

北京林业大学

23 学年—24 学年第 1 学期 JavaWeb 技术 实习报告书

专 业: 计算机科学与技术 班 级: 21-1

姓 名: 李嘉梁 学 号: 211302104

实习地点: 信息中心机房 辅导教师: 王新阳

实习题目: 通过编程实现一个真实的 Java Web 项目案例, 掌握基于 MVC 架构设计开发 B/S 网站的基本原理和方法。具体包括: 开发和运行框架的选择和搭建、业务需求的分析、业务控制流程的设计和实现、数据库访问的设计实现以及良好的人机交互界面等。

实习环境:

数据表现层: Html+CSS+JavaScript+JQuery+JSP

业务逻辑层: Java+Spring+SpringMVC

数据持久层: MySQL+MyBatis

开发工具: IntelliJ IDEA 2021.3.2 (Ultimate Edition)

项目管理工具: Maven-3.6.3

Java Web 服务器: Tomcat-9.0.24

实习内容:

(一) 系统需求分析与概要设计

1. 系统用户角色和功能分析设计:

1.1. 系统角色说明

该系统包含三类用户角色:

- (1) 系统管理员(Admin) (管理员账号密码登录, 内置用户, 用户名 admin, 密码 123)
- (2) 工作人员(staff), 即图书管理人员 (通过工号和密码登录, 由管理员添加)
- (3) 读者(reader) (有用户自己注册账户密码)

对三类用户的统一要求:

- ① 均支持登录、重置密码功能 (邮件、短信或回答问题等形式需要具体自己设计), 并且登录成功后具有修改自己信息的功能。
- ② 登录成功后, 页面右上方显示出用户名。点击用户名后, 可查看自身基本信息并进行编辑修改。点击“退出系统”后登出本系统, 且不允许再次通过直接访问地址的方式打开系统页面。
- ③ 用户登录过程中, 应该具有基本校验功能, 包括用户是否存在、用户名与密码是否匹配等; 若登录失败, 则要给出登录失败提示信息。

1.2. 管理员角色功能说明

(1) 用户管理

① 创建新用户：

输入工作人员基本信息后创建一名新的工作人员用户，其中用户名为工作员工号，密码工号后四位。用户信息至少应包含用户名、密码、真实姓名、联系电话、邮箱、住址等信息。创建用户的同时，可以指派用户所属的工作单位。

② 查询/修改用户信息：

用户管理页面以表格列表的形式显示所有用户信息。在最右侧一列设置“操作”列，包含“详情”、“修改”、“删除”三个操作，可以分别查看用户详情、修改用户信息、删除用户。

(2) 图书管理

点击进入页面后以表格列表的方式显示所有图书。在最右侧一列设置“操作”列，包含“详情”、“修改”、“删除”三个操作，可以分别查看图书详情、修改图书信息、删除图书。**系统管理员不进行图书入库操作**。在删除图书之前，需检查图书的借阅状态，尚未归还的图书需归还后才能删除。

(3) 单位管理

① 新建单位：可以创建一个单位信息，应至少包括单位名称、联系人、联系电话、邮箱地址、联系地址等信息；

② 单位信息维护：点击进入页面后以表格列表的方式显示所有单位信息。在最右侧一列设置“操作”列，包含“详情”、“修改”、“删除”三个操作，可以分别查看单位详情、修改单位信息、删除单位。在页面上方设置创建按钮，可以新建一个图书管理单位，具体单位信息参见数据库表；

③ 单位检索：可以在检索框中单位名称，进行模糊检索。

1.3. 工作人员角色功能说明

工作人员用户，主要五类操作模块：图书管理、图书流通、流入流出管理、借出与归还管理、统计分析管理。其中借入借出管理又具体分为借入管理和借出管理。

(1) 图书管理

由工作人员进行图书信息的入库、修改和维护等。**工作人员只能管理本单位的图书**。

① 图书信息入库

同时支持以下两种入库方式：

- 支持手工录入方式：将图书的基本信息录入到管理系统中，需录入的图书基本信息至少应包括以下信息：图书名、图书编号、出版时间、作者、出版社、图书分类、页数、价格等信息，同时支持上传图书图片到服务器中。
- 支持批量导入方式：可以从指定模板的 excel 表格中批量导入数据库中。

② 图书查看

- a) 图书列表：页面以表格列表的形式显示所有图书信息，至少包含图书名、图书编号、出版时间、作者、出版社、图书分类、页数、价格、借阅状态等信息；
- b) 缩略图视图：支持用户选择以表格列表或视图列表的方式切换
- c) 图书详情：点击列表中的详情按钮，可以查看当前图书的详细信息；
- d) 图书查询：在页面上方设置查询功能，当用户输入查询关键词后，通过模糊匹配查询到相应结果。至少支持工作人员按照图书名、图书编号、作者、出版社四个属性进行查询。

③ 图书维护

在图书列表最右侧一列设置“操作”列，可以对图书进行“修改”和“删除”操作。

- a) 图书信息修改：可以对图书的各项基本信息进行修改后重新提交；
- b) 图书删除：可以将指定的图书从图书库中删除。在删除图书之前，需检查图书的借阅状

态，尚未归还的图书需归还后才能删除。

④ 图书开放与隐藏

只有开放查看权限的图书才能被外单位用户查看、借阅，或者被注册的读者查看。被隐藏的图书只有本单位用户有查看权限。

(2) 图书流通

工作人员可以查阅其他单位图书列表，并向对方单位发起流通申请。

① 查阅其他单位图书：

- a) 可以查看其他单位的图书，展示形式同“图书查看”功能类似，但不允许对其他单位的图书进行编辑和删除；
- b) 在查阅列表最右侧一列增加“流通”操作，选择要借阅的图书并点击“流通”后进入流通页面，输入申请人的联系方式、地址等信息后提交，等待被流通单位工作人员的审核。

(3) 图书流入、流出管理

工作人员既可以对从其他单位流通进入的图书进行管理，也可以对其他单位提出的流出请求进行管理。

① 流入管理：

由本单位工作人员向外单位发起流入请求，根据流入请求被处理的情况，分为三种状态：

- a) 待审核：已发起流入请求，等待对方单位审核；
- b) 已审核：对方单位已审核流入请求，当前图书流入成功或被拒绝流出；在该状态下，工作人员可以点击“归还”发起归还请求，由对方单位审核后结束本次流入。
- c) 已结束：被拒绝流入的图书直接进入结束状态；成功流入的图书被归还后，表示本次流入结束，同样进入结束状态。

② 流出管理：

本单位工作人员审核外单位的流出请求，符合流出要求的通过流出审核，否则拒绝流出，同样分为三种状态：

- a) 待审核：对方单位已发起流出请求，等待工作人员审核，审核结果分为通过和不通过两种；
- b) 已流出：已通过本单位工作人员审核的被成功流出的图书列表。在该列表下可以审核归还请求，由申请人发起归还请求，工作人员线下检查无误后，线上操作“已返还”完成本次流出，该图书又变为可流出状态；
- c) 已结束：被拒绝流出或已归还的图书进入已结束状态。

(4) 图书借出与归还管理

本单位工作人员还可以管理图书的借阅情况，读者发起借阅请求后，由工作人员对借阅记录进行审核，审核通过后读者即可从当前单位借出本书，读者借完图书后，记录归还情况，修改图书借阅状态，该流程类似“流出管理”功能。

(5) 统计分析管理

对本单位管理的图书进行统计分析，至少包括以下统计信息：

- 本单位总图书数；
- 当前在库总图书数；
- 近年借阅情况统计（按年、按月借阅的折线图、柱状图等）

1.4. 读者角色功能说明

对于读者用户，只有注册登录账户后，才可以查看系统中的图书信息。

(1) 账户注册与管理

为游客用户提供账户注册功能，并能修改用户信息。

① 账户注册：

- a) 游客可以点击注册链接，注册个人信息后登录系统，基本信息至少包括：用户名、密码、真实姓名、联系电话、邮箱、住址。
- b) 用户注册时，要进行重复注册、用户名合法性等进行基本校验；对邮箱地址、联系电话格式的有效性进行校验。

(2) 图书信息查看

游客身份用户只能进行图书查看，不允许进行任何操作。可以查看所有单位公开查看权限的图书信息。

(3) 图书借阅与归还

对于感兴趣的图书，读者可以向图书所有单位发起借阅请求，由工作人员审核通过后即可借走图书；阅读完毕后可申请归还图书，由工作人员进行还书审核并登记后结束本次借阅，该流程类似工作人员角色中的“流入管理”功能。

2. 数据库设计：

(1) 用户表：<用户 ID (PK)、用户名、密码、真实姓名、联系电话、邮箱、住址、工作编号、是否注册、创建时间、用户性别、头像路径、账号状态、上次登录时间，工作单位 (FK)>；

字段名	数据类型	长度	NOT NULL
UserID (PK)	int	11	True
UserName	varchar	255	False
Password	varchar	255	False
RealName	varchar	255	False
ContactNumber	varchar	15	False
Email	varchar	255	False
Address	varchar	255	False
WorkID	int	11	False
IsRegistered	tinyint	1	False
CreatedTime	timestamp	0	True
Gender	varchar	10	False
AvatarPath	varchar	255	False
AccountStatus	varchar	20	False
LastLoginTime	timestamp	0	False
UnitID (FK)	int	11	False

(2) 角色表: <角色 ID (PK)、角色名、创建时间、角色状态、备注>;

字段名	数据类型	长度	NOT NULL
RoleID (PK)	int	11	True
RoleName	varchar	255	False
CreatedTime	timestamp	0	True
RoleStatus	varchar	20	False
Remarks	text	0	False

(3) 用户角色表: <用户 ID (PK) (FK), 角色 ID(FK)>;

字段名	数据类型	长度	NOT NULL
UserID(PK) (FK)	int	11	True
RoleID(FK)	int	11	True

(4) 图书信息表: <图书编号 (PK)、图书名、出版时间、作者、出版社、图书分类、页数、价格、

图书图片存放路径、本数、所属单位 (FK)>;

字段名	数据类型	长度	NOT NULL
BookID(PK)	int	11	True
BookName	varchar	255	False
PublicationDate	date	0	False
Author	varchar	255	False
Publisher	varchar	255	False
BookCategory	varchar	255	False
PageCount	int	11	False
Price	decimal	10	False
ImagePath	varchar	255	False
BookCount	int	11	False
UnitID(FK)	int	11	False

(5) 图书流通表: <流通 ID (PK)、图书 ID(FK)、所属单位、流通账户 ID (FK)、流通时间、归还时间、流通理由、申请人、申请人联系方式、备注、应还时间、流通状态>;

字段名	数据类型	长度	NOT NULL
CirculationID(PK)	int	11	True
BookID(FK)	int	11	False
Unit	varchar	255	False
CirculationAccountID(FK)	int	11	False
CirculationTime	timestamp	0	True
ReturnTime	timestamp	0	False
CirculationReason	varchar	255	False
Applicant	varchar	255	False
ApplicantContact	varchar	15	False
Remarks	text	0	False
DueTime	timestamp	0	False
CirculationStatus	int	11	False

(6) 图书借阅表: <借阅 ID (PK)、图书 ID(FK)、借阅账户 ID (FK)、借阅时间、归还时间、借阅理由、借阅人、借阅人联系方式、备注、应还时间、借阅状态>

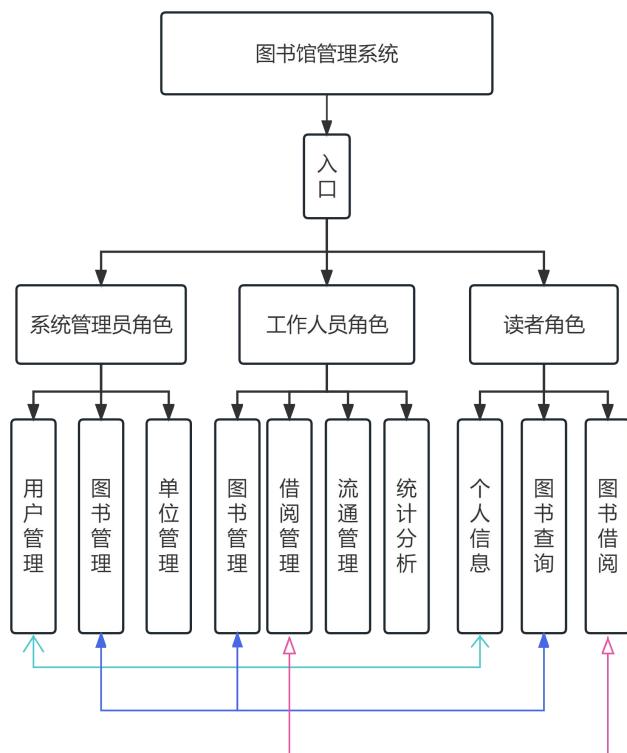
字段名	数据类型	长度	NOT NULL
BorrowID(PK)	int	11	True
BookID(FK)	int	11	False
BorrowAccountID(FK)	int	11	False
BorrowTime	timestamp	0	True
ReturnTime	timestamp	0	False
BorrowReason	varchar	255	False
Borrower	varchar	255	False
BorrowerContact	varchar	15	False
Remarks	text	0	False
DueTime	timestamp	0	False
BorrowStatus	int	11	False

(7) 单位表: <单位 ID (PK), 单位名称、联系人、联系电话、邮箱地址、联系地址, 单位性质>

字段名	数据类型	长度	NOT NULL
UnitID (PK)	int	11	True
UnitName	varchar	255	False
ContactPerson	varchar	255	False
ContactNumber	varchar	15	False
EmailAddress	varchar	255	False
ContactAddress	varchar	255	False
UnitType	varchar	255	False

3. 系统架构设计:

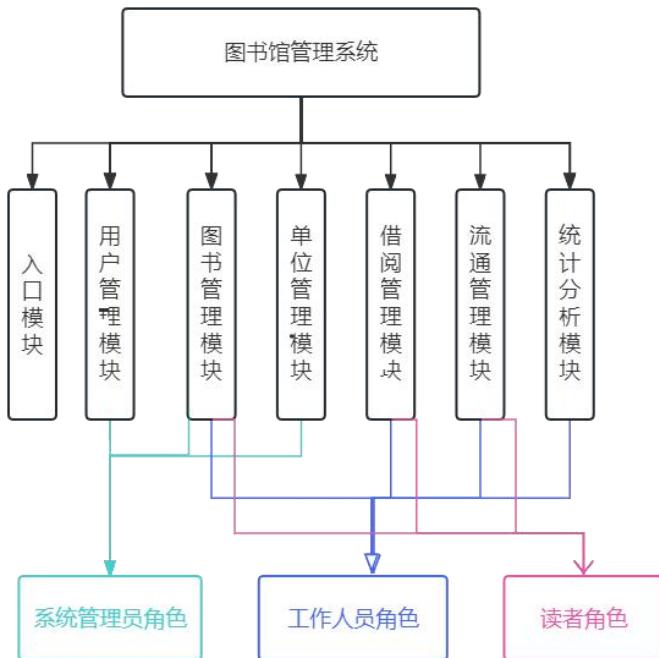
通过系统用户角色到功能的映射关系，我使用自顶向下方法，站在使用者的角度，设计出如下系统架构，每个角色虽然功能不同，但很多功能实现上是可复用的，只是权限分配不同。下方同色彩线指向了实现上可复用但是权限分配上有差异的功能。



4. 系统功能模块设计:

通过系统架构的设计推出的结论: `很多功能实现上是可复用的，只是权限分配不同`，可以进一步推出: 可复用的功能可以被归纳到同一功能模块中统一开发，得到的全部权限再差异性地分配到不同角色。这样做的好处就是: 从开发的角度看，可以大大减少开发的重复工作，使得开发逻

辑更清晰；从程序的角度看，不同业务逻辑被归纳为同一模块，降低了冗余度，每个模块可以独立开发，降低了耦合度。于是，我使用自底向上方法，站在开发者的角度，设计出如下功能模块，下方异色彩线有差异地将同一模块的不同权限分配给不同角色。



(二) 详细设计与关键技术

1. 入口模块:

1.1. MVC 结构设计:

M:

User:

```

User.java(com.lms.entity.)
UserService.java(com.lms.service,com.lms.service.impl.)
UserMapper.java(com.lms.mapper.)
UserMapperImpl.java(com.lms.mapper.impl.)
UserMapper.xml(resources/mapper/),
user(lms.) # 数据库表
  
```

Role:

```

Role.java(com.lms.entity.)
RoleService.java(com.lms.service,com.lms.service.impl.)
RoleMapper.java(com.lms.mapper.)
RoleMapperImpl.java(com.lms.mapper.impl.)
RoleMapper.xml(resources/mapper/),
role(lms.) # 数据库表
  
```

UserRole:

```

Userrole.java(com.lms.entity),
UserroleService.java(com.lms.service,com.lms.service.impl.),
  
```

*UserroleMapper.java(com.lms.mapper.),
UserroleMapperImpl.java(com.lms.mapper.impl.),
UserroleMapper.xml(resources/mapper/),
userrole(lms.) # 数据库表*

V:

entry[login.jsp,register.jsp,reset.jsp](webapp/WEB-INF/view/),

C:

*UtilController.java(com.lms.controller.),
UserController.java(com.lms.controller.),
OnlineFileter.java(com.lms.util.),*

1.2. 功能设计：

- 1.2.1. 登陆：前端对应 login.jsp；前端使用 jQuery 判断输入值合法与否，使用 Ajax 技术异步通信鉴权——post 后端接口 @RequestMapping("login")，后端通过 userService.selectByMap 使用 SQL 查询用户名和密码相匹配的账号，前端通过返回值得知用户是否存在并反馈给用户；
- 1.2.2. 注册：前端对应 register.jsp；前端使用 jQuery 通过正则表达式判断输入值合法与否，使用 Ajax 技术异步通信鉴权——post 后端接口 @RequestMapping("addUser")，后端通过 userService.selectByUsername 使用 SQL 查询用户名是否存在，如果不存在则通过 userService.addSelective 添加用户，前端通过返回值得知操作结果并反馈给用户；
- 1.2.3. 重置密码：前端对应 reset.jsp；前端使用 jQuery 判断输入值合法与否，使用 Ajax 技术异步通信鉴权——post 后端接口 @RequestMapping("resetUser")，后端通过 userService.selectByUsername 使用 SQL 查询用户名相匹配的账号，并判断答案正确与否，如果正确则密码重置，前端通过返回值得知操作结果并反馈给用户；
- 1.2.4. 登陆过期：使用 filter 技术——在进入登录后系统内每个网页（url 都以.do 为后缀）前都进行再次鉴权，检查 session 中的账号是否失效，如果失效则返回登录；

2. 用户管理模块：

2.1. MVC 结构设计：

M:

User:

*User.java(com.lms.entity.),
UserService.java(com.lms.service,com.lms.service.impl.),
UserMapper.java(com.lms.mapper.),
UserMapperImpl.java(com.lms.mapper.impl.),
UserMapper.xml(resources/mapper/),
user(lms.) # 数据库表*

Role:

*Role.java(com.lms.entity.),
RoleService.java(com.lms.service,com.lms.service.impl.),
RoleMapper.java(com.lms.mapper.),
RoleMapperImpl.java(com.lms.mapper.impl.),
RoleMapper.xml(resources/mapper/),
role(lms.) # 数据库表*

UserRole:

```
Userrole.java(com.lms.entity.),
UserroleService.java(com.lms.service,com.lms.service.impl.),
UserroleMapper.java(com.lms.mapper.),
UserroleMapperImpl.java(com.lms.mapper.impl.),
UserroleMapper.xml(resources/mapper/),
userrole(lms.) # 数据库表
```

V:

user[add.jsp,detail.jsp,list.jsp,update.jsp](webapp/WEB-INF/view/),

C:

UserController.java(com.lms.controller.),

2. 功能设计:

2.2.1. 用户列表:

对应后端接口 @RequestMapping("userList.do"), 后端通过 userService.selectByMap 使用 SQL 查询符合条件的用户; 前端也可以在进入页面后通过 jQuery 实时筛选现有列表中用户名符合条件的用户 (实时模糊查询); 前端对应 list.jsp;

2.2.2. 用户详情:

对应后端接口 @RequestMapping("toUserDetail.do"), 后端通过 userService.selectByUserId 使用 SQL 查询对应 userid 的用户; 前端对应 detail.jsp;

2.2.3. 用户修改:

对应后端接口 @RequestMapping("toUpdateUser.do") 和 @RequestMapping("updateUser"), 分别对应进入修改页面前的获取数据和进入页面后的修改数据, 后端通过 userService.selectByUserId 使用 SQL 查询对应 userid 的用户, 通过 userService.updateSelective 使用 SQL 修改对应 userid 的用户; 前端对应 update.jsp;

2.2.4. 用户删除:

对应后端接口 @RequestMapping("deleteUser"), 后端先通过 userroleService.deleteByUserId 删除 userrole 表记录, 再通过 userService.deleteByUserId 删除 user 表记录;

2.2.5. 用户添加:

前端对应 add.jsp, 通过正则表达式判断输入值合法与否;

对应后端接口 @RequestMapping("addUser"), 后端先通过 userService.selectByUsername 使用 SQL 查询用户名是否存在, 如果不存在则通过 userService.addSelective 添加用户;

批量导入: 对应后端接口 @RequestMapping("importExcelForUser"), 通过解析 Excel 表格的工具类 POI 实现

3. 图书管理模块:

3. 1. MVC 结构设计:

M:

BookInformation:

```
Bookinformation.java(com.lms.entity.),
BookinformationService.java(com.lms.service,com.lms.service.impl.),
BookinformationMapper.java(com.lms.mapper.),
BookinformationMapperImpl.java(com.lms.mapper.impl.),
BookinformationMapper.xml(resources/mapper/),
```

bookinformation(lms.) # 数据库表

V:

bookinformation[add.jsp,detail.jsp,list.jsp,update.jsp](webapp/WEB-INF/view/),

C:

BookinformationController.java(com.lms.controller.),

3.2. 功能设计:

3.2.1. 图书列表:

对应后端接口 @RequestMapping("bookList.do") 和 @RequestMapping("bookList2.do")，分别对应缩略图视图和列表视图，后端通过 bookinformationService.selectByMap 使用 SQL 查询符合条件（书名、类别、管理员所属单位）的图书，通过前后加通配符支持模糊查询；

3.2.2. 图书详情:

对应后端接口 @RequestMapping("toBookDetail.do")，后端通过 bookinformationService.selectByBookid 使用 SQL 查询对应 bookid 的用户；前端对应 detail.jsp；

3.2.3. 图书修改:

对应后端接口 @RequestMapping("toUpdateBook.do") 和 @RequestMapping("updateBook")，分别对应进入修改页面前的获取数据和进入页面后的修改数据，后端通过 bookinformationService.selectByBookid 使用 SQL 查询对应 bookid 的图书，通过 bookinformationService.updateSelective 使用 SQL 修改对应 bookid 的图书；前端对应 update.jsp；

3.2.4. 图书删除:

对应后端接口 @RequestMapping("deleteBook")，后端通过 bookinformationService.deleteByBookid 删除 bookinformation 表记录；

3.2.5. 图书添加:

前端对应 add.jsp，通过正则表达式判断输入值合法与否；

对应后端接口 @RequestMapping("addBook")，后端通过 bookinformationService.addSelective 添加图书；

批量导入：对应后端接口 @RequestMapping("imExcelForBook")，通过解析 Excel 表格的工具类 POI 实现

4. 单位管理模块：

4. 1. MVC 结构设计：

M:

Unit:

Unit.java(com.lms.entity.),

UnitService.java(com.lms.service,com.lms.service.impl.),

UnitMapper.java(com.lms.mapper.),

UnitMapperImpl.java(com.lms.mapper.impl.),

UnitMapper.xml(resources/mapper/),

unit(lms.) # 数据库表

V:

unit[add.jsp,detail.jsp,list.jsp,update.jsp](webapp/WEB-INF/view/),

C:

UnitController.java(com.lms.controller.),

4.2. 功能设计:

4.2.1. 单位列表:

对应后端接口 @RequestMapping("unitList.do"), 后端通过 unitService.selectByMap 使用 SQL 查询符合条件的单位, 通过前后加通配符支持模糊查询;

4.2.2. 单位详情:

对应后端接口 @RequestMapping("toBookDetail.do"), 后端通过 unitService.selectByUnitid 使用 SQL 查询对应 unitid 的单位; 前端对应 detail.jsp;

4.2.3. 单位修改:

对应后端接口 @RequestMapping("toUpdateUnit.do") 和 @RequestMapping("updateUnit"), 分别对应进入修改页面前的获取数据和进入页面后的修改数据, 后端通过 bookinformationService.selectByUnitid 使用 SQL 查询对应 Unitid 的单位, 通过 unitService.updateSelective 使用 SQL 修改对应 unitid 的单位; 前端对应 update.jsp;

4.2.4. 单位删除:

对应后端接口 @RequestMapping("deleteUnit"), 后端通过 unitService.deleteByUnitid 删除 unit 表记录;

4.2.5. 单位添加:

对应后端接口 @RequestMapping("addUnit"), 后端通过 unitService.addSelective 添加单位; 前端对应 add.jsp;

5. 借阅管理模块:

5.1. MVC 结构设计:

M:

BookBorrowing:

Bookborrowing.java(com.lms.entity.),

BookborrowingService.java(com.lms.service,com.lms.service.impl.),

BookborrowingMapper.java(com.lms.mapper.),

BookborrowingMapperImpl.java(com.lms.mapper.impl.),

BookborrowingMapper.xml(resources/mapper/),

bookborrowing(lms.) # 数据库表

V:

bookborrowing[borrow.jsp,list.jsp](webapp/WEB-INF/view/),

C:

BookborrowingController.java(com.lms.controller.),

5.2.1. 借阅图书列表:

对应后端接口 @RequestMapping("borrowList.do"), 后端通过 bookinformationService.selectByMap 使用 SQL 查询符合条件 (书名、类别、管理员所属单位) 的图书, 通过前后加通配符支持模糊查询; 前端对应 borrow.jsp;

5.2.2. 申请借阅:

对应后端接口 @RequestMapping("addBorrowing"), 后端通过 bookborrowingService.addSelective 使用 SQL 添加借阅记录;

5.2.3. 借阅记录列表:

对应后端接口 @RequestMapping("borrowingList.do"), 后端通过

bookborrowingService.selectByMap 使用 SQL 查询符合条件（读者 ID，管理员所属单位）的借阅记录；前端对应 list.jsp；

6. 流通管理模块：

6.1. MVC 结构设计：

M:

BookCirculation:

```
Bookcirculation.java(com.lms.entity.),
BookcirculationService.java(com.lms.service,com.lms.service.impl.),
BookcirculationMapper.java(com.lms.mapper.),
BookcirculationMapperImpl.java(com.lms.mapper.impl.),
BookcirculationMapper.xml(resources/mapper/),
bookcirculation(lms.) # 数据库表
```

V:

```
bookcirculation[circulate.jsp,list.jsp](webapp/WEB-INF/view/),
```

C:

```
BookcirculationController.java(com.lms.controller.),
```

6.2.1. 流通图书列表：

对应后端接口 @RequestMapping("circulateList.do")，后端通过

bookinformationService.selectByMap 使用 SQL 查询符合条件（书名、类别、非管理员所属单位）的图书，通过前后加通配符支持模糊查询；前端对应 circulate.jsp；

6.2.1. 申请流通：

对应后端接口 @RequestMapping("addCirculation")，后端通过

bookcirculationService.addSelective 使用 SQL 添加流通记录；

6.2.2. 流通记录列表：

对应后端接口 @RequestMapping("circulationList.do")，后端通过

bookcirculationService.selectByMap 使用 SQL 查询符合条件（管理员 ID，管理员所属单位）的流通记录；前端对应 list.jsp；

7. 统计分析模块：

7.1. MVC 结构设计：

M:

Statistics:

```
Statistics.java(com.lms.entity.),
StatisticsService.java(com.lms.service,com.lms.service.impl.),
StatisticsMapper.java(com.lms.mapper.),
StatisticsMapperImpl.java(com.lms.mapper.impl.),
StatisticsMapper.xml(resources/mapper/),
```

V:

```
statistics[statistics.jsp,list.jsp](webapp/WEB-INF/view/),
```

C:

```
StatisticsController.java(com.lms.controller.),
```

7.2.1. (借阅) 按占比统计：

对应后端接口 @RequestMapping("selectByUnitid")，后端通过

statisticsService.selectByUnitid 使用 SQL 查询借阅总数、在库总数、单位图书总数；前端对应 statistics.jsp，使用 echarts 呈现饼图；

7.2.1. (借阅) 按月统计：

对应后端接口 @RequestMapping("selectByUnitidMonthly")，后端通过 statisticsService.selectByUnitidMonthly 使用 SQL 查询月分记录；前端对应 statistics.jsp，使用 echarts 呈现折线图；

7.2.2. (借阅) 按年统计：

对应后端接口 @RequestMapping("selectByUnitidYearly")，后端通过 statisticsService.selectByUnitidYearly 使用 SQL 查询年分记录；前端对应 statistics.jsp，使用 echarts 呈现折线图；

(三) 系统部署与发布

1. 系统部署环境配置：

- 1.1. 数据表现层：Html+CSS+JavaScript+JQuery+JSP
- 1.2. 业务逻辑层：Java+Spring+SpringMVC
- 1.3. 数据持久层：MySQL+MyBatis
- 1.4. 开发工具：IntelliJ IDEA 2021.3.2 (Ultimate Edition)
- 1.5. 项目管理工具：Maven-3.6.3
- 1.6. Java Web 服务器：Tomcat-9.0.24

2. 系统部署说明：

2.1. 把项目中的 upload 目录导入 D:\ 盘根目录

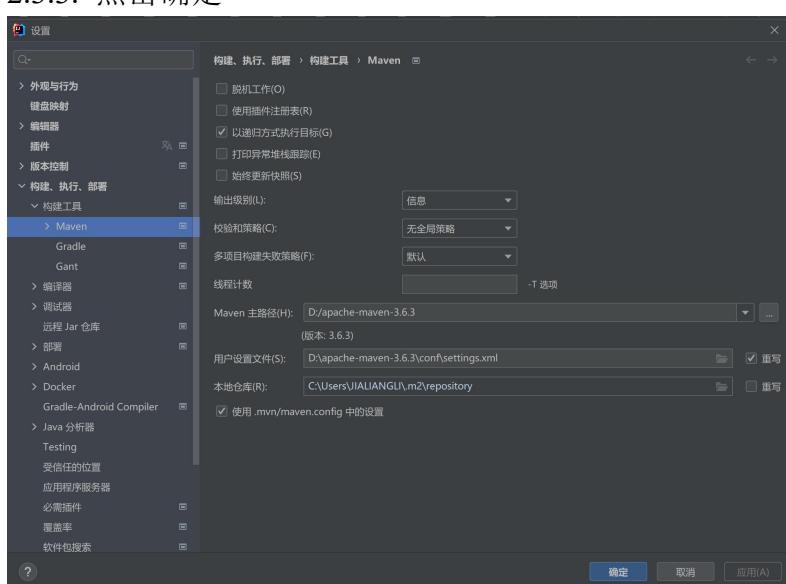
2.2. 将项目中的 LibraryManagementSystem 导入 IDEA

2.3. 在 IDEA 中，依次选择：文件/设置/项目、执行、部署/构建工具/Maven

2.3.1. 配置 Maven 主路径:D:/apache-maven-3.6.3

2.3.2. 配置用户设置文件:D:/apache-maven-3.6.3/conf/settings.xml

2.3.3. 点击确定

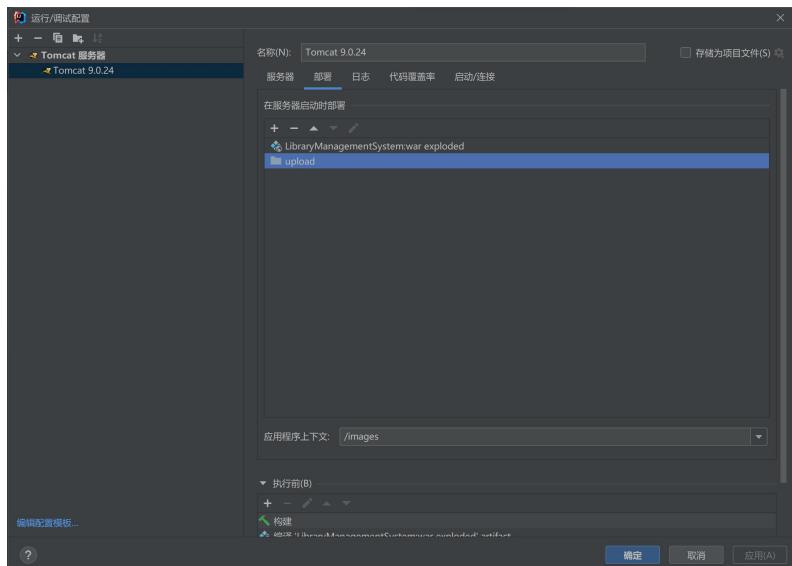


2.4. 在 IDEA 中，配置 Tomcat

2.4.1. 配置 Tomcat 服务器

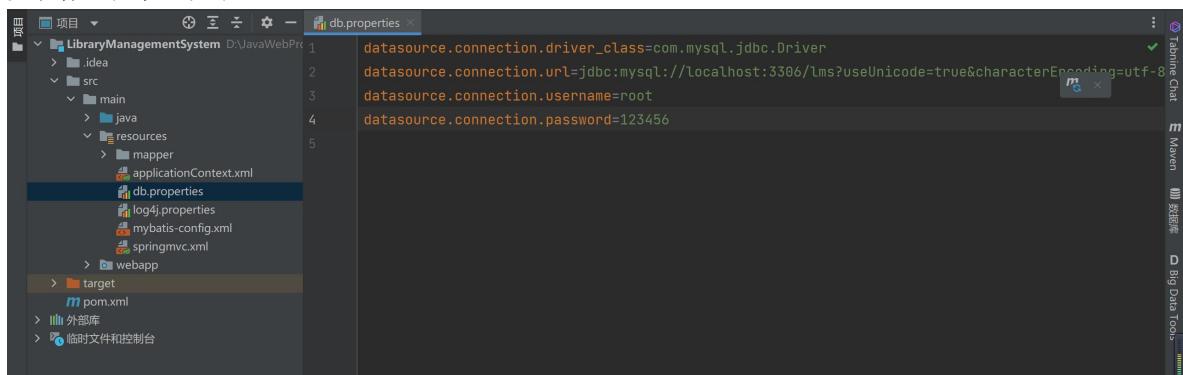
2.4.2. 在 Tomcat 的编辑配置下选择:部署, 点击“+”添加外部源, 选择 2.1 中导入的 D:\ 盘根目录下的 upload 目录, 配置应用程序上下文为 /images

2.4.3. 点击确定

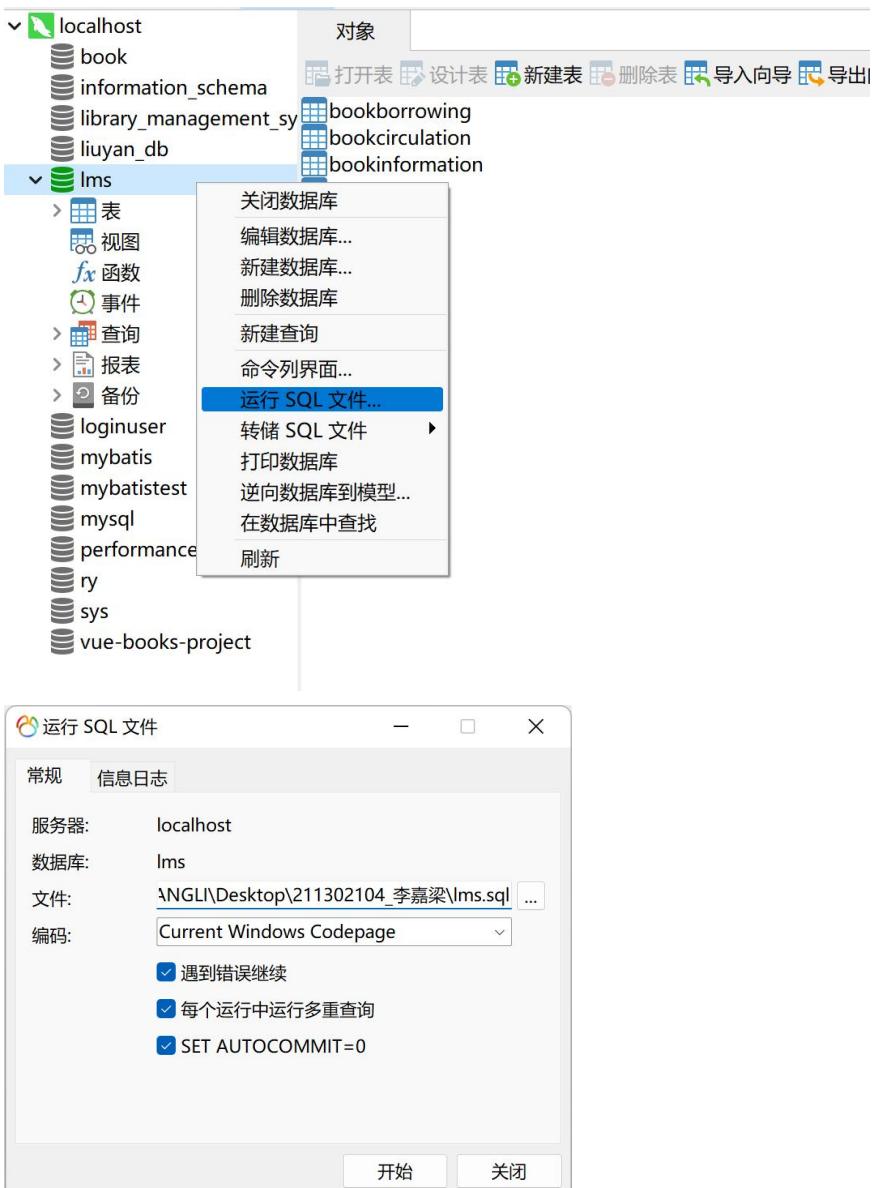


2.5. 配置数据库

2.5.1. 在 IDEA 中, LibraryManagementSystem/src/main/resources/db.properties 中配置用户名和密码



2.5.2. 在 Navicat 中新建 lms 数据库并右键运行 lms.sql 文件



3. 系统部署结果展示:

```
/*
```

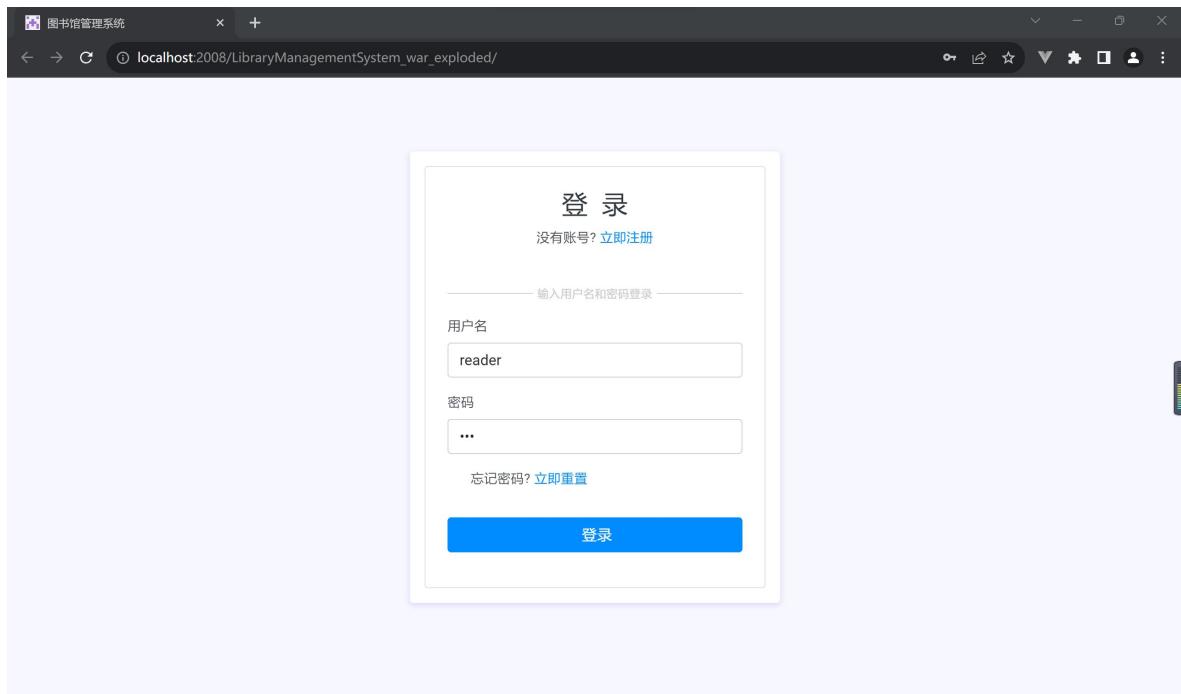
系统内置/测试用户如下:

	UserID	UserName	Password	RealName	ContactNumber
1	112594804	87654321	4321 清华工作人员	Li Jia Liang	18610822831
2	2023112317	12345678	5678 北林工作人员	Li Jia Liang	18610822831
3	2023112325	admin	123 管理员	Li Jia Liang	18610822831
4	2023112368	reader	123 读者	Li Jia Liang	18610822831
5	2023112393	reader2	123 读者2	Li Jia Liang	18610822831

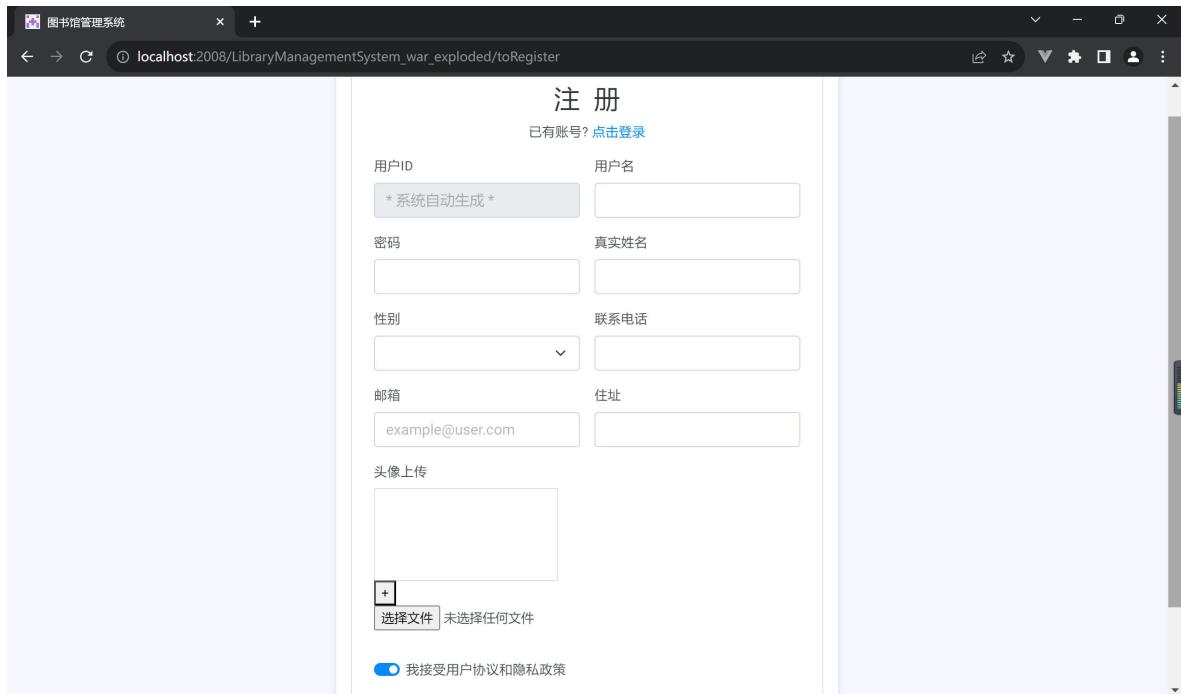
```
*/
```

3.1. 入口模块:

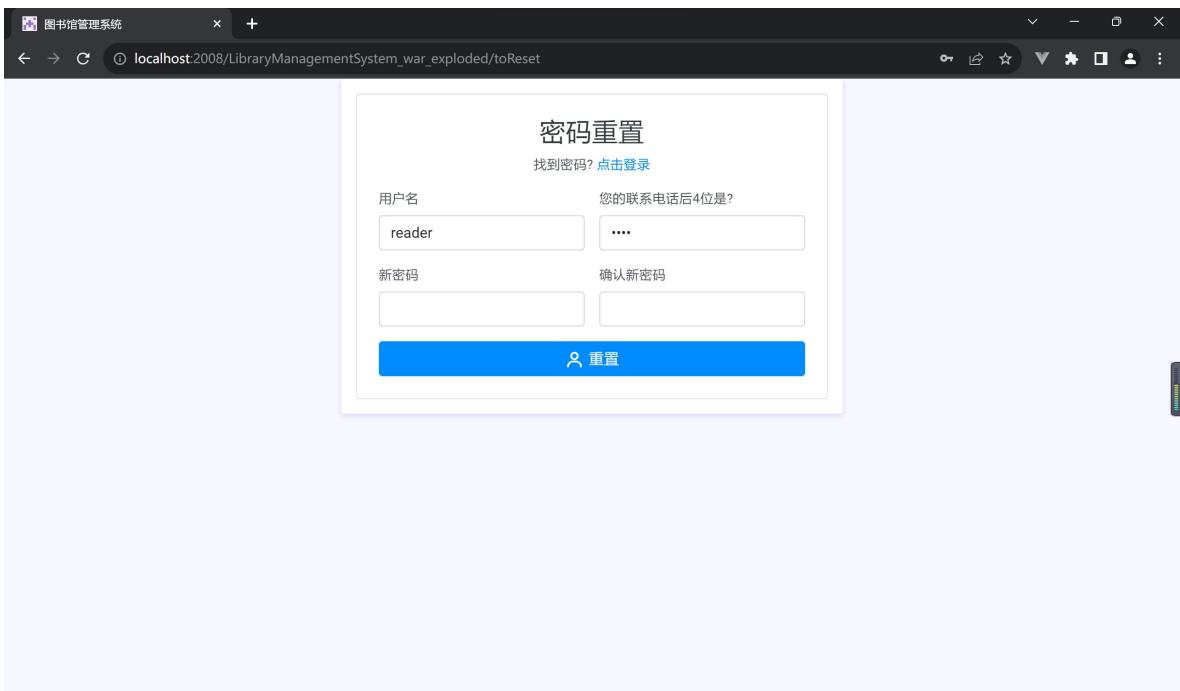
3.1.1. 登录



3.1.2. 注册



3.1.3. 密码重置



3.2. 用户管理模块

3.2.1. 用户列表（管理员）：

用户ID	用户名	头像	工作单位	角色	账号状态	操作
112594804	87654321		清华大学	工作人员	正常	<button>详情</button> <button>修改</button> <button>删除</button>
2023112317	12345678		北京林业大学	工作人员	正常	<button>详情</button> <button>修改</button> <button>删除</button>
2023112325	admin			管理员	正常	<button>详情</button> <button>修改</button> <button>删除</button>
2023112368	reader			读者	正常	<button>详情</button> <button>修改</button> <button>删除</button>
2023112393	reader2			读者	正常	<button>详情</button> <button>修改</button> <button>删除</button>

3.2.2. 添加用户（管理员）：

图书馆管理系统

用户管理 < +

localhost:2008/LibraryManagementSystem_war_exploded/toAddUser

管理员 admin

添加用户

用户名	工号
* 系统自动生成 *	* 4~10位工号(INT) *
用户名	密码
* 工号 *	* 工号后4位 *
角色	工作单位
工作人员	
真实姓名	性别
联系电话	邮箱
	example@example.com

3.2.3. 添加用户（批量导入文件格式，参照 user_sheet.xlsx）（管理员）：

	A	B	C
1	用户名	密码	
2	李嘉梁111	123	
3	李嘉梁222	123	
4			

3.2.4. 查看用户（管理员）：

图书馆管理系统

用户管理 <

图书管理 <

单位管理 <

图书详情

用户名	112594804
工作编号	87654321
工作单位	清华大学
角色	工作人员
用户名	87654321
密码	4321
真实姓名	Li Jia Liang
联系电话	18610822831
邮箱	lijiali030703@outlook.com
住址	No. 35 Qinghua East Road, Haidian District
注册时间	Sun Nov 26 23:13:57 CST 2023

Copyright © 2023. All right By bootstrapmb.

3.2.5. 修改用户（管理员）：

图书馆管理系统

修改用户

用户名	112594804	工号	87654321
用户名	87654321	密码	4321
真实姓名	Li Jia Liang	性别	(dropdown)
联系电话	18610822831	邮箱	lijialiang030703@outlook.com
住址	No. 35 Qinghua East Road, Haidian District	头像上传	(file input)

3.2.6. 个人信息（读者）：

图书馆管理系统

图书详情

用户名	2023112368
工作编号	
工作单位	
角色	读者
用户名	reader
密码	123
真实姓名	Li Jia Liang
联系电话	18610822831
邮箱	lijialiang030703@outlook.com
住址	No. 35 Qinghua East Road, Haidian District
注册时间	Sun Nov 26 23:11:16 CST 2023

Copyright © 2023. All right By bootstrapmb.

3.2.7. 编辑信息（读者）：

The screenshot shows the 'User Profile' editing page. On the left, there's a sidebar with '个人信息' (Personal Information) and '编辑信息' (Edit Information). The main area contains fields for '用户名' (Username), '真实姓名' (Real Name), '联系电话' (Phone Number), '住址' (Address), '工号' (Work Number), '密码' (Password), '性别' (Gender), and '邮箱' (Email). There's also a file upload field for '头像上传' (Avatar Upload) with a placeholder '选择文件' (Select File). A red '修改' (Update) button is at the bottom.

3.3. 图书管理模块

3.3.1. 图书列表缩略视图（管理员，工作人员）：

The screenshot shows the 'Book List' page. At the top, there's a search bar with '书名' (Book Name) and a dropdown for '类型' (Type), followed by a '查询' (Search) button. Below the search bar, there's a '切换视图' (Switch View) button. The main area displays four book entries in a grid:

图书编号: 123 书名: 计算机网络 <small>类型: 本 科数: 93 单位: 北京林业大学</small>	图书编号: 1234 书名: 数据库 <small>类型: 本数:单位:清 文学 97 华大学</small>	图书编号: 12345 书名: JavaWeb <small>类型: 本数: 单位: 北京技术 100 京大学</small>	图书编号: 1125105248 书名: dddd <small>类 本数: 单位: 北京型: 100 林业大学</small>
详情	详情	详情	详情
修改	修改	修改	修改

3.3.2. 图书列表列表视图（管理员，工作人员）：

The screenshot shows a web-based library management system interface. At the top, there's a header bar with a logo, the title '图书馆管理系统', and a user account for '管理员 admin'. Below the header is a sidebar with links for '用户管理', '图书管理', and '单位管理'. The main content area is titled 'Tables' and '图书列表'. It features a table with columns: 图书编号 (Book ID), 书名 (Book Name), 图片 (Image), 类型 (Type), 本数 (Quantity), 所属单位 (Unit), and 操作 (Operations). The table contains six rows of book data. Each row includes a '详情' (Details) button, a '修改' (Edit) button, and a '删除' (Delete) button.

图书编号	书名	图片	类型	本数	所属单位	操作
123	计算机网络		科学	93	北京林业大学	<button>详情</button> <button>修改</button> <button>删除</button>
1234	数据库		文学	97	清华大学	<button>详情</button> <button>修改</button> <button>删除</button>
12345	JavaWeb		技术	100	北京大学	<button>详情</button> <button>修改</button> <button>删除</button>
1125105248	ddddd			100	北京林业大学	<button>详情</button> <button>修改</button> <button>删除</button>
1125123055	bbbbbb			100	北京林业大学	<button>详情</button> <button>修改</button> <button>删除</button>
1125257495	cccccc			0	北京林业大学	<button>详情</button> <button>修改</button> <button>删除</button>

3.3.3. 图书列表缩略视图（读者）：

This screenshot shows the same library management system from a reader's perspective. The interface is similar but includes a search bar at the top with fields for '书名' (Book Name) and '类型' (Type), and a '查询' (Search) button. Below the search bar is a '切换视图' (Switch View) button. The main content area displays four large, colorful pixelated book icons. Each icon has a corresponding card below it providing detailed information:

- 图书编号: 123
书名: 计算机网络
类型: 科本数: 单位: 北京林业大学
学 93 大学 详情
- 图书编号: 1234
书名: 数据库
类型: 文 本数: 单位: 清华大学
学 97 大学 详情
- 图书编号: 12345
书名: JavaWeb
类型: 技 本数: 单位: 北京大学
术 100 大学 详情
- 图书编号: 1125105248
书名: dddd
类型: 本数: 单位: 北京林业大学
型: 100 大学 详情

3.3.4. 图书列表列表视图（读者）：

图书编号	书名	图片	类型	本数	所属单位	操作
123	计算机网络		科学	93	北京林业大学	<button>详情</button>
1234	数据库		文学	97	清华大学	<button>详情</button>
12345	JavaWeb		技术	100	北京大学	<button>详情</button>
1125105248	dddd			100	北京林业大学	<button>详情</button>
1125123055	bbbb			100	北京林业大学	<button>详情</button>
1125257495	cccc			0	北京林业大学	<button>详情</button>

3.3.5. 添加图书（工作人员）：

3.3.6. 添加图书（批量导入文件格式，参照 book_sheet.xlsx）（工作人员）：

	A	B	C	D
1	图书编号 (INT)	书名	本数 (INT)	
2	1111111111	软件工程	100	
3	2222222222	设计模式	100	
4				

3.4. 单位管理模块:

3.4.1. 单位列表（管理员）：

单位ID	单位名称	操作
0	北京林业大学	详情 修改 删除
1	清华大学	详情 修改 删除
2	北京大学	详情 修改 删除
2023112445	北京航空航天大学	详情 修改 删除

3.4.2. 添加单位（管理员）：

添加单位

单位ID <input type="text" value="* 系统自动生成 *"/>	单位名称 <input type="text"/>
联系人 <input type="text"/>	联系电话 <input type="text"/>
邮箱地址 <input type="text"/>	联系地址 <input type="text"/>
单位性质 <input type="text"/>	

[添加](#)

3.5. 借阅管理模块:

3.5.1. 借阅申请（读者）：

图书编号	书名	图片	类型	本数	所属单位	操作
123	计算机网络		科学	93	北京林业大学	<button>申请借阅</button> 详情
1234	数据库		文学	97	清华大学	<button>申请借阅</button> 详情
12345	JavaWeb		技术	100	北京大学	<button>申请借阅</button> 详情
1125105248	ddddd			100	北京林业大学	<button>申请借阅</button> 详情
1125123055	bbbbbb			100	北京林业大学	<button>申请借阅</button> 详情
1125257495	ccccc			0	北京林业大学	<button>申请借阅</button> 详情
1125363416	ddddd			100	北京林业大学	<button>申请借阅</button> 详情

3.5.2. 借阅记录（读者）：

借阅时间	归还时间	应还时间	状态	操作
Thu Nov 23 15:23:14 CST 2023	Thu Nov 23 16:06:35 CST 2023	Sat Dec 23 15:23:14 CST 2023	归还通过审核	<button>申请归还</button>
Wed Oct 18 12:34:57 CST 2023		Sun Dec 24 12:34:57 CST 2023	借阅未通过审核	<button>申请归还</button>
Wed Aug 16 12:35:31 CST 2023		Sun Dec 24 12:35:31 CST 2023	借阅通过审核	<button>申请归还</button>
Thu Jun 01 12:35:11 CST 2023		Sun Dec 24 12:35:11 CST 2023	借阅待审核	<button>申请归还</button>
Thu Feb 23 12:35:37 CST 2023		Sun Dec 24 12:35:37 CST 2023	借阅待审核	<button>申请归还</button>
Thu Nov 17 12:35:22 CST 2022		Sun Dec 24 12:35:22 CST 2023	借阅待审核	<button>申请归还</button>
Wed Dec 01 12:35:42 CST 2021		Sun Dec 24 12:35:42 CST 2023	借阅待审核	<button>申请归还</button>
Sat Aug 22 12:35:49 CST 2020		Sun Dec 24 12:35:49 CST 2023	借阅待审核	<button>申请归还</button>

3.5.3. 审核借阅（工作人员）：

时间	归还时间	应还时间	状态	操作
Nov 23 15:23:14 CST 2023	Thu Nov 23 16:06:35 CST 2023	Sat Dec 23 15:23:14 CST 2023	归还通过审核	<button>通过</button> <button>不通过</button>
Oct 18 12:34:57 CST 2023		Sun Dec 24 12:34:57 CST 2023	借阅未通过审核	<button>通过</button> <button>不通过</button>
Aug 16 12:35:31 CST 2023		Sun Dec 24 12:35:31 CST 2023	借阅通过审核	<button>通过</button> <button>不通过</button>
Jun 01 12:35:11 CST 2023		Sun Dec 24 12:35:11 CST 2023	借阅待审核	<button>通过</button> <button>不通过</button>
Feb 23 12:35:37 CST 2023		Sun Dec 24 12:35:37 CST 2023	借阅待审核	<button>通过</button> <button>不通过</button>
Nov 17 12:35:22 CST 2022		Sun Dec 24 12:35:22 CST 2023	借阅待审核	<button>通过</button> <button>不通过</button>
Dec 01 12:35:42 CST 2021		Sun Dec 24 12:35:42 CST 2023	借阅待审核	<button>通过</button> <button>不通过</button>
Aug 22 12:35:49 CST 2020		Sun Dec 24 12:35:49 CST 2023	借阅待审核	<button>通过</button> <button>不通过</button>

3.6. 流通管理模块:

3.6.1. 流通申请（北林工作人员）：

图书编号	书名	图片	类型	本数	所属单位	操作
1234	数据库		文学	97	清华大学	<button>申请流通</button>
12345	JavaWeb		技术	100	北京大学	<button>申请流通</button>

3.6.2. 流通记录（北林工作人员）：

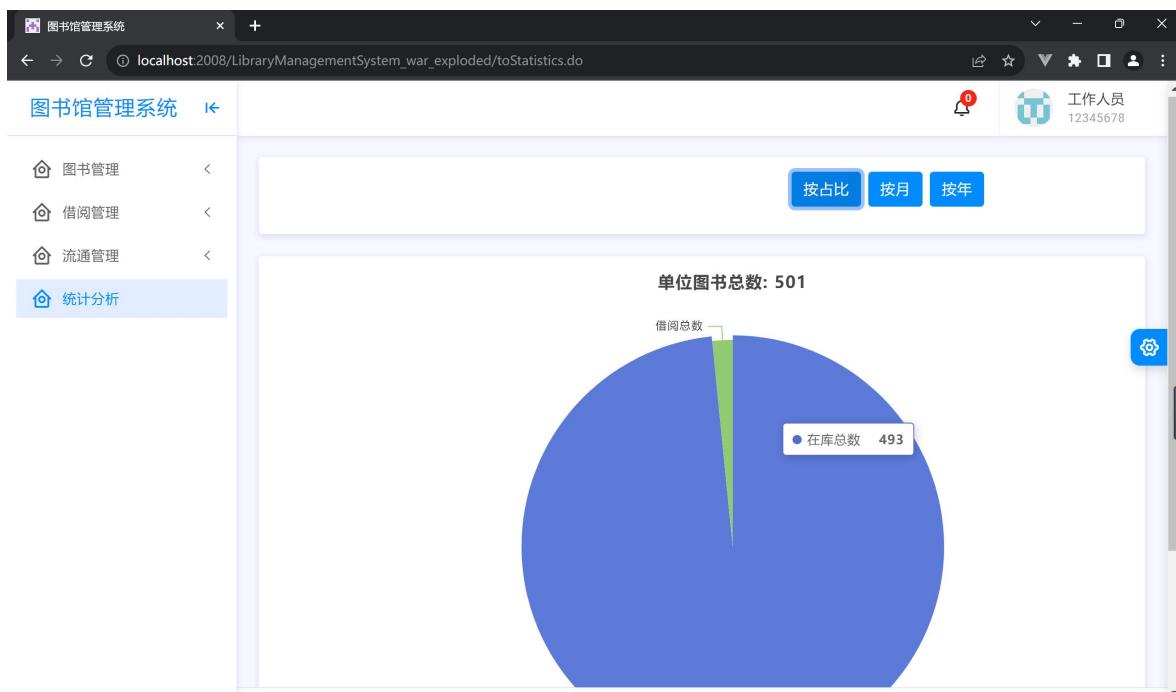
单位	申请人用户名	流通时间	归还时间	应还时间	状态	操作
大学	12345678	Sat Nov 25 11:24:44 CST 2023		Mon Dec 25 11:24:44 CST 2023	流通通过审核	<button>申请归还</button>
大学	12345678	Sat Nov 25 11:28:03 CST 2023		Mon Dec 25 11:28:03 CST 2023	流通待审核	<button>申请归还</button>
大学	12345678	Fri Nov 24 23:03:26 CST 2023		Sun Dec 24 23:03:26 CST 2023	流通待审核	<button>申请归还</button>

3.6.3. 审核流通（清华工作人员）：

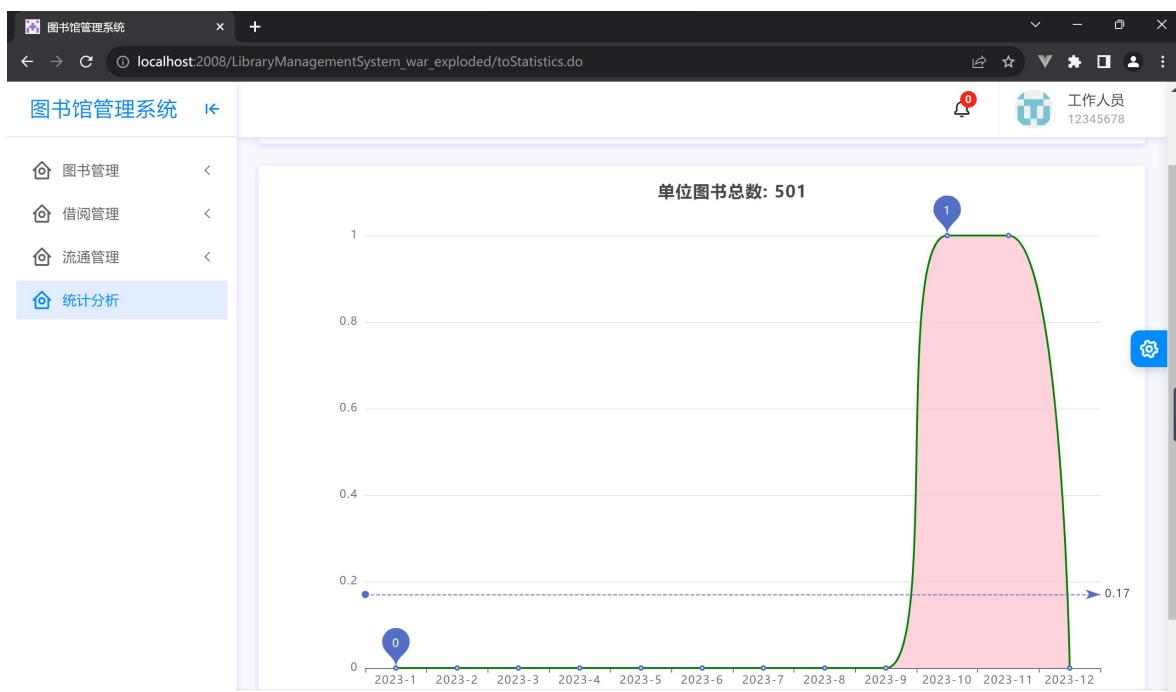
申请人用户名	流通时间	归还时间	应还时间	状态	操作
12345678	Sat Nov 25 11:24:44 CST 2023		Mon Dec 25 11:24:44 CST 2023	流通通过审核	<button>通过</button> <button>不通过</button>
12345678	Sat Nov 25 11:28:03 CST 2023		Mon Dec 25 11:28:03 CST 2023	流通待审核	<button>通过</button> <button>不通过</button>
12345678	Fri Nov 24 23:03:26 CST 2023		Sun Dec 24 23:03:26 CST 2023	流通待审核	<button>通过</button> <button>不通过</button>

3.7. 统计分析模块:

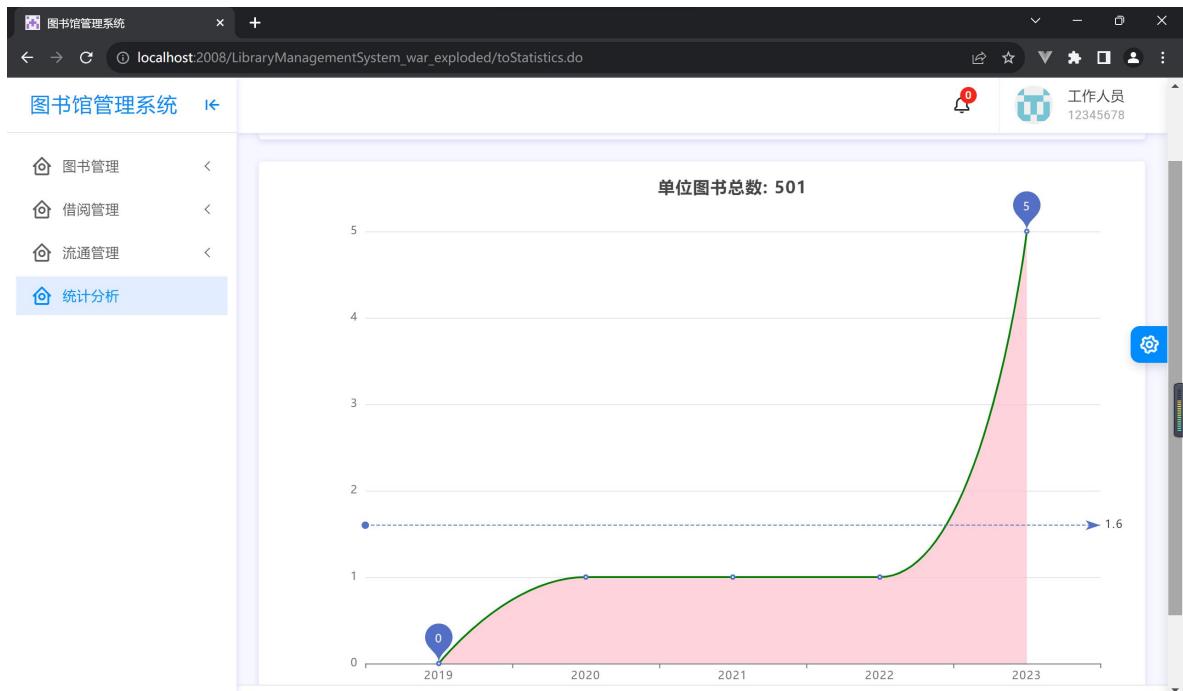
3.7.1. (借阅) 按占比统计 (工作人员) :



3.7.2. (借阅) 按月统计 (工作人员) :



3.7.3. (借阅) 按年统计 (工作人员) :



(四) 结论分析

1. 问题与解决方法:

1.1. 问题: 处理登录过期后页面显示? 怎么样返回登录界面?

解决:

- 1.1.1. 在每个 jsp 页面 include 的模板 jsp 中加入判断和跳转语句 (失败)
- 1.1.2. 使用 filter 技术——在进入登录后系统内每个网页 (url 都以.do 为后缀) 前都进行再次鉴权, 检查 session 中的账号是否失效, 如果失效则返回登录; 鉴权方式具体是: 通过异或运算符 (^) (只有一个条件为 true 时返回 true, 而其他情况都为 false) 确保用户 session 中此时有且仅有一个角色。代码如下:

```

24 ① @   public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws IOException {
25      {
26          RequestDispatcher dispatcher = request.getRequestDispatcher("index.jsp"); // 这里设置如果没有登录就转发到的页
27          HttpServletRequest req = (HttpServletRequest) request;
28          HttpServletResponse res = (HttpServletResponse) response;
29          HttpSession session = req.getSession(true);
30
31          // 从 session 里取的用户信息
32          // 这里获取 session, 为了检查 session 里有没有保存用户信息, 没有的话回转发到登陆页面
33          Object admin = session.getAttribute("admin");
34          Object worker = session.getAttribute("worker");
35          Object reader = session.getAttribute("reader");
36          if((admin != null) ^ (worker != null) ^ (reader != null)){ // 存在且仅存在一个角色则继续
37              // 已经登陆, 继续此次请求
38              System.out.println("用户已经登陆, 允许操作");
39
40              chain.doFilter(request, response);
41          }
42          else // 判断如果没有取到用户信息, 就跳转到登陆页面
43          {
44              // 跳转到登陆页面
45              System.out.println("用户没有登陆, 不允许操作");
46
47              dispatcher.forward(request, response);
        }
    }
}

```

1.2. 问题:即使在 selectByMap 中加了 unitid 筛选语句但并没有起到作用?

解决:

1.2.1. 经上网查询, 在 Mybatis 源码中发现问题:由于 `".length() = 0`, 传递的参数 `unitid` 值是 0 时 `viewId != "` 就变成 `0.0 != 0.0` 这个自然是 false。于是, 我将 SQL 中的判断条件`'and map.unitid!="'`去掉就成功解决了。Mybatis 源码和代码如下:

```

1 public static double doubleValue(Object value) throws NumberFormatException {
2     if (value == null) {
3         return 0.0D;
4     } else {
5         Class c = value.getClass();
6         if (c.getSuperclass() == Number.class) {
7             return ((Number)value).doubleValue();
8         } else if (c == Boolean.class) {
9             return (Boolean)value ? 1.0D : 0.0D;
10        } else if (c == Character.class) {
11            return (double)(Character)value;
12        } else {
13            String s = stringValue(value, true);
14            return s.length() == 0 ? 0.0D : Double.parseDouble(s);
15        }
16    }
17}

```

The screenshot shows a portion of a MyBatis XML configuration file. A specific line of code is highlighted with a red 'X' icon and a tooltip that reads "不加 and map.unitid!=". This indicates that the original code had a logical error where it was checking for a non-empty string length instead of a non-null value.

```

28 <select id="selectByMap" resultType="com.lms.entity.Bookinformation" parameterType="map">
29     select * from bookinformation where 1 = 1
30     <if test="map.bookid != null ">
31         and BookID = #{map.bookid}
32     </if>
33     <if test="map.unitid != null "> X 不加 and map.unitid!="
34         and UnitID = #{map.unitid}
35     </if>
36     <if test="map.keyword != null and map.keyword != '' ">
37         <bind name="keyword" value="%" + map.keyword + "%" />
38         and BookName like #{keyword}
39     </if>
40     <if test="map.bookcategory != null and map.bookcategory != '' ">
41         and BookCategory = #{map.bookcategory}
42     </if>
43     </select>
44     <delete id="deleteByBookid">

```

1.3. 问题:预先察觉 SQL 如果直接像 and 一样插入 or 可能会出现问题, 当前一个条件失效时, or 条件会直接与占位符并列组成 1=1 or ... , 条件永真。

解决:

1.3.1. 经深刻思考, 在 or 条件之上在加了一项对前一个条件失效的判断。代码如下:

The screenshot shows a more complex section of the MyBatis XML configuration. It includes multiple nested if statements and conditions involving circulation account IDs and units. The code is designed to handle various combinations of null and non-null values for these parameters.

```

95
96     <select id="selectByMap" resultType="com.lms.entity.Bookcirculation" parameterType="map" >
97         select * from bookcirculation where 1 = 1
98         <if test="map.circulationaccountid != null and map.circulationaccountid != '' ">
99             and CirculationAccountID = #{map.circulationaccountid}
100            </if>
101            <if test="map.unit != null and map.unit != '' ">
102                <if test="map.circulationaccountid != null and map.circulationaccountid != '' ">
103                    or Unit = #{map.unit}
104                </if>
105                <if test="map.circulationaccountid == null or map.circulationaccountid == '' ">
106                    and Unit = #{map.unit}
107                </if>
108            </if>
109            order by CirculationID desc;
110        </select>
111

```

1.4. 问题:无法修改对象。

解决:

1.4.1. 经深刻思考, 使用 updateSelective 方法时需要根据 ID 修改, 但如果表单中没有 ID 属性或 ID 属性设为 disabled="disabled", 则 spring 注入的对象获取不到 ID 值, 则无法修改对象。因此, 需要在表单中设置 ID 值并且设成 readonly="readonly"。代码如下:

```

30     :ID</label>
31     <ut-group-text bg-transparent"><i class="bx bxs-user"></i></span>
32     ="form-control border-start-0" id="unitid" value="${unit.unitid}" readonly="readonly">
33

```

1.5. 问题:mybatis mapper 方法返回值是 null 还是空对象, 或者空列表?

解决:

1.5.1. 经上网查询, 返回单个实体对象, 从数据库中没有查到数据时返回为 null; 返回 Map 类型数据, 从数据库中没有查到数据时返回为一个空 map (size==0), 内部用反射创建的 HashMap 对象, 有数据时放入这个 map, 没数据就直接返回这个没有任何元素的 map, 而不是 null;

返回 List 类型数据, 从数据库中没有查到数据时返回为一个空列表 (size==0), 内部返回的是自己 new 的 List, 有元素则添加进去, 没元素就返回这个 list, 不会为 null。

Mybatis 源码如下:

```

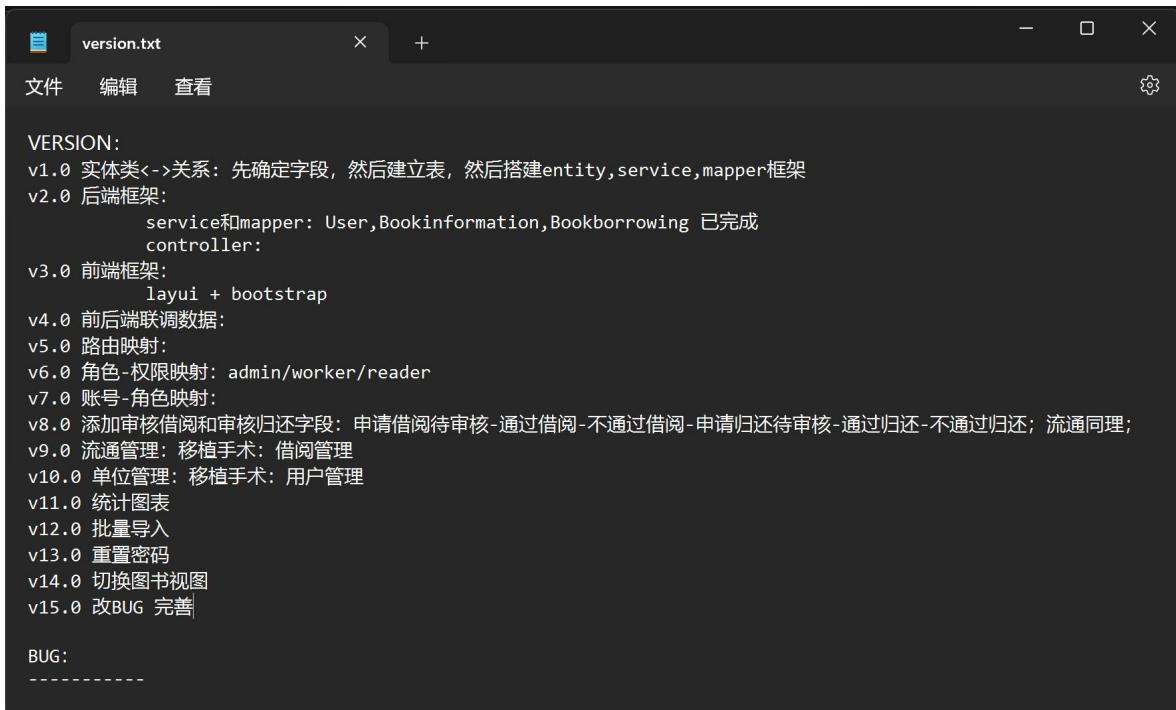
if (method>ReturnsVoid() && method.hasResultHandler()) {
    //如果有结果处理器
    executeWithResultHandler(sqlSession, args);
    result = null;
} else if (method>ReturnsMany()) {
    //如果结果有多条记录
    result = executeForMany(sqlSession, args);
} else if (method>ReturnsMap()) {
    //如果结果是map
    result = executeForMap(sqlSession, args);
} else {
    //否则就是一条记录
    Object param = method.convertArgsToSqlCommandParam(args);
    result = sqlSession.selectOne(command.getName(), param);
}

```

https://blog.csdn.net/qq_36951116

2. 收获和体会:

2.1. 体验到了一次完整的项目开发经历, 版本演变至今如下图:



```

version.txt

文件 编辑 查看 ⚙

VERSION:
v1.0 实体类<->关系: 先确定字段, 然后建立表, 然后搭建entity, service, mapper框架
v2.0 后端框架:
    service和mapper: User, Bookinformation, Bookborrowing 已完成
    controller:
v3.0 前端框架:
    layui + bootstrap
v4.0 前后端联调数据:
v5.0 路由映射:
v6.0 角色-权限映射: admin/worker/reader
v7.0 账号-角色映射:
v8.0 添加审核借阅和审核归还字段: 申请借阅待审核-通过借阅-不通过借阅-申请归还待审核-通过归还-不通过归还; 流通同理;
v9.0 流通管理: 移植手术: 借阅管理
v10.0 单位管理: 移植手术: 用户管理
v11.0 统计图表
v12.0 批量导入
v13.0 重置密码
v14.0 切换图书视图
v15.0 改BUG 完善

BUG:
-----

```

2.2. 通过自己搭建 SSM 框架并实现，相当于进行了一次代码上的课程总结，使我的知识框架更为系统、视角更为全面、理解也更为完整。经过“初步编写-出错调试-再次重构”的整个过程，我对基于 SSM 框架的项目开发有了系统、全面、完整的理解和掌握，完成了从理论到实践的知识迁移，正向使得原理与实现联系更加紧密，反向修补了理论学习中的知识漏洞。同时，我在本次实践中对断点调试追踪和软件设计模式也有了更深的理解和印证。非常感激王老师的生动引领教导！

2.3. 这个过程，我不仅复习了旧的知识，我也通过发现问题和技术学习到了很多新技能，每个创造、推导和重构的过程都锻造了我的开发思维和能力。

2.4. 学会了从不同角度切入问题，在作为开发者时的自底向上方法和作为使用者的自顶向下方法中来回切换时我可以把开发的问题看得更清，抓得更准。

3. 尚存在的问题:

3.1. 图书编号字段在初始确定数据类型时判断有误，设置成了 INT，导致后面发现应该是 ISBN 这样的 VARCHAR 时，修改比较麻烦，所以在初始确定字段及其数据类型时一定要谨慎周全。