

# 浙 江 大 学



## 《数据库系统原理》实验报告 (实验名称：实验5 图书管理系统)

姓    名：郭\*\*  
学    号：\*\*\*\*\*  
专    业：海洋工程与技术  
班    级：海工\*\*\*  
指导教师：林\*\*  
日    期：2022.5.27

# 图书管理系统

- 1 实验目的
- 2 实验内容
- 3 实验环境
- 4 图书管理系统说明展示

## 4.1 系统架构的描述

- 4.1.1 E-R图

- 4.1.2 模式图

## 4.2 数据库表结构设计

- 4.2.1 book

- 4.2.2 card

- 4.2.3 admin

- 4.2.4 record

## 4.3 各项技术

- 4.3.1 `PyMySql`

- 4.3.2 `Django`

- 4.3.3 `Numpy`

- 4.3.4 `Pandas`

## 4.4 模块的详细设计

- 4.4.1 总体流程图

- 4.4.2 模块功能及实现技术

- 4.4.2.1 检查登录认证

- 4.4.2.1.1 验证码

- 4.4.2.1.2 登录验证

- 4.4.2.1.3 保持登陆状态

- 4.4.2.1.4 注销

- 4.4.2.2 图书管理

- 4.4.2.2.1 添加图书

- 4.4.2.2.2 编辑图书

- 4.4.2.2.3 删除图书

- 4.4.2.2.4 借阅图书

- 4.4.2.2.5 查询图书

- 4.4.2.2.6 从excel批量导入图书

- 4.4.2.2.7 随机上架5本图书

- 4.4.2.2.8 清空图书馆

- 4.4.2.3 管理员管理

- 4.4.2.3.1 添加管理员

- 4.4.2.3.2 编辑管理员

- 4.4.2.3.3 删除管理员

- 4.4.2.4 借书证管理

- 4.4.2.4.1 添加借书证

- 4.4.2.4.2 查询借书记录

- 4.4.2.4.3 编辑借书证

- 4.4.2.4.4 删除借书证

- 4.4.2.5 书籍借阅

- 4.4.2.6 书籍归还

## 4.5 文件结构

**5** 要求之外的设计/功能

**6** 实验总结

  6.1 问题及解决过程

  6.2 错误及原因分析

  6.3 体会和收获

**7** 参考资料

## 1 实验目的

通过该图书馆系统的设计与实现，提高学生的系统编程能力，加深对数据库系统原理及应用的理解。

## 2 实验内容

设计并实现一个精简的[图书管理系统](#)，要求具有图书入库、查询、借书、还书、借书证管理等功能。

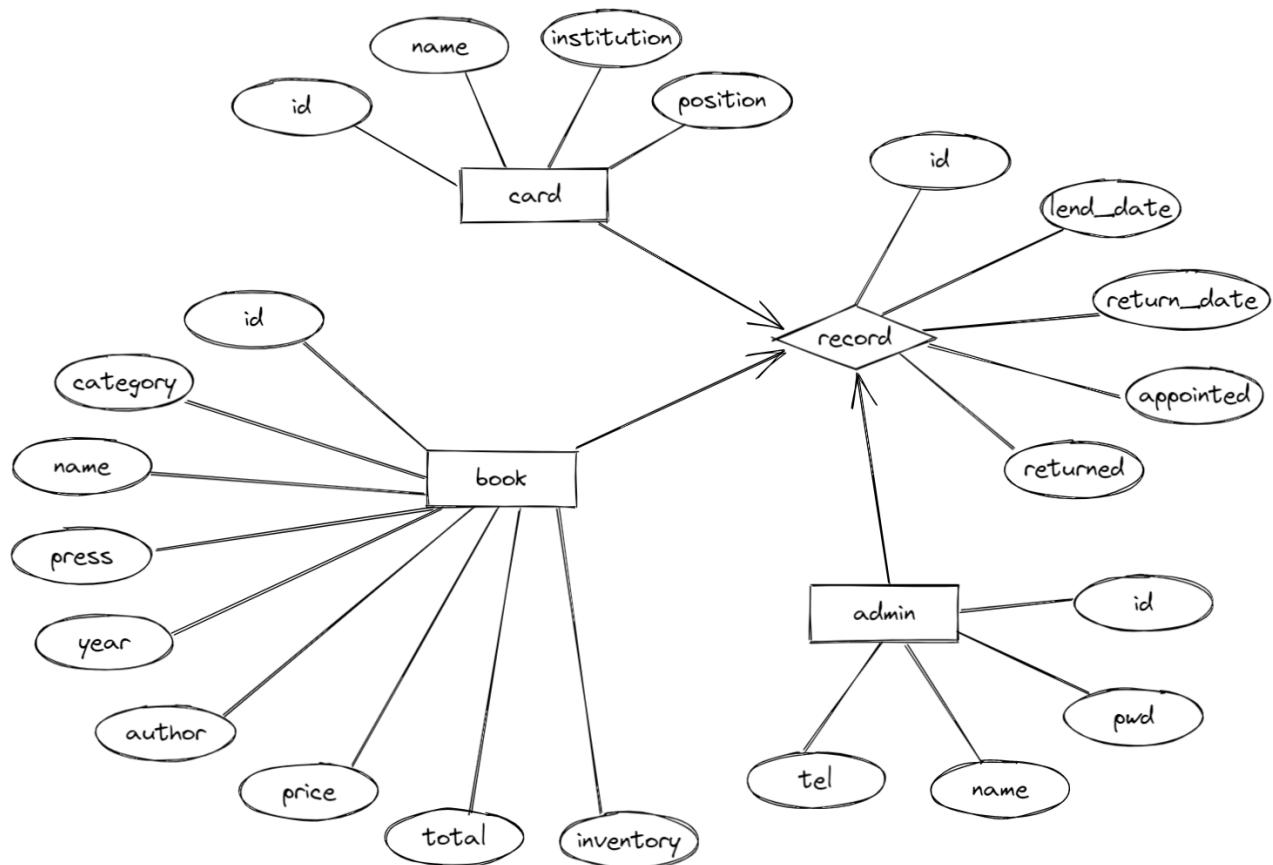
## 3 实验环境

- 数据库: [MySQL 5.7](#)
- 开发工具: [python 3.7](#)
  - [mysqlclient](#)
  - [pillow](#)
  - [pandas](#)
  - [openpyxl](#)
- 后端: [Django 3.2](#)
- 用户界面: 网页

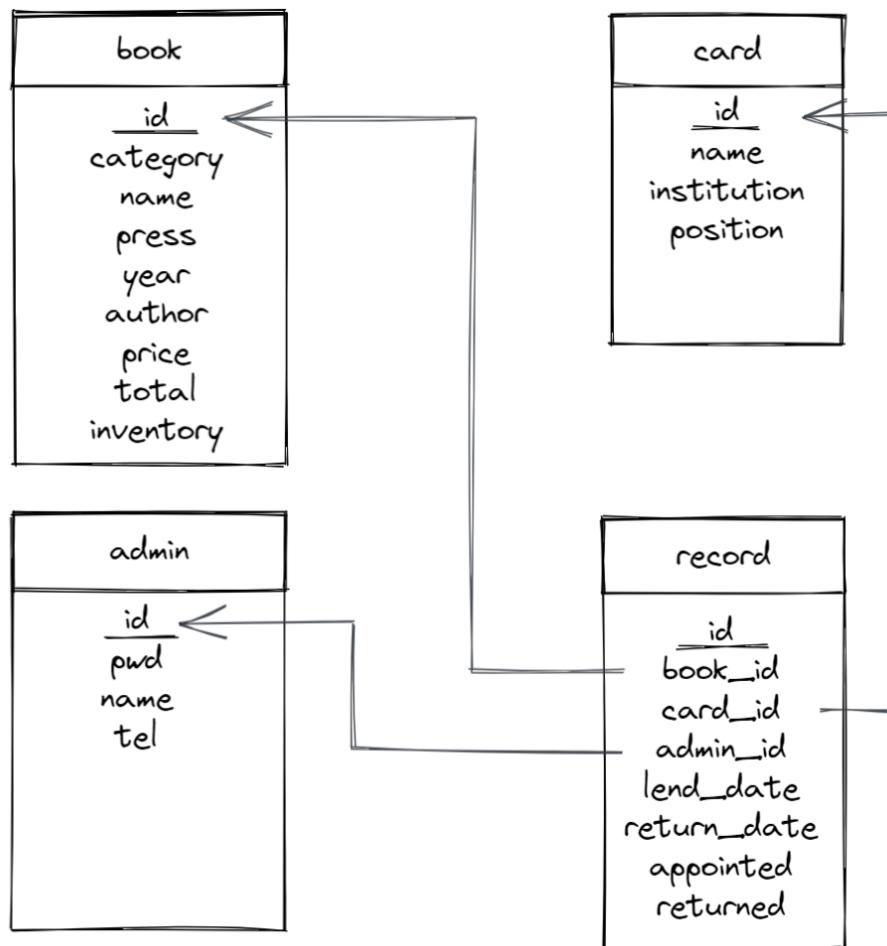
## 4 图书管理系统说明展示

### 4.1 系统架构的描述

#### 4.1.1 E-R图



#### 4.1.2 模式图



## 4.2 数据库表结构设计

### 4.2.1 book

| 字段        | 描述   | 数据             | 约束                                |
|-----------|------|----------------|-----------------------------------|
| id        | id   | bigint(20)     | primary, not null, auto_increment |
| category  | 类别   | smallint(6)    | not null                          |
| name      | 书名   | varchar(64)    | not null                          |
| press     | 出版社  | varchar(32)    | not null                          |
| year      | 出版年份 | int(11)        | not null                          |
| author    | 作者   | varchar(32)    | not null                          |
| price     | 价格   | decimal(10, 2) | not null                          |
| total     | 总藏书量 | int(11)        | not null                          |
| inventory | 当前库存 | int(11)        | not null                          |

#### 4.2.2 card

| 字段          | 描述    | 数据          | 约束                                |
|-------------|-------|-------------|-----------------------------------|
| id          | id    | bigint(20)  | primary, not null, auto_increment |
| name        | 持卡人姓名 | varchar(32) | not null                          |
| institution | 机构    | varchar(32) | not null                          |
| position    | 身份    | smallint(6) | not null                          |

#### 4.2.3 admin

| 字段   | 描述 | 数据          | 约束                                |
|------|----|-------------|-----------------------------------|
| id   | id | bigint(20)  | primary, not null, auto_increment |
| pwd  | 密码 | varchar(64) | not null                          |
| name | 姓名 | varchar(32) | not null                          |
| tel  | 电话 | varchar(11) | not null                          |

#### 4.2.4 record

| 字段          | 描述    | 数据         | 约束                                | 外键       |
|-------------|-------|------------|-----------------------------------|----------|
| id          | id    | bigint(20) | primary, not null, auto_increment |          |
| lend_date   | 借书日期  | date       | not null                          |          |
| return_date | 还书日期  | date       | not null                          |          |
| appointed   | 被预约   | tinyint(1) | not null                          |          |
| admin_id_id | 管理员id | bigint(20) | default null                      | admin.id |
| book_id_id  | 书籍id  | bigint(20) | default null                      | book.id  |
| card_id_id  | 借书证id | bigint(20) | default null                      | card.id  |
| returned    | 已归还   | tinyint(1) | not null                          |          |

## 4.3 各项技术

### 4.3.1 PyMySQL

PyMySQL 是在 Python3.x 版本中用于连接 MySQL 服务器的一个库。

### 4.3.2 Django

Django 是一个由 Python 编写的一个开放源代码的 Web 应用框架。

使用 Django，可以轻松地完成一个正式网站所需要的大部分内容，并进一步开发出全功能的 Web 服务。

Django 本身基于 MVC 模型，即 Model (模型) + View (视图) + Controller (控制器) 设计模式，MVC 模式使后续对程序的修改和扩展简化，并且使程序某一部分的重复利用成为可能。

### 4.3.3 Numpy

NumPy (Numerical Python) 是 Python 语言的一个扩展程序库，支持大量的维度数组与矩阵运算，此外也针对数组运算提供大量的数学函数库。

NumPy 是一个运行速度非常快的数学库，主要用于数组计算。

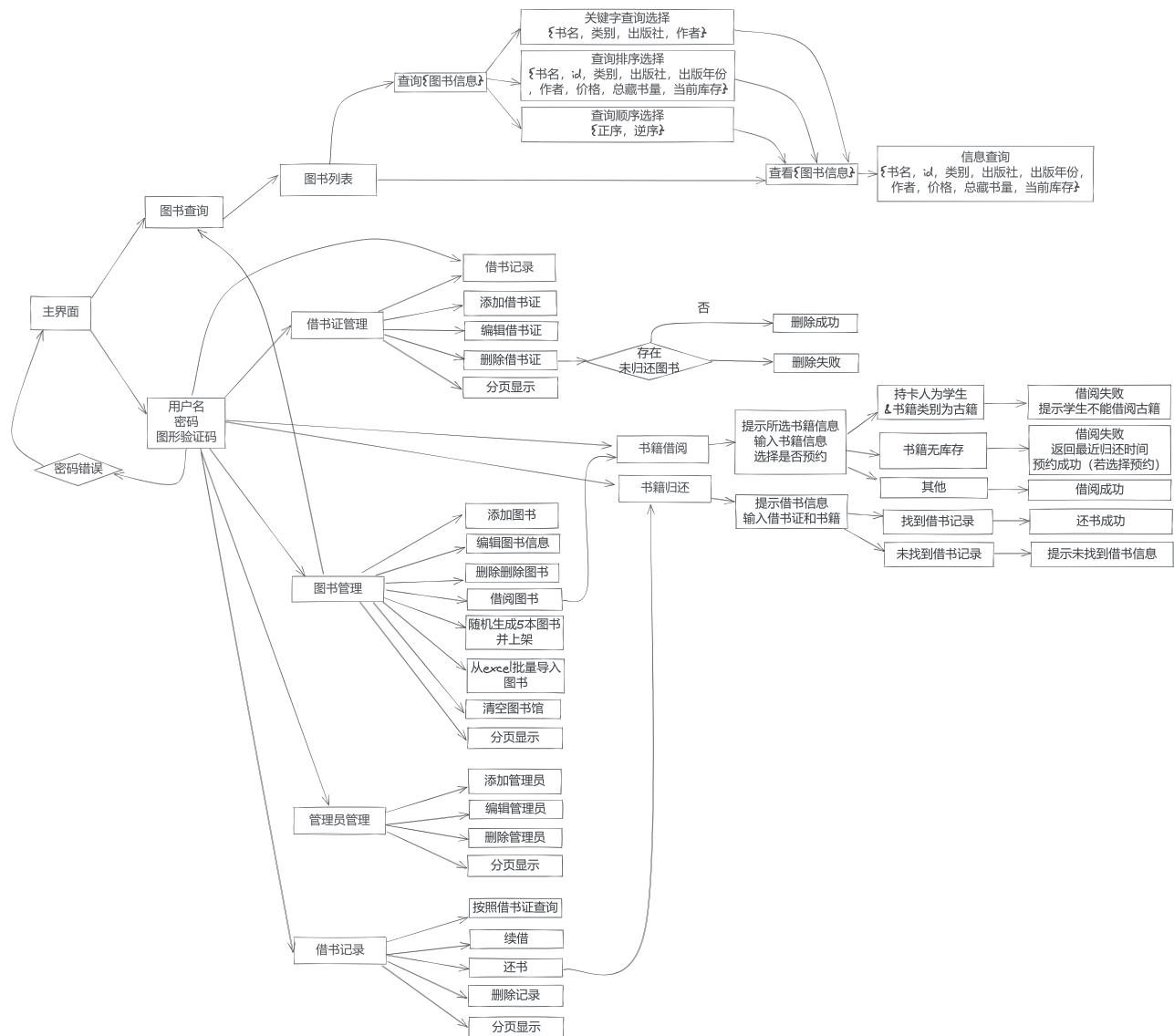
### 4.3.4 Pandas

Pandas 是 Python 的一个扩展程序库，用于数据分析，提供高性能、易于使用的数据结构和数据分析工具。

Pandas 可以从各种文件格式比如 Excel 导入数据，并对各种数据进行运算操作。

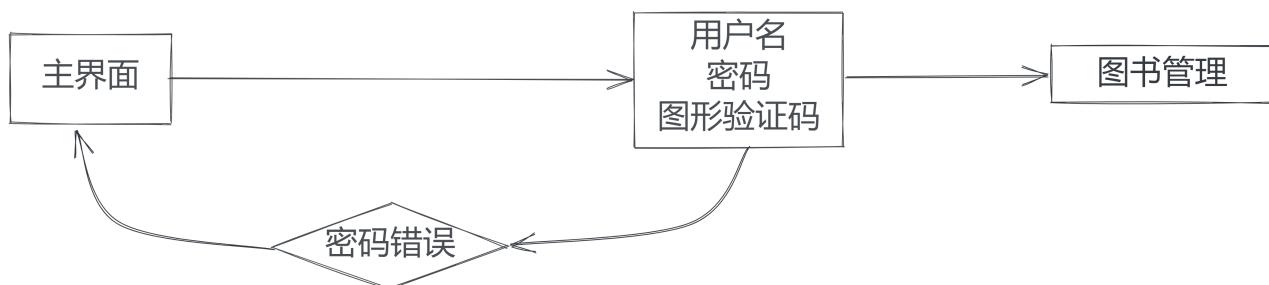
## 4.4 模块的详细设计

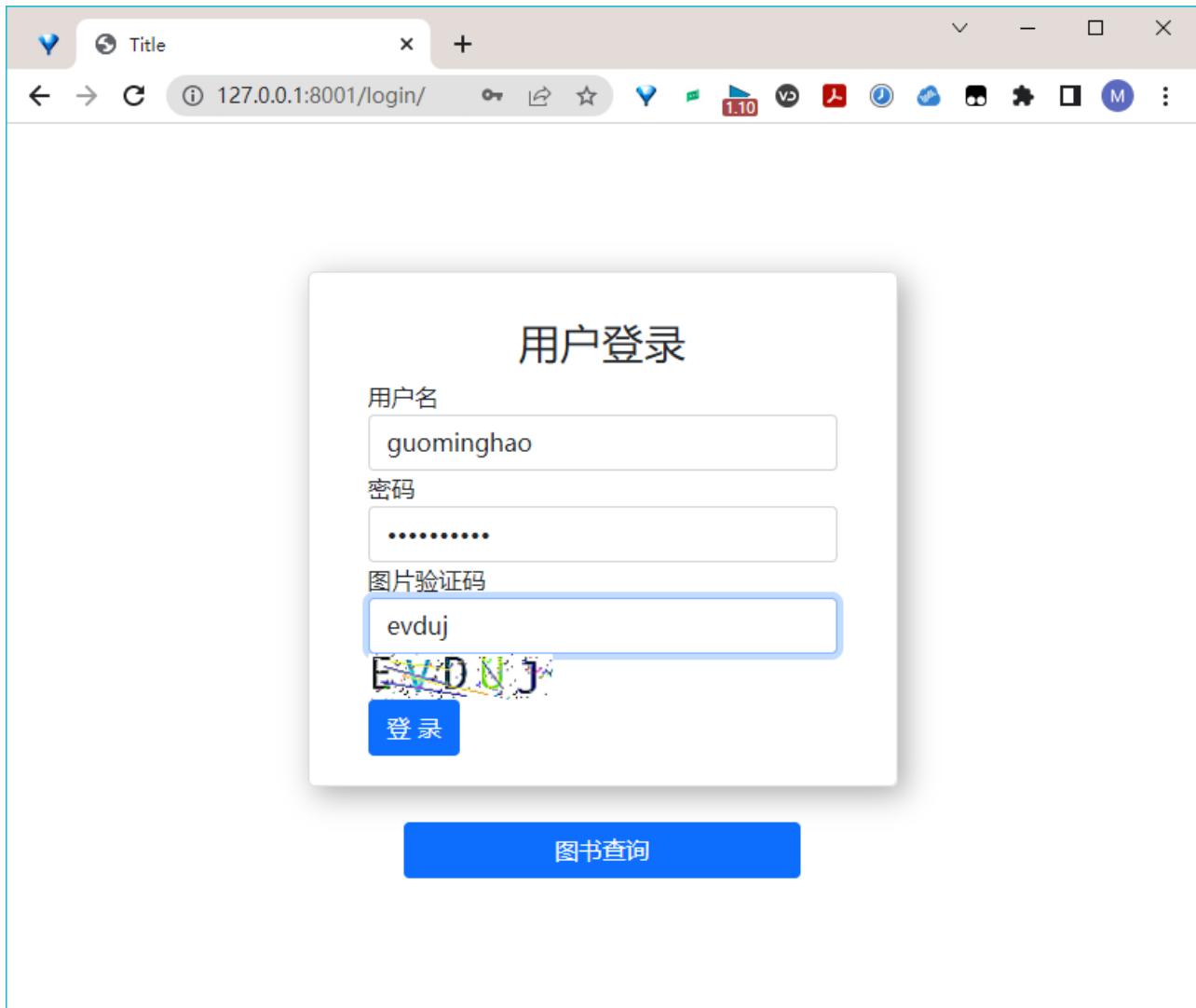
### 4.4.1 总体流程图



### 4.4.2 模块功能及实现技术

#### 4.4.2.1 检查登录认证





在中间件 `middleware` 中添加 `auth.py` 检查登录认证；

对于特定页面，不进行登录认证。

```
1 class AuthMiddleware(MiddlewareMixin):  
2  
3     def process_request(self, request):  
4         # 0.排除那些不需要登录就能访问的页面  
5         #    request.path_info 获取当前用户请求的URL /login/  
6  
7         if request.path_info in ["/login/",  
8             "/image/code/",  
9             "/book/list/",]:  
10            return  
11  
12         # 1.读取当前访问的用户的session信息，如果能读到，说明已登陆过，就可以继续向后走。  
13         info_dict = request.session.get("info")  
14         if info_dict:  
15             return  
16  
17         # 2.没有登录过，重新回到登录页面  
18         return redirect('/login/')
```

#### 4.4.2.1.1 验证码

```
1 def login(request):
2
3     ... # 跳转登陆页面
4
5     form = LoginForm(data=request.POST)
6     if form.is_valid():
7         # 验证码的校验
8         user_input_code = form.cleaned_data.pop('code')
9         code = request.session.get('image_code', "")
10        if code.upper() != user_input_code.upper():
11            form.add_error("code", "验证码错误")
12        return render(request, 'login.html', {'form': form})
13
14    ... # 用户名密码校验
15
16    ... # 保存登录信息
17
18    ...
```

#### 4.4.2.1.2 登录验证

```
1 def login(request):
2
3     ... # 跳转登陆页面
4
5     form = LoginForm(data=request.POST)
6     if form.is_valid():
7         ... # 验证码的校验
8
9     # 去数据库校验用户名和密码是否正确，获取用户对象、None
10    admin_object = models.Admin.objects.filter(**form.cleaned_data).first()
11    if not admin_object:
12        form.add_error("pwd", "用户名或密码错误")
13        return render(request, 'login.html', {'form': form})
14
15    ... # 保存登录信息
16
17    ...
```

#### 4.4.2.1.3 保持登陆状态

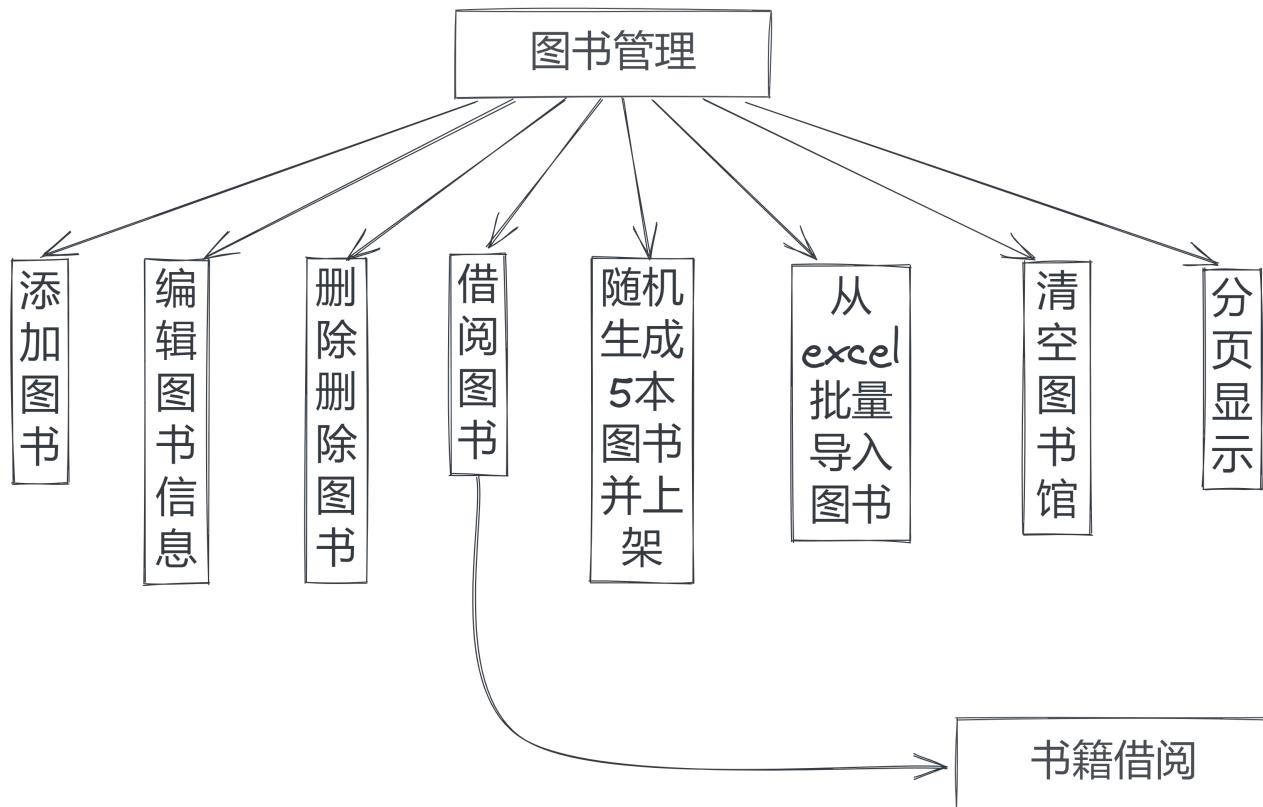
```
1 def login(request):
2
3     ... # 跳转登陆页面
4
5     form = LoginForm(data=request.POST)
6     if form.is_valid():
7         ... # 验证码的校验
8
9         ... # 用户名密码校验
10
11        # 用户名和密码正确
12        # 网站生成随机字符串；写到用户浏览器的cookie中；在写入到session中；
13        request.session["info"] = {'id': admin_object.id, 'name': admin_object.name}
14        # session可以保存7天
15        request.session.set_expiry(60 * 60 * 24 * 7)
16
17    ...
```

#### 4.4.2.1.4 注销

清除数据库中保存的 `session`

```
1 def logout(request):
2     """注销"""
3
4     request.session.clear()
5     return redirect('/login/')
```

#### 4.4.2.2 图书管理



Screenshot of the 'Book Management' system interface:

- Header:** 浙江大学图书馆 借书证 图书管理 管理员管理 借书记录 书籍借阅 书籍归还 guominghao 注销
- Toolbar:** 添加图书 清空图书馆 随机上架5本图书 从excel批量导入图书
- Search and Filter:** 排序依据(书名)、排序顺序(正序)、查询关键字(书名); 排序依据(书名)、排序顺序(正序)、查询关键字(价格)
- Table:** Displays a list of books with columns: id, 类别, 书名, 出版社, 出版年份, 作者, 价格, 总藏书量, 当前库存, 操作 (Edit, Delete, Borrow).
- Data:** Sample data rows (部分数据行):
 

| ID  | 类别        | 书名       | 出版社      | 出版年份 | 作者       | 价格    | 总藏书量 | 当前库存 | 操作  |
|-----|-----------|----------|----------|------|----------|-------|------|------|---|
| 249 | 医药、卫生     | mvkrj    | zmxn     | 1596 | zbwepof  | 20.22 | 59   | 57   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 237 | 文学        | mxhb     | bup      | 1357 | adzxhbvl | 15.83 | 6    | 2    | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 252 | 自然科学      | nfpkgw   | r        | 1717 | nkqrmo   | 78.56 | 26   | 17   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 220 | 社会科学      | N号房追踪记   | 湖南文艺出版社  | 2022 | 叶蓄蓄      | 48.03 | 4    | 3    | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 225 | 环境科学、安全科学 | odiahbtp | bkwyim   | 1721 | wktpi    | 81.90 | 90   | 83   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 235 | 医药、卫生     | ofqb     | kobv     | 1976 | qdwjrscl | 54.78 | 18   | 8    | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 236 | 医药、卫生     | oipbpd   | v        | 1002 | drkl     | 6.81  | 91   | 46   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 228 | 工业技术      | otps     | h        | 177  | stdylgv  | 40.31 | 86   | 75   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 243 | 工业技术      | pdvtz    | hcg      | 1638 | zulcbqv  | 25.07 | 30   | 14   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 261 | 自然科学      | powa     | lvtrqcdi | 748  | rfew     | 74.16 | 48   | 43   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
- Pagination:** 首页 上一页 1 2 3 4 5 6 下一页 尾页 页码 跳转

#### 4.4.2.2.1 添加图书

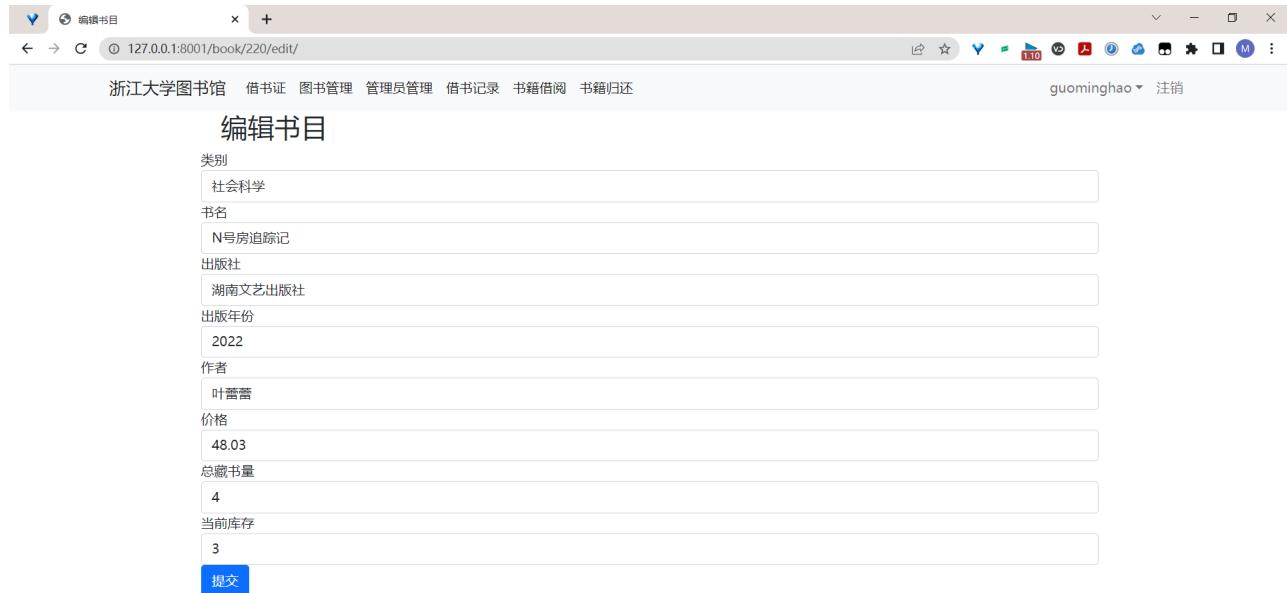
The screenshot shows a web browser window with the URL `127.0.0.1:8001/book/add/`. The page title is "书籍上架". The form fields include:

- 类别:
- 书名:
- 出版社:
- 出版年份:
- 作者:
- 价格:  0
- 总藏书量:
- 当前库存:

At the bottom left is a blue "提交" button.

```
1 def book_add(request):
2     """ 图书添加 """
3     # 直接点击 /book/list/ 上的添加图书
4     if request.method == "GET":
5         form = BookModelForm()
6         context_dict = {
7             "form": form,
8         }
9         return render(request, 'book_add.html', context_dict)
10
11    form = BookModelForm(data=request.POST)
12    if form.is_valid():
13        form.save()
14        return redirect('/book/list/')
15
16    context_dict = {
17        "form": form,
18    }
19    return render(request, 'book_add.html', context_dict)
```

#### 4.4.2.2.2 编辑图书



浙江大学图书馆 借书证 图书管理 管理员管理 借书记录 书籍借阅 书籍归还 guominghao ▾ 注销

编辑书目

类别  
社会科学

书名  
N号房追踪记

出版社  
湖南文艺出版社

出版年份  
2022

作者  
叶蔷薇

价格  
48.03

总藏书量  
4

当前库存  
3

提交

```
1 def book_edit(request, nid):
2     """ 书籍编辑 """
3     edit_object = models.Book.objects.filter(id=nid).first()
4
5     # 进入编辑页面
6     if request.method == "GET":
7         form = BookEditModelForm(instance=edit_object)
8         context_dict = {
9             "form": form
10        }
11        return render(request, 'book_edit.html', context_dict)
12
13    # 提交成功，返回list
14    form = BookEditModelForm(instance=edit_object, data=request.POST)
15    if form.is_valid():
16        form.save()
17        return redirect('/book/list/')
18
19    # 提交失败，跳回原页面
20    context_dict = {
21        "form": form
22    }
23    return render(request, 'book_edit.html', context_dict)
```

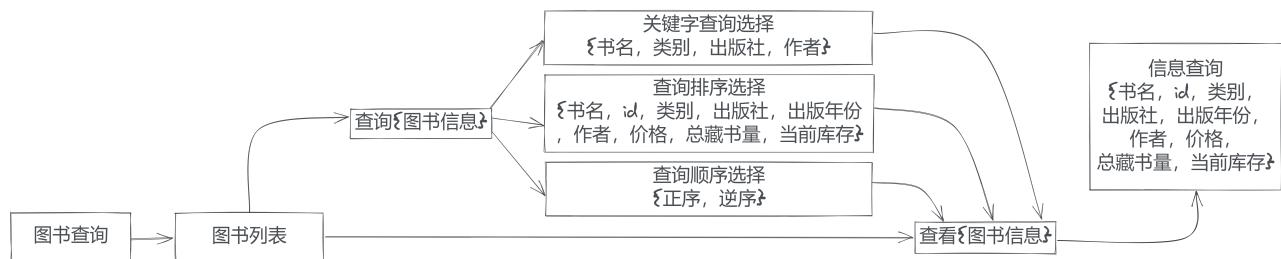
#### 4.4.2.2.3   删除图书

```
1 def book_delete(request, nid):
2     """ 图书删除 """
3     models.Book.objects.filter(id=nid).delete()
4     return redirect('/book/list/')
```

#### 4.4.2.2.4   借阅图书

[同4.4.2.5 书籍借阅](#)

#### 4.4.2.2.5   查询图书



无需登录

指定关键信息和排序方式

The screenshot shows a web-based book management system. At the top, there is a navigation bar with links like '浙江大学图书馆', '借书证', '图书管理', '管理员管理', '借书记录', '书籍借阅', and '书籍归还'. Below the navigation is a search/filter section with fields for '类别' (Category), '排序依据(书名)' (Sort by (Book Name)), '倒序' (Descending), '书名' (Name), '排序顺序(正序)' (Sort Order (Positive)), '查询关键字(价格)' (Query Keyword (Price)), and two search input fields. The main area displays a table of books with columns: id, 类别 (Category), 书名 (Name), 出版社 (Publisher), 出版年份 (Publication Year), 作者 (Author), 价格 (Price), 总藏书量 (Total Collection), 当前库存 (Current Inventory), and 操作 (Operations). Each row in the table includes edit, delete, and borrow buttons. At the bottom, there is a pagination bar with buttons for '首页', '上一页', '1', '2', '下一页', '尾页', '页码', and '跳转'.

| ID  | 类别        | 书名      | 出版社   | 出版年份 | 作者       | 价格    | 总藏书量 | 当前库存 | 操作  |
|-----|-----------|---------|-------|------|----------|-------|------|------|---|
| 238 | 综合性图书     | zoamydk | x     | 19   | igdc     | 67.52 | 76   | 70   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 250 | 环境科学、安全科学 | fkne    | cq    | 197  | peoca    | 30.46 | 42   | 20   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 249 | 医药、卫生     | mvkrj   | zmxn  | 1596 | zbwepof  | 20.22 | 59   | 57   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 258 | 医药、卫生     | iarzbck | auop  | 280  | oenrpzx  | 11.19 | 94   | 88   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 269 | 医药、卫生     | djgakot | iov   | 1611 | pehx     | 79.40 | 53   | 40   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 267 | 医药、卫生     | ykmola  | zenvl | 537  | zjyfg    | 8.08  | 17   | 13   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 227 | 哲学、宗教     | cexqzkb | h     | 1833 | dkpzj    | 70.01 | 92   | 85   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 231 | 哲学、宗教     | zdslki  | rl    | 1002 | qzytijd  | 82.78 | 69   | 18   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 247 | 哲学、宗教     | dakcxzp | lerx  | 91   | rbvcldya | 61.42 | 9    | 7    | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 252 | 自然科学      | nfpkgw  | r     | 1717 | nkqrmo   | 78.56 | 26   | 17   | <button>编辑</button> <button>删除</button> <button>借阅</button> |

| ID  | 类别    | 书名      | 出版社     | 出版年份 | 作者       | 价格    | 总藏书量 | 当前库存 | 操作  |
|-----|-------|---------|---------|------|----------|-------|------|------|---|
| 247 | 哲学、宗教 | dakcxzp | lerx    | 91   | rbvcldya | 61.42 | 9    | 7    | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 238 | 综合性图书 | zoamydk | x       | 19   | igdc     | 67.52 | 76   | 70   | <button>编辑</button> <button>删除</button> <button>借阅</button> |
| 239 | 综合性图书 | ibcvgl  | qwahoxm | 65   | yfhsrxw  | 47.45 | 91   | 82   | <button>编辑</button> <button>删除</button> <button>借阅</button> |

```

1 def book_list(request):
2     """ 图书列表 """
3     data_dict = {}
4     search_data = request.GET.get('query', "")
5     order_sequence = request.GET.get('order_sequence', "")
6     sequence_direct = request.GET.get('sequence_direct', "")
7     order_sequence = sequence_direct + order_sequence
8     if order_sequence == "":
9         order_sequence = "name"
10    search_data_2_1 = request.GET.get('query_2_1', "")
11    search_data_2_2 = request.GET.get('query_2_2', "")
12    what_to_query_data = request.GET.get('what_to_query', "")
13    if search_data:
14        data_dict[what_to_query_data + "__contains"] = search_data
15    elif search_data_2_1 and search_data_2_2:
16        num1 = int(search_data_2_1)
17        num2 = int(search_data_2_2)
18        data_dict[what_to_query_data + "__range"] = (num1, num2)
19
20    queryset = models.Book.objects.filter(**data_dict).order_by(order_sequence,
21 "category")
22    page_object = Pagination(request, queryset)
23    context_dict = {
24        "search_data": search_data,
25        "queryset": page_object.page_queryset, # 分页之后的数据
26        "page_string": page_object.html(), # html
27        "search_data_2_1": search_data_2_1,
28        "search_data_2_2": search_data_2_2,
29    }
30    return render(request, 'book_list.html', context_dict)

```

#### 4.4.2.2.6 从excel批量导入图书

```
1 def book_create_from_excel(request):
2     books_to_put_on = pd.read_excel(r'basic_func/static/file/book_info.xlsx')
3     for item in books_to_put_on.index:
4         info = books_to_put_on.loc[item]
5         models.Book.objects.create(category=info.values[0] ,
6                                     name=info.values[1] ,
7                                     press=info.values[2] ,
8                                     year=info.values[3] ,
9                                     author=info.values[4] ,
10                                    price=info.values[5] ,
11                                    total=info.values[6] ,
12                                    inventory=info.values[7])
13     print("### ", os.getcwd())
14     return redirect('/book/list/')
```

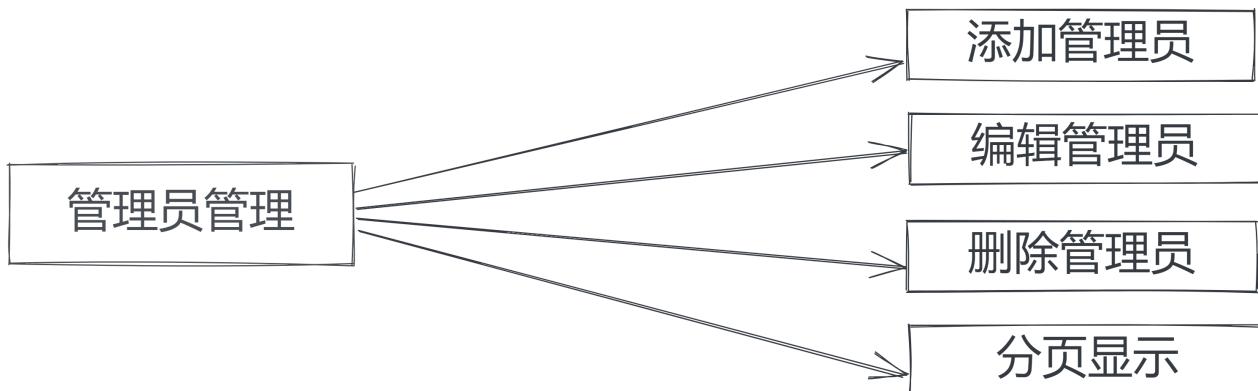
#### 4.4.2.2.7 随机上架5本图书

```
1 def book_random_create(request):
2     for i in range(5):
3         name = ''.join(random.sample(
4             ['z', 'y', 'x', 'w', 'v', 'u', 't', 's', 'r', 'q', 'p', 'o', 'n', 'm', 'l',
5             'k', 'j', 'i', 'h', 'g', 'f',
6             'e', 'd', 'c', 'b', 'a'], random.randint(4, 8)))
6         category = random.randint(1, 9)
7         press = ''.join(random.sample(
8             ['z', 'y', 'x', 'w', 'v', 'u', 't', 's', 'r', 'q', 'p', 'o', 'n', 'm', 'l',
9             'k', 'j', 'i', 'h', 'g', 'f',
10            'e', 'd', 'c', 'b', 'a'], random.randint(1, 8)))
10        year = random.randint(0, 2022)
11        # author = random.sample('zyxwvutsrqponmlkjihgfedcba', random.randint(4, 8))
12        author = ''.join(random.sample(
13            ['z', 'y', 'x', 'w', 'v', 'u', 't', 's', 'r', 'q', 'p', 'o', 'n', 'm', 'l',
14            'k', 'j', 'i', 'h', 'g', 'f',
15            'e', 'd', 'c', 'b', 'a'], random.randint(4, 8)))
16        price = random.uniform(0, 99)
17        total = random.randint(1, 99)
18        inventory = random.randint(1, total)
19        models.Book.objects.create(name=name, category=category, press=press, year=year,
20                                   author=author, price=price,
21                                   total=total, inventory=inventory)
22     return redirect('/book/list/')
```

#### 4.4.2.2.8 清空图书馆

```
1 def book_delete_all(request):
2     models.Book.objects.all().delete()
3     return redirect('/book/list/')
```

#### 4.4.2.3 管理员管理



管理员管理

| 添加管理员 |            |      |             |   |
|-------|------------|------|-------------|---|
| id    | 姓名         | 密码   | 手机号         | 操作                                      |
| 6     | guominghao | gmh  | 18019072907 | <button>编辑</button> <button>删除</button> |
| 5     | root       | root | 18019072907 | <button>编辑</button> <button>删除</button> |

首页 上一页 1 下一页 尾页 页码 跳转

#### 4.4.2.3.1 添加管理员

添加管理员

密码  
姓名  
手机号

添加

```
1 def admin_add(request):
2     """ 借书证添加 """
3     if request.method == "GET":
```

```
4     form = AdminModelForm()
5     context_dict = {
6         "form": form,
7     }
8     return render(request, 'admin_add.html', context_dict)
9
10    form = AdminModelForm(data=request.POST)
11    if form.is_valid():
12        form.save()
13        return redirect('/admin/list/')
14
15    context_dict = {
16        "form": form,
17    }
18    return render(request, 'admin_add.html', context_dict)
```

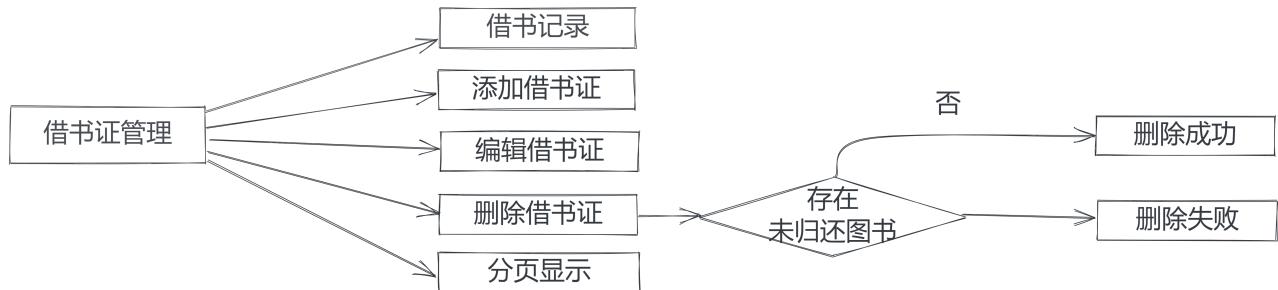
#### 4.4.2.3.2 编辑管理员

```
1 def admin_edit(request, nid):
2     """ 借书证编辑 """
3     edit_object = models.Admin.objects.filter(id=nid).first()
4
5     # 进入编辑页面
6     if request.method == "GET":
7         form = AdminEditModelForm(instance=edit_object)
8         context_dict = {
9             "form": form
10        }
11        return render(request, 'admin_edit.html', context_dict)
12
13    # 提交成功，返回list
14    form = AdminEditModelForm(instance=edit_object, data=request.POST)
15    if form.is_valid():
16        form.save()
17        return redirect('/admin/list/')
18
19    # 提交失败，跳回原页面
20    # print("### failed to submit")
21    context_dict = {
22        "form": form
23    }
24    return render(request, 'admin_edit.html', context_dict)
```

#### 4.4.2.3.3   删除管理员

```
1 def admin_delete(request, nid):
2     """ 借书证删除 """
3     models.Admin.objects.filter(id=nid).delete()
4     return redirect('/admin/list/')
```

#### 4.4.2.4   借书证管理



| id | 姓名           | 机构                  | 职位  | 操作                  |                     |                         |
|----|--------------|---------------------|-----|---------------------|---------------------|-------------------------|
|    |              |                     |     | 编辑                  | 删除                  | 查询借书记录                  |
| 10 | as           | zju                 | 学生  | <button>编辑</button> | <button>删除</button> | <button>查询借书记录</button> |
| 5  | cxy          | zju                 | 学生  | <button>编辑</button> | <button>删除</button> | <button>查询借书记录</button> |
| 1  | gmh          | zju                 | 学生  | <button>编辑</button> | <button>删除</button> | <button>查询借书记录</button> |
| 6  | GuoMinghao   | zju                 | 学生  | <button>编辑</button> | <button>删除</button> | <button>查询借书记录</button> |
| 3  | jll          | ayyz                | 教职工 | <button>编辑</button> | <button>删除</button> | <button>查询借书记录</button> |
| 9  | Julili       | ayyz                | 教职工 | <button>编辑</button> | <button>删除</button> | <button>查询借书记录</button> |
| 12 | test123      | zju                 | 学生  | <button>编辑</button> | <button>删除</button> | <button>查询借书记录</button> |
| 13 | Wu Xiaocheng | zhejiang university | 教职工 | <button>编辑</button> | <button>删除</button> | <button>查询借书记录</button> |
| 8  | Wu Zhaozhi   | zhejiang university | 教职工 | <button>编辑</button> | <button>删除</button> | <button>查询借书记录</button> |

首页 上一页 1 下一页 尾页 页码 跳转

#### 4.4.2.4.1 添加借书证

添加借书证

姓名  
机构  
身份  
-----

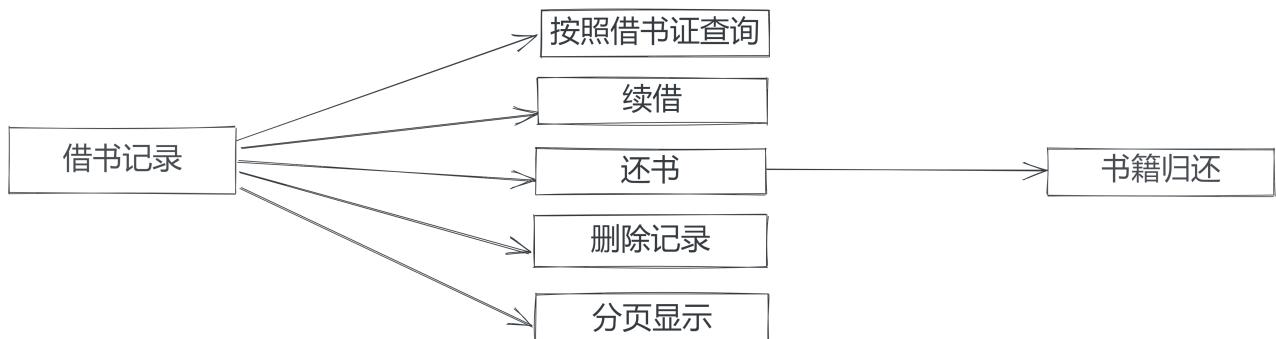
提交

```
1 def card_add(request):
2     """ 借书证添加 """
3     # return render(request, 'card_add.html')
4     if request.method == "GET":
5         form = CardModelForm()
6         context_dict = {
7             "form": form,
8         }
9         return render(request, 'card_add.html', context_dict)
10
11    form = CardModelForm(data=request.POST)
12    if form.is_valid():
13        form.save()
14        return redirect('/card/list/')
15
16    context_dict = {"form": form,}
17    return render(request, 'card_add.html', context_dict)
```

#### 4.4.2.4.2 查询借书记录

顶部“查询借书记录”按钮 => 查询任意借书证的借书记录

右侧操作栏“查询借书记录”按钮 => 查询选中借书证信息的借书证记录



浙江大学图书馆 借书证 图书管理 管理员管理 借书记录 书籍借阅 书籍归还 guominghao 注销

## 借书记录

| 书号                | 类别    | 书名      | 出版社     | 年份   | 作者      | 价格    | 数<br>量 | card_id | admin_id   | 结束日期       | 应还日期       | 是否归<br>还 | 是否被<br>预约 | 操作  |
|-------------------|-------|---------|---------|------|---------|-------|--------|---------|------------|------------|------------|----------|-----------|---|
| Book object (270) | 古籍    | azhkp   | doptm   | 656  | tguxld  | 54.51 | 0      | 13      | root       | 2022-05-26 | 2022-06-26 | False    | True      | <button>续借</button> <button>还书</button> <button>删除记录</button> |
| Book object (270) | 古籍    | azhkp   | doptm   | 656  | tguxld  | 54.51 | 0      | 13      | root       | 2022-05-26 | 2022-06-26 | False    | True      | <button>续借</button> <button>还书</button> <button>删除记录</button> |
| Book object (231) | 哲学、宗教 | zdsldki | rl      | 1002 | qzytjjd | 82.78 | 18     | 13      | None       | 2022-06-03 | 2022-07-03 | False    | False     | <button>续借</button> <button>还书</button> <button>删除记录</button> |
| Book object (220) | 社会科学  | N号房追踪记  | 湖南文艺出版社 | 2022 | 叶蓄蓄     | 48.03 | 2      | 13      | guominghao | 2022-06-03 | 2022-07-03 | False    | False     | <button>续借</button> <button>还书</button> <button>删除记录</button> |

首页 上一页 1 下一页 尾页 页码 跳转

```

1 def get_context_dict_from_record_query(request):
2     """ 用于以下函数的代码复用
3     record_query(request),
4     record_expand(request, nid),
5     record_return(request, nid),
6     record_delete(request, nid)"""
7     search_data = request.GET.get('query', "")
8     queryset = models.Record.objects.filter(card_id=search_data).order_by("-returned",
9 "lend_date")
10    record_book = queryset.select_related('book_id')
11
12    page_object = Pagination(request, record_book)
13    context_dict = {
14        "search_data": search_data,
15        "queryset": page_object.page_queryset, # 分页之后的数据
16        "page_string": page_object.html(), # html
17    }
18
19    return context_dict

```

```

1 def record_query(request):
2     """ 借书记录列表 """
3     # search_data = request.GET.get('query', "")
4     context_dict = get_context_dict_from_record_query(request)
5     return render(request, 'record_list.html', context_dict)

```

#### 4.4.2.4.3 编辑借书证

编辑借书证

姓名  
GuoMinghao

机构  
zju

身份  
学生

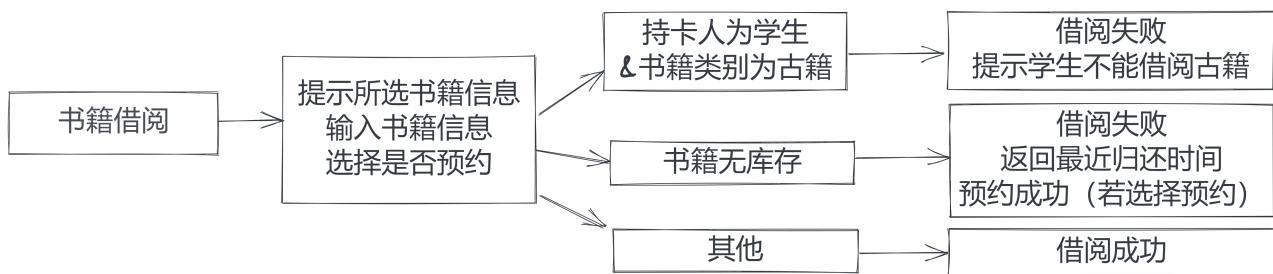
提交

```
1 def card_edit(request, nid):
2     """ 借书证编辑 """
3     edit_object = models.Card.objects.filter(id=nid).first()
4
5     # 进入编辑页面
6     if request.method == "GET":
7         form = CardEditModelForm(instance=edit_object)
8         context_dict = {
9             "form": form
10        }
11        return render(request, 'card_edit.html', context_dict)
12
13    # 提交成功，返回list
14    form = CardEditModelForm(instance=edit_object, data=request.POST)
15    if form.is_valid():
16        form.save()
17        return redirect('/card/list/')
18
19    # 提交失败，跳回原页面
20    # print("### failed to submit")
21    context_dict = {
22        "form": form
23    }
24    return render(request, 'card_edit.html', context_dict)
```

#### 4.4.2.4.4 删除借书证

```
1 def card_delete(request, nid):
2     """ 借书证删除 """
3     query_set = models.Record.objects.filter(card_id=nid).exists()
4     if query_set:
5         return HttpResponse("该图书证存在尚未归还的图书馆，不能删除")
6     models.Card.objects.filter(id=nid).delete()
7     return redirect('/card/list/')
```

#### 4.4.2.5 书籍借阅



The screenshot shows a web browser window titled "浙江大学图书馆 借书证 图书管理 管理员管理 借书记录 书籍借阅 书籍归还". The main content area is titled "书籍借阅". On the left, there is a sidebar with book information: "所选书籍: N号房追踪记", "id: 220", "库存: 3", and "当前管理员: guominghao". On the right, there are input fields for "书本id" (Book object (220)), "借书证id" (13), "管理员id" (guominghao), "借出日期" (2022/06/03), "还书日期" (2022/07/03), and "已归还". Below these fields is a dropdown menu with the option "若无库存, 是否预约一本(否)". At the bottom is a blue "提交" (Submit) button.

```
1 def book_borrow(request, nid):
2     """ 书籍借阅 """
3     borrow_object_query_set = models.Book.objects.filter(id=nid)
4     borrow_object = borrow_object_query_set.first()
5
6     info_dict = request.session.get("info")
7     form = RecordModelForm()
8     error_info = ""
9     success_info = ""
10    context_dict = {
11        "form": form,
12        "borrow_object": borrow_object,
13        "info_dict": info_dict,
14        "error_info": error_info,
15        "success_info": success_info
16    }
17
18    # 进入借阅页面
19    if request.method == "GET":
20        return render(request, 'book_borrow.html', context_dict)
```

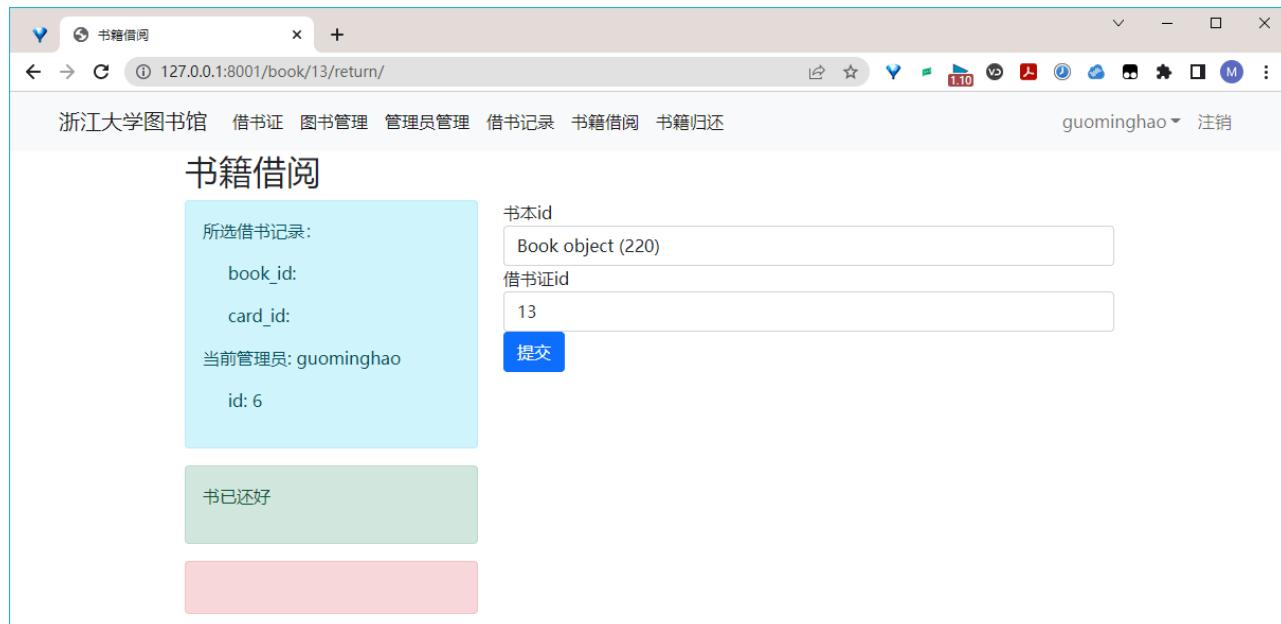
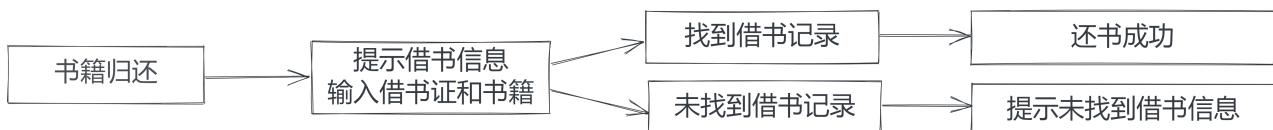
```
21
22     # 提交成功，返回list
23     form = RecordModelForm(data=request.POST)
24     selected_card_id = request.POST.get('card_id')
25     selected_card_object = models.Card.objects.filter(id=selected_card_id).first()
26     make_appoint = False
27     make_appoint = request.POST.get('make_appoint')
28     make_appoint = (make_appoint == "True")
29
30     if selected_card_object.position == 1 and borrow_object.category == 1:
31         error_info = "该借书证持有人为学生，暂无古籍借阅权限"
32         context_dict = {
33             "form": form,
34             "borrow_object": borrow_object,
35             "info_dict": info_dict,
36             "error_info": error_info,
37             "success_info": success_info
38         }
39         return render(request, 'book_borrow.html', context_dict)
40     if borrow_object.inventory == 0:
41         record_object =
42             models.Record.objects.filter(book_id=nid).order_by("return_date").first()
43         error_info = "该书籍暂无库存，最近归还日期为：" + str(record_object.return_date)
44         context_dict = {
45             "form": form,
46             "borrow_object": borrow_object,
47             "info_dict": info_dict,
48             "error_info": error_info,
49             "success_info": success_info
50         }
51         # print("### ", type(make_appoint), " ### ", bool(make_appoint), "122")
52
53         if make_appoint:
54             book_select = models.Record.objects.filter(book_id=nid).first()
55             if book_select.appointed:
56                 return HttpResponse("不能预约，已被其他人预约")
57             models.Record.objects.filter(book_id=nid).update(appointed=True)
58
59             return render(request, 'book_borrow.html', context_dict)
60
61         if form.is_valid():
62             form.save()
63             borrow_object_query_set.update(inventory=borrow_object.inventory - 1)
64             success_info = "借阅成功"
65             context_dict = {
66                 "form": form,
67                 "borrow_object": borrow_object,
68                 "info_dict": info_dict,
```

```

68         "error_info": error_info,
69         "success_info": success_info
70     }
71     return render(request, 'book_borrow.html', context_dict)
72
73     # return HttpResponse("借阅成功")
74
75     # 提交失败，跳回原页面
76     return render(request, 'book_borrow.html', context_dict)

```

#### 4.4.2.6 书籍归还



```

1 def book_return(request, nid):
2     """ 书籍归还 """
3     # nid: Record - id
4     record_object_query_set = models.Record.objects.filter(id=nid)
5     record_object = record_object_query_set.first()
6
7     info_dict = request.session.get("info")
8     form = RecordReturnModelForm()
9     error_info = ""
10    success_info = ""
11    context_dict = {
12        "form": form,
13        "record_object": record_object,
14        "info_dict": info_dict,
15        "error_info": error_info,
16        "success_info": success_info

```

```
17     }
18
19     # 进入还书页面
20     if request.method == "GET":
21         return render(request, 'book_return.html', context_dict)
22
23     # 提交成功，返回list
24     form = RecordReturnModelForm(data=request.POST)
25     selected_card_id = request.POST.get('card_id')
26     selected_book_id = request.POST.get('book_id')
27     selected_card_object_query_set = \
28         models.Record.objects.filter(book_id=selected_book_id,
29                                     card_id=selected_card_id,
30                                     returned=False)
31     if not selected_card_object_query_set.exists():
32         error_info = "找不到借书记录"
33         context_dict = {
34             "form": form,
35             "record_object": record_object,
36             "info_dict": info_dict,
37             "error_info": error_info,
38             "success_info": success_info
39         }
40     return render(request, 'book_return.html', context_dict)
41
42     selected_card_object = selected_card_object_query_set.first()
43
44     returned_book_query_set = models.Book.objects.filter(id=selected_book_id)
45     returned_book = returned_book_query_set.first()
46     returned_book_query_set.update(inventory=returned_book.inventory + 1)
47
48     models.Record.objects.filter(id=selected_card_object.id).update(returned=True)
49
50     success_info = "书已还好"
51     context_dict = {
52         "form": form,
53         "record_object": record_object,
54         "info_dict": info_dict,
55         "error_info": error_info,
56         "success_info": success_info
57     }
58     return render(request, 'book_return.html', context_dict)
59
60     # 提交失败，跳回原页面
61     return render(request, 'book_return.html', context_dict)
```

## 4.5 文件结构

```
1 lib_manege
2 |   db.sqlite3
3 |   manage.py
4 |   Monaco.ttf
5 |
6 |   .idea
7 |   |   .gitignore
8 |   |   lib_manage.iml
9 |   |   misc.xml
10 |   |   modules.xml
11 |   |   workspace.xml
12 |   |
13 |   |   inspectionProfiles
14 |   |       profiles_settings.xml
15 |
16 |   basic_func
17 |   |   admin.py
18 |   |   apps.py
19 |   |   models.py
20 |   |   test.py
21 |   |   tests.py
22 |   |   __init__.py
23 |
24 |   |   middleware
25 |   |   |   auth.py
26 |   |   |
27 |   |   |   __pycache__
28 |   |   |       auth.cpython-37.pyc
29 |
30 |   |   migrations
31 |   |   |   0001_initial.py
32 |   |   |   0002_alter_book_name.py
33 |   |   |   0003_alter_book_category.py
34 |   |   |   0004_auto_20220514_0942.py
35 |   |   |   0005_record_returned.py
36 |   |   |   0006_alter_record_book_id.py
37 |   |   |   0007_alter_record_appointed.py
38 |   |   |   0008_auto_20220523_2308.py
39 |   |   |   __init__.py
40 |
41 |   |   |   __pycache__
42 |   |   |       0001_initial.cpython-37.pyc
43 |   |   |       0002_alter_book_name.cpython-37.pyc
44 |   |   |       0003_alter_book_category.cpython-37.pyc
45 |   |   |       0004_auto_20220514_0942.cpython-37.pyc
46 |   |   |       0005_record_returned.cpython-37.pyc
```

```
47 | | | 0006_alter_record_book_id.cpython-37.pyc
48 | | | 0007_alter_record_appointed.cpython-37.pyc
49 | | | 0008_auto_20220523_2308.cpython-37.pyc
50 | | | __init__.cpython-37.pyc
51 |
52 | | static
53 | | | css
54 | | | | bootstrap.css
55 | | | | bootstrap.min.css
56 |
57 | | | file
58 | | | | book_info.xlsx
59 | | |
60 | | | img
61 | | | | code.jpg
62 | | | | zju_background.jpg
63 | | |
64 | | js
65 | | | bootstrap.bundle.min.js
66 | | | jquery-3.6.0.min.js
67 |
68 | | templates
69 | | | admin_add.html
70 | | | admin_edit.html
71 | | | admin_list.html
72 | | | book_add.html
73 | | | book_borrow.html
74 | | | book_edit.html
75 | | | book_list.html
76 | | | book_return.html
77 | | | card_add.html
78 | | | card_edit.html
79 | | | card_list.html
80 | | | layout.html
81 | | | login.html
82 | | | record_list.html
83 |
84 | | utils
85 | | | bootstrap.py
86 | | | form.py
87 | | | pagination.py
88 |
89 | | | __pycache__
90 | | | | bootstrap.cpython-37.pyc
91 | | | | form.cpython-37.pyc
92 | | | | pagination.cpython-37.pyc
93 |
94 | | views
```

```
95 |   |   | account.py
96 |   |   | admin.py
97 |   |   | book.py
98 |   |   | book_info.xlsx
99 |   |   | card.py
100 |   |   | code.py
101 |   |   |
102 |   |   └--__pycache__
103 |   |       account.cpython-37.pyc
104 |   |       admin.cpython-37.pyc
105 |   |       book.cpython-37.pyc
106 |   |       card.cpython-37.pyc
107 |   |       code.cpython-37.pyc
108 |
109 |   |   └--__pycache__
110 |   |       admin.cpython-37.pyc
111 |   |       apps.cpython-37.pyc
112 |   |       models.cpython-37.pyc
113 |   |       __init__.cpython-37.pyc
114 |
115 |   └--lib_manage
116 |       asgi.py
117 |       settings.py
118 |       urls.py
119 |       wsgi.py
120 |       __init__.py
121 |
122 |   └--__pycache__
123 |       settings.cpython-37.pyc
124 |       settings.cpython-39.pyc
125 |       urls.cpython-37.pyc
126 |       urls.cpython-39.pyc
127 |       wsgi.cpython-37.pyc
128 |       wsgi.cpython-39.pyc
129 |       __init__.cpython-37.pyc
130 |       __init__.cpython-39.pyc
131 |
132 |   └--__pycache__
133 |       manage.cpython-37.pyc
```

## 5 要求之外的设计/功能

1. 数据表结构的设计
2. 视觉呈现
  - (a) 网页界面，利用 `jQuery` 和 `bootstrap` 设计 `css`
  - (b) 所有信息列表分页显示
3. 用户认证
  - (a) 保持7天登录状态
  - (b) 登录验证码
4. 图书馆管理
  - (a) 随机生成5本图书并上架（测试用）
  - (b) 清空图书馆
5. 信息编辑，信息删除
  - (a) 书籍信息，借书证信息，管理员信息，借书信息的编辑和删除
6. 书籍信息查询
  - (a) 指定查询关键字
  - (b) 指定查询结果显示关键字及顺序（升序，降序）
7. 借阅书籍细节
  - (a) 借书日期自动填写，还书信息默认填写一个月后
  - (b) 一键续借（默认续借一个月）
  - (c) 学生身份不能借阅古籍
8. 预约机制
  - (a) 可选择是否预约无库存的图书
  - (b) 已被预约的图书不能续借

## 6 实验总结

### 6.1 问题及解决过程

教程中并没有讲解如果利用 `Django` 进行  $\Theta$  连接，网上教程也很少涉及  $\Theta$  连接，但我是需要用到  $\Theta$  连接。

### 6.2 错误及原因分析

经过 `baidu`, `csdn`, `google`, `overstackflow` 等处的失败搜索后，仔细阅读了 `Django` 官方说明文档中相关部分，找到了解决方法。

## 6.3 体会和收获

仔细阅读相关资料真的太重要了。[django](#)最开始只学了一点就开始着手去写项目，然后发现不会的太多，然后仔细阅读了相关的教程，并看了一些讲解之后，做起来就顺利了许多。

## 7 参考资料

1. Database System Concepts 7th Edition
2. Python语言程序设计 ([美]梁勇(Lang Y. D.))
3. [Django](#)从入门到实践 (B站教程)
4. [Django](https://docs.djangoproject.com/zh-hans/3.2/)说明文档 <https://docs.djangoproject.com/zh-hans/3.2/>
5. [Django Github](#) <https://github.com/django/django>
6. [Django的models操作](#)
7. [表单 ModelForm](#)
8. [django学习——objects.filter\(\)用法](#)
9. [表连接——B站](#)