**【新空间装饰业务管理系统】**

**详细设计规格说明书**

**[版本号V\_1.00.00]**

**2021年10月22日**

**组长：万理**

**副组长：孙圭楚**

**组员：王戬 刘耀壕 杨连佳 马倩 吴汶峰**

**目录**

[1. 引言 3](#_Toc90476668)

[1.1. 编写目的 3](#_Toc90476669)

[2. 系统结构 3](#_Toc90476670)

[2.1 硬件设计 3](#_Toc90476671)

[2.2 软件设计 3](#_Toc90476672)

[3. 模块设计说明 4](#_Toc90476673)

[4. 参考文献 24](#_Toc90476674)

# 引言

## 编写目的

本次的详细设计是基于核心代码基本开发完毕，在第一代系统基本正常运行后编写的，主要目的是为了后续开发与维护的便利性。

该文档主要受众为该系统后续开发人员，并且在阅读此文档前最后先阅读本系统的概要设计文档。在概要设计中，我们简单的依据需求文档将实现的功能进行的分块。而在此文档，我们将基于此对具体到类的部分进行详细设计。

# 系统结构

## 硬件设计

该系统主要基于软件部分的开发，所以对于硬件部分，只需要满足运行软件这一个需求即可，具体来说即：

PC机一台，安装Windows10操作系统

 CPU：P4 1.8GHz以上

 内存：2GB以上

## 软件设计

现将每一部分拆解开来，对于该系统，所包含的主要模块如下图所示：

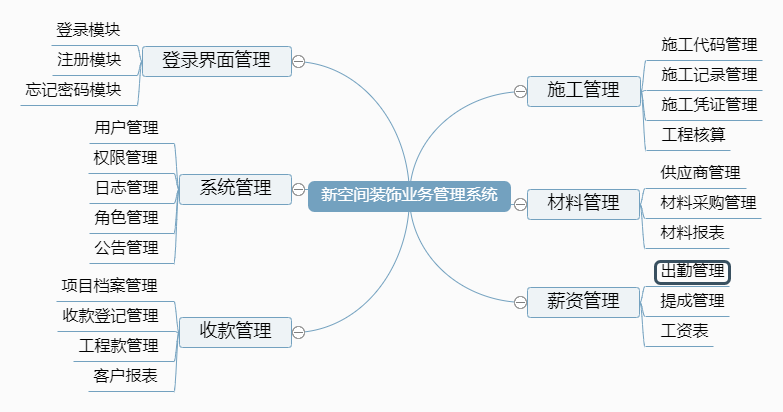


图1.新空间装饰业务管理系统模块设计图

# 模块设计说明

由上述给出软件设计框图，对各个模块展开设计如下：

1. 登录界面管理

|  |
| --- |
| 登录模块设计 |
| 1.1 模块描述：输入用户名，密码与验证码，登录进系统 |
| 1.2 功能：接受用户输入的用户名，密码与验证码，并逐一检查验证码，用户名与密码是否正确，如正确则登入，失败则提示。 |
| 1.3 性能：能够检查用户名、密码的正确性。 |
| 1.4 输入项：用户名，密码与验证码。 |
| 1.5 输出项：成功登录、“验证码错误”或“账号或密码错误”。 |
| 1.6流程逻辑：首先检验验证码是否正确，错误这返回“验证码错误”，正确则将用户名与密码打包成一个实体，传输到后端匹配，如找到，则登录成功，否则返回“账号或密码错误”。 |
| 1.7 流程逻辑图：  这里对这一流程进行建模与描述。简要逻辑如下图：    图2登录模块流程逻辑图 |
| 1.8 接口：  public Result<User> login(@RequestBody User user, HttpServletRequest request)：前端用户登录调用的后端数据的接口。 |

|  |
| --- |
| 注册模块设计 |
| 1.1 模块描述：注册一个新的用户。 |
| 1.2 功能：输入用户名等信息，注册一个新的用户。 |
| 1.3 性能：能过检查用户名、密码等的正确性。 |
| 1.4 输入项：用户名、密码、密保问题及答案。 |
| 1.5 输出项：“注册成功”、"请输入用户名"、"用户名不规范"、"请输入密码"、"密码格式不规范"、"密码太简单了哦，加上字母试试吧"、"请选择密保问题"、"请输入密保答案"。 |
| 1.6 设计方法（算法）：接受用户输入的数据，首先检查是否输入用户名，检查用户名是否规范、检查是否输入密码、检查密码是否规范、检查密码复杂度是否符合要求、检查是否选择密保问题、最后检查是否输入密保问题答案。 |
| 1.7 流程逻辑：  简要逻辑如下图：    图3注册模块流程逻辑图 |
| 1.8 接口：  public Result<User> register(@RequestBody User user, HttpServletRequest request)：前端用户登录调用的后端数据的接口。 |

|  |
| --- |
| 忘记密码模块设计 |
| 1.1 模块描述：输入用户名与密保问题等数据修改密码。 |
| 1.2 功能：根据输入的用户名，新密码，密保问题及答案修改该用户的密码。 |
| 1.3 性能：能过检查用户名、密码等的正确性。 |
| 1.4 输入项： 用户名，新密码，密保问题及答案 |
| 1.5 输出项："请输入用户名"、"请选择密保问题"、 "请输入密保答案"、"请输入新密码"、"密码格式不规范"、"密码太简单了哦，加上字母试试吧"、"修改成功" |
| 1.6 设计方法（算法）：接受数据，检查用户名是否输入，检查是否选择密保问题，检查密保问题是否正确，检查是否输入新密码，检查新密码格式，检查新密码复杂度。 |
| 1.7流程逻辑：  简要逻辑如下图：    图4修改密码模块流程逻辑图   |  | | --- | |  | |
| 1.8 接口：   |  | | --- | | public Result<User> reset(@RequestBody User user, HttpServletRequest request)：前端用户**登录调用的**后端数据的接口。 | |

1. 系统管理

|  |
| --- |
| 用户管理模块设计 |
| 1.1 模块描述：管理用户的ID、用户名等数据。 |
| 1.2 功能：对用户的数据的增删改查以及搜素功能 。 |
| 1.3 性能： 能检查增加，修改的数据的正确性。 |
| 1.4 输入项：一个用户实体类，包括用户ID，用户名，昵称  ，邮箱，电话，年龄，地址，角色，及相应权限的设置。 |
| 1.5 输出项：修改数据库 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7流程逻辑：  简要逻辑如下图：    图5用户管理模块新增用户流程逻辑图 |
| 1.8 接口：  public Result<?> save(@RequestBody User user)：新增用户数据接口   |  | | --- | | public Result<?> update(@RequestBody User user)：更新用户数据接口  public Result<?> delete(@PathVariable Long id)：删除用户数据接口  public Result<User> findByUsername(@PathVariable String username)：搜索用户数据接口 | |

|  |
| --- |
| 权限管理模块设计 |
| 1.1 模块描述：管理各个模块对应权限 |
| 1.2 功能：对各种模块的相应权限进行管理 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：一个权限实体类，包括ID，名称，描述，菜单路径及图标。 |
| 1.5 输出项：修改数据库 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7流程逻辑：  简要逻辑如下图：    图6权限管理模块更新权限流程逻辑图 |
| 1.8 接口：   |  | | --- | | public Result<?> save(@RequestBody Permission permission)：新增权限接口  public Result<?> update(@RequestBody Permission permission)：更新权限接口  public Result<?> delete(@PathVariable Long id)：删除权限接口  public Result<Permission> findById(@PathVariable Long id)：搜索权限接口 | |

|  |
| --- |
| 日志管理模块设计 |
| 1.1 模块描述：管理系统日志 |
| 1.2 功能：记录系统的登录情况，并且可以进行搜索 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：用户名 |
| 1.5 输出项： 相应日志 |
| 1.6 设计方法（算法）：记录数据，搜索日志。 |
| 1.7流程逻辑：  简要逻辑如下图：    图7日志管理模块搜索日志流程逻辑图 |
| 1.8 接口：   |  | | --- | | public Result<?> save(@RequestBody Log log)：记录日志接口  public Result<?> findById(@PathVariable Long id)：搜索日志接口 | |

|  |
| --- |
| 角色管理模块设计 |
| 1.1 模块描述：管理对应角色权限 |
| 1.2 功能：对不同角色权限的增删改查。 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：一个角色类，包括ID，名称，描述，权限 |
| 1.5 输出项： 修改数据库 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7流程逻辑：  简要逻辑如下图：    图7角色管理模块删除角色流程逻辑图 |
| 1.8 接口：   |  | | --- | | void AddBlockArea(BlockArea \*area);  增加障碍区域  public Result<?> save(@RequestBody Role role)：新增角色接口  public Result<?> update(@RequestBody Role role)：更新角色接口  public Result<?> delete(@PathVariable Long id)：删除角色接口  public Result<Role> findById：搜索角色接口 | |

|  |
| --- |
| 公告管理模块设计 |
| 1.1 模块描述：管理公告 |
| 1.2 功能：对公告的增删改查 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：一个公告类，包括ID，标题，内容，时间 |
| 1.5 输出项：修改数据库 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7 接口：   |  | | --- | | public Result<?> save(@RequestBody Message Message)：新增公告接口  public Result<?> update(@RequestBody Message Message)：更新公告接口 | | public Result<?> delete(@PathVariable Long id)：删除公告接口 | | public Result<?> findById(@PathVariable Long id)：  搜索公告接口 | |

1. 收款管理

|  |
| --- |
| 项目档案管理模块设计 |
| 1.1 模块描述：管理全部项目的项目ID、造价等数据。 |
| 1.2 功能：对项目这一个实体类的增删改查。 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：面积、小区id、项目状态、合同编号、完工日期、部门所属id、项目经理id、合同造价、客户姓名、客户性别。 |
| 1.5 输出项：修改数据库。 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7 接口：  public Result<?> save(@RequestBody Project project)：新增项目接口  public Result<?> update(@RequestBody Project project)：更新项目接口  public Result<?> delete(@PathVariable Long id)：删除项目接口  public Result<?> findById(@PathVariable Long id)：搜索项目接口 |

|  |
| --- |
| 收款登记管理模块设计 |
| 1.1 模块描述：管理全部项目的收款登记数据。 |
| 1.2 功能：对收款这一实体的增删改查。 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：收款日期、收款金额、备用、项目代码。 |
| 1.5 输出项：修改数据库。 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7 主要接口：  public Result<?> save(@RequestBody Receipt receipt)：新增收款接口  public Result<?> update(@RequestBody Receipt receipt)：更新收款接口  public Result<?> delete(@PathVariable Long id)：删除收款接口  public Result<?> findById(@PathVariable Long id)：搜素收款接口 |

|  |
| --- |
| 工程款管理模块设计 |
| 1.1 模块描述：管理全部项目的工程款数据。 |
| 1.2 功能：对工程款这一实体的增删改查。 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：日期、项目代码、增减金额、增减内容、备用。 |
| 1.5 输出项：修改数据库。 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7 接口：  public Result<?> save(@RequestBody Constructionfund constructionfund):新增工程款接口  public Result<?> update(@RequestBody Constructionfund constructionfund):更新工程款接口  public Result<?> delete(@PathVariable Long id):删除工程款接口  public Result<?> findById(@PathVariable Long id):搜索工程款接口 |

|  |
| --- |
| 客户报表模块设计 |
| 1.1 模块描述：生成包含各种数据的客户报表。 |
| 1.2 功能：根据输入的项目ID，生成对应项目的报表。 |
| 1.3 性能：检查项目ID的正确性。 |
| 1.4 输入项：项目ID。 |
| 1.5 输出项：日期、客户名称、小区号、项目经理ID、项目款项、合同总造价、工程款增减合计、收款合计、应收余额、项目经理名、联系电话。 |
| 1.6 设计方法（算法）：根据输入的项目ID，查询所需要的数据，并进行计算，将所得的数据输出到前端，展示给用户。 |
| 1.7流程逻辑：  简要逻辑如下图：    图7客户报表管理模块流程逻辑图 |
| 1.8 接口：  public Result<CustomerRep> generate(@RequestParam(required=false,defaultValue = "") String id)：根据项目ID得到报表 |

1. 施工管理

|  |
| --- |
| 施工代码管理模块设计 |
| 1.1 模块描述：对全部项目的施工代码管理。 |
| 1.2 功能：对施工代码的增删改查。 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：施工凭证号、备用。 |
| 1.5 输出项：修改数据库。 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7 接口：  public Result<?> save(@RequestBody Constructioninfo constructioninfo)：新增施工代码接口  public Result<?> update(@RequestBody Constructioninfo constructioninfo)：更新施工代码接口  public Result<?> delete(@PathVariable Long id)：删除施工代码接口  public Result<?> findById(@PathVariable Long id)：搜索施工代码接口 |

|  |
| --- |
| 施工记录管理模块设计 |
| 1.1 模块描述：对全部项目的施工记录管理。 |
| 1.2 功能：对施工记录的增删改查。 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：施工代码、施工日期、使用施工款、项目实施名称、项目代码、员工id。 |
| 1.5 输出项：修改数据库。 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7 接口：  public Result<?> save(@RequestBody Constructionrec constructionrec)：新增施工记录接口  public Result<?> update(@RequestBody Constructionrec constructionrec) ：更新施工记录接口  public Result<?> delete(@PathVariable Long id)：删除施工记录接口  public Result<?> findById(@PathVariable Long id)：搜索施工记录接口 |

|  |
| --- |
| 施工凭证管理模块设计 |
| 1.1 模块描述：对全部项目的施工凭证管理。 |
| 1.2 功能：对施工凭证的增删改查。 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：凭证名称。 |
| 1.5 输出项：修改数据库。 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7 接口：  public Result<?> save(@RequestBody Constructioncert constructioncert):新增施工凭证接口  public Result<?> update(@RequestBody Constructioncert constructioncert):更新施工凭证  public Result<?> delete(@PathVariable Long id):删除施工凭证  public Result<?> findById(@PathVariable Long id):搜索施工凭证 |

|  |
| --- |
| 工程核算模块设计 |
| 1.1 模块描述：生成包含各种数据的工程核算页面。 |
| 1.2 功能：根据输入的项目ID，生成对应项目的工程核算页面。 |
| 1.3 性能：检查项目ID的正确性。 |
| 1.4 输入项：项目ID。 |
| 1.5 输出项：总花销、项目号、凭证号、凭证名、施工名称、施工各项花销、工程项目、主体拆改、水电改造、木工、瓦工、油工、厨工吊顶安装、橱柜安装、木门安装、地板安装、开关插座安装、灯具安装、五金洁具安装、窗帘杆安装、凭证花销比例统计。 |
| 1.6 设计方法（算法）：根据输入的项目ID，查询所需要的数据，并进行计算，将所得的数据输出到前端，展示给用户。 |
| 1.7 接口：  public Result<ConstructionRepex> generateall(@RequestParam(required=false,defaultValue = "") String id)：根据项目ID得到工程核算 |

1. 材料管理

|  |
| --- |
| 供应商管理模块设计 |
| 1.1 模块描述：对全部项目的供应商管理。 |
| 1.2 功能：对供应商的增删改查。 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：地址、联系人、供应商名称、联系信息. |
| 1.5 输出项：修改数据库。 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7 接口：  public Result<?> save(@RequestBody Supplyinfo supplyinfo)：新增供应商接口  public Result<?> update(@RequestBody Supplyinfo supplyinfo)：更新供应商接口  public Result<?> delete(@PathVariable Long id)：删除供应商接口  public Result<?> findById(@PathVariable Long id)：搜索供应商接口 |

|  |
| --- |
| 材料采购管理模块设计 |
| 1.1 模块描述：对全部项目的材料采购管理。 |
| 1.2 功能：对材料采购的增删改查。 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：材料价格、材料名称、项目id、供应商id |
| 1.5 输出项：修改数据库。 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7 接口：  public Result<?> save(@RequestBody Materialpurchase materialpurchase)：新增材料采购接口  public Result<?> update(@RequestBody Materialpurchase materialpurchase)：更新材料采购接口  public Result<?> delete(@PathVariable Long id)：删除材料采购接口  public Result<?> findById(@PathVariable Long id)：搜索材料采购接口 |

|  |
| --- |
| 材料报表模块设计 |
| 1.1 模块描述：生成包含各种数据的工程核算页面。 |
| 1.2 功能：根据输入的项目ID，生成对应项目的工程核算页面。 |
| 1.3 性能：检查项目ID的正确性。 |
| 1.4 输入项：项目ID。 |
| 1.5 输出项：项目号、材料单号、材料名称、材料价格、供应商名称、总材料价格。 |
| 1.6 设计方法（算法）：根据输入的项目ID，查询所需要的数据，并进行计算，将所得的数据输出到前端，展示给用户。 |
| 1.7 接口：  public Result<MaterialRepex> generateall(@RequestParam(required=false,defaultValue = "") String id)：据项目ID得到材料报表 |

1. 薪资管理

|  |
| --- |
| 出勤管理模块设计 |
| 1.1 模块描述：对全部员工的出勤管理。 |
| 1.2 功能：对出勤管理的增删改查。 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：日期、员工名称、员工id。 |
| 1.5 输出项：修改数据库。 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7 接口：  public Result<?> save(@RequestBody Attendence attendence)：新增出勤管理接口  public Result<?> update(@RequestBody Attendence attendence)：更新出勤管理接口  public Result<?> delete(@PathVariable Long id)：删除出勤管理接口  public Result<?> findById(@PathVariable Long id)：搜索出勤管理接口 |

|  |
| --- |
| 提成管理模块设计 |
| 1.1 模块描述：对全部员工的提成管理。 |
| 1.2 功能：对提成管理的增删改查。 |
| 1.3 性能：能检查增加，修改的数据的正确性。 |
| 1.4 输入项：日期、提成、员工名称、员工id。 |
| 1.5 输出项：修改数据库。 |
| 1.6 设计方法（算法）：根据输入的数据修改数据库。 |
| 1.7 接口：  public Result<?> save(@RequestBody Commission commission)：新增提成管理接口  public Result<?> update(@RequestBody Commission commission)：更新提成管理接口  public Result<?> delete(@PathVariable Long id)：删除提成管理接口  public Result<?> findById(@PathVariable Long id)：搜索提成管理接口 |

|  |
| --- |
| 工资表模块设计 |
| 1.1 模块描述：生成包含各种数据的工程核算页面。 |
| 1.2 功能：根据输入的员工ID及时间，生成对应项目的工程核算页面。 |
| 1.3 性能：检查员工ID及使劲按的正确性。 |
| 1.4 输入项：项目ID、时间。 |
| 1.5 输出项： 应发工资来源、应发工资去向、id、姓名、工资卡号、出勤天数、基本工资、业务提成、合计、罚款、代扣养老保险、代扣医疗保险、代扣失业保险、代扣住房公积金、代扣个税、实发工资、备注、年月、公司名称。 |
| 1.6 设计方法（算法）：根据输入的员工ID及时间，查询所需要的数据，并进行计算，将所得的数据输出到前端，展示给用户。 |
| 1.7 接口：  public Result<Salary> generate(@RequestParam(required=false,defaultValue = "") String id,@RequestParam(value ="date", required = false, defaultValue = "1") String date)：据员工ID及时间得到工资表 |

# 4. 参考文献

[1] MyBatis-Plus官方文档：https://baomidou.com/

[2] Vue.js 官方文档：<https://cn.vuejs.org/v2/guide/>

[3] [vue-element-admin](https://panjiachen.gitee.io/vue-element-admin-site/zh/)官方文档：<https://panjiachen.gitee.io/vue-element-admin-site/zh/guide/>

[4] W3school教程:https://www.w3school.com.cn/

[5]Navicat Premium介绍：<http://www.navicat.com.cn/products/navicat-premium>

[6] Ajax官方文档：<https://api.jquery.com/category/ajax/>

[7]Mysql官方文档：<https://www.mysql.com/>

[8] Spring MVC框架入门教程：http://c.biancheng.net/spring\_mvc/