珞做 TodoList 日程管理软件需求规格说明书

1. **引言**
   1. **编写目的**

编写本《珞做 TodoList 日程管理软件需求规格说明书》的目的是确定珞做Todolist日程管理系统的功能边界，明确客户和各开发部门对本系统功能需求，作为下一步双方实施项目和测试验收的依据，便于功能设计与划分。

本文的预期读者是软件开发人员（含开发主管、技术主管）、测试人员、SQA 人员及其他需要评审、调阅本文档的人员。

* 1. **背景**

- 本文档所对应的软件项目名称为：珞做 TodoList 日程管理软件

- 用户单位：5号小组（校园二手交易平台）

- 开发单位：12号小组（本团队）

- 用户的大致期望功能：

要求在包含普通用户、团队管理者和外部访客的多团队多身份场景中实现如下功能：

a. 个人任务规划

b. 团队任务派遣和管理

c. 团队的创建和管理

d. 任务协调

e. 访客管理

* 1. **定义**

本文档中使用的术语包括：

1. 用户(普通用户)：任何本系统的使用者都是用户，拥有基本的个人任务管理功能。任何用户均可创建或受邀加入/申请退出团队。团队内的普通用户可以提出协作任务请求。
2. 团队所有者：本系统中团队最高权限的所有者，可以授予/修改/驳回团队内其他成员在本团队的权限，可以解散团队。创建团队的用户默认为团队所有者，也可转让给其他团队成员。
3. 管理员：团队内的普通用户接受团队所有者授予的不同于普通用户的特别权限而成为管理员。特殊权限包括：邀请/移出团队普通成员、邀请/移除访客、受理用户的协作任务请求。
4. 访客：团队外的成员收到群组所有者或管理员的邀请，获得限期内查看部分团队任务列表和接受协作/拒绝协作任务请求的权利。访客成员不能主动提出协作请求。
5. 操作人：指操作任务条目的用户。
6. 参与人：任务条目涉及的协作用户。协作任务的参与人必须在任务时间段内空闲。
7. 任务计划：至少具有任务名称的数据条目，可以包括时间、参与人、提醒方式、反馈方式、执行周期、其他备注。
8. 协作任务与非协作任务：协作任务指多人共同参与，在所有参与人的时间表上都占用时间段。非协作用户为单人任务。
9. 任务的分派：订立好的任务被分派到参与人的设备上，建立提醒。
   1. **参考资料**

[1]毛新军，董威. 软件工程：从理论到实践.北京：高等教育出版社.2022.

1. **任务概述**
2. **目标**

本软件的应用目标是为用户提供一个方便管理任务和日程的工具，以帮助他们提高工作效率和组织能力，具体包括个人任务日程规划和团队的任务派遣和管理。

本软件与外部环境的关系包括：

1.技术环境: 软件需要适应不同操作系统、设备和浏览器，以及可能使用的第三方服务和API。

2.竞争环境: 在类似功能的众多ToDoList应用中，软件需要具备吸引用户的特色和竞争优势。

范围和背景约束包括：

1.时间和资源约束: 确定开发周期和可用资源，以保证在限定的时间内交付符合要求的软件系统。

2.技术限制: 系统开发可能受到技术框架、语言选择、第三方库等技术方面的限制。

3.法律和合规性要求: 系统设计和实施需要符合法律、隐私和安全方面的要求，如数据保护法规、用户隐私保护等。

1. **用户的特点**

在个人和团队日益复杂的多任务、多角色场景下有效规划、分配和管理任务。本软件的用户一般较为自律，对日常生活、工作的规划有较高的要求。

1. **假定和前提**

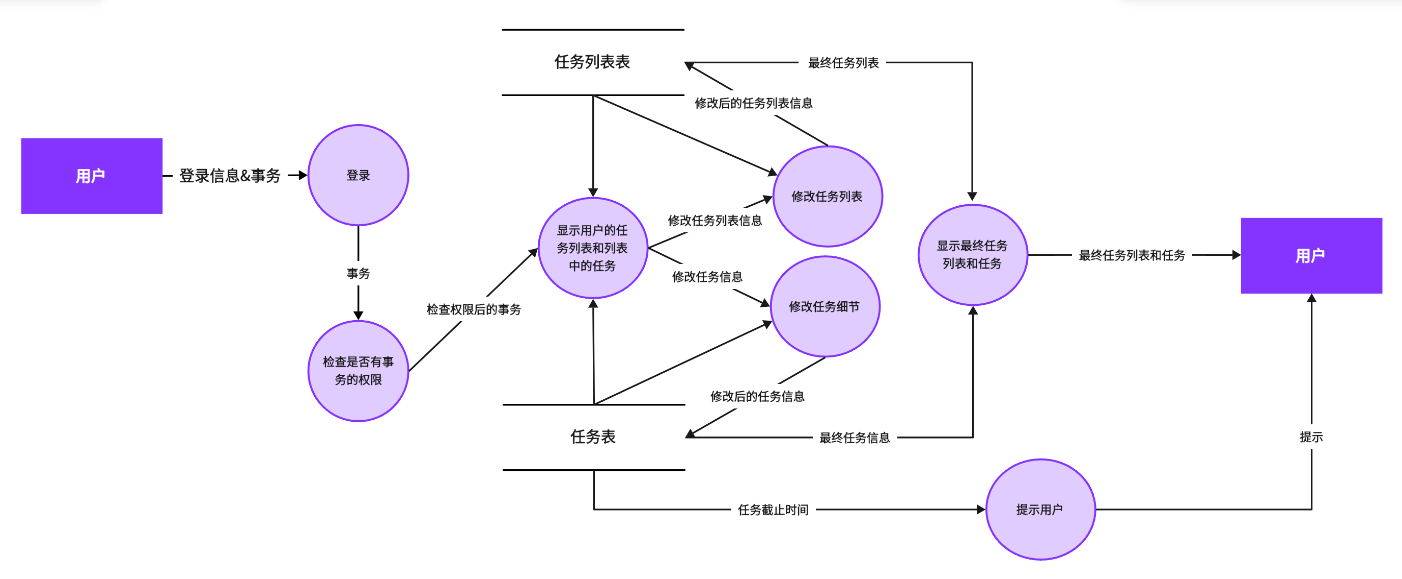
在开发本软件时,我们假设桌面端使用者熟悉基本的桌面端操作,管理员能够熟练的进行后台调试维护。

1. **需求规定** 
   1. **功能性需求**

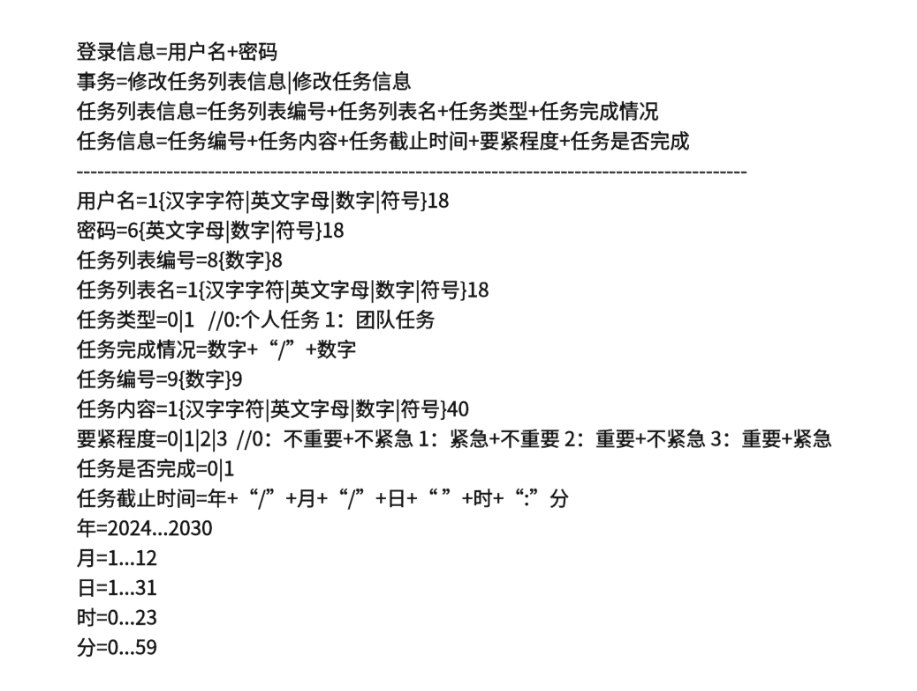
给出不同视角的分析模型。

3.1.1 结构化分析模型

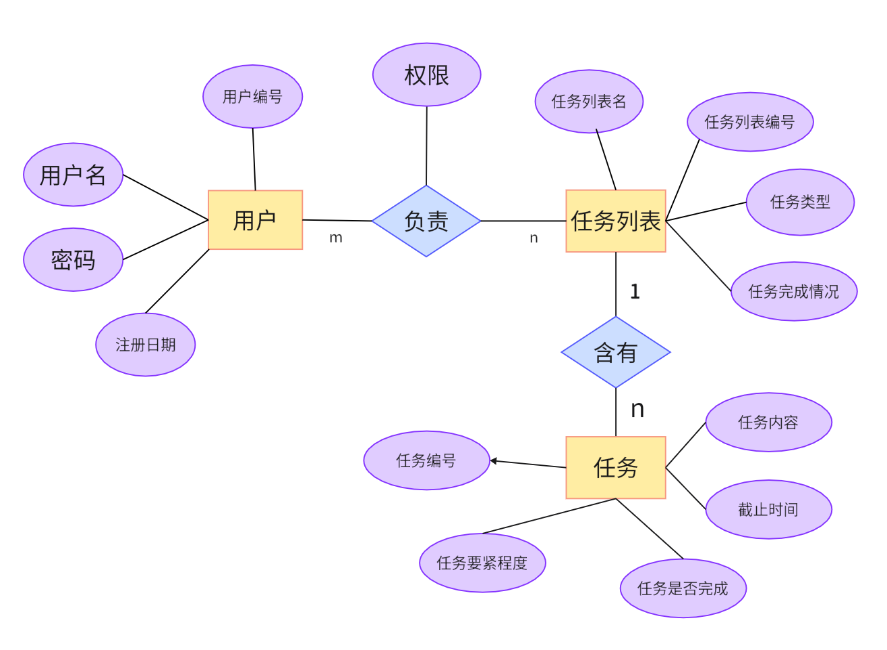
（1）数据流图



（2）数据字典



（3）E-R图

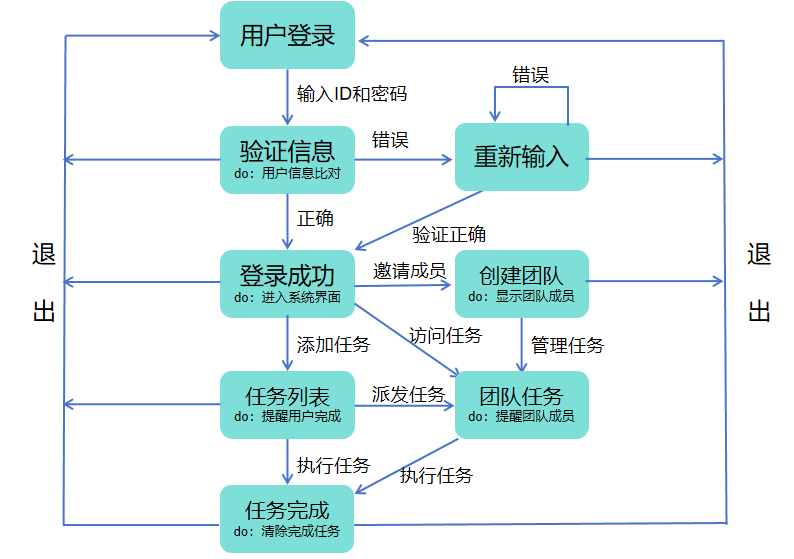


用户(User)：存储用户信息，每个用户拥有唯一的用户编号（ID）作为主键。包括用户名、密码和注册日期等属性。

任务（Task）：存储用户的任务信息，每个任务有唯一的任务编号（ID）作为主键。包括任务内容、截止时间、任务要紧程度、任务完成状态等属性。

任务列表（Task-List）:任务列表即一套任务的集合，存储任务列表中的任务。每个任务列表有唯一的任务列表编号（ID）作为主键。包括任务列表名、任务完成情况、任务类型等属性。

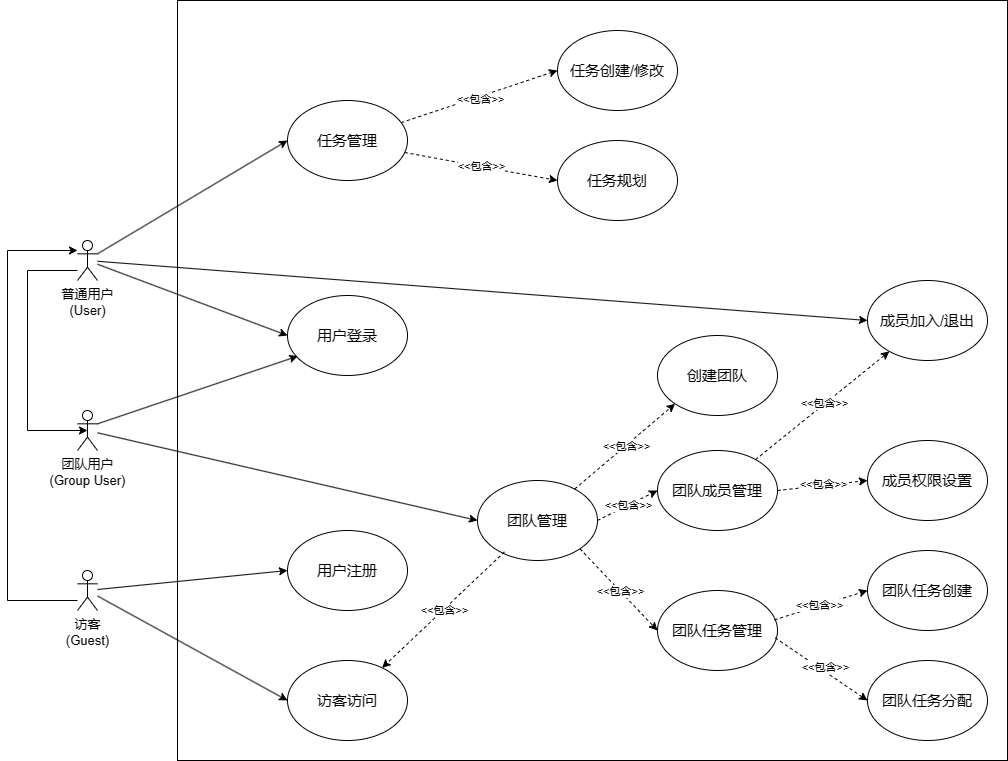
（4）状态转换图



用户登录状态下输入ID和密码，然后进入验证用户信息状态，用户登录信息错误则重新输入，验证正确则进入登录成功状态，登录成功后，用户可以添加个人任务，形成任务列表，用户可以邀请成员创建团队，作为团队管理者可以添加和派发团队任务，用户作为访客可以受邀访问查看团队任务，执行任务完毕后可以进入任务完成状态，清除任务列表中的任务，任何状态下退出后进入用户登录状态，呈现用户登录界面。

3.1.2 面向对象分析模型

（1）用例模型

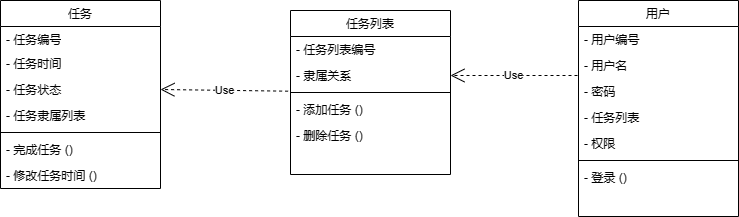


用户部分：所有用户刚进入时均为访客身份。访客经过注册和登录后变为普通用户。如果普通用户加入或者创建团队时则同时拥有团队用户身份。访客可以访问特定团队的公开部分内容。

任务部分：普通用户可以创建任务并进行任务规划。

团队管理：普通用户可以进行团队创建。团队所有者或管理员可以进行团队成员管理，包括邀请成员加入或踢出成员，并设置团队成员的访问权限。同时，团队创建者和管理员可以创建团队任务，并分配给团队成员任务。

（2）类图

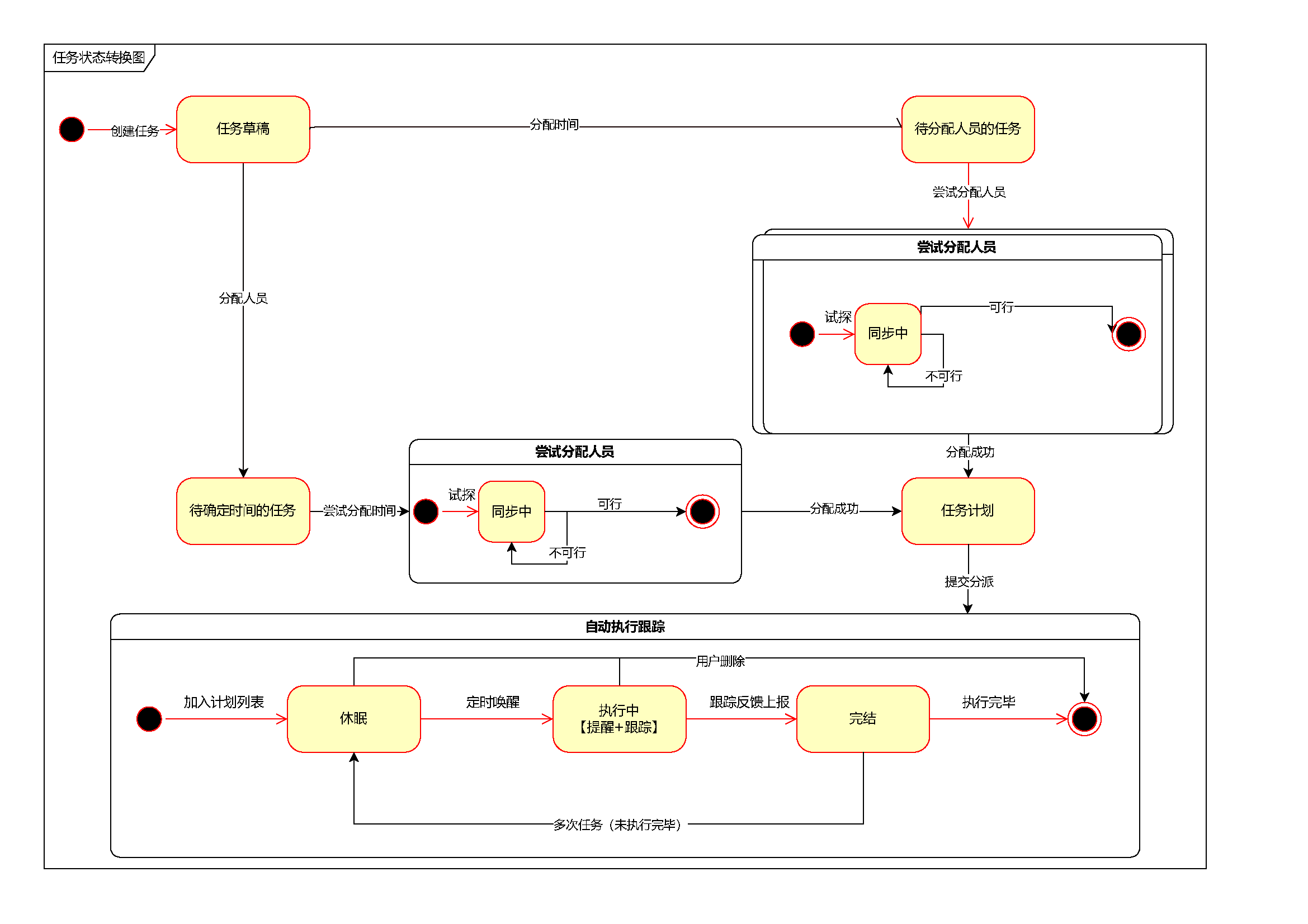


用户类：包括存储该用户的各种基本信息（用户编号、用户名、密码、任务列表、权限），以及登录等成员函数（其他待定），其中包含一个任务列表对象，记录了该用户的所有任务。

任务列表类：一个任务列表，包含一系列任务。当用户被团队分配任务或自主创建任务时，任务应加入任务列表。

任务类：包括每个任务的基本信息（任务编号、时间、完成状态等）。

（3）任务状态转换图：



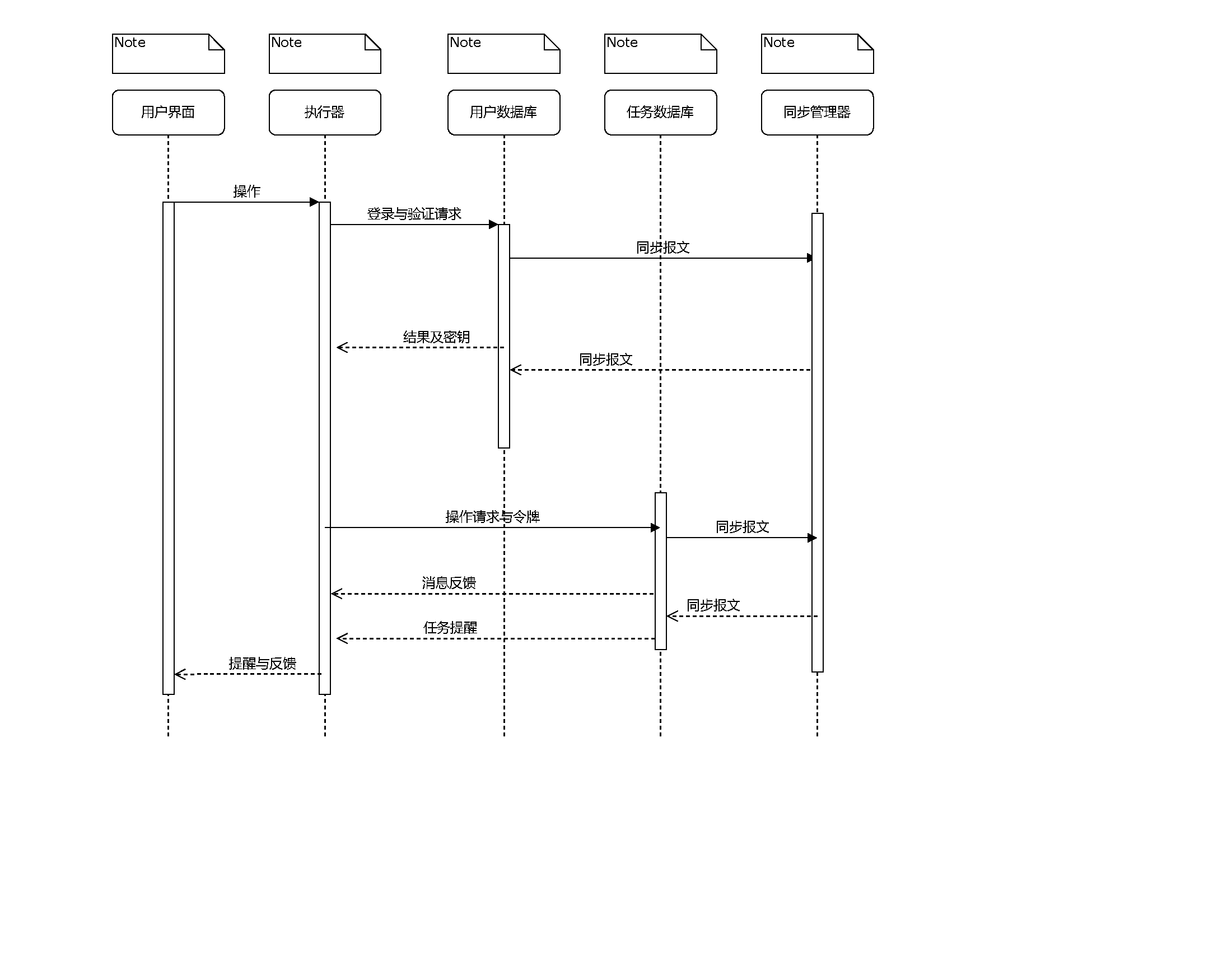
任务状态转换图说明：

1. 任务发起者发起创建任务请求，创建新的任务草稿，并在任务草稿中定义任务主题、附注等内容。

2. 任务发起者或管理者（下称操作人）为任务草稿分配人员或分配时间：（a）2.a1操作人首先分配时间，任务状态变为“待分配人员”。 2.a2 操作人向“待分配人员”的任务中尝试分配人员，系统逐一试探参与人员的空闲时间和操作人的权限，若全部可行则报错提示，直到所有人员都分配成功，得到待提交的完整任务计划。（b）2.b2操作人首先添加参与人，任务状态变为“待确定时间”。 2.b2 操作人尝试确当“待确定时间”任务的计划时间，系统逐一试探参与人员的空闲时间和操作人的权限，若全部可行则报错提示，直到所有人员都分配成功，得到待提交的完整任务计划。

3. 操作人配置提醒和反馈选项，提交任务。4. 提交的任务被加入到休眠任务列表中，并在指定时间被唤醒执行、跟踪上报。当最后一次执行完毕或被有权限的用户主动删除时，任务从执行跟踪机制中移除。

（4）产品结构顺序图：



结构顺序图说明：

1. 用户交互：外部用户只与用户界面直接交互。 用户界面将用户操作转换为执行指令，操作执行器；并根据执行器的提醒与反馈显示用户友好的提示和界面。

2. 用户管理机制：操作器向用户数据库发出登录与验证请求，用户数据库验证权限、反馈结果并向授权用户发回密钥。团队所有者可以修改用户数据库中的用户权限。

3. 任务管理机制：执行器向任务数据库发出读/写/查询操作请求和令牌，任务数据库验证令牌权限、完成操作并反馈操作结构消息。此外，任务数据库维护任务提醒表，在对应时间向执行器返回任务提醒。

4. 同步管理机制：用户数据库和任务数据库通过同步报文与同步管理器通信，同步管理器利用系统中的信道完成同步。

* 1. **非功能性需求**

1.性能优化

系统应具备高性能和稳定性，确保用户在任何时间都能够快速、流畅地使用应用程序。

应用程序应能够处理大量数据和多个同时操作，以满足用户和团队的需求，特别是在繁忙的工作时段。

响应时间应尽可能短，用户在进行任务添加、编辑、查看等操作时感知到的延迟应该很小。

2.可靠性

应用程序应具备高度的可靠性和稳定性，确保用户数据不丢失，系统不会频繁崩溃或出现不可预测的错误。

在遇到异常情况时，系统应能够自动进行故障恢复，并提供恢复用户数据的机制。

3.可扩展性

系统设计应具备良好的可扩展性，能够方便地支持未来的功能扩展和新的用户需求。

应用程序架构应设计灵活，允许添加新的模块或功能，而不会影响现有功能的稳定性和性能。

4.用户体验优化

用户界面设计应符合现代设计原则，简洁、直观、易用。

应用程序应提供清晰的操作指引和帮助文档，以便用户快速上手并充分利用各项功能。对于不同的用户角色，应提供个性化的用户体验，以满足其特定的需求和使用习惯。

5.安全性

应用程序应采取严格的安全措施，保护用户数据的机密性和完整性。数据传输应采用安全加密协议，防止数据在传输过程中被窃取或篡改。用户身份验证应采用安全可靠的机制，防止未经授权的访问。应用程序应定期进行安全审计和漏洞扫描，及时修复潜在的安全漏洞。

6.数据备份与恢复

系统应定期对用户数据进行备份，确保在意外情况下能够快速恢复用户数据。用户也应具备手动备份和恢复数据的功能，以便在需要时进行数据恢复操作。

* 1. **设计约束**

描述用户对目标软件系统的设计约束，如：运行环境/安全性/可靠性要求等等环境约束：

软件环境

|  |  |  |
| --- | --- | --- |
|  | **名称** | **版本** |
| 服务器端操作系统 | Ubuntu | Ubuntu Server 22.04.4 LTS |
| 数据库平台 | SQLite | 3.x |
| 应用平台 | 跨平台(Web/Windows/Android,其中Windows和Android通过ReactWebApp支持) | Android >=Android Q(Android 10)/Windows 7以上; |
| 客户端软件 | Chrome/Android/Windows | Chrome >=83 |

硬件环境

|  |  |
| --- | --- |
| **服务器** | **推荐配置** |
| 应用服务器（前端） | 4核8GB内存，100GB硬盘，带宽25Mbps以上（推荐的最低硬件配置） |
| 应用服务器（后端） | 4核8GB内存，100GB硬盘，带宽25Mbps以上（推荐的最低硬件配置） |
| 数据库服务器 | 4核8GB内存，100GB硬盘，带宽25Mbps以上（推荐的最低硬件配置） |

备注：应用服务器（前端）、应用服务器（后端）、数据库服务器可以是同一服务器，也可以是不同的物理服务器/容器。上述硬件环境是推荐的最低硬件配置，本系统保证能够在最低硬件配置上正常运行。

* 1. **数据要求**

说明数据的逻辑描述、输入输出要求以及能力等要求。

* 1. **其他要求**

列出其他可能的专门要求，例如：界面要求（提供原型和必要的文字说明）、进度要求、交付要求等。

1. 用户界面设计f

设计直观友好的用户界面，使用户能够轻松地创建、编辑和管理日程任务。

采用清晰的布局和易于理解的操作流程，提升用户体验。

2. 进度要求

课程结束前完成项目所有计划，包括文档撰写、界面设计、前后端开发等