中国科学技术大学大学数据库项目简介

课程名称：数据库基础

项目名称：吴氏全书

学院：网络空间安全学院

班级：王小谟网络空间安全英才班

学生姓名：吴立凡

学号：PB19061323

目录

项目简介

本次实验我制作了一个基于 apache，php，mysql 的 web 服务器，实现了一个简单的图书管理系统，目前该项目已经被部署到云服务器上，可通过 http://www.cggwz.com.cn/wusar/book_manage_system/login.html 来直接用浏览器访问，在此特别感谢陈巩固同学的云服务器。

前端界面（成果展示）

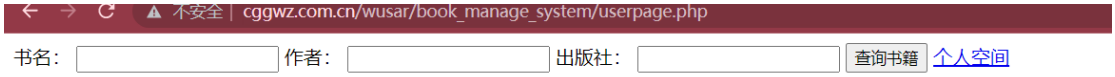
首先看一下前端界面：

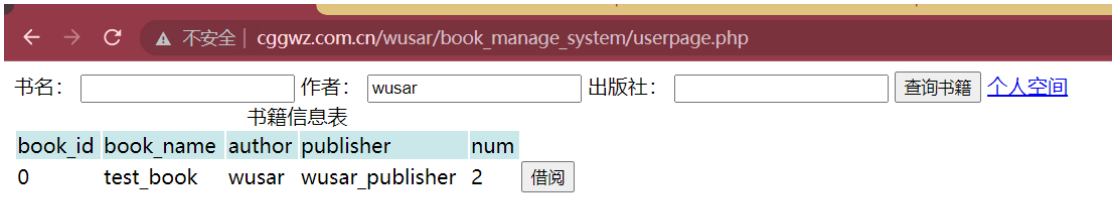
登陆界面：

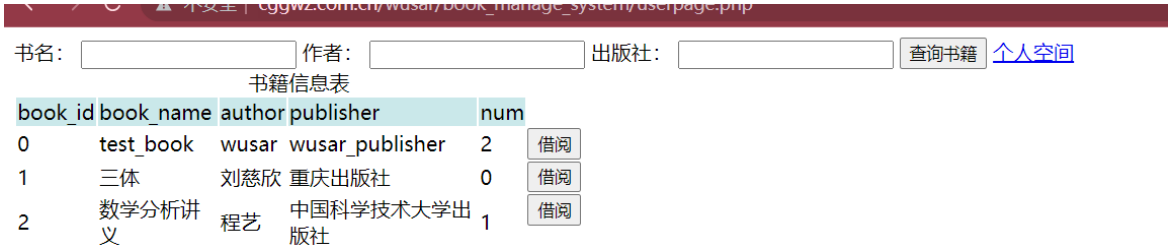
输入用户名和密码登录，有三种登陆模式，分别是普通用户、管理员和访客，普通用户拥有借书和还书的权限，管理员拥有增删改书籍信息的权限，访客还没做完。只能有一个管理员，他的用户 id 是 0。
目前只有一个普通用户，用户名是 wusar，密码是 123456。
我们尝试以普通用户登录：

书籍查询与借阅界面：

输入书籍信息，点击查询即可获得查询结果：







num 是剩余书籍数，点击借阅即可借阅书籍。

随后书籍数减一。
借阅 num 为 0 的书籍会借阅失败。

书籍归还界面

点击“个人空间”即可跳转到个人借阅的书籍的信息表:

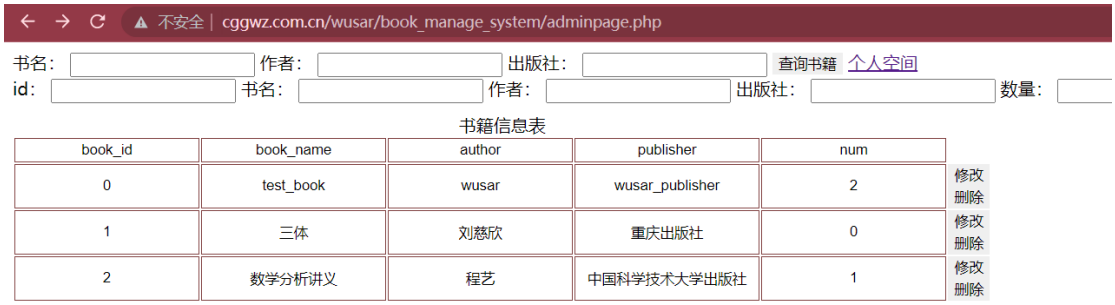
点击“归还”归还书籍。

普通用户能执行的操作就是这些。

管理员界面

管理员的用户名密码都是 admin，我们将登陆界面的下拉选项框切换到管理员，再登陆：

点击“查询书籍”即可对所查询到的书籍进行修改或删除操作：



点击删除将直接删除这本书，如果有人正在借阅的话无法删除。

点击修改后，表格文本会变为输入框：

直接在输入框上修改，点击确认即可提交修改信息：

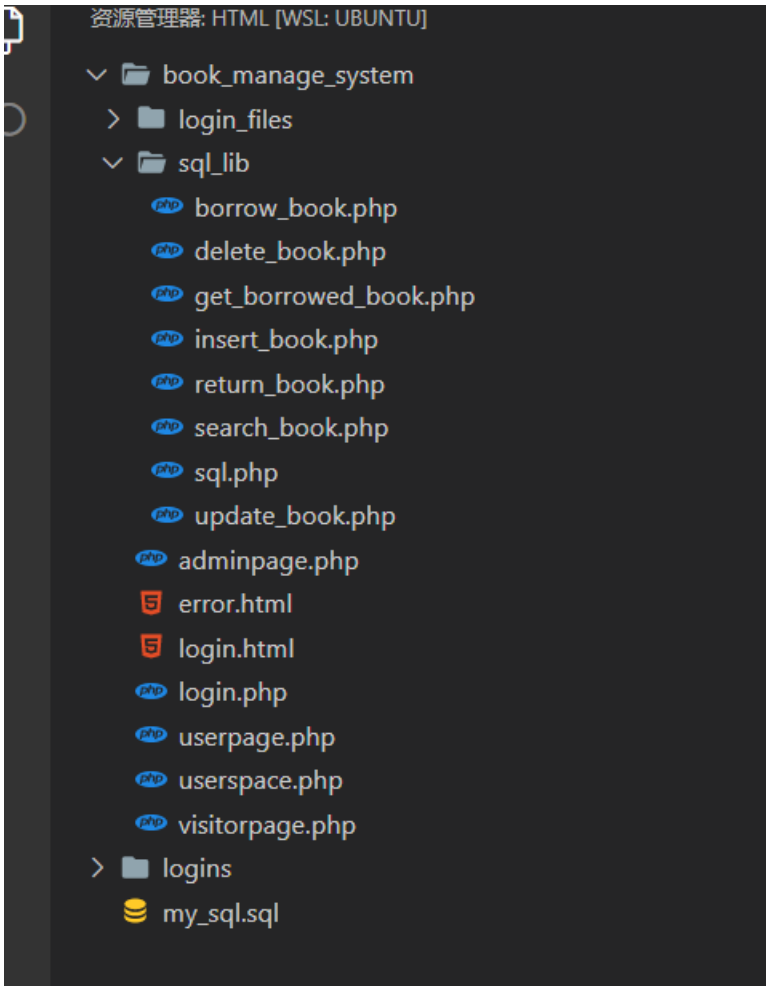
在增加书籍左边的输入框中输入书籍信息，点击“增加书籍”即可增加一条书记记录（id 和数量不能为空）：

项目的前端大概就是这样。

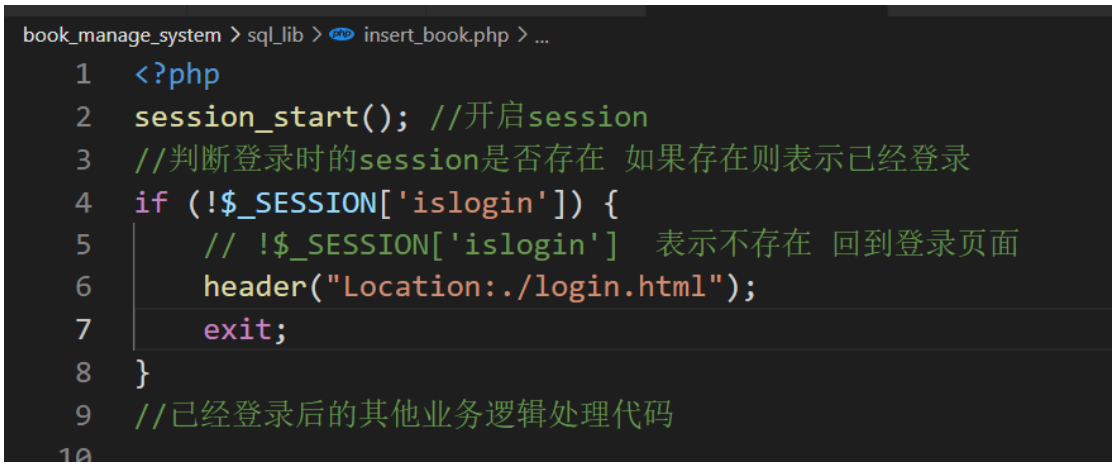
项目后端

项目的目录树

后端方面，项目的目录树如下：



数据库创建使用的 sql 语句可见于 my_sql.sql 文件。
sql_lib 中是一系列用 php 编写的数据库操作库，包含了书籍的查询、借阅、修改、增加、删除，以及用户的登录等操作。
login.html 是登陆界面的网页，userpage.php 是用户界面的网页，userspace.php 是用户个人空间的网页
尝试直接访问这些网页时，会因为没有服务器的 session 被跳转到 login.html 界面：



具体的文件：

数据库的结构

数据库创建使用的 sql 语句可见于 my_sql.sql 文件。
这里我只复制最关键的创建表格：

```
create table user_index (
    _user_id int,
    _username VARCHAR(25),
    _password VARCHAR(25),
    email VARCHAR(25),
    primary key(_user_id)
);

create table book_info(
    book_id int,
    book_name VARCHAR(25),
    author VARCHAR(25),
    publisher VARCHAR(25),
    num int,
    primary key(book_id),
    CONSTRAINT check(num>=0)
);

create table borrow(
    -- borrow_id int,
```

```
    _user_id int,
    book_id int,
    borrow_date datetime,
    return_date datetime,
    -- primary key(borrow_id),
    foreign key(_user_id) references user_index(_user_id),
    foreign key(book_id) references book_info(book_id)
);
```

只用到了三张数据表，分别表示用户信息、书籍信息、借阅信息。用户信息和书籍信息都有一个主键_user_id 和 book_id，借阅信息有两个外键_user_id 和 book_id，但是并没有主键！这也是本系统最大的问题，这个问题我在后面有介绍。

登陆界面 login.html

这个界面的结构与某个网站十分类似，可以点击“找回/修改密码”来看看到底是哪个网站，如有雷同，纯属巧合。
点击登录后，form 表单会将用户名和密码传给服务器：

```
<form
  id="submit_form"
  class="loginForm form-style"
  style="height: 210px"
  method="post"
  action="./login.php"
  accept-charset="UTF-8"
>
  <!-- 账户 -->
```

使用 http post 方法传给服务器的 login.php 来验证身份。
Login.php 则通过传来的信息，使用 mysqli 接口查询
$sql = "select _user_id from user_index where _username=\"$username\" and _password=\"$password\";";

```php
if ($selector == "ordinary_user") {
    $username = $_REQUEST['username'];

    $password = $_REQUEST['password'];

    try {
        $conn = start_sql();
        // Check connection
        $username = mysqli_real_escape_string($conn, $username);
        $password = mysqli_real_escape_string($conn, $password);
        $sql = "select _user_id from user_index where _username=\"$username\" and _password
        // echo $sql;
        $result = mysqli_query($conn, $sql);
        if (mysqli_num_rows($result) > 0) {
            $row = mysqli_fetch_assoc($result);
            session_start();
            $_SESSION['islogin'] = true;
            $_SESSION['_user_id'] = $row["_user_id"];
            $_SESSION['sql'] = $sql;
            echo "登陆成功!";
```

查询到了则设置 session：
$_SESSION['islogin'] = true;
$_SESSION['_user_id'] = $row["_user_id"];

并跳转到对应的界面。

用户界面 userpage.php

点击查询则会触发函数 search_book:

```
function search_book() {
    var book_name = document.getElementById('book_name').value;
    var author = document.getElementById('author').value;
    var publisher = document.getElementById('publisher').value;
    $.ajax({
        type: "post",
        url: "./sql_lib/search_book.php",
        data: {
            book_name: book_name,
            author: author,
            publisher: publisher
        }, //提交到demo.php的数据
        dataType: "json", //回调函数接收数据的数据格式
        success: function(msg) {
            console.log(msg); //控制台输出
            var data = '';
            if (msg != '') {
                data = eval(msg); //将返回的json数据进行解析，并赋给data
            }
            var temp = document.getElementById("book_info_table");
            if (temp != null)
                document.getElementById("book_table").removeChild(temp);
            document.getElementById("book_table").appendChild(buildTable(data));
        },
        error: function(msg) {
            console.log(msg);
```

首先使用 ajax 方法，将 php 请求上传至服务器，服务器对 PHP 请求进行解析，生成相应的 sql 查询语句（三个查询关键字都可以为空）：

```
try {
    $conn = start_sql();
    // Check connection
    $book_name = mysqli_real_escape_string($conn, $book_name);
    $author = mysqli_real_escape_string($conn, $author);
    $publisher = mysqli_real_escape_string($conn, $publisher);

    $sql = "select * from book_info where ";
    if ($book_name != "")
        $sql = $sql . "book_name=\"$book_name\" and ";
    if ($author != "")
        $sql = $sql . "author=\"$author\" and ";
    if ($publisher != "")
        $sql = $sql . "publisher=\"$publisher\" and ";
    $sql = $sql . "1=1;";
    // echo $sql;
    $result = mysqli_query($conn, $sql);
    mysqli_close($conn);
} catch (mysqli_sql_exception  $e) {
```

并将查询结果以 json 格式返回

返回后，search_book()函数调用 buildTable(data)函数建立表格，具体细节见代码附件。

点击“借阅”后则会将 php 请求发送到服务器的 borrow.php 文件，先将表 book_info 中的对应 book_id 的记录的 num 减去 1，如果成功则说明有 num>0（num 加入了非负的约束条件）。然后向 borrow 表插入一条包含借阅者 id、书籍 id 以及借阅归还时间的记录，归还时间默认为借阅时间十天以后。

个人空间 userspace.php

点击“个人空间”将会跳转到归还书籍的界面，在这里点击对应的书籍后面的“归还”按钮就可以归还对应的书籍。点击归还将会发送一条 php 请求到 return_book.php。先查询 borrow 表，判断这是不是一条合法的借阅记录，然后删除对应的借阅记录，将 book_info 表中对应 book_id 的 num 加一。

管理员界面 adminpage.php

然后是 adminpage.php,管理员界面。点击“插入”和“删除”将会分别发送一条 php 请求到 insert_book.php 和 delete_book.php 来对数据库进行相应的 insert 和 delete 操作：

```
$sql = "INSERT INTO book_info VALUE($book_id,'$book_name','$author','$publisher',$num);";
$sql = "delete from book_info where book_id=" . "$book_id" . ";";
```

删除书籍时由于有 borrow 表的外键约束，在有人借阅时无法删除。

修改操作时，点击“修改”将会调用 javascript 的 alter_book 方法，删除掉表中对应行的 input 元素的 readonly 属性，然后就可以直接修改，并将这个按钮改成确认按钮。再次点击确认按钮就可以再次向 input 加入 readonly 属性，将按钮改为“修改”，并调用 update_book 方法，向服务器的 update_book.php 发送请求，调用

```
$sql = "UPDATE book_info set book_name='$book_name',author='$author',publisher='$publisher',num='$num' where book_id=$book_id;";
```

向数据库进行修改操作。

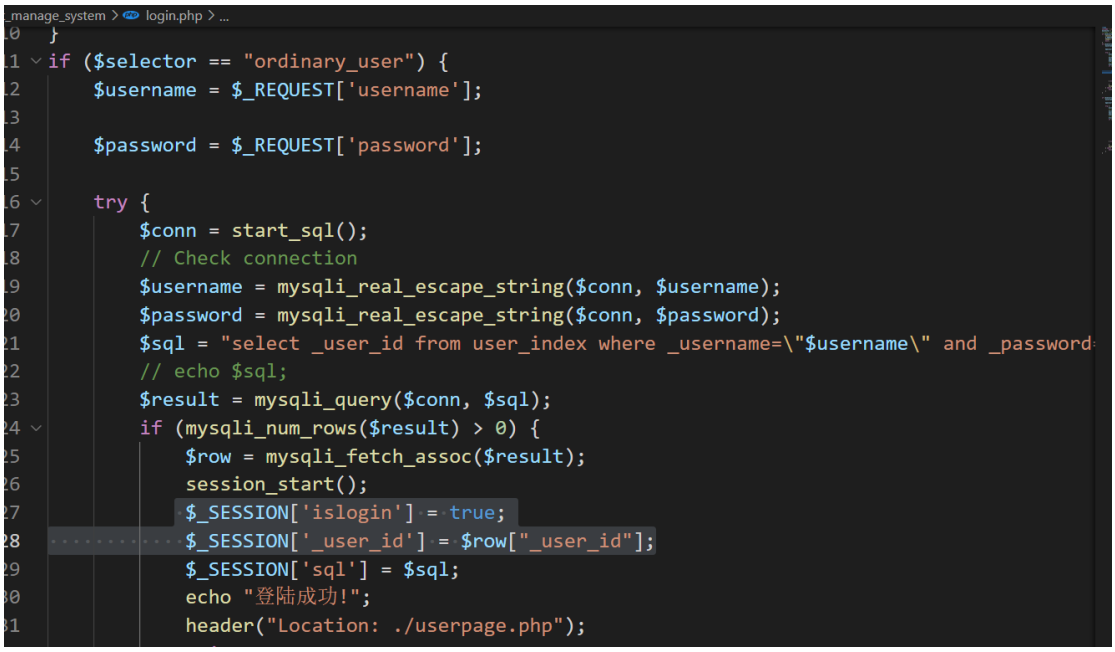
项目存在的问题

一个关于数据库表 borrow 的漏洞

这里有一个很严重的漏洞，就是 borrow 表是没有主键的。Borrow 表有两个外键，book_id 和 user_id，但是由于没有主键，就有可能产生完全相同的两条记录！在后面归还删除借阅记录的时候，就有可能发生一次删除多条记录的情况，这里只能通过借阅归还时间来区分，但是仍然可以通过脚本同时借阅两本书，创造两条完全相同的记录。
由于发现这个漏洞的时候已经很晚了，我也没有时间去改了，就这么用着吧。

Sql 注入漏洞

注意到，我对所有的 sql 查询语句的值，都执行力 mysqli_real_escape_string 进行转义操作，这是为了防止 sql 漏洞。
考虑登陆界面的后端 php：

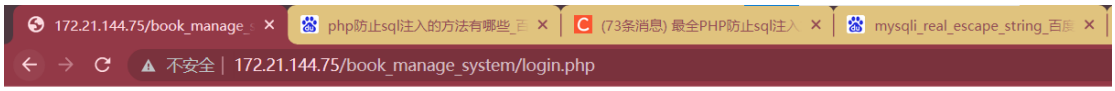


如果将转义的两行代码$username = mysqli_real_escape_string($conn, $username);
$password = mysqli_real_escape_string($conn, $password);删去，那么在登陆时，如果输入这样的用户名和密码：（密码也为 1" or "1"="1）

那么就可以直接登录!
因为此时的 sql 语句是
select _user_id from user_index where _username=“1” or “1”=”1” and _password=“1” or “1”=”1”;
条件是恒为真的

0

而加入 sql 转义语句后，sql 查询语句就变成了这样:



select _user_id from user_index where _username="1\" or \"1\"=\"1" and _password="1\" or \"1\"=\"1";0 结果

当然，这些都没有解决最根本的问题，sql 注入漏洞仍然存在，不过已经能够防范最基本的 sql 注入问题了。

实验总结

这一次实验一共花费了我一周半的时间，我首先花费了三天的时间来学习 php 、css、javascript、html 然后才能开始实验。总的来说学到的内容还是相当多的。直接从前端干穿到后端，相当于是个全栈工程师了。

我确实是头一次做这样的项目，有非常多很麻烦的细节，可以说跟数据库上课内容相关的内容是这个项目里面最简单的问题了。实践起来才能知道这些真实的项目到底有多麻烦。上课的内容还是以理论为主，但是一旦到实践中就会有很多很复杂的技术问题等待解决。

不过我对这一次的工程还是相当满意的，算是做出来一个能有实践意义的项目吧。之前的写的所有的项目基本上都是后端的项目，没有哪个是真正能拿来用到实际应用中的，这个项目其实已经可以应用于实际了。

这个项目的一个特点就是很注重系统的安全性。从登录认证到各个界面，session 和 cookie 的机制能保证用户在登陆一次后不用再次输入密码，同时也能将那些没输入过密码的用户拒之门外。

不过由于时间问题，这个项目还有很多内容没法完成。比如用户的账号注册、更改密码，以及用户相关的操作，都没时间去做了。还有一些漏洞，bug，看看以后有没有机会在改吧。