|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于jQuery大学生上课打卡系统 | | |
| 课 程 | 软件工程 | | |
| 学 院 | 信息工程学院 | | |
| 专 业 | 软件技术 | 年级 | 2023级 |
| 学生姓名 | 王官爱民 | 学号 | 012322100010 |

|  |  |  |  |
| --- | --- | --- | --- |
| **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 28 日

# 大学生上课打卡系统可行性研究报告

一、系统目标

为了提高大学生的课堂出勤率，加强教学管理，设计一个方便、高效的上课打卡系统。

二、系统功能

1. 学生打卡

- 学生可以通过手机 APP 或校园内的打卡设备进行上课打卡。

- 打卡时，系统自动记录打卡时间和地点。

- 学生可以查看自己的打卡记录，了解自己的出勤情况。

2. 教师管理

- 教师可以通过系统查看学生的打卡情况，了解学生的出勤情况。

- 教师可以对学生的出勤情况进行统计和分析，以便更好地进行教学管理。

3. 管理员管理

- 管理员可以对系统进行管理和维护，包括添加、删除学生和教师信息，设置打卡规则等。

- 管理员可以查看系统的使用情况和统计数据，以便更好地进行教学管理和决策。

三、系统架构

1. 前端

- 手机 APP：学生和教师可以通过手机 APP 进行打卡和查看打卡记录。

- 打卡设备：校园内设置打卡设备，学生可以通过刷卡或人脸识别等方式进行打卡。

2. 后端

- 服务器：负责存储学生和教师的信息、打卡记录等数据，并提供数据查询和统计功能。

- 数据库：存储系统的数据，包括学生和教师信息、打卡记录等。

四、系统流程

1. 学生打卡流程

- 学生打开手机 APP 或前往打卡设备。

- 学生通过手机 APP 扫描打卡设备上的二维码或通过刷卡、人脸识别等方式进行打卡。

- 系统自动记录打卡时间和地点，并将打卡信息发送到服务器。

- 学生可以在手机 APP 上查看自己的打卡记录。

2. 教师管理流程

- 教师登录系统，进入教师管理页面。

- 教师可以查看学生的打卡情况，包括打卡时间、地点和出勤情况。

- 教师可以对学生的出勤情况进行统计和分析，以便更好地进行教学管理。

3. 管理员管理流程

- 管理员登录系统，进入管理员管理页面。

- 管理员可以对系统进行管理和维护，包括添加、删除学生和教师信息，设置打卡规则等。

- 管理员可以查看系统的使用情况和统计数据，以便更好地进行教学管理和决策。

五、系统安全

1. 数据加密：对学生和教师的信息、打卡记录等数据进行加密存储，确保数据的安全性。

2. 权限管理：对学生、教师和管理员的权限进行严格管理，确保系统的安全性和稳定性。

3. 备份恢复：定期对系统的数据进行备份，以便在系统出现故障时能够及时恢复数据。

六、系统优势

1. 提高课堂出勤率：通过打卡系统，学生的出勤情况可以得到有效的监督和管理，从而提高课堂出勤率。

2. 方便快捷：学生可以通过手机 APP 或校园内的打卡设备进行打卡，方便快捷。

3. 数据分析：系统可以对学生的出勤情况进行统计和分析，为教师和管理员提供决策依据。

4. 提高教学管理效率：通过打卡系统，教师和管理员可以更加方便地了解学生的出勤情况，提高教学管理效率。

以上是一个大学生上课打卡系统的设计方案，具体的实现可以根据实际情况进行调整和优化。

# 需求分析

## 一、系统目标

本系统旨在为高校提供一种便捷、高效的大学生上课打卡管理方式，准确记录学生的出勤情况，提高教学管理效率，同时方便教师和学生随时了解出勤信息。

## 二、用户需求

1. 学生需求 - 能够快速、方便地进行上课打卡操作，避免繁琐的流程。 - 可以查看自己的出勤记录，了解自己的上课情况。 - 接收关于打卡异常的提醒，以便及时处理。 - 确保个人信息和打卡数据的安全。 2. 教师需求 - 实时查看学生的打卡情况，了解班级出勤状况。 - 能够导出学生的出勤报表，用于教学评估和成绩统计。 - 对打卡异常的学生进行标记和处理。 - 方便与学生沟通出勤问题。 3. 教学管理人员需求 - 对全校的上课打卡情况进行统计和分析。 - 生成各类出勤报表，为教学管理决策提供数据支持。 - 确保系统的稳定运行和数据安全。

三、功能需求

1. 打卡功能 - 学生可以通过手机 APP、校园卡或其他指定方式进行上课打卡。 - 系统能够准确记录打卡时间和地点。 - 支持多种打卡验证方式，如指纹识别、面部识别、二维码扫描等。 2. 出勤记录查询 - 学生和教师可以随时查询个人或班级的出勤记录。 - 记录应包括打卡时间、课程名称、授课教师等信息。 - 提供历史出勤记录的查询功能，方便回顾和统计。 3. 异常处理 - 对于打卡异常情况（如迟到、早退、未打卡等），系统应自动提醒学生和教师。 - 教师可以对异常情况进行标记和处理，如请假审批、迟到处罚等。 - 学生可以对异常情况进行申诉和说明。 4. 报表生成 - 教师和教学管理人员可以根据需要生成不同类型的出勤报表。 - 报表应包括班级出勤汇总、个人出勤明细、异常情况统计等内容。 - 支持报表的导出和打印功能。 5. 消息通知 - 系统可以向学生和教师发送关于打卡提醒、异常处理结果、课程变动等消息通知。 - 学生和教师可以设置消息接收方式和提醒频率。 6. 系统管理 - 教学管理人员可以对系统进行参数设置、用户管理、数据备份等操作。 - 确保系统的安全性和稳定性，防止数据泄露和系统故障。

## 四、性能需求

1. 响应时间 - 打卡操作的响应时间应在 1 秒以内。 - 出勤记录查询和报表生成的响应时间应在 3 秒以内。 2. 系统容量 - 系统应能够支持全校学生和教师的同时使用，保证高并发情况下的稳定运行。 3. 数据准确性 - 打卡数据应准确无误，避免出现重复打卡、漏打卡等情况。 - 出勤记录和报表的数据应与实际情况一致。

五、安全需求

1. 数据安全 - 对学生和教师的个人信息、打卡数据进行加密存储，防止数据泄露。 - 建立数据备份和恢复机制，确保数据的安全性和可靠性。 2. 系统安全 - 对系统进行权限管理，不同用户具有不同的操作权限。 - 防止非法用户的入侵和攻击，确保系统的稳定运行。

# 六、界面需求

1. 简洁明了 - 系统界面应简洁、直观，易于操作。 - 打卡流程应简单易懂，避免复杂的操作步骤。 2. 美观大方 - 界面设计应美观大方，符合大学生的审美需求。 色彩搭配协调，字体大小适中。 3. 兼容性 - 系统应兼容多种设备和操作系统，如手机、平板电脑、电脑等。

# 大学生上课打卡系统概要设计

1. 系统架构设计

1分层架构

表现层：负责与用户进行交互，接收用户输入并展示系统输出。包括学生端 APP、教师端 APP 和管理端 Web 页面，采用响应式设计以适应不同设备。

业务逻辑层：处理系统的核心业务逻辑，如打卡验证、出勤记录管理、异常处理、报表生成等。通过接口与表现层通信，调用数据访问层的方法获取和存储数据。

数据访问层：负责与数据库进行交互，执行数据的插入、查询、更新和删除操作。对业务逻辑层提供数据访问接口，实现数据的持久化存储。

2 系统组件

打卡模块：实现学生打卡功能，包括多种打卡方式的接口和验证逻辑。

出勤记录管理模块：处理出勤记录的查询、存储和历史记录管理。 异常处理模块：负责检测打卡异常情况，发送提醒通知，并处理学生申诉和教师标记。

报表生成模块：根据用户需求生成各类出勤报表，支持导出和打印。

消息通知模块：实现系统消息的发送和接收设置。

系统管理模块：提供系统参数设置、用户管理和数据备份等功能。

1. 数据库设计

1实体关系图（ER 图）

学生（Student）：包含学号、姓名、密码、专业、班级等属性，与出勤记录（AttendanceRecord）、打卡异常申诉（Appeal）有一对多关系。

教师（Teacher）：具有工号、姓名、密码、授课课程等属性，与课程（Course）、学生出勤记录有一对多关系。

课程（Course）：包括课程编号、课程名称、授课教师、上课时间、上课地点等属性，与学生、教师有多对多关系。

出勤记录（AttendanceRecord）：记录打卡时间、打卡地点、打卡方式等信息，关联学生和课程。

打卡异常（AttendanceException）：描述异常类型（迟到、早退、未打卡）、处理状态等，与学生、教师、出勤记录有关系。

报表（Report）：存储报表类型、生成时间、报表内容等，与教师、教学管理人员有关联。

消息通知（Message）：包含通知内容、发送时间、接收者等属性，与学生、教师有一对多关系。

2数据库表结构设计

#### 学生表（Student）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | 字段名 | 数据类型 | 主键 | 非空约束 | 描述 |
| **学生表** | 学号 | VARCHAR(50) | 是 | 是 | 唯一标识学生。 |
|  | 姓名 | VARCHAR(50) | 否 | 是 | 学生姓名。 |
|  | 专业 | VARCHAR(50) | 否 | 是 | 学生所学专业。 |
|  | 班级 | VARCHAR(50) | 否 | 是 | 学生所在班级。 |
|  | 联系电话 | VARCHAR(20) | 否 | 是 | 学生的联系电话 |

#### 教师表（Teacher）

|  |  |  |
| --- | --- | --- |
| 教师表 |  |  |
| 工号 | 主键，唯一标识教师。 |  |
| 姓名 | 教师姓名 |  |
| 密码 | 加密存储的密码 |  |
| 授课课程 | 存储教师授课的课程编号列表 |  |

#### 课程表（Course）



#### 出勤记录表（AttendanceRecord）



打卡异常表（AttendanceException）



报表表（Report）



#### 消息通知表（Message）

三 系统模块设计

#### 1打卡模块

打卡方式接口：定义不同打卡方式（如手机 APP 打卡、校园卡打卡、指纹识别打卡、面部识别打卡、二维码扫描打卡等）的统一接口，方便扩展和切换打卡方式。

打卡验证逻辑：根据选择的打卡方式，调用相应的验证接口进行打卡验证，验证通过后记录打卡时间和地点。

打卡记录存储：将打卡记录保存到出勤记录表中，同时检查是否存在异常情况，如有异常则触发异常处理流程。

#### 出勤记录查询模块

查询接口：提供学生查询个人出勤记录和教师查询班级出勤记录的接口，支持按时间范围、课程名称等条件查询。

记录获取与展示：根据查询条件从数据库中获取相应的出勤记录，并按照一定格式展示给用户，包括打卡时间、课程名称、授课教师等信息。

历史记录管理：实现对历史出勤记录的存储和查询功能，方便用户回顾和统计。

#### 异常处理模块

异常检测与提醒：定时检查出勤记录，判断是否存在迟到、早退、未打卡等异常情况，自动向学生和教师发送提醒通知。

教师处理界面：教师可以在界面上查看异常情况列表，对异常进行标记（如请假审批通过、迟到处罚等）和处理，并记录处理意见。

学生申诉功能：学生收到异常提醒后，可以在系统中提交申诉，说明情况，等待教师处理。

#### 报表生成模块

报表模板设计：根据不同的报表类型（班级出勤汇总、个人出勤明细、异常情况统计等）设计相应的报表模板，确定报表的格式和内容结构。

数据获取与填充：根据报表模板，从数据库中获取相关数据（如出勤记录、打卡异常记录等），并填充到报表模板中生成报表。

导出与打印功能：实现报表的导出为常见格式（如 PDF、Excel 等），并支持直接打印功能，方便教师和教学管理人员使用。

#### 消息通知模块

通知发送逻辑：根据系统中的业务流程（如打卡提醒、异常处理结果、课程变动等），生成相应的消息通知内容，并按照用户设置的接收方式（如短信、APP 推送、系统消息等）发送给学生和教师。

接收设置管理：学生和教师可以在系统中设置消息接收方式和提醒频率，以满足个人需求。

系统管理模块

参数设置：教学管理人员可以设置系统的一些参数，如打卡有效时间范围、异常判断标准等。

用户管理：对学生、教师和教学管理人员的用户信息进行管理，包括添加、删除、修改用户，分配角色和权限等。

数据备份与恢复：定期对系统数据进行备份，提供数据恢复功能，以防止数据丢失和损坏，确保数据的安全性和可靠性。

## 系统安全设计

数据加密

对学生和教师的个人信息（如学号、工号、姓名、密码等）以及打卡数据进行加密存储，采用安全的加密算法（如 AES、RSA 等），确保数据在存储和传输过程中的安全性。

#### 权限管理

定义不同的用户角色（学生、教师、教学管理人员），并为每个角色分配相应的操作权限。例如，学生只能进行打卡、查看自己的出勤记录和接收消息通知；教师可以查看班级出勤情况、处理异常、生成报表等；教学管理人员具有系统管理的全部权限。 采用基于角色的访问控制（RBAC）模型，在系统中实现权限验证机制，确保用户只能执行其权限范围内的操作。

#### 防止非法入侵和攻击

部署防火墙，阻止外部非法网络访问，限制系统的访问入口。 定期进行系统安全漏洞扫描和修复，及时更新系统软件和补丁，防止黑客利用漏洞入侵系统。 采用数据校验和验证码等技术，防止数据篡改和恶意攻击，如防止重复打卡、注入攻击等。

## 系统性能优化

缓存机制

在系统中引入缓存技术，如使用 Redis 缓存经常访问的数据（如课程信息、教师信息、学生基本信息等），减少数据库查询次数，提高系统响应速度。

数据库优化

对数据库表进行合理设计，建立适当的索引（如在学生表的学号字段、课程表的课程编号字段等建立索引），提高数据查询效率。 优化数据库查询语句，避免复杂的关联查询和全表扫描，确保数据库操作的高效性。

#### 异步处理

对于一些耗时的操作（如报表生成、数据备份等），采用异步处理方式，将任务放入消息队列中，由后台线程异步执行，避免影响系统的实时响应性能。 通过以上概要设计，大学生上课打卡系统将具备良好的架构、功能模块、数据库设计、安全性能和性能优化措施，能够满足高校对学生上课打卡管理的需求，提供高效、便捷、安全的服务。在后续的详细设计和开发过程中，将进一步细化和实现各个模块的具体功能，确保系统的顺利实施和运行。

# 层次图

# 

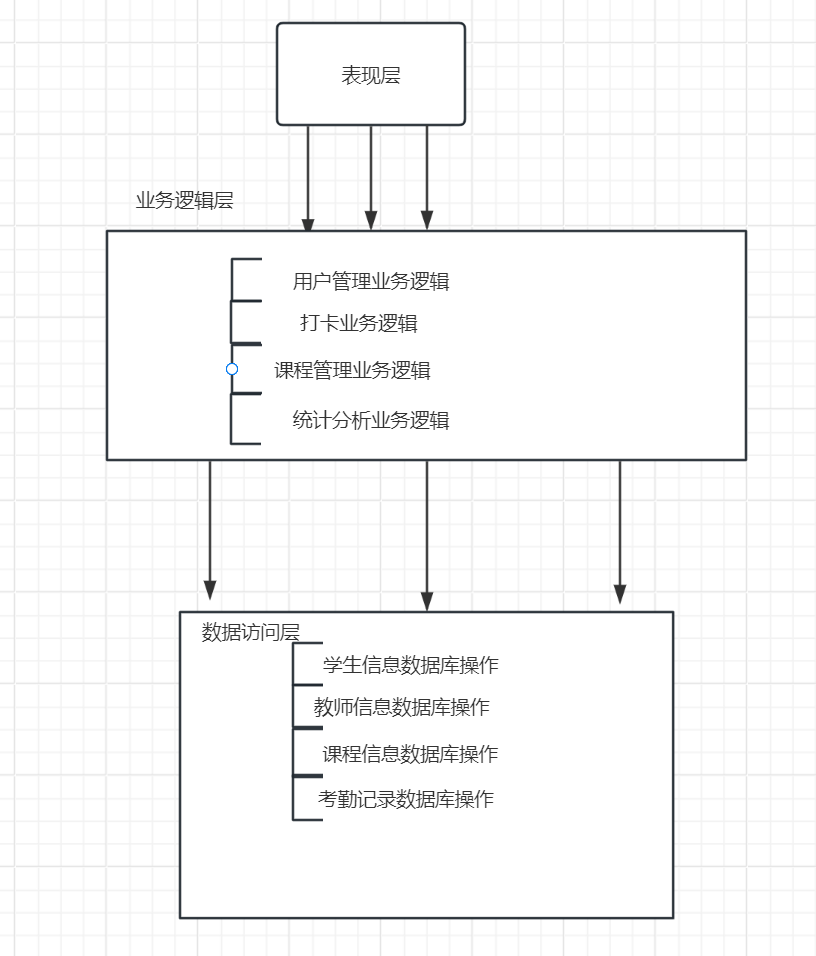
# ISO图

# 

### **结构图**

#### **整体结构**

* ****表现层（UI 层）****
  + 学生端界面（登录、打卡操作界面、打卡记录查看等）
  + 教师端界面（课程管理操作界面、考勤查看与统计界面等）
* ****业务逻辑层****
  + 用户管理业务逻辑（处理各类用户相关操作逻辑）
  + 打卡业务逻辑（实现不同打卡方式的逻辑判断、记录等）
  + 课程管理业务逻辑（课程相关增删改查等操作逻辑）
  + 统计分析业务逻辑（进行考勤数据统计、报表生成的逻辑）
* ****数据访问层****
  + 学生信息数据库操作（增删改查学生基本信息、打卡记录关联等）
  + 教师信息数据库操作（教师账号等相关数据操作）
  + 课程信息数据库操作（课程数据的存储、读取、更新等）
  + 考勤记录数据库操作（专门用于处理打卡记录、统计结果等数据的存储和查询）



### **用户接口设计**

* ****学生端接口****：
  + ****登录接口****：学生通过学号和密码登录系统，接口验证用户身份信息，若正确则返回登录成功及学生相关课程信息，包括课程名称、上课时间、上课地点等，以方便学生后续打卡操作及查看课程安排。例如，采用 RESTful API 设计，接口路径为 /student/login，请求方法为 POST，请求参数为 { "studentId": "学号", "password": "密码" }，成功响应数据为 { "status": "success", "studentInfo": { "name": "学生姓名", "courses": [ { "courseName": "课程 1", "classTime": "周一 9:00 - 10:30", "classroom": "教学楼 301" },... ] } }。
  + ****打卡接口****：支持多种打卡方式，如扫码打卡、定位打卡和手动打卡。扫码打卡时，学生扫描课程专属二维码，接口获取二维码信息并验证课程与学生当前所选课程的匹配性；定位打卡则获取学生的地理位置信息，判断是否在上课教室的有效范围内；手动打卡需教师在教师端操作后，学生端显示相应的手动打卡确认信息。以扫码打卡为例，接口路径为 /student/scanCheckIn，请求方法为 POST，请求参数为 { "qrCodeData": "二维码数据", "studentId": "学号" }，成功响应数据为 { "status": "success", "message": "打卡成功" }，若打卡失败则返回相应错误信息，如课程不匹配或已过打卡时间等。
  + ****打卡记录查询接口****：学生可查询自己的历史打卡记录，可按时间段（如本周、本月、本学期等）进行筛选查询。接口路径为 /student/checkInHistory，请求方法为 GET，可带查询参数如 { "startDate": "开始日期", "endDate": "结束日期" }，响应数据为 { "status": "success", "checkInRecords": [ { "courseName": "课程名称", "checkInTime": "打卡时间", "checkInMethod": "打卡方式" },... ] }。
* ****教师端接口****：
  + ****登录接口****：与学生端登录类似，教师通过账号和密码登录，接口验证成功后返回教师所授课程信息、班级学生名单等数据，方便教师进行课程管理和考勤操作。接口路径为 /teacher/login，请求方法和参数格式类似学生端登录接口，响应数据为 { "status": "success", "teacherInfo": { "name": "教师姓名", "courses": [ { "courseName": "课程 1", "classStudents": [ { "studentId": "学号 1", "studentName": "学生姓名 1" },... ] } ] } }。
  + ****课程管理接口****：包括课程信息的创建、编辑和删除操作。创建课程时，教师需填写课程名称、上课时间、上课地点、课程周期等信息，接口将数据保存到数据库，并返回课程创建成功信息及课程 ID。例如创建课程接口路径为 /teacher/createCourse，请求方法为 POST，请求参数为 { "courseName": "课程名称", "classTime": "上课时间", "classroom": "上课地点", "courseDuration": "课程周期" }，响应数据为 { "status": "success", "courseId": "课程唯一标识" }。编辑课程接口（/teacher/editCourse）和删除课程接口（/teacher/deleteCourse）类似，根据课程 ID 对相应课程信息进行修改或删除操作，并返回操作结果信息。
  + ****考勤管理接口****：教师可实时查看当前课程的学生打卡情况，包括已打卡人数、未打卡人数及学生名单，还可对考勤数据进行统计分析，如按周、月、学期统计学生的出勤天数、缺勤次数等。查看考勤接口路径为 /teacher/checkAttendance，请求方法为 GET，可带课程 ID 参数（{ "courseId": "课程 ID" }），响应数据为 { "status": "success", "attendanceData": { "checkedInStudents": [ { "studentId": "学号 1", "studentName": "学生姓名 1", "checkInTime": "打卡时间" },... ], "uncheckedInStudents": [ { "studentId": "学号 2", "studentName": "学生姓名 2" } ], "totalStudents": "班级总人数" } }。统计分析考勤接口（/teacher/analyzeAttendance）可根据教师选择的统计时间段和维度，返回相应的统计报表数据，如 { "status": "success", "attendanceReport": { "weekSummary": { "student1": { "presentDays": 3, "absentDays": 2 },... }, "monthSummary": { "averageAttendanceRate": "80%" },... } }。

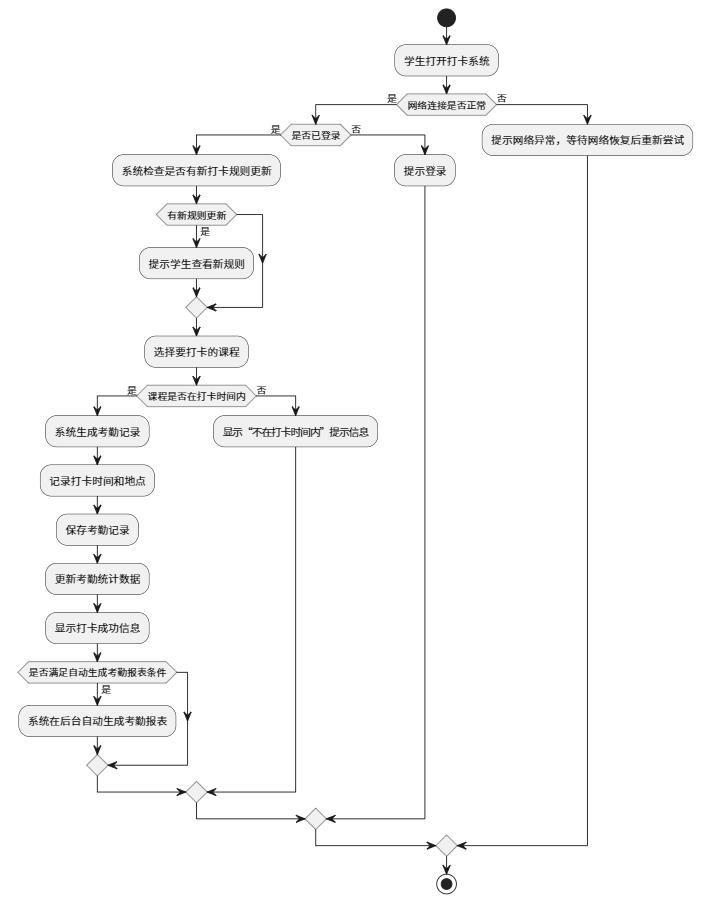
### **外部系统接口设计**

* ****教务管理系统接口****：
  + ****课程数据同步接口****：为保证上课打卡系统中的课程信息与学校教务管理系统中的课程数据保持一致，定时从教务管理系统获取最新的课程安排信息，包括新增课程、课程调整（如时间、地点变更）、课程取消等数据。接口采用定时任务触发，例如每天凌晨 2 点执行一次同步操作。接口路径为 /sync/courseData，请求方法为 GET，响应数据为教务管理系统中更新后的课程信息列表，格式为 { "courses": [ { "courseId": "课程 ID", "courseName": "课程名称", "classTime": "上课时间", "classroom": "上课地点", "teacherId": "教师 ID", "isActive": "是否有效（新增或修改为 true，取消为 false）" },... ] }。上课打卡系统根据 isActive 字段对本地课程数据进行相应的新增、更新或删除操作，确保数据的准确性和及时性，避免因课程信息不一致导致的打卡混乱问题，提高教学管理的协同性和效率。
  + ****学生信息同步接口****：定期同步学生的基本信息（如学号、姓名、专业、班级等）和学籍状态（入学、休学、退学等）变化情况，确保上课打卡系统能够准确识别学生身份并进行相应的权限管理和考勤记录。同步接口同样采用定时任务方式，每周执行一次，接口路径为 /sync/studentData，请求方法为 GET，响应数据为 { "students": [ { "studentId": "学号", "studentName": "学生姓名", "major": "专业", "class": "班级", "status": "学籍状态" },... ] }。系统根据同步的学生信息更新本地数据库，保证学生信息的完整性和准确性，为精准的考勤管理提供数据支持，同时也有助于学校对学生的全面管理，提升教育管理的信息化水平。

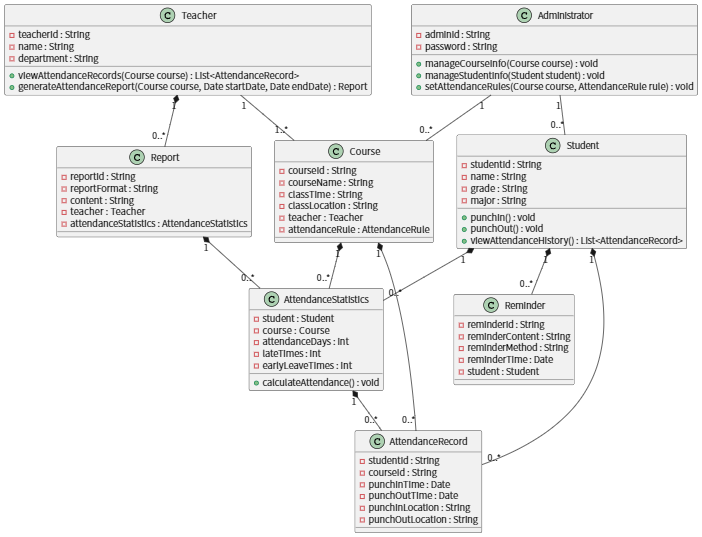
### **内部模块接口设计**

* ****打卡模块与用户管理模块接口****：
  + ****用户身份验证接口****：当学生或教师进行打卡操作时，打卡模块调用用户管理模块的身份验证接口，验证用户登录状态的合法性，防止非法打卡行为。接口接收用户输入的账号和密码（或其他身份标识信息），返回验证结果（成功或失败）及用户相关信息（如用户类型、权限级别、所属班级或课程等）。接口定义在用户管理模块中，打卡模块通过内部服务调用方式访问，例如在 Java 项目中可以使用 Spring Cloud 的 Feign 客户端进行服务间的远程调用，接口方法签名为 boolean authenticateUser(String username, String password)，返回 true 表示验证成功，false 表示验证失败。
  + ****用户信息获取接口****：打卡模块在打卡过程中需要获取用户的详细信息，如学生的班级信息用于考勤统计和报表生成，教师的授课课程信息用于确认打卡权限等。用户管理模块提供用户信息获取接口，根据用户 ID 返回用户的完整信息数据结构。接口方法签名为 User getUserById(String userId)，其中 User 为包含用户各种属性的实体类，如 { "userId": "用户唯一标识", "userName": "用户名", "userType": "用户类型（学生/教师）", "relatedData": { "studentClass": "学生班级", "teacherCourses": [ "课程 1 ID", "课程 2 ID",... ] } }。这样的接口设计实现了模块之间的低耦合和高内聚，提高了系统的可维护性和扩展性，使得打卡功能能够准确、高效地依赖用户信息进行业务逻辑处理，保障整个上课打卡系统的稳定运行。
* ****打卡模块与课程管理模块接口****：

## 活动图



类图



## 时序图

## 用例图

## 

## 状态图

## 

## 登录界面

## 

## 打卡界面

## 