软件需求说明书

**1、引言**

1.1编写目的

本软件需求说明书旨在明确“社交活动规划助手”软件的开发需求，描述软件功能、性能以及接口的详细要求。此文档预期读者包括项目的任务提出者、开发团队成员、最终用户以及负责本软件实施的计算中心技术人员。

1.2背景

社交活动规划助手是一款旨在解决日常生活中社交活动规划难题的创新应用程序。随着现代生活的快节奏，人们经常发现自己难以安排时间与朋友、家人和同事共度宝贵的社交时光。现有的规划工具或社交媒体平台虽然提供了部分帮助，但尚未提供一种全面的解决方案。

市场需求分析表明，用户渴望一种易于使用的工具，能够不仅规划各种社交活动，还能将他们的亲朋好友一同纳入计划，从而加强社交联系。同时，用户也希望有一个智能的系统，可以根据他们的偏好和需求为他们提供有趣的社交活动建议，减少规划活动所需的时间和精力。

说明：

1. 系统名称：社交活动规划助手
2. 任务提出者：软件工程G18组
3. 开发者：软件工程G18组

用户：所有有社交活动需求的广大群众

实现该软件的计算中心或者计算机网络：学习机房

4、该软件同其他系统或机构的基本来往关系：由浙大城市学院计算学院来做技术支持。

1.3定义

社交活动规划：指的是组织、协调和管理各类社交聚会或事件的过程。

RSVP管理：请柬回复管理，用于跟踪邀请的确认情况。

UI：用户界面，用户与软件交互的界面。

API：应用程序编程接口，允许软件组件之间相互通信的协议。

**2、任务概述**

2.1目标：

“社交活动规划助手”致力于为社会各界人士提供一个简洁、高效的社交活动规划平台。该软件开发的主要意图是简化活动的组织、管理和参与过程，使其更加便捷和无缝。它将支持活动的创建、邀请发放、日程安排及预算监控等功能。本软件作为独立产品存在，为用户提供自包含的服务，但设计上允许与外部服务（如社交媒体平台、在线日历等）集成，以扩展其功能和覆盖范围。

2.2用户特点：

该软件面向全社会公众，用户可能包括但不限于活动规划者、组织者和参与者，他们可能具有不同的计算机使用技能和技术背景，还有软件的管理员。软件的设计将考虑到这一用户多样性，确保简单易用，同时对于那些需要更高级功能的用户提供足够的灵活性。由于预期软件将广泛用于多种类型和规模的活动规划，因此用户交互将被设计为直观且易于导航。

2.3假定和约束：

开发假定包括用户将通过互联网访问软件服务，并且用户设备能够兼容常见的操作系统和浏览器。项目的约束因素包括有限的开发预算和市场上已有的竞争产品。软件开发团队必须在有限的时间内提供一个可靠、安全且用户友好的产品，以确保在激烈的市场竞争中占有一席之地。

**3、需求规定**

3.1功能需求：

活动创建：

输入：用户提供的活动信息，如活动名称、描述、地点、时间、预算、活动类型（线上或线下）、参与者数量限制等。增加可选项，如活动图片、特定标签等，增强活动描述。

处理：验证信息的完整性和合理性。例如，确保时间和地点不冲突，预算在合理范围内。

活动创建后，自动生成活动ID，并在数据库中记录活动信息。

输出：活动详情页面，展示所有活动信息，包括创建者、时间、地点、参与者等。提供分享功能，允许用户将活动详情通过社交媒体或邮件分享给他人。

用户管理：

输入：用户注册信息，包括用户名、密码、邮箱或手机号、头像、个性签名等。验证码或其他身份验证机制，以增强账户安全。

处理：验证注册信息的有效性，如邮箱格式、密码强度。创建用户账户，分配唯一的用户ID，并在数据库中存储用户信息。

输出：用户个人中心，展示用户信息、参与的活动、收藏的活动等。提供用户信息编辑功能，如更改密码、修改个人资料。

活动邀请：

输入：选择参与者的信息，如用户名或邮箱地址。

活动详情，包括活动ID、时间、地点等。

处理：检查被邀请者是否为注册用户，确保邀请信息的准确性。生成并发送邀请，包括活动详情和参加方式。

输出：邮件或应用内通知形式的邀请确认。被邀请者可以确认参与或拒绝邀请，并将结果反馈给活动发起者。

数据同步：

输入：用户对活动的任何更新，如更改时间、地点或取消活动。用户个人信息的更新，如更改头像或个性签名。

处理：实时同步更新到服务器，确保所有数据的一致性。使用有效的同步机制，如WebSocket或轮询，保证实时性。

输出：其他用户端接收到实时更新，如活动变更通知。系统在各个终端之间保持数据一致性和最新状态。

3.2性能需求：

精度：

数据校验和验证机制：实施严格的数据校验机制以确保所有输入数据满足精度要求。例如，可以在数据输入时进行即时校验，确保用户输入的数据符合精度要求。

浮点数运算的精度管理：考虑到浮点数运算可能出现的精度问题，使用特定算法或库来确保财务计算的准确性，如在涉及货币的计算中使用特定的货币处理库。

时间特性要求：

系统优化：对软件的关键部分进行性能优化，确保响应时间和处理时间符合要求。例如，对数据库查询进行优化，使用缓存机制来减少响应时间。

负载测试：进行定期的负载测试，以确保在不同的用户负载下，系统的响应时间和处理时间仍能满足要求。

异步处理和批量操作：对于更新处理和数据转换，考虑使用异步处理或批量操作来提高效率。

灵活性：

用户量适应性：设计可扩展的架构，以适应不同规模的用户量。例如，使用微服务架构或容器技术以便于扩展。

环境适应性：确保软件能够在不同的操作系统和硬件环境中稳定运行，考虑使用跨平台技术。

接口兼容性：为软件设计灵活的接口系统，方便与其他软件或服务集成。同时，确保软件能够适应接口标准的变化。

用户设置的灵活性：提供用户界面上的设置选项，让用户可以根据需要调整操作方式和数据显示的精度。

3.3可靠性和可用性需求：

故障恢复：

具体恢复步骤：在检测到主服务器故障时，自动切换至预先配置的备份服务器。实现故障自动检测机制，当系统检测到关键性能指标低于阈值时，自动触发故障恢复流程。定期进行故障恢复演练，确保故障发生时的流程可靠有效。

恢复时间目标（RTO）和数据恢复点目标（RPO）：设定RTO为1小时内，确保系统在发生故障后最多在1小时内恢复到正常运行状态。设定RPO为5分钟，意味着系统最多丢失最近5分钟内的数据。实现定期自动备份，例如每15分钟进行一次，以减少数据丢失的风险。

用户体验：

直观易用的用户界面设计：创建清晰的导航结构，确保用户可以轻松找到所需功能。采用一致的设计语言和元素，减少用户的学习成本。提供交互式教程或引导，帮助新用户快速熟悉应用程序。

响应式设计：使用流体布局和弹性网格系统，确保界面在不同屏幕尺寸和分辨率下都能保持良好的可读性和可用性。对触控设备进行优化，例如，增加按钮和其他可交互元素的大小，以适应触控操作。

3.4出错处理需求

错误日志：

日志内容：记录错误发生的确切时间，以便于追踪和定位问题。为每种错误类型指定一个唯一的错误代码，方便快速识别问题。提供清晰、简洁的描述错误的信息。记录发生错误时关键变量或系统状态，以帮助分析原因。如果可能，记录用户在错误发生前的操作步骤。

日志管理：定义不同级别的日志（如信息、警告、错误、致命），使得日志更有组织。定期轮换日志文件，避免单个文件过大，确保日志的长期存储和易于检索。确保日志中不包含敏感信息，如用户个人数据。

日志审计和分析：定期进行日志审计，以便及时发现潜在问题或异常模式。分析错误日志，识别常见或重复的错误模式，为系统改进提供依据。使用日志分析工具自动化常规的日志审计和错误趋势分析。

用户提示：

错误提示信息：使用非技术性语言，清晰描述问题和影响。提供用户可以采取的具体操作步骤，以解决或缓解问题。根据用户的语言偏好显示错误信息。

帮助链接和FAQ：对于常见错误，提供指向帮助中心或FAQ页面的链接。考虑提供交互式故障排除向导，帮助用户解决问题。允许用户直接从错误界面提交反馈或报告问题。

3.5接口需求

外部服务集成：

选择与微信、QQ等主流社交平台集成，使用基础的OAuth认证进行用户验证，确保安全性。仅实现基本的数据交互功能，如获取用户基本信息或发布更新。

API安全：

基本安全措施：使用HTTPS加密所有API通信，确保数据传输安全。实现基本的API密钥管理，为每个用户生成唯一的访问令牌。

基础安全监测：确保对所有API输入进行验证，防止常见的网络攻击。

定期检查系统日志，留意任何异常活动。

3.6约束需求

工具和语言约束：

主要开发语言：指定SwiftUI作为主要开发语言，突出其跨平台特性和现代化的UI设计能力。

辅助工具：使用Xcode作为主要的集成开发环境，并利用其调试、版本控制等功能。

代码管理：指定使用Git进行版本控制，并使用GitHub或GitLab作为代码托管和协作平台。

设计约束：

用户界面设计：强调采用苹果的人机界面指南（Human Interface Guidelines）来确保一致的用户体验。

响应式设计：要求设计必须适应不同尺寸的iPhone和iPad屏幕，确保应用在所有设备上的可用性。

应使用的标准：

编码标准：遵守Swift编码风格和最佳实践，如使用SwiftLint来维护代码质量。

界面设计标准：遵循iOS应用界面设计的最佳实践，确保应用的直观性和易用性。

应使用的硬件平台:

iOS设备兼容性：明确应用需支持的iOS设备和操作系统版本，应用需在iOS 14及以上版本的iPhone和iPad上运行”。

性能要求：确定应用对处理器、内存等硬件的最低要求，以保证应用的流畅运行。

3.7其他专门要求：

安全保密：软件必须遵守GDPR或类似的隐私法规，实现数据加密和安全访问控制。

使用方便：软件应有多语言支持和无障碍访问功能。

可维护性：代码应有文档支持，便于维护和升级。

可靠性：应保证99.9%的在线时间。

环境可转换性：应能在主流操作系统上无缝运行。

3.8数据字典：

1. 用户（User）

用户ID (UserID): 唯一标识每个用户的数字。

用户名 (Username): 用户的昵称或登录名。

密码 (Password): 加密的登录密码。

邮箱 (Email): 用户的电子邮箱地址。

手机号 (PhoneNumber): 用户的联系电话。

头像 (ProfilePic): 用户的头像图片链接。

个性签名 (Bio): 用户的个人简介。

2. 活动（Event）

活动ID (EventID): 唯一标识每个活动的数字。

活动名称 (Name): 活动的名称。

描述 (Description): 活动的详细描述。

地点 (Location): 活动的具体地点。

时间 (DateTime): 活动的具体日期和时间。

预算 (Budget): 活动的预算。

活动类型 (Type): 活动的类型（线上或线下）。

参与者限制 (ParticipantLimit): 活动允许的最大参与者数。

3. 活动参与者（Participant）

活动ID (EventID): 参与的活动ID。

用户ID (UserID): 参与者的用户ID。

参与状态 (Status): 表示用户对活动的响应状态（接受、拒绝、未响应）。

实体关系图（ER图）

4. 管理员（Admin）

管理员ID (AdminID): 唯一标识每个管理员的数字。

用户名 (Username): 管理员的登录名。

密码 (Password): 加密的登录密码。

邮箱 (Email): 管理员的电子邮箱地址。

**4、运行环境规定**

4.1设备：

处理器型号及内存容量：推荐使用Apple M1芯片或更高版本，至少16GB RAM。

外存容量：至少512GB SSD，以支持iOS开发和测试。

输入及输出设备：MacBook或iMac，至少一台；外接显示器，可选。

数据通信设备：内置或外置Wi-Fi适配器，用于联网和测试。

功能键及其他专用础础础：Xcode和iOS模拟器，用于开发和测试应用。

4.2支持软件：

支持软件包括：

操作系统：macOS最新版本。

编译程序：Xcode最新版本，包括Swift编译器和iOS SDK。

测试支持软件：Xcode内置的测试工具和性能分析器。

4.3接口：

该软件将与以下软件接口：

数据通信协议：使用HTTP/HTTPS协议与后端服务器进行数据交换。

外部服务接口：集成社交媒体API（如微信、QQ）用于活动分享；支持与电子邮件客户端接口用于发送邀请和通知。

控制：

4.4用户界面：通过应用内的用户界面进行直接操作。

系统设置：通过iOS系统设置进行权限和通知控制。

后端管理控制：使用Web-based管理界面进行用户管理、数据分析等操作。