package com.hbzz.project.idass.controller;

import com.alibaba.fastjson.JSON;

import com.alibaba.fastjson.JSONObject;

import com.hbzz.bigdata.auto\_redis.config.tools.RedisSource;

import com.hbzz.bigdata.auto\_redis.config.tools.RedisUtils;

import com.hbzz.bigdata.auto\_web\_tools.config.tools.ResponseModel;

import com.hbzz.project.common.Constants;

import com.hbzz.project.jwt.TokenUtil;

import com.hbzz.project.zhxq.system.rpcmapping.IConfigService;

import com.hbzz.project.zhxq.system.rpcmapping.IUserService;

import com.idsmanager.dingdang.jwt.DingdangUserRetriever;

import org.apache.commons.lang3.StringUtils;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

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

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

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

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

import javax.annotation.Resource;

import java.util.HashMap;

import java.util.Map;

@RestController

@RequestMapping("/api/idaas")

public class SSOController {

private static final Logger log = LoggerFactory.getLogger(com.hbzz.project.idass.controller.SSOController.class);

private final String KEY = "znxq-idass-public-key";

@RedisSource("global")

private RedisUtils redisUtils;

@Resource

private IConfigService iConfigService;

@Resource

private IUserService iUserService;

/\*\*

\* id\_token是IDaaS请求时带来的，参数名为 "id\_token" ， 支持使用GET/POST两种方式放入;

\* PublicKey是在IDaaS里注册应用时生成的，可在应用详情页中JWT PublicKey查看;

\* target\_url是IDaaS里注册应用时设置的target\_url，此示例代码是通过id\_token和PublicKey解析用户信息并完成单点登录。

\*/

@RequestMapping(value = "/public/sso", method = {RequestMethod.GET, RequestMethod.POST})

public ResponseModel<String> ssoUrl(@RequestParam String id\_token) {

if(StringUtils.isBlank(id\_token)){

return ResponseModel.executeFailure("id\_token参数为空!");

}

//接收方法，GET和POST均支持

//<解析令牌>为解析id\_token并验证用户信息

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

params.put("key",KEY);

ResponseModel<JSONObject> rm = iConfigService.getConfigByKey(params);

if (!rm.isSuccess() || rm.getData() == null) {

return ResponseModel.executeFailure("获取PublicKey错误!");

}

JSONObject config = rm.getData().getJSONObject("config");

if (config == null) {

return ResponseModel.executeFailure("获取PublicKey的配置信息错误!");

}

//使用公钥解析id\_token，使用publicKey解密上一步获取的id\_token令牌

DingdangUserRetriever retriever = new DingdangUserRetriever(id\_token, config.toJSONString());

DingdangUserRetriever.User user = null;

try {

//获取用户信息

user = retriever.retrieve();

} catch (Exception e) {

log.error("Retrieve Username error", e);

return ResponseModel.executeFailure("获取IDaaS用户信息错误!");

}

//如果publicKey不正确或者id\_token过期，获取到的用户信息可能为null

if (null == user) {

return ResponseModel.executeFailure("获取IDaaS用户信息为null!");

}

String userName = user.getName();

if(StringUtils.isBlank(userName)){

return ResponseModel.executeFailure("IDaaS用户名称为空!");

}

params.clear();

params.put("name",userName);

ResponseModel rmUser = iUserService.getUserByName(params);

if(!rmUser.isSuccess() || rmUser.getData() == null){

params.clear();

params.put("phone",user.getMobile());

rmUser = iUserService.getUserByPhone(params);

if(!rmUser.isSuccess() || rmUser.getData() == null){

return ResponseModel.executeFailure("系统用户名称不存在!");

}

}

Object data = rmUser.getData();

JSONObject jObject = JSON.parseObject(JSON.toJSONString(data));

String token = TokenUtil.token(jObject.getString("account"),jObject.getString("name"));

if(StringUtils.isNotBlank(token)){

redisUtils.set(Constants.REDIS\_KEY\_PREFIX\_PROJECT+jObject.getString("account"),token,60\*60\*24);

return ResponseModel.executeSuccess("登录成功",1,token);

}

else{

return ResponseModel.executeFailure("生成系统token错误!");

}

}

}