特别提示：

* 本文件必须按照“必须知道，最少授权”的原则交付，不得全部或部分提供给任何未经授权的人。
* 如果本文件不在您的访问权限范围内，应立即销毁，不得存储、传输、复制、打印或使用本文件的任何内容。

移动开发平台

客户端组件详细设计说明书

2023年03月29日

文档修订记录

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 版本 | \*变化  状态 | 修订说明 | 日期 | 变更人 | 备注 |
| V0.1 | A | 新建 | 2023/03/29 | persilee |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

\*变化状态：A——增加，M——修改，D——删除

目录

[目录 1](#_Toc1804029415)

[1 引言 2](#_Toc765912)

[1.1 编写目的 2](#_Toc2135264749)

[1.2 背景 2](#_Toc795411426)

[1.3 术语定义 2](#_Toc394134207)

[1.4 参考文档 2](#_Toc1374049701)

[2 系统功能 3](#_Toc1761668516)

[2.1 功能架构 3](#_Toc1005707223)

[2.2 功能模型 4](#_Toc77511424)

[2.3 功能概述 5](#_Toc1359413086)

[3 系统设计 8](#_Toc577215969)

[3.1 系统上下文 8](#_Toc1085157484)

[3.2 系统容器 9](#_Toc1810703264)

[3.3 接口设计 10](#_Toc498996411)

[3.3.1 OTP发送接口 10](#_Toc709038142)

[3.4 数据库设计 13](#_Toc417295391)

[3.4.1 总体数据模型 13](#_Toc1949529082)

[3.4.2 数据库表结构 13](#_Toc1577278895)

[4 系统关键方案 15](#_Toc788249697)

[4.1 角色权限管理 15](#_Toc286039136)

[4.1.1 功能概述 15](#_Toc1391356766)

[4.1.2 系统架构及技术原理 16](#_Toc583733979)

[5 系统可扩展设计 21](#_Toc1111685557)

[5.1 数据分表 21](#_Toc991427599)

# 引言

## 编写目的

本文档是详细说明了用户中心的整体架构、功能列表、业务流程设计、接口设计、数据库设计和与其他系统的接口关系。在项目实施阶段，将依据本文档进行系统的开发。

## 背景

用户中心是为其他业务系统提供用户基础数据服务，比如用户信息查询、用户信息注册、用户密码校验等。用户中心只处理简单的业务逻辑以及数据存储，是调用链路的末端，不依赖调用其他服务，只依赖数据库或者Redis。

## 术语定义

|  |  |
| --- | --- |
| 术语 | 定义 |
| back-end | 业务组合处理的产品后端，如app back-end 和portal back-end |

## 参考文档

1. 移动开发平台概要设计文档.docx

# 系统功能

## 功能架构

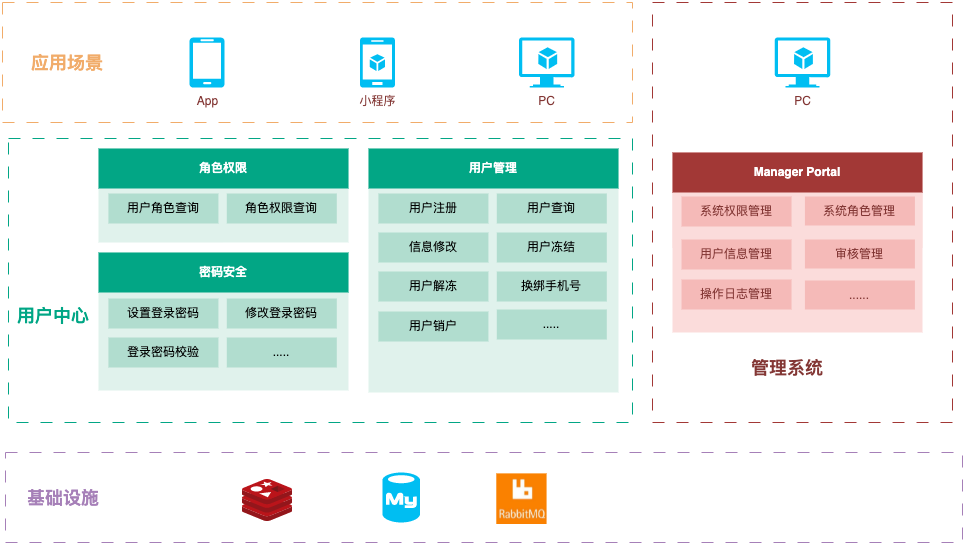


图 2‑1-1 功能架构图

## 功能模型

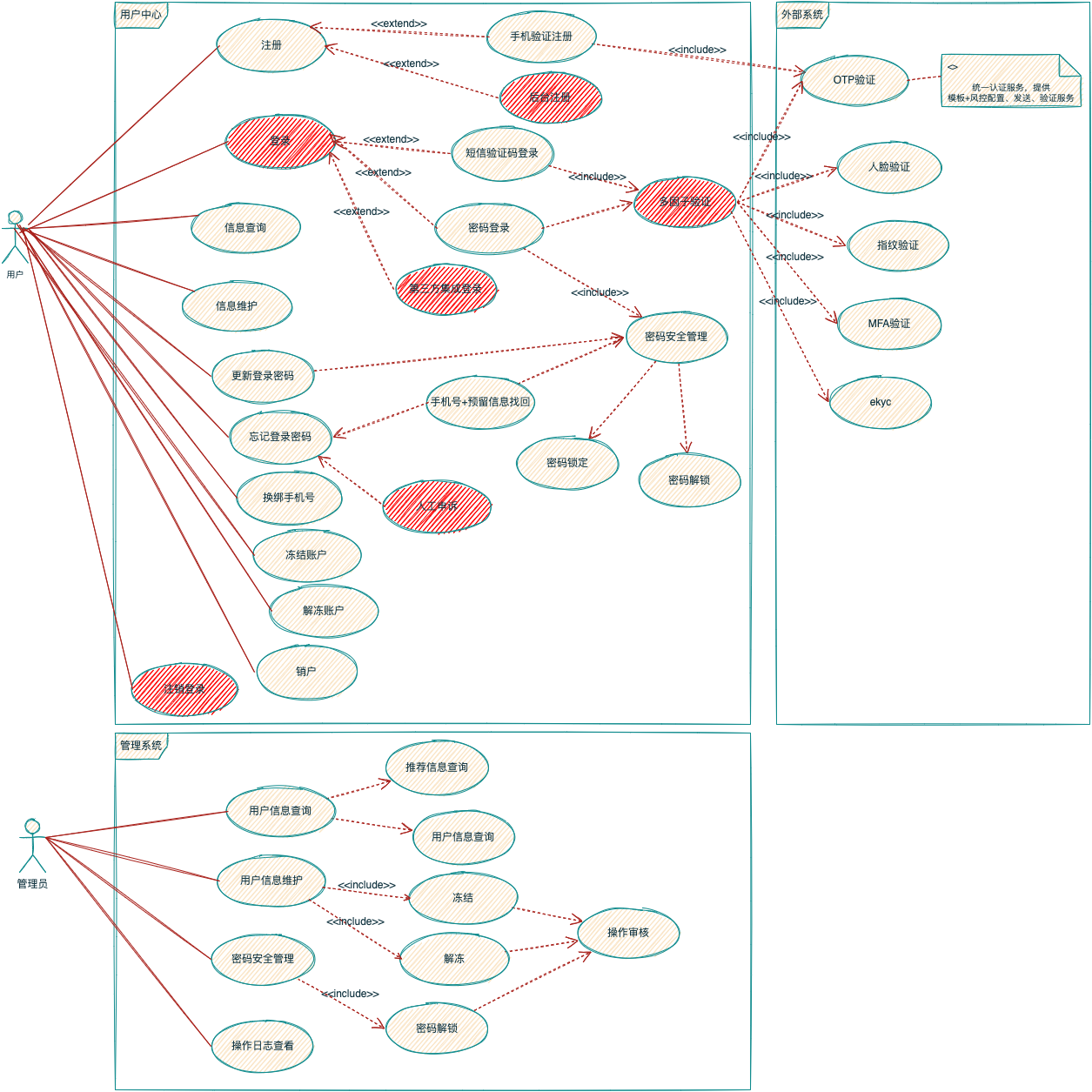


图 2‑2-1 用户管理



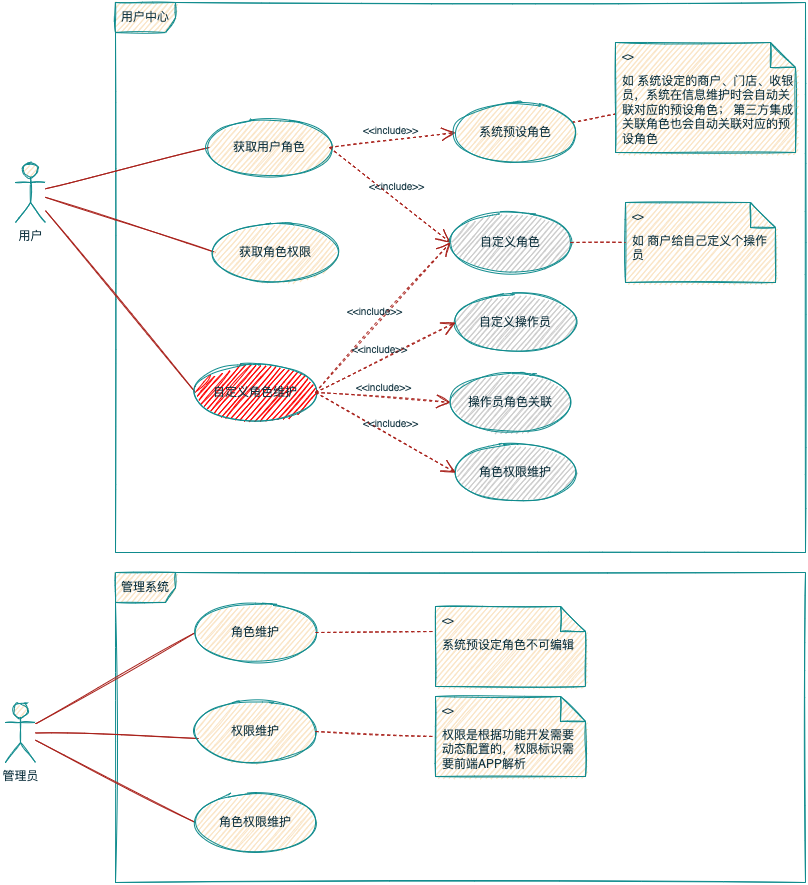


图 2‑2-2 角色权限

## 功能概述

用户管理包括基础的用户信息维护和查询，用户数据是应用系统最基础的核心模块，所有功能都围绕用户来展开。

|  |  |  |
| --- | --- | --- |
| 功能模块 | 功能 | 功能说明 |
| 用户管理 | 用户信息注册接口 | 由前置back-end验证后调用 |
| 用户信息查询接口 | 查询用户信息 |
| 用户信息修改接口 | 用户基本信息修改 |
| 用户冻结 | 由前置back-end 验证后调用 |
| 用户解冻 | 由前置back-end 验证后调用 |
| 换绑手机号 | 由前置back-end 验证后调用 |
| 用户销户接口 | 由前置back-end检查满足销户条件后调用 |
| 设置登录密码接口 | 首次设置登录密码或者重置时设置 |
| 修改登录密码接口 | 使用原密码修改 |
| 登录密码校验接口 | 账号密码校验时使用，会进行密码安全控制 |
| 角色权限 | 用户角色查询 |  |
| 角色权限查询 |  |
| Manager Portal | 系统权限管理 |  |
| 系统角色管理 |  |
| 用户信息管理 |  |

系统权限管理

|  |  |  |
| --- | --- | --- |
| 功能点 | 状态 | 备注 |
| 列表查询 | ✓ |  |
| 详情查询 | ✓ |  |
| 列表下载 | ✓ |  |
| 新增权限 | ✓ |  |
| 导入权限 | ✓ |  |
| 修改权限 | ✓ |  |
| 删除权限 | ✓ |  |
| 状态维护 | ✓ | 启用、停用 |
| 操作审核 | ✓ |  |
| 操作日志查看 | ✓ |  |

系统角色管理

|  |  |  |
| --- | --- | --- |
| 功能点 | 状态 | 备注 |
| 列表查询 | ✓ |  |
| 详情查询 | ✓ |  |
| 列表下载 | ✓ |  |
| 新增权限 | ✓ |  |
| 导入权限 | ✓ |  |
| 修改权限 | ✓ | 系统预设角色不允许编辑 |
| 删除权限 | ✓ | 系统预设角色不允许删除 |
| 状态维护 | ✓ | 启用、停用 |
| 角色授权 | ✓ |  |
| 角色权限查看 | ✓ |  |
| 操作审核 | ✓ |  |
| 操作日志查看 | ✓ |  |

用户信息管理

|  |  |  |
| --- | --- | --- |
| 功能点 | 状态 | 备注 |
| 列表查询 | ✓ |  |
| 详情查询 | ✓ |  |
| 列表下载 | ✓ |  |
| 状态维护 | ✓ | 冻结、解冻 |
| 密码解锁 | ✓ | 登录密码锁定后可解锁 |
| 操作审核 | ✓ |  |
| 操作日志查看 | ✓ |  |
| 明文查看 | ✓ | 默认敏感信息是脱敏感敏，具备明文权限可支持查询明文 |

# 系统设计

## 系统上下文

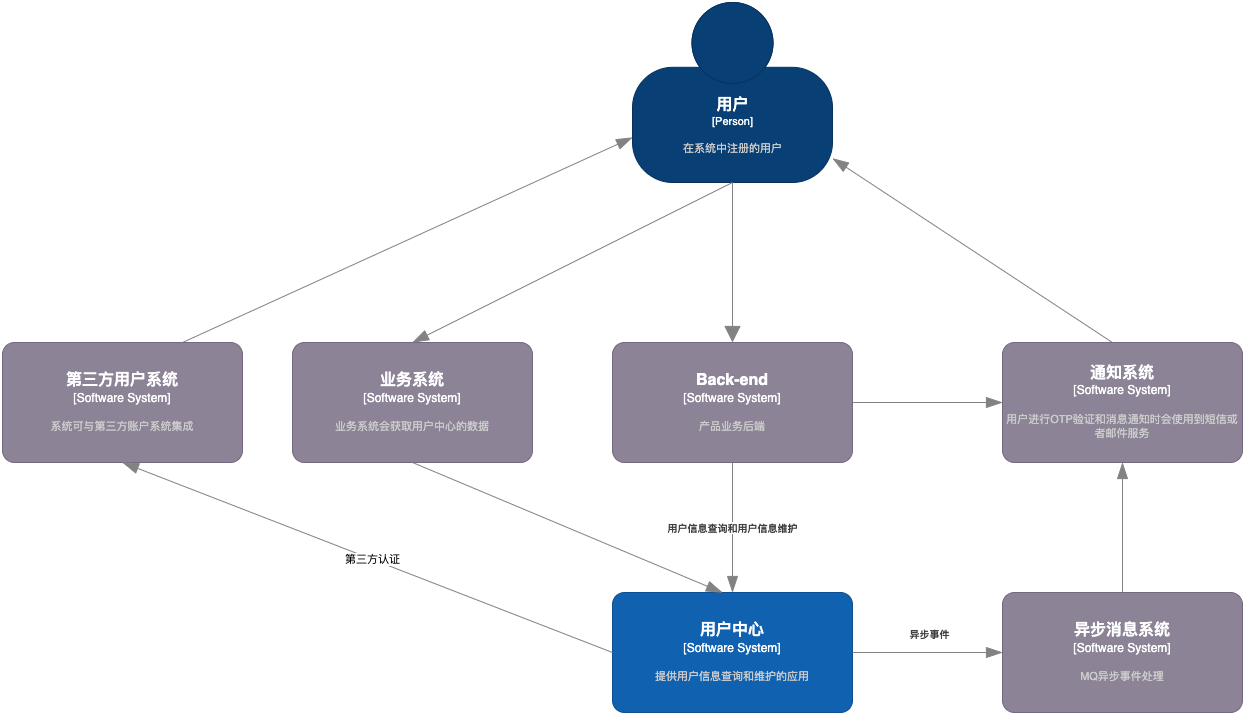


图 3‑1-1 系统关系

用户中心

提供原子数据查询和操作接口服务，如用户信息查询。为了尽可能实现最短链路达到最佳性能的目的，设计为只依赖数据库且做读写分离。

Back-end

聚合了各种业务逻辑和服务调用，比如登录时候需要校验的风控或者短信，由于APP、WEB、API 各方式的安全校验和交互流程方式都不一样需要拆分成不同的接口来提供服务，这层差异化的API都由Back-end来提供。

业务系统

微服务系统群中有其他业务模块需要获取用户数据，如交易系统需要获取用户数据进行交易检查和数据初始化。

异步消息系统

异步消息生产和消费，如用户注册后发送邮件通知和奖励积分。

## 系统容器

完整的用户系统由客户端、Back-end、网关、用户系统、管理系统组成。

客户端包括Web 和App，客户通过客户端业务组件发起请求，再由Back-end编排通过网关转发到用户系统完成用户数据的读写操作。

管理系统通过直接访问数据库的方式进行读写操作完成运营维护相关的功能实现。

系统处理流程图如下：

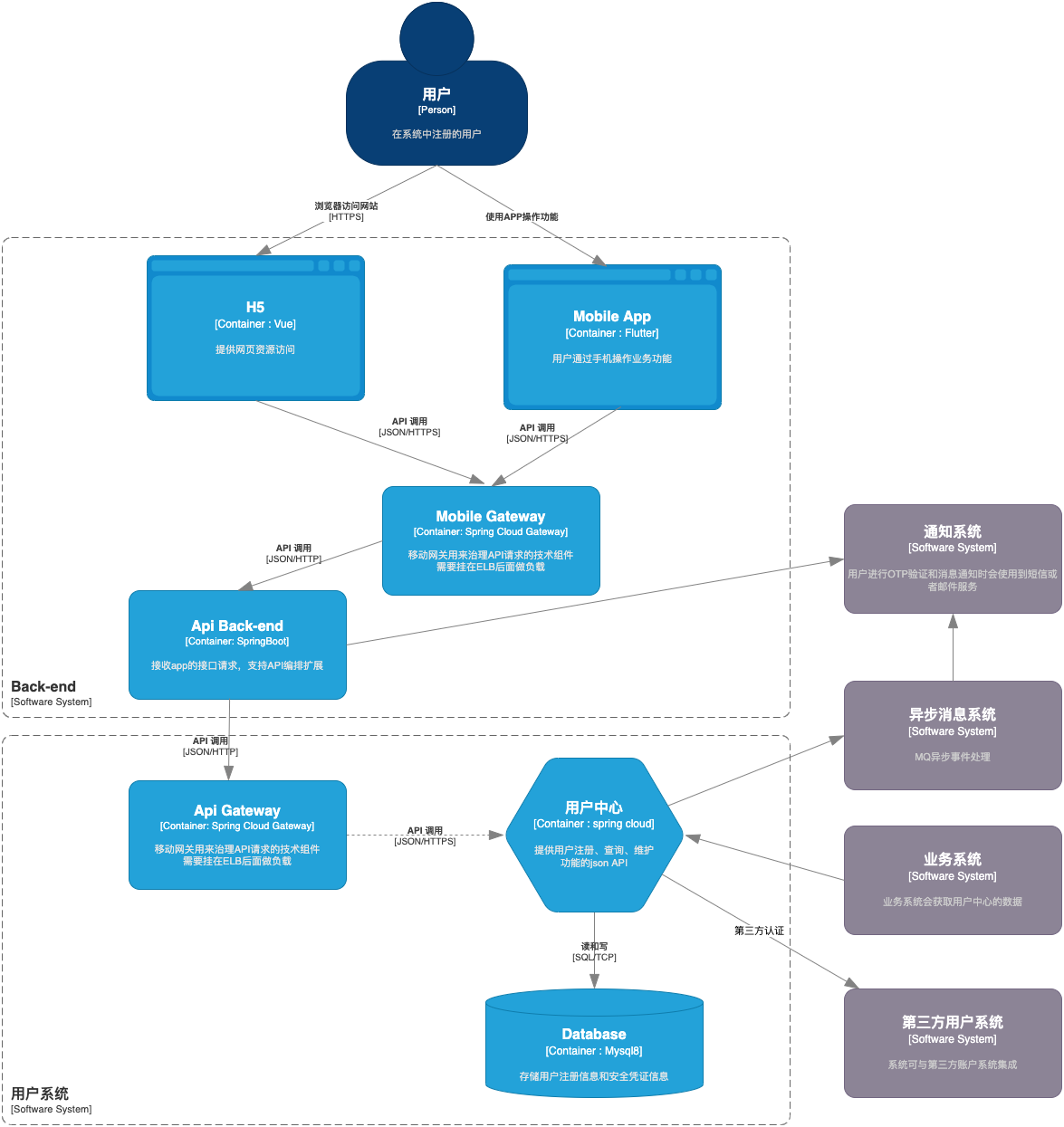


图 3‑2-1 应用逻辑关系

## 接口设计

### OTP发送接口

接入业务系统根据定义的通知模板ID所需要的补充的内容调用该接口完成内容组装完成并调用渠道完成发送到客户终端设备。

#### 设计方案

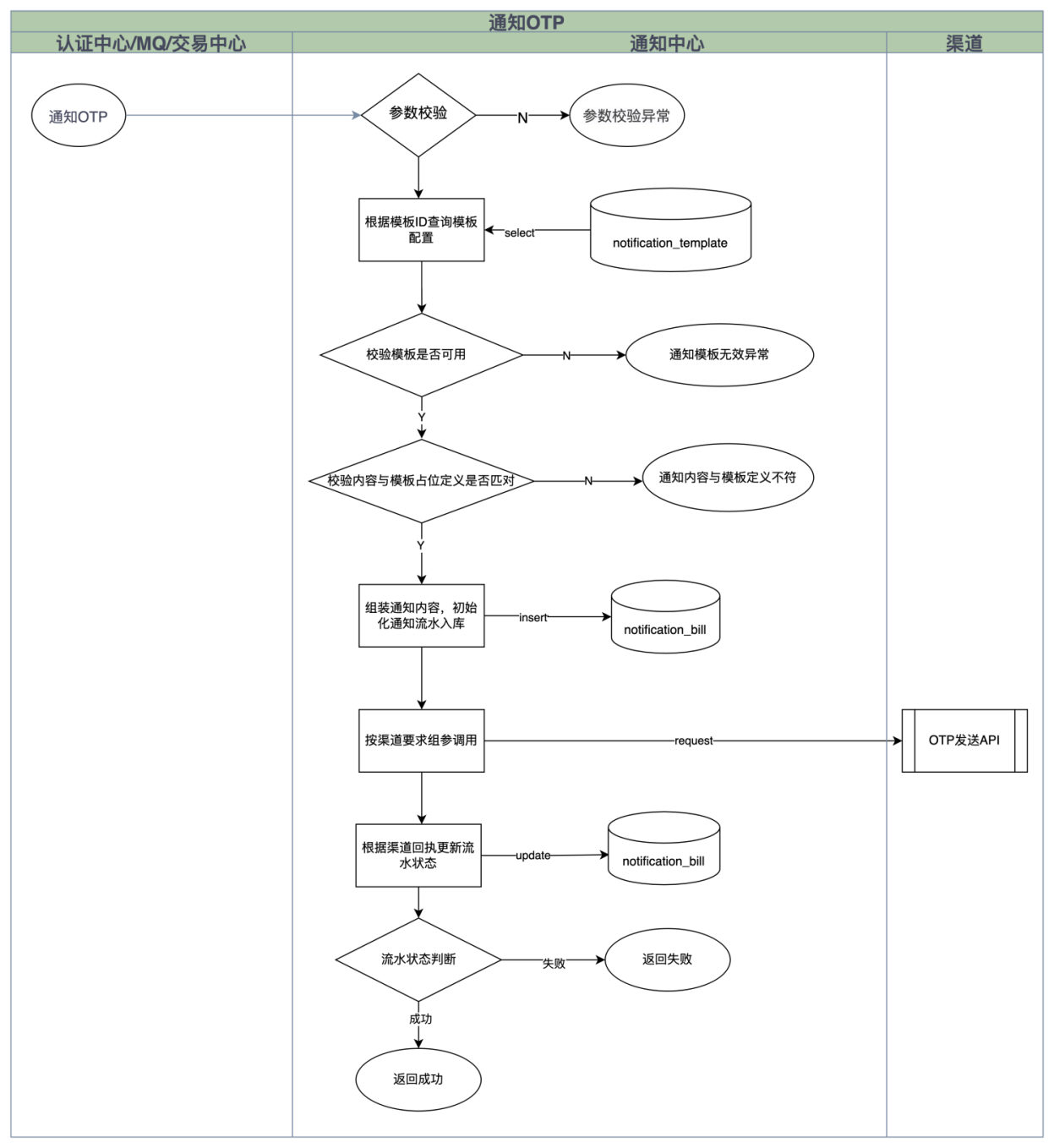


图 3‑3-1-1-1 应用逻辑关系

设计说明：

* 根据模板ID查找模板组装内容，可进行模板格式正则校验。
* 路由切换与配置通过NACOS进行配置切换，并在流水记录对应渠道信息。

#### 请求参数

| 参数名称 | 是否必须 | 示例 | 备注 |
| --- | --- | --- | --- |
| refTransId | 是 |  | 关联流水号 |
| productNo | 是 |  | 产品ID |
| terminalIp | 否 |  | 终端IP |
| notifyContent[0] | 是 |  | 通知内容，有序，第一个就是模板的第一个占位符，以此类推 |
| notifyTemplateNo | 否 |  | 模板No |
| mobile | 是 |  | 手机号 |

#### 响应参数

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 是否必须 | 默认值 | 备注 | 其他信息 |
| system | string | 必须 |  |  |  |
| state | integer | 非必须 |  |  |  |
| code | string | 必须 |  |  |  |
| message | string | 必须 |  |  |  |
| detail | object [] | 非必须 |  |  | item 类型： object |
| ├─ field | string | 非必须 |  |  |  |
| ├─ value | string | 非必须 |  |  |  |
| ├─ issue | string | 非必须 |  |  |  |
| data | object | 非必须 |  |  |  |

## 数据库设计

### 总体数据模型

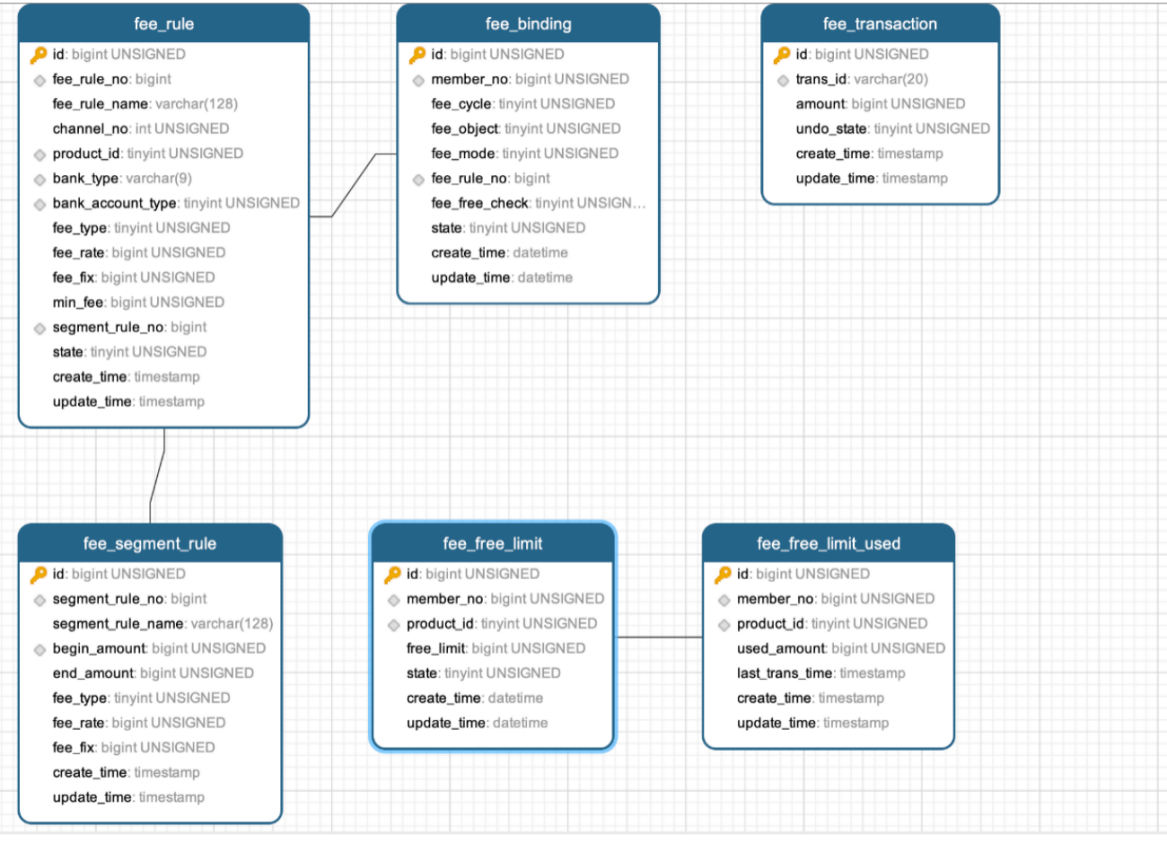


图 3‑4-1-1 数据库设计

### 数据库表结构

#### fee\_binding （计费绑定）

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 编号 | 名称 | 数据类型 | 长度 | 小数位 | 允许空值 | 主键 | 默认值 | 说明 |
| 1 | id | bigint | 20 | 0 | N | Y |  | id |
| 2 | member\_no | bigint | 20 | 0 | N | N |  | 关联客户号，商户、个人用户、代理商、机构都称之为客户 |
| 3 | fee\_cycle | tinyint | 3 | 0 | N | N | 1 | 计费周期，1实时扣收 |
| 4 | fee\_object | tinyint | 3 | 0 | N | N |  | 计费对象，1转入方 2转出方 |
| 5 | fee\_mode | tinyint | 3 | 0 | N | N |  | 扣费模式，1外扣模式，2内扣模式 |
| 6 | fee\_rule\_no | bigint | 20 | 0 | N | N |  | 计费规则编号 |
| 7 | fee\_free\_check | tinyint | 3 | 0 | N | N | 1 | 计费免费策略检查标识，0－开启；1禁用 |
| 8 | state | tinyint | 3 | 0 | N | N | 0 | 状态，0启用1禁用 |
| 9 | create\_time | datetime | 19 | 0 | N | N | CURRENT\_TIMESTAMP | 创建时间 |
| 10 | update\_time | datetime | 19 | 0 | N | N | CURRENT\_TIMESTAMP | 更新时间 |

#### fee\_free\_limit （计费免费额度配置）

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 编号 | 名称 | 数据类型 | 长度 | 小数位 | 允许空值 | 主键 | 默认值 | 说明 |
| 1 | id | bigint | 20 | 0 | N | Y |  | id |
| 2 | member\_no | bigint | 20 | 0 | N | N |  | 关联客户号，商户、个人用户、代理商、机构都称之为客户 |
| 3 | product\_id | tinyint | 3 | 0 | N | N |  | 产品ID，0－全部 |
| 4 | free\_limit | bigint | 20 | 0 | N | N |  | 免费总额度 |
| 5 | state | tinyint | 3 | 0 | N | N | 0 | 状态，0启用1禁用 |
| 6 | create\_time | datetime | 19 | 0 | N | N | CURRENT\_TIMESTAMP | 创建时间 |
| 7 | update\_time | datetime | 19 | 0 | N | N | CURRENT\_TIMESTAMP | 更新时间 |

# 系统关键方案

## 角色权限管理

### 功能概述

用户在实际应用场景中存在不同角色的定义，系统需要支持不同角色的前台功能菜单展示、后台API权限校验。具体有下面几种场景。

系统设定角色

如商家APP有 商户、门店、收银员三种不同的角色，对应操作的功能权限也都有区别。

用户自定义角色

如商家可自己创建操作员用来管理自己的交易和数据报表查看的需要。

登录用户类型关联不同角色

如微信关联登录后可简单查询交易，通过网银账号密码登录可使用完整功能

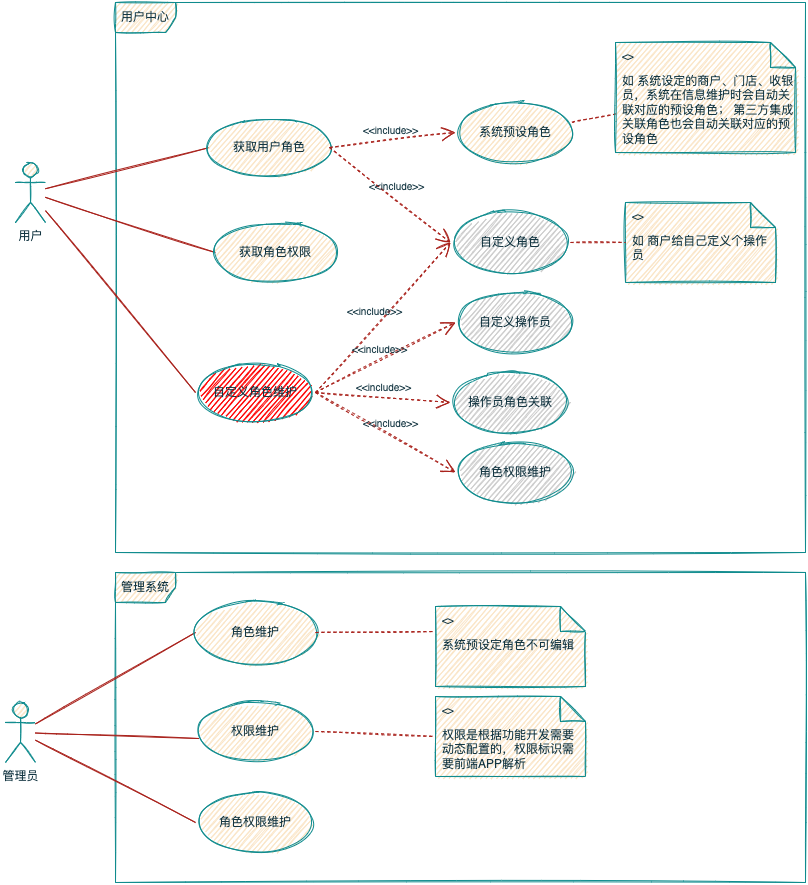


图 4-1-1-1 角色权限

### 系统架构及技术原理

为了方便维护大量用户的权限，系统采用用户－角色－权限的RBAC权限模型设计。其核心是引入角色的概念，用角色作为中间介质，使用户与权限配置更加灵活。

#### 角权限控制模型

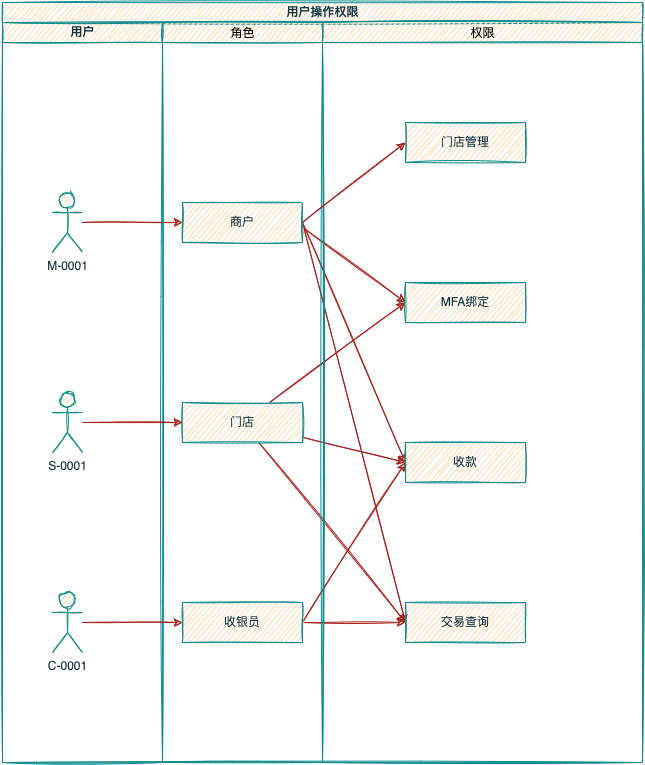


图 4-1-2-1 角权限控制模型

* 用户和角色关联，如果为系统预设角色则只支持1:1关联，如果是自定义操作员则可自1对多维护用户角色
* 角色和权限关联支持1对多
* 权限的定义需要程序解析和支持

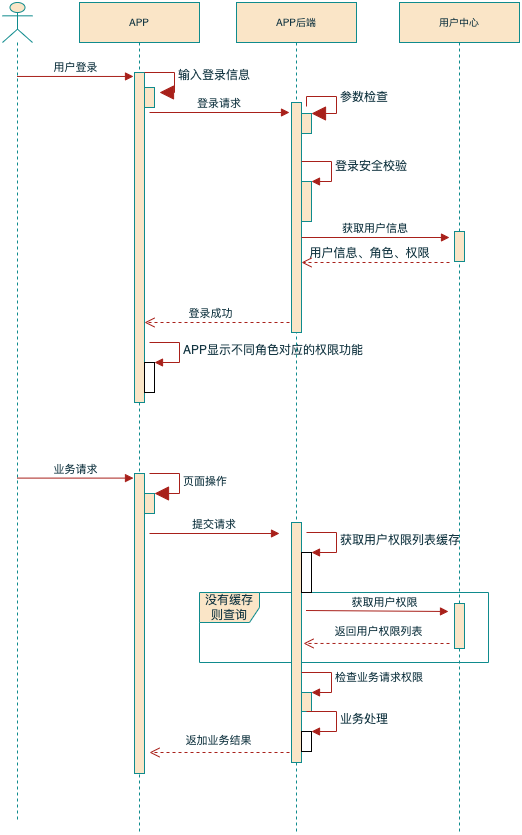


图 4-1-2-2 权限检查流程

* 权限从前端和后端都会进行校验
* 权限列表需要使用缓存来提高性能
* 后端权限检查通过注解统一拦截处理

#### 权限粒度

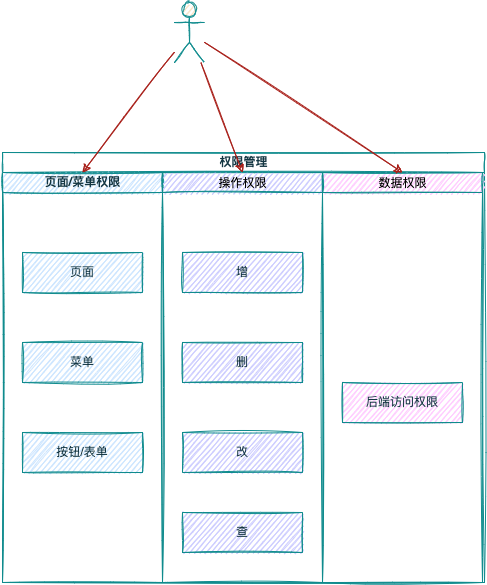


图 4-1-2-3 权限粒度

页面/菜单权限

对于没有权限操作的用户，直接隐藏对应的页面入口或菜单选项，实现所见即所得。这种方法简单快捷直接。

操作权限

操作权限通常是指对同一组数据，不同的用户是否可以增删改查。对某些用户来说是只读浏览数据，对某些用户来说是可编辑的数据。

数据权限

对于安全需求高的权限管理，仅从前端限制隐藏菜单，隐藏编辑按钮是不够的，还需要在数据接口上做限制。如果用户试图通过非法手段编辑不属于自己权限下的数据，服务器端会识别、记录并限制访问。

#### 角色扩展

默认角色

自动获得基础角色，避免维护的工作量

临时角色

需要协作时的临时角色，此类账号默认有固定失效时间，避免临时账号因为流程不完善，遗忘在系统中，引起安全隐患。

# 系统可扩展设计

## 数据分表

基因法的思路是我们将 username 或者 mobile 融入到 UID 中。具体做法如下：

* 用户注册时，根据用户的手机号码，利用函数生成 N bit 的基因 mobile\_gen，使得 mobile\_gen=f(mobile);
* 生成 M bit 全局唯一的 id，作为用户标识；
* 拼接 M 和 N，作为 UID 赋给用户；
* 根据 N bit 来取余来插入到特定数据库；
* 查找用户数据的时候，将用户 UID 的后 N bit 取余来落到最终的库中。