**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于jQuery的学生管理系统的分析与设计 | | |
| 课 程 | 软件工程 | | |
| 学 院 | 信息工程学院 | | |
| 专 业 | 软件工程 | 年级 | 2023级 |
| 学生姓名 | 谭梦 | 学号 | 012322100033 |

|  |  |  |  |
| --- | --- | --- | --- |
| **评分项** | **评分标准** | **满分** | **得分** |
| 文档整体 | 文档内容详实、规范，美观大方，内容: 可行性分析、需求分析、概要设计、详细设计。 | 20 |  |
| 用例图及规约 | 用例图完整，准确，能够完全体现需求 | 10 |  |
| 顺序图与活动图 | 顺序图能够完全描述用例的设计思路和业务流程，活动图能够描述部分算法的流程。 | 10 |  |
| 类图 | 类图能够完整，准确反映业务的需数据的组织结构 | 10 |  |
| 数据库 | 数据库表的设计来自于类图，合理 | 10 |  |
| 界面设计 | 界面设计美观，清晰，合理，能够完全反映用例图的设计思路 | 10 |  |
| API接口设计 | 为每个界面设计合理的API接口，以便进行前后端分离开发 | 15 |  |
| 内容一致性 | 用例图，类图、数据库、界面相互印证，相互依赖，环环相扣 | 15 |  |
| **得分合计** | | |  |

2024 年 12 月 19 日

# 一、引言

# 随着信息技术的发展，学生信息管理系统已经成为学校管理工作的重要组成部分。在当今教育领域，随着学生数量的不断增加和教育管理要求的日益提高，高效、准确地管理学生信息成为学校管理工作的关键部分。传统的人工管理方式在面对海量学生信息时，存在信息查找困难、更新不及时、容易出错等诸多问题，已经无法满足现代教育管理的需求。

# 本项目旨在设计和实现一个功能全面、操作便捷、安全可靠的学生信息管理系统，满足学校在学生管理方面的多样化需求，推动教育管理工作朝着信息化、智能化的方向发展。

# **项目背景与需求分析**

## 2.1项目背景

学生管理系统是学校管理学生信息和教务工作的重要工具。它能够快速、准确地完成学生信息的录入、维护和查询，提供优质的教务管理服务，对学校的教学工作起到了积极的推动作用。

## 2.2需求分析

### 2.2.1 用户角色

#### 学生端

学生端的功能分为登录、注册、查看个人信息、查看成绩、申请修改信息。

#### 教师端

教师端的功能分为登录、注册、查看学生信息、修改学生信息、删除学生信息、添加学生信息、查看学生申请的修改信息。

### 2.2.2 功能需求

* 个人信息管理：支持学生个人信息的录入、修改和查询。
* 课程信息管理：支持课程信息的录入、修改和查询，包括课程名称、教师、上课时间等。
* 成绩信息管理：支持学生成绩的录入、修改和查询，提供成绩统计和分析功能。
* 通知公告管理：支持管理员发布通知公告，学生和教师可以查看通知公告。
* 用户权限管理：支持管理员对用户权限的分配和管理。

**2.2.3 非功能需求：**

* 系统性能：响应速度快，能够处理大量并发请求。
* 系统安全性：数据保密性强，防止信息泄露和非法访问。
* 系统稳定性：系统稳定可靠，能够长时间稳定运行。

## **2.4 运行环境规定**

**2.4.1 硬件环境**

2.4.1.1 服务器

* 处理器：Intel或AMD多核处理器，至少4核，推荐8核及以上。
* 内存：至少8GB RAM，推荐16GB或以上，以支持高并发访问和大数据处理。
* 存储：至少1TB SATA/SAS硬盘或SSD，确保数据存储空间充足。
* 备份设备：配置冗余硬盘或外部备份存储设备，确保数据安全。

2.4.1.2 客户端设

* PC/笔记本电脑：Intel或AMD处理器，至少2核，4GB RAM，推荐配置更高以 优化用户体验。
* 移动设备：支持Android 5.0及以上或iOS 11及以上的智能手机和平板电脑。

**2.4.2 操作系统**

2.4.2.1 服务器端

* 操作系统：支持Windows Server 2016/2019、Linux（如Ubuntu Server 18.04/20.04、CentOS 7/8）或Unix系统。
* 权限要求：需具备管理员权限，以便安装、配置及运行服务。

2.4.2.2 客户端

* PC/笔记本：Windows 7及以上、macOS 10.12及以上或主流Linux发行版。
* 移动设备：Android或iOS系统，需支持最新版本的浏览器（如Chrome、Safari）或专用APP。

**2.4.3 数据库**

* 数据库管理系统：MySQL 5.7及以上、PostgreSQL 10及以上、Oracle 12c及以上或SQL Server 2016及以上。
* 存储要求：高效的数据索引和存储策略，确保数据读写速度，支持事务处理和并发控制。
* 备份与恢复：定期自动备份数据库，支持快速恢复机制。

**2.4.4 网络环境**

* 服务器网络：千兆以太网接口，确保高速数据传输。
* 客户端网络：稳定的网络连接，建议带宽不低于10Mbps，支持TCP/IP协议。
* 安全性：部署防火墙、SSL证书，确保数据传输安全；实施访问控制策略，防止非法访问。

**2.4.5 其他软件依赖**

* Web服务器：Apache、Nginx或IIS，用于部署Web应用。
* 中间件：Tomcat、JBoss或WebLogic，支持Java应用运行。
* 开发工具：Visual Studio、Eclipse、IntelliJ IDEA等，用于开发和调试。
* 浏览器：支持Chrome、Firefox、Edge、Safari等主流浏览器，确保兼容性和用户体验。

**2.4.6 维护与监控**

* 日志管理：建立完善的日志系统，记录系统运行、错误及用户操作日志。
* 性能监控：部署性能监控工具，实时监控CPU、内存、磁盘及网络使用情况。
* 安全审计：定期进行安全审计，包括漏洞扫描、权限审查等，确保系统安全。

1. **系统概要与系统构成**

**3.1系统化的业务范围**

该系统包括五大模块，分别是成绩管理模块、学籍管理模块、班级管理模块、课程管理模块、打印管理模块，涵盖了学生管理的基本功能，适用于高校和初等中学。

**3.2 功能定义：**

• 成绩管理：包括学生成绩管理（课程号、课程名、学号、成绩）的添加、更改、删除和查询（按学号、班级和查询全部）等功能。

• 学籍管理：包括学生基本信息管理。其中学生基本信息管理包括学生基本信息（学生编号、姓名、性别、所在专业等）的添加、更改、删除和查询。

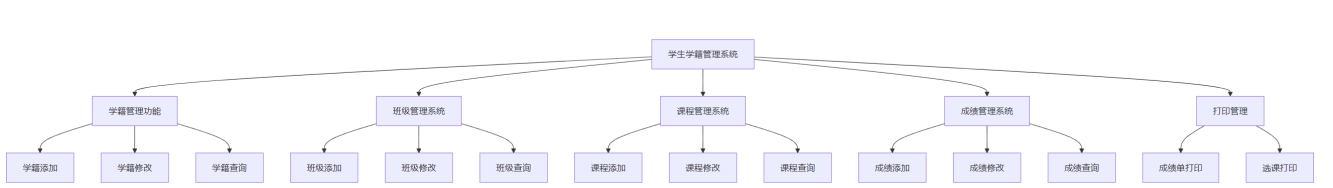
• 班级管理：班级添加、班级修改、班级查询等功能。

• 课程管理：包括课程信息（课程号、课程名称和授课老师）的添加和删除。

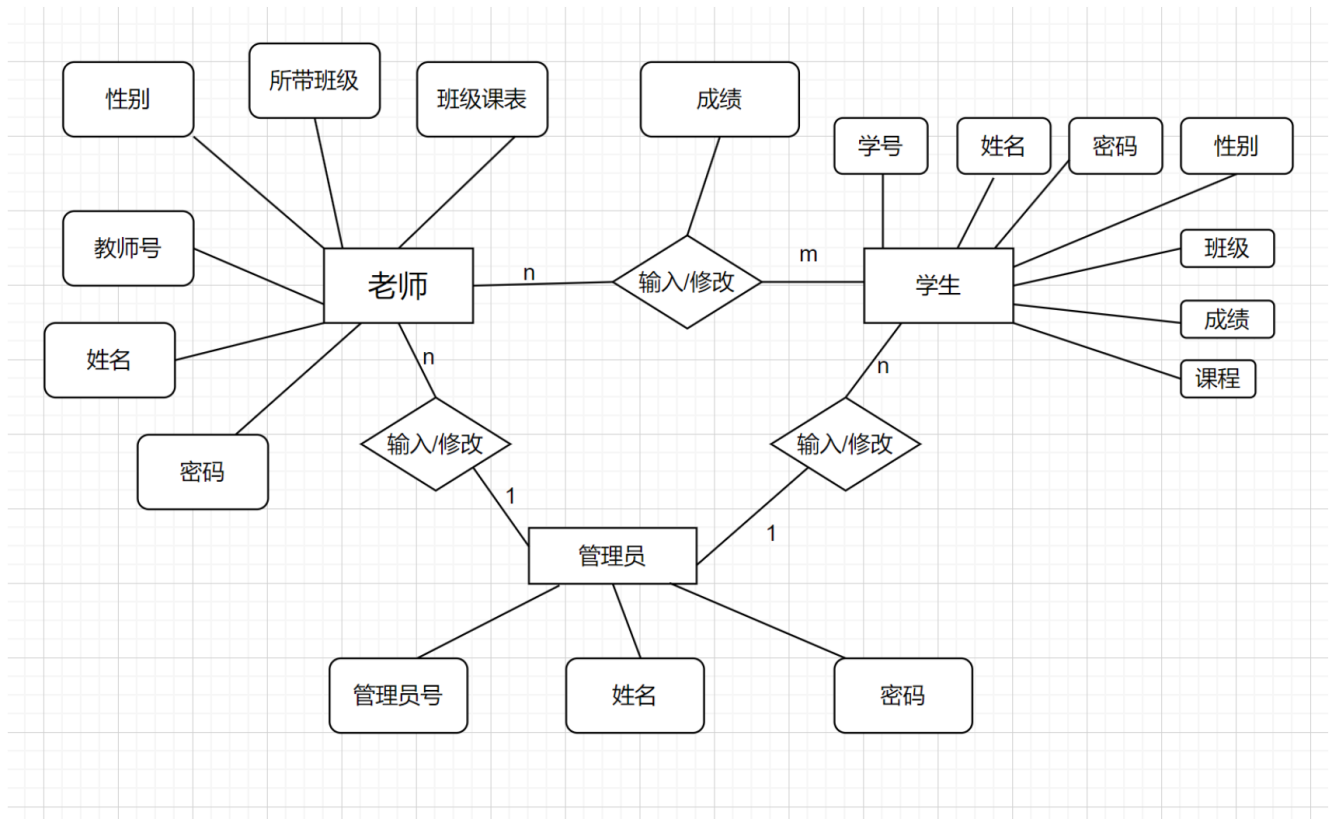
• 打印管理：包括输出学生的成绩单、学籍证明单和选课情况等的打印。

## 3.3 系统总体结构

### 3.3.1 系统层次结构图



### 3.3.2 基本实体间的E-R图



### 3.3.2数据字典

### 3.3.3 数据库设计

#### TEACHERS表（老师表）

| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| --- | --- | --- | --- | --- | --- | --- |
| TEACHER\_ID | VARCHAR2(50 BYTE) | 主键 | 否 |  |  | 老师的编号 |
| TEACHERS\_NAME | VARCHAR2(50 BYTE) |  | 否 |  |  | 老师真实姓名 |
| GITHUB\_USERNAME | VARCHAR2(50 BYTE) |  | 是 | 空 |  | 老师GitHUB用户名 |
| PASSWORD | VARCHAR2(512 BYTE) |  | 是 | 空 |  | 加密存储密码，为空表示密码就是学号 |
| DEPARTMENT | VARCHAR2(400 BYTE) |  | 否 |  |  | 老师属于的部门 |

​

​

#### STUDENTS表（学生表）

| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| --- | --- | --- | --- | --- | --- | --- |
| STUDENT\_ID | VARCHAR2(50 BYTE) | 主键 | 否 |  |  | 学生的学号 |
| STUDENTS\_NAME | VARCHAR2(50 BYTE) |  | 否 |  |  | 学生真实姓名 |
| MAJOR | VARCHAR2(20 BYTE) |  | 否 |  |  | 学生的专业 |
| CLASSNUM | VARCHAR2(20 BYTE) |  | 否 |  |  | 学生的班级号 |
| GITHUB\_USERNAME | VARCHAR2(50 BYTE) |  | 是 | 空 |  | 学生GitHUB用户名 |
| PASSWORD | VARCHAR2(512 BYTE) |  | 是 | 空 |  | 加密存储密码，为空表示密码就是学号 |
| RESULT\_SUM | VARCHAR2(400 BYTE) | 外键 | 是 | 空 |  | 成绩汇总（来自GRADES表），以逗号分开，第一个成绩是平均成绩,后面是每次实验的成绩，N表示未批改，平均分只计算已批改的。比如：“81.25,70,80,85,90,N”表示一共批改了4次，第5次未批改，4次的成绩分别是81.25,70,80,85,90,N，4次的平均分是81.25 |
| WEB\_SUM | VARCHAR2(400 BYTE) |  | 是 | 空 |  | GitHub网址是否正确，用逗号分开，Y代表正确，N代表不正确。第1位代表总的GitHUB地址是否正确，第2位表示第1次实验的地址，第3位表示第2位实验地址，依此类推。比如：“Y,Y,Y,Y,Y,N”表示第5次实验地址不正确，其他地址正确 |

​

​

#### LESSON表（课程表）

| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| --- | --- | --- | --- | --- | --- | --- |
| LESSON\_ID | VARCHAR2(50 BYTE) | 主键 | 否 |  |  | 课程的ID |
| TEACHER\_ID | VARCHAR2(50 BYTE) | 主键 | 否 |  |  | 老师的编号 |
| CLASSNUM | VARCHAR2(20 BYTE) |  | 否 |  |  | 学生的班级号 |
| YEAR | INT |  | 否 |  |  | 开课的年份 |
| SEMESTER | VARCHAR2(20 BYTE) |  | 否 |  |  | 开课的学期，"第一学期"或"第二学期" |
| LESSON\_TIME | DATE |  | 否 |  |  | 本学期上课的时间 |

​

​

#### CLASS表（班级表）

| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| --- | --- | --- | --- | --- | --- | --- |
| CLASS\_ID | VARCHAR2(50 BYTE) | 主键 | 否 |  |  | 班级的ID |
| GRADE | INT |  | 否 |  |  | 所属年级 |
| MAJOR | VARCHAR2(50 BYTE) |  | 否 |  |  | 专业名称 |
| CLASS\_NUM | INT |  | 否 |  |  | 班级序号，例如2班 |

​

​

#### CHOOSE\_LESSON表（选课表）

| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| --- | --- | --- | --- | --- | --- | --- |
| CHOOSE\_LESSON\_ID | VARCHAR2(50 BYTE) | 主键 | 否 |  |  | 选课的ID |
| LESSON\_ID | VARCHAR2(50 BYTE) | 外键 | 否 |  |  | 课程的ID |
| STUDENT\_ID | VARCHAR2(50 BYTE) | 外键 | 否 |  |  | 学生的学号 |
| CHOOSE\_LESSON\_TIME | DATE |  | 否 |  |  | 选课的时间 |

​

​

#### GRADES表（学生成绩表）

| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| --- | --- | --- | --- | --- | --- | --- |
| STUDENT\_ID | VARCHAR2(50 BYTE) | 联合主键1，外键 | 否 |  |  | 学生的学号，STUDENTS表外键 |
| TEST\_ID | NUMBER(6,0) | 联合主键2，外键 | 否 |  |  | 学科编号，TESTS表的外键 |
| RESULT | NUMBER | 主键 | 是 | 空 | 取值0-100 | 分数，这个值为空表示没有批改 |
| MEMO | VARCHAR2(400 BYTE) |  | 是 | 空 |  | 老师对学科的评语 |
| UPDATE\_DATE | DATE |  | 是 | 空 |  | 老师批改学科的日期，为空表示未批改 |

​

​

#### AVERAGE\_SCORE表（学生平均成绩表）

| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| --- | --- | --- | --- | --- | --- | --- |
| AVERAGE\_SCORE\_ID | NUMBER(6,0) | 主键 | 否 |  |  | 平均成绩的ID号 |
| LESSON\_ID | VARCHAR2(50 BYTE) | 主键 | 否 |  |  | 课程的ID |
| TITLE | VARCHAR2(100 BYTE) |  | 否 |  |  | 学科名称 |
| STUDENT\_ID | VARCHAR2(50 BYTE) | 主键 | 否 |  |  | 学生的学号 |
| AVERAGE\_SCORE | NUMBER | 主键 | 是 | 空 | 取值0-100 | 计算后得到的平均成绩 |

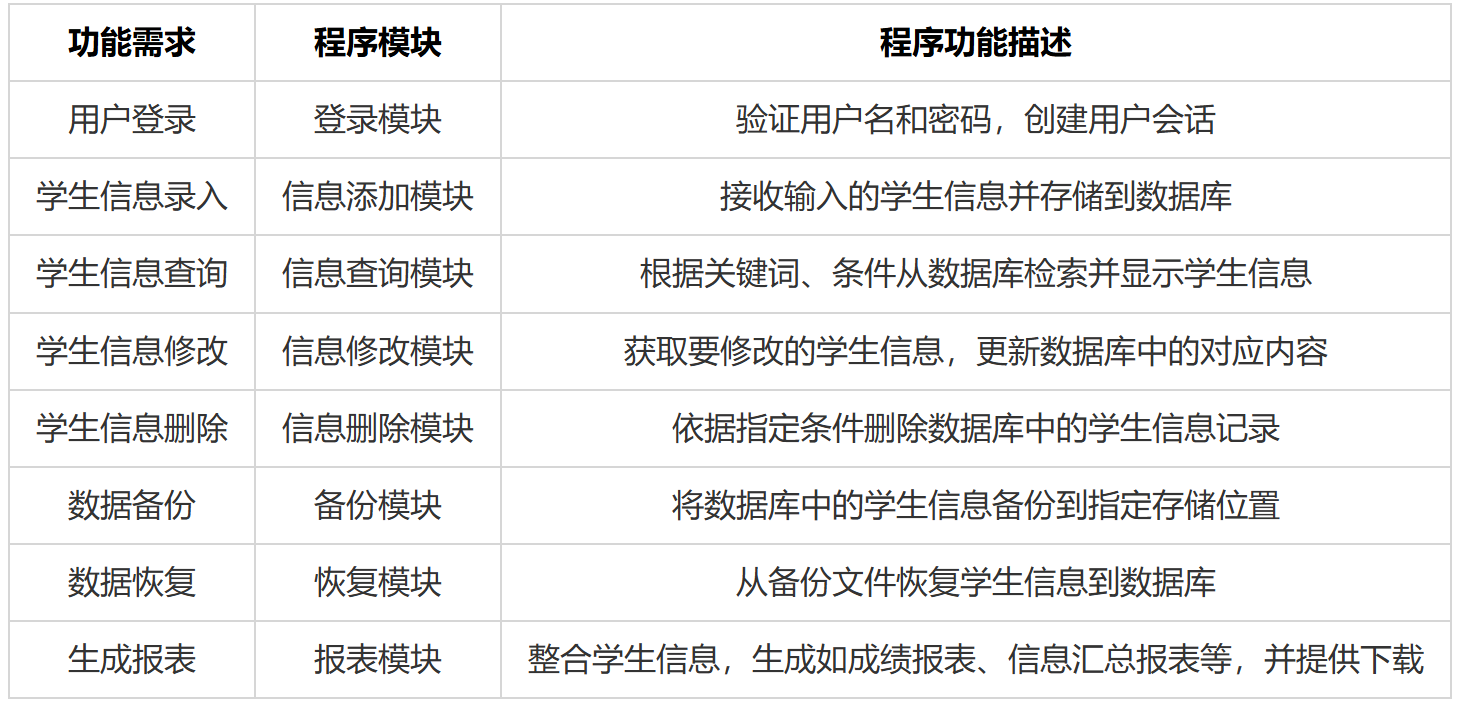
* 学生\_课程表

| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| --- | --- | --- | --- | --- | --- | --- |
| STUDENT\_ID | NUMBER(6,0) | 主键 | 否 |  |  | 用于连接学生表 |
| LESSON\_ID | NUMBER(6,0) | 主键 | 否 |  |  | 用于连接课程表 |

* 老师\_班级表

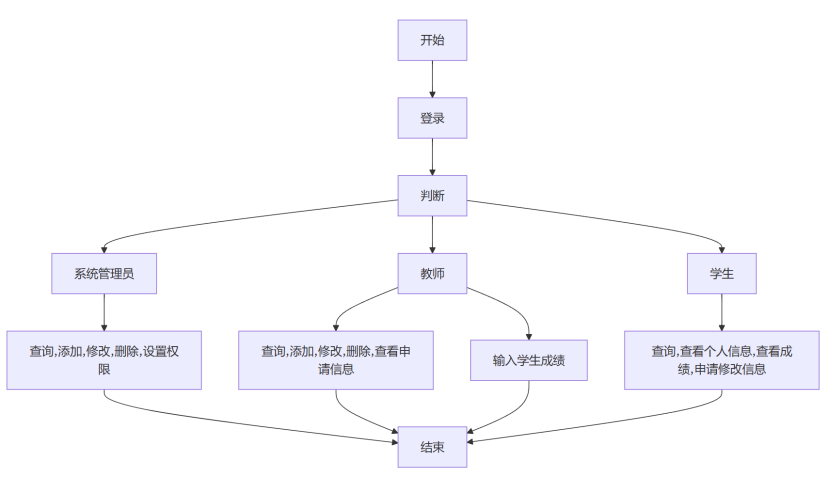
| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| --- | --- | --- | --- | --- | --- | --- |
| TEACHER\_ID | NUMBER(6,0) | 主键 | 否 |  |  | 用于连接老师表 |
| CLASS\_ID | NUMBER(6,0) | 主键 | 否 |  |  | 用于连接班级表 |

### 3.3.4 数据结构与程序的关系表

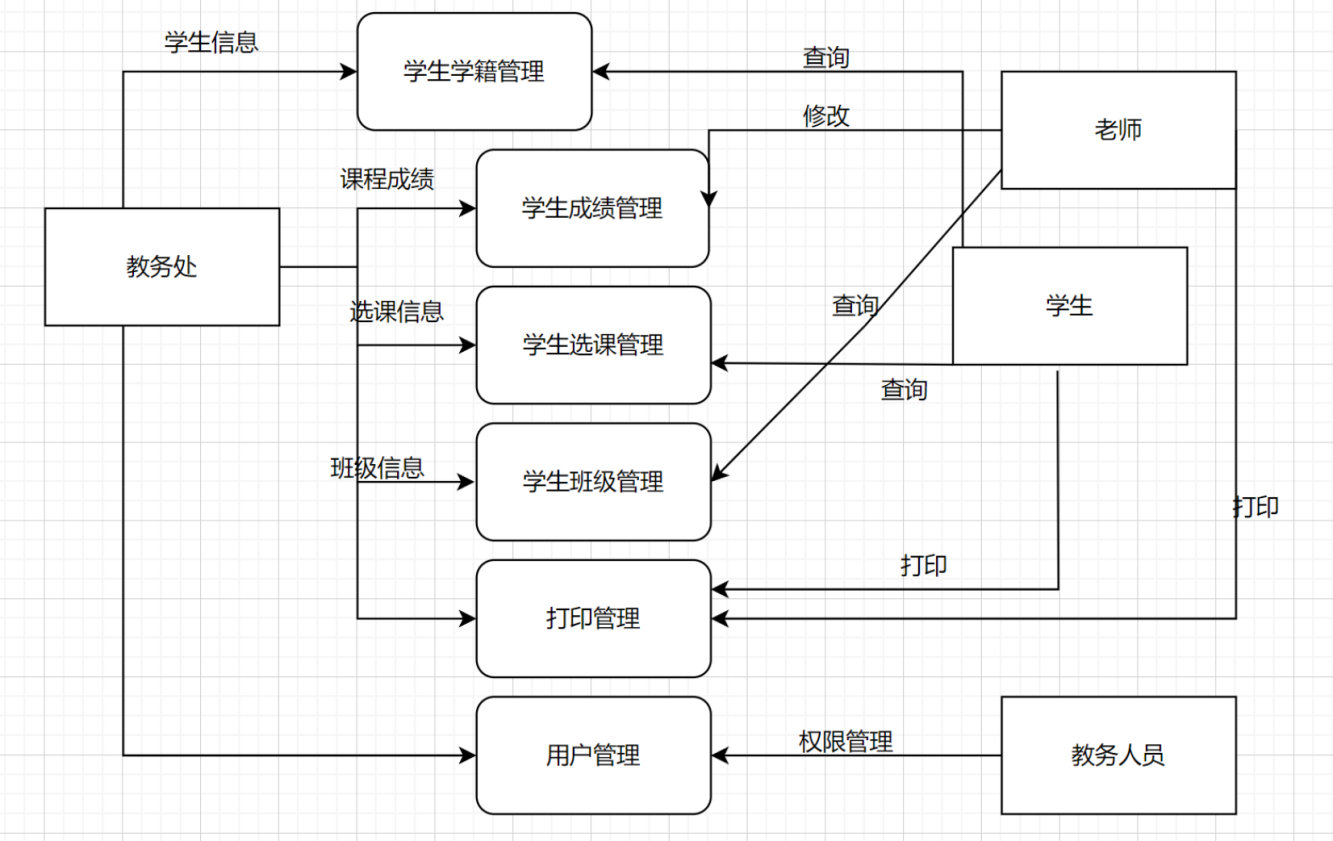


1. **结构化分析**

## 4.1整体的业务流程

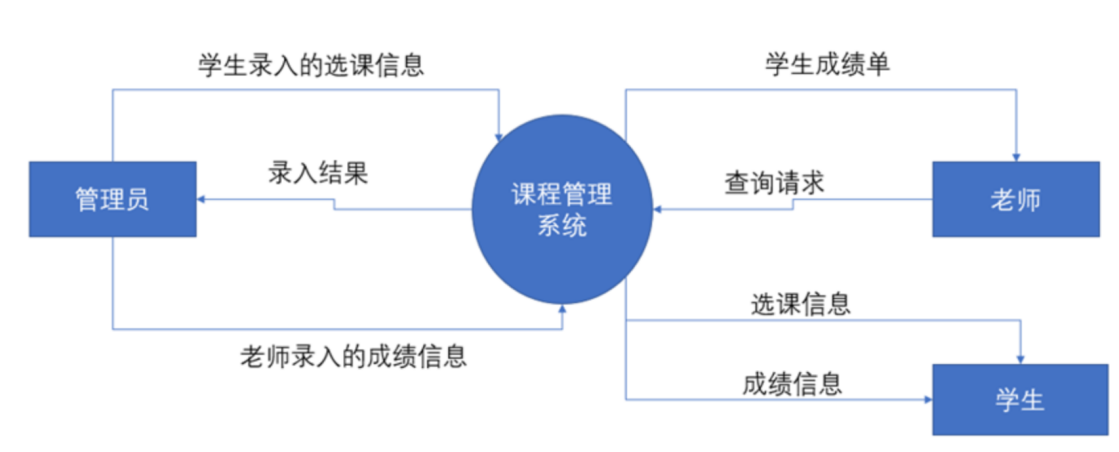


## 4.2系统基本数据框架

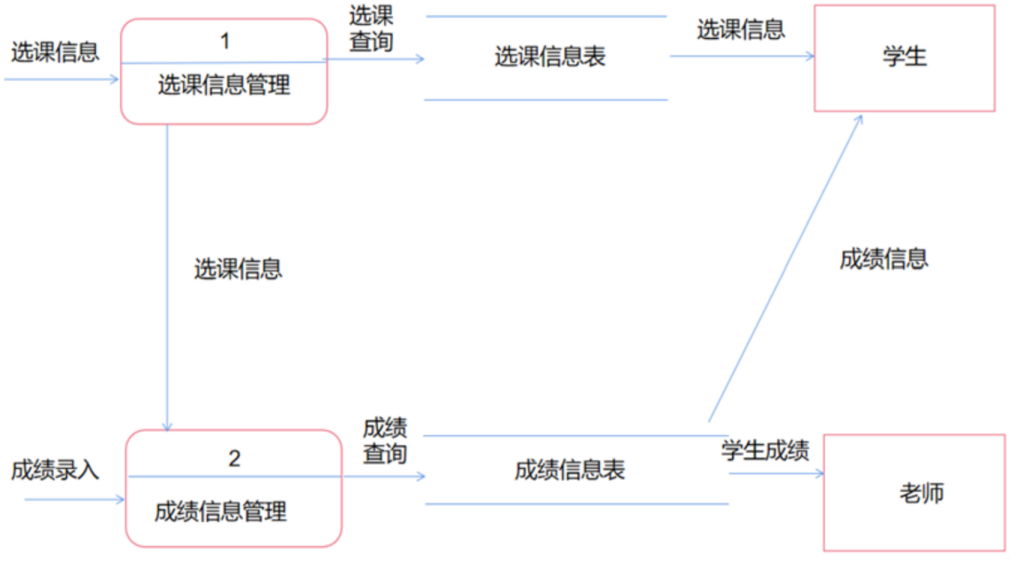


## 4.3 课程管理的业务流程

#### 顶层图

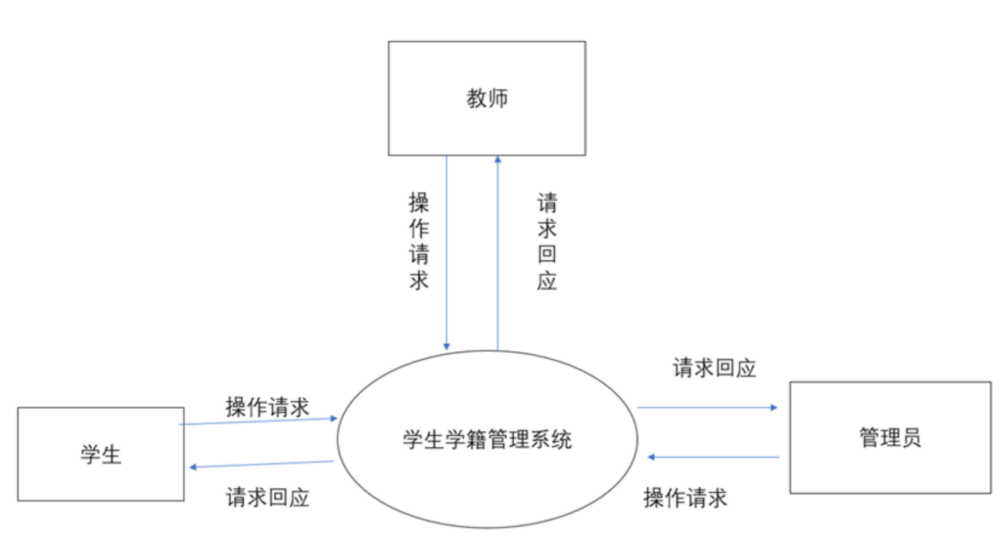


#### 0层图



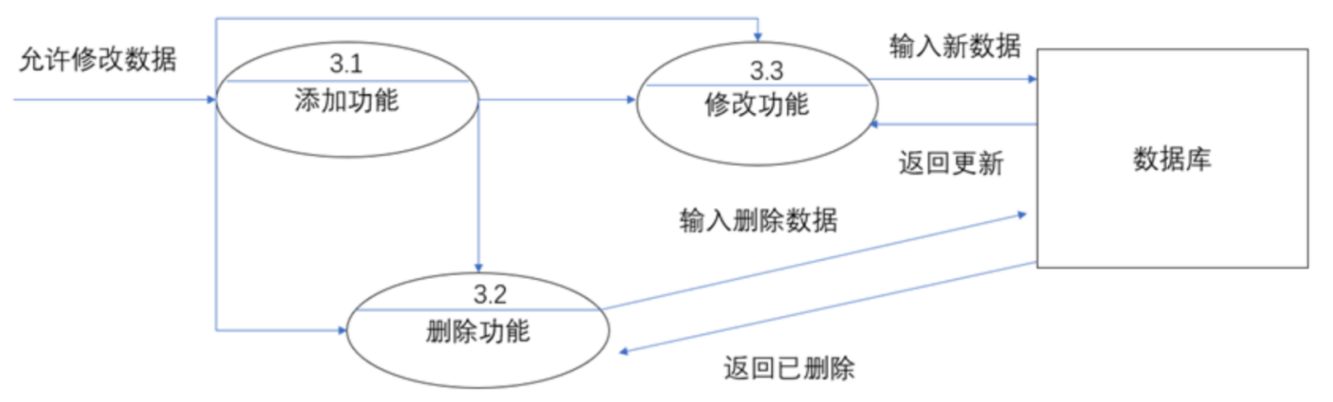
## 4.4 学籍管理的业务流程

#### 顶层图



0层图

#### 1层图



**五、学生信息管理系统登录用例**



接口login



接口：getUserInfo

接口：logout

接口：setUserInfo



接口：logout



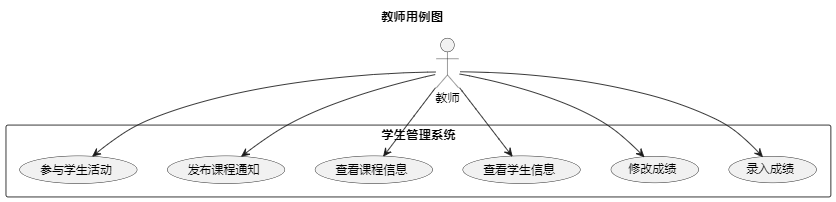
接口：logout

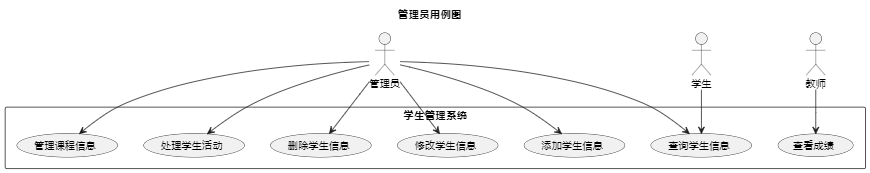


接口：logout

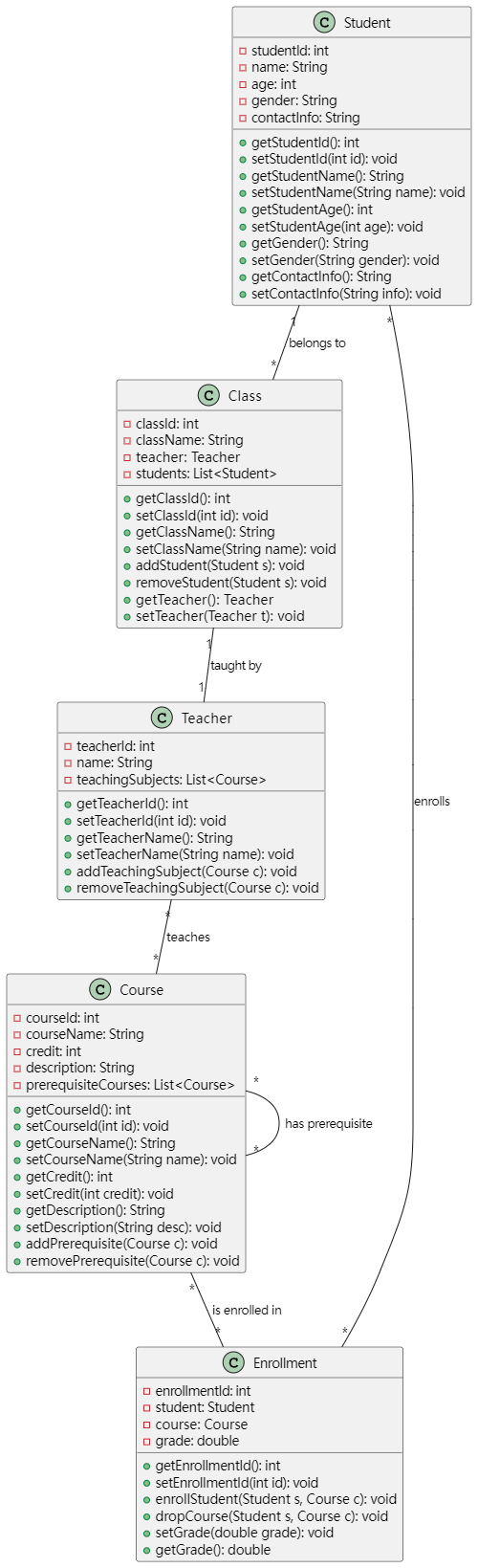
## 5.1 用例图设计



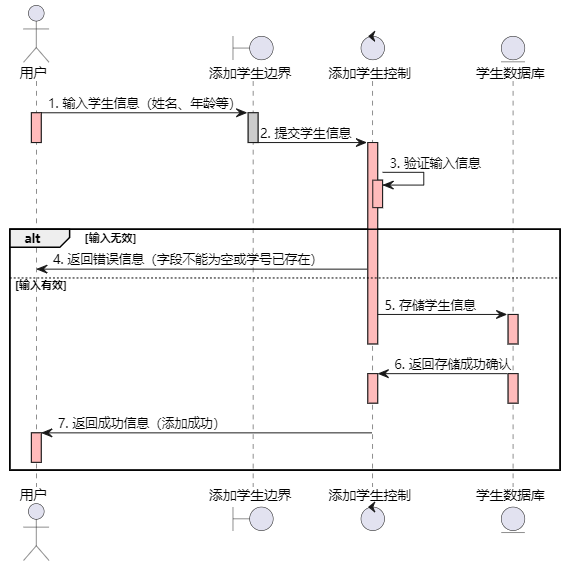


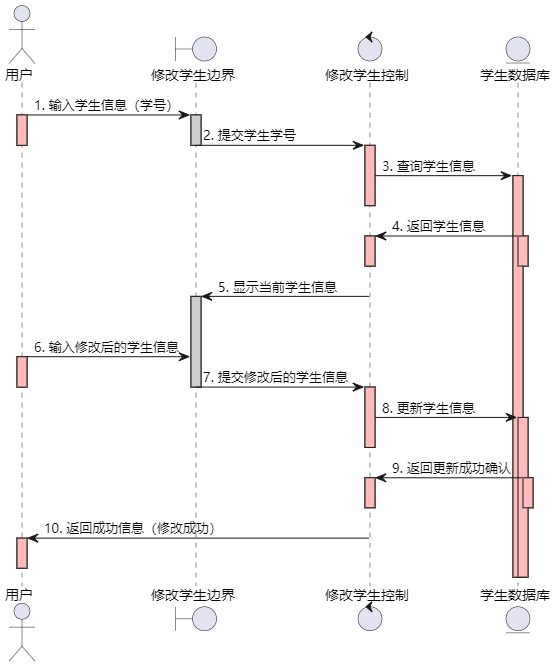


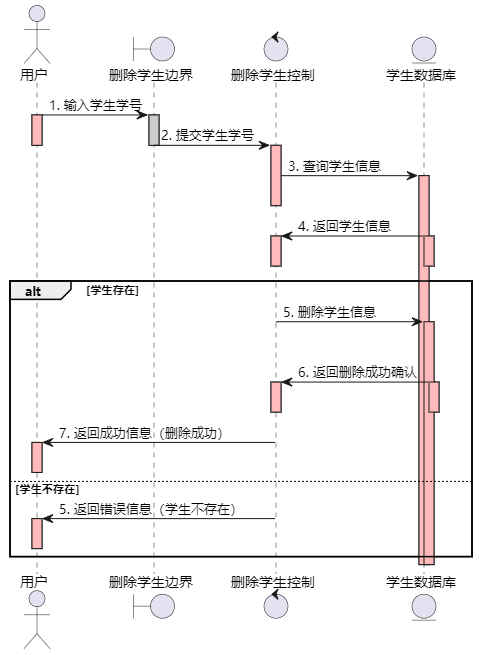
## 5.2 类图设计



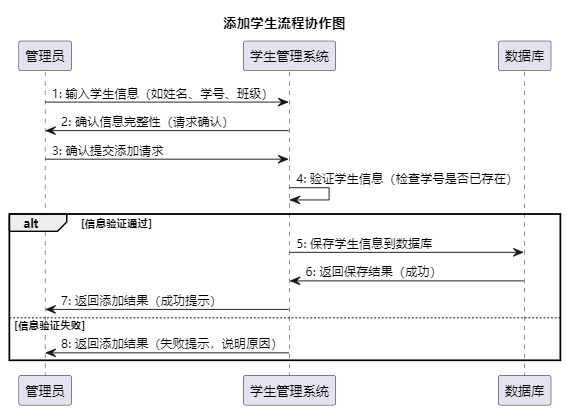
## 5.3时序图设计

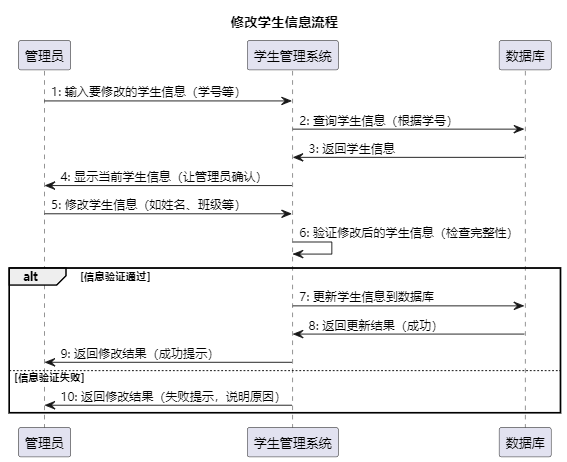


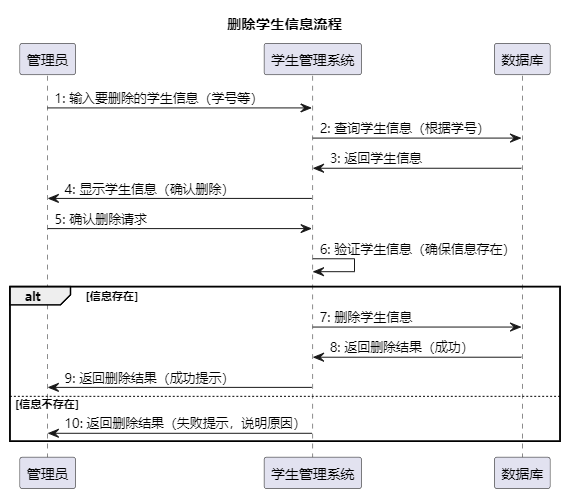




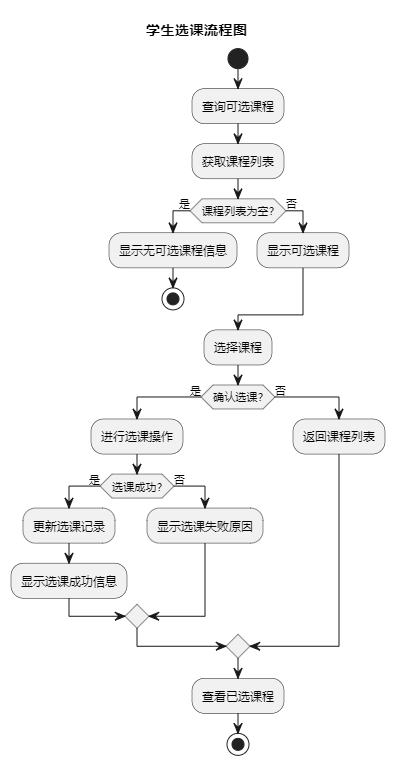
## 5.4 协作图设计

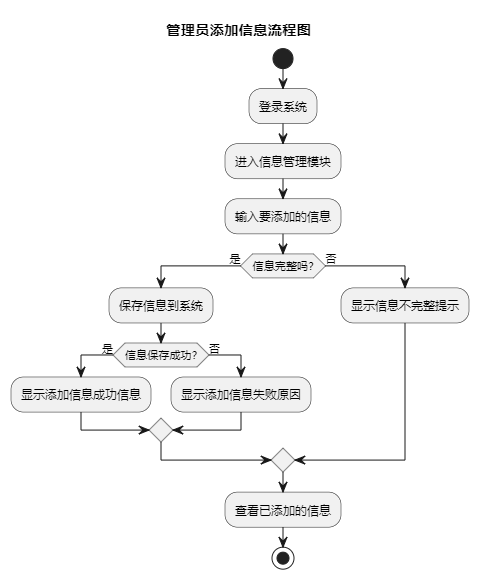
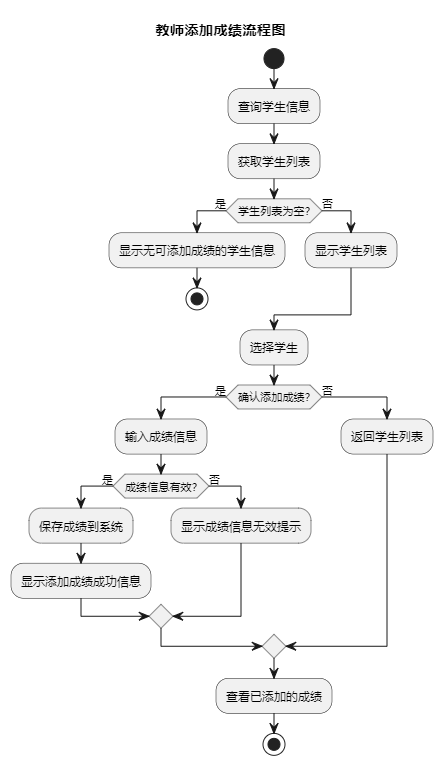




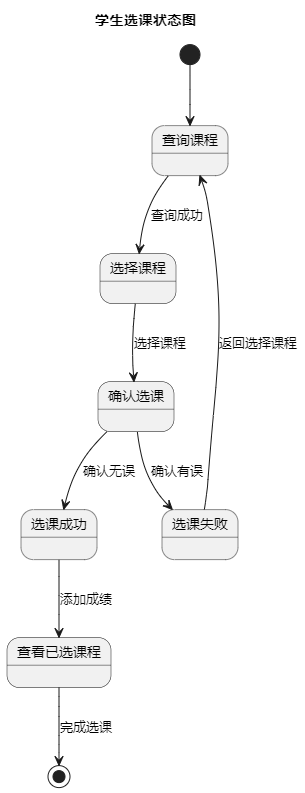
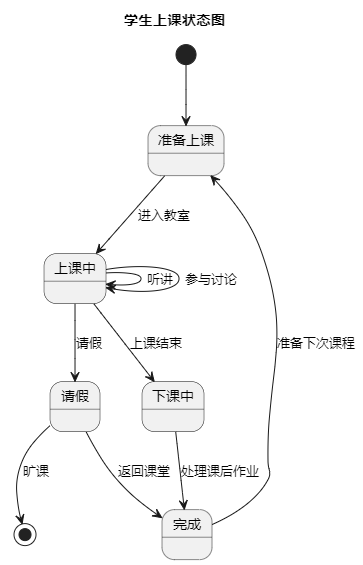


## 5.5 活动图设计

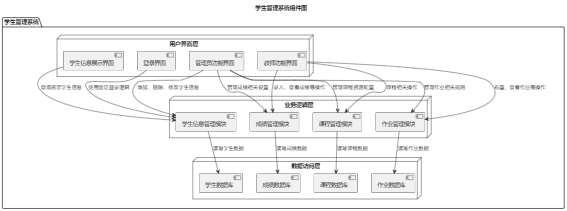




## 5.6 状态图



## 5.7 组件图



# 六、用例及界面详细设计

# 6.1 登录用户设计

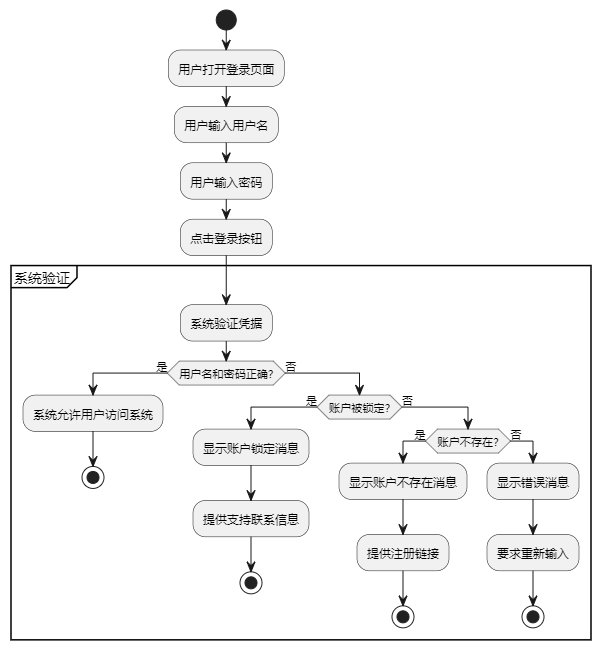
## 6.1.1 用例规约



## 6.1.2 业务流程（顺序图） —— 登录



活动图 —— 登录



### 6.1.3界面设计 —— 登录



loginUser

### 6.1.4接口设计

* 功能：用户登录
* API请求地址：接口基本地址/v1/api/loginUser
* 请求方式：POST
* 请求实例：

json

{

"username": "张三",

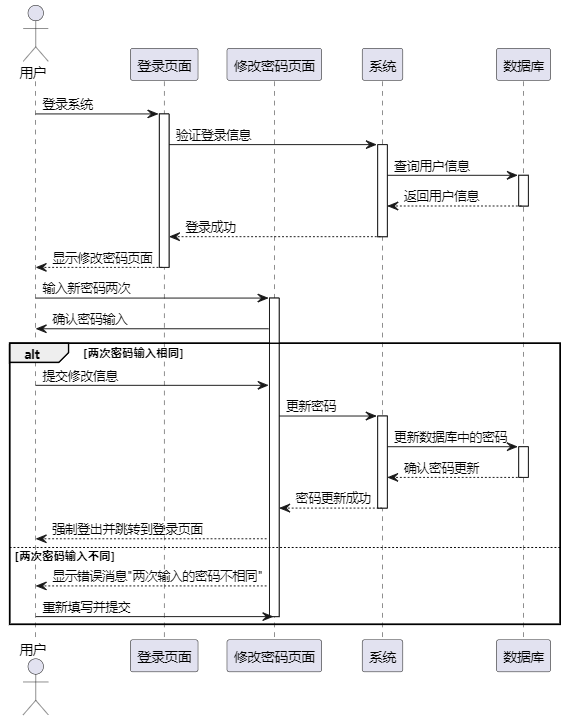
"password": "123456"

}

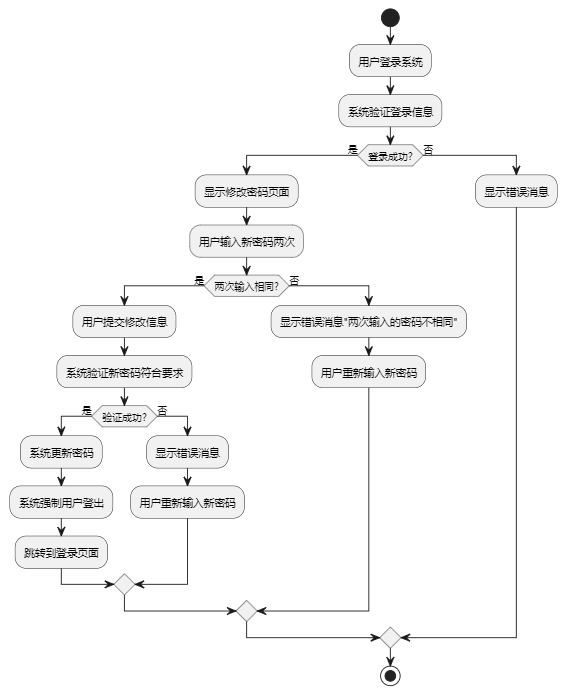
# 6.2 [修改](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E7%94%A8%E4%BE%8B\\%E4%BF%AE%E6%94%B9%E7%94%A8%E6%88%B7%E4%BF%A1%E6%81%AF.md)密码与用户名

## 6.2.1修改密码 —— 用例规约

| **用例名称** | **修改密码** |
| --- | --- |
| 功能 | 修改用户的密码 |
| 参与者 | 学生，老师 |
| 前置条件 | 必须先登录 |
| 后置条件 | 修改密码以后必须强制登出，再跳转到登录页面 |
| 主事件流 | 1.用户填写新密码（两次输入）  2.用户提交修改信息  3.系统存储修改后密码 |
| 备选事件流 | 1a. 用户两次输入的密码不同     1.系统提示两次输入的密码不相同      2. 用户重新填写并提交。 |

6.2.1.1 修改密码的流程 ——顺序图

### 6.2.1.2 活动图



6.2.1.3界面



setPassword

SetUserInfo

### 6.2.1.4 接口1：[setPassword](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E6%8E%A5%E5%8F%A3\\setPassword.md)

* 功能： 修改（设置）用户的密码。
* 权限： 学生/老师：修改自己的密码，必须先登录。
* API请求地址： 接口基本地址/v1/api/setPassword
* 请求方式 ： POST
* 请求实例：

{

"user\_id":张三,

"password":123456

}

* 请求参数说明:

| **参数名称** | **说明** |
| --- | --- |
| user\_id | 用户的ID号。对应表USERS.USER\_ID的值 |
| password | 用户的密码。不能为空，必须经过加密码,不能是密码的原文。 |

* 返回实例：

{

"status": true,

"info": null,

}

* 返回参数说明：

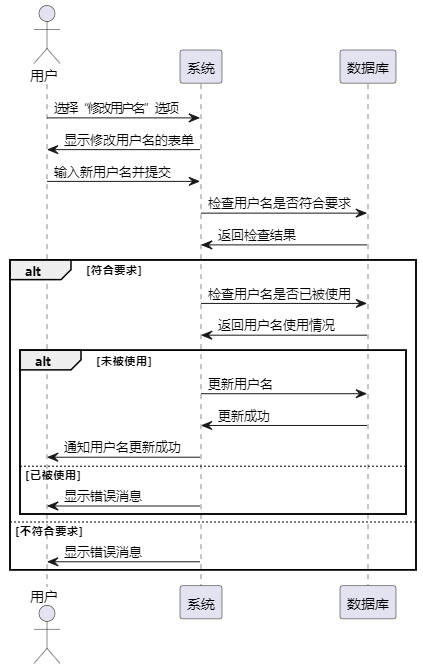
| **参数名称** | **说明** |
| --- | --- |
| status | bool类型，true表示正确的返回，false表示有错误 |
| info | 返回结果说明信息 |
|  |  |

## 6.2.2 修改用户名

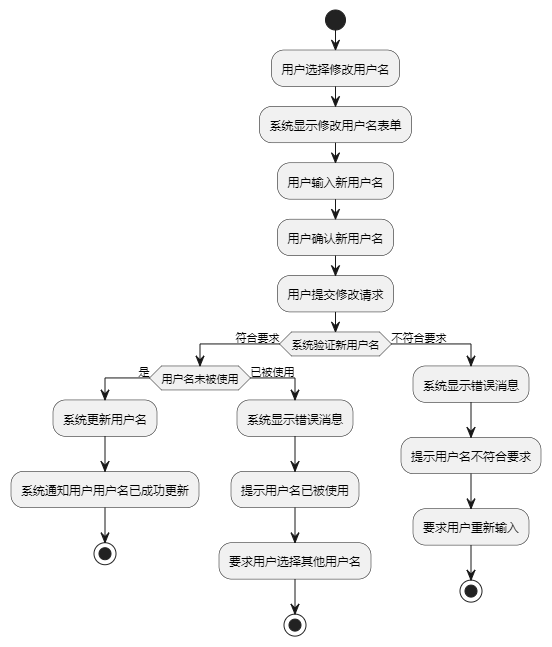
### 6.2.2.1 用例规约



### 6.2.2.2业务流程（顺序图）



#### 活动图



### 6.2.2.3 接口设计

### 6.2.2.4 接口1：[setUserInfo](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E6%8E%A5%E5%8F%A3\\setPassword.md)

用例： [修改用户信息](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E7%94%A8%E4%BE%8B\\%E4%BF%AE%E6%94%B9%E5%AF%86%E7%A0%81.md)

* 功能： 修改用户的管理系统用户名。
* 权限： 学生/老师：修改自己的密码，必须先登录。
* API请求地址： 接口基本地址/v1/api/setUserInfo
* 请求方式 ： POST
* 请求实例：

{

"id":"21048329823",

"github\_username":"ABCDE"

}

* 请求参数说明:

| **参数名称** | **说明** |
| --- | --- |
| user\_id | 用户的ID号。对应表USERS.USER\_ID的值 |
| github\_username | 用户管理系统用户名。 |

* 返回实例：

{

"status": true,

"info": null,

}

* 返回参数说明：

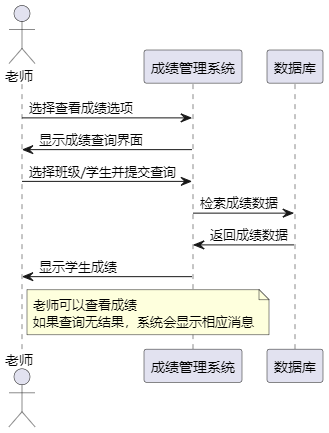
| **参数名称** | **说明** |
| --- | --- |
| status | bool类型，true表示正确的返回，false表示有错误 |
| info | 返回结果说明信息 |

# 6.3 [“查看成绩\_老师”用例](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E7%94%A8%E4%BE%8B\\%E6%9F%A5%E7%9C%8B%E6%88%90%E7%BB%A9_%E5%AD%A6%E7%94%9F.md),[界面](https://ssw383318348.github.io/is_analysis_pages/test6/%E6%9F%A5%E7%9C%8B%E6%88%90%E7%BB%A9_%E5%AD%A6%E7%94%9F.html)

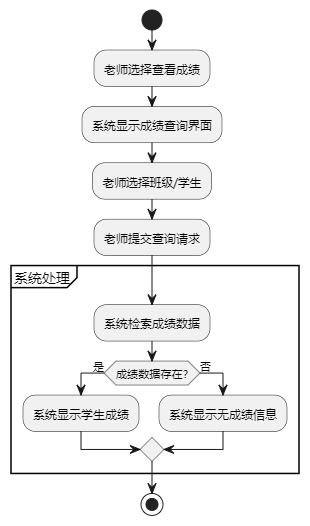
### 6.3.1 用例规约

### 

### 6.3.2业务流程（顺序图） —— 查看成绩



#### 活动图



### 6.3.3界面设计



[getStudentsByTeacher](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E6%8E%A5%E5%8F%A3\\setPassword.md)

### 6.3.4接口设计

接口1：[getStudentsByTeacher](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E6%8E%A5%E5%8F%A3\\setPassword.md)

* API请求地址：接口基本地址/v1/api/[getStudentsByTeacher](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E6%8E%A5%E5%8F%A3\\setPassword.md)
* 接口描述：该接口用于获取指定学生的报考结果信息，包括报考时间、报考编号、报考科目及考试状态。
* 请求方式 ： GET
* 请求实例：

{

"message": "成功",

"data": [

{

"examDate": "2021-01-13",

"examNumber": "2021010113",

"examSubject": "计算机网络原理",

"status": "不通过"

},

{

"examDate": "2021-01-15",

"examNumber": "2021010102",

"examSubject": "软件工程",

"status": "通过"

},

{

"examDate": "2021-01-11",

"examNumber": "2021010114",

"examSubject": "马克思主义原理",

"status": "通过"

},

{

"examDate": "2021-01-11",

"examNumber": "2021010113",

"examSubject": "web前端开发",

"status": "不通过"

}

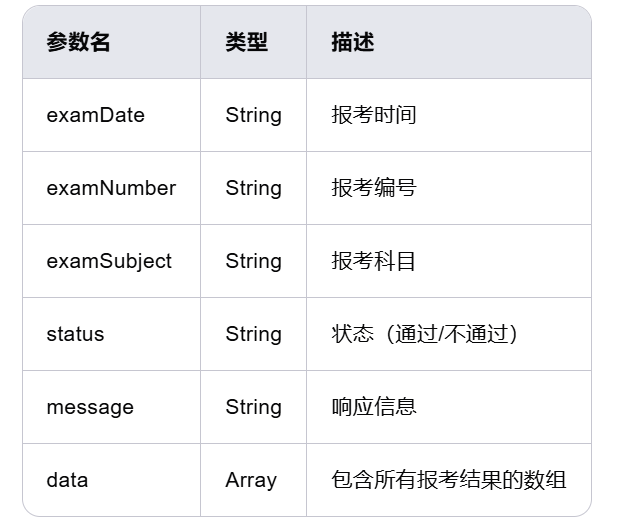
]

}

* 请求参数说明:



* 返回参数说明：

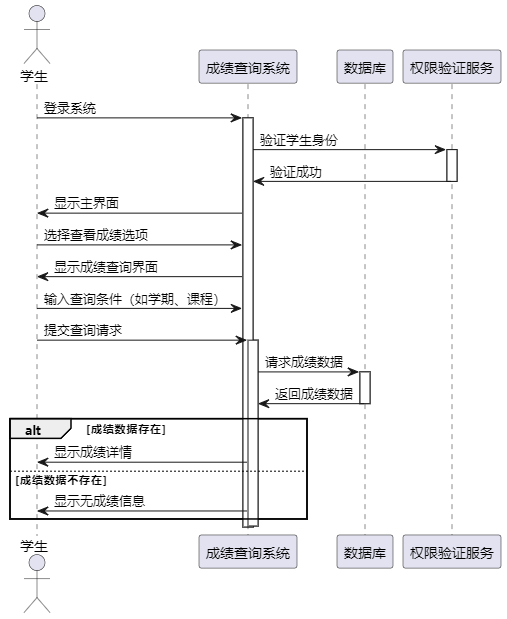


# 6.4 [“查看成绩\_学生”用例](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E7%94%A8%E4%BE%8B\\%E6%9F%A5%E7%9C%8B%E6%88%90%E7%BB%A9_%E8%80%81%E5%B8%88.md)

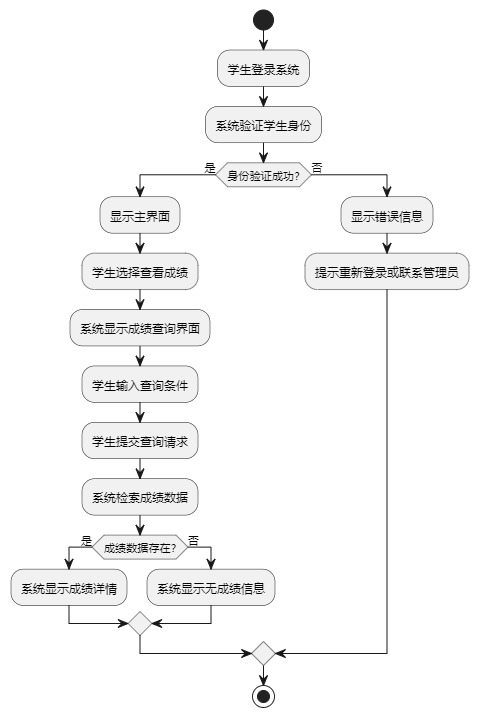
### 6.4.1用例规约

### 

### 6.4.2业务流程（顺序图）——查看成绩



#### 活动图



### 6.4.3界面设计



接口：getOneStudentResults

### 6.4.4接口设计

接口1： getOneStudentResults

用例： [查看成绩](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E7%94%A8%E4%BE%8B\\%E4%BF%AE%E6%94%B9%E5%AF%86%E7%A0%81.md)

* 接口描述：该接口用于获取指定学生的所有课程成绩信息，包括课程名称、学分、课程性质和成绩。
* API请求地址：接口基本地址/api/student/{studentId}/results
* 请求方式 ： GET
* 请求参数说明:



* 返回实例：

{

"success": true,

"message": "成绩信息获取成功",

"courses": [

{

"courseId": "C001",

"courseName": "数据库原理",

"credits": 3,

"nature": "选修课",

"score": 80

},

{

"courseId": "C002",

"courseName": "计算机网络原理",

"credits": 2,

"nature": "必修课",

"score": 72

}

// 其他课程成绩...

]

}

* 响应参数说明：

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 描述 |
| courses | Array | 包含所有课程成绩的数组 |
| message | String | 响应信息 |
| success | Boolean | 请求是否成功 |

* 课程成绩对象结构

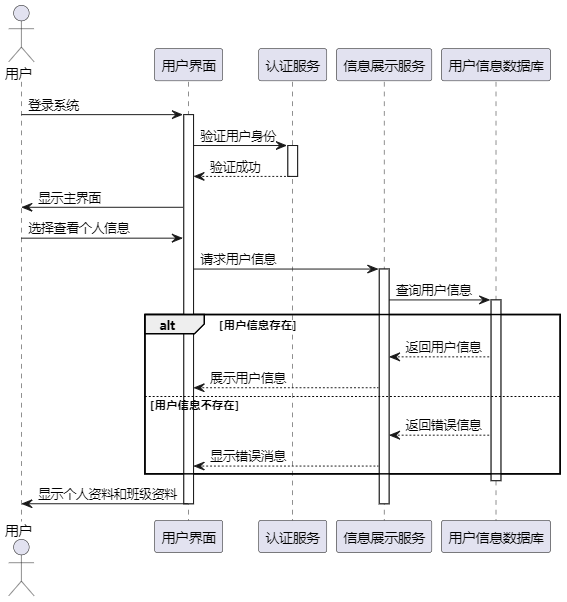
| **参数名** | **类型** | **描述** |
| --- | --- | --- |
| courseId | String | 课程编号 |
| courseName | String | 课程名称 |
| credits | Integer | 学分 |
| nature | String | 课程性质（必修/选修） |
| score | Integer | 成绩 |

# 6.5 [“查看用户信息”用例](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E7%94%A8%E4%BE%8B\\%E6%9F%A5%E7%9C%8B%E7%94%A8%E6%88%B7%E4%BF%A1%E6%81%AF.md),[界面](https://ssw383318348.github.io/is_analysis_pages/test6/%E6%9F%A5%E7%9C%8B%E7%94%A8%E6%88%B7%E4%BF%A1%E6%81%AF.html)

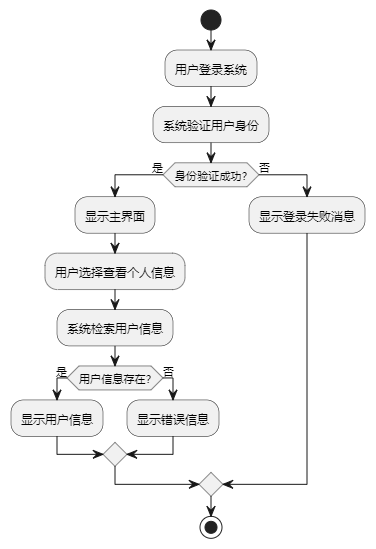
## 6.5.1 用例规约：

| **用例名称** | **查看用户信息** |
| --- | --- |
| 功能 | 用户查看个人信息 |
| 参与者 | 用户（学生/老师） |
| 前置条件 | 用户已成功登录系统。 |
| 后置条件 | 用户成功查看了个人信息。 |
| 主成功场景 | 1. 用户在个人中心选择查看个人信息的选项。2. 系统显示用户信息界面，包括个人资料和班级资料。3. 用户查看个人信息，包括姓名、性别、年龄、班级、联系方式等。 |
| 扩展/异常流程 | 1. 信息加载失败：系统显示错误消息，并提示用户稍后重试。2. 权限不足：系统显示错误消息，提示用户没有权限查看该信息。 |
| 特殊需求 | 系统应确保用户信息的安全性和隐私性。系统应提供清晰的用户界面以展示用户信息。 |
| 业务规则 | 1. 用户只能查看自己的信息。2. 系统应实时更新用户信息，确保用户查看的是最新数据。3. 系统应允许用户在一定条件下更新自己的信息。 |

### 6.5.2业务流程（顺序图） —— 学生信息



#### 活动图：



## 6.5.3界面设计



接口：getUserInfo

### 6.5.4接口设计

接口1：[getUserInfo](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E6%8E%A5%E5%8F%A3\\setPassword.md)

用例： [查看用户信息](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E7%94%A8%E4%BE%8B\\%E4%BF%AE%E6%94%B9%E5%AF%86%E7%A0%81.md)

* 功能： 查看用户的所有信息。
* 权限： 学生/老师：查看自己的信息，必须先登录，不能查看其他用户的信息。
* API请求地址： 接口基本地址/v1/api/getUserInfo/<user\_id>
* 请求方式 ：GET
* 请求参数说明:

| **参数名称** | **说明** |
| --- | --- |
| user\_id | 用户的ID号。对应表USERS.USER\_ID的值 |

* 返回实例：

{

"status": true,

"info": null,

"ID":"2101201281",

"name":"张三",

"class\_dept":"2019级软件技术4班"

"type":"学生"

}

* 返回参数说明：

| **参数名称** | **说明** |
| --- | --- |
| status | bool类型，true表示正确的返回，false表示有错误 |
| info | 返回结果说明信息 |
| ID | 学号或者工号 |
| name | 用户的真实姓名 |
| class\_dept | 班级或者部门名称 |
| type | 用户类型：老师或者学生 |
|  |  |

# 6.6 [“登出”用例](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E7%94%A8%E4%BE%8B\\%E7%99%BB%E5%87%BA.md),[界面](https://ssw383318348.github.io/is_analysis_pages/test6/%E7%99%BB%E5%87%BA.html)

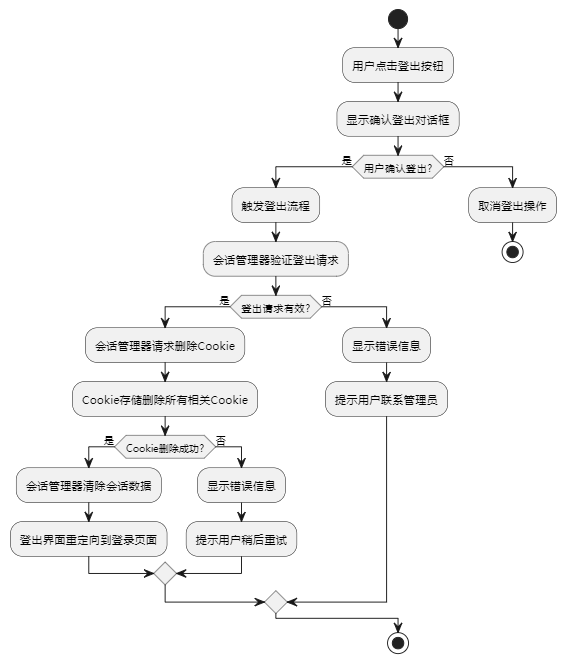
### 6.6.1用例规约

| **用例名称** | **登出** |
| --- | --- |
| 功能 | 用户登出平台 |
| 参与者 | 学生/老师 |
| 前置条件 |  |
| 后置条件 | 登出后，跳转到登录页面 |
| 主事件流 | 1. 系统清除客户端登录信息（Cookie） |
| 备选事件流 | 1. 如果用户登录之后，长时间不超作界面，导致Cookie失效   2.系统清除客户端登录信息(Cookie) |

## 6.6.2业务流程（顺序图）

#### 

#### 活动图



### 6.6.3界面设计



接口：logout

### 6.6.4接口设计

接口1：[logout](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E6%8E%A5%E5%8F%A3\\setPassword.md)

用例： [登出](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E7%94%A8%E4%BE%8B\\%E4%BF%AE%E6%94%B9%E5%AF%86%E7%A0%81.md)

* 功能： 已经登录的用户登出平台。
* 权限： 学生/老师已经登录。
* API请求地址： 接口基本地址/v1/api/logout
* 请求方式 ： POST
* 请求实例：

{

"user\_id":123456,

}

* 请求参数说明:

| **参数名称** | **说明** |
| --- | --- |
| user\_id | 用户的ID号。对应表USERS.USER\_ID的值 |

* 返回实例：

{

"status": true,

"info": null

}

* 返回参数说明：

| **参数名称** | **说明** |
| --- | --- |
| status | bool类型，true表示正确的返回，false表示有错误 |
| info | 返回结果说明信息 |

# 6.7 [“选课\_学生”用例](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E7%94%A8%E4%BE%8B\\%E9%80%89%E8%AF%BE_%E5%AD%A6%E7%94%9F.md)

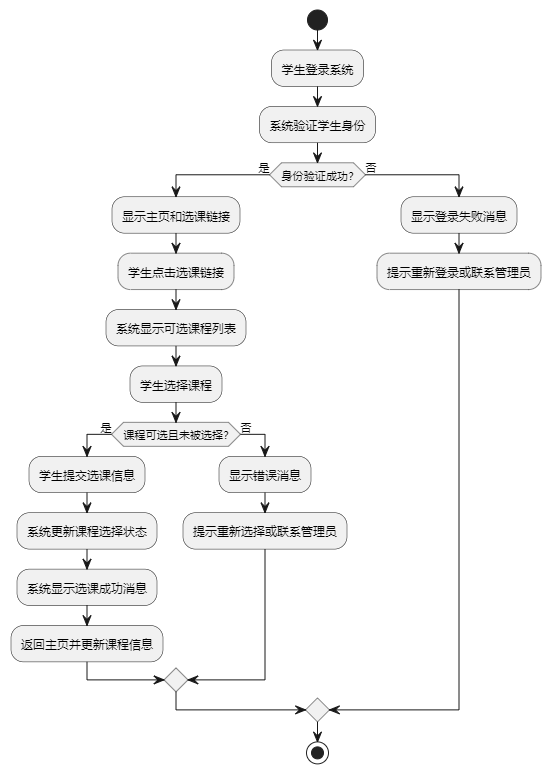
### 6.7.1用例规约

| **用例名称** | **选课\_学生** |
| --- | --- |
| 功能 | 学生选课平台 |
| 参与者 | 学生 |
| 前置条件 | 学生需要先登录，登录成功后，跳转到主页，点击选课链接，选择上课课程 |
| 后置条件 | 成功选择并提交上课课程的信息后，返回主页 |
| 主事件流 | 1. 学生选择要上课的课程 2.系统判断课程信息是否存在，是否被选择 3.学生选择并提交上课课程信息至系统，系统将课程与班级信息显示到学生主页。 |
| 备选事件流 | 1a. 学生未选择任何课程     1.提示重新选择     2.学生重新提交选课信息  2a.系统判断学生所选上课的课程不存在或不被允许选择     1.提示重新选择     2.学生重新提交选课信息 |

## 6.7.2业务流程（顺序图）

### 

### 活动图



### 6.7.3界面设计

### 

接口：selection\_student

### 6.7.4接口设计

接口：selection\_student

* API请求地址： 接口基本地址/api/courses/selection
* 接口描述：该接口用于学生在选课平台上选择课程。学生可以通过此接口提交选课请求，系统将处理请求并更新学生的课程列表。
* 请求方式 ： POST
* 响应参数

| **参数名** | **类型** | **描述** |
| --- | --- | --- |
| success | Boolean | 请求是否成功 |
| message | String | 响应信息 |
| courses | Array | 学生当前已选课程的列表 |

* 请求参数

| **参数名** | **类型** | **位置** | **必填** | **描述** |
| --- | --- | --- | --- | --- |
| studentId | String | Body | 是 | 学生的唯一标识符 |
| courseId | String | Body | 是 | 课程的唯一标识符 |

* 响应示例：

{

"success": true,

"message": "选课成功",

"courses": [

{

"courseId": "C001",

"courseName": "数据库原理",

"credits": 3,

"status": "已选"

},

{

"courseId": "C002",

"courseName": "计算机网络",

"credits": 2,

"status": "已选"

}

// 其他已选课程...

]

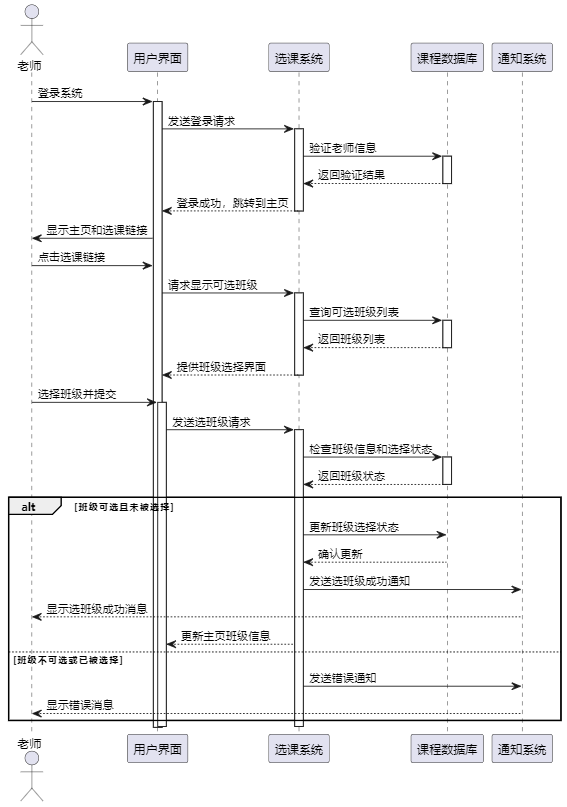
}

# 6.8 [“选课\_老师”用例](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E7%94%A8%E4%BE%8B\\%E9%80%89%E8%AF%BE_%E8%80%81%E5%B8%88.md),[界面](https://ssw383318348.github.io/is_analysis_pages/test6/%E9%80%89%E8%AF%BE_%E8%80%81%E5%B8%88.html)

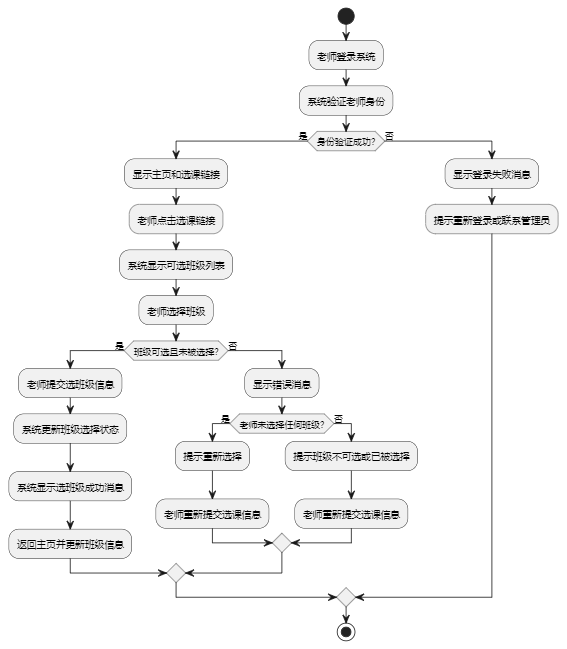
### 6.8.1用例规约

| **用例名称** | **选课\_老师** |
| --- | --- |
| 功能 | 老师选课平台 |
| 参与者 | 老师 |
| 前置条件 | 老师需要先登录，登录成功后，跳转到主页，点击选课链接，选择上课班级 |
| 后置条件 | 成功选择并提交上课班级的信息后，返回主页 |
| 主事件流 | 1. 老师选择要上课的班级 2.系统判断班级信息是否存在，是否被选择 3.老师选择并提交上课班级信息至系统，系统将课程与班级信息显示到老师主页。 |
| 备选事件流 | 1a. 老师未选择任何课程     1.提示重新选择     2.老师重新提交选课信息  2a.系统判断老师所选班级的课程不存在或不被允许选择     1.提示重新选择     2.老师重新提交选课信息 |

### 6.8.2业务流程（顺序图）



#### 活动图



### 6.8.3界面设计



接口：selection\_teacher

### 6.8.4接口设计

接口1：[selection\_teacher](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E6%8E%A5%E5%8F%A3\\setPassword.md)

用例： [选课\_老师](file:///D:\\CDUcourse_isAnalysis\\100ssw\\test6\\%E7%94%A8%E4%BE%8B\\%E4%BF%AE%E6%94%B9%E5%AF%86%E7%A0%81.md)

* 功能： 修改（设置）用户的密码。
* 权限： 学生/老师：修改自己的密码，必须先登录。
* API请求地址： 接口基本地址/v1/api/ selection\_teacher
* 请求方式 ： POST
* 请求实例：

{

"access\_token": $access\_token,

"lesson\_id": 1,

"class\_id": 2

}

* 请求参数说明:

| **参数名称** | **说明** |
| --- | --- |
| access\_token | 用于验证请求合法性的认证信息 |
| lesson\_id | 课程的id |
| class\_id | 班级的id |

* 返回实例：

{

"info": "选课成功",

"status": true,

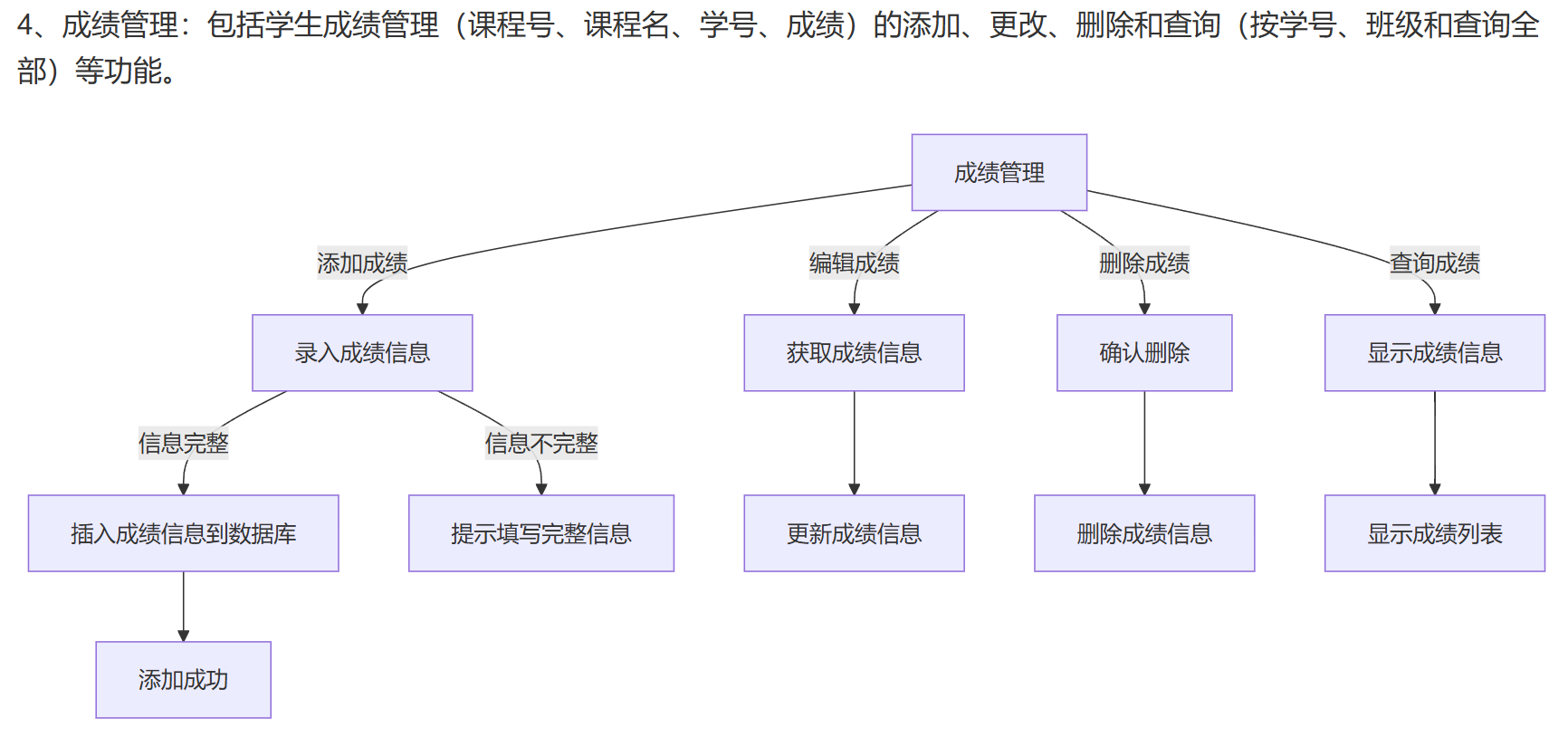
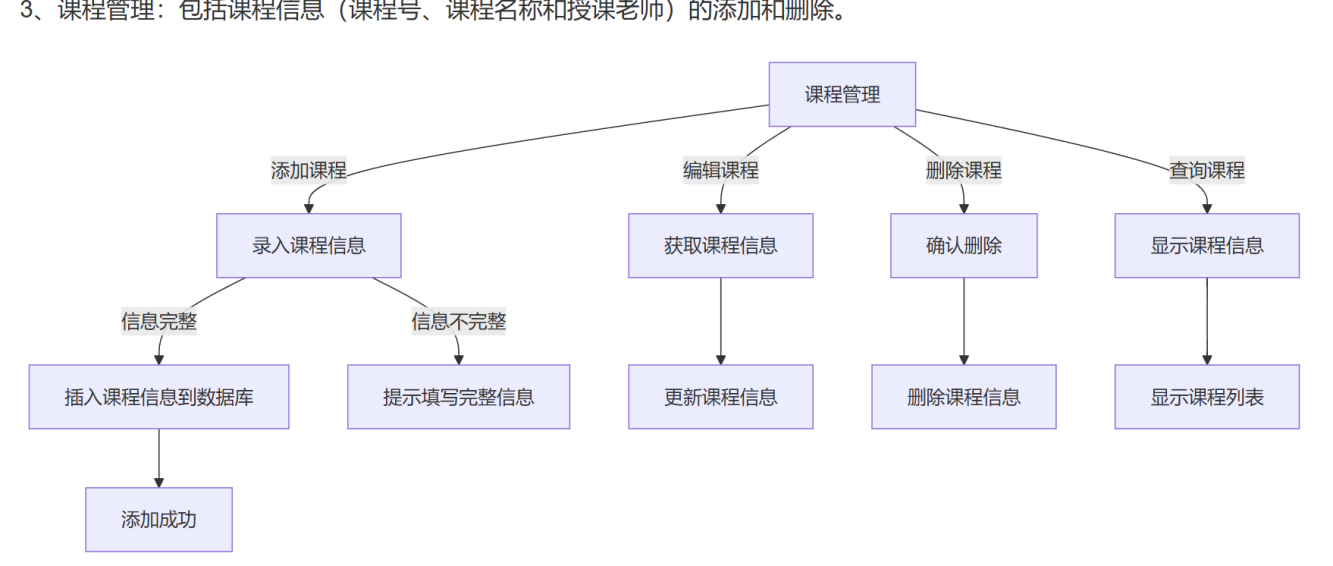
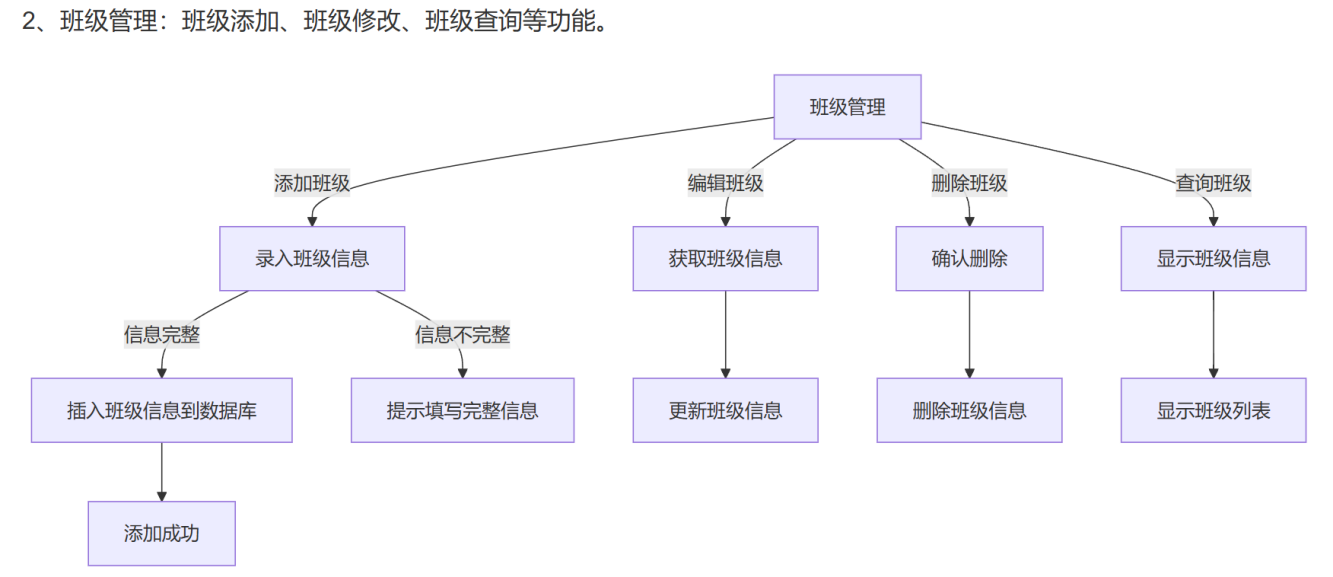
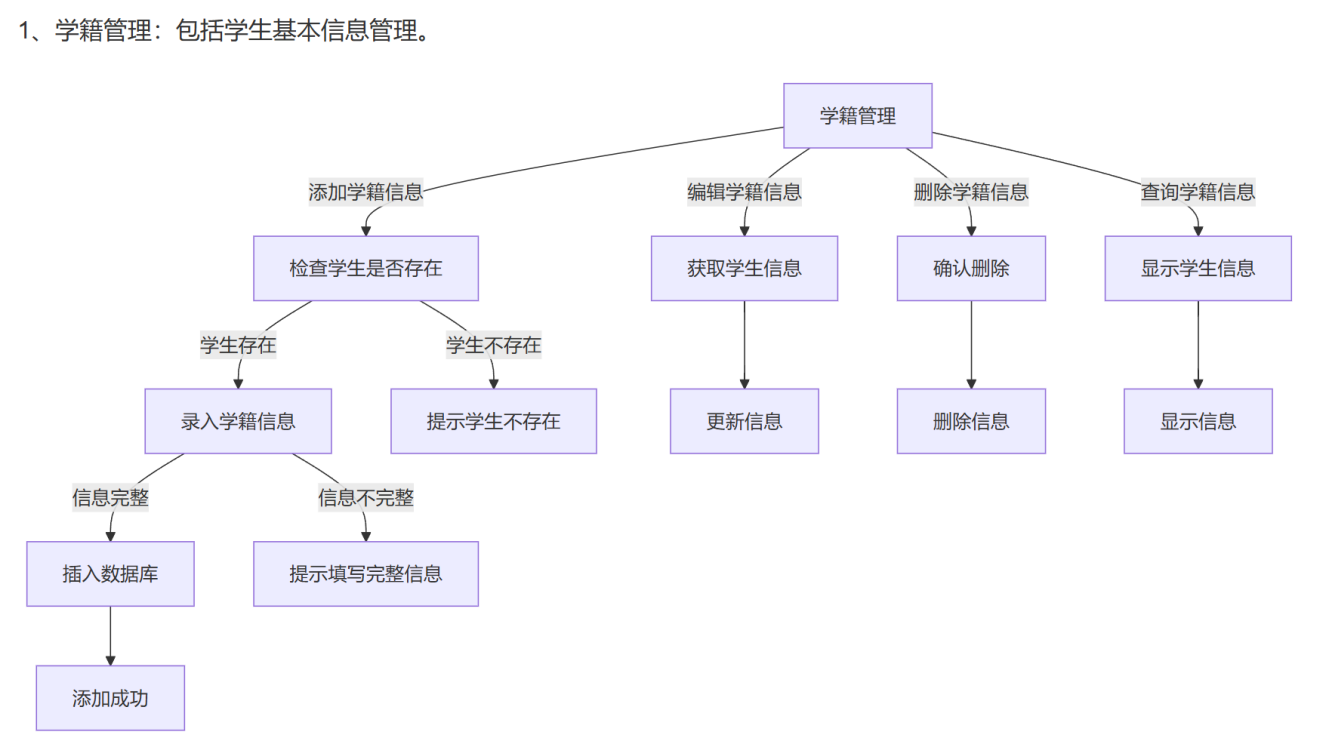
"code": 200

}

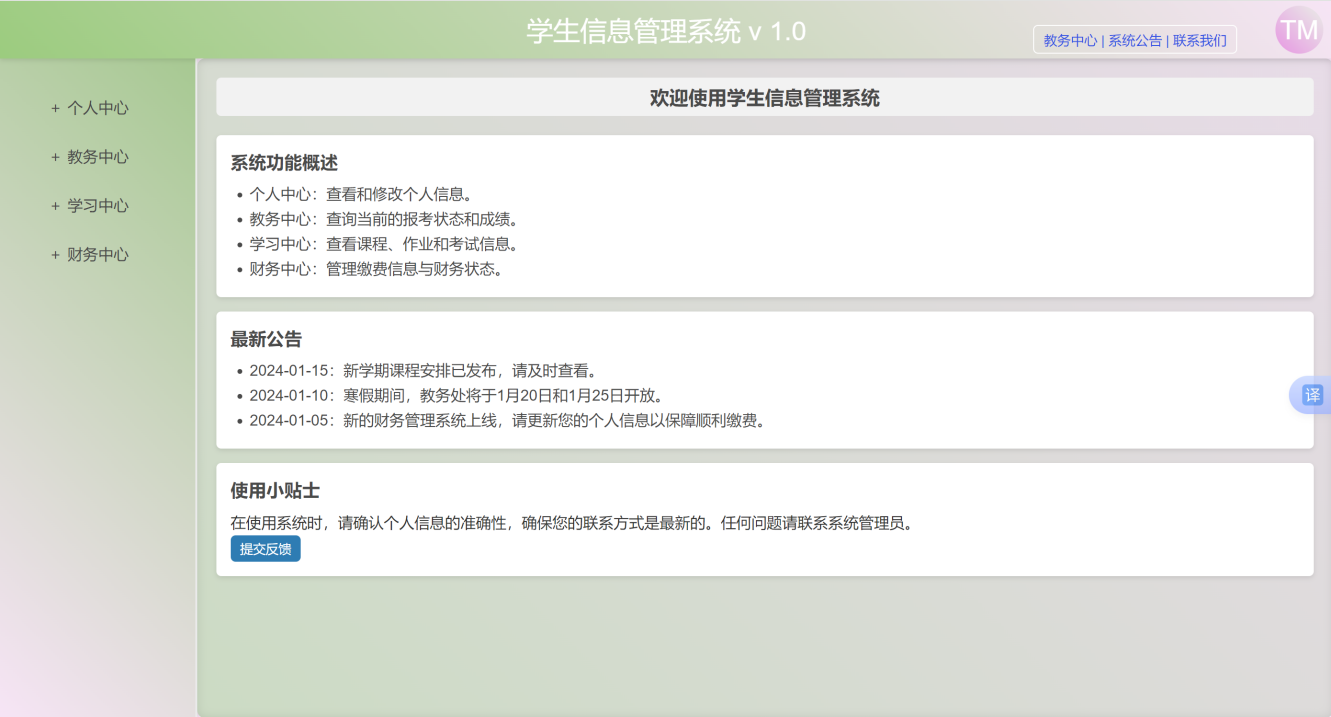
* 返回参数说明：

| **参数名称** | **说明** |
| --- | --- |
| info | 返回结果说明信息 |
| status | bool类型，true表示正确的返回，false表示有错误 |
| code | 返回码 |

1. 基本信息管理模块



1. **学生信息管理系统原型图**



# 系统架构设计

## 总体架构：系统采用前后端分离的架构模式，前端使用HTML、CSS、JavaScript等技术进行开发，后端使用Java语言进行开发，数据库采用MySQL。

## 前端设计：前端页面采用响应式设计，支持多种设备和浏览器。页面布局清晰，操作简便，用户体验良好。

## 后端设计：后端采用Spring Boot框架进行开发，实现RESTful API接口，支持前后端分离。后端服务采用微服务架构，便于系统的扩展和维护。

## 数据库设计：数据库采用MySQL关系型数据库，设计合理的表结构和索引，提高查询效率。同时，采用数据库事务和锁机制，确保数据的完整性和一致性。

**十、安全性考虑**

系统安全性是系统设计的重要方面之一。为了确保系统的安全性，我们采取以下措施：

### 数据加密：对敏感数据进行加密存储，防止数据泄露。

### 访问控制：采用基于角色的访问控制机制，确保不同用户角色的权限合理。

### 日志记录：记录用户的操作日志，便于追踪和审计。

### 安全审计：定期对系统进行安全审计，发现潜在的安全隐患并及时修复。

**十一、测试计划**

测试是确保系统质量和稳定性的重要环节。我们制定了以下测试计划：单元测试：对系统的各个模块进行单元测试，确保模块功能正确。

* 集成测试：将各个模块集成在一起进行测试，确保系统整体功能正确。
* 性能测试：对系统进行性能测试，包括响应时间、吞吐量等指标的测试。
* 安全测试：对系统进行安全测试，包括漏洞扫描、渗透测试等。
* 用户验收测试：邀请用户进行验收测试，收集用户反馈并进行改进。

**十二、总结与展望**

通过本次学生信息管理系统的设计与开发，我们实现了对学生信息的全面、高效、准确管理。系统具有友好的用户界面和强大的功能，能够满足学校和学生的需求。未来，我们将继续优化系统功能，提高系统性能和安全性，为用户提供更好的服务体验。同时，我们也将继续关注教育行业的发展趋势和技术变革，为系统的未来发展做好规划和准备。