源程序清单

农膜订单提交与抽奖管理系统 V1.0

package com.shenke.userforwx;

import com.shenke.userforwx.Configrations.ScheduleConf;

import com.shenke.userforwx.Service.impl.GuestInfoServiceimpl;

import com.shenke.userforwx.Utils.RandomUtil;

import com.shenke.userforwx.controller.SendAllController;

import org.apache.catalina.connector.Connector;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.EnableAutoConfiguration;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.boot.autoconfigure.jdbc.DataSourceAutoConfiguration;

import org.springframework.boot.web.embedded.tomcat.TomcatConnectorCustomizer;

import org.springframework.boot.web.embedded.tomcat.TomcatServletWebServerFactory;

import org.springframework.boot.web.servlet.server.ConfigurableServletWebServerFactory;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.Configuration;

import org.springframework.scheduling.annotation.EnableScheduling;

import org.springframework.scheduling.annotation.Scheduled;

import org.springframework.stereotype.Component;

import javax.annotation.Resource;

import java.time.LocalDateTime;

@SpringBootApplication

public class UserforwxApplication {

public final static Logger logger = LoggerFactory.getLogger(UserforwxApplication.class);

public static void main(String[] args) {

logger.info("提交&抽奖系统开始启动！！！！！！！！！");

SpringApplication.run(UserforwxApplication.class, args);

// logger.info(RandomUtil.getLongFormatRandom(3));

ScheduleConf sc=new ScheduleConf();

sc.GetAndWriteConfigFromFileIntoMemery();

sc.GetAndWriteProductionNameIntoMemery();

}

/\*\*

 \* 解决异常信息：

 \*  java.lang.IllegalArgumentException:

 \*  Invalid character found in the request target. The valid characters are defined in RFC 7230 and RFC 3986

 \* @return

 \*/

@Bean

public ConfigurableServletWebServerFactory webServerFactory() {

logger.info("配置内置tomcat，防止出现IllegalArgumentException");

TomcatServletWebServerFactory factory = new TomcatServletWebServerFactory();

factory.addConnectorCustomizers(new TomcatConnectorCustomizer() {

@Override

public void customize(Connector connector) {

connector.setProperty("relaxedQueryChars", "|{}[]");

}

});

return factory;

}

}

package com.shenke.userforwx.Configrations;

import okhttp3.ConnectionPool;

import okhttp3.OkHttpClient;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import java.util.concurrent.TimeUnit;

@Configuration

public class OKHttpConfig {

@Bean

public OkHttpClient okHttpClient() {

return new OkHttpClient().newBuilder().retryOnConnectionFailure(false).connectionPool(pool())

.connectTimeout(5, TimeUnit.SECONDS).readTimeout(5, TimeUnit.SECONDS).writeTimeout(5, TimeUnit.SECONDS)

.build();

}

@Bean

public ConnectionPool pool() {

return new ConnectionPool(50, 5, TimeUnit.MINUTES);

}

}

package com.shenke.userforwx.Configrations;

import com.shenke.userforwx.Model.Memery.ConfigFromFile;

import com.shenke.userforwx.Model.Memery.Production;

import com.shenke.userforwx.Service.impl.GuestInfoServiceimpl;

import com.shenke.userforwx.Utils.ConfigUtil;

import com.shenke.userforwx.Utils.OKHttpUtil;

import com.shenke.userforwx.controller.CommitController;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.context.annotation.Configuration;

import org.springframework.scheduling.annotation.EnableScheduling;

import org.springframework.scheduling.annotation.Scheduled;

import org.springframework.stereotype.Component;

import javax.annotation.Resource;

import java.time.LocalDateTime;

import java.util.List;

@Component

@Configuration //1.主要用于标记配置类，兼备Component的效果。

@EnableScheduling // 2.开启定时任务

public class ScheduleConf {

public final static Logger logger = LoggerFactory.getLogger(ScheduleConf.class);

@Resource

GuestInfoServiceimpl gisi;

@Scheduled(cron = "0 0 1 \* \* ?")//每天早上1点- "0 0 1 \* \* ?" 每20秒测试-"\*/5 \* \* \* \* ?"

public void reSetLoginFailCount() {

System.out.println("---- 执行 清除当日登录失败计数 任务时间: " + LocalDateTime.now());

gisi.reset();

}

@Scheduled(cron = "0 0/15 \* \* \* ?")//每天早上1点- "0 0 1 \* \* ?" 每20秒测试-"\*/5 \* \* \* \* ?"

public void GetAndWriteProductionNameIntoMemery() {

System.out.println("---- 执行 获取和写入产品名到内存 任务时间: " + LocalDateTime.now());

List<String> ls= ConfigUtil.findConfigFile();

String s = null;

try {

// s = HttpUtil.get("http://39.96.78.86/static/selectProduct");//比较耗时间，可能在关闭页面的时候才终止连接

if (ls!=null){

if (ls.size()>=1) {

System.out.println("获取产品名服务器地址从文件中获取，为 "+ls.get(0));

s= OKHttpUtil.get(ls.get(0),null);

}else{

System.out.println("配置文件中没有第0号元素,所以使用了默认地址： " +

"http://39.96.78.86/static/selectProduct");

s = OKHttpUtil.get("http://39.96.78.86/static/selectProduct", null);

}

}else {

System.out.println("未使用配置文件地址,使用了默认地址");

s = OKHttpUtil.get("http://39.96.78.86/static/selectProduct", null);

}

} catch (Exception e) {

logger.info("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!");

logger.info("---loadData--- 从erp服务器的数据库中获取产品名称失败！");

logger.info("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!");

}

Production.Json=s;

}

@Scheduled(cron = "0 0/15 \* \* \* ?")//每天早上1点- "0 0 1 \* \* ?" 每20秒测试-"\*/5 \* \* \* \* ?"

public void GetAndWriteConfigFromFileIntoMemery() {

System.out.println("---- 执行 获取和写入文件配置到内存 任务时间: " + LocalDateTime.now());

List<String> ls= ConfigUtil.findConfigFile();

ConfigFromFile.al=ls;

try {

System.out.println(ls.get(0));

} catch (Exception e) {

System.out.println("从文件到内存获取和写入配置失败");

}

}

}

package com.shenke.userforwx.controller;

import cn.hutool.json.JSONUtil;

import com.shenke.userforwx.Model.Memery.Production;

import com.shenke.userforwx.Utils.ConfigUtil;

import com.shenke.userforwx.Utils.IpUtil;

import com.shenke.userforwx.Utils.OKHttpUtil;

import com.shenke.userforwx.Utils.RandomUtil;

import com.shenke.userforwx.entity.GuestEnterFailedInfo;

import com.shenke.userforwx.entity.User;

import com.shenke.userforwx.respository.GuestInfoDao;

import com.shenke.userforwx.respository.UserDao;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Controller;

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

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

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import java.util.\*;

@Controller

public class CommitController {

public final static Logger logger = LoggerFactory.getLogger(CommitController.class);

@Resource

UserDao userDao;

@Resource

GuestInfoDao guestInfoDao;

@RequestMapping("/loginin")

public String loginin(){

//测试cookie

// Cookie[] cookie = request.getCookies();

// for(Cookie c : cookie){

// System.out.println((c.getName()+"---"+c.getValue()));

// }

return "abc/commit";

}

@ResponseBody

@RequestMapping("/loadData")

public Map<String, Object> loadDate(HttpServletRequest request) {

// List<String> ls=ConfigUtil.findConfigFile();

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

// String s = null;

// try {

//// s = HttpUtil.get("http://39.96.78.86/static/selectProduct");//比较耗时间，可能在关闭页面的时候才终止连接

// if (ls!=null){

// if (ls.size()>=1) {

// s= OKHttpUtil.get(ls.get(0),null);

// }else{

// System.out.println("配置文件中没有第0号元素,所以使用了默认地址");

// s = OKHttpUtil.get("http://39.96.78.86/static/selectProduct", null);

// }

// }else {

// System.out.println("未使用配置文件地址,使用了默认地址");

// s = OKHttpUtil.get("http://39.96.78.86/static/selectProduct", null);

// }

// } catch (Exception e) {

// logger.info("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!");

// logger.info("---loadData--- 从erp服务器的数据库中获取产品名称失败！");

// logger.info("!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!");

// }

// System.out.println(s);

map.put("success", true);

try {

// System.out.println("内存中读取json： "+production.getJson());

map.put("data", JSONUtil.parseArray(Production.Json));

} catch (Exception e) {

logger.info("-----------需解析的产品名json格式不正确！！！！");

// System.out.println("-----------需解析的产品名json格式不正确！！！！");

}

return map;

}

/\*\*

\* 登录相关

\* \*/

@ResponseBody

@RequestMapping("/commit")//处理登录页面点了登录按钮发来的信息，返回信息用于在前台跳转到提交订单页面

public String commitPage(String username,String password,HttpServletRequest request,

HttpServletResponse response){

String guestIp=IpUtil.getIpAddr(request);

List<GuestEnterFailedInfo> allByIpAddress = guestInfoDao.findAllByIpAddress(guestIp);

int random= RandomUtil.getFormatRandom();

int failedCountForToday=0;

if (!allByIpAddress.isEmpty()){

failedCountForToday = Integer.parseInt(allByIpAddress.get(0).getFailedCountForToday());

}

if (failedCountForToday>=10){

return "今日尝试次数过多，请改日再来！";

}

// logger.info(random+"=--------------------普通用户登录开始--------------------");

// logger.info("后台接收的登录输入：");

// logger.info("username: "+username);

// logger.info("password: "+password);

User u = userDao.findByUsername(username);

if (u==null){

// System.out.println("该用户尚未注册");

// logger.info("该用户尚未注册");

// logger.info(random+"=--------------------普通用户登录：失败--------------------");

/\*\*

\* 此时不进行log文件的打印，

\* 防止有人恶意提交用户名密码导致log文件过大或者无效信息过多

\* \*/

String failTime=new Date().toLocaleString();

System.out.println(new Date().toLocaleString()+" !!尝试登陆失败!!== -username-："+username+" -password-:"+password+

" ip= "+ guestIp);

GuestEnterFailedInfo gi=new GuestEnterFailedInfo(guestIp,failTime);

try {

if (!allByIpAddress.isEmpty()){

failedCountForToday = Integer.parseInt(allByIpAddress.get(0).getFailedCountForToday());

}

} catch (NumberFormatException e) {

logger.warn("数据库信息guest\_info-FailedCountForToday异常！！！");

}

gi.setFailedCountForToday(String.valueOf(++failedCountForToday));

gi.setLastEnterFailedTime(failTime);

if (!allByIpAddress.isEmpty()){

gi.setId(allByIpAddress.get(0).getId());

// System.out.println(allByIpAddress.get(0).toString());

}

guestInfoDao.save(gi);

System.out.println("当前ip："+guestIp+" 的尝试登陆失败（用户名错误）次数为 "+failedCountForToday);

return "该用户尚未注册";

}else {

logger.info(random+"=--------------------普通用户登录开始--------------------");

logger.info("后台接收的登录输入：");

logger.info("username: "+username);

logger.info("password: "+password);

String DAOusername = u.getUsername();

String DAOpassword = u.getPassword();

if (DAOusername.equals(username) && DAOpassword.equals(password)){

//这个判断是登陆验证

//登录状态信息存在session中

HttpSession session = request.getSession();

/\*\*

\* 保存登录状态信息

\* session:"userinf"

\* 0:username 1:password 2:clientname

\* \*/

ArrayList<String> userinf=new ArrayList<>();

userinf.add(DAOusername);

userinf.add(DAOpassword);

userinf.add(u.getClientname());

session.setAttribute("userinf",userinf);

//session生命周期（秒）

session.setMaxInactiveInterval(1800);

String sessionId = session.getId();

// System.out.println("sessionID= "+sessionId);

logger.info("sessionID= "+sessionId);

// //创建cookie

// Cookie nameCookie = new Cookie(DAOusername, DAOpassword);

// nameCookie.setPath(request.getContextPath()+"/");//设置cookie路径

// //设置cookie保存的时间 单位：秒

// nameCookie.setMaxAge(7\*24\*60\*60);

// //将cookie添加到响应

// response.addCookie(nameCookie);

//

// System.out.println("登录成功");

logger.info(random+"--------------------普通用户登录：成功--------------------");

return "登录成功";

}else {

// System.out.println("用户名或密码错误");

logger.info("用户名或密码错误");

logger.info(random+"--------------------普通用户登录：失败--------------------");

return "用户名或密码错误";

}

}

}

}

package com.shenke.userforwx.controller;

import com.shenke.userforwx.Utils.RandomUtil;

import com.shenke.userforwx.entity.CommitOrder;

import com.shenke.userforwx.respository.CommitOrderDao;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Controller;

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

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

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import java.util.ArrayList;

import java.util.List;

@Controller

public class HistoryDetailController {

public final static Logger logger = LoggerFactory.getLogger(HistoryDetailController.class);

@Resource

CommitOrderDao commitOrderDao;

@ResponseBody

@RequestMapping("/getAllRecordByHavesent")

public List<List<CommitOrder>> getAllRecordByHavesent(HttpServletRequest request){

int random= RandomUtil.getFormatRandom();

logger.info(random+"-------------------获取havesentTAG的订单数据并格式化：开始----------------------");

HttpSession session = request.getSession();

ArrayList<String> userinf=

(ArrayList<String>) session.getAttribute("userinf");

String username = null;

try {

username = userinf.get(0);

} catch (NullPointerException e) {

// System.out.println("！！！！登陆状态已过时，正在跳转回登陆页面！！！！");

logger.info("！！！！登陆状态已过时，正在跳转回登陆页面！！！！");

logger.info(random+"-------------------获取havesentTAG的订单数据并格式化：失败----------------------");

// e.printStackTrace();//不在控制台打印异常

}

List<List<CommitOrder>> list\_needtoReturn=new ArrayList<>();

List<CommitOrder> allOrder = commitOrderDao.findDifferentChangduShuliang(username,userinf.get(3));

logger.info("=====该历史详情大框数目为===== "+allOrder.size());

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

CommitOrder commitOrder = allOrder.get(i);

String beizhu = commitOrder.getBeizhu();

String chanpin = commitOrder.getChanpin();

String fukuan = commitOrder.getFukuan();

String houdu = commitOrder.getHoudu();

String yanse = commitOrder.getYanse();

String nonghu = commitOrder.getNonghu();

List<CommitOrder> allByChangduAndShuliang =

commitOrderDao.findAllByChangduAndShuliang(beizhu, chanpin,fukuan,houdu,yanse,username,nonghu,userinf.get(3));

// System.out.println("第 "+(i+1)+"项的子项数为 "+allByChangduAndShuliang.size());

logger.info("第"+(i+1)+"项的子项数为 "+allByChangduAndShuliang.size());

list\_needtoReturn.add(allByChangduAndShuliang);

}

// System.out.println("-------------------获取全部订单数据并格式化完成----------------------");

logger.info(random+"-------------------获取havesentTAG的订单数据并格式化：成功----------------------");

return list\_needtoReturn;

}

}

package com.shenke.userforwx.controller;

import com.shenke.userforwx.Utils.RandomUtil;

import com.shenke.userforwx.Utils.ShowUsernameUtil;

import com.shenke.userforwx.entity.CommitOrder;

import com.shenke.userforwx.respository.CommitOrderDao;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Controller;

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

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

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import java.util.ArrayList;

import java.util.List;

@Controller

public class HistoryOrderController {

public final static Logger logger = LoggerFactory.getLogger(HistoryOrderController.class);

@Resource

CommitOrderDao commitOrderDao;

@RequestMapping("/sendingHistory")

public String sendingHistory(){

return "abc/sendhistory";

}

@RequestMapping("/turnToKuwashiiHistory")

public String turnToKuwashiiHistory(HttpServletRequest request,String havesent){

//使用userinf[3]保存havesent这个标签值

// System.out.println("ahahhaahahahhahahahahahahahahaha"+havesent);

// return "...";

int random=RandomUtil.getFormatRandom();

logger.info(random+"-------------------追加session的havesentTAG：开始----------------------");

HttpSession session = request.getSession();

ArrayList<String> userinf = (ArrayList<String>) session.getAttribute("userinf");

if (userinf==null){

logger.info("获取当前登陆状态失败");

logger.info(random+"-------------------追加session的havesentTAG：失败----------------------");

return "login";

}

logger.info("保存用户 "+userinf.get(0)+" 的查看发送详情havesentTAG= "+havesent);

//不能直接用用add！！！！！！！

if (userinf.size() <= 3) {

userinf.add(havesent);

}else {

userinf.set(3,havesent);

}

logger.info(random+"-------------------追加session的havesentTAG：成功----------------------");

return "abc/sending\_history\_detail";

}

@ResponseBody

@RequestMapping("/getAllSentOrderSimple")

public List<List<String>> getAllSentOrderSimple(HttpServletRequest request){

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

int random = RandomUtil.getFormatRandom();

logger.info(random+"-----------------获取所有已发送数据：开始-------------------");

String username = ShowUsernameUtil.getUsername(request, logger);

List<CommitOrder> allNeedToSendOrder = commitOrderDao.findAllSentOrderGroupBySENTTAG(username);

logger.info("获取到的 "+username+" 用户所有已发送数据");

logger.info("大框数======"+String.valueOf(allNeedToSendOrder.size()));

// try {

// logger.info(allNeedToSendOrder.get(0).toString());

// } catch (Exception e) {

// logger.info("获取失败");

// }

//取出上一步查询的到的所有记录的havesend值，有多少记录就有多少大框

//用havesend值依次查记录

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

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

String havesent = allNeedToSendOrder.get(i).getHavesent();

String sendTime=allNeedToSendOrder.get(i).getSend\_time();

List<CommitOrder> allByHavesent = commitOrderDao.findAllByHavesent(username, havesent);

int countOfSend=allByHavesent.size();

lss.add(sendTime);lss.add("发送数量为 "+countOfSend+"次");lss.add(havesent);

ls.add(lss);

}

logger.info(random+"-----------------获取所有已发送数据：成功-------------------");

return ls;

}

}

package com.shenke.userforwx.controller;

import cn.hutool.core.io.FileUtil;

import cn.hutool.core.io.file.FileReader;

import cn.hutool.core.util.CharsetUtil;

import com.shenke.userforwx.Utils.IpUtil;

import com.shenke.userforwx.Utils.PathUtil;

import com.shenke.userforwx.entity.GuestEnterFailedInfo;

import com.shenke.userforwx.respository.GuestInfoDao;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Controller;

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

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

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import java.io.File;

import java.util.List;

@Controller

public class IndexController {

public final static Logger logger = LoggerFactory.getLogger(IndexController.class);

@Resource

GuestInfoDao guestInfoDao;

// /\*\*

// \* 用thymeleaf

// \* 在properties已经设置了目录，也不需要写.html

// \* 如果需要不经过controller访问html那就放置于static文件夹中直接输入要访问的资源名称即可

// \* \*/

// @RequestMapping("/index")

// public String ind(){

// System.out.println("index=====");

//// templates/commit.html

// return "templates/commit.html";

// }

@RequestMapping("/")

public String index1(HttpServletRequest request) {

// FileReader frBenji=new FileReader(FileUtil.file("C:\\Users\\duoduo\\Desktop\\123\\config\\config.txt"),

// CharsetUtil.systemCharset());

// FileReader fr=new FileReader(FileUtil.file("C:\\Users\\Administrator\\Desktop\\222\\config\\config.txt"),

// CharsetUtil.systemCharset());

// List<String> ls=frBenji.readLines();

// System.out.println("文件中读到的=== ");

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

// System.out.println(ls.get(i));

// }

String guestIp= IpUtil.getIpAddr(request);

List<GuestEnterFailedInfo> allByIpAddress = guestInfoDao.findAllByIpAddress(guestIp);

if (allByIpAddress.size()!=0){

GuestEnterFailedInfo guestEnterFailedInfo = allByIpAddress.get(0);

int FailedCountForToday=0;

try {

FailedCountForToday = Integer.parseInt(guestEnterFailedInfo.getFailedCountForToday());

} catch (NumberFormatException e) {

logger.warn("数据库信息guest\_info-FailedCountForToday异常！！！");

}

if (FailedCountForToday>=10){

return "abc/hello";

}

}

return "login";

}

@RequestMapping("/index")

public String index2() {

return "login";

}

@RequestMapping("/hello")//测试用页面以及方法，并无实际业务作用

public String hello() {

return "abc/hello";

}

@ResponseBody

@RequestMapping("/huoqu")

public String huoqu(HttpServletRequest request) {

return "用户名：" + request.getServletContext().getAttribute("username")+"======"+"密码："+request.getServletContext().getAttribute("password");

}

}

package com.shenke.userforwx.controller;

import com.shenke.userforwx.Utils.RandomUtil;

import com.shenke.userforwx.Utils.ShowUsernameUtil;

import com.shenke.userforwx.entity.LotteryReward;

import com.shenke.userforwx.entity.User;

import com.shenke.userforwx.respository.LotteryRewardDao;

import com.shenke.userforwx.respository.UserDao;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Controller;

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

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

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import java.util.ArrayList;

import java.util.Date;

@Controller

public class LotteryController {

public final static Logger logger = LoggerFactory.getLogger(LotteryController.class);

@Resource

LotteryRewardDao lotteryRewardDao;

@Resource

UserDao UserDao;

@RequestMapping("/lottery")

public String openLotteryPage(){

return "abc/lottery";

}

@RequestMapping("/indexoflottery")

public String registerForLottery(){

return "loginlottery";

}

@ResponseBody

@RequestMapping("/checklotterytime")

public String checkLotteryTime(HttpServletRequest request){

// System.out.println("验证该账号抽奖次数中-------");

int random= RandomUtil.getFormatRandom();

logger.info(random+"----------------------验证该账号抽奖：开始---------------------");

HttpSession session = request.getSession();

ArrayList<String> userinf=

(ArrayList<String>) session.getAttribute("userinf");

//

// if(userinf==null || userinf.size()==0){

//// System.out.println("/checklotterytime: 用户尚未登陆或者用户信息不存在");

// logger.info("用户尚未登陆或者用户信息不存在");

// return "用户尚未登陆或者用户信息不存在";

// }

String s = ShowUsernameUtil.getUsername(request, logger);

if (s.equals("请先登录再进行提交")){

logger.info(random+"----------------------验证该账号抽奖：失败---------------------");

return "请先登录再进行提交";

}

User currentUser = UserDao.findByUsername(userinf.get(0));

int choujiangcishu = currentUser.getChoujiangcishu();

// System.out.println("当前用户的抽奖次数为： "+choujiangcishu+" 次");

logger.info("当前用户的抽奖次数为： "+choujiangcishu+" 次");

logger.info(random+"----------------------验证该账号抽奖：成功---------------------");

return String.valueOf(choujiangcishu);//返回抽奖次数

}

@ResponseBody

@RequestMapping("/savelotteryresult")

public String saveReward(HttpServletRequest request,String reward){

// System.out.println("---------开始保存用户的抽奖结果-------");

int random= RandomUtil.getFormatRandom();

logger.info(random+"------------------保存用户的抽奖结果：开始-------------------");

HttpSession session = request.getSession();

ArrayList<String> userinf=

(ArrayList<String>) session.getAttribute("userinf");

// System.out.println("----------用户是： "+userinf.get(0)+" -----------");

String s = ShowUsernameUtil.getUsername(request, logger);

if (s.equals("请先登录再进行提交")){

logger.info(random+"----------------------保存用户的抽奖结果：失败---------------------");

return "请先登录再进行提交";

}

// System.out.println("----------reward是： "+reward+" -----------");

logger.info("=========从前台获取的抽取奖励是： "+reward+"==========");

if (userinf.get(0)==null || reward==null ||

userinf.get(0).equals("") || reward.equals("")) {//将来应该加验证机制

// System.out.println("抽奖次数或账号认证失败");

logger.info("抽奖次数或账号认证失败");

logger.info(random+"------------------保存用户的抽奖结果：失败-------------------");

return "抽奖次数或账号认证失败";

}

// //抽奖次数减少一次

// User currentUser = UserDao.findByUsername(userinf.get(0));

// int choujiangcishu = currentUser.getChoujiangcishu();

// currentUser.setChoujiangcishu(--choujiangcishu);

// UserDao.save(currentUser);

LotteryReward lr=new LotteryReward();

lr.setUsername(userinf.get(0));

lr.setReward(reward);

lr.setCreate\_lottery\_time(new Date().toLocaleString());

lotteryRewardDao.save(lr);

// System.out.println("保存奖励成功! 用户为： "+userinf.get(0)+"奖励为: "+reward);

logger.info("保存奖励成功! 用户为： "+userinf.get(0)+"奖励为: "+reward);

logger.info(random+"------------------保存用户的抽奖结果：成功！-------------------");

return "保存奖励成功";

}

@ResponseBody

@RequestMapping("/reducechoujiangcishu")

public String reduceChoujiangTime(HttpServletRequest request){

// System.out.println("---------开始减少抽奖次数-------");

int random= RandomUtil.getFormatRandom();

logger.info(random+"------------------减少抽奖次数：开始----------------------");

HttpSession session = request.getSession();

ArrayList<String> userinf=

(ArrayList<String>) session.getAttribute("userinf");

ShowUsernameUtil.getUsername(request,logger);

//抽奖次数减少一次

User currentUser = UserDao.findByUsername(userinf.get(0));

int choujiangcishu = currentUser.getChoujiangcishu();

currentUser.setChoujiangcishu(--choujiangcishu);

UserDao.save(currentUser);

logger.info(random+"------------------减少抽奖次数：成功----------------------");

return "当前用户抽奖次数减少一次";

}

@RequestMapping("/lottery\_manage")

public String turnToManageLottery(){

return "abc/lottery\_manage";

}

}

package com.shenke.userforwx.controller;

import com.shenke.userforwx.Utils.RandomUtil;

import com.shenke.userforwx.Utils.ShowUsernameUtil;

import com.shenke.userforwx.entity.LotteryReward;

import com.shenke.userforwx.entity.User;

import com.shenke.userforwx.respository.LotteryRewardDao;

import com.shenke.userforwx.respository.UserDao;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Controller;

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

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

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import java.util.ArrayList;

import java.util.List;

@Controller

public class ManageLotteryController {

public final static Logger logger = LoggerFactory.getLogger(ManageLotteryController.class);

@Resource

LotteryRewardDao lotteryRewardDao;

@Resource

UserDao userDao;

@ResponseBody

@RequestMapping("/getalllotterybyusername")

public List<LotteryReward> getAllLotteryByUsername(HttpServletRequest request){

// System.out.println("------------------开始获取该用户所有奖励---------------------");

int random= RandomUtil.getFormatRandom();

logger.info(random+"------------------开始获取该用户所有奖励---------------------");

HttpSession session = request.getSession();

ArrayList<String> userinf=

(ArrayList<String>) session.getAttribute("userinf");

String username = userinf.get(0);

ShowUsernameUtil.getUsername(request,logger);

//从LotteryReward表中根据 用户名 获取 该用户的所有奖励

List<LotteryReward> rewards = lotteryRewardDao.findByUsername(username);

// System.out.println("------------------获取该用户所有奖励：成功---------------------");

logger.info(random+"------------------获取该用户所有奖励：成功---------------------");

return rewards;

}

@ResponseBody

@RequestMapping("/getaddress")

public String getAddress(HttpServletRequest request){

// System.out.println("------------------开始获取该用户地址---------------------");

int random= RandomUtil.getFormatRandom();

logger.info(random+"------------------获取该用户地址:开始---------------------");

HttpSession session = request.getSession();

ArrayList<String> userinf=

(ArrayList<String>) session.getAttribute("userinf");

// if (userinf==null){

//// System.out.println("获取用户信息失败");

// logger.info("获取用户信息失败");

// return "获取用户信息失败";

// }

// if (userinf.isEmpty()){

//// System.out.println("获取用户信息失败");

// logger.info("获取用户信息失败");

// return "获取用户信息失败";

// }

String s = ShowUsernameUtil.getUsername(request, logger);

if (s.equals("请先登录再进行提交")){

logger.info(random+"----------------------获取该用户地址：失败---------------------");

return "请先登录再进行提交";

}

String username = userinf.get(0);

User user = userDao.findByUsername(username);

if (user==null){

// System.out.println("数据库中未找到该用户信息");

logger.info("数据库中未找到该用户信息");

logger.info(random+"----------------------获取该用户地址：失败---------------------");

return "数据库中未找到该用户信息";

}

String address=user.getAddress();

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

// System.out.println("请设置您的收获地址");

logger.info("请设置您的收获地址");

logger.info(random+"----------------------获取该用户地址：失败---------------------");

return "请设置您的收获地址";

}

// System.out.println("获取到用户 "+userinf.get(0)+" 的地址为 "+address);

// System.out.println("------------------获取该用户地址:成功！---------------------");

logger.info("获取到用户 "+userinf.get(0)+" 的地址为 "+address);

logger.info(random+"------------------获取该用户地址:成功！---------------------");

return address;

}

@ResponseBody

@RequestMapping("/setaddress")

public String setAddress(HttpServletRequest request,String address){

// System.out.println("------------------开始修改该用户地址---------------------");

int random= RandomUtil.getFormatRandom();

logger.info(random+"------------------修改该用户地址:开始---------------------");

HttpSession session = request.getSession();

ArrayList<String> userinf=

(ArrayList<String>) session.getAttribute("userinf");

// if (userinf.isEmpty()){

//// System.out.println("获取用户信息失败");

// logger.info("获取用户信息失败");

// return "获取用户信息失败";

// }

// if (userinf==null || userinf.equals("")){

//// System.out.println("获取用户名失败");

// logger.info("获取用户名失败");

// return "获取用户名失败";

// }

String s = ShowUsernameUtil.getUsername(request, logger);

if (s.equals("请先登录再进行提交")){

logger.info(random+"----------------------修改该用户地址：失败---------------------");

return "请先登录再进行提交";

}

String username = userinf.get(0);

User user = userDao.findByUsername(username);

if (user==null){

// System.out.println("数据库中未找到该用户信息");

logger.info("数据库中未找到该用户信息");

logger.info(random+"----------------------修改该用户地址：失败---------------------");

return "数据库中未找到该用户信息";

}

// System.out.println("用户 "+username+" 的新地址为： "+address);

logger.info("用户 "+username+" 的新地址为： "+address);

user.setUsername(username);

user.setAddress(address);

userDao.save(user);

// System.out.println("------------------修改该用户地址：成功---------------------");

logger.info(random+"------------------修改该用户地址：成功---------------------");

return "修改领奖地址成功";

}

}

package com.shenke.userforwx.controller;

import com.shenke.userforwx.Utils.RandomUtil;

import com.shenke.userforwx.Utils.ShowUsernameUtil;

import com.shenke.userforwx.entity.CommitOrder;

import com.shenke.userforwx.respository.CommitOrderDao;

import com.shenke.userforwx.respository.UserDao;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Controller;

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

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

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import java.util.ArrayList;

import java.util.Date;

@Controller

public class OrderController {

public final static Logger logger = LoggerFactory.getLogger(OrderController.class);

@Resource

UserDao userDao;

@Resource

CommitOrderDao commitOrderDao;//订单表查询dao

/\*\*

\* 验证当前登录状态

\* \*/

@ResponseBody

@RequestMapping("/yanzhengstatus")

public String yanzhengstatus(HttpServletRequest request){

// System.out.println("验证登录状态中-------");

int random= RandomUtil.getFormatRandom();

logger.info(random+"----------------验证登录状态:开始---------------");

HttpSession session = request.getSession();

ArrayList<String> userinf=

(ArrayList<String>) session.getAttribute("userinf");

if (userinf==null){

// System.out.println("验证登陆状态失败： 请先登录再进行提交");

logger.info("验证登陆状态失败： 请先登录再进行提交");

logger.info(random+"----------------验证登录状态:失败---------------");

return "请先登录再进行提交";

}else {

// System.out.println("验证登陆状态成功： 当前登录用户的clientname： "+userinf.get(2));

logger.info("当前执行操作的 普通用户 的Username： "+userinf.get(0));

logger.info("当前执行操作的 普通用户 的clientname： "+userinf.get(2));

logger.info(random+"----------------验证登录状态:成功---------------");

return userinf.get(2);//返回clientname用于显示代理商名（用的字段是clientname）

}

}

/\*\*

\* 退出当前登录

\* 并不是销毁特定session

\* 而是取出session后把其中对应的userinf的键值改为空值

\* \*/

@ResponseBody

@RequestMapping("/logoff")

public String LogOff(HttpServletRequest request){

int random= RandomUtil.getFormatRandom();

logger.info(random+"------------------------登出账号:开始------------------------");

HttpSession session = request.getSession();

ArrayList<String> userinf=

(ArrayList<String>) session.getAttribute("userinf");

// System.out.println("当前登录用户------ "+userinf.get(0)+"，现在准备登出---------");

logger.info("当前登录用户========= "+userinf.get(0)+"，现在准备登出。。。。。。");

session.setAttribute("userinf",null);

// System.out.println("------------------------登出账号成功------------------------");

logger.info(random+"------------------------登出账号:成功------------------------");

return "";

}

/\*\*

\* 提交当前框内的记录

\* \*/

@ResponseBody

@RequestMapping("/commitorder")

public String commitresult(HttpServletRequest request,CommitOrder order) {

// CommitOrder user = commitOrderDao.findByUsername(username);//根据username查出一条commit\_order表记录

int random= RandomUtil.getFormatRandom();

logger.info(random+"------------------提交订单:开始-------------------");

HttpSession session = request.getSession();

ArrayList<String> userinf=

(ArrayList<String>) session.getAttribute("userinf");

// if (userinf==null) {

//// System.out.println("请先登录再进行提交");

// logger.info("请先登录再进行提交");

// return "请先登录再进行提交";

// }

// String username=userinf.get(0);

// if (username==null) {

//// System.out.println("请先登录再进行提交");

// logger.info("请先登录再进行提交");

// return "请先登录再进行提交";

// }

String s = ShowUsernameUtil.getUsername(request, logger);

if (s.equals("请先登录再进行提交")){

logger.info(random+"------------------提交订单:失败-------------------");

}

String username=userinf.get(0);

order.setUsername(username);//把所属用户也添加到记录上

// if (order.isReadyToSubmit()) {

order.setCreate\_order\_time(new Date().toLocaleString());

commitOrderDao.save(order);

// }else {

// return "所有信息填写全才能提交";

// }

// System.out.println("成功提交： ----------------------");

logger.info("成功提交以下记录：");

// System.out.println(order.toString());

logger.info(order.toString());

logger.info(random+"-----------------提交订单:失败--------------------");

return "提交本条记录成功!";

}

}

package com.shenke.userforwx.controller;

import com.shenke.userforwx.Utils.RandomUtil;

import com.shenke.userforwx.entity.User;

import com.shenke.userforwx.respository.UserDao;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Controller;

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

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

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import java.util.Date;

@Controller

public class RegisterController {

public final static Logger logger = LoggerFactory.getLogger(RegisterController.class);

@Resource

UserDao userDao;

/\*\*

\* 注册相关

\* \*/

@RequestMapping("/reg")//在登录页面点击 注册 按钮后跳转到注册页面

public String reg(){

return "abc/register";

}

@ResponseBody

@RequestMapping("/sendregister")//在注册页面页面点击 注册 按钮进行数据库查询

public String sendreg(User inputbean, String passwordagain, HttpServletRequest request){

int random= RandomUtil.getFormatRandom();

logger.info(random+"----------------------注册：开始----------------------------");

User u = userDao.findByUsername(inputbean.getUsername());

if (u==null){

// System.out.println(inputbean);

logger.info("提交的注册信息：");

logger.info(inputbean.toString());

User userdo=new User();

if (inputbean.getUsername()==null ||

inputbean.getPassword()==null || inputbean.getClientname()==null){

// System.out.println("注册信息填写不完整");

logger.info("注册信息填写不完整");

logger.info(random+"----------------------注册：失败----------------------------");

return "注册信息填写不完整";

}

userdo.setUsername(inputbean.getUsername());

userdo.setPassword(inputbean.getPassword());

userdo.setClientname(inputbean.getClientname());

userdo.setCreateuser\_time(new Date().toLocaleString());

userDao.save(userdo);

// System.out.println("注册成功");

// logger.info("注册成功");

logger.info(random+"----------------------注册：成功----------------------------");

return "注册成功";

}else{

// System.out.println("当前用户名已经重复");

logger.info("当前用户名已经重复");

logger.info(random+"----------------------注册：失败----------------------------");

return "当前用户名已经重复";

}

}

}

package com.shenke.userforwx.controller;

import cn.hutool.json.JSONArray;

import cn.hutool.json.JSONUtil;

import com.shenke.userforwx.Model.Memery.ConfigFromFile;

import com.shenke.userforwx.Utils.ConfigUtil;

import com.shenke.userforwx.Utils.OKHttpUtil;

import com.shenke.userforwx.Utils.RandomUtil;

import com.shenke.userforwx.Utils.ShowUsernameUtil;

import com.shenke.userforwx.entity.CommitOrder;

import com.shenke.userforwx.respository.CommitOrderDao;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Controller;

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

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

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

@Controller

public class SendAllController {

public final static Logger logger = LoggerFactory.getLogger(SendAllController.class);

@Resource

CommitOrderDao commitOrderDao;

@RequestMapping("/sendall")

public String comeIntoSendAll(){

// System.out.println("跳转到sendall页面");

// logger.info("跳转到sendall页面");

return "abc/sendallorder";

}

@ResponseBody

@RequestMapping("/getallorderbyusername")

public List<List<CommitOrder>> getAllOrder(HttpServletRequest request){//从session中获取username

// System.out.println("-------------------获取全部订单数据并格式化开始----------------------");

int random= RandomUtil.getFormatRandom();

logger.info(random+"-------------------获取全部订单数据并格式化：开始----------------------");

HttpSession session = request.getSession();

ArrayList<String> userinf=

(ArrayList<String>) session.getAttribute("userinf");

String username = null;

try {

username = userinf.get(0);

} catch (NullPointerException e) {

// System.out.println("！！！！登陆状态已过时，正在跳转回登陆页面！！！！");

logger.info("！！！！登陆状态已过时，正在跳转回登陆页面！！！！");

logger.info(random+"-------------------获取全部订单数据并格式化：失败----------------------");

// e.printStackTrace();//不在控制台打印异常

}

List<List<CommitOrder>> list\_needtoReturn=new ArrayList<>();

List<CommitOrder> allOrder = commitOrderDao.findDifferentChangduShuliang(username,"0");

// System.out.println("-----大框数目为----- "+allOrder.size());

logger.info("=====大框数目为===== "+allOrder.size());

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

CommitOrder commitOrder = allOrder.get(i);

String beizhu = commitOrder.getBeizhu();

String chanpin = commitOrder.getChanpin();

String fukuan = commitOrder.getFukuan();

String houdu = commitOrder.getHoudu();

String yanse = commitOrder.getYanse();

String nonghu = commitOrder.getNonghu();

List<CommitOrder> allByChangduAndShuliang =

commitOrderDao.findAllByChangduAndShuliang(beizhu, chanpin,fukuan,houdu,yanse,username,nonghu,"0");

// System.out.println("第 "+(i+1)+"项的子项数为 "+allByChangduAndShuliang.size());

logger.info("第 "+(i+1)+"项的子项数为 "+allByChangduAndShuliang.size());

list\_needtoReturn.add(allByChangduAndShuliang);

}

// System.out.println("-------------------获取全部订单数据并格式化完成----------------------");

logger.info(random+"-------------------获取全部订单数据并格式化：成功----------------------");

return list\_needtoReturn;

}

@ResponseBody//不用thymeleaf而是返回字符串要加该注解

@RequestMapping("/deletethisrecord")

public String deleteThisRecord(HttpServletRequest request,int id){

// System.out.println("后台准备删除id为： "+id+" 的记录");

int random= RandomUtil.getFormatRandom();

logger.info(random+"------------------后台准备删除id为： "+id+" 的记录-----------------");

CommitOrder needToDelete=commitOrderDao.findRecordById(id);

if (needToDelete==null){

// System.out.println("库中未找到该id的记录");

logger.info("库中未找到该id的记录");

logger.info(random+"--------------------删除记录"+id+" 失败------------------------");

return "库中未找到该id的记录";

}

commitOrderDao.delete(needToDelete);

// System.out.println("后台成功删除id为： "+id+" 的记录");

logger.info(random+"-------------------后台成功删除id为： "+id+" 的记录---------------------");

return "后台成功删除id为： "+id+" 的记录";

}

@ResponseBody

@RequestMapping("/changethisrecord")

public String ChangeThisRecord(HttpServletRequest request,CommitOrder order){

int random= RandomUtil.getFormatRandom();

logger.info(random+"-----------------修改记录：开始--------------------");

HttpSession session = request.getSession();

ArrayList<String> userinf=

(ArrayList<String>) session.getAttribute("userinf");

// if (userinf==null) {

//// System.out.println("请先登录再进行修改");

// logger.info("请先登录再进行修改");

// return "请先登录再进行修改";

// }

ShowUsernameUtil.getUsername(request,logger);

String username=userinf.get(0);

if (username==null) {

// System.out.println("请先登录再进行修改");

logger.info("请先登录再进行修改");

logger.info(random+"-----------------修改记录：失败--------------------");

return "请先登录再进行修改";

}

order.setUsername(username);//把所属用户也添加到记录上

// if (order.isReadyToSubmit()) {

order.setCreate\_order\_time(new Date().toLocaleString());

commitOrderDao.save(order);

// System.out.println("成功修改： ----------------------");

logger.info("成功修改为传入的记录：=======");

// System.out.println(order.toString());

logger.info(order.toString());

logger.info(random+"-----------------修改记录：成功--------------------");

return "修改本条记录成功!";

}

@ResponseBody

@RequestMapping("sendAllOrder")

public String sendAllOrder(HttpServletRequest request){

int random = RandomUtil.getFormatRandom();

String randomForThisSend=RandomUtil.getLongFormatRandom(3);

logger.info(random+"-----------------发送该页所有订单：开始-------------------");

String username = ShowUsernameUtil.getUsername(request, logger);

List<CommitOrder> allNeedToSendOrder =

commitOrderDao.findAllNeedToSendOrder(username);

logger.info("==开始向农膜系统发送这些订单==");

JSONArray needToSendJsonArray = JSONUtil.parseArray(allNeedToSendOrder);

System.out.println(needToSendJsonArray);

List<String> ls= ConfigFromFile.al;

try {

// OKHttpUtil.postJsonParams("http://localhost:8080/shenke/userForWx/testSending",

// needToSendJsonArray.toString());

if (ls!=null){

if (ls.size()>=2) {

OKHttpUtil.postJsonParams(ls.get(1),

needToSendJsonArray.toString());

}else{

System.out.println("配置文件中没有第1号元素,所以使用了默认地址");

OKHttpUtil.postJsonParams("",

needToSendJsonArray.toString());

}

}else {

System.out.println("未使用配置文件地址,使用了默认地址");

OKHttpUtil.postJsonParams("",

needToSendJsonArray.toString());

}

} catch (Exception e) {

logger.info("!!!!数据发送异常!!!!");

logger.info(random+"-----------------发送该页所有订单：失败-------------------");

return "发送订单失败";

}

//修改这些订单的发送时间和havesent这个标识(使用randomForThisSend)

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

allNeedToSendOrder.get(i).setSend\_time(new Date().toLocaleString());

allNeedToSendOrder.get(i).setHavesent(randomForThisSend);

commitOrderDao.save(allNeedToSendOrder.get(i));

}

logger.info(random+"-----------------发送该页所有订单：成功-------------------");

return "发送订单成功";

}

}

package com.shenke.userforwx.controller.admin;

import com.shenke.userforwx.Utils.RandomUtil;

import com.shenke.userforwx.entity.User;

import com.shenke.userforwx.respository.UserDao;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Controller;

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

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

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import java.util.ArrayList;

@Controller

public class LoginAdminController {

public final static Logger logger = LoggerFactory.getLogger(LotteryItemController.class);

@Resource

UserDao userDao;

@RequestMapping("/admin")

public String adminPage(){

return "admin/login\_lottery\_admin";

}

@RequestMapping("/admin/lottery")

public String manageLottery(){

return "admin/login\_lottery\_admin";

}

@ResponseBody

@RequestMapping("/admin/checkduty")

public String LoginAdminCheckDuty(String username, String password, HttpServletRequest request) {

// System.out.println("后台处理管理员登录请求:");

// System.out.println("username: " + username);

// System.out.println("password: " + password);

int random= RandomUtil.getFormatRandom();

logger.info(random+"-----------------------管理员登录------------------------");

logger.info("后台接受管理员登录请求-----输入信息为：");

logger.info("username: " + username);

logger.info("password: " + password);

User u = userDao.findByUsername(username);

if (u == null) {

// System.out.println("该用户尚未注册");

logger.info("该（用户/管理员）尚未注册");

logger.info(random+"------------------------管理员登录:失败！------------------------");

return "该用户尚未注册";

} else {

String DAOusername = u.getUsername();

String DAOpassword = u.getPassword();

String DAOduty=u.getDuty();

if (DAOusername.equals(username) && DAOpassword.equals(password)

&& DAOduty.equals("admin")) {

//登录状态信息存在session中

HttpSession session = request.getSession();

/\*\*

\* 保存管理员登录状态信息到adminuserinf这个session的list中

\* session:"userinf"

\* 0:username 1:password 2:clientname

\* \*/

ArrayList<String> userinf = new ArrayList<>();

userinf.add(DAOusername);

userinf.add(DAOpassword);

userinf.add(u.getClientname());

session.setAttribute("adminuserinf", userinf);

// System.out.println("登录成功");

logger.info(random+"------------------------管理员 "+userinf.get(0)+" 登录：成功------------------------");

return "登录成功";

} else {

// System.out.println("用户类型错误也可能是用户名或密码错误");

logger.info("用户类型错误也可能是用户名或密码错误");

logger.info(random+"------------------------管理员登录:失败！------------------------");

return "用户类型错误也可能是用户名或密码错误";

}

}

}

@RequestMapping("/admin/turntolotterymanage")

public String turntoLotteryManage(){

return "admin/manage\_lotteryitem";

}

}

package com.shenke.userforwx.controller.admin;

import com.shenke.userforwx.Utils.RandomUtil;

import com.shenke.userforwx.Utils.ShowUsernameUtil;

import com.shenke.userforwx.entity.LotteryItem;

import com.shenke.userforwx.respository.LotteryItemDao;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Controller;

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

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

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

@Controller

public class LotteryItemController {

public final static Logger logger = LoggerFactory.getLogger(LotteryItemController.class);

@Resource

LotteryItemDao lotteryItemDao;

@ResponseBody

@RequestMapping("/admin/getitems")

public List<LotteryItem> getLotterItem(HttpServletRequest request){

// System.out.println("---------------返回所有奖励-----------------");

int random= RandomUtil.getFormatRandom();

logger.info(random+"------------------返回所有奖项------------------------");

List<LotteryItem> allLotteryItems = lotteryItemDao.findAll();

// System.out.println("----------可抽奖项个数： "+allLotteryItems.size());

// System.out.println("---------------返回所有奖励结束-----------------");

String s = ShowUsernameUtil.getAdminUsername(request, logger);

if (s.equals("请先登录再进行提交")){

logger.info(random+"----------------返回所有奖项：失败！--------------------");

}

logger.info(random+"----------------返回所有奖项：成功！--------------------");

return allLotteryItems;

}

@ResponseBody

@RequestMapping("/admin/setitems")

public String setLotterItem(HttpServletRequest request,String item,String type){//其中非空字段一定要添加

//该函数返回值将传给页面用于显示提示

// System.out.println("---------------开始设置奖励项-----------------");

int random= RandomUtil.getFormatRandom();

logger.info(random+"---------------开始设置奖励项-----------------");

HttpSession session = request.getSession();

ArrayList<String> adminuserinf=

(ArrayList<String>) session.getAttribute("adminuserinf");

String s = ShowUsernameUtil.getAdminUsername(request, logger);

if (s.equals("请先登录再进行提交")){

logger.info(random+"---------------设置奖励项：失败！-----------------");;

}

String adminAccount=adminuserinf.get(0);

if (item.isEmpty() || type.isEmpty()){

logger.info(random+"---------------设置奖励项：失败！-----------------");

return "请填写奖项内容";

}

LotteryItem li=new LotteryItem();

li.setLottery\_item(item);

li.setLottery\_type(type);

li.setLottery\_item\_setbyadmin(adminAccount);

String date=new Date().toLocaleString();

li.setLottery\_item\_settime(date);

lotteryItemDao.save(li);

// System.out.println("---------------设置奖励项结束-----------------");

logger.info(random+"---------------设置奖励项：成功-----------------");

return "设置奖励成功";

}

@ResponseBody

@RequestMapping("/admin/yanzheng")

public String yanzhengAdmin(HttpServletRequest request){//其中非空字段一定要添加

// System.out.println("---------------开始验证管理员-----------------");

int random= RandomUtil.getFormatRandom();

logger.info(random+"---------------开始验证管理员-----------------");

//

HttpSession session = request.getSession();

ArrayList<String> adminuserinf=

(ArrayList<String>) session.getAttribute("adminuserinf");

if (adminuserinf==null){

// System.out.println("请先登录再进行提交");

logger.info("请先登录再进行提交");

logger.info(random+"----------------验证管理员：失败！-----------------");

return "请先登录再进行提交";

}else {

// System.out.println("当前登录用户的clientname： "+adminuserinf.get(2));

logger.info("当前执行操作的 管理员用户 的Username： "+adminuserinf.get(0));

logger.info("当前执行操作的 管理员用户 的clientname： "+adminuserinf.get(2));

logger.info(random+"---------------验证管理员：成功！-----------------");

return adminuserinf.get(2);//返回clientname

}

}

@ResponseBody

@RequestMapping("/admin/deletethislottery")

public String deleteThisLottery(String id,HttpServletRequest request){

// System.out.println("--------准备删除奖项---------- "+id);

int random= RandomUtil.getFormatRandom();

logger.info(random+"--------准备删除奖项 "+id+"-------------------");

String s = ShowUsernameUtil.getAdminUsername(request, logger);

if (s.equals("请先登录再进行提交")){

logger.info(random+"----------------删除奖项：失败！--------------------");

}

try {

lotteryItemDao.deleteById(Long.valueOf(id));

} catch (NumberFormatException e) {

// System.out.println("传入id的格式不正确");

logger.info("传入id的格式不正确");

logger.info(random+"----------------删除奖项：失败！--------------------");

// e.printStackTrace();

return "传入id的格式不正确";

}

// System.out.println("--------奖项删除成功---------------");

logger.info(random+"--------id："+id+" 奖项删除：成功!---------------");

return "删除成功";

}

}

package com.shenke.userforwx.controller.admin;

import com.shenke.userforwx.Utils.RandomUtil;

import com.shenke.userforwx.Utils.ShowUsernameUtil;

import com.shenke.userforwx.entity.LotteryReward;

import com.shenke.userforwx.respository.LotteryRewardDao;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.stereotype.Controller;

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

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

import javax.annotation.Resource;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import java.util.List;

@Controller

public class LotteryStatusAdminController {

public final static Logger logger = LoggerFactory.getLogger(LotteryStatusAdminController.class);

@Resource

LotteryRewardDao lotteryRewardDao;

@RequestMapping("/admin/lotterystatus")

public String loginToLotteryStatus(){

return "admin/alllottery\_admin";

}

@ResponseBody

@RequestMapping("/admin/getalllottery")

public List<LotteryReward> getallLottery(HttpServletRequest request){//获取所有用户中得的奖励并返回给前台

// System.out.println("--------------开始获取所有抽奖结果------------------");

int random= RandomUtil.getFormatRandom();

logger.info(random+"--------------开始获取所有抽奖结果------------------");

List<LotteryReward> all = lotteryRewardDao.findAll();

String s = ShowUsernameUtil.getAdminUsername(request, logger);

if (s.equals("请先登录再进行提交")){

logger.info(random+"----------------获取所有抽奖结果：失败！--------------------");

}

logger.info(random+"--------------获取所有抽奖结果:成功！------------------");

return all;

}

@ResponseBody

@RequestMapping("/admin/setLotteryStatus")

public String setStatus(HttpServletRequest request,String id,String newStatus){

// System.out.println("------------------开始设置 "+id+" 号抽奖结果 状态---------------------");

int random= RandomUtil.getFormatRandom();

logger.info(random+"------------------开始设置 "+id+" 号抽奖结果 状态---------------------");

// ArrayList<String> adminuserinf=

// (ArrayList<String>) session.getAttribute("adminuserinf");

// if (adminuserinf==null){

//// System.out.println("请先登录再进行提交");

// logger.info("请先登录再进行提交");

// return "请先登录再进行提交";

// }else {

//// System.out.println("当前登录用户的clientname： "+adminuserinf.get(2));

// logger.info("当前登录用户的clientname： "+adminuserinf.get(2));

//

// }

// String adminAccount=adminuserinf.get(0);

String s = ShowUsernameUtil.getAdminUsername(request, logger);

if (s.equals("请先登录再进行提交")){

logger.info(random+"----------------获取所有抽奖结果：失败！--------------------");

}

LotteryReward LR\_needToChange = lotteryRewardDao.findById(id);

if (LR\_needToChange==null){

// System.out.println("查询该获奖记录失败");

logger.info("查询该获奖记录失败");

logger.info(random+"----------------获取所有抽奖结果：失败！--------------------");

return "查询该获奖记录失败";

}

if (newStatus.isEmpty()){

// System.out.println("要设置的新状态不能为空");

logger.info("要设置的新状态不能为空");

logger.info(random+"----------------获取所有抽奖结果：失败！--------------------");

return "要设置的新状态不能为空";

}

LR\_needToChange.setStatus(newStatus);

lotteryRewardDao.save(LR\_needToChange);

logger.info(random+"----------------获取所有抽奖结果：成功！--------------------");

return "修改新状态成功";

}

}

package com.shenke.userforwx.entity;

import javax.persistence.\*;

/\*\*

\* 订单实体类

\*

\*/

@Entity

@Table(name = "commit\_order")

public class CommitOrder {

@Id

@GeneratedValue(generator = "increment")

private int id;

@Column(length = 32)

private String username;

@Column(length = 32)

private String chanpin;

@Column(length = 32)

private String changdu;

@Column(length = 32)

private String houdu;

@Column(length = 32)

private String fukuan;

@Column(length = 32)

private String yanse;

@Column(length = 32)

private String shuliang;

@Column(length = 32)

private String beizhu;

@Column(length = 32)

private String create\_order\_time;

@Column(length = 32)

private String nonghu;

@Column(length = 32)

private String send\_time;

@Column(length = 64)

private String havesent="0";//0代表没有发送，其他数字代表已经发送，届时使用随机数表示,对应方法最多传参数6

@Override

public String toString() {

return "CommitOrder{" +

"id=" + id +

", username='" + username + '\'' +

", chanpin='" + chanpin + '\'' +

", changdu='" + changdu + '\'' +

", houdu='" + houdu + '\'' +

", fukuan='" + fukuan + '\'' +

", yanse='" + yanse + '\'' +

", shuliang='" + shuliang + '\'' +

", beizhu='" + beizhu + '\'' +

", create\_order\_time='" + create\_order\_time + '\'' +

", nonghu='" + nonghu + '\'' +

", send\_time='" + send\_time + '\'' +

", havesent='" + havesent + '\'' +

'}';

}

public String getSend\_time() {

return send\_time;

}

public void setSend\_time(String send\_time) {

this.send\_time = send\_time;

}

public String getHavesent() {

return havesent;

}

public void setHavesent(String havesent) {

this.havesent = havesent;

}

public String getNonghu() {

return nonghu;

}

public void setNonghu(String nonghu) {

this.nonghu = nonghu;

}

public String getCreate\_order\_time() {

return create\_order\_time;

}

public void setCreate\_order\_time(String create\_order\_time) {

this.create\_order\_time = create\_order\_time;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getChanpin() {

return chanpin;

}

public void setChanpin(String chanpin) {

this.chanpin = chanpin;

}

public String getChangdu() {

return changdu;

}

public void setChangdu(String changdu) {

this.changdu = changdu;

}

public String getHoudu() {

return houdu;

}

public void setHoudu(String houdu) {

this.houdu = houdu;

}

public String getFukuan() {

return fukuan;

}

public void setFukuan(String fukuan) {

this.fukuan = fukuan;

}

public String getYanse() {

return yanse;

}

public void setYanse(String yanse) {

this.yanse = yanse;

}

public String getShuliang() {

return shuliang;

}

public void setShuliang(String shuliang) {

this.shuliang = shuliang;

}

public String getBeizhu() {

return beizhu;

}

public void setBeizhu(String beizhu) {

this.beizhu = beizhu;

}

}

package com.shenke.userforwx.entity;

import javax.persistence.\*;

import javax.validation.constraints.NotNull;

@Entity

@Table(name="guest\_info")

public class GuestEnterFailedInfo {

@Id

@GeneratedValue(generator = "increment")

private int id;

@Column(unique = true)

private String ipAddress;

@Column

private String lastEnterFailedTime;

@Column

private String FailedCountForToday="0";

public GuestEnterFailedInfo() {

//构造了其他构造函数必须显式声明该默认构造函数

//否则出现 JpaSystemException: No default constructor for entity

}

public GuestEnterFailedInfo(String ipAddress, String lastEnterFailedTime, String failedCountForToday) {

this.ipAddress = ipAddress;

this.lastEnterFailedTime = lastEnterFailedTime;

FailedCountForToday = failedCountForToday;

}

public GuestEnterFailedInfo(String ipAddress, String lastEnterTime) {

this.ipAddress = ipAddress;

this.lastEnterFailedTime = lastEnterTime;

}

@Override

public String toString() {

return "GuestEnterFailedInfo{" +

"id=" + id +

", ipAddress='" + ipAddress + '\'' +

", lastEnterFailedTime='" + lastEnterFailedTime + '\'' +

", FailedCountForToday='" + FailedCountForToday + '\'' +

'}';

}

public String getFailedCountForToday() {

return FailedCountForToday;

}

public void setFailedCountForToday(String failedCountForToday) {

FailedCountForToday = failedCountForToday;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getIpAddress() {

return ipAddress;

}

public void setIpAddress(String ipAddress) {

this.ipAddress = ipAddress;

}

public String getLastEnterFailedTime() {

return lastEnterFailedTime;

}

public void setLastEnterFailedTime(String lastEnterFailedTime) {

this.lastEnterFailedTime = lastEnterFailedTime;

}

}

package com.shenke.userforwx.entity;

import javax.persistence.\*;

import java.util.Date;

@Entity

@Table(name = "lottery\_items")

public class LotteryItem {

@GeneratedValue

@Id

private Long id;

@Column(length = 64,nullable = false)

private String lottery\_item\_setbyadmin;

@Column(length = 32,nullable = false)

private String lottery\_type="losing";

@Column(length = 64,nullable = false)

private String lottery\_item="未中奖";

@Column(length = 64,nullable = false)

private String lottery\_item\_settime;

@Override

public String toString() {

return "LotteryItem{" +

"id=" + id +

", lottery\_item\_setbyadmin='" + lottery\_item\_setbyadmin + '\'' +

", lottery\_type='" + lottery\_type + '\'' +

", lottery\_item='" + lottery\_item + '\'' +

", lottery\_item\_settime='" + lottery\_item\_settime + '\'' +

'}';

}

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getLottery\_item\_setbyadmin() {

return lottery\_item\_setbyadmin;

}

public void setLottery\_item\_setbyadmin(String lottery\_item\_setbyadmin) {

this.lottery\_item\_setbyadmin = lottery\_item\_setbyadmin;

}

public String getLottery\_item\_settime() {

return lottery\_item\_settime;

}

public void setLottery\_item\_settime(String lottery\_item\_settime) {

this.lottery\_item\_settime = lottery\_item\_settime;

}

public String getLottery\_type() {

return lottery\_type;

}

public void setLottery\_type(String lottery\_type) {

this.lottery\_type = lottery\_type;

}

public String getLottery\_item() {

return lottery\_item;

}

public void setLottery\_item(String lottery\_item) {

this.lottery\_item = lottery\_item;

}

}

package com.shenke.userforwx.entity;

import javax.persistence.\*;

/\*\*

\* 用户实体类

\*

\*/

@Entity

@Table(name = "lottery\_reward")

public class LotteryReward {

@GeneratedValue

@Id

private Long id;

@Column(length = 32)

private String username;

@Column(length = 64)

private String reward;

@Column(length = 64)

private String status="尚未派奖";

@Column(length = 64)

private String create\_lottery\_time;

public String getCreate\_lottery\_time() {

return create\_lottery\_time;

}

public void setCreate\_lottery\_time(String create\_lottery\_time) {

this.create\_lottery\_time = create\_lottery\_time;

}

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getReward() {

return reward;

}

public void setReward(String reward) {

this.reward = reward;

}

public String getStatus() {

return status;

}

public void setStatus(String status) {

this.status = status;

}

}

package com.shenke.userforwx.entity;

import org.springframework.context.annotation.Primary;

import javax.persistence.\*;

/\*\*

\* 用户实体类

\*

\*/

@Entity

@Table(name = "user\_inf")

public class User {

@GeneratedValue

@Id

private Long id;

@Column(length = 32)

private String username;

@Column(length = 64)

private String password;

@Column(length = 32)

private String clientname;

@Column(length = 32,nullable = false)

private int jifen=0;//默认值

@Column(length = 32,nullable = false)

private int choujiangcishu=0;//默认值

@Column(length = 32,nullable = false)

private String createuser\_time;

@Column(length = 32,nullable = false)

private String duty="user";

@Column(length = 64)

private String address;

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public String getDuty() {

return duty;

}

public void setDuty(String duty) {

this.duty = duty;

}

public String getCreateuser\_time() {

return createuser\_time;

}

public void setCreateuser\_time(String createuser\_time) {

this.createuser\_time = createuser\_time;

}

public int getChoujiangcishu() {

return choujiangcishu;

}

public void setChoujiangcishu(int choujiangcishu) {

this.choujiangcishu = choujiangcishu;

}

public int getJifen() {

return jifen;

}

public void setJifen(int jifen) {

this.jifen = jifen;

}

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getClientname() {

return clientname;

}

public void setClientname(String clientname) {

this.clientname = clientname;

}

@Override

public String toString() {

return "User{" +

"id=" + id +

", username='" + username + '\'' +

", password='" + password + '\'' +

", clientname='" + clientname + '\'' +

", jifen=" + jifen +

", choujiangcishu=" + choujiangcishu +

", createuser\_time='" + createuser\_time + '\'' +

", duty='" + duty + '\'' +

", address='" + address + '\'' +

'}';

}

}

package com.shenke.userforwx.Model;

public class Commit {

private String input1;

private String input2;

private String input3;

private String input4;

private String input5;

private String input6;

public String getInput1() {

return input1;

}

public void setInput1(String input1) {

this.input1 = input1;

}

public String getInput2() {

return input2;

}

public void setInput2(String input2) {

this.input2 = input2;

}

public String getInput3() {

return input3;

}

public void setInput3(String input3) {

this.input3 = input3;

}

public String getInput4() {

return input4;

}

public void setInput4(String input4) {

this.input4 = input4;

}

public String getInput5() {

return input5;

}

public void setInput5(String input5) {

this.input5 = input5;

}

public String getInput6() {

return input6;

}

public void setInput6(String input6) {

this.input6 = input6;

}

// public String getInput7() {

// return input7;

// }

//

// public void setInput7(String input7) {

// this.input7 = input7;

// }

//

// private String input7;

@Override

public String toString() {

return "Commit{" +

"input1='" + input1 + '\'' +

", input2='" + input2 + '\'' +

", input3='" + input3 + '\'' +

", input4='" + input4 + '\'' +

", input5='" + input5 + '\'' +

", input6='" + input6 + '\'' +

'}';

}

}

package com.shenke.userforwx.Model;

public class FileConfig {

private String ProductionNameServer;

private String SendingServer;

public String getProductionNameServer() {

return ProductionNameServer;

}

public void setProductionNameServer(String productionNameServer) {

ProductionNameServer = productionNameServer;

}

public String getSendingServer() {

return SendingServer;

}

public void setSendingServer(String sendingServer) {

SendingServer = sendingServer;

}

}

package com.shenke.userforwx.Model;

import com.shenke.userforwx.entity.CommitOrder;

import javax.persistence.Column;

import javax.persistence.GeneratedValue;

import javax.persistence.Id;

import java.io.Serializable;

public class OrderWithCountPerSend implements Serializable {

private int id;

private String username;

private String chanpin;

private String changdu;

private String houdu;

private String fukuan;

private String yanse;

private String shuliang;

private String beizhu;

private String create\_order\_time;

private String nonghu;

private String send\_time;

private String havesent="0";

private String conutOfOneSend;

public OrderWithCountPerSend(int id, String username, String chanpin, String changdu, String houdu, String fukuan, String yanse, String shuliang, String beizhu, String create\_order\_time, String nonghu, String send\_time, String havesent, String conutOfOneSend) {

this.id = id;

this.username = username;

this.chanpin = chanpin;

this.changdu = changdu;

this.houdu = houdu;

this.fukuan = fukuan;

this.yanse = yanse;

this.shuliang = shuliang;

this.beizhu = beizhu;

this.create\_order\_time = create\_order\_time;

this.nonghu = nonghu;

this.send\_time = send\_time;

this.havesent = havesent;

this.conutOfOneSend = conutOfOneSend;

}

public String getConutOfOneSend() {

return conutOfOneSend;

}

public void setConutOfOneSend(String conutOfOneSend) {

this.conutOfOneSend = conutOfOneSend;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getChanpin() {

return chanpin;

}

public void setChanpin(String chanpin) {

this.chanpin = chanpin;

}

public String getChangdu() {

return changdu;

}

public void setChangdu(String changdu) {

this.changdu = changdu;

}

public String getHoudu() {

return houdu;

}

public void setHoudu(String houdu) {

this.houdu = houdu;

}

public String getFukuan() {

return fukuan;

}

public void setFukuan(String fukuan) {

this.fukuan = fukuan;

}

public String getYanse() {

return yanse;

}

public void setYanse(String yanse) {

this.yanse = yanse;

}

public String getShuliang() {

return shuliang;

}

public void setShuliang(String shuliang) {

this.shuliang = shuliang;

}

public String getBeizhu() {

return beizhu;

}

public void setBeizhu(String beizhu) {

this.beizhu = beizhu;

}

public String getCreate\_order\_time() {

return create\_order\_time;

}

public void setCreate\_order\_time(String create\_order\_time) {

this.create\_order\_time = create\_order\_time;

}

public String getNonghu() {

return nonghu;

}

public void setNonghu(String nonghu) {

this.nonghu = nonghu;

}

public String getSend\_time() {

return send\_time;

}

public void setSend\_time(String send\_time) {

this.send\_time = send\_time;

}

public String getHavesent() {

return havesent;

}

public void setHavesent(String havesent) {

this.havesent = havesent;

}

@Override

public String toString() {

return "OrderWithCountPerSend{" +

"conutOfOneSend='" + conutOfOneSend + '\'' +

", id=" + id +

", username='" + username + '\'' +

", chanpin='" + chanpin + '\'' +

", changdu='" + changdu + '\'' +

", houdu='" + houdu + '\'' +

", fukuan='" + fukuan + '\'' +

", yanse='" + yanse + '\'' +

", shuliang='" + shuliang + '\'' +

", beizhu='" + beizhu + '\'' +

", create\_order\_time='" + create\_order\_time + '\'' +

", nonghu='" + nonghu + '\'' +

", send\_time='" + send\_time + '\'' +

", havesent='" + havesent + '\'' +

'}';

}

}

package com.shenke.userforwx.Model.Memery;

import java.util.ArrayList;

import java.util.List;

public class ConfigFromFile {

public static List<String> al=new ArrayList<>();

public static String ProductionNameServer;

public static String SendOrdersServer;

}

package com.shenke.userforwx.Model.Memery;

import java.util.ArrayList;

public class Production {

public static ArrayList<String> al=new ArrayList<>();

public static String Json;

}

package com.shenke.userforwx.respository;

import com.shenke.userforwx.entity.CommitOrder;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.JpaSpecificationExecutor;

import org.springframework.data.jpa.repository.Query;

import java.util.List;

public interface CommitOrderDao extends JpaRepository<CommitOrder, Long>, JpaSpecificationExecutor<CommitOrder> {

//添加扩展查询

@Query(value = "select \* from commit\_order where username = ?1", nativeQuery = true)

List<CommitOrder> findAllByUsername(String username);

@Query(value = "select \* from commit\_order where id = ?1", nativeQuery = true)

CommitOrder findRecordById(int id);

//查出（changdu且shuliang）不同的结果集，其中数量决定了前端显示的大框数量

@Query(value = "SELECT \* , COUNT(\*) AS repeatnum FROM commit\_order where username=?1 and havesent=?2 GROUP BY beizhu,chanpin,fukuan,houdu,yanse,username,nonghu", nativeQuery = true)

List<CommitOrder> findDifferentChangduShuliang(String username,String haveset);

//根据上一条查询出的结果集中的长度和数量，查出对应的子结果集//其中username也是在这里过滤的

@Query(value = "SELECT \* FROM commit\_order WHERE havesent=?8 AND beizhu=?1 AND chanpin=?2 and fukuan=?3 and houdu=?4 AND yanse=?5 and username=?6 and nonghu=?7", nativeQuery = true)

List<CommitOrder> findAllByChangduAndShuliang(String beizhu,String chanpin,String fukuan,String houdu,String yanse,String username,String nonghu,String haveset);

/\*\*

\* 获取到的数量是该用户点全部发送的次数

\*/

@Query(value = "SELECT \* from commit\_order where"

+ " username=?1 AND havesent!='0' group by havesent",nativeQuery = true)

List<CommitOrder> findAllSentOrderGroupBySENTTAG(String username);

@Query(value = "SELECT \* from commit\_order where" + " username=?1 AND havesent='0'",nativeQuery = true)

List<CommitOrder> findAllNeedToSendOrder(String username);

@Query(value = "SELECT \* from commit\_order where username=?1 AND havesent=?2",nativeQuery = true)

List<CommitOrder> findAllByHavesent(String username,String haveset);

}

package com.shenke.userforwx.respository;

import com.shenke.userforwx.entity.GuestEnterFailedInfo;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.JpaSpecificationExecutor;

import org.springframework.data.jpa.repository.Query;

import java.util.List;

public interface GuestInfoDao extends JpaRepository<GuestEnterFailedInfo, Long>,

JpaSpecificationExecutor<GuestEnterFailedInfo> {

@Query(value = "select \* from guest\_info where ipAddress = ?1",nativeQuery = true)

List<GuestEnterFailedInfo> findAllByIpAddress(String ipAddress);

}

package com.shenke.userforwx.respository;

import com.shenke.userforwx.entity.LotteryItem;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import java.util.List;

public interface LotteryItemDao extends JpaRepository<LotteryItem,Long> {

}

package com.shenke.userforwx.respository;

import com.shenke.userforwx.entity.LotteryReward;

import com.shenke.userforwx.entity.User;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import java.util.List;

public interface LotteryRewardDao extends JpaRepository<LotteryReward,Long> {

//添加扩展查询

@Query(value = "select \* from lottery\_reward where username = ?1", nativeQuery = true)

List<LotteryReward> findByUsername(String username);

@Query(value = "select \* from lottery\_reward where id = ?1", nativeQuery = true)

LotteryReward findById(String id);

}

package com.shenke.userforwx.respository;

import com.shenke.userforwx.entity.User;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.JpaSpecificationExecutor;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

@Repository

public interface UserDao extends JpaRepository<User, Long>, JpaSpecificationExecutor<UserDao> {

//添加扩展查询

@Query(value = "select \* from user\_inf where username = ?1", nativeQuery = true)

User findByUsername(String username);

}

package com.shenke.userforwx.Service;

import org.springframework.stereotype.Service;

@Service

public interface GuestInfoService {

public void reset();

}

package com.shenke.userforwx.Service.impl;

import com.shenke.userforwx.Service.GuestInfoService;

import com.shenke.userforwx.entity.GuestEnterFailedInfo;

import com.shenke.userforwx.respository.GuestInfoDao;

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

import org.springframework.context.annotation.Bean;

import org.springframework.stereotype.Service;

import java.util.Iterator;

import java.util.List;

@Service

public class GuestInfoServiceimpl implements GuestInfoService {

@Autowired

GuestInfoDao guestInfoDao;

@Override

public void reset() {

List<GuestEnterFailedInfo> all = guestInfoDao.findAll();

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

all.get(i).setFailedCountForToday("0");

guestInfoDao.save(all.get(i));

}

}

}

package com.shenke.userforwx.Utils;

import cn.hutool.core.io.FileUtil;

import cn.hutool.core.io.file.FileReader;

import cn.hutool.core.util.CharsetUtil;

import java.util.List;

public class ConfigUtil {

public static List<String> findConfigFile(){

List<String> ls=null;

FileReader frBenji=null;

FileReader fr=null;

try {

frBenji=new FileReader(FileUtil.file("C:\\Users\\duoduo\\Desktop\\123\\config\\config.txt"),

CharsetUtil.systemCharset());

} catch (Exception e) {

System.out.println("在我的电脑上未找到对应文件");

}

try {

fr=new FileReader(FileUtil.file("C:\\Users\\Administrator\\Desktop\\222\\config\\config.txt"),

CharsetUtil.systemCharset());

} catch (Exception e) {

System.out.println("在服务器上未找到对应文件");

}

if (frBenji!=null){

ls=frBenji.readLines();

}else if(fr!=null){

ls=fr.readLines();

}

return ls;

}

}

package com.shenke.userforwx.Utils;

import java.lang.reflect.Constructor;

import java.lang.reflect.Field;

import java.lang.reflect.InvocationTargetException;

import java.lang.reflect.Method;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.annotation.Resource;

/\*\*

\* @author duoduo

\* @description V1.0

\*/

public class EntityUtils {

/\*\*

\* 数组集合转化为指定对象集合 指定的实体对象必须包含所以字段的构造方法，数组的元素的顺序将和构造方法顺序和类型一一对应

\*

\* @param list

\* 集合

\* @param clazz

\* c

\* @param <T>

\* 类型

\* @return List<T>

\*/

public static <T> List<T> castEntity(List<Object[]> list, Class<T> clazz) {

List<T> returnList = new ArrayList<>();

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

return returnList;

}

Class[] c2 = null;

Constructor[] constructors = clazz.getConstructors();

for (Constructor constructor : constructors) {

Class[] tClass = constructor.getParameterTypes();

if (tClass.length == list.get(0).length) {

c2 = tClass;

break;

}

}

// 构造方法实例化对象

for (Object[] o : list) {

Constructor<T> constructor = null;

try {

constructor = clazz.getConstructor(c2);

} catch (NoSuchMethodException e) {

e.printStackTrace();

}

try {

assert constructor != null;

returnList.add(constructor.newInstance(o));

} catch (InstantiationException | IllegalAccessException | InvocationTargetException e) {

e.printStackTrace();

}

}

return returnList;

}

public static <T> List<T> castEntity(List<Object[]> list, Class<T> clazz, Object model) {

List<T> returnList = new ArrayList<T>();

if (list.isEmpty()) {

return returnList;

}

//获取每个数组集合的元素个数

Object[] co = list.get(0);

//获取当前实体类的属性名、属性值、属性类别

List<Map> attributeInfoList = getFiledsInfo(model);

//创建属性类别数组

Class[] c2 = new Class[attributeInfoList.size()];

//如果数组集合元素个数与实体类属性个数不一致则发生错误

if (attributeInfoList.size() != co.length) {

return returnList;

}

//确定构造方法

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

c2[i] = (Class) attributeInfoList.get(i).get("type");

}

try {

for (Object[] o : list) {

Constructor<T> constructor = clazz.getConstructor(c2);

returnList.add(constructor.newInstance(o));

}

} catch (Exception ex) {

return returnList;

}

return returnList;

}

/\*\*

\* 根据属性名获取属性值

\*

\* @param fieldName 属性名

\* @param modle 实体类

\* @return 属性值

\*/

private static Object getFieldValueByName(String fieldName, Object modle) {

try {

String firstLetter = fieldName.substring(0, 1).toUpperCase();

String getter = "get" + firstLetter + fieldName.substring(1);

Method method = modle.getClass().getMethod(getter, new Class[]{});

Object value = method.invoke(modle, new Object[]{});

return value;

} catch (Exception e) {

return null;

}

}

/\*\*

\* 获取属性类型(type)，属性名(name)，属性值(value)的map组成的list

\*

\* @param model 实体类

\* @return list集合

\*/

private static List<Map> getFiledsInfo(Object model) {

Field[] fields = model.getClass().getDeclaredFields();

List<Map> list = new ArrayList(fields.length);

Map infoMap = null;

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

infoMap = new HashMap(3);

infoMap.put("type", fields[i].getType());

infoMap.put("name", fields[i].getName());

infoMap.put("value", getFieldValueByName(fields[i].getName(), model));

list.add(infoMap);

}

return list;

}

// /\*\*

// \* @Description:

// \* @Param:

// \* @return:

// \* @Author: Andy

// \* @Date:

// \*/

// public static String genCode() throws Exception {

// StringBuffer code = new StringBuffer("CK");

// code.append(DateUtil.getCurrentDateStr());

// String saleNumber = storageRepository.getTodayMaxOutNumber();

// if (saleNumber != null) {

// code.append(StringUtil.formatCode(saleNumber));

// } else {

// code.append("00001");

// }

// return code.toString();

// }

}

package com.shenke.userforwx.Utils;

import javax.servlet.http.HttpServletRequest;

import java.net.InetAddress;

import java.net.UnknownHostException;

public class IpUtil {

public static String getIpAddr(HttpServletRequest request) {

String ipAddress = null;

try {

ipAddress = request.getHeader("x-forwarded-for");

if (ipAddress == null || ipAddress.length() == 0 || "unknown".equalsIgnoreCase(ipAddress)) {

ipAddress = request.getHeader("Proxy-Client-IP");

}

if (ipAddress == null || ipAddress.length() == 0 || "unknown".equalsIgnoreCase(ipAddress)) {

ipAddress = request.getHeader("WL-Proxy-Client-IP");

}

if (ipAddress == null || ipAddress.length() == 0 || "unknown".equalsIgnoreCase(ipAddress)) {

ipAddress = request.getRemoteAddr();

if (ipAddress.equals("127.0.0.1")) {

// 根据网卡取本机配置的IP

InetAddress inet = null;

try {

inet = InetAddress.getLocalHost();

} catch (UnknownHostException e) {

e.printStackTrace();

}

ipAddress = inet.getHostAddress();

}

}

// 对于通过多个代理的情况，第一个IP为客户端真实IP,多个IP按照','分割

if (ipAddress != null && ipAddress.length() > 15) { // "\*\*\*.\*\*\*.\*\*\*.\*\*\*".length()

// = 15

if (ipAddress.indexOf(",") > 0) {

ipAddress = ipAddress.substring(0, ipAddress.indexOf(","));

}

}

} catch (Exception e) {

ipAddress="";

}

// ipAddress = this.getRequest().getRemoteAddr();

return ipAddress;

}

}

package com.shenke.userforwx.Utils;

import com.shenke.userforwx.controller.CommitController;

import okhttp3.\*;

import org.apache.commons.collections.MapUtils;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

import java.io.IOException;

import java.util.Iterator;

import java.util.Map;

import java.util.concurrent.TimeUnit;

public class OKHttpUtil {

public final static Logger logger = LoggerFactory.getLogger(OKHttpUtil.class);

private static OkHttpClient okHttpClient=new OkHttpClient();

/\*\*

\* get

\*

\* @param url 请求的url

\* @param queries 请求的参数，在浏览器？后面的数据，没有可以传null

\* @return

\*/

public static String get(String url, Map<String, String> queries) {

String responseBody = "";

StringBuffer sb = new StringBuffer(url);

if (MapUtils.isNotEmpty(queries)) {

boolean firstFlag = true;

Iterator iterator = queries.entrySet().iterator();

while (iterator.hasNext()) {

Map.Entry entry = (Map.Entry<String, String>) iterator.next();

if (firstFlag) {

sb.append("?" + entry.getKey() + "=" + entry.getValue());

firstFlag = false;

} else {

sb.append("&" + entry.getKey() + "=" + entry.getValue());

}

}

}

Request request = new Request.Builder().url(sb.toString()).build();

Response response = null;

try {

response = okHttpClient.newCall(request).execute();

//int status = response.code();

if (response.isSuccessful()) {

// System.out.println("--------OKHttp 发送get请求成功");

// System.out.println("--------OKHttp 发送get请求数据为: ");

logger.info("--------OKHttp 发送get请求成功");

// logger.info("--------OKHttp 发送get请求数据为: ");

String result=response.body().string();

// System.out.println(result);

// logger.info(result);

return result;

}

} catch (Exception e) {

// System.out.println("--------OKHttp 发送get请求失败");

logger.info("--------OKHttp 发送get请求失败");

// e.printStackTrace();//-TODO 在这个连接失败时暂时打印出函数调用栈

} finally {

if (response != null) {

response.close();

}

}

return responseBody;

}

/\*\*

\* Post请求发送JSON数据....{"name":"zhangsan","pwd":"123456"} 参数一：请求Url 参数二：请求的JSON

\* 参数三：请求回调

\*/

public static String postJsonParams(String url, String jsonParams) {

String responseBody = "";

RequestBody requestBody = RequestBody.create(MediaType.parse("application/json; charset=utf-8"), jsonParams);

Request request = new Request.Builder().url(url).post(requestBody).build();

Response response = null;

try {

response = okHttpClient.newCall(request).execute();

// int status = response.code();

if (response.isSuccessful()) {

return response.body().string();

}

} catch (Exception e) {

} finally {

if (response != null) {

response.close();

}

}

return responseBody;

}

}

package com.shenke.userforwx.Utils;

import cn.hutool.core.util.StrUtil;

public class PathUtil {

private String path;

public String getPath()

{

String path1 = this.getClass().getProtectionDomain().getCodeSource().getLocation().getPath();

if(System.getProperty("os.name").contains("dows"))

{

path1 = path1.substring(0,path1.length());

}

if(path1.contains("jar"))

{

path1 = path1.substring(0,path1.lastIndexOf("."));

return path1.substring(0,path1.lastIndexOf("/"));

}

StrUtil.removeSuffix(path1, "file:/");

return path1.replace("target/classes/", "");

}

}

package com.shenke.userforwx.Utils;

import java.util.ArrayList;

import java.util.List;

public class RandomUtil {

public static int getFormatRandom(){

return (int) (Math.random()\*999999999);

}

@org.jetbrains.annotations.NotNull

public static String getLongFormatRandom(int tens){

StringBuilder sb=new StringBuilder();

List<Character> templete=new ArrayList<>();

for (int i=0;i<9;++i){

templete.add((char)(i+48));

}

for (int i=0;i<26;++i){

templete.add((char) (i+65));

templete.add((char) (i+97));

}

for (int i=0;i<tens\*10;++i){

sb.append(templete.get((int) (Math.random()\*templete.size())));

}

return sb.toString();

}

}

package com.shenke.userforwx.Utils;

import org.slf4j.Logger;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpSession;

import java.util.ArrayList;

public class ShowUsernameUtil {

/\*\*

\* 外层接收到"ok"时不进行任何处理，表示验证通过

\* 否则需要在外层函数返回该方法的返回值

\* \*/

public static String getAdminUsername(HttpServletRequest request, Logger logger){

HttpSession session = request.getSession();

ArrayList<String> adminuserinf=

(ArrayList<String>) session.getAttribute("adminuserinf");

if (adminuserinf==null){

logger.info("请先登录再进行提交");

return "请先登录再进行提交";

}else {

logger.info("当前执行操作的 管理员用户 的Username： "+adminuserinf.get(0));

logger.info("当前执行操作的 管理员用户 的clientname： "+adminuserinf.get(2));

return "ok";

}

}

public static String getUsername(HttpServletRequest request, Logger logger){

HttpSession session = request.getSession();

ArrayList<String> userinf=

(ArrayList<String>) session.getAttribute("userinf");

if (userinf==null){

logger.info("请先登录再进行提交");

return "请先登录再进行提交";

}else {

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

//

}

logger.info("当前执行操作的 普通用户 的Username： "+userinf.get(0));

logger.info("当前执行操作的 普通用户 的clientname： "+userinf.get(2));

return userinf.get(0);

}

}

}