学生选课系统--

设计报告

姓名：

班级：

学号：

指导教师：

目录

[第一部分：开发前的分析与设计 4](#_Toc42331209)

[题目要求 4](#_Toc42331210)

[需求分析 4](#_Toc42331211)

[概念结构设计 5](#_Toc42331212)

[逻辑结构设计 5](#_Toc42331213)

[实体 5](#_Toc42331214)

[联系 5](#_Toc42331215)

[数据库表结构设计 6](#_Toc42331216)

[概览： 6](#_Toc42331217)

[学生表 6](#_Toc42331218)

[教师表 7](#_Toc42331219)

[课程表 7](#_Toc42331220)

[选课表 7](#_Toc42331221)

[课程安排表 8](#_Toc42331222)

[公告表 8](#_Toc42331223)

[用户登录信息表 8](#_Toc42331224)

[第二部分：代码开发工作 9](#_Toc42331225)

[开发环境相关： 9](#_Toc42331226)

[开发所用技术： 9](#_Toc42331227)

[源代码目录简介： 9](#_Toc42331228)

[项目代码 10](#_Toc42331229)

[数据库模型代码: 10](#_Toc42331230)

[第三部分：界面展示 14](#_Toc42331231)

[用户登录 14](#_Toc42331232)

[学生相关 15](#_Toc42331233)

[学生首页 15](#_Toc42331234)

[选课 15](#_Toc42331235)

[查看课表 16](#_Toc42331236)

[查看个人信息 16](#_Toc42331237)

[成绩查询 17](#_Toc42331238)

[修改密码 17](#_Toc42331239)

[教师相关 18](#_Toc42331240)

[教师首页 18](#_Toc42331241)

[个人信息维护 18](#_Toc42331242)

[课程评分和修改 19](#_Toc42331243)

[查看公告 20](#_Toc42331244)

[授课查看 20](#_Toc42331245)

[修改密码 21](#_Toc42331246)

[管理员相关 21](#_Toc42331247)

[管理员登入 21](#_Toc42331248)

[管理员首页 22](#_Toc42331249)

[安排课程 22](#_Toc42331250)

[添加学生 23](#_Toc42331251)

[添加教师 23](#_Toc42331252)

[添加新课 24](#_Toc42331253)

[运行控制 24](#_Toc42331254)

[发布公告 25](#_Toc42331255)

[第四部分：总结与反思 25](#_Toc42331256)

# 第一部分：开发前的分析与设计

## 题目要求

**建立一个学生选课系统, 编写应用程序完成系统开发。**

1. 建立基本表：

学生（学号，姓名，性别，年龄，系别）

课程（课程号，课程名，开课系，任课教师）

学习（学号，课程号，成绩）

1. 要求有以下子系统

（1）基本信息查询子系统

要求：能够显示出学生表、课程表、带参数学生详细信息和选课情况

（2）学生、课程信息维护子系统

要求：能够维护学生表和课程表

（3）学生选课子系统

要求：考虑参照完整性以及用户自定义完整性，实现学生选课功能

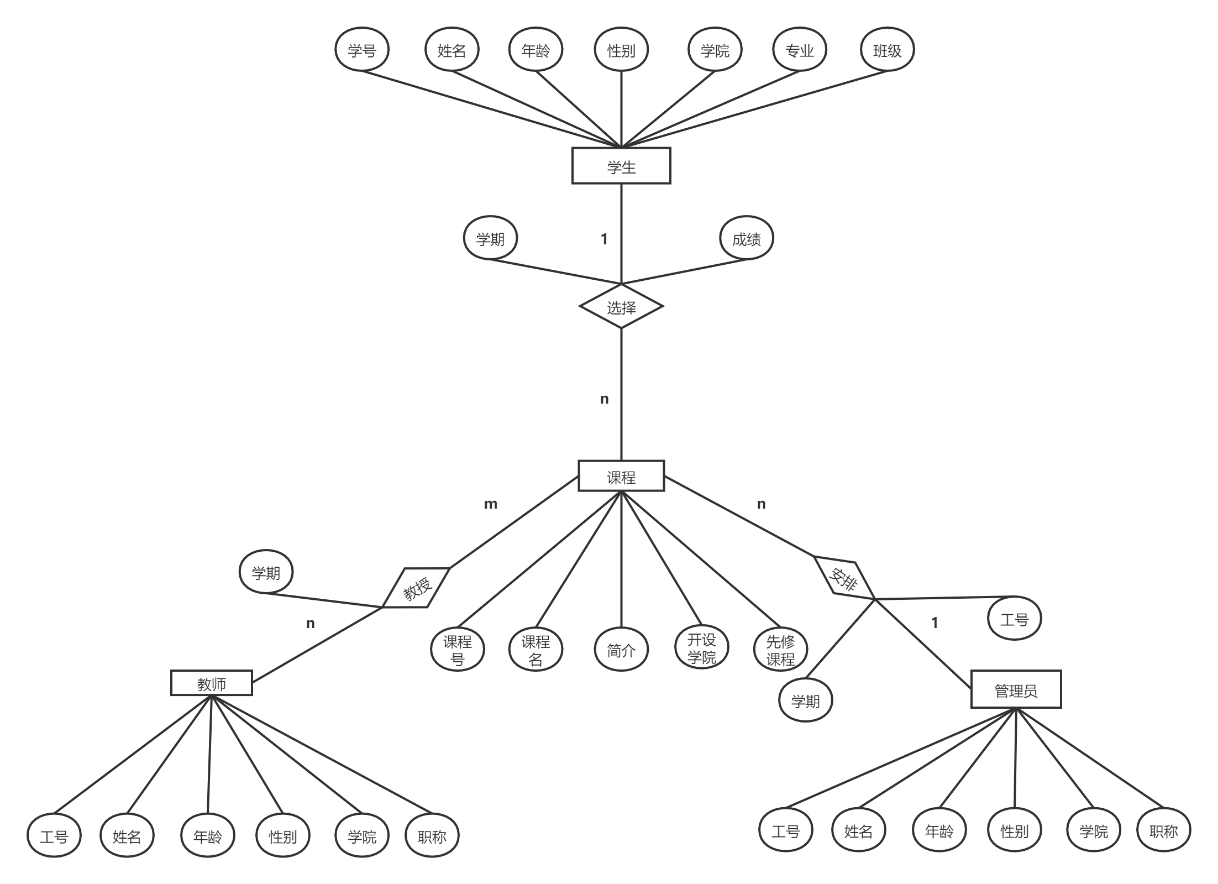
（4）系统维护

## 需求分析

1. 根据题目建立了学生选课系统，并且在此基础上实现了具备教务管理和信息查询及维护的教务管理系统，选课系统只是这个教务管理系统的一个子系统。
2. 教务管理系统根据学生、教师以及管理员三种身份实现了不同的页面及功能。
3. 学生登录后有相关的信息查询和部分信息修改的权限，信息查询包括查询自己的个人信息，选课信息，成绩信息；修改权限包括学生部分个人信息的修改，在选课阶段修改自己的所选课程信息，修改自己的登录密码等。
4. 任课教师登录后，可以查看自己现在这个学期教授哪些课程，可以查看这门课的详细教学安排时间表，这门课的上课地点等信息，可以查看选修了这门课的所有学生的信息，查看自己的个人信息。任课教师可以修改自己部分的个人信息，修改自己的登录密码，给选修了自己课程的学生进行评分以及修改分数等操作。
5. 管理员登录后，可以添加新课、安排课程、发布公告并对课程进行运行控制，可以添加新的教职工信息和学生信息。

## 概念结构设计

**系统的ER图如下：**



## 逻辑结构设计

### 实体

学生（学号，姓名，性别，学院，专业，班级，状态等）

教师（工号，姓名，性别，学院，职称，电话等）

管理员（工号，姓名，性别，学院，职称，电话等）

课程（课程号，课程名称，开课学院，先修课程号，简介，状态）

### 联系

选修（学期，学号，课程号，成绩）

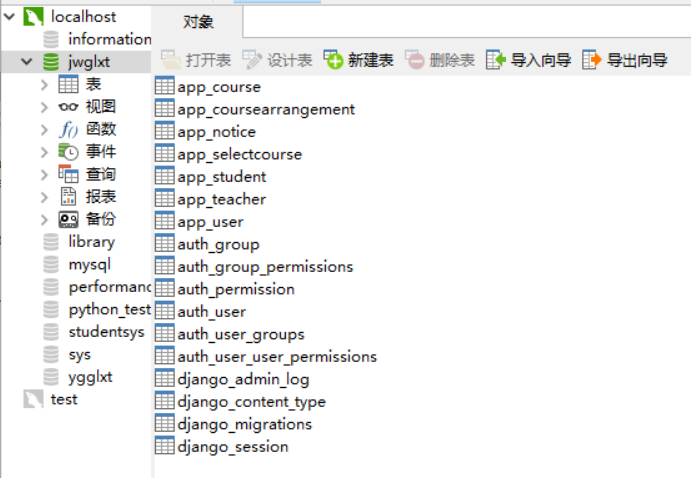
教授（学期，工号，课程号，教学开始周，教学结束周，教学班容量，教室地址，状态）

安排（学期，工号，课程号，教学开始周，教学结束周，教学班容量，教室地址，状态）

## 数据库表结构设计

**数据表信息如下：**

### 概览：

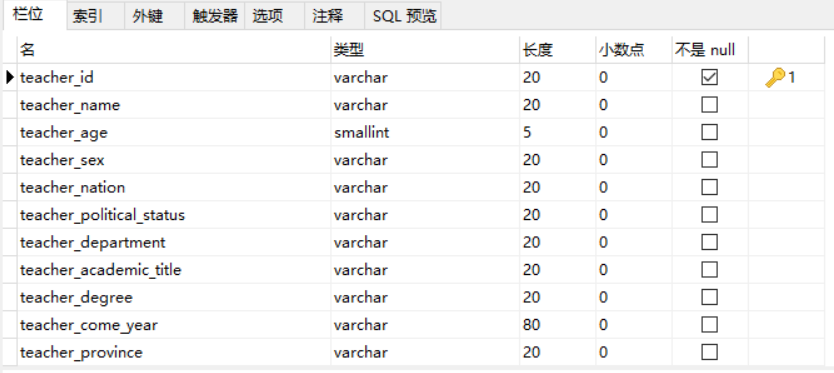


数据库名称为jwglxt，数据表数量为7个（Django自带除外）

### 学生表



### 教师表



### 课程表



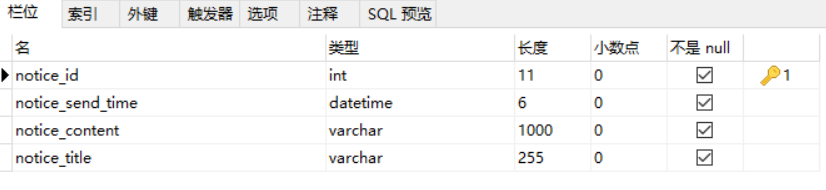
### 选课表



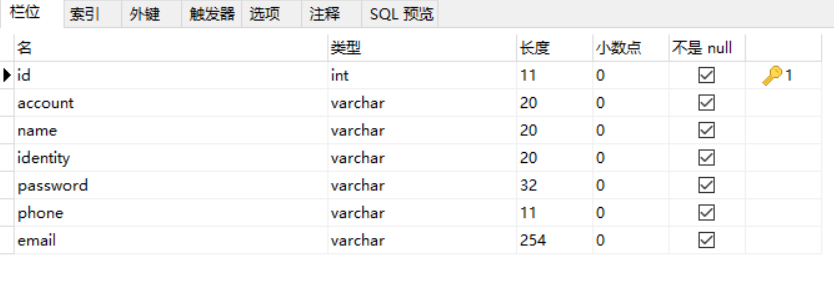
### 课程安排表



### 公告表



### 用户登录信息表



# 第二部分：代码开发工作

## 开发环境相关：

开发后台服务语言：Python 版本3.7

Web 框架： Django 版本 2.2.11

数据库：MySql 8.0.12

操作系统：Windows10

开发工具：Pycharm 2020.1、Navicat for MySQL

## 开发所用技术：

Web前端：Django，HTML，CSS，JavaScript，Bootstrap

Web后端：PyMysql

版本控制工具：Git 2.26.1

### 源代码目录简介：

app：项目主要文件夹，用于存放逻辑代码、html文件等

migration：用于存放Django自动生成的数据库迁移文件，以及每次数据库结构修改的操作

templates：用于存放相关页面

student：学生相关页面

teacher：教师及管理员相关页面

admin.py：配置Django后台超级管理员

apps.py：向项目声明App应用

models.py：配置数据库模型，每次修改之后都必须执行迁移命令

test.py：测试文件

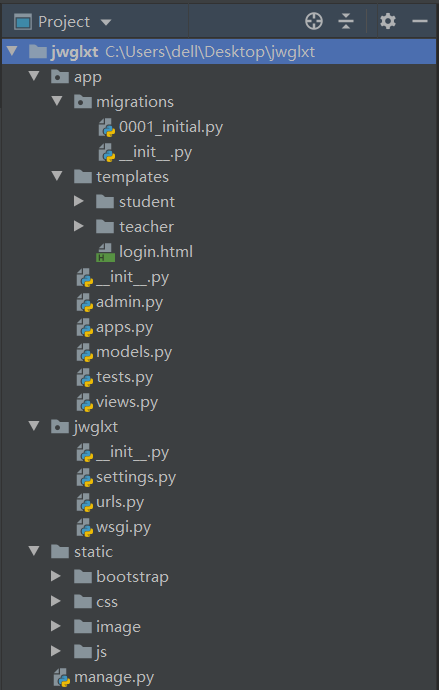
views.py：实现与前端交互的代码

jwglxt：项目的配置文件

url.py：配置项目的路由

setting.py：项目配置文件，配置数据库，静态文件目录，项目语言、时区等

static：静态css、js文件



## 项目代码

### 数据库模型代码:

from django.db import models  
  
class Student(models.Model):  
  
 *'''学生信息表'''* student\_id = models.CharField(max\_length=20, verbose\_name='学号', primary\_key=True, unique=True, db\_index=True)  
 student\_name = models.CharField(max\_length=20, verbose\_name='姓名', null=True)  
 student\_age = models.PositiveSmallIntegerField(verbose\_name='年龄', null=True)  
 student\_sex = models.CharField(max\_length=20, default='男', verbose\_name='性别', null=True)  
 student\_nation = models.CharField(max\_length=20, default='汉族', verbose\_name='民族', null=True)  
 student\_political\_status = models.CharField(max\_length=20, default='群众', verbose\_name='政治面貌', null=True)  
 student\_come\_year = models.CharField(max\_length=40, verbose\_name='入学日期', null=True)  
 student\_college = models.CharField(max\_length=20, default='信息科学与技术学院', verbose\_name='学院', null=True)  
 student\_major = models.CharField(max\_length=20, verbose\_name='专业', null=True)  
 student\_class = models.CharField(max\_length=20, verbose\_name='班级', null=True)  
 student\_province = models.CharField(max\_length=20, verbose\_name='省份', null=True)  
 student\_city = models.CharField(max\_length=20, verbose\_name='城市', null=True)  
 student\_birthday = models.CharField(max\_length=40, verbose\_name='出生日期', null=True)  
 #student\_phone = models.CharField(max\_length=20, verbose\_name='手机号码', null=True)  
 student\_qq = models.CharField(max\_length=20, verbose\_name='QQ号码', null=True)  
 student\_wechat = models.CharField(max\_length=20, verbose\_name='微信号码', null=True)  
 #student\_email = models.EmailField(verbose\_name='邮件地址', null=True)  
 student\_high\_school = models.CharField(max\_length=20, verbose\_name='毕业高中', null=True)  
 student\_foreign\_language = models.CharField(max\_length=20, default='英语', verbose\_name='外语', null=True)  
 student\_status = models.CharField(max\_length=20, default='在读', verbose\_name='状态', null=True)  
 #student\_image  
  
 class Meta:  
 verbose\_name = '学生信息表'  
 verbose\_name\_plural = verbose\_name  
  
class Teacher(models.Model):  
  
 *'''教师信息表'''* teacher\_id = models.CharField(max\_length=20, verbose\_name='工号', primary\_key=True, unique=True, db\_index=True)  
 teacher\_name = models.CharField(max\_length=20, verbose\_name='姓名', null=True)  
 teacher\_age = models.PositiveSmallIntegerField(verbose\_name='年龄', null=True)  
 teacher\_sex = models.CharField(max\_length=20, default='男', verbose\_name='性别', null=True)  
 teacher\_nation = models.CharField(max\_length=20, default='汉族', verbose\_name='民族', null=True)  
 teacher\_political\_status = models.CharField(max\_length=20, default='群众', verbose\_name='政治面貌', null=True)  
 teacher\_department = models.CharField(max\_length=20, verbose\_name='学院', default='信息科学与技术学院', null=True)  
 teacher\_academic\_title = models.CharField(max\_length=20, default='教授', verbose\_name='职称', null=True)  
 teacher\_degree = models.CharField(max\_length=20, default='学士', verbose\_name='学位', null=True)  
 teacher\_come\_year = models.CharField(max\_length=80, verbose\_name='入职日期', null=True)   
 teacher\_province = models.CharField(max\_length=20, verbose\_name='省份', null=True)  
 teacher\_city = models.CharField(max\_length=20, verbose\_name='城市', null=True)  
 teacher\_birthday = models.CharField(max\_length=255, verbose\_name='出生日期', null=True)   
 #teacher\_phone = models.CharField(max\_length=20, verbose\_name='手机号码', null=True)  
 teacher\_qq = models.CharField(max\_length=20, verbose\_name='QQ号码', null=True)  
 teacher\_wechat = models.CharField(max\_length=20, verbose\_name='微信号码', null=True)  
 #teacher\_email = models.EmailField(verbose\_name='邮件地址', null=True)  
 teacher\_graduate\_school = models.CharField(max\_length=20, verbose\_name='毕业院校', null=True)  
  
 class Meta:  
 verbose\_name = '教师信息表'  
 verbose\_name\_plural = verbose\_name  
  
class Course(models.Model):  
 *'''课程信息表'''* course\_id = models.CharField(max\_length=20, verbose\_name='课程号', primary\_key=True, unique=True, db\_index=True)  
 course\_name = models.CharField(max\_length=20, verbose\_name='课程名', unique=True)  
 course\_status = models.CharField(max\_length=20, default='未开', verbose\_name='课程状态')  
 course\_college = models.CharField(max\_length=20, default='信息科学与技术学院', verbose\_name='开课学院')  
 course\_pre\_id = models.CharField(max\_length=20, verbose\_name='先修课程号', null=True)  
 course\_pre\_name = models.CharField(max\_length=20, verbose\_name='先修课程名', null=True)  
 course\_introduction = models.CharField(max\_length=1000, verbose\_name='课程介绍')  
 course\_scale = models.CharField(max\_length=20, verbose\_name='平时成绩比例')  
 course\_volume = models.CharField(max\_length=20, verbose\_name='课程容量')  
  
 class Meta:  
 verbose\_name = '课程信息表'  
 verbose\_name\_plural = verbose\_name  
  
class SelectCourse(models.Model):  
 *'''选课表'''* student\_id = models.ForeignKey(to='Student', on\_delete=models.DO\_NOTHING, verbose\_name='学号')  
 term = models.CharField(max\_length=20, verbose\_name='学期')  
 course\_id = models.ForeignKey(to='Course', on\_delete=models.DO\_NOTHING, verbose\_name='课程号')  
 teacher\_id = models.ForeignKey(to='Teacher', on\_delete=models.DO\_NOTHING, verbose\_name='工号')  
 student\_usual\_grade = models.CharField(max\_length=20, verbose\_name='平时成绩')  
 student\_final\_grade = models.CharField(max\_length=20, verbose\_name='期末成绩')  
 student\_total\_grade = models.CharField(max\_length=20, verbose\_name='总评成绩')  
  
 class Meta:  
 verbose\_name = '选课表'  
 verbose\_name\_plural = verbose\_name  
 unique\_together = ('student\_id', 'term', 'course\_id', 'teacher\_id')  
  
class CourseArrangement(models.Model):  
 *'''课程安排表'''* course\_id = models.ForeignKey(to='Course', on\_delete=models.DO\_NOTHING, verbose\_name='课程号')  
 term = models.CharField(max\_length=20, verbose\_name='学期')  
 teacher\_id = models.ForeignKey(to='Teacher', on\_delete=models.DO\_NOTHING, verbose\_name='工号')  
 week = models.CharField(max\_length=20, verbose\_name='周次')  
 session = models.CharField(max\_length=20, verbose\_name='节次')  
 location = models.CharField(max\_length=20, verbose\_name='地点')  
 weekday = models.CharField(max\_length=20, verbose\_name='上课日期')  
  
 class Meta:  
 verbose\_name = '课程安排表'  
 verbose\_name\_plural = verbose\_name  
 unique\_together = ('course\_id', 'term', 'teacher\_id',)  
  
class Notice(models.Model):  
 *'''公告表'''* notice\_id = models.AutoField(primary\_key=True)  
 notice\_send\_time = models.DateTimeField(auto\_now\_add=True, verbose\_name='发送日期')  
 notice\_content = models.CharField(max\_length=1000, verbose\_name='通知内容')  
 notice\_title = models.CharField(max\_length=255, verbose\_name='通知标题')  
  
 class Meta:  
 verbose\_name = '公告表'  
 verbose\_name\_plural = verbose\_name  
  
class User(models.Model):  
 *'''用户登录表'''* account = models.CharField(max\_length=20, verbose\_name='账号')  
 name = models.CharField(max\_length=20, verbose\_name='姓名')  
 identity = models.CharField(max\_length=20, default='学生', verbose\_name='身份')  
 password = models.CharField(max\_length=32, verbose\_name='密码')  
 phone = models.CharField(max\_length=11, verbose\_name='手机号码', unique=True)  
 email = models.EmailField(verbose\_name='邮件地址', unique=True)  
  
 class Meta:  
 verbose\_name = '用户登录表'  
 verbose\_name\_plural = verbose\_name  
 unique\_together = ('account', 'identity',)

因为代码很多，word不便于查看，在此提供源代码地址：

<https://github.com/Lhw-686/buct->

# 第三部分：界面展示

## 用户登录

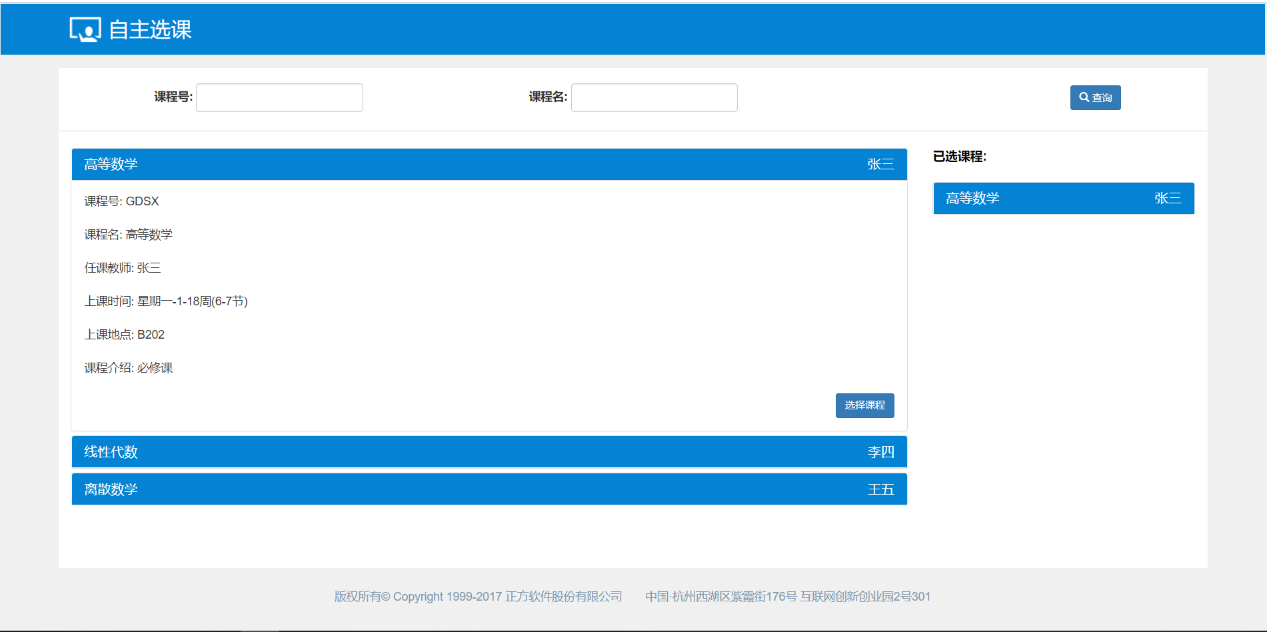


## 学生相关

### 学生首页



### 选课



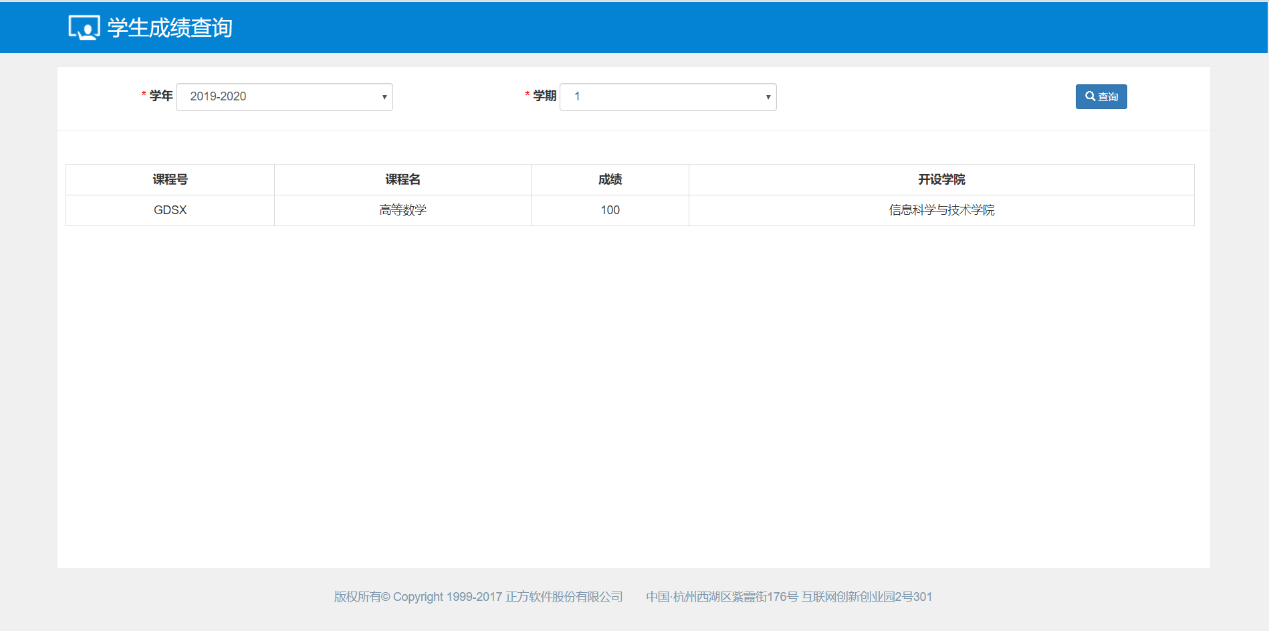
### 查看课表



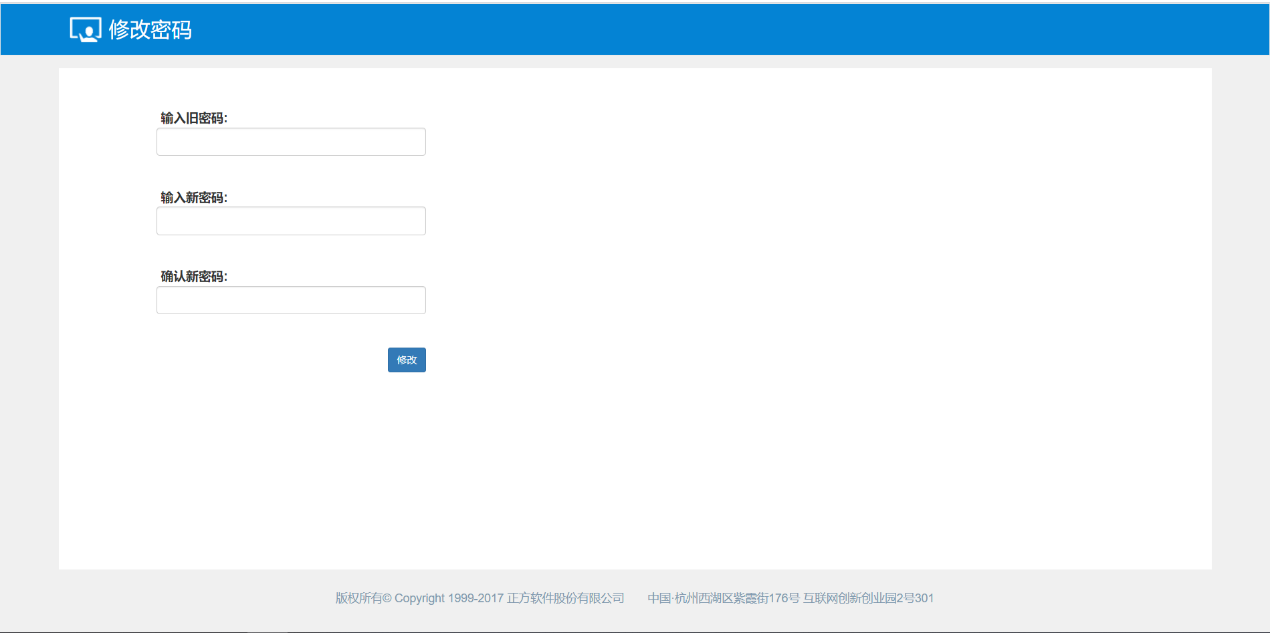
### 查看个人信息



### 成绩查询

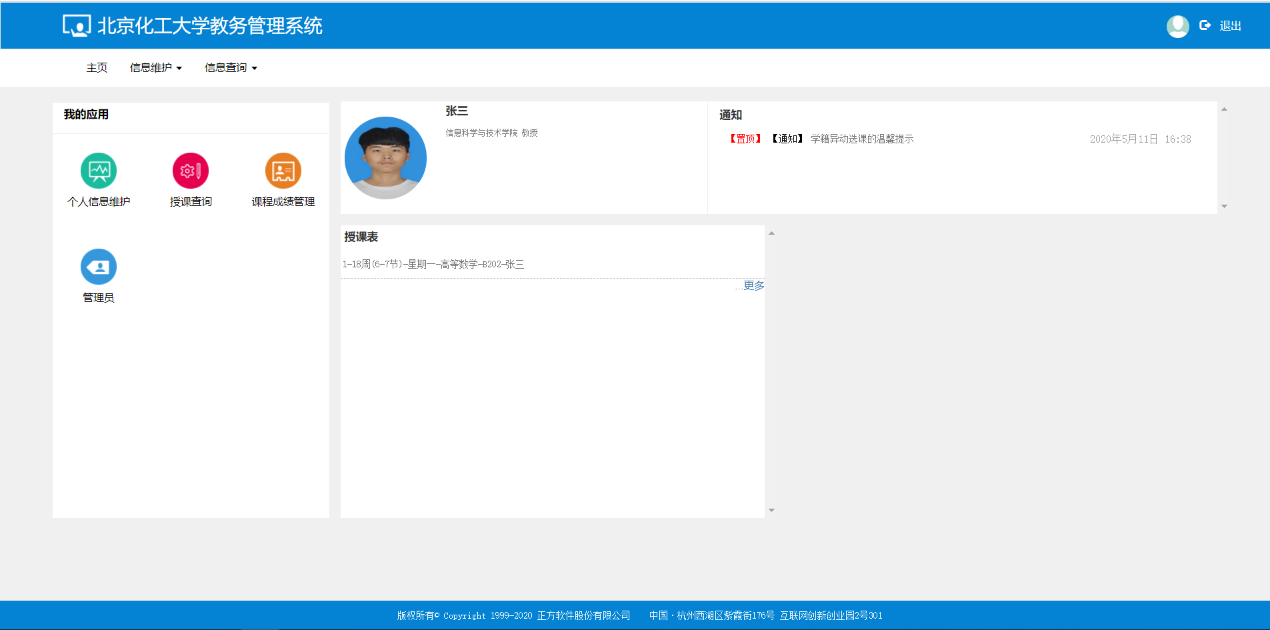


### 修改密码



## 教师相关

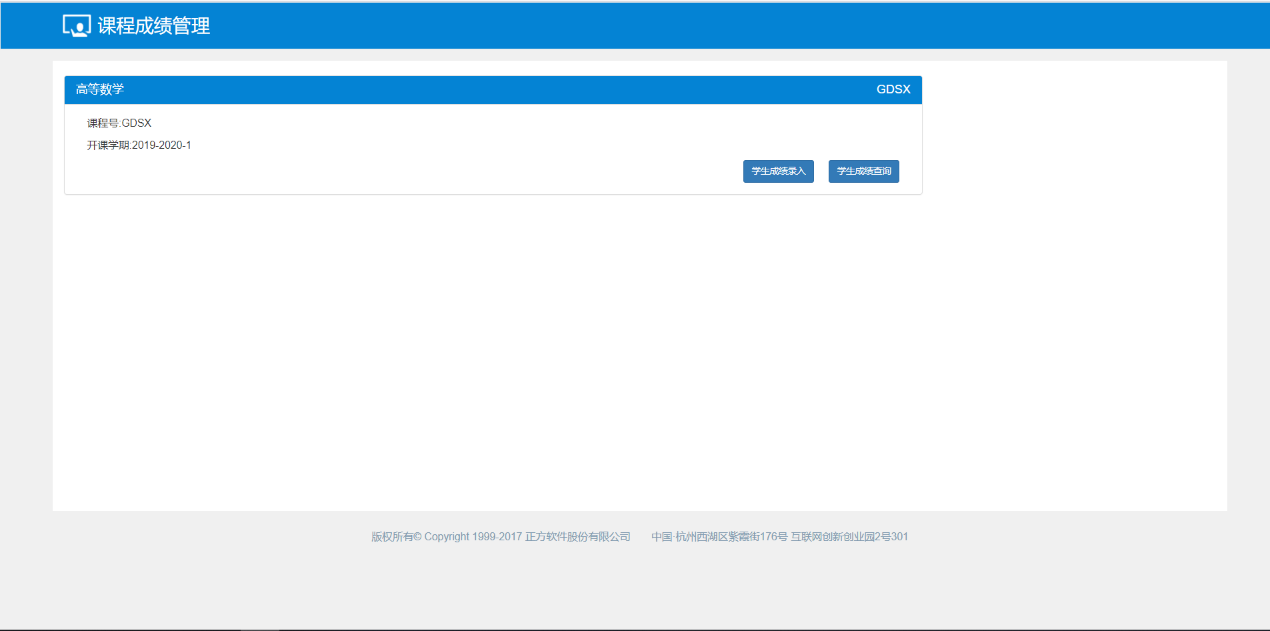
### 教师首页

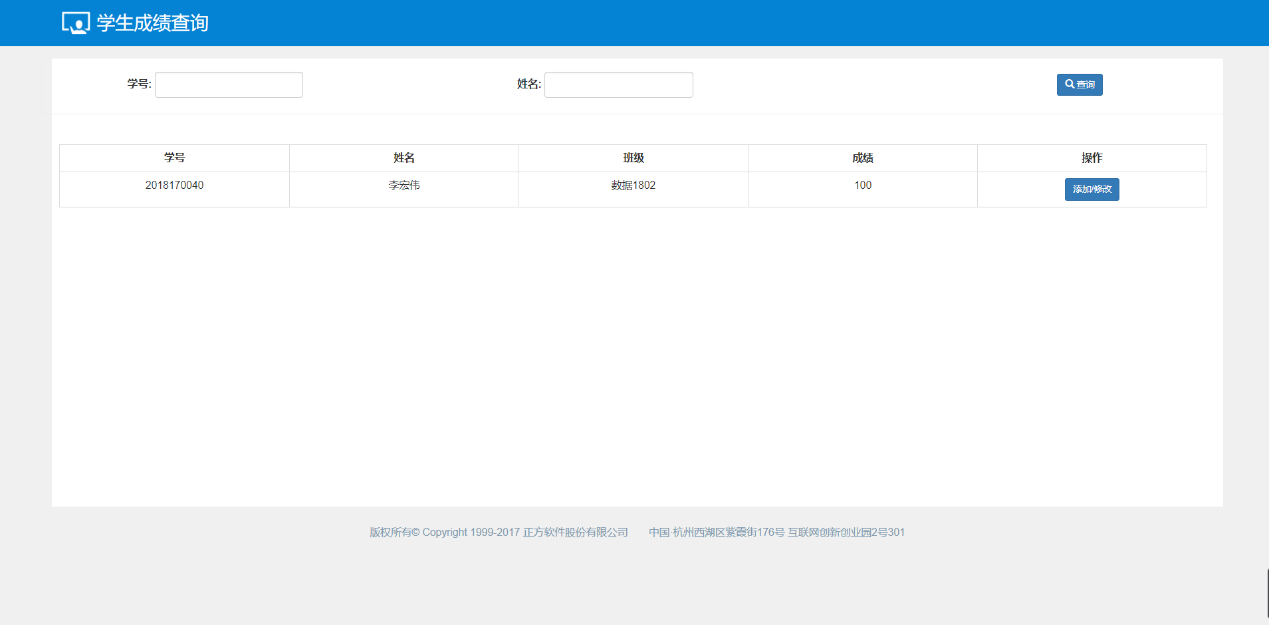


### 个人信息维护



### 课程评分和修改





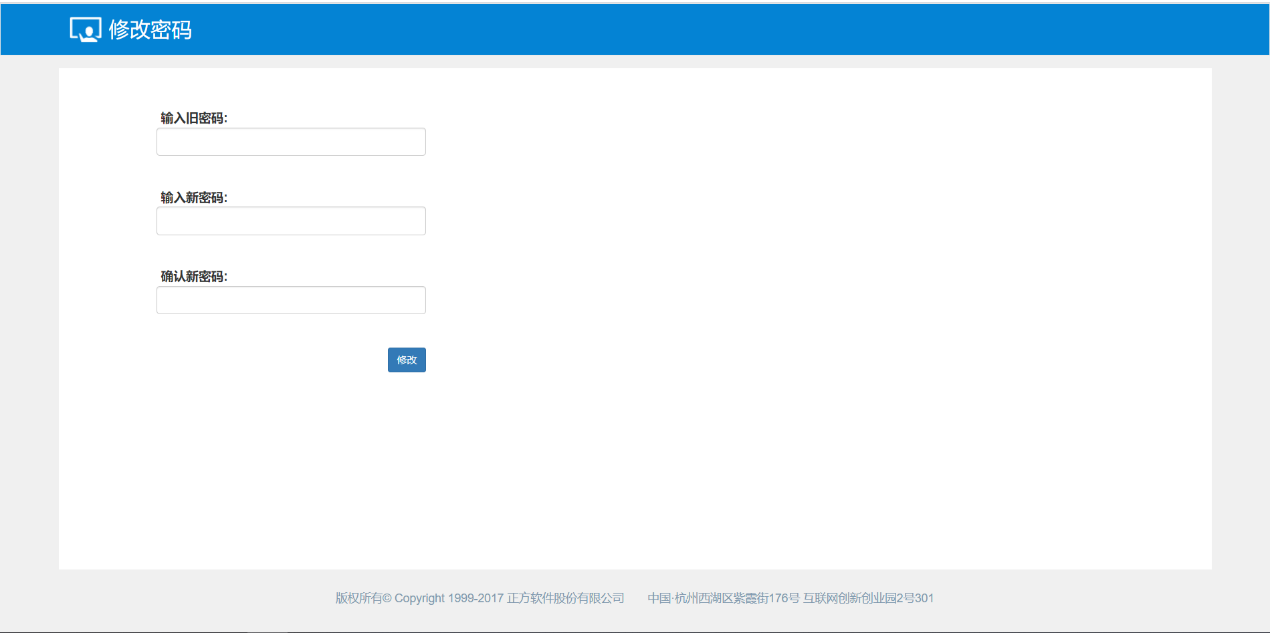
### 查看公告



### 授课查看

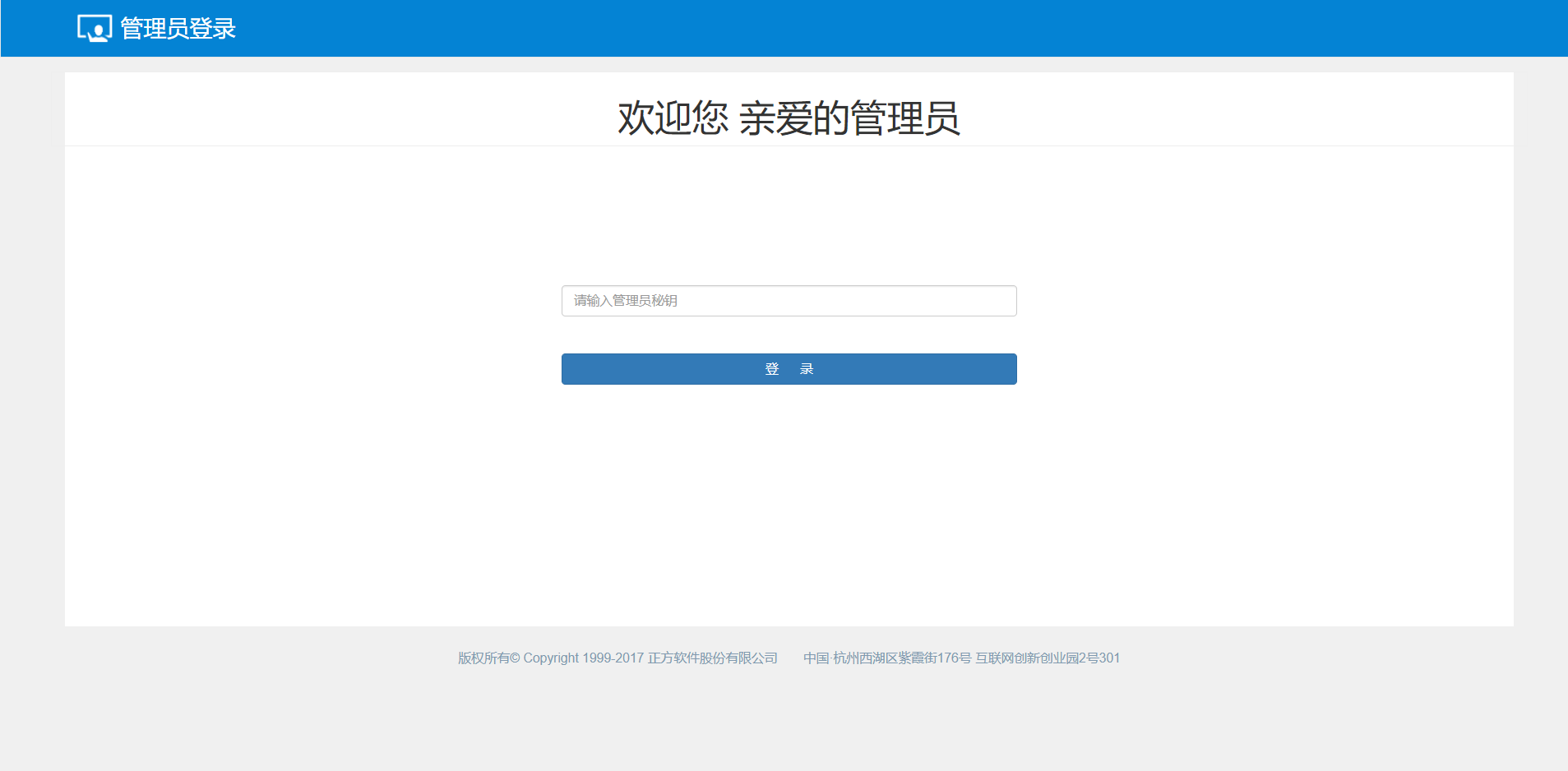


### 修改密码

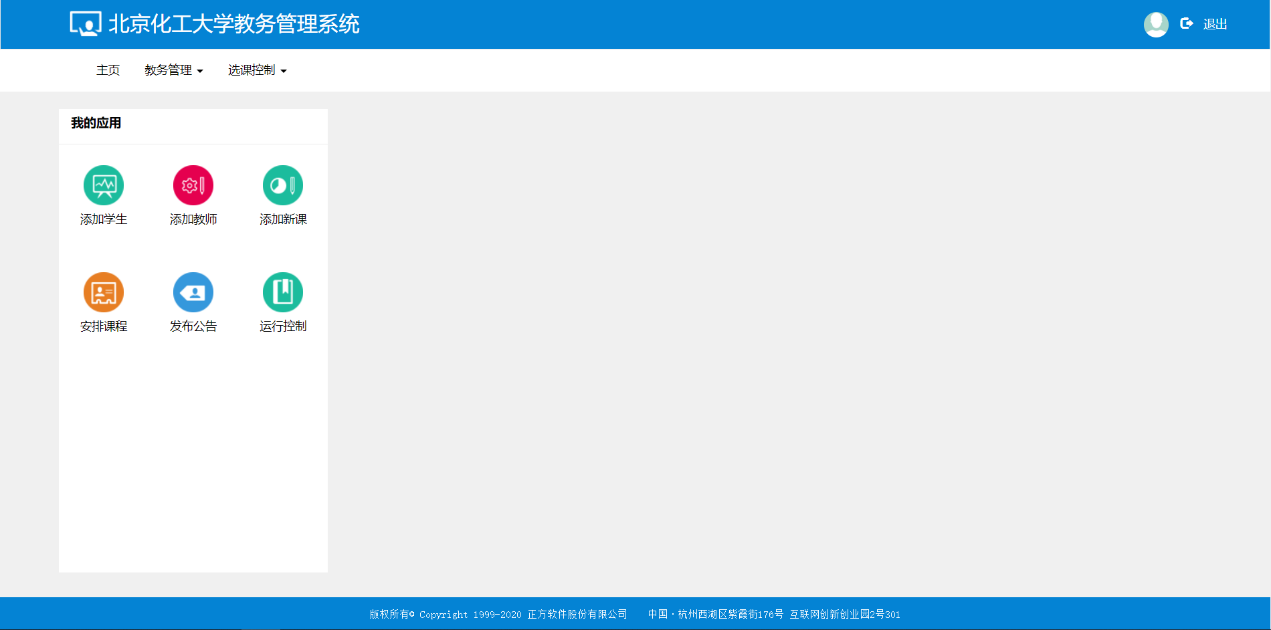


## 管理员相关

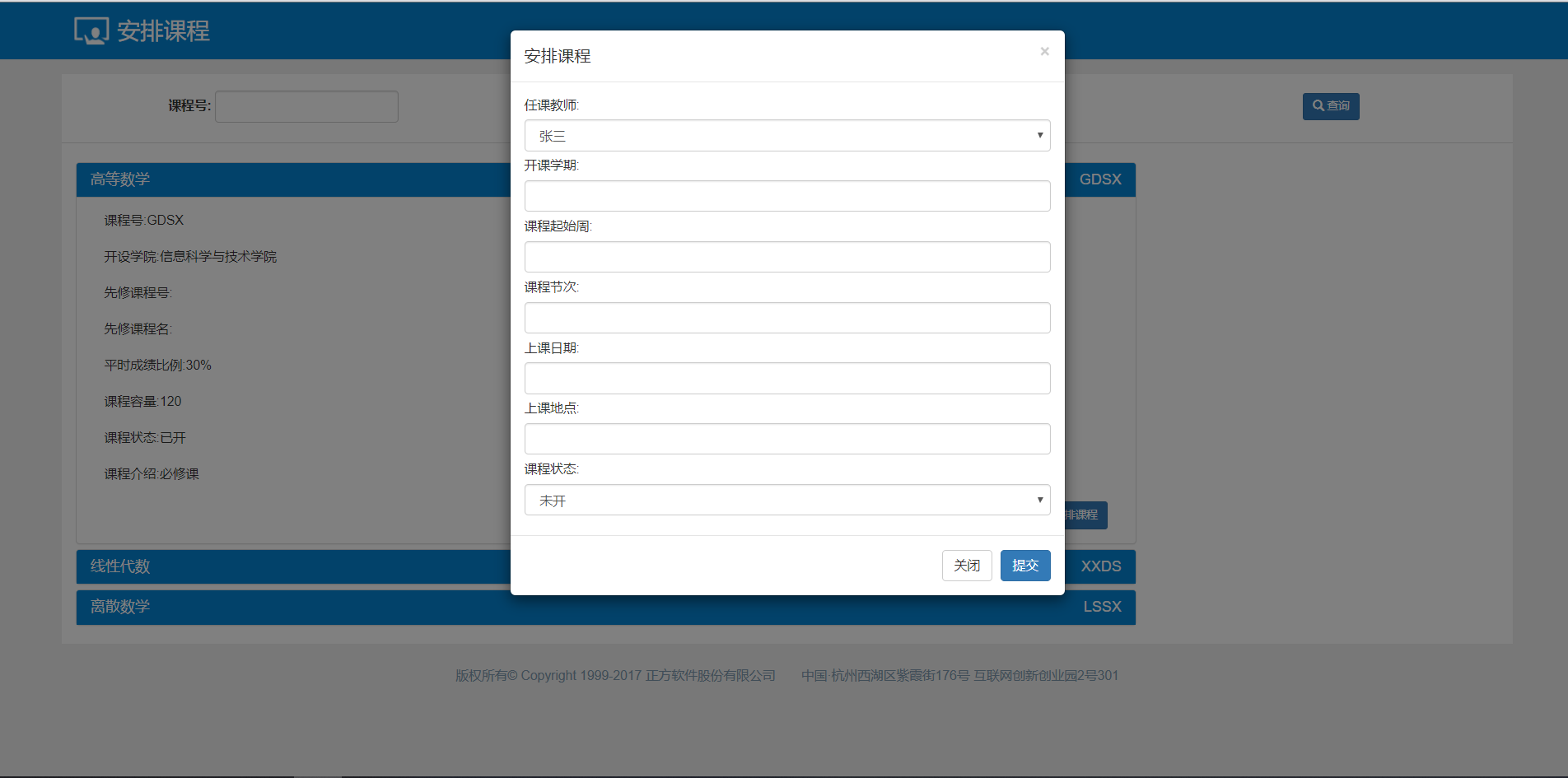
### 管理员登入



### 管理员首页



### 安排课程



### 添加学生



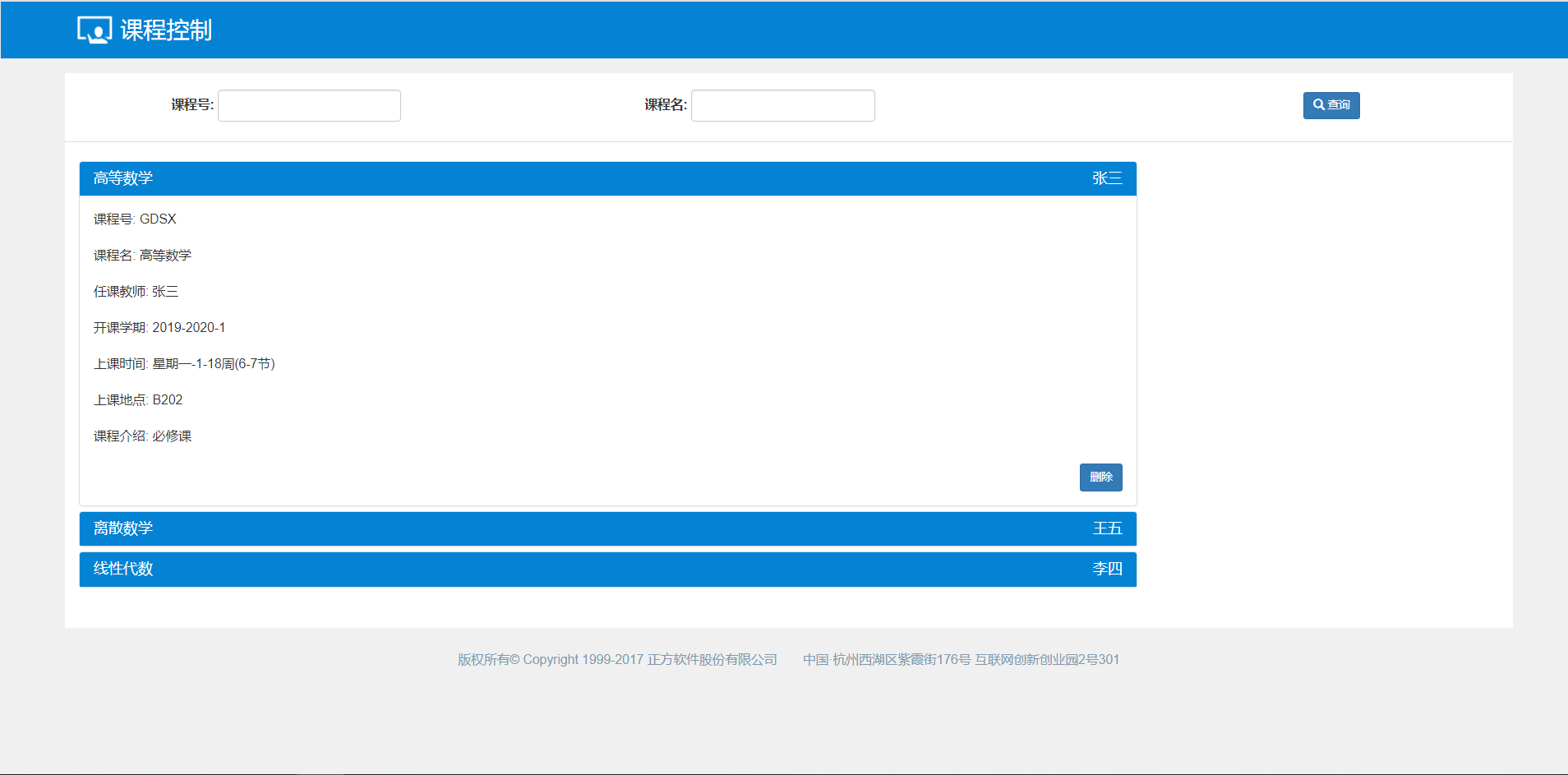
### 添加教师



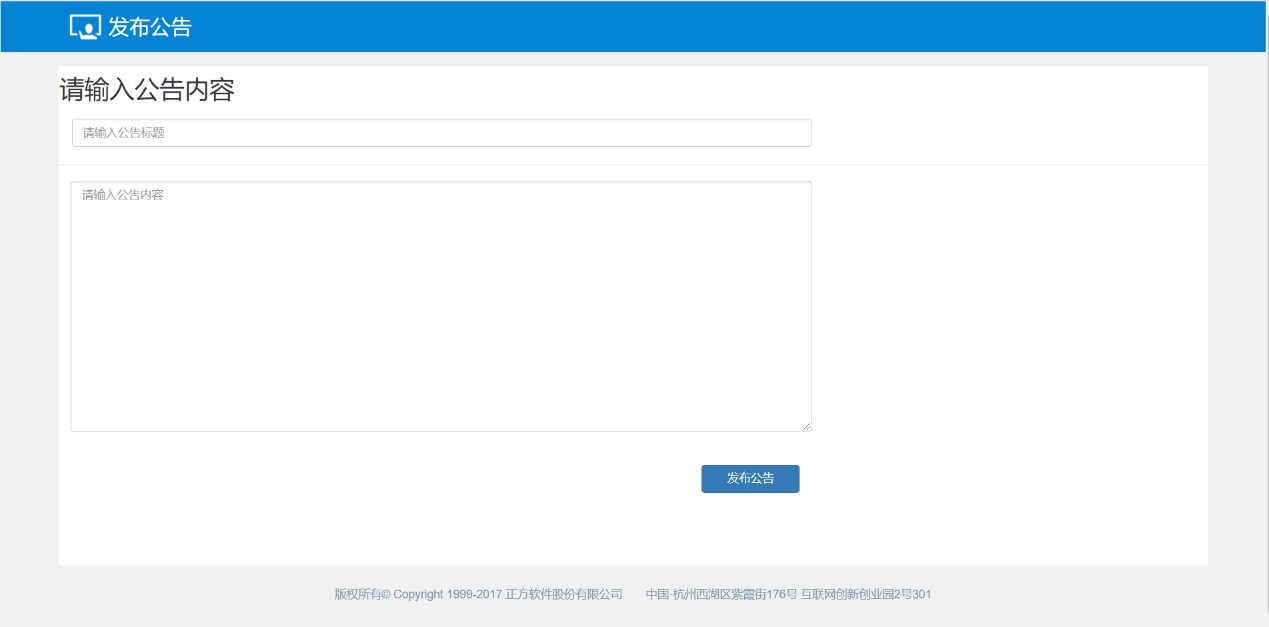
### 添加新课



### 运行控制



### 发布公告



# 第四部分：总结与反思

通过学习数据库原理这门课，我不仅学习了许多之前没有学过的新知识，更加重要的是学会了如何去进行需求分析，如何设计数据库，深刻体会到要想实现一个好的项目，在写代码之前一定要做好设计。