**数据库原理及安全实验四**

|  |  |  |  |
| --- | --- | --- | --- |
| **班级** | 周二上午3-5节 | | |
| **实验名称** | 数据库应用系统的设计与实现 | **日期** | 2022.6.1 |

|  |  |
| --- | --- |
| **队员** | **学号** |
| 崔浩然（队长） | 2020302181062 |
| 严石 | 2020302181047 |
| 靳晨浩 | 2020302181051 |

**【实验内容及要求】**

1. 熟悉开发实现一个数据库应用系统的基本流程。

2. 熟悉数据库操作以及应用与数据库交互过程的编写。

3. 掌握使用数据库作为工具完成访问控制的方法。

4. 掌握防止SQL注入的基本方法。

5. 掌握数据库并发、数据备份与恢复等功能的实现。

**【实验平台】**

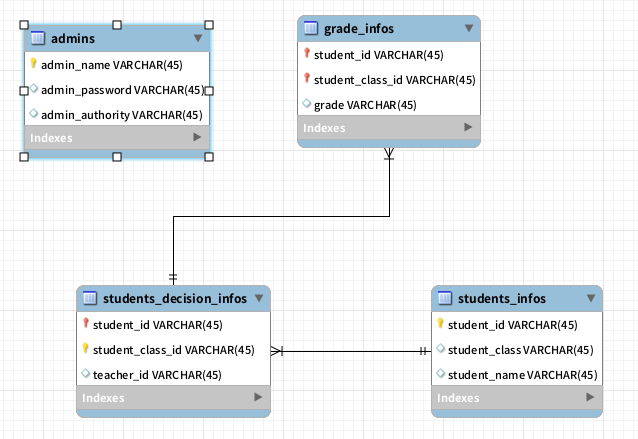
Ubuntu 20.04

Pycharm

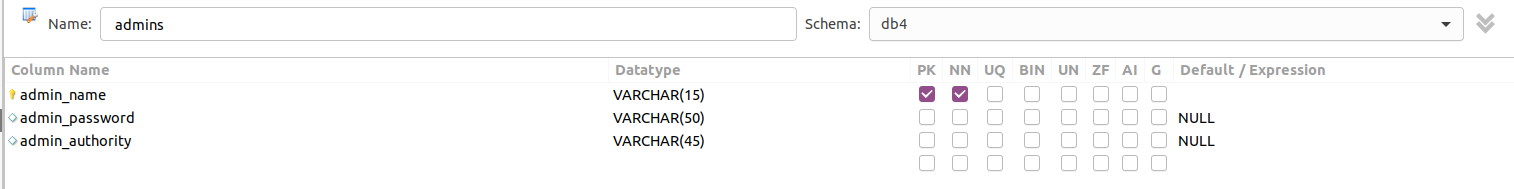
Flask

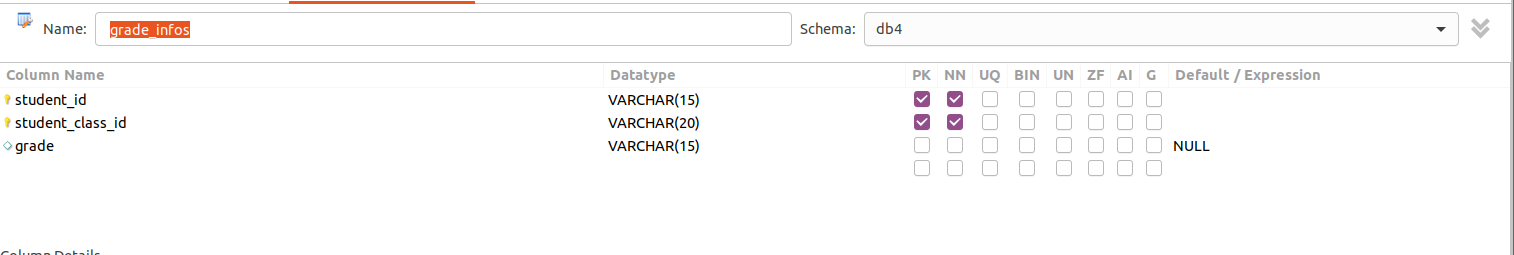
**【实验步骤】**

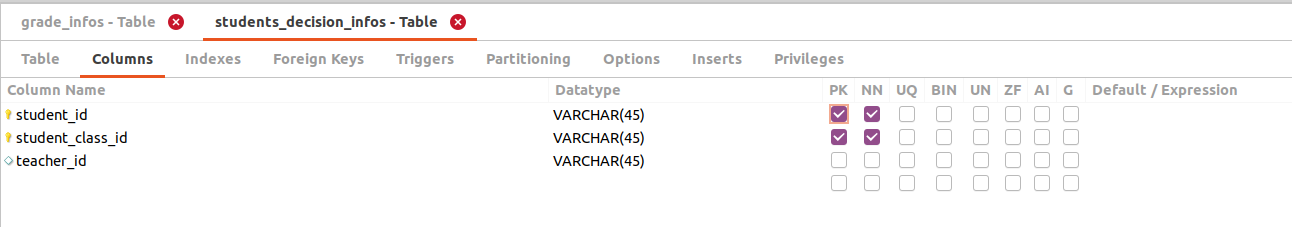
设计相应ER图：

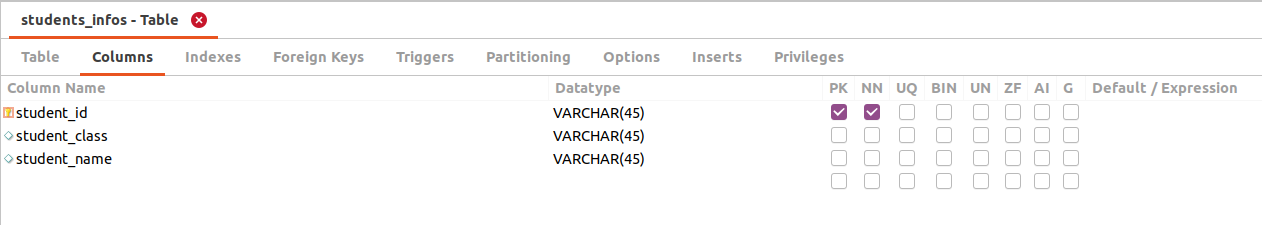


对应表结构：







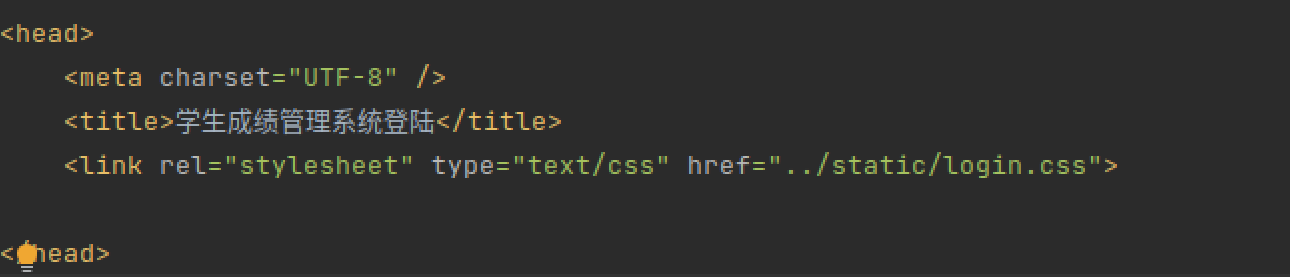


1.编写登陆及找回密码界面前端HTML（login.html,findCrpto.html,resetcrpto.html）界面及后端python flask代码：

1.1前端HTML代码编辑：

1.1.1 login.html

A. 网页界面头部部分，颜色背景图片在css文件中给出：

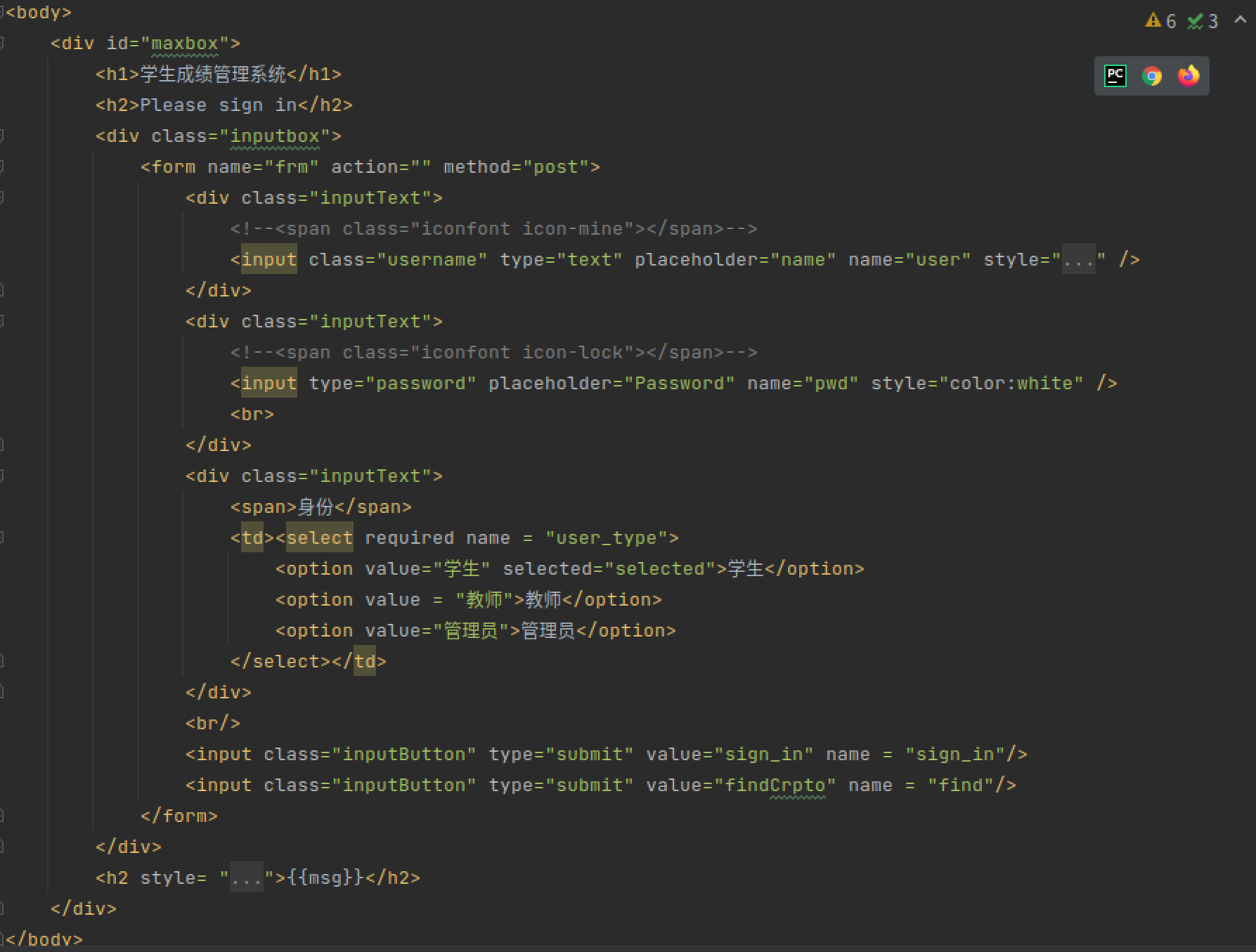


B.body部分主要有三个inputText以及两个inputButton:

其中inputText分别用于输入用户名，密码，以及身份（表单形式）

而两个inputButton分别实现系统入口的跳转按键与找回密码的跳转按键，具体功能实现会在py文件中给出

同时msg要返回一些信息



1.1.2 findCrpto:

A.网页界面头部部分,与login界面保持一致，为了前端的美观与一致:



B.body部分：

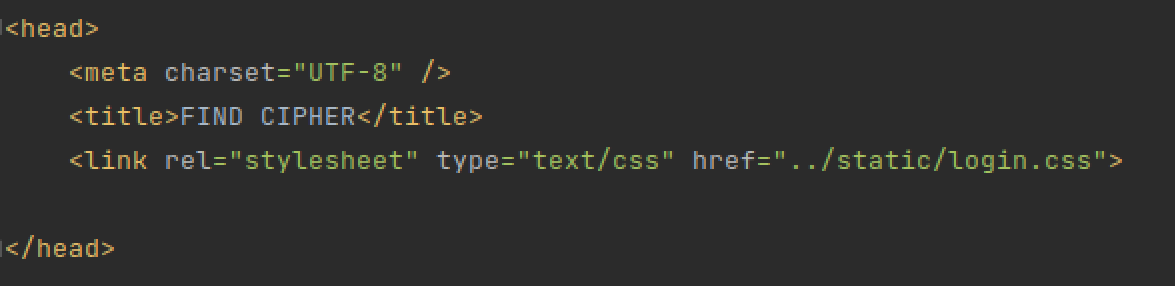
与之前大致相似，但这次只有一个inputText，用于输入找回密码的用户名，之后要跳转到下一个界面重制密码

msg用于返回是否通过验证



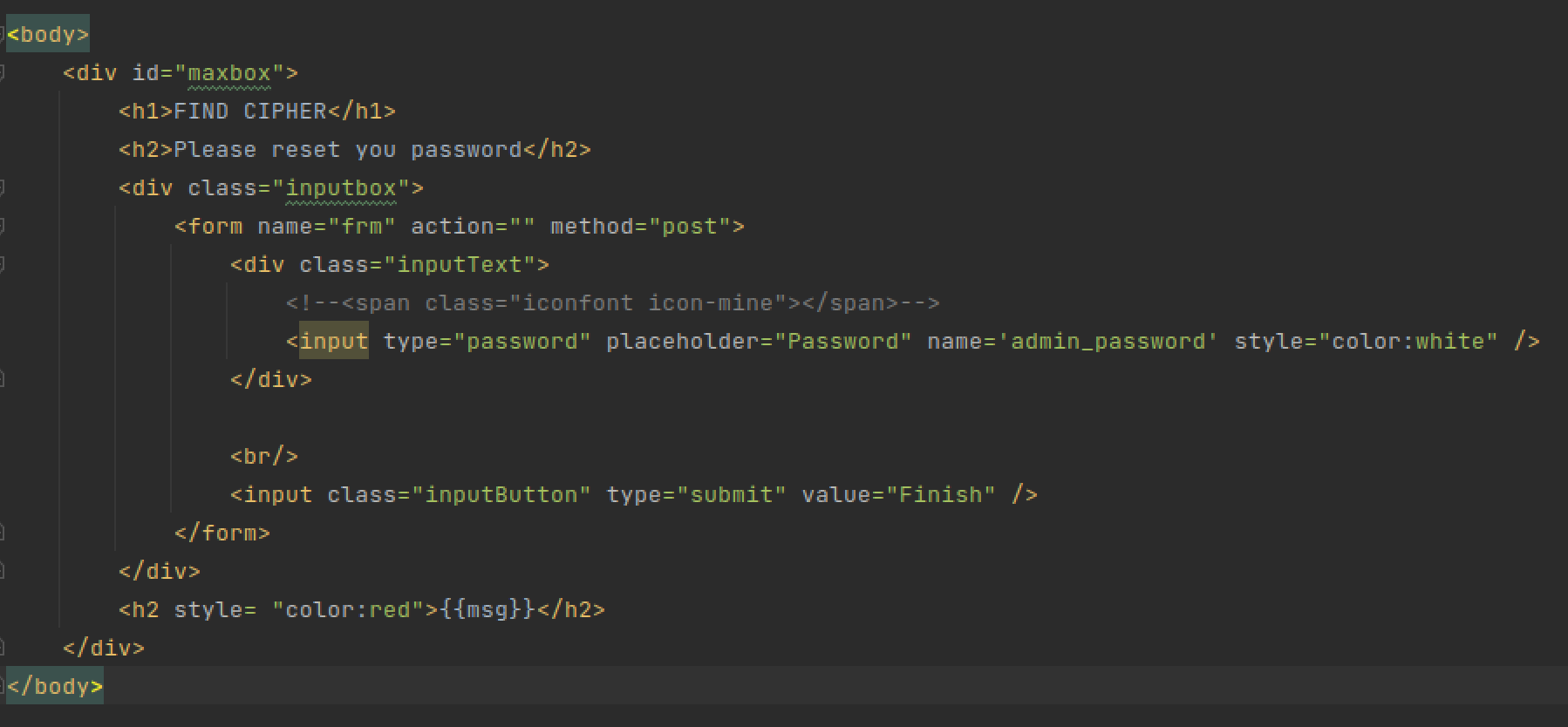
1.1.3：resetcrpto:

1. 网页界面头部部分,与login界面保持一致，为了前端的美观与一致:



1. body部分：

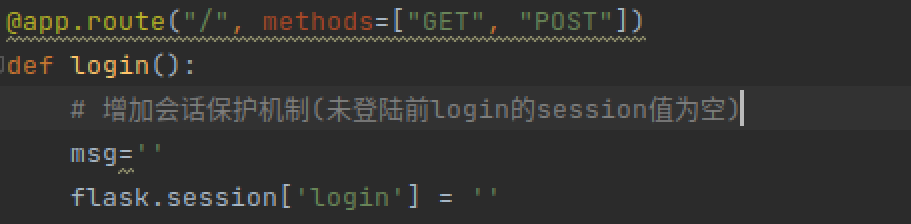
逻辑设计与之前相似，msg返回错误信息



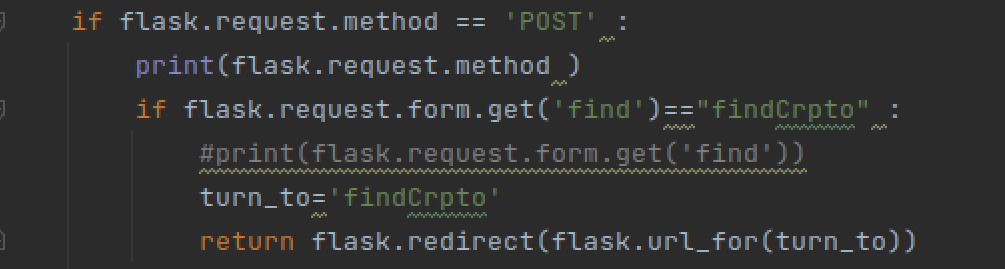
1.2 python flask框架下后端代码：

1.2.1 login():

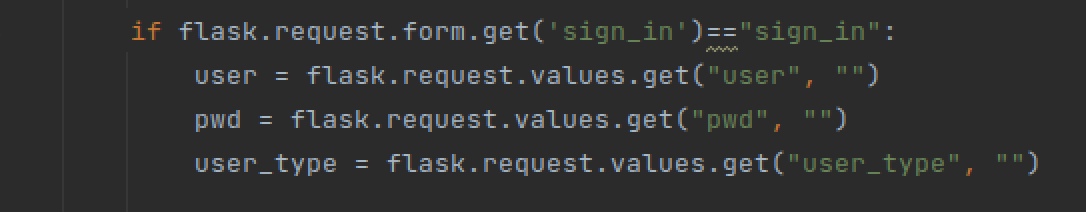
A.增加会话保护机制(未登陆前login的session值为空)



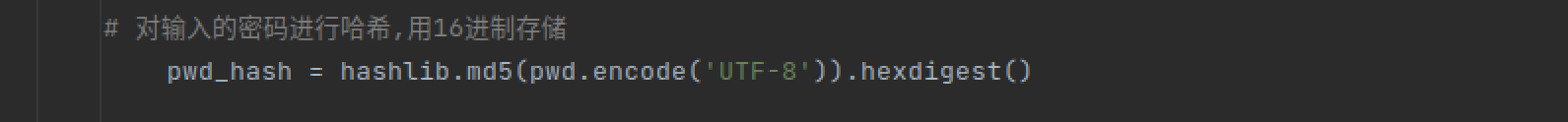
B.在post的请求模式下，判断是否点击找回密码，如果是则跳转界面：



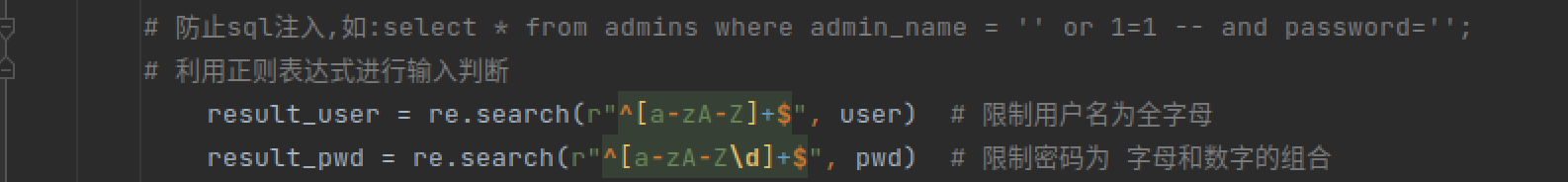
C. 判断是登陆请求后进行获取三个输入：



D. 对输入密码进行md5哈希运算，用于对比系统存储的密码（存储的是md5哈希值，one way function函数保证密码的安全）



E.对输入的用户用户名及密码进行正则表达式判断，防止sql注入

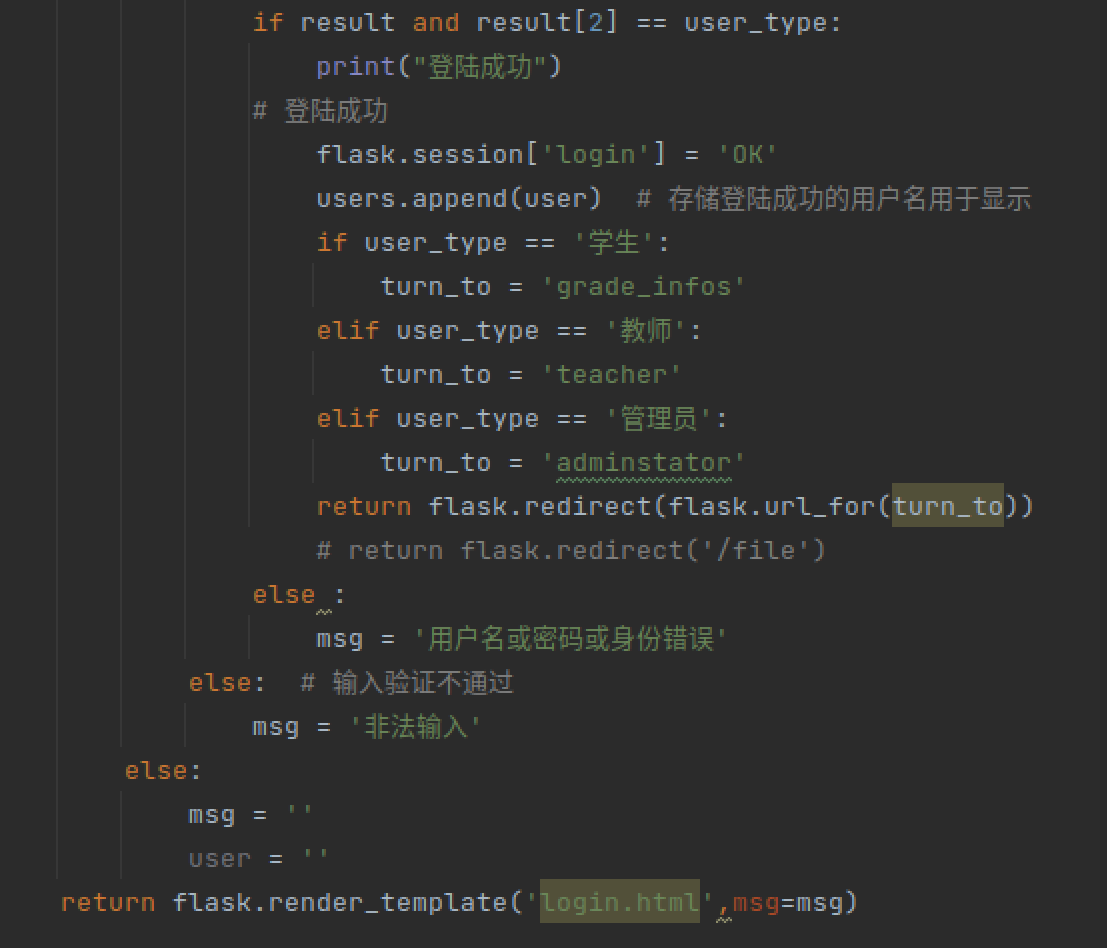


F.正则表达式验证通过后，判断身份，如果是初始管理员密码不经过哈希，否则密码哈希后查找，并返回数据库中的查找结果



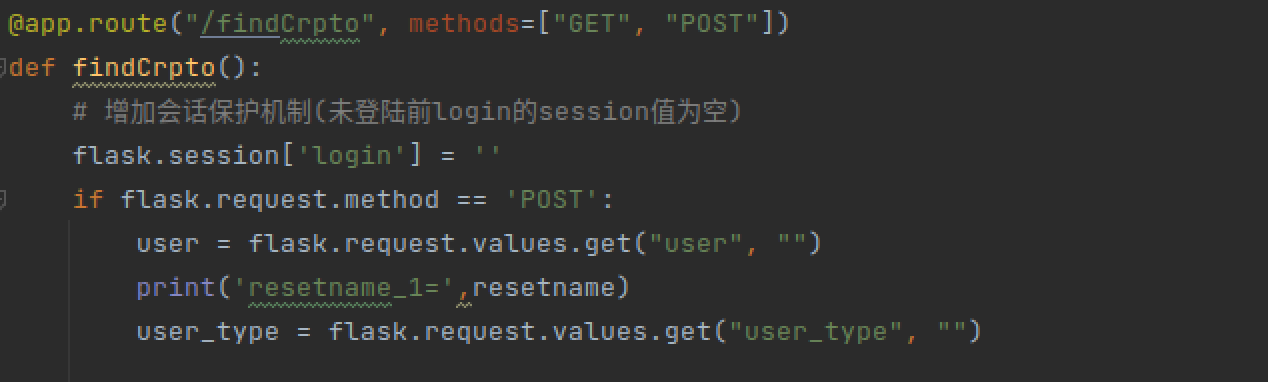
sql语句操作未截取完全

G.根据查询到的结果去跳转不同的界面，以及返回不同if分支的失败信息

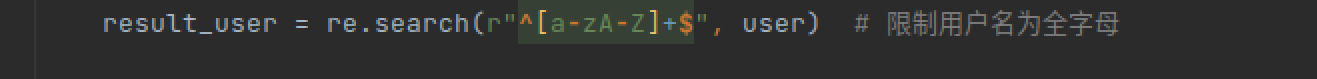


1.2.2 findCrpto():

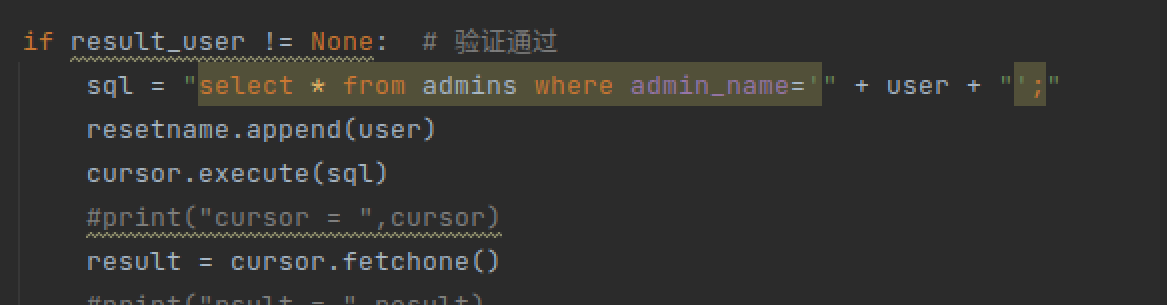
A.在POST状态下获取输入：



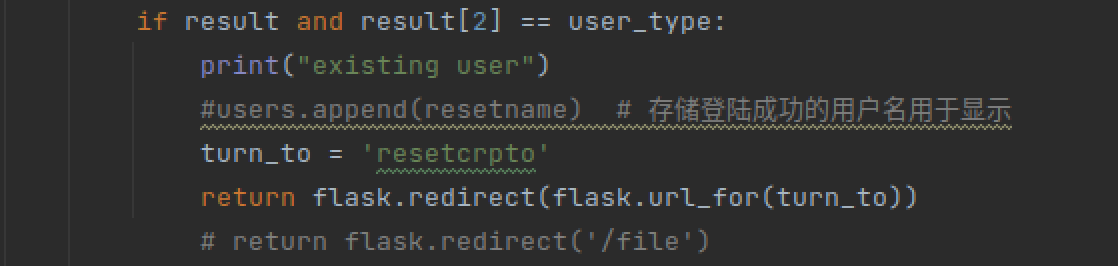
B.正则表达式防注入：



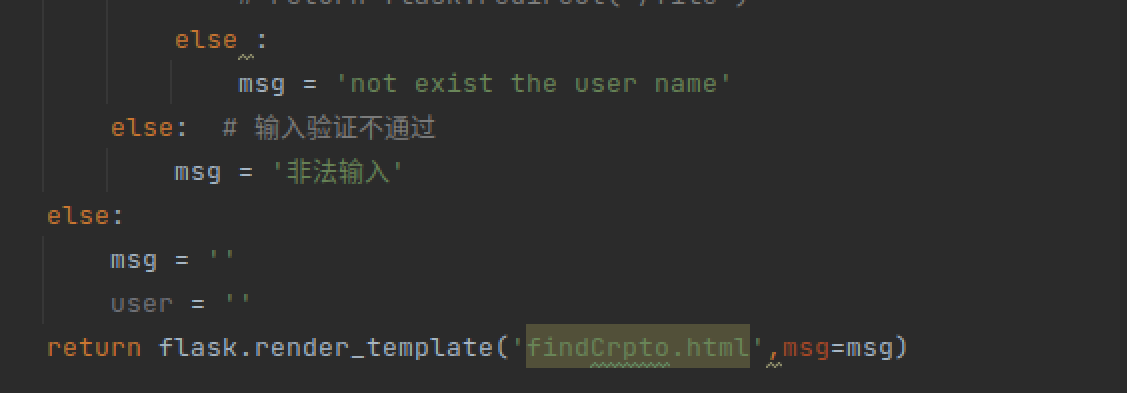
C.验证通过查询用户名信息，并返回查询结果：



D.查询成功则跳转到下一界面重制密码：

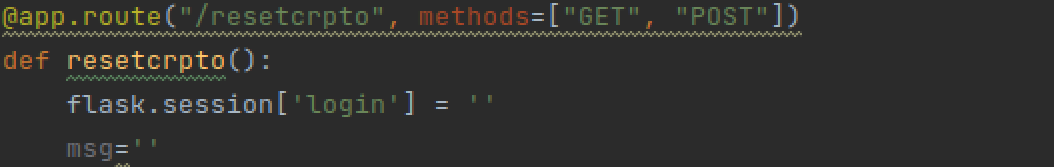


E.返回各if分支失败结果：

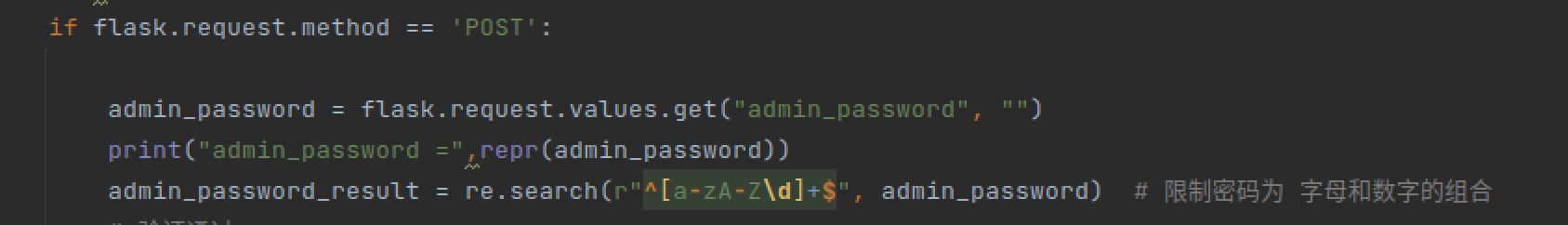


1.2.3 resetcrpto():

A.会话保护机制：



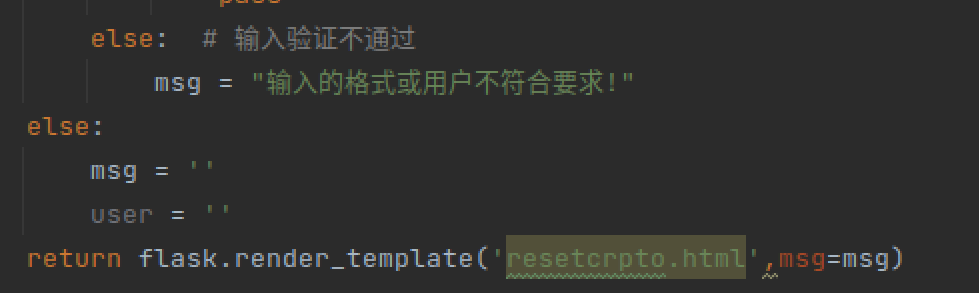
B. 在post状态下获取输入，并进行正则表达式防止注入：



C.验证通过对密码取哈希，并通过sql修改存储表相应位置的密码：



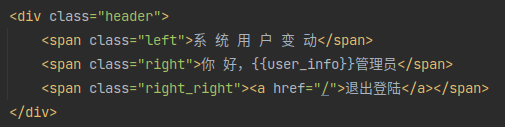
D. 返回错误信息



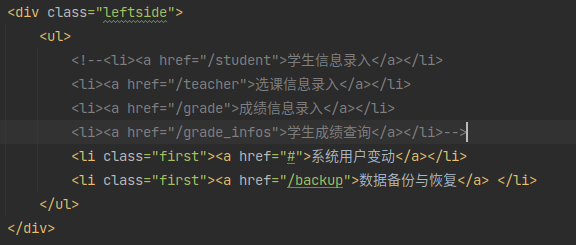
2.编写管理员界面前端HTML（adminstator.html）界面及后端python flask代码（adminstator()）：

2.1前端HTML代码编辑：

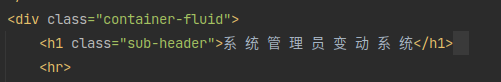
A.网页界面头部部分，‘user\_info’表示正在操作的管理员名字：



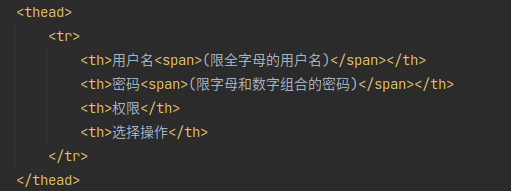
B.页面左侧跳转栏，管理员能够对“系统用户变动”和“数据备份与恢复”两个界面进行跳转：



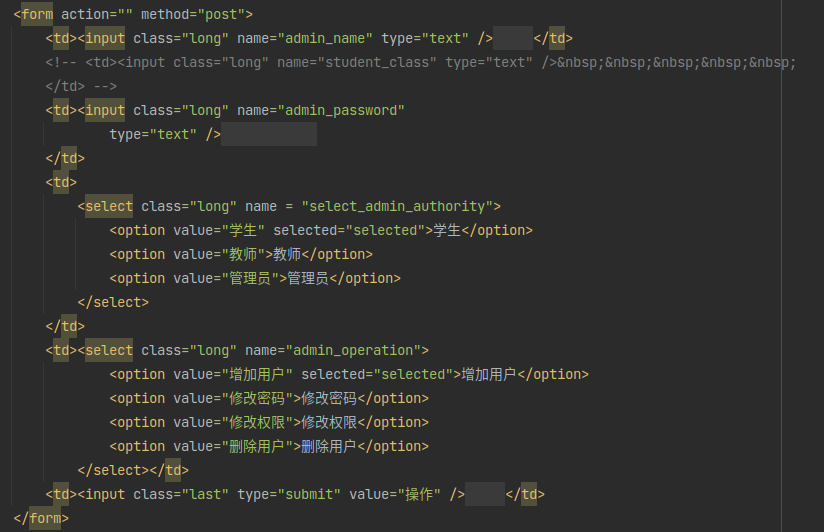
1. 界面功能描述：



1. 数据表格标题：



1. 所要提交的表单数据输入处，分别有要操作的用户姓名、用户密码、授予的权限以及所要进行的操作的选择

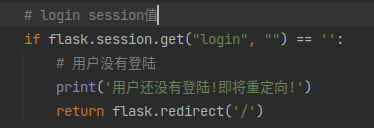


1. 查询结果展示处，将所有用户的名称、密码（hash值）及权限展示



2.2 python flask框架下后端代码：

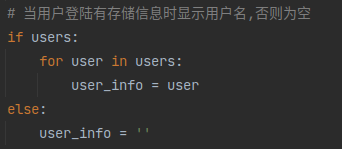
A.增加会话保护机制(未登陆前login的session值为空)



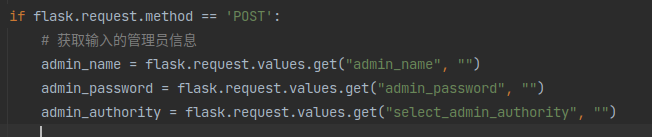
1. 获取用户信息



1. 记录当前登录的用户的用户名



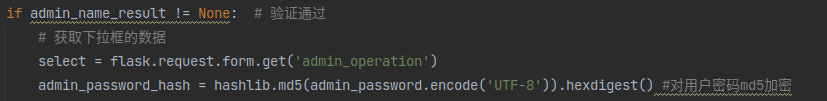
1. 获取从网页输入的信息



1. 对输入的用户用户名及密码进行正则表达式判断，防止sql注入

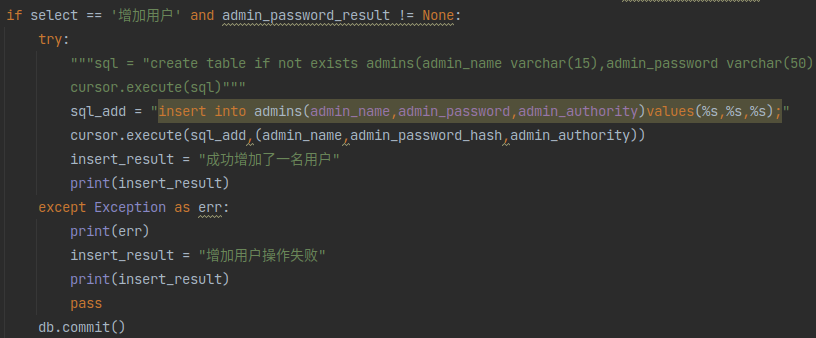


1. 正则表达式验证通过后，对输入的用户密码进行md5加密



1. 通过网页的下拉框，选择对应操作

增加用户：



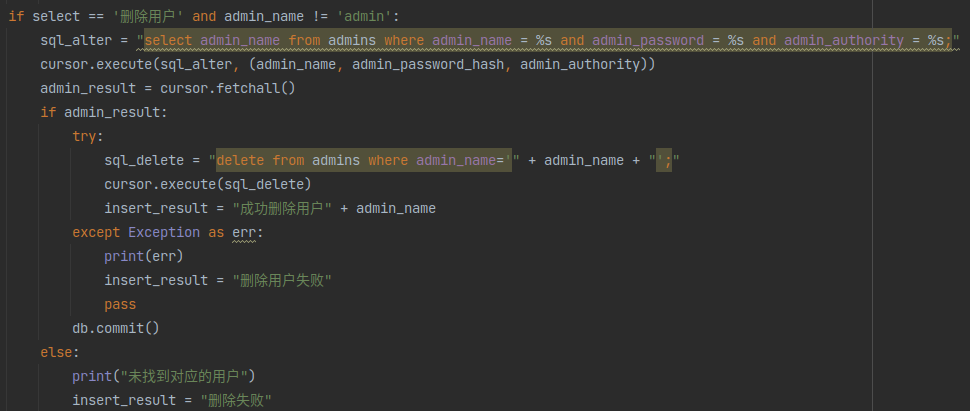
修改密码：



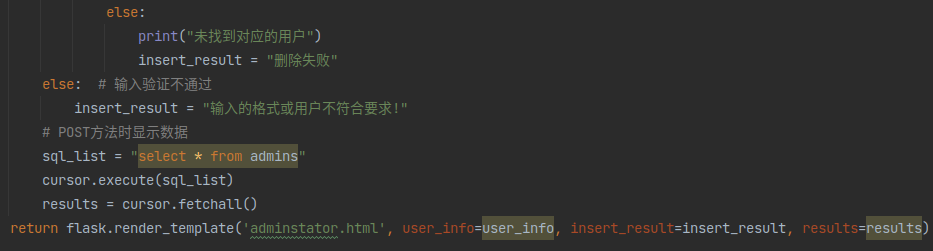
修改权限：



删除用户：

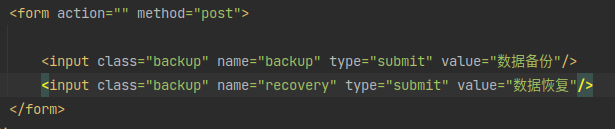


1. 验证不通过则向网页返回错误信息



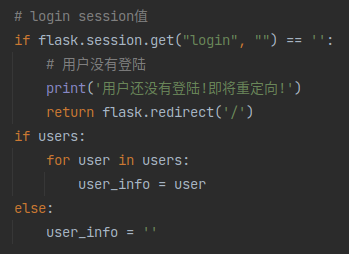
3.数据库备份与恢复网页HTML代码编写及后端代码编写

3.1 HTML代码介绍。使用表单，创建两个按钮，分别进行数据库备份与恢复



3.2 python flask框架后端代码编写

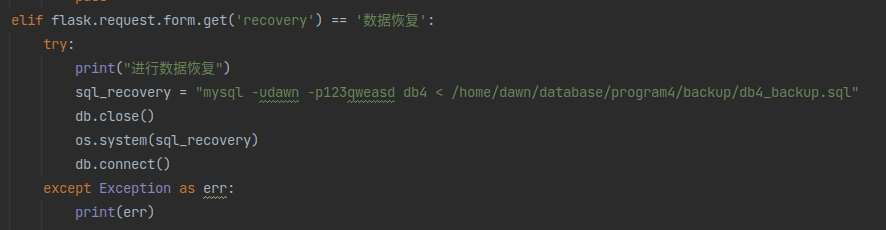
A.session会话保护及获取当前管理员名称



1. 进行数据备份，将生成的db4\_backup.sql备份文件存入/home/dawn/database/program4/backup/路径下



1. 进行数据恢复，从/home/dawn/database/program4/backup/路径下恢复所备份的sql文件

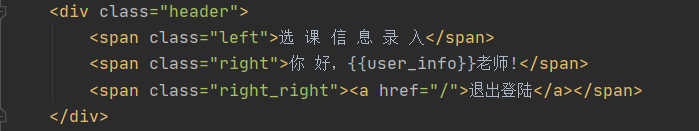


4.编写教师功能前端HTML界面及后端python flask代码：

由于教师功能的三个界面实现方式完全相同，故在此以选课信息录入功能为例进行分析。

4.1前端HTML代码编辑：

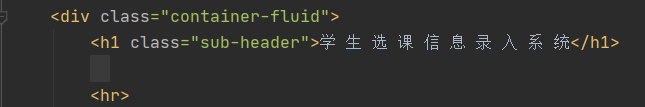
A.网页界面头部部分，‘user\_info’表示正在操作的教师名字：



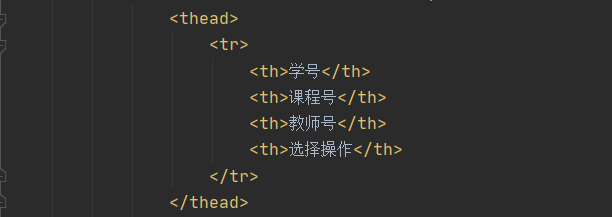
B.页面左侧跳转栏，教师能够在“学生信息录入”、“选课信息录入”与“成绩信息录入”三个界面之间进行跳转：



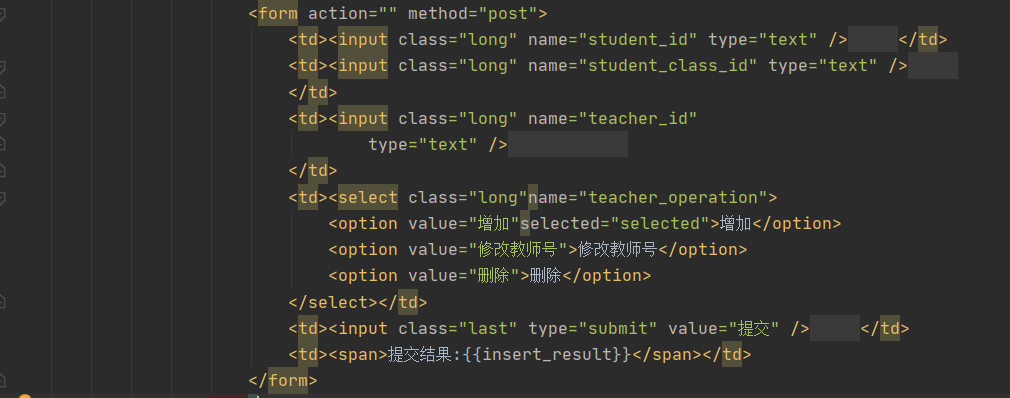
C.界面功能描述：



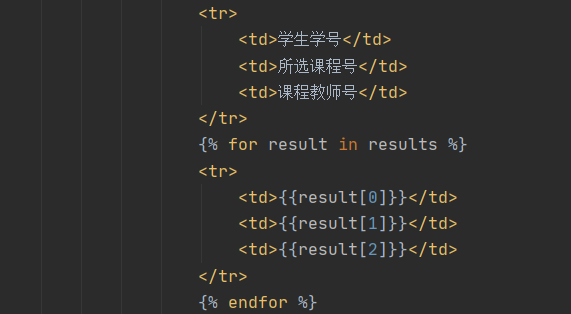
D.数据表格标题：



E.所要提交的表单数据输入处，分别有要操作的学号、课程号、教师号以及所要进行的操作的选择

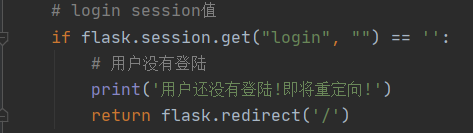


F.查询结果展示处，将所有的学号、教师号及课程号进行展示

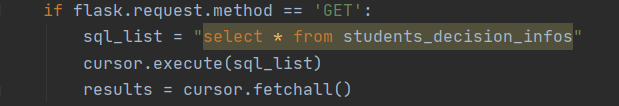


4.2 python flask框架下后端代码：

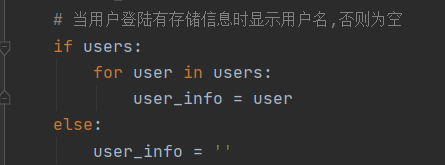
1. 增加会话保护机制(未登陆前login的session值为空)



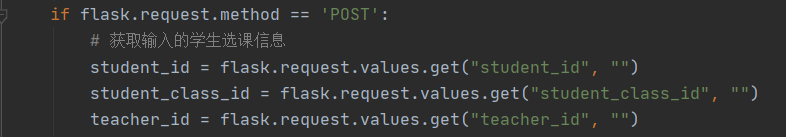
1. 获取用户信息



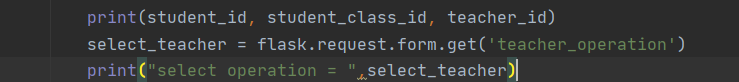
1. 记录当前登录的用户的用户名



1. 获取从网页输入的信息

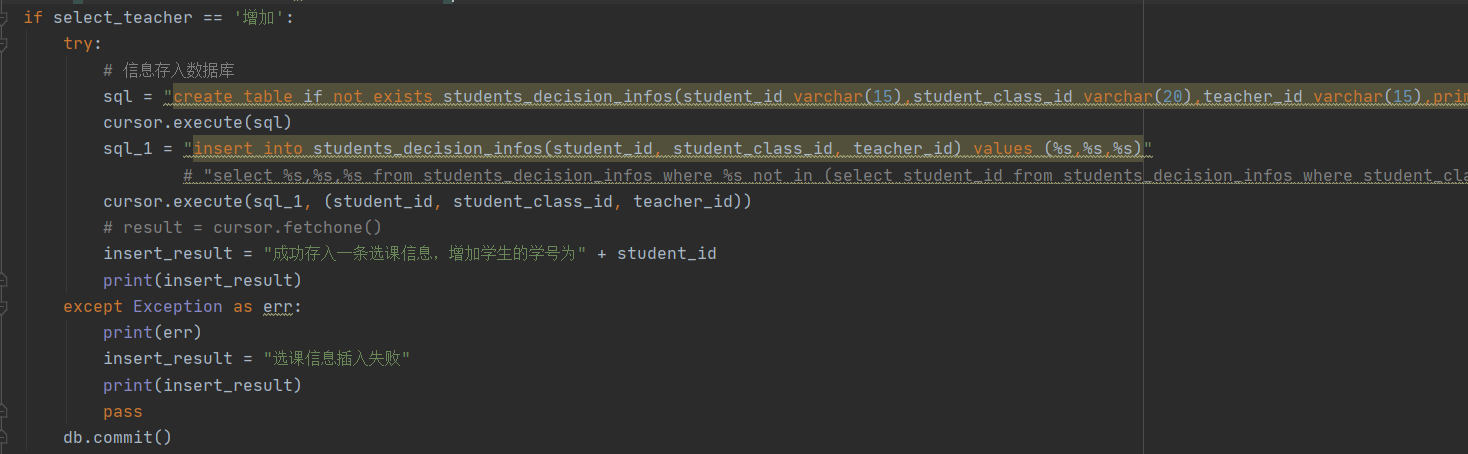


1. 从页面获得要进行的操作

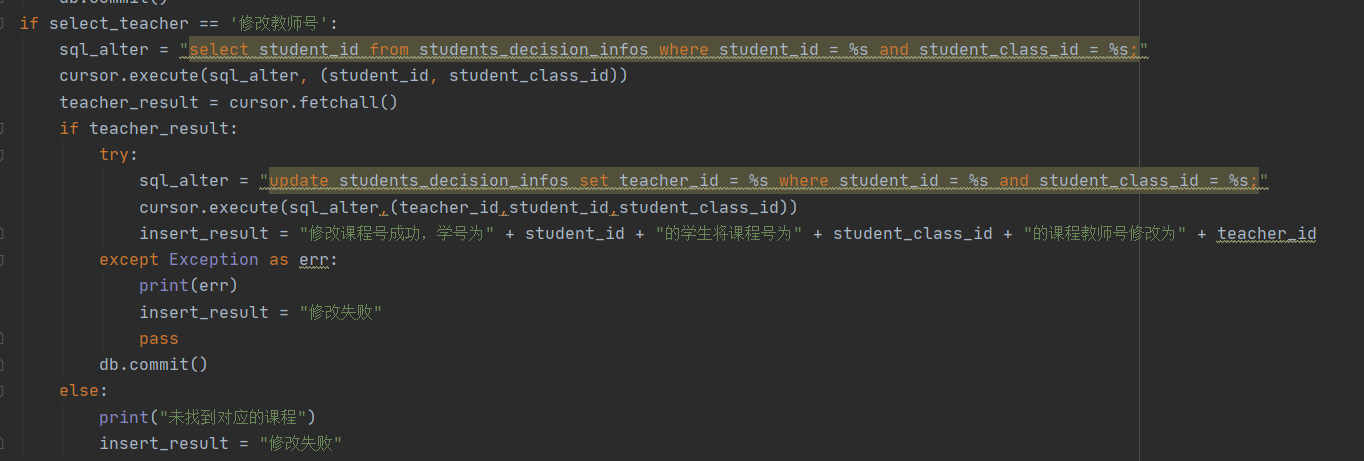


F．通过网页的下拉框，选择对应操作

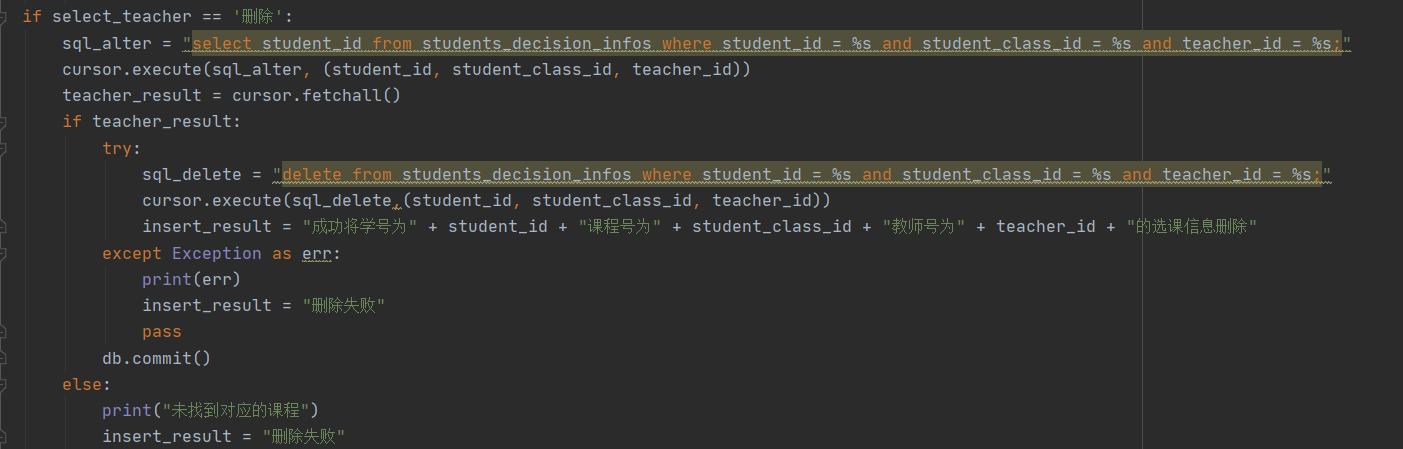
增加：



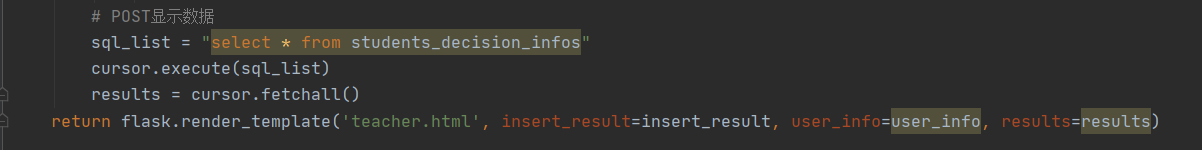
修改教师号：



删除：



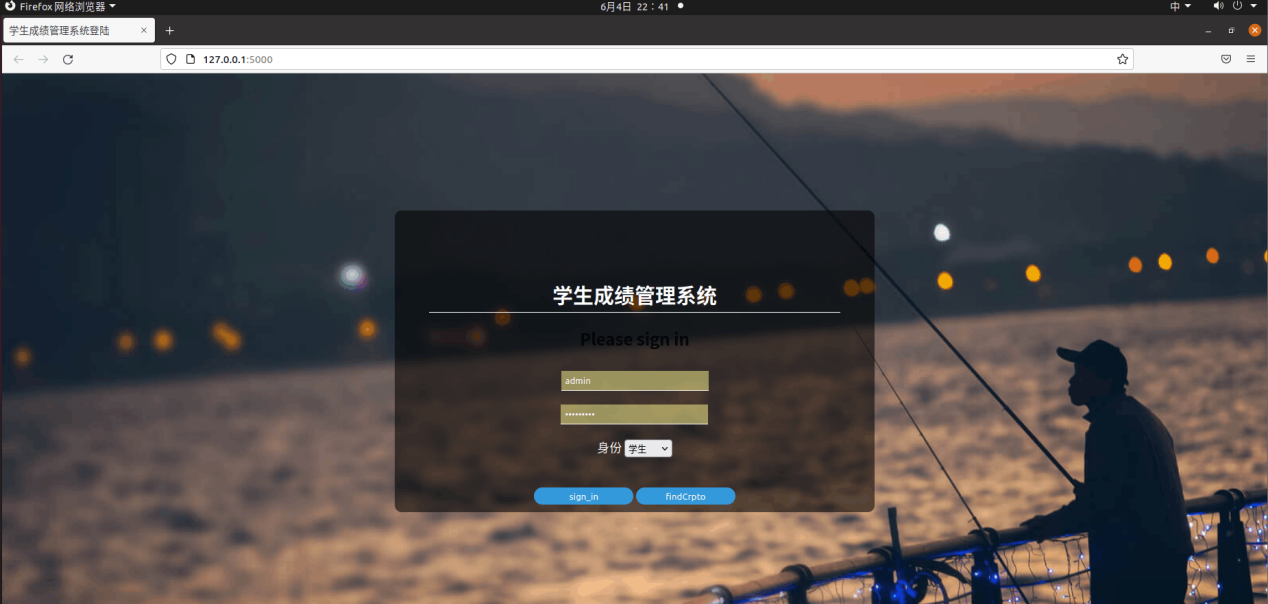
G．验证不通过则向网页返回错误信息



**【实验结果】**

1.登陆界面展示及效果

1.1 login.html界面展示：



1.2 输入密码登陆操作

1. 用户名admin，身份为管理员，密码为：123qweasd

登陆并跳转成功



1. 用户：chr，身份为学生

登陆并跳转成功

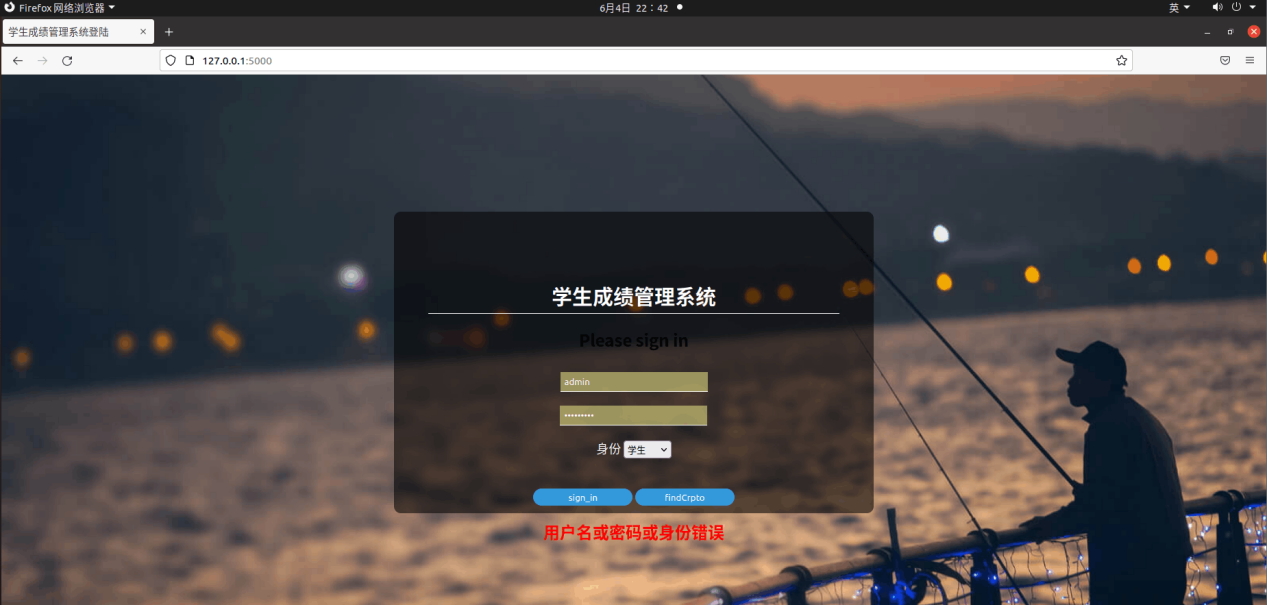


1. 用户：ym，身份为老师

登陆并跳转成功

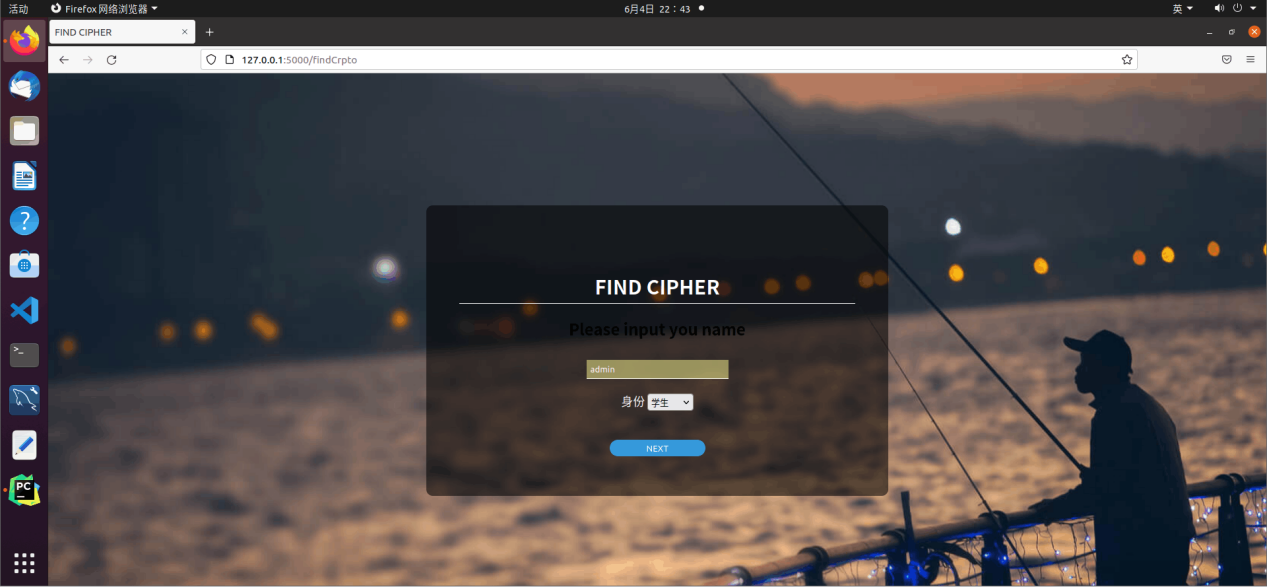


1. 登陆失败提示



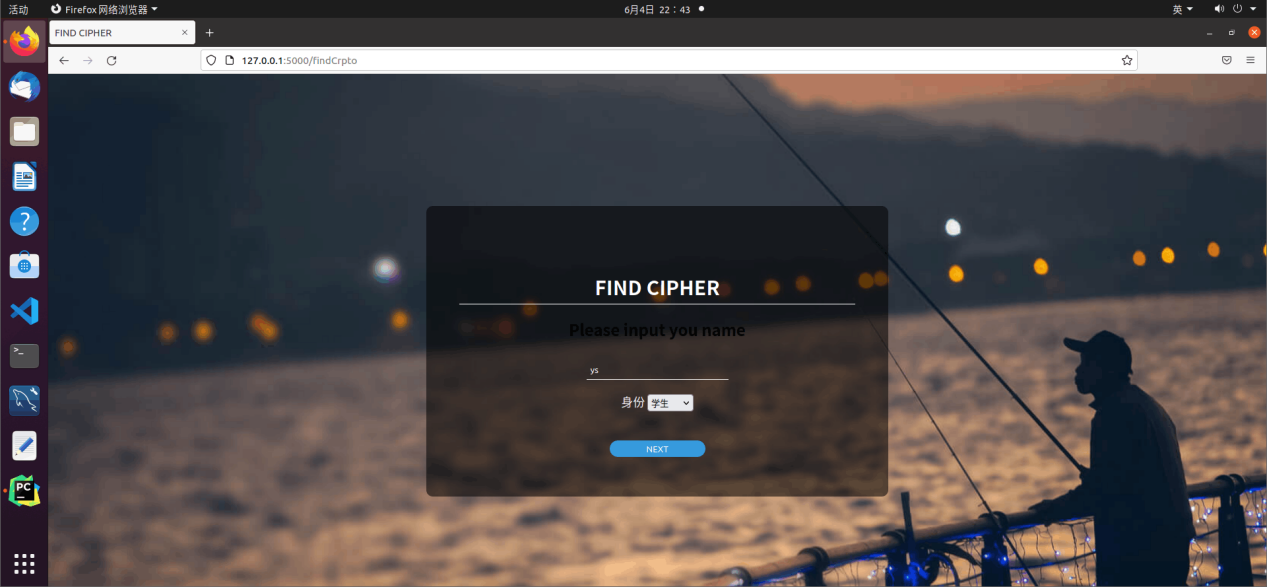
2.找回密码功能展示

2.1 findCrpto及resetcrpto界面展示：

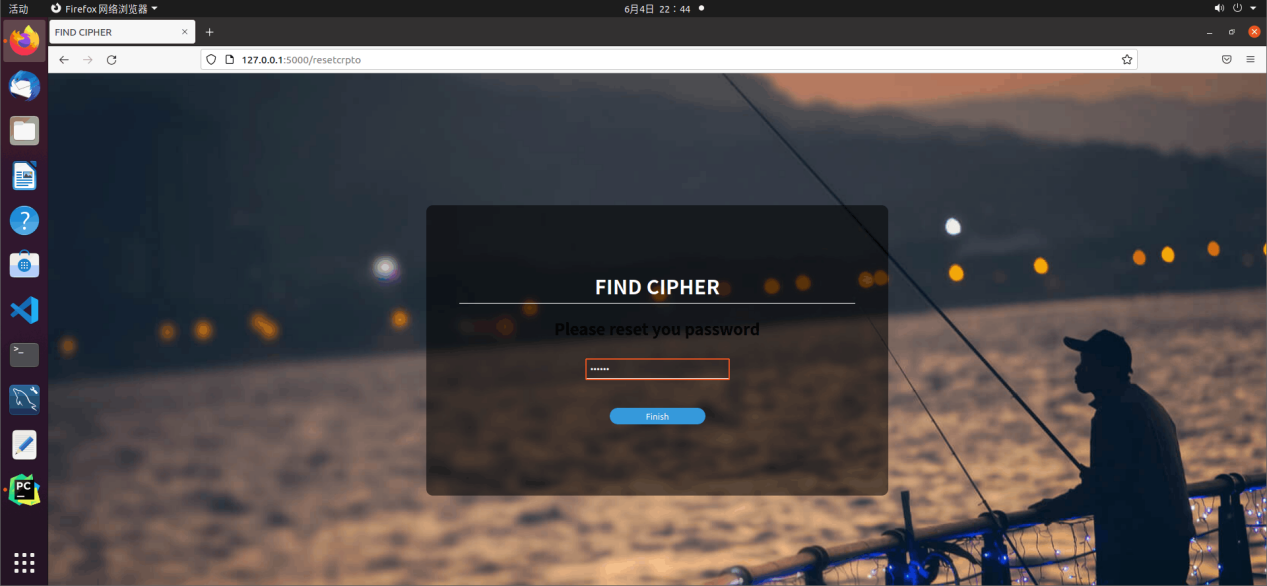


2.2 具体操作

A.对用户名进行查找，观察是否存在用户



1. 存在则跳转



1. 成功则跳转，再次登陆成功，并观察数据库表中是否已经：





3.管理员操作界面展示及效果

3.1 adminstator.html界面展示：



3.2 对用户进行操作

初始界面：



1. 增加test用户，权限为学生，密码为：123qwe

增加成功



1. 修改test用户密码，修改为：123qweasd

修改密码成功



1. 修改test用户权限，修改为教师

修改权限成功



1. 删除test用户

删除成功



4.数据库备份与恢复界面及效果展示

4.1 backup.html界面展示：

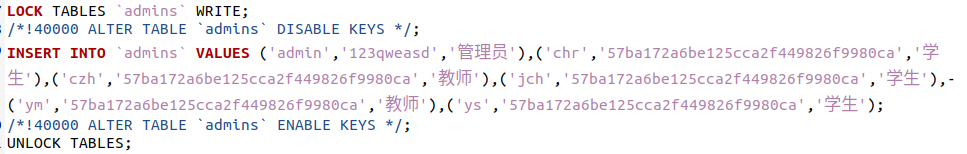


4.2 数据库备份与恢复操作

A.对此状态下数据库进行备份操作



1. 查看备份结果



1. 对数据库进行修改操作，增加test用户，密码为：123qweasd，权限为学生



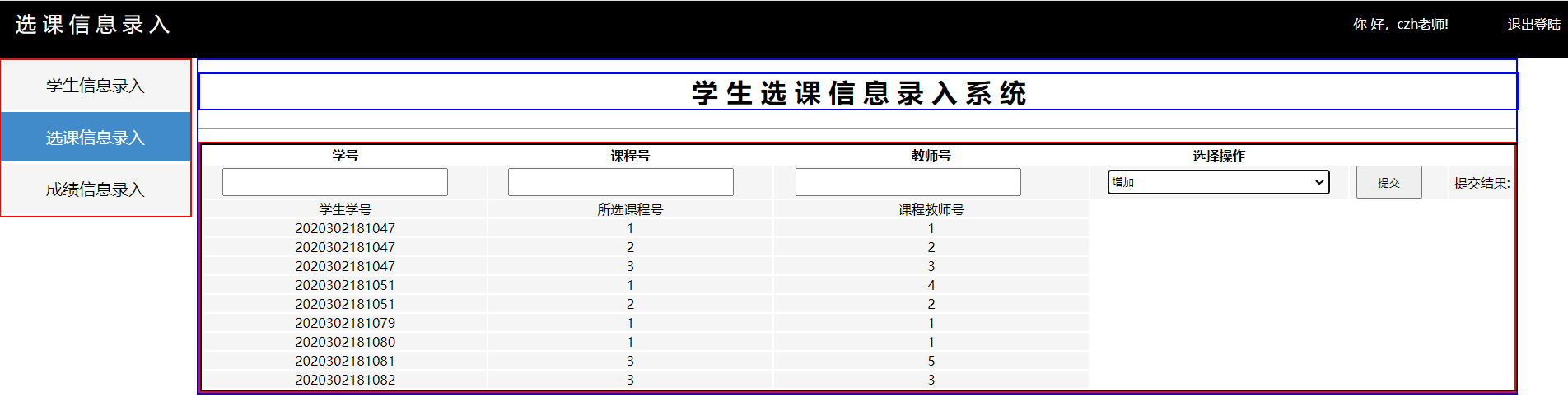
1. 进行数据库恢复

恢复成功，test用户消失



5.教师操作界面展示及效果

5.1.1 teacher.html界面展示：



5.1.2 对选课信息进行操作

初始界面：



1. 增加选课信息，学号为2020302181051，课程号为5，教师号为5

增加成功



1. 修改教师号，将学号为2020302181051的学生的课程号为5的课程教师号改为10

修改教师号成功



1. 删除选课信息，将学号为2020302181051、课程号为5、教师号为10的选课信息删除

删除成功



5.2.1 student.html界面展示：



5.2.2 对选课信息进行操作

初始界面：



1. 增加学生信息，学号为2020302181001，班级为20-1，姓名为张三

增加成功



1. 修改班级，将学号为2020302181001的学生班级修改为20-10

修改班级成功



C．修改姓名，将学号为2020302181001的学生姓名修改为赵四

修改姓名成功

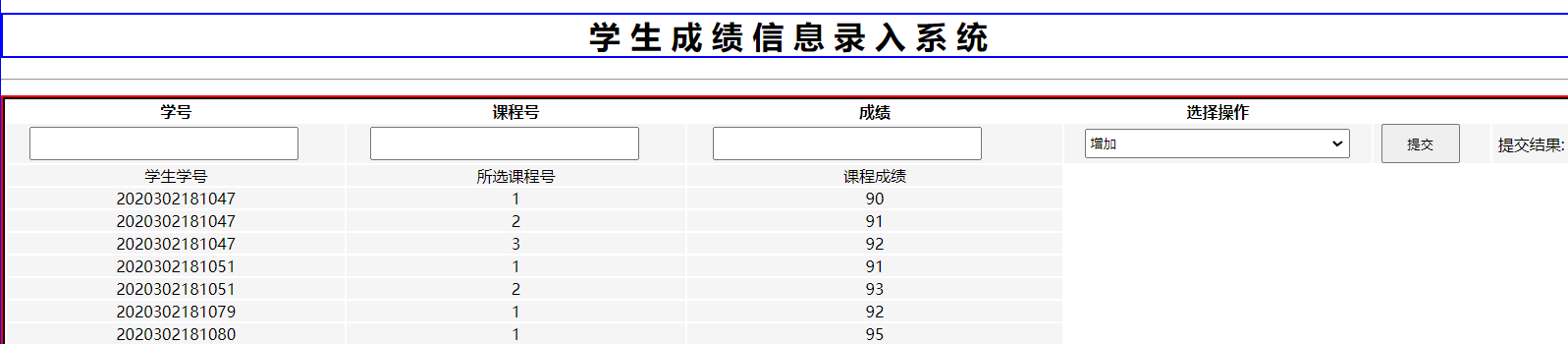


D．删除学生信息，将学号为2020302181001的学生信息删除

删除成功



5.3.1 grade.html界面展示：



5.3.2 对学生成绩进行操作

初始界面：



1. 增加学生成绩信息，将学号为2020302181081，课程号为3，成绩为99

增加成功



1. 修改课程号，将学号为2020302181081的学生的课程号由3改为4

修改课程号成功



C．修改成绩，将学号为2020302181081的学生的课程号为4的课程成绩改为100

修改成绩成功



D．删除学生成绩信息，将学号为2020302181081的学生的课程号为4的成绩信息删除

删除成功



**【实验小结】**

1. 在找回密码方面可以进行优化，比如说增加密保手机等等，一起存储到存储密码的数据库中。
2. 学生信息添加查找更改可优化，操作更加方便简化
3. 可增加cookie保护机制，保护用户隐私

|  |
| --- |
| **评语： 成绩：**  **签名：**  **日期： 年 月 日** |