**哦**

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** | **李佳波** |
| **学号** | **B20210304127** |
| **指导教师** | **潘怡，杨刚** |
| **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 SQL Server 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、菜品管理界面等人机接口。

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 | 卢卓云 | 学号 | B20210304128 | 班级 | 21软件01 |
| 专业 | 软件工程 | | 指导教师姓名 | 潘怡，杨刚 | |
| 项目名称 | 基于C/S架构的餐饮管理系统 | | | | |
| **实训过程**：（说明实训各阶段的工作过程，包括需求分析、系统设计、编码实现、系统测试、交付实施等阶段的主要工作。）  本次实训是一个C/S架构的Java餐饮管理系统的设计与开发项目。在这个项目中，我们按照以下步骤进行了实际操作：  1. 需求分析阶段  在需求分析阶段，我们首先收集了项目所需的各种需求，包括系统用例图、功能需求和非功能需求等。通过对这些需求的分析，我们获得了开发系统的理解，确定了系统的边界和用例，同时也做出了一些前期设计的可行性分析。  2. 系统设计阶段  在系统设计阶段，我们对软件开发进行了详细的规划和设计。首先，我们设计了整个系统的框架和基本模块，包括用户登录、员工管理、菜品管理、订单管理、库存管理、报表统计和系统设置等。然后，我们对需要实现的功能进行了更加详细和深入的设计，包括数据库设计、类图设计、状态图设计等。  3. 编码实现阶段  在编码实现阶段，我们根据设计方案进行了代码编写。我们使用Java语言对餐饮管理系统的各个模块进行了编程和开发，同时充分运用了MySQL等数据库，以及JDBC、Swing和JavaFX等技术，实现了系统的各种功能。在编码实现的过程中，我们也注重组员之间的交流和协作，通过即时沟通产生协同化效果，确保代码的质量和可靠性。  4. 系统测试阶段  在系统测试阶段，我们将系统进行了全面的测试和验收，包括单元测试、集成测试、系统测试和验收测试等。通过测试，我们发现了一些问题和缺陷，并及时进行了修复和改进。我们也着重记录了测试的过程和结果，以便更好地与顾客沟通和交流。  5. 交付实施阶段  在交付实施阶段，我们交付了系统已开发与测试完成的软件代码和用户指南给客户。客户会先进行用户验收测试，然后根据实际情况调整使用方式或提出其他的需求。我们根据问题的反馈进行更多的修改和改进，以提高系统的质量和性能。  在本次实训中，我们进行了全面的需求分析、系统设计、编码实现、系统测试和交付实施等多个阶段的工作。在每个阶段，我们都发现了不少问题，同时也积累了丰富的经验和技能，这将对我们今后的学习和工作都有着重要的意义。 | | | | | |
| **实训成果**：  参见基于C/S架构的餐饮管理系统设计说明书。 | | | | | |
| **实训总结**：（包括心得体会、存在的问题和改进方向。）  在这次实训中，我对Java编程、C/S系统设计与开发等方面有了更深入的了解，并在实践过程中学到了很多有用的知识和技能。下面是我在实训中的心得体会、问题和改进方向。  一、心得体会  1. 学习要注重思维方式和学习能力。在本次实训中，我深深体会到了这一点，通过实际操作和练习，我不仅掌握了相关技能，还开始逐渐形成了更为系统和全面的思维方式。  2. 解决问题的能力是很关键的。在实验过程中，我们遇到了很多问题，其中有一些涉及到编程技术、有些是关于餐饮行业管理的事情。解决问题的能力是非常重要的，这需要我们不仅具备高超的技术和行业知识，更需要我们拥有扎实的逻辑思维能力和实际处理问题的经验。  二、存在的问题  1.本次实训主要是针对我们对项目流程不熟悉和对整体项目的把握不清楚，学习数据库的设计和表的建设以及表与表之间的联系，还有一些代码的编写，这些都是我们所不熟悉的也是我们最薄弱的部分。  2. 更注重数据的处理和存储。在我们的实验中，我们通常需要对数据进行操作、存储和管理。这时，我们应该更加注重数据的质量和处理效率，以便满足项目对数据的高要求和水平。  三、改进方向  1. 学习更多的技术和行业知识。在实验中，我们感觉到自己还有很多不足之处，需要加强学习和提高技术和行业知识。我们将通过系统阅读相关文献资料来巩固并扩大我们的知识体系。  2. 不断提高团队协作能力。在实验中，我们感到有些困难，部分原因是缺乏良好的团队协作能力。我们将通过讨论交流、编写文档、分享经验等途径来提高团队协作能力。  我希望在以后的学习或工作中以务实的态度给提供更多的机会锻炼自己，为成才与就业奠定坚实的基础；并希望同学们要做好角色定位，将自己所学的知识，学熟、学精。 多学习和收集与项目有关的资料，来提高自己的业务水平，同时多请教经验丰富的老师，使他们好的经验能够对自己起到借鉴作用，在他们的帮助下，能够很快的培养自己的管理技能及节省时间，避免做无用功，也可以让自己了解很多项目管理上的捷径，从而减少自己会遇到的挫折和错误。 | | | | | |

基于C/S架构的餐饮管理系统

设计说明书

作者：B20210304127(学号)

计算机科学与工程学院

2023年 06 月 15日

**摘要**

这是一份关于基于C/S架构的餐饮管理系统的Java设计文档报告的摘要：

该系统通过C/S架构实现，客户端使用Java Swing作为界面库。系统主要功能包括餐厅菜单管理、订单管理、员工管理、客户管理、菜品信息管理，餐台管理等。

在菜单管理模块中，管理员可以通过系统添加、编辑和删除菜品，同时支持菜品分类和菜品图片上传。在订单管理模块中，顾客可以通过客户端下单，收银员可以进行订单查询、结账等操作。

在员工管理模块中，管理员可以添加、编辑和删除员工，同时设置员工权限和工资。在库存管理模块中，管理员可以根据实际情况进行库存盘点、进货和出货等操作。

在统计分析模块中，系统可以根据时间段进行销售额、订单量等数据的统计分析，输出报表可以帮助管理员进行决策。

本文档除了系统设计描述，还包含了系统使用方法、安装部署及维护等方面的详细说明。该系统可以有效地提高餐厅的管理效率，降低管理成本，使得餐厅管理更加科学化和智能化。

**关键词：**Java；C/S架构；菜单管理；订单管理；员工管理，菜品管理，餐台管理；。

目录

[已知技术参数和设计要求： 2](#_Toc2606)

[1. 问题描述（功能要求）： 2](#_Toc31437)

[2. 运行环境要求： 2](#_Toc21639)

[3. 技术要求： 2](#_Toc9993)

[实训工作量： 3](#_Toc10535)

[工作计划： 3](#_Toc11215)

[1 引言 1](#_Toc26369)

[1.1 编写目的 1](#_Toc4443)

[1.2 参考资料 1](#_Toc20012)

[2 需求规约 2](#_Toc13578)

[2.1 功能需求 2](#_Toc30124)

[2.2 界面需求 6](#_Toc11397)

[2.3 数据需求 6](#_Toc21614)

[3 系统设计 9](#_Toc4929)

[3.1 运行环境 9](#_Toc8016)

[3.2 系统静态结构设计 10](#_Toc7418)

[3.3 人机接口设计 11](#_Toc3110)

[4 数据库设计 20](#_Toc23831)

[4.1 数据库环境说明 20](#_Toc18966)

[4.2 数据库的命名规则 20](#_Toc29206)

[4.3 逻辑结构设计 20](#_Toc16844)

[4.4 物理结构设计 22](#_Toc32116)

[5 测试用例设计 25](#_Toc1106)

[附录 29](#_Toc21415)

[附录1 程序运行结果 29](#_Toc17802)

[附录2 程序源代码 35](#_Toc27946)

# 1 引言

## 1.1 编写目的

该系统旨在提供一个方便实用、高效稳定的餐饮管理平台，适用于各种不同规模和类型的餐饮企业。该系统实现了菜品管理、订单管理、员工管理、客户管理、餐桌管理，结账管理，收银管理等诸多功能。该系统还采用了流行的技术和工具进行实现，如 JavaFX、Eclipsse,MySQL 数据库、JDBC 等。

本文档从系统的设计和实现两个方面进行介绍。首先，对系统设计进行了详细的阐述，包括系统总体架构和各个模块的实现。进一步，对项目的具体实现进行了详细的分析和代码实现的介绍，使读者能够全面了解项目的开发过程和技术要点。

本文档适用于 Java 开发人员、餐饮企业管理人员或其他对餐饮管理系统感兴趣的读者。在阅读本文档之前，读者应该具备一定的 Java 编程经验和对餐饮管理领域的了解。

未来，该项目还将持续不断地更新和完善，以更好地满足餐饮企业的需求。

## 1.2 参考资料

[1] Java 高级程序设计实战教程 （第二版|微课版） 作者： 戴远泉 程宁等人

人民邮电出版社，中国工信出版社集团。

1. Java 面向对象程序设计案例教程，主编 谭义红 主审 刘华富

北京邮电大学出版社。 www.buptpress.com

# 2 需求规约

## 2.1 功能需求

餐饮管理系统可大致分为：员工管理，客户管理，餐台管理，点菜管理，菜品管理，结账管理六大模块。分别实现了员工、客户的增、删、改、查，餐台的增加删除，点菜的加菜减菜，菜品种类的增加减少，结账的找零功能

根据餐饮行业的特点和该企业的实际情况，本餐饮管理系统以餐饮业务为基础，突出管理，从专业角度出发，提供科学有效的管理模式。

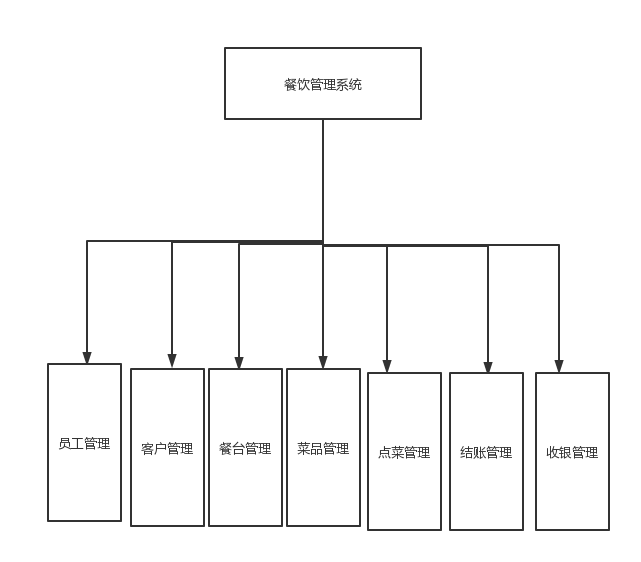
能够针对中餐为其菜品的多样化和特色化的服务提供标准化的管理。

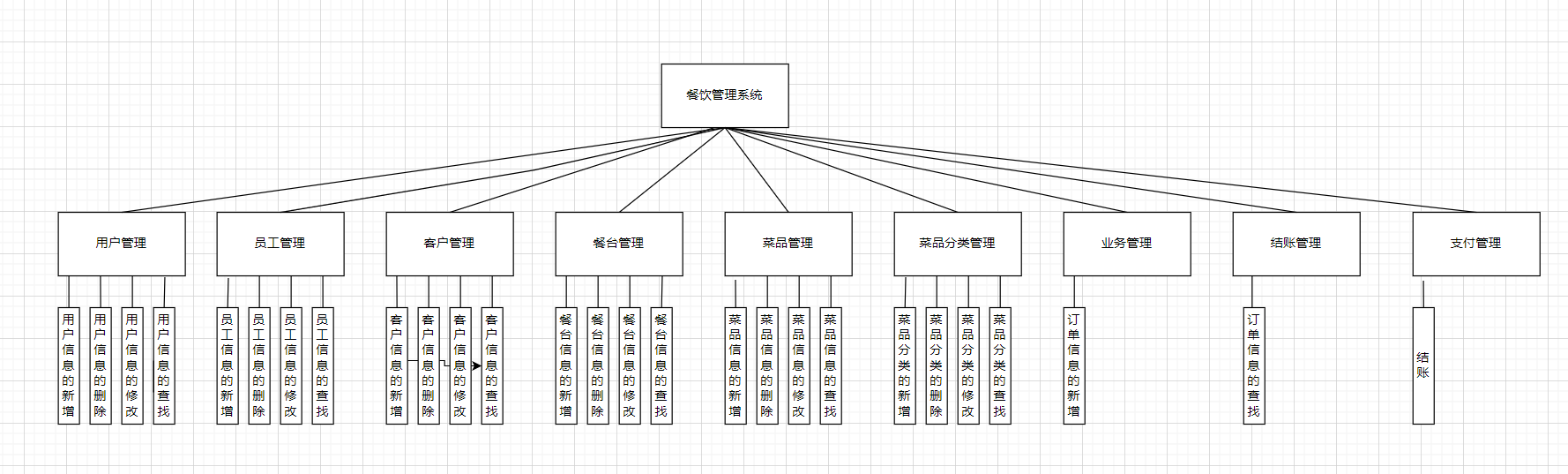
能够提供符合餐饮企业自身要求的较科学化的标准化、流程化管理，解决餐饮行业专业人才欠缺的问题。

能够依据订餐、点菜、结账等环节的繁重化及复杂化问题，实现强化管理、降低成本、堵漏节流等效益。

能够针对企业的经营现状做出科学的分析，使得企业对市场的应变能力得到提高。

图2.1 餐饮管理系统功能模块图





2.1.1 员工管理用例说明

用例名称：员工管理

用例描述：对员工信息实现增删改查

前置条件：删除或修改信息必须是已存在的员工

后置条件：无

活动步骤：

1. 选择要进行的增删改查操作。
2. 系统检查员工的状态。
3. 系统进行相应操作。

扩展点：

无

异常处理：

1. 如果删除或修改的员工信息不存在，则用例终止。

2.1.2 客户管理用例说明

用例名称：客户管理

用例描述：对客户实现增删改查

前置条件：删除或修改信息必须是已存在的客户

后置条件：无

活动步骤：

1. 选择要进行的增删改查操作。
2. 系统检查客户的状态。
3. 系统进行相关操作。

扩展：

无

异常处理：

1. 如果要删除或者修改的客户信息不存在，则用例终止。

2.1.3 餐台管理用例说明

用例名称：餐台管理

用例描述：对餐台实现增删改查

前置条件：删除或修改信息必须是已存在的餐台

后置条件：无

活动步骤：

1. 选择要进行的增删改查操作。
2. 系统检查餐台的状态。
3. 系统进行相关操作。

扩展：

无

异常处理：

1. 如果要删除或者修改的餐台信息不存在，则用例终止。

2.1.4 菜品管理用例说明

用例名称：菜品管理

用例描述：对菜品分类、实现增删改查

前置条件：删除或修改信息必须是已存在的菜品类别或者菜品信息。

后置条件：无

活动步骤：

1. 选择要进行的增删改查操作。
2. 系统检查菜品类别、菜品的状态。
3. 系统进行相关操作。

扩展：

无

异常处理：

1. 如果要删除或者修改的菜品类别不存在，则用例终止。
2. 如果要删除或者修改的菜品信息不存在，则用例终止。

2.1.5 点菜用例说明

用例名称：点菜用例。

用例描述：对点菜用例的操作说明

前置条件：该餐桌已经开台且该订单还未完成

后置条件：无

活动步骤：

1. 管理员选择一个就餐中的餐桌。
2. 进行添加菜品或者删除菜品操作。
3. 系统检查信息，并对该订单进行相关操作。

扩展：

菜品信息、已点菜品

异常处理：

1. 如果该餐桌未开台或订单已完成，则用例终止。

2.1.6 结账管理用例说明

用例名称：结账管理

用例描述：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、实收金额进行找零的计算，并显示，记录成统计未结账数据。

前置条件：

1. 该餐桌已经开台且该订单还未完成

后置条件：修改餐桌状态为空闲、订单状态为已完成。

2.1.7 收银管理用例说明

用例名称：收银管理

用例描述：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，通过手动输入优惠金额、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。

前置条件：

1. 该餐桌已经开台且该订单还未完成

后置条件：修改餐桌状态为空闲、订单状态为已完成。

活动步骤：

1. 管理员选择一个就餐中的餐桌。
2. 进行结账操作。
3. 找零，生成数据。

扩展：

无

异常处理：

1. 如果该餐桌未开台或订单已完成，则用例终止。

## 2.2 界面需求

1. 界面友好，操作简单。

2. 界面美观，用户体验感良好。

## 2.3 数据需求

表2.1 管理员信息表user

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 长度 | 是否允许为空 | 是否为主键 | 说明 |
| id | int | 10 | no | yes | 序号 |
| username | varchar | 20 | no | no | 用户名 |
| password | varchar | 20 | no | no | 密码 |

表2.2 员工信息表employee

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 长度 | 是否允许为空 | 是否为主键 | 说明 |
| id | int | 10 | no | yes | 序号 |
| name | varchar | 20 | no | no | 用户名 |
| sex | varchar | 2 | yes | no | 性别 |
| identityID | varchar | 18 | yes | no | 身份证号 |
| address | varchar | 40 | yes | no | 家庭住址 |
| tel | varchar | 11 | yes | no | 电话 |
| position | varchar | 4 | no | no | 职位 |
| freeze | varchar | 4 | no | no | 是否在职 |

表2.3 客户信息表customer

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 长度 | 是否允许为空 | 是否为主键 | 说明 |
| id | int | 10 | no | yes | 序号 |
| name | varchar | 20 | no | no | 用户名 |
| sex | varchar | 4 | yes | no | 性别 |
| company | varchar | 20 | yes | no | 单位 |
| tel | varchar | 11 | yes | no | 电话 |
| cardID | varchar | 10 | no | no | 贵宾卡号 |

表2.4 菜品分类表category

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 长度 | 是否允许为空 | 是否为主键 | 说明 |
| id | int | 10 | no | yes | 序号 |
| name | varchar | 20 | no | no | 名称 |
| describ | varchar | 20 | yes | no | 描述 |

表2.5 菜品信息表dish

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 长度 | 是否允许为空 | 是否为主键 | 说明 |
| id | int | 10 | no | yes | 序号 |
| name | varchar | 20 | no | no | 菜品名 |
| category | int | 10 | no | no | 类别编号 |
| pic | blob | 50 | no | no | 图片 |
| code | varchar | 8 | no | no | 菜品代码 |
| unit | varchar | 4 | yes | no | 单位 |
| price | double | 6 | yes | no | 价格 |
| status | varchar | 4 | yes | no | 状态 |

表2.6 餐台信息表desk

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 长度 | 是否允许为空 | 是否为主键 | 说明 |
| id | int | 10 | no | yes | 序号 |
| no | varchar | 8 | no | no | 餐台编号 |
| seating | int | 4 | no | no | 座位数 |
| status | varchar | 10 | no | no | 状态 |

表2.7 订单信息表order

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 长度 | 是否允许为空 | 是否为主键 | 说明 |
| id | int | 10 | no | yes | 序号 |
| orderNo | varchar | 20 | no | yes | 订单编号 |
| deskID | int | 10 | no | no | 餐台号 |
| createtime | date | 40 | no | no | 就餐时间 |
| money | double | 6 | no | no | 金额 |
| customerID | int | 10 | no | no | 客户编号 |
| status | varchar | 4 | no | no | 状态 |
| number | int | 4 | no | no | 就餐人数 |

表2.8 订单明细表orderitem

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 长度 | 是否允许为空 | 是否为主键 | 说明 |
| id | int | 10 | no | yes | 序号 |
| orderID | int | 10 | no | no | 订单编号 |
| dishID | int | 10 | no | no | 菜品编号 |
| amount | double | 4 | no | no | 菜品数量 |

# 3 系统设计

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：服务员对某一餐台点菜，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。
7. 收银管理：收银员结账系统自动统计数据，并显示找零完成结账的操作。

## 3.1 运行环境

编译器：Eclipse IDE Version: Photon Release (4.8.0)

Build id: 20180619-1200,

Mysql数据库：MySQLWorkbench以及Navicat Premium 15

Window11.系统

## 3.2 系统静态结构设计

图3.1 点餐系统总体类图

系统静态结构说明

先登录进入系统主界面，选择员工、客户、餐桌、菜品、开台和点菜管理；

进入菜品管理有两个选择：菜品分类管理和菜品信息，再进行相关操作；

进入业务管理有两个选择：点菜和开台管理，进行相关操作；

点菜管理中可以进入结账管理，进行结账操作。

## 3.3 人机接口设计

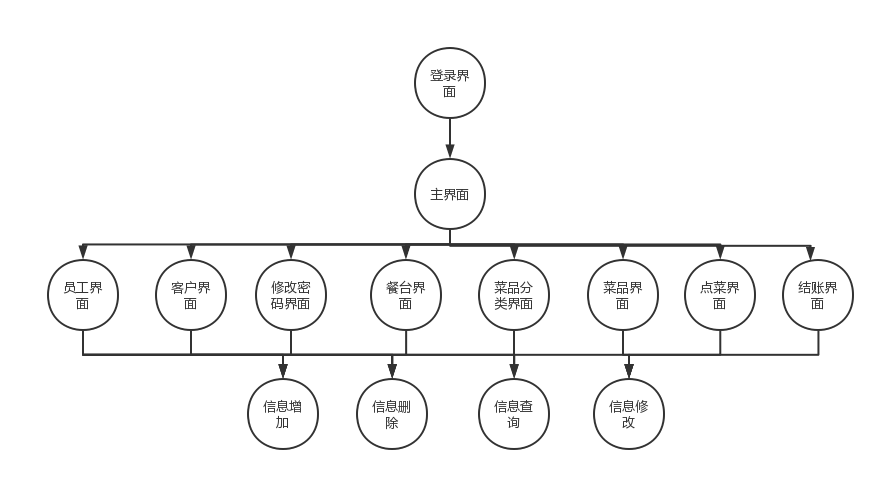
3.3.1 设计应用类

图3.2 餐饮管理的设计类图

3.3.2 登陆界面设计



图3.3登陆界面

登陆界面包括用户、密码的输入，登陆按钮，可以实现登陆操作。为使用者操作方便，输入完用户名后按Enter或者TAB可以切换至密码输入，密码输入完后按Enter即可进行登录操作。

3.3.3 主界面设计

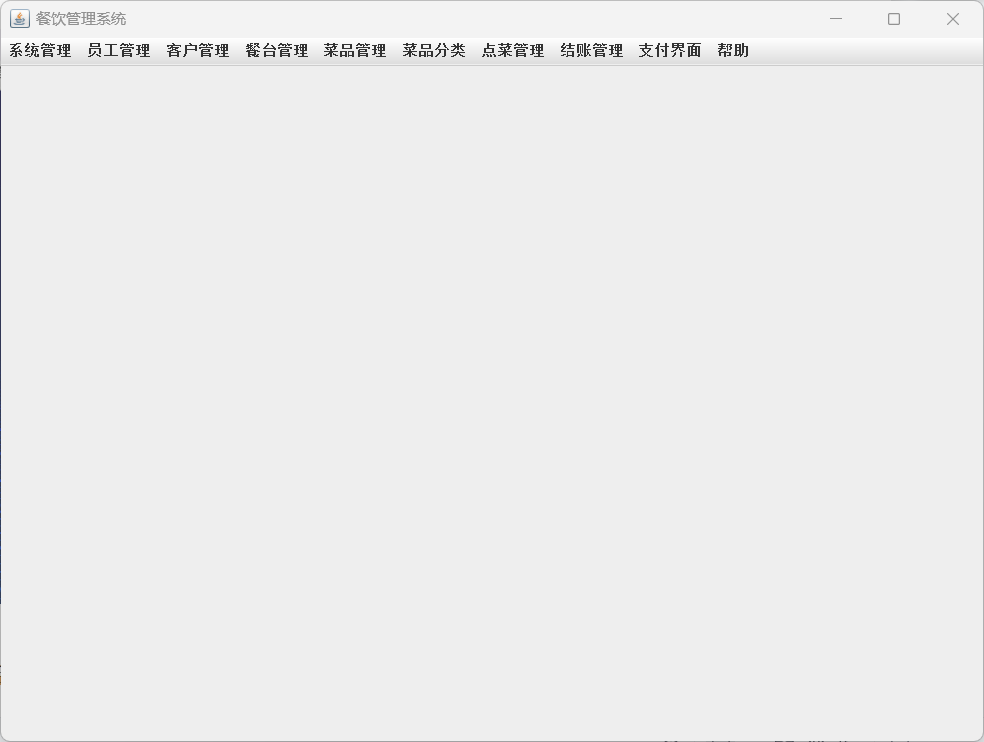


图3.4 系统主界面

主界面包括一行菜单栏，选择菜单栏中的组件可以在主界面的JPanel上进行切换不同界面的操作。

3.3.4 系统信息管理设计



图3.5 系统信息管理

系统管理界面，主要可以添加管理员，修改密码，查询以及删除的操作。

3.3.5 员工信息管理设计

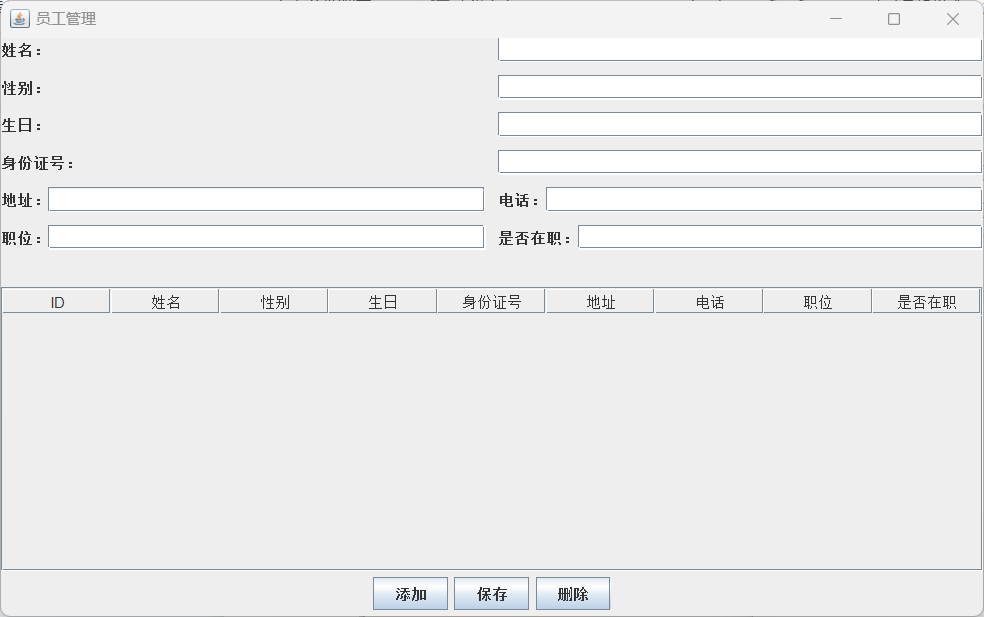
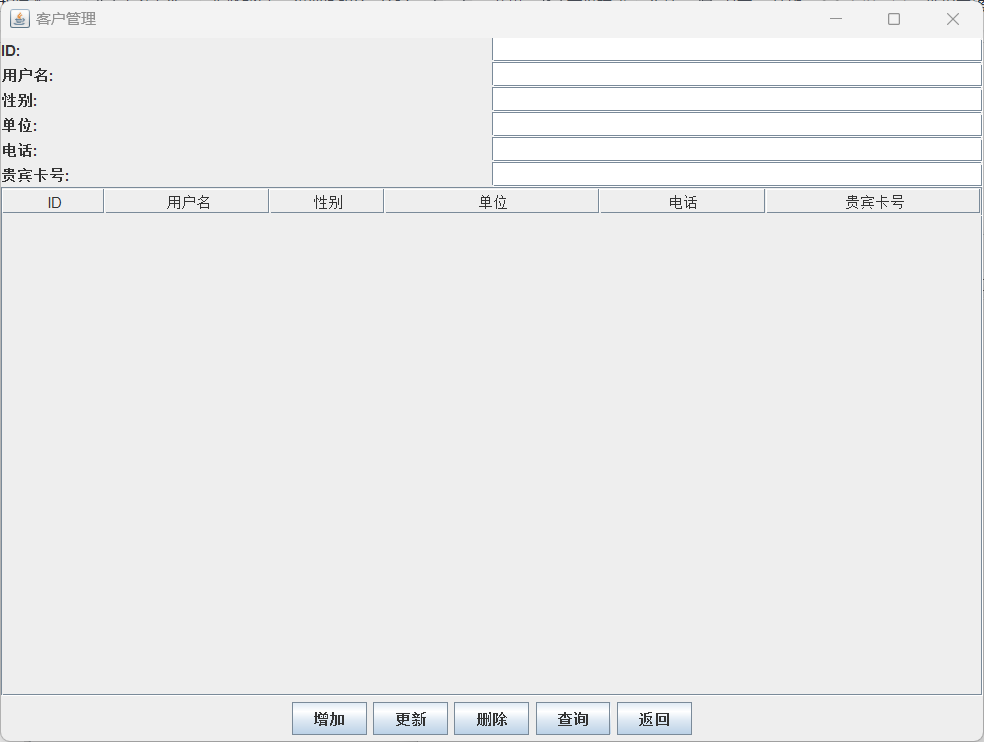


图3.6员工信息管理

员工信息界面包括员工信息表以及表中选中的某一行数据的详细信息，增删改查四个按钮分别进行对员工信息的相应操作。清空按钮进行清空右边数据栏的内容，不会改变员工信息。

3.3.6 客户信息管理设计

图3.7 客户信息管理

客户信息界面包括客户信息表以及表中选中的某一行数据的详细信息，增删改查四个按钮分别进行对客户信息的相应操作。清空按钮进行清空右边数据栏的内容，不会改变客户信息。

3.3.7餐台信息管理设计

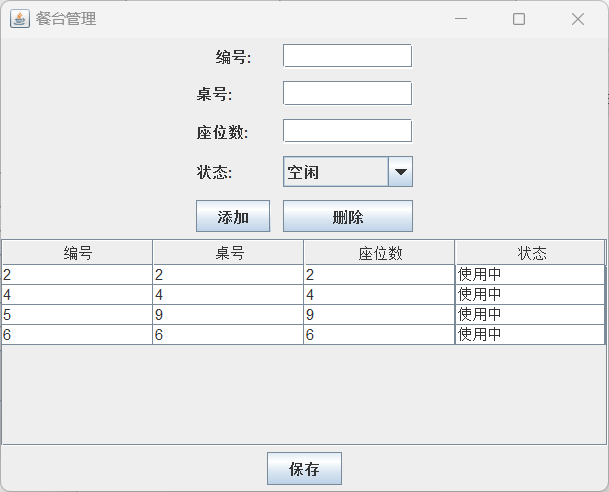


图3.8 餐台信息管理

餐台信息界面包括餐台信息表以及表中选中的某一行数据的详细信息，增删改查四个按钮分别进行对餐台信息的相应操作。清空按钮进行清空右边数据栏的内容，不会改变餐台信息。

3.3.8菜品分类信息管理设计



图3.9 菜品分类信息管理

菜品分类信息界面包括菜品分类信息表以及表中选中的某一行数据的详细信息，增删改查四个按钮分别进行对菜品分类信息的相应操作，清空按钮进行清空右边数据栏的内容，不会改变菜品分类信息。

3.3.9 菜品信息管理设计

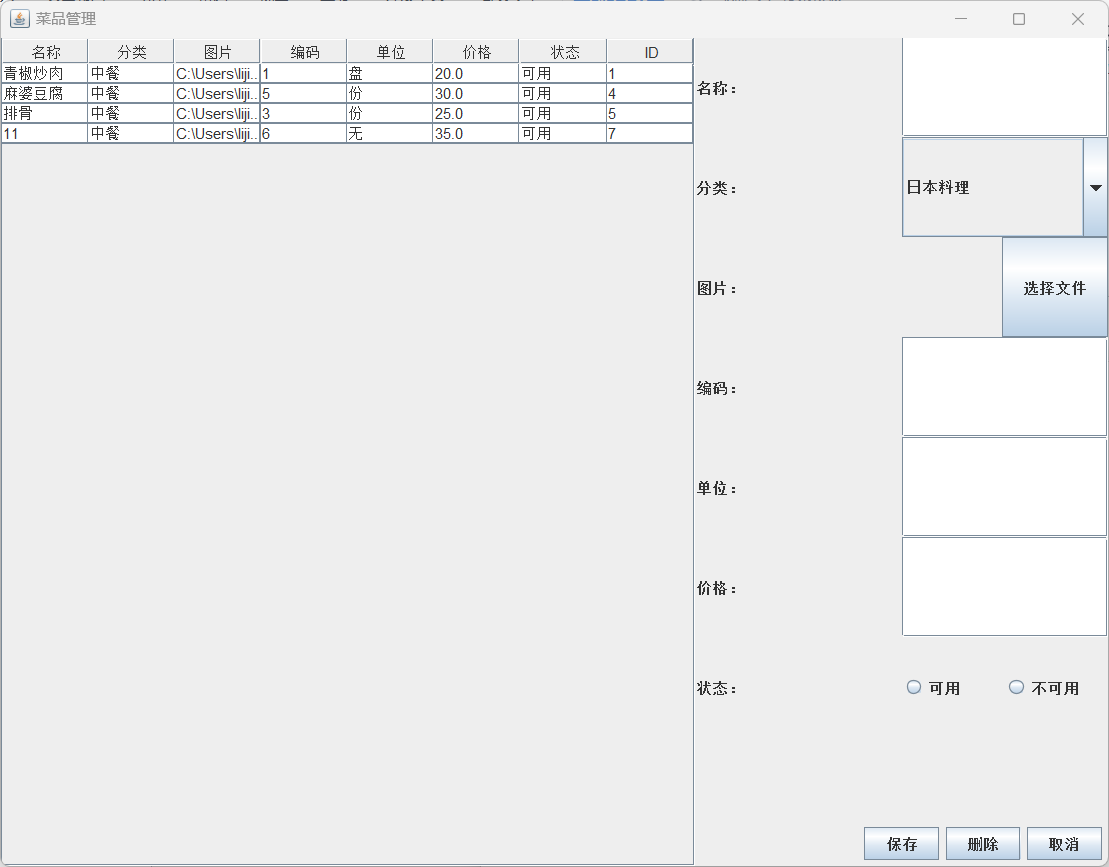


图3.10菜品信息管理

菜品信息界面包括菜品信息表以及表中选中的某一行数据的详细信息，增删改查四个按钮分别进行对菜品信息的相应操作。清空按钮进行清空右边数据栏的内容，不会改变菜品分类信息。

3.3.10 点菜管理设计

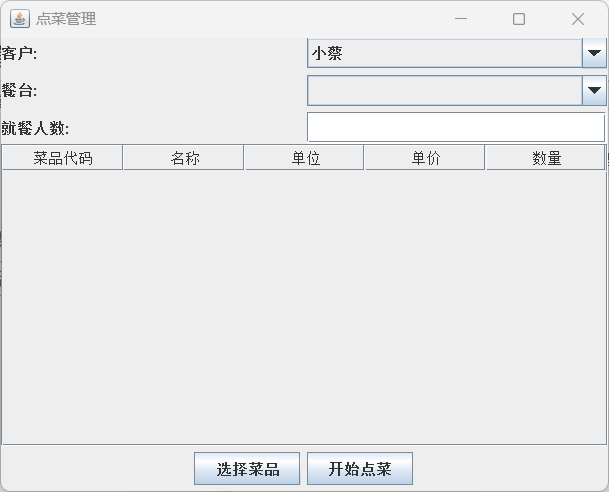


图3.11点菜管理

点菜界面包括菜品信息表以及菜品类别（选中类别后更新表中相应类别的菜品信息），已点菜品表。餐台编号（必须是正在就餐且订单未支付的餐桌），当前订单编号，添加和删除菜品按钮。总金额以及结账按钮。点击结账按钮会弹出结账对话框。

3.3.11 结账管理设计

图3.12结账管理

结账界面包括餐桌编号和客户名，应收金额。输入优惠金额以及实收金额后，点击确认按钮会自动计算找零金额，然后该餐桌状态变为“空闲”，订单状态变为“已支付”。

3.3.12 收银管理设计



图3.13收银管理

收银员结账系统自动统计数据，并显示找零完成结账的操作、

# 4 数据库设计

## 4.1 数据库环境说明

数据库使用MySQL 5.6

## 4.2 数据库的命名规则

1. 数据库名应该简短、具有描述性和易于理解；

2. 数据库名应该使用有意义的字符，不包含特殊字符；

3. 数据库名应该是大小写敏感的，因此应该使用相同的大小写格式来引用数据库名；

4. 数据库名可以包含字符、数字和下划线，但不能以数字开头；

5. 数据库名应该反映其用途或内容；

6. 数据库名应该使用有意义的单词或短语，避免缩写和简写；

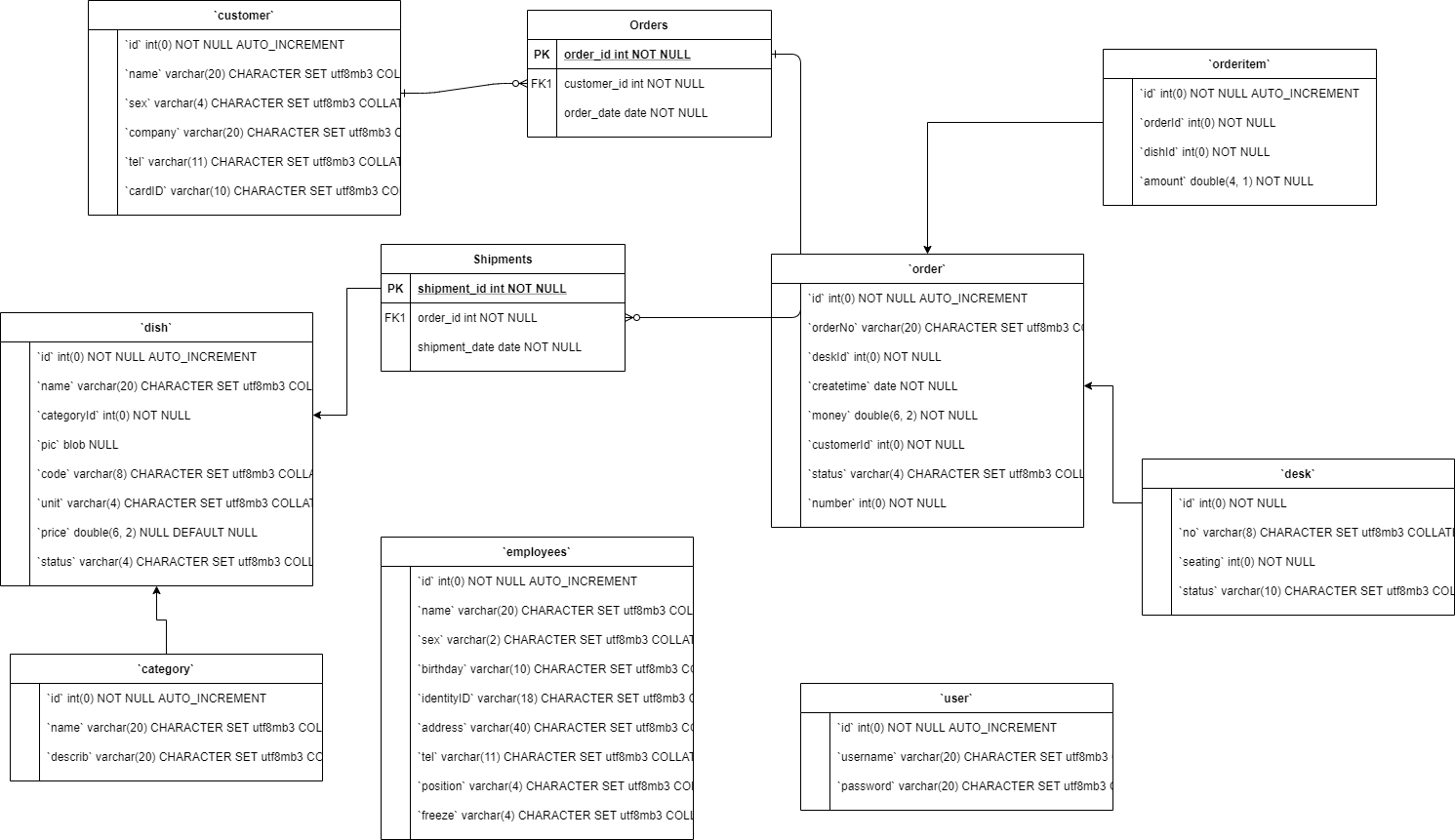
7. 数据库名应该使用单数名称，例如“student”而不是“students”；

8. 数据库名应该避免与保留字冲突，例如SQL关键字。

在实际应用中，还应该考虑公司的命名规则、团队的约定以及未来的扩展性等因素。因此，在确定数据库名称之前，需要进行评估和讨论，以确保名称具有描述性、简洁、直观并且易于维护。

## 4.3 逻辑结构设计

E-R图：



关系模型

user(id,username,password)

employee(id,name,sex,birthday,identityID,address,tel,position,freeze)

customer(id,name,sex,company,tel,cardID)

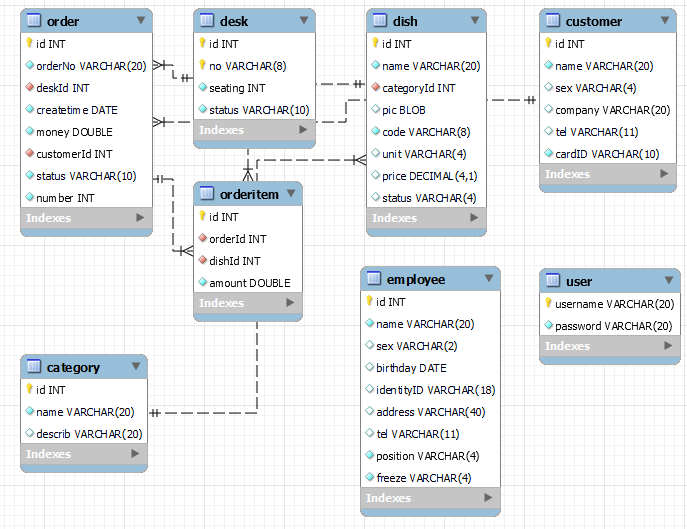
category(id,name,describ)

dish(id,name,pic,code,unit,price,status)

desk(id,no,seating,status)

order(id,orderNo,deskId,creatingtime,money,customerId,status,number)

范式分析

不存在决定因素非码的非平凡函数依赖，因此最高属于BCNF。

逻辑模型图

## 4.4 物理结构设计

表4.1 用户实体user

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| id | int | 10 | no | yes | 序号 |
| username | varchar | 20 | no | no | 用户名 |
| password | varchar | 20 | no | no | 密码 |

表4.2 员工实体employees

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| id | int | 10 | no | yes | 序号 |
| name | varchar | 20 | no | no | 用户名 |
| sex | varchar | 2 | yes | no | 性别 |
| birthday | datatime | 8 | yes | no | 出生日期 |
| identityID | varchar | 18 | yes | no | 身份证号 |
| address | varchar | 40 | yes | no | 家庭住址 |
| tel | varchar | 11 | yes | no | 电话 |
| position | varchar | 4 | no | no | 职位 |
| freeze | varchar | 4 | no | no | 是否在职 |

表4.3 客户实体customer

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| id | int | 10 | no | yes | 序号 |
| name | varchar | 20 | no | no | 用户名 |
| sex | varchar | 4 | yes | no | 性别 |
| company | varchar | 20 | yes | no | 单位 |
| tel | varchar | 11 | yes | no | 电话 |
| cardID | varchar | 10 | no | no | 贵宾卡号 |

表4.4 菜品分类实体category

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| id | int | 10 | no | yes | 序号 |
| name | varchar | 20 | no | no | 名称 |
| describ | varchar | 20 | yes | no | 描述 |

表4.5 菜品信息实体dish

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| id | int | 10 | no | yes | 序号 |
| name | varchar | 20 | no | no | 菜品名 |
| categoryId | int | 10 | no | no | 类别编号 |
| pic | Blob | 50 | no | no | 图片 |
| code | varchar | 8 | no | no | 菜品代码 |
| unit | varchar | 4 | yes | no | 单位 |
| price | double | 6 | yes | no | 价格 |
| status | varchar | 4 | yes | no | 状态 |

表4.6 餐台实体desk

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| id | int | 10 | no | yes | 序号 |
| no | varchar | 8 | no | yes | 餐台编号 |
| seating | int | 4 | no | no | 座位数 |
| status | varchar | 10 | no | no | 状态 |

表4.7 订单实体order

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| id | int | 10 | no | yes | 序号 |
| orderNo | varchar | 20 | no | yes | 订单编号 |
| deskId | int | 10 | no | no | 餐台号 |
| createtime | date | 40 | no | no | 就餐日期时间 |
| money | double | 6 | no | no | 金额 |
| customerId | int | 10 | no | no | 客户编号 |
| status | varchar | 4 | no | no | 状态 |
| number | int | 4 | no | no | 就餐人数 |

表4.8 订单明细实体orderitem

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 类型 | 长度 | 是否为空 | 唯一标识 | 说明 |
| id | int | 10 | no | yes | 序号 |
| orderId | int | 10 | no | no | 订单编号 |
| dishId | int | 10 | no | no | 菜品序号 |
| amount | double | 4 | no | no | 菜品数量 |

# 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’ | admin，111111 | 登录成功 | 通过 |
| yl-002 | admin，123456 | 登录失败 | 通过 |
| yl-003 | admin1，1234567 | 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20 | 登录失败 | 通过 |
| yl-009 | 用户名超过20，123456 | 登录失败 | 通过 |

表5.2 用户管理测试用例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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 |
| 添加 | yl-001 | 输入ID号，‘用户名’和‘密码’ | 1，admin，123456 | 添加成功 | 通过 |
| yl-002 | 1，admin，123456 | 添加失败 | 通过 |
| yl-003 | 空，admin1，123456 | 添加失败 | 通过 |
| yl-004 | 空，admin,空 | 添加失败 | 通过 |
| yl-005 | 空,空，123456 | 添加失败 | 通过 |
| yl-006 | 2，admin,123456 | 添加成功 | 通过 |
| yl-007 | 一个空格，123456 | 添加失败 | 通过 |
| yl-008 | admin，密码长度超过20 | 添加失败 | 通过 |
| yl-009 | 用户名超过20，123456 | 添加失败 | 通过 |
| 删除 | 1 | 选择要删除的一行 | 1，admin，123456 | 删除成功 | 通过 |
| 2 | 不选择 | 请下选择要删除的行 | 通过 |
| 更新 | 1 | 选择要更新的数据 | 1，admin，123456 | 更新成功 | 通过 |
| 2 | 更新密码 | 更新成功 | 通过 |
| 3 | 更新admin，密码长度超过20 | 更新失败 | 通过 |
| 4 | 更新用户名超过20，123456 | 更新失败 | 通过 |
| 5 | 不选择更新数据 | 更新失败 | 通过 |
| 查询 | 1 | 输入ID号 | 1， | 查询成功 | 通过 |
| 2 | 空 | 查询失败 | 通过 |
| 3 | 空格 | 查询失败 | 通过 |
| 4 | 未添加的ID | 查询失败 | 通过 |

表5.3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 |
| 添加 | yl-001 | 输入ID号，‘用户名’和‘性别’，出生日期，身份证号，家庭住址，电话，职位，是否在职 | 1，admin，男，2002-10-19，431124，北京，123456，员工，是 | 添加成功 | 通过 |
| yl-002 | 1，admin，男，空，..... | 添加失败 | 通过 |
| yl-003 | 空，admin1，男,.... | 添加失败 | 通过 |
| yl-004 | 1，admin，男，2002-10-19，431124，北京，123456，员工，是,,重复添加 | 添加失败 | 通过 |
| yl-005 | 空,空，空，全为空 | 添加失败 | 通过 |
| yl-006 | 空格，空格，全为空格 | 添加失败 | 通过 |
| yl-007 | 输入数据类型长度超过设计 | 添加失败 | 通过 |
| yl-008 | 出生日期，输入数据类型错误，如：2002年 | 添加失败 | 通过 |
| yl-009 | 2，admin1，男，2002-10-19，431124，武汉，123456，员工，是 | 添加成功 | 通过 |
| 删除 | 1 | 选择要删除的一行 | 1，admin，男，2002-10-19，431124，长沙，123456，员工，是 | 删除成功 | 通过 |
| 2 | 不选择 | 请下选择要删除的行 | 通过 |
| 保存 | 1 | 添加数据成功后 | 1，admin，男，2002-10-19，431124，长沙，123456，员工，是 | 保存成功 | 通过 |
| 2 | 未添加数据 | 未添加数据 | 通过 |
| 3 | 保存 | 保存成功 | 通过 |

表5.4 客户管理测试用例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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 |
| 添加 | yl-001 | 输入序号，‘用户名’和‘性别’，单位，电话，贵宾卡号 | 1，admin，男，无，123456，666 | 添加成功 | 通过 |
| yl-002 | 1，admin，男，无，123456，，66 添加已有数据 | 添加失败 | 通过 |
| yl-003 | 空，admin，男，无，123456，666 | 添加失败 | 通过 |
| yl-004 | 空，空,空，全为空 | 添加失败 | 通过 |
| yl-005 | 空格，空格，全为空格 | 添加失败 | 通过 |
| yl-006 | 2，admin2，男，无，123456，666 | 添加成功 | 通过 |
| yl-007 | 2，空，男，无，123456，666 | 添加失败 | 通过 |
| yl-008 | 输入数据长度超出 | 添加失败 | 通过 |
| yl-009 | 输入数据类型错误 | 添加失败 | 通过 |
| 删除 | 1 | 选择要删除的一行 | 1，admin，男，无，123456，666 | 删除成功 | 通过 |
| 2 | 不选择 | 请下选择要删除的行 | 通过 |
| 保存 | 1 | 添加数据成功后 | 1，admin，男，无，123456，666 | 保存成功 | 通过 |
| 2 | 未添加数据 | 未添加数据 | 通过 |
| 3 | 保存 | 保存成功 | 通过 |
| 查询 | 1 | 输入序号和姓名 | 1，admin | 查询成功 | 通过 |
| 2 | 空，admin | 查询失败 | 通过 |
| 3 | 1，空 | 查询失败 | 通过 |
| 4 | 未添加的ID，和姓名 | 查询失败 | 通过 |

表5.5餐台管理测试用例

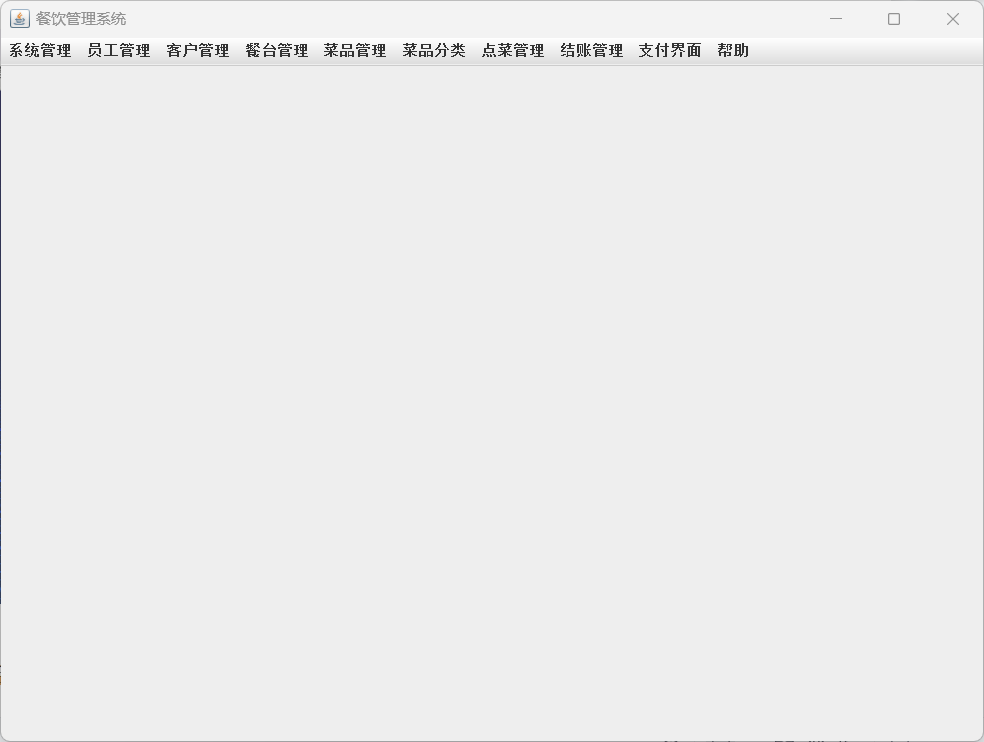
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 功能 | 用例编号 | 操作步骤  （输入条件） | 测试数据 | 预期结果 | 实际结果/判定 |
| 添加 | yl-001 | 输入序号，‘餐台编号’和座位数，状态（已预定，就餐中，已结账） | 1，1，4，已预订（空闲） | 添加成功 | 通过 |
| yl-002 | 1，1，4，已预订（空闲）添加已有数据 | 添加失败 | 通过 |
| yl-003 | 空，1，4，已预订（空闲） | 添加失败 | 通过 |
| yl-004 | 1，空，4，已预订（空闲） | 添加失败 | 通过 |
| yl-005 | 空,空，空，全为空 | 添加失败 | 通过 |
| yl-006 | 空格，空格，全为空格 | 添加失败 | 通过 |
| yl-007 | 输入数据类型长度超过设计 | 添加失败 | 通过 |
| yl-008 | 输入数据类型错误 | 添加失败 | 通过 |
| yl-009 | 2，2，4，已预订（空闲） | 添加成功 | 通过 |
| 删除 | 1 | 选择要删除的一行 | 1，1，4，已预订（空闲） | 删除成功 | 通过 |
| 2 | 不选择 | 请下选择要删除的行 | 通过 |
| 保存 | 1 | 添加数据成功后 | 1，1，4，已预订（空闲） | 保存成功 | 通过 |
| 2 | 未添加数据 | 未添加数据 | 通过 |
| 3 | 保存 | 保存成功 | 通过 |

# 附录

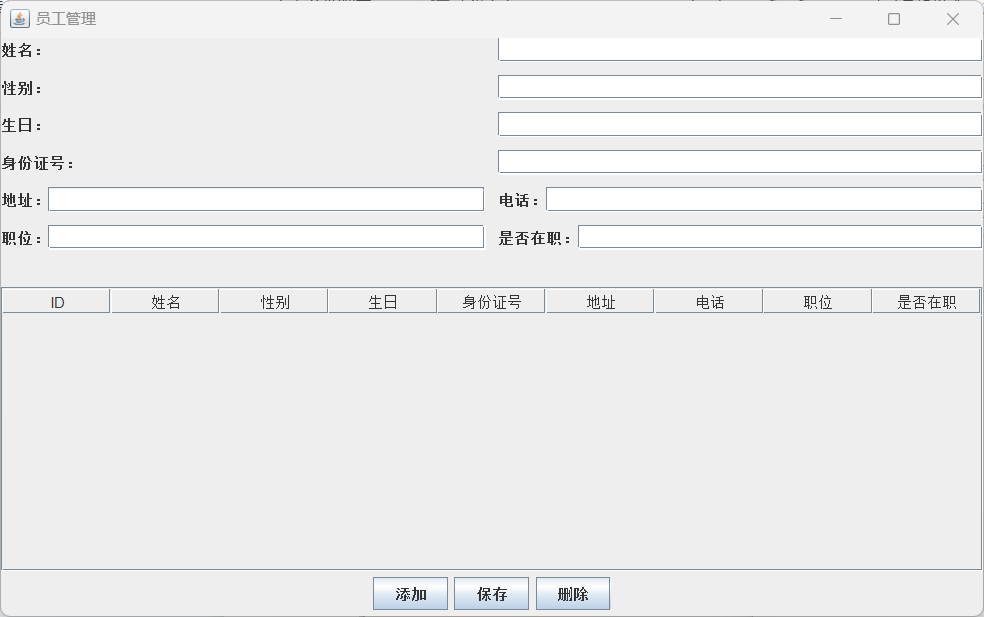
## 附录1 程序运行结果

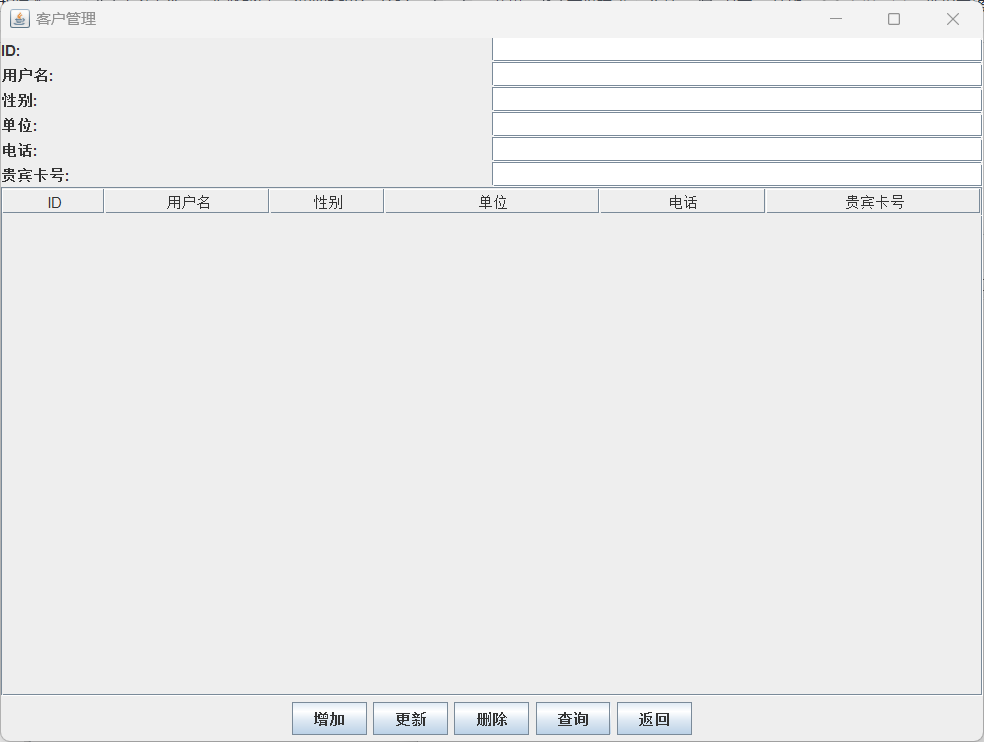


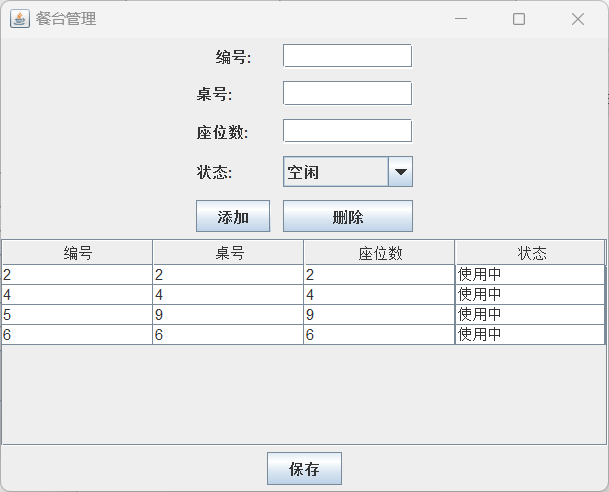
图1 查询界面

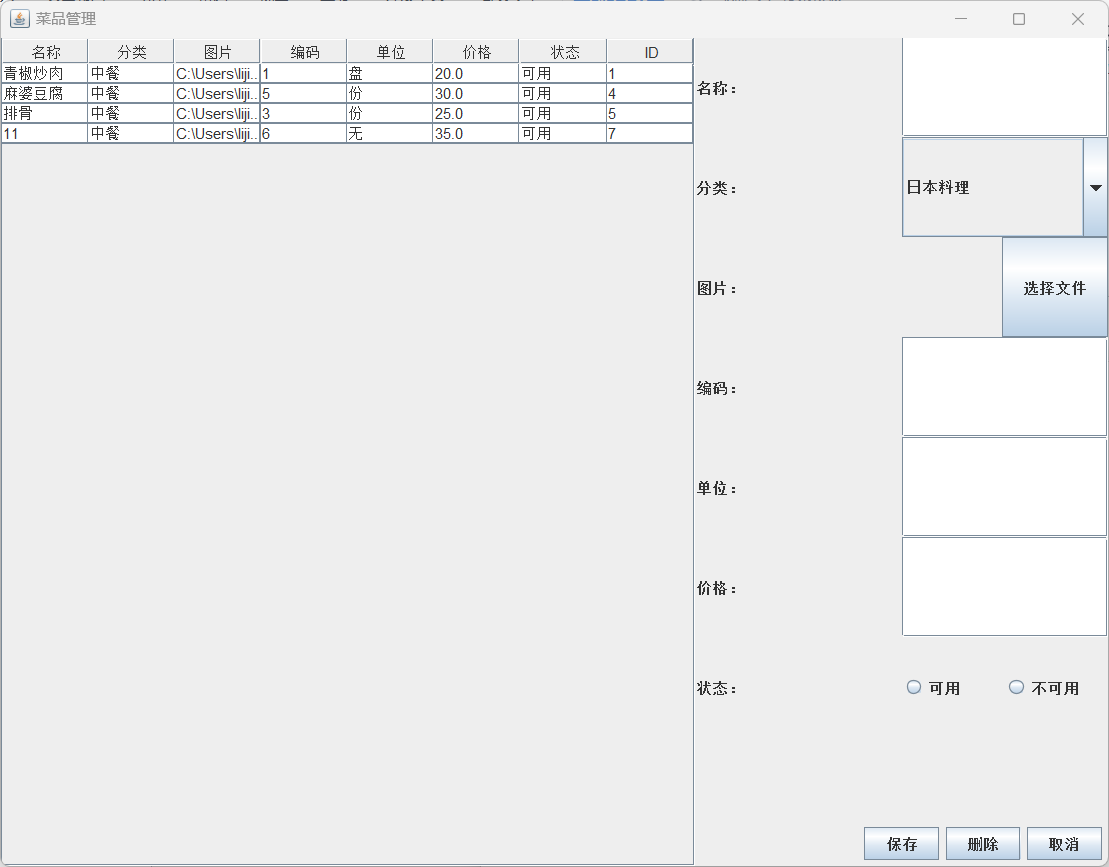


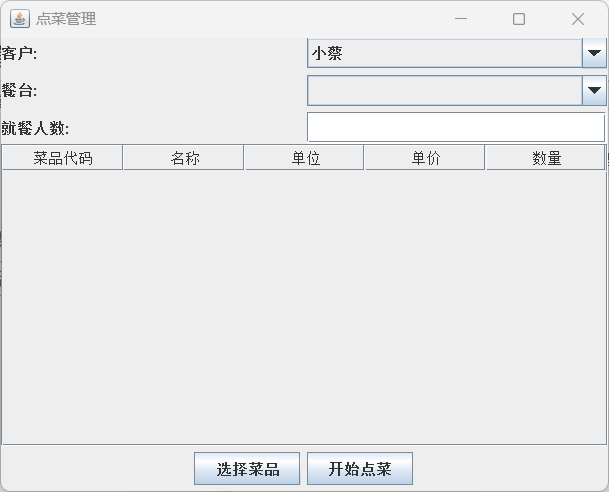


















## 附录2 程序源代码

列出自己设计的主要代码

**登录界面代码：**

public class LoginUI extends JFrame {

private static final String DB\_URL = "jdbc:mysql://127.0.0.1/cysjk?useUnicode=true&characterEncoding=utf8&useSSL=false";

private static final String DB\_USER = "root";

private static final String DB\_PASS = "131420hy";

private JTextField usernameField;

private JPasswordField passwordField;

private JButton loginButton;

public LoginUI() {

super("欢迎进入餐饮管理系统");

// 创建文本框组件

usernameField = new JTextField(20);

passwordField = new JPasswordField(20);

// 创建按钮组件

loginButton = new JButton("登录");

// 添加事件监听器

loginButton.addActionListener(new LoginButtonListener());

// 创建面板并添加组件

JPanel inputPanel = new JPanel(new GridLayout(2, 2));

inputPanel.add(new JLabel("用户名："));

inputPanel.add(usernameField);

inputPanel.add(new JLabel("密码："));

inputPanel.add(passwordField);

JPanel buttonPanel = new JPanel(new FlowLayout());

buttonPanel.add(loginButton);

// 添加组件到窗口中

getContentPane().add(inputPanel, BorderLayout.NORTH);

getContentPane().add(buttonPanel, BorderLayout.SOUTH);

// 设置窗口大小和位置

setSize(400, 150);

setLocationRelativeTo(null);

}

private boolean validateUser(String username, String password) {

try (Connection conn = DriverManager.getConnection(DB\_URL, DB\_USER, DB\_PASS)) {

String sql = "SELECT COUNT(\*) FROM user WHERE username=? AND password=?";

PreparedStatement stmt = conn.prepareStatement(sql);

stmt.setString(1, username);

stmt.setString(2, password);

ResultSet rs = stmt.executeQuery();

rs.next();

return rs.getInt(1) > 0;

} catch (SQLException ex) {

ex.printStackTrace();

return false;

}

}

private class LoginButtonListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

String username = usernameField.getText();

String password = new String(passwordField.getPassword());

if (validateUser(username, password)) {

JOptionPane.showMessageDialog(LoginUI.this, "登录成功");

dispose(); // 关闭登录窗口

// TODO: 跳转到主界面

MainFrame mainFrame = new MainFrame();

mainFrame.setVisible(true);

} else {

JOptionPane.showMessageDialog(LoginUI.this, "用户名或密码错误", "错误", JOptionPane.ERROR\_MESSAGE);

passwordField.setText("");

passwordField.requestFocus();

}

}

}

public static void main(String[] args) {

LoginUI frame = new LoginUI();

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setVisible(true);

}

}

**主界面代码：**

public class MainFrame extends JFrame {

private JMenuBar menuBar;

private JMenu systemMenu;

private JMenuItem userMenuItem;

private JMenuItem tableMenuItem;

private JMenuItem dishMenuItem;

private JMenu businessMenu;

private JMenuItem orderMenuItem;

private JMenuItem payMenuItem;

private JMenu helpMenu;

private JMenuItem aboutMenuItem;

private JMenu tableMenu;

private JMenu dishMenu;

private JMenu payMenu;

private JMenu employeeMenu;

private JMenuItem employeeMenuItem;

private JMenu customerMenu;

private JMenuItem customerMenuItem;

private JMenu ChangeMenu;

private JMenuItem ChangeMenuItem;

private JMenu categoryMenu;

private JMenuItem categoryMenuItem;

public MainFrame() {

this.setTitle("餐饮管理系统");

this.setSize(800, 600);

this.setLocationRelativeTo(null);

this.setDefaultCloseOperation(JFrame.DO\_NOTHING\_ON\_CLOSE); // 修改为DO\_NOTHING\_ON\_CLOSE，不执行默认关闭操作

//添加主界面的背景

JLabel Background = new JLabel();

ImageIcon ico = new ImageIcon("D:\\wuyeguanli.jpg");

Background.setIcon(ico);

this.getContentPane().add(Background);

this.setBounds(200,100,800,500);

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

// 菜单栏

menuBar = new JMenuBar();

systemMenu = new JMenu("系统管理");

userMenuItem = new JMenuItem("用户管理");

tableMenuItem = new JMenuItem("餐台管理");

dishMenu = new JMenu("菜品管理");

dishMenuItem = new JMenuItem("菜品管理");

dishMenu.add(dishMenuItem);

systemMenu.add(userMenuItem);

tableMenu = new JMenu("餐台管理");

tableMenu.add(tableMenuItem);

businessMenu = new JMenu("业务管理");

orderMenuItem = new JMenuItem("订单管理");

payMenuItem = new JMenuItem("结账管理");

payMenu = new JMenu("结账管理");

employeeMenu = new JMenu("员工管理");

employeeMenuItem = new JMenuItem("员工管理");

employeeMenu.add(employeeMenuItem);

customerMenu = new JMenu("客户管理");

customerMenuItem = new JMenuItem("客户管理");

customerMenu.add(customerMenuItem);

ChangeMenu = new JMenu("支付界面");

ChangeMenuItem = new JMenuItem("支付界面");

ChangeMenu.add(ChangeMenuItem);

categoryMenu = new JMenu("菜品分类");

categoryMenuItem = new JMenuItem("菜品分类");

categoryMenu.add(categoryMenuItem);

businessMenu.add(orderMenuItem);

payMenu.add(payMenuItem);

helpMenu = new JMenu("帮助");

aboutMenuItem = new JMenuItem("关于");

helpMenu.add(aboutMenuItem);

menuBar.add(systemMenu);

menuBar.add(employeeMenu);

menuBar.add(customerMenu);

menuBar.add(tableMenu);

menuBar.add(dishMenu);

menuBar.add(categoryMenu);

menuBar.add(businessMenu);

menuBar.add(payMenu);

menuBar.add(ChangeMenu);

menuBar.add(helpMenu);

this.setJMenuBar(menuBar);

userMenuItem.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

UserFrame userFrame = new UserFrame();

userFrame.setDefaultCloseOperation(JFrame.HIDE\_ON\_CLOSE); // 设置关闭操作为隐藏窗口

userFrame.setVisible(true);

}

});

employeeMenuItem.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

EmployeeFrame employeeFrame = new EmployeeFrame();

employeeFrame.setDefaultCloseOperation(JFrame.HIDE\_ON\_CLOSE); // 设置关闭操作为隐藏窗口

employeeFrame.setVisible(true);

}

});

customerMenuItem.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

CustomerFrame customerFrame = new CustomerFrame();

customerFrame.setDefaultCloseOperation(JFrame.HIDE\_ON\_CLOSE); // 设置关闭操作为隐藏窗口

customerFrame.setVisible(true);

}

});

tableMenuItem.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

TableFrame tableFrame = new TableFrame();

tableFrame.setDefaultCloseOperation(JFrame.HIDE\_ON\_CLOSE); // 设置关闭操作为隐藏窗口

tableFrame.setVisible(true);

}

});

dishMenuItem.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

DishFrame dishFrame = new DishFrame();

dishFrame.setDefaultCloseOperation(JFrame.HIDE\_ON\_CLOSE); // 设置关闭操作为隐藏窗口

dishFrame.setVisible(true);

}

});

categoryMenuItem.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

CategoryFrame categoryFrame = new CategoryFrame();

categoryFrame.setDefaultCloseOperation(JFrame.HIDE\_ON\_CLOSE); // 设置关闭操作为隐藏窗口

categoryFrame.setVisible(true);

}

});

orderMenuItem.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

OrderFrame orderFrame = new OrderFrame();

orderFrame.setDefaultCloseOperation(JFrame.HIDE\_ON\_CLOSE); // 设置关闭操作为隐藏窗口

orderFrame.setVisible(true);

}

});

payMenuItem.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

UnpaidOrderFrame payFrame = new UnpaidOrderFrame();

payFrame.setDefaultCloseOperation(JFrame.HIDE\_ON\_CLOSE); // 设置关闭操作为隐藏窗口

payFrame.setVisible(true);

}

});

ChangeMenuItem.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

ChangeFrame ChangeFrame = new ChangeFrame();

ChangeFrame.setDefaultCloseOperation(JFrame.HIDE\_ON\_CLOSE); // 设置关闭操作为隐藏窗口

ChangeFrame.setVisible(true);

}

});

aboutMenuItem.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

JOptionPane.showMessageDialog(MainFrame.this, "餐饮管理系统 作者：卢卓云", "关于系统", JOptionPane.INFORMATION\_MESSAGE);

}

});

// 添加窗口监听器，防止误关闭主窗口

this.addWindowListener(new java.awt.event.WindowAdapter() {

@Override

public void windowClosing(java.awt.event.WindowEvent e) {

int option = JOptionPane.showConfirmDialog(MainFrame.this, "真的要退出程序吗？", "提示", JOptionPane.YES\_NO\_OPTION);

if (option == JOptionPane.YES\_OPTION) {

System.exit(0); // 确认关闭

}

}

});

}

public static void main(String[] args) {

new MainFrame().setVisible(true);

}

}

**系统管理管理员信息**：

public class UserFrame extends JFrame {

static final String JDBC\_DRIVER = "com.mysql.cj.jdbc.Driver";

static final String DB\_URL = "jdbc:mysql://localhost:3306/cysjk";

static final String USERNAME = "root";

static final String PASSWORD = "131420hy";

private JPanel contentPane;

private JTextField idField;

private JTextField usernameField;

private JTextField passwordField;

private Connection conn = null;

private PreparedStatement pstmt = null;

public UserFrame() {

try {

// 加载 MySQL JDBC 驱动程序

Class.forName(JDBC\_DRIVER);

// 建立数据库连接

conn = DriverManager.getConnection(DB\_URL, USERNAME, PASSWORD);

} catch (ClassNotFoundException e) {

e.printStackTrace();

} catch (SQLException e) {

e.printStackTrace();

}

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setBounds(100, 100, 450, 300);

contentPane = new JPanel();

contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));

setContentPane(contentPane);

contentPane.setLayout(null);

JLabel lblId = new JLabel("ID:");

lblId.setBounds(10, 10, 60, 15);

contentPane.add(lblId);

idField = new JTextField();

idField.setBounds(80, 10, 120, 21);

contentPane.add(idField);

idField.setColumns(10);

JLabel lblUsername = new JLabel("用户名:");

lblUsername.setBounds(10, 40, 60, 15);

contentPane.add(lblUsername);

usernameField = new JTextField();

usernameField.setBounds(80, 40, 120, 21);

contentPane.add(usernameField);

usernameField.setColumns(10);

JLabel lblPassword = new JLabel("密码:");

lblPassword.setBounds(10, 70, 60, 15);

contentPane.add(lblPassword);

passwordField = new JTextField();

passwordField.setBounds(80, 70, 120, 21);

contentPane.add(passwordField);

passwordField.setColumns(10);

JButton addButton = new JButton("添加");

addButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int id = Integer.parseInt(idField.getText());

String username = usernameField.getText();

String password = passwordField.getText();

addUser(id, username, password);

}

});

addButton.setBounds(10, 100, 80, 23);

contentPane.add(addButton);

JButton deleteButton = new JButton("删除");

deleteButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int id = Integer.parseInt(idField.getText());

deleteUser(id);

}

});

deleteButton.setBounds(100, 100, 80, 23);

contentPane.add(deleteButton);

JButton updateButton = new JButton("更新");

updateButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int id = Integer.parseInt(idField.getText());

String username = usernameField.getText();

String password = passwordField.getText();

updateUser(id, username, password);

}

});

updateButton.setBounds(190, 100, 80, 23);

contentPane.add(updateButton);

JButton queryButton = new JButton("查询");

queryButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

int id = Integer.parseInt(idField.getText());

queryUser(id);

}

});

queryButton.setBounds(280, 100, 80, 23);

contentPane.add(queryButton);

JScrollPane scrollPane = new JScrollPane();

scrollPane.setBounds(10, 133, 414, 118);

contentPane.add(scrollPane);

JTextArea textArea = new JTextArea();

textArea.setEditable(false);

scrollPane.setViewportView(textArea);

// 添加窗口关闭事件监听器，释放资源

addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent e) {

close();

}

});

}

public void addUser(int id, String username, String password) {

try {

// 实现插入数据操作

String sql = "INSERT INTO user(id, username, password) VALUES (?, ?, ?)";

pstmt = conn.prepareStatement(sql);

pstmt.setInt(1, id);

pstmt.setString(2, username);

pstmt.setString(3, password);

pstmt.executeUpdate();

JOptionPane.showMessageDialog(null, "插入数据成功");

} catch (SQLException e) {

e.printStackTrace();

JOptionPane.showMessageDialog(null, "插入数据失败，请检查输入是否正确");

} finally {

// 关闭PreparedStatement对象

if(pstmt != null) {

try {

pstmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

}

public void deleteUser(int id) {

try {

// 实现删除数据操作

String sql = "DELETE FROM user WHERE id=?";

pstmt = conn.prepareStatement(sql);

pstmt.setInt(1, id);

pstmt.executeUpdate();

JOptionPane.showMessageDialog(null, "删除数据成功");

} catch (SQLException e) {

e.printStackTrace();

JOptionPane.showMessageDialog(null, "删除数据失败，请检查输入是否正确");

} finally {

// 关闭PreparedStatement对象

if(pstmt != null) {

try {

pstmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

}

public void updateUser(int id, String username, String password) {

try {

// 实现更新数据操作

String sql = "UPDATE user SET username=?, password=? WHERE id=?";

pstmt = conn.prepareStatement(sql);

pstmt.setString(1, username);

pstmt.setString(2, password);

pstmt.setInt(3, id);

pstmt.executeUpdate();

JOptionPane.showMessageDialog(null, "更新数据成功");

} catch (SQLException e) {

e.printStackTrace();

JOptionPane.showMessageDialog(null, "更新数据失败，请检查输入是否正确");

} finally {

// 关闭PreparedStatement对象

if(pstmt != null) {

try {

pstmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

}

public void queryUser(int id) {

try {

// 实现查询数据操作

String sql = "SELECT \* FROM user WHERE id=?";

pstmt = conn.prepareStatement(sql);

pstmt.setInt(1, id);

ResultSet rs = pstmt.executeQuery();

JScrollPane scrollPane = (JScrollPane) contentPane.getComponentAt(10, 133);

JTextArea textArea = (JTextArea) scrollPane.getViewport().getView();

textArea.setText("");

while(rs.next()) {

int userId = rs.getInt("id");

String username = rs.getString("username");

String password = rs.getString("password");

textArea.append("id: " + userId + ", username:" + username + ", password: " + password + "");

}

} catch (SQLException e) {

e.printStackTrace();

JOptionPane.showMessageDialog(null, "查询数据失败，请检查输入是否正确");

}finally { // 关闭PreparedStatement对象

if(pstmt != null) {

try {

pstmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

}

public void close() { // 关闭PreparedStatement对象

if(pstmt != null) {

try {

pstmt.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

}

public static void main(String[] args) {

UserFrame frame = new UserFrame();

frame.setVisible(true);

}

}

**员工管理系统**：

package zs;

public class EmployeeFrame extends JFrame {

private static final long serialVersionUID = 1L;

private JPanel contentPane;

private JTextField nameField;

private JTextField sexField;

private JTextField birthdayField;

private JTextField identityIDField;

private JTextArea addressArea;

private JTextField telField;

private JTextField positionField;

private JTextField freezeField;

private JTable table;

private DefaultTableModel model;

private JButton addButton;

private JButton saveButton;

private JButton deleteButton;

private int id;

public EmployeeFrame() {

setTitle("员工管理");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setBounds(100, 100, 800, 500);

contentPane = new JPanel();

setContentPane(contentPane);

contentPane.setLayout(new BorderLayout(0, 0));

JPanel inputPanel = new JPanel();

contentPane.add(inputPanel, BorderLayout.NORTH);

inputPanel.setLayout(new GridLayout(7, 1, 10, 10));

JLabel nameLabel = new JLabel("姓名：");

inputPanel.add(nameLabel);

nameField = new JTextField();

inputPanel.add(nameField);

nameField.setColumns(10);

JLabel sexLabel = new JLabel("性别：");

inputPanel.add(sexLabel);

sexField = new JTextField();

inputPanel.add(sexField);

sexField.setColumns(10);

JLabel birthdayLabel = new JLabel("生日：");

inputPanel.add(birthdayLabel);

birthdayField = new JTextField();

inputPanel.add(birthdayField);

birthdayField.setColumns(10);

JLabel identityIDLabel = new JLabel("身份证号：");

inputPanel.add(identityIDLabel);

identityIDField = new JTextField();

inputPanel.add(identityIDField);

identityIDField.setColumns(18);

JLabel addressLabel = new JLabel("地址：");

JPanel addressPanel = new JPanel(new BorderLayout());

addressPanel.add(addressLabel, BorderLayout.WEST);

addressArea = new JTextArea();

addressPanel.add(new JScrollPane(addressArea), BorderLayout.CENTER);

inputPanel.add(addressPanel);

JLabel telLabel = new JLabel("电话：");

telField = new JTextField();

JPanel telPanel = new JPanel(new BorderLayout());

telPanel.add(telLabel, BorderLayout.WEST);

telPanel.add(telField, BorderLayout.CENTER);

inputPanel.add(telPanel);

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

positionField = new JTextField();

JPanel positionPanel = new JPanel(new BorderLayout());

positionPanel.add(positionLabel, BorderLayout.WEST);

positionPanel.add(positionField, BorderLayout.CENTER);

inputPanel.add(positionPanel);

JLabel freezeLabel = new JLabel("是否在职：");

freezeField = new JTextField();

JPanel freezePanel = new JPanel(new BorderLayout());

freezePanel.add(freezeLabel, BorderLayout.WEST);

freezePanel.add(freezeField, BorderLayout.CENTER);

inputPanel.add(freezePanel);

JPanel buttonPanel = new JPanel();

contentPane.add(buttonPanel, BorderLayout.SOUTH);

addButton = new JButton("添加");

buttonPanel.add(addButton);

saveButton = new JButton("保存");

buttonPanel.add(saveButton);

deleteButton = new JButton("删除");

buttonPanel.add(deleteButton);

model = new DefaultTableModel(new Object[][] {}, new String[] { "ID", "姓名", "性别", "生日", "身份证号", "地址", "电话", "职位", "是否在职" });

table = new JTable(model);

JScrollPane scrollPane = new JScrollPane(table);

contentPane.add(scrollPane, BorderLayout.CENTER);

loadEmployeeData(); // 加载已有的员工数据

addButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

model.addRow(new Object[] {0, nameField.getText(), sexField.getText(), birthdayField.getText(), identityIDField.getText(), addressArea.getText(), telField.getText(), positionField.getText(), freezeField.getText()});

clearFields();

}

});

saveButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=true&useUnicode=true&characterEncoding=utf-8", "root", "131420hy");

PreparedStatement stmt = null;

for (int i = 0; i < model.getRowCount(); i++) {

if (model.getValueAt(i, 1) != null && ((String)model.getValueAt(i, 1)).trim().length() > 0) {

id = (int) model.getValueAt(i, 0);

if (id != 0) {

stmt = conn.prepareStatement("UPDATE employees SET name=?, sex=?, birthday=?, identityID=?, address=?, tel=?, position=?, freeze=? WHERE id=?");

stmt.setString(1, (String)model.getValueAt(i, 1));

stmt.setString(2, (String)model.getValueAt(i, 2));

stmt.setString(3, (String)model.getValueAt(i, 3));

stmt.setString(4, (String)model.getValueAt(i, 4));

stmt.setString(5, (String)model.getValueAt(i, 5));

stmt.setString(6, (String)model.getValueAt(i, 6));

stmt.setString(7, (String)model.getValueAt(i, 7));

stmt.setString(8, (String)model.getValueAt(i, 8));

stmt.setInt(9, id);

} else {

stmt = conn.prepareStatement("INSERT INTO employees (name, sex, birthday, identityID, address, tel, position, freeze) VALUES (?, ?, ?, ?, ?, ?, ?, ?)", Statement.RETURN\_GENERATED\_KEYS);

stmt.setString(1, (String)model.getValueAt(i, 1));

stmt.setString(2, (String)model.getValueAt(i, 2));

stmt.setString(3, (String)model.getValueAt(i, 3));

stmt.setString(4, (String)model.getValueAt(i, 4));

stmt.setString(5, (String)model.getValueAt(i, 5));

stmt.setString(6, (String)model.getValueAt(i, 6));

stmt.setString(7, (String)model.getValueAt(i, 7));

stmt.setString(8, (String)model.getValueAt(i, 8));

}

stmt.executeUpdate();

if (id == 0) {

ResultSet rs = stmt.getGeneratedKeys();

if (rs.next()) {

id = rs.getInt(1);

model.setValueAt(id, i, 0);

}

rs.close();

}

}

}

stmt.close();

conn.close();

JOptionPane.showMessageDialog(null, "保存成功", "提示", JOptionPane.INFORMATION\_MESSAGE);

} catch (SQLException | ClassNotFoundException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(null, "保存失败：" + ex.getMessage(),"错误", JOptionPane.ERROR\_MESSAGE);

}

}

});

deleteButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

int row = table.getSelectedRow();

// 获取选中行的索引，如果没有选中任何行则返回-1。

if (row != -1) {

int confirm = JOptionPane.showConfirmDialog(null, "确定删除该员工信息吗？", "确认删除", JOptionPane.YES\_NO\_OPTION);

if (confirm == JOptionPane.YES\_OPTION) {

int id = (int) model.getValueAt(row, 0);

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=true&useUnicode=true&characterEncoding=utf-8", "root", "131420hy");

PreparedStatement stmt = conn.prepareStatement("DELETE FROM employees WHERE id=?");

stmt.setInt(1, id);

stmt.executeUpdate();

stmt.close();

conn.close();

model.removeRow(row);

clearFields();

JOptionPane.showMessageDialog(null, "删除成功", "提示", JOptionPane.INFORMATION\_MESSAGE);

} catch (SQLException | ClassNotFoundException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(null, "删除失败：" + ex.getMessage(), "错误", JOptionPane.ERROR\_MESSAGE);

}

}

} else {

JOptionPane.showMessageDialog(null, "请先选择要删除的员工信息", "提示", JOptionPane.WARNING\_MESSAGE);

}

}

});

table.getSelectionModel().addListSelectionListener(event -> {

if (table.getSelectedRow() != -1) {

id = (int) model.getValueAt(table.getSelectedRow(), 0);

nameField.setText((String)model.getValueAt(table.getSelectedRow(), 1));

sexField.setText((String)model.getValueAt(table.getSelectedRow(), 2));

birthdayField.setText((String)model.getValueAt(table.getSelectedRow(), 3));

identityIDField.setText((String)model.getValueAt(table.getSelectedRow(), 4));

addressArea.setText((String)model.getValueAt(table.getSelectedRow(), 5));

telField.setText((String)model.getValueAt(table.getSelectedRow(), 6));

positionField.setText((String)model.getValueAt(table.getSelectedRow(), 7));

freezeField.setText((String)model.getValueAt(table.getSelectedRow(), 8));

}

});

}

private void clearFields() {

nameField.setText(null);

sexField.setText(null);

birthdayField.setText(null);

identityIDField.setText(null);

addressArea.setText(null);

telField.setText(null);

positionField.setText(null);

freezeField.setText(null);

}

private void loadEmployeeData() {

try {

Class.forName("com.mysql.cj.jdbc.Driver");

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=true&useUnicode=true&characterEncoding=utf-8", "root", "131420hy");

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM employees");

while (rs.next()) {

int id = rs.getInt("id");

String name = rs.getString("name");

String sex = rs.getString("sex");

String birthday = rs.getString("birthday");

String identityID = rs.getString("identityID");

String address = rs.getString("address");

String tel = rs.getString("tel");

String position = rs.getString("position");

String freeze = rs.getString("freeze");

model.addRow(new Object[] {id, name, sex, birthday, identityID, address, tel, position, freeze});

}

rs.close();

stmt.close();

conn.close();

} catch (SQLException | ClassNotFoundException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(null, "加载数据失败：" + ex.getMessage(),"错误", JOptionPane.ERROR\_MESSAGE);

}

}

public static void main(String[] args) {

SwingUtilities.invokeLater(() -> {

EmployeeFrame frame = new EmployeeFrame();

frame.setVisible(true);

});

}

}

**客户管理系统：**

**package** zs;

**public** **class** CustomerFrame **extends** JFrame {

**private** **static** **final** String ***DB\_URL*** = "jdbc:mysql://localhost:3306/cysjk?useSSL=true&useUnicode=yes&characterEncoding=UTF-8";

**private** **static** **final** String ***DB\_USER*** = "root";

**private** **static** **final** String ***DB\_PASS*** = "131420hy";

**private** JTextField idField, nameField, sexField, companyField, telField, cardIdField;

**private** JButton addButton, updateButton, deleteButton, searchButton;

**private** JTable table;

**public** CustomerFrame() {

**super**("客户管理");

// 创建表格模型

DefaultTableModel model = **new** DefaultTableModel(**new** Object[][] {},

**new** String[] { "ID", "用户名", "性别", "单位", "电话", "贵宾卡号" });

// 创建表格组件

table = **new** JTable(model);

JScrollPane scrollPane = **new** JScrollPane(table);

// 设置表格列宽

table.getColumnModel().getColumn(0).setPreferredWidth(30);

table.getColumnModel().getColumn(1).setPreferredWidth(80);

table.getColumnModel().getColumn(2).setPreferredWidth(40);

table.getColumnModel().getColumn(3).setPreferredWidth(120);

table.getColumnModel().getColumn(4).setPreferredWidth(80);

table.getColumnModel().getColumn(5).setPreferredWidth(120);

// 创建文本框组件

idField = **new** JTextField(10);

nameField = **new** JTextField(20);

sexField = **new** JTextField(5);

companyField = **new** JTextField(30);

telField = **new** JTextField(20);

cardIdField = **new** JTextField(30);

// 创建按钮组件

addButton = **new** JButton("添加");

updateButton = **new** JButton("保存");

deleteButton = **new** JButton("删除");

searchButton = **new** JButton("查询");

// 添加事件监听器

addButton.addActionListener(**new** AddButtonListener());

updateButton.addActionListener(**new** UpdateButtonListener());

deleteButton.addActionListener(**new** DeleteButtonListener());

searchButton.addActionListener(**new** SearchButtonListener());

// 创建面板并添加组件

JPanel inputPanel = **new** JPanel(**new** GridLayout(6, 2));

inputPanel.add(**new** JLabel("ID:"));

inputPanel.add(idField);

inputPanel.add(**new** JLabel("用户名:"));

inputPanel.add(nameField);

inputPanel.add(**new** JLabel("性别:"));

inputPanel.add(sexField);

inputPanel.add(**new** JLabel("单位:"));

inputPanel.add(companyField);

inputPanel.add(**new** JLabel("电话:"));

inputPanel.add(telField);

inputPanel.add(**new** JLabel("贵宾卡号:"));

inputPanel.add(cardIdField);

JPanel buttonPanel = **new** JPanel(**new** FlowLayout());

buttonPanel.add(addButton);

buttonPanel.add(updateButton);

buttonPanel.add(deleteButton);

buttonPanel.add(searchButton);

// 添加组件到窗口中

getContentPane().add(inputPanel, BorderLayout.***NORTH***);

getContentPane().add(scrollPane, BorderLayout.***CENTER***);

getContentPane().add(buttonPanel, BorderLayout.***SOUTH***);

// 设置窗口大小和位置

setSize(800, 600);

setLocationRelativeTo(**null**);

// 加载表格数据

loadTableData();

}

**private** **void** clearFields() {

idField.setText("");

nameField.setText("");

sexField.setText("");

companyField.setText("");

telField.setText("");

cardIdField.setText("");

}

**private** **void** loadTableData() {

DefaultTableModel model = (DefaultTableModel) table.getModel();

// 清空表格数据

model.setRowCount(0);

**try** (Connection conn = DriverManager.*getConnection*(***DB\_URL***, ***DB\_USER***, ***DB\_PASS***)) {

String sql = "SELECT \* FROM customer";

PreparedStatement stmt = conn.prepareStatement(sql);

ResultSet rs = stmt.executeQuery();

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

**int** id = rs.getInt("id");

String name = rs.getString("name");

String sex = rs.getString("sex");

String company = rs.getString("company");

String tel = rs.getString("tel");

String cardId = rs.getString("cardID");

model.addRow(**new** Object[] { id, name, sex, company, tel, cardId });

}

} **catch** (SQLException ex) {

ex.printStackTrace();

}

}

**private** **class** AddButtonListener **implements** ActionListener {

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

String idText = idField.getText();

**if** (idText.isEmpty()) {

JOptionPane.*showMessageDialog*(CustomerFrame.**this**, "请输入有效的ID", "错误", JOptionPane.***ERROR\_MESSAGE***);

**return**;

}

**int** id;

**try** {

id = Integer.*parseInt*(idText);

} **catch** (NumberFormatException ex) {

JOptionPane.*showMessageDialog*(CustomerFrame.**this**, "请输入有效的ID", "错误", JOptionPane.***ERROR\_MESSAGE***);

**return**;

}

String name = nameField.getText();

String sex = sexField.getText();

String company = companyField.getText();

String tel = telField.getText();

String cardId = cardIdField.getText();

**try** (Connection conn = DriverManager.*getConnection*(***DB\_URL***, ***DB\_USER***, ***DB\_PASS***)) {

String sql = "INSERT INTO customer (id, name, sex, company, tel, cardID) VALUES (?, ?, ?, ?, ?, ?)";

PreparedStatement stmt = conn.prepareStatement(sql);

stmt.setInt(1, id);

stmt.setString(2, name);

stmt.setString(3, sex);

stmt.setString(4, company);

stmt.setString(5, tel);

stmt.setString(6, cardId);

stmt.executeUpdate();

loadTableData();

clearFields();

} **catch** (SQLException ex) {

ex.printStackTrace();

}

}

}

**private** **class** UpdateButtonListener **implements** ActionListener {

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

String idText = idField.getText();

**if** (idText.isEmpty()) {

JOptionPane.*showMessageDialog*(CustomerFrame.**this**, "请输入有效的ID", "错误", JOptionPane.***ERROR\_MESSAGE***);

**return**;

}

**int** id;

**try** {

id = Integer.*parseInt*(idText);

} **catch** (NumberFormatException ex) {

JOptionPane.*showMessageDialog*(CustomerFrame.**this**, "请输入有效的ID", "错误", JOptionPane.***ERROR\_MESSAGE***);

**return**;

}

**try** (Connection conn = DriverManager.*getConnection*(***DB\_URL***, ***DB\_USER***, ***DB\_PASS***)) {

String sql = "SELECT \* FROM customer WHERE id=?";

PreparedStatement stmt = conn.prepareStatement(sql);

stmt.setInt(1, id);

ResultSet rs = stmt.executeQuery();

**if** (rs.next()) {

String name = rs.getString("name");

String sex = rs.getString("sex");

String company = rs.getString("company");

String tel = rs.getString("tel");

String cardId = rs.getString("cardID");

// 获取当前记录的其他字段值

String newName = nameField.getText();

String newSex = sexField.getText();

String newCompany = companyField.getText();

String newTel = telField.getText();

String newCardId = cardIdField.getText();

// 只更新修改的字段，保持其他字段不变

**if** (!newName.isEmpty()) {

name = newName;

}

**if** (!newSex.isEmpty()) {

sex = newSex;

}

**if** (!newCompany.isEmpty()) {

company = newCompany;

}

**if** (!newTel.isEmpty()) {

tel = newTel;

}

**if** (!newCardId.isEmpty()) {

cardId = newCardId;

}

// 执行更新操作

sql = "UPDATE customer SET name=?, sex=?, company=?, tel=?, cardID=? WHERE id=?";

stmt = conn.prepareStatement(sql);

stmt.setString(1, name);

stmt.setString(2, sex);

stmt.setString(3, company);

stmt.setString(4, tel);

stmt.setString(5, cardId);

stmt.setInt(6, id);

stmt.executeUpdate();

JOptionPane.*showMessageDialog*(CustomerFrame.**this**, "记录已成功更新", "提示", JOptionPane.***INFORMATION\_MESSAGE***);

loadTableData();

clearFields();

} **else** {

JOptionPane.*showMessageDialog*(CustomerFrame.**this**, "无法找到指定的ID记录", "错误", JOptionPane.***ERROR\_MESSAGE***);

}

} **catch** (SQLException ex) {

ex.printStackTrace();

}

}

}

**private** **class** DeleteButtonListener **implements** ActionListener {

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

String idText = idField.getText();

**int** id = -1;

**if** (!idText.isEmpty()) {

**try** {

id = Integer.*parseInt*(idText);

} **catch** (NumberFormatException ex) {

JOptionPane.*showMessageDialog*(CustomerFrame.**this**, "请输入有效的ID", "错误", JOptionPane.***ERROR\_MESSAGE***);

**return**;

}

}

**try** (Connection conn = DriverManager.*getConnection*(***DB\_URL***, ***DB\_USER***, ***DB\_PASS***)) {

String sql;

PreparedStatement stmt;

**if** (id != -1) {

// 通过ID删除记录

sql = "DELETE FROM customer WHERE id=?";

stmt = conn.prepareStatement(sql);

stmt.setInt(1, id);

} **else** {

// 通过选定行删除记录

**int** selectedRow = table.getSelectedRow();

**if** (selectedRow == -1) {

JOptionPane.*showMessageDialog*(CustomerFrame.**this**, "请选择要删除的行或输入有效的ID", "错误", JOptionPane.***ERROR\_MESSAGE***);

**return**;

}

id = (**int**) table.getValueAt(selectedRow, 0);

sql = "DELETE FROM customer WHERE id=?";

stmt = conn.prepareStatement(sql);

stmt.setInt(1, id);

}

stmt.executeUpdate();

loadTableData();

clearFields();

} **catch** (SQLException ex) {

ex.printStackTrace();

}

}

}

**private** **class** SearchButtonListener **implements** ActionListener {

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

String searchText = nameField.getText().trim();

DefaultTableModel model = (DefaultTableModel) table.getModel();

model.setRowCount(0);

**try** (Connection conn = DriverManager.*getConnection*(***DB\_URL***, ***DB\_USER***, ***DB\_PASS***)) {

String sql = "SELECT \* FROM customer WHERE id=? OR name=? OR sex=? OR company=? OR tel=? OR cardID=?";

PreparedStatement stmt = conn.prepareStatement(sql);

stmt.setString(1, searchText);

stmt.setString(2, searchText);

stmt.setString(3, searchText);

stmt.setString(4, searchText);

stmt.setString(5, searchText);

stmt.setString(6, searchText);

ResultSet rs = stmt.executeQuery();

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

**int** id = rs.getInt("id");

String name = rs.getString("name");

String sex = rs.getString("sex");

String company = rs.getString("company");

String tel = rs.getString("tel");

String cardId = rs.getString("cardID");

**if** (searchText.equals(String.*valueOf*(id)) || searchText.equalsIgnoreCase(name) ||

searchText.equalsIgnoreCase(sex) || searchText.equalsIgnoreCase(company) ||

searchText.equalsIgnoreCase(tel) || searchText.equalsIgnoreCase(cardId)) {

model.addRow(**new** Object[]{id, name, sex, company, tel, cardId});

}

}

} **catch** (SQLException ex) {

ex.printStackTrace();

}

}

}

**public** **static** **void** main(String[] args) {

SwingUtilities.*invokeLater*(() -> {

**try** {

UIManager.*setLookAndFeel*(UIManager.*getSystemLookAndFeelClassName*());

} **catch** (Exception e) {

e.printStackTrace();

}

CustomerFrame frame = **new** CustomerFrame();

frame.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

frame.setVisible(**true**);

});

}

}

**菜品分类管理：**

package zs;

public class CategoryFrame extends JFrame implements ActionListener {

private JLabel label1, label2, label3;

private JTextField tf1, tf2, tf3;

private JButton btn1, btn2, btn3, btn4;

private JTable table;

private DefaultTableModel model;

private Connection conn;

private PreparedStatement pstmt;

private ResultSet rs;

public CategoryFrame() {

super("菜品分类管理");

// 初始化界面组件

label1 = new JLabel("ID：");

label2 = new JLabel("名称：");

label3 = new JLabel("描述：");

tf1 = new JTextField(10);

tf2 = new JTextField(10);

tf3 = new JTextField(10);

btn1 = new JButton("查询");

btn2 = new JButton("添加");

btn3 = new JButton("修改");

btn4 = new JButton("删除");

JPanel p1 = new JPanel(new GridLayout(3, 2));

p1.add(label1);

p1.add(tf1);

p1.add(label2);

p1.add(tf2);

p1.add(label3);

p1.add(tf3);

JPanel p2 = new JPanel(new FlowLayout(FlowLayout.LEFT));

p2.add(btn1);

p2.add(btn2);

p2.add(btn3);

p2.add(btn4);

model = new DefaultTableModel();

model.addColumn("ID");

model.addColumn("名称");

model.addColumn("描述");

table = new JTable(model);

JScrollPane sp = new JScrollPane(table);

JPanel p3 = new JPanel(new BorderLayout());

p3.add(sp, BorderLayout.CENTER);

p3.add(p2, BorderLayout.NORTH);

add(p1, BorderLayout.NORTH);

add(p3, BorderLayout.CENTER);

// 添加事件监听器

btn1.addActionListener(this);

btn2.addActionListener(this);

btn3.addActionListener(this);

btn4.addActionListener(this);

setSize(700, 400);

setLocationRelativeTo(null);

// 连接数据库并显示数据

try {

Class.forName("com.mysql.cj.jdbc.Driver");

conn = DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/cysjk", "root", "131420hy");

displayData();

} catch (ClassNotFoundException e) {

e.printStackTrace();

} catch (SQLException e) {

e.printStackTrace();

}

setSize(500, 300);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setVisible(true);

}

public void actionPerformed(ActionEvent e) {

if (e.getSource() == btn1) { // 查询操作

int id = Integer.parseInt(tf1.getText().trim());

try {

pstmt = conn.prepareStatement("select \* from category where id=?");

pstmt.setInt(1, id);

rs = pstmt.executeQuery();

if (rs.next()) {

tf2.setText(rs.getString("name"));

tf3.setText(rs.getString("describ"));

} else {

JOptionPane.showMessageDialog(null, "未找到该分类！");

}

} catch (SQLException ex) {

ex.printStackTrace();

}

} else if (e.getSource() == btn2) { // 添加操作

String name = tf2.getText().trim();

String describ = tf3.getText().trim();

try {

pstmt = conn.prepareStatement("insert into category(name, describ) values (?, ?)");

pstmt.setString(1, name);

pstmt.setString(2, describ);

int result = pstmt.executeUpdate();

if (result > 0) {

JOptionPane.showMessageDialog(null, "添加成功！");

displayData();

} else {

JOptionPane.showMessageDialog(null, "添加失败！");

}

} catch (SQLException ex) {

ex.printStackTrace();

}

} else if (e.getSource() == btn3) { // 修改操作

int id = Integer.parseInt(tf1.getText().trim());

String name = tf2.getText().trim();

String describ = tf3.getText().trim();

try {

pstmt = conn.prepareStatement("update category set name=?, describ=? where id=?");

pstmt.setString(1, name);

pstmt.setString(2, describ);

pstmt.setInt(3, id);

int result = pstmt.executeUpdate();

if (result > 0) {

JOptionPane.showMessageDialog(null, "修改成功！");

displayData();

} else {

JOptionPane.showMessageDialog(null, "修改失败！");

}

} catch (SQLException ex) {

ex.printStackTrace();

}

} else if (e.getSource() == btn4) { // 删除操作

int id = Integer.parseInt(tf1.getText().trim());

try {

pstmt = conn.prepareStatement("delete from category where id=?");

pstmt.setInt(1, id);

int result = pstmt.executeUpdate();

if (result > 0) {

JOptionPane.showMessageDialog(null, "删除成功！");

displayData();

} else {

JOptionPane.showMessageDialog(null, "删除失败！");

}

} catch (SQLException ex) {

ex.printStackTrace();

}

}

}

private void displayData() throws SQLException {

model.setRowCount(0);

pstmt = conn.prepareStatement("select \* from category");

rs = pstmt.executeQuery();

while (rs.next()) {

Object[] rowData = new Object[3];

rowData[0] = rs.getInt("id");

rowData[1] = rs.getString("name");

rowData[2] = rs.getString("describ");

model.addRow(rowData);

}

}

public static void main(String[] args) {

new CategoryFrame();

}

}

**菜品管理：**

package zs;

class Category implements Serializable {

private int id;

private String name;

public Category(int id, String name) {

this.id = id;

this.name = name;

}

public int getId() {

return id;

}

public String getName() {

return name;

}

@Override

public String toString() {

return name;

}

}

public class DishFrame extends JFrame {

private JTable table;

private DefaultTableModel model;

private JTextField nameField;

private JComboBox categoryComboBox;

private JLabel imageLabel;

private String imagePath;

private JTextField codeField;

private JTextField unitField;

private JTextField priceField;

private JRadioButton availableRadio;

private JRadioButton unavailableRadio;

private int SId = -1;

public DishFrame() {

setTitle("菜品管理");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(900, 700);

JPanel panel = new JPanel(new BorderLayout());

getContentPane().add(panel);

// 表格

table = new JTable();

JScrollPane scrollPane = new JScrollPane(table);

panel.add(scrollPane, BorderLayout.CENTER);

// 表单

JPanel formPanel = new JPanel(new BorderLayout());

JPanel fieldPanel = new JPanel(new GridLayout(0, 2));

formPanel.add(fieldPanel, BorderLayout.NORTH);

fieldPanel.add(new JLabel("名称："));

nameField = new JTextField();

fieldPanel.add(nameField);

fieldPanel.add(new JLabel("分类："));

categoryComboBox = new JComboBox();

loadCategories();

fieldPanel.add(categoryComboBox);

fieldPanel.add(new JLabel("图片："));

JPanel imagePanel = new JPanel(new BorderLayout());

imageLabel = new JLabel();

imageLabel.setPreferredSize(new Dimension(80, 80));

imagePanel.add(imageLabel, BorderLayout.WEST);

JButton imageButton = new JButton("选择图片文件");

imagePanel.add(imageButton, BorderLayout.CENTER);

fieldPanel.add(imagePanel);

fieldPanel.add(new JLabel("编码："));

codeField = new JTextField();

fieldPanel.add(codeField);

fieldPanel.add(new JLabel("单位："));

unitField = new JTextField();

fieldPanel.add(unitField);

fieldPanel.add(new JLabel("价格："));

priceField = new JTextField();

fieldPanel.add(priceField);

fieldPanel.add(new JLabel("状态："));

JPanel statusPanel = new JPanel(new GridLayout(0, 2));

ButtonGroup statusGroup = new ButtonGroup();

availableRadio = new JRadioButton("可用");

unavailableRadio = new JRadioButton("不可用");

statusGroup.add(availableRadio);

statusGroup.add(unavailableRadio);

statusPanel.add(availableRadio);

statusPanel.add(unavailableRadio);

fieldPanel.add(statusPanel);

JButton saveButton = new JButton("保存");

JButton deleteButton = new JButton("删除");

JButton cancelButton = new JButton("取消");

JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.RIGHT));

buttonPanel.add(saveButton);

buttonPanel.add(deleteButton);

buttonPanel.add(cancelButton);

formPanel.add(buttonPanel, BorderLayout.SOUTH);

panel.add(formPanel, BorderLayout.EAST);

// 加载数据

model = new DefaultTableModel(new Object[]{"名称", "分类", "图片", "编码", "单位", "价格", "状态", "ID"}, 0) {

@Override

public boolean isCellEditable(int row, int column) {

return false;

}

};

table.setModel(model);

table.getSelectionModel().addListSelectionListener(new ListSelectionListener() {

@Override

public void valueChanged(ListSelectionEvent e) {

if (!e.getValueIsAdjusting()) {

int row = table.getSelectedRow();

if (row != -1) {

SId = (int) model.getValueAt(row, 7);

nameField.setText((String) model.getValueAt(row, 0));

Category category = (Category) model.getValueAt(row, 1);

categoryComboBox.setSelectedItem(category);

String pic = (String) model.getValueAt(row, 2);

if (pic != null && !pic.isEmpty()) {

imagePath = pic;

ImageIcon icon = new ImageIcon(imagePath);

imageLabel.setIcon(icon);

} else {

imageLabel.setIcon(null);

}

codeField.setText((String) model.getValueAt(row, 3));

unitField.setText((String) model.getValueAt(row, 4));

priceField.setText(model.getValueAt(row, 5).toString());

String status = (String) model.getValueAt(row, 6);

if (status.equals("可用")) {

availableRadio.setSelected(true);

} else {

unavailableRadio.setSelected(true);

}

}

}

}

});

saveButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if (nameField.getText().isEmpty()) {

JOptionPane.showMessageDialog(null, "名称不能为空", "错误", JOptionPane.ERROR\_MESSAGE);

return;

}

if (categoryComboBox.getSelectedItem() == null) {

JOptionPane.showMessageDialog(null, "分类不能为空", "错误", JOptionPane.ERROR\_MESSAGE);

return;

}

if (SId == -1 && imagePath == null) {

JOptionPane.showMessageDialog(null, "请选择一个图片文件，或者先添加一个新的菜品", "错误", JOptionPane.ERROR\_MESSAGE);

return;

}

String name = nameField.getText().trim();

Category category = (Category) categoryComboBox.getSelectedItem();

String code = codeField.getText().trim();

String unit = unitField.getText().trim();

double price = Double.parseDouble(priceField.getText().trim());

String status = availableRadio.isSelected() ? "可用" : "不可用";

if (SId == -1) {

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=true&useUnicode=yes&characterEncoding=UTF-8", "root", "131420hy");

String sql = "INSERT INTO dish(name, categoryId, code, unit, price, status, pic) VALUES(?, ?, ?, ?, ?, ?, ?)";

PreparedStatement stmt = conn.prepareStatement(sql);

stmt.setString(1, name);

stmt.setInt(2, category.getId());

stmt.setString(3, code);

stmt.setString(4, unit);

stmt.setDouble(5, price);

stmt.setString(6, status);

stmt.setString(7, imagePath);

stmt.executeUpdate();

stmt.close();

conn.close();

JOptionPane.showMessageDialog(null, "添加成功！", "提示", JOptionPane.INFORMATION\_MESSAGE);

loadDishes();

resetFields();

} catch (SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(null, "添加失败：" + ex.getMessage(), "错误", JOptionPane.ERROR\_MESSAGE);

}

} else {

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=true&useUnicode=yes&characterEncoding=UTF-8", "root", "131420hy");

String sql = "UPDATE dish SET name=?, categoryId=?, code=?, unit=?, price=?, status=?, pic=? WHERE id=?";

PreparedStatement stmt = conn.prepareStatement(sql);

stmt.setString(1, name);

stmt.setInt(2, category.getId());

stmt.setString(3, code);

stmt.setString(4, unit);

stmt.setDouble(5, price);

stmt.setString(6, status);

stmt.setString(7, imagePath);

stmt.setInt(8, SId);

stmt.executeUpdate();

stmt.close();

conn.close();

JOptionPane.showMessageDialog(null, "修改成功！", "提示", JOptionPane.INFORMATION\_MESSAGE);

loadDishes();

resetFields();

} catch (SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(null, "修改失败：" + ex.getMessage(), "错误", JOptionPane.ERROR\_MESSAGE);

}

}

}

});

deleteButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

if (SId == -1) {

JOptionPane.showMessageDialog(null, "请选择要删除的记录", "错误", JOptionPane.ERROR\_MESSAGE);

return;

}

int dialogResult = JOptionPane.showConfirmDialog(null, "确定要删除吗？", "删除记录", JOptionPane.YES\_NO\_OPTION);

if (dialogResult == JOptionPane.YES\_OPTION) {

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=true&useUnicode=yes&characterEncoding=UTF-8", "root", "131420hy");

String sql = "DELETE FROM dish WHERE id=?";

PreparedStatement stmt = conn.prepareStatement(sql);

stmt.setInt(1, SId);

stmt.executeUpdate();

stmt.close();

conn.close();

JOptionPane.showMessageDialog(null, "删除成功！", "提示", JOptionPane.INFORMATION\_MESSAGE);

loadDishes();

resetFields();

} catch (SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(null, "删除失败：" + ex.getMessage(), "错误", JOptionPane.ERROR\_MESSAGE);

}

}

}

});

cancelButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

resetFields();

}

});

imageButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

JFileChooser fileChooser = new JFileChooser();

FileNameExtensionFilter filter = new FileNameExtensionFilter("图片文件", "jpg", "jpeg", "png", "gif");

fileChooser.setFileFilter(filter);

int result = fileChooser.showOpenDialog(null);

if (result == JFileChooser.APPROVE\_OPTION) {

imagePath = fileChooser.getSelectedFile().getAbsolutePath();

ImageIcon icon = new ImageIcon(imagePath);

imageLabel.setIcon(icon);

}

}

});

setVisible(true);

setLocationRelativeTo(null);

loadDishes();

}

private void loadCategories() {

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=true&useUnicode=yes&characterEncoding=UTF-8", "root", "131420hy");

String sql = "SELECT \* FROM category";

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(sql);

while (rs.next()) {

Category category = new Category(rs.getInt("id"), rs.getString("name"));

categoryComboBox.addItem(category);

}

rs.close();

stmt.close();

conn.close();

} catch (SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(null, "加载分类失败：" + ex.getMessage(), "错误", JOptionPane.ERROR\_MESSAGE);

}

}

private void loadDishes() {

model.setRowCount(0);

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=true&useUnicode=yes&characterEncoding=UTF-8", "root", "131420hy");

String sql = "SELECT dish.\*, category.name AS categoryName FROM dish INNER JOIN category ON dish.categoryId=category.id";

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(sql);

while (rs.next()) {

Category category = new Category(rs.getInt("categoryId"), rs.getString("categoryName"));

model.addRow(new Object[]{

rs.getString("name"),

category,

rs.getString("pic"),

rs.getString("code"),

rs.getString("unit"),

rs.getDouble("price"),

rs.getString("status"),

rs.getInt("id")

});

}

rs.close();

stmt.close();

conn.close();

} catch (SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(null, "加载数据失败：" + ex.getMessage(), "错误", JOptionPane.ERROR\_MESSAGE);

}

}

private void resetFields() {

SId = -1;

nameField.setText("");

categoryComboBox.setSelectedIndex(0);

imageLabel.setIcon(null);

imagePath = null;

codeField.setText("");

unitField.setText("");

priceField.setText("");

availableRadio.setSelected(true);

}

public static void main(String[] args) {

DishFrame frame = new DishFrame();

}

}

、

**餐台管理系统：**

public class TableFrame extends JFrame {

private JLabel idLabel;

private JTextField idField;

private JLabel noLabel;

private JTextField noField;

private JLabel seatingLabel;

private JTextField seatingField;

private JLabel statusLabel;

private JComboBox<String> statusComboBox;

private JButton addButton;

private JButton deleteButton;

private JButton saveButton;

private JTable table;

private DefaultTableModel model;

public TableFrame() {

this.setTitle("餐台管理");

this.setSize(500, 400);

this.setLocationRelativeTo(null);

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

JPanel panel = new JPanel();

panel.setLayout(new GridBagLayout());

GridBagConstraints c = new GridBagConstraints();

idLabel = new JLabel("编号:");

c.gridx = 0;

c.gridy = 0;

c.insets = new Insets(5, 5, 5, 5);

panel.add(idLabel, c);

idField = new JTextField(10);

c.gridx = 1;

c.gridy = 0;

c.fill = GridBagConstraints.HORIZONTAL;

panel.add(idField, c);

noLabel = new JLabel("桌号:");

c.gridx = 0;

c.gridy = 1;

panel.add(noLabel, c);

noField = new JTextField(10);

c.gridx = 1;

c.gridy = 1;

panel.add(noField, c);

seatingLabel = new JLabel("座位数:");

c.gridx = 0;

c.gridy = 2;

panel.add(seatingLabel, c);

seatingField = new JTextField(10);

c.gridx = 1;

c.gridy = 2;

panel.add(seatingField, c);

statusLabel = new JLabel("状态:");

c.gridx = 0;

c.gridy = 3;

panel.add(statusLabel, c);

statusComboBox = new JComboBox<String>();

statusComboBox.addItem("空闲");

statusComboBox.addItem("使用中");

c.gridx = 1;

c.gridy = 3;

panel.add(statusComboBox, c);

addButton = new JButton("添加");

c.gridx = 0;

c.gridy = 4;

c.gridwidth = 1;

panel.add(addButton, c);

deleteButton = new JButton("删除");

c.gridx = 1;

c.gridy = 4;

c.gridwidth = 1;

panel.add(deleteButton, c);

this.add(panel, BorderLayout.NORTH);

// 表格

String[] columnNames = {"编号", "桌号", "座位数", "状态"};

model = new DefaultTableModel(columnNames, 0);

table = new JTable(model);

JScrollPane scrollPane = new JScrollPane(table);

this.add(scrollPane, BorderLayout.CENTER);

// 按钮

JPanel buttonPanel = new JPanel();

saveButton = new JButton("保存");

buttonPanel.add(saveButton);

this.add(buttonPanel, BorderLayout.SOUTH);

addButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String idStr = idField.getText();

String no = noField.getText();

String seatsStr = seatingField.getText();

String status = (String)statusComboBox.getSelectedItem();

if (idStr.isEmpty() || no.isEmpty() || seatsStr.isEmpty()) {

JOptionPane.showMessageDialog(null, "请输入编号、桌号和座位数", "提示", JOptionPane.WARNING\_MESSAGE);

return;

}

try {

int id = Integer.parseInt(idStr);

int seats = Integer.parseInt(seatsStr);

// 添加到表格中

Object[] rowData = {id, no, seats, status};

model.addRow(rowData);

// 添加到数据库中

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=true&useUnicode=yes&characterEncoding=UTF-8", "root", "131420hy");

String sql = "INSERT INTO desk(id, no, seating, status) VALUES(?, ?, ?, ?)";

PreparedStatement stmt = conn.prepareStatement(sql);

stmt.setInt(1, id);

stmt.setString(2, no);

stmt.setInt(3, seats);

stmt.setString(4, status);

stmt.executeUpdate();

// 关闭数据库连接

stmt.close();

conn.close();

} catch (SQLException ex) {

ex.printStackTrace();

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(null, "编号和座位数必须是整数", "提示", JOptionPane.WARNING\_MESSAGE);

return;

}

}

});

deleteButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

int rowIndex = table.getSelectedRow();

if (rowIndex == -1) {

JOptionPane.showMessageDialog(null, "请选择要删除的行", "提示", JOptionPane.WARNING\_MESSAGE);

return;

}

// 从表格中删除

model.removeRow(rowIndex);

// 从数据库中删除

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=true&useUnicode=yes&characterEncoding=UTF-8", "root", "131420hy");

String sql = "DELETE FROM desk WHERE id=" + table.getValueAt(rowIndex, 0);

Statement stmt = conn.createStatement();

stmt.executeUpdate(sql);

// 关闭数据库连接

stmt.close();

conn.close();

} catch (SQLException ex) {

ex.printStackTrace();

}

}

});

saveButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=true&useUnicode=yes&characterEncoding=UTF-8", "root", "131420hy");

// 先清空表中数据

Statement stmt = conn.createStatement();

stmt.executeUpdate("DELETE FROM desk");

// 再保存表格中所有数据

for (int i = 0; i < model.getRowCount(); i++) {

int id = (int)model.getValueAt(i, 0);

String no = (String)model.getValueAt(i, 1);

int seats = (int)model.getValueAt(i, 2);

String status = (String)model.getValueAt(i, 3);

String sql = "INSERT INTO desk(id, no, seating, status) VALUES(" + id + ", '" + no + "', " + seats + ", '" + status + "')";

stmt.executeUpdate(sql);

}

// 关闭数据库连接

stmt.close();

conn.close();

JOptionPane.showMessageDialog(null, "保存成功", "提示", JOptionPane.INFORMATION\_MESSAGE);

} catch (SQLException ex) {

ex.printStackTrace();

}

}

});

// 加载数据库中的数据到表格中

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=true&useUnicode=yes&characterEncoding=UTF-8", "root", "131420hy");

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM desk");

while (rs.next()) {

Object[] rowData = {rs.getInt("id"), rs.getString("no"), rs.getInt("seating"), rs.getString("status")};

model.addRow(rowData);

}

// 关闭数据库连接

rs.close();

stmt.close();

conn.close();

} catch (SQLException ex) {

ex.printStackTrace();

}

}

public static void main(String[] args) {

new TableFrame().setVisible(true);

}

}

**点菜管理：**

public class DishDialog extends JDialog {

private OrderFrame orderFrame;

private DefaultTableModel model;

private JTable table;

public DishDialog(OrderFrame orderFrame) {

super(orderFrame, "选择菜品", true);

this.orderFrame = orderFrame;

setSize(600, 400);

// 表格

String[] columnNames = {"编号", "名称", "单位", "单价"};

model = new DefaultTableModel(columnNames, 0);

table = new JTable(model);

JScrollPane scrollPane = new JScrollPane(table);

getContentPane().add(scrollPane, BorderLayout.CENTER);

// 加载菜品数据到表格中

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=false", "root", "131420hy");

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM dish");

while (rs.next()) {

Object[] rowData = {rs.getString("code"), rs.getString("name"), rs.getString("unit"), rs.getDouble("price")};

model.addRow(rowData);

}

rs.close();

stmt.close();

conn.close();

} catch (SQLException ex) {

ex.printStackTrace();

}

// 确定和取消按钮

JPanel buttonPanel = new JPanel();

JButton okButton = new JButton("确定");

buttonPanel.add(okButton);

JButton cancelButton = new JButton("取消");

buttonPanel.add(cancelButton);

getContentPane().add(buttonPanel, BorderLayout.SOUTH);

okButton.addActionListener(e -> {

int[] rows = table.getSelectedRows();

if (rows.length == 0) {

JOptionPane.showMessageDialog(null, "请选择要点的菜品", "提示", JOptionPane.WARNING\_MESSAGE);

return;

}

DefaultTableModel orderModel = orderFrame.getOrderTableModel();

for (int row : rows) {

String code = (String)model.getValueAt(row, 0);

String name = (String)model.getValueAt(row, 1);

String unit = (String)model.getValueAt(row, 2);

double price = (double)model.getValueAt(row, 3);

Object[] rowData = {code, name, unit, price, 1};

orderModel.addRow(rowData);

}

dispose();

});

cancelButton.addActionListener(e -> dispose());

}

}

**点菜管理（2）：**

public class OrderFrame extends JFrame {

private JLabel customerLabel;

private JComboBox<String> customerComboBox;

private JLabel tableLabel;

private JComboBox<String> tableComboBox;

private JLabel numLabel;

private JTextField numField;

private JButton startButton;

private JTable table;

private DefaultTableModel model;

private int customerId = -1; // 添加一个customerId变量

public OrderFrame() {

this.setTitle("点菜管理");

this.setSize(500, 400);

this.setLocationRelativeTo(null);

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

JPanel panel = new JPanel();

panel.setLayout(new GridLayout(3, 2, 5, 5));

customerLabel = new JLabel("客户:");

panel.add(customerLabel);

customerComboBox = new JComboBox<String>();

panel.add(customerComboBox);

tableLabel = new JLabel("餐台:");

panel.add(tableLabel);

tableComboBox = new JComboBox<String>();

panel.add(tableComboBox);

numLabel = new JLabel("就餐人数:");

panel.add(numLabel);

numField = new JTextField();

panel.add(numField);

this.add(panel, BorderLayout.NORTH);

// 表格

String[] columnNames = {"菜品代码", "名称", "单位", "单价", "数量"};

model = new DefaultTableModel(columnNames, 0);

table = new JTable(model);

JScrollPane scrollPane = new JScrollPane(table);

this.add(scrollPane, BorderLayout.CENTER);

// 按钮

JPanel buttonPanel = new JPanel();

JButton dishButton = new JButton("选择菜品");

buttonPanel.add(dishButton);

startButton = new JButton("开始点菜");

buttonPanel.add(startButton);

this.add(buttonPanel, BorderLayout.SOUTH);

dishButton.addActionListener(e -> showDishDialog());

customerComboBox.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String selectedCustomer = (String) customerComboBox.getSelectedItem();

customerId = getCustomerId(selectedCustomer);

}

});

startButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

String customer = (String)customerComboBox.getSelectedItem();

String table = (String)tableComboBox.getSelectedItem();

String numStr = numField.getText();

if (customer.isEmpty() || table.isEmpty() || numStr.isEmpty()) {

JOptionPane.showMessageDialog(null, "请输入客户、餐台和就餐人数", "提示", JOptionPane.WARNING\_MESSAGE);

return;

}

try {

int num = Integer.parseInt(numStr);

// 计算总金额

double totalMoney = 0.0;

for (int i = 0; i < model.getRowCount(); i++) {

Object value = model.getValueAt(i, 3);

if (value instanceof Double) {

double price = (Double) value;

int quantity = (int)model.getValueAt(i, 4);

totalMoney += price \* quantity;

} else if (value instanceof String) {

double price = Double.parseDouble((String)value);

int quantity = (int)model.getValueAt(i, 4);

totalMoney += price \* quantity;

} else {

// 处理其他类型的数据

}

}

// 添加到数据库中

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=false", "root", "131420hy");

// 生成订单号

SimpleDateFormat sdf = new SimpleDateFormat("yyyyMMddHHmmss");

String orderNo = sdf.format(new Date()) + String.format("%04d", (int)(Math.random() \* 10000));

String sql = "INSERT INTO `order`(orderNo, deskId, createtime, money, customerId, status, number) VALUES(?, ?, ?, ?, ?, ?, ?)";

PreparedStatement stmt = conn.prepareStatement(sql, Statement.RETURN\_GENERATED\_KEYS);

stmt.setString(1, orderNo);

stmt.setInt(2, Integer.parseInt(table));

stmt.setTimestamp(3, new Timestamp(System.currentTimeMillis()));

stmt.setDouble(4, totalMoney); // 将总金额设置为double类型

stmt.setInt(5, customerId); // 将customerId添加到SQL语句中

stmt.setString(6, "未支付"); // 假设默认为未支付

stmt.setInt(7, num);

stmt.executeUpdate();

ResultSet rs = stmt.getGeneratedKeys();

rs.next();

int orderId = rs.getInt(1);

for (int i = 0; i < model.getRowCount(); i++) {

String code = (String)model.getValueAt(i, 0);

int quantity = (int)model.getValueAt(i, 4);

sql = "INSERT INTO orderitem(orderId, dishId, amount) VALUES(?, ?, ?)";

PreparedStatement stmt2 = conn.prepareStatement(sql);

stmt2.setInt(1, orderId);

stmt2.setInt(2, Integer.parseInt(code));

stmt2.setDouble(3, quantity);

stmt2.executeUpdate();

stmt2.close();

}

// 修改desk表中的状态

sql = "UPDATE desk SET status='使用中' WHERE no=?";

PreparedStatement stmt3 = conn.prepareStatement(sql);

stmt3.setString(1, table);

stmt3.executeUpdate();

stmt3.close();

// 关闭数据库连接

rs.close();

stmt.close();

conn.close();

JOptionPane.showMessageDialog(null, "点菜成功", "提示", JOptionPane.INFORMATION\_MESSAGE);

} catch (SQLException ex) {

ex.printStackTrace();

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(null, "就餐人数和菜品数量必须是整数", "提示", JOptionPane.WARNING\_MESSAGE);

return;

} catch (Exception ex) {

ex.printStackTrace();

}

}

});

// 加载客户和餐台列表

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=false", "root", "131420hy");

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM customer");

while (rs.next()) {

customerComboBox.addItem(rs.getString("name"));

}

rs.close();

stmt.close();

stmt = conn.createStatement();

rs = stmt.executeQuery("SELECT \* FROM desk WHERE status='空闲'");

while (rs.next()) {

tableComboBox.addItem(rs.getString("no"));

}

rs.close();

stmt.close();

conn.close();

} catch (SQLException ex) {

ex.printStackTrace();

}

}

private int getCustomerId(String customerName) {

int customerId = -1;

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=false", "root", "131420hy");

String sql = "SELECT id FROM customer WHERE name=?";

PreparedStatement stmt = conn.prepareStatement(sql);

stmt.setString(1, customerName);

ResultSet rs = stmt.executeQuery();

if (rs.next()) {

customerId = rs.getInt("id");

}

rs.close();

stmt.close();

conn.close();

} catch (SQLException ex) {

ex.printStackTrace();

}

return customerId;

}

public DefaultTableModel getOrderTableModel() {

Component comp = getContentPane().getComponent(1);

if (!(comp instanceof JScrollPane)) {

// 第二个组件不是滚动面板，返回一个空模型

return new DefaultTableModel();

}

JTable orderTable = (JTable) ((JScrollPane) comp).getViewport().getView();

return (DefaultTableModel) orderTable.getModel();

}

private void showDishDialog() {

DishDialog dialog = new DishDialog(this);

dialog.setVisible(true);

}

public static void main(String[] args) {

new OrderFrame().setVisible(true);

}

}

**结账管理：**

public class UnpaidOrderFrame extends JFrame {

private static final long serialVersionUID = 1L;

private JPanel contentPane;

private JTable table;

private DefaultTableModel model;

private JButton refreshButton;

public UnpaidOrderFrame() {

setTitle("未支付订单查询");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setBounds(100, 100, 800, 500);

contentPane = new JPanel();

setContentPane(contentPane);

contentPane.setLayout(new BorderLayout(0, 0));

JPanel buttonPanel = new JPanel();

contentPane.add(buttonPanel, BorderLayout.NORTH);

refreshButton = new JButton("刷新");

buttonPanel.add(refreshButton);

model = new DefaultTableModel(new Object[][] {}, new String[] {"订单编号", "餐台编号", "创建日期", "总金额", "客户编号", "状态", "就餐人数"});

table = new JTable(model);

JScrollPane scrollPane = new JScrollPane(table);

contentPane.add(scrollPane, BorderLayout.CENTER);

refreshButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

refreshTable();

}

});

refreshTable();

}

private void refreshTable() {

model.setRowCount(0);

try {

Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/cysjk?useSSL=false", "root", "131420hy");

String sql = "SELECT \* FROM `order` WHERE status='未支付'";

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(sql);

while (rs.next()) {

int id = rs.getInt("id");

String orderNo = rs.getString("orderNo");

int tableId = rs.getInt("deskId");

Timestamp createTime = rs.getTimestamp("createtime");

double totalPrice = rs.getDouble("money");

int customerId = rs.getInt("customerId");

String status = rs.getString("status");

int peopleCount = rs.getInt("number");

model.addRow(new Object[] {orderNo, tableId, createTime, totalPrice, customerId, status, peopleCount});

}

rs.close();

stmt.close();

conn.close();

} catch (SQLException ex) {

ex.printStackTrace();

}

}

public static void main(String[] args) {

UnpaidOrderFrame frame = new UnpaidOrderFrame();

frame.setVisible(true);

}

}

**支付管理：**

public class ChangeFrame extends JFrame implements ActionListener {

private JComboBox<String> deskNumberBox, customerNameBox;

private JTextField receivableAmountField, receivedAmountField, changeAmountField;

public ChangeFrame() {

setTitle("找零界面");

setSize(400, 300);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

JPanel panel = new JPanel();

panel.setLayout(new GridLayout(6, 2));

JLabel deskNumberLabel = new JLabel("餐桌号：");

deskNumberBox = new JComboBox<>();

panel.add(deskNumberLabel);

panel.add(deskNumberBox);

JLabel customerNameLabel = new JLabel("客户名：");

customerNameBox = new JComboBox<>();

panel.add(customerNameLabel);

panel.add(customerNameBox);

JLabel receivableAmountLabel = new JLabel("应收金额：");

receivableAmountField = new JTextField(10);

receivableAmountField.setEditable(false);

panel.add(receivableAmountLabel);

panel.add(receivableAmountField);

JLabel receivedAmountLabel = new JLabel("实收金额：");

receivedAmountField = new JTextField(10);

panel.add(receivedAmountLabel);

panel.add(receivedAmountField);

JLabel changeAmountLabel = new JLabel("找零金额：");

changeAmountField = new JTextField(10);

changeAmountField.setEditable(false);

panel.add(changeAmountLabel);

panel.add(changeAmountField);

JButton confirmButton = new JButton("确认");

confirmButton.addActionListener(this);

panel.add(new JLabel(""));

panel.add(confirmButton);

add(panel);

setSize(800, 600);

setLocationRelativeTo(null);

setVisible(true);

// 初始化餐桌号和客户名列表

initDeskNumbers();

initCustomerNames();

}

private void initDeskNumbers() {

Connection conn = null;

try {

Class.forName("com.mysql.cj.jdbc.Driver");

String url = "jdbc:mysql://localhost:3306/cysjk";

String user = "root";

String password = "131420hy";

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

String query = "SELECT no FROM desk ORDER BY id";

PreparedStatement stmt = conn.prepareStatement(query);

ResultSet rs = stmt.executeQuery();

while (rs.next()) {

deskNumberBox.addItem(rs.getString("no"));

}

rs.close();

stmt.close();

} catch (ClassNotFoundException | SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(this, "连接数据库失败！", "错误", JOptionPane.ERROR\_MESSAGE);

} finally {

if (conn != null) {

try {

conn.close();

} catch (SQLException ex) {

ex.printStackTrace();

}

}

}

}

private void initCustomerNames() {

Connection conn = null;

try {

Class.forName("com.mysql.cj.jdbc.Driver");

String url = "jdbc:mysql://localhost:3306/cysjk";

String user = "root";

String password = "131420hy";

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

String query = "SELECT name FROM customer ORDER BY id";

PreparedStatement stmt = conn.prepareStatement(query);

ResultSet rs = stmt.executeQuery();

while (rs.next()) {

customerNameBox.addItem(rs.getString("name"));

}

rs.close();

stmt.close();

} catch (ClassNotFoundException | SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(this, "连接数据库失败！", "错误", JOptionPane.ERROR\_MESSAGE);

} finally {

if (conn != null) {

try {

conn.close();

} catch (SQLException ex) {

ex.printStackTrace();

}

}

}

}

@Override

public void actionPerformed(ActionEvent e) {

if (e.getActionCommand().equals("确认")) {

String deskNumber = (String) deskNumberBox.getSelectedItem();

String customerName = (String) customerNameBox.getSelectedItem();

// 连接MySQL数据库

Connection conn = null;

try {

Class.forName("com.mysql.cj.jdbc.Driver");

String url = "jdbc:mysql://localhost:3306/cysjk";

String user = "root";

String password = "131420hy";

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

// 查询desk表和customer表获取餐桌号和客户名对应的信息

int deskId = 0;

int customerId = 0;

String deskQuery = "SELECT id FROM desk WHERE no=?";

PreparedStatement deskStmt = conn.prepareStatement(deskQuery);

deskStmt.setString(1, deskNumber);

ResultSet deskRs = deskStmt.executeQuery();

if (deskRs.next()) {

deskId = deskRs.getInt("id");

} else {

JOptionPane.showMessageDialog(this, "该餐桌号不存在！", "错误", JOptionPane.ERROR\_MESSAGE);

return;

}

deskRs.close();

deskStmt.close();

String customerQuery = "SELECT id FROM customer WHERE name=?";

PreparedStatement customerStmt = conn.prepareStatement(customerQuery);

customerStmt.setString(1, customerName);

ResultSet customerRs = customerStmt.executeQuery();

if (customerRs.next()) {

customerId = customerRs.getInt("id");

} else {

JOptionPane.showMessageDialog(this, "该客户名不存在！", "错误", JOptionPane.ERROR\_MESSAGE);

return;

}

customerRs.close();

customerStmt.close();

// 查询order表获取应收金额

double receivableAmount = 0;

String orderQuery = "SELECT money FROM `order` WHERE deskId=? AND customerId=?";

PreparedStatement orderStmt = conn.prepareStatement(orderQuery);

orderStmt.setInt(1, deskId);

orderStmt.setInt(2, customerId);

ResultSet orderRs = orderStmt.executeQuery();

if (orderRs.next()) {

receivableAmount = orderRs.getDouble("money");

} else {

JOptionPane.showMessageDialog(this, "该订单不存在！", "错误", JOptionPane.ERROR\_MESSAGE);

return;

}

orderRs.close();

orderStmt.close();

double receivedAmount = 0;

try {

receivedAmount = Double.parseDouble(receivedAmountField.getText());

} catch (NumberFormatException ex) {

JOptionPane.showMessageDialog(this, "请输入正确的实收金额！", "错误", JOptionPane.ERROR\_MESSAGE);

return;

}

double changeAmount = receivedAmount - receivableAmount;

if (changeAmount < 0) {

JOptionPane.showMessageDialog(this, "实收金额不能小于应收金额！", "错误", JOptionPane.ERROR\_MESSAGE);

} else {

// 修改订单状态为已支付

String updateQuery = "UPDATE `order` SET status='已支付' WHERE deskId=? AND customerId=?";

PreparedStatement updateStmt = conn.prepareStatement(updateQuery);

updateStmt.setInt(1, deskId);

updateStmt.setInt(2, customerId);

updateStmt.executeUpdate();

updateStmt.close();

// 更新desk表中对应餐桌的status属性为"空闲"

String updateDeskQuery = "UPDATE desk SET status='空闲' WHERE id=?";

PreparedStatement updateDeskStmt = conn.prepareStatement(updateDeskQuery);

updateDeskStmt.setInt(1, deskId);

updateDeskStmt.executeUpdate();

updateDeskStmt.close();

receivableAmountField.setText(String.format("%.2f", receivableAmount));

changeAmountField.setText(String.format("%.2f", changeAmount));

JOptionPane.showMessageDialog(this, "支付成功！", "提示", JOptionPane.INFORMATION\_MESSAGE);

}

} catch (ClassNotFoundException | SQLException ex) {

ex.printStackTrace();

JOptionPane.showMessageDialog(this, "连接数据库失败！", "错误", JOptionPane.ERROR\_MESSAGE);

} finally {

if (conn != null) {

try {

conn.close();

} catch (SQLException ex) {

ex.printStackTrace();

}

}

}

}

}

public static void main(String[] args) {

new ChangeFrame();

}

}