# 系统设计说明书

第八组：梁爽 杨祥睿 黄英伦 杜家琛

1. **概述**
   1. **系统概述**

该图书推荐系统致力于为收集图书信息来为用户推荐最相关的书籍。对于用户，通过获取数据库用户信息、借阅记录、图书信息等原始数据，快速分析出用户兴趣图书的集合，完成个性化推荐功能。对于管理者，该系统能完成读者查询个人信息、图书评分信息检索等功能。

* 1. **需求概述**

**（1）功能需求**

图书推荐系统分为两个子系统，后台管理系统和客户端系统。后台管理系统由管理员操作执行，客户端由读者操作执行。

**后台管理系统：**

1.用户管理

2.信息导入

3.热门图书管理

4.分类热门图书管理

**客户端：**

1.读者账户管理

2.读者信息管理

3. 个性化推荐

4. 热门排行

**（2）性能需求**

1.普通情况下，用户从操作到得到系统反馈的时间最大不超过5秒，平均时间在1～3秒以内。

2.数据库至少能储存10000条数据记录，以确保能够录入足够大小的用户的阅读信息与图书量，保证程序正常运行。

3.数据库服务器能够承载至少1000人同时进行操作。

1. **接口需求**

用户界面要求：

程序将采用1920×1080分辨率的窗口模式运行。

**（4）其他需求**

1.安全性

2.高可用性

3.易用性

4.高性能

* 1. **参考文档**

CSDN：协同过滤算法（推荐算法）、归一化处理

道客巴巴：图书推荐系统的设计与实现

《C++ primer》[美]Stanley B.Lippman 著，王刚 杨巨峰 李忠伟改编

《计算机算法设计与分析》（第五版） 王晓东著

《数据库系统概念》(美国)希尔伯沙茨著

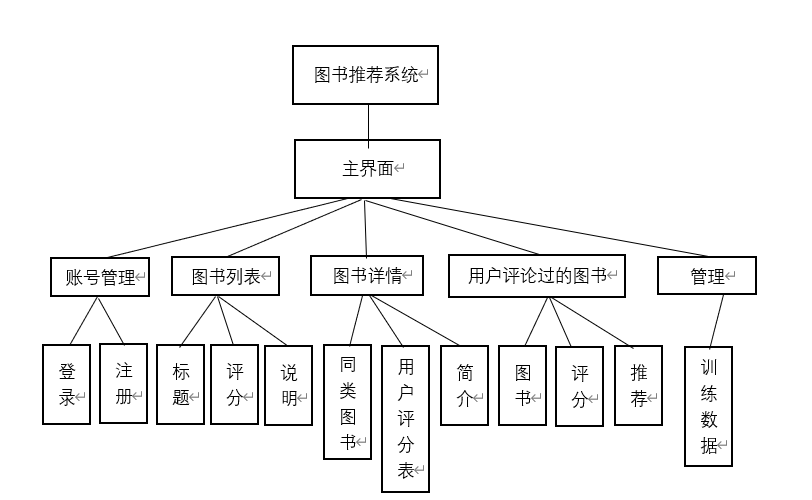
《Android应用开发揭秘》 杨丰盛著

《Android studio应用开发实战详解》 王翠萍著

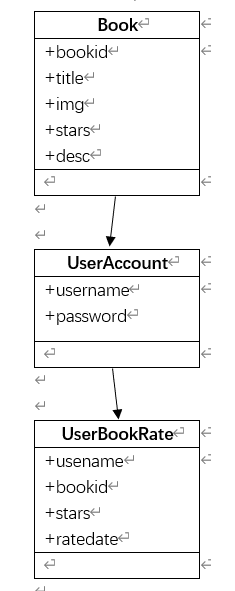
《MySQL技术内幕：SQL编程》 姜承尧著

1. **架构设计**
   1. **系统结构**

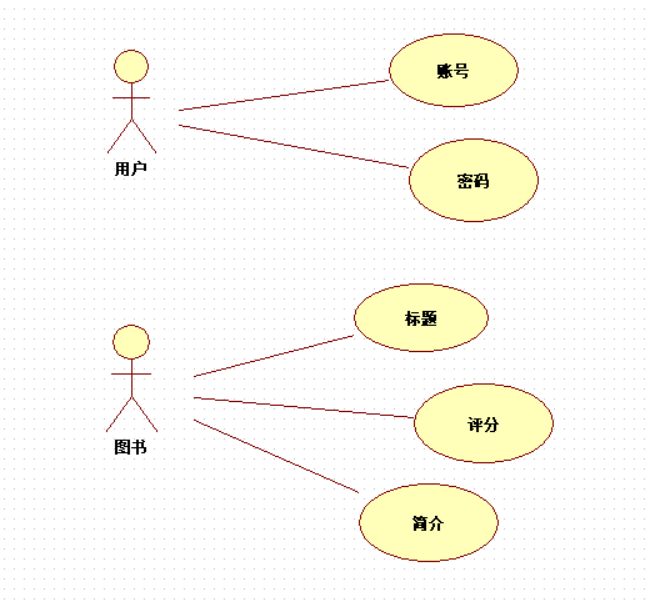
**功能模块图：**

****

**类图：**

****

**用例图：**

****

**2.2系统组成**

分为硬件和软件部分，硬件部分需求足够高的性能以及内存和磁盘空间用作系统运行和部署，软件部分则分为数据库，后端以及前端部分，数据库用于储存各类数据，后端用于接收衔接前端和数据库，用于处理需求和数据，前端用于可视化操作和输入操作。

**2.3硬件平台**

办公用笔记本电脑。CPU：i3-530以上，内存2G以上，硬盘容量足够大。

**2.4软件平台**

windows 7及以上操作系统，IE6.0版本及以上浏览器。

开发环境：Python

数据库系统：MySQL

开发工具：PyCharm

**2.5系统可靠性设计**

普通情况下，用户从操作到得到系统反馈的时间最大不超过5秒，平均时间在1～3秒以内。

系统能够保护用户信息，包括用户账号密码等。且能阻止外部侵入，防止数据库信息泄露或被窃取。

系统能够确保在一年内最多因故障停机5次，且每次停机时间不超过2小时。

系统能够在多个并发读写中保持稳定，保证数据库服务器不会崩溃。

1. **软件设计**

**3.1 页面设计**

**3.1.1 登录与注册**

登录与注册设计采用BootStrap的Navbar导航条，插入Button和Entry对象来创建按钮及输入框，并设置了绑定变量来获取输入框输入。

具体实现步骤为:

a.设置导航条，其中包括图书推荐系统、图书列表、用户评价后的图书、管理。

b.登录部分设置两个标签，分别命名为“用户名”、“密码”，放置两个输入框，两个按钮“登录”和“注册”。注册部分设置三个标签分别为“用户名”、“密码”，“确认密码”放置三个输入框，一个“注册按钮”。

c.密码的文本设置输入密码后显示\*号。

**3.1.2 图书列表部分**

在该部分，选择Web API类型的MVT，使用Django是一款python的web开发框架：与MVC有所不同，属于MVT框架。

m表示model，负责与数据库交互。

v表示view，是核心，负责接收请求、获取数据、返回结果。

t表示template，作用就是将内容反馈并呈现在浏览器上。

通过与数据库的交互，获取数据并返回结果到图书列表的界面上。

首先获取到图书的图片，通过book.title获取标题并将其显示在界面上，通过book.stars获取评分情况将其显示。通过book.desc得到图书简介。

**3.1.3图书详情部分：**

这部分最关键的部分是显示与此书相似的图书，对训练的结果中的相似图书按照一定的格式显示在页面上。判断是否曾经对其进行评分，如果评分过，显示之前的评分。反之，提示用户“您还没有评分过，给个评分吧”

**3.2 Python实现核心功能**

**3.2.1 跳转方式**

当用户在aw和bw未登录时，在SSO上设置登录态，那么在aw和bw上应该设置登录态。如上所述，还是应该在aw和bw上设置各自的登录态，这样在访问aw时首先会在aw域上检测授权，如果没有授权，则跳转到SSO进行登录授权。但是aw和bw应该为登录态一般设为浏览器进程存活期，即aw和bw的登录态的存活期直到浏览器关闭。SSO域上登录态的存活期取决于具体的业务，本系统中设为7天。代码如下：

app = Flask(name)

app.config[‘SECRET\_KEY’] = os.urandom(24)

# 服务器启动一次上次的session就清除，因为设置为随机产生的24位的字符，也就是说每次运行服务器都是不同的。

app.config[‘PERMANENT\_SESSION\_LIFETIME’] = timedelta(days=7)

# 设置session的保存时间。

**3.2.2 数据库连接**

首先创建数据库连接，打开数据库连接。代码如下：

import pymysql

def create\_connection():

db = pymysql.connect(“localhost”， “root”， “12345678”， “bookrecommend”)

return db

**3.2.3 ItemCF算法的实现**

1.ItemCF算法的实现是本图书推荐系统的核心内容，首先要计算出物品之间的相似度，相似度是推荐的关键评判指标，然后根据物品的相似度，并结合用户的历史行为，即那里评分矩阵，从而给用户生成可靠的推荐列表。

具体步骤：

首先建立物品的同现矩阵。

其次建立用户对物品的评分矩阵。

最后矩阵计算推荐结果。

def ItemSimilarity(train):

# 物品-物品的共同矩阵

C = dict()

# 物品被多少个不同用户购买

N = dict()

for u， items in train.items():

for i in items.keys():

N.setdefault(i， 0)

N[i] += 1

C.setdefault(i， {})

for j in items.keys():

if i == j:

continue

C[i].setdefault(j， 0)

C[i][j] += 1

2.计算问题之间的相似度

W = dict()

for i， related\_items in C.items():

W.setdefault(i， {})

for j， cij in related\_items.items():

W[i][j] = cij / math.sqrt(N[i] \* N[j])

return W

问题i与问题j之间的相似度这么定义：同时关注问题i与问题j的人数/关注问题i人数关注问题j的人数的平方根。

由此我们就需要计算n个问题之间两两相似度，就是一个对角线为1的对称邻接矩阵，也就是说，想要成功求出n个问题之间的相似度，至少需要计算n(n-1)/2次，这样才可以计算n个问题之间的两两相似度。

推荐前K个用户：

def Recommend(train， user\_id， W， K):

rank = dict()

if user\_id not in train: return []

action\_item = train[user\_id]

for item， score in action\_item.items():

for j， wj in sorted(W[item].items()， key=lambda x:x[1]， reverse=True)[0:K]:

if j in action\_item.keys():

continue

rank.setdefault(j， 0)

rank[j] += score \* wj

return sorted(rank.items()， key=lambda x:x[1]， reverse=True)

1. **数据设计**

根据系统业务和DBMS的需求，建立了最佳的数据存储模型。另外，通过建立数据库内的表结构与表与表之间的关系的处理，能够有效地将数据存储到应用系统中，高效访问存储的数据。好的数据库设计需要以下部分：

减少数据冗余；

避免数据维护异常；

节约存储空间；

高效的访问；

需求分析。

同时需要分析数据和属性各自的特点，以便了解系统中所要存储的数据、了解数据的存储特点、了解数据的存储周期。需求分析中需要了解的问题是实体之间的关系、包含的属性。

该系统采用MySQL数据库，保存用户的用户名、密码等数据。

数据库的主要表如下：

用户（账号，密码）如表1所示。

图书（标题、评分、简介）如表2所示。



1. **总结**

本文首先阐述了推荐系统开发的经过，分析了推荐系统目前的研究状况。其次，阐述了推荐结果的主要推荐方法和评估指标。最后分析了主流的推荐算法以及它们各自的优缺点，尤其是基于物品的协同过滤算法。

推荐系统的开发，一方面使用户和信息精确一致，另一方面降低信息过载时的信息获得成本。

目前，深层神经网络发展迅速，为推荐系统提供了新的思路，例如特征提取和排序法。现在越来越多的推荐引擎将传统的推荐算法与深层神经网络结合，以解决数据分区和推荐排名问题。深层神经网络与推荐系统的组合是今后推荐系统的研究课题。

总而言之，推荐系统是一个巨大的信息系统。它依赖于引擎工作以及业务系统、日志系统和许多其他方面。同时，它结合了网络安全和数据挖掘等诸多研究领域。这可以为企业和用户带来值得详细研究的价值，因此，对此领域需要更深入的研究。

而随着当代社会突飞猛进的发展，在可预见的将来，数据与信息的量级只会越来越大。彼时，对信息筛选的需求也会日益增长，推荐系统将会在未来造成越来越大的影响，对推荐系统的研究也将达到新的高度。

在一边进行学习，一边进行系统开发和设计过程中，遇到了很多短时间无法解决的问题，走了很多弯路，但是，这次课题研究锻炼了团队解决问题的能力，提高了编程能力，锻炼了逻辑思维和整体设计的能力，收获颇多。由此我们意识到完整的软件开发思路是非常重要的，它关系到我们开发过程能否顺利实现。在项目开始之前，必须有软件工程的系统化的知识体系，不仅如此，还应包括软件结构位置、代码编写和模块划分和整体布局，只有做完这些步骤，才能逐步开发出想要实现的功能。这样定能达到事半功倍的效果。