**武汉大学国家网络安全学院**

**实验报告**

**课程名称 数据库**

**专业年级 信安21级**

**姓名学号 赵伯俣2021302181156**

**赵晨阳2021302181066**

**实验学期 2022-2023　 学年 第二 学期**

**课堂时数 32 课外时数**

**填写时间 2023 年 5 月 25 日**

# 实验4 数据库应用系统的设计与实现

一、实验目的

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。

二、实验内容

1.对系统进行需求分析，主要针对数据库进行需求分析，可以采用数据流图、数据字典等方式。

2.对系统和数据库进行设计，根据需求分析的有关内容，画出系统的E-R图，并据此设计对应的数据库表结构，然后对系统进行简单范式分析等。

3. 设计实现一个数据库应用系统，使用网页的前端形式，数据库、语言、平台、框架选择Linux+html+python+Mysql。有如下要求：

1）用户登录，至少两种不同类型的用户权限访问控制；

2）用户口令哈希存储；

3）体现数据库完整性检查；

4）体现防SQL注入；

5）进行系统并发测试；

6）具有数据备份与恢复功能。

三、实验内容

1.设计数据库

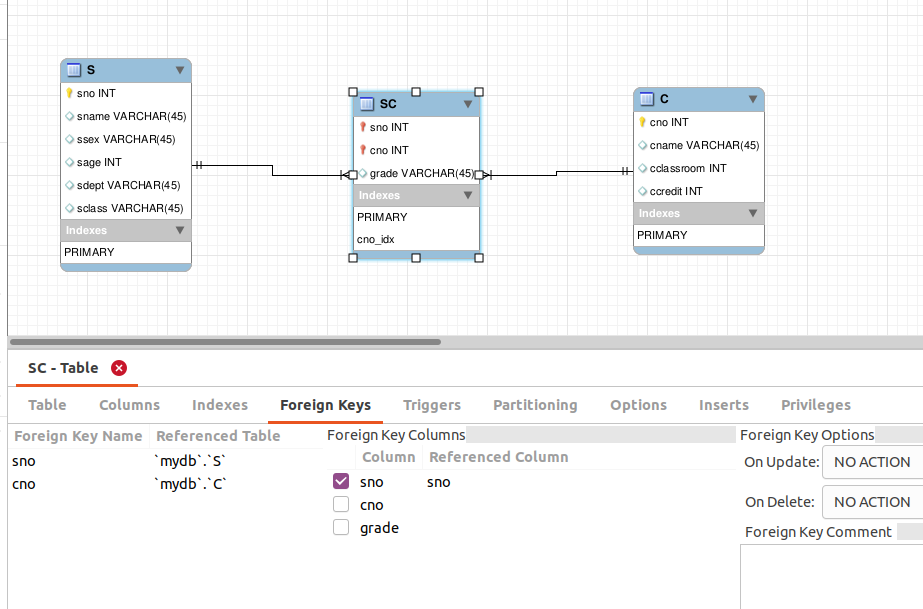
并定义相关的约束，绘制出对应ER图导入到数据库中,设计一个学生信息表，其中含有三个表分别为

S（sno,sname,ssex,sage,sdept,sclass）主键为sno

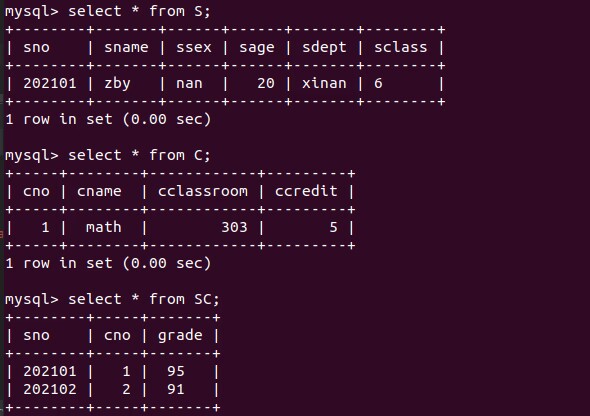
C（cno,cname,ccredit,cclassroom）主键为cno

SC(sno,cno,grade)主键为(sno,cno)，外键为sno和cno

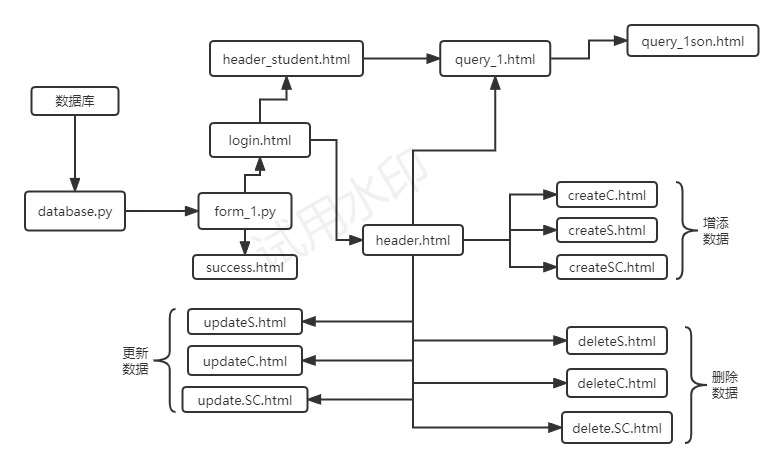
表结构的ER图如下图所示



导入数据库之后并插入相应的数据，查看数据库结果如下图所示。



2.整体设计。



（1）所有文件设计的整体框架图如上图所示，其中database文件的作用是链接数据库，将数据库与form\_1形成链接，在运行整体文件后会出现登陆界面，对应后端中的login函数，登陆界面有两种可选项即学生身份和教师身份，选择过程在后端中的home函数中进行实现，学生身份登录之后会跳转到header\_student中，学生身份只拥有查询某一个学号的学生的选课信息的权限，教师身份登录之后会跳转到header中。

（2）在header中有不同功能所对应的按钮，增加学生信息按钮跳转到createS页面；删除学生信息按钮跳转到deleteS页面；修改学生信息按钮跳转到 updateS页面；查询学生信息跳转到query\_1页面。其他表的按钮与学生表按钮相同。

（3）createS页面。在该页面实现的功能是输入学生的所有信息点击提交按钮之后将对应的信息插入到数据库中，对应后端中的insert\_S函数。

（4）deleteS页面。在该页面实现的功能是输入想要删除的学生的学号，后端会将数据库中对应学号的学生信息进行删除，对应后端中的delete\_S函数。

（5）updateS页面。在该页面实现的功能是输入想要修改的学生的全部信息，后端会按照提供的信息对数据库进行修改，对应后端中的update\_S函数。

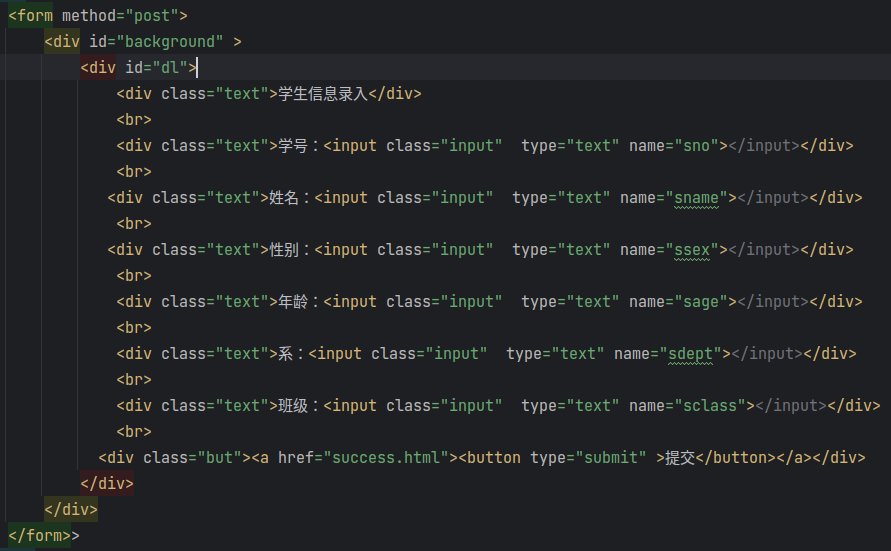
（6）query\_1页面。在该页面实现的功能是输入想要查询的学生的学号，跳转到query\_1son页面展示该学生的选课信息，对应后端中的select\_table函数。

3.设计应用系统后端python核心代码

（1）database



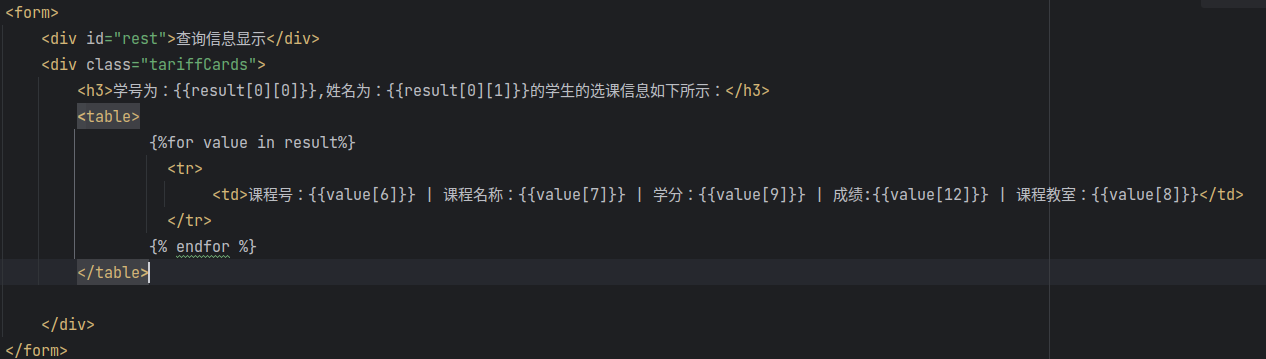
（2）form\_1

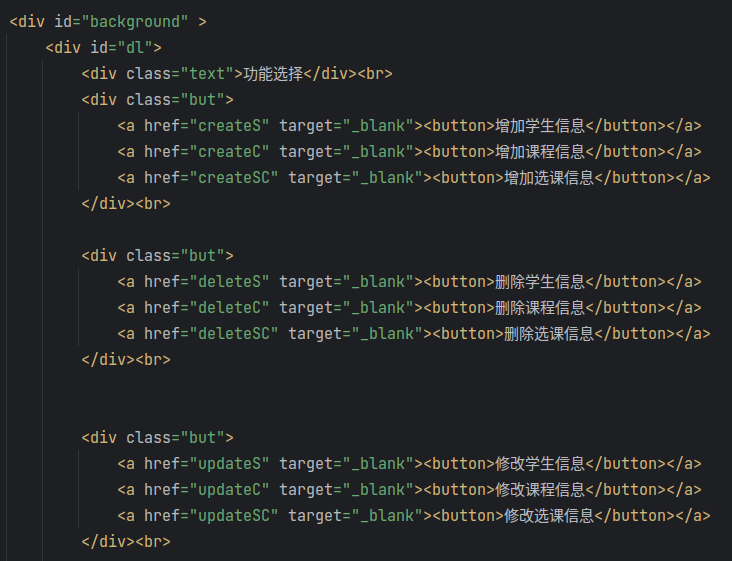


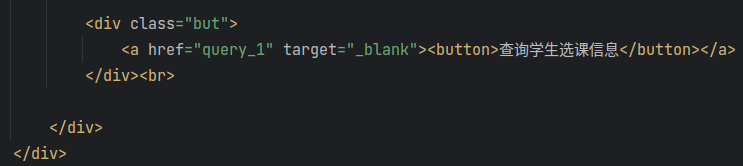


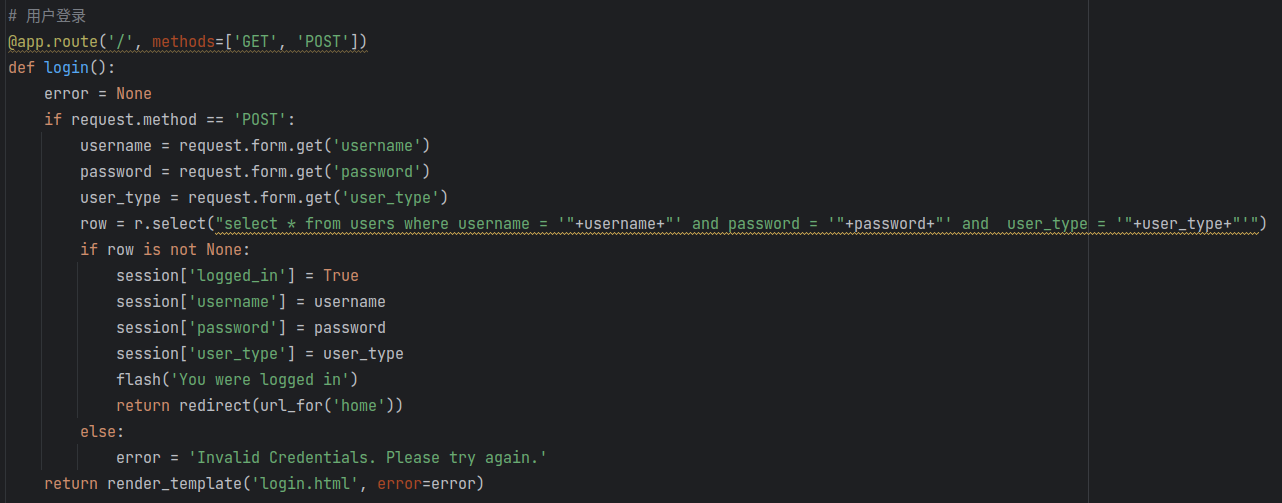




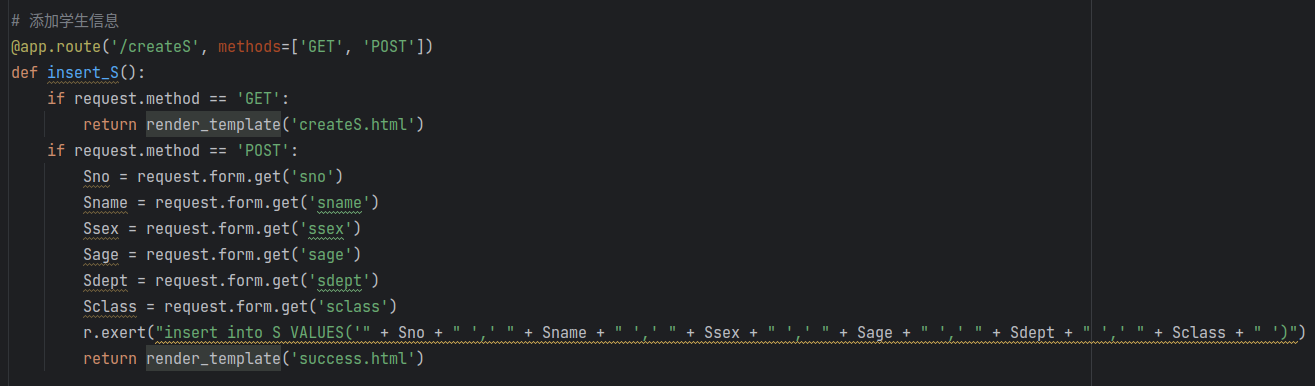




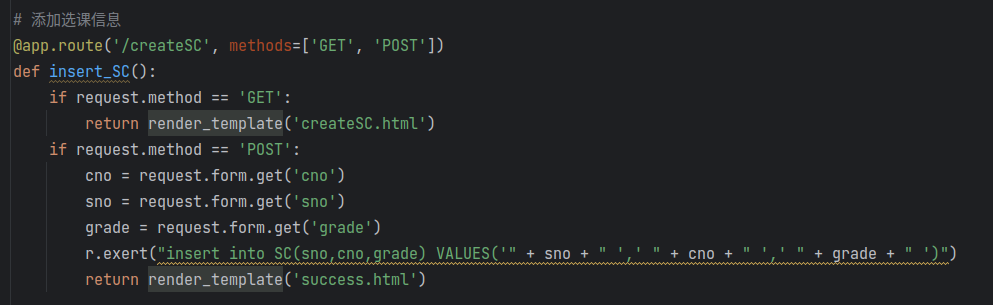


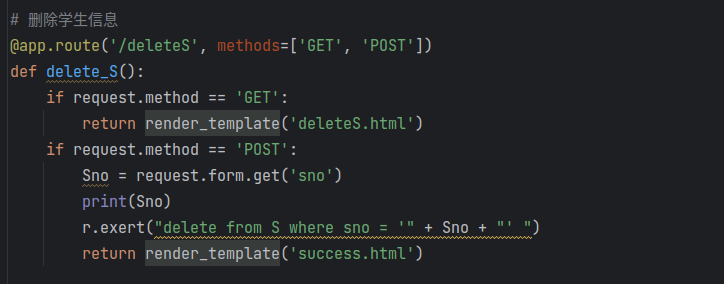






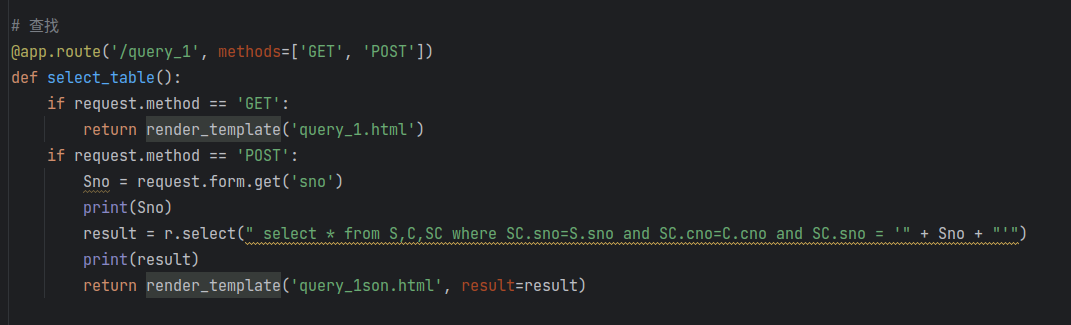


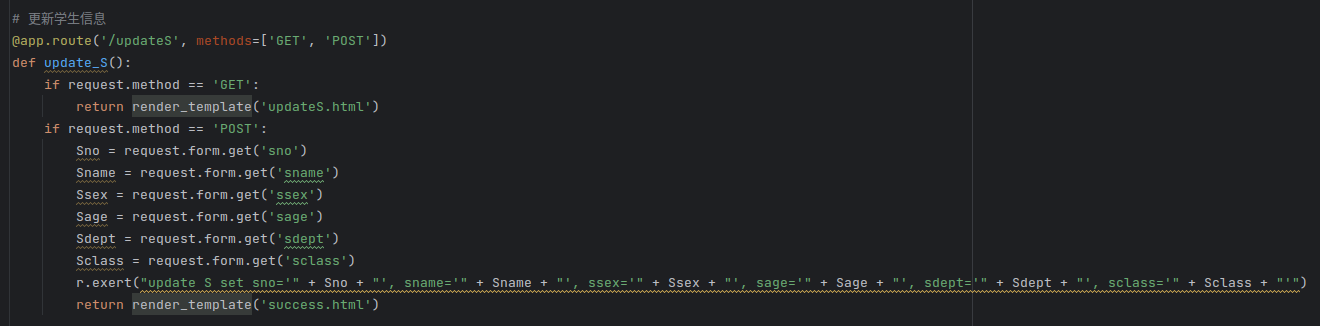


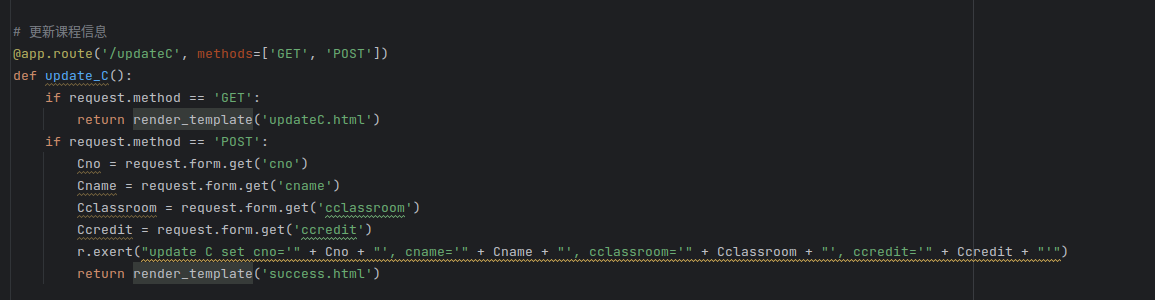


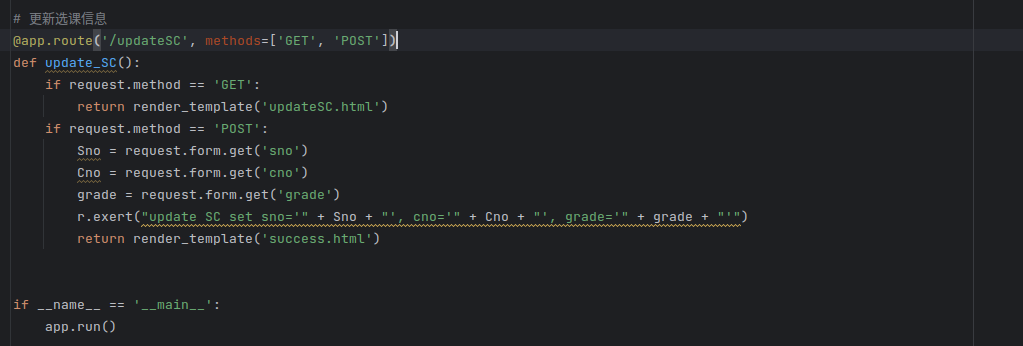






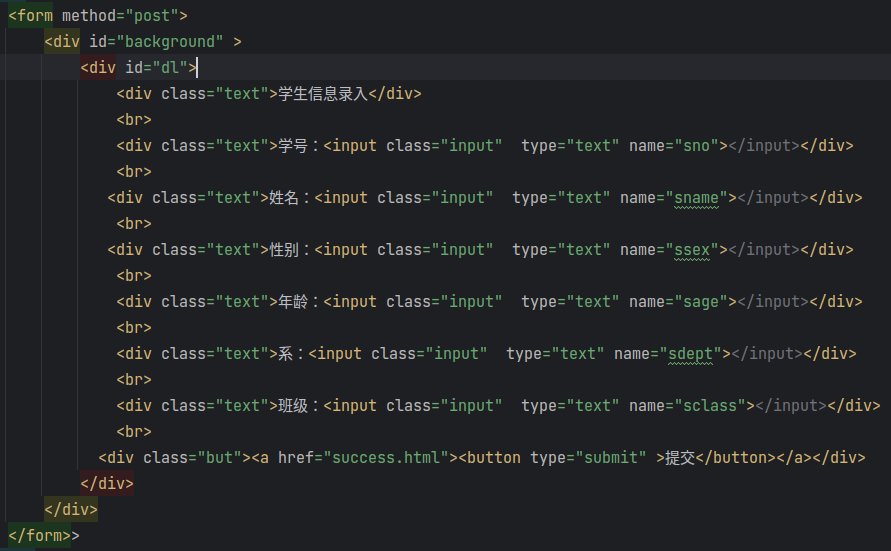




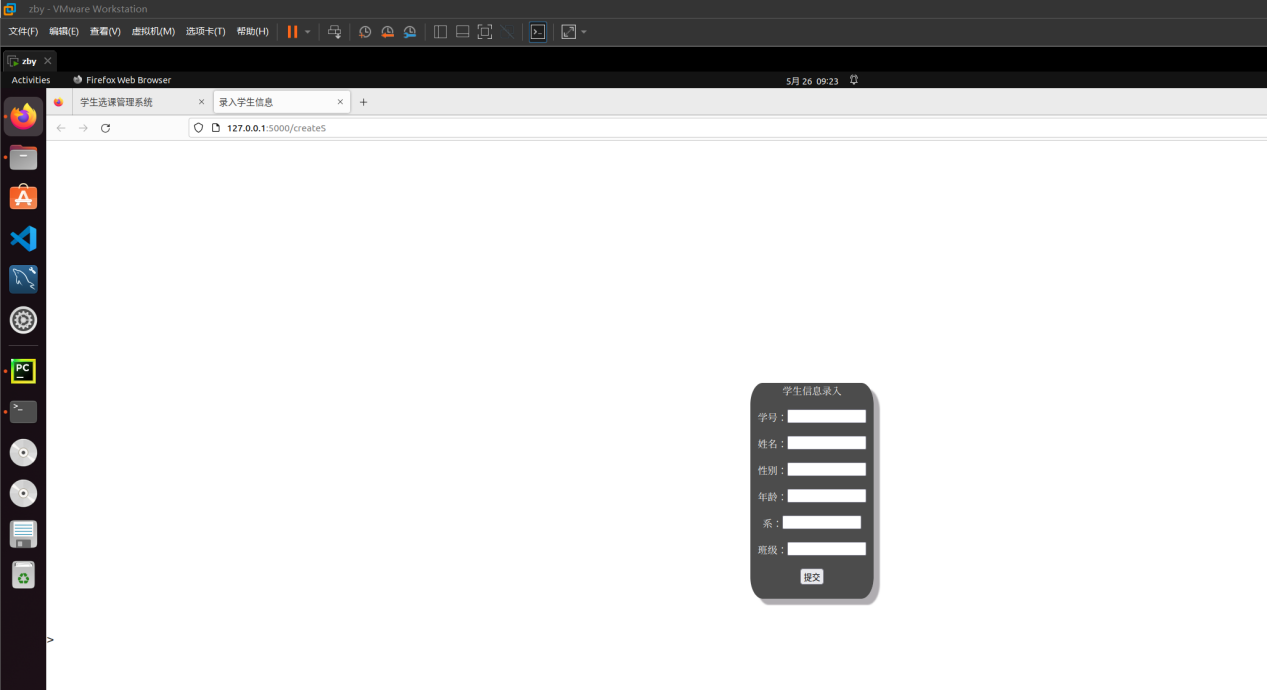


4.设计应用系统前端html核心代码

(1)createS。实现增添学生信息的功能，用户需要在对应的输入框中输入学生的信息点击提交按钮后便可将学生信息插入到数据库中。



运行后效果如下图所示

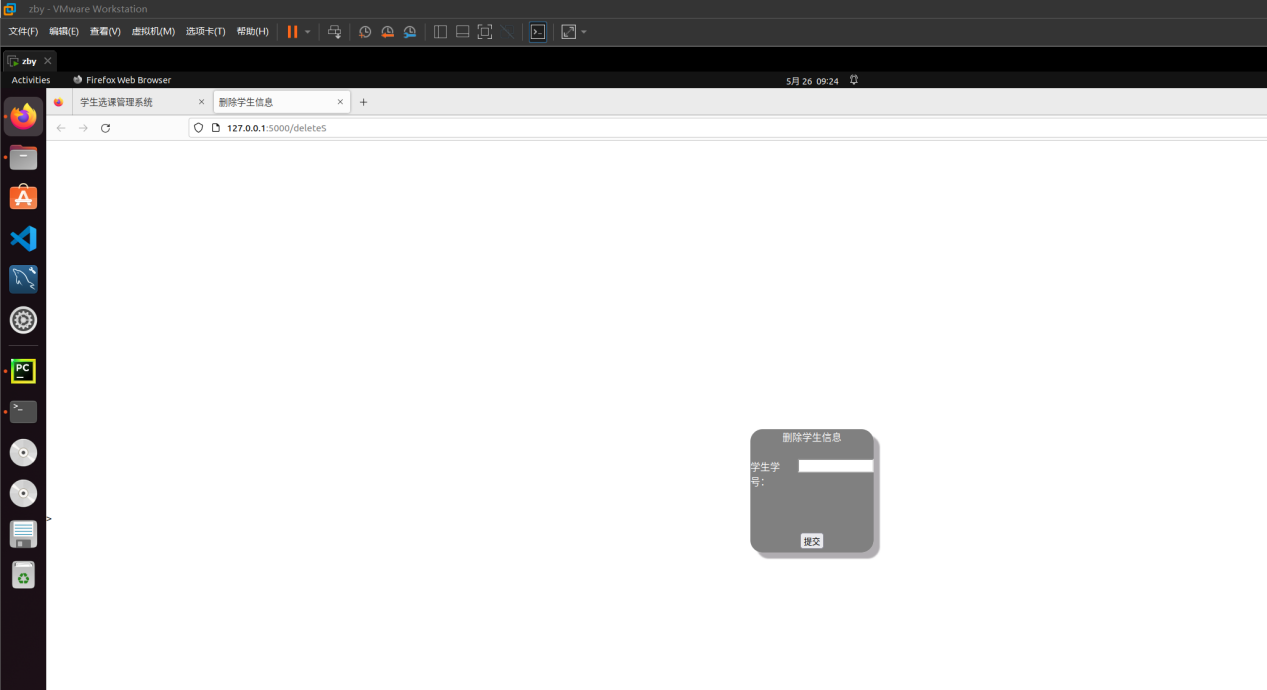


(2)deleteS

删除学生信息页面实现的功能是使用者输入想要删除的学生的学号，实现将数据库中对应学号的学生信息删除的操作。



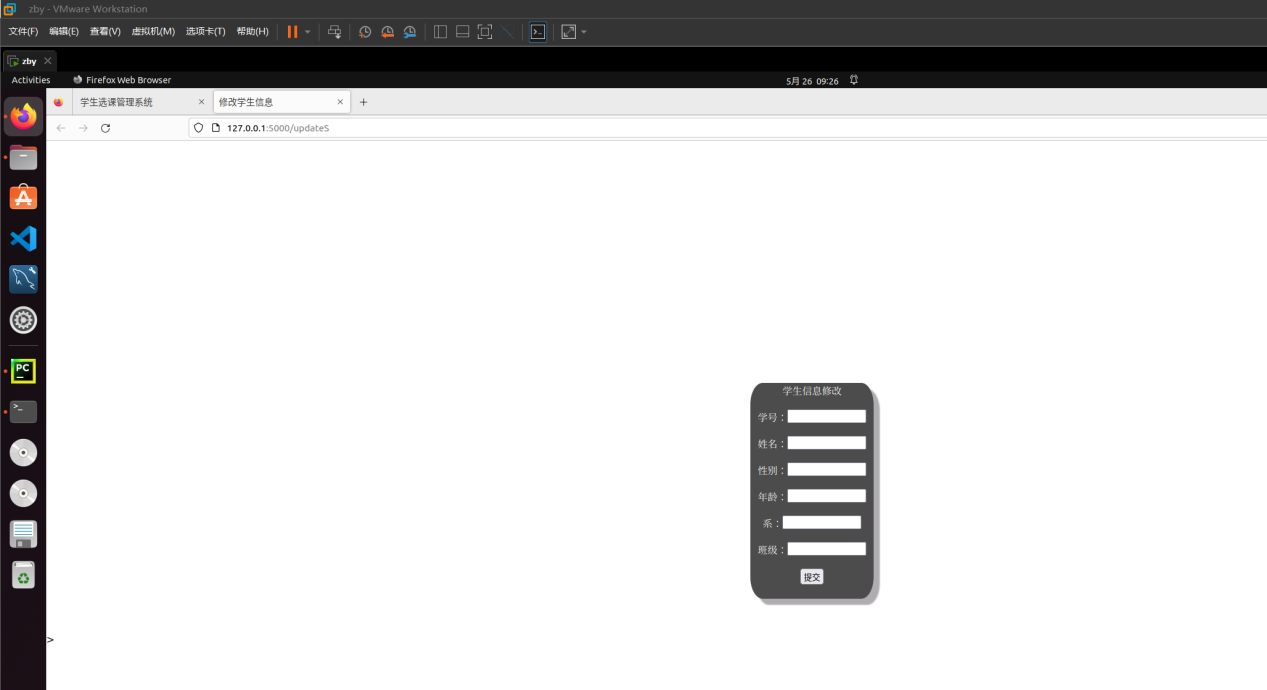
运行后效果如下图所示



(3)updateS  
更新学生信息界面实现的功能是使用者输入学生的信息后对对应学号的学生的信息进行修改



运行后的效果如下图所示

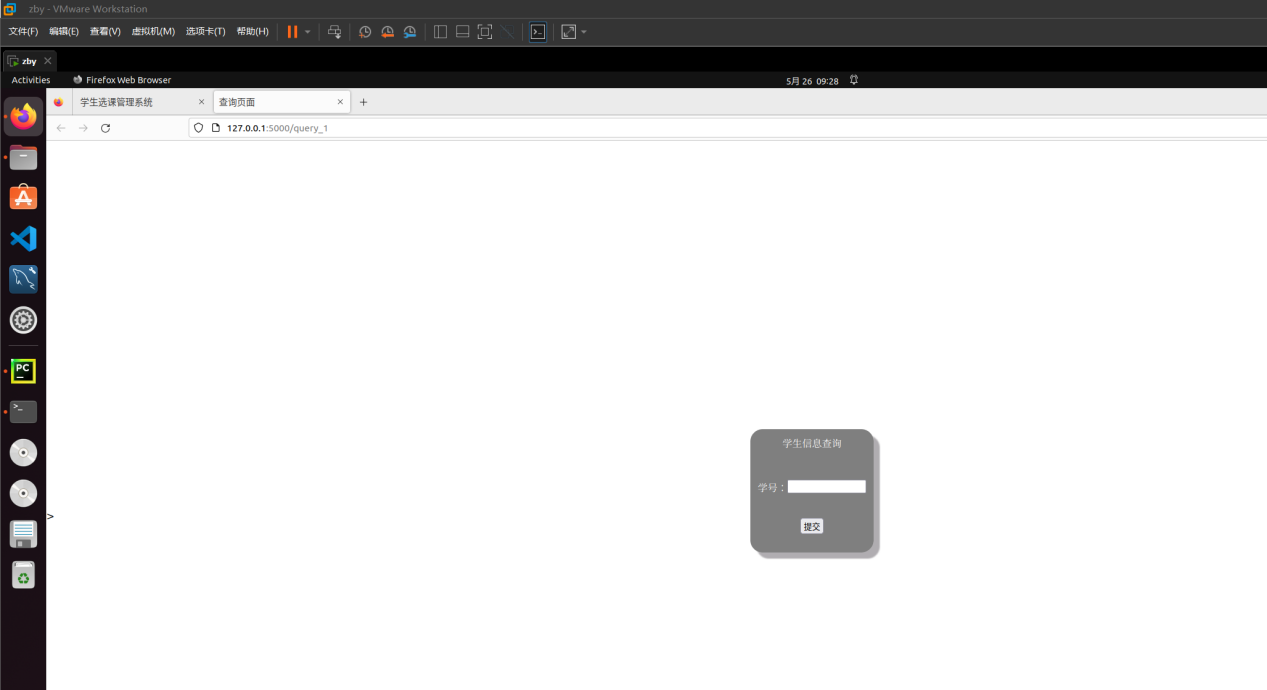


(4)query\_1

查询学生信息页面实现的功能是输入想要查询的学生的学号，从数据库中查询对应学号的学生的所有信息显示在query\_1页面中

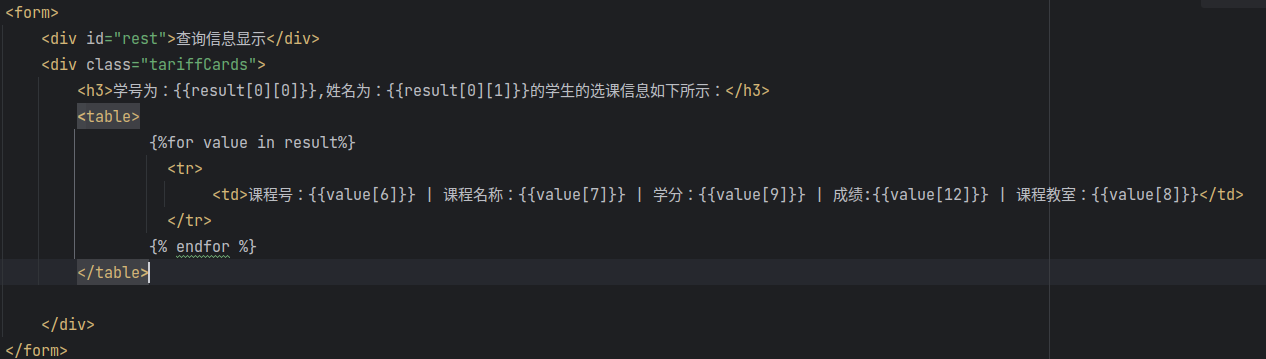


运行效果如下图所示

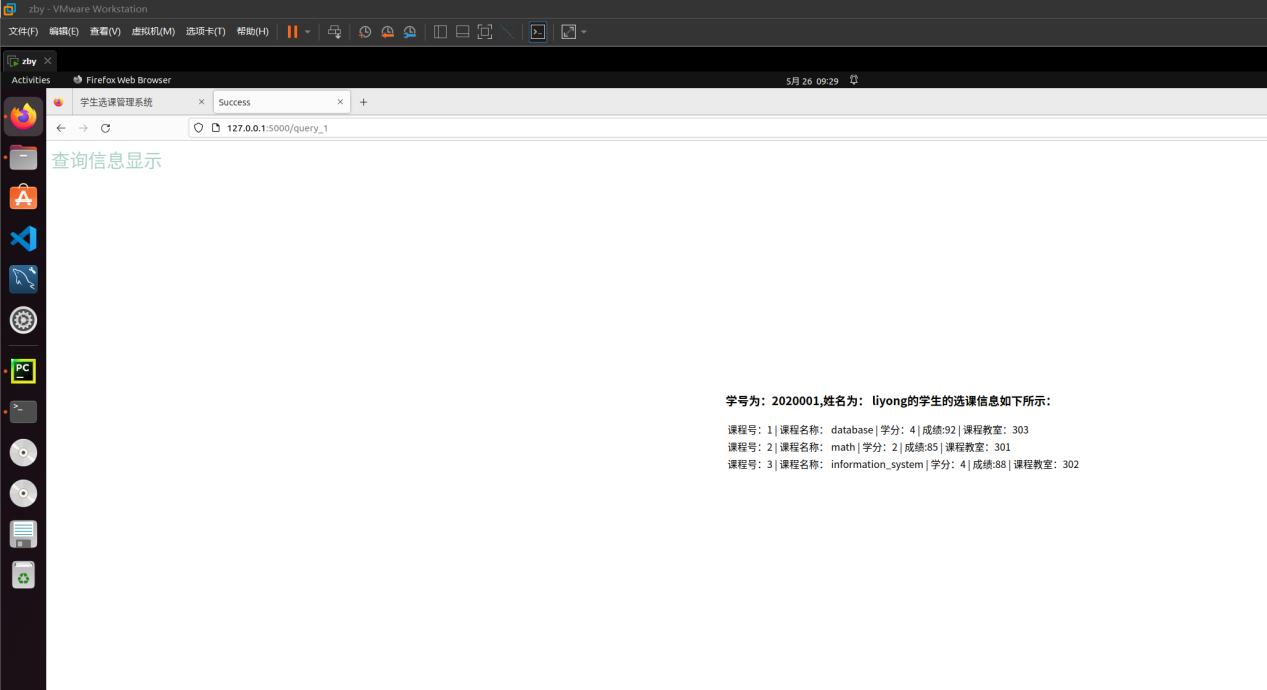


(5)query\_1son

实现的功能是接受来自后端的输入result数组查询结果，将查询结果显示有需要的数据，在该页面中显示学生的课程号，课程名称，学分，成绩，课程教室的信息。

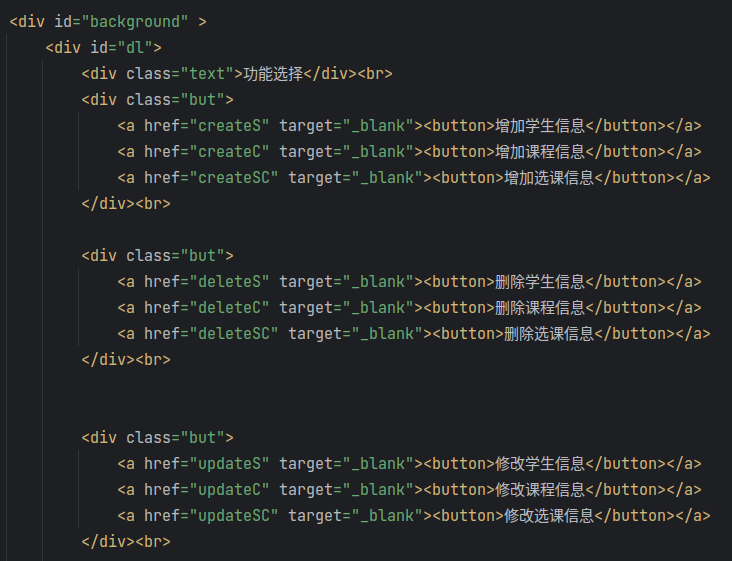


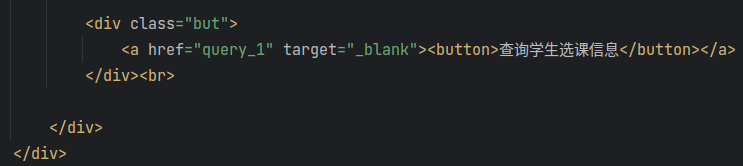
运行效果如下图所示



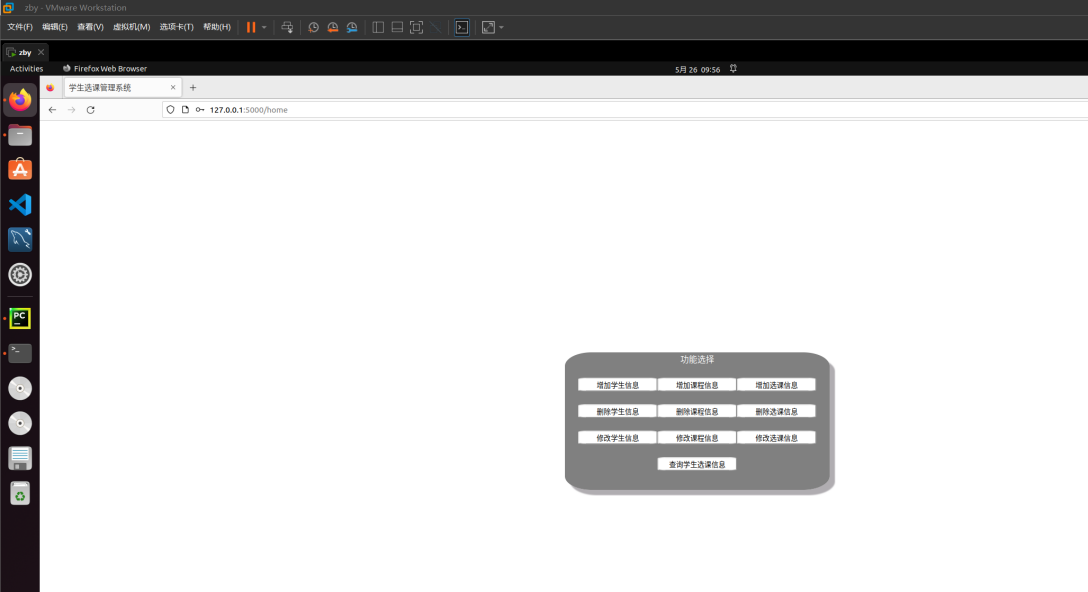
(6)header

该页面起一个功能选择菜单的作用，根据用户点击的不同的按钮跳转到不同的界面并执行相关的操作。





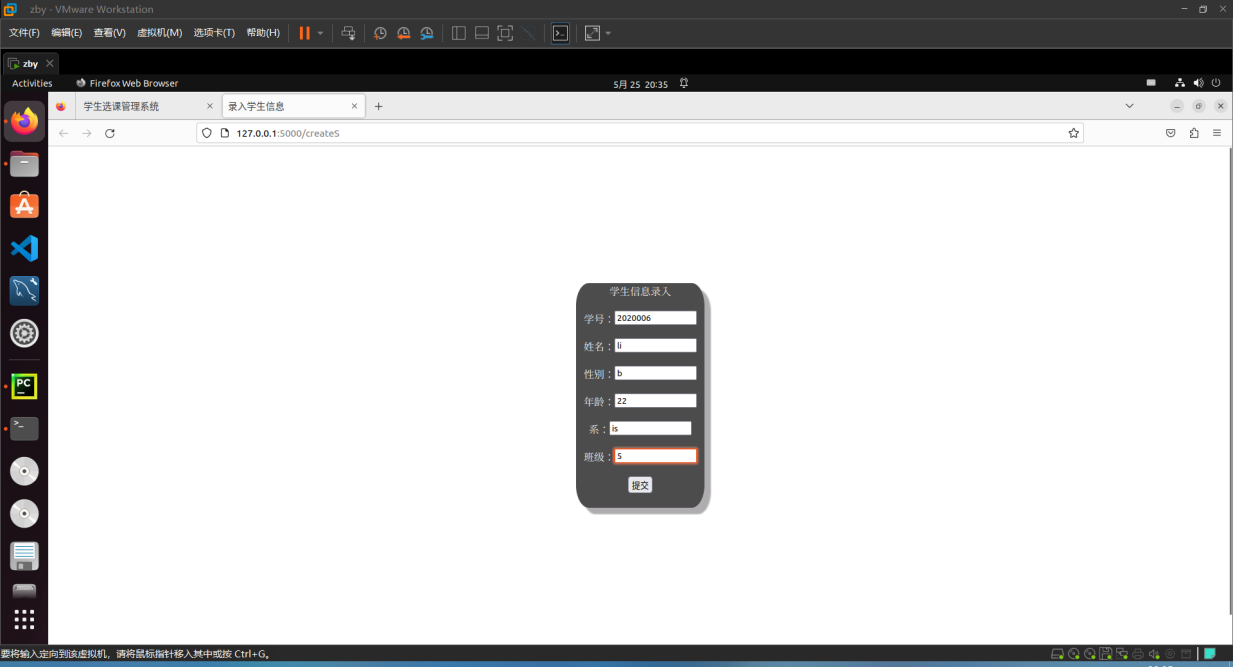
运行效果如下图所示

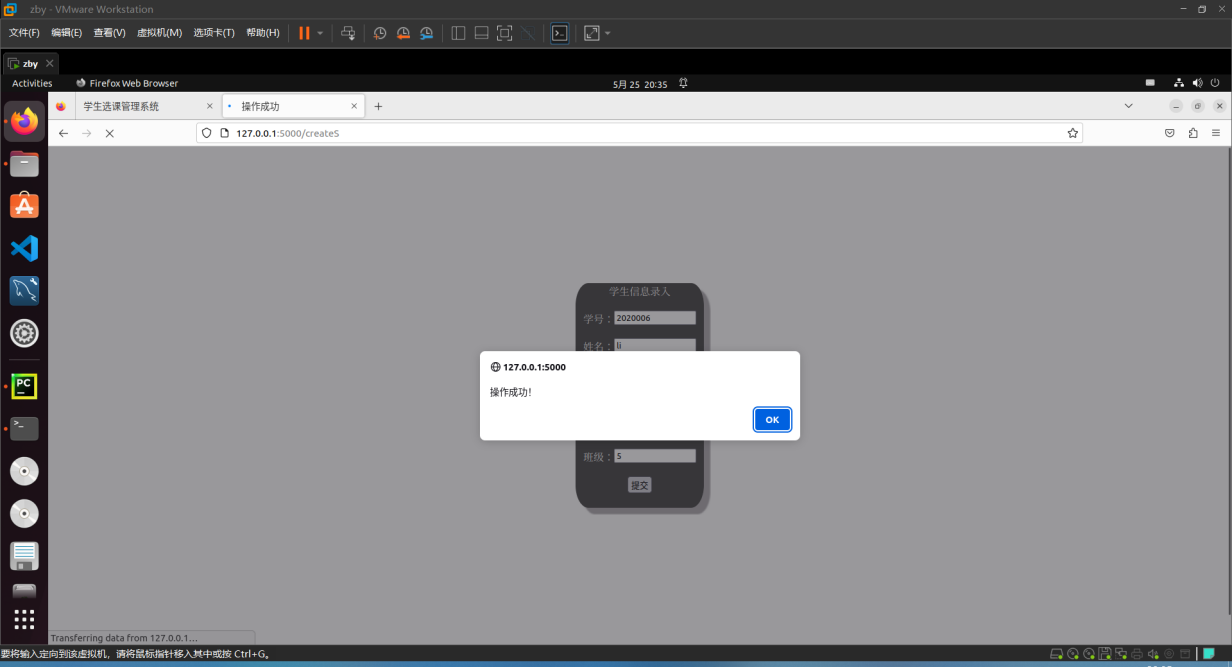


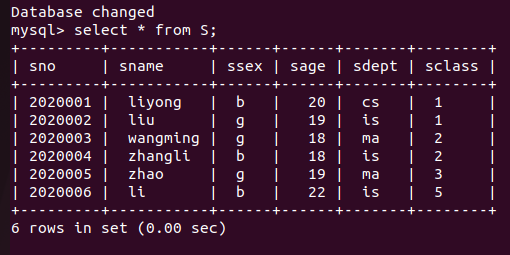
四、实验结果展示

1.实现学生信息的插入功能

下面实现将学生数据（‘2020006’，‘li’，‘b’，‘22’，‘is’,‘5’）数据通过前端页面插入到数据库中。

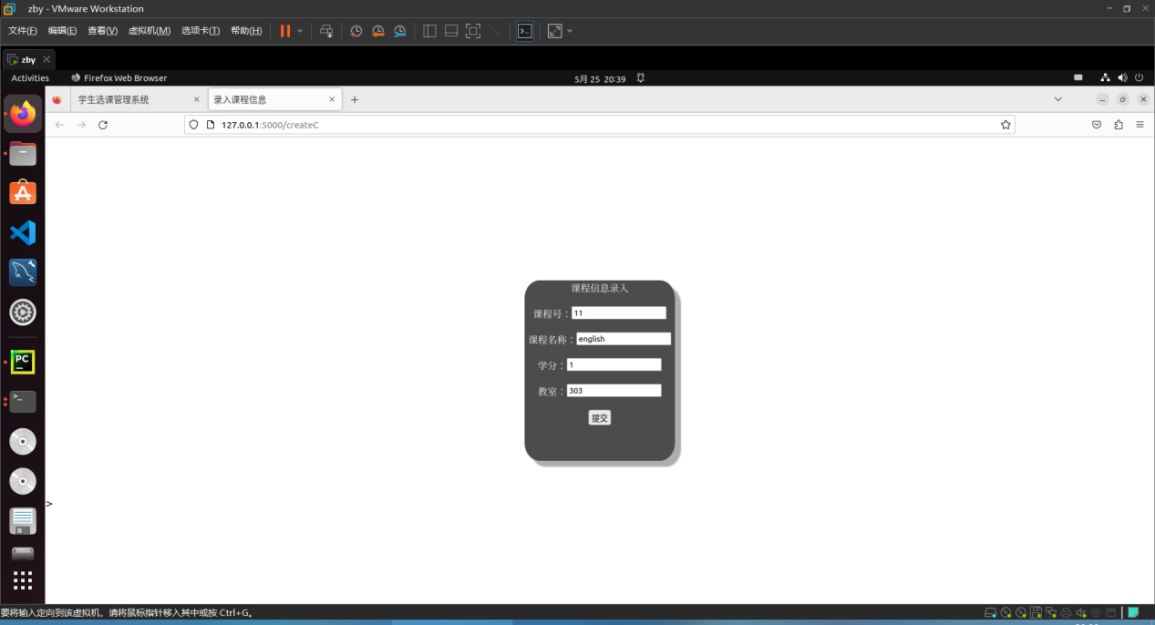


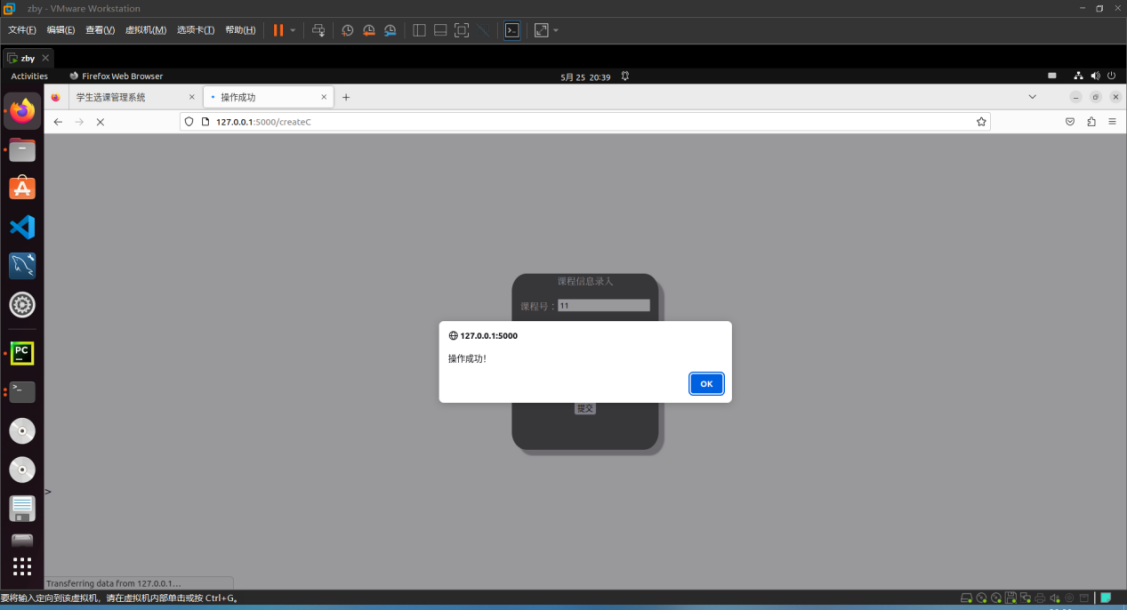


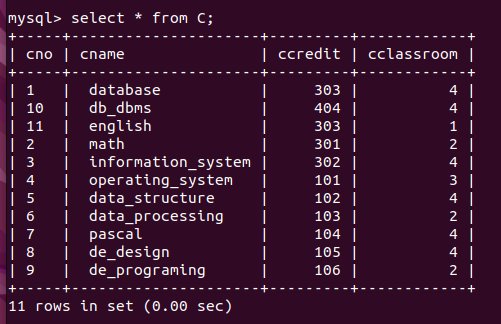


2.实现课程信息的插入功能

将课程信息(‘11’,‘english’,‘1’，‘303’)插入到数据库中



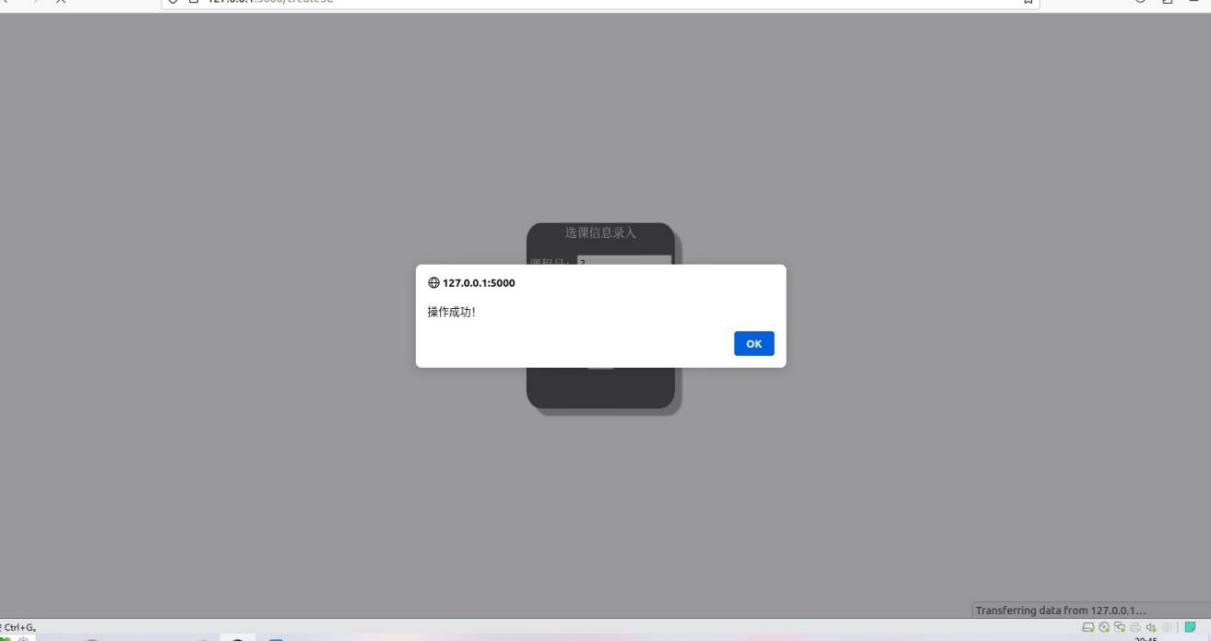


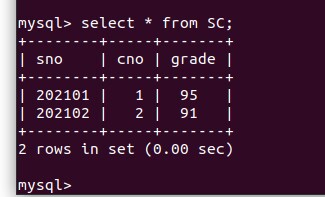


3.实现选课信息的插入功能

将选课信息（‘202102’，‘2’，‘91’）插入到SC选课表中。







4.实现学生信息的删除功能

用户输入想要删除学生信息的学生学号后将对应学生的信息从数据库中删除，删除成功后会弹出成功界面





5.实现课程信息的删除功能

用户输入想要删除课程信息的课程号后将对应课程的信息从数据库中删除，删除成功后会弹出成功界面

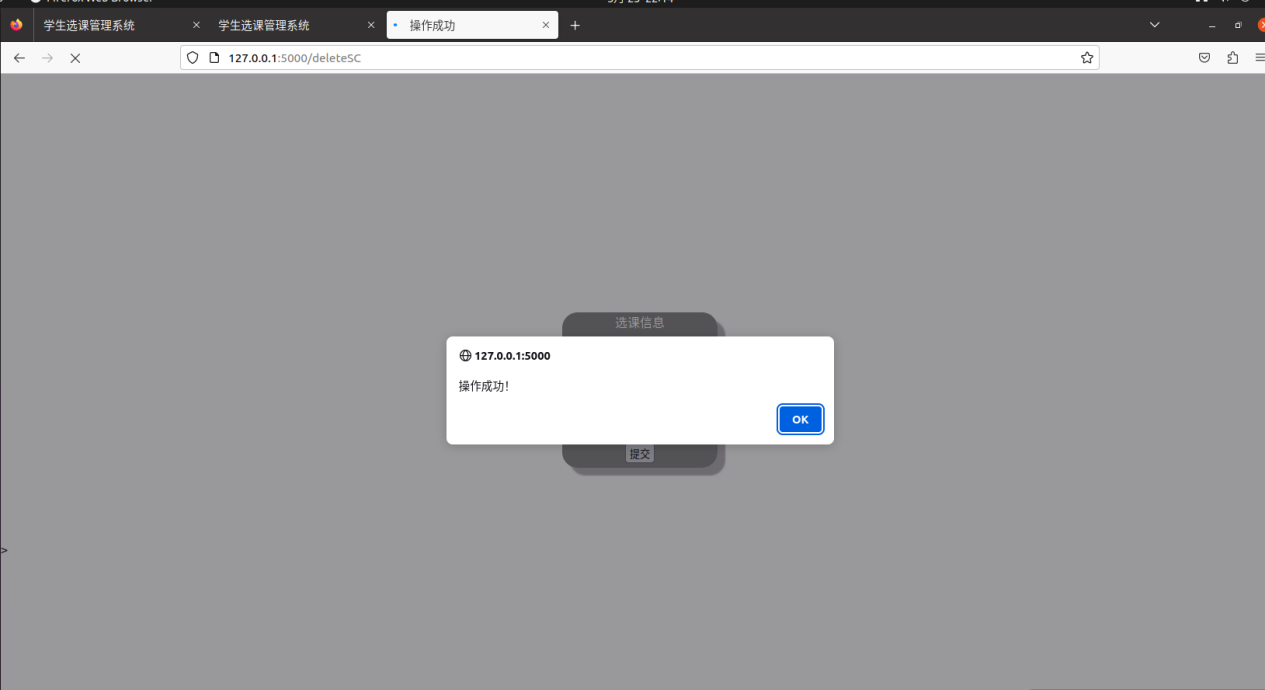




6.实现选课信息的删除功能

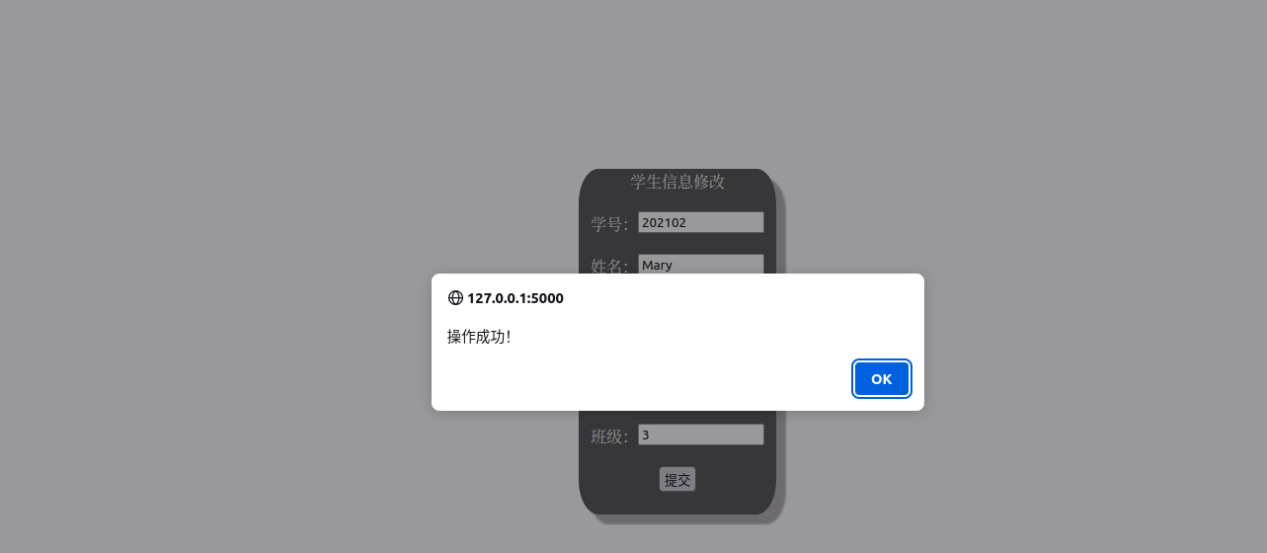
用户输入想要删除的选课信息的学号和课程号之后，将对应的选课信息从数据库中进行删除，删除成功后弹出成功删除界面。



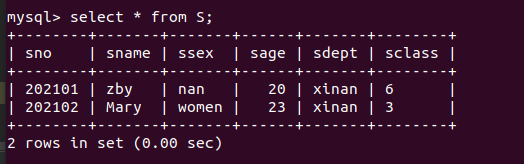


7.实现学生信息的修改功能

用户输入想要修改的学生的信息，在数据库中将对应的学生信息进行修改



修改后的数据库如下图所示

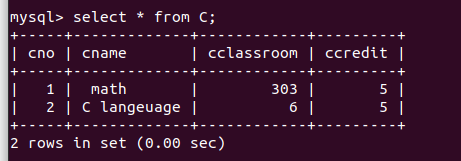


8.实现课程信息的修改功能

用户输入想要修改的课程信息后在数据库中将相应的课程进行修改。



修改后的数据库如下图所示

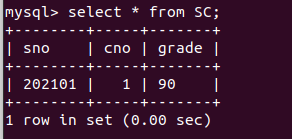


9.实现选课信息的修改功能

用户输入想要修改的选课信息后将数据库对应位置的信息进行修改

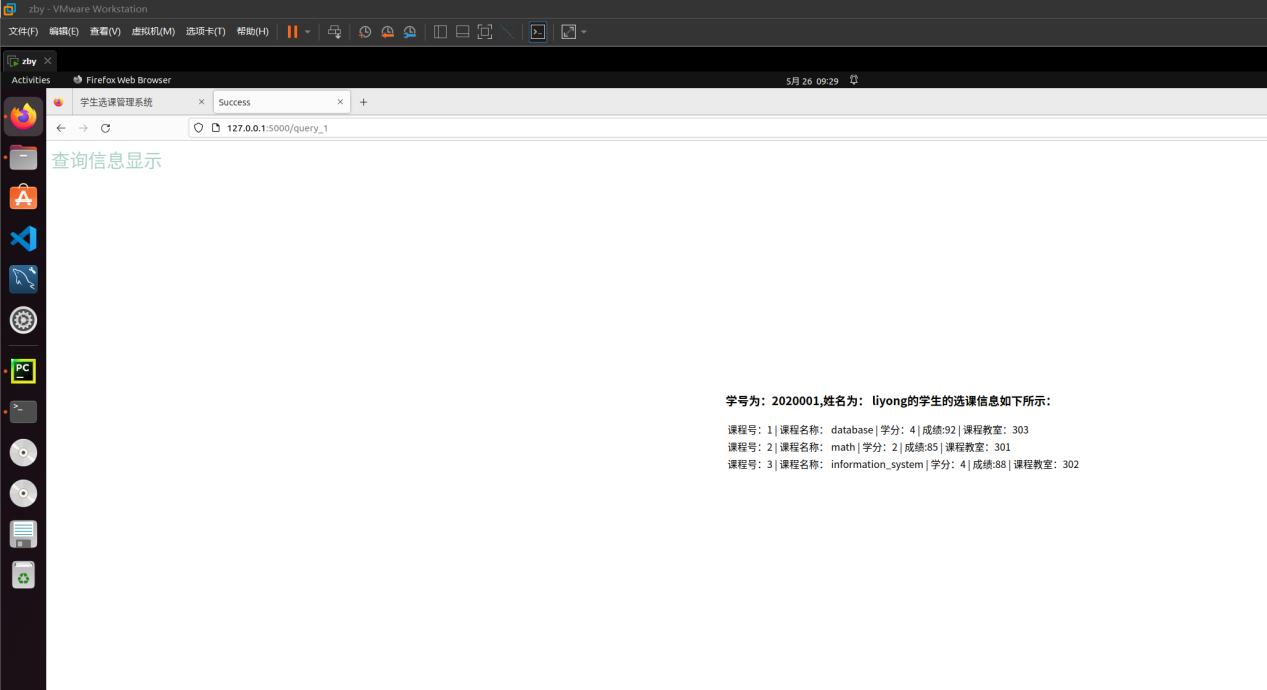


修改完成后的数据库如下图所示



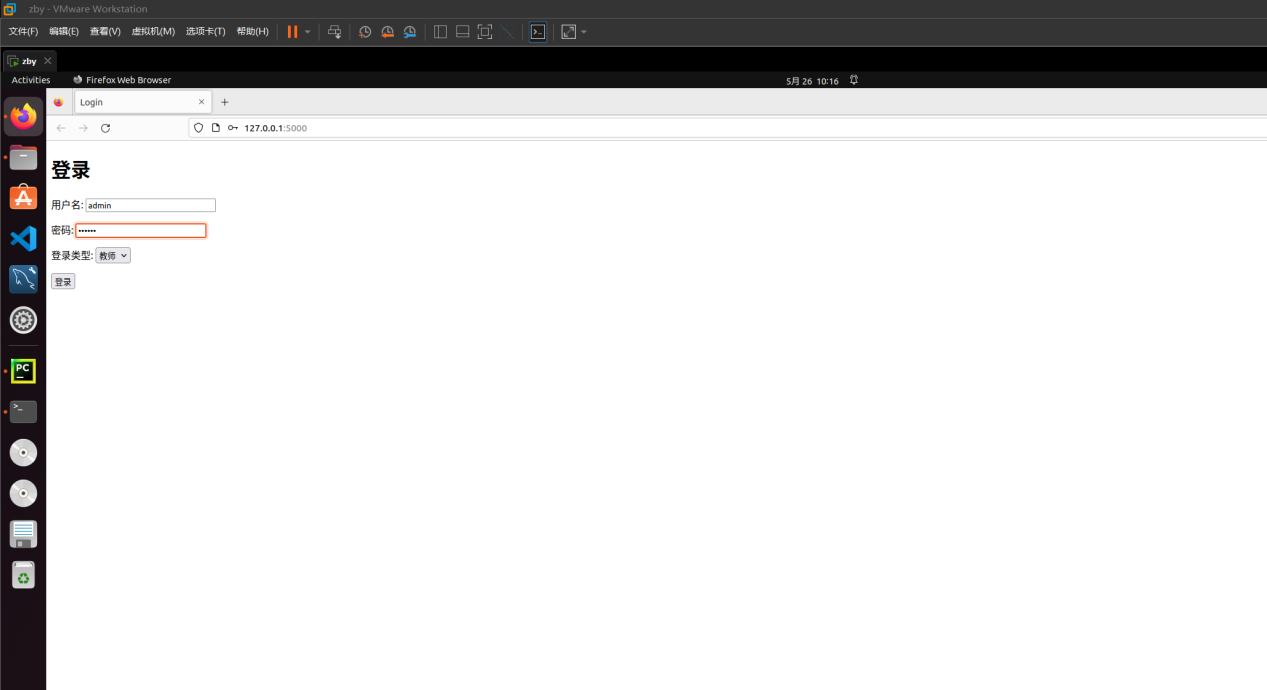
10.实现对应学号的学生信息查询功能

用户输入想要查询的学生的学号后将显示出该学生的所有选课信息，包括课程号，课程名称，学分，成绩和课程教室，执行结果如下图所示。

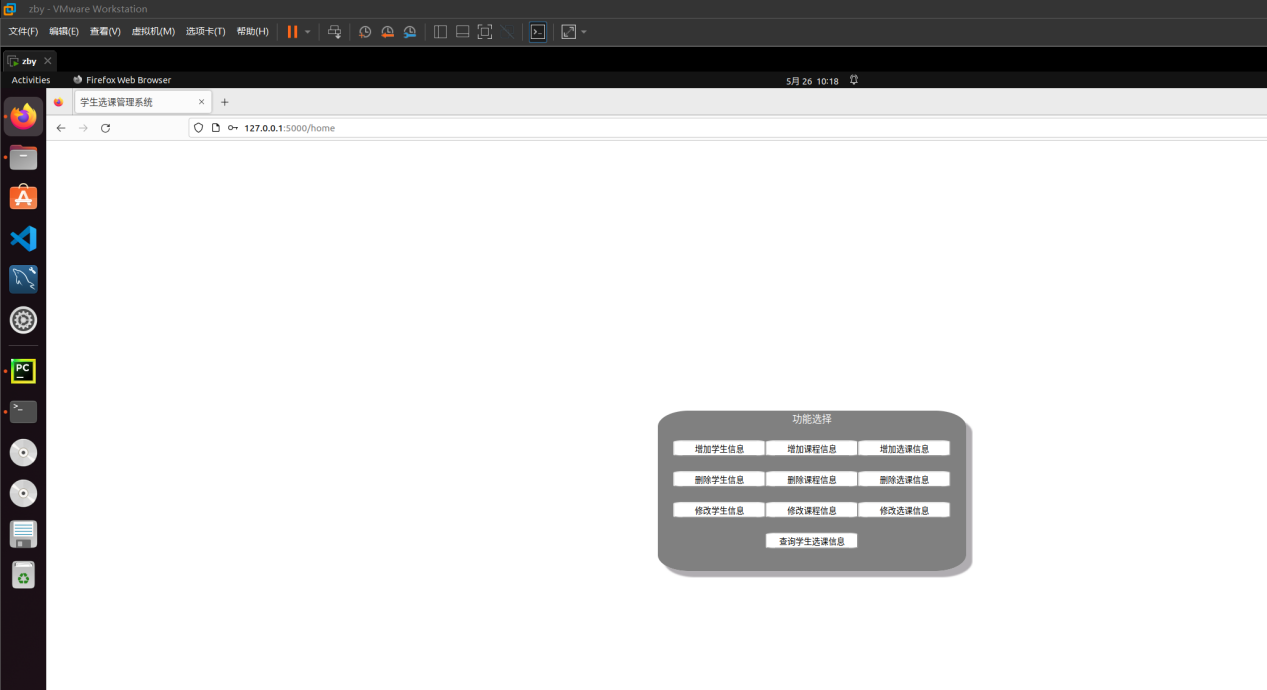


11.实现两种不同用户的登录功能

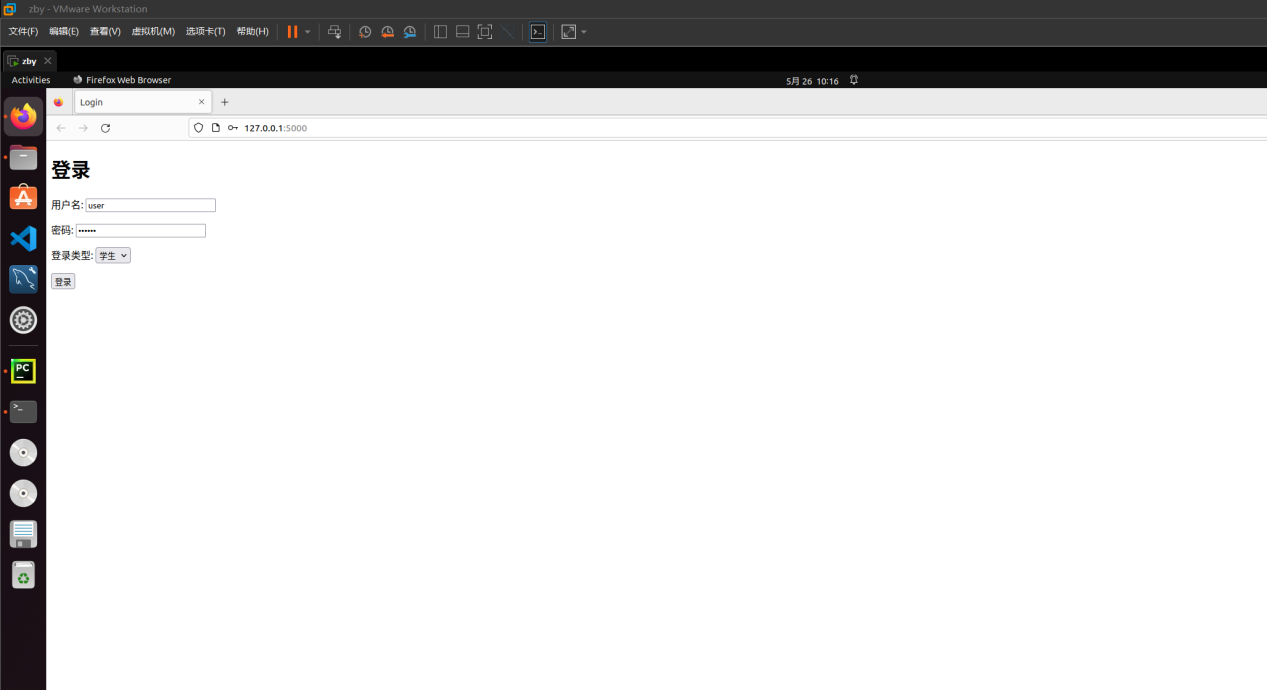
通过下方登陆类型的选择选择用户为学生还是教师从而对其分配不同的数据库操作权限，如下图所示，登录类型为教师，用户名为admin，密码为123456，



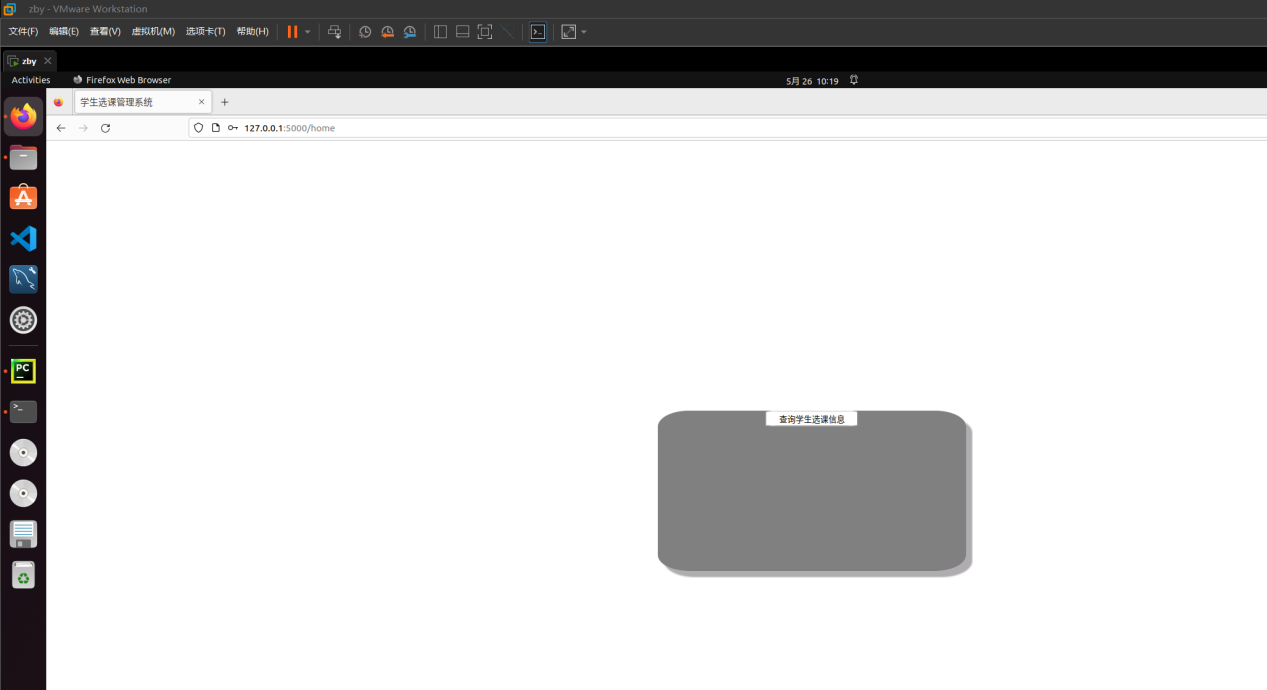
登陆成功后的页面为完整的功能选择页面，拥有操作数据库的全部权限。



登陆类型为学生，用户名为user,密码为123456



登陆成功后跳转到的页面为不完整的功能选择页面，学生仅有根据学号查询选课信息的权限。

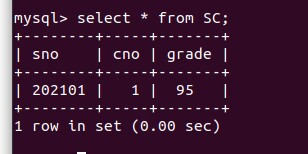


12.实现参照完整性约束的功能

使用页面的删除功能后会出现报错并且pycharm终端中会出现以下错误信息，说明该数据库在设计时实现了参照完整性。

ZKT(T1}02S8R_3@{5QUMR@S

在删除SC对应的元素之后再次进行删除操作即可完成删除



五、实验的不足与改进

1.页面的设计太过简单。在页面设计过程中仅仅实现了数据库对应的功能，但是并未对整体的页面进行系统性排版和布局，使得整体页面虽然有相应的功能性但是并不具备较高的美观性。

2.实现的查询功能过于简陋，仅能根据用户输入的学号对该学生固定的信息进行查询，没有设计多个不同方法的查询按钮使得页面能够支持多功能的查询。

3.设计的网页过多，在实验设计时为了便于将设计页面进行模块化管理，使得每个功能对应一个页面，导致实验后期页面数量较多。应该将相近功能的页面统合在一个大的页面中使得前端代码更为简洁。

4.用户数量过少。该实验中仅设计了两个角色的用户，并未添加更多更复杂的角色的用户，且新用户注册需要在后端的数据库中进行添加操作，并未在前端呈现，使得用户注册和修改密码功能较为繁琐。