YatSpark 软件设计书

> 撰写单位：统筹管理部 撰写人：傅祉珏、庞子良 制定日期：2025-06-08 文档版本：v2.0

文档概述

项目背景

随着物联网（IoT）技术和智能家居设备的快速普及，全屋智能已逐渐从高端配置走向大众家庭生活。从智能照明、恒温控制，到语音助手与安防监控，智能设备极大提升了生活便利性与安全性。然而，现有系统在面对多用户环境时，普遍存在以下问题：

- **权限控制单一**：大多数设备仅支持主用户操作，缺乏对不同用户的权限差异控制；
- **共享管理混乱**：缺少清晰的共享规则，容易导致使用冲突或误操作；
- **使用行为不可追踪**：设备使用缺乏日志记录与审计，难以监管与追责；
- **安全隐患显著**：未经授权的访问、控制权限滥用等问题频发，影响用户隐私和财产安全。

针对以上痛点，本项目提出开发一款具备多用户管理与权限共享功能的智能家居系统——**YatSpark**（Smart Permission and Access for Resource-sharing Kit，简称**Spark**）。该系统旨在通过身份认证、权限分配、共享管理与日志审计等手段，全面提升智能家居设备在多用户场景下的安全性、可控性与协作性，助力构建更加可靠、智能与个性化的全屋环境。

项目需求

在智能家居权限管理系统的开发过程中，考虑到系统复杂度与实现可行性，建议采用场景聚焦的设计思路。当前系统开发面临的主要挑战在于需要处理过多的对象类型（如业主、访客、维修人员等）和复杂的权限逻辑（如临时权限、分级权限等）。为提升开发效率和系统实用性，建议将应用场景具体化为以下两个典型方向：

1. 酒店租户管理场景：
    - 主要面向酒店式公寓、短租民宿等商业住宿场景
    - 用户角色可简化为：前台管理员、客房服务员、住客三类

    - 权限逻辑聚焦于：入住时段控制、客房服务权限、紧急情况处理等核心需求
2. 智能宿舍管理场景：
    - 适用于高校学生公寓、企业员工宿舍等集体居住环境
    - 用户角色包括：宿管员、学生/员工、维修人员
    - 权限管理重点：门禁时段控制、访客管理、设备报修权限等

通过选择上述任一具体场景，可以有效规避智能家居全场景中的过度复杂性，同时保留权限管理系统的核心功能需求。在后续开发中，建议优先完成基础权限框架的搭建，再根据选定的具体场景进行功能模块的定制化开发，最终实现一个具有明确应用场景和完整功能的权限管理系统原型。这种聚焦式开发策略既能保证系统核心价值的实现，又能有效控制开发难度和周期。

术语标准

本术语规范用于统一“智能家居控制系统”在 UI 设计与业务逻辑设计中使用的关键术语，提升文档一致性与开发协作效率。

1. 用户角色与身份
    - **管理员**：具有系统管理权限的用户，可进行权限分配、模板配置、用户管理与日志查看等操作。
    - **普通用户**：仅可控制授权设备、查看个人权限与操作日志的用户，不能进行管理行为。
2. 页面与组件
    - **登录界面**：系统入口界面，用户通过输入 ID 和密码进行身份验证。
    - **注册界面**：新用户注册入口，填写信息后完成账号创建。
    - **工作台**：用户登录后默认进入的主界面，根据角色加载功能组件。
    - **设置界面**：用户管理个人资料与账号安全设置的页面。
    - **模板管理界面**：管理员专属界面，用于创建、编辑设备模板与默认权限。
    - **设备控制界面**：普通用户控制房间内设备的操作界面。
    - **权限编辑组件**：用于配置用户对设备的控制权限的功能模块。
    - **用户管理组件**：管理员管理房间用户信息的功能模块。
    - **房间查看组件**：查看房间结构、模板、电器分布等信息的功能模块。
    - **日志查看组件**：查看权限或设备操作记录的功能模块。
    - **开发者信息页**：展示系统开发者信息与联系方式的页面。
3. 功能术语
    - **模板（Template）**：一组电器及其默认权限的组合配置，用于快速生成标准房间配置，即房型。

- **房间（Room）**：系统中的实际空间单位，由模板实例化生成，绑定设备与用户。
- **设备（Device）**：房间中的智能控制终端，如门锁、灯光、空调等。
- **权限（Permission）**：用户对设备的操作权限，包含查看、控制、编辑等细化项。
- **个人权限（UserPermission）**：针对单一用户的个性化权限配置，可覆盖模板默认权限。
- **电器控制请求**：用户对某一设备发起的操作指令，如开关、调节亮度等。
- **操作日志（Operation Log）**：记录用户对设备操作或权限变更行为的时间、内容与主体。
- **权限日志（Permission Log）**：记录用户权限修改相关行为的系统日志。

4. 控件与交互术语
   - **导航栏**：界面顶部的统一操作区，包含头像、用户名、设置入口与登出按钮。
   - **卡片组件**：功能模块在 UI 中的表现形式，用于展示并跳转至详细功能页。
   - **图标按钮**：结合图形与文字的操作按钮，常用于代表功能模块入口。
   - **权限按钮**：用于开关型权限设置的交互控件，如开/关控制权限。
   - **禁用状态（Disabled）**：在无权限的场景下，按钮或控件不可操作，呈灰色显示。
5. 业务与模块术语
   - **电器基类**：统一定义各类电器通用字段与操作的抽象类。
   - **权限基类**：定义控制/查看等基本权限字段的抽象类。
   - **门锁权限**：包含门锁开关控制与门铃权限，下文若出现“门权限”均指代门锁权限。
   - **灯光权限**：包含开关控制与亮度调节权限。
   - **空调权限**：包含温度调节、运行模式与风速调节权限。
   - **日志模块**：负责记录系统中权限或控制行为的独立逻辑模块。
   - **模块解耦**：系统设计原则之一，强调权限判断不嵌入设备层逻辑，以保持系统扩展性。

UI界面设计

UI功能设计

本系统的 UI 界面设计遵循“简洁直观、角色分离、功能清晰”的设计原则，面向管理员与普通用户两个角色，主要功能划分为以下三大板块：

1. 登录注册板块： 该板块为系统入口，提供用户登录认证与注册功能，包括：

- **登录界面**：用户通过输入 ID 和密码登录系统，支持基础错误提示（如 ID 不存在、密码错误），页面中展示项目 Logo、"忘记密码"提示与跳转注册入口。
- **注册界面**：新用户通过填写个人信息进行注册，包括头像上传（图片裁剪）、用户名、密码设置、手机号或邮箱输入等。注册成功后自动跳转回登录界面。

2. 主界面板块：用户登录后进入工作台视图，依用户身份加载不同组件。主界面板块包括以下模块：
   - **工作台界面**：
     - 管理员工作台组件包含：权限编辑、用户管理、模板管理、日志查看、房间查看、开发者信息等；
     - 普通用户工作台组件包含：设备控制、权限查看、房间查看、日志查看、开发者信息；
     - 页面顶部设有导航栏（含头像、用户名、设置按钮、登出按钮等）；
     - 页面主体为组件区，以卡片或图标按钮方式布局，点击后进入相应功能模块。
   - **设置界面**：
     - 供用户设置头像、昵称、联系方式（如手机号、邮箱）、登录密码；
     - 管理员可访问更高级别配置选项（如模板创建权限、设备绑定授权等）。
   - **模板管理界面**（管理员专属）：
     - 模板列表页：展示当前管理员已创建模板；
     - 模板编辑页：可设置模板绑定的默认电器（空调、门锁、电灯）、预设权限项；
     - 每项设备均支持添加、删除、重命名、图标设置。
   - **设备控制界面**（普通用户专属）：
     - 当前房间设备列表页：列出房间中已有电器，展示当前状态与控制按钮；
     - 单个设备控制页：根据设备类型提供对应的操作，如温度调节（空调）、亮度调节（灯光）、远程开关（门锁）；
     - 控制按钮受限于权限配置，未授权的功能自动禁用。
3. **组件功能板块**：该板块为工作台中的可点击功能组件，提供更细粒度的用户与权限管理：
   - **权限编辑**：
     - 管理员视图：权限总览列表（可筛选用户），点击进入单用户权限详情页，支持修改不同设备的控制权限（查看、控制、编辑）；
     - 普通用户视图：展示当前权限配置，禁止编辑。
   - **用户查看/管理**：
     - 管理员视图：可查看自己管理的所有用户列表，并在用户详情页中更改其房间绑定或注销其账号；
     - 普通用户视图：仅能查看当前房间内的其他用户，禁止操作。

- **查看房间**：
  - 管理员：查看所有已创建房间及其房型信息、绑定模板、电器设备、创建时间等，支持删除房间；
  - 普通用户：查看当前所在房间的信息，仅支持浏览。
- **日志查看**：
  - 管理员可查看权限修改日志（含操作用户、目标用户、修改项、时间等）；
  - 普通用户可查看自身权限变更记录与电器使用日志（如电灯开启/关闭时间）。
- **开发者信息**：
  - 所有用户均可访问，展示开发团队成员列表、联系方式、项目说明链接等。

UI交互设计

整体交互流程力求清晰、逻辑闭环，以下为主要页面间的跳转路径及说明：

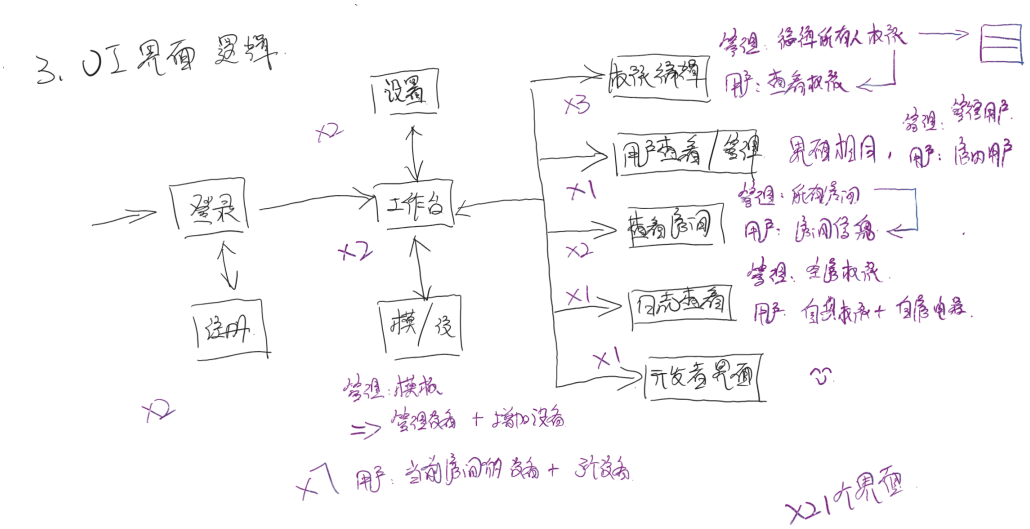


图1 UI界面逻辑示意图

> TODO: [这里应该加入UI的界面跳转图]

1. **启动阶段**：
   - 用户打开小程序或访问系统网站 → 自动进入【登录页面】；
   - 若无账号，点击“注册”跳转至【注册页面】 → 注册完成后返回【登录页面】；
   - 成功登录后跳转进入【工作台】页面。
2. **主界面交互**：
   - 工作台顶部设有【设置入口】、【退出按钮】；
   - 工作台中部加载功能组件区，依据用户角色动态呈现不同模块组件。

3. **功能组件跳转关系**：
    - **设置按钮** → 【设置界面】
    - **模板管理组件（管理员）** → 【模板列表页】 → 点击模板 → 【模板编辑页】
    - **设备控制组件（用户）** → 【设备列表页】 → 点击设备 → 【设备控制页】
    - **权限编辑组件**：
        - 管理员：→【用户权限列表】→点击某用户→【权限编辑页】
        - 用户：→【权限查看页】
    - **用户管理组件**：
        - 管理员：→【用户列表页】→点击某用户→【用户详情页】
        - 用户：→【房间用户查看页】
    - **房间查看组件**：
        - 管理员：→【所有房间列表】→点击房间→【房间信息页】
        - 用户：→【当前房间信息页】
    - **日志查看组件**：
        - 管理员：→【权限日志列表页】
        - 用户：→【权限与电器使用日志列表页】
    - **开发者组件（YatSpark 图标）** → 【开发者信息页】
4. **退出机制**：
    - 用户在任意页面可通过顶部导航栏点击“退出”按钮 → 自动跳转至【登录页面】。

该交互设计支持界面跳转图的绘制，便于明确页面流向及用户操作路径。

界面风格设计

本系统整体包含 21 个以上的 UI 界面（依据模块扩展可能更多），每个界面需根据其功能与角色定位采用统一但灵活的风格设计，以下为各界面风格的详细设计要求：

> TODO: [设计UI的同学应该在这里完成UI的设计后将设计好的UI插入到相应位置上]

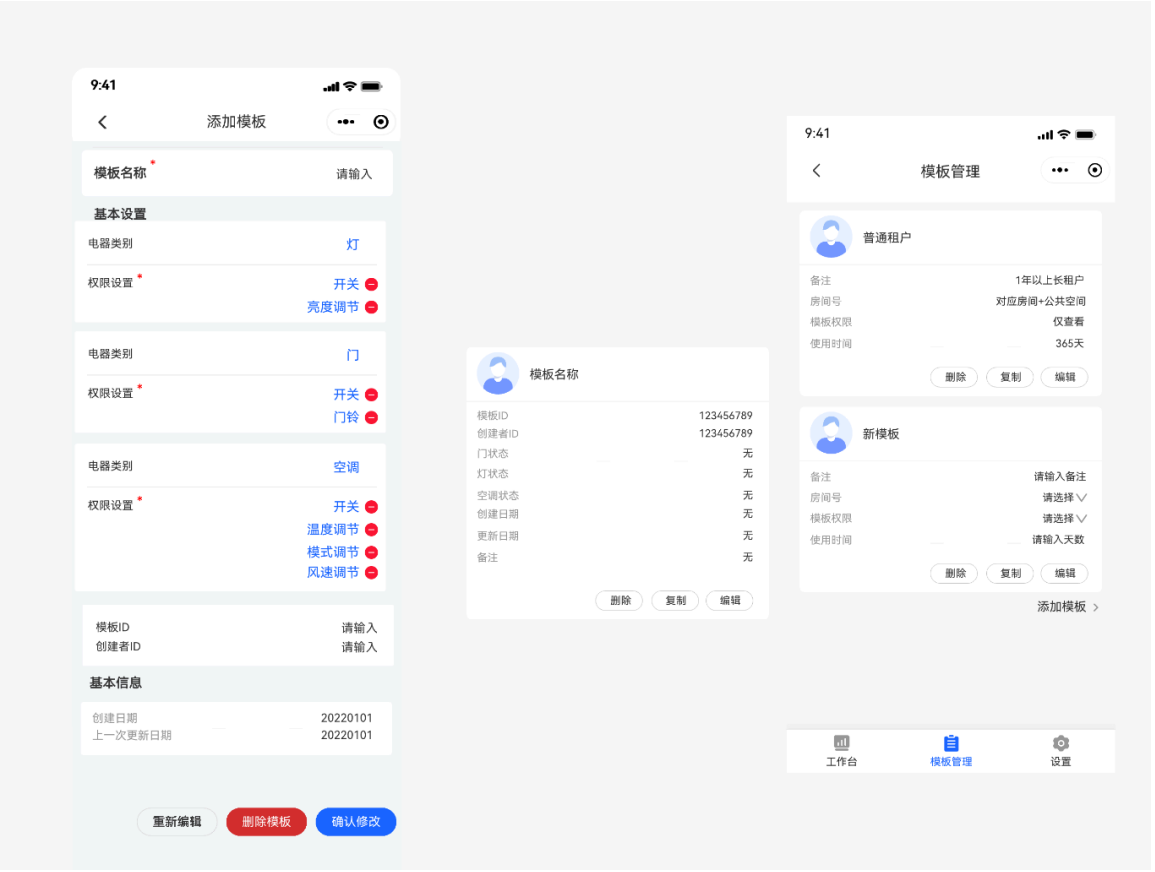
(注册登录界面)

1. 登录注册界面（2个）
    - **风格要求**：
        - 页面整体应简洁明了，背景可采用浅色渐变或柔和灰白色；
        - 页面中央居中显示项目 Logo，增强品牌识别度；
        - 表单区域应包含清晰边框，输入框圆角设计，按钮使用高对比色突出操作；
        - 注册页中应配备头像上传模块（支持图片裁剪）、表单输入区域、注册按钮与“返回登录”按钮。
    - **交互提示**：
        - 输入错误（如密码不一致、格式错误）应即时反馈，提示文本使用红色并靠近输入框下方；
        - 登录失败提示需醒目，避免遮挡操作区域。
2. 主界面板块（8～10个）
    - **工作台与设置界面**：

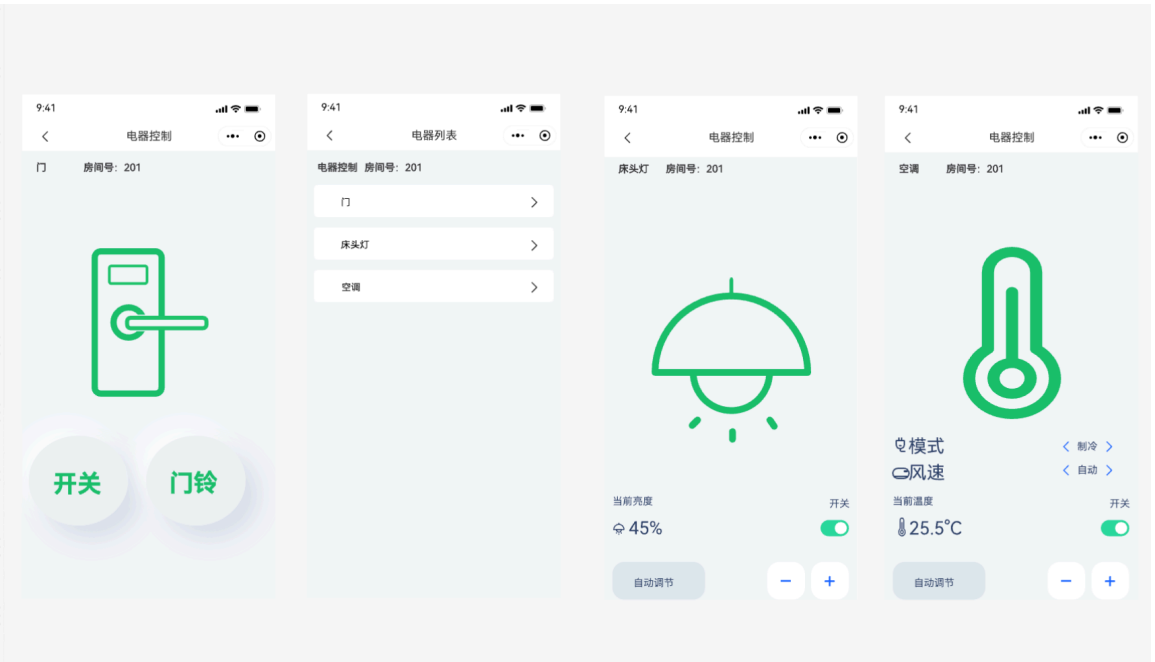
(工作台与设置界面)

- 统一采用卡片式组件布局，组件排列使用响应式栅格布局；
- 管理员与用户登录后加载的工作台组件样式应统一，但组件内容因角色而异；
- 每个功能组件图标清晰，按钮可点击范围需足够大，颜色深浅可用于角色功能区分（如管理员功能组件颜色更沉稳）；
- 设置界面表单布局紧凑有序，支持头像修改、密码变更等操作，操作按钮固定于页面底部，支持保存/取消。

- **模板管理界面**（管理员）：



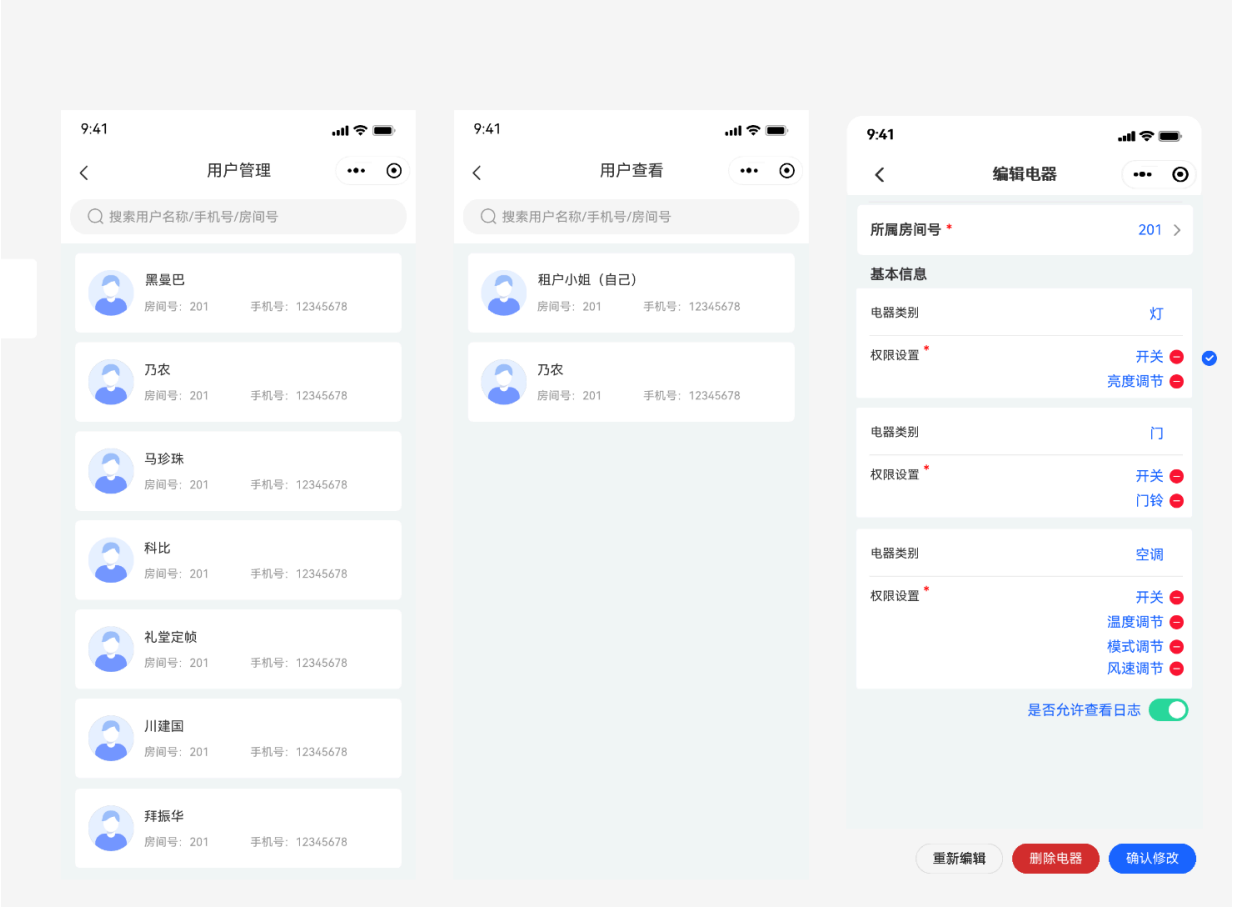
- *(模板管理界面)*
  - 模板列表使用卡片网格展示，每张卡片包含模板名称、创建时间与“编辑”按钮；
  - 模板详情页面应结构清晰，电器列表展示可控范围、绑定状态等，操作按钮排列规整，支持添加、删除、重命名；
  - 页面应保留“返回模板列表”快捷操作。
- **设备页面**（用户）：



- *(设备交互页面)*
  - 设备列表页面以图标+文字呈现房间当前电器状态（如开启、关闭、温度、亮度）；

- 控制页面风格贴近设备属性，如空调控制采用滑动温度条，灯光控制采用亮度调节器；
- 权限受限按钮显示为禁用灰色，并配文字提示如“无操作权限”。

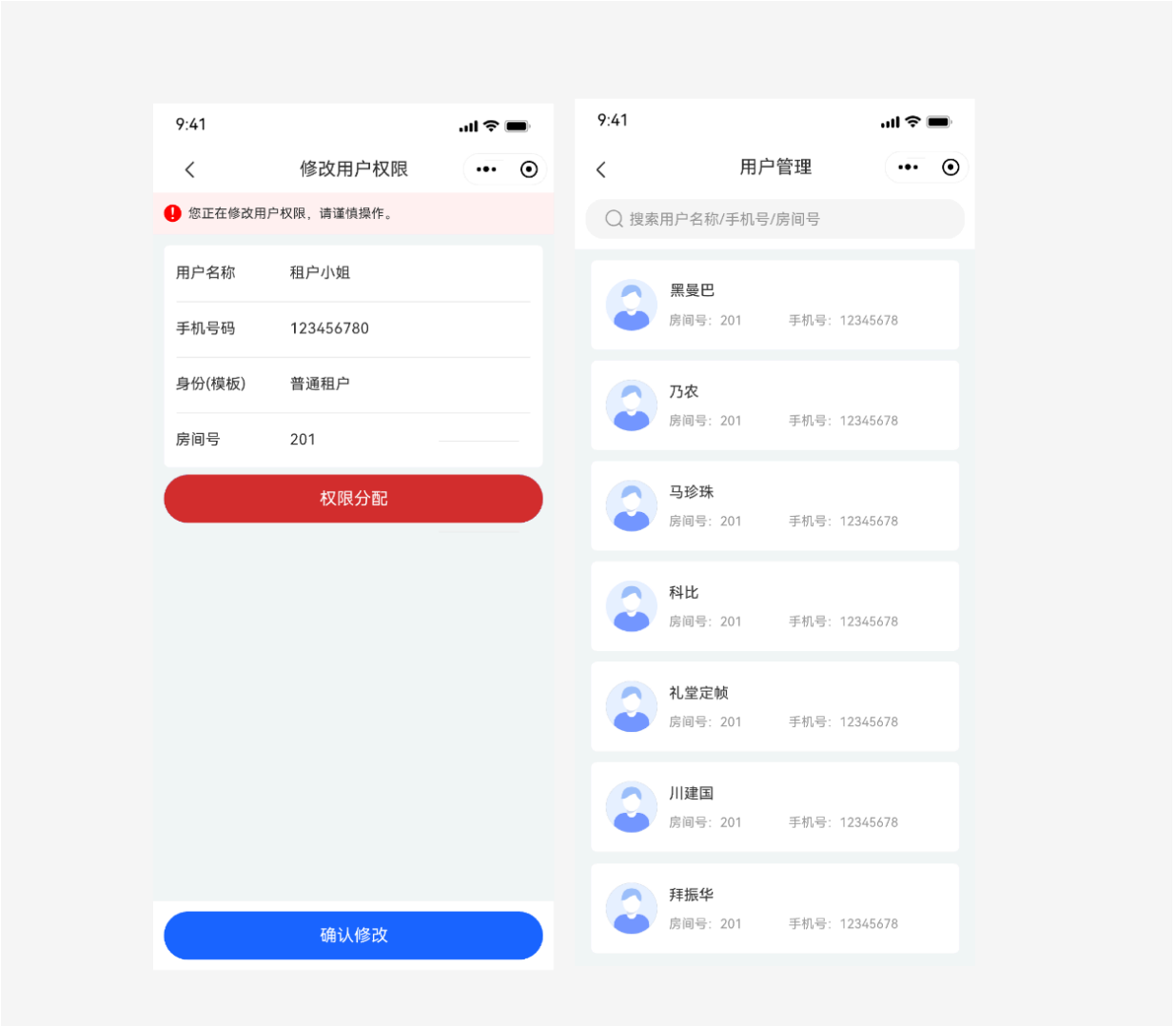
3. 权限编辑界面（3个）



(权限编辑与查看界面)

- **管理员权限列表页面**：
  - 使用列表或表格方式展示所有用户权限信息，支持搜索用户功能；
  - 每条用户信息右侧应提供“查看详情”按钮，点击后跳转至权限编辑页。
- **权限编辑页面（管理员）**：
  - 设备名称与权限类型以表单形式呈现，每项权限可使用开关按钮设置（查看/控制/管理）；
  - 页面底部固定“保存修改”按钮，改动成功后弹出确认提示框。
- **用户权限查看页面**：
  - 信息展示为只读模式，使用淡色边框框选权限项；
  - 无任何交互操作，仅提供“返回”按钮。

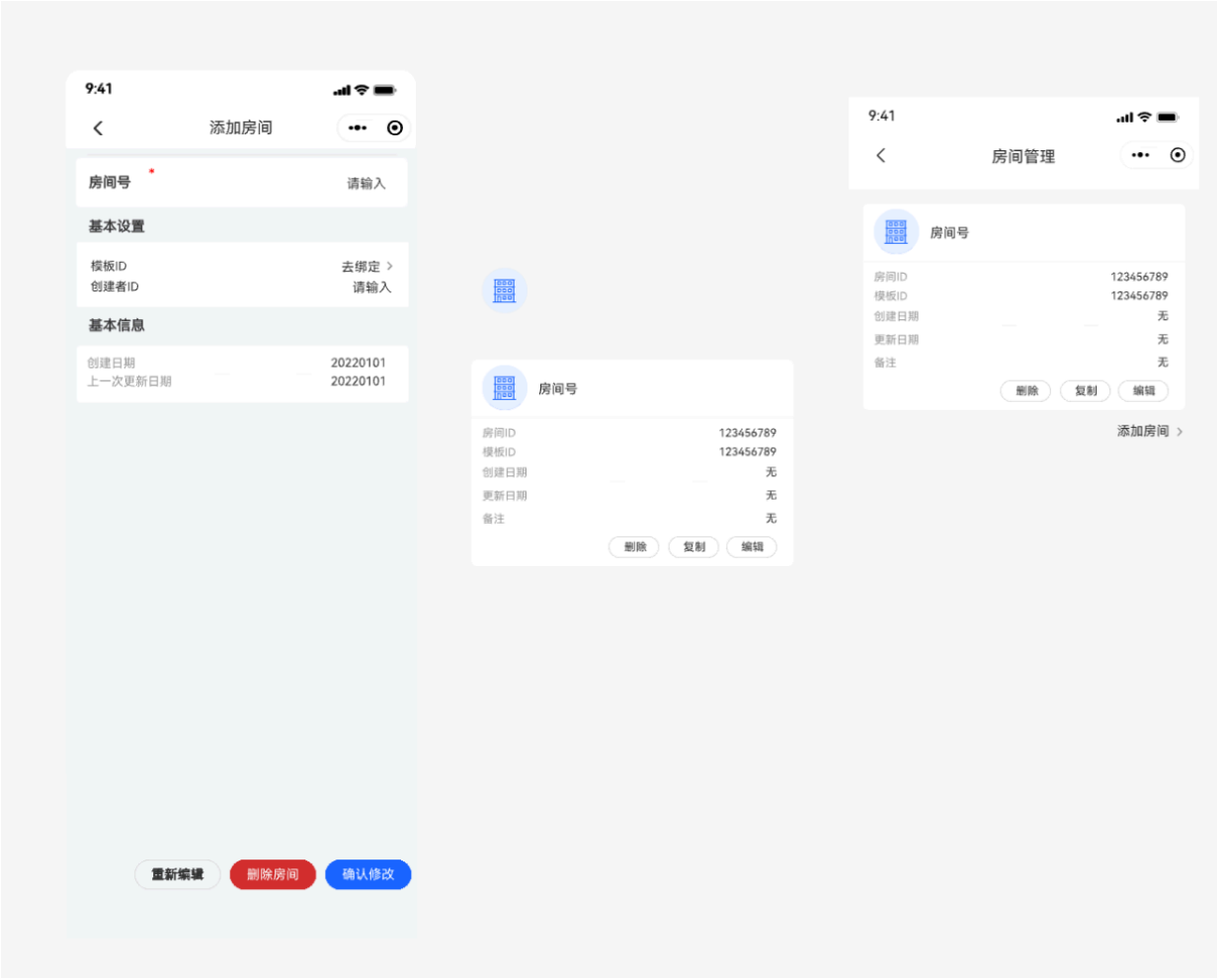
4. 用户查看/管理（2个）



(用户查看/管理界面)

- **管理员用户列表：**
  - 用户信息列表页使用头像+昵称+房间号组合展示，点击进入详情页；
  - 用户详情页中展示用户基本信息，并配有“更换房间”、“设为管理员”、“注销”按钮，按钮风格需明显区分（如“注销”按钮使用红色强调）。
- **用户查看房间内成员页面：**
  - 展示格式一致，但功能受限，仅显示用户名称与房间号，不提供操作按钮；
  - 禁止交互区域应使用浅灰色与文字提示区分。

5. 查看房间（2个）



(查看/编辑房间界面)

- **管理员房间列表页**：
  - 可以选择放行添加房间；
  - 卡片式展示所有房间，点击进入房间详情；
  - 房间详情页显示房型（模板）、电器种类与状态、创建时间等；
  - 页面底部保留“删除房间”操作按钮（红色、二次确认）。
- **用户房间信息页**：
  - 仅展示当前房间的模板信息与设备列表，不含任何操作功能；
  - 页面风格简洁，主要为信息展示型。

6. 日志查看（1个）

(日志查看界面)

- **风格统一**：
  - 使用分页表格或时间线风格展示日志条目；
  - 每条日志应包含操作时间、操作类型、目标对象、操作者等基本信息；
  - 管理员查看权限变更日志；用户查看个人权限及房间内电器使用日志。

7. 开发者界面（1个）

(开发者界面)

- **展示样式**：
  - 使用卡片或列表展示开发成员头像、昵称、角色与联系方式；
  - 页面底部可提供“联系我们”按钮，跳转外部链接或发送反馈邮件。

业务逻辑设计

核心业务流程

本系统的业务逻辑层是系统中数据处理的核心，负责用户交互产生数据的业务校验、组织存储与提取，并作为数据传输层与数据库之间的中介。同时也为未来可能接入的设备控制层提供接口扩展能力。

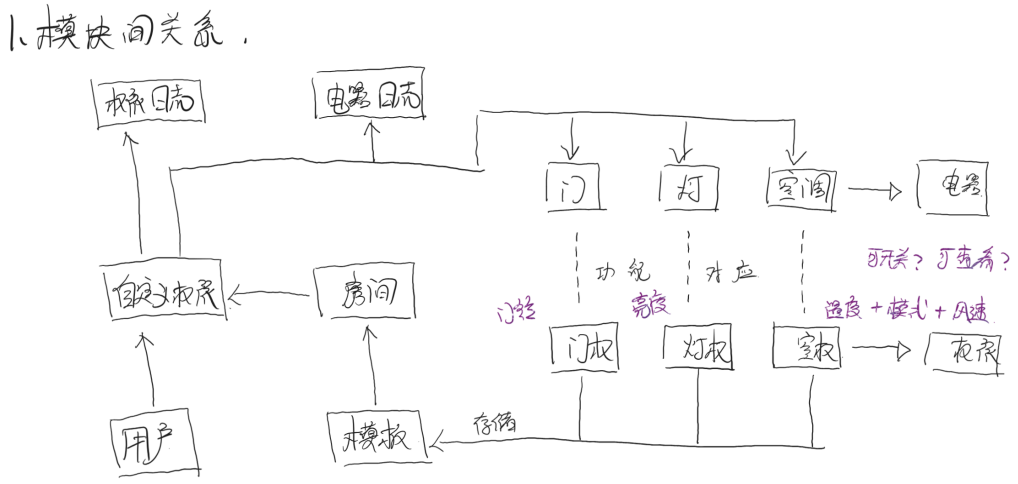


图2 模块间关系图

> TODO: [你应该在这里插入业务逻辑模块图，按照类图或者是其它的方式插入]

主要核心流程如下：

1. **权限控制模型建立**
   - 首先抽象出电器的通用权限，例如“开关权限”和“查看权限”作为权限基类。
   - 基于该权限基类，衍生出三类电器权限子类：门锁权限、灯光权限、空调权限。各自扩展了特有权限（如门铃权限、亮度调节、温度/模式/风速调节等）。
   - 权限类统一以布尔值形式定义是否开放某项控制，构成权限粒度细致的权限控制系统。
2. **模板与房间构建**
   - 电器权限类组合形成模板实体，表示某种房型的标准配置。
   - 模板由管理员用户创建，并指定其电器权限组成。
   - 房间是模板的实例，每个房间对应一个模板，并绑定一个唯一房间号。
   - 用户与房间形成 N:1 的关系，即多个用户可属于同一房间。
3. **个人权限个性化配置**
   - 模板提供默认电器权限设置，但在房间级别每位用户可以拥有独立的个性化权限（`UserPermission`），覆盖模板默认配置。

- 例如，同一房间内，家长用户可拥有完全控制权限，而孩子用户可能仅有查看或开关权限。
- 系统支持对个人权限的增删改操作，管理员和用户本人均可发起权限修改请求（依据身份限制）。

4. **用户控制流程**
   - 用户通过前端界面发起电器控制请求。
   - 后端接收到请求后，校验用户与房间绑定关系以及其个人权限。
   - 若权限允许，调用相应电器控制接口并记录日志；否则返回权限不足提示。
5. **权限与电器操作日志记录**
   - 系统对所有权限变更行为与电器控制行为进行记录。
   - 权限日志包括操作用户、被操作用户、时间与操作描述。
   - 电器日志包括操作用户、房间号、电器类型、时间及具体操作内容。
6. **模块解耦设计**
   - 电器本身不判断权限，权限判断逻辑应由业务逻辑或 UI 层完成。
   - 若未来需引入电器物理控制层，可在数据传输层并行添加“设备控制接口层”。

模块详细逻辑

以下针对关键模块展开详细功能设计：

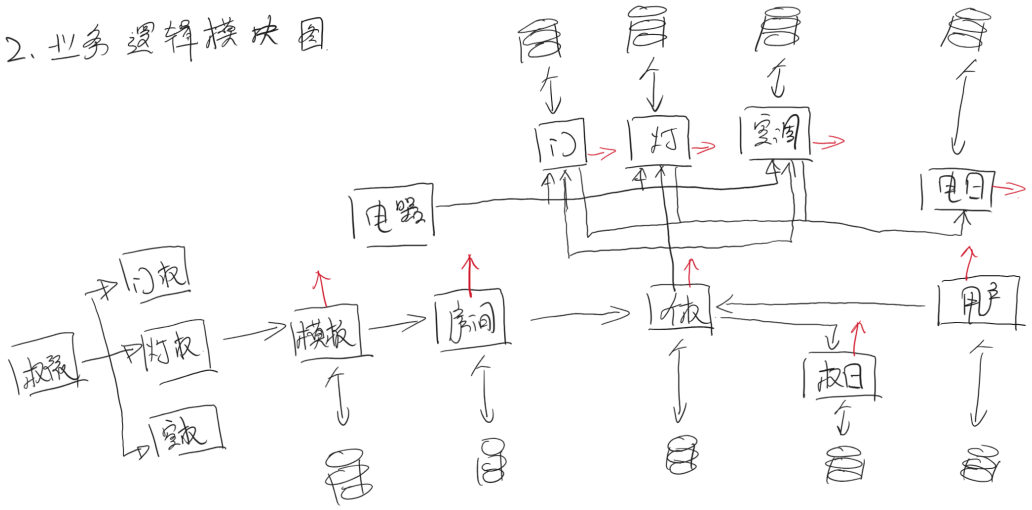


图3 业务逻辑模块示意图

> TODO: [如果还有时间的话，可以尝试在这里对每个模块都插入图] TODO: [那这样子的话，这里的图就应该是类图，而上边那个图就是构件图]

1. 电器模块（Device）
   - 定义一个电器基类，包含通用属性与方法：
     - `is_on`: 当前开关状态（Boolean）

        - `view_log()`: 查看当前电器的操作日志
    - 派生三类电器：
        - `SmartDoor`: 继承基础开关控制 + `bell_enabled`（门铃权限）
        - `SmartLight`: 继承基础 + `brightness_level`（亮度）
        - `SmartAC`: 继承基础 + `temperature`、`mode`、`wind_speed`，其中后两者为枚举类型
    - 每种电器类均实现日志生成方法 `log_operation()`，用于记录状态变更。
2. 权限模块（Permission）
    - 权限基类定义通用权限字段（如是否允许控制/查看电器）。
    - 派生电器权限类：
        - `DoorPermission`: 如门锁控制、门铃权限
        - `LightPermission`: 如灯光开关、亮度调节权限
        - `ACPermission`: 温度、模式、风速调节权限
    - 模板类 `Template`：组合上述权限类，并记录创建者ID。
    - 个人权限类 `UserPermission`：允许对模板权限进行个性化修改，绑定用户ID和房间。
3. 房间模块（Room）
    - 房间为模板的实例，包含：
        - `room_id`: 房间编号
        - `template`: 引用模板对象
    - 同时绑定房间当前三类电器状态（门/灯/空调）
4. 用户模块（User）
    - 包含用户基本信息与房间归属、管理员ID等。
    - 持有指向 `UserPermission` 的引用，用于权限判断。
    - 可调用系统接口完成对电器的控制或权限调整操作。
5. 日志模块（Log）
    - `PermissionLog`: 记录权限相关操作
    - `DeviceLog`: 记录电器控制操作
    - 两者皆以时间戳为索引，支持审计与溯源操作

异常处理机制

系统各模块在执行过程中需考虑输入校验、权限不足、数据缺失、状态冲突等异常场景，以下为各类异常情况及其处理策略：

> TODO: [对于每个异常，都需要添加异常处理状态图]

1. 用户权限不足

- **触发场景**：用户尝试控制无权限的电器。
- **处理机制**：
  - 在业务逻辑层拦截操作。
  - 返回错误码（403）及提示“权限不足，无法执行操作”。
  - 不执行控制指令，亦不记录成功日志。

2. 非法输入参数
   - **触发场景**：输入字段格式错误（如非法亮度、温度范围）。
   - **处理机制**：
     - 校验失败后返回400错误码。
     - 提示前端“输入不合法”，同时写入异常日志。
3. 电器状态冲突
   - **触发场景**：两个用户同时操作造成数据冲突。
   - **处理机制**：
     - 使用状态版本号或锁机制进行冲突控制。
     - 返回“设备状态冲突，请刷新后重试”。
     - 写入冲突失败日志。
4. 房间/用户/管理员数据缺失
   - **触发场景**：操作中引用了不存在的房间、用户或管理员。
   - **处理机制**：
     - 返回404错误码。
     - 提示“数据未找到，请确认信息正确”。
     - 同时记录异常事件。
5. 日志写入失败
   - **触发场景**：数据库日志写入超时或失败。
   - **处理机制**：
     - 主业务流程不回滚。
     - 日志写入失败项进入补偿队列或标记重试。
     - 提示系统管理员查看日志模块健康状态。
6. 权限变更冲突
   - **触发场景**：多个管理员同时调整用户权限。
   - **处理机制**：
     - 最新写入覆盖旧权限或使用时间戳控制版本。
     - 所有变更均写入权限日志。
7. 个人权限与模板权限冲突
   - **触发场景**：模板权限被修改后，个人权限未更新导致状态不一致。
   - **处理机制**：
     - 后台自动检测模板与个人权限冲突并提示管理员。
     - 若选择自动同步，则根据模板更新个人权限。

- 若选择保留差异，则仅记录日志供管理员审阅。

数据库设计

实体关系说明

在本系统中，数据库设计以用户权限控制与电器状态管理为核心，涉及用户、房间、模板、电器设备及日志等多个实体。这些实体之间通过合理的外键与逻辑绑定，共同支持系统功能的实现。主要实体关系如下：

> TODO: [这里补充一个EM图]

1. **用户与权限**
   - 每个用户（以电话为唯一标识）拥有一套权限配置，用于控制其在所属房间中对门、灯、空调等设备的操作权限。
   - 权限信息细化为布尔类型的控制位：门权限 3 项、灯权限 3 项、空调权限 5 项，支持精细化控制。
2. **用户与权限日志**
   - 权限日志记录用户权限的增删改行为，包括操作时间、操作者（执行权限操作的用户）、被操作对象（权限被更改的用户）及操作内容描述。
   - 操作者与被操作者均通过电话字段关联至用户表，用于审计和追责。
3. **用户与电器日志**
   - 电器日志记录用户对房间内电器的操作，包括时间、用户ID（电话）、房间号、电器类型及操作内容。
   - 日志中电器类型可用字符串（如“门”）或整型配合枚举实现，以支持前端显示与系统扩展。
4. **房间与模板**
   - 模板作为房型定义，描述房间所包含的设备类型（门、灯、空调）。
   - 每个房间通过模板 ID 与一个模板绑定，一个模板可被多个房间复用。
   - 模板由用户创建，创建者 ID 即为其电话号码。
5. **房间与设备状态**
   - 每个房间关联一条门状态、灯状态和空调状态记录，用于实时反映设备当前工作情况。
   - 状态信息包括门/灯是否开启，灯光亮度，空调开关状态、温度、风速与工作模式等。
6. **用户与房间**
   - 每位用户属于一个房间，通过房间号进行标识。

- 房间号作为多个表（如用户表、权限表、状态表、日志表）的外键，构成系统数据的统一索引依据。

数据表说明

以下为各核心数据表的字段说明与功能解析：

1. 用户表（`user`, User）
   - 字段：
     - 头像URL（`avatar_url`, String）：用户头像地址。
     - 电话（`phone`, String，主键）：用户唯一标识。
     - 房间ID（`room_id`, long）：用户所属房间。
     - 角色（`role`, int）：用户身份类型，后在业务逻辑层中以enum类型确定具体身份（管理员、普通用户）。
     - 密码（`password`, String）：登录密码。
     - 管理员电话（`admin_phone`, String）：用于标识该用户的管理员。
     - 用户创建日期（`created_at`, LocalDateTime）：用户的创建日期。
     - 用户更改日期（`updated_at`, LocalDateTime）：用户个人信息发生更改的最新日期。
   - 描述： 存储用户的基础信息，是权限管理的基础实体。建议将角色使用 Enum 枚举进行统一管理，数据库中使用整数表示。
2. 权限表（`permission`, Permission）
   - 字段：
     - 用户ID（`user_phone`, String，主键）：即用户电话。
     - 房间ID（`room_id`, long）：所控制的房间ID。
     - 门锁权限1～3（Boolean）：
       - 开关门锁权限（`door_open_permission`）
       - 日志查看权限（`door_check_permission`）
       - 使用门铃权限（`door_bell_permission`）
     - 灯权限1～3（Boolean）：
       - 开关电灯权限（`light_turn_on_permission`）
       - 日志查看权限（`light_check_permission`）
       - 亮度调整权限（`light_brightness_permission`）
     - 空调权限1～5（Boolean）：
       - 开关空调权限（`ac_turn_on_permission`）
       - 日志查看权限（`ac_check_permission`）
       - 温度调整权限（`ac_temperature_permission`）
       - 模式调整权限（`ac_mode_permission`）

        - 风速调整权限（ac_speed_permission）
    - 权限创建日期（created_at, LocalDateTime）：个性化权限的创建日期。
    - 权限更改日期（updated_at, LocalDateTime）：个性化权限信息发生更新的最新日期。
  - 描述： 每位用户在其房间中的设备控制权限配置，使用布尔值表示是否具备相应控制能力。该表与用户表是一一对应关系。
3. 权限日志表（permission_log, PermissionLog）
  - 字段：
    - 日志ID（id, long，主键）：权限日志的日志ID。
    - 时间（operation_time, DateTime）：权限更改发生的时间。
    - 被操作用户（target_user_phone, String）：权限被修改的用户电话。
    - 操作用户（operator_phone, String）：执行修改操作的用户电话。
    - 操作日志（operation_description, String）：权限更改的具体说明。
  - 描述： 记录权限的历史操作，用于安全审计与回溯权限变更过程。
4. 房间表（room, Room）
  - 字段：
    - 房间ID（room_id, long，主键）：唯一标识一个房间。
    - 房间号（room_number, String）：房间的号码
    - 模板ID（template_id, String）：指明该房间所属的房型模板。
    - 备注（note, String）：其它该房间需要备注的信息。
    - 创建日期（created_at, LocalDateTime）：房间创建的日期。
    - 更新日期（updated_at, LocalDateTime）：房间发生信息更改的最新日期。
  - 描述： 用于建立房间与房型之间的对应关系。每个房间绑定一个模板以确定其设备配置。
5. 模板表（template, Template）
  - 字段：
    - 模板ID（template_id, long，主键）：系统生成的唯一标识。
    - 模板名称（template_name, String）：作为方便管理员管理辨认的标识。
    - 创建者ID（creator_phone, String）：创建者电话。
    - 门状态（has_door, Boolean）：该模板是否包含门。
    - 灯状态（has_light, Boolean）：是否包含灯。
    - 空调状态（has_ac, Boolean）：是否包含空调。
    - 备注（note, String）：其它该模板需要备注的信息。
    - 创建日期（created_at, LocalDateTime）：房间创建的日期。
    - 更新日期（updated_at, LocalDateTime）：房间发生信息更改的最新日期。

- 描述： 模板定义一个房型的标准配置，适用于多个房间的批量初始化。

6. 门锁状态表（door_status, DoorStatus）
   - 字段：
     - 房间ID（room_id, long）：所属房间。
     - 开关状态（is_on, Boolean）：门锁当前是否开启。
     - 更新日期（updated_at, LocalDateTime）：门锁状态发生更改的最新日期。
   - 描述： 实时反映房间门的开关状态，每个房间一条记录。
7. 灯状态表（light_status, LightStatus）
   - 字段：
     - 房间ID（room_id, long）：所属房间。
     - 开关状态（is_on, Boolean）：电灯当前是否开启
     - 亮度状态（brightness, int）：亮度值，一般为1～100区间。
     - 更新日期（updated_at, LocalDateTime）：灯状态发生更改的最新日期。
   - 描述： 表示当前房间灯具的开关及亮度设置。
8. 空调状态表（air_conditioner_status, AirConditionerStatus）
   - 字段：
     - 房间ID（room_id, long）：所属房间。
     - 开关状态（is_on, Boolean）：空调当前是否开启。
     - 温度状态（temperature, int）：设置温度值。
     - 模式状态（mode, int）：工作模式（如制冷、制热、送风），可通过枚举映射。
     - 风速状态（speed, int）：风速档位，同样可枚举转换。
     - 更新日期（updated_at, LocalDateTime）：空调状态发生更改的最新日期。
   - 描述： 表示房间空调当前运行状态，支持多维调控。
9. 电器日志表（device_log, DeviceLog）
   - 字段：
     - 日志ID（id, long，主键）：电器日志的日志ID。
     - 时间（operation_time, LocalDateTime）：电器状态更改发生的时间。
     - 房间ID（room_id, long）：电器所对应的房间。
     - 用户ID（user_phone, String）：操作该设备的用户电话。
     - 电器（device_type, String）：设备类型，如门、灯、空调。
     - 相应操作（operation_detail, String）：操作内容，例如“空调调至25度”。
   - 描述： 记录用户对电器的操作，便于分析使用情况及进行问题追溯。
10. 说明：

- 所有“用户ID”与“创建者ID”均使用用户电话字段作为唯一标识。
- 所有设备状态表均以房间号为主索引，用于标识设备状态归属房间。
- 枚举类字段（如角色、空调模式、风速等）建议在系统中使用 Enum 管理，数据库中以 int 存储以提升查询性能。

接口设计

接口规范

为了保障前后端联调顺利、接口使用统一，系统所有 API 均遵循以下设计规范：

1. **统一请求格式**
   - 所有请求使用 RESTful 风格设计，URL 表示资源，HTTP 方法表示操作类型。
   - 请求数据统一使用 JSON 格式，采用 UTF-8 编码。
   - 请求 Header 应包含必要的认证字段（如 token）以识别用户身份。
2. **统一响应结构** 后端返回结果均封装为如下结构：

```
{
  "code": 200,
  "msg": "success",
  "data": { ... }
}
```

   - code：整型，表示状态码，200 表示成功，4xx/5xx 表示失败。
   - msg：字符串，返回结果的文字描述。
   - data：对象或数组，返回的业务数据内容；错误时可为空。
3. **状态码约定**

状态码	含义	说明
200	成功	正常响应
400	参数错误	请求参数缺失或格式错误
401	未认证	用户未登录或 Token 失效
403	无权限	操作超出用户权限范围

状态码	含义	说明
404	未找到资源	请求路径或资源不存在
500	服务端错误	后端抛出异常或处理失败

4. **接口安全机制**
    - 所有接口应在用户登录后携带 Token 进行访问，防止越权操作。
    - 日志相关接口仅限管理员用户访问，用于安全审计与追责。
    - 权限控制操作须校验操作者身份与权限，防止普通用户越权修改。

API详细说明

为支持智能控制系统的各类功能模块，系统共定义若干 API 接口，按功能模块划分如下：

1. 用户管理模块（登录 / 注册 / 用户信息）

接口名称	描述	请求方式	请求参数	响应字段	关联数据表
`/api/login`	用户登录验证	POST	`phone`，`password`	`code`，`msg`，是否成功登录等	用户表
`/api/register`	用户注册	POST	用户基本信息	`code`，`msg`，是否注册成功	用户表
`/api/checkUser`	校验手机号与管理员是否存在	GET	`phone`，`managerPhone`	`exists`（bool），提示信息等	用户表
`/api/user/info`	获取当前用户信息	GET	当前用户登录态	`avatar`，`phone`，`role`，`roomNo`	用户表

接口名称	描述	请求方式	请求参数	响应字段	关联数据表
/api/user/update	更新用户个人信息	POST	用户更新字段	code，msg	用户表

备注：

- api/checkUser 的响应信息应包含两个 exists，分别为 phone 是否已存在，及 managerPhone 是否存在。前端调用的时候若 phone 对应的返回 exists 为 true，则需要提示用户电话已存在，不得重复注册；若 managerPhone 对应的返回 false，则需要提示管理员不存在。
- api/user/info 当中当前用户登录态指当前用户是否已成功登录（bool），如果已登录则返回相应的用户信息。该接口中用户登录态默认为 true。下文所指用户登录态同理。此处另行设置请求类型首先保证接口完整性，其次为二次保险二次防范。如技术实现时认为实无必要可对接口做适当增删改。

2. 权限管理模块（用户权限配置）

接口名称	描述	请求方式	请求参数	响应字段	关联数据表
/api/permission/info	获取指定用户或当前用户的权限信息	GET	user_phone	各权限项（门/灯/空调权限位）	权限表
/api/permission/update	更新用户权限配置	POST	权限布尔配置项	code，msg	权限表

接口名称	描述	请求方式	请求参数	响应字段	关联数据表
/api/permission/editors	查询该管理员所管辖的所有用户信息	GET	admin_phone	用户电话列表	用户表

备注：

- api/permission/update 中的权限布尔配置项为方便处理，在UI界面将所有的权限布尔配置项处理完后，传输所有的权限布尔配置项到业务逻辑层后再行处理。

3. 房间与模板管理模块

接口名称	描述	请求方式	请求参数	响应字段	关联数据表
/api/room/info	获取房间信息（如房间号、模板ID）	GET	roomNo	房间号，模板ID，房间内人数，房间创建日期	房间表
/api/room/save	创建或更新房间信息	POST	房间号、模板ID	code，msg	房间表
/api/template/list	获取所有房型模板信息	GET	无	模板ID，包含设备类型	模板表
/api/template/save	创建或更新模板信息	POST	模板数据	code，msg	模板表

4. 工作台统计信息接口

接口名称	描述	请求方式	请求参数	响应字段	关联数据表
/api/dashboard/stats	获取当前房间的统计信息	GET	user_phone	userCount，deviceCount	权限表，用户表

备注：

- 注意 userCount 及 deviceCount 并不在数据表中，需要业务逻辑层进行数据处理后才可返回相应的值。

5. 设备控制与状态接口

接口名称	描述	请求方式	请求参数	响应字段	关联数据表
/api/device/permission	获取当前用户对设备的权限	GET	当前用户登录态	权限布尔位信息	权限表
/api/device/status	获取当前房间设备的实时状态	GET	roomNo	状态字段（开关/亮度等）	设备状态表
/api/device/updateStatus	更新设备状态（如开关、亮度、温度）	POST	状态字段与房间号	code，msg	设备状态表

备注：

- 此处所指的当前房间设备的实时状态及更新设备状态均分别对三个不同的设备进行接口的实现，此处为设计书的简洁不重复赘述。

6. 日志模块（权限与设备操作记录）

接口名称	描述	请求方式	请求参数	响应字段	关联数据表
`/api/log/permission`	获取权限变更日志	GET	可选筛选参数	操作时间、操作者、操作内容等	权限日志表
`/api/log/device`	获取设备操作日志	GET	可选筛选参数	操作时间、设备类型、操作内容等	电器日志表

备注：

- `/api/log/permission` 当中的可选筛选参数包括选择查询的操作时间间隔（默认全部），选择查询的操作者（默认全部），选择查询的被操作者（默认全部）。
- `/api/log/device` 当中的可选筛选参数包括选择查询的操作时间间隔（默认全部），选择查询的设备类型（默认全部），选择查询的操作者（默认全部）。
- 所有的时间间隔参数传递应为传递起始时间及结束时间，粒度按天级计算。

非功能性设计

YatSpark 系统在非功能性设计方面，致力于确保系统的性能、可用性、安全性、可维护性与可扩展性，满足在智能宿舍或酒店租户管理场景下的长期稳定运行与良好用户体验。

1. 性能要求

- **响应时间**：系统应在用户进行常规操作（如权限查看、设备控制、日志查询）时，页面响应时间不超过 2 秒，设备控制指令的执行反馈不超过 1 秒。
- **并发支持**：支持至少 500 个并发在线用户，系统具备自动负载调度能力，保障高峰期流畅访问。
- **数据处理能力**：日志模块可支撑每天不少于 5 万条权限变更与设备使用日志的存储与查询操作。

2. 可用性设计

- **系统可用性**：保证 99.9% 的年平均正常运行时间（Annual Uptime），系统更新维护应安排在低使用时段，并提供提示与恢复策略。
- **界面一致性**：各模块界面统一风格与操作逻辑，降低用户学习成本，支持快速上手。
- **设备兼容性**：系统前端兼容主流浏览器与移动端 WebView，支持嵌入微信小程序运行环境。

3. 安全性设计

- **身份验证机制**：采用双重验证机制，用户登录需输入账号密码并绑定手机号/邮箱验证，支持验证码找回密码。
- **权限隔离机制**：多角色用户分级访问控制，后台管理接口需验证身份与角色权限，防止越权操作。
- **日志审计机制**：对每一次权限变更、电器控制操作均自动生成日志，支持时间、用户、对象多维度检索，便于追溯与审查。
- **数据加密传输**：客户端与服务端通信采用 HTTPS 协议，敏感信息（如密码、用户信息）使用加盐哈希与对称加密存储。

4. 可维护性设计

- **模块解耦架构**：系统采用前后端分离架构，后端接口统一使用 RESTful API 标准，便于模块更新与迭代维护。
- **配置化支持**：模板配置、权限规则通过后端配置项实现灵活调整，减少代码改动对业务逻辑的侵入。
- **异常处理机制**：前后端统一设计错误码与提示语，用户端友好提示，开发端完整日志记录并邮件预警关键异常。

5. 可扩展性设计

- **角色可扩展**：系统用户角色设计基于通用类继承机制，支持未来引入新角色（如保安、家政人员等）与其权限模型。
- **设备可扩展**：电器模型与控制接口采用抽象工厂模式设计，新增设备类型仅需实现其控制接口并注册即可集成。

- **平台对接能力**：系统预留第三方平台接入接口（如门禁系统、视频监控平台），支持以 Webhook 或开放 API 形式对接。

6. 本地化与国际化支持（预留）

- **多语言支持**：界面文案抽离为国际化资源文件，后期可快速添加多语言包。
- **时区适配**：系统日志与操作时间基于服务器统一时间戳展示，支持用户自定义时区显示。