

哈尔滨师范大学

课程论文

学年学期 2023-2024 学年第一学期

课程名称 数据库应用与开发

题 目 Tkinter 图书馆信息管理系统

姓 名 _____

学 号 _____

学 院 计算机科学与信息工程学院

任课教师 _____

论文评分标准:

优秀 (90-100): 能熟练综合运用所学知识, 设计内容完整、规范, 有大量创新, 方案合理可行, 文字条理清晰, 图表丰富, 版式风格统一。

良好 (80-90): 能较好运用所学知识, 设计内容完整、较规范, 有一定创新, 内容较丰富、文字条理清晰, 图表较丰富, 版式风格统一。

中等 (70-80): 能运用所学知识, 设计内容基本完整, 有规范但创新性差, 内容不够丰富, 文字无较大问题, 图表不够丰富。

及格 (60-70): 基本完成设计任务, 内容欠完整, 设计工作量不大; 文字版式质量一般, 图表较少且存在个别非原则性错误。

不及格 (0-60): 勉强完成设计任务, 设计内容不完整, 工作量偏低, 态度不认真; 存在多处较严重的设计或文字错误。

总评成绩	
------	--

基于 Python Tkinter 和 MySQL 的图书馆信息管理系统设计与实现

摘要：本论详细介绍了一个基于 Python 语言、Tkinter 图形化界面和 MySQL 数据库的图书馆信息管理系统的设计与实现过程。系统旨在提高图书馆管理效率，简化借阅流程，并提供便捷的信息检索服务。通过系统架构、设计思路、详细设计与实现步骤、功能展示、实现细节与技术挑战等方面的阐述，本文旨在向读者展示该图书馆信息管理系统的构建过程与特性。

关键词：Tkinter 图形化界面；MySQL 数据库；图书馆信息管理系统

1 引言

在 21 世纪的知识经济时代，信息资源已成为社会发展的重要推动力，而图书馆作为信息资源的集散地，其管理方式的现代化、信息化水平直接影响着信息资源的利用效率。传统的图书馆管理模式大多依赖于手工操作，这不仅效率低下，而且容易出错，难以满足现代社会对信息快速流通的需求。因此，构建一个基于计算机技术的图书馆信息管理系统（Library Information Management System, LIMS）显得尤为重要。

图书馆信息管理系统旨在通过自动化的手段优化图书馆的日常运作，包括图书信息的添加、删除、修改和读者服务等各个环节。这样的系统不仅可以提高图书管理的工作效率，减少人为错误，还可以为读者提供更加便捷的服务。

信息技术的不断进步，特别是数据库技术和图形用户界面的发展，为图书馆信息管理系统的实现提供了强大的技术支持。Python 作为一种高级编程语言，以其清晰的语法和强大的功能，成为开发此类系统的理想选择。Tkinter 作为 Python 的标准 GUI 库，提供了丰富的组件和灵活的布局方式，使得界面设计变得简单直观。而 MySQL 数据库则以其高性能、高可靠性和易用性，成为存储和管理图书信息的理想选择。

本系统的设计目标是实现一个功能全面、操作简便、界面友好的图书馆信息管理系统。系统将提供用户登录、书籍检索、借阅管理、归还处理以及管理员对书籍信息的增删改查等功能。通过实现这些功能，系统将极大地提升图书馆的管理效率和服务质量，满足现代图书馆信息化管理的需求。

综上所述，本系统将采用 Python 语言进行开发，利用 Tkinter 构建图形用户界面，使用 MySQL 数据库存储和管理数据。通过这种技术组合，我们旨在构建一个高效、可靠、易用的图书馆信息管理系统，以为图书馆管理和读者服务提供强有力的技术支持。

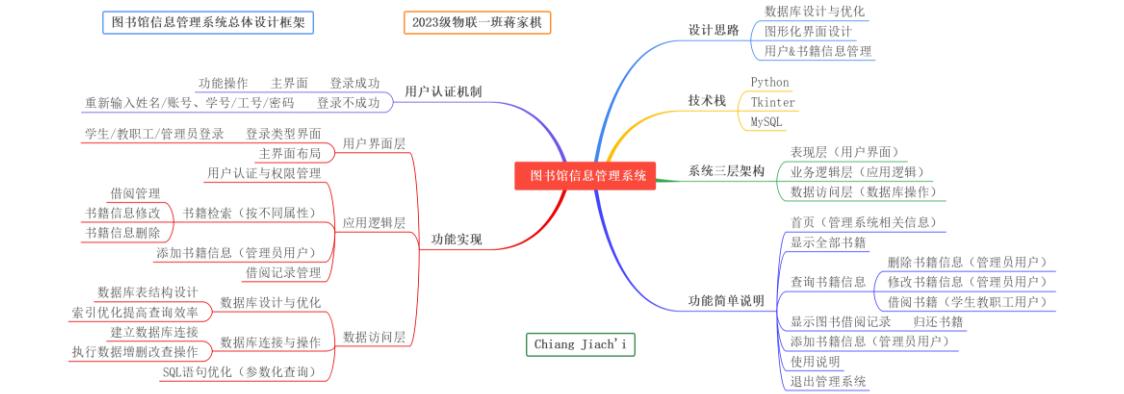


图 1 图书馆信息管理系统总体设计框架



2 设计思路

2.1 数据库设计与优化

设计两个关系型数据库：一个用于存储学生、教师和管理员人员信息，包括学生和教师的姓名和学号(工号)以及管理员用户的账号和密码；另一个用于存储书籍信息和借阅记录，其中书籍信息包含关键字段，如书名、作者、ISBN、存放地址等，借阅记录包括姓名、学号(工号)、书籍名称、图书编号、借阅时间和借阅时长等。在数据库设计过程中，本系统将严格遵循数据库范式化的原则，这包括但不限于第一范式(1NF)、第二范式(2NF)和第三范式(3NF)。这样的设计将确保数据的一致性、减少数据冗余，并提高数据的完整性。

2.2. 用户认证与权限管理

本系统将利用 Python 的 Tkinter 库构建用户登录界面，并通过 Entry 组件获取用户的登录信息，通过 Pymysql 库与 MySQL 数据库建立连接交互，实现用户认证机制。在后端，系统将实施严格的权限管理策略，确保只有经过授权的用户才能访问特定的系统功能。这包括但不限于学生和教师只能查看和借阅书籍，而管理员则拥有除了借阅书籍之外几乎全面的系统管理权限，包括用户管理、书籍信息维护和借阅记录的审核。

2.3. 图形化界面设计

在图书馆信息管理系统的设计中，图形化界面是提升用户体验的关键因素。本系统采用 Python 的 Tkinter 库来构建 GUI 界面，因其简单易用且功能全面，适合快速开发和部署。图形化界面设计的核心目标是实现直观、易用、高效和美观的交互环境。

2.4 书籍信息管理

书籍信息管理是图书馆信息管理系统中的核心组成部分，它涉及到图书馆藏书の詳細记录、检索、借阅状态跟踪以及维护工作。然而系统与数据库的交互是保证系统功能的重要因素，所以本系统将通过 Python 的第三方库 Pymysql 连接 MySQL 数据库并执行系统后端代码中的 SQL 语句与数据库进行交互，该操作将实现书籍信息和借阅记录信息的添加、删除、修改、查询、显示等功能。

3 系统架构

为了构建一个高效、可维护且具有扩展性的图书馆信息管理系统，本系统选择采用三层架构(用户界面层、应用逻辑层和数据访问层)，以更好地组织系统的各个组件和实现系统的各功能。

3.1 用户界面层

用户界面层是系统与用户直接交互的部分，本系统采用 Python 的 Tkinter 库实现，其作为 Python 语言的标准库，具有系统兼容性好、调用便利等特点，同时该库包含丰富而简洁的 GUI 组件，对于开发者和用户都十分友好，便于实现信息管理系统这样操作广泛操作项目的清晰的项目。本系统各个界面模块包括登录、书籍信息管理等功能界面，将负责接收用户输入、展示信息并向应用逻辑层传递用户请求等后端操作。

3.2 应用逻辑层

应用逻辑层用于获取用户的请求并将用户请求转为代码逻辑访问数据库，起到承上启下

的关键作用。通过 MVC 模式，将业务逻辑分离出来，以便更好地维护和扩展。这一层包括用户认证、权限管理、学生信息的增删改查等功能的具体实现。它接收用户界面层传递的请求，调用数据访问层进行数据库操作，并将处理结果返回给用户界面。

3.3 数据访问层

数据访问层用于系统与数据库进行交互。本系统通过调用 Pymysql 库中的函数与方法执行 SQL 语句，从而获取或更新数据库中的信息。通过使用 Pymysql 库，系统能够轻松地与 MySQL 数据库建立连接。这一层的设计旨在最大程度地隔离数据库细节，使得更换数据库或进行优化不会对系统的其他部分产生影响。

采用三层架构有助于保持系统的清晰结构，提高代码的可读性和可维护性。同时，MVC 设计模式的引入使得各层之间的职责明确，减少耦合，便于团队协作和系统的后续扩展。

4 系统设计

4.1 用户登录

4.1.1 登录界面设计

用户登录是系统的入口，因此设计一个直观安全且有图书馆所属单位特色的登录界面是十分有意义的。系统采用 Tkinter 创建登录界面，首先包括用户登录类型（学生登录、教师登录和管理员登录）的选择界面窗口，在选择登录类型后可打开对应的正式登录窗口。学生登录窗口包括用 Entry 组件实现的姓名输入栏和学号输入栏，教师登录窗口则包含姓名和工号输入栏，管理员登录窗口则是账号和密码输入栏，每个登录窗口下方都有登录和取消按钮。



图 2 登陆类型选择界面设计



图 3 学生登录窗口设计

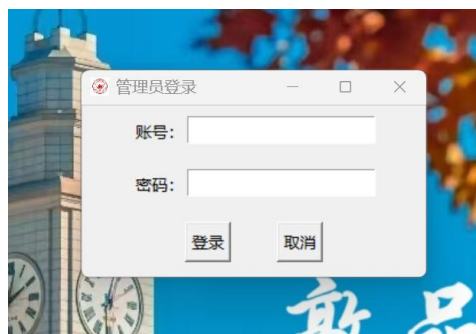


图 4 管理员登陆窗口设计

4.1.2 用户认证机制

在用户登录时，设计用户认证机制以验证用户身份。通过在数据库中建立学生数据表、教师数据表和管理员数据表来分别存储学生、教师和管理员的登录信息，并通过 Pymysql 访问数据库实现用户登录。

```
921     """登录按钮逻辑"""
922     # 全局变量声明
923     login_student_window = None
924     login_teacher_window = None
925     login_admin_window = None
926     def auto_login(window):
927         # 连接数据库
928         db = pymysql.connect(host='127.0.0.1', user='root', password='1205', db='school', port=3306)
929         # 获取操作游标
930         cur = db.cursor()
931         # 分类查询数据库
932         if window == login_student_window:
933             sql = 'select * from student'
934         elif window == login_teacher_window:
935             sql = 'select * from teacher'
936         else:
937             sql = 'select * from admin'
938
939         entry1 = name.get()
940         entry2 = no.get()
941         flag = False
942         try:
943             cur.execute(sql) # 执行查询
944             results = cur.fetchall() # 获取所有查询数据
945             for row in results:
946                 uid = row[0] # 姓名或账号
947                 pwd = row[1] # 学号或工号或密码
948                 # 判断输入的账号和密码是否正确
949                 if entry1 == uid and entry2 == pwd:
950                     username = row[0]
951                     usereno = row[1]
952                     success_tip(username, usereno, window)
953                     flag = True
954                     break
955             if not flag:
956                 fail_tip(window)
957         except Exception as e:
958             fail_tip(window)
959         finally:
960             cur.close()
961             db.close()
```

图 5 用户认证机制

4.2 图书馆信息管理界面

4.2.1 图形化界面设计

使用 Tkinter 来实现本图书馆信息管理系统的图形化界面设计，实现总体框架借用的组件包括 PaneWinodw 和 Notebook。该图形化界面的主界面通过 PaneWindow 组件将主页面分为左右两部分（如图表 6 所示），左边部分包括“欢迎信息”、“登陆时间”和“退出按钮”，右边部分通过 Notebook 组件实现多个功能页面，如在普通用户（学生和教师）登录后的页面中有“首页”、“全部书籍”、“馆藏查询/书籍借阅”、“借阅记录/归还书籍”和“使用说明”等功能，在管理员用户登录后进入的页面中还包括“添加书籍”的功能页面。

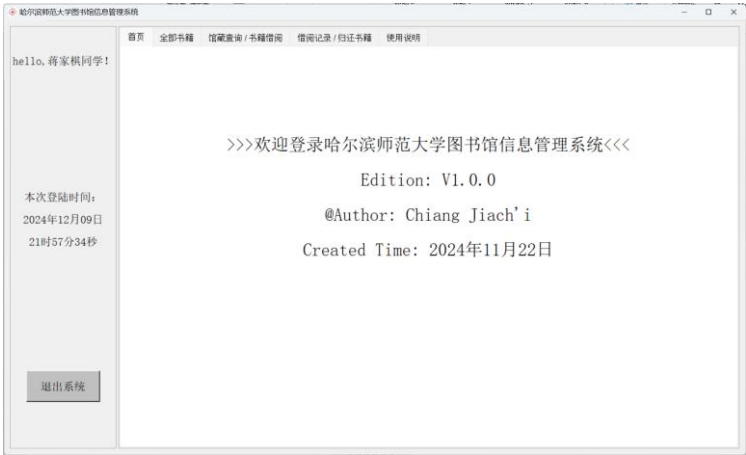
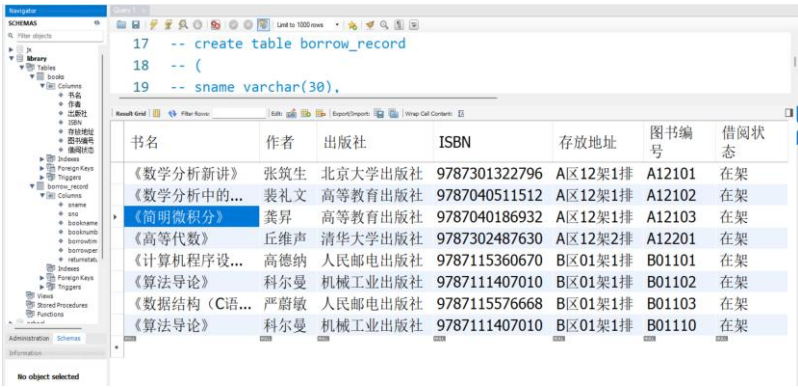


图 6 普通用户的图形化界面

4.2.2 书籍信息数据表

通过 MySQL 关系型数据库实现书籍信息管理数据表，便于通过 Python 语言对书籍信息进行操作。表格应包括书名、作者、出版社、ISBN、存放地址、图书编号和借阅状态，其中因为图书编号是唯一能表示每条书籍信息的属性，所以将图书编号设为主码，方便实现数据库的完整性控制、关系索引和数据排序等功能。



书名	作者	出版社	ISBN	存放地址	图书编号	借阅状态
《数学分析新讲》	张筑生	北京大学出版社	9787301322796	A区12架1排	A12101	在架
《数学分析中的...	裴礼文	高等教育出版社	9787040511512	A区12架1排	A12102	在架
《简明微积分》	龚昇	高等教育出版社	9787040186932	A区12架1排	A12103	在架
《高等代数》	丘维声	清华大学出版社	9787302487630	A区12架2排	A12201	在架
《计算机程序设...	高德纳	人民邮电出版社	9787115360670	B区01架1排	B01101	在架
《算法导论》	科尔曼	机械工业出版社	9787111407010	B区01架1排	B01102	在架
《数据结构（C语...	严蔚敏	人民邮电出版社	9787115576668	B区01架1排	B01103	在架
《算法导论》	科尔曼	机械工业出版社	9787111407010	B区01架1排	B01110	在架

图 7 书籍信息数据表

4.2.3 各功能页面

本系统利用 Notebook 组件实现系统各功能页面。用户可通过点击这些 Notebook 页面的导航按钮，转跳到不同页面，以实现书记查询、书籍借阅、书籍归还等功能。查询功能支持多条件组合查询，以满足不同查询需求。

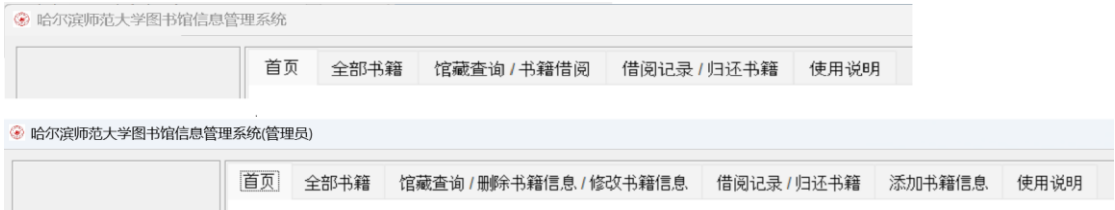


图 8 各功能页面的导航按钮

通过以上设计,用户可以通过直观的界面完成登录、查询和书籍借阅、书籍归还等操作,提高了系统的易用性和用户体验。

5 功能展示

5.1 登录功能

用户打开系统,首先进入登录类型选择页面,然后打开面向不同用户的登录窗口。普通用户需要输入姓名和学号/工号,并点击登录按钮;管理员用户需输入账号和密码,并点击登录按钮。系统将进行用户认证,验证用户身份。若认证成功,系统跳转至图书馆信息管理界面;否则,给予相应的错误提示。

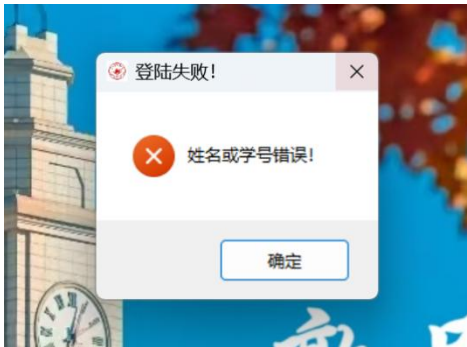


图 9 学生用户登陆失败提示弹窗

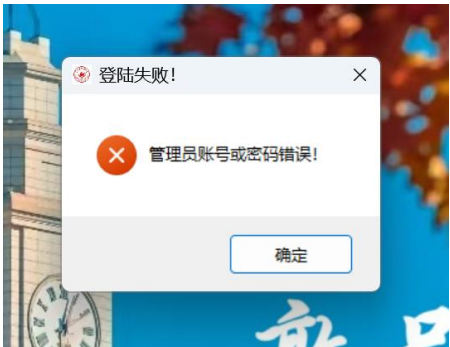


图 10 管理员用户登陆失败提示弹窗

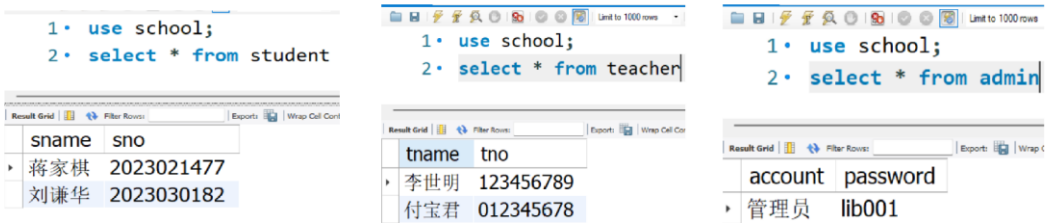


图 11 数据库存储登录信息

5.2 图书馆信息管理功能

5.2.1 首页

在登录成功后,图书馆信息管理系统进入主界面,首先展示的是通过 Notebook 页面设

置的“首页”。该页面主要是显示欢迎信息以及系统的版本和开发者信息，意在为开发者和用户提供良好的情绪价值。



图 12 首页设计

5.2.2 浏览全部书籍

用户点击“全部书籍”导航按钮后，系统的 Notebook 组件设置的功能页面将转到“全部书籍”页面，该页面直接连接访问 MySQL 数据库获取书籍信息，并通过 Tkinter 的 scrolledtext 组件将获取的书籍文本信息在页面显示，实现用户对全部书籍的浏览。同时由于在数据库中已经设置了图书编号为主码，所以显示的信息也会依照主码字段排列，提高了书籍信息的可读性和便于查找等特性。



图 13 浏览全部书籍页面设计

5.2.3 馆藏查询

用户点击“馆藏查询/书籍借阅”导航按钮，功能页面将转跳到“书籍查询”页面，该页面由一行提示语、五个字段输入栏和一个确定按钮组成。用户可根据字段提示输出需要查询的书籍信息字段，点击“确定查询”后该信息被后端代码获取，系统连接并打开 MySQL 数据库，进而从数据库中获取满足约束条件的信息，并在通过 Listbox 组件设置的页面上显示从数据库中获取的书籍信息。用户可输入单一字段也可输入多个字段，如果用户没有输入任何字段信息，系统将从数据库中提取出全部书籍的信息。



图 14 馆藏查询页面设计

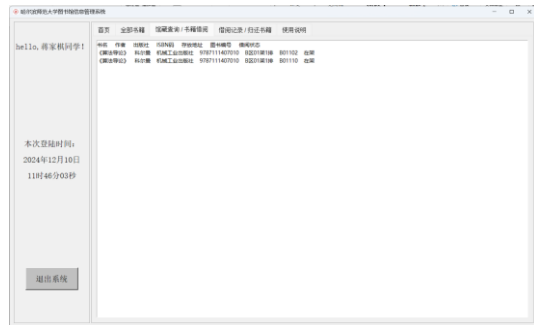


图 15 查询后的书籍信息显示

5.2.4 书籍借阅

书籍借阅是图书馆信息管理系统最核心最常用的功能，本系统将其设计到书记查询页面之中。用户查询出想要的书籍信息之后，可以通过鼠标选中对应的书籍信息，随后鼠标左键点击该条信息，系统就会弹出“书籍借阅”的提示框，用户可以在提示框中选择“确定”或“取消”选项。用户点击“确定”后，若书籍在架，就会弹出“借阅时长”选择提示框，在选择好借阅时长后点击“确定”就会弹出借阅成功并提示按时归还的弹窗，此时系统会同时打开数据库，修改数据库中对应的书籍信息的“借阅状态”为“在借”，同时在数据库中添加借阅记录；若书籍在借，那么就会弹出“借阅失败，本书籍在借中”的提示框。

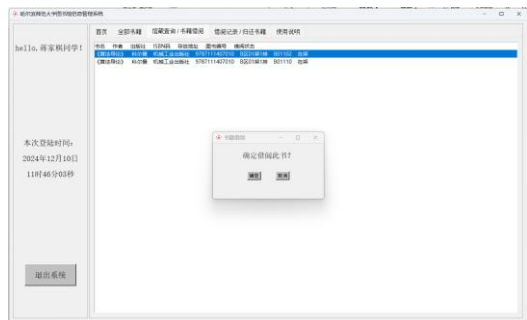


图 16 书籍借阅提示框

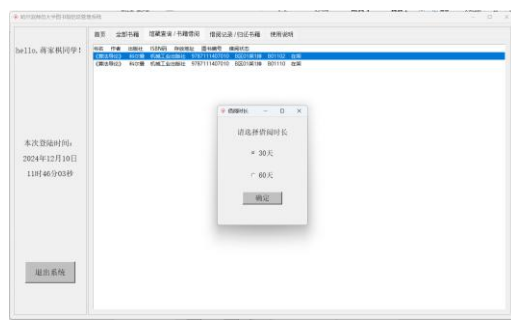


图 17 借阅时长提示框

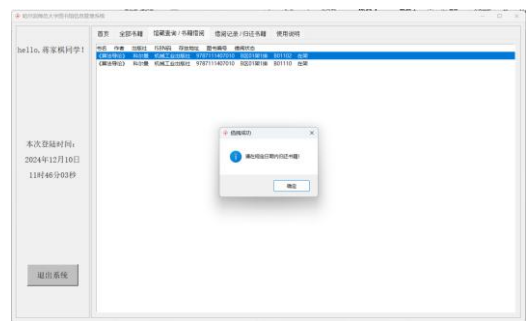


图 18 借阅成功弹窗页面

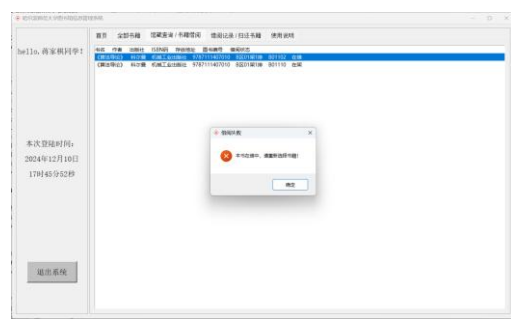


图 19 借阅失败弹窗页面

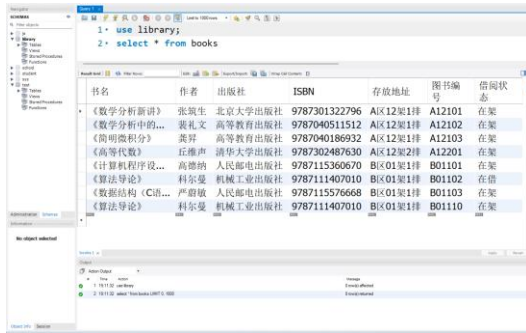


图 20 自动修改数据库书籍借阅状态

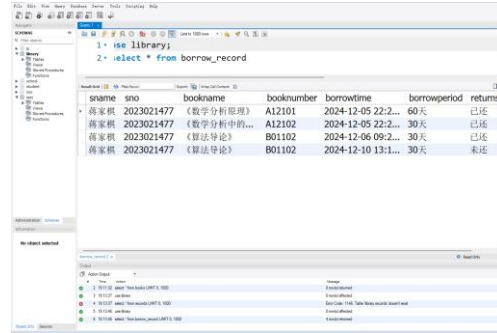


图 21 添加书籍借阅记录

5.2.5 删除书籍信息（管理员用户功能）

作为一款图书馆信息管理系统，除了可以实现查询书记、借阅书籍等普通用户功能外，还应具备面向管理员用户的对书籍信息的管理功能，如“删除书籍信息”、“修改书籍信息”、“添加书籍信息”等。本系统将“删除书籍信息”的功能像上述的“书籍借阅”一样，嵌入到“书籍查询”的 Notebook 功能页面之内。当用户查询出预想的书籍信息之后，点击书籍信息，将弹出“删除修改书籍信息”的选择页面，用户继续点击“删除书籍信息”将弹出一个“是否删除书籍信息”的提示弹窗，点击确定后将弹出一个“删除成功，退出并重新进入系统后将更新书籍信息的”提示弹窗，并且系统将同时连接数据库并修改数据库中的书籍信息。由于 Tkinter 界面的实时交互性不好，所以要更新各类已经显示的信息只能通过退出系统重新进入的方式。

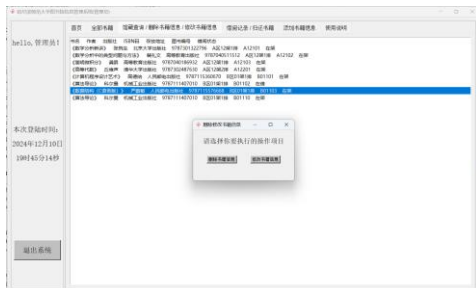


图 22 删除修改操作选择页面

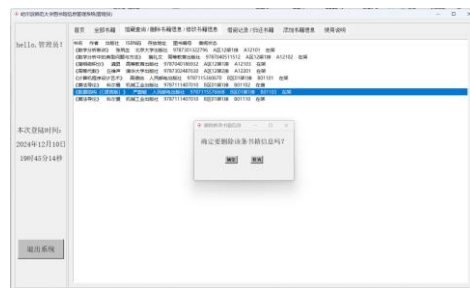


图 23 确定删除书籍页面

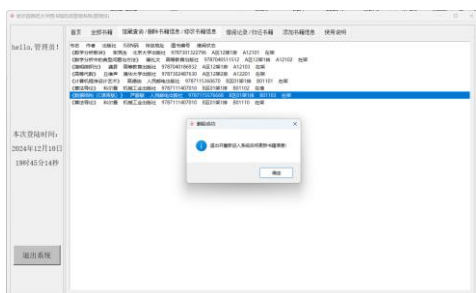


图 24 书籍删除成功页面

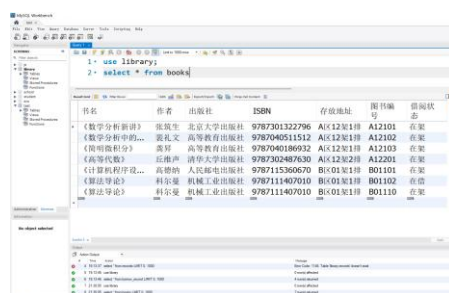


图 25 书籍删除成功后的数据库

5.2.6 修改书籍信息（管理员用户功能）

在上一个功能介绍中的“删除修改书籍信息”选择页面中，还有一个“修改书籍信息”的操作选项，在点击“修改书籍信息”按钮后，将弹出一个“修改书籍信息”的弹窗页面，

该页面由一条提示信息和多个输入栏以及两个按钮组成。用户在相应的输入栏输入需要修改的书籍字段信息，点击“确定修改”按钮后将在数据库中修改书籍信息。



图 26 修改书籍信息页面

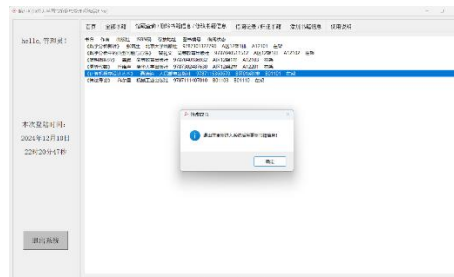


图 27 修改书籍信息成功

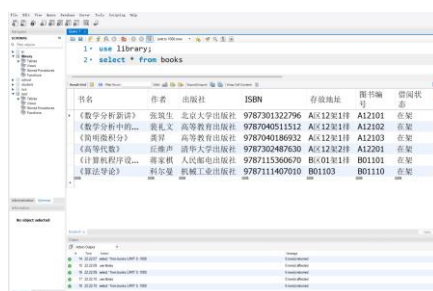


图 28 修改书籍信息后的数据库

5.2.7 借阅记录

作为图书馆信息管理系统，当有人借书后产生并保存借阅记录是十分必要的，它既可以作为借阅者归还书籍时系统操作的判断依据，也可以在图书馆书籍发生丢失时作为书籍寻找的线索。一条基础的借阅记录应包含借阅者的姓名、借阅者的学号（工号）、书籍名称、图书编号、借阅时间和借阅时长。本系统在借阅者借阅书籍的同时产生借阅记录，打开并将其保存在 MySQL 数据库中，同时在用户打开系统时将数据库中的借阅记录显示在 Listbox 组件做的空白页面上。普通用户登陆系统后将只能看见自己的借阅记录，而管理员用户则可以看见所有人的借阅记录。

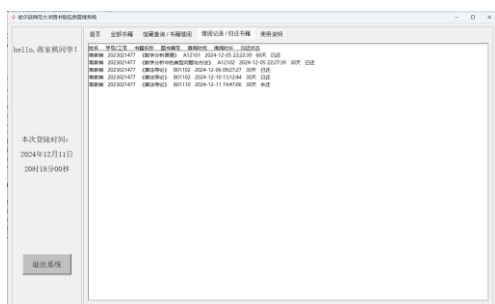


图 29 普通用户的借阅记录页面



图 30 管理员用户的借阅记录页面

5.2.8 书籍归还

书籍归还功能被嵌入在“借阅记录”页面之中。如果借阅记录的“归还状态”为“未还”，那么当用户选择用鼠标左键点击该条借阅记录时，就会弹出“归还书籍”的提示弹窗，用户

继续点击“确定”并退出系统重新进入后，被归还书籍信息的借阅状态将会变为“在架”，借阅记录的归还状态也会变为“已还”，数据库同步更新。

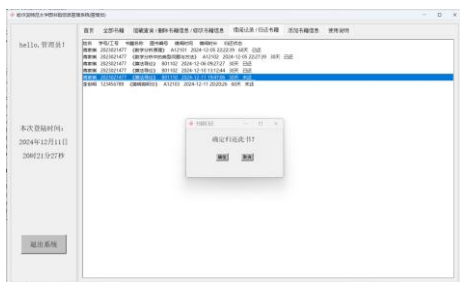


图 31 归还书籍

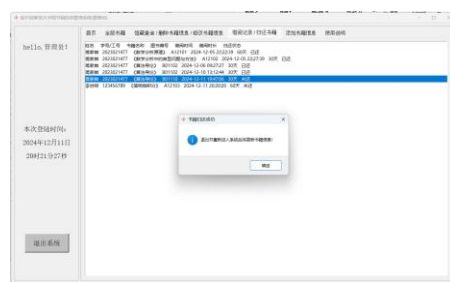


图 32 归还成功



图 33 归还成功后的书籍信息

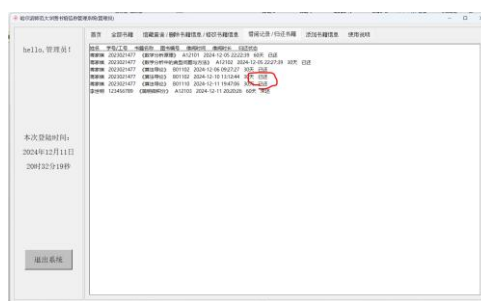


图 34 归还成功后的借阅记录

5.2.9 添加书籍（管理员用户功能）

图书馆的书籍不是不变的，往往需要添加新的书籍信息，所以本系统在管理员用户的主页面中专门设置一个 Notebook 页面作为“添加书籍信息”的功能页面。用户点击“添加书籍信息”导航按钮后，便会转跳到添加书籍信息功能页，之后按照对应的书籍信息字段向输入栏中输入书籍信息，输入完成之后点击“确认添加书籍信息”按钮，如果全部书籍信息之中没有和想要添加的书籍相同的图书编号信息，之后便会向数据库中成功添加书籍信息，否则弹出书籍添加失败的提示弹窗。



图 35 添加书籍信息页面



图 36 添加之后的全部书籍

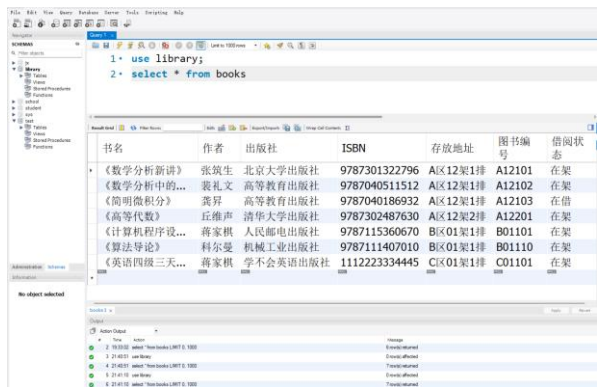


图 37 添加书籍信息后的数据库

5.2.10 使用说明

图书馆信息管理系统功能繁多，故添加使用说明帮助普通用户和管理员账户使用。该使用说明也单独使用一个 Notebook 页面，通过 Listbox 组件显示。使用说明的文本信息保存在一个 txt 文件之中，通过 Python 的 Open 方法打开文本文件，并将内容添加到 Listbox 组件页面中显示出来。



图 38 使用说明页面

通过以上功能展示，系统实现了用户登录、图书馆书籍信息的增删改查和对借阅记录的一些操作，使得图书馆信息管理更加方便、直观、高效。用户通过简单的操作，即可完成基本的书籍借阅、归还和书籍信息管理。

6 实现细节和技术挑战

6.1 数据库表的设计与优化

在数据库表的设计中，考虑到用户登录信息表、书籍信息表和借阅记录表的字段设置。采用合适的数据类型、约束和关联，确保表结构的合理性和一致性。例如，在书籍信息表中使用主键约束确保图书编号的唯一性。



6.2 Tkinter 界面的布局

6.2.1 用户界面设计

在 Tkinter 中,设计用户界面时需考虑布局的合理性和用户体验。采用 pack 布局管理器,以 pack 形式划分界面,使得控件的排列更加灵活。使用 Frame、Notebook 等组件进行模块化设计,使 GUI 页面代码更易维护。

6.2.2 控件的选择与配置

选择合适的 Tkinter 控件,例如 Entry、Listbox 等,根据实际需要配置控件的属性,确保界面的美观性和易用性。使用事件绑定机制处理用户的操作,提高界面的响应性。

6.3 SQL 查询语句的优化

通过分析查询语句的执行计划,进行必要的索引优化,减少不必要的全表扫描,提高查询效率。并将数据库操作被封装在函数中,使得数据操作更加集中和一致。

6.4 技术挑战与解决方案

6.4.1 挑战:不同用户的功能分离

图书馆信息管理系统对于不同用户,在功能上有一定差异,所以系统对不同用户所授予的权限不同。如普通用户可以借阅书籍而管理员用户不可以、管理员用户可以添加、删除和修改书籍信息,而普通用户不可以……

解决方案:采用函数式编程和引入全局变量。

通过函数式编程,将不同用户类型有差异的功能的操作封装在不同的函数里,并将登录信息和登陆后的主页面等变量设为全局变量,通过分支语句实现不同用户的不同功能。

6.4.2 挑战:SQL 查询的安全性

在数据库操作中,直接将变量拼接到 SQL 语句中可能会导致 SQL 注入攻击。

解决方案:使用参数化查询

在少部分使用案例中采用参数化查询,将用户参数与 SQL 查询语句分离,提高了 SQL 语言查询的安全性。

通过上述实现细节的考虑和技术挑战的解决方案,系统能够更稳定、高效地运行,提高用户体验,提高代码的可读性和数据库的安全性。

7 结论

通过本文对图书馆信息管理系统的设计与实现过程进行详细介绍,我们构建了一个基于 Python 语言、Tkinter 图形化界面和 MySQL 数据库的系统,实现了用户登录和对书籍信息和借阅记录信息的全面管理。在系统开发中,我们充分考虑了数据库表的设计、Tkinter 界面的布局、SQL 语句的优化等关键细节,以确保系统的稳定性、用户友好性和数据安全性。

系统采用了三层架构和 MVC 设计模式,使得系统各个模块相对独立,提高了系统的可维护性和可扩展性。数据库表的设计遵循范式化原则,优化了查询性能,保障了数据的一致性和完整性。Tkinter 图形化界面的设计考虑了用户友好性,通过合理的布局和控件选择,提高了系统的易用性。

在实现过程中,我们面临了 SQL 语言的安全性、不同用户的功能分离等技术挑战。通



过引入参数化查询和函数式编程等解决方案，成功应对了这些挑战，使系统在实际应用中更加可靠和高效。

展望未来，为了不断提升系统的性能和功能，可以考虑以下改进方向：

（1）采用面向对象编程：对于不同类型用户，虽然部分操作上有差异，但也有许多相同重复的功能操作。所以可以采用面向对象编程的方法，并将对象代码封装在不同的文件中，提供引入头文件的形式将不同用户界面实例化，并通过类的继承和派生的方法实现差异化，从而降低代码冗余提高代码可读性。

（2）用户输入验证：在用户输入数据时，可以增加更多的验证逻辑，确保输入数据的合法性和合理性。

综合来看，本图书馆信息管理系统在设计与实现中充分考虑了系统的各个方面。在未来的升级与改进中，我们将不断优化系统，以更好地满足用户需求，推动学校信息化管理的进一步发展。

参考文献：

[1] Python 官方文档. [<https://docs.python.org/>]

[2] Tkinter 官方文档.[<http://docs.python.org/3/library/tkinter.html>]

[3] MySQL 文档. [<https://dev.mysql.com/doc/>]



附录:

"""

项目名称: 图书馆信息管理系统

功能:

学生登陆, 教职工登录, 管理员登陆(√)

依照不同属性(书籍名称, 书籍作者, ISBN, 书籍编号)检索书籍信息(书籍名称, 书籍作者, 出版社, ISBN, 书籍编号, 存放地址, 借阅状态)(√)

全部书籍信息(√)

借阅书籍, 并添加借阅记录(√)

删除或修改书籍信息(√)

添加书籍信息(√)

显示图书借阅记录, 并还书(√)

退出图书馆信息管理系统(√)

作者: Chiang Jiach'i

项目创建时间: 西元 2024 年 11 月 22 日

"""

```
import tkinter as tk
from tkinter import font
from PIL import Image, ImageTk
from tkinter import messagebox, scrolledtext
from tkinter.ttk import *
import pymysql.cursors
from datetime import datetime as dt
```

"""窗口位置大小"""

```
def location(window, w, h):
    screenwidth = window.winfo_screenwidth()
    screenheight = window.winfo_screenheight()
    x = (screenwidth - w) / 2
    y = (screenheight - h) / 2
    window.geometry("%dx%d+%d+%d" % (w, h, x, y))
```

""" 右页: 首页 """

```
def begin(window, notbook):
    begin_frame = tk.Frame(window, bg='white')
    content = tk.Label(begin_frame,
                        text="\n\n>>> 欢迎登录哈尔滨师范大学图书馆信息管理系统\n\n<<<\n\nEdition: V1.0.0\n\n@Author: Chiang Jiach'i\n\nCreated Time: 2024 年 11 月 22 日",
                        font=font.Font(size=25), bg='white')
    content.pack(pady=100)
    notbook.add(begin_frame, text='首页')
```



```
""" 右页：浏览全部书籍 """
def all_books(window, notbook):
    all_books_frame = tk.Frame(window, bg='white')
    books_text_box = scrolledtext.ScrolledText(all_books_frame, width=200, height=100)
    books_text_box.pack(padx=10, pady=10)
    books_db = pymysql.connect(host='127.0.0.1', user='root', password='1205',
database='library', port=3306)
    cur = books_db.cursor()
    try:
        cur.execute('select * from books')
        books = cur.fetchall()
        s = ('\n' + '\t' + '书名' + '\t' + '\t' + '\t' + '\t' +
            '\t' + '作者' + '\t' + '\t' +
            '\t' + '出版社' + '\t' + '\t' +
            '\t' + 'ISBN 码' + '\t' + '\t' +
            '\t' + '存放地址' + '\t' + '\t' +
            '\t' + '图书编号' + '\t' + '\t' +
            '\t' + '借阅状态' + '\t' + '\t' + '\n' + '\n' + '\n')
        books_text_box.insert(tk.END, s)
        for row in books:
            book_data = '\t'
            for i in range(len(row)):
                if i == 0:
                    book_data += (str(row[i]) + '\t' + '\t' + '\t' + '\t')
                else:
                    book_data += (str(row[i]) + '\t' + '\t')

            books_text_box.insert(tk.END, book_data + '\n' + '\n' + '\n')
    except Exception as e:
        print("图书信息异常")
    finally:
        books_db.close()
        cur.close()
    notbook.add(all_books_frame, text='浏览全部书籍')

"""修改借阅状态并添加借阅记录"""
def add_borrow_record(window, num, select_text):
    window.destroy()
    messagebox.showinfo('借阅成功', '请在规定日期内归还书籍！')

    # 修改 MySQL 数据库借阅状态
    books_db = pymysql.connect(host='127.0.0.1', user='root', password='1205',
database='library', port=3306)
```



```
curl = books_db.cursor()
select_text_list = list(select_text.split(' '))
sql1 = f"UPDATE books SET 借阅状态 = '在借' WHERE 图书编号 =
'{select_text_list[5]}'"
curl.execute(sql1)
books_db.commit()
curl.close()
books_db.close()

# 添加借阅记录
borrow_record = []
# 这里需要确保 username 和 uuserno 是已经定义并且有值的变量
borrow_record.append(username) # 确保这些变量已经被定义
borrow_record.append(uuserno) # 确保这些变量已经被定义
borrow_record.append(select_text_list[0])
borrow_record.append(select_text_list[5])

current_time = dt.now().strftime('%Y-%m-%d %H:%M:%S') # 格式化当前时间为字符串
borrow_record.append(current_time)

if num == 1:
    period = '30 天'
    borrow_record.append(period)
else:
    period = '60 天'
    borrow_record.append(period)

borrow_record.append('未还')

borrow_record_db = pymysql.connect(host='127.0.0.1', user='root', password='1205',
database='library', port=3306)
cur2 = borrow_record_db.cursor()
# 使用参数化查询来避免 SQL 注入和格式错误
sql2 = """
    INSERT INTO borrow_record(sname, sno, bookname, booknumber, borrowtime,
borrowperiod, returnstatus)
    VALUES (%s, %s, %s, %s, %s, %s, %s)
    """
cur2.execute(sql2, tuple(borrow_record))
cur2.close()
borrow_record_db.commit() # 提交事务
borrow_record_db.close()
```




```
"""确认借阅"""
def Borrow_Book_Yes(borrow_book_frame, selected_text):
    borrow_book_frame.destroy()
    if selected_text[-6:-4] != '在架':
        messagebox.showerror('借阅失败', '本书在借中，请重新选择书籍！')
    else:
        borrow_period_frame = tk.Toplevel() # 创建一个新的 Toplevel 窗口
        borrow_period_frame.title("借阅时长")
        borrow_period_frame.iconbitmap("output_icon.ico")

        location(borrow_period_frame, 250, 300) # 假设这是设置位置的函数

        var = tk.IntVar()
        var.set(1)

        large_font = font.Font(size=15)

        label = tk.Label(borrow_period_frame, text='请选择借阅时长', font=large_font)
        label.pack(pady=30)

        period1 = tk.Radiobutton(borrow_period_frame, text='30 天', font=large_font,
                                variable=var, value=1)
        period1.pack(pady=0)

        period2 = tk.Radiobutton(borrow_period_frame, text='60 天', font=large_font,
                                variable=var, value=2)
        period2.pack(pady=30)

        # 直接在按钮的命令中获取 Radiobutton 的值
        button = tk.Button(borrow_period_frame, text='确定', bg='silver', font=large_font,
                           width=10,
                           command=lambda: add_borrow_record(borrow_period_frame,
                                                               var.get(), selected_text))
        button.pack()

        borrow_period_frame.mainloop()

"""借阅书籍 """
def borrow_book():
    selection = books_text_box.curselection()
    if selection:
        # 获取选中项的文本
        selected_text = books_text_box.get(selection)
        if selected_text[-6:-4] == '在架' or selected_text[-6:-4] == '在借':
```



```
borrow_book_frame = tk.Toplevel() # 创建一个新的 Toplevel 窗口
borrow_book_frame.title("书籍借阅")
borrow_book_frame.iconbitmap("output_icon.ico")

location(borrow_book_frame, 300, 150) # 假设这是设置位置的函数

frame1=tk.Frame(borrow_book_frame)
frame1.pack(pady=20)
frame2=tk.Frame(borrow_book_frame)
frame2.pack(pady=10)

large_font = font.Font(size=15)
lite_font = font.Font(size=10)

label = tk.Label(frame1,text='确定借阅此书? ',font=large_font)
label.pack()

borrow_yes = tk.Button(frame2,text='确定',bg='silver',font=lite_font, command =
lambda : Borrow_Book_Yes(borrow_book_frame,selected_text))
borrow_yes.pack(side='left',padx=20)
borrow_no = tk.Button(frame2,text='取消',bg='silver',font=lite_font, command =
lambda : borrow_book_frame.destroy())
borrow_no.pack(side='left',padx=20)

borrow_book_frame.mainloop()

# 销毁页面上的所有组件
def clear_page(page):
    for widget in page.winfo_children():
        widget.destroy()

"""检索书籍"""
def book_search_button(page, book_name, book_author, book_publisher, book_ISBN,
book_number):
    clear_page(page)
    global books_text_box # 使用全局变量 books_text_box
    books_text_box = tk.Listbox(page, width=200, height=100)
    books_text_box.pack(padx=10, pady=10)
    books_db = pymysql.connect(host='127.0.0.1', user='root', password='1205',
database='library', port=3306)
    cur = books_db.cursor()
    try:
        if book_name != "" or book_author != "" or book_publisher != "" or book_ISBN != "" or
```



```

book_number != "":
    sql = 'select * from books where'
    book_entry = [book_name, book_author, book_publisher, book_ISBN,
book_number]
    book_attribute = ['书名', '作者', '出版社', 'ISBN', '图书编号']
    space_count = 1
    for i in range(len(book_entry)):
        if book_entry[i] != "":
            if space_count == 1:
                sql += f" {book_attribute[i]} = '{book_entry[i]}'"
                space_count += 1
            else:
                sql += f" and {book_attribute[i]} = '{book_entry[i]}'"
    else:
        sql = 'select * from books'

cur.execute(sql)
books = cur.fetchall()
top = '书名      作者      出版社      ISBN 码      存放地址      图书编号
借阅状态 '
books_text_box.insert(tk.END, top)
for row in books:
    book_data = ""
    for i in range(len(row)):
        book_data += (row[i] + '      ')
    books_text_box.insert(tk.END, book_data)

if page == search_books_frame:
    books_text_box.bind('<<ListboxSelect>>', lambda event: borrow_book()) # 绑定选择事件
else:
    books_text_box.bind('<<ListboxSelect>>', lambda event: choose_delete_modify())
# 绑定选择事件

except Exception as e:
    print("图书信息异常", e)
finally:
    books_db.close()
    cur.close()

""" 右页：根据不同属性检索书籍信息 """
search_books_frame = None
def search_books(window, notbook):
    large_font = font.Font(size=30)

```



```
global search_books_frame
search_books_frame = tk.Frame(window)

tips_frame = tk.Frame(search_books_frame)
tips_frame.pack(pady=50)
tips_label = tk.Label(tips_frame, text='>>>请输入检索字段（可不全填），输入书名时请加
上书名号<<<', font=font.Font(size=17))
tips_label.pack(side='left')

book_name_frame = tk.Frame(search_books_frame)
book_name_frame.pack(pady=0)
book_name_label = tk.Label(book_name_frame, text='书 名：', font=large_font)
book_name_label.pack(side='left')
global book_name_entry
book_name_entry = tk.Entry(book_name_frame, font=large_font)
book_name_entry.pack()

book_author_frame = tk.Frame(search_books_frame)
book_author_frame.pack(pady=50)
book_author_label = tk.Label(book_author_frame, text='作 者：', font=large_font)
book_author_label.pack(side='left')
global book_author_entry
book_author_entry = tk.Entry(book_author_frame, font=large_font)
book_author_entry.pack()

book_publisher_frame = tk.Frame(search_books_frame)
book_publisher_frame.pack(pady=0)
book_publisher_label = tk.Label(book_publisher_frame, text='出版社：', font=large_font)
book_publisher_label.pack(side='left')
global book_publisher_entry
book_publisher_entry = tk.Entry(book_publisher_frame, font=large_font)
book_publisher_entry.pack()

book_ISBN_frame = tk.Frame(search_books_frame)
book_ISBN_frame.pack(pady=50)
book_ISBN_label = tk.Label(book_ISBN_frame, text='ISBN 码：', font=large_font)
book_ISBN_label.pack(side='left')
global book_ISBN_entry
book_ISBN_entry = tk.Entry(book_ISBN_frame, font=large_font)
book_ISBN_entry.pack()

book_number_frame = tk.Frame(search_books_frame)
book_number_frame.pack(pady=0)
book_number_label = tk.Label(book_number_frame, text='编 号：', font=large_font)
```



```
book_number_label.pack(side='left')
global book_number_entry
book_number_entry = tk.Entry(book_number_frame, font=large_font)
book_number_entry.pack()

# 确认查询按钮的回调函数
def on_search():
    book_search_button(search_books_frame, book_name_entry.get(),
book_author_entry.get(), book_publisher_entry.get(), book_ISBN_entry.get(),
book_number_entry.get())

    book_search = tk.Button(search_books_frame, text='确认查询', font=font.Font(size=20),
padx=50, bg='silver',
                                command=on_search)
    book_search.pack(pady=50)

notbook.add(search_books_frame, text='馆藏查询 / 书籍借阅')

"""删除书籍信息"""
def Delete(frame, selected_text):
    frame.destroy()

    # 修改 MySQL 数据库书籍信息
    books_db = pymysql.connect(host='127.0.0.1', user='root', password='1205',
database='library', port=3306)
    cur = books_db.cursor()
    select_text_list = list(selected_text.split(' '))
    sql = f"DELETE from books WHERE 图书编号 = '{select_text_list[5]}'"
    try:
        # SQL 执行代码
        cur.execute(sql)
        books_db.commit()
        messagebox.showinfo('删除成功', '退出并重新进入系统后将更新书籍信息!')

    except pymysql.Error as e:
        print(f"Error: {e}")
        messagebox.showerror('删除失败', '无法删除书籍信息，请检查错误信息!')
    finally:
        cur.close()
        books_db.close()

"""确定删除书籍信息? """
def delete_book(frame):
    frame.destroy()
```




```
selection = books_text_box.curselection()
if selection:
    # 获取选中项的文本
    selected_text = books_text_box.get(selection)

    delete_book_frame = tk.Toplevel() # 创建一个新的 Toplevel 窗口
    delete_book_frame.title("删除修改书籍信息")
    delete_book_frame.iconbitmap("output_icon.ico")

    location(delete_book_frame, 300, 150) # 假设这是设置位置的函数

    frame1 = tk.Frame(delete_book_frame)
    frame1.pack(pady=20)
    frame2 = tk.Frame(delete_book_frame)
    frame2.pack(pady=10)

    large_font = font.Font(size=15)
    lite_font = font.Font(size=10)

    label = tk.Label(frame1, text='确定要删除该条书籍信息吗?', font=large_font)
    label.pack()

    delete_yes = tk.Button(frame2, text='确定', bg='silver', font=lite_font,
                           command=lambda: Delete(delete_book_frame,
selected_text))
    delete_yes.pack(side='left', padx=20)
    delete_no = tk.Button(frame2, text='取消', bg='silver', font=lite_font,
                           command=lambda: delete_book_frame.destroy())
    delete_no.pack(side='left', padx=20)

    delete_book_frame.mainloop()

"""修改书籍信息"""
def modify(frame, book_name, book_author, book_publisher, book_ISBN_, book_address,
book_status):
    frame.destroy()
    selection = books_text_box.curselection()
    if selection:
        # 获取选中项的文本
        selected_text = books_text_box.get(selection)
        lst = list(selected_text.split(' '))
        book_number = lst[5]
        # book_number = selected_text[-16:-10]
        # print(book_number)
```



```

        book_entry
    [book_name,book_author,book_publisher,book_ISBN_,book_address,book_status]
    book_attribute = ['书名','作者','出版社','ISBN','存放地址','借阅状态']
    books_db = pymysql.connect(host='127.0.0.1', user='root', password='1205',
database='library', port=3306)
    cur = books_db.cursor()
    try:
        if lst[6] == '在架':
            for i in range(len(book_entry)):
                if book_entry[i] != '':
                    sql = f'update books set {book_attribute[i]} = '{book_entry[i]}'
where 图书编号 = '{book_number}'"
                    cur.execute(sql)
                    books_db.commit()
                    messagebox.showinfo('修改成功','退出并重新进入系统后将更新书籍信息！')
    ')
        else:
            messagebox.showerror('修改失败','在借状态禁止修改书籍信息！')
    except pymysql.Error as e:
        print(f"Error: {e}")
        messagebox.showerror('修改失败','无法修改书籍信息，请检查错误信息！')
    finally:
        cur.close()
        books_db.close()

"""修改书籍信息界面"""
def modify_book(frame):
    frame.destroy()

    large_font = font.Font(size=17)

    modify_frame = tk.Toplevel() # 创建一个新的 Toplevel 窗口
    modify_frame.title("修改书籍信息")
    modify_frame.iconbitmap("output_icon.ico")

    location(modify_frame, 500, 650)

    tips_frame = tk.Frame(modify_frame)
    tips_frame.pack(pady=40)
    tips_label = tk.Label(tips_frame, text='>>>请依照字段提示输入修改后的内容<<<',
font=font.Font(size=15))
    tips_label.pack(side='left')

```



```
book_name_frame1 = tk.Frame(modify_frame)
book_name_frame1.pack(pady=0)
book_name_label1 = tk.Label(book_name_frame1, text='书 名: ', font=large_font)
book_name_label1.pack(side='left')
book_name_entry1 = tk.Entry(book_name_frame1, font=large_font)
book_name_entry1.pack()

book_author_frame1 = tk.Frame(modify_frame)
book_author_frame1.pack(pady=40)
book_author_label1 = tk.Label(book_author_frame1, text='作 者: ', font=large_font)
book_author_label1.pack(side='left')
book_author_entry1 = tk.Entry(book_author_frame1, font=large_font)
book_author_entry1.pack()

book_publisher_frame1 = tk.Frame(modify_frame)
book_publisher_frame1.pack(pady=0)
book_publisher_label1 = tk.Label(book_publisher_frame1, text='出版社: ', font=large_font)
book_publisher_label1.pack(side='left')
book_publisher_entry1 = tk.Entry(book_publisher_frame1, font=large_font)
book_publisher_entry1.pack()

book_ISBN_frame1 = tk.Frame(modify_frame)
book_ISBN_frame1.pack(pady=40)
book_ISBN_label1 = tk.Label(book_ISBN_frame1, text='ISBN 号: ', font=large_font)
book_ISBN_label1.pack(side='left')
book_ISBN_entry1 = tk.Entry(book_ISBN_frame1, font=large_font)
book_ISBN_entry1.pack()

book_address_frame1 = tk.Frame(modify_frame)
book_address_frame1.pack(pady=0)
book_address_label1 = tk.Label(book_address_frame1, text='地 址: ', font=large_font)
book_address_label1.pack(side='left')
book_address_entry1 = tk.Entry(book_address_frame1, font=large_font)
book_address_entry1.pack()

# book_number_frame1 = tk.Frame(modify_frame)
# book_number_frame1.pack(pady=40)
# book_number_label1 = tk.Label(book_number_frame1, text='编 号: ', font=large_font)
# book_number_label1.pack(side='left')
# book_number_entry1 = tk.Entry(book_number_frame1, font=large_font)
# book_number_entry1.pack()

book_status_frame1 = tk.Frame(modify_frame)
book_status_frame1.pack(pady=40)
```



```
book_status_label1 = tk.Label(book_status_frame1, text='状态: ', font=large_font)
book_status_label1.pack(side='left')
book_status_entry1 = tk.Entry(book_status_frame1, font=large_font)
book_status_entry1.pack()

def on_modify():

modify(modify_frame,book_name_entry1.get(),book_author_entry1.get(),book_publisher_entry1.
get(),book_ISBN_entry1.get(),book_address_entry1.get(),book_status_entry1.get())

button_frame = tk.Frame(modify_frame)
button_frame.pack(pady=0)
modify_YES = tk.Button(button_frame, text='确认修改', font = font.Font(size = 15), padx =
10, bg='silver',width = 6,
                        command = on_modify)
modify_YES.pack(side='left', padx=30)
modify_NO = tk.Button(button_frame, text='取消修改', font = font.Font(size = 15), padx =
10, bg='silver',width = 6,
                        command = lambda : modify_frame.destroy())
modify_NO.pack(side='left', padx=30)

modify_frame.mainloop()

"""删除修改书籍信息选择界面"""
def choose_delete_modify():
    selection = books_text_box.curselection()
    if selection:
        # 获取选中项的文本
        selected_text = books_text_box.get(selection)
        if selected_text[-6:-4] == '在架' or selected_text[-6:-4] == '在借':

            choose_delete_modify_frame = tk.Toplevel() # 创建一个新的 Toplevel 窗口
            choose_delete_modify_frame.title("删除修改书籍信息")
            choose_delete_modify_frame.iconbitmap("output_icon.ico")

            location(choose_delete_modify_frame, 300, 150) # 假设这是设置位置的函数

            frame1 = tk.Frame(choose_delete_modify_frame)
            frame1.pack(pady=20)
            frame2 = tk.Frame(choose_delete_modify_frame)
            frame2.pack(pady=10)
```



```
large_font = font.Font(size=15)
lite_font = font.Font(size=10)

label = tk.Label(frame1, text='请选择你要执行的操作项目', font=large_font)
label.pack()

delete_button = tk.Button(frame2, text='删除书籍信息', bg='silver', font=lite_font,
                           command=lambda:
delete_book(choose_delete_modify_frame))
delete_button.pack(side='left', padx=20)
modify_button = tk.Button(frame2, text='修改书籍信息', bg='silver',
font=lite_font,
                           command=lambda:
modify_book(choose_delete_modify_frame))
modify_button.pack(side='left', padx=20)

choose_delete_modify_frame.mainloop()
```

""" 右页：根据不同属性检索书籍信息（管理员版） """

```
def search_books_admin(window, notbook):
    large_font = font.Font(size=30)
    search_books_frame_admin = tk.Frame(window)

    tips_frame = tk.Frame(search_books_frame_admin)
    tips_frame.pack(pady=50)
    tips_label = tk.Label(tips_frame, text='>>>请输入检索字段（可不全填），输入书名时请加上书名号<<<', font=font.Font(size=17))
    tips_label.pack(side='left')

    book_name_frame = tk.Frame(search_books_frame_admin)
    book_name_frame.pack(pady=0)
    book_name_label = tk.Label(book_name_frame, text='书 名：', font=large_font)
    book_name_label.pack(side='left')
    global book_name_entry
    book_name_entry = tk.Entry(book_name_frame, font=large_font)
    book_name_entry.pack()

    book_author_frame = tk.Frame(search_books_frame_admin)
    book_author_frame.pack(pady=50)
    book_author_label = tk.Label(book_author_frame, text='作 者：', font=large_font)
    book_author_label.pack(side='left')
    global book_author_entry
    book_author_entry = tk.Entry(book_author_frame, font=large_font)
    book_author_entry.pack()
```




```
book_publisher_frame = tk.Frame(search_books_frame_admin)
book_publisher_frame.pack(pady=0)
book_publisher_label = tk.Label(book_publisher_frame, text='出版社: ', font=large_font)
book_publisher_label.pack(side='left')
global book_publisher_entry
book_publisher_entry = tk.Entry(book_publisher_frame, font=large_font)
book_publisher_entry.pack()

book_ISBN_frame = tk.Frame(search_books_frame_admin)
book_ISBN_frame.pack(pady=50)
book_ISBN_label = tk.Label(book_ISBN_frame, text='ISBN 码: ', font=large_font)
book_ISBN_label.pack(side='left')
global book_ISBN_entry
book_ISBN_entry = tk.Entry(book_ISBN_frame, font=large_font)
book_ISBN_entry.pack()

book_number_frame = tk.Frame(search_books_frame_admin)
book_number_frame.pack(pady=0)
book_number_label = tk.Label(book_number_frame, text='编 号: ', font=large_font)
book_number_label.pack(side='left')
global book_number_entry
book_number_entry = tk.Entry(book_number_frame, font=large_font)
book_number_entry.pack()

# 确认查询按钮的回调函数
def on_search():
    book_search_button(search_books_frame_admin, book_name_entry.get(),
book_author_entry.get(), book_publisher_entry.get(), book_ISBN_entry.get(),
book_number_entry.get())

    book_search = tk.Button(search_books_frame_admin, text=' 确 认 查 询 ',
font=font.Font(size=20), padx=50, bg='silver',
command=on_search)
    book_search.pack(pady=50)

    notbook.add(search_books_frame_admin, text='馆藏查询 / 删除书籍信息 / 修改书籍信
息')

"""确定还书? """
def Return_Book_Yes(return_book_frame, selected_text):
    return_book_frame.destroy()
    messagebox.showinfo('书籍归还成功', '退出并重新进入系统后将更新书籍信息!')
```



```
# 修改 MySQL 数据库借阅状态
books_db = pymysql.connect(host='127.0.0.1', user='root', password='1205',
database='library', port=3306)
cur1 = books_db.cursor()
select_text_list = list(selected_text.split(' '))
sql1 = f'UPDATE books SET 借阅状态 = '在架' WHERE 图书编号 =
'{select_text_list[3]}'
cur1.execute(sql1)
books_db.commit()
cur1.close()
books_db.close()

# 修改书籍归还状态
records_db = pymysql.connect(host='127.0.0.1', user='root', password='1205',
database='library', port=3306)
cur2 = records_db.cursor()
sql2 = f'UPDATE borrow_record SET returnstatus = '已还' WHERE booknumber =
'{select_text_list[3]}' and borrowtime = '{select_text_list[4]}' '
cur2.execute(sql2)
records_db.commit()
cur2.close()
records_db.close()

"""还书"""
def return_book():
    selection = records_text_box.curselection()
    if selection:
        # 获取选中项的文本
        selected_text = records_text_box.get(selection)
        if selected_text[-5:-3] == '未还':

            return_book_frame = tk.Toplevel() # 创建一个新的 Toplevel 窗口
            return_book_frame.title("书籍归还")
            return_book_frame.iconbitmap("output_icon.ico")

            location(return_book_frame, 300, 150) # 假设这是设置位置的函数

            frame1=tk.Frame(return_book_frame)
            frame1.pack(pady=20)
            frame2=tk.Frame(return_book_frame)
            frame2.pack(pady=10)

            large_font = font.Font(size=15)
            lite_font = font.Font(size=10)
```



```
label = tk.Label(frame1,text='确定归还此书? ',font=large_font)
label.pack()

return_yes = tk.Button(frame2,text='确定',bg='silver',font=lite_font, command =
lambda : Return_Book_Yes(return_book_frame,selected_text))
return_yes.pack(side='left',padx=20)
return_no = tk.Button(frame2,text='取消',bg='silver',font=lite_font, command =
lambda : return_book_frame.destroy())
return_no.pack(side='left',padx=20)

return_book_frame.mainloop()
```

"""右页：借阅记录界面"""

```
def borrow_records(window,notbook,basedwindow):
```

```
    borrow_records_frame = tk.Frame(window)
```

```
    global records_text_box
```

```
    records_text_box = tk.Listbox(borrow_records_frame, width=200, height=100)
```

```
    records_text_box.pack(padx=10, pady=10)
```

```
    top = '姓名      学号/工号      书籍名称      图书编号      借阅时间      借阅时长
归还状态'
```

```
    records_text_box.insert(tk.END,top)
```

```
    records_db = pymysql.connect(host='127.0.0.1', user='root', password='1205',
database='library', port=3306)
```

```
    cur = records_db.cursor()
```

```
    if basedwindow == student_teacher:
```

```
        sql = f'select * from borrow_record where sno = {uuserno}'
```

```
    else:
```

```
        sql = 'select * from borrow_record'
```

```
    cur.execute(sql)
```

```
    records = cur.fetchall()
```

```
    for record in records:
```

```
        format_record = ''
```

```
        for data in record:
```

```
            format_record += (data+'  ')
```

```
        records_text_box.insert(tk.END,format_record)
```

```
    records_text_box.bind('<<ListboxSelect>>', lambda event: return_book()) # 绑定选择事件
```



```
notbook.add(borrow_records_frame,text='借阅记录 / 归还书籍')

records_db.close()
cur.close()

"""使用说明界面"""
def instruction(window, notbook):
    instructions_frame = tk.Frame(window, bg='white')
    text_box = tk.Listbox(instructions_frame, width=200, height=100)
    text_box.pack(padx=10, pady=10)

    try:
        with open(r'using_instruction.txt', 'r', encoding='utf-8') as file:
            lines = file.readlines()
            for line in lines:
                text_box.insert(tk.END, line.strip())
    except FileNotFoundError:
        print("文件未找到， 请检查文件路径。")
    except Exception as e:
        print(f"读取文件时发生错误: {e}")

    notbook.add(instructions_frame, text='使用说明')

"""右页"""
def right_frame(window):
    rightframe_notbook = Notebook(window)
    begin(window,rightframe_notbook)
    all_books(window,rightframe_notbook)
    search_books(window,rightframe_notbook)
    borrow_records(window, rightframe_notbook,student_teacher)
    instruction(window,rightframe_notbook)

    # 配置 Notebook 的样式
    style = Style()
    style.configure("TNotebook.Tab", font=('Arial', 12), padding=(10, 5)) # 设置字体大小
    和选项卡内边距

    rightframe_notbook.pack(fill='both', expand=True)

"""添加书籍信息"""
def add(book_name, book_author, book_publisher, book_ISBN, book_address, book_number,
book_status):
```



```
books_db = pymysql.connect(host='127.0.0.1', user='root', password='1205',
database='library', port=3306)
cur1 = books_db.cursor()
sql1 = 'select 图书编号 from books'
cur1.execute(sql1)
book_numbers = cur1.fetchall()
book_numberss = []
for number in book_numbers:
    book_numberss.append(number)
cur1.close()
books_db.close()

books_db = pymysql.connect(host='127.0.0.1', user='root', password='1205',
database='library', port=3306)
cur2 = books_db.cursor()
sql2 = f"insert into books(书名,作者,出版社,ISBN,存放地址,图书编号,借阅状态)values('{book_name}','{book_author}','{book_publisher}','{book_ISBN}','{book_address}',
'{book_number}','{book_status}')"
try:
    if book_number not in book_numberss:
        cur2.execute(sql2)
        books_db.commit()
        messagebox.showinfo('添加成功','退出并重新进入系统后将更新书籍信息！')
    else:
        messagebox.showerror('添加失败','图书编号已存在！')
except pymysql.Error as e:
    print(f"Error: {e}")
    messagebox.showerror('添加失败','无法修改书籍信息，请检查错误信息！')
finally:
    cur2.close()
    books_db.close()

"""右页：管理员添加书籍界面"""
def add_book(window, notbook):
    add_book_frame = tk.Frame(window)
    large_font = font.Font(size=25)

    tips_frame = tk.Frame(add_book_frame)
    tips_frame.pack(pady=30)
    tips_label = tk.Label(tips_frame, text='>>>请依照字段提示输入待添加的书籍信息<<<',
font=font.Font(size=15))
    tips_label.pack(side='left')

    book_name_frame1 = tk.Frame(add_book_frame)
```



```
book_name_frame1.pack(pady=0)
book_name_label1 = tk.Label(book_name_frame1, text='书 名: ', font=large_font)
book_name_label1.pack(side='left')
book_name_entry1 = tk.Entry(book_name_frame1, font=large_font)
book_name_entry1.pack()

book_author_frame1 = tk.Frame(add_book_frame)
book_author_frame1.pack(pady=30)
book_author_label1 = tk.Label(book_author_frame1, text='作 者: ', font=large_font)
book_author_label1.pack(side='left')
book_author_entry1 = tk.Entry(book_author_frame1, font=large_font)
book_author_entry1.pack()

book_publisher_frame1 = tk.Frame(add_book_frame)
book_publisher_frame1.pack(pady=0)
book_publisher_label1 = tk.Label(book_publisher_frame1, text='出版社: ', font=large_font)
book_publisher_label1.pack(side='left')
book_publisher_entry1 = tk.Entry(book_publisher_frame1, font=large_font)
book_publisher_entry1.pack()

book_ISBN_frame1 = tk.Frame(add_book_frame)
book_ISBN_frame1.pack(pady=30)
book_ISBN_label1 = tk.Label(book_ISBN_frame1, text='ISBN 号: ', font=large_font)
book_ISBN_label1.pack(side='left')
book_ISBN_entry1 = tk.Entry(book_ISBN_frame1, font=large_font)
book_ISBN_entry1.pack()

book_address_frame1 = tk.Frame(add_book_frame)
book_address_frame1.pack(pady=0)
book_address_label1 = tk.Label(book_address_frame1, text='地 址: ', font=large_font)
book_address_label1.pack(side='left')
book_address_entry1 = tk.Entry(book_address_frame1, font=large_font)
book_address_entry1.pack()

book_number_frame1 = tk.Frame(add_book_frame)
book_number_frame1.pack(pady=30)
book_number_label1 = tk.Label(book_number_frame1, text='编 号: ', font=large_font)
book_number_label1.pack(side='left')
book_number_entry1 = tk.Entry(book_number_frame1, font=large_font)
book_number_entry1.pack()

book_status_frame1 = tk.Frame(add_book_frame)
book_status_frame1.pack(pady=0)
book_status_label1 = tk.Label(book_status_frame1, text='状 态: ', font=large_font)
```




```
book_status_label1.pack(side='left')
book_status_entry1 = tk.Entry(book_status_frame1, font=large_font)
book_status_entry1.pack()

def on_add():
    add(book_name_entry1.get(), book_author_entry1.get(), book_publisher_entry1.get(),
book_ISBN_entry1.get(), book_address_entry1.get(), book_number_entry1.get(),
book_status_entry1.get())

add_YES = tk.Button(add_book_frame, text='确认添加书籍信息', font=font.Font(size=20),
padx=20, bg='silver', command=on_add)
add_YES.pack(pady=30)

notbook.add(add_book_frame, text='添加书籍信息')

"""右页（管理员版）"""
def right_frame_admin(window):
    rightframe_notbook = Notebook(window)
    begin(window, rightframe_notbook)
    all_books(window, rightframe_notbook)
    search_books_admin(window, rightframe_notbook)
    borrow_records(window, rightframe_notbook, admin)
    add_book(window, rightframe_notbook)
    instruction(window, rightframe_notbook)

# 配置 Notebook 的样式
style = Style()
style.configure("TNotebook.Tab", font=('Arial', 12), padding=(10, 5)) # 设置字体大小
和选项卡内边距

rightframe_notbook.pack(fill='both', expand=True)

"""左页"""
def left_frame(login_window, username, window):
    large_font = font.Font(size=16)
    if login_window == login_student_window:
        top = tk.Label(window, text=f"hello, {username} 同学!", font=large_font)
        top.pack(pady=50)
    if login_window == login_teacher_window:
        top = tk.Label(window, text=f"hello, {username} 老师!", font=large_font)
        top.pack(pady=50)
    if login_window == login_admin_window:
        top = tk.Label(window, text=f"hello, 管理员!", font=large_font)
```



```
top.pack(pady=50)

current_time = dt.now()
formatted_time = current_time.strftime("%Y 年%m 月%d 日\n\n%H 时%M 分%S 秒")
time_label = tk.Label(window, text=f'本次登陆时间: \n\n{formatted_time}', font=large_font)
time_label.pack(pady=180)

# 检查 student_teacher 是否为 None
if login_window == login_student_window and student_teacher is not None:
    exit_button = tk.Button(window, text='退出系统', padx=20, pady=12, font=large_font,
bg='silver',
                                command=lambda: student_teacher.destroy())
    exit_button.pack(pady=50)
# 检查 admin 是否为 None
elif login_window == login_admin_window and admin is not None:
    exit_button = tk.Button(window, text='退出系统', padx=20, pady=12, font=large_font,
bg='silver',
                                command=lambda: admin.destroy())
    exit_button.pack(pady=50)
else:
    # 如果 student_teacher 或 admin 为 None, 不创建退出按钮或提供其他处理方式
    pass

"""学生教师界面"""
student_teacher = None
def studentteacher(username, userno, login_window):
    global username
    username = username
    global userno
    userno = userno

    global student_teacher
    student_teacher = tk.Tk()
    student_teacher.title("哈尔滨师范大学图书馆信息管理系统")
    student_teacher.iconbitmap("output_icon.ico")

    location(student_teacher, 1400, 820)

    # 设置左右栏
    pw = tk.PanedWindow()
    leftframe = tk.LabelFrame(pw, width=400, height=1175)
    pw.add(leftframe)
    rightframe = tk.LabelFrame(pw, width=400, height=1175)
```



```
pw.add(rightframe)
pw.pack(fill='both',expand=True,padx=10,pady=10)

left_frame(login_window,username,leftframe)
right_frame(rightframe)

student_teacher.mainloop()

"""管理员界面"""
admin = None
def adminn(username,login_window):
    global admin
    admin = tk.Tk()
    admin.title("哈尔滨师范大学图书馆信息管理系统(管理员)")
    admin.iconbitmap("output_icon.ico")

    location(admin,1400,820)

    # 设置左右栏
    pw = tk.PanedWindow()
    leftframe = tk.LabelFrame(pw,width=400,height=1175)
    pw.add(leftframe)
    rightframe = tk.LabelFrame(pw,width=400,height=1175)
    pw.add(rightframe)
    pw.pack(fill='both',expand=True,padx=10,pady=10)

    left_frame(login_window, username, leftframe)
    right_frame_admin(rightframe)

    admin.mainloop()

"""登录成功"""
def success_tip(username,userno,login_window):
    login_global_window.destroy()
    login_window.destroy()
    if login_window == login_student_window or login_window == login_teacher_window:
        studentteacher(username,userno,login_window)
    else:
        adminn(username,login_window)

"""登陆失败"""
def fail_tip(login_window):
    if login_window == login_student_window:
```



```
        messagebox.showerror("登陆失败！","姓名或学号错误！")
    elif login_window == login_teacher_window:
        messagebox.showerror("登陆失败！","姓名或工号错误！")
    else:
        messagebox.showerror("登陆失败！","管理员账号或密码错误！")

"""登录按钮逻辑"""
# 全局变量声明
login_student_window = None
login_teacher_window = None
login_admin_window = None
def auto_login(window):
    # 连接数据库
    db = pymysql.connect(host='127.0.0.1', user='root', password='1205', db='school',
port=3306)
    # 获取操作游标
    cur = db.cursor()
    # 分类查询数据库
    if window == login_student_window:
        sql = 'select * from student'
    elif window == login_teacher_window:
        sql = 'select * from teacher'
    else:
        sql = 'select * from admin'

    entry1 = name.get()
    entry2 = no.get()
    flag = False
    try:
        cur.execute(sql) # 执行查询
        results = cur.fetchall() # 获取所有查询数据
        for row in results:
            uid = row[0] # 姓名或账号
            pwd = row[1] # 学号或工号或密码
            # 判断输入的账号和密码是否正确
            if entry1 == uid and entry2 == pwd:
                username = row[0]
                userno = row[1]
                success_tip(username,userno, window)
                flag = True
                break
        if not flag:
            fail_tip(window)
    except Exception as e:
```



```
        fail_tip(window)
    finally:
        cur.close()
        db.close()

"""学生登录"""
def login_student():
    global login_student_window
    login_student_window=tk.Tk()
    login_student_window.title("学生登录")
    login_student_window.iconbitmap("output_icon.ico")

    location(login_student_window,300,150)

    fsname=tk.Frame(login_student_window)
    fsname.pack(pady=10)
    fsno = tk.Frame(login_student_window)
    fsno.pack(pady=10)
    fslog = tk.Frame(login_student_window)
    fslog.pack(pady=10)

    lsname = tk.Label(fsname, text="姓名: ")
    lsname.pack(side="left")
    global name
    name = tk.Entry(fsname)
    name.pack()

    lsno = tk.Label(fsno, text="学号: ")
    lsno.pack(side="left")
    global no
    no = tk.Entry(fsno)
    no.pack()

    login_student = tk.Button(fslog, text=" 登 录 ",command= lambda :
auto_login(login_student_window))
    login_student.pack(side='left',padx=20)
    login_student_no = tk.Button(fslog, text=" 取 消 ", command=lambda:
login_student_window.destroy())
    login_student_no.pack(side='left',padx=20)

    login_student_window.mainloop()

"""教师登录"""
def login_teacher():
```



```
global login_teacher_window
login_teacher_window=tk.Tk()
login_teacher_window.title("教师登录")
login_teacher_window.iconbitmap("output_icon.ico")

location(login_teacher_window,300,150)

ftname=tk.Frame(login_teacher_window)
ftname.pack(pady=10)
ftno = tk.Frame(login_teacher_window)
ftno.pack(pady=10)
ftlog = tk.Frame(login_teacher_window)
ftlog.pack(pady=10)

ltname = tk.Label(ftname, text="姓名: ")
ltname.pack(side="left")
global name
name = tk.Entry(ftname)
name.pack()

ltno = tk.Label(ftno, text="工号: ")
ltno.pack(side="left")
global no
no = tk.Entry(ftno)
no.pack()

login_teacher = tk.Button(ftlog, text="登 录",command=lambda :
auto_login(login_teacher_window))
login_teacher.pack(side='left',padx=20)
login_teacher_no = tk.Button(ftlog, text="取 消", command=lambda:
login_teacher_window.destroy())
login_teacher_no.pack(side='left',padx=20)

login_teacher_window.mainloop()

"""管理员登陆"""
def login_admin():
    global login_admin_window
    login_admin_window=tk.Tk()
    login_admin_window.title("管理员登录")
    login_admin_window.iconbitmap("output_icon.ico")

    location(login_admin_window,300,150)
```




```
faccount=tk.Frame(login_admin_window)
faccount.pack(pady=10)
fpassword = tk.Frame(login_admin_window)
fpassword.pack(pady=10)
falog = tk.Frame(login_admin_window)
falog.pack(pady=10)

laccount = tk.Label(faccount, text="账号: ")
laccount.pack(side="left")
global name
name = tk.Entry(faccount)
name.pack()

lpassword = tk.Label(fpassword, text="密码: ")
lpassword.pack(side="left")
global no
no = tk.Entry(fpassword)
no.pack()

login_admin = tk.Button(falog, text="登 录", command=lambda :
auto_login(login_admin_window))
login_admin.pack(side='left', padx=20)
login_admin_no = tk.Button(falog, text="取 消", command=lambda:
login_admin_window.destroy())
login_admin_no.pack(side='left', padx=20)

login_admin_window.mainloop()

"""创建登录窗口"""
def login_global():
    global login_global_window
    login_global_window = tk.Tk()
    login_global_window.title("登陆页面")
    login_global_window.iconbitmap("output_icon.ico")

    location(login_global_window, 1200, 800)

    image = Image.open("微信图片_20241124172307.jpg")
    library_photo = ImageTk.PhotoImage(image)
    canvas = tk.Canvas(login_global_window, width=library_photo.width(),
height=library_photo.height())
    canvas.pack(fill="both", expand=True)
    canvas.create_image(0, 0, image=library_photo, anchor="nw")
```



```
login_student_button = tk.Frame(canvas)
frame_window1 = canvas.create_window(100,50,window=login_student_button)
login_student_button.pack(side='left',padx=100)
large_font = font.Font(size=18)
button_student = tk.Button(login_student_button, text=" 学生登录 ", font = large_font,
padx=30, pady=10,bg='white',
                                command=login_student)
button_student.pack()

login_teacher_button = tk.Frame(canvas)
frame_window2 = canvas.create_window(100, 50, window=login_teacher_button)
login_teacher_button.pack(side='left',padx=100)
large_font = font.Font(size=18)
button_teacher = tk.Button(login_teacher_button, text=" 教师登录 ", font=large_font,
padx=30, pady=10,bg='white',
                                command=login_teacher)
button_teacher.pack()

login_admin_button = tk.Frame(canvas)
frame_window3 = canvas.create_window(100, 50, window=login_admin_button)
login_admin_button.pack(side='left',padx=100)
large_font = font.Font(size=18)
button_admin = tk.Button(login_admin_button, text=" 管理员登录 ", font=large_font,
padx=30,pady=10,bg='white',
                                command=login_admin)
button_admin.pack()

login_global_window.mainloop()

if __name__ == '__main__':
    login_global()
```