1. 登录



@RequestMapping(value = { "login" })

**public** String login(Model model, HttpServletRequest request,

HttpServletResponse response) {

**if** (request.getMethod() == "POST") {

String account = request.getParameter("account");

String password = request.getParameter("password");

System.*out*.println(MD5String.*getMD5Str*(password));

**if** (userService.checkLogin(account, MD5String.*getMD5Str*(password))) {

/\*User user = userService.findById(userService.loginId(

account, MD5String.getMD5Str(password)));\*/

//int roleId = user.getRole();

request.getSession().setAttribute("account", account);// 登录成功放入session的内容，供前端页面访问

//request.getSession().setAttribute("roleId",roleId);

List<Row> list = userService.findUser(account);

request.getSession().setAttribute("name",list.get(0).getString("name"));

**return** "index/index";

} **else**

//return ajaxReturn(response, null, "账号或密码错误", 0);

**return** "index/login";

} **else** {

**return** "index/login";

}

}

1. 注册



@RequestMapping(value={"register"})

**public** String registerUser(Model model,HttpServletRequest request,HttpServletResponse response,User entity) **throws** Exception {

**if** (request.getMethod().equals("POST")) {

**if** (userService.userNameIsExist(entity.getAccount())) {

**return** *ajaxReturn*(response, **null**, "用户名已存在", 0);

} **else** {

//System.out.println(entity.getPassword());

entity.setPassword(MD5String.*getMD5Str*(entity.getPassword()));

//entity.setAddTime(Fn.time());

**int** flag = userService.save(entity);

System.*out*.println(flag);

**if** (flag == 1) {

**return** *ajaxReturn*(response, **null**, "添加成功", 1);

} **else** {

**return** *ajaxReturn*(response, **null**, "添加失败", 0);

}

}

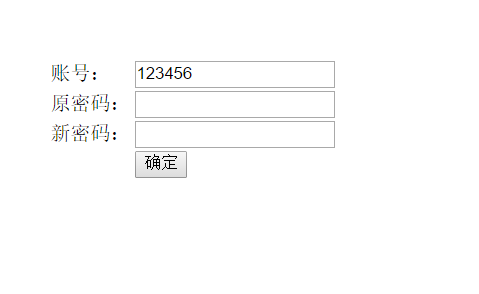
} **else** {

**return** "index/login";

}

}

1. 密码修改



@RequestMapping(value={"change\_pass"})

**public** String changePass(Model model,HttpServletRequest request,HttpServletResponse response,String name,

String OPass, String NPass) **throws** Exception {

**if** (request.getMethod().equals("POST")) {

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

where.put("account", request.getSession().getAttribute("account"));

where.put("password",request.getParameter("OPass"));

User user = userService.getObjByProperties(where);

**if** (StringUtil.*isEmpty*(user))

**return** *ajaxReturn*(response, **null**, "原密码错误", 0);

**else** {

user.setPassword(MD5String.*getMD5Str*(NPass));

**int** temp = userService.update(user);

**if** (temp == 1)

**return** *ajaxReturn*(response, **null**, "修改成功", 1);

**else**

**return** *ajaxReturn*(response, **null**, "修改失败", 0);

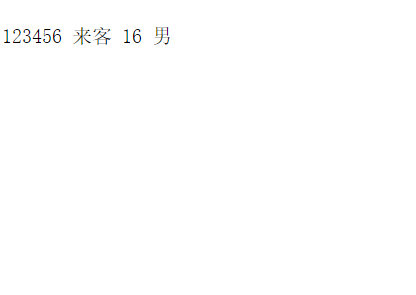
}

}

**return** "user/change\_pass";

}

1. 个人信息



@RequestMapping(value={"info"})

**public** String info(Model model,HttpServletRequest request,HttpServletResponse response) **throws** UnsupportedEncodingException {

//model.addAttribute("users", userService.findAllUser());

//System.out.println(request.getSession().getAttribute("account"));

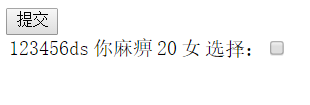
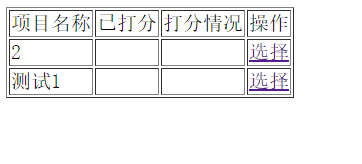
model.addAttribute("userinfo",userService.findUser((String)request.getSession().getAttribute("account")));

//model.addAttribute("info", "下面是所有用户");

**return** "user/info";

}

1. 添加打分人员



@RequestMapping(value={"alluser"})

**public** String alluser(Model model,HttpServletRequest request,HttpServletResponse response) **throws** UnsupportedEncodingException {

String project=request.getParameter("project");

//System.out.println(project);

project=**new** String(project.getBytes("ISO-8859-1"),"utf-8");

request.getSession().setAttribute("project2", project);

model.addAttribute("users",userService.findAllUser());

**return** "admin/chose2";

}

@RequestMapping(value = "chose2",method=RequestMethod.*POST*)

**public** String chose(Model model,HttpServletRequest request,HttpServletResponse response) {

//System.out.println("chose2");

// String project =request.getParameter("project");

// System.out.println(project);

String []a=request.getParameterValues("account");

**for**(**int** i=0;i<a.length;i++)

{

**if**(pointService.isExist(a[i],(String)request.getSession().getAttribute("project2"))){**continue**;}

**else**{

Point p=**new** Point();

p.setAccount(a[i]);

p.setProject((String)request.getSession().getAttribute("project2"));

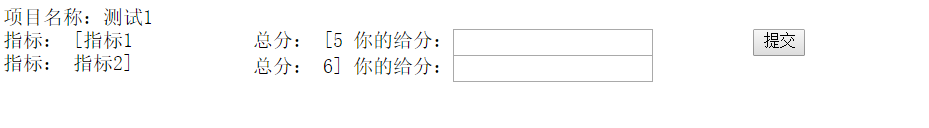
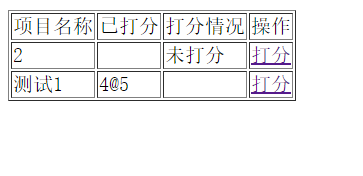
pointService.save(p);}

}

**return** "admin/prolist";

}

1. 打分



@RequestMapping(value = "doit")

**public** String doit(Model model,HttpServletRequest request,HttpServletResponse response) {

String project=request.getParameter("project");

**try** {

project=**new** String(project.getBytes("ISO-8859-1"),"utf-8");

} **catch** (UnsupportedEncodingException e) {

e.printStackTrace();

}

//System.out.println(project);

List<Row> doit=PointService.findtarget(project);

String[] t=doit.get(0).getString("target").split("\\@");

String[] s=doit.get(0).getString("score").split("\\@");

ArrayList<String> target=**new** ArrayList<String>();

**for**(**int** i=0;i<t.length;i++)

{

target.add(t[i]);

}

ArrayList<String> score=**new** ArrayList<String>();

**for**(**int** i=0;i<s.length;i++)

{

score.add(s[i]);

}

request.getSession().setAttribute("target",target);

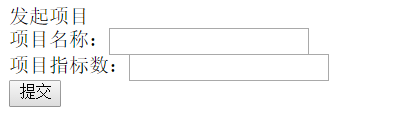
request.getSession().setAttribute("score", score);

model.addAttribute("doit",PointService.findtarget(project));

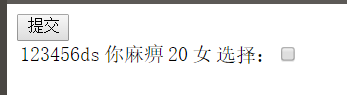
**return** "user/doit";

}

1. 发起项目







@RequestMapping(value = "add",method=RequestMethod.*POST*)

**public** String addtarget(Model model,HttpServletRequest request,HttpServletResponse response) {

String[] ta = request.getParameterValues("target");

String[] s =request.getParameterValues("score");

String project =request.getParameter("project");

request.getSession().setAttribute("project", project);

String target="";

**for**(**int** i=0;i<ta.length;i++)

{

**if**(i==ta.length-1){target=target+ta[i];}

**else**{

target=target+ta[i]+"@";}

//System.out.println(target[i]);

}

String score="";

**for**(**int** i=0;i<s.length;i++)

{

**if**(i==s.length-1){score=score+s[i];}

**else**{

score=score+s[i]+"@";}

}

Target t = **new** Target();

t.setTarget(target);

t.setScore(score);

t.setProject(project);

t.setAccount((String)request.getSession().getAttribute("account"));

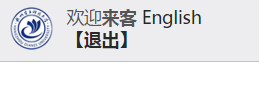
TargetService.save(t);

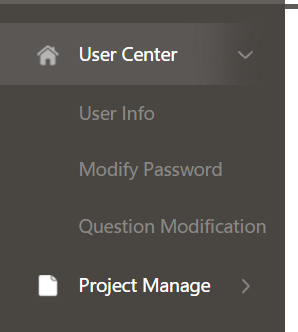
model.addAttribute("users",userService.findAllUser());

**return** "user/chose";

}

9、多国语言





package com.zlzkj.app.Interceptor;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Locale;

import java.util.Map;

import javax.servlet.http.Cookie;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.commons.codec.digest.DigestUtils;

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

import org.springframework.web.servlet.HandlerInterceptor;

import org.springframework.web.servlet.ModelAndView;

import org.springframework.web.servlet.i18n.CookieLocaleResolver;

import org.springframework.web.servlet.support.RequestContextUtils;

/\*import com.zlzkj.app.service.BaseUrlService;

import com.zlzkj.app.service.DictService;\*/

import com.zlzkj.app.util.DynamicDataSource;

import com.zlzkj.core.sql.Row;

/\*\*

\* 核心拦截器，配置request的一些初始值

\* @author Simon

\*

\*/

public class CoreInterceptor implements HandlerInterceptor{

/\*

@Autowired

private DictService dictService;

@Autowired

private BaseUrlService baseUrlService;\*/

@Autowired

CookieLocaleResolver resolver;

//@Autowired SessionLocaleResolver resolver;

/\*\*

\* 拦截器白名单列表

\*/

/\* public String[] exclude;

public void setExclude(String[] exclude) {

this.exclude = exclude;

}\*/

@Override

public boolean preHandle(HttpServletRequest request,

HttpServletResponse response, Object handler) throws Exception {

//动态数据源切换

if(request.getSession().getAttribute("language")==null){

Locale locale = RequestContextUtils.getLocaleResolver(request) .resolveLocale(request);

DynamicDataSource.setCurrentLookupKey(locale.getLanguage());

if(locale.getLanguage().contains("zh")){

request.getSession().setAttribute("language", "zh\_CN");

resolver.setLocale(request, response, Locale.CHINA );

}

else{

request.getSession().setAttribute("language", "en");

resolver.setLocale(request, response, Locale.ENGLISH );

}

//System.out.println("locale1=="+locale.getLanguage());

}else{

String language = String.valueOf(request.getSession().getAttribute("language"));

if(language.contains("zh")){

DynamicDataSource.setCurrentLookupKey("zh");

resolver.setLocale(request, response, Locale.CHINA );

}

else{

DynamicDataSource.setCurrentLookupKey("en");

resolver.setLocale(request, response, Locale.ENGLISH );

}

//System.out.println("locale2=="+request.getSession().getAttribute("language"));

}

String language = null;

if(( language = request.getParameter("language"))!=null){

if(language.contains("zh")){

DynamicDataSource.setCurrentLookupKey("zh");

resolver.setLocale(request, response, Locale.CHINA );

}

else{

DynamicDataSource.setCurrentLookupKey("en");

resolver.setLocale(request, response, Locale.ENGLISH );

}

request.getSession().setAttribute("language", language);

//System.out.println("locale3=="+language);

}

//白名单放行

/\*if(exclude!=null && exclude.length!=0){

String uri = request.getRequestURI().substring(request.getContextPath().length());

for(String one:exclude){

if(uri.startsWith(one)){

return true;

}

}

}

//解析url验证是否显示左侧

//String path = request.getServletPath();

//String[] str = path.split("/");

//拦截地址清单, 如果请求中是以以下路径开头，则进行拦截

String[] urls = {

\*

};

if(request.getAttribute("url")==null){//如果没有获取友情链接数据，则重新获取

//友情链接数据

// List<Row> rowlst = dictService.getTypeList("url\_type");

//

// for(Row row : rowlst){

// String item\_value = row.getString("id");

List<Row> urlList = baseUrlService.getUrlListByType("Links");

//row.put("list", urlList);

//}

request.setAttribute("url", urlList);

}\*/

return true;

}

@Override

public void postHandle(HttpServletRequest request,

HttpServletResponse response, Object handler,

ModelAndView modelAndView) throws Exception {

// TODO Auto-generated method stub

}

@Override

public void afterCompletion(HttpServletRequest request,

HttpServletResponse response, Object handler, Exception ex)

throws Exception {

// TODO Auto-generated method stub

}

}