**(颐养社区系统)项目管理文档**

1. **项目综述**
2. **项目简要介绍**

自二十一世纪以来，我国老龄化人口逐年上升，如今，现在越来越多的年轻人喜欢到大城市就业，这已经导致了一个日益严重的社会现象：子女漂泊他乡，老人独守空巢。独居老人不仅要独自面对生活中的所有问题，还要承受无人陪伴的心情所带来的孤独与无助。目前最适合我国基本国情的养老模式是社区养老模式，它不仅能为老年人在生活上提供帮助，还能迎合老人想要待在熟悉环境的心愿。然而，在人口老龄化的趋势下，相对应的老年人口的数据信息量高速膨胀，传统的数据采集方式日益显现出其弊端，无法满足如今的需求。因为开发一款高效，快捷，便于统计管理的颐养社区系统就显得很有必要了。本文通过对养老社区办公流程的分析研究，设计出一款流程简洁，响应迅速的颐养社区系统，旨在提高人员信息采集的效率，从而腾出更多的时间与空间陪伴老人，进一步改善老人的生活质量，推动养老社区的进步。

本文首先对颐养社区系统的研究背景和国内外发展现状进行了调研。其次，介绍了颐养社区的整体结构、功能需求和非功能需求。再次，对颐养社区系统中的用户管理、楼宇管理、问卷管理、病患管理进行了详细的描述。最后，提出的了现阶段颐养社区系统中还需改进的地方以及未来的发展趋势。

本文主要采用C/S（Client-Server）结构和Java语言完成系统软件部分的开发与设计。主要使用目前广泛应用的SpringBoot框架搭建系统的后端，结合MySQL数据库存储用户的信息，主要使用JavaFx框架构建系统的前端。

通过编写测试用例对系统进行功能测试和性能测试。经过测试，系统达到了预期效果，具有较好的实用性。

1. **项目面向的对象**

用户群体：颐养社区中的员工；颐养社区中的老板

特征：都是颐养社区中的人

需求：高效的日常办公

业务现状：管理员工和老人

正待解决的问题：传统方式工作效率太低

职责权限：普通员工 —> 用户权限、老板 —> 管理员权限

1. **项目特点**

颐养社区系统利用信息化技术打造一套面向颐养社区员工的信息管理化平台，该系统提供实时、快捷、高效、低成本的智能化养老看护服务。帮助颐养社区中的员工大幅提升管理效率，节约时间、人力成本，全方位减轻养老的社会负担。

颐养社区系统采用SpringBoot+Mybatis+JavaFX技术进行开发，SpringBoot之所以能增加如此多的独有特点，是因为它在Spring的基础之上扩展了非常多的相关代码。导致如果要想使用SpringBoot必须依赖大量的子组件，这样反倒不会起到简化的效果。所以SpringBoot为开发者准备了丰富的环境用于简化依赖。Springboot最核心的特点，就是自动配置。它为大多数开发场景准备对应的配置逻辑。在使用SpringBoot开发时，可以不考虑如何配置这些技术、环境，只需要按照SpringBoot提供的逻辑准备你的技术、环境。MyBatis 是一款优秀的持久层框架，它支持自定义 SQL、存储过程以及高级映射。MyBatis 免除了几乎所有的 JDBC 代码以及设置参数和获取结果集的工作。MyBatis 可以通过简单的 XML 或注解来配置和映射原始类型、接口和 Java POJO（Plain Old Java Objects，普通老式 Java 对象）为数据库中的记录。MyBatis实现了接口绑定，使用更加方便。JavaFX是一种新型桌面应用开发平台，带有强大的桌面组件库,可独立于浏览器运行，需要安装独立的运行环境、便捷访问本地资源，属于桌面型应用开发，而非网页开发，目前逐步可以容纳CSS、JS等，甚至HTML、FXML, 同时支持Java实现跨平台。

该系统解决的主要业务需求包括：医生管理、护士管理、护工管理、楼宇管理、病患管理、房间管理、用户管理、问卷管理等。颐养社区系统实现了服务管理信息化、智能化、对社区老人入住进行了全流程管理、服务过程记录、为老人的安全保驾护航。具有重要的商业价值以及出色的竞争优势。

颐养社区系统特色：

（1）操作简单，界面友好：满足客户已有的操作习惯；

（2）开放性和扩展性：完备的二次开发能力，以适应该系统升级和功能扩充维护，便于系统容量的增加、支撑功能的增强以及和其它系统的兼容；

（3）使用可靠：系统核心部件稳定可靠，全面提高了系统使用的方便性、可靠性和稳定性等；

（4）适用范围广，可移植性强：适用于养老院、居家养老、福利院等所有产品系列，而且可以轻松移植到其它产品系列；

（5）强大的统计分析：可以方便统计服务过程中产生的各种数据，进行各种报表分析，为管理者提供便利，为决策者提供依据。

（6）系统提供标准的数据接口，且有严格的数据权限控制机制，能够保证业务数据安全。

1. **项目中的角色**

|  |  |
| --- | --- |
| 角色名称 | 职责描述 |
| 项目经理 | 总揽全局，设计系统，规划开发 |
| 前端工程师 | 开发系统的用户页面 |
| 后端工程师 | 开发系统的数据接口 |
| 数据库工程师 | 制定系统的数据表 |
| 测试工程师 | 测试系统的功能 |

1. **运行环境**

服务端要求：

操作系统：Windows10系统

软件要求：系统安装了Java JDK1.8

硬件要求：CPU：Intel Core i5-7300 2.5GHZ，内存8GB

客户端要求：

操作系统：Windows7及以上

软件要求：系统安装了Java JDK1.8

硬件要求：64位处理器，核心至少两个，频率1GHZ以上， 内存至少4GB

1. **技术和实现**

JDK版本：Jdk 1.8

后端技术：SpringBooot 2.6框架，Mybatis 3.5.9框架，

前端技术：JavaFX 8

数据库：MySQL 8.0

Web容器：Apache Tomcat 9.0

项目管理工具：Maven 3.6.0

版本控制工具：Git

开发工具：IDEA

开发标准：MVC架构

数据转换格式标准：JSON

1. **需求**
2. **需求背景**

（1）项目启动原因

自二十一世纪以来，我国老龄化人口逐年上升，如今，现在越来越多的年轻人喜欢到大城市就业，这已经导致了一个日益严重的社会现象：子女漂泊他乡，老人独守空巢。独居老人不仅要独自面对生活中的所有问题，还要承受无人陪伴的心情所带来的孤独与无助。目前最适合我国基本国情的养老模式是社区养老模式，它不仅能为老年人在生活上提供帮助，还能迎合老人想要待在熟悉环境的心愿。然而，在人口老龄化的趋势下，相对应的老年人口的数据信息量高速膨胀，传统的数据采集方式日益显现出其弊端，无法满足如今的需求。因为开发一款高效，快捷，便于统计管理的颐养社区系统就显得很有必要了。本文通过对养老社区办公流程的分析研究，设计出一款流程简洁，响应迅速的颐养社区系统，旨在提高人员信息采集的效率，从而腾出更多的时间与空间陪伴老人，进一步改善老人的生活质量，推动养老社区的进步。

1. 竞品分析

目前市场上有关于社区老人管理的系统较少，或其功能不够全面，大多数只能对社区人口、楼栋、社区居委会进行一系列的管理，而并非将侧重点注重于改善老人的生活质量、推动养老社区的发展这一方向。因此，本项目在全国老龄化的严峻形势之下，能够更好的适应国家和社会的现实情况，有着突出的优势和竞争力。

1. **项目目标**
2. 社区老人的信息实现90%以上的覆盖面。
3. 统一管理老人住院信息以及该系统分类管理人员的基本信息。
4. 系统能够很好的满足老人的各项需求，最大覆盖老人的各类需求。
5. 系统能够基本满足颐养社区管理人员的日常工作需求。
6. 系统能够基本满足小、中和大型社区的不同数据量的处理，达到流畅的工作效果。
7. 系统调查问卷可以更好的了解用户对该系统的满意程度，获取对该系统的建议，便于更好地完善该系统功能。
8. **修订记录**

|  |  |  |  |
| --- | --- | --- | --- |
| 修订日期 | 修订人 | 修订说明 | 修订原因 |
| 2022年4月1日 | 樊庆元 | 修改文档错别字 | 需求内容存在错别字 |
| 2022年4月2日 | 陈洋 | 补充竞品分析内容 | 竞品分析方面内容的编写未完善 |
| 2022年4月2日 | 王小同 | 补充项目目标内容 | 项目目标内容的编写未完善 |
| 2022年4月3日 | 王小同 | 补充功能列表到29条 | 功能有所扩充，修改文档 |
| 2022年4月4日 | 陈洋 | 增加逻辑展示图中的思维导图 | 完善逻辑展示的内容 |

1. **功能列表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 功能模块 | 主要功能点 | 优先级 | 功能描述 |
| 1 | 老人管理 | 新增老人 | P1 | 往系统中添加老人 |
| 2 | 修改老人 | P1 | 修改已经入住的老人信息 |
| 3 | 删除老人 | P2 | 删除已经入住的老人信息 |
| 4 | 搜索老人 | P0 | 万能搜索，便于查找老人 |
| 5 | 浏览老人 | P1 | 浏览所有老人信息，统筹管理 |
| 6 | 员工管理 | 录入员工 | P1 | 往系统中录入员工 |
| 7 | 移除员工 | P2 | 从系统中移除员工 |
| 8 | 搜索员工 | P0 | 万能搜索，便于管理员查找员工 |
| 9 | 分类浏览员工 | P1 | 浏览系统中的所有员工 |
| 10 | 更新员工信息 | P1 | 更新员工的个人信息 |
| 11 | 楼宇管理 | 添加楼栋 | P1 | 添加社区中的楼栋信息 |
| 12 | 删除楼栋 | P2 | 移除社区中的楼栋信息 |
| 13 | 添加房间 | P1 | 为楼栋添加房间信息 |
| 14 | 移除房间 | P2 | 为楼栋移除房间信息 |
| 15 | 添加床位 | P1 | 为房间添加床位信息 |
| 16 | 移除床位 | P2 | 为房间移除床位信息 |
| 17 | 分配床位 | P0 | 为床位分配老人 |
| 18 | 问卷调查 | 添加问题 | P1 | 预定义问题 |
| 19 | 删除问题 | P2 | 移除已定义问题 |
| 20 | 分配问题 | P0 | 给问卷模板分配问题 |
| 21 | 添加模板 | P1 | 为问卷制定模板 |
| 22 | 修改模板 | P1 | 修改模板的信息 |
| 23 | 删除模板 | P2 | 移除问卷的模板 |
| 24 | 制作问卷 | P1 | 根据模板制作可以回答的问卷 |
| 25 | 发布问卷 | P1 | 将制作完成的问卷发布出去 |
| 26 | 回答问卷 | P0 | 回答已经发布的问卷 |
| 27 | 浏览回答 | P1 | 浏览对问卷的回答 |
| 28 | 个人中心 | 修改资料 | P2 | 用户修改自己的个人信息 |
| 29 | 管理员 | 修改资料 | P2 | 管理员修改自己的个人信息 |

1. **逻辑展示**

图示

描述已自动生成

图 1 功能结构图

图示

描述已自动生成

图 2 产品结构框架

图示

描述已自动生成

图 3 查找职工功能流程图

图示

描述已自动生成

图 4 分配床位功能流程图

1. **详细描述**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 主要功能点 | 功能描述 | 边界 | 异常 |
| 1 | 新增老人 | 往系统中添加老人 | 人名字数限制，电话号码规范等等 | 参数不合法 |
| 2 | 修改老人 | 修改已经入住的老人信息 | 人名字数限制，电话号码规范等等 | 参数不合法 |
| 3 | 删除老人 | 删除已经入住的老人信息 | 老人需与床位无关联 | 未解除关联 |
| 4 | 搜索老人 | 万能搜索，便于查找老人 | 搜索条件可以是老人信息的任意字段 | 非法输入 |
| 5 | 浏览老人 | 浏览所有老人信息，统筹管理 | 老人需在系统中 | 信息展示错误 |
| 6 | 录入员工 | 往系统中录入员工 | 人名字数限制，电话号码规范等等 | 参数不合法 |
| 7 | 移除员工 | 从系统中移除员工 | 员工需在系统中 | 参数不合法 |
| 8 | 搜索员工 | 万能搜索，便于管理员查找员工 | 搜索条件可以是员工信息的任意字段 | 参数不合法 |
| 9 | 分类浏览员工 | 浏览系统中的所有员工 | 员工类别将在医生护士护工中 | 参数不合法 |
| 10 | 更新员工信息 | 更新员工的个人信息 | 人名字数限制，电话号码规范等等 | 参数不合法 |
| 11 | 添加楼栋 | 添加社区中的楼栋信息 | 楼栋名称中文，字数有一定限制 | 参数不合法 |
| 12 | 删除楼栋 | 移除社区中的楼栋信息 | 楼栋下不能有房间，需先移除房间 | 参数不合法 |
| 13 | 添加房间 | 为楼栋添加房间信息 | 同楼栋下房间不能重名，字数有一定限制 | 参数不合法 |
| 14 | 移除房间 | 为楼栋移除房间信息 | 房间下不能已有床位，需先移除床位 | 参数不合法 |
| 15 | 添加床位 | 为房间添加床位信息 | 同房间内床位不能重名，有一定字数限制 | 参数不合法 |
| 16 | 移除床位 | 为房间移除床位信息 | 床位上不能有病人，需要先移除病人 | 参数不合法 |
| 17 | 分配床位 | 为床位分配老人 | 没有病人的床位添加未被分配的病人，已有病人的床位移除病人 | 参数不合法 |
| 18 | 添加问题 | 预定义问题 | 添加问题以及四个选项，长度有一定限制 | 参数不合法 |
| 19 | 删除问题 | 移除已定义问题 | 问题不能已经被模板引用，需先解除 | 参数不合法 |
| 20 | 分配问题 | 给问卷模板分配问题 | 将未引用的问题添加进模板，将已引用的问题移除模板 | 参数不合法 |
| 21 | 添加模板 | 为问卷制定模板 | 添加问卷模板 | 参数不合法 |
| 22 | 修改模板 | 修改模板的信息 | 修改模板的名字 | 参数不合法 |
| 23 | 删除模板 | 移除问卷的模板 | 移除问题模板，先删除问题选项，再删除问卷模块 | 参数不合法 |
| 24 | 制作问卷 | 根据模板制作可以回答的问卷 | 将模板当前的问题状态定格，制作成为问卷 | 参数不合法 |
| 25 | 发布问卷 | 将制作完成的问卷发布出去 | 设置问卷的发布状态，只有发布了才能做 | 参数不合法 |
| 26 | 回答问卷 | 回答已经发布的问卷 | 用户对已经发布的问卷进行回答 | 参数不合法 |
| 27 | 浏览回答 | 浏览对问卷的回答 | 浏览用户已经对问卷的回答 | 参数不合法 |
| 28 | 修改资料 | 用户修改自己的个人信息 | 用户修改自己的个人资料，各字段有一定的校验规则 | 参数不合法 |

1. **风控说明**

|  |  |  |  |
| --- | --- | --- | --- |
| 风险点 | 影响 | 等级 | 改善方案 |
| 老人隐私信息泄露 | 导致系统信用体系崩溃 | P0 | 加强数据库的加密保护以及系统的安全管理体系，以局域网方式访问系统。 |
| 大型社区老人、员工以及管理员数据庞大 | 导致系统运行缓慢、崩溃等情况 | P1 | 对数据库的数据进行组织管理，后期可考虑采用分布式数据库等减轻数据库负载的架构。 |
| 管理系统数据的同步慢问题 | 导致真实情况与系统中存储情况存在差异 | P2 | 通过采取结构化的流程对系统进行操作。 |

1. **设计**
2. **概要设计**

1.1设计原则

* 数据实时性：数据满足基本的实时性，每次提交请求都能获取到最新数据
* 可扩充性：项目各模块之间应具有较低的耦合度，可以轻松添加新功能而不需改变现有功能
* 可维护性：项目各模块的代码应具有较低的冗余度，注释齐全，便于日后维护
* 先进性：项目不应使用过时的依赖版本，不调用过时的API方法
* 数据完整性：项目中应对传入数据库的字段进行基本的合法性校验
* 安全性：项目应具有基本的安全性，对诸如密码这样的字段进行加密

1.2设计规范

* 命名约定
* 系统中的控制器命名为xxxController
* 系统中的持久层接口命名为xxxDao
* 系统中的业务代码类命名为xxxService
* 系统中的数据表命名为tb\_xxx
* 系统中的类名以大驼峰方式命名，字段名、方法名以小驼峰方式命名
* 系统中数据表的字段单词之间以下划线分割，全小写命名
* 界面约定

系统中的页面总体以三个区域布局：

* Header页面头
* Aside左边栏
* Main页面主体部分
* 程序编写规范

项目包结构：

* Controller包：系统中的控制器
* Dao包：与Mybatis的Mapper相关联的接口
* Service包：业务代码
* Entity包：系统需要的实体类
* Utils包：系统中的工具类
* …

方法编写规则：

* 一个类中多次使用的代码提取出公共部分作为私有公共方法
* 多个类中多次使用的代码提取出公共部分作为共有的工具代码

1.3总体结构

如图所示：

图示

描述已自动生成

系统整体分为老人管理、问卷管理、员工管理和楼宇管理四个模块，系统的用户有普通员工和系统管理员。

员工可以进入老人管理、问卷管理模块；管理员可以进入员工管理、楼宇管理模块。

系统中的公共部分为个人资料模块，不论是员工还是管理员，都可以修改他们的个人资料。

1.4流程设计

各模块的操作流程如图：

图示

描述已自动生成

1.5性能设计

安全性：诸如密码这样的字段应该加密处理

稳定性：系统应保证代码中出现的异常都得到合理的处理而不是产生错误

扩展性：系统中各模块之间应保持较低耦合度，满足开闭原则

1.6尚未解决的问题

1. 团队成员还不能够完美配合
2. 团队成员对技术的理解参差不齐
3. **数据库设计**

2.1物理结构设计

2.1.1库存储

* 系统选取MySQL数据库
* 数据存放在磁盘

2.1.2表存储

数据表选取Innodb存储引擎，采用utf8mb4字符集

2.1.3索引

* + 每张表主键均有主键索引
  + 经常查询而修改较少的列设置索引

2.2数据字典

表格

中度可信度描述已自动生成

表格

描述已自动生成

表格

描述已自动生成

2.3注意事项

1. 修改表结构前做好备份
2. 字段类型的长度不能太长
3. 不过多使用复杂存储过程
4. 遵循可读性原则，用有意义的单词作为字段名
5. 遵循长名原则，尽量不使用缩写命名
6. 不同类型数据应该分开管理

2.4数据库表设计

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑表名 | 回答表 | | 物理表名 | | tb\_answer | |
| 主键 | id | | 索引 | | id | |
| 逻辑字段名 | 物理字段名 | 数据类型 | 长度 | 允许空值 | 默认值 | 备注 |
| id | 回答id | int | 10 | N |  | 问卷回答id |
| publish\_paper\_id | 发布问卷id | int | 10 | N |  | 出版的问卷id |
| patient\_id | 回答问卷用户id | int | 10 | N |  | 回答问卷用户id |
| answer | 答案序列 | varchar | 255 | N |  | 答案序列，比如{1,2,3,4}代表4个问题分别选了第1，2，3，4个选项 |
| create\_time | 填写时间 | datetime | 19 | N | CURRENT\_TIMESTAMP | 填写的时间 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑表名 | 床位表 | | 物理表名 | | tb\_bed | |
| 主键 | id | | 索引 | | id | |
| 逻辑字段名 | 物理字段名 | 数据类型 | 长度 | 允许空值 | 默认值 | 备注 |
| id | 床位id | int | 10 | N |  | 床位id |
| room\_id | 房间号id | int | 10 | Y |  | 房间号id |
| name | 床的名字 | varchar | 255 | N |  | 床的名字 |
| bed\_type | 床的类型 | int | 10 | N | 2 | 床的类型：1为软床；2为硬床 |
| create\_time | 创建时间 | datetime | 19 | N | CURRENT\_TIMESTAMP | 创建时间 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑表名 | 楼栋表 | | 物理表名 | | tb\_building | |
| 主键 | id | | 索引 | | id | |
| 逻辑字段名 | 物理字段名 | 数据类型 | 长度 | 允许空值 | 默认值 | 备注 |
| id | 楼栋id | int | 10 | N |  | 楼栋id |
| name | 楼栋名字 | varchar | 255 | N |  | 楼栋名字 |
| detail | 楼栋介绍 | varchar | 255 | N | 暂无介绍 | 楼栋介绍 |
| create\_time | 创建时间 | datetime | 19 | N | CURRENT\_TIMESTAMP | 创建时间 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑表名 | 日志表 | | 物理表名 | | tb\_log | |
| 主键 | id | | 索引 | | id | |
| 逻辑字段名 | 物理字段名 | 数据类型 | 长度 | 允许空值 | 默认值 | 备注 |
| id | 日志id | bigint | 19 | N |  | 日志id |
| url | 访问地址 | varchar | 255 | N |  | 请求的访问地址 |
| method | 请求方法 | varchar | 10 | N |  | 请求方法 |
| user\_id | 用户的id | bigint | 19 | N |  | 请求的用户的id |
| start\_time | 请求开始时间 | datetime | 19 | N | CURRENT\_TIMESTAMP | 请求的开始时间 |
| execution\_time | 执行时间 | bigint | 19 | N |  | 执行时间,毫秒数 |
| request\_param | 请求参数 | json | 1073741824 | N |  | 请求参数 |
| response\_body | 返回结果 | json | 1073741824 | N |  | 返回结果 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑表名 | 问卷模板表 | | 物理表名 | | tb\_paper | |
| 主键 | id | | 索引 | | id | |
| 逻辑字段名 | 物理字段名 | 数据类型 | 长度 | 允许空值 | 默认值 | 备注 |
| id | 问卷模板id | int | 10 | N |  | 问卷模板id |
| name | 问卷模板名字 | varchar | 20 | N |  | 问卷模板名字 |
| create\_time | 创建时间 | datetime | 19 | N | CURRENT\_TIMESTAMP | 创建时间 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑表名 | 发布问卷表 | | 物理表名 | | tb\_paper\_publish | |
| 主键 | id | | 索引 | | id | |
| 逻辑字段名 | 物理字段名 | 数据类型 | 长度 | 允许空值 | 默认值 | 备注 |
| id | 发布问卷表id | int | 10 | N |  | 发布问卷表的id |
| name | 试卷名称 | varchar | 255 | N |  | 试卷名称 |
| detail | 试卷具体内容,json格式 | json | 1073741824 | N |  | 试卷具体内容,json格式 |
| register\_time | 发布试卷的试卷 | datetime | 19 | N | CURRENT\_TIMESTAMP | 发布试卷的试卷 |
| status | 问卷的状态，1是不能做，2是可以做 | tinyint | 3 | N | 1 | 问卷的状态，1是不能做，2是可以做 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 逻辑表名 | 问卷模板问题映射表 | | 物理表名 | | | tb\_paper\_question | | |
| 主键 | id | | 索引 | | | id | | |
| 逻辑字段名 | 物理字段名 | 数据类型 | | 长度 | 允许空值 | | 默认值 | 备注 |
| id | 问卷模板问题映射表id | int | | 10 | N | |  | 问卷模板问题映射表id |
| paper\_id | 问卷模板id | int | | 10 | N | |  | 问卷模板id |
| question\_id | 题目id | int | | 10 | N | |  | 题目id |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑表名 | 老人表 | | 物理表名 | | tb\_patient | |
| 主键 | id | | 索引 | | id | |
| 逻辑字段名 | 物理字段名 | 数据类型 | 长度 | 允许空值 | 默认值 | 备注 |
| id | 老人id | int | 10 | N |  | 老人id |
| name | 老人姓名 | varchar | 255 | N |  | 老人姓名 |
| sex | 性别 ：1为男；2为女 | int | 10 | N |  | 性别 ：1为男；2为女 |
| age | 年龄 | int | 10 | N |  | 年龄 |
| phone | 电话 | varchar | 255 | N |  | 电话 |
| contact\_name | 紧急联系人姓名 | varchar | 255 | N |  | 紧急联系人姓名 |
| contact\_phone | 紧急联系人电话 | varchar | 255 | N |  | 紧急联系人电话 |
| bed\_id | 床位id | int | 10 | Y |  | 床位id |
| manager\_id | 负责人的id | int | 10 | Y |  | 负责人的id |
| create\_time | 创建时间 | datetime | 19 | N | CURRENT\_TIMESTAMP | 创建时间 |
| last\_update\_time | 更新时间 | datetime | 19 | N | CURRENT\_TIMESTAMP | 更新时间 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑表名 | 问题表 | | 物理表名 | | tb\_question | |
| 主键 | id | | 索引 | | id | |
| 逻辑字段名 | 物理字段名 | 数据类型 | 长度 | 允许空值 | 默认值 | 备注 |
| id | 题目id | int | 10 | N |  | 题目id |
| name | 题目名字 | varchar | 100 | N |  | 题目名字 |
| text | 题目内容 | json | 1073741824 | N |  | 题目内容，json格式 |
| create\_time | 创建时间 | datetime | 19 | N | CURRENT\_TIMESTAMP | 创建时间 |
| last\_update\_time | 更新时间 | datetime | 19 | N | CURRENT\_TIMESTAMP | 更新时间 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑表名 | 房间表 | | 物理表名 | | tb\_room | |
| 主键 | id | | 索引 | | id | |
| 逻辑字段名 | 物理字段名 | 数据类型 | 长度 | 允许空值 | 默认值 | 说明 |
| id | 房间号id | int | 10 | N |  | 房间号id |
| building\_id | 楼栋 id | int | 10 | Y |  | 楼栋 id |
| name | 房间名 | varchar | 255 | N |  | 房间名 |
| room\_type | 房间类型 | int | 10 | N | 3 | 房间类型：1为史诗病房；2为稀有病房；3为普通病房 |
| create\_time | 创建时间 | datetime | 19 | N | CURRENT\_TIMESTAMP | 创建时间 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑表名 | 用户表 | | 物理表名 | | tb\_user | |
| 主键 | id | | 索引 | | id | |
| 逻辑字段名 | 物理字段名 | 数据类型 | 长度 | 允许空值 | 默认值 | 说明 |
| id | 用户id | int | 10 | N |  | 用户id |
| username | 用户名 | varchar | 255 | N |  | 用户名 |
| password | 密码 | varchar | 255 | N |  | 密码 |
| name | 真实姓名 | varchar | 255 | N |  | 真实姓名 |
| type | 用户类型 | int | 10 | N |  | 用户类型 1 护工 ；2 护士 3 医生 ；4 管理员 |
| phone | 电话 | varchar | 255 | N |  | 电话 |
| register\_time | 注册时间 | datetime | 19 | N | CURRENT\_TIMESTAMP | 注册时间 |
| last\_update\_time | 更新时间 | datetime | 19 | N | CURRENT\_TIMESTAMP | 最后一次修改的时间 |
| show | 显示状态 | int | 10 | N | 1 | 显示状态：0为不存在，1为存在 |

1. **接口设计（前后端分离项目）**

3.1外部接口

本系统中没有外部接口

3.2内部接口

说明本系统之内的各个系统元素之间的接口的安排。

|  |  |  |  |
| --- | --- | --- | --- |
| 接口ID | 接口地址 | 接口类型 | 接口描述 |
| 1 | /building/all | GET | 获取所有楼栋信息 |
| 2 | /building/{id} | GET | 根据id获取楼栋信息 |
| 3 | /building | POST | 注册楼栋 |
| 4 | /building | DELETE | 删除楼栋 |
| 5 | /building | PUT | 更新楼栋信息 |
| 6 | /building/room | POST | 注册房间 |
| 7 | /building/room | DELETE | 删除房间 |
| 8 | /building/room | PUT | 更新房间信息 |
| 9 | /building/bed | POST | 注册床位 |
| 10 | /building/bed | DELETE | 删除床位 |
| 11 | /building/bed | PUT | 更新床位信息 |
| 12 | /paper/all | GET | 获取所有问卷模板 |
| 13 | /paper | PUT | 更新问卷 |
| 14 | /paper | DELETE | 删除问卷 |
| 15 | /paper/{paperId}/{questionId} | PUT | 向问卷添加问题 |
| 16 | /paper/{paperId}/{questionId} | DELETE | 从问卷删除问题 |
| 17 | /paper | POST | 新增问卷 |
| 18 | /paper/publish/all | GET | 获得所有已发布问卷 |
| 19 | /paper/publish/{id} | GET | 根据id获取已发布问卷 |
| 20 | /paper/publish | POST | 新增发布问卷 |
| 21 | /paper/publish | PUT | 更新已发布的问卷 |
| 22 | /paper/question | GET | 获取所有问题 |
| 23 | /paper/question/notIn/{paperId}/ | GET | 获取所有不在所传id问卷中的问题 |
| 24 | /paper/question/{paperId} | GET | 获取所有在所传id问卷中的问题 |
| 25 | /paper/question | POST | 向系统中新增问题 |
| 26 | /paper/question | PUT | 更新问题 |
| 27 | /paper/question | DELETE | 删除问题 |
| 28 | /paper/answer/all | GET | 查看所有回答 |
| 29 | /paper/{paperId}/answer | POST | 对问卷进行回答 |
| 30 | /patient | POST | 添加老人 |
| 31 | /patient | PUT | 更改老人 |
| 32 | /patient | DELETE | 删除老人 |
| 33 | /patient/{id} | GET | 根据id查询老人 |
| 34 | /patient/all | GET | 查询所有老人信息 |
| 35 | /patient/notHaveBed | GET | 查询所有没分配床位的老人 |
| 36 | /user/all | GET | 查询系统中所有员工信息 |
| 37 | /user/{id} | GET | 根据id查询系统中的员工信息 |
| 38 | /user/username | GET | 根据名字查询用户 |
| 39 | /user/login | POST | 登陆 |
| 40 | /user/register | POST | 注册用户 |
| 41 | /user | DELETE | 删除用户 |
| 42 | /user | PUT | 更新用户信息 |

1. **详细设计**

4.1开发环境

4.1.1数据库管理系统

* 产品名称:Navicat
* 版本:15.x

4.1.2开发工具、中间件

* 开发工具:IntelliJ IDEA
* 版本:2021.2.x
* 版本控制工具:Git
* 版本:2.1

4.1.3硬件环境

* 系统版本:Windows10
* 内存：16GB
* 磁盘空间：500GB

4.2程序系统的结构

图示

描述已自动生成

图 1 程序系统结构图

4.3模块命名的规则

项目包结构：

* Controller包：系统中的控制器
* Dao包：与Mybatis的Mapper相关联的接口
* Service包：业务代码
* Entity包：系统需要的实体类
* Utils包：系统中的工具类
* …

方法编写规则：

* 一个类中多次使用的代码提取出公共部分作为私有公共方法
* 多个类中多次使用的代码提取出公共部分作为共有的工具代码

4.4系统架构设计

4.4.1模块汇总表

|  |  |
| --- | --- |
| **职工子系统** | |
| 模块名称 | 功能简述 |
| 老人管理 | 对颐养社区内的老人进行管理 |
| 问卷管理 | 对颐养社区内的问卷进行管理 |
| **管理员子系统** | |
| 模块名称 | 功能简述 |
| 职工管理 | 对颐养社区内的员工进行管理 |
| 楼宇管理 | 对颐养社区内的楼宇进行管理 |

4.4.2模块关系图

图示

描述已自动生成

图 2 颐养社区管理系统模块关系图

4.5子系统A的模块设计

4.5.1模块A-1

|  |  |
| --- | --- |
| 模块名称 | 老人管理 |
| 功能描述 | 管理老人，老人信息增删改查 |
| 接口与属性 | |  |  |  |  | | --- | --- | --- | --- | | 接口地址 | 方法类型 | 输入参数 | 返回值 | | /patient | POST | Patient | int | | /patient | PUT | Patient | int | | /patient | DELETE | id | int | | /patient/{id} | GET | id | Patient | | /patient/all | GET | 无参数 | List<Patient> | | /patient/notHaveBed | GET | 无参数 | List<Patient> | |
| 数据结构  与算法 | public class Patient {  private Integer id; //病人id  private String name; //病人姓名  private Integer sex; //病人性别  private Integer age; //病人年龄  private String phone; //病人电话  private String contactName; //紧急联系人  private String contactPhone; //紧急联系人电话  private Integer bedId; //床位id  private Integer managerId; //负责人的id  private User manager; //负责人的信息  private String loadingName; //住宿名字  private String createTime; //注册时间  private String lastUpdateTime; //最后一次修改的时间  } |
| 补充说明 |  |

4.5.2模块A-2

|  |  |
| --- | --- |
| 模块名称 | 问卷管理 |
| 功能描述 | 管理问卷，问卷模板增删改查，问题增删改查，问卷的发布 |
| 接口与属性 | |  |  |  |  | | --- | --- | --- | --- | | 接口地址 | 方法类型 | 输入参数 | 返回值 | | /paper/all | GET | 无参数 | List<Paper> | | /paper | PUT | Paper | int | | /paper | DELETE | id | int | | /paper/{paperId}/{questionId} | PUT | paperId, questionId | int | | /paper/{paperId}/{questionId} | DELETE | paperId, questionId | int | | /paper | POST | Paper | int | | /paper/publish/all | GET | 无参数 | List<Paper> | | /paper/publish/{id} | GET | id | Paper | | /paper/publish | POST | PublishPaper | int | | /paper/publish | PUT | PublishPaper | int | | /paper/question | GET | 无参数 | List<Question> | | /paper/question/notIn/{paperId}/ | GET | paperId | List<Question> | | /paper/question/{paperId} | GET | paperId | List<Question> | | /paper/question | POST | Question | int | | /paper/question | PUT | Question | int | | /paper/question | DELETE | questionId | int | | /paper/answer/all | GET | 无参数 | List<Answer> | | /paper/{paperId}/answer | POST | Answer | int | |
| 数据结构  与算法 | public class Paper {  private Integer id; //问卷id  private String name; //问卷名字  private String registerTime; //注册时间  private List<Question> questionList;  }  public class Answer {  private Integer id; //回答问卷的id  private Integer publishPaperId; //问卷id  private Integer patientId; //回答问卷用户的id  private String answer; //答案的序列，比如{1,2,3,4}代表4个问题分别选了第1，2，3，4个选项  private String createTime; //注册的时间  private String patientName; //病人的姓名  private String publishPaperName;  }  public class Option {  private String text;  }  public class PublishQuestion {  private Integer id;  private String name;  private List<String> options = new ArrayList<>(4);  }  public class PublishPaper {  private Integer id; //问卷id  private String name; //问卷名字  private String registerTime; //注册时间  private String detail; //问卷详情  private Integer status; //发布的状态  }  public class Question {  private Integer id; //题目id  private String name; //题目名字  private String text; //题目内容，json格式  private String registerTime; //注册时间  private String lastUpdateTime; //最后一次修改的时间  } |
| 补充说明 |  |

4.6子系统B的模块设计

4.6.1模块B-1

|  |  |
| --- | --- |
| 模块名称 | 职工管理 |
| 功能描述 | 管理职工，职工信息的增删查改 |
| 接口与属性 | |  |  |  |  | | --- | --- | --- | --- | | 接口地址 | 方法类型 | 输入参数 | 返回值 | | /user/all | GET | 无参数 | List<User> | | /user/{id} | GET | id | User | | /user/username | GET | username | User | | /user/login | POST | User | User | | /user/register | POST | User | int | | /user | DELETE | id | int | | /user | PUT | User | int | |
| 数据结构  与算法 | public class User {  private Integer id; //用户id  private String username; //用户名  private String password; //密码  private String name; //真实姓名  private Integer type; //用户类型  private String phone; //用户电话  private String registerTime; //注册时间  private String lastUpdateTime; //最后一次修改的时间  private Integer show; //用户存在的状态  } |
| 补充说明 |  |

4.6.2模块B-2

|  |  |
| --- | --- |
| 模块名称 | 楼宇管理 |
| 功能描述 | 管理职工，职工信息的增删查改 |
| 接口与属性 | |  |  |  |  | | --- | --- | --- | --- | | 接口地址 | 方法类型 | 输入参数 | 返回值 | | /building/all | GET | 无参数 | List<Building> | | /building/{id} | GET | id | Building | | /building | POST | Building | int | | /building | DELETE | id | int | | /building | PUT | Building | int | | /building/room | POST | Room | int | | /building/room | DELETE | int | int | | /building/room | PUT | Room | int | | /building/bed | POST | Bed | int | | /building/bed | DELETE | int | int | | /building/bed | PUT | Bed | int | |
| 数据结构  与算法 | public class Building {  private Integer id; //楼栋id  private String name; //楼栋名字  private String detail; //楼栋介绍  private String createTime; //注册时间  private List<Room> roomList;  }  public class Room {  private Integer id; //房间号id  private Integer buildingId; //楼栋id  private String name; //房间名  private Integer roomType; //房间类型 0为传说病房 1为史诗病房 2为稀有病房 3为普通病房  private String createTime; //注册时间  private List<Bed> bedList;  }  public class Bed {  private Integer id; //床位id  private Integer roomId; //房间号id  private String name; //床的名字  private Integer BedType; //床的类型 0为软床，1为硬床  private Patient patient; //床绑定病人的信息  private String createTime; //注册时间  } |
| 补充说明 |  |

4.7裁剪说明

以上各章内容，可根据项目需要增加内容，或者根据实际情况修改章节结构。

1. 开发
2. 测试
3. 测试方式

例如：本次测试为系统测试，采用黑盒测试方式。测试用例情况，测试的流程等等。

1. 测试资料

例如：测试方案、测试用例、测试流程等。

1. 测试内容

|  |  |  |  |
| --- | --- | --- | --- |
| 测试系统 | 版本号 | 模块数 | 模块标号 |
| XX子系统 | XXXXX\_vX.X.1 | 2 | …… |
| …… |
| XX子系统 | XXXX\_vX.X.0 | 2 | …… |
|  |  |  | …… |
| XX子系统 | XXXX\_v2.2.0 | 1 | …… |

1. 测试结果与缺陷分析

4.1测试覆盖分析

例如：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **测试系统** | **需求编号** | **测试类型** | **用例个数** | **执行个数** | **执行用例BUG数** | **BUG未解决个数** | **是否通过** |
| XX系统 | XXXX-950 | 系统测试 | 3 | 3 | 0 | 0 | 是 |
| XXXX-952 | 系统测试 | 8 | 8 | 1 | 0 | 是 |
| XX系统 | XXXXX-81 | 系统测试 | 10 | 10 | 0 | 0 | 是 |
| XXXXX-92 | 系统测试 | 12 | 12 | 0 | 0 | 是 |
| XX系统 | XXX-71 | 系统测试 | 12 | 12 | 3 | 0 | 是 |

4.2缺陷的统计与分析

|  |  |  |
| --- | --- | --- |
| **测试系统** | **BUG率 （BUG数/测试用例个数）** | **测试未通过率 （未通过BUG个数/BUG个数）** |
| XX系统 | X.X% | 0% |
| XX系统 | XX.XX% | 0% |

1. 测试结论

例如：本次测试XX系统XXXX\_vX.X.X版本需求xxx个、xxx个用例，均测试通过。

XXXX年XX月，测试过程中记录的的缺陷均已全部解决，并且回归测试通过。

目前，系统的运行状况为：

a.各系统流程正常；

b.需求各个功能点实现正确；

c.暂不修复问题，不影响系统正常使用，已经记录XXXX；

综上可知，各系统功能正确实现，流程稳定，均达到上线要求。

附件：

项目预算核算模板

项目变更申请表