学生管理系统

**详细设计说明书**

|  |  |
| --- | --- |
| **执 笔 者** | 崔婷婷 |
| **日 期** | 2019.11.5 |
| **指导教师** |  |

二○一九年十二月五日

目录

[1引言 3](#_Toc471214258)

[1.1编写目的 3](#_Toc471214259)

[1.2背景 3](#_Toc471214260)

[1.3基线 3](#_Toc471214261)

[1.4范围 3](#_Toc471214262)

[1.5定义 3](#_Toc471214263)

[1.5.1学生角色功能 3](#_Toc471214264)

[1.5.2辅导员角色功能 3](#_Toc471214265)

[1.5.3管理员角色功能 4](#_Toc471214266)

[1.6参考资料 4](#_Toc471214267)

[2总体设计 4](#_Toc471214268)

[2.1概述 4](#_Toc471214269)

[2.1.1角色描述 4](#_Toc471214270)

[2.2运行环境 4](#_Toc471214271)

[2.2.1软件环境 4](#_Toc471214272)

[2.2.2硬件环境 5](#_Toc471214273)

[2.3开发环境 5](#_Toc471214274)

[2.3.1服务器软件环境 5](#_Toc471214275)

[2.3.2服务器硬件环境 5](#_Toc471214276)

[2.3.3开发机器软件环境 6](#_Toc471214277)

[2.3.4开发机器硬件环境 6](#_Toc471214278)

[2.4设计思想 6](#_Toc471214279)

[2.4.1关于Struts2 6](#_Toc471214280)

[2.4.2关于SpringMVC 7](#_Toc471214281)

[2.4.3关于Hibernate 7](#_Toc471214282)

[2.4.4系统结构 8](#_Toc471214283)

[3系统模块设计 9](#_Toc471214284)

[3.1登录模块 9](#_Toc471214285)

[3.2学生信息修改模块 9](#_Toc471214286)

[3.3班级管理模块 10](#_Toc471214287)

[3.4授权辅导员管理班级模块 11](#_Toc471214288)

[3.5工作信息模块 12](#_Toc471214289)

[3.6角色和菜单管理模块 13](#_Toc471214290)

# 1引言

## 1.1编写目的

本说明书目的在于明确说明考勤管理系统各功能的实现方式，指导开发员进行编码。

本说明书的预期读者为：项目组内成员、其他开发项目组成员、指导老师

## 1.2背景

待开发软件系统的名称：考勤管理系统

此软件系统任务用户：公司员工

## 1.3基线

员工考勤管理系统详细设计。

## 1.4范围

系统包括的范围：员工考勤管理，管理人员管理，时间管理。

## 1.5定义

### 1.5.1员工角色功能

个人信息：指个人的各项信息，如姓名、性别、ID、部门等。

打卡：员工实现打卡功能。

申请功能：可对管理员申请休假、放假，考勤变更等功能。

查询功能：查看自己的通勤记录

### 1.5.2管理员角色功能

个人信息：指个人的各项信息，如姓名、性别、ID、部门等。

答复员工申请：标记员工请、休假申请。

查询：可以查询员工考勤结果。

## 1.6参考资料

1.项目资料：可行性研究报告、需求分析说明书、概要设计说明书。

2.书籍资料：MySQL8.0数据库使用教程、MySQL8.0数据库使用帮助文档、C++中文教程、Qt开发手册、C++高级编程指南、功能结构图。

# 2总体设计

## 2.1概述

### 2.1.1角色描述

本系统分两个个角色：员工、管理员。不同的账号进入不同的页面，完成不同的功能。其中管理员拥有最高权限，可管理专业班级和辅导员，而辅导员可管理学生，学生只对自己的信息有权限。

## 2.2运行环境

### 2.2.1软件环境

表1：软件环境需求

|  |  |  |  |
| --- | --- | --- | --- |
| 分类 | 名称 | 版本 | 语种 |
| 操作系统 | Linux | Ubuntu18.04 | 中文 |
| 数据库平台 | Mysql | 8.0 | 英文 |
| 应用平台 | Tomcat | 8.0 | 英文 |
| 客户端软件 | Google chrome | 49.0 | 英文 |

### 2.2.2硬件环境

表2：硬件环境需求

|  |  |  |
| --- | --- | --- |
| 服务器 | 最低配置 | 推荐配置 |
| 应用服务器和 | Tomcat 8.0 | Apache Tomcat 8.0 |
| 数据库服务器 | Mysql 8.0 | Mysql 5.6 |
|  | PC机 | Linux |

## 2.3开发环境

### 2.3.1服务器软件环境

表3：服务器软件环境

|  |  |  |  |
| --- | --- | --- | --- |
| 分类 | 名称 | 版本 | 语种 |
| 操作系统 | Linux | Ubuntu18.04 | 中文 |
| 数据库平台 | Mysql | 8.0 | 英文 |
| 应用平台 | 计算机 | 任何平台 | 中文 |
| 客户端软件 |  |  |  |

### 2.3.2服务器硬件环境

表4：服务器硬件环境配置表

|  |  |  |
| --- | --- | --- |
| 服务器 | 最低配置 | 推荐配置 |
| 应用服务器 | Tomcat 8.0 | Apache Tomcat 8.0 |
| 数据库服务器 | Mysql 5.0 | Mysql 5.6 |

### 2.3.3开发机器软件环境

表5：开发机器软件配置表

|  |  |  |  |
| --- | --- | --- | --- |
| 分类 | 名称 | 版本 | 语种 |
| 操作系统 | Linux | Ubuntu18.04 | 中文 |
| 数据库平台 | Mysql | 8.0 | 英文 |
| 应用平台 | PC机 |  | 中文 |
| 开发平台 | Qt |  | 英文 |
| 客户端软件 |  |  |  |

### 2.3.4开发机器硬件环境

表6：开发机器硬件配置表

|  |  |  |  |
| --- | --- | --- | --- |
| 分类 | 名称 | 版本 | 语种 |
| 开发机器 | 个人PC机 | 笔记本 | 中文 |
| 操作系统 | Linux | Ubuntu 18.04 | 中文 |

## 2.4设计思想

### 2.4.1关于Struts2

1.Struts2框架： :Struts2对Model，View和Controller都提供了对应的组件。

ActionServlet类是Struts的核心控制器，负责拦截来自用户的请求。

Action类通常由用户提供，该控制器负责接收来自ActionServlet的请求，并根据该请求调用模型的业务逻辑方法处理请求，并将处理结果返回给JSP页面显示。

2.Model部分由ActionForm和JavaBean组成，其中ActionForm用于封装用户的请求参数，封装成ActionForm对象，该对象被ActionServlet转发给Action，Action根据ActionForm里面的请求参数处理用户的请求。JavaBean则封装了底层的业务逻辑，包括数据库访问等。

3.View部分采用JSP实现。Struts提供了丰富的标签库，通过标签库可以减少脚本的使用，自定义的标签库可以实现与Model的有效交互，并增加了现实功能。对应上图的JSP部分。

4.Controller组件有两个部分组成——系统核心控制器，业务逻辑控制器。

系统核心控制器，对应ActionServlet。该控制器由Struts框架提供，继承HttpServlet类，因此可以配置成标注的Servlet。该控制器负责拦截所有的HTTP请求，然后根据用户请求决定是否要转给业务逻辑控制器业务逻辑控制器，负责处理用户请求，本身不具备处理能力，而是调用Model来完成处理。对应Action部分。

### 2.4.2关于SpringMVC

MVC框架是一个全功能的构建 Web应用程序的 MVC 实现。通过策略接口，MVC框架变成为高度可配置的，MVC 容纳了大量视图技术，其中包括 JSP、Velocity、Tiles、iText 和 POI。模型由javabean构成，存放于Map；视图是一个接口，负责显示模型；控制器表示逻辑代码，是Controller的实现。Spring框架的功能可以用在任何J2EE服务器中，大多数功能也适用于不受管理的环境。Spring 的核心要点是：支持不绑定到特定 J2EE服务的可重用业务和数据访问对象。毫无疑问，这样的对象可以在不同J2EE 环境（Web 或EJB）、独立应用程序、测试环境之间重用。

### 2.4.3关于Hibernate

Hibernate的核心接口一共有5个，分别为:Session、SessionFactory、Transaction、Query和Configuration。

1.Session接口　Session接口负责执行被持久化对象的CRUD操作(CRUD的任务是完成与数据库的交流，包含了很多常见的SQL语句。)。但需要注意的是Session对象是非线程安全的。同时，Hibernate的session不同于JSP应用中的HttpSession。这里当使用session这个术语时，其实指的是Hibernate中的session，而以后会将HttpSesion对象称为用session。

2.SessionFactory接口　SessionFactory接口负责初始化Hibernate。它充当数据存储源的代理，并负责创建Session对象。这里用到了工厂模式。需要注意的是SessionFactory并不是轻量级的，因为一般情况下，一个项目通常只需要一个SessionFactory就够，当需要操作多个数据库时，可以为每个数据库指定一个SessionFactory。

3.Configuration接口　Configuration接口负责配置并启动Hibernate，创建SessionFactory对象。在Hibernate的启动的过程中，Configuration类的实例首先定位映射文档位置、读取配置，然后创建SessionFactory对象。

4.Transaction接口　Transaction接口负责事务相关的操作。它是可选的，开发人员也可以设计编写自己的底层事务处理代码。

5.Query和Criteria接口　Query和Criteria接口负责执行各种数据库查询。它可以使用HQL语言或SQL语句两种表达方式。

### 2.4.4系统结构

系统实现结构如图1所示。

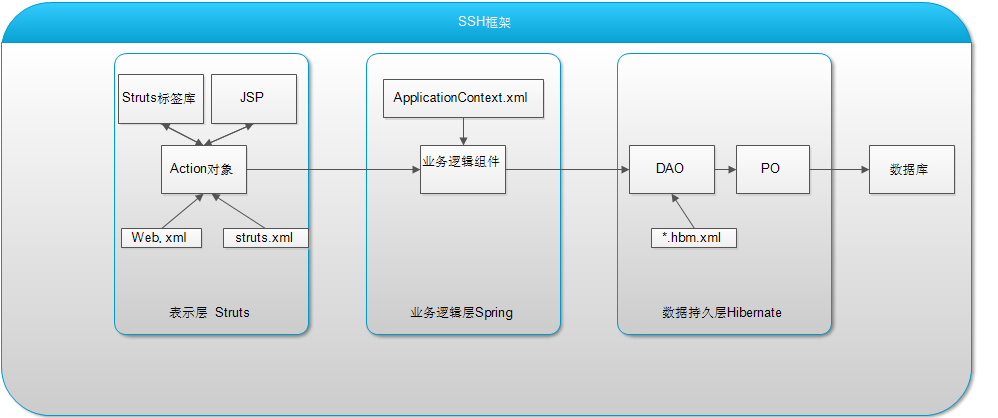


图1系统实现结构图

# 3系统模块设计

## 3.1登录模块

表7：登录模块功能分析表

|  |  |
| --- | --- |
| 模块名称 | 登录模块 |
| 功能描述 | 学生、辅导员、管理员分角色登录，配置过滤器防止其他人恶意跳过登录进入系统。 |
| 子功能模块 | 此模块包含的功能有：用户管理、角色管理。  （1）用户管理对用户信息进行管理，用户信息操作；  （2）角色管理对用户角色进行管理，为系统用户分配角色。 |
| 数据结构  与算法 | （1）用户登录校验：  用户登录系统的过程中，对用户信息进行校验，如判断用户名是否存在，密码和验证码的输入是否正确等。  （2）设置filter：  通过设置filter，可以防止恶意登录，保证系统正常运行。  （3）设置用户信息的操作  登录成功后的用户，进入系统，可以对在权限范围内进行相应功能的操作。 |

## 3.2学生信息修改模块

以学生新修改为例，辅导员信息和管理员信息与此类似。类图见图2学生信息类图。

表8：学生信息修改功能分析表

|  |  |
| --- | --- |
| 模块名称 | 学生信息管理 |
| 功能描述 | 当用户登录以后点击操作用户信息以后，系统将修改的数据存进数据库。 |
| 接口与属性  接口与属性 | Students类中，定义用户的属性。  @Entity  **public** **class** Students {  **private** String id;  **private** String password;  **private** String name;  **private** String sex;  **private** String tel1;  **private** String email;  **private** String qq;  **private** String weChat;  **private** String address;  **private** String parentName;  **private** String parentTel;  **private** String motherName;  **private** String motherTel;  **private** **int** hasChecked;  **private** Classes clas;  **private** List<WorkInfo> workInfos = **new** ArrayList<WorkInfo>(); |
| 数据结构  与算法 | 属性通过get()和set()方法进行数据封装  通过界面按钮，执行提交指令完成对系统数据库中用户信息的修改。 |

|  |
| --- |
| Students |
| +id |
| +password |
| +name |
| +sex |
| +tell |
| +email |
| +qq |
| +weChat |
| +address |
| +parentName |
| +parentTel |
| +motherName |
| +motherTel |
| +hasChecked |
| +clas |
| +workInfos |

图2 学生信息类图

## 3.3班级管理模块

以班级管理为例，专业管理与此类似。类图见图3班级信息类图。

表9：班级管理分析表

|  |  |
| --- | --- |
| 模块名称 | 班级管理 |
| 功能描述 | 管理员为辅导员授权班级，规定了哪位辅导员管理哪个班级。 |
| 接口与属性 | **public** **class** Classes {  **private** **int** id;  **private** String name;  **private** Majors major;  **private** List<Students> students = **new** ArrayList<Students>();  **private** List<Tutors\_classes> cts = **new** ArrayList<Tutors\_classes>(); |
| 数据结构  与算法 | 通过界面按钮，可以添加删除专业班级，同样可以解除辅导员对班级的管理。 |

|  |
| --- |
| Classes |
| +id |
| +name |
| +major |
| +List<students> |
| +List<Tutors\_classes> |

图3 班级信息类图

## 3.4授权辅导员管理班级模块

类图见图4辅导员类和班级类关联关系图。

表10：辅导员功能分析表

|  |  |
| --- | --- |
| 模块名称 | 辅导员信息 |
| 功能描述 | 管理员为辅导员授权班级，规定了哪位辅导员管理哪个班级。 |
| 接口与属性 | **public** **class** Tutors\_classes {  **private** **int** id;  **private** Tutors tutor;  **private** Classes cla;  **private** **int** isNow; |
| 数据结构  与算法 | 通过界面按钮，为辅导员授权班级，同样可以解除辅导员对班级的管理。 |

|  |
| --- |
| Tutor\_classes |
| +id |
| +tutor |
| +cla |
| +isNow |

|  |
| --- |
| Classes |
| +id |
| +name |
| +major |
| +List<students> |
| +List<Tutors\_classes> |

图4 辅导员类和班级类关联关系图

## 3.5工作信息模块

类图见图5工作信息类图。

表11：工作信息提交功能分析表

|  |  |
| --- | --- |
| 模块名称 | 工作信息 |
| 功能描述 | 学生根据自身情况修改和添加工作信息。 |
| 接口与属性 | **public** **class** WorkInfo {  **private** **int** id;  **private** Date startDate;  **private** String companyName;  **private** String address;  **private** String contact;  **private** **float** salary;  **private** Students student;  WorkInfo类通过get()和set()方法对数据进行封装 |
| 数据结构  与算法 | 通过界面按钮，执行操作指令完成对系统数据库中用户信息的操作。  执行的过程是用户界面事件=》Service层=》Dao层操作数据库，从而完成操作。 |

|  |
| --- |
| WorkInfo |
| +id |
| +startDate |
| +companyName  +address  +contact |
| +salary |
| +student |

图5 工作信息类图

## 3.6角色和菜单管理模块

类图见图6角色类和菜单类关联关系图。

表12：角色和菜单功能分析表

|  |  |
| --- | --- |
| 模块名称 | 角色和菜单管理 |
| 功能描述 | 登录时，不同的账号角色，进入不同的页面。 Menus\_Roles 类中，定义用户的属性，同时声明用户关联的菜单。 |
| 接口与属性 | **public** **class** Menus\_Roles {  **private** **int** id;  **private** **int** role\_id;  **private** Menus menu;  **public** **class** Menus {  **private** **int** id;  **private** String name; |
| 数据结构  与算法 | 属性通过get()和set()方法进行数据封装，不同的账号角色，进入不同的页面。 |

|  |
| --- |
| Menus\_Roles |
| +id |
| +role\_id |
| +menu |

|  |
| --- |
| Menus |
| +id |
| +name |

图6 角色类和菜单类关联关系图

教师评语

|  |  |  |  |
| --- | --- | --- | --- |
| 课程设计  评 语 |  | | |
| 课程设计  成 绩 |  | 指导教师  （签字） | 年 月 日 |