# ****智能家居管理系统需求文档****

## ****1. 项目背景****

随着智能技术的快速发展，智能家居系统已成为现代家庭生活的一部分。智能家居管理系统旨在提供一个集成化平台，能够让用户通过智能设备远程控制家庭中的电器、照明、安全、温控等设备。系统支持语音控制、自动化场景控制、设备管理等功能，提供便捷、安全、智能的居住体验。

## ****2. 项目目标****

该项目的目标是开发一个智能家居管理平台，支持以下功能：

**设备管理**：用户能够通过平台管理和控制家中的智能设备，如灯光、空调、电视、智能锁等。

**场景控制**：用户可以创建自定义场景，例如“离家模式”、“回家模式”等，系统会自动控制设备。

**远程控制**：用户可以在任何地方通过手机应用或网页远程控制家中的设备。

**语音控制**：通过与语音助手（如 Alexa、Google Assistant）集成，实现语音控制家居设备。

**安全监控**：系统能够接入家庭安全摄像头、传感器等设备，进行24小时监控，并在有异常时向用户发送警报。

## ****3. 功能需求****

### ****3.1 用户功能****

#### ****3.1.1 用户注册与登录****

**用户注册**

用户可通过邮箱或手机号码注册。

注册时需要验证邮箱或手机号码。

用户注册后可自定义个人资料，如昵称、头像等。

**用户登录**

用户可通过邮箱/手机和密码登录。

支持忘记密码功能。

支持第三方登录（如微信、Facebook等）。

#### ****3.1.2 设备管理****

**添加设备**

用户可以通过扫描设备二维码或手动添加设备。

支持设备类型选择和设备名称编辑。

**设备控制**

用户可通过应用控制家中设备的开关、亮度、温度等。

支持设备状态反馈，实时显示设备的当前状态。

**设备分组**

用户可以将设备按房间或功能分组，方便管理。

#### ****3.1.3 场景控制****

**创建场景**

用户可创建多种场景（例如：回家模式、离家模式、夜间模式）。

每个场景可以包含多个设备的状态配置，如开关、亮度、温度等。

**自动化控制**

用户可设定时间触发场景，如每天早上8点自动开启空调。

系统支持地理围栏功能，根据用户的位置自动切换场景（例如，用户离家时自动开启离家模式）。

#### ****3.1.4 语音控制****

**集成语音助手**

支持与主流语音助手（如 Alexa、Google Assistant）对接，允许用户通过语音控制设备。

**语音命令**

用户可以使用简单的语音命令控制设备，如“打开客厅的灯”、“设置空调为24度”。

#### ****3.1.5 安全监控****

**实时监控**

用户可以通过系统查看家庭安全摄像头的实时视频。

**报警系统**

系统通过传感器检测到异常情况时，自动向用户推送警报。

支持警报类型（如火灾、入侵）和推送方式（短信、APP通知）。

### ****3.2 管理员功能****

#### ****3.2.1 用户管理****

**用户权限管理**

管理员可查看所有用户的基本信息及设备使用记录。

管理员可以设置用户权限，如管理员、普通用户等。

**用户操作审计**

系统应记录所有用户的操作日志，包括设备控制、场景创建、设备添加等。

#### ****3.2.2 系统设置****

**设备管理**

管理员可以添加、删除、修改家庭中的智能设备。

管理员可以查看设备的运行状态和故障诊断。

**场景管理**

管理员可以查看所有用户创建的场景，并对场景进行修改或删除。

### ****3.3 非功能需求****

#### ****3.3.1 性能需求****

**响应时间**

用户请求响应时间应不超过2秒，设备控制延迟应低于1秒。

**并发用户**

系统能够支持同时1000个用户在线操作。

#### ****3.3.2 安全需求****

**数据加密**

所有用户数据、设备控制指令应加密传输，确保用户隐私。

**访问控制**

系统需实现基于角色的访问控制，确保不同权限的用户只能访问相应的资源。

#### ****3.3.3 可扩展性需求****

**设备类型支持**

系统应能够支持新增的智能设备和硬件接口。

**模块化设计**

系统应支持模块化设计，以便未来可以集成更多智能硬件和新功能。

#### ****3.3.4 可维护性需求****

**日志管理**

系统应保存系统日志和操作日志，方便问题追踪与诊断。

**错误处理**

系统应能自动检测错误并提供简明的错误信息，帮助用户进行操作修复。

### ****4. 系统架构****

#### ****4.1 架构概述****

该系统采用 **客户端-服务器架构**，客户端包括移动端应用（iOS/Android）和网页端。服务器端负责处理用户请求、设备控制、数据存储等任务，数据库用于存储用户信息、设备状态、日志等数据。

#### ****4.2 数据库设计****

**用户表（Users）**

user\_id（主键）

email（用户邮箱）

password（加密密码）

role（用户角色）

**设备表（Devices）**

device\_id（主键）

user\_id（外键，关联用户）

device\_name（设备名称）

device\_type（设备类型，如空调、灯光）

status（设备状态，开/关）

**场景表（Scenes）**

scene\_id（主键）

user\_id（外键，关联用户）

scene\_name（场景名称）

scene\_settings（场景设置，JSON格式）

**操作日志表（Logs）**

log\_id（主键）

user\_id（外键，关联用户）

operation\_type（操作类型，如设备控制、场景创建）

operation\_details（操作详情）

timestamp（操作时间）

### ****5. 接口设计****

#### ****5.1 用户注册接口****

**URL**: /api/register

**请求方法**: POST

**请求参数**:

email: 用户邮箱

password: 用户密码

**返回结果**:

status: 操作结果（成功或失败）

message: 错误信息（如果有）

#### ****5.2 设备控制接口****

**URL**: /api/device/control

**请求方法**: POST

**请求参数**:

device\_id: 设备ID

action: 控制操作（如开、关）

**返回结果**:

status: 操作结果（成功或失败）

message: 错误信息（如果有）

### ****6. 系统实现计划****

#### ****6.1 开发阶段****

**需求分析**：第1月

**系统设计**：第2月

**开发与集成**：第3-6月

**测试与优化**：第7月

#### ****6.2 部署与维护****

**部署**：第8月

**持续维护与更新**：第9月及以后