数据库课程设计报告

学号：161820318

姓名：李昌建

指导教师：周良

2021/7/1

目录

一、 需求分析

1. 背景：随着信息技术的不断发展，越来越多的高校开始进行信息化建设。其中对学生信息的管理，如：学生的选课信息、成绩信息、教材订购信息等都是这方面的重要内容。学生信息规模大、项目条数多、信息量庞大，传统的人工管理方式显然已经无法满足要求，需要借助计算机来进行现代化信息管理，从而提高管理的准确性与高效性。

2. 可行性： 传统人工管理学生订购教材信息存在诸多弊病，比如容易出错、效率低等。计算机具有存储快、查找便利、准确性高的特点，能非常好的解决人工管理的弊病。

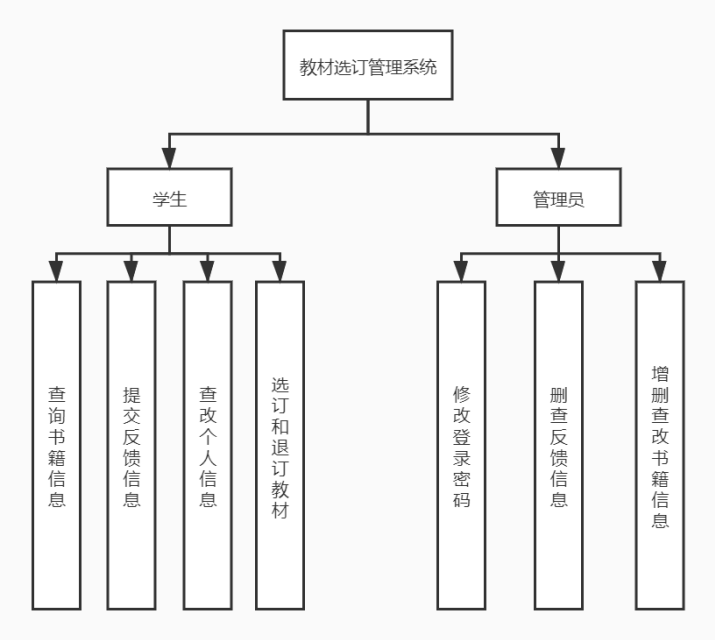
3. 开发环境

   编程工具：vscode、intellij IDEA；
   数据库：MySql8.0；
   技术路线：vue.js+java springboot+jdbc

1.1 系统功能需求

系统功能框架介绍，处理模块描述

4. 管理员：
   1. 查询教材信息
   2. 添加教材信息
   3. 修改教材信息
   4. 查看反馈信息
   5. 删除反馈信息
   6. 修改登录密码
5. 学生：
   1. 查看个人信息
   2. 修改个人信息
   3. 修改登录密码
   4. 提交反馈信息
   5. 查询教材信息
   6. 选订教材
   7. 退订教材



1.2 其他性能需求

1. 分管理员、学生不同身份登录，并给予不同的权限，提高系统的安全性；
2. 使用前端代码验证用户输入的正确性；
3. 充分考虑系统报错情况，尽量防止系统崩溃。

二、 概念结构设计

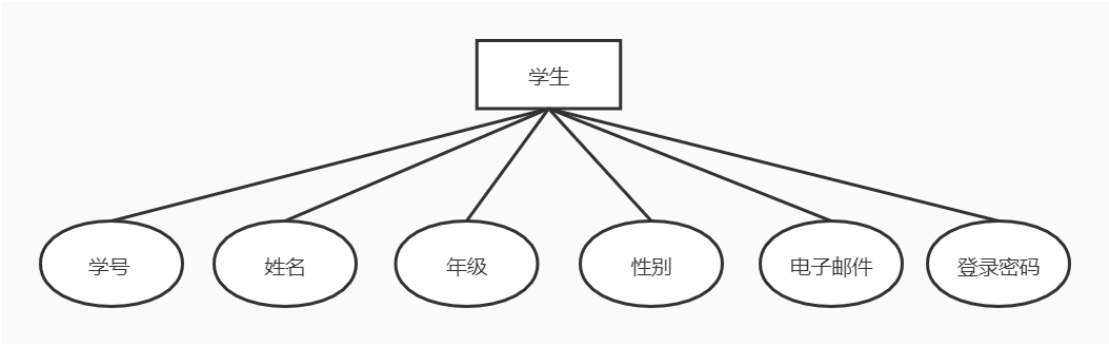
2.1 实体关系分析

1. 设置年级，每个年级包含若干学生；
2. 每本书可以被多个学生选订，但是每个学生只能选订同种的一本书；
3. 管理员可以对书籍信息进行管理；
4. 需要考虑年级，对应年级的学生只能选订对应年级的教材；
5. 学生可以提交反馈信息供管理员查看和处理；

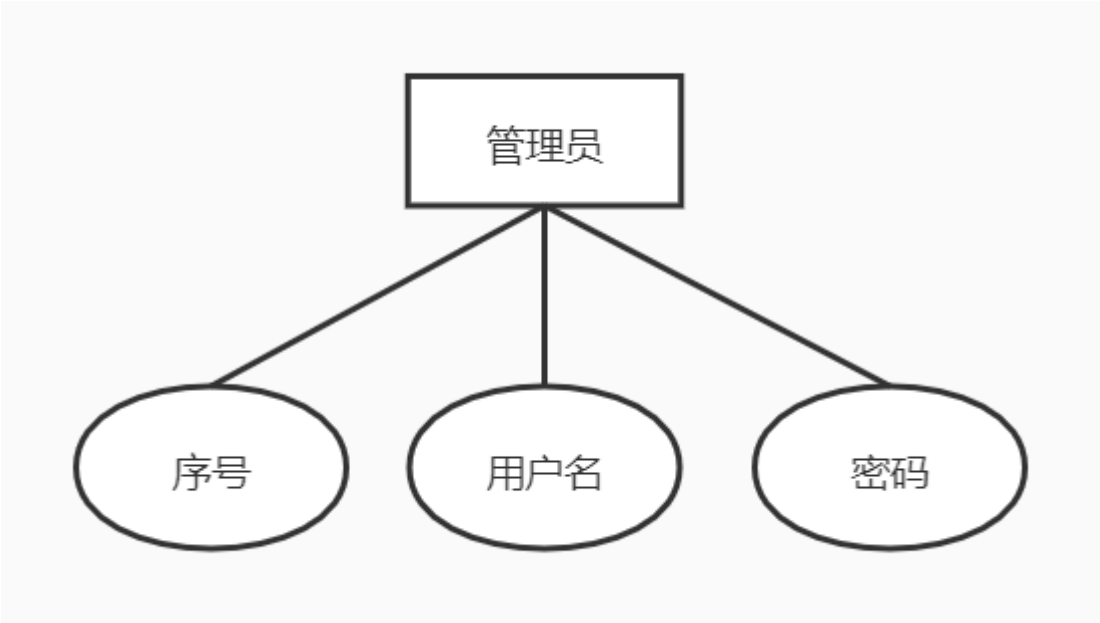
2.2 E-R 图

各实体的属性图：

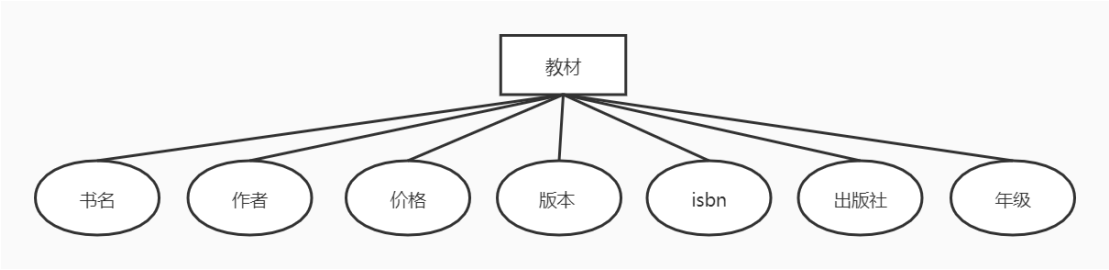
- 学生：



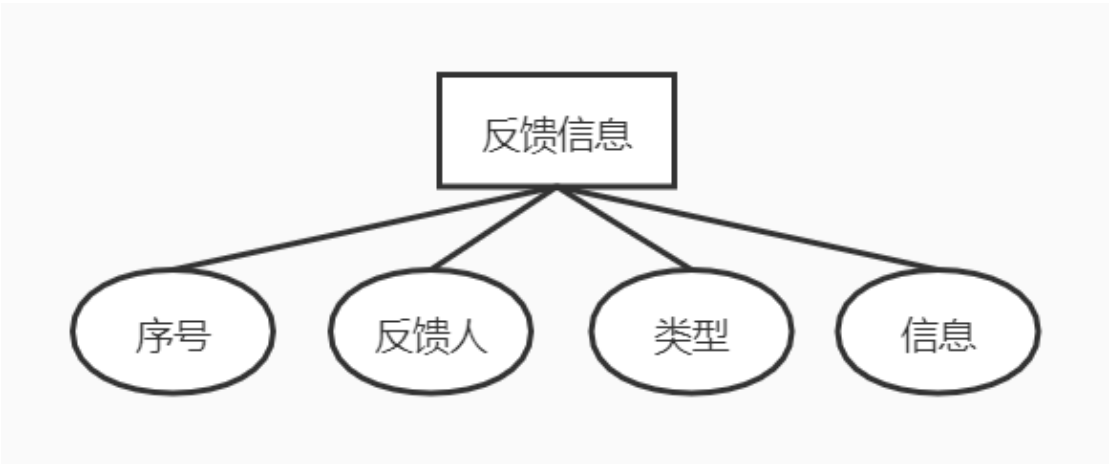
- 管理员：



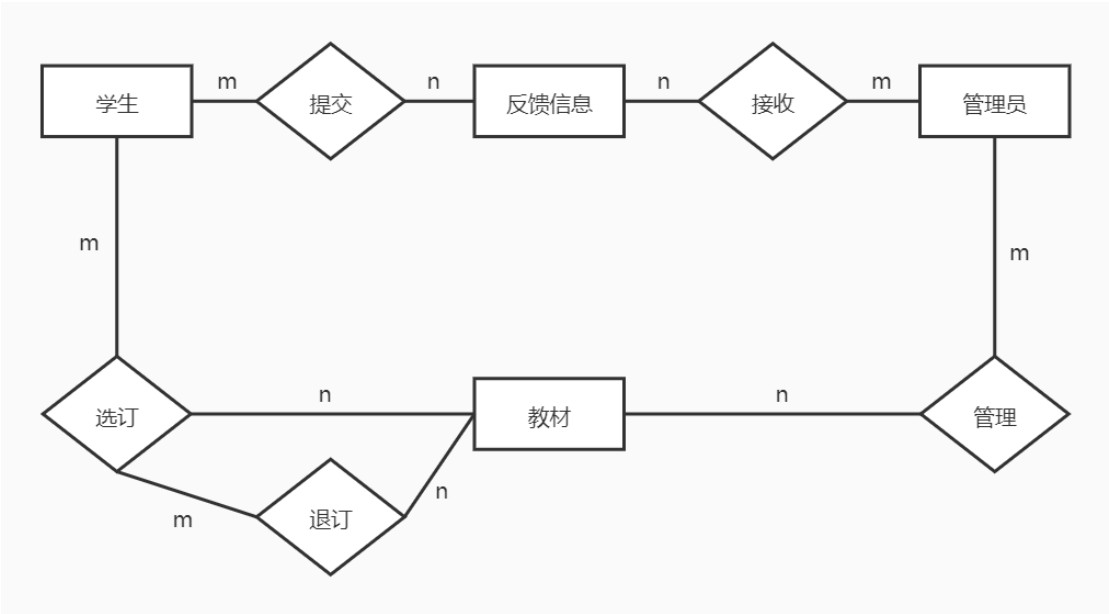
- 教材：



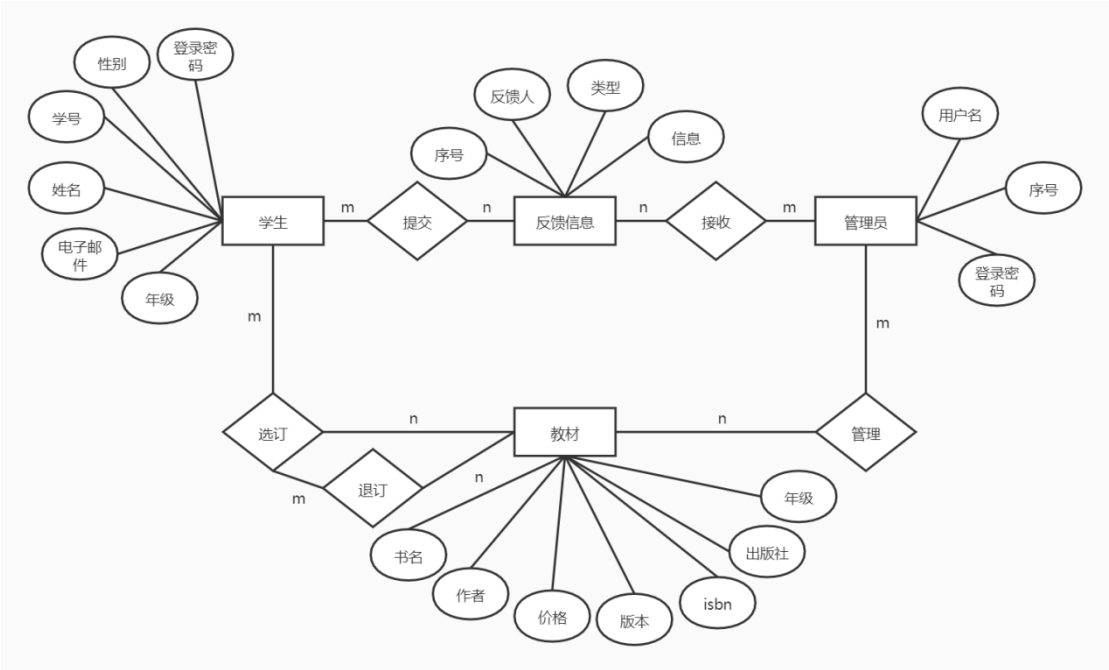
- 反馈信息



实体联系图：



完整的 E-R 图：



三、 逻辑结构设计

3.1 关系模式设计

将 E-R 图转换为以下关系模式：（其中加粗的为主键，斜体为外键）

- 学生（**学号**，姓名，性别，电子邮件，登录密码，年级）
- 管理员（**序号**，用户名，登录密码）
- 书籍（书名，作者，价格，版本，**isbn**，出版社，年级）
- 反馈信息（**序号**，反馈人，类型，信息）
- 订书信息（***学号***，***isbn***）

3.2 数据类型定义

对关系模式中的属性定义类型、长度和约束：

1. 学生：

数据项名	数据类型	长度	完整性约束	备注
id	varchar	20	主键，唯一，非空	学号
name	varchar	50		姓名
sex	varchar	10		性别
email	varchar	50		电子邮件
password	varchar	50	非空	登陆密码
grade	varchar	10		年级

2. 教材：

数据项名	数据类型	长度	完整性约束	备注
name	varchar	50	非空	书名
author	varchar	30	非空	作者
value	int	4	非空	价格
edition	int	4	非空	版本
isbn	varchar	50	主键，唯一，非空	isbn
publisher	varchar	50	非空	出版社
grade	varchar	10	非空	年级

3. 管理员

数据项名	数据类型	长度	完整性约束	备注
id	int	4	主键,非空，自动递增	序号
name	varchar	255	非空	用户名
password	varchar	255	非空	登陆密码

4. 反馈信息

数据项名	数据类型	长度	完整性约束	备注
id	int	4	主键,非空，自动递增	序号
user	varchar	20	非空	反馈人
type	varchar	20	非空	类型
text	varchar	512	非空	信息

5. 订书信息

数据项名	数据类型	长度	完整性约束	备注
id	varchar	20	联合主键，外键	学生学号
isbn	varchar	50	联合主键，外键	isbn

四、 创建数据库

4.1 创建学生表（student）

```
DROP TABLE IF EXISTS `student`;
CREATE TABLE `student`  (
  `id` varchar(20) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `name` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NULL DEFAULT NULL,
  `sex` varchar(10) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NULL DEFAULT NULL,
  `email` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NULL DEFAULT NULL,
  `passward` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NULL DEFAULT NULL,
  `grade` varchar(10) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NULL DEFAULT NULL,
```

```
  PRIMARY KEY (`id`) USING BTREE
) ENGINE = InnoDB CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci ROW_FORMAT = DYNAMIC;
```

4.2 创建教材表（book）

```
DROP TABLE IF EXISTS `book`;
CREATE TABLE `book`  (
  `name` varchar(50) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL,
  `author` varchar(30) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL,
  `value` int NULL DEFAULT NULL,
  `edition` int NULL DEFAULT NULL,
  `isbn` varchar(50) CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL,
  `publisher` varchar(50) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL,
  `grade` varchar(10) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL,
  PRIMARY KEY (`isbn`) USING BTREE
) ENGINE = InnoDB CHARACTER SET = utf8 COLLATE = utf8_general_ci ROW_FORMAT = DYNAMIC;
```

4.3 创建管理员表（admin）

```
DROP TABLE IF EXISTS `admin`;
CREATE TABLE `admin`  (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(255) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL,
  `password` varchar(255) CHARACTER SET utf8 COLLATE utf8_general_ci NULL DEFAULT NULL,
  PRIMARY KEY (`id`) USING BTREE
) ENGINE = InnoDB AUTO_INCREMENT = 1 CHARACTER SET = utf8 COLLATE = utf8_general_ci ROW_FORMAT = DYNAMIC;
```

4.4 创建反馈信息表（inform）

```
DROP TABLE IF EXISTS `inform`;
CREATE TABLE `inform`  (
  `id` int NOT NULL AUTO_INCREMENT,
  `user` varchar(20) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NULL DEFAULT NULL,
  `type` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NULL DEFAULT NULL,
  `text` varchar(512) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NULL DEFAULT NULL,
  PRIMARY KEY (`id`) USING BTREE
) ENGINE = InnoDB AUTO_INCREMENT = 7 CHARACTER SET = utf8mb4 COLLATE = utf8mb4_0900_ai_ci ROW_FORMAT = Dynamic;
```

4.5 创建订书信息表（subscribe）

```
CREATE TABLE `subscribe`  (
  `id` varchar(20) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
```

```
  `isbn` varchar(50) CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL,
  PRIMARY KEY (`id`, `isbn`) USING BTREE,
  INDEX `isbn`(`isbn`) USING BTREE,
  CONSTRAINT `subscribe_ibfk_2` FOREIGN KEY (`isbn`) REFERENCES `book` (`isbn`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `subscribe_ibfk_3` FOREIGN KEY (`id`) REFERENCES `student` (`id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE = InnoDB CHARACTER SET = utf8 COLLATE = utf8_general_ci ROW_FORMAT = DYNAMIC;
```

五、 服务端连接数据库

服务端使用了 java springboot 框架通过 jdbc 连接数据库

1、 在 maven 项目的配置文件 pom.xml 添加依赖：

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-jdbc</artifactId>
</dependency>

<dependency>
    <groupId>com.microsoft.sqlserver</groupId>
    <artifactId>mssql-jdbc</artifactId>
    <scope>runtime</scope>
</dependency>
```

2、 在项目中的 application.properties 文件中配置数据库的连接信息和设置：

```
spring.datasource.url=jdbc:mysql://127.0.0.1/bookos
spring.datasource.username=root
spring.datasource.password=root
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.max-idle=10
spring.datasource.max-wait=10000
spring.datasource.min-idle=5
spring.datasource.initial-size=5
```

3、 使用时通过 JdbcTemplate 类创建一个对象，构造 sql 语句之后通过类方法：update、query 等对数据库进行操作，例如：

```
JdbcTemplate jdbcTemplate;

public List<Map<String,Object>> allBook(){
    try {
        String sql="SELECT * FROM book";
```

```
        return jdbcTemplate.queryForList(sql);
    }catch (Exception e){
        System.out.println(e.toString());
    }
    return null;
}

public Boolean deleteBook(List<Book> bookList){
    try {
        String sql="DELETE FROM book WHERE isbn=?";
        for(Book book:bookList){
            jdbcTemplate.update(sql,book.getIsbn());
        }
        return true;
    }
    catch (Exception e){
        System.out.println(e.toString());
    }
    return false;
}
```

六、 部分功能实现介绍

6.1 系统登录功能

前端实现：
登录功能有两种方式管理员登录和用户登录，通过前端的 js 代码来控制两种不同的登录方式。用户通过前端的表单控件进行登录信息输入，输入的信息存储在 form 中，js 代码先对输入的合法性进行检验，然后通过 form.identity 判断登录方式，接着通过 axios 向服务端 url 发送 get 请求，并提交参数。然后通过后端的返回判断登录是否成功并给出相应的信息，或直接进行路由跳转。

```
form: {
    username: '',
    passward: '',
    identity: '',
}

Signin: function(){
    //console.log(this.form);
    this.$refs['form'].validate((valid) => {
        if (valid) {
            if(this.form.identity=='student'){
                axios.get('http://localhost:8081/login',{
                    params:{
                        id: this.form.username,
```

```
                        passward: this.form.passward
                    }
                }).then(resp=>{
                    if(resp.data=="success")
                        this.$router.push('/home');
                    else
                         失败处理…
                });
            }
            else if(this.form.identity=='root'){
                axios.get('http://localhost:8081/admin/login',{
                    params:{
                        name: this.form.username,
                        passward: this.form.passward
                    }
                }).then(resp=>{
                    if(resp.data=="success")
                    this.$router.push('/admin');
                    else
                      失败处理…
                });
            }
        }
        else{
            this.$message({message: "请输入登录信息", type:"error"});
        }
    })
}
```

后端实现：
前端向后端 url 提交登录信息后，后端使用 GetMapping 映射接收请求，然后将参数存储在 student 对象中，并通过参数中的学号使用 sql 语句去数据库中查找对应的密码，并比较是否相同，返回 success 表示成功，passworderr 表示密码错误，noninform 表示没有用户信息。

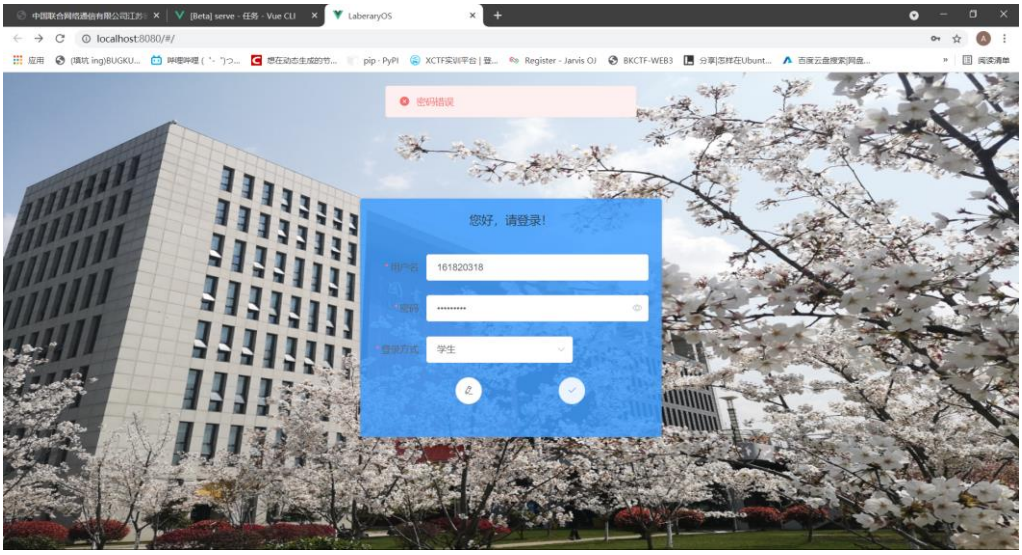
学生登录：

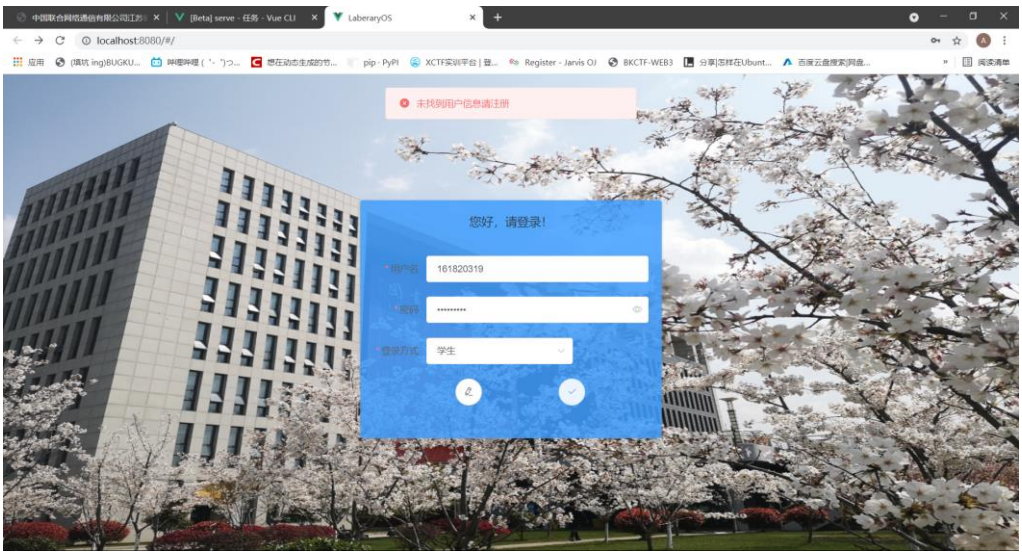
```
@GetMapping("/login")
public String userLogin(Student student){
    String status = user.login(student);
    if(status.equals("success"))
        stu=student;
    return status;
}
```

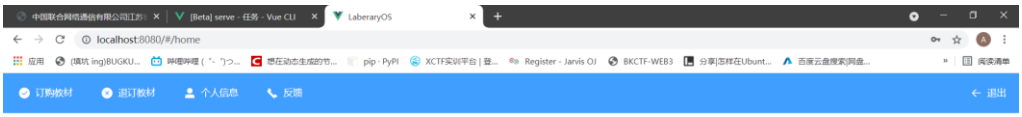
```
public String login(Student student){
    try{
        String sql="SELECT * FROM student WHERE id=?";
        Map<String,Object>
stu=jdbcTemplate.queryForMap(sql,student.getId());
        if(!stu.get("passward").equals(student.getPassward()))
            return "passworderr";
    }catch (Exception e){
        System.out.println(e.toString());
        return "noninform";
    }
    return "success";
}
```

管理员登录与学生登录实现方式相似

演示：







6.2 学生订购、退订教材功能

订购

前端实现：前端中用户通过复选框控件选择想要订购的教材，选中的教材信息会加入到列表中，然后通过 axios 向后端 url 发送 post 请求，并提交参数，接收后端返回的还未选订的教材的列表，显示在前端的表格中。

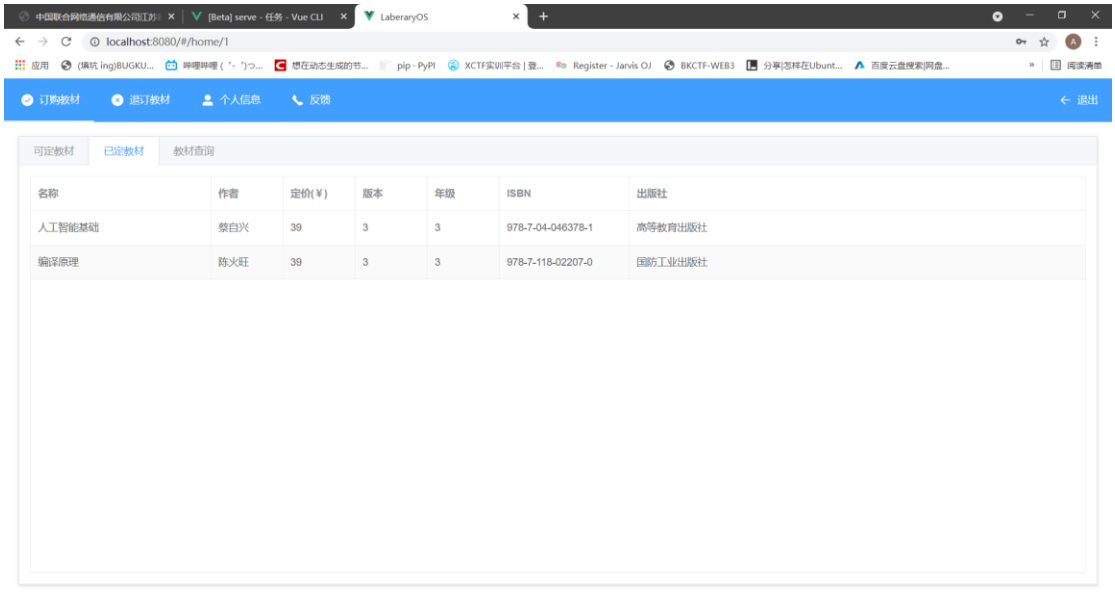
```
multipleSelection: []


confirm(){
    axios.post("http://localhost:8081/subscribe",this.multipleSelection)
.then(resp=>{
        this.unsubscribeTable=resp.data;
    });
    this.dialogTableVisible=false;
    this.$message({message:"选订成功",type:"success"});
}
```

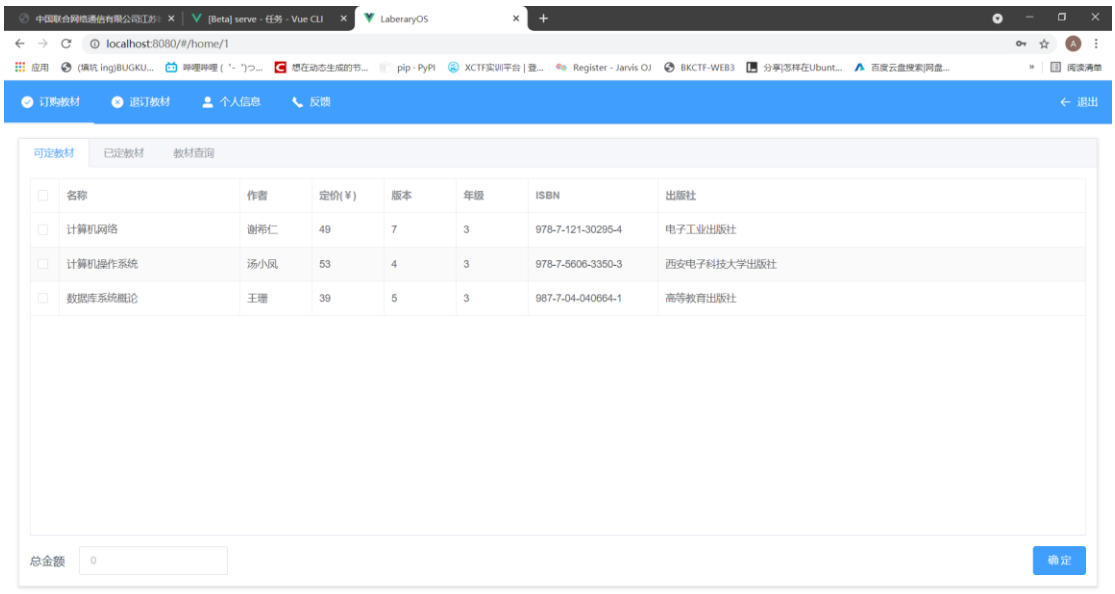
后端实现：接收前端传来的订购的教材列表，使用循环处理，构造 sql 语句，将学生的学号和教材的 isbn 信息插入订购信息表表示学生已经选订了这些教材。然后调用查询方法得到该学生还未订购的教材列表并返回。

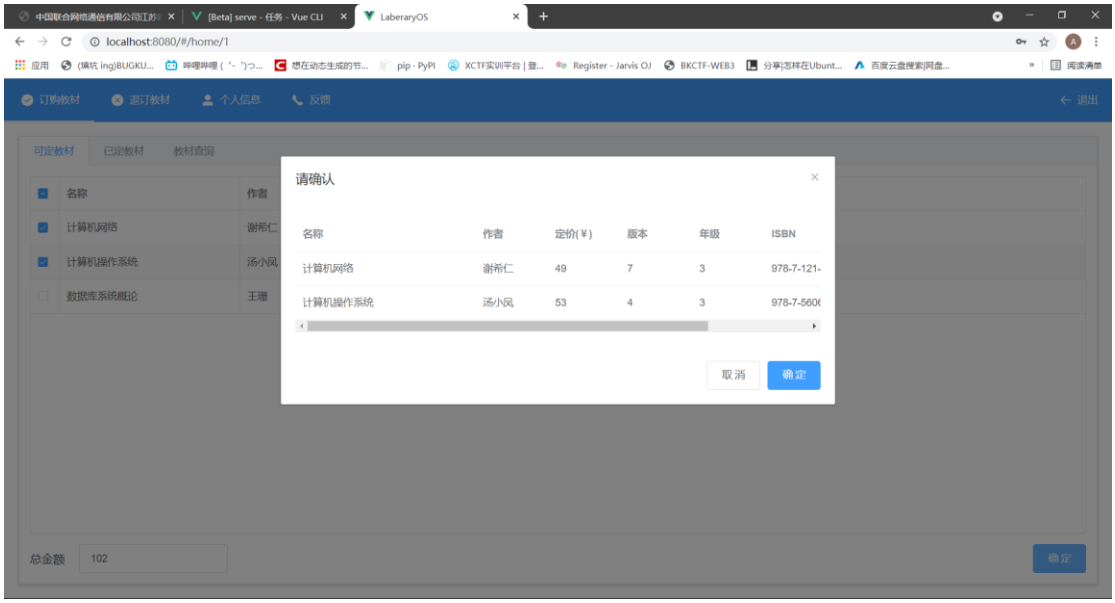
```
@PostMapping("/subscribe")
public List<Map<String,Object>> userSubscribe(@RequestBody List<Book>
bookList){
    if(user.subscribe(bookList,stu))
        return user.searchUnsub(stu);
    else
        return null;
}
```

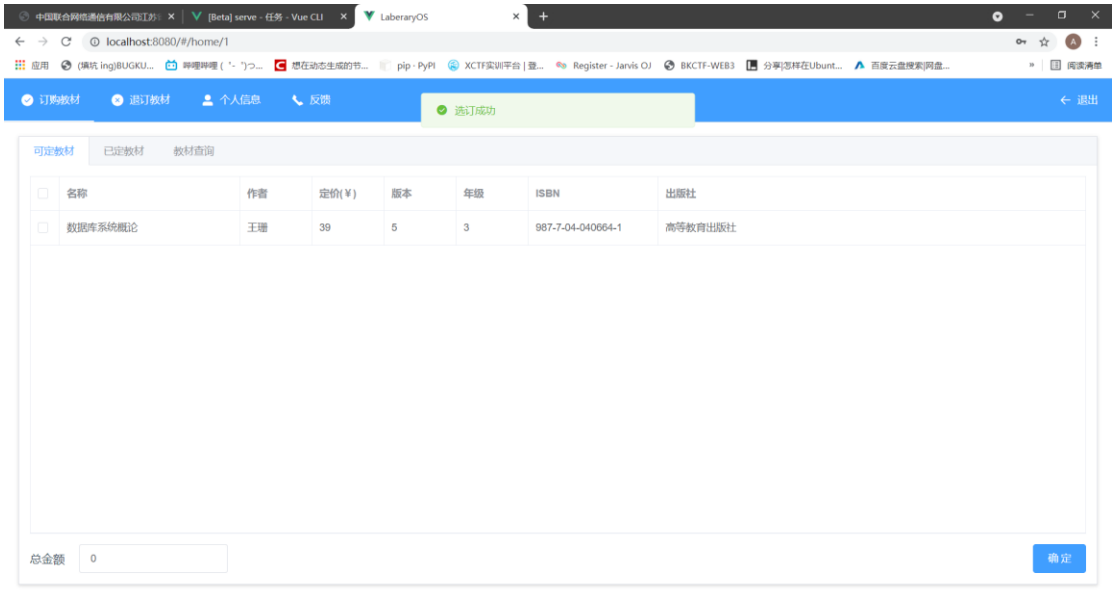
```java
public Boolean subscribe(List<Book> bookList,Student student){
    try{
        String sql="INSERT INTO subscribe VALUES(?,?)";
        for(Book book:bookList){
            jdbcTemplate.update(sql,student.getId(),book.getIsbn());
        }
        return true;
    }catch (Exception e){
        System.out.println(e.toString());
    }
    return false;
}
```

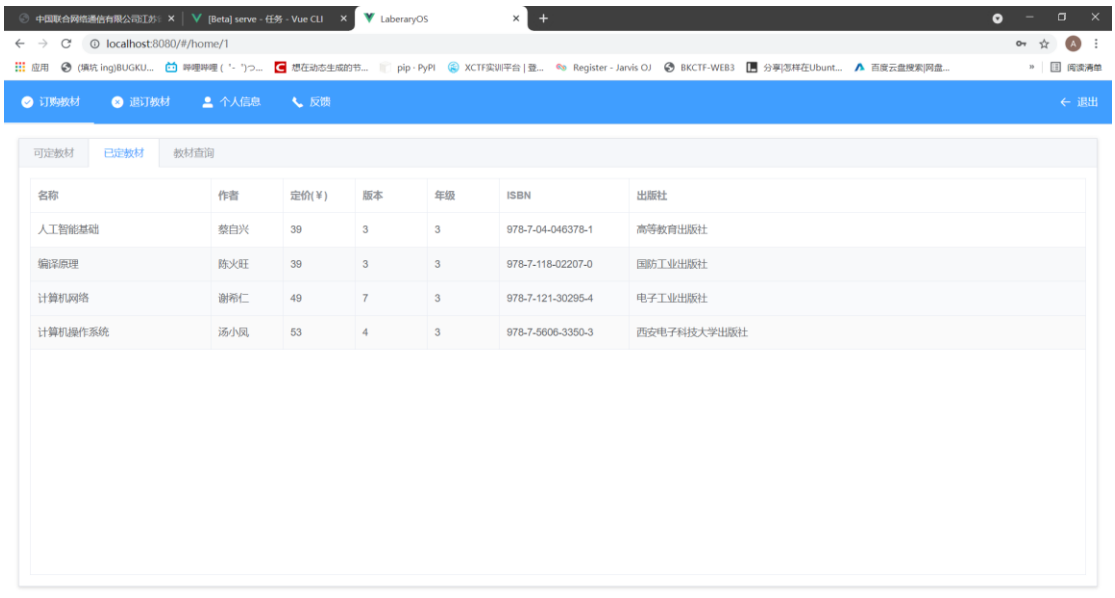
演示：






LaberaryOS
localhost:8080/#/home/1
订购教材
退订教材
个人信息
反馈
退出
可定教材
已定教材
教材查询
名称
作者
计算机网络
谢希仁
计算机操作系统
汤小凤
数据库系统概论
王珊
请确认
名称
作者
定价(￥)
版本
年级
ISBN
计算机网络
谢希仁
49
7
3
978-7-121-
计算机操作系统
汤小凤
53
4
3
978-7-560(
取消
确定
总金额
102
确定


LaberaryOS
localhost:8080/#/home/1
订购教材
退订教材
个人信息
反馈
选订成功
退出
可定教材
已定教材
教材查询
名称
作者
定价(￥)
版本
年级
ISBN
出版社
数据库系统概论
王珊
39
5
3
987-7-04-040664-1
高等教育出版社
总金额
0
确定


LaberaryOS
localhost:8080/#/home/1
订购教材
退订教材
个人信息
反馈
退出
可定教材
已定教材
教材查询
名称
作者
定价(￥)
版本
年级
ISBN
出版社
人工智能基础
蔡自兴
39
3
3
978-7-04-046378-1
高等教育出版社
编译原理
陈火旺
39
3
3
978-7-118-02207-0
国防工业出版社
计算机网络
谢希仁
49
7
3
978-7-121-30295-4
电子工业出版社
计算机操作系统
汤小凤
53
4
3
978-7-5606-3350-3
西安电子科技大学出版社

退订

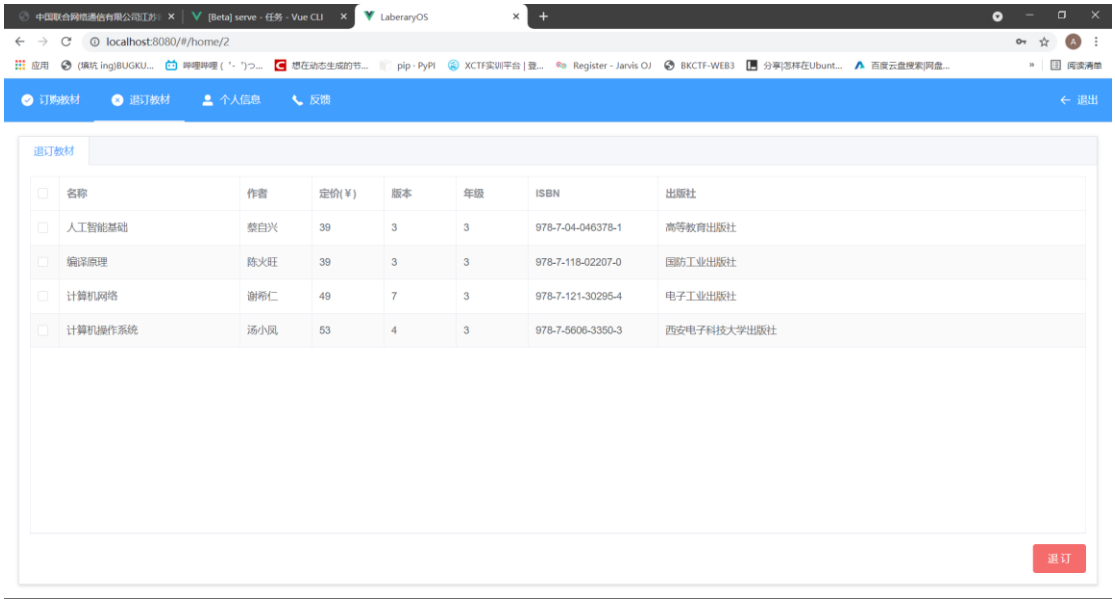
前端实现：退订界面的前端控件也是使用了一个带复选框的列表，被选择的列表项会存在 multipleSelection 中，前端通过 axios 向后端发送 delete 请求，并传递参数。接收后端返回的列表信息并显示出来。

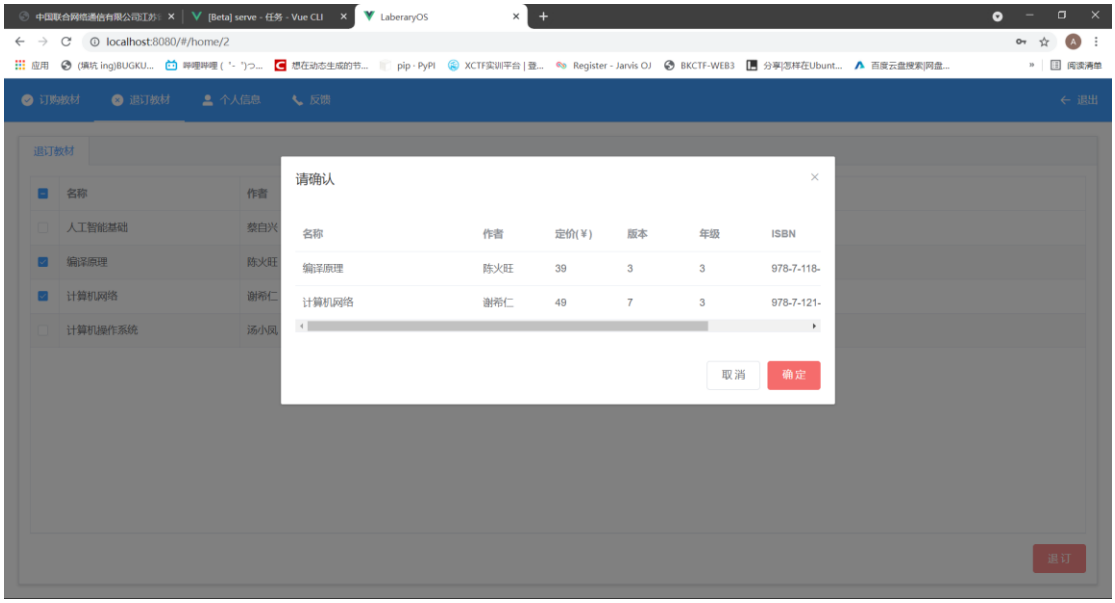
```
multipleSelection: []

confirm(){
    axios.delete("http://localhost:8081/desubscribe",{
        data: this.multipleSelection
    }).then(resp=>{
        this.tableData=resp.data;
    });
    this.dialogTableVisible=false;
    this.$message({message: "退订成功",type: "success"});
}
```

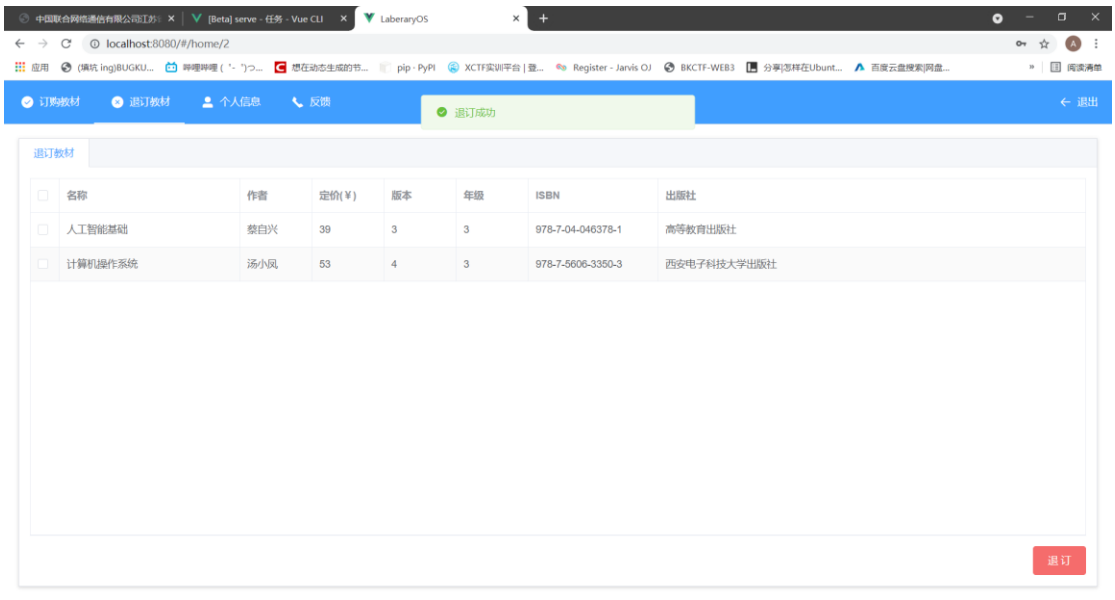
后端实现：接收前端传递的列表信息，使用循环处理，构造 sql 语句将对应的学生学号和教材 isbn 作为条件，从订书信息表中删除学生的订书信息。然后调用查询方法返回该学生已经订购的教材信息列表。

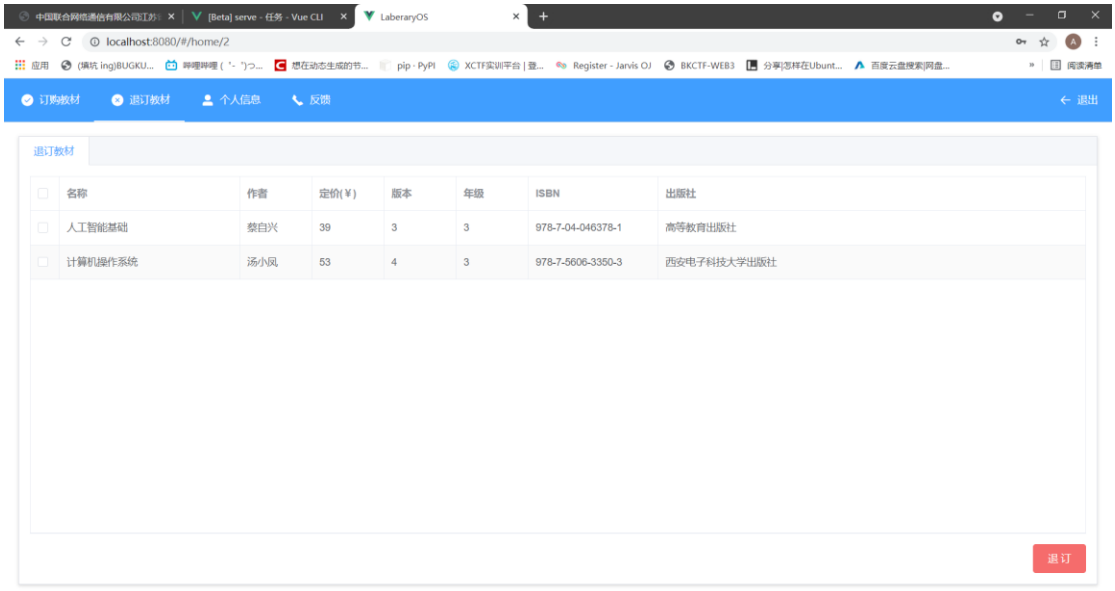
```
@DeleteMapping("/desubscribe")
public List<Map<String,Object>> userDesubscribe(@RequestBody List<Book>
bookList){
    if(user.desubscribe(bookList,stu))
        return user.searchSub(stu);
    else
        return null;
}

public Boolean desubscribe(List<Book> bookList,Student student){
    try{
        String sql="DELETE FROM subscribe WHERE isbn=? AND id=?";
        for(Book book:bookList){
            jdbcTemplate.update(sql,book.getIsbn(),student.getId());
        }
        return true;
    }catch (Exception e){
        System.out.println(e.toString());
    }
    return false;
}
```

演示：


LaberaryOS
localhost:8080/#/home/2
订购教材
退订教材
个人信息
反馈
退出
退订教材
名称 作者 定价(￥) 版本 年级 ISBN 出版社
人工智能基础 蔡自兴 39 3 3 978-7-04-046378-1 高等教育出版社
编译原理 陈火旺 39 3 3 978-7-118-02207-0 国防工业出版社
计算机网络 谢希仁 49 7 3 978-7-121-30295-4 电子工业出版社
计算机操作系统 汤小凤 53 4 3 978-7-5606-3350-3 西安电子科技大学出版社
退订


LaberaryOS
localhost:8080/#/home/2
请确认
名称 作者 定价(￥) 版本 年级 ISBN
编译原理 陈火旺 39 3 3 978-7-118-
计算机网络 谢希仁 49 7 3 978-7-121-
取消
确定


LaberaryOS
localhost:8080/#/home/2
订购教材
退订教材
个人信息
反馈
退订成功
退出
退订教材
名称 作者 定价(￥) 版本 年级 ISBN 出版社
人工智能基础 蔡自兴 39 3 3 978-7-04-046378-1 高等教育出版社
计算机操作系统 汤小凤 53 4 3 978-7-5606-3350-3 西安电子科技大学出版社
退订



6.3 其他功能说明

剩下的一些功能由于篇幅的限制就不一一介绍了，这些功能基本的实现思路是：从前端控件处获得请求参数，通过 axios 向后端对应的 url 发送对应的 http 请求，后端接受请求后保存参数然后通过参数和构造的 sql 语句对数据库进行相应的操作，然后返回结果。

七、 遇到的问题与解决方式

7.1 设置外键时报错的问题

在建表时需要考虑完整性，在订书信息表中，如果教材信息被删除，这里的对应教材的订书信息应该一并被删除，因此这里需要设置一个外键。但是在设置外键时遇到了如下问题：

解决方式：
经过检查后发现，在学生表中 id 字段的字符集为：utf8mb4，排序规则为：utf8mb4_0900_ai_ci，但是在订书信息表中 id 字段的字符集为：utf8，排序规则为：utf8_general_ci。因此导致了不匹配的问题，将两边更改成一样后即可。

7.2 前端向后端发送 http 请求时的跨域问题

在前端通过 axios 向后端发送 http 请求时可能会遇见跨域问题，如：

```
Access to XMLHttpRequest at 'http://localhost:8081/login?id=161820318&passward :8080/#/:1
=123456' from origin 'http://localhost:8080' has been blocked by CORS policy: No 'Access-
Control-Allow-Origin' header is present on the requested resource.
```

解决方式：
在服务端自定义 WebMvc 的配置，使其可以支持跨域资源共享：

```
@Configuration
public class WebMvcConfig implements WebMvcConfigurer {
    @Override
    public void addCorsMappings(CorsRegistry registry){
        registry.addMapping("/**")
                .allowedMethods("POST","GET","PUT","DELETE");
    }
}
```

八、 实验心得体会

这次实验让我了解到数据库在对数据的存储，操作方面的重要作用，在之前的课程设计中编写的系统中，涉及的关系型数据只能定义相应的数据结构存储然后手动地对文件进行操作，其实现过程非常复杂，也容易出现各种各样的 bug。而数据库的使用可以很好的解决这些问题，其中对数据的操作只需要通过 sql 即可完成。在实验过程中，我学习了 mysql 数据库从安装到建表到操作的全部过程，学习了 vue 前端框架和 springboot 后端框架，对这些知识的掌握对以后的学习、工作大有好处。