



系统开发说明报告

院 系：计算机学院

课程名称：移动智能应用开发

项目名称：Teamical—日程管理 APP

指导老师：李慧

专业：计算机科学与技术

成员：陈烁明 20182131017

宋金龙 20183231027

刘远航 20183231056

陈羨琿 20182131062

目录

.....	1
第一章 产品设计方案	5
1、项目实施可行性.....	5
1.1 行业市场分析.....	5
1.2 同类产品分析.....	5
1.3 自身条件分析.....	5
2、产品定位及目标.....	6
2.1 产品需求定位.....	6
2.2 差异化价值点定位.....	6
2.2.1 用户群体方面	6
2.2.2 使用功能方面	7
2.3 主要用户	7
3、产品内容总策划.....	7
3.1 应用流程规划.....	7
3.2 设计与测试规范	7
3.3 功能模块图	8
3.4 用例建模.....	9
3.5 产品结构图	12
3.6 操作流程图	13
3.6.1 个人用户操作流程图	13
3.6.2 团队用户操作流程图	14
3.7 详细需求.....	14
3.7.1 总体功能需求	14
3.7.2 具体功能需求	18
需求分析	18
用例矩阵.....	19
3.7.2.2 团队日程管理.....	19
模块描述	19
用例矩阵.....	21
3.8.2 系统需求.....	22
3.8.3 运营需求.....	22
3.8.4 安全需求.....	22
3.9 数据库设计	23
3.9.1 ER 图设计	23
3.9.2 CDM 模型.....	24
3.9.3 PDM 模型.....	24
3.10 开发日程表	25
4、技术解决方案	26
4.1 客户端开发.....	26
4.2 界面框架选择.....	26
4.2.1 选择 XUI 界面框架的原因	26

4.2.2 选择其他组件/库的说明	27
4.3 客户端安全	28
4.4 服务端开发	29
4.5 系统架构	29
4.6 项目部署	30
5、推广方案	30
5.1 企业的产品销售策略	30
5.2 产品推广计划	30
6、运用规划书	31
6.1 运营目标	31
6.2 维护目标	31
第二章 产品实现方案	32
1、系统的主要功能	32
1.1 登录/注册	32
1.2 时间提醒/规划	32
1.3 任务分配/查看	32
1.4 时间统计	32
1.5 信息发送	32
2、UI 界面设计	33
2.1 登录界面	33
2.2 主界面	34
2.3 各功能界面	35
2.4 主要界面的操作流程	36
2.4.1 主界面操作流程	36
2.4.2 日程创建界面操作流程	36
.....	36
.....	36
3、关键技术和技术难点	37
3.1 界面框架选择	37
3.2 客户端的开发与安全	37
3.3 服务端开发	38
3.4 系统架构	38
3.5 技术细节	38
3.5.1 框架的基本使用	38
3.5.2 侧边导航栏的实现	40
3.5.3 底部菜单栏的实现	41
3.5.4 日历组件	42
3.5.5 listview 动画	43
3.5.6 二维码扫描	44
3.5.7 团队用户管理界面	45
3.5.8 新增日程界面	47
3.5.9 悬浮按钮	47
3.5.10 携带数据跳转	48
3.5.11 获取管理的团队和加入的团队	49

3.5.12 团队权限判断	50
3.5.13 解散团队	51
4、用户体验记录和分析	52
4.1 用户的体验记录	52
4.2 用户体验的总结分析	54
5、已完成的改进和存在的问题	54
5.1 已完成的改进	54
5.2 仍存在的问题	55
第三章 测试大纲和测试报告	55
1.1 测试结论	55
1.2 性能测试	56
1.3 问题概述	59
1.4 测试机型参考	60
第四章 产品安装和使用说明	61
1.1 产品安装	61
1.2 使用说明	62
1.2.1 客户端运行	62
1.2.2 用户注册	62
1.2.3 用户登录	63
1.2.4 功能界面的使用	64

第一章 产品设计方案

1、项目实施可行性

1.1 行业市场分析

现代的人们正处于一个快速发展的时代，每个人都有着自己的事情工作，面临各种各样的选择。在这种快节奏的工作生活中，人们很容易忘记自己的时间日程安排，团队项目成员之间的沟通协作也可能因成员的时间安排遗忘和任务分配不明确等而效率低下。因此拥有一款可以面向各类需求人群，比如个人普通用户、团队项目负责人和团队项目成员的时间日程管理 APP 可以解决时间安排遗忘，团队成员间沟通协作效率低下等问题，这样用户可以合理地安排日程时间不会遗忘，团队成员可以高效率地沟通协作完成任务项目。

1.2 同类产品分析

APP 类型	服务对象	个人定制	时间规划	时间提醒	任务分配	日志记录
Worktile	较全面	无	良	优	良	优
Ticktick	单一	无	优	优	差	优
日程吧	单一	无	良	良	差	良
时光序	单一	无	优	优	差	优

通过对现有较多人使用的日程管理 APP 的调查了解，相类似的日程管理 APP 虽然某些方面的功能如时间提醒，时间规划等功能有着较好的用户体验，但他们的功能不是特别全面，服务的对象人群较为单一，而我们设计日程管理 APP 功能较为全面，服务人群较广包含个人普通用户和团队型用户，且使用简单，对于某些专业人群还提供了个性化定制需求。

1.3 自身条件分析

我们希望我们设计的这款日程管理 APP 可以为不同类型的用户提供便捷的日程时间管理安排，满足用户的各类需求。

服务对象：服务对象全面，既包含普通个人用户，也可以服务团队负责人及成员。

个人定制：系统提供了一些 API 供专业人员二次开发，从而实现个性化的定制需求。

时间规划：根据用户提供的空闲制定适合个人用户或团队的时间规划安排。

时间提醒：对用户的工作任务安排进行提醒，避免用户遗忘带来不便。

任务分配：对于团队负责人用户可以对团队合作项目进行任务分配，团队成员获取任务信息进行分工合作。

日志记录：用户可以看到任务工作完成进度。

2、产品定位及目标

此款产品致力于提供最方便的日常管理操作，为不同角色的用户打造多功能全方位的产品，把多类角色的日程管理整合成一个日程表。此款产品不仅基本包含市面上同款产品的功能，更有一些差异化价值点。

2.1 产品需求定位

- (1). 便于用户对自己的日程进行安排以及及时获得提醒。
- (2). 满足团队对于团队成员的时间管理规划需求。
- (3). 便于用户查看自己的在工作、学习、娱乐等方面的时间花费。

2.2 差异化价值点定位

2.2.1 用户群体方面

与市面上的同类型时间管理产品相比，此款产品不仅面向个体，同时面向团队。便于团队对其成员进行活动安排的设置和调整。

与市面上的同类型时间管理产品相比，此款产品提供二次开发的接口，便于用户开发个性化功能。

2.2.2 使用功能方面

与市面上的同类型时间管理产品相比，此款产品提供文本自动识别功能，方便用户进行日程设置。

与市面上的同类型时间管理产品相比，此款产品自动进行国内外时间转换，方便国外的活动提醒。

与市面上的同类型时间管理产品相比，在团队使用方面，此款产品可以收集某一次活动的团队成员的空闲时间表并自动统计时间。

与市面上的同类型时间管理产品相比，在团队使用方面，此款产品可以收集成员的完成情况。

2.3 主要用户

- (1). 需要进行日程安排和管理的用户。
- (2). 对于团队成员日程安排有需求团队用户。
- (3). 对于有个性化开发需求的用户。

3、产品内容总策划

3.1 应用流程规划

游客通过 App 注册账号，成为 App 用户，用户拥有个人日程管理的一系列功能，比如添加日程，查看日程，删除日程，到期提醒，快速添加日程等功能。App 用户也可以创建团队，邀请其他用户进入团队，从而实现团队日程管理等功能。

3.2 设计与测试规范

设计规范：

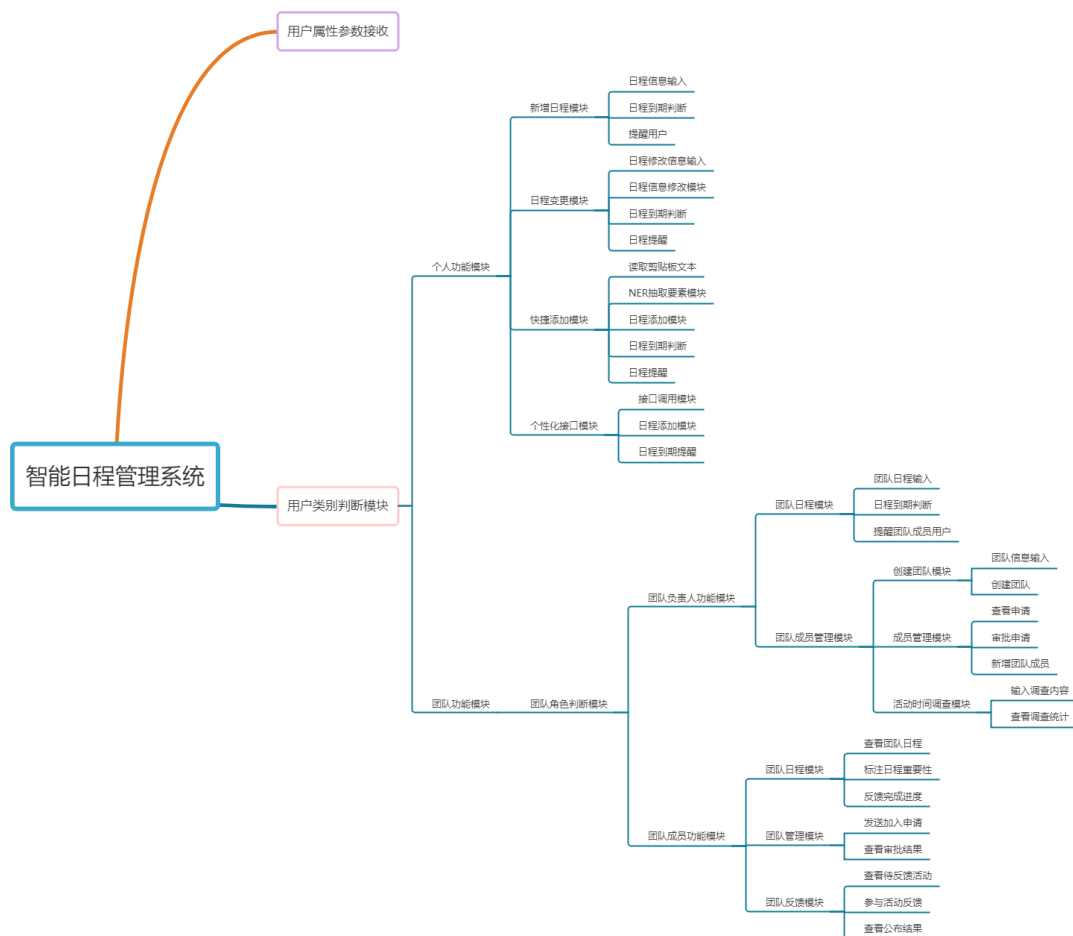
- (1). 我们采用 XUI 的组件库，在整体风格上保持一致。
- (2). 菜单深度尽量控制在三层以内。
- (3). 相似或相近的功能按钮尽量放一起。

- (4). 工具栏太多的时候可以使用工具箱。
- (5). 一条工具栏的长度最长不能超出屏幕宽度。
- (6). 菜单前的图标尽量能直观的代表要完成的操作。
- (7). 提示、警告、或错误说明应该清楚、明了、恰当。
- (8). 布局要合理, 不宜过于密集, 也不能过于空旷, 合理的利用空间。
- (9). 当选项特别多时, 可以采用列表框, 下拉式列表框。
- (10). 对重要信息的删除操作最好支持可逆性处理, 如取消系列操作。

测试规范:

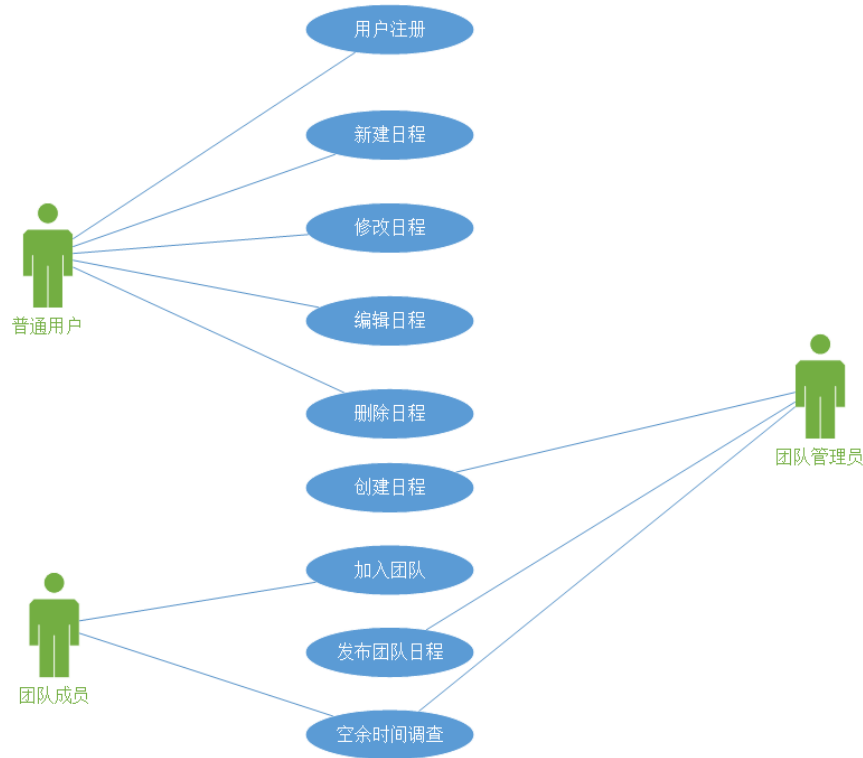
- (1). 利用 Postman 等工具对接口进行测试。
- (2). 使用 Appium 等自工具进行自动化测试
- (3). 利用 Monkey、LoadRunner 进行压力测试。

3.3 功能模块图



3.4 用例建模

用例图：



用例描述：

用例名称: 用户注册

描述: 用户在初次使用软件时需要注册用于记录日程的账号

参与者: 普通用户

前置条件: 用户能够在网络连接正常的情况下正确安装及打开软件

描述:

1. 当顾客点击注册按钮时进入该用例。
2. 用户输入账号的基本信息，包括手机号，用户名等。
3. 系统向用户手机发送验证码。
4. 用户设置登录密码
5. 系统确认注册信息后将用户信息登记到数据库

异常情况: 第 4 步中，如果两次输入的密码不一致，系统将提示用户重新输入

后置条件: 系统为用户创建新的账号。

用例名称: 添加日程

描述: 用户在需要添加新的日程时通过该用例设置日程信息

参与者: 普通用户

前置条件: 用户正常登录了软件

描述:

1. 当顾客点击添加日程按钮时进入该用例。
2. 用户输入日程的基本信息，包括时间、具体事件等。
3. 用户设置日程的提醒周期。
4. 用户点击保存按钮
5. 系统首先将日程保存到本地，随后通过网络同步到云服务器。

异常情况: 无

后置条件: 系统数据库中储存了用户的一个新的日程。

用例名称: 快速添加日程

描述: 用户在需要便捷的天际新的日程时通过该用例进行操作。

参与者: 普通用户

前置条件: 用户正常登录了软件

描述:

1. 用户切换到系统软件主界面。
2. 读取用户剪贴板的信息进行解析。
3. 根据解析结果抽取出日程的要素信息，提示用户进行快速添加。
4. 系统首先将日程保存到本地，随后通过网络同步到云服务器。

异常情况: 在第 2 步尝试读取用户剪贴板时需要向用户申请权限，如果用户拒绝则停止后续流程。

后置条件: 系统数据库中储存了用户的一个新的日程。

用例名称: 修改日程

描述: 用户对已有的日程信息进行修改，如日程的时间、事件、提醒周期等。

参与者: 普通用户

前置条件: 用户正常登录了软件并且存在一个待修改的日程

描述:

1. 用户在主页面中选择需要修改的日程
2. 用户修改对应日程的基本信息，包括时间、具体事件等。
3. 用户点击保存按钮
4. 系统首先将日程保存到本地，随后通过网络同步到云服务器。

异常情况: 无

后置条件: 系统数据库中对用户已有的一个日程信息进行修改。

用例名称: 删除日程

描述: 用户删除已有的一个日程信息。

参与者: 普通用户

前置条件: 用户正常登录了软件并且存在一个待删除的日程

描述:

1. 用户在主页面中选择需要删除的日程
2. 系统弹出确认窗口对用户进行询问确认。
3. 用户点击确认。
4. 系统将相应日程删除同时删除对应的数据库条目。

异常情况: 如果在确认阶段用户选择了取消，那么就暂停该过程

后置条件: 系统数据库中相应的日程条目已删除。

用例名称: 创建团队

描述: 用户可以创建一个新的团队。

参与者: 团队管理员

前置条件: 用户注册了一个账号并且正常登录

描述:

1. 用户在主页面中点击团队按钮。
2. 用户点击创建团队选项。
3. 用户输入团队的基本信息, 如团队名称。
4. 系统完成相应请求, 为用户新建团队。

异常情况: 如果在第三步中团队名称存在重复等异常情况, 则退出流程并向用户返回错误提示信息。

后置条件: 系统中会出现一个新创建的团队。

用例名称: 加入团队

描述: 用户可以加入一个已经存在的团队。

参与者: 团队成员

前置条件: 用户注册了一个账号并且正常登录

描述:

1. 用户在主页面中点击团队按钮。
2. 用户点击搜索团队按钮。
3. 用户输入团队编号进行查询。
4. 团队管理员对加入团队申请进行审核
4. 系统完成相应请求, 根据用户输入将用户添加到相应团队。

异常情况:

- 1、如果第 3 步中用户输入的团队编号不存在, 那么向用户返回错误信息。
- 2、如果第 4 步中团队管理员拒绝加入申请, 则无后续操作。

后置条件: 系统中的指定用户加入了特点的团队。

用例名称: 发布团队日程

描述: 用户可以在一个团队范围内发布日程。

参与者: 团队管理员、团队成员

前置条件: 用户注册了一个账号并加入了一个团队。

描述:

1. 用户选择指定日程。
2. 用户点击团队按钮将某个日程设置为团队日程。
3. 团队成员在自己的日程界面可以查看团队日程。

异常情况:

在第 2 步中用户不是团队管理员, 则无权添加团队日程。

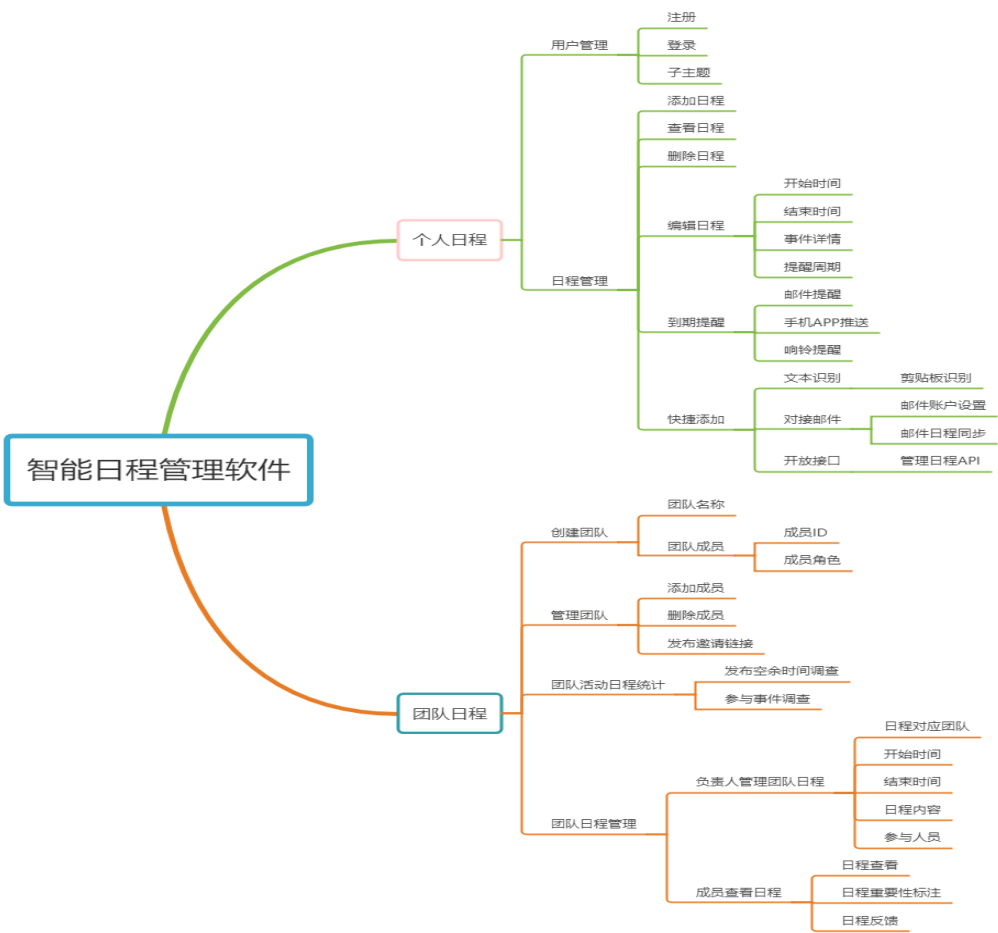
后置条件: 系统中新增了新一项的团队日程。

用例名称: 发布团队空余时间调查

描述: 团队管理员在发起团队活动前能够发布调查来调查成员的空闲时间

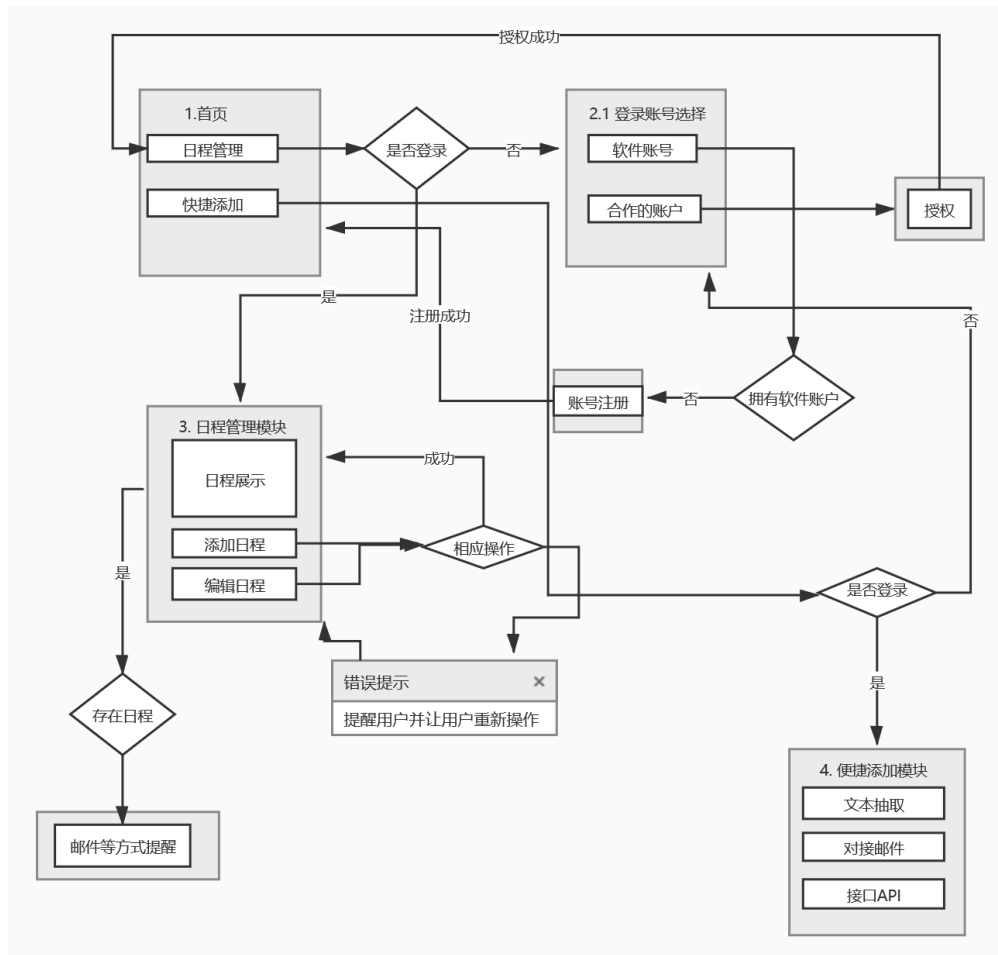
参与者： 团队管理员、团队成员
前置条件: 用户注册了一个账号并加入了一个团队。
描述:
1. 用户点击团队按钮。
2. 用户选择相应的操作团队。
3. 用户发布团队空闲时间调查，填写包括活动内容，调查范围。
4. 团队成员参与调查。
5. 团队管理员查看调查结果
6. 系统给出可视化分析结果，更加直观。
异常情况:
无

3.5 产品结构图

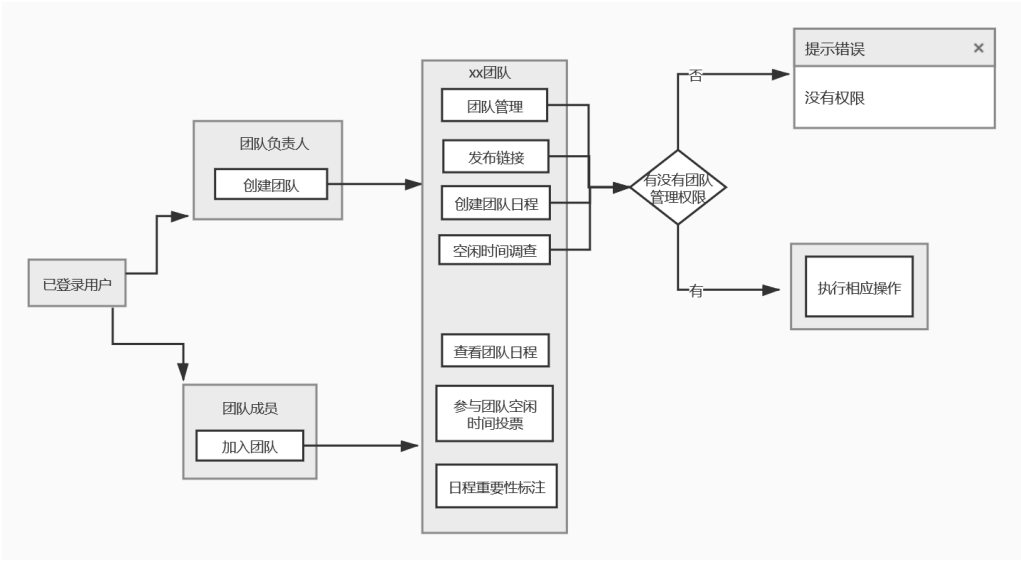


3.6 操作流程图

3.6.1 个人用户操作流程



3.6.2 团队用户操作流程



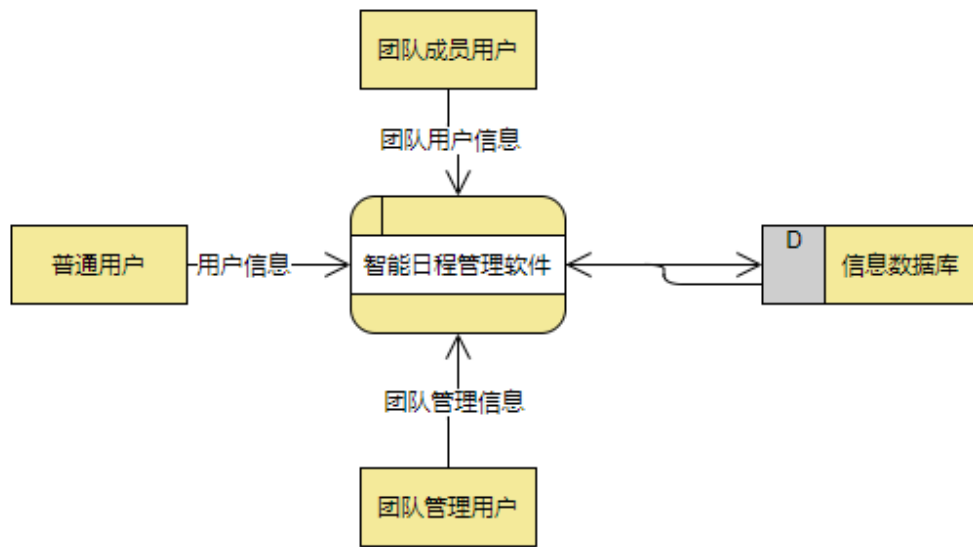
3.7 详细需求

3.7.1 总体功能需求

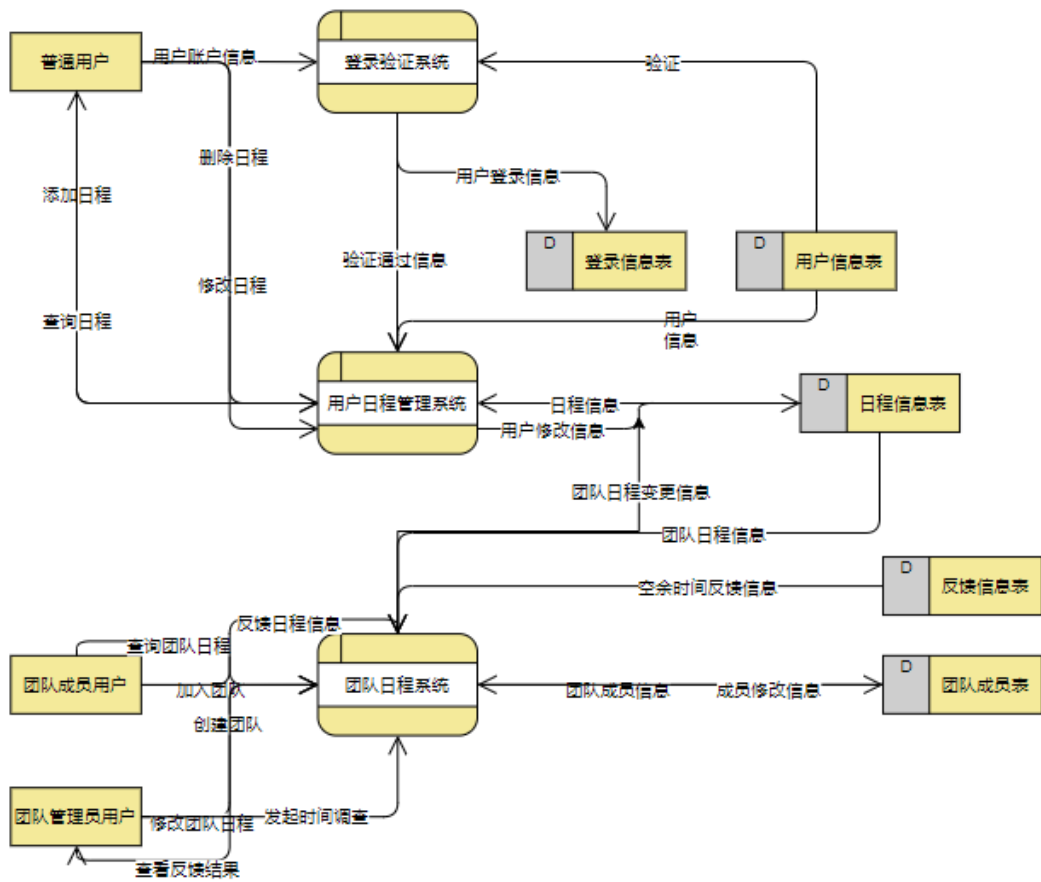
软件项目考虑到实际的日程管理软件的应用需求，将分为个人端和团队端两部分。软件一共有 个主要功能模块，对于个人端，我们提供个人日程管理功能，希望能够提供常见的日程管理软件所具备的功能的同时，给用户提供方便快捷的日程管理体验。

具体而言，对于个人端，每一个用户都有自己独立的账号，并且系统提供了相应的账号管理功能，包括账号注册，账号登录，账号密码找回。此外，系统还提供了常规日程管理，例如查看日程，添加日程，修改日程，删除日程操作。有别于传统的日程管理软件的是，我们在添加日程功能上做了大量优化，可以更加方便的添加日程。此外，系统针对日程管理还提供了到期提醒，标注日程重要性等其他功能。

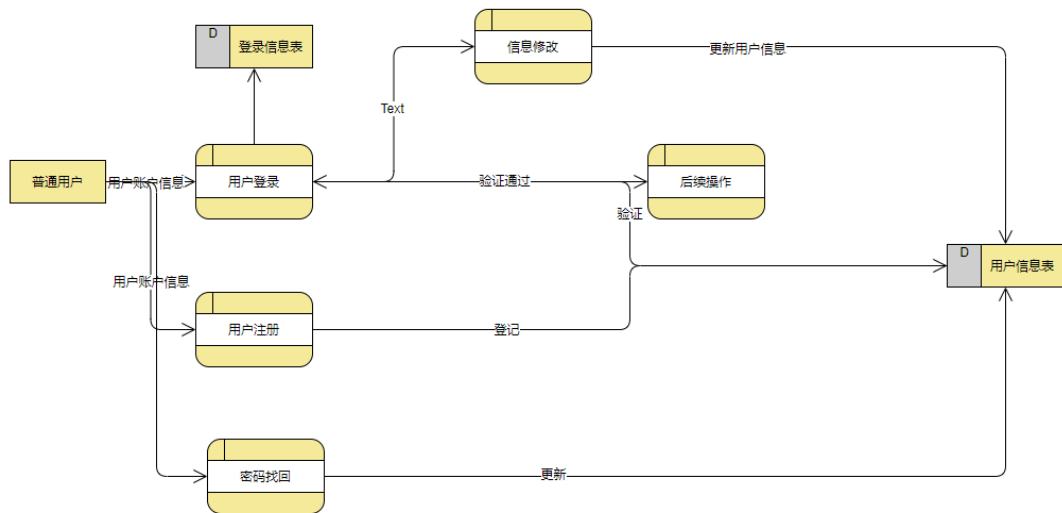
对于团队端，系统针对团队中的不同用户共分为两种角色：团队负责人，团队成员。团队负责人具有创建团队，管理团队成员，创建集体日程，收集成员反馈的功能。团队成员则具备查看团队日程，标注日程重要程度，设置私人时间等功能。



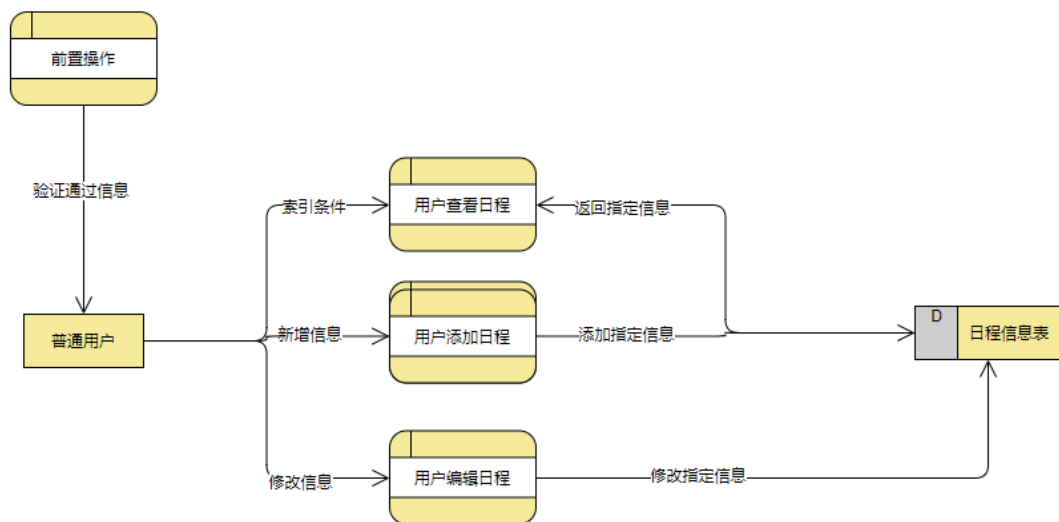
顶层数据流图



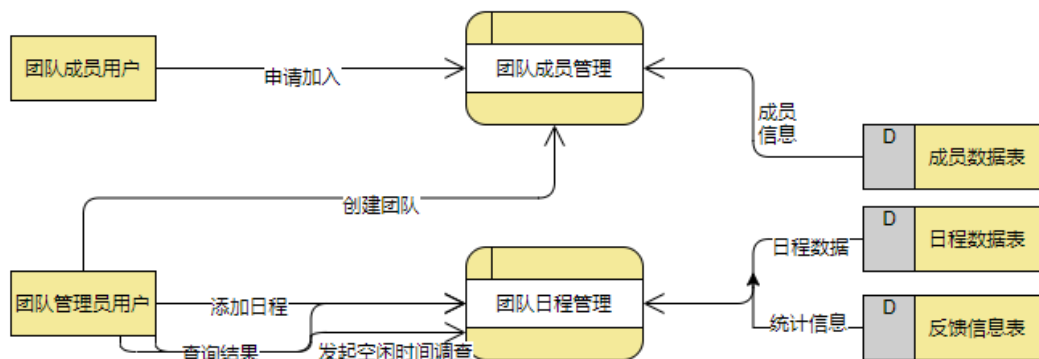
1层数据流程图



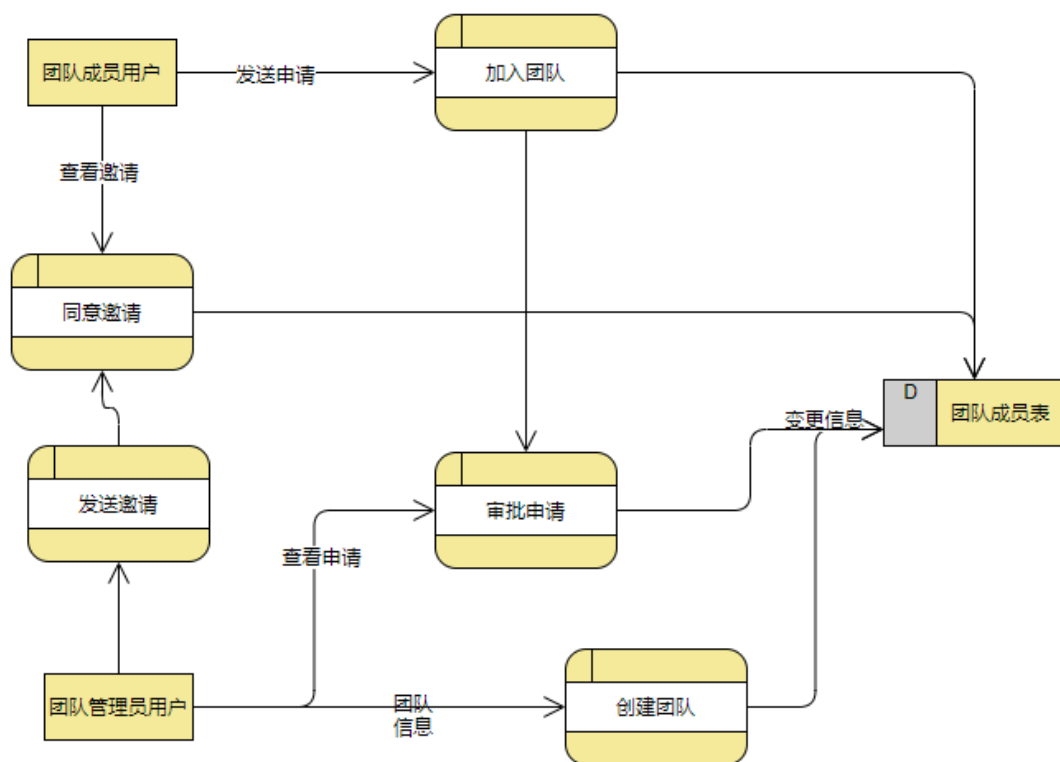
2 层数据流图（登录验证模块）



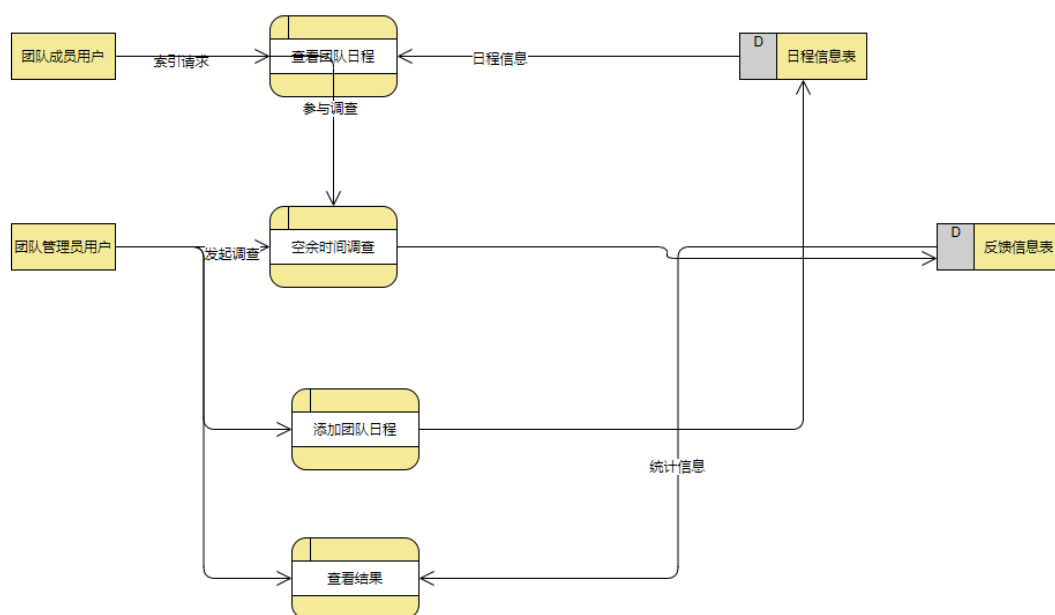
2 层数据流图（用户日程管理模块）



2 层数据流图（团队日程模块）



3 层数据流图（团队成员管理模块）



3 层数据流图（团队日程模块）

3.7.2 具体功能需求

3.7.2.1 个人日程管理

系统提供基础的日程管理功能。用户能够像使用传统日程管理软件一样，执行查看日程，添加日程，编辑日程，删除日程，日程到期提醒等操作。此外，我们还提供了额外的添加日程体验优化，大大提高了日程管理的效率（详细信息见需求分析表）

需求分析

功能模块	功能规定	原型说明
个人日程管理模块	添加日程	用户主界面是一个日历，在日历上选择时间后进入到具体日程内容的设置界面。此外，也可以通过点击界面内的添加按钮添加日程。
	查看日程	通过点击日历上的日期能够查看当前日期的日程，此外，还需要有独立的日程列表，列出所有日程。
	删除日程	点击相应的日程后提供删除按钮。
	编辑日程	点击相应的日程后提供编辑按钮。
	快捷添加-文本抽取	在用户打开软件时自动识别粘贴板的文本内容
	快捷添加-对接邮件	在功能区提供按钮进入添加邮件界面，输入邮箱的基本账户信息进行对接。
	快捷添加-接口 API	提供开发文档，供有能力的开发者二次开发。

日程提醒	在日程即将开始或到期时，系统会根据用户设定的提醒时间进行提醒。	在达到预定出发条件时由系统自动触发。
------	---------------------------------	--------------------

用例矩阵

用例名称	个人日程管理模块
用例目的	为个人用户提供方便快捷的日程管理
用例概述	用户进入软件界面后，给服务器后台发送请求查询日程详情，服务器后台查询服务器返回结果。
范围	数据统计、数据查询
级别	用户级别
参与者	用户和数据库
项目相关人员	用户：希望能够准确及时的查询后台的日程信息
和利益	系统：根据用户的请求，查询给定范围内的日程信息，并提供响应内容。
访问频率	高
前置条件	用户提交查询或修改请求
最小保证	数据库后端正常执行查询语句，提供日程查询功能
成功保证	1、系统能够对用户的修改操作产生正确的响应 2、用户可以根据相应的结果，进一步查询日程的详情信息。
触发事件	用户打开软件时自动触发
事件流	1、用户打开软件 2、用户执行登录操作同步云端日程 3、软件自动根据当前的事件日期信息，生成查询请求，向服务器发起查询 4、服务器收到请求后查询相应的数据库，返回日程信息。 5、软件根据收到的响应信息，将日程信息显示在软件界面中。 6、用户点击日程信息，可以查看具体的详情信息，或者进行编辑删除操作。
拓展流	1、每个用户都有自己的独立账号，对于没有注册账号或处于离线状态的用户，只能使用部分离线功能。 2、在其他平台端更新的数据会自动实时同步到当前端。 3、用户可以自行标注日程的类型，并提供统计界面统计各模块的时间安排详情。

3.7.2.2 团队日程管理

模块描述

系统同时还提供了团队日程管理的功能。每一个个人用户同时也可以时也可以是团队负责人或团队成员的角色。团队负责人能够进行团队管理，包括管理成员，添加团队日程，收集团

队空余时间的功能。团队成员具备查看团队日程，投票选择空余时间，标注团队日程的重要性的功能。

功能模块		功能规定	原型说明
团队 日程 管理 模块	团队负责人-创建团队	团队负责人可以创建一个团队，这是应用团队日程功能的基础。在创建团队处，可以设置团队的名称。	在功能区提供创建团队入口。
	团队负责人-团队管理	团队管理包括邀请其他用户加入团队，添加或删除新的成员等功能。	功能区首先会有一个选择团队的列表，在选择特定团队后，进入到相应团队的成员管理列表。
	团队负责人-发布邀请链接	为了简化加入团队的流程，团队负责人可以发布邀请链接，由团队成员自主点击链接后加入。	在团队管理界面，提供生成邀请链接按钮，生成邀请链接后通过其他渠道发布。
	团队负责人-创建团队日程	团队负责人能够创建日程，具体的设置和个人日程添加类似，但所创建的日程会同步到其他团队成员的界面内。	在个人添加日程的界面，提供选项选择添加到个人还是团队，并可以选择需要发布到的相应团队。
	团队负责人-空余时间调查	团队负责人可以发布一项团队活动空闲时间调查，提供几个待选择的空闲时间，发布给团队成员选择。在选择完毕后，可以很清晰的看到统计结果。	在团队管理界面，选择进入相应的团队，即可发起空余时间调查
	团队成员-加入团队	成员可以通过搜索团队编号，点击团队链接，扫描相应二维码方式加入团队。	在用户界面的主界面处进入加入团队功能
	团队成员-查看团队日程	团队日程的显示和个人日程是一样的，但是会有额外的标注提示这是一个团队日程	主界面处显示所有日程时同时选择
	团队成员-参与团队空闲时间投票	在团队负责人发布了团队空余时间调查后，软件会自动提示与个人当前日程安排没有冲突的时间给用户参考。	在有新的调查时，会在用户的消息列表中出现待办事项，供用户投票。
	团队成员-日程重要性标注	对于团队中的日程，成员可以自主选择其优先级，并设定自己的私人时间，该设置信息不会出现在团队负责人的界面中	在查看团队日程界面中，提供相应的按钮开关进行设置。

用例矩阵

用例名称	团队日程管理模块
用例目的	提供基于团队的日程管理服务
用例概述	用户以团队成员/管理员的身份进入软件后,可以分别进行团队日程管理或团队日程查询操作。
范围	数据查询
级别	用户级别
参与者	用户和数据库
项目相关人员和利益	用户: 希望能够获取到所属团队的相关信息, 并进行相应的管理编辑。 系统: 根据用户的请求, 查询给定范围内用户所对应团队的日程信息, 并提供响应内容。
访问频率	高
前置条件	用户提交查询或修改请求
最小保证	数据库后端根据用户所属的团队和当前的时间范围, 进行正常执行查询语句, 提供日程查询功能
成功保证	1、系统能够对不同团队不同时间范围内的日程做出相应的查询操作。 2、系统提供了团队管理功能, 具备管理团队成员等功能。
触发事件	用户进入团队管理功能时触发。
事件流	1、用户打开软件 2、用户登录自己的账号 3、用户进入团队管理界面 4、客户端发送请求查询用户所属的团队 5、用户选择相应的团队后可以进行管理团队, 编辑团队日程等操作。 6、用户作为团队管理员, 后续还可以收到团队成员关于团队日程安排的反馈信息。
拓展流	1、用户只能参与与自己团队相关的日程安排。 2、用户可以通过邀请链接自主选择加入相应团队。 3、系统具备空闲时间统计功能, 供团队成员间协商日程安排时间。

3.8 非功能需求说明

3.8.1 性能需求

- 该软件需满足到分钟的精度, 意味着用户发送预处理的信息到该软件, 后台必须在 60s 内实现数据的处理并且数据的往返传输。
- 该软件的硬件拟选用可靠的云服务商, 大概率保证了硬件设施的可靠性。
- 通过大量测试确保软件在交付的时候不会出现致命错误, 达到良好的软件可靠性。数据库支持超过 500 个用户的并发访问能力。

3.8.2 系统需求

- 该软件呈现了用户最近的日程，降低用户的时间管理成本，避免用户错过一些重要的活动。还帮助团队管理人员了解团队进展，加强团队间的沟通协作。该软件还提供了接口供部分有二次开发需求的用户调用。
- 软件的设计简洁且易于上手，尽量减少完成某个功能的操作步骤，提高用户的使用体验。
- 系统需合理的利用资源，保证前后台数据操作的效率，以及在数据响应和界面承载方面都要达到不会出现界面混乱、数据报错、触发按钮功能缺失、操作频繁或者快速容易崩溃的问题。
- 作为一款日程工具类的软件，应用的体积不宜过大，尽量控制在 10MB 以内。
- 为了进一步方便用户使用，软件应尽可能支持不同平台间的跨平台数据同步。

3.8.3 运营需求

- 该软件的采取低耦合高内聚的设计模式，保证软件的可扩充性，以适应技术变化和业务变化所带来的的影响。
- 采用模块化的思想，提高模块的可重用性，降低后期的维护成本。该软件应当可以在多平台中使用，具有良好的可移植性。
- 系统应具备高性价比，能对系统资源的使用进行优化，在实现系统功能的前提下，尽量节省硬件资源的开销。

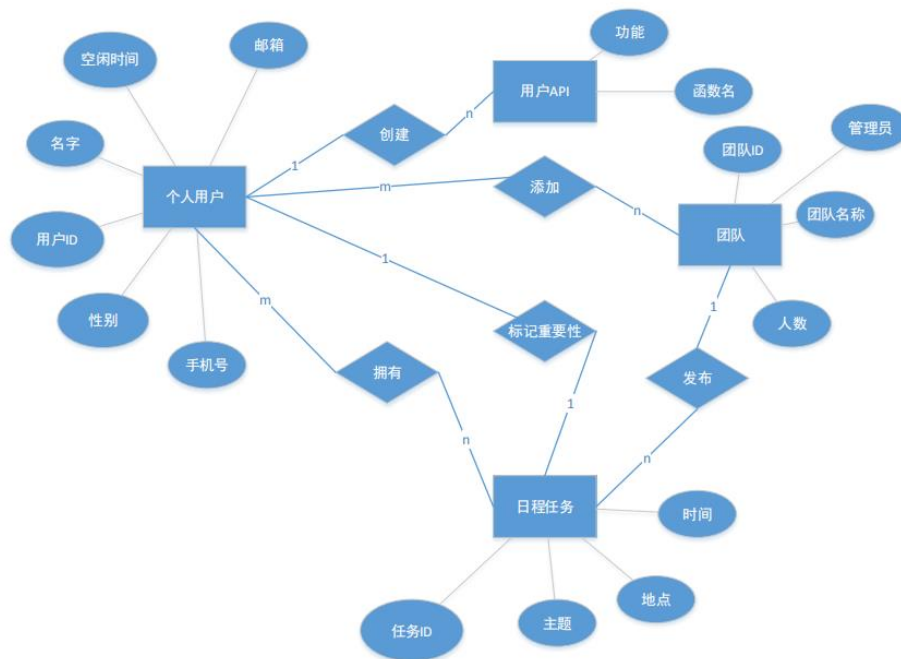
3.8.4 安全需求

- 由于日程安排信息往往涉及到用户隐私相关的内容，所以系统需要具备有良好的安全性，防止用户隐私泄露。
- 该软件记录的日程会经过鉴权之后才会被访问，充分保护数据的安全性。
- 此外，团队中不同权限的用户看到的日程信息也是不同的，不同的角色对应不同权限应该仅能查看自己所属组别的信息。

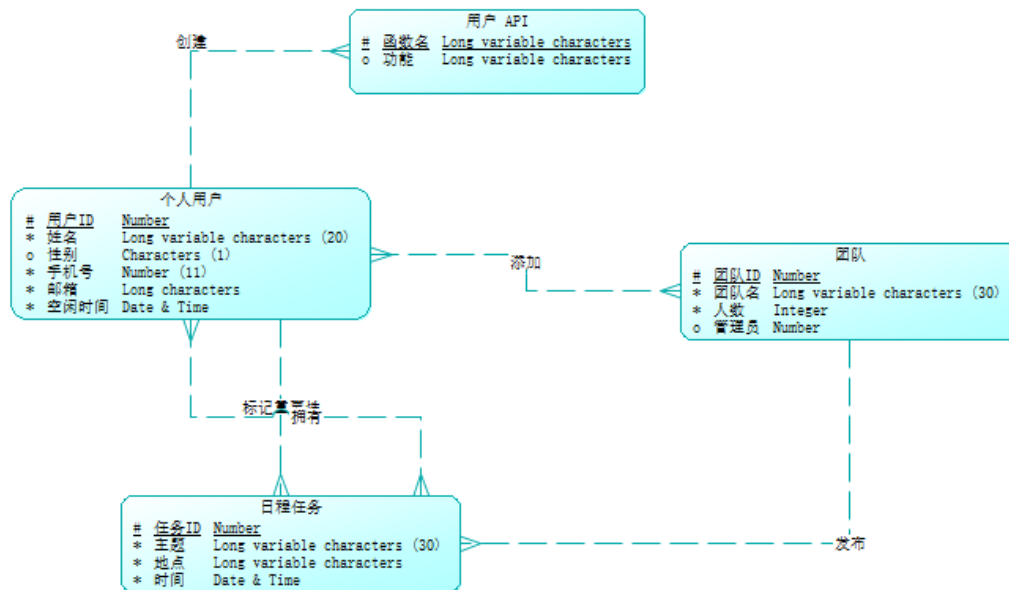
- 团队中的日程信息有的时候涉及到其他一些商业机密的内容, 为此可以考虑引入盲水印等手段溯源截图泄露信息的情况发生。

3.9 数据库设计

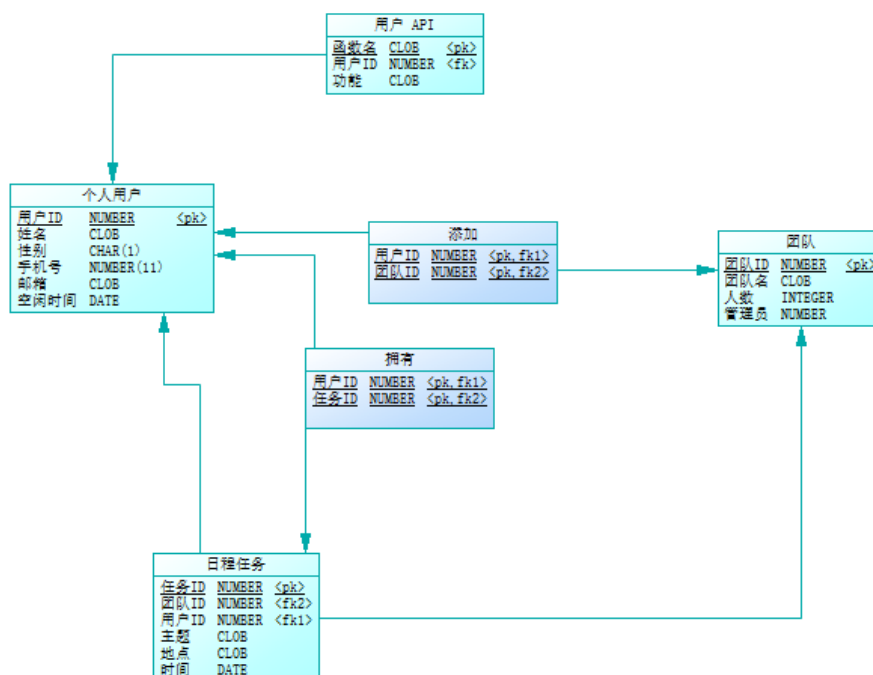
3.9.1 ER 图设计



3.9.2 CDM 模型



3.9.3 PDM 模型



3.10 开发日程表

时间	需要完成的需求
第四周 (9.28-10.11)	1. 确定开发团队的组织结构, 人员组成与人员分工。 2. 确定项目, 编写用户故事、可行性分析和需求分析
第六-七周 (10.12-10.25)	设计阶段以及前期准备
第八周-第九周 (10.26-11.4)	完成个人日程管理中前四个功能模块, 分别为: 添加日程、查看日程、删除日程、编辑日程
第九周-第十周 (11.5-11.15)	完成个人日程管理中快捷添加的模块, 分别为: 文本抽取、对接邮件、接口 API
第十一周-第十二周 (11.16-11.25)	完成团队日程管理中的前三个功能模块, 分别为: 创建团队、团队管理、发布邀请链接
第十二周-第十三周 (11.26-12.2)	完成团队日程管理中的剩余的团队负责人的模块, 分别为: 创建团队日程、空余时间调查
第十三周-第十四周 (12.3-12.13)	完成团队成员的模块, 分别为: 加入

	团队、查看团队
第十五周 (12.14-12.17)	列出第一版的总结与需要调整的地方
第十六周 (12.21-12.27)	完成第二版
第十七周 (12.28-1.3)	项目结束总结

4、技术解决方案

4.1 客户端开发

客户端开发拟采用安卓原生开发，选用 java 为开发语言，运用 google 官方推荐的 IDE Android Studio 进行开发。尽管 google 推荐使用最新的 kotlin 作为下一代的安卓应用开发语言，但我们经过考虑后认为目前情况下 kotlin 的技术支持还不够丰富，故选择了具备更多参考资料的 java 作为项目开发语言。

4.2 界面框架选择

4.2.1 选择 XUI 界面框架的原因

经过比较多个现有的安卓开源用户界面组件库，最终我们选用了 XUI (<https://github.com/xuexiangjys/XUI>) 作为我们的界面框架，具体的理由有如下几点：

- 1、与直接使用原生的安卓组件开发相比，运用现有的组件库能够较为方便快捷的开发出美观的界面，同时兼顾开发质量和开发效率。
- 2、XUI 组件库提供了许多常用的组件，如轮播条组件、常用布局组件、按钮组件等，基本覆盖了我们项目的实际需要。
- 3、XUI 组件库支持高度定制化，仅通过修改少量的顶层样式就可以控制整体

的 UI 风格，能够方便的根据项目的实际开发需要修改组件的样式。



4.2.2 选择其他组件/库的说明

此外，考虑到本项目是一个日程管理软件，需要使用日历组件显示日程条目信息，而 XUI 中并没有提供该组件。经过调研比较开源项目中的几款日历组件，最后决定选用 CalendarView 组件 (<https://github.com/huanghaibin-dev/CalendarView>) 作为项目的日历组件，具体的理由如下：

1、该组件是基于 Canvas 绘制实现的，与基于 ViewGroup 实现的其他同类型组件相比，具备渲染效率高的优点。

2、该组件支持自定义样式，通过调整样式规则能够使得其风格与 XUI 组件库的风格整体相近，从而保证一致的 UI 风格。

3、在 XUI 官方的 demo 示例程序中，有提供载入该组件的例程供参考，一定程度上可以避免不同组件库间的兼容性问题。



与此同时，客户端开发除了考虑界面效果，还需要实现的一些功能，为此还需要引入其他的第三方支持库，例如：

1、okhttp: okhttp 库封装了 http 请求的常见操作，如 get, post 等，与直接调用 android 原生的 api 实现相应请求相比，调用封装接口能够提高编程效率和代码的可读性。

2、fastjson: fastjson 是由阿里巴巴开源的一款 json 序列化和反序列化支持库，它提供了类实例与 json 序列化文本间的相互转换功能，在调用服务器接口及解析服务端数据时都需要使用。

3、其他的一些第三方支持库随项目开发进度的推进根据实际的需要再进行补充。

4.3 客户端安全

由于 java 语言自身的特性，编译所产生的字节码如果不经任何特殊处理可以非常轻松的反编译出源码，为了避免这种情况发生，我们考虑将编译产生的 apk 进行加壳保护。市面上所提供的免费加壳服务有梆梆加固、360 加固等，届时将视实际情况进行选择。

4.4 服务端开发

服务端开发主体上选用基于 python 的轻量级 web 框架 flask 实现，遵循 restful API 的风格规范，为客户端提供接口。

数据库选用关系式数据库 mysql 作为 backbone 替换 flask 内置支持的 sqlite。之所以进行这样的选择，是考虑到 sqlite 作为轻量级数据库，拓展性较差，往往仅支持单机存储，不利于后期的横向拓展。为了更好的操作数据库，我们选用 SQLAlchemy 库作为操作数据库的支持库。与直接执行 SQL 语句相比，使用这一类 ORM 框架的好处包括：

- 1、不需要过多的关注 SQL 语句的编写实现，还可以利用 ORM 框架自带的语句优化功能提高查询语句的执行效率。
- 2、避免手动拼接 SQL 语句过滤不严造成 SQL 安全漏洞，直接运用 ORM 框架提高了 web 服务的安全性。

4.5 系统架构

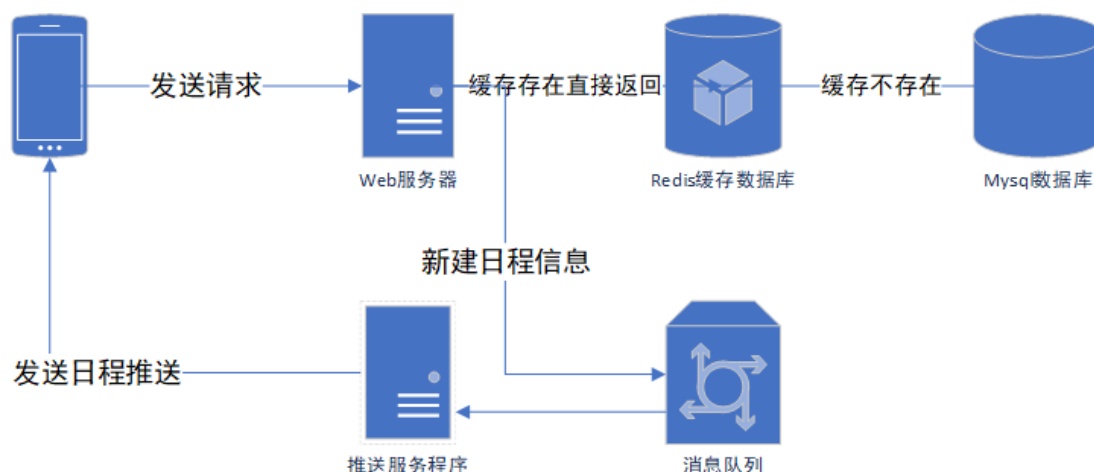
考虑到日程管理软件需要具备实时性的功能，即事件推送需要准确靠谱，涉及消息推送的部分我们选用 RabbitMQ 信息队列来实现。

RabbitMQ 是一套开源的消息队列服务软件，同时提供了各编程语言实现的调用接口。在本项目中，我们会将服务端作为消息队列的生产者，同时另外实现一个推送信息发送程序作为消费者，生产者根据用户提供的日程计划将日程信息添加进消息队列，推送信息发送程序定时读取消息队列中的待处理消息，并对符合条件的日程项发送推送消息，及时提醒用户。与此同时，为了减轻后端数据库的压力，对于高频访问的日程数据，我们选用 redis 作为我们的缓存数据库。Redis 为非关系式数据库，能够储存 key-value 型的键值对数据，对于一个查询请求，后端程序会先查看 Redis 中是否有缓存的查询结果，如果有并且是在缓存有效期内的，那么就直接使用，否则的话才去 mysql 数据库中执行查询，可以减轻数据库的访问压力。

4.6 项目部署

为了更好的践行 devops 的理念，我们选用 docker 容器部署技术作为我们后端服务最终上线的部署方案，运用 docker 容器，可以方便快捷的打包开发好的服务应用并且部署到其他的 linux 服务器上，避免了繁琐的各类配置。

最后，完整的技术解决方案示意图如下：



5、推广方案

5.1 企业的产品销售策略

主要以差异化价值点为主要优势，建立以团体和有编程能力的人为核心人群，并以此为基点，扩散用户范围。即先通过各种宣传手段稳定核心用户，并通过核心用户的宣传以及其人脉关系带动周边用户使用此产品。

产品先在小范围内做试运行，后逐步扩大。

5.2 产品推广计划

先在集中的几个月做好核心人群的宣传，宣传方式包括群发产品海报等。可以考虑以学校为基点进行推广，由于学校内有部分学院的学生修过编程类的课程，具备一定的开发能力，其次学校内的同学大部分会参加项目如大创、金种子等，

以及团队比赛如 ACM、数学建模等。因此，学校内的学生符合产品的核心人群要求。

之后以这些用户为基点，逐步扩大用户所在范围。

6、运用规划书

6.1 运营目标

前期：团队用户占主体的 70%，其余为有编程能力的用户以及个人用户

中期：团队用户占主体 60%，有编程能力的用户 10%，个人用户 30%

后期：个人用户占 70%，团队用户占 30%，其中包括有编程能力的用户

6.2 维护目标

维护范围：“智能日程管理软件” app

维护周期：一个月

维护流程：

1. 对用户反馈进行回复
2. 采纳合理的用户建议
3. 修护漏洞
4. 调整需求变更的功能

第二章 产品实现方案

1、系统的主要功能

1.1 登录/注册

该 APP 作为日程管理类 APP, 新用户可以在登录界面注册属于自己的新账号, 已注册账号的用户可以直接认证登录, 登录后享受相应的服务和功能。

1.2 时间提醒/规划

日程管理 APP 的核心功能就是时间的规划与提醒, APP 根据用户提供的空闲时间为个人用户或者团队制定合适的时间规划安排, 在时间提醒功能方面, APP 根据用户上传的工作生活任务安排信息进行提醒, 避免用户遗忘带来不便。

1.3 任务分配/查看

对于团队负责人用户可以对团队合作项目进行任务分配, 团队成员可以获取属于自己的任务事项, 团队成员可以查询任务完成进度, 根据进度调整自己的时间安排, 。

1.4 时间统计

APP 为用户提供时间统计功能, 用户可以查看自己在工作、学习、娱乐等方面的时间的花费, 通过查看时间花费统计, 为用户接下来的时间安排提供参考。

1.5 信息发送

为了便于团队间进行信息交流, 提高团队协调与工作效率, APP 对于信息发送部分设置了团队成员可以通过扫描二维码发送团队信息。

2、UI 界面设计

2.1 登录界面



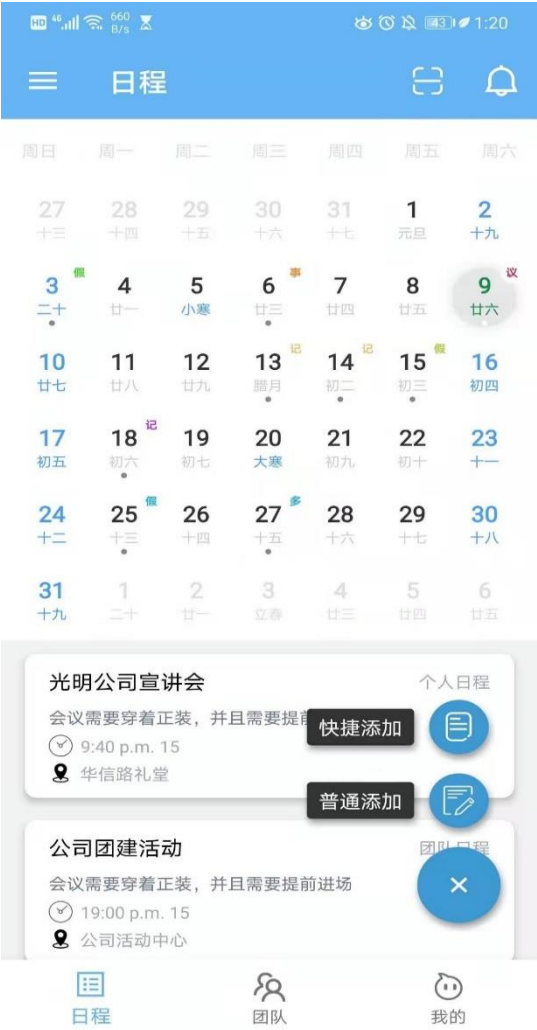
构图

APP 登录界面采用 APP 的小图标日历，充分体现 APP 的日程管理功能。对于主界面的设计，我们采用了长矩形输入框和功能小按钮。整体界面看起来功能清晰，简洁大方。

配色

在登录界面的配色上，我们以浅蓝色为背景色，不同深度的蓝色进行搭配可以让人感觉简洁大方，看起来功能明确、自然舒适。

2.2 主界面



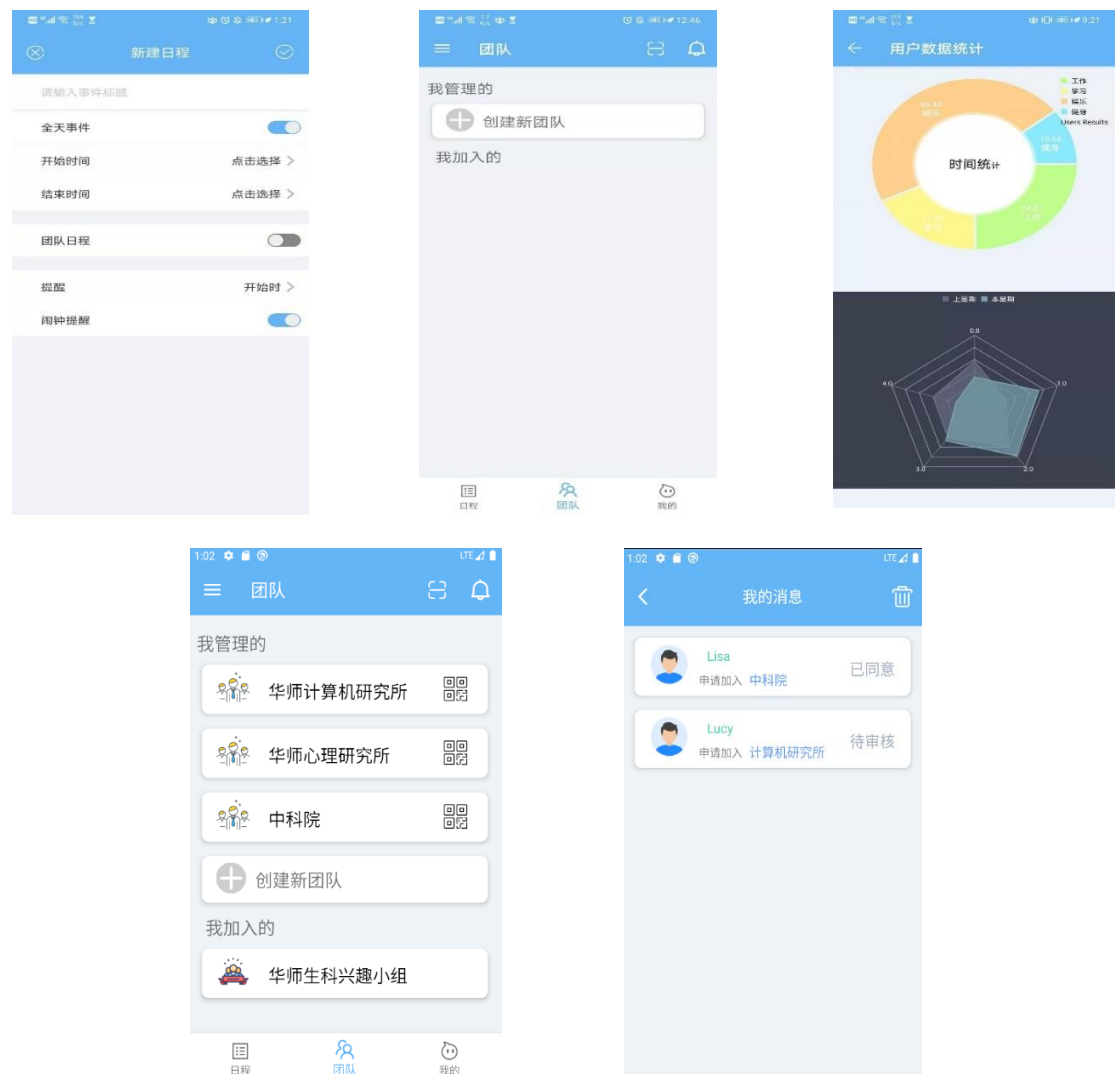
构图

主界面构图采用时间日历表和日程安排消息列表相结合，突出日程管理 APP 的核心功能。

配色

在主界面配色方面，我们采用白色为主体背景色与蓝色配合，整体配色给人感觉不会过于艳丽，而是清新淡雅，简洁大方，功能突出。

2.3 各功能界面



构图

由于我们的 APP 是应用于个人用户或团队的成员的日程管理提醒，所以各功能界面应该清晰明了，对于该界面的构图设计，我们采用了不同矩形的功能按钮和大小圆圈的查看和选定按钮。这样整体界面看起来会清晰明了，舒适清新。

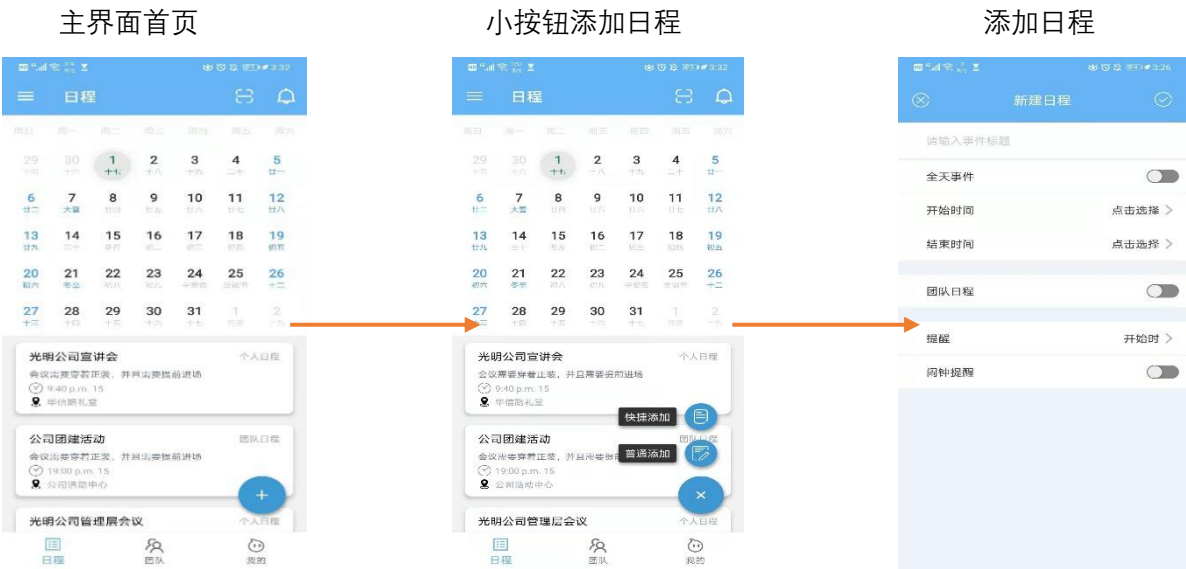
配色

在配色上，我们以蓝色和白色为主调，两种淡色的搭配可以让人感觉简洁清

晰，看起来更加自然。

2.4 主要界面的操作流程图

2.4.1 主界面操作流程图



2.4.2 日程创建界面操作流程





3、关键技术和技术难点

3.1 界面框架选择

经过比较多个现有的安卓开源用户界面组件库，最终我们选用了 XUI (<https://github.com/xuexiangjys/XUI>) 作为我们的界面框架。此外，考虑到本项目是一个日程管理软件，需要使用日历组件显示日程条目信息，而 XUI 中并没有提供该组件。经过调研比较开源项目中的几款日历组件，最后决定选用 CalenderView 组件 (<https://github.com/huanghaibin-dev/CalendarView>) 作为项目的日历组件。

3.2 客户端的开发与安全

客户端开发拟采用安卓原生开发，选用 java 为开发语言，运用 google 官方推荐的 IDE Android Studio 进行开发。尽管 google 推荐使用最新的 kotlin 作为下一代的的安卓应用开发语言，但我们经过考虑后认为目前情况下 kotlin 的技术支持还不够丰富，故选择了具备更多参考资料的 java 作为项目开发语言。

由于 java 语言自身的特性，编译所产生的字节码如果不经任何特殊处理可以非常轻松的反编译出源码，为了避免这种情况发生，我们考虑将编译产生的 apk 进行加壳保护。市面上所提供的免费加壳服务有梆梆加固、360 加固等，届时将视实际情况进行选择。

3.3 服务端开发

服务端开发主体上选用基于 python 的轻量级 web 框架 flask 实现，遵循 restful API 的风格规范，为客户端提供接口。

数据库选用关系式数据库 mysql 作为 backbone 替换 flask 内置支持的 sqlite。之所以进行这样的选择，是考虑到 sqlite 作为轻量级数据库，拓展性较差，往往仅支持单机存储，不利于后期的横向拓展。为了更好的操作数据库，我们选用 SQLAlchemy 库作为操作数据库的支持库。

3.4 系统架构

考虑到日程管理软件需要具备实时性的功能，即事件推送需要准确靠谱，涉及消息推送的部分我们选用 RabbitMQ 信息队列来实现。

RabbitMQ 是一套开源的消息队列服务软件，同时提供了各编程语言实现的调用接口。在本项目中，我们会将服务端作为消息队列的生产者，同时另外实现一个推送信息发送程序作为消费者，生产者根据用户提供的日程计划将日程信息添加进消息队列，推送消息发送程序定时读取消息队列中的待处理消息，并对符合条件的日程项发送推送消息，及时提醒用户。与此同时，为了减轻后端数据库的压力，对于高频访问的日程数据，我们选用 redis 作为我们的缓存数据库。Redis 为非关系式数据库，能够储存 key-value 型的键值对数据，对于一个查询请求，后端程序会先查看 Redis 中是否有缓存的查询结果，如果有并且是在缓存有效期内的，那么就直接使用，否则的话才去 mysql 数据库中执行查询，可以减轻数据库的访问压力。

3.5 技术细节

3.5.1 框架的基本使用

为了提高开发效率，避免重复的功能开发，我们选用了 XUI 系列的框架加快开发进度。该系

列的框架包括 XUI 用户界面框架, XPage 页面管理框架等。此外, 开源作者还提供了一个集成了以上系列框架的空壳模板。该模板中集成了以上框架, 并且还包括一个基本的示例, 我们小组在该模板的基础上二次开发, 进行了大量的工作以适应我们自己的需求。(模板链接: <https://github.com/xuexiangjys/TemplateAppProject>)

使用框架的主要好处就是可以避免一些重复繁琐的工作。例如, 界面跳转功能, 如果使用传统的方法开发, 需要机械的进行一些声明, 如果使用 XPage 框架, 问题就会简单一些。例如, 要声明一个新的界面, 只需要建立如下所示的一个类:

```
@Page(anim = CoreAnim.none)
public class NotificationFragment extends BaseFragment {
}
```


然后重写如下方法:

```
@Override
protected int getLayoutId() {
    return R.layout.fragment_notifications;
}

@Override
protected void initView() {
    XToastUtils.toast("CurrentUser: " + CurrentUser.getUserName());
    initNotificationRecyclerView();
}

@Override
protected void initListeners() {
    super.initListeners();
    initNotificationListeners();
}
```

从函数名不难推断出函数的用处, 即分别对应布局文件, 初始化视图和初始化监听器。其外, 我们要将一个控件与实例绑定时, 往往会使用 findViewById, 框架提供了一种更简洁优雅的表达方式:

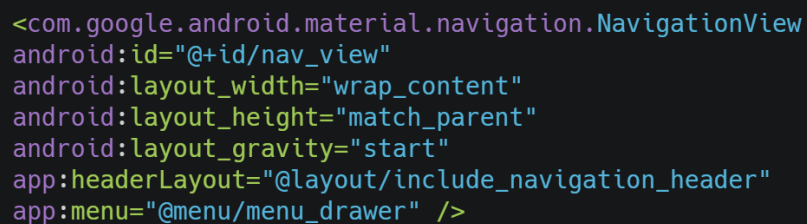


```
@BindView(R.id.notification_recyclerView)  
RecyclerView notificationRecyclerView;
```

这样即可将控件对象和对象实例进行绑定。

3.5.2 侧边导航栏的实现

在我们日常使用的很多应用中，都会有侧边导航栏，方便进行功能的跳转。结合框架模板，我们也实现了这一功能。具体而言，侧边导航栏总体上使用的谷歌原生提供的 material 组件。该组件声明如下：



```
<com.google.android.material.navigation.NavigationView  
    android:id="@+id/nav_view"  
    android:layout_width="wrap_content"  
    android:layout_height="match_parent"  
    android:layout_gravity="start"  
    app:headerLayout="@layout/include_navigation_header"  
    app:menu="@menu/menu_drawer" />
```

然后在 headerlayout 再具体配置头像等显示在侧边栏头部的样式。
而 app:menu 中则配置侧边栏菜单项的具体内容即可。
完成后的效果如下图所示：



3.5.3 底部菜单栏的实现

底部菜单栏的实现同样使用了 google 官方提供的 material 系列组件。组件声明如下：

```
<com.google.android.material.bottomnavigation.BottomNavigationView
    android:id="@+id/bottom_navigation"
    android:layout_width="match_parent"
    android:layout_height="?actionBarSize"
    android:layout_gravity="bottom"
    android:background="@color/xui_config_color_white"
    app:labelVisibilityMode="labeled"
    app:menu="@menu/menu_navigation_bottom" />
```

然后在 menu 中配置菜单项即可。最后实现的效果如下：



3.5.4 日历组件

作为一个日程管理软件，日历组件是必不可少的。为了实现美观易用的界面，进一步提升用户体验，我们使用了开源的日历组件 `CalendarView`。为了实现日历的可伸缩性（即滚动到下方待办事项时日历组件只显示当前周的日历，留出更多的空间以展示待办事项），需要将整个界面用 `calendarview.CalendarLayout` 组件进行包裹，然后在该被包裹的组件内再包含其他的组件。）

另外，由于 `CalendarView` 是一个高度定制化的组件，需要自己指定配色方案。在这里，我们使用了与主界面相符的配色方案。

最后的实现如下：



当向上滑动时，日历会自动隐藏，留出更多的空间展示待办事项。



3.5.5 listview 动画

长按 listview 里面的 item 会出现菜单项，包括编辑功能和删除功能。点击删除功能会删除 item。



在一开始实现时，我们只是简单的删除 list 中的项并且 refresh，这样的话就会非常的突兀，没有过渡。为此，通过搜索，最后选用了 recyclerview-animators 库来实现 listview 动画。

该库的使用较为简单，只需要声明一个由该库提供的 adapter。此外，在操作 list 的时候需要通过 adapter 提供的接口来实现，才会在执行操作时有相应的过度动画。

具体的效果请看视频。

3.5.6 二维码扫描

在我们的项目实现上，加入团队是通过扫描二维码来实现的，这就需要涉及到两方面的工作。


(1) 二维码生成 (2) 扫描解析二维码，下面分别对以下功能的实现方式进行介绍。

其实，这两部分功能的实现都使用了 XUI 框架作者提供的另一个开源库 XQRCode。该库提供了生成二维码和解析二维码的功能。

生成二维码功能较为简单，只需要调用 XQRCode.createQRCodeWithLogo 即可，传入的参数内容即为编码的二维码内容。

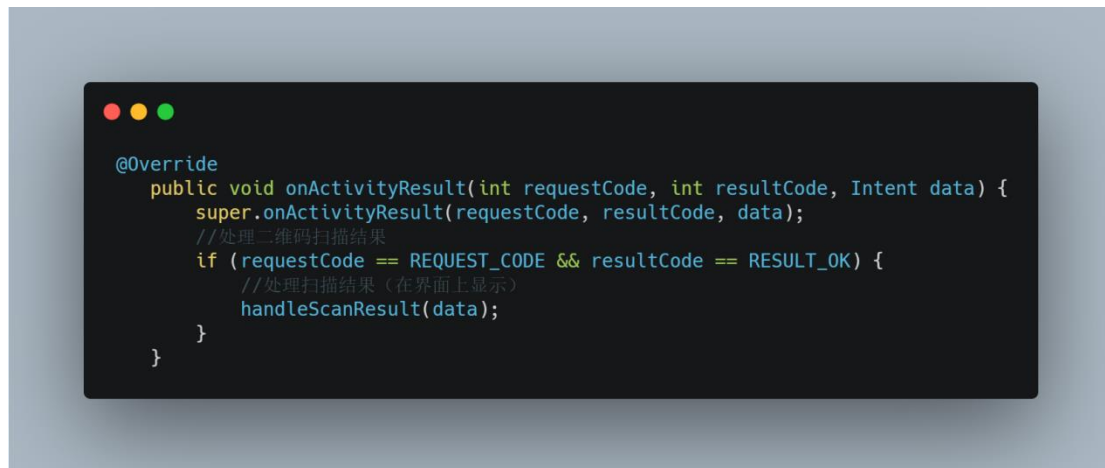
解析的话需要调用手机摄像头，这就涉及到动态获取权限的问题。同样得益于 XUI 框架，只需要在执行对应功能段时通过封装好的接口调用 @Permission(CAMERA) 即可动态声明权限。

此外，二维码扫描界面需要一个独立的线程来进行，则需要调用 @IOThread(ThreadType.Single) 防止主线程阻塞。完整代码段如下：



```
@Permission(CAMERA)
@IOThread(ThreadType.Single)
private void startScan() {
    XQRCode.startScan(this, REQUEST_CODE);
}
```

然后需要通过回调函数接收和处理扫码结果



扫码界面效果如下：



扫码成功会发出团队申请

3.5.7 团队用户管理界面

团队用户管理界面我们仿照常见的好友列表，按照拼音字母序排序。同时，在顶部有一个搜

索框便于快速检索用户。此外，在滑动到下一个字母首序的组时还会出现首字母提示。



点击后还可以设置用户权限



3.5.8 新增日程界面

该界面使用了多种控件综合完成。



3.5.9 悬浮按钮

在主页面中，有快捷添加操作和普通添加操作，隐藏在悬浮按钮中，这一布局给主页面添了不少美感。



实现代码如下:

```
<com.github.clans.fab.FloatingActionMenu
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/fab_menu"
    style="@style/FabMenu"
    android:layout_alignParentBottom="true"
    android:layout_alignParentEnd="true"
    android:paddingRight="20dp"
    android:paddingBottom="20dp"
    app:menu_colorNormal="#3E99D2"
    app:menu_colorPressed="#2488C6"
    app:menu_colorRipple="#AF2488C6"
    app:menu_labels_ellipsize="end"
    app:menu_labels_singleLine="true"
    tools:ignore="RtlHardcoded,RtlSymmetry">

    <com.github.clans.fab.FloatingActionButton
        android:id="@+id/fab_quick_add"
        style="@style/FabMenu.Buttons"
        android:src="@drawable/ic_quick_text"
        app:fab_label="快捷添加"/>

    <com.github.clans.fab.FloatingActionButton
        android:id="@+id/fab_simple_add"
        style="@style/FabMenu.Buttons"
        android:src="@drawable/ic_simply_text"
        app:fab_label="普通添加"/>

</com.github.clans.fab.FloatingActionMenu>
```

3.5.10 携带数据跳转

存储团队名字和管理员手机号, 然后携带这个数据进行跳转, 跳转到团队管理页面。

```
Bundle params = new Bundle();
params.putString("TeamName", item.getTeamName());
params.putString("ManagerPN", item.getManagerPN());
openNewPage(TeamManagerFragment.class, "key", params);
```

团队管理页面接收参数, 并取出传给 teamName 和 managerPN 变量。


```
Bundle arguments = getArguments().getBundle("key");
teamName = arguments.getString("TeamName");
managerPN = arguments.getString("ManagerPN");
```

3.5.11 获取管理的团队和加入的团队

获取创建的团队,加入 itemList 表中, 然后更新适配器 mAdapter。

```
private void getTeamFromServer() {
    BmobQuery<TeamCreate> categoryBmobQuery = new BmobQuery<>();
    categoryBmobQuery.addWhereEqualTo("ManagerPN", CurrentUser.getPhoneNum());
    categoryBmobQuery.findObjects(new FindListener<TeamCreate>() {
        @Override
        public void done(List<TeamCreate> objectLt, BmobException e) {
            if (e == null) {
                itemList.clear();
                // 将所有object装进去
                itemList.addAll(objectLt);
                mAdapter.refresh(itemList);
            } else {
                Log.e("BMOB, 查询数据失败", e.toString());
                XToastUtils.toast(e.getMessage());
            }
        }
    });
}
```

可以看到创建的团队查询的是 TeamCreate 表, 但是查找加入的团队确实另一种思路: 在 Teammate 中通过成员手机号搜索成员与队伍的联系, 再加入到 joinList 列表, 进而再让适配器更新。代码实现难度不会特别大, 主要技术点在表格的架构上。

```

private void getJoinTeamFromServer() {
    BmobQuery<Teammate> categoryBmobQuery = new BmobQuery<>();
    categoryBmobQuery.addWhereEqualTo("MatePN", CurrentUser.getPhoneNum());
    categoryBmobQuery.findObjects(new FindListener<Teammate>() {
        @Override
        public void done(List<Teammate> objectLt, BmobException e) {
            if (e == null) {
                joinList.clear();
                // 将所有object装进去
                joinList.addAll(objectLt);
                joinAdapter.refresh(joinList);
            } else {
                Log.e("BMOB, 查询数据失败", e.toString());
                XToastUtils.toast(e.getMessage());
            }
        }
    });
}

```

3.5.12 团队权限判断

一个团队中，只有团队管理员具有管理权限。因此我们在团队管理模块中需要对成员的权限进行判断。

4.7 团队用户管理界面中，只有团队管理员才具有设置用户权限和删除成员的权利，这里实现的方法是如果当前用户手机号与负责人手机号一致，才监听点击事件弹出下滑窗。

```

if (managerPN.equals(CurrentUser.getPhoneNum())) {
    initListViewListener();
}

```

解散团队的权限同样需要进行判断，如果用户只是普通的团队成员，则右上角是没有垃圾箱的图标，该图标是解散团队的操作入口。这里是通过一个简单的判断来决定要不要增加这个图标并设置监听事件。

```

if (managerPN.equals(CurrentUser.getPhoneNum())) {
    titleBar.addAction(new TitleBar.ImageAction(R.drawable.ic_dustbin) {
        @Override
        public void performAction(View view) {
            showSimpleConfirmDialog(teamName, managerPN);
        }
    });
}
}

```

3.5.13 解散团队

解散团队后需要删除 Teammate 表中所有团队成员与该团队的关联(数据项), 删除 TeamCreate 表中该团队的记录, 删除申请加入该团队的消息。第一个任务和第三个任务需要删除多个, 采用批量删除。这里以删除团队成员为例, 首先需要去数据库查询该表中的所有成员, 然后将他们的 Objectid 放入一个列表 delTeammateLt, 再通过该列表传给 doBatchDeleteTeammate 函数进行操作。这里注意不要把 doBatchDeleteTeammate 函数中操作放在 CategoryBmobQuery.findObjects 函数的后面, 因为在执行查找操作的时候, 系统会自动开一个线程, 主线程将会继续执行下去, 就会出现 delTeammateLt 还没有要删除的 Objectid 就去批量删除了, 结果就是没有数据被删除。

```

private void delGroupTeammate(String teamName, String managerPN) {
    List<BmobObject> delTeammateLt = new ArrayList<>();
    BmobQuery<Teammate> categoryBmobQuery = new BmobQuery<>();
    categoryBmobQuery.addWhereEqualTo("TeamName", teamName);
    categoryBmobQuery.addWhereEqualTo("ManagerPN", managerPN);
    categoryBmobQuery.findObjects(new FindListener<Teammate>() {
        @Override
        public void done(List<Teammate> objectLt, BmobException e) {
            if (e == null) {
                for (int i = 0; i < objectLt.size(); i++) {
                    Teammate tm = new Teammate();
                    tm.setObjectId(objectLt.get(i).getObjectId());
                    delTeammateLt.add(tm);
                }
                doBatchDeleteTeammate(delTeammateLt);
            } else {
                Log.e("BMOB", e.getMessage());
            }
        }
    });
}
}

```

DoBatchDeleteTeammate 操作:

```
private void doBatchDeleteTeammate(List<BmobObject> delTeammateLt) {  
    // 批量删除  
    new BmobBatch().deleteBatch(delTeammateLt).doBatch(new QueryListListener<BatchResult>() {  
        @Override  
        public void done(List<BatchResult> results, BmobException e) {  
            if (e == null) {  
                Log.d("delTeammate", "删除团队成员成功");  
            } else {  
                Log.e("delTeammate: ", "失败: " + e.getMessage() + ", " + e.getErrorCode());  
            }  
        }  
    });  
}
```

4、用户体验记录和分析

4.1 用户的体验记录

这次对于用户的体验感受，我们找了 10 位同学，这十位同学来自不同的学校，我们分别记录了他们的手机型号、系统版本、使用体验的优缺点，我们根据他们的反馈进行了总结。

学生序号	手机型号	系统版本	优点	缺点
1	荣耀 10	Android 10	界面风格简约，功能界面特点突出明显。	时间花费统计功能过于简单，没有细致的分析。
2	小米 9	Android 10	界面简洁，功能简	功能界面缺少快捷

			单明了。	键调用相关功能。
3	小米 10	Android 10	这界面有日历功能，突出了日程管理的特点，功能清晰，使用简单。	团队成员管理功能还可以更加完善，比如成员可以申请团队负责人权限，管理团队成员。
4	华为 P30	Android 10	一款不错的 APP，界面精美，内容丰富。	团队的日程新建功能可以再独立放置到团队板块。
5	OPPO Reno5	Android 10	功能使用简单，方便查询时间日期，建立个人日程安排	时间花费统计功能还可以更加完善。
6	荣耀 30	Android 10	APP 做得不错，可以面向个人和团队型用户，界面简洁清晰。	日程创建功能可以添加详细的日程说明，还有小部分功能可以更加完善。
7	VIVO X50	Android 10	APP 的界面构图配色都很清晰明了，给人感觉很舒服，功能也很齐全。	可以添加一些功能的快捷键直接进入，方便功能使用。
8	荣耀 X10	Android 10	界面简单，字体清	某些功能可以继续

			晰，功能齐全。	完善。
9	华为 P40pro	Android 10	界面简洁，操作简单。	时间花费统计功能可以继续完善。
10	红米 K30S	Android 10	APP 免费，界面不错，功能齐全。	界面可以弄得更好些，功能可以继续完善。

4.2 用户体验的总结分析

优点总结：

- 1). 免费
- 2). 界面简洁，字体清晰
- 3). 功能明了齐全，操作简单
- 4). 服务对象全面，包括个人用户和团队用户
- 5). 有突出日程管理 APP 的特点

缺点总结：

- 1). 某些功能可以继续完善
- 2). 可以添加一些功能快捷键

5、已完成的改进和存在的问题

5.1 已完成的改进

我们对体验用户所反映的一些问题进行了总结，并对其中的一些问题进行了反思和改进。

其中，我们对 APP 的界面继续进行了完善，对界面的字体进行了调整，使得界面和字体更加清晰明了；在主界面添加了日程添加的快捷键，这样便于用户添加新的日程，凸显 APP 的核心功能，完善了用户时间花费统计功能。

5.2 仍存在的问题

APP 整体的功能是完善齐全的，但在细节上处理还有很多需要改进的地方，对于界面的设计可以更加多样化，各种的功能的处理可以更加细致，例如团队管理的功能可以继续细化，添加团队成员的权限申请和使用。需要继续提高用户的使用体验，对于项目的日程管理 APP，我们后期需要做的事情很有很多。

第三章 测试大纲和测试报告

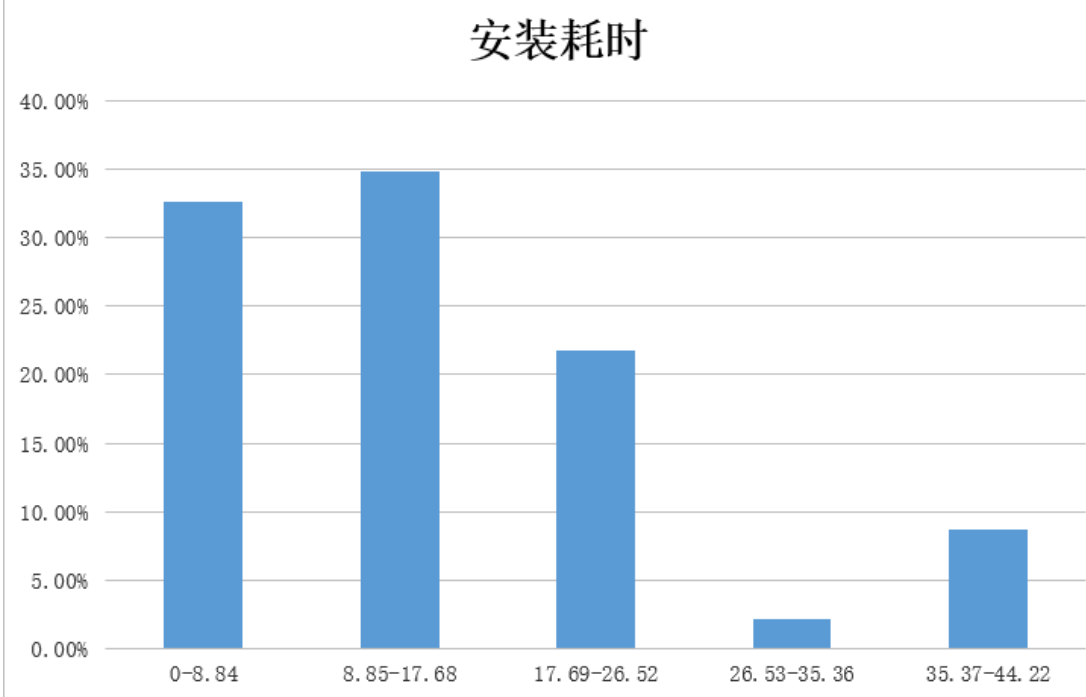
1.1 测试结论

由测试图片可以得知该 APP 的测试安装通过率有 95.65%，拥有较高的安装通过率。

1.2 性能测试

性能指标概况							
	安装耗时(s)	启动耗时(s)	CPU占用(%)	内存占用(MB)	电池温度(℃)	网络流量(MB)	FPS
平均值	14.72	1.73	1.05	56.04	27.27	0.00	29.59
峰值	2.81	0.29	0.0	3.74	21.8	0.00	64.0
(手机型号)	一加 7T Pro	努比亚 Z17 S	锤子 坚果 3	OPPO N1 Mini	一加 5T	华为 荣耀V30 PRO	三星 Galaxy S6 Edge

(1). 安装耗时

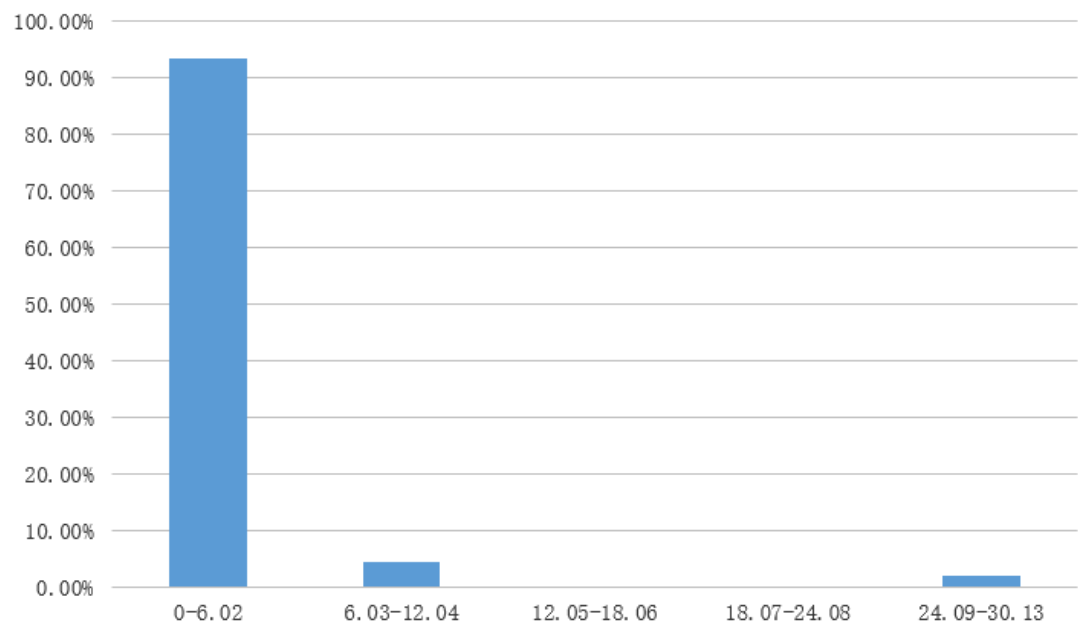


(2). 启动耗时

启动耗时					
区间(s)	0-6.02	6.03-12.04	12.05-18.06	18.07-24.08	24.09-30.13
个数	43	2	0	0	1
占比	93.48%	4.35%	0.00%	0.00%	2.17%

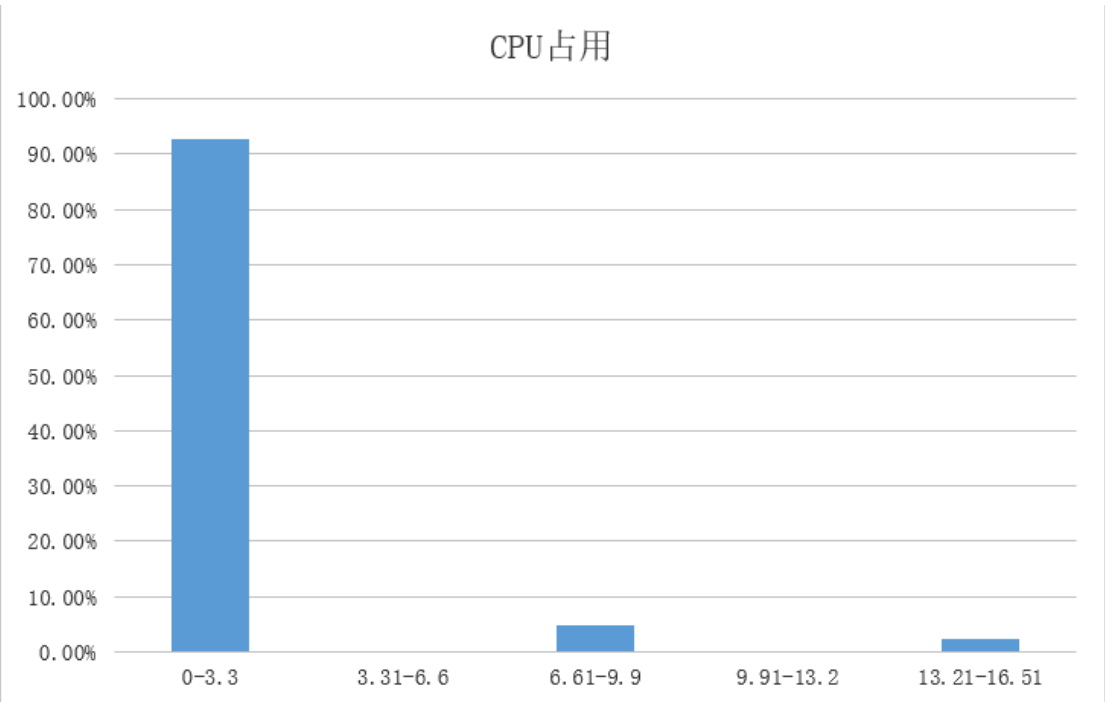
安装耗时					
区间(s)	0-8.84	8.85-17.68	17.69-26.52	26.53-35.36	35.37-44.22
个数	15	16	10	1	4
占比	32.61%	34.78%	21.74%	2.17%	8.70%

启动耗时



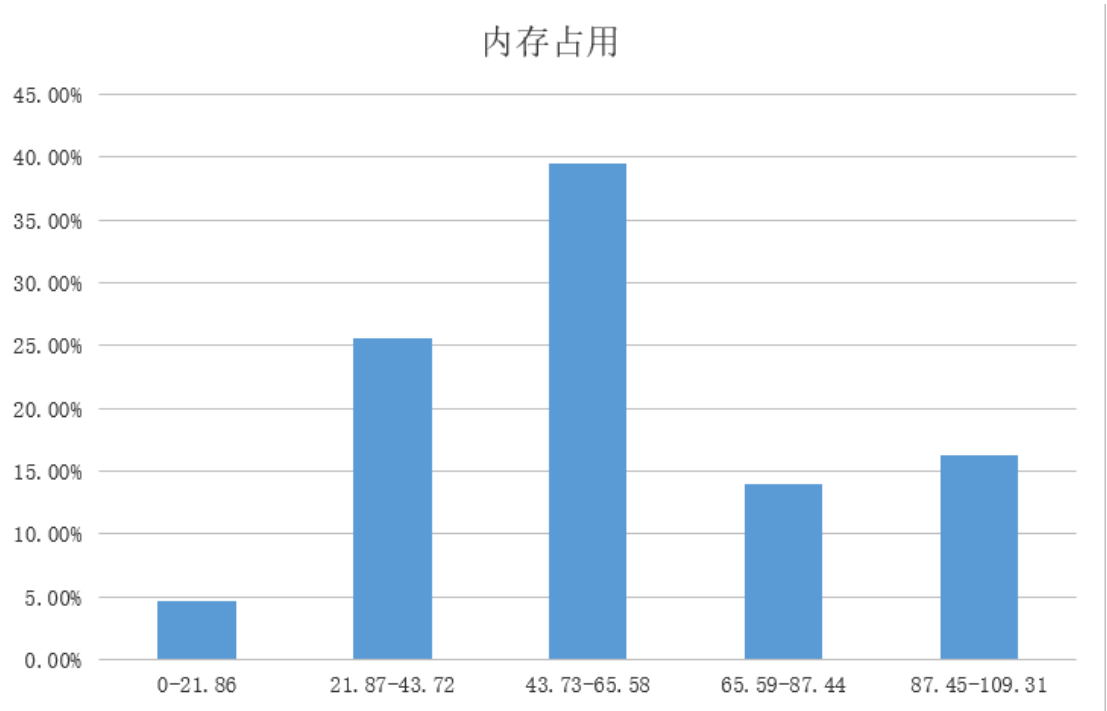
(3). CPU 占用

CPU占用					
区间 (%)	0-3.3	3.31-6.6	6.61-9.9	9.91-13.2	13.21-16.51
个数	39	0	2	0	1
占比	92.86%	0.00%	4.76%	0.00%	2.38%



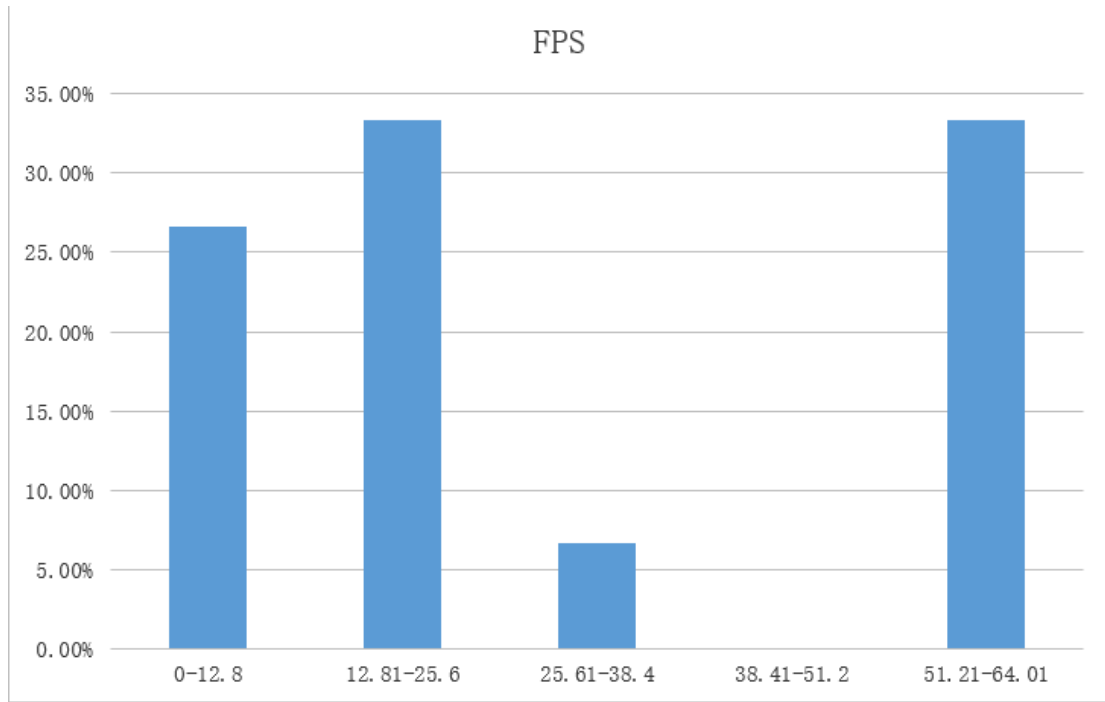
(4). 内存占用

内存占用					
区间 (MB)	0-21.86	21.87-43.72	43.73-65.58	65.59-87.44	87.45-109.31
个数	2	11	17	6	7
占比	4.65%	25.58%	39.53%	13.95%	16.29%



(5). FPS

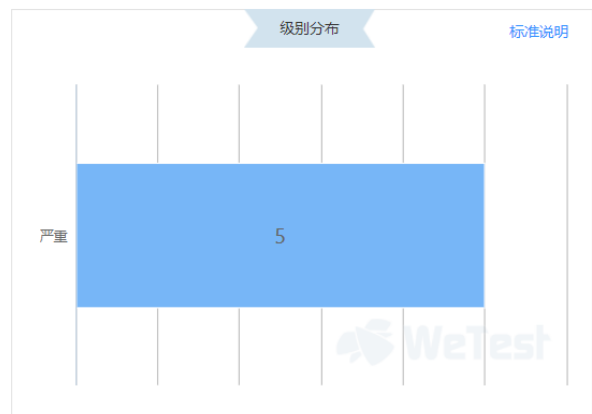
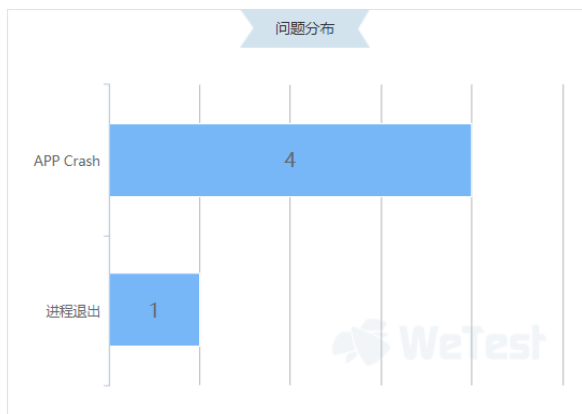
FPS					
区间	0-12.8	12.81-25.6	25.61-38.4	38.41-51.2	51.21-64.01
个数	4	5	1	0	5
占比	26.67%	33.33%	6.67%	0.00%	33.33%



根据图片的测试数据，在性能指标方面，APP 在各种机型的测试结果处于正常水平内，满足预期结果。

1.3 问题概述

问题分布



问题等级说明：

- 1、致命问题：导致游戏无法进入或运行
- 2、严重问题：可以进入游戏，但主流场景受严重影响
- 3、一般问题：可以进入游戏，可以运行主流场景，主流场景用户体验受一定影响；或其他场景用户体验受严重影响
- 4、提示问题：可以正常运行游戏，主流场景正常运行，其他场景出现问题，用户体验受一定影响。

问题级别会根据复现概率做一定范围内的微调

通过测试我们可以知道主要问题的种类，也为我们后期继续完善 APP 提供了有价值的参考。

设备名称	状态	初始化	安装	启动	遍历	monkey	卸载	Crash	ANR	Exception	错误描述	操作
小米 红米Note 4G	失败	✓	✓	✗	-	-	-	5	0	0	启动失败	--
OPPO N1 Mini	失败	✓	✓	✗	-	-	-	5	0	2	启动失败	--
红米 Note4X	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
OPPO A57	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
魅蓝 S6 (3GB RAM/全网通)	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
努比亚 Z17 mini	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
努比亚 Z17 S	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
努比亚 Z17 mini S	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
一加 7T Pro	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
华为 nova	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
华为 荣耀V30 PRO	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
华为 荣耀8青春版	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
小米 MIX 2	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
一加 5	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
华为 Mate 9	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
华为 Mate 30	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
华为 nova 6 5G	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
魅族 魅蓝metal	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
vivo Y31	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
三星 Galaxy S6 Edge+	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
金立 S10	通过	✓	✓	✓	-	-	✓	0	0	0	--	--

1.4 测试机型参考

vivo X20	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
vivo Xplay5S	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
锤子 M1L	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
华为 荣耀 V9	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
ASUS_X015D	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
红米 5	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
V1809A	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
小米 9	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
红米 4X	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
vivo Y66	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
小米 6	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
华为 畅享8	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
华为 荣耀畅玩 5	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
三星 GALAXY S4	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
vivo X6D	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
vivo Y17T (移动3G)	通过	✓	✓	✓	-	-	✓	6	0	0	--	--
一加 5T	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
魅族 PRO 7	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
努比亚 Z17	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
VIVO X30	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
OPPO A3	通过	✓	✓	✓	-	-	✓	0	0	0	--	--
乐视 乐 2	通过	✓	✓	✓	-	-	✓	0	0	0	--	--

第四章 产品安装和使用说明

1.1 产品安装

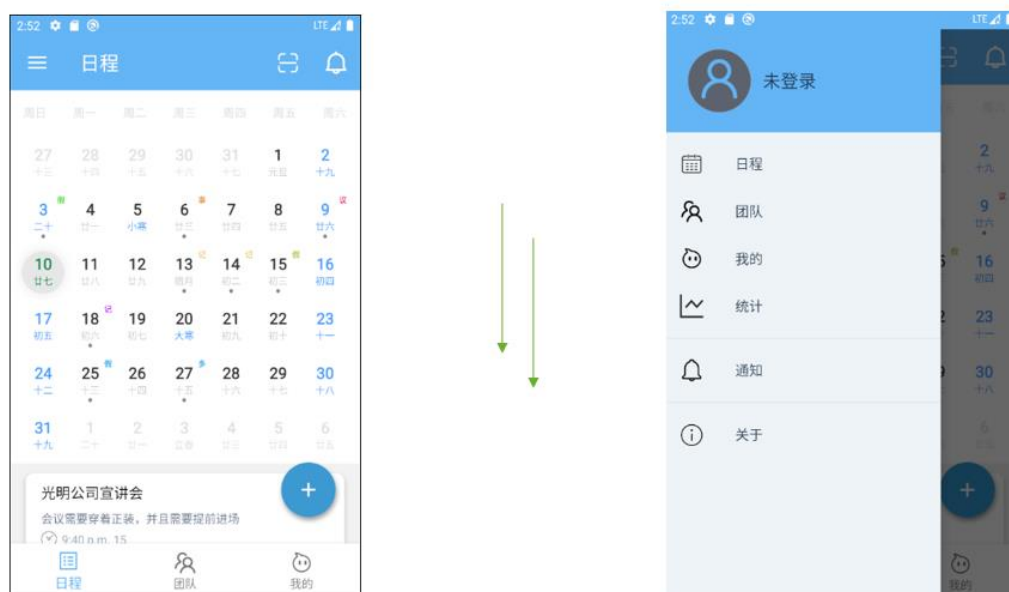
该 APP 可以直接使用 apk 安装文件进行安装，点击 apk 文件即可自动完成安装。通过手机系统自带的卸载功能即可完成软件卸载，要求 Android 手机版本最低为 5.0。

1.2 使用说明

1.2.1 客户端运行

用户在安装“日程管理 APP”完毕后，点击手机桌面的图标即可进入 APP，主界面如图所示：

主界面包括了日历时间查询、日程添加、团队管理等模块。



1.2.2 用户注册

用户下载安装软件后，进入登录界面，如果已经注册过账户，可以直接登录使用服务。未注册用户可点击“注册”按钮，输入手机号、用户名、密码。确定注册。注册界面如图：



1.2.3 用户登录

注册成功的用户通过以下步骤进行登录：点击桌面图标->输入手机号码和密码-> 点击“登录”。登录成功后进入主界面。登录界面如图：



1.2.4 功能界面的使用

功能界面包括日程添加、团队管理和用户时间花费统计功能，日程添加可以提醒用户重要的日程时间，时间花费统计可以让用户清楚了解自己在工作、学习和娱乐等方面的时间花费占比，总之用户可以根据界面的不同功能和特点进行 APP 的使用。各功能界面如图所示：



如下操作添加日程



以下界面进行团队管理



