**学生专业学习管理**

**信息系统**

计科1701 石京儒 170950113

目录

[一、 项目概述 3](#_Toc38578386)

[二、 技术介绍 3](#_Toc38578387)

[三、 项目分析 4](#_Toc38578388)

[四、 项目设计 7](#_Toc38578389)

[1. 关键数据结构设计 7](#_Toc38578390)

[2. 功能实现设计 7](#_Toc38578391)

[五、 系统实验说明 17](#_Toc38578392)

[六、 小结 21](#_Toc38578393)

[七、 参考文献 22](#_Toc38578394)

# 项目概述

学生专业信息管理系统是为了提供学生在读期间，选择各自专业课程的一个平台，是一个教育单位不可缺少的部分。学生专业信息管理系统应该能够为用户提供充足的信息和快捷的查询手段。

随着科学技术的不断提高，计算机科学日渐成熟，其强大的功能己为人们深刻认识，它已进入人类社会的各个领域并发挥着越来越重要的作用。作为计算机应用的一部分，使用计算机对学生专业信息进行管理，有着检索迅速、查找方便、可靠性高、存储量大、保密性好、寿命长、成本低等优点，能够极大地提高管理的效率。[1]

因此，开发这样一套成绩管理系统成为很有必要的事情。学生专业信息管理系统提供了强大的学生成绩管理管理功能，方便教师对学生成绩等信息的添加、修改、删除、查询等操作，同时方便学生对自己各科成绩查询，进行学习上的交流。

# 技术介绍

本系统采用了前后端分离开发的方法（如图2.1）：后端采用了MVC的框架模式的由Python写成的Web应用框架Django(3.0.4)；前端框架采用基于HTML、CSS、JavaScript的Bootstrap(4.41)和Bootstrap Table(1.16)；数据库部分使用的关系型数据库管理系统MySQL。

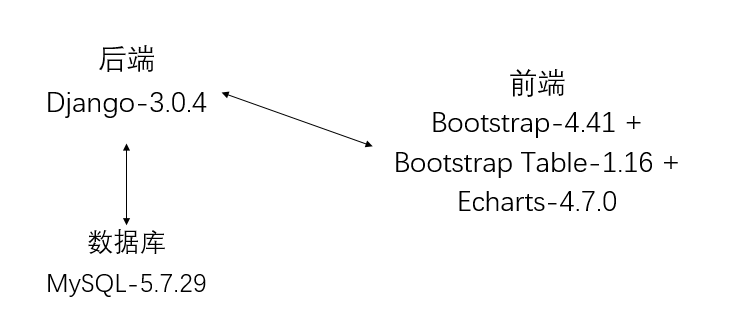


图2.1 前后端架构

本项目采用的三个关键部件均是开放源代码的项目，为日后实际部署降低了总体拥有成本(Total Cost of Ownership)。

模型(Model)-视图(View)-控制器(Controller)是Xerox PARC在二十世纪八十年代为编程语言Smalltalk－80发明的一种软件设计模式，已被广泛使用。这种设计模式有助于管理复杂的应用程序，同时也简化了分组开发。不同的开发人员可同时开发视图、控制器逻辑和业务逻辑。[2]

# 项目分析

就项目本身而言，需要从各个方面进行考虑。下面从题目要求，项目完整性以及开发方式三个方向进行分析

1. **题目要求**
2. 系统可以准确地记录和查询学生信息，包括学生的姓名、单位、年龄、性别以及身份证号码等。
3. 系统可以准确地记录学生地每一次奖惩情况。
4. 系统可以对学校的院系情况进行管理，包括设置学院名称、修改某学院某专业方向的名称等。
5. 系统应该可以对基础数据进行维护。
6. 系统能够对开设的课程进行管理。
7. 学生选课管理、考试(登记分数)、补考重修管理。
8. 系统还应该提供强大数据统计、查询、报表生成以及打印等功能。
9. 用户权限管理。
10. 异常处理。

以上是题目中对学生专业信息管理系统所做的要求，其中可以分为几个部分理解。

1. 要求拥有学生信息管理、奖惩管理、院系管理、课程管理、选课管理、成绩管理、权限管理这七个功能模块
2. 要求对每个功能模块都具有查、插、删和改的数据库操作
3. 要求具有数据统计、异常处理等辅助功能
4. **项目完整性**

选择这个项目的理由，是因为这个项目的要求和我们大学生实际使用的教务管理系统十分接近，有实际使用过的经验。但题目中存在一些说法模糊的部分，需要自行考虑。

1. 权限管理我们分为三个类别：学生、老师和管理员。
2. 异常处理我们采用了比较完善的日志系统，可以对网站的每次访问以及数据库的所有操作分成普通(Info)、警告(Warning)、错误(Error)、调试(Debug)四个等级进行记录。
3. 大规模数据的界面采用表格的方式进行呈现，数据统计使用图表来体现。同时表格提供数据导出的功能，解决报表生成的问题。
4. **开发方式**

项目实际的开发周期是三周的时间，从教学周的第5周到第8周。我们小组由两人组成，代码迭代次数多、代码量大，所以我们决定采用GitHub托管代码(https://github.com/TTopoo/Student)，使用QQ进行沟通，作为协同办公的平台。截至项目尾声，进行了105commit，最高的单周提交达到44次（如图3.1、图3.2）。

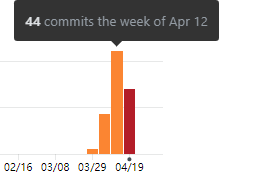


图3.1 单周提交次数分布

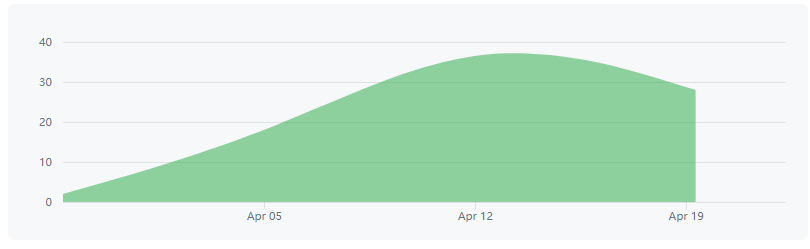


图3.2 每日提交次数分布

数据库我们采用的方案是托管在云服务器上（如图3.3），实际部署在百度云学生机。双方使用同一份数据库，避免了数据库结构的差距可能产生的问题。

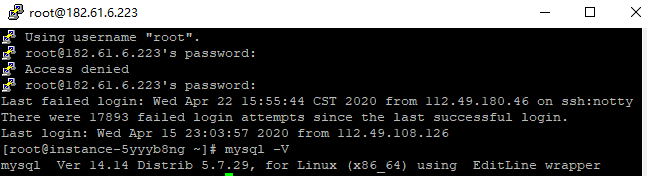


图3.3 部署在云服务器上的数据库

# 项目设计

## 关键数据结构设计

从项目整体角度来说，我根据任务的需求，使用了面向对象的软件开发方法，奠定了整体代码的结构。

项目分为两个类型的对象，使用者和操作（如图4.1）。通过类继承的方式对不同实例内的数据和方法进行更新,提高了代码的重用率，代码结构简洁明了。

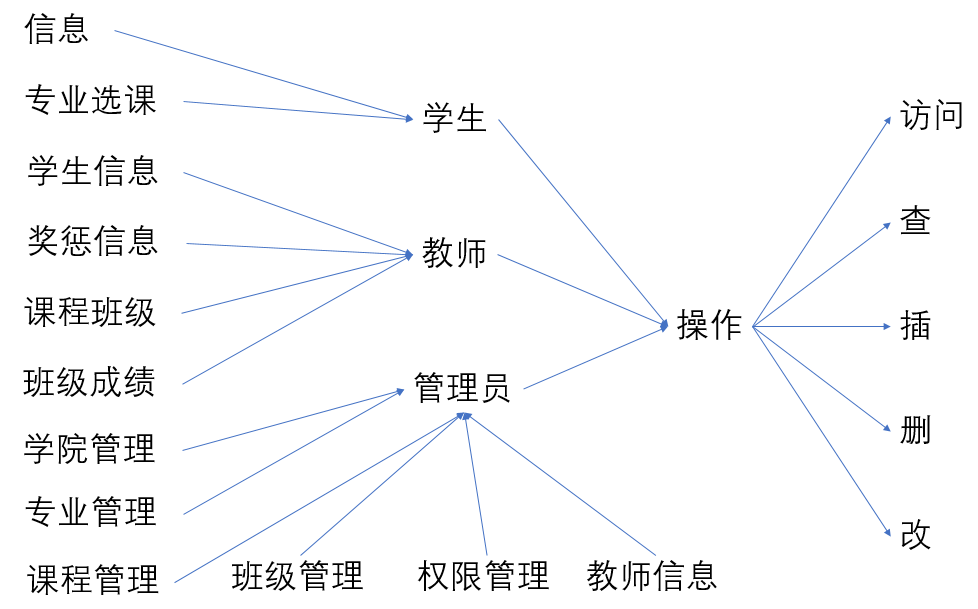


图4.1 面向对象设计模块

## 功能实现设计

1. 网页导航栏模板设计

网页导航栏采用了Bootstrap带有的navbar，实现了导航栏的置顶显示。并且将常用的功能页放在导航栏中，用户可以点击导航进行功能的快速跳转。并且导航栏会根据用户的登陆状态，限制用户可以跳转的功能。（如图4.2、图4.3、图4.4）

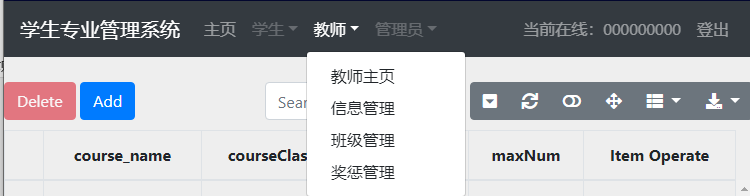
图4.2未登录状态下的导航栏，无法点击学生教师以及管理员。

图4.3教师权限状态下的导航栏，可以点击教师的功能跳转列表。

图4.4通过session中的authority来判断的当前用户可以跳转的界面。

导航栏作为网站的基础，通过Django的模板继承方法，将其设定为一个可以被继承的页面模板，其他的所有页面都从这个页面中增加所需要的脚本即可。（如图4.5）

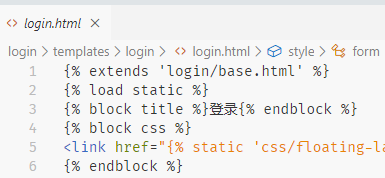


图4.5 其他页面可在block中填写内容，实现对模板页面的内容添加

导航栏还具有页面自适应的功能，当页面比例发生变化时，会自动调整显示的样式。（如图4.6）

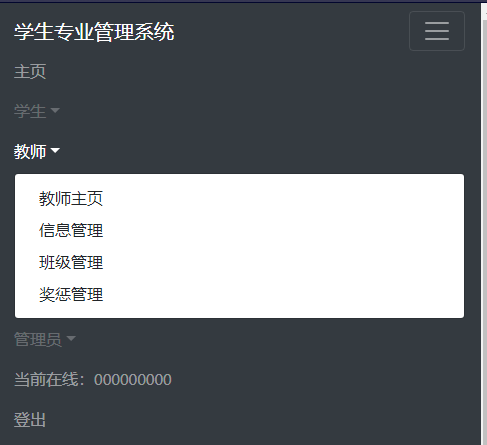


图4.6 页面宽度较小时，变成展开式的导航

1. 登录、登出模块设计

登录登出功能作为管理系统的灵魂功能，涉及到逻辑、以及细节的问题。

1. 导航栏显示逻辑
   1. 登陆前在导航栏显示登录按钮
   2. 登录后在导航栏显示登陆的账号名和登出按钮，取消显示登录按钮。
2. 输入文本框的设计

文本框采用浮动标签表单，输入提示根据用户的输入状态来决定。（如图4.7）

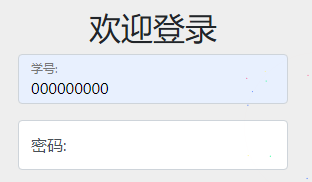


图4.7 输入后标签浮动到文本框内上侧

1. 输入文本框的内容校验

采用了Django的Form类，校验全部在后端进行。有内容非空（如图4.8）以及对应项不正确的提示（如图4.9）

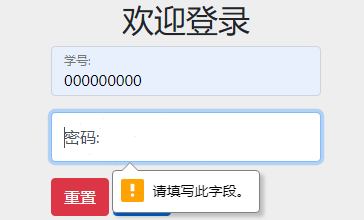


图4.8 内容为空的报错提示



图4.9 密码错误的报错提示

1. 对应的user表，拥有三个属性，分别为账号、密码和注册时间（如图4.10）

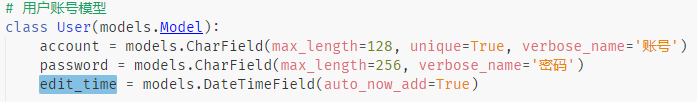


图4.10 User表的内容

1. 登录密码采用一次加盐的MD5加密，成功后向session中存入用户数据。（如图4.11）

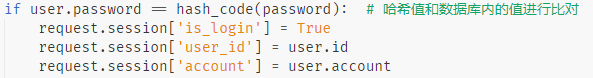


图4.11 session中存入数据

1. 登出时，清除session记录，并重定向至登陆界面。（如图4.12）

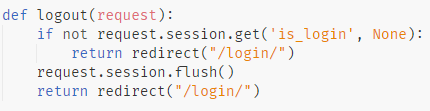


图4.12 清除session并重定向至登录页

1. 操作基类设计

操作基类包括了常见的查、插、删和改四个操作，另外还有权限等级计算和访问功能界面的功能。具有查找所属功能记录以及按名查找函数并执行的操作。

1. 权限等级计算

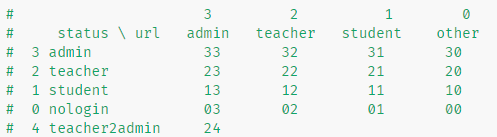


图4.12 清除session并重定向至登录页

根据当前用户的类别以及访问的地址，生成一个状态码，每次进行地址访问时，都需要根据状态码进行判断，若出现状态码异常（包括权限不匹配或是地址错误）时，在日志系统中记录异常。

1. 按名查找函数

当一类用户功能的子功能过多时，根据URL来判断该执行什么函数，在代码上是非常繁琐的事，利用python的特性，将函数存入字典中，根据键来寻找可执行的函数。

例如管理员的学院管理功能中，存有查插删改进这五种操作，采用函数查找字典的方式（如图4.13），不仅可以减少外层代码量，而且更加直观（如图4.14）。



图4.13 函数查找字典

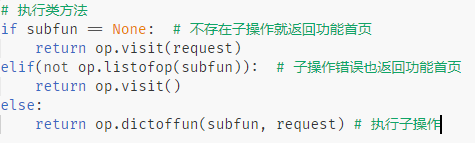


图4.14 外层调用接口的代码

1. 用户对象基类设计

对象基类包括学生、教师和管理员三种，下面以教师为例进行介绍。

每个对象基类都包括四个部分，初始化方法、析构方法、访问用户主页方法以及用户功能查找方法。（如图4.15）



图4.15 对象类中存在的方法

在初始化和析构方法中，调用日志中写入记录。在访问方法中，重定向至教师主页。函数查找方法中用于检查当前用户的功能是否在用户中存在。

1. 信息表格显示模板设计

信息表格显示模板主要包括了前端以及后端的部分。

1. 前端部分

前端界面主要由Bootstrap Table和Bootstrap的模态框（如图4.16）组成。

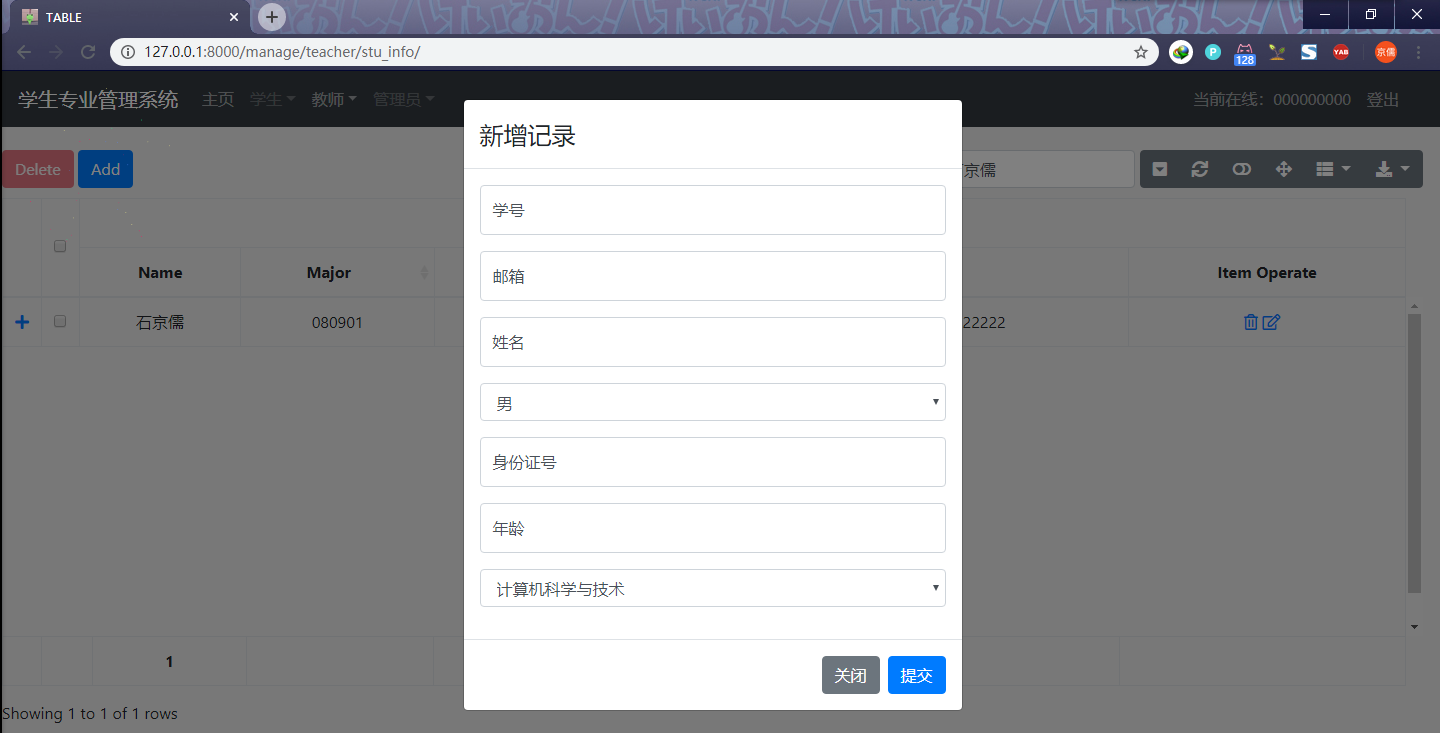


图4.16 模态框用于增加或是修改信息，修改信息时会对已有信息自动填充

表格的显示支持选择显示列（如图4.17）。

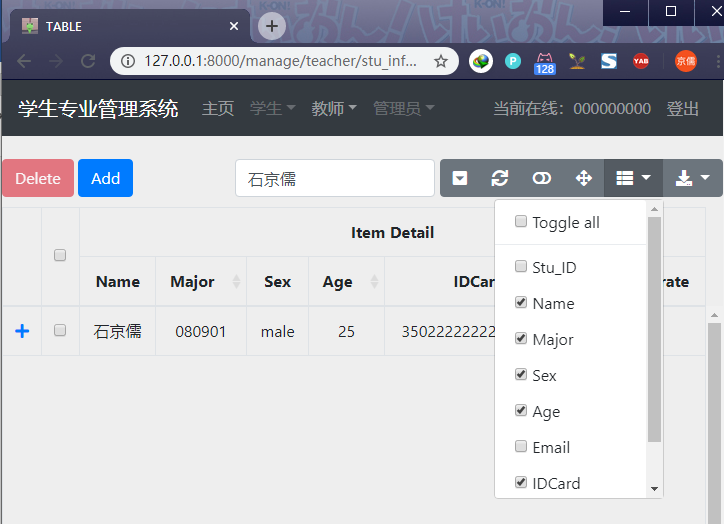


图4.17 下拉列表可以选择显示的列

查看列具体信息（如图4.18）。

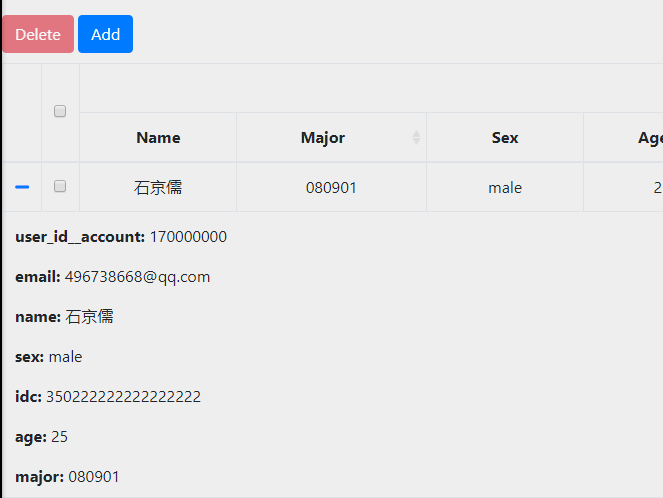


图4.18 左侧展开栏可以查看详情

可以对单列或是多列进行删改（如图4.19）。

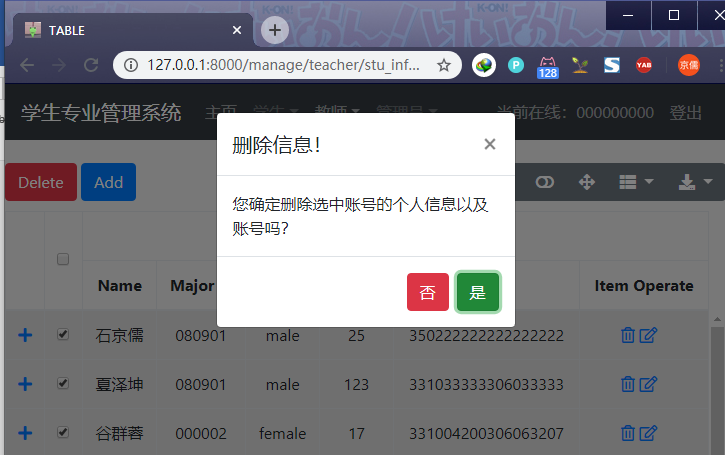


图4.19 多选删除功能

表格支持多种格式导出文件到本地（如图4.20）。

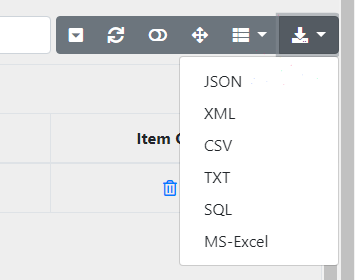


图4.20 可以选择保存的格式

1. 后端部分

前后端的交互采用ajax技术，并采用Django的跨站请求伪造(Cross-site request forgery)验证机制（如图4.21），防止跨站请求伪造攻击。



图4.21 ajax携带验证指令

对于插删改的操作，文本框的数据与先前登录模块一样，在后端进行验证。通过非阻塞通知的JavaScript库toastr，实现了错误弹出提示框（如图4.22）。



图4.22 验证结果反馈至前端

对于查询操作，实现了对全列字段的模糊搜索（如图4.23、图4.24），并且实现了服务端分页（如图4.25），节省服务器资源。

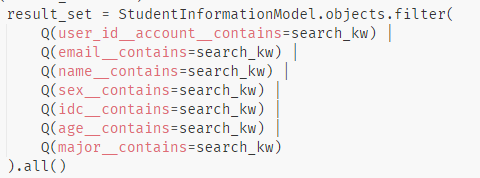


图4.23 模糊搜索后端实现

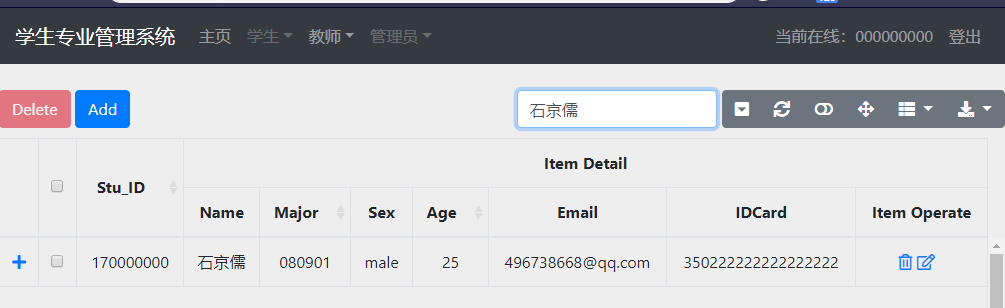


图4.24 模糊搜索前端效果

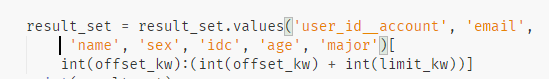


图4.25 服务器分页后端实现

1. 日志系统工具类设计

日志根据是否涉及数据库操作，分为两种类型。

涉及数据库操作的日志系统，会对每次数据库操作进行记录，并保存在数据库中。（如图4.26）

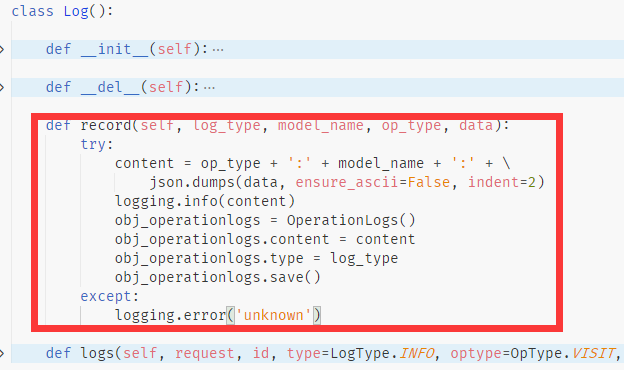


图4.26 对数据库的操作保留日志

不涉及数据库操作的日志系统，会对三种类型的信息进行记录，包括用户类实例的声明与析构，用户功能类的调用和结束，URL权限正确或错误。

# 系统实验说明

主要用于展示网站登录到使用的流程。

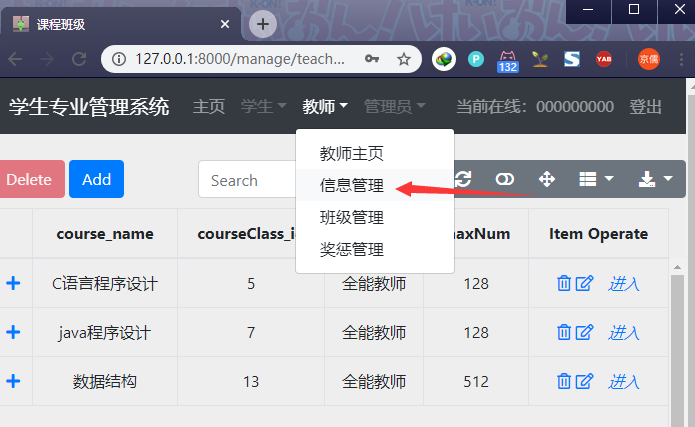
1. 登录页（此处以教师账号登录）

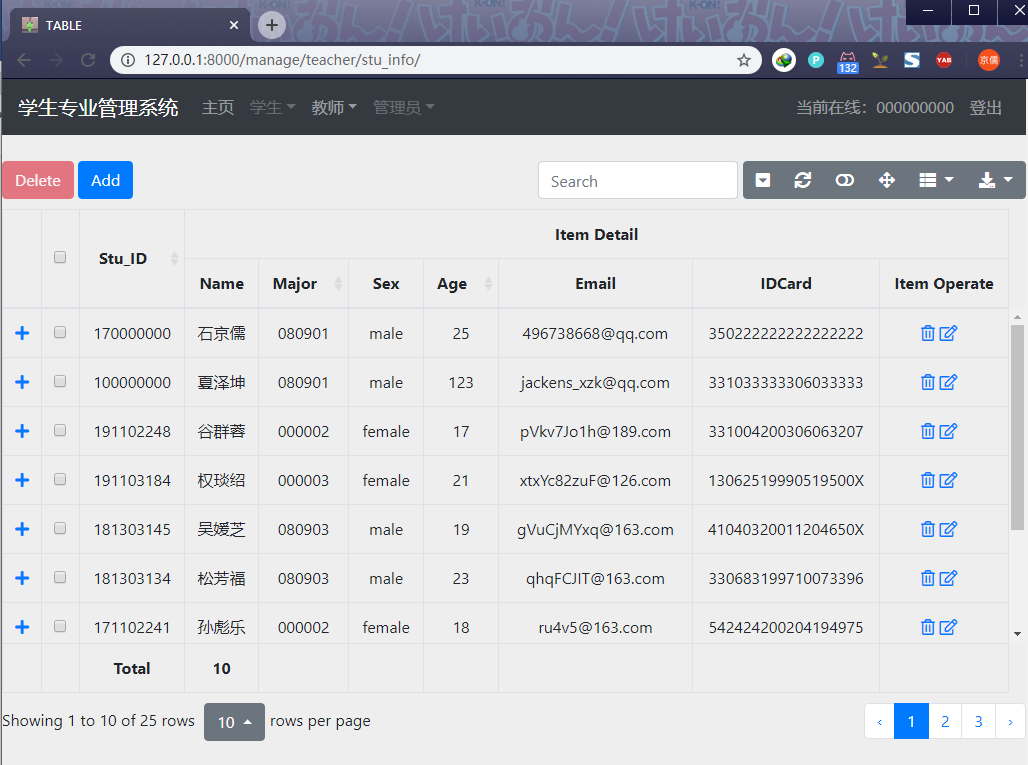


1. 登录后跳转到教师主页（默认显示该教师所教学的班级）

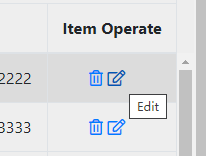


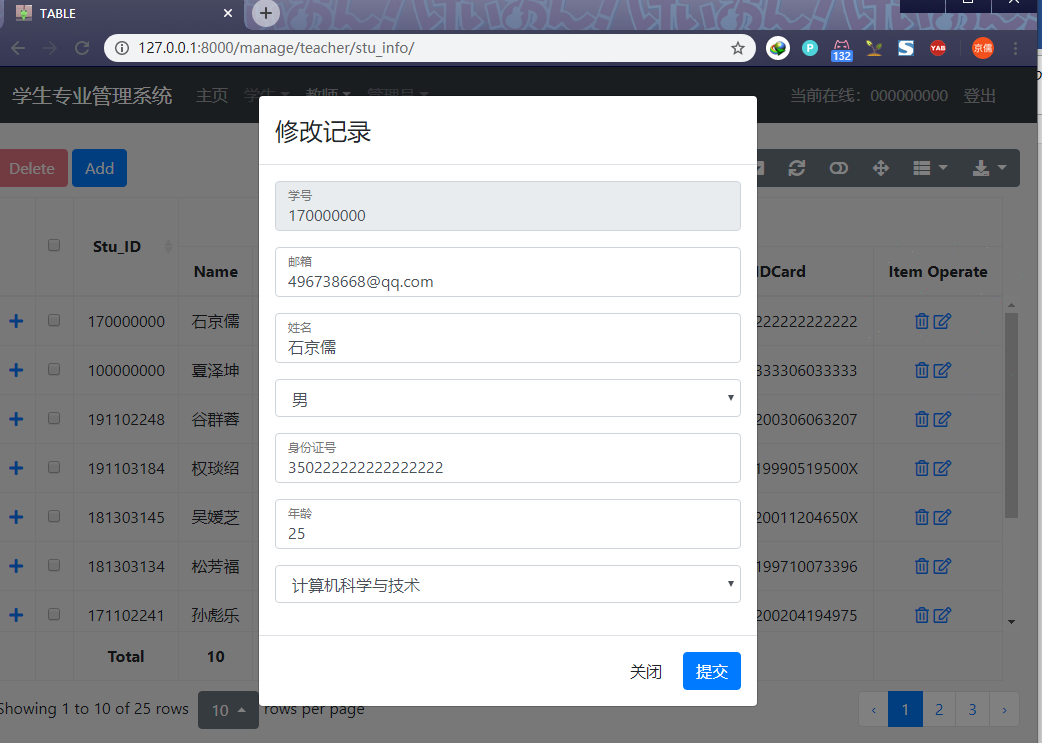
1. 切换至学生信息管理页面





1. 点击图标编辑学生信息

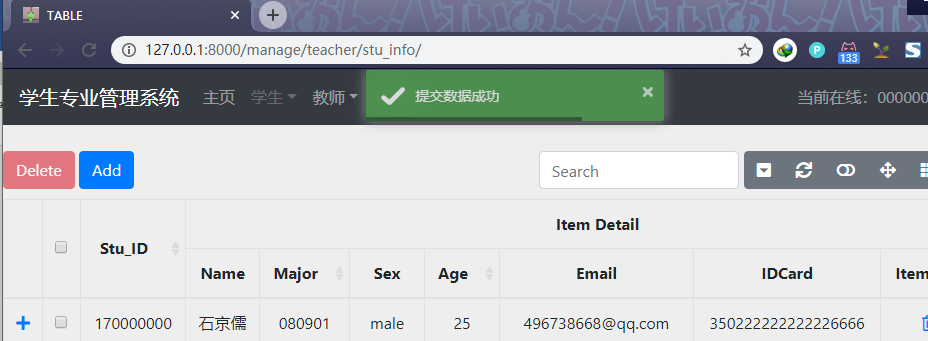




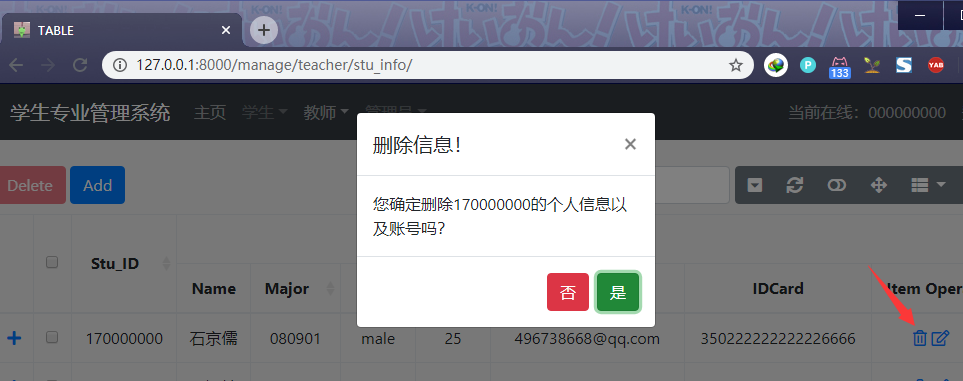
1. 可以看到自动填充的学生信息，并对其身份证号做修改。



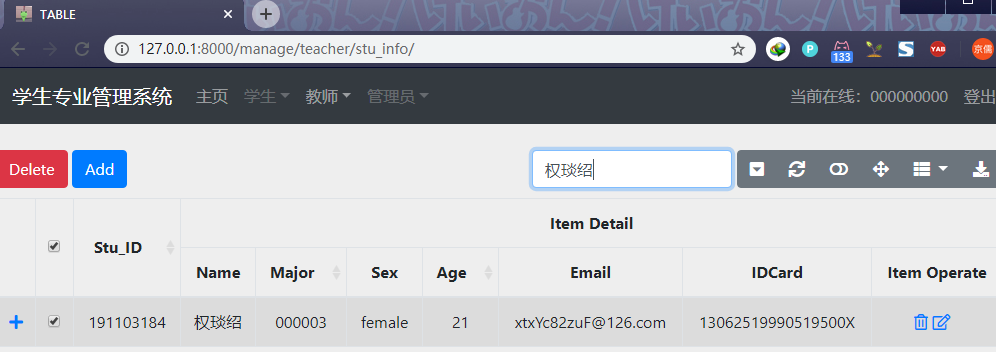
1. 修改成功后弹出成功提示。



1. 点击右侧删除按钮，可以对单条记录进行删除。



1. 可以使用上方搜索栏对记录进行搜索



# 小结

小结分为两个部分，一是关于项目的结语，二是关于个人的感悟。

对于项目本身而言，整体的难度是非常大的。我们选择的学生专业信息管理系统，由于非常贴合学习生活实际，我们对标的方向是东华大学的教务管理系统，因此在项目初期，虽然有一个比较好的实现方向，但是其中还是遇到了诸多问题。

首先是先修课程的问题，我是没有选择企业JAVA这门课程的，所以在做个人任务的时候就明显感觉需要查过多的资料来解决一些问题，所以我在决定项目框架的时候，偏向了选择非JAVA语言的框架，选择了基于Python的Django框架，减少了自学的成本。

其次是项目协同开发的问题，由于疫情原因，双方无法在机房或是实验室一同开发，导致我们项目初期的代码版本总是出现不一致，以及数据库不同步的问题。后来在采用了云服务器搭建数据库，并且使用GitHub来协助版本控制，才算是勉强解决了同步性的问题。

在项目中期以及末期，时间紧张，双方都有几门大作业或是报告需要写，并且项目功能对比其他小组而言实在是多太多了，对于两个人的小组来说过于困难，紧密的开发进度安排成为了最头疼的事情，好在双方都生活自律不拖沓，最终才顺利完成了开发。

最后是一些课程设计组队以及选题的反思。通过这次数据库设计，让我意识到了选题的重要性。在选题初期由于自己思维简单而放下的错误，导致小组两个人在项目上花费了三周的课余时间（因为水平菜，几乎是全天都在做），来做一个极具挑战性并且功能结构相比其他小组更加复杂的项目。因此选择课题应量力而行，把队员收满，选择一个功能较少的项目可以极大地节约组内成员的时间。

# 参考文献

[1] https://wenku.baidu.com/view/63bdb7c5974bcf84b9d528ea81c758f5f71f299c.html

[2] <https://baike.baidu.com/item/MVC%E6%A1%86%E6%9E%B6>