成果二 动态Web页面设计实验

一、.实验目的

1、熟悉JSP的开发工具，掌握服务器端Web程序的工作原理

2、熟悉JSP编译指令，动作标记

3、熟悉JSP的隐含对象，正确理解request、session、application三个对象的作用域

4、掌握编写JavaBean的方法，使用JSP <jsp:useBean>、<jsp:set Property>、 <jsp:getProperty>3个动作指令。

5、掌握JSP中表单和表单Bean的映射

二、实验要求

1.实验之前认真查阅相关资料，准备好实验方案。

2.认真实验，对实验过程、结果进行分析，注意验证实验效果。

3.完成实验报告和实验成果

三、实验内容

1、修改实验1中的静态页面，改为JSP页面。

2、依据注册页面，设计1个JavaBean并实现服务器端的验证，防止错误数据的录入。

3、使用session对象，实现登录、注销功能。

4、练习使用客户端跳转、服务器端跳转。

四、实验结果与截图。

由于最近在做的项目中就是使用JSP技术，所以就直接呈现给您我在项目中写的几个页面。

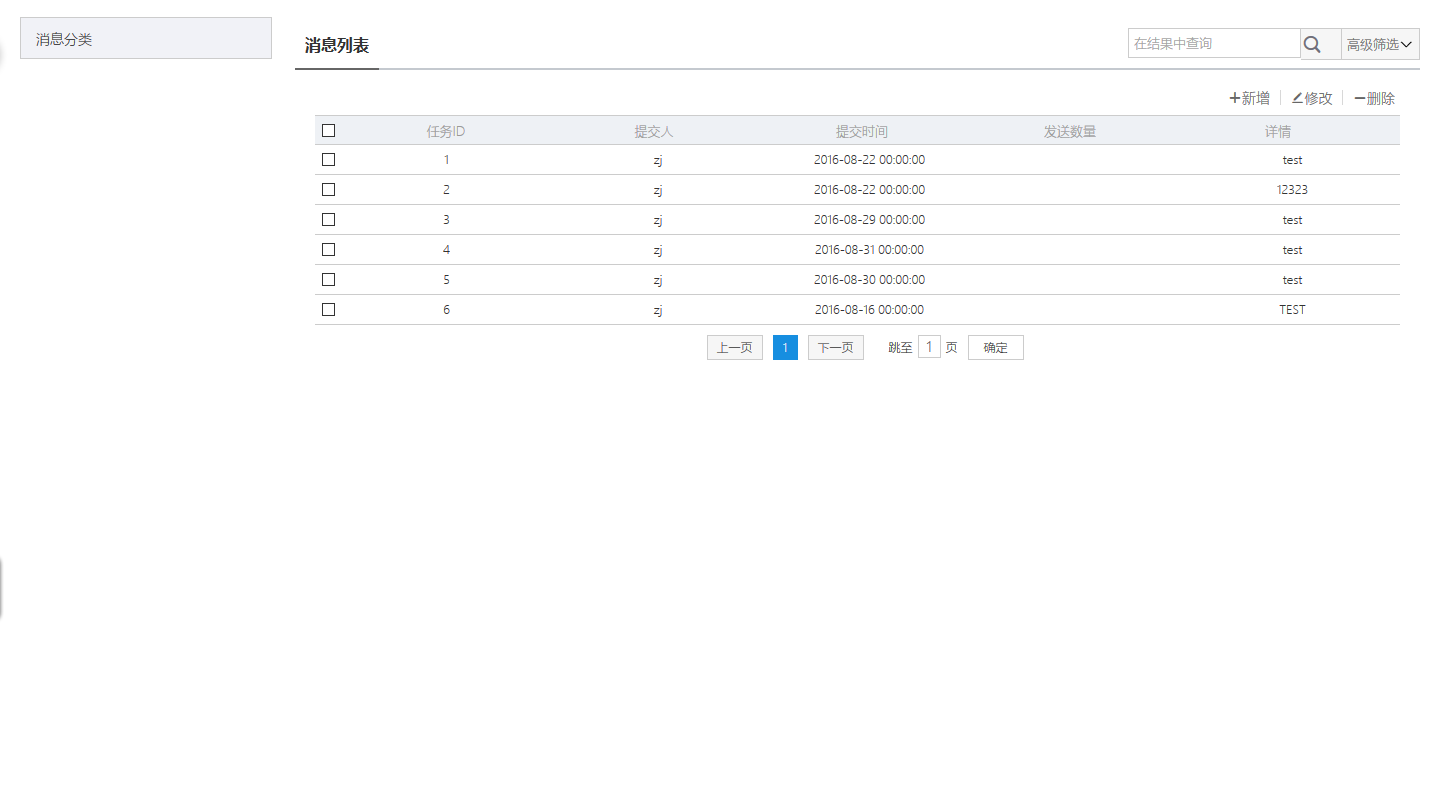
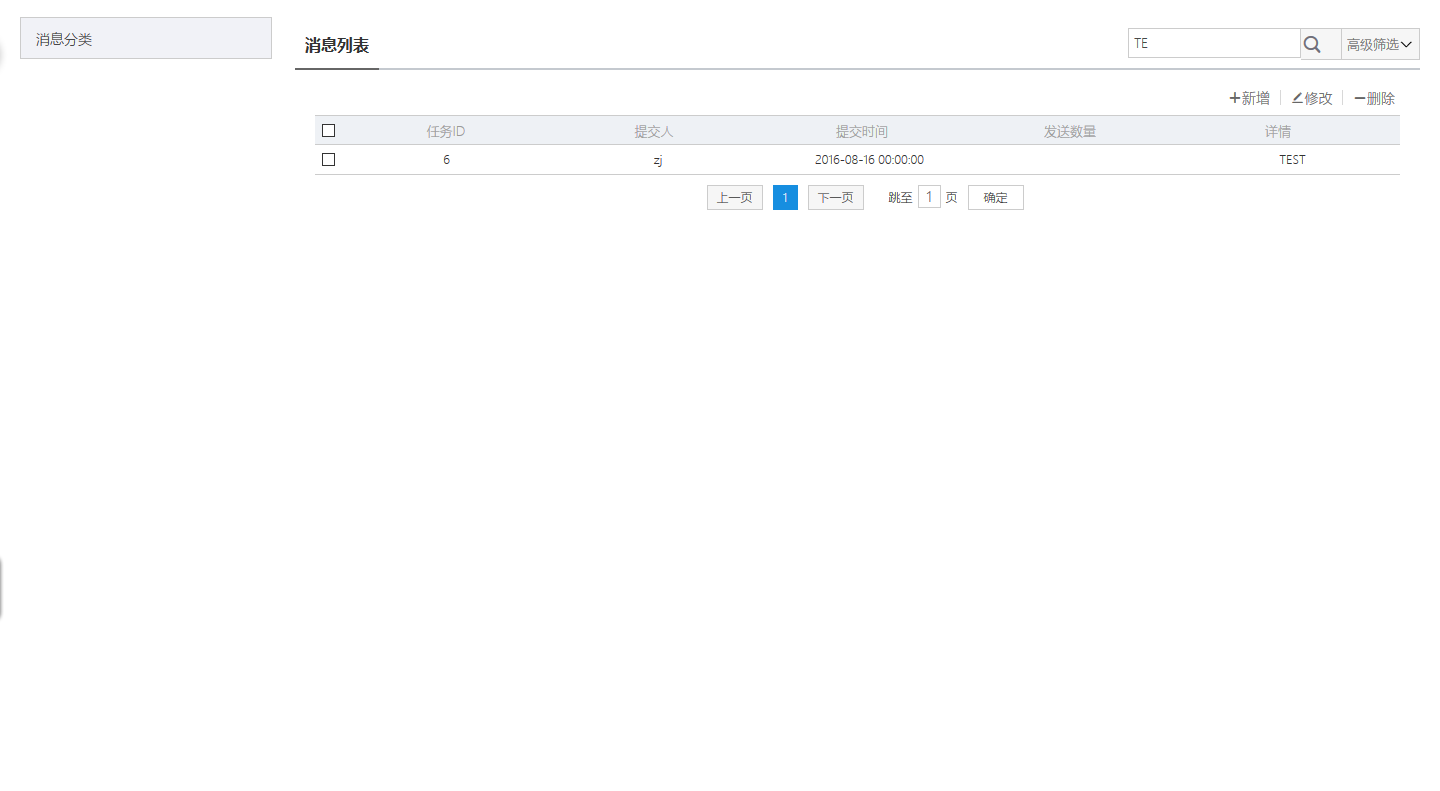
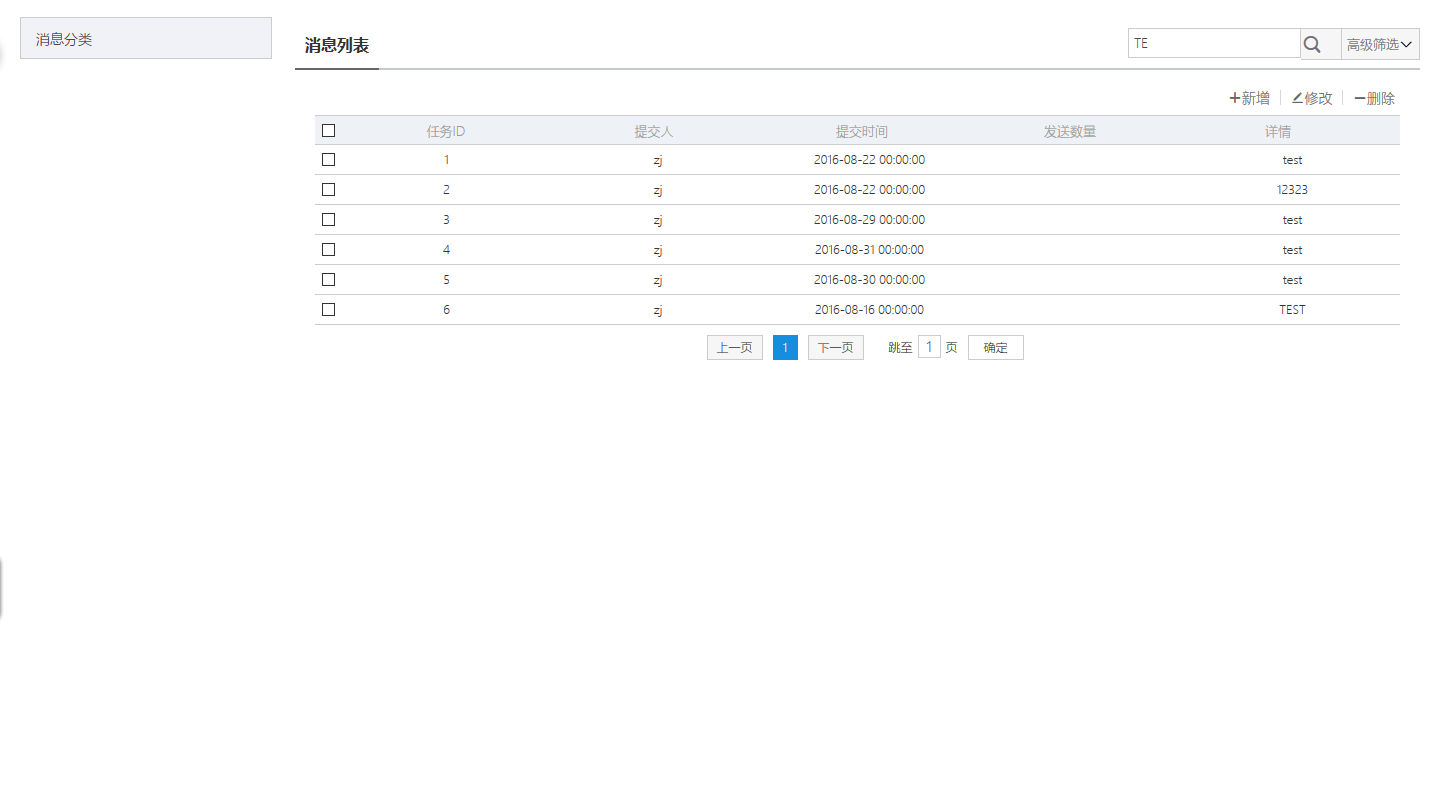
参与的是一个统一推送平台的设计和开发，其中我参与的工作是用户登陆界面的开发，关键词敏感词管理模块开发以及消息管理模块的开发这几大块。使用的是JSP技术，数据库是sql server。在开发过程中还使用了JQUERY框架以及DWR框架。

以下是我开发的页面的截图：（源码作为附录呈现）

登陆界面：



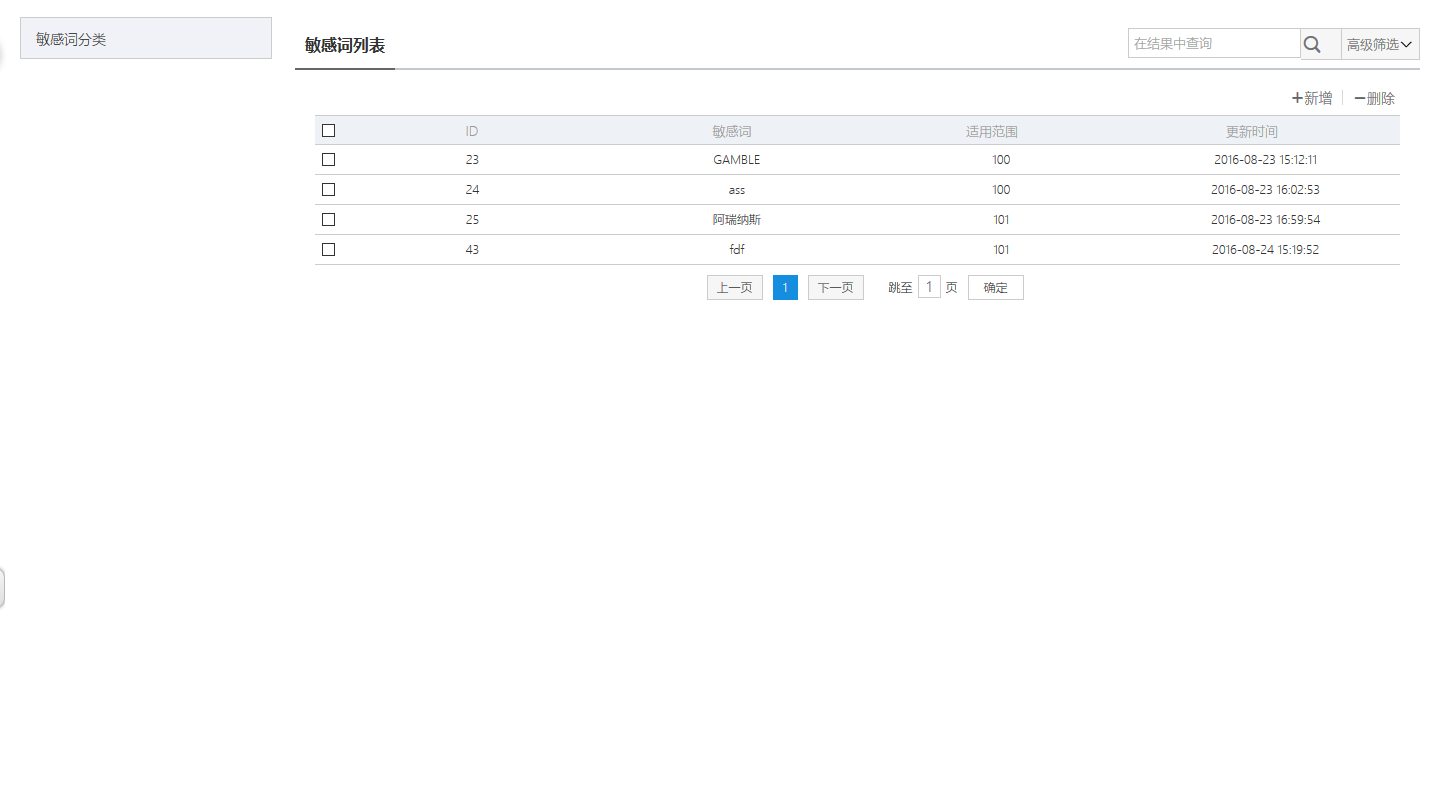
消息管理模块：查找消息和管理功能：



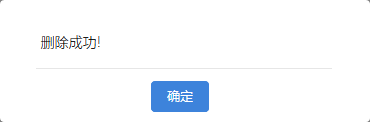
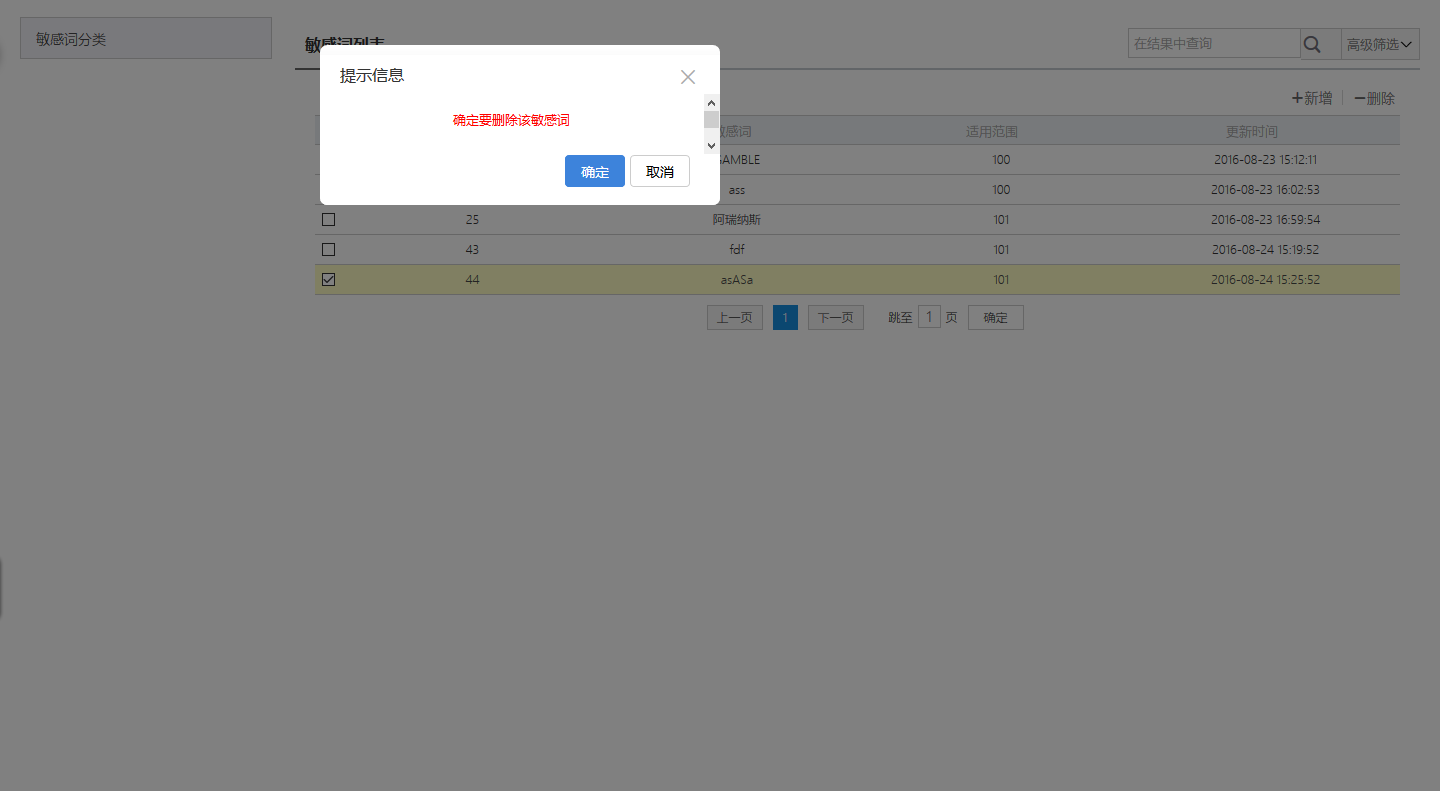
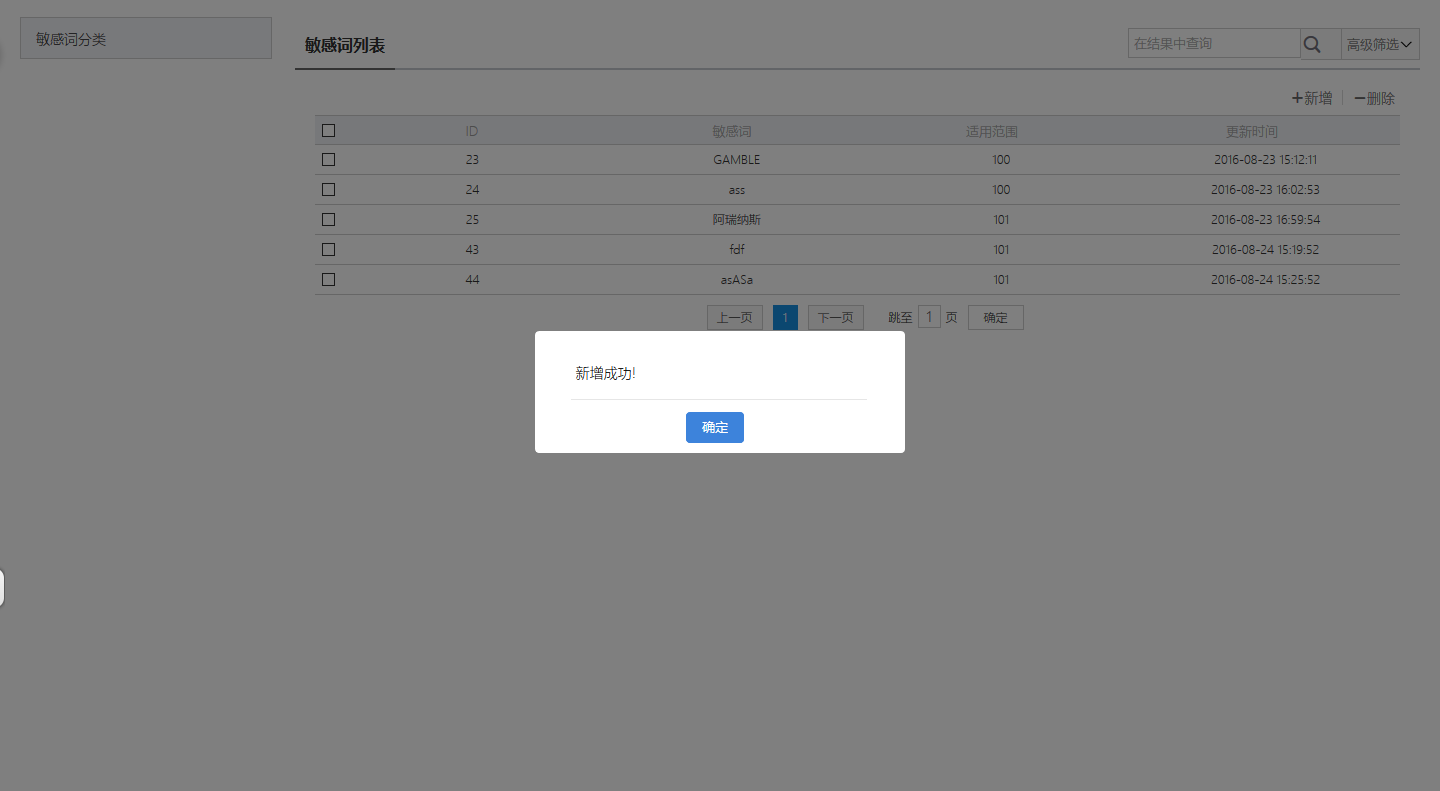
消息添加功能：



敏感词管理模块：



敏感词的添加和删除：



五、实验代码。

见附录。

附录：

1.登陆界面的javabean：

/\*\*

\* 人员--组织； 人员--角色； 角色--权限； 人员--权限； 权限--菜单-------------

\* 人员：状态state=00A有效，00X无效-------------------------------------

\* **@author** zeng.jian

\*/

**public** **class** LoginDAO {

/\*\*

\* 0 细化验证逻辑，分开验证用户名和密码

\* 根据用户名获取用户ID

\* **@author** zeng.jian

\*/

@SuppressWarnings("rawtypes")

**public** Map getStaffID(String staff\_code) {

String sql = "select staff\_id from dm\_staff where staff\_code = ? and state = '00A'";

List list = DAO.*queryForMap*(sql, **new** String[] { staff\_code });

**if** (!ListUtil.*isEmpty*(list)) {

**return** (Map) list.get(0);

}

**return** **new** HashMap();

}

/\*\*

\* 1 登录，根据用户名/密码获取用户

\*/

@SuppressWarnings("rawtypes")

**public** Map getStaffByCodeAnPwd(String staff\_code, String password) {

String sql = "select staff\_id from dm\_staff where staff\_code = ? and password = ? and state = '00A'";

List list = DAO.*queryForMap*(sql, **new** String[] { staff\_code, password });

**if** (!ListUtil.*isEmpty*(list)) {

**return** (Map) list.get(0);

}

**return** **new** HashMap();

}

/\*\*

\* 2.1 获取菜单权限ID-----------------------------------------------

\* 根据人员ID获取对应的权限列表，用户--角色--权限， 用户--权限

\* 此情况是一次性查询登录用户的所有角色下的所有权限+用户对应的权限，然后去重，获取一个权限列表，不存在切换权限的功能

\*/

@SuppressWarnings("rawtypes")

**public** List getPrivileges(String staff\_id) {

String sql = "SELECT privilege\_id FROM dm\_role\_privilege "

+ "WHERE page\_menu\_id is null and role\_id IN "

+ "(SELECT t1.ROLE\_ID FROM dm\_staff\_role t1, dm\_role t2 "

+ " WHERE t1.ROLE\_ID = t2.ROLE\_ID and t1.staff\_id =(SELECT STAFF\_ID FROM dm\_staff WHERE STAFF\_ID = ?)) "

+ "UNION "

+ "SELECT privilege\_id FROM dm\_staff\_privilege "

+ "WHERE page\_menu\_id is null and staff\_id = (SELECT STAFF\_ID FROM dm\_staff WHERE STAFF\_ID = ?) "

+ "ORDER BY PRIVILEGE\_ID";

**return** DAO.*queryForMap*(sql, staff\_id, staff\_id);

}

/\*\*

\* 2.2 根据菜单权限列表获取对应的菜单

\*/

@SuppressWarnings("rawtypes")

**public** List getPrivilegeMenus(List privileges) {

String sql = "SELECT \* from dm\_menu where MENU\_ID in ("

+ "select MENU\_ID from dm\_privilege WHERE PRIVILEGE\_ID in (";

String param[] = **new** String[privileges.size()];

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

Map map = (Map) privileges.get(i);

param[i] = MapUtil.*getStrValue*(map, "privilege\_id");

sql += "?,";

}

sql = sql.substring(0, sql.length() - 1) + "))";

**return** DAO.*queryForMap*(sql, param);

}

/\*\*

\* 3. 根据进入页面，获取页面按钮列表

\* **@param** params

\* **@return**

\*/

@SuppressWarnings({ "rawtypes", "unchecked" })

**public** List<?> getPageBtnPrivilege(List menus, String staff\_id) {

List list = **new** ArrayList();

// 先获取所有的有url的页面

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

Map menu = (Map) menus.get(i);

String url = MapUtil.*getStrValue*(menu, "url");

**if** (StrUtils.*isEmpty*(url)) {

**continue**;

}

String page\_menu\_id = MapUtil.*getStrValue*(menu, "menu\_id");

String sql = "select DISTINCT menu\_code,menu\_name,class\_name from dm\_menu t1, dm\_privilege t2 "

+ "where t1.MENU\_ID = t2.MENU\_ID and t1.TYPE = ? "

+ "and t2.PRIVILEGE\_ID in ( "

+ "select PRIVILEGE\_ID from dm\_role\_privilege where ROLE\_ID in "

+ "(select ROLE\_ID from dm\_staff\_role where STAFF\_ID = ?) "

+ "and PAGE\_MENU\_ID = ? "

+ "UNION "

+ "select PRIVILEGE\_ID from dm\_staff\_privilege where STAFF\_ID = ? "

+ "and PAGE\_MENU\_ID = ? )";

List btns = DAO.*queryForMap*(sql, SysMgrKeyValues.*MENU\_TYPE\_BUTTON*,

staff\_id, page\_menu\_id, staff\_id, page\_menu\_id);

Map map = **new** HashMap();

map.put("menu\_id", MapUtil.*getStrValue*(menu, "menu\_id"));

map.put("btns", btns);

list.add(map);

}

**return** list;

}

/\*\*

\* 4. 根据登录人id获取登录人信息

\*

\* **@param** staff\_id

\* **@return**

\*/

@SuppressWarnings("rawtypes")

**public** Map getUserInfoById(String staff\_id) {

String sql = "select t1.staff\_id, t1.staff\_code, t1.staff\_name, t1.org\_id, t2.org\_name from dm\_staff t1 left join dm\_organization t2 "

+ "on t1.ORG\_ID = t2.ORG\_ID "

+ "where t1.staff\_id = ? and t1.state = '00A'";

List list = DAO.*queryForMap*(sql, **new** String[] { staff\_id });

**if** (!ListUtil.*isEmpty*(list)) {

**return** (Map) list.get(0);

}

**return** **null**;

}

/\*\*

\* 5. 登录日志

\*

\* **@param**

\*/

@SuppressWarnings("rawtypes")

**public** **void** addEntryLog(Map params){

String sql = "insert into LOG\_STAFF\_LOGIN "

+ "(ID, STAFF\_ID, STAFF\_NAME, STAFF\_CODE, "

+ "ORG\_ID, LOGIN\_ACTION, IS\_SUCCESS, "

+ "CREATE\_TIME, DESCRIBE) "

+ "values(?,?,?,?,?,?,?,to\_date(?, 'YYYY-MM-DD HH24:MI:SS'),?)";

String id = SeqUtil.*getPrimaryKey*("LOG\_STAFF\_LOGIN", "ID");

String staff\_id = MapUtil.*getStrValue*(params, "staff\_id");

String staff\_name = MapUtil.*getStrValue*(params, "staff\_name");

String staff\_code = MapUtil.*getStrValue*(params, "staff\_code");

String org\_id = MapUtil.*getStrValue*(params, "org\_id");

String login\_action = MapUtil.*getStrValue*(params, "login\_action");

String is\_success = MapUtil.*getStrValue*(params, "is\_success");

String create\_time = DateUtil.*getFormatedDateTime*();

String describe = MapUtil.*getStrValue*(params, "describe");

DAO.*update*(sql, **new** String[] {id, staff\_id, staff\_name, staff\_code, org\_id,

login\_action, is\_success, create\_time, describe});

}

/\*\*

\* 6.1 查询最近的几条日志

\* **@param**

\*/

@SuppressWarnings("rawtypes")

**public** List getCurrentEntryLog(Map params){

String rownum = MapUtil.*getStrValue*(params, "rownum");

String sql = "select STAFF\_CODE, CREATE\_TIME from "

+ "(select STAFF\_CODE, CREATE\_TIME from LOG\_STAFF\_LOGIN "

+ "where IS\_SUCCESS = '1' "

+ "order by CREATE\_TIME DESC) "

+ "where rownum<? ";

**return** DAO.*queryForMap*(sql, **new** String[] { rownum });

}

/\*\*

\* 6.2 查询全部日志

\* **@param**

\*/

// **TODO** 根据分页数据修改查询

@SuppressWarnings("rawtypes")

**public** PageModel getAllEntryLog(Map params){

**int** pageSize = Integer.*parseInt*(MapUtil.*getStrValue*(params, "rows"));

**int** pageIndex = Integer.*parseInt*(MapUtil.*getStrValue*(params, "page"));

String sql = "select STAFF\_CODE, CREATE\_TIME from LOG\_STAFF\_LOGIN "

+ "where IS\_SUCCESS = '1' "

+ "order by CREATE\_TIME DESC";

PageModel p = DAO.*queryForPageModel*(sql, pageSize, pageIndex,

**new** String[] { });

**return** p;

}

}

2.登陆界面的JSP：

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*

pageEncoding=*"UTF-8"*%>

<!DOCTYPE html>

<html>

<head>

<title>统一推送平台</title>

<!--字符编码-->

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=UTF-8"* />

<meta http-equiv=*"library"* content=*"kernel,zteUI,dialog"*>

<!--使用最高版本的IE内核进行渲染-->

<meta http-equiv=*"X-UA-Compatible"* content=*"IE=Edge"* />

<!--自动适应手机屏幕宽度-->

<meta name=*"viewport"* content=*"width=device-width, initial-scale=1.0"* />

<script type=*"text/javascript"* src=*"*${pageContext.request.contextPath}*/public/common/Common.js"*></script>

<script type=*"text/javascript"* src=*"*${pageContext.request.contextPath}*/public/jquery/jquery.jcryption-1.1.js"*></script>

<!--登录组件类-->

<link rel=*"stylesheet"* href=*"*<%=request.getContextPath()%>*/css/login.css"* />

<script src=*"*<%=request.getContextPath()%>*/page/login/js/login.js"*></script>

</head>

<body class=*"themes"*>

<div class=*"login"*>

<div class=*"header"*>

<img src=*"*<%=request.getContextPath()%>*/images/logintit.png"*> <span

class=*"fs22 color-fff"*>统一推送平台</span>

</div>

<div class=*"login-form"*>

<div class=*"login-box"* id=*"login-box"*>

<div class=*"title"*>用户登录</div>

<div class=*'login-input'*>

<div id=*"msg-error"* class=*"msg-error"* style="display: *block*;"></div>

<div class=*"item"*>

<i class=*"iconfont"*>&#xe60c;</i>

<input type=*"text"* id=*"staff\_code"* name=*"staff\_code"* value=*""* placeholder=*"用户名"* class=*"form-control"* />

</div>

<div class=*"item"*>

<i class=*"iconfont"*>&#xe60d;</i>

<input type=*"password"* id=*"password"* name=*"password"* value=*""* placeholder=*"密码"* class=*"form-control"* />

</div>

<div class=*"code"*>

<input type=*"text"* id=*"verCode"* name=*"verCode"* value=*""* placeholder=*"验证码"* class=*"form-control"* />

<img src=*"*<%=request.getContextPath()%>*/servlet/RandomImageServlet"* id=*"safecode"* onclick="reloadVerifyCode();" alt=*"点击刷新"* />

</div>

<div class=*"submit"*>

<button id=*"login\_btn"* type=*"button"* class=*"btn btn-submit"*>登录</button>

</div>

</div>

</div>

<div class=*"clear"*></div>

</div>

</div>

<div class=*"footer"*></div>

</body>

</html>

3.登陆界面的JS

$(**function**() {

LoginUtil.init();

});

**var** LoginUtil = {

params : {},

init : **function**() {

**this**.checkFilter();

**this**.getPubKeys();

**this**.clickLogin();

},

// 判断有无参数sign，有则显示拦截

checkFilter: **function**(){

**var** sign = getUrlParam("sign")==**null**?"":getUrlParam("sign");

**if**(sign == "0"){

using(['messager'], **function**(){

$.messager.alert('登录超时，请重新登录');

});

}

},

// 获取公钥

getPubKeys: **function**() {

**var** me = **this**;

$.jCryption.getKeys(js\_path\_prefix+'servlet/PubKeyServlet.do',**function**(receivedKeys) {

me.params.keys = receivedKeys;

});

},

// 点击登录

clickLogin : **function**() {

**var** me = **this**;

$("#login\_btn").unbind("click").click(**function**() {

$("#msg-error").html("");

// 获取表单值

**var** requestParams = {};

$("#login-box").find("input").each(**function**(index, ele) {

**var** name = $(**this**).attr("name");

**var** value = $(**this**).val();

requestParams[name] = value;

});

**if**(requestParams.staff\_code == ""){

$("#msg-error").html("用户名输入不能为空");

**return**;

}

**if**(requestParams.password == ""){

$("#msg-error").html("密码输入不能为空");

**return**;

}

**if**(requestParams.verCode == ""){

$("#msg-error").html("验证码输入不能为空");

**return**;

}

//密码加密，然后登录

$.jCryption.encrypt(requestParams.password, me.params.keys, **function**(encryptedPasswd) {

**var** newPassword = encryptedPasswd;

requestParams.password = newPassword;

Ajax.getSy().remoteCall("LoginService", 'loginPost', [ requestParams ], **function**(data) {

**var** result = data.getResult() || {};

**if**(result.logsign == "true"){

$("#msg-error").html("");

window.location.href = js\_path\_prefix + "page/main/index.jsp"

}**else**{

$("#msg-error").html(result.msg);

reloadVerifyCode();

}

}, **function**(msg, exc){

$("#msg-error").html("登录失败：系统异常");

});

});

});

}

}

// 加载验证码

**function** reloadVerifyCode(){

**var** d=Math.random()\*(9999-1000)+1000;

**var** verifyCorrectCode=parseInt(d);

document.getElementById("safecode").src= js\_path\_prefix+ "servlet/RandomImageServlet?verifyCorrectCode="+d;

}

//获取url中的参数

**function** getUrlParam(name){

**var** reg = **new** RegExp("(^|&)" + name + "=([^&]\*)(&|$)"); //构造一个含有目标参数的正则表达式对象

**var** r = window.location.search.substr(1).match(reg); //匹配目标参数

**if** (r != **null**) **return** unescape(r[2]); **return** **null**; //返回参数值

}

4.消息管理模块的后端代码:

/\*\*

\* 消息管理

\* @author zeng.jian

\* @date 2016-10-18

\* \*/

public class MsgDAO {

/\*\*

\* 使用log4j记录日志

\* \*/

private Logger logger = Logger.getLogger(MsgDAO.class);

/\*\*

\* 2. 删除给定id的消息

\*

\* @param

\* @return

\* @author zeng.jian

\*/

public void deleteMsg(int id){

}

/\*\*

\* 3. 改

\*

\* @param

\* @return

\*/

public void editMsg(Map params){

}

/\*\*

\* 查找消息列表

\* @param param

\* @return

\* @author zeng.jian

\* @date 2016-10-18

\*/

@SuppressWarnings("rawtypes")

public PageModel getMsgList(Map params){

logger.log(Level.INFO,"getMsgList---------------");

//可以通过标题模糊匹配查找到想要的消息

String id= MapUtil.getStrValue(params, "ID");

String summary = MapUtil.getStrValue(params, "msg\_summary");

int pageSize=Integer.parseInt(MapUtil.getStrValue(params, "rows"));

int pageIndex=Integer.parseInt(MapUtil.getStrValue(params, "page"));

StringBuffer SQL = new StringBuffer();

SQL.append("select l.id,l.cerate\_staff\_name,l.publish\_time,l.msg\_summary from PUSH\_MSG l join MSG\_OPT c on l.id = c.msg\_id ");

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

if(!StrUtil.isEmpty(summary)){

logger.info("对summary进行模糊查找！");

SQL.append(" where msg\_summary like ?");

sqlParams.add("%"+summary+"%");

}

SQL.append(" order by ID");

PageModel list = DAO.queryForPageModel(SQL.toString(), pageSize,

pageIndex, sqlParams.toArray(new String[] {}));

return list;

}

/\*\*

\* 消息分类列表

\* @author zeng.jian

\* @date 2016-10-18

\* \*/

@SuppressWarnings({ "rawtypes", "unchecked" })

public List<Map> getMsgClasses(Map params) {

logger.info("getMsgClasses-----------------");

String id = MapUtil.getStrValue(params, "id");

Map temp = new HashMap();

temp.put("id", "-1");// -1为根结点的父结点ID

List<Map> roots = getSubClass(temp);

List<Map> result = new ArrayList<Map>(roots.size());

for (Map item : roots) {

result.add(getClass(item));

}

JSONArray array = JSONArray.fromObject(result);

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

return result;

}

/\*\*

\* 取某个目录的有效的下级目录

\*

\* @param menu\_id

\* @return

\* @author zeng.jian

\* @date 2016-10-18

\*/

@SuppressWarnings("rawtypes")

private List<Map> getSubClass(Map map) {

String id = (String) map.get("id");

String sql = "select id,msg\_summary from push\_msg t where t.id'"

+ id + "'";

sql += " and state=1 ";// 1为有效的目录

List<Map> list = DAO.queryForMap(sql);

return list;

}

/\*\*

\* 将一个结点及它的子树全部转化为一个map类型的对象

\*

\* @param map

\* 原始的结点，为map类型 : {id:XX,parent\_id:XX,class\_name:XX}

\* @return 改变后的结点，为带2个属性的map: {text:XXX,children:[]}

\* (text的值为原结点的class\_name值,[]中仍然保持这种结构,叶节点没有children属性)

\* @author zeng.jian

\* @date 2016-10-18

\*/

private Map getClass(Map map) {

Map result = new HashMap();

result.put("text", map.get("class\_name"));// 先将本节点的内容放入text的值中

logger.info("id:" + map.get("id"));

result.put("id", map.get("id"));

List<Map> children = getSubClass(map);

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

List<Map> newChildren = new ArrayList<Map>();

// 递归进行,把所有子节点变为对应的格式

for (Map item : children) {

newChildren.add(getClass(item));

}

result.put("children", newChildren);// 将改变后的子节点放入children的值中

}

return result;

}

/\*\*

\* 添加消息内容

\* @author zeng.jian

\* @date 2016-10-22

\* \*/

@SuppressWarnings({ "unchecked", "rawtypes" })

public Map addMssage(Map params) {

// 获取主键，大写

String id = SeqUtil.getPrimaryKey("PUSH\_MSG","ID");

String msg\_title = MapUtil.getStrValue(params, "MSG\_TITLE");

String msg\_summary = MapUtil.getStrValue(params, "MSG\_SUMMARY");

String msg\_content = MapUtil.getStrValue(params, "MSG\_CONTENT");

String create\_staff\_id = MapUtil.getStrValue(params, "CREATE\_STAFF\_ID");

String create\_staff\_name = MapUtil.getStrValue(params, "CERATE\_STAFF\_NAME");

// 执行插入

Map result = new HashMap();

try {

String sql = "insert into PUSH\_MSG(ID,MSG\_TITLE,MSG\_SUMMARY,MSG\_CONTENT,CREATE\_STAFF\_ID,CREATE\_STAFF\_NAME) "

+ "values(?,?,?,?,?,?)";

DAO.update(sql, new String[] { id,msg\_title,msg\_summary,msg\_content,create\_staff\_id,create\_staff\_name });

result.put(SysMgrKeyValues.STATUSSIGN, SysMgrKeyValues.SUCCESS);

result.put(SysMgrKeyValues.MSGSIGN, "新增成功!");

} catch (Exception e) {

e.printStackTrace();

result.put(SysMgrKeyValues.STATUSSIGN, SysMgrKeyValues.FAIL);

result.put(SysMgrKeyValues.MSGSIGN, "新增失败!");

}

return result;

}

}

@Controller

@RemoteProxy

public class MsgService {

/\*\*

\* 消息管理

\* @author zeng.jian

\* @date 2016-10-18

\* \*/

private MsgDAO getMsgDAO(){

LogicInvokerFactory logic = LogicInvokerFactory.getInstance();

MsgDAO dao = logic.getBO(MsgDAO.class);

return dao;

}

/\*\*

\* 查找消息列表

\* @author zeng.jian

\* @return

\* @param

\* @date 2016-10-18

\* \*/

@SuppressWarnings("rawtypes")

@RemoteMethod

@Transactional

@ContextMeta(cls = IContextLocal.DefaultWebContext)

public PageModel getMsgList(Map param){

return this.getMsgDAO().getMsgList(param);

}

/\*\*

\* 添加消息

\* @author zeng.jian

\* @param

\* @return

\* @date 2016-10-22

\* \*/

@SuppressWarnings("rawtypes")

@RemoteMethod

@Transactional

@ContextMeta(cls = IContextLocal.DefaultWebContext)

public Map addMessage(Map param){

return this.getMsgDAO().addMssage(param);

}

/\*\*

\* 查找消息分类列表

\* \*/

@SuppressWarnings("rawtypes")

@RemoteMethod

@Transactional

@ContextMeta(cls = IContextLocal.DefaultWebContext)

public List<Map> getMsgClasses(Map param){

return this.getMsgDAO().getMsgClasses(param);

}

}

5.消息管理模块的JS：

/\*\*

\* 消息管理

\* **@author** zeng.jian

\* **@date** 2016-10-18

\*/

$(document).ready(**function**() {

using(['ztesidemenu','datagrid','dialog','messager'],**function**(){

getMsgList(**null**);

//getMsgClasses();

//默认点击第一个

$(".titleList").find('.zte-title').eq(0).trigger('click');

});

});

/\*\*

\* 查找消息

\* **@author** zeng.jian

\* **@param** keyword

\* 当参数为空时查找全部数据，参数不为空时进行模糊搜索

\*/

**function** getMsgList(params) {

$('#demoTable').datagrid({

url : "MsgService.getMsgList",

queryParams : params,

pagination : **true**,

collapsible : **true**,

fitColumns : **true**,

height : 'auto',

nowrap : **true**,// 换行

singleSelect : **false**,//为true时不能全选

pageSize : 10,

columns : [ [

{field: 'ck',

title: "全选",

checkbox:**true**

},

{

field : 'id',

title : '任务ID',

align : 'center',

width:100,

}, {

field : 'cerate\_staff\_name',

title : '提交人',

align : 'center',

width:100,

}, {

field : 'publish\_time',

title : '提交时间',

align : 'center',

width:100,

}, {

field : 'publish\_number',

title : '发送数量',

align : 'center',

width:100,

}, {

field : 'msg\_summary',

title : '详情',

align : 'center',

width:100,

} ] ]

});

$("#demoTable").datagrid("reload");

}

/\*\*

\* 根据输入的关键词搜索消息summary

\* **@author** zeng.jian

\* **@date** 2016-10-24

\*/

**function** getMsgListLike() {

**var** keyword = $("#msgInfo").val();

**var** params={};

params.msg\_summary=keyword;

getMsgList(params);

}

/\*\*

\* 查找消息类型

\* **@author** zeng.jian

\* **@date** 2016-10-19

\* \*/

**function** getMsgClasses() {

**var** msgClasses = **null**;

Ajax.getSy().remoteCall("MsgService", 'getMsgClasses', [],

**function**(data) {

console.log("data:" + data);

$('#sidemenu').ztesidemenu({

multiple : **false**,

onClick : **function**(data, sidemenu) {

},

onBeforeAdd : **function**(params) {

},

onAdd : **function**(parent, text) {

},

onBeforeEdit : **function**(target) {

},

onEdit : **function**(target, text) {

},

onBeforeDelete : **function**(target) {

},

onDelete : **function**(target) {

},

data :data.data//要加.data,不然无法解析

});

}, **function**(msg, exc) {

$.messager.alert('系统异常', 'error');

});

$('#sidemenu').ztesidemenu("reload");

}

6.敏感词管理模块的JAVAbean：

/\*\*

\* 敏感词管理

\* **@author** zeng.jian

\* **@date** 2016-10-23

\* \*/

**public** **class** SensitiveWordDAO {

/\*\*

\* 使用log4j记录日志

\* \*/

**private** Logger logger = Logger.*getLogger*(SensitiveWordDAO.**class**);

/\*\*

\* 获取敏感词列表

\***@author** zeng.jian

\* **@date** 2016-10-23

\* \*/

**public** PageModel getSensitiveList(Map params){

logger.log(Level.*INFO*,"getSensitiveList---------------");

//可以通过内容模糊匹配查找到想要的敏感词

String id= MapUtil.*getStrValue*(params, "ID");

String words\_content = MapUtil.*getStrValue*(params, "words\_content");

**int** pageSize=Integer.*parseInt*(MapUtil.*getStrValue*(params, "rows"));

**int** pageIndex=Integer.*parseInt*(MapUtil.*getStrValue*(params, "page"));

StringBuffer SQL = **new** StringBuffer();

SQL.append("select id,words\_content,use\_range,update\_time from SENSITIVE\_WORDS ");

List<String> sqlParams = **new** ArrayList<String>();

**if**(!StrUtil.*isEmpty*(words\_content)){

logger.info("对敏感词进行模糊查找！");

SQL.append(" where words\_content like ?");

sqlParams.add("%"+words\_content+"%");

}

SQL.append(" order by ID");

PageModel list = DAO.*queryForPageModel*(SQL.toString(), pageSize,

pageIndex, sqlParams.toArray(**new** String[] {}));

**return** list;

}

/\*\*

\* 添加敏感词

\* **@author** zeng.jian

\* **@date** 2016-10-23

\*

\* \*/

@SuppressWarnings({ "unchecked", "rawtypes" })

**public** Map addSensitive(Map params){

logger.log(Level.*INFO*,"addSensitive---------------");

String id = SeqUtil.*getPrimaryKey*("SENSITIVE\_WORDS","ID");

String words\_content = MapUtil.*getStrValue*(params, "words\_content");

String words\_level = MapUtil.*getStrValue*(params, "words\_level");

String state = MapUtil.*getStrValue*(params, "state");

String use\_range = MapUtil.*getStrValue*(params, "use\_range");

//执行插入

Map result = **new** HashMap();

**try** {

String sql = "insert into SENSITIVE\_WORDS(id,words\_content,words\_level,update\_time,state,use\_range,create\_time)"+

" values (?,?,?,to\_date(?,'yyyy-mm-dd hh24:mi:ss'),?,?,to\_date(?,'yyyy-mm-dd hh24:mi:ss'))";

Date date=**new** Date();

DateFormat format=**new** SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

String time=format.format(date);

//检查是否已有待加入的关键词

String checkSql = "select words\_content from SENSITIVE\_WORDS";

//将已存在的关键词保存在current\_word\_list

List<Map> current\_word\_list =DAO.*queryForMap*(checkSql);

//遍历查看新增敏感词是否和已存在数据库中的敏感词相同;

//如果存在一致的敏感词则抛出SameException；添加敏感词失败

**for** (Map map : current\_word\_list) {

**if**(map.get("words\_content").equals(words\_content)){

**throw** **new** SameException();

}

//System.out.println(map.get("words\_content"));

}

//System.out.println(current\_word\_list);

DAO.*update*(sql, **new** String[]{

id,words\_content,words\_level,time,state,use\_range,time});

result.put(SysMgrKeyValues.*STATUSSIGN*, SysMgrKeyValues.*SUCCESS*);

result.put(SysMgrKeyValues.*MSGSIGN*, "新增成功!");

}**catch**(SameException ee){

ee.printStackTrace();

result.put(SysMgrKeyValues.*STATUSSIGN*, SysMgrKeyValues.*FAIL*);

result.put(SysMgrKeyValues.*MSGSIGN*, "新增失败!（请核对是否已有想要添加的敏感词）");

}**catch** (Exception e) {

e.printStackTrace();

result.put(SysMgrKeyValues.*STATUSSIGN*, SysMgrKeyValues.*FAIL*);

result.put(SysMgrKeyValues.*MSGSIGN*, "新增失败!");

}

**return** result;

}

/\*\*

\* 删除敏感词

\* **@author** zeng.jian

\* **@date** 2016-10-23

\* \*/

@SuppressWarnings({ "rawtypes", "unchecked" })

**public** Map deleteSensitive(Map params){

//获取主键

logger.log(Level.*INFO*,"deleteSensitive---------------");

String id = MapUtil.*getStrValue*(params, "id");

String words\_content = MapUtil.*getStrValue*(params, "words\_content");

Map result = **new** HashMap();

**try**{

//System.out.println(id);

//根据获取到的id删除掉对应的敏感词

String sql = "delete from SENSITIVE\_WORDS where ID = ?";

DAO.*update*(sql, **new** String[]{ id});

result.put(SysMgrKeyValues.*STATUSSIGN*, SysMgrKeyValues.*SUCCESS*);

result.put(SysMgrKeyValues.*MSGSIGN*, "删除成功!");

}**catch** (Exception e) {

e.printStackTrace();

result.put(SysMgrKeyValues.*STATUSSIGN*, SysMgrKeyValues.*FAIL*);

result.put(SysMgrKeyValues.*MSGSIGN*, "删除失败!");

}

**return** result;

}

/\*\*

\***@author** zeng.jian

\***@date** 2016-10-24

\*存在相同的敏感词的异常类

\*/

**public** **class** SameException **extends** Exception{

**public** SameException(){

**super**();

}

}

}

@Controller

@RemoteProxy

**public** **class** SensitiveWordService {

/\*\*

\* 敏感词管理

\* **@author** zeng.jian

\* **@date** 2016-10-23

\*

\* \*/

**private** SensitiveWordDAO getSensitiveWordDAO(){

LogicInvokerFactory logic = LogicInvokerFactory.*getInstance*();

SensitiveWordDAO dao = logic.getBO(SensitiveWordDAO.**class**);

**return** dao;

}

/\*\*

\* 查找敏感词列表

\* **@author** zeng.jian

\* **@return**

\* **@param**

\* **@date** 2016-10-23

\* \*/

@SuppressWarnings("rawtypes")

@RemoteMethod

@Transactional

@ContextMeta(cls = IContextLocal.*DefaultWebContext*)

**public** PageModel getSensitiveList(Map param){

**return** **this**.getSensitiveWordDAO().getSensitiveList(param);

}

/\*\*

\* 添加关键词

\* **@author** zeng.jian

\* **@return**

\* **@param**

\* **@date** 2016-10-23

\* \*/

@SuppressWarnings("rawtypes")

@RemoteMethod

@Transactional

@ContextMeta(cls = IContextLocal.*DefaultWebContext*)

**public** Map addSensitive(Map param){

**return** **this**.getSensitiveWordDAO().addSensitive(param);

}

/\*\*

\* 删除关键词

\* **@author** zeng.jian

\* **@return**

\* **@param**

\* **@date** 2016-10-23

\* \*/

@SuppressWarnings("rawtypes")

@RemoteMethod

@Transactional

@ContextMeta(cls = IContextLocal.*DefaultWebContext*)

**public** Map deleteSensitive(Map param){

**return** **this**.getSensitiveWordDAO().deleteSensitive(param);

}

}

7.敏感词模块的JSP：

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"* pageEncoding=*"UTF-8"*%>

<!DOCTYPE html>

<!-- 菜单弹窗 -->

<div id=*"SensitiveInfo"* class=*"content resource-main zte-form"* style="width:*800px*;height:*auto*;overflow: *hidden*;">

<input type=*"hidden"* id=*"menu\_id"* />

<table class=*'pop-table'*>

<tr>

<td class=*'tr'*><em class=*"zte-star"*>\*</em>敏感词：</td>

<td class=*'tl'*><input type=*"text"* class=*"zte-form-control zte-validatebox"* data-options=*"required:true"* id=*"words\_content"* /></td>

<td class=*'tr'*><em class=*"zte-star"*>\*</em>敏感词级别：</td>

<td class=*'tl'*><input type=*"text"* class=*"zte-form-control zte-validatebox"* data-options=*"required:true"* id=*"words\_level"* /></td>

</tr>

<tr>

<td class=*'tr'*><em class=*"zte-star"*>\*</em>敏感词状态：</td>

<td class=*'tl'*><input type=*"text"* class=*"zte-form-control"* id=*"state"* /></td>

<td class=*'tr'*><em class=*"zte-star"*>\*</em>适用范围：</td>

<td class=*'tl'*><input type=*"text"* class=*"zte-form-control zte-numberbox"* data-options=*"required:true"* id=*"use\_range"* /></td>

</tr>

</table>

</div>

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"* pageEncoding=*"UTF-8"*%>

<!DOCTYPE html>

<div id=*"deleteInfo"* class=*"content resource-main zte-form"* style="width:*auto*;height:*auto*;">

<table class=*'pop-table'*>

<tr>

<td class=*'tr'* style="width:*30%*;"></td>

<td class=*'tr'* style="color:*red*;text-align:*center*;width:*40%*;">确定要删除该敏感词吗？</td>

<td class=*'tr'* style="width:*30%*;"></td>

</tr>

</table>

</div>

8.敏感词管理模块的js：

/\*\*

\* 敏感词管理

\* author zeng.jian

\* date 2016-10-23

\*/

$(document).ready(**function**() {

using(['ztesidemenu','datagrid','dialog','messager'],**function**(){

getSensitiveList(**null**);

//默认点击第一个

$(".titleList").find('.zte-title').eq(0).trigger('click');

});

});

/\*\*

\* 查找关键词

\* **@author** zeng.jian

\* **@param** keyword

\* 当参数为空时查找全部数据，参数不为空时进行模糊搜索

\*/

**function** getSensitiveList(params) {

$('#demoTable').datagrid({

url : "SensitiveWordService.getSensitiveList",

queryParams : params,

pagination : **true**,

collapsible : **true**,

fitColumns : **true**,

height : 'auto',

nowrap : **true**,// 换行

singleSelect : **false**,//为true时不能全选

pageSize : 10,

columns : [ [

{field: 'ck',

title: "全选",

checkbox:**true**

},

{

field : 'id',

title : 'ID',

align : 'center',

width:100,

},

{

field : 'words\_content',

title : '敏感词',

align : 'center',

width:100,

}, {

field : 'use\_range',

title : '适用范围',

align : 'center',

width:100,

}, {

field : 'update\_time',

title : '更新时间',

align : 'center',

width:100,

}] ]

});

$("#demoTable").datagrid("reload");

}

/\*\*

\* 根据敏感词模糊匹配

\* **@author** zeng.jian

\* **@date** 2016-10-24

\*

\* \*/

**function** getSensitiveListLike(){

**var** keyword=$("#sensitiveInfo").val();

**var** params={};

params.words\_content=keyword;

getSensitiveList(params);

}

/\*\*

\* 添加敏感词

\* author zeng.jian

\* date 2016-10-23

\*

\* \*/

**function** addSensitive(){

$('#rootDiv').dialog({

title: '新增敏感词',

width: 800,

height: 350,

closed: **false**,

cache: **false**,

href: 'add\_new\_sensitive\_pop.jsp',

modal: **true**,

buttons:[{

text:'确定',

cls:'zte-btn-sure',

handler:**function**(){

**if**(!$('#SensitiveInfo').form('validate')){

**return**;

}

// 保存数据

**var** params = $('#SensitiveInfo').form('getData');

Ajax.getSy().remoteCall("SensitiveWordService", 'addSensitive', [ params ], **function**(data) {

**var** result = data.getResult() || {};

$.messager.alert(result.msg);

**if**(result.logsign=="true"){

$("#demoTable").datagrid("reload");

$('#rootDiv').dialog('close');

}

}, **function**(msg, exc){

$.messager.alert('系统异常','error');

});

}

},{

text:'取消',

handler:**function**(){

$('#rootDiv').dialog('close');

}

}]

});

}

/\*\*

\* 删除敏感词

\* **@author** zeng.jian

\* **@date** 2016-10-23

\*

\* \*/

**function** deleteSensitive(){

**var** rowData = $("#demoTable").datagrid('getSelected');

**if**(!rowData||rowData.length==0||rowData.length>1){

$.messager.alert("请选择一个分组！");

**return**;

}

$('#rootDiv').dialog({

title: '提示信息',

width: 400,

height: 150,

closed: **false**,

cache: **false**,

href: 'pop\_delete\_sensitive.jsp',

modal: **true**,

buttons:[{

text:'确定',

cls:'zte-btn-sure',

handler:**function**(){

// 保存数据

**var** id = rowData.id;

**var** words\_content=rowData.words\_content;

**var** params = {};

params.id = id;

params.words\_content=words\_content;

Ajax.getSy().remoteCall("SensitiveWordService", 'deleteSensitive', [ params ], **function**(data) {

**var** result = data.getResult() || {};

$.messager.alert(result.msg);

**if**(result.logsign=="true"){

$("#demoTable").datagrid("reload");

$('#rootDiv').dialog('close');

}

}, **function**(msg, exc){

$.messager.alert('系统异常','error');

});

}

},{

text:'取消',

handler:**function**(){

$('#rootDiv').dialog('close');

}

}]

});

}

8．Session对象配置：

<!-- 生产F5session超时30分钟，勿随意更改，保持session和redis的一致性 -->

<session-config>

<session-timeout>30</session-timeout>

</session-config>