智慧社区流动人口服务管理平台的设计与实现

刘训利，刘浩，彭恩
广东东软学院，广东 佛山 528225

DOI: 10.61369/TACS.2026080012

摘　　要： 随着我国城市化进程的不断深入，人口跨区域流动规模持续扩大，流动人口服务管理已成为基层社区治理的重要课题。本文针对传统流动人口管理存在的工作量大、效率低下、信息滞后等突出问题，设计并实现了一套智慧社区流动人口服务管理平台。系统采用 " 管理端 + 居民端 + 后端服务 " 三位一体的 SaaS 架构，基于 Vue 3和 HarmonyOS ArkUI 构建双端应用，利用 Spring Boot 3.x 搭建后端服务，集成 MySQL 数据库实现数据持久化。平台实现了流动人口信息管理、核查任务管理、安全隐患排查、政策法规管理、问卷调查管理等核心功能，支持 RBAC 精细化权限控制和智能任务调度，有效提升了社区流动人口管理的数字化、智能化水平。系统测试结果表明，各模块功能完善，性能稳定，具有较好的实际应用价值和推广前景。

关 键 词： 智慧社区；流动人口管理；Harmonyos；Spring Boot；Saas 架构

Design and Implementation of a Smart Community Floating Population Service Management Platform

Liu Xunli, Liu Hao, Peng En
Neusoft Institute Guangdong, Foshan, Guangdong 528225

Abstract: With the continuous advancement of China's urbanization process, the scale of cross-regional population mobility continues to expand, making the service and management of the floating population an important issue in grassroots community governance. Addressing prominent issues in traditional floating population management, such as heavy workload, low efficiency, and information lag, this paper designs and implements a smart community floating population service management platform. The system adopts a tripartite SaaS architecture comprising an "Admin End + Resident End + Backend Service." It builds dual-end applications based on Vue 3 and HarmonyOS ArkUI, utilizes Spring Boot 3.x to construct the backend services, and integrates a MySQL database to achieve data persistence. The platform implements core functions including floating population information management, verification task management, safety hazard investigation, policy and regulation management, and questionnaire survey management. It supports fine-grained Role-Based Access Control (RBAC) and intelligent task scheduling, effectively enhancing the digitalization and intelligence level of community floating population management. System testing results indicate that the modules are fully functional and perform stably, demonstrating significant practical application value and broad prospects for promotion.

Keywords: smart community; floating population management; HarmonyOS; Spring Boot; SaaS architecture

引言

随着我国城市化进程的不断深入，人口跨区域流动规模持续扩大，外来务工人员、租住人员、灵活就业人员、陪读人员等流动人口已成为城市社区人口结构的重要组成部分。流动人口为城市建设、产业发展和社会活力提升作出了重要贡献，但其居住变动频繁、就业形态多样、服务需求复杂，给基层社区治理带来了显著压力，亟需借助数字化手段提升管理效能[1]。当前许多社区在流动人员管理方面仍主要依赖人工上门摸排、纸质登记和线下台账管理，存在工作量大、效率低下、信息滞后、数据重复录入、服务对接不精准、协同治理困难等突出问题。在公共服务供给方面，社区面向流动人员提供的就业咨询、子女教育、社会保障、医疗卫生等服务往往缺乏精准触达渠道，“有服务但不知道”“有需求但找不到”的情况较为普遍。

在智慧城市建设的大背景下，如何利用信息化手段提升流动人口服务管理水平，已成为基层治理体系现代化能力的重要课题。本课

项目信息：2026年大学生创新创业项目《云栖慧治—面向社区流动人口的轻量化 SaaS 平台构建与协同服务机制研究》(项目编号：X202612574064)。

Copyright © This Work is Licensed under A Commons Attribution-Non Commercial 4.0 International License.

题设计并实现了一套完整的智慧社区流动人口服务管理平台。平台采用前后端分离架构，实现管理端 Web 与居民端 HarmonyOS App 的双端协同。同时引入 AI 辅助政策咨询服务，提升公共服务智能化水平。通过以上技术手段的融合，构建全链路数字化闭环，实现社区治理全流程线上化、数据化、可追溯。

一、系统总体设计

（一）设计目标

本系统旨在将传统以人工为主的流动人员管理模式升级为以数据驱动、双端协同、智能辅助为核心的现代化治理范式。首先，建立统一的流动人口信息管理中枢。针对传统模式下人口数据分散于各类孤立台账中导致的查询困难与统计滞后问题，系统实现了流动人口从首次登记、日常信息更新到离境注销的全生命周期数字化管理。其次，构建管理端与居民端双轨协同的服务模式。管理端面向社区工作人员与网格员，提供人口采集、房屋巡查、服务审批等全功能操作工作台。居民端则面向流动人口群体，支持个人信息自助申报、社区服务在线预约及办事进度实时追踪。第三，实现核查任务的全流程线上管控。系统内置完整的任务流转引擎，支持管理员在线创建专项核查任务并按规则或人工判断精准分派至对应网格员的责任辖区。第四，提供政策法规的精准推送与智能问答服务。平台建立了结构化的政策知识库并集成基于自然语言处理技术的智能问答引擎。第五，支撑安全隐患的闭环管理。针对消防通道占用、违规群租等常见隐患场景，系统支持网格员在巡查过程中一键上传隐患描述与现场照片证据，系统自动根据类型和严重程度归入处置队列并触发分级预警通知。第六，赋能多维数据分析与科学决策。平台内置了涵盖总量趋势预测、区域密度热力图生成、人群结构占比分析等多个维度的数据分析模型，结果以可视化图表呈现并支持一键导出报表。

（二）设计架构

本平台遵循软件工程中经典的分层设计原则，采用 B/S 与 C/S 相结合的四层架构模式进行整体设计。如图 1 所示，系统自上而下依次划分为表示层、业务逻辑层、数据访问层与基础设施层四个核心层次，同时在架构两侧预留了信息资源与外部系统的标准化对接通道，从而在保障系统内部高内聚低耦合的同时，兼顾与智慧城市生态中其它业务子系统的互联互通能力。

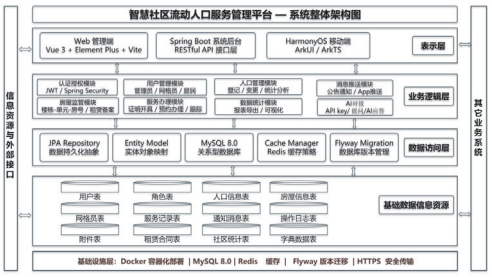


图 1 系统架构图

表示层作为用户与系统之间的唯一交互界面，采用了 Web 端与移动端双端并行且数据统一的设计策略。其中，Web 管理端基于 Vue 3 前端框架结合 Element Plus 组件库构建，采用 Vite 作为构建工具以提升开发效率与运行时性能 [2]。该端主要面向社区管理员与网格员两类管理角色，提供人口信息采集录入、房屋租赁备案审核、社区服务在线审批、多维度数据统计等全功能管理操作界面。移动端采用华为 HarmonyOS NEXT 操作系统原生技术栈开发，以 ArkUI 声明式 UI 框架配合 ArkTS 编程语言实现跨设备的响应式布局 [3]。该端面向社区居民群体，提供个人信息自助查询、社区服务预约申请、通知消息实时接收等便民服务功能。

业务逻辑层是整个平台的功能实现载体，该层基于 Spring Boot 3.x 企业级应用框架构建，按照单一职责原则将全部业务能力解耦为认证授权、用户管理、人口管理、房屋监管、服务办理、数据统计与消息推送七个相对独立的业务模块，各模块内部严格遵循 Controller 接收请求、Service 编排业务逻辑、Repository 执行数据访问的标准三层调用范式。

数据访问层为上层业务代码屏蔽底层数据存储的技术差异，提供统一且高效的数据存取抽象。在该层的具体实现中，Spring Data JPA 作为持久化框架的基础设施，通过 Repository 接口编程范式将常见的增删改查、分页排序、动态条件查询等数据库操作封装为声明式方法签名，使开发者得以专注于业务语义表达而无须编写冗余的 SQL 语句。基础数据信息资源层以实体 – 关系模型的形态定义了平台所需的全部数据资产。经过充分的领域分析与范式化设计过程，最终确定了核心业务表作为系统的数据基石。

基础设施层构成了系统运行的物理与技术底座。在生产部署环节，Docker 容器化技术被选为应用的交付与运行载体，它将应用程序代码及其全部依赖环境打包为标准化的镜像制品，借助 Docker Compose 编排工具可实现一键部署包括应用服务、数据库实例、缓存服务在内的全套运行环境。

二、系统功能

本平台的功能体系围绕流动人口社区治理的核心业务需求进行设计开发，按照服务对象与职责分工的差异可划分为以下若干相互协作的功能模块。系统模块功能图如图 2 所示。



图 2 系统功能模块图

（一）人口信息管理

人口信息管理是整个平台的基石，承载着辖区流动人口基础数据的采集、维护与运用等核心职责。在信息登记环节，网格员既可通过移动端在现场走访时即时录入新迁入人员的姓名、身份证号、户籍地址、联系方式等必填字段并上传证件照片作为核验

 Copyright © This Work is Licensed under A Commons Attribution-Non Commercial 4.0 International License.

依据，也可通过管理端批量导入由上级部门下发的底册数据。已入库的人员信息支持按条件组合检索与逐字段编辑修改，且每一次编辑操作均被系统自动记录至操作日志中以确保数据变更的可审计性。针对日常管理中高频使用的统计分析需求，模块内置了按时间段、户籍来源、居住区域、年龄结构等多维度灵活配置的统计引擎，分析结果以表格与可视化图表两种形式输出并可一键导出为 Excel 报表文件供汇报使用。

（二）房屋与居住地监管

该功能模块围绕“人－房关联”的治理理念构建了一套楼栋－单元－房号三级地理地址编码管理体系。管理员首先在系统中完成辖区内全部建筑物的层级结构建档工作，为每一处独立住所分配全局唯一的房屋编码标识。此后在进行人口信息登记时，操作员须指定该人员当前的实际居住房屋编号，从而建立起从房屋到人的反向索引关系，便于快速查询某处住所内的全部在住人员清单或识别群租超员等违规情形。模块同时支持租赁合同的备案登记与到期预警提醒，当某份租赁合同临近约定期限时系统将自动向相关网格员推送续签核查任务提示，辅助基层落实以房管人的管控策略。

（三）核查任务管理

核查任务管理功能为社区管理者提供了一套将线下巡查工作转化为线上可控流程的数字化工具。管理员可根据阶段性工作重点（如节前安全大检查、 periodic 人口清查摸排等）在管理端创建专项核查任务，设定任务的覆盖范围、完成时限与具体核查要点后将其发布至系统。任务发布后，系统依据各网格员的责任区域划分规则自动完成初步分派，管理员亦可在此基础之上进行人工调整以平衡工作量。被指派的网格员将在移动端收到待办事项通知，点击即可查看任务详情并在执行过程中逐一填写每项核查要点的检查结果，必要时还可附加拍摄现场照片作为佐证材料提交。

（四）社区服务办理

社区服务办理功能面向流动人口群体在日常生产生活中所涉及的各类证明开具、事项预约及进度咨询需求提供在线化服务通道。居民可通过移动端浏览平台所支持的办事事项清单（如居住证明开具、政审协助函申请等），选择所需事项后在线填报表单并上传必要的身份凭证附件。提交后的申请将由系统自动路由至对应的管理员审批队列中，审批人在管理端对材料的合规性与完整性进行审核后作出准予受理或退回补正的处理决定，处理结果将以站内消息与短信双通道形式同步通知申请人。

（五）政策资讯与智能问答

管理员可在后台编辑发布各类通知公告与政策解读文章并通过标签定向规则将内容精准推送至特定人群，确保信息投递的有效性与针对性。在智能问答方面，平台集成了基于知识库匹配的自然语言问答引擎，当居民在移动端的对话界面输入关于居住证续期条件、子女就学政策等方面的提问时，系统自动对问题文本进行语义解析后在预置的政策知识库中检索最相关的条目并以简明语言返回解答答案。

（六）安全隐患排查处置

安全隐患排查功能为网格员的日常巡检工作配备了标准化的上报与跟踪工具。当网格员在巡查中发现诸如消防通道堵塞、电线私拉乱接、疑似非法群租等安全隐患时，可通过移动端的一键上报界面迅速录入隐患类型、所在位置坐标、现场文字说明与取证照片。系统接收上报后将根据预设的风险等级评估规则自动确定该条隐患记录的紧急程度并触发相应级别的预警通知——高风险隐患即刻推送至社区负责人手机终端督促其优先处置，中低风险隐患则纳入常规整改队列等待统一排期。后续的整改责任认定、限期整改指令下发、复查验收确认直至最终销号的全过程均在系统中留有结构化的流转记录，管理者随时可调阅任意一条隐患的完整处置轨迹。

（七）数据统计与决策支持

数据统计功能位于平台能力金字塔的顶端，旨在将沉淀于各业务模块中的原始数据转化为可供管理层直接消费的分析洞察。平台内置了多套预置的统计报表模板，涵盖辖区流动人口总量时序变化趋势、各小区人口密度热力分布、年龄段与性别比例结构、户籍来源省份排名、近三十日新增与注销净流入量等常用分析场景。每套报表均支持自定义筛选条件的二次配置与时间范围的自由选取，生成结果除以数据表格呈现外还配套了柱状图、折线图、饼图等多种图表可视化组件以便直观阅读。

（八）用户与权限管理

系统采用 RBAC 模型进行权限控制，每个角色关联一组权限集合，支持功能级别的精细化控制。系统设计的用户分为管理员、网格员、居民三种角色。管理员具有最高权限，负责系统配置和数据管理。网格员负责所属网格区域内的人另采集、房屋检查和服务引导。居民通过移动端查询个人信息、申请服务和接收消息通知。

三、系统实现

（一）开发环境及工具链

平台后端采用 Spring Boot 3.x 开发，前端 Web 端基于 Vue 3 技术栈，移动端使用华为 ArkTS + ArkUI 框架在 DevEco Studio 中完成编码。版本控制采用 Gitee 分支管理策略协调多人协作。构建环节由 Maven 与 Vite 分别负责后端 JAR 打包与前端的资源压缩优化。数据库 Schema 迭代通过 Flyway 插件自动执行增量迁移脚本管理，最终以 Docker Compose 实现应用服务、MySQL 与 Redis 的一键容器化部署。

（二）后端核心模块实现

1. 认证授权机制

登录时 Authentication Controller 将凭证交由 Authentication Manager 完成核验，通过后从数据库查询该用户角色标识与权限集合并封装为 JWT Payload 签发令牌返回客户端，有效期两小时并支持滑动刷新。后续每次请求客户端须将令牌置于请求头中传递，自定义的 Jwt Authentication Filter 在过滤器链中拦截入站请求完成令牌校验后将用户身份写入 Security Context，验证失败则直接返回 401 状态码。伪代码如下所示。

Copyright © This Work is Licensed under A Commons Attribution-Non Commercial 4.0 International License.

登录与令牌签发伪代码

```
AuthResult ← AuthenticationManager.authenticate(C)
IF AuthResult IS NOT SUCCESS THEN
    RETURN AuthenticationException
END IF
User ← Database.queryUser(AuthResult)
Payload ← { User.id, User.roles, User.permissions, Exp = Now + 2h }
T ← JWT.sign(Payload)
RETURN T
```

JWT 请求拦截与上下文注入伪代码

```
Token ← R.getHeader(“Authorization”)
IF Token IS NULL THEN
    PassToNextFilter(R)
    RETURN
END IF
TRY
    Claims ← JWT.validate(Token)
    Auth ← BuildAuthentication(Claims)
   SecurityContext.set(Auth)
CATCH JwtException
    RETURN HTTP.401_UNAUTHORIZED
END TRY
PassToNextFilter(R)
```

2. 人口信息管理

数据持久层基于 Spring Data JPA 的 Repository 接口代理机制，PopulationInfo Repository 继承 JpaRepository 后框架自动生成标准 CRUD 方法。多条件组合检索通过遵循 Spring Data 命名规范的衍生查询方法（如按姓名模糊匹配且按创建时间降序排列）由框架自动翻译为 SQL 执行，复杂聚合统计则通过 @Query 注解手写 JPQL 精确控制查询计划。Service 层以 @Transactional 标注写操作方法保证事务一致性——如人员注销时需在同一事务内同步更新状态字段、追加审计日志及递减关联房屋的在住人数计数器，任一异常触发回滚防止数据中间态不一致。业务实现逻辑的伪代码如下所示。

人口信息管理实现逻辑伪代码

```
// 1. 基础查询：框架代理自动生成 CRUD
IF Req.type == BASIC_QUERY THEN
    RETURN JpaRepository.findByNameAndOrderByCreateTime(...)
END IF
// 2. 复杂查询：通过注解自定义 JPQL
IF Req.type == COMPLEX_AGGREGATION THEN
    RETURN Repository.executeCustomJPQL(Req.params)
END IF
// 3. 写操作：声明式事务控制
@Transactional
FUNCTION executeWriteOperation(Req):
    TRY
        Entity.status ← CANCELLED       // 更新状态字段
        AuditLog.append(Entity.id)      // 追加审计日志
        House.residentCount ← DECREMENT  // 递减关联计数器
    CATCH Exception
        ROLLBACK TRANSACTION            // 异常触发回滚
        RETURN DataInconsistencyError
    END TRY
    RETURN Success
END FUNCTION
```

3. 核查任务流转引擎

核查任务的闭环管理依托一套轻量级状态机模型：每条任务持有 status 字段标识当前生命周期节点，合法迁移路径预定义为"待分派→已分派→执行中→待审核→已完成→已归档"的有向图。TaskStateMachine 工具类封装全部转移规则并在变更前校验目标状态合法性，每次变迁同时将旧值、新值、操作人及时间戳序列化写入附表形成完整审计轨迹。任务分派默认依据网格员责任区域与目标地点的空间匹配关系自动指派，管理员亦可手动覆写调整。移动端通过 /api/tasks/my-pending 接口接收待办、通过 multipart 格式的 submit-result 接口提交结构化核查数据与现场照片附件。

核查任务流转引擎实现伪代码

```
// 1. 状态机核心：合法性校验与审计轨迹
FUNCTION transition(T_id, S_target, U):
    S_current ← Task.get(T_id).status
    IF (S_current, S_target) NOT IN ValidTransitions THEN
        RETURN IllegalStateError
    END IF
    AuditLog.insert(S_current, S_target, U, Now()) // 写入审计附表
    Task.updateStatus(T_id, S_target)
   RETURN Success
END FUNCTION
// 2. 任务分派：空间匹配与人工覆写
FUNCTION assign(T_id, GridWorker):
    IF GridWorker IS NULL THEN
        GridWorker ← SpatialMatch(T_id.location)  // 默认空间匹配
    END IF
    transition(T_id, ASSIGNED, Admin)
    RETURN GridWorker
END FUNCTION
// 3. 移动端闭环交互：待办拉取与结果提交
FUNCTION submitResult(T_id, StructuredData, Photos):
    Validate(StructuredData, Photos)  // 校验结构化数据与附件
    transition(T_id, PENDING_REVIEW, Worker) // 状态流转至待审核
    RETURN Success
END FUNCTION
```

（三）前端界面实现

1.Web 管理端

Web 端基于 Vue 3 Composition API 构建 SPA 单页应用，路由集中定义于 router/index.ts 并为每条路由附加认证与角色白名单元属性。全局前置导航守卫 beforeEach() 根据当前用户角色动态判定跳转权限，未满足条件则重定向至对应提示页。UI 采用 Element Plus 组件库提供表格、表单、对话框等企业级控件。网络层基于 Axios 二次封装统一实例——请求拦截器自动注入 LocalStorage 中的 JWT 令牌，响应拦截器统一处理 401 跳转登录与全局错误消息提示。主界面采用左侧垂直菜单加右侧内容区双栏布局，菜单数据来源于后端根据角色动态返回的菜单树接口，不同角色呈现差异化导航选项集合。

 Copyright © This Work is Licensed under A Commons Attribution-Non Commercial 4.0 International License.

前端 SPA 架构与动态权限控制伪代码

```
// 1. 全局路由守卫：动态权限判定
FUNCTION beforeEach(R):
    Role ← CurrentUser.getRole()
    Whitelist ← R.meta.requiredRoles
    IF Role NOT IN Whitelist THEN
        RETURN Redirect("ForbiddenPage")
    END IF
    RETURN Next()
END FUNCTION
// 2. 网络层拦截器：统一鉴权与异常处理
FUNCTION onRequest(H):
    H.headers["Authorization"] ← "Bearer " + Storage.get(JWT)
    RETURN H
END FUNCTION
FUNCTION onResponse(Res):
    IF Res.status == 401 THEN
        Storage.clear()
        RETURN Redirect("LoginPage")
    END IF
    RETURN Res
END FUNCTION
// 3. 动态菜单渲染：基于角色驱动视图
FUNCTION renderLayout():
    MenuTree ← API.getMenusByRole(CurrentUser.role)
    RETURN Layout(LeftMenu=MenuTree,
RightContent=RouterView)
END FUNCTION
```

2.HarmonyOS 移动端

移动端采用 ArkUI 声明式 UI 框架开发，以 @Entry 装饰器标注页面入口，build() 方法内以链式调用描述界面层级。导航结构由 Navigation 容器维护页面栈，NavRouter 注册 URI 到页面构造函数的映射关系。首页以纵向列表展示个人信息、办事预约、通知消息、智能问答四类快捷入口卡片，点击即触发路由跳转。网络通信基于 @ohos.net.http 模块封装 HTTP 工具类并与 Web 端共享同一套 JWT 认证机制。

鸿蒙端实现伪代码

```
// 1. 声明式 UI 渲染：链式构建与路由跳转
@Entry FUNCTION build():
  Column(
    QuickEntryCard("个人信息", E.onClick → NavRouter.push("/
user")),
    QuickEntryCard("办事预约", E.onClick → NavRouter.push("/
booking")),
    QuickEntryCard("通知消息", E.onClick → NavRouter.push("/
notify")),
    QuickEntryCard("智能问答", E.onClick → NavRouter.push("/
ai"))
  )
END FUNCTION
// 2. 导航栈管理：URI 映射与页面调度
FUNCTION initNavigation():
    NavRouter.register("/user", UserPage)
    NavRouter.register("/booking", BookingPage)
    Navigation.bindStack(NavRouter.stack)
END FUNCTION
// 3. 网络通信封装：统一 JWT 鉴权
FUNCTION httpRequest(H):
    H.headers["Authorization"] ←" Bearer "+Storage.get(JWT) /
    Res ← @ohos.net.http.send(H)
    RETURN Res
END FUNCTION
```

四、系统测试

（一）测试环境与方法

测试硬件环境：Intel Core i7-12700H / 16GB RAM / 1T SSD

服务器环境：Docker Desktop 24.0 (Windows)

数据库：MySQL 8.0.35 Container (2CPU / 4GB Memory)

测试测览器：Google Chrome 124.0 / HarmonyOS Simulator

测试工具：Postman 11.0 (API测试) / JMeter 5.6 (性能测试)

本系统采用分层测试策略，包括单元测试、集成测试、系统测试和接受测试四个阶段。后端单元测试采用 JUnit 5 + Mockito 框架，其中 Service 层单元测试覆盖率达到85%；前端采用 Vitest 进行组件单元测试。

（二）功能测试

功能测试主要验证系统各模块是否满足需求规范中定义的功能点。以下选取部分典型功能测试用例，如下表所示。

表 功能测试用例（部分）

用例 ID	测试项目	测试步骤	预期结果	测试结果
TC-01	用户登录	输入正确用户名密码	登录成功，跳转首页	通过
TC-02	人员增加	填写完整人员信息	保存成功，列表显示新记录	通过
TC-03	人员查询	按姓名模糊查询	返回匹配的人另列表	通过
TC-04	房屋添加	填写房屋基本信息	房源信息保存成功	通过
TC-05	租房签约	提交完整租房合同	合同状态为已签约	通过
TC-06	服务申请	提交居住证申请	申请提交成功，审核中	通过
TC-07	数据统计	查看人口趋势图	正确显示月度趋势图	通过
TC-08	权限控制	居民访问管理功能	跳转403页面	通过

测试结果表明，系统全部功能测试用例均通过，各模块功能运行正常，业务流程符合预期设计，权限控制有效防止了超权访问。接受测试中，3名网格员反馈系统操作直观简单，基本功能均能独立完成，平均学习成本为15分钟左右，符合实际应用要求。

（三）性能测试

性能测试采用 JMeter 5.6工具，对系统核心 API 接口进行压力测试。测试场景设置如下：线程数100，Ramp-Up Period 10秒，持续时间60秒，目标并发数为80用户 /s，测试系统吞吐量指标（TPS）和错误率。

人员列表接口 (GET /api/v1/population)：平均响应时间85ms，TPS 456，错误率0%

人员新增接口 (POST /api/v1/population)：平均响应时间120ms，TPS 312，错误率0%

数据统计接口 (GET /api/v1/statistics/overview)：平均响应

Copyright © This Work is Licensed under A Commons Attribution-Non Commercial 4.0 International License.

时间200ms，TPS 498，错误率0%

测试结果表明，在50用户 /s 的并发负荷下，系统各接口的平均响应时间均在200ms 以内，系统吞吐量指标符合需求规范中的性能指标，未出现内存泄漏和 CPU 过载等问题。通过 Docker 统计的资源限制 (app 容器2CPU/4GB)，系统在资源受限时仍能稳定运行，验证了架构的健壮性。

五、工作总结

本文针对当前社区流动人员管理中存在的信息孤岛、服务渠道不畅通、管理效率低下等问题，设计并实现了一套智慧社区流动人口服务管理平台。完成了系统的需求调研与分析，明确了多角色、多端的功能需求和非功能性指标，为系统设计奠定了基础。设计了基于 B/S 架构的多端联动系统方案，包括 Spring Boot 3.x 后端、Vue 3 Web 端和 HarmonyOS 移动端，实现了数据层、业务层、表示层的分离设计。完成了后端核心模块的开发实现，包括用户身份认证、人口管理、房屋监管、服务办理、数据统计和消息推送，通过 RBAC 模型实现了精细化的权限控制。实现了 Web 管理前端和 HarmonyOS 移动客户端，分别服务于网格员 / 管理员和居民用户，完善了多端联动的服务体系。最后我们对系统进行了功能测试、性能测试和安全测试，测试结果表明系统功能完善、性能稳定、安全可靠，满足设计需求。虽然本系统已实现了预定的设计目标，但仍存在以下不足之处，将在后续研究中进一步完善。

（1）智能化功能拓展：引入机器学习和自然语言处理技术，实现人员数据的智能填写、异常行为自动检测和更多的服务需求智能推荐等功能。

（2）移动端功能强化：充分利用 HarmonyOS 的分布式能力，实现离线数据同步和本地数据库快照，提升移动端的离线使用能力。

参考文献

[1] 胡龙，李妙 . 租住新业态智能化管理系统及其在房屋治安监管工作中的应用——以金华永康流动人口服务管理为例 [J]. 警察技术 ,2025,(6):18-21.

[2] 花树峰 .Spring Boot+MVC+Vue3项目全流程开发指南：从需求分析到上线部署 [M]. 机械工业出版社 :2024.

[3] 夏彬 . 基于 ArkUI 框架的 OpenHarmony EasyUI 组件库设计与实现 [D]. 北京林业大学 ,2024.DOI:10.26949/d.cnki.gblyu.2024.001003.

 Copyright © This Work is Licensed under A Commons Attribution-Non Commercial 4.0 International License.