package com.sandu.api.shop.common.constant;

/\*\*

\* 店铺编码前缀enum类型

\*

\* @auth xiaoxc

\* @data 2018-06-11

\*/

public enum ShopCodePrefixEnum {

//品牌馆

BRAND\_PAVILION("BPS"),

//建材家居

BUILDING\_MATERIALS\_HOME("BMS"),

//设计师

DESIGNER("DS"),

//设计公司

DESIGN\_COMPANY("DES"),

//装修公司

DECORATION\_COMPANY("DCS"),

//施工单位

CONSTRUCTION\_UNIT("CUS");

private String code;

ShopCodePrefixEnum(String code) {

this.code = code;

}

public String getCode() {

return code;

}

}

package com.sandu.api.shop.common.constant;

/\*\*

\* 店铺封面图片资源类型

\*

\* @auth xiaoxc

\* @data 2018-06-19

\*/

public class ShopCoverResTypeConstant {

/\*\*

\* 封面图片列表类型

\*/

public static final int COVER\_PIC\_LIST\_TYPE = 1;

/\*\*

\* 封面全景图类型

\*/

public static final int COVER\_PANORAMA\_TYPE = 2;

/\*\*

\* 封面视频类型

\*/

public static final int COVER\_VIDEO\_TYPE = 3;

}

package com.sandu.api.shop.common.constant;

/\*\*

\* 店铺资源类型

\*

\* @auth xiaoxc

\* @data 2018-06-20

\*/

public class ShopResTypeConstant {

/\*\*

\* 店铺logo图片

\*/

public static final String SHOP\_RES\_LOGO\_PIC\_TYPE = "店铺logo图片";

public static final String SHOP\_RES\_LOGO\_PIC\_KEY = "company.shop.logoPic.upload.path";

/\*\*

\* 店铺封面图片

\*/

public static final String SHOP\_RES\_COVER\_PIC\_TYPE = "店铺封面图片";

public static final String SHOP\_RES\_COVER\_PIC\_KEY = "company.shop.coverPic.upload.path";

/\*\*

\* 店铺介绍文本

\*/

public static final String SHOP\_RES\_INTRODUCED\_FILE\_TYPE = "店铺介绍文本";

public static final String SHOP\_RES\_INTRODUCED\_FILE\_KEY = "company.shop.introducedFile.upload.path";

/\*\*

\* 店铺介绍图片

\*/

public static final String SHOP\_RES\_INTRODUCED\_PIC\_TYPE = "店铺介绍图片";

public static final String SHOP\_RES\_INTRODUCED\_PIC\_KEY = "company.shop.introducedPic.upload.path";

/\*\*

\* 工程案例文本

\*/

public static final String SHOP\_RES\_CASE\_FILE\_TYPE = "工程案例文本";

public static final String SHOP\_RES\_CASE\_FILE\_KEY = "company.shop.caseFile.upload.path";

/\*\*

\* 工程案例图片

\*/

public static final String SHOP\_RES\_CASE\_PIC\_TYPE = "工程案例图片";

public static final String SHOP\_RES\_CASE\_PIC\_KEY = "company.shop.casePic.upload.path";

/\*\*

\* 店铺博文图片

\*/

public static final String SHOP\_RES\_ARTICLE\_PIC\_TYPE = "店铺博文图片";

public static final String SHOP\_RES\_ARTICLE\_PIC\_KEY = "company.shop.articlePic.upload.path";

}

package com.sandu.api.shop.common.tool;

/\*\*

\* Created by kono on 2018/3/28 0028.

\*/

public class RandomGeneraterTool {

/\*\*

\* 起始的时间戳

\*/

private final static long START\_STMP = System.currentTimeMillis();

/\*\*

\* 每一部分占用的位数

\*/

private final static long SEQUENCE\_BIT = 12; //序列号占用的位数

private final static long MACHINE\_BIT = 5; //机器标识占用的位数

private final static long DATACENTER\_BIT = 5;//数据中心占用的位数

/\*\*

\* 每一部分的最大值

\*/

private final static long MAX\_DATACENTER\_NUM = -1L ^ (-1L << DATACENTER\_BIT);

private final static long MAX\_MACHINE\_NUM = -1L ^ (-1L << MACHINE\_BIT);

private final static long MAX\_SEQUENCE = -1L ^ (-1L << SEQUENCE\_BIT);

/\*\*

\* 每一部分向左的位移

\*/

private final static long MACHINE\_LEFT = SEQUENCE\_BIT;

private final static long DATACENTER\_LEFT = SEQUENCE\_BIT + MACHINE\_BIT;

private final static long TIMESTMP\_LEFT = DATACENTER\_LEFT + DATACENTER\_BIT;

private long datacenterId; //数据中心

private long machineId; //机器标识

private long sequence = 0L; //序列号

private long lastStmp = -1L;//上一次时间戳

public RandomGeneraterTool(long datacenterId, long machineId) {

if (datacenterId > MAX\_DATACENTER\_NUM || datacenterId < 0) {

throw new IllegalArgumentException("datacenterId can't be greater than MAX\_DATACENTER\_NUM or less than 0");

}

if (machineId > MAX\_MACHINE\_NUM || machineId < 0) {

throw new IllegalArgumentException("machineId can't be greater than MAX\_MACHINE\_NUM or less than 0");

}

this.datacenterId = datacenterId;

this.machineId = machineId;

}

/\*\*

\* 产生下一个ID

\*

\* @return

\*/

public synchronized long nextId() {

long currStmp = getNewstmp();

if (currStmp < lastStmp) {

throw new RuntimeException("Clock moved backwards. Refusing to generate id");

}

if (currStmp != lastStmp) {

//相同毫秒内，序列号自增

sequence = (sequence + 1) & MAX\_SEQUENCE;

//同一毫秒的序列数已经达到最大

if (sequence == 0L) {

currStmp = getNextMill();

}

} else {

//不同毫秒内，序列号置为0

sequence = 0L;

}

lastStmp = currStmp;

return (currStmp - START\_STMP) << TIMESTMP\_LEFT //时间戳部分

| datacenterId << DATACENTER\_LEFT //数据中心部分

| machineId << MACHINE\_LEFT //机器标识部分

| sequence; //序列号部分

}

private long getNextMill() {

long mill = getNewstmp();

while (mill <= lastStmp) {

mill = getNewstmp();

}

return mill;

}

private long getNewstmp() {

return System.currentTimeMillis();

}

public static long getRandom() {

RandomGeneraterTool randomGenerater = new RandomGeneraterTool(2, 3);

return randomGenerater.nextId();

}

public static void main(String[] args) {

System.out.println(getRandom());

}

}

package com.sandu.api.shop.input;

import com.sandu.api.shop.model.CompanyShop;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import org.hibernate.validator.constraints.NotBlank;

import javax.validation.constraints.NotNull;

import javax.validation.constraints.Pattern;

import javax.validation.constraints.Size;

import java.io.Serializable;

/\*\*

\* 新增企业店铺

\*

\* @auth xiaoxc

\* @data 2018-06-04

\*/

@Data

public class CompanyShopAdd implements Serializable {

private static final long serialVersionUID = 6569747046941898171L;

@ApiModelProperty(value = "店铺名称", required = true)

@NotBlank(message = "店铺名称不能为空")

@Size(min = 1, max = 50, message = "店铺名称长度度限{min}-{max}个字符")

private String shopName;

@ApiModelProperty(value = "联系人姓名", required = true)

@NotBlank(message = "联系人姓名不能为空")

@Size(min = 1, max = 50, message = "联系人姓名长度度限{min}-{max}个字符")

private String contactName;

@ApiModelProperty(value = "联系人电话", required = true)

@NotBlank(message = "联系人电话不能为空")

@Pattern(regexp = "^\\b800-[0-9]{6,7}\\b|\\b400-[0-9]{6,7}\\b|\\b((0[0-9]{2,3}-)([1-9][0-9]{6,7}))\\b|\\b(((13[0-9])|(14[5|7])|(15([0-9]))|(17[0-3,5-8])|(18[0-9])|166|198|199)\\d{8}\\b)$",message = "号码格式错误")

private String contactPhone;

@ApiModelProperty(value = "区域省编码", required = true)

@NotBlank(message = "区域信息不能为空")

private String provinceCode;

@ApiModelProperty(value = "区域市编码", required = true)

@NotBlank(message = "请完整选择店铺区域信息")

private String cityCode;

@ApiModelProperty(value = "区域区编码", required = true)

@NotBlank(message = "请完整选择店铺区域信息")

private String areaCode;

@ApiModelProperty(value = "区域街道编码")

private String streetCode;

@ApiModelProperty(value = "详细地址", required = true)

@NotBlank(message = "店铺详细地址不能为空")

@Size(min = 1, max = 100, message = "banner名称长度度限{min}-{max}个字符")

private String shopAddress;

@ApiModelProperty(value = "店铺logo图片Id")

private Integer logoPicId;

@ApiModelProperty(value = "店铺封面资源Ids")

private String coverResIds;

@ApiModelProperty(value = "店铺封面资源类型1:图片列表,2:全景图,3:视频")

private Integer coverResType;

@ApiModelProperty(value = "富文本店铺介绍内容")

private String shopIntroduced;

@ApiModelProperty(value = "富文本店铺介绍图片ids")

private String introducedPicIds;

@ApiModelProperty(value = "分类ids 设计师-擅长风格，施工单位-施工类型，家居建材-产品分类等")

private String categoryIds;

@ApiModelProperty(value = "发布平台类型 1:小程序2:选装网3:同城联盟")

private String platformValues;

@ApiModelProperty(value = "企业ID", required = true)

@NotNull(message = "企业信息不能为空")

private Long companyId;

//add by wangHaiLin

@ApiModelProperty(value = "经度")

private Double longitude;

@ApiModelProperty(value = "纬度")

private Double latitude;

@ApiModelProperty(value = "设计最低价格")

private Double designFeeStarting;

@ApiModelProperty(value = "设计最高价格")

private Double designFeeEnding;

@ApiModelProperty(value = "装修报价区间")

private Double decorationPriceStart;

@ApiModelProperty(value = "装修报价区间")

private Double decorationPriceEnd;

@ApiModelProperty("装修方式:0.半包,1.全包")

private Integer decorationType;

@ApiModelProperty("从业年限")

private Integer workingYears;

/\*\*

\* 转换model对象

\*

\* @return

\*/

public CompanyShop getCompanyShop() {

CompanyShop companyShop = new CompanyShop();

companyShop.setShopName(this.shopName);

companyShop.setContactName(this.contactName);

companyShop.setContactPhone(this.contactPhone);

companyShop.setProvinceCode(this.provinceCode);

companyShop.setCityCode(this.cityCode);

companyShop.setAreaCode(this.areaCode);

companyShop.setStreetCode(this.streetCode);

companyShop.setShopAddress(this.shopAddress);

companyShop.setCategoryIds(this.categoryIds);

companyShop.setReleasePlatformValues(this.platformValues);

companyShop.setLogoPicId(this.logoPicId);

companyShop.setCoverResIds(this.coverResIds);

companyShop.setCoverResType(this.coverResType);

companyShop.setShopIntroduced(this.shopIntroduced);

companyShop.setIntroducedPicIds(this.introducedPicIds);

companyShop.setCompanyId(this.companyId.intValue());

companyShop.setLongitude(this.longitude);

companyShop.setLatitude(this.latitude);

if (null!=this.designFeeStarting){

companyShop.setDesignFeeStarting(this.designFeeStarting);

}

if (null!=this.designFeeEnding){

companyShop.setDesignFeeEnding(this.designFeeEnding);

}

if (workingYears != null){

companyShop.setWorkingYears(this.workingYears);

}

if (decorationPriceStart != null){

companyShop.setDecorationPriceStart(decorationPriceStart);

}

if (decorationPriceEnd != null){

companyShop.setDecorationPriceEnd(decorationPriceEnd);

}

if (decorationType != null){

companyShop.setDecorationType(this.decorationType);

}

return companyShop;

}

}

package com.sandu.api.shop.input;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import org.hibernate.validator.constraints.NotBlank;

import javax.validation.constraints.NotNull;

import javax.validation.constraints.Size;

import java.io.Serializable;

/\*\*

\* 店铺博文新增入参

\* @author WangHaiLin

\* @date 2018/8/9 10:18

\*/

@Data

public class CompanyShopArticleAdd implements Serializable {

@ApiModelProperty(value = "店铺ID", required = true)

@NotNull(message = "店铺Id不能为空")

private Long shopId;

@ApiModelProperty(value = "博文标题", required = true)

@NotBlank(message = "博文标题不能为空")

@Size(min = 1, max = 30, message = "博文标题长度度限{min}-{max}个字符")

private String title;

@ApiModelProperty(value = "图片Ids")

private String picIds;

@ApiModelProperty(value = "博文内容")

private String content;

}

package com.sandu.api.shop.input;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import javax.validation.constraints.NotNull;

import java.io.Serializable;

/\*\*

\* 店铺博文--列表查询入参

\* @author WangHaiLin

\* @date 2018/8/9 10:53

\*/

@Data

public class CompanyShopArticleQuery implements Serializable {

@ApiModelProperty(value = "店铺ID", required = true)

@NotNull(message = "店铺Id不能为空")

private Long shopId;

@ApiModelProperty(value="每页数量:默认10")

private Integer limit;

@ApiModelProperty(value="当前页:默认第一页")

private Integer page;

@ApiModelProperty(value="开始位置")

private Integer start;

}

package com.sandu.api.shop.input;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import javax.validation.constraints.NotNull;

import java.io.Serializable;

/\*\*

\* 修改发布状态

\* @author WangHaiLin

\* @date 2018/8/10 16:59

\*/

@Data

public class CompanyShopArticleReleaseUpdate implements Serializable {

@ApiModelProperty(value = "博文ID", required = true)

@NotNull(message = "博文Id不能为空")

private Long articleId;

@ApiModelProperty(value = "发布状态", required = true)

@NotNull(message = "发布状态不能为空")

private Integer releaseStatus;

}

package com.sandu.api.shop.input;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import org.hibernate.validator.constraints.NotBlank;

import javax.validation.constraints.NotNull;

import javax.validation.constraints.Size;

import java.io.Serializable;

/\*\*

\* 店铺博文修改入参

\* @author WangHaiLin

\* @date 2018/8/9 10:25

\*/

@Data

public class CompanyShopArticleUpdate implements Serializable{

@ApiModelProperty(value = "博文ID", required = true)

@NotNull(message = "博文Id不能为空")

private Long articleId;

@ApiModelProperty(value = "店铺ID", required = true)

@NotNull(message = "店铺Id不能为空")

private Long shopId;

@ApiModelProperty(value = "博文标题", required = true)

@NotBlank(message = "博文标题不能为空")

@Size(min = 1, max = 30, message = "博文标题长度度限{min}-{max}个字符")

private String title;

@ApiModelProperty(value = "博文内容")

private String content;

@ApiModelProperty(value = "图片IDs")

private String picIds;

}

package com.sandu.api.shop.input;

import com.sandu.api.shop.model.po.ShopPO;

import com.sandu.commons.Mapper;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import java.io.Serializable;

/\*\*

\* 查询店铺列表条件

\*

\* @auth xiaoxc

\* @data 2018-06-04

\*/

@Data

public class CompanyShopQuery extends Mapper implements Serializable {

@ApiModelProperty(value="店铺编码")

private String shopCode;

@ApiModelProperty(value="店铺名称")

private String shopName;

@ApiModelProperty(value = "联系人姓名")

private String contactName;

@ApiModelProperty(value = "联系人电话")

private String contactPhone;

@ApiModelProperty(value = "区域省编码")

private String provinceCode;

@ApiModelProperty(value = "区域市编码")

private String cityCode;

@ApiModelProperty(value = "区域区编码")

private String areaCode;

@ApiModelProperty(value = "区域街道编码")

private String streetCode;

@ApiModelProperty(value = "企业Id")

private Long companyId;

@ApiModelProperty(value = "企业类型")

private String companyType;

@ApiModelProperty(value = "用户Id")

private Integer userId;

/\*\*

\* 转换PO对象

\* @param query

\* @return

\*/

public ShopPO getShopPOFromShopQuery(CompanyShopQuery query) {

ShopPO shopPO = new ShopPO();

if (query != null) {

shopPO.setShopCode(query.getShopCode());

shopPO.setShopName(query.getShopName());

shopPO.setContactName(query.getContactName());

shopPO.setContactPhone(query.getContactPhone());

shopPO.setProvinceCode(query.getProvinceCode());

shopPO.setCityCode(query.getCityCode());

shopPO.setAreaCode(query.getAreaCode());

shopPO.setStreetCode(query.getStreetCode());

shopPO.setCompanyId(query.getCompanyId());

shopPO.setCompanyType(query.getCompanyType());

shopPO.setUserId(query.getUserId());

shopPO.setStart(query.getStart());

shopPO.setLimit(query.getLimit());

}

return shopPO;

}

}

package com.sandu.api.shop.input;

import com.sandu.api.shop.model.po.ShopPO;

import com.sandu.commons.Mapper;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import java.io.Serializable;

/\*\*

\* 查询店铺列表条件

\*

\* @auth xiaoxc

\* @data 2018-06-04

\*/

@Data

public class CompanyShopQuery extends Mapper implements Serializable {

@ApiModelProperty(value="店铺编码")

private String shopCode;

@ApiModelProperty(value="店铺名称")

private String shopName;

@ApiModelProperty(value = "联系人姓名")

private String contactName;

@ApiModelProperty(value = "联系人电话")

private String contactPhone;

@ApiModelProperty(value = "区域省编码")

private String provinceCode;

@ApiModelProperty(value = "区域市编码")

private String cityCode;

@ApiModelProperty(value = "区域区编码")

private String areaCode;

@ApiModelProperty(value = "区域街道编码")

private String streetCode;

@ApiModelProperty(value = "企业Id")

private Long companyId;

@ApiModelProperty(value = "企业类型")

private String companyType;

@ApiModelProperty(value = "用户Id")

private Integer userId;

/\*\*

\* 转换PO对象

\* @param query

\* @return

\*/

public ShopPO getShopPOFromShopQuery(CompanyShopQuery query) {

ShopPO shopPO = new ShopPO();

if (query != null) {

shopPO.setShopCode(query.getShopCode());

shopPO.setShopName(query.getShopName());

shopPO.setContactName(query.getContactName());

shopPO.setContactPhone(query.getContactPhone());

shopPO.setProvinceCode(query.getProvinceCode());

shopPO.setCityCode(query.getCityCode());

shopPO.setAreaCode(query.getAreaCode());

shopPO.setStreetCode(query.getStreetCode());

shopPO.setCompanyId(query.getCompanyId());

shopPO.setCompanyType(query.getCompanyType());

shopPO.setUserId(query.getUserId());

shopPO.setStart(query.getStart());

shopPO.setLimit(query.getLimit());

}

return shopPO;

}

}

package com.sandu.api.shop.input;

import com.sandu.api.shop.model.CompanyShop;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import org.hibernate.validator.constraints.NotBlank;

import javax.validation.constraints.NotNull;

import javax.validation.constraints.Pattern;

import javax.validation.constraints.Size;

import java.io.Serializable;

/\*\*

\* 新增企业店铺

\*

\* @auth xiaoxc

\* @data 2018-06-04

\*/

@Data

public class CompanyShopUpdate implements Serializable {

private static final long serialVersionUID = 6569747046941898171L;

@ApiModelProperty(value = "店铺Id", required = true)

@NotNull(message = "店铺Id不能为空")

private Integer shopId;

@ApiModelProperty(value = "店铺名称", required = true)

@NotBlank(message = "店铺名称不能为空")

@Size(min = 1, max = 50, message = "店铺名称长度度限{min}-{max}个字符")

private String shopName;

@ApiModelProperty(value = "联系人姓名", required = true)

@NotBlank(message = "联系人姓名不能为空")

@Size(min = 1, max = 50, message = "联系人姓名长度度限{min}-{max}个字符")

private String contactName;

@ApiModelProperty(value = "联系人电话", required = true)

@NotBlank(message = "联系人电话不能为空")

@Pattern(regexp = "^\\b800-[0-9]{6,7}\\b|\\b400-[0-9]{6,7}\\b|\\b((0[0-9]{2,3}-)([1-9][0-9]{6,7}))\\b|\\b(((13[0-9])|(14[5|7])|(15([0-9]))|(17[0-3,5-8])|(18[0-9])|166|198|199)\\d{8}\\b)$",message = "号码格式错误")

private String contactPhone;

@ApiModelProperty(value = "区域省编码", required = true)

@NotBlank(message = "区域信息不能为空")

private String provinceCode;

@ApiModelProperty(value = "区域市编码", required = true)

@NotBlank(message = "请完整选择店铺区域信息")

private String cityCode;

@ApiModelProperty(value = "区域区编码", required = true)

@NotBlank(message = "请完整选择店铺区域信息")

private String areaCode;

@ApiModelProperty(value = "区域街道编码")

private String streetCode;

@ApiModelProperty(value = "详细地址", required = true)

@NotBlank(message = "店铺详细地址不能为空")

@Size(min = 1, max = 100, message = "banner名称长度度限{min}-{max}个字符")

private String shopAddress;

@ApiModelProperty(value = "店铺logo图片Id")

private Integer logoPicId;

@ApiModelProperty(value = "店铺封面资源Ids")

private String coverResIds;

@ApiModelProperty(value = "店铺封面资源类型1:图片列表,2:全景图,3:视频")

private Integer coverResType;

@ApiModelProperty(value = "富文本店铺介绍内容")

private String shopIntroduced;

@ApiModelProperty(value = "富文本店铺介绍图片ids")

private String introducedPicIds;

@ApiModelProperty(value = "分类ids 设计师-擅长风格，施工单位-施工类型，家居建材-产品分类等")

private String categoryIds;

@ApiModelProperty(value = "发布平台类型 1:小程序2:选装网3:同城联盟")

private String platformValues;

@ApiModelProperty(value = "发布状态 1:发布，0:未发布")

private Integer displayStatus;

//add by wangHaiLin

@ApiModelProperty(value = "经度")

private Double longitude;

@ApiModelProperty(value = "纬度")

private Double latitude;

@ApiModelProperty(value = "设计最低价格")

private Double designFeeStarting;

@ApiModelProperty(value = "设计最高价格")

private Double designFeeEnding;

@ApiModelProperty(value = "装修报价区间")

private Double decorationPriceStart;

@ApiModelProperty(value = "装修报价区间")

private Double decorationPriceEnd;

@ApiModelProperty("装修方式:0.半包,1.全包")

private Integer decorationType;

@ApiModelProperty("从业年限")

private Integer workingYears;

/\*\*

\* 转换model对象

\*

\* @param companyShop

\* @param update

\* @return

\*/

public CompanyShop getCompanyShopFromCompanyShopUpdate(CompanyShop companyShop, CompanyShopUpdate update) {

if (update != null) {

companyShop.setId(update.getShopId());

companyShop.setShopName(update.getShopName());

companyShop.setContactName(update.getContactName());

companyShop.setContactPhone(update.getContactPhone());

companyShop.setProvinceCode(update.getProvinceCode());

companyShop.setCityCode(update.getCityCode());

companyShop.setAreaCode(update.getAreaCode());

companyShop.setStreetCode(update.getStreetCode());

companyShop.setShopAddress(update.getShopAddress());

companyShop.setCategoryIds(update.getCategoryIds());

companyShop.setReleasePlatformValues(update.getPlatformValues());

companyShop.setLogoPicId(update.getLogoPicId());

companyShop.setCoverResIds(update.getCoverResIds());

companyShop.setCoverResType(update.getCoverResType());

companyShop.setShopIntroduced(update.getShopIntroduced());

companyShop.setLatitude(update.getLatitude());

companyShop.setLongitude(update.getLongitude());

companyShop.setDesignFeeStarting(update.getDesignFeeStarting());

companyShop.setDesignFeeEnding(update.getDesignFeeEnding());

companyShop.setWorkingYears(update.getWorkingYears());

companyShop.setDecorationType(update.getDecorationType());

companyShop.setDecorationPriceStart(update.getDecorationPriceStart());

companyShop.setDecorationPriceEnd(update.getDecorationPriceEnd());

}

return companyShop;

}

}

package com.sandu.api.shop.input;

import com.sandu.api.shop.model.ProjectCase;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import org.hibernate.validator.constraints.NotBlank;

import javax.validation.constraints.NotNull;

import javax.validation.constraints.Size;

import java.io.Serializable;

import java.math.BigInteger;

/\*\*

\* 新增企业店铺

\*

\* @auth xiaoxc

\* @data 2018-06-19

\*/

@Data

public class ProjectCaseAdd implements Serializable {

@ApiModelProperty(value = "店铺ID", required = true)

@NotNull(message = "店铺Id不能为空")

private Integer shopId;

@ApiModelProperty(value = "案例标题", required = true)

@NotBlank(message = "案例标题不能为空")

@Size(min = 1, max = 30, message = "案例标题长度度限{min}-{max}个字符")

private String caseTitle;

@ApiModelProperty(value = "案例内容")

private String content;

@ApiModelProperty(value = "图片IDs")

private String picIds;

/\*\*

\* 转换model对象

\*

\* @param caseAdd

\* @return

\*/

public ProjectCase getProjectCaseFromProjectCaseAdd(ProjectCaseAdd caseAdd) {

ProjectCase projectCase = new ProjectCase();

if (caseAdd != null) {

projectCase.setCaseTitle(caseAdd.getCaseTitle());

projectCase.setContent(caseAdd.getContent());

projectCase.setShopId(caseAdd.getShopId());

projectCase.setPicIds(caseAdd.getPicIds());

}

return projectCase;

}

}

package com.sandu.api.shop.input;

import com.sandu.api.shop.model.po.CasePO;

import com.sandu.api.shop.model.po.ShopPO;

import com.sandu.commons.Mapper;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import java.io.Serializable;

/\*\*

\* 查询工程案例列表条件

\*

\* @auth xiaoxc

\* @data 2018-06-20

\*/

@Data

public class ProjectCaseQuery extends Mapper implements Serializable {

@ApiModelProperty(value="标题")

private String caseTitle;

@ApiModelProperty(value="店铺Id")

private Integer shopId;

/\*\*

\* 转换PO对象

\* @param query

\* @return

\*/

public CasePO getCasePOFromCaseQuery(ProjectCaseQuery query) {

CasePO casePO = new CasePO();

if (query != null) {

casePO.setCaseTitle(query.getCaseTitle());

casePO.setShopId(query.getShopId());

casePO.setStart(query.getStart());

casePO.setLimit(query.getLimit());

}

return casePO;

}

}

package com.sandu.api.shop.input;

import com.sandu.api.shop.model.ProjectCase;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import org.hibernate.validator.constraints.NotBlank;

import org.hibernate.validator.constraints.NotEmpty;

import javax.validation.constraints.NotNull;

import javax.validation.constraints.Size;

import java.io.Serializable;

/\*\*

\* 新增企业店铺

\*

\* @auth xiaoxc

\* @data 2018-06-19

\*/

@Data

public class ProjectCaseUpdate implements Serializable {

@ApiModelProperty(value = "案例ID", required = true)

@NotNull(message = "案例Id不能为空")

private Integer caseId;

@ApiModelProperty(value = "店铺ID", required = true)

@NotNull(message = "店铺Id不能为空")

private Integer shopId;

@ApiModelProperty(value = "案例标题", required = true)

@NotBlank(message = "案例标题不能为空")

@Size(min = 1, max = 30, message = "案例标题长度度限{min}-{max}个字符")

private String caseTitle;

@ApiModelProperty(value = "案例内容")

private String content;

@ApiModelProperty(value = "图片IDs")

private String picIds;

@ApiModelProperty(value = "发布状态(1：发布，0：取消发布)")

private Integer releaseStatus;

/\*\*

\* 转换model对象

\*

\* @param caseUpdate

\* @return

\*/

public ProjectCase getProjectCaseFromProjectCaseUpdate(ProjectCaseUpdate caseUpdate) {

ProjectCase projectCase = new ProjectCase();

if (caseUpdate != null) {

projectCase.setId(caseUpdate.getCaseId());

projectCase.setCaseTitle(caseUpdate.getCaseTitle());

projectCase.setContent(caseUpdate.getContent());

projectCase.setShopId(caseUpdate.getShopId());

projectCase.setPicIds(caseUpdate.getPicIds());

}

return projectCase;

}

}

package com.sandu.api.shop.model;

import com.sandu.commons.Mapper;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import java.io.Serializable;

import java.util.Date;

/\*\*

\* 企业商铺-实体类

\* @auth xiaoxc

\* @date 2018-06-04

\* @version V1.0

\*/

@Data

public class CompanyShop extends Mapper implements Serializable{

private static final long serialVersionUID = 1L;

/\*\* 主键ID \*\*/

private Integer id;

/\*\* 企业Id \*\*/

private Integer companyId;

/\*\* 企业父ID(经销商-->厂商) \*\*/

private Integer companyPid;

/\*\* 用户Id \*\*/

private Integer userId;

/\*\* 店铺编码 \*\*/

private String shopCode;

/\*\* 店铺名称 \*\*/

private String shopName;

/\*\* 店铺类型 设计师、设计公司、装修公司、工长、品牌馆、家居建材 \*\*/

private Integer businessType;

/\*\* 分类ids（设计师---擅长风格，施工单位---施工类型，家居建材---产品分类等）\*\*/

private String categoryIds;

/\*\* 一级分类ids（建材、家居、电器）\*\*/

private String firstCategoryIds;

/\*\* 省编码 \*\*/

private String provinceCode;

/\*\* 市编码 \*\*/

private String cityCode;

/\*\* 区编码 \*\*/

private String areaCode;

/\*\* 街道编码 \*\*/

private String streetCode;

/\*\* 地区长编码 \*\*/

private String longAreaCode;

/\*\* 详细地址 \*\*/

private String shopAddress;

/\*\* 联系电话 \*\*/

private String contactPhone;

/\*\* 联系人姓名 \*\*/

private String contactName;

/\*\* logo图ID \*\*/

private Integer logoPicId;

/\*\* 店铺封面资源Ids \*\*/

private String coverResIds;

/\*\* 店铺封面资源类型 1:图片列表,2:全景图,3:视频 \*\*/

private Integer coverResType;

/\*\* 发布平台类型 1:小程序2:选装网3:同城联盟 \*\*/

private String releasePlatformValues;

/\*\* 显示状态(1是0否) \*\*/

private Integer displayStatus;

/\*\* 访问次数 \*\*/

private Integer visitCount;

/\*\* 好评率 \*\*/

private Double praiseRate;

/\*\* 店铺介绍 \*\*/

private String shopIntroduced;

/\*\* 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;

/\*\* 备注 \*\*/

private String remark;

//富文本介绍文本Id

private Long introducedFileId;

//富文本介绍图片Ids

private String introducedPicIds;

//add by wangHaiLin

/\*\* 经度 \*\*/

private Double longitude;

/\*\* 纬度 \*\*/

private Double latitude;

/\*\*设计最低价格\*\*/

private Double designFeeStarting;

/\*\*设计最高价格\*\*/

private Double designFeeEnding;

/\*\*装修报价区间\*\*/

private Double decorationPriceStart;

/\*\*装修报价区间\*\*/

private Double decorationPriceEnd;

/\*\*装修方式:0.半包,1.全包\*\*/

private Integer decorationType;

/\*\*从业年限\*\*/

private Integer workingYears;

}

package com.sandu.api.shop.model;

import lombok.Data;

import java.io.Serializable;

import java.util.Date;

/\*\*

\* 店铺博文对应实体

\* @author WangHaiLin

\* @date 2018/8/9 10:04

\*/

@Data

public class CompanyShopArticle implements Serializable {

/\*\*主键ID\*\*/

private Long id;

/\*\*店铺ID\*\*/

private Long shopId;

/\*\*博文标题\*\*/

private String title;

/\*\*发布状态（1是；0否）\*\*/

private Integer releaseStatus;

/\*\*图片Ids\*\*/

private String picIds;

/\*\*内容\*\*/

private String content;

/\*\*系统编码\*\*/

private String sysCode;

/\*\*创建者\*\*/

private String creator;

/\*\*创建时间\*\*/

private Date gmtCreate;

/\*\*更新者\*\*/

private String modifier;

/\*\*更新时间\*\*/

private Date gmtModified;

/\*\*是否删除\*\*/

private Integer isDeleted;

/\*\*备注\*\*/

private String remark;

/\*\*浏览次数\*\*/

private Integer browseCount;

/\*\*发布时间\*\*/

private Date releaseTime;

}

package com.sandu.api.shop.model;

import lombok.Data;

import java.io.Serializable;

import java.util.Date;

/\*\*

\* 店铺-方案表--实体类

\* @author WangHaiLin

\* @date 2018/8/30 16:47

\*/

@Data

public class CompanyShopDesignPlan implements Serializable {

/\*\* 主键ID \*\*/

private Long id;

/\*\* 店铺Id \*\*/

private Long shopId;

/\*\* 方案ID \*\*/

private Long planId;

/\*\* 创建者 \*\*/

private String creator;

/\*\* 创建时间 \*\*/

private Date gmtCreate;

/\*\* 修改者\*\*/

private String modifier;

/\*\* 修改时间 \*\*/

private Date gmtModified;

/\*\* 发布OR 取消发布\*\*/

private Integer isDeleted;

/\*\* 方案类型 1普通 2智能\*\*/

private Integer planRecommendedType;

}

package com.sandu.api.shop.model;

import lombok.Data;

import java.io.Serializable;

import java.math.BigInteger;

import java.util.Date;

/\*\*

\* 工程案例--实体类

\*

\* @auth xiaoxc

\* @data 2018-06-19

\*/

@Data

public class ProjectCase implements Serializable {

/\*\* 主键ID \*\*/

private Integer id;

/\*\* 店铺Id \*\*/

private Integer shopId;

/\*\* 案例标题 \*\*/

private String caseTitle;

/\*\* 发布状态(1是0否) \*\*/

private Integer releaseStatus;

/\*\* 图片Id \*\*/

/\* private BigInteger picId;\*/

/\*\* 文本内容 \*\*/

private String content;

/\*\* 文本内容 \*\*/

private BigInteger fileId;

/\*\* 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;

/\*\* 备注 \*\*/

private String remark;

/\*\* 富文本图片ids \*\*/

private String picIds;

}

package com.sandu.api.shop.output;

import com.sandu.api.pic.model.ResPic;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import java.io.Serializable;

import java.util.List;

/\*\*

\* 店铺博文--详情展示--输出实体

\* @author WangHaiLin

\* @date 2018/8/9 10:36

\*/

@Data

public class CompanyShopArticleDetailVO implements Serializable{

@ApiModelProperty(value = "博文Id")

private Long articleId;

@ApiModelProperty(value = "店铺ID")

private Long shopId;

@ApiModelProperty(value = "博文标题")

private String title;

@ApiModelProperty(value = "博文内容")

private String content;

@ApiModelProperty(value = "图片Ids")

private String picIds;

@ApiModelProperty(value = "博文图片集合")

private List<ResPic> resPics;

}

package com.sandu.api.shop.output;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import java.io.Serializable;

import java.util.Date;

/\*\*

\* 店铺博文列表展示实体

\* @author WangHaiLin

\* @date 2018/8/9 10:31

\*/

@Data

public class CompanyShopArticleListVO implements Serializable {

@ApiModelProperty(value="博文Id")

private Long articleId;

@ApiModelProperty(value="博文标题")

private String articleTitle;

@ApiModelProperty(value="图片路径")

private String picPath;

@ApiModelProperty(value="创建时间")

private Date createDate;

@ApiModelProperty(value="发布状态(发布：1,未发布：0)")

private Integer releaseStatus;

}

package com.sandu.api.shop.output;

import com.sandu.api.pic.model.po.ResPicPO;

import com.sandu.api.shop.model.CompanyShop;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import java.io.Serializable;

import java.util.List;

/\*\*

\* 店铺详情界面

\*

\* @auth xiaoxc

\* @data 2018-06-06

\*/

@Data

public class CompanyShopDetailsVO implements Serializable {

@ApiModelProperty(value="店铺Id")

private Integer shopId;

@ApiModelProperty(value="店铺编码")

private String shopCode;

@ApiModelProperty(value="店铺名称")

private String shopName;

@ApiModelProperty(value = "联系人姓名")

private String contactName;

@ApiModelProperty(value = "联系人电话")

private String contactPhone;

@ApiModelProperty(value = "详细地址")

private String shopAddress;

@ApiModelProperty(value = "店铺类型值(品牌馆or家居建材or设计师or设计公司or装修公司or工长)")

private Integer businessTypeValue;

@ApiModelProperty(value = "店铺类型名称(品牌馆or家居建材or设计师or设计公司or装修公司or工长)")

private String businessTypeName;

@ApiModelProperty(value = "分类信息值(产品分类or擅长风格or施工类型)")

private String categoryIds;

@ApiModelProperty(value = "分类信息名称(产品分类or擅长风格or施工类型)")

private String categoryName;

@ApiModelProperty(value = "区域省编码")

private String provinceCode;

@ApiModelProperty(value = "区域市编码")

private String cityCode;

@ApiModelProperty(value = "区域区编码")

private String areaCode;

@ApiModelProperty(value = "区域街道编码")

private String streetCode;

@ApiModelProperty(value = "区域省名称")

private String provinceName;

@ApiModelProperty(value = "区域市名称")

private String cityName;

@ApiModelProperty(value = "区域区名称")

private String areaName;

@ApiModelProperty(value = "区域街道名称")

private String streetName;

@ApiModelProperty(value = "logoId")

private Integer logoId;

@ApiModelProperty(value = "logo地址")

private String logoUrl;

@ApiModelProperty(value = "店铺封面资源Ids")

private String coverResIds;

@ApiModelProperty(value = "店铺封面资源集合地址")

private List<ResPicPO> coverList;

@ApiModelProperty(value = "店铺封面资源类型1:图片列表,2:全景图,3:视频")

private Integer coverResType;

@ApiModelProperty(value = "店铺介绍")

private String shopIntroduced;

@ApiModelProperty(value = "店铺介绍文本Id")

private Long introducedFileId;

@ApiModelProperty(value = "店铺介绍文本路径")

private String introducedFilePath;

@ApiModelProperty(value = "发布平台value")

private String releasePlatformValues;

@ApiModelProperty(value = "发布平台名称")

private String releasePlatformName;

//add by wangHaiLin

@ApiModelProperty(value = "经度")

private Double longitude;

@ApiModelProperty(value = "纬度")

private Double latitude;

@ApiModelProperty(value = "设计最低价格")

private Double designFeeStarting;

@ApiModelProperty(value = "设计最高价格")

private Double designFeeEnding;

/\*\*装修报价区间\*\*/

private Double decorationPriceStart;

/\*\*装修报价区间\*\*/

private Double decorationPriceEnd;

/\*\*装修方式:0.半包,1.全包\*\*/

private Integer decorationType;

/\*\*从业年限\*\*/

private Integer workingYears;

/\*\*

\* 转换vo对象

\*

\* @param shop

\* @return

\*/

public CompanyShopDetailsVO getShopDetailsVOFromShop(CompanyShop shop) {

CompanyShopDetailsVO shopVO = new CompanyShopDetailsVO();

if (shop != null) {

shopVO.setShopId(shop.getId());

shopVO.setShopCode(shop.getShopCode());

shopVO.setShopName(shop.getShopName());

shopVO.setBusinessTypeValue(shop.getBusinessType());

shopVO.setContactName(shop.getContactName());

shopVO.setContactPhone(shop.getContactPhone());

shopVO.setProvinceCode(shop.getProvinceCode());

shopVO.setCityCode(shop.getCityCode());

shopVO.setAreaCode(shop.getAreaCode());

shopVO.setStreetCode(shop.getStreetCode());

shopVO.setCategoryIds(shop.getCategoryIds());

shopVO.setShopAddress(shop.getShopAddress());

shopVO.setShopIntroduced(shop.getShopIntroduced());

shopVO.setReleasePlatformValues(shop.getReleasePlatformValues());

shopVO.setLogoId(shop.getLogoPicId());

shopVO.setCoverResIds(shop.getCoverResIds());

shopVO.setCoverResType(shop.getCoverResType());

shopVO.setIntroducedFileId(shop.getIntroducedFileId());

shopVO.setLatitude(shop.getLatitude());

shopVO.setLongitude(shop.getLongitude());

shopVO.setDecorationPriceStart(shop.getDecorationPriceStart());

shopVO.setDecorationPriceEnd(shop.getDecorationPriceEnd());

shopVO.setDecorationType(shop.getDecorationType());

shopVO.setWorkingYears(shop.getWorkingYears());

if (shop.getBusinessType()==4||shop.getBusinessType()==3){

shopVO.setDesignFeeStarting(shop.getDesignFeeStarting());

shopVO.setDesignFeeEnding(shop.getDesignFeeEnding());

}

}

return shopVO;

}

}

package com.sandu.api.shop.output;

import com.sandu.api.shop.model.CompanyShop;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import java.io.Serializable;

/\*\*

\* 店铺管理列表展示Vo界面

\*

\* @auth xiaoxc

\* @data 2018-06-04

\*/

@Data

public class CompanyShopVO implements Serializable {

@ApiModelProperty(value="店铺Id")

private Integer shopId;

@ApiModelProperty(value="店铺编码")

private String shopCode;

@ApiModelProperty(value="店铺名称")

private String shopName;

@ApiModelProperty(value = "联系人姓名")

private String contactName;

@ApiModelProperty(value = "联系人电话")

private String contactPhone;

@ApiModelProperty(value = "区域区名称")

private String areaName;

@ApiModelProperty(value = "发布平台名称")

private String releasePlatformName;

@ApiModelProperty(value = "发布状态 1:发布，0:未发布")

private Integer displayStatus;

/\*\*

\* 转换vo对象

\*

\* @param shop

\* @return

\*/

public CompanyShopVO getCompanyShopVOFromCompanyShop(CompanyShop shop) {

CompanyShopVO shopVO = new CompanyShopVO();

if (shop != null) {

shopVO.setShopId(shop.getId());

shopVO.setShopCode(shop.getShopCode());

shopVO.setShopName(shop.getShopName());

shopVO.setContactName(shop.getContactName());

shopVO.setContactPhone(shop.getContactPhone());

shopVO.setDisplayStatus(shop.getDisplayStatus());

}

return shopVO;

}

}

package com.sandu.api.shop.output;

import lombok.Data;

import java.io.Serializable;

/\*\*

\* Created by kono on 2018/6/23 0023.

\*/

@Data

public class CompanyTypeVO implements Serializable {

//公司Id

private Long companyId;

//公司类型(1:厂商，其他非厂商)

private Integer companyType;

//是否是超级管理员

private Integer amdinUser;

}

package com.sandu.api.shop.output;

import com.sandu.api.pic.model.ResPic;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import java.io.Serializable;

import java.util.List;

/\*\*

\* 工程案例展示详情vo界面

\*

\* @auth xiaoxc

\* @data 2018-06-19

\*/

@Data

public class ProjectCaseDetailsVO implements Serializable {

@ApiModelProperty(value="案例Id")

private Integer caseId;

@ApiModelProperty(value="案例标题")

private String caseTitle;

@ApiModelProperty(value="案例内容文本路径")

private String filePath;

@ApiModelProperty(value="案例内容文本")

private String content;

@ApiModelProperty(value = "图片Ids")

private String picIds;

@ApiModelProperty(value = "博文图片集合")

private List<ResPic> resPics;

}

package com.sandu.api.shop.output;

import io.swagger.annotations.ApiModelProperty;

import lombok.Data;

import java.io.Serializable;

import java.util.Date;

/\*\*

\* 工程案例展示列表vo界面

\*

\* @auth xiaoxc

\* @data 2018-06-19

\*/

@Data

public class ProjectCaseVO implements Serializable {

@ApiModelProperty(value="案例Id")

private Integer caseId;

@ApiModelProperty(value="案例标题")

private String caseTitle;

@ApiModelProperty(value="图片路径")

private String picPath;

@ApiModelProperty(value="图片ids")

private String picIds;

@ApiModelProperty(value="创建时间")

private Date createDate;

@ApiModelProperty(value="富文本地址")

private String filePath;

@ApiModelProperty(value="发布状态(发布：1,未发布：0)")

private Integer releaseStatus;

}

package com.sandu.api.shop.service.biz;

import org.springframework.stereotype.Component;

/\*\*

\* @author WangHaiLin

\* @date 2018/8/31 9:44

\*/

@Component

public interface ShopBizService {

/\*\*

\* 删除用户店铺，博文，案例，方案

\* @param userId 用户Id

\* @return -1:不存在店铺；0:删除失败；其他数字：删除成功

\*/

int delUsersShop(Long userId,String loginUserName);

}

package com.sandu.api.shop.service;

import com.sandu.api.shop.input.CompanyShopArticleAdd;

import com.sandu.api.shop.input.CompanyShopArticleQuery;

import com.sandu.api.shop.input.CompanyShopArticleReleaseUpdate;

import com.sandu.api.shop.input.CompanyShopArticleUpdate;

import com.sandu.api.shop.model.CompanyShopArticle;

import com.sandu.api.shop.output.CompanyShopArticleDetailVO;

import com.sandu.api.shop.output.CompanyShopArticleListVO;

import com.sandu.commons.LoginUser;

import org.springframework.stereotype.Component;

import java.util.List;

/\*\*

\* 店铺博文-Service层接口

\* @author WangHaiLin

\* @date 2018/8/9 10:14

\*/

@Component

public interface CompanyShopArticleService {

/\*\*

\* 新增

\* @param articleAdd 新增对象

\* @param loginUser 当前用户对象

\* @return long 新增数据Id

\*/

Long add(CompanyShopArticleAdd articleAdd, LoginUser loginUser);

/\*\*

\* 删除

\* @param id 主键Id

\* @param loginUser 当前登录用户对象

\* @return 删除数量

\*/

int delete(Long id, LoginUser loginUser);

/\*\*

\* 更新

\* @param articleUpdate 修改对象

\* @param loginUser 当前登录用户对象

\* @return 更新数量

\*/

int update(CompanyShopArticleUpdate articleUpdate, LoginUser loginUser);

/\*\*

\* 获取满足条件的数据列表

\* @param query 查询条件

\* @return 集合列表

\*/

List<CompanyShopArticleListVO> findList(CompanyShopArticleQuery query);

/\*\*

\* 获取满足条件的数据数量

\* @param query 查询条件

\* @return int

\*/

Integer getCount(CompanyShopArticleQuery query);

/\*\*

\* 查看店铺博文详情

\* @param articleId 店铺博文Id

\* @return CompanyShopArticleDetailVO 博文详情

\*/

CompanyShopArticleDetailVO select(Long articleId);

/\*\*

\* 发布店铺博文

\* @param update 博文发布入参

\* @param loginUser 当前登录用户

\* @return boolean

\*/

boolean release(CompanyShopArticleReleaseUpdate update, LoginUser loginUser);

/\*\*

\* 根据店铺Id批量删除

\* @param shopId 店铺Id

\* @param loginUser 当前用户对象

\* @return 删除数量

\*/

int deleteByShopId(Long shopId, LoginUser loginUser);

/\*\*

\* add by WangHL

\* 通过店铺Id删除属于店铺的博文列表

\* @param shopIds 店铺Id

\*

\* @return list

\*/

Integer deleteArticleByShopId(List<Integer> shopIds,String loginUserName);

/\*\*

\* add by WangHL

\* 通过店铺Id查找属于店铺的博文数量

\* @param shopIds 店铺Id

\* @return list

\*/

Integer getArticleByShopId(List<Integer> shopIds);

}

package com.sandu.api.shop.service;

import org.springframework.stereotype.Component;

import java.util.List;

/\*\*

\* 店铺方案Service层

\* @author WangHaiLin

\* @date 2018/8/30 16:58

\*/

@Component

public interface CompanyShopDesignPlanService {

/\*\*

\* 通过店铺Id删除属于店铺的方案列表

\* @param shopIds 店铺Id

\* @return list 方案列表

\*/

Integer deleteDesignPlanByShopId(List<Integer> shopIds);

/\*\*

\* 通过店铺Id查找属于店铺的方案数量

\* @param shopIds 店铺Id

\* @return list 方案列表

\*/

Integer getDesignPlanByShopId(List<Integer> shopIds);

}

package com.sandu.api.shop.service;

import com.sandu.api.shop.input.CompanyShopAdd;

import com.sandu.api.shop.input.CompanyShopQuery;

import com.sandu.api.shop.input.CompanyShopUpdate;

import com.sandu.api.shop.model.CompanyShop;

import com.sandu.api.shop.output.CompanyShopDetailsVO;

import com.sandu.api.shop.output.CompanyShopVO;

import com.sandu.commons.LoginUser;

import org.springframework.stereotype.Component;

import java.util.List;

/\*\*

\* 企业店铺-service接口

\*

\* @auth xiaoxc

\* @data 2018-06-04

\*/

@Component

public interface CompanyShopService {

/\*\*

\* 新增店铺

\* @param shopAdd 新增对象

\* @param loginUser 当前用户对象

\* @return int 新增数据Id

\*/

int addShop(CompanyShopAdd shopAdd, LoginUser loginUser);

/\*\*

\* 删除店铺

\* @param shopId 店铺ID

\* @param loginUser 当前用户对象

\* @return 删除数量

\*/

int deleteShop(Integer shopId, LoginUser loginUser);

/\*\*

\* 更新店铺

\* @param shopUpdate 修改对象

\* @param loginUser 当前用户对象

\* @return 更新数量

\*/

int updateShop(CompanyShopUpdate shopUpdate, LoginUser loginUser);

/\*\*

\* 更新店铺

\* @param companyShop 修改对象

\* @param loginUser 当前用户对象

\* @return 更新数量

\*/

int updateCompanyShop(CompanyShop companyShop, LoginUser loginUser);

/\*\*

\* 通过店铺ID查询店铺信息

\* @param shopId 店铺ID

\* @return CompanyShop对象

\*/

CompanyShop getShopById(Integer shopId);

/\*\*

\* 通过店铺ID查询店铺信息

\* @param shopId 店铺ID

\* @return CompanyShopVO对象

\*/

CompanyShopDetailsVO getShopDetails(Integer shopId);

/\*\*

\* 获取满足条件的数据数量

\* @param query 查询条件对象

\* @return int 满足条件数据数量

\*/

int getCount(CompanyShopQuery query);

/\*\*

\* 获取满足条件的数据列表

\* @param query 查询条件对象

\* @return List<CompanyShop> 集合列表

\*/

List<CompanyShopVO> findList(CompanyShopQuery query);

/\*\*

\* 查询品牌馆信息

\* @param userId 用户ID

\* @param companyId 企业ID

\* @param businessType 店铺类型

\* @return CompanyShop

\*/

CompanyShopDetailsVO findBrandPavilion(Integer userId, Long companyId, Integer businessType);

/\*\*

\* 验证店铺唯一性

\* @param shopName 店铺名称

\* @param shopId 企业ID

\* @return

\*/

Integer checkShopName(String shopName, Long shopId);

/\*\*

\* 处理店铺分类（为厂商、经销商类型）

\* @param companyId

\* @param categoryIds

\* @return

\*/

boolean disposeShopCategory(Long companyId, String categoryIds);

/\*\*

\* 根据企业Id更新店铺分类

\*

\* @param companyShop

\* @return

\*/

int updateShopCategoryByCompanyId(CompanyShop companyShop);

/\*\*

\* 通过用户id查询用户所属店铺列表

\* @param userId 用户Id

\* @return list

\*/

List<CompanyShop> getShopListByUserId(Long userId);

/\*\*

\* 通过用户Id删除属于用户的店铺

\* @param userId 用户Id

\* @return

\*/

Integer deleteShopByUserId(Long userId,String loginUserName);

}

package com.sandu.api.shop.service;

import com.sandu.api.shop.input.ProjectCaseAdd;

import com.sandu.api.shop.input.ProjectCaseQuery;

import com.sandu.api.shop.input.ProjectCaseUpdate;

import com.sandu.api.shop.model.ProjectCase;

import com.sandu.api.shop.output.ProjectCaseDetailsVO;

import com.sandu.api.shop.output.ProjectCaseVO;

import com.sandu.commons.LoginUser;

import org.springframework.stereotype.Component;

import java.util.List;

/\*\*

\* 工程案例-service接口

\*

\* @auth xiaoxc

\* @data 2018-06-04

\*/

@Component

public interface ProjectCaseService {

/\*\*

\* 新增

\* @param caseAdd 新增对象

\* @param loginUser 当前用户对象

\* @return int 新增数据Id

\*/

int add(ProjectCaseAdd caseAdd, LoginUser loginUser);

/\*\*

\* 删除

\* @param id 主键Id

\* @param loginUser 当前用户对象

\* @return 删除数量

\*/

int delete(Integer id, LoginUser loginUser);

/\*\*

\* 根据店铺Id批量删除

\* @param shopId 主键Id

\* @param loginUser 当前用户对象

\* @return 删除数量

\*/

int batchDelByShopId(Integer shopId, LoginUser loginUser);

/\*\*

\* 更新

\* @param caseUpdate 修改对象

\* @param loginUser 当前用户对象

\* @return 更新数量

\*/

int update(ProjectCaseUpdate caseUpdate, LoginUser loginUser);

/\*\*

\* 更新

\* @param projectCase 修改对象

\* @param loginUser 当前用户对象

\* @return 更新数量

\*/

int updateCase(ProjectCase projectCase, LoginUser loginUser);

/\*\*

\* 通过ID查询信息

\* @param id 主键ID

\* @return ProjectCase对象

\*/

ProjectCase findById(Integer id);

/\*\*

\* 通过ID查询信息

\* @param id 主键ID

\* @return vo对象

\*/

ProjectCaseDetailsVO getDetails(Integer id);

/\*\*

\* 获取满足条件的数据数量

\* @param query 查询条件

\* @return int 满足条件数据数量

\*/

int getCount(ProjectCaseQuery query);

/\*\*

\* 获取满足条件的数据列表

\* @param query 查询条件

\* @return 集合列表

\*/

List<ProjectCaseVO> findList(ProjectCaseQuery query);

/\*\*

\* 通过店铺Id删除属于店铺的工程案例

\* @param shopIds 店铺Id

\* @return list

\*/

Integer deleteCaseByShopId(List<Integer> shopIds,String loginUserName);

/\*\*

\* 通过店铺Id查找属于店铺的工程案例数量

\* @param shopIds 店铺Id

\* @return list

\*/

Integer getCaseByShopId(List<Integer> shopIds);

}

package com.sandu.service.shop.dao;

import com.sandu.api.shop.input.CompanyShopArticleQuery;

import com.sandu.api.shop.model.CompanyShopArticle;

import com.sandu.api.shop.output.CompanyShopArticleListVO;

import org.apache.ibatis.annotations.Param;

import org.springframework.stereotype.Repository;

import java.util.List;

/\*\*

\* 店铺博文--dao层接口

\* @author WangHaiLin

\* @date 2018/8/9 11:02

\*/

@Repository

public interface CompanyShopArticleDao {

/\*\*

\* 新增

\* @param article 新增对象

\* @return int 新增结果

\*/

int add(CompanyShopArticle article);

/\*\*

\* 通过Id查询店铺博文

\* @param id 博文Id

\* @return CompanyShopArticle

\*/

CompanyShopArticle get(@Param("id") Long id);

/\*\*

\* 修改店铺博文

\* @param article 博文实体

\* @return int 修改结果

\*/

int update(CompanyShopArticle article);

/\*\*

\* 查询店铺博文列表

\* @param query 查询入参

\* @return list 返回结果

\*/

List<CompanyShopArticle> getList(CompanyShopArticleQuery query);

/\*\*

\* 查询店铺博文数量

\* @param query 查询条件

\* @return int 数量

\*/

int getCount (CompanyShopArticleQuery query);

/\*\*

\* 通过店铺Id进行批量删除

\* @param shopId 店铺Id

\* @param modifier 修改人

\* @return int 操作结果

\*/

int deleteByShopId(@Param("shopId") Long shopId,@Param("modifier") String modifier);

/\*\*

\* add by WangHL

\* 通过店铺Ids删除属于店铺的博文列表

\* @param shopIds 店铺Ids

\* @return list

\*/

Integer deleteArticleByShopId(@Param("shopIds") List<Integer> shopIds,@Param("loginUserName") String loginUserName);

/\*\*

\* add by WangHL

\* 通过店铺Id查找属于店铺的博文数量

\* @param shopIds 店铺Id

\* @return list

\*/

Integer getArticleByShopId(@Param("shopIds")List<Integer> shopIds);

}

package com.sandu.service.shop.dao;

import com.sandu.api.shop.model.CompanyShop;

import com.sandu.api.shop.model.po.ShopPO;

import org.apache.ibatis.annotations.Param;

import org.springframework.stereotype.Repository;

import java.util.List;

/\*\*

\* 店铺管理--接口持久层

\*

\* @auth xiaoxc

\* @data 2018-06-04

\*/

@Repository

public interface CompanyShopDao {

/\*\*

\* 插入店铺

\* @param companyShop 新增对象

\* @return int 新增数量

\*/

int insertShop(CompanyShop companyShop);

/\*\*

\* 删除店铺

\* @param companyShop 删除对象

\* @return 删除数量

\*/

int deleteShop(CompanyShop companyShop);

/\*\*

\* 更新店铺

\* @param companyShop 修改对象

\* @return 更新数量

\*/

int updateShop(CompanyShop companyShop);

/\*\*

\* 通过店铺ID查询店铺信息

\* @param shopId 店铺ID

\* @return CompanyShop对象

\*/

CompanyShop getShopById(Integer shopId);

/\*\*

\* 获取满足条件的数据数量

\* @param query 查询条件对象

\* @return int 满足条件数据数量

\*/

int findCount(ShopPO query);

/\*\*

\* 获取满足条件的数据列表

\* @param query 查询条件对象

\* @return List<CompanyShop> 集合列表

\*/

List<CompanyShop> findList(ShopPO query);

/\*\*

\* 查询品牌馆信息

\* @param userId

\* @param companyId

\* @param businessType

\* @return

\*/

CompanyShop findBrandPavilion(@Param("userId") Integer userId, @Param("companyId") Long companyId, @Param("businessType") Integer businessType);

/\*\*

\* 验证店铺唯一性

\* @param shopName 店铺名称

\* @param shopId 企业ID

\* @return

\*/

Integer checkShopName(@Param("shopName") String shopName, @Param("shopId") Long shopId);

/\*\*

\* 根据企业Id更新店铺分类

\*

\* @param companyShop

\* @return

\*/

int updateShopCategoryByCompanyId(CompanyShop companyShop);

/\*\*

\* add by WangHL

\* 通过用户ID查询属于用户的店铺列表

\* @param userId 用户Id

\* @return list

\*/

List<CompanyShop> getShopListByUserId(@Param("userId") Long userId);

/\*\*

\* add by WangHL

\* 通过用户Id删除属于用户的店铺

\* @param userId 用户Id

\* @return

\*/

Integer deleteShopByUserId(@Param("userId") Long userId,@Param("loginUserName") String loginUserName);

}

package com.sandu.service.shop.dao;

import org.apache.ibatis.annotations.Param;

import org.springframework.stereotype.Repository;

import java.util.List;

/\*\*

\* @author WangHaiLin

\* @date 2018/8/30 17:03

\*/

@Repository

public interface CompanyShopDesignPlanDao {

/\*\*

\* 通过店铺Id删除属于店铺的方案列表

\* @param shopIds 店铺Id

\* @return list 方案列表

\*/

Integer deleteDesignPlanByShopId(@Param("shopIds")List<Integer> shopIds);

/\*\*

\* 通过店铺Id查找属于店铺的方案数量

\* @param shopIds 店铺Id

\* @return list 方案列表

\*/

Integer getDesignPlanByShopId(@Param("shopIds")List<Integer> shopIds);

}

package com.sandu.service.shop.dao;

import com.sandu.api.shop.input.ProjectCaseQuery;

import com.sandu.api.shop.model.ProjectCase;

import com.sandu.api.shop.output.ProjectCaseVO;

import org.apache.ibatis.annotations.Param;

import org.springframework.stereotype.Repository;

import java.util.List;

/\*\*

\* 工程案例--接口持久层

\*

\* @auth xiaoxc

\* @data 2018-06-19

\*/

@Repository

public interface ProjectCaseDao {

/\*\*

\* 新增

\* @param projectCase 新增对象

\* @return int 新增数据Id

\*/

int insert(ProjectCase projectCase);

/\*\*

\* 删除

\* @param projectCase 主键Id

\* @return 删除数量

\*/

int delete(ProjectCase projectCase);

/\*\*

\* 根据店铺ID批量删除

\* @param projectCase 主键Id

\* @return 删除数量

\*/

int batchDelByShopId(ProjectCase projectCase);

/\*\*

\* 更新

\* @param projectCase 修改对象

\* @return 更新数量

\*/

int update(ProjectCase projectCase);

/\*\*

\* 通过ID查询信息

\* @param id 主键ID

\* @return ProjectCase对象

\*/

ProjectCase findById(Integer id);

/\*\*

\* 获取满足条件的数据数量

\* @param query 查询条件店

\* @return int 满足条件数据数量

\*/

int findCount(ProjectCaseQuery query);

/\*\*

\* 获取满足条件的数据列表

\* @param query 查询条件

\* @return 集合列表

\*/

List<ProjectCaseVO> findList(ProjectCaseQuery query);

/\*\*

\* add by WangHL

\* 通过店铺Ids删除属于店铺的工程案例

\* @param shopIds 店铺Id

\* @return list

\*/

Integer deleteCaseByShopId(@Param("shopIds")List<Integer> shopIds,@Param("loginUserName")String loginUserName);

/\*\*

\* 通过店铺Id查找属于店铺的工程案例数量

\* @param shopIds 店铺Id

\* @return list

\*/

Integer getCaseByShopId(@Param("shopIds")List<Integer> shopIds);

}

package com.sandu.service.shop.impl.biz;

import com.sandu.api.shop.model.CompanyShop;

import com.sandu.api.shop.service.\*;

import com.sandu.api.shop.service.biz.ShopBizService;

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 java.util.ArrayList;

import java.util.List;

/\*\*

\* @author WangHaiLin

\* @date 2018/8/31 9:47

\*/

@Service("shopBizService")

public class ShopBizServiceImpl implements ShopBizService {

private Logger logger = LoggerFactory.getLogger(ShopBizServiceImpl.class);

@Autowired

private CompanyShopService companyShopService;

@Autowired

private CompanyShopArticleService companyShopArticleService;

@Autowired

private CompanyShopDesignPlanService designPlanService;

@Autowired

private ProjectCaseService projectCaseService;

@Override

@Transactional

public int delUsersShop(Long userId,String loginUserName) {

//1.通过用户Id查找属于用户的店铺集合

List<CompanyShop> shopList = companyShopService.getShopListByUserId(userId);

//2.用户店铺集合不为空，取出店铺id,放入list中

List<Integer> shopIds=new ArrayList<>();

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

for (CompanyShop shop:shopList) {

shopIds.add(shop.getId());

}

//查看要删除的店铺是否存在工程案例

Integer count = projectCaseService.getCaseByShopId(shopIds);

logger.info("待删除店铺工程案例总数量：{}",count);

//存在，通过店铺Id删店铺发布的工程案例

if (count>0){

Integer result = projectCaseService.deleteCaseByShopId(shopIds, loginUserName);

logger.info("工程案例删除结果：{}",result);

}

//查看要删除的店铺是否存在博文

Integer count2 = companyShopArticleService.getArticleByShopId(shopIds);

logger.info("待删除店铺博文总数量：{}",count2);

//存在，通过店铺Id删店铺发布的博文

if (count2>0){

Integer result =companyShopArticleService.deleteArticleByShopId(shopIds,loginUserName);

logger.info("博文删除结果：{}",result);

}

//查看要删除的店铺是发布方案

Integer count3 = designPlanService.getDesignPlanByShopId(shopIds);

logger.info("待删除店铺方案总数量：{}",count3);

//存在，通过店铺Id删店铺发布的方案

if (count3>0){

Integer result =designPlanService.deleteDesignPlanByShopId(shopIds);

logger.info("方案删除结果：{}",result);

}

//通过用户Id删用户创建的店铺

return companyShopService.deleteShopByUserId(userId,loginUserName);

}

return -1;

}

}

package com.sandu.service.shop.impl;

import com.sandu.api.shop.model.CompanyShopDesignPlan;

import com.sandu.api.shop.service.CompanyShopDesignPlanService;

import com.sandu.service.shop.dao.CompanyShopDesignPlanDao;

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

import org.springframework.stereotype.Service;

import java.util.List;

/\*\*

\* @author WangHaiLin

\* @date 2018/8/30 17:01

\*/

@Service("companyShopDesignPlanService")

public class CompanyShopDesignPlanServiceImpl implements CompanyShopDesignPlanService {

@Autowired

private CompanyShopDesignPlanDao companyShopDesignPlanDao;

@Override

public Integer deleteDesignPlanByShopId(List<Integer> shopIds) {

return companyShopDesignPlanDao.deleteDesignPlanByShopId(shopIds);

}

@Override

public Integer getDesignPlanByShopId(List<Integer> shopIds) {

return companyShopDesignPlanDao.getDesignPlanByShopId(shopIds);

}

}

package com.sandu.service.shop.impl;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Date;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import java.util.stream.Collectors;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import com.sandu.api.area.service.BaseAreaService;

import com.sandu.api.category.service.ProCategoryService;

import com.sandu.api.company.common.StringUtil;

import com.sandu.api.company.model.BaseCompany;

import com.sandu.api.company.service.BaseCompanyService;

import com.sandu.api.dictionary.service.DictionaryService;

import com.sandu.api.file.model.ResFile;

import com.sandu.api.file.service.ResFileService;

import com.sandu.api.pic.model.ResPic;

import com.sandu.api.pic.model.po.ResPicBusinessUpdate;

import com.sandu.api.pic.model.po.ResPicPO;

import com.sandu.api.pic.service.ResPicService;

import com.sandu.api.shop.common.constant.ShopCodePrefixEnum;

import com.sandu.api.shop.common.constant.ShopCoverResTypeConstant;

import com.sandu.api.shop.common.constant.ShopResTypeConstant;

import com.sandu.api.shop.input.CompanyShopAdd;

import com.sandu.api.shop.input.CompanyShopArticleQuery;

import com.sandu.api.shop.input.CompanyShopQuery;

import com.sandu.api.shop.input.CompanyShopUpdate;

import com.sandu.api.shop.input.ProjectCaseQuery;

import com.sandu.api.shop.model.CompanyShop;

import com.sandu.api.shop.model.po.ShopPO;

import com.sandu.api.shop.output.CompanyShopDetailsVO;

import com.sandu.api.shop.output.CompanyShopVO;

import com.sandu.api.shop.service.CompanyShopArticleService;

import com.sandu.api.shop.service.CompanyShopDesignPlanService;

import com.sandu.api.shop.service.CompanyShopService;

import com.sandu.api.shop.service.ProjectCaseService;

import com.sandu.commons.LoginUser;

import com.sandu.commons.Utils;

import com.sandu.commons.constant.BusinessTypeConstant;

import com.sandu.commons.constant.SysDictionaryConstant;

import com.sandu.commons.util.StringUtils;

import com.sandu.constant.Constants;

import com.sandu.queue.SyncMessage;

import com.sandu.queue.service.QueueService;

import com.sandu.service.shop.dao.CompanyShopDao;

/\*\*

\* 企业店铺service实现类

\*

\* @auth xiaoxc

\* @data 2018-06-04

\*/

@Service("companyShopService")

public class CompanyShopServiceImpl implements CompanyShopService {

private Logger logger = LoggerFactory.getLogger(CompanyShopServiceImpl.class);

@Value("${upload.base.path}")

private String rootPath;

@Value("${company.shop.introducedFile.upload.path}")

private String introducedUpdateFilePath;

@Autowired

private CompanyShopDao companyShopDao;

@Autowired

private BaseCompanyService baseCompanyService;

@Autowired

private ProCategoryService proCategoryService;

@Autowired

private ResPicService resPicService;

@Autowired

private BaseAreaService baseAreaService;

@Autowired

private DictionaryService dictionaryService;

@Autowired

private ResFileService resFileService;

@Autowired

private ProjectCaseService projectCaseService;

@Autowired

private CompanyShopArticleService articleService;

@Autowired

private CompanyShopDesignPlanService designPlanService;

@Autowired

private QueueService queueService;

@Override

public int addShop(CompanyShopAdd shopAdd, LoginUser loginUser) {

CompanyShop companyShop = shopAdd.getCompanyShop();

// 设置区域长编码

companyShop.setLongAreaCode(this.getLongAreaCode(companyShop));

// 设置用户Id

companyShop.setUserId(loginUser.getId());

// 获取企业信息

BaseCompany baseCompany = null;

// 获取登录用户企业id

Long franchiserId = shopAdd.getCompanyId();

if (franchiserId != null && franchiserId > 0) {

// 设置企业Id

companyShop.setCompanyId(franchiserId.intValue());

baseCompany = baseCompanyService.getCompanyById(franchiserId);

// 设置店铺编码、分类、类型

setShopInfo(companyShop, baseCompany);

// 经销商冗余厂商Id

if (BusinessTypeConstant.BUSINESSTYPE\_FRANCHISER == baseCompany.getBusinessType().intValue()) {

companyShop.setCompanyPid(baseCompany.getPid());

} else {

companyShop.setCompanyPid(franchiserId.intValue());

}

} else {

logger.error("addShop: userId={},companyId={}", loginUser.getId() , franchiserId);

return 0;

}

// 存储店铺介绍富文本文件

String content = companyShop.getShopIntroduced();

if (StringUtils.isNotEmpty(content)) {

Integer resId = resFileService.saveFile(introducedUpdateFilePath, content, loginUser, null,

ShopResTypeConstant.SHOP\_RES\_INTRODUCED\_FILE\_KEY, ShopResTypeConstant.SHOP\_RES\_INTRODUCED\_FILE\_TYPE);

if (resId == null || resId == 0) {

logger.error("CompanyShopServiceImpl.addShop{}:保存配置文件失败");

return 0;

} else {

companyShop.setIntroducedFileId(resId.longValue());

}

}

// 设置系统字段值

saveSystemInfo(companyShop, loginUser);

// 插入数据库

int addResult = companyShopDao.insertShop(companyShop);

if (addResult > 0) {

// 回填资源业务ID

boolean falg = backfill(companyShop);

if (falg) {

return companyShop.getId();

}

}

return 0;

}

@Override

@Transactional

public int deleteShop(Integer shopId, LoginUser loginUser) {

CompanyShop companyShop = getShopById(shopId);

if (companyShop == null) {

return 0;

}

// 删除对应的工程案例

ProjectCaseQuery query = new ProjectCaseQuery();

query.setShopId(companyShop.getId());

int total = projectCaseService.getCount(query);

if (total > 0) {

projectCaseService.batchDelByShopId(companyShop.getId(), loginUser);

}

//删除对应店铺博文 add by WangHaiLin

CompanyShopArticleQuery articleQuery=new CompanyShopArticleQuery();

articleQuery.setShopId(Long.valueOf(companyShop.getId()));

Integer count = articleService.getCount(articleQuery);

if (count>0){

articleService.deleteByShopId(Long.valueOf(companyShop.getId()),loginUser);

}

//删除店铺发布的方案 add by WangHaiLin

List<Integer> shopIds=new ArrayList<>();

shopIds.add(companyShop.getId());

Integer count2 = designPlanService.getDesignPlanByShopId(shopIds);

if (count2>0){

designPlanService.deleteDesignPlanByShopId(shopIds);

}

// 存储系统字段值

saveSystemInfo(companyShop, loginUser);

int result = companyShopDao.deleteShop(companyShop);

if(result > 0) {

logger.info("deleteShop发送消息开始:id:{},时间:{}",shopId,new Date());

try {

sycMessageDoSend(SyncMessage.ACTION\_DELETE, Arrays.asList(shopId));

}catch(Exception e) {

logger.error("deleteShop发送消息失败,id:{},错误信息:{}",shopId,e.getMessage());

}

logger.info("deleteShop发送消息结束:id:{},时间:{}",shopId,new Date());

}

return result;

}

@Override

public int updateShop(CompanyShopUpdate shopUpdate, LoginUser loginUser) {

// 未修改之前的值

CompanyShop oldCompanyShop = getShopById(shopUpdate.getShopId());

// 修改的值

CompanyShop companyShop = shopUpdate.getCompanyShopFromCompanyShopUpdate(oldCompanyShop, shopUpdate);

// 更新店铺介绍富文本内容

Long oldResId = oldCompanyShop.getIntroducedFileId();

String content = companyShop.getShopIntroduced();

if (oldResId == null || oldResId.intValue() == 0) {

if (StringUtils.isNotEmpty(content)) {

Integer resId = resFileService.saveFile(introducedUpdateFilePath, content, loginUser, companyShop.getId(),

ShopResTypeConstant.SHOP\_RES\_INTRODUCED\_FILE\_KEY, ShopResTypeConstant.SHOP\_RES\_INTRODUCED\_FILE\_TYPE);

if (resId == null || resId == 0) {

logger.error("CompanyShopServiceImpl.updateShop{}:保存配置文件失败");

return 0;

} else {

companyShop.setIntroducedFileId(resId.longValue());

}

}

} else {

boolean falg = resFileService.updateFile(oldResId.longValue(), content);

if (!falg) {

logger.error("CompanyShopServiceImpl.resFileService.updateFile{}:更新配置文件失败");

return 0;

}

}

//存储系统字段值

saveSystemInfo(companyShop, loginUser);

int result = companyShopDao.updateShop(companyShop);

if(result > 0) {

String oldPlatFormValues = oldCompanyShop.getReleasePlatformValues();

String newPlatFormValues = shopUpdate.getPlatformValues();

//发布平台有变动再推送消息

boolean flag = checkPlatformChange(oldPlatFormValues,newPlatFormValues);

logger.info("oldPlatFormValues:{},newPlatFormValues:{},flag:{}",oldPlatFormValues,newPlatFormValues,flag);

if(flag) {

logger.info("updateShop发送消息开始:id:{},时间:{}",shopUpdate.getShopId(),new Date());

try {

sycMessageDoSend(SyncMessage.ACTION\_UPDATE, Arrays.asList(shopUpdate.getShopId()));

}catch(Exception e) {

logger.error("updateShop发送消息失败,id:{},错误信息:{}",shopUpdate.getShopId(),e.getMessage());

}

logger.info("updateShop发送消息结束:id:{},时间:{}",shopUpdate.getShopId(),new Date());

}

}

return result;

}

/\*\*

\* 校验发布平台是否有变动

\* @param oldPlatFormValues

\* @param newPlatFormValues

\* @return

\*/

private static boolean checkPlatformChange(String oldPlatFormValues, String newPlatFormValues) {

if(StringUtils.isEmpty(oldPlatFormValues)) {

if(StringUtils.isEmpty(newPlatFormValues)) {

return false;

}else {

return true;

}

}else {

if(StringUtils.isEmpty(newPlatFormValues)) {

return true;

}

List<String> oldPlatformList = new ArrayList<String>(Arrays.asList(oldPlatFormValues.split(",")));

List<String> newPlatformList = new ArrayList<String>(Arrays.asList(newPlatFormValues.split(",")));

if(oldPlatformList.size()!=newPlatformList.size()) {

return true;

}

//判断是否存在差集

oldPlatformList.removeAll(newPlatformList);

if(oldPlatformList.size()> 0) {

return true;

}else {

return false;

}

}

}

@Override

public int updateCompanyShop(CompanyShop companyShop, LoginUser loginUser) {

companyShop.setGmtModified(new Date());

companyShop.setModifier(loginUser.getLoginName());

return companyShopDao.updateShop(companyShop);

}

@Override

public CompanyShop getShopById(Integer shopId) {

return companyShopDao.getShopById(shopId);

}

@Override

public int getCount(CompanyShopQuery query) {

ShopPO shopPO = query.getShopPOFromShopQuery(query);

return companyShopDao.findCount(shopPO);

}

@Override

public List<CompanyShopVO> findList(CompanyShopQuery query) {

List<CompanyShopVO> shopVOList = new ArrayList<>();

ShopPO shopPO = query.getShopPOFromShopQuery(query);

List<CompanyShop> shopList = companyShopDao.findList(shopPO);

if (shopList != null && 0 < shopList.size()) {

shopList.forEach(companyShop -> {

CompanyShopVO shopVO = new CompanyShopVO().getCompanyShopVOFromCompanyShop(companyShop);

// 获取区域名称

shopVO.setAreaName(getAreaNames(companyShop, null));

// 获取平台名称

shopVO.setReleasePlatformName(getPlatfromTypeNames(companyShop));

// 存入列表

shopVOList.add(shopVO);

});

}

return shopVOList;

}

@Override

public CompanyShopDetailsVO getShopDetails(Integer shopId) {

CompanyShop companyShop = getShopById(shopId);

CompanyShopDetailsVO detailsVO = new CompanyShopDetailsVO().getShopDetailsVOFromShop(companyShop);

if (detailsVO != null && detailsVO.getShopId() != null) {

// 获取区域名称

getAreaNames(companyShop, detailsVO);

// 获取店铺分类(数据字典擅长风格、施工类型名称)

getCategoryNames(detailsVO);

// 获取资源相对路径

getResPath(detailsVO);

}

return detailsVO;

}

@Override

public CompanyShopDetailsVO findBrandPavilion(Integer userId, Long companyId, Integer businessType) {

CompanyShop companyShop = companyShopDao.findBrandPavilion(userId, companyId, businessType);

CompanyShopDetailsVO detailsVO = new CompanyShopDetailsVO().getShopDetailsVOFromShop(companyShop);

if (detailsVO != null && detailsVO.getShopId() != null) {

// 获取区域名称

getAreaNames(companyShop, detailsVO);

// 获取店铺分类(数据字典擅长风格、施工类型名称)

//getCategoryNames(detailsVO); //品牌馆不需要

// 获取资源相对路径

getResPath(detailsVO);

}

return detailsVO;

}

@Override

public Integer checkShopName(String shopName, Long shopId) {

return companyShopDao.checkShopName(shopName, shopId);

}

@Override

public boolean disposeShopCategory(Long companyId, String categoryIds) {

CompanyShop companyShop = new CompanyShop();

companyShop.setCompanyId(companyId.intValue());

companyShop.setGmtModified(new Date());

if (StringUtils.isNotEmpty(categoryIds)) {

// 转换为二级分类Id

companyShop.setCategoryIds(getCategoryIds(categoryIds));

// 冗余产品一级分类Id

companyShop.setFirstCategoryIds(getCategoryIds(companyShop.getCategoryIds()));

} else {

companyShop.setCategoryIds("");

}

try {

this.updateShopCategoryByCompanyId(companyShop);

} catch (Exception e) {

logger.error("updateCategoryRelData{} e " + e.getMessage());

return false;

}

return true;

}

@Override

public int updateShopCategoryByCompanyId(CompanyShop companyShop) {

return companyShopDao.updateShopCategoryByCompanyId(companyShop);

}

@Override

public List<CompanyShop> getShopListByUserId(Long userId) {

return companyShopDao.getShopListByUserId(userId);

}

@Override

public Integer deleteShopByUserId(Long userId,String loginUserName) {

return companyShopDao.deleteShopByUserId(userId,loginUserName);

}

/\*\*

\* 自动存储系统字段

\*

\* @param companyShop 店铺model对象

\* @param loginUser 当前用户对象

\*/

@SuppressWarnings("all")

private void saveSystemInfo(CompanyShop companyShop, LoginUser loginUser) {

if (companyShop != null) {

//新增

if (companyShop.getId() == null) {

companyShop.setGmtCreate(new Date());

companyShop.setCreator(loginUser.getLoginName());

companyShop.setIsDeleted(0);

if (companyShop.getSysCode() == null || "".equals(companyShop.getSysCode())) {

companyShop.setSysCode(Utils.getCurrentDateTime(Utils.DATETIMESSS) + "\_" + Utils.generateRandomDigitString(6));

}

}

companyShop.setGmtModified(new Date());

companyShop.setModifier(loginUser.getLoginName());

}

}

/\*\*

\* 设置区域长编码

\*

\* @param companyShop

\* @return

\*/

private String getLongAreaCode(CompanyShop companyShop) {

StringBuilder longAreaCodeBuilder = new StringBuilder("");

if (StringUtils.isNotEmpty(companyShop.getProvinceCode())) {

longAreaCodeBuilder.append(".").append(companyShop.getProvinceCode());

}

if (StringUtils.isNotEmpty(companyShop.getCityCode())) {

longAreaCodeBuilder.append(".").append(companyShop.getCityCode());

}

if (StringUtils.isNotEmpty(companyShop.getAreaCode())) {

longAreaCodeBuilder.append(".").append(companyShop.getAreaCode());

}

if (StringUtils.isNotEmpty(companyShop.getStreetCode())) {

longAreaCodeBuilder.append(".").append(companyShop.getStreetCode());

}

if (longAreaCodeBuilder != null && 0 < longAreaCodeBuilder.length()) {

longAreaCodeBuilder.append(".");

} else {

return "";

}

return longAreaCodeBuilder.toString();

}

/\*\*

\* 1、设置店铺编码(编码规则标识+7位随机数)

\* 2、根据企业类型获取店铺类型

\* 3、根据企业类型获取店铺产品分类

\*

\* @param companyShop

\* @param baseCompany

\*/

private void setShopInfo(CompanyShop companyShop, BaseCompany baseCompany) {

switch (baseCompany.getBusinessType()) {

case BusinessTypeConstant.BUSINESSTYPE\_COMPANY:

// 店铺编码

companyShop.setShopCode(ShopCodePrefixEnum.BRAND\_PAVILION.getCode() + Utils.generateRandomDigitString(7));

// 企业厂商---店铺品牌馆

companyShop.setBusinessType(SysDictionaryConstant.SHOP\_TYPE\_BRAND\_PAVILION\_VALUE);

// 通过企业可见范围产品分类获取父级分类(二级分类)

companyShop.setCategoryIds(getCategoryIds(baseCompany.getProductVisibilityRange()));

// 冗余产品分类一级分类

companyShop.setFirstCategoryIds(getCategoryIds(companyShop.getCategoryIds()));

break;

case BusinessTypeConstant.BUSINESSTYPE\_FRANCHISER:

// 店铺编码

companyShop.setShopCode(ShopCodePrefixEnum.BUILDING\_MATERIALS\_HOME.getCode() + Utils.generateRandomDigitString(7));

// 经销商---家居建材

companyShop.setBusinessType(SysDictionaryConstant.SHOP\_TYPE\_FURNITURE\_VALUE);

// 通过企业可见范围产品分类获取父级分类(二级分类)

companyShop.setCategoryIds(getCategoryIds(baseCompany.getProductVisibilityRange()));

// 冗余产品分类一级分类

companyShop.setFirstCategoryIds(getCategoryIds(companyShop.getCategoryIds()));

break;

case BusinessTypeConstant.BUSINESSTYPE\_DESIGN:

// 店铺编码

companyShop.setShopCode(ShopCodePrefixEnum.DESIGN\_COMPANY.getCode() + Utils.generateRandomDigitString(7));

// 设计公司---设计公司

companyShop.setBusinessType(SysDictionaryConstant.SHOP\_TYPE\_DESIGNER\_COMPANY\_VALUE);

break;

case BusinessTypeConstant.BUSINESSTYPE\_DECORATION:

// 店铺编码

companyShop.setShopCode(ShopCodePrefixEnum.DECORATION\_COMPANY.getCode() + Utils.generateRandomDigitString(7));

// 装修公司---装修公司

companyShop.setBusinessType(SysDictionaryConstant.SHOP\_TYPE\_DECORATION\_COMPANY\_VALUE);

break;

case BusinessTypeConstant.BUSINESSTYPE\_DESIGNER:

// 店铺编码

companyShop.setShopCode(ShopCodePrefixEnum.DESIGNER.getCode() + Utils.generateRandomDigitString(7));

// 设计师---设计师

companyShop.setBusinessType(SysDictionaryConstant.SHOP\_TYPE\_PROFESSION\_DESIGNER\_VALUE);

break;

case BusinessTypeConstant.BUSINESSTYPE\_FOREMAN:

// 店铺编码

companyShop.setShopCode(ShopCodePrefixEnum.CONSTRUCTION\_UNIT.getCode() + Utils.generateRandomDigitString(7));

// 工长(施工单位)---工长(施工单位)

companyShop.setBusinessType(SysDictionaryConstant.SHOP\_TYPE\_CONSTRUCTION\_UNIT\_VALUE);

break;

default:

break;

}

}

/\*\*

\* 通过企业可见范围产品分类获取父级

\*

\* @param categoryIds

\* @return

\*/

private String getCategoryIds(String categoryIds) {

StringBuilder builder = new StringBuilder("");

if (StringUtils.isNotEmpty(categoryIds)) {

List<Integer> idList = StringUtil.getListByString(categoryIds);

if (idList != null && 0 < idList.size()) {

List<Integer> parentIds = proCategoryService.getParentId(idList);

if (parentIds != null && 0 < parentIds.size()) {

for (Integer pid : parentIds) {

builder.append(pid).append(",");

}

}

}

}

if (builder.length() > 0) {

return builder.substring(0, builder.length() - 1).toString();

} else {

return "";

}

}

/\*\*

\* 回填资源业务ID

\*

\* @param companyShop

\* @return

\*/

@SuppressWarnings("all")

private boolean backfill(CompanyShop companyShop) {

Integer logoId = companyShop.getLogoPicId();

String coverIds = companyShop.getCoverResIds();

Long introducedFileId = companyShop.getIntroducedFileId();

String introducedPicIds = companyShop.getIntroducedPicIds();

//logo

if (logoId != null && logoId > 0) {

ResPic resPic = new ResPic();

resPic.setId(logoId.longValue());

resPic.setBusinessId(companyShop.getId());

try {

resPicService.updateByPrimaryKeySelective(resPic);

} catch (Exception e) {

logger.error("backfill{}: shopId="+companyShop.getId()+";logoId="+logoId);

return false;

}

}

//封面

if (StringUtils.isNotEmpty(coverIds)) {

switch (companyShop.getCoverResType()) {

//图片列表

case ShopCoverResTypeConstant.COVER\_PIC\_LIST\_TYPE:

List<Integer> idList = Utils.getIntegerListFromStringList(coverIds);

if (idList != null && 0 < idList.size()) {

ResPicBusinessUpdate resPic = new ResPicBusinessUpdate();

resPic.setIdList(idList);

resPic.setBusinessId(companyShop.getId());

try {

resPicService.batchUpdateBusinessId(resPic);

} catch (Exception e) {

logger.error("backfill{}: shopId="+companyShop.getId()+";coverIds="+coverIds);

return false;

}

}

break;

//全景图

case ShopCoverResTypeConstant.COVER\_PANORAMA\_TYPE:

ResPic resPic = new ResPic();

resPic.setId(logoId.longValue());

resPic.setBusinessId(companyShop.getId());

try {

resPicService.updateByPrimaryKeySelective(resPic);

} catch (Exception e) {

logger.error("backfill{}: shopId="+companyShop.getId()+";logoId="+logoId);

return false;

}

break;

//视频

case ShopCoverResTypeConstant.COVER\_VIDEO\_TYPE:

break;

default:

break;

}

}

//富文本文件

if (introducedFileId != null && introducedFileId.intValue() > 0) {

ResFile resFile = new ResFile();

resFile.setId(introducedFileId.intValue());

resFile.setBusinessId(companyShop.getId());

try {

resFileService.update(resFile);

} catch (Exception e) {

logger.error("backfill{}: shopId="+companyShop.getId()+";fileId="+introducedFileId);

return false;

}

}

//富文本图片ids

if (StringUtils.isNotEmpty(introducedPicIds)) {

List<Integer> idList = Utils.getIntegerListFromStringList(introducedPicIds);

if (idList != null && 0 < idList.size()) {

ResPicBusinessUpdate resPic = new ResPicBusinessUpdate();

resPic.setIdList(idList);

resPic.setBusinessId(companyShop.getId());

try {

resPicService.batchUpdateBusinessId(resPic);

} catch (Exception e) {

logger.error("backfill{}: shopId="+companyShop.getId()+";picIds="+introducedPicIds);

return false;

}

}

}

return true;

}

/\*\*

\* 获取当前企业 行政区域 名称

\*

\* @param detailsVO

\*/

private String getAreaNames(CompanyShop companyShop, CompanyShopDetailsVO detailsVO) {

if (companyShop == null) {

return "";

}

List<String> codeList = new ArrayList<>();

codeList.add(companyShop.getProvinceCode());

codeList.add(companyShop.getCityCode());

codeList.add(companyShop.getAreaCode());

codeList.add(companyShop.getStreetCode());

String areaNames = baseAreaService.getAreaNames(codeList);

if (detailsVO == null) {

return areaNames;

}

String[] codeNames = StringUtils.isNotEmpty(areaNames) ? areaNames.split(",") : new String[4];

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

switch (i) {

case 0:

detailsVO.setProvinceName(codeNames[0]);

break;

case 1:

detailsVO.setCityName(codeNames[1]);

break;

case 2:

detailsVO.setAreaName(codeNames[2]);

break;

case 3:

detailsVO.setStreetName(codeNames[3]);

break;

default:

break;

}

}

return areaNames;

}

/\*\*

\* 通过type、value查询数据字典名称

\*

\* @param detailsVO

\*/

private void getCategoryNames(CompanyShopDetailsVO detailsVO) {

if (detailsVO == null) {

return;

}

Integer businessType = detailsVO.getBusinessTypeValue();

String categoryIds = detailsVO.getCategoryIds();

// 设计师、设计师公司、装修公司:擅长风格

if (SysDictionaryConstant.SHOP\_TYPE\_PROFESSION\_DESIGNER\_VALUE == businessType

|| SysDictionaryConstant.SHOP\_TYPE\_DESIGNER\_COMPANY\_VALUE == businessType

|| SysDictionaryConstant.SHOP\_TYPE\_DECORATION\_COMPANY\_VALUE == businessType) {

if (StringUtils.isNotEmpty(categoryIds)) {

List<Integer> dicValueList = StringUtil.getListByString(categoryIds);

if (dicValueList != null && 0 <dicValueList.size()) {

String dicNames = dictionaryService.getNameByTypeOrValues(SysDictionaryConstant.DIRECTION\_GOOD\_STYLE, dicValueList);

detailsVO.setCategoryName(dicNames);

}

}

}

// 工长（施工单位）：施工类型

if (SysDictionaryConstant.SHOP\_TYPE\_CONSTRUCTION\_UNIT\_VALUE == businessType) {

if (StringUtils.isNotEmpty(categoryIds)) {

List<Integer> dicValueList = StringUtil.getListByString(categoryIds);

if (dicValueList != null && 0 <dicValueList.size()) {

String dicNames = dictionaryService.getNameByTypeOrValues(SysDictionaryConstant.DIRECTION\_CONSTRUCTION\_TYPE, dicValueList);

detailsVO.setCategoryName(dicNames);

}

}

}

}

/\*\*

\* 获取店铺发布平台的名称

\* @param companyShop

\* @return

\*/

private String getPlatfromTypeNames(CompanyShop companyShop) {

if (companyShop == null) {

return "";

}

String platfromValues = companyShop.getReleasePlatformValues();

if (StringUtils.isNotEmpty(platfromValues)) {

List<Integer> dicValueList = StringUtil.getListByString(platfromValues);

if (dicValueList != null && 0 < dicValueList.size()) {

return dictionaryService.getNameByTypeOrValues(SysDictionaryConstant.DIRECTION\_SHOP\_RELEASE\_PLATFORM, dicValueList);

}

}

return "";

}

/\*\*

\* 获取资源相对路径

\* @param detailsVO

\*/

private void getResPath(CompanyShopDetailsVO detailsVO) {

if (detailsVO == null) {

return;

}

Integer logoId = detailsVO.getLogoId();

String coverIds = detailsVO.getCoverResIds();

Long fileId = detailsVO.getIntroducedFileId();

//logo资源地址

if (logoId != null && 0 < logoId.intValue()) {

ResPic resPic = resPicService.selectByPrimaryKey(logoId.longValue());

detailsVO.setLogoUrl(resPic != null ? resPic.getPicPath() : "");

}

//封面资源地址

if (StringUtils.isNotEmpty(coverIds)) {

//图片列表和全景图

if (ShopCoverResTypeConstant.COVER\_PIC\_LIST\_TYPE == detailsVO.getCoverResType()

|| ShopCoverResTypeConstant.COVER\_PANORAMA\_TYPE == detailsVO.getCoverResType()) {

List<Integer> idList = Utils.getIntegerListFromStringList(coverIds);

if (idList != null && 0 < idList.size()) {

List<ResPicPO> resPicPOList = resPicService.findByIds(idList);

if (resPicPOList != null && 0 < resPicPOList.size()) {

detailsVO.setCoverList(resPicPOList);

}

}

}

//视频

if (ShopCoverResTypeConstant.COVER\_VIDEO\_TYPE == detailsVO.getCoverResType()) {

}

}

//店铺介绍富文本地址

if (fileId != null && 0 < fileId) {

ResFile resFile = resFileService.getById(fileId);

// detailsVO.setIntroducedFilePath(resFile != null ? resFile.getFilePath() : "");

String filePath = resFile != null ? resFile.getFilePath() : "";

if (StringUtils.isNotEmpty(filePath)) {

// 读取配置

String fileContext = Utils.getFileContext(rootPath + filePath);

detailsVO.setShopIntroduced(fileContext);

}

}

}

/\*\*

\* 发送异步消息

\* @param messageAction

\* @param ids

\*/

private void sycMessageDoSend(Integer messageAction, List<Integer> ids) {

List<Map> content = ids.stream().map(item -> {

HashMap<String, Integer> tmp = new HashMap<>(1);

tmp.put("id", item);

return tmp;

}).collect(Collectors.toList());

SyncMessage message = new SyncMessage();

message.setAction(messageAction);

message.setMessageId("S-" + System.currentTimeMillis());

message.setModule(SyncMessage.SHOP\_MODULE);

message.setPlatformType(Constants.PLATFORM\_CODE\_MERCHANT\_MANAGE);

message.setObject(content);

queueService.sendSyncSearch(message);

}

public static void main(String[] args) {

String oldPlatformvalues = "0,3";

String newPlatformvalues = "2,3";

boolean flag = checkPlatformChange(oldPlatformvalues,newPlatformvalues);

System.out.println(flag);

}

}

package com.sandu.service.shop.impl;

import com.sandu.api.file.model.ResFile;

import com.sandu.api.file.service.ResFileService;

import com.sandu.api.pic.model.ResPic;

import com.sandu.api.pic.model.po.ResPicBusinessUpdate;

import com.sandu.api.pic.service.ResPicService;

import com.sandu.api.shop.common.constant.ShopResTypeConstant;

import com.sandu.api.shop.input.ProjectCaseAdd;

import com.sandu.api.shop.input.ProjectCaseQuery;

import com.sandu.api.shop.input.ProjectCaseUpdate;

import com.sandu.api.shop.model.ProjectCase;

import com.sandu.api.shop.output.ProjectCaseDetailsVO;

import com.sandu.api.shop.output.ProjectCaseVO;

import com.sandu.api.shop.service.ProjectCaseService;

import com.sandu.commons.LoginUser;

import com.sandu.commons.Utils;

import com.sandu.commons.util.StringUtils;

import com.sandu.service.shop.dao.ProjectCaseDao;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

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

import org.springframework.stereotype.Service;

import java.math.BigInteger;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

/\*\*

\* 工程案例实现

\*

\* @auth xiaoxc

\* @data 2018-06-19

\*/

@Service("projectCaseService")

public class ProjectCaseServiceImpl implements ProjectCaseService {

private Logger logger = LoggerFactory.getLogger(ProjectCaseServiceImpl.class);

@Value("${upload.base.path}")

private String rootPath;

@Value("${company.shop.caseFile.upload.path}")

private String caseUpdateFilePath;

@Autowired

private ProjectCaseDao projectCaseDao;

@Autowired

private ResFileService resFileService;

@Autowired

private ResPicService resPicService;

@Override

public int add(ProjectCaseAdd caseAdd, LoginUser loginUser) {

ProjectCase projectCase = caseAdd.getProjectCaseFromProjectCaseAdd(caseAdd);

String content = projectCase.getContent();

// 存储富文本文件

if (StringUtils.isNotEmpty(content)) {

Integer resId = resFileService.saveFile(caseUpdateFilePath, content, loginUser, null,

ShopResTypeConstant.SHOP\_RES\_CASE\_FILE\_KEY, ShopResTypeConstant.SHOP\_RES\_CASE\_FILE\_TYPE);

if (resId == null || resId == 0) {

logger.error("ProjectCaseServiceImpl.add{}:保存配置文件失败");

return 0;

} else {

projectCase.setFileId(BigInteger.valueOf(resId));

}

}

//将富文本中所有图片Id都保存

if (StringUtils.isNotEmpty(projectCase.getPicIds())) {

projectCase.setPicIds(projectCase.getPicIds());

}

// 设置系统字段值

saveSystemInfo(projectCase, loginUser);

// 插入数据库

int addResult = projectCaseDao.insert(projectCase);

if (addResult > 0) {

// 回填资源业务ID

boolean falg = backfill(projectCase);

if (falg) {

return projectCase.getId();

}

}

return 0;

}

@Override

public int delete(Integer id, LoginUser loginUser) {

ProjectCase projectCase = findById(id);

//存储系统字段值

saveSystemInfo(projectCase, loginUser);

return projectCaseDao.delete(projectCase);

}

@Override

public int batchDelByShopId(Integer shopId, LoginUser loginUser) {

ProjectCase projectCase = new ProjectCase();

projectCase.setShopId(shopId);

projectCase.setModifier(loginUser.getLoginName());

projectCase.setGmtModified(new Date());

return projectCaseDao.batchDelByShopId(projectCase);

}

@Override

public int update(ProjectCaseUpdate caseUpdate, LoginUser loginUser) {

//未修改之前的值

ProjectCase oldProjectCase = findById(caseUpdate.getCaseId());

//修改的对象

ProjectCase projectCase = caseUpdate.getProjectCaseFromProjectCaseUpdate(caseUpdate);

// 更新富文本内容

BigInteger oldResId = oldProjectCase.getFileId();

String content = projectCase.getContent();

if (oldResId == null || oldResId.intValue() == 0) {

if (StringUtils.isNotEmpty(content)) {

Integer resId = resFileService.saveFile(caseUpdateFilePath, content, loginUser, caseUpdate.getCaseId(),

ShopResTypeConstant.SHOP\_RES\_CASE\_FILE\_KEY, ShopResTypeConstant.SHOP\_RES\_CASE\_FILE\_TYPE);

if (resId == null || resId == 0) {

logger.error("ProjectCaseServiceImpl.update{}:保存配置文件失败");

return 0;

} else {

projectCase.setFileId(BigInteger.valueOf(resId));

}

}

} else {

boolean falg = resFileService.updateFile(oldResId.longValue(), content);

if (!falg) {

logger.error("ProjectCaseServiceImpl.resFileService.updateFile{}:更新配置文件失败");

return 0;

}

}

// 存储富文本内图片Ids

if (StringUtils.isNotEmpty(projectCase.getPicIds())) {

projectCase.setPicIds(projectCase.getPicIds());

}else{

projectCase.setPicIds(0+"");

}

// 插入数据库

int updateResult = projectCaseDao.update(projectCase);

if (updateResult > 0) {

return updateResult;

}

return 0;

}

@Override

public int updateCase(ProjectCase projectCase, LoginUser loginUser) {

// 更新数据库

projectCase.setGmtModified(new Date());

projectCase.setModifier(loginUser.getLoginName());

int updateResult = projectCaseDao.update(projectCase);

if (updateResult > 0) {

return updateResult;

}

return 0;

}

@Override

public ProjectCase findById(Integer id) {

return projectCaseDao.findById(id);

}

@Override

public ProjectCaseDetailsVO getDetails(Integer id) {

ProjectCaseDetailsVO detailsVO = new ProjectCaseDetailsVO();

ProjectCase projectCase = this.findById(id);

logger.info("查询工程案例详情 ProjectCase:{}",projectCase);

if (projectCase != null && projectCase.getId() != null) {

detailsVO.setCaseId(projectCase.getId());

detailsVO.setCaseTitle(projectCase.getCaseTitle());

detailsVO.setPicIds(projectCase.getPicIds());

if (projectCase.getFileId() != null && projectCase.getFileId().intValue() > 0) {

ResFile resFile = resFileService.getById(projectCase.getFileId().longValue());

String filePath = resFile != null ? resFile.getFilePath() : "";

// detailsVO.setFilePath(filePath);

if (StringUtils.isNotEmpty(filePath)) {

// 读取配置

String fileContext = Utils.getFileContext(rootPath + filePath);

detailsVO.setContent(fileContext);

}

}

//处理图片地址

if (StringUtils.isNotEmpty(projectCase.getPicIds())) {

List<Integer> idList = Utils.getIntegerListFromStringList(projectCase.getPicIds());

if (idList != null && 0 < idList.size()) {

List<ResPic> list=new ArrayList<>();

for (Integer picId:idList){

if (0!=picId){

ResPic resPic = resPicService.selectByPrimaryKey(Long.valueOf(picId));

logger.info("查询店铺博文列表 查询展示图片信息结果 ResPic:{}",resPic);

if (null!=resPic){

list.add(resPic);

}

}

}

detailsVO.setResPics(list);

}

}

}

return detailsVO;

}

@Override

public int getCount(ProjectCaseQuery query) {

return projectCaseDao.findCount(query);

}

@Override

public List<ProjectCaseVO> findList(ProjectCaseQuery query) {

List<ProjectCaseVO> list = projectCaseDao.findList(query);

for (ProjectCaseVO caseVO:list) {

String picIds = caseVO.getPicIds();

if (!"0".equals(picIds)){

List<Integer> idList = Utils.getIntegerListFromStringList(caseVO.getPicIds());

if (idList != null && 0 < idList.size()) {

ResPic resPic = resPicService.selectByPrimaryKey(Long.valueOf(idList.get(0)));

logger.info("查询工程案例列表 查询展示图片信息结果 ResPic:{}",resPic);

if (null!=resPic){

caseVO.setPicPath(resPic.getPicPath());

}

}

}

}

return list;

}

@Override

public Integer deleteCaseByShopId(List<Integer> shopIds,String loginUserName) {

return projectCaseDao.deleteCaseByShopId(shopIds,loginUserName);

}

@Override

public Integer getCaseByShopId(List<Integer> shopIds) {

return projectCaseDao.getCaseByShopId(shopIds);

}

/\*\*

\* 回填资源业务ID

\*

\* @param companyShop

\* @return

\*/

@SuppressWarnings("all")

private boolean backfill(ProjectCase projectCase) {

BigInteger fileId = projectCase.getFileId();

String picIds = projectCase.getPicIds();

//富文本文件

if (fileId != null && fileId.intValue() > 0) {

ResFile resFile = new ResFile();

resFile.setId(fileId.intValue());

resFile.setBusinessId(projectCase.getId());

try {

resFileService.update(resFile);

} catch (Exception e) {

logger.error("backfill{}: caseId="+projectCase.getId()+";fileId="+fileId);

return false;

}

}

//富文本图片ids

if (StringUtils.isNotEmpty(picIds)) {

List<Integer> idList = Utils.getIntegerListFromStringList(picIds);

if (idList != null && 0 < idList.size()) {

ResPicBusinessUpdate resPic = new ResPicBusinessUpdate();

resPic.setIdList(idList);

resPic.setBusinessId(projectCase.getId());

try {

resPicService.batchUpdateBusinessId(resPic);

} catch (Exception e) {

logger.error("backfill{}: caseId="+projectCase.getId()+";picIds="+picIds);

return false;

}

}

}

return true;

}

/\*\*

\* 自动存储系统字段

\*

\* @param ProjectCase model对象

\* @param loginUser 当前用户对象

\*/

@SuppressWarnings("all")

private void saveSystemInfo(ProjectCase projectCase, LoginUser loginUser) {

if (projectCase != null) {

//新增

if (projectCase.getId() == null) {

projectCase.setGmtCreate(new Date());

projectCase.setCreator(loginUser.getLoginName());

projectCase.setIsDeleted(0);

if (projectCase.getSysCode() == null || "".equals(projectCase.getSysCode())) {

projectCase.setSysCode(Utils.getCurrentDateTime(Utils.DATETIMESSS) + "\_" + Utils.generateRandomDigitString(6));

}

}

projectCase.setGmtModified(new Date());

projectCase.setModifier(loginUser.getLoginName());

}

}

}

<?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.sandu.service.shop.dao.CompanyShopArticleDao">

<!--新增博文-->

<insert id="add" parameterType="com.sandu.api.shop.model.CompanyShopArticle" useGeneratedKeys="true" keyProperty="id">

insert into company\_shop\_article

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

<if test="shopId!= null">shop\_id, </if>

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

<if test="releaseStatus!= null">release\_status, </if>

<if test="picIds!= null">pic\_ids, </if>

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

</trim>

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

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

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

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

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

<if test="content!= null"> #{content,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>

</trim>

</insert>

<!--修改博文(博文修改、发布状态修改、是否删除)-->

<update id="update" parameterType="com.sandu.api.shop.model.CompanyShopArticle" >

update company\_shop\_article

<set>

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

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

<if test="releaseStatus!= null"> release\_status = #{releaseStatus,jdbcType=INTEGER}, </if>

<if test="picIds!= null"> pic\_ids = #{picIds,jdbcType=VARCHAR}, </if>

<if test="content!= null"> content = #{content,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="browseCount!= null"> browse\_count = #{browseCount,jdbcType=INTEGER}, </if>

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

</set>

where id = #{id,jdbcType=INTEGER}

</update>

<!--查询博文详情-->

<select id="get" resultType="com.sandu.api.shop.model.CompanyShopArticle">

SELECT \* FROM company\_shop\_article

WHERE is\_deleted = 0

AND id = #{id,jdbcType=INTEGER}

</select>

<!--查询店铺博文数量-->

<select id="getCount" parameterType="com.sandu.api.shop.input.CompanyShopArticleQuery" resultType="java.lang.Integer">

SELECT COUNT(1) FROM company\_shop\_article

WHERE is\_deleted = 0

AND shop\_id = #{shopId,jdbcType=INTEGER}

</select>

<!--查询店铺博文列表-->

<select id="getList" parameterType="com.sandu.api.shop.input.CompanyShopArticleQuery"

resultType="com.sandu.api.shop.model.CompanyShopArticle">

SELECT \* FROM company\_shop\_article

WHERE is\_deleted = 0

AND shop\_id = #{shopId,jdbcType=INTEGER}

ORDER BY gmt\_create DESC

LIMIT #{start}, #{limit}

</select>

<!--根据店铺Id进行批量逻辑删除-->

<update id="deleteByShopId" >

UPDATE company\_shop\_article

SET

is\_deleted= 1,

modifier = #{modifier,jdbcType=VARCHAR},

gmt\_modified = NOW()

WHERE

shop\_id= #{shopId,jdbcType=INTEGER}

</update>

<update id="deleteArticleByShopId" >

UPDATE company\_shop\_article

SET

is\_deleted = 1,

modifier = #{loginUserName,jdbcType=VARCHAR},

gmt\_modified = NOW()

WHERE

AND shop\_id IN

<foreach item="item" collection="shopIds" separator="," open="(" close=")" index="">

#{item}

</foreach>

</update>

<select id="getArticleByShopId" resultType="java.lang.Integer">

SELECT COUNT(id)

FROM company\_shop\_article

WHERE is\_deleted = 0

AND shop\_id IN

<foreach item="item" collection="shopIds" separator="," open="(" close=")" index="">

#{item}

</foreach>

</select>

</mapper>

<?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.sandu.service.shop.dao.CompanyShopDesignPlanDao">

<delete id="deleteDesignPlanByShopId">

DELETE FROM company\_shop\_design\_plan

WHERE shop\_id IN

<foreach item="item" collection="shopIds" separator="," open="(" close=")" index="">

#{item}

</foreach>

</delete>

<select id="getDesignPlanByShopId" resultType="java.lang.Integer">

SELECT COUNT(id)

FROM company\_shop\_design\_plan

WHERE 1=1

AND shop\_id IN

<foreach item="item" collection="shopIds" separator="," open="(" close=")" index="">

#{item}

</foreach>

</select>

</mapper>

<?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.sandu.service.shop.dao.CompanyShopDao">

<!-- \*\*常量定义\*\* -->

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

id,company\_id,company\_pid,user\_id,shop\_code,shop\_name,business\_type,category\_ids,first\_category\_ids,province\_code,

city\_code,area\_code,street\_code,long\_area\_code,shop\_address,contact\_phone,contact\_name,

logo\_pic\_id,cover\_res\_ids,cover\_res\_type,introduced\_file\_id,release\_platform\_values,display\_status,visit\_count,praise\_rate,

sys\_code,creator,gmt\_create,modifier,gmt\_modified,is\_deleted,remark,longitude,latitude,design\_fee\_starting,design\_fee\_ending,

decoration\_price\_start,decoration\_price\_end,working\_years,decoration\_type

</sql>

<!-- \*\*插入定义\*\* -->

<insert id="insertShop" parameterType="com.sandu.api.shop.model.CompanyShop" useGeneratedKeys="true" keyProperty="id">

insert into company\_shop

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

<if test="companyId!= null">company\_id, </if>

<if test="companyPid!= null">company\_pid, </if>

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

<if test="shopCode!= null">shop\_code, </if>

<if test="shopName!= null">shop\_name, </if>

<if test="businessType!= null">business\_type, </if>

<if test="categoryIds!= null">category\_ids, </if>

<if test="firstCategoryIds!= null">first\_category\_ids, </if>

<if test="provinceCode!= null">province\_code, </if>

<if test="cityCode!= null">city\_code, </if>

<if test="areaCode!= null">area\_code, </if>

<if test="streetCode!= null">street\_code, </if>

<if test="longAreaCode!= null">long\_area\_code, </if>

<if test="shopAddress!= null">shop\_address, </if>

<if test="contactPhone!= null">contact\_phone, </if>

<if test="contactName!= null">contact\_name, </if>

<if test="logoPicId!= null">logo\_pic\_id, </if>

<if test="coverResIds!= null">cover\_res\_ids, </if>

<if test="coverResType!= null">cover\_res\_type, </if>

<if test="introducedFileId!= null">introduced\_file\_id, </if>

<if test="releasePlatformValues!= null">release\_platform\_values, </if>

<if test="displayStatus!= null">display\_status, </if>

<if test="visitCount!= null">visit\_count, </if>

<if test="praiseRate!= null">praise\_rate, </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="longitude!= null">longitude, </if>

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

<if test="designFeeStarting!= null">design\_fee\_starting, </if>

<if test="designFeeEnding!= null">design\_fee\_ending, </if>

<if test="workingYears!= null">working\_years, </if>

<if test="decorationPriceStart!= null">decoration\_price\_start, </if>

<if test="decorationPriceEnd!= null">decoration\_price\_end, </if>

<if test="decorationType!= null">decoration\_type, </if>

</trim>

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

<if test="praiseRate!= null"> #{praiseRate,jdbcType=DOUBLE}, </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="longitude!= null"> #{longitude,jdbcType=DOUBLE}, </if>

<if test="latitude!= null"> #{latitude,jdbcType=DOUBLE}, </if>

<if test="designFeeStarting!= null"> #{designFeeStarting,jdbcType=DOUBLE}, </if>

<if test="designFeeEnding!= null"> #{designFeeEnding,jdbcType=DOUBLE}, </if>

<if test="workingYears!= null">#{workingYears}, </if>

<if test="decorationPriceStart!= null"> #{decorationPriceStart}, </if>

<if test="decorationPriceEnd!= null">#{decorationPriceEnd}, </if>

<if test="decorationType!= null">#{decorationType}, </if>

</trim>

</insert>

<!-- \*\*更新定义\*\* -->

<update id="updateShop" parameterType="com.sandu.api.shop.model.CompanyShop">

update company\_shop

<set>

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

<if test="companyPid!= null"> company\_pid = #{companyPid,jdbcType=INTEGER}, </if>

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

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

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

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

<if test="categoryIds!= null"> category\_ids = #{categoryIds,jdbcType=VARCHAR}, </if>

<if test="firstCategoryIds!= null"> first\_category\_ids = #{firstCategoryIds,jdbcType=VARCHAR}, </if>

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

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

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

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

<if test="longAreaCode!= null"> long\_area\_code = #{longAreaCode,jdbcType=VARCHAR}, </if>

<if test="shopAddress!= null"> shop\_address = #{shopAddress,jdbcType=VARCHAR}, </if>

<if test="contactPhone!= null"> contact\_phone = #{contactPhone,jdbcType=VARCHAR}, </if>

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

<if test="logoPicId!= null"> logo\_pic\_id = #{logoPicId,jdbcType=INTEGER}, </if>

<if test="coverResIds!= null"> cover\_res\_ids = #{coverResIds,jdbcType=VARCHAR}, </if>

<if test="coverResType!= null"> cover\_res\_type = #{coverResType,jdbcType=INTEGER}, </if>

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

<if test="releasePlatformValues!= null"> release\_platform\_values = #{releasePlatformValues,jdbcType=INTEGER}, </if>

<if test="displayStatus!= null"> display\_status = #{displayStatus,jdbcType=INTEGER}, </if>

<if test="visitCount!= null"> visit\_count = #{visitCount,jdbcType=INTEGER}, </if>

<if test="praiseRate!= null"> praise\_rate = #{praiseRate,jdbcType=DOUBLE}, </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="longitude!= null"> longitude = #{longitude,jdbcType=DOUBLE}, </if>

<if test="latitude!= null"> latitude = #{latitude,jdbcType=DOUBLE}, </if>

<if test="workingYears!= null">working\_years = #{workingYears}, </if>

<if test="decorationPriceStart!= null">decoration\_price\_start = #{decorationPriceStart}, </if>

<if test="decorationPriceEnd!= null">decoration\_price\_end = #{decorationPriceEnd}, </if>

<if test="decorationType!= null">decoration\_type = #{decorationType}, </if>

design\_fee\_starting = #{designFeeStarting,jdbcType=DOUBLE},

design\_fee\_ending = #{designFeeEnding,jdbcType=DOUBLE},

</set>

where id = #{id,jdbcType=INTEGER}

</update>

<!--逻辑删除店铺-->

<update id="deleteShop" parameterType="com.sandu.api.shop.model.CompanyShop" >

UPDATE

company\_shop

SET

is\_deleted= 1,

modifier = #{modifier,jdbcType=VARCHAR},

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

WHERE

id=#{id,javaType=INTEGER}

</update>

<!--通过Id查询店铺信息-->

<select id="getShopById" parameterType="java.lang.Integer" resultType="com.sandu.api.shop.model.CompanyShop">

SELECT

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

FROM

company\_shop

WHERE

id=#{shopId,jdbcType=INTEGER}

AND is\_deleted = 0

</select>

<!--查询条件下的店铺数量-->

<select id="findCount" parameterType="com.sandu.api.shop.model.po.ShopPO" resultType="java.lang.Integer">

SELECT

count(id)

FROM company\_shop WHERE 1=1

<if test="companyId != null">

<choose>

<when test="companyType != null and companyType == 'true'">

and company\_pid = #{companyId,jdbcType=INTEGER}

and company\_id != #{companyId,jdbcType=INTEGER}

</when>

<otherwise>

and company\_id = #{companyId,jdbcType=INTEGER}

</otherwise>

</choose>

</if>

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

AND shop\_code LIKE CONCAT(CONCAT('%',#{shopCode,jdbcType=VARCHAR}),'%')

</if>

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

AND shop\_name LIKE CONCAT(CONCAT('%',#{shopName,jdbcType=VARCHAR}),'%')

</if>

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

AND contact\_name LIKE CONCAT(CONCAT('%',#{contactName,jdbcType=VARCHAR}),'%')

</if>

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

AND contact\_phone LIKE CONCAT(CONCAT('%',#{contactPhone,jdbcType=VARCHAR}),'%')

</if>

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

AND province\_code = #{provinceCode,jdbcType=VARCHAR}

</if>

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

AND city\_code = #{cityCode,jdbcType=VARCHAR}

</if>

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

AND area\_code = #{areaCode,jdbcType=VARCHAR}

</if>

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

AND street\_code = #{streetCode,jdbcType=VARCHAR}

</if>

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

AND user\_id = #{userId}

</if>

AND is\_deleted = 0

</select>

<!--查询条件下的店铺列表-->

<select id="findList" parameterType="com.sandu.api.shop.model.po.ShopPO" resultType="com.sandu.api.shop.model.CompanyShop">

SELECT

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

FROM company\_shop WHERE 1=1

<if test="companyId != null">

<choose>

<when test="companyType != null and companyType == 'true'">

and company\_pid = #{companyId,jdbcType=INTEGER}

and company\_id != #{companyId,jdbcType=INTEGER}

</when>

<otherwise>

and company\_id = #{companyId,jdbcType=INTEGER}

</otherwise>

</choose>

</if>

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

AND shop\_code LIKE CONCAT(CONCAT('%',#{shopCode,jdbcType=VARCHAR}),'%')

</if>

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

AND shop\_name LIKE CONCAT(CONCAT('%',#{shopName,jdbcType=VARCHAR}),'%')

</if>

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

AND contact\_name LIKE CONCAT(CONCAT('%',#{contactName,jdbcType=VARCHAR}),'%')

</if>

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

AND contact\_phone LIKE CONCAT(CONCAT('%',#{contactPhone,jdbcType=VARCHAR}),'%')

</if>

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

AND province\_code = #{provinceCode,jdbcType=VARCHAR}

</if>

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

AND city\_code = #{cityCode,jdbcType=VARCHAR}

</if>

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

AND area\_code = #{areaCode,jdbcType=VARCHAR}

</if>

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

AND street\_code = #{streetCode,jdbcType=VARCHAR}

</if>

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

AND user\_id = #{userId}

</if>

AND is\_deleted = 0

ORDER BY id DESC

limit #{start},#{limit}

</select>

<!--通过用户id和店铺类型查询店铺信息-->

<!--品牌馆去掉用户过滤条件-->

<select id="findBrandPavilion" resultType="com.sandu.api.shop.model.CompanyShop">

SELECT

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

FROM

company\_shop

WHERE

company\_id = #{companyId}

and business\_type = #{businessType}

AND is\_deleted = 0

limit 1;

</select>

<select id="checkShopName" resultType="java.lang.Integer" >

SELECT

count(id)

FROM company\_shop WHERE 1=1

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

and shop\_name = #{shopName,jdbcType=VARCHAR}

</if>

<if test="shopId != null and shopId > 0">

and id <![CDATA[ <> #{shopId} ]]>

</if>

and is\_deleted = 0

</select>

<update id="updateShopCategoryByCompanyId" parameterType="com.sandu.api.shop.model.CompanyShop">

update company\_shop

SET

category\_ids = #{categoryIds},

first\_category\_ids = #{firstCategoryIds},

gmt\_modified=#{gmtModified}

WHERE

company\_id = #{companyId}

AND is\_deleted = 0

</update>

<select id="getShopListByUserId" parameterType="java.lang.Long" resultType="com.sandu.api.shop.model.CompanyShop">

SELECT

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

FROM company\_shop WHERE 1=1

AND is\_deleted = 0

AND user\_id = #{userId}

</select>

<update id="deleteShopByUserId" >

UPDATE company\_shop

SET

is\_deleted = 1,

modifier = #{loginUserName},

gmt\_modified = NOW()

WHERE user\_id = #{userId}

</update>

</mapper>

<?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.sandu.service.shop.dao.ProjectCaseDao">

<!-- \*\*常量定义\*\* -->

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

id,shop\_id,case\_title,release\_status,pic\_ids,file\_id,sys\_code,creator,gmt\_create,modifier,gmt\_modified,is\_deleted,remark

</sql>

<!-- \*\*插入定义\*\* -->

<insert id="insert" parameterType="com.sandu.api.shop.model.ProjectCase" useGeneratedKeys="true" keyProperty="id">

insert into project\_case

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

<if test="shopId!= null">shop\_id, </if>

<if test="caseTitle!= null">case\_title, </if>

<if test="releaseStatus!= null">release\_status, </if>

<if test="picIds!= null">pic\_ids, </if>

<if test="fileId!= null">file\_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>

</trim>

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

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

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

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

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

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

</trim>

</insert>

<!-- \*\*更新定义\*\* -->

<update id="update" parameterType="com.sandu.api.shop.model.ProjectCase">

update project\_case

<set>

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

<if test="caseTitle!= null"> case\_title = #{caseTitle,jdbcType=VARCHAR}, </if>

<if test="releaseStatus!= null"> release\_status = #{releaseStatus,jdbcType=INTEGER}, </if>

<if test="picIds!= null"> pic\_ids = #{picIds,jdbcType=VARCHAR}, </if>

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

</set>

where id = #{id,jdbcType=INTEGER}

</update>

<!--逻辑删除-->

<update id="delete" parameterType="com.sandu.api.shop.model.ProjectCase" >

UPDATE

project\_case

SET

is\_deleted= 1,

modifier = #{modifier,jdbcType=VARCHAR},

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

WHERE

id=#{id,javaType=INTEGER}

</update>

<!--根据店铺ID批量逻辑删除-->

<update id="batchDelByShopId" parameterType="com.sandu.api.shop.model.ProjectCase" >

UPDATE

project\_case

SET

is\_deleted= 1,

modifier = #{modifier,jdbcType=VARCHAR},

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

WHERE

shop\_id=#{shopId,javaType=INTEGER}

</update>

<!--通过Id查询信息-->

<select id="findById" parameterType="java.lang.Integer" resultType="com.sandu.api.shop.model.ProjectCase">

SELECT

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

FROM

project\_case

WHERE

id=#{id,jdbcType=INTEGER}

AND is\_deleted = 0

</select>

<!--查询条件下的数量-->

<select id="findCount" parameterType="com.sandu.api.shop.input.ProjectCaseQuery" resultType="java.lang.Integer">

SELECT

count(id)

FROM project\_case WHERE 1=1

<choose>

<when test="shopId != null and shopId != 0">

and shop\_id = #{shopId,jdbcType=INTEGER}

</when>

<otherwise>

and shop\_id = -1

</otherwise>

</choose>

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

and case\_title LIKE CONCAT(CONCAT('%',#{caseTitle,jdbcType=VARCHAR}),'%')

</if>

AND is\_deleted = 0

</select>

<!--查询条件下的列表-->

<select id="findList" parameterType="com.sandu.api.shop.input.ProjectCaseQuery"

resultType="com.sandu.api.shop.output.ProjectCaseVO">

SELECT

pc.id as case\_id,pc.case\_title,pc.gmt\_create as create\_date,

pc.release\_status,pc.pic\_ids, rf.file\_path

FROM project\_case pc

LEFT join res\_file rf ON rf.id = pc.file\_id

WHERE 1=1

<choose>

<when test="shopId != null and shopId != 0">

and pc.shop\_id = #{shopId,jdbcType=INTEGER}

</when>

<otherwise>

and pc.shop\_id = -1

</otherwise>

</choose>

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

and pc.case\_title LIKE CONCAT(CONCAT('%',#{caseTitle,jdbcType=VARCHAR}),'%')

</if>

AND pc.is\_deleted = 0

ORDER BY pc.id DESC

limit #{start},#{limit}

</select>

<update id="deleteCaseByShopId" >

UPDATE project\_case

SET

is\_deleted = 1,

modifier = #{loginUserName,jdbcType=VARCHAR},

gmt\_modified = NOW()

WHERE shop\_id IN

<foreach item="item" collection="shopIds" separator="," open="(" close=")" index="">

#{item}

</foreach>

</update>

<select id="getCaseByShopId" resultType="java.lang.Integer">

SELECT COUNT(id)

FROM project\_case

WHERE is\_deleted = 0

AND shop\_id IN

<foreach item="item" collection="shopIds" separator="," open="(" close=")" index="">

#{item}

</foreach>

</select>

</mapper>