# 1、需求说明

## 所需要开发的软件系统边界

## 系统所有的相关及使用人员角色

## 系统关键的使用场景

及时提醒用户自己复习读过的书，这样复习21次后，把知识变成自己的。

## 系统规模、性能要求以及部署方式等非功能性需求

用户数：100W；

四个量：

1. 请求量：100W/天；峰值：1W/s；
2. 流量：100M/天；峰值：1M/s；
3. 包量：200W/天；峰值：1W/s；
4. 容量：1G；

时延：1s；

成功率：99.9%；

单点部署；

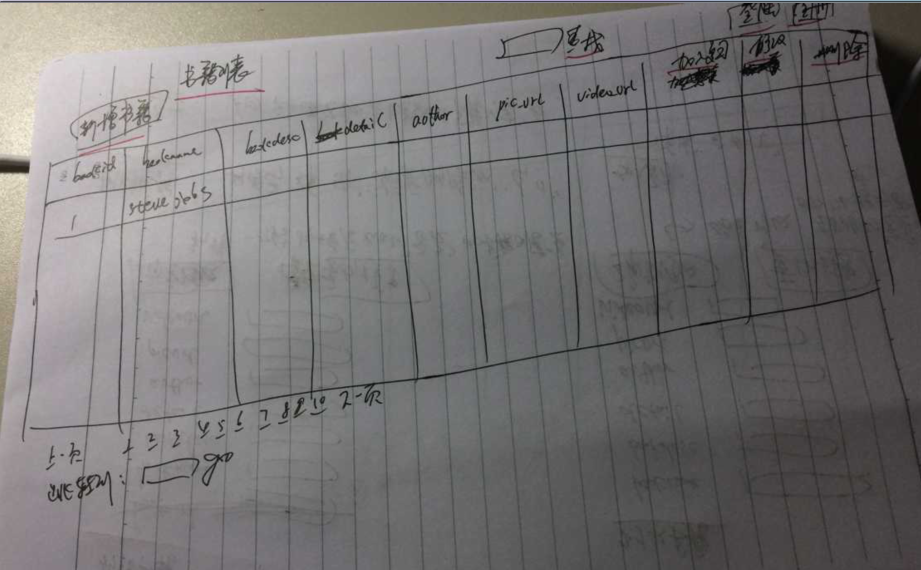
# 2、功能设计

### 书籍列表页（首页）

展示书籍列表，右上方有个“登录”、“注册”按钮。

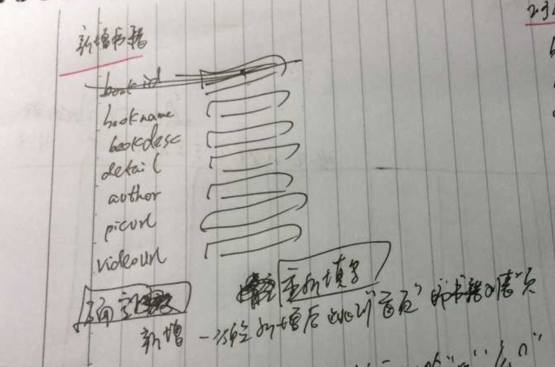
功能：

1. 新增书籍；
2. 删除书籍；
3. 修改书籍；
4. 查询书籍；
5. 加入review



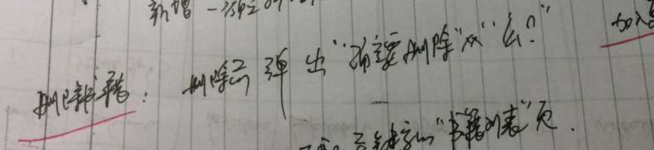
#### 新增

入口：书籍列表页；点击“新增书籍”弹出书籍登记页面；登记完成点击“确定新增”，弹出“操作成功”，给出首页链接；如果新增失败，弹出“操作失败，错误提示语”，给出首页链接。



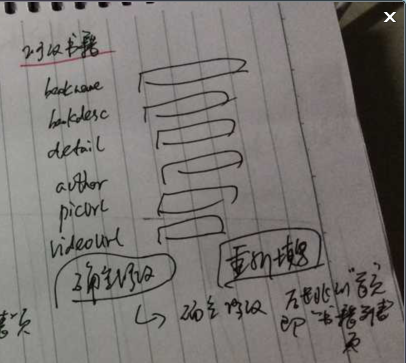
#### 删除

入口：书籍列表页；点击“删除”弹出删除提示。



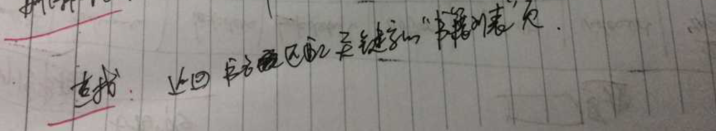
#### 修改

入口：书籍列表页；点击“修改”跳转到书籍编辑页面。



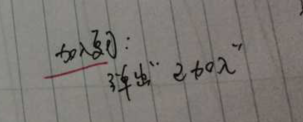
#### 查询

入口：书籍列表页；搜索框中输入关键字，返回匹配书名列表。



#### 加入review

入口：书籍列表页

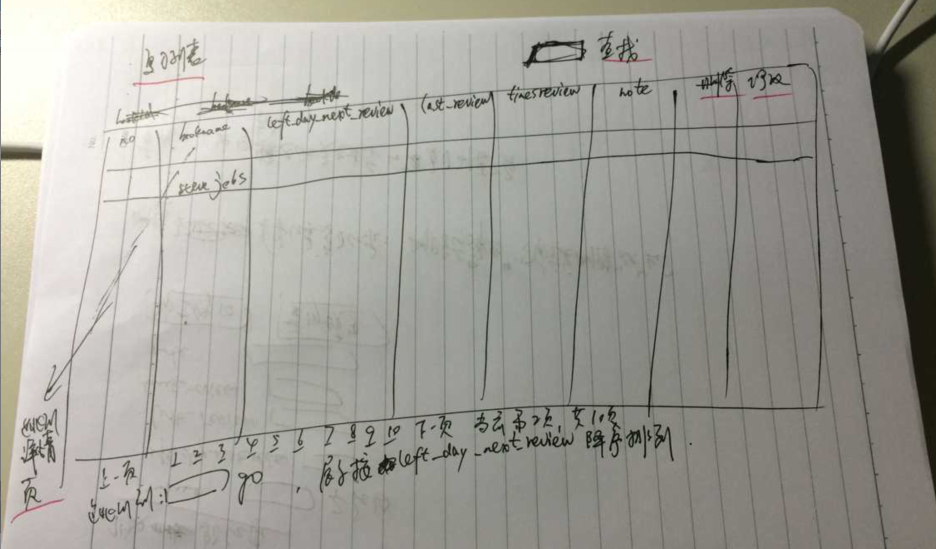


### Review列表页

展示用户复习列表。

功能：

1. 删除；
2. 修改；
3. 查找；

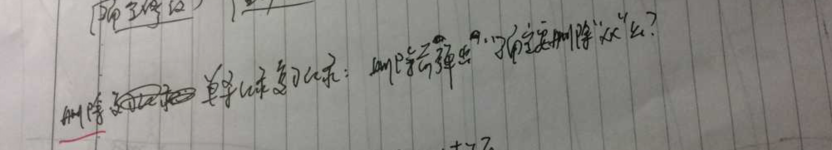


#### 新增

入口：书籍列表页，点击“加入复习”；

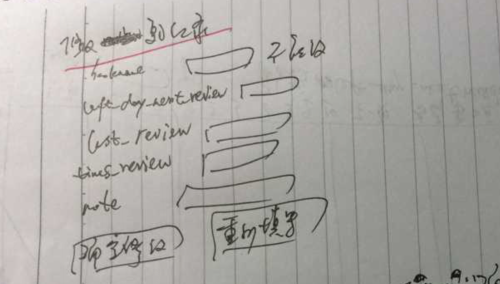
#### 删除

入口：review列表页；



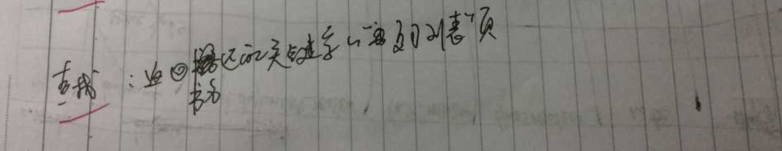
#### 修改

入口：review列表页



#### 查询

入口：review列表页；



#### 普通用户

登录后，展示“我的review 列表页”tab，“想读”tab

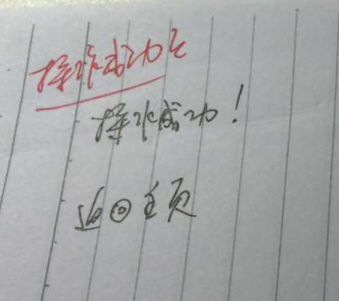
书本详情

加入想读列表

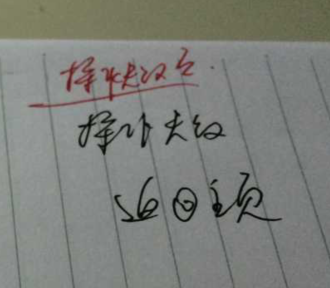
加入review列表

#### 管理员

### 操作成功页



### 操作失败页



### 管理员登录

查询用户：展示用户列表

对单个用户的编辑；

通知概要

通过详情

# 3、系统架构设计

## 系统开发视图及技术路线选择

## 系统逻辑视图



## 系统部署视图

## 系统模块视图

## 系统的领域模型

# 4、模块概要设计

## 模块实现机制设计

## 数据库表设计

涉及2个实体：

1. 书；
2. 用户。

“书-用户”表:table\_book\_review

|  |  |  |
| --- | --- | --- |
| 列名 | 类型 | 备注 |
| user\_id | Int unsigned | 用户id，primary key(personid, bookid), foreign key(table\_user.user\_id) |
| book\_id | int unsigned | 书id, foreign key(table\_book.book\_id) |
| last\_review | int | 上次复习时间 |
| times\_reviewed | int | 已复习次数 |
| left\_days\_next\_review | int | 距离下次复习还有几天 |
| note | text | 读书笔记 |
| reserv1 | blob | 保留字段1 |
| reserv2 | blob | 保留字段2 |
| reserv3 | blob | 保留字段3 |
| reserv4 | blob | 保留字段4 |
| reserv5 | blob | 保留字段5 |
| version | int | 版本号，不一样代表的reserv字段的值不同，为以后新特性做准备 |

“书”表:table\_book

|  |  |  |
| --- | --- | --- |
| 列表 | 类型 | 备注 |
| book\_id | Int unsigned | 书id，primary key auto\_increment |
| book\_name | varchar(50) | 书名，not null |
| book\_desc | varchar(100) | 书描述 |
| detail | text | 详情 |
| author | varchar(100) | 作者 |
| pic\_url | varchar(200) | 书的照片url |
| video\_url | varchar(200) | 书的视频url |
| reserv1 | blob | 保留字段1 |
| reserv2 | blob | 保留字段2 |
| reserv3 | blob | 保留字段3 |
| reserv4 | blob | 保留字段4 |
| reserv5 | blob | 保留字段5 |
| version | int | 版本号，不一样代表的reserv字段的值不同，为以后新特性做准备 |

“用户”表：table\_user

|  |  |  |
| --- | --- | --- |
| 列名 | 类型 | 备注 |
| user\_id | int unsigned | 用户id |
| name | varchar(50) | 用户名 |
| sexual | char(1) | 性别, not null |
| age | smallint | 年龄, >0 |
| reserv1 | blob | 保留字段1 |
| reserv2 | blob | 保留字段2 |
| reserv3 | blob | 保留字段3 |
| reserv4 | blob | 保留字段4 |
| reserv5 | blob | 保留字段5 |
| version | smallint | 版本号，不一样代表的reserv字段的值不同，为以后新特性做准备 |
| pass | varchar(200) | 用户密码, not null |

**教训**：

1、“用户”表必须包含密码和用户的基本信息，如age。其他地方的user\_id全部都是引用“用户”表的user\_id，如：有新的业务需要加到额外的用户信息，如现在有了买书的业务，需要增加一个表，即user\_id对应一个买书等级的表，以后有了看电影的业务，如有了买电影票、看电影签到的等级，又需要增加一个表，即user\_id对应一个看电影的表。

2、先建立表，建立多个表时考虑先后关系，如foreign key；当设定好先后关系再各个击破，即考虑一个表，一个表需要考虑：

表名；

字段名；

主键；

类型；

约束条件；

默认值；

是否auto\_increment；

## 模块接口设计

## 关键类设计

## 画出时序图

## 交互图等。

# 5、模块详细设计