福州大学

“微签”需求分析报告

选　　题：微签

组　　别：第四组

组　　名：从零开始

成　　员：杨泽隆、房蕴锦、许雅晶、方浩宇、刘洋、丘英杰、刘心怡、李臻、王宇航

指导教师：程永利

2021 年 10 月 17 日

目录

1. 导言

1.1目的

本报告目的在于明确说明“微签”各功能的实现方式，指导团队进行编码，并解决实现该系统的程序模块设计问题。保证软件开发的质量、需求的完整与可追溯性，在需求方面让业务需求提出者与需求分析人员、开发人员、测试人员及其其他相关人员达成共识，同时也作为软件总体测试的依据。

1.2文档约定

本文档按以下要求和约定进行书写：

1、标题最多分三级，分别为黑体小二、黑体三号、黑体小三，标题均加粗。

2、正文字体为宋体小四，行距 20 磅。

3、需求优先级说明：每个需求陈述都有其自己的优先级。

1.3 适用人群和阅读建议

1、项目经理：项目经理可以根据该文档了解预期产品的功能，并据此进行系统设计、项目管理。

2、设计员：对需求进行分析，并设计出系统，包括数据库的设计。

3、程序员：配合《设计报告》，了解系统功能，编写《用户手册》。

4、测试员：根据本文档编写测试用例，并对软件产品进行功能性测试和非功能性测试。

5、用户：了解预期产品的功能和性能，并与分析人员一起对整个需求进行讨论和协商。

在阅读本文档时，首先要了解产品的功能概貌，然后可以根据自身的需要对每一功能进行适当的了解。

1.4 项目范围

“微签”适用于帮助老师打造一个智能精准的课堂签到环境，利用各种反作弊功能精准确定学生的出勤情况。

2. 总体描述

2.1产品愿景

微签是一款轻量化的签到软件，体积小、效率高、易上手。 微签将成为教师课堂的好帮手，培养学生学习习惯的好伙伴。

2.2产品特性

微签是一款体积小、效率高、易上手的软件。同时具有人脸识别、定位、签到码实时更新等多种反作弊手段。

对教师而言，能够利用微签打造一个智能精准的课堂签到环境，营造良好课堂出勤氛围。对学生而言，能够防止学生作息紊乱、避免迟到旷课现象，帮助学生养成良好的作息，正常出勤。

2.3用户类型和特征

用户包括教师和学生两种类型。

教师具有以下特征：由于传统的课堂签到模式占用课堂教学时间，影响教学任务，同时不可避免的出现代签代课现象，使得课堂出勤存在虚假性。教师一直希望能有一种工具来解决上述问题。

学生具有以下特征：部分学生可能出于某些原因，企图旷课，或者代签代课，“微签”可以很大程度上避免代签代课，同时可以精准的找到未出勤学生，让学生按时出勤，认真上课。

2.4操作环境

前端环境	微信小程序
后端环境	Ubuntu 18
	Python 3

2.5设计和实施约束

时间约束	开发时间约为 6 周
人员约束	前端 4 人，后端 4 人
技术约束	适用微信开发者工具和 python 语言进行开发

2.6用户文档

将与软件产品一同交付的用户文档如下

用户手册	电子版
常见问题解答	电子版

2.7假设和依赖

本项目是否能够成功实施，主要依赖于以下的假设：

（1）团队成员的积极合作配合，为了项目的开发和实施，对个人时间进行合理规划同时为团队做出合理牺牲，配合队友完成任务。

（2）团队成员积极学习，努力掌握先进的能够适用于该项目的技术，这是系统的性能是否优化和项目能否成功的保证。

（3）团队项目的设计是否合理，这是实现难易程度的根本。

（4）假设项目需求在确定后不会有较大的改动。

（5）假设成品能极大程度地还原原型设计。

3. 外部接口需求

3.1用户接口

3.1.1 学生

我的课程：可以查看课程信息。

查看课程信息：可以查看当前课程的历史签到、退出课程、如果当前课程已开启签到，则可以由此进入签到界面。

加入课程：通过输入班级码或扫码进行签到时，需要先加入课程。

扫一扫：扫码进行签到。

个人中心：编辑个人资料

签到： 点击签到按钮，进行签到，通过反作弊验证后完成签到。

以下是各个用例的文字描述：

用例 1：登录。

主参与者：教师（点名者）、学生（被点名者）。

情境目标：通过微信账号登录小程序，实现账号绑定，并且填写个人资料。

前提条件：参与者的设备必须能够联网；参与者拥有微信账号。

触发器：进入小程序。

场景：

1. 跳出微信授权登录窗口。
2. 用户授权同意登录。
3. 登录成功。

异常处理：

1. 用户拒绝授权登录——提示“登录失败”，重新登录

优先级：必须实现

用例 2：填写或更改个人信息。

主参与者：学生（被点名者）。

情境目标：通过微信账号登录小程序，填写或更改个人资料。

前提条件：参与者的设备必须能够联网；参与者拥有微信账号；参与者已登录。

触发器：用户手动点击；课程签到时发现没有填写个人信息。

场景：

1. 用户填写信息

异常处理：

1. 没有登录——参看用例“登录”。

优先级：必须实现

用例 3：学生加入课程。

主参与者：学生（被点名者）。

情境目标：通过老师发布的课程信息加入课程。

前提条件：参与者的设备必须能够联网；参与者拥有微信账号；学生已经登录小程序；学生已经填写了个人资料。

触发器：老师发布课程信息。

场景：

1. 学生进入微签小程序。
2. 通过微信账号登录小程序。
3. 通过扫描二维码、输入教师发布的课程码进入课程详情页。
4. 学生加入课程。
5. 小程序提示：加入成功。

异常处理：

1. 没有微信账号——微信账号注册。

2. 没有登录——参看用例“登录”。
3. 没有填写个人资料——参看用例“填写个人信息”

优先级：必须实现

次要参与者：教师

次要参与者的使用方式：

1. 教师：发布签到

用例 4：学生查看已加入课程。

主参与者：学生（被点名者）。

情境目标：查看已加入课程。

前提条件：学生已经登录。

触发器：点击“我的课程”。

场景：

1. 学生点击“我的课程”。
2. 显示已加入课程。

异常处理：

1. 没有登录——参看用例“登录”。

优先级：实现基础功能之后实现

用例 5：使用微签小程序进行签到。

主参与者：学生（被点名者）。

情境目标：通过老师发布的课程信息进行签到。

前提条件：参与者的设备必须能够联网；参与者拥有微信账号；学生已经加入了课程。

触发器：老师发布的课程信息让学生签到。

场景：

1. 学生进入微签小程序。
2. 通过微信账号登录小程序。
3. 通过扫描二维码、输入教师发布的课程码或者查看我的课程进入签到界面。
4. 成功通过反作弊验证。
5. 小程序提示：签到成功。

异常处理：

1. 没有网络——参看用例“签到失败处理“。
2. 没有微信账号——微信账号注册。

3. 学生没有登录——参看用例“登录”。
4. 学生没有加入课程——参看用例“学生加入课程”。
5. 没有通过反作弊验证——参看用例“签到失败处理”。

优先级：必须实现

次要参与者：教师

次要参与者的使用方式：

1. 教师：发布签到

用例 6：学生查看签到记录。

主参与者：学生（被点名者）。

情境目标：查看签到记录。

前提条件：学生已经登录；学生已加入课程。

触发器：点击已加入的课程。

场景：

1. 学生点击已加入的课程。
2. 显示签到记录。

异常处理：

1. 没有登录——参看用例“登录”。
2. 没有加入课程——参看用例“学生加入课程”。

优先级：实现基础功能之后实现

3.1.2 教师

如图 1 和图 5 所示。包括以下功能界面。

创建课程：新建课程。

查看课程信息：查看已创建的课程。包括历史签到统计和创建签到。

创建签到：如图 6 所示，设置好签到后，发布签到。签到界面还包括签到统计、补签（如图 7 所示）。

以下是各个用例的文字描述：

用例 7：教师创建课堂

主参与者：教师（点名者）。

情境目标：教师创建新课堂。

前提条件：教师已经登录。

触发器：用户手动点击。

场景：

1. 用户填写信息。

异常处理：

1. 没有登录——参看用例“登录”。

优先级：必须实现

用例 8：教师查看我的课堂

主参与者：教师（点名者）。

情境目标：教师查看我创建的课堂。

前提条件：教师已经登录。

触发器：用户手动点击。

场景：

1. 用户点击“我的课堂”。
2. 显示已创建课程。

异常处理：

1. 没有登录——参看用例“登录”。

优先级：必须实现

用例 9：教师查看历史课堂

主参与者：教师（点名者）。

情境目标：教师查看课堂历史记录。

前提条件：教师已经登录。

触发器：用户手动点击。

场景：

1. 用户点击课堂名称。
2. 显示课堂历史签到记录。

异常处理：

1. 没有登录——参看用例“登录”。

优先级：必须实现

用例 10：教师查看签到统计

主参与者：教师（点名者）。

情境目标：教师查看已签到和未签到名单。

前提条件：教师已经登录，课堂已创建。

触发器：用户手动点击。

场景：

1. 用户点击课堂时间。

优先级：必须实现

3.1.3 需求分析模型

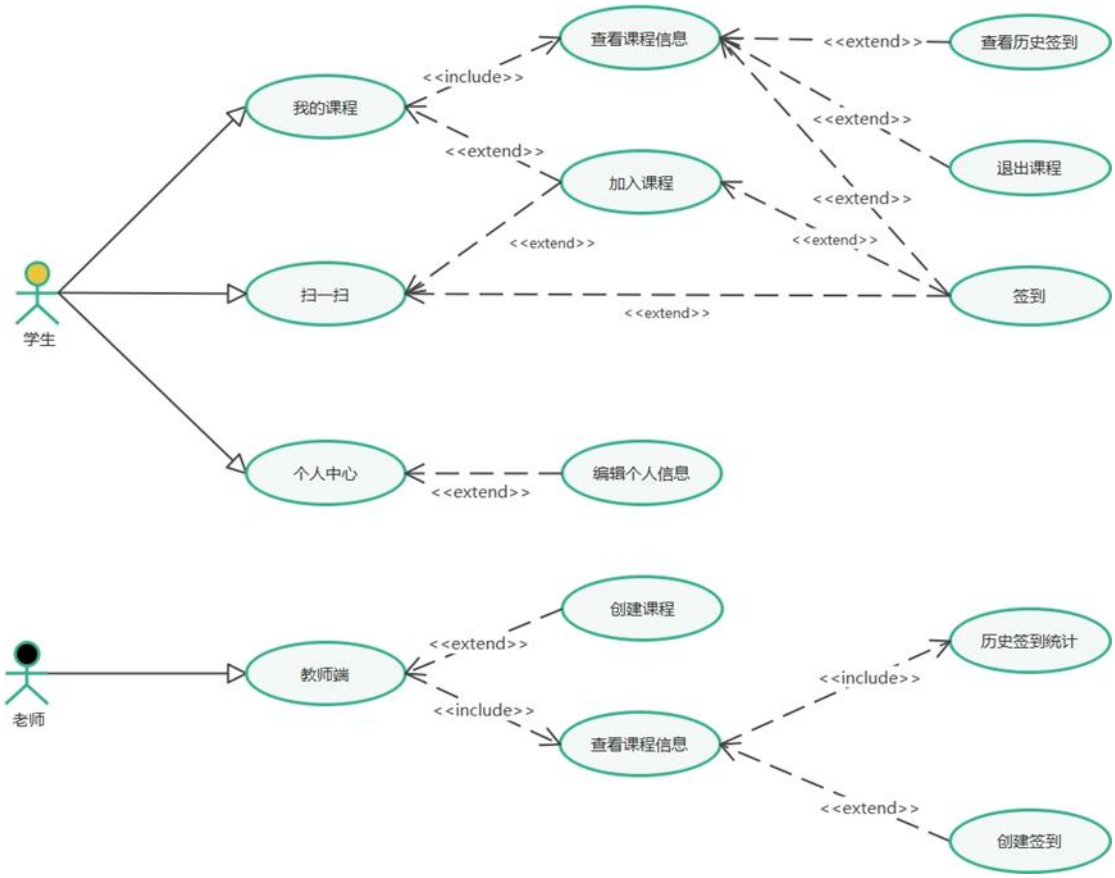


图 1. “微签”总用例图

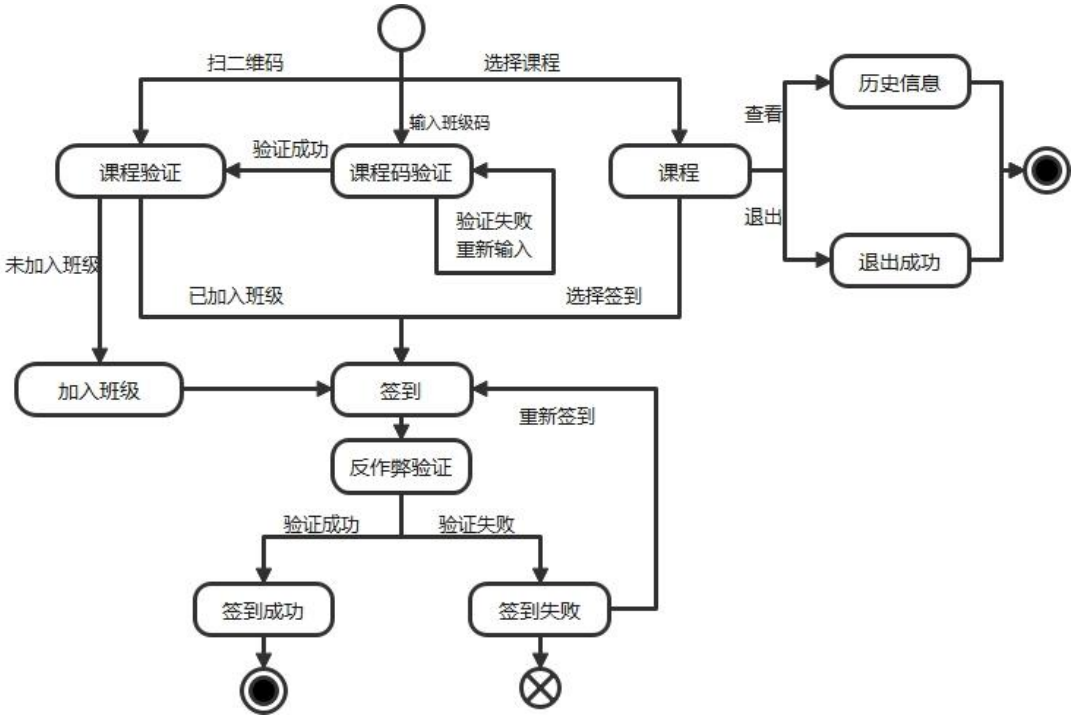


图 2. 学生端总状态图

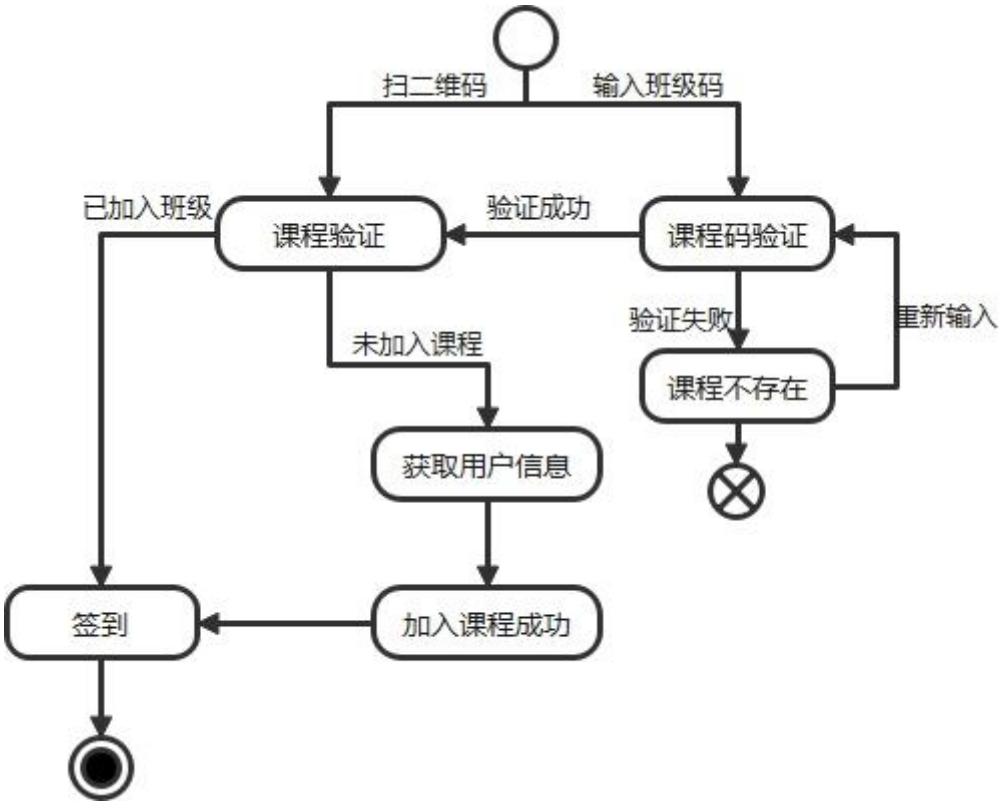


图 3. 学生加入课程状态图

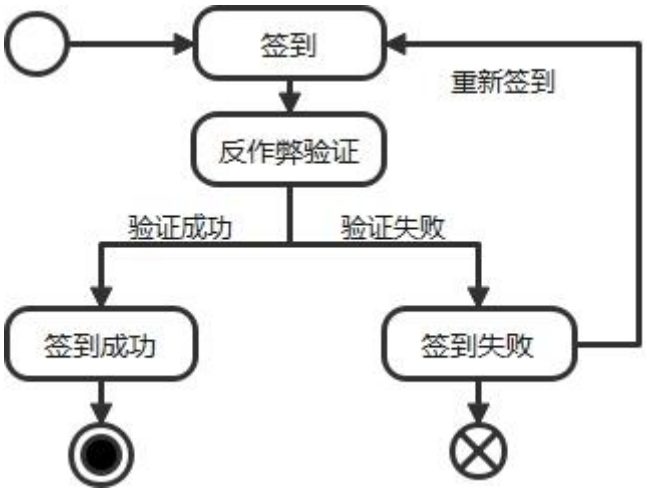


图 4. 学生签到状态图

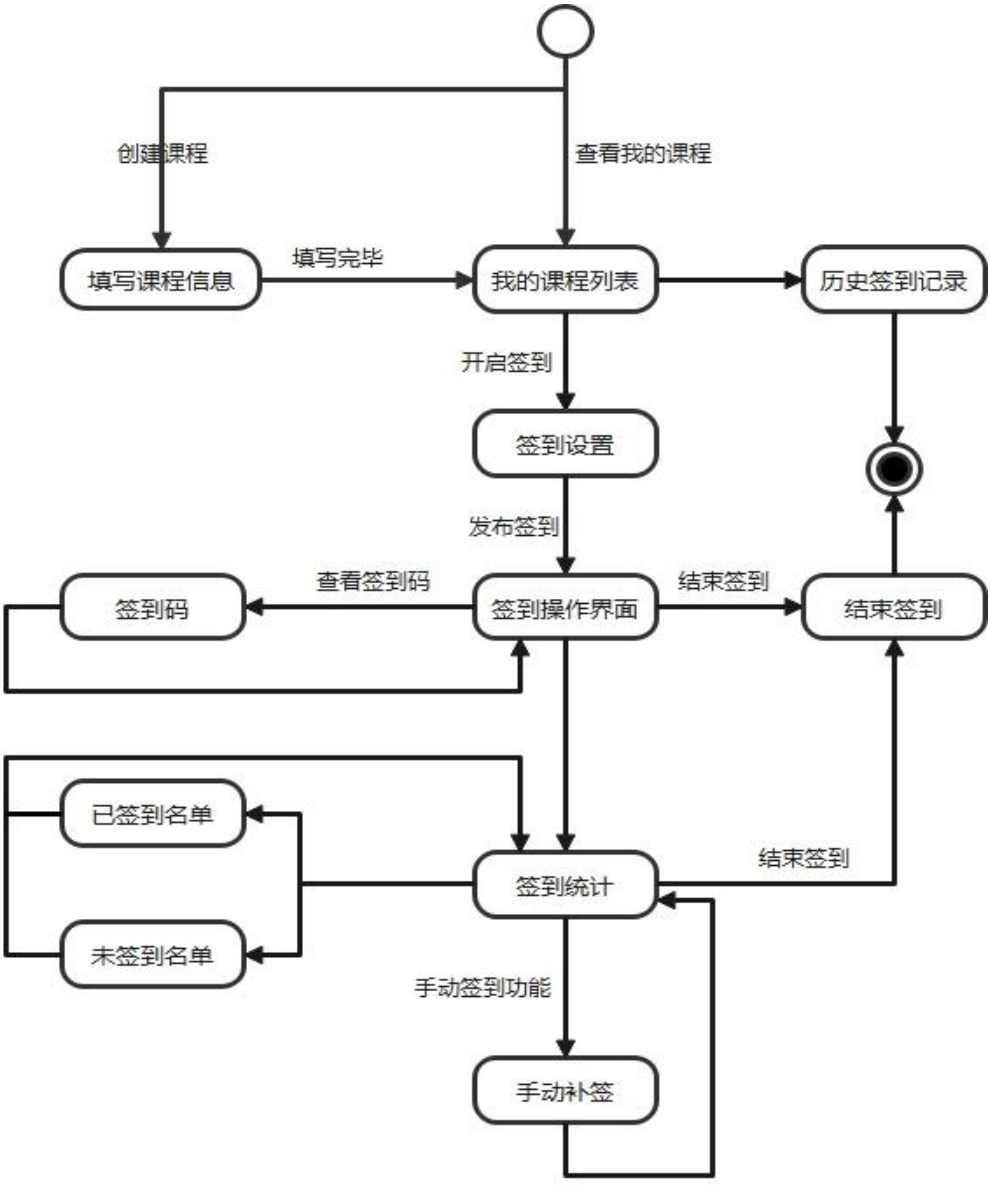

创建课程
查看我的课程
填写课程信息
填写完毕
我的课程列表
历史签到记录
开启签到
签到设置
发布签到
查看签到码
签到码
签到操作界面
结束签到
结束签到
已签到名单
未签到名单
签到统计
结束签到
手动签到功能
手动补签

图 5. 教师端总状态图

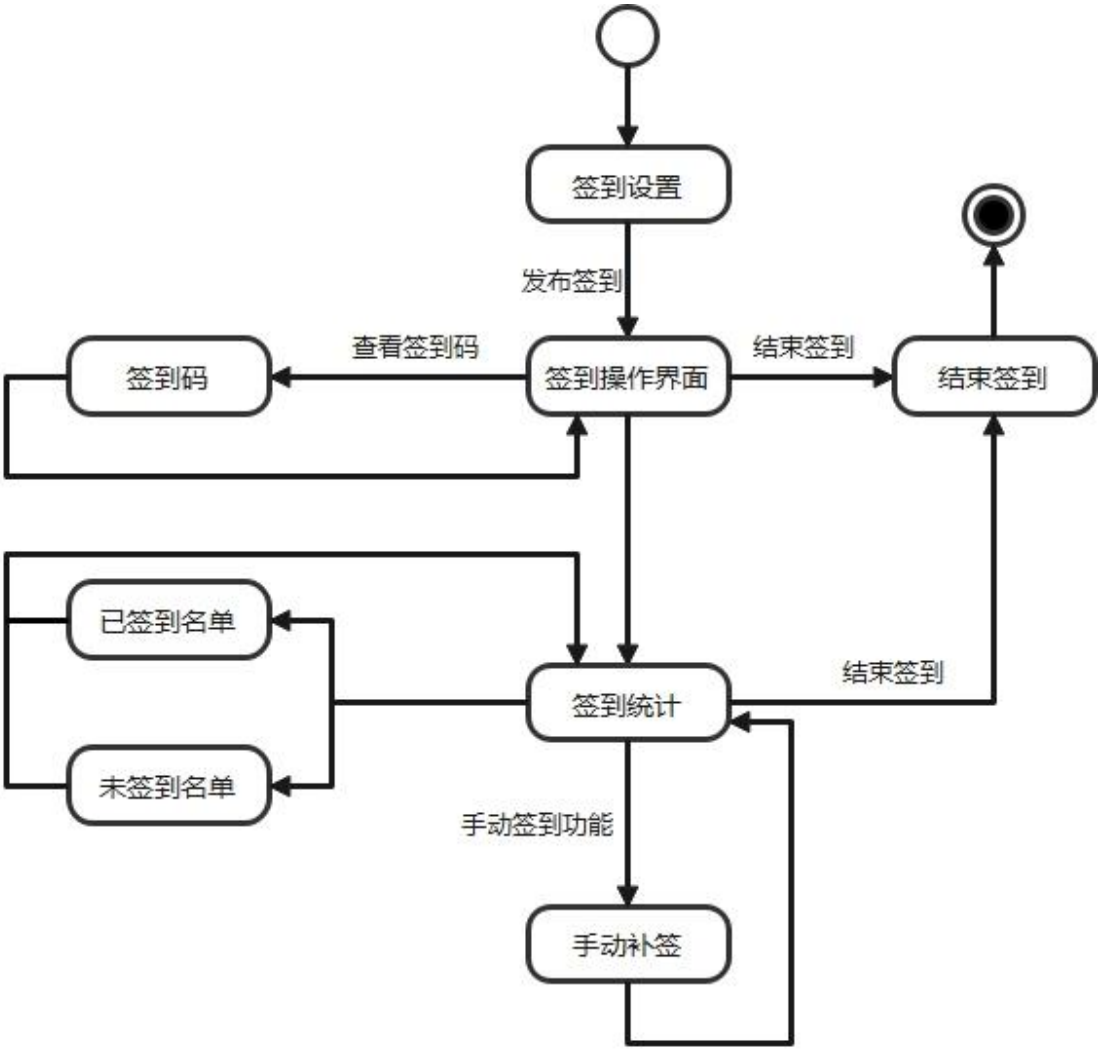


图 6. 教师端发布签到状态图

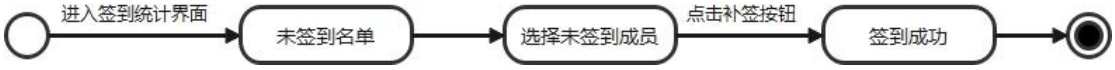


图 7. 教师端发布补签状态图

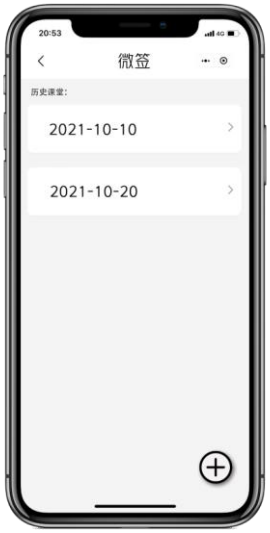
3.1.4 原型展示


微签
我是老师
我是学生


微签
课堂信息:
课程名
教师姓名


微签
已签到(2)
未签到(0)
杨泽隆
刘洋
结束签到

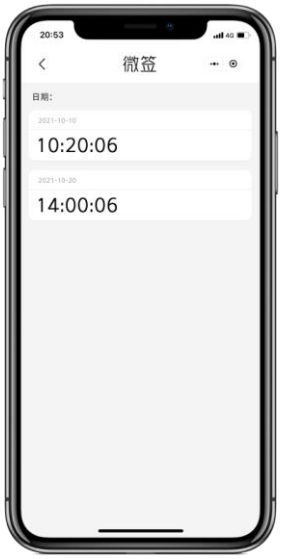

微签
我创建的课堂:
软工实践A班
软工实践B班


微签
历史课堂:
2021-10-10
2021-10-20


微签
个人信息:
学号
姓名


微签
我加入的课堂:
软工实践
数据库系统原理


微签
输入课堂码
PIN


微签
日期:
10:20:06
14:00:06

3.2 硬件接口

支持的硬件类型	手机、电脑
使用的通讯协议	HTTPS 协议

3.3 软件接口

操作系统	安卓、IOS
数据库	SQL 数据库
工具	微信

3.4 通信接口

网络通讯标准或者协议	TCP/IP 协议、HTTPS 协议
通讯安全和加密	适用 HTTPS 协议等技术，对通信信息进行加密

4. 其他非功能性需求

4.1性能需求

用户数量	10000
系统支持的并发操作数量	100
响应时间	2 秒之内

容量需求	内存	64M
	磁盘空间	128M
	数据库中表的最大行数	256

4.2 软件质量属性

4.2.1 可修改性

1、局部化变更

局部化意味着实现“模块化”思想。本软件坚持设计模式中的“单一职责原则”的设计原则。一个模块只完成一个部分，使每一个模块责任单一，防止职责过多引起整体变更时的繁琐，复杂，主要表现在类、函数、方法和接口的时候，实现“高内聚，低耦合”。

2、防止连锁反应

尽量防止修改被调用的函数或类影响到调用他的函数。包括以下措施：

（1） 信息隐藏。信息隐藏就是把某个实体地责任分解为更小地部分并选择哪些信息成为共有，哪些信息成为私有的。

（2） 维持现有的接口。该战术的模式包括添加接口、添加适配器、提供一个占位程序。

（3） 限制通信路径。限制与一个给定的模块共享的模块，减少联系，一旦变更影响会小很多。

（4） 使用仲裁者。插入仲裁者来管理依赖之间的关系，就比如数据库的使用，通过数据库来管理不同的数据信息。

4.2.2 可测试性

可测试性的目标是允许在完成软件开发的一个增量后，轻松地对软件进行测试，从而发现错误。

1、管理输入/输出

（1）记录/回放。指将捕获跨接口地信息，并将其作为测试专用软件地输入。

（2）将接口与现实分离。将接口与实现分离允许实现的代替。

（3）特化访问路线/接口。具有特化的测试接口允许通过测试工具并独立于其正常操作，来捕获或指定组件变量的值。

2、内部监视。组件可以维持状态、性能负载、容量、安全性或其他

可通过接口访问的信息。当监视状态被激活时可以记录事件。例如可以使用编译器控制台实时显示程序运行时的各种查询或输入输出结果，一遍监视。