.利用JAVA的GUI类设计开发友好的人机接口。

（4）.利用JAVA语言设计监听用户操作和JDBC进行数据库的处理。

餐饮管理系统主要有6大模块：

（1）.员工管理：对员工实现增删改查。

（2）.客户管理：对客户实现增删改查。

（3）.餐台管理：对餐台实现增删改查。

（4）.菜品管理：对菜品分类、菜品实现增删改查。

（5）.点菜管理：服务员对某客户、某一空闲餐台实行开台，同时实现点菜，将餐台号与所点的菜品对应起来，分别显示出来，并记录开台时间。

（6）.结账管理：收银员对某一餐台通过统计显示消费的菜品清单统计出消费金额，通过手动输入实收金额进行找零的计算，并显示，完成结账的操作，并记录成统计数据。

基础实训II有2周时间40学时，学生基本必做任务：独立完成基于C/S架构的餐饮管理系统。

选做扩展：菜品推荐、销售统计。

**2. 运行环境要求：**

（1）客户端：

windows操作系统，JDK1.6，Idea或Eclipse。

（2）服务器：

MySQL5.6或 S 数据库服务器。

3. 技术要求：

1） 掌握软件工程的需求分析和系统设计方法。

2) 掌握数据库的分析与设计，完成餐饮管理系统数据库的设计。

3) 掌握JAVA的GUI设计方法，完成登录、主界面、菜品分类管理界面、菜品管理界面。

4） 掌握JAVA面向对象程序设计方法，设计系统所需的各种类。

5) 掌握JDBC方法，对数据库进行操作，完成主程序和菜品分类、菜品管理等功能。

### 实训工作量：

40课时

### 工作计划：

1.班级

21软件1-4

2.课时及教室安排

见实训课程工作计划表。

实训报告

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 学生姓名 | 唐伟杰 | 学号 | B202103042102 | 班级 | 21软件01 |
| 专业 | 软件工程 | | 指导教师姓名 | 杨刚、潘怡 | |
| 项目名称 | 基于C/S架构的餐饮管理系统 | | | | |
| **实训过程**：  需求分析阶段： 在这个阶段，学生需要对餐饮管理系统的功能进行分析和设计。首先，学生需要了解餐饮管理系统的基本功能，包括员工管理、客户管理、餐台管理、菜品管理、点菜管理和结账管理。然后，学生需要对每个功能进行详细的分析和设计，包括功能的输入、输出、处理流程和界面设计等。  系统设计阶段： 在这个阶段，学生需要对餐饮管理系统的数据库进行设计。首先，学生需要确定数据库的表结构和字段，包括员工表、客户表、餐台表、菜品表、点菜表和结账表等。然后，学生需要设计数据库的关系和索引，以提高系统的查询和处理效率。  编码实现阶段： 在这个阶段，学生需要使用JAVA语言和GUI类进行开发。首先，学生需要设计登录界面和主界面，以及各个功能模块的界面。然后，学生需要编写各个功能模块的JAVA类，包括员工管理类、客户管理类、餐台管理类、菜品管理类、点菜管理类和结账管理类等。最后，学生需要使用JDBC方法对数据库进行操作，实现各个功能模块的数据处理。  系统测试阶段： 在这个阶段，学生需要对餐饮管理系统进行测试，包括功能测试、性能测试和安全测试等。首先，学生需要对各个功能模块进行测试，确保系统的功能正常。然后，学生需要对系统的性能进行测试，包括查询和处理效率等。最后，学生需要对系统的安全性进行测试，确保系统的数据安全和用户权限控制等。  交付实施阶段： 在这个阶段，学生需要将餐饮管理系统交付给客户，并进行实施。首先，学生需要对系统进行打包和部署，以便客户可以方便地安装和使用系统。然后，学生需要对客户进行培训，以便客户可以熟练地使用系统。最后，学生需要对系统进行维护和升级，以保证系统的稳定性和可靠性。 | | | | | |
| **实训成果**：  一、概述 本文档是基于C/S架构的餐饮管理系统的设计说明书，主要包括系统需求分析、系统设计、数据库设计、界面设计、程序实现等内容。  二、系统需求分析 本系统主要包括员工管理、客户管理、餐台管理、菜品管理、点菜管理和结账管理六大模块。具体功能如下：  1. 员工管理 员工管理模块主要包括员工信息的增删改查功能，包括员工号、姓名、性别、年龄、电话、职位等信息。  2. 客户管理 客户管理模块主要包括客户信息的增删改查功能，包括客户号、姓名、性别、电话、会员卡号等信息。  3. 餐台管理 餐台管理模块主要包括餐台信息的增删改查功能，包括餐台编号、可供就餐人数、餐台状态等信息。  4. 菜品管理 菜品管理模块主要包括菜品分类和菜品信息的增删改查功能，分为菜品类和菜品两个小模块，菜品类包括菜类编号、菜类名称、描述，菜品包括菜品名称、菜品号、菜品分类、单位、菜品价格、状态等信息。  5. 订单管理 订单模块主要订单进行管理、点菜管理以及结账管理，点菜管理主要包括对某某空闲餐台实行开台，同时实现点菜，将餐台号与所点的菜品对应起来，分别显示出来，并记录开台时间。 结账管理模块主要包括收银员对某一餐台通过统计显示消费的菜品清单统计出消费金额，通过手动输入实收金额进行找零的计算，并显示，完成结账的操作，并记录成统计数据。  三、系统设计 本系统采用C/S架构，客户端采用JAVA语言开发，服务器采用MySQL数据库。系统主要分为客户端和服务器两部分，客户端主要负责用户交互和数据展示，服务器主要负责数据存储和处理。  客户端设计：  1. 客户端主要包括登录界面、主界面，主界面内含员工管理、客户管理、餐台管理、菜品管理（内含菜品类管理和菜品管理界面）、订单管里五张任务卡、用于实操的Dialog等界面。   1. 服务器设计 服务器主要负责数据存储和处理，采用MySQL数据库存储数据。数据库主要包括员工表、客户表、餐台表、菜品分类表、菜品表、订单表等表。   数据库设计 本系统采用MySQL数据库，主要包括员工表、客户表、餐台表、菜品分类表、菜品表、订单表等表。   1. 员工表 员工表主要包括员工号、姓名、性别、年龄、电话、职位等字段。 2. 客户表 客户表主要包括客户号、姓名、性别、联系方式、会员卡号等字段。 3. 餐台表 餐台表主要包括餐台号、可供就餐人数、餐台状态等字段。 4. 菜品分类表 菜品分类表主要包括菜类号、菜品分类名称、描述等字段。 5. 菜品表 菜品表主要包括菜品名、菜品号、菜类名、单位、菜品价格、销售状态等字段。 6. 订单表 订单表主要包括订单号、餐台编号、服务员编号、客户编号、开台时间、结账时间、消费金额等字段。   四、界面设计 本系统采用JAVA的GUI类设计开发友好的人机接口，主要包括登录界面、主界面、员工管理界面、客户管理界面、餐台管理界面、菜品分类管理界面、菜品管理界面、点菜界面和结账界面等界面。  五、程序实现 本系统采用JAVA语言设计监听用户操作和JDBC进行数据库的处理。主要包括客户端和服务器两部分，客户端主要负责用户交互和数据展示，服务器主要负责数据存储和处理。  六、总结 本文档详细介绍了基于C/S架构的餐饮管理系统的设计过程，包括系统需求分析、系统设计、数据库设计、界面设计、程序实现等内容。通过本次实训，学生们掌握了软件工程的需求分析和系统设计方法，掌握了数据库的分析与设计，掌握了JAVA的GUI设计方法，掌握了JAVA面向对象程序设计方法，掌握了JDBC方法，对数据库进行操作，完成主程序和菜品分类、菜品管理等功能。 | | | | | |
| **实训总结**：  在本次实训中，我学习了软件工程的需求分析和系统设计方法，掌握了数据库的分析与设计，完成了餐饮管理系统数据库的设计。同时，我也掌握了JAVA的GUI设计方法，学会了登录界面和界面排版，菜单栏的设计。在JAVA面向对象程序设计方法方面，我设计了系统所需的各种类，并掌握了JDBC方法，对数据库进行操作，完成 了主程序和菜品分类、菜品管理等功能。还有数据库的链接，并且一些MySQL语句的使用，在这方面掌握的还比较好，通过调用表里面的东西，存进表里面等等操作。 在实训过程中，查询了资料和网络上的文献，虽然出现了非常多的错误，但是通过不断的尝试和学习，我最终完成了任务。 今后实训，我应该加快一些基本的操作，花更多的时间在关键代码上和算法上，和同学一起讨论更好的方法，得到最好的、自己能够完成的代码。 | | | | | |

基于C/S架构的餐饮管理系统

设计说明书

作者：唐伟杰 B20210304102

计算机科学与工程学院

2023年 06 月 15日

**摘要**

一、引言 餐饮管理系统是一种基于计算机技术的管理系统，它可以帮助餐饮企业实现对餐饮业务的全面管理，包括餐厅管理、菜品管理、库存管理、订单管理等。本文将介绍一种基于C/S架构的餐饮管理系统的设计方案。

二、系统架构 本系统采用C/S架构，客户端使用Java Swing开发，服务端使用Java Socket编程实现。客户端与服务端通过Socket通信进行数据传输。

三、系统功能

1. 登录功能：用户输入用户名和密码进行登录，系统验证用户信息后进入主界面。

2. 信息管理：管理员可以添加、修改、删除任意模块信息。

3. 订单管理：管理员可以查看订单信息，包括订单编号、下单时间、订单状态等。

4. 点菜管理：管理员可以选择餐桌进行点菜。

5. 支付功能：客户可以通过系统进行在线支付。

四、系统界面

1. 登录界面：用户输入用户名和密码进行登录。

2. 主界面：包括员工管理、客户管理、餐台管理、菜品管理、订单管理等界面等功能。

3. 菜品管理界面：管理员可以添加、修改、删除菜品信息。

4. 员工管理 员工管理模块主要包括员工信息的增删改查功能，包括员工号、姓名、性别、年龄、电话、职位等信息。

5.客户管理 客户管理模块主要包括客户信息的增删改查功能，包括客户号、姓名、性别、电话、会员卡号等信息。

6. 餐台管理 餐台管理模块主要包括餐台信息的增删改查功能，包括餐台编号、可供就餐人数、餐台状态等信息。

7. 菜品管理 菜品管理模块主要包括菜品分类和菜品信息的增删改查功能，分为菜品类和菜品两个小模块，菜品类包括菜类编号、菜类名称、描述，菜品包括菜品名称、菜品号、菜品分类、单位、菜品价格、状态等信息。

8. 订单管理 订单模块主要订单进行管理、点菜管理以及结账管理，点菜管理主要包括对某某空闲餐台实行开台，同时实现点菜，将餐台号与所点的菜品对应起来，分别显示出来，并记录开台时间。 结账管理模块主要包括收银员对某一餐台通过统计显示消费的菜品清单统计出消费金额，通过手动输入实收金额进行找零的计算，并显示，完成结账的操作，并记录成统计数据。

五、系统流程

1. 用户登录系统。

2. 管理员进入主界面，选择员工、客户、餐台、菜品、订单对应任务卡。

3. 客户进入任务卡，进行信息修改，或点餐、支付等操作。

4. 系统根据用户操作进行相应的数据处理和更新。

5. 用户退出系统。

六、总结 本文介绍了一种基于C/S架构的餐饮管理系统的设计方案，包括系统架构、功能、界面和流程等。该系统可以帮助餐饮企业实现对餐饮业务的全面管理，提高管理效率和客户满意度。

**关键词：**java ，C/S架构，餐饮管理系统

目录

[1 引言 1](#_Toc11882504)

[1.1 编写目的 1](#_Toc11882505)

[1.2 参考资料 1](#_Toc11882506)

[2 需求规约 2](#_Toc11882507)

[2.1 功能需求 2](#_Toc11882508)

[2.2 界面需求 4](#_Toc11882509)

[2.3 数据需求 4](#_Toc11882510)

[3 系统设计 5](#_Toc11882511)

[3.1 运行环境 8](#_Toc11882512)

[3.2 系统静态结构设计 8](#_Toc11882513)

[3.3 人机接口设计 9](#_Toc11882514)

[3.4 XXX模块设计说明 15](#_Toc11882515)

[3.5 XXX模块设计说明 17](#_Toc11882516)

[4 数据库设计 18](#_Toc11882517)

[4.1 数据库环境说明 18](#_Toc11882518)

[4.2 数据库的命名规则 18](#_Toc11882519)

[4.3 逻辑结构设计 18](#_Toc11882520)

[4.4 物理结构设计 19](#_Toc11882521)

[4.5 安全性设计 21](#_Toc11882522)

[5 测试用例设计 22](#_Toc11882523)

[附录 23](#_Toc11882524)

[附录1 程序运行结果 2](#_Toc11882525)3

[附录2 程序源代码 24](#_Toc11882526)

# 1 引言

## 1.1 编写目的

本文档是“餐饮管理系统”的软件设计说明书。编写目的是：明确软件需求、概要设计、详细设计和功能测试用例，为软件后续开发和维护提供指导。本文档的读者对象是需求分析人员、系统设计人员、系统开发人员、系统测试人员。

## 1.2 参考资料

[1] 《软件工程》（第8版），Ian Sommerville著，机械工业出版社，2015年。

[SQL Server 2016权威指南》（第2版），Brian Larson著，人民邮电出版社，2017年。

[java高级程序设计》（第2版 微课版）人民邮电出版社

# 2 需求规约

## 2.1 功能需求

本系统主要有以下6大模块：

（1）员工管理：对员工实现增删改查。

（2）客户管理：对客户实现增删改查。

（3）餐台管理：对餐台实现增删改查。

（4）菜品管理：对菜品分类、菜品实现增删改查。

（5）订单管理：分点菜管理和结账管理，点菜管理：对某一空闲餐台实行开台，同时实现点菜，将餐台号与所点的菜品对应起来，分别显示出来，并记录开台时间。 结账管理：收银员对某一餐台通过统计显示消费的菜品清单统计出消费金额，通过手动输入实收金额进行找零的计算，并显示，完成结账的操作，记录数据。

2.1.1 员工管理用例说明

用例名称：员工管理

用例描述：管理员对餐饮员工进行增删改查。

前置条件：必须是本店员工。

活动步骤：

点击新增，输入员工信息点击添加即可。

选中要修改的员工数据，直接修改即可。

选中要删除的员工，点击删除即可。

在输入框输入需要查询的员工编号。

2.1.2 点菜管理用例说明

用例名称：点菜管理

用例描述：客户就餐进行开台点菜。

前置条件：需要先在空闲餐台进行开台

活动步骤：

1. 先在空闲餐台进行开台
2. 点击点菜，进入点菜
3. 选中菜品，进行添加

异常处理：

如果餐台不为空闲和餐台不存在就弹框。 添加空菜品也弹框。

2.1.3 餐台管理用例说明

用例名称：餐台管理

用例描述：管理员对餐饮餐台进行增删改查。

活动步骤：

选中空白行，输入餐台信息点击添加即可。

选中要修改的餐台数据，直接修改即可。

选中要删除的餐 点击删除即可。

在输入框输入需要查询的餐台编号。

扩展点：

无

## 2.2 界面需求

界面友好，交互友好，符合人们审美要求。

## 2.3 数据需求

表2.1 用户实体

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Id | 整形 | 8 | 否 | 是 | 用户编号 |
| Username | 字符串 | 32 | 否 | 否 | 用户账户 |
| Password | 字符串 | 32 | 否 | 否 | 用户密码 |

表2.2 员工实体

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Id | 整形 | 8 | 否 | 是 | 编号 |
| Name | 字符串 | 32 | 否 | 否 | 姓名 |
| Sex | 字符串 | 32 | 否 | 否 | 性别 |
| Age | 字符串 | 32 | 否 | 否 | 年龄 |
| Telephone | 字符串 | 32 | 否 | 否 | 电话 |
| Position | 字符串 | 323 | 否 | 否 | 职位 |

表2.2 员工实体

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Id | 整形 | 8 | 否 | 是 | 编号 |
| Name | 字符串 | 32 | 否 | 否 | 姓名 |
| Sex | 字符串 | 32 | 否 | 否 | 性别 |
| Telephone | 字符串 | 32 | 否 | 否 | 电话 |
| Cardid | 字符串 | 32 | 否 | 否 | 会员卡号 |

表2.3 餐台实体

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Id | 整形 | 32 | 否 | 否 | 餐台编号 |
| seating | 字符串 | 32 | 否 | 否 | 餐台座位 |
| Status | 字符串 | 32 | 否 | 否 | 餐台状态 |

表2.4菜类实体

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Id | 整形 | 8 | 否 | 是 | 餐台序号 |
| Name | 字符串 | 32 | 否 | 否 | 菜类名 |
| Describe | 字符串 | 32 | 否 | 否 | 描述 |

表2.6菜品实体

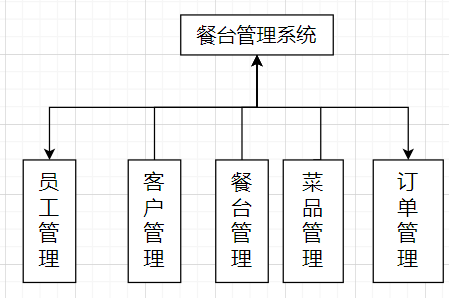
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Name | 字符串 | 32 | 否 | 否 | 菜品名 |
| Id | 整形 | 8 | 否 | 是 | 菜品代号 |
| Category | 字符串 | 32 | 否 | 否 | 菜类名 |
| Unit | 字符串 | 32 | 否 | 否 | 单位 |
| Price | 浮点数 | 16 | 否 | 否 | 价格 |
| Status | 字符串 | 32 | 否 | 否 | 状态 |

表2.6订单实体

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Id | 字符串 | 32 | 否 | 是 | 订单号 |
| Tableid | 整形 | 8 | 否 | 否 | 餐台号 |
| Category | 字符串 | 32 | 否 | 否 | 菜类名 |
| Cuisine | 字符串 | 32 | 否 | 否 | 菜品名 |
| Price | 浮点数 | 16 | 否 | 否 | 价格 |
| Status | 字符串 | 32 | 否 | 否 | 状态 |

# 3 系统设计

因为员工管理和客户管理以及餐台管理，菜品管理是个部分的设置都仅仅涉及增删改查，在大部分方面存在相同，于是时设计思路仅给出员工管理的思路。





##### 员工管理（客户管理、餐台管理、菜品管理原理相同）：

首先在jframe窗体上放置2块面板，占据上中2个位置，在上部，添加上四个按钮和一个查询框，在中部，添加一个table表格，里面存放员工数据，并在一开始就刷新表格数据（查询所有的数据，遍历添加至表格）

等。此为界面设计。以下为具体功能设计：

##### 添加功能：

在空白行写入需要添加的员工参数（id，name等等）否则弹出此人以存在。

点击添加按钮后，会弹出Dialog界面，输入新增的信息

将输入信息封装入一个employee实体，传入数据库

重置表格。

##### 查询功能：

在文本框中输入要查询的内容

点击查询，弹出dialog界面，展示信息

##### 修改功能：

选中修改的行

点击修改按钮后，会弹出Dialog界面，对需要修改的数据人为更新

将输入信息封装入一个employee实体，传入数据库

重置表格。

##### 删除功能：

选中要删除的员工（行），点击删除按钮，删除行

，在重置表格，将集合内容添加到表格里面。

点菜管理：

点菜之前，需要先在空闲的餐桌开台，只有开台后，才能继续点菜和结账功能。已经处于就餐状态的餐桌无法再次开台。

只有存在的餐桌才能开台。

##### 开台功能：

首先调用餐台管理的list方法展示所有的餐台信息。（和员工管理无区别，此处不再解释）。

员工编号：利用employeeservice的listemployeeByParams方法，查询所有的员工信息。

然后遍历查询到的员工信息，将其添加到下拉框中。

客户编号:和员工编号相同。

点击开台按钮：

先获取选中行的行号，然后根据此行号获取该行的各列信息，创建餐台的实体对象，传入该行餐台名字以及状态为就餐，判断该行是否为空，如果为空则弹出没这个位置。如果不为空，先调用deskservice的updates方法修改餐台状态为就餐，如果成功就弹出开台成功，否则此位置已经有人了，然后创建opencord的实体对象，传入该餐台的名字，将下拉框的选中内容以及当前时间封装到该实体对象，

调用opencordservice的insert方法，将内容插入到开台表中，然后重新刷新表格。就此开台完毕。

##### 点菜功能：

选中要点菜的餐台，点击点菜，跳转到点菜界面。 注意：如果在未开台的餐台点菜和空餐台点菜会弹出此位置已经有人。

点击点菜，获取当前鼠标选中的行，获取行号，根据行号获取餐台no以及状态，如果获取到的是空就弹出没这个座位，如果不为空值，就判断餐台状态，如果状态不为就餐就弹出此位置还没人，否则就传入此餐台no创建点菜页面。

点菜：

左边为菜品表，鼠标选中左边的一行，点击选中该菜品，然后在右边填入数量（或者直接点击添加，默认数量为1），点击添加即可，如果不想要某个，选中此行，点击删除即可。

显示俩个表格的操作为此俩个方法（与上面无区别）。

选择菜：

选中想要选中的行，点击选择按钮，根据选择的行，得到该行的菜品列数据，如果右边没有选中任何行，就遍历右边的表格，一直到没有数据的那一行，将左边读取到的菜品数据加入。

添加菜：

选择只有菜品名字的行，填入数量，点击添加按钮，或者直接点击添加，（默认数量为一）。

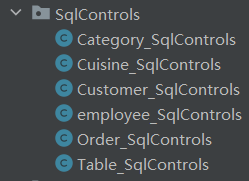
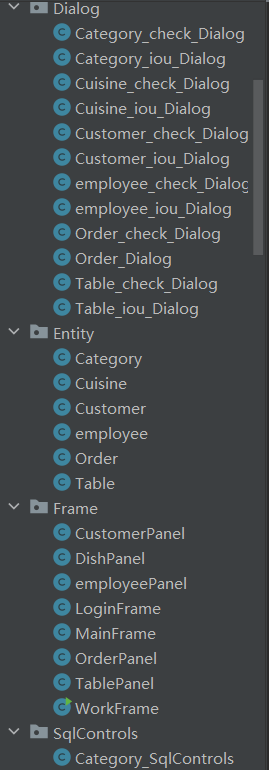
添加菜，自动生成序号，封装开台界面传入的餐台no以及根据菜品名字查到的价格，以及在表格上的数据，插入到点菜表。

## 3.1 运行环境

客户端：Windows操作系统，JDK1.8，Idea或Eclipse。

服务器：MySQL5.6或SQL Server数据库服务器。

## 3.2 系统静态结构设计



系统静态结构说明

本系统采用C/S架构，客户端采用Java Swing实现，服务器采用MySQL WorkBench数据库。 客户端主要包括以下类：

（1）LoginFrame：登录界面。

（2）MainFrame：主界面，包括各个模块的功能按钮。

（3）EmployeePanel：员工管理界面。

（4）CustomerPanel：客户管理界面。

（5）TablePanel：餐台管理界面。

（6）DishPanel：菜类菜品管理界面

（7）OrderPanel：点菜界面。

服务器主要包括以下类：

（1）Employee\_Sqlcontrols：员工数据访问对象。

（2）Customer\_Sqlcontrols：客户数据访问对象。

（3）Table\_Sqlcontrols：餐台数据访问对象。

（4）Category\_Sqlcontrols：菜品分类数据访问对象。

（5）Cuisine\_Sqlcontrols：菜品数据访问对象。

（6）Order\_Sqlcontrols：订单数据访问对象。

## 3.3 人机接口设计

（1）登录界面：包括用户名和密码输入框、登录按钮。

（2）主界面：包括员工管理、客户管理、餐台管理、菜品管理、订单管理等功能按钮。

（3）员工管理界面：包括员工列表、增加员工、删除员工、修改员工和查询员工等功能按钮。

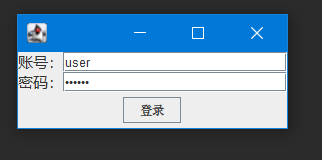
（4）客户管理界面：包括客户列表、增加客户、删除客户、修改客户和查询客户等功能按钮。

（5）餐台管理界面：包括餐台列表、增加餐台、删除餐台、修改餐台和查询餐台等功能按钮。

（6）菜品管理界面：分菜品和菜类，菜品分类管理：包括菜品分类列表、增加菜品分类、删除菜品分类、修改菜品分类和查询菜品分类等功能按钮。 菜品管理：包括菜品列表、增加菜品、删除菜品、修改菜品和查询菜品等功能按钮。

（7）订单管理界面：包括订单列表、餐台开台点菜、餐台查询、餐台结账和刷新表等功能按钮。

3.3.1 登录设计



输入账号密码，完成登录

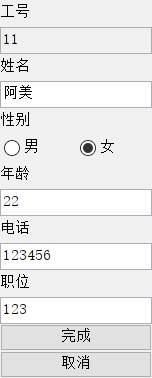
3.3.1 员工管理设计（其余仅涉及增删改查与员工管理无异）



点击新增弹出Dialog界面，输入信息完成新增，工号重复不予新建



在文本框上输入工号，弹出Dialog界面，展示内容



选中行，修改信息，即可完成修改



选中行，直接删除改行

3.3.3 订单管理设计（点菜与结账）



点菜管理：向文本框输入餐台号，点击点菜管理，弹出Dialog界面，Dialog界面中选中左侧菜品，点击下单即可向右侧预览订单表输入已选菜品，选中右侧表中内容点击退订可移除菜品，点击完成可将右表所有数据生成order实体并输入数据库，完成点菜



查询订单：向文本框输入餐台号，点击查询管理，弹出Dialog界面，展示该餐台未支付的订单内容



结账管理：向文本框输入餐台号，点击查询管理，弹出Dialog界面，展示该餐台未支付的订单内容，

下方文本框输入金额，弹出找零提示框

显示未支付：只展示状态为未支付的订单

刷新：显示所有订单

## 3.4 员工管理模块设计说明

首先在jPanel窗体上放置3块面板，占据上中2个位置，在上部，添加上一个查询框和4个按钮，分别是添加、查询、修改和删除，在中部，添加一个table表格，里面存放员工数据，并在一开始就刷新表格数据（查询所有的数据，遍历添加至表格）。

此为界面设计。以下为具体功能设计：

##### 添加功能：

点击新增弹出Dialog界面，输入信息完成新增，工号重复不予新建

向Dialog界面的文本框输入对应内容

点击完成按钮后，将所有内容封装到employee实体类里面，然后调用Sql\_controls中的employee\_controls中的employee\_insert方法将此行数据输入数据库

调用brush方法重置表格，将集合内容添加到表格里面。

##### 查询功能：

在文本框内填入一个需要查询的员工id，点击查询，获取文本框输入的员工id并弹出Dialog界面

如果获取到的内容为空，弹出提示框，提示输入要查询的工号

调用Sql\_controls中的employee\_controls中的employee\_check,将查询到的结果封装成employee实体并分别调用get方法输入dialog界面中对应文本框

##### 修改功能:

选中要删除的员工（行），获取各属性

点击修改，弹出Dialog界面，并将选中行各属性数据输入Dialog界面中对应文本框

在Dialog的文本框内修改数据

点击完成按钮后，将所有内容封装到employee实体类里面，然后调用Sql\_controls中的employee\_controls中的employee\_update方法将此行数据输入数据库

调用brush方法重置表格

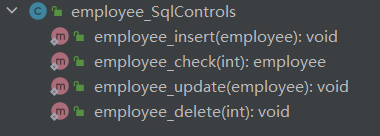
##### 删除功能：

选中要删除的员工（行），点击删除按钮，根据鼠标选中行，获取该行该列的id值

直接调用Sql\_controls中的employee\_controls中的employee\_delete方法直接删除

调用brush方法重置表格

Employee\_controls方法如下



## 3.5点菜模块设计说明（点菜与结账）

首先在jPanel窗体上放置3块面板，占据上中2个位置，在上部，添加上一个查询框和5个按钮，分别是点菜管理、查询订单、结账管理、显示未支付、刷新，在中部，添加一个table表格，里面存放订单数据，并在一开始就刷新表格数据（查询所有的数据，遍历添加至表格）。

此为界面设计。以下为具体功能设计：

##### 点菜功能：

向文本框输入餐台号，点击点菜管理，弹出Dialog界面

Dialog界面中选中左侧菜品，点击下单即可向右侧预览订单表输入已选菜品

选中右侧表中内容点击退订可移除菜品

点击完成可将右表所有菜品分别生成order实体并for循环调用Sql\_controls中的order\_controls中的order\_insert方法将所有生成的order实体数据输入数据库

调用brush方法重置表格，将集合内容添加到表格里面。

调用Sql\_controls中的order\_controls中的order\_table\_update方法修改餐台状态

调用Brush方法刷新表

查询订单：

向文本框输入餐台号，点击查询管理，弹出Dialog界面

调用Sql\_controls中的order\_controls中的order\_check方法生成model

生成Jtable展示该餐台未支付的订单内容

结账管理：

向文本框输入餐台号，点击查询管理，弹出Dialog界面

调用Sql\_controls中的order\_controls中的order\_check方法生成model

生成Jtable展示该餐台未支付的订单内容

向Dialog界面下方文本框输入金额

获取文本框数据，若不足则弹出提示框表示金额不足，否则弹出提示框显示找零

调用Sql\_controls中的order\_controls中的had\_paid方法修改餐台状态

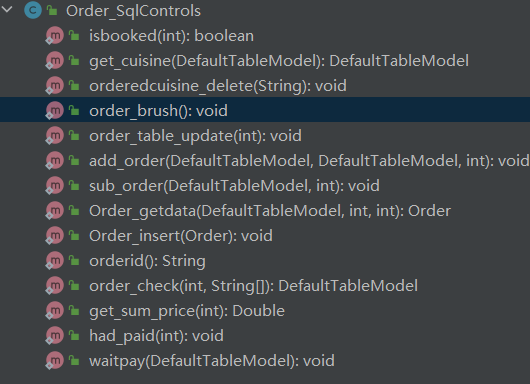
调用Brush方法刷新表

显示未支付：

调用Sql\_controls中的order\_controls中的waitpay方法刷新表

刷新：

调用Brush\_table方法刷新表



4 数据库设计

## 4.1 数据库环境说明

本系统采用MySQL WorkBench数据库服务器。

## 4.2 数据库的命名规则

（1）表名：采用小写字母，单词之间不分隔，如employee、customer、desk等。

（2）字段名：采用小写字母，如id、name等。

## 4.3 逻辑结构设计

（1）员工表（employee）：包括员工编号、姓名、性别、年龄、职位、联系方式等字段。

（2）客户表（customer）：包括客户编号、姓名、性别、联系方式、会员卡号等字段。

（3）餐台表（desk）：包括餐台编号、可供就餐人数、状态等字段。

（4）菜品分类表（category）：包括分类编号、分类名称、描述等字段。

（5）菜品表（cuisine）：包括菜品名称、菜品编号、菜类名称、单位、价格、销售状态等字段。

（6）订单表（order）：包括订单编号、餐台号、菜类名称、菜品名称、消费金额、支付状态等字段。

## 4.4 物理结构设计

4.4.1 数据库表

说明数据库的表的设计。

表4.1 表汇总

|  |  |
| --- | --- |
| 表 名 | 功 能 说 明 |
| employee | 进行员工管理 |
| Customer | 进行客户管理 |
| Desk | 进行餐台管理 |
| Category | 进行菜类管理 |
| Cuisine | 进行菜品管理 |
| Order | 进行订单管理 |

表2.1 用户表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Id | 整形 | 8 | 否 | 是 | 用户编号 |
| Username | 字符串 | 32 | 否 | 否 | 用户账户 |
| Password | 字符串 | 32 | 否 | 否 | 用户密码 |

表2.2 员工表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Id | 整形 | 8 | 否 | 是 | 编号 |
| Name | 字符串 | 32 | 否 | 否 | 姓名 |
| Sex | 字符串 | 32 | 否 | 否 | 性别 |
| Age | 字符串 | 32 | 否 | 否 | 年龄 |
| Telephone | 字符串 | 32 | 否 | 否 | 电话 |
| Position | 字符串 | 323 | 否 | 否 | 职位 |

表2.2 员工表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Id | 整形 | 8 | 否 | 是 | 编号 |
| Name | 字符串 | 32 | 否 | 否 | 姓名 |
| Sex | 字符串 | 32 | 否 | 否 | 性别 |
| Telephone | 字符串 | 32 | 否 | 否 | 电话 |
| Cardid | 字符串 | 32 | 否 | 否 | 会员卡号 |

表2.3 餐台表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Id | 整形 | 32 | 否 | 否 | 餐台编号 |
| seating | 字符串 | 32 | 否 | 否 | 餐台座位 |
| Status | 字符串 | 32 | 否 | 否 | 餐台状态 |

表2.4菜类表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Id | 整形 | 8 | 否 | 是 | 餐台序号 |
| Name | 字符串 | 32 | 否 | 否 | 菜类名 |
| Describe | 字符串 | 32 | 否 | 否 | 描述 |

表2.6菜品表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Name | 字符串 | 32 | 否 | 否 | 菜品名 |
| Id | 整形 | 8 | 否 | 是 | 菜品代号 |
| Category | 字符串 | 32 | 否 | 否 | 菜类名 |
| Unit | 字符串 | 32 | 否 | 否 | 单位 |
| Price | 浮点数 | 16 | 否 | 否 | 价格 |
| Status | 字符串 | 32 | 否 | 否 | 状态 |

表2.6订单表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 属性名称 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| Id | 字符串 | 32 | 否 | 是 | 订单号 |
| Tableid | 整形 | 8 | 否 | 否 | 餐台号 |
| Category | 字符串 | 32 | 否 | 否 | 菜类名 |
| Cuisine | 字符串 | 32 | 否 | 否 | 菜品名 |
| Price | 浮点数 | 16 | 否 | 否 | 价格 |
| Status | 字符串 | 32 | 否 | 否 | 状态 |

# 5 测试用例设计

表5.1 登录测试用例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 功能 | 用例编号 | 操作步骤  （输入条件） | 测试数据 | 预期结果 | 实际结果/判定 |
| 登录 | yl-001 | 输入‘用户名’和‘密码’ | 李康，123456 | 登录成功 | 通过 |
| yl-002 | admin，1234567 | 登录失败 | 通过 |
| yl-003 | admin1，123456 | 登录失败 | 通过 |
| yl-004 | admin,空 | 登录失败 | 通过 |
| yl-005 | 空,123456 | 登录失败 | 通过 |
| yl-006 | admin,一个空格 | 登录失败 | 通过 |
| yl-007 | 一个空格，123456 | 登录失败 | 通过 |
| yl-008 | admin，密码长度超过32 | 登录失败 | 通过 |
| yl-009 | 用户名超过36，123456 | 登录失败 | 通过 |

表5.2 员工管理测试用例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 功能 | 用例编号 | 操作步骤  （输入条件） | 测试数据 | 预期结果 | 实际结果/判定 |
| 增删改查 | yl-001 | 输入员工数据  修改员工数据  查询员工数据  删除员工数据 | 13 熙哥 男 2003-03-03 111 湖南 222 扫厕所 是 | 成功 | 通过 |
| yl-002 | 13 熙哥 男 2003-03-03 111 湖南 222 扫厕所 是 | 失败 | 通过 |
| yl-003 | 1 | 成功 | 通过 |
| yl-004 | 2 | 成功 | 通过 |
| yl-005 | 3 | 成功 | 通过 |
| yl-006 | 4 | 成功 | 通过 |
| yl-007 | 1 | 成功 | 通过 |
| yl-008 | 空 | 失败 | 通过 |
| yl-009 | 2 | 成功 | 通过 |

# 附录1 程序运行结果

（其余仅涉及增删改查与员工管理无异）



## 附录2 程序源代码

**登录界面**

package Restaurant\_system.Frame;

import database.BookDb;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.\*;

public class LoginFrame extends JFrame {

boolean isLive = true;

JLabel account = new JLabel("账号："), passWord = new JLabel("密码：");

JTextField inputAcc = new JTextField(20);

JPasswordField inputPas = new JPasswordField(20);

JButton button = new JButton("登录");

Box hBox1 = Box.*createHorizontalBox*();

Box hBox2 = Box.*createHorizontalBox*();

Box vBox1 = Box.*createVerticalBox*();

Box vBox2 = Box.*createVerticalBox*();

JPanel panel = new JPanel(new FlowLayout(FlowLayout.*CENTER*));

LoginFrame(){

String style = UIManager.*getSystemLookAndFeelClassName*();

try {

UIManager.*setLookAndFeel*(style);

} catch (ClassNotFoundException | UnsupportedLookAndFeelException | IllegalAccessException |

InstantiationException e) {

throw new RuntimeException(e);

}

}

public void init() {

account.setFont(new Font("微软雅黑",Font.*PLAIN*,15));

passWord.setFont(new Font("微软雅黑",Font.*PLAIN*,15));

hBox1.add(account);

hBox1.add(inputAcc);

hBox2.add(passWord);

hBox2.add(inputPas);

vBox1.add(hBox1);

vBox1.add(hBox2);

panel.add(button);

vBox2.add(vBox1);

vBox2.add(panel);

button.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String code = new String(inputPas.getPassword());

String sql = "select password from user where username = ?";

String password = "";

try {

PreparedStatement ps = *getConn*().prepareStatement(sql);

ps.setString(1, inputAcc.getText());

ResultSet rs = ps.executeQuery();

while(rs.next()) {

password = rs.getString(1);

}

} catch (Exception ex){

ex.printStackTrace();

}

if (password.equals(code)) {

isLive = false;

JOptionPane.*showMessageDialog*(null, "登陆成功！");

createMainFrame();

} else {

JOptionPane.*showMessageDialog*(null, "账号或密码错误，请重新登录。");

}

}

});

this.add(vBox2);

this.pack();

this.setLocationRelativeTo(null);

this.setDefaultCloseOperation(JFrame.*DISPOSE\_ON\_CLOSE*);

this.setVisible(true);

}

public void createMainFrame(){

if (!isLive){

new MainFrame().init();

this.dispose();

}

}

public static Connection getConn(){

String driver = "com.mysql.jdbc.Driver";

String url = "jdbc:mysql://localhost:3306/restaurant?useUnicode=true&characterEncoding=utf8";

String user = "root";

String password = "123456";

try{

Class.*forName*(driver);

Connection conn;

conn = DriverManager.*getConnection*(url,user,password);

/\* System.out.println("数据库链接成功");\*/

return conn;

} catch (Exception e) {

e.printStackTrace();

return null;

}

}

}

**主界面**

package Restaurant\_system.Frame;

import javax.swing.\*;

public class MainFrame extends JFrame{

public JTabbedPane tabbedPanel = new JTabbedPane();

public employeePanel employeePanel = new employeePanel();

public CustomerPanel customerPanel = new CustomerPanel();

public TablePanel tablePanel = new TablePanel();

public DishPanel dishPanel = new DishPanel();

public OrderPanel orderPanel = new OrderPanel();

public void init(){

setTitle("餐饮管理系统");

setBounds(500,200,600,480);

setDefaultCloseOperation(*EXIT\_ON\_CLOSE*);

tabbedPanel.addTab("员工管理",employeePanel);

tabbedPanel.addTab("客户管理",customerPanel);

tabbedPanel.addTab("餐台管理",tablePanel);

tabbedPanel.addTab("菜品管理",dishPanel);

tabbedPanel.addTab("订单管理",orderPanel);

add(tabbedPanel);

setVisible(true);

}

}

**员工管理**（其余仅涉及增删改查与员工管理无异，可参考）

**实体类**

package Restaurant\_system.Entity;

public class employee {

// "工号","姓名","性别","年龄","电话","职位"

private int id;

private String name;

private String sex;

private int age;

private String tel;

private String position;

public void setid(int id){

this.id = id;

}

public void setname(String name){

this.name = name;

}

public void setsex(String sex){

this.sex = sex;

}

public void setage(int age){

this.age = age;

}

public void settel(String tel){

this.tel = tel;

}

public void setposition(String position){

this.position = position;

}

public int getid(){

return id;

}

public String getname(){

return name;

}

public String getsex(){

return sex;

}

public int getage(){

return age;

}

public String gettel(){

return tel;

}

public String getposition(){

return position;

}

}

**界面**

package Restaurant\_system.Frame;

import Restaurant\_system.Dialog.employee\_iou\_Dialog;

import Restaurant\_system.SqlControls.employee\_SqlControls;

import java.sql.\*;

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.PreparedStatement;

import Restaurant\_system.Dialog.\*;

public class employeePanel extends JPanel {

private JPanel btpl = new JPanel();

private String[] str = new String[]{"工号","姓名","性别","年龄","电话","职位"};

private DefaultTableModel model =new DefaultTableModel(new Object[][]{},str);

private JTable table = new JTable(model){

@Override

public boolean isCellEditable(int row,int column){

return false;

}

};

private JButton bt1 = new JButton("新增");

private JButton bt2 = new JButton("查询");

private JButton bt3 = new JButton("修改");

private JButton bt4 = new JButton("删除");

private JTextField checkText = new JTextField(10);

private int checkno;

employeePanel(){

setLayout(new BorderLayout());

//model.addRow(new Object[]{});

btpl.add(bt1);

btpl.add(bt2);

btpl.add(checkText);

btpl.add(bt3);

btpl.add(bt4);

add(btpl,BorderLayout.*NORTH*);

add(new JScrollPane(table),BorderLayout.*CENTER*);

bt1.addActionListener(actionListener1);

bt2.addActionListener(actionListener2);

bt3.addActionListener(actionListener3);

bt4.addActionListener(actionListener4);

*Brush\_table*(model);

}

ActionListener actionListener1 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

new employee\_iou\_Dialog((MainFrame)getRootPane().getParent(),table,true);

}

};

ActionListener actionListener2 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

try{

checkno = Integer.*valueOf*(checkText.getText());

if(!(checkText.getText()).equals("")) {

new employee\_check\_Dialog((MainFrame) getRootPane().getParent(), checkno);

}else{

JOptionPane.*showMessageDialog*(null, "请输入想要查询的工号。");

}

}catch (Exception ex){

JOptionPane.*showMessageDialog*(null, "请输入正确的工号。");

}

}

};

ActionListener actionListener3 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if (table.getSelectedRow() == -1) {JOptionPane.*showMessageDialog*(null, "请选择需要修改的行。");}

else {new employee\_iou\_Dialog((MainFrame)getRootPane().getParent(),table,false);}

}

};

ActionListener actionListener4 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if (table.getSelectedRow() == -1) {JOptionPane.*showMessageDialog*(null, "请选择需要删除的行。");}

else{

int r = table.getSelectedRow();

employee\_SqlControls.*employee\_delete*((int)table.getValueAt(r, 0));

*Brush\_table*(model);

}

}

};

public static void Brush\_table(DefaultTableModel model){

model.getDataVector().clear();

String sql = "select \* from employee";

try {

PreparedStatement ps = LoginFrame.*getConn*().prepareStatement(sql);

ResultSet rs = ps.executeQuery();

while(rs.next()) {

model.addRow(new Object[]{

rs.getInt(1),

rs.getString(2),

rs.getString(3),

rs.getInt(4),

rs.getString(5),

rs.getString(6)

});

}

} catch(Exception e){}

}

}

**新增与修改的Dialog界面**

package Restaurant\_system.Dialog;

import Restaurant\_system.Entity.employee;

import Restaurant\_system.Frame.employeePanel;

import Restaurant\_system.SqlControls.employee\_SqlControls;

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class employee\_iou\_Dialog extends JDialog {

private JLabel idLabel = new JLabel("工号");

private JLabel nameLabel = new JLabel("姓名");

private JLabel sexLabel = new JLabel("性别");

private JLabel ageLabel = new JLabel("年龄");

private JLabel telLabel = new JLabel("电话");

private JLabel positionLabel = new JLabel("职位");

private JTextField idText = new JTextField();

private JTextField nameText = new JTextField();

private JRadioButton maleRadio = new JRadioButton("男");

private JRadioButton femaleRadio = new JRadioButton("女");

private ButtonGroup sexGroup = new ButtonGroup();

private JPanel sexPanel = new JPanel(new GridLayout(1, 2));

private JTextField ageText = new JTextField();

private JTextField telText = new JTextField();

private JTextField positionText = new JTextField();

private JButton okButton = new JButton("完成");

private JButton cancelButton = new JButton("取消");

private employee employee= new employee();

private JTable table;

public employee\_iou\_Dialog(Frame owner,JTable table,boolean isiou) {

super(owner, "员工", true);

if(isiou){

this.table = table;

setTitle("员工");

setLocation(600,300);

setLayout(new GridLayout(14, 1));

sexPanel.add(maleRadio);

sexPanel.add(femaleRadio);

sexGroup.add(maleRadio);

sexGroup.add(femaleRadio);

add(idLabel);

add(idText);

add(nameLabel);

add(nameText);

add(sexLabel);

add(sexPanel);

add(ageLabel);

add(ageText);

add(telLabel);

add(telText);

add(positionLabel);

add(positionText);

add(okButton);

add(cancelButton);

okButton.addActionListener(Listener1);

cancelButton.addActionListener(Listener3);

pack();

setVisible(true);

}else{

int r = table.getSelectedRow();

idText.setText(table.getValueAt(r, 0).toString());

nameText.setText(table.getValueAt(r, 1).toString());

if (table.getValueAt(r,2).toString().equals("男")) maleRadio.setSelected(true);

else if (table.getValueAt(r,2).toString().equals("女")) femaleRadio.setSelected(true);

ageText.setText(table.getValueAt(r, 3).toString());

telText.setText(table.getValueAt(r, 4).toString());

positionText.setText(table.getValueAt(r, 5).toString());

this.table = table;

setTitle("员工");

setLocation(600,300);

setLayout(new GridLayout(14, 1));

sexPanel.add(maleRadio);

sexPanel.add(femaleRadio);

sexGroup.add(maleRadio);

sexGroup.add(femaleRadio);

idText.setEditable(false);

add(idLabel);

add(idText);

add(nameLabel);

add(nameText);

add(sexLabel);

add(sexPanel);

add(ageLabel);

add(ageText);

add(telLabel);

add(telText);

add(positionLabel);

add(positionText);

add(okButton);

add(cancelButton);

okButton.addActionListener(Listener2);

cancelButton.addActionListener(Listener3);

pack();

setVisible(true);

}

}

ActionListener Listener1 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

employee.setid(Integer.*valueOf*(idText.getText()));

employee.setname(nameText.getText());

if (maleRadio.isSelected()) employee.setsex("男");

else if (femaleRadio.isSelected()) employee.setsex("女");

employee.setage(Integer.*valueOf*(ageText.getText()));

employee.settel(telText.getText());

employee.setposition(positionText.getText());

employee\_SqlControls.*employee\_insert*(employee);

dispose();

employeePanel.*Brush\_table*((DefaultTableModel)table.getModel());

};

};

ActionListener Listener2 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

employee.setid(Integer.*valueOf*(idText.getText()));

employee.setname(nameText.getText());

if (maleRadio.isSelected()) employee.setsex("男");

else if (femaleRadio.isSelected()) employee.setsex("女");

employee.setage(Integer.*valueOf*(ageText.getText()));

employee.settel(telText.getText());

employee.setposition(positionText.getText());

employee\_SqlControls.*employee\_update*(employee);

dispose();

employeePanel.*Brush\_table*((DefaultTableModel)table.getModel());

};

};

ActionListener Listener3 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

dispose();

}

};

}

**查询Dialog界面**

package Restaurant\_system.Dialog;

import Restaurant\_system.Entity.employee;

import Restaurant\_system.SqlControls.employee\_SqlControls;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class employee\_check\_Dialog extends JDialog{

private JLabel idLabel= new JLabel("工号");

private JLabel nameLabel = new JLabel("姓名");

private JLabel sexLabel = new JLabel("性别");

private JLabel ageLabel = new JLabel("年龄");

private JLabel telLabel = new JLabel("电话");

private JLabel positionLabel = new JLabel("职位");

private JButton bt2 = new JButton("完成");

private JTextField idText = new JTextField(10);

private JTextField nameText = new JTextField(10);

private JTextField ageText = new JTextField(10);

private JTextField sexText = new JTextField(10);

private JTextField telText = new JTextField(10);

private JTextField positionText = new JTextField(10);

private int checkno;

private employee emloyee;

public employee\_check\_Dialog(Frame owner,int checkno){

super(owner,"员工查询",true);

this.checkno = checkno;

setTitle("员工查询");

setSize(300,300);

setLocation(600,300);

setLayout(new GridLayout(13,1));

idText.setEditable(false);

idText.setText(((Integer)checkno).toString());

nameText.setEditable(false);

sexText.setEditable(false);

ageText.setEditable(false);

telText.setEditable(false);

positionText.setEditable(false);

add(idLabel);

add(idText);

add(nameLabel);

add(nameText);

add(sexLabel);

add(sexText);

add(ageLabel);

add(ageText);

add(telLabel);

add(telText);

add(positionLabel);

add(positionText);

add(bt2);

bt2.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

dispose();

}

});

emloyee = employee\_SqlControls.*employee\_check*(checkno);

nameText.setText(emloyee.getname());

sexText.setText(emloyee.getsex());

ageText.setText(String.*valueOf*(emloyee.getage()));

telText.setText(emloyee.gettel());

positionText.setText(emloyee.getposition());

setVisible(true);

}

}

**Sql\_controls方法**

package Restaurant\_system.SqlControls;

import Restaurant\_system.Entity.\*;

import Restaurant\_system.Frame.LoginFrame;

import javax.swing.\*;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

public class employee\_SqlControls {

public static void employee\_insert(employee employee){

String sql = "insert into employee values (?,?,?,?,?,?)";

try {

PreparedStatement ps = LoginFrame.*getConn*().prepareStatement(sql);

ps.setInt(1,Integer.*valueOf*(employee.getid()));

ps.setString(2,employee.getname());

ps.setString(3,employee.getsex());

ps.setInt(4,Integer.*valueOf*(employee.getage()));

ps.setString(5,employee.gettel());

ps.setString(6,employee.getposition());

ps.executeUpdate();

} catch(Exception ex){

JOptionPane.*showMessageDialog*(null, "输入的信息有误，请遵循正确的新增格式（工号不重复且为数字，年龄为数字）。");

}

}

public static employee employee\_check(int checkno){

employee employee = new employee();

employee.setid(checkno);

String sql = "select \* from employee where id = ?";

try {

PreparedStatement ps = LoginFrame.*getConn*().prepareStatement(sql);

ps.setInt(1,checkno);

ResultSet rs = ps.executeQuery();

//ps.executeUpdate();

while (rs.next()){

employee.setname(rs.getString(2));

employee.setsex(rs.getString(3));

employee.setage(rs.getInt(4));

employee.settel(rs.getString(5));

employee.setposition(rs.getString(6));

}

}catch(Exception ex){

ex.printStackTrace();

}

return employee;

}

public static void employee\_update(employee employee){

String sql = "update employee set name = ?,sex = ?,age = ?,tel= ?,position = ? where id = ?";

try {

PreparedStatement ps = LoginFrame.*getConn*().prepareStatement(sql);

ps.setInt(6,Integer.*valueOf*(employee.getid()));

ps.setString(1,employee.getname());

ps.setString(2,employee.getsex());

ps.setInt(3,Integer.*valueOf*(employee.getage()));

ps.setString(4,employee.gettel());

ps.setString(5,employee.getposition());

ps.executeUpdate();

} catch(Exception ex){

ex.printStackTrace();

}

}

public static void employee\_delete(int r){

String sql = "delete from employee where id = ?";

try {

PreparedStatement ps = LoginFrame.*getConn*().prepareStatement(sql);

ps.setInt(1,r);

ps.executeUpdate();

}catch(Exception ex){

ex.printStackTrace();

}

}

}

**订单管理**

**实体类**

package Restaurant\_system.Entity;

public class Order {

// ""订单号","餐台号","客户号","菜类","菜品","价格","状态"

private String id;

private int tableid;

private String category;

private String cuisine;

private Double price;

private String status;

public void setId(String id){

this.id = id;

}

public void setTableid(int id){

this.tableid = id;

}

public void setCategory(String category){

this.category = category;

}

public void setCuisine(String cuisine){

this.cuisine = cuisine;

}

public void setPrice(Double price){

this.price = price;

}

public void setStatus(String status){

this.status = status;

}

public String getId(){

return id;

}

public int getTableid(){

return tableid;

}

public String getCategory(){

return category;

}

public String getCuisine(){

return cuisine;

}

public Double getPrice(){

return price;

}

public String getStatus(){

return status;

}

}

**界面**

package Restaurant\_system.Frame;

import Restaurant\_system.Dialog.Order\_check\_Dialog;

import Restaurant\_system.Dialog.Order\_Dialog;

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import static Restaurant\_system.SqlControls.Order\_SqlControls.*waitpay*;

public class OrderPanel extends JPanel {

private JPanel btpl = new JPanel();

private String[] str= new String[]{"订单号","餐台号","菜类","菜品","价格","状态"};

private DefaultTableModel model = new DefaultTableModel(new Object[][]{},str);

public JTable table = new JTable(model){

@Override

public boolean isCellEditable(int row,int column){

return false;

}

};

private JButton bt1 = new JButton("点菜管理"),

bt2 = new JButton("查询订单"),

bt3 = new JButton("结账管理"),

bt4 = new JButton("显示未支付"),

bt5 = new JButton("刷新");

private JTextField check= new JTextField(10);

public OrderPanel(){

setLayout(new BorderLayout());

model.addRow(new Object[]{});

btpl.add(bt1);

btpl.add(bt2);

btpl.add(check);

btpl.add(bt3);

btpl.add(bt4);

btpl.add(bt5);

add(btpl,BorderLayout.*NORTH*);

add(new JScrollPane(table),BorderLayout.*CENTER*);

bt1.addActionListener(Listner1);

bt2.addActionListener(Listner2);

bt3.addActionListener(Listner3);

bt4.addActionListener(Listner4);

bt5.addActionListener(Listner5);

*Brush\_table*(model);

}

ActionListener Listner1 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

try {

if(check.getText().equals("")){

JOptionPane.*showMessageDialog*(null, "请输入餐台号。");

}else {

new Order\_Dialog((MainFrame)getRootPane().getParent(),table,Integer.*valueOf*(check.getText()));

}

}catch (Exception ex){

JOptionPane.*showMessageDialog*(null, "请输入正确的餐台号。");

}

}

};

ActionListener Listner2 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

try {

if(check.getText().equals("")){

JOptionPane.*showMessageDialog*(null, "请输入餐台号。");

}else {

new Order\_check\_Dialog((MainFrame)getRootPane().getParent(),Integer.*valueOf*(check.getText()),false);

}

}catch (Exception ex){

JOptionPane.*showMessageDialog*(null, "请输入正确的餐台号。");

}

}

};

ActionListener Listner3 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

try {

if(check.getText().equals("")){

JOptionPane.*showMessageDialog*(null, "请输入餐台号。");

}else {

new Order\_check\_Dialog((MainFrame)getRootPane().getParent(),Integer.*valueOf*(check.getText()),true);

}

}catch (Exception ex){

JOptionPane.*showMessageDialog*(null, "请输入正确的餐台号。");

}

}

};

ActionListener Listner4 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

*waitpay*(model);

table.repaint();

}

};

ActionListener Listner5 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

*Brush\_table*((DefaultTableModel)table.getModel());

}

};

public static void Brush\_table(DefaultTableModel model){

model.getDataVector().clear();

String sql = "select \* from `order`";

try {

PreparedStatement ps = LoginFrame.*getConn*().prepareStatement(sql);

ResultSet rs = ps.executeQuery();

while(rs.next()) {

model.addRow(new Object[]{

rs.getString(1),

rs.getInt(2),

rs.getString(3),

rs.getString(4),

rs.getDouble(5),

rs.getString(6)

});

}

} catch(Exception e){}

}

}

**点菜界面**

package Restaurant\_system.Dialog;

import Restaurant\_system.Entity.Order;

import Restaurant\_system.Frame.MainFrame;

import Restaurant\_system.Frame.OrderPanel;

import Restaurant\_system.Frame.TablePanel;

import Restaurant\_system.SqlControls.Order\_SqlControls;

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import static Restaurant\_system.SqlControls.Order\_SqlControls.\*;

public class Order\_Dialog extends JDialog {

private Order order = new Order();

private String[] str1= new String[]{"菜类","菜品","价格","状态"};

private String[] str2= new String[]{"菜类","菜品","价格"};

private DefaultTableModel model1 = new DefaultTableModel(new Object[][]{},str1);

private DefaultTableModel model2 = new DefaultTableModel(new Object[][]{},str2);

private JTable table;

private JTable table1 ;

private JTable table2 = new JTable(model2){

@Override

public boolean isCellEditable(int row,int column){

return false;

}

};

private JPanel checkpl = new JPanel();

private JPanel btpl = new JPanel(new GridLayout(4,1));

private JLabel check\_orderidLabel = new JLabel("餐台号");

private JTextField check\_orderidText = new JTextField(10);

private JButton bt1 = new JButton("下单");

private JButton bt2 = new JButton("退订");

private JButton bt3 = new JButton("完成");

private JButton bt4 = new JButton("取消");

private int tableid;

public Order\_Dialog(Frame owner,JTable table,int tableid){

super(owner,"点菜",true);

this.table = table;

this.tableid = tableid;

setTitle("点菜");

setLocation(400,250);

setLayout(new GridLayout(4,1));

setLayout(new BorderLayout());

table1 = new JTable(Order\_SqlControls.*get\_cuisine*(model1)){

@Override

public boolean isCellEditable(int row,int column){

return false;

}

};

check\_orderidText.setText(tableid+"");

check\_orderidText.setEditable(false);

checkpl.add(check\_orderidLabel);

checkpl.add(check\_orderidText);

btpl.add(bt1);

btpl.add(bt2);

btpl.add(bt3);

btpl.add(bt4);

bt1.addActionListener(Listener1);

bt2.addActionListener(Listener2);

bt3.addActionListener(Listener3);

bt4.addActionListener(Listener4);

add(checkpl,BorderLayout.*NORTH*);

add(new JScrollPane(table1),BorderLayout.*WEST*);

add(new JScrollPane(table2),BorderLayout.*EAST*);

add(btpl,BorderLayout.*CENTER*);

pack();

setVisible(true);

}

ActionListener Listener1 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

int r = table1.getSelectedRow();

*add\_order*(model1,model2,r);

}

};

ActionListener Listener2 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

int r = table2.getSelectedRow();

*sub\_order*(model2,r);

}

};

ActionListener Listener3 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

*order\_table\_update*(tableid);

for(int r = 0;r < model2.getRowCount();r++){

*Order\_insert*(*Order\_getdata*(model2,r,tableid));

dispose();

}

OrderPanel.*Brush\_table*((DefaultTableModel)table.getModel());

TablePanel.*Brush\_table*((DefaultTableModel) ((MainFrame)table.getRootPane().getParent()).tablePanel.table.getModel());

}

};

ActionListener Listener4 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

dispose();

}

};

}

**查询与结账界面**

package Restaurant\_system.Dialog;

import Restaurant\_system.Frame.MainFrame;

import Restaurant\_system.Frame.OrderPanel;

import Restaurant\_system.Frame.TablePanel;

import Restaurant\_system.SqlControls.Order\_SqlControls;

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import static Restaurant\_system.SqlControls.Order\_SqlControls.\*;

public class Order\_check\_Dialog extends JDialog{

private String[] str= new String[]{"订单号","餐台号","客户号","菜品","价格","状态"};

private DefaultTableModel model = new DefaultTableModel(new Object[][]{},str);

private MainFrame owner;

private JTable table1;

private JPanel checkpl = new JPanel();

private JPanel btpl = new JPanel();

private JLabel check\_orderidLabel = new JLabel("订单号");

private JButton bt1 = new JButton("完成");

private JButton bt2 = new JButton("结账");

private JTextField payText = new JTextField(10);

private int tableid;

private int selected;

private Double price\_sum;

public Order\_check\_Dialog(Frame owner,int tableid,boolean pay){

super(owner,"订单查询",true);

this.owner = (MainFrame)owner;

this.tableid = tableid;

setTitle("订单查询");

setLocation(600,250);

setLayout(new GridLayout(4,1));

setLayout(new BorderLayout());

table1 = new JTable(Order\_SqlControls.*order\_check*(tableid,str)){

@Override

public boolean isCellEditable(int row,int column){

return false;

}

};

checkpl.add(check\_orderidLabel);

if(pay){

price\_sum = *get\_sum\_price*(tableid);

btpl.add(bt1);

btpl.add(bt2);

btpl.add(payText);

add(checkpl,BorderLayout.*NORTH*);

add(new JScrollPane(table1),BorderLayout.*CENTER*);

add(btpl,BorderLayout.*SOUTH*);

bt1.addActionListener(Listener1);

bt2.addActionListener(Listener2);

}else{

btpl.add(bt1);

add(checkpl,BorderLayout.*NORTH*);

add(new JScrollPane(table1),BorderLayout.*CENTER*);

add(btpl,BorderLayout.*SOUTH*);

bt1.addActionListener(Listener1);

}

pack();

setVisible(true);

}

ActionListener Listener1 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

dispose();

}

};

ActionListener Listener2 = new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if(Double.*valueOf*(payText.getText()) >= price\_sum){

Double change = Double.*valueOf*(payText.getText())-price\_sum;

*had\_paid*(tableid);

dispose();

OrderPanel.*Brush\_table*((DefaultTableModel)owner.orderPanel.table.getModel());

TablePanel.*Brush\_table*((DefaultTableModel) owner.tablePanel.table.getModel());

JOptionPane.*showMessageDialog*(null, "谢谢惠顾，找零："+change);

}else {

JOptionPane.*showMessageDialog*(null, "金额不足");

}

}

};

}

**Sql\_control方法**

package Restaurant\_system.SqlControls;

import Restaurant\_system.Entity.Cuisine;

import Restaurant\_system.Entity.Order;

import Restaurant\_system.Entity.Table;

import Restaurant\_system.Frame.LoginFrame;

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Random;

public class Order\_SqlControls {

public static boolean isbooked(int tableid){

String sql = "select status from desk where id = ?";

try {

PreparedStatement ps = LoginFrame.*getConn*().prepareStatement(sql);

ps.setInt(1,tableid);

ResultSet rs = ps.executeQuery();

//ps.executeUpdate();

while (rs.next()){

if(rs.getString(1).equals("已开台")){

return true;

}

}

}catch(Exception ex){

ex.printStackTrace();

}

return false;

}

public static DefaultTableModel get\_cuisine(DefaultTableModel model){

String sql = "select \* from cuisine";

try {

PreparedStatement ps = LoginFrame.*getConn*().prepareStatement(sql);

//ps.setString(1,"在售");

ResultSet rs = ps.executeQuery();

//ps.executeUpdate();

while(rs.next()) {

if(rs.getString(6).equals("在售")){

model.addRow(new Object[]{

rs.getString(3),

rs.getString(1),

rs.getDouble(5),

rs.getString(6),

});}

}

}catch(Exception ex){

ex.printStackTrace();

}

return model;

}

public static void orderedcuisine\_delete(String orderid){

String sql = "delete from `order` where id = ?";

try {

PreparedStatement ps = LoginFrame.*getConn*().prepareStatement(sql);

ps.setString(1,orderid);

ps.executeUpdate();

}catch(Exception ex){

ex.printStackTrace();

}

}

public static void order\_brush(){

}

public static void order\_table\_update(int tableid){

String sql = "update desk set status = ? where id = ?";

try {

PreparedStatement ps = LoginFrame.*getConn*().prepareStatement(sql);

ps.setInt(2,tableid);

ps.setString(1,"已开台");

ps.executeUpdate();

} catch(Exception ex){

ex.printStackTrace();

}

}

public static void add\_order(DefaultTableModel model1, DefaultTableModel model2,int r){

String category = model1.getValueAt(r,0).toString();

String cuisine = model1.getValueAt(r,1).toString();

Double price = Double.*valueOf*(model1.getValueAt(r,2).toString());

model2.addRow(new Object[]{

category,cuisine,price

});

}

public static void sub\_order(DefaultTableModel model, int r){

model.removeRow(r);

}

public static Order Order\_getdata(DefaultTableModel model,int r,int t){

Order order = new Order();

String category;

String cuisine;

Double price;

category = model.getValueAt(r,0).toString();

cuisine = model.getValueAt(r,1).toString();

price = (double)model.getValueAt(r,2);

order.setId(*orderid*());

order.setTableid(t);

order.setCategory(category);

order.setCuisine(cuisine);

order.setPrice(price);

order.setStatus("未支付");

return order;

}

public static void Order\_insert(Order order){

String sql = "insert into `order` values (?,?,?,?,?,?)";

try {

PreparedStatement ps = LoginFrame.*getConn*().prepareStatement(sql);

ps.setString(1,order.getId());

ps.setInt(2,order.getTableid());

ps.setString(3,order.getCategory());

ps.setString(4,order.getCuisine());

ps.setDouble(5,order.getPrice());

ps.setString(6,order.getStatus());

ps.executeUpdate();

} catch(Exception ex){

ex.printStackTrace();

}

}

public static String orderid(){

SimpleDateFormat df = new SimpleDateFormat("yyyyMMddHHmmss");

String timeStr = df.format(new java.util.Date());

Random random = new Random();

return timeStr + random.nextInt(10)+random.nextInt(10)+random.nextInt(10)+random.nextInt(10);

}

public static DefaultTableModel order\_check(int tableid,String[] str){

DefaultTableModel model = new DefaultTableModel(new Object[][]{},str);

String sql1 = "select \* from `order` where deskId = ? and status = '未支付'";

String sql2 = "select sum(price) from `order` where deskId = ? and status = '未支付'";

try {

PreparedStatement ps1 = LoginFrame.*getConn*().prepareStatement(sql1);

PreparedStatement ps2 = LoginFrame.*getConn*().prepareStatement(sql2);

ps1.setInt(1,tableid);

ps2.setInt(1,tableid);

ResultSet rs1 = ps1.executeQuery();

ResultSet rs2 = ps2.executeQuery();

//ps.executeUpdate();

while(rs1.next()) {

model.addRow(new Object[]{

rs1.getString(1),

rs1.getInt(2),

rs1.getString(3),

rs1.getString(4),

rs1.getDouble(5),

rs1.getString(6)

});

}

while (rs2.next()){

model.addRow(new Object[]{"","","","","总价",rs2.getDouble(1)});

}

}catch(Exception ex){

ex.printStackTrace();

}

return model;

}

public static Double get\_sum\_price(int tableid){

String sql = "select sum(price) from `order` where deskId = ? and status = '未支付'";

Double sum\_price = 0.0;

try {

PreparedStatement ps = LoginFrame.*getConn*().prepareStatement(sql);

ps.setInt(1,tableid);

ResultSet rs = ps.executeQuery();

while (rs.next()){

sum\_price = rs.getDouble(1);

}

//ps.executeUpdate();

}catch (Exception e){

e.printStackTrace();

}

return sum\_price;

}

public static void had\_paid(int tableid){

String sql1 = "update desk set status = ? where id = ?";

String sql2 = "update `order` set status = ? where deskId = ?";

try {

PreparedStatement ps1 = LoginFrame.*getConn*().prepareStatement(sql1);

PreparedStatement ps2 = LoginFrame.*getConn*().prepareStatement(sql2);

ps1.setInt(2,tableid);

ps1.setString(1,"未开台");

ps2.setInt(2,tableid);

ps2.setString(1,"已支付");

ps1.executeUpdate();

ps2.executeUpdate();

} catch(Exception ex){

ex.printStackTrace();

}

}

public static void waitpay(DefaultTableModel model){

model.getDataVector().clear();

String sql = "select \* from `order` where status = '未支付'";

try {

PreparedStatement ps = LoginFrame.*getConn*().prepareStatement(sql);

ResultSet rs = ps.executeQuery();

while(rs.next()) {

model.addRow(new Object[]{

rs.getString(1),

rs.getInt(2),

rs.getString(3),

rs.getString(4),

rs.getDouble(5),

rs.getString(6)

});

}

} catch(Exception e){

e.printStackTrace();

}

}

}