package com.cqupt.mis.erp.action.registerlogin;

import java.util.HashMap;

import java.util.Map;

import javax.annotation.Resource;

import com.cqupt.mis.erp.manager.tool.CommonDao;

import com.cqupt.mis.erp.model.registerlogin.LoginInfo;

import com.cqupt.mis.erp.service.admin.AdminOperateService;

import com.cqupt.mis.erp.service.registerlogin.GameGroupMemberService;

import com.cqupt.mis.erp.service.registerlogin.LoginService;

import com.cqupt.mis.erp.utils.JSONUtils;

import com.opensymphony.xwork2.ActionContext;

import com.opensymphony.xwork2.ModelDriven;

public class LoginAction implements ModelDriven<LoginInfo> {

@Resource

private LoginService loginService;

@Resource

private CommonDao commonDao;

@Resource

private AdminOperateService adminOperateService;

@Resource

private GameGroupMemberService gameGroupMemberService;

private LoginInfo loginInfo;

// 接收客户端传来的验证码

private String checkcode;

private String status;

/\*\*

\* login 登录操作

\* 教师登录操作作者: 张金胜

\* 用户登录操作作者: 何海源

\*

\*

\*void

\* @exception

\* @since 1.0.0

\*/

public void login() { // TODO 重构

String userId = loginInfo.getUsername();

String password = loginInfo.getPassword();

String serverCode = (String) ActionContext.getContext().getSession().get("SESSION\_SECURITY\_CODE");

Map<String,Object> jsonMap = new HashMap<String,Object>();

if (!checkcode.equals(serverCode)) {

jsonMap.put("resultMessage", "验证码错误");

jsonMap.put("location",

"index.html");

JSONUtils.toJSON(jsonMap);

}

//这里开始教师登录操作.

if("teacher".equals(status)){

Map<String, Object> map = adminOperateService.adminUserLogin(userId,password);

ActionContext.getContext().getSession().put("adminID",userId);

ActionContext.getContext().getSession().put("pwd",password);

if(map.get("status").equals("login success")){

ActionContext.getContext().getSession().put("userUnique", "admin");

jsonMap.put("adminUserName",map.get("adminUserName"));

jsonMap.put("location", "pages/admin/user/user.html");

}else{

jsonMap.put("resultMessage","登陆失败");

}

JSONUtils.toJSON(jsonMap);

return;

}

//这里开始用户登录操作.

int result = loginService.loginForword(userId, password);

if (!serverCode.equals(checkcode)) {

jsonMap.put("resultMessage", "验证码错误");

jsonMap.put("location","index.html");

JSONUtils.toJSON(jsonMap);

}

if (result > -10) {// TODO 为什么是-10？

if (result == -1) {

jsonMap.put("resultMessage", "你的账号不存在");

jsonMap.put("location","pages/registerlogin/welcome\_register.html");

JSONUtils.toJSON(jsonMap);

} else if (result == 2) { // 密码不正确的

int number;

if (ActionContext.getContext().getSession().get("number") == null) {

number = 0;

} else {

number = (Integer) ActionContext.getContext().getSession().get("number");

}

number++;

ActionContext.getContext().getSession().put("number", number);

if (number + 1 > 3) {

ActionContext.getContext().getSession().remove("number");

}

jsonMap.put("resultMessage", "你的密码错误");

jsonMap.put("number", number);//TODO 返回有什么用？

JSONUtils.toJSON(jsonMap);

} else if (result == 6) { // 正在比赛中

String userUnique = commonDao.findUserUniqueByUserId(userId);

String userName = commonDao.findUsernameByUserUnique(userUnique);

String groupName = commonDao.findGameGroupNameByUserUnique(userUnique);

ActionContext.getContext().getSession().put("userUnique", userUnique);

ActionContext.getContext().getSession().put("userId",userId);

ActionContext.getContext().getSession().put("userName", userName);

ActionContext.getContext().getSession().put("groupName", groupName);

jsonMap.put("location", "pages/mainframework/main.html");

JSONUtils.toJSON(jsonMap);

} else if(result == 4){// 破产

String userUnique = commonDao.findUserUniqueByUserId(userId);

ActionContext.getContext().getSession().put("userUnique", userUnique);

jsonMap.put("location", "pages/bankrupt.html");

JSONUtils.toJSON(jsonMap);

} else if(result == 5){ //已经完成了比赛

String userUnique = commonDao.findUserUniqueByUserId(userId);

ActionContext.getContext().getSession().put("userUnique", userUnique);

jsonMap.put("location", "pages/bankrupt.html");

JSONUtils.toJSON(jsonMap);

} else {

// 其他的情况都应该要跳到这个页面里面去.

ActionContext.getContext().getSession().put("userId", userId);

jsonMap.put("location", "pages/registerlogin/useroperate.html");

JSONUtils.toJSON(jsonMap);

}

}else{

System.out.println("进入这里之后什么都不是.");

jsonMap.put("location", "index.html");

JSONUtils.toJSON(jsonMap);

}

}

/\*\*

\*

\* loginForward 登录成功之后应该天赚到相应的页面, 同时将用户信息注册到session 中.

\*void

\* @exception

\* @since 1.0.0

\*/

public void loginForward() {

String userId = (String) ActionContext.getContext().getSession().get("userId");

if (userId != null && !"".equals(userId)) {

String userUnique = commonDao.findUserUniqueByUserId(userId);

ActionContext.getContext().getSession().put("userUnique", userUnique);

}

return;

}

public void signout(){

Map<String, Object> map = new HashMap<>();

String userId = (String) ActionContext.getContext().getSession().get("userId");

if (userId == null || "".equals(userId)) {

map.put("status", 1);

map.put("message", "您还没登陆");

map.put("data", null);

JSONUtils.toJSON(map);

}

ActionContext.getContext().getSession().clear();

map.put("status", 0);

map.put("message", "注销成功");

map.put("data", null);

JSONUtils.toJSON(map);

}

/\*\*

\* showMenuList

\*void

\* @exception

\* @since 1.0.0

\*/

public void showMenuList(){

String userId = (String) ActionContext.getContext().getSession().get("userId");

if (userId != null && !"".equals(userId)) {

int result = loginService.loginStatus(userId);

if(result==3 || result ==4 || result ==5){

ActionContext.getContext().getSession().put("groupName", gameGroupMemberService.findGroupNameByUserId(userId));

}

JSONUtils.toJSON(result);

}

}

/\*\*

\* 清空所有session

\*/

public void clean(){

ActionContext.getContext().getSession().clear();

return ;

}

@Override

public LoginInfo getModel() {

if (loginInfo == null) {

loginInfo = new LoginInfo();

}

return loginInfo;

}

public void setCheckcode(String checkcode) {

this.checkcode = checkcode;

}

public void setStatus(String status) {

this.status = status;

}

}package com.cqupt.mis.erp.action.registerlogin;

import java.io.IOException;

import java.util.HashMap;

import java.util.Map;

import javax.annotation.Resource;

import org.apache.struts2.ServletActionContext;

import com.cqupt.mis.erp.model.registerlogin.ApprovedUserInfo;

import com.cqupt.mis.erp.model.registerlogin.RegisterInfo;

import com.cqupt.mis.erp.service.registerlogin.LoginService;

import com.cqupt.mis.erp.service.registerlogin.RegisterInfoService;

import com.cqupt.mis.erp.utils.JSONUtils;

import com.opensymphony.xwork2.ActionContext;

import com.opensymphony.xwork2.ModelDriven;

public class RegisterInfoAction implements ModelDriven {

private RegisterInfo registerInfo;

private String identifyId;

private String userName;

//这些是冗余字段.

private String confirm;

private String agree;

@Resource

private RegisterInfoService registerInfoService;

@Resource

private LoginService loginService;

/\*\*

\* addRegister 用户注册

\* @return

\*String

\* @exception

\* @since 1.0.0

\*/

public String addRegister(){ // TODO 用message返回

registerInfo.setStatus("等待审批");

if(registerInfoService.addRegister(registerInfo)){

ActionContext.getContext().getSession().put("userId", registerInfo.getUserID());

return "add\_success";

}else

return "back\_registe";

}

/\*\*

\*

\* getRegister 根据登录的用户直接获取用户的详细信息 包括已经通过审批的或者没有通过审批的. 用json直接往前台发送

\*void

\* @exception

\* @since 1.0.0

\*/

public void getRegister(){

String userId = (String) ActionContext.getContext().getSession().get("userId");

if(loginService.isExistsApprover(userId, "")!=-1){

ApprovedUserInfo approvedUser = loginService.findApprovedUserByUserId(userId);

Map<String,Object> map = new HashMap<String,Object>();

map.put("identifyId", "approvedUser");// TODO 用枚举类来表示identifyId更合理吧

map.put("user", approvedUser);

JSONUtils.toJSON(map);

return ;

}else if(loginService.isExistsRegister(userId, "")!=-1){

RegisterInfo registerUser = loginService.findRegisterByUserId(userId);

Map<String,Object> map = new HashMap<String,Object>();

map.put("identifyId", "registerUser");

map.put("user", registerUser);

JSONUtils.toJSON(map);

return ;

}else

return ;//这里就是什么都没有了.

}

/\*\*

\* modifyUser 用户自行修改信息.

\* @return

\*String

\* @exception

\* @since 1.0.0

\*/

public String modifyUser(){

if(identifyId!=null){

if("approvedUser".equals(identifyId)){

registerInfoService.updateModifyApprovedUser(registerInfo);

return "success";

}else if("registerUser".equals(identifyId)){

registerInfoService.updatemodifyRegister(registerInfo);

return "success";

}

}

return "error";

}

/\*\*

\*

\* isExistsUser 在注册的环节里面,判断用户是否存在

\*void

\* @exception

\* @since 1.0.0

\*/

public void isExistsUser(){

String userID = registerInfo.getUserID();

//true：存在; false：是不存在

String existCode ="ture";

if(userID!=null && !"".equals(userID)){

if(loginService.isExistsRegister(userID, "")==-1 && loginService.isExistsApprover(userID, "")==-1){

existCode = "false";

}else

existCode = "true";

}

Map<String,Object> map = new HashMap<String,Object>();

map.put("existCode", existCode);

JSONUtils.toJSON(map);

}

public String getIdentifyId() {

return identifyId;

}

public void setIdentifyId(String identifyId) {

this.identifyId = identifyId;

}

public String getUserName() {

return userName;

}

public void setUserName(String userName) {

this.userName = userName;

}

public void setConfirm(String confirm) {

this.confirm = confirm;

}

public void setAgree(String agree) {

this.agree = agree;

}

@Override

public Object getModel() {

if(registerInfo==null)

registerInfo = new RegisterInfo();

return registerInfo;

}

}

package com.cqupt.mis.erp.action.registerlogin;

import java.io.ByteArrayInputStream;

import java.util.Map;

import org.apache.struts2.interceptor.SessionAware;

import com.cqupt.mis.erp.utils.validator.SecurityCode;

import com.cqupt.mis.erp.utils.validator.SecurityImage;

import com.opensymphony.xwork2.ActionContext;

import com.opensymphony.xwork2.ActionSupport;

/\*\*

\* 提供图片验证码

\*

\* @author 毛家杰

\*/

@SuppressWarnings("serial")

public class SecurityCodeImageAction extends ActionSupport{

//图片流

private ByteArrayInputStream imageStream;

public String execute() throws Exception {

//如果开启Hard模式，可以不区分大小写

//String securityCode = SecurityCode.getSecurityCode(4,SecurityCodeLevel.Hard, false).toLowerCase();

//获取默认难度和长度的验证码

String securityCode = SecurityCode.getSecurityCode();

imageStream = SecurityImage.getImageAsInputStream(securityCode);

//放入session中

ActionContext.getContext().getSession().put("SESSION\_SECURITY\_CODE", securityCode);

return SUCCESS;

}

public ByteArrayInputStream getImageStream() {

return imageStream;

}

public void setImageStream(ByteArrayInputStream imageStream) {

this.imageStream = imageStream;

}

}

package com.cqupt.mis.erp.action.registerlogin;

import java.util.List;

import javax.annotation.Resource;

import javax.servlet.ServletContext;

import org.apache.struts2.ServletActionContext;

import org.apache.struts2.util.ServletContextAware;

import org.directwebremoting.WebContextFactory;

import org.springframework.web.context.support.ServletContextFactoryBean;

import com.cqupt.mis.erp.model.registerlogin.GameGroupInfo;

import com.cqupt.mis.erp.service.inigame.IniGameService;

import com.cqupt.mis.erp.service.registerlogin.GameGroupMemberService;

import com.cqupt.mis.erp.service.registerlogin.GameGroupService;

import com.cqupt.mis.erp.utils.JSONUtils;

import com.opensymphony.xwork2.ActionContext;

import com.opensymphony.xwork2.ModelDriven;

public class GameGroupAction implements ModelDriven<GameGroupInfo>{

@Resource

private GameGroupService gameGroupService;

@Resource

private GameGroupMemberService gameGroupMemberService;

@Resource

private IniGameService iniGameService;

private GameGroupInfo gameGroupInfo;

/\*\*

\*

\* addGameGroup 新建一个gamegroup

\* @return

\*String

\* @exception

\* @since 1.0.0

\*/

public String addGameGroup(){ //TODO return 统一用一个message

String groupCreaterId = (String) ActionContext.getContext().getSession().get("userId");

try {

gameGroupInfo.setGroupCreaterId(groupCreaterId);

boolean result = gameGroupService.addGameGroup(gameGroupInfo);

if(result){

ActionContext.getContext().getSession().put("groupName", gameGroupInfo.getGroupName());

}else{

return "error";

}

} catch (Exception e) {

//e.printStackTrace();

System.out.println("重复组");

return "error";

}

return "admin\_member\_manager";

}

/\*\*

\*

\* getGameGroupMessage 获取分组信息 然后直接展示, 满人的不展示,已经开始的也不会展示.

\*void

\* @exception

\* @since 1.0.0

\*/

public void getGameGroupMessage(){

String groupName = (String)ActionContext.getContext().getSession().get("groupName");

GameGroupInfo gameGroupMessage = null;

if(groupName==null || "".equals(groupName)){

String userId = (String)ActionContext.getContext().getSession().get("userId");

groupName = gameGroupMemberService.findGroupNameByUserId(userId);

ActionContext.getContext().getSession().put("groupName", groupName);

}

gameGroupMessage = gameGroupService.findGameGroupInfoByGroupName(groupName);

JSONUtils.toJSON(gameGroupMessage);

}

/\*\*

\* deleteGameGroup 管理员删除分组

\*void

\* @exception

\* @since 1.0.0

\*/

public void deleteGameGroup(){

String groupName = (String)ActionContext.getContext().getSession().get("groupName");

if(groupName!=null && !groupName.equals("")){

if(!gameGroupService.deleteGameGroupByGroupName(groupName))

return ;//FIXME ：这里应该是报错的

}

}

public void showGameGroupList(){ //TODO 用gameGroupInfo来代替g

List<GameGroupInfo> g = gameGroupService.showGameGroupList(ServletActionContext.getRequest().getSession());

JSONUtils.toJSON(g);

}

/\*\*

\*

\* addGameGroupInput 跳转到相应的表单页面.

\* @return

\*String

\* @exception

\* @since 1.0.0

\*/

public String addGameGroupInput(){//TODO return 统一用一个message

//在添加分组之前最好判断一下这个人有没有已经加入了其他分组了.有的话自动跳转.没有才让他过来添加

return "add\_input";

}

/\*\*

\*

\* startGame 开始比赛的按钮按下之后,首先让所有人都跳转到 相应的html

\* 然后阻塞在相应的html中,等到唤醒跳转的操作.

\*

\* 初始化成功之后就会唤醒所有人了.

\*

\*void

\* @exception

\* @since 1.0.0

\*/

public synchronized void startGame(){

String groupName = (String)ActionContext.getContext().getSession().get("groupName");

//System.out.println("stat game--"+groupName);

gameGroupMemberService.startGameForward(groupName);

try {

GameGroupInfo g = gameGroupService.findGameGroupInfoByGroupName(groupName);

if(g.getCurrentPeriod()>0){

return ;

}

boolean result = iniGameService.iniGame(groupName);

//Thread.currentThread().sleep(2000); //这个秒数可能要影响加载的问题. 可能后面要根据问题来解决

if(result){

gameGroupMemberService.startGameForwardToMain(groupName);

}else {

System.out.println("初始化失败");

}

} catch (Exception e) {

e.printStackTrace();

}

return;

}

/\*\*

\* isGameGroupRunning 判断游戏组是否正在进行.如果开始了 就直接返回 true

\*void

\* @exception

\* @since 1.0.0

\*/

public void isGameGroupRunning(){

String groupName = (String)ActionContext.getContext().getSession().get("groupName");

//System.out.println(groupName);

if(groupName != null && !"".equals(groupName)){

GameGroupInfo g = gameGroupService.findGameGroupInfoByGroupName(groupName);

if(g == null){

JSONUtils.toJSON(false);

return;

}

if(g.getCurrentPeriod()>0){

JSONUtils.toJSON(true);

System.out.print("go");

}

}

JSONUtils.toJSON(false);

}

/\*\*

\* isGameGroupExist 判断这个小组是否存在

\*void

\* @exception

\* @since 1.0.0

\*/

public void isGameGroupExist(){

GameGroupInfo g = gameGroupService.findGameGroupInfoByGroupName(gameGroupInfo.getGroupName());

if(g != null){

JSONUtils.toJSON(false);

}

JSONUtils.toJSON(true);

}

/\*\*

\* void

\* 前端页面用来判断是否已经登录&session 是否过期

\*/

public void isLogin() {

String userId = (String)ActionContext.getContext().getSession().get("userId");

if(userId == null || "".equals(userId)){

JSONUtils.toJSON(false);

}

JSONUtils.toJSON(true);

}

@Override

public GameGroupInfo getModel() {

if(gameGroupInfo==null){

gameGroupInfo = new GameGroupInfo();

}

return gameGroupInfo;

}

}package com.cqupt.mis.erp.action.registerlogin;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.annotation.Resource;

import com.cqupt.mis.erp.manager.registerlogin.GameGroupMemberDao;

import com.cqupt.mis.erp.model.registerlogin.ApprovedUserInfo;

import com.cqupt.mis.erp.service.advertisement.AdvertisementService;

import com.cqupt.mis.erp.service.forwardquarter.ForwardQuarterService;

import com.cqupt.mis.erp.service.registerlogin.GameGroupMemberService;

import com.cqupt.mis.erp.service.registerlogin.LoginService;

import com.cqupt.mis.erp.utils.JSONUtils;

import com.opensymphony.xwork2.ActionContext;

/\*\*

\*

\* GameGroupMemberAction

\* @author heinz\_ho

\* @version 1.0.0

\*

\*/

public class GameGroupMemberAction {

private String gameGroupName;

private String userID;

@Resource

private GameGroupMemberService gameGroupMemberService;

@Resource

private GameGroupMemberDao gameGroupMemberDao;

@Resource

private LoginService loginService;

@Resource

private ForwardQuarterService forwardQuarterService;

@Resource

private AdvertisementService advertisementService;

/\*\*

\* joinGroup 用户在加入分组页中点击加号, 加入当前分组.

\*void

\* @exception

\* @since 1.0.0

\*/

public void joinGroup(){

//System.out.println(gameGroupName);

String userId = (String) ActionContext.getContext().getSession().get("userId");

if(gameGroupName != null && userId != null && !"".equals(userId)){

if(!loginService.isGroupMember(userId) && gameGroupMemberService.addJoinInGroup(gameGroupName,userId)){

//成功添加

ActionContext.getContext().getSession().put("groupName",gameGroupName);

JSONUtils.toJSON(true);

}else{

//添加成员不成功

JSONUtils.toJSON(false);

}

}

}

/\*\*

\* showGameGroupMemberListAJAX 进入房间之后立马加载一次或者有新人新人进来之后立马加载.

\* 与在service的方法不一样, 这个是主动获取的, 可以避免推送出现的故障

\*void

\* @exception

\* @since 1.0.0

\*/

public void showGameGroupMemberListAJAX(){

String groupName = (String)ActionContext.getContext().getSession().get("groupName");

List<ApprovedUserInfo> members = gameGroupMemberService.findGameGroupUserMessageByGroupName(groupName);

JSONUtils.toJSON(members);

}

/\*\*

\* exitGroup 用户主动退出小组.

\*void

\* @exception

\* @since 1.0.0

\*/

public void exitGroup(){

String userId = (String) ActionContext.getContext().getSession().get("userId");

String groupName = (String)ActionContext.getContext().getSession().get("groupName");

try {

if(userId!=null && groupName !=null){

gameGroupMemberService.deleteExitGroup(groupName,userId);

ActionContext.getContext().getSession().remove("groupName");

}

} catch (Exception e) {

e.printStackTrace();

return ;

}

JSONUtils.toJSON(userId);

return ;

}

/\*\*

\* adminDeleteGroupMember 管理员剔除用户

\*void

\* @exception

\* @since 1.0.0

\*/

public void adminDeleteGroupMember(){

String groupName = (String)ActionContext.getContext().getSession().get("groupName");

if(userID!=null && !userID.equals("")){

String [] userIDs = userID.split(",");

for(String userId: userIDs){

gameGroupMemberService.deleteExitGroup(groupName,userId);

}

gameGroupMemberService.exitGroupReload(groupName);

}

return;

}

/\*\*

\* exitGame 退出游戏

\*void

\* @exception

\* @since 1.0.0

\*/

public void exitGame(){

//为了能够有记录, 这里不能删除这些人的信息. 但是要跟破产的处理方式一样

String groupName = (String)ActionContext.getContext().getSession().get("groupName");

String userUnique = (String)ActionContext.getContext().getSession().get("userUnique");

//完成这里的所有投放广告.

advertisementService.updateAdvertisementFinish(userUnique);

forwardQuarterService.bankruptcy(userUnique);

gameGroupMemberDao.updateBankruptcyUserStatus(userUnique, -1);

gameGroupMemberService.updateGameGroupMemberNumber(groupName);

gameGroupMemberDao.updateBankruptcyUserStatus(userUnique, -1);

ActionContext.getContext().getSession().clear();

return ;

}

public void replay(){

String userUnique = (String)ActionContext.getContext().getSession().get("userUnique");

gameGroupMemberDao.updateBankruptcyUserStatus(userUnique, -1);

//System.out.println(userUnique);

}

public void setGameGroupName(String gameGroupName) {

this.gameGroupName = gameGroupName;

}

public void setUserID(String userID) {

this.userID = userID;

}

}

package com.cqupt.mis.erp.action.admin;

import java.util.HashMap;

import java.util.Map;

import javax.annotation.Resource;

import com.cqupt.mis.erp.service.admin.AdminOperateService;

import com.cqupt.mis.erp.utils.JSONUtils;

import com.opensymphony.xwork2.ActionContext;

/\*\*

\*

\* @author zjs

\*

\*/

public class AdminOperateAction {

@Resource

private AdminOperateService adminOperateService;

private String adminID;

private String oldPWD;

private String newPWD;

private String pwd;

// 管理员登陆

public void adminUserLogin() {

adminID = (String) ActionContext.getContext().getSession().get("adminID");

pwd = (String)ActionContext.getContext().getSession().get("pwd");

Map<String, Object> map = adminOperateService.adminUserLogin(adminID,pwd);

JSONUtils.toJSON(map);

}

// 修改密码

public void updatePWD() {

boolean update = adminOperateService.updatePWD(adminID, oldPWD, newPWD);

String result;

Map<String, Object> map = new HashMap<String, Object>();

if(update){

result = "修改成功";

map.put("code", 0);

}else{

result = "修改失败";

map.put("code", 1);

}

map.put("result", result);

JSONUtils.toJSON(map);

}

public void adminLogout(){

ActionContext.getContext().getSession().clear();

JSONUtils.toJSON("success");

}

public void setAdminName(String adminID) {

this.adminID = adminID;

}

public void setOldPWD(String oldPWD) {

this.oldPWD = oldPWD;

}

public void setNewPWD(String newPWD) {

this.newPWD = newPWD;

}

public void setPwd(String pwd) {

this.pwd = pwd;

}

}

package com.cqupt.mis.erp.action.admin;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.annotation.Resource;

import org.apache.struts2.ServletActionContext;

import com.cqupt.mis.erp.model.registerlogin.ApprovedUserInfo;

import com.cqupt.mis.erp.model.registerlogin.RegisterInfo;

import com.cqupt.mis.erp.service.admin.ApprovedUserService;

import com.cqupt.mis.erp.service.registerlogin.LoginService;

import com.cqupt.mis.erp.service.registerlogin.RegisterInfoService;

import com.cqupt.mis.erp.utils.JSONUtils;

/\*\*

\*

\* @author zjs

\*

\*/

public class UserManagerAction {

@Resource

private ApprovedUserService approvedUserService;

@Resource

private RegisterInfoService registerInfoService;

@Resource

private LoginService loginService;

private String userId;

private String[] userIds;

private ApprovedUserInfo approvedUserInfo;

private String userID;

private String name;

private String major;

private String className;

private String studentID;

// 取出待审批用户

public void findAllRegister() {

List<RegisterInfo> registerInfos = registerInfoService

.findAllRegister();

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("user", registerInfos);

JSONUtils.toJSON(map);

}

// 取出审批用户

public void findAllApprovedUser() {

List<ApprovedUserInfo> ApprovedUserInfos = approvedUserService

.showApprovedUserList();

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("user", ApprovedUserInfos);

JSONUtils.toJSON(map);

}

// 通过审批

public void passRegisterUser() {

String status;

boolean result = approvedUserService.updatePassUser(userId);

if (result) {

status = userId + "用户通过审批";

} else {

status = userId + "用户未通过审批";

}

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("status", status);

JSONUtils.toJSON(map);

}

// 批量审批,接收一个要审批用户的id数组，遍历

public void passBatchRegisterUsers() {

String[] userIds = ServletActionContext.getRequest().getParameterValues("userIds[]");

String status;

for (int i = 0; i < userIds.length; i++) {

approvedUserService.updatePassUser(userIds[i]);

}

status = userIds + "用户通过审批";

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("status", status);

JSONUtils.toJSON(map);

}

// 修改审批用户信息

public void updateApprovedUserInfo() {

approvedUserInfo = loginService.findApprovedUserByUserId(userID);

approvedUserInfo.setClassName(className);

approvedUserInfo.setMajor(major);

approvedUserInfo.setName(name);

approvedUserInfo.setStudentID(studentID);

approvedUserInfo.setUserID(userID);

approvedUserService.updateApprovedUserInfo(approvedUserInfo);

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("status", approvedUserInfo.getUserID() + "用户更新信息成功");

JSONUtils.toJSON(map);

}

// 删除审批用户

public void deleteApprUser() {

approvedUserService.deleteApprovedUserById(userId);

// TODO 为什么 没有返回？

}

// 批量删除,接收一个要删除审批用户的id数组，遍历

public void deleteBatchApprUsers() {

String[] userIds = ServletActionContext.getRequest().getParameterValues("userIds[]");

Map<String, Object> map = new HashMap<String, Object>();

for (int i = 0; i < userIds.length; i++) {

boolean result = approvedUserService.deleteApprovedUserById(userIds[i]);

if(!result){

map.put("code", 1);

}else{

map.put("code", 0);

}

}

map.put("status", userIds + "用户成功删除");

JSONUtils.toJSON(map);

}

// 删除待审批用户

public void deleteRegiUser() {

registerInfoService.deleteRegisterById(userId);

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("status", userId + "待审批用户成功删除");

JSONUtils.toJSON(map);

}

// 批量删除,接收一个要删除待审批用户的id数组，遍历

public void deleteBatchRegiUsers() {

String[] userIds = ServletActionContext.getRequest().getParameterValues("userIds[]");

for (int i = 0; i < userIds.length; i++) {

registerInfoService.deleteRegisterById(userIds[i]);

}

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("status", userIds + "待审批用户成功删除");

JSONUtils.toJSON(map);

}

// 获取前台要待审批用户的id

public void setUserId(String userId) {

this.userId = userId;

}

// 获取前台要批量待操作用户的ids

public void setUserIds(String[] userIds) {

this.userIds = userIds;

}

// 获取要修改的用户信息

public void setApprovedUserInfo(ApprovedUserInfo approvedUserInfo) {

this.approvedUserInfo = approvedUserInfo;

}

public void setUserID(String userID) {

this.userID = userID;

}

public void setName(String name) {

this.name = name;

}

public void setMajor(String major) {

this.major = major;

}

public void setClassName(String className) {

this.className = className;

}

public void setStudentID(String studentID) {

this.studentID = studentID;

}

}

package com.cqupt.mis.erp.action.admin;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.annotation.Resource;

import com.cqupt.mis.erp.manager.registerlogin.GameGroupMemberDao;

import com.cqupt.mis.erp.model.registerlogin.GameGroupInfo;

import com.cqupt.mis.erp.model.registerlogin.GameGroupMemberStatus;

import com.cqupt.mis.erp.service.advertisement.AdvertisementService;

import com.cqupt.mis.erp.service.forwardquarter.ForwardQuarterService;

import com.cqupt.mis.erp.service.order.ChooseOrderService;

import com.cqupt.mis.erp.service.registerlogin.GameGroupService;

import com.cqupt.mis.erp.utils.JSONUtils;

/\*\*

\*

\* @author zjs

\*

\*/

public class GameGroupManagerAction {

@Resource

private GameGroupService gameGroupService;

@Resource

private ForwardQuarterService forwardQuarterService;

@Resource

private AdvertisementService advertisementService;

@Resource

private GameGroupMemberDao gameGroupMemberDao;

@Resource

private ChooseOrderService chooseOrderService;

private String groupName;

private String userUnique;

private String period;

// 展示游戏组列表

public void showGameGroups() {

List<GameGroupInfo> GameGroupInfos = gameGroupService.showGameGroups();

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("GameGroups", GameGroupInfos);

JSONUtils.toJSON(map);

}

// 删除游戏组

public void deteleGameGroup() {

boolean result = gameGroupService.delete\_GameGroupByGroupName(groupName);

Map<String, Object> map = new HashMap<String, Object>();

if(result){

map.put("code", 0);

}else{

map.put("code", 1);

}

JSONUtils.toJSON(map);

}

// 连表查询用户基本信息和游戏状态信息

public void findGameGroupMemberStatusByGroupName() {

List<GameGroupMemberStatus> GameGroupMemberStatuss = gameGroupService.findGameGroupMemberStatusByGroupName(groupName);

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("GameGroupMemberStatuss", GameGroupMemberStatuss);

JSONUtils.toJSON(map);

}

// 结束经营

public void endPlayGame() {

int result = gameGroupService.endPlayGame(userUnique, groupName);

String results;

Map<String, Object> map = new HashMap<String, Object>();

if (result == 1) {

results = "结束经营成功";

map.put("code", 0);

}else{

results = "结束经营失败";

map.put("code", 1);

}

map.put("result", results);

JSONUtils.toJSON(map);

}

// 结束投广告

public void endAdvertising() {

int code = advertisementService.updateAdvertisementFinish(userUnique);

String result;

Map<String, Object> map = new HashMap<String, Object>();

if(code==1){

result="投放的钱太多.不能投了";

map.put("code", 0);

}else if(code==2){

result="投放成功";

map.put("code", 0);

}else{

result="已经进行过结束广告费投放操作";

map.put("code", 1);

}

map.put("result", result);

JSONUtils.toJSON(map);

}

// 结束选订单

public void endChooseOrder(){

int code = chooseOrderService.updateEndChooseOrder(userUnique);

String result;

Map<String, Object> map = new HashMap<String, Object>();

if(code==1){

result="结束选订单成功";

map.put("code", 0);

}else if(code==2){

result="已经完成这一期的选择订单的操作";

map.put("code", 1);

}else{

result="没有完成投放广告";

map.put("code", 1);

}

map.put("result", result);

JSONUtils.toJSON(map);

}

// 推进下一周期

public void ForwarPeriod(){

Integer status = forwardQuarterService.ForwardStatus(userUnique);

Map<String, Object> map = new HashMap<String, Object>();

map.put("code",status);

JSONUtils.toJSON(map);

/\*\*boolean result = gameGroupMemberDao.updateBankruptcyUserStatus(userUnique);

String results;

if(result){

results = "推进下一周期成功";

}else{

results = "推进下一周期失败";

}

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("result", results);

JSONUtils.toJSON(map);\*/

}

/\*\*

\* changeToHistory

\* 将小组的记录转成历史记录,同时删除记录内容

\*void

\* @exception

\* @since 1.0.0

\*/

public void changeToHistory(){

System.out.println(groupName);

boolean result = gameGroupService.addChangeToHistory(groupName);

Map<String, Object> map = new HashMap<String, Object>();

if(result){

map.put("code", 0);

}else{

map.put("code", 1);

}

JSONUtils.toJSON(map);

}

public void setGroupName(String groupName) {

this.groupName = groupName;

}

public void setUserUnique(String userUnique) {

this.userUnique = userUnique;

}

public void setPeriod(String period) {

this.period = period;

}

}

package com.cqupt.mis.erp.action.admin;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.annotation.Resource;

import com.cqupt.mis.erp.model.enterpriseevaluate.MemberCost;

import com.cqupt.mis.erp.model.enterpriseevaluate.MemberSaleOfMarket;

import com.cqupt.mis.erp.model.enterpriseevaluate.MemberSaleOfProduct;

import com.cqupt.mis.erp.model.enterpriseevaluate.ProduceCapacity;

import com.cqupt.mis.erp.model.enterpriseevaluate.ProductProfit;

import com.cqupt.mis.erp.service.enterpriseevaluate.EnterPriseEvaluateService;

import com.cqupt.mis.erp.utils.JSONUtils;

/\*\*

\*

\* @author zjs

\*

\*/

public class EnterpriseEvaluateAction {

@Resource

private EnterPriseEvaluateService enterpriseEvaluateService;

private String groupName;

private String year;

private String period;

private String userunique;

public void setGroupName(String groupName) {

this.groupName = groupName;

}

public void setYear(String year) {

this.year = year;

}

public void setUserunique(String userunique) {

this.userunique = userunique;

}

public void setPeriod(String period) {

this.period = period;

}

//根据游戏组名获得各小组信息

public void showGroupMembers(){

List<Object> list = enterpriseEvaluateService.showGroupMembers();

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("allGroupMembers", list);

JSONUtils.toJSON(map);

}

// 查看所有者权益

public void showEndValue() {

List<Object> endValues = enterpriseEvaluateService.showEndValue(year, groupName,userunique);

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("endValues", endValues);

JSONUtils.toJSON(map);

}

// 查看所有者权益(新)

public void showEndValues() {

List<Object> endValues = enterpriseEvaluateService.showEndValues(year,period, groupName);

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("endValues", endValues);

JSONUtils.toJSON(map);

}

// 广告投入产出分析

public void getUserIORatesOfAd() {

List<Map<String, Object>> userIORates = enterpriseEvaluateService.getUserIORatesOfAd(groupName, year);

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("userIORates", userIORates);

JSONUtils.toJSON(map);

}

// 综合市场占有率分析

public void getGeneralMarketShare() {

List<MemberSaleOfMarket> memberSaleOfMarkets = enterpriseEvaluateService.getGeneralMarketShare(groupName, year);

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("memberSaleOfMarkets", memberSaleOfMarkets);

JSONUtils.toJSON(map);

}

// 产品市场占有率分析

public void getProductMarketShare() {

List<MemberSaleOfProduct> memberSaleOfProducts = enterpriseEvaluateService.getProductMarketShare(groupName, year);

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("memberSaleOfProducts", memberSaleOfProducts);

JSONUtils.toJSON(map);

}

// 成本费用占销售比例分析

public void getCostStructure() {

List<MemberCost> memberCosts = enterpriseEvaluateService.getCostStructure(groupName, year);

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("memberCosts", memberCosts);

JSONUtils.toJSON(map);

}

// 成本费用占销售比例变化分析

public void getCostStructureChanges() {

List<MemberCost> memberCosts = enterpriseEvaluateService.getCostStructureChanges(groupName,userunique);

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("memberCosts", memberCosts);

JSONUtils.toJSON(map);

}

// 产品贡献利润分析

public void getProductsProfit() {

List<ProductProfit> productProfits = enterpriseEvaluateService.getProductsProfit(groupName, year);

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("productProfits", productProfits);

JSONUtils.toJSON(map);

}

// 生产能力分析

public void getMembersCapacity() {

List<ProduceCapacity> produceCapacities = enterpriseEvaluateService.getMembersCapacity(groupName);

Map<String, Object> map = new HashMap<String, Object>();

map.put("code", 0);

map.put("produceCapacities", produceCapacities);

JSONUtils.toJSON(map);

}

}

package com.cqupt.mis.erp.manager.factory;

import java.util.List;

import com.cqupt.mis.erp.model.factory.FactoryBasicInfo;

import com.cqupt.mis.erp.model.factory.FactoryCommonInfo;

public interface FactoryMakingDao {

/\*\*

\* 获取“显示管理在建厂房”要显示的数据信息

\*

\* @author zy

\* @param userUnique

\* @return List<FactoryCommonInfo>

\*/

public List<FactoryCommonInfo> findFactoryMakings(String userUnique);

/\*\*

\* findFactoryMakings 根据市场 来加载正在修建的factory

\* @param userUnique

\* @param marketName

\* @return

\*List<FactoryCommonInfo>

\* @exception

\* @author hhy

\*/

public List<FactoryCommonInfo> findFactoryMakings(String userUnique,String marketName);

/\*\*

\* 查看在建厂房的明细信息

\*

\* @author zy

\* @param userUnique

\* ,factoryId

\* @return FactoryCommonInfo

\*/

public FactoryCommonInfo findMakingDetail(String userUnique,

String factoryId);

/\*\*

\* 新建厂房

\*

\* @author zy

\* @param userUnique

\* @param FactoryCommonInfo

\* @return

\*/

public boolean addMakingFactory(FactoryCommonInfo factoryCommonInfo);

/\*\*

\* 暂停修建厂房

\*

\* @author zy

\* @param userUnique

\* @param factoryId

\* @return

\*/

public boolean updateStatusToZero(String userUnique, String factoryId);

/\*\*

\* 恢复修建厂房

\*

\* @author zy

\* @param userUnique

\* @param factoryId

\* @return

\*/

public boolean updateStatusToOne(String userUnique, String factoryId);

/\*\*

\* findfindAllFactoryType 查找所有基本表中的factoryType

\* @return

\*List<String>

\* @exception

\* @since 1.0.0

\* @author hhy

\*/

public List<String> findAllFactoryType();

}

package com.cqupt.mis.erp.manager.factory;

import java.util.List;

import com.cqupt.mis.erp.model.factory.FactoryBasicInfo;

import com.cqupt.mis.erp.model.factory.FactoryCommonInfo;

public interface FactoryRentDao {

/\*\*

\* 显示租用厂房信息

\*

\* @author zy

\* @param userUnique

\* @return List<FactoryCommonInfo>

\* \*\*/

public List<FactoryCommonInfo> findFactoryRents(String userUnique);

/\*\*

\* findFactoryRents

\* @param userUnique

\* @param marketName

\* @return

\*List<FactoryCommonInfo>

\* @exception

\* @since 1.0.0

\* @author hhy

\*/

public List<FactoryCommonInfo> findFactoryRents(String userUnique,String marketName);

/\*\*

\* 查询租用详细厂房信息当已安装生产线为0时

\*

\* @author zy

\* @param userUnique

\* @param factoryId

\* @return

\*/

public FactoryCommonInfo findProductLineNumberIsNull(String userUnique,

String factoryId);

/\*\*

\* 显示租用详细厂房信息

\*

\* @author zy

\* @param userUnique

\* @param factoryId

\* @return List<FactoryCommonInfo>

\*/

public FactoryCommonInfo findRentDetail(String userUnique, String factoryId);

/\*\*

\* 入“新租赁厂房”的时候，获取所有厂房基本信息表（FACTORYBASIC）里面所拥有的

\* 厂房类型;租用该厂房时，每期需要交纳的租金;还需要等待多少周期该租赁的厂房才可以使用

\*

\* @author zy

\* @param factoryType

\* @return List<FactoryCommonInfo>

\*/

public List<FactoryBasicInfo> findAllFactoryBasics();

/\*\*

\* 入“新租赁厂房”的时候，获取特定厂房基本信息表（FACTORYBASIC）里面所拥有的

\* 厂房类型;租用该厂房时，每期需要交纳的租金;还需要等待多少周期该租赁的厂房才可以使用

\*

\* @author zy

\* @param factoryType

\* @return List<FactoryCommonInfo>

\*/

public FactoryBasicInfo findUniqueFactoryBasic(String factoryType);

/\*\*

\* 增加“新租赁厂房”

\*

\* @author zy

\* @param userUnique

\* @param factoryId

\* @return List<FactoryCommonInfo>

\*/

public boolean addRentFactory(FactoryCommonInfo factoryCommonInfo);

/\*\*

\* 停止租用厂房

\* @author zy

\* @param userUnique

\* @param factoryId

\* @return

\*/

public boolean deleteRentedFactory(String userUnique, String factoryId);

}

**package** com.cqupt.mis.erp.service.finance.account;

**public** **interface** AccountDetailService {

/\*\*

\* 向向分录明细表(ACCOUNTDETAIL)表中添加两条记录

\* 当Item对应的账目为资产时，ItemType为“借”符号为“+”，贷为“-”；

\* 当Item对应的账目为负债时，ItemType为“借”符号为“-”，贷为“+”；

\* 当Item对应的账目为收入时，ItemType为“借”符号为“-”，贷为“+”；

\* 当Item对应的账目为成本时，ItemType为“借”符号为“+”，贷为“-”；

\* 当Item对应的账目为利润时，ItemType为“借”符号为“-”，贷为“+”；

\* 当Item对应的账目为所有者权益时，ItemType为“借”符号为“-”，贷为“+”；

\*

\* **@author** 毛家杰

\* **@param** userUnique

\* 用户在竞赛中的唯一标识

\* **@param** item1

\* 科目的名称,外码，参照 ACCOUNTTWOTOONEBASIC表的TwoItem

\* **@param** item2

\* 科目的名称,外码，参照 ACCOUNTTWOTOONEBASIC表的TwoItem

\* **@param** itemType1

\* 科目对应的借或贷类型，其值只能是“借”或者“贷”

\* **@param** itemType2

\* 科目对应的借或贷类型，其值只能是“借”或者“贷”

\* **@param** money

\* 金额

\* **@param** calValue1

\* CalValue是为了方便计算“资产负债表”和“利润”表而设立的值，其数值大小与Money字段相同，但是带有符号。

\* **@param** calValue2

\* CalValue是为了方便计算“资产负债表”和“利润”表而设立的值，其数值大小与Money字段相同，但是带有符号。

\*/

**public** **void** addAccountDetail(String userUnique, String item1, String item2,

String itemType1, String itemType2, Double money, Double calValue1,

Double calValue2);

/\*\*

\* 向向分录明细表(ACCOUNTDETAIL)表中添加3条记录, 注意itemType1和itemType2最好一样,以后显示在网页上会好看点,

\* 最好让calValue1,calValue2的绝对值等于calValue3的绝对值,

\* 当Item对应的账目为资产时，ItemType为“借”符号为“+”，贷为“-”；

\* 当Item对应的账目为负债时，ItemType为“借”符号为“-”，贷为“+”；

\* 当Item对应的账目为收入时，ItemType为“借”符号为“-”，贷为“+”；

\* 当Item对应的账目为成本时，ItemType为“借”符号为“+”，贷为“-”；

\* 当Item对应的账目为利润时，ItemType为“借”符号为“-”，贷为“+”；

\* 当Item对应的账目为所有者权益时，ItemType为“借”符号为“-”，贷为“+”；

\*

\* **@author** 毛家杰

\* **@param** userUnique

\* 用户在竞赛中的唯一标识

\* **@param** item1

\* 科目的名称,外码，参照 ACCOUNTTWOTOONEBASIC表的TwoItem

\* **@param** item2

\* 科目的名称,外码，参照 ACCOUNTTWOTOONEBASIC表的TwoItem

\* **@param** item3

\* 科目的名称,外码，参照 ACCOUNTTWOTOONEBASIC表的TwoItem

\* **@param** itemType1

\* 科目对应的借或贷类型，其值只能是“借”或者“贷”

\* **@param** itemType2

\* 科目对应的借或贷类型，其值只能是“借”或者“贷”

\* **@param** itemType3

\* 科目对应的借或贷类型，其值只能是“借”或者“贷”

\* **@param** calValue1

\* CalValue是为了方便计算“资产负债表”和“利润”表而设立的值，其数值大小与Money字段相同，但是带有符号。

\* **@param** calValue2

\* CalValue是为了方便计算“资产负债表”和“利润”表而设立的值，其数值大小与Money字段相同，但是带有符号。

\* **@param** calValue3

\* CalValue是为了方便计算“资产负债表”和“利润”表而设立的值，其数值大小与Money字段相同，但是带有符号。

\*/

**public** **void** addAccountDetail(String userUnique, String item1, String item2,

String item3, String itemType1, String itemType2, String itemType3,

Double calValue1, Double calValue2, Double calValue3);

}

package com.cqupt.mis.erp.service.finance.account.impl;

import javax.annotation.Resource;

import org.springframework.stereotype.Service;

import com.cqupt.mis.erp.manager.bank.LoanOfUserDao;

import com.cqupt.mis.erp.manager.finance.account.AccountDetailDao;

import com.cqupt.mis.erp.manager.finance.account.AccountHeadDao;

import com.cqupt.mis.erp.service.finance.account.AccountDetailService;

import com.cqupt.mis.erp.utils.ERPUtils;

@Service("accountDetailService")

public class AccountDetailServiceImpl implements AccountDetailService {

@Resource

AccountDetailDao accountDetailDao;

@Resource

AccountHeadDao accountHeadlDao;

@Resource

LoanOfUserDao loanOfUserDao;

@Override

public void addAccountDetail(String userUnique, String item1, String item2,

String itemType1, String itemType2, Double money, Double calValue1,

Double calValue2) {

;//lx:四舍五入的double处理

accountDetailDao.addAccountDetail(userUnique,

accountHeadlDao.findAccountID(userUnique), item1, itemType1,

money, ERPUtils.round(calValue1));

accountDetailDao.addAccountDetail(userUnique,

accountHeadlDao.findAccountID(userUnique), item2, itemType2,

money, ERPUtils.round(calValue2));

}

@Override

public void addAccountDetail(String userUnique, String item1, String item2,

String item3, String itemType1, String itemType2, String itemType3,

Double calValue1, Double calValue2, Double calValue3) {

Double money1 = ERPUtils.round(Math.abs(calValue1));

Double money2 = ERPUtils.round(Math.abs(calValue2));

Double money3 = ERPUtils.round(Math.abs(calValue3));

//lx:四舍五入处理

accountDetailDao.addAccountDetail(userUnique,

accountHeadlDao.findAccountID(userUnique), item1, itemType1,

money1, calValue1);

accountDetailDao.addAccountDetail(userUnique,

accountHeadlDao.findAccountID(userUnique), item2, itemType2,

money2, calValue2);

accountDetailDao.addAccountDetail(userUnique,

accountHeadlDao.findAccountID(userUnique), item3, itemType3,

money3, calValue3);

}

}

package com.cqupt.mis.erp.service.finance.account;

import java.util.List;

import java.util.Map;

import com.cqupt.mis.erp.model.finance.AccountHead;

public interface AccountHeadService {

/\*\*

\* 填写分录头表(ACCOUNTHEAD)

\*

\* @author 毛家杰

\* @param userUnique

\* 用户在竞赛中的唯一标识

\* @param happenTime

\* 发生此事件的时间，即第几期

\* @param description

\* 产生此分录的说明

\*/

public void addAccountHead(String userUnique, int happenTime,

String description);

/\*\*

\* 查询会计分录表

\*

\* @author 毛家杰

\* @param userUnique

\* 用户唯一码

\* @param minYear

\* 开始年

\* @param minPeriod

\* 开始期

\* @param maxYear

\* 结束年

\* @param maxPeriod

\* 结束期

\* @param pageIndex

\* 要查询的页码

\* @return

\*/

public List<AccountHead> findAccount(String userUnique, Integer minYear,

Integer minPeriod, Integer maxYear, Integer maxPeriod,

Integer pageIndex);

/\*\*

\* 查询总的记录数和总的页数,都装在一个map里面了 ,key分别为recordCount和pageCount,这样可以减少访问数据库的次数

\*

\* @author 毛家杰

\* @param userUnique

\* 用户唯一码

\* @param minYear

\* 开始年

\* @param minPeriod

\* 开始期

\* @param maxYear

\* 结束年

\* @param maxPeriod

\* 结束期

\* @return

\*/

public Map<String, Object> findPageMsg(String userUnique,

Integer minYear, Integer minPeriod, Integer maxYear,

Integer maxPeriod, Integer pageIndex);

}

package com.cqupt.mis.erp.service.finance.account.impl;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.annotation.Resource;

import org.springframework.stereotype.Service;

import com.cqupt.mis.erp.manager.finance.account.AccountDetailDao;

import com.cqupt.mis.erp.manager.finance.account.AccountHeadDao;

import com.cqupt.mis.erp.manager.tool.CommonDao;

import com.cqupt.mis.erp.model.finance.AccountDetail;

import com.cqupt.mis.erp.model.finance.AccountHead;

import com.cqupt.mis.erp.service.finance.account.AccountHeadService;

@Service("accountHeadService")

public class AccountHeadServiceImpl implements AccountHeadService {

@Resource

AccountHeadDao accountHeadDao;

@Resource

AccountDetailDao accountDetailDao;

@Resource

CommonDao commonDao;

// 设定每一页显示的记录数量

private Integer pageSize = 7;

@Override

public void addAccountHead(String userUnique, int happenTime,

String description) {

accountHeadDao.addAccountHead(userUnique, happenTime, description);

}

@Override

public List<AccountHead> findAccount(String userUnique, Integer minYear,

Integer minPeriod, Integer maxYear, Integer maxPeriod,

Integer pageIndex) {

List<AccountHead> accountList;

List<AccountDetail> detailList;

// 设定开始时间,结束时间

Integer minTime = minPeriod + commonDao.findPeriodOfYear(userUnique)

\* (minYear - 1);

Integer maxTime = maxPeriod + commonDao.findPeriodOfYear(userUnique)

\* (maxYear - 1);

// 计算rowNum

Integer minNum = (pageIndex - 1) \* pageSize;

//limit 后面的数字是pagesize 意思

Integer maxNum = pageSize;

accountList = accountHeadDao.findWithPageNumAndTime(userUnique, minNum,

maxNum, minTime, maxTime);

//如果accountList.size为0,那么get(0)操作会出错

if(accountList.size()==0){

return accountList;

}

// 由于accoutList是按照accountID的升序排列的,所以可以得出最小最大accoutID

Integer minID = accountList.get(0).getAccountID();

Integer maxID = accountList.get(accountList.size() - 1).getAccountID();

detailList = accountDetailDao.findAccountDetail(userUnique, minID,

maxID);

for (int i = 0; i < accountList.size(); i++) {

accountList.get(i).setAccountDetailList(

new ArrayList<AccountDetail>());

for (int j = 0; j < detailList.size(); j++) {

if (accountList.get(i).getAccountID().equals(detailList.get(j).getAccountId())){

accountList.get(i).getAccountDetailList()

.add(detailList.get(j));

}

}

}

return accountList;

}

@Override

public Map<String, Object> findPageMsg(String userUnique, Integer minYear,

Integer minPeriod, Integer maxYear, Integer maxPeriod,

Integer pageIndex) {

Integer periodOfYear = commonDao.findPeriodOfYear(userUnique);

Integer currentTime = commonDao.findCurrentTime(userUnique);

// 设定开始时间,结束时间

Integer minTime = minPeriod + commonDao.findPeriodOfYear(userUnique)

\* (minYear - 1);

Integer maxTime = maxPeriod + commonDao.findPeriodOfYear(userUnique)

\* (maxYear - 1);

Map<String, Object> map = new HashMap<String, Object>();

Integer recordCount = accountHeadDao.findRecordCount(userUnique,

minTime, maxTime);

Integer pageCount;

if(recordCount == 0){

pageCount = 0;

pageIndex = 0;

}else{

pageCount = (recordCount-1) / pageSize + 1;

}

map.put("recordCount", recordCount);

map.put("pageCount", pageCount);

map.put("pageIndex", pageIndex);

map.put("minYear", minYear);

map.put("minPeriod", minPeriod);

map.put("maxYear", maxYear);

map.put("maxPeriod", maxPeriod);

map.put("periodOfYear", periodOfYear);

map.put("yearInGame", currentTime / periodOfYear + 1);

return map;

}

}

**package** com.cqupt.mis.erp.service.forwardquarter;

**public** **interface** ForwardQuarterService {

/\*\*

\* 进入下一周期前检测跳转状态 ,用(int)forwardStatus来判断 <br>

\* 0:用户已经破产 <br>

\* 1:用户正在经营 <br>

\* 2:是年初但未完成广告费投放和订单选取确认 <br>

\* 3:游戏已经结束 <br>

\* **@param** userUnique

\*

\* **@return**

\*/

**public** **int** ForwardStatus(String userUnique);

/\*\*

\* 判断用户是否gameOver <br>

\* true: GameOver <br>

\* false: 继续玩呗！ <br>

\* **@param** userUnique

\* **@author** lx

\* **@return**

\*/

**public** **boolean** isGameOver(String userUnique);

/\*\*

\* 判断资产负债-用户是否破产 <br>

\* ture：破产 <br>

\* false：未破产 <br>

\* **@param** userUnique

\* **@author** lx

\* **@return**

\*/

**public** **boolean** isBalancesheetOut(String userUnique);

/\*\*

\* 判断用户是否还有cash <br>

\* true: 有 <br>

\* false： 木有 <br>

\* **@param** userUnique

\* **@author** lx

\* **@return**

\*/

**public** **boolean** isHaveCash(String userUnique);

/\*\*

\* 破产清算,设置破产用户在后面每年年初选订单的状态

\* **@author** lx

\* **@param** userUnique

\*/

**public** **void** bankruptcy(String userUnique);

/\*\*

\* 如果用户已经破产，我们需要在以后的每年年初设置该破产用户的选单状态为1(已经选单)，

\* 不去影响其他人的游戏进度.

\* **@param** userUnique

\* **@author** lx

\*/

**public** **void** insertDataToAdststusofuser(String userUnique, **int** D);

/\*\*

\* 判断用户是否已经结束比赛，如果结束，进行相应的操作，如果没有，则什么也不做！

\* **@param** userUnique

\* **@author** lx

\*/

**public** **boolean** updateFinishedGame(String userUnique);

/\*\*

\* 如果所有用户里最小的currentPeriod都比GameGroup里的currentPeriod大，则更新游戏组的currentPeriod

\* **@author** lx

\*/

**public** **void** updateGroupCurrentPeriod(String groupName);

}

package com.cqupt.mis.erp.service.forwardquarter.impl;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.annotation.Resource;

import org.springframework.stereotype.Component;

import com.cqupt.mis.erp.manager.advertisement.AdvertisementStatusOfUserDao;

import com.cqupt.mis.erp.manager.finance.balancesheet.BalancesheetDao;

import com.cqupt.mis.erp.manager.finance.fund.CashDao;

import com.cqupt.mis.erp.manager.registerlogin.GameGroupDao;

import com.cqupt.mis.erp.manager.registerlogin.GameGroupMemberDao;

import com.cqupt.mis.erp.manager.tool.CommonDao;

import com.cqupt.mis.erp.model.advertisement.AdvertisementStatusOfUser;

import com.cqupt.mis.erp.model.registerlogin.GameGroupInfo;

import com.cqupt.mis.erp.model.vo.BalancesheetVO;

import com.cqupt.mis.erp.model.vo.GameGroupVO;

import com.cqupt.mis.erp.service.advertisement.AdvertisementService;

import com.cqupt.mis.erp.service.advertisement.AdvertisementStatusOfUserService;

import com.cqupt.mis.erp.service.forwardquarter.ForwardQuarterService;

import com.cqupt.mis.erp.service.registerlogin.GameGroupMemberService;

import com.cqupt.mis.erp.service.registerlogin.GameGroupService;

import com.cqupt.mis.erp.utils.dwr.DWRPush;

@Component("forwardQuarterService")

public class ForwardQuarterServiceImpl implements ForwardQuarterService {

@Resource

private GameGroupMemberService gameGroupMemberService;

@Resource

private GameGroupMemberDao gameGroupMemberDao;

@Resource

private GameGroupService gameGroupService;

@Resource

private GameGroupDao gameGroupDao;

@Resource

private AdvertisementStatusOfUserService advertisementStatusOfUserService;

@Resource

private CommonDao commonDao;

@Resource

private CashDao cashDao;

@Resource

private AdvertisementStatusOfUserDao advertisementStatusOfUserDao;

@Resource

private BalancesheetDao balancesheetDao;

@Resource

private AdvertisementService advertisementService;

@Override

public int ForwardStatus(String userUnique) {

int forwardStatus = -1; //0:用户已破产 , 1:进入下一年 , 2:年初并且用户未完成广告投放和选订单

int status = 0;

status = gameGroupMemberService.findStatusByUserUnique(userUnique);

if(status == 0) {

forwardStatus = status;

} else {

//检查是否是年初

boolean isYearBegin = gameGroupService.isYearBegin(userUnique);

//检查用户是否完成了“广告投放”和“选订单”

boolean isFinish = advertisementStatusOfUserService.findFinishOrderFlag(userUnique);

if(isYearBegin && !isFinish) {

//表示是年初并且用户“未”完成广告投放和选订单

forwardStatus = 2;

return forwardStatus;

} else {

//orcale下一周期的存储函数调用

commonDao.runFunctionInOracle(userUnique);

//检查用户是否破产

boolean isGameOver = isGameOver(userUnique);

if(isGameOver) {

//破产清算

bankruptcy(userUnique);

forwardStatus = 0;//表示用户破产

return forwardStatus;

}

//更新用户的时间

gameGroupMemberDao.updateIncreaseUserCurrentPeriod(userUnique);

//第15步 是否更新GAMEGROUP表中的CurrentPeriod字段值。

String groupName =commonDao.findGameGroupNameByUserUnique(userUnique);

updateGroupCurrentPeriod(groupName);

//第16步 检查用户是否已经完成比赛

boolean isEndGame = updateFinishedGame(userUnique);//注意：我们直接处理需要更新的操作，如果不需要更新，则不会对数据库进行操作的.

if(isEndGame){

forwardStatus = 3;

return forwardStatus;

}

if(gameGroupService.isYearBegin(userUnique)) {

advertisementService.initAdOfUser(userUnique);

advertisementStatusOfUserService.initAdStatusOfUser(userUnique);

}

forwardStatus = 1;

}

}

return forwardStatus;

}

@Override

public boolean isGameOver(String userUnique) {

if(isBalancesheetOut(userUnique) | !isHaveCash(userUnique)) {

return true;

} else {

return false;

}

}

@Override

public boolean isBalancesheetOut(String userUnique) {

List<BalancesheetVO> balancesheetVOs= balancesheetDao.findBalancesheetVO(userUnique);

float assets = 0,liability = 0;

for(int i=0; i<balancesheetVOs.size(); i++) {

BalancesheetVO balancesheetVO = balancesheetVOs.get(i);

if("资产".equals(balancesheetVO.getType1())) {

assets += balancesheetVO.getCvalue();

//System.out.println("assets=" + assets);

}

if("负债".equals(balancesheetVO.getType1())) {

liability += balancesheetVO.getCvalue();

//System.out.println("liability=" + liability);

}

}

if(assets<liability) {

return true;

}

return false;

}

@Override

public boolean isHaveCash(String userUnique) {

double cash = cashDao.findCash(userUnique);

if(cash < 0) {

return false;

} else {

return true;

}

}

@Override

public void bankruptcy(String userUnique) {

DWRPush.refresh();

//先更新status-设置用户状态为0(破产)

gameGroupMemberDao.updateBankruptcyUserStatus(userUnique,0);

GameGroupInfo gameGroupInfo = new GameGroupInfo();

gameGroupInfo = gameGroupDao.findGameGroupYearAndCurrentPeriod(userUnique);

//理论上是不需要这样的. 但是还是要将这个人这期给结束掉. 如果第一期就退出比赛的话.

advertisementStatusOfUserDao.updateFinishAdFlag(userUnique, 1, 1);

advertisementStatusOfUserDao.updateFinishOrderFlag(userUnique, 1, 1);

int A = gameGroupInfo.getCurrentPeriod(); //当前期数

int B = gameGroupInfo.getPeriodsOfOneYear(); //每年的期数

int C = gameGroupInfo.getYears(); //年数

int D = 1; //计算出的每年开始的第一期

//System.out.println("A=" + A +" B:"+ B +" C:"+ C + " D:" + D);

while(true) {

if( D>A && D<=(C-1)\*B + 1){

insertDataToAdststusofuser(userUnique,D);

}

if(D>(C-1)\*B+1){

break ;

}else{

D = D+B;

//继续执行if语句

}

}

}

@Override

public void insertDataToAdststusofuser(String userUnique, int D) {

AdvertisementStatusOfUser advertisementStatusOfUser = new AdvertisementStatusOfUser();

advertisementStatusOfUser.setUserUnique(userUnique);

advertisementStatusOfUser.setPeriod(D);

advertisementStatusOfUser.setFinishAdvertiseFlag(1);

advertisementStatusOfUser.setFinishOrderFlag(1);

advertisementStatusOfUser.setChooseOrderFlag(1);

advertisementStatusOfUserDao.addAdStatusOfUser(advertisementStatusOfUser);

}

@Override

public boolean updateFinishedGame(String userUnique) {

String groupName = commonDao.findGameGroupNameByUserUnique(userUnique);

Map<String, Object> map = new HashMap<String, Object>();

map.put("userUnique", userUnique);

map.put("groupName", groupName);

GameGroupVO gameGroupVO = new GameGroupVO();

gameGroupVO = gameGroupDao.findGameGroupVO(map);

int CurrentPeriod = gameGroupVO.getCurrentPeriod();

int getTotalPeriodOfGroup = gameGroupVO.getTotalPeriodOfGroup();

if(CurrentPeriod > getTotalPeriodOfGroup) {

gameGroupMemberDao.updateUserStatusToFinishGame(userUnique);

return true;

}

return false;

}

@Override

public void updateGroupCurrentPeriod(String groupName) {

int minPeriod = gameGroupMemberDao.findLeastCurrentPeriodByGroupName(groupName);

int currentPeriod = gameGroupDao.findCurrentPeriodByGroupName(groupName);

if(currentPeriod < minPeriod) {

gameGroupDao.updateGroupCurrentPeriod(groupName, minPeriod);

}

}

}

package com.cqupt.mis.erp.service.iso;

import java.util.List;

import com.cqupt.mis.erp.model.iso.ISOBasic;

import com.cqupt.mis.erp.model.iso.ISODeveloped;

import com.cqupt.mis.erp.model.iso.ISODeveloping;

import com.cqupt.mis.erp.model.iso.ISOUndevelop;

public interface ISOManageService {

/\*\*

\* 查看所有未开拓的iso

\* @author zy

\* @param userUnique

\* @return

\*/

public List<ISOUndevelop> findAllISOUndevelop(String userUnique);

/\*\*

\* 查看所有开发中的iso

\* @author zy

\* @param userUnique

\* @return

\*/

public List<ISODeveloping> findAllISODeveloping(String userUnique);

/\*\*

\*查看单个未开拓的iso

\* @author zy

\* @param userUnique

\* @param isoName

\* @return

\*/

public ISOUndevelop findOneUndevelop(String userUnique,String isoName);

/\*\*

\* 开拓iso

\* @author zy

\* @param userUnique

\* @param isoName

\* @return

\*/

public boolean addISOToISODeveloping(String userUnique, String isoName);

/\*\*

\* 恢复开发某个iso认证

\* @author zy

\* @param userUnique

\* @param isoName

\* @return

\*/

public boolean updateISODevelopingStatusToOne(String userUnique,String isoName);

/\*\*

\* 停止开发某个iso认证

\* @author zy

\* @param userUnique

\* @return

\*/

public boolean updateISODevelopingStatusToZero(String userUnique,String isoName);

/\*\*

\*查看所有已开发的iso认证

\* @author zy

\* @param userUnique

\* @return

\*/

public List<ISODeveloped> findAllISODeveloped(String userUnique);

/\*\*

\* 向已开拓iso中增加一条记录

\* @author zy

\* @param ISOManageAction

\* @return

\*/

public boolean addISOToISODeveloped(String userUnique, String isoName);

/\*\*

\* 恢复对停止维护的已开发ISO的维护

\* @author zy

\* @param userUnique

\* @param isoName

\* @return

\*/

public boolean updateISODevelopedStatusToOne(String userUnique,String isoName);

/\*\*

\* 停止已开发ISO的维护

\* @author zy

\* @param userUnique

\* @return

\*/

public boolean updateISODevelopedStatusToZero(String userUnique,String isoName);

/\*\*

\* 取出所有ISO认证基础信息

\* @author LX

\* @return

\*/

public List<ISOBasic> findAllISOBasic();

/\*\*

\* 取出基础ISO认证的条数

\* @author LX

\* @return

\*/

public Integer findISOBasicNum();

}

package com.cqupt.mis.erp.service.iso.impl;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.stereotype.Service;

import com.cqupt.mis.erp.manager.iso.ISOBasicDao;

import com.cqupt.mis.erp.manager.iso.ISOManageDao;

import com.cqupt.mis.erp.manager.iso.impl.ISOBasicDaoImpl;

import com.cqupt.mis.erp.manager.tool.CommonDao;

import com.cqupt.mis.erp.model.iso.ISOBasic;

import com.cqupt.mis.erp.model.iso.ISODeveloped;

import com.cqupt.mis.erp.model.iso.ISODeveloping;

import com.cqupt.mis.erp.model.iso.ISOUndevelop;

import com.cqupt.mis.erp.service.iso.ISOManageService;

@Service("iSOManageService")

public class ISOManageServiceImpl implements ISOManageService{

@Resource

private CommonDao commonDao;

@Resource

private ISOManageDao iSOManageDao;

@Resource

private ISOBasicDao iSOBasicDao;

@Override

public boolean addISOToISODeveloping(String userUnique, String isoName) {

ISODeveloping iso=new ISODeveloping();

int beginTime=commonDao.findCurrentTime(userUnique);

ISOUndevelop unIso=iSOManageDao.findOneUndevelop(userUnique, isoName);

iso.setUserUnique(userUnique);

iso.setIsoName(unIso.getIsoName());

iso.setResearchCost(unIso.getResearchCost());

iso.setResearchPeriod(unIso.getResearchPeriod());

iso.setBeginTime(beginTime);

iSOManageDao.addIsoToISODeveloping(iso);

iSOManageDao.delelteISOUndevelop(userUnique, isoName);

return true;

}

@Override

public List<ISOUndevelop> findAllISOUndevelop(String userUnique) {

return iSOManageDao.findAllISOUndevelop(userUnique);

}

@Override

public ISOUndevelop findOneUndevelop(String userUnique, String isoName) {

return iSOManageDao.findOneUndevelop(userUnique, isoName);

}

@Override

public boolean updateISODevelopingStatusToOne(String userUnique,

String isoName) {

iSOManageDao.updateISODevelopingStatus(userUnique, isoName, 1);

return true;

}

@Override

public boolean updateISODevelopingStatusToZero(String userUnique,

String isoName) {

iSOManageDao.updateISODevelopingStatus(userUnique, isoName, 0);

return true;

}

@Override

public List<ISODeveloped> findAllISODeveloped(String userUnique) {

return iSOManageDao.findAllISODeveloped(userUnique);

}

@Override

public boolean updateISODevelopedStatusToOne(String userUnique,

String isoName) {

iSOManageDao.updateISODevelopedStatus(userUnique, isoName, 1);

return true;

}

@Override

public boolean updateISODevelopedStatusToZero(String userUnique,

String isoName) {

iSOManageDao.updateISODevelopedStatus(userUnique, isoName, 0);

return true;

}

@Override

public boolean addISOToISODeveloped(String userUnique, String isoName) {

ISOBasic isoBasic=iSOBasicDao.findOneISOBasic(isoName);

ISODeveloped isoed=new ISODeveloped();

int endTime=commonDao.findCurrentTime(userUnique);

isoed.setUserUnique(userUnique);

isoed.setIsoName(isoName);

isoed.setBeginTime(endTime);

isoed.setEndTime(endTime);

isoed.setLastStatus(0);

isoed.setStatus(1);

isoed.setMaintainCost(isoBasic.getMaintainCost());

iSOManageDao.addISOToISODeveloped(isoed);

iSOManageDao.deleteDevelopingISO(userUnique, isoName);

return true;

}

@Override

public List<ISODeveloping> findAllISODeveloping(String userUnique) {

return iSOManageDao.findAllISODeveloping(userUnique);

}

@Override

public List<ISOBasic> findAllISOBasic() {

return iSOBasicDao.findAllISOBasic();

}

@Override

public Integer findISOBasicNum() {

return iSOBasicDao.findISOBasicNum();

}

}

package com.cqupt.mis.erp.service.chatroom;

import java.util.List;

import com.cqupt.mis.erp.model.chatroom.ChatRoom;

public interface ChatRoomService {

/\*\*

\* 寻找聊天记录

\*

\* @author 毛家杰

\* @param userUnique 用户唯一码

\* @return

\*/

public List<ChatRoom> findChatRecords(String userUnique);

/\*\*

\*

\* addChatRecordByUserId

\* 何海源重新修改

\* @param groupName 组名

\* @param userName 用户名

\* @param record 聊天记录

\*void

\* @exception

\*

\*/

public void addChatRecordByUserId(String groupName, String userName, String record);

/\*\*

\* addChatRecordByUserUnique 针对 开始游戏之后的方法.

\*

\* @param userUnique

\* @param groupName

\* @param record

\*void

\* @exception

\* @since 1.0.0

\*/

public void addChatRecordByUserUnique(String userUnique, String groupName, String record);

}

package com.cqupt.mis.erp.service.chatroom.impl;

import java.util.List;

import javax.annotation.Resource;

import org.springframework.stereotype.Service;

import com.cqupt.mis.erp.manager.chatroom.ChatRoomDao;

import com.cqupt.mis.erp.manager.tool.CommonDao;

import com.cqupt.mis.erp.model.chatroom.ChatRoom;

import com.cqupt.mis.erp.service.chatroom.ChatRoomService;

import com.cqupt.mis.erp.utils.JSONUtils;

import com.cqupt.mis.erp.utils.dwr.DWRPush;

@Service("chatRoomService")

public class ChatRoomServiceImpl implements ChatRoomService{

@Resource

private ChatRoomDao chatRoomDao;

@Resource

private CommonDao commonDao;

private static final Integer numOfRecords = 50;//如果输入50 , 那么就会返回50条记录

@Override

public List<ChatRoom> findChatRecords(String userUnique) {

String groupName = commonDao.findGameGroupNameByUserUnique(userUnique);

if(groupName == null){

groupName = commonDao.findCurrentGroupName(userUnique);

if(groupName == null){

groupName = "";

}

}

return chatRoomDao.findChatRecords(groupName,numOfRecords);

}

/\*\*

\* @author yangqing

\* 将chatRoomDao.refreshRecord();从chatRoomDao.addRecord(userName, record, groupName);

\* 中提取到这里，避免DAO层感染

\*/

@Override

public void addChatRecordByUserId(String groupName, String userName, String record) {

chatRoomDao.refreshRecord();

chatRoomDao.addRecord(userName, record, groupName);

this.apendDialogGameStart(userName, record, groupName);

return;

}

/\*\*

\* @author yangqing

\* 将chatRoomDao.refreshRecord();从chatRoomDao.addRecord(userName, record, groupName);

\* 中提取到这里，避免DAO层感染

\*/

@Override

public void addChatRecordByUserUnique(String userUnique, String groupName,String record) {

String userName = commonDao.findUsernameByUserUnique(userUnique);

chatRoomDao.refreshRecord();

chatRoomDao.addRecord(userName, record, groupName);

this.apendDialog(userName, record, groupName);

return;

}

public void apendDialog(String userName, String record,String groupName){

ChatRoom chat = new ChatRoom();

chat.setRecord(record);

chat.setUserName(userName);

chat.setGroupName(groupName);

DWRPush.pushMessageWithFilter("apendDialog",JSONUtils.toJSONString(chat),"adAndOrderFlag"+groupName);

}

public void apendDialogGameStart(String userName, String record,String groupName){

ChatRoom chat = new ChatRoom();

chat.setRecord(record);

chat.setUserName(userName);

chat.setGroupName(groupName);

//这个标记是还没有开始游戏的时候分的.

DWRPush.pushMessageWithFilter("apendDialog",JSONUtils.toJSONString(chat),"showGameGroupMemberList"+groupName);

}

}

package com.cqupt.mis.erp.service.prediction;

import java.util.List;

import java.util.Map;

import com.cqupt.mis.erp.model.prediction.PredictionOfGroup;

import com.cqupt.mis.erp.model.product.ProductOfUser;

import com.cqupt.mis.erp.model.vo.PredictionOfGroupVO;

public interface PredictionOfGroupService {

/\*\*

\* 添加市场预测信息

\* @author lx

\* @param predictionOfGroup

\*/

public void addPredictionOfGroup(PredictionOfGroup predictionOfGroup);

/\*\*

\* 取出所有市场预测信息

\* @author lx

\* @param groupName

\* @return

\*/

public List<PredictionOfGroup> findAllPredictionOfGroup(String groupName);

/\*\*

\* 批量添加市场预测信息

\*

\* @param predictionOfGroup

\*/

public boolean addPredictionOfGroups(List<PredictionOfGroup> predictionOfGroups);

/\*\*

\* 根据市场名称取出相应的预测信息

\* @param marketName

\* @return

\*/

public Map<String,Object> findPredictionOfGroupByMarketName(String marketName, String userUnique);

}

package com.cqupt.mis.erp.service.prediction.impl;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.annotation.Resource;

import org.springframework.stereotype.Component;

import com.cqupt.mis.erp.manager.prediction.PredictionOfGroupDao;

import com.cqupt.mis.erp.manager.prediction.impl.PredictionOfGroupDaoImpl;

import com.cqupt.mis.erp.manager.tool.CommonDao;

import com.cqupt.mis.erp.model.prediction.PredictionOfGroup;

import com.cqupt.mis.erp.model.vo.PredictionOfGroupVO;

import com.cqupt.mis.erp.service.prediction.PredictionOfGroupService;

@Component("predictionOfGroupService")

public class PredictionOfGroupServiceImpl implements PredictionOfGroupService{

@Resource

private PredictionOfGroupDao predictionOfGroupDao;

@Resource

private CommonDao commonDao;

@Override

public void addPredictionOfGroup(PredictionOfGroup predictionOfGroup) {

predictionOfGroupDao.addPredictionOfGroup(predictionOfGroup);

}

@Override

public List<PredictionOfGroup> findAllPredictionOfGroup(String groupName) {

return predictionOfGroupDao.findAllPredictionOfGroup(groupName);

}

@Override

public boolean addPredictionOfGroups(List<PredictionOfGroup> predictionOfGroups) {

return predictionOfGroupDao.addPredictionOfGroups(predictionOfGroups);

}

@Override

//TODO 考虑重构

public Map<String, Object> findPredictionOfGroupByMarketName(String marketName, String userUnique) {

String groupName = commonDao.findGameGroupNameByUserUnique(userUnique);

//先取出所有预测信息

List<PredictionOfGroupVO> li = predictionOfGroupDao.findPredictionOfGroupByMarketName(marketName, groupName);

//准备装入的集合

Map<String, Object> map = new HashMap<String, Object>(5);

String productName = "";

//游戏总年数

int year = commonDao.findTotalYear(userUnique);

//每种产品各年的预测量装入该集合(重复使用该集合装4种产品)

List<Integer> list;

for(int n=0; n<4; n++){

list = new ArrayList<Integer>();

for(int i=0; i<(li.size()/4); i++) {

//根据期数不同，取出相应的信息，period实质上是数据库取出的信息的序列号，因为从数据库取出信息是排好序的。

int period = i+year\*n;

productName = li.get(period).getProductName();

list.add(li.get(period).getMount());

}

map.put(productName, list);

}

// System.out.println(map);

Integer min=99999;

Integer max=0;

for(int m=0; m<li.size(); m++) {

if(min > li.get(m).getMount()){

min = li.get(m).getMount();

}

if(max < li.get(m).getMount()){

max = li.get(m).getMount();

}

}

map.put("mountMin", min);

map.put("mountMax", max);

Map<String, Object> map2 = new HashMap<String, Object>(5);

List<Float> list2;

for(int n=0; n<4; n++){

list2 = new ArrayList<Float>();

for(int i=0; i<(li.size()/4); i++) {

int period = i+year\*n;

productName = li.get(period).getProductName();

list2.add(li.get(period).getPrice());

}

map2.put(productName, list2);

}

Float min2=99999f;

Float max2=0f;

for(int m=0; m<li.size(); m++) {

if(min2 > li.get(m).getPrice()){

min2 = li.get(m).getPrice();

}

if(max2 < li.get(m).getPrice()){

max2 = li.get(m).getPrice();

}

}

map2.put("priceMin", min2);

map2.put("priceMax", max2);

Map<String,Object> map3 = new HashMap<String,Object>(2);

map3.put("mountMap", map);

map3.put("priceMap", map2);

map3.put("marketName", marketName);

//System.out.println(map3);

return map3;

}

}

package com.cqupt.mis.erp.manager.factory.impl;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import org.springframework.stereotype.Repository;

import com.cqupt.mis.erp.manager.factory.FactoryMakingDao;

import com.cqupt.mis.erp.manager.tool.impl.BaseDaoImpl;

import com.cqupt.mis.erp.model.factory.FactoryBasicInfo;

import com.cqupt.mis.erp.model.factory.FactoryCommonInfo;

@Repository("factoryMakingDao")

public class FactoryMakingDaoImpl extends BaseDaoImpl implements

FactoryMakingDao {

@Override

public List<FactoryCommonInfo> findFactoryMakings(String userUnique) {

return super.findList(FactoryCommonInfo.class,"find\_all\_factory\_making", userUnique);

}

@Override

public List<FactoryCommonInfo> findFactoryMakings(String userUnique,String marketName) {

Map<String,Object> map = new HashMap<String,Object>();

map.put("userUnique", userUnique);

map.put("marketName", marketName);

return super.findList(FactoryCommonInfo.class,"find\_all\_factory\_making\_by\_marketname", map);

}

public FactoryCommonInfo findMakingDetail(String userUnique,String factoryId) {

Map<String, Object> param = new HashMap();

param.put("userUnique", userUnique);

param.put("factoryId", factoryId);

return super.load(FactoryCommonInfo.class,

"find\_factory\_making\_detail", param);

}

public boolean addMakingFactory(FactoryCommonInfo factoryCommonInfo) {

super.add(FactoryCommonInfo.class, "add\_making\_factory",factoryCommonInfo);

return true;

}

public boolean updateStatusToZero(String userUnique, String factoryId) {

Map<String, Object> param = new HashMap();

param.put("userUnique", userUnique);

param.put("factoryId", factoryId);

super.update(FactoryCommonInfo.class, "update\_status\_to\_zero",param);

return true;

}

public boolean updateStatusToOne(String userUnique, String factoryId) {

Map<String, Object> param = new HashMap();

param.put("userUnique", userUnique);

param.put("factoryId", factoryId);

super.update(FactoryCommonInfo.class, "update\_status\_to\_one", param);

return true;

}

@Override

public List<String> findAllFactoryType() {

return getSession().selectList(FactoryBasicInfo.class.getName()+".find\_all\_factorytype");

}

}

package com.cqupt.mis.erp.manager.factory.impl;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import org.springframework.stereotype.Repository;

import com.cqupt.mis.erp.manager.factory.FactoryRentDao;

import com.cqupt.mis.erp.manager.tool.impl.BaseDaoImpl;

import com.cqupt.mis.erp.model.factory.FactoryBasicInfo;

import com.cqupt.mis.erp.model.factory.FactoryCommonInfo;

@Repository("factoryRentDao")

public class FactoryRentDaoImpl extends BaseDaoImpl implements FactoryRentDao {

public List<FactoryCommonInfo> findFactoryRents(String userUnique) {

return super.findList(FactoryCommonInfo.class, "find\_all\_factory\_rent",userUnique);

}

@Override

public List<FactoryCommonInfo> findFactoryRents(String userUnique,String marketName) {

Map<String,Object> map = new HashMap<String,Object>();

map.put("userUnique", userUnique);

map.put("marketName", marketName);

return super.findList(FactoryCommonInfo.class, "find\_all\_factory\_rent\_by\_marketName",map);

}

public FactoryCommonInfo findRentDetail(String userUnique, String factoryId) {

Map param = new HashMap();

param.put("userUnique", userUnique);

param.put("factoryId", factoryId);

FactoryCommonInfo factoryCommonInfo = super.load(FactoryCommonInfo.class, "find\_factory\_rent\_detail", param);

return factoryCommonInfo;

}

public List<FactoryBasicInfo> findAllFactoryBasics() {

return getSession()

.selectList(

"com.cqupt.mis.erp.model.factory.FactoryBasicInfo.find\_all\_factory");

}

public FactoryBasicInfo findUniqueFactoryBasic(String factoryType) {

return super.load(FactoryBasicInfo.class, "find\_unique\_factory",

factoryType);

}

public boolean addRentFactory(FactoryCommonInfo factoryCommonInfo) {

try {

super.add(FactoryCommonInfo.class, "add\_new\_rent\_factory",

factoryCommonInfo);

} catch (Exception e) {

e.printStackTrace();

return false;

}

return true;

}

public boolean deleteRentedFactory(String userUnique, String factoryId) {

Map param = new HashMap();

param.put("userUnique", userUnique);

param.put("factoryId", factoryId);

try {

super.delete(FactoryCommonInfo.class, "delete\_rent\_factory", param);

} catch (Exception e) {

e.printStackTrace();

return false;

}

return true;

}

@Override

public FactoryCommonInfo findProductLineNumberIsNull(String userUnique,

String factoryId) {

Map param = new HashMap();

param.put("userUnique", userUnique);

param.put("factoryId", factoryId);

FactoryCommonInfo factoryCommonInfo = super.load(

FactoryCommonInfo.class,

"find\_factory\_rent\_detail\_with\_productlinenumber\_zero", param);

return factoryCommonInfo;

}

}

package com.cqupt.mis.erp.manager.factory.impl;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import org.springframework.stereotype.Repository;

import com.cqupt.mis.erp.manager.factory.FactoryUsingDao;

import com.cqupt.mis.erp.manager.tool.impl.BaseDaoImpl;

import com.cqupt.mis.erp.model.factory.FactoryCommonInfo;

@Repository("factoryUsingDao")

public class FactoryUsingDaoImpl extends BaseDaoImpl implements FactoryUsingDao {

@Override

public List<FactoryCommonInfo> findFactoryUsings(String userUnique) {

return super.findList(FactoryCommonInfo.class,

"find\_all\_factory\_using", userUnique);

}

@Override

public List<FactoryCommonInfo> findFactoryUsings(String userUnique,String marketName) {

Map<String, Object> map = new HashMap<String,Object>();

map.put("userUnique", userUnique);

map.put("marketName", marketName);

return super.findList(FactoryCommonInfo.class,"find\_all\_factory\_using\_by\_marketname", map);

}

@Override

public FactoryCommonInfo findUsingDetail(String userUnique, String factoryId) {

Map<String, Object> param = new HashMap();

param.put("userUnique", userUnique);

param.put("factoryId", factoryId);

return super.load(FactoryCommonInfo.class, "find\_factory\_using\_detail",

param);

}

public FactoryCommonInfo findUsingDetailWithProductLineNumber(

String userUnique, String factoryId) {

Map<String, Object> param = new HashMap();

param.put("userUnique", userUnique);

param.put("factoryId", factoryId);

return super.load(FactoryCommonInfo.class,

"find\_factory\_using\_detail\_with\_productlinenumber", param);

}

@Override

public Map findABCDE(String userUnique, String factoryId, String factoryType) {

Map<String,Object> parameterMap=new HashMap();

Map<String,Object> resultMap=new HashMap();

parameterMap.put("userUnique", userUnique);

parameterMap.put("factoryId", factoryId);

parameterMap.put("factoryType", factoryType);

float a=super.getSession().selectOne("com.cqupt.mis.erp.model.factory.FactoryCommonInfo.find\_account\_a", parameterMap);

float c=super.getSession().selectOne("com.cqupt.mis.erp.model.factory.FactoryBasicInfo.find\_account\_c",parameterMap);

float b=c-a;

float e=super.getSession().selectOne("com.cqupt.mis.erp.model.factory.FactoryBasicInfo.find\_account\_e",parameterMap);

float d=a-e;

double A=new Double(a);

double B=new Double(b);

double C=new Double(c);

double D=new Double(d);

double E=new Double(e);

resultMap.put("A", A);

resultMap.put("B", B);

resultMap.put("C", C);

resultMap.put("D", D);

resultMap.put("E", E);

return resultMap;

}

@Override

public int findFactoryDelayTime(String factoryType) {

return super.getSession().selectOne("com.cqupt.mis.erp.model.factory.FactoryBasicInfo.find\_delay\_time", factoryType);

}

@Override

public void deleteFactory(String userUnique, String factoryId) {

Map<String,Object> params=new HashMap();

params.put("userUnique", userUnique);

params.put("factoryId", factoryId);

super.delete(FactoryCommonInfo.class, "delete\_factory", params);

}

}

package com.cqupt.mis.erp.manager.factory.impl;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import org.springframework.cache.annotation.Cacheable;

import org.springframework.stereotype.Repository;

import com.cqupt.mis.erp.manager.factory.ProductLineDao;

import com.cqupt.mis.erp.manager.tool.impl.BaseDaoImpl;

import com.cqupt.mis.erp.model.factory.ProductLineBasic;

import com.cqupt.mis.erp.model.factory.ProductLineCommonInfo;

import com.cqupt.mis.erp.model.factory.ProductLineInfo;

@Repository("productLineDao")

public class ProductLineDaoImpl extends BaseDaoImpl implements ProductLineDao {

public List<ProductLineInfo> findProductLines(String userUnique, String factoryId) {

Map<String, Object> param = new HashMap<String, Object>();

param.put("userUnique", userUnique);

param.put("factoryId", factoryId);

return super.findList(ProductLineInfo.class, "find\_all\_productline\_basic", param);

}

public ProductLineCommonInfo findProductLineDetail(String userUnique,

String productLineId) {

Map<String, Object> param = new HashMap<String, Object>();

param.put("userUnique", userUnique);

param.put("productLineId", productLineId);

return super.load(ProductLineCommonInfo.class,

"find\_productline\_detail", param);

}

@Override

//@Cacheable(value={"ProductLineBasic"})

public List<ProductLineBasic> findAllProductLineBasics() {

return getSession().selectList("com.cqupt.mis.erp.model.factory.ProductLineCommonInfo.find\_all\_productline");

}

@Override

//@Cacheable(value={"ProductLineBasicByType"},key="#productLineType")

public ProductLineInfo findUniqueProductLineBasic(String productLineType) {

return super.load(ProductLineInfo.class, "find\_unique\_productline",

productLineType);

}

@Override

//@Cacheable(value={"FactoryCapacity"},key="#userUnique+#factoryId")

public int findFactoryCapacity(String userUnique, String factoryId) {

Map<String, Object> param = new HashMap<String, Object>();

param.put("userUnique", userUnique);

param.put("factoryId", factoryId);

return getSession()

.selectOne(

"com.cqupt.mis.erp.model.factory.FactoryBasicInfo.find\_factorycapacity",

param);

}

@Override

public int findProductLineNumber(String userUnique, String factoryId) {

Map<String, Object> param = new HashMap<String, Object>();

param.put("userUnique", userUnique);

param.put("factoryId", factoryId);

return getSession()

.selectOne(

"com.cqupt.mis.erp.model.factory.ProductLineCommonInfo.find\_productline\_number",

param);

}

@Override

public boolean addProductLine(ProductLineCommonInfo productlineinfo) {

try {

super.add(ProductLineCommonInfo.class, "add\_productline",productlineinfo);

} catch (Exception e) {

e.printStackTrace();

return false;

}

return true;

}

@Override

public int findProductLineStatus(String userUnique, String productLineId) {

Map<String, Object> param = new HashMap<String, Object>();

param.put("userUnique", userUnique);

param.put("productLineId", productLineId);

return getSession()

.selectOne(

"com.cqupt.mis.erp.model.factory.ProductLineCommonInfo.find\_productline\_status",

param);

}

@Override

public String findProductNameByProductLine(String userUnique,

String productLineId) {

Map<String, Object> param = new HashMap<String, Object>();

param.put("userUnique", userUnique);

param.put("productLineId", productLineId);

return getSession()

.selectOne(

"com.cqupt.mis.erp.model.factory.ProductLineCommonInfo.find\_productline\_productname",

param);

}

@Override

public boolean updateProductLineStatus(String userUnique,String productLineId, int status) {

Map<String, Object> param = new HashMap<String, Object>();

param.put("userUnique", userUnique);

param.put("productLineId", productLineId);

param.put("status", status);

try {

super.update(ProductLineCommonInfo.class, "update\_status", param);

} catch (Exception e) {

e.printStackTrace();

return false;

}

return true;

}

@Override

//@Cacheable(value={"ProductLineType"},key="#userUnique+#productLineId")

public String findProductLineType(String userUnique, String productLineId) {

Map<String, Object> param = new HashMap<String, Object>(2);

param.put("userUnique", userUnique);

param.put("productLineId", productLineId);

return super.getSession().selectOne("com.cqupt.mis.erp.model.factory.ProductLineCommonInfo.find\_productlinetype",param);

}

@Override

public int findProductLineFinishPeriod(String userUnique,String productLineId) {

Map<String, Object> param = new HashMap<String, Object>(2);

param.put("userUnique", userUnique);

param.put("productLineId", productLineId);

return super.getSession().selectOne("com.cqupt.mis.erp.model.factory.ProductLineCommonInfo.find\_productline\_finishperiod",param);

}

@Override

//@Cacheable(value={"ProductLineChangePeriod"},key="#productLineType")

public int findProductLineChangePeriod(String productLineType) {

return super.getSession().selectOne("com.cqupt.mis.erp.model.factory.ProductLineCommonInfo.find\_productline\_changeperiod",productLineType);

}

@Override

public boolean updateProductLineStatusTofive(String userUnique,String productLineId, String productName) {

Map<String, Object> param = new HashMap<String, Object>(3);

param.put("userUnique", userUnique);

param.put("productLineId", productLineId);

param.put("productName", productName);

super.update(ProductLineCommonInfo.class, "update\_status\_to\_five",param);

return true;

}

@Override

public Map<String, Object> findABCList(String userUnique, String productLineId,String productlineType) {

Map<String, Object> param = new HashMap<String, Object>();

Map<String, Object> result = new HashMap<String, Object>();

param.put("userUnique", userUnique);

param.put("productLineId", productLineId);

float a=super.getSession().selectOne("find\_A", param);

float c=super.getSession().selectOne("find\_C", productlineType);

float b=c-a;

double A=new Double(a);

double B=new Double(b);

double C=new Double(c);

result.put("A", A);

result.put("B", B);

result.put("C", C);

return result;

}

@Override

public int findDelayTime(String productLineType) {

return super.getSession().selectOne("find\_productline\_delay\_time",productLineType);

}

@Override

public boolean deleteProductLine(String userUnique, String productLineId) {

Map<String, Object> param = new HashMap<String, Object>();

param.put("userUnique", userUnique);

param.put("productLineId", productLineId);

super.getSession().selectOne("delete\_productLine",param);

return true;

}

@Override

//@Cacheable(value={"ProductLineBasicByProductLineId"},key="#productLineId")

public ProductLineBasic findProductLineBasic(String productLineId) {

return getSession().selectOne(ProductLineCommonInfo.class.getName()+".find\_product\_basic\_by\_productlineid",productLineId);

}

@Override

public ProductLineInfo findProductLineInfo(String productLineId) {

return getSession().selectOne(ProductLineInfo.class.getName()+".find\_productlineinfo\_by\_productlineid",productLineId);

}

}