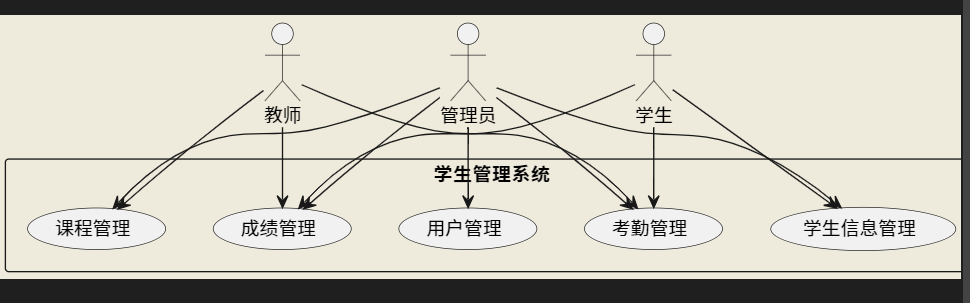
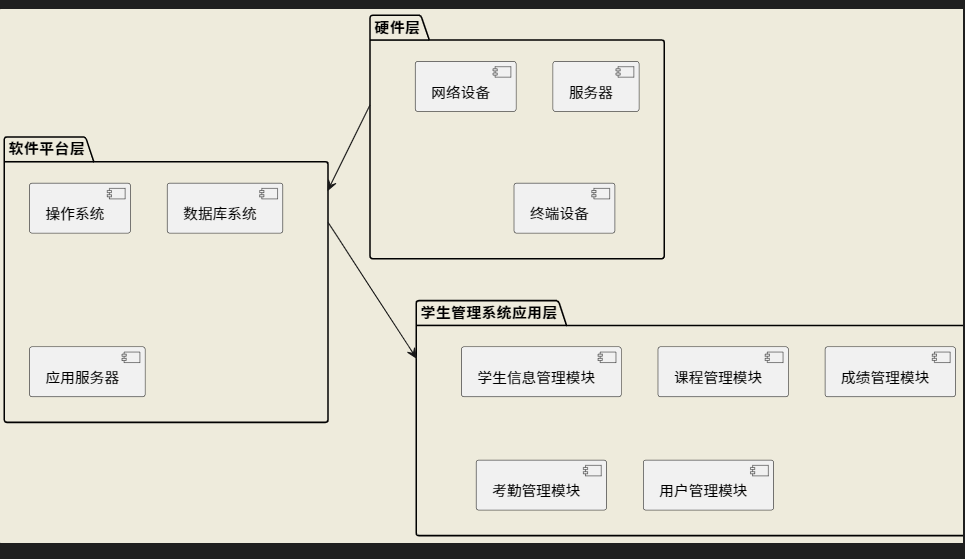
**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 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 | 赵梓锐 | 学号 | 赵梓锐 |

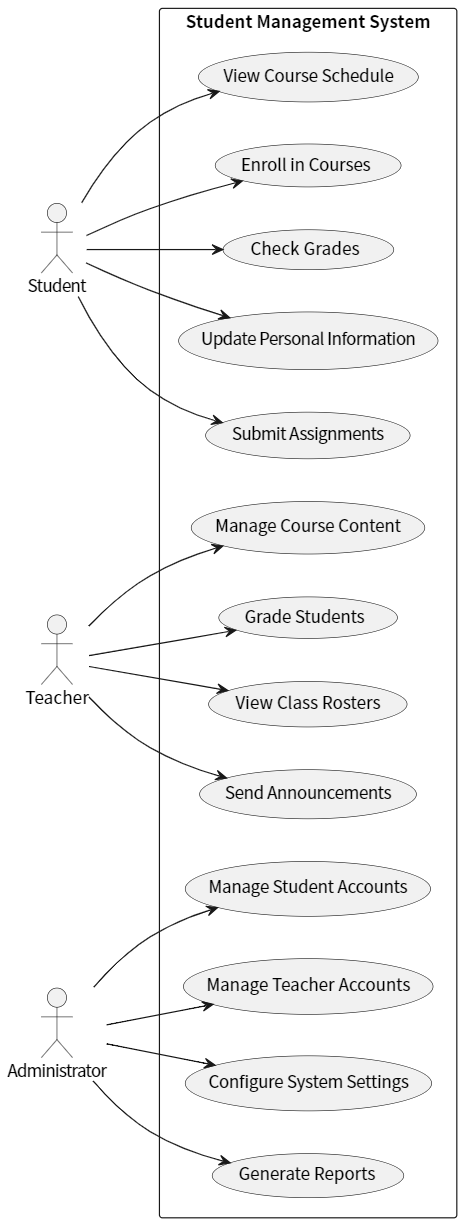
|  |  |  |  |
| --- | --- | --- | --- |
| **评分项** | **评分标准** | **满分** | **得分** |
| 文档整体 | 文档内容详实、规范，美观大方，内容: 可行性分析、需求分析、概要设计、详细设计。 | 20 |  |
| 用例图及规约 | 用例图完整，准确，能够完全体现需求 | 10 |  |
| 顺序图与活动图 | 顺序图能够完全描述用例的设计思路和业务流程，活动图能够描述部分算法的流程。 | 10 |  |
| 类图 | 类图能够完整，准确反映业务的需数据的组织结构 | 10 |  |
| 数据库 | 数据库表的设计来自于类图，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

# 2. 系统总体结构



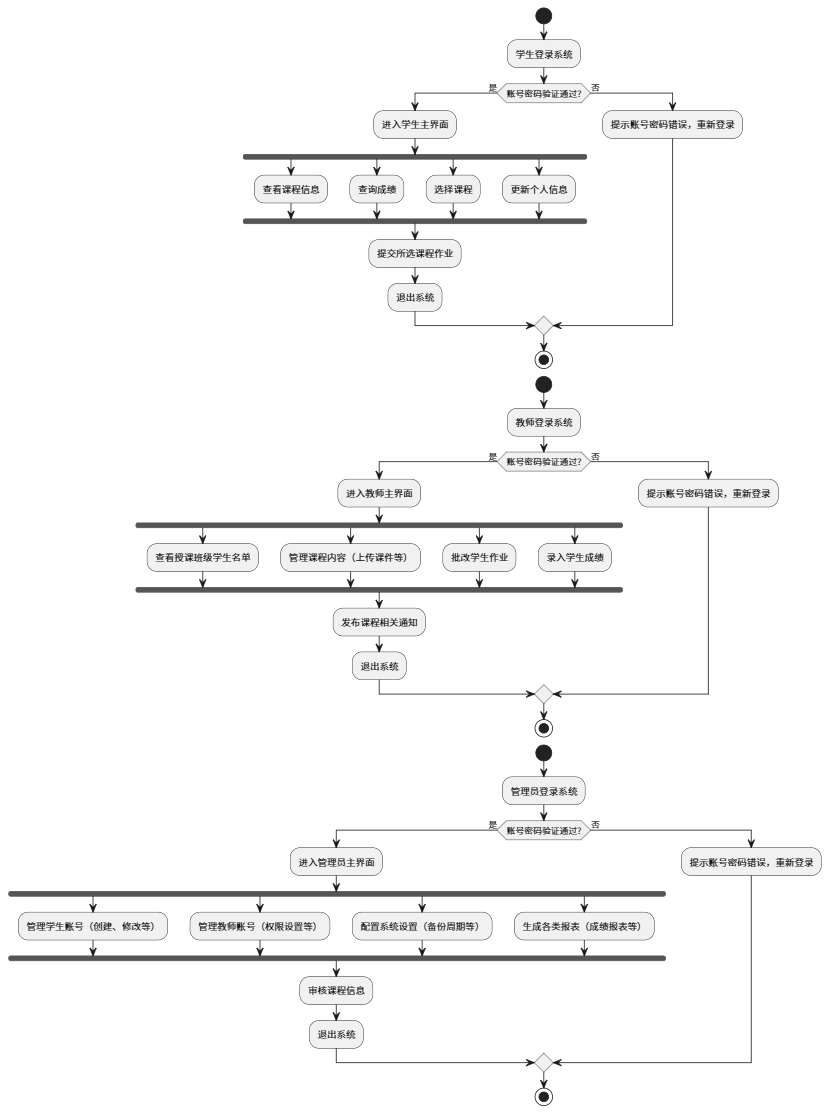
# 3. 用例图设计



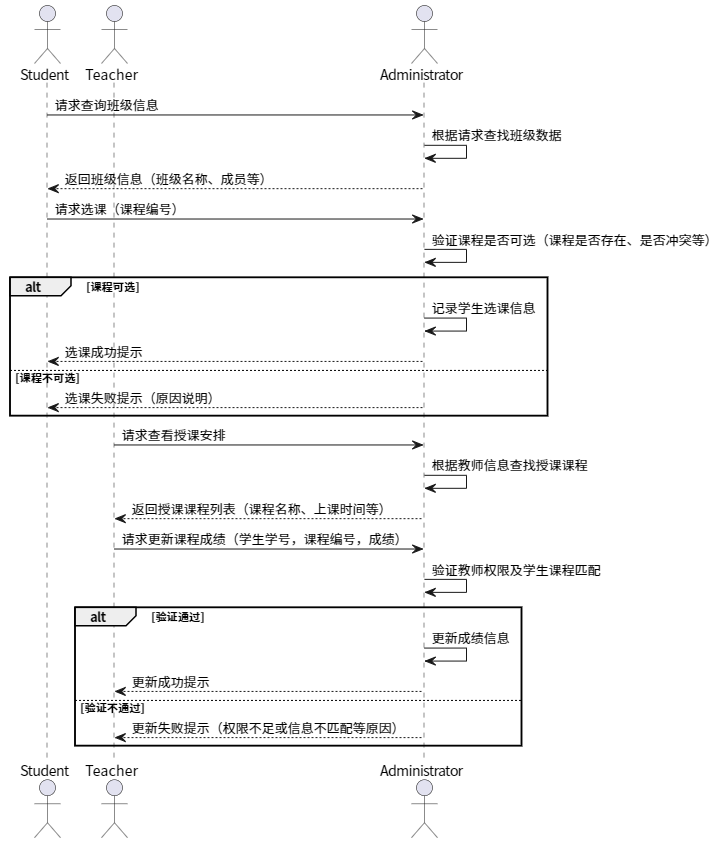
# 4. 类图设计

# 

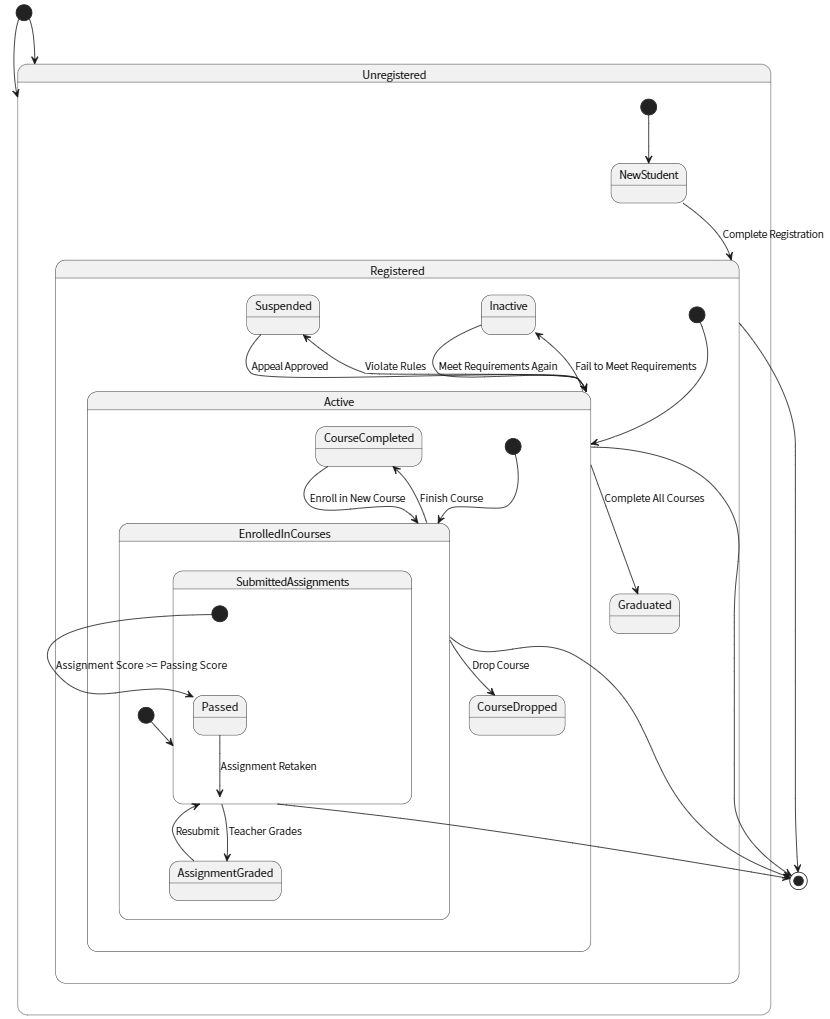
活动图



时序



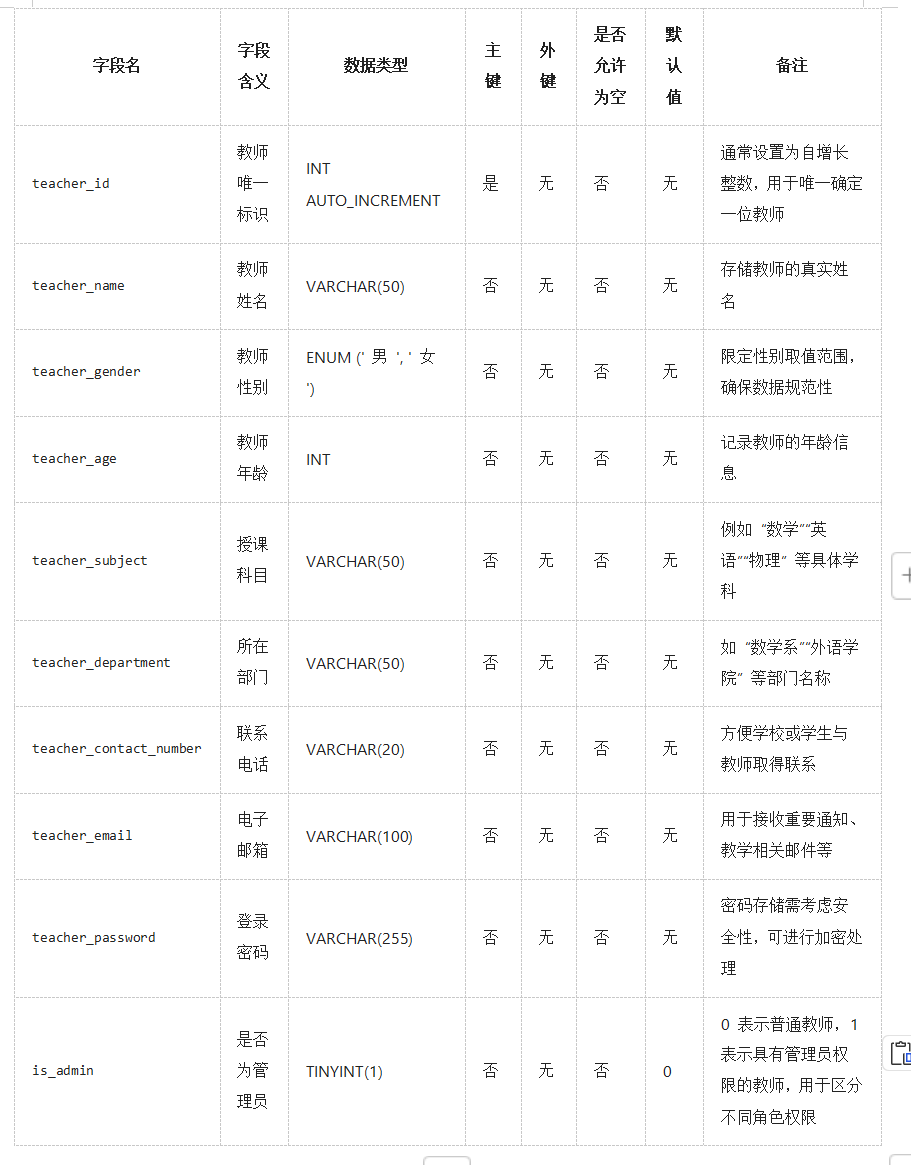
状态图



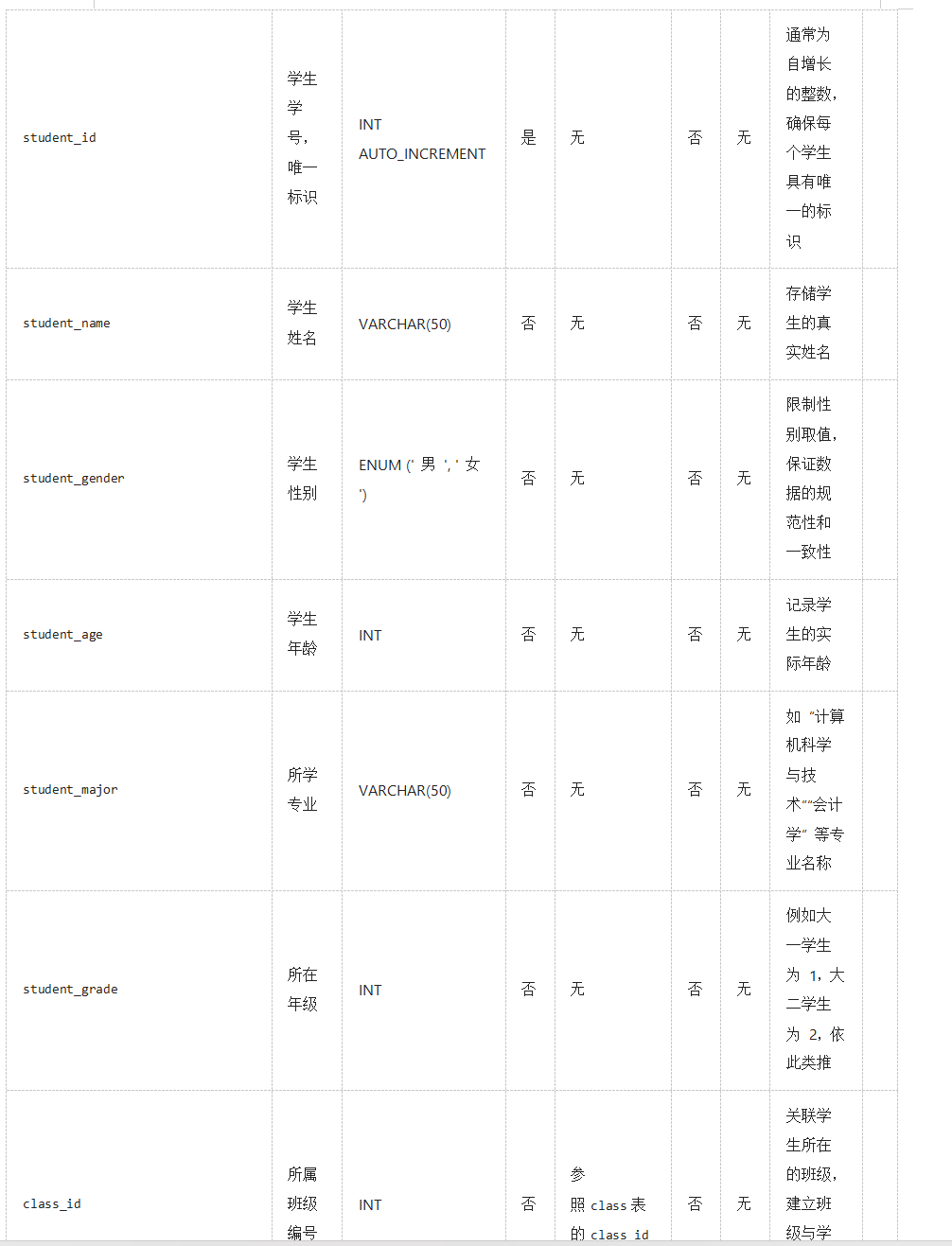
# 5. 数据库设计

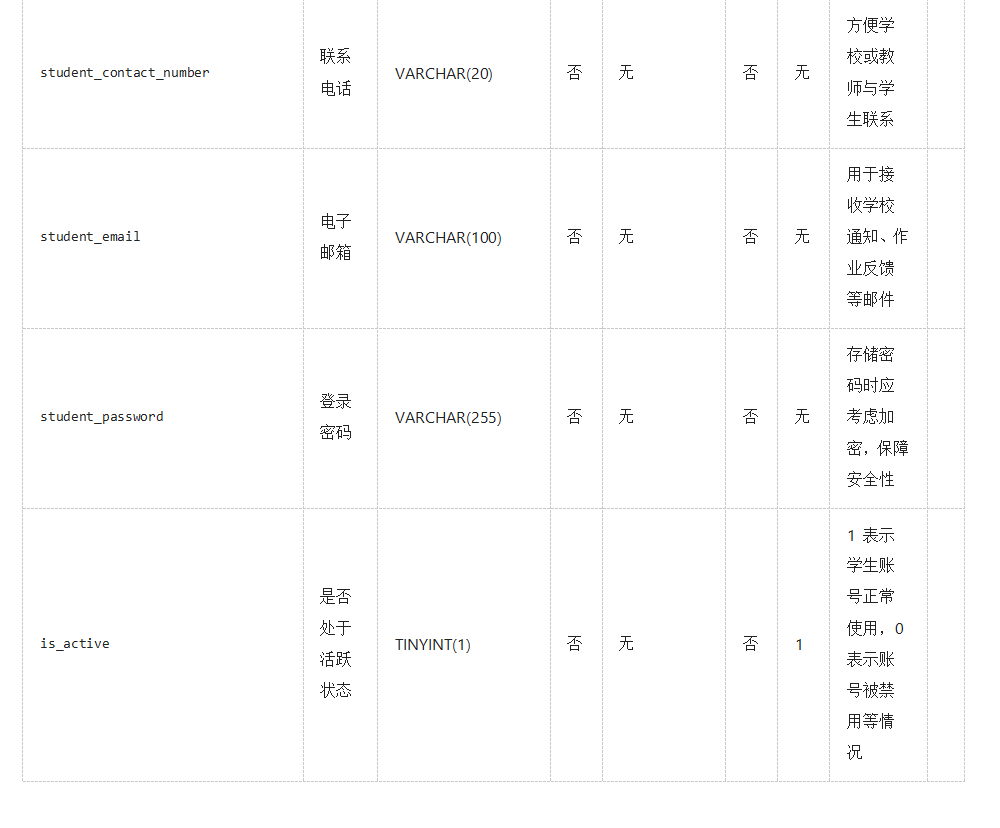
# 5. 数据库设计

## TEACHERS表（老师表）



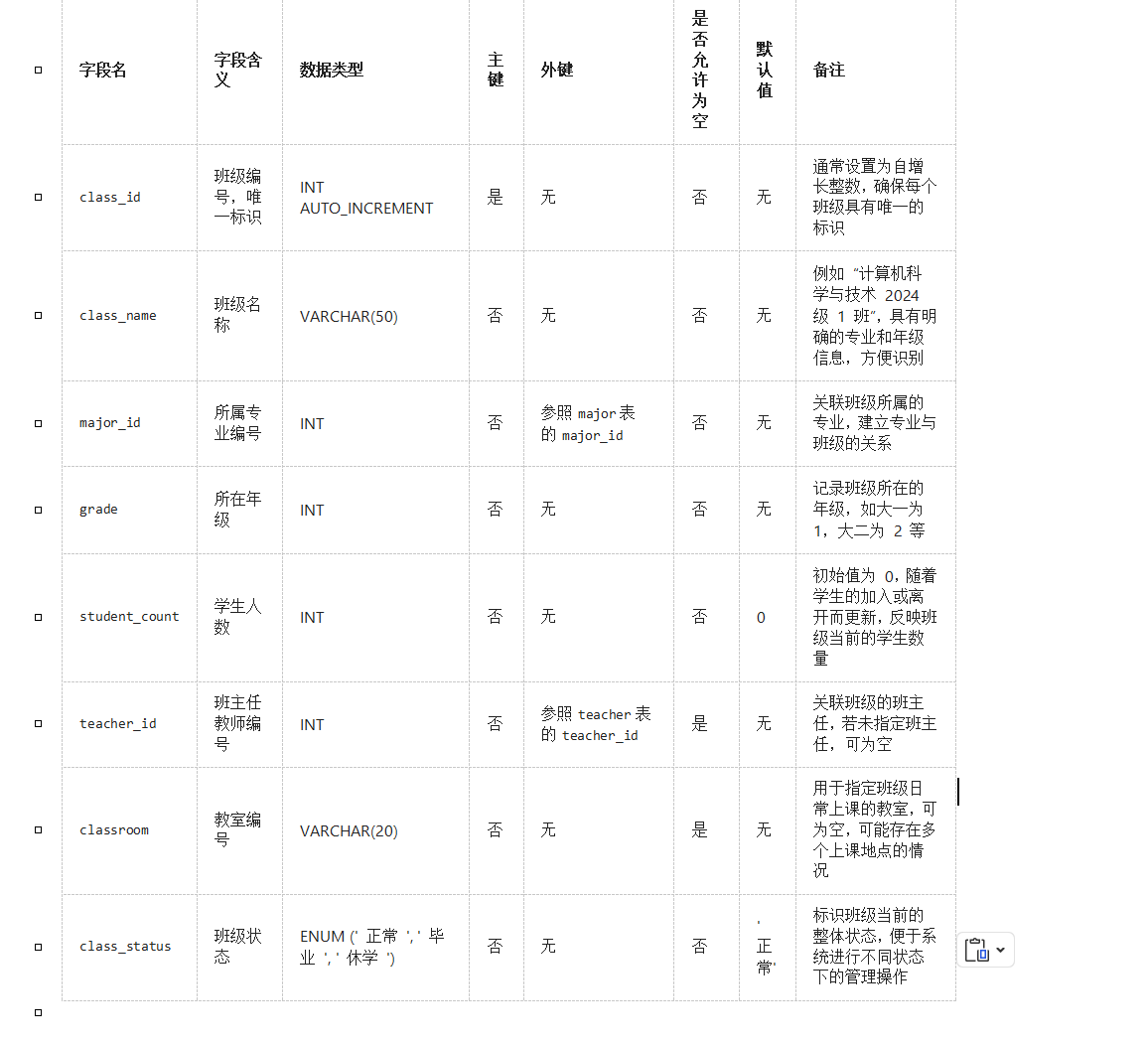
STUDENTS表（学生表）



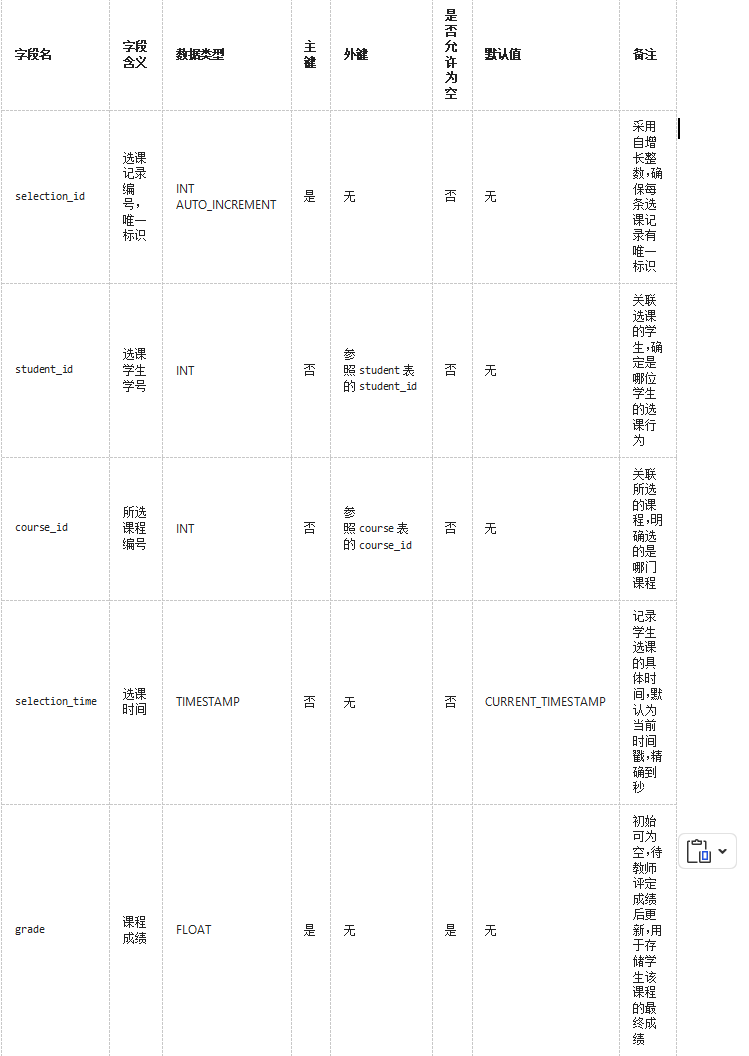


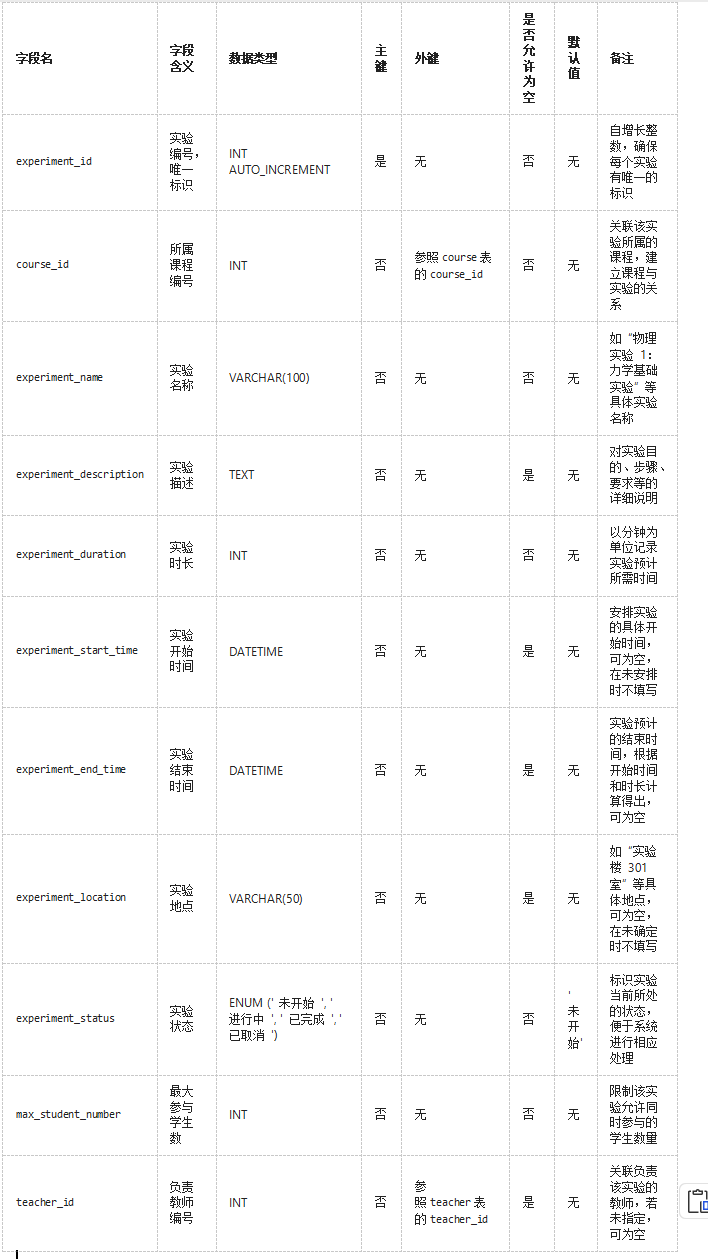
## LESSON表（课程表）屏幕截图 2024-12-22 154746屏幕截图 2024-12-22 154741

CLASS表（班级表）



## CHOOSE\_LESSON表（选课表）



TESTS表（实验项目表）

## GRADES表（学生实验成绩表）

| **字段名** | **字段含义** | **数据类型** | **主键** | **外键** | **是否允许为空** | **默认值** | **备注** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| grade\_id | 成绩记录编号，唯一标识 | INT AUTO\_INCREMENT | 是 | 无 | 否 | 无 | 自增长整数，确保每条成绩记录有唯一标识 |
| student\_id | 学生学号 | INT | 否 | 参照 student 表的 student\_id | 否 | 无 | 关联对应的学生，确定是哪位学生的成绩 |
| experiment\_id | 实验编号 | INT | 否 | 参照 experiment 表的 experiment\_id | 否 | 无 | 关联对应的实验，明确是哪次实验的成绩 |
| grade | 实验成绩 | FLOAT | 否 | 无 | 否 | 无 | 记录学生该实验的成绩，可根据实际评分标准设定范围，如 0 - 100 分 |
| grade\_comment | 成绩评语 | VARCHAR(200) | 否 | 无 | 是 | 无 | 教师对学生实验表现的评语，可为空 |
| submission\_time | 成绩提交时间 | TIMESTAMP | 否 | 无 | 否 | CURRENT\_TIMESTAMP | 记录教师提交成绩的时间，默认为当前时间戳 |

## AVERAGE\_SCORE表（实验平均成绩表）

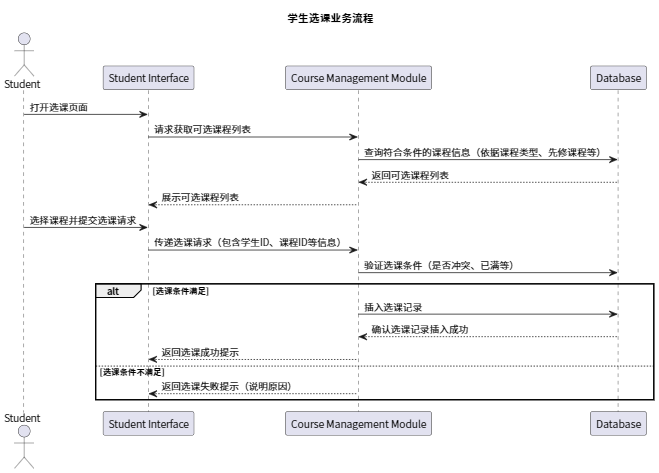
| **字段名** | **字段含义** | **数据类型** | **主键** | **外键** | **是否允许为空** | **默认值** | **备注** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| average\_grade\_id | 平均成绩记录编号，唯一标识 | INT AUTO\_INCREMENT | 是 | 无 | 否 | 无 | 自增长整数，用于区分每条平均成绩记录 |
| experiment\_id | 实验编号 | INT | 否 | 参照 experiment 表的 experiment\_id | 否 | 无 | 关联对应的实验，确定是哪次实验的平均成绩 |
| average\_grade | 实验平均成绩 | FLOAT | 否 | 无 | 否 | 无 | 存储该实验所有学生成绩的平均值，保留一定小数位数，如两位小数 |
| total\_student\_count | 参与学生总数 | INT | 否 | 无 | 否 | 无 | 记录参与该实验的学生数量，用于计算平均成绩和统计分析 |
| grade\_statistics | 成绩统计信息（如各分数段人数分布） | VARCHAR(200) | 否 | 无 | 是 | 无 | 以特定格式存储成绩统计情况，例如 “0 - 59 分：5 人；60 - 79 分：10 人；80 - 100 分：8 人”，方便后续分析成绩分布情况 |
| update\_time | 数据更新时间 | TIMESTAMP | 否 | 无 | 否 | CURRENT\_TIMESTAMP | 记录该平均成绩数据的最后更新时间，默认为当前时间戳，每次成绩有变化时自动更新 |

# 6. 用例及界面详细设计

### 6.1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业务流程（顺序图）



### 6.3界面设计



### 6.4接口设计

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

用例： [修改密码](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/setPassword
* 请求方式 ： POST
* 请求实例：

{

"user\_id":1234,

"password":AF#W@#AAAASDF

}

* 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：

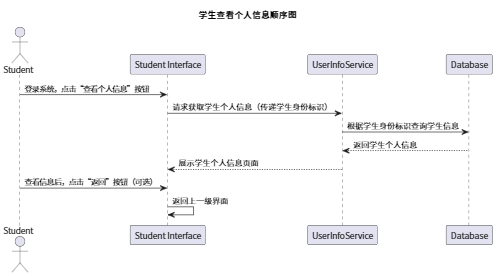
| **参数名称** | **说明** |
| --- | --- |
| status | bool类型，true表示正确的返回，false表示有错误 |
| info | 返回结果说明信息 |

## 6.2 [“修改用户信息”用例](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),[界面](https://ssw383318348.github.io/is_analysis_pages/test6/%E4%BF%AE%E6%94%B9%E7%94%A8%E6%88%B7%E4%BF%A1%E6%81%AF.html)

### 6.2.1用例规约

| **用例名称** | **修改用户信息** |
| --- | --- |
| 功能 | 修改用户的GitHub用户名称 |
| 参与者 | 学生，老师 |
| 前置条件 | 必须先登录，并且查看用户现有的GitHub用户名 |
| 后置条件 |  |
| 主事件流 | 1.用户填写GitHub用户名称  2.用户提交修改信息  3.系统存储修改后的GitHub用户名称 |
| 备选事件流 | 1a. 如果用户输入的GitHub用户名称为空     1.系统清空用户的GitHub用户名称 |

### 6.2.2业务流程（顺序图）



### 6.2.4接口设计

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GitHub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GitHub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 |

## **需求分析**

### **功能需求**

#### **学生信息管理**

涵盖基本信息（姓名、性别、出生日期、民族等）、学籍信息（学号、入学时间、专业、班级等）、联系方式（电话、邮箱）录入、修改、删除、查询（按单一或多条件精准检索）、导出打印，跟踪学籍异动（休学、复学、转学等）全程。

#### **课程管理**

课程信息（课程名、课程编号、学分、授课教师等）维护，支持排课（安排上课时间、教室、周次），关联学生选课记录，统计课程选修人数、分析热门冷门课程，生成课程表供师生查看下载。

#### **成绩管理**

教师录入成绩（平时、期中、期末按比例合成），系统自动计算总分、排名、学分绩点；学生在线查询个人成绩、成绩分析报告；管理人员审核成绩、处理补考、重修事务，导出成绩统计报表作教学评估用。

#### **考勤管理**

教师移动端（APP 或小程序）点名考勤（定位、蓝牙、扫码等签到方式），记录迟到、早退、旷课；后台统计个人、班级考勤率，分析考勤趋势，预警异常考勤学生，关联奖惩机制。

#### **用户管理**

创建不同角色账号（管理员、教师、学生），分配对应权限（管理员全功能、教师课程成绩等、学生个人信息成绩查询等），支持密码重置、账号冻结解封，保障系统安全访问。

### **性能需求**

#### **响应时间**

日常操作（信息查询、录入）响应时间控制在 1 - 3 秒，复杂查询（多表关联、大数据量统计）不超 5 秒，确保操作流畅，避免用户等待烦躁，提升使用体验。

#### **吞吐量**

预估系统高峰并发用户量（开学季学籍处理、期末成绩录入查询），设计满足至少 [X] 并发用户，保障高流量下系统稳定，数据读写正常，不出现卡顿崩溃。

#### **可靠性**

系统具备容错机制，应对硬件故障（服务器宕机）、软件错误（代码异常）、网络中断，自动切换备份、记录错误日志；定期数据备份（日备、周备）防止数据丢失，年故障停机时间控制在 [X] 小时内，保障教学管理持续运行。

### **数据需求**

#### **数据存储**

存储学生、课程、教师、成绩、考勤等多元数据，预估初始数据量 [X] GB，年增长 [X] GB，设计合理数据库表结构（学生表含多字段、关联课程成绩表），用索引优化查询，规划充足存储空间（本地硬盘或云端存储）。

#### **数据安全**

传输加密（HTTPS 协议防数据窃取篡改），存储加密敏感数据（身份证号、成绩）；严格用户权限管理，限制越权访问，定期审计数据操作日志，追踪异常行为，维护数据保密性、完整性、可用性。

# **概要设计**

## **一、引言**

本概要设计旨在为学生管理系统提供一个高层次的设计蓝图，明确系统的功能模块、数据库结构、系统架构以及运行平台等关键方面，为后续的详细设计和开发工作奠定基础，确保系统能够满足用户的需求，具备良好的性能、可扩展性和可维护性。

## **二、功能模块设计**

### **（一）学生信息管理模块**

* ****学生信息录入****：允许管理员录入学生的基本信息，包括姓名、性别、出生日期、身份证号、民族、家庭住址、联系方式、入学时间、专业、班级等。
* ****信息查询与修改****：支持根据多种条件（如学号、姓名、班级等）查询学生信息，并对错误或变更的信息进行修改。
* ****学籍异动处理****：对学生的休学、复学、转学、退学等学籍变动情况进行记录和管理，更新相应的学籍状态和信息。

### **（二）课程管理模块**

* ****课程信息维护****：教师或管理员可以添加、修改、删除课程信息，如课程名称、课程编号、学分、课程类型（必修课、选修课）、授课教师等。
* ****排课管理****：根据教学计划和资源情况，安排课程的上课时间、地点、周次等信息，生成课程表。
* ****选课管理****：学生在规定的选课时间内，根据专业要求和个人兴趣选择课程，系统自动处理选课冲突，并记录学生的选课结果。

### **（三）成绩管理模块**

* ****成绩录入****：教师在学期末录入学生的考试成绩，包括平时成绩、期中成绩、期末成绩等，系统根据预设的权重计算总评成绩。
* ****成绩统计与分析****：对学生成绩进行统计分析，如计算班级平均分、最高分、最低分、成绩分布情况等，生成成绩统计报表，为教学评估提供数据支持。
* ****成绩查询与导出****：学生可以查询自己的成绩，管理员和教师可以查询和导出学生的成绩数据，以便进行教学管理和家长沟通。

### **（四）考勤管理模块**

* ****考勤记录****：教师通过点名、扫码等方式记录学生的考勤情况，包括出勤、迟到、早退、旷课等，系统自动记录考勤时间和地点。
* ****考勤统计与报表****：对学生的考勤数据进行统计分析，生成考勤报表，如考勤日报、周报、月报等，方便教师和管理员了解学生的出勤情况，对考勤异常的学生进行预警和处理。

### **（五）用户管理模块**

* ****用户账号创建****：管理员为教师、学生和其他管理人员创建账号，并分配初始密码。
* ****权限管理****：根据用户角色（管理员、教师、学生）设置不同的系统权限，确保用户只能访问和操作其授权范围内的功能模块和数据。
* ****密码重置与修改****：用户可以通过邮箱或手机号码重置密码，也可以自行修改密码，保障账号安全。