

数据库系统实验报告（五）

课程名称： 数据库系统原理 实验项目名称： 数据库程序设计

学生姓名： 刘轩铭 专业： 软件工程 学号： 3180106071

指导老师： 周波 实验日期： 2020 年 4 月 24 日

一、实验目的和要求

1. 掌握数据库应用开发程序设计方法

二、实验内容和要求

1. 设计简单的图书管理数据库概念模式。
2. 设计相应的关系模式。
3. 实现一个图书管理程序，实现图书、借书证及图书借阅的管理的基本功能。

三、主要仪器设备

1. 操作系统： Windows
2. 数据库管理系统： SQL Server 或 MySQL （本次实验选用 MySQL）
3. 实现框架： Django

四、操作方法与实验步骤

4.1 确定图书馆的组成等要素

1. 图书馆有各种图书，数据规模庞大。
2. 每本图书都有书名、编号、作者、出版社、价格、藏书量、库存（库存小于藏书量）等信息。
3. 用户可凭借书号和密码登陆系统查询图书和查询自己已借的书。借书证上的信息包括卡号、姓名、单位和类别。借书最长时间为 30 天。
4. 图书管理员可以为借书证借书和还书。借书最长时间为 30 天。
5. 借书记录会记录借书证、所借书籍的书号、借书日期、借书时间和经手人。

4.2 建立数据库概念模式

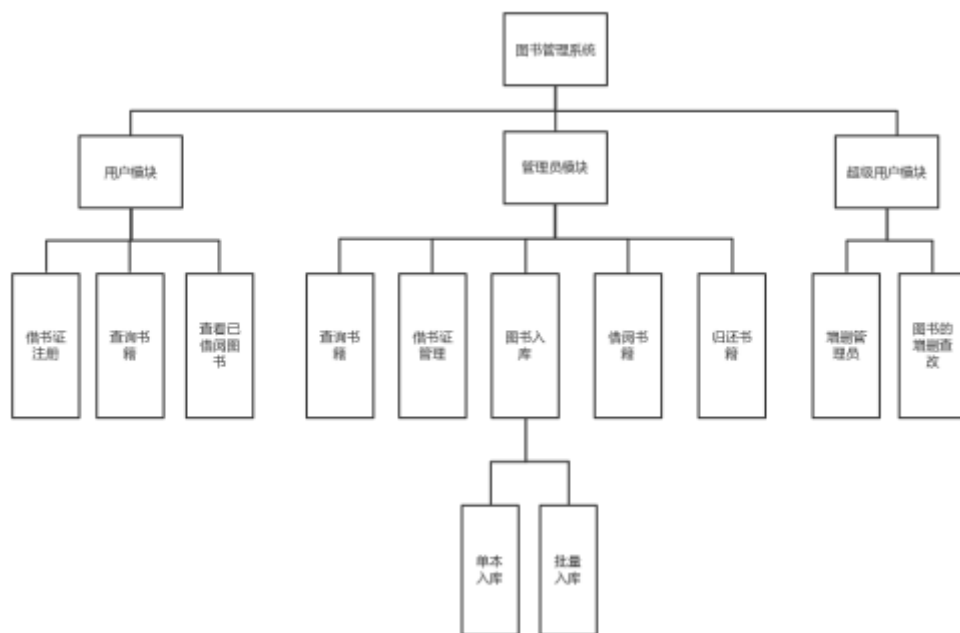
对象名称	对应属性
书	书号, 类别, 书名, 出版社, 年份, 作者, 总藏书量, 库存
借书证	卡号, 密码, 姓名, 单位, 类别(教师或学生)
管理员	管理员 ID, 密码, 姓名
借书记录	记录号, 书号, 卡号, 借期, 还期, 管理员 ID

4.3 建立数据库关系模式

```
book( book_ID, type, book_name, publisher, year, author, total_number, real_number)
card( ID, password, name, dept_name, type)
admin( admin_ID, password, name)
record( record_ID, ID, book_ID, bor_date, ret_date, admin_ID)
```

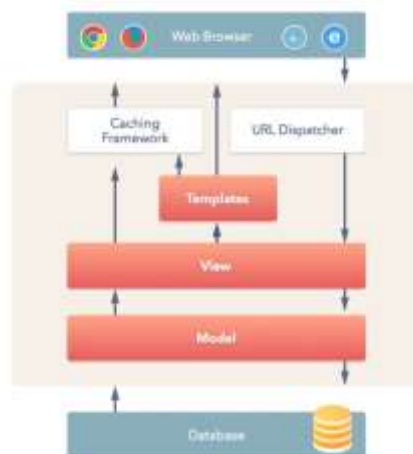
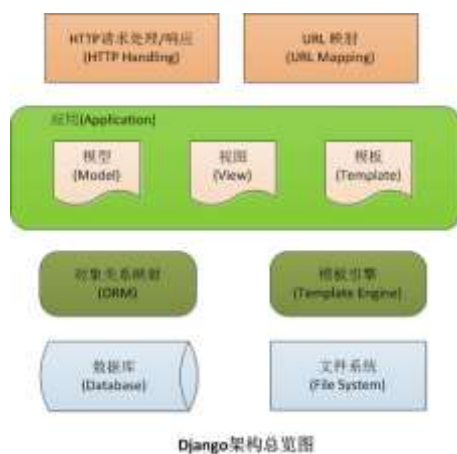
4.4 图书馆管理系统功能设计

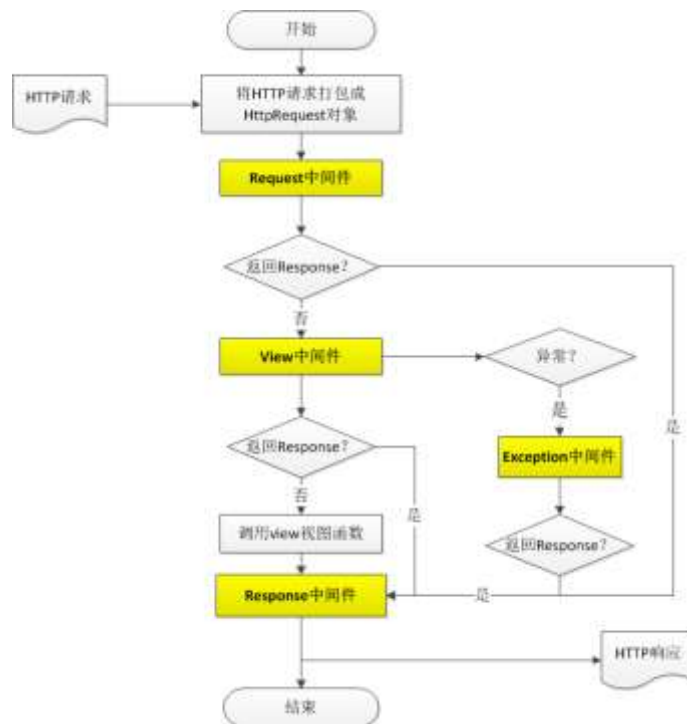
模块名称	功能描述
用户登陆	输入用户 ID, 密码; 登入系统的查询界面 或返回密码错误.
管理员登陆	输入管理员 ID, 密码; 登入系统或返回 ID/密码错误.
图书入库	1. 单本入库 2. 批量入库
图书查询	要求可以对书的类别, 书名, 出版社, 年份(年份区间), 作者, 进行查询。 可选要求: 可以按用户指定属性对图书信息进行排序。
借书	1. 输入借书证卡号, 显示该借书证所有已借书籍 2. 输入书号(如果该书还有库存, 则借书成功, 同时库存数减一。否则输出该书无库存, 且输出最近归还的时间。)
还书	1. 输入借书证卡号, 显示该借书证所有已借书籍 2. 输入书号(如果该书在已借书籍列表内, 则还书成功, 同时库存加一。否则输出出错信息。)
借书证管理	增加或删除一个借书证.



4.5 开始搭建数据库

按照 Django 的技术框架有：





其技术流程为：

- 1) 用户通过浏览器请求一个页面
- 2) 请求到达 Request Middlewares，中间件对 request 做一些预处理或者直接 response 请求
- 3) URLConf 通过 urls.py 文件和请求的 URL 找到相应的 View
- 4) View Middlewares 被访问，它同样可以对 request 做一些处理或者直接返回 response
- 5) 调用 View 中的函数
- 6) View 中的方法可以选择性的通过 Models 访问底层的数据
- 7) 所有的 Model-to-DB 的交互都是通过 manager 完成的
- 8) 如果需要，Views 可以使用一个特殊的 Context
- 9) Context 被传给 Template 用来生成页面
- 10) Template 使用 Filters 和 Tags 去渲染输出
- 11) 输出被返回到 View
- 12) HTTPResponse 被发送到 Response Middlewares
- 13) 任何 Response Middlewares 都可以丰富 response 或者返回一个完全不同的 response

14)Response 返回到浏览器，呈现给用户

了解清楚了基本的数据框架，我们着手建立数据库：

为实现登陆查询借书还书入库等功能，数据库中存储了四张数据表：Book，Card，Borrow 和 Manager。

创建模型：

在 MySite/models.py 文件中建立一个模型，内容如下：

```
from django.db import models
# Create your models here.
class Book(models.Model):
    bno = models.CharField(primary_key=True, max_length=60)
    category = models.CharField(max_length=60)
    title = models.CharField(max_length=60)
    press = models.CharField(max_length=60)
    year = models.IntegerField()
    author = models.CharField(max_length=60)
    price = models.DecimalField(max_digits=15, decimal_places=2)
    total = models.IntegerField()
    stock = models.IntegerField()

class Card(models.Model):
    cno = models.CharField(primary_key=True, max_length=60)
    cname = models.CharField(max_length=60)
    department = models.CharField(max_length=60)
    types = models.CharField(max_length=60)
    cpassword = models.CharField(max_length=60)
```

```

class Manager(models.Model):
    mno = models.CharField(primary_key=True, max_length=60)
    mpassword = models.CharField(max_length=60)
    mname = models.CharField(max_length=60)
    mphone = models.CharField(max_length=60)

class Borrow(models.Model):
    cno = models.CharField(max_length=60)
    bno = models.CharField(max_length=60)
    borrow_date = models.CharField(max_length=60)
    return_date = models.CharField(max_length=60)
    mno = models.CharField(max_length=60)

```

建立数据表：

a. 创建映射文件

```
python manage.py makemigrations
```

b. 将映射文件中的映射到数据库中

```
python manage.py migrate
```

至此，Django 已经连接到数据库。

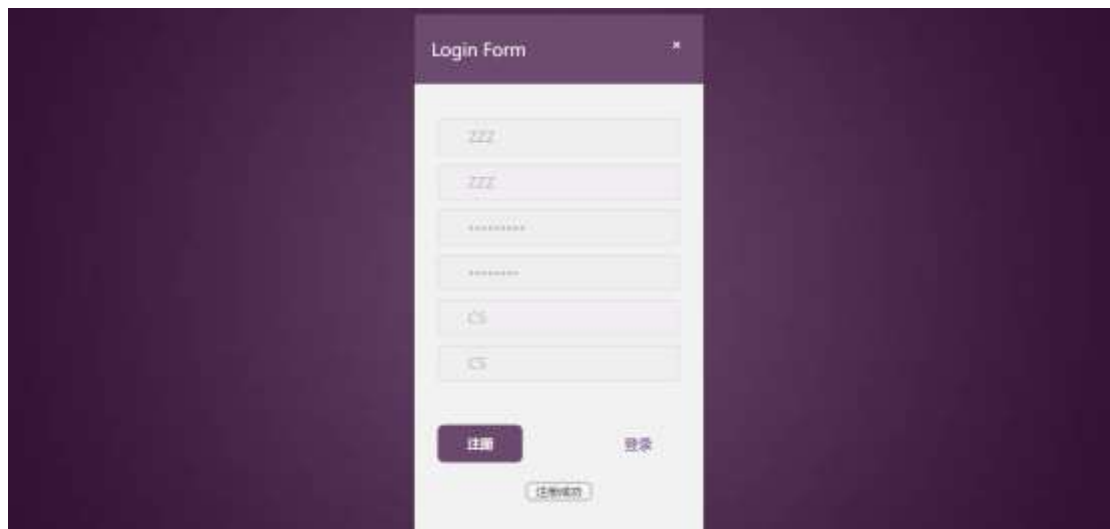
4.6 编写 html 和 css 文件，建立各个网页

一个视图函数，简称视图，是一个简单的 Python 函数。它接受 Web 请求并且返回 Web 响应。响应可以是一张网页的 HTML 内容，一个重定向，一个 404 错误，一个 XML 文档，或者一张图片... 任何东西都可以。无论视图本身包含什么逻辑，都要返回响应。

在项目之中，URL 定向到一个 view 函数，并且用 HttpRequest 传入 HTML 界面里表单的内容，在 view 中用 request.get 接收 HTML 表单里传进来的值，然后是一系列的操作，涉及到对数据库的访问。也就是说，view 对应的是一个网页。我们只需要编写这些文件即可。

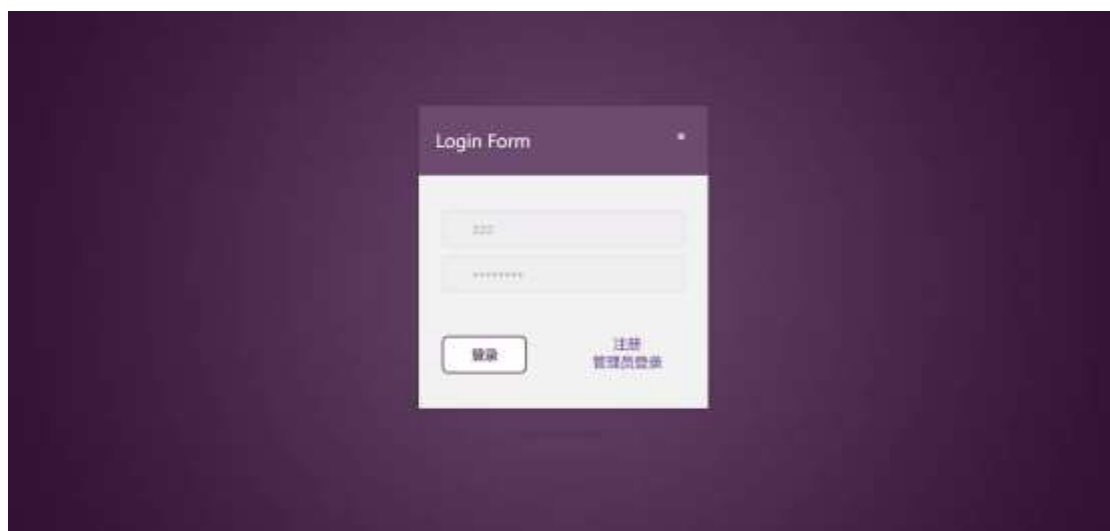
五、实验结果与分析

5.1 用户注册



A screenshot of a user registration form titled "Login Form" (though it contains registration fields). The form is centered on a dark purple background. It features several input fields: a text field with "123" (username), a text field with "123" (password), a masked password field with "*****", another masked password field with "*****", a text field with "CS" (email), and another text field with "CS" (phone number). At the bottom, there are two buttons: a dark purple "注册" (Register) button and a light purple "登录" (Login) button. Below these buttons is a small, faint "注册成功" (Registration Successful) message.

5.2 用户登录



A screenshot of a user login form titled "Login Form". The form is centered on a dark purple background. It features two input fields: a text field with "123" (username) and a masked password field with "*****". Below these fields are two buttons: a light purple "登录" (Login) button and a dark purple "注册" (Register) button. To the right of the "注册" button, there is a link that says "注册" and "管理员登录" (Admin Login).

5.3 图书分类查询

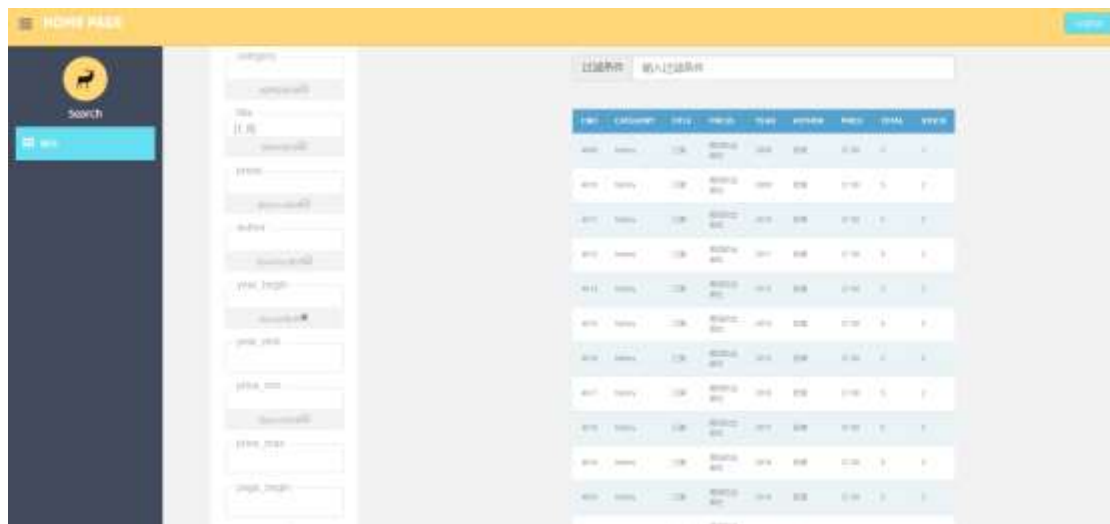


5.4 管理员登录



5.5 图书入库

[illegible]



5.6 图书借阅

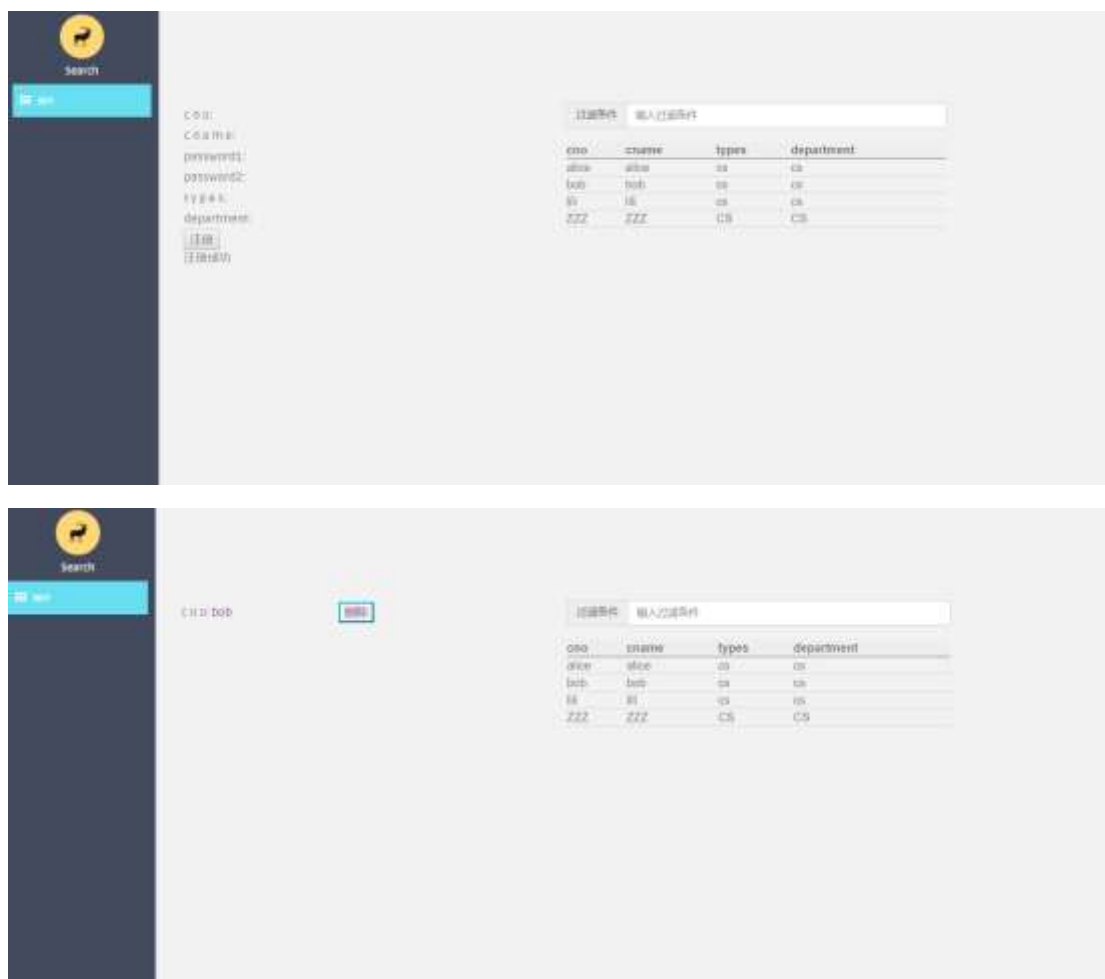




5.6 图书归还



5.7 增删用户



六、讨论与心得

这一次大作业写了很长时间，由于之前有学习过 Django 的使用，觉得这是个不错的框架，于是决定使用 Django 框架进行开发图书管理系统，python 开发更加简单，而且有很好的安全性。

在此处大作业过程之中，遇到很多难题。在 W3school 上学习系统 Django，模型和模板等概念困扰了我很久。慢慢了解了机制。之后再遇到问题，查阅官方文档时，也更容易看明白。

比如 Django 中加载 CSS 文件的静态路由和 PHP 中的设置方法并不相同，花费了半天的时间学习 Django 的路由设置。一个人的思路可能会狭窄，在期间多次请问同学，使得问题更容易得到解决。感受到技术还是需要多交流，才能互相促进。

在系统的安全性上面考虑的不多，由于到网络的传输机制了解的不够深入，

所以在初期做的时候网站存在很多漏洞。以后在深入学习网络之后，可以对网站进行优化。

最大的感受，累却很有成就感。在此处学习过程中学到很多有用的知识，对于数据库的一些深入知识，仍需要加强运用和学习。