

课程设计选题系统--

设计报告

专业班级： 计科1701班

小组成员：王澳星，徐鑫强，吴锦、李雅婷

课程教师：

# 第一阶段：开发前的设置和思考

## 题目要求：

该系统由3个功能模块组成：学生功能模块、教师功能模块、管理员功能模块。

### 学生功能：

1. 学生可以在任何能够连接Internet的计算机**登录**到课程设计选题系统中。
2. 学生可以在选题功能模块中浏览到自己心仪的题目并且选择该题目，之后等待出题老师确认。每个学生只能选择一个课程设计题目，并明确标识是否是组长。
3. 每个课程设计题目有选课学生人数的限制，并且，其中有且仅有一个学生是组长。
4. 在老师没有确认之前学生能退掉自己已经选择的题目并且重新选择（是否确认标记）。
5. 学生能修改自己注册时的个人信息，方便信息更新准确。
6. 一旦某个学生成为一个课题的组长，则负责上传课程设计相关文档到系统中。

### 教师功能：

1. 教师可以在任何能够连接Internet的计算机登录到课程设计选题系统中。
2. 教师可以查询到已经选择了自己建立的题目的学生，并且有权利根据学生具体情况决定是否录取还是退选。
3. 教师可以新建课程，并发布该课程的课程设计题目供学生选择，同时指定每组人数。
4. 教师可以修改自己已经建立的题目信息，确保信息的准确性。
5. 教师可以修改自己的注册资料，确保个人信息的准确性。
6. 教师可以导出课程设计选题清单，包括分组编号，题目，学生名单及组长等信息。

### 管理员功能：

1. 管理员可以在任何能够连接Internet的计算机登录到课程设计选题系统中。
2. 管理员可以查看、修改和删除系统内的所有学生个人信息和教师个人信息。
3. 管理员通过新建班级信息，再在班级中导入学生名单，可通过excel文件导入。
4. 管理员可以通过可选题目的查询，待录取题目的查询和已录取题目的查询了解整个课程设计的总体工作进展情况。

要求：该系统需经过实际测试，最终本学期的《数据库实用技术》全部课程设计信息将存储和管理在该系统中。需提供完整的数据库建库脚本和系统部署手册。

## 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，年级，性别，专业，班级，学院，电话，邮箱，密码等）

课程设计选题（题号，题名，内容，教师工号，教师名，课程，学院，人数，学期等）

老师（工号，姓名，性别，学院，职称，电话，邮箱，密码等）

管理员（工号，姓名，性别，学院，职称，电话，邮箱，密码等）

### 联系（红色表示主键）

选题（学期，题号，题名，学号，学生名，组长标记，审核状态，成绩等）

设计（学期，题号，题名，教师工号，教师名等）

管理（学期，题号，题名，教师工号，教师名，学号，学生名，选题开始时间，选题截止时间等）

## 数据库物理结构设计

**数据表信息如下：**

### 概览：

数据库名称为SelectTitleDB，数据表数量为个

### 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）

### 课程设计选题表

### 选择表

### 课程设计选题设计表

### 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

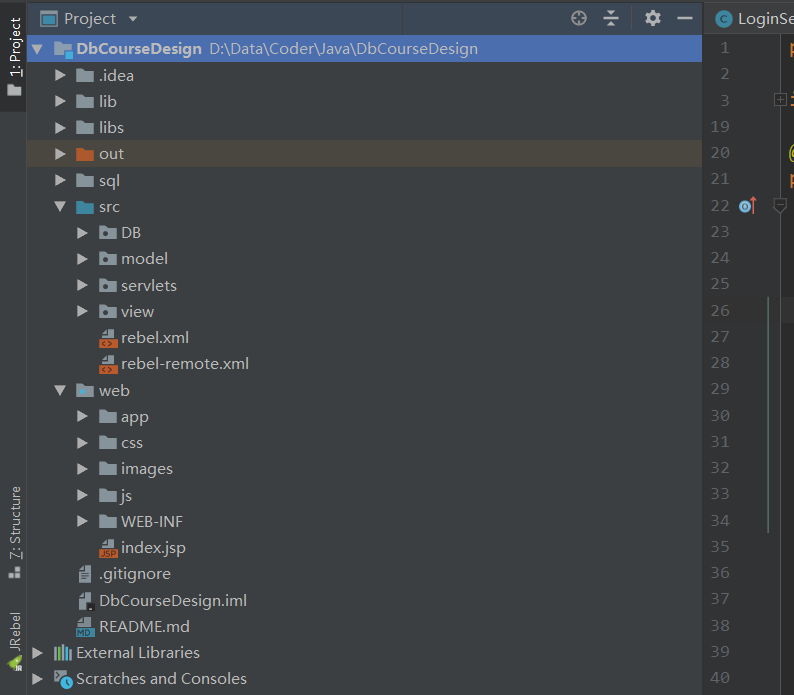
web：存放静态资源和网页模板的类：

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

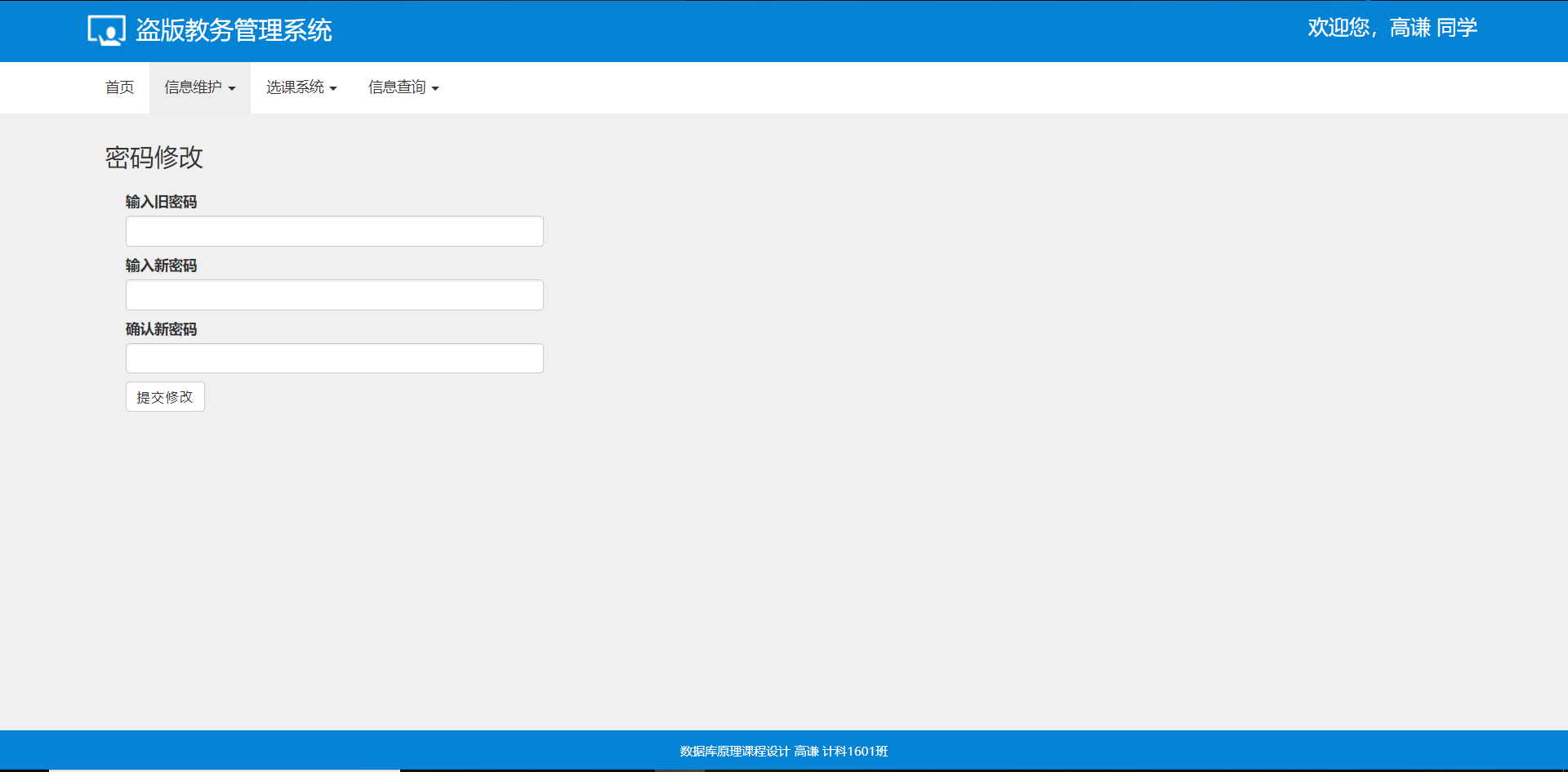


## 学生相关

### 学生首页



### 修改密码

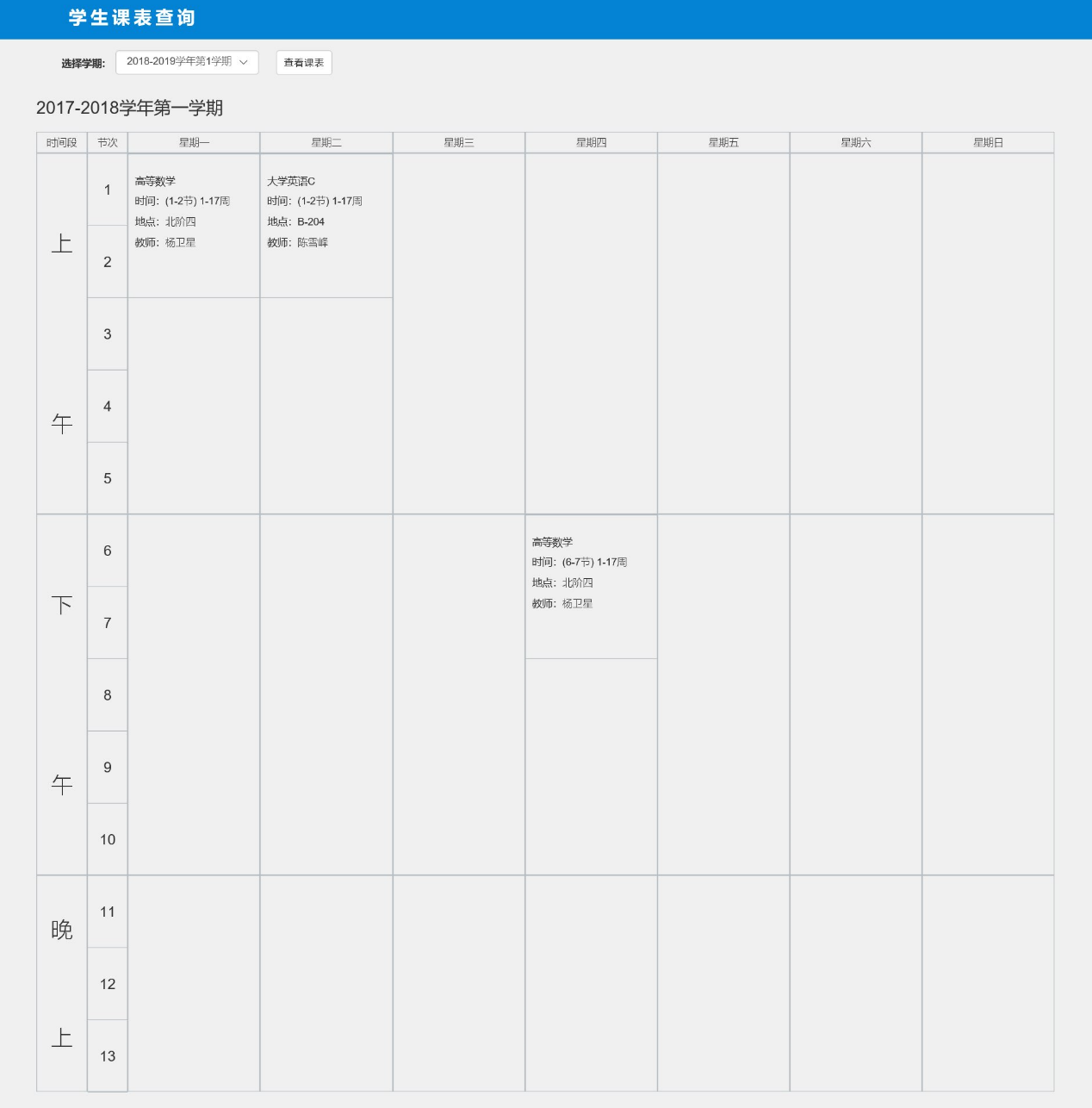


### 选课





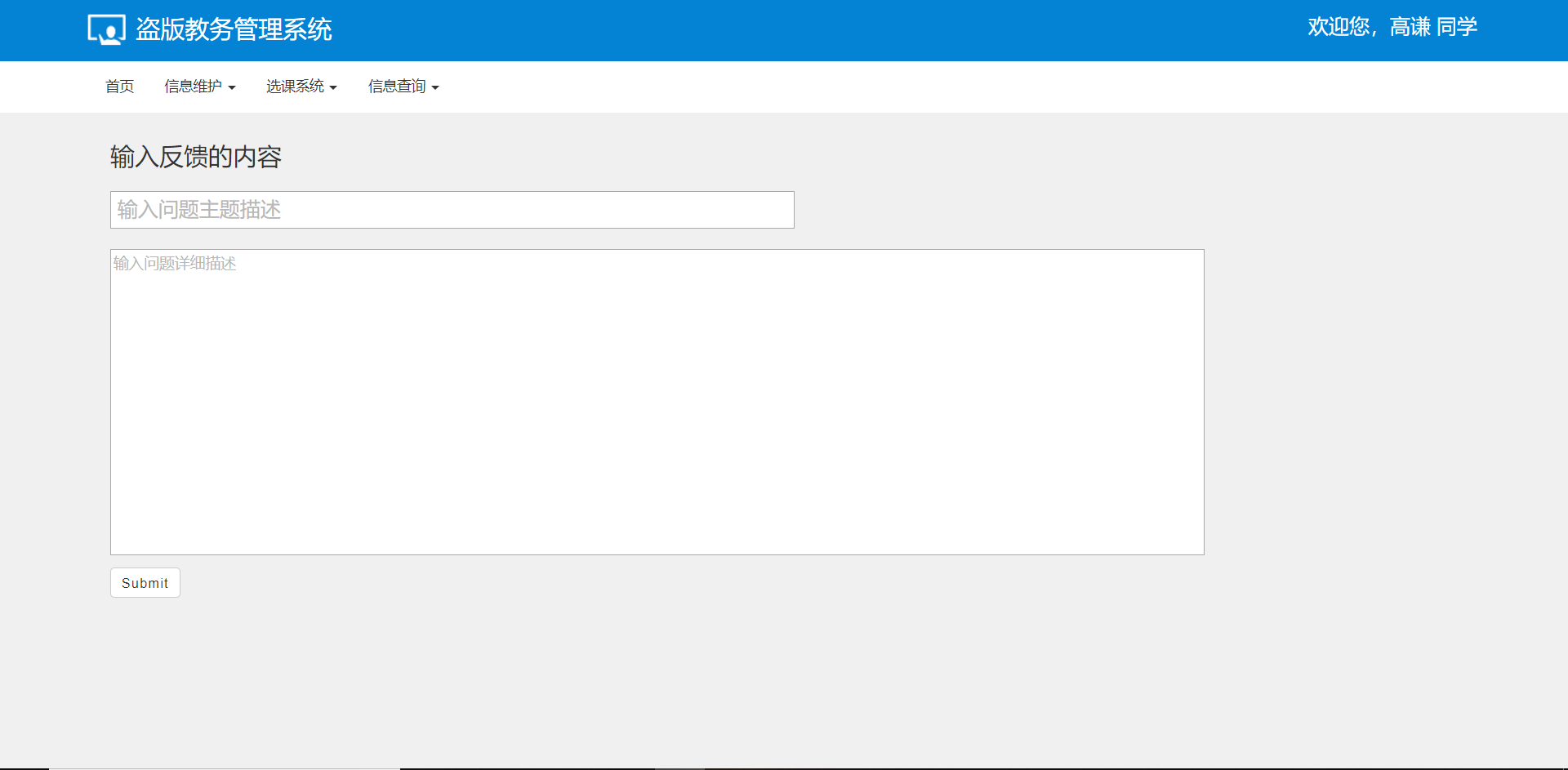
### 查看课表



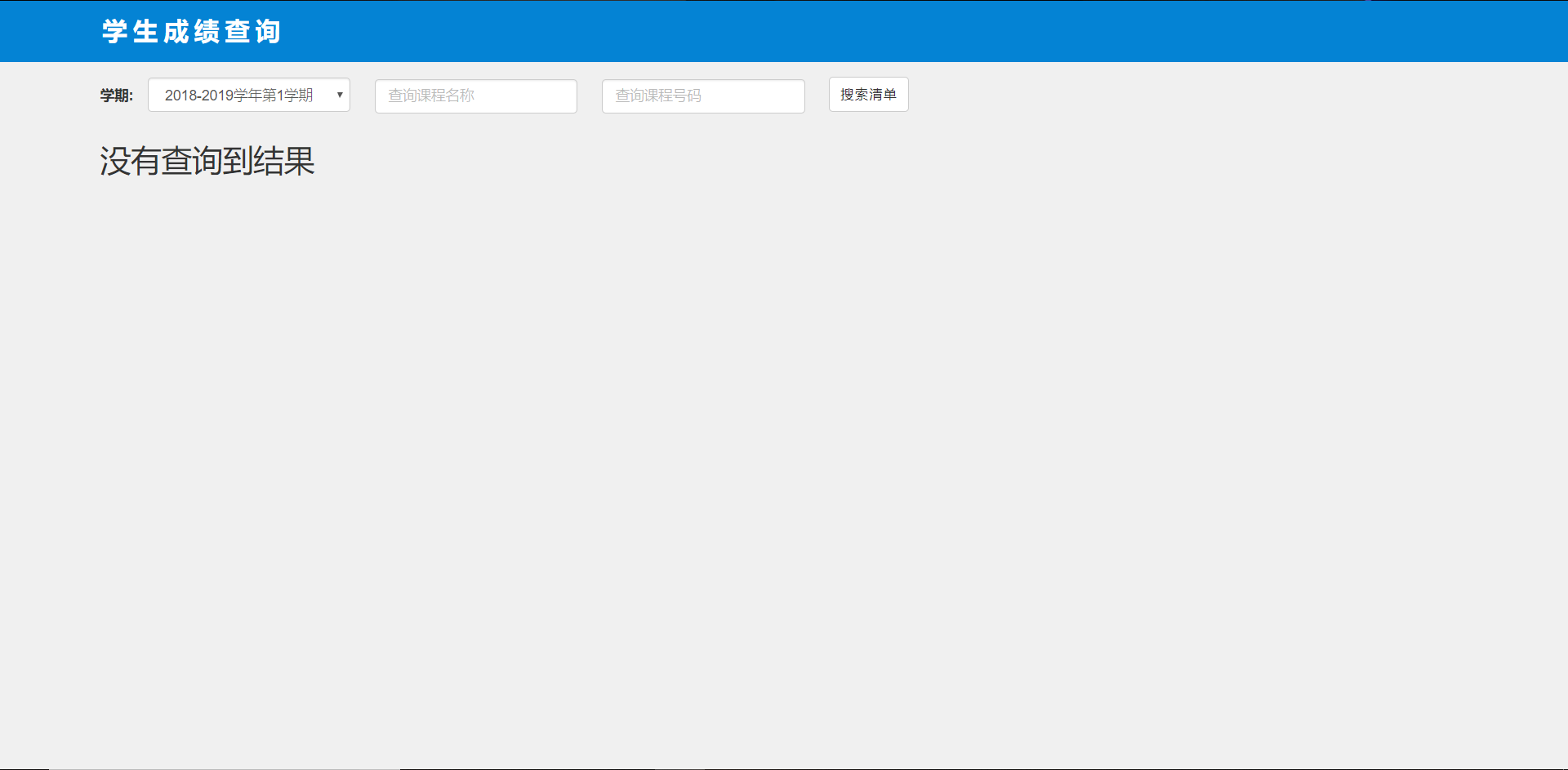
### 查看个人信息



### 联系老师



### 成绩查询



## 教师相关

### 教师首页



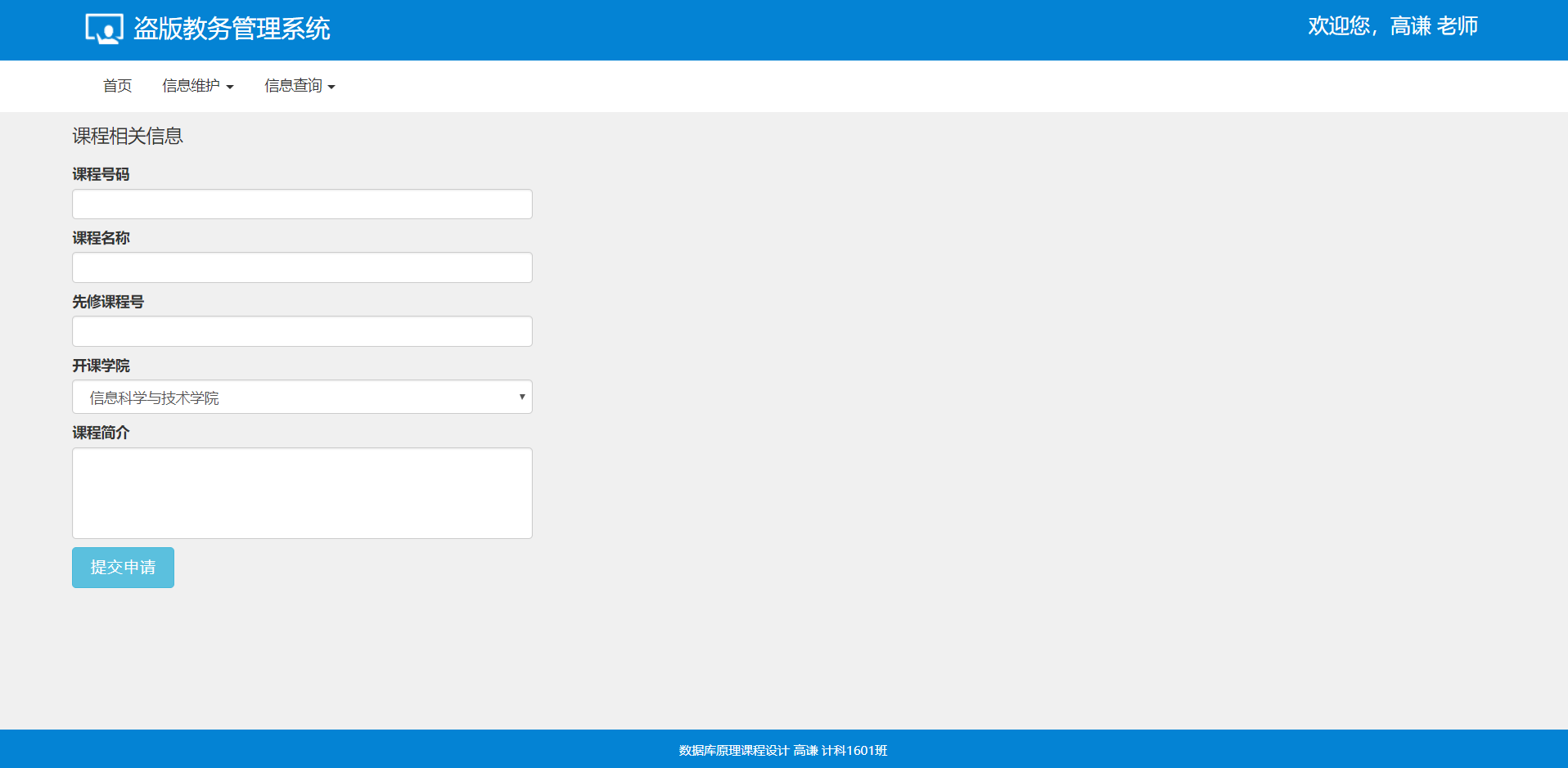
### 个人信息维护

（同学生版页面）

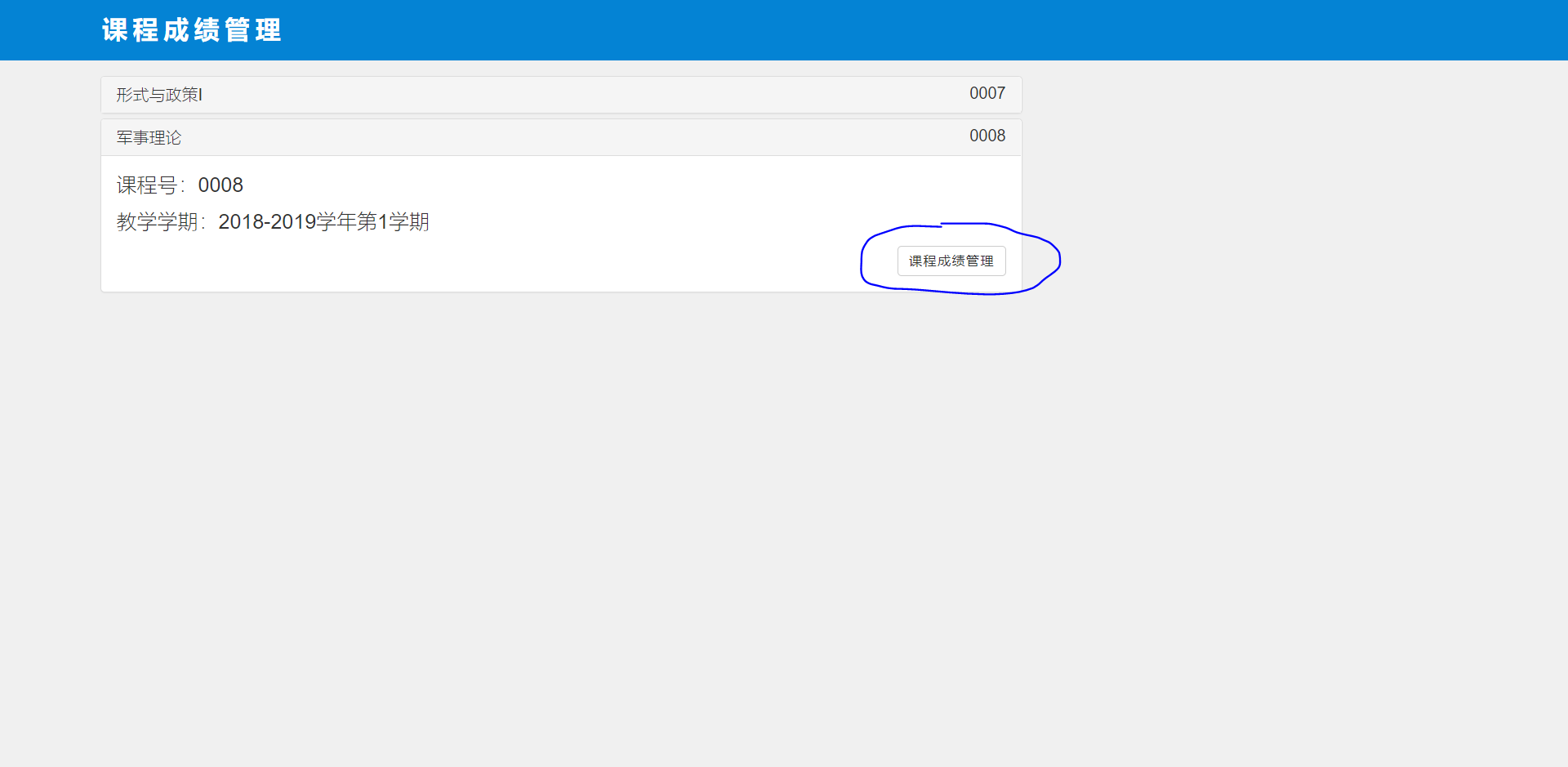
### 修改密码

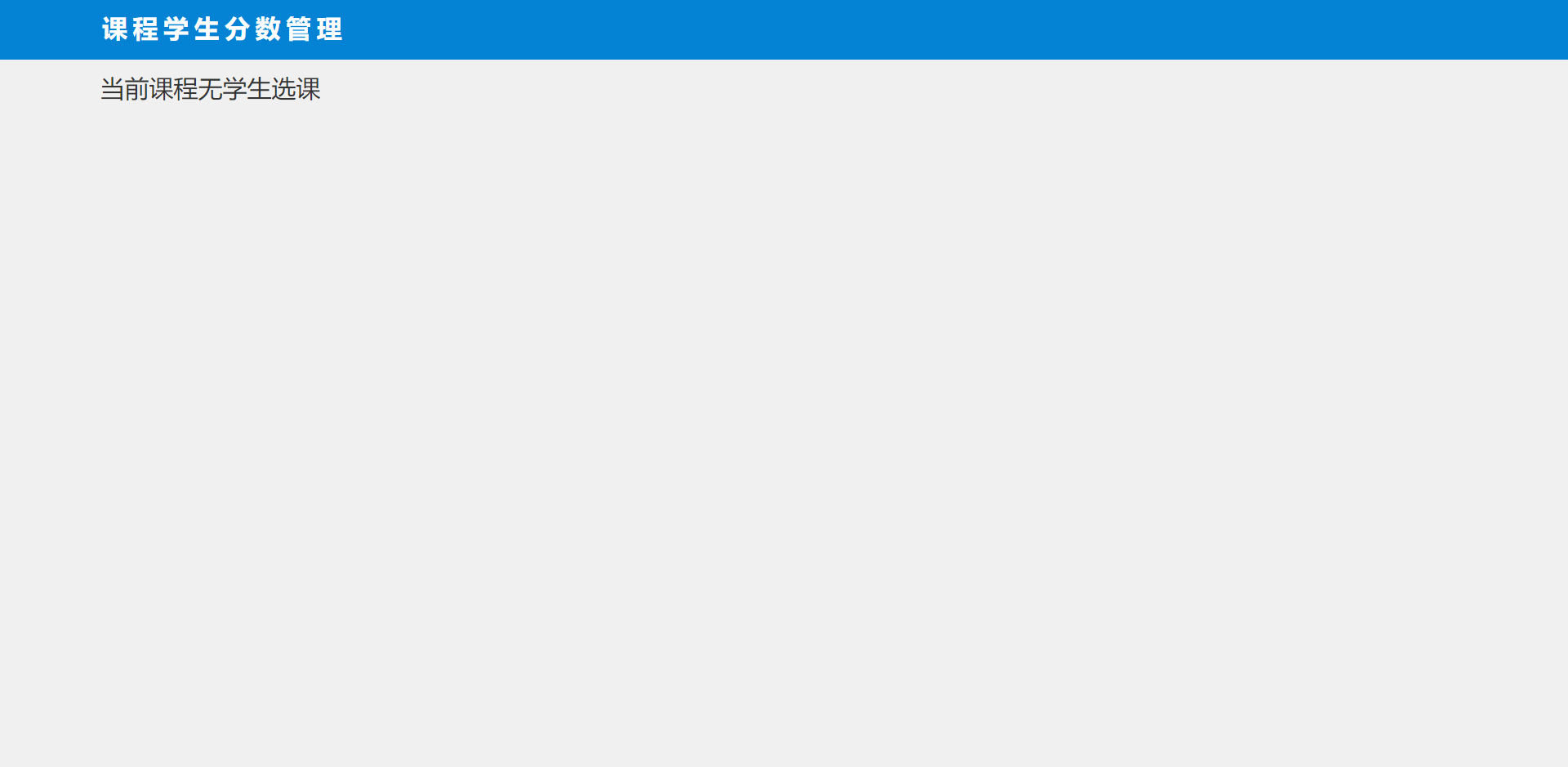
（同学生版页面）

### 申请新课



### 课程评分和修改





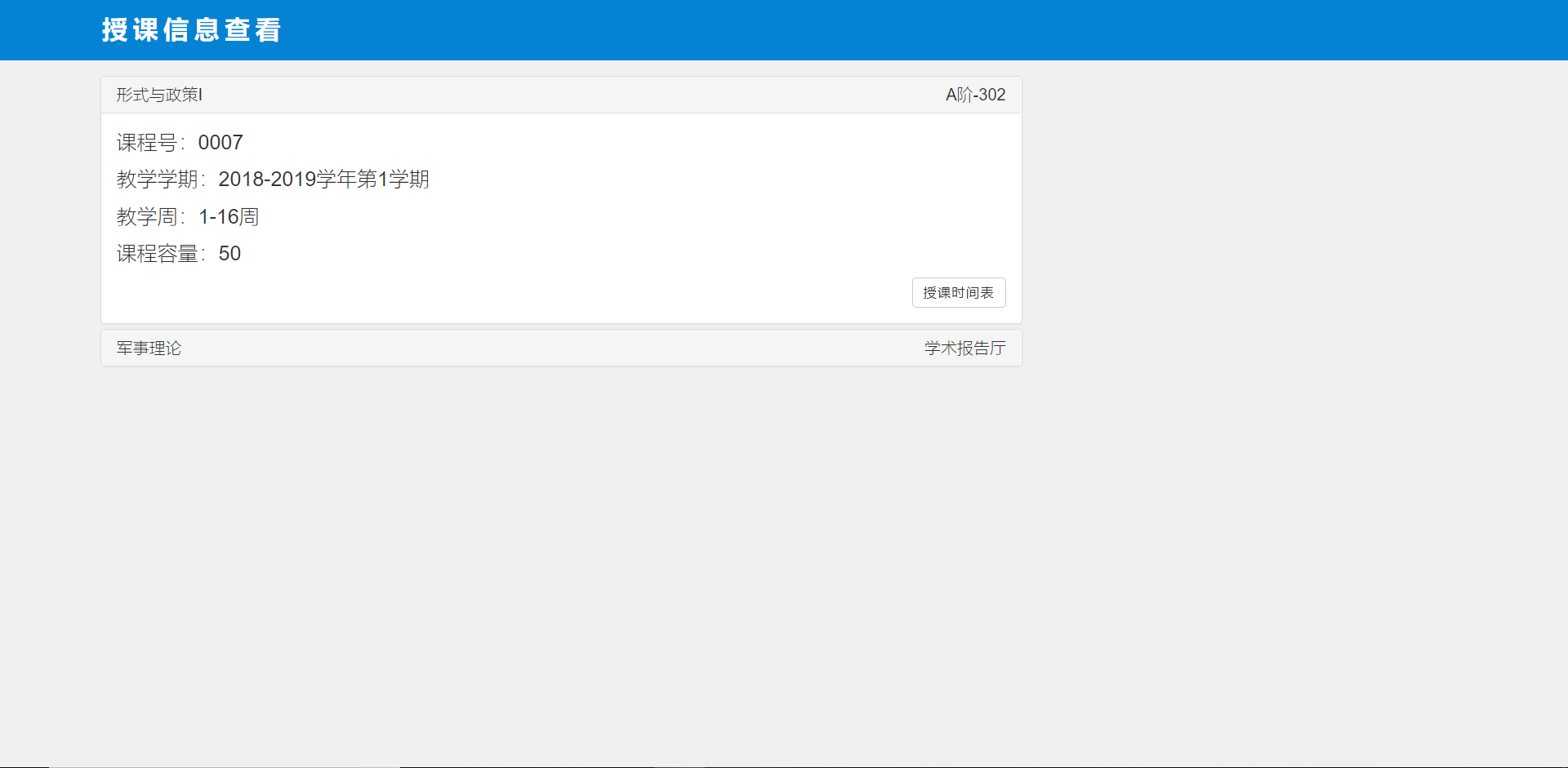
### 查看公告

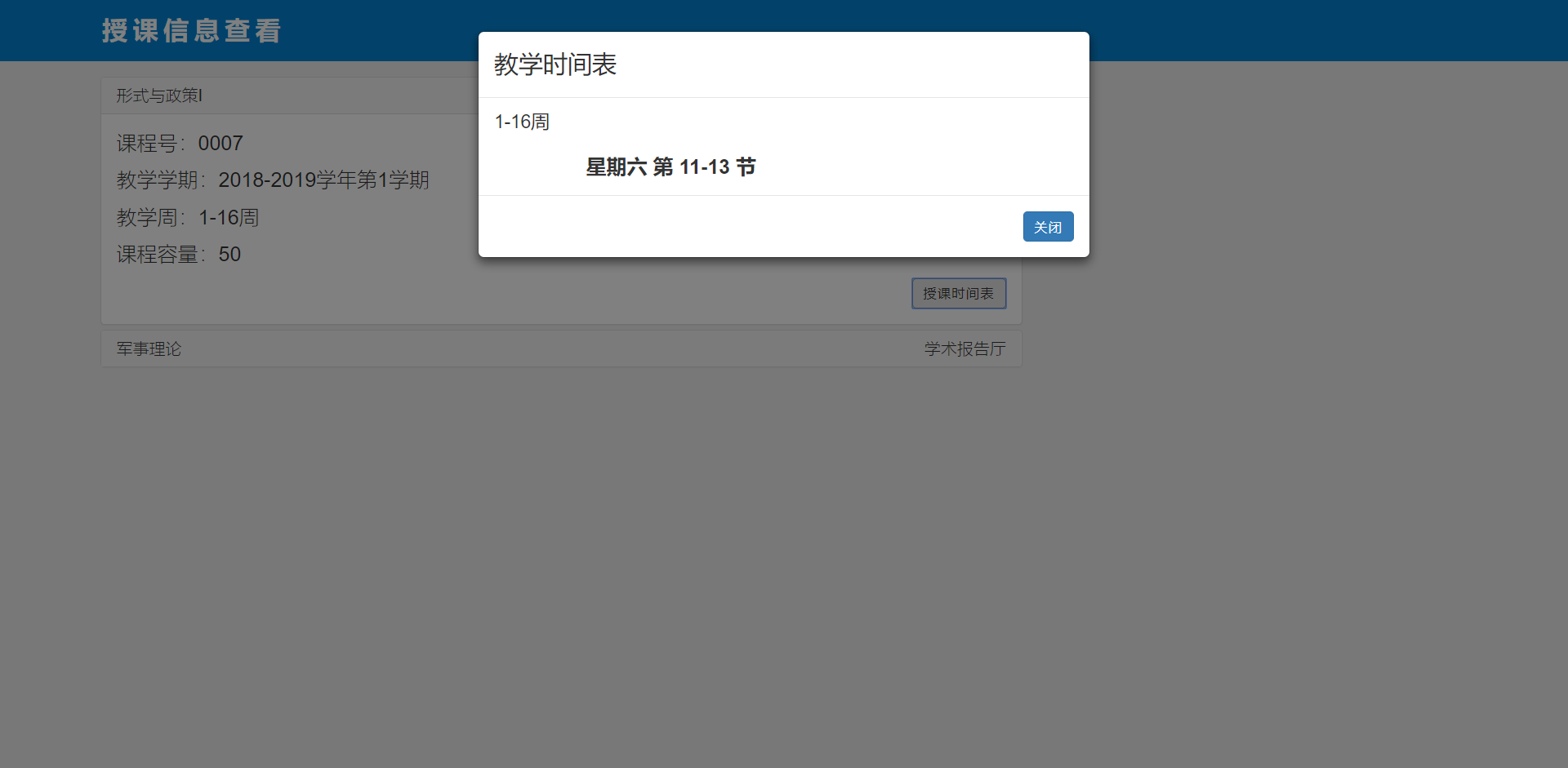


### 查看学生信息

（需要选课之后才可以看）

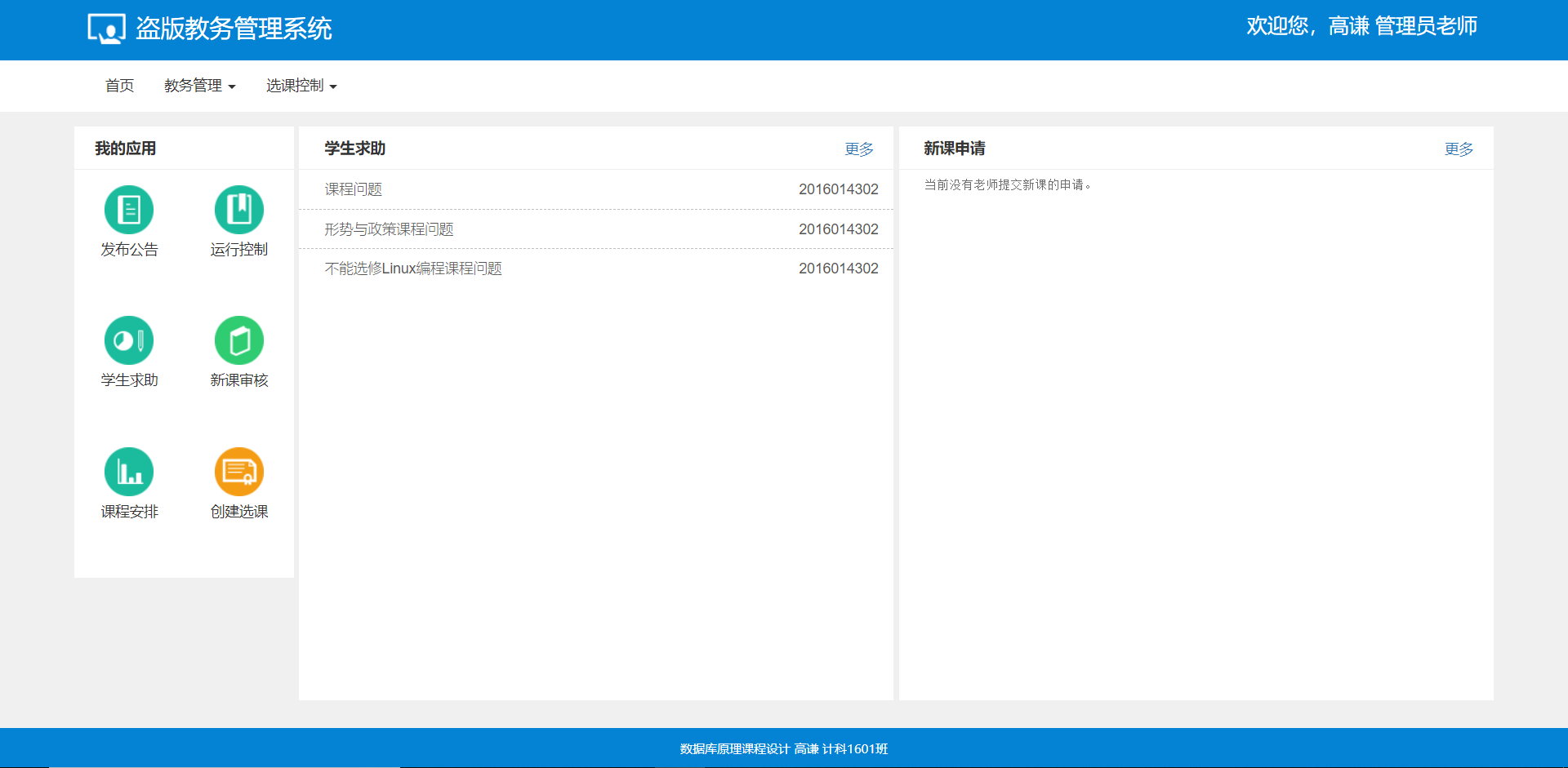
### 授课查看





## 管理员相关

### 管理员首页



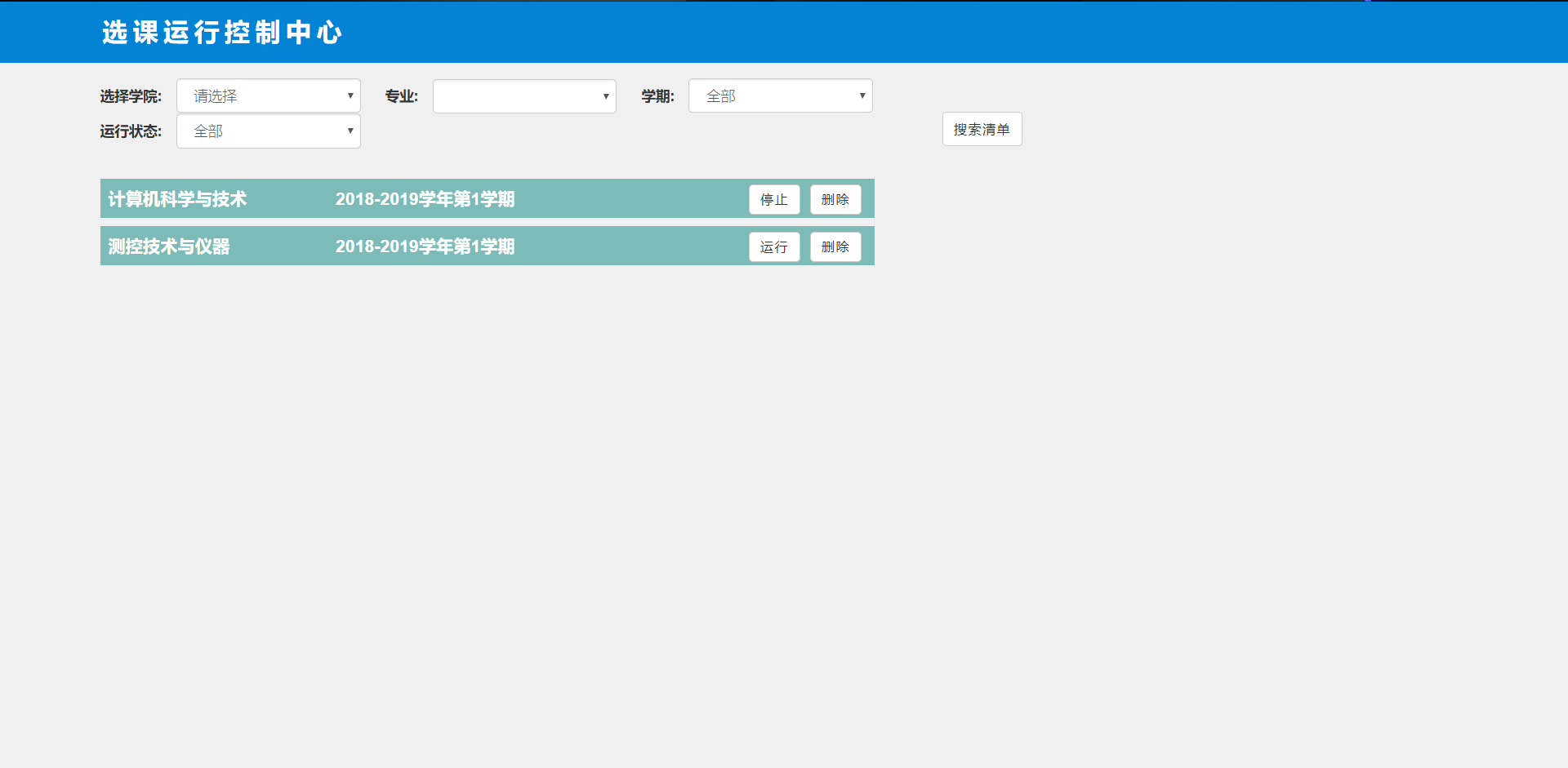
### 安排课程



### 添加人员



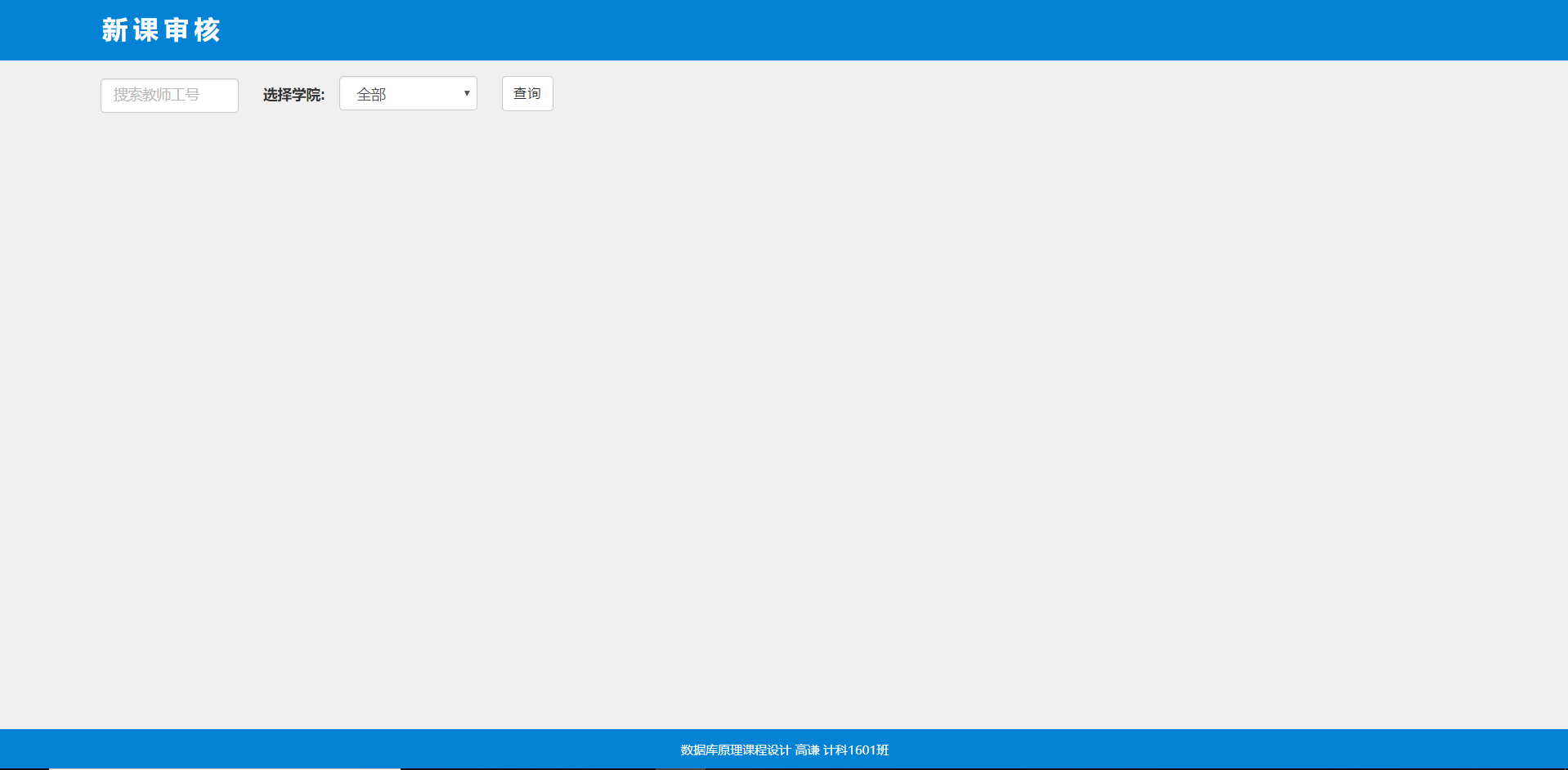
### 运行控制



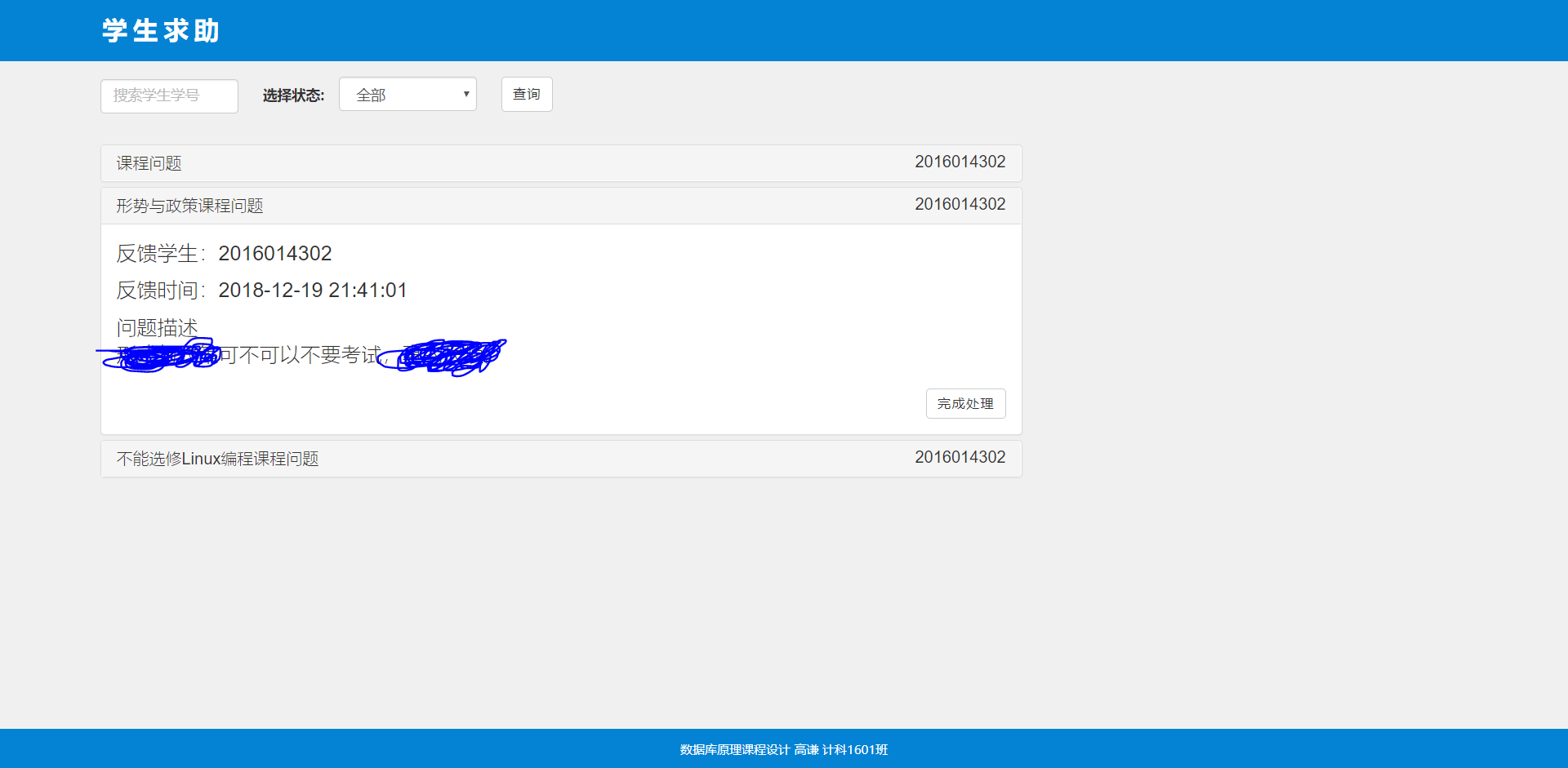
### 创建选课清单



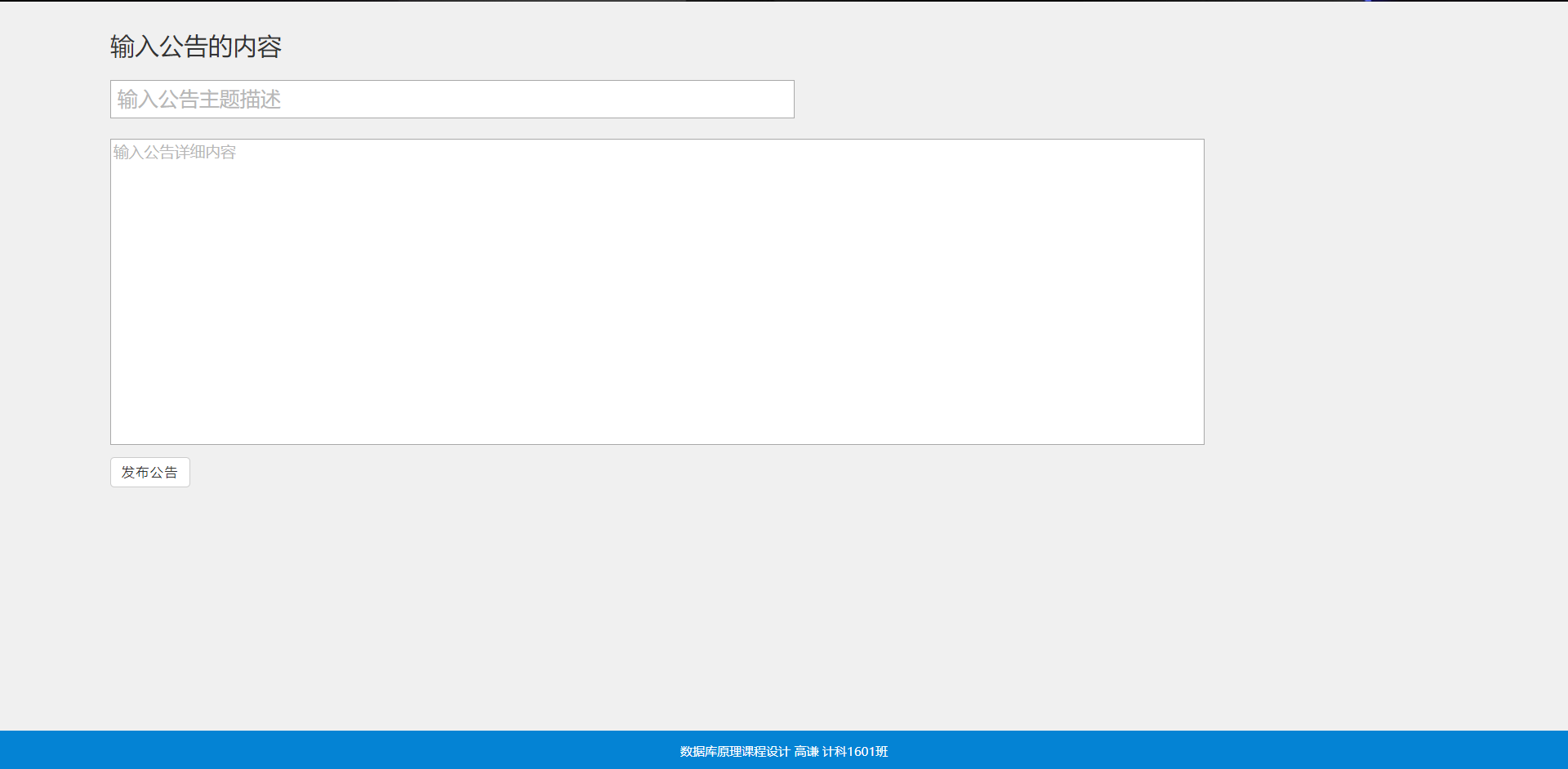
### 新课审核



### 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。