# 《宿舍管理系统》

目录

[一、前言 3](#_Toc1857)

[1.项目描述： 3](#_Toc22357)

[2.项目背景： 3](#_Toc24014)

[3.项目目的： 3](#_Toc24627)

[二、宿舍管理系统的需求分析 4](#_Toc27080)

[1.功能需求： 4](#_Toc25645)

[2.性能需求： 4](#_Toc4926)

[3.环境需求： 5](#_Toc16442)

[4.接口需求： 6](#_Toc3050)

[5.用户界面需求： 7](#_Toc27998)

[6.数据需求： 7](#_Toc1577)

[三、宿舍管理系统的逻辑模型 9](#_Toc9284)

[1.基础应知： 9](#_Toc28851)

[2.数据流图的生成 9](#_Toc13969)

[3.数据字典 16](#_Toc17698)

[四、概要设计： 24](#_Toc5089)

[五、详细设计 36](#_Toc10110)

[系统管理： 36](#_Toc24458)

[公寓管理： 37](#_Toc18285)

[出入登记： 38](#_Toc26010)

[学生管理; 39](#_Toc14274)

[六、系统测试： 40](#_Toc28846)

[1.白盒测试： 41](#_Toc22384)

[2.黑盒测试： 45](#_Toc20367)

# 一、前言

## 项目描述：

本系统主要服务于在校大学生和宿舍管理员，通过构建一个功能完善的宿舍管理平台，实现对学生宿舍、公寓设施以及日常出入管理的高效监管。系统设计满足不同用户权限需求，包括系统管理、公寓管理、学生管理以及出入登记管理等核心功能。这些功能旨在提升宿舍管理的自动化和信息化水平，减少人工操作错误，提高管理效率和住户满意度。宿舍管理系统不仅存储大量的宿舍和学生信息于数据库中，还由管理员进行数据的规范统一管理。此外，系统支持与高校现有的校园管理系统对接，成为学校管理体系的一部分，推动校园信息化进程。

## 2.项目背景：

高校宿舍管理的效率和质量直接影响到学生的生活质量和学习环境。随着学生人数的增加和校园设施的扩展，传统的宿舍管理方式已经无法满足现代教育机构对于效率、安全和服务质量的高标准要求。高效的宿舍管理不仅是提升学生居住体验的需求，也是提高教育质量和学校形象的重要因素；它反映了高校从传统管理模式向现代化、信息化管理模式的转变。

目前，许多高校已开始探索建立自己的宿舍管理系统。这些系统的建立，对于提高我国高校宿舍管理的现代化水平起到了积极的推动作用。然而，现有的系统普遍存在信息集成不全、用户交互不友好、安全性和可扩展性不足等问题。针对这些问题，本项目旨在开发一套综合宿舍管理系统，通过集成创新的信息技术和自动化管理工具，旨在解决传统管理中的痛点，提升管理效率，保证信息的安全性和准确性，最终为学生提供一个更安全、便捷、舒适的居住环境，同时为管理人员提供强大的数据支持和操作便利，推动校园管理的信息化进程。

## 3.项目目的：

由于现今的学生公寓管理非常繁琐，通常为手工操作，行政人员付出大量的工作时间，得到的效率很低。因此为提高工作效率，减轻校方人员的工作负担，决定开发学生公寓管理系统软件。

开发此学生公寓系统管理软件，以供教学人员及操作者进行学生宿舍管理，方便操作者随时添加、查询、修改等。

### 关键问题和需求：

系统管理:用户登录、用户密码修改、用户管理、退出系统

公寓管理:楼房管理、宿舍管理、员工管理

出入登记:进楼物品登记、出楼物品登记、人员来访登记、结束访问登记

信息查询:按学号进行查询、按姓名进行查询、按班级查询、按寝室号查询、每栋楼房住宿情况查询

信息修改:学生基本信息修改、员工基本信息修改。

# 二、宿舍管理系统的需求分析

## 1.功能需求：

1. 系统要求用户必须输入正确的用户名和密码才能进入系统；
2. 系统应该提供学生住宿情况的基木登记。
3. 系统应提供人员来访登记及结束访问的详细登记。
4. 系统应提供学生在校期间物品出入宿舍楼的详细情况登记。
5. 系统应提供查询功能，以方便用户对学生基本信息的查询 (要实现按多种条件的查询)及楼房信息的查询。
6. 系统应提供增加、删除、修改用户帐户的功能。

## 2.性能需求：

### 响应时间限制

用户界面响应时间：原则上，所有用户界面的响应时间不得超过3秒，确保用户体验流畅。

严格限制：对于核心操作，如寝室信息查询和宿舍状态更新的响应时间，不得超过2秒。

### 系统处理能力

用户请求处理：系统应能够在正常运行期间每秒至少处理50次用户请求，以满足日常操作需求。

严格限制：系统数据库在高峰时段每秒的事务处理次数不得低于100次，以确保在学期开始或结束等关键时期系统的流畅运行。

### 系统可用性和稳定性

系统恢复：在发生故障时，系统应能在5分钟内自动恢复或切换到备用系统，减少服务中断时间。

运行时间和故障间隔：本系统的正常运行时间应达到99.9%，故障间隔时间应超过1000小时，确保系统的稳定性和可靠性。

### 系统输入项

必要输入：系统应至少能处理以下输入项：用户注册信息、宿舍与公寓管理数据、学生信息、安全与维修记录。

用户反馈和错误报告：应包括用户反馈和错误报告机制，以便及时收集用户建议和系统错误信息，优化系统性能和用户体验。

### 系统输出项

主要输出：本系统的输出项至少包括：用户管理记录、寝室和楼房状态报告、安全和维修更新记录。

数据备份和报告：应提供定期的数据备份和系统状态报告，以供系统管理员和开发团队进行审核和参考，确保数据的安全和系统的持续优化。

## 3.环境需求：

操作简便，界面简洁大方，快捷的信息管理，系统运行稳定，安全可靠，并且视觉上易于理解。

## 4.接口需求：

### 用户信息接口

**用户数据上传：**当用户注册或更新个人信息时，系统需验证数据的完整性和有效性。

**接口需求：**提供一个RESTful API接收用户的注册信息或更新请求，返回用户的数据库记录ID。

### 宿舍信息管理接口

**宿舍数据操作：**

单个宿舍：基础数据操作。

宿舍楼层管理：包含多个宿舍的数据批量操作。

包含详细维修记录或安全检查的宿舍：额外处理相关记录。

**接口需求：**提供一个API来接受宿舍信息的添加、修改、查询和删除请求，返回操作结果和宿舍的数据库记录ID。

### 宿舍查询接口

**按类型查询：**返回指定类型宿舍的所有信息。

**复合查询：**

筛选需要维修的宿舍：标记为维修优先级高。

最近添加或修改的宿舍：在添加或更新后的24小时内标记为新更新。

**接口需求：**提供一个API，允许用户根据宿舍状态、新更新等筛选条件进行搜索，返回宿舍列表。

### 出入登记接口

**常规登记：**普通的访客和物品登记。

**紧急登记（如紧急情况进出）：**记录的处理时间小于1小时，标记为紧急。

**安全登记（使用高级安全监控）**：记录包括安全检查和保安验证。

**接口需求：**提供一个API处理出入登记请求，记录详细的出入时间、物品或人员信息，并在完成登记时更新状态。

### 系统管理接口

**宿舍和用户管理：**

添加、删除宿舍或用户信息。

更新宿舍信息，删除过时的宿舍记录。

**接口需求：**提供管理界面的后端API，处理管理员的请求，返回操作结果和系统管理记录。

## 5.用户界面需求：

对用户登录密码要有安全保护

## 6.数据需求：

为了能够准确的获取数据需求，采用E-R图的形式对需求进行分析和描述。具体分析如下图所示：

|  |
| --- |
| ER.drawio |
| 宿舍管理系统——系统全局E-R图 |

# 三、宿舍管理系统的逻辑模型

## 1.基础应知：

|  |  |
| --- | --- |
|  | 箭头，表示数据流 |
|  | 圆或椭圆，表示变换数据的处理 |
|  | 方框，表示数据的三原点或者终点 |
|  | 双杠或者单杠，表示数据存储文件 |

源点和终点是系统之外的实体，表明数据的来源和去向；

加工是对数据处理的单元，通常为动词短语；

数据流是数据在系统中运动的方向，通常为名词或者名词短语；

数据存储是暂时存储数据的。

## 2.数据流图的生成

### （1）首先确定：

实体：管理员。

加工主要分为：系统管理、公寓管理、出入登记管理、学生管理。

### （2）数据流图：

#### 第0层

|  |
| --- |
| 0层数据流图.drawio |
| 图0 宿舍信息管理系统顶层数据流程图 |

#### 第1层

#### 如下图所示：

#### 对于宿舍信息管理系统进行进一步划分：

|  |
| --- |
| 数据流图.drawio |
| 图1 宿舍信息管理系统1层数据流程图 |

#### 第2层

#### 如下图所示：

#### 主要是对1，2，3，4进行了进一步的划分：

|  |
| --- |
| ****系统管理数据流图****  2.drawio  ****公寓管理数据流图****  2.2.drawio  ****出入登记管理数据流图****  2.3.drawio  ****学生管理数据流图****  2.4.drawio |
| 图2 宿舍信息管理系统2层数据流程图 |

## 3.数据字典

### （1）数据流条目

管理员=[普通用户│超级用户]

事务=[系统事务|公寓事务|出入事务|学生事务]

接收事务=事务命令+事务

系统事务=[修改密码|用户管理]

公寓事件=[楼房管理|员工管理|宿舍管理]

出入事务=[外来人员登记|物品出入登计]

学生事务=[学生信息查询|公寓信息查询]

### （2）数据存储条目

文件名：用户表

组成：用户ID + 用户名 + 密码 + 用户类型

组织方式：索引文件，以用户ID为关键

文件名：学生基本情况表

组成：学生ID + 姓名 + 性别 + 联系电话 + 邮箱 + 专业

组织方式：索引文件，以学生ID为关键

文件名：学生住宿表

组成：记录ID + 学生ID + 寝室ID + 入住日期 + 退宿日期

组织方式：索引文件，以记录ID为关键

文件名：寝室表

组成：寝室ID + 楼房ID + 容纳人数 + 当前住宿人数

组织方式：索引文件，以寝室ID为关键

文件名：员工表

组成：员工ID + 姓名 + 联系电话 + 职位

组织方式：索引文件，以员工ID为关键

文件名：人员来访表

组成：记录ID + 访客姓名 + 访问日期 + 被访问学生ID

组织方式：索引文件，以记录ID为关键

文件名：物品出入表

组成：记录ID + 物品描述 + 进出日期 + 进出类型（进/出）

组织方式：索引文件，以记录ID为关键

文件名：楼房表

组成：楼房ID + 楼房地址 + 楼层数

组织方式：索引文件，以楼房ID为关键

### 数据项

#### 用户表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 描述 | 数据类型 | 数据长度 | NULL | Primarykey |
| UserName | 用户名 | char | 10 | N | Y |
| UseID | 用户密码 | char | 10 | N | Y |
| UserrPower | 用户权限 | char | 20 | N | N |

#### 学生基本情况表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 描述 | 数据类型 | 数据长度 | NULL | Primarykey |
| Stu\_Key | 学号 | char | 20 | N | N and F |
| Stu\_Name | 学生姓名 | char | 10 | Y | N |
| Stu\_Sex | 性别 | char | 10 | Y | N |
| Stu\_Enr | 入学日期 | datetime | 10 | Y | N |
| Stu\_Birth | 出生日期 | datetime | 10 | Y | N |
| Stu\_College | 学院 | char | 10 | Y | N |
| Stu\_Major | 专业 | char | 10 | Y | N |
| Stu\_Polit | 政治面貌 | char | 10 | Y | N |
| Stu\_Address | 家庭住址 | char | 50 | Y | N |
| Stu\_Tel | 联系电话 | char | 10 | Y | N |
| Stu\_Des | 备注 | char | 50 | Y | N |

#### 学生住宿表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 描述 | 数据类型 | 数据长度 | NULL | Primarykey |
| Stu\_Key | 学号 | char | 20 | N | Y and F |
| Stu\_dor | 宿舍 | char | 10 | Y | N |

#### 寝室表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 描述 | 数据类型 | 数据长度 | NULL | Primarykey |
| Dor\_Id | 寝室号 | char | 10 | N | Y |
| Dor\_Sex | 居住性别 | char | 10 | Y | N |
| Dor\_Num | 应住人数 | char | 10 | Y | N |
| Dor\_Fact | 实住人数 | char | 10 | Y | N |
| Dor\_Price | 住宿费用 | char | 10 | Y | N |
| Dor\_Tel | 电话 | char | 10 | Y | N |
| Dor\_Des | 备注 | char | 50 | Y | N |

#### 员工表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 描述 | 数据类型 | 数据长度 | NULL | Primarykey |
| Emp\_Id | 员工号 | char | 20 | N | Y |
| Emp\_Name | 姓名 | char | 10 | Y | N |
| Emp\_Age | 年龄 | char | 10 | Y | N |
| Emp\_Sex | 性别 | char | 10 | Y | N |
| Hou\_Id | 楼房号 | char | 10 | N | F |
| Emp\_Job | 职位 | char | 10 | Y | N |
| Emp\_Add | 家庭住址 | char | 50 | Y | N |
| Emp\_Tel | 联系方式 | char | 10 | Y | N |
| Emp\_Des | 备注 | char | 50 | Y | N |

#### 楼房表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 描述 | 数据类型 | 数据长度 | NULL | Primarykey |
| Hou\_Id | 楼房号 | char | 10 | N | F |
| Hou\_Num | 房间数 | char | 10 | Y | N |
| Hou\_Flr | 楼层数 | char | 10 | Y | N |
| Hou\_Peo | 应住人数 | char | 10 | Y | N |

#### 人员来访表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 描述 | 数据类型 | 数据长度 | NULL | Primarykey |
| Vis\_Name | 来访人姓名 | char | 10 | N | N |
| Vis\_Host | 被访人姓名 | char | 10 | Y | N |
| Dor\_Id | 寝室号 | char | 10 | N | F |
| Relation | 所属关系 | char | 10 | Y | N |
| Vis\_Data | 来访日期 | datetime | 10 | Y | N |
| Vis\_Time | 来访时间 | datetime | 10 | Y | N |
| Lea\_Time | 离开时间 | datetime | 10 | Y | N |
| Vis\_Watch | 值班人 | char | 10 | Y | N |
| Vis\_Cred | 证件名称 | char | 10 | Y | N |
| Vis\_Des | 备注 | char | 50 | Y | N |

#### 物品出入楼表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名 | 描述 | 数据类型 | 数据长度 | NULL | Primarykey |
| Stu\_Name | 学生名 | char | 10 | N | N |
| G\_Id | 物品名 | char | 20 | N | Y |
| G\_In | 进楼物品 | char | 10 | Y | N |
| G\_Out | 出楼物品 | char | 10 | Y | N |
| G\_Date | 时间 | datetime | 10 | Y | N |
| Watch | 值班人 | char | 10 | Y | N |
| G\_Des | 备注 | char | 50 | Y | N |

### 加工条目

#### 1.系统管理

加工名：系统管理

编号：1.1

输入：管理员请求

输出：系统状态

加工逻辑：根据管理员请求调整系统设置和参数

加工名：超级用户登录

编号：1.2

输入：超级用户凭证

输出：用户表（登录状态）

加工逻辑：验证超级用户凭证，授予系统访问权限

加工名：普通用户登录

编号：1.3

输入：用户凭证

输出：用户表（登录状态）

加工逻辑：验证用户凭证，授予相应权限

加工名：修改密码

编号：1.4

输入：旧密码、新密码

输出：用户表（修改状态）

加工逻辑：验证旧密码，成功后更新为新密码

加工名：新用户注册

编号：1.5

输入：用户信息

输出：账户创建状态

加工逻辑：用户表（验证信息无误后创建新用户账户）

#### 2.公寓管理

加工名：事务分类

编号：2.1

输入：事务请求

输出：分类结果

加工逻辑：根据事务类型分配到相应的管理类别

加工名：楼房管理

编号：2.2

输入：楼房数据

输出：楼房表（管理操作结果）

加工逻辑：维护和管理楼房设施和学生信息

加工名：寝室管理

编号：2.3

输入：寝室信息

输出：寝室表（管理操作结果）

加工逻辑：分配寝室，维护寝室状态和学生分配

加工名：员工管理

编号：2.4

输入：员工数据

输出：员工表（管理操作结果）

加工逻辑：招聘、培训、调度和管理员工

#### 3.出入登记管理

加工名：事务分类

编号：3.1

输入：事务请求

输出：分类结果

加工逻辑：根据事务类型分配到相应的管理类别

加工名：人员来访

编号：3.2

输入：访客信息

输出：人员来访表（访问许可）

加工逻辑：验证访客身份和访问目的，授权访问

加工名：访问结束

编号：3.3

输入：访问记录

输出：人员来访表（登记结束）

加工逻辑：记录访问结束时间，更新访问状态

加工名：物品进楼

编号：3.4

输入：物品进楼信息

输出：物品出入表（进楼登记）

加工逻辑：验证物品允许进楼，记录进楼物品

加工名：物品出楼

编号：3.5

输入：物品出楼信息

输出：物品出入表（出楼登记）

加工逻辑：检查物品，记录出楼物品

#### 4.学生管理

加工名：信息分类

编号：4.1

输入：信息请求

输出：分类结果

加工逻辑：将信息按类别进行排序和存储

加工名：信息录入

编号：4.2

输入：学生基本信息

输出：学生基本情况表（录入结果）

加工逻辑：将新信息输入系统，确保数据完整性

加工名：信息修改

编号：4.3

输入：原信息及修改请求

输出：学生基本情况表（更新结果）

加工逻辑：修改现有信息，确保更新后的准确性

加工名：信息查询

编号：4.4

输入：查询请求

输出：查询分类信息

加工逻辑：根据请求提供相应的信息数据

加工名：分类查询

编号：4.5、4.6、4.7、4.8、4.9

输入：分类条件

输出：查询结果

加工逻辑：根据分类条件检索学生基本情况表、学生住宿表、寝室表

# 四、概要设计：

用结构化设计方法从需求分析得到的数据流图导出系统初始结构图：

分成两大部分：

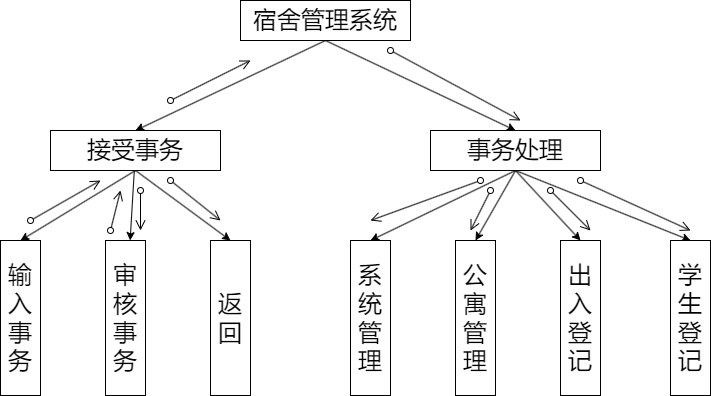
**接受事务部分：**

分为输入、审核、返回三个模块；

**分类处理部分：**

分为系统管理、公寓管理、出入登记、学生管理四个模块；

根据上述得系统初始结构图：



根据模块独立性对事务处理进行细分：

分成四大部分：

**系统管理部分：**

分为系统管理、超级用户登录、普通用户登录、修改密码、新用户注册五个 模块；

**公寓管理部分：**

分为事务分类、楼房管理、寝室管理、员工管理四个模块；

**出入登记部分：**

分为事务分类、人员来访、访问结束、物品进楼、物品出楼五个模块；

**学生管理部分：**

分为信息分类、信息录入、信息修改、信息查询四个模块；

根据上述得细化调整后系统结构图：

|  |
| --- |
| **第一部分数据流图：**  2.drawio |
| **第一部分结构图：**  1图.drawio |
| **第二部分数据流图：**  2.2.drawio |
| **第二部分结构图：**  2图.drawio |
| **第三部分数据流图：**  2.3.drawio |
| **第三部分结构图：**  3图.drawio |
| **第四部分数据流图：**  2.4.drawio |
| **第四部分结构图：**  4图.drawio |

### 1. 全局数据结构定义

#### 用户表（Users）

UserID (int): 唯一标识符

Username (string): 用户名

PasswordHash (string): 密码哈希，确保安全存储

UserType (string): 用户类型（例如："Admin", "SuperUser", "User"）

Status (string): 表示用户的登录状态（例如："Active", "LoggedOut", "Suspended"）

#### 楼房表（Buildings）

BuildingID (int): 唯一标识符

BuildingName (string): 建筑名称

Floors (int): 楼层数量

RoomCount (int): 房间总数

#### 寝室表（DormRooms）

DormID (int): 唯一标识符

BuildingID (int): 对应的楼房ID

Capacity (int): 容纳人数

Occupants ([int]): 当前入住的学生ID列表

#### 学生基本情况表（Students）

StudentID (int): 唯一标识符

Name (string): 学生姓名

DormID (int): 分配的寝室ID

ContactInfo (string): 联系方式

#### 访客记录表（Visitors）

VisitorID (int): 唯一标识符

VisitorName (string): 访客姓名

ArrivalTime (datetime): 来访时间

DepartureTime (datetime): 离开时间

VisitedStudentID (int): 被访问的学生ID

#### 物品出入记录表（ItemLogs）

LogID (int): 唯一标识符

ItemDescription (string): 物品描述

EntryExitTime (datetime): 进出时间

HolderID (int): 物品持有人ID

Direction (string): 进出方向（"Entry", "Exit"）

### 2. 模块接口定义

#### 系统管理接口

##### login(username, password):

输入：用户名（string），密码（string）

输出：登录状态（bool），错误消息（string）

功能：验证用户名和密码，返回登录成功与否。

##### logout(userID):

输入：用户ID（int）

输出：登出结果（bool）

功能：用户登出系统。

##### changePassword(userID, oldPassword, newPassword):

输入：用户ID（int），旧密码（string），新密码（string）

输出：修改结果（bool），错误消息（string）

功能：验证旧密码，若正确则更新密码。

##### registerUser(userInfo):

输入：用户信息（dict）

输出：创建结果（bool），错误消息（string）

功能：验证提供的用户信息并创建新账户。

#### 公寓管理接口

##### addBuilding(buildingData):

输入：建筑数据（dict）

输出：添加结果（bool），错误消息（string）

功能：添加新建筑到系统。

##### updateBuilding(buildingID, newBuildingData):

输入：建筑ID（int），新建筑数据（dict）

输出：更新结果（bool），错误消息（string）

功能：更新现有建筑信息。

##### manageDorm(dormID, newDormData):

输入：寝室ID（int），新寝室数据（dict）

输出：管理结果（bool），错误消息（string）

功能：更新寝室信息。

#### 出入登记管理接口

##### logItemEntry(itemData):

输入：物品数据（dict），包括物品描述、持有人ID

输出：操作结果（bool），错误消息（string）

功能：记录物品进入楼宇的信息，包括物品描述、进入时间及其持有人。

##### logItemExit(itemData):

输入：物品数据（dict），包括物品描述、持有人ID

输出：操作结果（bool），错误消息（string）

功能：记录物品离开楼宇的信息，包括物品描述、离开时间及其持有人。

##### logVisitor(visitorData):

输入：访客信息（dict），包括访客姓名、到访学生ID

输出：操作结果（bool），错误消息（string）

功能：记录访客的来访信息，包括访问时间、访问学生以及访问目的。

##### endVisit(visitorID):

输入：访客ID（int）

输出：操作结果（bool），错误消息（string）

功能：记录访客的离开信息，更新访问状态和离开时间。

#### 学生管理接口

##### addStudent(studentData):

输入：学生数据（dict），包括姓名、联系信息等

输出：操作结果（bool），错误消息（string）

功能：在系统中添加新学生的信息，确保数据的完整性和准确性。

##### updateStudent(studentID, newStudentData):

输入：学生ID（int），新的学生数据（dict）

输出：更新结果（bool），错误消息（string）

功能：更新现有学生的信息，如联系信息、寝室分配等。

##### queryStudent(queryParams):

输入：查询参数（dict），如学生ID、姓名等

输出：学生信息（list of dict），查询错误消息（string）

功能：根据提供的查询条件返回相关学生的详细信息。

##### deleteStudent(studentID):

输入：学生ID（int）

输出：操作结果（bool），错误消息（string）

功能：从系统中删除指定学生的信息。

# 五、详细设计

|  |
| --- |
| 系统管理： 流程图1.drawio 盒图1.drawio 公寓管理： 流程图2.drawio  盒图2.drawio  出入登记：  流程图3.drawio盒图3.drawio 学生管理; 流程图4.drawio  盒图4.drawio |

# 六、系统测试：

测试用例设计的基本目的是确定一组最有可能发现某个错误或某类错误的测试数据。好的测试用例可以在测试过程中重复使用，但不可能测试程序的每一条路径，也不可能把所有的输入数据都试一次。因此，测试用例的设计人员必须努力以最少量的测试用例来发现最大量的可能错误。

|  |
| --- |
| 流程图1.drawio流程图2.drawio流程图3.drawio流程图4.drawio |

## 1.白盒测试：

把测试对象看做一个透明盒子，允许利用程序内部逻辑结构及有关信息，进行测试。通过在不同点检查程序的状态，确定实际的状态是否与预期的状态一致。又称为结构测试或逻辑驱动测试。

完全测试的困难性：对于一个具有多重选择和循环嵌套的程序，不同的路径数目可能是天文数字。

1. 语句覆盖：就是设计若干测试用例，运行被测试程序，使得每一可执行语句至少执行一次。
2. 分支覆盖：就是设计若干个测试用例，运行被测试程序，使得程序中每个判断的取真分支和取假分支至少经历一次，分支覆盖又称为判定覆盖。
3. 条件覆盖：设计若干个测试用例，运行被测程序，使得程序中每个判断的每个条件的可能取值至少执行一次。
4. 条件组合覆盖：设计足够的测试用例，运行被测程序，使得每个判断的所有可能的条件取值组合至少执行一次。
5. 控制流图覆盖测试：将代码变为控制流图（CFG），基于其进行测试的技术。
6. 单条件嵌套：如果判断中的条件表达式是由一个或多个逻辑运算符连接的复合条件表达式，则需要改为一系列只有单个条件的嵌套的判断。
7. 节点、边覆盖（语句覆盖、分支覆盖）。
8. 路径覆盖。

**第一部分：**

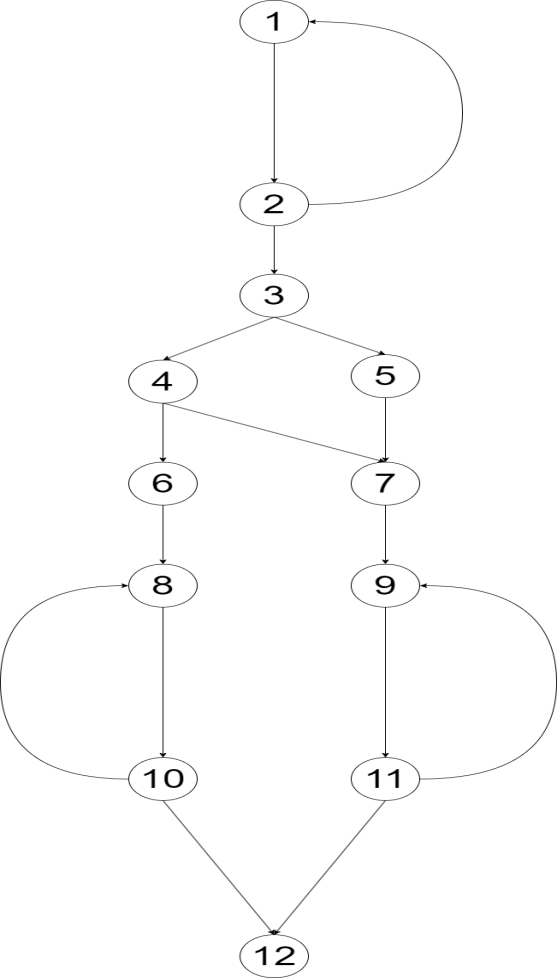
主要对于系统管理进行分析（第二、第三、第四部分同理）

### 语句覆盖：

|  |  |  |  |
| --- | --- | --- | --- |
| **测试用例编号** | **测试用例描述** | **输入** | **结果** |
| 1 | 普通用户成功登录 | 用户名: user 密码: user123 | 凭证验证成功。 授予基本操作权限。 |
| 2 | 超级用户成功登录 | 用户名: admin 密码: admin123 | 凭证验证成功。 授予系统管理权限。 |
| 3 | 用户名正确密码错误 | 用户名: user 密码: wrong123 | 凭证验证失败。 返回错误并提示用户重试。 |
| 4 | 用户名错误密码正确 | 用户名: wronguser 密码: user123 | 凭证验证失败。 返回错误并提示用户重试。 |
| 5 | 用户名和密码均错误 | 用户名: wronguser 密码: wrong123 | 凭证验证失败。 返回错误并提示用户重试。 |
| 6 | 普通用户修改密码成功 | 用户登录，选择修改密码 旧密码: user123 新密码:newpass123 | 旧密码验证成功。 密码更新成功。 |
| 7 | 普通用户修改密码失败 | 用户登录，选择修改密码 旧密码: wrong123 新密码:newpass123 | 旧密码验证失败。 系统不更新密码并提示错误。 |
| 8 | 新用户注册成功 | 新用户注册，输入所有必要且正确的用户信息 | 所有信息验证无误。 账户创建成功。 |
| 9 | 新用户注册信息不完整 | 新用户注册，部分必要信息未填写 | 信息验证发现错误。 系统不创建账户并提示错误。 |
| 10 | 新用户注册信息格式错误 | 新用户注册，电子邮件格式不正确 | 信息验证发现格式错误。 系统不创建账户并提示错误。 |
| 11 | 尝试使用过期的登录凭证登录 | 使用之前登录过但已过期的凭证尝试登录 | 系统识别凭证过期。 返回错误并提示用户重新登录。 |
| 12 | 注销并尝试执行操作 | 用户登录后注销，然后尝试执行修改密码操作 | 系统识别用户已注销。 拒绝执行操作并要求重新登录。 |

**注：分支覆盖、条件覆盖、条件组合覆盖同上；**

### 控制流图覆盖测试

****

### 路径覆盖：

**1、2、3、4、6、8、10、12**

**1、2、3、4、7、9、11、12**

**1、2、3、5、7、9、11、12**

**1、2、1、2、3、4、6、8、10、8、10、12**

**1、2、1、2、3、4、7、9、11、9、11、12**

**1、2、1、2、3、5、7、9、11、9、11、12**

## 2.黑盒测试：

测试对象看做一个黑盒子，测试人员完全不考虑程序内部的逻辑结构和内部特性。只依据程序的需求规格说明书，检查程序的功能是否符合它的功能说明。又叫做功能测试或数据驱动测试。

完全测试的困难性：如果考虑所有可能的输入条件和输出条件中，黑盒测试同样可能是天文数字。

1. 等价类划分：把所有可能的输入数据，即程序的输入域划分成若干部分，然后从每一部分中选取少数有代表性的数据作为测试用例。
2. 边界值分析：对等价类划分方法的补充，大量的错误是发生在输入或输出范围边界上，边界测试可以查出更多的错误。
3. 状态测试：在黑盒测试阶段，通过对状态的测试间接的加以验证功能。

### 系统管理测试：

#### 1.1用户管理

1. 划分等级类并编号

|  |  |  |
| --- | --- | --- |
| 输入条件 | 合理等价类 | 不合理等价类 |
| 用户名长度 | 1、1-10位之间的任何字符 | 2、小于1位字符 |
| 3、大于10位字符 |
| 4、非字符 |
| 密码是否正确 | 5、密码正确 | 6、密码错误 |

1. 设计等价类测试用例

用户输入密码错误或用户未输入密码，则提示用户“输入密码错误，请重试！”，输入密码正确则进入系统。

1. 为合理等价类设计测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 测试数据 | 期望结果 | 实际结果 | 覆盖范围 |
| super，super | 输入有效，进入超级用户管理权限界面 | 与期望结果相符 | 1,5 |
| manage，manager | 输入有效，进入普通用户管理权限界面 | 与期望结果相符 | 1,5 |

1. 为不合理类设计测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 测试数据 | 期望结果 | 实际结果 | 覆盖范围 |
| 未输入密码 | 输入无效 | 与期望结果相符 | 2 |
| 0123456789 | 输入无效 | 与期望结果相符 | 3 |
| ✅ | 输入无效 | 与期望结果相符 | 4 |
| 111 | 输入无效 | 与期望结果相符 | 5 |

#### 1.2修改密码：

用户修改密码时，如果输入的原始密码错误，则会提示“原始密码错误”，如果两次输入的新密码不一样，则提示“两次输入密码不一致”，修改成功后则提示“修改成功”，并退出修改密码界面。

1. 划分等级类并编号

|  |  |  |
| --- | --- | --- |
| 输入条件 | 合理等价类 | 不合理等价类 |
| 原始密码的长度 | 1、1-10位之间的任何字 | 2、小于1位字符 |
| 3、大于10位字符 |
| 原始密码是否正确 | 4、密码正确 | 5、密码错误 |
| 第一次新密码的长度 | 6、1-10位之间的任何字 | 7、小于1位字符 |
| 8、大于10位字符 |
| 第二次新密码的长度 | 9、1-10位之间的任何字 | 10、小于1位字符 |
| 11、大于10位字符 |
| 两次密码是否相同 | 12、相同 | 13、不相同 |

1. 设计等价类测试用例
2. 为合理等价类设计测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 测试数据 | 期望结果 | 实际结果 | 覆盖范围 |
| super,manager,manager | 输入有效 | 与期望结果相符 | 1,4,6,9,12 |

1. 为不合理类设计测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 测试数据 | 期望结果 | 实际结果 | 覆盖范围 |
| 未输入密码 | 输入无效 | 与期望结果相符 | 2 |
| 0123456789 | 输入无效 | 与期望结果相符 | 3 |
| 222 | 输入无效 | 与期望结果相符 | 5 |
| 第一次新密码未输入密码 | 输入无效 | 与期望结果相符 | 7 |
| 98765432100 | 输入无效 | 与期望结果相符 | 8 |
| 第二次新密码未输入密码 | 输入无效 | 与期望结果相符 | 10 |
| 45678912300 | 输入无效 | 与期望结果相符 | 11 |
| 333555 | 输入无效 | 与期望结果相符 | 13 |

### 系统管理测试：

#### 2.1员工信息

1. 划分等级类并编号

|  |  |  |
| --- | --- | --- |
| 输入条件 | 合理等价类 | 不合理等价类 |
| 员工号类型及长度 | 1、3-11位数字字符 | 2、有非数字字符 |
| 3、小于3位数字字符 |
| 4、大于11位数字字符 |
| 姓名类型及长度 | 5、6-11位字符 | 6、有非字符 |
| 7、小于6位的字符 |
| 8、大于11位的字符 |

1. 设计等价类测试用例
2. 为合理等价类设计测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 测试数据 | 期望结果 | 实际结果 | 覆盖范围 |
| 001 | 输入有效 | 与期望结果相符 | 1 |
| 张三 | 输入有效 | 与期望结果相符 | 5 |

1. 为不合理类设计测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 测试数据 | 期望结果 | 实际结果 | 覆盖范围 |
| Aaaa | 输入无效 | 与期望结果相符 | 2 |
| 12 | 输入无效 | 与期望结果相符 | 3 |
| 123456123456 | 输入无效 | 与期望结果相符 | 4 |
| ✅ | 输入无效 | 与期望结果相符 | 6 |
| 李白 | 输入无效 | 与期望结果相符 | 7 |
| 奥林匹克运动会 | 输入无效 | 与期望结果相符 | 8 |

#### 2.2寝室信息

1. 划分等级类并编号

|  |  |  |
| --- | --- | --- |
| 输入条件 | 合理等价类 | 不合理等价类 |
| 寝室号的类型及长度 | 1、1-10位之间的任何字 | 2、有非数字字符 |
| 3、小于10位数字字符 |
| 4、大于10位数字字符 |

1. 设计等价类测试用例
2. 为合理等价类设计测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 测试数据 | 期望结果 | 实际结果 | 覆盖范围 |
| FZZ0506241 | 输入有效 | 与期望结果相符 | 1 |

1. 为不合理类设计测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 测试数据 | 期望结果 | 实际结果 | 覆盖范围 |
| Aaaa | 输入无效 | 与期望结果相符 | 2 |
| 12 | 输入无效 | 与期望结果相符 | 3 |
| 123456123456 | 输入无效 | 与期望结果相符 | 4 |