

学生选课系统--

设计报告

姓名： 高谦

班级： 计科1601

学号： 2016014302

指导教师： 尚颖

目录

[第一阶段：开发前的设置和思考 4](#_Toc533965128)

[题目要求： 4](#_Toc533965129)

[需求分析 4](#_Toc533965130)

[概念结构设计 5](#_Toc533965131)

[逻辑结构设计 6](#_Toc533965132)

[实体（红色表示主键） 6](#_Toc533965133)

[联系（红色表示主键） 6](#_Toc533965134)

[数据库物理结构设计 6](#_Toc533965135)

[概览： 6](#_Toc533965136)

[学生表 7](#_Toc533965137)

[教师表 7](#_Toc533965138)

[管理员 7](#_Toc533965139)

[课程表 8](#_Toc533965140)

[选课表 8](#_Toc533965141)

[课程安排表 8](#_Toc533965142)

[教师授课表 8](#_Toc533965143)

[课程时间表 9](#_Toc533965144)

[学期表 9](#_Toc533965145)

[消息表 9](#_Toc533965146)

[公告表 9](#_Toc533965147)

[选课清单表 10](#_Toc533965148)

[用户登录信息表 10](#_Toc533965149)

[第二阶段：开发编码调试工作 10](#_Toc533965150)

[开发环境相关： 10](#_Toc533965151)

[开发所用技术： 10](#_Toc533965152)

[源代码目录简介： 10](#_Toc533965153)

[项目典型代码讲解 12](#_Toc533965154)

[数据库连接工具类：DB 12](#_Toc533965155)

[登录Servlet 14](#_Toc533965156)

[首页初始化Servlet: 18](#_Toc533965157)

[第三部分：系统演示和说明 21](#_Toc533965158)

[用户登录 21](#_Toc533965159)

[学生相关 21](#_Toc533965160)

[学生首页 21](#_Toc533965161)

[修改密码 22](#_Toc533965162)

[选课 22](#_Toc533965163)

[查看课表 24](#_Toc533965164)

[查看个人信息 25](#_Toc533965165)

[联系老师 25](#_Toc533965166)

[成绩查询 26](#_Toc533965167)

[教师相关 26](#_Toc533965168)

[教师首页 26](#_Toc533965169)

[个人信息维护 26](#_Toc533965170)

[修改密码 27](#_Toc533965171)

[申请新课 27](#_Toc533965172)

[课程评分和修改 27](#_Toc533965173)

[查看公告 28](#_Toc533965174)

[查看学生信息 28](#_Toc533965175)

[授课查看 29](#_Toc533965176)

[管理员相关 30](#_Toc533965177)

[管理员首页 30](#_Toc533965178)

[安排课程 30](#_Toc533965179)

[添加人员 31](#_Toc533965180)

[运行控制 31](#_Toc533965181)

[创建选课清单 32](#_Toc533965182)

[新课审核 32](#_Toc533965183)

[学生求助信息 33](#_Toc533965184)

[发布公告 33](#_Toc533965185)

[课程信息 34](#_Toc533965186)

[第四部分：总结与反思 34](#_Toc533965187)

# 第一阶段：开发前的设置和思考

## 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

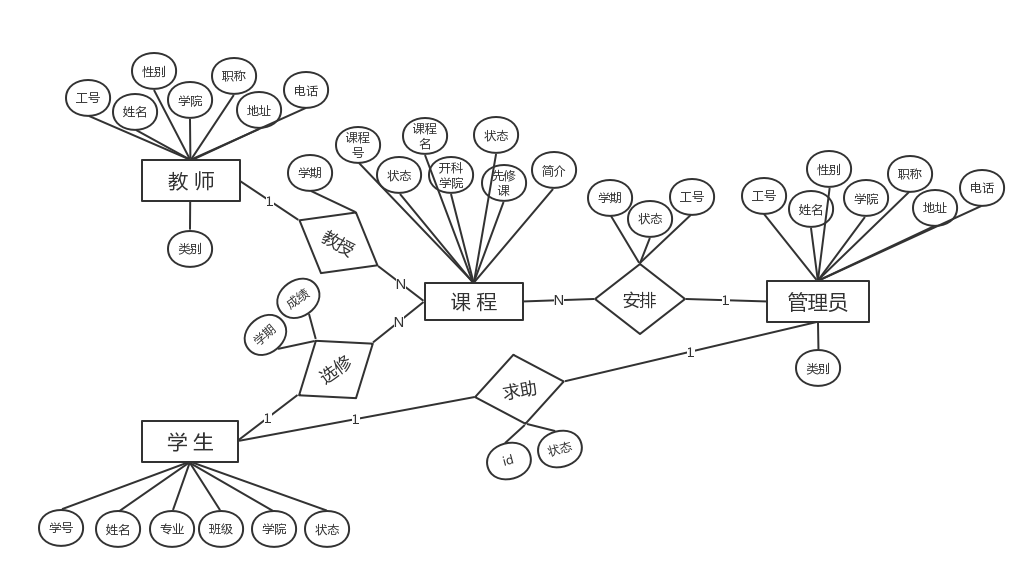
要求：考虑参照完整性以及用户自定义完整性，实现学生选课功能

（4）系统维护

## 需求分析

1. 根据描述，这个系统不仅是选课系统，而是有其他教务管理和信息查询和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. 使用教务管理系统的用户有三种，一种是学生，一种是任课教师，另外一种损失教务处的老师，可以认为是管理员，所以可以将用户也分为这三种，不同的用户登录到这个系统中要有不同的界面，不同的功能。
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，（自己的学号，班级，专业，籍贯，政治面貌，邮箱，联系方式等）选课信息（选了哪些课，课程表查看，每门课上课的时间，地点，任课教师，教学周起始等），成绩信息（查看自己的成绩信息，包括课程名称，分数，哪一个学期选修的这门课，任课教师等）。修改权限包括学生部分个人信息的修改（比如学生的联系电话，qq和微信，家庭住址，政治面貌等），在选课阶段修改自己的所选课程信息（在选课阶段，自由的增加选课和取消选课），修改自己的登录密码等。另外，仿照现有教务管理系统，增加私信老师的功能，使得学生可以向老师提交反馈，老师可以根据反馈进行相应的操作，学生可以不用打电话或者到教务处办公室寻求帮助。
4. 任课教师登录后功能和学生不同，任课教师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（教务相关等基本信息，其他私人信息不会看到，另外，只有选修了自己所教授课程的学生的信息老师才可以查看，其他学生的不可以），查看自己的个人信息，和学生一样。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，在一个时间范围内给选修了自己课程的学生进行评分以及修改分数等操作，另外，任课教师还可以申请新开一门课程，在得到教务处的允许后方可成为正式的课程。
5. 教务处教师登录后的功能比较多，可以审核通过任课教师提出的新课的审核，可以发布系统公告，查看学生的求助并作出相应的处理，可以进行课程的安排，可以创建每个专业的选课清单，可以对每个选课清单进行控制（停止选课和开始选课等），可以添加新的教职工信息和学生信息。
6. 所有的信息查看功能都有搜索功能，可以在信息比较多的情况下根据用户输入的学号，工号，专业，以及状态等进行结果的缩减，达到快速检索到信息的目的，支持模糊搜索。

## 概念结构设计

**系统的ER图如下：**

## 逻辑结构设计

### 实体（红色表示主键）

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，地址，电话，类别等）

管理员（工号，姓名，性别，学院，职称，地址，电话，类别等）

课程（课程号，课程名称，开课学院，先修课程号，开课学院，简介，状态）

### 联系（红色表示主键）

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）

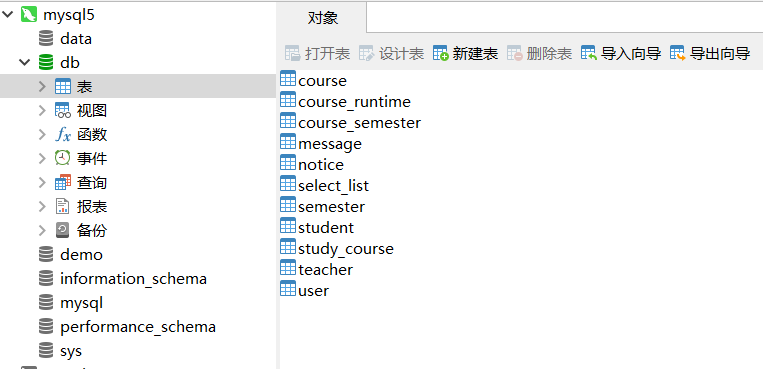
求助（消息id，学号，发送时间，标题，内容，状态）

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）

## 数据库物理结构设计

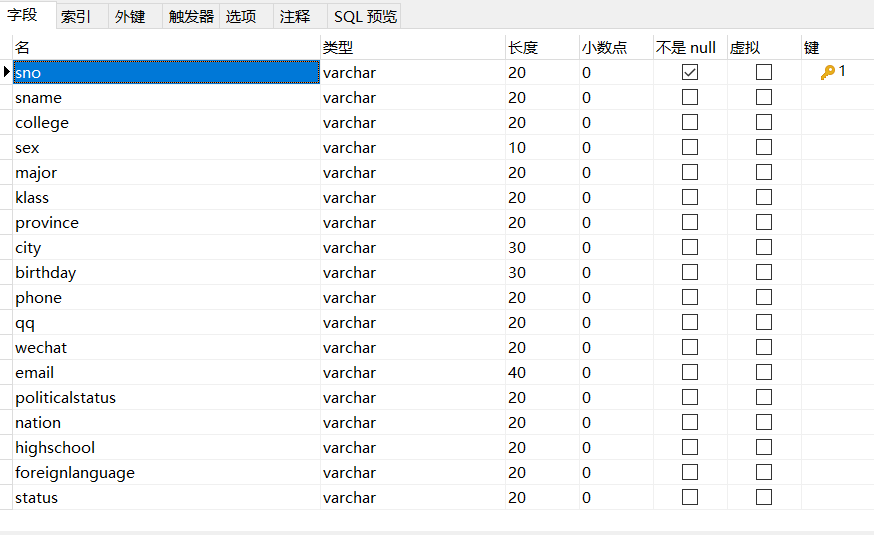
**数据表信息如下：**

### 概览：

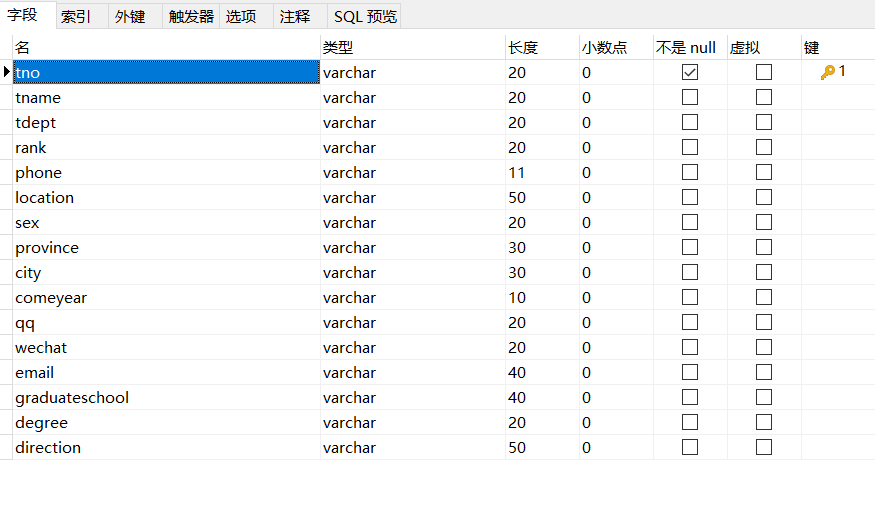


数据库名称为DB，数据表数量为11个

### 学生表



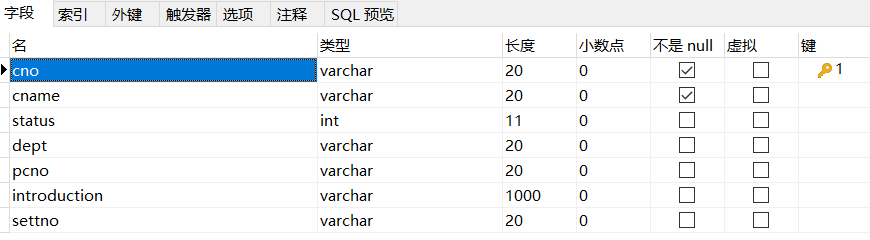
### 教师表



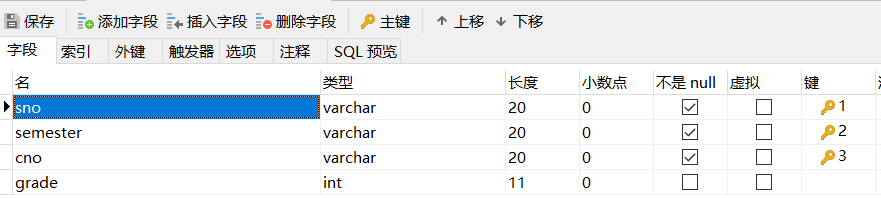
### 管理员

（根据逻辑结构设计部分发现，教师和管理员其实可以合并）

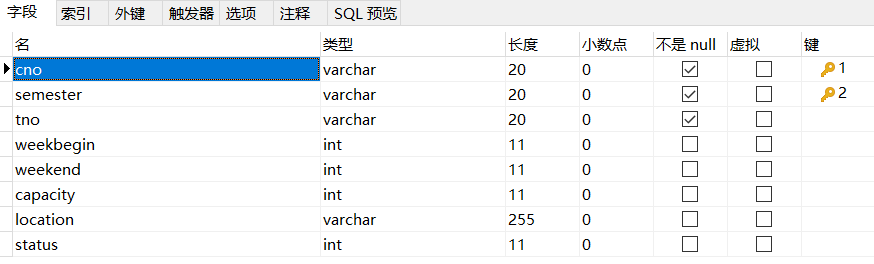
### 课程表



### 选课表



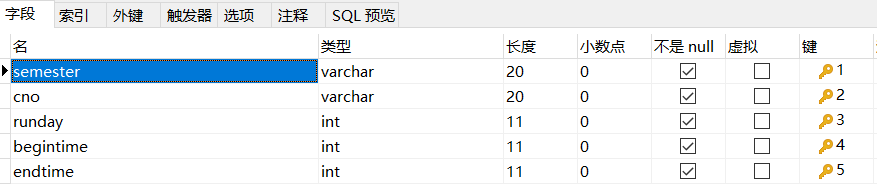
### 课程安排表



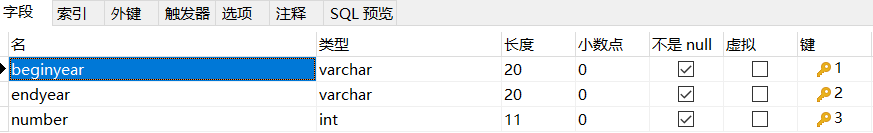
### 教师授课表

（研究发现，可以和安排表合并）

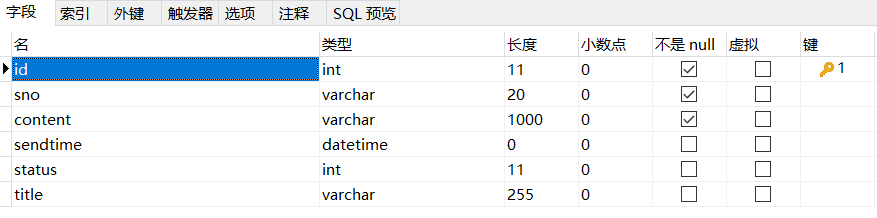
### 课程时间表



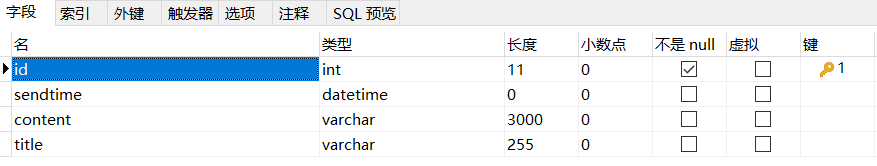
### 学期表



### 消息表



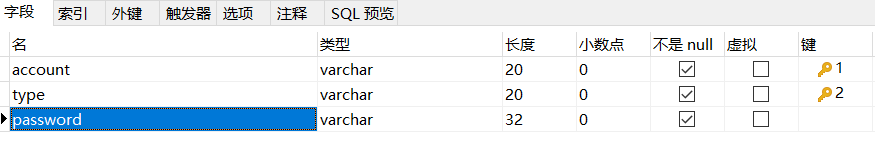
### 公告表



### 选课清单表



### 用户登录信息表



# 第二阶段：开发编码调试工作

## 开发环境相关：

开发后台服务语言：Java 版本1.8

服务器： Tomcat 版本 9.0.14

数据库：MySql 5.7

操作系统：Windows10

开发工具：Jetbrains Intelij IDEA

## 开发所用技术：

Web前端：HTML，CSS，JavaScript，Jquery，Bootstrap。

Web后端：原生Java Web，Servlet，JDBC，JSTL，数据库连接池DHCP，JSP

### 源代码目录简介：

.idea：IDEA编译器的配置文件，系统默认生成，无序关心

lib和libs：自己下载的驱动包和第三方jar包，包括数据库驱动和jstl

out：字节码文件，为Java虚拟机实际运行的文件，编译之后生成，无需关心。

sql：数据库备份文件，完整的数据表结构和部分数据。

src：Java代码编写处

DB：存放有关数据库操作的类

model：存放数据表结构相关的Java bean类

servlet：存放业务逻辑处理的servlet类

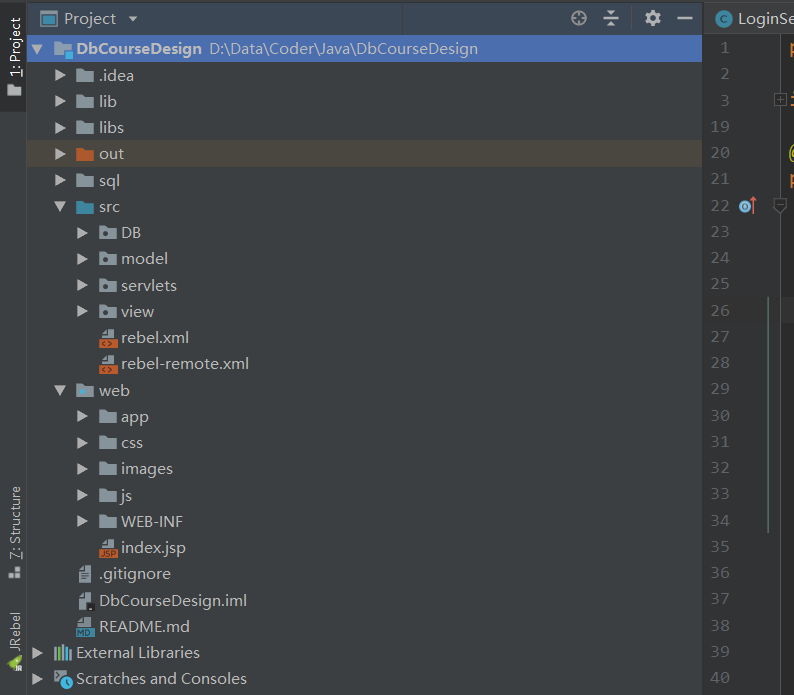
web：存放静态资源和网页模板的类：

app：放所有的jsp网页模板

css：放所有的层叠样式表文件

js：放所有的JavaScript脚本文件

WEB-INF：系统自动生成 ，无需关心，是网站的配置信息文件。



## 项目典型代码讲解

### 数据库连接工具类：DB

package DB;

import java.sql.\*;

public class DB {

//初始化连接参数;

public static String url="jdbc:mysql://localhost:3306/db?serverTimezone=UTC&useSSL=false&characterEncoding=utf8";

public static String user="root";

public static String password="root";

//初始化驱动加载：

static{

try {

Class.forName("com.mysql.cj.jdbc.Driver");

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

}

public static Connection getConnection()

{

try {

Connection connection=DriverManager.getConnection(url,user,password);

return connection;

} catch (SQLException e) {

e.printStackTrace();

}

return null;

}

public static void close(Connection connection,Statement st,ResultSet rs)

{

try {

connection.close();

st.close();

rs.close();

}

catch(Exception e) {

e.printStackTrace();

}

}

public static void close(Connection connection,PreparedStatement st)

{

try {

connection.close();

st.close();

}

catch(Exception e)

{

e.printStackTrace();

}

}

public static void close(Statement st)

{

try {

st.close();

}

catch(Exception e)

{

e.printStackTrace();

}

}

public static void close(Connection connection)

{

try {

connection.close();

}

catch(Exception e)

{

e.printStackTrace();

}

}

public static void close(ResultSet resultSet)

{

try {

resultSet.close();

}

catch(Exception e)

{

e.printStackTrace();

}

}

}

以上类用于数据库的连接，获取连接对象，然后还有关闭连接的函数，对关闭连接的函数进行了重载，可以关闭连接对象，语句对象，结果集对象的任何一个子集组合，完善的异常处理机制，可以简化其他用到数据库连接的地方，减少代码量。

### 登录Servlet

package servlets.publicfun;

import DB.SemesterDB;

import DB.StudentDB;

import DB.TeacherDB;

import DB.UserDB;

import model.Student;

import model.Teacher;

import model.User;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.SQLException;

@WebServlet("/LoginServlet")

public class LoginServlet extends HttpServlet {

protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

String account=request.getParameter("account");

String password=request.getParameter("password");

String type=request.getParameter("type");

response.setCharacterEncoding("UTF-8");

response.setContentType("text/html");

if(account==""||password=="")

{

PrintWriter printWriter=response.getWriter();

printWriter.println("您的用户名或者密码为空，请重新登陆");

printWriter.flush();

printWriter.close();

}

else

{

UserDB userDB=new UserDB();

User user=userDB.login(account,password,type);

if(user==null) {

PrintWriter printWriter=response.getWriter();

printWriter.print("<script type='text/javascript'>alert('用户名或者密码错误！');window.location.href='"+request.getContextPath()+"/index.jsp'</script>");

printWriter.flush();

printWriter.close();

}

if(type.equals("student".toString()))

{

//查询学生表详细信息，并设置session

StudentDB studentDB=new StudentDB();

Student student=studentDB.getInfo(user.getAccount());

if(student==null)

{

PrintWriter printWriter=response.getWriter();

printWriter.print("<script type='text/javascript'>alert('没有您的登录信息，请联系管理员！');window.location.href='"+request.getContextPath()+"/index.jsp'</script>");

printWriter.flush();

printWriter.close();

}

else {

SemesterDB semesterDB=new SemesterDB();

String semester=null;

try {

semester=semesterDB.getCurrentSemester();

} catch (SQLException e) {

e.printStackTrace();

}

if(semester==null)

{

PrintWriter printWriter=response.getWriter();

printWriter.print("<script type='text/javascript'>alert('学期列表出现错误，请联系管理员！');window.location.href='"+request.getContextPath()+"/index.jsp'</script>");

printWriter.flush();

printWriter.close();

response.setStatus(500);

}

HttpSession session = request.getSession();

session.setAttribute("userinfo", student);

session.setAttribute("type","student");

session.setAttribute("semester",semester);

//重定向到另外一个servlet

response.setStatus(302);

response.sendRedirect("content");

}

}

else if(type.equals("teacher"))

{

//查询教师表详细信息，并设置session

TeacherDB teacherDB=new TeacherDB();

Teacher teacher=teacherDB.getInfo(user.getAccount());

if(teacher==null)

{

PrintWriter printWriter=response.getWriter();

printWriter.print("<script type='text/javascript'>alert('没有您的登录信息，请联系管理员！');window.location.href='"+request.getContextPath()+"/index.jsp'</script>");

printWriter.flush();

printWriter.close();

}

else {

SemesterDB semesterDB=new SemesterDB();

String semester=null;

try {

semester=semesterDB.getCurrentSemester();

} catch (SQLException e) {

e.printStackTrace();

}

if(semester==null)

{

PrintWriter printWriter=response.getWriter();

printWriter.print("<script type='text/javascript'>alert('学期列表出现错误，请联系管理员！');window.location.href='"+request.getContextPath()+"/index.jsp'</script>");

printWriter.flush();

printWriter.close();

response.setStatus(500);

}

HttpSession session = request.getSession();

session.setAttribute("userinfo", teacher);

session.setAttribute("type","teacher");

session.setAttribute("semester",semester);

response.setStatus(302);

response.sendRedirect("content");

}

}

else if(type.equals("root")) {

TeacherDB teacherDB = new TeacherDB();

Teacher teacher = teacherDB.getInfo(user.getAccount());

if (teacher == null) {

PrintWriter printWriter = response.getWriter();

printWriter.print("<script type='text/javascript'>alert('没有您的登录信息，请联系管理员！');window.location.href='" + request.getContextPath() + "/index.jsp'</script>");

printWriter.flush();

printWriter.close();

} else {

//设置管理员身份标识

teacher.setType("root");

//设置管理员登录标识

SemesterDB semesterDB=new SemesterDB();

String semester=null;

try {

semester=semesterDB.getCurrentSemester();

} catch (SQLException e) {

e.printStackTrace();

}

if(semester==null)

{

PrintWriter printWriter=response.getWriter();

printWriter.print("<script type='text/javascript'>alert('学期列表出现错误，请联系管理员！');window.location.href='"+request.getContextPath()+"/index.jsp'</script>");

printWriter.flush();

printWriter.close();

response.setStatus(500);

}

HttpSession session = request.getSession();

session.setAttribute("userinfo", teacher);

session.setAttribute("type","root");

session.setAttribute("semester",semester);

//重定向到另外一个servlet

response.setStatus(302);

response.sendRedirect("content");

}

}

}

}

}

这里登录只是验证一下是不是用户名和密码都正确，同时保存用户登录的信息到session中，设置登录状态的标志，以后的所有界面如果没有session的话都会自动跳转到登录界面，保证了只有是系统已经登录成功的用户才可以进行相关的查询和修改操作

### 首页初始化Servlet:

因为所有首页登录进去之后都有一些数据是在数据库中取到的，所以要在加载页面之前从数据库中取出数据加到request域中，让前端可以访问并显示这些数据。因为教师，学生，管理员的首页界面不同，所以需要进行判断身份，然后连接不同的数据库，发送不同的信息到前端。

package servlets;

import DB.\*;

import model.\*;

import view.Course\_select;

import view.Course\_teach;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

@WebServlet("/content")

public class content extends HttpServlet {

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

HttpSession session=request.getSession();

try {

String type = (String) session.getAttribute("type");

if (type.equals("root")) {

MessageDB messageDB = new MessageDB();

List<Message> result = messageDB.getAllUncomplete();

CourseDB courseDB = new CourseDB();

List<Course> courseapp = courseDB.getAllUncompleCourse();

request.setAttribute("helpList", result);

request.setAttribute("newcourse", courseapp);

request.getRequestDispatcher("app/content.jsp").forward(request, response);

} else if (type.equals("student")) {

Student student =(Student) session.getAttribute("userinfo");

String sno=student.getSno();

String semester=(String) session.getAttribute("semester");

if(semester==null) {

response.setStatus(500);

PrintWriter printWriter =response.getWriter();

printWriter.print("<script>alert('学期表出现错误！请联系管理员！');window.location.href='"+request.getContextPath()+"/index.jsp';</script>");

printWriter.flush();

printWriter.close();

}

NoticeDB noticeDB = new NoticeDB();

StudyCourseDB studyCourseDB=new StudyCourseDB();

try {

List<Notice> result = noticeDB.getNotice();

List<Course\_select> course\_table=studyCourseDB.getCourseTableList(sno,semester);

List<StudyCourse> gradelist=studyCourseDB.getGradeList(sno,semester);

request.setAttribute("noticelist", result);

request.setAttribute("course\_table\_list",course\_table);

request.setAttribute("grade\_list",gradelist);

request.getRequestDispatcher("app/content.jsp").forward(request, response);

} catch (SQLException e) {

e.printStackTrace();

}

} else if (type.equals("teacher")) {

String semester=(String) session.getAttribute("semester");

Teacher teacher=(Teacher) session.getAttribute("userinfo");

String tno=teacher.getTno();

NoticeDB noticeDB = new NoticeDB();

Course\_teacheDB db=new Course\_teacheDB();

ArrayList<Course\_teach> teach\_list=db.getTeachList(semester,tno);

List<Notice> result = noticeDB.getNotice();

request.setAttribute("noticelist", result);

request.setAttribute("teachlist",teach\_list);

request.getRequestDispatcher("app/content.jsp").forward(request, response);

}

}

catch (Exception e)

{

PrintWriter printWriter=response.getWriter();

printWriter.println("<script>window.location.href='"+request.getContextPath()+"'</script>");

printWriter.flush();

printWriter.close();

}

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

doPost(request,response);

}

}

因为代码很多，我无法详细讲解，提供源代码地址以供学习：

https://github.com/qianqianjun/DB\_Course\_Design

# 第三部分：系统演示和说明

## 用户登录

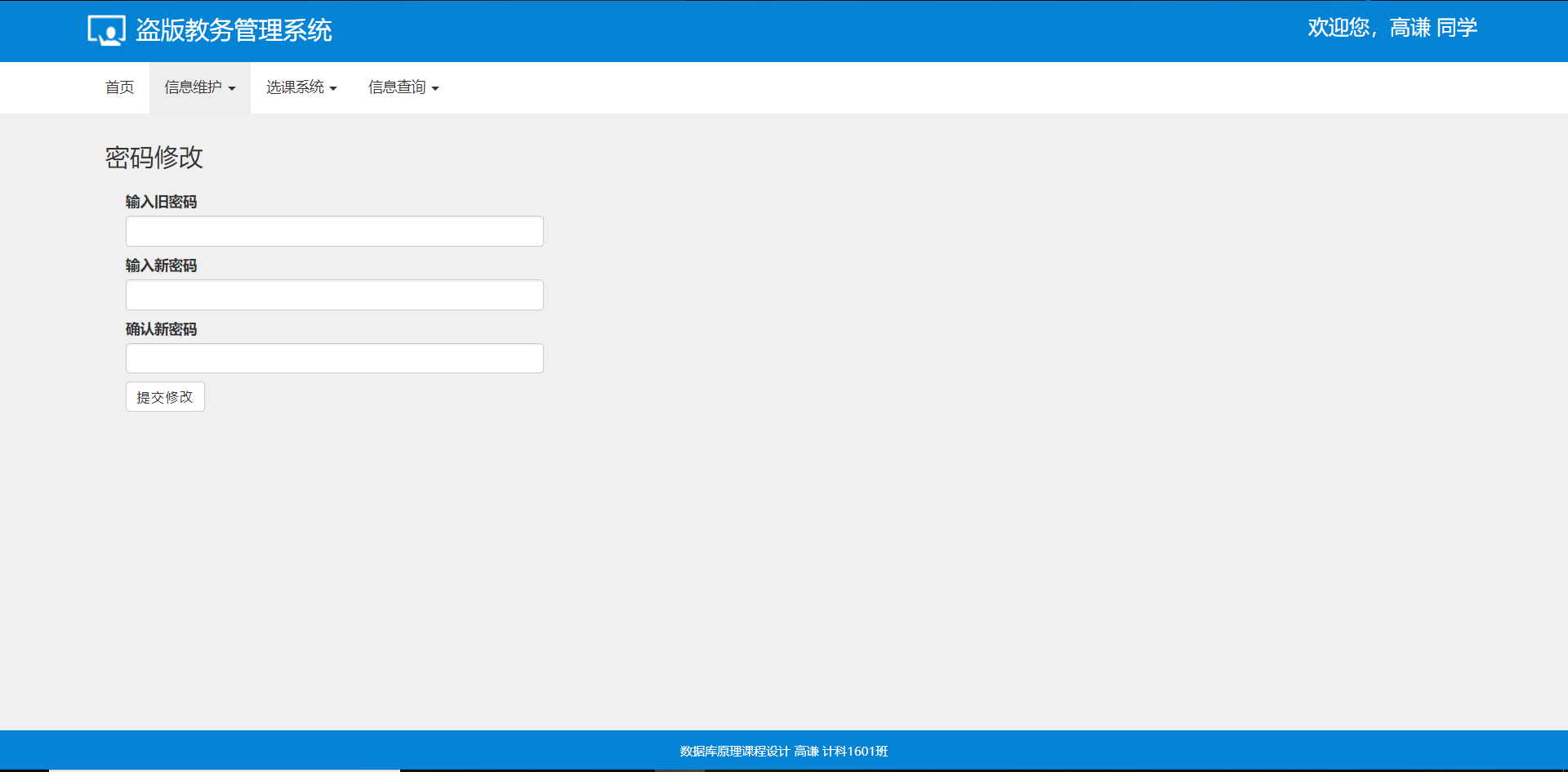


## 学生相关

### 学生首页



### 修改密码

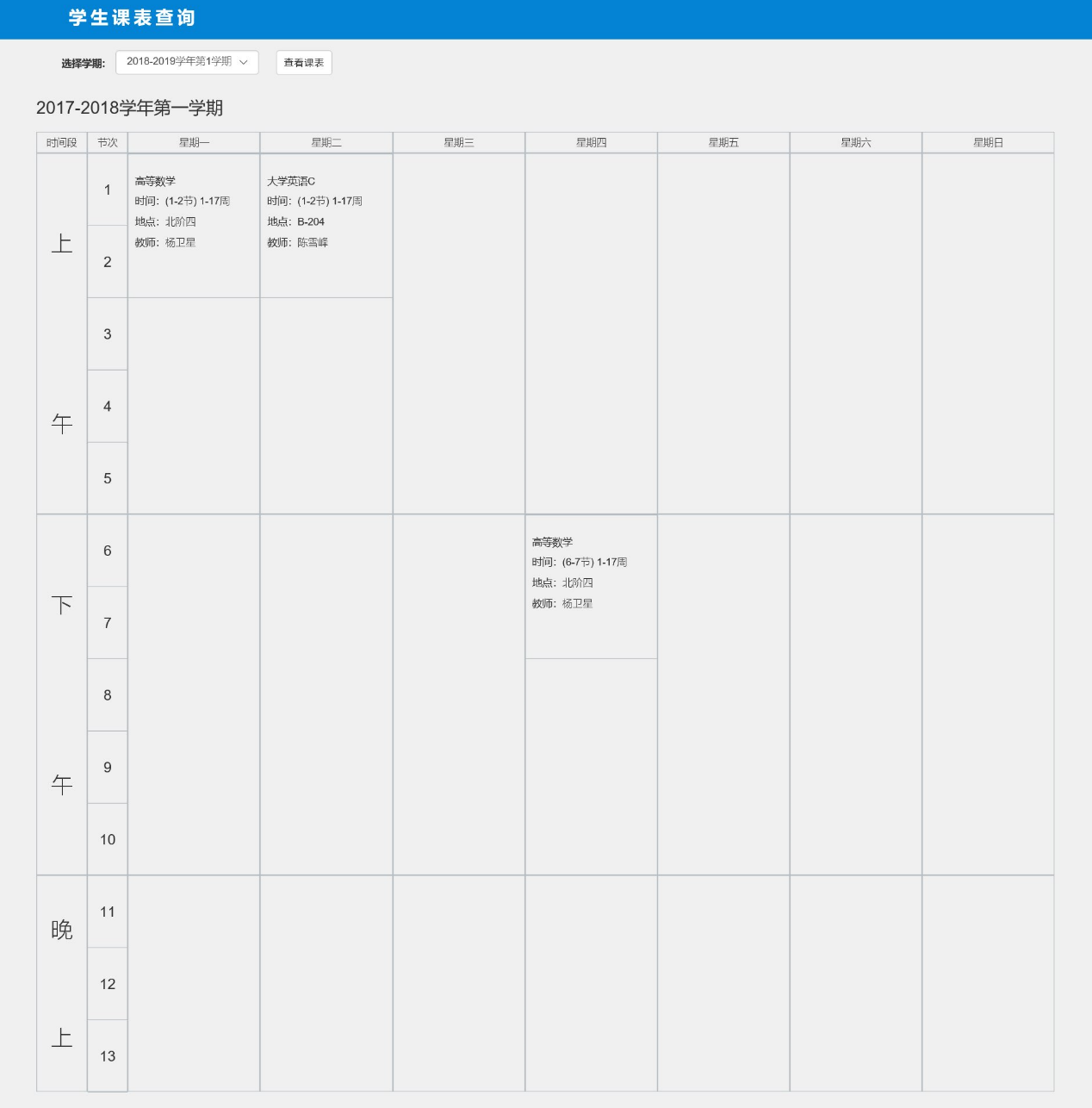


### 选课





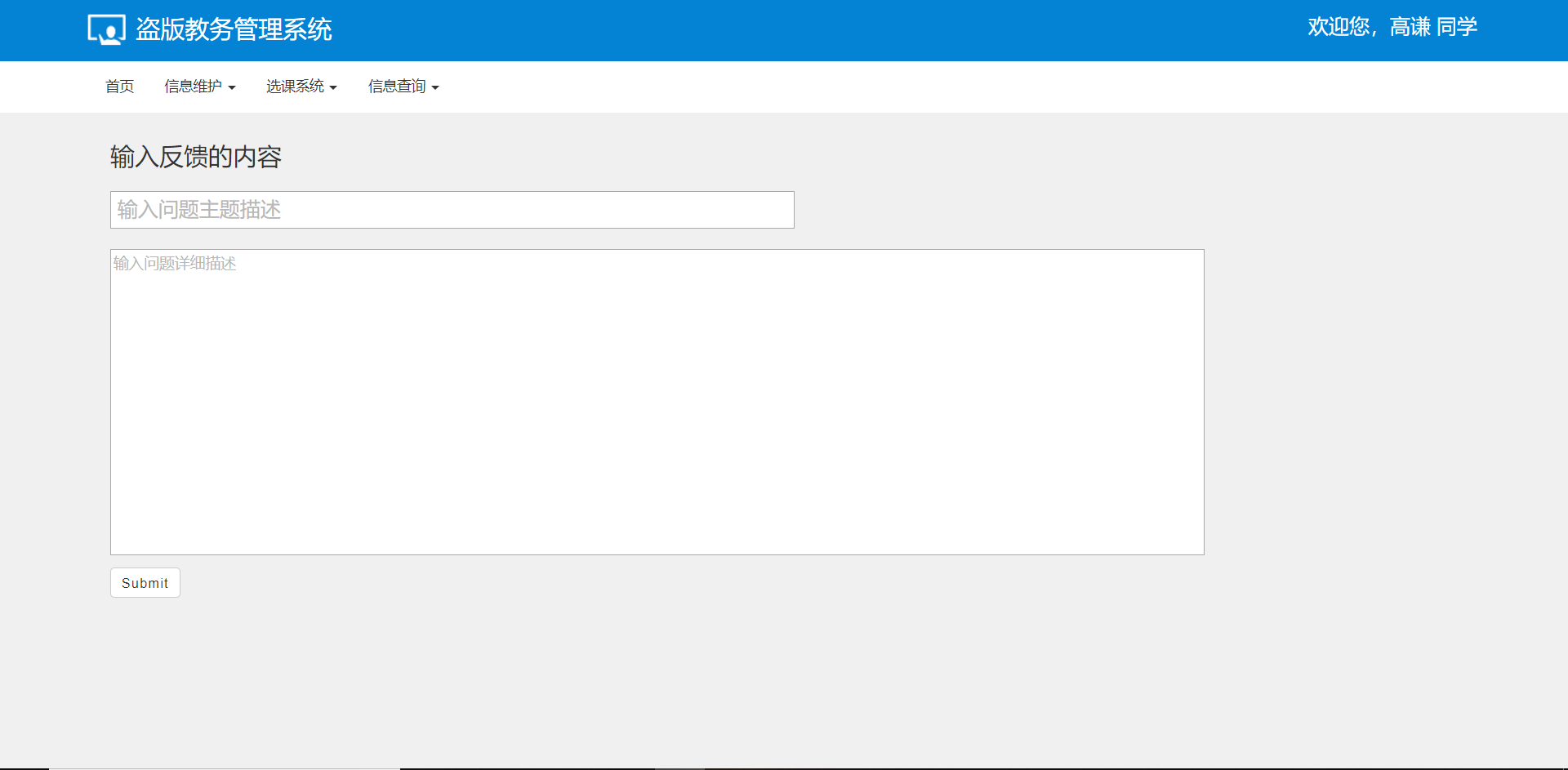
### 查看课表



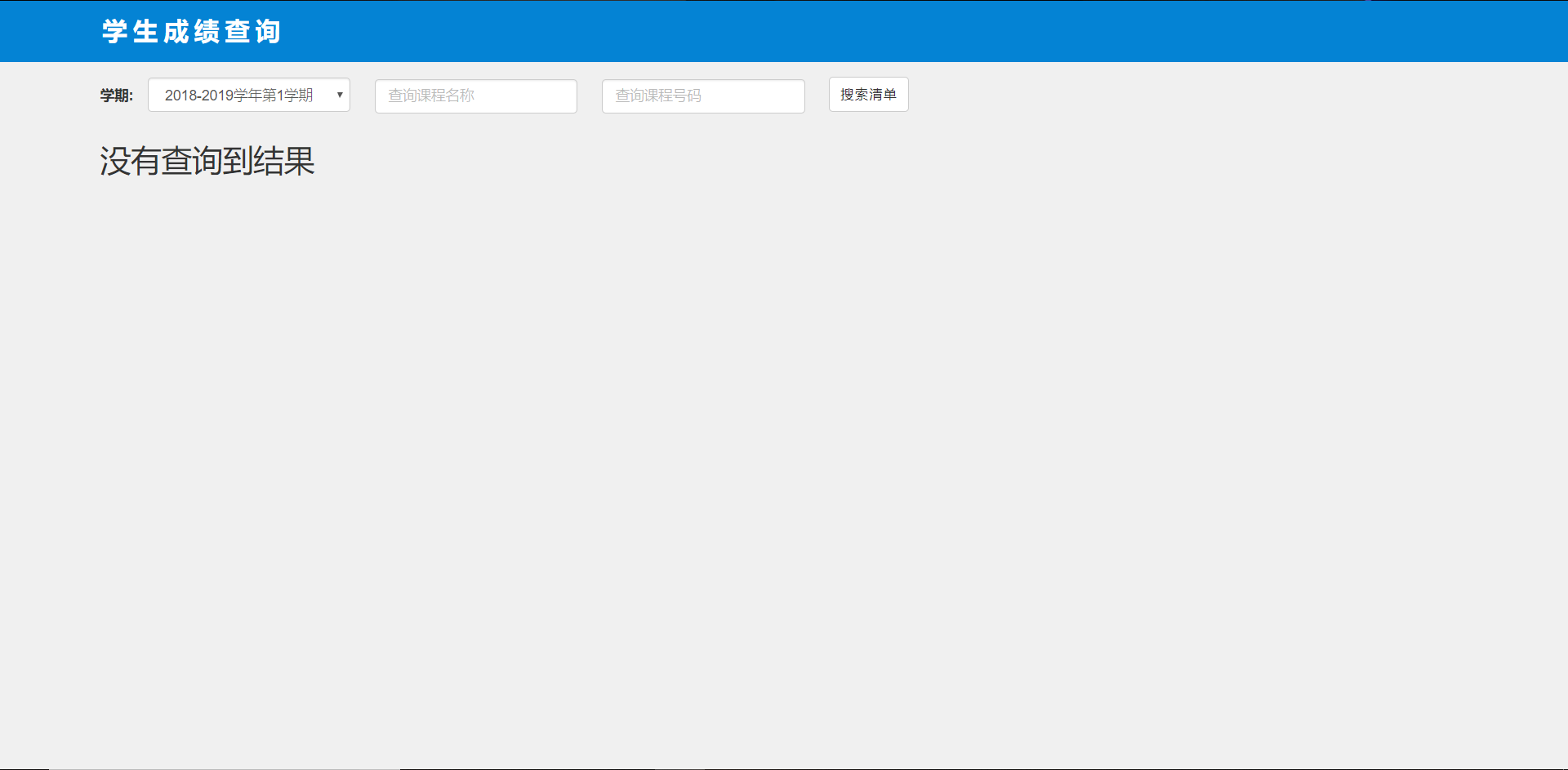
### 查看个人信息



### 联系老师



### 成绩查询



## 教师相关

### 教师首页



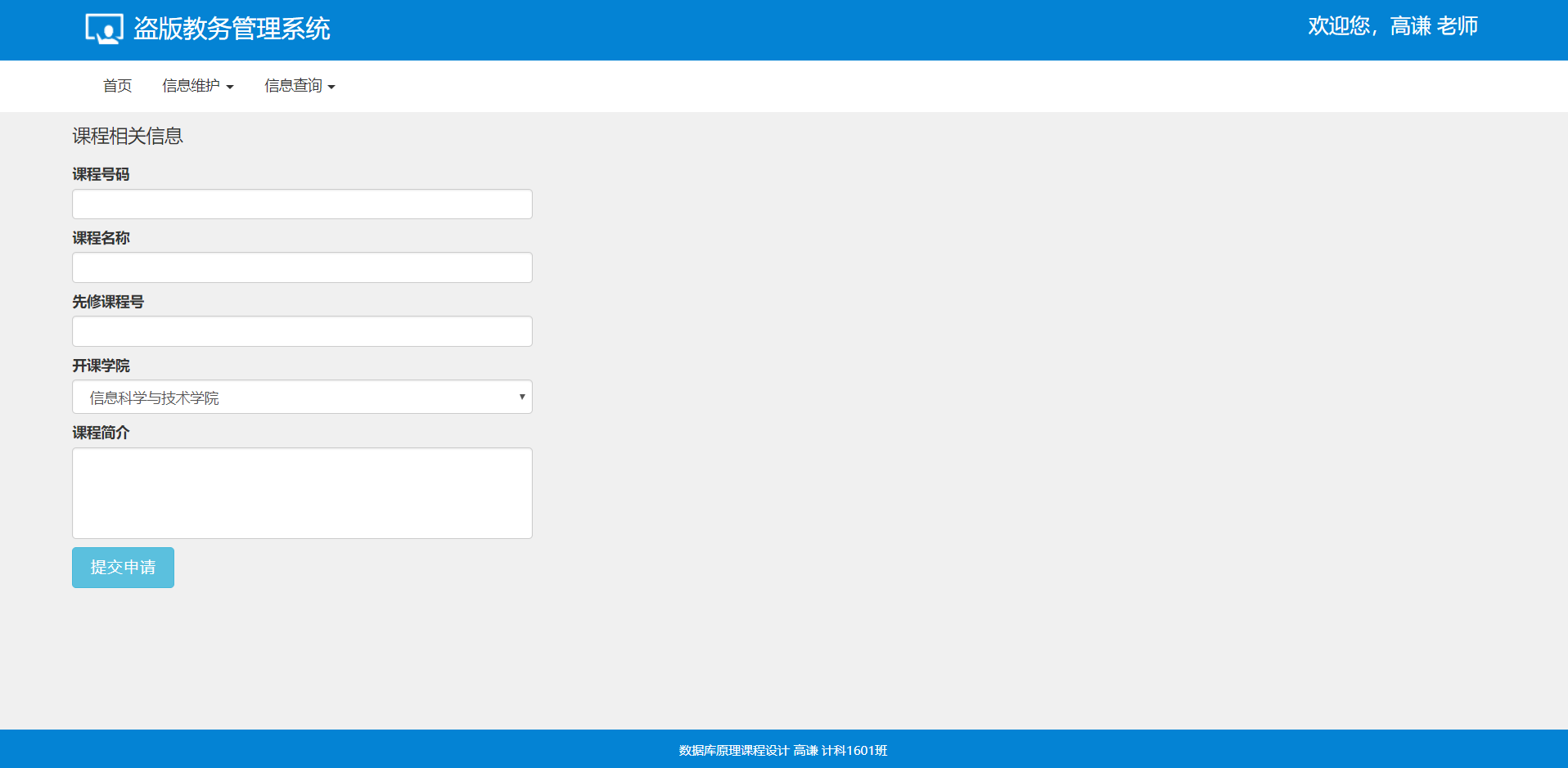
### 个人信息维护

（同学生版页面）

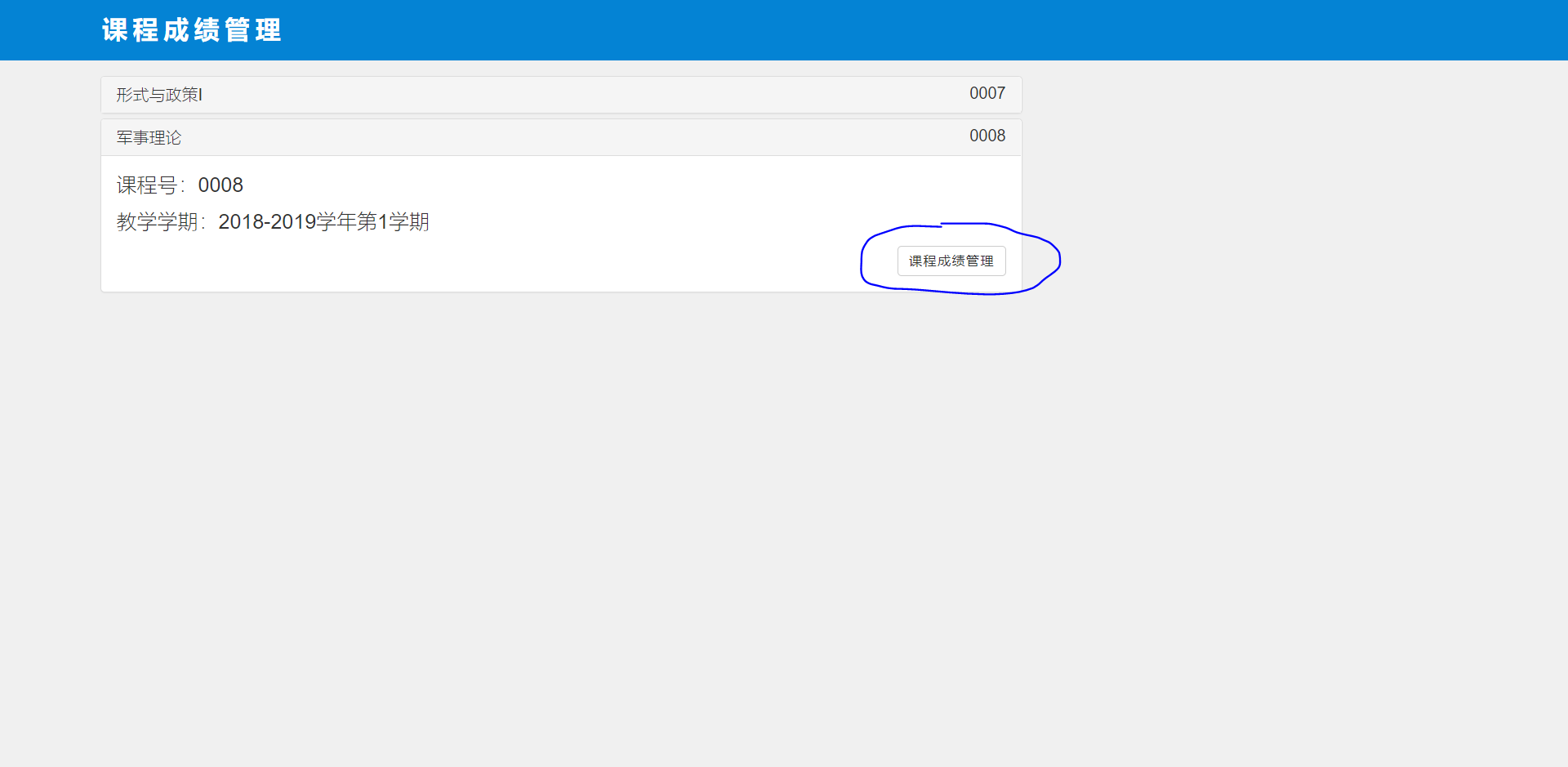
### 修改密码

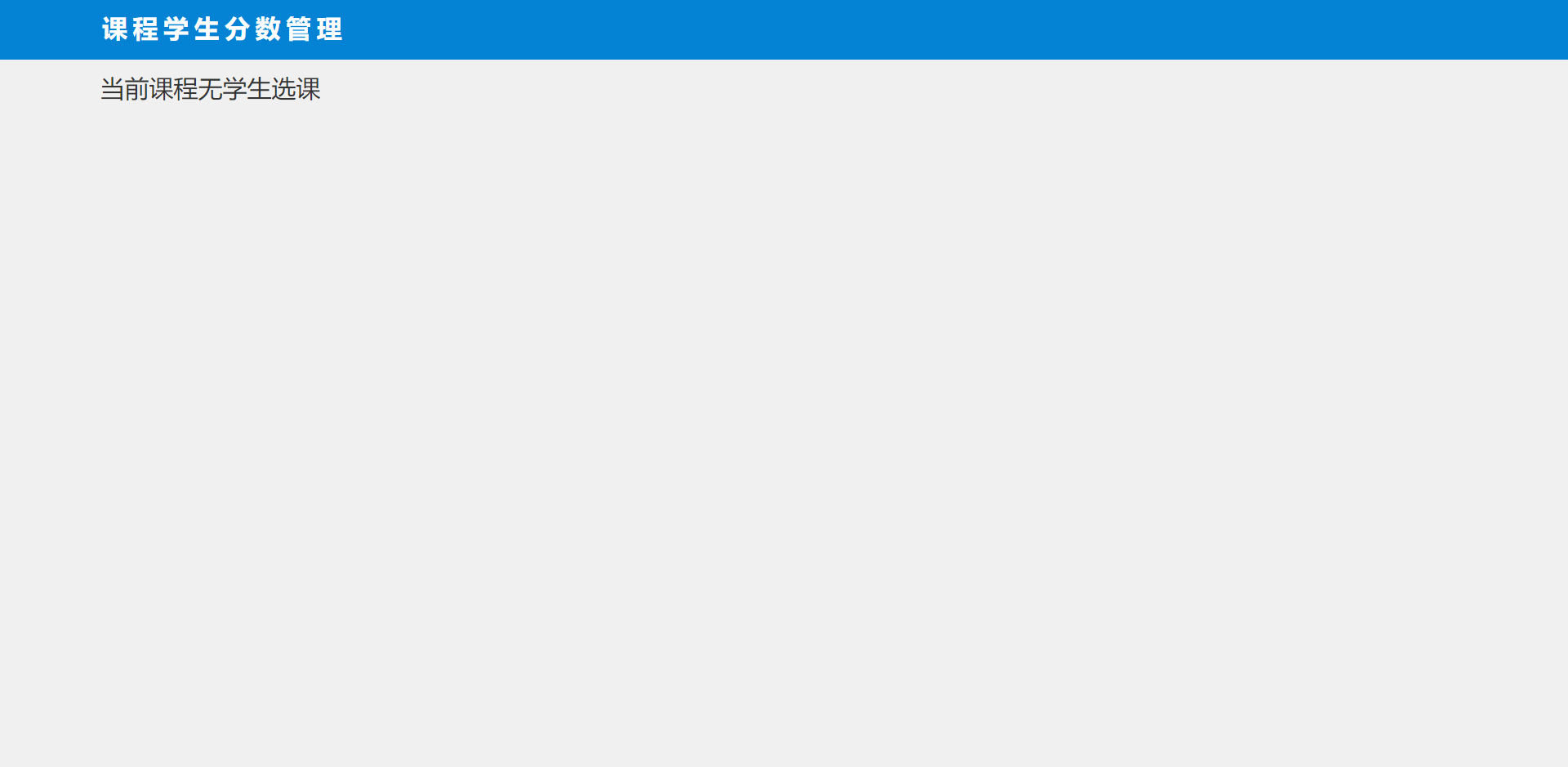
（同学生版页面）

### 申请新课



### 课程评分和修改





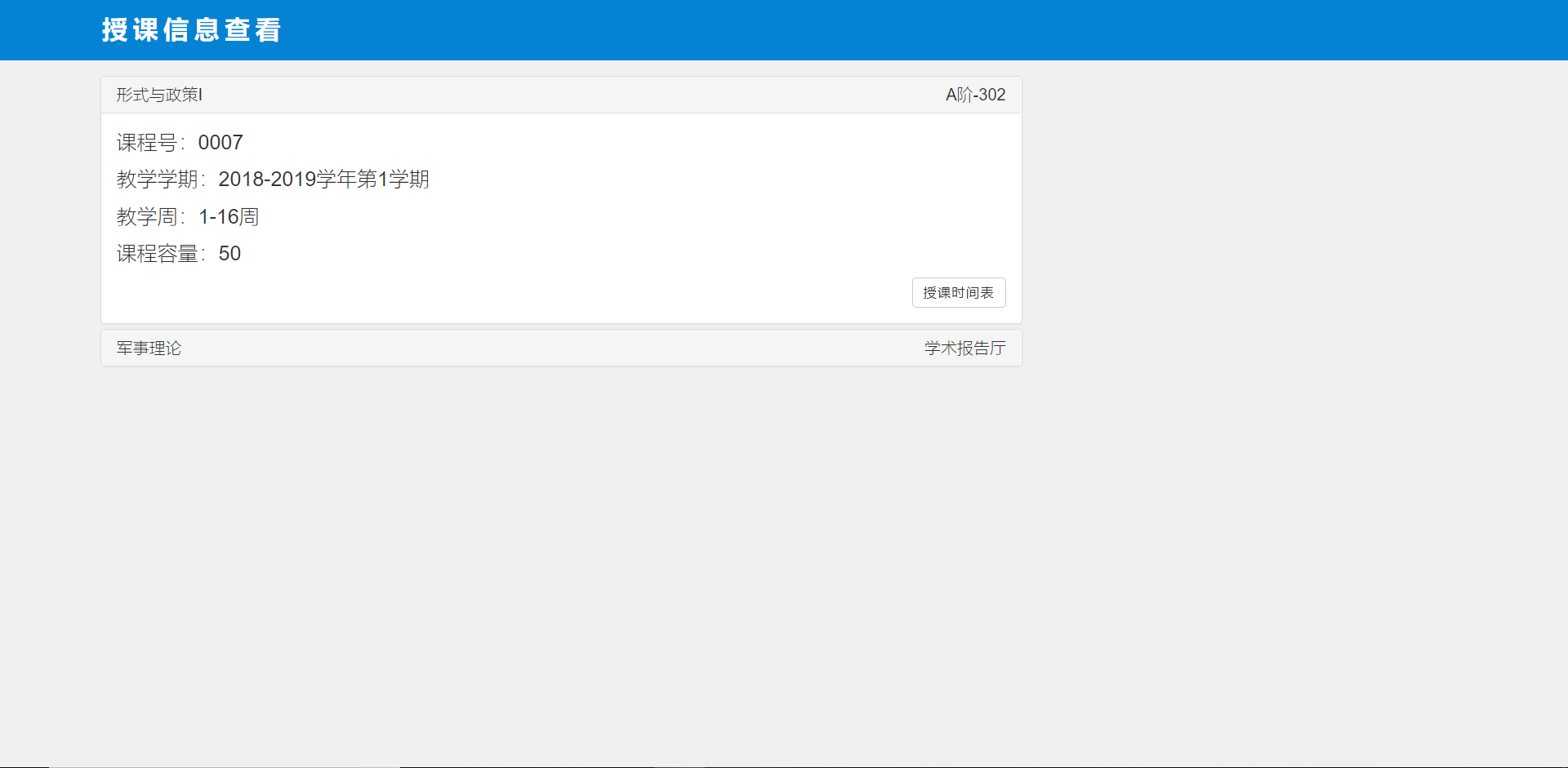
### 查看公告

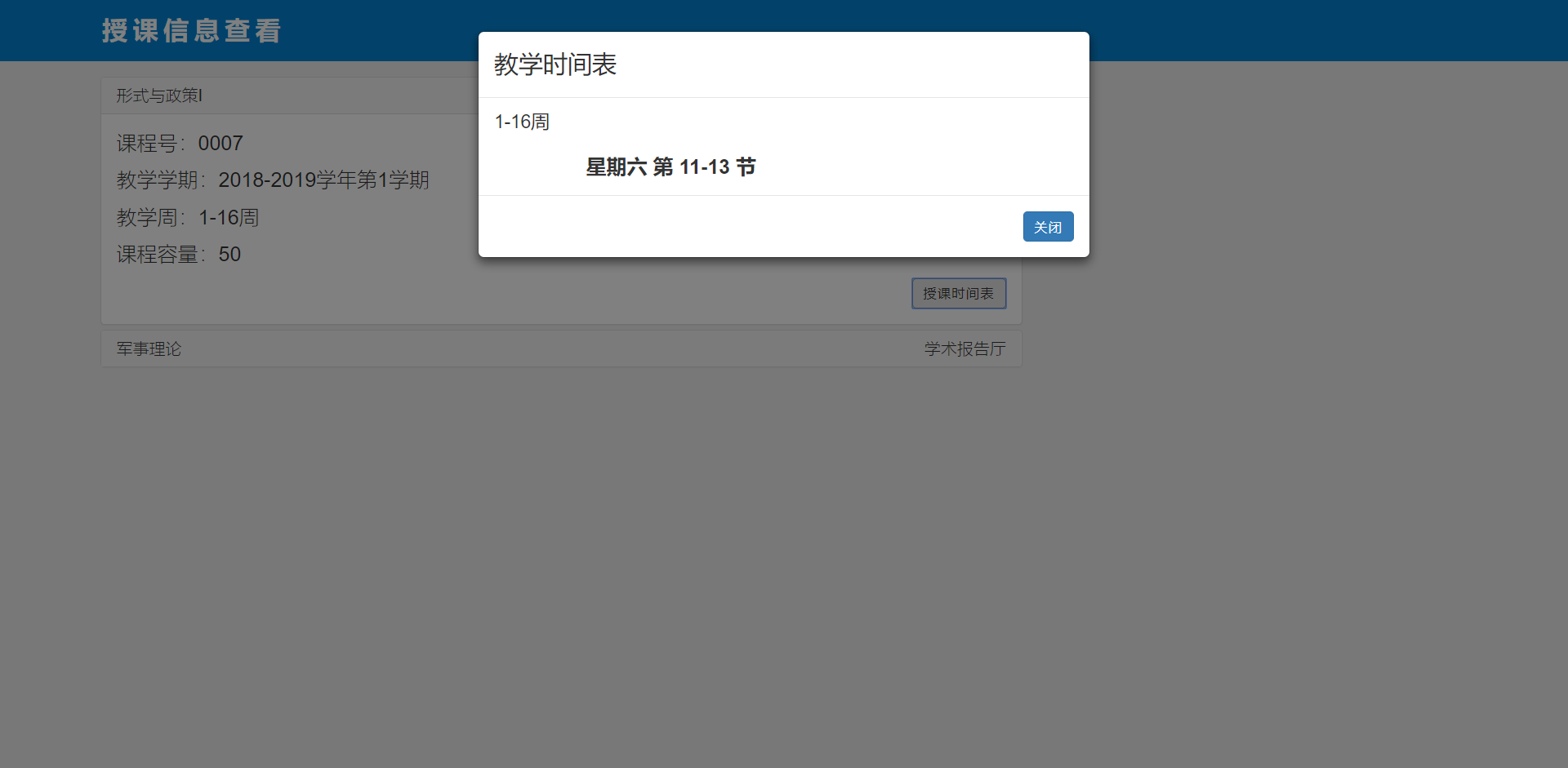


### 查看学生信息

（需要选课之后才可以看）

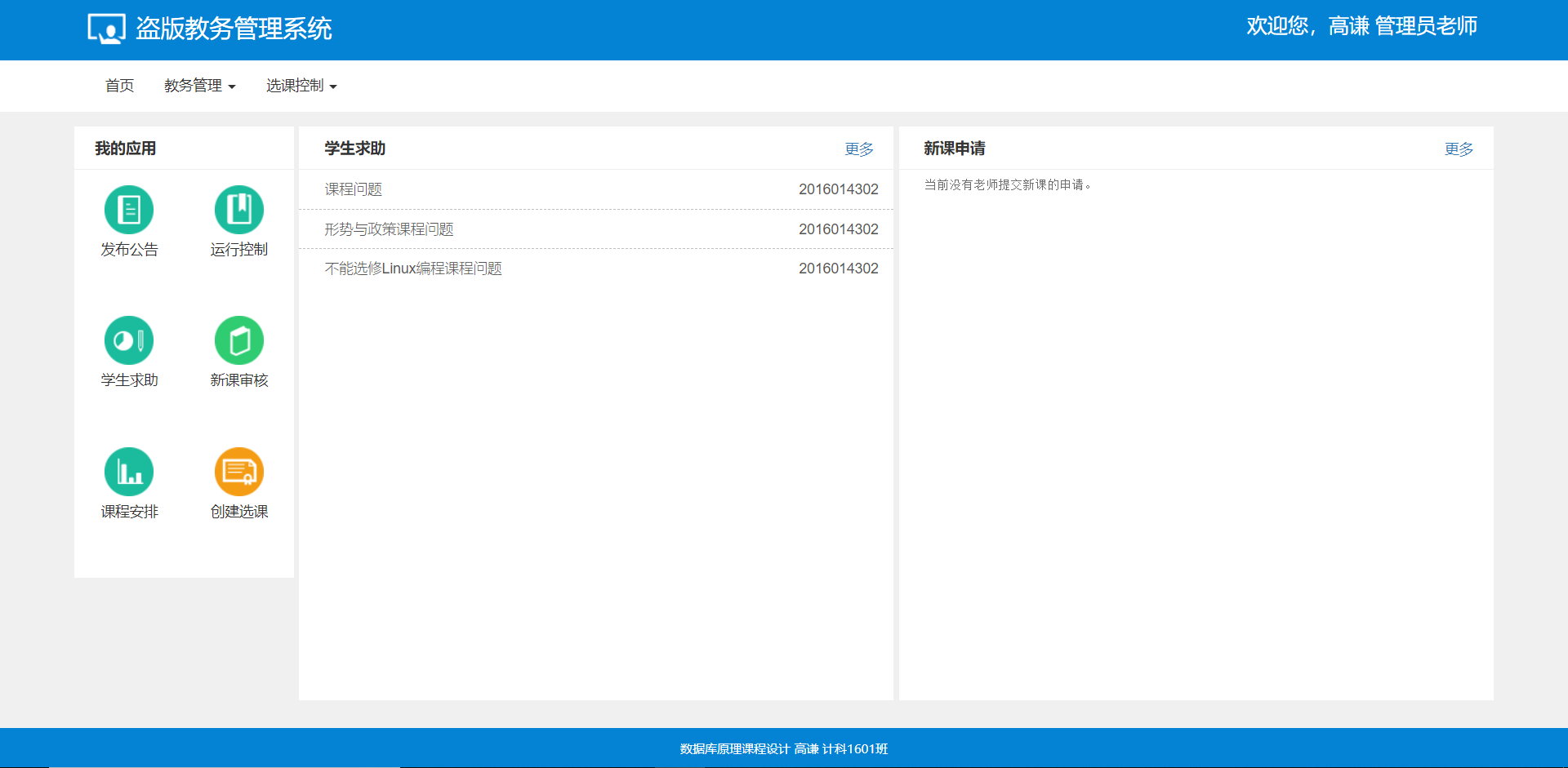
### 授课查看





## 管理员相关

### 管理员首页



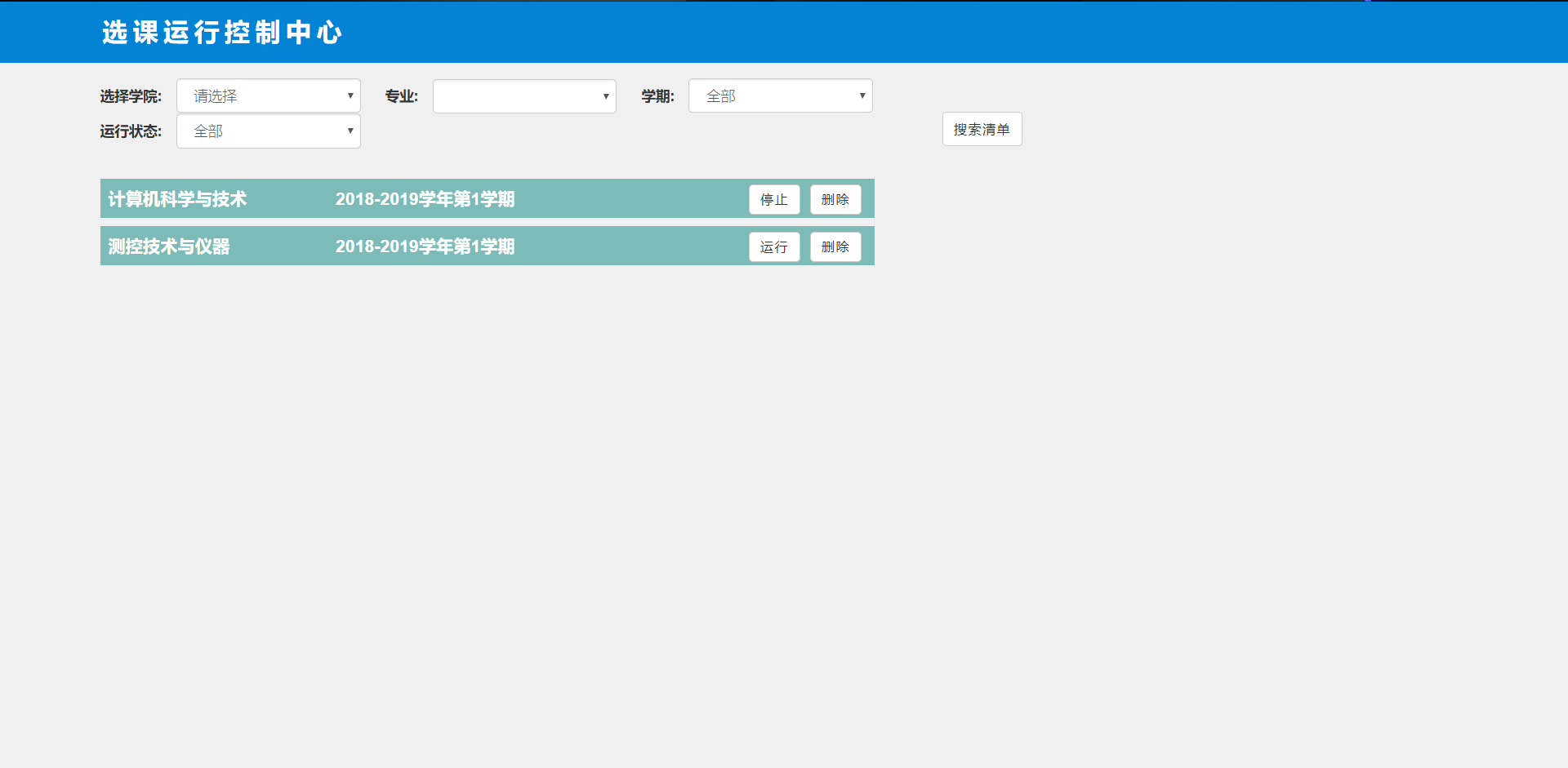
### 安排课程



### 添加人员



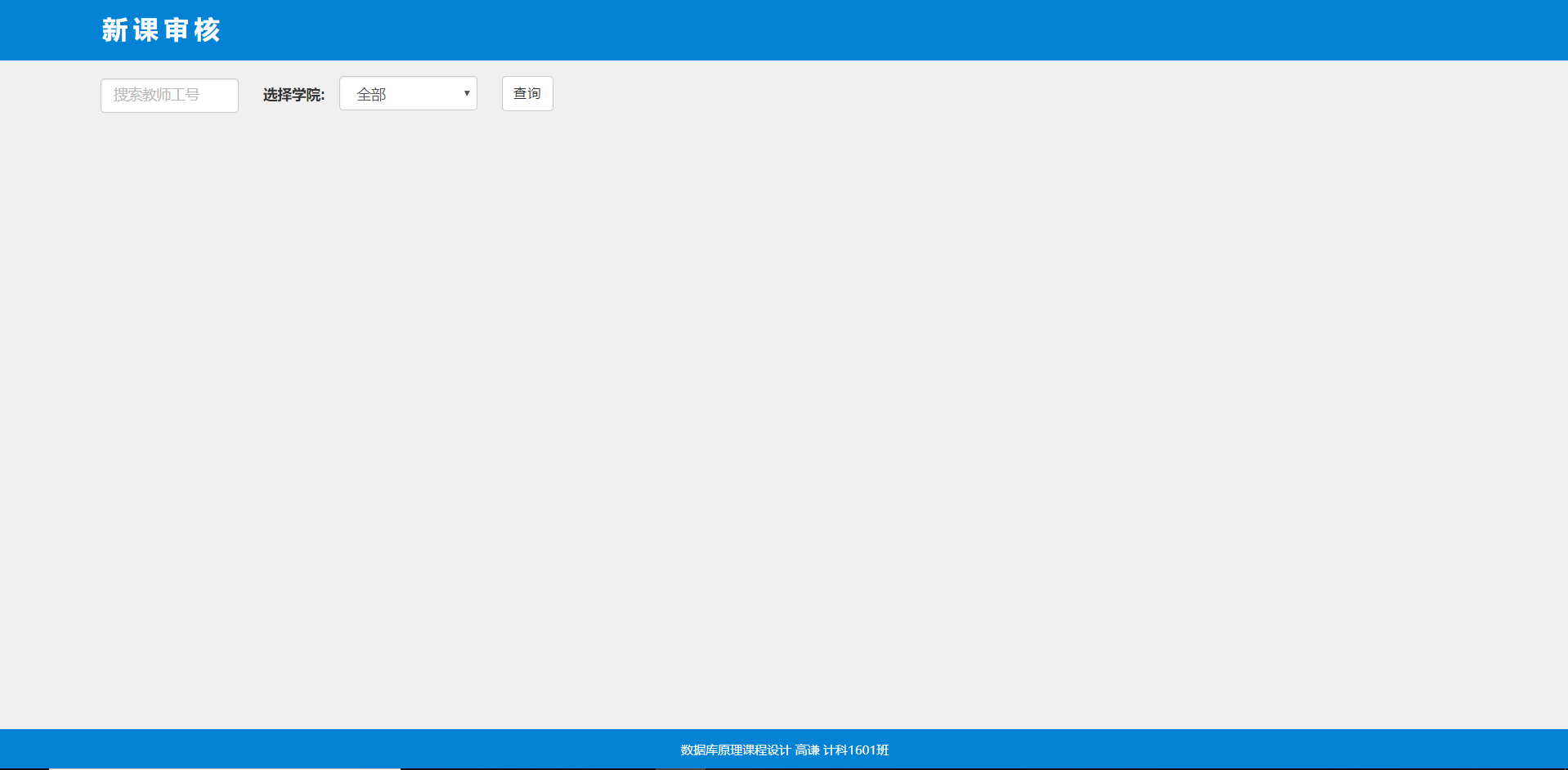
### 运行控制



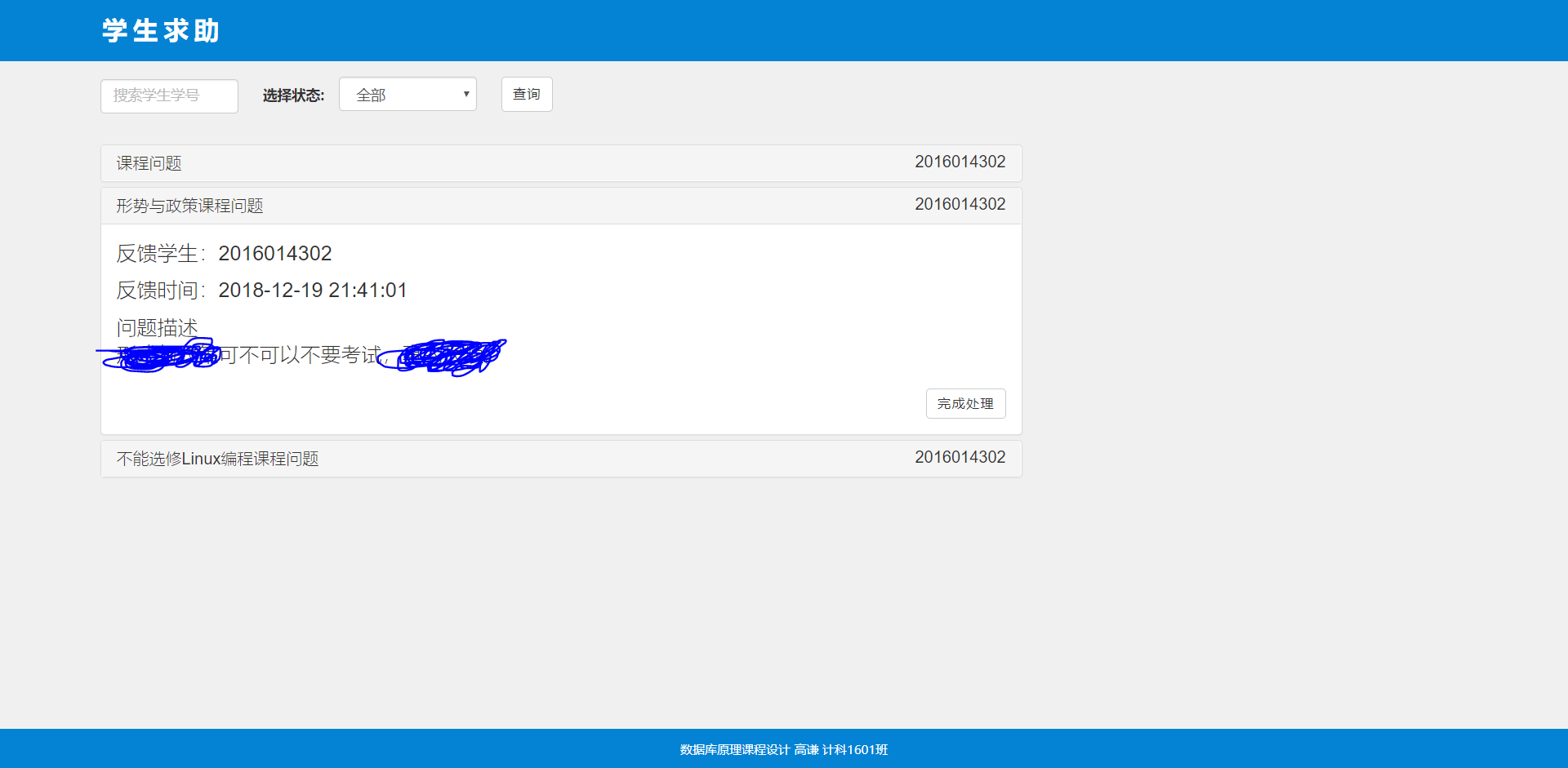
### 创建选课清单



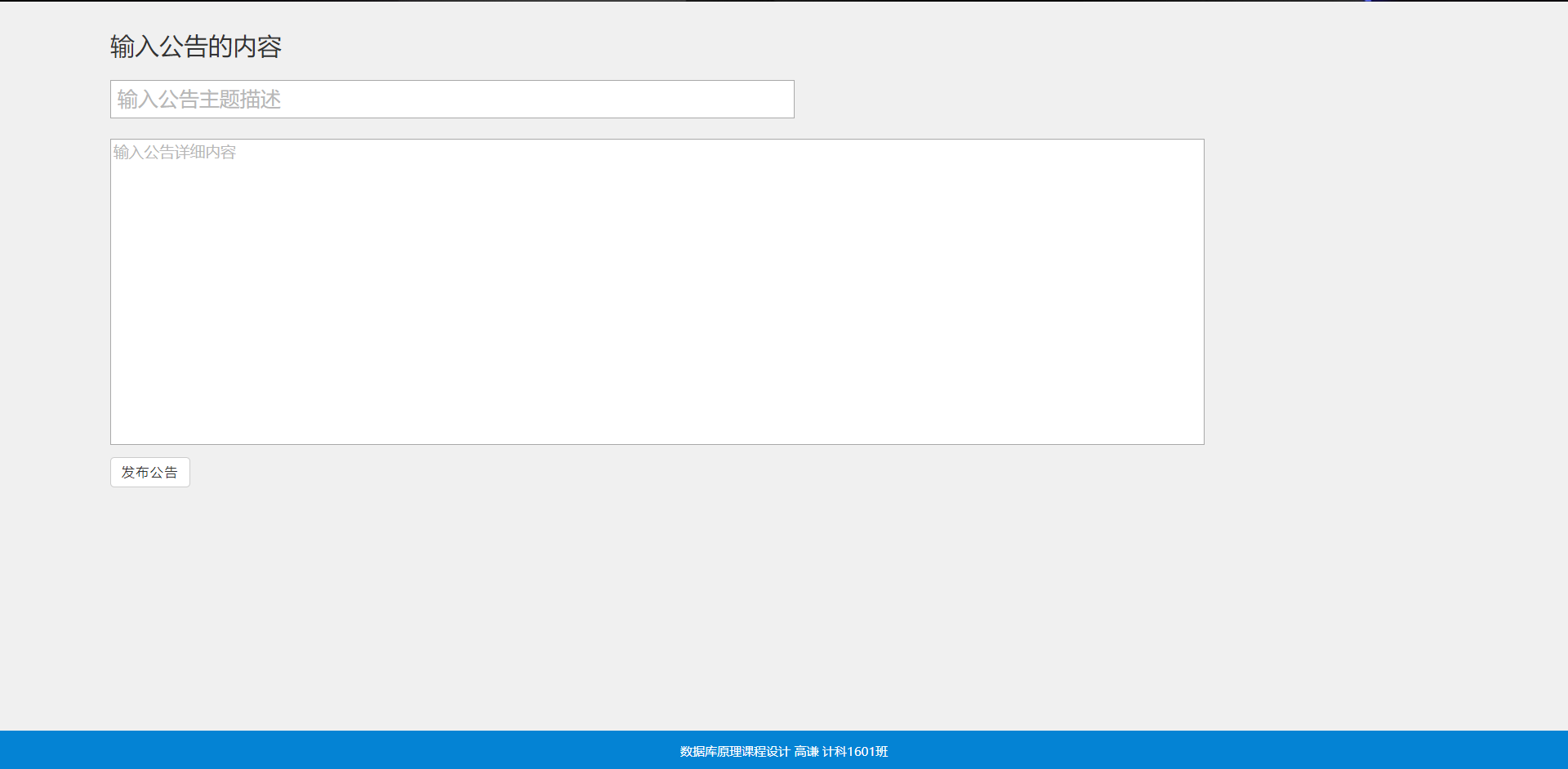
### 新课审核



### 学生求助信息



### 发布公告



### 课程信息



# 第四部分：总结与反思

通过学习数据库原理这门课，我学到的不仅仅是技术方面的东西，更加重要的是学会了如何去进行需求分析，对系统开发有了一个清晰的认识，对数据库的设计有了更深层次的认知，或许只有真正自己开发了之后才会有很深刻的理解吧，或许这就是数据库原理课程设计的目的所在。这门课程我从零开始学习Java web开发，从零开始学习JDBC，感谢老师给了我机会在同学们面前展示自己，锻炼自己。