合肥工業大學

毕 业 论 文

设计题目 基于 Android 的实时共享白板的设计与实现

学生姓名 李磊

学　　号 2013214381

专业班级 软件工程 13-2 班

指导教师 张本宏、杨瑞

院系名称 软件学院

2017 年 6 月 1 日

目录

基于 Android 的实时共享白板的设计与实现

摘要：随着社会的飞速发展，人们生活水平大大提高，各种需求越来越复杂，因此各种项目的规模越来越大，项目成员越来越复杂，并且他们很有可能分布在不同地点。因此，提高项目组内成员之间沟通协作的效率成为亟待解决的问题。随着互联网的迅猛发展，人们已习惯于通过智能手机上的软件解决问题，他们需要一款简单、便捷、高效的软件来解决上述沟通问题，而众所周知，图文结合的沟通相比纯文字更加直观，也更加高效。基于此，本文研究了实现白板的方法和 Android 实时通信的技术，发现 HTML5 的 Canvas 可以方便地实现白板、HTML5 的 WebSocket 协议可用于实时通信并效率很高、Android 的 WebView 可以实现 Java 代码和 JavaScript 代码的互相调用，据此设计并实现了一个基于 Android 的实时共享白板 APP。本软件大致包括以下功能：安排会议、发送会议邀请、加入会议、会内权限管理、会议共享白板、会议群聊、联系人管理、账户管理。总结来说，通过使用本软件，用户不再需要坐在会议室里或电脑前，只要使用 Android 手机就可以随时随地召开白板会议，与自己的伙伴进行远程沟通与协作，可以有效提高团队成员之间沟通协作的效率。

关键词：安卓、实时、白板、会议、协作

Design and Implementation of Real-time Sharing Whiteboard Based on Android

Abstract: With the rapid development of society, people's living standards have been greatly improved and the demand is more and more complex, so the scale of various projects is getting bigger and bigger, the project members are more and more complex and likely to be distributed in different locations. Therefore, improving the efficiency of communication becomes an urgent problem to be solved. With the rapid development of the Internet, people have been accustomed to solve the problem through the software on the smart phone, and now they need a simple, convenient and efficient software to solve the above communication problem, as we all known, graphic communication compared to pure text is more intuitive and more efficient. Based on this, this thesis studied the realization of the drawing board and Android real-time communication technology, found that HTML5 Canvas can easily achieve the drawing board, HTML5 WebSocket protocol can be used for real-time communication and it is efficient, and through Android WebView, Java and JavaScript can call each other easily, and according to this, design and implement a real-time sharing whiteboard APP based on Android. The software mainly includes the following functions: scheduling meetings, sending meeting invitations, joining meetings, permission management within the meeting, meeting sharing whiteboards, meeting group chat, contact management, account management. In summary, through the use of the software, users no longer need to sit in the conference room or computer, you can hold board meetings at anytime in anywhere, and communicate and collaborate with your partners remotely.

Keywords: Android、real-time、whiteboard、meeting、collaboration

1. 绪论

1.1. 课题的提出与意义

随着我国普遍进入互联网时代,“互联网+”已经深入到国民工作生活的方方面面,人们逐渐习惯使用智能手机上的 APP 解决问题，譬如购物软件、外卖订餐软件、同城租房软件、打车软件等。随后，基于互联网，出现了另一个概念“共享经济”，指的是一种共用人力与资源的社会运作方式，它包括不同个人与组织对商品和服务的创造、生成、分配、交易、和消费的共享，源于“共享经济”的应用也比比皆是，譬如 Uber、摩拜单车等。互联网以其“连接万物”的能力，高效地传递信息，同时结合分布式计算，可以带来超大的计算能力;“共享”很好地解决了社会资源不足的问题，提高了社会的整体资源利用率。这两点在某种程度上揭示了人们的需求，即更简单快捷地解决以往需要现场沟通并支付资源独享费用的问题。

与此同时，人们生活工作中的要求和期望越来越高，例如，公司规模在变大，经常有多个公司并分布在不同的地方，项目的复杂度也在不断提升，项目组成员也随之不断增多。如此一来，沟通变得越来越重要，但也越来越难，尤其是跨地区沟通的效率低而成本高。如何解决这个问题？网络给出了答案，因为高速网络联通了你我，使得我们可以跨越地理相隔实现交互，而传统的即时通讯工具一般局限于富文本聊天，交互性不足，难以保证沟通的简单高效。而基于 Android 的实时共享白板希望通过创建一块虚拟会议空间,多人使用 Android 手机随时随地连入并共享这块空间，在里面可以通过白板绘画、文字、声音等方式实现信息的交互共享，从而简单快捷地解决团队沟通问题。

1.2. 课题的研究背景

在国内方面，网易云最近开放了多个通信与视频产品的云服务，其中就包括互动白板，提供包括多通道轨迹同步、自定义传输内容、多人白板互动、白板录制、文档转码共享等服务，主要应用于教学白板、涂鸦、“你画我猜”三种场景，但是该服务为收费服务，且费用高昂，月功能费为 1000 元。

相比国内，国外关于实时共享白板的研究开展的较早，应用也较多。有一款名为SyncPad 的软件，可以涂鸦，并且可以多人实时交互，但是它运行在 iPad 上，并且是收费软件；有一款名为 Whiteboard Pro 的软件，也是可以涂鸦，可以多人实时交互，并支持蓝牙连接。此外，还有多个网站提供多人协作功能，包括实时共享白板。

由此可以看出，实时共享白板在国内尚未普及，尤其是在手机端应用上，而目前Android 于智慧型手机市场的占有率已超过 80%，Android 开发技术也已相当成熟，因此，一款基于 Android 的手机端共享白板会议软件可以有效填补市场空缺，解决团队远程沟通协作难的问题。

1.3. 系统特色介绍

传统的会议白板一般运行于 PC 或者会议室专用设备，而随着生活节奏不断加快，人们的时间更加宝贵，特别是在大城市，交通堵塞难以避免，如果用户可以有效利用交通堵塞等碎片时间，高效地完成简短但必要的会议沟通，就能够节约时间处理其他的事情，从而使得工作更加游刃有余。在这个生活场景中，传统的会议白板就不能满足需求了，需要在手机端提出解决方案。

手机端会议也有很多种，如视频会议、电话会议等等，但是如果用户在室外，一个没有无线网的地方，视频会议会消耗大量流量，其费用是用户不愿支付的；电话会议，只能传递声音，不够直观，不适合讨论复杂问题。

而本系统是一个基于 Android 的手机端实时共享白板 APP，通过本软件，用户可以简单方便地开会、加会、邀请；通过白板绘画、共享资源和群聊结合，用户可以既直观又详细地表达自己的想法，本系统适合多种问题的讨论协商，例如界面设计问题、数学类问题、工程类问题等等；通过加会者权限管理，主持人可以控制会议的秩序。

总之，使用本系统，用户可以随时随地进行开会沟通，不再需要在会议室或电脑前，而且本系统的界面简洁、操作简单、交互性强、流量消耗少。

1.4. 内容安排

本论文整体分为六个大章节：

第一章为绪论，简要叙述了课题的意义、研究背景、系统特色以及内容安排。

第二章为基础知识，简要介绍了系统开发相关的技术。

第三章为系统需求分析与总体设计，该章节分析了系统的功能性需求和其他需求，制定了系统目标，并进行了系统架构、系统功能以及数据库的设计。

第四章为详细设计，文章将将系统划分为多个模块，然后逐个模块进行具体分析设计，每个模块一般包括概述、具体过程两部分。

第五章为系统实现与测试，文章介绍了系统主要界面的实现效果和系统部署情况，并对系统进行了必要的测试。

第六章为结论与展望，该章节对毕业设计进行了综述，说明了系统的完成情况与不足之处，并结合当前时代背景，展开了对未来的畅想。

2. 基础知识

2.1. Android

Android 是一个基于 Linux 的移动平台的操作系统，它是开源的，开发者可以按照一定的规则自由地对其进行定制开发。Google 提供了一套完整的的规范化的开发平台和开发体系，开发者通过 Google 提供的系统 API 和系统机制可以方便地开发出各种各样的应用软件。

2.2. PHP

PHP 是一种通用脚本语言，主要用于生成动态网页内容、操作服务器上的文件和数据库、数据加密、用户授权访问等。PHP 凭借其开源、免费、易学、服务器兼容性好、跨平台等特性，被称为“世界上最好的语言”。PHP 拥有超过 150 个扩展库，可供开发人员调用，很大程度上提高了开发效率和代码的性能，例如 PDO 扩展，它提供了 PDO 类来对数据库进行访问，开发人员使用不同数据库时候调用的函数名是相同的，使得应用层不用去关心具体要连接的数据库服务器的类型，除此之外，相比原生 MySQL，可以有效防御 SQL 注入攻击。

2.3. JavaScript

JavaScript 是一种解释型脚本语言，它是弱类型、函数优先的，一般作为开发网页的脚本语言，在其他环境也有使用。随着当前网站开发前后端更严格的分离，JavaScript 负责越来越多的业务逻辑检查、处理等工作，而不仅仅是单纯的 DOM 操作和实现网站特效。

2.4. React

React 是 Facebook 研发的 JavaScript 库，具有易入门、代码易重构、更好支持

响应式网页、高性能等特点。React 的主要原理包括 Virtual DOM、Components、State 和 Render。其中 Virtual DOM 就是在真实 DOM 上面抽象出一个对象，用来表示 DOM 应该怎么呈现，当需要更新页面的时候，不是直接更新真实 DOM，而是更新 Virtual DOM，React 会等到当前事件循环结束，通过 diff 算法计算 Virual DOM 和真实 DOM 的差别，并计算出最小步数来进行更新，从而提高了性能；Virual DOM 的每一个节点就是一个 commponent，它的存在使得 diff 算法更高效；State 包含定义 commponents 所需的数据，当数据改变时，它会调用 Render 进行重新渲染。

2.5. WebSocket

WebSocket 是 HTML5 中的一个新的协议，它和 HTTP 协议基本没关系，HTTP 中每个 response 都对应一个 request，也就是只有客户端主动请求后服务器才会回复，而不能做到服务器主动推送消息给客户端，基于这个原理的 AJAX 轮询和 polling 技术,可以变向实现实时交互,但是性能不佳;而 WebSocket 只需要客户端一次 request，建立持久连接后，服务器便可以主动向该客户端推送任意多次消息，同时较 AJAX 轮询和 polling 技术具有更高的效率和性能。WebSocket.org 对传统的轮询方式和 WebSocket 调用方式做过一个详细的测试和比较，将一个简单的 Web 应用分别用轮询方式和 WebSocket 方式来实现，并把测试结果用柱状图进行表示，如图 2.1。

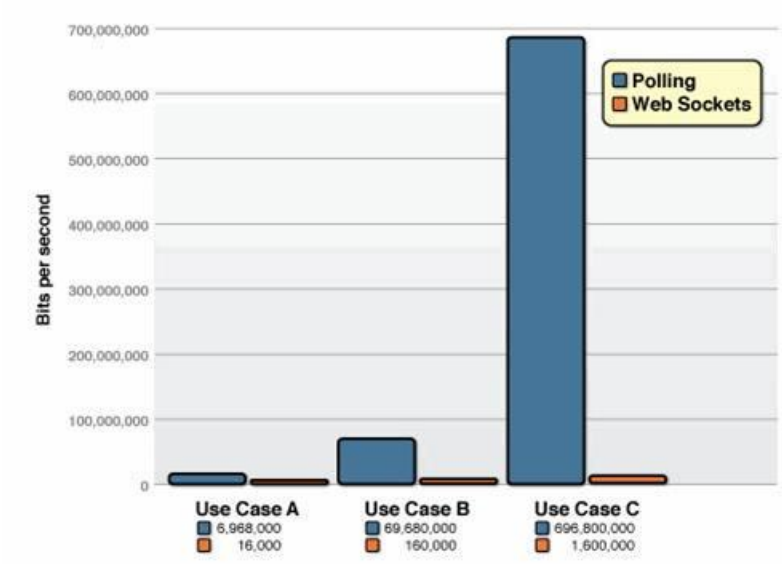


图 2.1　轮询和 WebSocket 的性能对比

2.6. GatewayWorker

GatewayWorker 是一个开源的 Socket 服务器框架，用 PHP 编写，该框架采用的是 Gateway 和 Worker 进程模型，它支持分布式部署、支持高并发、支持多种应用层、支持代码热更新、支持 HVVM 协议、提供心跳检测、在 Linux 服务器上可以做到守护进程化。使用该框架，开发者只需要简单配置并通过实现框架内部 Events 类的函数，就可以轻松监听 WebSocket 连接的建立、消息请求、连接断开等事件，并实现消息的单发、群发、广播等功能。

3. 需求分析和总体设计

3.1. 需求分析

3.1.1. 功能性需求

1) 为了方便用户随时随地进行团队沟通，首先需要把用户添加到系统中，因此需要注册、登录功能；
2) 为了映射实际中团队成员之间的关系，需要联系人管理功能，包括联系人的增删查；
3) 为了用户可以有序地加入沟通，并且保证沟通的安全性与私密性，一次团队沟通需要一个授权管理模块，也就是会议管理模块，包括会议的增删改查以及加入会议的密码验证；
4) 为了保证会议内部用户绘画、发言的可控性，需要在会议内部设置一位主持人，并且赋予他管理加会者的权利，因此需要会内权限管理；
5) 为了更加方便邀请别人加入会议，需要多种邀请加会方式；
6) 会议内部主要通过绘画、发言等方式进行信息交互，因此需要共享白板功能和群聊功能。

系统整体的业务流程以用户注册登录为起点。用户进入系统主页面后，可以安排会议，安排好会议可以查看、再次编辑、删除、邀请别人加会、进入会议，以主持人身份进入会议后，可以继续邀请别人加会、锁定会议、共享资源、使用白板、使用群聊、查看参与者列表、控制加会者权限等。以加会者的身份加入会议，可以邀请别人加会、使用白板、使用群聊、查看参与者列表。除此之外，用户还可以添加、删除、查看联系人以及进行一些账户设置操作等。系统业务流程图见图 3.1。

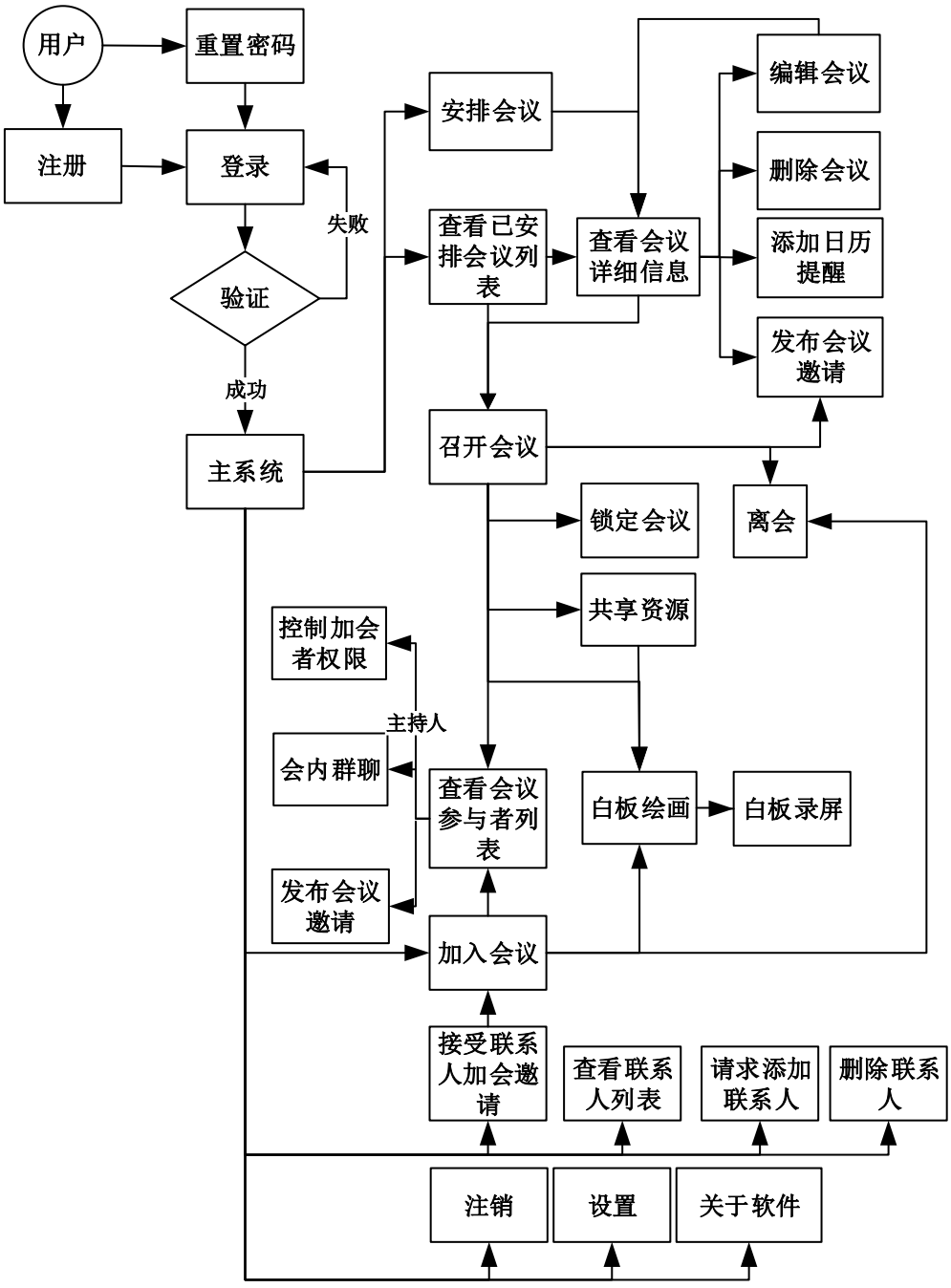


图 3.1　系统业务流程图

3.1.2. 其他需求

3.1.2.1. 速度

1）　注册验证码必须在 15 秒内发送到用户注册所用邮箱。

2）　登录必须在 5 秒内完成。

3) 所有用户查询必须在 5 秒内完成。

4) 开会、加会必须在 10 秒内完成。

3.1.2.2. 实时性

1) 白板绘画、群聊信息必须在 1.5 秒内同步到会议的其他参与者。

2) 联系人的添加申请、接受申请、拒绝申请、删除、加会邀请通知必须在 2 秒内推送到目标客户端。

3.1.2.3. 安全性

1) 需要保障用户账户安全，防止他人越权访问。

2) 需要保障服务器的数据安全,包括数据库和用户上传文件；

3) 需要保障通信安全，通信数据需要加密，防止传递的信息被他人监听并轻松得到信息的内容。

3.2. 总体设计

3.2.1. 系统目标

本软件是将传统的白板和手机端应用相结合开发设计的，主要实现以下目标：

1) 打造手机端白板会议，让用户可以随时随地进行开会沟通。

2) 尽量做到界面简洁美观、用户操作简单便捷。

3) 开会、加会流程简单易懂，会议邀请方式多样化，让用户可以轻松开会。

4) 共享白板绘画流畅，数据同步及时。

5) 群聊支持多种消息形式，滑动流畅，数据同步及时。

6) 对用户的联系人进行管理。

7) 对用户账户进行管理。

8) 软件运行稳定、安全可靠。

3.2.2. 系统架构设计

系统采用 C/S 架构，系统架构图如图 3.2 所示，其中客户端部分为安卓智能手机，通过互联网与服务器进行通信，服务器主要涉及到四个：Web 服务器、Socket 服务器、QQ 邮件服务器、极光推送服务器，四个服务器的作用如下：

1) Web 服务器：购买的腾讯云服务器，系统为 Ubuntu14.04，通过 Apache2.4.7 处理客户端发送的 HTTP 请求，包括授权访问检查、业务逻辑处理、文件上传

下载、数据库操作等。

2) Socket 服务器：运行 GatewayWorker，用户进行白板绘画、会内群聊的时候，每个客户端都和 Socket 服务器之间维持着一个 WebSocket 连接，用来进行数据和消息的实时同步。Socket 服务器就是负责处理 Socket 请求，单发、群发、广播消息的。

3) QQ 邮件服务器：用来给用户发送邮箱验证码。

4) 极光推送服务器：用于推送联系人添加、接受、拒绝、删除、邀请加会请求到客户端。

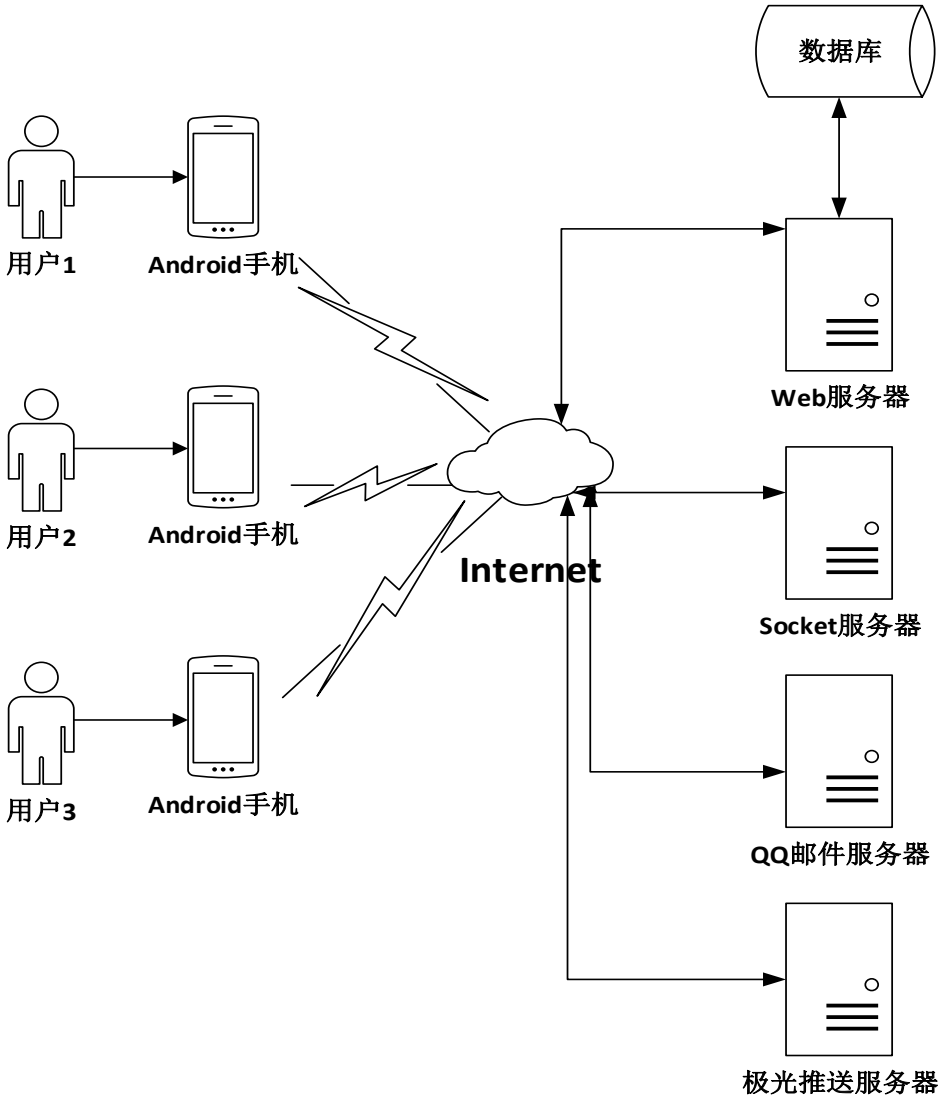


图 3.2　系统架构图

3.2.3. 系统功能设计

本系统功能设计如图 3.3 。

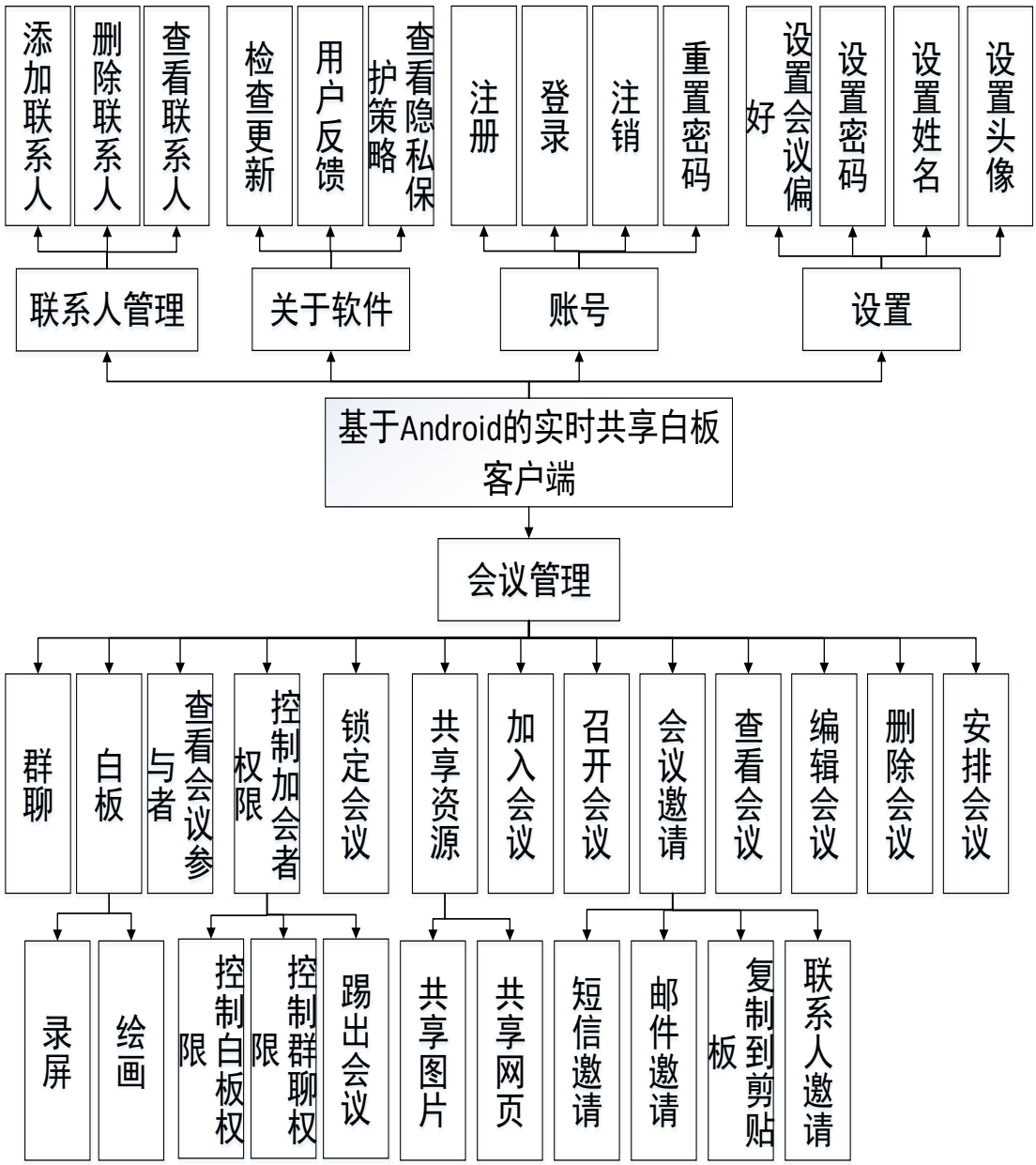


图 3.3　系统功能结构图

3.2.4. 数据库设计

数据库设计是系统总体设计中极其重要的一部分，因为数据库设计的好坏，会对系统性能、可维护性等产生相当巨大的影响。一个优秀的数据库设计方案既要依据范式的要求，又要紧密结合实际项目的需要，设计者需要在规范和性能之间进行斟

酌与平衡。

3.2.4.1. 数据库概念设计

通过对系统进行以上的需求分析，了解了系统的功能结构与业务流程，可以大致把数据库实体对象分为用户信息实体、会议信息实体、版本信息实体、系统消息信息实体等组成部分。

用户信息实体包括用户 ID、邮箱、姓、名、注册时间、最近登录时间、登录密码、登录 TOKEN、头像等属性。用户信息实体 E-R 图如图 3.4 所示。

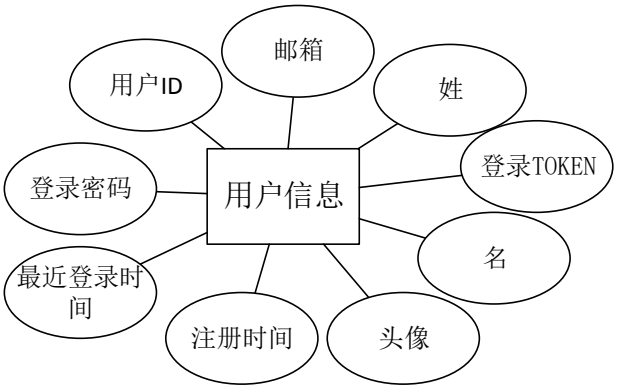


图 3.4　用户信息实体 E-R 图

会议信息实体包括会议 ID、会议号、主题、主持人用户 ID、加会者默认能否使用白板、加会者默认能否聊天、是否添加到日历提醒、入会密码、预期开始时间、预期结束时间、会议状态、会议描述、日历事件 ID 等属性。会议信息实体 E-R 图如图 3.5 所示。

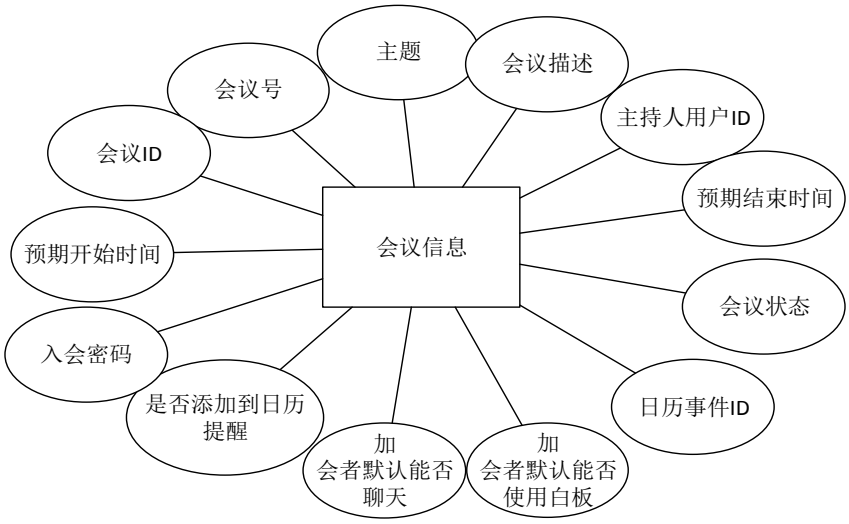


图 3.5　会议信息实体 E-R 图

版本信息实体包括版本 ID、应用名称、版本号、是否强制更新、标记、APK 下载地址、更新提示信息、发布时间等属性。版本信息实体 E-R 图如图 3.6 所示。

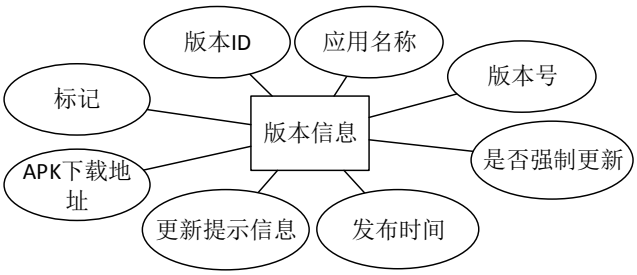


图 3.6　版本信息实体 E-R 图

系统消息信息实体包括系统消息 ID、标题、内容、类别、状态、目的用户邮箱、目的用户的姓、目的用户的名、目的用户头像、到达时间、全局唯一标识等属性。版本信息实体 E-R 图如图 3.7 所示。

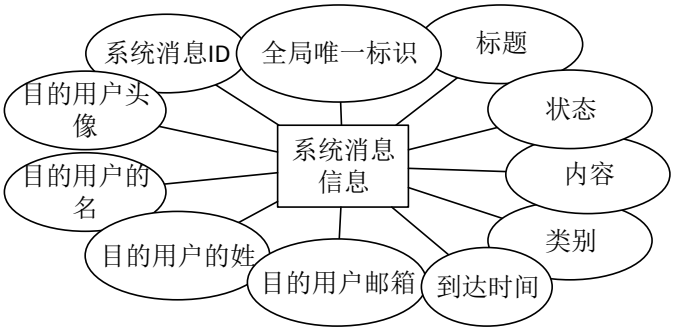


图 3.7　系统消息信息实体 E-R 图

系统总体 E-R 图如图 3.8 所示。

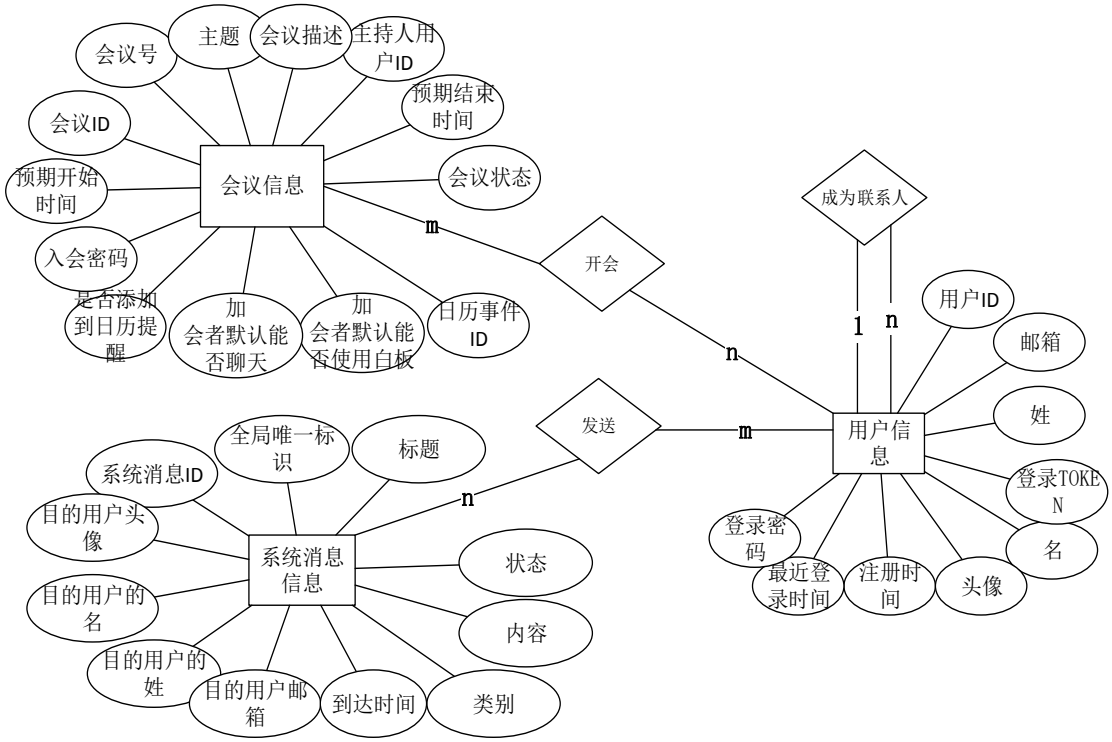


图 3.8　系统总体 E-R 图

3.2.4.2. 数据库逻辑设计

根据在数据库概念设计中给出的数据库实体 E-R 图，设计如下数据表结构。服务器端数据库采用 MySQL，客户端数据库采用 SQLite。

用户信息表用于保存用户注册信息，bd_user 表的结构如表 3.1 所示。

表 3.1　bd_user 表的结构

字段名	类型	长度	索引	注释
user_id	INT	10	主键	用户 ID
user_email	VARCHAR	30	主键	邮箱
user_family_name	VARCHAR	15		姓
user_given_name	VARCHAR	15		名
user_register_time	DATETIME	32		注册时间
user_login_recent_time	DATETIME	32		最近一次登陆时间
user_password	VARCHAR	32		登录密码

user_token	VARCHAR	32		Token
user_avatar	VARCHAR	60		用户头像地址

会议信息表用于保存用户安排的会议的信息，bd_meeting 表的结构如表 3.2 所示。

表 3.2 bd_meeting 表的结构

字段名	类型	长度	索引	注释
meeting_id	INT	10	主键	会议 ID
meeting_url	VARCHAR	12	主键	会议号
meeting_theme	VARCHAR	20		会议主题
meeting_host_user_id	INT	10	外键	主持人用户 ID
meeting_is_drawable	TINYINT	4		加会者默认能否使用白板绘画： 1：不能 2： 可以
meeting_is_talkable	TINYINT	4		加会者默认能否聊天： 1：不能 2：可以
meeting_is_add_to_calendar	TINYINT	4		是否添加到日历提醒：1：不添加 2：添加
meeting_password	VARCHAR	32		入会密码
meeting_start_time	DATETIME	32		会议预期开始时间
meeting_end_time	DATETIME	32		会议预期结束时间
meeting_status	TINYINT	4		会议状态： 1：未开始并且未到期 2：未开始并且过期了 3：正在进行 4：开会结束 5：锁定

event_id	BIGINT	20		日历事件 ID
meeting_desc	TEXT			会议描述

用户参加会议信息表用于保存用户参加会议的记录，bd_user_and_meeting 表的结构如表 3.3 所示。

表 3.3 bd_user_and_meeting 表的结构

字段名	类型	长度	索引	注释
user_and_meeting_id	INT	10	主键	用户参加会议 ID
check_in_type	TINYINT	4		用户参会类型：1 ：加会 2：主持会议
check_in_time	DATETIME	32		最新入会时间
check_out_time	DATETIME	32		最新离会时间
bd_user_user_id	INT	10	外键	参会者的用户 ID
bd_meeting_meeting_id	INT	10	外键	参会者进入会议的会议 ID

联系人信息表用于保存用户之间添加、删除联系人的记录，bd_friend 表的结构如表 3.4 所示。

表 3.4 bd_friend 表的结构

字段名	类型	长度	索引	注释
friend_id	INT	10	主键	联系人 ID
response_status	TINYINT	4		回复状态：1：未回复 2：拒绝 3：同意 4.好友关系已经删除
message_time	DATETIME	32		消息时间
bd_user_user_id	INT	10	外键	用户 1 的 ID
bd_user_user_id1	INT	10	外键	用户 2 的 ID

版本信息表用于保存版本发布信息，bd_version 表的结构如表 3.5 所示。

表 3.5 bd_version 表的结构

字段名	类型	长度	索引	注释
version_id	INT	10	主键	版本 ID
app_name	VARCHAR	32		应用名称
server_version	VARCHAR	32		服务器端 APP 版本号
last_force	TINYINT	4		是否强制更新：0：不强制 1：强制
server_flag	TINYINT	4		标记
update_url	TEXT			新版本 APP 下载地址
upgrade_info	TEXT			更新提示信息
update_time	DATETIME	32		版本发布时间

系统消息信息表位于客户端的 SQLite 数据库中，用于保存系统信息，bd_msg 表的结构如表 3.6 所示。

表 3.6　bd_msg 表的结构

字段名	类型	长度	索引	注释
_id	INTEGER	根据值的大小存储在 1、2、3、4、6 或 8 字节中	主键	系统消息 ID
EMAIL	TEXT			用户邮箱
TITLE	TEXT			消息标题
CONTENT	TEXT			消息内容
FAMILY_NAME	TEXT			用户的姓
GIVEN_NAME	TEXT			用户的名
FEATURE	TEXT			消息类别
AVATAR	TEXT			用户头像地址
STATUS	INTEGER	根据值的大小存		消息状态：0：初始

		储在 1、2、3、4、6 或 8 字节中		态 1：已同意 2：已拒绝
MSG_TIME	INTEGER	根据值的大小存储在 1、2、3、4、6 或 8 字节中		消息到达时间
TAG	TEXT			消息全局唯一标识：用户邮箱+时间戳毫秒值

4. 详细设计

4.1. 数据库操作类设计

在面向对象编程中，类用来封装和组织常用的方法和数据。具有良好的抽象层次和模式的数据库操作类的编写不仅可以方便代码维护，还可以减少编写重复代码工作量，是系统的后面优化的基础。

本系统的服务器端主要包括两个实体类，分别是用户信息实体类 User、会议信息实体类 Meeting；数据操作大致分为五种，分别是注册 Register、登录 Login、会议操作 MeetingOp、好友操作 FriendOp、基本用户信息更新操作 Update。本系统通过 PHP 的 PDO 扩展操作 MySQL 数据库，因此还需要封装一个数据库基本操作类 DBPDO，包括数据库连接及数据的增删改查操作。数据库操作类之间的关系如图 4.1 所示。

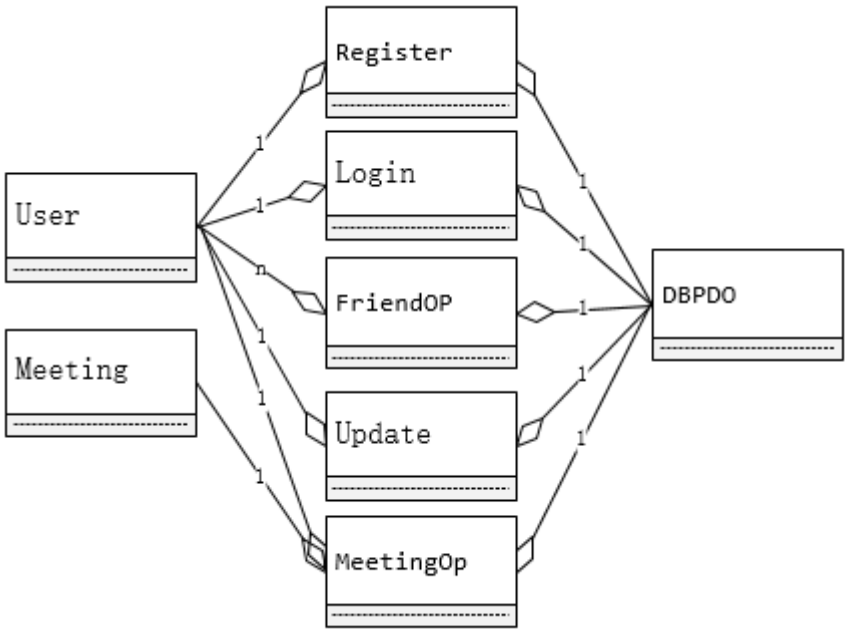


图 4.1　数据库操作类的类图

4.2. 账号模块设计

4.2.1. 账号注册

4.2.1.1. 账号注册概述

用户首次使用软件，需要注册个人账户，需要使用有效的邮箱进行注册，系统会发送验证码到该邮箱，用户查看后，填写正确的验证码，即可开始注册，设置头像、姓名、密码后，注册完成，直接登录进系统。

4.2.1.2. 账号注册具体过程

账户注册具体过程如图 4.2 所示。

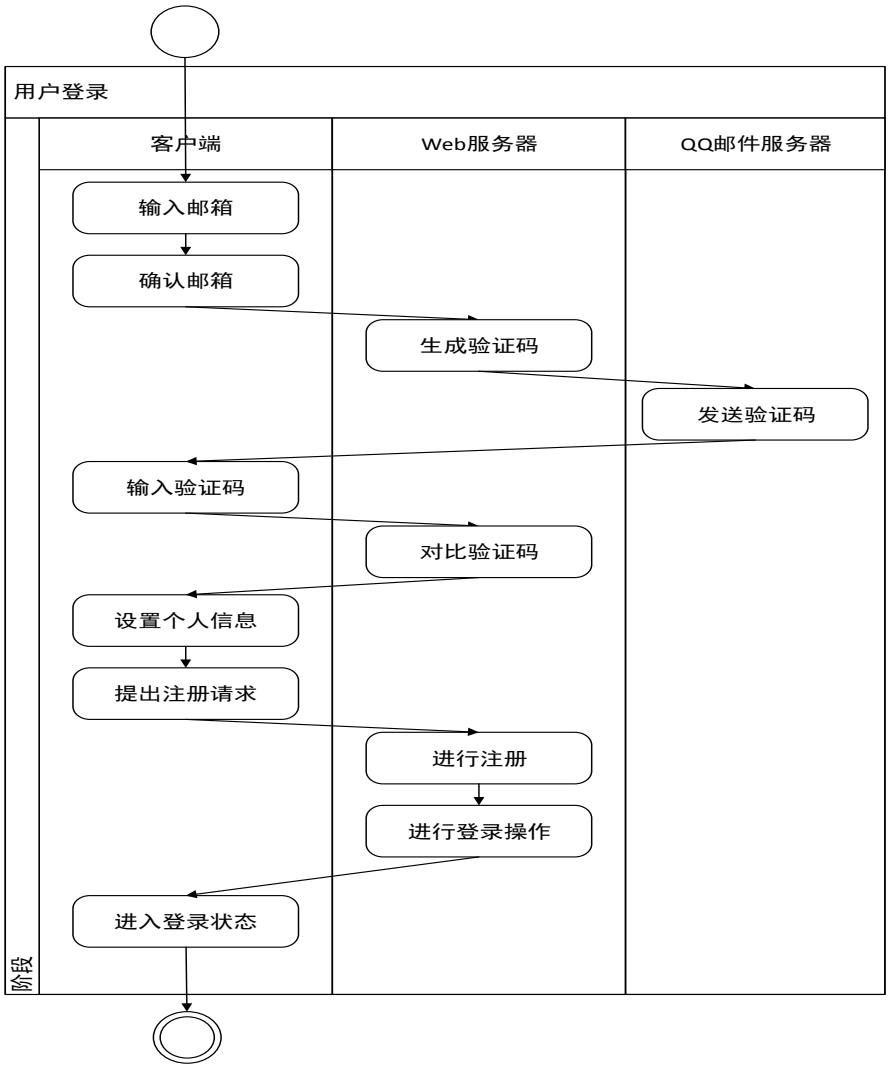


图 4.2 用户注册泳道图

对比验证码关键代码如下：

```
new AsyncTask<Void, Void, Integer>() {
    @Override
    protected Integer doInBackground(Void... voids) {
    if (StringUtil.isEmpty(verifyCode)||!StringUtil.isNumeric(verifyCode)
|| (StringUtil.length(verifyCode) != 6)) { //验证码格式不对}
post(URL_SEND_VERIFY_CODE).tag(this).params(post_check_verify_code,
verifyCode).execute(new JSONCallback<CommonJSON>() {
    @Override
     public void onSuccess(CommonJSON o, Call call, Response response) {
       if (o.getCode() == SUCCESS) {//验证码对比正确
        } else { //验证码对比失败}}
    @Override
     public void onError(Call call, Response response, Exception e) { //系
     统错误} });
        return -1;
    }
    @Override
    protected void onPostExecute(Integer integer) { ......}} }.execute();
```

4.2.2. 账号登录

4.2.2.1. 账号登录概述

用户登录主要出现在三种场景，分别是用户注册完成后，就会直接登录进去、用户在同一设备上一次登录后，下次会自动使用 TOKEN 登录、用户在注销后或在新设备上使用本软件时，需要使用用户邮箱、密码进行登录。

4.2.2.2. 账号登录具体过程

用户在注册成功后，服务器会生成一个 32 位字符串 TOKEN 作为用户的全局唯一标记，这个标记被保存在用户信息表中，然后再保存在服务器的 SESSION 中，之后把 TOKEN 返回给客户端，客户端把 TOKEN 保存到 SharePreference 中，之后进入登

录状态，后面的客户端每次请求服务器都要携带这个 TOKEN 作为凭证，服务器先在 SESSION 中寻找并比较用户的 TOKEN，如果 SESSION 中找不到，就根据用户邮箱去数据库中比较，如果用户提交的 TOKEN 和数据库中该用户的 TOKEN 一致，则允许访问，否则拒绝访问。之后用户打开软件的时候，程序都会查询 SharePreference 获得 TOKEN，并且和服务器端的 TOKEN 进行比较，如果一致，自动登录成功，否则需要用户输入用户邮箱、密码进行登录。当用户注销账户或者在一台新的设备上登录时，由于 SharePreference 中找不到 TOKEN，需要用户输入用户邮箱、密码进行登录。关键代码略。

4.2.3. 账号注销

4.2.3.1. 账号注销概述

用户可以通过账号注销，退出登录状态，这样可以防止其他人访问该设备时候越权查看并使用自己的账户，从而更好地保障账户安全。

4.2.3.2. 账号注销具体过程

用户点击注销按钮，程序会删除保存在 SharePreference 中的 TOKEN，并且跳转到登录页面。下次启动软件的时候，由于程序在 SharePreference 中找不到 TOKEN，不能使用 TOKEN 进行自动登录，需要用户输入用户邮箱、密码进行登录。

账户注销关键代码如下:

```
new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int i) {
                if (!SharedPrefUtil.getInstance()
                        .deleteData(share_token)) {//注销失败
                } else { //注销成功
                    if (MainActivity.instance !=
                   null) {
                        MainActivity.instance.finish();
                    }
                   //跳转到登录页面
                }
 } })
```

4.2.4. 重置密码

4.2.4.1. 重置密码概述

如果用户因忘记了自己的登录密码而不能登录系统，可以通过邮箱验证，然后重置登录密码。

4.2.4.2. 重置密码具体过程

该过程和注册过程相似，也是通过邮箱接受验证码，填写正确的验证码后可以设置新的登录密码，重置之后会跳到登录界面。关键代码略。

4.3. 设置模块设计

4.3.1. 设置头像

4.3.1.1. 设置头像概述

用户在注册时候可以添加用户头像，登录后可以修改用户头像，可以通过拍照或从相册中选择图片来设置。

4.3.1.2. 设置头像具体过程

用户点击按钮，弹出自定义菜单，包括照相机、相册两个菜单项，用户选择其中一个，通过 Intent 打开相机或相册，拍照或选择后再通过 Intent 获取系统裁剪功能，对图片进行裁剪，裁剪成 200*200 的方图，然后进行上传。服务器保存图片文件后，把图片路径保存在用户信息表，之后把图片路径返回给客户端并显示头像。

设置头像关键代码如下:

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (resultCode != Activity.RESULT_OK)
        return;
    switch (requestCode) {
        case SettingUtil.ALBUM_REQUEST_CODE:
            ImageUtil.startCrop(this, data.getData(), cropUri, 200, 200);
            break;
```

```
case SettingUtil.CAMERA_REQUEST_CODE:
    File picture = new File(baseDir,
            IMG_PATH_FOR_CAMERA);//拍照后保存的路径
    ImageUtil.startCrop(this, Uri.fromFile(picture),
            cropUri, 200, 200);
    break;
case CROP_REQUEST_CODE:
  //上传头像} }
```

4.3.2. 设置姓名

4.3.2.1. 设置姓名概述

用户在注册时候必须添加用户姓名，登录后可以修改用户姓名，姓名作为系统中用户的易读性标识。

4.3.2.2. 设置姓名具体过程

采用弹出输入框的方式，用户编辑自己的姓名后，点击保存即可。点击保存后，系统会对用户的输入进行检查，姓名必须非空，可以包括汉字、英文字母、数字（不能数字开头），如果是更改姓名，姓或名必须有改动。检查通过后发送请求到服务器，服务器对用户信息表进行操作。关键代码略。

4.3.3. 设置密码

4.3.3.1. 设置密码概述

用户在注册时候必须添加登录密码，登录后可以修改登录密码。

4.3.3.2. 设置密码具体过程

注册时候必须设置密码，密码需要满足长度为 12 位，可以包括字母、数字、下划线，必须以字母开头。登录后修改密码，需要输入旧的密码、新的密码、确认密码，点击保存后，进行合法检查，除了满足正则表达式的要求，还要求新的密码和旧的密码不一样，新的密码和确认密码一样。检查通过后，对密码进行第一次 MD5 加密，发送请求到服务器，服务器对密码进行第二次 MD5 加密，查询用户信息表，比较数据表中该用户的密码和用户提交的旧密码，如果一致，更新数据库中该用户的密码为用户提交的新密码。关键代码略。

4.3.4. 设置会议偏好

4.3.4.1. 设置会议偏好概述

系统有一套默认的会议偏好，用户可以对其进行修改，以满足自己的要求。会议偏好是用户安排会议时候一些参数的系统默认值，设置符合自己要求的会议偏好，可以简化安排会议的操作。

4.3.4.2. 设置会议偏好具体过程

其中会议主题、入会密码采用输入框 EditText，加会者默认能否绘画、加会者默认能否发言、是否添加至日历项采用滑动按钮 SwitchButton，会议主题需要满足非空，只允许汉字、英文字母、数字,且不能以数字开头，入会密码需要满足 8 位,只允许英文字母、数字、下划线,且以字母开头。用户点击保存，进行合法性检查，检查通过后，把参数保存到 SharePreference 中。关键代码略。

4.4. 会议管理模块设计

4.4.1. 安排会议

4.4.1.1. 安排会议概述

用户可以制定会议计划，需要设置会议的主题、入会密码、预期开始时间、预期结束时间、加会者默认能否使用白板绘画、加会者默认能否使用会内群聊、是否添加到日历事件提醒，会议计划会保存到服务器数据库。

4.4.1.2. 安排会议具体过程

安排会议页面默认加载用户的会议偏好设置参数，用户可以对其进行修改。点击会议日期弹出 DatePickerDialog 供用户选择，点击会议开始时间、结束时间弹出 TimePickerDialog 供用户选择，点击保存，首先进行合法性检查，要求会议日期不早于今天，会议开始时间不早于此时，如果会议结束时间在会议开始时间之前，假定会议结束于开始日期的后一天，检查会议主题和入会密码是否满足要求，如果用户选中了添加到日历项，则系统生成会议信息文本，并添加到日历提醒。检查通过后，发送请求给服务器，服务器生成会议号，连同请求中的会议信息一并保存到会议信息表中，然后返回会议号给客户端。

安排会议关键代码如下:

```
ContentResolver cr = context.getContentResolver();
ContentValues values = new ContentValues();
//添加具体日历事件参数到 values
try {
    Uri uri = cr.insert(CalendarContract.Events.CONTENT_URI, values);
    long mEventID = Long.parseLong(uri.getLastPathSegment());
    // 添加提前 15 分钟提醒
    context.getContentResolver()
            .insert(CalendarContract.Reminders.CONTENT_URI, values2);
    return mEventID;
} catch (SecurityException e) {
    return -1;
}
```

4.4.2. 查看会议

4.4.2.1. 查看会议概述

用户可以查看自己安排的会议列表，点击列表项，进入会议详细信息；点击列表项中的开始按钮，可以开始会议。

4.4.2.2. 查看会议具体过程

页面创建时，发送请求到服务器，服务器查询会议信息表中主持人为该用户的会议，并且该会议处于安排了但没有召开也没有过期的状态，将满足以上要求的会议信息按照会议 ID 的倒序进行分页，把客户端请求的页码的会议信息以 JSON 的数据格式返回给客户端。客户端接收后，解析 JSON 为对象，添加到 List 中，加载到 ListView。

查看会议页面通过实现 SwipeRefreshLayout 的 OnRefreshListener 和 OnLoadListener 接口，可以监听用户的下拉事件，从而实现下拉刷新。用户也可以点击刷新按钮进行刷新。

通过给 ListView 设置 ItemClickListener，监听到用户点击 Item，跳转到会议详细信息页面，在会议详细信息页面，用户可以开始会议、添加会议到日历、添加受邀者、编辑会议、删除会议。

通过在 Item 中添加开始按钮并进行监听，实现点击开始按钮，直接开始会议。

关键代码略。

4.4.3. 加入会议

4.4.3.1. 加入会议概述

用户有两种方式加入会议，分别是通过会议号、密码加入会议和点击联系人的加会邀请通知加入会议。

4.4.3.2. 加入会议具体过程

加入会议包括两个过程：入会登记检查和加载会议。入会登记检查，需要在加会页面发送请求给服务器，请求内容包括会议号、入会密码、用户邮箱等，如果通过输入会议号、密码来加入会议，则请求内容直接从输入框 EditText 中获得，如果通过点击联系人的加会邀请通知来加入会议，则请求内容从通知的 PendingIntent 中获得，服务器的入会登记检查过程如下：

1) 判断该会议号的会议是否存在，不存在则回复客户端，存在则继续。
2) 判断该会议号的会议是否正在进行，不是则回复客户端，是则继续。
3) 登记用户入会信息到用户参加会议信息表。失败则回复客户端，成功则继续。
4) 返回会议信息给客户端，包括主持人邮箱、加会者默认能否绘画、加会者默认能否发言。

客户端收到服务器的返回信息后，跳转到会议页面，初始化界面，通过 WebView 加载白板网页，并且建立和 GatewayWorker 的 WebSocket 连接，通过会议号把该连接加入到 GatewayWorker 的 WebSocket 连接分组，后面就可以通过会议号来发送消息给会议组中的所有参与者。把用户邮箱和 WebSocket 连接绑定，这样就可以通过邮箱来发送消息给指定会议参与者。建立连接后，同步参与者列表。

加入会议关键代码如下:

```
public static function onMessage($client_id, $message){
// 根据类型执行不同的业务
switch ($message_data['type']) {
    case 'login':
        // 判断是否有会议号
        if (!isset($message_data['room_id'])) {
            throw new \Exception("\$message_data['room_id'] not set.
client_ip:{$_SERVER['REMOTE_ADDR']} \$message:$message");
```

```
        }
        // 把会议号、用户姓名、加会类型等信息放到 SESSION 中
        //绑定 clientid 和会议号
        Gateway::joinGroup($client_id, $room_id);
        //绑定 clientid 和用户邮箱
        Gateway::bindUid($client_id, $client_email);
        // 转播进会消息给当前会议的所有客户端,其他人的参与者列表增加该用户
        Gateway::sendToGroup($room_id, $message);
        // 获取会议内之前所有用户的列表
        $clients_list = Gateway::getClientSessionsByGroup($room_id);
        $members_list = array();
        foreach ($clients_list as $tmp_client_id => $item) {
            array_push($members_list, $item);
        }
        // 同步全部参与者列表到该用户
        $member_info = array('type' => 'all_members', 'client_email' => $client_email, 'client_list' => $members_list);
        Gateway::sendToCurrentClient(JSON_encode($member_info));
        return;
```

4.4.4. 会议邀请

4.4.4.1. 会议邀请概述

用户安排会议后或者进入会议后，可以通过会议邀请来邀请其他人加入这个会议。会议邀请包括会外会议邀请和会内会议邀请。会外会议邀请包括发邮件、发短信、复制到剪贴板三种，会内会议邀请在此基础上，增加选择联系人进行邀请。

4.4.4.2. 会议邀请具体过程

邮箱邀请是通过用户手机上安装的邮箱客户端发邮件进行邀请，邮件内容是系统生成的，收件人需要用户自己填写；短信邀请是通过用户手机上的短信应用发短信进行邀请，内容是系统生成的，收信人需要用户自己填写；复制到剪贴板是通过获取 CLIPBOARD_SERVICE 并设置其内容实现的；选择联系人邀请，用户可以浏览联系人列

表，并且勾选需要邀请的联系人，只有用户选择了联系人后，才能看到邀请按钮，点击邀请按钮，发送请求（内容包括用户邮箱、想邀请的联系人的邮箱列表）给服务器，服务器接收到请求后，解析请求内容，通过极光推送服务器推送加会邀请通知到用户想邀请的联系人的设备。

客户端通过 BroadcastReceiver 监听极光推送的通知，判断如果是加会邀请，则构造 NotificationManager，并设置其 PendingIntent 内容（包括会议号、密码、点击事件），然后显示通知。用户点击该通知后，会跳转到加会页面，程序获取 PendingIntent 的内容并自动进行加会操作。关键代码略。

4.4.5. 白板

4.4.5.1. 白板概述

白板功能包括画笔涂鸦、绘制简单几何图形（圆、方框、箭头、直线等）、添加文字、橡皮擦、撤销、清空等，还可以对白板进行屏幕录制，用户在白板上的绘画等操作会实时同步到会议内的其他人。

4.4.5.2. 白板具体过程

白板绘画部分采用 React 编写，关键函数如下：

1) init(element, options={})：初始化白板

   element:与白板绑定的 html 节点；

   options:设置选项，包括：imageURLPrefix（图片文件的路径，最后不加斜杠）、imageSize（图片大小，例如{width: 500, height: null}）、primaryColor（开始是画笔的颜色，默认黑色）、secondaryColor（开始时的填充色，默认白色）、backgroundColor（开始时的背景色，默认透明）、snapshot（初始时的白板数据）、toolbarPosition（工具栏的位置）、tools（工具列表）、strokeWidths（画笔粗细值选择列表，默认是[1, 2, 5, 10, 20, 30]）、defaultStrokeWidth（开始时的画笔粗细，默认 5）。

2) on('drawingChange', callback)：设置白板内容变化监听器

   “drawingChange”：监听的事件名，指的是白板内容的改变，不包括工具中画笔颜色和填充色的变化；

   Callback：处理事件的回调函数。

3) setPan(x, y)：把白板的内容的移动到指定位置（x,y）。

4) setZoom(zoom)：把白板的内容缩放到指定值。

5) setColor(colorName, colorValue)：设置某种颜色的 CSS 色值

colorName ：'background', 'primary', 或'secondary'；

colorValue ：CSS 色值。

6) getColor(colorName)：获取指定名称颜色的 CSS 色值

colorName ：'background', 'primary', 或'secondary'。

7) getPixel(x, y)：获取指定位置（x,y）的像素的 CSS 色值。

8) clear()：清空白板内容。

9) undo()：撤销上一步。

10) redo()：重做上一步。

11) getSnapshot(keys=['shapes', 'imageSize', 'colors', 'position', 'scale', 'backgroundShapes'])：获取当前白板的 JSON 对象，默认包括对象全部内容，如果不需要这么多，可以通过 keys 来指定需要获取的信息类别。

12) loadSnapshot(snapshot)：加载白板 JSON 对象 snapshot 到白板。

13) teardown()：销毁白板。

白板通信部分是首先通过 WebView 加载服务器端的白板网页，并且建立白板网页和 GatewayWorker 之间的 WebSocket 连接，之后的白板同步都是通过 WebSocket 连接进行发送。白板通信架构如图 4.3 所示

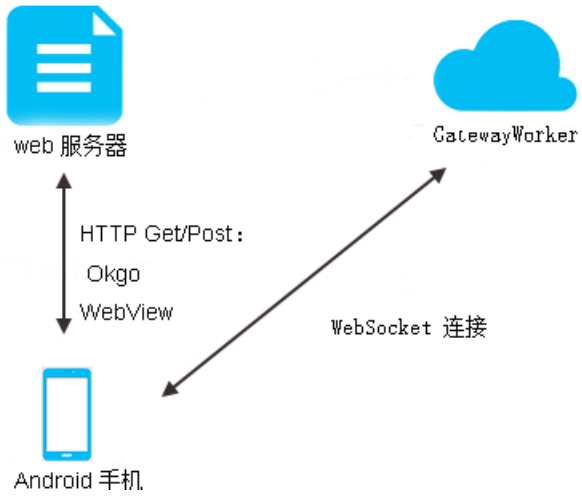


图 4.3　白板通信架构图

用户进入会议后，首先需要向主持人请求白板绘画记录，通过 WebSocket 和 GatewayWorker 转发完成。初始化白板绘画记录后，还需要向主持人请求共享资源，如果此时主持人没有在共享，返回 nothing，如果主持人在共享，返回共享资源（网页或图片）的截图的 Base64 编码字符串，然后对其进行解码，作为白板的背景图片。初始化全部完成后，用户可以进行绘画了。白板的 on('drawingChange', callback) 会监听白板的内容改动，每次变动后都会调用 getSnapshot 方法获取白板对象，并将其导出为 JSON 数据格式，然后通过 WebSocket 和 GatewayWorker 转发给其他会议参与者。其他参与者的白板接收到消息后，会调用 loadSnapshot，重新加载白板内容，从而实现白板的实时同步。

白板同步关键代码如下:

```
function onmessage(e) {
var data = eval("(" + e.data + ")");
switch (data['type']) {
    case 'say'://接收白板数据
            say(data['from_client_id'],          data['from_client_name'],
         data['content'], data['time']); break;
    case 'sync'://发送 base64 给 Native 加会者
        window.board.syncContent(data['sync_pic']); break;
    case 'cancle_sync'://广播主持人取消共享的消息给 Native 加会者
        window.board.cancleSync();break;
    case 'getInitCanvasData'://主持人收到加会者请求 白板数据的请求
        ..... break;
    case 'CanvasData'://接收到主持人传来的白板初始数据
        //加载初始化数据到白板
        break;
    case 'getInitShareData':
        if (check_in_type == 2) {
            //调用主持人 native 函数
            window.board.getSharePic(data['from_client_email']);
        }
```

```
        break;
    case 'ShareData'://接收到主持人传来的白板初始数据
        if (check_in_type == 1) {
            //调用加会者 native 函数
            window.board.initShareContent(data['content']);
            getInitChatData();//请求聊天数据
        }
        break;
}}
```

白板屏幕录制过程如下：

1) 申请权限：需要申请 2 个权限，即 android.permission.RECORD_AUDIO（录屏）和 android.permission.WRITE_EXTERNAL_STORAGE（保存录像），且如果 targetSdkVersion 是 23 以上，还要进行动态权限申请。
2) 创建一个 service 负责创建屏幕镜像、保存录像、停止录像、提示保存路径等操作。
3) 设置服务连接与断开连接的回调函数，绑定 Service 和会议页面。
4) 获取 MediaProjectionManager。
5) 构造捕获屏幕的 Intent。
6) 获取 MediaProjection，并使用 Service 开始录屏，录像保存在本地。
7) 记录当前状态为正在录制，如果用户再次点击悬浮按钮，弹出菜单会有停止录制菜单项，点击，停止录制，并提醒用户录像保存的路径。

4.4.6. 群聊

4.4.6.1. 群聊概述

用户可以发送文字、GIF 表情、图片、语音，可以看到消息发送者的头像、姓名，可以看到消息时间，点击语音消息，会播放，点击图片消息，会全屏显示，然后长按会保存该图片到本地相册。群聊消息会实时同步给会议中的每一个参与者。

4.4.6.2. 群聊具体过程

首先，群聊消息同步和白板同步的通信架构是一样的。文字和表情消息实质上都是文本消息，只需要先进行必要的加密（这里采用的是 AES 加密），然后使用刚刚进入会议就建立好的 WebSocket 进行通信，过程如下:

用户刚进入会议时，首先向主持人请求聊天记录信息，通过 WebSocket 和 GatewayWorker 转发完成。初始化聊天记录后，用户就可以发送消息了。

聊天页面通过 EventBus 把聊天消息传给会议页面，在会议页面中通过白板 WebView 调用 JavaScript 函数，JavaScript 函数通过 WebSocket 发送消息给 Socket 服务器，Socket 服务器转发消息到会议内的所有人，白板网页的 onMessage 方法接收到消息，通过 window.board.xxx()传递数据给 Android，会议页面通过 EventBus 传递聊天数据到聊天页面，加载并显示新收到的聊天消息。

但是对于图片和语音消息就需要先把聊天文件上传到 Web 服务器，然后返回文件地址给消息发送方，消息发送方在通过上面的过程发送图片或语音消息，同步到会议内的其他参与者后，其他参与者根据消息类型，通过文件地址获取服务器上的聊天文件，并进行显示。关键代码略。

4.4.7. 控制加会者权限

4.4.7.1. 控制加会者权限概述

用户作为主持人，在会议中，可以点击加会者列表项的“话筒”按钮来控制他能否发送群聊信息，但是加会者一直可以收到看到群聊信息；点击“画笔”按钮，控制他能否使用白板进行绘画，但是用户一直可以收到看到白板的内容；点击列表项，弹出“踢人”确认框，确认则把他踢出会议。

4.4.7.2. 控制加会者权限具体过程

主持人可以在会议参与者列表中进行权限管理操作。会议参与者列表开始时根据会议设置显示加会者当前权限的状态，绿色的“画笔”代表可以使用白板绘画，红色的“画笔”代表不可以使用白板绘画，绿色的“话筒”代表可以发送群聊消息，红色的“话筒”代表不可以发送群聊消息。

主持人点击“画笔”后，该加会者的绘画权限发生改变，通过 javascript:alterDrawPermission(to_client_email, is_drawable) 发送 "alter_draw_permission"消息给 Socket 服务器，Socket 服务器转发给指定邮箱的加会者，该加会者的白板网页的 onMessage 方法接收到消息，判断“is_drawable”参数，如果为 True，销毁之前的白板 Canvas，新建一个可绘画的白板 Canvas；如果为 False，销毁之前的白板 Canvas，新建一个不可绘画只能显示的白板 Canvas（把新建可绘画白板的 init 的 options 列表中的 tools 参数中除了 LC.tools.Pan(缩放按钮）和 LC.tools.SelectShape（选择移动按钮）的工具都去掉)。然后设置内容改

变监听并重新向主持人请求数据（白板数据+共享资源），然后加载请求到的数据到白板。

主持人点击“话筒”后，该加会者的群聊权限发生了，通过javascript:alterDrawPermission(to_client_email, is_talkable) 发送"alter_talk_permission"消息给 Socket 服务器，Socket 服务器转发，该加会者的白板网页接收到消息，通过window.board.alterTalkPermission(data['is_talkable'])操作 Android 的界面，alterTalkPermission 方法对用户进行权限改变提示，并通过 EventBus 发送消息通知聊天页面，聊天页面的 listenTalkPermissionChange 函数接受消息，并根据istalkable 的值设置群聊输入栏的可见性,如果 istalkable 为 False,隐藏输入栏，这样用户就没法发送群聊消息了。

如果主持人点击一个加会者所在的列表项空白处，会弹出“踢人”确认框，提示是否确认把该加会者踢出会议，点击“确定”，则会通过 javascript:kickout（to_email）发送"kickout"消息给 Socket 服务器，Socket 服务器转发，该加会者的白板网页接收到消息，通过 window.board.kickout()改变 Android 的界面（弹出“你被主持人踢出会议”的提示框，并且只有当用户点击确定后才会消失，点击确认，会议页面关闭，离开会议），紧接着白板网页还会通过 syncLeaveMeeting(name)发送"leaveMeeting"消息通知会议内的其他人该用户离会，其他人接收到消息，并从自己的会议参与者列表中移除该用户。

4.4.8. 删除会议

4.4.8.1. 删除会议概述

用户可以在查看会议详细信息时删除该会议。

4.4.8.2. 删除会议具体过程

用户点击会议详细信息页面的删除会议按钮，发送请求给服务器，服务器根据请求中的会议号，获取会议信息表中该会议号所对的会议信息，比较请求中的用户邮箱和该会议主持人的邮箱是否一致，如果一致，则从会议信息表中删除该会议，并且返回成功信号给客户端，销毁该页面。关键代码略。

4.5. 联系人管理模块设计

联系人管理的通信机制不同于白板和群聊。联系人管理通信架构图如图 4.4 所示。

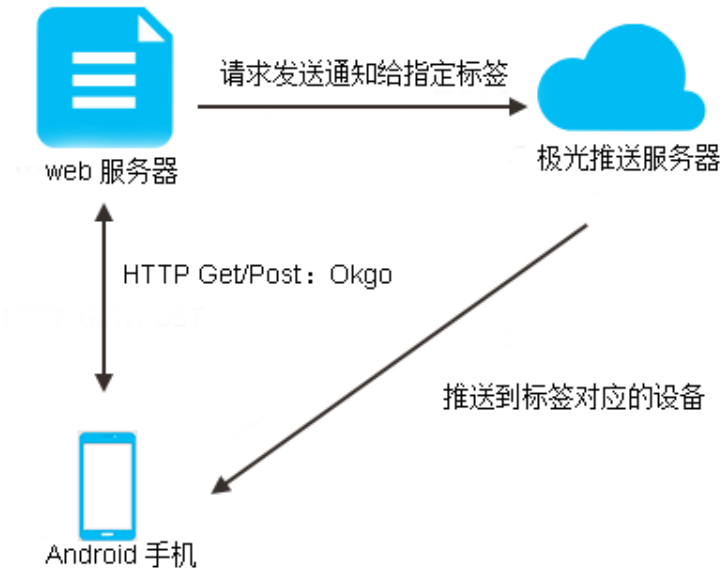


图 4.4　联系人管理通信架构图

为了实现推送，首先需要配置极光推送 SDK,包括 Android 和 PHP 两部分。因为需要给使用本软件的手机设置标签才能实现推送消息到用户，所以采用用户邮箱作为标签，在用户注册完毕后和每次登录时，都重新设置设备标签为用户邮箱。除此之外，还要定义一个 BroadcaseReceiver，用于接受推送消息。

4.5.1. 添加联系人

4.5.1.1. 添加联系人概述

用户可以通过邮箱来申请添加联系人，成功添加联系人后，可以在联系人列表中看到自己所有的联系人的信息，也可以更加方便地通过选择联系人来发布加会邀请。

4.5.1.2. 添加联系人具体过程

用户点击添加联系人，在输入框中填写合法邮箱，且不为自己当前账户邮箱，点击确定，发送请求给服务器，服务器查询联系人信息表，如果目前两个用户不是联系人关系，且联系人信息表中不存在这两个用户之间添加联系人的记录，则新增一行并把 response_status 置为 1，然后推送消息给被请求的用户，如果推送成功，回复

请求者“请求发送成功”，并返回被请求联系人的用户信息。收到“请求发送成功”消息后，请求者添加消息到本地数据库 Msg 表（此时消息的 status 为 0，表示初始态）。随后被请求用户的 BroadcaseReceiver 收到通知消息，解析消息数据，判断消息类型，如果为"requestAddFriend"，在 Msg 数据表中新增消息记录（此时消息的 status 状态为 0，表示初始态），并新建通知栏通知提示用户有“添加联系人请求”，系统未读消息数加一，通过 EventBus 发送消息通知系统主页面更新系统未读消息提示。用户看到通知栏提示，点击通知栏会进入到系统主页面，点击“系统消息”，进入查看，系统消息界面是由一个 ListView 实现的，每一个 Item 显示联系人添加的消息，点击后弹出“删除”菜单，点击即可删除该列表项，并从数据库中删除该消息记录。

如果被申请者点击“接受”，发送请求到服务器，更新联系人信息表相应记录的 response_status 为 3，然后推送“接受”消息给请求者，推送成功后，更新本地数据库的 Msg 表相应记录的 status 为 1，表示接受好友申请，然后把新的联系人添加到系统主页面的联系人页面的列表中；请求者的 broadcastReceiver 接收到消息，通过 post_message_data 参数查找本地数据库 Msg 表，找到 tag 为该 post_message_data 的记录，更新 status 字段为 1，表示接受好友申请，创建“XXX 同意了你的添加好友申请”通知提醒用户，把系统未读消息数加一，通过 EventBus 通知系统消息页面更新、通知系统主页面更新系统未读消息提示、通知联系人列表添加该联系人。

如被申请者点击“拒绝”，和点击“同意”的操作过程类似，只是设置数据库中相应字段为预先规定好的“拒绝”对应的常量，并且通知提醒用户“XXX 拒绝了你的添加好友申请”。关键代码略。

4.5.2. 删除联系人

4.5.2.1. 删除联系人概述

用户可以在联系人列表中长按想要删除的联系人所在的列表项进行删除。

4.5.2.2. 删除联系人具体过程

给联系人列表设置 ItemLongClickListener，监听到用户 a 长按操作，弹出删除联系人确认框，点击确认即可删除联系人，发送请求给服务器，服务器根据请求中的用户 a 邮箱、想删除联系人 b 邮箱，查询联系人信息表，将 response_status 置为 4（表示好友关系已删除），然后通过极光推送服务器通知想删除联系人 b 他已经被用

户 a 从联系人列表中删除，想删除联系人 b 的客户端的 broadcastReceiver 接受到消息，发布通知到通知栏，通过 EventBus 通知并更新联系人列表。推送完成后，返回成功信号给用户 a，更新用户 a 的联系人列表。关键代码略。

4.6. 关于软件模块设计

4.6.1. 检查更新

4.6.1.1. 检查更新概述

用户可以手动检查版本更新，如果有新的版本，就会提醒用户进行下载和更新。

4.6.1.2. 检查更新具体过程

在用户安装软件后第一次启动的时候，程序就把版本号信息保存到 SharePreference 中，关于软件页面的版本更新栏显示的版本号就是通过读取 SharePreference 获取到的。用户点击版本更新栏，发送请求给服务器，获取服务器端最新的版本信息（通过查询版本信息表，获取最后一条记录），比较 SharePreference 中的版本号和服务器端的最新版本号，如果需要更新，则弹出提示框，如果是强制更新，则只有点击更新弹出框才会消失。更新后启动软件，程序会更新 SharePreference 中的版本号。

版本更新关键代码如下:

```
OkGo.post(URL_VERSION_UPDATE)
.tag(this)
.params(post_token, token)
.params(post_user_email, email)
.execute(new JSONCallback<UpdateAppJSON>() {
 @Override
 public void onSuccess(UpdateAppJSON o, Call call,
                       Response response) {
    if (o.getCode() == SUCCESS) {
       String server_version = o.getData()
              .getServerVersion();
       //比较服务器最新版本号和当前已安装版本号
```

```
        if (StringUtil.appVersionCompare(server_version,
                current_version) <= 0) {
            MyAppUtil.noneUpdate(mContext);
            return;
        }
        int isForce = o.data.getLastForce();//是否需要强制更新
        String downUrl = URL_UPGRADE + o.data.getUpdateurl();
        String updateinfo = o.data.getUpgradeinfo();//apk 更新详情
        String appName = o.data.getAppname();
        if (isForce == 1) {//强制更新
            MyAppUtil.forceUpdate(mContext, appName, downUrl,
                    updateinfo);
        } else {//非强制更新
            MyAppUtil.normalUpdate(mContext, appName, downUrl,
                    updateinfo);
}
```

4.6.2. 用户反馈

4.6.2.1. 用户反馈概述

用户可以反馈自己的使用感受、建议等，字数要求在 20-150 字，反馈内容会通过服务器以邮件的形式发送给系统管理员。

4.6.2.2. 用户反馈具体过程

用户在输入框中填写反馈信息，要求字数不少于 20 字不多于 150 字，通过给输入框 EditText 设置 TextChanged 监听器，实现实时计算用户还可以输入的字数，并通过文本框 TextView 告诉用户。用户点击发送，进行合法性检查，检查通过后，发送请求给服务器，服务器进行转发，通过 QQ 邮件服务器发送邮件给系统管理员。关键代码略。

4.6.3. 查看隐私保护策略

4.6.3.1. 查看隐私保护策略概述

用户可以查看软件开发者声明的隐私保护策略、免责条款等。

4.6.3.2. 查看隐私保护策略具体过程

隐私保护策略是 HTML 写的一个网页，部署在服务器，这样可以方便地对其内容进行修改而不必要求用户更新整个软件。客户端采用 WebView 对网页进行显示，设置对应的网址加载显示即可。关键代码略。

5. 系统实现与测试

5.1. 系统实现

5.1.1. 功能实现

1） 启动页

启动页为用户打开软件首先看到的页面，下方展示了应用图标和应用名称，启动页后台主要进行一些初始化、TOKEN 自动登录等操作，如果 TOKEN 登录成功，就进入系统首页，否则进入登录页。（见图 5.1）

图 5.1 启动页

图 5.2 注册

图 5.3 设置个人信息

2） 注册

注册页面包括邮箱输入框、获取验证码按钮、验证码输入框、进入下一步的按钮，验证码 2 分钟内有效，用户填写收到的验证码后就可以点击下一步进行后续操作了。（见图 5.2）

3） 设置个人信息

设置个人信息页面包括头像选择、姓名输入框、密码输入框。邮箱验证之后，用户在这里设置完个人信息，点击完成就完成注册了。（见图 5.3）

4） 登录

登录页包括用户名、密码输入框、登录按钮，下方还有注册和重置密码的链接。（见图 5.4）

5） 系统首页

系统首页是系统的入口，它包括三部分，可以通过左右滑动进行跳转，也可以点击底部的按钮进行跳转。三部分包括会议（见图 5.5）、联系人（见图 5.6）、设置（见图 5.7）。

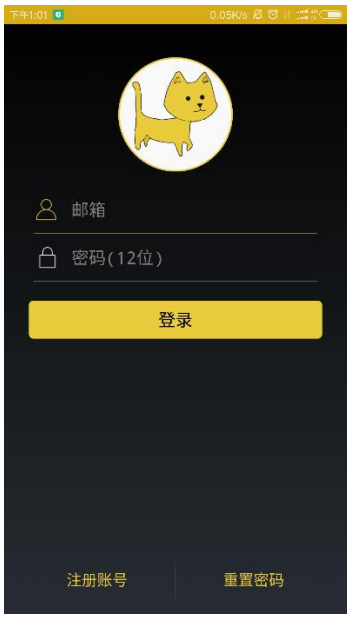


图 5.4 登录

图 5.5 会议

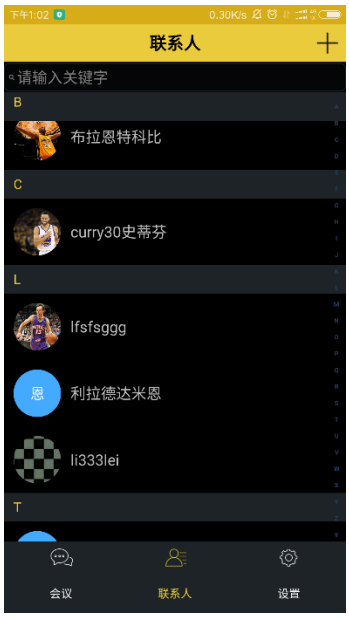


图 5.6 联系人

6） 加入会议

加入会议页面包括会议号输入框、入会密码输入框、加会按钮，用户可以通过这里加入别人主持的会议。（见图 5.8）

7） 安排或主持会议

该页面是系统二级导航页，可以点击召开会议按钮立刻开始一个会议，也可以点击安排会议、我的会议进入相应页面。（见图 5.9）

8） 我的会议

我的会议页面显示我安排过的会议中没开始且没有过期的会议，可以通过下拉刷新，也可以通过点击右上角的刷新按钮进行刷新，点击每一项，可以查看会议详细信息，点击开始按钮，可以开始该会议。（见图 5.10）

9） 安排会议

用户通过安排会议页面，可以设置会议信息，包括主题、开始时间、结束时间、是否添加到日历事件提醒等，并点击保存按钮保存会议安排到服务器。（见图 5.11）

10）会议信息

会议信息页面用于显示已安排好的会议的详细信息，包括主题、会议号、会议时长等，并且可以进行开始会议、添加到日历、添加受邀者、删除会议等操作。（见图 5.12）

图 5.7 设置

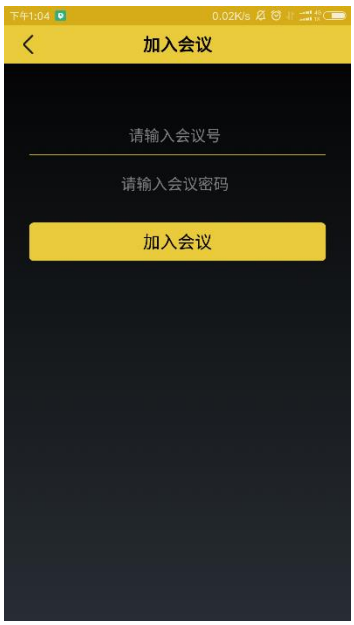


图 5.8 加入会议

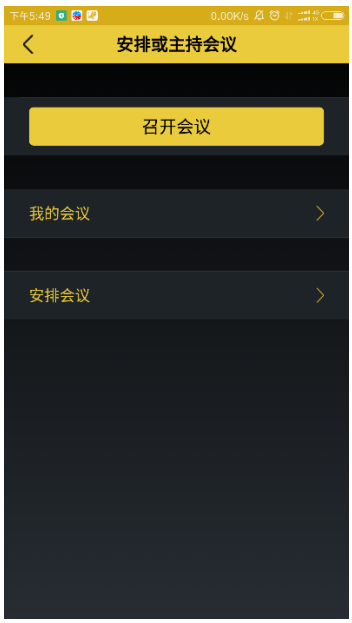
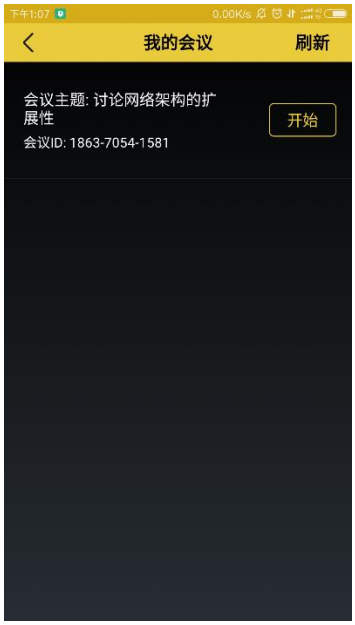


图 5.9 安排或主持会议



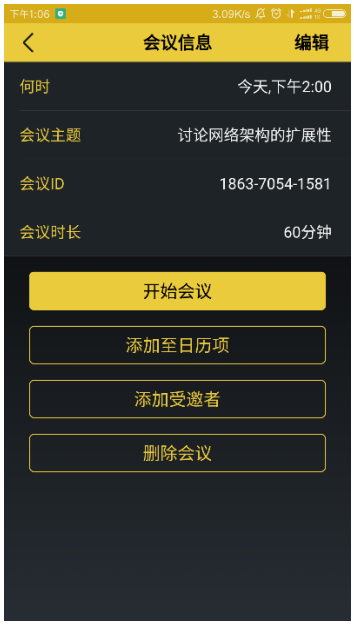


图 5.10　我的会议　　图 5.11　安排会议　　图 5.12　会议信息

11) 会议主页

会议主页为用户进入会议看到的，最上面是会议号和离会按钮，中间是背景图片，采用的是用户头像，侧面有一个悬浮按钮，下面是按钮栏，包括参与者、共享、锁定会议，如果是主持人，三个按钮都可见，如果是加会者，则只能看到参与者按钮。（见图 5.13）

12) 共享白板

共享白板页面包括左侧工具栏、中央舞台区以及一个悬浮按钮，用户可以绘画，录屏。（见图 5.14）

13) 参与者

参与者页面显示会议当前的参与者列表，主持人可以在这里控制加会者的权限，用户可以通过这里进入聊天页面，也可以邀请别人加会。（见图 5.15）

14) 邀请联系人

用户可以浏览自己的联系人列表，勾选其中的联系人，进行推送加会邀请。（见图 5.16）

图 5.13　会议主页

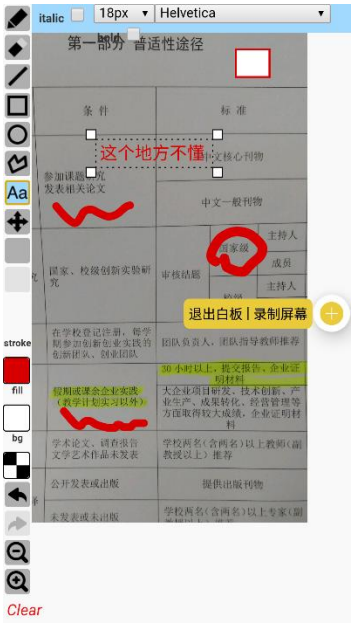


图 5.14　共享白板

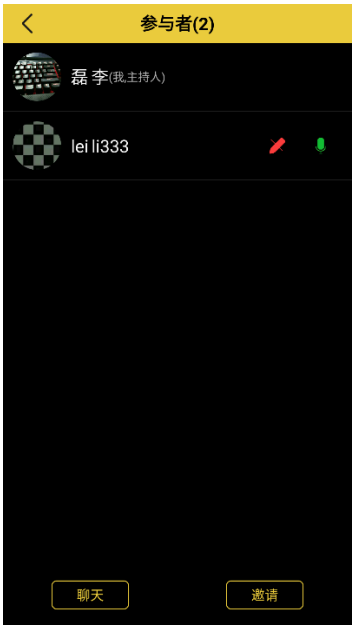


图 5.15　参与者

15) 聊天

会议参与者可以在这里进行聊天，支持文字、GIF 表情、图片、语音。（见图 5.17）

16) 系统消息

系统消息显示用户接受到的消息列表，消息主要是添加联系人的消息，用户可以接受或拒绝添加联系人申请。（见图 5.18）

17) 我的资料

我的资料页面是系统二级导航页，通过这里可以进入到姓名和登录密码更改页面，也可以修改头像、注销。（见图 5.19）

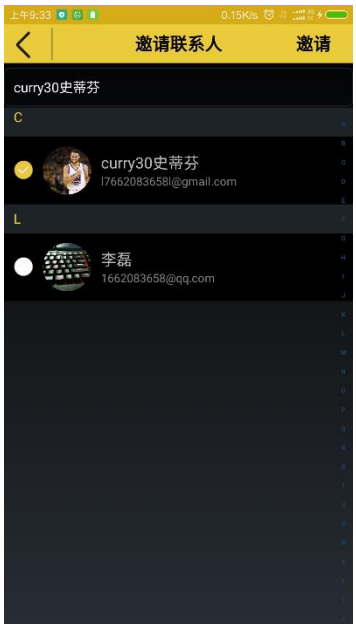


图 5.16　邀请联系人

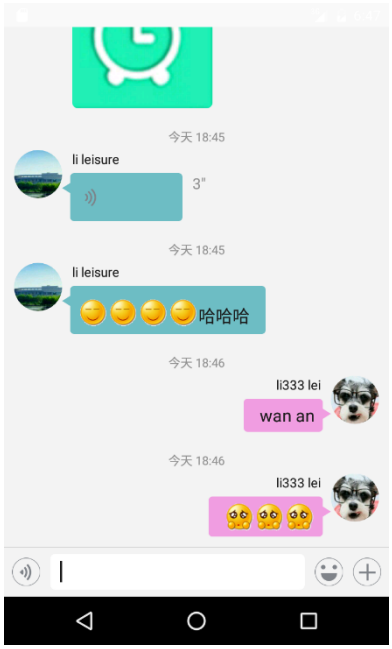


图 5.17　聊天

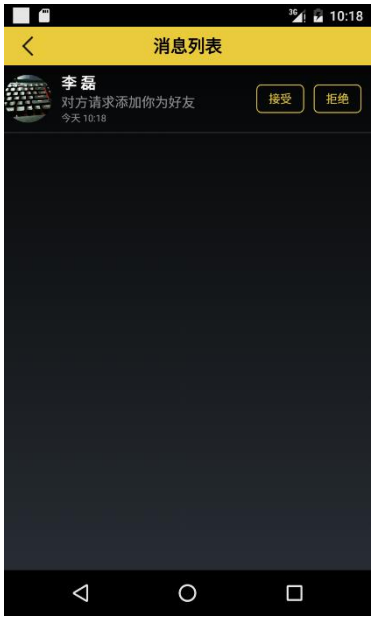


图 5.18　系统消息

图 5.19　我的资料

图 5.20　更改密码

图 5.21　会议设置

18) 更改密码

更改密码页面包括旧密码输入框、新密码输入框、确认密码输入框和保存按钮。（见图 5.20）

19) 会议设置

会议设置页面主要用于设置会议的默认偏好，作为之后安排会议的默认参数。（见图 5.21）

20) 关于小喵白板

关于小喵白板是系统的二级导航页，可以进入发送反馈、查看隐私策略页面，也可以进行版本更新检查。（见图 5.22）

21) 发送反馈

发送反馈页面包括一个输入框，最多输入 150 字，输入完毕，点击发送即可通过服务器发送邮件给系统管理员。（见图 5.23）

22) 隐私策略

用户可以在这里查看软件的隐私保护策略。（见图 5.24）

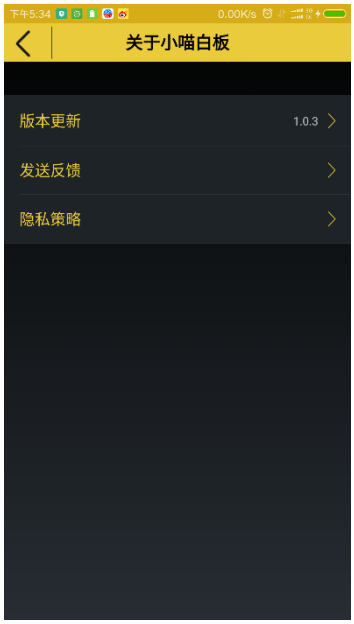
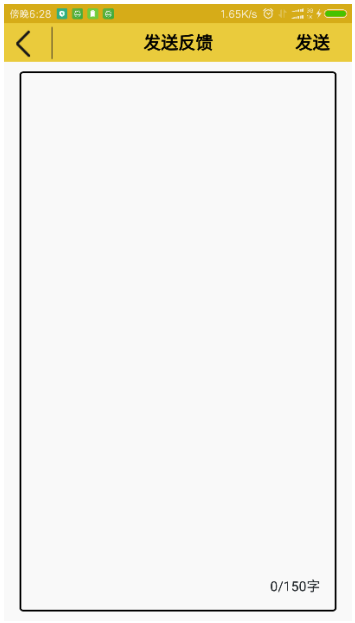


图 5.22 关于小喵白板



5.23 发送反馈

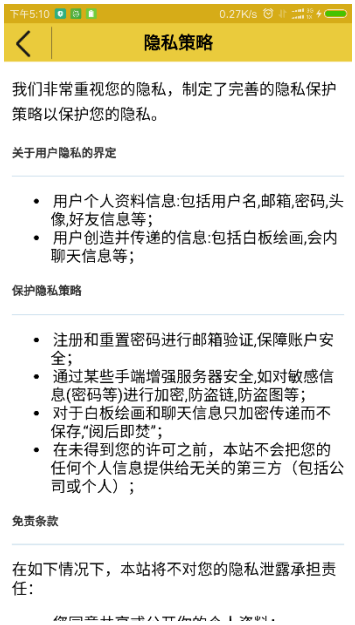


图 5.24 隐私策略

5.1.2. 系统部署

系统开发阶段后，需要将程序部署到真实环境中，才能进行进一步的测试，包括功能测试、性能测试等等。

服务器端程序的部署环境是腾讯云的 Linux 服务器，首先进行 Apache、PHP、MySQL 的安装和配置，之后根据官方文档配置 GatewayWorker，然后通过 FileZilla 与服务器建立 SFTP 连接，上传代码，然后进行相关启动、初始化工作，部署完毕。服务器部署日志截图如图 5.25 所示。

手机端程序的部署可以通过把真机开启 USB 调试后连接 PC 进行调试，也可以把 APK 安装包拷贝到真机，直接安装。

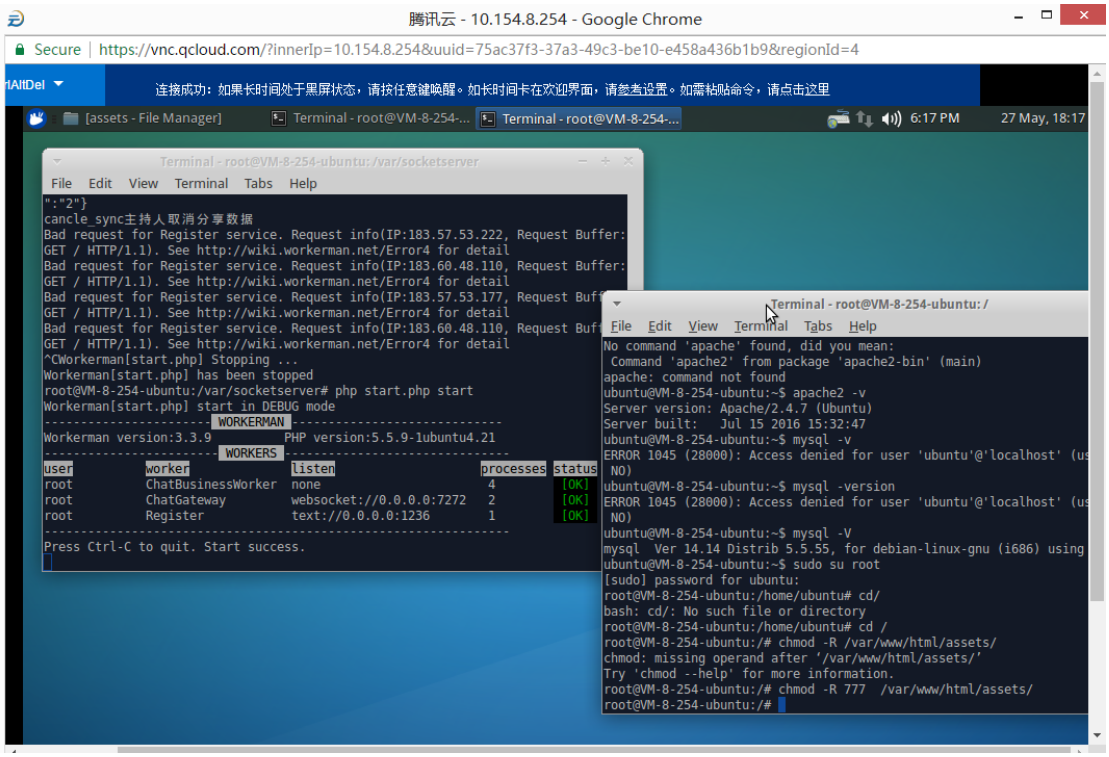


图 5.25　服务器部署日志截图

5.2. 系统测试

5.2.1. 系统功能与兼容性测试

在系统各部分均部署完成后，即开始进行系统的测试工作。这里对系统的各个模块按照粒度从小到大开展测试工作，同时如果发现问题则立即对 bug 进行修复。在这个过程中，设计并执行了许多测试用例，对系统从各个方面进行了充分的测试。如加入会议的功能，测试了各种加会失败情况下，系统的反馈信息及是否会出现崩溃等问题。

另外针对 Android 系统的碎片化问题，我除了使用自己的手机，还借用了同学的各种具有代表性的手机作为部署与测试对象。在每个机子上都跑一遍最核心的功能流程，测试结果如表 5.1 所示，总体情况良好。

表 5.1 兼容性测试结果表

机型	安卓系统版本	测试情况
小米 4	安卓 6.0	聊天页面滑动有些卡顿
三星 S7	安卓 7.1	正常
小米 5	安卓 6.0	正常
华为荣耀 6	安卓 6.0	正常

5.2.2. 性能测试

性能测试方面，主要测试了系统开启速度、占用物理内存、流程消耗、网络请求响应时间速度部分，测试结果如表 5.2 所示。

表 5.2　性能测试结果表

性能指标	测试结果
占用内存	55-65MB
流量消耗	5 人以内会议，群聊部分只发送文本消息情况下，每分钟会议大约消耗流量 100KB
应用启动时间	小于 2S
网络请求响应时间	平均小于 3S

各项指标获取途径有所不同，如占用内存指标的获取是从系统应用运行时数据中直接获取的（如图 5.26），而流量消耗是使用手机管家进行统计的。

图 5.26　应用占用内存图

6. 总结与展望

6.1. 总结

本课题的目标是设计并实现一个运行在 Android 手机上的方便用户随时随地进行团队沟通与协作的的软件。截至目前，本系统已经依据软件工程的方法，经过了需求分析、总体设计、详细设计、实现、测试、部署各各阶段，按照预期，顺利完成。在此过程中遇到了各种各样的问题，例如安卓的系统版本兼容与机型兼容、服务器部署异常等，但通过查阅资料等方式不断地解决问题，最终完成了基于 Android 的实时共享白板。

本系统是一个功能完善的 Android 手机软件，实现了会议管理功能，包括安排会议、查看会议、发布会议邀请、召开会议、加入会议、实时共享会议白板、会议群聊、锁定会议、查看会议参与者、控制加会者权限、删除会议等；实现了联系人管理功能,包括请求添加、拒绝添加、同意添加联系人，以及查看和删除联系人，发布会议邀请时，用户可以选择自己的联系人进行邀请；实现了设置功能，包括设置头像、姓名、密码、会议偏好；实现了账户管理功能，包括注册、登录、注销、重置密码；实现了关于软件功能，包括检查更新、用户反馈、查看隐私保护策略。

通过使用本软件，用户可以随时随地地与自己的伙伴进行白板会议，而不必赶往会议室，也不必坐在 PC 前。通过简单的开会流程、高效的会议交互、必要的会内权限管理，可以有效提高团队远程沟通协作的效率。

当然由于时间和精力的问题，本系统也有很多不足的地方，比如并发冲突的解决、不能一键添加用户手机通讯录中的联系人到系统联系人、可能存在兼容性问题等。日后我一定会逐渐完善本系统。

6.2. 展望

基于 Android 的手机端实时共享白板是解决团队远程沟通协作问题的一种方案，但是随着新需求的提出与新技术的产生，可能有越来越好的方式来解决团队沟通协作问题。一个软件要想一直被大众喜欢，就要紧跟时代步伐，洞察受众的期望，通过

不断满足客户的需求，占据更广阔的市场，赢得一个好的行业口碑。而这个毕设的完成，对我来说无疑是一个很好的开端，我也必将更加努力，争取可以在这个互联网的时代用实力证明自己。

致谢

毕业在即，写好论文是一项非常重要的工作。在写论文之前，我对论文的内容要求和版面要求都不是很懂，是在指导老师张本宏老师的辛勤指导下才一步步明白的。其实早在毕设选题的时候，张老师就开始对我进行指导了，他仔细斟酌了我提交的几个选题，根据多年经验，帮我确定了这个课题，之后在我进行系统的设计与实现的过程中，他多次督促我的开发进度，帮我寻找问题的解决方案，提供极其重要的建议。在此，我想对张老师表达最诚挚的感谢与敬意。

大学四年，光阴荏苒，陪伴我的不只是张老师，还有好多位老师，他们尽心竭力地教导我，把我从软件开发的门外汉变成了一个理论基础扎实、实践能力突出的程序员，让我有机会继续读研深造。除了知识，他们还教会了我为人处世的道理，树立了正确的理想和追求。

还要感谢大四实习时所在的公司，让我对企业级的软件开发有了初步的了解，对之前学习的专业知识有了更加深刻的认识。

最后，衷心感谢在百忙中抽出宝贵时间对本论文进行评阅与审查的老师们！

[参考文献]

[1] Roger S.Pressman，郑仁杰，马素. 软件工程实践者的研究方法（第七版）[M].北京:机械工业出版社，2011.
[2] Zakas Nicholas C.，李松峰，曹力. JavaScript高级编程（第三版）[M]. 北京:人民邮电出版社，2012.
[3] Jason Lengstorf，Phil Leggetter，肖智清. 构建实时Web应用: 基于HTML5 WebSocket、PHP和jQuery[M].北京:机械工业出版社，2013.
[4] 古曼兹，瑞桑斯，简张桂. php5权威编程[M].北京:电子工业出版社，2007.
[5] 王珊，萨师煊.数据库系统概述[M].北京:高等教育出版社,2014.
[6] 菲利普斯，斯图尔特. Android编程权威指南（第二版）[M].北京:人民邮电出版社，2016.
[7] 特南鲍姆，韦瑟罗尔，严伟，潘爱民. 计算机网络（第五版）[M].北京:清华大学出版社，2012.
[8] Matt Zandstra，陈浩. 深入PHP: 面向对象、模式与实践（第三版）[M]. 北京: 人民邮电出版社，2011.
[9] 唐汉明，翟振兴，关宝军. 深入浅出MySQL 数据库开发优化与管理维护（第二版）[M].北京:人民邮电出版社，2014.
[10] David Geary. HTML5 Canvas核心技术: 图形、动画与游戏开发[M].北京:机械工业出版社，2013.
[11] 陈文. 深入理解Android网络编程[M].北京:机械工业出版社，2013.
[12] Smith D.，Friesen J.. Android 5.0开发范例代码大全（第四版）[M]. 北京:清华大学出版社，2015.
[13] 陶松，刘雍，韩海玲，周洪林. Ubuntu Linux从入门到精通[M].北京:人民邮电出版社，2014.
[14] 赵振，王顺. Web异步与实时交互 iframe AJAX WebSocket开发实战[M].北京:人民邮电出版社，2016.