

数据库设计文档

Mesh团队协作任务管理系统

1752985 刘雪迪

1751926 王泽钜

1751445 吕麒远

1851911 张思璇

1752459 曾 泽

1852036 熊章词

1851346 黄子君

1751047 蒋金峰

1753764 韩 嵩

1751466 王新宇

**目录**

[1 系统设计 1](#_Toc51442842)

[1.1 架构设计 1](#_Toc51442843)

[1.2 用户管理 2](#_Toc51442844)

[1.2.1 用户注册 2](#_Toc51442845)

[1.2.2 用户登录 3](#_Toc51442846)

[1.2.3 用户权限判断 4](#_Toc51442847)

[1.2.4 用户信息增删、修改 4](#_Toc51442848)

[1.2.5 用户信息查询 5](#_Toc51442849)

[1.2.6 用户偏好设置 5](#_Toc51442850)

[1.3 团队管理 6](#_Toc51442851)

[1.3.1 创建团队 6](#_Toc51442852)

[1.3.2 邀请团队成员 7](#_Toc51442853)

[1.3.3 查看团队成员 7](#_Toc51442854)

[1.3.4 团队日程管理 8](#_Toc51442855)

[1.3.5 团队知识库管理 8](#_Toc51442856)

[1.3.6 删除团队 9](#_Toc51442857)

[1.4 项目管理 9](#_Toc51442858)

[1.4.1 团队管理员创建项目 9](#_Toc51442859)

[1.4.2 项目管理员修改项目属性 10](#_Toc51442860)

[1.4.3 项目管理员管理项目成员 11](#_Toc51442861)

[1.4.4 项目管理员删除项目 12](#_Toc51442862)

[1.4.5 查看项目详情 12](#_Toc51442863)

[1.4.6 搜索项目 13](#_Toc51442864)

[1.4.7 上传项目知识库内容 13](#_Toc51442865)

[1.4.8 查看项目知识库内容 14](#_Toc51442866)

[1.4.9 项目管理员/上传者删除项目知识库内容 15](#_Toc51442867)

[1.4.10 项目管理员/上传者修改项目知识库内容 16](#_Toc51442868)

[1.4.11 查看数据统计 17](#_Toc51442869)

[1.5 任务管理 18](#_Toc51442870)

[1.5.1 项目管理员创建任务 18](#_Toc51442871)

[1.5.2 项目管理员修改任务详情 18](#_Toc51442872)

[1.5.3 项目管理员添加子任务 19](#_Toc51442873)

[1.5.4 任务负责人/项目管理员完成任务 22](#_Toc51442874)

[1.5.5 使用筛选器创建任务视图 22](#_Toc51442875)

[1.5.6 查看任务详情 23](#_Toc51442876)

[1.5.7 查看数据统计 23](#_Toc51442877)

[1.6 消息功能 24](#_Toc51442878)

[1.6.1 用户接受消息提醒 24](#_Toc51442879)

[1.6.2 查看消息提醒 25](#_Toc51442880)

[1.6.3 查看消息详情 26](#_Toc51442881)

[1.7 公告功能 28](#_Toc51442882)

[1.7.1 项目管理员发布项目公告 28](#_Toc51442883)

[1.7.2 项目管理员删除项目公告 28](#_Toc51442884)

[1.7.3 查看公告详情 29](#_Toc51442885)

[1.7.4 搜索项目公告 30](#_Toc51442886)

[1.8 工具箱功能需求 31](#_Toc51442887)

[1.8.1 计算器 31](#_Toc51442888)

[1.8.2 图标设计 31](#_Toc51442889)

[1.8.3 文本记事 32](#_Toc51442890)

[1.8.4 流程图绘制 33](#_Toc51442891)

[1.8.5 排版绘制 34](#_Toc51442892)

[1.9 组织结构 35](#_Toc51442893)

[2 概念设计 36](#_Toc51442894)

[2.1 用户管理系统实现 36](#_Toc51442895)

[2.1.1 用户登录系统 36](#_Toc51442896)

[2.1.2 用户注册系统 38](#_Toc51442897)

[2.1.3 用户设置系统 40](#_Toc51442898)

[2.2 团队管理系统实现 44](#_Toc51442899)

[2.2.1 创建新团队系统 44](#_Toc51442900)

[2.2.2 团队知识库系统 56](#_Toc51442901)

[2.3 项目管理系统实现 62](#_Toc51442902)

[2.4 任务管理系统实现 72](#_Toc51442903)

[2.4.1 任务管理系统 72](#_Toc51442904)

[2.4.2 子任务管理系统 76](#_Toc51442905)

[2.5 消息管理系统实现 78](#_Toc51442906)

[2.5.1 查看消息 79](#_Toc51442907)

[2.6 公告管理系统实现 81](#_Toc51442908)

[2.6.1 发布公告 81](#_Toc51442909)

[2.6.2 删除公告 85](#_Toc51442910)

[2.6.3 搜索公告 87](#_Toc51442911)

[2.7 工具箱的实现 88](#_Toc51442912)

[3 逻辑设计 90](#_Toc51442913)

[3.1 总体E-R图 91](#_Toc51442914)

[3.2 模块E-R图 91](#_Toc51442915)

[3.2.1 团队管理模块E-R图 91](#_Toc51442916)

[3.2.2 项目管理模块E-R图 93](#_Toc51442917)

[3.2.3 任务管理模块E-R图 94](#_Toc51442918)

[3.2.4 公告管理模块E-R图 96](#_Toc51442919)

[3.2.5 消息管理模块E-R图 97](#_Toc51442920)

[3.3 数据库表的设计 98](#_Toc51442921)

[3.3.1 用户表 98](#_Toc51442922)

[3.3.2 任务表 98](#_Toc51442923)

[3.3.3 子任务表 99](#_Toc51442924)

[3.3.4 任务标签表 99](#_Toc51442925)

[3.3.5 团队表 99](#_Toc51442926)

[3.3.6 项目表 101](#_Toc51442927)

[3.3.7 公告表 101](#_Toc51442928)

[3.3.8 参与表 101](#_Toc51442929)

[3.3.9 公告栏表 102](#_Toc51442930)

[3.3.10 任务栏表 102](#_Toc51442931)

[3.3.11 任务推送表 102](#_Toc51442932)

[3.3.12 分配表 103](#_Toc51442933)

[3.3.13 公告推送表 103](#_Toc51442934)

[3.3.14 管理员表 103](#_Toc51442935)

[3.3.15 团队知识库表 104](#_Toc51442936)

[3.3.16 项目知识库表 104](#_Toc51442937)

[3.3.17 团队知识库条目表 104](#_Toc51442938)

[3.3.18 项目知识库条目表 105](#_Toc51442939)

[3.3.19 备注 105](#_Toc51442940)

[3.4 数据库关系图 105](#_Toc51442941)

[4 API文档 106](#_Toc51442942)

[4.1 用户功能 106](#_Toc51442943)

[4.1.1 用户登录 106](#_Toc51442944)

[4.1.2 用户注册 109](#_Toc51442945)

[4.1.3 查询用户 112](#_Toc51442946)

[4.1.4 用户修改个人信息 114](#_Toc51442947)

[4.1.5 用户修改密码 117](#_Toc51442948)

[4.1.6 管理员修改普通用户密码 118](#_Toc51442949)

[4.2 团队功能 119](#_Toc51442950)

[4.2.1 以当前用户为团队管理员创建一个新团队 119](#_Toc51442951)

[4.2.2 团队管理员邀请新成员加入该团队 121](#_Toc51442952)

[4.2.3 用户加入某个团队(必须是被团队管理员邀请过的) 122](#_Toc51442953)

[4.2.4 查询当前项目信息 124](#_Toc51442954)

[4.3 任务管理 126](#_Toc51442955)

[4.3.1 创建子任务 126](#_Toc51442956)

[4.3.2 创建任务 128](#_Toc51442957)

[4.3.3 删除任务 131](#_Toc51442958)

[4.3.4 删除任务 132](#_Toc51442959)

[4.3.5 查询一个项目下的任务 133](#_Toc51442960)

[4.3.6 查询一个团队下的任务 135](#_Toc51442961)

[4.3.7 更新子任务 137](#_Toc51442962)

[4.3.8 更新任务 139](#_Toc51442963)

[4.4 统计管理 142](#_Toc51442964)

[4.4.1 查询总体统计数据 142](#_Toc51442965)

[4.4.2 查询用户数量 143](#_Toc51442966)

[4.4.3 查询用户 144](#_Toc51442967)

[4.4.4 查询用户统计数据 147](#_Toc51442968)

[4.5 项目管理 149](#_Toc51442969)

[4.5.1 创建项目 149](#_Toc51442970)

[4.5.2 删除项目 151](#_Toc51442971)

[4.5.3 邀请新成员 152](#_Toc51442972)

[4.5.4 用户加入项目 153](#_Toc51442973)

[4.5.5 查询用户项目 155](#_Toc51442974)

[4.5.6 更新项目 157](#_Toc51442975)

[4.6 通知管理 159](#_Toc51442976)

[4.6.1 删除通知 159](#_Toc51442977)

[4.6.2 完成通知 160](#_Toc51442978)

[4.6.3 请求通知 162](#_Toc51442979)

[4.7 公告管理 164](#_Toc51442980)

[4.7.1 新增公告 164](#_Toc51442981)

[4.7.2 删除公告 166](#_Toc51442982)

[4.7.3 查看公告 167](#_Toc51442983)

[4.7.4 更新公告 169](#_Toc51442984)

[附录A：图表索引 171](#_Toc51442985)

1. 系统设计

本团队协作任务管理系统规划管理工作项目，轻松管理团队协作中涉及的各种任务安排。功能覆盖团队管理、项目管理、任务管理、日程、在线知识库、公告、数据统计等应用，满足各行业各场景的个性化办公管理需求。根据需要在现有基础上进行可行性拓展。能有效提升团队协作效率。

* 1. 架构设计

本项目的前端部分使用了Vue作为基本框架。Vue是基于MVC模式的渐进式架构，在MVC模式下，视图和模型得到有效分离，使得我们可以根据路由地址动态地刷新界面或组件。同时，利用Vuex进行状态管理，确保公共数据的一致性，数据的绑定更新会自动驱动视图进行刷新。Vuex提供了定义和封装异步函数的新方式，方便调用。

服务器基于ASP.NETCore3.1模型实现。前端通过axios向其发送http/https请求。入口文件进行json化的统一解析后分别调用不同的控制器文件（Controller），在其中进行逻辑的梳理，通过EntityFrameworkCore使用linq表达式转化为SQL语句访问数据库，取得或修改相应数据，层层返回。

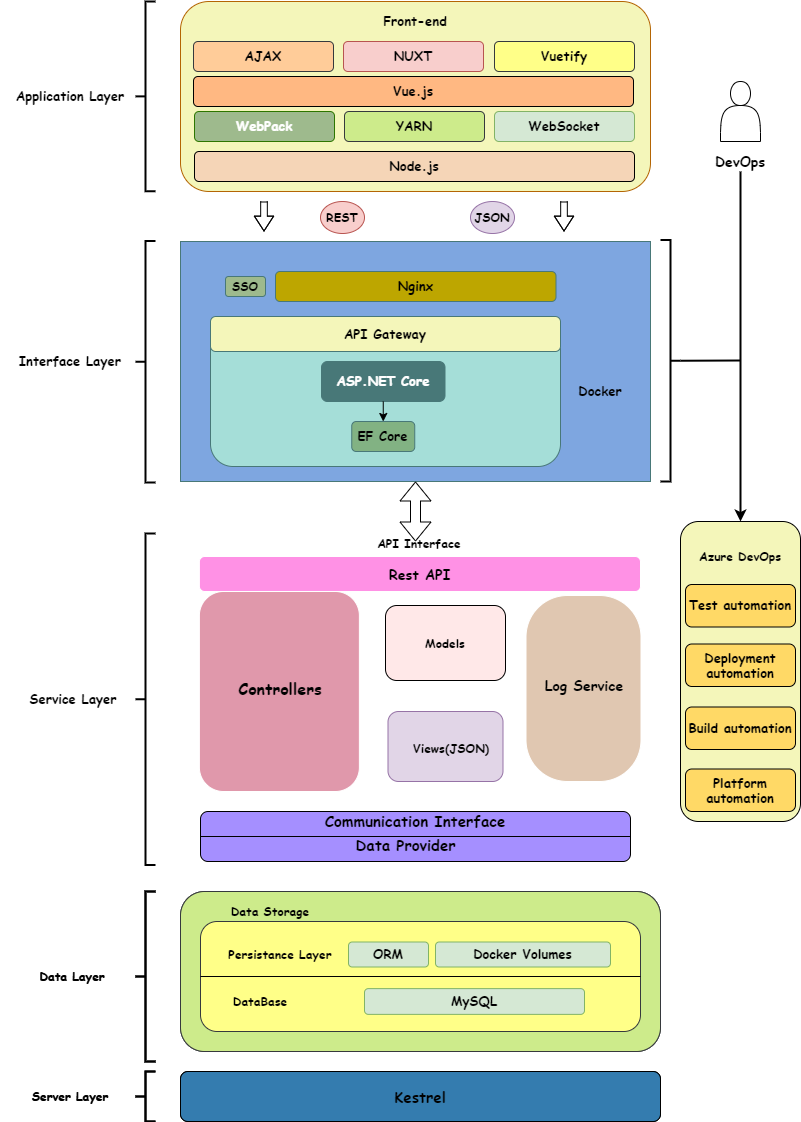


图1-1 项目架构图

* 1. 用户管理

用户注册

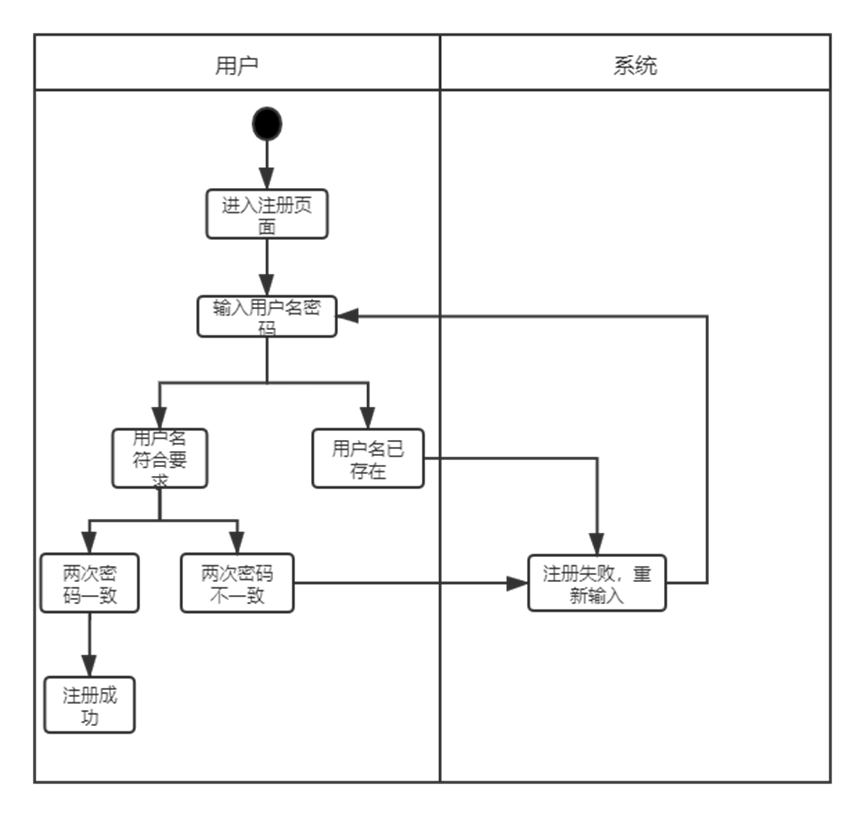


图1-2 用户注册流程

根据用户的注册信息，在普通用户表中查找对应的用户名是否已经存在。如果不存在，则允许注册。用户填写个人信息提交后，系统生成新ID，并在用户表中加入新元组。

用户登录

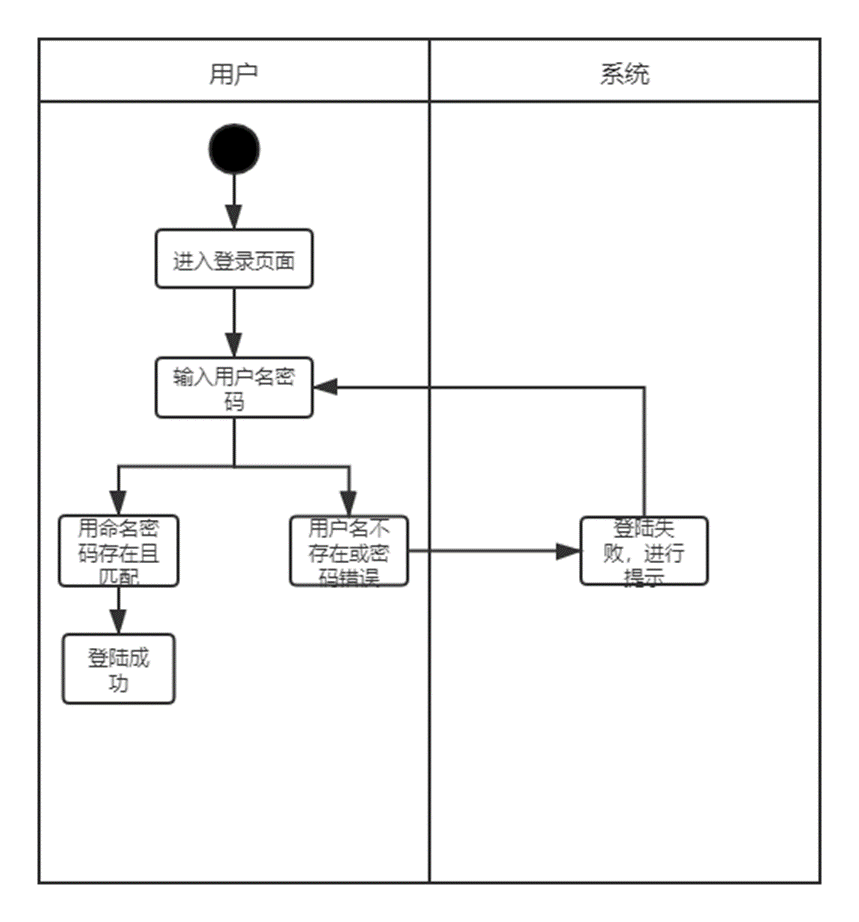


图1-3 用户登录流程

根据用户的登录信息，查询用户表，若ID和密码匹配成功，则登录成功。

用户权限判断

用户成功登录后，根据不同的用户类型，查询用户表中的用户权限，得到不同的用户权限。

用户信息增删、修改

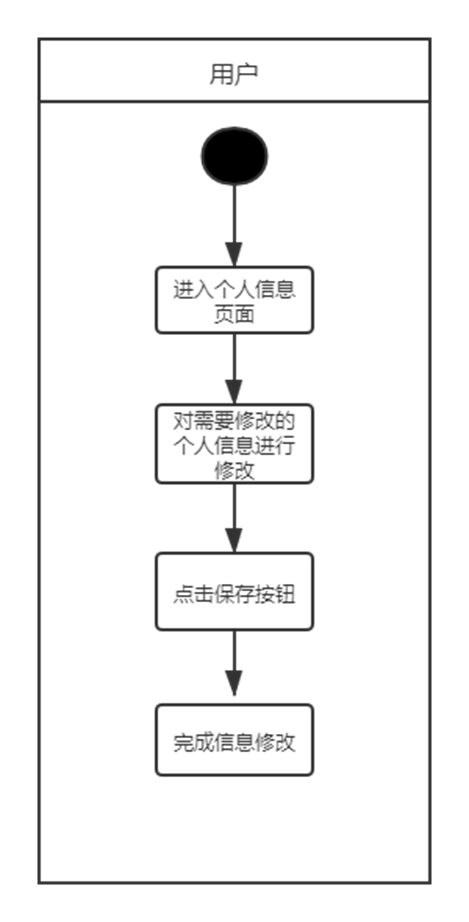


图1-4 用户信息增删、修改流程

用户可以修改自己的昵称信息。超级管理员也可以修改或删除已存在的用户信息。

用户信息查询

超级管理员可以查询指定用户的所有信息。

* + 1. 用户偏好设置

用户可以设置自己偏好的颜色主题、布局、和展示方式，系统会记录用户的偏好和最近常用的项目。

* 1. 团队管理

创建团队

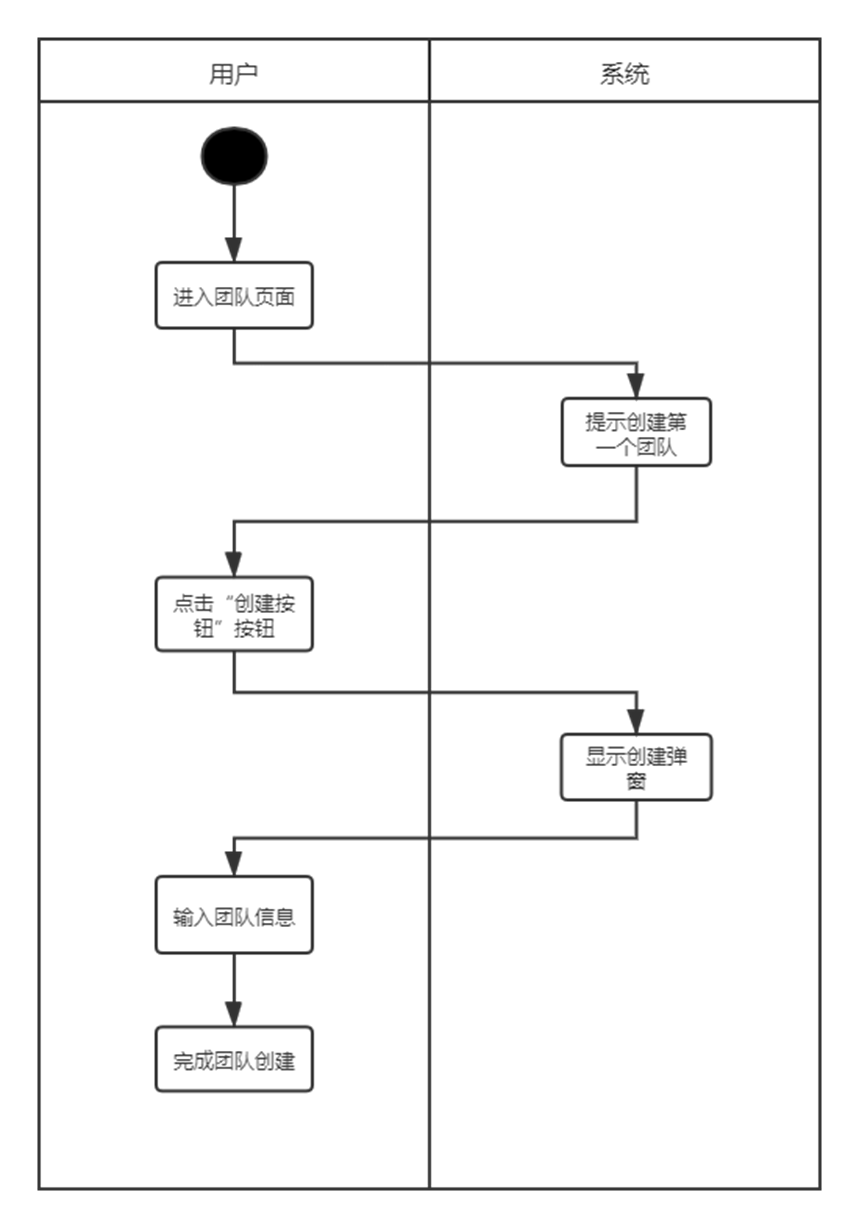


图1-5 创建团队流程

根据用户输入的信息创建团队，该用户自动成为团队的创建者（团队管理员）。

邀请团队成员

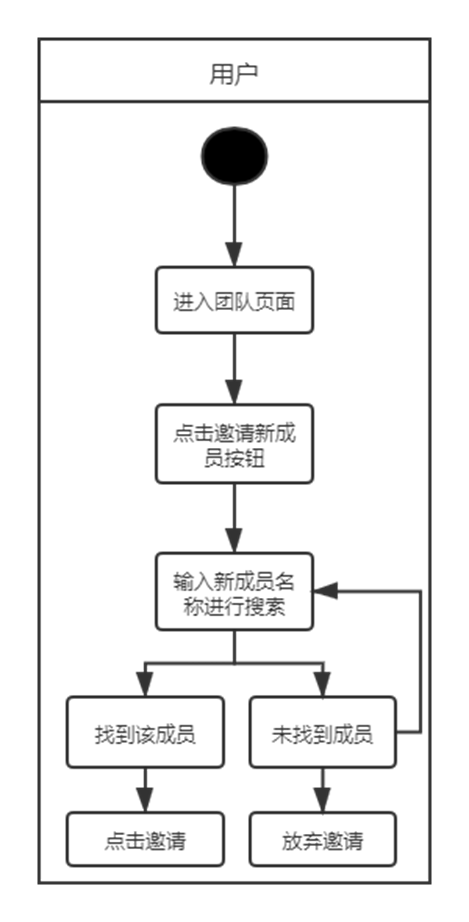


图1-6 邀请团队成员流程

团队管理员为团队添加团队成员。

查看团队成员

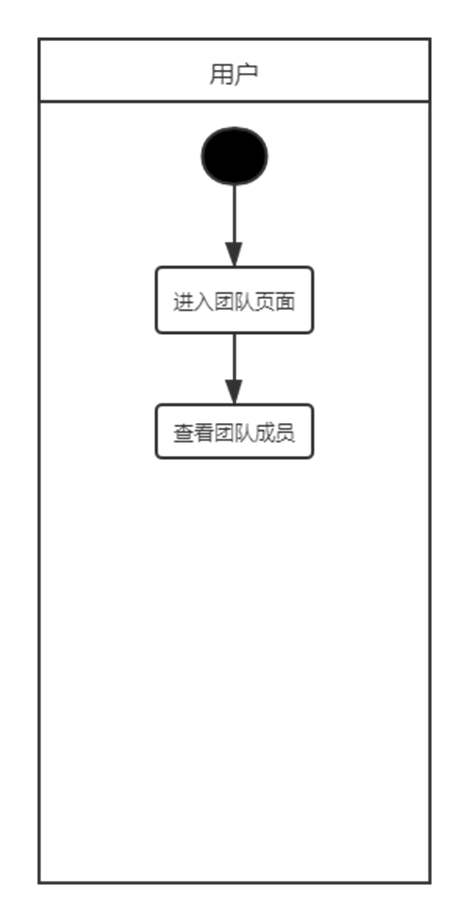


图1-7 查看团队成员流程

用户在团队信息页面查看团队成员，团队管理员有特殊标识。

团队日程管理

团队成员可以在日程页面查看该用户参与的所有项目的所有任务的名称、所属项目、完成情况、时间跨度、负责人、优先级信息。并支持任务按照所属项目、是否完成、负责人是否为该用户筛选显示的任务。

团队知识库管理

团队成员可以在团队知识库页面查看该团队的知识库中所有知识。知识显示知识标题、创建人、创建时间，点击标题可以跳转至外链。

团队所有人可以浏览知识库，并添加知识信息。每一条知识的创建者和团队管理员有权限修改或删除知识。

删除团队

团队管理员可以删除团队。与该团队相关的所有信息将被删除。

* 1. 项目管理

团队管理员创建项目

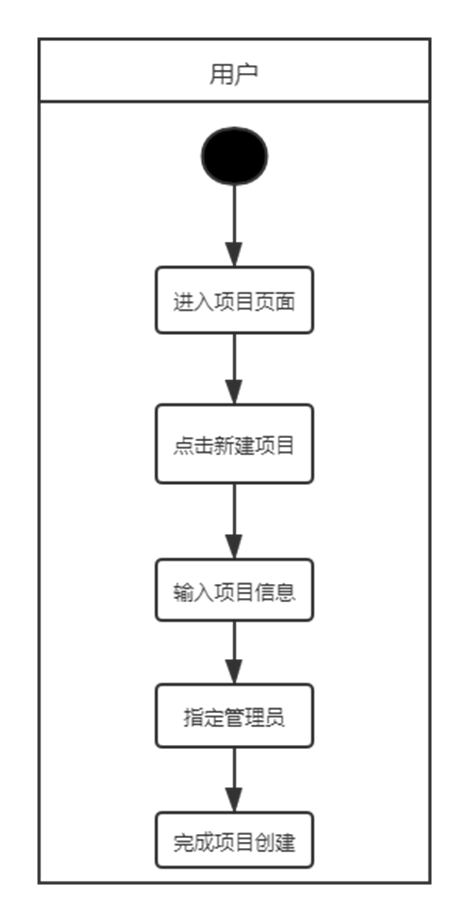


图1-8 创建项目流程

团队管理员创建一个项目，根据团队管理员输入的信息设置项目属性，并在创建的同时分配项目管理员。

项目管理员修改项目属性

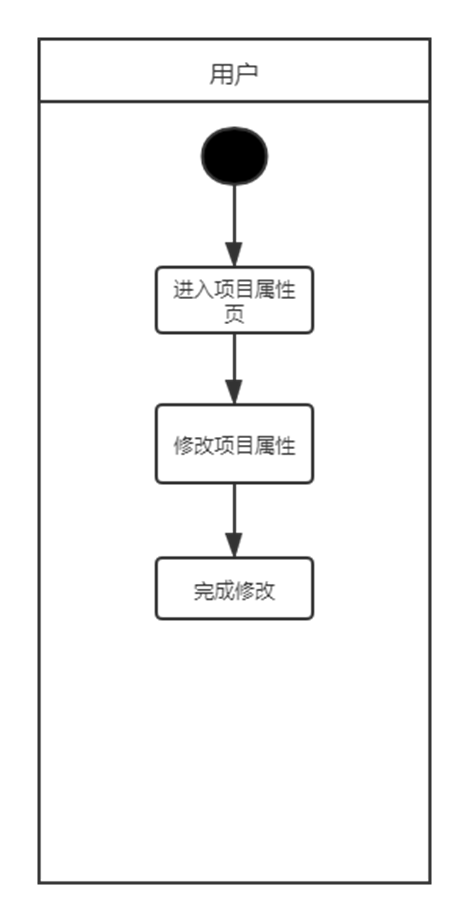


图1-9 修改项目属性流程

项目管理员可以在项目属性修改页修改特定的项目信息，如项目名称、描述、公私有属性等。

项目管理员管理项目成员

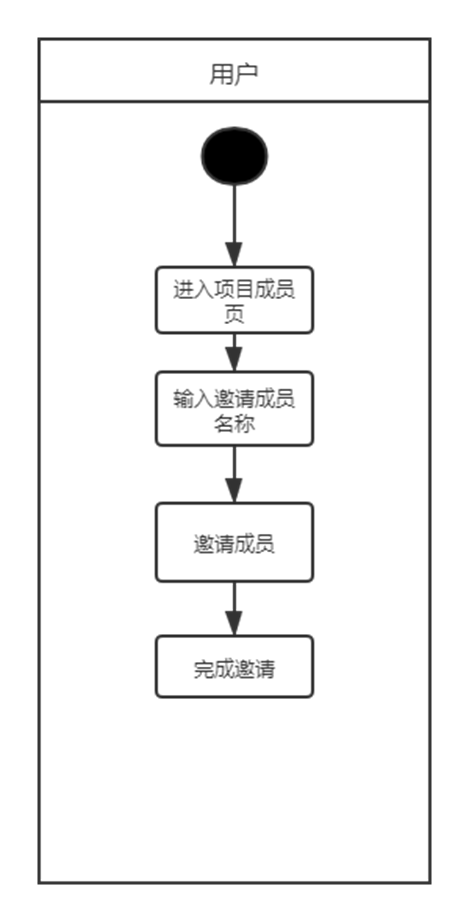


图1-9 管理项目成员流程

项目管理员可以管理项目成员，即邀请项目外人员加入项目或踢出项目成员。

项目管理员删除项目

项目管理员可以删除特定的项目，与该项目相关的所有内容将被删除。

查看项目详情

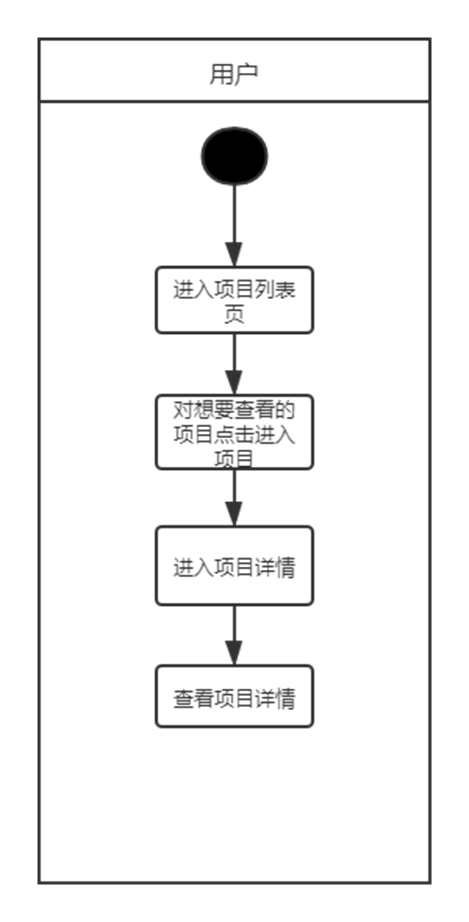


图1-10 查看项目详情流程

用户可以在项目列表里选择特定的项目，项目列表显示的项目顺序与用户偏好有关。用户可以查看详细信息，如项目描述、公私有属性、项目成员等。

搜索项目

根据用户在搜索栏中输入的关键词，检索出相应的项目，默认情况下按时间排序。

上传项目知识库内容



图1-11 向项目知识库上传知识流程

用户可以向项目知识库上传知识。知识以外链的形式关联于知识标题。

查看项目知识库内容

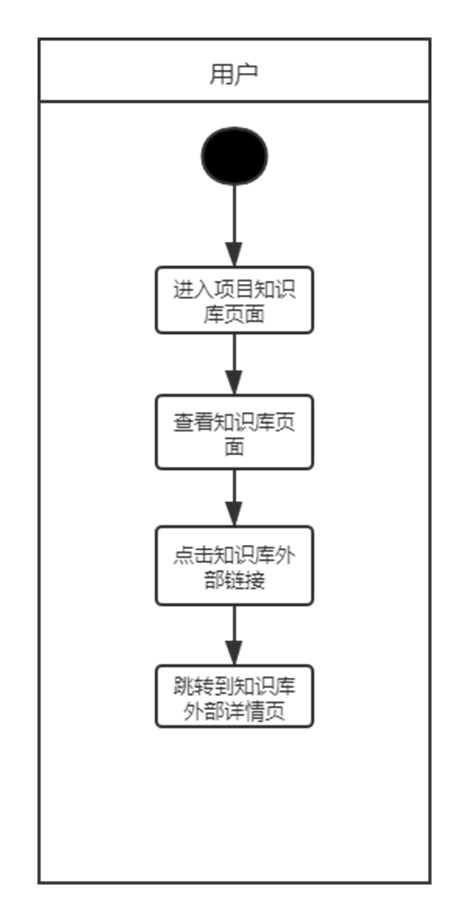


图1-12 查看项目知识库内容流程

用户可以点击知识标题，通过链接跳转到知识的详细内容。

项目管理员/上传者删除项目知识库内容



图1-13 删除项目知识库内容流程

项目管理员可以删除项目知识库中的所有知识。上传者可以删除自己上传的知识。

项目管理员/上传者修改项目知识库内容

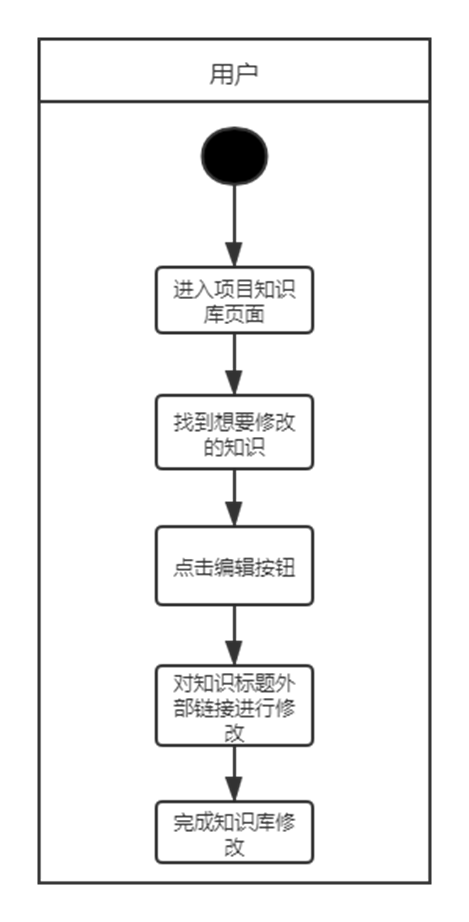


图1-14 修改项目知识库内容流程

项目管理员可以修改项目知识库中的所有知识的内容。上传者可以修改自己上传的知识的内容。

* + 1. 查看数据统计

用户在数据统计面板查看可视化的统计数据展示。并按照用户需要的维度筛选、调整数据展示。

* 1. 任务管理

项目管理员创建任务

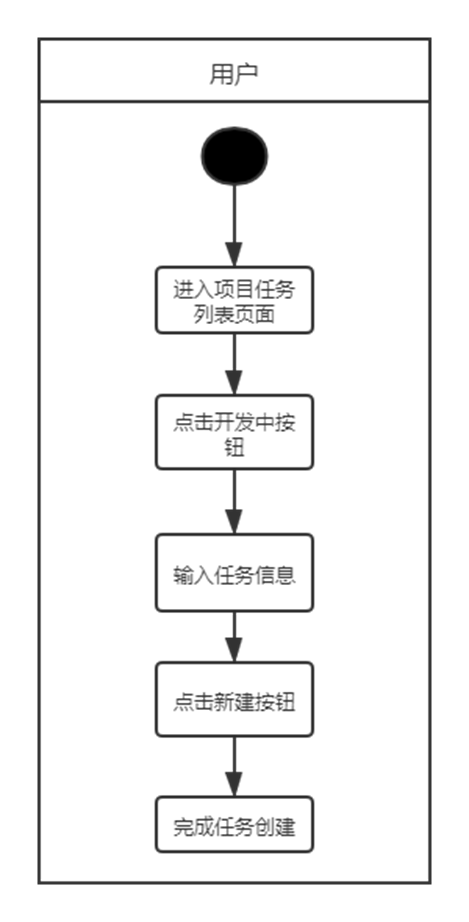


图1-15 创建任务流程

根据项目管理员输入的任务名称、具体描述、优先级、当前状态（开发中，已上线）、截止日期、负责人等信息创建任务，并更改当前任务视图。

项目管理员修改任务详情

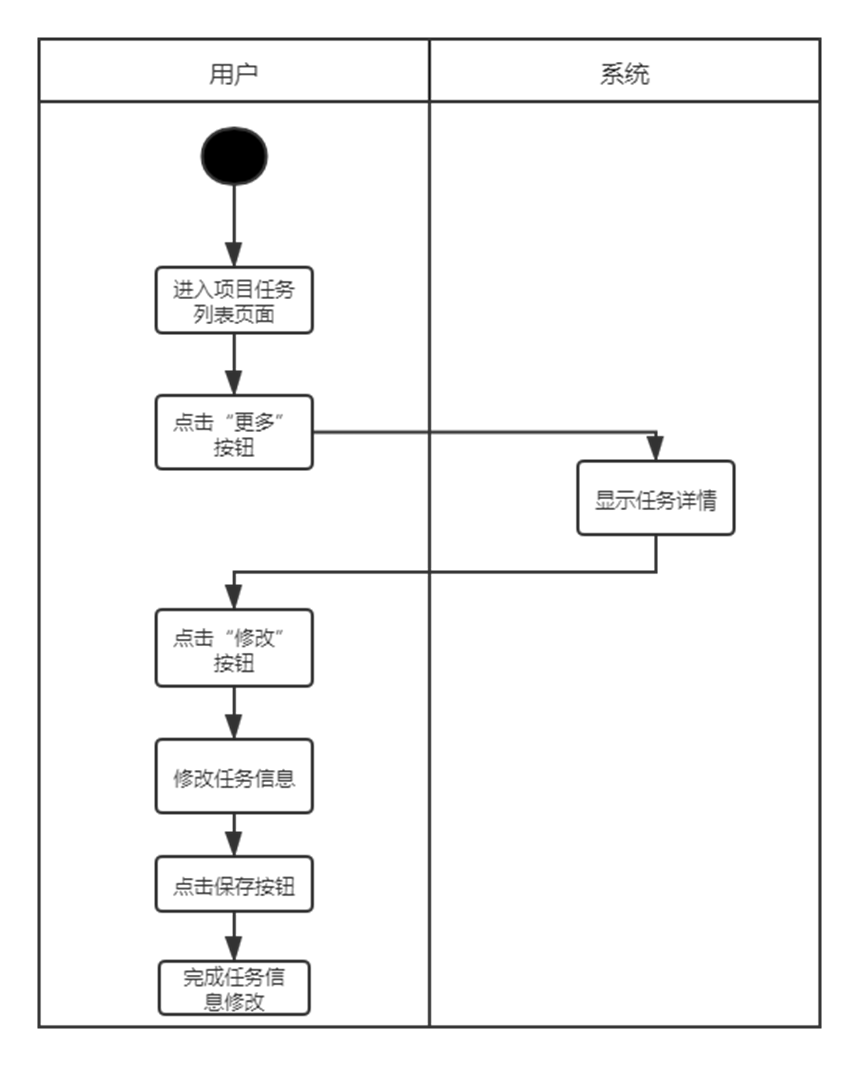


图1-16 修改任务详情流程

项目管理员可以选择特定的任务，修改其标题、状态、优先级、截止日期、负责人等信息。

项目管理员添加子任务

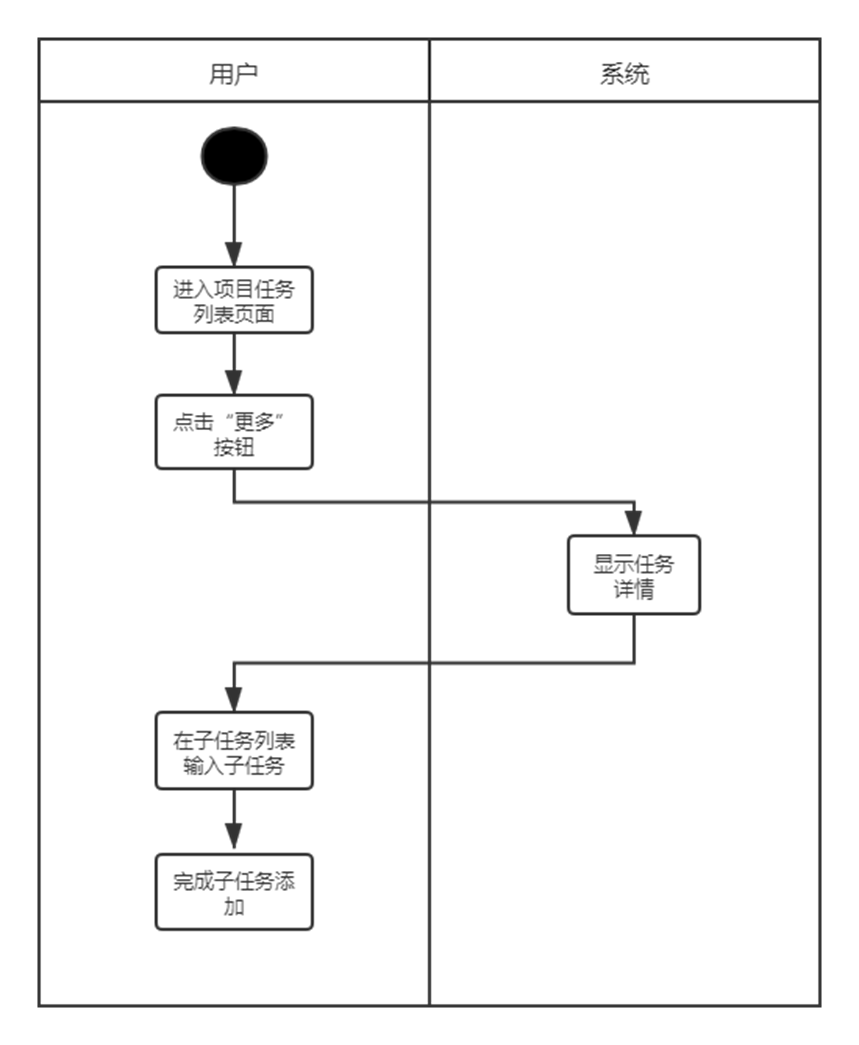


图1-17 添加子任务流程

项目管理员可以选择特定的任务，为其添加子任务，子任务信息包含名称、负责人以及截止日期。

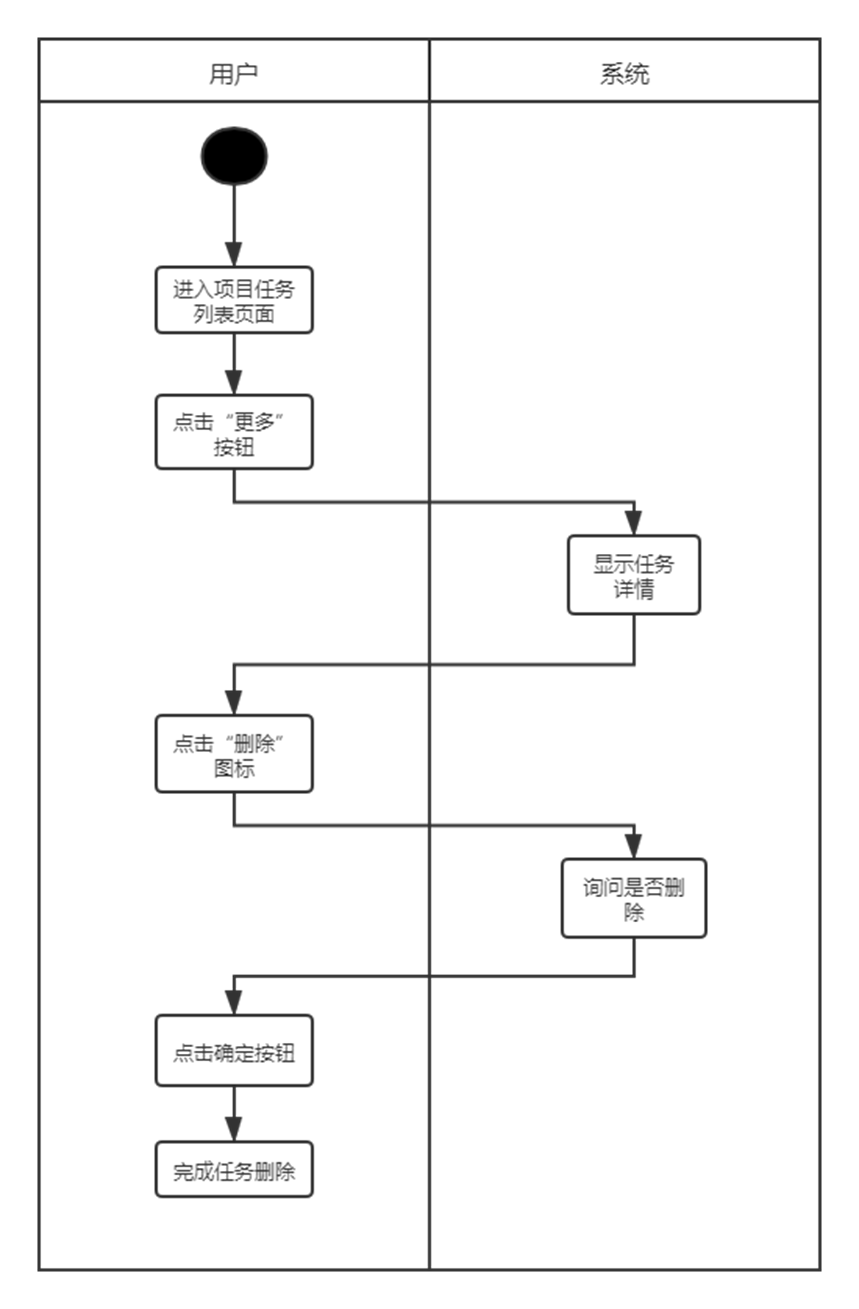
项目管理员删除任务

图1-18 删除任务流程

删除项目管理员选择的特定任务，并更新当前任务视图。

任务负责人/项目管理员完成任务

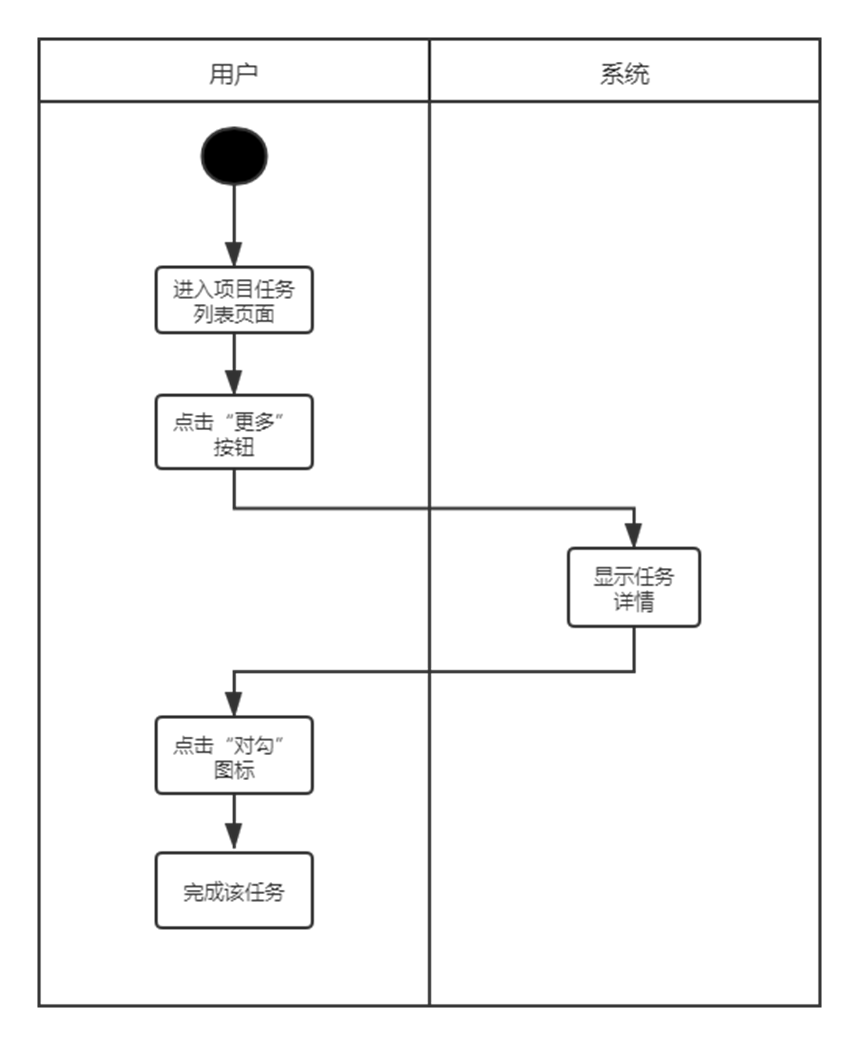


图1-19 完成任务流程

任务负责人/项目管理员可以选择特定的任务，将其标记为完成。

使用筛选器创建任务视图

根据用户需求创建不同视图：未完成任务/所有任务，我负责的/所有任务。分组形式如下：状态/负责人/优先级。

查看任务详情

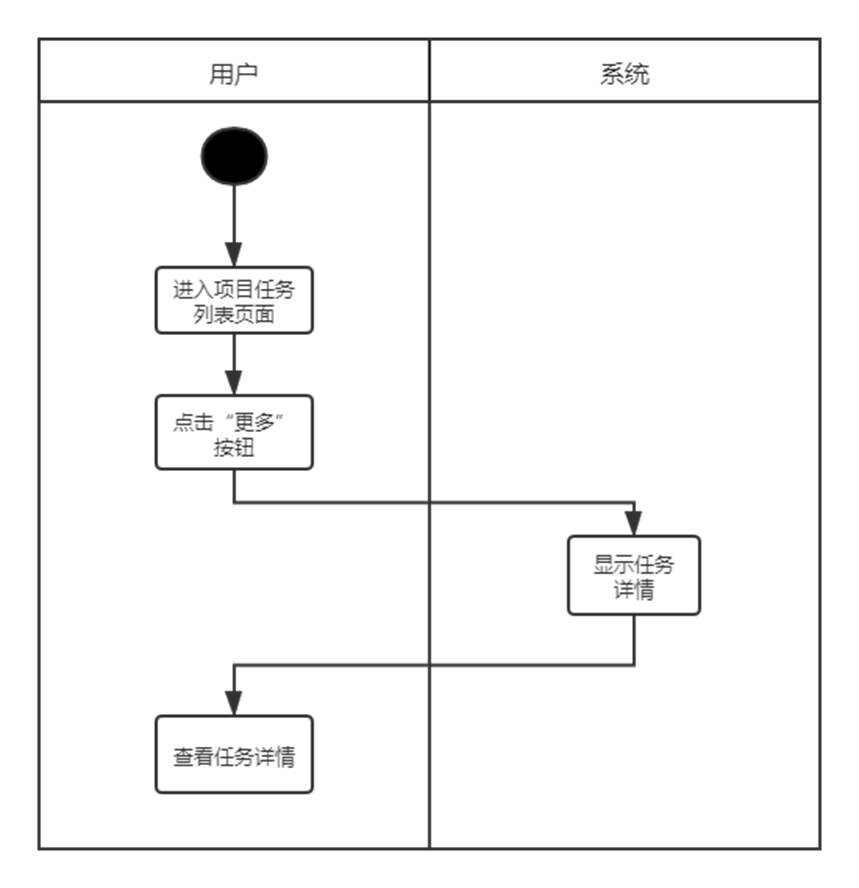


图1-20 查看任务详情流程

用户可以选择特定的任务，查看其详细信息，如状态、优先级、截止日期、负责人等。

查看数据统计

用户在数据统计面板查看可视化的统计数据展示。并按照用户需要的维度筛选、调整数据展示。

* 1. 消息功能

用户接受消息提醒

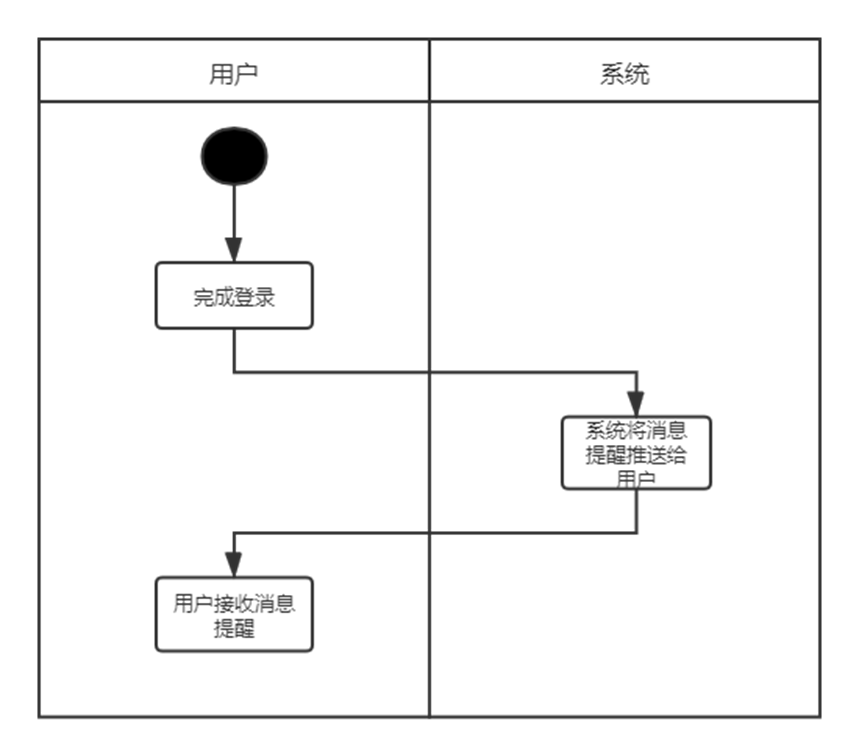


图1-21 接收消息流程

用户在以下情况会收到消息提醒：

1. 被团队管理员授予项目管理员
2. 团队新增成员
3. 项目新增成员
4. 被新团队邀请
5. 被分配任务或子任务
6. 负责的任务或子任务被项目管理员修改
7. 负责的任务临近或超出截止时间
8. 创建的知识被管理员修改或删除
9. 被公告通知

项目管理员在以下情况会收到消息提醒：

1. 分配的任务或子任务被负责人完成
2. 分配的任务或子任务在未完成且已到截止时间

查看消息提醒

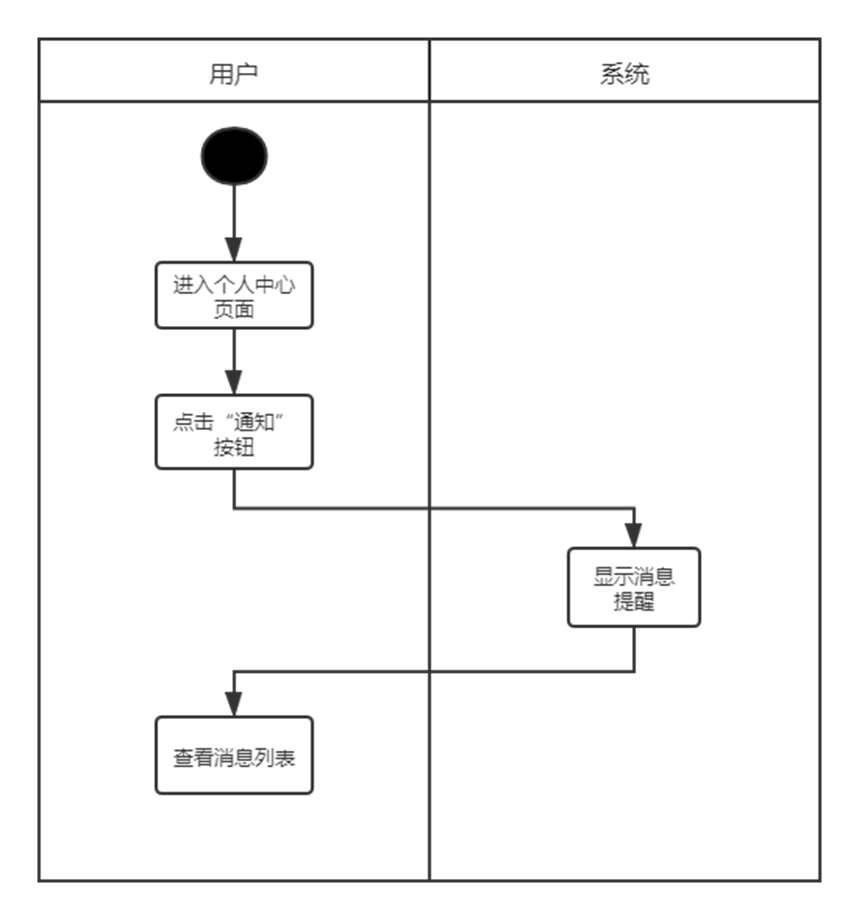


图1-22 查看消息提醒流程

当接受到与自己有关的消息时，用户会收到通知。用户可以打开消息列表查看。

查看消息详情

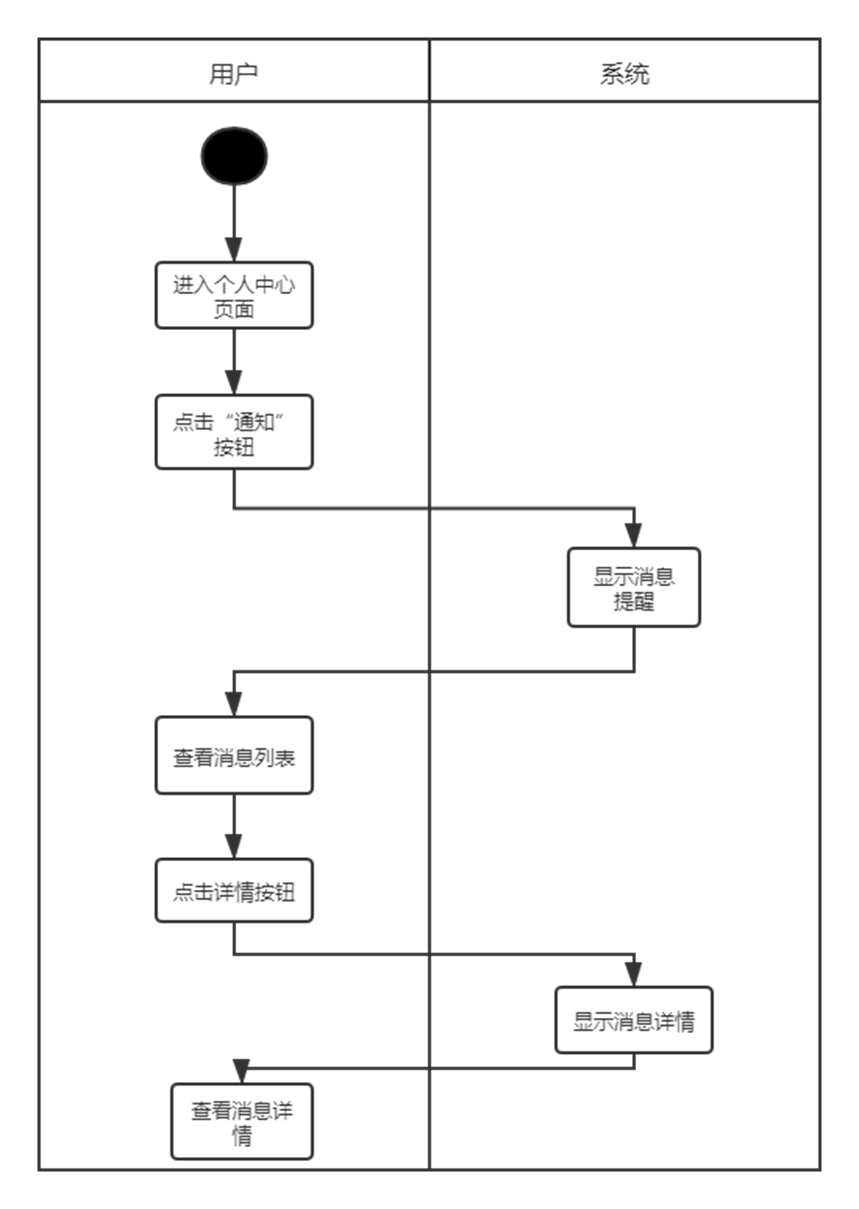


图1-23 查看消息详情流程

用户在消息列表可以查看消息详情。

* 1. 公告功能

项目管理员发布项目公告

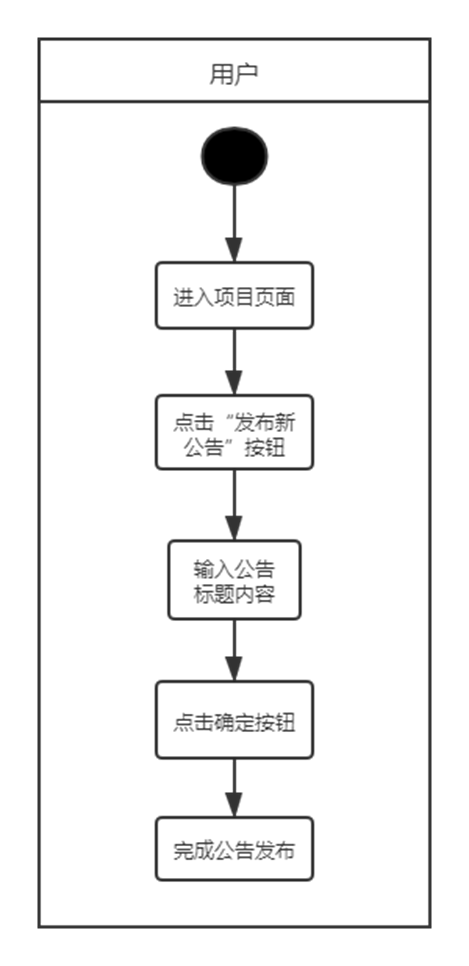


图1-24 发布项目公告流程

项目管理员可以发布项目公告。发布公告可以设置权限，公告会向拥有相应权限的成员发送通知。

项目管理员删除项目公告

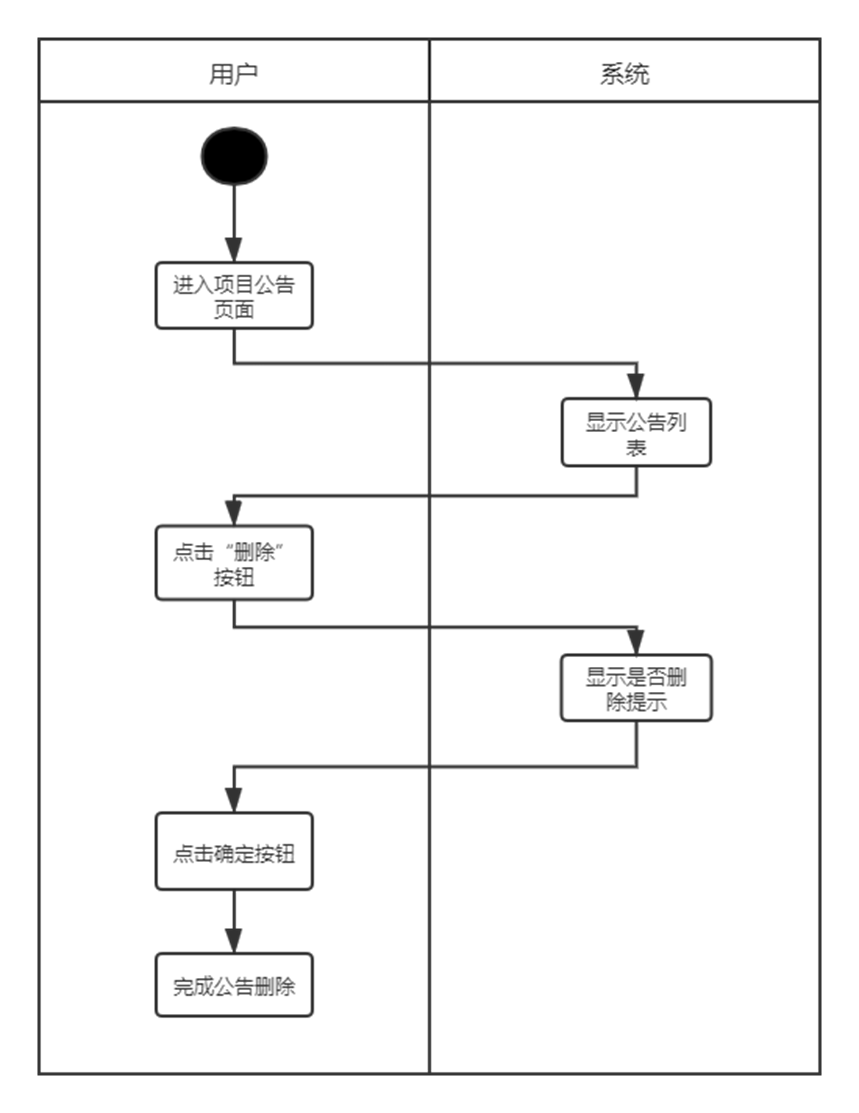


图1-25 删除项目公告流程

项目管理员可以删除指定的项目公告。

查看公告详情

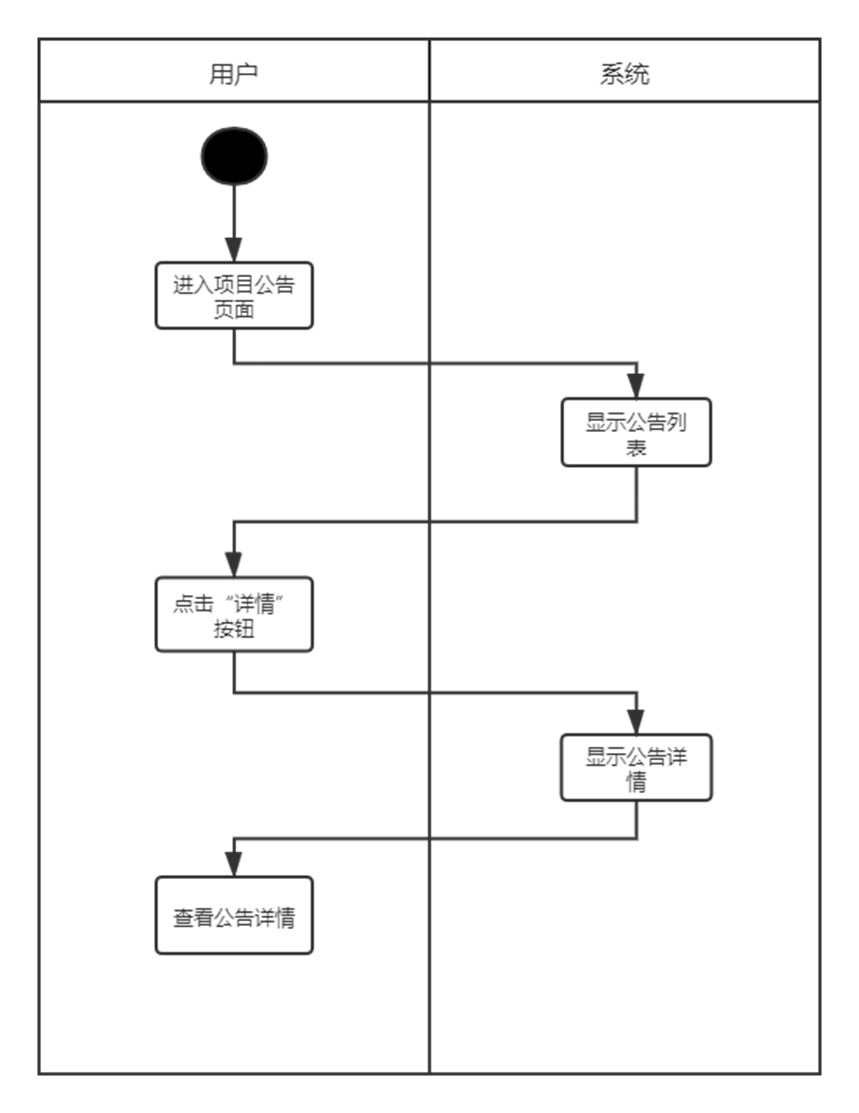


图1-26 查看公告详情流程

用户在公告板可以查看公告标题和详情。公告页面默认按照时间顺序显示公告，点击公告可以进入公告详情页面查看详细公告内容。

* + 1. 搜索项目公告

根据用户在搜索栏中输入的关键词，检索出相应的项目公告，默认情况下按时间排序。

* 1. 工具箱功能需求

计算器

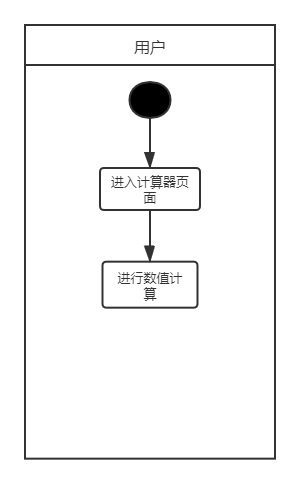


图1-27 使用计算器流程

用户可以在计算器页面进行数值计算

图标设计

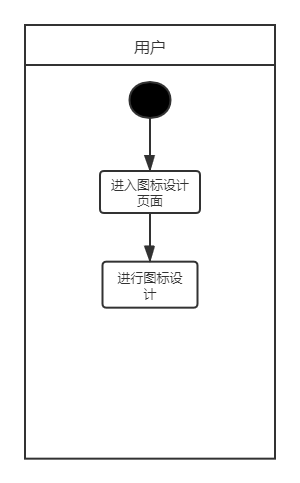


图1-28 使用图标设计流程

用户可以在图标设计页面进行图标设计

文本记事

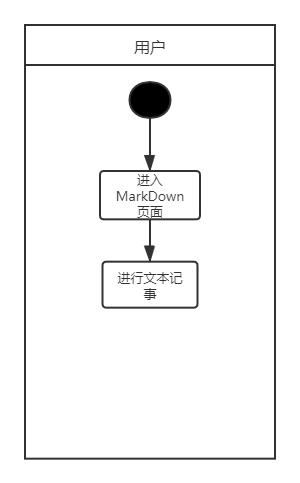


图1-26 使用文本记事流程

用户可以在MarkDown页面进行文本记事

流程图绘制

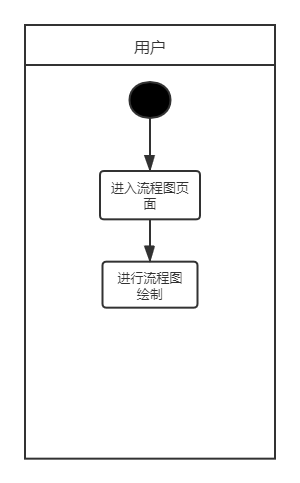


图1-30 进行流程图绘制流程

用户可以在流程图页面进行流程图绘制

排版绘制

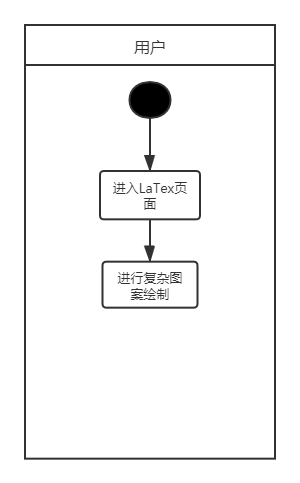


图1-31 书写复杂排版图样流程

用户可以在LaTex页面进行复杂图案绘制

* 1. 组织结构

本设计文档分为五部分：包括四个章节和一个附录

* 第一章：数据需求部分。按照各模块的功能以可视化的形式分别阐述本系统的数据需求；
* 第二章：概念设计部分，展示数据库总体上的概念设计和各模块系统概念设计；
* 第三章：逻辑设计部分，根据E-R图设计对应数据表和数据字段，并提供对应文字说明；
* 第四章：API文档，包含前后端对接所使用的HTTP API（REST风格）；
* 附录A：本文档的图表索引。

1. 概念设计
   1. 用户管理系统实现

用户登录系统

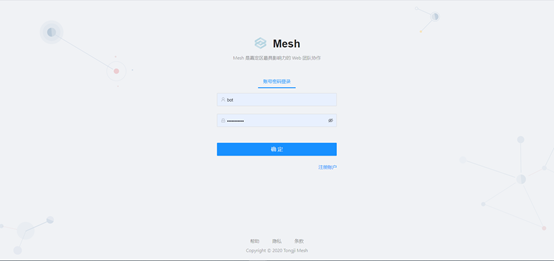
****

图2-1 用户登录系统

前端相关代码：

handleSubmit(e) {

e.preventDefault()

this.form.validateFields((err, values) => {

if (!err) {

console.log('Received values of form: ', values)

this.state.loginBtn = true

setTimeout(() => {

this.state.loginBtn = false

}, 1000)

this.Login({ username: values.username, password: values.password })

.then((res) => {

console.log('success,boy', res)

if (res.data.isSuccess) {

if (this.role === 'user') {

if (!this.teams ||this.teams.length === 0) {

this.$router.push({name: 'noTeam' })

} else this.$router.push({name: 'projectList' })

} else this.$router.push({name: 'management' })

*// 延迟 1秒显示欢迎信息*

setTimeout(() => {

this.$notification.success({

message: '欢迎',

description: `${timeFix()}，欢迎回来`,

})

}, 1000)

this.isLoginError = false

} else {

this.$notification.error({

message: res.data.msg,

})

}

})

.catch((err) => {

console.log('error, boy: ', err)

this.$notification.error({

message: err,

})

})

}

})

}

后端相关代码：

[HttpPost]

[Route("login")]

public JsonResult Login(UserRequest request)

{

if(!CornerCaseCheckHelper.Check(request.username,50,CornerCaseCheckHelper.Username))

{

return JsonReturnHelper.ErrorReturn(104,"Invalid username");

}

if (!CornerCaseCheckHelper.Check(request.password, 0,CornerCaseCheckHelper.PassWord))

{

returnJsonReturnHelper.ErrorReturn(111, "Invalid password.");

}

var user = \_meshContext.Users.FirstOrDefault(u => u.Email ==request.username);

*//Check Password*

if (user == null|| !CheckHashPassword(request.password,user.PasswordSalt, user.PasswordDigest))

{

returnJsonReturnHelper.ErrorReturn(201, "Incorrect username or password.");

}

if (HttpContext.Session.IsAvailable &&HttpContext.Session.GetString(request.username) == null)

{

HttpContext.Session.SetString(request.username,Guid.NewGuid().ToString());

}

return UserReturnValue(user);

}

用户注册系统

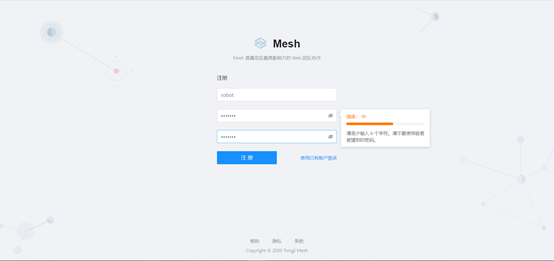


图2-2 用户注册系统

前端相关代码：

handleSubmit() {

this.form.validateFields({ force: true }, (err, values) => {

if (!err) {

this.state.passwordLevelChecked = false

sendRequest('register', { ...values })

.then((response) => {

*//TODO : store commit*

this.$router.push({ name:'registerResult', params: { ...values } })

})

.catch((err) => {})

}

})

}

后端相关代码：

[HttpPost]

[Route("register")]

public JsonResult Register(UserRequest request)

{

if(!CornerCaseCheckHelper.Check(request.username,50,CornerCaseCheckHelper.Username))

{

returnJsonReturnHelper.ErrorReturn(104, "Invalid username");

}

if (!CornerCaseCheckHelper.Check(request.password, 0,CornerCaseCheckHelper.PassWord))

{

returnJsonReturnHelper.ErrorReturn(111, "Invalid password.");

}

var user = \_meshContext.Users.FirstOrDefault(u => u.Email ==request.username);

if (user != null)

{

returnJsonReturnHelper.ErrorReturn(101, "User already exists.");

}

HashPassword hashPassword = GetHashPassword(request.password);

*//Create new user*

var newUser = new User()

{

Email = request.username,

Nickname = request.username,

PasswordDigest =hashPassword.PasswordDigest,

PasswordSalt =hashPassword.PasswordSalt,

Avatar =AvatarSaveHelper.PutObject(""),

ColorPreference ="blue",

LayoutPreference ="default",

RevealedPreference ="card"

};

*//try to save the user*

try

{

\_meshContext.Users.Add(newUser);

\_meshContext.SaveChanges();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

returnJsonReturnHelper.ErrorReturn(1, "Unexpected error.");

}

return UserReturnValue(newUser);

}

用户设置系统

****

图2-3 用户设置系统

前端相关代码：

saveSettings() {

*// console.log('test console, cities is:', this.cities)*

this.form.validateFields((err, values) => {

if (!err) {

console.log('Received values of form: ', values)

values.city = values.city[0] + ' ' + values.city[1] + ' ' +values.city[2]

console.log('new values of form: ', values)

let sendData = {

username: this.username,

nickname: values.nickname,

status: Number(values.status),

address: values.city || store.getters.address,

description: values.description,

birthday: this.birthdayStr || store.getters.birthday,

gender: this.gender,

}

this.updateUserInfo(sendData)

.then((res) => {

if (res.data.isSuccess) {

this.$notification.success({

message: '修改个人信息成功！',

})

} else {

this.$notification.error({

message: '修改个人信息失败，请重试',

})

}

})

.catch((err) => {

this.$notification.error({

message: '修改个人信息失败，请重试',

description: `${err.name}:${err.message}`,

})

})

}

})

}

后端相关代码：

[HttpPut]

[Route("user")]

public JsonResult UpdateUserInformation(UserInfo request)

{

if (!CornerCaseCheckHelper.Check(request.username,50,CornerCaseCheckHelper.Username))

{

returnJsonReturnHelper.ErrorReturn(104, "Invalid username");

}

if (HttpContext.Session.GetString(request.username) == null)

{

return JsonReturnHelper.ErrorReturn(2,"User status error.");

}

if (!CornerCaseCheckHelper.Check(request.nickname, 50,CornerCaseCheckHelper.Username))

{

returnJsonReturnHelper.ErrorReturn(120, "Invalid nickname.");

}

if (!CornerCaseCheckHelper.Check(request.birthday, 0,CornerCaseCheckHelper.Time))

{

returnJsonReturnHelper.ErrorReturn(121, "Invalid birthday.");

}

if (!CornerCaseCheckHelper.Check(request.description, 100,CornerCaseCheckHelper.Description))

{

returnJsonReturnHelper.ErrorReturn(122, "Invalid description.");

}

var user = \_meshContext.Users.First(u => u.Email ==request.username);

try

{

user.Nickname =request.nickname;

user.Gender = request.gender;

user.Status = request.status;

user.Address = request.address;

user.Description =request.description;

user.Birthday =Convert.ToDateTime(request.birthday);

user.Avatar =AvatarSaveHelper.PutObject(request.avatar,user.Avatar);

\_meshContext.Users.Update(user);

\_meshContext.SaveChanges();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

returnJsonReturnHelper.ErrorReturn(1, "Unexpected error.");

}

return UserReturnValue(user);

}

[HttpGet]

[Route("user")]

public JsonResult QueryUser(string username, string keyword)

{

if (!CornerCaseCheckHelper.Check(username, 50,CornerCaseCheckHelper.Username))

{

returnJsonReturnHelper.ErrorReturn(104, "Invalid username.");

}

if (!CornerCaseCheckHelper.Check(keyword, 50,CornerCaseCheckHelper.Username))

{

returnJsonReturnHelper.ErrorReturn(457, "Invalid keyword.");

}

if (HttpContext.Session.IsAvailable &&HttpContext.Session.GetString(username) == null)

{

returnJsonReturnHelper.ErrorReturn(2, "User status error.");

}

var userAccordingToUsername = \_meshContext.Users

.Where(u =>u.Email.Contains(keyword));

var userAccordingToNickname = \_meshContext.Users

.Where(u =>u.Nickname.Contains(keyword));

var userAccordingToDescription = \_meshContext.Users

.Where(u =>u.Description.Contains(keyword));

var users = userAccordingToUsername

.Union(userAccordingToNickname)

.Union(userAccordingToDescription)

.Select(u => new

{

username = u.Email,

avatar =AvatarSaveHelper.GetObject(u.Avatar),

nickname = u.Nickname,

gender = u.Gender,

status = u.Status,

address = u.Address,

description =u.Description,

birthday = u.Birthday

})

.ToList();

return Json(new

{

err\_code = 0,

data = new

{

users = users

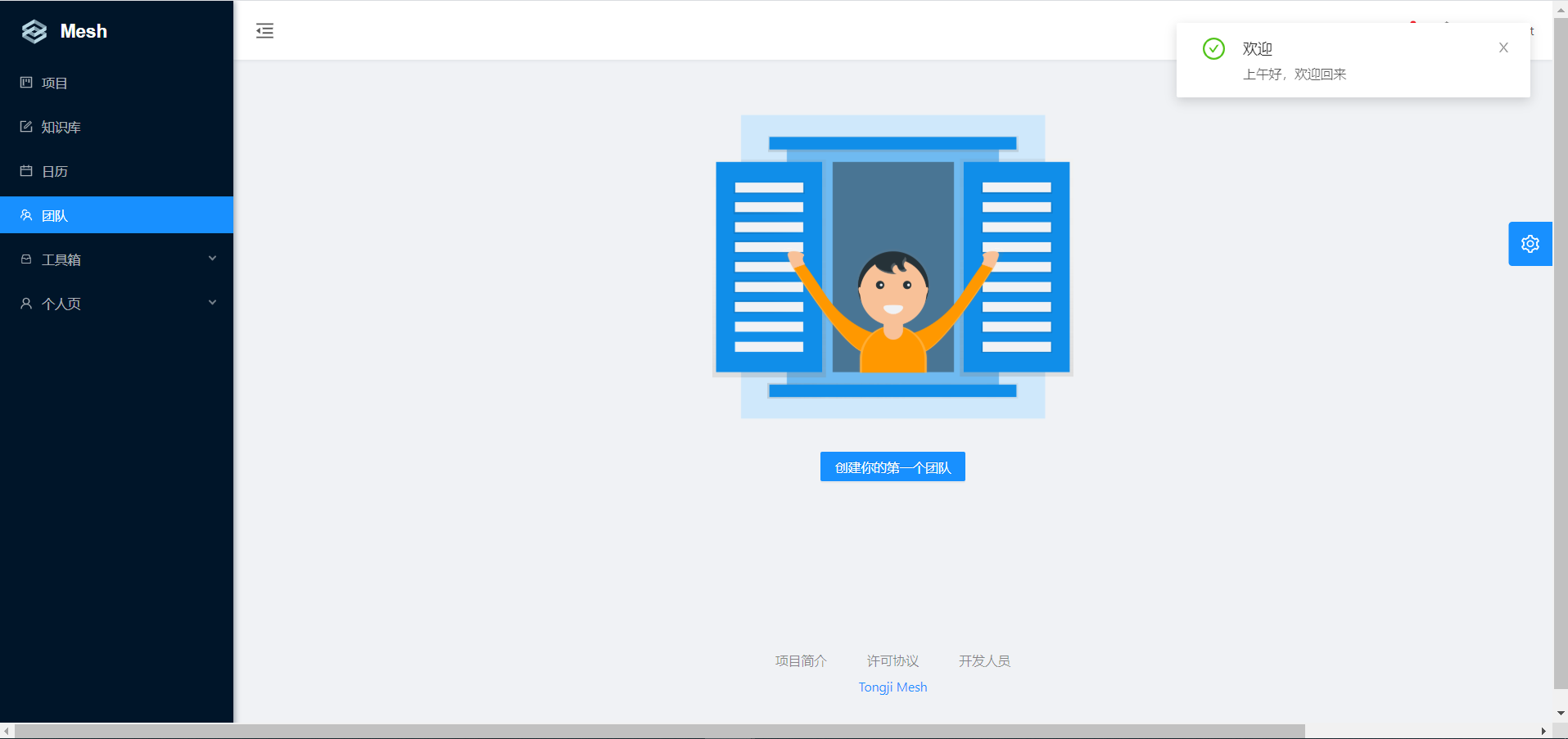
}

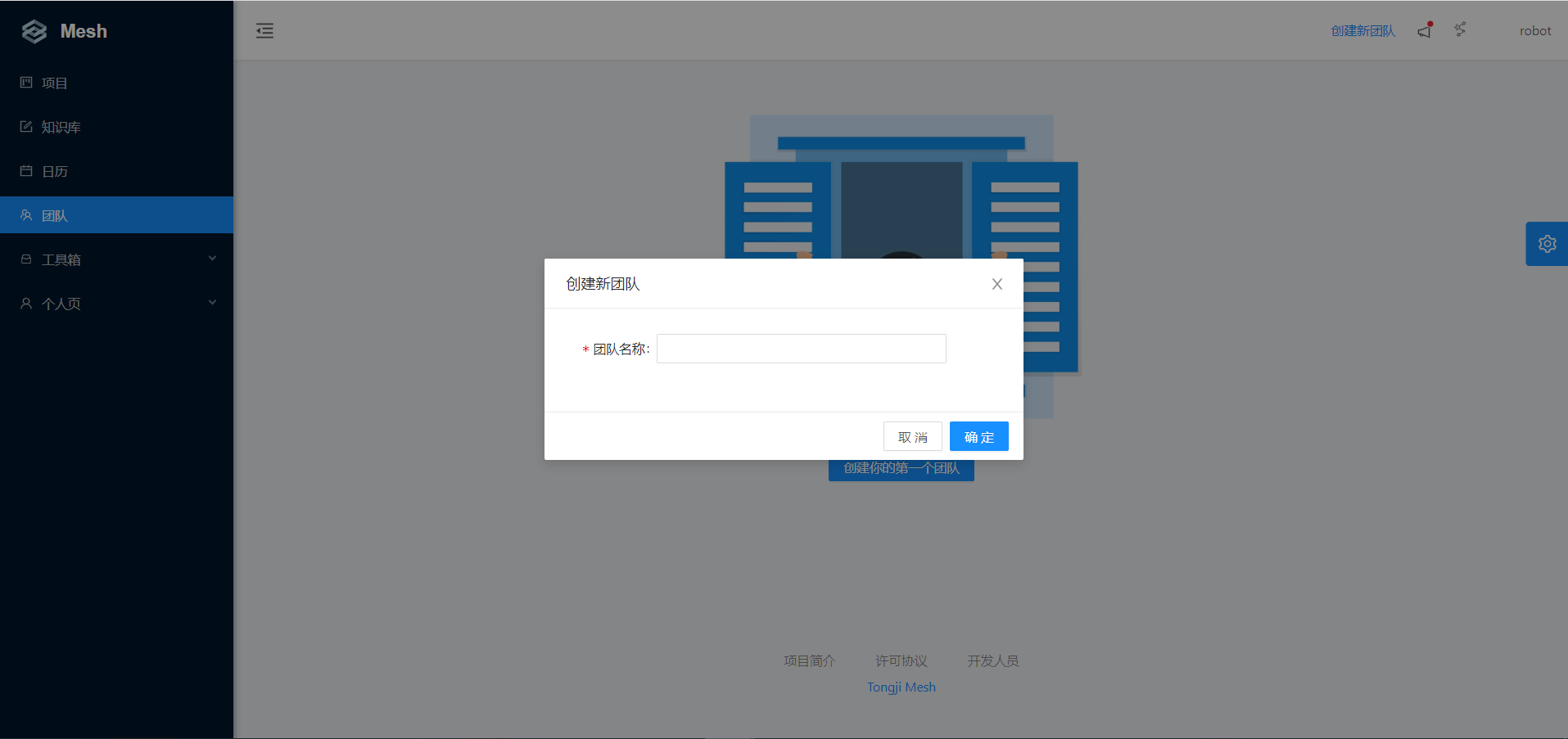
});

}

* 1. 团队管理系统实现

创建新团队系统





前端相关代码：

handleSubmit(e) {

console.log('创建新团队')

e.preventDefault()

this.TOGGLE\_CREATING\_TEAM(false)

this.form.validateFields((err, values) => {

if (!err) {

console.log('Received values of form: ', values)

const promises = []

this.createTeam({ username: store.getters.username, teamName:values.teamName })

.then((response) => {

console.log('success,boy',response)

*// 增加新建的团队到团队列表*

this.ADD\_NEW\_TEAM(response.data.team)

console.log('after add new team',store.getters.teams)

*// 更新当前团队下的项目，消息和知识库*

promises.push(this.queryNotification({ username: this.username }))

const requestData = { username:this.username, teamId: response.data.team.teamId }

promises.push(this.queryTeam(requestData),this.queryTeamKB(requestData))

*// 异步显示成功信息*

setTimeout(() => {

this.$notification.success({

message: '创建成功',

description: `${timeFix()}，已成功创建新团队`,

})

}, 0)

*// 跳转到团队路由界面*

this.$router.push({ name:'membersList' })

})

.catch((err) => {

console.log('error, boy: ', err)

this.$notification.error({

message: '创建失败',

description: err,

})

})

if (promises.length) {

Promise.all(promises)

.then(() => {

this.$notification.success({ message:'团队信息加载成功！' })

})

.catch((err) => {

this.$notification.error({

message: '请求团队信息失败，请重试',

description: `${err.name}:${err.message}`,

})

this.$router.push('/404')

})

}

}

})

}

后端相关代码：

[HttpGet]

public JsonResult QueryTeam(string username, int teamId)

{

var checkResult = CheckUsername(username);

if (checkResult != null)

{

return checkResult;

}

if (!CornerCaseCheckHelper.Check(teamId, 0, CornerCaseCheckHelper.Id))

{

returnJsonReturnHelper.ErrorReturn(301, "Invalid teamId");

}

var user = \_meshContext.Users.First(u => u.Email ==username);

var team = \_meshContext.Teams.FirstOrDefault(t => t.Id == teamId);

if (team == null)

{

returnJsonReturnHelper.ErrorReturn(302, "Team does not exist.");

}

if (\_permissionCheck.CheckTeamPermission(username, team) ==PermissionCheckHelper.TeamOutsider)

{

returnJsonReturnHelper.ErrorReturn(305, "Permission denied.");

}

*//Find team members*

var teamCooperation = \_meshContext.Cooperations

.Where(c => c.TeamId ==team.Id);

var adminName =\_meshContext.Users.First(u => u.Id == team.AdminId).Nickname;

var members = \_meshContext.Users

.Join(teamCooperation, u =>u.Id, c => c.UserId, (u, c) =>

new Member()

{

Id = u.Id,

Username = u.Email,

Nickname = u.Nickname,

Avatar =AvatarSaveHelper.GetObject(u.Avatar)

}).ToList();

*//Find projects of the team*

var project = \_meshContext.Projects

.Where(p => p.TeamId==teamId);

var teamProjects = \_meshContext.Users

.Join(project, u => u.Id, p=> p.AdminId, (u, p) =>

new TeamProject()

{

ProjectId = p.Id,

ProjectName = p.Name,

AdminName = u.Nickname,

ProjectLogo =AvatarSaveHelper.GetObject(p.Icon)

}).ToList();

var userTeamCooperation = teamCooperation.First(c => c.UserId ==user.Id);

try

{

userTeamCooperation.AccessCount+= 1;

\_meshContext.Cooperations.Update(userTeamCooperation);

\_meshContext.SaveChanges();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

returnJsonReturnHelper.ErrorReturn(1, "Unexpected error.");

}

return Json(new

{

err\_code = 0,

data = new

{

isSuccess = true,

msg = "",

team = new

{

teamId = team.Id,

teamName = team.Name,

createTime =team.CreatedTime,

adminName = adminName,

members = members,

teamProjects =teamProjects

}

}

});

}

[HttpPost]

public JsonResult CreateTeam(CreateRequest request)

{

var checkResult =CheckUsername(request.username);

if (checkResult != null)

{

return checkResult;

}

if (!CornerCaseCheckHelper.Check(request.teamName, 50,CornerCaseCheckHelper.Title))

{

returnJsonReturnHelper.ErrorReturn(310, "Invalid teamName.");

}

var user = \_meshContext.Users.First(u => u.Email ==request.username);

var createdTeam = new Team()

{

Name = request.teamName,

AdminId = user.Id

};

*//Start transaction to save the new team*

using (var transaction = \_meshContext.Database.BeginTransaction())

{

try

{

\_meshContext.Teams.Add(createdTeam);

\_meshContext.SaveChanges();

\_meshContext.Cooperations.Add(newCooperation()

{

TeamId =createdTeam.Id,

UserId = user.Id

});

\_meshContext.TeamMemoCollections.Add(new TeamMemoCollection()

{

TeamId = createdTeam.Id

});

\_meshContext.SaveChanges();

transaction.Commit();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

returnJsonReturnHelper.ErrorReturn(1, "Unexpected error.");

}

}

*//Return team members*

var teamMembers = new List<Member>();

teamMembers.Add(new Member(){Username = user.Email,Id = user.Id});

return Json(new

{

err\_code = 0,

data = new

{

isSuccess = true,

team = new

{

teamId =createdTeam.Id,

createTime =createdTeam.CreatedTime,

teamName =createdTeam.Name,

adminName =user.Nickname,

members =teamMembers

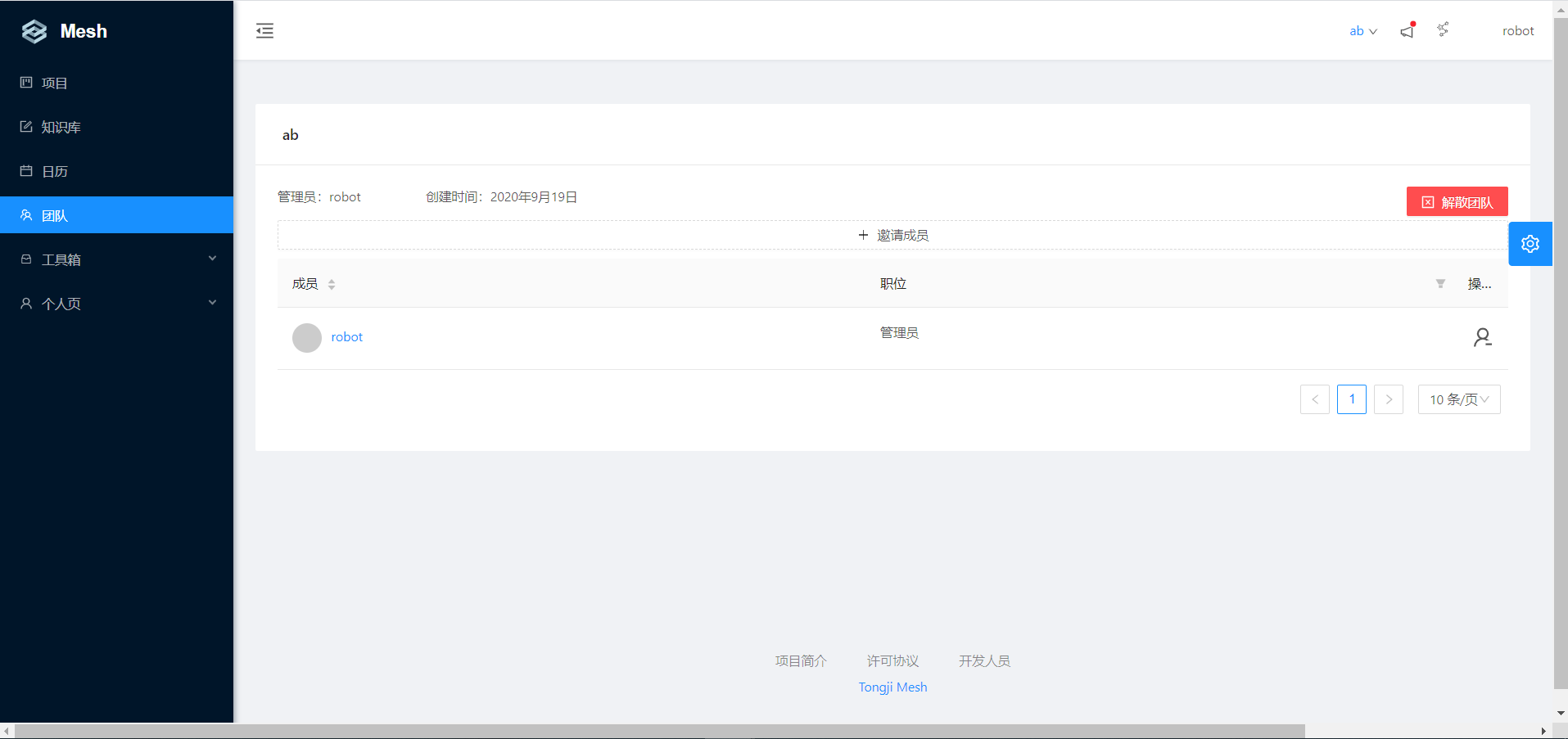
}

}

}

);

}

****

****

图2-4 图2-5 创建新团队系统

前端相关代码：

handleInviteUsers() {

this.inviting = true

if (!this.selectedUsers || this.selectedUsers.length === 0) returnthis.hideModal()

const promises = []

this.selectedUsers.forEach((username) =>

promises.push(

this.inviteMember({

inviteName: username,

teamId: this.teamId,

username: this.username,

})

)

)

if (promises.length) {

return Promise.all(promises)

.then((res) => {

console.log('add team members success!, new members: ',this.selectedUsers)

this.$notification.success({

message: '成功添加新成员',

})

})

.catch((err) => {

this.$notification.error({

message: '添加新团队成员失败',

})

})

.finally(() => {

this.queryTeam({

username: this.username,

teamId: this.teamId,

})

.then((res) => {

this.$notification.success({

message: '成功更新团队',

})

})

.catch((err) => {

this.$notification.error({

message: '更新团队失败',

})

})

this.inviting = false

this.hideModal()

})

}

this.inviting = false

this.hideModal()

}

后端相关代码：

[HttpPost]

public JsonResult CreateTeam(CreateRequest request)

{

var checkResult = CheckUsername(request.username);

if (checkResult != null)

{

return checkResult;

}

if (!CornerCaseCheckHelper.Check(request.teamName, 50,CornerCaseCheckHelper.Title))

{

returnJsonReturnHelper.ErrorReturn(310, "Invalid teamName.");

}

var user = \_meshContext.Users.First(u => u.Email ==request.username);

var createdTeam = new Team()

{

Name = request.teamName,

AdminId = user.Id

};

*//Start transaction to save the newteam*

using (var transaction = \_meshContext.Database.BeginTransaction())

{

try

{

\_meshContext.Teams.Add(createdTeam);

\_meshContext.SaveChanges();

\_meshContext.Cooperations.Add(new Cooperation()

{

TeamId =createdTeam.Id,

UserId = user.Id

});

\_meshContext.TeamMemoCollections.Add(new TeamMemoCollection()

{

TeamId = createdTeam.Id

});

\_meshContext.SaveChanges();

transaction.Commit();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

returnJsonReturnHelper.ErrorReturn(1, "Unexpected error.");

}

}

*//Return team members*

var teamMembers = new List<Member>();

teamMembers.Add(new Member(){Username = user.Email,Id = user.Id});

return Json(new

{

err\_code = 0,

data = new

{

isSuccess = true,

team = new

{

teamId =createdTeam.Id,

createTime =createdTeam.CreatedTime,

teamName =createdTeam.Name,

adminName =user.Nickname,

members = teamMembers

}

}

}

);

}

[HttpPost]

[Route("invite")]

public JsonResult InviteNewTeamMember(InviteRequest request)

{

var checkUsernameResult = CheckUsername(request.username);

if (checkUsernameResult != null)

{

return checkUsernameResult;

}

if (!CornerCaseCheckHelper.Check(request.inviteName,50,CornerCaseCheckHelper.Username))

{

returnJsonReturnHelper.ErrorReturn(108, "Invalid inviteName");

}

if (!CornerCaseCheckHelper.Check(request.teamId, 0,CornerCaseCheckHelper.Id))

{

returnJsonReturnHelper.ErrorReturn(301, "Invalid teamId");

}

var team = \_meshContext.Teams.FirstOrDefault(t => t.Id ==request.teamId);

if (team == null)

{

returnJsonReturnHelper.ErrorReturn(302, "Team not exist.");

}

if (\_permissionCheck.CheckTeamPermission(request.username, team) !=PermissionCheckHelper.TeamAdmin)

{

returnJsonReturnHelper.ErrorReturn(305, "Permission denied.");

}

var inviteUser = \_meshContext.Users.FirstOrDefault(u => u.Email ==request.inviteName);

if (inviteUser==null)

{

returnJsonReturnHelper.ErrorReturn(108, "Username or inviteName notexists.");

}

if (\_permissionCheck.CheckTeamPermission(request.inviteName, team) !=PermissionCheckHelper.TeamOutsider)

{

returnJsonReturnHelper.ErrorReturn(306, "User already in team.");

}

var cooperation = new Cooperation()

{

TeamId = request.teamId,

UserId = inviteUser.Id

};

try

{

\_meshContext.Cooperations.Add(cooperation);

\_meshContext.SaveChanges();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

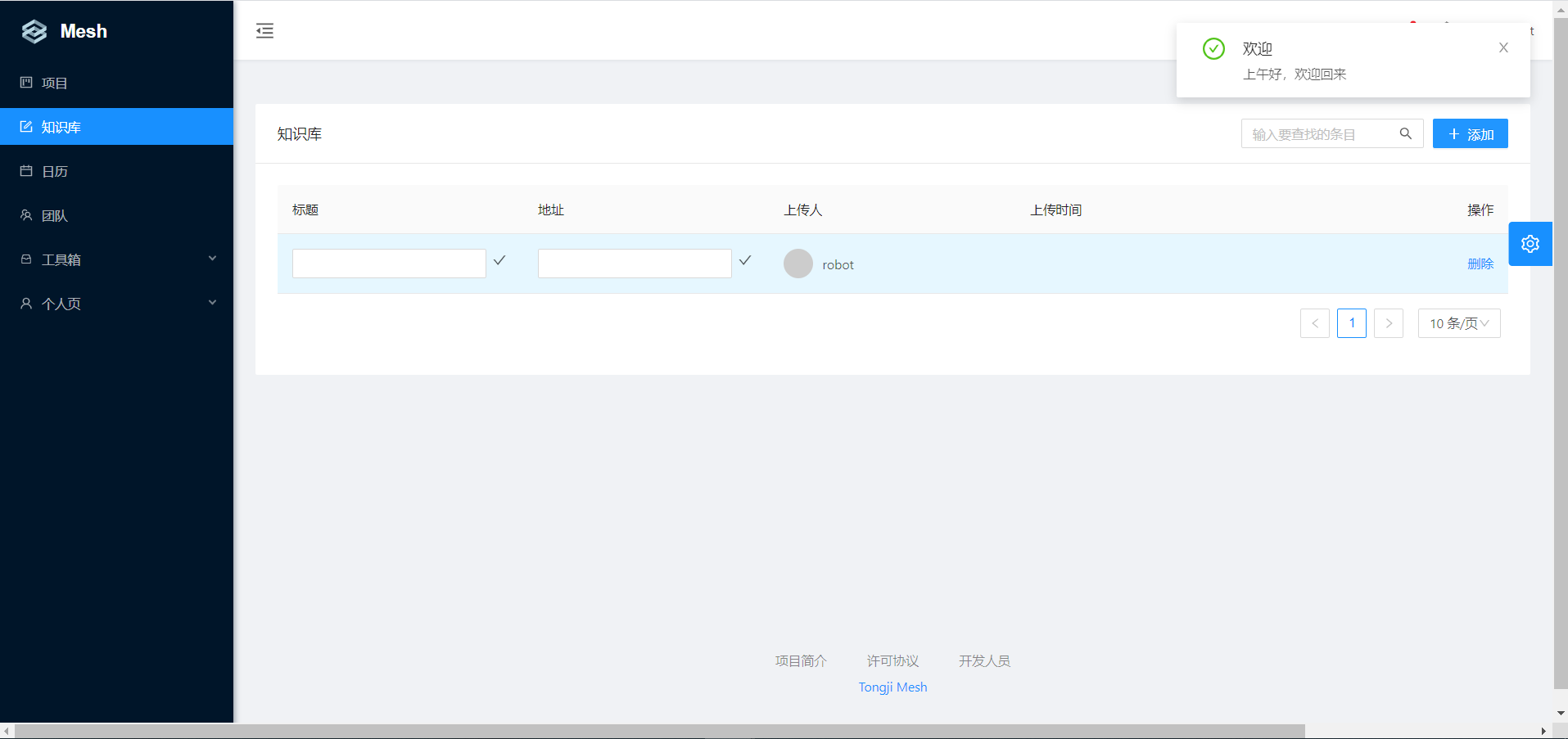
return JsonReturnHelper.ErrorReturn(1,"Unexpected error.");

}

return JsonReturnHelper.SuccessReturn();

}

团队知识库系统

****

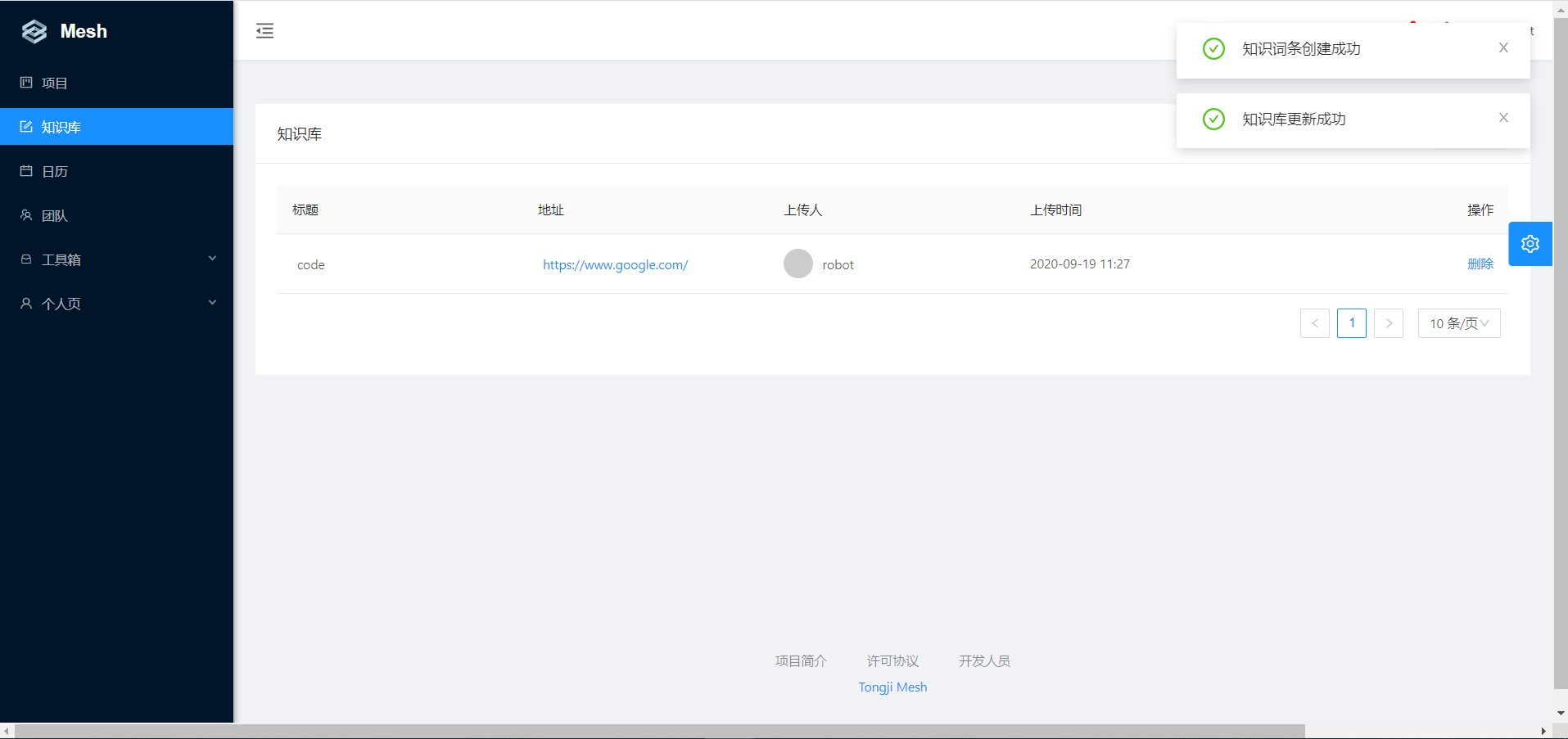
****

图2-6 图2-7 团队知识库系统

前端相关代码：

handleInviteUsers() {

this.inviting = true

if (!this.selectedUsers || this.selectedUsers.length === 0) returnthis.hideModal()

const promises = []

this.selectedUsers.forEach((username) =>

promises.push(

this.inviteMember({

inviteName: username,

teamId: this.teamId,

username: this.username,

})

)

)

if (promises.length) {

return Promise.all(promises)

.then((res) => {

console.log('add team members success!, new members: ',this.selectedUsers)

this.$notification.success({

message: '成功添加新成员',

})

})

.catch((err) => {

this.$notification.error({

message: '添加新团队成员失败',

})

})

.finally(() => {

this.queryTeam({

username: this.username,

teamId: this.teamId,

})

.then((res) => {

this.$notification.success({

message: '成功更新团队',

})

})

.catch((err) => {

this.$notification.error({

message: '更新团队失败',

})

})

this.inviting = false

this.hideModal()

})

}

this.inviting = false

this.hideModal()

}

后端相关代码：

[HttpPost]

public JsonResult CreateTeam(CreateRequest request)

{

var checkResult = CheckUsername(request.username);

if (checkResult != null)

{

return checkResult;

}

if (!CornerCaseCheckHelper.Check(request.teamName, 50,CornerCaseCheckHelper.Title))

{

returnJsonReturnHelper.ErrorReturn(310, "Invalid teamName.");

}

var user = \_meshContext.Users.First(u => u.Email ==request.username);

var createdTeam = new Team()

{

Name = request.teamName,

AdminId = user.Id

};

*//Start transaction to save the newteam*

using (var transaction = \_meshContext.Database.BeginTransaction())

{

try

{

\_meshContext.Teams.Add(createdTeam);

\_meshContext.SaveChanges();

\_meshContext.Cooperations.Add(new Cooperation()

{

TeamId =createdTeam.Id,

UserId = user.Id

});

\_meshContext.TeamMemoCollections.Add(new TeamMemoCollection()

{

TeamId = createdTeam.Id

});

\_meshContext.SaveChanges();

transaction.Commit();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

returnJsonReturnHelper.ErrorReturn(1, "Unexpected error.");

}

}

*//Return team members*

var teamMembers = new List<Member>();

teamMembers.Add(new Member(){Username = user.Email,Id = user.Id});

return Json(new

{

err\_code = 0,

data = new

{

isSuccess = true,

team = new

{

teamId =createdTeam.Id,

createTime =createdTeam.CreatedTime,

teamName =createdTeam.Name,

adminName =user.Nickname,

members = teamMembers

}

}

}

);

}

[HttpPost]

[Route("invite")]

public JsonResult InviteNewTeamMember(InviteRequest request)

{

var checkUsernameResult = CheckUsername(request.username);

if (checkUsernameResult != null)

{

return checkUsernameResult;

}

if (!CornerCaseCheckHelper.Check(request.inviteName,50,CornerCaseCheckHelper.Username))

{

returnJsonReturnHelper.ErrorReturn(108, "Invalid inviteName");

}

if (!CornerCaseCheckHelper.Check(request.teamId, 0,CornerCaseCheckHelper.Id))

{

returnJsonReturnHelper.ErrorReturn(301, "Invalid teamId");

}

var team = \_meshContext.Teams.FirstOrDefault(t => t.Id ==request.teamId);

if (team == null)

{

returnJsonReturnHelper.ErrorReturn(302, "Team not exist.");

}

if (\_permissionCheck.CheckTeamPermission(request.username, team) !=PermissionCheckHelper.TeamAdmin)

{

returnJsonReturnHelper.ErrorReturn(305, "Permission denied.");

}

var inviteUser = \_meshContext.Users.FirstOrDefault(u => u.Email ==request.inviteName);

if (inviteUser==null)

{

returnJsonReturnHelper.ErrorReturn(108, "Username or inviteName notexists.");

}

if (\_permissionCheck.CheckTeamPermission(request.inviteName, team) !=PermissionCheckHelper.TeamOutsider)

{

returnJsonReturnHelper.ErrorReturn(306, "User already in team.");

}

var cooperation = new Cooperation()

{

TeamId = request.teamId,

UserId = inviteUser.Id

};

try

{

\_meshContext.Cooperations.Add(cooperation);

\_meshContext.SaveChanges();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

return JsonReturnHelper.ErrorReturn(1,"Unexpected error.");

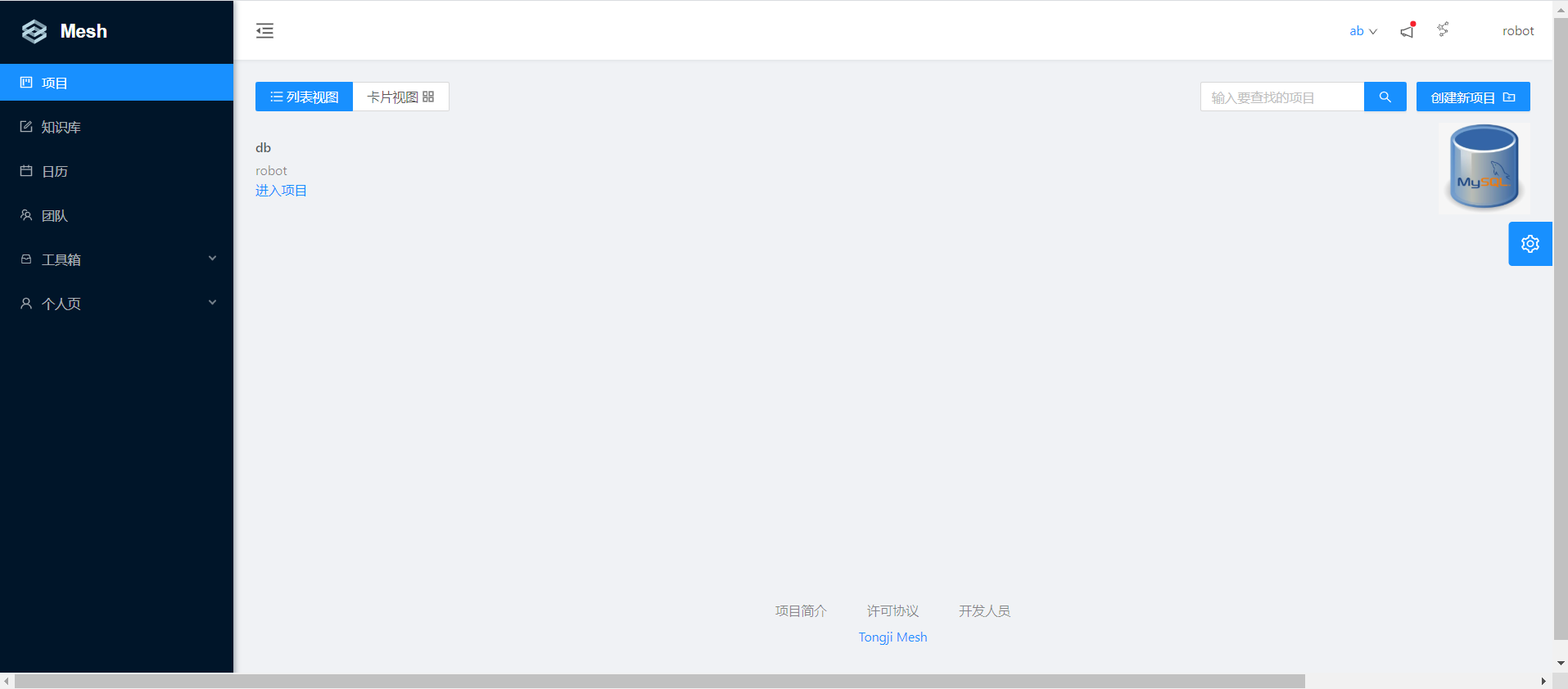
}

return JsonReturnHelper.SuccessReturn();

}

* 1. 项目管理系统实现

****

****

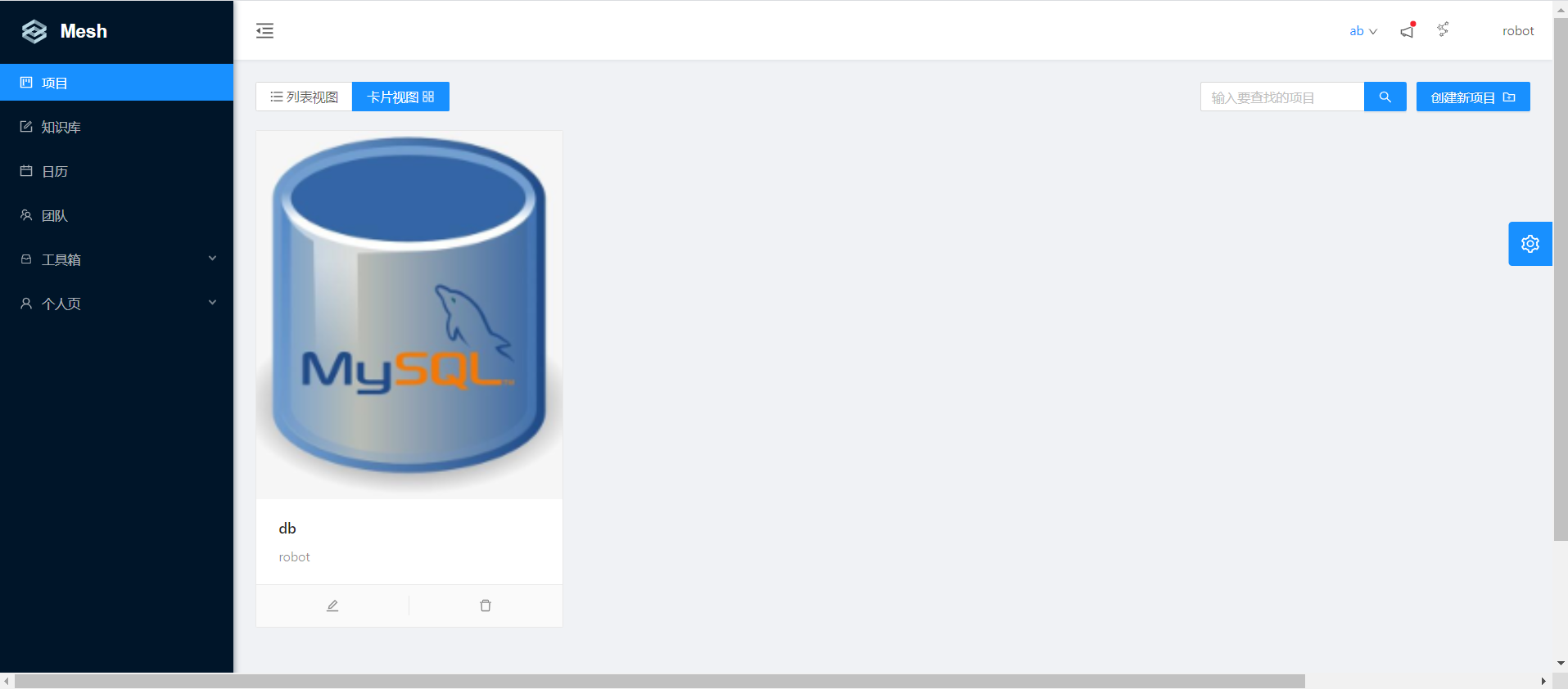
****

图2-8 图2-9 图2-10 项目管理系统

前端相关代码：

handleCreateSubmit(e) {

e.preventDefault()

this.createLoading = true

this.createForm.validateFields((err, values) => {

if (!err) {

console.log('Received values of form: ', values)

this.createProject({

username: store.getters.username,

teamId: store.getters.teamId,

projectName: values.prjName,

adminName: values.prjAdmin,

isPublic: values.prjAuthority ==='public',

})

.then((response) => {

console.log('success,boy',response)

*// 延迟显示欢迎信息*

setTimeout(() => {

this.$notification.success({

message: '创建成功',

description: `${timeFix()}，已成功添加新项目`,

})

}, 0)

})

.catch((err) => {

console.log('error, boy: ', err)

this.$notification.error({

message: '创建失败',

description: err.message,

})

})

.finally(() => {

this.createLoading = false

this.createProjForm = false

})

}

})

}

后端相关代码：

[HttpPost]

[Route("project")]

public JsonResult CreateProject(ProjectRequest request)

{

var checkResult = CheckUsername(request.Username);

if (checkResult != null)

{

return checkResult;

}

if (!CornerCaseCheckHelper.Check(request.TeamId, 0,CornerCaseCheckHelper.Id))

{

returnJsonReturnHelper.ErrorReturn(301, "Invalid teamId.");

}

if (!CornerCaseCheckHelper.Check(request.ProjectName, 50,CornerCaseCheckHelper.Title))

{

returnJsonReturnHelper.ErrorReturn(710, "Invalid projectName.");

}

if (!CornerCaseCheckHelper.Check(request.AdminName, 50,CornerCaseCheckHelper.Username))

{

return JsonReturnHelper.ErrorReturn(711,"Invalid adminName");

}

var user = \_meshContext.Users.First(u => u.Email ==request.Username);

*//Check if admin exists*

var admin = \_meshContext.Users.FirstOrDefault(a => a.Email ==request.AdminName);

if (admin == null)

{

returnJsonReturnHelper.ErrorReturn(704, "Admin does not exist.");

}

*//Check if team exists*

var team = \_meshContext.Teams.FirstOrDefault(t => t.Id ==request.TeamId);

if (team == null)

{

returnJsonReturnHelper.ErrorReturn(302, "Team does not exist.");

}

*//Check if admin is in the team*

var teamCheckResult =\_permissionCheck.CheckTeamPermission(request.AdminName, team);

if (teamCheckResult ==PermissionCheckHelper.TeamOutsider)

{

returnJsonReturnHelper.ErrorReturn(702, "Invalid admin.");

}

*//Check if user is the admin of the team*

teamCheckResult = \_permissionCheck.CheckTeamPermission(request.Username,team);

if (teamCheckResult !=PermissionCheckHelper.TeamAdmin)

{

returnJsonReturnHelper.ErrorReturn(701, "Permission denied.");

}

var newProject = new Project()

{

Name = request.ProjectName,

AdminId = admin.Id,

TeamId = team.Id,

Publicity = request.IsPublic,

Icon = AvatarSaveHelper.PutObject("")

};

var members = new List<MemInfo> {new MemInfo()

{

UserId = admin.Id, Username =admin.Email,Nickname = admin.Nickname,Avatar =AvatarSaveHelper.GetObject(admin.Avatar)

}};

*//Start a transaction to save the project*

using (var transaction = \_meshContext.Database.BeginTransaction())

{

try

{

\_meshContext.Projects.Add(newProject);

\_meshContext.SaveChanges();

\_meshContext.Develops.Add(new Develop()

{

ProjectId =newProject.Id,

UserId = admin.Id

});

\_meshContext.BulletinBoards.Add(new BulletinBoard()

{

ProjectId =newProject.Id

});

\_meshContext.TaskBoards.Add(newTaskBoard()

{

ProjectId =newProject.Id

});

\_meshContext.ProjectMemoCollections.Add(new ProjectMemoCollection()

{

ProjectId =newProject.Id

});

\_meshContext.SaveChanges();

transaction.Commit();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

returnJsonReturnHelper.ErrorReturn(1, "Unexpected error.");

}

}

return ProjectResult(newProject, admin.Nickname);

}

[HttpPost]

[Route("project/invite")]

public JsonResult InviteNewProjectMember(ProjectRequest request)

{

var checkResult = CheckUsername(request.Username);

if (checkResult != null)

{

return checkResult;

}

if (!CornerCaseCheckHelper.Check(request.TeamId, 0,CornerCaseCheckHelper.Id))

{

returnJsonReturnHelper.ErrorReturn(301, "Invalid teamId.");

}

if (!CornerCaseCheckHelper.Check(request.ProjectId, 0,CornerCaseCheckHelper.Id))

{

returnJsonReturnHelper.ErrorReturn(701, "Invalid projectId.");

}

if (!CornerCaseCheckHelper.Check(request.InviteName, 50,CornerCaseCheckHelper.Username))

{

returnJsonReturnHelper.ErrorReturn(101, "Invalid inviteName");

}

*//Check if inviteUser exists*

var inviteUser = \_meshContext.Users.FirstOrDefault(a => a.Email ==request.InviteName);

if (inviteUser == null)

{

returnJsonReturnHelper.ErrorReturn(704, "Admin does not exist.");

}

*//Check if team exists*

var team = \_meshContext.Teams.FirstOrDefault(t => t.Id ==request.TeamId);

if (team == null)

{

returnJsonReturnHelper.ErrorReturn(302, "Team does not exist.");

}

*//Check if user in the team*

var teamUserCheckResult =\_permissionCheck.CheckTeamPermission(request.Username, team);

if (teamUserCheckResult ==PermissionCheckHelper.TeamOutsider)

{

return JsonReturnHelper.ErrorReturn(801,"Permission denied");

}

*//Check if inviteUser in the team*

var teamCheckResult =\_permissionCheck.CheckTeamPermission(request.InviteName, team);

if (teamCheckResult ==PermissionCheckHelper.TeamOutsider)

{

returnJsonReturnHelper.ErrorReturn(801, "InviteUser is not in the team");

}

*//Check if project exists*

var project = \_meshContext.Projects.FirstOrDefault(p => p.Id ==request.ProjectId);

if (project == null)

{

returnJsonReturnHelper.ErrorReturn(707, "Project does not exist.");

}

*//Check if user is the admin of the project*

var projectCheckResult =\_permissionCheck.CheckProjectPermission(request.Username, project);

if (projectCheckResult != PermissionCheckHelper.ProjectAdmin)

{

returnJsonReturnHelper.ErrorReturn(701, "Permission denied.");

}

try

{

\_meshContext.Develops.Add(newDevelop()

{

UserId = inviteUser.Id,

ProjectId = project.Id

});

\_meshContext.SaveChanges();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

returnJsonReturnHelper.ErrorReturn(1, "Unexpected error.");

}

return JsonReturnHelper.SuccessReturn();

}

[HttpGet]

[Route("project")]

public JsonResult QueryProject(string username, int projectId)

{

var checkResult = CheckUsername(username);

if (checkResult != null)

{

return checkResult;

}

*//Check if project exists*

var project = \_meshContext.Projects.FirstOrDefault(p => p.Id ==projectId);

if (project == null)

{

returnJsonReturnHelper.ErrorReturn(707, "Invalid projectId.");

}

*//Check if user is in the project*

var projectCheckResult =\_permissionCheck.CheckProjectPermission(username, project);

if (projectCheckResult == PermissionCheckHelper.ProjectOutsider)

{

returnJsonReturnHelper.ErrorReturn(701, "Permission denied.");

}

var user = \_meshContext.Users.First(u => u.Id == project.AdminId);

return ProjectResult(project, user.Nickname);

}

[HttpPatch]

[Route("project")]

public JsonResult UpdateProject(ProjectRequest request)

{

var checkResult = CheckUsername(request.Username);

if (checkResult != null)

{

return checkResult;

}

if (!CornerCaseCheckHelper.Check(request.TeamId, 0,CornerCaseCheckHelper.Id))

{

returnJsonReturnHelper.ErrorReturn(301, "Invalid teamId.");

}

if (!CornerCaseCheckHelper.Check(request.ProjectId, 0,CornerCaseCheckHelper.Id))

{

returnJsonReturnHelper.ErrorReturn(701, "Invalid projectId.");

}

if (!CornerCaseCheckHelper.Check(request.ProjectName, 50, CornerCaseCheckHelper.Title))

{

returnJsonReturnHelper.ErrorReturn(710, "Invalid projectName.");

}

*//Check if team exists*

var team = \_meshContext.Teams.FirstOrDefault(t => t.Id == request.TeamId);

if (team == null)

{

returnJsonReturnHelper.ErrorReturn(302, "Invalid teamId.");

}

*//Check if project exists*

var project = \_meshContext.Projects.FirstOrDefault(p => p.Id ==request.ProjectId);

if (project == null)

{

returnJsonReturnHelper.ErrorReturn(707, "Invalid projectId.");

}

*//Check if user in the team*

var teamCheckResult =\_permissionCheck.CheckTeamPermission(request.Username, team);

if (teamCheckResult ==PermissionCheckHelper.TeamOutsider)

{

return JsonReturnHelper.ErrorReturn(702,"Invalid username.");

}

*//Check if user is the admin of the project*

var projectCheckResult =\_permissionCheck.CheckProjectPermission(request.Username, project);

if (projectCheckResult != PermissionCheckHelper.ProjectAdmin)

{

returnJsonReturnHelper.ErrorReturn(701, "Permission denied.");

}

try

{

project.Publicity =request.IsPublic;

project.Name =request.ProjectName;

project.Icon =AvatarSaveHelper.PutObject(request.ProjectLogo, project.Icon);

\_meshContext.Projects.Update(project);

\_meshContext.SaveChanges();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

returnJsonReturnHelper.ErrorReturn(1, "Unexpected error.");

}

var user = \_meshContext.Users.First(u => u.Id == project.AdminId);

return ProjectResult(project, user.Nickname);

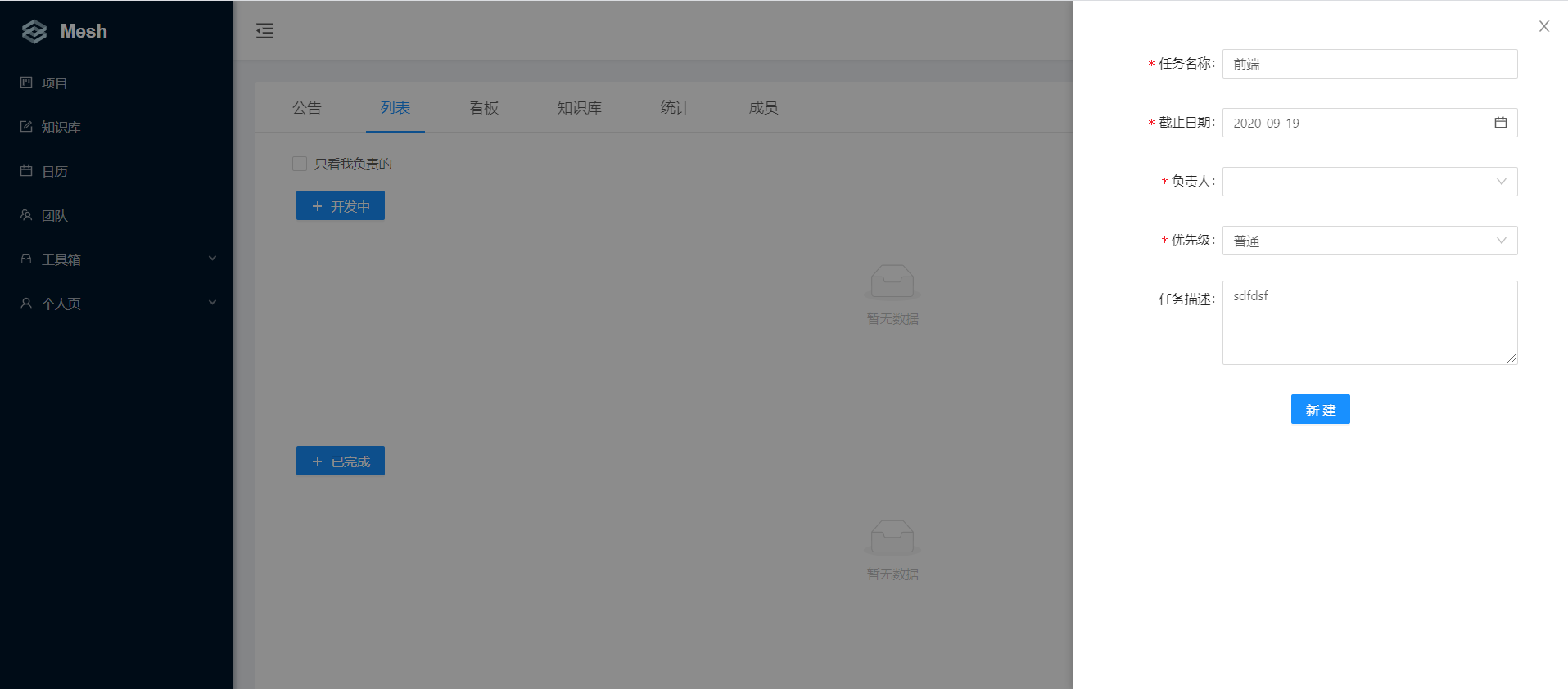
}

}

}

* 1. 任务管理系统实现

任务管理系统

****

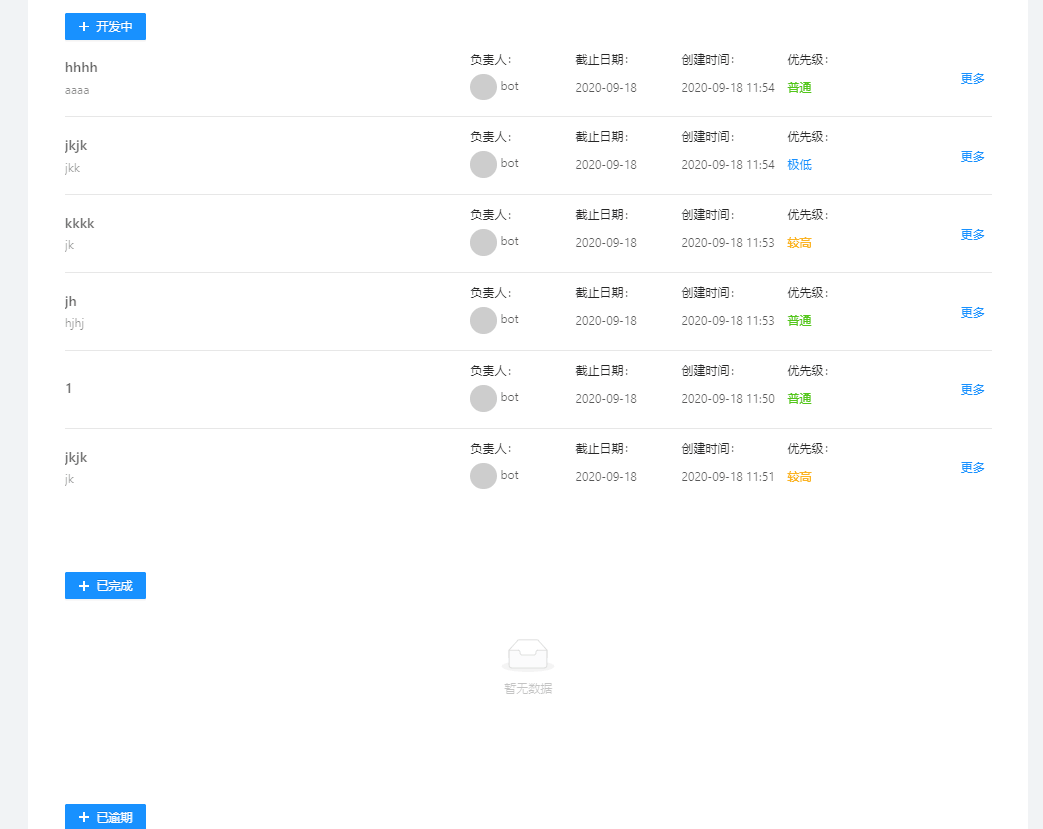
****

图2-11 图2-12 任务管理系统

前端相关代码：

tryCreateTask($event) {

this.loading = true

this.createTask($event.requestData)

.then((res) => {

console.log('创建任务成功')

})

.catch((err) => {

this.$notification.error({

message: err.message,

description: err.message,

})

})

.finally(() => {

this.closeDrawer()

this.loading = false

})

}

后端相关代码：

[HttpPost]

[Route("task")]

public JsonResult CreateTask(TaskRequest request)

{

var checkResult = CheckUsername(request.username);

if (checkResult != null)

{

return checkResult;

}

if (!CornerCaseCheckHelper.Check(request.projectId, 0,CornerCaseCheckHelper.Id))

{

returnJsonReturnHelper.ErrorReturn(701, "Invalid projectId.");

}

if (!CornerCaseCheckHelper.Check(request.taskName, 50,CornerCaseCheckHelper.Title))

{

returnJsonReturnHelper.ErrorReturn(601, "Invalid taskName.");

}

if (!CornerCaseCheckHelper.Check(request.description, 100,CornerCaseCheckHelper.Description))

{

returnJsonReturnHelper.ErrorReturn(602, "Invalid description.");

}

if (!CornerCaseCheckHelper.Check(request.principal, 50,CornerCaseCheckHelper.Username))

{

returnJsonReturnHelper.ErrorReturn(101, "Invalid principal.");

}

if (!CornerCaseCheckHelper.Check(request.deadline, 0, CornerCaseCheckHelper.Time))

{

returnJsonReturnHelper.ErrorReturn(610, "Invalid deadline.");

}

var user = \_meshContext.Users.First(u => u.Email ==request.username);

var project = \_meshContext.Projects.FirstOrDefault(p => p.Id ==request.projectId);

if (project == null)

{

returnJsonReturnHelper.ErrorReturn(707, "Project does not exist.");

}

if (\_permissionCheck.CheckProjectPermission(request.username,project)!=PermissionCheckHelper.ProjectAdmin)

{

returnJsonReturnHelper.ErrorReturn(701, "Permission denied.");

}

var taskBoard = \_meshContext.TaskBoards.First(b => b.ProjectId ==request.projectId);

var principalUser = \_meshContext.Users.FirstOrDefault(u => u.Email ==request.principal);

if (principalUser == null)

{

returnJsonReturnHelper.ErrorReturn(603, "Principal does not exist.");

}

var endTime = Convert.ToDateTime(request.deadline);

var task = new Task()

{

BoardId = taskBoard.Id,

Description =request.description,

LeaderId = principalUser.Id,

Name = request.taskName,

Priority = request.priority,

EndTime = endTime,

StartTime = DateTime.Now

};

*//Start Transaction to save the task*

using (var transaction = \_meshContext.Database.BeginTransaction())

{

try

{

\_meshContext.Tasks.Add(task);

\_meshContext.SaveChanges();

\_meshContext.TaskFeeds.Add(new TaskFeed()

{

UserId =principalUser.Id,

TaskId = task.Id

});

\_meshContext.SaveChanges();

transaction.Commit();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

returnJsonReturnHelper.ErrorReturn(1, "Unexpected error.");

}

}

return TaskResult(new TaskInfo()

{

Name = task.Name,

CreatedTime = task.CreatedTime,

Description = task.Description,

EndTime = task.EndTime,

Founder = user.Nickname,

Id = task.Id,

isFinished = task.Finished,

Principal =principalUser.Nickname,

Status =GetStatus(task.EndTime,task.Finished),

SubTasks = newList<SubTaskInfo>()

});

}

子任务管理系统



图2-14 子任务系统

前端相关代码：

createSubTask: ({ commit, rootGetters }, requestData) => {

return new Promise((resolve, reject) => {

sendRequest('createSubTask', requestData)

.then((res) => {

if (res.data.isSuccess) {

commit('ADD\_SUBTASK', {

parentTaskId: requestData.taskId,

newSubTask: res.data.subTask,

})

return resolve(res)

}

reject(new Error(res.data.msg))

})

.catch((err) => {

console.log('err from createSubProjectTasks is:', err)

reject(err)

})

})

},

后端相关代码：

[HttpDelete]

[Route("subtask")]

public JsonResult DeleteSubTask(string username, int projectId,int taskId,string subTaskName)

{

var checkResult = CheckUsername(username);

if (checkResult != null)

{

return checkResult;

}

if (!CornerCaseCheckHelper.Check(projectId, 0, CornerCaseCheckHelper.Id))

{

return JsonReturnHelper.ErrorReturn(701, "Invalid projectId.");

}

if (!CornerCaseCheckHelper.Check(taskId, 0, CornerCaseCheckHelper.Id))

{

return JsonReturnHelper.ErrorReturn(501, "Invalid taskId.");

}

if (!CornerCaseCheckHelper.Check(subTaskName, 50, CornerCaseCheckHelper.Title))

{

return JsonReturnHelper.ErrorReturn(511, "Invalid subTaskName.");

}

var user = \_meshContext.Users.First(u => u.Email == username);

var project = \_meshContext.Projects.FirstOrDefault(p => p.Id == projectId);

if (project == null)

{

return JsonReturnHelper.ErrorReturn(707, "Project does not exist.");

}

var task = \_meshContext.Tasks.FirstOrDefault(t => t.Id == taskId);

if (task == null)

{

return JsonReturnHelper.ErrorReturn(604, "Task does not exist.");

}

if (\_permissionCheck.CheckProjectPermission(username, project) != PermissionCheckHelper.ProjectAdmin &&

task.LeaderId != user.Id)

{

return JsonReturnHelper.ErrorReturn(701, "Permission denied.");

}

var subTask = \_meshContext.Subtasks.FirstOrDefault(s => s.TaskId == taskId && s.Title == subTaskName);

if (subTask == null)

{

return JsonReturnHelper.ErrorReturn(606, "SubTask does not exist.");

}

try

{

\_meshContext.Subtasks.Remove(subTask);

\_meshContext.SaveChanges();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

return JsonReturnHelper.ErrorReturn(1, "Unexpected error.");

}

return JsonReturnHelper.SuccessReturn();

}

* 1. 消息管理系统实现

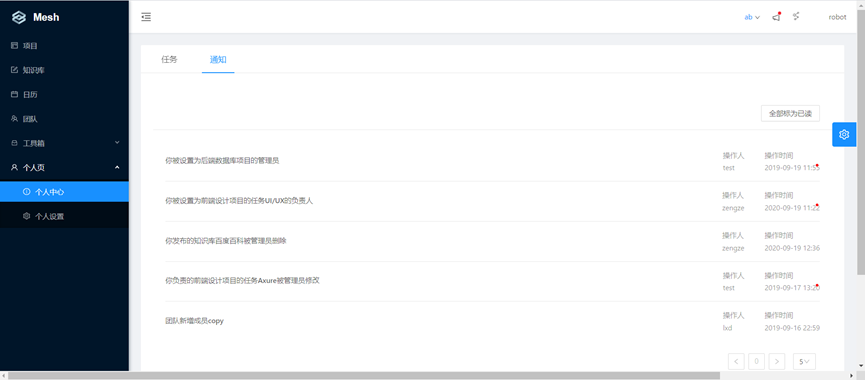


图2-13 消息管理系统

查看消息

前端相关代码：

queryNotification: ({ commit }, requestData) => {

return new Promise((resolve, reject) => {

sendRequest('queryNotification', requestData)

.then((response) => {

const { data } = response

if (data.isSuccess) commit('SET\_NOTIFICATIONS', data.notifications)

resolve(response)

})

.catch((error) => reject(error))

})

},

后端相关代码：

[HttpGet]

public JsonResult QueryNotification(string username)

{

var checkResult = CheckUsername(username);

if (checkResult != null)

{

return checkResult;

}

var user = \_meshContext.Users.First(u => u.Email==username);

var bulletinFeeds = \_meshContext.BulletinFeeds

.Where(f => f.UserId == user.Id)

.Join(\_meshContext.Bulletins, f => f.BulletinId, b => b.Id, (f, b) => b)

.Join(\_meshContext.BulletinBoards, b => b.BoardId, bb => bb.Id, (b, bb) => new

{

projectId = bb.ProjectId,

bulletin = b

})

.Join(\_meshContext.Projects, b => b.projectId, p => p.Id, (b, p) => new

{

project = p,

bulletin = b.bulletin

})

.Select(f=>new

{

teamId = f.project.TeamId,

projectId = f.project.Id,

title = f.bulletin.Title,

description = f.bulletin.Content,

createdTime = f.bulletin.CreatedTime,

}).ToList();

var taskFeeds = \_meshContext.TaskFeeds

.Where(f => f.UserId == user.Id)

.Join(\_meshContext.Tasks, f => f.TaskId, t => t.Id, (f, t) => t)

.Join(\_meshContext.TaskBoards, t => t.BoardId, b => b.Id, (t, b) => new

{

projectId = b.ProjectId,

task = t,

})

.Join(\_meshContext.Projects, b => b.projectId, p => p.Id, (b, p) => new

{

project = p,

task = b.task

})

.Select(f => new

{

teamId = f.project.TeamId,

projectId = f.project.Id,

title = f.task.Name,

description = f.task.Description,

createdTime = f.task.CreatedTime,

isFinished = f.task.Finished,

}).ToList();

return Json(new

{

err\_code = 0,

data = new

{

isSuccess = true,

msg = "",

notifications = new

{

bulletins = bulletinFeeds,

tasks = taskFeeds

}

}

});

}

* 1. 公告管理系统实现

发布公告



图2-16 公告系统实现

前端：

createBulletin({ commit, state }, requestData) {

return new Promise((resolve, reject) => {

sendRequest('createBulletin', requestData)

.then((res) => {

const { data } = res

if (data.isSuccess) {

commit('ADD\_BULLETIN', data.bulletin)

return resolve(res)

}

reject(new Error(res.data.msg))

})

.catch((err) => {

reject(err)

})

})

},

后端：

[HttpPost]

public JsonResult CreateBulletin(BulletinRequest request)

{

var checkUsername = CheckUsername(request.username);

if (checkUsername != null)

{

return checkUsername;

}

if (!CornerCaseCheckHelper.Check(request.bulletinName, 50, CornerCaseCheckHelper.Title))

{

return JsonReturnHelper.ErrorReturn(402, "Invalid bulletinName.");

}

if (!CornerCaseCheckHelper.Check(request.description, 100, CornerCaseCheckHelper.Description))

{

return JsonReturnHelper.ErrorReturn(403, "Invalid Description.");

}

if (!CornerCaseCheckHelper.Check(request.projectId, 0, CornerCaseCheckHelper.Id))

{

return JsonReturnHelper.ErrorReturn(401, "Invalid projectId.");

}

var project = \_meshContext.Projects.FirstOrDefault(u => u.Id == request.projectId);

if (project == null)

{

return JsonReturnHelper.ErrorReturn(411, "Project does not exist.");

}

*//Find bulletinBoard of this project*

var bulletinBoard = \_meshContext.BulletinBoards.First(b => b.ProjectId == request.projectId);

var user = \_meshContext.Users.First(u => u.Email == request.username);

if (\_permissionCheck.CheckProjectPermission(request.username,project)!=PermissionCheckHelper.ProjectAdmin)

{

return JsonReturnHelper.ErrorReturn(421, "Permission denied.");

}

*//Check if the bulletin already exists.*

var bulletin =

\_meshContext.Bulletins.FirstOrDefault(b => b.Title == request.bulletinName && b.BoardId == bulletinBoard.Id);

if (bulletin != null)

{

return JsonReturnHelper.ErrorReturn(411, "Bulletin already exists.");

}

*//Create the bulletin*

var newBulletin = new Bulletin()

{

Title = request.bulletinName,

Content = request.description,

BoardId = bulletinBoard.Id

};

*//Update feed*

var feedUsers = \_meshContext.Develops

.Where(d => d.ProjectId == request.projectId)

.ToList();

*//Start Transaction to save the bulletin*

using (var transaction = \_meshContext.Database.BeginTransaction())

{

try

{

\_meshContext.Bulletins.Add(newBulletin);

\_meshContext.SaveChanges();

var bulletinId = newBulletin.Id;

var feedList = new List<BulletinFeed>();

foreach (var feedUser in feedUsers)

{

feedList.Add(new BulletinFeed()

{

UserId = feedUser.UserId,

BulletinId = bulletinId

});

}

\_meshContext.AddRange(feedList);

\_meshContext.SaveChanges();

transaction.Commit();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

return JsonReturnHelper.ErrorReturn(1, "Unexpected error.");

}

}

return Json(new

{

err\_code = 0,

isSuccess = true,

msg = "",

bulletin = new

{

bulletinId = newBulletin.Id,

bullentinName = newBulletin.Title,

description = newBulletin.Content,

createTime = newBulletin.CreatedTime

}

});

}

删除公告

前端：

deleteBulletin({ commit }, requestData) {

return new Promise((resolve, reject) => {

sendRequest('deleteBulletin', requestData)

.then((res) => {

console.log('response in deleteBulletin: ', res)

const { data } = res

if (data.isSuccess) {

commit('REMOVE\_BULLETIN', requestData.bulletinId)

resolve(res)

}

reject(new Error(data.msg))

})

.catch((err) => {

console.error('error in deleteBulletin: ', err)

reject(err)

})

})

},

后端：

[HttpDelete]

public JsonResult DeleteBulletin(string username, int projectId, int bulletinId)

{

var checkResult = CheckUsername(username);

if (checkResult != null)

{

return checkResult;

}

if (!CornerCaseCheckHelper.Check(bulletinId, 0, CornerCaseCheckHelper.Id))

{

return JsonReturnHelper.ErrorReturn(401, "Invalid bulletinId.");

}

if (!CornerCaseCheckHelper.Check(projectId, 0, CornerCaseCheckHelper.Id))

{

return JsonReturnHelper.ErrorReturn(401, "Invalid projectId.");

}

*//Check if target bulletin exists*

var user = \_meshContext.Users.First(u => u.Email == username);

var project = \_meshContext.Projects.Find(projectId);

if (project == null)

{

return JsonReturnHelper.ErrorReturn(420, "Project does not exist.");

}

var bulletin = \_meshContext.Bulletins.Find(bulletinId);

if (bulletin == null)

{

return JsonReturnHelper.ErrorReturn(401, "Bulletin does not exist.");

}

if (project.AdminId != user.Id)

{

return JsonReturnHelper.ErrorReturn(422, "Permission denied.");

}

try

{

\_meshContext.Bulletins.Remove(bulletin);

\_meshContext.SaveChanges();

}

catch (Exception e)

{

\_logger.LogError(e.ToString());

return JsonReturnHelper.ErrorReturn(1, "Unexpected error.");

}

return JsonReturnHelper.SuccessReturn();

}

搜索公告

前端：

queryBulletin({ commit, state }, requestData) {

return new Promise((resolve, reject) => {

sendRequest('queryBulletin', requestData)

.then((res) => {

const { data } = res

if (data.isSuccess) {

commit('SET\_BULLETINS', data.bulletins)

console.log(state.bulletins)

}

resolve(res)

})

.catch((err) => reject(err))

})

},

后端：

[HttpGet]

public JsonResult QueryBulletin(string username, int projectId)

{

var checkUsername = CheckUsername(username);

if (checkUsername != null)

{

return checkUsername;

}

*//Find the target project*

var project = \_meshContext.Projects.FirstOrDefault(p => p.Id == projectId);

if (project == null)

{

return JsonReturnHelper.ErrorReturn(401, "Invalid projectId.");

}

var bulletins = \_meshContext.BulletinBoards

.Where(b => b.ProjectId == projectId)

.Join(\_meshContext.Bulletins, bb => bb.Id, b => b.BoardId, (bb, b) => new

{

Id = b.Id,

Name = b.Title,

Content = b.Content,

BoardId = b.BoardId,

CreatedTime = b.CreatedTime

}).ToList();

return Json(new

{

err\_code = 0,

data = new

{

isSuccess = true,

msg = "",

bulletins = bulletins

}

});

}

* 1. 工具箱的实现

工具箱提供了与项目相关的若干实用工具的入口聚合。如下图所示：

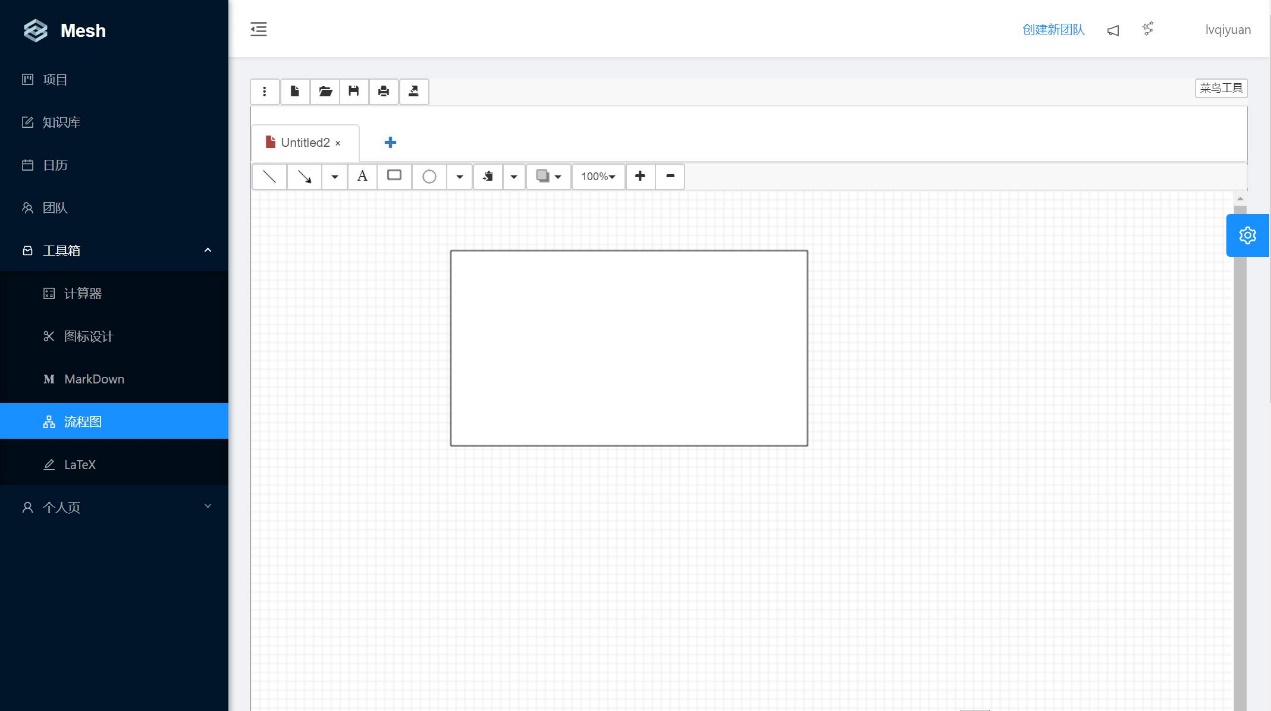


图2-17 流程图

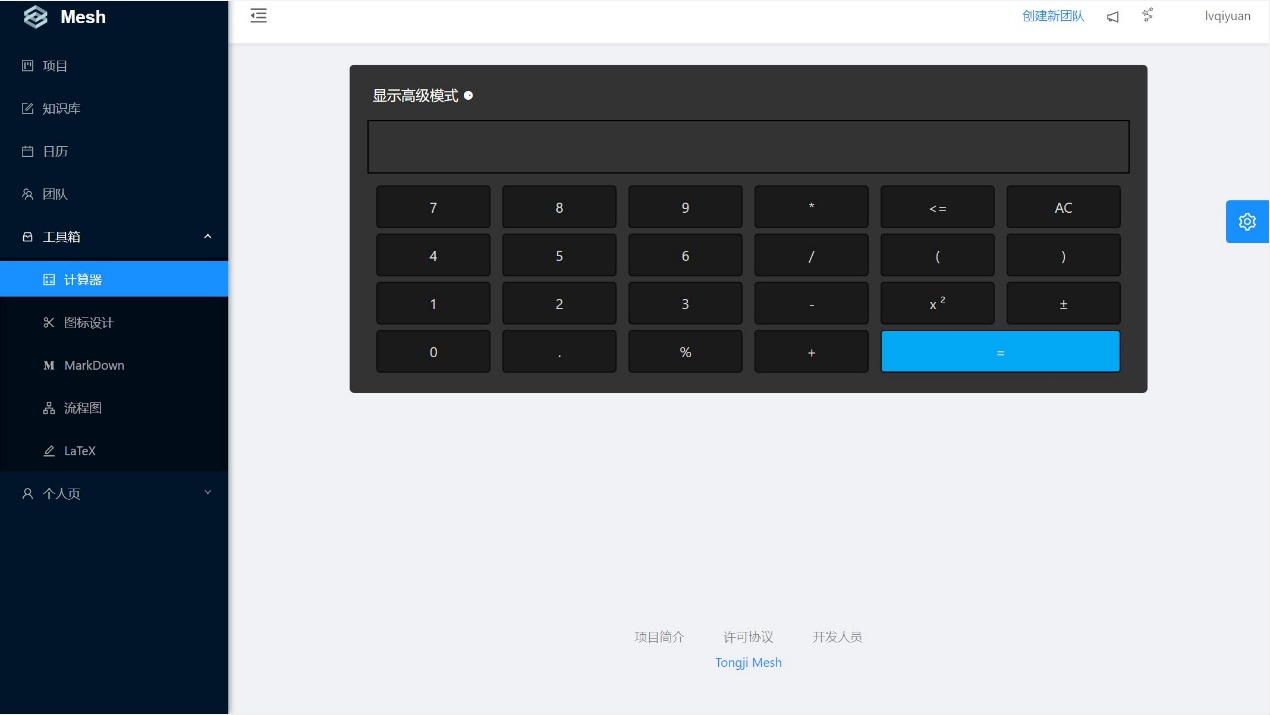


图2-18 计算器

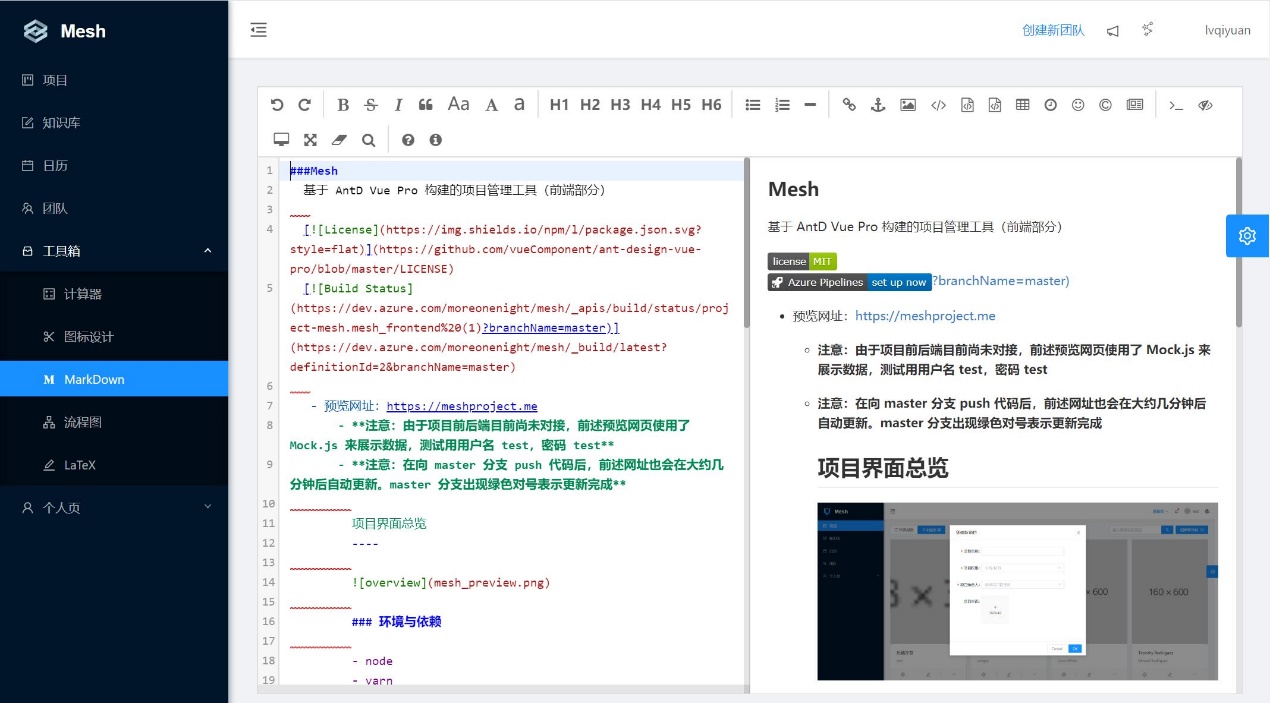


图2-19 Markdown编辑器

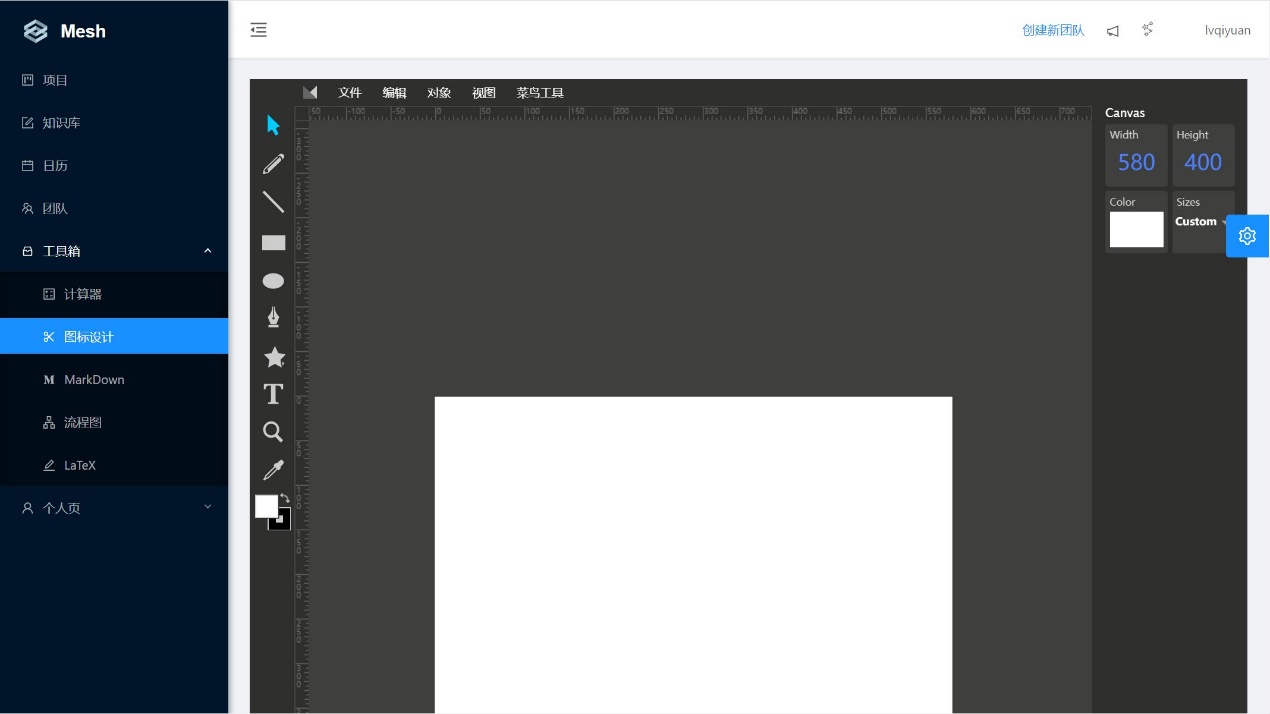


图2-20 图标设计

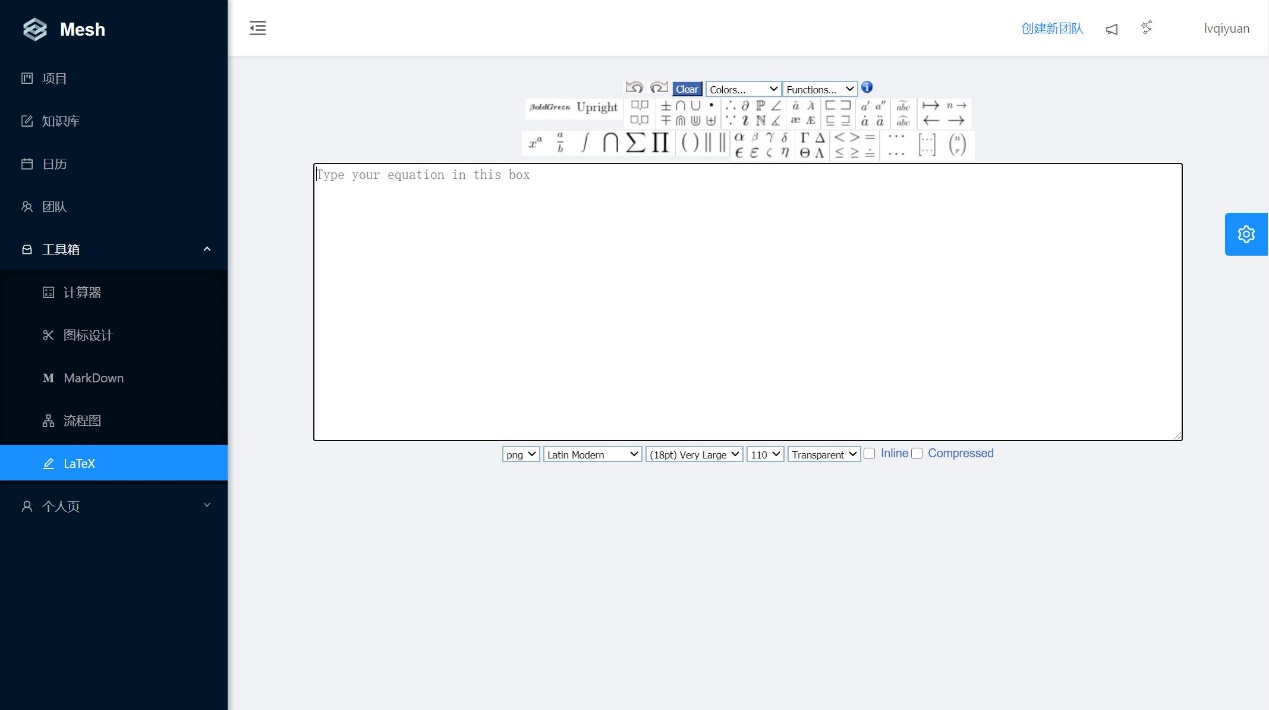


图2-21 LaTeX排版设计

1. 逻辑设计

本团队协作任务管理系统的数据库逻辑设计主要依据第一节中描述的系统数据需求进行。通过分析本系统需要抽象的实体和实体间的联系，并运用在数据库原理及应用课程中所学的知识，在分析对应关系、确定实体集属性和排除冗余后，小组成员构建出了数据库的总体E-R图和各模块E-R图，以图形化的方式表现出了本系统数据库的逻辑设计。

* 1. 总体E-R图

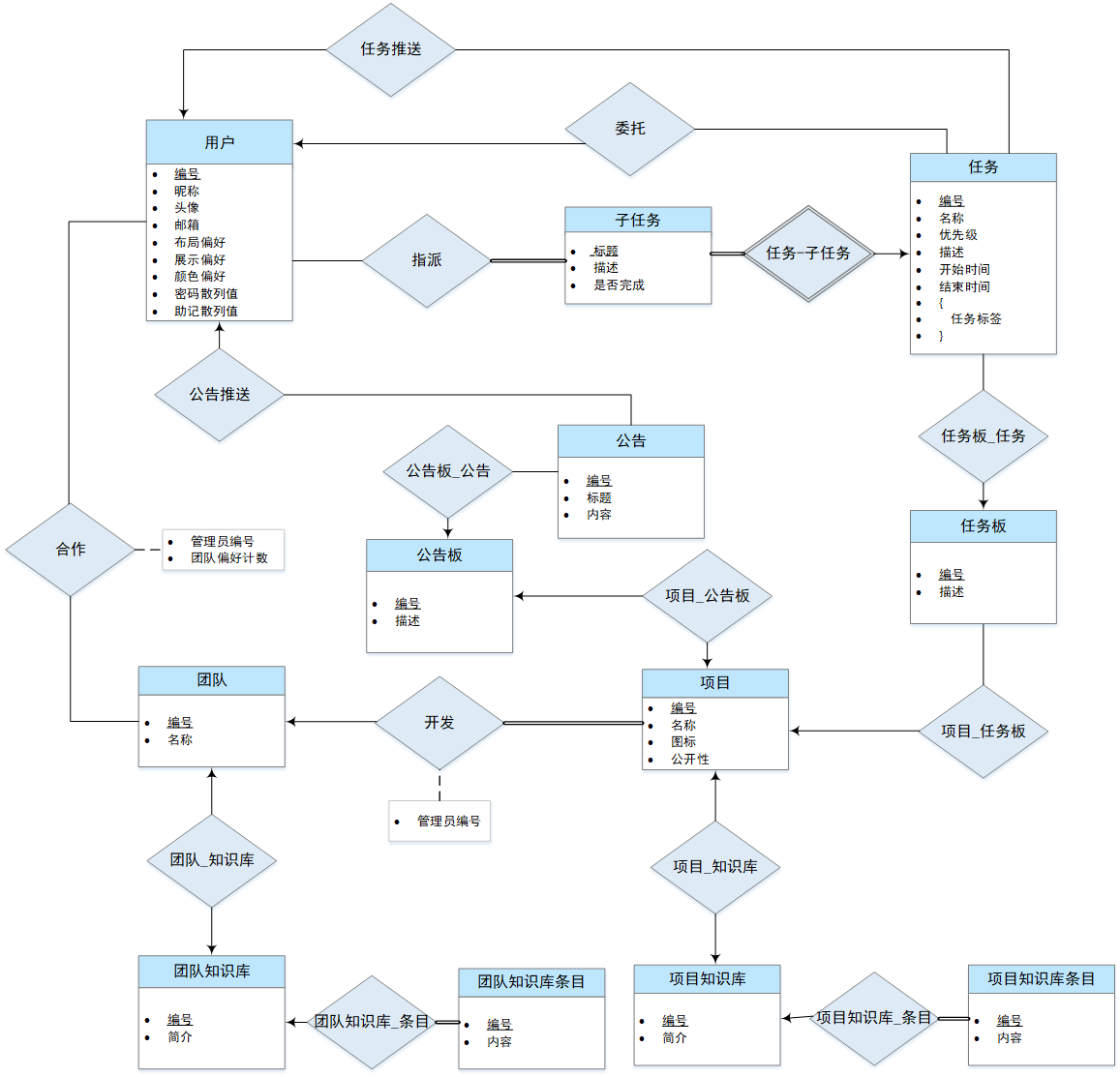


图3-1 总体E-R图

图中各实体及联系的文字说明参见3.2中的模块E-R图。

* 1. 模块E-R图

团队管理模块E-R图

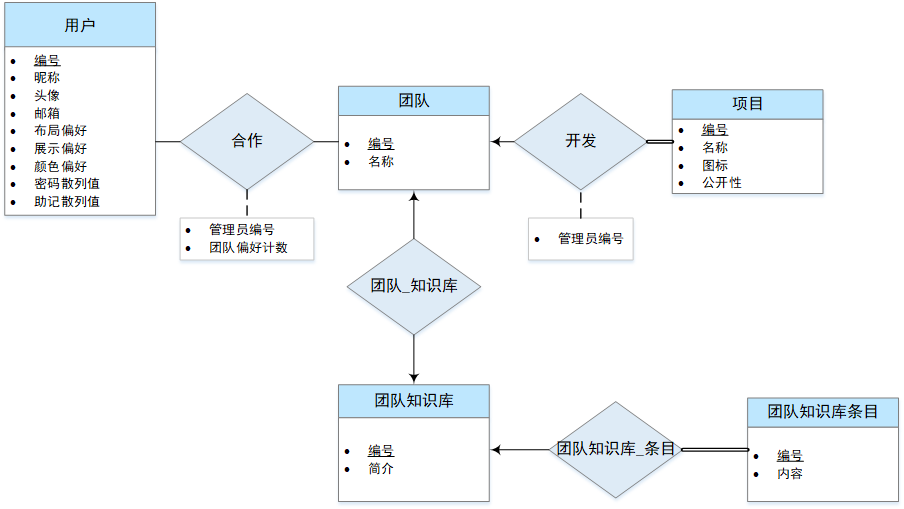


图3‑2 团队管理模块E-R图

**实体**

1. 用户：保存用户的基本信息，属性包括id、昵称、头像、邮箱、布局偏好、展示偏好、颜色偏好、密码散列值、助记散列值
2. 团队：保存团队的基本信息，属性包括id、名称
3. 项目：保存开发一个项目的基本信息，属性包括id、名称、图标、权限（公开性）
4. 团队知识库：用于存储各个团队的知识库，属性包括id和简介
5. 团队知识库条目：存储团队知识库中的具体条目内容，属性包括id和具体内容

**联系**

1. 合作：

用户与团队之间的联系，用户自发组成团队，这些组成团队的用户之间存在合作关系。其属性包括了合作关系中管理员的编号。由于一个用户可以参与不止一个团队，一个团队可以有多个用户，故采取多对多关系。

1. 开发：

团队和项目之间的关系，表示一个团队开发项目。其属性包括了项目管理员的编号。由于一个团队可以对多个项目进行开发，而一个项目只能由一个团队来开发，所以采取一对多的关系，且项目在联系集中全部参与。

1. 团队\_知识库：

团队和知识库之间的关系，表示团队拥有知识库。由于一个团队只能有一个知识库且一个知识库只能属于一个团队，故采用一对一关系。

1. 团队知识库\_条目：

团队知识库和团队知识库条目之间的关系，表示团队知识库拥有团队知识库条目，由于一个团队知识库可以拥有多个团队知识库条目，但一个团队知识库条目只能且必须对应一个团队知识库，故采用一对多的关系，且知识库条目全部参与。

项目管理模块E-R图

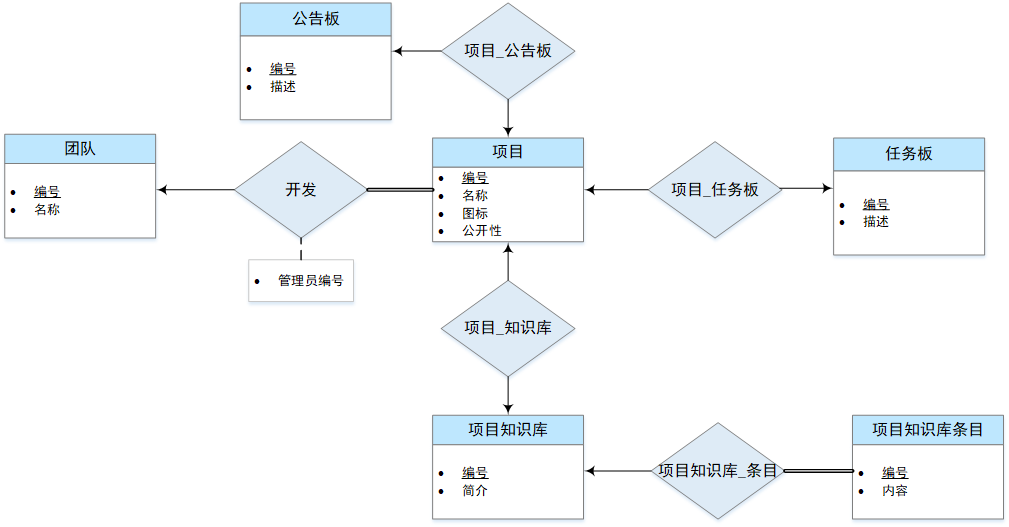


图3‑3 项目管理模块E-R图

**实体**

1. 团队：保存团队的基本信息，属性包括id、名称
2. 项目：保存开发一个项目的基本信息，属性包括id、名称、图标、权限（公开性）
3. 任务板：保存项目的任务板信息，属性包括id、描述
4. 公告板：保存项目的公告板信息，属性包括id、描述
5. 项目知识库条目：存储项目知识库中的具体条目内容，属性包括id和具体内容
6. 项目知识库：用于存储各个项目的知识库，属性包括id和简介

**联系**

1. 开发：

团队和项目之间的关系，表示一个团队开发项目。其属性包括了项目管理员的编号。由于一个团队可以对多个项目进行开发，而一个项目只能由一个团队来开发，所以采取一对多的关系，且项目在联系集中全部参与。

1. 项目\_公告板：

项目和公告板之间的关系，表示一个团队拥有一个公告板。由于一个项目只能有一个公告板，一个公告板只属于一个项目，所以采取一对一的关系。

1. 项目\_任务板：

项目和任务板之间的关系，表示一个团队拥有一个任务板。由于一个项目只能有一个任务板，一个任务板只属于一个项目，所以采取一对一的关系。

1. 项目\_知识库：

项目和知识库之间的关系，表示项目拥有知识库。由于一个项目只能有一个知识库且一个知识库只能属于一个项目，故采用一对一关系。

1. 项目知识库\_条目：

项目知识库和项目知识库条目之间的关系，表示项目知识库拥有项目知识库条目，由于一个项目知识库可以拥有多个项目知识库条目，但一个项目知识库条目只能且必须对应一个项目知识库，故采用一对多的关系，且知识库条目全部参与。

任务管理模块E-R图

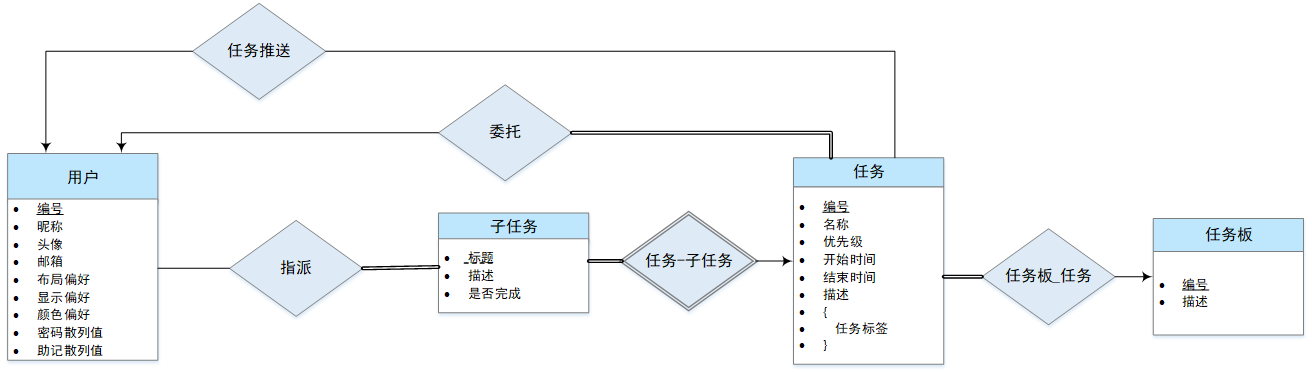


图3‑4 任务管理模块E-R图

**实体**

1. 用户：保存用户的基本信息，属性包括id、昵称、头像、邮箱、布局偏好、展示偏好、颜色偏好、密码散列值、助记散列值
2. 任务板：保存项目的任务板信息，属性包括id、描述
3. 任务：保存任务信息和任务状态，属性包括id、名称、优先级、开始时间、结束时间、描述、任务标签。其中{任务标签}为多值属性
4. 子任务：保存子任务的描述和完成状态，属性包括标题、描述、是否完成。子任务是弱实体集，其依赖于任务这一属主实体集

**联系**

1. 任务板\_任务：

任务板和任务之间的关系，表示任务板展示任务，由于一个任务板可以展示多项任务，但每项任务必须且只能在一个任务板上展示，故采取一对多的关系，且任务全部参与本联系。

1. 任务\_子任务：

任务和子任务之间的关系，表示一项任务对应分解为多项子任务。由于一项任务可以分解为多项子任务，且每项子任务必须且只能包含在一项任务中，故采取从子任务到任务多对一的关系，且子任务全部参与这一标识性联系。每项子任务都必须通过任务\_子任务联系与一项任务关联。

1. 任务推送：

任务和用户之间的关系，表示一项任务会被推送给需要完成该任务的用户。由于一项任务是按单个用户逐用户推送，且每个用户可以进行若干项任务，故采取用户到任务一对多的关系。

1. 委托：

用户和任务之间的关系，表示用户可以委托一项任务由其他用户完成。由于一名用户可以委托多项任务，且每项任务必须且只能由一名用户委托，故采取用户到任务一对多的关系，且任务全部参与本联系。

1. 指派：

用户和任务之间的关系，表示用户可以指派子任务。由于一名用户可以指派多项子任务，且每项子任务必须且只能由一名用户指派，故采取用户到子任务一对多的关系，且子任务全部参与本联系。

公告管理模块E-R图

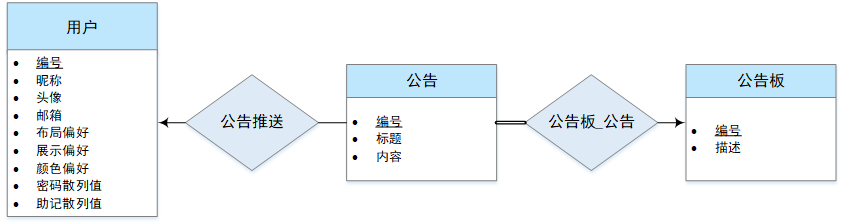


图3‑5 公告管理模块E-R图

**实体**

1. 用户：保存用户的基本信息，属性包括id、昵称、头像、邮箱、布局偏好、展示偏好、颜色偏好、密码散列值、助记散列值
2. 公告板：保存项目的公告板信息，属性包括id、描述
3. 公告：保存公告板中的公告信息，属性包括id、标题、内容

**联系**

1. 公告推送：

公告和用户之间的关系，表示一项公告会被推送给参与项目的用户。由于一项公告是按单个用户逐用户推送，且每个用户可以接收若干项公告，故采取用户到公告一对多的关系。

1. 公告板\_公告：

公告板和公告之间的关系，表示公告板展示公告，由于一个公告板可以展示多项公告，但每项公告必须且只能在一个公告板上展示，故采取公告板到公告一对多的关系，且公告全部参与本联系。

消息管理模块E-R图

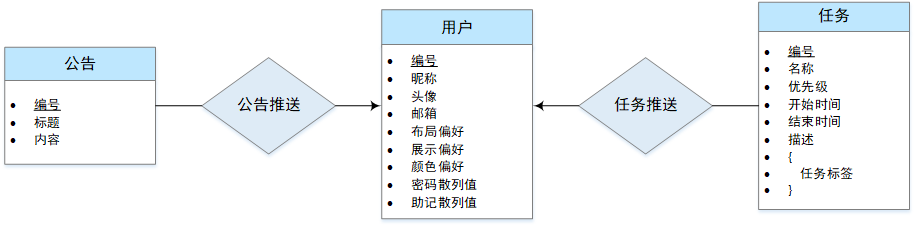


图3‑6 消息管理模块E-R图

**实体**

1. 用户：保存用户的基本信息，属性包括id、昵称、头像、邮箱、布局偏好、展示偏好、颜色偏好、密码散列值、助记散列值
2. 公告：保存公告板中的公告信息，属性包括id、标题、内容
3. 任务：保存任务信息和任务状态，属性包括id、名称、优先级、开始时间、结束时间、描述、任务标签。其中{任务标签}为多值属性

**联系**

1. 任务推送：

任务和用户之间的关系，表示一项任务会被推送给需要完成该任务的用户。由于一项任务是按单个用户逐用户推送，且每个用户可以进行若干项任务，故采取用户到任务一对多的关系。

1. 公告推送

公告和用户之间的关系，表示一项公告会被推送给参与项目的用户。由于一项公告是按单个用户逐用户推送，且每个用户可以接收若干项公告，故采取用户到公告一对多的关系。

* 1. 数据库表的设计

用户表

表格3‑1 用户表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| ID | integer |  | 用户在数据库中的id | 主键 |
| nickname | varchar | 50 | 用户的网名 |  |
| email | varchar | 50 | 用户的电子邮箱 |  |
| password\_digest | varchar | 70 | 用户登录密码的散列值 |  |
| remember\_digest | varchar | 70 | 记录用户登录信息的散列值 |  |
| avatar | varchar | 2048 | 用户头像的URL |  |
| ColorPreference | varchar | 50 | 用户偏好颜色 |  |
| LayoutPreference | varchar | 50 | 用户偏好布局 |  |
| RevealedPreference | varchar | 50 | 用户偏好显示 |  |

任务表

表格3‑2 任务表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| ID | integer |  | 任务的id | 主键 |
| board\_ID | integer |  | 任务所属任务栏的id | 外键，参照任务栏表的ID |
| name | varchar | 50 | 任务名 |  |
| description | varchar | 100 | 任务描述 |  |
| priority | integer |  | 任务优先级 |  |
| start\_time | datetime |  | 任务开始时间 |  |
| end\_time | datetime |  | 任务结束时间 |  |
| leader\_ID | integer |  | 任务负责人的id | 外键，参照用户表的ID |
| finished | bit | 1 | 任务的完成状态 |  |

子任务表

表格3‑3 子任务表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| task\_ID | integer |  | 子任务对应任务的id | 主键  外键，参照任务的ID |
| title | varchar | 50 | 子任务标题 | 主键 |
| description | varchar | 100 | 子任务描述 |  |
| finished | bit | 1 | 子任务的完成状态 |  |

任务标签表

表格3‑4 标签表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| task\_ID | integer |  | 任务的id | 主键  外键，参照任务的ID |
| tag | varchar | 20 | 标签的内容 |  |

团队表

表格3‑5 团队表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| ID | integer |  | 团队的id | 主键 |
| name | varchar | 50 | 团队名 |  |
| admin\_ID | integer |  | 团队负责人的id | 外键，参照用户表的ID |

项目表

表格3‑6 项目表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| ID | integer |  | 项目的id | 主键 |
| team\_ID | integer |  | 项目所属的团队 | 外键，参照团队表的ID |
| name | varchar | 50 | 项目名 |  |
| icon | varchar | 2048 | 项目图标的URL |  |
| admin\_ID | integer |  | 项目负责人的id | 外键，参照用户表的ID |
| Publicity | tinyint | 1 | 项目是否为公开 |  |

公告表

表格3‑7 公告表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| ID | integer |  | 公告的id | 主键 |
| board\_ID | integer |  | 公告所处公告栏的id | 外键，参照公告板表的ID |
| title | varchar | 50 | 公告的标题 |  |
| content | varchar | 100 | 公告的内容 |  |

参与表

表格3‑8 参与表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| user\_ID | integer |  | 参与关系中用户的id | 主键  外键，参照用户表的ID |
| team\_ID | integer |  | 参与关系中团队的id | 主键  外键，参照团队表的ID |

公告栏表

表格3‑9 公告栏表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| ID | integer |  | 公告栏的id | 主键 |
| project\_ID | integer |  | 公告栏所属项目的id | 外键，参照项目表的ID |
| description | varchar | 100 | 公告栏需知 |  |

任务栏表

表格3‑10 任务栏表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| ID | integer |  | 任务栏的id | 主键 |
| project\_ID | integer |  | 任务栏所属项目的id | 外键，参照项目表的ID |
| description | varchar | 100 | 任务栏描述 |  |

任务推送表

表格3‑11 任务推送表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| task\_ID | integer |  | 任务的id | 主键  外键，参照任务表的ID |
| user\_ID | integer |  | 被推送任务信息接收者的id | 外键，参照用户表的ID |

分配表

表格3‑12 分配表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| task\_ID | integer |  | 子任务所属任务的id | 主键  外键，参照子任务表的ID |
| title | varchar | 50 | 子任务的标题 | 主键  外键，参照子任务表的title |
| user\_ID | integer |  | 子任务负责人的id | 主键  外键，参照用户表的ID |

公告推送表

表格3‑13 公告推送表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| bulletin\_ID | integer |  | 公告的id | 主键  外键，参照任务表的ID |
| user\_ID | integer |  | 被推送公告信息接收者的id | 外键，参照用户表的ID |

管理员表

表格3‑14 管理员表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| ID | integer |  | 管理员在数据库中的id | 主键 |
| nickname | varchar | 50 | 管理员的网名 |  |
| email | varchar | 50 | 管理员的电子邮箱 |  |
| password\_digest | varchar | 70 | 管理员登录密码的散列值 |  |
| remember\_digest | varchar | 70 | 记录管理员登录信息的散列值 |  |

团队知识库表

表格3‑15 团队知识库表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| ID | integer |  | 团队知识库的id | 主键 |
| team\_ID | integer |  | 团队知识库所属团队的id | 主键  外键，参照团队表的title |
| description | varchar | 100 | 团队知识库描述 |  |

项目知识库表

表格3‑16 项目知识库表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| ID | integer |  | 项目知识库的id | 主键 |
| team\_ID | integer |  | 项目知识库所属项目的id | 主键  外键，参照项目表的title |
| description | varchar | 100 | 项目知识库描述 |  |

团队知识库条目表

表格3‑17 团队知识库条目表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| ID | integer |  | 团队知识库条目的id | 主键 |
| title | varchar | 50 | 团队知识库条目的标题 |  |
| text | varchar | 100 | 团队知识库条目的内容 |  |
| collection\_ID | integer |  | 该条目所属的团队知识库的id | 外键，参照团队知识库表的ID |
| user\_ID | integer |  | 上传用户的id | 外键，参照用户表的ID |

项目知识库条目表

表格3‑18 项目知识库条目表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| ID | integer |  | 项目知识库条目的id | 主键 |
| title | varchar | 50 | 项目知识库条目的标题 |  |
| text | varchar | 100 | 项目知识库条目的内容 |  |
| collection\_ID | integer |  | 该条目所属的项目知识库的id | 外键，参照团队知识库表的ID |
| user\_ID | integer |  | 上传用户的id | 外键，参照用户表的ID |

备注

除上述各表所列的字段外，每张表都还具有创建时间和最后修改时间的记录。

表格3‑19 备注

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 数据类型 | 长度 | 说明 | 备注 |
| created\_time | datetime |  | 创建时间 |  |
| updated\_time | datetime |  | 最后修改时间 |  |

* 1. 数据库关系图

整理上述18张表，用下划线和箭头分别表示其中的主码、外码关系，所得到的数据库关系图如下所示：

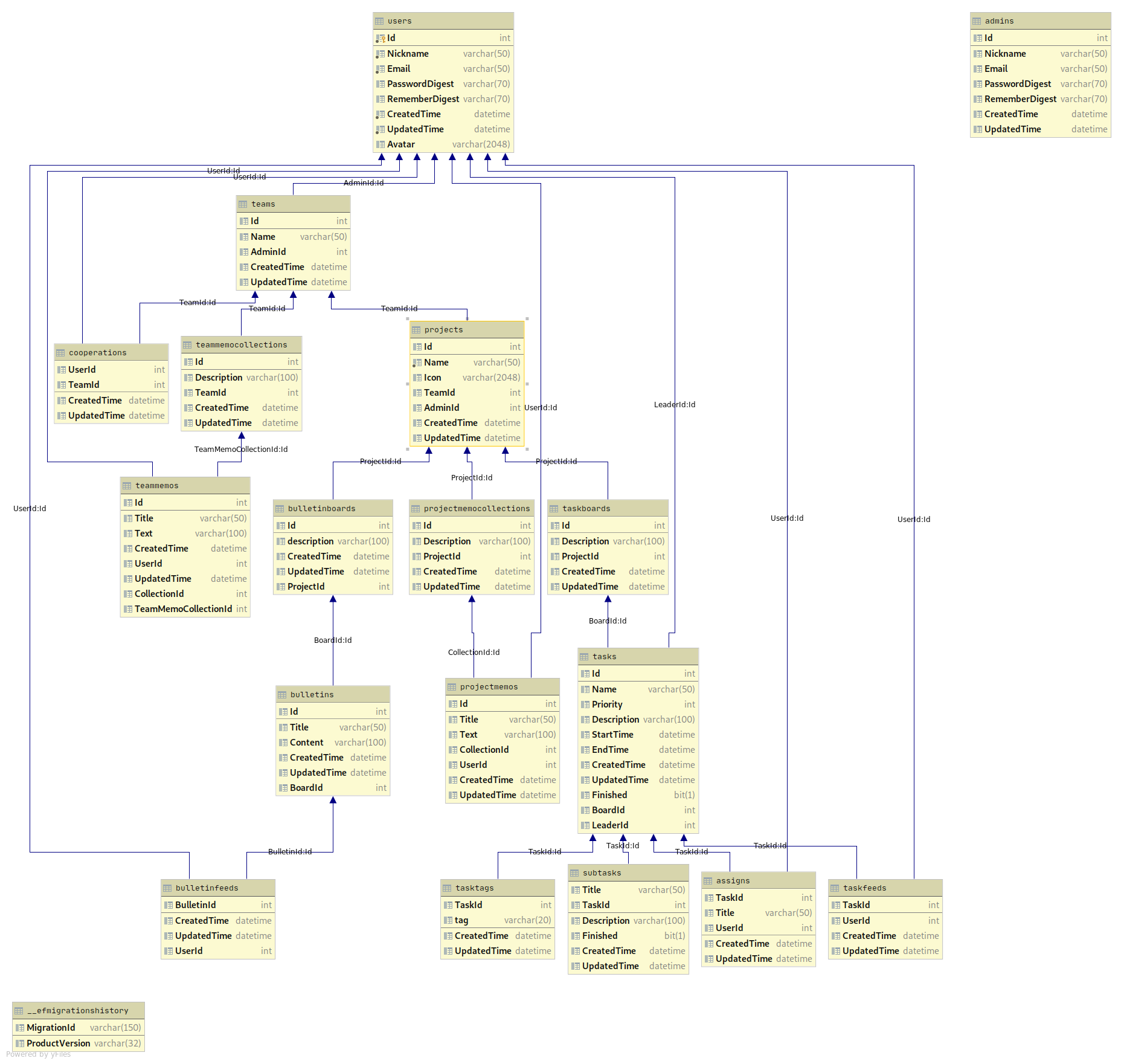


图3-1 数据库关系图

1. API文档
   1. 用户功能

用户登录

**简要描述：**

* 用户登录接口
* 权限：all

**请求 URL：**

* http://xx.com/api/mesh/login

**请求方式：**

* POST

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| password | 是 | String | 密码 |
| token | 否 | String | 已获取的 token |

**注意，这里要么是通过用户名加密码登录，要么是通过已经获取的 token 登录，即 Data 中要么包含 username + password，要么包含 token**

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"username": "王泽钜",

"nickname": "软件之王",

"gender": 1,

"status": 1,

"address": "河北省 保定市 某某区",

"description": "我是天命之灭世者万物之终结者软件之王",

"birthday": "yyyy-MM-DD",

"avatar": "xxxxxxxx",

"role": "user",

"token": "sdfjlklsfssf",

"preference": {

"preferenceShowMode": "card",

"preferenceColor": "blue",

"preferenceLayout": "Default",

"preferenceTeam": "asdgsdfgsdf"

},

"teams": [

{

"teamName": "数据库课程团队",

"teamId": "sadgfasdhfawejkfak",

"adminName": "王泽钜"

}

]

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功登录 |
| msg | String | 错误信息，若成功返回空 |
| username | String | 用户名 |
| nickname | String | 昵称 |
| gender | Number | 性别，0为无性别，1为男性，2位女性 |
| status | Number | 状态tag，0为无状态，1-8分别为摸鱼，会议，生病，休假，出差，在家，运动，游戏 |
| address | String | 地址，可以为空 |
| description | String | 用户个人描述 |
| birthday | String | 生日，可以为空 |
| avatar | String | 用户头像 |
| preferenceShowMode | String | 用户使用模式的偏好,card 代表卡片模式,list 代表列表模式（默认 card） |
| role | String | 用户角色 |
| token | String | token |
| preference | Object | 用户偏好 |
| preferenceColor | String | 用户偏好主题色（默认 blue） |
| preferenceLayout | String | 用户偏好布局（默认 Default） |
| preferenceTeam | String | 用户偏好的团队 Id |
| teams | Object | 用户参与的团队 |
| teamId | String | 团队 Id |
| teamName | String | 团队名称 |
| adminName | String | 团队管理员用户名 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

用户注册

**简要描述：**

* 用户注册接口
* 权限：all

**请求 URL：**

* http://xx.com/api/mesh/register

**请求方式：**

* POST

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| password | 是 | String | 密码 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"username": "王泽钜",

"nickname": "王泽钜",

"gender": 0,

"status": 0,

"address": "",

"description": "",

"birthday": "",

"avatar": "xxxxxxxx",

"role": "user",

"token": "sdfjlklsfssf",

"preference": {

"preferenceShowMode": "card",

"preferenceColor": "blue",

"preferenceLayout": "Default",

"preferenceTeam": "asdgsdfgsdf"

},

"teams": []

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功登录 |
| msg | String | 错误信息，若成功返回空 |
| username | String | 用户名 |
| nickname | String | 昵称，注册时默认与用户名相同 |
| gender | Number | 性别，0为无性别，1为男性，2位女性 |
| status | Number | 状态tag，0为无状态，1-8分别为摸鱼，会议，生病，休假，出差，在家，运动，游戏 |
| address | String | 地址，可以为空 |
| description | String | 用户个人描述 |
| birthday | String | 生日，可以为空 |
| avatar | String | 用户头像 |
| preferenceShowMode | String | 用户使用模式的偏好,card 代表卡片模式,list 代表列表模式（默认 card） |
| role | String | 用户角色 |
| token | String | token |
| preference | Object | 用户偏好 |
| preferenceColor | String | 用户偏好主题色（默认 blue） |
| preferenceLayout | String | 用户偏好布局（默认 Default） |
| preferenceTeam | String | 用户偏好的团队 Id |
| teams | Object | 用户参与的团队 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

查询用户

**简要描述：**

* 根据关键词查询用户（注意：这条 api 返回的 users 数组仅包含普通用户，不包含管理员）
* 权限：所有用户

**请求 URL：**

* http://xx.com/api/mesh/user

**请求方式：**

* GET

**Query Params：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| keyword | 是 | String | 关键词 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"users": [

{

"username": "王泽钜",

"avatar": "xxxxxxxx",

"nickname": "软件之王",

"gender": 1,

"status": 1,

"address": "河北省 保定市 某某区",

"description": "我是天命之灭世者万物之终结者软件之王",

"birthday": "yyyy-MM-DD"

}

]

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功加入队伍 |
| msg | String | 错误信息，若成功返回空 |
| project | Object | 查询返回的项目信息 |
| username | String | 用户名 |
| avatar | String | 用户头像 |
| nickname | String | 昵称 |
| gender | Number | 性别，0 为无性别，1 为男性，2 位女性 |
| status | Number | 状态 tag，0 为无状态，1-8 分别为摸鱼，会议，生病，休假，出差，在家，运动，游戏 |
| address | String | 地址 |
| description | String | 用户个人描述 |
| birthday | String | 生日 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

用户修改个人信息

**简要描述：**

* 用户修改个人信息
* 权限：user

**请求 URL：**

* `<http://xx.com/api/mesh/user>

**请求方式：**

* PATCH

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| nickname | 否 | String | 昵称 |
| gender | 否 | Number | 性别 |
| status | 否 | Number | 状态tag |
| address | 否 | String | 地址 |
| description | 否 | String | 用户个人描述 |
| birthday | 否 | String | 生日 |
| avatar | 否 | String | 头像，base64编码 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"username": "王泽钜",

"nickname": "软件之王",

"gender": 1,

"status": 1,

"address": "河北省 保定市 某某区",

"description": "我是天命之灭世者万物之终结者软件之王",

"birthday": "yyyy-MM-DD",

"avatar": "xxxxxxxx",

"role": "user",

"token": "sdfjlklsfssf",

"preference": {

"preferenceShowMode": "card",

"preferenceColor": "blue",

"preferenceLayout": "Default",

"preferenceTeam": "asdgsdfgsdf"

},

"teams": [

{

"teamName": "数据库课程团队",

"teamId": "sadgfasdhfawejkfak",

"adminName": "王泽钜"

}

]

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功修改 |
| msg | String | 返回错误信息，若成功则返回空， |
| username | String | 用户名 |
| nickname | String | 昵称 |
| gender | Number | 性别，0为无性别，1为男性，2位女性 |
| status | Number | 状态tag，0为无状态，1-8分别为摸鱼，会议，生病，休假，出差，在家，运动，游戏 |
| address | String | 地址，可以为空 |
| description | String | 用户个人描述 |
| birthday | String | 生日，可以为空 |
| avatar | String | 用户头像 |
| preferenceShowMode | String | 用户使用模式的偏好,card 代表卡片模式,list 代表列表模式（默认 card） |
| role | String | 用户角色 |
| token | String | token |
| preference | Object | 用户偏好 |
| preferenceColor | String | 用户偏好主题色（默认 blue） |
| preferenceLayout | String | 用户偏好布局（默认 Default） |
| preferenceTeam | String | 用户偏好的团队 Id |
| teams | Object | 用户参与的团队 |
| teamId | String | 团队 Id |
| teamName | String | 团队名称 |
| adminName | String | 团队管理员用户名 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

用户修改密码

**简要描述：**

* 用户修改密码
* 权限：user

**请求 URL：**

* `<http://xx.com/api/mesh/user/password>

**请求方式：**

* PATCH

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名称 |
| oldPassword | 是 | String | 原始密码 |
| password | 是 | String | 新密码 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": ""

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功修改 |
| msg | String | 返回错误信息，若成功则返回空， |

**备注**

* 更多返回错误代码请看首页的错误代码描述

管理员修改普通用户密码

**简要描述：**

* 管理员修改普通用户密码
* 权限：admin

**请求 URL：**

* `<http://xx.com/api/mesh/admin/password>

**请求方式：**

* PATCH

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名称 |
| password | 是 | String | 新密码 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": ""

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功修改 |
| msg | String | 返回错误信息，若成功则返回空， |

**备注**

* 更多返回错误代码请看首页的错误代码描述
  1. 团队功能

以当前用户为团队管理员创建一个新团队

**简要描述：**

* 以当前用户为团队管理员创建一个新团队
* http://xx.com/api/mesh/team

**请求方式：**

* POST

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| teamName | 是 | String | 团队名称 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"team": {

"teamId": "asdfhgdsfxbsd",

"createTime": "1595217702138",

"teamName": "后端开发",

"adminName": "王泽钜",

"members": [

{

"username": "王泽钜",

"avatar": "xxxxxxxxxx"

}

],

"teamProjects": []

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功创建团队 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |
| team | Obejct | 团队信息 |
| teamId | String | 团队 Id |
| createTime | Number | 团队创建时间时间戳 |
| teamName | String | 团队的名称 |
| adminName | String | 团队负责人 |
| members | Array | 项目成员 |
| username | String | 成员名 |
| avatar | String | 成员头像 |
| teamProjects | Array | 团队下属的项目，是对象数组，数组中每一个对象由 projectName 和 projectLogo 两个属性组成 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

团队管理员邀请新成员加入该团队

**简要描述：**

* 团队管理员邀请新成员加入该团队
* 权限：团队管理员
* http://xx.com/api/mesh/team/invite

**请求方式：**

* POST

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| teamId | 是 | String | 团队 Id |
| inviteName | 是 | String | 受邀人用户名 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": ""

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功发出邀请 |
| msg | String | 返回错误信息，若成功则返回空 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

用户加入某个团队(必须是被团队管理员邀请过的)

**简要描述：**

* 当前用户加入某个团队(必须是被团队管理员邀请过的)
* http://xx.com/api/mesh/team/join

**请求方式：**

* POST

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| teamId | 是 | String | 团队 Id |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"team": {

"teamId": "asdfhgdsfxbsd",

"createTime": "1595217702138",

"teamName": "前端开发",

"adminName": "王泽钜",

"members": [

{ "username": "王泽钜", "avatar": "xxxxxxx" },

{ "username": "刘学迪", "avatar": "xxxxxxxx" },

{ "username": "王新宇", "avatar": "xxxxxxxxxxx" }

],

"teamProjects": [

{

"projectName": "后端开发",

"projectId": "asdgsdfhlaksdf",

"projectLogo": "xxxxxxx",

"adminName": "王泽钜"

},

{

"projectName": "前端开发",

"projectId": "asdgsdfhlaksdf",

"projectLogo": "xxxxxxx",

"adminName": "刘学迪"

}

]

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功加入队伍 |
| msg | String | 错误信息，若成功返回空 |
| team | Obejct | 团队信息 |
| teamId | String | 团队 Id |
| createTime | Number | 团队创建时间时间戳 |
| teamName | String | 团队的名称 |
| adminName | String | 团队负责人 |
| members | Array | 项目成员 |
| username | String | 成员名 |
| avatar | String | 成员头像 |
| teamProjects | Array | 团队下属的项目，是对象数组，数组中每一个对象由 projectName 和 projectLogo 两个属性组成 |
| projectName | String | 项目名称 |
| projectId | String | 项目 Id |
| projectLogo | String | 项目 logo |
| adminName | String | 项目管理员 Id |

**备注**

* 更多返回错误代码请看首页的错误代码描述

查询当前项目信息

**简要描述：**

* 查询当前项目信息
* http://xx.com/api/mesh/team

**请求方式：**

* GET

**Query Params：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| teamId | 是 | String | 团队 Id |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"team": {

"teamId": "asdfhgdsfxbsd",

"createTime": "1595217702138",

"teamName": "前端开发",

"adminName": "王泽钜",

"members": [

{ "username": "王泽钜", "avatar": "xxxxxxx" },

{ "username": "刘学迪", "avatar": "xxxxxxxx" },

{ "username": "王新宇", "avatar": "xxxxxxxxxxx" }

],

"teamProjects": [

{

"projectName": "后端开发",

"projectId": "asdgsdfhlaksdf",

"projectLogo": "xxxxxxx",

"adminName": "王泽钜"

},

{

"projectName": "前端开发",

"projectId": "asdgsdfhlaksdf",

"projectLogo": "xxxxxxx",

"adminName": "刘学迪"

}

]

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功加入队伍 |
| msg | String | 错误信息，若成功返回空 |
| team | Obejct | 团队信息 |
| teamId | String | 团队 Id |
| createTime | Number | 团队创建时间时间戳 |
| teamName | String | 团队的名称 |
| adminName | String | 团队负责人 |
| members | Array | 项目成员 |
| username | String | 成员名 |
| avatar | String | 成员头像 |
| teamProjects | Array | 团队下属的项目，是对象数组，数组中每一个对象由 projectName 和 projectLogo 两个属性组成 |
| projectName | String | 项目名称 |
| projectId | String | 项目 Id |
| projectLogo | String | 项目 logo |
| adminName | String | 项目管理员 Id |

**备注**

* 更多返回错误代码请看首页的错误代码描述
  1. 任务管理

创建子任务

**简要描述：**

* 新增子任务
* 权限：项目管理员

**请求 URL：**

* http://xx.com/api/mesh/subtask

**请求方式：**

* POST

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| projectId | 是 | String | 当前项目 Id |
| taskId | 是 | String | 任务 Id |
| subTaskName | 是 | String | 子任务名称 |
| description | 否 | String | 子任务具体描述 |
| principal | 是 | String | 子任务负责人 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"subtask": {

"taskId": "aklgnlkjsald",

"taskName": "UI/UX设计",

"isFinished": false,

"createTime": "16546513231",

"description": "任务内容详细描述xxxxxxxx",

"founder": "刘雪迪",

"principal": "蒋金峰"

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功创建 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |
| taskId | String | 子任务 Id |
| taskName | String | 子任务名称 |
| isFinished | Boolean | 子任务是否已经被完成 |
| createTime | Number | 创建时间时间戳 |
| description | String | 子任务具体描述 |
| founder | String | 子任务创建人 |
| principal | String | 子任务负责人 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

创建任务

**简要描述：**

* 新增任务
* 权限：项目管理员

**请求 URL：**

* http://xx.com/api/mesh/task

**请求方式：**

* POST

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| projectId | 是 | String | 当前项目 Id |
| taskName | 是 | String | 任务名称 |
| priority | 否 | Number | 任务优先级，0：极低，1：普通，2：很高，3：极高 |
| deadline | 否 | String | 任务截止日期 |
| description | 否 | String | 任务具体描述 |
| principal | 是 | String | 任务负责人 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"task": {

"taskId": "aklgnlkjsald",

"taskName": "UI/UX设计",

"isFinished": false,

"priority": 1,

"createTime": "16546513231",

"deadline": "2020-06-18",

"description": "任务内容详细描述xxxxxxxx",

"founder": "刘雪迪",

"principal": "蒋金峰",

"subTasks": []

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功创建 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |
| taskId | String | 任务 Id |
| taskName | String | 任务名称 |
| isFinished | Boolean | 任务是否已经被完成 |
| priority | Number | 任务优先级，0：极低，1：普通，2：很高，3：极高 |
| createTime | Number | 创建时间时间戳 |
| deadline | String | 任务截止日期 |
| description | String | 任务具体描述 |
| founder | String | 任务创建人 |
| principal | String | 任务负责人 |
| subTasks | Array | 当前任务下所有子任务的对象数组 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

删除任务

**简要描述：**

* 删除子任务
* 权限：项目管理员及任务负责人

**请求 URL：**

* http://xx.com/api/mesh/subtask

**请求方式：**

* DELETE

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| projectId | 是 | String | 当前项目 Id |
| taskId | 是 | String | 任务 Id |
| subTaskName | 是 | String | 子任务名 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": ""

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功删除 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

删除任务

**简要描述：**

* 删除任务
* 权限：项目管理员及任务负责人

**请求 URL：**

* http://xx.com/api/mesh/task

**请求方式：**

* DELETE

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| projectId | 是 | String | 当前项目 Id |
| taskId | 是 | String | 任务 Id |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": ""

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功删除 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

查询一个项目下的任务

**简要描述：**

* 查询一个项目下的任务
* 权限：项目成员

**请求 URL：**

* http://xx.com/api/mesh/task/project

**请求方式：**

* GET

**Query Params：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| projectId | 是 | String | 项目 Id |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"tasks": [

{

"taskId": "aklgnlkjsald",

"taskName": "UI/UX设计",

"isFinished": false,

"priority": 1,

"createTime": "16546513231",

"deadline": "2020-06-18",

"description": "任务内容详细描述xxxxxxxx",

"founder": "刘雪迪",

"principal": "蒋金峰",

"subTasks": [

{

"taskId": "aklgnlkjsald",

"taskName": "UI/UX设计",

"isFinished": false,

"createTime": "16546513231",

"description": "任务内容详细描述xxxxxxxx",

"founder": "刘雪迪",

"principal": "蒋金峰"

}

]

}

]

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功查询 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |
| tasks | Array | 当前项目下的所有任务的对象数组 |
| taskId | String | 任务 Id |
| taskName | String | 任务名称 |
| isFinished | Boolean | 任务是否已经被完成 |
| status | String | 任务当前状态，如：待处理，开发中，已上线，已丢弃 |
| priority | Number | 任务优先级，0：极低，1：普通，2：很高，3：极高 |
| createTime | Number | 创建时间时间戳 |
| deadline | String | 任务截止日期 |
| description | String | 任务具体描述 |
| founder | String | 任务创建人 |
| principal | String | 任务负责人 |
| subTasks | Array | 该任务对应的子任务数组 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

查询一个团队下的任务

**简要描述：**

* 查询一个团队下的任务
* 权限：项目成员

**请求 URL：**

* http://xx.com/api/mesh/task/team

**请求方式：**

* GET

**Query Params：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| teamId | 是 | String | 项目 Id |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"tasks": [

{

"projectId": "sdgsdgsdf",

"projectName": "前端开发",

"taskId": "aklgnlkjsald",

"taskName": "UI/UX设计",

"isFinished": false,

"priority": 1,

"createTime": "16546513231",

"deadline": "2020-06-18",

"description": "任务内容详细描述xxxxxxxx",

"founder": "刘雪迪",

"principal": "蒋金峰"

}

]

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功查询 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |
| tasks | Array | 当前团队下的所有任务的对象数组 |
| projectId | String | 对应项目 Id |
| projectName | String | 对应项目名字 |
| taskId | String | 任务 Id |
| taskName | String | 任务名称 |
| isFinished | Boolean | 任务是否已经被完成 |
| priority | Number | 任务优先级，0：极低，1：普通，2：很高，3：极高 |
| createTime | Number | 创建时间时间戳 |
| deadline | String | 任务截止日期 |
| description | String | 任务具体描述 |
| founder | String | 任务创建人 |
| principal | String | 任务负责人 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

更新子任务

**简要描述：**

* 更新子任务
* 权限：项目管理员及任务负责人

**请求 URL：**

* http://xx.com/api/mesh/subtask

**请求方式：**

* PATCH

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| taskId | 是 | String | 任务 Id |
| projectId | 是 | String | 当前项目 Id |
| subTaskName | 是 | String | 子任务名 |
| isFinished | 否 | Boolean | 子任务是否已经被完成 |
| description | 否 | String | 子任务具体描述 |
| principal | 否 | String | 子任务负责人 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"task": {

"taskId": "aklgnlkjsald",

"taskName": "UI/UX设计",

"isFinished": false,

"createTime": "16546513231",

"description": "任务内容详细描述xxxxxxxx",

"founder": "刘雪迪",

"principal": "蒋金峰"

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功更新 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |
| taskId | String | 子任务 Id |
| taskName | String | 子任务名称 |
| isFinished | Boolean | 子任务是否已经被完成 |
| createTime | Number | 创建时间时间戳 |
| description | String | 子任务具体描述 |
| founder | String | 子任务创建人 |
| principal | String | 子任务负责人 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

更新任务

**简要描述：**

* 更新任务
* 权限：项目管理员及任务负责人

**请求 URL：**

* http://xx.com/api/mesh/task

**请求方式：**

* PATCH

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| projectId | 是 | String | 当前项目 Id |
| taskId | 是 | String | 任务 Id |
| isFinished | 否 | Boolean | 任务是否已经被完成 |
| priority | 否 | Number | 任务优先级，0：极低，1：普通，2：很高，3：极高 |
| deadline | 否 | String | 任务截止日期 |
| description | 否 | String | 任务具体描述 |
| principal | 否 | String | 任务负责人 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"task": {

"taskId": "aklgnlkjsald",

"taskName": "UI/UX设计",

"isFinished": false,

"priority": 1,

"createTime": "16546513231",

"deadline": "2020-06-18",

"description": "任务内容详细描述xxxxxxxx",

"founder": "刘雪迪",

"principal": "蒋金峰",

"subTasks": []

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功更新 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |
| taskId | String | 任务 Id |
| taskName | String | 任务名称 |
| isFinished | Boolean | 任务是否已经被完成 |
| priority | Number | 任务优先级，0：极低，1：普通，2：很高，3：极高 |
| createTime | Number | 创建时间时间戳 |
| deadline | String | 任务截止日期 |
| description | String | 任务具体描述 |
| founder | String | 任务创建人 |
| principal | String | 任务负责人 |
| subTasks | Array | 当前任务下所有子任务的对象数组 |

**备注**

* 更多返回错误代码请看首页的错误代码描述
  1. 统计管理

查询总体统计数据

**简要描述：**

* 查询总体统计数据
* 权限：admin

**请求 URL：**

* http://xx.com/api/mesh/stastistics/general

**请求方式：**

* GET

**Query Params：**

无

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"currentOnlineUser": "1000",

"avgTeamUser": "10.24",

"avgTeamProject": "1.63",

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功查询统计数据 |
| msg | String | 错误信息，若成功返回空 |
| currentOnlineUser | Number | 当前在线用户数 |
| avgTeamUser | Number | 团队平均成员数（浮点数） |
| avgTeamProject | Number | 团队平均项目数（浮点数） |

**备注**

* 更多返回错误代码请看首页的错误代码描述

查询用户数量

**简要描述：**

* 查询当前及历史用户量
* 权限：admin

**请求 URL：**

* http://xx.com/api/mesh/stastistics/totaluser

**请求方式：**

* GET

**Query Params：**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| timeInterval | Number | 两条历史用户量的时间间隔 |
| itemCount | Number | 返回历史用户量的条数 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"currentTotalUser": "59873",

"historyTotalUser": [

{

"totalUser": "59541"

}

]

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功查询统计数据 |
| msg | String | 错误信息，若成功返回空 |
| currentTotalUser | Number | 当前用户总量 |
| historyTotalUser | Array | 历史用户量数组 |
| totalUser | Number | 历史用户量 |

**备注**

* 更多返回错误代码请看首页的错误代码描述
* historyTotalUser 对象数组包含共 itemCount 条历史用户量数据，第 n 条（从 1 开始数）历史用户量数据是时间为 currentTime - n \* timeInterval 时的历史用户量
* 历史用户量可以用 users 的 CreatedTime 字段或者任何合适的方法计算

查询用户

**简要描述：**

* 根据关键词查询用户
* 权限：管理员

**请求 URL：**

* http://xx.com/api/mesh/stastistics/search-user

**请求方式：**

* GET

**Query Params：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| keyword | 是 | String | 关键词 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"users": [

{

"username": "王泽钜",

"avatar": "xxxxxxxx",

"nickname": "软件之王",

"gender": 1,

"status": 1,

"address": "河北省 保定市 某某区",

"description": "我是天命之灭世者万物之终结者软件之王",

"birthday": "yyyy-MM-DD",

"preference": {

"preferenceShowMode": "card",

"preferenceColor": "blue",

"preferenceLayout": "Default",

"preferenceTeam": "asdgsdfgsdf"

},

"teams": [

{

"teamName": "数据库课程团队",

"teamId": "sadgfasdhfawejkfak",

"adminName": "王泽钜",

"createTime": 1595217702138

}

]

}

]

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功加入队伍 |
| msg | String | 错误信息，若成功返回空 |
| project | Object | 查询返回的项目信息 |
| username | String | 用户名 |
| avatar | String | 用户头像 |
| nickname | String | 昵称 |
| gender | Number | 性别，0 为无性别，1 为男性，2 位女性 |
| status | Number | 状态 tag，0 为无状态，1-8 分别为摸鱼，会议，生病，休假，出差，在家，运动，游戏 |
| address | String | 地址 |
| description | String | 用户个人描述 |
| birthday | String | 生日 |
| preference | Object | 用户偏好 |
| preferenceShowMode | String | 偏好展示模式 |
| preferenceColor | String | 用户偏好主题色（默认 blue） |
| preferenceLayout | String | 用户偏好布局（默认 Default） |
| preferenceTeam | String | 用户偏好的团队 Id |
| teams | Array | 用户参与的团队 |
| teamId | String | 团队 Id |
| teamName | String | 团队名称 |
| adminName | String | 团队管理员用户名 |
| createTime | Number | 团队创建时间时间戳 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

查询用户统计数据

**简要描述：**

* 查询用户统计数据
* 权限：admin

**请求 URL：**

* http://xx.com/api/mesh/stastistics/user

**请求方式：**

* GET

**Query Params：**

无

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"maleUser": "10340",

"femaleUser": "11280",

"unknownUser": "5460",

"userAge": [

{

"age": "20",

"userCount": "165"

}

]

"userLocation": [

{

"location": "Shanghai",

"userCount": "10232"

}

]

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功查询统计数据 |
| msg | String | 错误信息，若成功返回空 |
| maleUser | Number | 男性用户总数 |
| femaleUser | Number | 女性用户总数 |
| unknownUser | Number | 未知性别用户数 |
| userAge | Array | 用户年龄对象数组（包括 0 - 120 岁或者其他合适的年龄范围） |
| age | Number | 年龄 |
| userCount | Number | 特定年龄的用户数 |
| userLocation | Array | 用户位置对象数组（包括中国各省级行政区） |
| location | Number | 位置（精确到省级行政区） |
| userCount | Number | 特定省级行政区的用户数 |

**备注**

* 更多返回错误代码请看首页的错误代码描述
  1. 项目管理

创建项目

**简要描述：**

* 团队管理员创建项目
* 权限：团队管理员

**请求 URL：**

* http://xx.com/api/mesh/project

**请求方式：**

* POST

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| teamId | 是 | String | 团队 Id |
| projectName | 是 | String | 项目名称 |
| adminName | 是 | String | 项目管理员 |
| isPublic | 否 | Boolean | 是否公开给团队所有人,默认 true |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"project": {

"projectId": "asdfhgdsfxbsd",

"createTime": "1595217702138",

"projectName": "后端开发",

"projectLogo": "xxxxxxx",

"adminName": "王泽钜",

"isPublic": true,

"members": [

{

"username": "王泽钜",

"avatar": "xxxxxxxxxx"

}

]

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功创建该项目 |
| msg | String | 返回错误信息，若成功则返回空 |
| projectId | String | 项目 Id |
| createTime | Number | 项目创建时间时间戳 |
| projectName | String | 项目的名称 |
| projectLogo | String | 项目的 Logo |
| adminName | String | 项目管理员 |
| isPublic | Boolean | 是否公开给团队所有人,默认 true |
| members | Array | 项目成员 |
| username | String | 成员名 |
| avatar | String | 成员头像 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

删除项目

**简要描述：**

* 项目管理员删除项目
* 权限：项目管理员

**请求 URL：**

* http://xx.com/api/mesh/project

**请求方式：**

* DELETE

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| teamId | 是 | String | 团队 Id |
| projectId | 是 | String | 项目 Id |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": ""

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功删除该项目 |
| msg | String | 返回错误信息，若成功则返回空 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

邀请新成员

**简要描述：**

* 项目管理员邀请新成员加入该项目(该成员已经存在于该团队了)
* 权限：项目管理员
* http://xx.com/api/mesh/project/invite

**请求方式：**

* POST

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| teamId | 是 | String | 团队 Id |
| projectId | 是 | String | 项目 Id |
| inviteName | 是 | String | 受邀人用户名 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": ""

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功发出邀请 |
| msg | String | 返回错误信息，若成功则返回空 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

用户加入项目

**简要描述：**

* 当前用户加入某个项目(必须是被项目管理员邀请过的且属于同一个团队)
* 权限：团队成员
* http://xx.com/api/mesh/project/join

**请求方式：**

* POST

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| teamId | 是 | String | 团队 Id |
| projectId | 是 | String | 团队 Id |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"project": {

"projectId": "asdfhgdsfxbsd",

"createTime": "1595217702138",

"projectName": "后端开发",

"projectLogo": "xxxxxxx",

"adminName": "王泽钜",

"members": [

{ "username": "王泽钜", "avatar": "xxxxxxx" },

{ "username": "刘学迪", "avatar": "xxxxxxxx" },

{ "username": "王新宇", "avatar": "xxxxxxxxxxx" }

]

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功加入队伍 |
| msg | String | 错误信息，若成功返回空 |
| project | Object | 加入的项目 |
| projectId | String | 项目的 ID |
| createTime | Number | 项目创建时间时间戳 |
| projectName | String | 项目的名称 |
| projectLogo | String | 项目的 Logo |
| adminName | String | 项目管理员 |
| members | Array | 项目成员 |
| username | String | 成员名 |
| avatar | String | 成员头像 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

查询用户项目

**简要描述：**

* 查询该团队下当前用户可见的项目
* 权限：团队成员

**请求 URL：**

* http://xx.com/api/mesh/project

**请求方式：**

* GET

**Query Params：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| projectId | 是 | String | 项目 Id |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"project": {

"projectId": "asdfhgdsfxbsd",

"createTime": "1595217702138",

"projectName": "后端开发",

"projectLogo": "xxxxxxx",

"adminName": "王泽钜",

"isPublic": true,

"members": [

{ "username": "王泽钜", "avatar": "xxxxxxx" },

{ "username": "刘学迪", "avatar": "xxxxxxxx" },

{ "username": "王新宇", "avatar": "xxxxxxxxxxx" }

]

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功加入队伍 |
| msg | String | 错误信息，若成功返回空 |
| project | Object | 查询返回的项目信息 |
| projectId | String | 项目的 ID |
| createTime | Number | 项目创建时间时间戳 |
| projectName | String | 项目的名称 |
| projectLogo | String | 项目的 Logo |
| adminName | String | 项目管理员 |
| isPublic | Boolean | 是否公开给团队所有人,默认 true |
| members | Array | 项目成员 |
| username | String | 成员名 |
| avatar | String | 成员头像 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

更新项目

**简要描述：**

* 项目管理员更新项目设置
* 权限：项目管理员

**请求 URL：**

* http://xx.com/api/mesh/project

**请求方式：**

* PATCH

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| teamId | 是 | String | 团队 Id |
| projectId | 是 | String | 项目 Id |
| isPublic | 否 | Boolean | 是否公开给团队所有人,默认 true |
| projectName | 否 | String | 项目名称 |
| projectLogo | 否 | String | Logo，base64 编码 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"project": {

"projectId": "asdfhgdsfxbsd",

"createTime": "1595217702138",

"projectName": "后端开发",

"projectLogo": "xxxxxxx",

"adminName": "王泽钜",

"isPublic": true,

"members": [

{ "username": "王泽钜", "avatar": "xxxxxxx" },

{ "username": "刘学迪", "avatar": "xxxxxxxx" },

{ "username": "王新宇", "avatar": "xxxxxxxxxxx" }

]

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功更新 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |
| projectId | String | 项目 Id |
| createTime | Number | 项目创建时间时间戳 |
| projectName | String | 项目的名称 |
| projectLogo | String | 项目的 Logo url |
| adminName | String | 项目管理员 |
| isPublic | Boolean | 是否公开给团队所有人,默认 true |
| members | Array | 项目成员 |
| username | String | 成员名 |
| avatar | String | 成员头像 |

**备注**

* 更多返回错误代码请看首页的错误代码描述
  1. 通知管理

删除通知

**简要描述：**

* 用户删除当前一条通知
* 权限：通知指向人员

**请求 URL：**

* http://xx.com/api/mesh/notification

**请求方式：**

* DELETE

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| notificationId | 是 | String | 通知 Id |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": ""

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功删除 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

完成通知

**简要描述：**

* 用户将一条消息提醒标识为已完成
* 权限：通知指向人员

**请求 URL：**

* http://xx.com/api/mesh/notification

**请求方式：**

* PATCH

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| notificationId | 是 | String | 通知 Id |
| isFinished | 是 | Boolean | 消息提醒是否标志为已完成 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"notification": {

"teamId": "asdfawdhkfjlasdj",

"projectId": "asdgsdggsd",

"notificationId": "asdgsedrgsrthsdf",

"title": "刘雪迪邀请您加入数据库课程团队",

"description": "",

"createTime": "1595215568570",

"isFinished": true

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功更新 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |
| teamId | String | 所属团队 Id |
| projectId | String | 所属项目 Id |
| notificationId | String | 提醒 Id |
| title | String | 提醒标题 |
| description | String | 提醒详细内容 |
| createTime | Number | 事件产生时间时间戳 |
| isFinished | Boolean | 此条消息是否标志为已读 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

请求通知

**简要描述：**

* 用户请求当前通知
* 全部通知按照是否完成和时间顺序排序
* 权限：所有已登录人员

**请求 URL：**

* http://xx.com/api/mesh/notification

**请求方式：**

* GET

**Query Params:**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"notifications": [

{

"teamId": "asdfawdhkfjlasdj",

"projectId": "asdgsdggsd",

"notificationId": "sadgsdgfsdsafsd",

"title": "刘雪迪邀请您加入数据库课程团队",

"description": "",

"createTime": "1595215568570",

"isFinished": false

}

]

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功查询 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |
| notifications | Array | 该用户的全部消息提醒 对象数组 |
| teamId | String | 所属团队 Id |
| projectId | String | 所属项目 Id |
| notificationId | String | 提醒 Id |
| title | String | 提醒标题 |
| description | String | 提醒详细内容 |
| createTime | Number | 事件产生时间时间戳 |
| isFinished | Boolean | 此条消息是否标志为已读 |

**备注**

* 更多返回错误代码请看首页的错误代码描述
  1. 公告管理

新增公告

**简要描述：**

* 由项目管理员新增当前项目下公告
* 权限：项目管理员

**请求 URL：**

* http://xx.com/api/mesh/bulletin

**请求方式：**

* POST

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| projectId | 是 | String | 当前项目 Id |
| bulletinName | 是 | String | 公告名称 |
| description | 否 | String | 公告具体描述 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"bulletin": {

"bulletinId": "asgdsdfshjaewligkfnsedgfnasdlkfkansd",

"bulletinName": "请按时完成第一次会议决定的各自任务",

"description": "",

"createTime": "1595210136999"

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功创建 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |
| bulletinId | String | 创建 bulletin 的 Id |
| createTime | Number | 公告创建时间时间戳 |
| bulletinName | String | 公告标题 |
| description | String | 公告详细内容 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

删除公告

**简要描述：**

* 由项目管理员删除当前项目下公告
* 权限：项目管理员

**请求 URL：**

* http://xx.com/api/mesh/bulletin

**Data:**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| bulletinId | 是 | String | 要删除的 bulletinId |
| username | 是 | String | 用户名 |
| projectId | 是 | String | 当前项目 Id |

**请求方式：**

* DELETE

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": ""

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功删除 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

查看公告

**简要描述：**

* 查看当前项目下的所有公告(按时间排序)
* 权限：项目成员

**请求 URL：**

* http://xx.com/api/mesh/bulletin

**请求方式：**

* GET

**Query Params：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| username | 是 | String | 用户名 |
| projectId | 是 | String | 当前项目 Id |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"bulletins": [

{

"bulletinId": "asgdsdfshjaewligkfnalsofkansd",

"bulletinName": "第一次讨论时间已经发布到日程了注意查看",

"description": "请大家提前注册并浏览一下tower网站,会议使用Zoom软件",

"createTime": "1595210136967"

},

{

"bulletinId": "asgdsdfshjaewligkfnsedgfnasdlkfkansd",

"bulletinName": "请按时完成第一次会议决定的各自任务",

"description": "",

"createTime": "1595210136999"

}

]

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功查询 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |
| bulletins | Array | 该项目下的公告对象数组（按发布时间排序） |
| bulletinName | String | 公告标题 |
| description | String | 公告详细内容 |
| createTime | Number | 公告创建时间时间戳 |
| bulletinId | String | ID |

**备注**

* 更多返回错误代码请看首页的错误代码描述

更新公告

**简要描述：**

* 由项目管理员更新当前项目下公告
* 权限：项目管理员

**请求 URL：**

* http://xx.com/api/mesh/bulletin

**请求方式：**

* PATCH

**Data：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **必选** | **类型** | **说明** |
| bulletinId | 是 | String | 公告 Id |
| username | 是 | String | 用户名 |
| projectId | 是 | String | 当前项目 Id |
| bulletinName | 否 | String | 公告名称 |
| description | 否 | String | 公告具体描述 |

**返回示例**

{

"error\_code": 0,

"data": {

"isSuccess": true,

"msg": "",

"bulletin": {

"bulletinId": "asgdsdfshjaewligkfnsedgfnasdlkfkansd",

"bulletinName": "请按时完成第一次会议决定的各自任务",

"description": "",

"createTime": "1595210136999"

}

}

}

**返回参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| isSuccess | Boolean | 是否成功更新 |
| msg | String | 返回错误信息，若成功则返回空，否则之后所有项均为 null 或不存在该项 |
| bulletinId | String | 创建 bulletin 的 Id |
| createTime | Number | 公告创建时间时间戳 |
| bulletinName | String | 公告标题 |
| description | String | 公告详细内容 |

**备注**

* 更多返回错误代码请看首页的错误代码描述

附录A：图表索引

[图1-1 项目架构图 2](#_Toc51442625)

[图1-2 用户注册流程 3](#_Toc51442626)

[图1-3 用户登录流程 4](#_Toc51442627)

[图1-4 用户信息增删、修改流程 5](#_Toc51442628)

[图1-5 创建团队流程 7](#_Toc51442629)

[图1-6 邀请团队成员流程 7](#_Toc51442630)

[图1-7 查看团队成员流程 8](#_Toc51442631)

[图1-8 创建项目流程 10](#_Toc51442632)

[图1-9 修改项目属性流程 11](#_Toc51442633)

[图1-9 管理项目成员流程 12](#_Toc51442634)

[图1-10 查看项目详情流程 13](#_Toc51442635)

[图1-11 向项目知识库上传知识流程 14](#_Toc51442636)

[图1-12 查看项目知识库内容流程 15](#_Toc51442637)

[图1-13 删除项目知识库内容流程 16](#_Toc51442638)

[图1-14 修改项目知识库内容流程 17](#_Toc51442639)

[图1-15 创建任务流程 18](#_Toc51442640)

[图1-16 修改任务详情流程 19](#_Toc51442641)

[图1-17 添加子任务流程 20](#_Toc51442642)

[图1-18 删除任务流程 21](#_Toc51442643)

[图1-19 完成任务流程 22](#_Toc51442644)

[图1-20 查看任务详情流程 23](#_Toc51442645)

[图1-21 接收消息流程 24](#_Toc51442646)

[图1-22 查看消息提醒流程 26](#_Toc51442647)

[图1-23 查看消息详情流程 27](#_Toc51442648)

[图1-24 发布项目公告流程 28](#_Toc51442649)

[图1-25 删除项目公告流程 29](#_Toc51442650)

[图1-26 查看公告详情流程 30](#_Toc51442651)

[图1-27 使用计算器流程 31](#_Toc51442652)

[图1-28 使用图标设计流程 32](#_Toc51442653)

[图1-26 使用文本记事流程 33](#_Toc51442654)

[图1-30 进行流程图绘制流程 34](#_Toc51442655)

[图1-31 书写复杂排版图样流程 35](#_Toc51442656)

[图2-1 用户登录系统 36](#_Toc51442657)

[图2-2 用户注册系统 38](#_Toc51442658)

[图2-3 用户设置系统 40](#_Toc51442659)

[图2-4 图2-5 创建新团队系统 51](#_Toc51442660)

[图2-6 图2-7 团队知识库系统 57](#_Toc51442661)

[图2-8 图2-9 图2-10 项目管理系统 63](#_Toc51442662)

[图2-11 图2-12 任务管理系统 72](#_Toc51442663)

[图2-14 子任务系统 76](#_Toc51442664)

[图2-13 消息管理系统 79](#_Toc51442665)

[图2-16 公告系统实现 81](#_Toc51442666)

[图2-17 流程图 88](#_Toc51442667)

[图2-18 计算器 89](#_Toc51442668)

[图2-19 Markdown编辑器 89](#_Toc51442669)

[图2-20 图标设计 90](#_Toc51442670)

[图2-21 LaTeX排版设计 90](#_Toc51442671)

[图3-1 总体E-R图 91](#_Toc51442672)

[图3‑2 团队管理模块E-R图 92](#_Toc51442673)

[图3‑3 项目管理模块E-R图 93](#_Toc51442674)

[图3‑4 任务管理模块E-R图 95](#_Toc51442675)

[图3‑5 公告管理模块E-R图 96](#_Toc51442676)

[图3‑6 消息管理模块E-R图 97](#_Toc51442677)

[表格3‑1 用户表 98](#_Toc51442678)

[表格3‑2 任务表 98](#_Toc51442679)

[表格3‑3 子任务表 99](#_Toc51442680)

[表格3‑4 标签表 99](#_Toc51442681)

[表格3‑5 团队表 99](#_Toc51442682)

[表格3‑6 项目表 101](#_Toc51442683)

[表格3‑7 公告表 101](#_Toc51442684)

[表格3‑8 参与表 101](#_Toc51442685)

[表格3‑9 公告栏表 102](#_Toc51442686)

[表格3‑10 任务栏表 102](#_Toc51442687)

[表格3‑11 任务推送表 102](#_Toc51442688)

[表格3‑12 分配表 103](#_Toc51442689)

[表格3‑13 公告推送表 103](#_Toc51442690)

[表格3‑14 管理员表 103](#_Toc51442691)

[表格3‑15 团队知识库表 104](#_Toc51442692)

[表格3‑16 项目知识库表 104](#_Toc51442693)

[表格3‑17 团队知识库条目表 104](#_Toc51442694)

[表格3‑18 项目知识库条目表 105](#_Toc51442695)

[表格3‑19 备注 105](#_Toc51442696)

[图3-1 数据库关系图 106](#_Toc51442697)