学　　号：20125041

天 津 商 业 大 学 毕 业 设 计（论 文）

# 基于多点随机触控的异步高并发签到系统的设计与实现

# Design and Implementation of the Asynchronous High Concurrent Attendance System Based on Random Multi-Touch


学　　院:　　信息工程学院

教 学 系：　　计算机系

专业班级:　　软件工程 1201 班

学生姓名:　　王靖伟

指导教师:　　杨　亮　　讲师


**2016 年 6 月 1 日**

# 目　录

**内容摘要：**课堂考勤不仅是保证大学课堂教学质量的一个重要手段，同时也是学校掌握学生动向的一个重要信息来源。诸如课上点名和随堂测验的传统考勤方式存在耗时长、统计难、作弊容易及信息流通较慢等诸多弊端。而诸如指纹识别和人脸识别等最新手段存在成本较高且可靠性较低等尚待解决的问题。本文在对国内外现有课堂考勤类软件进行充分调研的基础上，提出了一种基于多点随机触控的异步高并发签到系统设计方案。该系统通过分析学生与教师之间的相对地理位置，并使学生在一段特定的时间内双手持续触控屏幕上随机出现的多点进行考勤验证。在技术上，本系统一方面通过使用时间戳同步及异步数据验证技术来实现系统的高并发和高可用性；另一方系统通过数据签名传输及数据库不可逆加密存储等技术保证系统的安全性。此外，在用户界面和交互体验的设计上我们遵循了 Google 最新提出的原质化设计方法。对系统进行的大量集成测试表明：系统具有极高的稳定性、可用性和易用性。本系统在实际使用中也获得了令人满意的反馈。

**关键词：** 课堂考勤；多点触控；异步验证；高并发；负载均衡

**Abstract**: Class attendance is not only an important means to ensure the teaching quality of college classes, but also a significant information source for the school to grasp the trend of the students. The traditional attendance methods such as the roll call in class and the drop quizzes have got a big time cost, difficult statistic, easy cheat and slow information communication, and many other disadvantages. Meanwhile, the latest means such as fingerprint identification and face recognition and so on, also exist the issues to be solved with a high cost and low reliability, etc. This paper has proposed a design scheme of asynchronous high concurrent sign-in system based on multi-point random touch, on the basis of the adequate investigation and research towards the existing class attendance software in China and abroad. This system performs the attendance verification, by analyzing the relative geographical position between the students and the teacher, and making the students keep touching the multi points that randomly appear on the screen with both hands in a specific period of time. In technology on one hand, the system achieves high concurrency and high availability by applying the timestamp synchronization and asynchronous data verification technologies; on the other hand, it ensures the system security through the technologies of signature data transmission and irreversible database encryption storage, etc. In addition, it follows the Material Design method newly put forward by Google in the design of user interface and interactive experience. Finally, a large number of integration tests on the system show that the system has a high stability, availability and usability. It has also gained satisfactory feedbacks in practical application.

**Key Words**: class attendance multi-touch asynchronous verification high concurrency load balancing

# 1 导言

## 1.1 研究背景及意义

课堂考勤是大学生课堂管理的一部分，传统的课堂考勤存在耗时长、难统计、浪费教育资源等诸多弊端。随着计算机及其信息技术的快速发展，人们对信息处理的速度要求也越来越高。

“智慧校园”的提出，加快了教学、科研、管理和校园生活的充分融合。“智慧校园”的发展和创建是当今高校完善自身教育体制，落实信息化教育，提高自身竞争力的重要途径，是未来教育的发展方向。“智慧校园”的建立使学校、社会、科技三者直观地连接在一起，在理论与实践之间架起桥梁，方便师生利用共享的校园资源，完成一体化、智能化的目标，为教育信息化发展起到重要的作用。

## 1.2 国内外研究综述

一些附带硬件的签到方式起到了举足轻重的作用，也很好解决的了这一问题，但是随之而来的是另外一些问题，硬件难配置、难部署、价格贵等问题，例如指纹机、人脸识别机、虹膜识别机、打卡机、NFC 近场通讯磁贴等[1-4]。即使不考虑硬件的部署和价格问题，在学生操作的时候和统计的时候也会比较耗时并且结果很难导出。有一些使用传统台式机的桌面环境部署的课堂签到平台，具有易开发、易部署、使用方便等特点,但是只适用于机房和多媒体教室等环境，不适用于普通的课堂环境[5-8]。还有一些基于 OpenCV 的方法用来进行课堂签到，其算法涵盖神经网络、机器学习、人工智能等复杂算法，其识别度不高也是现阶段计算机视觉的一大缺陷。使用手机扫二维码方式签到也存在代替签到、二维码容易破解、学生不在教室内签到等弊端。

现阶段的课堂考勤类的工具的研究及开发对于课堂考勤并没有这一事件有全局性的把握，一些学生会犀利其中的某些漏洞，从而仍然不按时到课堂甚至不到课堂。例如手持设备互借、代签、微信短信等方式传递考勤签到验证必要信息（二维码、验

证码等）、非教室内签到等问题。

## 1.3 研究内容

本文主要研究解决的问题为如何使用移动手持设备解决大学生课堂签到，并且可以有效防止代签等风险。

包括设计一套完善的课堂签到逻辑及相关策略、设计并实现一个基于 Android 平台的客户端 Native 应用、设计并实现能与 Android 客户端互通的服务器端程序、服务器可以承受签到上报时的高并发负载压力以及服务器相关策略的设计。

涉及到的难点有信息安全相关问题、时间同步机制、随机多点触控相关问题、高并发及负载均衡相关问题、异步验证相关问题及地理围栏等问题。

## 1.4 本论文的章节安排

本文共分为六个章节，各章节内容和组织结构安排如下：

第 1 章：介绍了本文的研究背景及意义，探究了国内外对于大学生课堂签到的形式，制定本文的研究目标及方案，确定了本文的总体结构安排；

第 2 章：对本系统进行了系统分析，包括了详细的需求分析及可行性研究，介绍了本系统开发时用到的主要工具；

第 3 章：对本系统进行系统设计，详细设计了其功能结构，对本系统的主要业务流程进行了详细的设计，并且设计了系统整体架构和数据库结构；

第 4 章：详细了描述本系统从注册登录到功能菜单，再到教师端、学生端、管理员端，最后到个人中心和关于的实现过程，完整的展示了本系统的实现效果，展示了系统运行时，学生端、教师端、管理员端的各个功能及操作流程。

第 5 章：深入研究了本系统在开发过程中使用到的关键技术、遇到的难点，重点研究了信息安全体系、时间戳同步机制、随机多点触控、高并发及负载均衡、异步验证、地理围栏、Google Material Design、国际化与本地化等问题，并提出了与之相对应的解决方案；

第 6 章：提出了本系统的特点和创新点，对本文进行总结并提出进一步的研究方向。

# 2 系统分析

## 2.1 需求分析

### 2.1.1 系统的目的和范围

随着高校的日益扩招，每门课程选课的人数也在不断上升，从最初的几十人迅速增加到几百人。据不完全统计，2015 年高等院校在校学生人数已达 2804 万人，这就给平时课堂考勤带来了一定的困难。一方面课上点名过于浪费时间，另一方面随堂测验的方式给教师课下统计带来很大的麻烦。因此，设计一种更高效、更快捷的签到方式成为时下各高校师生最急迫的需求。

### 2.1.2 业务环境

本系统致力于保证普通高等院校在校大学生课堂出勤率、方便教师统计与管理签到数据，使课堂签到的管理更加系统、规范与智能。

### 2.1.3 约束与假定

在正式确定需求之前，对本系统的开发进行了一些约束性的条件，这些约束是根据现有的开发资源或要求提出的。包括对编程语言的约束、对开发工具的约束、对系统性能的约束、对代码体积的约束、对功能设计的约束以及对于环境部署的约束等。

（1）编程语言的约束

本系统主要针对 Android 手持设备进行开发，主要使用到的编程语言为 Java；在服务端使用 PHP 与 Node.js，允许使用 PHP 语言和 JavaScript 语言作为服务端语言进行开发；如在开发过程中遇到必须使用的语言种类，例如 Groovy 或 Bash，则按实际情况使用即可。

（2）开发工具的约束

在本系统开发过程中，严格使用 Android Studio、PhpStorm、WebStorm、MAMP、Paw、Charles 进行开发，不允许使用 Eclipse 等其他 IDE 进行开发。

（3）性能约束

系统响应时间：对于管理员、教师、学生在信息查询方面在 0.1s 内响应，对于上报数据的结算，采用异步策略，结算时间在可以接受范围内即可，初步确定为 30 分钟，但实际可以缩短此时间间隔。

高并发：对于签到上报数据时，服务器应该能够进行负载均衡以及可以承担相应的高并发量，单个服务器可承受最大并发量达到每秒 2000 次以上。并且避免在大量用户同时上报数据时，网络阻塞甚至服务器宕机。

（4）代码体积约束

为了便于用户在 Android 手持设备上下载安装此系统，要求 APK 正式打包后大小不超过 20MB。

（5）功能设计约束

主要业务流程（学生签到）必须使用多点随机触控；上报数据后必须异步验证其结果。

（6）环境部署约束

截至 2016 年 5 月 2 日，装载 Android 4.0 或更高版本的移动手持设备达到 97.7%[9]。为了适配大部分 Android 设备，客户端部署环境确定为 Android 4.0 或更高版本。

服务端架设在阿里云 ECS 云服务器上，环境配置要求如下：Ubuntu 14.04.4 LTS amd64、Nginx 1.8.0、PHP 5.6.9、MariaDB 5.5.42、Node.js 5.10.1、Memcached 1.4.25。

### 2.1.4 风险及解决措施

在开发本系统或正式上线后不免会出现一些风险性问题。在此，对此类可以预见的风险进行预估，并且做出一些相应的解决措施，以保证系统的正常开发及运行。

（1）系统配置

由于系统配置问题，可能导致服务器运行不稳定或运行结果与测试环境不匹配。

解决措施：严格按照 2.1.3（6）中环境部署版本号进行部署，如需升级必须经过

严格的测试。

（2）数据完整性

由于不可预知的原因，服务器可能宕机，Memcached[10]中的数据完全丢失。

解决措施：严格服务器日志的写入，Memcached 宕机后，可由日志进行数据追溯，确保数据完整性。

（3）数据安全

上报数据篡改或个人信息泄露。

解决措施：定期更换 RSA 密钥对，防止恶意破解。

2.1.5　功能性需求

根据 2.1.1-2.1.4 小节，对本系统的功能性需求进行确定如下：

（1）基础需求

注册：允许学生自主注册账户，注册后与本机绑定并直接登录。

登录：允许管理员、教师、学生通过自己的账号进行登录。首次登录后立即与手机进行永久绑定，不可登出，再次登录时可自动登录。

忘记密码：当用户忘记密码时，可以根据自己的相应信息找回密码。

查看并修改个人信息：用户可以查看、上传或修改头像、电话号码、性别；查看姓名、编号。

修改密码：用户可在知道当前密码的情况下修改密码。

关于：用户可以在此看到本应用的信息和版本号。

（2）角色：学生

学生可以定位、签到、查看已签到的课程及其详情、请假（不在首个版本中实现）

（3）角色：教师

教师可以定位、发起签到、添加课程、查看课程、修改课程状态、修改课程信息、查看课程中的学生详细签到情况、查看请假条（不在首个版本中实现）

（4）角色：管理员

管机员可以注册教师、查看教师列表、查看所有课程、查看课程中的学生详细签到情况。

2.1.6　非功能性需求

根据 2.1.1-2.1.4 小节，对本系统的非功能性需求进行确定如下：

（1）安全性需求

与服务器交互数据时，端上使用 SHA1WithRSA 算法进行签名，服务器进行验证签名，验签通过后方可继续交互数据，防止恶意上报数据或攻击。密码使用 MD5 算法加密储存，防止在网络传输中或者服务器被托库后暴露用户信息。要求确保用户及服务器数据安全性达到 99.99%，避免请求伪造的情况发生。

（2）易用性需求

考虑整体使用者情况，要求该系统操作信息条目清晰、易于学习、简单易用。

（3）响应速度需求

要求系统在 0.1s 时间内响应查询服务的请求。

（4）高并发需求

要求服务器在可以接受高并发数据量，单服务器可承受最大并发量达到每秒 2000 次，后期可逐步提高。

（5）可靠性需求

要求系统失败发生率控制在万分之五以内。

（6）可用性需求

保证本系统在生产环境运行时，宕机时间不超过总运行时间的千分之一。

（7）UI 设计需求

管理员、教师、学生通过各自的学号和工号登录并操作，每个角色自动匹配相应界面。界面简单、大气、美观，并严格按照 Google Material Design[9]进行设计。

（7）领域需求

使用者完全遵循所在地区的各项管理规定及信息安全法规范。

## 2.2 可行性研究

可行性研究是对本系统进行全面的、预见的、公正的、可靠的、科学的分析。通过这些分析，论证此项目是否可行。下面从投资必要性、技术可行性、社会可行性和风险控制可行性四个角度分别论证本系统的可行性。

### 2.2.1 投资必要性

现在市场上类似的签到产品有指纹识别、人脸识别等。但由于这些识别技术还不够成熟，识别精准度不高，无法完全满足用户需求，目前还不适合大范围推广。相比之下，本签到系统最大的优势是成本低、精准度高、适应能力强，各个方面都能给使用者带来良好的体验。

### 2.2.2 技术可行性

#### 2.2.2.1 技术可行性分析指标

（1）在当前的限制条件下，该系统的功能目标能否达到；

（2）利用现有的技术，该系统的功能能否实现；

（3）规定的期限内，本系统的开发能否完成。

#### 2.2.2.2 技术可行性分析

（1）本项目主要的技术难点在于解决高并发问题和手持设备的多点触控监听，使用 Node.js 和 Memcached 方案将大量并发数据存入 Memcached，再由 PHP 定期将 Memcached 中的数据固化到 MariaDB 中，解决了高并发这一问题。由于现在 Android 手持设备全部使用电容屏，理论上支持 10 点，问题只在于如何解决监听的逻辑；

（2）安全性：由于所有上行数据全部使用签名、加密或者策略，确保了数据的安全性，防止伪造数据或请求；

（3）可靠性：要考虑通信的可靠性和服务器的可靠性，在部署环境下增加带宽

并利用相应容错、容灾策略即可解决；

（4）在规定期限内，本系统的开发可以完成。

2.2.2.3　使用到的主要技术

（1）Android：用于开发 Android 手机客户端；

（2）PHP：用于开发 REST 接口、与数据库交互；

（3）Node.js：用于接收高并发数据；

（4）MariaDB：MySQL 的替代品；

（5）Memcached：用于暂存由 Node.js 接收的高并发数据。

2.2.3　社会可行性

随着互联网的飞速发展，打造数字化校园已经被列入现代校园建设的主体部分。通过开展一系列数字化校园项目，在原有的校园管理系统上不断增加功能和服务，完善和丰富原有系统的架构，使其更好地为学校、为师生服务。本系统满足了师生签到便捷、高效的刚性需求，将能够为打造数字化校园、提升教育信息化生产起到非常重要的作用。

2.2.4　风险控制可行性

根据 2.1.4 小节提到的风险以及所给出的解决方案，符合对本系统的风险进行有效的规避，为本项目全程的风险管理提供了有效的依据。

## 2.3　开发工具

根据 2.1.3（2）小节对开发工具进行的约束，严格使用 Android Studio、PhpStorm、WebStorm、MAMP、Paw、Charles 进行开发，下面将分别介绍这五个开发工具。

### 2.3.1　Android Studio

Android Studio 是一个 Android 开发环境，基于 IntelliJ IDEA。类似 Eclipse ADT，Android Studio 提供了集成的 Android 开发工具用于开发和调试。在 IDEA 的基础上，Android Studio 提供[11-14]：

（1）基于 Gradle 的构建支持；

（2）Android 专属的重构和快速修复；

（3）提示工具以捕获性能、可用性、版本兼容性等问题；

（4）支持 ProGuard 和应用签名；

（5）基于模板的向导来生成常用的 Android 应用设计和组件；

（6）功能强大的布局编辑器，可以让你拖拉 UI 控件并进行效果预览。

在本系统开发过程中，使用 Android Studio 2.1.1 版本来开发 Android Native 应用，使用到的 Android 支持库有 support-v4、appcompat-v7、design、recyclerview、cardview；第三方类库有 gson 2.6.2、volley 1.0.19、okhttp 3.3.1、okio 1.8.0。

### 2.3.2 PhpStorm

PhpStorm 是一个轻量级且便捷的 PHP IDE，其旨在提供用户效率，可深刻理解用户的编码，提供智能代码补全，快速导航以及即时错误检查。

在本系统开发过程中，使用 PhpStorm 2016.1.2 版本来开发 PHP REST 接口，使其可以被 Android Native 调用，访问数据接口或进行业务逻辑的执行。

### 2.3.3 WebStorm

WebStorm 是 jetbrains 公司旗下一款 JavaScript 开发工具。被广大中国 JS 开发者誉为“Web 前端开发神器”、“最强大的 HTML5 编辑器”、“最智能的 JavaScript IDE”等。与 IntelliJ IDEA 同源，继承了 IntelliJ IDEA 强大的 JS 部分的功能[15]。

在本系统开发过程中，使用 WebStorm 2016.1.2 版本来开发 Node.js 相关业务，包括接受从客户端上报的高并发数据、定时任务激活等。

### 2.3.4 MAMP

MAMP 代表苹果的 OS X 系统上的 Macintosh、Apache、MySQL 和 PHP，MAMP 内含 Apache、Nginx、PHP 以及 MySQL。可以通过可视化界面稍作设定，在苹果电脑上架设自己专属的网站，与 Windows 下的 XAMPP，Linux 下的 LAMP 一样。都是

Apache/Nginx+Mysql+PHP 的集成环境[16]。

在本系统开发过程中，作为本地测试环境来开发调试本系统。

### 2.3.5 Paw

Mac 下最先进的 HTTP 客户端。Paw 是一款功能全面且设计美观的，用来与 REST 服务轻松交互的 Mac 应用。无论是 API 构建者或是使用者，Paw 都能够帮助其构建 HTTP 请求，检查来自服务器的响应，甚至可生成客户端代码。

### 2.3.6 Charles

Charles 是一个 HTTP 代理服务器，HTTP 监视器，反转代理服务器。它允许一个开发者查看所有连接互联网的 HTTP 通信，这些包括 request、response 以及 HTTP headers（包含 cookies 与 caching 信息）。

# 3　系统设计

## 3.1　功能结构

根据 2.1.5 小节功能性需求所述，将本系统分为三个子系统：学生子系统、教师子系统、管理员子系统，每个子系统各成一端。本系统的主要功能结构图如图 1 所示。

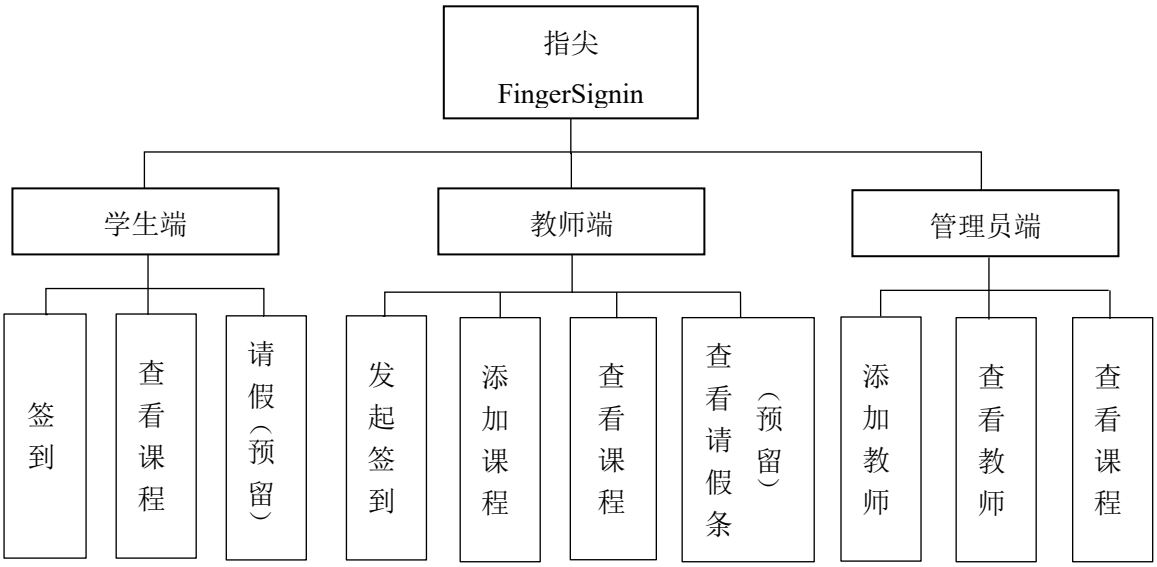


图 1　本系统的主要功能结构图

## 3.2　主要业务流程

本系统的主要业务就是签到，根据 2.1 小节需求分析，对主要业务流程进行设计。第一步，教师选择需要发起签到的课程开始发起签到，在此过程中系统会自动定位教师当前所在的地理位置的经度和纬度，教师会得到一个四位数字作为临时验证码。第二步，学生输入四位数字临时验证码并验证学生当前地理位置的经度和纬度是否在其范围内（半径 50m），如验证成功会显示该签到课程的课程名和课程信息。第三步，教师确认开始，确认后，四位数字临时验证码失效，此时学生不可再验证进入。第四部，教师通知学生已经确认开始。第五步，学生点击确认听到教师确认完毕并进入多指验证环节。第六步，所有学生同步时间戳，在教师确认后 15 秒内为准备时间，15

秒后正式记录学生的具体签到过程，此过程维持 15 秒。第七步，上报数据。以上签到业务流程如图 2 所示。

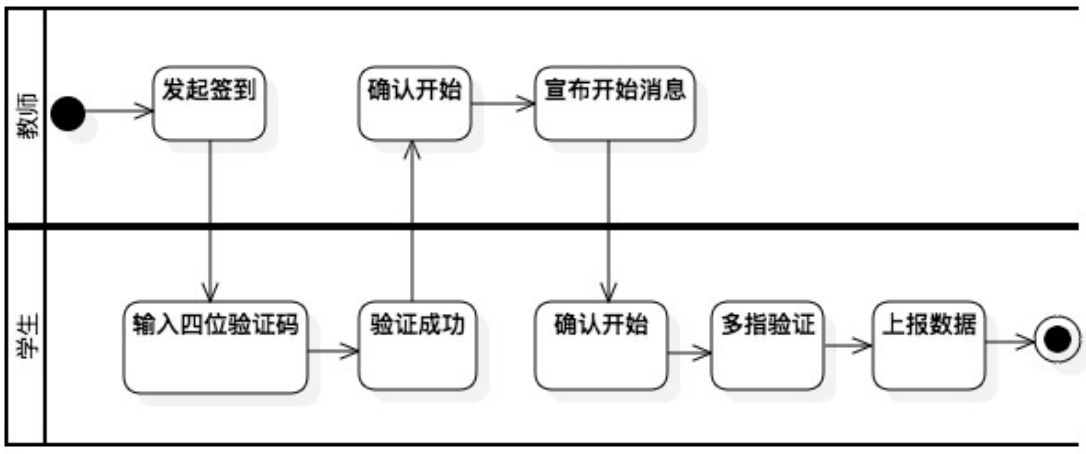


图 2　本系统的主要业务流程图（签到）

### 3.3　系统整体架构

根据本系统的特殊性，对其采用了 C/S 四层模式对其进行整体架构，四层依次为：

（1）数据存储层：使用 MariaDB 作为数据库，高速缓存服务器 Memcached 以减轻数据库的负担。MariaDB 由于其性能高于 MySQL，作为 MySQL 的替代品负责固化存储所有数据。Memcached 负责临时存储对并发量要求较高的数据，一般这些数据是由 Node.js 分发而来。

（2）数据接口层：部署在业务服务器上，负责访问数据源，提供底层的数据接口。为 PHP 提供 MariaDB 和 Memcached 的数据接口，为 Node.js 提供 Memcached 的数据接口。

（3）业务逻辑层：部署在业务服务器上，负责接收用户层客户端的指令或者数据，分发或固化数据。业务服务器主要有 Ngnix、PHP 与 Node.js，Ngnix 是一个高性能的 HTTP 服务器负责 PHP 的底层支持，PHP 主要负责本系统的各种业务逻辑，例如与数据层进行交互（固化数据至 MariaDB）、与客户层进行交互（注册、登录、签

到等业务）。Node 负责处理高并发数据的负载，以保证整体服务器的正常运转。

（4）表现层：部署在用户的手持智能终端上，负责本系统与用户的衔接与互操作。

本系统的整体架构图如图 3 所示。

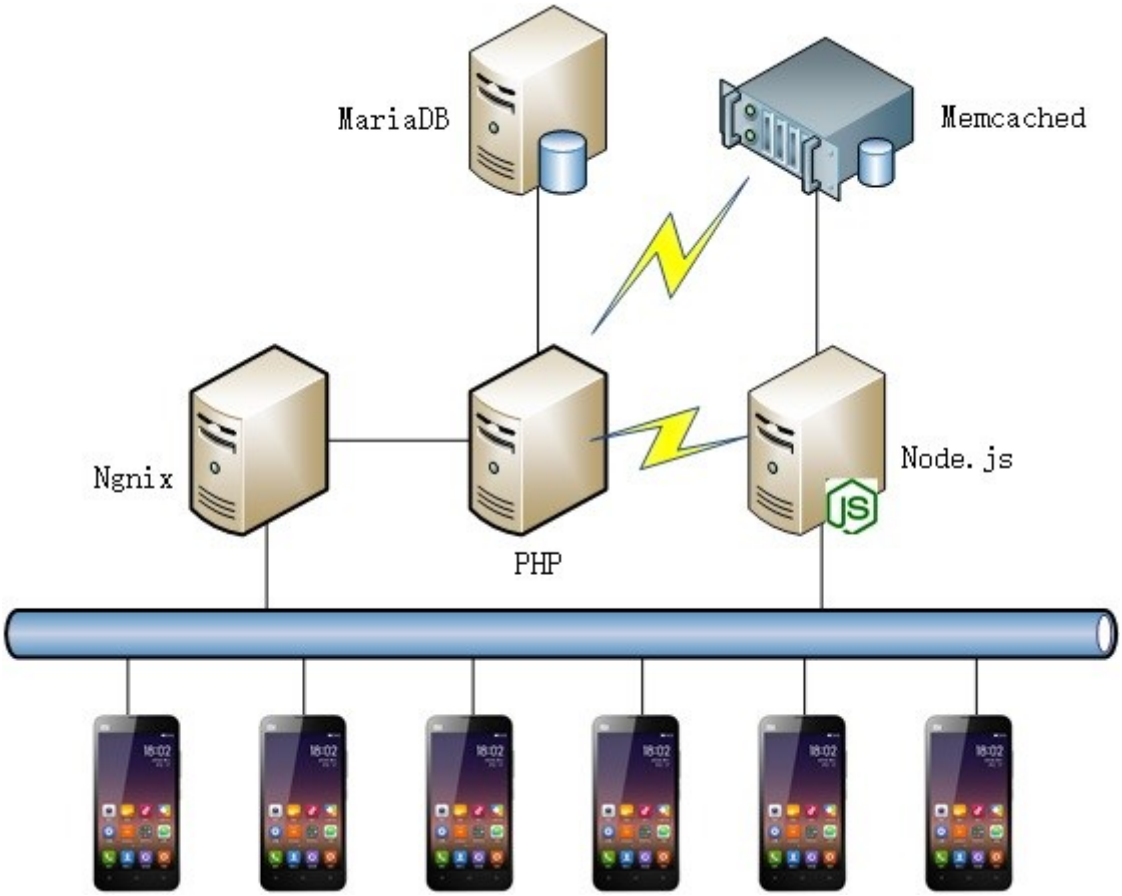


图 3　系统整体架构图

### 3.4　数据库结构

通过需求分析，可以得到实体-关系（E-R）模型图，在这里采用 Information Engineering 模型，此模型采用 Crow's Foot 表示法，也被称为“鸭掌模型”，关联关系的关联基数中采用到了一个鸭掌形的三叉线来表示。

在此系统中，一共建立了 6 张表，分别是 ACCOUNT（账户表）、IMEI（串号表）、COURSE（课程表）、FREQUENCY（课次表）、SIGNIN（签到表）、LEAVEAPPLICATON（请假表）。本系统的 E-R 模型图如图 4 所示。

在ACCOUNT表中，主键为_id；IMEI表中，主键为imei，外键为ACCOUNT__id；在COURSE表中，主键为idCOURSE，外键为ACCOUNT__id；在FREQUENCY表中，主键为COURSE_idCOURSE和frequency；在SIGNIN表中，外键为ACCOUNT__id、FREQUENCY_COURSE_idCOURSE和FREQUENCY_frequency；在LEAVEAPPLICATON表中，外键为ACCOUNT__id和COURSE_idCOURSE。

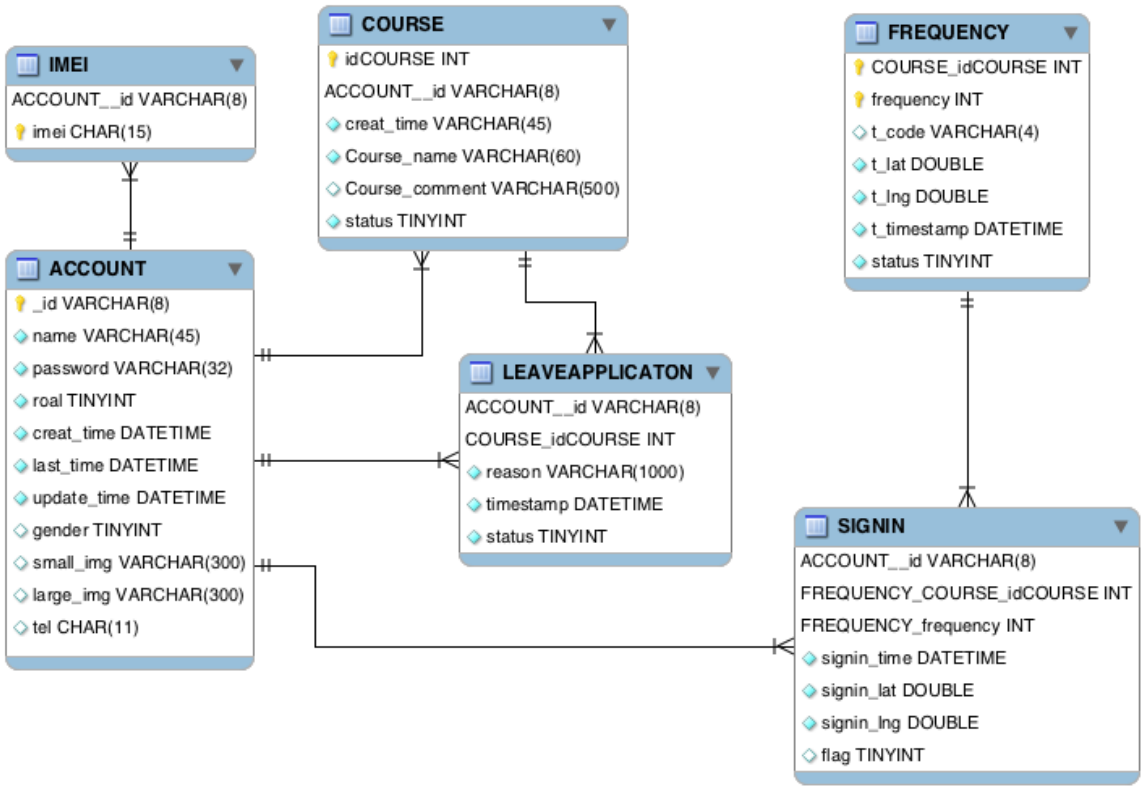


图4　本系统的E-R模型图

其关联关系为：ACCOUNT与IMEI为一对多关系，ACCOUNT与COURSE为一对多关系，ACCOUNT与SIGNIN为一对多关系，ACCOUNT与LEAVEAPPLICATON为一对多关系，LEAVEAPPLICATON与COURSE为多对一关系，SIGNIN与FREQUENCY为多对一关系；ACCOUNT与COURSE为多对多关系；ACCOUNT与FREQUENCY为多对多关系。

下面分别对6个表的结构进行详细描述。

（1）ACCOUNT 账户表：用来记录所有用户的账户详细信息，包括编号、姓名、密码、角色、账户的创建时间、上一次登录时间、修改时间、性别、大头像、小头像、电话等信息。ACCOUNT 表的结构如表 1 所示。

表 1　ACCOUNT 表的结构

| 字段 | 类型 | 空 | 默认 | 注释 |
|---|---|---|---|---|
| _id | varchar(8) | 否 | | teacher or student's number |
| name | varchar(45) | 否 | | user name |
| password | varchar(32) | 否 | | password of MD5 |
| roal | tinyint(3) | 否 | | 1=admin, 2=teacher, 3=student |
| creat_time | datetime | 否 | | create time |
| last_time | datetime | 否 | | last login time |
| update_time | datetime | 否 | | update time |
| gender | tinyint(3) | 是 | NULL | 0=female, 1=male |
| small_img | varchar(300) | 是 | NULL | small photo url |
| large_img | varchar(300) | 是 | NULL | large photo url |
| tel | char(11) | 是 | NULL | user telephone number |

（2）IMEI 串号表：用来记录用户用来登录的手机的串号，作为本系统的自动登录功能和账号绑定的基础。IMEI 表的结构如表 2 所示。

表 2　IMEI 表的结构

| 字段 | 类型 | 空 | 默认 | 注释 |
|---|---|---|---|---|
| ACCOUNT__id | varchar(8) | 否 | | |
| imei | char(15) | 否 | | |

（3）COURSE 课程表：用来记录课程号、课程名、教师编号、课程创建时间、课程信息、课程状态。COURSE 表的结构如表 3 所示。

表 3　COURSE 表的结构

| 字段 | 类型 | 空 | 默认 | 注释 |
|---|---|---|---|---|
| idCOURSE | int(11) | 否 | | course id (Transparent to users) |
| ACCOUNT__id | varchar(8) | 否 | | teacher id |
| creat_time | varchar(45) | 否 | | add course time |
| Course_name | varchar(60) | 否 | | course' name |
| Course_comment | varchar(500) | 是 | NULL | comment for course |

续表 3

| 字段 | 类型 | 空 | 默认 | 注释 |
|---|---|---|---|---|
| status | tinyint(3) | 否 | 0 | 0=init, 1=close, 2=cancel |

（4）FREQUENCY 课次表：用来记录课程号、课次号、临时唯一码、教师所在地理坐标、教师发起签到时的时间戳、签到状态。FREQUENCY 表的结构如表 4 所示。

表 4　FREQUENCY 表的结构

| 字段 | 类型 | 空 | 默认 | 注释 |
|---|---|---|---|---|
| COURSE_idCOURSE | int(11) | 否 | | |
| frequency | int(11) | 否 | | number of times |
| t_code | varchar(4) | 是 | NULL | Four random numbers |
| t_lat | double | 否 | | Registered course latitude |
| t_lng | double | 否 | | Registered course longitude |
| t_timestamp | datetime | 否 | | Registered course timestamp |
| status | tinyint(3) | 否 | | 0=init, 1=open signin, 2=close/finish |

（5）SIGNIN 签到表：用来记录学生编号、课程号、课次号、学生签到时间戳、学生签到地理坐标，其中包括经度和纬度，签到验证结果。SIGNIN 表的结构如表 5 所示。

表 5　SIGNIN 表的结构

| 字段 | 类型 | 空 | 默认 | 注释 |
|---|---|---|---|---|
| ACCOUNT__id | varchar(8) | 否 | | student id |
| FREQUENCY_COURSE_idCOURSE | int(11) | 否 | | |
| FREQUENCY_frequency | int(11) | 否 | | |
| signin_time | datetime | 否 | | student signin timestamp |
| signin_lat | double | 否 | | student signin latitude |
| signin_lng | double | 否 | | student signin longitude |
| flag | tinyint(3) | 是 | NULL | 0=fail, 1=success, 2=leaveapplication |

（6）LEAVEAPPLICATON 请假表：用来记录学生编号、课程号、请假原因、提交时间戳、教师确认的结果。此表为请假功能而预留。LEAVEAPPLICATON 表的结构如表 6 所示。

表 6　LEAVEAPPLICATON 表的结构

| 字段 | 类型 | 空 | 默认 | 注释 |
|---|---|---|---|---|
| ACCOUNT__id | varchar(8) | 否 | | |
| COURSE_idCOURSE | int(11) | 否 | | |
| reason | varchar(1000) | 否 | | student leave reason |
| timestamp | datetime | 否 | | |
| status | tinyint(3) | 否 | 0 | 0=init, 1=success & close, 2=fail & close，3=timeout & close |

# 4　系统实现

通过系统分析和系统设计，已经对本系统的认知达到一定水平，接下来进行对本系统的实现阶段。

## 4.1　注册及登录

进入本系统后，在未登录的情况下显示界面如图 5 所示，可以在此界面中选择注册或登录功能。按钮注册和登录监听用户的点击事件，如果点击后会触发相应的 Intent 事件，其核心代码如下：

```
bn_Register.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intent = new Intent(MainActivity.this, RegisterActivity.class);
        startActivityForResult(intent, 1);
    }
});
bn_login.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intent = new Intent(MainActivity.this, LoginActivity.class);
        startActivityForResult(intent, 2);
    }
});
```


图 5　未登录界面


图 6　注册界面

按钮形式遵循 Google Material Design，采用扁平式设计，字体为深蓝色（#031967），

以注册按钮为例，其布局代码如下：

```
<Button
    android:id="@+id/bn_Register"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_alignParentStart="true"
    android:background="?android:attr/selectableItemBackground"
    android:text="@string/Register"
    android:textColor="#031967" />
```

如点击注册按钮，如图 6 所示，在此界面可以通过姓名、编号、密码等信息注册学生账号。如点击登录按钮，如图 7 所示，在此界面可以通过编号、密码进行登录，如果忘记密码，可在此界面找回密码。如果尚未录入电话号码，请联系管理员修改密码，如图 8 所示。


图 7　登录界面

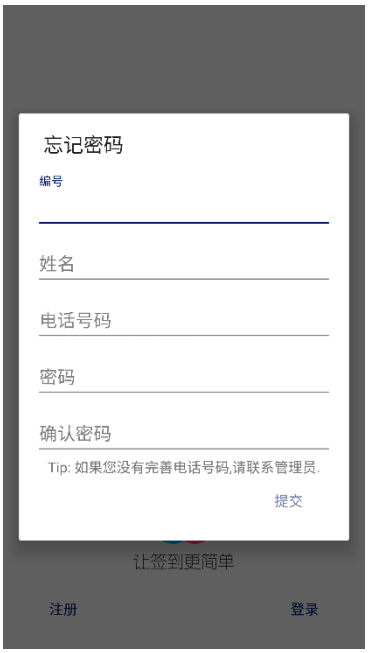


图 8　忘记密码界面

注册、登录、忘记密码三项功能的界面采用 Dialog 样式进行设计，其 styles.xml 代码如下：

```
<style name="AppTheme.popover" parent="Theme.AppCompat.Light.Dialog.MinWidth">
    <item name="colorAccent">@color/colorLogoMiddle</item>
</style>
```

为了减少用户每次打开本应用时登录所耗费的时间，也为了防止使用同一个手持设备进行代替签到，采取了手机绑定和自动登录的策略，在注册或第一次登录时候

向数据库存入手机串号，获取本机串号的核心代码如下：

```
IMEI = ((TelephonyManager) getSystemService(Context.TELEPHONY_SERVICE))
        .getDeviceId();
```

每次登录的时候会先检查此手机是否登录过某个账号，如果有相应记录则自动登录，否则可以进行登录。自动登录相关代码如下：

```
RequestQueue requestQueue = Volley.newRequestQueue(getApplicationContext());

JsonRequest<JSONObject> jsonRequest = new JsonObjectRequest(
        Request.Method.POST, Util.Final_HOST + "/account/autologin.php", json,
        new Response.Listener<JSONObject>() {
        },
        new Response.ErrorListener() {
        }) {
    @Override
    public Map<String, String> getHeaders() {
        Map<String, String> headers = new HashMap<String, String>();
        headers.put("Accept", "application/json");
        headers.put("Content-Type", "application/json; charset=UTF-8");

        return headers;
    }
};
requestQueue.add(jsonRequest);

private void toLogin() {
    Intent intent = new Intent(WelcomeActivity.this, MainActivity.class);
    ActivityOptionsCompat options =
ActivityOptionsCompat.makeSceneTransitionAnimation(WelcomeActivity.this,
findViewById(R.id.title_logo), "title_logo_transition");
    ActivityCompat.startActivity(WelcomeActivity.this, intent, options.toBundle());
}
```

## 4.2 功能菜单及快捷按钮

登录成功后，根据用户的账户自动进入到相应角色的功能界面，在 4.3-4.5 小节会具体说明教师端、学生端、管理员端的具体功能。

每个角色的具体功能都会在其功能菜单上有所体现，如图 9-图 11 所示分别展示教师端、学生端和管理员端的功能菜单界面。

功能菜单采用侧拉菜单设计，侧拉菜单分为上下两部分，第一部分显示用户的头像、姓名、电话等个人信息；第二部分为功能选项菜单，监听用户的点击事件，通过 FragmentManager 进入相应的界面。侧拉菜单的核心代码如下：

```
DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
```

```
        this, drawer, toolbar, R.string.navigation_drawer_open,
R.string.navigation_drawer_close);
    drawer.setDrawerListener(toggle);
    toggle.syncState();

    NavigationView navigationView = (NavigationView) findViewById(R.id.nav_view);
    navigationView.setNavigationItemSelectedListener(this);
```


图 9　教师端功能菜单界面


图 10　学生端功能菜单界面


图 11　管理员端功能菜单界面

如图 9-图 11 中界面右下角所示，圆形按钮为相应角色的快捷功能按钮，采用 FloatingActionButton 样式进行设计，在学生端中点击此按钮会迅速进入签到功能界面；在教师端中点击此按钮会迅速进入发起签到功能界面；在管理员端中点击此按钮会迅速进入添加教师功能界面。以学生端为例，其功能核心代码如下：

```
    FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);
    fab.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            toolbar.setTitle(getString(R.string.nav_Signup));
            FragmentTransaction transaction = fragmentManager.beginTransaction();
            transaction.remove(studentCourseListFragment);
            transaction.remove(personalFragment);
            transaction.remove(modifyPasswordFragment);
            transaction.remove(aboutFragment);
            transaction.remove(developingFragment);
            if (beginStudentSigninFragment.getArguments() == null) {
                beginStudentSigninFragment.setArguments(bundle);
            }
            transaction.replace(R.id.content_student, beginStudentSigninFragment);
            transaction.addToBackStack(null);
            transaction.commit();
```

```
        }
    });
```

## 4.3 教师端

当登录身份为教师时，系统自动进入到教师端界面，在此端下，拥有 3 个主要功能，分别为签到发起、添加课程、查看课程。

### 4.3.1 签到发起


图 12 签到发起界面


图 13 确认发起信息界面

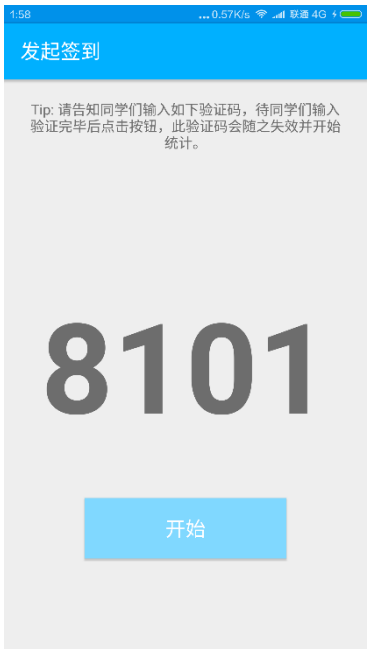


图 14 临时验证码界面

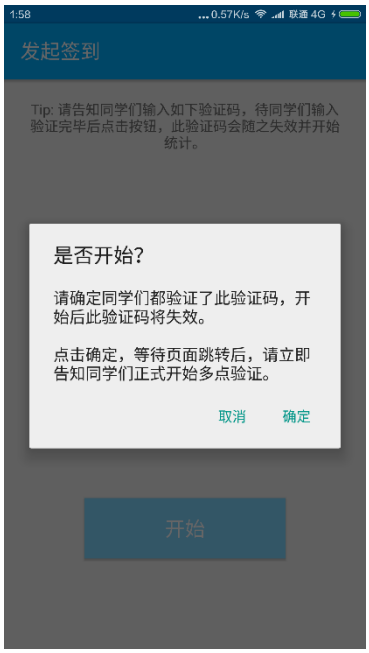


图 15 确认开始界面


图 16 开始成功界面

根据 3.2 小节主要业务流程实现教师发起签到，如图 12-图 16 所示。

教师进入发起签到功能后，显示界面如图 12 所示，在此界面中列出了课程状态为开课中的课程列表，每一个列表项展示了课程名称、课程状态、课程信息和已签到次数。列表使用 RecyclerView 与 CardView 实现，每个列表项监听用户的点击事件，进入图 13 所示界面。RecyclerView 的核心代码如下：

```
recyclerView = (RecyclerView) view.findViewById(R.id.recyclerView_course_list);
recyclerView.setLayoutManager(new
GridLayoutManager(CourseListFragment.super.getActivity().getApplicationContext(), 1));
RecyclerView.Adapter adapter = new CourseListAdapter();
recyclerView.setAdapter(adapter);
```

进入图 13 所示界面后，可以看到本界面分为三个部分，第一部分为地理位置信息，显示教师当前所在位置的地图以便确认位置；第二部分为课程名和课程信息展示，便于确认发起签到的课程；第三部分为发起签到按钮，监听用户的点击事件，用户可以点击发起签到按钮，点击后进入图 14 所示界面。本系统使用 Baidu Map Api，其功能核心代码如下：

```
mMapView = (MapView) view.findViewById(R.id.bmapView);
mBaiduMap = mMapView.getMap();
mBaiduMap.setMyLocationEnabled(true);
mBaiduMap.setMapStatus(MapStatusUpdateFactory.newMapStatus(new
MapStatus.Builder().zoom((float) 19).build()));
UpdateMap updateMap = new UpdateMap();
updateMap.execute();
```

进入图 14 所示界面后，可以看到一个四位临时验证码。用于学生签到时输入验证，生成此码时需要保证全局唯一，其 PHP 核心代码如下：

```
$query = "INSERT INTO `FingerSignin`.`FREQUENCY`
(`COURSE_idCOURSE`,`frequency`,`t_code`,`t_lat`,`t_lng`,`t_timestamp`)VALUES('" . $course_id .
"',(SELECT IFNULL(MAX(`FREQUENCY`),0) FROM `FingerSignin`.`FREQUENCY` tmp WHERE
`COURSE_idCOURSE`='" . $course_id . "')+1,RIGHT(CONCAT('0000',floor(rand()*10000)),4),'" .
$t_lat . "','" . $t_lng . "',FROM_UNIXTIME('" . $t_timestamp . "'))";

$result = $pdo->exec($query);
while ($result != 1) {
    $result = $pdo->exec($query);
}
```

点击开始后进入如图 15 所示界面，继续确认开始。点击确定即正式开始签到流程并且四位临时验证码失效，如图 16 所示，其 PHP 核心代码下：

```
$query = "UPDATE `FingerSignin`.`FREQUENCY` SET `t_code`= NULL,`t_timestamp` =
```

```
NOW(),`status`='1' WHERE `COURSE_idCOURSE`='" . $course_id . "'AND `frequency`='" .
$frequency . "'";
    $result = $pdo->exec($query);
```

### 4.3.2 添加课程

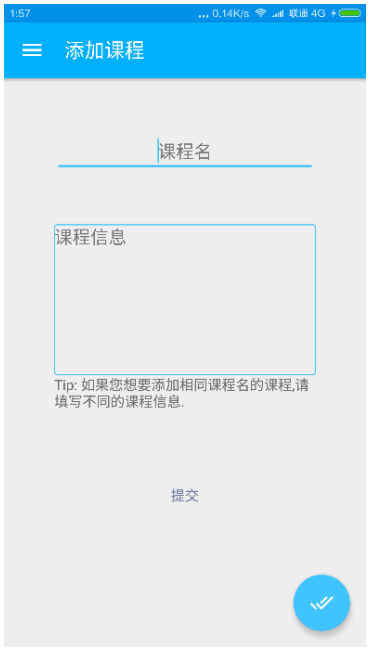


图 17 添加课程界面

当教师需要添加新的课程时候，可以进入添加课程界面添加新的课程，如图 17 所示。提交成功后，可以发起此课程的签到或查看此课程。添加课程的 REST 接口如下：

```
post:/teacher/addCourse.php
body:{
   t_number: "",
   course_name: "",
   course_comment: "",
   condition: ""
}
```

### 4.3.3 查看课程

教师可以在通过查看课程界面查看到自己创建的所有课程，如图 18 所示。

灰色（#E0E0E0）列表项表示本门课程已结课或已取消，白色列表项表示此门课程正在开课中，其核心代码如下：

```
if (courseListTeacherVO.getBody().getCourseList().get(position).getStatus().equals("0")) {
      //开课中
      holder.tv_course_status.setText(getString(R.string.status_class_starting));
} else if (courseListTeacherVO.getBody().getCourseList().get(position).getStatus().equals("1")) {
      //已结课
```

```
            holder.tv_course_status.setText(getString(R.string.status_class_done));
            holder.course_item_teacher.setCardBackgroundColor(0xFFE0E0E0);
        } else if (courseListTeacherVO.getBody().getCourseList().get(position).getStatus().equals("2")) {
            //已取消
            holder.tv_course_status.setText(getString(R.string.status_class_cancelled));
            holder.course_item_teacher.setCardBackgroundColor(0xFFE0E0E0);
        }
```


图 18　查看课程界面


图 19　课程详情界面(1)

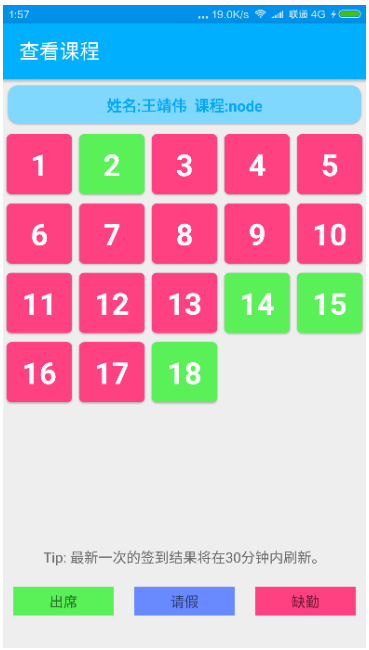


图 20　课程详情界面(2)

点击任意列表项进入课程详情界面，如图 19 所示。此界面中分为三个部分，第一部分展示了课程名、课程信息和教师头像。第二部分为两个按钮，修改课程状态和修改课程信息点击修改课程状态按钮，可以选择此课程状态为开课中、已结课或已取消。点击修改课程信息按钮可以修改此课程的信息。第三部分为学生签到情况列表，显示了每个学生的姓名、编号、头像以及出勤次数、请假次数和缺勤次数。使用绿色（#5AF158）表示出勤，蓝色（#6889FF）表示请假，粉红色（#FF4081）表示缺勤。

点击任意学生进入查看学生的每次签到情况，如图 20 所示。此列表也为使用 RecyclerView 与 CardView 实现，只不过定义布局为 GridLayoutManager，其核心代码如下：

```
        recyclerView = (RecyclerView) findViewById(R.id.recyclerView_course_freq);
        recyclerView.setLayoutManager(new GridLayoutManager(StudentFreqActivity.this,
(Util.px2dip(StudentFreqActivity.this, WelcomeActivity.getScreenW()) / 100) + 2));
        RecyclerView.Adapter adapter = new CourseFreqAdapter();
        recyclerView.setAdapter(adapter);
```

## 4.4 学生端

当登录身份为学生时，系统自动进入到学生端界面，在此端下，拥有 2 个主要功能，分别为签到、查看课程。

### 4.4.1 签到

根据 3.2 小节主要业务流程实现学生签到功能，如图 21-图 24 所示。

教师公布四位临时验证码后，学生填入，如图 21 所示。此界面分为两个部分，第一部分为确认学生当前所在位置，实现方式与 4.3.1 小节图 13 相似。第二部分为临时码输入框，其核心代码如下：

```
@Override
protected void onDraw(Canvas canvas) {
    int width = getWidth();
    int height = getHeight();

    RectF rect = new RectF(0, 0, width, height);
    borderPaint.setColor(borderColor);
    canvas.drawRoundRect(rect, borderRadius, borderRadius, borderPaint);

    RectF rectIn = new RectF(rect.left + defaultContMargin, rect.top + defaultContMargin,
            rect.right - defaultContMargin, rect.bottom - defaultContMargin);
    borderPaint.setColor(Color.WHITE);
    canvas.drawRoundRect(rectIn, borderRadius, borderRadius, borderPaint);

    borderPaint.setColor(borderColor);
    borderPaint.setStrokeWidth(defaultSplitLineWidth);
    for (int i = 1; i < contentLength; i++) {
        float x = width * i / contentLength;
        canvas.drawLine(x, 0, x, height, borderPaint);
    }

    float cx, cy = height / 2;
    float half = width / contentLength / 2;
    for (int i = 0; i < textLength; i++) {
        cx = width * i / contentLength + half;
        canvas.drawCircle(cx, cy, contentWidth, contentPaint);
    }

}
```

点击提交后进入如图 22 所示界面，此界面显示了课程名、课程信息和教师头像，以确保签到课程无误。当教师发出确认口令后，点击“我已经听到老师说开始了”，即进入随机多点触控验证阶段，如图 23 所示。


图 21　输入临时码


图 22　确认课程信息界面

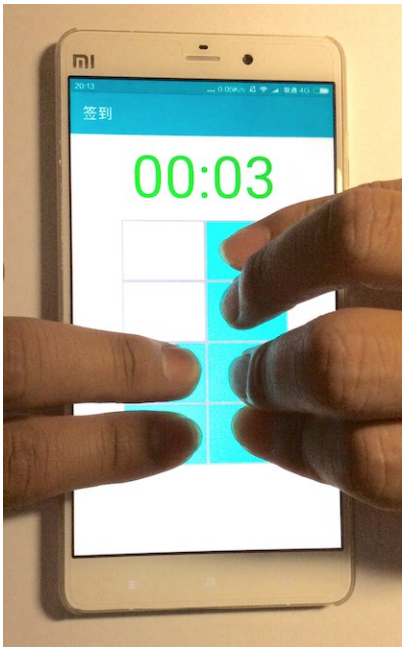


图 23　多指验证


图 24　数据上报成功界面

待计时结束后，自动上报数据并跳转如图 24 所示界面。数据以 json 格式上报，具体的格式如下：

```
{
  "key": "1463419730201250 41",
  "value": {
        "account_id": "20125041",
        "lat": "39.186251",
        "lng": "117.177553",
```

```
            "Course_id": "5",
            "Frequency": "16",
            "t_timestamp": "1463419730",
            "timestamp": "1463419760",
            "selectFinger": "1",
            "okSecond": "15",
            "condition":
"D5cRDWqeWf8aaF5qzMyg/pEtlHlSPNFToPTscbmlopgJMqxXopXcN6xdHzWN7/OQN4ML1w9or
EGwZrn8F8Z32TxMvM6rmJyV6D3lS4oU/ebAMOXjyqFkWuSYnsCbyfQf5VA1a0ptRJwBQucQAo
Vswdf3sV6XkhoauGEwt59O444="
        }
    }
```


### 4.4.2 查看课程

学生可以在此界面看到签到过的所有课程，如图 25 所示。灰色（#E0E0E0）列表项表示本门课程已结课或已取消，白色列表项表示此门课程正在开课中。


图 25 查看课程界面

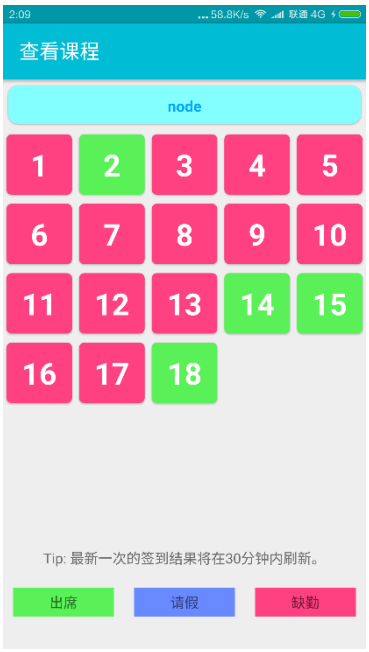


图 26 课程签到详情界面

点击进入任意课程，可以看到签到的详细情况，如图 26 所示。使用绿色（#5AF158）表示出勤，蓝色（#6889FF）表示请假，粉红色（#FF4081）表示缺勤。

其实现方式与 4.3.3 小节相似。

## 4.5 管理员端

当登录身份为管理员时，系统自动进入到管理员端界面，在此端下，拥有 3 个主要功能，分别为添加教师、查看教师、查看课程。

### 4.5.1 添加及查看教师

本系统不允许教师自主申请账号，需要管理员授予其教师权限。管理员可以在添加教师界面添加教师，如图 27 所示。


图 27 添加教师界面


图 28 查看教师详情界面

管理员可以在查看教师界面查看所有教师的详情，包括姓名、编号、头像、联系方式等信息，如图 28 所示。

### 4.5.2 查看课程

管理员可以在通过查看课程界面查看到所有教师创建的所有课程，如图 29 所示。

点击任意列表项进入课程详情界面，如图 30 所示。

此界面中分为两个部分，第一部分展示了课程名、课程信息和教师头像。第二部分为学生签到情况列表，显示了每个学生的姓名、编号、头像以及出勤次数、请假次数和缺勤次数。

点击任意学生进入查看学生的每次签到情况，如图 31 所示。此列表也为使用 RecyclerView 与 CardView 实现，定义布局为 GridLayoutManager。

为了体现一致性，仍然使用绿色（#5AF158）表示出勤，蓝色（#6889FF）表示

请假，粉红色（#FF4081）表示缺勤。

其实现方式与 4.3.3 小节、4.4.2 小节相似。

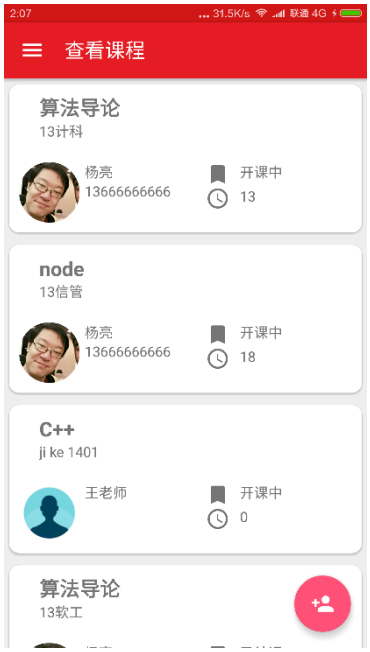


图 29　查看课程界面

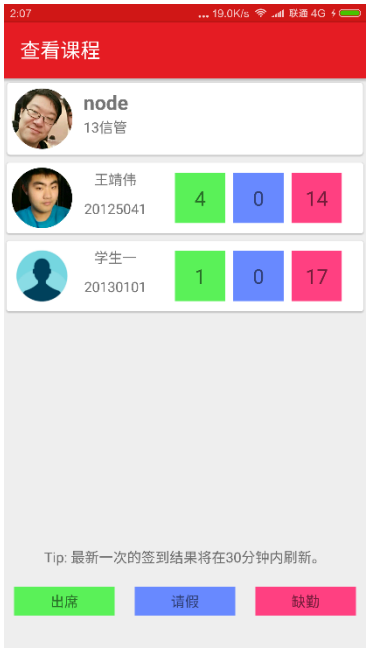


图 30　课程详情界面(1)

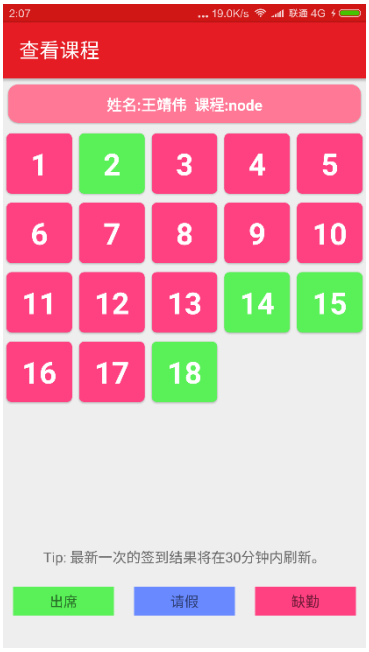


图 31　课程详情界面(2)

## 4.6　个人中心

个人中心是每个账号的账号信息所在的位置，包括了个人信息和修改密码功能。

### 4.6.1　个人信息

进入个人信息界面，如图 32 所示。用户可以在此界面查看个人信息或修改头像、电话号码或性别。

用户头像几乎是每个应用都标准品配备的功能。在 Android 开发中，头像的截取、压缩、上传至今没有一套完整的解决方案。在不同的 Android 机型中，由于系统的碎片化程度高、定制化程度深，往往对一些底层 framework 进行了修改，导致同样的一段裁剪图像的代码不能在所有的机型中适配。对于此种情况，我们找到了一个完整的解决方案，从选取到上传，完美解决这一难题。

第一步，选取图片进行剪裁。调用系统自身的 com.android.camera.action.CROP，设置过滤器为"image/*"进行图片的选取。相应核心代码如下：

```
EXTRA_OUTPUT = new File(Environment.getExternalStorageDirectory(),
        File.separator + Util.APP_Folder_name + File.separator + "my_icon");
Uri uri = data.getData();
Intent intentCrop = new Intent("com.android.camera.action.CROP");
intentCrop.setDataAndType(uri, "image/*");
intentCrop.putExtra("aspectX", 1);
intentCrop.putExtra("aspectY", 1);
intentCrop.putExtra("scale", true);
intentCrop.putExtra(MediaStore.EXTRA_OUTPUT, Uri.fromFile(EXTRA_OUTPUT));
intentCrop.putExtra("return-data", false);
startActivityForResult(intentCrop, Const.RESULT_IMG_CROP);
```


图 32　个人信息界面

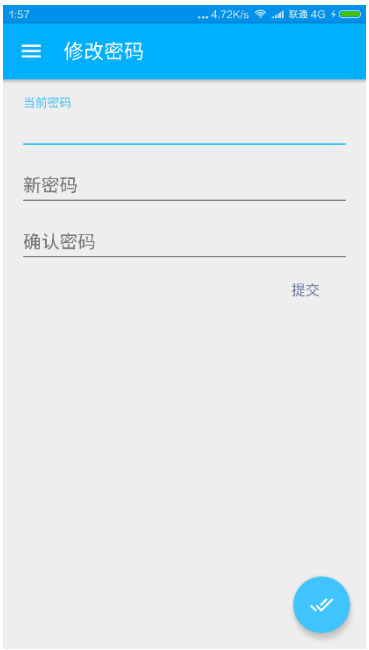


图 33　修改密码界面

第二步，剪裁后进行压缩，按照输出的尺寸进行压缩。相应的核心代码如下：

```
private Bitmap ratio(String imgPath, float pixelW) {
    BitmapFactory.Options newOpts = new BitmapFactory.Options();
    newOpts.inJustDecodeBounds = true;
    newOpts.inPreferredConfig = Bitmap.Config.RGB_565;
    Bitmap bitmap = BitmapFactory.decodeFile(imgPath, newOpts);

    newOpts.inJustDecodeBounds = false;
    int w = newOpts.outWidth;
    float ww = pixelW;
    int be;
    if (w > ww) {
        be = (int) (w / ww);
    } else {
        be = 1;
    }
    if (be <= 0) {
        be = 1;
    }
    newOpts.inSampleSize = be;
    bitmap = BitmapFactory.decodeFile(imgPath, newOpts);
```

```
        return bitmap;
    }
```

第三步，上传头像。使用 AsyncTask 异步上传，避免主线程阻塞。相应 Android 代码如下：

```
    private class Upload extends AsyncTask<File, Void, String> {
        @Override
        protected String doInBackground(File... f) {
            try {
                OkHttpClient client = new OkHttpClient();
                RequestBody fileBody = RequestBody.create(MediaType.parse("image/jpeg"),
f[0]);
                MultipartBody multipartBody = new MultipartBody.Builder()
                        .setType(MultipartBody.FORM)
                        .addFormDataPart("number", account_id)
                        .addFormDataPart("MAX_FILE_SIZE", "300000")
                        .addFormDataPart("userfile", MD5_Util.MD5(account_id +
account_name) + ".jpg", fileBody)
                        .build();
                Request request = new Request.Builder()
                        .url(Util.Final_HOST + "/account/uploadimg.php")
                        .post(multipartBody)
                        .build();
                okhttp3.Response response = client.newCall(request).execute();
                Log.d("1", "response ->" + response.body().string());
            } catch (Exception e) {
                e.printStackTrace();
            }

            return null;
        }
    }
```

在服务器端 PHP 用来接收并保存相应的头像，相应的 PHP 代码如下：

```
    $number = $_POST['number'];
    $uploaddir = '../upload/';
    $uploadfile = $uploaddir . basename($_FILES['userfile']['name']);

    $res = array();
    $type = strstr($uploadfile, ".");
    $allow_file = explode("|", "jpeg|jpg");

    $new_upload_file_ext = strtolower(end(explode(".", $_FILES['userfile']['name'])));

    if (move_uploaded_file($_FILES['userfile']['tmp_name'], $uploadfile)) {
        $body = array();
        $body['small_img'] = '/upload/' . $_FILES['userfile']['name'];
        $pdo = new PDO(DB_DSN, DB_USER, DB_PASSWORD);
        $pdo->query("SET NAMES utf8");
        $query = "UPDATE `FingerSignin`.`ACCOUNT` SET `small_img`='" . $body['small_img'] .
"'WHERE `_id`='" . $number . "'";
        $pdo->exec($query);
    }
```

### 4.6.2 修改密码

进入修改密码界面，如图 33 所示。用户可以在此界面修改密码。其 PHP 核心代码如下：

```
$query = "UPDATE `FingerSignin`.`ACCOUNT` SET `password`='" . $newpwd . "' WHERE `_id`='" . $number . "'";
$result = $pdo->exec($query);
```

## 4.7 关于指尖

用户可以在关于界面查看关于信息，如图 34 所示。

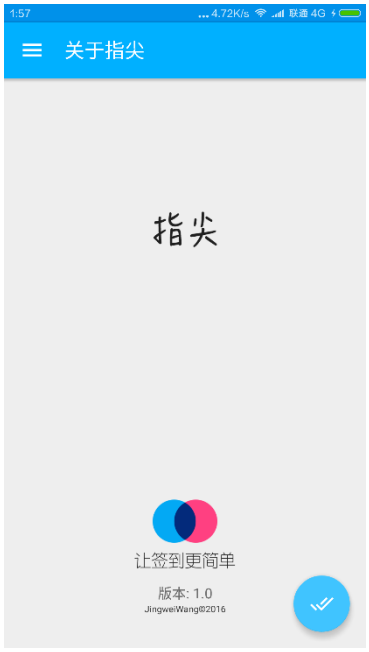


图 34 关于界面

在此界面内，不仅能够看到本系统的 Logo，还可以看到版本号，获取版本号的核心代码如下：

```
private static String getVersionName(Context context) {
    return getPackageInfo(context).versionName;
}

private static PackageInfo getPackageInfo(Context context) {
    PackageInfo pi = null;

    try {
        PackageManager pm = context.getPackageManager();
        pi = pm.getPackageInfo(context.getPackageName(),
                PackageManager.GET_CONFIGURATIONS);

        return pi;
    } catch (Exception e) {
```

```
            e.printStackTrace();
        }

        return pi;
    }
```

# 5 关键技术及解决方案

在本系统设计及实现的过程中，遇到了一些难题和关键技术，其中包括了信息安全体系、时间戳同步机制、随机多点触控、高并发及负载均衡、异步验证、地理围栏、Google Material Design、国际化与本地化等问题，为了解决这些问题，我们对这些问题提出了与之相对应的解决方案。

## 5.1 信息安全体系

信息安全主要包括以下五方面的内容，即需保证信息的保密性、真实性、完整性、未授权拷贝和所寄生系统的安全性。网络环境下的信息安全体系是保证信息安全的关键，包括计算机安全操作系统、各种安全协议、安全机制（数字签名、消息认证、数据加密等），直至安全系统，如 UniNAC、DLP 等，只要存在安全漏洞便可以威胁全局安全。信息安全是指信息系统（包括硬件、软件、数据、人、物理环境及其基础设施）受到保护，不受偶然的或者恶意的原因而遭到破坏、更改、泄露，系统连续可靠正常地运行，信息服务不中断，最终实现业务连续性[17]。

本系统在设计开发时及其注重安全，设计了一整套安全体系用于确保用户在使用过程中的信息安全，也极大的降低了系统被黑客攻击的几率。

在本系统中，主要确保了用户密码的安全存储，即使数据库泄露，黑客也不会轻易拿到用户的真实密码，其次注重了数据上报的安全性，极大的避免数据在上报过程中遭到黑客的篡改从而对系统的数据完整性造成的极大威胁。

（1）密码使用安全

在密码安全的解决方案上，采用 MD5 码策略，MD5 即 Message-Digest Algorithm 5（信息-摘要算法 5），用于确保信息传输完整一致。MD5 算法具有压缩性、容易计算、抗修改性、强抗碰撞等特点。

在端上，将用户输入的密码先转换成 MD5 码再上传服务器，数据传输、服务器

验证与数据库存储，都使用转换后的 MD5 码。其核心代码如下：

```
private static String getMD5(String info) {
    MessageDigest md5 = null;
    try {
        md5 = MessageDigest.getInstance("MD5");
    } catch (NoSuchAlgorithmException e) {
        e.printStackTrace();
        Log.e("MD5_Util", e.getMessage());
    }
    try {
        md5.update(info.getBytes("UTF-8"));
    } catch (UnsupportedEncodingException e) {
        e.printStackTrace();
        Log.e("MD5_Util", e.getMessage());
    }
    byte[] encryption = md5.digest();

    StringBuilder stringBuffer = new StringBuilder();
    for (int i = 0; i < encryption.length; i++) {
        if (Integer.toHexString(0xff & encryption[i]).length() == 1) {
            stringBuffer.append("0").append(Integer.toHexString(0xff & encryption[i]));
        } else {
            stringBuffer.append(Integer.toHexString(0xff & encryption[i]));
        }
    }

    return stringBuffer.toString();
}
```

（2）数据上报安全

为了解决这一问题，使用 SHA1WithRSA 签名算法，对上报数据进行签名。首先生成 RSA 密钥对：

第一步：生成私钥

```
$ openssl genrsa -out private_key.pem 1024
```

第二步：生成公钥，供服务器验证使用

```
$ openssl rsa -in private_key.pem -pubout -out public_key.pem
```

第三步：将 pem 格式私钥转为 pkcs#8 格式，供 Android 端签名使用

```
$ openssl pkcs8 -topk8 -inform PEM -in private_key.pem -outform PEM -nocrypt -out private_key_pkcs8.pem
```

在 Android 端，对数据进行签名，核心代码如下：

```
bytetemp = Base64.decode(pkcs8, Base64.DEFAULT);
privateKey = getPrivateKey(bytetemp);
sign = Signature.getInstance("SHA1WithRSA");
sign.initSign(privateKey);
sign.update(posttemp.getBytes());
byte[] data = sign.sign();
```

```
return Base64.encodeToString(data, Base64.DEFAULT).replace("\n", "").replace("\t", "").replace(" ", "").trim();
```

在服务端对签名进行验证，成功后方可继续进行操作，PHP 验证签名的核心代码如下：

```
function verity($data, $signature)
{
    $pubKey = file_get_contents(__DIR__ . '/public_key.pem');
    $res = openssl_get_publickey($pubKey);
    $result = (bool)openssl_verify($data, base64_decode($signature), $res);
    openssl_free_key($res);

    return $result;
}
```

## 5.2 时间戳同步机制

有些设备由于种种原因，系统时间与真实时间出现偏差，例如有意调快或调慢系统时间，导致验证结果出现偏差。为了保证随机多点触控验证阶段在精确的几秒钟内，我们要保证所有进行此次签到的学生的设备时间戳保持一致，所以对于时间戳的同步机制做出以下解决方案。

（1）当前时间戳获取

在时间戳获取上，对系统时间保持透明，使用网络时间戳进行时间戳的获取。为了减轻自己服务器的压力，网络时间戳获取统一通过访问 http://www.baidu.com/ 进行获取，获取后截取秒级时间戳即可。相应的核心代码如下：

```
public static String getNetWorkTime() {
    long timestamp;
    try {
        URL url = new URL("http://www.baidu.com");
        URLConnection urlConnection;
        urlConnection = url.openConnection();
        urlConnection.setConnectTimeout(5000);
        urlConnection.setReadTimeout(5000);
        urlConnection.connect();
        timestamp = urlConnection.getDate() / 1000;
        return String.valueOf(timestamp);
    } catch (Exception e) {
        timestamp = new Date().getTime() / 1000;
        return String.valueOf(timestamp);
    }
}
```

（2）多设备同步计时

为了使多个设备同步计时，除了获取当前网络时间戳 curr_timestamp_long，还需要获取教师确认开始的时间戳 t_timestamp_long。相关代码如下：

```
curr_timestamp_long = Long.valueOf(Util.getNetWorkTime());
t_timestamp_long = Long.valueOf(t_timestamp);
```

在正式开始随机多点触控验证之前，会给学生 15 秒倒计时准备时间，15 秒后正式开始计时验证。为了保证时间同步，使用相对时间进行计时。获取设备开机时长，在此基础上减去当前网络时间戳与教师确认时间戳之差，再预留 15 秒准备之间。将此作为计时的基数据，解决了时间戳的同步问题。相关核心代码如下：

```
time = (int) (curr_timestamp_long - 15 - t_timestamp_long);//倒计时 15s 准备时间
final Long base = SystemClock.elapsedRealtime() - time * 1000;
chronometer.setBase(base);
chronometer.start();
```

### 5.3 随机多点触控

使用 6-8 点多点触控事件在特定极短时间段内触发上报签到数据策略，可以有效防止代签。由于现在 Android 手持设备全部使用电容屏，理论上支持 10 点触控，问题只在于如何解决监听的逻辑。

（1）多点监听问题

在 Android 中，普通的对按钮进行监听，无法同时监听多点，所有使用触摸监听器对此进行了设计。在需要监听多点触摸的 8 个位置设置 setOnTouchListener 并声明一个布尔数组实时记录触摸状态，代码如下：

```
private Boolean finger_bool[] = new Boolean[]{false, false, false, false, false, false, false, false};
```

接下来，重载 onTouch 方法进行多点触控监听，当触发 ACTION_DOWN 事件，相应的多点计数加 1，并且在布尔数组的相应位置记录为 true；当触发 ACTION_UP 事件，相应的多点计数减 1，并且在布尔数组的相应位置记录为 false。这样，就可以极好的解决多点触控记录的难题。核心代码如下：

```
@Override
public boolean onTouch(View v, MotionEvent event) {
    if (v == finger_0 || v == finger_1
            || v == finger_2 || v == finger_3
            || v == finger_4 || v == finger_5
            || v == finger_6 || v == finger_7) {
```

```
                int i = 0;
                if (v == finger_0) i = 0;
                if (v == finger_1) i = 1;
                if (v == finger_2) i = 2;
                if (v == finger_3) i = 3;
                if (v == finger_4) i = 4;
                if (v == finger_5) i = 5;
                if (v == finger_6) i = 6;
                if (v == finger_7) i = 7;

                if (event.getAction() == MotionEvent.ACTION_DOWN) {
                    multiCount++;
                    finger_bool[i] = true;
                }
                if (event.getAction() == MotionEvent.ACTION_UP) {
                    multiCount--;
                    finger_bool[i] = false;
                }
            }
            return true;
        }
```

（2）6-8 点随机问题

在一个特定的位置预留 8 个点位，从这 8 个点位中随机出 6-8 个需求点位，按照公式（1）计算出，一共可以随机出 37 种不同的位置。

$$C_8^6 + C_8^7 + C_8^8 = 37 \quad (1)$$

对于这 37 种位置，可以通过一套可验证的策略来分配给用户。在本系统中，使用用户编号、经度、纬度、时间戳等信息进行 SHA1WithRSA 签名，后提取 MD5 码，最后取 MD5 码的前 8 位 16 进制数转换为 10 进制，再与 37 求余数，则得到分配给用户的随机点位。相应核心代码如下：

```
condition = SHA1WithRSA_to_BASE64.getBase64(getApplicationContext(), account_id + lat + lng + Course_id + Frequency + t_timestamp);
String condition_md5 = MD5_Util.md5(condition).substring(0, 7);
selectFinger = Integer.valueOf(condition_md5, 16) % 37;
```

而这个分配结果在服务器端也是可验证的，与 5.1 小节相辅相成，极大的确保了信息的安全性。

## 5.4 高并发及负载均衡

对于并发量极高的事件，例如大量学生同时上报签到数据，将这样的事件交由 Node.js 处理，Node 有如下特性：搭建与快速响应、事件驱动、异步 I/O、单线程、

跨平台等特点。所以 Node.js 就更适应于处理高并发、实时交互型以及 I/O 阻塞的应用。

对于不同的生产环境，设计了多种解决策略，并分析了其优缺点。

（1）单核心 CPU 服务器

固定一个监听端口，使用单一进程来接受从端上高并发来的数据，并把数据直接存入 Memcached，核心代码如下：

```
function pushToMemcache(key, value) {
    if (memcached == null) {
        memcached = new nMemcached("127.0.0.1:11211");
    }
    memcached.set(key, value, 0, function (err, result) {
        if (err) {
            console.error(err);
            var fail = {
                key: key,
                value: value
            };
            console.log("fail:" + JSON.stringify(fail));
        }
        console.dir(result);
    });
}
```

其缺点是，如果此进程异常退出，则会丢失其后续上报的数据，造成数据不完整。

（2）多核心 CPU 服务器自动负载均衡

使用 cluster 启动一个 Node 主进程，并打开和运行环境 CPU 核心数量的 Node 子进程，由主进程进行负载均衡，固定一个监听端口。由 cluster 进行自动负载均衡的核心代码如下：

```
if (cluster.isMaster) {
    require('os').cpus().forEach(function () {
        cluster.fork();
    });
    cluster.on('exit', function (worker) {
        console.log('worker ' + worker.process.pid + ' died');
    });
    cluster.on('listening', function (worker, address) {
        console.log("A worker with #" + worker.id + " is now connected to " + address.address
+ ":" + address.port);
    });
}
```

使用此策略的优点为可以极大程度的使用多核心 CPU 资源，避免资源浪费。子

进程如果异常退出，主进程则会自动避开异常退出的子进程从而自动负载。缺点为由于使用主进程进行负载均衡，消耗资源较大，可能会在负载时出现异常。

（3）多核心 CPU 服务器手动负载均衡

使用 Bash 脚本手动开启多个 Node 进程进行监听，监听多个端口。由移动端开发者进行端口分发负载均衡。Bash 核心代码如下：

```
function start() {
    port=22122
    for ((j=1; j<=$i; j++))
    do
        nohup node post_to_memcache_m_p.js $port >> node_m_p.log 2>&1 &
        ((port++))
    done
    i=0
    port=22122

    nohup node cureData.js >> cureData.log 2>&1 &
}
```

使用此策略的优点为，不用消耗主进程的负载资源。但是缺点为需要移动端开发者手动负载均衡，另外，如果某一个进程异常退出，也会丢失其后续上报的数据，造成数据不完整。

在选择策略时候，我们一般采取权衡利弊，采取和当前生产环境最适合的一种策略部署，在本文系统部署时，我们选用策略（1）。

## 5.5 异步验证

为了减轻服务器的压力，采用异步验证的策略验证学生签到结果并将数据固化至 MariaDB。所谓异步验证就是将学生上报的数据先缓存到 Memcached 中，不立即验证，Node.js 执行定时任务，每隔 10 分钟唤醒 PHP，使其从 Memcached 拉取数据并验证、固化。

Node 使用 node-schedule 设置定时任务，核心代码如下：

```
var rule = new schedule.RecurrenceRule();
rule.minute = [0, 10, 20, 30, 40, 50];
rule.second = 0;

var j = schedule.scheduleJob(rule, function () {
    http.get("http://10.161.67.26//memcache/cureData.php", function (res) {
        console.log("Got response: " + res.statusCode);
```

```
        }).on('error', function (e) {
            console.log("Got error: " + e.message);
        });
    });
```

每次唤醒 PHP 后执行验证并将签到数据固化至 MariaDB，核心代码如下：

```
    if (verity($verity_data, $condition)) {
        if ($okSecond >= 10) {
            $query = "INSERT INTO `FingerSignin`.`SIGNIN`
(`ACCOUNT__id`,`FREQUENCY_COURSE_idCOURSE`,`FREQUENCY_frequency`,`signin_time`,
`signin_lat`,`signin_lng`,`flag`)VALUES ('" . $account_id . "','" . $Course_id . "','" . $Frequency .
"',FROM_UNIXTIME($timestamp),'" . $lat . "','" . $lng . "','1')";
        } else {
            $query = "INSERT INTO `FingerSignin`.`SIGNIN`
(`ACCOUNT__id`,`FREQUENCY_COURSE_idCOURSE`,`FREQUENCY_frequency`,`signin_time`,
`signin_lat`,`signin_lng`,`flag`)VALUES ('" . $account_id . "','" . $Course_id . "','" . $Frequency .
"',FROM_UNIXTIME($timestamp),'" . $lat . "','" . $lng . "','0')";
        }
    } else {
        $query = "INSERT INTO `FingerSignin`.`SIGNIN`
(`ACCOUNT__id`,`FREQUENCY_COURSE_idCOURSE`,`FREQUENCY_frequency`,`signin_time`,
`signin_lat`,`signin_lng`,`flag`)VALUES ('" . $account_id . "','" . $Course_id . "','" . $Frequency .
"',FROM_UNIXTIME($timestamp),'" . $lat . "','" . $lng . "','0')";
    }
    $result = $pdo->exec($query);
    if ($result == 1) {
        $memcache->delete($key);
    }
```

## 5.6 其他技术

在本系统设计与开发的过程中，除了上面五个关键技术以外，还涉及到以下几个技术。

### 5.6.1 地理围栏

地理围栏（Geo-fencing）是 LBS 的一种新应用，就是用一个虚拟的栅栏围出一个虚拟地理边界。当手机进入、离开某个特定地理区域，或在该区域内活动时，手机可以接收自动通知和警告。有了地理围栏技术，位置社交网站就可以帮助用户在进入某一地区时自动登记。

在本系统使用地理围栏技术判定学生签到位置是否在可以接受的范围内。我们将根据教室的大小，误差考虑进去，将围栏范围设定为半径 50 米，在范围以内则可以继续进行多指验证，如果超出地理围栏，则会标记出课程教师的位置以示提醒，如图 35 所示。


图 35　超出围栏提示

本系统采用 Baidu Map Api，其核心代码如下：

```
    LatLng t_latlng = new LatLng(Double.parseDouble(verificationVO.getBody().getT_lat()),
Double.parseDouble(verificationVO.getBody().getT_lng()));
    LatLng my_latlng = new LatLng(lat, lng);

    double distance = DistanceUtil.getDistance(t_latlng, my_latlng);
    if (distance <= 50.0) {
        Intent intent = new Intent(getActivity(), MultitouchActivity.class);
        intent.putExtras(bundle);
        startActivity(intent);
    } else {
        mBaiduMap.clear();
        LatLng target_latlng = new
LatLng((Double.parseDouble(verificationVO.getBody().getT_lat()) + lat) / 2,
                (Double.parseDouble(verificationVO.getBody().getT_lng()) + lng) / 2);
        BitmapDescriptor bitmap = BitmapDescriptorFactory
                .fromResource(R.drawable.ic_local);
        OverlayOptions option = new MarkerOptions()
                .position(t_latlng).icon(bitmap);
        mBaiduMap.addOverlay(option);
        mBaiduMap.setMapStatus(MapStatusUpdateFactory.newLatLng(target_latlng));
        mBaiduMap.setMapStatus(MapStatusUpdateFactory.newLatLngBounds(new
LatLngBounds.Builder()
                .include(t_latlng).include(my_latlng).build()));
        Snackbar.make(view.findViewById(R.id.fragment_begin_student_signin),
R.string.Student_verify_distance_error, Snackbar.LENGTH_LONG)
                .show();
    }
```

### 5.6.2　Google Material Design

通过构建系统化的动效和空间合理化利用，并将两个理念合二为一，构成了实体

隐喻。与众不同的触感是实体的基础，这一灵感来自 Google 对纸墨的研究，Google 相信，随着科技的进步，应用前景将不可估量。

实体的表面和边缘提供基于真实效果的视觉体验，熟悉的触感让用户可以快速地理解和认知。实体的多样性可以让我们呈现出更多反映真实世界的设计效果，但同时又绝不会脱离客观的物理规律。这就是原质化设计（Material Design）。

在设计与开发本系统时，完全遵守了 Google Material Design，遵循了优秀设计的经典。

例如，使用 SnackBar 代替了 Toast，使用 Toolbar 代替了 ActionBar，使用 RecyclerView 代替了 ListView 等等，这些既美观了系统的界面、丰富了功能、简化了开发步骤，更重要的是，新的技术对于系统的优化更加彻底，使用户体验更加流畅。

### 5.6.3 国际化与本地化

国际化的 Internet 需要国际化的软件，国际化的软件意味着能够打进不同地区的市场。Android 的设计本身就是国际化的，所以本系统在设计和开发的时候，适配了简体中文、繁体中文、英文三种语言的国际化版本。


图 36 简体中文版欢迎界面


图 37 繁体中文版欢迎界面


图 38 英文版欢迎界面

如图 36-图 38 所示，分别现实简体中文、繁体中文、英文的欢迎页面。

当 Android 系统为简体中文环境时，就加载 res/drawable-zh-rCN 目录下的 Drawable 资源和 res/values-zh-rCN/string.xml 目录下的字符串文件；当 Android 系统为繁体中文环境时，就加载 res/drawable-zh-rTW 目录下的 Drawable 资源和 res/ values-zh-rTW/string.xml 目录下的字符串文件；当 Android 系统为英文环境时，就加载 res/drawable-en-rUS 目录下的 Drawable 资源和 res/values-en-rUS /string.xml 目录下的字符串文件。

# 6 总结与展望

## 6.1 本系统特点

本系统是为高等院校在校大学生所设计开发的课堂签到系统。本系统充分对现阶段的课堂考勤这一事件有全局性的把握，通过技术与策略相结合的方式来规避各项弊端和漏洞，使学生不再有机会代替别人签到。

对系统进行的大量集成测试表明：系统具有极高的稳定性、可用性和易用性。本系统在实际使用中也获得了令人满意的反馈。

## 6.2 本系统的创新点

本文首次开创了使用双手 6-8 指随机多点触控进行大学生课堂签到的模式，为今后课堂签到的方式方法、流程优化提供了重要的参考价值。

在本系统设计及实现的过程中，对于一些难题和关键技术也进行了相应的创新，其中包括了信息安全体系、时间戳同步机制、随机多点触控、高并发及负载均衡、异步验证、地理围栏等。

## 6.3 进一步的研究方向

继续调研普通高等院校教师和学生的具体需求，完善请假制度及其功能，继续优化签到流程。设计并实现 iOS 客户端，使终端覆盖面更广。优化服务器性能，提升整体用户体验。

后期，在完善系统本身的情况下，研究本系统的商业运营方案及推广方案，使本系统正式投入大规模使用。

# 参考文献


[1] 许晨, 苏忠, 李燕. 基于 NFC 近场通讯技术的课堂智能签到系统的设计[J]. 金陵科技学院学报, 2015, 01:37-41.

[2] 肖潇. 基于手机签到与课堂答题系统开发[J]. 通讯世界, 2015, 14:83-84.

[3] 吴鹏, 吴国平, 徐强, 王磊. 基于移动终端的网络签到系统[J]. 沈阳师范大学学报(自然科学版), 2015, 04:551-554.

[4] 张建权, 刘宝静. 网络环境课堂签到系统的设计与开发[J]. 石家庄理工职业学院学术研究, 2014, 01:11-15.

[5] 雷金莉, 张伟, 李永博. 基于单片机的课堂签到系统设计与实现[J]. 宝鸡文理学院学报(自然科学版), 2007, 01:67-69.

[6] 李孟山, 武燕. 高校开放型课堂签到系统的设计与实施[J]. 福建电脑, 2009, 02:93+80.

[7] 魏晋. 基于 Android 的课堂签到与手机违规监测系统的设计与实现[J]. 软件工程, 2016, 02:37-38+54.

[8] 王丁, 王磐炬. 一种简易实用的课堂签到器[J]. 电子世界, 2005, 07:39-40.

[9] Google. Android[EB/OL]. https://www.android.com/.

[10] Dormando. memcached - a distributed memory object caching system[EB/OL]. http://memcached.org/.

[11] 王宏扬. 基于移动智能设备的停车场管理系统设计与实现[D]. 南开大学, 2014.

[12] 李刚. 疯狂 Android 讲义(第 3 版). 电子工业出版社, 2015.

[13] Ed Burnette. Android 基础教程(第 4 版). 人民邮电出版社, 2015.

[14] Oracle. Oracle Technology Network for Java Developers[EB/OL]. http://www.oracle.com/technetwork/java/index.html.

[15] 龚民. 基于人脸识别技术的新农保身份验证系统研究[D]. 湖南大学, 2014.

[16] 杨明华, 谭励, 于重重. LAMP 网站开发黄金组合 Linux+Apache+MySQL+PHP[M]. 北京：电子工业出版社, 2008.

[17] 钱纪初. 浅论网络信息安全的重要性[J]. 今日湖北(中旬刊), 2015, (10):143-143, 144.

# 附录

## 附录 A　开题报告

### 毕业设计（论文）开题报告

| 二级学院 | 信息工程学院 | 教学系 | 计算机系 | 专业班级 | 软件工程1201班 |
|---|---|---|---|---|---|
| 学生姓名 | 王靖伟 | 学　号 | 20125041 | 指导教师 | 杨　亮 |
| 毕业设计（论文）题目 | 基于多点随机触控的异步高并发签到系统的设计与实现 | | | | |

一、选题依据

课堂考勤是大学生课堂管理的一部分，传统的课堂考勤存在耗时长、难统计、浪费教育资源等诸多弊端。随着计算机及其信息技术的快速发展，人们对信息处理的速度要求也越来越高。

“智慧校园”的提出，加快了教学、科研、管理和校园生活的充分融合。一些附带硬件的签到方式起到了举足轻重的作用，也很好解决的了这一问题，但是随之而来的是另外一些问题，硬件难配置、难部署、价格贵等问题，例如指纹机、人脸识别机、虹膜识别机、打卡机、NFC近场通讯磁贴等。即使不考虑硬件的部署和价格问题，在学生操作的时候和统计的时候也会比较耗时并且结果很难导出。有一些使用传统台式机的桌面环境部署的课堂签到平台,具有易开发、易部署、使用方便等特点,但是只适用于机房和多媒体教室等环境，不适用于普通的课堂环境。还有一些基于OpenCV的方法用来进行课堂签到，其算法涵盖神经网络、机器学习、人工智能等复杂算法，其识别度不高也是现阶段计算机视觉的一大缺陷。使用手机扫二维码方式签到也存在代替签到、二维码容易破解、学生不在教室内签到等弊端。

现阶段的课堂考勤类的工具的研究及开发对于课堂考勤并没有这一事件有全局性的把握，一些学生会犀利其中的某些漏洞，从而仍然不按时到课堂甚至不到课堂。例如手持设备互借、代签、微信短信等方式传递考勤签到验证必要信息（二维码、验证码等）、非教室内签到等问题。

本课题针对于以上的各项弊端进行整体化、全局化的把握，通过技术+策略的方式来规避以上各项弊端，使学生不再有机会代替别人签到，也不会有在非教室内签到等问题。

本课题通过研究移动端设备多点触控、异步处理、高并发通信等技术难题，使用Android、PHP、Node.js、Memcached、MariaDB等技术设计并实现一个基于移动端设备的课堂签到系统，以上涉及到的这些技术也是现在在生产实践中非常热门的技术。在数据通信过程中全程使用SHA1WithRSA签名、加密、策略等方式保证了数据安全；使用Node.js + Memcached解决了签到上报时瞬时高并发问题；解决了地理围栏、手机绑定、自动登录问题；实现简体中文、繁体中文和英文的I18N全球化；严格使用Google Material Design进行UI设计并实现；使用6-8点多点触随机控事件触发上报签到数据策略，有效防止代签。即时、准确、高效地签到及统计，节省教学资源，以作为“智慧校园”的一部分。

“智慧校园”的发展和创建是当今高校完善自身教育体制，落实信息化教育，提高自身竞争力的重要途径，是未来教育的发展方向。“智慧校园”的建立使学校、社会、科技三者直观地连接在一起，在理论与实践之间架起桥梁，方便师生利用共享的校园资源，完成一体化、智能化的目标，为教育信息化发展起到重要的作用。

随着科技进一步的发展和网络的提速，必定还会出现更先进的签到方法和策略，但是某些时候势必会涉及到用户的隐私数据上报，例如地理位置信息。如何能够更精准的确认签到及上报信息并且不侵犯用户的相应隐私和权利，是今后研究的进一步方向。

二、研究内容和研究方法

(一)、研究的目标和内容主要包括：

1.研究目标和内容

(1)设计并实现一个基于Android平台的客户端Native应用；

(2)设计并实现能与Android客户端互通的服务器端程序；

(3)课堂签到的逻辑及相关策略的设计；

(4)服务器可以承受签到上报时的高并发负载压力；

(5)服务器相关策略的设计。

2.关键技术及难点

在数据通信过程中全程使用SHA1WithRSA签名、加密、策略等方式要保证数据安全；使用Node.js + Memcached解决签到上报时瞬时高并发的问题；解决地理围栏、手机绑定、自动登录等问题；初步实现简体中文、繁体中文和英文三种语言的I18N全球化；严格使用Google Material Design进行UI设计并实现；使用6-8点多点随机触控事件触发上报签到数据策略，有效防止代签；异步验证上报数据是否合法等。

(二)、实施方案：

1.大量阅读、参考相应文献，制定一整套的可行的课堂签到流程及策略。

2.初步验证相应的流程及策略，例如使用Apache ab工具测试Node及php性能。

3.开发服务端整体架构，包括PHP、Node.js、Memcached、MariaDB，完成相应服务器端的功能，放出REST接口以供Android端调用。

4.开发基于Android平台的客户端Native应用。

5.测试所有功能并部署上线。

(三)、技术可行性论证：

1.技术可行性分析指标

(1)在当前的限制条件下，该系统的功能目标能否达到；

(2)利用现有的技术，该系统的功能能否实现；

(3)规定的期限内，本系统的开发能否完成。

2.技术可行性分析

(1)本课题主要的技术难点在于解决高并发问题和手持设备的多点触控监听，使用Node.js + Memcached方案将大量并发数据存入Memcached再由PHP定期将Memcached中的数据固化到MariaDB中，解决了高并发这一问题。由于现在Android手持设备全部使用电容屏，理论上支持10点，问题只在于如何解决监听的逻辑。

(2)安全性：由于所有上行数据全部使用签名、加密或者策略，确保了数据的安全性，防止伪造数据或请求。

(3)可靠性：要考虑通信的可靠性和服务器的可靠性，在部署环境下增加带宽并利用相应容错、容灾策略即可解决。

(4)在规定期限内，本系统的开发可以完成。

3.技术可行性分析结论

技术方案可行，可以立即开始本课题。

三、预计可获得的成果（含可能取得的独特之处）

本课题是在熟悉Android平台开发技术和开发工具的基础上，构建一个基于Android移动手持设备的大学生课堂签到系统，解决了传统课堂考勤存在的耗时长、难统计、浪费教学资源等诸多弊端，实现及时、准确、高效地签到及统计，节省教学资源，作为智慧校园的一部分。

本系统使用Android、PHP、Node.js、Memcached、MariaDB等多项技术和策略实现，在数据通信过程中全程使用SHA1WithRSA签名、加密、策略等方式保证了数据安全；使用Node.js + Memcached解决了签到上报时瞬时高并发问题；解决了地理围栏、手机绑定、自动登录问题；实现了简体中文、繁体中文和英文的I18N全球化；严格使用Google Material Design进行UI设计并实现；使用6-8点多点触控事件触发上报签到数据策略，有效防止代签。

四、工作进度计划

| | |
|---|---|
| 2015/11/10 | 布置毕业设计、开题报告要求，开始需求分析 |
| 2016/3/1 | 交开题报告，检查初步分析，进一步完善系统分析 |
| 2016/4/3 | 完成系统分析，开始系统设计 |
| 2016/4/10 | 基本完成系统设计，逐步开始系统实现、调试 |
| 2016/5/1 | 开始毕业设计说明书的撰写，继续系统实现、调试 |
| 2016/5/15 | 软件系统设计完毕，全面进入毕业设计说明书的撰写 |
| 2016/5/22 | 修改、完善毕业设计说明书 |
| 2016/5/29 | 完成全部毕业设计任务 |

五、与开题有关的主要参考文献

[1] 李刚. 疯狂Android讲义（第3版）. 电子工业出版社，2015.
[2] Truong Hoang Dung. React Native：用JavaScript开发移动应用. 电子工业出版社，2015.
[3] Ed Burnette. Android基础教程（第4版）. 人民邮电出版社，2015.
[4] http://java.sun.com[EB/OL]
[5] 杨明华,谭励,于重重.LAMP网站开发黄金组合Linux+Apache+MySQL+PHP[M].北京：电子工业出版社,2008
[6] http://ieeexplore.ieee.org/Xplore/home.jsp[EB/OL]
[7] https://www.android.com/[EB/OL]
[8] http://memcached.org/[EB/OL]

六、已取得的与论文研究内容相关的成果

我已具有多个Android项目开发经验，并且已经较多地阅读关于Android平台开发和服务端开发的相关技术书籍。对于本课题整体的架构体系已经有了基本的把握。

指导教师意见

同意本课题进入设计（论文）阶段。

指导教师签字：　　2016年3月1日

说明：1.本报告必须在第八学期开学两周内经指导教师审阅并形成正式报告。
2.本报告作为指导教师审查学生能否开展课题研究和是否按时完成进度的检查依据，并接受学校的抽查。

## 附录 B　服务器环境部署

在本系统开发的过程中，将所有的代码托管在 coding.net 上，目的是容易进行版本控制、容易部署。下面将详细描述服务器部署过程：

（1）从 coding.net 获取代码

进入 wwwroot 网站根目录：

```
$ cd /home/wwwroot/
```

从 coding.net 上 clone 其托管的代码至本地：

```
$ git clone https://git.coding.net/jingwei_wang/FingerSignin_PHP.git
$ git clone https://git.coding.net/jingwei_wang/FingerSignin_node.js.git
```

（2）部署 PHP 代码

进入 wwwroot 网站根目录：

```
$ cd /home/wwwroot/
```

复制 php 工程文件夹至 default 文件夹：

```
$ cp -rf Figersigin_PHP/* default/
```

进入 default 文件夹：

```
$ cd default
```

新建 upload 文件，用于上传的保存头像图片：

```
$ mkdir upload
```

修改 upload 文件夹权限为 rwxrwxrwx，使其能正常保存图片：

```
$ chmod 777 upload
```

进入 database 文件夹：

```
$ cd database
```

使用 vim 修改数据库配置文件 db_config.php，使其用户名及密码可以正常登录：

```
$ vim db_config.php
```

（3）部署 Memcached

进入 FingerSignin_node.js 工程文件夹：

```
$ cd /home/wwwroot/FingerSignin_node.js
```

使用 bash 脚本开启 memcached 服务：

```
$ bash memcached.sh start
```

（4）部署 Node.js 代码

进入 FingerSignin_node.js 工程文件夹：

```
$ cd /home/wwwroot/FingerSignin_node.js
```

使用 npm 初始化安装 package.json 中注册的模块以供 Node.js 正常调用：

```
$ npm install
```

使用 bash 脚本开启 Node.js 服务：

```
$ bash post_to_memcache_s.js start
```

## 附录 C　网络请求接口状态码定义

表 C.1　状态码族定义

| 状态码族 | 含义 |
|---|---|
| 10000 | 消息 |
| 20000 | 成功 |
| 30000 | 失败 |
| 40000 | 错误 |

表 C.2　消息状态码族详细定义

| Stauts | Body | Message |
|---|---|---|
| 10001 | - | This device has logged on to other accounts. |
| 10002 | - | No This IMEI record. |
| 10003 | - | This type of account registration prohibited. |
| 10004 | - | Your account already exists. |
| 10005 | - | Please do not repeat add course. |
| 10006 | - | Course info already exists. |
| 10007 | - | No this course info. |
| 10008 | - | The file format is illegal. |
| 10009 | - | Gender parameters is illegal. |
| 10010 | - | Tel is illegal. |

表 C.3　成功状态码族详细定义

| Stauts | Body | Message |
|---|---|---|
| 20001 | Y | Login Successful. |
| 20002 | Y | Register Successful. |
| 20003 | - | Modify password successfully. |
| 20004 | - | Data into memcached Successful. |
| 20005 | Y | Add course successfully. |
| 20006 | Y | Add Freq successfully. |
| 20007 | - | Modify Course info successfully. |
| 20008 | - | Modify Course Status successfully. |
| 20009 | Y | Course List for teacher query successfully. |
| 20010 | Y | Course Freq List for teacher query successfully. |

续表 C.3

| Stauts | Body | Message |
|---|---|---|
| 20019 | Y | Get Freq status info Successful. |
| 20011 | Y | File is valid, and was successfully uploaded. |
| 20012 | - | Gender changed successfully. |
| 20013 | - | Tel changed successfully. |
| 20014 | Y | Get personal info Successful. |
| 20015 | - | Start Signin successfully. |
| 20016 | Y | Verify successfully. |
| 20017 | Y | Course List for student query successfully. |
| 20018 | Y | Course Freq List for student query successfully. |
| 20020 | Y | Get Data in Memcache Successful. |
| 20021 | Y | Course get result successfully. |
| 20022 | Y | Course List for admin query successfully. |
| 20023 | Y | Teacher List for admin query successfully. |

表 C.4　失败状态码族详细定义

| Stauts | Body | Message |
|---|---|---|
| 30001 | - | Login failed. Please check the number or password. |
| 30002 | - | Auto login failed. |
| 30003 | - | Register failed. |
| 30004 | - | Verify that failed, can not modify the password. |
| 30005 | - | Add course failed. |
| 30006 | - | Add Freq failed. |
| 30007 | - | Modify Course info failed. |
| 30008 | - | Modify password failed. |
| 30009 | - | Modify Course Status failed. |
| 30010 | - | Possible file upload attack! |
| 30011 | - | Gender changed failed. |
| 30012 | - | Tel changed failed. |
| 30013 | - | Get personal info faild. |
| 30014 | - | Start Signin failed. |
| 30015 | - | Verify failed. |
| 30016 | - | Get Freq status info faild. |

表 C.5　错误状态码族定义

| Stauts | Body | Message |
|---|---|---|
| 40001 | - | Signature error. |
| 40002 | - | Data submission Mode or Path Error. |
| 40003 | - | Data Format Error. |

## 附录 D　REST 接口定义 Markdown 文档

见另册。

## 附录 E　Node.js 使用方式 Markdown 文档

见另册。

## 附录 F　代码清单

见另册。

# 发表论文和科研情况说明

发表的论文：

[1] 王靖伟, 刘英超, 孟巍. 电子商务推荐系统研究综述[J]. 电子商务, 2014, 06:52+75.

参与的科研项目：

1. 天津商业大学大学生创新创业训练计划, 题目:电子商务推荐系统模型的设计与实现, 编号:2014008.

# 致谢

本论文在导师杨亮的精心指导下历经半年圆满完成。在这半年里我遇到的困难重重，多亏杨亮老师的帮助，我才能越过障碍，一步步走下去。杨亮老师对整篇论文的完成起到了非常重要的作用。恩师思路开阔、知识渊博、严谨治学、平易近人，将是我一生学习的榜样。

深深感谢我的母亲和家人给予我的帮助，是他们的关心和支持才激励我顺利完成学业。

感谢潘旭华教授、高岩副教授、侯立坤副教授、孟巍老师、尉斌老师、苗序娟老师，在本科期间得到了老师们大量的指导和帮助，使我受益良多。感谢书记王艳军老师、班主任金音老师、辅导员郭君伟老师和王斌老师，在本科期间对我的帮助。感谢天津商业大学理学院赵芬霞副教授和张与鸿老师，在全国大学生数学建模竞赛期间对我的帮助。

感谢天津商业大学国家级大学生创新创业训练计划 iCampus 团队给予我的思路与灵感。感谢我的师兄吴康明、白宇为本文的创新提供了主要的思路。感谢我的师兄吕承彬在本文系统分析阶段给予我的思路与灵感。感谢我的师妹李冰丹在本文写作阶段给予我的思路与灵感。感谢我的师弟丛雨楠为我提供大量参考书籍。

同时，我还要感谢在我本科期间所有给予我帮助的老师和同学。

特别感谢我的师妹耿婧文在我身边的陪伴。

最后，再次对杨亮老师致以诚挚的感谢和敬意。

**谨以此文献给我最敬爱的母亲！**