<TimeGenie>

软件架构文档

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| <01/07/2024> | <1.0> | 初步编写项目架构文档 | 朱涵，齐佳怡，骆镒妤，赵楷越，孙恬然 |
| <02/07/2024> | <2.0> | 对架构文档进行修改，最后确认 | 朱涵，齐佳怡，骆镒妤，赵楷越，孙恬然 |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 4

1.1 目的 4

1.2 参考资料 4

2. 用例视图 4

3. 逻辑视图 4

3.1 概述 4

3.2 在构架方面具有重要意义的设计包 4

4. 进程视图 4

5. 部署视图 5

6. 实现视图 5

7. 技术视图 5

8. 数据视图 5

9. 质量属性的设计 6

软件架构文档

# 简介

## 目的

本文档将从构架方面对系统进行综合概述，其中会使用多种不同的构架视图来描述系统的各个方面。它用于记录并表述已对系统的构架方面作出的重要决策。

这份文档展示了多个架构视图，旨在帮助代码维护人员更深入、更快速地了解这个产品的整体架构和代码组成。最好在阅读本文档时结合具体的代码，以达到更好的效果。

## 参考资料

《软件需求规约》 日期2024-06-28

# 用例视图

1. 用户登陆注册
2. 用户添加修改日程
3. 用户导入日程
4. 用户获取智能日程安排
5. 用户查看日程
6. 用户获取活动注意事项智能提示
7. 用户获取阶段总结
8. 用户描绘个人画像

# 逻辑视图

## 概述

逻辑视图采用微服务的架构风格，将应用划分为客户端和服务端，其中服务端又根据不同的业务功能来划分为不同的微服务，包括：用户鉴权、语言模型处理、日程管理业务处理等

## 在构架方面具有重要意义的设计包

用户鉴权服务（Firebase Admin SDK）

Java

AI日程管理服务

Java

Python

语言模型服务（langchain）

业务逻辑服务（Spring Boot）

Java

# 进程视图

重量级进程：

1.前端本地数据库进程

2.后端服务器进程（多个 包括上述三种服务 一个微服务可能开启多个线程进行并发处理）

3.云端数据库进程

轻量级进程：

1.前端app进程

进程间通信：restful的请求响应 消息队列(未确定)

# 部署视图

如果两个进程分开需要使用nginx类似的web服务器

应用服务器（Spring Boot框架的后端 以jar包形式部署到服务器）

客户端（安卓或IOS）

产生请求的进程

登录鉴权进程

业务逻辑控制的进程

云端数据库服务器（Firebase）

数据库管理系统的进程

语言模型进程

# 实现视图

软件的实现主要包括用户管理、日程管理和AI助手三个部分。整体结构上，软件分解为实现模型中的三个层：用户界面层、业务逻辑层和数据存储层，以及对应的子系统。在构架方面具有重要意义的构件包括数据库模块、用户界面模块、日程管理模块和AI引擎模块。

# 技术视图

编程语言：前端使用TypeScript，后端使用Java，Python,

开发工具：使用Docker容器进行微服务的部署管理，Intellij IDEA进行后端开发，VSCode进行前端开发

框架：前端使用React Native，后端使用Spring Boot、Spring Cloud、Firebase的SDK，调用AI服务使用FastAPI

数据库：SQLite（本地数据库）、Firebase（云端数据库）

中间件： 负载均衡 主从复制 数据同步

# 数据视图

数据存储于数据库中，分为两部分的持久化存储。

移动端本地存储采用嵌入式数据库SQLite，云端数据库采用Firebase，当用户离线做出修改操作后，日程数据应该存放于一个专门存储未同步成功的数据表，待下次与云端成功连接后重新同步上传至云端。

云端数据库进程

用户移动端的应用进程

上传同步

云端数据库

嵌入式本地数据库

初次登录时拉取

# 质量属性的设计

1.性能：平均响应时间小于1.5s，每秒处理事务数目标至少大于1000并发。

2.可扩展性：前后端分离，架构清晰，有分层，便于后期修改和增加功能

3.可靠性：有效运行时间长，平均故障间隔时间长，平均修复时间短，系统错误少。

4.易用性：功能清晰简单，对于初次接触该软件的人也容易上手。

5.可移植性：适用于移动端不同的分辨率或者尺寸。