<TimeGenie>

软件需求规约

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 25/06/2024 | <1.0> | 最初版本的软件需求规约，初步确定基本需求和性能。 | 齐佳怡，骆镒妤，赵楷越，孙恬然，朱涵 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

**1. 简介 4**

**1.1 目的 4**

**1.2 定义、首字母缩写词和缩略语 4**

**1.3 参考资料 4**

**2. 整体说明 4**

**2.1 产品总体效果 4**

**2.2 产品功能 5**

**2.3 用户特征 5**

**2.4 约束 5**

**2.5 假设与依赖关系 5**

**2.6 需求子集 5**

**3. 功能需求 6**

**3.1 <Use case 图> 6**

**3.2 <Use case规约> 6**

**3.2.1 <登录> 6**

**3.2.2 <注册> 7**

**3.2.3 <修改用户设置> 7**

**3.2.4 <借助AI助手进行日程管理> 7**

**3.2.5 <借助AI助手对大任务进行细分> 7**

**3.2.6 <查看个人日程> 7**

**3.2.7 <借助AI助手获得阶段总结> 8**

**3.2.8 <个人画像页面> 8**

**4. 非功能需求 8**

**4.1 易用性 8**

**4.2 可靠性 8**

**4.3 性能 8**

**4.4 可支持性 9**

**4.5 设计约束 9**

**5. 其它产品需求 10**

**5.1 联机用户文档和联机帮助的需求 10**

**5.2 接口需求 10**

**5.2.1 用户界面 10**

**5.2.2 硬件接口 10**

**5.2.3 软件接口 11**

**5.2.4 通信接口 11**

**5.3 适用的标准 11**

软件需求规约 (简化版)

# 简介

## 目的

本SRS确定了产品的功能，非功能需求和影响产品的其他功能需求，针对产品需求进行相关的需求满足，功能设计和特性的支持。

对于往后的迭代以及开发起到指导和约束作用。

## 定义、首字母缩写词和缩略语

Scrum: 迭代式增量软件开发过程

RUP: Rational Unified Process, 统一软件开发过程

Git: 分布式版本控制系统

Junit: 一种JAVA语言的单元测试框架

Spring Cloud：一个用于构建分布式系统的开源框架，基于Spring Boot简化了分布式系统的开发。

React: 一个用于构建用户界面的JAVASCRIPT库

React native: 一种跨平台移动应用开发框架，可使用JavaScript和React构建原生移动应用。

SRS： Software Requirements Specification

UML： Unified Model Language

Redis：内存中数据结构存储系统，优势在于其快速的读写速度和高效的内存使用

Nacos：一款开源的动态服务发现、服务配置和服务管理平台。

LangChain：用于提升大型语言模型功能的框架。

Fastapi：一个现代的Python Web框架，使构建API变得快速简单。

## 参考资料

《1\_2024年大二小学期大作业题目》 6月20日 互联网产品设计与开发课程组

《2\_2024年大二小学期大作业安排》 6月20日 互联网产品设计与开发课程组

《软件项目计划》 6月24日 赵楷越，朱涵，孙恬然，骆镒妤，齐佳怡

# 整体说明

## 产品总体效果

随着人们生活和工作压力增大，时间管理变得越来越重要。传统的日历和提醒工具已经不能满足人们的需求，因此利用人工智能技术开发出的日程管理助手应运而生。这种应用利用机器学习和智能算法帮助用户更高效地安排时间、提醒重要事件和任务，以应对繁忙的生活和工作压力。本产品可以提高用户时间管理效率，帮助用户合理规划和安排每天的活动，降低忘记任务和失误的风险，增强用户生活和工作的组织性和产出。

## 产品功能

|  |  |
| --- | --- |
| **客户利益** | **支持特性** |
| 希望节省时间精力，方便规划活动 | 根据用户输入的文字识别出其中的活动，自动对这些活动进行分类，并根据用户之前的习惯判断活动的优先级。 |
| 希望得到具体任务的提醒，以提高准确性，减少差错。 | 依据用户的行为习惯，为每项活动提供具体的提醒事项和行为指导。 |
| 希望得到较大任务的详细划分，使得规划更为清晰。 | 对于模糊抽象的任务给出具体建议和任务细分供用户选择。 |
| 希望依据个人需求和喜好对AI助手给出的建议进行最终裁定。 | AI助手负责给出合适的建议与条目，用户可在此基础上进行修改。 |
| 希望可以了解一段时间内的获得的成就以及完成的任务。 | 罗列并回顾一段时间之内所完成的任务事项，对其进行总结，从而方便用户管理追踪。 |
| 希望AI助手更加了解自己的习惯，便于生成更加合适的日程 | 用户可以在用户画像页面输入关键词等，便于AI尽快熟悉用户习惯。 |

## 用户特征

适用于生活和工作节奏较快，需要合理规划时间，提高工作效率的用户。用户希望通过软件来管理自己的日程，避免遗漏重要任务，提高时间利用率。

## 约束

（1）兼容性：适用于不同分辨率，不同尺寸的移动端

（2）产品开发基于华为云平台

（3）前端使用React native框架

（4）后端搭建使用springboot为主体、同时采用fastapi调用AI的服务

## 假设与依赖关系

假设用户会按照软件提供的建议和安排自己的日程；依赖于用户提供日程信息和活动分类。

华为云平台为产品开发提供分布式管理支持

## 需求子集

用户登录和注册功能

添加、编辑和删除日程安排的功能

可视化展示时间利用情况的功能

不同种类活动的分类功能

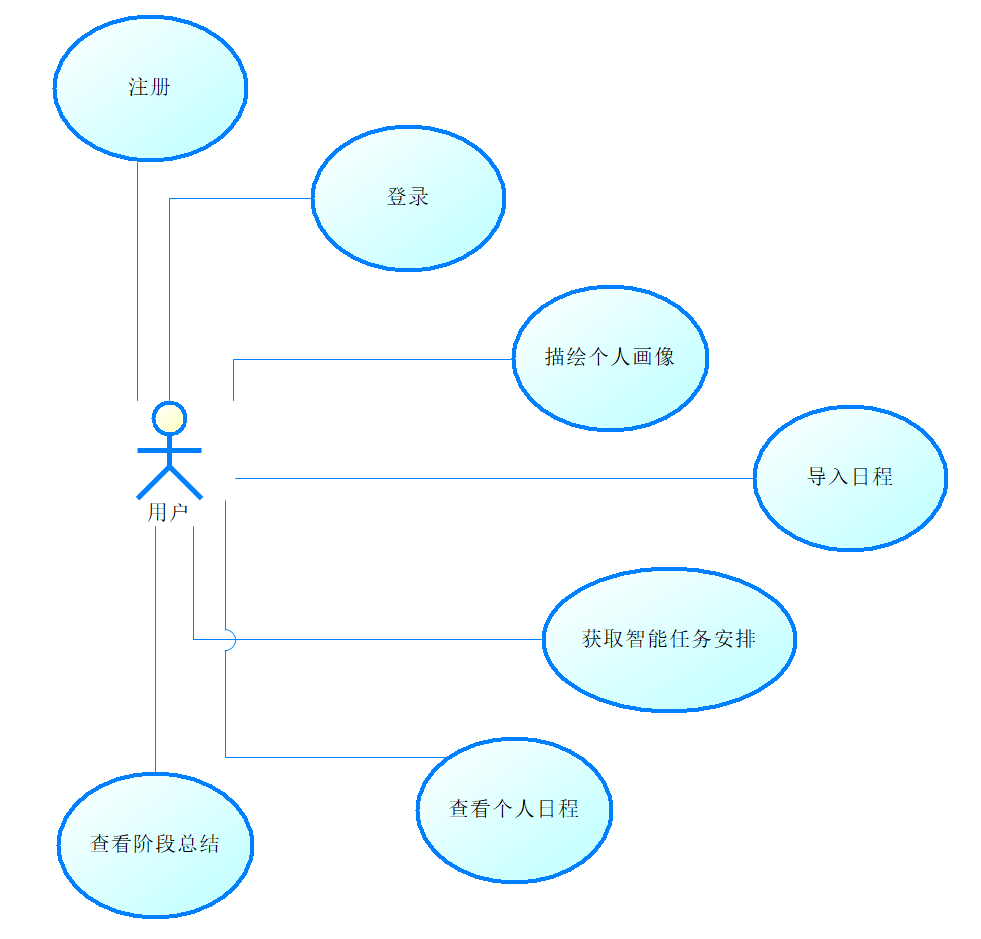
设置和接收提醒的功能

大任务细分建议的功能

总结回顾功能

# 功能需求

## <Use case 图>



Actor只包含用户。

**登录：**用户使用用户名和密码登入该软件。

**注册：**新用户第一次进入软件，需要进行注册。

**描绘个人画像：**用户输入合适的关键词，或通过一系列问答，使得AI助手更好地了解用户。

**导入日程：**用户提供活动信息，AI助手返回日程安排建议，用户最终裁定，确定日程规划。

**获取智能任务安排：**用户输入任务相关信息，AI助手提供关于任务细分、安排的建议，用户确定最终安排。

**查看个人日程：**用户可在日程显示页面查看总日程安排，点击具体日程项查看详细信息及AI助手的建议。

**查看阶段总结：**用户选取一定时间段，查看AI助手提供的阶段总结。

## <Use case规约>

### **<登录>**

基本流：

1. 用户进入软件后，在对应的输入框输入正确的用户名和密码。
2. 点击登录进入主页面。

备选流：

2a. 若用户名存在，但是密码错误，系统提示“密码错误”，并显示剩余输入密码的次数，共计十次机会。

2b. 若用户忘记密码，可点击“忘记密码”选项，通过邮箱重新设置密码。

2c. 若用户名不存在，系统提示“用户名或密码错误”。

### **<注册>**

基本流：

1. 新用户进入软件后，点击“注册”按钮进入注册页面。
2. 新用户在对应输入框分别输入符合要求的用户名以及邮箱。
3. 新用户设置密码，两次输入相同方可生效。

备选流：

使用第三方账号登录注册（可选）。

2a. 若输入的邮箱地址不符合规范，系统提示“邮箱地址无效”。

3a. 若两次输入的密码不相同，系统提醒“输入密码不一致”。

### **<修改用户设置>**

基本流：

1. 用户通过软件主页进入个人资料页面。
2. 用户在个人资料页面相应位置进行点击，可修改头像、昵称以及邮箱。

备选流：

2a. 若输入的邮箱地址不符合规范，系统提示“邮箱地址无效”。

### **<借助AI助手进行日程管理>**

基本流：

1. 用户进入日程输入界面。
2. 用户在该界面导入日程（形式包括但不限于文字输入，AI导入，第三方日历导入等）。对于有确定时间限制的活动（例如上课等确定的时间），用户需要把具体时间提供给AI助手。
3. 用户对于AI助手返回的建议日程进行修改裁定，确定最终日程。

备选流：

1a. 用户没有向AI助手提供特定任务的确定时间，AI助手将会把这些任务分配到它认为合适的时间。

2a. 用户认为AI助手所提供的日程完全符合预期，则无需进行修改，直接点击“确定”。

### **<借助AI助手对大任务进行细分>**

基本流：

1. 用户进入日程输入页面。
2. 用户在输入框向AI助手输入任务要求、完成任务的要求时间以及其他额外要求。
3. 用户对于AI助手返回的建议任务细分进行修改裁定，确定最终安排。

备选流：

1a. 用户没有向AI助手提供任务的明确时间，AI助手将会把任务分配到它认为合适的时间。

2a. 用户认为AI助手所提供的安排完全符合预期，则无需进行修改，直接点击“确定”。

### **<查看个人日程>**

基本流：

1. 用户点击进入日程显示页面查看日程。
2. 用户点击具体日程项进入日程详情页面，查看详细信息以及AI助手的建议。

### **<借助AI助手获得阶段总结>**

基本流：

1. 用户从主页进入日程统计页面。
2. 用户选择期望的日期范围，点击“生成总结”，获得AI助手提供的阶段总结。
3. 用户根据阶段总结回顾自己这段时间的成果。

备选流：

2a. 用户所选择的日期非法（例如尚未开始使用AI助手），系统提示“日期错误”。

### **<个人画像页面>**

基本流：

1. 用户点击进入个人画像页面。
2. 用户输入合适的关键词用来描绘个人画像，用于AI助手更好的了解用户。

# 非功能需求

## 易用性

AI秘书是一款日程管理工具，适合于普通人群使用，要求普通用户无需培训也可以掌握大部分功能的使用，其他功能也可以通过阅读help文档很快掌握。

风格上应与各种主流型号手机自带的日历功能类似，界面简洁，使用友好，方便用户上手使用。其设计风格和页面布局都有值得学习借鉴的地方。

IBM的CUA标准包括统一的菜单结构，快捷键，对话框布局以及交互设计准则等；Microsoft的GUI标准包括外观设计，交互准则，导航准则等。对话框的设计应当简洁，功能选择应当直观。

## 可靠性

可用性：总的有效运行时间 / 总运行时间，要求达到90%，要求平均每天可以使用24小时。

平均故障间隔时间应不低于500h。

平均修复时间：1小时

系统的输出精确到小数点后3位

最高错误或缺陷率：bugs/KLOC为3个

最高错误或缺陷率应低于2%

## 性能

1、响应时间

平均响应时间: 目标小于1.5秒

最长响应时间: 在高负载情况下，不超过3秒

2、吞吐量

每秒处理的事务数目标: 至少大于 1000并发

3、容量

系统可容纳的客户或事务数目标: 不少于10,000

4、降级模式

当系统负载过高时，系统将采用降级模式，保证核心功能、查阅用户日程数据等仍可使用，但可能会减少非必要功能和服务的响应速度和可用性。

5、资源利用

内存、磁盘、通信等资源的使用需求将根据具体微服务的部署需求和负载情况进行动态调整和优化。尽可能利用好每台服务器的资源。

## 可支持性

1、编码标准和命名约定

遵循小组内部制定的编码标准和命名约定，具体见github仓库。

2、类库和维护访问权

使用经过审查和批准的类库，并确保维护访问权的管理。代码通过单元测试后方可入库。

3、维护实用程序

使用华为云上的持续集成持续部署工具链、便于项目进行实时更新部署，便于维护。

## 设计约束

1、软件语言和框架

使用Spring Cloud作为微服务架构的基础，

后端搭建使用springboot为主体、同时采用fastapi调用AI的服务，

ReactNative框架搭建移动端前端，

整合LangChain服务接入fastapi服务接口，

利用Kafka消息队列优化用户请求处理。

需要注意后端框架需要处理一定数量并发的情况。

2、软件流程需求：

使用Scrum敏捷开发，采用迭代模型。每次迭代后与助教进行答辩审核。

3、开发工具

使用Docker容器进行微服务的部署和管理。

采用GithubDesktop以及华为云的CodeArts进行项目的管理，

使用IDEA进行后端应用的开发，使用Android Studio进行前端代码的编辑。

4、构架设计约束

良好的客户体验：基于用户体验进行设计，用户体验放在第一位，

安全性：能够应对外部的恶意攻击，

可靠性：处理服务器宕机或者数据库崩塌时对数据的非易失性存储，

可扩展性：项目应易于维护和迭代。

5、购买的构件：

使用多台云主机，进行后端架构的部署

# 其它产品需求

## 联机用户文档和联机帮助的需求

第一次进入软件时、跳出提示助手、指引用户如何使用核心功能、并输入自己的画像。在软件的主页面有一个帮助选项，里面对常见问题的回答可以涵盖大部分软件的操作，能帮助用户逐步熟悉软件的使用，解决大部分用户在操作上遇到的问题。

## 接口需求

### 用户界面

软件将实现基于ReactNative框架的移动端用户界面，以提供直观且易于使用的交互体验。

1. 登录/注册界面：

用于用户的登录和注册。

1. 日程显示界面：

显示每一周或每个月（可选）内的所有日程，显示每个日程的重要信息。

1. 单个日程项详情显示界面：

显示出单个日程的详情信息、每一个日程属性（可修改），以及AI生成的注意事项等。

1. 日程输入界面：

可通过手动导入、AI导入、第三方（如Google Calendar、Outlook Calendar，sjtu课程表等）日程自动导入。

1. 日程统计界面：

显示这一周（月、年）内日程情况的分布信息，重在给用户日程管理后的成就感。

1. 用户个人画像信息设置界面：

用户输入个人画像的界面、用于AI更好地生成推荐内容。

### 硬件接口

软件支持的硬件接口将根据移动设备的逻辑结构和物理地址进行设计，以确保与各种移动设备的兼容性和可靠性。主要包括：

1、ios系统的不同尺寸的移动端设备（APP）

2、安卓系统的不同尺寸的移动端设备（APP）

3、（可选）桌面端的应用程序（APP）

4、（可选）桌面端的web程序

### 软件接口

系统中的各个微服务将通过定义良好的软件接口与其他组件进行交互，确保模块化设计和解耦合。

举例如下（为暂定、将根据系统安全性及实际设计将进行调整）：

1、Redis数据库服务端口：5432

2、Mysql数据库服务端口：3306

3、Nacos客户端请求端口：9848

4、React Native前端项目端口：8081

5、各项后端微服务应用的端口

### 通信接口

1、使用Kafka消息队列实现与其他系统或设备通信的9092端口

2、华为云服务器用于ssh的22端口

3、本地局域网WiFi

## 适用的标准

1、安全标准：本产品应当严格符合中华人民共和国国家和地方的相关法律规范,使用TCP/IP通讯协议，使用Windows和Android的平台一致性标准和ISO相关质量和安全标准

2、国际化标准：确保系统支持中文和英文的用户需求。

3、运行环境兼容性标准：确保系统能至少在各种安卓和IOS移动端设备上正常运行和交互。