





综合实训报告

|  |  |
| --- | --- |
| 学 院 | 计算机科学与工程学院 |
| 课 程 | Java程序设计综合实训 |
| 项目名称 | 记账系统的设计与实现 |
| 班 级 | 软工22101 |
| 组 长 | 张家明2205006483 |
| 成 员 | 高梦丹2205006320 |
|  | 朱泳僖2205006498  卢思彤2205006516 |
| 地 点 | 新工科4301 |
| 时 间 | 2023-2024学年第二学期 |

基于Java的记账系统的设计与实现

# 摘要

Java记账系统是一款基于Java编程语言开发的应用程序，旨在帮助用户记录和管理个人或企业的财务信息。该系统具有用户友好的界面，提供了各种功能，包括记录收入和支出、生成报表和图表、设定预算等。

未来，Java记账系统可以进一步完善功能，如增加数据分析功能、提供自定义报表生成等，以满足用户不断增长的需求。同时，可以考虑开发移动端应用，让用户随时随地方便地记录财务信息。

为了进一步提升Java记账系统的数据安全性，可以考虑引入加密技术，对用户的敏感信息进行加密存储和传输。同时，可以加强用户身份验证机制，如双因素认证，确保只有授权用户可以访问系统。除了手动输入和导入文件外，可以考虑增加扫描识别功能，让用户通过扫描票据或发票快速记录收支信息。同时，也可以集成第三方支付平台，方便用户直接导入交易记录。引入机器学习算法，分析用户的记账习惯和消费行为，为用户提供个性化的记账建议和财务规划。通过智能化推荐，帮助用户更好地管理财务，实现理财目标。为了增加用户互动和参与度，可以添加社交化分享功能，让用户可以分享自己的记账经验、理财心得，与他人交流和学习。同时，也可以开设论坛或社区，建立用户之间的交流平台。

关键词：Java；记账系统；数据分析功能；

Design and Implementation of a Java-Based Accounting System

# Abstract

The Java accounting system is an application developed using the Java programming language, designed to assist users in recording and managing personal or corporate financial information. This system features a user-friendly interface and provides various functionalities, including income and expense tracking, report and chart generation, and budget setting.

In the future, the Java accounting system can be further enhanced with additional features such as data analysis capabilities and customizable report generation to meet the growing needs of users. Additionally, developing a mobile application could enable users to conveniently record financial information anytime and anywhere.

To improve data security, the Java accounting system could implement encryption technologies to secure sensitive user information during storage and transmission. Strengthening user authentication mechanisms, such as two-factor authentication, would ensure that only authorized users can access the system. Beyond manual input and file imports, incorporating scan recognition capabilities could allow users to quickly record income and expenditure information by scanning receipts or invoices. Integration with third-party payment platforms would facilitate direct importing of transaction records.

Introducing machine learning algorithms to analyze user accounting habits and spending behaviors could provide personalized accounting advice and financial planning. Through intelligent recommendations, users could better manage finances and achieve financial goals. To increase user interaction and engagement, social sharing features could be added, enabling users to share accounting experiences and financial insights, and learn from others. Additionally, establishing forums or communities could create a platform for user interaction.

Key words: Java; accounting system; data analysis capabilities

目录

[摘要 I](#_Toc171494997)

[Abstract II](#_Toc171494998)

[1. 绪论 1](#_Toc171494999)

[1.1 课题的研究背景 1](#_Toc171495000)

[1.2 课题的目的和意义 1](#_Toc171495004)

[1.3 课题的研究现状 1](#_Toc171495006)

[1.3.1记账系统的功能丰富化 1](#_Toc171495007)

[1.3.2记账系统的用户体验优化 1](#_Toc171495009)

[1.3.3记账系统的数据安全性 1](#_Toc171495011)

[1.4 论文组织结构 2](#_Toc171495013)

[1.5本章小结 2](#_Toc171495014)

[2. 系统关键技术介绍 3](#_Toc171495015)

[2.1系统的开发环境 3](#_Toc171495016)

[2.2 MySQL 3](#_Toc171495017)

[2.3 Java编程语言 3](#_Toc171495018)

[2.4 Spring Boot 4](#_Toc171495019)

[2.6 本章小结 5](#_Toc171495020)

[3. 需求分析 6](#_Toc171495021)

[3.1可行性分析 6](#_Toc171495022)

[3.1.1经济可行性 6](#_Toc171495023)

[3.1.2技术可行性 6](#_Toc171495024)

[3.1.3操作可行性 6](#_Toc171495025)

[3.1.4法律可行性 6](#_Toc171495026)

[3.2系统分析 6](#_Toc171495027)

[3.3业务需求分析 7](#_Toc171495028)

[3.3.1 业务描述 7](#_Toc171495029)

[3.4功能需求分析 7](#_Toc171495030)

[3.4.1 业务功能 7](#_Toc171495031)

[3.5.1 环境需求 9](#_Toc171495032)

[3.5.2 易用性 9](#_Toc171495033)

[3.5.3 安全性 9](#_Toc171495034)

[3.5.4 可靠性 9](#_Toc171495035)

[3.5.5 可维护性 9](#_Toc171495036)

[3.6 本章小结 9](#_Toc171495037)

[4. 系统设计 10](#_Toc171495038)

[4.1 系统总体设计 10](#_Toc171495039)

[4.2系统功能设计 10](#_Toc171495040)

[4.3系统详细设计 10](#_Toc171495041)

[4.4数据库设计 11](#_Toc171495042)

[4.4.1概念结构设计 11](#_Toc171495043)

[4.4.2逻辑结构设计 12](#_Toc171495047)

[4.5本章小结 13](#_Toc171495044)

[5.功能实现 14](#_Toc171495045)

[5.1系统代码实现 14](#_Toc171495046)

[5.1.1系统登录主界面 14](#_Toc171495047)

[图5.3 登录成功界面 15](#_Toc171495048)

[5.1.2 系统首页界面 15](#_Toc171495049)

[5.1.3文件管理界面 16](#_Toc171495050)

[5.1.4账单信息管理界面 20](#_Toc171495051)

[5.1.5查询信息管理界面 21](#_Toc171495052)

[5.1.6添加分类和用户信息界面 26](#_Toc171495053)

[5.2本章小结 27](#_Toc171495054)

[6.系统测试 28](#_Toc171495055)

[6.1环境测试 28](#_Toc171495056)

[6.2功能测试 28](#_Toc171495057)

[6.2.1用户登录注册模块 28](#_Toc171495058)

[6.2.2文件管理功能模块表 28](#_Toc171495059)

[6.2.3账单信息管理模块 29](#_Toc171495060)

[6.2.4查询功能管理模块 29](#_Toc171495061)

[6.2.5添加分类功能管理模块 29](#_Toc171495062)

[6.2.6关于软件和帮助模块 30](#_Toc171495063)

[6.3本章小结 30](#_Toc171495064)

[7.总结与展望 31](#_Toc171495065)

[参考文献 32](#_Toc171495066)

[附录 32](#_Toc171495067)

[附录 1 32](#_Toc171495068)

[附录2 35](#_Toc171495069)

# 1. 绪论

## 1.1 课题的研究背景

## 随着现代社会经济的发展和人们生活水平的提高，个人和企业的财务管理变得越来越重要。传统的手工记账方式已经无法满足人们对财务管理的需求，因此开发一款高效、便捷的记账系统成为当务之急。

## 在这样的背景下，Java编程语言作为一种广泛应用于软件开发领域的高级编程语言，具有跨平台性、稳定性和灵活性等优势，逐渐成为开发记账系统的首选语言之一。Java记账系统能够利用其强大的功能和丰富的库，为用户提供便捷、安全、高效的财务管理服务。

## 通过对Java记账系统的课题研究，可以进一步完善系统功能，提升用户体验，同时也有助于推动财务管理领域的发展和创新。因此，对Java记账系统进行深入研究具有重要的现实意义和应用前景。

## 1.2 课题的目的和意义

## 开发一款基于Java编程语言的记账系统，实现用户友好的财务管理功能。提供便捷、高效的记账方式，帮助用户记录和管理个人或企业的财务信息。探索Java在财务管理领域的应用，促进技术与金融的结合。为用户提供一个方便、安全的财务管理工具，帮助他们更好地掌握财务状况，实现理财目标。推动Java编程语言在财务管理领域的应用和发展，促进软件技术与金融行业的融合。促进财务管理领域的数字化转型，提高个人和企业财务管理效率，降低管理成本。通过开展Java记账系统课题研究，不仅可以满足用户对财务管理的需求，还可以促进技术创新和行业发展。这一课题的实施具有重要的现实意义和长远意义，有助于推动财务管理领域的进步和改善。

## 1.3 课题的研究现状

通过对记账系统的调查做了如下分析：

1.3.1记账系统的功能丰富化

现有的Java记账系统不仅提供基本的记账功能，还包括预算管理、报表生成、数据分析等高级功能，满足用户不同需求

1.3.2记账系统的用户体验优化

研究者们致力于提升Java记账系统的用户体验，通过界面设计、操作流程优化等手段，使用户能够更轻松地进行财务管理。

1.3.3记账系统的数据安全性

## 针对财务数据的敏感性，研究者们加强了Java记账系统的数据安全性，采用加密技术、权限控制等手段保护用户数据。

## 1.4 论文组织结构

本论文主要是围绕基层党组织的记账系统的设计与实现，构建七个章节分别对记账系统如何实现进行阐述，各章的基本内容如下所示：

第1章，绪论，主要是介绍了基层党组织党建工作的基本建设以及相关研究背景、目的和现状进行分析，并对系统主要研究内容进行叙述。

第2章，关键技术介绍，本章重点阐述了构建记账系统中所使用的关键技术包括Java编程语言、MySQL、Spring Boot框架、Vue框架等。

第3章，需求分析，本章着重对系统的可行性和需求概要进行分析，通过系统功能分析、业务需求分析、功能需求分析、非功能性需求分析，对系统的需求得到了完整的梳理，这些分析对后续的系统实现提供了重要的指引。

第4章，系统设计，本章对系统总体实现进行阐述，包括系统总体设计、系统功能设计、系统详细设计。数据库设计包括物理结构设计、逻辑结构设计、物理结构设计等。这些实现的过程中，参照第三章的需求分析和可行性分析对本系统进行了全面、细致的完善。

第5章，系统功能实现，本章对系统功能实现进行阐述，包含党员发展业务流程管理、支部要闻管理、系统管理、数据统计等主要功能模块的运作情况进行说明。这些功能模块在第三章的需求分析和第四章系统设计的基础上得到了完整的开发和运作。

第6章，系统测试，本章介绍了记账系统核心功能的测试，同时对系统所使用的软件环境和硬件配置进行了相关测试并进行总结。

第7章，总结，本章主要对记账系统的实现进行总结，并对仍然需要改进的问题进行了陈述。通过一系列的实践和总结，记账系统得到了良好的应用和推进。但是随着新技术和新需求的不断涌现，该系统还需要不断地进行更新功能模块和完善信息统计。

## 1.5本章小结

本章叙述了从记账管理系统的研究为背景，再到记账系统工作的研究课题的目的和意义。并针对记账系统所面临的形势，对目前面临的问题进行了分析，并提出解决方案。在最后给出课题的论文组织结构，后续各个章节内容也会将按照此论文组织结构进行阐述。

# 系统关键技术介绍

## 2.1系统的开发环境

为了方便使用，采用客户端和服务器端分离的架构来开发记账系统，使用MySQL5.5数据库系统来存储高校基层党组织的管理建设等相关信息。选择Java编程语言和IDEA开发工具对系统进行编程，具体开发环境如表2.1所示。

表2.1 系统开发环境表

|  |  |
| --- | --- |
| 环境 | 配置 |
| 操作系统 | Windows10专业版64位 |
| 开发平台 | IDEA |
| 开发语言 | Java |
| 数据库 | MySQL5.5 |

## 2.2 MySQL

MySQL是开放源码的小型关联式数据库管理系统，包含了大量的API（应用程序编程接口）与其他的大型系统相比，记账系统的用户承载量是微不足道的，MySQL对SQL算法进行了优化，提升了数据库中的查询速度，并且可以使用管理和检查数据库的工具，考虑预算在内MySQL是首选，可以有效杜绝记账系统呈现信息较慢的麻烦。一个良好的数据库能满足应用系统的业务需求,准确的表达数据间关系。它具有一定的命名规范和明确的业务需求，在数据方面，也能有效的降低数据的冗余性，增强数据的安全性[5]。

## 2.3 Java编程语言

Java编程语言的使用，主要是用于对软件的背景进行撰写，在软件中，使用Java语言来撰写实体类、服务类和界面，并利用配置文件将数据库进行链接。Java是一种纯粹的面向对象编程语言，支持面向对象的编程风格，如封装、继承、多态等。Java内置了对多线程编程的支持，开发者可以方便地创建多线程应用程序，利用现代多核处理器的优势。Java有着广泛的应用领域，从企业级应用到移动应用和嵌入式系统，拥有庞大的开发者社区和成熟的生态系统。总体而言，Java因其跨平台性、安全性、性能和广泛的应用领域而被广泛采用和推崇。具体优点如表2.2所示。

#### 表2.2 Java编程语言开发优点

|  |  |
| --- | --- |
| 优点 | 配置 |
| 使用广泛 | Java是目前使用最为广泛的网络编程语言之一 |
| 面向对象 | Java语言与其它面向对象语言一样，引入了类的概念,类是用来创建对象的模板，它包含被创建的对象的状态描述和方法的定义 |
| 解释型 | Java设计成为解释执行的程序，即翻译一句，执行一句，不产生整个的机器代码程序.翻译过程不出现错误，就一直进行到完毕，否则将在错误处停止执行 |
| 多线程 | Java的特点之一就是内置对多线程的支持，多线程允许同时完成多个任务，实际上多线程使人产生多个任务在同时执行的错觉 |

## 2.4 Spring Boot

Spring Boot的体系结构经过多年的改良，已经越来越完善，和Spring相比，Spring Boot的体系结构更加的关键，Spring Boot可以很方便地与之前的网页进行交互同时Spring Boot可以使用接口进行交互，方便日后进行修改。

Spring Boot开源框架的应用得益于Java的流行，诞生的目的主要针对企业级Java Bean,摆脱其繁琐的开发模式。Spring Boot可以与各种框架相结合使用，适用于构建大型系统应用。但对小型项目而言，仍需要大量配置，影响开发效率。

该框架改进了配置繁琐且容易出现异常的问题，整合了许多第三方库，方便开发人员设计和编写业务逻辑代码，提升了开发的效率。

Spring Boot是一种可以改善并优化Spring架构中复杂的配置缺陷的轻量级架构，其基础是IOC和AOP，这极大地降低了Java的开发难度。然而，在一个大型的项目中，会有许多其它的部件要进行整合，为了构建一个网络应用，至少需要集成MVC框架、Web容器、日志记录工具和对象关系映射框架，如果需要连接数据存储系统，还需要进行相应的选用和集成。如果要启用Spring，那么每个部件都要先去编写它们的配置文件，这样会显得非常麻烦，而且还会出现错误。优于配置的思想，能够大幅度地降低应用中的配置文件数量，让程序开发人员从枯燥乏味的配置文件中解脱出来，将全部精力都放在对业务逻辑的处理上，能够有效地提升应用程序的开发效率。

#### （1）独立运行的Spring项目

使用Spring Boot，可以将应用打包成jar文件，并通过指令“Java-jar xx.jar”来运行应用。

#### （2）内嵌Servlet容器

Spring Boot使用嵌入式的Servlet容器（例如Tomcat、Jetty或者Undertow等），应用无需打成WAR包 。

#### （3）提供starter简化Maven配置

Spring Boot提供了一系列的“starter”项目对象模型（POMS）来简化Maven配置。

#### （4）提供了大量的自动配置

Spring Boot有很多的自动设定，可以使项目变得更容易，并且还可以用这些设定来进行修正。

#### （5）自带应用监控

Spring Boot可以对正在运行的项目提供监控。

#### （6）无代码生成和xml配置

Spring Boot不需要任何xml配置即可实现Spring的所有配置。

2.5 SceneBuilder

SceneBuilder（场景构建器）是一个用于JavaFX的拖放设计工具，允许开发人员在不编写代码的情况下快速设计用户界面。它通过提供用户界面组件的可视化布局，帮助开发人员创建JavaFX应用程序。

使用SceneBuilder，开发人员可以轻松地拖放各种UI组件，设置它们的属性和布局，以及定义事件处理程序，而无需手动编写繁琐的代码。这样可以加快界面设计和开发过程，提高开发效率。

总的来说，SceneBuilder是一个强大的工具，使JavaFX应用程序的设计变得更加直观和高效。

## 2.6 本章小结

本章中，着重分析了记账系统的主要技术，Java作为基础程序设计语言，MySQL作为数据库，运用Spring Boot框架，并使用前后端分离技术来完成该系统的开发；在此基础上，还整合了Spring Security等开放源代码架构以支持该体系的实施。

# 需求分析

## 3.1可行性分析

3.1.1经济可行性

使用MySQL数据库来存储数据，开发记账系统需要的硬件设备和软件环境在市场上容易租借，该方案开发与维护管理体系，不需要高成本的人力和财力投入，因为该体系并不是非常复杂同时开发周期也很短。

3.1.2技术可行性

采用SceneBuilder及MySQL数据库，主要由两大模块组成：要达到功能完善、易用的要求，就必须具有上述特性。既要确保资料的完整，又要确保资料的安全，避免资料被非授权人士盗用。另外，在长期保存的情况下，数据库必须具备较强的稳定性能，才能确保数据不被破坏或盗用。所以，为了保证数据的安全性和完整性，在数据的保存和管理上，都应予以足够的重视。

结论：从设计技术上看，设计记账管理系统是可行的，同时，建设工作中也有较多的实践经验。所以，从技术角度来看，该系统的研制具有一定的可行性。

3.1.3操作可行性

记账系统的登陆界面非常简单，可以利用网页端进行访问操作，党员在日常工作期间都可以进行登陆操作。系统操作简便，管理方便，交互性强使得该系统得以实现。

3.1.4法律可行性

本文介绍了记账系统，它是一种新型的、实用的管理体系，不存在任何的侵权等问题，并且具有充分的合法可能性。总的来说，这一管理体系不论是在技术上、在经济上、在操作上、在法律上都具有高度的可行性，因此，设计并实现该系统是非常有意义的。

## 3.2系统分析

 记账系统是高校信息化发展中重要组成部分，因此，进行了深入的研究和探索，对记账系统的功能设计、系统模块划分和具体设计进行了充分的思考和论证。  
  首先，对记账系统的功能模块进行了设计，包括注册、登录、主界面、导入、导出、备份、恢复、退出、添加、删除、修改、查找、图表、报告等功能。这些功能都要符合记账系统当前现状下的同时实现智能化和信息化，充分发挥现代信息技术的优势。  
  其次，对系统的各功能模块划分进行设计。模块包括用户管理模块、记账模块、报表模块、预算管理模块、数据分析模块、设置模块、数据备份和恢复模块。系统模块划分的合理性和科学性是记账系统有效运行的关键。  
  最后，进行了各个模块的功能测试和整合，不断扩展和完善系统的功能。在测试和整合过程中，高度重视用户体验和系统稳定性，不断优化系统的性能和功能。  
  综上所述，对记账系统发展规范过程进行了深入的研究和探索，并成功地设计记账系统。该系统可以帮助提高记账管理效率。

## 3.3业务需求分析

3.3.1 业务描述

该系统将服务对象：用户。

用户端：

（1）登录注册管理，用户可以输入账号和密码进行登录注册。

（2）导入导出管理，用户可以导出自己的账单信息导excel中，也可以从excel中导入账单信息。

（3）账单信息管理，用户可以对账单信息进行添加，删除，修改，查找等功能。

（4）查找信息管理，用户可以按日期条件查询，可以按分类条件查询，可以按备注条件查询。

（5）添加分类功能管理，用户可以自定义收入和支出分类，并通过右键菜单实现修改和删除。

## 3.4功能需求分析

（1）为保证数据的准确性，对于用户需要填写的数据，系统要有相应的限制并对其进行实时提醒，防止出现输入错误，在遇到计算的算法时，一定要保证精度，以避免造成精度的损失。

（2）实用化，系统中的所有功能应具有一定的意义，是用户所必需应用的。

（3）安全问题，在记账系统中，存在着许多敏感数据，这些数据大多是真实、可靠的，需要对系统进行安全保护，避免数据外泄。

### 3.4.1 业务功能

Java记账系统是一款旨在帮助用户有效管理个人或企业财务的应用程序。其主要业务功能包括账目管理、分类管理、报表生成、预算管理、数据分析、提醒功能、数据备份与恢复、用户管理和多语言支持。

通过账目管理功能，用户可以记录和管理收入和支出的账目，包括金额、日期和分类等信息，实现财务信息的清晰记录和追踪。分类管理功能允许用户自定义收入和支出的分类，便于统计和分析财务数据。报表生成功能可以生成各类报表，如收支明细、月度报表和年度报表，帮助用户全面了解财务状况。

数据备份与恢复功能保证用户数据的安全性，用户管理功能包括注册、登录和权限管理，确保系统安全可靠。多语言支持功能使不同地区用户可以方便地使用系统，提升用户体验。

综上所述，Java记账系统业务功能的完善设计旨在提供用户便捷、高效的财务管理体验，帮助用户实现理财目标，提高财务管理效率。

3.5 非功能性需求

Java记账系统的非功能性需求包括系统的性能、可靠性、安全性、可维护性和可扩展性等方面。系统应具备高性能，能够快速响应用户操作，并支持大量数据的处理和存储。系统的可靠性要求高，确保数据的完整性和可靠性，同时具备自动备份和恢复功能，以防止数据丢失。

可维护性是系统的重要特性，系统应具备良好的代码结构和注释，方便后续维护和升级。系统的模块化设计和清晰的文档说明可以降低维护成本，提高系统的可维护性。

系统的可扩展性也是重要的非功能性需求，系统应具备良好的扩展性，能够方便地添加新功能或模块，以适应未来业务发展的需求。系统的架构应该灵活，支持插件化开发和集成，以便于系统的扩展和定制

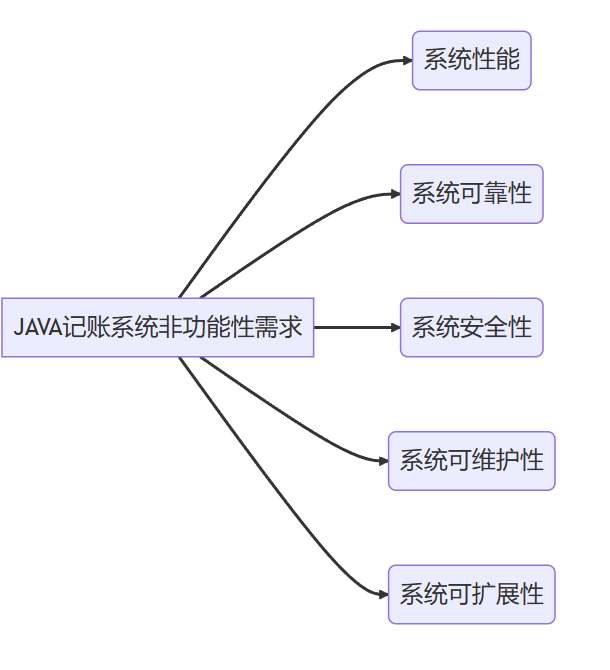
综上所述，Java记账系统的非功能性需求是保证系统性能、可靠性、安全性、可维护性和可扩展性的重要方面，这些需求对系统的设计和开发具有重要的指导意义。

图3.18 非功能性需求图

### 3.5.1 环境需求

服务器环境需求如表3.2所示。

|  |  |
| --- | --- |
| 名称 | 内容 |
| 系统所使用数据库 | MySQL 5.5 |
| 操作系统 | Windows 10 |
| 软件开发环境 | Java |

表3.2 环境需求

### 3.5.2 易用性

易于用性可以使系统与用户习惯的预期相一致，各个用户符合当前用户使用并且符合非功能性需求中易用性的特点。

### 3.5.3 安全性

安全是指一种系统能够避免可能出现的危险。为了防止资料外泄，必须根据实际情况，为每类用户分配合理的权限。在这一模块中，必须对其进行严谨的设计，以便该系统的安全系数达到其所能达到的要求。

### 3.5.4 可靠性

系统功能完善，设计严谨，均可反映出每个阶段用户的状态以及用户的发展业务流程， 而可靠性则可以在特定的时期和特定的情况下，保持其正常的工作状态，并保持良好的运行状态。

### 3.5.5 可维护性

在没有人为介入的情况下，系统的稳定性、自排错和可测试性都是在此范围内，该系统提供独立的用户管理、统计管理、身份管理、记账流程管理等功能，这些功能通常会极大决定系统的运行维护成本以及使用维护的难度

## 3.6 本章小结

本章主要对基于Java的记账系统的要求进行了深入的探讨，主要内容有：对记账功能的业务需求，功能需求以及非功能性需求等。经过研究与分析，认为具备实施的必要条件。

# 系统设计

## 4.1 系统总体设计

以Java为基础的记账系统，采用了SceneBuilder场景构建，与此同时，系统还可以通过增加模块来实现更多的功能，编辑信息来实现用户信息同步的更新。

## 4.2系统功能设计

用户可以根据基层支部的具体情况，增加不同的用户并为其赋予不同的身份，从而在该系统上实现不同的应用要求同时可以添加不同身份的用户并赋予不同点权限。这意味着可以控制普通用户的访问权限，以保证系统的安全性。此外，用户还可以查看所有用户信息，实时更新组织发展工作流程信息。

## 4.3系统详细设计

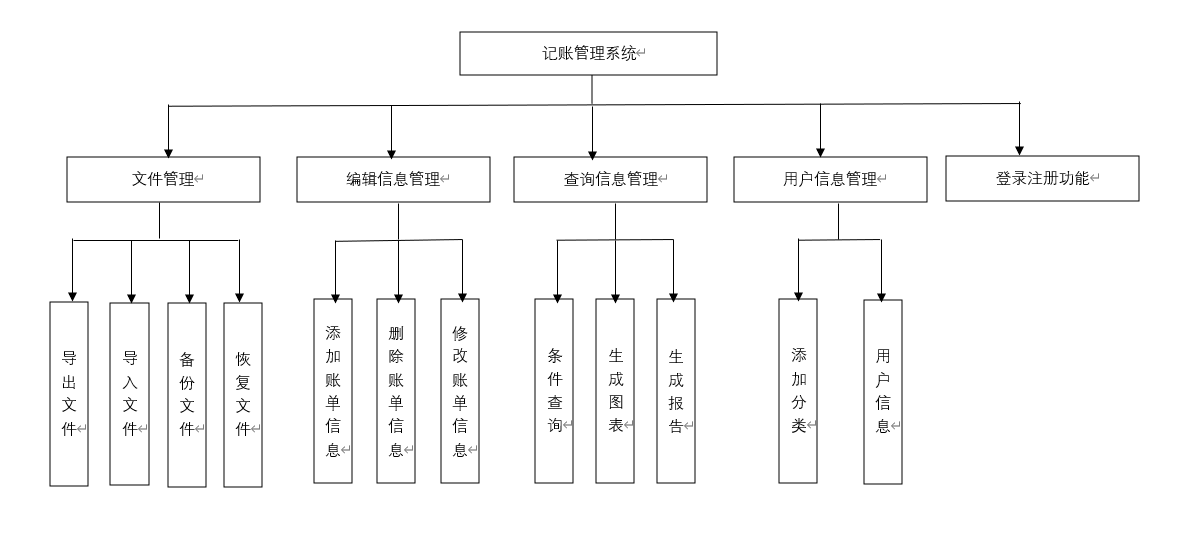
随着信息技术的快速发展，越来越多的组织开始重视信息化建设。在这一背景下，记账系统的出现，是为党建工作的信息化管理提供了更多的解决方案。该系统由多个模块组成，包括支部动态管理、个人信息管理、党费缴纳管理、系统管理和数据统计管理等。能够方便支部书记对党内各项工作进行跟踪和管理，提高工作效率的同时也能够方便党员进行实时学习和参加党内组织的各项活动。  
系统功能设计图如图4.1所示

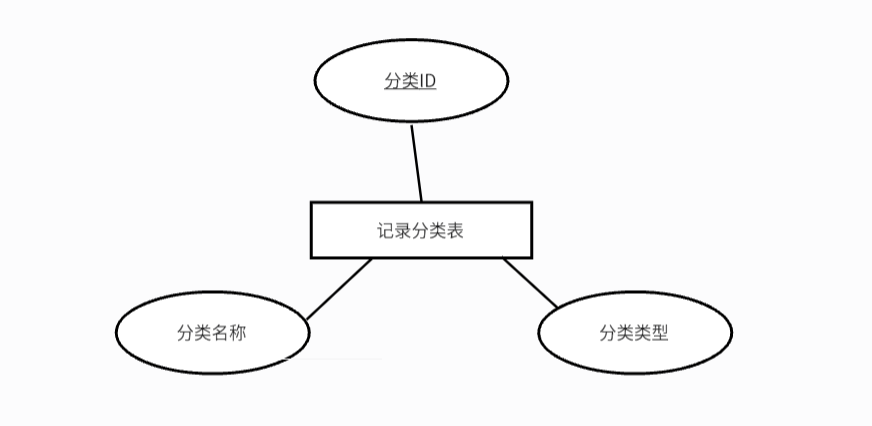
图4.1 记账管理系统功能模块图

## 4.4数据库设计

数据库的重要性在所有系统中是最基本的部件，无论是信息的输入、处理、存储、查询、更新、删除等各种操作都需要通过数据库进行。而数据库的建设质量则直接关系到整个信息管理系统的运行效率和质量。为了建立一个高效、可靠、易于维护的数据库，设计理念必须是基于对数据的深入分析，由下至上地设计出一个完整的数据库概念结构。这意味着需要先从最基本的数据元素开始，逐一分析其特性、关系、约束条件等，并按照一定的规则将其组织成数据表。接着，需要对不同的数据表之间的关系进行分析，确定它们之间的连接方式和级别，最终形成一个完整的数据库概念结构。

### 4.4.1概念结构设计

记账系统可以满足的需要比较多，因此，实体也比较多，主要包括了记录分类表，记录表，用户表等。

基于Java的记账系统的实体及其关系如下：

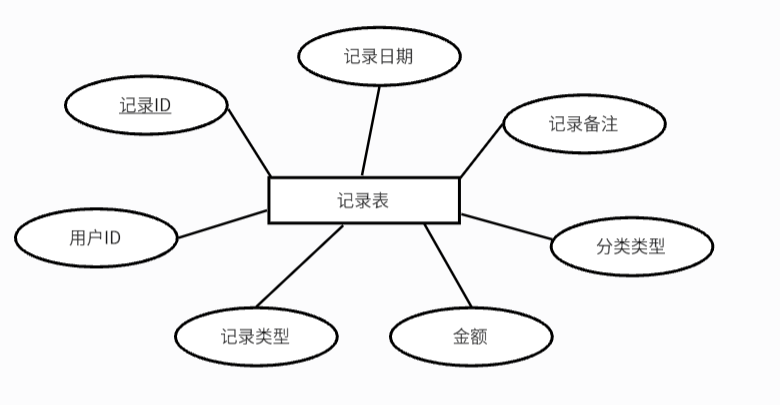
图4.2 记录分类表实体E-R图

图4.3 记录表实体E-R图

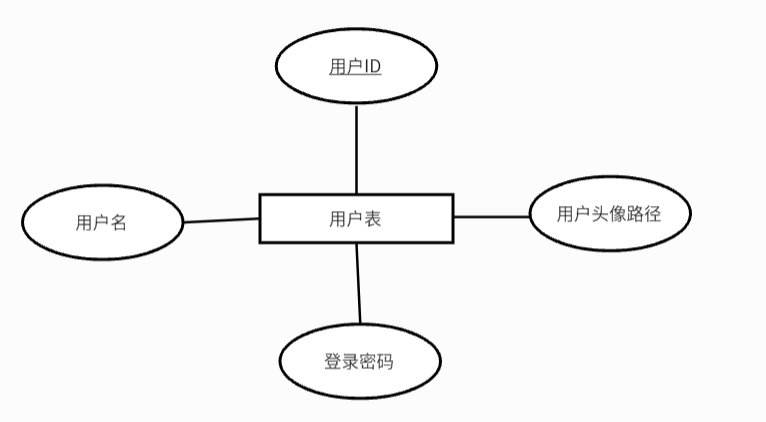


图4.4 用户表实体E-R图

4.4.2逻辑结构设计

根据上述主要E-R图设计、实体之间转换规则以及系统实际需求，下面罗列出该系统的所有数据库表设计。

记录(记录ID，用户ID，记录类型，金额，分类类型，记录备注，记录日期。)

用户(用户ID，用户名，登录密码，用户头像类型)

记录分类 (分类ID，分类名称，分类类型)

记录分类表的结构如表4.1所示。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 允许空值 | 字段说明 | 备注 |
| cID | Int | 否 | 分类ID | 主键 |
| cName | Varchar(20) | 否 | 分类名称 |  |
| cType | Varchar(20) | 否 | 分类类型 |  |

表4.1 记录分类表

记录表的结构如表4.2所示。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 允许空值 | 字段说明 | 备注 |
| rID | Int | 否 | 记录ID | 主键 |
| uID | Int | 否 | 用户ID |  |
| rType | Varchar(20) | 否 | 记录类型 |  |
| rMoney | Float | 否 | 金额 |  |
| rClassification | Varchar(20) | 否 | 分类类型 |  |
| rMemo | Varchar(3000) | 否 | 记录备注 |  |
| rDate | date | 否 | 记录日期 |  |

表4.2 记录表

用户表的结构如表4.3所示

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名称 | 数据类型 | 允许空值 | 字段说明 | 备注 |
| uID | Int | 否 | 用户ID | 主键 |
| uName | Varchar(20) | 否 | 用户名 |  |
| uPassword | Varchar(40) | 否 | 登录密码 |  |
| uImagepath | Varchar(200) | 是 | 用户头像路径 |  |

表4.3 用户表

## 4.5本章小结

本文详细介绍了记账系统的设计方案，并探讨了数据库系统方面的设计，以及采用前后端分离技术的设计优势。  
   在设计数据库系统方面，本文探讨了数据库系统的概念结构设计、物理结构设计和逻辑设计等方面。这些方面的设计在系统开发过程中起到至关重要的作用，能够有效提高系统的稳定性和可维护性。

# 5.功能实现

## 5.1系统代码实现

### 5.1.1系统登录主界面

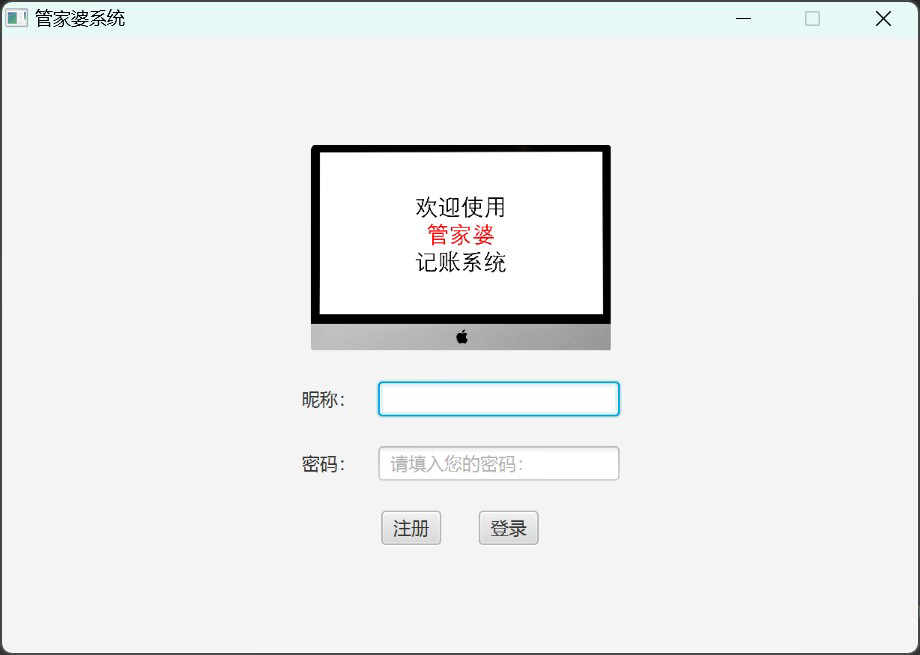
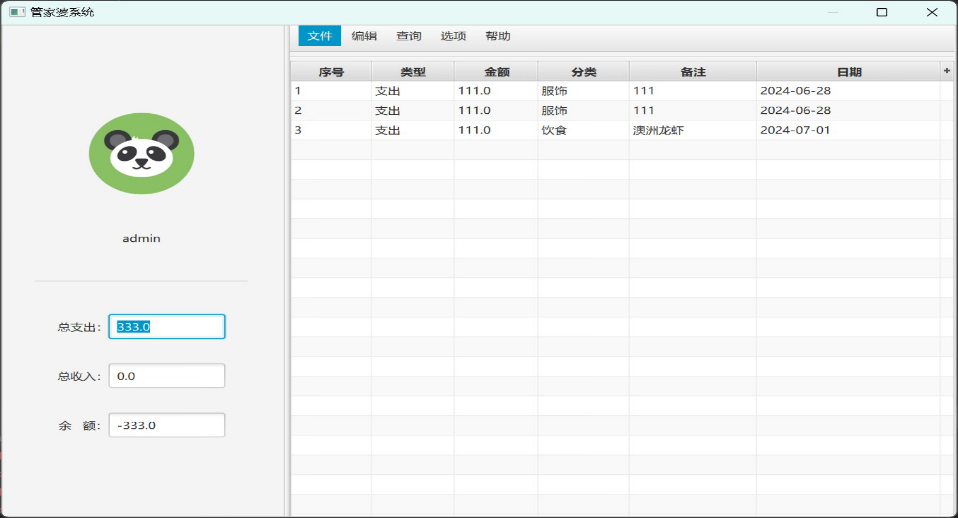
在此系统的开发中，为了达到安全的实际运作，必须通过用户的登录请求，进行注册，在登录时，将用户输入的信息与数据库中的用户信息进行对比，如果相等，则登录成功。登录主界面，登录，注册具体如下：

图5.1 登录主界面

图5.2 注册成功界面

### 图5.3 登录成功界面

### 5.1.2 系统首页界面

用户登录系统后，对记账系统的功能模块进行了设计，包括注册、登录、主界面、导入、导出、备份、恢复、退出、添加、删除、修改、查找、图表、报告等功能。如图5.4所示。

#### 图5.4 登入后系统首页

### 5.1.3文件管理界面

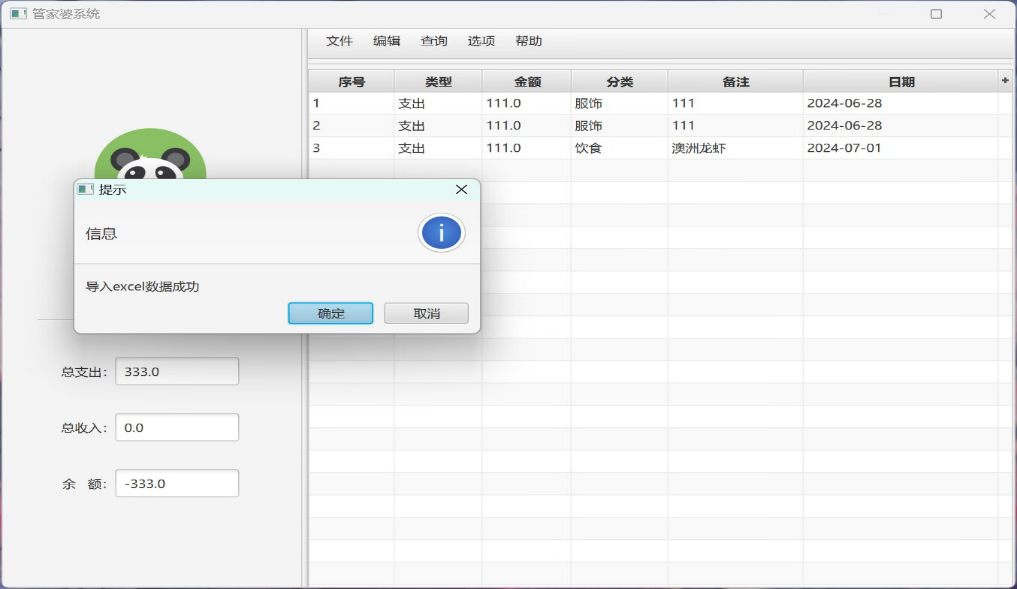
在此界面中展示了文件的导入导出，备份恢复的功能。实现将用户的收入和支出记录导出到excel表中，也可以从excel中导入到记账系统中。可以将该项目所使用的数据库表结构及数据备份保存到本地。将保存到本地的SQL文件再次执行，恢复表结构和数据。文件导入功能如图5.5所示。文件导出功能如图5.6所示。文件备份功能如图5.7所示。文件恢复功能如图5.8所示。

图5.5 文件导入

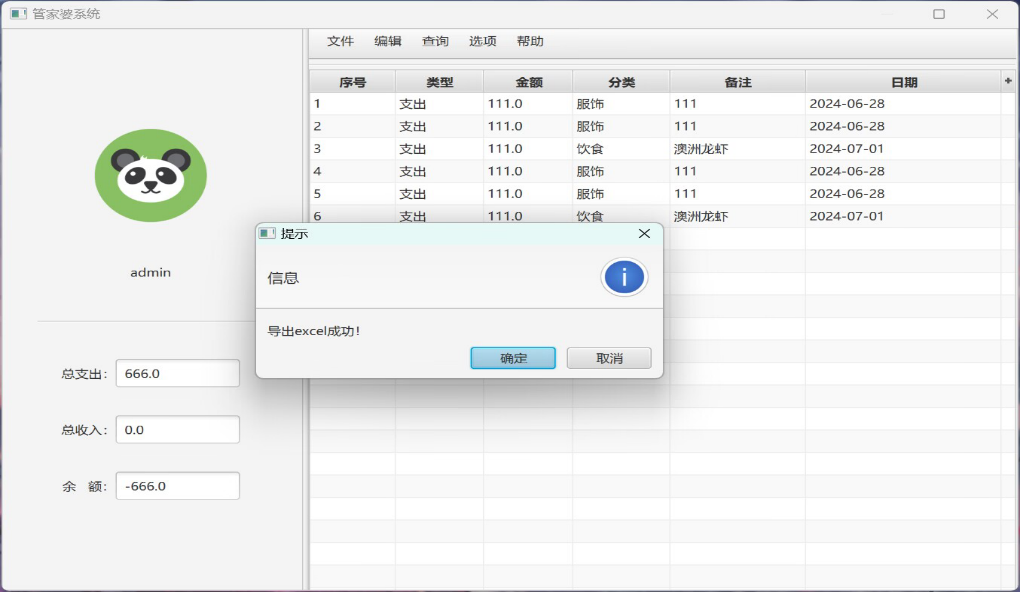


图5.6 文件导出

具体代码如下：

public static String exportExcel(String[] title, String[][] values, String exportPath) {

HSSFWorkbook wb = null;

HSSFSheet sheet = null;

HSSFRow row = null;

HSSFCellStyle style = null;

HSSFCell cell = null;

//第一步，创建一个HSSFWorkbook对象，对应一个excel文件

wb = new HSSFWorkbook();

//第二步，在wb添加一个sheet，对应excel文件中的sheet

sheet = wb.createSheet("Sheet1");

//第三步，在sheet中添加表头第0行,注意老版本poi对Excel的行数列数有限制short

row = sheet.createRow(0);

//第四步，创建单元格，并设置值表头 设置表头居中

style = wb.createCellStyle();

//创建一个居中格式

style.setAlignment(HSSFCellStyle.ALIGN\_CENTER);

//第五步，添加数据到表头

for (int i = 0; i < title.length; i++) {

cell = row.createCell((short) i);

cell.setCellValue(title[i]);

cell.setCellStyle(style);

}

//第六步，写入实体数据，实际应用中这些数据从数据库获得

for (int i = 0; i < values.length; i++) {

row = sheet.createRow(i + 1);

for (int j = 0; j < values[i].length; j++) {

//将内容按顺序赋给对应的列对象

row.createCell((short) j).setCellValue(values[i][j]);

}

}

//第七步，输出excel文件

Date date = new Date();

//定义日期格式

DateFormat format = new SimpleDateFormat("yyyyMMdd");

//拼接文件名

String name = exportPath + format.format(date) + ".xls";

//创建一个文件字节输入流

FileOutputStream outputStream = null;

try {

outputStream = new FileOutputStream(name);

wb.write(outputStream);

outputStream.close();

} catch (Exception e) {

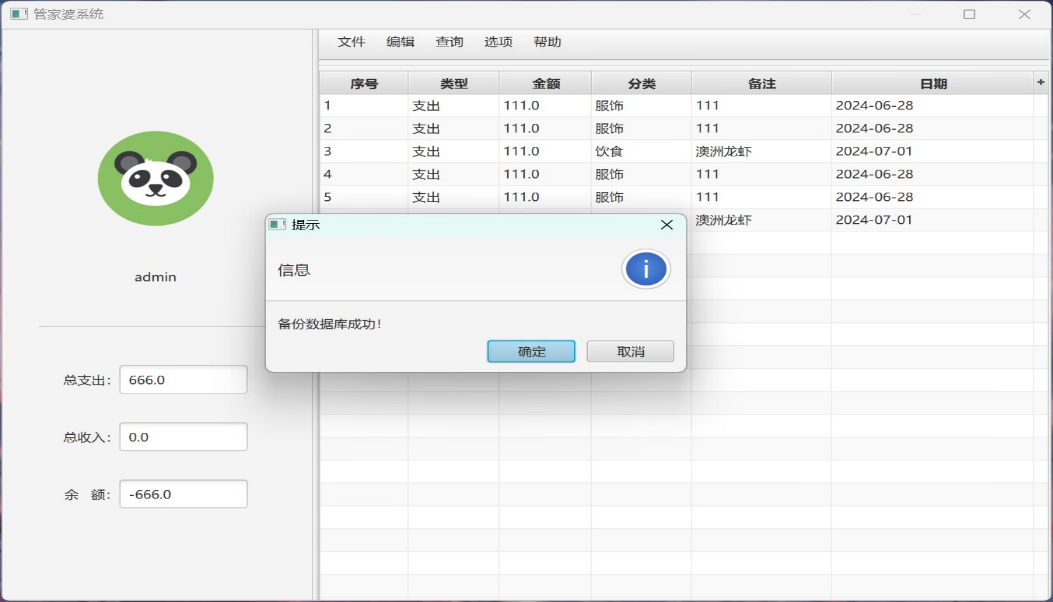
e.printStackTrace();

}

return name;

}

}



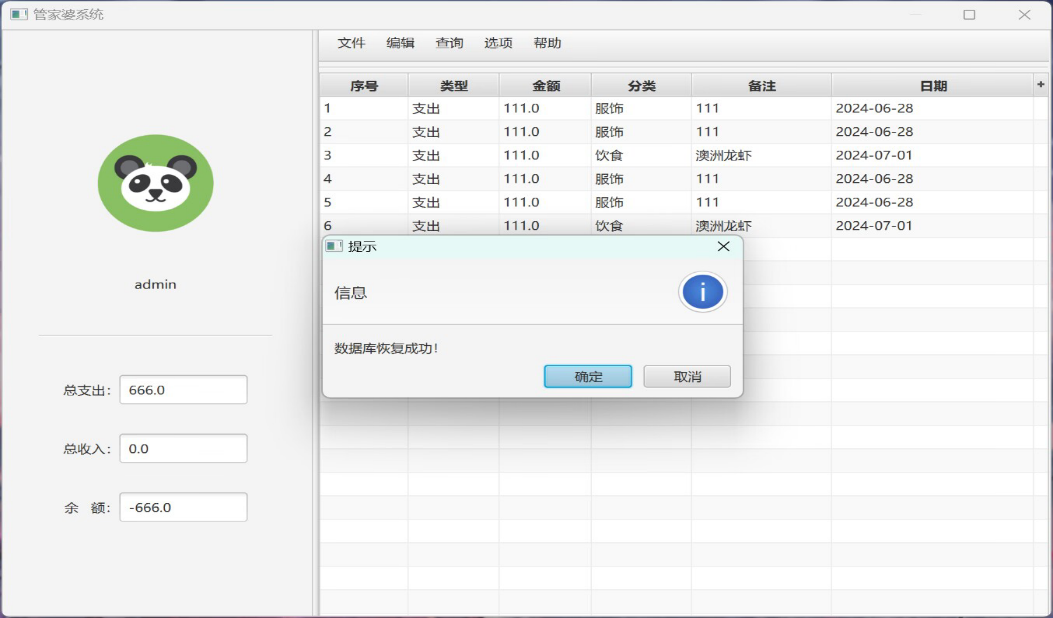
图5.7 文件备份

图5.8 文件恢复

### 5.1.4账单信息管理界面

该界面，实现了对记账系统信息的添加，删除，修改功能。获取用户输入的信息，然后将其插入到数据库记录表中，完成添加功能。删除功能首先获取要删除的记录的ID号，然后从数据库表中查询到该条记录，并使用JDBC的知识实现删除。根据用户输入的要进行修改的记录ID号，从数据库表中查询获取到界面控件中加以显示出来，并由用户进行修改数据，再次重新更改数据到数据库表中。添加信息界面如下图5.9所示，删除信息界面如下图5.10所示，修改信息界面如下图5.11所示。

图5.9 添加信息

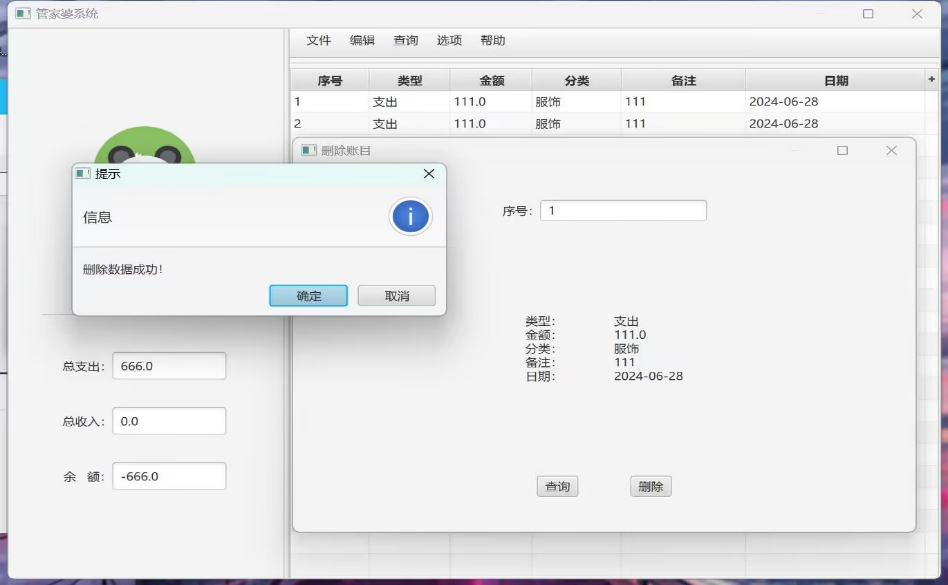


图5.10 删除信息

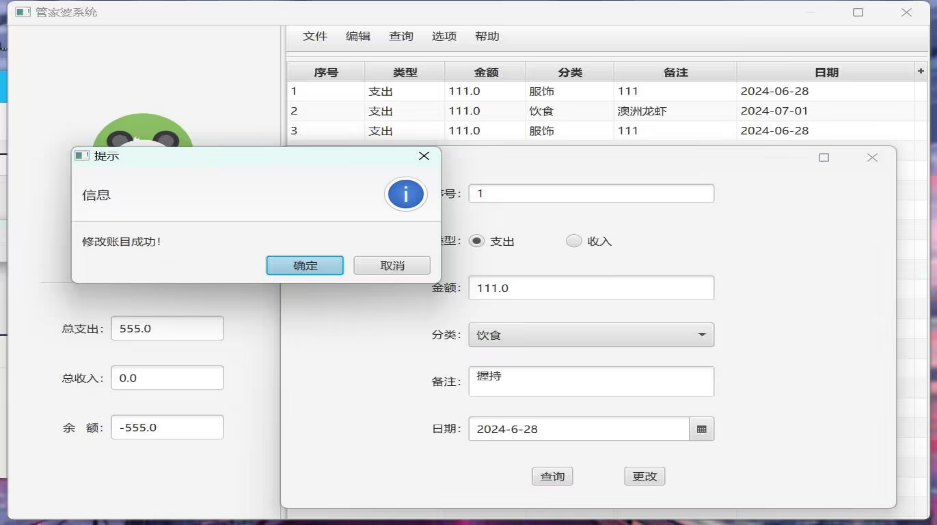


图5.11 修改信息

### 5.1.5查询信息管理界面

该界面可以查询所有账单信息。可以通过按日期查询，按分类条件查询，按备注条件查询。该界面可以通过折线图，条形图，饼图的形式显示图表。跟主界面的表格显示数据是一样的，查询并显示该用户的所有的支出收入记录。获取用户输入的起始日期和结束日期到数据库表中查询符合条件的记录并显示在下面的表格中即可。两个面板分别是收入面板和支出面板，然后可以选择不同的分类进行条件查询，还是获取用户选择然后带着条件在数据库表中获取符合条件的记录。获取用户输入的备注，在数据库表中进行模糊查询即可。获取用户选中的条件，然后根据条件去数据库表查询记录，将查询到的数据填充到条形图，折线图，饼图中显示出来并打出报告。具体界面如下图所示。

图5.12 查询所有信息

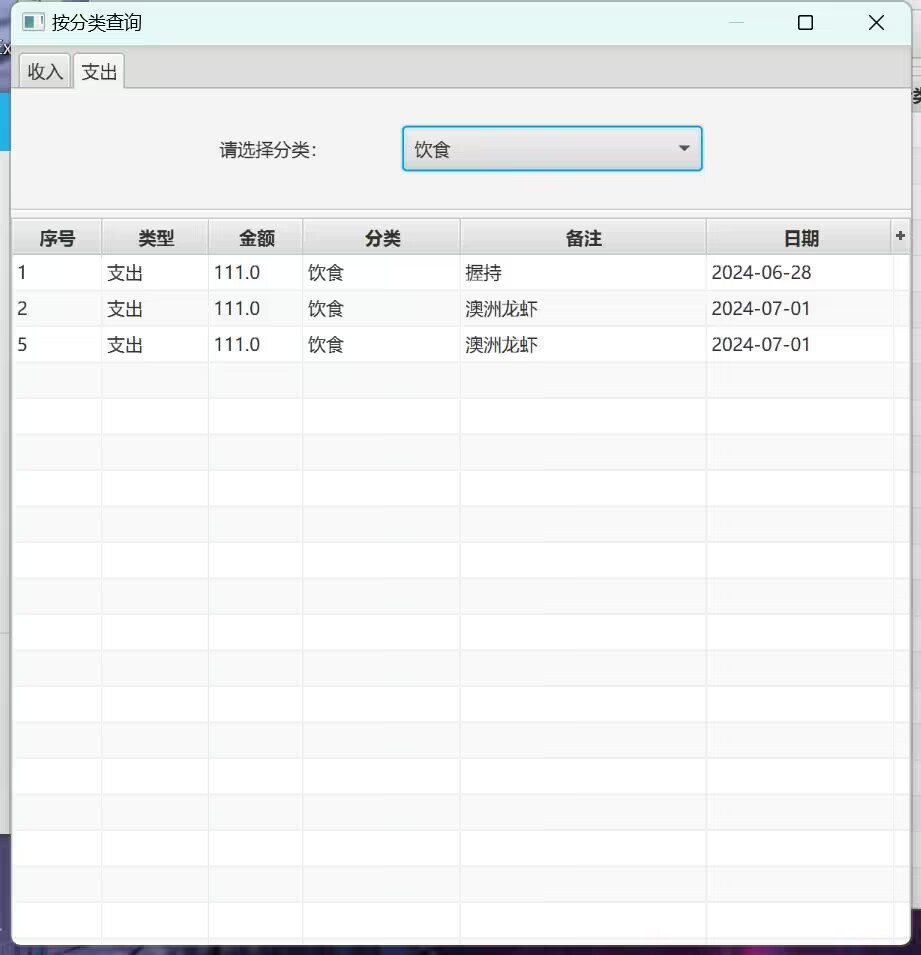
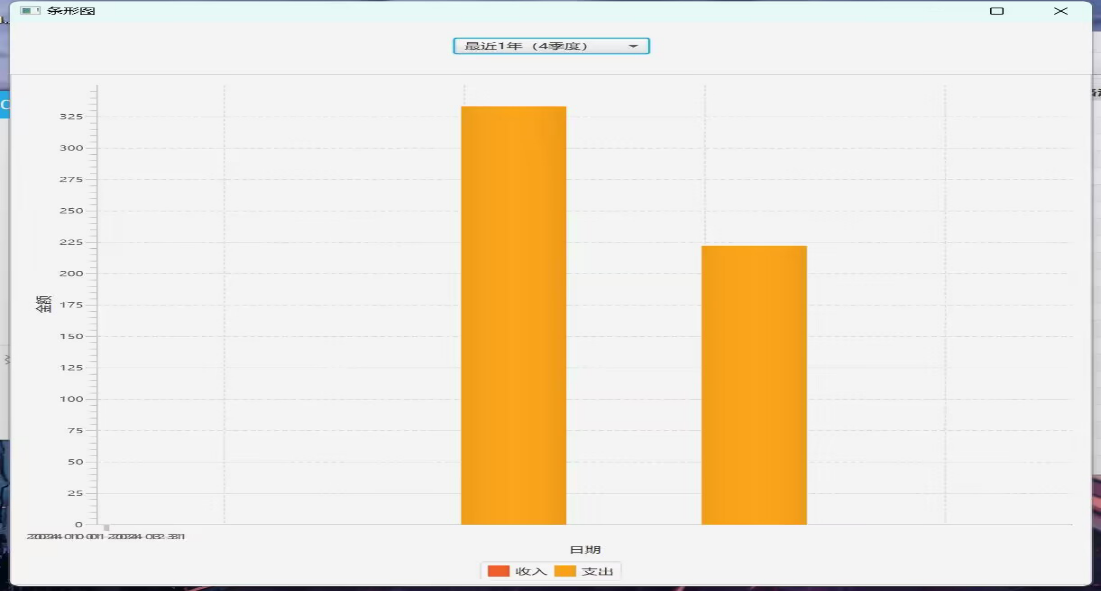
图5.13 按日期查询

图5.14 按分类查询

图5.15 按备注查询

图5.16 条形图

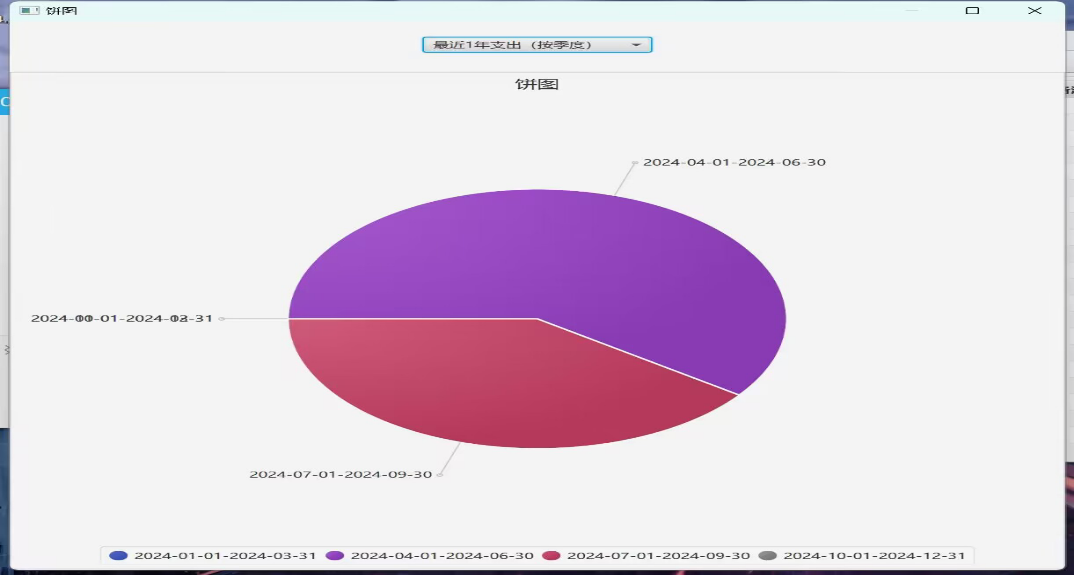
图5.17 折线图

图5.18 饼图

图5.19 报告

具体代码如下：

public void public\_setMonthBarChartData(int month,

BarChart<?, ?> barChart,

CategoryAxis categoryAxis,

NumberAxis numberAxis) {

// 实例化RecordDao对象

RecordDao recordDao = new RecordDao();

// 设置条形图水平轴的标签名称

categoryAxis.setLabel("日期");

// 设置条形图垂直轴的标签名称

numberAxis.setLabel("金额");

// 表示“收入”类型

XYChart.Series monthInputSeries = new XYChart.Series();

// 并设置该类型的名称

monthInputSeries.setName("收入");

// 表示“支出”类型

XYChart.Series monthOutputSeries = new XYChart.Series();

// 并设置该类型的名称

monthOutputSeries.setName("支出");

// 清空集合日期标签中的内容

categoryAxis.getCategories().clear();

// 循环月数，填充数据

for (int i = 0; i < month; i++) {

// 得到一个Calendar日历对象

Calendar calendar = Calendar.getInstance();

// 设置时间

calendar.setTime(new Date());

calendar.add(calendar.MONDAY, -i);

Date monthDate = calendar.getTime();

// 将monthDate进行格式化处理获取“yyyy-MM-dd”格式

String monthStringDate = dateTools.dateFormat(monthDate, "yyyy-MM-dd");

// 拼接SQL语句，获取当前日期的收入总额

String monthInputsql = "select SUM(rMoney) from tb\_records where rType='收入' and MONTH(rDate)= MONTH('" + monthStringDate + "') and uId=" + Session.getUser().getUserId() + ";";

// 拼接SQL语句，获取当前日期的支出总额

String monthOutputsql = "select SUM(rMoney) from tb\_records where rType='支出' and MONTH(rDate)= MONTH('" + monthStringDate + "') and uId=" + Session.getUser().getUserId() + ";";

// 执行收入SQL语句查询得到收入总额

float monthInput = recordDao.getResultValueBySql(monthInputsql);

// 执行支出SQL语句查询得到支出总额

float monthOutput = recordDao.getResultValueBySql(monthOutputsql);

// 为水平轴添加时间标签

categoryAxis.getCategories().add(monthStringDate);

// 填充“收入”类型的数据

monthInputSeries.getData().add(new XYChart.Data<>(dateTools.dateFormat(dateTools.stringToDate(monthStringDate, "yyyy-MM-dd"), "yyyy-MM"), monthInput));

### 5.1.6添加分类和用户信息界面

此界面可以实现用户自定义收入和支出分类，并通过右键菜单实现修改和删除。用户可以查看自己的个人信息并且可以修改密码和修改图像。添加分类界面如图5.20所示，用户信息界面如图5.21所示。



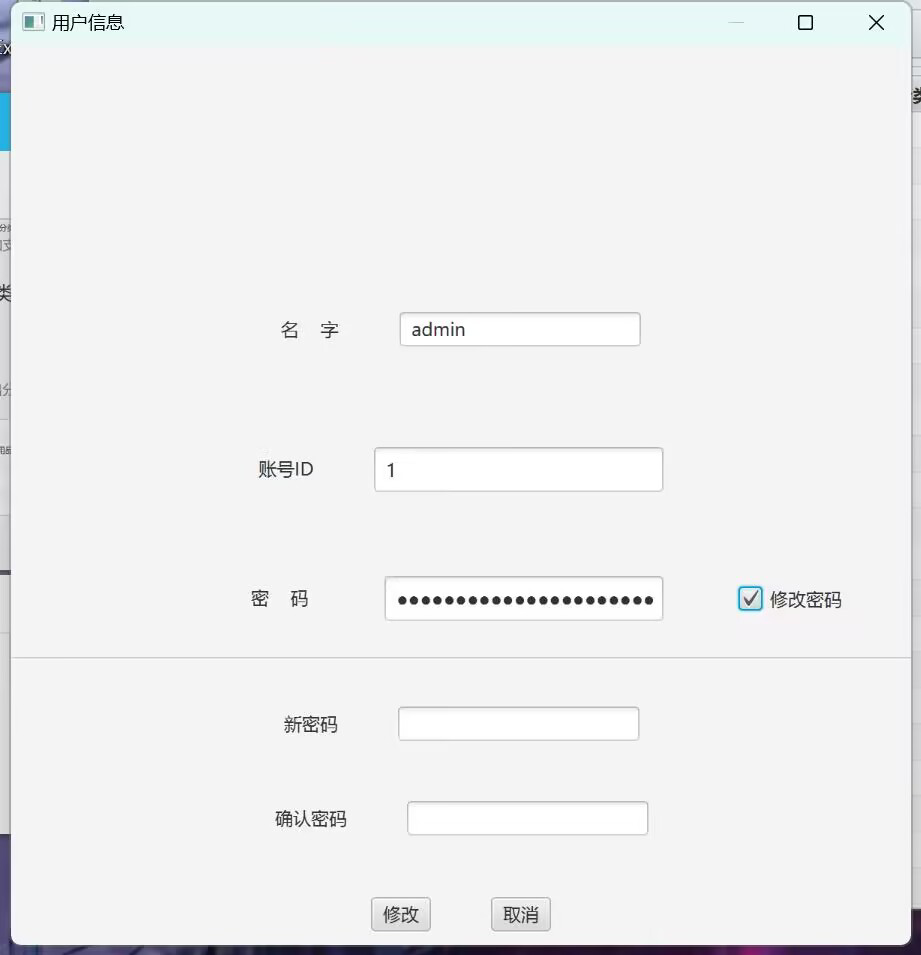
图5.20 添加分类

图5.21 用户信息

## 5.2本章小结

本章中，对记账系统的设计方案进行了详细的介绍，进行了充分的探讨，涉及指特定体系的功能单元和系统软件体系结构的设计方案，以及数据管理系统的概念模型设计、物理模型设计和功能模型设计。

# 6.系统测试

## 6.1环境测试

本项目适合于Windows 10或更高的操作系统，项目可以在服务器上进行部署。因为这个系统是由Maven来负责本项目的，因此开发者只需要执行Maven的命令就可以打包，然后放到Tomcat的服务器上，用cmd来运行这个War包。

表6.1 环境参数

|  |  |
| --- | --- |
| 类型 | 详细参数 |
| 电脑版本 | Windows10 |
| Java 环境 | JDK>=1.8 |
| 数据库版本 | Mysql>=5.5 |

## 6.2功能测试

### 6.2.1用户登录注册模块

#### 表6.2 用户登录注册功能测试表

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例名称 | 具体操作 | 预期结果 | 测试结果 |
| 登录界面测试 | 输入正确的账号密码 | 登录成功，可以使用本系统 | 测试通过 |
| 登录界面测试 | 输入错误的账号密码 | 系统提示登录失败，并进入错误提示 | 测试通过 |
| 注册界面测试 | 输入用户名密码 | 注册成功，可以使用本系统 | 测试通过 |

### 

### 6.2.2文件管理功能模块表

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例名称 | 具体操作 | 预期结果 | 测试结果 |
| 进入主网站首页 | 默认 | 进入到主网站首页。 | 测试通过 |
| 导入账单 | 点击文件下拉列表导入 | 从excel表中导入数据 | 测试通过 |
| 导出账单 | 点击文件下拉列表导出 | 把账单信息导出到excel中 | 测试通过 |
| 备份 | 点击文件下拉列表备份 | 将使用的数据库表结构及数据备份保存到本地 | 测试通过 |
| 恢复 | 点击文件下拉列表恢复 | 将保存到本地的sql文件再次执行，恢复表结构和数据 | 测试通过 |

表6.3 文件管理功能测试表

### 6.2.3账单信息管理模块

##### 表6.4 账单信息管理功能测试表

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例名称 | 具体操作 | 预期结果 | 测试结果 |
| 账单信息管理 | 点击上方菜单栏编辑 | 展开下列菜单 | 测试通过 |
| 添加信息 | 点击添加按钮 | 输入用户信息，插入到数据库记录表中，添加成功 | 测试通过 |
| 删除信息 | 点击删除按钮 | 获取删除记录的ID号，从数据库中查询到该条记录，删除成功 | 测试通过 |
| 修改信息 | 点击修改按钮 | 用户输入要修改的记录ID号，修改数据成功 | 测试通过 |

### 

### 6.2.4查询功能管理模块

#### 表6.5查询功能测试表

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例名称 | 具体操作 | 预期结果 | 测试结果 |
| 查询功能 | 点击上方菜单栏查询 | 展现查询管理下级菜单 | 测试通过 |
| 条件查询 | 点击条件查询按钮 | 出现按日期、分类、备注查询 | 测试通过 |
| 按日期查询 | 点击按日期查询按钮 | 用户输入起始日期和结束日期，查找符合条件的记录 | 测试通过 |
| 按分类查询 | 点击按分类查询按钮 | 用户选择收入和支出面板，查找记录 | 测试通过 |
| 按备注查询 | 点击按备注查询按钮 | 获取用户输入的备注，进行查询 | 测试通过 |
| 图表 | 点击图表按钮 | 获取用户选中的条件，将查到的数据填充到条形图中显示出来 | 测试通过 |
| 报告 | 点击报告按钮 | 获取用户的选择，将查询到的数据填充到饼图中 | 测试通过 |

### 6.2.5添加分类功能管理模块

#### 表6.6添加分类功能测试表

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例名称 | 具体操作 | 预期结果 | 测试结果 |
| 添加分类功能模块 | 点击上方菜单栏选项 | 展现选项下的下级菜单 | 测试通过 |
| 添加分类功能 | 点击添加分类按钮 | 实现用户自定义收入和支出分类，通过右键菜单实现修改和删除 | 测试通过 |
| 用户信息 | 点击用户信息按钮 | 用户可以查看个人信息且可以修改密码和图像 | 测试通过 |

### 6.2.6关于软件和帮助模块

#### 表6.7 关于软件和帮助功能测试表

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例名称 | 具体操作 | 预期结果 | 测试结果 |
| 帮助功能模块 | 点击上方菜单栏帮助 | 展示帮助菜单下的下级菜单 | 测试通过 |
| 关于软件 | 点击关于软件按钮 | 弹出一个面板显示该系统的信息 | 测试通过 |
| 帮助 | 点击帮助按钮 | 用电脑本地的浏览器打开链接 | 测试通过 |

### 

## 6.3本章小结

在对记账管理系统进行综合性测试时，主要的检测目标是确保系统符合设计要求，同时，还需要检测系统在记账工作的业务流程和运行流程方面是否准确。除此之外，还需要对记账工作系统中的所有主要功能模块进行全面的检测，并给出测试结果。

# 

# 总结与展望

本项目所涉及的知识几乎概括了Java基础知识，其中包含JDCB、MySQL、JavaFX、Scene Builder等知识，是一个综合度非常高的项目，能达到练手熟悉知识的目的。而每一章的最后都提供了本章的源代码，而不是一次性提供整个项目的源代码，因为一次性提供整个项目的源代码，不便于学习具体功能，而代码多了便难以分清楚哪些地方调用了哪些方法，实例化了哪些类，为什么要这么做。

在小组分工合作中小组成员对需求理解不一致或不清晰。这需要确保在项目启动前进行充分的需求分析，制定详细的需求文档和用例。定期召开会议，进行需求讨论和澄清，确保每个成员理解一致。使用工具等进行需求跟踪和管理，记录所有讨论和决策。制定清晰的项目计划和任务分配，明确每个成员的责任和工作范围。使用项目管理工具进行任务追踪和进度管理，及时发现和解决分工不均或重叠的问题。确保每个成员了解整体进展并协调工作。

基于Java的记账系统的设计与实现，除了能够解决当前的需求，还有很多潜力和展望可以考虑：功能扩展与模块化，初始版本可能只包含基本的记账和报表功能。未来可以通过添加新的模块来扩展功能，如预算管理、财务分析、税务报表等，以满足不同用户群体的需求。移动端适配，随着移动互联网的发展，开发移动端的应用或响应式网页版本，让用户能够随时随地进行账务管理，提升系统的实用性和用户体验。国际化和多语言支持，如果未来考虑面向全球市场，支持多种语言和国际化功能是必要的，使得记账系统能够适应不同国家和地区的用户需求。

## 参考文献

【1】"Java Programming for Beginners: Learn Java in One Day and Learn It Well" by Jamie Chan - 该书适合初学者学习Java编程，可以帮助开发人员掌握Java语言的基础知识和技能。

【2】"Java Software Solutions: Foundations of Program Design" by John Lewis, William Loftus, and Cara Cocking - 这本书提供了关于Java软件设计和开发的基础知识，适合想要深入学习Java编程的开发人员。

【3】"Effective Java" by Joshua Bloch - 该书介绍了Java编程中的一些最佳实践和技巧，可以帮助开发人员编写高质量、高效的Java代码。

【4】"Java Design Patterns: A Hands-On Experience with Real-World Examples" by Vaskaran Sarcar - 该书介绍了Java设计模式的实际应用，可以帮助开发人员设计和实现可维护、灵活的Java应用程序。

【5】"Java Performance: The Definitive Guide" by Scott Oaks - 这本书介绍了如何优化Java应用程序的性能，包括性能调优技术和工具的使用。

# 附录

附录 1

登录核心代码如下：

  void loginButtonEvent() {  
        // 判断用户是否输入用户名和密码  
        if (nameTextField.getText().equals("") || passwordTextField.getText().equals("")) {  
            SimpleTools.informationDialog(Alert.AlertType.WARNING, "提示", "警告", "请按照文本框内容提示正确填写内容！");  
        } else {  
            // 实例化UserDao对象  
            UserDao userDao = new UserDao();  
            // 封装用户输入的数据到User实体类  
            User user = new User(nameTextField.getText(), SimpleTools.MD5(passwordTextField.getText()), "src\\AccountSystem\\images\\panda.png");  
            // 注册用户，并返回注册结果  
            boolean isLoginSuccess = userDao.register(user);  
            // 对注册结果进行反馈  
            if (isLoginSuccess) {  
                SimpleTools.informationDialog(Alert.AlertType.INFORMATION, "提示", "信息", "恭喜您，注册成功，欢迎使用本系统！");  
            } else {  
                SimpleTools.informationDialog(Alert.AlertType.ERROR, "错误", "错误", "抱歉，您注册失败了，请重新尝试！");  
            }  
        }  
    }  
  
    /\*\*  
     \* “登录”按钮事件监听器  
     \*/  
    @FXML  
    void logupButtonEvent() {  
        // 判断用户是否输入用户名和密码  
        if (nameTextField.getText().equals("") || passwordTextField.getText().equals("")) {  
            SimpleTools.informationDialog(Alert.AlertType.WARNING, "提示", "警告", "请按照文本框内容提示正确填写内容！");  
        } else {  
            // 实例化UserDao对象  
            UserDao userDao = new UserDao();  
            // 登录用户  
            User loginUser = userDao.login(nameTextField.getText(), SimpleTools.MD5(passwordTextField.getText()));  
            // 对是否登录成功进行判断  
            if (loginUser.getUserName() != null && loginUser.getUserPassword() != null) {  
                // 设置通信对象，建立登录成功通信  
                Session.setUser(loginUser);  
                // 在弹出的提示框种获取用户反馈  
                boolean b = SimpleTools.informationDialog(Alert.AlertType.INFORMATION, "提示", "信息", "恭喜" + Session.getUser().getUserName() + "，登录成功，欢迎使用本系统！");  
                // 如果用户确定登录，则跳转到主界面  
                if (b) {  
                    // 关闭登录界面  
                    logupStage.close();  
                    // 打开主窗口  
                    new MainApp().initMainFrame();  
                }  
            } else {  
                SimpleTools.informationDialog(Alert.AlertType.ERROR, "错误", "错误", "用户名或密码错误！");  
            }  
        }  
    }  
}

附录2

账单信息管理核心代码如下：

 public void deleteContextMenuEvent(ActionEvent event) {  
        // 给出提示框提示用户是否选择删除  
        boolean b = SimpleTools.informationDialog(Alert.AlertType.CONFIRMATION, "提示", "提示", "请问是否删除？");  
        if (b) {  
            RecordDao recordDao = new RecordDao();  
            // 获取所选择行的索引，从0开始  
            int selectedIndex = tableView.getSelectionModel().getSelectedIndex();  
            // 获取所选中行是数据  
            TableData td = tableView.getSelectionModel().getSelectedItem();  
            // 判断是否选中表格行  
            if (selectedIndex >= 0) {  
                // 删除选中的表格行根据索引  
                tableView.getItems().remove(selectedIndex);

                // 执行删除操作  
                recordDao.deleteRecord(new Record(Integer.parseInt(td.getId())));  
            }  
            // 刷新表格数据  
            initialize();  
        }  
    }

  /\*\*  
     \* “添加”右键菜单的事件监听器  
     \*

     \* @param event 事件  
     \*/  
    @FXML  
    public void addContextMenuEvent(ActionEvent event) {  
        // 跳转到添加界面即可  
        mainApp.initAddFrame();  
    }

  /\*\*  
     \* “修改”右键菜单的事件监听器  
     \*  
     \* @param event 事件  
     \*/  
    @FXML  
    public void alterContextMenuEvent(ActionEvent event) {  
        // 获取所选中行的数据  
        TableData td = tableView.getSelectionModel().getSelectedItem();  
        // 设置一个标志，判断是右键菜单触发的修改还是由菜单条上的菜单触发的修改  
        boolean isContextMenu = true;  
        mainApp.initAlterFrame(td, isContextMenu);  
    }

  /\*\*  
     \* ”刷新“右键菜单的事件监听器  
     \*  
     \* @param actionEvent 事件  
     \*/  
    public void refreshContextMenuEvent(ActionEvent actionEvent) {  
        // 刷新数据  
        initialize();  
    }

  /\*\*  
     \* ”查询“菜单项的事件监听器  
     \*  
     \* @param actionEvent 事件  
     \*/  
    @FXML

    public void checkMenuItemEvent(ActionEvent actionEvent) {  
        // 打开查询界面  
        mainApp.initTableView();

        initialize();  
    }

  /\*\*  
     \* ”按日期查询“菜单项的事件监听器  
     \*  
     \* @param event 事件  
     \*/  
    @FXML  
    public void dateCheckMenuItemEvent(ActionEvent event) {  
        // 打开按日期查询界面  
        mainApp.initDateCheckTableView();  
    }

 /\*\*

     \* ”按分类查询“菜单项的事件监听器  
     \*

\* @param event 事件  
     \*/  
    @FXML  
    public void classificationCheckMenuItemEvent(ActionEvent event) {  
        // 打开按分类查询界面  
        mainApp.initClassificationTableView();  
    }

/\*\*  
     \* ”按备注查询“菜单项的事件监听器  
     \*  
     \* @param event 事件  
     \*/  
    @FXML  
    public void memoCheckMenuItemEvent(ActionEvent event) {  
        // 打开按备注查询界面  
        mainApp.initMemoTableView();  
    }

   /\*\*  
     \* ”条形图“菜单项的事件监听器  
     \*  
     \* @param actionEvent 事件

     \*/  
    @FXML

    public void barChartMenuItemEvent(ActionEvent actionEvent) {  
        // 打开条形图界面  
        mainApp.initBarChart();  
    }

 /\*\*  
     \* “折线图”菜单项的事件监听器  
     \*  
     \* @param actionEvent 事件

     \*/  
    @FXML  
    public void lineChartMenuItemEvent(ActionEvent actionEvent) {  
        // 打开折线图界面  
        mainApp.initLineChart();  
    }

  /\*\*  
     \* ”饼图“菜单项的事件监听器  
     \*  
     \* @param actionEvent 事件  
     \*/  
    @FXML  
    public void pieChartMenuItemEvent(ActionEvent actionEvent) {  
        // 打开饼图界面  
        mainApp.initPieChart();  
    }

 /\*\*  
     \* “报告”菜单项的事件监听器  
     \*  
     \* @param event 事件  
     \*/  
    @FXML

    public void reportMenuItemEvent(ActionEvent event) {  
        // 打开报告界面

        mainApp.initReportFrame();  
    }

 /\*\*

  \* ”添加分类“菜单项的事件监听器  
     \*  
     \* @param actionEvent 事件  
     \*/  
    @FXML  
    public void addClassificationMenuItemEvent(ActionEvent actionEvent) {  
        // 打开添加分类界面  
        mainApp.initAddClassificationFrame();  
    }