五星药房连锁管理系统V1.1源程序-前30页

**package**com.masiinc.webser.service.retail.dao;

**import** java.math.BigDecimal;

**import** java.util.ArrayList;

**import** java.util.HashMap;

**import** java.util.LinkedList;

**import** java.util.List;

**import** java.util.Map;

**import** java.util.Objects;

**import** java.util.Random;

**import** java.util.WeakHashMap;

**import** java.util.logging.Level;

**import** java.util.logging.Logger;

**import** javax.annotation.Resource;

**import** net.sf.json.JSONArray;

**import** net.sf.json.JSONObject;

**import** org.springframework.context.annotation.Scope;

**import** org.springframework.stereotype.Repository;

**import** cn.com.ebsi.platform.common.util.Utility;

**import** cn.com.ebsi.platform.web.dao.EbsiBaseDao;

**import** com.masiinc.common.CommUtil;

**import** com.masiinc.common.gifts.IGiftsService;

**import** com.masiinc.common.gifts.tools.GiftsEnum;

**import** com.masiinc.common.inventoryChange.IinventoryChange;

**import** com.masiinc.common.topushlog.entity.PushLog;

**import** com.masiinc.common.topushlog.service.PushLogService;

**import** com.masiinc.common.utils.MallPushUtil;

**import** com.masiinc.common.warehous.IWarehousService;

**import** com.masiinc.common.warehous.tools.CodeTool;

**import** com.masiinc.common.warehous.tools.WarehousEnum;

**import** com.masiinc.isip.common.util.RandomUtil;

**import** com.masiinc.pay.interfaces.alipay.IalipayService;

**import** com.masiinc.utility.MasiParam;

**import** com.masiinc.utility.MasiUtil;

**import** com.masiinc.webser.interfaces.MasiBussion;

**import** com.masiinc.webser.utils.MasiWebSrvsSupportUtil;

/\*\*

\* 购药结算数据库操作类

\* **@author**<a href="mailTo:lairongjie@masiinc.com">MASIINC</a>

\* **@time** 2016-7-16 16:27:20

\* **@version** 1.0

\* **@since**jdk 1.7.0

\*/

@Scope("prototype")

@Repository("settleDao")

**publicclass**SettleDao**extends** EbsiBaseDao **implements** MasiBussion

{

@Resource(name = "alipayservice")

IalipayService ialipayService;

@Resource(name = "inventoryChange")

**private** IinventoryChange inventoryChange;

@Resource(name = "warehousService")

**private** IWarehousService warehousService;

@Resource(name = "giftsService")

**private** IGiftsService giftsService;

@Resource(name = "pushLogService")

**private** PushLogService pushlogservice;

@Override

**public** String execute(Map<String, Object> param) **throws** RuntimeException

{

// 获取操作信息

String userid = String.*valueOf*(param.get("USERID"));

String username = String.*valueOf*(param.get("USERNAME"));

String deptid = String.*valueOf*(param.get("DEPTID"));

String date = Utility.*get19BitCurTimeAPP*();

BigDecimal daa507 = BigDecimal.*ZERO*;// 保存实收金额

BigDecimal daa508 = BigDecimal.*ZERO*;// 应找零金额

**boolean** isDiscount = Utility.*notNull*(param.get("DAA040"));// 是否手工打折

BigDecimal daa504 = BigDecimal.*ZERO*;// 手工优惠额度

**boolean** isCash = **false**;// 本单是否包含现金结算

BigDecimal cashNumber = BigDecimal.*ZERO*;// 本单现金支付金额

String daa032 = "";// 本单参与的优惠

// 获取预结算数据

Map<String, Object> record = getObject("getRecord", param);

List<Map<String, Object>> detail = list("getDetail", param);

delete("deleteRecordTemp", record); // 删除临时表中的汇总

delete("deleteDetailTemp", record);// 删除临时表中的明细

// 判断是否存在预结算数据

**if** (Utility.*isEmpty*(record) || Utility.*isEmpty*(detail))

{

**return** MasiWebSrvsSupportUtil

.*errorRet*("未检索到需要结算的明细信息，请检查是否已做过结算或重试");

}

daa032 = Utility.*isNull*(record.get("DAA032")) ? "" : record.get(

"DAA032").toString();

// 判断是否有手工输入优惠

**if** (isDiscount)

{

String daa040 = param.get("DAA040").toString();

daa504 = MasiUtil.*toBigDecimal*(param.get("DAA504").toString());

**if** (daa040.equals("1"))

{

record.put("DAA504", daa504);

record.put("DAA506",

MasiUtil.*toBigDecimal*(record.get("DAA506").toString())

.subtract(daa504));

} **else**

{

BigDecimal oldDaa506 = MasiUtil.*toBigDecimal*(record.get(

"DAA506").toString());

BigDecimal daa506 = oldDaa506.multiply(daa504).divide(

**new** BigDecimal(100));

record.put("DAA506", daa506);

record.put("DAA504", oldDaa506.subtract(daa506));

}

record.put("DAA512", "1");

daa032 = daa032 + "9|";

}

record.put("OPERDATE", date);

List<Map<String, String>> aaa417List = (List<Map<String, String>>) param

.get("aaa417List"); // 获取支付方式

List<Map<String, String>> dba001List = (List<Map<String, String>>) param

.get("dba001List"); // 获取储值卡列表

**boolean** IsAllAddScore = **true**; // 单据支付方式是否都积分

BigDecimal notAddScoreSum = BigDecimal.*ZERO*;// 不积分支付方式总额

// 保存支付方式

**for** (Map<String, String> map : aaa417List)

{

map.put("AAA012", String.*valueOf*(param.get("AAA012")));

map.put("DAA001", String.*valueOf*(record.get("DAA001")));// 会员号

map.put("DAA500", String.*valueOf*(record.get("DAA500")));// 单据流水

map.put("DAA654", "1");

map.put("DAA674", Utility.*getUUID*());

map.put("OPERUSERID", userid);

map.put("OPERUSERNAME", username);

map.put("OPERDATE", date);

map.put("ORGID", deptid);

daa507 = daa507.add(MasiUtil.*toBigDecimal*(map.get("DAA673")));

// 如果是现金结算

**if** (map.get("AAA417").equals("1"))

{

isCash = **true**;

cashNumber = cashNumber.add(MasiUtil.*toBigDecimal*(map

.get("DAA673")));

}

// 如果是积分支付

**if** (map.get("AAA417").equals("8"))

{

map.put("DAA672", String.*valueOf*(record.get("DAA001")));

// 如果是线上单据，将会员积分账户解锁

**if** (Objects.*equals*("2", record.get("DAA424")))

{

update("unlockScore", map);

}

// 扣减积分并保存积分流水

**if** (!deductScore(map))

{

**thrownew** RuntimeException("扣减积分失败，请重试或联系管理员！");

}

}

// 如果是支付宝或微信支付

**if** (map.get("AAA417").equals("9") || map.get("AAA417").equals("10"))

{

**if** (param.get("MOBILEPAY").equals("1"))

{

map.put("DAA672", String.*valueOf*(param.get("MIA045")));

// 组织入参

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

inputData.put("REQUESTEPT", param.get("REQUESTEPT"));// 支付类型

inputData.put("MIA001", record.get("DAA526"));// 订单号

inputData.put("MIA006", param.get("DEPTID") + "门店零售收款");// 订单标题

inputData.put("MIA045", param.get("MIA045"));// 授权码

inputData.put("MIA003", map.get("DAA673"));// 支付金额

inputData.put("MIA008", param.get("DEPTID"));// 门店编号

inputData.put("PBA002", param.get("USERID"));// 用户编号

List<Map<String, Object>> listDeltai = **new** ArrayList<>();

**for** (Map<String, Object> goodsMap : detail)

{

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

goodsDetail.put("PBA033", goodsMap.get("AAA008"));

goodsDetail.put("PBA034", goodsMap.get("AAA014"));

goodsDetail.put("PBA035", goodsMap.get("DAA422"));

goodsDetail.put("PBA036", goodsMap.get("DAA522"));

listDeltai.add(goodsDetail);

}

inputData.put("GOODS\_DETAIL", listDeltai);

**try**

{

// 调用云服务接口支付

String args = ialipayService.pay(inputData);

JSONObject jsono = JSONObject.*fromObject*(args);

**if** (!"1".equals(String.*valueOf*(jsono.get("REQUESTRET")))

&& !"0".equals(String.*valueOf*(jsono

.get("REQUESTRET"))))

{

**thrownew** RuntimeException(jsono.get("REQUESTMSG")

.toString());

}

} **catch** (Exception ex)

{

Logger.*getLogger*(SettleDao.**class**.getName()).log(

Level.*SEVERE*, **null**, ex);

**thrownew** RuntimeException("支付失败，错误原因："

+ ex.getMessage());

}

}

}

// 如果是专属余额

**if** (map.get("AAA417").equals("2"))

{

map.put("DAA672", String.*valueOf*(record.get("DAA001")));

// 扣减专属账户并保存流水

**if** (!deductExclusive(map))

{

**thrownew** RuntimeException("专属账户扣减失败，请重试或联系管理员！");

}

}

// 如果是通用账户结算

**if** (map.get("AAA417").equals("3"))

{

map.put("DAA672", String.*valueOf*(record.get("DAA001")));

**if** (!deductCurrent(map))

{

**thrownew** RuntimeException("通用账户扣减失败，请重试或联系管理员！");

}

}

// 如果是代金券结算

**if** (map.get("AAA417").equals("7"))

{

map.put("DAA672", String.*valueOf*(map.get("DBA181")));

updateCoupon(map);

// 将代金券的状态更改为已使用

}

// 该结算方式是否积分,如果不积分，则累计不积分支付金额

**if** (!isAddScore(map.get("AAA417")))

{

IsAllAddScore = **false**;

notAddScoreSum = notAddScoreSum.add(MasiUtil.*toBigDecimal*(map

.get("DAA673")));

// break;

}

insert("savePayWay", map);

}

// 储值卡列表

**for** (Map<String, String> map : dba001List)

{

daa507 = daa507.add(MasiUtil.*toBigDecimal*(String.*valueOf*(map

.get("DBA013"))));// 实收金额

// 更新储值卡消费记录并记录消费明细

map.put("OPERUSERID", userid);

map.put("OPERUSERNAME", username);

map.put("OPERDATE", date);

map.put("ORGID", deptid);

map.put("DBA016", String.*valueOf*(record.get("DAA526")).concat("#")

.concat(map.get("DBA001")));// 从储值卡流水记录号

map.put("DBA015", String.*valueOf*(record.get("DAA526")));// 业务单据号

map.put("DBA011", date);

deductCard(map);

map.put("AAA417", "5");// 结算方式

map.put("DAA674", Utility.*getUUID*());

map.put("DAA500", String.*valueOf*(record.get("DAA500")));// 单据流水

map.put("DAA672", map.get("DBA001"));// 支付号码

map.put("DAA673", String.*valueOf*(map.get("DBA013")));// 支付金额

map.put("DAA654", "1");

insert("savePayWay", map);

**if** (!isAddScore("5"))

{

IsAllAddScore = **false**;

notAddScoreSum = notAddScoreSum.add(MasiUtil.*toBigDecimal*(map

.get("DAA673")));

}

}

BigDecimal sumRebate = BigDecimal.*ZERO*;

// 保存购药明细

**int** i = 0;

**for** (Map<String, Object> map : detail)

{

// 如果返佣状态为是 并且返佣积分大于0，则汇总返佣积分

**if** (CommUtil.*null2Float*(map.get("REBATE")) > 0f

&& CommUtil.*null2String*(map.get("AAA803")).equals("1"))

{

sumRebate = sumRebate.add(MasiUtil.*toBigDecimal*(

map.get("REBATE").toString()).multiply(

Utility.*toBigDecimal*(map.get("DAA422").toString())));

}

// 是否有手工优惠并且是打折优惠，更新明细金额

**if** (isDiscount && param.get("DAA040").equals("2"))

{

BigDecimal daa521 = MasiUtil.*toBigDecimal*(map.get("DAA521")

.toString());// 原单价

BigDecimal daa522 = daa521.multiply(

MasiUtil.*toBigDecimal*(daa504)).divide(

**new** BigDecimal(100));// 折后单价

map.put("DAA522", daa522);

}

map.put("OPERUSERID", userid);

map.put("OPERUSERNAME", username);

map.put("OPERDATE", date);

map.put("ORGID", deptid);

// 更新库存

List<Map<String, Object>> list;

// 如果是自动选择批次

**if** (MasiParam.*get*("P4000013", "1").equals("1"))

{

list = subtractStock(record, map);

} **else**

{

list = subtractStockByBatch(record, map);// 手动选择批次

}

**if** (Utility.*isEmpty*(list) || Utility.*isNull*(list))

{

**thrownew** RuntimeException("库存记录更新失败，请联系管理员处理");

}

**for** (Map<String, Object> mapBatch : list)

{

// 保存库存流转记录

i++;

map.put("DAA422", mapBatch.get("DAA422"));

map.put("BAA332", mapBatch.get("ACA002"));

map.put("DAA523",

MasiUtil.*toBigDecimal*(

String.*valueOf*(map.get("DAA522")), 5).multiply(

MasiUtil.*toBigDecimal*(String.*valueOf*(mapBatch

.get("DAA422")))));

map.put("DAA525", String.*valueOf*(i));

map.put("DAA520",

String.*valueOf*(map.get("DAA500")).concat(

String.*valueOf*(i)));

String aaa742 = getObject("selectAA02ofAAA742", map).toString();

map.put("AAA742",

Utility.*isNull*(aaa742) ? "0.00" : String

.*valueOf*(aaa742));

// 请或批次、采购成本、请货成本

map.put("BAA653", mapBatch.get("BAA653"));

map.put("BAE012", mapBatch.get("BAE012"));

map.put("BAE013", mapBatch.get("BAE013"));

map.put("AAA350", mapBatch.get("AAA350"));

insert("saveDetail", map);

}

}

String daa001 = Utility.*isNull*(record.get("DAA001")) ? "" : record.get(

"DAA001").toString();

// 如果是线上单据，将会员积分账户解锁

JSONArray jSONArray = **new** JSONArray();

List<Map<String, Object>> couponList = **new** ArrayList();

**if** (Objects.*equals*("1", record.get("DAA424")))

{

// 校验是否满足赠送代金券

List<Map<String, Object>> proList = getDeptPromotion(param, detail,

daa001);

// 代金券列表

**if** (Utility.*notEmpty*(proList))

{

// 生成代金券并发送 通知

Map<String, Object> couponMap = generateCoupon(proList, daa001,

param);

**if** (Utility.*notNull*(couponMap.get("DBA180")))

{

jSONArray = (JSONArray) couponMap.get("DBA180");

}

**if** (Utility.*notNull*(couponMap.get("INFROLIST")))

{

couponList = (List<Map<String, Object>>) couponMap

.get("INFROLIST");

}

}

}

record.put("INFROLIST", couponList);

// 保存零售单据信息

**if** (daa507.compareTo(MasiUtil.*toBigDecimal*(String.*valueOf*(record

.get("DAA506")))) == 1)

{

daa508 = daa507.subtract(MasiUtil.*toBigDecimal*(String

.*valueOf*(record.get("DAA506"))));// 应找零金额

} **else**

{

daa508 = BigDecimal.*ZERO*;// 应找零金额

}

record.put("DAA507", daa507);// 实收金额

record.put("DAA508", daa508);// 应找零金额

record.put("DAA509", param.get("DAA509"));// 找零转储值金额

// 如果有应找零金额

**if** (MasiUtil.*toBigDecimal*(daa508).compareTo(BigDecimal.*ZERO*) == 1)

{

// 是否有现金收费

**if** (isCash)

{

**if** (cashNumber.compareTo(daa508) < 1)// 如果现金收付金额小于或等找零金额，则实际找零金额为现金部分金额减去转储值金额

{

record.put("DAA510", cashNumber.subtract(MasiUtil

.*toBigDecimal*(String.*valueOf*(param.get("DAA509")))));

} **else**

{

record.put("DAA510", daa508.subtract(MasiUtil

.*toBigDecimal*(String.*valueOf*(param.get("DAA509")))));

}

} **else**

{

record.put("DAA510", BigDecimal.*ZERO*);// 非现金收费一律不找零

}

} **else**

{

record.put("DAA510", BigDecimal.*ZERO*);// 没有找零金额时，应找零金额为零

}

record.put("DAA515", "1");// 已结算

record.put("OPERUSERID", userid);

record.put("OPERUSERNAME", username);

record.put("OPERDATE", date);

record.put("ORGID", deptid);

record.put("DAA032", daa032);

record.put("DBA180", jSONArray.toString());

// 判断是否有会员

**if** (Utility.*notEmpty*(record.get("DAA001").toString())

&& getMemberLimit(param))

{

// 是否有不积分的结算方式，如果都积分，则直接将预结算积分保存

**if** (IsAllAddScore)

{

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

updateParaMap.put("DAA001", record.get("DAA001"));

updateParaMap.put("AAA012", record.get("AAA012"));

updateParaMap.put("DAA202", record.get("DAA511"));

updateParaMap.put("OPERUSERID", userid);

updateParaMap.put("OPERUSERNAME", username);

updateParaMap.put("OPERDATE", date);

updateParaMap.put("ORGID", deptid);

update("updateIntegral", updateParaMap);// 增减账户积分

// 消息推送

Map<String, String> res = **new** HashMap<>();

res.put("DAA001", updateParaMap.get("DAA001").toString());

String message = "您的账户消费累计:【" + updateParaMap.get("DAA202")

+ "】分。";

res.put("message", message);// 消息推送提示语

**if** (!(CommUtil.*null2BigDecimal*(updateParaMap.get("DAA202"))

.compareTo(BigDecimal.*ZERO*) == 0))

{

**this**.push(res);// 消息推送

}

// **TODO** 插入会员经验增加记录信息

updateParaMap.put("DAA500", record.get("DAA500"));

updateParaMap.put("DAA201", "5");

updateParaMap.put("DAA207", "消费产生积分");

updateParaMap.put(

"DAA200",

Utility.*getCurDateAPP*(

Utility.*DATE\_FORMAT\_YYYYMMDDHHMMSSSSS*).concat(

Utility.*getUUID*().substring(0, 3)));

**if** (MasiUtil.*toBigDecimal*(record.get("DAA511").toString())

.compareTo(BigDecimal.*ZERO*) != 0)

{

insert("insertAddIntegralLog", updateParaMap);// 记录积分消费明细

}

} **else**

{

BigDecimal daa511 = MasiUtil.*toBigDecimal*(record.get("DAA511")

.toString());// 本单积分金额

BigDecimal ZL;// 本单应找零金额

**if** (daa507.compareTo(MasiUtil.*toBigDecimal*(String

.*valueOf*(record.get("DAA506")))) == 1)

{

ZL = daa507.subtract(MasiUtil.*toBigDecimal*(String

.*valueOf*(record.get("DAA506"))));

} **else**

{

ZL = BigDecimal.*ZERO*;// 应找零金额

}

BigDecimal rule = getIntegralRule(

MasiUtil.*toBigDecimal*(record.get("DAA511").toString()),

MasiUtil.*toBigDecimal*(record.get("DAA506").toString()));// 获取积分规则

**if** (notAddScoreSum.compareTo(BigDecimal.*ZERO*) == 1)

{

**if** (rule.compareTo(BigDecimal.*ZERO*) == 1)

{

daa511 = daa511.subtract(notAddScoreSum.multiply(rule));

**if** (daa511.compareTo(BigDecimal.*ZERO*) == -1)

{

daa511 = BigDecimal.*ZERO*;

}

} **else**

{

daa511 = BigDecimal.*ZERO*;

}

}

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

updateParaMap.put("DAA001", record.get("DAA001"));

updateParaMap.put("AAA012", record.get("AAA012"));

updateParaMap.put("DAA202", daa511);

updateParaMap.put("OPERUSERID", userid);

updateParaMap.put("OPERUSERNAME", username);

updateParaMap.put("OPERDATE", date);

updateParaMap.put("ORGID", deptid);

update("updateIntegral", updateParaMap);// 增减账户积分

updateParaMap.put("DAA500", record.get("DAA500")); // 插入会员积分增加记录

// 消息推送

Map<String, String> res = **new** HashMap<>();

res.put("DAA001", updateParaMap.get("DAA001").toString());

String message = "您的账户积分:【" + updateParaMap.get("DAA202")

+ "】分。";

res.put("message", message);// 消息推送提示语

**if** (!(CommUtil.*null2BigDecimal*(updateParaMap.get("DAA202"))

.compareTo(BigDecimal.*ZERO*) == 0))

{

**this**.push(res);// 消息推送

}

updateParaMap.put("DAA201", "5");

updateParaMap.put("DAA207", "消费产生积分");

updateParaMap.put(

"DAA200",

Utility.*getCurDateAPP*(

Utility.*DATE\_FORMAT\_YYYYMMDDHHMMSSSSS*).concat(

Utility.*getUUID*().substring(0, 3)));

**if** (MasiUtil.*toBigDecimal*(daa511).compareTo(BigDecimal.*ZERO*) != 0)

{

insert("insertAddIntegralLog", updateParaMap);// 记录积分消费明细

}

record.put("DAA511", daa511);// 本单积分

}

// 组织增加会员经验需要得参数值

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

experienceMap.put("DAA001", record.get("DAA001"));// 会员账号

experienceMap.put("OPERUSERID", userid);

experienceMap.put("OPERUSERNAME", username);

experienceMap.put("OPERDATE", date);

experienceMap.put("ORGID", deptid);

experienceMap.put("ORGID", deptid);

BigDecimal daa202 = MasiUtil.*toBigDecimal*(

record.get("DAA506").toString()).divide(

MasiUtil.*toBigDecimal*(MasiParam.*get*("P4000059", "1")), 5);

experienceMap.put("DAA202", daa202.intValue());// 增加经验值额度

experienceMap.put("DAA500", record.get("DAA500"));// 单据号

addExperience(experienceMap);

// 获取当前会员的上级会员

Map<String, Object> superMap = getObject("selectSuperior", record);

// 如果有上级会员

**if** (Utility.*notNull*(superMap))

{

superMap.put("Endtime", Utility.*get10BitCurTimeAPP*());

Map<String, Object> DA58Map = getObject("selectDA58Map",

superMap);

// 返佣时判断协议是否锁定,协议日期是否过期过期

**if** (Utility.*notNull*(DA58Map))

{

// TODO给上级会员增加返佣金额(员工只返佣金,会员返佣金和积分)

addRebate(superMap, sumRebate, userid, username, date,

deptid, detail, record);

}

}

// 当前会员增加累计消费金额

update("updateMemberDA03", record);

}

insert("saveRecord", record);

// 如果有找零转储值金额，应给对应账户增加金额

**if** (Utility.*notEmpty*(String.*valueOf*(record.get("DAA509")))

&& MasiUtil.*toBigDecimal*(String.*valueOf*(record.get("DAA509")))

.compareTo(BigDecimal.*ZERO*) == 1

&& Utility.*notEmpty*(String.*valueOf*(record.get("DAA001"))))

{

// 保存账户流水并更新通用余额

addCurrent(record);

}

// 如果有赠品，更新对应赠品库存及流水

**if** (Utility.*notNull*(param.get("giftList")))

{

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

giftMap.put("giftList", param.get("giftList"));

giftMap.put("OPERUSERID", userid);

giftMap.put("OPERUSERNAME", username);

giftMap.put("OPERDATE", date);

giftMap.put("ORGID", deptid);

giftMap.put("DAA500", record.get("DAA500"));

giftMap.put("AAA012", record.get("AAA012"));

subtractGift(giftMap);

}

// 保存处方

**if** (MasiParam.*get*("P4000003").equals("1"))

{

generatePrescribe(record, detail);

}

**if** (Utility.*notEmpty*(daa001))

{

// 消费推送

String message = "您本次消费"

+ MasiUtil.*toBigDecimal*(

String.*valueOf*(record.get("DAA506")), 2) + "元。";

**this**.pushLog(record.get("DAA001").toString(), message, "2");

String AAA779 = "";

**for** (**int** f = 0; f < detail.size(); f++)

{

**if** (Utility.*notNull*(detail.get(f).get("AAA779")))

{

AAA779 += "商品" + detail.get(f).get("AAA014").toString()

+ "的用药指导:" + detail.get(f).get("AAA779");

**if** (f < (detail.size() - 1))

{

AAA779 += ",";

}

}

}

**if** (Utility.*notNull*(AAA779))

{

**if** (Utility.*notEmpty*(AAA779))

{

// 如果拼出来的字符串最后一位是,就截取

String AAA779Name = AAA779.substring(AAA779.length() - 1);

**if** (",".equals(AAA779Name))

{

AAA779 = AAA779.substring(0, AAA779.length() - 1);

}

**this**.pushLog(record.get("DAA001").toString(), AAA779, "2");

}

}

}

**return** MasiWebSrvsSupportUtil.*sucessRet*(record);

}

/\*\*

\* 校验是否满足赠送代金券

\* **@param**

\* **@param**

\* **@return**

\*/

**private** List<Map<String, Object>> getDeptPromotion(

Map<String, Object> parm, List<Map<String, Object>> retailList,

String Daa001)

{

HashMap<Object, Object> memberMap = **new** HashMap<>();

memberMap.put("DAA001", Daa001);// 00代表非会员

memberMap.put("DAA331", "00");// 00代表非会员

**if** (Utility.*notEmpty*(Daa001))

{

memberMap = getObject("getMemberGrade", memberMap);

**if** (Utility.*isNull*(memberMap))

{

memberMap.put("DAA331", "00");// 00代表非会员

}

}

List<Map<String, Object>> allPromotion = **new** ArrayList<>();

List<Map<String, Object>> deptPromotion = **new** ArrayList<>();// 校验完部门和会员后能满足的促销

parm.put("OPERDATE", Utility.*get19BitCurTimeAPP*());

List<Map<String, Object>> list = list("getRule", parm);

// 过滤掉非本门店的促销

**for** (Map<String, Object> map : list)

{

// 校验部门

JSONArray dept = JSONArray.*fromObject*(map.get("DEPTID"));

**if** (dept.contains(parm.get("DEPTID")))

{

// 校验会员等级

JSONArray DAA071 = JSONArray.*fromObject*(map.get("DAA071"));

**if** (DAA071.contains(memberMap.get("DAA331")))

{

deptPromotion.add(map);

}

}

}

// 校验本单购买商品

// 循环本单能满足的所有代金券促销

**for** (Map<String, Object> promotionMap : deptPromotion)

{

BigDecimal sum = BigDecimal.*ZERO*;

BigDecimal Gross = BigDecimal.*ZERO*;

JSONArray AAA010 = JSONArray.*fromObject*(promotionMap.get("AAA010"));

JSONArray AAA011 = JSONArray.*fromObject*(promotionMap.get("AAA011"));

// 循环商品分类

// 循环本单购买商品

**for** (Map<String, Object> retailMap : retailList)

{

// 判断是否是代金券商品

**if** (AAA011.contains(retailMap.get("AAA008")))

{

sum = sum.add(MasiUtil.*toBigDecimal*(retailMap.get("DAA523")

.toString()));

Gross = Gross.add(MasiUtil.*toBigDecimal*(

retailMap.get("BAE012").toString()).multiply(

MasiUtil.*toBigDecimal*(retailMap.get("DAA422")

.toString())));

**continue**;

}

List groupList = getallgroup(retailMap);

**for** (Object object : AAA010)

{

**if** (groupList.contains(object.toString()))

{

sum = sum.add(MasiUtil.*toBigDecimal*(retailMap.get(

"DAA523").toString()));

Gross = Gross.add(MasiUtil.*toBigDecimal*(

retailMap.get("BAE012").toString()).multiply(

MasiUtil.*toBigDecimal*(retailMap.get("DAA422")

.toString())));

**break**;

}

}

}

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

**if** (Objects.*equals*("1", promotionMap.get("DAA390")))

{

sumMap.put("DAA523", sum);

sumMap.put("DAA389", promotionMap.get("DAA389"));

Map<String, Object> map = getObject("getLadder", sumMap);

**if** (Utility.*notNull*(map))

{

promotionMap.putAll(map);

allPromotion.add(promotionMap);

}

} **else**

{

sumMap.put("DAA523", Gross);

sumMap.put("DAA389", promotionMap.get("DAA389"));

Map<String, Object> map = getObject("getLadder", sumMap);

**if** (Utility.*notNull*(map))

{

promotionMap.putAll(map);

allPromotion.add(promotionMap);

}

}

}

**return** allPromotion;

}

/\*\*

\* 生成代金券

\*/

**private**Map generateCoupon(List<Map<String, Object>> proList,

String daa001, Map<String, Object> param)

{

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

List<Map<String, Object>> list = **new** ArrayList<>();

JSONArray array = **new** JSONArray();// 代金券

**for** (Map<String, Object> map : proList)

{

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

Map<String, Object> serMap = getObject("getSerId");

BigDecimal decimal = MasiUtil.*toBigDecimal*(

serMap.get("serialval").toString(), 0).add(BigDecimal.*ONE*);

serMap.put("serialval", decimal);

update("updateSer", serMap);

coupon.put("DBA180", RandomUtil.*getRandomIntByLength*(20));

coupon.put("DBA181", decimal);

coupon.put("DBA192", map.get("DAA389"));

coupon.put("DBA182", "1");

coupon.put("DBA183", "1");

coupon.put("DBA184", map.get("DAA042"));

coupon.put("DBA185", map.get("DAA394"));

coupon.put("DBA187", map.get("DAA395"));

coupon.put("DBA188", map.get("DAA396"));

coupon.put("DBA189", map.get("DAA397"));

coupon.put("DBA190", map.get("DAA398"));

coupon.put("DBA191", map.get("DAA399"));

**if** (Objects.*equals*("1", map.get("DAA393")))

{

coupon.put("DAA001", daa001);

}

coupon.put("DBA193", Utility.*get19BitCurTimeAPP*());

coupon.put(

"DBA194",

Utility.*getDate*(Utility.*get10BitCurTimeAPP*(),

Integer.*valueOf*(map.get("DAA392").toString())));

coupon.put("DBA195", "1");

coupon.put("DBA199", "1");

coupon.put("DEPTID", param.get("DEPTID"));

coupon.put("USERID", param.get("USERID"));

coupon.put("DBA198", Utility.*get19BitCurTimeAPP*());

coupon.put("OPERUSERNAME", param.get("USERNAME"));

insert("insertDb21", coupon);

pushLog(daa001,

"为您赠送了一张价值"

+ MasiUtil.*toBigDecimal*(coupon.get("DBA184")

.toString()) + "元的代金券，代金券号码:" + decimal

+ ",请在" + coupon.get("DBA194") + "日前使用", "2");

array.add(decimal.toString());

list.add(coupon);

}

couponMap.put("DBA180", array);

couponMap.put("INFROLIST", list);

**return** couponMap;

}

/\*\*

\* 获取当前分类的父类和分类的父类的父类直到父类为空

\* **@param** map

\* **@return**

\* **@see** getallgroup(..)

\* **@time** 2016-10-12

\*/

**private** List<String> getallgroup(Map<String, Object> map)

{

List<String> list = **new** LinkedList<>();

**if** (Utility.*notEmpty*(map))

{

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

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

groupMap.put("AAA010", map.get("AAA010"));

groupsMap.put("AAA004", map.get("AAA010"));

list.add(groupsMap.get("AAA004").toString());

**while** (**true**)

{

Map<String, Object> group = getObject("getparentgroup",

groupMap);

**if** (Utility.*notEmpty*(group))

{

groupMap.put("AAA010", group.get("AAA004"));

list.add(group.get("AAA004").toString());

} **else**

{

**break**;

}

}

}

**return** list;

}

/\*\*

\* 消息推送

\* **@param** DAA001 会员账号

\* **@param** Message 消息内容

\* **@param** pushType 推送类别，1-短信 2-手机APP

\* **@return**

\*/

**privateint** pushLog(String DAA001, String Message, String pushType)

{

// 如果没有会员，返回

**if** (Utility.*isEmpty*(DAA001))

{

**return** -1;

}

**int** code = 0;

**try**

{

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

param.put("DAA001", DAA001);

Map<String, Object> res = getObject("getMemInfo", param);

**if** (Utility.*notEmpty*(res))

{

**if** (Utility.*notNull*(res.get("pushuid")))

{

**if** (Utility.*notEmpty*(String.*valueOf*(res.get("pushuid"))))

{

String daa031 = Utility.*isNull*(res.get("daa031")) ? ""

: res.get("daa031").toString();

String message = "尊敬的会员" + daa031 + "您好，" + Message;

code = MallPushUtil.*mallPush*(

String.*valueOf*(res.get("pushuid")), message);

Map<String, Object> log = **new** WeakHashMap<>();

log.put("uuid", RandomUtil.*getRandomIntByLength*(30));

log.put("daa001", res.get("daa001"));

log.put("daa002", res.get("daa002"));

log.put("daa313", pushType);

log.put("dba035", Message);

log.put("daa004", Utility.*get19BitCurTimeAPP*());

log.put("daa731", code > 0 ? "success" : "fail");

log.put("daa733", "2");

log.put("pushuid", res.get("pushuid"));

log.put("operuserid", "sysadmin");

log.put("operusername", "sysadmin");

log.put("operdate", Utility.*get19BitCurTimeAPP*());

log.put("orgid", "00");

log.put("daa003", "1");

insert("saveLog", log);

}

}

}

} **catch** (Exception e)

{

code = -1;

}

**return** code;

}

/\*\*

\* 更新库存（按近效期由近到远）

\* **@param** params

\* **@param** map2

\* **@return**

\*/

**public** List<Map<String, Object>> subtractStock(Map<String, Object> record,

Map<String, Object> params)

{

List<Map<String, Object>> listBatch = **new** ArrayList<>();

BigDecimal number = MasiUtil.*toBigDecimal*(String.*valueOf*(params

.get("DAA422")));

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

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

goodInfo.put("aaa008", params.get("AAA008"));

goodInfo.put("aca003", BigDecimal.*ZERO*.subtract(number));

goodInfo.put("baa507", params.get("DAA521"));

stockInfo.put("deptid", record.get("DEPTID"));

stockInfo.put("baa501", record.get("DAA500"));

stockInfo.put("baa503", 1);

stockInfo.put("operuserid", record.get("OPERUSERID"));

stockInfo.put("operusername", record.get("OPERUSERNAME"));

stockInfo.put("operdate", record.get("OPERDATE"));

stockInfo.put("orgid", record.get("ORGID"));

stockInfo.put("data", goodInfo);

Map<String, Object> resmap = warehousService.updateStock(stockInfo,

WarehousEnum.*TYPE10*);

**if** (Long.*parseLong*(resmap.get("code").toString()) < 1)

{

**thrownew** RuntimeException(resmap.get("msg").toString());

}

listBatch = (List<Map<String, Object>>) resmap.get("reslist");

**for** (Map<String, Object> mapBatch : listBatch)

{

mapBatch.put("DAA422",

Math.*abs*(Double.*valueOf*(mapBatch.get("aca003").toString())));

mapBatch.put("ACA002", mapBatch.get("aca002"));

mapBatch.put("BAA653", mapBatch.get("baa653"));

mapBatch.put("BAE012", mapBatch.get("bae012"));

mapBatch.put("BAE013", mapBatch.get("bae013"));

mapBatch.put("AAA350", mapBatch.get("aaa350"));

}

**return** listBatch;

}

/\*\*

\* 更新库存（手工选择批次）

\* **@param** params

\* **@param** map2

\* **@return**

\*/

**public** List<Map<String, Object>> subtractStockByBatch(

Map<String, Object> record, Map<String, Object> params)

{

List<Map<String, Object>> listBatch = **new** ArrayList<>();

BigDecimal number = MasiUtil.*toBigDecimal*(String.*valueOf*(params

.get("DAA422")));

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

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

goodInfo.put("aaa008", params.get("AAA008"));

goodInfo.put("aca003", BigDecimal.*ZERO*.subtract(number));

goodInfo.put("aca002", params.get("BAA332"));

goodInfo.put("baa337", params.get("DAA521"));

stockInfo.put("deptid", record.get("DEPTID"));

stockInfo.put("baa501", record.get("DAA500"));

stockInfo.put("baa503", 1);

stockInfo.put("operuserid", record.get("OPERUSERID"));

stockInfo.put("operusername", record.get("OPERUSERNAME"));

stockInfo.put("operdate", record.get("OPERDATE"));

stockInfo.put("orgid", record.get("ORGID"));

stockInfo.put("data", goodInfo);

Map<String, Object> resmap = warehousService.updateStock(stockInfo,

WarehousEnum.*TYPE10*);

**if** (Long.*parseLong*(resmap.get("code").toString()) < 1)

{

**thrownew** RuntimeException(resmap.get("msg").toString());

}

listBatch = (List<Map<String, Object>>) resmap.get("reslist");

**for** (Map<String, Object> mapBatch : listBatch)

{

mapBatch.put("DAA422",

Math.*abs*(Double.*valueOf*(mapBatch.get("aca003").toString())));

mapBatch.put("ACA002", mapBatch.get("aca002"));

mapBatch.put("BAA653", mapBatch.get("baa653"));

mapBatch.put("BAE012", mapBatch.get("bae012"));

mapBatch.put("BAE013", mapBatch.get("bae013"));

mapBatch.put("AAA350", mapBatch.get("aaa350"));

}

**return** listBatch;

}

/\*\*

\* 积分支付

\* **@param** param

\* **@return** true 积分扣减成功标志

\*/

**publicboolean** deductScore(Map<String, String> param)

**throws** RuntimeException

{

String money = String.*valueOf*(param.get("DAA673"));

String score = calcScore(money);

Map<String, String> map = getObject("getDaa023Score", param);

**if** (!"1".equals(map.get("DAA024")))

{

**thrownew** RuntimeException("该会员积分账户已被锁定，请联系服务人员");

}

**if** (MasiUtil.*toBigDecimal*(String.*valueOf*(map.get("DAA023"))).compareTo(

MasiUtil.*toBigDecimal*(score)) == -1)

{

**thrownew** RuntimeException("该会员积分账户余额不足，请选择其它支付方式");

}

param.put("DAA202", score);

update("updateScore", param);// 增减账户积分

param.put("DAA201", "5");

param.put("DAA207", "积分抵现消费");

insert("insertScoreLog", param);// 记录积分消费明细

// 消息推送、

param.put("message", "零售积分抵现扣减账户:【" + param.get("DAA202") + "】分。");// 提示语

**this**.push(param);// 消息推送

**returntrue**;

}

/\*\*

\* 根据支付金额获取支付积分

\* **@param** money

\* **@return**

\*/

**public** String calcScore(String money)

{

String p000014 = MasiParam.*get*("P4000014");

String p000015 = MasiParam.*get*("P4000015");

BigDecimal ratio = MasiUtil.*toBigDecimal*(money).divideToIntegralValue(

MasiUtil.*toBigDecimal*(p000015));

**return** MasiUtil.*toBigDecimal*(p000014).multiply(ratio).toString();

}

/\*\*

\* 专属账户支付

\* **@param** param

\* **@return** true 专属账户扣减标志

\*/

**publicboolean** deductExclusive(Map<String, String> param)

**throws** RuntimeException

{

String aaa012 = param.get("AAA012").toString();

**switch** (aaa012)

{

**case**"01":

param.put("ACCOUNTTYPE", "1");

**break**;

**case**"03":

param.put("ACCOUNTTYPE", "2");

**break**;

**case**"04":

param.put("ACCOUNTTYPE", "4");

**break**;

**default**:

param.put("ACCOUNTTYPE", aaa012);

**break**;

}

Map<String, String> map = getObject("getDaa023Exclusive", param);

**if** ("2".equals(map.get("DAA024")))

{

**thrownew** RuntimeException("该会员专属账户已被锁定，请联系服务人员");

}

**if** (MasiUtil.*toBigDecimal*(String.*valueOf*(map.get("DAA023"))).compareTo(

MasiUtil.*toBigDecimal*(map.get("Daa673"))) == -1)

{

**thrownew** RuntimeException("该会员专属账户余额不足，请选择其它支付方式");

}

update("updateExclusive", param);

param.put("DAA201", param.get("ACCOUNTTYPE"));

param.put("DAA207", "门店消费");

insert("insertExclusiveLog", param);

// 账户消息推送

String codevalue = CodeTool.*getCodeName*("account",

param.get("ACCOUNTTYPE"));// 账户类型

String message = "您的" + codevalue + "在零售时扣除:【" + param.get("DAA673")

+ "】分。";

param.put("message", message);// 消息推送提示语

**this**.push(param);

**returntrue**;

}

/\*\*

\* 通用账户支付

\* **@param** param

\* **@return** true 专属账户扣减标志

\*/

**publicboolean** deductCurrent(Map<String, String> param)

**throws** RuntimeException

{

Map<String, String> map = getObject("getDaa023Current", param);

**if** ("2".equals(map.get("DAA024")))

{

**thrownew** RuntimeException("该会员通用账户已被锁定，请联系服务人员");

}

**if** (MasiUtil.*toBigDecimal*(String.*valueOf*(map.get("DAA023"))).compareTo(

MasiUtil.*toBigDecimal*(map.get("Daa673"))) == -1)

{

**thrownew** RuntimeException("该会员通用账户余额不足，请选择其它支付方式");

}

update("updateCurrent", param);

param.put("DAA201", "0");

// param.put("DAA672", param);

param.put("DAA207", "药店消费");

insert("insertCurrentLog", param);

// 账户消息推送

String message = "您的通用账户在零售时扣除:【" + param.get("DAA673") + "】分。";

param.put("message", message);// 消息推送提示语

**this**.push(param);

**returntrue**;

}

/\*\*

\* 扣减储值卡金额

\* **@param** param

\* **@return**

\*/

**publicboolean** deductCard(Map<String, String> param)

{

update("updateCard", param);

insert("insertCardLog", param);

**returntrue**;

}

/\*\*

\* 扣减储值卡金额

\* **@param** param

\* **@return**

\*/

**publicboolean** updateCoupon(Map<String, String> param)

{

update("updateCoupon", param);

**returntrue**;

}

/\*\*

\* 会员在本门店是否积分

\* **@param** paramMap

\* **@return**

\*/

**privateboolean** getMemberLimit(Map<String, Object> paramMap)

{

Map<String, String> map = getObject("getMemberLimit", paramMap);

**return** Utility.*isEmpty*(map);

}

/\*\*

\* 根据积分和金额计算积分比例规则

\* **@param** score

\* **@param** total

\* **@return**

\*/

**private** BigDecimal getIntegralRule(BigDecimal score, BigDecimal amount)

{

**return** score.divide(amount, 5);

}

/\*\*

\* 结算方式是否积分

\* **@param** aaa417

\*/

**privateboolean** isAddScore(String aaa417)

{

**boolean** isAddscore;

**switch** (aaa417)

{

// 现金结算

**case**"1":

isAddscore = MasiParam.*get*("P4000031", "1").equals("1");

**break**;

// 专属余额

**case**"2":

isAddscore = MasiParam.*get*("P4000034", "1").equals("1");

**break**;

// 通用余额

**case**"3":

isAddscore = MasiParam.*get*("P4000030", "1").equals("1");

**break**;

// 医保刷卡

**case**"4":

isAddscore = MasiParam.*get*("P4000028", "1").equals("1");

**break**;

// 储值卡支付

**case**"5":

isAddscore = MasiParam.*get*("P4000032", "1").equals("1");

**break**;

// 银联刷卡

**case**"6":

isAddscore = MasiParam.*get*("P4000029", "1").equals("1");

**break**;

// 代金券

**case**"7":

isAddscore = MasiParam.*get*("P4000007", "1").equals("1");

**break**;

// 积分抵现

**case**"8":

isAddscore = MasiParam.*get*("P4000033", "1").equals("1");

**break**;

// 微信支付

**case**"9":

isAddscore = MasiParam.*get*("P4000061", "1").equals("1");

**break**;

// 支付宝支付

**case**"10":

isAddscore = MasiParam.*get*("P4000062", "1").equals("1");

**break**;

**default**:

isAddscore = **true**;

**break**;

}

**return** isAddscore;

}

/\*\*

\* 找零转储值

\* **@param** param

\*/

**publicvoid** addCurrent(Map<String, Object> param)

{

update("addCurrent", param);

insert("insertCurrentAddLog", param);

Map<String, String> res = **new** HashMap<>();

res.put("DAA001", param.get("DAA001").toString());

String message = "您的通用账户在零售找零转储:【" + param.get("DAA509") + "】分。";

res.put("message", message);// 消息推送提示语

**if** (!(CommUtil.*null2BigDecimal*(param.get("DAA509")).compareTo(

BigDecimal.*ZERO*) == 0))

{

**this**.push(res);// 消息推送

}

}

// 更新赠品库存

**publicvoid** subtractGift(Map<String, Object> params)

**throws** RuntimeException

{

List<Map<String, String>> giftList = (List<Map<String, String>>) params

.get("giftList");

**for** (Map<String, String> giftMap : giftList)

{

giftMap.put("OPERUSERID", String.*valueOf*(params.get("OPERUSERID")));

giftMap.put("OPERUSERNAME",

String.*valueOf*(params.get("OPERUSERNAME")));

giftMap.put("OPERDATE", String.*valueOf*(params.get("OPERDATE")));

giftMap.put("ORGID", String.*valueOf*(params.get("ORGID")));

giftMap.put("DAA031", giftMap.get("DAA031"));

giftMap.put("DAA564", "1");

giftMap.put("DAA565", String.*valueOf*(giftMap.get("DBA014")));

giftMap.put("DAA566", String.*valueOf*(params.get("OPERDATE")));

giftMap.put("DAA567", "2");

giftMap.put(

"DAA560",

String.*valueOf*(params.get("DAA500"))

.concat(String.*valueOf*(giftMap.get("DAA031")))

.concat(giftMap.get("DAA562"))

.concat(giftMap.get("DAA561")));

giftMap.put("DAA500", String.*valueOf*(params.get("DAA500")));

// 如果是商品

**if** (giftMap.get("DAA561").equals("1"))

{

List<Map<String, String>> list = list("getGiftSotck", giftMap);

**if** (Utility.*isEmpty*(list))

{

**thrownew** RuntimeException("查询赠品库存信息失败，请检查赠品库存是否为空或赠品已发放完毕");

}

Map<String, String> map = list.get(0);

giftMap.putAll(map);

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

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

goodInfo.put("aaa008", giftMap.get("DAA562"));

goodInfo.put("aca003",

0 - CommUtil.*null2Float*(giftMap.get("DAA564")));

stockInfo.put("deptid", giftMap.get("ORGID"));

stockInfo.put("baa501", giftMap.get("DAA500"));

stockInfo.put("baa503", 3);

stockInfo.put("operuserid", giftMap.get("OPERUSERID"));

stockInfo.put("operusername", giftMap.get("OPERUSERNAME"));

stockInfo.put("operdate", giftMap.get("OPERDATE"));

stockInfo.put("orgid", giftMap.get("ORGID"));

stockInfo.put("data", goodInfo);

Map<String, Object> resmap = warehousService.updateStock(

stockInfo, WarehousEnum.*TYPE10*);

**if** (Long.*parseLong*(resmap.get("code").toString()) < 1)

{

**thrownew** RuntimeException(resmap.get("msg").toString());

} **else**

{

update("updateGiftNumber", giftMap);// 更新赠品表库存

}

} **else**

// 如果是非商品

{

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

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

turnoverlog.put("deptid", params.get("ORGID"));// 出库仓库

turnoverlog.put("aaa012", params.get("AAA012"));// 财务类别

turnoverlog.put("dba041", params.get("DAA500"));// 流转流水号

turnoverlog.put("dba044", "4");// 赠品业务类型(仓库减库存)

turnoverlog.put("operdeptid", params.get("ORGID"));// 部门编码

turnoverlog.put("operuserid", params.get("OPERUSERID"));// 部门编码

turnoverlog.put("operusername", params.get("OPERUSERNAME"));// 部门编码

turnoverlog.put("orgid", params.get("ORGID"));// 部门编码

turnoverlog.put("operdate", Utility.*get19BitCurTimeAPP*());// 部门编码

data.put("dba010", giftMap.get("DAA562"));// 礼品编号

data.put("dba031", "-" + giftMap.get("DAA564"));// 仓库减库存(数量)

data.put("dba014", giftMap.get("DBA014"));

turnoverlog.put("data", data);

giftsService.updateStock(turnoverlog, GiftsEnum.*TYPE10*);

}

insert("insertGiftLog", giftMap);

}

}

/\*\*

\* 生成处方信息

\* **@param** params 单据信息

\* **@param** goodsList 商品明细列表

\*/

**publicvoid** generatePrescribe(Map<String, Object> params,

List<Map<String, Object>> goodsList)

{

String date = Utility.*get19BitCurTimeAPP*();

List<Map<String, String>> doctorList = list("getDoctorInfo", params);

**if** (Utility.*isEmpty*(doctorList))

{

**return**;

}

String doctor = doctorList.get(0).get("USERID");

Map<String, String> daa610map = getObject("getNextValDaa610");

String daa610 = RandomUtil.*getRandomIntByLength*(20);

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

Map<String, String> memberMap;

// 如果是会员结算，使用会员信息生成处方单

**if** (Utility.*notNull*(params.get("DAA001"))

&& Utility.*notEmpty*(params.get("DAA001").toString()))

{

String daa001 = String.*valueOf*(params.get("DAA001"));

Map<String, Object> daa001Map = **new** HashMap<>();

daa001Map.put("DAA001", daa001);

memberMap = getObject("getMemberInfo", daa001Map);

} **else**

{

// 如果是非会员结算，随机抽取会员信息生成处方单

Map<String, Object> daa001Map = **new** HashMap<>();

daa001Map.put("RANDOM", getRandomDaa001());

memberMap = getObject("getMemberInfoLimt", daa001Map);

}

// 保存处方信息

String age = Utility.*calcAge*(Utility.*toDate*(memberMap.get("DAA035")));

prescribeMap.put("DAA610", daa610);

prescribeMap.put("DAA619", params.get("DEPTID"));

prescribeMap.put("DAA001", params.get("DAA001"));

prescribeMap.put("DAA650", memberMap.get("DAA031"));

prescribeMap.put("DAA612", memberMap.get("DAA034"));

prescribeMap.put("DAA613", age);

prescribeMap.put("USERTEL", memberMap.get("DAA002"));

prescribeMap.put("DAA618", memberMap.get("DAA037"));

prescribeMap.put("USERADD", memberMap.get("DAA038"));

prescribeMap.put("DAA614", doctor);

prescribeMap.put("DAA615", "");

prescribeMap.put("DAA616", date);

prescribeMap.put("DAA617", "1");

prescribeMap.put("USERID", params.get("OPERUSERID"));

prescribeMap.put("USERNAME", params.get("OPERUSERNAME"));

prescribeMap.put("DEPTID", params.get("ORGID"));

prescribeMap.put("OPERDATE", date);

insert("insertPre", prescribeMap);

// 保存处方明细

**for** (Map<String, Object> goodsMap : goodsList)

{

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

preDetail.put("DAA610", daa610);

preDetail.put("DAA620",

daa610.concat(goodsMap.get("AAA008").toString()));

preDetail.put("AAA008", goodsMap.get("AAA008"));

preDetail.put("DAA621", goodsMap.get("DAA422"));

preDetail.put("DAA622", goodsMap.get("1"));

preDetail.put("USERID", goodsMap.get("OPERUSERID"));

preDetail.put("USERNAME", goodsMap.get("OPERUSERNAME"));

preDetail.put("DEPTID", goodsMap.get("ORGID"));

preDetail.put("OPERDATE", date);

insert("insertPreDetail", preDetail);

}

}

/\*\*

\* 随机抽取会员信息，用于生成处方信息

\* **@return**

\*/

**publicint** getRandomDaa001()

{

Map<String, String> memberCountMember = getObject("getMemberCount");

String memberCount = String.*valueOf*(memberCountMember

.get("MEMBERCOUNT"));

Random random = **new** Random();

**int** rodom = random.nextInt(Integer.*valueOf*(memberCount));

**return** rodom;

}

/\*\*

\* 根据单据应收金额增加会员经验值

\* **@param** updateParaMap

\*/

**publicvoid** addExperience(Map<String, Object> updateParaMap)

{

update("updateExperience", updateParaMap);// 更新账户经验

BigDecimal daa202 = MasiUtil.*toBigDecimal*(updateParaMap.get("DAA202")

.toString());

**if** (MasiUtil.*toBigDecimal*(daa202).compareTo(BigDecimal.*ZERO*) != 0)

{

updateParaMap

.put("DAA200",

Utility.*getCurDateAPP*(

Utility.*DATE\_FORMAT\_YYYYMMDDHHMMSSSSS*)

.concat(Utility.*getUUID*().substring(0, 3)));

insert("insertExperienceRecod", updateParaMap);// 插入会员经验值增减明细

}

}

**publicvoid** addRebate(Map<String, Object> params, BigDecimal score,

String operuserid, String operusername, String operdate,

String deptid, List<Map<String, Object>> list,

Map<String, Object> record)

{

**try**

{

**if** (Utility.*notEmpty*(params))

{

String type = "";

**if** (CommUtil.*null2String*(params.get("PERTYPE")).equals("1"))

{

// 员工

type = MasiParam.*get*("P8000011");

} **elseif** (CommUtil.*null2String*(params.get("PERTYPE")).equals(

"2"))

{

// 会员

type = MasiParam.*get*("P8000009");

}

**if** (type.equals("1"))

{

// 现金

**if** (score.compareTo(BigDecimal.*ZERO*) > 0)

{

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

rebateMap.put("DAB552", score);// 可提现金额

rebateMap.put("PERTYPE", params.get("PERTYPE"));// 推荐人类型

rebateMap.put("PERID", params.get("PERID"));// 推荐人ID

rebateMap.put("DAB551", params.get("DAB551"));// 推荐人ID

rebateMap.put("OPERUSERID", operuserid);

rebateMap.put("OPERUSERNAME", operusername);

rebateMap.put("OPERDATE", operdate);

insert("updatememberda55", rebateMap);

**for** (Map<String, Object> map : list)

{

// 是否返佣 0：否 1：是 并且返佣金额大于0

**if** (CommUtil.*null2String*(map.get("AAA803")).equals(

"1")

&& CommUtil.*null2Float*(map.get("REBATE")) > 0f)

{

map.put("DAB551", params.get("DAB551"));

map.put("DAA001", params.get("DAA001"));

map.put("DAB561", "1");

map.put("DAB562", (MasiUtil.*toBigDecimal*(map

.get("REBATE").toString())

.multiply(Utility.*toBigDecimal*(map.get(

"DAA422").toString()))));

map.put("DAB563", operdate);

map.put("DAA520", map.get("DAA520"));

map.put("DEPTID", deptid);

map.put("DAB564", "2");

map.put("OPERUSERID", operuserid);

map.put("OPERUSERNAME", operusername);

map.put("OPERDATE", operdate);

map.put("ORGID", deptid);

insert("insertmemberda56", map);

}

}

}

} **elseif** (type.equals("2"))

{

// 积分

**float** allnum = 0f;

**for** (Map<String, Object> map : list)

{

// 是否返佣 0：否 1：是

**if** (CommUtil.*null2String*(map.get("AAA803")).equals("1")

&& CommUtil.*null2Float*(map.get("AAA802")) > 0f)

{

allnum += CommUtil.*null2Float*(map.get("AAA802"))

\* CommUtil.*null2Float*(map.get("DAA422"));

map.put("DAB551", params.get("DAB551"));

map.put("DAA001", params.get("DAA001"));

map.put("DAB561", "3"); // 积分收入

map.put("DAB562",

CommUtil.*null2Float*(map.get("AAA802"))

\* CommUtil.*null2Float*(map

.get("DAA422")));

map.put("DAB563", operdate);

map.put("DAA520", map.get("DAA520"));

map.put("DEPTID", deptid);

map.put("DAB564", "2");

map.put("OPERUSERID", operuserid);

map.put("OPERUSERNAME", operusername);

map.put("OPERDATE", operdate);

map.put("ORGID", deptid);

insert("insertmemberda56", map);

}

}

**if** (allnum > 0f)

{

// 插入积分流水 并推送

Map<String, Object> inserMap = **new** WeakHashMap<>();

inserMap.put("DAA200",

RandomUtil.*getRandomIntByLength*(20));

inserMap.put("DAA001", params.get("PERID"));

inserMap.put("DAA500", list.get(0).get("DAA500"));

inserMap.put("DAA500", list.get(0).get("DAA500"));

inserMap.put("DAA201", "5");

inserMap.put("DAA202", allnum);

inserMap.put("DAA207", "消费返佣");

inserMap.put("AAA012", record.get("AAA012"));

inserMap.put("DAA209", record.get("DEPTID"));

inserMap.put("DAA210", operuserid);

inserMap.put("DAA211", operdate);

inserMap.put("OPERUSERID", operuserid);

inserMap.put("OPERUSERNAME", operusername);

inserMap.put("OPERDATE", operdate);

inserMap.put("ORGID", deptid);

inserMap.put("DAA023", allnum);

**int** code = update("UPDATEDA25", inserMap);

**if** (code > 0)

{

insert("SAVEDA28", inserMap);

String message = "积分账户变更：+"

+ MasiUtil.*subZeroAndDot*(allnum);

**int** pushcode = MallPushUtil.*managePush*(

String.*valueOf*(params.get("pushuid")),

message);

// 插入推送记录

PushLog plog = **new** PushLog();

plog.preInsert();

plog.setDaa001(CommUtil.*null2String*(params

.get("PERID")));

plog.setDaa003("1");

plog.setPushuid(CommUtil.*null2String*(params

.get("pushuid")));

plog.setDaa313("2");// 1-短信 2-手机APP

plog.setDba035(message);

plog.setDaa731(pushcode > 0 ? "success" : "fail");

plog.setDaa004(Utility.*get19BitCurTimeAPP*());

pushlogservice.save(plog);

}

inserMap = **null**;

}

}

}

} **catch** (Exception e)

{

e.printStackTrace();

}

}

// 消息推送

**privatevoid** push(Map<String, String> params)

{

**try**

{

// 账户变更消息推送

Map<String, Object> res = getObject("getMemInfo", params);

**if** (Utility.*notEmpty*(res))

{

**if** (Utility.*notNull*(res.get("pushuid")))

{

**if** (Utility.*notEmpty*(String.*valueOf*(res.get("pushuid"))))

{

**int** pushcode = MallPushUtil.*managePush*(

String.*valueOf*(res.get("pushuid")),

params.get("message").toString());

// 插入推送记录

PushLog plog = **new** PushLog();

plog.preInsert();

plog.setDaa001(String.*valueOf*(res.get("daa001")));

plog.setDaa003("1");

plog.setPushuid(String.*valueOf*(res.get("pushuid")));

plog.setDaa313("2");// 1-短信 2-手机APP

plog.setDba035(params.get("message").toString());

plog.setDaa731(pushcode > 0 ? "success" : "fail");

plog.setDaa004(Utility.*get19BitCurTimeAPP*());

pushlogservice.save(plog);

}

}

}

} **catch** (Exception e)

{

}

}

}

package com.masiinc.webser.service.retail.dao;

import java.math.BigDecimal;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.annotation.Resource;

import net.sf.json.JSONArray;

import org.apache.log4j.Logger;

import org.springframework.stereotype.Repository;

import cn.com.ebsi.platform.common.util.Utility;

import cn.com.ebsi.platform.web.dao.EbsiBaseDao;

import com.masiinc.common.CommUtil;

import com.masiinc.common.gifts.IGiftsService;

import com.masiinc.common.gifts.tools.GiftsEnum;

import com.masiinc.common.inventoryChange.IinventoryChange;

import com.masiinc.common.topushlog.entity.PushLog;

import com.masiinc.common.topushlog.service.PushLogService;

import com.masiinc.common.utils.MallPushUtil;

import com.masiinc.common.warehous.IWarehousService;

import com.masiinc.common.warehous.tools.WarehousEnum;

import com.masiinc.shop.ordermanage.service.OrderManageService;

import com.masiinc.utility.MasiParam;

import com.masiinc.utility.MasiUtil;

import com.masiinc.webser.interfaces.MasiBussion;

import com.masiinc.webser.utils.MasiWebSrvsSupportUtil;

/\*\*

\* 整单退费

\* @author <a href="raoyuan@masiinc.com">MASIINC</a>

\* @time 2016年7月24日 下午5:14:51

\* @version 1.0

\* @since jdk 1.7.0\_07

\*/

@Repository("fullBillsrRecedeDao")

public class FullBillsrRecedeDao extends EbsiBaseDao implements MasiBussion

{

@Resource(name = "inventoryChange")

private IinventoryChange inventoryChange;

@Resource(name = "warehousService")

private IWarehousService warehousService;

@Resource(name = "giftsService")

private IGiftsService giftsService;

@Resource(name = "pushLogService")

private PushLogService pushlogservice;

private static final Logger LOG = Logger

.getLogger(OrderManageService.class);

@Override

public String execute(Map<String, Object> params)

{

// 获取零售单信息

Map<String, Object> resDA24 = getObject("getRetailDA24", params); // 商品零售记录信息

if (Utility.isEmpty(resDA24))

{

return MasiWebSrvsSupportUtil.errorRet("未找到该单据信息！");

}

if (resDA24.get("DAA515").equals("3"))

{

return MasiWebSrvsSupportUtil.errorRet("此单已经退费部分商品,请使用分单退费！");

}

List<Map<String, Object>> resDA26 = list("getRetailDA26", params);// 商品零售详细信息

if (Utility.isEmpty(resDA26))

{

return MasiWebSrvsSupportUtil.errorRet("获取单据详细失败！");

}

String DAA506 = resDA24.get("DAA506").toString();// 应该退费金额

String nowTime = Utility.get19BitCurTimeAPP();

// 1.插入退费记录

String DAA530 = Utility

.getCurDateAPP(Utility.DATE\_FORMAT\_YYYYMMDDHHMMSSSSS); // 退费单号

params.put("DAA530", DAA530);

params.put("DAA533", resDA24.get("DAA506"));// 退费总额

params.put("DAA461", resDA24.get("DAA511"));// 扣除积分总额

params.put("DAA760", resDA24.get("DAA760"));// 标识

params.put("DAA534", nowTime);// 退费时间

params.put("DAA424", resDA24.get("DAA424"));// 退费时间

// 插入零售退费记录表信息

int resDA27 = insert("insertRETAILDA27", params);

// if (resDA27 < 1)

// {

// return MasiWebSrvsSupportUtil.errorRet("保存零售退费记录失败,请与服务人员联系！");

// }

// 插入零售退费记录详细

for (Map<String, Object> map : resDA26)

{

map.put("DAA550", DAA530 + "|" + map.get("DAA525").toString());// 唯一号

map.put("DAA530", DAA530);// 退费单号

map.put("USERID", params.get("USERID"));

map.put("USERNAME", params.get("USERNAME"));

map.put("DEPTID", params.get("DEPTID"));

map.put("OPERDATE", nowTime);

int resDA29 = insert("insertRETAILDA29", map);

// if (resDA29 != 1)

// {

// return MasiWebSrvsSupportUtil

// .errorRet("保存零售退费记录详细失败,请与服务人员联系！");

// }

}

// 插入退费方式

params.put("DAA540", DAA530.concat("1")); // 零售退费方式唯一号

int resDA28 = insert("insertRETAILDA28", params);

// if (resDA28 != 1)

// {

// return MasiWebSrvsSupportUtil.errorRet("保存零售退费方式失败,请与服务人员联系！");

// }

// 2、 如果是会员， 扣除会员积分

if (resDA24.containsKey("DAA001")

&& Utility.notEmpty(String.valueOf(resDA24.get("DAA001"))))

{

Map<String, Object> da03Map = new HashMap<>();

da03Map.put("DAA001", resDA24.get("DAA001"));

da03Map.put("DEPTID", params.get("DEPTID"));

da03Map.put("USERID", params.get("USERID"));

da03Map.put("USERNAME", params.get("USERNAME"));

da03Map.put("OPERDATE", nowTime);

da03Map.put("DAA533", params.get("DAA533"));

// 更新会员累计消费金额

update("updateDa03", da03Map);

// 校验会员积分是否足够扣减

Map daa023Map = getObject("getMemberDa25", resDA24);

BigDecimal daa023 = MasiUtil.toBigDecimal(String.valueOf(daa023Map

.get("DAA023")));

// 获取允许积分下限，当前余额+积分负值下限为可扣减总积分

String p6000007 = MasiParam.get("P6000007", "0");

daa023 = daa023.add(MasiUtil.toBigDecimal(p6000007));

if (daa023.compareTo(MasiUtil.toBigDecimal(String.valueOf(resDA24

.get("DAA511")))) == -1)

{

throw new RuntimeException("该笔结算需退还积分，账户积分不足扣减，退费失败");

}

resDA24.put("DEPTID", params.get("DEPTID"));

resDA24.put("USERID", params.get("USERID"));

resDA24.put("USERNAME", params.get("USERNAME"));

resDA24.put("OPERDATE", nowTime);

int index = update("updateRetailDA25", resDA24);

// if (index != 1)

// {

// return MasiWebSrvsSupportUtil.errorRet("更新会员积分信息失败！");

// }

// 账户变更消息推送

if (index > 0)

{

if (!(CommUtil.null2BigDecimal(params.get("DAA511")).compareTo(

BigDecimal.ZERO) == 0))

{

this.push(resDA24);

}

}

// 插入会员支出明细

Map<String, Object> parmDA29 = new HashMap<>();

parmDA29.put("DAA001", resDA24.get("DAA001"));

parmDA29.put("DAA221", params.get("DAA500"));

parmDA29.put("DAA201", "5");

parmDA29.put("DAA202", resDA24.get("DAA511"));

parmDA29.put("DAA207", "零售退货积分退回");

parmDA29.put("DAA223", nowTime);

parmDA29.put("AAA012", params.get("AAA012"));

parmDA29.put("DAA209", params.get("DEPTID"));

parmDA29.put("DAA210", params.get("USERID"));

parmDA29.put("DAA211", resDA24.get("DAA516"));

parmDA29.put("OPERUSERID", params.get("USERID"));

parmDA29.put("OPERUSERNAME", params.get("USERNAME"));

parmDA29.put("OPERDATE", nowTime);

parmDA29.put("ORGID", params.get("DEPTID"));

int resMEMBERDA29 = insert("insertMEMBERDA29", parmDA29);

}

// 3.门店库存 返现

for (Map<String, Object> map : resDA26)

{

BigDecimal number = MasiUtil.toBigDecimal(String.valueOf(map

.get("DAA422")));

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

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

goodInfo.put("aaa008", map.get("AAA008"));

goodInfo.put("aca003", number);

goodInfo.put("aca002", map.get("BAA332"));

goodInfo.put("baa507", map.get("DAA522"));

goodInfo.put("baa653", map.get("BAA653"));

stockInfo.put("deptid", params.get("DEPTID"));

stockInfo.put("baa501", params.get("DAA500"));

stockInfo.put("baa503", 2);

stockInfo.put("operuserid", params.get("USERID"));

stockInfo.put("operusername", params.get("USERNAME"));

stockInfo.put("operdate", nowTime);

stockInfo.put("orgid", params.get("DEPTID"));

stockInfo.put("data", goodInfo);

Map<String, Object> resmap = warehousService.updateStock(stockInfo,

WarehousEnum.TYPE9);

if (Long.parseLong(resmap.get("code").toString()) < 1)

{

throw new RuntimeException(resmap.get("msg").toString());

}

}

// 更新单据详细

for (Map<String, Object> mapDetail : resDA26)

{

mapDetail.put("DEPTID", params.get("DEPTID"));

mapDetail.put("USERID", params.get("USERID"));

mapDetail.put("USERNAME", params.get("USERNAME"));

mapDetail.put("OPERDATE", nowTime);

mapDetail.put("DAA500", params.get("DAA500"));

int updateDA26 = update("updateRetailDA26", mapDetail);

// if (updateDA26 != 1)

// {

// return MasiWebSrvsSupportUtil.errorRet("更新单据详细状态失败！");

// }

}

// 更新单据状态

int resRETAILDA24 = update("updateRETAILDA24", params);

// if (resRETAILDA24 != 1)

// {

// return MasiWebSrvsSupportUtil.errorRet("更新商品零售记录单据状态失败！");

// }

// 获取单据赠品信息

List<Map<String, Object>> GiftInfoOfBill = list("getGiftInfoOfBill",

params);

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

StringBuffer giftmessage = new StringBuffer();

// 赠品库存退回

if (Utility.notEmpty(GiftInfoOfBill))

{

giftmessage.append("请回收赠品：");

for (Map<String, Object> map : GiftInfoOfBill)

{

map.put("USERID", params.get("USERID"));

map.put("USERNAME", params.get("USERNAME"));

map.put("OPERDATE", nowTime);

if (map.get("DAA561").toString().equals("2"))

{

// Integer updateres = update("updateGiftStore", map);

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

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

turnoverlog.put("deptid", params.get("DEPTID"));// 出库仓库

// turnoverlog.put("dba021", params.get("aaa012"));// 出库仓库

turnoverlog.put("aaa012", params.get("AAA012"));// 财务类别

turnoverlog.put("dba041", params.get("DAA500"));// 流转流水号

turnoverlog.put("dba044", "3");// 赠品业务类型(仓库减库存)

turnoverlog.put("operdeptid", params.get("DEPTID"));// 部门编码

turnoverlog.put("operuserid", params.get("USERID"));// 部门编码

turnoverlog.put("operusername", params.get("USERNAME"));// 部门编码

turnoverlog.put("orgid", params.get("DEPTID"));// 部门编码

turnoverlog.put("operdate", Utility.get19BitCurTimeAPP());// 部门编码

data.put("dba010", map.get("DAA562"));// 礼品编号

data.put("dba031", map.get("DAA564"));// 仓库减库存(数量)

data.put("dba014", map.get("DAA565"));

turnoverlog.put("data", data);

giftsService.updateStock(turnoverlog, GiftsEnum.TYPE8);

} else

{

Map<String, Object> infoMap = getObject(

"getaca003ofmaxaca006", map);

infoMap.put("DAA564", map.get("DAA564"));

infoMap.put("USERID", params.get("USERID"));

infoMap.put("USERNAME", params.get("USERNAME"));

infoMap.put("OPERDATE", nowTime);

update("addaca003ofba02", infoMap);// 返回门店库存

update("updateGiftStore", map);

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

turnoverlog.put("deptid", params.get("DEPTID"));

turnoverlog.put("aaa008", map.get("DAA562"));

turnoverlog.put("baa501", params.get("DAA500"));

turnoverlog.put("baa502", "1");

turnoverlog.put("baa503", "2");

turnoverlog.put("baa504", map.get("DAA564"));

turnoverlog.put("baa505", map.get("DAA565"));

turnoverlog.put("aaa012", params.get("AAA012"));

turnoverlog.put("operuserid", params.get("USERID"));

turnoverlog.put("operusername", params.get("USERNAME"));

turnoverlog.put("operdate", nowTime);

turnoverlog.put("orgid", params.get("DEPTID"));

inventoryChange.saveInventoryChangeRecord(turnoverlog);

}

// if (updateres != 1)

// {

// return MasiWebSrvsSupportUtil.errorRet("更新赠品库存信息失败！");

// }

giftmessage.append("[" + map.get("DAA563") + " 数量："

+ map.get("DAA564") + "]");

}

reslutMap.put("giftList", GiftInfoOfBill);

reslutMap.put("giftmsg", giftmessage.toString());

}

reslutMap.put("DAA541", "1");

reslutMap.put("DAA506", DAA506);

reslutMap.put("DAA530", DAA530);

updateCoupon(resDA24);

return MasiWebSrvsSupportUtil.sucessRet(reslutMap);

}

/\*\*

\* 更新代金券状态

\* @param recodMap

\*/

private void updateCoupon(Map<String, Object> recodMap)

{

JSONArray jSONArray = JSONArray.fromObject(recodMap.get("DBA180"));

for (Object object : jSONArray)

{

recodMap.put("DBA181", object);

update("updateCoupon", recodMap);

}

}

// 消息推送

private void push(Map<String, Object> params)

{

try

{

// 整单退费积分消息推送

Map<String, Object> res = getObject("getMemInfo", params);

if (Utility.notEmpty(res))

{

if (Utility.notNull(res.get("pushuid")))

{

if (Utility.notEmpty(String.valueOf(res.get("pushuid"))))

{

String message = "整单退费扣除:【" + params.get("DAA511")

+ "】分。";

int pushcode = MallPushUtil.managePush(

String.valueOf(res.get("pushuid")), message);

// 插入推送记录

PushLog plog = new PushLog();

plog.preInsert();

plog.setDaa001(String.valueOf(res.get("daa001")));

plog.setDaa003("1");

plog.setPushuid(String.valueOf(res.get("pushuid")));

plog.setDaa313("2");// 1-短信 2-手机APP

plog.setDba035(message);

plog.setDaa731(pushcode > 0 ? "success" : "fail");

plog.setDaa004(Utility.get19BitCurTimeAPP());

pushlogservice.save(plog);

}

}

}

} catch (Exception e)

{

LOG.error("整单退费消息推送失败", e);

}

}

}