兔子大学北京校区

**程序设计报告**

**基于SpringBoot + Vue的学生成绩管理系统的设计与实现**

学 院： 计算机学院

专 业 名 称： 软件工程

学 生 姓 名： 忧伤大白兔

学 号： 00000001

完成日期： xxxx年 xx月xx日

目录

[一、需求分析 7](#_Toc160396503)

[1. 引言 7](#_Toc160396504)

[1.1 编写目的 7](#_Toc160396505)

[1.2 项目说明 7](#_Toc160396506)

[2. 产品介绍 7](#_Toc160396507)

[2.1 产品概要说明 7](#_Toc160396508)

[2.2 产品用户定位 7](#_Toc160396509)

[2.3 产品中的角色 7](#_Toc160396510)

[3. 产品总体业务流程图 8](#_Toc160396511)

[4. 产品功能结构图 8](#_Toc160396512)

[5.任务管理 9](#_Toc160396513)

[5.1 项目现状 9](#_Toc160396514)

[5.2 项目目标 9](#_Toc160396515)

[6.功能需求 9](#_Toc160396516)

[6.1 功能概述 9](#_Toc160396517)

[6.2 功能点清单 9](#_Toc160396518)

[6.3 功能点描述 10](#_Toc160396519)

[6.3.1 课程信息管理 10](#_Toc160396520)

[6.3.2 课程表管理 11](#_Toc160396521)

[6.3.3 成绩管理 11](#_Toc160396522)

[6.3.4 用户功能 11](#_Toc160396523)

[6.3.5 用户信息管理 11](#_Toc160396524)

[二、可行性分析 12](#_Toc160396525)

[1. 引言 12](#_Toc160396526)

[1.1 标识 12](#_Toc160396527)

[1.2 背景 12](#_Toc160396528)

[1.3 项目概述 13](#_Toc160396529)

[1.4 文档概述 13](#_Toc160396530)

[2. 可行性分析的前提 13](#_Toc160396531)

[2.1 项目的要求 13](#_Toc160396532)

[2.2 项目的目标 14](#_Toc160396533)

[2.3 项目的环境、条件、假定和限制 14](#_Toc160396534)

[2.4 进行可行性分析的方法 14](#_Toc160396535)

[3. 可选的方案 15](#_Toc160396536)

[3.1 可选的系统方案一 15](#_Toc160396537)

[3.2 可选的系统方案二 15](#_Toc160396538)

[3.3 选用的方案 16](#_Toc160396539)

[4. 所建议的系统 16](#_Toc160396540)

[4.1 对所建议的系统的说明 16](#_Toc160396541)

[4.3 影响 16](#_Toc160396542)

[4.3.1 设备 16](#_Toc160396543)

[4.3.2 软件 16](#_Toc160396544)

[4.3.3 运行 16](#_Toc160396545)

[4.3.4 开发 17](#_Toc160396546)

[4.3.5 环境 17](#_Toc160396547)

[4.4 局限性 17](#_Toc160396548)

[5. 经济可行性 17](#_Toc160396549)

[5.1 投资 17](#_Toc160396550)

[5.2 预期的经济效益 17](#_Toc160396551)

[5.3 市场预测 18](#_Toc160396552)

[6. 技术可行性 18](#_Toc160396553)

[6.1 Java语言 18](#_Toc160396554)

[6.2 MySQL 18](#_Toc160396555)

[6.3 HTML、JavaScript 18](#_Toc160396556)

[7 社会可行性 18](#_Toc160396557)

[8. 结论 19](#_Toc160396558)

[三、概要设计 20](#_Toc160396559)

[1. 引言 20](#_Toc160396560)

[1.1目的 20](#_Toc160396561)

[1.2 范围 20](#_Toc160396562)

[1.2.1 系统目标 20](#_Toc160396563)

[1.2.2 主要软件需求 20](#_Toc160396564)

[1.2.3 软件设计约束、限制 20](#_Toc160396565)

[1.3 参考资料 21](#_Toc160396566)

[1.4版本信息 21](#_Toc160396567)

[2.数据设计 21](#_Toc160396568)

[2.1数据对象和形成的数据结构 21](#_Toc160396569)

[2.1.1管理员信息表（admin） 21](#_Toc160396570)

[2.1.2 课程基本信息表（course） 22](#_Toc160396571)

[2.1.3 课程扩展信息表（course\_info） 22](#_Toc160396572)

[2.1.4 专业表（profession） 22](#_Toc160396573)

[2.1.5 页面主题切换表（silent） 22](#_Toc160396574)

[2.1.6 学生信息表（student） 22](#_Toc160396575)

[2.1.7 学生课程关联表（student\_course） 22](#_Toc160396576)

[2.1.8 教师信息表（teacher） 22](#_Toc160396577)

[2.1.9 教师课程关联表（teacher\_course） 22](#_Toc160396578)

[2.1.10 排课时间信息表（timetable） 23](#_Toc160396579)

[2.1.11 文件管理表（upload） 23](#_Toc160396580)

[2.1.12 排课表（week） 23](#_Toc160396581)

[2.2数据库结构 23](#_Toc160396582)

[2.3 全局数据 23](#_Toc160396583)

[3 体系结构设计 24](#_Toc160396584)

[3.1数据和控制流复审 24](#_Toc160396585)

[3.2 得出的程序结构 24](#_Toc160396586)

[4.界面设计 24](#_Toc160396587)

[4.1 人机界面规约 24](#_Toc160396588)

[4.1.1 课程信息管理界面 24](#_Toc160396589)

[4.1.2 课程表界面 24](#_Toc160396590)

[4.1.3 成绩管理界面 25](#_Toc160396591)

[4.1.4 用户管理界面 25](#_Toc160396592)

[4.2人机界面设计规约 25](#_Toc160396593)

[4.3 内部模块 25](#_Toc160396594)

[5、模块过程设计 26](#_Toc160396595)

[5.1 登录模块 26](#_Toc160396596)

[5.2 课程管理模块 26](#_Toc160396597)

[5.3 课程表（排课）模块 26](#_Toc160396598)

[5.4 成绩管理模块 26](#_Toc160396599)

[5.5 用户信息管理模块 26](#_Toc160396600)

[5.5.1 学生用户管理模块 26](#_Toc160396601)

[5.5.2 教师用户管理模块 27](#_Toc160396602)

[5.5.3 管理员用户管理模块 27](#_Toc160396603)

[四、详细设计 28](#_Toc160396604)

[1 引言 28](#_Toc160396605)

[1.1 编写目的 28](#_Toc160396606)

[1.2 项目背景 28](#_Toc160396607)

[1.3 参考资料 28](#_Toc160396608)

[2 总体设计 28](#_Toc160396609)

[2.1 需求概述 28](#_Toc160396610)

[2.2 软件结构 29](#_Toc160396611)

[3 模块设计 29](#_Toc160396612)

[3.1 模块基本信息 29](#_Toc160396613)

[3.2 功能概述 30](#_Toc160396614)

[3.3 模块处理逻辑 31](#_Toc160396615)

[4 数据库设计 34](#_Toc160396616)

[4.1 E-R图 34](#_Toc160396617)

[4.2 表设计 34](#_Toc160396618)

[4.2.1 管理员信息表 34](#_Toc160396619)

[4.2.2 课程基本信息表 35](#_Toc160396620)

[4.2.3 课程扩展信息表 35](#_Toc160396621)

[4.2.4 专业信息表 35](#_Toc160396622)

[4.2.5 页面主题表 35](#_Toc160396623)

[4.2.6 学生信息表 35](#_Toc160396624)

[4.2.7 学生-课程关联表 36](#_Toc160396625)

[4.2.8 教师信息表 36](#_Toc160396626)

[4.2.9 教师-课程关联表 36](#_Toc160396627)

[4.2.10 排课时间表 37](#_Toc160396628)

[4.2.11 文件信息表 37](#_Toc160396629)

[4.2.12 周课程表信息表 37](#_Toc160396630)

[5 接口设计 38](#_Toc160396631)

[5.1 外部接口 38](#_Toc160396632)

[5.1.1 登录界面 38](#_Toc160396633)

[5.1.2管理员首页 38](#_Toc160396634)

[5.1.3 教师首页 39](#_Toc160396635)

[5.1.4 学生首页 39](#_Toc160396636)

[5.1.5 课程表 39](#_Toc160396637)

[5.1.6 成绩查询界面 40](#_Toc160396638)

[5.1.7 成绩统计界面 40](#_Toc160396639)

[5.1.8 课程管理 41](#_Toc160396640)

[5.1.9 成绩管理 41](#_Toc160396641)

[5.1.10 用户管理 42](#_Toc160396642)

[5.2 内部接口 42](#_Toc160396643)

[5.2.1 登录接口 42](#_Toc160396644)

[5.2.2 课程管理接口 43](#_Toc160396645)

[5.2.3 成绩管理接口 44](#_Toc160396646)

[5.2.4 用户管理接口 45](#_Toc160396647)

[5.2.5 文件管理接口 45](#_Toc160396648)

[6 性能 47](#_Toc160396649)

[6.1 精度 47](#_Toc160396650)

[6.2 时间特性 47](#_Toc160396651)

[6.3 灵活性 47](#_Toc160396652)

[7 测试 47](#_Toc160396653)

# 一、需求分析

# 1. 引言

## 1.1 编写目的

编写此需求分析说明书是因为项目持续周期长，为了使用户和团队开发人员对所开发的系统有一致性的理解。通过阅读此文档，团队开发人员能够迅速了解项目的具体需求与实现，同时用户通过此文档能够迅速了解系统。

## 1.2 项目说明

该项目名称为学生成绩管理系统，技术选型为前后端分离，前端基于Vue.js，后端基于Java语言开发，使用了SpringBoot和MyBatis框架提高开发效率和质量。主要面向高校教育中学生管理、课程管理、教师管理、成绩管理、成绩统计等需求。

# 2. 产品介绍

## 2.1 产品概要说明

实现对学生成绩管理过程中的课程表管理、成绩查询、成绩详情、课程管理、用户管理、账号管理，站在纯手工解决这些问题存在的繁琐步骤和容错率，故设计了这个学生成绩管理系统大大加强日常成绩管理过程中的办事效率和数据管理。

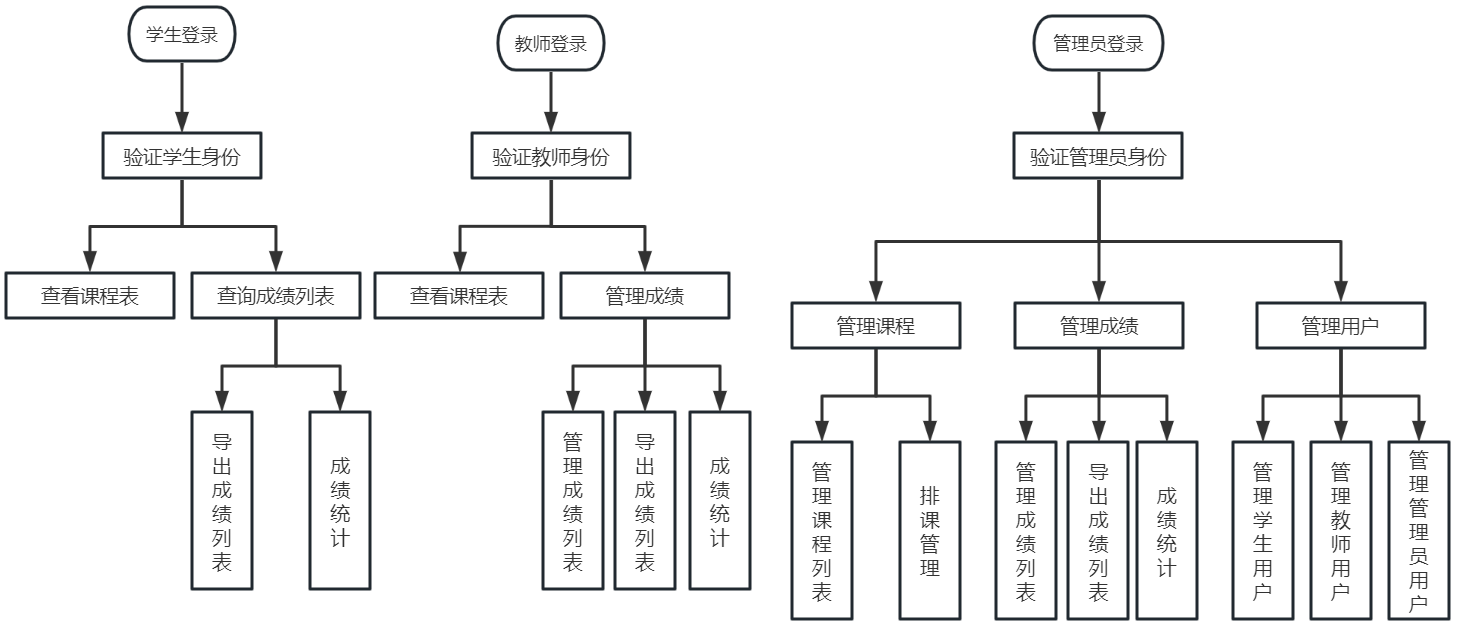
## 2.2 产品用户定位

有大量学生和教学任务的学校成员，包括任课教师、学生、高校领导管理人员。

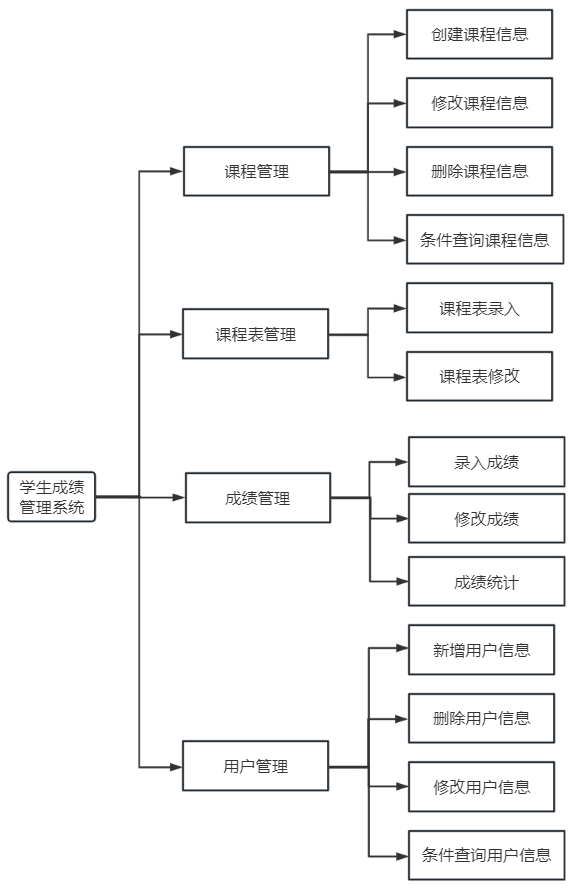
## 2.3 产品中的角色

学生成绩管理员、教师、学生

# 3. 产品总体业务流程图



# **4. 产品功能结构图**



# 5.任务管理

## 5.1 项目现状

目前处于需求分析阶段，后续小组会根据项目需求分析进一步进行可行性分析，并逐步对页面的设计、详细功能进行设计。

## 5.2 项目目标

在规定时间内完成一个学生成绩管理系统网页版，可以在本地成功运行，并为后续项目上线做准备。

# 6.功能需求

## 6.1 功能概述

实现图书管理系统以下功能：用户登录、课程管理、课程表管理、成绩管理、用户管理、账号管理。

## 6.2 功能点清单

|  |  |
| --- | --- |
| **功能模块** | **功能点** |
| **用户功能** | **用户登录** |
| **修改密码** |
| **课程管理** | **新增课程信息** |
| **修改课程信息** |
| **删除课程信息** |
| **条件分页查询课程信息** |
| **课程表管理** | **录入课程表** |
| **修改课程表** |
| **成绩管理** | **录入成绩** |
| **修改成绩** |
| **统计成绩** |
| **用户管理** | **新增用户信息** |
| **修改用户信息** |
| **删除用户信息** |
| **条件分页查询用户信息** |

## 6.3 功能点描述

### 6.3.1 课程信息管理

管理员针对课程信息进行新增、删除、修改、条件分页查询。

### 6.3.2 课程表管理

管理员针对课程表信息进行录入、修改。学生和教师可以查询对应课程表。

### 6.3.3 成绩管理

管理员针对成绩信息进行录入、修改、条件分页查询、统计。学生可以查询对应的成绩和成绩统计。教师可以查询对应的成绩、录入成绩和成绩统计。

### 6.3.4 用户功能

用户可以选择对应的身份（管理员、学生、教师），输入对应的用户名密码进行登录，登录后也可以进行修改密码。

### 6.3.5 用户信息管理

管理员针对用户信息进行新增、删除、修改、条件分页查询。

# 二、可行性分析

# 1. 引言

## 1.1 标识

  电脑系统：Windows 10  
  编程软件：IntelliJ IDEA 2019、visual studio code  
  Web服务器：Nginx-1.14.8、Tomcat—7

## 1.2 背景

随着我国高等教育的发展，数字化校园将成为必然趋势，国内高校迫切需要提高教育工作的质量和效率，学生绩效管理是高校信息化管理的重要组成部分，与国外高校不同，他们一般都有规模更大、更稳定的技术团队来提供服务和技术支持。国内高校信息化建设起步较晚。由于整体信息化水平相对落后、资金短缺、理论体系不完善，国内高校的教务管理体制在机构设置、服务范围、服务质量、人员要求等方面与国外高校有一定差距。尽管困难重重，许多高校还是根据学校的规模和特点制定了学生绩效管理系统，使管理者从繁琐的日常工作中解放出来，提高了工作效率，实现了信息共享。因为还处于起步阶段，国内很多大学和很多软件公司都把大学的管理制度看过，但目前很多大学的学生绩效管理体系还存在很多问题，最重要的问题就是功能单一，一个体系是针对一个群体的。相关系统之间没有紧密有效的联系，例如，管理系统仅供学校管理人员使用，如果学生对自己的信息进行相关查询，则必须设计另一个系统，并且必须重新导入数据，这影响了信息的重复使用和共享。管理和维护也很不方便。

由于每年阅读的人数不断增加，关于学生的信息量也翻了一番，对各类学生的统计分析也越来越困难，面对如此复杂的工作，为了给大学生信息管理提供更高效、更实用的管理手段， 每个学生的基本信息在一个平台上统一管理，为学生信息的存储、计算、统计、分析和交流提供了更安全、更快捷的信息平台，同时减少了大量的人工操作。避免人工操作造成的错误，确保学生信息和数据的安全性和完整性，使学生管理员能够轻松正确地完成各项任务，为学生管理服务。

对于这种传统的数据收集和整理学生使用人工采集，而处理这些数据的任务已经变得极其困难，引入计算机代替人工操作也变得越来越迫切，引入成绩管理系统是实现学习自动化管理、标准化的必由之路。对于学校数量众多的学校来说，绩效管理制度是必不可少的。

学生成绩管理系统可以向所有师生以及教务处开放，不同的人可以登录获取不同的权限，使用不同的功能，获取不同的信息。

## 1.3 项目概述

为高校提供完整的学生成绩管理系统，节省学生成绩管理员以及教师和学生查询统计成绩所花的时间和精力。

由于该项目是新提出的，正处于需求分析、可行性分析等的前期准备，项目 开发、运行和维护还待完成前期工作后进行。

## 1.4 文档概述

  本文档用于团队内部商量项目，是决策是否执行的依据，不得外泄。文档内 含是项目的可行性分析，如：经济可行性、技术可行性、社会可行性。

# 2. 可行性分析的前提

## 2.1 项目的要求

  建议开发软件的基本要求:  
  （1）功能：该系统要求能根据需求实现课程表管理、成绩查询、成绩详情、课程管理、用户管理、账号管理等功能；  
  （2）性能：要求性能稳定，不能出现数据丢失等情况；  
  （3）输出：要求将各种输出数据以表格形式输出；  
  （4）输入：要求系统各模块具有添加新数据的功能；  
  （5）安全与保密要求：要求系统软件具有权限功能，使不同的用户具有不同的权限;防止数据外泄；  
  （6）与软件相关的其他系统：本软件应使用 Windows10 操作系统；  
  （7）完成期限：本网站应在这学期内设计完成。

## 2.2 项目的目标

保证信息的准确性和时效性，随时通过计算机对系统进行管理，使得学生信息处于最新状态，保证统计数据和分析的准确。降低管理成本，减轻劳动强度、提高工作效率、增加学生管理的透明度善和优化学生工作。把成绩管理系统和学生的查询系统有机的结合起来。方便管理于维护。

## 2.3 项目的环境、条件、假定和限制

  （1）建议开发软件运行的最短寿命为 5 年；  
  （2）经费来源和使用限制为自筹资金；  
  （3）法律和政策方面的限制：本学生成绩管理系统为兔子大学北京校区计算机学院软件工程课程设计软件开发小组所有；  
  （4）硬件：Intel i5 处理器，内存 512M，64 位操作系统；  
  （5）网络设备:具有 10 到 100 兆的网卡；  
  （6）软件：visual studio code、MySQL、Intellij IDEA；  
  （7）运行环境：本软件应使用 Linux 操作系统运行；  
  （8）开发环境：本软件应使用 Windows10 操作系统开发；  
  （9）建议开发软件投入使用的最迟时间：二零二三年六月。

## 2.4 进行可行性分析的方法

建议本应用的研究主要集中在拥有大量学生成绩数据需要管理的学校，以电子化的形式对学生成绩进行统一管理，通过调研相关高校，联系学生成绩的管理员、学生、教师，搜集意见以制定可行性分析方案。

# 3. 可选的方案

## 3.1 可选的系统方案一

 选用 Eclispe，eclispe 开发工具具有：

（1）非常适合 java 语言，开发、编写、查错、编译、帮助等各方面非常方 便；

（2）有中文版，上手较快；

（3）更新速度快；

（4）插件功能强大，免费；

（5）开源、免费。

技术采用单体架构，目前很多金融机构的架构是典型的单体式架构，一般由 反向代理服务器，数据库和应用组成，所有业务模块都打包在一个应用里面运行， 一般为了高可用考虑，应用至少会部署两个节点。单体式架构在业务简单的时候 有很多它自身的优点：  
  开发，测试简单；  
  部署简单，运维简单；  
  扩容简单，只要给应用加机器就行。

## 3.2 可选的系统方案二

  选用 IDEA 作为后端项目开发工具，VSCode 作为前端项目开发工具，拥有以 下优势:  
  （1）提示功能的快速、便捷；  
  （2）提示功能的范围比较广；  
  （3）好用的快捷键和代码模板；  
  （4）精准搜索。

## 3.3 选用的方案

  本系统初步采用方案二，当下，项目成员人数不足，精力有限。并且开发刚 刚起步，足以应对当下的访问量。但为了以后可扩充性，采用各种组件以及消息 中间件面向微服务架构，所以采用方案二。

# 4. 所建议的系统

## 4.1 对所建议的系统的说明

学生成绩管理系统包括课程表管理、成绩查询、成绩详情、课程管理、用户管理、账号管理等功能模块。  
  （1）用户可以通过 PC访问平台；   
  （2）对于 PC，H5 等客户端请求，首先请求 UI 层，渲染用户界面；  
  （3）客户端 UI 请求服务层获取进行具体的业务操作；  
  （4）服务层将数据层持久化到数据库；

## 4.3 影响

### 4.3.1 设备

  开发人员所有的设备和网络设备。

### 4.3.2 软件

  visual studio code、MySQL、Intellij IDEA。

### 4.3.3 运行

  运行环境：Linux 系统。

### 4.3.4 开发

  平台后端基于 Spring Boot 构建。平台前端基于 Vue.js 构建。

### 4.3.5 环境

  开发环境：Windows10，MySQL 8.0.19，Intellij IDEA，visual studio code。

## 4.4 局限性

  处理时间较长，随着对于未来日益增 长用户数量增加。无法满足用户数据收集，在数据资源更新和存储上必须扩容。

# 5. 经济可行性

## 5.1 投资

除了idea开发工具需要收费，其他开发工具均可以免费使用，均免费面向开发者，这是我们经济可行性最为关键的要素，但后期项目上线的云服务器是需要付费的，整体开发成本是比较低的，所涉及的技术栈也是开源免费的，几乎0成本，在开发工作结束后，是以出租的方式给第三方机构使用，一旦我们可以获得较高的客源的话，收入也是非常的可观。

## 5.2 预期的经济效益

预期会与高校、培训机构协商，前期以较低的价格租赁出去，等系统稳定了可以面向社会层面深入挖掘客源，增加预期效益。

## 5.3 市场预测

目前社会各行业都向互联网智能化靠拢，大部分管理工作都有系统实现，学生成绩管理系统也是其中之一，学生成绩大数据分析的实现少不了一个学生成绩管理系统，故市场前景还是比较广泛的。

# 6. 技术可行性

## 6.1 Java语言

Java作为企业级开发最常用的编程语言，其作为该系统的后端开发是非常合适的，加上Java代码的可维护性及运行效率，搭配SpringBoot框架和MyBatis框架协助开发，可以让开发效率大大翻倍。

## 6.2 MySQL

MySQL作为一款经久不衰的数据库，作为该系统的数据持久化工具是比较合理的，加上其强大的技术生态圈，以及编程延伸出来的各种ORM框架，对MySQL的操作是比较友好的。

## 6.3 HTML、JavaScript

HTML作为传统网站页面标记语言，用来建设网站是必不可少的。搭配javaScript编写的脚本可以完美的与后端服务器完成数据操作，这里再借助Vue.js框架针对系统进行模块化控制，大大加强代码效率及可维护性。

# 7 社会可行性

目前社会各行业都向互联网智能化靠拢，大部分管理工作都有系统实现，本系统也是其中之一，智能系统的搭建少不了一个功能完善的管理系统，故社会发展前景还是很客观的。

针对用户使用，该系统功能清晰，界面简洁，功能齐全，学习成本低。基本上上手就可以直接使用，针对各种角色和用户都有清晰的导航栏和文字提示。

# 8. 结论

针对该系统进行经济可行性、技术可行性、社会可行性等进行分析，加上各种调研，该系统的设计与实现是着实可行的。

# 三、概要设计

# 1. 引言

## 1.1目的

该文档的目的是描述学生成绩管理系统的概要设计，主要内容包括系统功能简介、系统结构设计、模块设计和界面设计等。

本文档的预期读者包括：产品设计者、编程人员、系统用户。

## 1.2 范围

### 1.2.1 系统目标

  开发一个满足学生成绩管理的网页版管理系统。方便高校学生成绩管理员对高校内的学生、教师、课程、成绩进行分类管理、更加高效地维护学生成绩信息，方便教师查看任课信息，操作学生成绩。方便学生查询课程表和成绩。

### 1.2.2 主要软件需求

该系统的主要功能包括：课程表管理、成绩查询、成绩详情、课程管理、用户管理、账号管理。

### 1.2.3 软件设计约束、限制

数据安全性：系统应具备严格的数据安全措施，包括对学生和教师信息的保密性、数据备份和恢复机制，以及防止非授权用户访问系统的控制措施。

系统性能：系统应具备高性能和稳定性，能够处理大量的学生和课程数据，并能够快速响应用户的请求和查询。

用户权限控制：系统应设定不同的用户角色和权限，例如管理员、教师和学生，以确保只有授权的用户能够进行相应的操作和访问相关的信息。

数据一致性：系统应保证学生成绩数据的准确性和一致性，防止因为并发操作或系统故障导致数据错误或冲突。

界面友好性：系统界面应简洁、直观，并且易于使用，以提高用户的使用体验和效率。

扩展性：系统应具备良好的扩展性，能够方便地添加新的功能模块或支持新的需求，以满足日益增长的学生和课程管理需求。

兼容性：系统应能够与现有的教务管理系统或其他相关系统进行数据交换和集成，以保证数据的一致性和流畅的信息传递。

可靠性和可恢复性：系统应具备故障自恢复的能力，能够在系统故障或意外中断后快速恢复，并能够记录异常事件和错误日志，以便进行故障排查和修复。

可维护性：系统应采用模块化的设计和规范的编码风格，以便于系统的维护和升级，并且应提供相关的文档和培训支持，以便于后续的系统维护工作。

法律和合规性：系统应符合相关的法律法规和教育部门的规定，包括个人信息保护、学生成绩公平性等方面的要求，以确保系统的合法性和公正性。

总之，设计学生成绩管理系统的软件需要考虑数据安全性、系统性能、用户权限控制、数据一致性、界面友好性、扩展性、兼容性、可靠性和可恢复性、可维护性以及法律和合规性等方面的约束和限制。。

## 1.3 参考资料

[1] 窦万峰.软件工程与实践[M].北京：机械工业出版社,2017.

[2] 冯畅,焦智成,秦朋.关于计算机软件开发的JAVA编程语言研究[J].艺术科技,2016

## 1.4版本信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **修改编号** | **修改日期** | **修改后版本** | **修改位置** | **内容概述** |
|  |  |  |  |  |

# 2.数据设计

## 2.1数据对象和形成的数据结构

### 2.1.1管理员信息表（admin）

该数据结构包括管理员id、用户名、密码、真实姓名、等级、归属学校、邮箱、联系电话、性别、创建时间。

### 2.1.2 课程基本信息表（course）

该数据结构包括课程基本信息id、课程名、学分、满分、课时、届时、学期、类别（1必修，2选秀）、专业、创建时间。

### 2.1.3 课程扩展信息表（course\_info）

该数据结构包括课程扩展信息id、课程基本信息id、课程开始时间、课程结束时间、教室信息、专业信息。

### 2.1.4 专业表（profession）

该数据结构包括专业id、专业名称。

### 2.1.5 页面主题切换表（silent）

该数据结构包括主题id、页面主题状态（0正常，2静默模式）。

### 2.1.6 学生信息表（student）

该数据结构包括学生id、用户名、密码、真实姓名、等级、学校、入学时间、邮箱、专业、年级、联系电话、性别、创建时间。

### 2.1.7 学生课程关联表（student\_course）

该数据结构包括关联id、学生id、课程id、课程名称、成绩、几点、学分、学期、届时。

### 2.1.8 教师信息表（teacher）

该数据结构包括教师id、用户名、密码、真实姓名、等级、学校名称、邮箱、联系电话、性别、创建时间。

### 2.1.9 教师课程关联表（teacher\_course）

该数据结构包括关联id、教师id、用户名、课程名称、专业、年级、学期、课程id。

### 2.1.10 排课时间信息表（timetable）

该数据结构包括排课时间信息id、对应排课表id、专业、年级、学期。

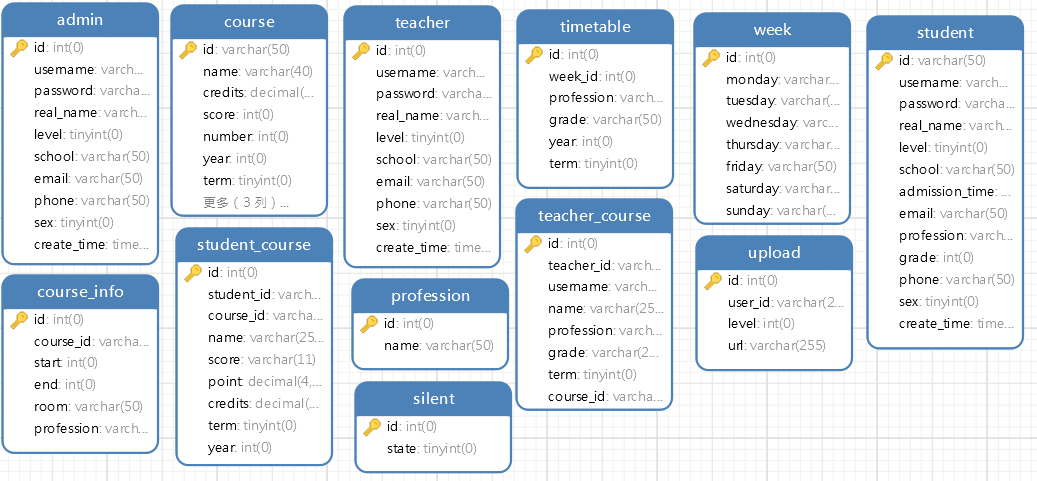
### 2.1.11 文件管理表（upload）

该数据结构包括文件id、上传用户id、等级、文件地址。

### 2.1.12 排课表（week）

该数据结构包括排课信息id、周一课程、周二课程、周三课程、周四课程、周五课程、周六课程、周日课程。

## 2.2数据库结构



## 2.3 全局数据

该系统需要的全局变量有：

页面主题（包括主题标记）

用户个人资料（包括用户id，姓名）

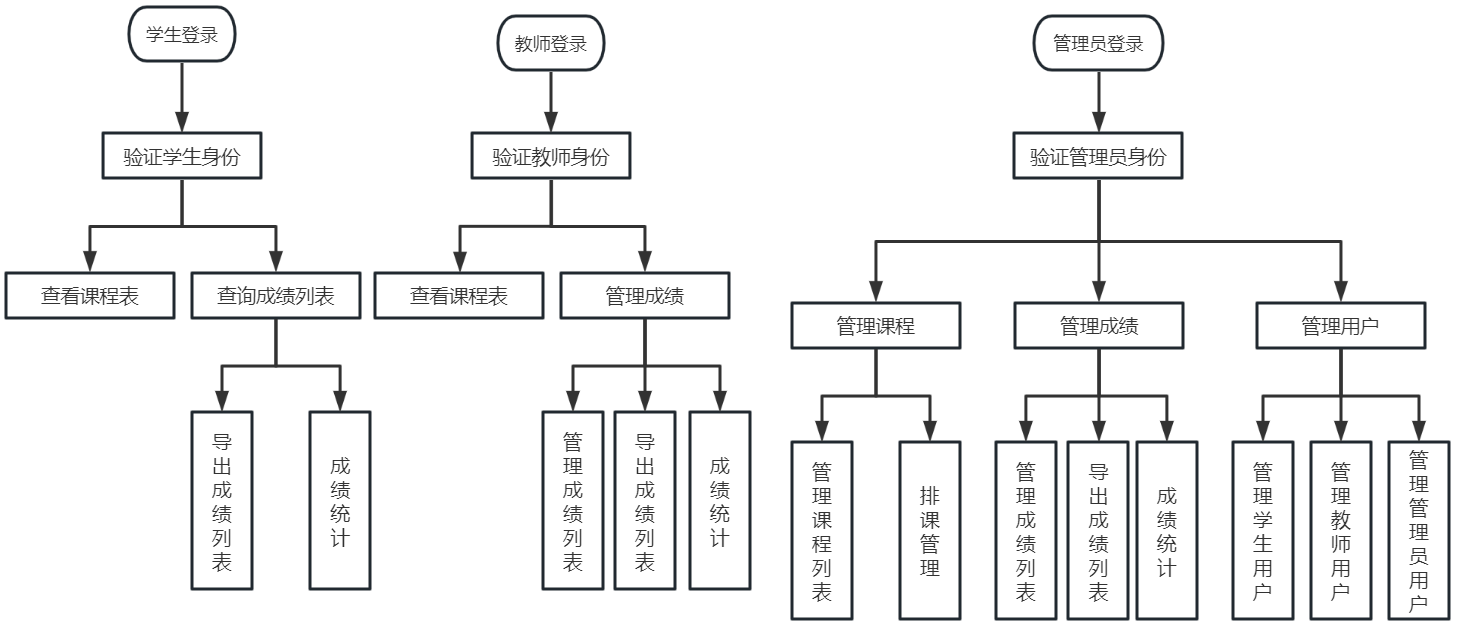
# 3 体系结构设计

## 3.1数据和控制流复审

学生成绩管理系统的数据流主要包括学生信息输入、课程信息输入、成绩录入和查询等功能，通过管理员或教师输入和查询相关信息。控制流程方面，用户需要登录系统，系统根据用户角色和权限限制用户的操作范围，管理员可以管理学生和课程信息，教师可以录入成绩和查询相关信息。系统还能进行成绩统计和分析，提供相应的报表和图表。

## 3.2 得出的程序结构

根据复审的数据流图，逐步得出软件的逻辑结构组成。利用优化思想，对软件结构图进行优化设计，得出模块层次结构适中的软件结构图。如下所示：



# 4.界面设计

## 4.1 人机界面规约

### 4.1.1 课程信息管理界面

该界面可根据用户输入的课程名、专业检索课程信息。并针对课程信息进行增、删、改。

### 4.1.2 课程表界面

该界面可获取课程表信息。管理员可以针对课程信息进行录入、修改。教师和学生可以查看对应的课程表信息。

### 4.1.3 成绩管理界面

该界面可以获取成绩列表，管理员和教师可以录入或修改成绩，学生可以查询成绩。

### 4.1.4 用户管理界面

该界面可以为管理员用户提供获取学生用户、教师用户、管理员用户信息的功能。并提供增、删、改。

## 4.2人机界面设计规约

给出界面序列关系，以及每个界面的操作规则和处理规则。学生成绩管理系统有以下几种界面序列。

登录首界面：可以选择对应的角色进行用户登录。

学生的界面：进入首页——查看对应课程成绩统计——查询对应课程表——查询成绩。

教师的界面：进入首页——查看对应课程成绩统计——查询对应课程表——查询、录入或修改成绩。

管理员的界面：登录—进入首页——查看所有课程成绩统计——课程管理——成绩管理——课程表管理——用户管理。

## 4.3 内部模块

登录模块

课程管理模块

课程表管理模块

成绩管理模块

用户管理模块

# 5、模块过程设计

## 5.1 登录模块

验证身份。正确即登入，错误即拒绝接入，退出。

## 5.2 课程管理模块

分页查询界面：可以根据条件进行查询。

新增课程信息界面：填入课程名、学分、课时、专业等必填项可新增，否则提示新增失败。

修改课程信息：填入课程名、学分、课时、专业等必填项可修改，否则提示修改失败。

删除课程：可以删除对应课程。

## 5.3 课程表（排课）模块

查询界面：可以根据权限自动筛选出课程表。

录入课程：填入课程等必填项可录。

修改排课：填入课程等必填项可修改，否则置空。

## 5.4 成绩管理模块

分页查询界面：可以根据条件进行查询。

录入成绩信息界面：填入成绩等必填项可录入，否则提示录入失败。

修改成绩信息：填入成绩等必填项可修改，否则提示修改失败。

## 5.5 用户信息管理模块

### 5.5.1 学生用户管理模块

分页查询界面：可以根据条件进行查询。

新增用户界面：填入姓名、学校、专业、班级、入学时间、电子邮箱、手机号码等必填项可新增，否则提示新增失败。

修改用户信息：填入姓名、学校、专业、班级、入学时间、电子邮箱、手机号码等必填项可修改，否则提示修改失败。

### 5.5.2 教师用户管理模块

分页查询界面：可以根据条件进行查询。

新增用户界面：填入姓名、学校、电子邮箱、手机号码等必填项可新增，否则提示新增失败。

修改用户信息：填入姓名、学校、电子邮箱、手机号码等必填项可修改，否则提示修改失败。

### 5.5.3 管理员用户管理模块

分页查询界面：可以根据条件进行查询。

新增用户界面：填入姓名、学校、电子邮箱、手机号码等必填项可新增，否则提示新增失败。

修改用户信息：填入姓名、学校、电子邮箱、手机号码等必填项可修改，否则提示修改失败。

# 四、详细设计

# 1 引言

## 1.1 编写目的

本报告的目的是对学生成绩管理系统进行详细设计说明，以便用户及项目开发人员了解产品详细的设计与实现，为开发人员提供开发参考书。以下叙述将结合文字描述、伪代码，图表等来描述学生成绩管理系统的详细设计和相关的模块描述。本报告的预期读者有客户、项目经理、开发人员以及跟该项目相关的其他人员。

## 1.2 项目背景

高校学生成绩管理系统由兔子大学北京校区XXX老师提出，由XXX、XXX等人进行后续的开发和实施工作，该项目适用于高校管理员、教师、学生。

## 1.3 参考资料

[1] 窦万峰.软件工程方法与实践(第三版).北京：机械工业出版社，2016

[2] 王珊/萨师煊.数据库系统概论(第五版).北京：高等教育出版社，2014

# 2 总体设计

## 2.1 需求概述

(1) 课程信息的管理，包括课程信息的条件查询、录入、修改和删除。

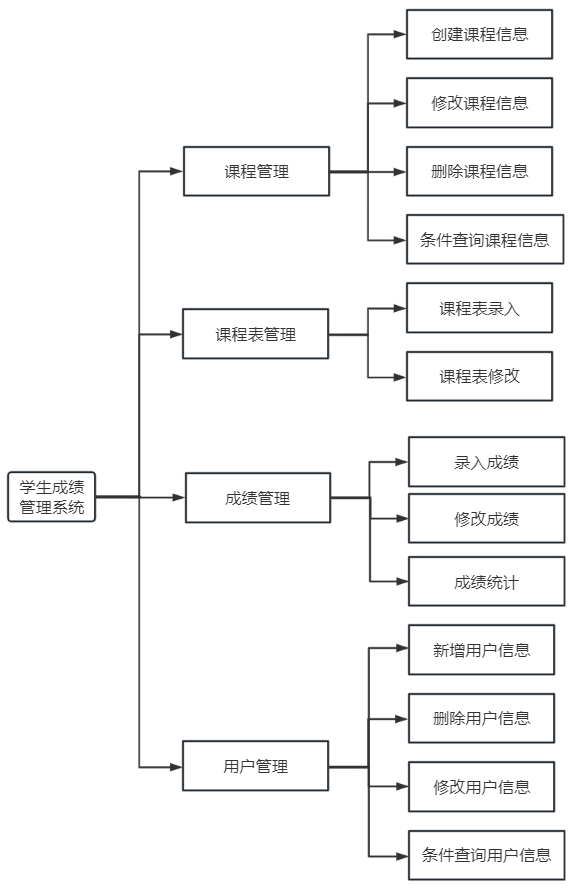
(2) 课程表的管理，包括排课、录入课程表、修改课程表。

(3) 用户信息管理。包括对系统管理员、教师和学生的信息进行维护管理，可以新增、修改、删除和条件分页查询用户信息。

(4) 成绩信息管理，可以查询对应成绩信息，教师和管理员可以录入和修改成绩。

(5) 用户登录，读者可以通过登录页面注册新账号，管理员、教师、学生可以使用用户名和密码登录系统。

## 2.2 软件结构



        学生成绩管理系统的软件结构图

描述：学生成绩管理系统向下划分为课程管理、课程表管理、成绩管理、用户管理四个模块。课程管理主要负责对课程的信息进行管理，以及管理员和教师进行课程的查询；课程表管理主要关联教师排课、任课信息、学生获取上课信息；成绩管理中，管理员和教师可以录入或修改对应权限的成绩信息，学生可以查询对应的成绩信息；用户信息管理可以针对管理员、教师和学生的信息进行管理。

# 3 模块设计

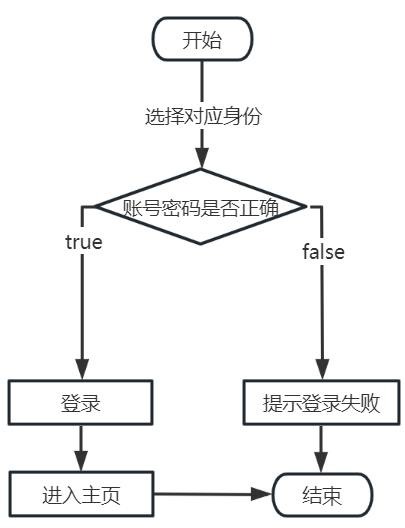
## 3.1 模块基本信息

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 编号 | 设计者 | 所在文件 |
| 课程管理 | 001 | 兔子 | com/rabbiter/sms/controller/Course/CourseController.java |
| 课程表管理 | 002 | 兔子 | com/rabbiter/sms/controller/Timetable/TimetableController.java |
| 成绩管理 | 003 | 兔子 | com/rabbiter/sms/controller/Score/ScoreController.java |
| 用户管理 | 004 | 兔子 | com/rabbiter/sms/controller/User/\*.java |

## 3.2 功能概述

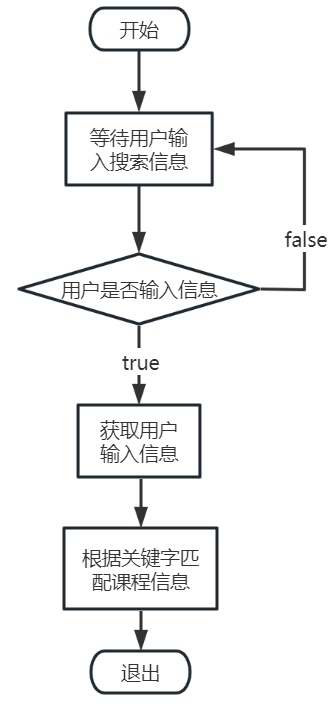
|  |  |  |
| --- | --- | --- |
| 名称 | 编号 | 说明 |
| 课程管理 | 001 | 对课程的信息进行管理，以及管理员和教师进行课程的查询。 |
| 课程表管理 | 002 | 关联教师排课、任课信息、学生获取上课信息，管理员可以针对课程表信息进行编辑，实现排课功能。 |
| 成绩管理 | 003 | 管理员和教师可以录入或修改对应权限的成绩信息，学生可以查询对应的成绩信息 |
| 用户管理 | 004 | 针对管理员、教师和学生的信息进行管理 |

## 3.3 模块处理逻辑



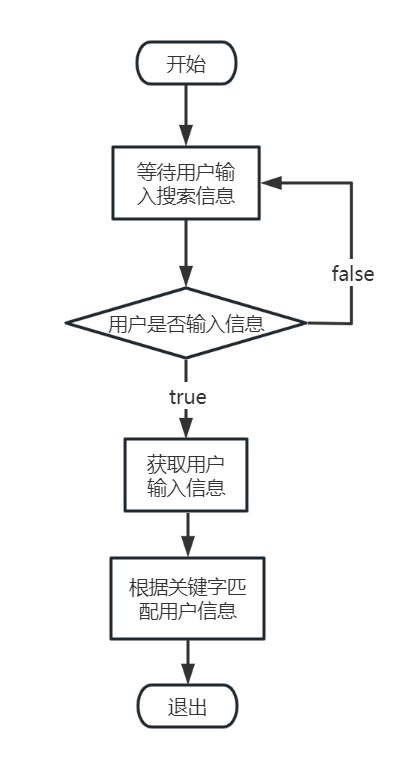
用户登录系统程序流程图

图为用户登录系统的程序流程图，选择登录，已经存在系统的管理员、教师或者学生可以直接进行登录，登录之后进入主页面。



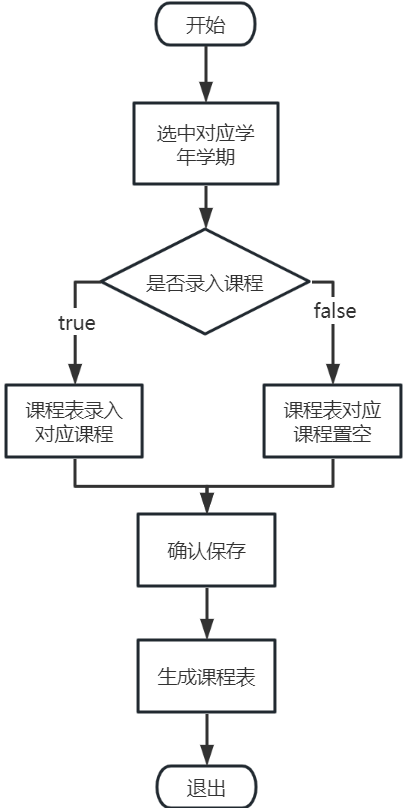
查询课程信息程序流程图

图为查询课程信息的程序流程图，该模块首先进入等待用户输入搜索信息状态。如果用户已经输入则获取用户输入并根据输入的搜索关键字在课程信息表中匹配课程信息，最后退出。如果用户未输入则模块继续处于等待用户输入搜索信息状态。



查询用户信息程序流程图

图为查询用户信息的程序流程图，该模块首先进入等待用户输入搜索信息状态。如果用户已经输入则获取用户输入并根据输入的搜索关键字在课程信息表中匹配学生用户、教师用户、管理员用户的信息，最后退出。如果用户未输入则模块继续处于等待用户输入搜索信息状态。

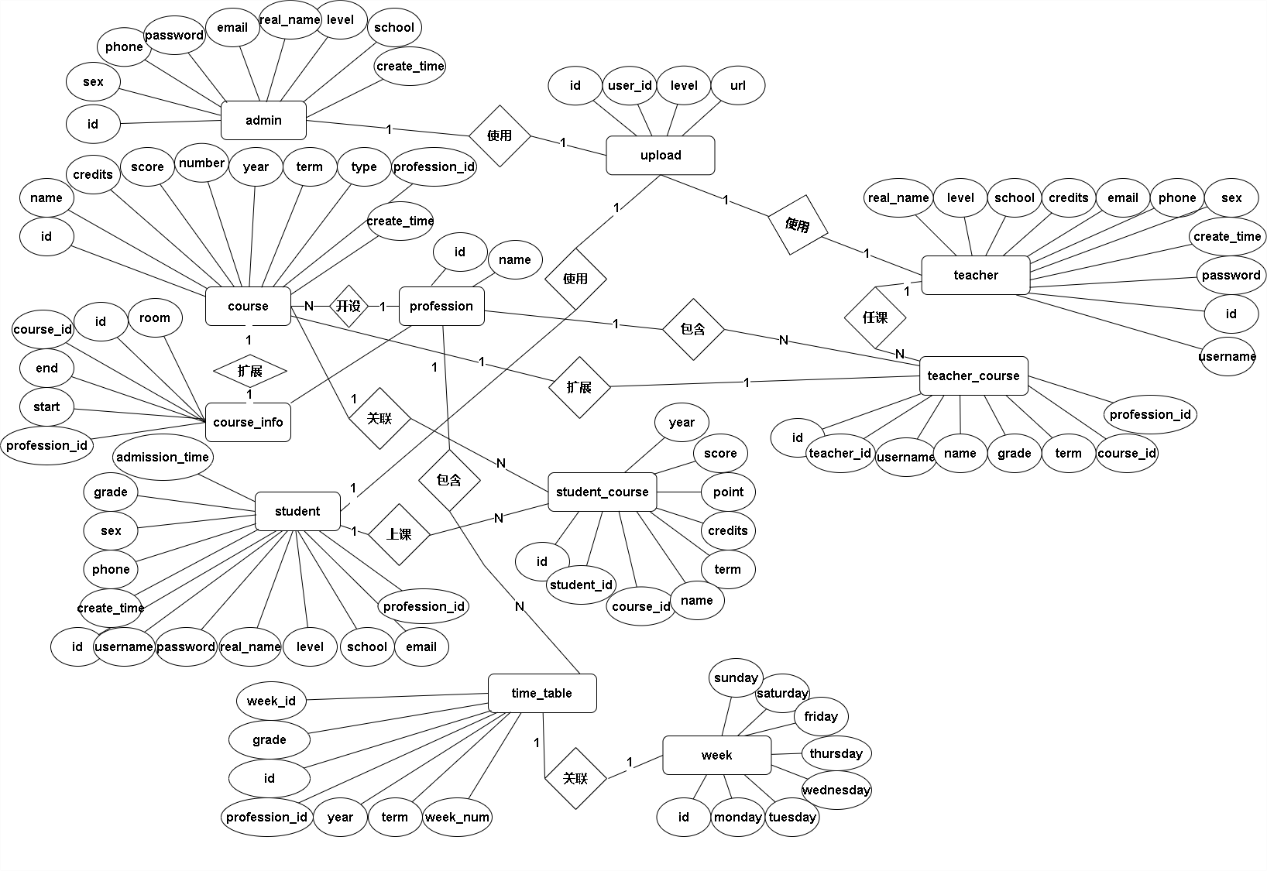


录入课程表（排课）程序流程图

图为录入课程表程序流程图，管理员具有课程表编辑和排课的权限，登录系统后，选择对应的学年和学期条件，系统回显课程表信息，管理员可以选中对应的课程去排课，未选中课程则对应课程表位置留空，确认保存，则可以生成最终课程表。

# 4 数据库设计

## 4.1 E-R图



## 4.2 表设计

### 4.2.1 管理员信息表

|  |  |  |
| --- | --- | --- |
| 数据名称 | 数据类型 | 数据描述 |
| Id | Int | 管理员id |
| Username | Varchar | 用户名 |
| Password | Varchar | 密码 |
| Real\_name | Varchar | 姓名 |
| Level | Tinyint | 级别 |
| School | Varchar | 所属学校 |
| Email | Varchar | 邮箱 |
| Phone | Varchar | 联系电话 |
| Sex | Tinyint | 性别 |
| Create\_time | timestamp | 创建时间 |

### 4.2.2 课程基本信息表

|  |  |  |
| --- | --- | --- |
| 数据名称 | 数据类型 | 数据描述 |
| Id | Varchar | 课程基本信息id |
| Name | Varchar | 课程名称 |
| Credits | Decimal | 学分 |
| Score | Int | 分数 |
| Number | Int | 课时 |
| Year | Int | 学年 |
| Term | Tinyint | 学期 |
| Type | Tinyint | 类型。1必修课 2选修课 |
| Profession | Varchar | 专业 |
| Create\_time | Timestamp | 创建时间 |

### 4.2.3 课程扩展信息表

|  |  |  |
| --- | --- | --- |
| 数据名称 | 数据类型 | 数据描述 |
| Id | Varchar | 课程扩展信息id |
| Course\_id | Varchar | 课程基本信息id |
| Start | Int | 开始时间 |
| End | Int | 结束时间 |
| Room | Varchar | 教室信息 |
| Profession | Varchar | 专业 |

### 4.2.4 专业信息表

|  |  |  |
| --- | --- | --- |
| 数据名称 | 数据类型 | 数据描述 |
| Id | Varchar | 专业信息id |
| Name | Varchar | 专业名称 |

### 4.2.5 页面主题表

|  |  |  |
| --- | --- | --- |
| 数据名称 | 数据类型 | 数据描述 |
| Id | Varchar | 页面主题信息id |
| State | Tinyint | 主题标记。0正常模式 1静默模式 |

### 4.2.6 学生信息表

|  |  |  |
| --- | --- | --- |
| 数据名称 | 数据类型 | 数据描述 |
| Id | Varchar | 学生信息id |
| Username | Varchar | 用户名 |
| Password | Varchar | 密码 |
| Real\_name | Varchar | 姓名 |
| Level | Tinyint | 级别 |
| School | Varchar | 学校 |
| Admission\_time | Varchar | 入校时间 |
| Email | Varchar | 电子邮箱 |
| Profession | Varchar | 专业 |
| Grade | Int | 年级 |

### 4.2.7 学生-课程关联表

|  |  |  |
| --- | --- | --- |
| 数据名称 | 数据类型 | 数据描述 |
| Id | int | 关联id |
| Student\_id | varchar | 学生id |
| Course\_id | Varchar | 课程id |
| Name | Varchar | 课程名称 |
| Score | Varchar | 分数 |
| Point | Decimal | 绩点 |
| Credits | Decimal | 学分 |
| Term | Tinyint | 学期 |
| Year | Int | 学年 |

### 4.2.8 教师信息表

|  |  |  |
| --- | --- | --- |
| 数据名称 | 数据类型 | 数据描述 |
| Id | Varchar | 教师信息id |
| Username | Varchar | 用户名 |
| Password | Varchar | 密码 |
| Real\_name | Varchar | 姓名 |
| Level | Tinyint | 级别 |
| School | Varchar | 学校名称 |
| Email | Varchar | 邮箱 |
| Phone | Varchar | 联系电话 |
| Sex | Tinyint | 性别 |
| Create\_time | Timestamp | 创建时间 |

### 4.2.9 教师-课程关联表

|  |  |  |
| --- | --- | --- |
| 数据名称 | 数据类型 | 数据描述 |
| Id | int | 关联id |
| Teacher\_id | varchar | 教师id |
| Username | Varchar | 教师用户名 |
| Name | Varchar | 课程名称 |
| Profession | Varchar | 专业 |
| Grade | Varchar | 年级 |
| Term | Tinyint | 学期 |
| Course\_id | Varchar | 课程id |

### 4.2.10 排课时间表

|  |  |  |
| --- | --- | --- |
| 数据名称 | 数据类型 | 数据描述 |
| Id | Varchar | 排课时间id |
| Week\_id | Int | 周课程表id |
| Profession | Varchar | 专业 |
| Grade | Varchar | 年级 |
| Year | Int | 学年 |
| Term | Tinyint | 学期 |

### 4.2.11 文件信息表

|  |  |  |
| --- | --- | --- |
| 数据名称 | 数据类型 | 数据描述 |
| Id | Int | 文件信息id |
| User\_id | Varchar | 上传用户id |
| Level | Int | 级别 |
| url | Varchar | 文件地址 |

### 4.2.12 周课程表信息表

|  |  |  |
| --- | --- | --- |
| 数据名称 | 数据类型 | 数据描述 |
| Id | Varchar | 周课程表信息id |
| Monday | Varchar | 周一课程 |
| Tuesday | Varchar | 周二课程 |
| Wednesday | Varchar | 周三课程 |
| Thurday | Varchar | 周四课程 |
| Friday | Varchar | 周五课程 |
| Saturday | Varchar | 周六课程 |
| Sunday | Varchar | 周日课程 |

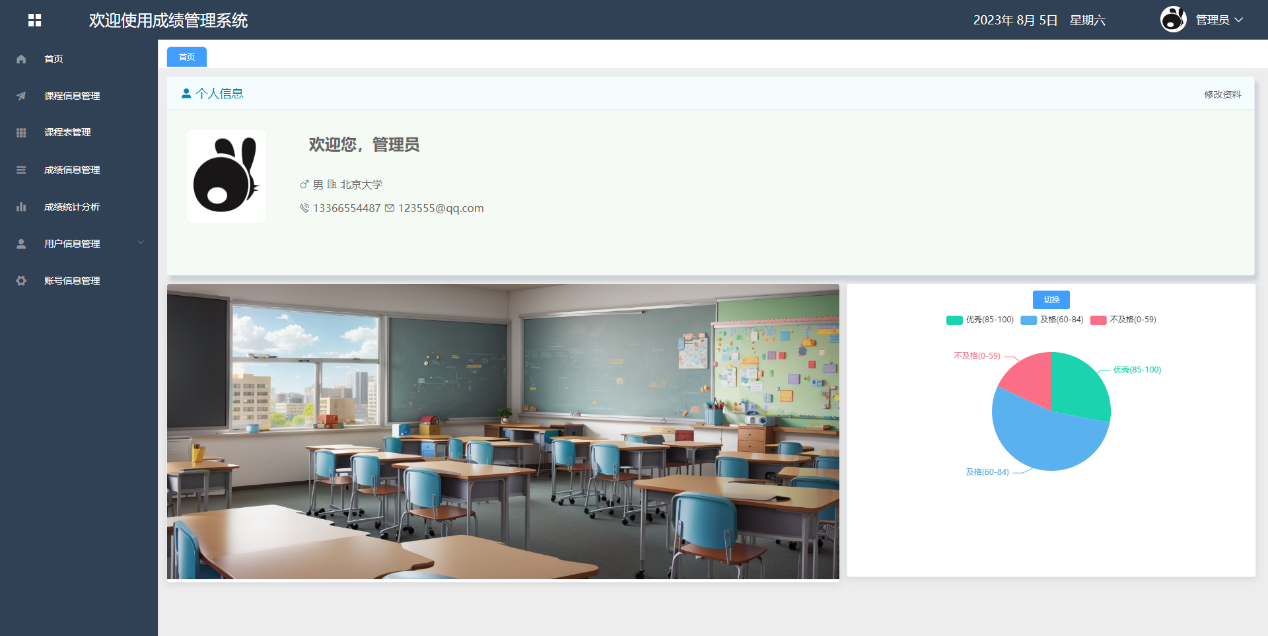
# 5 接口设计

## 5.1 外部接口

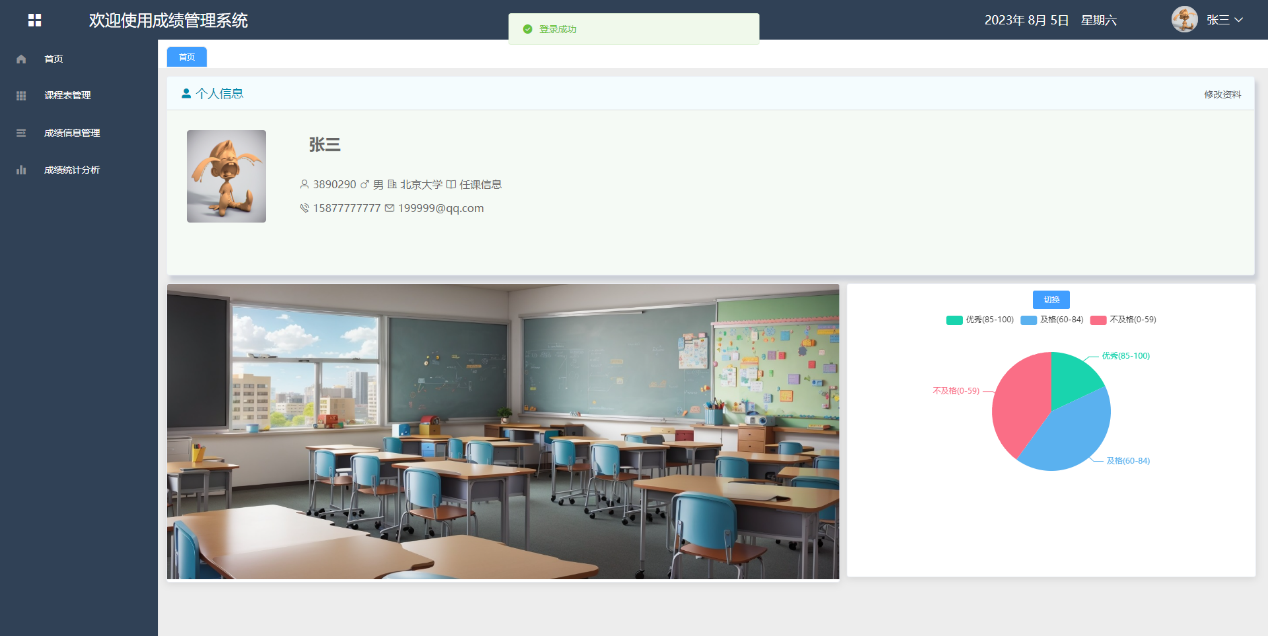
### 5.1.1 登录界面



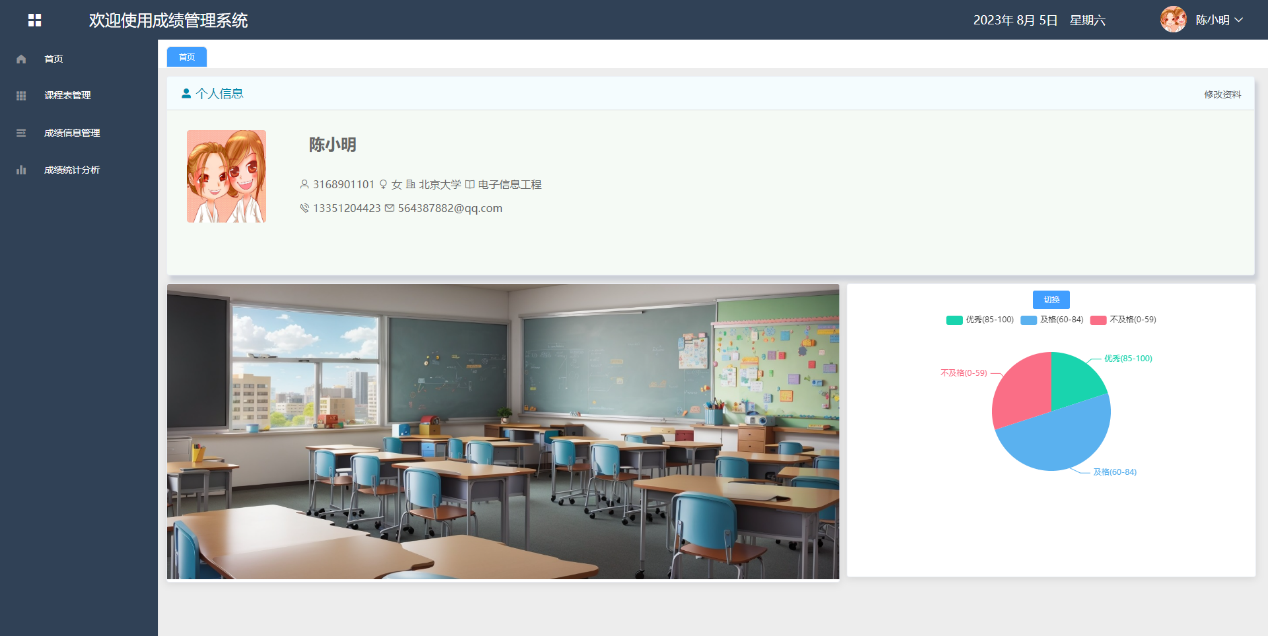
5.1.2管理员首页



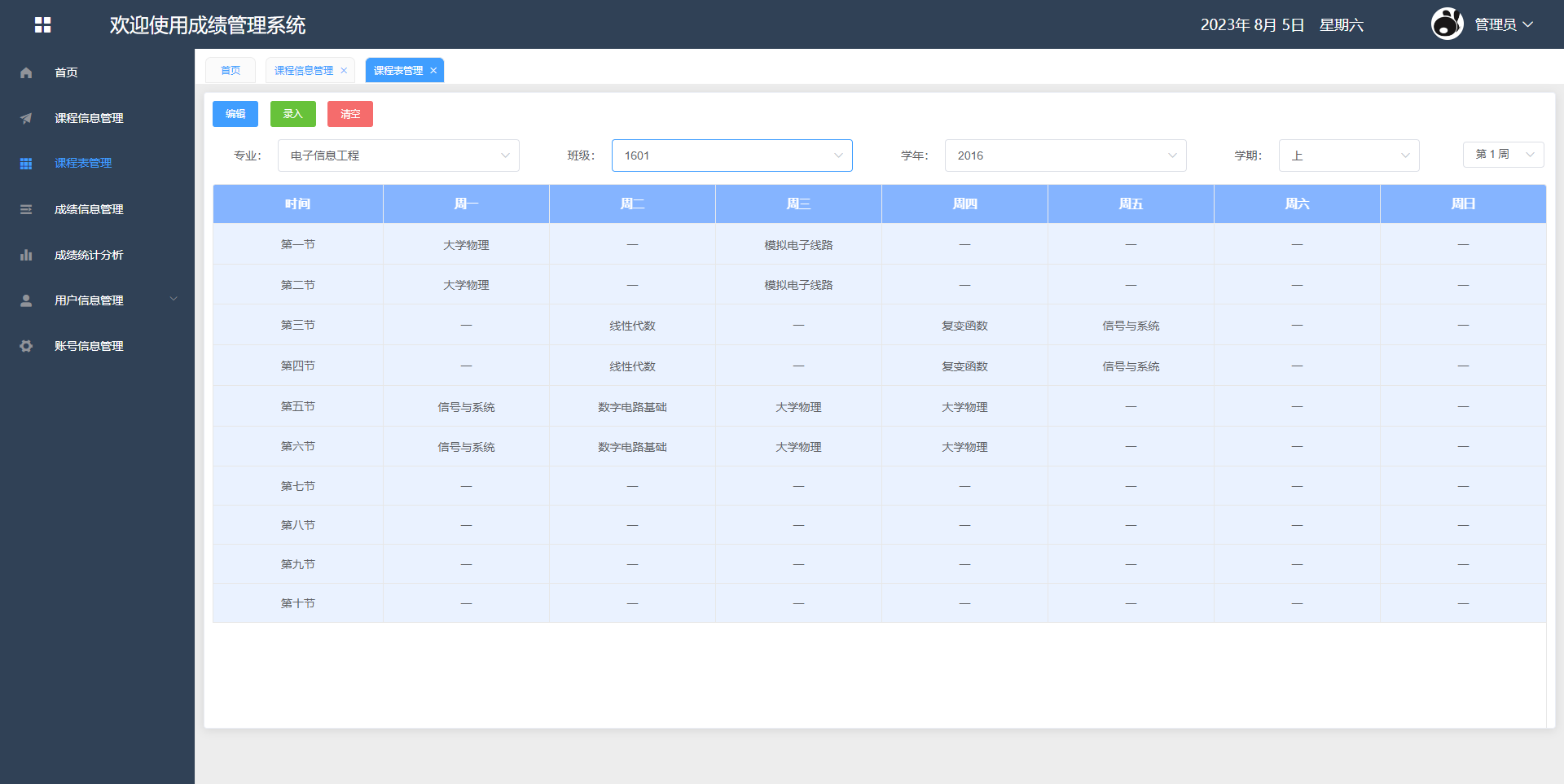
5.1.3 教师首页



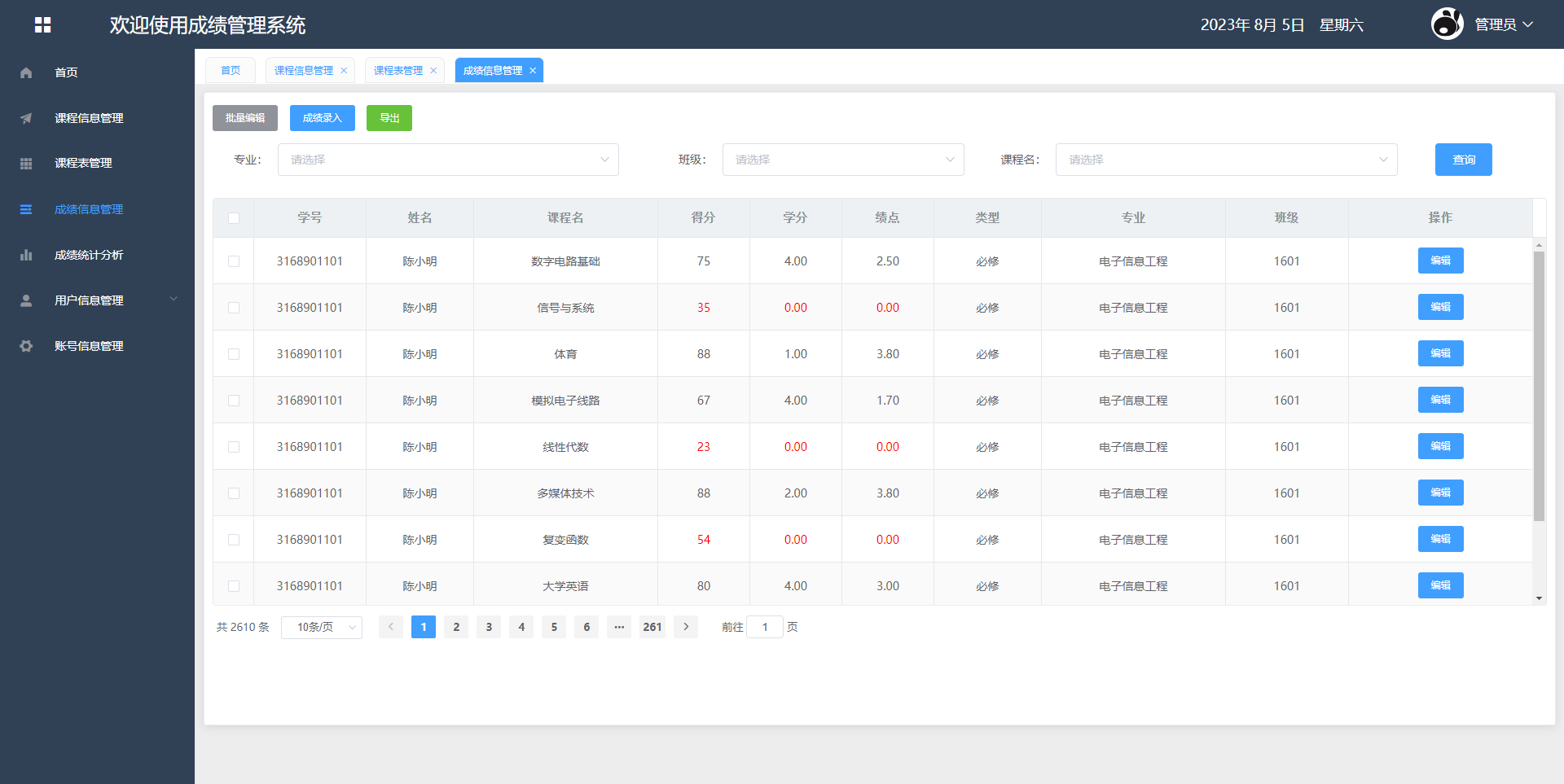
5.1.4 学生首页



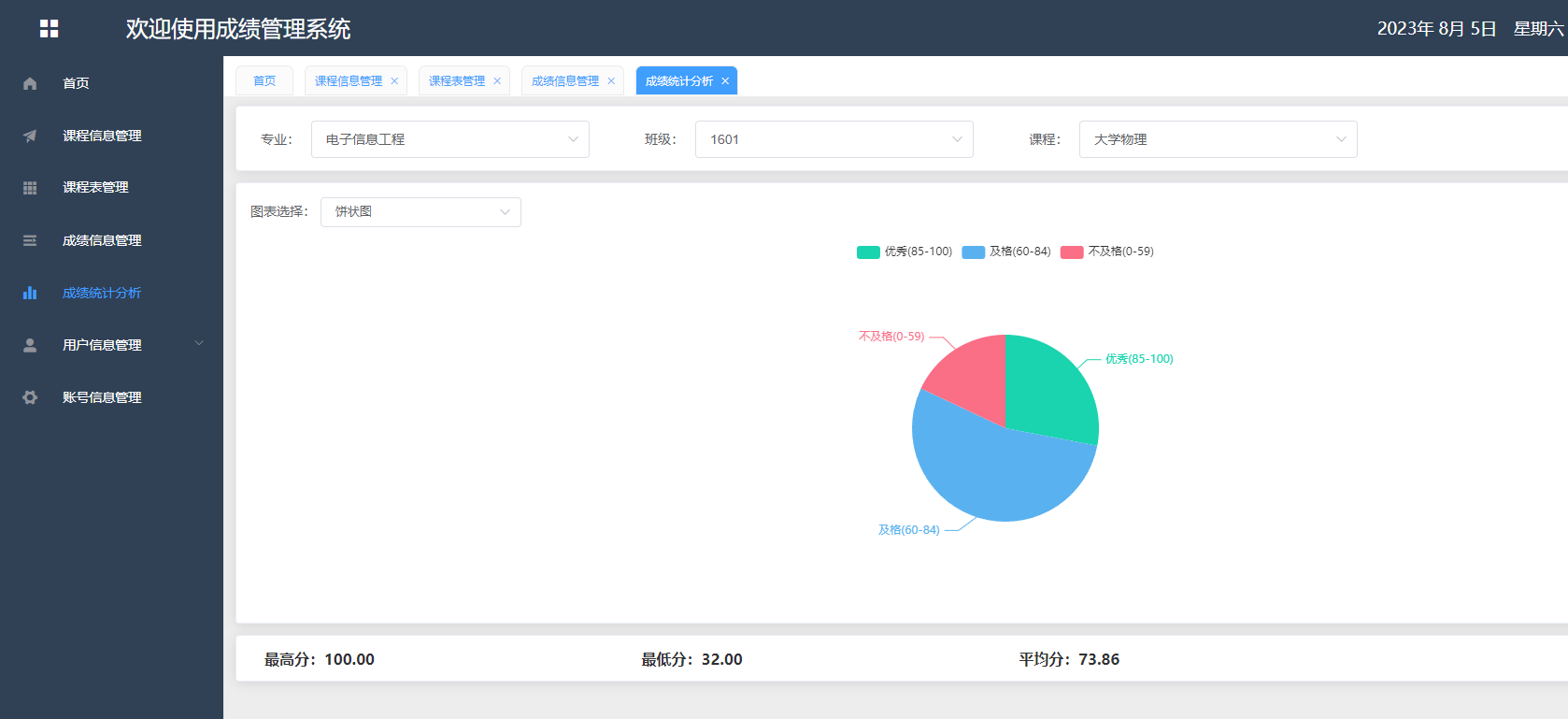
### 5.1.5 课程表



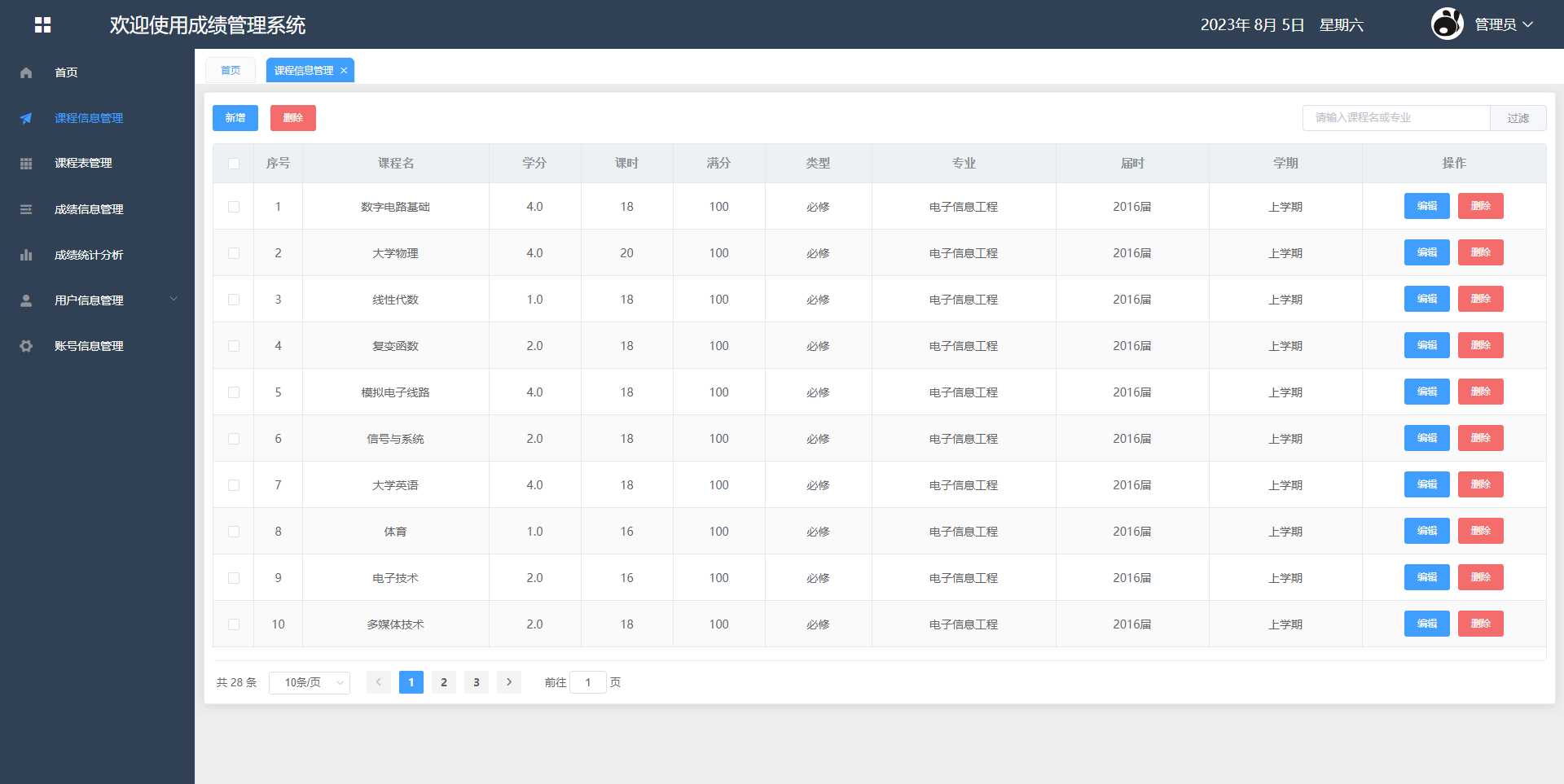
### 5.1.6 成绩查询界面



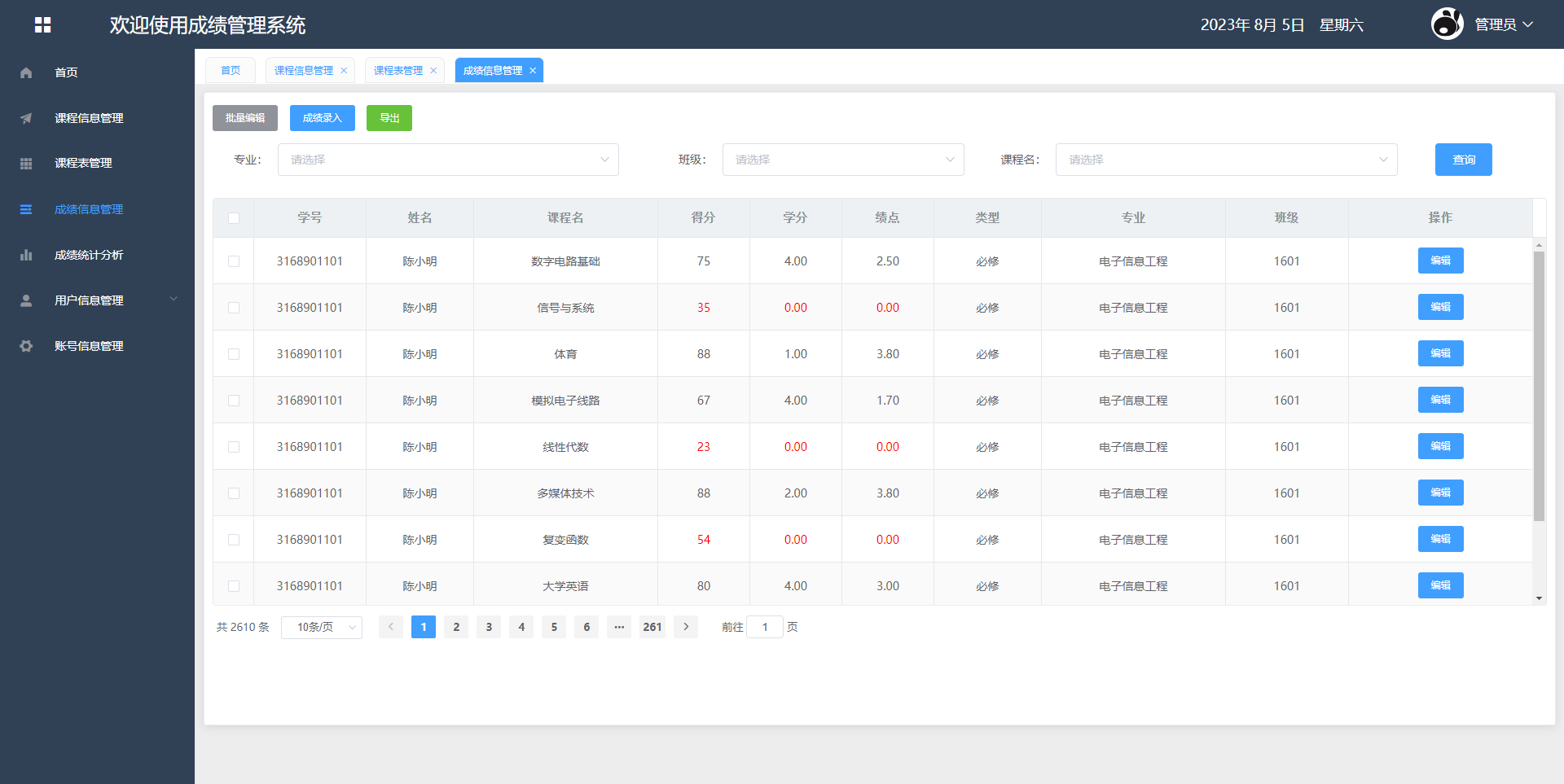
### 5.1.7 成绩统计界面



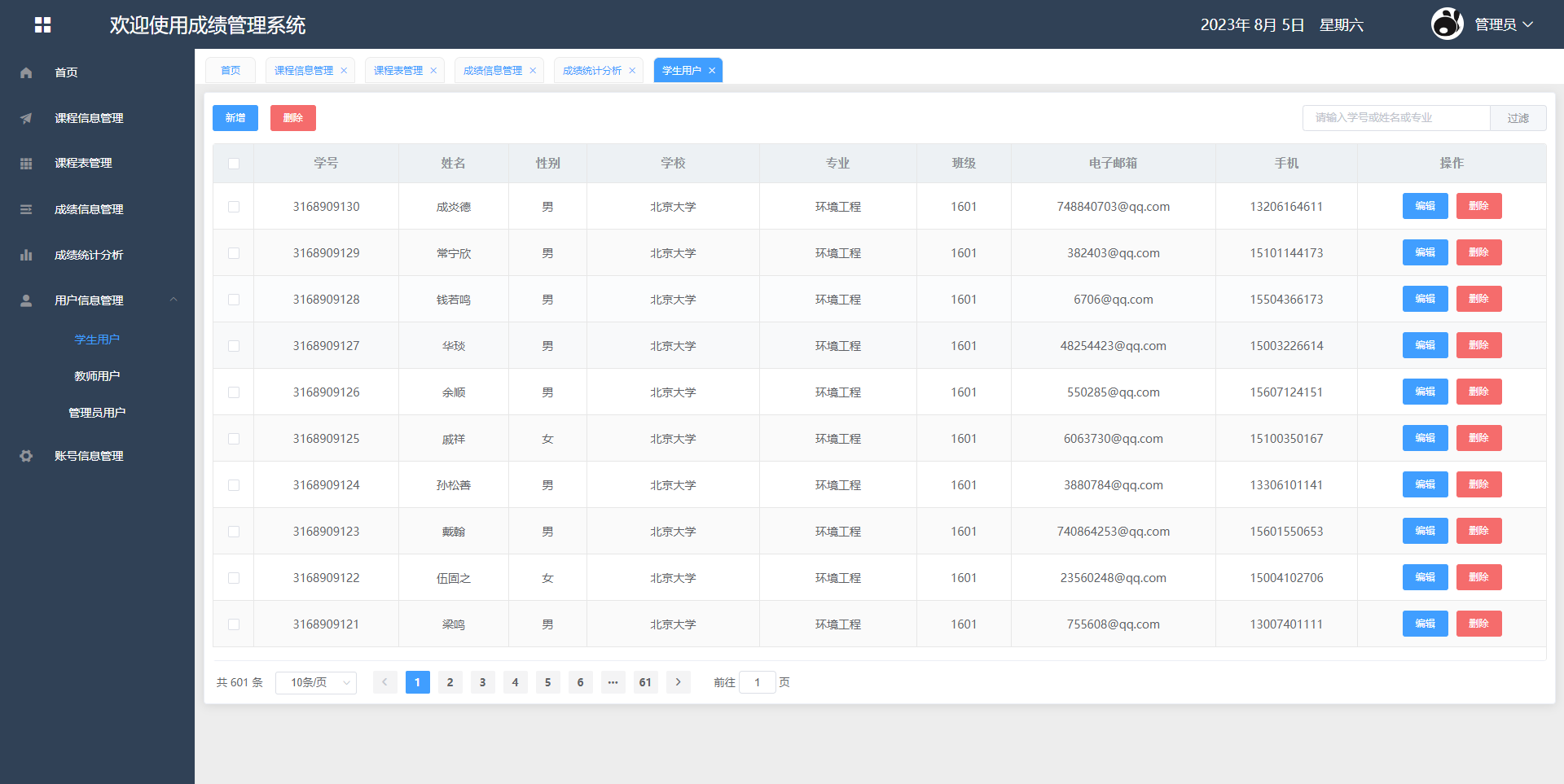
### 5.1.8 课程管理



### 5.1.9 成绩管理



### 5.1.10 用户管理



## 5.2 内部接口

### 5.2.1 登录接口

@GetMapping("/login")  
@PassToken  
public User getStudentInfo(@RequestParam Map<String, Object> condition) {  
 Map<String, Object> map = new HashMap<>();  
 map.put("username", condition.get("username").toString());  
 map.put("password", condition.get("password").toString());  
 map.put("level", condition.get("level"));  
 User user = userService.getStudentInfo(map);  
 String token = userService.getToken(user, 24 \* 60 \* 60 \* 1000);  
 String refreshToken = userService.getToken(user, 24 \* 60 \* 60 \* 1000); *// 有效期一天* user.setToken(token);  
 user.setRefreshToken(refreshToken);  
 return user;  
}

### 5.2.2 课程管理接口

@PostMapping  
public void addCourse(@RequestBody Course course) {  
 courseService.addCourse(course);  
}  
  
@DeleteMapping("/{ids}")  
public void delete(@PathVariable("ids") Integer[] ids) {  
 List<Integer> idsList = Arrays.*asList*(ids);  
 courseService.delete(idsList);  
}  
  
@PutMapping  
public void update(@RequestBody Course course) {  
 courseService.update(course);  
}  
  
@GetMapping("/getCourseList")  
private PagingResult<Course> getCourseList(@RequestParam Map<String, Object> condition,  
 @RequestParam(required = false, name = "$limit", defaultValue = "10") Integer limit,  
 @RequestParam(required = false, name = "$offset", defaultValue = "0") Integer offset) {  
 RowBounds rowBounds = new RowBounds(offset, limit);  
 return courseService.getCourseList(rowBounds, condition);  
}  
  
@GetMapping("/getCourseByMap")  
private List<Course> getCourseByMap(@RequestParam Map<String, Object> condition) {  
 return courseService.getCourseByMap(condition);  
}

### 5.2.3 成绩管理接口

@GetMapping("/getCourseList")  
public PagingResult<Course> getCourseList(@RequestParam Map<String, Object> condition,  
 @RequestParam(required = false, name = "$limit", defaultValue = "10") Integer limit,  
 @RequestParam(required = false, name = "$offset", defaultValue = "0") Integer offset) {  
 RowBounds rowBounds = new RowBounds(offset, limit);  
 return scoreService.getCourseList(rowBounds, condition);  
}  
  
@PostMapping  
private void addEntry(@RequestBody JSONArray UserScore) {  
 List<Score> list = JSONObject.*parseArray*(UserScore.toJSONString(), Score.class);  
 scoreService.addEntry(list);  
}  
  
@GetMapping("/export")  
public List<Course> getExportList(@RequestParam Map<String, Object> condition) {  
 return scoreService.getExportList(condition);  
}  
  
@GetMapping("/getUserNum")  
public List<Map<String, Object>> getUserNum(@RequestParam Map<String, Object> condition) {  
 return scoreService.getUserNum(condition);  
}  
  
@GetMapping("/getUserTotal")  
public Map<String, Object> getUserTotal(@RequestParam Map<String, Object> condition) {  
 return scoreService.getUserTotal(condition);  
}

### 5.2.4 用户管理接口

@GetMapping("/edit/password")  
public boolean update(@RequestParam Map<String, Object> condition) {  
 Map<String, Object> map = new HashMap<>();  
 map.put("username", condition.get("username").toString());  
 map.put("password", condition.get("password").toString());  
 map.put("passwordAgain", condition.get("passwordAgain").toString());  
 ;  
 map.put("level", condition.get("level").toString());  
 return userService.update(map);  
}  
  
@GetMapping("/getTree")  
public List<Object> getTree() {  
 return userService.getTree();  
}  
  
@PassToken  
@GetMapping("/getSilent")  
public boolean getSilent() {  
 return userService.getSilent();  
}  
  
@PutMapping("/setSilent/{state}")  
public boolean setSilent(@PathVariable("state") Integer state) {  
 return userService.setSilent(state);  
}

### 5.2.5 文件管理接口

@PostMapping("/headImg")  
@ResponseBody  
public String upload(MultipartFile file, HttpServletRequest request) throws IOException {  
 if (!file.isEmpty()) {  
 try {  
 byte[] bytes = file.getBytes();  
 *// 储存位置* String staticDir = ResourceUtil.*getPath*();  
  
 *// 图片名* String ImgName = file.getOriginalFilename();  
  
 String uid = UUID.*randomUUID*().toString();  
 assert ImgName != null;  
 *// 获取后缀名* String str = ImgName.substring(ImgName.lastIndexOf("."));  
 *// 重定义文件名* String newName = uid + str;  
  
 *// 图片存储地址* Path path = Paths.*get*(staticDir + newName);  
 *// 写入文件* Files.*write*(path, bytes);  
 String imgUrl = "/files/" + newName;  
  
 String userId = request.getParameter("id");  
 int level = *parseInt*(request.getParameter("level"));  
 Upload upload = new Upload();  
 upload.setUserId(userId);  
 upload.setLevel(level);  
 upload.setUrl(imgUrl);  
 uploadService.upload(upload);  
  
 *// url去除"sms"* return imgUrl;  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 return "";  
}  
  
@GetMapping("/getHeadImg")  
@UserLoginToken  
public String getAdminList(@RequestParam Map<String, Object> condition, HttpServletRequest httpServletRequest) {  
 return uploadService.getHeader(condition);  
}

# 6 性能

## 6.1 精度

要按照严格的数据格式输入，不能输入非法字符，否则系统不给予响应进行处理，查询时要保证准确率为100%，所有包含查询关键字的书籍都应能查到，不能有遗漏。

## 6.2 时间特性

（1）响应时间：用户任意操作后5秒内系统给予反馈信息。

（2）更新处理时间：由系统运行状态来决定。

（3）数据的转换和传送时间：能够在20秒内完成。

## 6.3 灵活性

当需求发生某些变化时，该软件的基本操作、数据结构、运行环境等等基本不会发生变化，只是对系统的数据库的文件和记录进行处理，就可以满足需求。

# 7 测试

|  |  |  |
| --- | --- | --- |
| 功能点 | 测试用例 | 输出结果 |
| 管理员用户登录 | 选择管理员角色，输入管理员的用户名和密码，点击登录 | 跳转到学生成绩管理系统首页，并展示管理员相关功能菜单 |
| 教师用户登录 | 选择教师角色，输入管理员的用户名和密码，点击登录 | 跳转到学生成绩管理系统首页，并展示教师相关功能菜单 |
| 学生用户登录 | 选择管理员角色，输入管理员的用户名和密码，点击登录 | 跳转到学生成绩管理系统首页，并展示读者相关功能菜单 |
| 课程信息新增 | 点击新增，输入课程信息，点击确认新增 | 未填写完整表单，提示对应异常，填写完整表单，提示新增成功 |
| 课程信息查询 | 输入对应查询条件，点击查询 | 课程信息列表 |
| 课程信息修改 | 选中课程信息，弹出课程信息修改框，修改对应信息，点击修改 | 未填写完整表单，提示对应异常，填写完整表单，提示修改成功 |
| 课程信息删除 | 选中删除的数据 | 提示删除成功 |
| 录入课程表 | 选中对应学年对应学期的课程表，录入对应课程 | 出现下拉框录入课程信息 |
| 成绩信息查询 | 输入对应查询条件，点击查询 | 成绩信息列表 |
| 录入成绩 | 选中对应成绩信息 | 录入成绩 |
| 修改成绩 | 选中对应成绩信息 | 修改成绩 |
| 修改课程表 | 选中对应学年学期的课程表，修改已有的课程表信息 | 出现下拉框修改课程信息 |
| 用户新增 | 点击新增，输入用户信息，点击确认新增 | 未填写完整表单，提示对应异常，填写完整表单，提示新增成功 |
| 用户信息查询 | 输入对应查询条件，点击查询 | 用户信息列表 |
| 用户修改 | 选中用户信息，弹出用户信息修改框，修改对应信息，点击修改 | 未填写完整表单，提示对应异常，填写完整表单，提示修改成功 |
| 用户删除 | 选中删除的数据 | 提示删除成功 |