武汉纺织大学

Java应用开发课程设计

**基于MyBatis的学生选课管理系统**

**学 院： 数学与计算机学院**

**班 级： 软工11801**

**姓 名： 李旭**

**学 号： 1804240731**

**指导老师： 聂刚**

**成 绩：**

**完成日期： 2020年6月20日**

# 1 需求分析

采用MyBatis框架实现基于DAO设计模式的学生选课管理系统：

用户角色划分：学生、教师、管理员

各角色基本功能如下：

学生角色：登陆、查看所有课程、选课、退选、成绩查询统计、修改密码

教师角色：登陆、查看自己担任的课程信息，查看课程的选课名单、名单导出、输入某课程学生成绩、成绩导出、个人信息维护

管理员角色：登陆、用户管理（教师用户，学生用户）、课程管理。

具体要求如下：

## 1.1用户登陆

当程序运行时，显示“欢迎使用教务管理系统，请登陆”，提示用户依次输入用户名及其密码，当用户名及密码不正确时，显示提示信息“用户名或密码不正确，请重新输入”，**用户最多可以尝试3次，若输入3次均不正确，则给出提示信息“最多只能尝试3次”**，退出系统；

## 1.2功能

**1.2.1增加用户信息**

角色身份为“管理员”的用户进入后，在选择1时，即可添加用户信息，从键盘依次录入信息，输入格式为“用户名，id，密码，角色”，添加完成后提示“信息添加成功！”。

**1.2.2删除用户信息**

角色身份为“管理员”的用户进入后，在选择2时，根据提示“请输入删除的账号id:”输入账号id后，即可完成用户的删除，在删除成功后提示“信息删除成功！”。

**1.2.3增加课程信息**

角色身份为“管理员”的用户进入后，在选择3时，即可增加课程信息，从键盘依次录入信息，输入格式为“教师ID，教师姓名，课程编号，课程名,课程学时”，添加完成后提示“课程添加成功！”。

**1.2.4删除课程信息**

角色身份为“管理员”的用户进入后，在选择4时，根据提示“请输入删除的课程编号:”输入课程编号后，即可完成课程的删除，在删除成功后提示“课程删除成功！”。

**1.2.5返回主菜单**

角色身份为“管理员”的用户进入后，在选择6时，即可返回主菜单并重新进行选择。

**1.3修改账户密码**

当选择修改密码时，显示“请输入当前用户的原密码：”，若原密码不正确，则提示“密码不正确!”，并返回主菜单；若输入正确后，则显示“请输入新的密码:”，输入完成后要求再次确认密码，两次输入一致则密码更改成功，并提示“修改成功!”并返回主菜单。

**1.4退出**

选择退出时，显示“感谢您的使用！”并退出系统。

# 2 系统设计

**2.1用户用例图**

系统分为学生，教师，管理员三个角色，各角色的用户用例如下图2-1所示：

白色的地图

描述已自动生成

图2-1 用户用例图

## 2.2 ER图

数据库设计4张表，用户表，成绩表，课程表，选课记录表，对应的ER图如图所示。

图片包含 白色, 空地, 游戏机, 一群

描述已自动生成

## 2.3 UML类图（Class Diagram）

### 2.3.1 登陆

登录操作需要验证用户名，密码，只有当用户名和密码同时存在即用户存在的情况下才能成功登录系统，否则会提示错误信息，具体流程如图所示。

手机屏幕截图

描述已自动生成

### 2.3.2 查看课程

显示数据库中存在的所有课程信息，用户确认信息后，返回主菜单。

查看课程功能的活动图如图所示。

图片包含 游戏机, 钟表, 标志

描述已自动生成

### 2.2.3 学生选课/退选

显示“请输入选课/退选课程号:”，如果课程号不存在于数据库中，则显示“课程不存在”，如果课程号存在，则将该课程增加至该学生的学生信息中或从学生信息中移除。学生选课/退选功能的活动图如图所示。

手机屏幕的截图

描述已自动生成

### 2.3.4 学生成绩查询

当选择成绩查询时，显示该学生学习信息中的姓名，学号，课程，成绩字段。确认信息后返回主菜单。学生成绩查询的活动图如图所示。

图片包含 游戏机

描述已自动生成

### 2.3.5 修改密码

显示“请输入当前用户的原密码：”，若原密码不正确，则提示“原密码输入不正确，请重新输入”，输入正确后，显示“请设置新的密码：”，输入的密码必须满足必要的复杂性，如果不符合要求，则提示“您的密码不符合复杂性要求（密码长度不少于6个字符，至少有一个小写字母，至少有一个大写字母，至少一个数字），请重新输入：”，输入成功后，显示“请输入确认密码：”，若两次输入不一样，则提示“两次输入的密码必须一致，请重新输入确认密码：”，输入成功后，修改数据库中的用户表，并提示“成功修改密码”，返回主界面。修改密码功能的活动图如图所示。

手机屏幕的截图

描述已自动生成

### 2.3.6 教师查看自己担任课程信息

当用户为教师身份登录时，选择“查看自己所任课程信息”，显示该教师用户信息中的姓名，教师号，课程字段。确认信息后返回主菜单。教师查看自己担任课程信息的活动图如图所示。

手机屏幕的截图

描述已自动生成

### 2.3.7 学生成绩查询

当选择成绩查询时，显示该学生学习信息中的姓名，学号，课程，成绩字段。确认信息后返回主菜单。学生成绩查询的活动图如图所示。

手机屏幕的截图

描述已自动生成

### 2.3.8 学生成绩查询

当选择成绩查询时，显示该学生学习信息中的姓名，学号，课程，成绩字段。确认信息后返回主菜单。学生成绩查询的活动图如图所示。

手机屏幕的截图

描述已自动生成

# 3 系统实现

## 3.1 项目结构

手机屏幕截图

描述已自动生成

## 3.2 配置文件

在项目下新建Source Folder，名为config

### 3.2.1 log4j.properties文件

DBDRIVER=com.mysql.jdbc.Driver

DBURL=jdbc:mysql://localhost:3306/testdb?useUnicode=true&characterEncoding=UTF-8

DBUSER=su

PASSWORD=Java1212

### 3.2.3 mybatis-config.xml文件

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE configuration

PUBLIC "-//mybatis.org//DTD Config 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-config.dtd">

<!-- 根标签 -->

<configuration>

<!-- 指定 资源文件 -->

<properties resource=*"jdbc.properties"*></properties>

<!-- 指定 MyBatis 所用日志的具体实现，未指定时将自动查找-->

<settings>

<setting name=*"logImpl"* value=*"LOG4J"* />

</settings>

<!-- 环境，可以配置多个，default：指定采用哪个环境 -->

<environments default=*"test"*>

<!-- id：唯一标识 -->

<environment id=*"test"*>

<!-- 事务管理器，JDBC类型的事务管理器 -->

<transactionManager type=*"JDBC"* />

<!-- 数据源，池类型的数据源 -->

<dataSource type=*"POOLED"*>

<property name=*"driver"* value=*"${DBDRIVER}"* />

<property name=*"url"* value=*"${DBURL}"* />

<property name=*"username"* value=*"${DBUSER}"* />

<property name=*"password"* value=*"${PASSWORD}"* />

</dataSource>

</environment>

</environments>

<!-- 配置mapper映射文件 -->

<mappers>

<mapper resource=*"dao/mapper/UserMapper.xml"* />

</mappers>

</configuration>

## 3.3 VO类User.java

package vo;

import java.util.Date;

public class User {

private Long id;

private Long usernumber;

private String username;

private String password;

private Long classid;

private Integer credit;

private Integer identity;

private String imageurl;

private Integer sex;

private Date createtime;

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public Long getUsernumber() {

return usernumber;

}

public void setUsernumber(Long usernumber) {

this.usernumber = usernumber;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username == null ? null : username.trim();

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password == null ? null : password.trim();

}

public Long getClassid() {

return classid;

}

public void setClassid(Long classid) {

this.classid = classid;

}

public Integer getCredit() {

return credit;

}

public void setCredit(Integer credit) {

this.credit = credit;

}

public Integer getIdentity() {

return identity;

}

public void setIdentity(Integer identity) {

this.identity = identity;

}

public String getImageurl() {

return imageurl;

}

public void setImageurl(String imageurl) {

this.imageurl = imageurl == null ? null : imageurl.trim();

}

public Integer getSex() {

return sex;

}

public void setSex(Integer sex) {

this.sex = sex;

}

public Date getCreatetime() {

return createtime;

}

public void setCreatetime(Date createtime) {

this.createtime = createtime;

}

@Override

public String toString() {

return "User [id=" + id + ", usernumber=" + usernumber + ", username=" + username + ", password=" + password

+ ", classid=" + classid + ", credit=" + credit + ", identity=" + identity + ", imageurl=" + imageurl

+ ", sex=" + sex + ", createtime=" + createtime + "]";

}

}

3.4 DAO接口类IUserDAO.java

package dao;

import java.util.List;

import java.util.Map;

import vo.User;

public interface UserDao {

int deleteByPrimaryKey(Long id);

int insert(User record);

int insertSelective(User record);

User selectByPrimaryKey(Long id);

int updateByPrimaryKeySelective(User record);

int updateByPrimaryKey(User record);

List<Map<String, Object>> studentsCourseSelection(long courseId);

// 根据学号查询该用户是否存在

public String selectUserByUserNumber(long userNumber);

//验证登录信息

public Map<String, Object> verifyByUser(long userNumber, String password, int identity);

// 修改密码

boolean updatePwd(long id, String password);

//查询所有的用户

List<Map<String, Object>> selectAllUser();

//分页查询所有的用户

Map<String, Object> selectAllUser(int pageNum,int pageSize,String whereSql);

//批量删除用户

boolean deleteByPrimaryKeys(String ids);

//查询用户的数量

Map<String, Object> selectUserNumber();

//根据用户角色获取该角色的所有的用户

List<Map<String, Object>> selectAllUserByIdentity(int identity);

//老师根据条件查询学生列表

Map<String, Object> selectUserByTeacher(int pageNum, int pageSize, String whereSql);

Map<String, Object> selectAllStudent(int pageNum, int pageSize, String whereSql);

//根据学号查询用户

Map<String, Object> selectOneByUserNumber(long userNumber);

// 根据学生id修改学分

boolean updateCreditById(long id, int credit);

}

## 3.5 接口映射文件UserMapper.xml

<?xml version="1.0" encoding="UTF-8" ?>

<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd" >

<mapper namespace="com.bota.dao.UserDao" >

<resultMap id="BaseResultMap" type="com.bota.bean.User" >

<id column="id" property="id" jdbcType="BIGINT" />

<result column="userNumber" property="usernumber" jdbcType="BIGINT" />

<result column="username" property="username" jdbcType="VARCHAR" />

<result column="password" property="password" jdbcType="VARCHAR" />

<result column="classid" property="classid" jdbcType="BIGINT" />

<result column="credit" property="credit" jdbcType="INTEGER" />

<result column="identity" property="identity" jdbcType="INTEGER" />

<result column="imageUrl" property="imageurl" jdbcType="VARCHAR" />

<result column="sex" property="sex" jdbcType="INTEGER" />

<result column="createTime" property="createtime" jdbcType="TIMESTAMP" />

</resultMap>

<sql id="Base\_Column\_List" >

id, userNumber, username, password, classid, credit, identity, imageUrl, sex, createTime

</sql>

<select id="selectByPrimaryKey" resultType="com.bota.bean.User" parameterType="java.lang.Long" >

select

<include refid="Base\_Column\_List" />

from user

where id = #{id,jdbcType=BIGINT}

</select>

<delete id="deleteByPrimaryKey" parameterType="java.lang.Long" >

delete from user

where id = #{id,jdbcType=BIGINT}

</delete>

<insert id="insert" parameterType="com.bota.bean.User" >

insert into user (id, userNumber, username,

password, classid, credit,

identity, imageUrl, sex,

createTime)

values (#{id,jdbcType=BIGINT}, #{usernumber,jdbcType=BIGINT}, #{username,jdbcType=VARCHAR},

#{password,jdbcType=VARCHAR}, #{classid,jdbcType=BIGINT}, #{credit,jdbcType=INTEGER},

#{identity,jdbcType=INTEGER}, #{imageurl,jdbcType=VARCHAR}, #{sex,jdbcType=INTEGER},

#{createtime,jdbcType=TIMESTAMP})

</insert>

<insert id="insertSelective" parameterType="com.bota.bean.User" >

insert into user

<trim prefix="(" suffix=")" suffixOverrides="," >

<if test="id != null" >

id,

</if>

<if test="usernumber != null" >

userNumber,

</if>

<if test="username != null" >

username,

</if>

<if test="password != null" >

password,

</if>

<if test="classid != null" >

classid,

</if>

<if test="credit != null" >

credit,

</if>

<if test="identity != null" >

identity,

</if>

<if test="imageurl != null" >

imageUrl,

</if>

<if test="sex != null" >

sex,

</if>

<if test="createtime != null" >

createTime,

</if>

</trim>

<trim prefix="values (" suffix=")" suffixOverrides="," >

<if test="id != null" >

#{id,jdbcType=BIGINT},

</if>

<if test="usernumber != null" >

#{usernumber,jdbcType=BIGINT},

</if>

<if test="username != null" >

#{username,jdbcType=VARCHAR},

</if>

<if test="password != null" >

#{password,jdbcType=VARCHAR},

</if>

<if test="classid != null" >

#{classid,jdbcType=BIGINT},

</if>

<if test="credit != null" >

#{credit,jdbcType=INTEGER},

</if>

<if test="identity != null" >

#{identity,jdbcType=INTEGER},

</if>

<if test="imageurl != null" >

#{imageurl,jdbcType=VARCHAR},

</if>

<if test="sex != null" >

#{sex,jdbcType=INTEGER},

</if>

<if test="createtime != null" >

#{createtime,jdbcType=TIMESTAMP},

</if>

</trim>

</insert>

<update id="updateByPrimaryKeySelective" parameterType="com.bota.bean.User" >

update user

<set >

<if test="usernumber != null" >

userNumber = #{usernumber,jdbcType=BIGINT},

</if>

<if test="username != null" >

username = #{username,jdbcType=VARCHAR},

</if>

<if test="password != null" >

password = #{password,jdbcType=VARCHAR},

</if>

<if test="classid != null" >

classid = #{classid,jdbcType=BIGINT},

</if>

<if test="credit != null" >

credit = #{credit,jdbcType=INTEGER},

</if>

<if test="identity != null" >

identity = #{identity,jdbcType=INTEGER},

</if>

<if test="imageurl != null" >

imageUrl = #{imageurl,jdbcType=VARCHAR},

</if>

<if test="sex != null" >

sex = #{sex,jdbcType=INTEGER},

</if>

<if test="createtime != null" >

createTime = #{createtime,jdbcType=TIMESTAMP},

</if>

</set>

where id = #{id,jdbcType=BIGINT}

</update>

<update id="updateByPrimaryKey" parameterType="com.bota.bean.User" >

update user

set userNumber = #{usernumber,jdbcType=BIGINT},

username = #{username,jdbcType=VARCHAR},

password = #{password,jdbcType=VARCHAR},

classid = #{classid,jdbcType=BIGINT},

credit = #{credit,jdbcType=INTEGER},

identity = #{identity,jdbcType=INTEGER},

imageUrl = #{imageurl,jdbcType=VARCHAR},

sex = #{sex,jdbcType=INTEGER},

createTime = #{createtime,jdbcType=TIMESTAMP}

where id = #{id,jdbcType=BIGINT}

</update>

</mapper>

# 5 系统总结

主要存在的问题就是对**MyBatis并不是很熟练，环境配置还存在很大问题。还存在很多bug，还有一些功能还没完善。准备假期再重新系统的学习一遍Java，争取把漏补上。**