

实验五 数据库程序设计

章启航 3170104343

求是科学班（计算机）1701

实验目的

1. 设计并实现一个精简的图书管理系统，具有入库、查询、借书、还书、借书证管理 等基本功能。
2. 通过本次设计来加深对数据库的了解和使用，同时提高自身的系统编程能力。

实验平台

1. 开发工具：
 - 后端 `python` `FLASK` 框架
 - 前端 `html`, `css`
2. 数据库平台：
 - `PHPMyAdmin` `MySql`

总体设计

1. 实验基本要求

| 模块名称 | 功能描述 |
|-------|--|
| 管理员登陆 | 根据管理员 ID 和密登录系统 |
| 图书入库 | 1. 单本入库，直接从程序界面上输入。 2. 批量入库，从文本文件中批量导入图书数据。 |
| 图书查询 | 1. 按书的类别、书名、出版社、年份、作者、价格进行查询。 2. 可以点击标题来对相应的字段进行排序。 |
| 借书 | 1. 输入借书证卡号，自动显示该借书证所有已借书籍。 2. 输入书号，若该书还有库存，则提示借书成功，库存数减一。 |
| 还书 | 1. 输入借书证卡号，自动显示该借书证所有已借书籍。 2. 输入书号，若如果该书被借，则还书成功，同时库存加一。 |

2. 设计思路

- 基本思想
 - 为了尽量符合真实的使用场景，我将管理员与用户的功能区分开来。管理员拥有管理图书入库出库，增加删除借书卡的功能。普通用户拥有借书、还书、查看已经借阅图书的功能。
 - 增加借书卡的机制，用户可以随意注册帐号，但是想要借书的话必须要拥有借书卡，新用户的要求管理员为其添加借书卡。
- 细节修改
 - 相较于实验基本要求，我在用户借书之前加入了查询功能。因为用户在借书之前可能并不知道自己想要什么图书，也不知道库中有什么书，所以我增加了相应功能，用户可以按照图书类型关键词来查询借阅。

- 用户如果没有还清书，无法注销借书卡。

3. 数据库表设计

- 因为FLASK框架中的数据库采用了ORM(Object Relation Model)模式，数据库中的表在后端中采用对象的形式管理，所以接下来我将按照类来介绍数据库表的设计。
- 后端一共包含了4个类：User, Book, Card, Borrow。

| 类 | 属性与方法 |
|--------|--|
| User | 属性： <u>id</u> , username, email, is_admin, password_hash 方法：set_password, checkpassword |
| Book | 属性： <u>id</u> , bookname, type, author, remain, year |
| Card | 属性： <u>id</u> , (User) user_id, borrow_num |
| Borrow | 属性： <u>id</u> , (User) user_id, (Book) book_id |

- 这四个类相应地对应了四张表格。

4. 具体效果及实现效果

- 如何与数据库进行交互
 - 我采用的是MySQL数据库。使用flask的扩展包 flask_sqlalchemy可以对其进行访问。
 - 在配置文件config.py中加入

```
SQLALCHEMY_DATABASE_URI =
'mysql+pymysql://**:*@localhost/csdb'
SQLALCHEMY_TRACK_MODIFICATIONS = False
```

- 用户登陆

- 管理员和用户都得事先登陆。
- 界面展示

Library Home Login

Sign In

Username

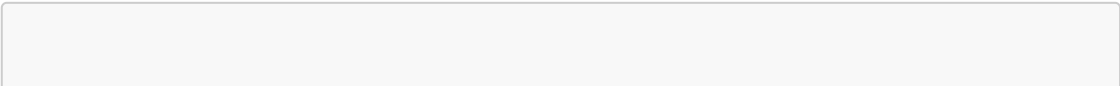
Password

☐ Remember Me

Sign In

New User? [Click to Register!](#)

- 后端接收前端表单返回的数据，在数据库中进行查询，如果查询成功且密码正确，则登陆成功，跳转到用户（或管理员）界面，如果登陆失败，则会停留在此页面，刷新出一条登陆失败消息。

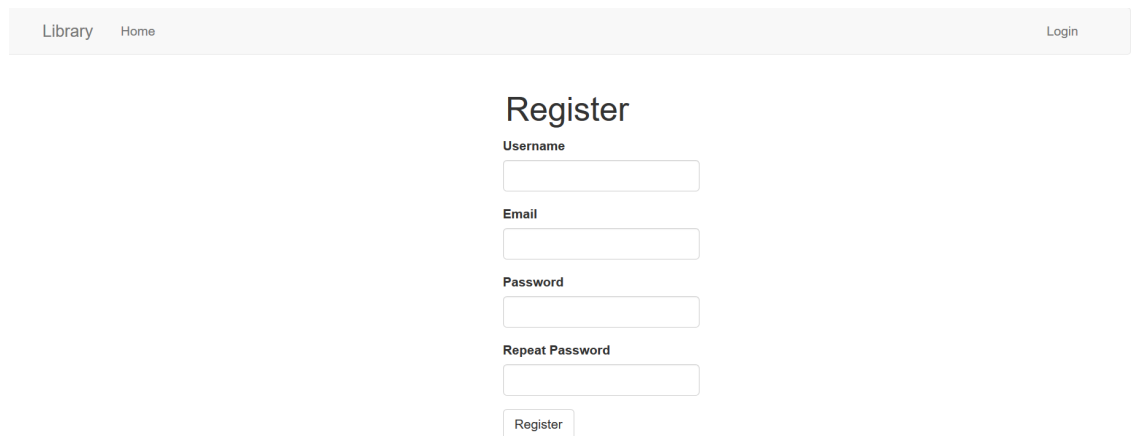


```
form = LoginForm()
if form.validate_on_submit():
    user =
    User.query.filter_by(username=form.username.data).first()
    if user is None or not
    user.check_password(form.password.data):
        flash('Invalid username or password')
        return redirect(url_for('login'))
    login_user(user, remember=form.remember_me.data)
    return redirect(url_for('index'))
```

- 登陆界面底部也包含一个新用户注册链接。

- 用户注册

- 可以在register.html中注册普通用户。
- 界面展示



Library Home Login

Register

Username

Email

Password

Repeat Password

- 后端接受到表单数据，像数据库执行插入语句。

```
if form.validate_on_submit():
    user = User(username=form.username.data, email =
form.email.data)
    user.set_password(form.password.data)
    db.session.add(user)
    db.session.commit()
    flash('Congratulations, you are now a registered user!')
    return redirect(url_for('login'))
```

- 因为User的username, email字段要求Unique，所以表单会自动拒绝跟数据库中重复用户名或邮箱的注册申请。

Register

Username

z

Please use a different username.

Email

1@qq.com

Please use a different email address.

- 管理员界面

- 管理员初始界面： 可以选择执行相应的四种功能

Hi, Admin admin !

Add Book

Delete Book

Add Card

Delete Card

- 图书入库界面： 填写表单向数据库添加图书

Add book

Book's name

Book's type

Author

Public Year

Amount

Execute

如果数据库中存在与当前添加的图书书名相同的图书，则只需要修改相应图书的remain(余额)，否则执行insert。

```
form = AddBookForm()
if form.validate_on_submit():
    find_book = db.session.query(Book).filter_by(bookname =
form.bookname.data).first()
    if find_book is None:
        book = Book(bookname = form.bookname.data, remain =
form.amount.data, author = form.author.data, type =
form.type.data, year = form.year.data)
        db.session.add(book)
    else:
        find_book.remain += form.amount.data
    db.session.commit()
    flash('Succeed to add books to the library!')
```