图书借阅管理系统设计实验报告

## 实现方式及细节

**技术架构**

客户端技术：Bootstrap

服务端技术：Django Python Web框架

数据库：MySQL

编程环境：Python 3.6.5

Django 2.0

PyCharm Professional 2018.1

**使用场景及其实现**

本课程设计以python IDE PyCharm作为基本开发环境，以django作为服务端框架，首先完成了服务端框架和后端MySQL数据库的连接。其主要工作原理是：通过django ORM创建数据库表结构，并通过ORM框架实现对数据库的增删查改，从而以较为简单直接的方式避免了直接书写sql语句来操作数据库。

Django的使用模式可以概括为MTV，即将应用程序分解成三个组成部分：model，view，template. 对应到我们的数据库设计，model处理与数据相关的事务，定义了关系数据库的各个字段；template处理与前端表现相关的决定，以在页面或其他类型的文档中进行显示；view包含了存取模型及调取恰当模板的相关逻辑，是模型与模板之间的桥梁，给出了对数据库的各种操作的具体实现，是设计的核心部分。

整个设计的流程为：首先确定系统要实现的主要功能以及服务的对象；然后定义相应的实体与实体之间的联系，理清数据库里要存储的主要内容；接下来对每个操作编写一个视图函数，从前端获取用户请求之后进行相应处理，返回结果至后端数据库，或从数据库中取出相应数据显示在前端；再进行url映射，将每个url地址与一个特定的视图函数进行关联；最后是渲染前端HTML页面，进行调试与美化。这些步骤也是在实践中相互交叉、不断修改完善的。

由于使用了django的ORM，对数据库的操作其实是非常简单的。比如在对书的增删查改的过程中，往往是先使用objects.get或objects.filter按照ISBN编号（主键）从数据库中取出相应的数据，然后对其进行操作，再把取出的对象save即可保存修改到数据库。而用户的注册登录系统和登录权限等的控制也是基于django自带的auth\_user，我只是将其进行了字段的映射和前端显示的美化，并增添了修改头像等一些功能。

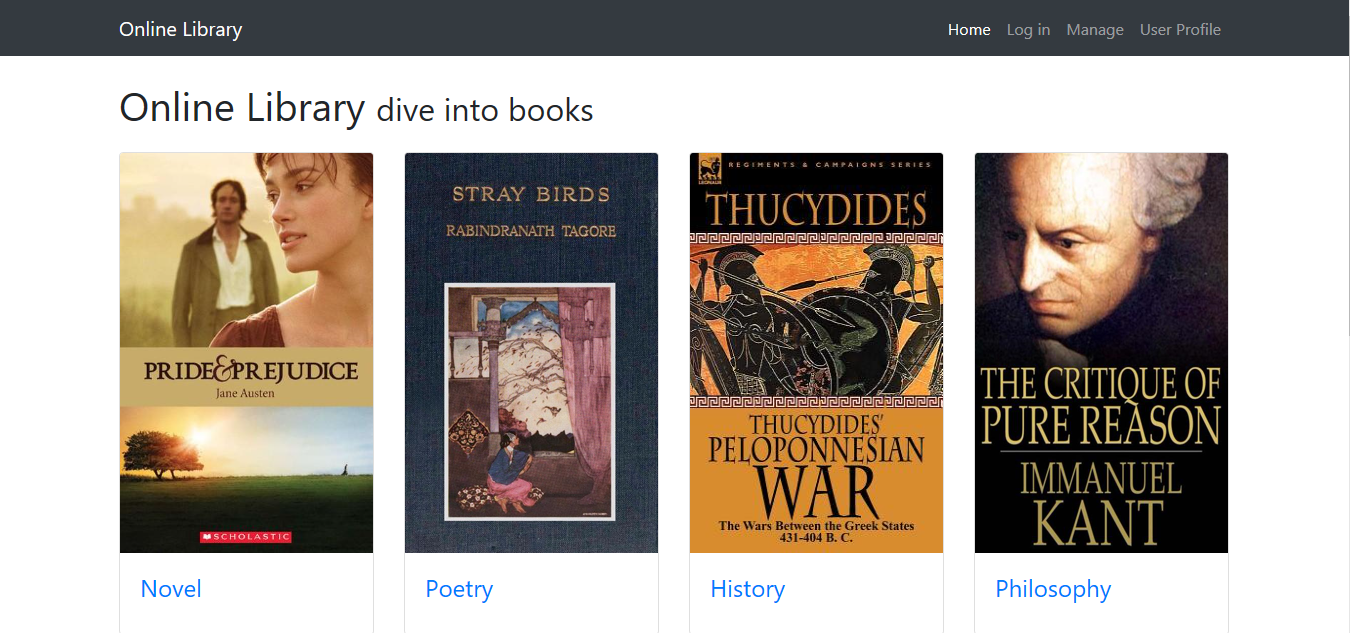
**主要功能**

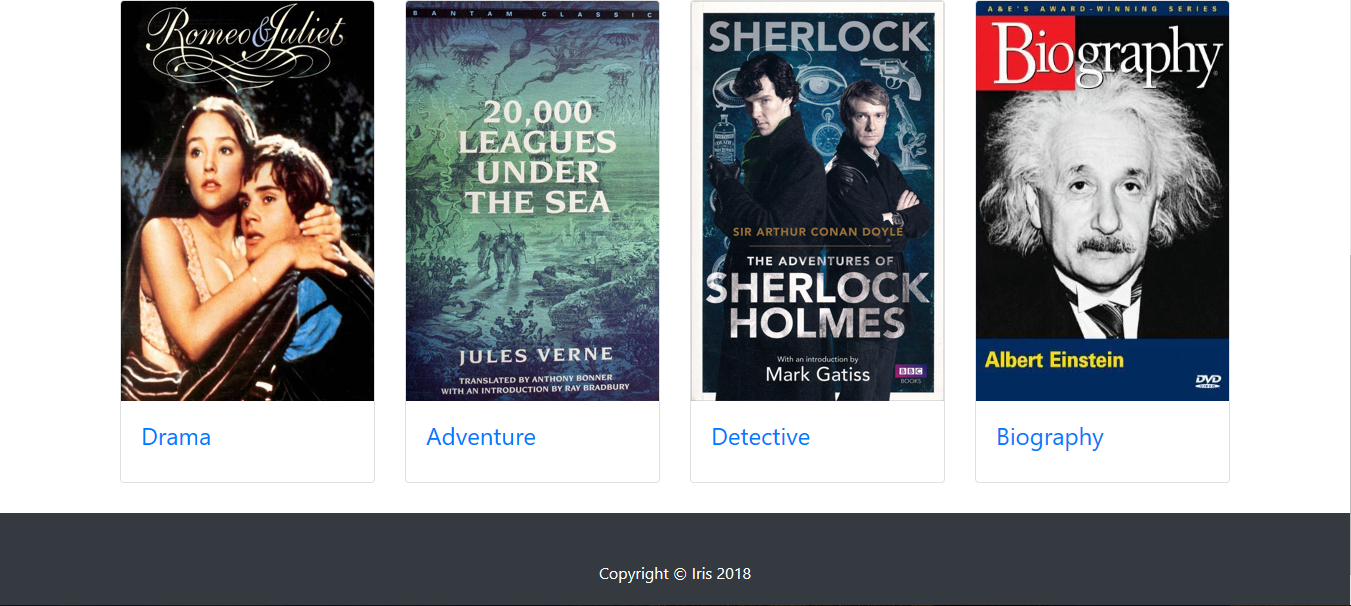
按照不同用户，可以分为：

### 普通读者

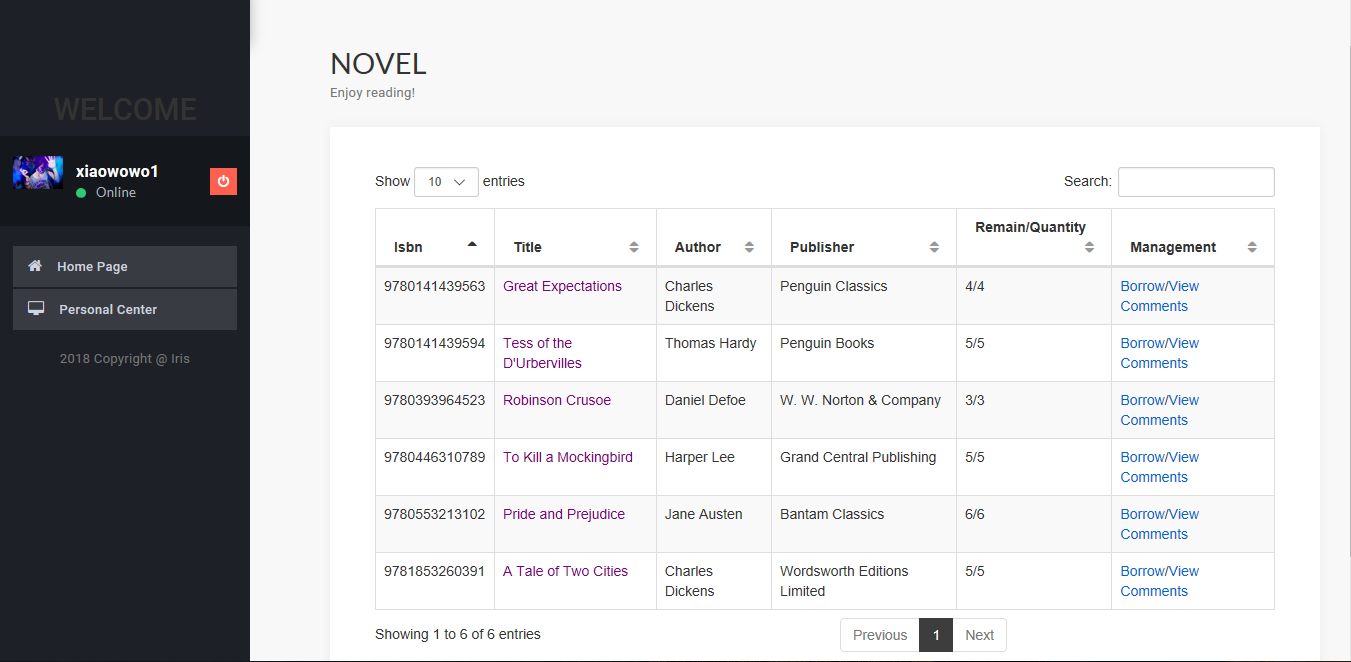
**无需登录：**

* 进入主页，根据主页上的图书分类，查看相应书籍：

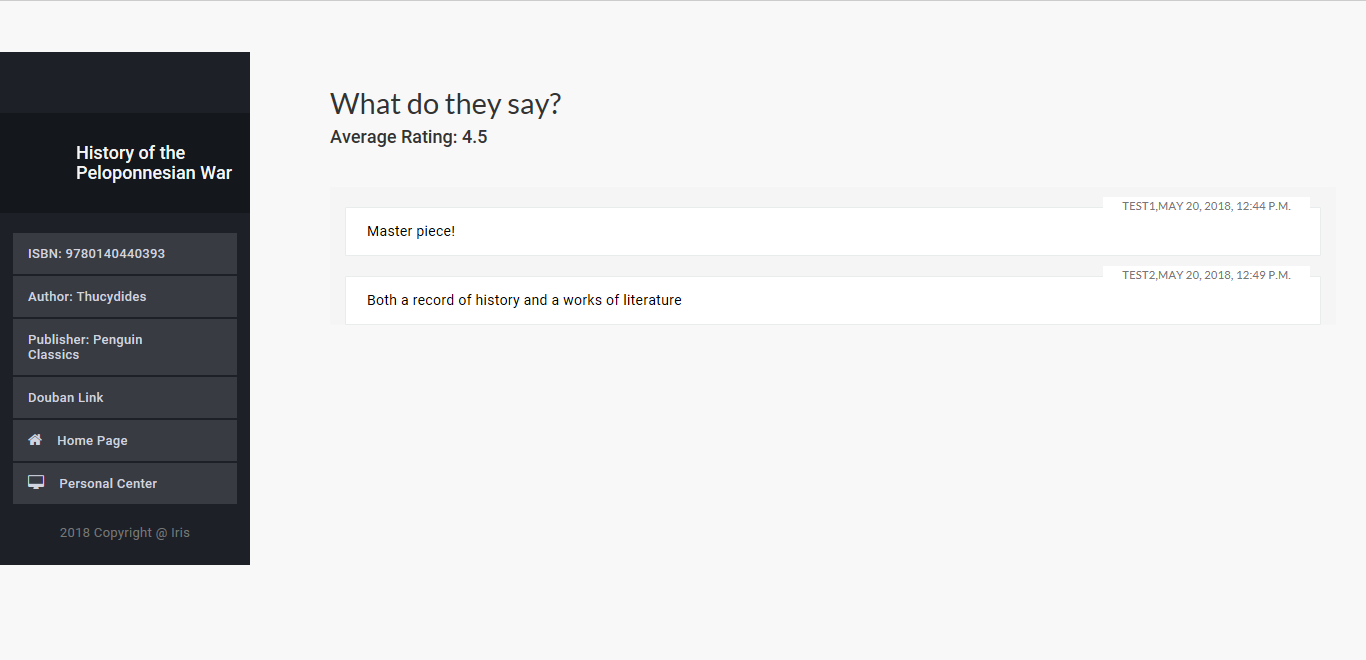




* 进入对应分类的书籍界面，查看所有图书。带有分页和模糊搜索、多关键字搜索功能，支持按照每个属性对书本进行排序。（登录后左侧菜单栏才会出现当前账户状态信息，包括头像，昵称和退出按钮）

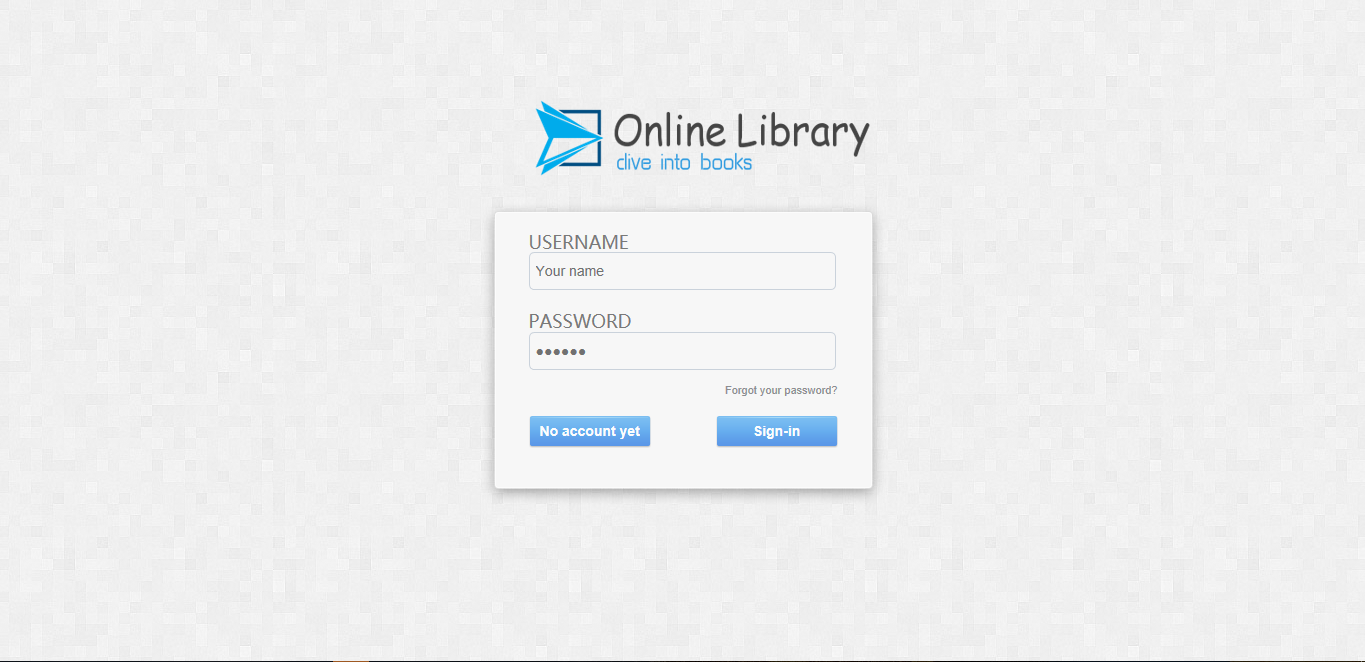


* 通过点击书名链接，可以进入该书的豆瓣主页。点击view comments，可以查看所有读者对该书的书评。

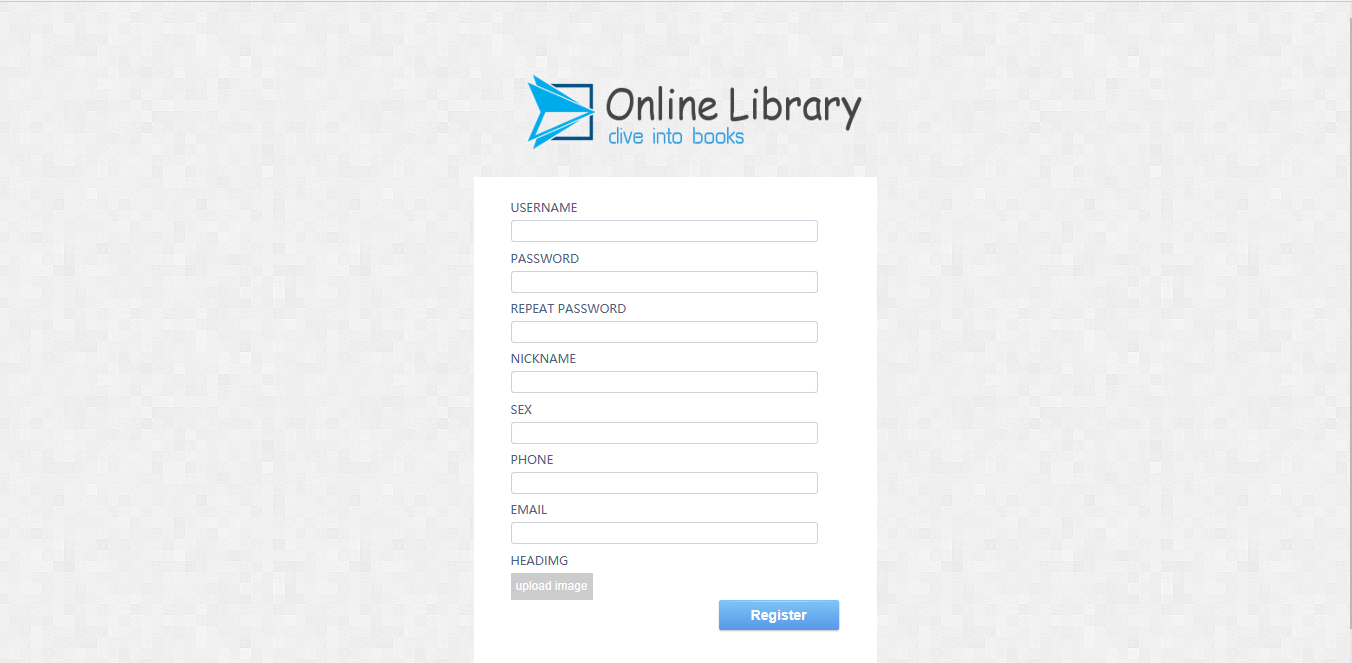


**登录后解锁更多功能**

* 点击主页顶部导航栏的log in 按钮，登录

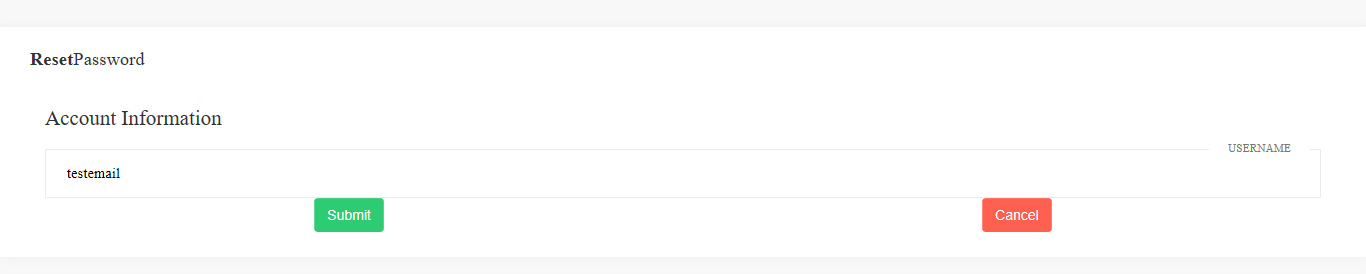


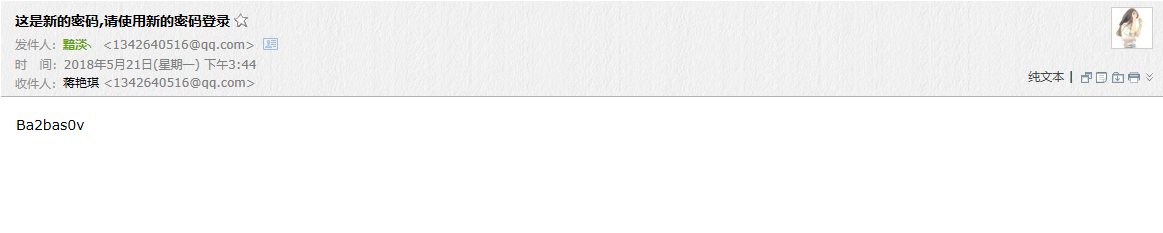
* 若还没有账号，点击No account yet进行注册



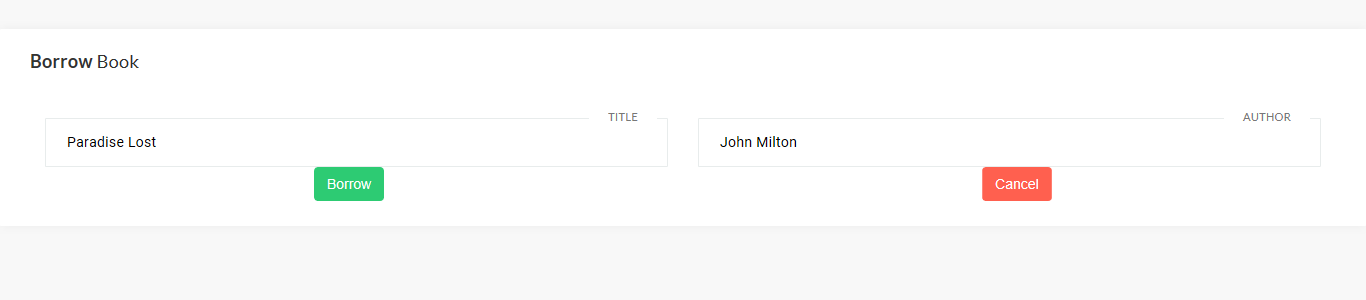
注册后需要登录。

* 忘记密码点击forgot password，输入用户名则系统自动向注册时的邮箱发送新密码。





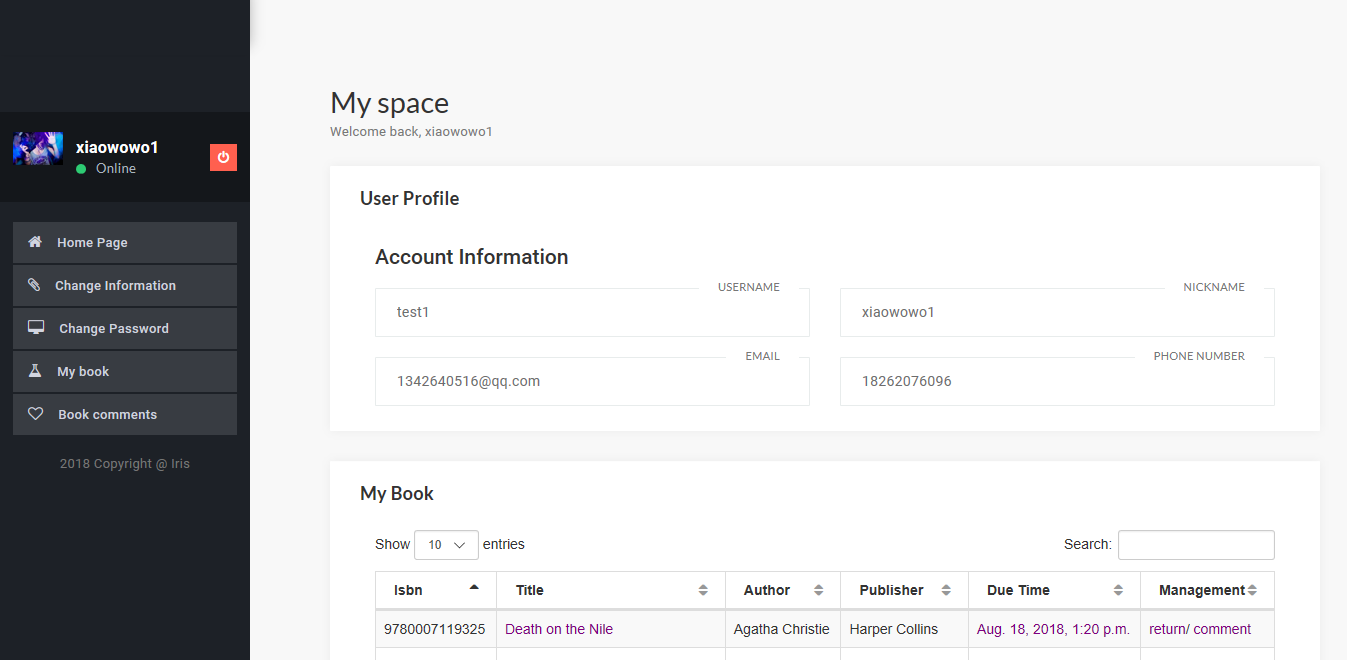
* 在书籍列表中点击Borrow，进行借阅

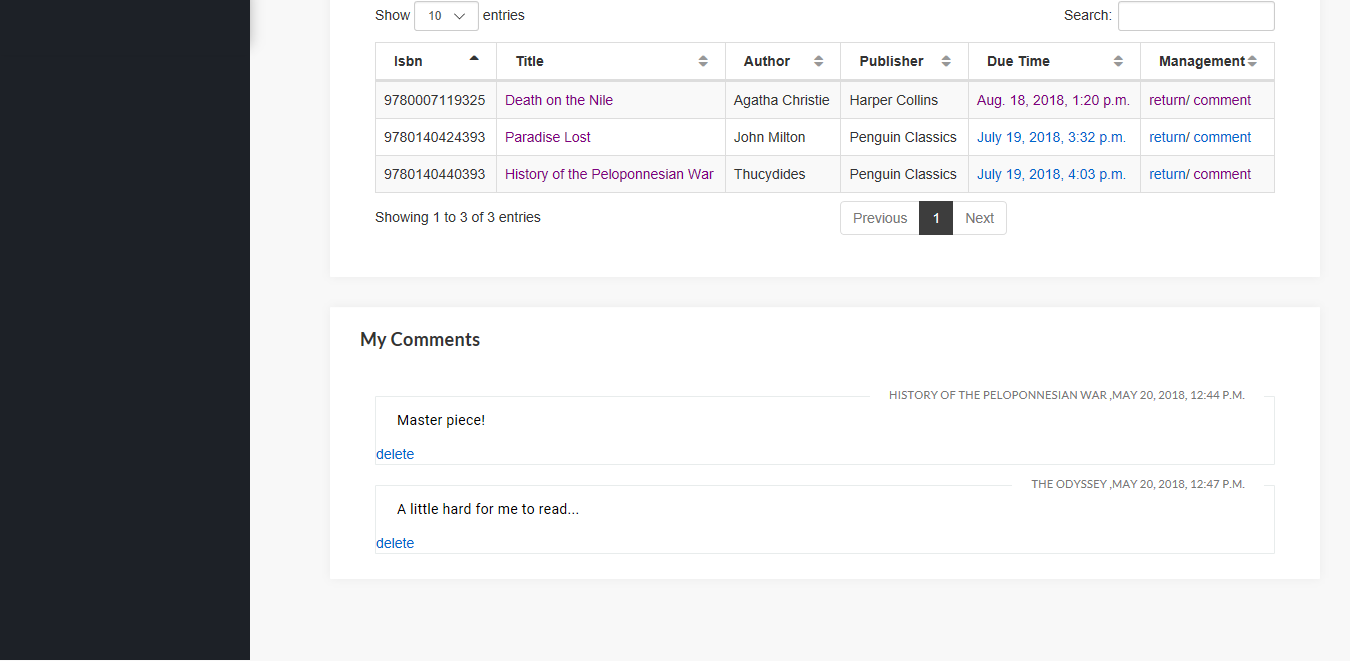


确认借阅信息。

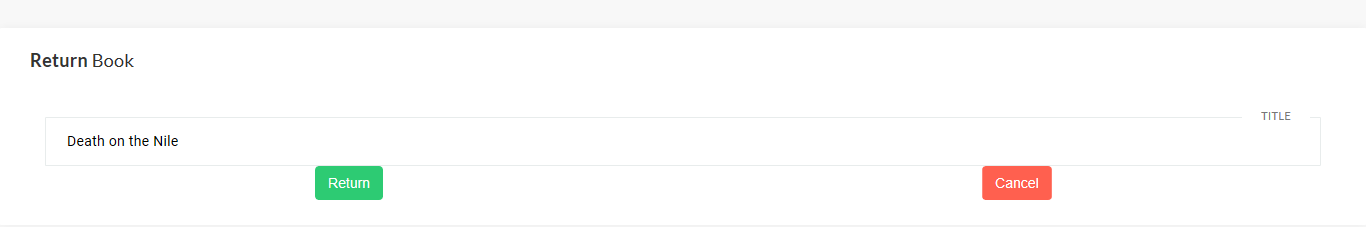
* 借阅后自动跳转至个人中心，或在主页顶端导航栏点击User Profile进入个人中心。

Account Information显示账户信息，My Book显示当前借阅书籍，My Comments显示当前用户的历史评论。点击delete可以删除用户自己的评论。

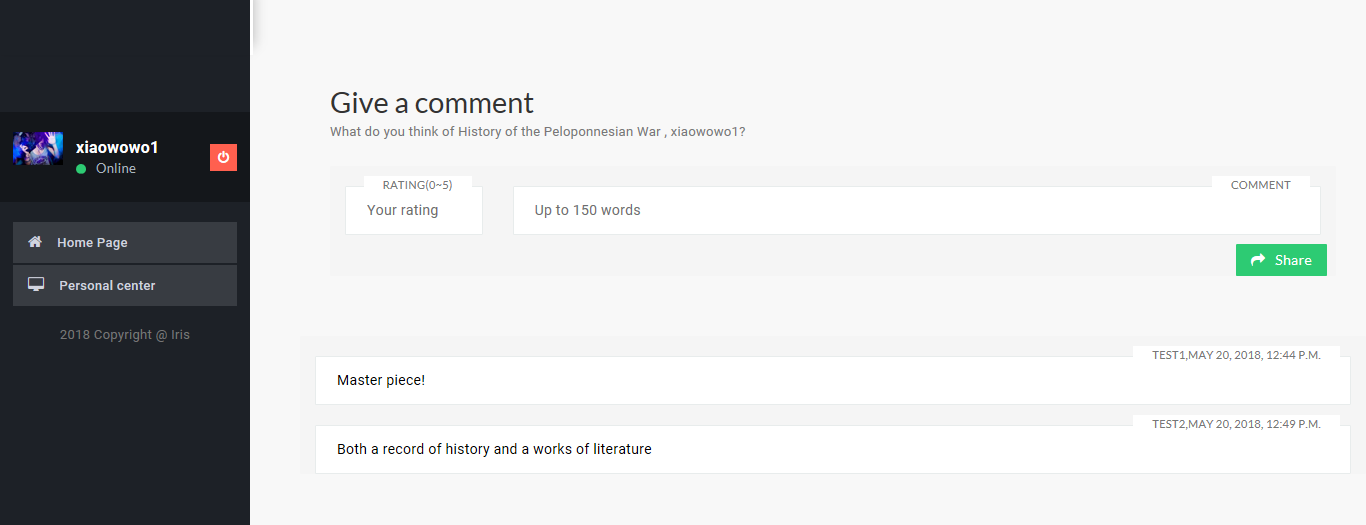




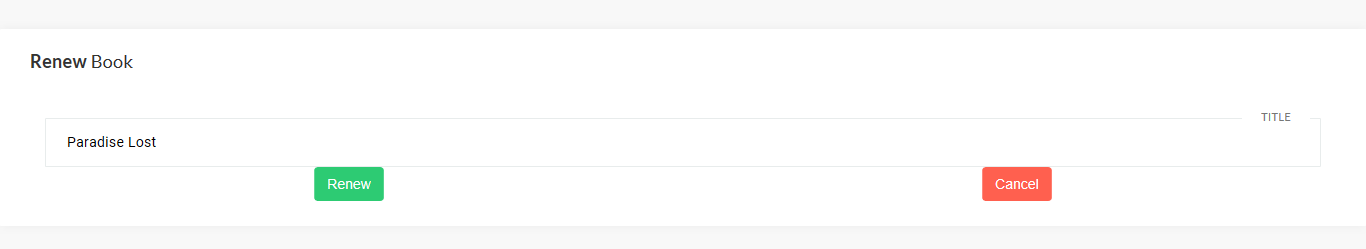
* 点击return 按钮，确认还书信息，还书完成后自动跳转到个人中心



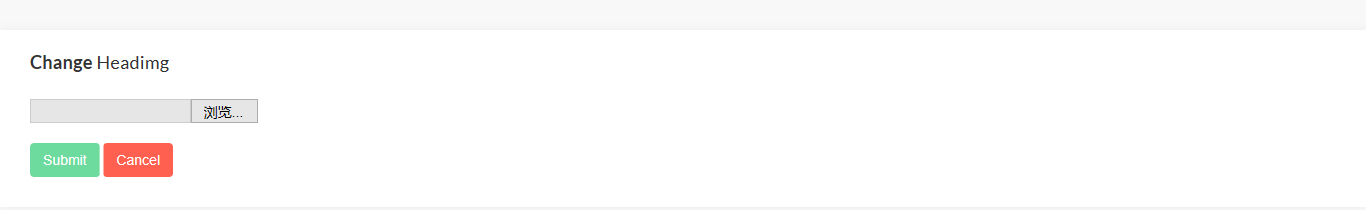
* 点击comment按钮，对相应书籍发表评论，同时下方会显示该书的所有评论。



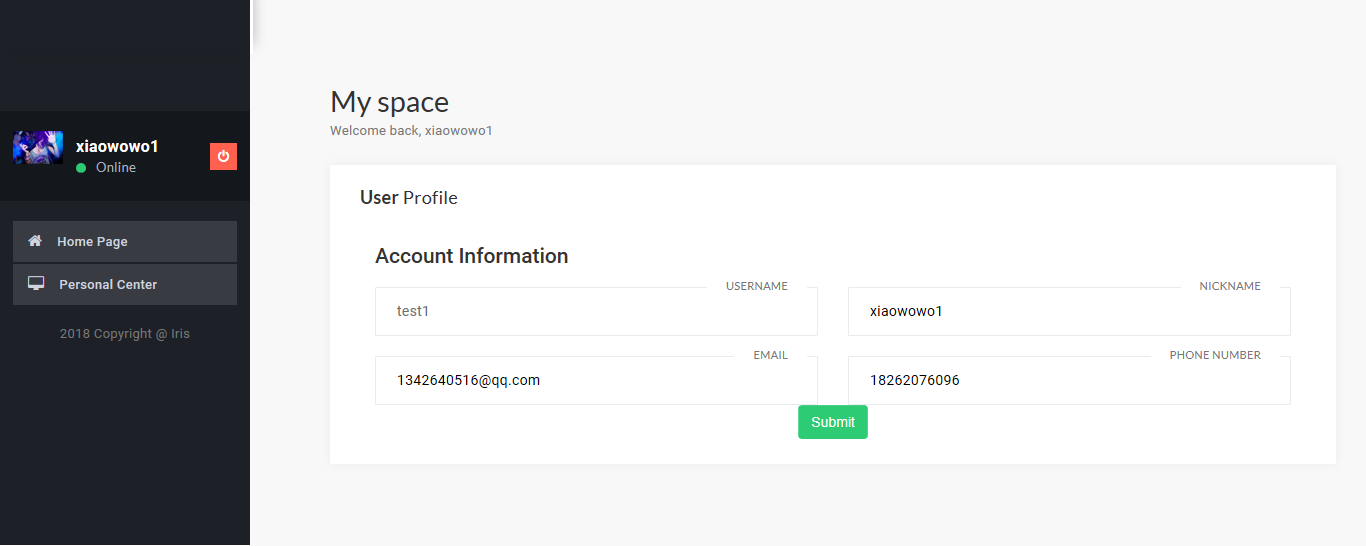
* 点击到期时间，可以续借一个月。



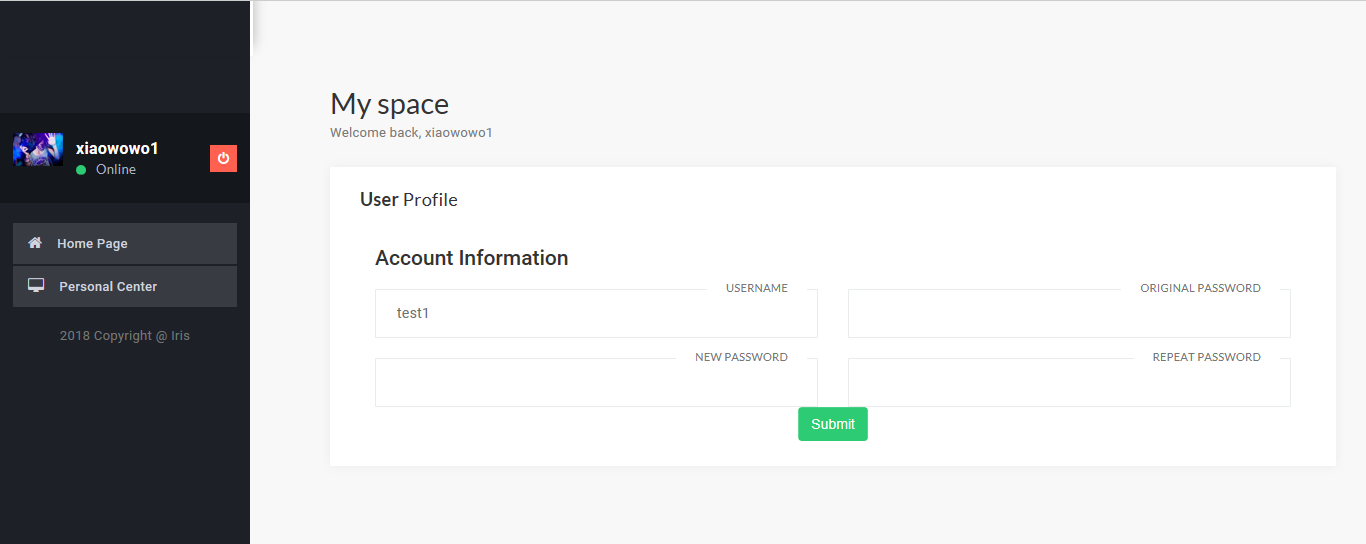
* 点击用户头像，可以从本地上传文件，更改头像



* 点击Change Information，可以修改除了用户名之外的账户信息

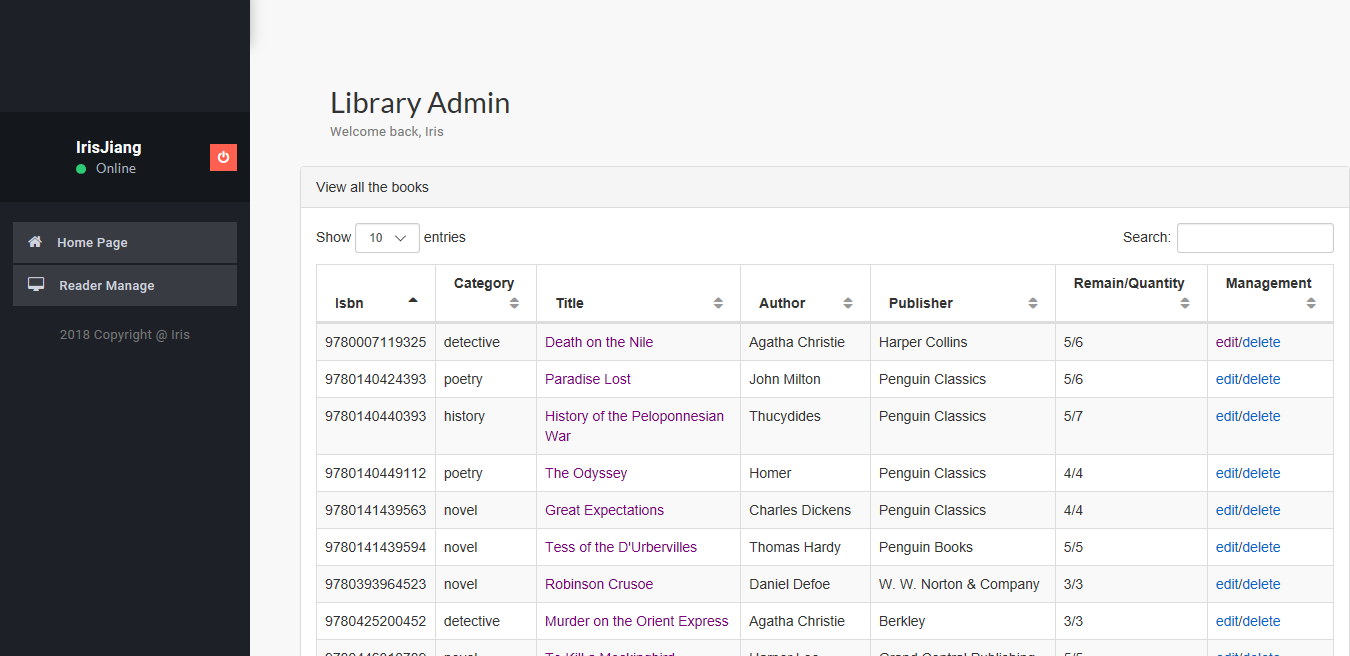


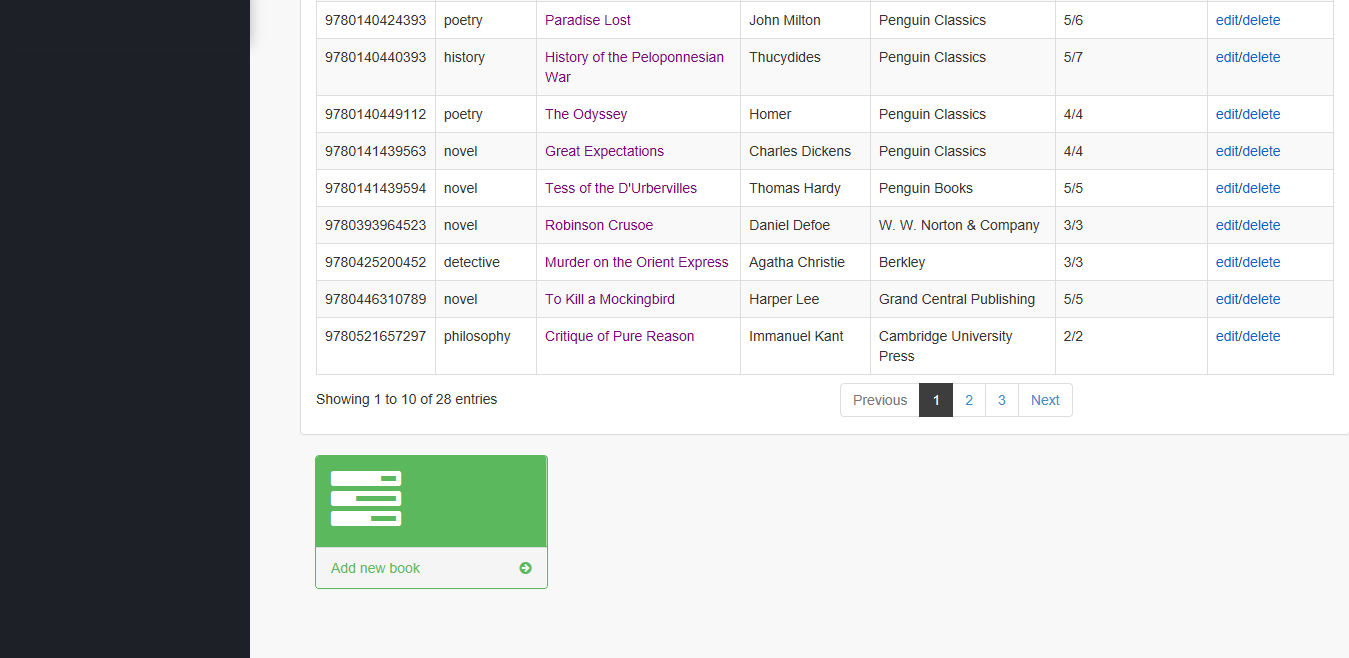
* 点击Change Password，可以修改密码



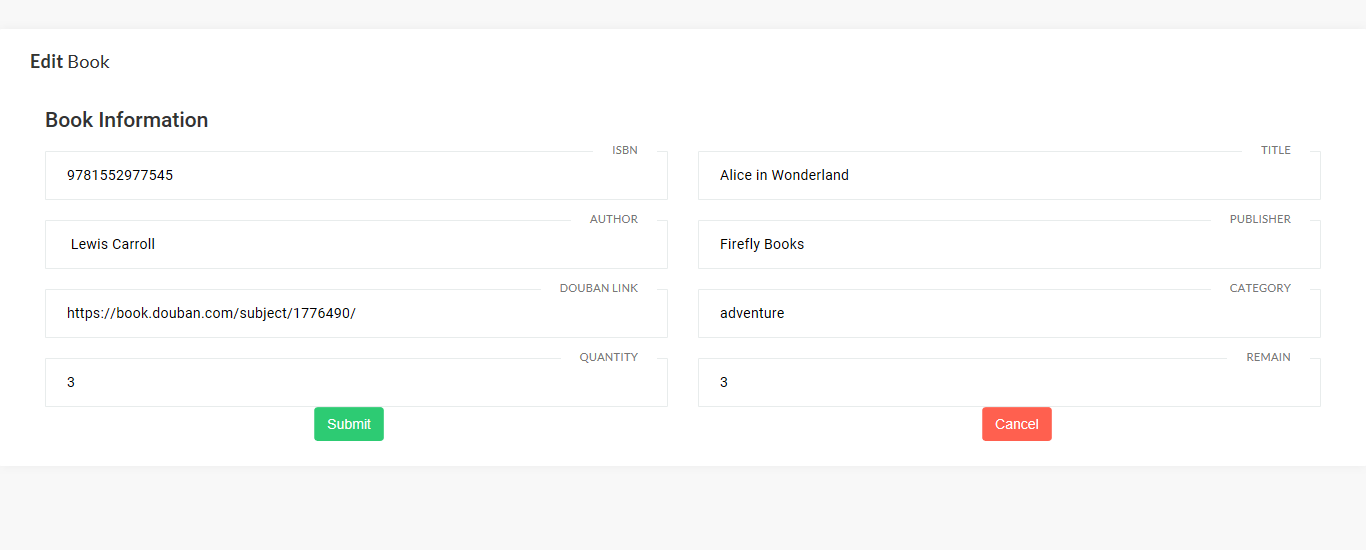
### 图书管理员

* 点击主页顶端导航栏的Manage。若当前登录的用户不是管理员，或未登录，则跳转到管理员登录界面（与读者登录界面相似）登录后进入图书管理页面

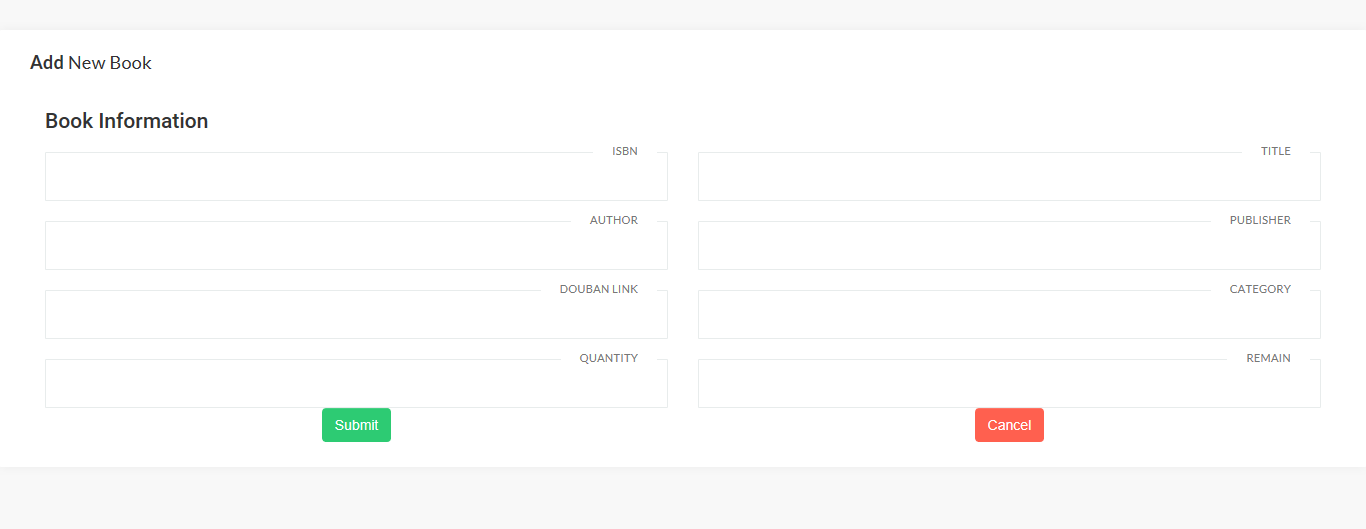




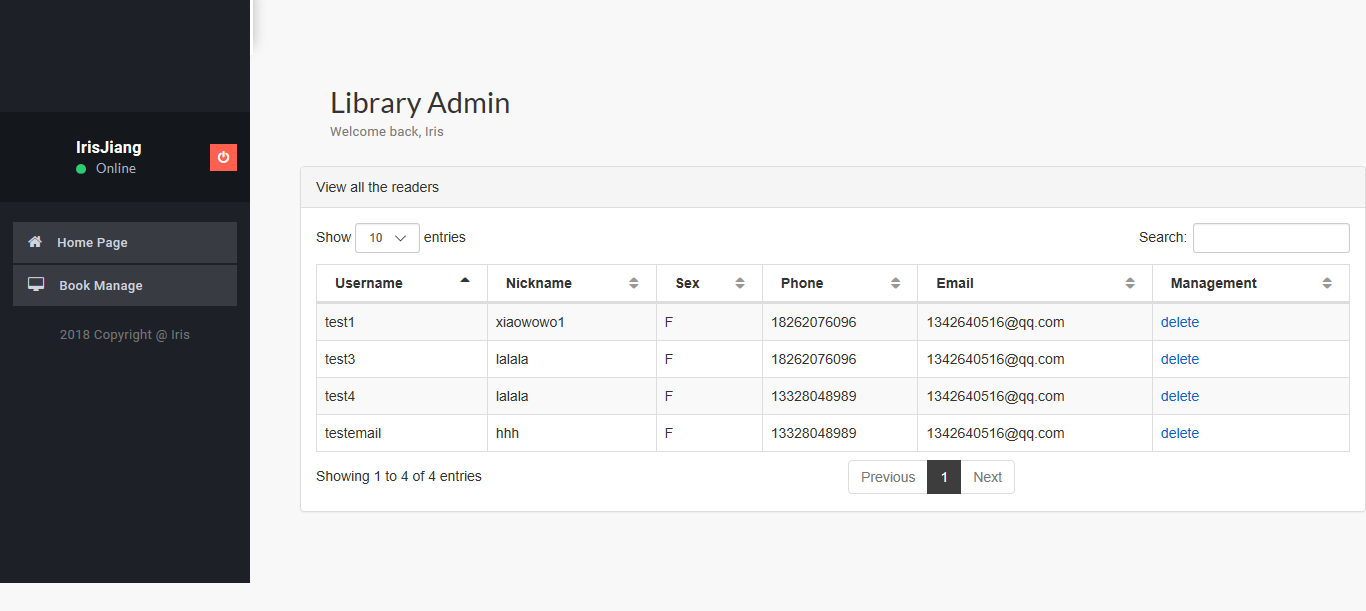
* 点击edit按钮，可编辑除了ISBN编号之外的其他图书信息



* 点击delete，可以删除图书
* 点击add new book，可以添加新书



* 点击reader manage，可以进入读者管理界面。可以对读者进行删除。（由于reader是borrow和comment的外键，删除读者的同时必须同时将User、Reader、Borrow、Comment中的相关记录删除）



## 数据库设计

**表结构**

**Book类：ISBN编号，书名，作者，出版社，豆瓣链接，分类，总数，剩余数**

Book（isbn，title，author，publisher，details，category，quantity，remain）

**Reader类：账户（由django自带User类映射而来），昵称，性别，电话号码，邮箱，头像**

Reader（account，nickname，sex，phone，email，icon）

**Borrow类：读者（由Reader类映射而来），ISBN编号（由Book类映射而来），借阅时间，到期时间（默认为借阅后两个月）**

Borrow（user，isbn，borr\_time，due\_time）

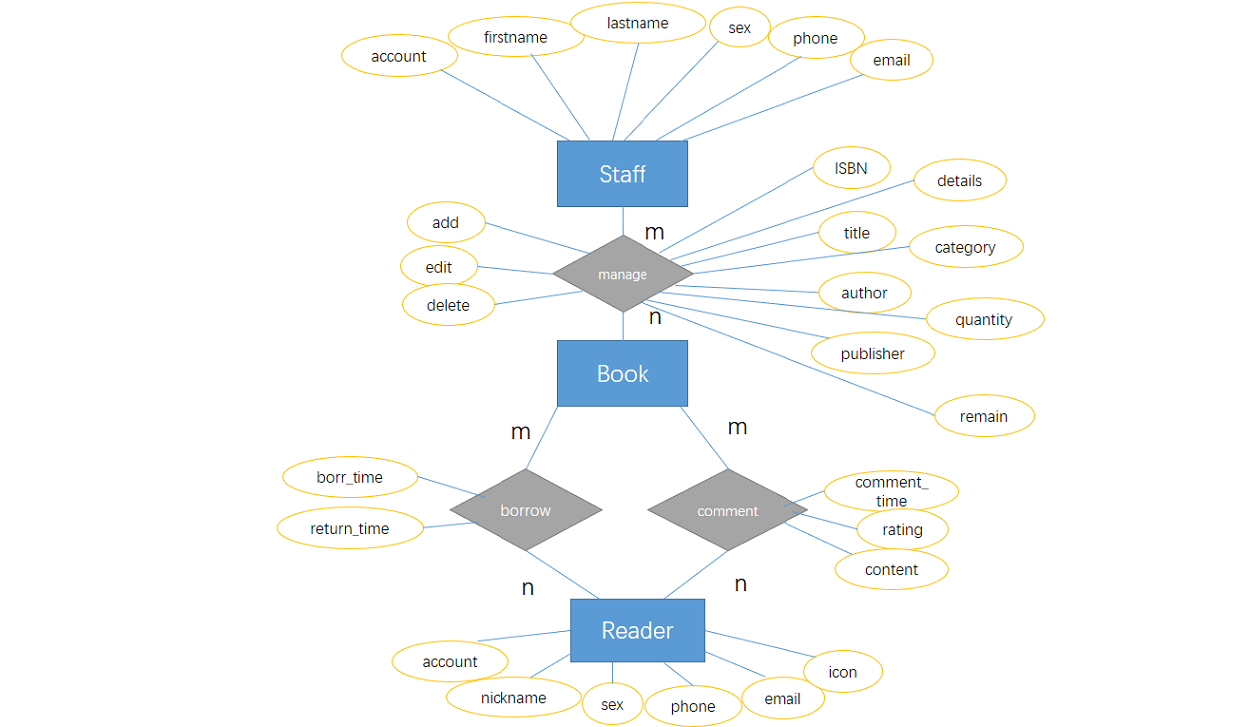
**Comment类：读者（由Reader类映射而来），ISBN编号（由Book类映射而来），评论时间，评分，评论内容**

Comment（user，isbn，comment\_time，rating，content）

**Staff类：账户（由django自带User类映射而来），名，姓，性别，电话号码，邮箱**

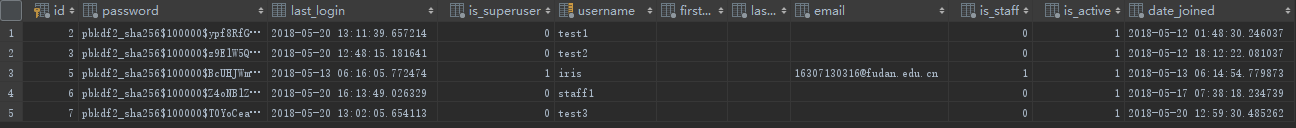
Staff（account，firstname，lastname，sex，phone，email）

**ER图**

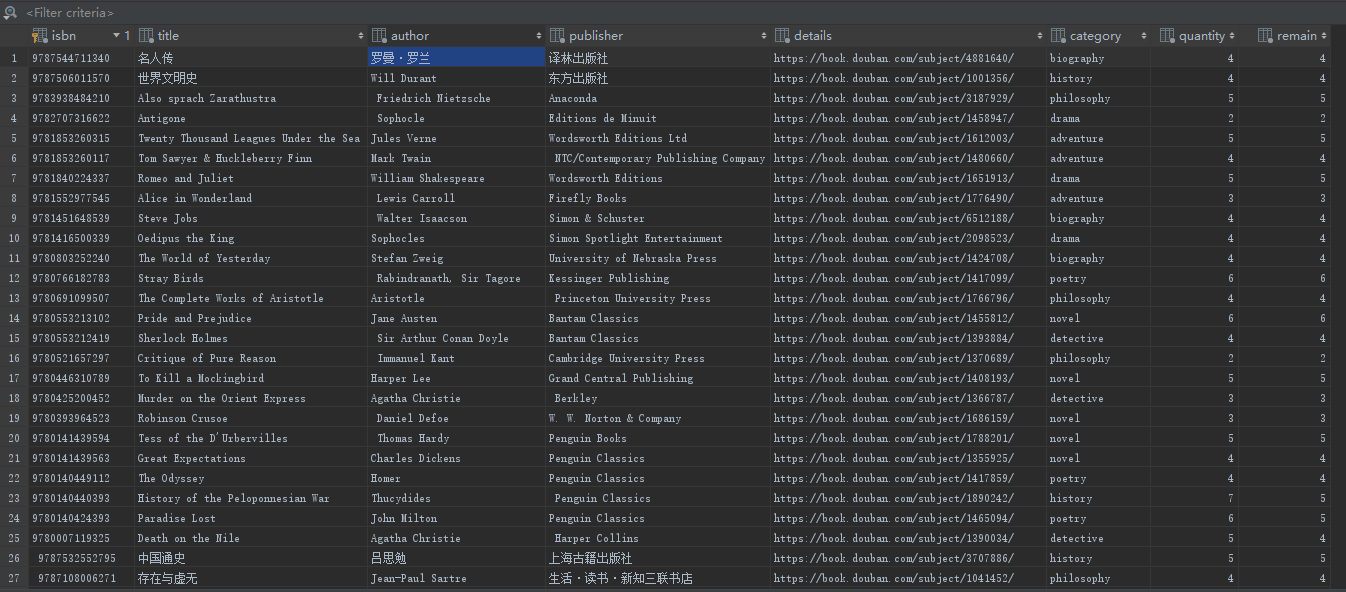


使用一段时间后数据库中的表：

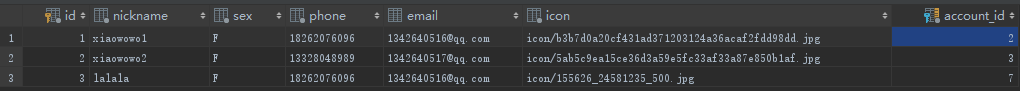
auth\_user



book



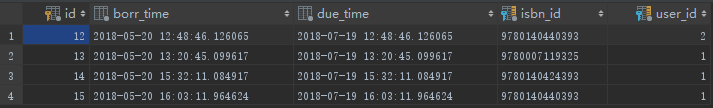
reader



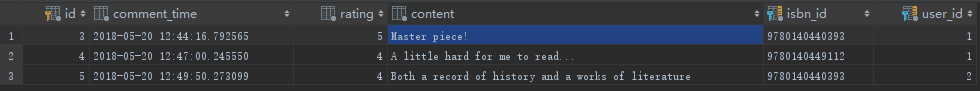
staff



borrow



comment



## 技术难点

遇到的困难：

1. 刚开始使用django的时候，第一次测试服务器，默认端口是8000，而当时我开着酷狗音乐，该端口被酷狗音乐占用而导致我多次尝试都报错，后来我每次启动服务器都指定8080端口。
2. 我原来对python、html、css的语法都不了解，可以说都是从头开始。往往都是先跟着教程里的实例走，通过自己的探索尝试来学习，最常见的问题就是各种报错。不过django的调试功能很强大，能给出报错的类型、具体位置，也能查看本地变量，为修改错误提供了便利。另外还要感谢CSDN、Stack Overflow、Segment Fault的帮助。
3. 一旦向数据库中加入了数据，migrate就变得很麻烦。我在删除了数据库中的数据之后，服务器仍然报错，原因是某个属性没有赋值，但数据库中其实是没有这个属性的。然后我就把migrate的记录文件全删了，导致整个崩溃，不得已重开了一个工程，代码复制过去之后就解决了。
4. django自带的user类非常方便，能较好地提供登录验证、session以及缓存的功能，但是属性不够全面，因此采用了OneToOneField映射的方式，对User类进行扩展，衍生出Reader类和Staff类。
5. 在实现修改用户密码的过程中，我的函数中需要用户输入原密码、新密码和确认密码，但是django自带的认证系统中密码不是存储明文而是存储一个Hash值，直接的字符串比较和存储都是会出错的，必须要注意。
6. 在使用bootstrap模板的过程中，学会了根据引用的class去css文件中寻找对应的样式进行自定义，但是还有很多时候怎么改都不太和谐，距离纯粹靠自己开发还有很大的一段距离。
7. DDL当天对系统的功能做了最后的丰富与优化：增添了续借功能，每次可以续借一个月；增添了忘记密码功能，利用QQ邮箱的SMTP服务，向用户注册时填写的验证邮箱中发送系统自动生成的密码用于重新登录；增添了图书管理员对用户的查看和删除操作。