**专业方向综合项目 开发计划书**

**张尧 陈东 姜其升**

**TEam 12**

**MEETING！**

基于微信小程序的协同日程安排应用

**目录**

[1 引言 2](#_Toc41762879)

[1.1 编写目的 2](#_Toc41762880)

[1.2 项目背景和机遇 2](#_Toc41762881)

[1.3 项目介绍 2](#_Toc41762882)

[1.4 项目技术与工具 3](#_Toc41762883)

[1.5 参考资料 6](#_Toc41762884)

[2 项目概述 7](#_Toc41762885)

[2.1 项目范围 7](#_Toc41762886)

[2.2 过程模型和管理方法 7](#_Toc41762887)

[2.3 主要任务 9](#_Toc41762888)

[2.4 项目度量 10](#_Toc41762889)

[2.4.1 使用FP度量 10](#_Toc41762890)

[2.4 资源分配 11](#_Toc41762891)

[2.4.1 识别资源 11](#_Toc41762892)

[2.4.2 识别资源需求 11](#_Toc41762893)

[2.4.3 调度资源 13](#_Toc41762894)

[2.5 风险管理计划 15](#_Toc41762895)

[2.5.1 风险辨认 15](#_Toc41762896)

[2.5.2 风险评估 16](#_Toc41762897)

[2.5.3 风险策划 18](#_Toc41762898)

[2.6 监督与控制 19](#_Toc41762899)

[2.6.1 目的 19](#_Toc41762900)

[2.6.2 角色与职责 19](#_Toc41762901)

[2.6.3 流程图 20](#_Toc41762902)

[3 实施计划 20](#_Toc41762903)

[3.1 项目WBS 21](#_Toc41762904)

[3.2 识别任务及依赖 21](#_Toc41762905)

[3.3 甘特图 23](#_Toc41762906)

[3.4 详细计划 24](#_Toc41762907)

[4 产品 25](#_Toc41762908)

[4.1 项目产品 25](#_Toc41762909)

[4.2 运行环境 25](#_Toc41762910)

[4.3 服务 26](#_Toc41762911)

[4.4 验收标准 26](#_Toc41762912)

# 1 引言

## 1.1 编写目的

为了保证项目团队按时保质地完成项目目标，便于项目团队成员更好地了解项目情况，使项目工作开展的各个过程合理有序，因此以文件化的形式，把对于在项目生命周期内的工 作任务范围、各项工作的任务分解、项目团队组织结构、各团队成员的工作责任、开发进度、 项目内外环境条件、风险对策等内容做出的安排以书面的方式，作为项目团队成员以及项目 干系人之间的共识与约定，项目生命周期内的所有项目活动的行动基础，项目团队开展和检 查项目工作的依据。.

本项目开发计划书，就是为了更好的规划开发小组成员之间的工作任务而编写的。

## 1.2 项目背景和机遇

在众多需要多人参与的任务中，例如会议安排、团队协作、各种约会等等，在这些事件中我们可能都亲身体验过这样的一个痛点，事件参与成员的空闲时间不统一，就难以为一件共同事件安排计划。作为大学生的我们更能体会到这一点。我们每个人在课外都有着多多少少的项目，每个人上课时间不同，每个项目的团队成员不同，大家都有着各自的计划，而且很多时候项目都是同等重要的，所以有效的安排团队事件将很大程度上提升我们的工作效率，增加团队默契。

如今市面上存在着各式各样的日程管理软件，且几乎都为独立app的形式。其中大多数日程管理软件都只是专注与个人日程的管理，而忽略了团队事件与个人日程之间的冲突。当然也中不少日程管理应用功能全面，但从另一个角度来说这也增加了软件使用的复杂程度。而多数用户都会因为功能复杂和难以上手的原因放弃使用这样的日程管理应用。此外在团队事件都存在这样的一个特点，团队在一定的时间内是相对固定，而且多数的团队都会拥有各自的交流空间，而这其中大多数的都会选择微信群。所以这里存在着几个机会，Meeting系统在保留最基础的个人日程管理功能的基础上，增加对与团队事件的管理，对功做加法和减法，尽可能保证功能的必要性和使用的简单性。同时通过微信小程平台呈现将避免用户下载独立应用，降低产品接触目标用户的难度，与微信群之间建立良好互动。

## 1.3 项目介绍

“Meeting”是一款允许允许共享协作的日程安排小程序。

本系统将在提供日程管理的功能基础上，提供小组事件以及共享事件的管理。以小组中所有成员的日程信息为基本数据，为组长快速确定小组事件时间提供便捷。作为日程管理应用，Meeting允许个人向日程中添加事件管理事件。对于小组及共享事件管理模块，Meeting允许用户创建组并成为组长或以组长的权限删除组或以小组成员身份退出组。用户可以通过微信群中的邀请链接加入工作组。成为小组成员后可以查看组中其他成员的日程。小组长创建小组事件时可以获得基于小组成员日程信息的小组共有空闲时间推荐。此外，小组成员还可以针对事件设置隐私，以及对小组设置勿扰等。

本系统将业务范围集中在一周内的个人日程及小组事件管理上，尽最大可能将功能集中并考虑详尽，主要提供的核心功能为工作组中共享事件的自动推荐和创建、管理，辅以小组成员及权限的管理、个人事件的管理等功能。而将不会涉及到超过一周的事件管理、万年历查看功能等与核心功能无关的功能。

本系统的最终目标是提供一个能够便捷地通过微信小程序和微信群的良好联动生态，提供给用户管理个人短期内日程、便捷创建工作组并快速约定集体事件时间的应用，增加小组合作或活动策划的效率。

小组

近期日程管理

共享及小组事件管理

个人

“Meeting”



## 1.4 项目技术与工具

**Redis**

Redis是一个开源的使用 ANSI C语言编写，支持网络，可基于内存也可持久化的日志型，Key-Value数据库，并提供了多种语言的 API ,相比 Memcached 它支持存储的类型相对更多 (字符，哈希，集合，有序集合，列表等)，同时Redis是线程安全的。

客户端连接 Redis 使用的是 TCP协议，直连的方式每次需要建立 TCP连接，而连接池的方式是可以预先初始化好客户端连接，所以每次只需要从 连接池借用即可，而借用和归还操作是在本地进行的，只有少量的并发同步开销，远远小于新建TCP连接的开销。另外，直连的方式无法限制 redis客户端对象的个数，在极端情况下可能会造成连接泄漏，而连接池的形式可以有效的保护和控制资源的使用。

Jedis 在实现上是直连 redis server，多线程环境下非线程安全，除非使用连接池，为每个 redis实例增加 物理连接。

我们的项目使用Redis来支持微信用户登录服务相关。当新用户登录时，后端会向微信发送请求以获取该用户的“openid”、“session\_key”和后端将分配一个新的“userid”，这是我们在关系数据库中对用户的主键。这三个条目都存储在Redis作为后端uuid生成的密钥的值。下次用户登录时，微信小程序会向微信发送请求，查看其会话密钥是否过期。如果不是，小程序告诉后端使用现有的条目。然后小程序通信从上一次使用uuid的后端，后端将检查相应的“userid”告诉哪个用户在线。

**MyBatis**

MyBatis 是支持普通 SQL查询，存储过程和高级映射的优秀持久层框架。MyBatis 消除了几乎所有的JDBC代码和参数的手工设置以及结果集的检索。MyBatis 使用简单的 XML或注解用于配置和原始映射，将接口和 Java 的POJOs（Plain Ordinary Java Objects，普通的 Java对象）映射成数据库中的记录。

MyBatis是半自动的ORM框架，原来叫ibatis， 10年apache软件基金组织把它托管给了goole code，就重新命名了MyBatis，功能相对以前更强大了。它相对全自动的持久层框架Hibernate，更加灵活，更轻量级，这点我还是深有体会的。MyBatis的一个强大特性之一就是动态SQL能力了，能省去我们很多串联判断拼接SQL的痛苦，根据项目而定，在一定的场合下使用，能大大减少程序的代码量和复杂程度，不过还是不是过度太过复杂的使用，以免不利于后期的维护和扩展。

**HikariCP连接池**

HikariCP是由日本程序员开源的一个数据库连接池组件，代码非常轻量，并且速度非常的快。根据官方提供的数据，在i7,开启32个线程32个连接的情况下，进行随机数据库读写操作，HikariCP的速度是现在常用的C3P0数据库连接池的数百倍。在SpringBoot2.0中，官方也是推荐使用HikariCP。优势是

1.字节码更加精简，所以可以加载更多代码到缓存。

2.实现了一个无锁的集合类型，来减少并发造成的资源竞争。

3.使用了自定义的数组类型，相对与ArrayList极大地提升了性能。

4.针对CPU的时间片算法进行优化，尽可能在一个时间片里面完成各种操作。

**Lettuce**

Lettuce是一个高性能基于Java编写的Redis驱动框架，底层集成了Project Reactor提供天然的反应式编程，是 一种可伸缩，线程安全，完全非阻塞的Redis客户端，多个线程可以共享一个RedisConnection,它利用Netty NIO 框架来高效地管理多个连接，从而提供了异步和同步数据访问方式，用于构建非阻塞的反应性应用程序。通信框架集成了Netty使用了非阻塞IO，5.x版本之后融合了JDK1.8的异步编程特性，在保证高性能的同时提供了十分丰富易用的API，

支持Redis的新增命令ZPOPMIN, ZPOPMAX, BZPOPMIN, BZPOPMAX。

支持通过Brave模块跟踪Redis命令执行。

支持Redis Streams。

支持异步的主从连接。

支持异步连接池。

新增命令最多执行一次模式（禁止自动重连）。

全局命令超时设置（对异步和反应式命令也有效）。

**Apache Shiro**

Apache Shiro 是一个功能强大、灵活的，开源的安全框架。它可以干净利落地处理身份验证、授权、企业会话管理和加密。

实际上，它实现了管理应用程序安全性的所有方面，同时尽可能避免出现问题。它建立在完善的接口驱动设计和面向对象的原则之上，可以在任何你想象得到的地方实现自定义行为。但是，对于所有事情来说，默认情况下都是合理的，这与应用程序安全性是一样的。至少这是我们所追求的。

我们 POST 用户名与密码到 /login 进行登入，如果成功返回一个sessionKey，失败的话直接返回 401 错误。

之后用户访问每一个需要权限的网址请求必须在 header 中添加 Authorization 字段，例如 Authorization: token ，token 为密钥。

后台会进行 token 的校验，如果有误会直接返回 401。

**微信小程序**

微信小程序是一种不用下载就能使用的应用，也是一项创新，经过将近两年的发展，已经构造了新的微信小程序开发环境和开发者生态。微信小程序也是这么多年来中国IT行业里一个真正能够影响到普通程序员的创新成果，已经有超过150万的开发者加入到了微信小程序的开发，与我们一起共同发力推动微信小程序的发展，微信小程序应用数量超过了一百万，覆盖200多个细分的行业，日活用户达到两个亿，微信小程序还在许多城市实现了支持地铁、公交服务。微信小程序发展带来更多的就业机会，2017年小程序带动就业104万人，社会效应不断提升

本项目采用微信小程序的客户端形式，充分利用跨平台一次开发的遍历，也是让我们在同类产品中具有更多可得性的地方。

**HyperText Transfer Protocol Secure (HTTPS)**

HTTPS （全称：Hyper Text Transfer Protocol over SecureSocket Layer），是以安全为目标的 HTTP 通道，在HTTP的基础上通过传输加密和身份认证保证了传输过程的安全性。HTTPS 在HTTP 的基础下加入SSL 层，HTTPS 的安全基础是 SSL，因此加密的详细内容就需要 SSL。 HTTPS 存在不同于 HTTP 的默认端口及一个加密/身份验证层（在 HTTP与 TCP 之间）。这个系统提供了身份验证与加密通讯方法。它被广泛用于万维网上安全敏感的通讯，例如交易支付等方面。

HTTP 协议虽然使用极为广泛, 但是却存在不小的安全缺陷, 主要是其数据的明文传送和消息完整性检测的缺乏, 而这两点恰好是网络支付, 网络交易等新兴应用中安全方面最需要关注的。

关于 HTTP协议的明文数据传输, 攻击者最常用的攻击手法就是网络嗅探, 试图从传输过程当中分析出敏感的数据, 例如管理员对 Web 程序后台的登录过程等等, 从而获取网站管理权限, 进而渗透到整个服务器的权限。即使无法获取到后台登录信息, 攻击者也可以从网络中获取普通用户的隐秘信息, 包括手机号码, 身份证号码, 信用卡号等重要资料, 导致严重的安全事故。进行网络嗅探攻击非常简单, 对攻击者的要求很低。使用网络发布的任意一款抓包工具, 一个新手就有可能获取到大型网站的用户信息。

另外,HTTP协议在传输客户端请求和服务端响应时, 唯一的数据完整性检验就是在报文头部包含了本次传输数据的长度, 而对内容是否被篡改不作确认。 因此攻击者可以轻易的发动中间人攻击, 修改客户端和服务端传输的数据, 甚至在传输数据中插入恶意代码, 导致客户端被引导至恶意网站被植入木马。

## 1.5 参考资料

1. 普雷斯曼《软件工程实践者的研究方法》

2. 拉曼《UML和模式应用》

3. Arun Gupta《Java EE 8精粹》

4. 休斯《软件项目管理》

5. 雷磊《微信小程序开发入门与实践》

6. 刘增辉《MyBatis从入门到精通》

7. 克雷格·沃斯《Spring Boot实战》

# 2 项目概述

## 2.1 项目范围

项目当前大周期为1个月，于2020年5月20日起，至2020年6月27日止。在当前大周期 结束前，预期完成“开会！MEETING！”前后端主体框架开发(如用户注册、登录等功能)，并完成包括但不限于个人事件的添加与管理、工作组的创建及退出、邀请自动加入同一个工作组、查看工作组中他人日程、自动推荐公共事件时间、勿扰模式等功能模块的开发，以及商务合作、广告等营收方式的技术开发。

|  |  |
| --- | --- |
| 项目范围 | 非项目范围 |
| 可运行的小程序  前后端代码  数据库设计  服务器部署  广告接入模块 | 用户数量  商务合作  广告 |

## 2.2 过程模型和管理方法

本项目采用Scrum敏捷开发模型，即在每个Sprint完成当前冲刺的需求分析与评级、产品开发以及交付迭代工作。对于本项目而言Sprint周期为一周，由产品经理担任Sprint负责人角色，软件质量保证工程师完成Sprint迭代交付和验收工作，全体团队成员对当前Sprint周期内出现的问题与收获的经验进行总结，获得软件过程管理的优化与提升。

对于Sprint任务部署与Scrum计划，本项目主要采用两种管理方式：Scrum在线管理工具Teambition与线下backlog。

**Teambition**

Teambition是国内团队协作工具的创导者，通过帮助团队轻松共享和讨论工作中的任务、文件、分享、日程等内容，让团队协作焕发无限可能。Teambition在网页、桌面、移动环境都打造了体验出众的应用，所以你随时随地都可以和团队协作，其iPhone应用还被苹果公司评为2015年度最佳应用。目前，已经有超过一百万用户通过Teambition进行团队协作，其中也包括多个行业的龙头企业。Teambition公司已获得来自戈壁投资、IDG、盘古及北极光的多轮融资，并于2014年入选中国最具投资价值企业50强，2019年3月26日Teambition被阿里巴巴全资收购。

Teambition是国内敏捷开发在线管理平台的最佳实践之一，该工具为敏捷开发设计，可同团队协同工作使用，便于Sprint任务设定与任务分配，便于子任务、子需求设定与控制，便于开发人员登记工作时长与任务完成度，对开发人员当前的任务通过发送邮件等方式进行提醒。并且，该工具还可以生成实际工作时长与计划时长报表，便于Sprint负责人(产品经理)对下一个Sprint的软件开发过程进行更合理有效的安排与管理优化。

同时，Teambition还可以将子任务赋予不同的颜色，在界面的的待办区、正在进行区与任务完成区进行切换，并将人事安排(如休假、请假等)与对当前周期的调整、团队成员的情绪管理等体现在界面上，方便Scrum团队的沟通。

**Backlog**

在项目开始的讨论阶段，我们创建了一个待定项backlog列表，记录所有的有商业价值的项目需求或特性的优先级列表，并按照优先级进行排序，给每一个sprint分配故事并进行故事点估算，如ProjectEverySprintEstimation.xls中所示。在随后的项目开发中，我们将持续更新Meeting-scrum.xlsx表，在每个sprint中，我们会每天估算自己完成的工作量，并计算剩余工作量，绘制相应燃尽图。

在开发过程中，要求产品经理主要引导Sprint任务的部署与分配，维护Teambition平台与backlog上的内容；要求开发工程师、架构工程师、测试工程师、质量保证工程师等人员及时于Teambition平台登记个人的工作时长，调整任务状态(未开始、进行中与已结束)；要求团队成员积极维护Teambition平台，需要提出请假等需求时及时与团队成员沟通；此外，还要求团队成员在每日工作开始前进行15分钟站会，与团队成员分享自己前日进行的工作、当日计划进行的工作与在工作中遇到的问题，同样记录于Teambiton平台，并由产品经理及时调整工作安排。

项目版本控制标准

本项目通过GitHub实现版本控制和团队协作。团队成员被要求在通过单元测试后，及时将自己的工作成果上传到分支中，开启pull request请求，待审核通过无版本冲突后，谨慎进行merge操作。团队成员被要求严禁使用git push -f (强制推送)指令，如遇版本冲突需及时解决。分支合并工作由开发人员结对完成，共同确认后进行分支合并。

GitHub项目权限需严格控制，前端与后端代码分别建立仓库，由产品经理对仓库权限负责。离开项目组的成员需及时取消仓库权限。

会议

**Sprint 计划会议**

该过程预计耗时 2 小时。会议前，由产品经理整理 Product Backlog，其粒度较粗，每个需求称为一个“故事（Story）”。Sprint 计划会议需要团队全员参与，全员共同商议选择一个“故事”， 设定为本次 Sprint 的目标。之后由全体成员共同拆分当前“故事”，细化粒度，按照需求的重要程度分为 5 级（非常重要、重要、一般、不重要、很不重要），并进行可行性分析，对细化后的功能点分析其实现的可行性。当完成以上工作后，由产品经理整理总结当前Sprint 的Sprint Backlog， 并为每位团队成员设定工作计划，由前文所述的 Ones.ai 在线平台进行管理。于本项目而言，该活动定于周日进行。

**每日站会**

该过程预计耗时 15 分钟。Sprint 周期中，每日（周一至周六）早上开始工作前，团队成员需进行站会，向团队其他成员分享本人前日完成的工作进度、当日预计进行的工作，并向团队成员提出当前遇到的问题。

在当日站会完成后，团队成员需于线下白板和线上 Scrum 管理工具 Ones.ai 上更新核对自己当前的工作进度，更新每项任务的工作状态，及时对 Sprint 燃尽图进行更新。

**Sprint 演示会议**

该过程预计耗时 1 小时。在 Sprint 演示会议（Sprint Review Meeting）中，团队的每个成员都需要参与，由产品经理带领团队分别展示每位成员完成的工作，以及当次迭代的产品结果。该活动计划于周六工作结束前进行。

**Sprint 回顾会议**

该过程预计耗时 0.5 小时。该会议（Sprint Retrospective Meeting）主要供各位团队成员回顾总结当次 Sprint 中遇到的问题、任务分配不合理等状况，并向产品经理（项目经理）提出意见， 以便在下一个 Sprint 中得到解决，使软件过程得到优化和提升，提高团队效率。该活动计划与Sprint 演示会议合并进行。

## 2.3 主要任务

**项目中产生的交付产物为：**需求规约、需求分析规约、系统分析、系统设计、系统测试文档、项目程序客户端和服务器端、用户帮助文档等。

**项目过程中的里程碑有**：项目前期里程碑、项目策划里程碑、建模里程碑、构建里程碑、测试里程碑、上线里程碑。里程碑象征着一阶段的完成，有产出的交付文档或者程序，但在实际中，为了加快进度，一个阶段还未结束，可能另一阶段的任务已经开始并行着手解决。

|  |
| --- |
| 系统分析与设计 |
| 数据库设计 |
| 登陆注册微信接口模块分析设计 |
| 事件模块分析设计 |
| 日历（会议组）模块分析设计 |
|  |
| 服务器配置 |
| 数据库部署 |
| 登录注册微信接口模块 |
| 用户账户 |
| 微信方面用户安全认证 |
| 事件模块 |
| 创建个人事件 |
| 修改个人事件 |
| 删除个人事件 |
| 事件创建修改显示前端界面 |
| 日历（会议组）模块 |
| 用户查看会议组成员事件 |
| 创建团队事件 |
| 修改团队事件 |
| 创建团队日历 |
| 解散团队日历 |
| 分享团队日历 |
| 加入团队日历 |
| 退出团队日历 |
| 设置勿扰模式 |
| 推荐组内时间 |
| 日历（会议组）显示界面 |
| 我的主页显示界面 |
|  |
| 整体调试 |
| 部署 |

## 2.4 项目度量

在软件项目中，对项目进行度量是十分必要的。通过软件度量，项目组可以及时评估正在进行的项目的状态，跟踪潜在风险；并且可以在问题变得严重之前发现问题，以调整工作流程或任务；还可以评估项目团队控制软件工作产品质量的能力。

### 2.4.1 使用FP度量

在软件项目中以 LOC（Line of Codes，代码行数）和 FP（Function Points，功能点个数） 作为衡量标准，由于前端后端开发语言不同，选择更为合理的 FP 作为度量方式。

使用 FP 作为度量标准的优势有以下几点：FP 度量与编程语言无关；LOC 度量往往会导致鼓励冗余的代码，而使用FP 度量则不会导致简短实现的阻碍；FP 度量更容易衡量可重用组件影响。

在此，以Event模块为例，对功能点 FP 进行度量评估。

大致功能复述：

|  |
| --- |
| Event 模块 |
| 添加个人日历事件 |
| 添加会议组日历事件 |
| 修改个人日历事件 |
| 修改会议组日历事件 |
| 删除个人日历事件 |
| 删除会议组日历事件 |
| 与他人共享个人日历事件 |
| 创建会议组 |
| 邀请成员加入会议组 |
| 将成员移除会议组 |
| 设置会议组的隐私模式 |
| 设置会议组的请假模式 |
| 安排事件推荐时间 |
| 安排事件冲突处理 |
| 查询个人日历事件 |
| 查询会议组日历事件 |
| 查询会议组内成员空闲情况（成员日历事件） |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 软件系统组成 | 复杂度分级 | | | | | | | | | 总CFP |
| 简单 | | | 适中 | | | 复杂 | | |
| 计数 | 权重 | 积分 | 计数 | 权重 | 积分 | 计数 | 权重 | 积分 |
| 用户输入 | 6 | 2 | 12 | 13 | 3 | 39 | 8 | 5 | 40 | 91 |
| 用户输出 | 3 | 4 | 12 | 3 | 5 | 15 | 1 | 6 | 6 | 33 |
| 用户在线查询 | 2 | 3 | 6 | 1 | 6 | 6 | 1 | 10 | 10 | 22 |
| 逻辑文件 | 3 | 1 | 3 | 4 | 3 | 12 | 5 | 5 | 25 | 40 |
| 外部接口 | 3 | 1 | 3 | 12 | 2 | 24 | 6 | 5 | 30 | 57 |
| 总CFP |  |  |  |  |  |  |  |  |  | 243 |

计算结果：

## 2.4 资源分配

### 2.4.1 识别资源

|  |  |
| --- | --- |
| 劳动力 | 产品经理、前端开发人员、后端开发人员、数据库管理员、系统分析师、系统设计师 |
| 设备 | 个人电脑、服务器 |
| 空间 | 个人家中 |
| 时间 | 大把时间 |
| 资金 | 租服务器的钱 |

### 2.4.2 识别资源需求

针对劳动力资源的分配紧张的问题，专门分析劳动力资源的需求。

|  |  |  |
| --- | --- | --- |
| 活动 | 所需天数 | 所需资源 |
| 1 项目前期 | | |
| 团队建立 | 11 | 项目经理 |
| 举办启动会议 | 2 | 项目经理 |
| 识别利益相关者 | 4 | 项目经理 |
| 收集市场需求 | 11 | 项目经理、系统分析师 |
| 收集用户需求 | 6 | 项目经理、系统分析师 |
| 里程碑 | 0 |  |
| 2 项目策划 | | |
| 定义项目目的 | 5 | 项目经理 |
| 确定所需资源 | 3 | 项目经理 |
| 确定项目风险 | 3 | 项目经理 |
| 创建项目WBS | 3 | 项目经理 |
| 确定任务 | 5 | 项目经理、系统分析师、系统设计师 |
| 确定任务周期 | 3 | 项目经理 |
| 确定任务依赖 | 5 | 项目经理 |
| 创建甘特图 | 8 | 项目经理 |
| 复审甘特图 | 3 | 项目经理、系统分析师、系统设计师 |
| 里程碑 | 0 |  |
| 3 建模 | | |
| 用例分析 | 11 | 系统分析师 |
| 需求确定 | 2 | 系统分析师 |
| 界面原型设计 | 14 | 前端工程师、系统分析师 |
| 接口设计 | 8 | 项目经理、前端工程师、后端工程师 |
| Calendar模块概要设计 | 6 | 系统设计师 |
| User模块概要设计 | 6 | 系统设计师 |
| Event模块概要设计 | 13 | 系统设计师 |
| 数据库设计 | 11 | 数据库管理员、系统设计师 |
| 概要设计文档 | 4 | 系统设计师 |
| Calendar模块详细设计 | 11 | 系统设计师 |
| User模块详细设计 | 11 | 系统设计师 |
| Event模块详细设计 | 16 | 系统设计师 |
| 详细设计文档 | 4 | 系统设计师 |
| 里程碑 | 0 |  |
| 4 构建 | | |
| 服务器搭建 | 2 | 后端工程师、前端工程师 |
| 数据库建立 | 2 | 数据库管理员 |
| 日历界面构建 | 16 | 前端工程师 |
| 会一组成员界面构建 | 16 | 前端工程师 |
| 事件详情界面构建 | 16 | 前端工程师 |
| User模块编码 | 16 | 后端工程师 |
| Calendar模块编码 | 16 | 后端工程师 |
| Event模块编码 | 16 | 后端工程师 |
| 里程碑 | 0 |  |
| 5 测试 | | |
| 单元测试 | 11 | 测试员、前端工程师、后端工程师 |
| 集成测试 | 11 | 测试员、后端工程师 |
| 系统测试 | 11 | 测试员、前端工程师、后端工程师 |
| 验收测试 | 11 | 测试员、项目经理 |
| 测试文档 | 3 | 测试员 |
| 里程碑 |  |  |
| 6 项目上线 | | |
| 用户帮助文档 | 11 | 项目经理、前端工程师、后端工程师 |
| 处理用户问题 | 4 | 项目经理、前端工程师、后端工程师 |
| 里程碑 | 0 |  |

### 2.4.3 调度资源

由于本组成员较少，因此人员分配上限为3人且一个人会兼任多个职位。

回顾各个人员资源的限制条件：

|  |  |
| --- | --- |
| 项目经理 | 1 |
| 系统分析师 | 1 |
| 系统设计师 | 3 |
| 前端工程师 | 2 |
| 后端工程师 | 3 |
| 数据库管理员 | 1 |
| 测试员 | 1 |

项目经理1人

项目经理(Project Manager)是企业中专门负责产品管理的职位，项目经理负责市场调查并根据产品、市场及用户等的需求，确定开发何种产品，选择何种业务模式、商业模式等。并推动相应产品的开发组织，他还要根据产品的生命周期，协调研发、营销、运营等，确定和组织实施相应的产品策略，以及其他一系列相关的产品管理活动。

由于本项目采用Scrum敏捷开发模式，因此项目经理在获取用户需求与团队全部成员进行需求可行性分析与需求评级，确定当前Sprint的开发任务，并进行Sprint工作安排与统筹，对下个Sprint的工作方式等进行控制、优化与提升。

系统分析师1人

系统分析师是根据软件产品需求分析及可行性分析，进行软件开发过程中所有流程与架构的分析。

在本项目中，系统分析师承担Sprint需求会中可行性分析与需求评级的重要工作，并且对服务器后端架构与数据库进行设计、控制与管理。

系统设计师3人

系统设计师是根据软件产品需求分析及可行性分析，进行软件开发过程中所有流程与架构的设计。

在本项目中，系统设计师承担Sprint中对服务器后端架构与数据库进行设计、控制与管理。

前端开发工程师2人

Web前端开发技术是一个先易后难的过程，主要包括三个要素：HTML(标准通用标记语言下的一个应用)、级联样式表CSS和JavaScript。并且，其不仅要掌握基本的Web前端开发技术，网站性能优化、SEO和服务器端的基础知识，而且要学会运用各种工具进行辅助开发以及理论层面的知识，包括代码的可维护性、组件的易用性、分层语义模板和浏览器分级支持等。

在本项目中，两名前端开发工程师共同负责微信小程序前端开发的工作，同时兼顾视图层和前端逻辑层的实现，保证用户可用度的质量。

后端开发工程师3人

后端开发工程师主要负责数据库的维护与服务器后端代码的开发，并对服务端的性能进行优化，以更好地满足安全性、并发性等相关需求。

在本项目中，两名后端开发工程师共同完成服务器后端的开发工作。并且由于项目采用前后端分离的开发模式，后端开发工程师同样需要负责API文档的维护工作，保证与前端开发工程师的沟通。

测试工程师1人

测试工程师负责编写测试计划、规划详细的测试方案、编写测试用例，并根据测试计划搭建和维护测试环境。此外，还需执行测试工作，提交测试报告。包括编写用于测试的自动测试脚本，完整地记录测试结果，编写完整的测试报告等相关的技术文档。

测试工程师还需与开发工程师保持良好沟通，对测试中发现的问题进行详细分析和准确定位，与开发人员讨论缺陷解决方案，提出对产品的进一步改进的建议，并评估改进方案是否合理;对测试结果进行总结与统计分析，对测试进行跟踪，并提出反馈意见。

在本项目中，测试工程师还需参积极参与Sprint交付时的评估会议工作，对当期Sprint交付的迭代产品进行测试，与软件质量管理工程师共同完成当前Spring迭代的质量把控。

数据库管理员1人

负责数据库的分析建模、设计、构建表等，并在项目后期维护数据库。

软件质量保证工程师1人

SQA工程师是软件质量保证工程师。从事软件产品的评审和审计，验证软件是否合乎标准的专业测试人员。软件质量保证(SQA)是软件测试人员建立一套有计划，有系统的方法，来向管理层保证拟定出的标准、步骤、实践和方法能够正确地被所有项目所采用。

软件质量保证的目的是使软件过程对于管理人员来说是可见的。它通过对软件产品和活动进行评审和审计来验证软件是否合乎标准。软件质量保证组在项目开始时就一起参与建立计划、标准和过程。这些将使软件项目满足机构方针的要求。

在本项目中，软件质量保证工程师负责控制软件开发过程的代码质量，进行代码审查(Code Review)，负责对并发性、安全性、用户可用性等软件质量方面进行把控，并在Sprint评审交付时对当前迭代的质量进行把关。

客户代表1人

在本项目中，客户代表负责与目标客户(高校在校生、公司工作人员等)进行接洽，对需求进行初步收集与分析，并负责产品的商业洽谈和市场推广等工作。

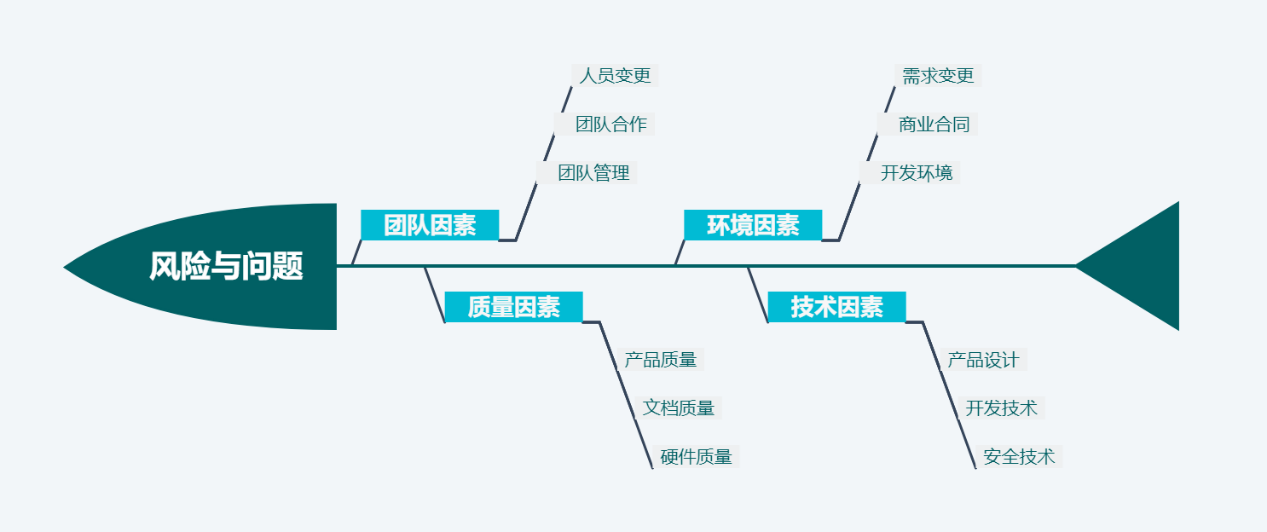
## 2.5 风险管理计划

### 2.5.1 风险辨认

鱼骨图

在风险辨认阶段，为了充分地挖掘出项目过程中可能存在的每一个潜在风险，仅仅将所有成员所想到的风险总结出来是远远不够的，我们需要使用系统化的方法来识别风险，将复杂问题化解为简单问题，从而探究出风险发生的本质。

为了更好地分解问题，我们使用了鱼骨图。作为一种分析方法，鱼骨图能帮助人们通过现象来看本质，也称为因果分析图。在团队讨论和决策过程中，鱼骨图有助于帮助团队成员分析问题。我们首先对可能存在的所有风险的来源进行粗略地分类，分为环境因素、团队因素、技术因素和质量因素四个方面。进一步，风险因素可以进行细化，分解为许多风险项。我们进行头脑风暴来找出这些风险项，并将这些风险项根据相关性清晰排列，得到下面的鱼骨图:



RBS

基于鱼骨图，我们在每个风险因素下进一步考虑并划分出多个风险项，汇总并罗列出所有的风险项。对这些风险项进行评估和筛选，最终抽取出风险概率最大的若干个高风险项。我们使用风险分解结构(RBS，Risk Breakdown Structure)图来更准确地描述潜在风险。风险分解结构图对项目可能面对的潜在风险进行了分类，能更加直观清晰地向我们展示风险的来源和划分，从而帮助我们确定风险。

|  |  |  |
| --- | --- | --- |
| RBS0级 | RBS1级 | RBS2级 |
| 0.项目风险所有来源 | 1.外部不可测风险 | 1.1人为破坏 |
| 1.2政策因素 |
| 2.外部可测风险 | 2.1需求变更 |
| 2.2市场竞争 |
| 3.技术风险 | 3.1系统安全 |
| 3.2低性能 |
| 3.3软件缺陷 |
| 4.内部非技术风险 | 4.1团队合作 |
| 4.2运营维护 |
| 4.3人员管理 |
| 5.法律风险 | 5.1合同漏洞 |

### 2.5.2 风险评估

风险概率

风险概率是一个介于0和1之间的值，表示该风险项目的发生概率。对于不同的风险和风险情况，我们使用适当的定性和定量方法来确定风险概率。风险概率范围被映射到风险概率水平，用于风险水平计算。

|  |  |  |  |
| --- | --- | --- | --- |
| 得分 | 概率范围 | 等级 | 描述 |
| 10 | 超过90% | 极高 | 几乎确定发生 |
| 9 | 80%-90% |
| 8 | 0%-80% | 高 | 很有可能发生 |
| 7 | 60%-70% |
| 6 | 50%-60% | 中 | 可能发生 |
| 5 | 40%-50% |
| 4 | 30%-40% | 低 | 很少可能发生 |
| 3 | 20%-30% |
| 2 | 10%-20% | 极低 | 几乎不发生 |
| 1 | 0%-10% |

风险影响

风险影响是衡量风险影响的指标。对于风险评估，风险影响也很重要。如果风险影响程度很大，即使发生概率很小，也会对项目产生很大影响并造成严重后果。因此，必须严格控制风险。

某些风险影响很容易衡量。例如，成本风险可以通过财务数据来衡量；进度风险可以用天来衡量。有些风险无法通过数字直接衡量，例如技术风险。因此，我们设定影响程度的标准，并通过分析和判断确定某些风险的风险影响。

|  |  |  |
| --- | --- | --- |
| 得分 | 等级 | 描述 |
| 10 | 极高 | 会导致整个项目的失败 |
| 9 |
| 8 | 高 | 会严重影响项目的进行 |
| 7 |
| 6 | 中 | 对项目有一定的影响 |
| 5 |
| 4 | 低 | 对项目有较少的影响 |
| 3 |
| 2 | 极低 | 对项目的影响可以忽略 |
| 1 |

风险等级

风险等级是风险评估工作中获得的最重要的数据。根据风险概率和风险影响程度，通过风险等级根据严重程度对风险进行分类，这是制定和跟进我们的风险应对措施的重要基础。其计算方法是：风险等级=风险概率\*风险影响。

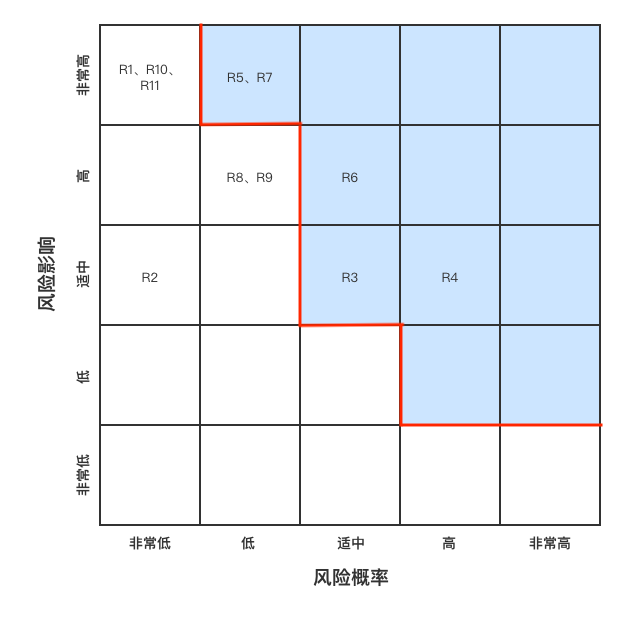
|  |  |  |
| --- | --- | --- |
| 风险等级 | 风险等级描述 | 决策结果 |
| 81-100 | 非常高 | 无法接受的风险 |
| 61-80 | 高 | 不期望的风险，需要指定策略 |
| 41-60 | 适中 | 需要管理人员审查控制风险 |
| 21-40 | 低 | 可接受的风险，无需审查 |
| 1-20 | 非常低 | 完全可接受 |

风险值

根据我们上面定义的风险概率和风险影响，我们可以计算每一个风险项的风险等级，对应到风险等级表里，我们可以找到对应应该采取的措施。针对上述定义的是一个风险项，我们通过商讨和分析确定了风险概率和影响程度，从而计算出风险等级，得到下表:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 风险项 | 风险概率 | 风险影响 | 风险等级 |
| R1 | 人为破坏 | 1 | 9 | 9 |
| R2 | 政策因素 | 2 | 5 | 10 |
| R3 | 需求变更 | 5 | 6 | 30 |
| R4 | 市场竞争 | 8 | 6 | 48 |
| R5 | 系统安全 | 3 | 9 | 27 |
| R6 | 低性能 | 5 | 7 | 35 |
| R7 | 软件缺陷 | 3 | 10 | 30 |
| R8 | 团队合作 | 3 | 8 | 24 |
| R9 | 运营维护 | 4 | 7 | 28 |
| R10 | 人员管理 | 2 | 9 | 18 |
| R11 | 合同漏洞 | 2 | 9 | 18 |

概率-影响度矩阵



### 2.5.3 风险策划

至此我们已经完成了主要风险的识别和分类，下面的工作是计划如何处理他们。我们这里采用的处理方法主要有以下几种：

接受风险

针对一些发生概率较低且影响较弱的风险我们选择忽略这些风险，以便我们将更多的资源投入到发生概率较高、危害较大的风险上。有一些风险，一旦发生后造成的损失要比降低他们发生概率投资的资本要少的多，所以我们就没有理由去提前规避这些风险。

规避风险

一些风险一旦发生就会给项目造成较大的影响，轻则延缓项目进度，重者可能导致整个项目的失败，对于这些风险我们就要提前进行规避，以免发生后造成损失。

缓解风险

缓解风险是针对一些发生概率较高的风险，一旦已经发生，我们只能尽力使得损失降到最低。比如:我们可以通过备份数据库来缓解由于数据库故障而引起的数据丢失的风险。

转移风险

在这种情况下，风险将会被转移到其他的人或者公司那里去。例如，我们可以在项目的一开始就把其中一部分工作以合适的价格外包出去，这样我们就不必处理这部分工作可能引起的风险。此外，对于一些突发的意外事件，比如自然灾害等，我们可以购买保险来转移风险。

## 2.6 监督与控制

### 2.6.1 目的

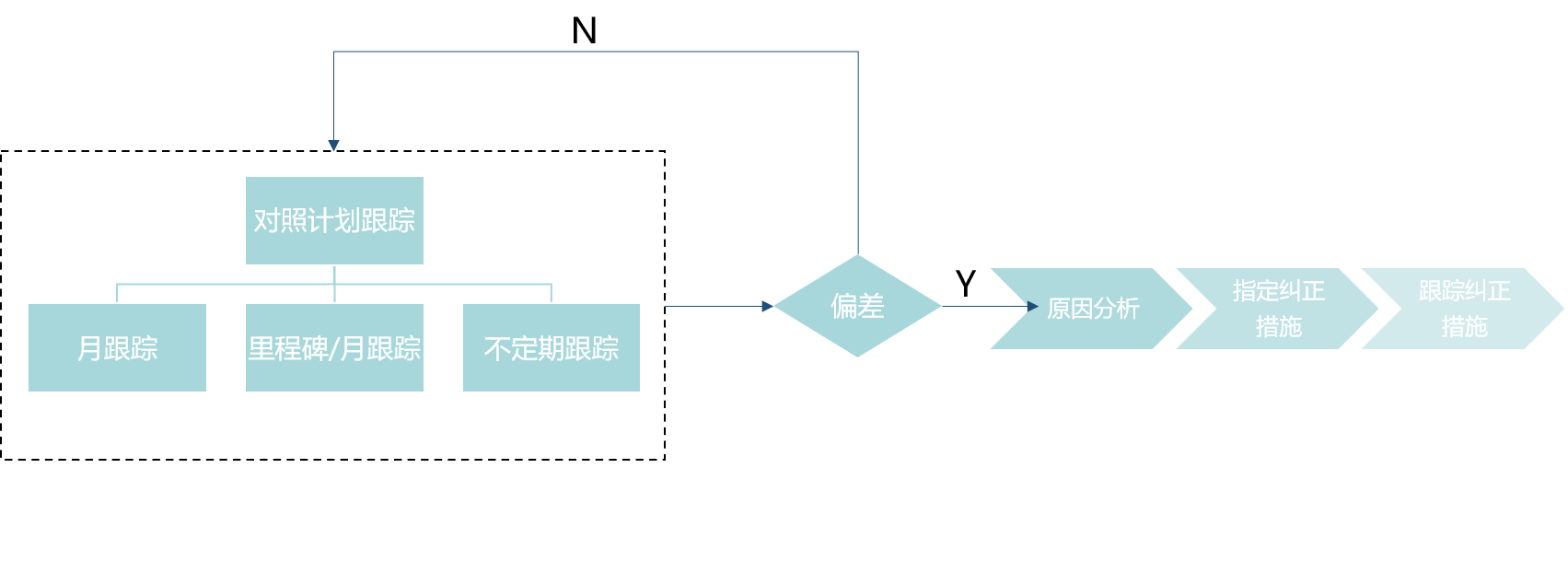
项目监督和控制（Project Monitoring and Control, PMC）的目的是通过周期性地跟踪项目计划的各种性能参数如工作产品的规模、工作量、成本、进度、风险等，不断地了解项目的进展情况，以便当项目实际进展状况显著偏离项目计划时能够及时采取纠正措施。最终目的都是为了使项目按计划交付合格的产品。

项目经理应使用和遵循本过程，对照项目计划监督该项目的实际性能和进展。当项目的性能或结果明显偏离计划时，管理各项纠正措施，直到结束。

### 2.6.2 角色与职责

|  |  |
| --- | --- |
| 角色 | 职责 |
| 项目经理 | * 按照计划执行项目的管理工作 * 对照计划进行项目的监督和控制，进行偏差控制 * 向项目管理部门进行工作汇报 |
| 项目组成员 | * 按照计划开展项目工作 * 编写进度报告，参加例会，向项目经理汇报工作 |
| 项目管理部门 | * 组织里程碑评审 * 跟踪项目经理对项目的监督和控制活动，及时了解项目进展情况。 |
| QA工程师 | * 软件质量保证工作是有计划进行的 * 客观地验证软件项目产品和工作是否遵循恰当的标准、步骤和需求 * 将软件质量保证工作及结果通知给相关组别和个人 |

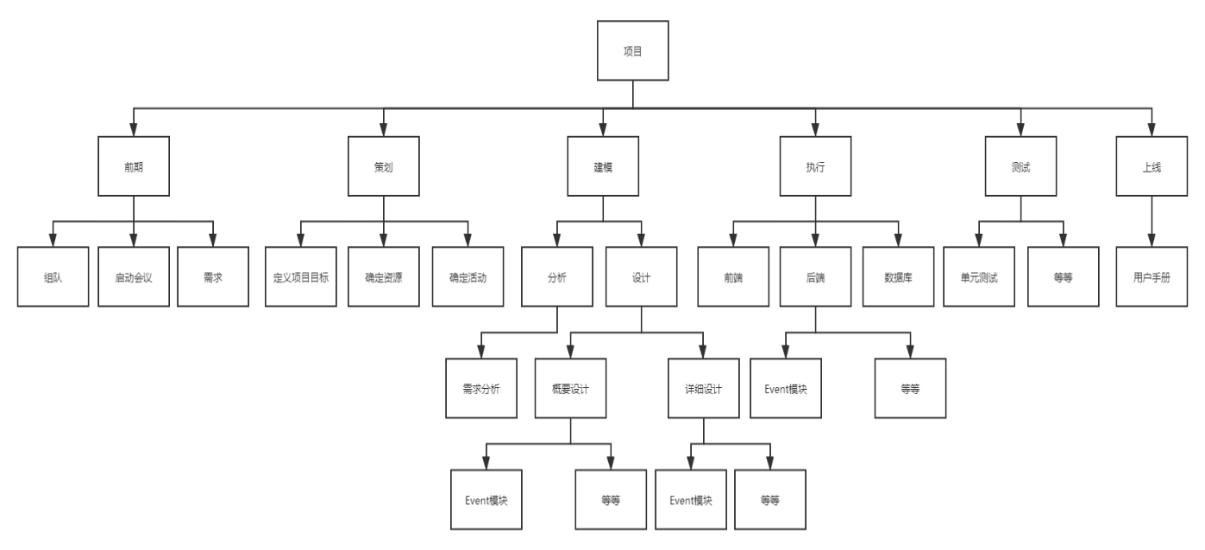
### 2.6.3 流程图



# 3 实施计划

本项目自2020年5月30日开始，最晚于2020年6月27日结束，应尽早完成。项目中产生的交付产物为：需求规约、需求分析规约、系统分析、系统设计、系统测试文档、项目程序客户端和服务器端、用户帮助文档等。项目过程中的里程碑有：项目前期里程碑、项目策划里程碑、建模里程碑、构建里程碑、测试里程碑、上线里程碑。里程碑象征着一阶段的完成，有产出的交付文档或者程序，但在实际中，为了加快进度，一个阶段还未结束，可能另一阶段的任务已经开始并行着手解决。

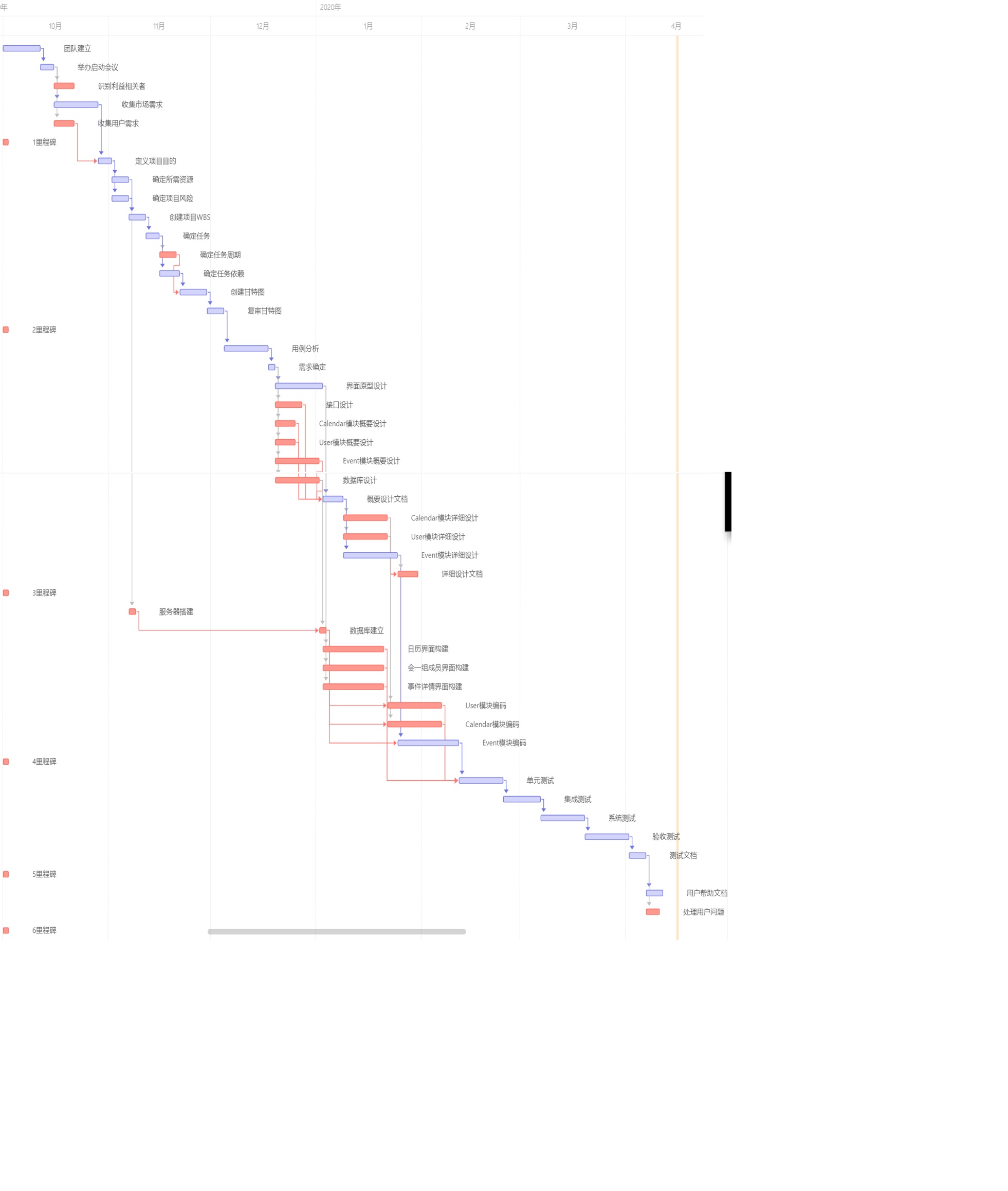
## 3.1 项目WBS



## 3.2 识别任务及依赖

|  |  |  |
| --- | --- | --- |
| 编号 | 活动 | 依赖 |
|  |  |  |
|  | 1 项目策划 |  |
| A | 定义项目目标 | 1DE |
| B | 确定所需资源 | 2A |
| C | 确定项目风险 | 2A |
| D | 创建项目WBS | 2BC |
| E | 确定任务 | 2D |
| F | 确定任务周期 | 2E |
| G | 确定任务依赖 | 2E |
| H | 创建甘特图 | 2FG |
| I | 复审甘特图 | 2H |
|  | 里程碑 |  |
|  | 2 建模 |  |
| A | 用例分析 | 2H |
| B | 需求确定 | 3A |
| C | 界面原型设计 | 3B |
| D | 接口设计 | 3B |
| E | Calendar模块概要设计 | 3B |
| F | User模块概要设计 | 3B |
| G | Event模块概要设计 | 3B |
| H | 数据库设计 | 3B |
| I | 概要设计文档 | 3CDEFGH |
| J | Calendar模块详细设计 | 3I |
| K | User模块详细设计 | 3I |
| L | Event模块详细设计 | 3I |
| M | 详细设计文档 | 3JKL |
|  | 里程碑 |  |
|  | 3 构建 |  |
| A | 服务器搭建 | 2B |
| B | 数据库建立 | 3H4A |
| C | 日历界面构建 | 3C |
| D | 会一组成员界面构建 | 3C |
| E | 事件详情界面构建 | 3C |
| F | User模块编码 | 3K4B |
| G | Calendar模块编码 | 3J4B |
| H | Event模块编码 | 3L4B |
|  | 里程碑 |  |
|  | 4 测试 |  |
| A | 单元测试 | 4CDEFGH |
| B | 集成测试 | 5A |
| C | 系统测试 | 5B |
| D | 验收测试 | 5C |
| E | 测试文档 | 5D |
|  | 里程碑 |  |
|  | 5 项目上线 |  |
| A | 用户帮助文档 |  |
| B | 处理用户问题 |  |
|  | 里程碑 |  |

## 3.3 甘特图



## 3.4 详细计划

|  |  |  |
| --- | --- | --- |
| 任务 | 截止日期 | 负责人员 |
| 系统分析与设计 |  |  |
| 数据库设计 | 5月30日 | 陈东 |
| 登陆注册微信接口模块分析设计 | 5月30日 | 姜其升 |
| 事件模块分析设计 | 5月30日 | 张尧 |
| 日历（会议组）模块分析设计 | 5月30日 | 张尧 |
|  |  |  |
|  |  |  |
| 服务器配置 | 5月31日 | 陈东 |
| 数据库部署 | 5月31日 | 陈东 |
|  |  |  |
| 登录注册微信接口模块 |  |  |
| 用户账户 | 6月10日 | 姜其升 |
| 微信方面用户安全认证 | 6月10日 | 姜其升 |
|  |  |  |
| 事件模块 |  |  |
| 创建个人事件 | 6月6日 | 陈东 |
| 修改个人事件 | 6月7日 | 陈东 |
| 删除个人事件 | 6月7日 | 陈东 |
| 事件创建修改显示前端界面 | 6月10日 | 张尧 |
|  |  |  |
| 日历（会议组）模块 |  |  |
| 用户查看会议组成员事件 | 6月20日 | 陈东 |
| 创建团队事件 | 6月10日 | 陈东 |
| 修改团队事件 | 6月13日 | 陈东 |
| 创建团队日历 | 6月12日 | 姜其升 |
| 解散团队日历 | 6月14日 | 姜其升 |
| 分享团队日历 | 6月16日 | 姜其升 |
| 加入团队日历 | 6月12日 | 姜其升 |
| 退出团队日历 | 6月23日 | 陈东 |
| 设置勿扰模式 | 6月20日 | 姜其升 |
| 推荐组内时间 | 6月23日 | 姜其升 |
| 日历（会议组）显示界面 | 6月17日 | 张尧 |
| 我的主页显示界面 | 6月23日 | 张尧 |
|  |  |  |
|  |  |  |
| 整体调试 | 6月25日 |  |
| 部署 | 6月27日 |  |

# 4 产品

## 4.1 项目产品

程序代码

前端代码

后端代码

数据库配置

项目过程资源

会议记录

页面设计图

项目过程文件等

文档

项目开发计划书

需求分析文档

系统设计文档

## 4.2 运行环境

前端

微信小程序平台

后端

服务器：阿里云服务器

数据库：MySQL、Redis

操作系统：centos

运行容器：Tomcat

## 4.3 服务

后续将根据上线后用户反馈，进行迭代更新。

## 4.4 验收标准

经用户和开发小组双方确认的“需求规格说明书”。 重点确认软件的可靠性、易使用性和功能完整性。