三度云享家移动版源码

package com.nork.mobile.controller;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import com.nork.common.cache.CacheManager;

import com.nork.common.constant.SystemCommonConstant;

import com.nork.common.model.LoginUser;

import com.nork.common.model.ResponseEnvelope;

import com.nork.common.util.StringUtils;

import com.nork.design.model.DesignPlanRecommendFavoriteRef;

import com.nork.design.model.DesignPlanRecommendedResult;

import com.nork.design.model.Favorite;

import com.nork.design.model.FavoriteRecommendedModel;

import com.nork.design.service.DesignPlanRecommendFavoriteService;

import com.nork.mobile.model.search.DesignPlanRecommendedModel;

/\*\*

\* 移动端关于收藏功能的controller

\* @author yangzhun

\*/

@Controller

@RequestMapping("/{style}/mobile/designPlanCollect")

public class MobileDesignPlanCollectController {

@Autowired

private DesignPlanRecommendFavoriteService designPlanRecommendFavoriteService;

/\*\*

\*getFavoritesList 方法描述：列表展示用户个人创建的收藏夹

\*/

@SuppressWarnings({ "rawtypes", "unchecked" })

@ResponseBody

@RequestMapping(method = RequestMethod.POST, value = "/getFavoritesList")

public ResponseEnvelope getFavoritesList(@RequestBody FavoriteRecommendedModel model, HttpServletRequest request) {

ResponseEnvelope envelope = new ResponseEnvelope();

String msgId = model.getMsgId();

//获取登录用户信息

HashMap<String, String> map = (HashMap<String, String>) request.getAttribute("AuthorizationData");

String appKey = map.get("appKey");

String cacheKey = "user\_H5Token:" + appKey;

LoginUser loginUser = (LoginUser) CacheManager.getInstance().getCacher().getObject(cacheKey);

envelope.setMsgId(msgId);

if (StringUtils.isEmpty(msgId)) {

envelope.setSuccess(false);

envelope.setMessage("params error");

return envelope;

}

if (loginUser==null) {

envelope.setSuccess(false);

envelope.setMessage(SystemCommonConstant.PLEASE\_LOGIN);

return envelope;

}

int userId = loginUser.getId();

Favorite favorite = new Favorite();

favorite.setUserId(userId);

List listSearch = designPlanRecommendFavoriteService.listFavorites(favorite);

if(listSearch == null || listSearch.size()<=0){

Favorite favoriteAdd = new Favorite();

favoriteAdd.setName("默认");

favoriteAdd.setUserId(userId);

designPlanRecommendFavoriteService.addFavorite(favoriteAdd);

}

List<Favorite> list = designPlanRecommendFavoriteService.listFavorites(favorite);

if(list == null || list.size()<=0){

envelope.setDatalist(new ArrayList<>());

envelope.setTotalCount(0);

return envelope;

}else{

Favorite defaultFavorite = null;

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

if("默认".equals(list.get(i).getName().trim())){

defaultFavorite = list.get(i);

list.remove(i);

}

}

if(defaultFavorite!=null){

list.add(0, defaultFavorite);

}

envelope.setDatalist(list);

envelope.setTotalCount(list.size());

return envelope;

}

}

/\*\*

\* add 将一个推荐方案 加入 收藏夹

\*/

@SuppressWarnings("rawtypes")

@RequestMapping(method = RequestMethod.POST, value = "/addDesingPlanCollect")

@ResponseBody

public ResponseEnvelope addDesingPlanCollect(@RequestBody DesignPlanRecommendedModel model, HttpServletRequest request) {

ResponseEnvelope envelope = new ResponseEnvelope();

String msgId = model.getMsgId();

Integer recommendId = model.getPlanRecommendedId();// 推荐方案的id

String favoriate\_businessId = model.getFid();// 收藏夹的业务id

//获取登录用户

@SuppressWarnings("unchecked")

HashMap<String, String> map = (HashMap<String, String>) request.getAttribute("AuthorizationData");

String appKey = map.get("appKey");

String cacheKey = "user\_H5Token:" + appKey;

LoginUser loginUser = (LoginUser) CacheManager.getInstance().getCacher().getObject(cacheKey);

envelope.setMsgId(msgId);

if (StringUtils.isEmpty(msgId) || StringUtils.isEmpty(favoriate\_businessId)) {

envelope.setSuccess(false);

envelope.setMessage("params error");

return envelope;

}

if (loginUser==null) {

envelope.setSuccess(false);

envelope.setMessage(SystemCommonConstant.PLEASE\_LOGIN);

return envelope;

}

DesignPlanRecommendFavoriteRef favoriteRef = new DesignPlanRecommendFavoriteRef();

favoriteRef.setFid(favoriate\_businessId);

favoriteRef.setRecommendId(recommendId.intValue());

int count = designPlanRecommendFavoriteService.existInFavorite(favoriteRef);

if (count > 0) {

envelope.setSuccess(false);

envelope.setMessage("请勿重复收藏");

return envelope;

}

boolean res = designPlanRecommendFavoriteService.moveInFavorite(favoriteRef);

envelope.setSuccess(res);

// if (res) {

// designPlanRecommendFavoriteService.get

// envelope.setMessage("");

// }else {

// envelope.setMessage("params error");

// }

return envelope;

}

/\*\*

\* del 把设计方案移除个人收藏夹

\*/

@SuppressWarnings("rawtypes")

@RequestMapping(method=RequestMethod.POST,value = "/delDesingPlanCollect")

@ResponseBody

public ResponseEnvelope del(@RequestBody DesignPlanRecommendedModel model, HttpServletRequest request){

ResponseEnvelope envelope = new ResponseEnvelope();

String msgId = model.getMsgId();

String recommendRef\_busyneesId = model.getFid();// 推荐方案的业务id

//获取登录用户

@SuppressWarnings("unchecked")

HashMap<String, String> map = (HashMap<String, String>) request.getAttribute("AuthorizationData");

String appKey = map.get("appKey");

String cacheKey = "user\_H5Token:" + appKey;

LoginUser loginUser = (LoginUser) CacheManager.getInstance().getCacher().getObject(cacheKey);

envelope.setMsgId(msgId);

if (StringUtils.isEmpty(msgId) || StringUtils.isEmpty(recommendRef\_busyneesId)) {

envelope.setSuccess(false);

envelope.setMessage("params error");

return envelope;

}

if (loginUser==null) {

envelope.setSuccess(false);

envelope.setMessage(SystemCommonConstant.PLEASE\_LOGIN);

return envelope;

}

DesignPlanRecommendFavoriteRef favoriteRef = new DesignPlanRecommendFavoriteRef();

favoriteRef.setBid(recommendRef\_busyneesId);

boolean res = designPlanRecommendFavoriteService.moveOutFavorite(favoriteRef);

envelope.setSuccess(res);

if (!res)

envelope.setMessage("params error");

return envelope;

}

/\*\*

\* get获取个人收藏夹量的设计方案id 同时支持针对 收藏夹里面显示的名称搜索 或者 通过搜索夹的业务id获取指定设计方案计划

\*/

@RequestMapping(method = RequestMethod.POST, value = "/getCollectList")

@ResponseBody

public ResponseEnvelope<DesignPlanRecommendedResult> favoritePlanRecommendedList(

@RequestBody FavoriteRecommendedModel model, HttpServletRequest request){

String msgId = model.getMsgId();

//获取登录用户信息

@SuppressWarnings("unchecked")

HashMap<String, String> map = (HashMap<String, String>) request.getAttribute("AuthorizationData");

String appKey = map.get("appKey");

String cacheKey = "user\_H5Token:" + appKey;

LoginUser loginUser = (LoginUser) CacheManager.getInstance().getCacher().getObject(cacheKey);

boolean valid = this.checkParam(msgId);

if(!valid) {

return new ResponseEnvelope<>(false,"参数不完整",msgId);

}

if(loginUser == null || loginUser.getId() <= 0 ){

return new ResponseEnvelope<DesignPlanRecommendedResult>(false,"请重新登录！",msgId);

}

model.setUserId(loginUser.getId());

model.setUserType(loginUser.getUserType());

return designPlanRecommendFavoriteService.favoritePlanRecommendedList(model);

}

/\*\*

\* 参数完整性判断

\* @param args

\* @return

\*/

private boolean checkParam(String... args) {

boolean result = true;

for(String arg :args) {

if(StringUtils.isEmpty(arg)) {

result = false;

return result;

}

}

return result;

}

/\*\*

\*getHasCollected 判断方案是否被收藏

\*/

@ResponseBody

@RequestMapping(method = RequestMethod.POST, value = "/getDesignPlanHasBeCollected")

public ResponseEnvelope getHasCollected(@RequestBody DesignPlanRecommendFavoriteRef ref) {

String fid = designPlanRecommendFavoriteService.getHasCollected(ref);

boolean success = true;

Object obj = fid;

if(fid == null){

success = false;

obj = "-1";

}

return new ResponseEnvelope(success, obj);

}

}

package com.nork.design.service;

import java.util.List;

import com.nork.common.model.ResponseEnvelope;

import com.nork.design.model.DesignPlanRecommendFavoriteRef;

import com.nork.design.model.DesignPlanRecommendedResult;

import com.nork.design.model.Favorite;

import com.nork.design.model.FavoriteRecommendedModel;

/\*\*

\*

\*

\* 项目名称：timeSpace 类名称：DesignPlanFavoriteService 类描述： 创建人：Timy.Liu 创建时间：2017-7-7

\* 上午11:08:20 修改人：Timy.Liu 修改时间：2017-7-7 上午11:08:20 修改备注：

\*

\* @version

\*

\*/

public interface DesignPlanRecommendFavoriteService {

/\*\*

\*

\*

\* addFavorite 方法描述： 添加收藏夹

\*

\* @param favorite

\* @return

\*

\* @return boolean 返回类型

\*

\* @Exception 异常对象

\*

\* @since CodingExample Ver(编码范例查看) 1.1

\*/

public String addFavorite(Favorite favorite);

/\*\*

\*

\*

\* updateFavorite 方法描述： 修改收藏夹

\*

\* @param favorite

\* @return

\*

\* @return boolean 返回类型

\*

\* @Exception 异常对象

\*

\* @since CodingExample Ver(编码范例查看) 1.1

\*/

public boolean updateFavorite(Favorite favorite);

/\*\*

\*

\*

\* listFavorites 方法描述： 列出个人所有的收藏夹

\*

\* @param favorite

\* @return

\*

\* @return List<Favorite> 返回类型

\*

\* @Exception 异常对象

\*

\* @since CodingExample Ver(编码范例查看) 1.1

\*/

public List<Favorite> listFavorites(Favorite favorite);

/\*\*

\*

\*

\* deleteFavorite 方法描述： 删除收藏夹

\*

\* @param favorite

\* @return

\*

\* @return boolean 返回类型

\*

\* @Exception 异常对象

\*

\* @since CodingExample Ver(编码范例查看) 1.1

\*/

public ResponseEnvelope deleteFavorite(Favorite favorite);

/\*\*

\*

\*

\* moveInFavorite 方法描述： 收藏推荐方案

\*

\* @param designPlanRecommendFavoriteRef

\* @return

\*

\* @return boolean 返回类型

\*

\* @Exception 异常对象

\*

\* @since CodingExample Ver(编码范例查看) 1.1

\*/

public boolean moveInFavorite(DesignPlanRecommendFavoriteRef designPlanRecommendFavoriteRef);

/\*\*

\*

\*

\* moveOutFavorite 方法描述： 取消收藏推荐方案

\*

\* @param designPlanRecommendFavoriteRef

\* @return

\*

\* @return boolean 返回类型

\*

\* @Exception 异常对象

\*

\* @since CodingExample Ver(编码范例查看) 1.1

\*/

public boolean moveOutFavorite(DesignPlanRecommendFavoriteRef designPlanRecommendFavoriteRef);

/\*\*

\*

\*

\* listMyDesignPlanRecommend 方法描述： 分页获取所有的我收藏过的推荐方案

\*

\* @param designPlanRecommendFavoriteRef

\* @return

\*

\* @return List 返回类型

\*

\* @Exception 异常对象

\*

\* @since CodingExample Ver(编码范例查看) 1.1

\*/

public List listMyDesignPlanRecommend(DesignPlanRecommendFavoriteRef designPlanRecommendFavoriteRef);

/\*\*

\* 获取方案推荐收藏夹列表

\* @param model

\* @return

\*/

public ResponseEnvelope<DesignPlanRecommendedResult> favoritePlanRecommendedList(FavoriteRecommendedModel model);

/\*\*

\* 通过bid 查询收藏夹

\* @param businessId

\* @return

\*/

public Favorite getFavoritesByBid(String businessId);

public int existInFavorite(DesignPlanRecommendFavoriteRef favoriteRef);

/\*\*

\* 通过recommendId 删 推荐收藏关系

\* @param id

\*/

public void deleteFavoriteRefByRecommendId(Integer recommendId);

/\*\*

\* 判断是否已经被此次 和 其他收藏夹收藏

\* @param favoriteRef

\* @return

\*/

public boolean existInFavoriteNew(DesignPlanRecommendFavoriteRef favoriteRef);

/\*\*

\* 根据方案id判断方案是否已经被收藏

\* @param ref

\* @return "0":未被收藏，"-1":方案id和用戶id为null，其他:已被收藏

\*/

public String getHasCollected(DesignPlanRecommendFavoriteRef ref);

}

package com.nork.design.service.impl;

import java.util.ArrayList;

import java.util.List;

import java.util.UUID;

import org.apache.log4j.Logger;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.nork.common.constant.SystemCommonConstant;

import com.nork.common.model.ResponseEnvelope;

import com.nork.common.util.StringUtils;

import com.nork.common.util.Utils;

import com.nork.design.dao.SysBusinessFavoriteMapper;

import com.nork.design.model.DesignPlanRecommendFavoriteRef;

import com.nork.design.model.DesignPlanRecommended;

import com.nork.design.model.DesignPlanRecommendedResult;

import com.nork.design.model.Favorite;

import com.nork.design.model.FavoriteRecommendedModel;

import com.nork.design.service.DesignPlanRecommendFavoriteService;

import com.nork.design.service.DesignPlanRecommendedServiceV2;

import com.nork.product.dao.AuthorizedConfigMapper;

import com.nork.product.model.AuthorizedConfig;

import com.nork.system.cache.SysDictionaryCacher;

import com.nork.system.dao.ResRenderPicMapper;

import com.nork.system.dao.SysDictionaryMapper;

import com.nork.system.model.SysDictionary;

import com.nork.system.service.SysDictionaryService;

/\*\*

\*

\*

\* 项目名称：timeSpace

\* 类名称：DesignPlanRecommendFavoriteServiceImpl

\* 类描述：

\* 创建人：Timy.Liu

\* 创建时间：2017-7-7 上午11:09:06

\* 修改人：Timy.Liu

\* 修改时间：2017-7-7 上午11:09:06

\* 修改备注：

\* @version

\*

\*/

@Service

public class DesignPlanRecommendFavoriteServiceImpl implements DesignPlanRecommendFavoriteService {

private static Logger logger = Logger.getLogger(DesignPlanRecommendFavoriteServiceImpl.class);

@Autowired

private SysBusinessFavoriteMapper sysBusinessFavoriteMapper;

@Autowired

private ResRenderPicMapper resRenderPicMapper;

@Autowired

private DesignPlanRecommendedServiceV2 designPlanRecommendedServiceV2;

@Autowired

private AuthorizedConfigMapper authorizedConfigMapper;

@Autowired

private SysDictionaryService sysDictionaryService;

/\* (non-Javadoc)

\* @see com.nork.design.service.DesignPlanFavoriteService#addFavorite(com.nork.design.model.Favorite)

\*/

@Override

public String addFavorite(Favorite favorite) {

if(favorite==null)

return "";

if(StringUtils.isEmpty(favorite.getName()) || favorite.getUserId()<=0)

return "";

List list= sysBusinessFavoriteMapper.listFavorites(favorite);

if(list!=null && list.size()>0)//同名存在，认为创建成功

return ((Favorite)list.get(0)).getBid();

if(favorite.getName().trim().length()>16)

favorite.setName(favorite.getName().substring(0, 16));

favorite.setBid(UUID.randomUUID().toString().replaceAll("-", ""));

sysBusinessFavoriteMapper.addFavorite(favorite);

return favorite.getBid();

}

/\* (non-Javadoc)

\* @see com.nork.design.service.DesignPlanFavoriteService#updateFavorite(com.nork.design.model.Favorite)

\*/

@Override

public boolean updateFavorite(Favorite favorite) {

if(favorite==null)

return false;

if(StringUtils.isEmpty(favorite.getName()) || favorite.getUserId()<=0 || StringUtils.isEmpty(favorite.getBid()))

return false;

List list= sysBusinessFavoriteMapper.listFavorites(favorite);

if(list!=null && list.size()>0)//同名存在，认为创建成功

{

logger.error("bid="+favorite.getBid()+"\_name="+favorite.getName()+"\_has\_exist");

return false;

}

if(favorite.getName().trim().length()>16)

favorite.setName(favorite.getName().substring(0, 16));

sysBusinessFavoriteMapper.updateFavorite(favorite);

return true;

}

/\* (non-Javadoc)

\* @see com.nork.design.service.DesignPlanFavoriteService#listFavorites(com.nork.design.model.Favorite)

\*/

@Override

public List<Favorite> listFavorites(Favorite favorite) {

if(favorite==null)

return null;

if(favorite.getUserId()<=0)

return null;

return sysBusinessFavoriteMapper.listFavorites(favorite);

}

/\* (non-Javadoc)

\* @see com.nork.design.service.DesignPlanFavoriteService#deleteFavorite(com.nork.design.model.Favorite)

\*/

@Override

public ResponseEnvelope deleteFavorite(Favorite favorite) {

ResponseEnvelope envelope = new ResponseEnvelope();

if(favorite == null){

envelope.setSuccess(false);

envelope.setMessage("params error");

return envelope;

}

if(StringUtils.isEmpty(favorite.getBid())){

envelope.setSuccess(false);

envelope.setMessage("params error");

return envelope;

}

if(favorite.getUserId()<=0){

envelope.setSuccess(false);

envelope.setMessage(SystemCommonConstant.PLEASE\_LOGIN);

return envelope;

}

Favorite favoriteSearch = this.getFavoritesByBid(favorite.getBid());

if(favoriteSearch!=null){

if(favoriteSearch.getUserId() != favorite.getUserId()){

envelope.setSuccess(false);

envelope.setMessage("无法删除他人收藏夹");

return envelope;

}

if("默认".equals(favoriteSearch.getName().trim())){

envelope.setSuccess(false);

envelope.setMessage("默认收藏夹无法删除");

return envelope;

}

}else{

envelope.setSuccess(false);

envelope.setMessage("已删除，请刷新列表");

return envelope;

}

sysBusinessFavoriteMapper.deleteFavorite(favorite);

sysBusinessFavoriteMapper.deleteFavoriteRefByFid(favorite);

envelope.setSuccess(true);

return envelope;

}

/\* (non-Javadoc)

\* @see com.nork.design.service.DesignPlanFavoriteService#moveInFavorite(com.nork.design.model.DesignPlanRecommendFavoriteRef)

\*/

@Override

public boolean moveInFavorite(DesignPlanRecommendFavoriteRef recommendFavoriteRef) {

if (recommendFavoriteRef == null)

return false;

if (recommendFavoriteRef.getRecommendId() <= 0 || StringUtils.isEmpty(recommendFavoriteRef.getFid()))

return false;

DesignPlanRecommended designPlanRecommended = designPlanRecommendedServiceV2

.get(recommendFavoriteRef.getRecommendId());

if (null == designPlanRecommended) {

logger.error(recommendFavoriteRef.getRecommendId() + "\_do\_not\_exist");

return false;

}

/\*int count = sysBusinessFavoriteMapper.existInFavorite(recommendFavoriteRef);

if (count > 0) {// 目的达到，认为成功

logger.error(recommendFavoriteRef.getRecommendId() + " has\_in\_favorite\_" + recommendFavoriteRef.getFid());

return true;

}\*/

recommendFavoriteRef.setBid(UUID.randomUUID().toString().replaceAll("-", ""));

sysBusinessFavoriteMapper.moveInFavorite(recommendFavoriteRef);

return true;

}

/\* (non-Javadoc)

\* @see com.nork.design.service.DesignPlanFavoriteService#moveOutFavorite(com.nork.design.model.DesignPlanRecommendFavoriteRef)

\*/

@Override

public boolean moveOutFavorite(DesignPlanRecommendFavoriteRef recommendFavoriteRef) {

if (recommendFavoriteRef == null)

return false;

if (StringUtils.isEmpty(recommendFavoriteRef.getBid()))

return false;

sysBusinessFavoriteMapper.moveOutFavorite(recommendFavoriteRef);

return true;

}

/\* (non-Javadoc)

\* @see com.nork.design.service.DesignPlanFavoriteService#listMyDesignPlanRecommend(com.nork.design.model.DesignPlanRecommendFavoriteRef)

\*/

@Override

public List listMyDesignPlanRecommend(DesignPlanRecommendFavoriteRef designPlanRecommendFavoriteRef) {

// TODO Auto-generated method stub

return null;

}

/\*\*

\* 获取方案推荐收藏夹列表

\* @param model

\* @return

\*/

@Override

public ResponseEnvelope<DesignPlanRecommendedResult> favoritePlanRecommendedList(FavoriteRecommendedModel model) {

Integer total = 0;

List<DesignPlanRecommendedResult> list = null;

String msgId = model.getMsgId();

String houseType = model.getHouseType();//空间类型

String livingName = model.getLivingName();//小区名称

String areaValue = model.getAreaValue();//面积

String designRecommendedStyleId = model.getDesignRecommendedStyleId();//风格id

Integer userId = model.getUserId();//用户id

Integer userType = model.getUserType();//用户类型

String limit = model.getLimit();

String start = model.getStart();

String favoriteBid = model.getFavoriteBid(); //方案推荐收藏夹id

if(userId == null || userId.intValue() == -1 || userId.intValue() == 0){

return new ResponseEnvelope<DesignPlanRecommendedResult>(false,"请重新登录！",msgId);

}

if(StringUtils.isEmpty(msgId)){

return new ResponseEnvelope<DesignPlanRecommendedResult>(false,"params error",msgId);

}

DesignPlanRecommended designPlanRecommended = new DesignPlanRecommended();

String sysVersionType = Utils.getPropertyName("app", "sys.version.type", "1").trim();/\* 1为外网 2 为内网 \*/

if ("2".equals(sysVersionType) && userType!=null && userType.intValue() == 1) { /\*内网内部用户的能看到测试发布中的\*/

designPlanRecommended.setIsInternalUser("yes");

}

if (StringUtils.isNotEmpty(favoriteBid)) { /\*空间功能类型 \*/

designPlanRecommended.setFavoriteBid(favoriteBid);

}

if (StringUtils.isNotEmpty(houseType)) { /\*空间功能类型 \*/

designPlanRecommended.setSpaceFunctionId(Integer.parseInt(houseType));

}

if(StringUtils.isNotEmpty(areaValue)){

designPlanRecommended.setAreaValue(areaValue);

}

if (StringUtils.isNotEmpty(livingName)) { /\* 小区名称 \*/

designPlanRecommended.setLivingName(livingName);

}

if (StringUtils.isNotEmpty(designRecommendedStyleId)) { /\* 推荐方案风格 \*/

designPlanRecommended.setDesignRecommendedStyleId(Integer.parseInt(designRecommendedStyleId));

}

if(StringUtils.isNotEmpty(limit)){

designPlanRecommended.setLimit(Integer.parseInt(limit));

}

if(StringUtils.isNotEmpty(start)){

designPlanRecommended.setStart(Integer.parseInt(start));

}

designPlanRecommended.setUserId(userId);

total = designPlanRecommendedServiceV2.getFavoritePlanRecommendedCount(designPlanRecommended);

if(total>0){

list = designPlanRecommendedServiceV2.getFavoritePlanRecommendedList(designPlanRecommended);

if (list != null && list.size() > 0) {

for (DesignPlanRecommendedResult result : list) {

SysDictionary sysDictionary = null;

if(Utils.enableRedisCache()){

sysDictionary = SysDictionaryCacher.getSysDictionaryByValue("houseType",result.getSpaceFunctionId());

}else{

sysDictionary = sysDictionaryService.getSysDictionaryByValue("houseType",result.getSpaceFunctionId());

}

if (sysDictionary != null) {

result.setHouseTypeName(sysDictionary.getName());

}

}

}

}

return new ResponseEnvelope<DesignPlanRecommendedResult>(total, list, msgId);

}

/\*\*

\* 通过bid 查询收藏夹

\* @param businessId

\* @return

\*/

@Override

public Favorite getFavoritesByBid(String businessId) {

return sysBusinessFavoriteMapper.getFavoritesByBid(businessId);

}

@Override

public int existInFavorite(DesignPlanRecommendFavoriteRef favoriteRef) {

return sysBusinessFavoriteMapper.existInFavorite(favoriteRef);

}

/\*\*

\* 通过recommendId 删 推荐收藏关系

\* @param recommendId

\*/

@Override

public void deleteFavoriteRefByRecommendId(Integer recommendId) {

sysBusinessFavoriteMapper.deleteFavoriteRefByRecommendId(recommendId);

}

@Override

public boolean existInFavoriteNew(DesignPlanRecommendFavoriteRef favoriteRef) {

if(favoriteRef == null){

return false;

}

if(favoriteRef.getRecommendId() < 1 || favoriteRef.getUserId() == null || favoriteRef.getUserId().intValue() < 1){

return false;

}

int count = sysBusinessFavoriteMapper.existInFavoriteNew(favoriteRef);

if(count > 0 ){

return false;

}

return true;

}

/\*\*

\* 通过recommendId 判断是否被收藏

\* @param ref

\*/

@Override

public String getHasCollected(DesignPlanRecommendFavoriteRef ref) {

if(ref == null){

return "-1";//没有设计方案id和用戶id

}

String fid = sysBusinessFavoriteMapper.getHasCollected(ref);

if(fid == null){

return "0";//0表示没有被收藏

}

return fid;

}

}

package com.nork.design.service;

import java.util.List;

import java.util.Map;

import java.util.Set;

import com.nork.common.model.LoginUser;

import com.nork.common.model.ResponseEnvelope;

import com.nork.design.model.DesignPlanBrand;

import com.nork.design.model.DesignPlanRecommended;

import com.nork.design.model.DesignPlanRecommendedCheck;

import com.nork.design.model.DesignPlanRecommendedProductResult;

import com.nork.design.model.DesignPlanRecommendedResult;

import com.nork.design.model.DesignPlanRenderScene;

import com.nork.design.model.PlanRecommendedListModel;

import com.nork.design.model.ReleaseDesignPlanModel;

import com.nork.design.model.input.FranchiserPlanListDTO;

import com.nork.design.model.input.PlanRecommendedListForFranchiserDTO;

import com.nork.design.model.output.FranchiserPlanListVO;

import com.nork.product.model.BaseBrand;

import com.nork.product.model.BaseProductStyle;

import com.nork.product.model.search.BaseBrandSearch;

import com.nork.system.model.ResRenderPic;

public interface DesignPlanRecommendedServiceV2 {

public DesignPlanRecommended add(DesignPlanRecommended designPlanRecommended);

public int update(DesignPlanRecommended designPlanRecommended);

public int delete(Integer id);

public DesignPlanRecommended get(Integer id);

public List<DesignPlanRecommended> getList(DesignPlanRecommended designPlanRecommended);

public int getCount(DesignPlanRecommended designPlanRecommended);

/\*\*

\* 《发布界面》

\* 已经选择 的品牌列表 接口

\* @param request

\* @param response

\* @return

\*/

public ResponseEnvelope<BaseBrand> myBrandList(String msgId, String thumbId);

/\*\*

\* 发布方法

\* @param model

\* @param msgId

\* @param loginUser

\* @return

\*/

public ResponseEnvelope<DesignPlanRecommendedResult> recommendDesignPlan(ReleaseDesignPlanModel model, String msgId,

LoginUser loginUser);

/\*\*

\* 用于正式发布、测试发布、改造，复制 校验方法

\* 需要校验的条件

\* 1.样板房 空间 是否为发布中

\* 2.验证设计方案产品是否发布

\* 3.品牌不能重复添加

\* 4.是否有封面，

\* 5.是否有720渲染，

\* 6.是否有照片级渲染

\* @param designPlan

\* @return

\*/

public DesignPlanRecommendedCheck planFormalIsReleaseCheck(String functionType,DesignPlanRenderScene designPlanRenderScene);

/\*\*

\*

\* 方案推荐 列表 数据

\* @param isMainList

\* @param creator

\*@param brandName

\* @param houseType

\* @param livingName

\* @param areaValue

\* @param designRecommendedStyleId

\* @param msgId

\* @param loginUser @return

\*/

public Object getPlanRecommendedList(PlanRecommendedListModel model);

/\*\*

\* 删除已经选择的品牌

\* @param msgId

\* @param designPlanBrandId

\* @return

\*/

public ResponseEnvelope<DesignPlanBrand> deletedBrand(String msgId, String designPlanBrandId,LoginUser loginUser);

/\*\*

\* 获取设计方案风格 接口

\* @param msgId

\* @param planRecommendedId

\* @param houseType

\* @return

\*/

public ResponseEnvelope<BaseProductStyle> getDesignStyleList(String msgId, String thumbId,String planRecommendedId,

String houseType);

/\*\*

\* 获取设计方案风格 接口 新

\* @param msgId

\* @param planRecommendedId

\* @param houseType

\* @return

\*/

public ResponseEnvelope<BaseProductStyle> getDesignStyleListNew(String msgId, String thumbId ,String planRecommendedId, String houseType);

/\*\*

\* 方案推荐总条数

\* @param designPlanBrand

\* @param brandIds 必传

\* @return

\*/

public Integer getPlanRecommendedCount(DesignPlanRecommended designPlanRecommended);

/\*\*

\* 方案推荐数据

\* @param brandIds 必传

\* @param designPlanBrand

\* @return

\*/

public List<DesignPlanRecommendedResult> getPlanRecommendedList(DesignPlanRecommended designPlanRecommended);

/\*\*

\* 审核列表

\* @param msgId

\* @param userId

\* @param spaceFunctionId

\* @param planName

\* @return

\*/

public ResponseEnvelope<DesignPlanRecommendedResult> designPlanRecommendedCheckList(PlanRecommendedListModel model);

/\*\*

\* 审核接口

\* @param msgId

\* @param planId

\* @param planRecommendedId

\* @param userId

\* @param isReleaseState

\* @return

\*/

public ResponseEnvelope<DesignPlanRecommended> designPlanRecommendedCheck(String msgId, String planRecommendedId,

LoginUser loginUser, String isReleaseState,String failCause);

/\*\*

\* 方案推荐详情

\* @param msgId

\* @param planId

\* @param planRecommendedId

\* @return

\*/

/\*public ResponseEnvelope<DesignPlanRecommended> designPlanRecommendedDetails\_old(String msgId,String planRecommendedId);\*/

/\*\*

\* 取消发布接口

\* @param msgId

\* @param loginUser

\* @param planRecommendedId

\* @return

\*/

public ResponseEnvelope<DesignPlanRecommended> cancelRelease(String msgId,LoginUser loginUser, String planRecommendedId);

/\*\*

\* 发布用的品牌列表 接口

\* @param msgId

\* @return

\*/

public ResponseEnvelope<BaseBrand> myBrandList(String msgId,BaseBrandSearch baseBrandSearch);

/\*\*

\* 方案推荐详情

\* @param msgId

\* @param planRecommendedId

\* @return

\*/

public ResponseEnvelope<DesignPlanRecommended> designPlanRecommendedDetails(String msgId, String planRecommendedId,LoginUser loginUser);

/\*\*

\* 提交审核

\* @param msgId

\* @param loginUser

\* @param planRecommendedId

\* @param styleId

\* @param brandIds

\* @param recommendedType 1:普通方案;2:智能方案

\* @return

\*/

public ResponseEnvelope<DesignPlanRecommended> submitCheck(String msgId, LoginUser loginUser,

String planRecommendedId,String spaceAreas, String brandIds, String styleId

/\*, Integer recommendedType\*/

);

/\*\*

\* 方案推荐详情 产品列表

\* @param msgId

\* @param planRecommendedId

\* @param loginUser

\* @return

\*/

public ResponseEnvelope<DesignPlanRecommendedProductResult> getDesignPlanRecommendedProductList(String msgId,

String planRecommendedId, LoginUser loginUser);

/\*\*

\* 设置封面

\* @param msgId

\* @param loginUser

\* @param planRecommendedId

\* @param picId

\* @return

\*/

public ResponseEnvelope<DesignPlanRecommended> updateCoverPic(String msgId, LoginUser loginUser, String planRecommendedId, String picId);

/\*\*

\* 修改方案推荐

\* @param msgId

\* @param loginUser

\* @param planRecommendedId

\* @param remark

\* @param planName

\* @return

\*/

public ResponseEnvelope<DesignPlanRecommended> updatePlanRecommended(String msgId, LoginUser loginUser,

String planRecommendedId, String remark, String planName);

/\*\*

\* 详情查看

\* @param msgId

\* @param smallPicId

\* @return

\*/

public ResponseEnvelope<ResRenderPic> detailsSee(String msgId, String picId,String detailsSeeType,LoginUser loginUser);

/\*\*

\* 获取方案推荐收藏夹数据量

\* @param designPlanRecommended

\* @return

\*/

public Integer getFavoritePlanRecommendedCount(DesignPlanRecommended designPlanRecommended);

/\*\*

\* 获取方案推荐收藏夹列表

\* @param designPlanRecommended

\* @return

\*/

public List<DesignPlanRecommendedResult> getFavoritePlanRecommendedList(

DesignPlanRecommended designPlanRecommended);

/\*\*

\* 获取空间类型面积列表

\* @author xiaoxc

\* @param spaceFunctionId

\* @return

\*/

public ResponseEnvelope getSpaceAreaList(Integer spaceFunctionId,String msgId);

public List<DesignPlanRecommendedResult> getPlanRecommendedListMobile(DesignPlanRecommended designPlanRecommended) ;

public ResponseEnvelope<DesignPlanRecommendedResult> getPlanRecommendedListMobile(PlanRecommendedListModel model) ;

public Integer getPlanRecommendedCountMobile(DesignPlanRecommended designPlanRecommended);

/\*\*

\* 分享发布 好 一键装修测试发布 前校验

\* @param msgId

\* @param thumbId

\* @param recommendedType

\* @return

\*/

public ResponseEnvelope<DesignPlanRecommendedResult> planFormalIsReleaseCheck(String msgId, String thumbId,String recommendedType);

/\*\*

\* 在样板房推荐页面显示推荐方案审核状态

\* @param msgId

\* @return

\*/

ResponseEnvelope getRecommendedPlanCheckState(String msgId);

/\*\*

\* 根据空间类型获取推荐方案ids

\* @author xiaoxc

\* @param spaceFunctionId

\* @return

\*/

List<Integer> findRecommendedPlanIdListBySpaceFunctionId(Integer spaceFunctionId);

/\*\*

\* 获取空间布局类型

\* @author xiaoxc

\* @param list

\* @param type

\* @return

\*/

String getLayoutType(List<String> list, String type);

/\*\*

\* 查询公开方案size 定制方法

\*

\* @author huangsongbo

\* @date 2018.3.28

\* @param dto

\* @return

\*/

public int getPlanRecommendedListSizeByDTO(PlanRecommendedListForFranchiserDTO dto);

/\*\*

\* 查询公开方案list 定制方法

\*

\* @author huangsongbo

\* @date 2018.3.28

\* @param dto

\* @return

\*/

public List<DesignPlanRecommendedResult> getPlanRecommendedListByDTO(PlanRecommendedListForFranchiserDTO dto);

public int getFranchiserPlanListSizeByDTO(FranchiserPlanListDTO dto);

public List<FranchiserPlanListVO> getFranchiserPlanListByDTO(FranchiserPlanListDTO dto);

void setBrandIds(List<Integer> brandIdList, Integer designPlanRecommendedId, Long companyId, LoginUser loginUser);

/\*\*

\* 验证用户有没有发布智能方案/普通方案的权限

\*

\* @author huangsongbo

\* @date 2018.4.9

\*

\* @param roles

\* @param shelfStatusList

\* @return

\*/

public boolean hasReleasePermission(Set<String> roles, List<String> shelfStatusList);

/\*\*

\* 验证用户有没有发布智能方案/普通方案的权限

\*

\* @author huangsongbo

\* @date 2018.4.13

\*

\* @param roles

\* @param recommendedType

\* @return

\*/

public boolean hasReleasePermissionParamRecommendedType(Set<String> roles, Integer recommendedType);

Integer getPlatformIdByPlatformCode(String code);

public boolean hasReleasePermissionParamRecommendedType(Integer recommendedType);

public ResponseEnvelope<DesignPlanRecommendedResult> decorateTestRecommend(ReleaseDesignPlanModel model, DesignPlanRenderScene designPlanRenderScene,

List<String> brandIdList,String msgId,LoginUser loginUser);

public ResponseEnvelope<DesignPlanRecommendedResult> shareRecommend(ReleaseDesignPlanModel model, DesignPlanRenderScene designPlanRenderScene,

List<String> brandIdList,String msgId,LoginUser loginUser);

}

<?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.nork.design.dao.DesignPlanMapper">

<!--<cache type="org.mybatis.caches.ehcache.LoggingEhcache"/>-->

<!-- \*\*插入定义\*\* -->

<insert id="insertSelective" parameterType="com.nork.design.model.DesignPlan" useGeneratedKeys="true" keyProperty="id">

insert into design\_plan

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

<if test="planCode!= null">plan\_code, </if>

<if test="planName!= null">plan\_name, </if>

<if test="planSource!= null">plan\_source, </if>

<if test="houseId!= null">house\_id, </if>

<if test="livingId!= null">living\_id, </if>

<if test="residentialUnitsName!= null">residential\_units\_name,</if>

<if test="userId!= null">user\_id, </if>

<if test="mediaType!= null">media\_type, </if>

<if test="designSourceType!= null">design\_source\_type, </if>

<if test="designId!= null">design\_id, </if>

<if test="designTemplateId!= null">design\_template\_id, </if>

<if test="designStyleId!= null">design\_style\_id, </if>

<if test="picId!= null">pic\_id, </if>

<if test="modelId!= null">model\_id, </if>

<if test="model3dId!= null">model\_3d\_id, </if>

<if test="modelU3dId!= null">model\_u3d\_id, </if>

<if test="webModelU3dId!= null">web\_model\_u3d\_id, </if>

<if test="iosModelU3dId!= null">ios\_model\_u3d\_id, </if>

<if test="ipadModelU3dId!= null">ipad\_model\_u3d\_id, </if>

<if test="pcModelU3dId!= null">pc\_model\_u3d\_id, </if>

<if test="androidModelU3dId!= null">android\_model\_u3d\_id, </if>

<if test="macBookpcModelU3dId!= null">macBookpc\_model\_u3d\_id, </if>

<if test="configFileId!= null">config\_file\_id, </if>

<if test="planDesc!= null">plan\_desc, </if>

<if test="spaceCommonId!= null">space\_common\_id, </if>

<if test="sysCode!= null">sys\_code, </if>

<if test="creator!= null">creator, </if>

<if test="gmtCreate!= null">gmt\_create, </if>

<if test="modifier!= null">modifier, </if>

<if test="gmtModified!= null">gmt\_modified, </if>

<if test="isDeleted!= null">is\_deleted, </if>

<if test="remark!= null">remark, </if>

<!--<if test="eveningFileId!= null">evening\_file\_id, </if>

<if test="dawnFileId!= null">dawn\_file\_id, </if>

<if test="nightFileId!= null">night\_file\_id, </if>-->

<if test="shareTotal!= null">share\_total, </if>

<if test="isOpen!= null">is\_open, </if>

<if test="isChange!= null">is\_change, </if>

<if test="isDecorated!= null">is\_decorated, </if>

<if test="draftState!= null">draft\_state, </if>

<if test="baiMoState!= null">baiMo\_state, </if>

<if test="stuffFinishState!= null">stuff\_finish\_state, </if>

<if test="decorateFinishState!= null">decorate\_finish\_state, </if>

<if test="coverPicId!= null">cover\_pic\_Id, </if>

<!-- <if test="isRelease!= null">is\_release, </if> -->

<if test="releaseTime!= null">release\_time, </if>

<if test="designRecommendedStyleId!= null">design\_recommended\_style\_id, </if>

<if test="isRecommended!= null">is\_recommended, </if>

<if test="isDefaultDecorate!= null">is\_default\_decorate, </if>

<if test="planNumber!= null">plan\_number, </if>

<if test="designSceneId!= null">design\_scene\_id, </if>

<if test="visible!= null">visible, </if>

<if test="sceneModified!= null">scene\_modified,</if>

<if test="recommendedPlanId!= null">recommended\_plan\_id, </if>

</trim>

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

<if test="planCode!= null"> #{planCode,jdbcType=VARCHAR}, </if>

<if test="planName!= null"> #{planName,jdbcType=VARCHAR}, </if>

<if test="planSource!= null"> #{planSource,jdbcType=VARCHAR},</if>

<if test="houseId!= null"> #{houseId,jdbcType=INTEGER}, </if>

<if test="livingId!= null"> #{livingId,jdbcType=INTEGER}, </if>

<if test="residentialUnitsName!= null">#{residentialUnitsName,jdbcType=VARCHAR},</if>

<if test="userId!= null"> #{userId,jdbcType=INTEGER}, </if>

<if test="mediaType!= null"> #{mediaType,jdbcType=INTEGER}, </if>

<if test="designSourceType!= null"> #{designSourceType,jdbcType=INTEGER}, </if>

<if test="designId!= null"> #{designId,jdbcType=INTEGER}, </if>

<if test="designTemplateId!= null"> #{designTemplateId,jdbcType=INTEGER}, </if>

<if test="designStyleId!= null"> #{designStyleId,jdbcType=INTEGER}, </if>

<if test="picId!= null"> #{picId,jdbcType=INTEGER}, </if>

<if test="modelId!= null"> #{modelId,jdbcType=INTEGER}, </if>

<if test="model3dId!= null"> #{model3dId,jdbcType=INTEGER}, </if>

<if test="modelU3dId!= null"> #{modelU3dId,jdbcType=INTEGER}, </if>

<if test="webModelU3dId!= null"> #{webModelU3dId,jdbcType=INTEGER}, </if>

<if test="iosModelU3dId!= null"> #{iosModelU3dId,jdbcType=INTEGER}, </if>

<if test="ipadModelU3dId!= null"> #{ipadModelU3dId,jdbcType=INTEGER}, </if>

<if test="pcModelU3dId!= null"> #{pcModelU3dId,jdbcType=INTEGER}, </if>

<if test="androidModelU3dId!= null"> #{androidModelU3dId,jdbcType=INTEGER}, </if>

<if test="macBookpcModelU3dId!= null"> #{macBookpcModelU3dId,jdbcType=INTEGER}, </if>

<if test="configFileId!= null"> #{configFileId,jdbcType=INTEGER}, </if>

<if test="spaceCommonId!= null"> #{spaceCommonId,jdbcType=INTEGER}, </if>

<if test="planDesc!= null"> #{planDesc,jdbcType=VARCHAR}, </if>

<if test="sysCode!= null"> #{sysCode,jdbcType=VARCHAR}, </if>

<if test="creator!= null"> #{creator,jdbcType=VARCHAR}, </if>

<if test="gmtCreate!= null"> #{gmtCreate,jdbcType=TIMESTAMP}, </if>

<if test="modifier!= null"> #{modifier,jdbcType=VARCHAR}, </if>

<if test="gmtModified!= null"> #{gmtModified,jdbcType=TIMESTAMP}, </if>

<if test="isDeleted!= null"> #{isDeleted,jdbcType=INTEGER}, </if>

<if test="remark!= null"> #{remark,jdbcType=VARCHAR}, </if>

<!--<if test="eveningFileId!= null"> #{eveningFileId,jdbcType=INTEGER}, </if>

<if test="dawnFileId!= null"> #{dawnFileId,jdbcType=INTEGER}, </if>

<if test="nightFileId!= null"> #{nightFileId,jdbcType=INTEGER}, </if>-->

<if test="shareTotal!= null">#{shareTotal,jdbcType=INTEGER}, </if>

<if test="isOpen!= null">#{isOpen,jdbcType=INTEGER}, </if>

<if test="isChange!= null">#{isChange,jdbcType=INTEGER}, </if>

<if test="isDecorated!= null">#{isDecorated,jdbcType=INTEGER}, </if>

<if test="draftState!= null">#{draftState,jdbcType=INTEGER}, </if>

<if test="baiMoState!= null">#{baiMoState,jdbcType=INTEGER}, </if>

<if test="stuffFinishState!= null">#{stuffFinishState,jdbcType=INTEGER}, </if>

<if test="decorateFinishState!= null">#{decorateFinishState,jdbcType=INTEGER}, </if>

<if test="coverPicId!= null">#{coverPicId,jdbcType=INTEGER}, </if>

<!-- <if test="isRelease!= null">#{isRelease,jdbcType=INTEGER}, </if> -->

<if test="releaseTime!= null">#{releaseTime,jdbcType=TIMESTAMP}, </if>

<if test="designRecommendedStyleId!= null">#{designRecommendedStyleId,jdbcType=INTEGER}, </if>

<if test="isRecommended!= null">#{isRecommended,jdbcType=INTEGER}, </if>

<if test="isDefaultDecorate!= null">#{isDefaultDecorate,jdbcType=INTEGER}, </if>

<if test="planNumber!= null">#{planNumber,jdbcType=VARCHAR}, </if>

<if test="designSceneId!= null">#{designSceneId,jdbcType=INTEGER}, </if>

<if test="visible!= null">#{visible,jdbcType=INTEGER}, </if>

<if test="sceneModified!= null">#{sceneModified,jdbcType=INTEGER}, </if>

<if test="recommendedPlanId!= null"> #{recommendedPlanId,jdbcType=INTEGER}, </if>

</trim>

</insert>

<!-- \*\*更新定义\*\* -->

<update id="updateByPrimaryKeySelective" parameterType="com.nork.design.model.DesignPlan">

update design\_plan

<set>

<if test="planCode!= null"> plan\_code = #{planCode,jdbcType=VARCHAR}, </if>

<if test="planName!= null"> plan\_name = #{planName,jdbcType=VARCHAR}, </if>

<if test="userId!= null"> user\_id = #{userId,jdbcType=INTEGER}, </if>

<if test="mediaType!= null"> media\_type = #{mediaType,jdbcType=INTEGER}, </if>

<if test="designSourceType!= null"> design\_source\_type = #{designSourceType,jdbcType=INTEGER}, </if>

<if test="designId!= null"> design\_id = #{designId,jdbcType=INTEGER}, </if>

<if test="designTemplateId!= null"> design\_template\_id = #{designTemplateId,jdbcType=INTEGER}, </if>

<if test="designStyleId!= null"> design\_style\_id = #{designStyleId,jdbcType=INTEGER}, </if>

<if test="picId!= null"> pic\_id = #{picId,jdbcType=INTEGER}, </if>

<if test="modelId!= null"> model\_id = #{modelId,jdbcType=INTEGER}, </if>

<if test="model3dId!= null"> model\_3d\_id = #{model3dId,jdbcType=INTEGER}, </if>

<if test="modelU3dId!= null"> model\_u3d\_id = #{modelU3dId,jdbcType=INTEGER}, </if>

<if test="webModelU3dId!= null"> web\_model\_u3d\_id = #{webModelU3dId,jdbcType=INTEGER}, </if>

<if test="iosModelU3dId!= null"> ios\_model\_u3d\_id = #{iosModelU3dId,jdbcType=INTEGER}, </if>

<if test="pcModelU3dId!= null"> pc\_model\_u3d\_id = #{pcModelU3dId,jdbcType=INTEGER}, </if>

<if test="ipadModelU3dId!= null"> ipad\_model\_u3d\_id = #{ipadModelU3dId,jdbcType=INTEGER}, </if>

<if test="androidModelU3dId!= null"> android\_model\_u3d\_id = #{androidModelU3dId,jdbcType=INTEGER}, </if>

<if test="macBookpcModelU3dId!= null"> macBookpc\_model\_u3d\_id = #{macBookpcModelU3dId,jdbcType=INTEGER}, </if>

<if test="configFileId!= null"> config\_file\_id = #{configFileId,jdbcType=INTEGER}, </if>

<if test="spaceCommonId!= null"> space\_common\_id = #{spaceCommonId,jdbcType=INTEGER}, </if>

<if test="planDesc!= null"> plan\_desc = #{planDesc,jdbcType=VARCHAR}, </if>

<if test="sysCode!= null"> sys\_code = #{sysCode,jdbcType=VARCHAR}, </if>

<if test="creator!= null"> creator = #{creator,jdbcType=VARCHAR}, </if>

<if test="gmtCreate!= null"> gmt\_create = #{gmtCreate,jdbcType=TIMESTAMP}, </if>

<if test="modifier!= null"> modifier = #{modifier,jdbcType=VARCHAR}, </if>

<if test="gmtModified!= null"> gmt\_modified = #{gmtModified,jdbcType=TIMESTAMP}, </if>

<if test="isDeleted!= null"> is\_deleted = #{isDeleted,jdbcType=INTEGER}, </if>

<if test="remark!= null"> remark = #{remark,jdbcType=VARCHAR}, </if>

<!--<if test="eveningFileId!= null"> evening\_file\_id = #{eveningFileId,jdbcType=INTEGER}, </if>

<if test="dawnFileId!= null"> dawn\_file\_id = #{dawnFileId,jdbcType=INTEGER}, </if>

<if test="nightFileId!= null"> night\_file\_id = #{nightFileId,jdbcType=INTEGER}, </if>-->

<if test="shareTotal!= null"> share\_total = #{shareTotal,jdbcType=INTEGER}, </if>

<if test="isOpen!= null"> is\_open = #{isOpen,jdbcType=INTEGER}, </if>

<if test="isChange!= null"> is\_change = #{isChange,jdbcType=INTEGER}, </if>

<if test="isDecorated!= null"> is\_decorated = #{isDecorated,jdbcType=INTEGER}, </if>

<if test="isDecorated!= null"> is\_decorated = #{isDecorated,jdbcType=INTEGER}, </if>

<if test="draftState!= null"> draft\_state = #{draftState,jdbcType=INTEGER}, </if>

<if test="baiMoState!= null"> baiMo\_state = #{baiMoState,jdbcType=INTEGER}, </if>

<if test="stuffFinishState!= null"> stuff\_finish\_state = #{stuffFinishState,jdbcType=INTEGER}, </if>

<if test="decorateFinishState!= null"> decorate\_finish\_state = #{decorateFinishState,jdbcType=INTEGER}, </if>

<if test="coverPicId!= null">cover\_pic\_Id = #{coverPicId,jdbcType=INTEGER}, </if>

<!-- <if test="isRelease!= null">is\_release = #{isRelease,jdbcType=INTEGER}, </if> -->

<if test="releaseTime!= null">release\_time =#{releaseTime,jdbcType=TIMESTAMP}, </if>

<if test="designRecommendedStyleId!= null">design\_recommended\_style\_id = #{designRecommendedStyleId,jdbcType=INTEGER}, </if>

<if test="effectsConfig!= null">effects\_config = #{effectsConfig,jdbcType=VARCHAR}, </if>

<if test="isRecommended!= null">is\_recommended = #{isRecommended,jdbcType=INTEGER}, </if>

<if test="isDefaultDecorate!= null">is\_default\_decorate = #{isDefaultDecorate,jdbcType=INTEGER}, </if>

<if test="planNumber!= null">plan\_number = #{planNumber,jdbcType=VARCHAR}, </if>

<if test="designSceneId!= null">design\_scene\_id = #{designSceneId,jdbcType=INTEGER}, </if>

<if test="visible!= null">visible = #{visible,jdbcType=INTEGER}, </if>

<if test="sceneModified!= null and sceneModified > 0 ">scene\_modified = #{sceneModified,jdbcType=INTEGER}, </if>

</set>

where id = #{id,jdbcType=INTEGER}

</update>

<!-- \*\*删除定义\*\* -->

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

delete from design\_plan

where id = #{id,jdbcType=INTEGER}

</delete>

<!-- \*\*常量定义\*\* -->

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

id,is\_recommended,is\_default\_decorate,plan\_number,design\_recommended\_style\_id,release\_time,cover\_pic\_Id,plan\_code,plan\_name,plan\_source,house\_id,living\_id,residential\_units\_name,user\_id,media\_type,design\_source\_type,design\_id,share\_total,is\_open,is\_change,

is\_decorated,draft\_state,baiMo\_state,stuff\_finish\_state,decorate\_finish\_state,design\_template\_id,design\_style\_id,pic\_id,model\_id,model\_3d\_id,model\_u3d\_id,web\_model\_u3d\_id,ios\_model\_u3d\_id,ipad\_model\_u3d\_id,pc\_model\_u3d\_id,

android\_model\_u3d\_id,macBookpc\_model\_u3d\_id,config\_file\_id,space\_common\_id,plan\_desc,sys\_code,creator,gmt\_create,modifier,gmt\_modified,is\_deleted,remark,evening\_file\_id,dawn\_file\_id,night\_file\_id,effects\_config,

design\_scene\_id,visible,scene\_modified,recommended\_plan\_id

</sql>

<!-- \*\*结果定义\*\* -->

<resultMap id="AllResultMap" type="com.nork.design.model.DesignPlan">

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

<result column="plan\_code" property="planCode" jdbcType="VARCHAR" />

<result column="plan\_name" property="planName" jdbcType="VARCHAR" />

<result column="plan\_source" property="planSource" jdbcType="VARCHAR"/>

<result column="house\_id" property="houseId" jdbcType="INTEGER"/>

<result column="living\_id" property="livingId" jdbcType="INTEGER"/>

<result column="residential\_units\_name" property="residentialUnitsName" jdbcType="VARCHAR"/>

<result column="user\_id" property="userId" jdbcType="INTEGER" />

<result column="media\_type" property="mediaType" jdbcType="INTEGER" />

<result column="design\_source\_type" property="designSourceType" jdbcType="INTEGER" />

<result column="design\_id" property="designId" jdbcType="INTEGER" />

<result column="design\_template\_id" property="designTemplateId" jdbcType="INTEGER" />

<result column="design\_style\_id" property="designStyleId" jdbcType="INTEGER" />

<result column="pic\_id" property="picId" jdbcType="INTEGER" />

<result column="model\_id" property="modelId" jdbcType="INTEGER" />

<result column="model\_3d\_id" property="model3dId" jdbcType="INTEGER" />

<result column="model\_u3d\_id" property="modelU3dId" jdbcType="INTEGER" />

<result column="web\_model\_u3d\_id" property="webModelU3dId" jdbcType="INTEGER" />

<result column="ios\_model\_u3d\_id" property="iosModelU3dId" jdbcType="INTEGER" />

<result column="ipad\_model\_u3d\_id" property="ipadModelU3dId" jdbcType="INTEGER" />

<result column="pc\_model\_u3d\_id" property="pcModelU3dId" jdbcType="INTEGER" />

<result column="android\_model\_u3d\_id" property="androidModelU3dId" jdbcType="INTEGER" />

<result column="macBookpc\_model\_u3d\_id" property="macBookpcModelU3dId" jdbcType="INTEGER" />

<result column="config\_file\_id" property="configFileId" jdbcType="INTEGER" />

<result column="space\_common\_id" property="spaceCommonId" jdbcType="INTEGER" />

<result column="space\_function\_id" property="spaceFunctionId" jdbcType="INTEGER" />

<result column="space\_areas" property="spaceAreas" jdbcType="VARCHAR" />

<result column="plan\_desc" property="planDesc" jdbcType="VARCHAR" />

<result column="sys\_code" property="sysCode" jdbcType="VARCHAR" />

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

<result column="gmt\_create" property="gmtCreate" jdbcType="TIMESTAMP" />

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

<result column="gmt\_modified" property="gmtModified" jdbcType="TIMESTAMP" />

<result column="is\_deleted" property="isDeleted" jdbcType="INTEGER" />

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

<!--<result column="evening\_file\_id" property="eveningFileId" jdbcType="INTEGER" />

<result column="dawn\_file\_id" property="dawnFileId" jdbcType="INTEGER" />

<result column="night\_file\_id" property="nightFileId" jdbcType="INTEGER" />-->

<result column="share\_total" property="shareTotal" jdbcType="INTEGER" />

<result column="is\_open" property="isOpen" jdbcType="INTEGER" />

<result column="is\_change" property="isChange" jdbcType="INTEGER" />

<result column="is\_decorated" property="isDecorated" jdbcType="INTEGER" />

<result column="draft\_state" property="draftState" jdbcType="INTEGER" />

<result column="baiMo\_state" property="baiMoState" jdbcType="INTEGER" />

<result column="stuff\_finish\_state" property="stuffFinishState" jdbcType="INTEGER" />

<result column="decorate\_finish\_state" property="decorateFinishState" jdbcType="INTEGER" />

<result column="cover\_pic\_Id" property="coverPicId" jdbcType="INTEGER" />

<!-- <result column="is\_release" property="isRelease" jdbcType="INTEGER" /> -->

<result column="release\_time" property="releaseTime" jdbcType="TIMESTAMP" />

<result column="design\_recommended\_style\_id" property="designRecommendedStyleId" jdbcType="INTEGER" />

<result column="effects\_config" property="effectsConfig" jdbcType="VARCHAR" />

<result column="is\_recommended" property="isRecommended" jdbcType="INTEGER" />

<result column="is\_default\_decorate" property="isDefaultDecorate" jdbcType="INTEGER" />

<result column="plan\_number" property="planNumber" jdbcType="VARCHAR" />

<result column="design\_scene\_id" property="designSceneId" jdbcType="INTEGER" />

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

<result column="scene\_modified" property="sceneModified" jdbcType="INTEGER" />

<result column="recommended\_plan\_id" property="recommendedPlanId" jdbcType="INTEGER" />

</resultMap>

<!-- \*\*查询定义\*\* -->

<!-- 按主键查询 -->

<select id="selectByPrimaryKey" resultMap="AllResultMap"

parameterType="java.lang.Integer">

select

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

from design\_plan

where id = #{id,jdbcType=INTEGER}

</select>

<!-- 查询所有，可排序 -->

<select id="selectList" resultMap="AllResultMap"

parameterType="com.nork.design.model.DesignPlan">

select

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

from design\_plan

where 1=1

<if test="isRecommended!= null">and is\_recommended = #{isRecommended,jdbcType=INTEGER} </if>

<if test="isDefaultDecorate!= null">and is\_default\_decorate = #{isDecorate,jdbcType=INTEGER} </if>

<if test="planNumber!= null">and plan\_number = #{planNumber,jdbcType=VARCHAR} </if>

<if test="designRecommendedStyleId!= null">and design\_recommended\_style\_id = #{designRecommendedStyleId,jdbcType=INTEGER} </if>

<if test="releaseTime!= null">and release\_time =#{releaseTime,jdbcType=TIMESTAMP} </if>

<!-- <if test="isRelease!= null"> and is\_release = #{isRelease,jdbcType=INTEGER} </if> -->

<if test="coverPicId!= null">and cover\_pic\_Id = #{coverPicId,jdbcType=INTEGER} </if>

<if test="id!= null"> and id = #{id,jdbcType=INTEGER} </if>

<if test="planCode!= null"> and plan\_code = #{planCode,jdbcType=VARCHAR} </if>

<if test="planName!= null"> and plan\_name = #{planName,jdbcType=VARCHAR} </if>

<if test="userId!= null"> and user\_id = #{userId,jdbcType=INTEGER} </if>

<if test="mediaType!= null"> and media\_type = #{mediaType,jdbcType=INTEGER} </if>

<if test="designSourceType!= null"> and design\_source\_type = #{designSourceType,jdbcType=INTEGER} </if>

<if test="designId!= null"> and design\_id = #{designId,jdbcType=INTEGER} </if>

<if test="designStyleId!= null"> and design\_style\_id = #{designStyleId,jdbcType=INTEGER} </if>

<if test="picId!= null"> and pic\_id = #{picId,jdbcType=INTEGER} </if>

<if test="modelId!= null"> and model\_id = #{modelId,jdbcType=INTEGER} </if>

<if test="model3dId!= null"> and model\_3d\_id = #{model3dId,jdbcType=INTEGER} </if>

<if test="modelU3dId!= null"> and model\_u3d\_id = #{modelU3dId,jdbcType=INTEGER} </if>

<if test="webModelU3dId!= null"> and web\_model\_u3d\_id = #{webModelU3dId,jdbcType=INTEGER} </if>

<if test="configFileId!= null"> and config\_file\_id = #{configFileId,jdbcType=INTEGER} </if>

<if test="spaceCommonId!= null"> and space\_common\_id = #{spaceCommonId,jdbcType=INTEGER} </if>

<if test="planDesc!= null"> and plan\_desc = #{planDesc,jdbcType=VARCHAR} </if>

<if test="draftState!= null"> and draft\_state = #{draftState,jdbcType=INTEGER} </if>

<if test="baiMoState!= null"> and baiMo\_state = #{baiMoState,jdbcType=INTEGER} </if>

<if test="stuffFinishState!= null"> and stuff\_finish\_state = #{stuffFinishState,jdbcType=INTEGER} </if>

<if test="decorateFinishState!= null"> and decorate\_finish\_state = #{decorateFinishState,jdbcType=INTEGER} </if>

<if test="sysCode!= null"> and sys\_code = #{sysCode,jdbcType=VARCHAR} </if>

<if test="creator!= null"> and creator = #{creator,jdbcType=VARCHAR} </if>

<if test="gmtCreate!= null"> and gmt\_create = #{gmtCreate,jdbcType=TIMESTAMP} </if>

<if test="modifier!= null"> and modifier = #{modifier,jdbcType=VARCHAR} </if>

<if test="gmtModified!= null"> and gmt\_modified = #{gmtModified,jdbcType=TIMESTAMP} </if>

<if test="isDeleted!= null"> and is\_deleted = #{isDeleted,jdbcType=INTEGER} </if>

<if test="order != null and order !='' ">

order by ${order}

<if test="orderNum != null and orderNum !='' ">

${orderNum}

</if>

</if>

<if test="orders != null and orders !='' ">

order by ${orders}

</if>

</select>

<!-- 分页汇总 -->

<select id="selectCount" resultType="int"

parameterType="com.nork.design.model.search.DesignPlanSearch">

select count(\*)

from design\_plan

where 1=1

<if test="isRecommended!= null">and is\_recommended = #{isRecommended,jdbcType=INTEGER} </if>

<if test="isDefaultDecorate!= null">and is\_default\_decorate = #{isDecorate,jdbcType=INTEGER} </if>

<if test="planNumber!= null">and plan\_number = #{planNumber,jdbcType=VARCHAR} </if>

<if test="designRecommendedStyleId!= null">and design\_recommended\_style\_id = #{designRecommendedStyleId,jdbcType=INTEGER} </if>

<if test="releaseTime!= null">and release\_time =#{releaseTime,jdbcType=TIMESTAMP} </if>

<!-- <if test="isRelease!= null"> and is\_release = #{isRelease,jdbcType=INTEGER} </if> -->

<if test="coverPicId!= null">and cover\_pic\_Id = #{coverPicId,jdbcType=INTEGER} </if>

<if test="planCode != null and planCode != '' "> and plan\_code = #{planCode,jdbcType=VARCHAR} </if>

<if test="sch\_PlanCode\_ != null and sch\_PlanCode\_ != '' ">

and plan\_code like CONCAT(CONCAT('%',#{sch\_PlanCode\_,jdbcType=VARCHAR}),'%')

</if>

<if test="sch\_PlanCode != null and sch\_PlanCode != '' ">

and plan\_code like CONCAT('%',#{sch\_PlanCode,jdbcType=VARCHAR})

</if>

<if test="schPlanCode\_ != null and schPlanCode\_ != '' ">

and plan\_code like CONCAT(#{schPlanCode\_,jdbcType=VARCHAR},'%')

</if>

<if test="planCodeStart != null and planCodeEnd!= null ">

and plan\_code between #{planCodeStart,jdbcType=VARCHAR} and #{planCodeEnd,jdbcType=VARCHAR}

</if>

<if test="planName != null and planName != '' "> and plan\_name = #{planName,jdbcType=VARCHAR} </if>

<if test="sch\_PlanName\_ != null and sch\_PlanName\_ != '' ">

and plan\_name like CONCAT(CONCAT('%',#{sch\_PlanName\_,jdbcType=VARCHAR}),'%')

</if>

<if test="sch\_PlanName != null and sch\_PlanName != '' ">

and plan\_name like CONCAT('%',#{sch\_PlanName,jdbcType=VARCHAR})

</if>

<if test="schPlanName\_ != null and schPlanName\_ != '' ">

and plan\_name like CONCAT(#{schPlanName\_,jdbcType=VARCHAR},'%')

</if>

<if test="planNameStart != null and planNameEnd!= null ">

and plan\_name between #{planNameStart,jdbcType=VARCHAR} and #{planNameEnd,jdbcType=VARCHAR}

</if>

<if test="modelId!= null"> and model\_id = #{modelId,jdbcType=INTEGER} </if>

<if test="userName != null and userName!='' "> and user\_id in (select u.id FROM sys\_user u where u.user\_name LIKE CONCAT(CONCAT('%',#{userName,jdbcType=VARCHAR}),'%')) </if>

<if test="designSourceType != null "> and design\_source\_type = #{designSourceType,jdbcType=INTEGER} </if>

<if test="designId != null "> and design\_id = #{designId,jdbcType=INTEGER} </if>

<if test="mediaType != null "> and media\_type = #{mediaType,jdbcType=INTEGER} </if>

<if test="designStyleId != null "> and design\_style\_id = #{designStyleId,jdbcType=INTEGER} </if>

<if test="draftState != null"> and draft\_state = #{draftState,jdbcType=INTEGER} </if>

<if test="baiMoState != null"> and baiMo\_state = #{baiMoState,jdbcType=INTEGER} </if>

<if test="stuffFinishState != null"> and stuff\_finish\_state = #{stuffFinishState,jdbcType=INTEGER} </if>

<if test="decorateFinishState != null"> and decorate\_finish\_state = #{decorateFinishState,jdbcType=INTEGER} </if>

<if test="planDesc != null and planDesc != '' "> and plan\_desc = #{planDesc,jdbcType=VARCHAR} </if>

<if test="sch\_PlanDesc\_ != null and sch\_PlanDesc\_ != '' ">

and plan\_desc like CONCAT(CONCAT('%',#{sch\_PlanDesc\_,jdbcType=VARCHAR}),'%')

</if>

<if test="sch\_PlanDesc != null and sch\_PlanDesc != '' ">

and plan\_desc like CONCAT('%',#{sch\_PlanDesc,jdbcType=VARCHAR})

</if>

<if test="schPlanDesc\_ != null and schPlanDesc\_ != '' ">

and plan\_desc like CONCAT(#{schPlanDesc\_,jdbcType=VARCHAR},'%')

</if>

<if test="planDescStart != null and planDescEnd!= null ">

and plan\_desc between #{planDescStart,jdbcType=VARCHAR} and #{planDescEnd,jdbcType=VARCHAR}

</if>

<if test="sysCode != null and sysCode != '' "> and sys\_code = #{sysCode,jdbcType=VARCHAR} </if>

<if test="sch\_SysCode\_ != null and sch\_SysCode\_ != '' ">

and sys\_code like CONCAT(CONCAT('%',#{sch\_SysCode\_,jdbcType=VARCHAR}),'%')

</if>

<if test="sch\_SysCode != null and sch\_SysCode != '' ">

and sys\_code like CONCAT('%',#{sch\_SysCode,jdbcType=VARCHAR})

</if>

<if test="schSysCode\_ != null and schSysCode\_ != '' ">

and sys\_code like CONCAT(#{schSysCode\_,jdbcType=VARCHAR},'%')

</if>

<if test="sysCodeStart != null and sysCodeEnd!= null ">

and sys\_code between #{sysCodeStart,jdbcType=VARCHAR} and #{sysCodeEnd,jdbcType=VARCHAR}

</if>

<if test="creator != null and creator != '' "> and creator = #{creator,jdbcType=VARCHAR} </if>

<if test="sch\_Creator\_ != null and sch\_Creator\_ != '' ">

and creator like CONCAT(CONCAT('%',#{sch\_Creator\_,jdbcType=VARCHAR}),'%')

</if>

<if test="sch\_Creator != null and sch\_Creator != '' ">

and creator like CONCAT('%',#{sch\_Creator,jdbcType=VARCHAR})

</if>

<if test="schCreator\_ != null and schCreator\_ != '' ">

and creator like CONCAT(#{schCreator\_,jdbcType=VARCHAR},'%')

</if>

<if test="creatorStart != null and creatorEnd!= null ">

and creator between #{creatorStart,jdbcType=VARCHAR} and #{creatorEnd,jdbcType=VARCHAR}

</if>

<if test="gmtCreate != null ">

and gmt\_create = #{gmtCreate,jdbcType=TIMESTAMP}

</if>

<if test="gmtCreateStart != null and gmtCreateEnd!= null ">

and gmt\_create between #{gmtCreateStart,jdbcType=TIMESTAMP} and #{gmtCreateEnd,jdbcType=TIMESTAMP}

</if>

<if test="modifier != null and modifier != '' "> and modifier = #{modifier,jdbcType=VARCHAR} </if>

<if test="sch\_Modifier\_ != null and sch\_Modifier\_ != '' ">

and modifier like CONCAT(CONCAT('%',#{sch\_Modifier\_,jdbcType=VARCHAR}),'%')

</if>

<if test="sch\_Modifier != null and sch\_Modifier != '' ">

and modifier like CONCAT('%',#{sch\_Modifier,jdbcType=VARCHAR})

</if>

<if test="schModifier\_ != null and schModifier\_ != '' ">

and modifier like CONCAT(#{schModifier\_,jdbcType=VARCHAR},'%')

</if>

<if test="modifierStart != null and modifierEnd!= null ">

and modifier between #{modifierStart,jdbcType=VARCHAR} and #{modifierEnd,jdbcType=VARCHAR}

</if>

<if test="gmtModified != null ">

and gmt\_modified = #{gmtModified,jdbcType=TIMESTAMP}

</if>

<if test="gmtModifiedStart != null and gmtModifiedEnd!= null ">

and gmt\_modified between #{gmtModifiedStart,jdbcType=TIMESTAMP} and #{gmtModifiedEnd,jdbcType=TIMESTAMP}

</if>

<if test="isDeleted != null "> and is\_deleted = #{isDeleted,jdbcType=INTEGER} </if>

<if test="remark != null and remark != '' "> and remark = #{remark,jdbcType=VARCHAR} </if>

<if test="sch\_Remark\_ != null and sch\_Remark\_ != '' ">

and remark like CONCAT(CONCAT('%',#{sch\_Remark\_,jdbcType=VARCHAR}),'%')

</if> <if test="sch\_Remark != null and sch\_Remark != '' ">

and remark like CONCAT('%',#{sch\_Remark,jdbcType=VARCHAR})

</if> <if test="schRemark\_ != null and schRemark\_ != '' ">

and remark like CONCAT(#{schRemark\_,jdbcType=VARCHAR},'%')

</if> <if test="remarkStart != null and remarkEnd!= null ">

and remark between #{remarkStart,jdbcType=VARCHAR} and #{remarkEnd,jdbcType=VARCHAR}

</if>

</select>

<resultMap id="DesigPlanWithResult" type="com.nork.design.model.DesigPlanWithResult">

<result column="file\_name" property="fileName" jdbcType="VARCHAR"/>

</resultMap>

<!-- 查询所有，可排序 -->

<select id="selectDesignList" resultMap="DesigPlanWithResult"

parameterType="com.nork.design.model.DesignPlan">

SELECT rf.file\_name AS fileName

FROM design\_plan dp ,res\_pic rp,res\_file rf WHERE 1=1 AND dp.`pic\_id`=rp.`id` AND dp.`config\_file\_id` = rf.`id`

<if test="order != null and order !='' ">

order by ${order}

<if test="orderNum != null and orderNum !='' ">

${orderNum}

</if>

</if>

<if test="orders != null and orders !='' ">

order by ${orders}

</if>

<if test="start !=-1 and limit !=-1"> LIMIT #{start}, #{limit}</if>

</select>

<!-- 分页查询 ,可在DesignPlanSearch中自定义查询条件-->

<select id="selectPaginatedList" resultMap="AllResultMap"

parameterType="com.nork.design.model.search.DesignPlanSearch">

select

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

from design\_plan

where 1=1

<if test="isRecommended!= null">and is\_recommended = #{isRecommended,jdbcType=INTEGER} </if>

<if test="isDefaultDecorate!= null">and is\_default\_decorate = #{isDecorate,jdbcType=INTEGER} </if>

<if test="planNumber!= null">and plan\_number = #{planNumber,jdbcType=VARCHAR} </if>

<if test="designRecommendedStyleId!= null">and design\_recommended\_style\_id = #{designRecommendedStyleId,jdbcType=INTEGER} </if>

<if test="releaseTime!= null">and release\_time =#{releaseTime,jdbcType=TIMESTAMP} </if>

<!-- <if test="isRelease!= null"> and is\_release = #{isRelease,jdbcType=INTEGER} </if> -->

<if test="coverPicId!= null">and cover\_pic\_Id = #{coverPicId,jdbcType=INTEGER} </if>

<if test="planCode != null and planCode != '' "> and plan\_code = #{planCode,jdbcType=VARCHAR} </if>

<if test="sch\_PlanCode\_ != null and sch\_PlanCode\_ != '' ">

and plan\_code like CONCAT(CONCAT('%',#{sch\_PlanCode\_,jdbcType=VARCHAR}),'%')

</if>

<if test="sch\_PlanCode != null and sch\_PlanCode != '' ">

and plan\_code like CONCAT('%',#{sch\_PlanCode,jdbcType=VARCHAR})

</if>

<if test="schPlanCode\_ != null and schPlanCode\_ != '' ">

and plan\_code like CONCAT(#{schPlanCode\_,jdbcType=VARCHAR},'%')

</if>

<if test="planCodeStart != null and planCodeEnd!= null ">

and plan\_code between #{planCodeStart,jdbcType=VARCHAR} and #{planCodeEnd,jdbcType=VARCHAR}

</if>

<if test="planName != null and planName != '' "> and plan\_name = #{planName,jdbcType=VARCHAR} </if>

<if test="sch\_PlanName\_ != null and sch\_PlanName\_ != '' ">

and plan\_name like CONCAT(CONCAT('%',#{sch\_PlanName\_,jdbcType=VARCHAR}),'%')

</if>

<if test="sch\_PlanName != null and sch\_PlanName != '' ">

and plan\_name like CONCAT('%',#{sch\_PlanName,jdbcType=VARCHAR})

</if>

<if test="schPlanName\_ != null and schPlanName\_ != '' ">

and plan\_name like CONCAT(#{schPlanName\_,jdbcType=VARCHAR},'%')

</if>

<if test="planNameStart != null and planNameEnd!= null ">

and plan\_name between #{planNameStart,jdbcType=VARCHAR} and #{planNameEnd,jdbcType=VARCHAR}

</if>

<if test="userName != null and userName!='' "> and user\_id in (select u.id FROM sys\_user u where u.user\_name like CONCAT(CONCAT('%',#{userName,jdbcType=VARCHAR}),'%')) </if>

<if test="designSourceType != null "> and design\_source\_type = #{designSourceType,jdbcType=INTEGER} </if>

<if test="designId != null "> and design\_id = #{designId,jdbcType=INTEGER} </if>

<if test="mediaType != null "> and media\_type = #{mediaType,jdbcType=INTEGER} </if>

<if test="designStyleId != null "> and design\_style\_id = #{designStyleId,jdbcType=INTEGER} </if>

<if test="draftState != null"> and draft\_state = #{draftState,jdbcType=INTEGER} </if>

<if test="baiMoState != null"> and baiMo\_state = #{baiMoState,jdbcType=INTEGER} </if>

<if test="stuffFinishState != null"> and stuff\_finish\_state = #{stuffFinishState,jdbcType=INTEGER} </if>

<if test="decorateFinishState != null"> and decorate\_finish\_state = #{decorateFinishState,jdbcType=INTEGER} </if>

<if test="planDesc != null and planDesc != '' "> and plan\_desc = #{planDesc,jdbcType=VARCHAR} </if>

<if test="sch\_PlanDesc\_ != null and sch\_PlanDesc\_ != '' ">

and plan\_desc like CONCAT(CONCAT('%',#{sch\_PlanDesc\_,jdbcType=VARCHAR}),'%')

</if>

<if test="sch\_PlanDesc != null and sch\_PlanDesc != '' ">

and plan\_desc like CONCAT('%',#{sch\_PlanDesc,jdbcType=VARCHAR})

</if>

<if test="schPlanDesc\_ != null and schPlanDesc\_ != '' ">

and plan\_desc like CONCAT(#{schPlanDesc\_,jdbcType=VARCHAR},'%')

</if>

<if test="planDescStart != null and planDescEnd!= null ">

and plan\_desc between #{planDescStart,jdbcType=VARCHAR} and #{planDescEnd,jdbcType=VARCHAR}

</if>

<if test="sysCode != null and sysCode != '' "> and sys\_code = #{sysCode,jdbcType=VARCHAR} </if>

<if test="sch\_SysCode\_ != null and sch\_SysCode\_ != '' ">

and sys\_code like CONCAT(CONCAT('%',#{sch\_SysCode\_,jdbcType=VARCHAR}),'%')

</if>

<if test="sch\_SysCode != null and sch\_SysCode != '' ">

and sys\_code like CONCAT('%',#{sch\_SysCode,jdbcType=VARCHAR})

</if>

<if test="schSysCode\_ != null and schSysCode\_ != '' ">

and sys\_code like CONCAT(#{schSysCode\_,jdbcType=VARCHAR},'%')

</if>

<if test="sysCodeStart != null and sysCodeEnd!= null ">

and sys\_code between #{sysCodeStart,jdbcType=VARCHAR} and #{sysCodeEnd,jdbcType=VARCHAR}

</if>

<if test="creator != null and creator != '' "> and creator = #{creator,jdbcType=VARCHAR} </if>

<if test="sch\_Creator\_ != null and sch\_Creator\_ != '' ">

and creator like CONCAT(CONCAT('%',#{sch\_Creator\_,jdbcType=VARCHAR}),'%')

</if>

<if test="sch\_Creator != null and sch\_Creator != '' ">

and creator like CONCAT('%',#{sch\_Creator,jdbcType=VARCHAR})

</if>

<if test="schCreator\_ != null and schCreator\_ != '' ">

and creator like CONCAT(#{schCreator\_,jdbcType=VARCHAR},'%')

</if>

<if test="creatorStart != null and creatorEnd!= null ">

and creator between #{creatorStart,jdbcType=VARCHAR} and #{creatorEnd,jdbcType=VARCHAR}

</if>

<if test="gmtCreate != null ">

and gmt\_create = #{gmtCreate,jdbcType=TIMESTAMP}

</if>

<if test="gmtCreateStart != null and gmtCreateEnd!= null ">

and gmt\_create between #{gmtCreateStart,jdbcType=TIMESTAMP} and #{gmtCreateEnd,jdbcType=TIMESTAMP}

</if>

<if test="modifier != null and modifier != '' "> and modifier = #{modifier,jdbcType=VARCHAR} </if>

<if test="sch\_Modifier\_ != null and sch\_Modifier\_ != '' ">

and modifier like CONCAT(CONCAT('%',#{sch\_Modifier\_,jdbcType=VARCHAR}),'%')

</if>

<if test="sch\_Modifier != null and sch\_Modifier != '' ">

and modifier like CONCAT('%',#{sch\_Modifier,jdbcType=VARCHAR})

</if>

<if test="schModifier\_ != null and schModifier\_ != '' ">

and modifier like CONCAT(#{schModifier\_,jdbcType=VARCHAR},'%')

</if>

<if test="modifierStart != null and modifierEnd!= null ">

and modifier between #{modifierStart,jdbcType=VARCHAR} and #{modifierEnd,jdbcType=VARCHAR}

</if>

<if test="gmtModified != null ">

and gmt\_modified = #{gmtModified,jdbcType=TIMESTAMP}

</if>

<if test="gmtModifiedStart != null and gmtModifiedEnd!= null ">

and gmt\_modified between #{gmtModifiedStart,jdbcType=TIMESTAMP} and #{gmtModifiedEnd,jdbcType=TIMESTAMP}

</if>

<if test="isDeleted != null "> and is\_deleted = #{isDeleted,jdbcType=INTEGER} </if>

<if test="remark != null and remark != '' "> and remark = #{remark,jdbcType=VARCHAR} </if>

<if test="order != null and order !='' ">

order by ${order} <if test="orderNum != null and orderNum !='' ">${orderNum}</if>

</if>

<if test="orders != null and orders !='' "> order by ${orders}</if>

<if test="order == null and orders == null "> order by id desc</if>

<if test="start !=-1 and limit !=-1"> LIMIT #{start}, #{limit}</if>

</select>

<!-- 其他Map等查询方式 -->

<!-- 设计方案查询 -->

<select id="getDesignList" resultMap="AllResultMap" parameterType="com.nork.design.model.DesignPlan">

SELECT

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

<if test="mostType!=null and mostType!='' and mostType=='mostComment'">

,(SELECT count(c.design\_plan\_id) FROM design\_plan\_comment c where plan.id=c.design\_plan\_id) as comment

</if>

<if test="mostType!=null and mostType!='' and mostType=='mostLike'">

,(SELECT count(l.design\_id) FROM design\_plan\_like l where plan.id=l.design\_id) as comment

</if>

<if test="mostType!=null and mostType!='' and mostType=='mostShare'">

,plan.share\_total as comment

</if>

FROM design\_plan plan

WHERE 1=1 and plan.is\_open = 1

<if test="designSourceType!= null"> AND plan.design\_source\_type = #{designSourceType,jdbcType=INTEGER} </if>

<if test="spaceName!=null and spaceName!='' ">

AND plan.space\_common\_id in

(

SELECT space.id FROM space\_common space

WHERE space.space\_name LIKE CONCAT(CONCAT('%',#{spaceName,jdbcType=VARCHAR}),'%')

)

</if>

<if test="spaceAreas!=null and spaceAreas!='' and spaceFunctionId!=null and spaceFunctionId!=''">

AND plan.space\_common\_id in

(

SELECT space.id FROM space\_common space

WHERE space.space\_function\_id = #{spaceFunctionId,jdbcType=INTEGER} AND space.space\_areas = #{spaceAreas,jdbcType=VARCHAR}

)

</if>

<if test="userId!= null"> AND plan.user\_id = #{userId,jdbcType=INTEGER} </if>

<if test="spaceCommonId!= null"> AND plan.space\_common\_id = #{spaceCommonId,jdbcType=INTEGER} </if>

<if test="designStyleId!= null and designStyleId!= '' and designStyleId!= -1 "> AND plan.design\_style\_id = #{designStyleId,jdbcType=INTEGER} </if>

<if test="mediaType != null">

and plan.media\_type = #{mediaType,jdbcType=INTEGER}

</if>

<if test="isOpen != null">

AND plan.is\_open = #{isOpen,jdbcType=INTEGER}

</if>

<if test="order != null and order !='' ">

order by

<if test="mostType!=null and mostType!='' and mostType=='mostComment'">

comment desc,

</if>

<if test="mostType!=null and mostType!='' and mostType=='mostLike'">

comment desc,

</if>

<if test="mostType!=null and mostType!='' and mostType=='mostShare'">

comment desc,

</if>

${order}

</if>

<if test="start !=-1 and limit !=-1"> LIMIT #{start}, #{limit}</if>

</select>

<!-- 设计方案查询 -->

<select id="getPlanList" resultMap="AllResultMap" parameterType="com.nork.design.model.DesignPlan">

SELECT

plan.\*

<!-- ,dpr.is\_release -->

<if test="mostType!=null and mostType!='' and mostType=='mostComment'">

,(SELECT COUNT(c.design\_plan\_id) FROM design\_plan\_comment c WHERE plan.id=c.design\_plan\_id) as comment

</if>

<if test="mostType!=null and mostType!='' and mostType=='mostLike'">

,(SELECT COUNT(l.design\_id) FROM design\_plan\_like l WHERE plan.id=l.design\_id) as comment

</if>

<if test="mostType!=null and mostType!='' and mostType=='mostShare'">

,plan.share\_total as comment

</if>

FROM design\_plan plan

LEFT JOIN design\_plan\_recommended as dpr on(plan.id = dpr.plan\_id)

JOIN space\_common space ON plan.space\_common\_id = space.id

<if test="multifunctionQuery!=null and multifunctionQuery!='' ">

JOIN design\_plan\_brand dpb on plan.id = dpb.plan\_id

JOIN base\_brand bb on dpb.brand\_id = bb.id

</if>

WHERE 1=1

AND visible=0

<if test="multifunctionQuery!=null and multifunctionQuery!='' ">

AND bb.brand\_name LIKE CONCAT(CONCAT('%',#{brandName,jdbcType=VARCHAR}),'%')

</if>

<if test="planName!=null and planName!='' ">

AND plan.plan\_name LIKE CONCAT(CONCAT('%',#{planName,jdbcType=VARCHAR}),'%')

</if>

<if test="planNumber!= null and planNumber!='' ">

and plan.plan\_number like CONCAT(CONCAT('%',#{planNumber,jdbcType=VARCHAR}),'%')

</if>

<if test="spaceAreas==null and spaceFunctionId!=null and spaceFunctionId!=''">

AND space.space\_function\_id = #{spaceFunctionId,jdbcType=INTEGER}

</if>

<!-- edit by yanghz 解决不能查看复制过来的方案问题-->

<!-- <if test="designSourceType!= null"> AND plan.design\_source\_type = #{designSourceType,jdbcType=INTEGER} </if> -->

AND plan.design\_source\_type in(6,7)

<if test="spaceName!=null and spaceName!='' ">

AND plan.space\_common\_id in

(

SELECT space.id FROM space\_common space

WHERE space.space\_name LIKE CONCAT(CONCAT('%',#{spaceName,jdbcType=VARCHAR}),'%')

)

</if>

<!-- <if test="spaceAreas ==null and spaceFunctionId!=null and spaceFunctionId!=''">

AND plan.space\_common\_id in

(

SELECT space.id FROM space\_common space

WHERE space.space\_function\_id = #{spaceFunctionId,jdbcType=INTEGER}

)

</if> -->

<if test="spaceAreas!=null and spaceAreas!='' and spaceFunctionId!=null and spaceFunctionId!=''">

AND plan.space\_common\_id in

(

SELECT space.id FROM space\_common space

WHERE space.space\_function\_id = #{spaceFunctionId,jdbcType=INTEGER} AND space.space\_areas = #{spaceAreas,jdbcType=VARCHAR}

)

</if>

<if test="userId!= null"> AND plan.user\_id = #{userId,jdbcType=INTEGER} </if>

<if test="spaceCommonId!= null"> AND plan.space\_common\_id = #{spaceCommonId,jdbcType=INTEGER} </if>

<if test="designStyleId!= null and designStyleId!= '' and designStyleId!= -1 "> AND plan.design\_style\_id = #{designStyleId,jdbcType=INTEGER} </if>

<if test="mediaType != null">

AND plan.media\_type = #{mediaType,jdbcType=INTEGER}

</if>

<if test="isOpen != null">

AND plan.is\_open = #{isOpen,jdbcType=INTEGER}

</if>

<if test="order != null and order !='' ">

order by

<if test="mostType!=null and mostType!='' and mostType=='mostComment'">

comment desc,

</if>

<if test="mostType!=null and mostType!='' and mostType=='mostLike'">

comment desc,

</if>

<if test="mostType!=null and mostType!='' and mostType=='mostShare'">

comment desc,

</if>

${order}

</if>

<if test="start !=-1 and limit !=-1"> LIMIT #{start}, #{limit}</if>

</select>

<select id="getPlanListByAreas" resultMap="AllResultMap" parameterType="com.nork.design.model.DesignPlan">

SELECT

plan.id,plan\_code,plan\_name,plan.user\_id,

media\_type,design\_source\_type,

design\_id,design\_template\_id,

design\_style\_id,plan.pic\_id,plan.model\_id,plan.model\_3d\_id,

plan.model\_u3d\_id,plan.web\_model\_u3d\_id,plan.ios\_model\_u3d\_id,

plan.ipad\_model\_u3d\_id,plan.pc\_model\_u3d\_id,plan.android\_model\_u3d\_id,

plan.macBookpc\_model\_u3d\_id,plan.config\_file\_id,plan.space\_common\_id,

plan.plan\_desc,plan.sys\_code,plan.creator,

plan.gmt\_create,plan.modifier,plan.gmt\_modified,plan.share\_total

<if test="mostType!=null and mostType!='' and mostType=='mostComment'">

,(SELECT COUNT(c.design\_plan\_id) from design\_plan\_comment c where plan.id=c.design\_plan\_id) as comment

</if>

FROM design\_plan plan

LEFT JOIN space\_common sc on plan.space\_common\_id = sc.id

WHERE 1=1 and plan.is\_open = 1

<if test="designSourceType!= null"> and design\_source\_type = #{designSourceType,jdbcType=INTEGER} </if>

<if test="spaceName!=null and spaceName!='' ">

AND plan.space\_common\_id in

(

SELECT space.id FROM space\_common space

WHERE space.space\_name like CONCAT(CONCAT('%',#{spaceName,jdbcType=VARCHAR}),'%')

)

</if>

<if test="userId!= null"> and user\_id = #{userId,jdbcType=INTEGER} </if>

<if test="spaceCommonId!= null"> and plan.space\_common\_id = #{spaceCommonId,jdbcType=INTEGER} </if>

<if test="designStyleId!= null and designStyleId!= -1 "> and plan.design\_style\_id = #{designStyleId,jdbcType=INTEGER} </if>

<if test="spaceFunctionId!= null and spaceFunctionId!='' "> and sc.space\_function\_id = #{spaceFunctionId,jdbcType=INTEGER} </if>

<if test="spaceAreas!= null and spaceAreas!='' "> and sc.space\_areas = #{spaceAreas,jdbcType=VARCHAR} </if>

<if test="spaceName!= null and spaceName!='' "> and sc.space\_name like CONCAT(CONCAT('%',#{spaceName,jdbcType=VARCHAR}),'%') </if>

<if test="mediaType != null">

AND plan.media\_type = #{mediaType,jdbcType=INTEGER}

</if>

<if test="order != null and order !='' ">

ORDER BY

<if test="mostType!=null and mostType!='' and mostType=='mostComment'">

comment desc,

</if>

${order}

</if>

<!-- <if test="start !=-1 and limit !=-1"> LIMIT #{start}, #{limit}</if> -->

</select>

<!-- 设计方案汇总 -->

<select id="getPlanCount" resultType="int" parameterType="com.nork.design.model.DesignPlan">

SELECT count(\*)

FROM design\_plan plan

LEFT JOIN design\_plan\_recommended as dpr on(plan.id = dpr.plan\_id)

JOIN space\_common space ON plan.space\_common\_id = space.id

<if test="multifunctionQuery!=null and multifunctionQuery!='' ">

JOIN design\_plan\_brand dpb on plan.id = dpb.plan\_id

JOIN base\_brand bb on dpb.brand\_id = bb.id

</if>

WHERE 1=1

<if test="multifunctionQuery!=null and multifunctionQuery!='' ">

AND bb.brand\_name LIKE CONCAT(CONCAT('%',#{brandName,jdbcType=VARCHAR}),'%')

</if>

<if test="planName!=null and planName!='' ">

AND plan.plan\_name LIKE CONCAT(CONCAT('%',#{planName,jdbcType=VARCHAR}),'%')

</if>

<if test="planNumber!= null and planNumber!='' ">

and plan.plan\_number like CONCAT(CONCAT('%',#{planNumber,jdbcType=VARCHAR}),'%')

</if>

<if test="spaceAreas==null and spaceFunctionId!=null and spaceFunctionId!=''">

AND space.space\_function\_id = #{spaceFunctionId,jdbcType=INTEGER}

<!-- AND plan.space\_common\_id in

(

SELECT space.id FROM space\_common space

WHERE space.space\_function\_id = #{spaceFunctionId,jdbcType=INTEGER}

) -->

</if>

<!-- edit by yanghz 解决不能查看复制过来的方案问题-->

<!-- <if test="designSourceType!= null"> AND plan.design\_source\_type = #{designSourceType,jdbcType=INTEGER} </if> -->

AND plan.design\_source\_type in(6,7)

<if test="spaceName!=null and spaceName!='' ">

AND plan.space\_common\_id in

(

SELECT space.id FROM space\_common space

WHERE space.space\_name like CONCAT(CONCAT('%',#{spaceName,jdbcType=VARCHAR}),'%')

)

</if>

<if test="designStyleId!= null and designStyleId!= -1 "> and plan.design\_style\_id = #{designStyleId,jdbcType=INTEGER} </if>

<if test="spaceAreas!=null and spaceAreas!='' and spaceFunctionId!=null and spaceFunctionId!=''">

AND plan.space\_common\_id in

(

SELECT space.id FROM space\_common space

WHERE space.space\_function\_id = #{spaceFunctionId,jdbcType=INTEGER} AND space.space\_areas = #{spaceAreas,jdbcType=VARCHAR}

)

</if>

<if test="userId!= null"> and plan.user\_id = #{userId,jdbcType=INTEGER} </if>

<if test="mediaType != null">

and plan.media\_type = #{mediaType,jdbcType=INTEGER}

</if>

<if test="isOpen != null">

AND plan.is\_open = #{isOpen,jdbcType=INTEGER}

</if>

</select>

<!-- 用户最新的一个方案 -->

<select id="getUserMaxPlan" resultType="java.util.Map" parameterType="java.util.Map">

SELECT user\_id,space\_common\_id,design\_id,max(gmt\_create)

FROM design\_plan

WHERE user\_id = #{userId,jdbcType=INTEGER}

AND design\_id = #{designId,jdbcType=INTEGER}

GROUP BY space\_common\_id,design\_id,user\_id

<if test="times > 0">

having max(gmt\_create) > now() - interval #{times,jdbcType=INTEGER} second

</if>

ORDER BY max(gmt\_create) desc

</select>

<!-- 设计详情Result -->

<resultMap id="designPlanResultMap" type="com.nork.design.model.DesignPlanResult">

<id column="id" property="designId" jdbcType="INTEGER"/>

<result column="user\_id" property="userId" jdbcType="INTEGER"/>

<result column="space\_function\_id" property="spaceFunctionId" jdbcType="INTEGER"/>

<result column="design\_style\_id" property="designStyleId" jdbcType="INTEGER"/>

<result column="is\_open" property="isOpen" jdbcType="INTEGER"/>

<result column="plan\_code" property="designCode" jdbcType="VARCHAR"/>

<result column="plan\_name" property="designName" jdbcType="VARCHAR"/>

<result column="design\_id" property="designTempletId" jdbcType="VARCHAR"/>

<result column="nick\_name" property="userName" jdbcType="VARCHAR"/>

<result column="space\_code" property="spaceCode" jdbcType="VARCHAR"/>

<result column="space\_name" property="spaceName" jdbcType="VARCHAR"/>

<result column="space\_areas" property="spaceAreas" jdbcType="VARCHAR"/>

<result column="gmt\_create" property="gmtCreate" jdbcType="TIMESTAMP"/>

<result column="gmt\_modified" property="gmtModified" jdbcType="TIMESTAMP"/>

<result column="pic\_path" property="picPath" jdbcType="VARCHAR"/>

<result column="share\_total" property="shareTotal" jdbcType="INTEGER"/>

<!-- <result column="cover\_pic\_Id" property="coverPicId" jdbcType="INTEGER" /> -->

</resultMap>

<!-- 设计详情查询 -->

<select id="designList" resultMap="designPlanResultMap" parameterType="com.nork.design.model.DesignPlanResult">

SELECT dp.id,dp.plan\_code,dp.plan\_name,dp.design\_id,dp.user\_id,dp.gmt\_create,dp.gmt\_modified,

sc.space\_code,sc.space\_name,sc.space\_areas,su.nick\_name,rp.pic\_path,sc.space\_function\_id,dp.design\_style\_id ,dp.is\_open,dp.share\_total

FROM design\_plan dp

LEFT JOIN space\_common sc ON dp.space\_common\_id=sc.id

LEFT JOIN sys\_user su ON dp.user\_id=su.id

LEFT JOIN res\_pic rp ON dp.pic\_id=rp.id

WHERE 1=1

<if test="id != null">

AND dp.id = #{id,jdbcType=INTEGER}

</if>

</select>

<!-- 样板展示 -->

<!-- <select id="modelShowList" resultMap="designPlanResultMap" parameterType="com.nork.design.model.DesignPlanResult">

SELECT dp.id,dp.plan\_code,dp.plan\_name,dp.user\_id,dp.gmt\_create,dp.gmt\_modified,

sc.space\_code,sc.space\_name,sc.space\_areas,su.user\_name,rp.pic\_path

FROM design\_plan dp

LEFT JOIN space\_common sc ON dp.space\_common\_id=sc.id

LEFT JOIN sys\_user su ON dp.user\_id=su.id

LEFT JOIN res\_pic rp ON dp.pic\_id=rp.id

<if test="">

LEFT JOIN design\_plan\_collect dc ON dp.pic\_id=dc.design\_id

</if>

LEFT JOIN design\_plan\_comment dco ON dp.id=dco.design\_plan\_id

WHERE 1=1

<if test="id != null">

AND dp.id = #{id,jdbcType=INTEGER}

</if>

</select> -->

<!--获取样板房产品Id-->

<select id="getTemplateProductId" resultType="int" parameterType="java.util.Map">

SELECT dtp.id FROM design\_templet\_product dtp

JOIN design\_plan dp ON dp.design\_template\_id = dtp.design\_templet\_id

WHERE 1=1

and dp.id = #{planId,jdbcType=INTEGER}

and dtp.product\_id = #{productId,jdbcType=INTEGER}

GROUP BY dtp.product\_id

</select>

<!-- end-->

<!-- 通过plan\_id获得设计方案状态 -->

<select id="getDesignPlanState" resultType="int" parameterType="java.lang.Integer">

<!-- SELECT COUNT(\*) from base\_product as a -->

<!-- LEFT JOIN design\_plan\_product as b on(a.id=b.product\_id) where b.plan\_id =#{designPlanId} and a.product\_code like 'baimo%' -->

SELECT count(\*) from design\_plan\_product as a

LEFT JOIN base\_product as b on(a.product\_id=b.id) where a.plan\_id =#{designPlanId}

and a.is\_deleted=0

and b.product\_code like 'baimo%'

</select>

<!-- Result -->

<resultMap id="designPlanModelMap" type="com.nork.design.model.DesignPlanModel">

<id column="id" property="designPlanId" jdbcType="INTEGER"/>

<result column="plan\_code" property="planCode" jdbcType="VARCHAR"/>

<result column="plan\_name" property="planName" jdbcType="VARCHAR"/>

<result column="plan\_source" property="planSource" jdbcType="VARCHAR"/>

<result column="residential\_units\_name" property="residentialUnitsName" jdbcType="VARCHAR"/>

<result column="model\_id" property="modelId" jdbcType="INTEGER"/>

<result column="space\_function\_id" property="spaceFunctionId" jdbcType="INTEGER"/>

<result column="config\_file\_id" property="configFileId" jdbcType="INTEGER"/>

<result column="space\_common\_id" property="spaceCommonId" jdbcType="INTEGER"/>

<result column="design\_template\_id" property="designTemplateId" jdbcType="INTEGER"/>

<result column="model\_path" property="modelPath" jdbcType="VARCHAR"/>

<result column="file\_path" property="filePath" jdbcType="VARCHAR"/>

<result column="design\_code" property="designTempletCode" jdbcType="VARCHAR"/>

<result column="space\_code" property="spaceCode" jdbcType="VARCHAR"/>

<result column="space\_name" property="spaceName" jdbcType="VARCHAR"/>

<result column="space\_areas" property="spaceAreas" jdbcType="VARCHAR"/>

<result column="value" property="houseTypeValue" jdbcType="INTEGER"/>

<result column="name" property="houseTypeName" jdbcType="VARCHAR"/>

<result column="daylight\_u3d\_model\_id" property="daylightU3dModelId" jdbcType="INTEGER"/>

<result column="dusklight\_u3d\_model\_id" property="dusklightU3dModelId" jdbcType="INTEGER"/>

<result column="nightlight\_u3d\_model\_id" property="nightlightU3dModelId" jdbcType="INTEGER"/>

</resultMap>

<select id="selectDesignPlanInfo" resultMap="designPlanModelMap" parameterType="java.lang.Integer">

SELECT

dp.id,dp.residential\_units\_name,dp.plan\_source,dp.`plan\_code`,dp.`plan\_name`,dp.`model\_id`,dp.`config\_file\_id`,

dp.`space\_common\_id`,dp.`design\_template\_id`,rm.`model\_path`,rf.`file\_path`,

dt.`design\_code`,sc.space\_code,sc.space\_name,sc.space\_areas,sc.daylight\_u3d\_model\_id,sc.dusklight\_u3d\_model\_id,sc.nightlight\_u3d\_model\_id,

sd.`value`,sd.`name`

FROM design\_plan dp

LEFT JOIN res\_model rm ON rm.`id` = dp.`model\_id` AND rm.`is\_deleted` = 0

LEFT JOIN res\_design rf ON rf.`id` = dp.`config\_file\_id` AND rf.`is\_deleted` = 0

LEFT JOIN design\_templet dt ON dt.`id` = dp.`design\_template\_id` AND dt.`is\_deleted` = 0

LEFT JOIN space\_common sc ON sc.`id` = dp.`space\_common\_id` AND sc.`is\_deleted` = 0

LEFT JOIN sys\_dictionary sd ON (sc.`space\_function\_id` = sd.`value` AND sd.`type` = 'houseType' AND sd.`is\_deleted` = 0)

WHERE dp.is\_deleted = 0 and dp.id = #{id,jdbcType=INTEGER}

</select>

<!-- <select id="selectDesignPlanInfo" resultMap="designPlanModelMap" parameterType="java.lang.Integer"> -->

<!-- SELECT -->

<!-- dp.id designPlanId,dp.residential\_units\_name residentialUnitsName,dp.plan\_source planSource,dp.`plan\_code` planCode,dp.`plan\_name` planName,dp.`model\_id` modelId,dp.`config\_file\_id` configFileId, -->

<!-- dp.`space\_common\_id` spaceCommonId,dp.`design\_template\_id` designTemplateId,rm.`model\_path` modelPath,rf.`file\_path` filePath, -->

<!-- dt.`design\_code` designTempletCode,sc.`space\_code` spaceCode,sc.`space\_name` spaceName,sd.`value` houseTypeValue,sd.`name` houseTypeName -->

<!-- FROM design\_plan dp -->

<!-- LEFT JOIN res\_model rm ON rm.`id` = dp.`model\_id` AND rm.`is\_deleted` = 0 -->

<!-- LEFT JOIN res\_file rf ON rf.`id` = dp.`config\_file\_id` AND rf.`is\_deleted` = 0 -->

<!-- LEFT JOIN design\_templet dt ON dt.`id` = dp.`design\_template\_id` AND dt.`is\_deleted` = 0 -->

<!-- LEFT JOIN space\_common sc ON sc.`id` = dp.`space\_common\_id` AND sc.`is\_deleted` = 0 -->

<!-- LEFT JOIN sys\_dictionary sd ON (sc.`space\_function\_id` = sd.`value` AND sd.`type` = 'houseType' AND sd.`is\_deleted` = 0) -->

<!-- WHERE dp.is\_deleted = 0 and dp.id = #{id,jdbcType=INTEGER} -->

<!-- </select> -->

<select id="getProductDTOList" resultType="com.nork.design.model.ProductDTO" parameterType="java.lang.Integer">

SELECT d.product\_id as productId, d.product\_type\_value as productTypeValue, ssd.valuekey as valueKey,d.pos\_index\_path as posIndexPath FROM

(SELECT c.\*,b.product\_type\_value,b.product\_small\_type\_value FROM (select a.\* FROM design\_plan\_product a WHERE a.plan\_id = #{designPlanId} AND a.is\_deleted = 0) c

LEFT JOIN base\_product b ON b.is\_deleted = 0 AND c.init\_product\_id = b.id ) d

LEFT JOIN sys\_dictionary bsd ON bsd.type='productType' AND bsd.value=d.product\_type\_value AND bsd.is\_deleted=0

INNER JOIN sys\_dictionary ssd ON ssd.type=bsd.valuekey AND ssd.value= d.product\_small\_type\_value AND ssd.is\_deleted=0

</select>

<select id="getPlansBySpaceInfo" resultMap="AllResultMap">

SELECT

dp.id,

dp.is\_recommended,

dp.is\_default\_decorate,

dp.plan\_number,

dp.design\_recommended\_style\_id,

dp.release\_time,

<!-- dp.is\_release, -->

dp.cover\_pic\_Id,

dp.plan\_code,

dp.plan\_name,

dp.plan\_source,

dp.house\_id,

dp.living\_id,

dp.residential\_units\_name,

dp.user\_id,

dp.media\_type,

dp.design\_source\_type,

dp.design\_id,

dp.share\_total,

dp.is\_open,

dp.is\_change,

dp.is\_decorated,

dp.draft\_state,

dp.baiMo\_state,

dp.stuff\_finish\_state,

dp.decorate\_finish\_state,

dp.design\_template\_id,

dp.design\_style\_id,

dp.pic\_id,

dp.model\_id,

dp.model\_3d\_id,

dp.model\_u3d\_id,

dp.web\_model\_u3d\_id,

dp.ios\_model\_u3d\_id,

dp.ipad\_model\_u3d\_id,

dp.pc\_model\_u3d\_id,

dp.android\_model\_u3d\_id,

dp.macBookpc\_model\_u3d\_id,

dp.config\_file\_id,

dp.space\_common\_id,

dp.designPlan\_render\_pic\_ids,

dp.plan\_desc,

dp.sys\_code,

dp.creator,

dp.gmt\_create,

dp.modifier,

dp.gmt\_modified,

dp.is\_deleted,

dp.remark,

dp.evening\_file\_id,

dp.dawn\_file\_id,

dp.night\_file\_id,

dp.effects\_config

FROM

design\_plan dp

LEFT JOIN space\_common sc ON sc.id = dp.space\_common\_id

WHERE

1 = 1

AND sc.space\_areas = #{spaceAreas,jdbcType=VARCHAR}

AND sc.space\_function\_id = #{spaceFunctionId,jdbcType=INTEGER}

AND sc.is\_deleted = 0

AND dp.is\_deleted = 0

AND dp.is\_recommended = 1

AND dp.is\_default\_decorate = 1

</select>

<select id="getSceneModifiedById" parameterType="int" resultType="com.nork.design.model.DesignPlan">

SELECT id,scene\_modified sceneModified FROM design\_plan WHERE is\_deleted=0 AND id=#{id}

</select>

<select id="getTempDesignPaln4RenderBakSceneTask" resultMap="AllResultMap">

SELECT

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

FROM design\_plan where is\_deleted=0 AND visible=1 AND design\_scene\_id > 0 AND TIMESTAMPDIFF(MINUTE,gmt\_create,NOW()) > 10

</select>

<delete id="batchDelTempDesign">

delete from design\_plan

where visible=1 AND id in

<foreach collection="delPlanIdList" index="index" item="item" open="(" separator="," close=")">

#{item}

</foreach>

</delete>

<select id="getTempDesignPalnBySceneId" parameterType="com.nork.design.model.DesignPlan" resultType="com.nork.design.model.DesignPlan">

SELECT id FROM design\_plan WHERE design\_scene\_id=#{designSceneId} AND user\_id=#{userId} AND visible=1 AND is\_deleted=0 LIMIT 1

</select>

</mapper>

package com.nork.mobile.controller;

import java.net.URLEncoder;

import java.text.MessageFormat;

import java.util.ArrayList;

import java.util.Date;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import java.util.ResourceBundle;

import java.util.UUID;

import java.util.regex.Matcher;

import java.util.regex.Pattern;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.commons.lang.StringUtils;

import org.apache.log4j.Logger;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.ResponseBody;

import org.springframework.web.bind.annotation.RestController;

import com.nork.common.cache.CacheManager;

import com.nork.common.cache.utils.JedisUtils;

import com.nork.common.constant.SystemCommonConstant;

import com.nork.common.jwt.Jwt;

import com.nork.common.model.LoginUser;

import com.nork.common.model.ResponseEnvelope;

import com.nork.common.util.Utils;

import com.nork.common.util.collections.CustomerListUtils;

import com.nork.design.model.DesignPlan;

import com.nork.mobile.model.search.MobileRenderingModel;

import com.nork.pay.metadata.PayState;

import com.nork.pay.model.PayOrder;

import com.nork.pay.model.search.PayOrderSearch;

import com.nork.pay.service.PayOrderService;

import com.nork.product.model.AuthorizedConfig;

import com.nork.product.service.AuthorizedConfigService;

import com.nork.system.cache.SysUserCacher;

import com.nork.system.controller.web.WebSysUserController;

import com.nork.system.model.SysUser;

import com.nork.system.service.SysUserService;

import com.nork.system.sms.httpclient.SmsClient;

@CrossOrigin(origins = "\*", maxAge = 3600)

@RestController

@Controller

@RequestMapping(value = "/{style}/mobile/user")

public class MobileUserController {

@Autowired

private SysUserService sysUserService;

@Autowired

private PayOrderService payOrderService;

@Autowired

private AuthorizedConfigService authorizedConfigService;

private static Logger logger = Logger.getLogger(WebSysUserController.class);

/\*\*\* 获取配置文件 tmg.properties \*/

private final static ResourceBundle app = ResourceBundle.getBundle("app");

/\*\*\* 获取配置文件 webSocket.properties \*/

// private final static ResourceBundle webSocket = ResourceBundle

// .getBundle("config/webSocket");

private final String SERVERURL = app.getString("app.server.url");

private final String RESOURCESURL = app.getString("app.resources.url");

private final String SITENAME = Utils.getValue("app.server.siteName",

"nork");

// private final String SITEKEY = Utils.getValue("app.server.siteKey",

// "online");

/\*\*

\* 移动端登录接口

\* @param sysUser

\* @return

\*/

// @RequestMapping(value = "/login")

@RequestMapping(value="/login", method = RequestMethod.POST)

@ResponseBody

public ResponseEnvelope login(@RequestBody SysUser sysUser, HttpServletRequest request) {

String account = sysUser.getAccount();

String pwd = sysUser.getPassword();

if(StringUtils.isBlank(account))

return new ResponseEnvelope<SysUser>(false, "账号不能为空!");

if(StringUtils.isBlank(pwd))

return new ResponseEnvelope<SysUser>(false, "密码不能为空!");

// SysUser user = mobileUserService.findByMobileAndPwd(account, pwd);

//判断账号是否存在

SysUser checkUserExist = sysUserService.findWithAccount(account);

if(checkUserExist == null) {

return new ResponseEnvelope<SysUser>(false, "账号未注册，请先在电脑端注册后再登录!");

}

SysUser user = sysUserService.checkUserAccount(account,pwd);

if (user == null) {

return new ResponseEnvelope<SysUser>(false, "账号或密码错误!");

}

String sessionId = request.getRequestedSessionId();

if(StringUtils.isBlank(sessionId)) {

sessionId = UUID.randomUUID().toString().replace("-", "");

}

LoginUser loginUser = this.getLoginUser(user,sessionId);

/\*查询用户关联的序列号list\*/

AuthorizedConfig authorizedConfig = new AuthorizedConfig();

authorizedConfig.setUserId(loginUser.getId());

authorizedConfig.setState(new Integer(1));

List<AuthorizedConfig> list = authorizedConfigService.getList(authorizedConfig);

if(list == null || list.size() == 0) {

return new ResponseEnvelope<SysUser>(false, "尚未绑定序列号或序列号已过期，请在电脑端绑定后再登录！", "");

}

/\*查询用户关联的序列号list->end\*/

//用户是否开通移动端

if(user.getExistsMobile() != SystemCommonConstant.HAS\_EXISTS\_MOBILE) {

return new ResponseEnvelope<SysUser>(false, "您尚未开通移动版功能，请联系客服开通!");

}else {

Date mobileClosedDate = user.getMobileClosedDate();

Date date = new Date();

if(mobileClosedDate.before(date)) {

return new ResponseEnvelope<SysUser>(true, "移动版已到期，请续费开通！");

}

}

loginUser.setServerUrl(SERVERURL);

loginUser.setResourcesUrl(RESOURCESURL);

loginUser.setSiteName(SITENAME);

return new ResponseEnvelope<LoginUser>(true, loginUser);

}

private LoginUser getLoginUser(SysUser user, String sessionId) {

LoginUser loginUser = new LoginUser();

// String token = UUID.randomUUID().toString().replace("-", "");

String appKey = UUID.randomUUID().toString().replace("-", "");

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

Date date=new Date();

payload.put("uid", user.getId());//用户ID

payload.put("appKey", appKey);//用户ID

payload.put("utype", user.getUserType());

payload.put("iat", date.getTime());//生成时间

payload.put("uname", user.getNickName());

payload.put("ext",date.getTime()+1000\*60\*60);//过期时间1小时

String token=Jwt.createToken(payload);

loginUser.setAppKey(appKey);

loginUser.setToken(token);

loginUser.setId(user.getId());

loginUser.setUserType(user.getUserType());

loginUser.setServerUrl(SERVERURL);

loginUser.setResourcesUrl(RESOURCESURL);

loginUser.setSiteName(SITENAME);

loginUser.setBalanceAmount(user.getBalanceAmount());

loginUser.setConsumAmount(user.getConsumAmount());

String loginFlag =SystemCommonConstant.LOGIN\_FROM\_MOBILE; //移动端登陆标识

String key = loginFlag + appKey;

String onlineUserKey = loginFlag + user.getId().toString(); //当前登录用户信息key

int expireInSeconds = 60 \* 60 \* 24 ;

if(Utils.enableRedisCache()) {

LoginUser onlineUser = new LoginUser();

onlineUser = (LoginUser) CacheManager.getInstance().getCacher().getObject(onlineUserKey);

if(onlineUser != null && onlineUser.getAppKey() != null) {

//移除之前登录用户的缓存

CacheManager.getInstance().getCacher().remove(onlineUserKey);

CacheManager.getInstance().getCacher().remove(key);

}

}

CacheManager.getInstance().getCacher().setObject(onlineUserKey, loginUser, expireInSeconds);

CacheManager.getInstance().getCacher().setObject(key, loginUser, expireInSeconds);

return loginUser;

}

/\*\*

\* 获取 验证码 的接口

\*/

@RequestMapping(value = "/app/sms")

@ResponseBody

public Object sms(@RequestBody SysUser sysUser2,

HttpServletRequest request, HttpServletResponse response)

throws Exception {

String msgId = sysUser2.getMsgId();

String mobile = sysUser2.getMobile();

String type = "";

if (StringUtils.isEmpty(msgId)) {

return new ResponseEnvelope<SysUser>(false, "参数msgId不能为空！！", msgId);

}

if (StringUtils.isEmpty(mobile)) {

return new ResponseEnvelope<SysUser>(false, "手机号码不能为空！", msgId);

}

if (!isMobile(mobile)) {

return new ResponseEnvelope<SysUser>(false, "请输入正确的手机号！", msgId);

}

//判断账号是否存在

SysUser checkUserExist = sysUserService.findWithAccount(mobile);

if(checkUserExist == null) {

return new ResponseEnvelope<SysUser>(false, "账号未注册，请先在电脑端注册后再登录!");

}

String cacheEnable = Utils.getValue("redisCacheEnable", "0");

String result = "0";

/\*\* 短信验证码 \*/

String code = Utils.generateRandomDigitString(6);

if ("1".equals(cacheEnable)) {

CacheManager.getInstance().getCacher().remove("REGISTER\_CODE\_" + mobile);

CacheManager.getInstance().getCacher().set("REGISTER\_CODE\_" + mobile, code, 0);

CacheManager.getInstance().getCacher().remove("REGISTER\_CODE\_TIMEOUT\_" + mobile);

CacheManager.getInstance().getCacher().set("REGISTER\_CODE\_TIMEOUT\_" + mobile,System.currentTimeMillis()+"",0);

CacheManager.getInstance().getCacher().remove("REGISTER\_CODE\_Count" + mobile);

CacheManager.getInstance().getCacher().set("REGISTER\_CODE\_Count" + mobile, "0", 0);

} else {

request.getServletContext().setAttribute("REGISTER\_CODE\_" + mobile, code);

request.getServletContext().setAttribute("REGISTER\_CODE\_TIMEOUT\_" + mobile,System.currentTimeMillis());

request.getServletContext().setAttribute("REGISTER\_CODE\_Count" + mobile, "0");

}

// //System.out.println("验证码------------------："+code);

/\*\* 发送短信的方法的参数准备 \*/

// 短信内容+随机生成的6位短信验证码

String message = "";

if ("register".equals(type)) {

message=MessageFormat.format(

SmsClient.app.getString("registerContext"), code,

SmsClient.VALID\_TIME / 60000, SmsClient.SERVICE\_PHONE);

}else{

message=MessageFormat.format(

SmsClient.app.getString("updateMobileContext"), code,

SmsClient.VALID\_TIME / 60000, SmsClient.SERVICE\_PHONE);

}

message = URLEncoder.encode(message, "UTF-8");

// 短信接口参数

// mobile = "13906513994,18358100817,15816860922";

long seqId = System.currentTimeMillis();

String params = "phone=" + mobile + "&message=" + message

+ "&addserial=&seqid=" + seqId;

result = SmsClient.sendSMS(SmsClient.SEND\_MESSAGE, params);

if ("0".equals(result)) {

return new ResponseEnvelope<String>(true, "发送成功！", msgId,true);

} else {

return new ResponseEnvelope<String>(false, "发送验证短信异常！", msgId);

}

}

/\*\*

\* 手机号验证

\* @param str

\* @return 验证通过返回true

\*/

public static boolean isMobile(String str) {

Pattern p = null;

Matcher m = null;

boolean b = false;

p = Pattern

.compile("^(0|86|17951)?(13[0-9]|15[012356789]|17[0135678]|18[0-9]|14[579])[0-9]{8}$"); // 验证手机号

m = p.matcher(str);

b = m.matches();

return b;

}

/\*

\* 找回密码接口

\*/

@RequestMapping(value = "/app/findPassword")

@ResponseBody

public Object findPassword(/\*String mobile, String newPassword, String code,String msgId,\*/

@RequestBody SysUser sysUser2, HttpServletRequest request,

HttpServletResponse response) throws Exception {

String msgId = sysUser2.getMsgId();

String newPassword = sysUser2.getPassword();

String code = sysUser2.getRemark();

String mobile = sysUser2.getAccount();

if (StringUtils.isEmpty(msgId)) {

return new ResponseEnvelope<DesignPlan>(false, "参数msgId不能为空", msgId);

}

if (StringUtils.isEmpty(newPassword)) {

return new ResponseEnvelope<SysUser>(false, "新密码不能为空", msgId);

}

if (StringUtils.isEmpty(mobile)) {

return new ResponseEnvelope<SysUser>(false, "手机号不能为空", msgId);

}

if (!isMobile(mobile)) {

return new ResponseEnvelope<SysUser>(false, "请输入正确的手机号！", msgId);

}

if (StringUtils.isEmpty(code)) {

return new ResponseEnvelope<SysUser>(false, "请输入验证码！", msgId);

}

String cacheEnable = Utils.getValue("redisCacheEnable", "0");

/\*\* 获取session中存放的手机短信验证码 \*/

long currentTime = System.currentTimeMillis();

long sendCodeTime = 0L;

String yzm = "";

int num = 0;

//允许输入验证码次数

String verify=SmsClient.app.getString("verifyCount");

int verifyCount=3;

if(verify!=null){

verifyCount=Integer.parseInt(verify);

}

if ("1".equals(cacheEnable)) {

Object codeObject = CacheManager.getInstance().getCacher()

.get("REGISTER\_CODE\_" + mobile);

Object timeoutObject = CacheManager.getInstance().getCacher()

.get("REGISTER\_CODE\_TIMEOUT\_" + mobile);

if (codeObject != null) {

yzm = codeObject.toString();

}

if (timeoutObject != null) {

sendCodeTime = Long.parseLong(timeoutObject.toString());

}

Object count = CacheManager.getInstance().getCacher()

.get("REGISTER\_CODE\_Count" + mobile);

logger.debug("缓存验证码次数获取："+count);

if (count != null) {

num = Integer.parseInt(count.toString());

}

} else {

sendCodeTime = (long) request.getServletContext().getAttribute(

"REGISTER\_CODE\_TIMEOUT\_" + mobile);

yzm = (String) request.getServletContext().getAttribute(

"REGISTER\_CODE\_" + mobile);

String count = (String) request.getServletContext().getAttribute(

"REGISTER\_CODE\_Count" + mobile);

logger.debug("非缓存验证码次数获取："+count);

if (count != null) {

num = Integer.parseInt(count);

}

}

logger.debug("session数据：REGISTER\_CODE\_TIMEOUT=" + sendCodeTime

+ " REGISTER\_CODE=" + yzm +" 验证码次数：" + num);

if (StringUtils.isBlank(yzm) || sendCodeTime <= 0) {

return new ResponseEnvelope<SysUser>(false, "验证码已失效，请重新获取！", msgId);

}

// 一分钟验证码失效

logger.debug("------忘记密码：当前验证码：" + yzm + " 时间(currentTime："

+ currentTime + ";sendCodeTime：" + sendCodeTime + ")："

+ (currentTime - sendCodeTime));

if ((currentTime - sendCodeTime) > SmsClient.VALID\_TIME) {

return new ResponseEnvelope<SysUser>(false, "验证码已超时，请重新获取！", msgId);

}

if (num < verifyCount) {

if (!code.equals(yzm)) {

num++;

if ("1".equals(cacheEnable)) {

CacheManager.getInstance().getCacher()

.set("REGISTER\_CODE\_Count" + mobile, num + "", 0);

} else {

request.getServletContext().setAttribute(

"REGISTER\_CODE\_Count" + mobile, num + "");

}

return new ResponseEnvelope<SysUser>(false, "验证码不对，请重新输入！",

msgId);

}

} else {

//success设置为true便于app关闭计时

return new ResponseEnvelope<SysUser>(false, "验证码已失效，请重新获取！", msgId);

}

SysUser sysUser = new SysUser();

sysUser.setMobile(mobile);

List<SysUser> list = sysUserService.getList(sysUser);

/\*\* List<SysUser> list = SysUserCacher.getAllList(sysUser); \*/

if (CustomerListUtils.isEmpty(list)) {

return new ResponseEnvelope<SysUser>(false, "找不到该用户", msgId);

} else {

SysUser user = list.get(0);

user.setPassword(newPassword);

/\*\* sysUserService.update(sysUser); \*/

sysUserService.update(user);

List<SysUser> list1 = null;

if (StringUtils.equals("1", cacheEnable)) {

list1 = SysUserCacher.getSysList();

} else {

list1 = sysUserService.getSysList();

}

request.getSession().setAttribute("sysUserList", list1);

if ("1".equals(cacheEnable)) {

CacheManager.getInstance().getCacher().remove("REGISTER\_CODE\_Count" + mobile);

CacheManager.getInstance().getCacher().set("REGISTER\_CODE\_Count" + mobile, "4", 0);

} else {

request.getServletContext().setAttribute("REGISTER\_CODE\_Count" + mobile, "4");

}

return new ResponseEnvelope<SysUser>(true, msgId, true);

}

}

/\*\*

\* 个人消费记录列表

\* @return

\*/

@RequestMapping(value = "/findExpenseRecordList")

@ResponseBody

public Object findExpenseRecordList(@RequestBody MobileRenderingModel model) {

PayOrderSearch payOrderSearch = new PayOrderSearch();

Integer userId = model.getUserId();

if (StringUtils.isEmpty(model.getMsgId())) {

return new ResponseEnvelope<PayOrder>(false, "参数msgId为空!", "");

}

if (userId == 0 || userId.intValue() == 0) {

return new ResponseEnvelope<PayOrder>(false, "请登录系统!", model.getMsgId());

}

//只有支付成功的订单消费记录，排除序列号的订单，sql处理

payOrderSearch.setUserId(userId);

payOrderSearch.setStart(model.getStart());

payOrderSearch.setLimit(model.getLimit());

payOrderSearch.setPayState(PayState.SUCCESS);

List<PayOrder> list = new ArrayList<PayOrder>();

int total = 0;

total = payOrderService.getCount(payOrderSearch);

if (total > 0) {

list = payOrderService.getPaginatedList(payOrderSearch);

}

SysUser user = sysUserService.get(userId);

Double balanceAmount = user.getBalanceAmount();

return new ResponseEnvelope<PayOrder>(true, "Success",model.getMsgId(),user, total, list);

// return new ResponseEnvelope<PayOrder>(total, list, model.getMsgId());

}

}

package com.nork.system.service;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import javax.xml.rpc.ServiceException;

import com.nork.common.model.LoginUser;

import com.nork.common.model.ResponseEnvelope;

import com.nork.system.model.SysUser;

import com.nork.system.model.search.SysUserSearch;

/\*\*

\* @Title: SysUserService.java

\* @Package com.nork.system.service

\* @Description:系统-用户Service

\* @createAuthor pandajun

\* @CreateDate 2015-05-19 12:30:46

\* @version V1.0

\*/

public interface SysUserService {

/\*\*

\* 新增数据

\*

\* @param sysUser

\* @return int

\*/

public int add(SysUser sysUser);

public int add(SysUser sysUser,HttpServletRequest request);

/\*\*

\* 更新数据

\*

\* @param sysUser

\* @return int

\*/

public int update(SysUser sysUser);

public boolean updateFinance(SysUser user);

/\*\*

\* 删除数据

\*

\* @param id

\* @return int

\*/

public int delete(Integer id);

public SysUser findWithAccount(String account);

/\*\*

\* 获取数据详情

\*

\* @param id

\* @return SysUser

\*/

public SysUser get(Integer id);

/\*\*\*

\* 获取系统用户

\* @return

\*/

List<SysUser> getSysList();

/\*\*

\* 所有数据

\*

\* @param sysUser

\* @return List<SysUser>

\*/

public List<SysUser> getList(SysUser sysUser);

/\*\*

\* 获取数据数量

\*

\* @param sysUser

\* @return int

\*/

public int getCount(SysUserSearch sysUserSearch);

/\*\*

\* 分页获取数据

\*

\* @param sysUser

\* @return List<SysUser>

\*/

public List<SysUser> getPaginatedList(SysUserSearch sysUsertSearch);

/\*\*

\* 验证用户名和密码

\* @param account

\* @param password

\* @return

\*/

public ResponseEnvelope<SysUser> checkUserAccount(SysUser user);

/\*\*

\* 用户登录验证

\* @param account

\* @param password

\* @return

\*/

public SysUser checkUserAccount(String account, String password);

/\*\*

\* 用户登录只验证账号

\* @param account

\* @param password

\* @return

\*/

public SysUser checkOnlyUserAccount(String account, String password);

/\*\*

\* 修改用户密码

\* @param user

\* @param oldPassword

\* @param checkOldPassword

\* @return

\*/

public boolean updateUserPwd(SysUser user, String oldPassword,

boolean checkOldPassword);

/\*\*

\* 分页查询用户详细信息

\* @param sysUserSearch

\* @return

\*/

List<SysUser> userDetailPaginatedList(SysUserSearch sysUserSearch);

public void processregister(String email);

public void processActivate(String email , String validateCode)throws ServiceException;

/\*\*通过loginName查找用户\*/

public List<SysUser> findOneByLoginName(String loginName);

/\*得到有gmt\_create字段的所有表名\*/

public List<String> getTableNames();

/\*\*

\* 密码md5加密规则

\* @author huangsongbo

\* @param password 明文密码

\* @return 加密后的字符串

\*/

public String getMd5Password(String password);

/\*\*

\* 判断需不需要给用户解绑设备号->满足条件则解绑

\* @author huangsongbo

\* @param userId

\*/

public void unwrapImei(Integer userId);

/\*\*

\* @author huangsongbo

\* 检测用户是否已登录

\* @param id

\* @return true:已登录,不允许再登录;false:允许登录

\*/

//public boolean checkisLoginIn(Integer id);

/\*\*

\* 更新该用户的在缓存中的信息(登录有效时长信息:validTime)

\* @author huangsongbo

\* @param userId

\*/

//public void updateValidTime(Integer userId);

/\*\*

\* 检测用户是否已登录(检测缓存信息是否存在)

\* @author huangsongbo

\* @param id

\* @return

\*/

public boolean checkisLoginIn2(Integer id, String terminalImei);

/\*\*

\* LoginUser转化为SysUser

\* @author huangsongbo

\* @param loginUser

\* @return

\*/

public SysUser getSysUserByLoginUser(LoginUser loginUser);

/\*\*

\* 通过角色CODE查询用户

\* @param roleCode

\* @return

\*/

public List<SysUser> getUserByRoleCode(String roleCode);

/\*\*

\* 通过登录四参数验证该用户是否异常

\* @author huangsongbo

\* @param appKey

\* @param token

\* @param deviceId

\* @param mediaType

\* @return

\*/

public LoginUser isAllowedThrough(String appKey, String token, String deviceId, String mediaType);

/\*\*

\* 获取session登录信息

\* @author xiaoxc

\* @param request

\* @return LoginUser

\*/

public LoginUser getSessionLoginUser(HttpServletRequest request);

/\*\*

\* 用token来加密app端需要的key

\* @param user

\* @return

\*/

SysUser EencryptKey(SysUser user);

public void updateBalanceAmountByUserId(SysUser user);

/\*\*

\* 获取用户头像地址

\* @param sysUser

\* @return

\*/

public SysUser getUserPic(SysUser sysUser);

/\*\*

\* 用户登录验证

\* @param account

\* @param password

\* @return

\*/

public boolean checkWhiteList(String terminalImei);

/\*\*

\* 移动端续费

\* @param sysUser

\* @return

\*/

void renew(SysUser sysUser);

/\*\*

\* 添加基本角色组权限

\* @return

\*/

int addRoleGroup(Integer userId);

/\*\*

\* 根据手机号码查询信息

\* \*/

List<SysUser> selectByMobile(String mobile);

/\*\*

\* 根据手机号码去修改用户信息

\* \*/

int updateByMobileInfo(SysUser sysUser);

}

package com.nork.system.service.impl;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

import java.util.List;

import javax.servlet.http.HttpServletRequest;

import javax.xml.rpc.ServiceException;

import com.nork.common.properties.AesProperties;

import com.nork.common.util.AESUtil2;

import com.nork.system.model.SysUserRole;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import com.alibaba.druid.util.StringUtils;

import com.nork.common.errorkeys.ErrorKeys;

import com.nork.common.model.LoginUser;

import com.nork.common.model.ResponseEnvelope;

import com.nork.common.model.WhiteDeviceList;

import com.nork.common.properties.AesProperties;

import com.nork.common.util.AESUtil2;

import com.nork.common.util.SendEmail;

import com.nork.common.util.Utils;

import com.nork.common.util.Validator;

import com.nork.common.util.collections.CustomerListUtils;

import com.nork.mobile.dao.MobileRenderRecordMapper;

import com.nork.product.model.AuthorizedConfig;

import com.nork.product.model.search.AuthorizedConfigSearch;

import com.nork.product.service.AuthorizedConfigService;

import com.nork.system.cache.SysUserCacher;

import com.nork.system.dao.SysDictionaryMapper;

import com.nork.system.dao.SysUserMapper;

import com.nork.system.model.ResPic;

import com.nork.system.model.SysDictionary;

import com.nork.system.model.SysUser;

import com.nork.system.model.search.SysUserSearch;

import com.nork.system.service.ResPicService;

import com.nork.system.service.SysDictionaryService;

import com.nork.system.service.SysUserService;

/\*\*

\* @Title: SysUserServiceImpl.java

\* @Package com.nork.system.service.impl

\* @Description:系统-用户ServiceImpl

\* @createAuthor pandajun

\* @CreateDate 2015-05-19 12:30:46

\* @version V1.0

\*/

@Service("sysUserService")

@Transactional

public class SysUserServiceImpl implements SysUserService {

private SysUserMapper sysUserMapper;

@Autowired

private SysDictionaryMapper sysDictionaryMapper;

@Autowired

private SysDictionaryService sysDictionaryService;

@Autowired

private AuthorizedConfigService authorizedConfigService;

@Autowired

private ResPicService resPicServicer;

@Autowired

private MobileRenderRecordMapper mobileRenderRecordMapper;

@Autowired

public void setSysUserMapper(SysUserMapper sysUserMapper) {

this.sysUserMapper = sysUserMapper;

}

private static Logger logger = LoggerFactory.getLogger(SysUserServiceImpl.class);

// 密匙

private static String key=Utils.getValueByFileKey(AesProperties.AES, AesProperties.AES\_RESOURCES\_ENCRYPT\_KEY\_FILEKEY, "41e5c74dd46e4ddcb942dc8ce6224a2e").trim();

/\*\*

\* 新增数据

\*

\* @param sysUser

\* @return int

\*/

@Override

public int add(SysUser sysUser) {

sysUserMapper.insertSelective(sysUser);

return sysUser.getId();

}

/\*\*

\* 新增数据

\*

\* @param sysUser

\* @return int

\*/

@Override

public int add(SysUser sysUser, HttpServletRequest request) {

setAppKey(sysUser, request);

sysUserMapper.insertSelective(sysUser);

return sysUser.getId();

}

/\*\* 设置sysUser的mediaType属性 \*/

private void setAppKey(SysUser sysUser, HttpServletRequest request) {

Integer mediaType = sysUser.getMediaType();

if (mediaType != null) {

/\*

\* SysDictionarySearch sysDictionarySearch=new

\* SysDictionarySearch(); sysDictionarySearch.setType("mediaType");

\* sysDictionarySearch.setValue(mediaType); List<SysDictionary>

\* sysDictionaries=sysDictionaryMapper.selectPaginatedList(

\* sysDictionarySearch);

\* if(sysDictionaries!=null&&sysDictionaries.size()>0){

\* sysUser.setAppKey(sysDictionaries.get(0).getAtt1());

\* sysUser.setMediaType(mediaType); }

\*/

SysDictionary sysDictionary = sysDictionaryService.getSysDictionaryByValue("mediaType", mediaType);

if (sysDictionary != null) {

sysUser.setAppKey(sysDictionary.getAtt1());

sysUser.setMediaType(mediaType);

}

}

}

public List<SysUser> getSysList(){

return sysUserMapper.getSysList();

}

/\*\*

\* 更新数据

\*

\* @param sysUser

\* @return int

\*/

@Override

public int update(SysUser sysUser) {

sysUserMapper.updateByPrimaryKeySelective(sysUser);

// setAppKey(sysUser);

return sysUser.getId();

}

/\*\*

\* 删除数据

\*

\* @param id

\* @return int

\*/

@Override

public int delete(Integer id) {

return sysUserMapper.deleteByPrimaryKey(id);

}

/\*\*

\* 获取数据详情

\*

\* @param id

\* @return SysUser

\*/

@Override

public SysUser get(Integer id) {

return sysUserMapper.selectByPrimaryKey(id);

}

/\*\*

\* 所有数据

\*

\* @param sysUser

\* @return List<SysUser>

\*/

@Override

public List<SysUser> getList(SysUser sysUser) {

sysUser.setIsDeleted(0);

return sysUserMapper.selectList(sysUser);

}

/\*\*

\* 获取数据数量

\*

\* @param sysUser

\* @return int

\*/

@Override

public int getCount(SysUserSearch sysUserSearch) {

sysUserSearch.setIsDeleted(0);

return sysUserMapper.selectCount(sysUserSearch);

}

/\*\*

\* 分页获取数据

\*

\* @param sysUser

\* @return List<SysUser>

\*/

@Override

public List<SysUser> getPaginatedList(SysUserSearch sysUserSearch) {

sysUserSearch.setIsDeleted(0);

return sysUserMapper.selectPaginatedList(sysUserSearch);

}

public SysUser findWithAccount(String account){

SysUser user=null;

SysUser search=new SysUser();

search.setAccount(account);

List<SysUser> lst=sysUserMapper.selectWithAccount(search);

if (CustomerListUtils.isNotEmpty(lst)){

user=lst.get(0);

}

return user;

}

/\*\*

\* 查询用户

\*

\*/

public ResponseEnvelope<SysUser> checkUserAccount(SysUser user) {

String account = user.getAccount();

String password = user.getPassword();

if (StringUtils.isEmpty(account) || StringUtils.isEmpty(password)) {

return new ResponseEnvelope<SysUser>(false, "未传递正确的参数!", user.getMsgId());

}

user.setIsDeleted(0);

List<SysUser> list = null;

// 手机登录验证

if (Validator.MOBILE.validate(account)) {

user.setMobile(account);

list = sysUserMapper.selectList(user);

}

// 邮箱登录验证

else if (Validator.EAMIL.validate(account)) {

user.setEmail(account);

list = sysUserMapper.selectList(user);

}

// 用户名登录验证

else {

user.setNickName(account);

list = sysUserMapper.selectList(user);

}

if (CustomerListUtils.isNotEmpty(list)) {

user.setPassword(password);

list = sysUserMapper.selectList(user);

if (CustomerListUtils.isNotEmpty(list)) {

return new ResponseEnvelope<SysUser>(user, user.getMsgId(), true);

} else {

// 密码不正确

return new ResponseEnvelope<SysUser>(false, ErrorKeys.INVALID\_PASSWORD, user.getMsgId());

}

} else {

// 用户不存在

return new ResponseEnvelope<SysUser>(false, ErrorKeys.INVALID\_LOGIN\_ACCOUNT, user.getMsgId());

}

}

/\*\*

\* 用户登录验证

\*

\*/

public SysUser checkUserAccount(String account, String password) {

SysUser user = new SysUser();

user.setIsDeleted(0);

user.setPassword(password);

List<SysUser> list = null;

// 手机登录验证

if (Validator.MOBILE.validate(account)) {

user.setMobile(account);

list = sysUserMapper.selectList(user);

}

// 邮箱登录验证

else if (Validator.EAMIL.validate(account)) {

user.setEmail(account);

list = sysUserMapper.selectList(user);

}

// 用户名登录验证

else {

if (StringUtils.isEmpty(account) || StringUtils.isEmpty(password))

return null;

user.setNickName(account);

list = sysUserMapper.selectList(user);

//modify by qiu.jun 增加对用户名登录的支持

if (CustomerListUtils.isEmpty(list)) {

user.setUserName(account);

list = sysUserMapper.selectList(user);

}

}

if (CustomerListUtils.isNotEmpty(list)) {

user = list.get(0);

return user;

} else {

return null;

}

}

/\*\*

\* 用户登录验证

\*

\*/

public SysUser checkOnlyUserAccount(String account, String password) {

SysUser user = new SysUser();

user.setIsDeleted(0);

user.setPassword(password);

List<SysUser> list = null;

// 手机登录验证

if (Validator.MOBILE.validate(account)) {

user.setMobile(account);

list = sysUserMapper.selectList(user);

}

// 邮箱登录验证

else if (Validator.EAMIL.validate(account)) {

user.setEmail(account);

list = sysUserMapper.selectList(user);

}

// 用户名登录验证

else {

if (StringUtils.isEmpty(account) || StringUtils.isEmpty(password))

return null;

user.setNickName(account);

list = sysUserMapper.selectList(user);

//modify by qiu.jun 增加对用户名登录的支持

if (CustomerListUtils.isEmpty(list)) {

user.setUserName(account);

list = sysUserMapper.selectList(user);

}

}

if (CustomerListUtils.isNotEmpty(list)) {

user = list.get(0);

return user;

} else {

return null;

}

}

public boolean updateFinance(SysUser user){

int result=sysUserMapper.updateFinanceById(user);

return result>0 ? true : false;

}

/\*\*

\* 修改用户密码

\*

\* @param user

\* @param oldPassword

\* @param checkOldPassword

\* @return

\*/

public boolean updateUserPwd(SysUser user, String oldPassword, boolean checkOldPassword) {

int count = sysUserMapper.updateByPrimaryKeySelective(user);

return count > 0 ? true : false;

}

/\*\*

\* 分页查询用户详细信息

\*

\* @param sysUserSearch

\* @return

\*/

@Override

public List<SysUser> userDetailPaginatedList(SysUserSearch sysUserSearch) {

sysUserSearch.setIsDeleted(0); //过滤被逻辑删除用户

return sysUserMapper.userDetailPaginatedList(sysUserSearch);

}

public void processregister(String email) {

SysUser user = new SysUser();

/// 邮件的内容

StringBuffer sb = new StringBuffer("点击下面链接激活账号，48小时生效，否则重新注册账号，链接只能使用一次，请尽快激活！</br>");

sb.append("<a href=\"http://localhost:8080/springmvc/user/register?action=activate&email=");

sb.append(email);

sb.append("&validateCode=");

sb.append(user.getAtt3());

sb.append("\">http://localhost:8080/springmvc/user/register?action=activate&email=");

sb.append(email);

sb.append("&validateCode=");

sb.append(user.getAtt3());

sb.append("</a>");

// 发送邮件

SendEmail.send(email, sb.toString());

//////System.out.println("发送邮件");

}

/\*\*

\* 密码md5加密规则

\* @author huangsongbo

\* @param password 明文密码

\* @return 加密后的字符串

\*/

public String getMd5Password(String password){

return md5(md5("WeB"+password));

//

}

/\*\*

\* md5加密(copy)

\* @author huangsongbo

\* @param plainText

\* @return

\*/

private String md5(String plainText) {

String str="";

try {

MessageDigest md = MessageDigest.getInstance("MD5");

md.update(plainText.getBytes());

byte b[] = md.digest();

int i;

StringBuffer buf = new StringBuffer("");

for (int offset = 0; offset < b.length; offset++) {

i = b[offset];

if (i < 0)

i += 256;

if (i < 16)

buf.append("0");

buf.append(Integer.toHexString(i));

}

////////System.out.println("result: " + buf.toString());// 32位的加密

////////System.out.println("result: " + buf.toString().substring(8, 24));// 16位的加密

str=buf.toString();

} catch (NoSuchAlgorithmException e) {

e.printStackTrace();

}

return str;

}

/\*\*

\* 判断需不需要给用户解绑设备号->满足条件则解绑

\* @author huangsongbo

\* @param userId

\*/

public void unwrapImei(Integer userId) {

AuthorizedConfigSearch authorizedConfigSearch2=new AuthorizedConfigSearch();

authorizedConfigSearch2.setStart(0);

authorizedConfigSearch2.setLimit(1);

authorizedConfigSearch2.setUserId(userId);

authorizedConfigSearch2.setState(1);

List<AuthorizedConfig> authorizedConfigs2=authorizedConfigService.getPaginatedList(authorizedConfigSearch2);

if(authorizedConfigs2==null||authorizedConfigs2.size()==0){

/\*解绑user和设备号的关联\*/

SysUser sysUser=get(userId);

sysUser.setUserImei("");

update(sysUser);

}

}

/\*\*

\* 处理激活

\*

\* @throws ParseException

\*/

/// 传递激活码和email过来

public void processActivate(String email, String validateCode) throws ServiceException {

// 数据访问层，通过email获取用户信息

SysUser user = null;

// 验证用户是否存在

// 激活成功， //并更新用户的激活状态，为已激活

user.setEmail(email);

sysUserMapper.updateByPrimaryKeySelective(user);

}

@Override

public List<SysUser> findOneByLoginName(String loginName) {

return sysUserMapper.findOneByLoginName(loginName);

}

@Override

public List<String> getTableNames() {

// TODO Auto-generated method stub

return sysUserMapper.getTableNames();

}

/\*\*

\* @author huangsongbo

\* 检测用户是否已登录

\* @param id

\* @return true:已登录,不允许再登录;false:允许登录

\*/

// public boolean checkisLoginIn(Integer id) {

// SysUser sysUser=SysUserCacher.checkTimeOutUser(id);

// if(sysUser!=null){

// /\*缓存信息存在->检测信息中的有效时长\*/

// if(sysUser.getValidTime()==null){

// return false;

// }else{

// long now=new Date().getTime();

// if(now>sysUser.getValidTime())

// /\*用户已退出登录(没有通过心跳接口更新登录有效时长)\*/

// return false;

// /\*该用户还在登录状态中\*/

// return true;

// }

// }else{

// return false;

// }

// }

/\*\*

\* 更新该用户的在缓存中的信息(登录有效时长信息:validTime)

\* @author huangsongbo

\* @param userId

\*/

// public void updateValidTime(Integer userId) {

// SysUser sysUser=SysUserCacher.checkTimeOutUser(userId);

// if(sysUser==null){

// logger.error("------更新用户登录有效时长接口中,缓存找不到对应用户信息:这种情况不允许出现");

// return;

// }

// SysUserCacher.updateTimeOutUser(sysUser);

// }

/\*\*

\* 检测用户是否已登录(检测缓存信息是否存在)

\* @author huangsongbo

\* @param id

\* @return

\*/

public boolean checkisLoginIn2(Integer id, String terminalImei) {

boolean flag=false;

SysUser sysUser=get(id);

logger.info("======id : " + id + "; terminalImei : " + terminalImei + "; sysUser.getToken() : " + sysUser.getToken());

SysUser sysUserFromCache=SysUserCacher.checkTimeOutUserByToken(sysUser.getToken());

logger.info("======id : " + id + "; sysUserFromCache : " + sysUserFromCache);

if(sysUserFromCache==null||sysUserFromCache.getId().equals(new Integer(0))){

/\*缓存信息不存在,帐号没有被登录\*/

flag=false;

}else{

logger.info("======id : " + id + "; sysUserFromCache.getTerminalImei() : " + sysUserFromCache.getTerminalImei());

if(StringUtils.equals(terminalImei, sysUserFromCache.getTerminalImei())){

flag=false;

}else{

flag=true;

}

}

return flag;

}

/\*\*

\* LoginUser转化为SysUser

\* @author huangsongbo

\* @param loginUser

\* @return

\*/

public SysUser getSysUserByLoginUser(LoginUser loginUser) {

SysUser sysUser = null;

if(loginUser!=null){

sysUser=new SysUser();

sysUser.setId(loginUser.getId());

sysUser.setUserType(loginUser.getUserType());

sysUser.setSysCode(loginUser.getLoginId());

sysUser.setNickName(loginUser.getLoginName());

sysUser.setMobile(loginUser.getLoginPhone());

sysUser.setEmail(loginUser.getLoginEmail());

sysUser.setUserName(loginUser.getName());

sysUser.setAppKey(loginUser.getAppKey());

sysUser.setToken(loginUser.getToken());

sysUser.setMediaType(org.apache.commons.lang3.StringUtils.isBlank(loginUser.getDeviceId())?null:Integer.valueOf(loginUser.getDeviceId()));

sysUser.setMsgId(loginUser.getMsgId());

sysUser.setGroupId(loginUser.getGroupId());

sysUser.setSex(loginUser.getSex());

sysUser.setPicPath(loginUser.getPicPath());

sysUser.setBrandIds(loginUser.getBrandIds());

}

return sysUser;

}

/\*\*

\* 通过角色CODE查询用户

\* @param roleCode

\* @return

\*/

@Override

public List<SysUser> getUserByRoleCode(String roleCode){

if( org.apache.commons.lang3.StringUtils.isNotBlank(roleCode) ){

return sysUserMapper.getUserByRoleCode(roleCode);

}

return null;

}

/\*\*

\* 通过登录四参数验证该用户是否异常

\* @author huangsongbo

\* @param appKey

\* @param token

\* @param deviceId

\* @param mediaType

\* @return

\*/

public LoginUser isAllowedThrough(String appKey, String token, String deviceId, String mediaType) {

// 参数验证 ->start

if(

org.apache.commons.lang3.StringUtils.isBlank(appKey)

|| org.apache.commons.lang3.StringUtils.isBlank(token)

|| org.apache.commons.lang3.StringUtils.isBlank(deviceId)

){

return null;

}

// 参数验证 ->end

LoginUser loginUser = new LoginUser();

SysUser sysUser = new SysUser();

sysUser.setIsDeleted(0);

sysUser.setToken(token);

sysUser.setAppKey(appKey);

sysUser.setMediaType(new Integer(deviceId));

List<SysUser> list = this.getList(sysUser);

if (list != null && list.size() == 1) {

SysUser user = list.get(0);

if(user != null){

loginUser.setId(user.getId());

loginUser.setLoginId(user.getSysCode());

loginUser.setLoginName(user.getNickName());

loginUser.setLoginEmail(user.getEmail());

loginUser.setName(user.getUserName());

loginUser.setUserType(user.getUserType());

loginUser.setGroupId(user.getGroupId());

loginUser.setSex(user.getSex());

}else{

logger.info("user is null.....");

return null;

}

}

return loginUser;

}

/\*\*

\* 获取session登录信息

\* @author xiaoxc

\* @param request

\* @return LoginUser

\*/

@Override

public LoginUser getSessionLoginUser(HttpServletRequest request){

LoginUser loginUser = new LoginUser();

if (com.nork.common.constant.util.SystemCommonUtil.getCurrentLoginUserInfo(request) == null || com.nork.common.constant.util.SystemCommonUtil.getCurrentLoginUserInfo(request) == null) {

loginUser.setId(-1);

loginUser.setLoginName("nologin");

} else {

loginUser = com.nork.common.constant.util.SystemCommonUtil.getCurrentLoginUserInfo(request);

}

return loginUser;

}

/\*\*

\* 用token对加密文件的key进行加密

\*/

@Override

public SysUser EencryptKey(SysUser user) {

String token = "";

if(org.apache.commons.lang3.StringUtils.isNotBlank(user.getToken())){

//截取token的前八位

if(user.getToken().length() < 8){

//字符长度不够8位则在后面补0

token = user.getToken()+String.format("%1$0"+(8-user.getToken().length())+"d",0);

}else{

token = user.getToken().substring(0,8);

}

String encrypt = AESUtil2.encryptFile(key, token);

user.setCryptKey(encrypt);

return user;

}else{

logger.error("用户id="+user.getId()+"返回token信息为null！");

}

return user;

}

@Override

public void updateBalanceAmountByUserId(SysUser user) {

sysUserMapper.updateBalanceAmountByUserId(user);

}

/\*\*

\* 获得用户头像

\*/

@Override

public SysUser getUserPic(SysUser sysUser) {

if(sysUser ==null || sysUser.getId()==null) {

return sysUser;

}

if (StringUtils.isEmpty(sysUser.getPicPath())) {

Integer sexValue=sysUser.getSex();

if(sexValue == null ) {

sexValue = 1;

}

//查找默认头像地址

SysDictionary dictionary=new SysDictionary();

dictionary=sysDictionaryService.getSysDictionary("userDefaultPicture", sexValue);

if(dictionary !=null && dictionary.getPicId()!=null) {

ResPic userPic=resPicServicer.get(dictionary.getPicId());

if(userPic != null && userPic.getPicPath() !=null) {

sysUser.setUserPic(userPic.getPicPath());

}

}

}

return sysUser;

}

@Override

public boolean checkWhiteList(String terminalImei) {

// TODO Auto-generated method stub

WhiteDeviceList whiteDeviceList = new WhiteDeviceList();

List<WhiteDeviceList> list = mobileRenderRecordMapper.checkWhiteList(whiteDeviceList);

for (WhiteDeviceList whiteDeviceList2 : list) {

if (terminalImei.equals(whiteDeviceList2.getDeviceKey())) {

return true;

}else {

continue;

}

}

return false;

}

/\*\*

\* 移动端续费

\* @param sysUser

\* @return

\*/

@Override

public void renew(SysUser sysUser) {

sysUserMapper.renew(sysUser);

}

/\*\*

\* 添加基本角色组权限

\* @return

\*/

@Override

public int addRoleGroup(Integer userId){

int roleGroupId = sysUserMapper.getRoleGroupId();

SysUserRole sysUserRole = new SysUserRole();

sysUserRole.setUserId(userId);

sysUserRole.setRoleGroupId(roleGroupId);

sysUserRole.setIsDeleted(0);

return sysUserMapper.addRoleGroup(sysUserRole);

}

/\*\*

\* 根据手机号码查询用户信息

\* \*/

@Override

public List<SysUser> selectByMobile(String mobile) {

// TODO Auto-generated method stub

return sysUserMapper.selectByMobile(mobile);

}

/\*\*

\* 根据手机号码去修改用户信息

\* \*/

@Override

public int updateByMobileInfo(SysUser sysUser) {

// TODO Auto-generated method stub

return sysUserMapper.updateByMobileInfo(sysUser);

}

}

package com.nork.system.dao;

import java.util.List;

import com.nork.system.model.SysUserRole;

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import com.nork.system.model.SysUser;

import com.nork.system.model.search.SysUserSearch;

/\*\*

\* @Title: SysUserMapper.java

\* @Package com.nork.system.dao

\* @Description:系统-用户Mapper

\* @createAuthor pandajun

\* @CreateDate 2015-05-19 12:30:46

\* @version V1.0

\*/

@Repository

@Transactional

public interface SysUserMapper {

int insertSelective(SysUser record);

int updateByPrimaryKeySelective(SysUser record);

int updateByMobileInfo(SysUser sysUser);

int updateFinanceById(SysUser user);

int deleteByPrimaryKey(Integer id);

/\*\*\*

\* 获取系统用户

\* @param sysUser

\* @return

\*/

List<SysUser> getSysList();

SysUser selectByPrimaryKey(Integer id);

/\*\*

\* 根据手机号码查询信息

\* \*/

List<SysUser> selectByMobile(String mobile);

int selectCount(SysUserSearch sysUserSearch);

List<SysUser> selectPaginatedList(SysUserSearch sysUserSearch);

List<SysUser> selectList(SysUser sysUser);

List<SysUser> selectWithAccount(SysUser sysUser);

/\*\*

\* 分页查询用户详细信息

\*

\* @param sysUserSearch

\* @return

\*/

List<SysUser> userDetailPaginatedList(SysUserSearch sysUserSearch);

List<SysUser> findOneByLoginName(String loginName);

List<String> getTableNames();

/\*\*

\* 通过角色CODE查询用户

\* @param roleCode

\* @return

\*/

public List<SysUser> getUserByRoleCode(String roleCode);

List<Integer> getAllId();

public void updateBalanceAmountByUserId(SysUser sysUser);

/\*\*

\* 移动端续费

\* @param sysUser

\* @return

\*/

void renew(SysUser sysUser);

/\*\*

\* 获取基本角色组id

\* @return

\*/

int getRoleGroupId();

/\*\*

\* 添加角色组

\* @param sysUserRole

\* @return

\*/

int addRoleGroup(SysUserRole sysUserRole);

}

<?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.nork.system.dao.SysUserEquipmentMapper">

<!-- \*\*插入定义\*\* -->

<insert id="insertSelective" parameterType="com.nork.system.model.SysUserEquipment" useGeneratedKeys="true" keyProperty="id">

insert into sys\_user\_equipment

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

<if test="userId!= null">user\_id, </if>

<if test="userImei!= null">user\_imei, </if>

<if test="equipmentType!= null">equipment\_type, </if>

<if test="gmtCreate!= null">gmt\_create, </if>

<if test="gmtModified!= null">gmt\_modified, </if>

<if test="isDeleted!= null">is\_deleted, </if>

<if test="usbTerminalImei!= null">usb\_user\_imei, </if>

</trim>

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

<if test="userId!= null">#{userId,jdbcType=INTEGER}, </if>

<if test="userImei!= null">#{userImei,jdbcType=VARCHAR}, </if>

<if test="equipmentType!= null">#{equipmentType,jdbcType=VARCHAR}, </if>

<if test="gmtCreate!= null"> #{gmtCreate,jdbcType=TIMESTAMP}, </if>

<if test="gmtModified!= null"> #{gmtModified,jdbcType=TIMESTAMP}, </if>

<if test="isDeleted!= null"> #{isDeleted,jdbcType=INTEGER}, </if>

<if test="usbTerminalImei!= null"> #{usbTerminalImei,jdbcType=VARCHAR}, </if>

</trim>

</insert>

<!-- \*\*更新定义\*\* -->

<update id="updateByPrimaryKeySelective" parameterType="com.nork.system.model.SysUserEquipment">

update sys\_user\_equipment

<set>

<if test="userId!= null"> user\_id = #{userId,jdbcType=INTEGER}, </if>

<if test="userImei!= null"> user\_imei = #{userImei,jdbcType=VARCHAR}, </if>

<if test="equipmentType!= null"> equipment\_type = #{equipmentType,jdbcType=VARCHAR}, </if>

<if test="gmtCreate!= null"> gmt\_create = #{gmtCreate,jdbcType=TIMESTAMP}, </if>

<if test="gmtModified!= null"> gmt\_modified = #{gmtModified,jdbcType=TIMESTAMP}, </if>

<if test="isDeleted!= null"> is\_deleted = #{isDeleted,jdbcType=INTEGER}, </if>

</set>

where id = #{id,jdbcType=INTEGER}

</update>

<!-- \*\*删除定义\*\* -->

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

delete from sys\_user\_equipment

where id = #{id,jdbcType=INTEGER}

</delete>

<!-- \*\*常量定义\*\* -->

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

id,user\_id,user\_imei,equipment\_type,gmt\_create,gmt\_modified,is\_deleted

</sql>

<!-- \*\*结果定义\*\* -->

<resultMap id="AllResultMap" type="com.nork.system.model.SysUserEquipment">

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

<result column="user\_id" property="userId" jdbcType="INTEGER" />

<result column="user\_imei" property="userImei" jdbcType="VARCHAR" />

<result column="equipment\_type" property="equipmentType" jdbcType="VARCHAR" />

<result column="gmt\_create" property="gmtCreate" jdbcType="TIMESTAMP" />

<result column="gmt\_modified" property="gmtModified" jdbcType="TIMESTAMP" />

<result column="is\_deleted" property="isDeleted" jdbcType="INTEGER" />

<result column="usb\_user\_imei" property="usbTerminalImei" jdbcType="VARCHAR" />

</resultMap>

<!-- \*\*查询定义\*\* -->

<!-- 按主键查询 -->

<select id="selectByPrimaryKey" resultMap="AllResultMap"

parameterType="java.lang.Integer">

select

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

from sys\_user\_equipment

where id = #{id,jdbcType=INTEGER}

</select>

<!-- 查询所有，可排序 -->

<select id="selectList" resultMap="AllResultMap"

parameterType="com.nork.system.model.SysUserEquipment">

select

id,user\_id,user\_imei,equipment\_type,gmt\_create,gmt\_modified,is\_deleted,usb\_user\_imei

from sys\_user\_equipment

where 1=1

<if test="userId!= null">and user\_id = #{userId,jdbcType=INTEGER} </if>

<if test="userImei!= null"> and user\_imei = #{userImei,jdbcType=VARCHAR} </if>

<if test="equipmentType!= null">and equipment\_type = #{equipmentType,jdbcType=VARCHAR} </if>

<if test="gmtCreate!= null"> and gmt\_create = #{gmtCreate,jdbcType=TIMESTAMP} </if>

<if test="gmtModified!= null"> and gmt\_modified = #{gmtModified,jdbcType=TIMESTAMP} </if>

<if test="isDeleted!= null"> and is\_deleted = #{isDeleted,jdbcType=INTEGER} </if>

<if test="order != null and order !='' ">

order by ${order}

<if test="orderNum != null and orderNum !='' ">

${orderNum}

</if>

</if>

<if test="orders != null and orders !='' ">

order by ${orders}

</if>

</select>

<!-- 分页汇总 -->

<select id="selectCount" resultType="int"

parameterType="com.nork.system.model.SysUserEquipment">

select count(\*)

from sys\_user\_equipment

where 1=1

<if test="userId!= null">and user\_id = #{userId,jdbcType=INTEGER} </if>

<if test="userImei!= null"> and user\_imei = #{userImei,jdbcType=VARCHAR} </if>

<if test="equipmentType!= null">and equipment\_type = #{equipmentType,jdbcType=VARCHAR} </if>

<if test="gmtCreate!= null"> and gmt\_create = #{gmtCreate,jdbcType=TIMESTAMP} </if>

<if test="gmtModified!= null"> and gmt\_modified = #{gmtModified,jdbcType=TIMESTAMP} </if>

<if test="isDeleted!= null"> and is\_deleted = #{isDeleted,jdbcType=INTEGER} </if>

</select>

<!-- 通过用户id,和设备类型 删除设备号 -->

<update id="deleteByUserId" >

update sys\_user\_equipment

set is\_deleted = 1 , gmt\_modified = NOW()

where user\_id = #{userId,jdbcType=INTEGER}

AND equipment\_type = #{equipmentType,jdbcType=VARCHAR}

</update>

</mapper>

package com.nork.system.model;

import java.io.Serializable;

import java.math.BigDecimal;

import java.util.ArrayList;

import java.util.Date;

import java.util.HashMap;

import java.util.Map;

import java.util.List;

import com.nork.common.model.LoginMenu;

import com.nork.common.model.LoginUser;

import com.nork.common.model.Mapper;

/\*\*

\* @Title: SysUser.java

\* @Package com.nork.system.model

\* @Description:系统-用户

\* @createAuthor pandajun

\* @CreateDate 2015-05-19 12:30:46

\* @version V1.0

\*/

public class SysUser extends Mapper implements Serializable{

private static final long serialVersionUID = 1L;

private Integer id;

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

private String webSocketServer;

public String getWebSocketServer() {

return webSocketServer;

}

public void setWebSocketServer(String webSocketServer) {

this.webSocketServer = webSocketServer;

}

/\*\* 系统编码 \*\*/

private String sysCode;

/\*\* 创建者 \*\*/

private String creator;

/\*\* 创建时间 \*\*/

private Date gmtCreate;

/\*\* 修改人 \*\*/

private String modifier;

/\*\* 修改时间 \*\*/

private Date gmtModified;

/\*\* 是否删除 \*\*/

private Integer isDeleted;

/\*\* 用户名 \*\*/

private String userName;

/\*\* 昵称 \*\*/

private String nickName;

/\*\* 密码 \*\*/

private String password;

/\*\* 邮箱 \*\*/

private String email;

/\*\* 移动 \*\*/

private String mobile;

/\*\* 用户类型 \*\*/

private Integer userType;

/\*\* 性别 \*\*/

private Integer sex;

/\*\* 职业 \*\*/

private String job;

/\*\* 区域 \*\*/

private Integer areaId;

/\*\* 区域 长编码（省市） \*\*/

private String areaLongCode;

/\*\* 头像id \*\*/

private Integer picId;

/\*\* 媒介类型 \*\*/

private Integer mediaType;

/\*\* 字符备用1 \*\*/

private String appKey;

/\*\* 字符备用2 \*\*/

private String token;

/\*\* 字符备用3 \*\*/

private String att3;

/\*\* 字符备用4 \*\*/

private String att4;

/\*\* 字符备用5 \*\*/

private String att5;

/\*\* 字符备用6 \*\*/

private String att6;

/\*\* 时间备用1 \*\*/

private Date dateAtt1;

/\*\* 时间备用2 \*\*/

private Date dateAtt2;

/\*\* 组织id \*\*/

private Integer groupId;

/\*\* 整数备用2 \*\*/

private Integer numAtt2;

/\*\* 数字备用1 \*\*/

private Double numAtt3;

/\*\* 数字备用2 \*\*/

private Double numAtt4;

/\*\* 备注 \*\*/

private String remark;

/\*\* 真实姓名 \*\*/

private String realName;

/\*\* QQ \*\*/

private String qq;

/\*\* 公司电话 \*\*/

private String companyTel;

/\*\* 公司地址 \*\*/

private String companyAddress;

/\*\* 个人专长 \*\*/

private String specialityValue;

/\*\* 个人简介 \*\*/

private String intro;

/\*\* 申请时间 \*\*/

private Date applicationDate;

/\*\* 邮箱验证状态 \*\*/

private Integer emailVerifyState;

/\*\* 手机验证状态 \*\*/

private Integer mobileVerifyState;

/\*\* 身份证验证状态 \*\*/

private Integer idcardVerifyState;

/\*\* 等级 \*\*/

private Integer level;

/\*\* 验证人员 \*\*/

private String verifyUser;

/\*\* 验证时间 \*\*/

private Date verifyDate;

/\*\* 用户角色id集合 \*\*/

private String roleIds;

//\*\*\*\*\*\*登录账号

private String account;

//\*\*\*\*\*\*用户类型

private String userTypeName;

//头像路径

private String picPath;

private String userPic;

/\*心跳间隔\*/

private Integer heartbeatTime;

/\*登录有效时间\*/

private Long validTime;

/\*识别该用户是否在其他设备登录

\* 0:未在其他设备登录

\* 1:已在其他设备登录\*/

private Integer isLoginIn;

/\*用户登录的设备号\*/

private String terminalImei;

/\*消费总金额\*/

private Double consumAmount;

/\*账户余额\*/

private Double balanceAmount;

//登录设备限制1:允许所有类型网卡登录，2:所有网卡均不允许登录，3:仅允许PCI网卡登录，4:仅允许USB网卡登录

private Integer deviceRestrict;

/\*\*详细地址\*/

String address;

String payCallBackServer;

String webSocketMessage;

// String webSocketRender;

String serverUrl;

/\*该用户是否为 方案推荐审核管理员 1 为是 \*/

private Integer designPlanCheckAdministrator;

//服务化地址集合

private List<Map<String,String>> servitizationList;

private List<Integer> userId;

/\*\*

\* 文件加密key，app端用来解密

\* add by yanghz

\*/

String cryptKey;

/\*用户角色集合对象\*/

List<SysRole> roleList = new ArrayList<SysRole>();

/\*\*

\* app菜单列表

\*\*/

List<LoginMenu> appMenuList = new ArrayList<LoginMenu>();

/\*\* 是否开通移动端 (0:未开通，1：已开通) \*\*/

private Integer existsMobile;

/\*\* 移动端开通时间 \*\*/

private Date mobileOpenedDate;

/\*\* 移动端截止时间 \*\*/

private Date mobileClosedDate;

/\*\* 移动端续费年限 \*\*/

private Integer renewNumber;

/\*\*

\* 使用类型（0：试用、1：正式）

\*/

private Integer useType;

/\*\*

\* 是否展示三度一键方案：0:否,1:是

\*/

private Integer showSanduPlan;

public Integer getUseType() {

return useType;

}

public void setUseType(Integer useType) {

this.useType = useType;

}

public Integer getShowSanduPlan() {

return showSanduPlan;

}

public void setShowSanduPlan(Integer showSanduPlan) {

this.showSanduPlan = showSanduPlan;

}

public Date getMobileOpenedDate() {

return mobileOpenedDate;

}

public void setMobileOpenedDate(Date mobileOpenedDate) {

this.mobileOpenedDate = mobileOpenedDate;

}

public Date getMobileClosedDate() {

return mobileClosedDate;

}

public void setMobileClosedDate(Date mobileClosedDate) {

this.mobileClosedDate = mobileClosedDate;

}

public Integer getRenewNumber() {

return renewNumber;

}

public void setRenewNumber(Integer renewNumber) {

this.renewNumber = renewNumber;

}

public List<Integer> getUserId() {

return userId;

}

public void setUserId(List<Integer> userId) {

this.userId = userId;

}

/\*\*

\* 分布式资源存储,域名配置

\*/

private List<ResourcesUrl> resourcesUrls;

public class ResourcesUrl{

private List<String> modelName;

private String domain;

public List<String> getModelName() {

return modelName;

}

public void setModelName(List<String> modelName) {

this.modelName = modelName;

}

public String getDomain() {

return domain;

}

public void setDomain(String domain) {

this.domain = domain;

}

}

public List<ResourcesUrl> getResourcesUrls() {

return resourcesUrls;

}

public void setResourcesUrls(List<ResourcesUrl> resourcesUrls) {

this.resourcesUrls = resourcesUrls;

}

public List<Map<String, String>> getServitizationList() {

return servitizationList;

}

public void setServitizationList(List<Map<String, String>> servitizationList) {

this.servitizationList = servitizationList;

}

public List<LoginMenu> getAppMenuList() {

return appMenuList;

}

public void setAppMenuList(List<LoginMenu> appMenuList) {

this.appMenuList = appMenuList;

}

public List<SysRole> getRoleList() {

return roleList;

}

public void setRoleList(List<SysRole> roleList) {

this.roleList = roleList;

}

public String getCryptKey() {

return cryptKey;

}

public void setCryptKey(String cryptKey) {

this.cryptKey = cryptKey;

}

public Integer getDesignPlanCheckAdministrator() {

return designPlanCheckAdministrator;

}

public void setDesignPlanCheckAdministrator(Integer designPlanCheckAdministrator) {

this.designPlanCheckAdministrator = designPlanCheckAdministrator;

}

/\*\*用作压力测试\*/

private String designPlanId;

public String getDesignPlanId() {

return designPlanId;

}

public void setDesignPlanId(String designPlanId) {

this.designPlanId = designPlanId;

}

public String getWebSocketMessage() {

return webSocketMessage;

}

public void setWebSocketMessage(String webSocketMessage) {

this.webSocketMessage = webSocketMessage;

}

// public String getWebSocketRender() {

// return webSocketRender;

// }

//

// public void setWebSocketRender(String webSocketRender) {

// this.webSocketRender = webSocketRender;

// }

public String getPayCallBackServer() {

return payCallBackServer;

}

public void setPayCallBackServer(String payCallBackServer) {

this.payCallBackServer = payCallBackServer;

}

public String getServerUrl() {

return serverUrl;

}

public void setServerUrl(String serverUrl) {

this.serverUrl = serverUrl;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public String getTerminalImei() {

return terminalImei;

}

public String getRoleIds() {

return roleIds;

}

public void setRoleIds(String roleIds) {

this.roleIds = roleIds;

}

public void setTerminalImei(String terminalImei) {

this.terminalImei = terminalImei;

}

public Integer getIsLoginIn() {

return isLoginIn;

}

public void setIsLoginIn(Integer isLoginIn) {

this.isLoginIn = isLoginIn;

}

public Long getValidTime() {

return validTime;

}

public void setValidTime(Long validTime) {

this.validTime = validTime;

}

public Integer getHeartbeatTime() {

return heartbeatTime;

}

public void setHeartbeatTime(Integer heartbeatTime) {

this.heartbeatTime = heartbeatTime;

}

public String getUserPic() {

return userPic;

}

public void setUserPic(String userPic) {

this.userPic = userPic;

}

//个人专长

private String specialityName;

//驳回原因

private String reasonRejected;

//个人资料旧手机

private String oldMobile;

//是否已关注

private Integer attentionState;

//绑定品牌id

private String brandIds;

//省份

private String provinceCode;

//市

private String cityCode;

/\*\*

\* 组织ID

\*/

private Integer orgId;

/\*\*

\* 组织名称

\*/

private String orgName;

/\*\*所在地\*\*/

private String areaName;

/\*\*所在地\*\*/

private String jobName;

private String serviceUrl;

private String resourcesUrl;

private String siteName;

private String sitekey;

//粉丝数量

private Integer fansCount;

//关注数量

private Integer attentionCount;

//作品数量

private Integer worksCount;

//访问数量

private Integer accessCount;

//审核1人员

private String auditor1;

//审核2人员

private String auditor2;

//审核3人员

private String auditor3;

/\*用户绑定设备号\*/

private String userImei;

private Map<String,String> resourceMap;

/\*\*

\* 用户绑定的公司ids

\*/

private String companyIds;

//是否需要更新密码(0:不需要更新,1:需要更新)

private Integer passwordUpdateFlag;

public Integer getPasswordUpdateFlag() {

return passwordUpdateFlag;

}

public void setPasswordUpdateFlag(Integer passwordUpdateFlag) {

this.passwordUpdateFlag = passwordUpdateFlag;

}

/\*\*

\* 用户公司/经销商id

\*/

private Integer businessAdministrationId;

public Integer getBusinessAdministrationId() {

return businessAdministrationId;

}

public void setBusinessAdministrationId(Integer businessAdministrationId) {

this.businessAdministrationId = businessAdministrationId;

}

public String getCompanyIds() {

return companyIds;

}

public void setCompanyIds(String companyIds) {

this.companyIds = companyIds;

}

public Map<String, String> getResourceMap() {

return resourceMap;

}

public void setResourceMap(Map<String, String> resourceMap) {

this.resourceMap = resourceMap;

}

public String getUserImei() {

return userImei;

}

public void setUserImei(String userImei) {

this.userImei = userImei;

}

public String getBrandIds() {

return brandIds;

}

public void setBrandIds(String brandIds) {

this.brandIds = brandIds;

}

public String getOldMobile() {

return oldMobile;

}

public void setOldMobile(String oldMobile) {

this.oldMobile = oldMobile;

}

public String getReasonRejected() {

return reasonRejected;

}

public void setReasonRejected(String reasonRejected) {

this.reasonRejected = reasonRejected;

}

public String getSpecialityName() {

return specialityName;

}

public void setSpecialityName(String specialityName) {

this.specialityName = specialityName;

}

public String getAuditor1() {

return auditor1;

}

public void setAuditor1(String auditor1) {

this.auditor1 = auditor1;

}

public String getAuditor2() {

return auditor2;

}

public void setAuditor2(String auditor2) {

this.auditor2 = auditor2;

}

public String getAuditor3() {

return auditor3;

}

public void setAuditor3(String auditor3) {

this.auditor3 = auditor3;

}

public Integer getAccessCount() {

return accessCount;

}

public void setAccessCount(Integer accessCount) {

this.accessCount = accessCount;

}

public Integer getFansCount() {

return fansCount;

}

public void setFansCount(Integer fansCount) {

this.fansCount = fansCount;

}

public Integer getAttentionCount() {

return attentionCount;

}

public void setAttentionCount(Integer attentionCount) {

this.attentionCount = attentionCount;

}

public Integer getWorksCount() {

return worksCount;

}

public void setWorksCount(Integer worksCount) {

this.worksCount = worksCount;

}

public String getSiteName() {

return siteName;

}

public void setSiteName(String siteName) {

this.siteName = siteName;

}

public String getSitekey() {

return sitekey;

}

public void setSitekey(String sitekey) {

this.sitekey = sitekey;

}

public String getSysCode() {

return sysCode;

}

public void setSysCode(String sysCode){

this.sysCode = sysCode;

}

public String getCreator() {

return creator;

}

public void setCreator(String creator){

this.creator = creator;

}

public Date getGmtCreate() {

return gmtCreate;

}

public void setGmtCreate(Date gmtCreate){

this.gmtCreate = gmtCreate;

}

public String getModifier() {

return modifier;

}

public void setModifier(String modifier){

this.modifier = modifier;

}

public Date getGmtModified() {

return gmtModified;

}

public void setGmtModified(Date gmtModified){

this.gmtModified = gmtModified;

}

public Integer getIsDeleted() {

return isDeleted;

}

public void setIsDeleted(Integer isDeleted){

this.isDeleted = isDeleted;

}

public String getUserName() {

return userName;

}

public void setUserName(String userName){

this.userName = userName;

}

public String getNickName() {

return nickName;

}

public void setNickName(String nickName){

this.nickName = nickName;

}

public String getPassword() {

return password;

}

public void setPassword(String password){

this.password = password;

}

public String getEmail() {

return email;

}

public void setEmail(String email){

this.email = email;

}

public String getMobile() {

return mobile;

}

public void setMobile(String mobile){

this.mobile = mobile;

}

public Integer getUserType() {

return userType;

}

public void setUserType(Integer userType){

this.userType = userType;

}

public Integer getSex() {

return sex;

}

public void setSex(Integer sex){

this.sex = sex;

}

public String getJob() {

return job;

}

public void setJob(String job){

this.job = job;

}

public Integer getAreaId() {

return areaId;

}

public void setAreaId(Integer areaId){

this.areaId = areaId;

}

public String getAreaLongCode() {

return areaLongCode;

}

public void setAreaLongCode(String areaLongCode) {

this.areaLongCode = areaLongCode;

}

public String getProvinceCode() {

return provinceCode;

}

public void setProvinceCode(String provinceCode) {

this.provinceCode = provinceCode;

}

public String getCityCode() {

return cityCode;

}

public void setCityCode(String cityCode) {

this.cityCode = cityCode;

}

public Integer getPicId() {

return picId;

}

public void setPicId(Integer picId) {

this.picId = picId;

}

public Integer getMediaType() {

return mediaType;

}

public void setMediaType(Integer mediaType){

this.mediaType = mediaType;

}

public String getAppKey() {

return appKey;

}

public void setAppKey(String appKey) {

this.appKey = appKey;

}

public String getToken() {

return token;

}

public void setToken(String token) {

this.token = token;

}

public String getAtt3() {

return att3;

}

public void setAtt3(String att3){

this.att3 = att3;

}

public String getAtt4() {

return att4;

}

public void setAtt4(String att4){

this.att4 = att4;

}

public String getAtt5() {

return att5;

}

public void setAtt5(String att5){

this.att5 = att5;

}

public String getAtt6() {

return att6;

}

public void setAtt6(String att6){

this.att6 = att6;

}

public Date getDateAtt1() {

return dateAtt1;

}

public void setDateAtt1(Date dateAtt1){

this.dateAtt1 = dateAtt1;

}

public Date getDateAtt2() {

return dateAtt2;

}

public void setDateAtt2(Date dateAtt2){

this.dateAtt2 = dateAtt2;

}

public Integer getGroupId() {

return groupId;

}

public void setGroupId(Integer groupId){

this.groupId = groupId;

}

public Integer getNumAtt2() {

return numAtt2;

}

public void setNumAtt2(Integer numAtt2){

this.numAtt2 = numAtt2;

}

public Double getNumAtt3() {

return numAtt3;

}

public void setNumAtt3(Double numAtt3){

this.numAtt3 = numAtt3;

}

public Double getNumAtt4() {

return numAtt4;

}

public void setNumAtt4(Double numAtt4){

this.numAtt4 = numAtt4;

}

public String getRemark() {

return remark;

}

public void setRemark(String remark){

this.remark = remark;

}

public String getRealName() {

return realName;

}

public void setRealName(String realName) {

this.realName = realName;

}

public String getQq() {

return qq;

}

public void setQq(String qq) {

this.qq = qq;

}

public String getCompanyTel() {

return companyTel;

}

public void setCompanyTel(String companyTel) {

this.companyTel = companyTel;

}

public String getCompanyAddress() {

return companyAddress;

}

public void setCompanyAddress(String companyAddress) {

this.companyAddress = companyAddress;

}

public String getSpecialityValue() {

return specialityValue;

}

public void setSpecialityValue(String specialityValue) {

this.specialityValue = specialityValue;

}

public String getIntro() {

return intro;

}

public void setIntro(String intro) {

this.intro = intro;

}

public Date getApplicationDate() {

return applicationDate;

}

public void setApplicationDate(Date applicationDate) {

this.applicationDate = applicationDate;

}

public Integer getEmailVerifyState() {

return emailVerifyState;

}

public void setEmailVerifyState(Integer emailVerifyState) {

this.emailVerifyState = emailVerifyState;

}

public Integer getMobileVerifyState() {

return mobileVerifyState;

}

public void setMobileVerifyState(Integer mobileVerifyState) {

this.mobileVerifyState = mobileVerifyState;

}

public Integer getIdcardVerifyState() {

return idcardVerifyState;

}

public void setIdcardVerifyState(Integer idcardVerifyState) {

this.idcardVerifyState = idcardVerifyState;

}

public Integer getLevel() {

return level;

}

public void setLevel(Integer level) {

this.level = level;

}

public String getVerifyUser() {

return verifyUser;

}

public void setVerifyUser(String verifyUser) {

this.verifyUser = verifyUser;

}

public Integer getDeviceRestrict() {

return deviceRestrict;

}

public void setDeviceRestrict(Integer deviceRestrict) {

this.deviceRestrict = deviceRestrict;

}

public Date getVerifyDate() {

return verifyDate;

}

public void setVerifyDate(Date verifyDate) {

this.verifyDate = verifyDate;

}

public String getOrgName() {return orgName; }

public void setOrgName(String orgName) {this.orgName = orgName;}

public Integer getOrgId() {return orgId;}

public void setOrgId(Integer orgId) {this.orgId = orgId;}

public String getAccount() {

return account;

}

public void setAccount(String account) {

this.account = account;

}

public String getUserTypeName() {

return userTypeName;

}

public void setUserTypeName(String userTypeName) {

this.userTypeName = userTypeName;

}

public String getPicPath() {

return picPath;

}

public void setPicPath(String picPath) {

this.picPath = picPath;

}

public String getAreaName() {

return areaName;

}

public void setAreaName(String areaName) {

this.areaName = areaName;

}

public String getJobName() {

return jobName;

}

public void setJobName(String jobName) {

this.jobName = jobName;

}

public String getServiceUrl() {

return serviceUrl;

}

public void setServiceUrl(String serviceUrl) {

this.serviceUrl = serviceUrl;

}

public String getResourcesUrl() {

return resourcesUrl;

}

public void setResourcesUrl(String resourcesUrl) {

this.resourcesUrl = resourcesUrl;

}

public Integer getAttentionState() {

return attentionState;

}

public void setAttentionState(Integer attentionState) {

this.attentionState = attentionState;

}

public Integer getExistsMobile() {

return existsMobile;

}

public void setExistsMobile(Integer existsMobile) {

this.existsMobile = existsMobile;

}

@Override

public boolean equals(Object that) {

if (this == that) {

return true;

}

if (that == null) {

return false;

}

if (getClass() != that.getClass()) {

return false;

}

SysUser other = (SysUser) that;

return (this.getId() == null ? other.getId() == null : this.getId().equals(other.getId()))

&& (this.getSysCode() == null ? other.getSysCode() == null : this.getSysCode().equals(other.getSysCode()))

&& (this.getCreator() == null ? other.getCreator() == null : this.getCreator().equals(other.getCreator()))

&& (this.getGmtCreate() == null ? other.getGmtCreate() == null : this.getGmtCreate().equals(other.getGmtCreate()))

&& (this.getModifier() == null ? other.getModifier() == null : this.getModifier().equals(other.getModifier()))

&& (this.getGmtModified() == null ? other.getGmtModified() == null : this.getGmtModified().equals(other.getGmtModified()))

&& (this.getIsDeleted() == null ? other.getIsDeleted() == null : this.getIsDeleted().equals(other.getIsDeleted()))

&& (this.getUserName() == null ? other.getUserName() == null : this.getUserName().equals(other.getUserName()))

&& (this.getNickName() == null ? other.getNickName() == null : this.getNickName().equals(other.getNickName()))

&& (this.getPassword() == null ? other.getPassword() == null : this.getPassword().equals(other.getPassword()))

&& (this.getEmail() == null ? other.getEmail() == null : this.getEmail().equals(other.getEmail()))

&& (this.getMobile() == null ? other.getMobile() == null : this.getMobile().equals(other.getMobile()))

&& (this.getUserType() == null ? other.getUserType() == null : this.getUserType().equals(other.getUserType()))

&& (this.getSex() == null ? other.getSex() == null : this.getSex().equals(other.getSex()))

&& (this.getJob() == null ? other.getJob() == null : this.getJob().equals(other.getJob()))

&& (this.getAreaId() == null ? other.getAreaId() == null : this.getAreaId().equals(other.getAreaId()))

&& (this.getPicId() == null ? other.getPicId() == null : this.getPicId().equals(other.getPicId()))

&& (this.getMediaType() == null ? other.getMediaType() == null : this.getMediaType().equals(other.getMediaType()))

&& (this.getAppKey() == null ? other.getAppKey() == null : this.getAppKey().equals(other.getAppKey()))

&& (this.getToken() == null ? other.getToken() == null : this.getToken().equals(other.getToken()))

&& (this.getAtt3() == null ? other.getAtt3() == null : this.getAtt3().equals(other.getAtt3()))

&& (this.getAtt4() == null ? other.getAtt4() == null : this.getAtt4().equals(other.getAtt4()))

&& (this.getAtt5() == null ? other.getAtt5() == null : this.getAtt5().equals(other.getAtt5()))

&& (this.getAtt6() == null ? other.getAtt6() == null : this.getAtt6().equals(other.getAtt6()))

&& (this.getDateAtt1() == null ? other.getDateAtt1() == null : this.getDateAtt1().equals(other.getDateAtt1()))

&& (this.getDateAtt2() == null ? other.getDateAtt2() == null : this.getDateAtt2().equals(other.getDateAtt2()))

&& (this.getGroupId() == null ? other.getGroupId() == null : this.getGroupId().equals(other.getGroupId()))

&& (this.getNumAtt2() == null ? other.getNumAtt2() == null : this.getNumAtt2().equals(other.getNumAtt2()))

&& (this.getNumAtt3() == null ? other.getNumAtt3() == null : this.getNumAtt3().equals(other.getNumAtt3()))

&& (this.getNumAtt4() == null ? other.getNumAtt4() == null : this.getNumAtt4().equals(other.getNumAtt4()))

&& (this.getRemark() == null ? other.getRemark() == null : this.getRemark().equals(other.getRemark()))

&& (this.getOrgId() == null ? other.getOrgId() == null : this.getOrgId().equals(other.getOrgId()))

&& (this.getOrgName() == null ? other.getOrgName() == null : this.getOrgName().equals(other.getOrgName()))

&& (this.getBrandIds() == null ? other.getBrandIds() == null : this.getBrandIds().equals(other.getBrandIds()))

;

}

@Override

public int hashCode() {

final int prime = 31;

int result = 1;

result = prime \* result + ((getId() == null) ? 0 : getId().hashCode());

result = prime \* result + ((getSysCode() == null) ? 0 : getSysCode().hashCode());

result = prime \* result + ((getCreator() == null) ? 0 : getCreator().hashCode());

result = prime \* result + ((getGmtCreate() == null) ? 0 : getGmtCreate().hashCode());

result = prime \* result + ((getModifier() == null) ? 0 : getModifier().hashCode());

result = prime \* result + ((getGmtModified() == null) ? 0 : getGmtModified().hashCode());

result = prime \* result + ((getIsDeleted() == null) ? 0 : getIsDeleted().hashCode());

result = prime \* result + ((getUserName() == null) ? 0 : getUserName().hashCode());

result = prime \* result + ((getNickName() == null) ? 0 : getNickName().hashCode());

result = prime \* result + ((getPassword() == null) ? 0 : getPassword().hashCode());

result = prime \* result + ((getEmail() == null) ? 0 : getEmail().hashCode());

result = prime \* result + ((getMobile() == null) ? 0 : getMobile().hashCode());

result = prime \* result + ((getUserType() == null) ? 0 : getUserType().hashCode());

result = prime \* result + ((getSex() == null) ? 0 : getSex().hashCode());

result = prime \* result + ((getJob() == null) ? 0 : getJob().hashCode());

result = prime \* result + ((getAreaId() == null) ? 0 : getAreaId().hashCode());

result = prime \* result + ((getPicId() == null) ? 0 : getPicId().hashCode());

result = prime \* result + ((getMediaType() == null) ? 0 : getMediaType().hashCode());

result = prime \* result + ((getAppKey() == null) ? 0 : getAppKey().hashCode());

result = prime \* result + ((getToken() == null) ? 0 : getToken().hashCode());

result = prime \* result + ((getAtt3() == null) ? 0 : getAtt3().hashCode());

result = prime \* result + ((getAtt4() == null) ? 0 : getAtt4().hashCode());

result = prime \* result + ((getAtt5() == null) ? 0 : getAtt5().hashCode());

result = prime \* result + ((getAtt6() == null) ? 0 : getAtt6().hashCode());

result = prime \* result + ((getDateAtt1() == null) ? 0 : getDateAtt1().hashCode());

result = prime \* result + ((getDateAtt2() == null) ? 0 : getDateAtt2().hashCode());

result = prime \* result + ((getGroupId() == null) ? 0 : getGroupId().hashCode());

result = prime \* result + ((getNumAtt2() == null) ? 0 : getNumAtt2().hashCode());

result = prime \* result + ((getNumAtt3() == null) ? 0 : getNumAtt3().hashCode());

result = prime \* result + ((getNumAtt4() == null) ? 0 : getNumAtt4().hashCode());

result = prime \* result + ((getRemark() == null) ? 0 : getRemark().hashCode());

result = prime \* result + ((getOrgId() == null) ? 0 : getOrgId().hashCode());

result = prime \* result + ((getOrgName() == null) ? 0 : getOrgName().hashCode());

result = prime \* result + ((getBrandIds() == null) ? 0 : getBrandIds().hashCode());

return result;

}

/\*\*获取对象的copy\*\*/

public SysUser copy(){

SysUser obj = new SysUser();

obj.setSysCode(this.sysCode);

obj.setCreator(this.creator);

obj.setGmtCreate(this.gmtCreate);

obj.setModifier(this.modifier);

obj.setGmtModified(this.gmtModified);

obj.setIsDeleted(this.isDeleted);

obj.setUserName(this.userName);

obj.setNickName(this.nickName);

obj.setPassword(this.password);

obj.setEmail(this.email);

obj.setMobile(this.mobile);

obj.setUserType(this.userType);

obj.setSex(this.sex);

obj.setJob(this.job);

obj.setAreaId(this.areaId);

obj.setPicId(this.picId);

obj.setMediaType(this.mediaType);

obj.setAppKey(this.appKey);

obj.setToken(this.token);

obj.setAtt3(this.att3);

obj.setAtt4(this.att4);

obj.setAtt5(this.att5);

obj.setAtt6(this.att6);

obj.setDateAtt1(this.dateAtt1);

obj.setDateAtt2(this.dateAtt2);

obj.setGroupId(this.groupId);

obj.setNumAtt2(this.numAtt2);

obj.setNumAtt3(this.numAtt3);

obj.setNumAtt4(this.numAtt4);

obj.setRemark(this.remark);

obj.setOrgId(this.orgId);

obj.setOrgName(this.orgName);

obj.setBrandIds(this.brandIds);

return obj;

}

/\*\*获取对象的copy\*\*/

public LoginUser toLoginUser(){

LoginUser obj = new LoginUser();

obj.setId(this.id);

obj.setUserType(this.userType);

obj.setLoginId(this.sysCode);

obj.setLoginName(this.nickName);

obj.setLoginPhone(this.mobile==null?"":this.mobile.toString());

obj.setLoginEmail(this.email);

obj.setName(this.userName);

obj.setAppKey(this.appKey);

obj.setToken(this.token);

obj.setDeviceId(this.mediaType==null?"":this.mediaType.toString());

obj.setMsgId(this.getMsgId());

obj.setGroupId(this.groupId);

obj.setSex(this.sex);

obj.setPicPath(this.picPath);

obj.setBrandIds(this.brandIds);

return obj;

}

/\*\*获取对象的map\*\*/

public Map<String,Object> toMap(){

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

map.put("sysCode",this.sysCode);

map.put("creator",this.creator);

map.put("gmtCreate",this.gmtCreate);

map.put("modifier",this.modifier);

map.put("gmtModified",this.gmtModified);

map.put("isDeleted",this.isDeleted);

map.put("userName",this.userName);

map.put("nickName",this.nickName);

map.put("password",this.password);

map.put("email",this.email);

map.put("mobile",this.mobile);

map.put("userType",this.userType);

map.put("sex",this.sex);

map.put("job",this.job);

map.put("areaId",this.areaId);

map.put("picId",this.picId);

map.put("mediaType",this.mediaType);

map.put("appKey",this.appKey);

map.put("token",this.token);

map.put("att3",this.att3);

map.put("att4",this.att4);

map.put("att5",this.att5);

map.put("att6",this.att6);

map.put("dateAtt1",this.dateAtt1);

map.put("dateAtt2",this.dateAtt2);

map.put("numAtt1",this.groupId);

map.put("numAtt2",this.numAtt2);

map.put("numAtt3",this.numAtt3);

map.put("numAtt4",this.numAtt4);

map.put("remark",this.remark);

map.put("orgId",this.orgId);

map.put("orgName",this.orgName);

map.put("brandIds",this.brandIds);

return map;

}

public Double getConsumAmount() {

return consumAmount;

}

public void setConsumAmount(Double consumAmount) {

this.consumAmount = consumAmount;

}

public Double getBalanceAmount() {

return balanceAmount;

}

public void setBalanceAmount(Double balanceAmount) {

this.balanceAmount = balanceAmount;

}

@Override

public String toString() {

return "SysUser [id=" + id + ", userName=" + userName + ", nickName=" + nickName + ", password="

+ password + ", email=" + email + ", mobile=" + mobile + ", userType=" + userType + "]";

}

}