# 系统功能实现

## 1.1 程序的类图

### 1.1.1 公告Gonggao类图

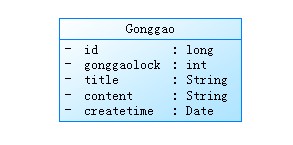


图5-1 Gonggao类图

对应的程序代码：

public class Gonggao implements Serializable{

private static final long *serialVersionUID*=-7141419035239709511L;

private long id;

private int gonggaolock;

private String title;

private String content;

private Date createtime;

### 1.1.2 简历Jianli类图

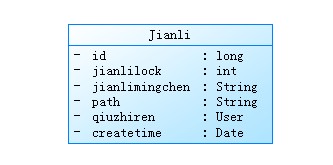


图5-2 Jianli类图

对应的程序代码：

public class Jianli implements Serializable{

private static final long *serialVersionUID*=-7141419035239709511L;

private long id;

private int jianlilock;

private String jianlimingchen;//简历名称

private String path;//简历路径

private User qiuzhiren;//求职者

private Date createtime;//时间

### 1.1.3 就业Jiuye类图

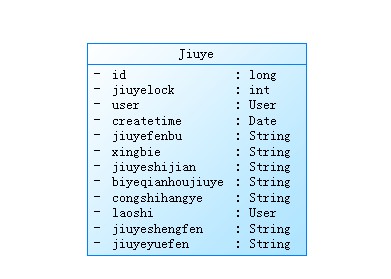


图5-3 Jiuye类图

对应的程序代码：

public class Jiuye implements Serializable{

private static final long *serialVersionUID*=-7141419035239709511L;

private long id;

private int jiuyelock;

private User user;//关联学生

private Date createtime;//

private String jiuyefenbu;//就业分布南方北方

private String xingbie;//性别男，女

private String jiuyeshijian;//就业时间

private String biyeqianhoujiuye;//毕业前就业毕业后就业

private String congshihangye;//从事行业

private User laoshi;//关联老师

private String jiuyeshengfen;//就业省份

private String jiuyeyuefen;//就业月份

### 1.1.4 留言Liuyan类图

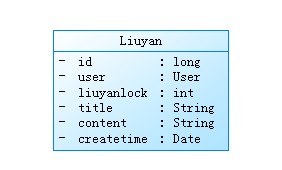
****

图5-4 Liuyan类图

对应的代码：

public class Liuyan implements Serializable{

private static final long *serialVersionUID*=-7141419035239709511L;

private long id;

private User user;

private int liuyanlock;

private String title;

private String content;

private Date createtime;

### 1.1.5 投递记录Toudijilu类图

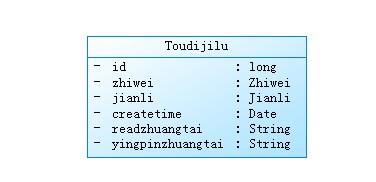
****

图5-5 Toudijilu类图

对应的代码：

public class Toudijilu implements Serializable{

private static final long *serialVersionUID*=-7141419035239709511L;

private long id;

private Zhiwei zhiwei;

private Jianli jianli;

private Date createtime;

private String readzhuangtai;//未阅读已阅读

private String yingpinzhuangtai;//应聘状态

### 1.1.6 文档Wendang类图

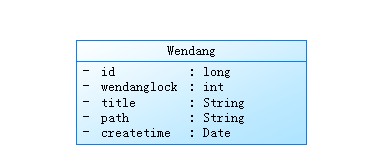
****

图5-6 Wendang类图

对应的代码：

public class Wendang implements Serializable{

private static final long *serialVersionUID*=-7141419035239709511L;

private long id;

private int wendanglock;

private String title;

private String path;//简历路径

private Date createtime;

### 1.1.7 系Xi类图

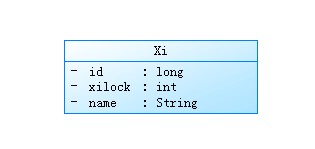
****

图5-7 Xi类图

对应的代码：

public class Xi implements Serializable{

private static final long *serialVersionUID*=-7141419035239709511L;

private long id;

private int xilock;

private String name;

### 1.1.8 新闻Xinwen类图

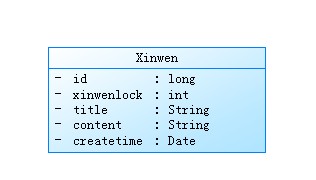
****

图5-8 Xinwen类图

对应的代码：

public class Xinwen implements Serializable{

private static final long *serialVersionUID*=-7141419035239709511L;

private long id;

private int xinwenlock;

private String title;

private String content;

private Date createtime;

### 1.1.9 用户User类图

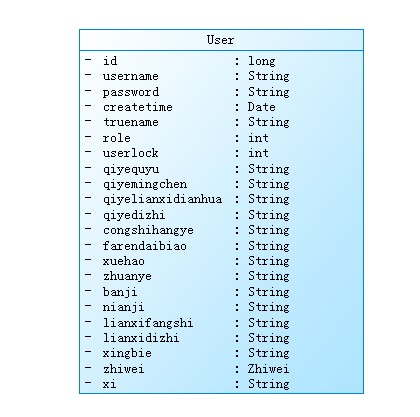
****

图5-9 User类图

对应的代码：

public class User implements Serializable{

private static final long *serialVersionUID*=-7141419035239709511L;

private long id;

private String username;

private String password;

private Date createtime;

private String truename;

private int role;//4表示系统管理员,3表示老师管理员,2表示毕业生用户,1表示企业用户

private int userlock;

private String qiyequyu;//企业区域南方北方

private String qiyemingchen;//企业名称

private String qiyelianxidianhua;//企业联系电话

private String qiyedizhi;//企业地址

private String congshihangye;//从事行业

private String farendaibiao;//法人代表

private String xuehao;//学号

private String zhuanye;//专业

private String banji;//班级

private String nianji;//年级

private String lianxifangshi;//联系方式

private String lianxidizhi;//联系地址

private String xingbie;//性别

private Zhiwei zhiwei;//岗位

private String xi;//系

### 1.1.10 职位Zhiwei类图

****

图5-10 Zhiwei类图

对应的代码：

public class Zhiwei implements Serializable{

private static final long *serialVersionUID*=-7141419035239709511L;

private long id;

private int zhiweilock;

private String zhiweileibie;//职位类别

private String zhiweimingchen;//岗位名称

private String zhaopinrenshu;//招聘人数

private String gongzuodidian;//工作地点

private String yuexin;//月薪

private String xueli;//学历

private String gongzuonianxian;//工作年限

private String gongzuoleixing;//全职，兼职，不限

private String xingbie;//性别，男，女

private String zhiweimiaoshu;//岗位描述

private String youxianqi;//有效期

private Date createtime;

private User zhiweifaburen;//岗位发布人

private String gongzuoshengfen;//工作省份

### 1.1.11 专业Zhuanye类图

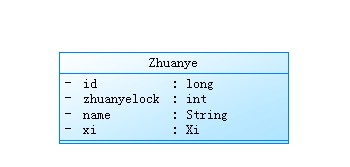
****

图5-11 Zhuanye类图

对应的代码：

public class Zhuanye implements Serializable{

private static final long *serialVersionUID*=-7141419035239709511L;

private long id;

private int zhuanyelock;

private String name;

private Xi xi;

## 1.2 程序的时序图

### 1.2.1 公告模块时序图

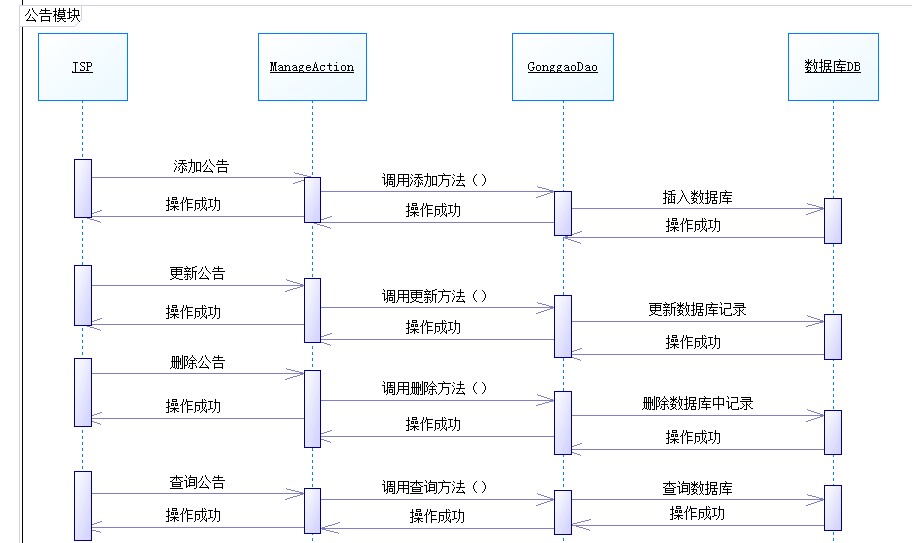


图5-12 公告模块时序图

### 1.2.2 简历模块时序图

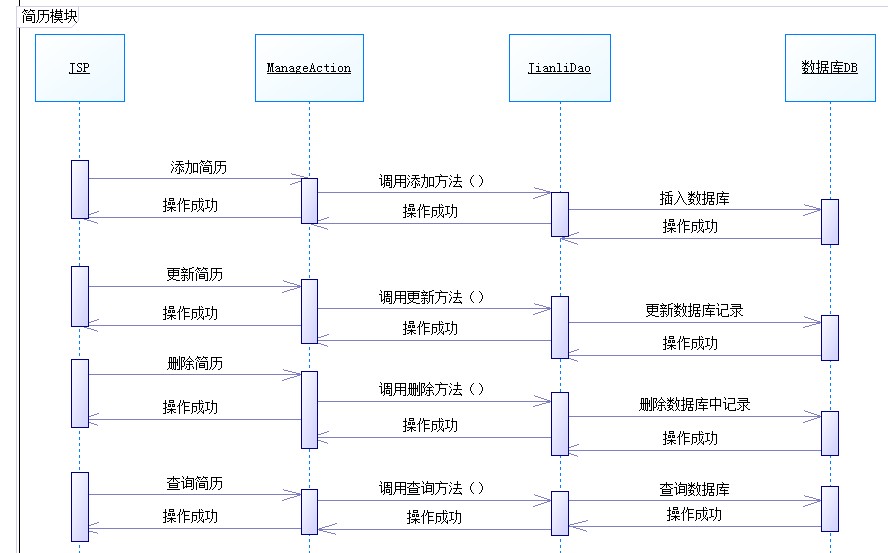


图5-13 简历模块时序图

### 1.2.3 就业模块时序图

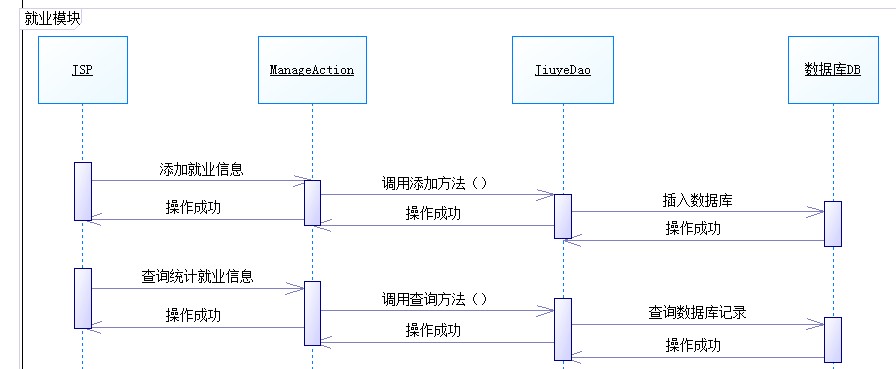


图5-14 就业模块时序图

### 1.2.4 留言模块时序图

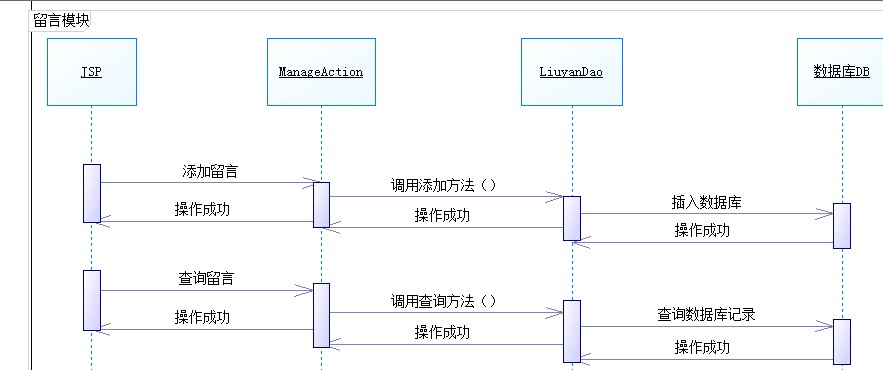


图5-15 留言模块时序图

### 1.2.5 投递记录模块程序流程图

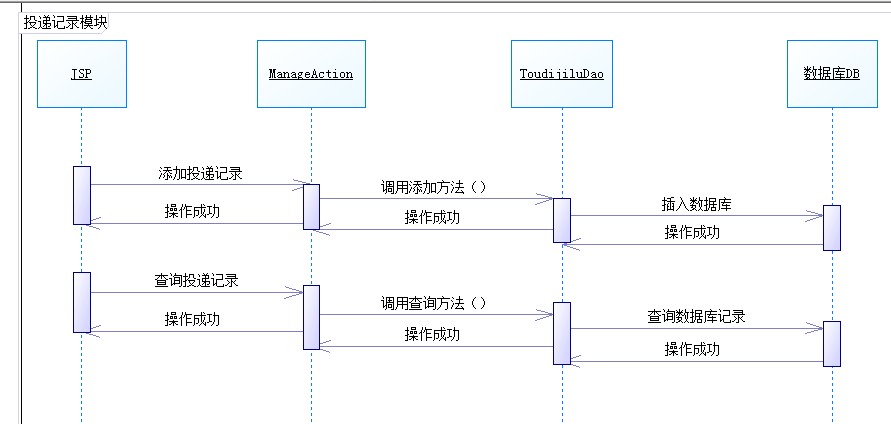


图5-16 投递记录模块程序流程图

### 1.2.6 文档模块时序图

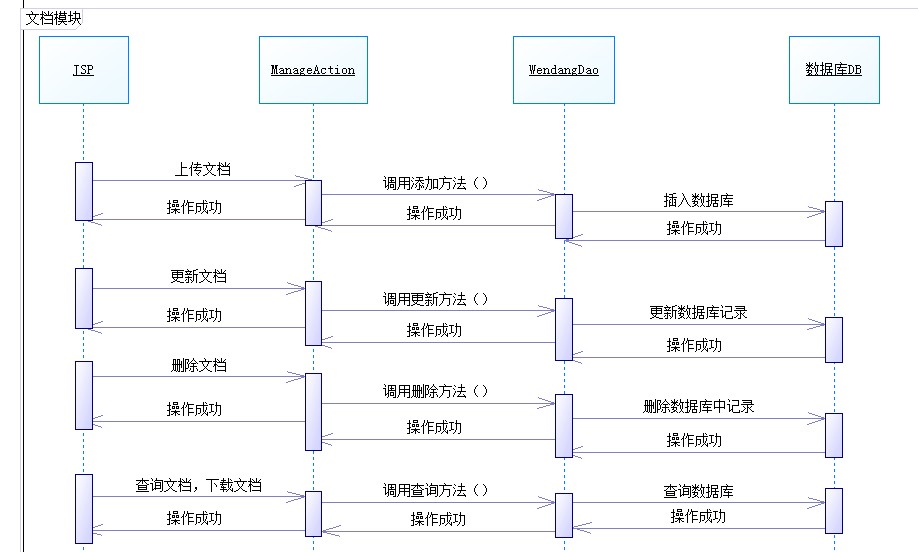


图5-17 文档模块时序图

### 1.2.7 系别模块时序图

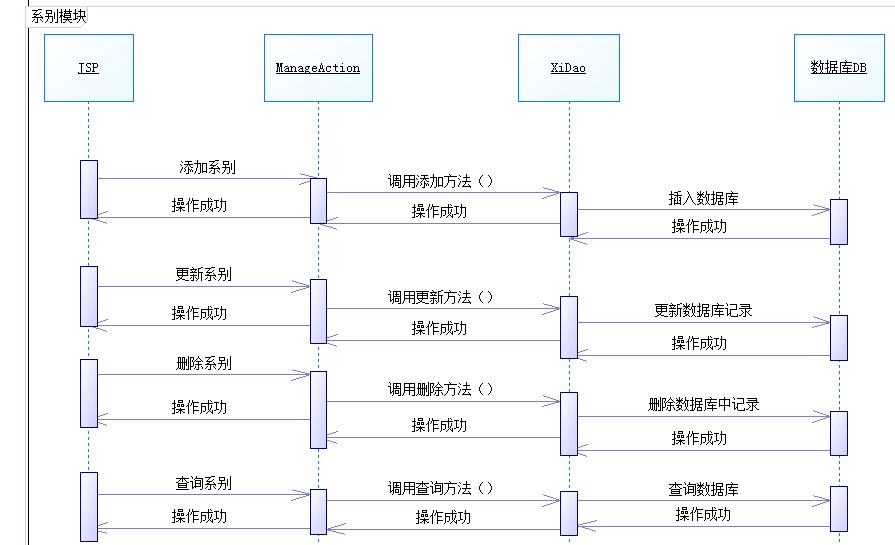
-

图5-18 系别模块时序图

### 1.2.8 新闻模块时序图

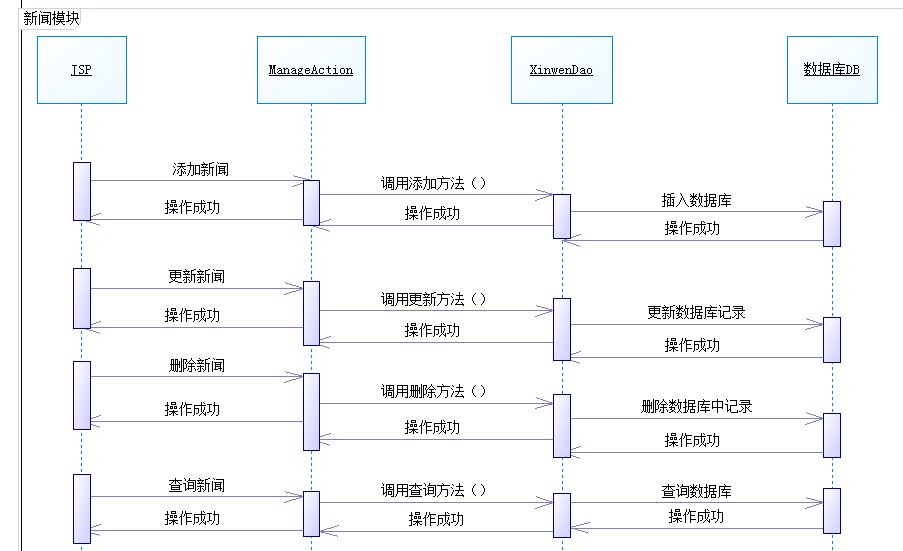


图5-19 新闻模块时序图

### 1.2.9 用户模块时序图

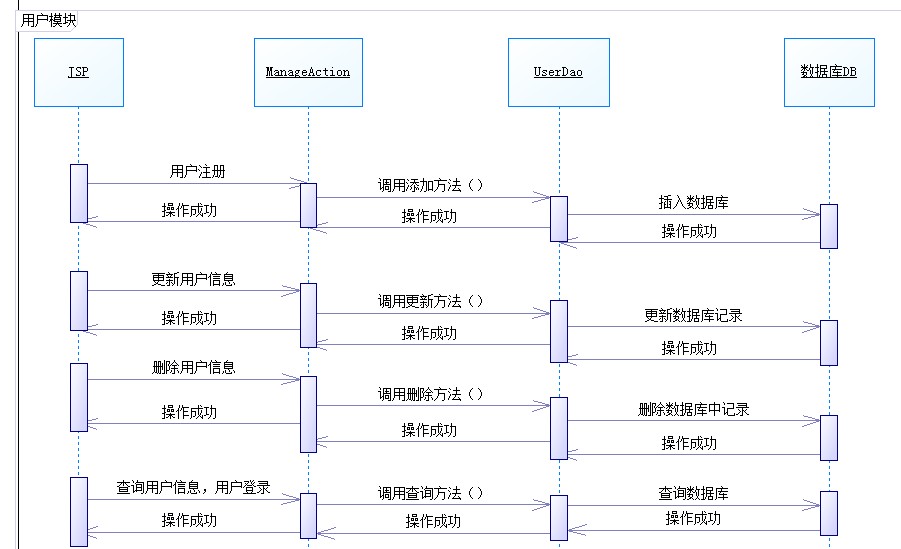


图5-20 用户模块时序图

## 1.3 系统功能实现的主要框架

### 1.3.1 SSH框架

SSH框架就是struts2,spring,hibernate的简称，运用3个框架的集成协作使用，可以有效的提高代码编写的效率，从而有效的降低程序开发的周期。

### 1.3.2 hibernate配置文件

<hibernate-configuration>

<session-factory>

<!--Database connection settings-->

<property name="proxool.pool\_alias">DBPool</property>

<property name="proxool.xml">proxool.xml</property>

<property name="connection.provider\_class">

org.hibernate.connection.ProxoolConnectionProvider

</property>

<!--Echo all executed SQL to stdout-->

<property name="show\_sql">true</property>

<property name="format\_sql">true</property>

<property name="dialect">org.hibernate.dialect.MySQLDialect</property>

<!--Enable Hibernate's automatic session context management-->

<property name="current\_session\_context\_class">thread</property>

<property name="hibernate.hbm2ddl.auto">update</property>

<mapping class="model.User"/>

<mapping class="model.Xinwen"/>

<mapping class="model.Zhiwei"/>

<mapping class="model.Jianli"/>

<mapping class="model.Toudijilu"/>

<mapping class="model.Wendang"/>

<mapping class="model.Gonggao"/>

<mapping class="model.Liuyan"/>

<mapping class="model.Jiuye"/>

<mapping class="model.Xi"/>

<mapping class="model.Zhuanye"/>

</session-factory>

</hibernate-configuration>

这是Hibernate的配置文件，在该文件中<property name="proxool.xml">proxool.xml</property>

首先加载了连接池的一个配置，就是加载了数据库的连接信息，

<property name="show\_sql">true</property>

<property name="format\_sql">true</property>

表示控制台打印SQL语句，并格式化SQL语句

<property name="dialect">org.hibernate.dialect.MySQLDialect</property>

表示当前数据库的语言是MYSQL

<mapping class="model.User"/>

<mapping class="model.Xinwen"/>

<mapping class="model.Zhiwei"/>

<mapping class="model.Jianli"/>

<mapping class="model.Toudijilu"/>

<mapping class="model.Wendang"/>

<mapping class="model.Gonggao"/>

<mapping class="model.Liuyan"/>

<mapping class="model.Jiuye"/>

<mapping class="model.Xi"/>

<mapping class="model.Zhuanye"/>

都表示当前数据库建立起来的表和类之间的映射关系，比如<mapping class="model.User"/>表示User类和t\_user表之间的建立起来的对应关系，其他也都类似。

### 1.3.3 struts配置文件

<struts>

<constant name="struts.objectFactory"value="spring"/>

<constant name="struts.i18n.encoding"value="UTF-8"/>

<package name="index"namespace="/"extends="struts-default">

<action name="index"method="index"class="manageAction">

<result name="success1">index.jsp</result>

<result name="success2">login.jsp</result>

</action>

<action name="method"class="manageAction">

<result type="dispatcher">${url}</result>

<result name="redirect"type="redirect">${url}</result>

</action>

</package>

</struts>

这是struts的配置文件，该文件的作用主要起到来自JSP的请求的跳转分配的作用。该struts配置文件是个总的配置文件。

### 1.3.4 spring配置文件

Spring在本项目中主要用到了IOC的功能，就是把对象实例化

<!--daos-->

<bean id="userDao"class="dao.impl.UserDaoImpl">

<property name="sessionFactory"><ref bean="sessionFactory"/></property>

</bean>

<bean id="xinwenDao"class="dao.impl.XinwenDaoImpl">

<property name="sessionFactory"><ref bean="sessionFactory"/></property>

</bean>

<bean id="zhiweiDao"class="dao.impl.ZhiweiDaoImpl">

<property name="sessionFactory"><ref bean="sessionFactory"/></property>

</bean>

<bean id="jianliDao"class="dao.impl.JianliDaoImpl">

<property name="sessionFactory"><ref bean="sessionFactory"/></property>

</bean>

<bean id="toudijiluDao"class="dao.impl.ToudijiluDaoImpl">

<property name="sessionFactory"><ref bean="sessionFactory"/></property>

</bean>

<bean id="wendangDao"class="dao.impl.WendangDaoImpl">

<property name="sessionFactory"><ref bean="sessionFactory"/></property>

</bean>

<bean id="gonggaoDao"class="dao.impl.GonggaoDaoImpl">

<property name="sessionFactory"><ref bean="sessionFactory"/></property>

</bean>

<bean id="liuyanDao"class="dao.impl.LiuyanDaoImpl">

<property name="sessionFactory"><ref bean="sessionFactory"/></property>

</bean>

<bean id="jiuyeDao"class="dao.impl.JiuyeDaoImpl">

<property name="sessionFactory"><ref bean="sessionFactory"/></property>

</bean>

<bean id="xiDao"class="dao.impl.XiDaoImpl">

<property name="sessionFactory"><ref bean="sessionFactory"/></property>

</bean>

<bean id="zhuanyeDao"class="dao.impl.ZhuanyeDaoImpl">

<property name="sessionFactory"><ref bean="sessionFactory"/></property>

</bean>

本段代码是把DAO实例化的代码，通过spring的IOC模式。

<bean id="manageAction"class="action.ManageAction"scope="prototype">

<property name="userDao"><ref bean="userDao"/></property>

<property name="xinwenDao"><ref bean="xinwenDao"/></property>

<property name="zhiweiDao"><ref bean="zhiweiDao"/></property>

<property name="jianliDao"><ref bean="jianliDao"/></property>

<property name="toudijiluDao"><ref bean="toudijiluDao"/></property>

<property name="wendangDao"><ref bean="wendangDao"/></property>

<property name="gonggaoDao"><ref bean="gonggaoDao"/></property>

<property name="liuyanDao"><ref bean="liuyanDao"/></property>

<property name="jiuyeDao"><ref bean="jiuyeDao"/></property>

<property name="xiDao"><ref bean="xiDao"/></property>

<property name="zhuanyeDao"><ref bean="zhuanyeDao"/></property>

</bean>

本段代码是把ManageAction实例化的代码，原理也是spring的IOC模式。

## 1.4 系统功能实现的主要模块

### 1.4.1 登陆页面

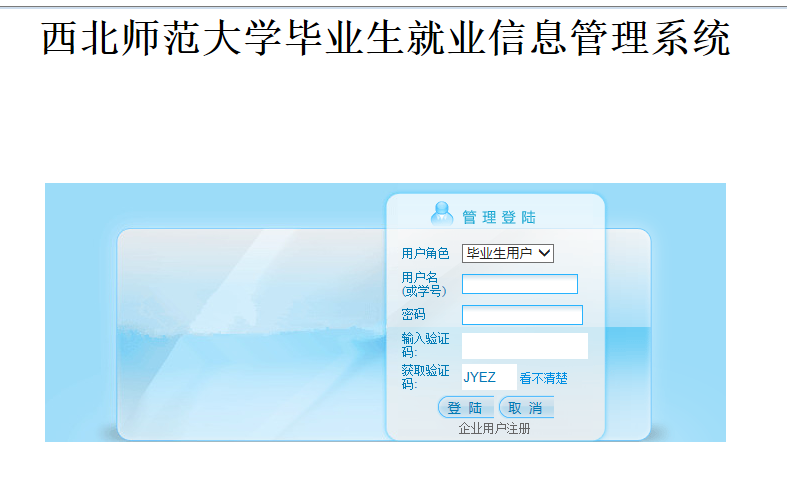


图5-21 登录页面图

登陆代码：

//用户登录操作

**public** String login()**throws** IOException{

HttpServletRequest request=ServletActionContext.*getRequest*();

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

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

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

User user=userDao.selectBean("where username='"+username+"'and password='"+password+"'and userlock=0 and role="+role);

**if**(user!=**null**){

HttpSession session=request.getSession();

session.setAttribute("user",user);

**this**.setUrl("index");

**return**"redirect";

}**else**{

HttpServletResponse response=ServletActionContext.*getResponse*();

response.setCharacterEncoding("gbk");

response

getWriter()

print(

"<script language=javascript>alert('用户名或者密码错误');window.location.href='login.jsp';</script>");

}

**return** **null**;

}

本段代码是程序的登陆代码，首先从获取从页面输入的用户名和密码还有用户的登陆角色，然后调用userDao.selectBean方法，从数据库里取出该条记录进行判断，如果该条记录存在，则说明用户登陆成功，同时把用户信息存在session里面方便调用，如果记录为空，则跳转到登陆页面，同时提醒用户错误信息。

### 1.4.2 用户注册页面

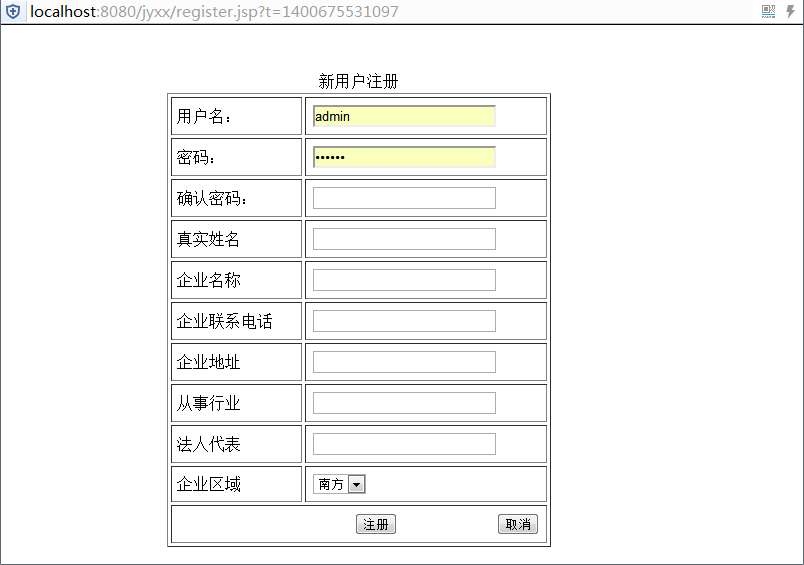


图5-22 用户注册页面

注册代码：

//企业用户注册操作

**public** **void** register()**throws** Exception{

HttpServletRequest request=ServletActionContext.*getRequest*();

HttpServletResponse response=ServletActionContext.*getResponse*();

String username=java.net.URLDecoder.*decode*(request.getParameter("username"),"utf-8");

User user=userDao.selectBean("where username='"+username+"'and userlock=0");

**if**(user==**null**){

user=**new** User();

user.setUsername(username);

user.setPassword(request.getParameter("password"));

user.setTruename(java.net.URLDecoder.*decode*(request.getParameter("truename"),"utf-8"));

user.setQiyedizhi(java.net.URLDecoder.*decode*(request.getParameter("qiyedizhi"),"utf-8")); user.setQiyelianxidianhua(java.net.URLDecoder.*decode*(request.getParameter("qiyelianxidianhua"),"utf-8"));

user.setQiyemingchen(java.net.URLDecoder.*decode*(request.getParameter("qiyemingchen"),"utf-8")); user.setFarendaibiao(java.net.URLDecoder.*decode*(request.getParameter("farendaibiao"),"utf-8")); user.setCongshihangye(java.net.URLDecoder.*decode*(request.getParameter("congshihangye"),"utf-8")); user.setQiyequyu(java.net.URLDecoder.*decode*(request.getParameter("qiyequyu"),"utf-8"));

user.setCreatetime(**new** Date());

user.setRole(1);

userDao.insertBean(user);

response.setCharacterEncoding("utf-8");

response.getWriter().write("注册新用户成功！您的用户名"+user.getUsername()+",请妥善保管！");

}**else**{

response.setCharacterEncoding("utf-8");

response.getWriter().write("该用户名已经存在，请重新注册！");

}

}

本段代码是程序的注册代码，首先从获取从页面获取注册的信息，同时新建一个User对象，把从页面获取的信息都写在USER对象里面。同时在插入到数据库之前先调用userDao.selectBean("where username='"+username+"'and userlock=0");方法，验证下该用户名在数据库中是否存在，如果不存在，则调用userDao.insertBean(bean);方法，把该条记录写入数据库，如果该条记录存在，则返回到注册页面，同时提醒用户该用户已经注册，请不要重复注册。

### 1.4.3 系统管理主页面

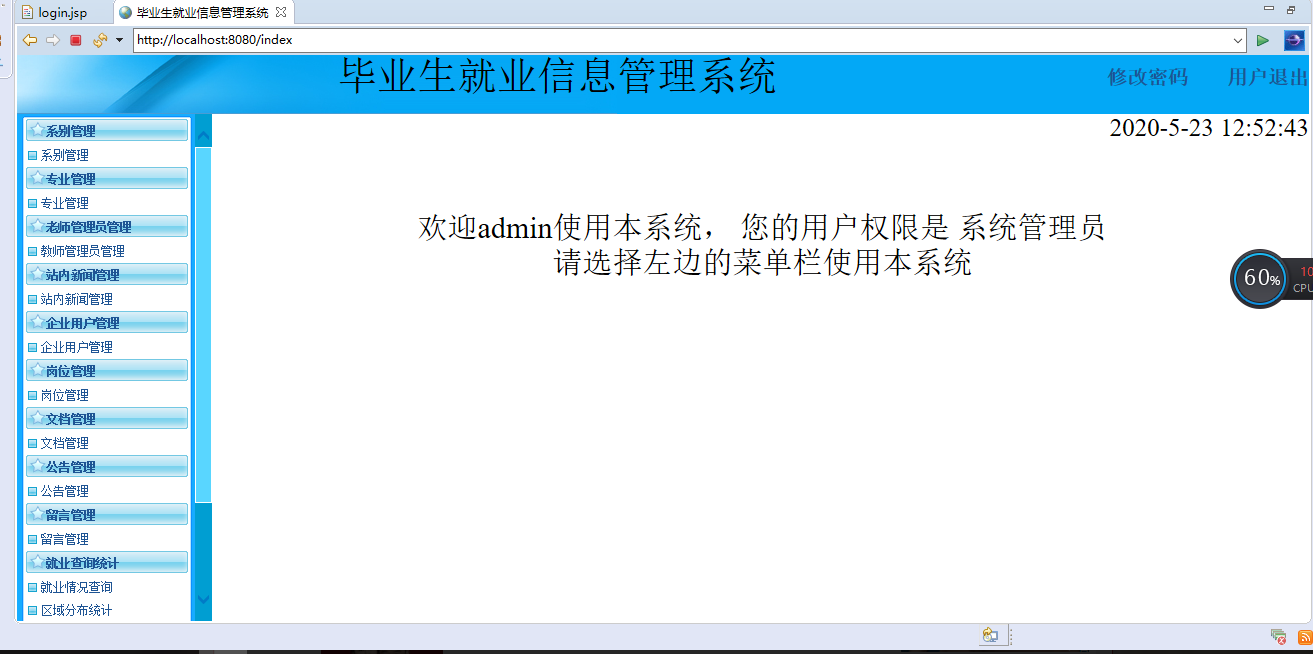


图5-23 系统管理页面

代码：

//程序入口界面

**public** String index(){

HttpServletRequest request=ServletActionContext.*getRequest*();

HttpSession session=request.getSession();

User user=(User)session.getAttribute("user");

**if**(user==**null**){

**return**"success2";

}**else**{

**return**"success1";

}

}

本段代码是主页面代码，判断用户的session的值是否为空，为空则跳转到登陆界面，不为空则根据session的值的不同跳转到不同的主页。

### 1.4.4 教师用户管理主页面

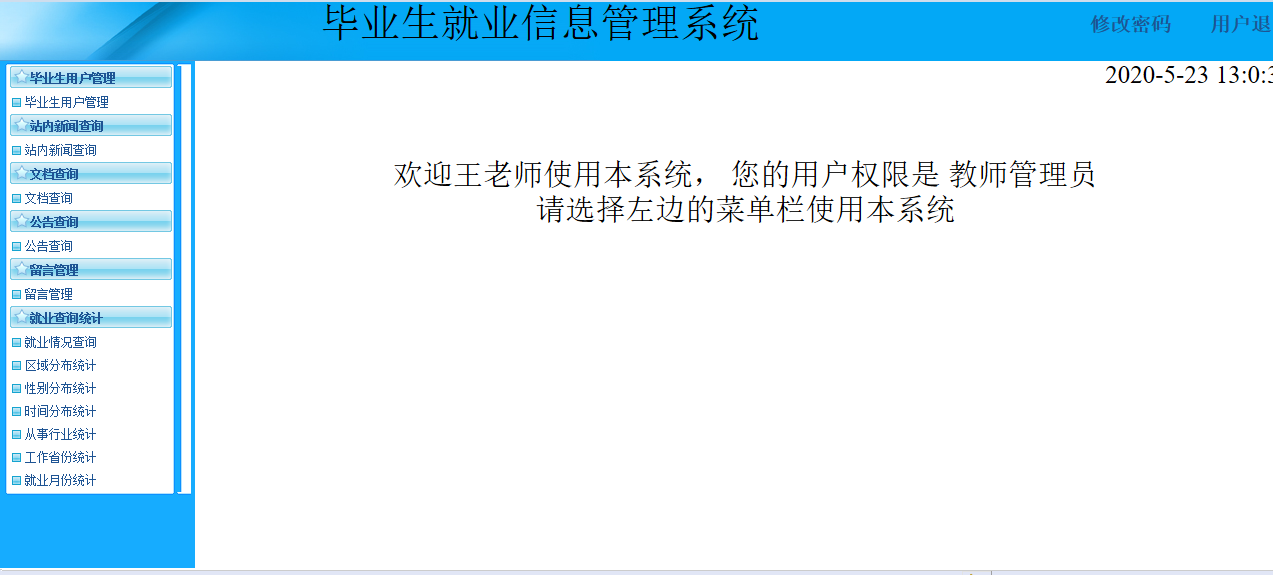


图5-24 教师用户管理主页面

代码：

//老师管理员列表

**public** String userlist(){

HttpServletRequest request=ServletActionContext.*getRequest*();

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

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

StringBuffer sb=**new** StringBuffer();

sb.append("where");

**if**(username!=**null**&&!"".equals(username)){

sb.append("username like'%"+username+"%'");

sb.append("and");

request.setAttribute("username",username);

}

**if**(truename!=**null**&&!"".equals(truename)){

sb.append("truename like'%"+truename+"%'");

sb.append("and");

request.setAttribute("truename",truename);

}

sb.append("userlock=0 and role=3 order by id desc");

String where=sb.toString();

**int** currentpage=1;

**int** pagesize=10;

**if**(request.getParameter("pagenum")!=**null**){

currentpage=Integer.*parseInt*(request.getParameter("pagenum"));

}

**int** total=userDao.selectBeanCount(where);

request.setAttribute("list",userDao.selectBeanList((currentpage-1)

\*pagesize,pagesize,where));

request.setAttribute("pagerinfo",Pager.*getPagerNormal*(total,pagesize,

currentpage,"method!userlist","共有"+total+"条记录"));

request.setAttribute("url","method!userlist");

request.setAttribute("url2","method!user");

request.setAttribute("title","老师管理员管理");

**this**.setUrl("user/userlist.jsp");

**return** *SUCCESS*;

}

首先通过String username=request.getParameter("username")和Stringtruename=request.getParameter("truename");这句代码从页面获取查询的选项，，然后通过

if(username!=null&&!"".equals(username))和if(truename!=null&&!"".equals(truename))语句，通过这句条件语句分别判断下这个值是否为空，当不为空时则通过StingBuffer来叠加组装查询的sql语句，最后当sql语句组装好了之后，通过userDao.selectBeanList方法来调用数据库的查询方法来查到符合条件的记录，在通过request.setAttribute方法把查询到的记录传到显示的页面。同时通过this.setUrl("user/userlist.jsp");方法指定传回到具体指定的是哪个JSP页面。通过这样实现对数据查询功能的实现，同时在该老师用户列表页面定义了对老师信息的增加，修改，删除等操作。

### 1.4.5 就业情况查询管理主页面



图5-25 就业情况查询管理主页面

代码：

//就业情况查询

**public** String jiuyelist(){

HttpServletRequest request=ServletActionContext.*getRequest*();

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

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

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

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

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

StringBuffer sb=**new** StringBuffer();

sb.append("where");

StringBuffer sb2=**new** StringBuffer();

sb2.append("where");

**if**(truename!=**null**&&!"".equals(truename)){

sb.append("user.truename like'%"+truename+"%'");

sb.append("and");

sb2.append("user.truename like'%"+truename+"%'");

sb2.append("and");

request.setAttribute("truename",truename);

}

**if**(jiuyefenbu!=**null**&&!"".equals(jiuyefenbu)){

sb.append("jiuyefenbu like'%"+jiuyefenbu+"%'");

sb.append("and");

sb2.append("jiuyefenbu like'%"+jiuyefenbu+"%'");

sb2.append("and");

request.setAttribute("jiuyefenbu",jiuyefenbu);

}

**if**(xingbie!=**null**&&!"".equals(xingbie)){

sb.append("xingbie like'%"+xingbie+"%'");

sb.append("and");

sb2.append("xingbie like'%"+xingbie+"%'");

sb2.append("and");

request.setAttribute("xingbie",xingbie);

}

**if**(congshihangye!=**null**&&!"".equals(congshihangye)){

sb.append("congshihangye like'%"+congshihangye+"%'");

sb.append("and");

sb2.append("congshihangye like'%"+congshihangye+"%'");

sb2.append("and");

request.setAttribute("congshihangye",congshihangye);

}

**if**(zhuanye!=**null**&&!"".equals(zhuanye)){

sb.append("user.zhuanye like'%"+zhuanye+"%'");

sb.append("and");

sb2.append("user.zhuanye like'%"+zhuanye+"%'");

sb2.append("and");

request.setAttribute("zhuanye",zhuanye);

}

HttpSession session=request.getSession();

User user=(User)session.getAttribute("user");

**if**("admin".equals(user.getUsername())){

sb.append("1=1 order by id desc");

sb2.append("1=1");

}**else**{

sb.append("laoshi.id="+user.getId()+"order by id desc");

sb2.append("laoshi.id="+user.getId());

}

String where=sb.toString();

String where2=sb2.toString();

**int** currentpage=1;

**int** pagesize=10;

**if**(request.getParameter("pagenum")!=**null**){

currentpage=Integer.*parseInt*(request.getParameter("pagenum"));

}

**int** total=jiuyeDao.selectBeanCount(where2);

request.setAttribute("list",jiuyeDao.selectBeanList((currentpage-1)\*pagesize,pagesize,where));

request.setAttribute("pagerinfo",Pager.*getPagerNormal*(total,pagesize,

currentpage,"method!jiuyelist","共有"+total+"条记录"));

request.setAttribute("url","method!jiuyelist");

**this**.setUrl("jiuye/jiuyelist.jsp");

**return** *SUCCESS*;

}

首先通过以下语句：

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

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

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

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

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

这些代码从页面获取查询的选项，，然后通过

if(truename!=null&&!"".equals(truename)){

if(jiuyefenbu!=null&&!"".equals(jiuyefenbu)){

if(xingbie!=null&&!"".equals(xingbie)){

if(congshihangye!=null&&!"".equals(congshihangye)){

if(zhuanye!=null&&!"".equals(zhuanye)){

通过这句条件语句分别判断下这个值是否为空，当不为空时则通过StingBuffer来叠加组装查询的sql语句，最后当sql语句组装好了之后，通过jiuyeDao.selectBeanList方法来调用数据库的查询方法来查到符合条件的记录，在通过request.setAttribute方法把查询到的记录传到显示的页面。同时通过this.setUrl("jiuye/jiuyelist.jsp");方法指定传回到具体指定的是哪个JSP页面。通过这样实现对数据查询功能的实现，同时在该就业信息列表了对就业的打印，导出excel等操作。

### 1.4.6 区域分布统计主页面

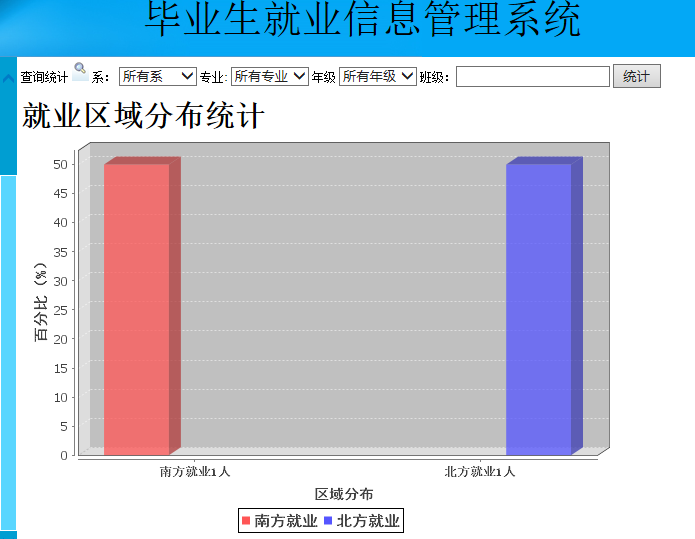


图5-26 区域分布统计主页面

代码：

//统计列表(区域分布统计)

**public** String jiuyelist4()**throws** IOException{

HttpServletRequest request=ServletActionContext.*getRequest*();

request.setAttribute("xilist",xiDao.selectBeanList(0,999,"where xilock=0"));

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

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

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

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

StringBuffer sb=**new** StringBuffer();

sb.append("where");

StringBuffer sb2=**new** StringBuffer();

sb2.append("where");

**if**(banji!=**null**&&!"".equals(banji)){

sb.append("user.banji like'%"+banji+"%'");

sb.append("and");

sb2.append("user.banji like'%"+banji+"%'");

sb2.append("and");

request.setAttribute("banji",banji);

}

**if**(xi!=**null**&&!"0".equals(xi)){

Xi x=xiDao.selectBean("where id="+xi);

sb.append("user.xi like'%"+x.getName()+"%'");

sb.append("and");

sb2.append("user.xi like'%"+x.getName()+"%'");

sb2.append("and");

request.setAttribute("xi",x.getName());

}

**if**(zhuanye!=**null**&&!"0".equals(zhuanye)){

Zhuanye z=zhuanyeDao.selectBean("where id="+zhuanye);

sb.append("user.zhuanye like'%"+z.getName()+"%'");

sb.append("and");

sb2.append("user.zhuanye like'%"+z.getName()+"%'");

sb2.append("and");

request.setAttribute("zhuanye",z.getName());

}

**if**(nianji!=**null**&&!"".equals(nianji)){

sb.append("user.nianji like'%"+nianji+"%'");

sb.append("and");

sb2.append("user.nianji like'%"+nianji+"%'");

sb2.append("and");

request.setAttribute("nianji",nianji);

}

HttpSession session=request.getSession();

User user=(User)session.getAttribute("user");

**if**("admin".equals(user.getUsername())){

sb.append("jiuyefenbu='南方'");

sb2.append("jiuyefenbu='北方'");

}**else**{

sb.append("jiuyefenbu='南方'and laoshi.id="+user.getId());

sb2.append("jiuyefenbu='北方'and laoshi.id="+user.getId());

}

**int** count1=jiuyeDao.selectBeanCount(sb.toString());

**int** count2=jiuyeDao.selectBeanCount(sb2.toString());

**int** count3=count1+count2;

DefaultCategoryDataset dataset=**new** DefaultCategoryDataset();

dataset.addValue(((**double**)count1/count3)\*100,"南方就业","南方就业"+count1+"人");

dataset.addValue(((**double**)count2/count3)\*100,"北方就业","北方就业"+count2+"人");

JFreeChart chart=ChartFactory.*createBarChart3D*(**null**,"区域分布","百分比（%）",dataset,PlotOrientation.*VERTICAL*,**true**,**false**,**false**);

//柱状图(CategoryPlot):

CategoryPlot plot=chart.getCategoryPlot();

//获取图表区域对象

CategoryAxis domainAxis=plot.getDomainAxis();

//水平底部列表

domainAxis.setLabelFont(**new** Font("黑体",Font.*BOLD*,14));

//水平底部标题

domainAxis.setTickLabelFont(**new** Font("宋体",Font.*BOLD*,12));

//垂直标题

ValueAxis rangeAxis=plot.getRangeAxis();

//获取柱状

rangeAxis.setLabelFont(**new** Font("黑体",Font.*BOLD*,15));

chart.getLegend().setItemFont(**new** Font("黑体",Font.*BOLD*,15));

String s=**new** Date().getTime()+"";

request.setAttribute("time",s);

String savaPath=ServletActionContext.*getServletContext*().getRealPath(

"/")

+"/uploadfiles/"+s+".png";

ChartUtilities.*saveChartAsPNG*(**new** File(savaPath),chart,600,400);

**this**.setUrl("jiuye/jiuyelist4.jsp");

**return** *SUCCESS*;

}

统计的图标表示主要通过jfreechart组件来完成的。首先通过以下方法：

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

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

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

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

这句代码从页面获取查询的选项，，然后通过：

if(banji!=null&&!"".equals(banji)){

if(xi!=null&&!"".equals(xi)){

if(zhuanye!=null&&!"".equals(zhuanye)){

if(nianji!=null&&!"".equals(nianji)){

通过这句条件语句分别判断下这个值是否为空，当不为空时则通过StingBuffer来叠加组装查询的sql语句，最后当sql语句组装好了之后，通过jiuyeDao.selectBeanList方法来调用数据库的查询方法来查到符合条件的记录，在把取出来得数据库通过jfreechart组件生成对应的图片，

通过this.setUrl("jiuye/jiuyelist4.jsp");方法指定传回到具体指定的是哪个JSP页面。通过这样实现对数据统计功能的实现。

### 1.4.7 岗位管理主页面



图5-27 岗位管理页面

代码：

//岗位列表

**public** String zhiweilist(){

HttpServletRequest request=ServletActionContext.*getRequest*();

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

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

StringBuffer sb=**new** StringBuffer();

sb.append("where");

StringBuffer sb2=**new** StringBuffer();

sb2.append("where");

**if**(zhiweimingchen!=**null**&&!"".equals(zhiweimingchen)){

sb.append("zhiweimingchen like'%"+zhiweimingchen+"%'");

sb.append("and");

sb2.append("zhiweimingchen like'%"+zhiweimingchen+"%'");

sb2.append("and");

request.setAttribute("zhiweimingchen",zhiweimingchen);

}

**if**(zhiweileibie!=**null**&&!"".equals(zhiweileibie)){

sb.append("zhiweileibie like'%"+zhiweileibie+"%'");

sb.append("and");

sb2.append("zhiweileibie like'%"+zhiweileibie+"%'");

sb2.append("and");

request.setAttribute("zhiweileibie",zhiweileibie);

}

HttpSession session=request.getSession();

User user=(User)session.getAttribute("user");

sb.append("zhiweilock=0 and zhiweifaburen.id="+user.getId()+"order by id desc");

String where=sb.toString();

sb2.append("zhiweilock=0 and zhiweifaburen.id="+user.getId());

String where2=sb2.toString();

**int** currentpage=1;

**int** pagesize=10;

**if**(request.getParameter("pagenum")!=**null**){

currentpage=Integer.*parseInt*(request.getParameter("pagenum"));

}

**int** total=zhiweiDao.selectBeanCount(where2);

request.setAttribute("list",zhiweiDao.selectBeanList((currentpage-1)\*pagesize,pagesize,where));

request.setAttribute("pagerinfo",Pager.*getPagerNormal*(total,pagesize,

currentpage,"method!zhiweilist","共有"+total+"条记录"));

request.setAttribute("url","method!zhiweilist");

request.setAttribute("url2","method!zhiwei");

**this**.setUrl("zhiwei/zhiweilist.jsp");

**return** *SUCCESS*;

}

首先通过String zhiweimingchen=request.getParameter("zhiweimingchen")和 String zhiweileibie=request.getParameter("zhiweileibie");这句代码从页面获取查询的选项，再用if(zhiweimingchen!=null&&!"".equals(zhiweimingchen))if(zhiweileibie!=null&&!"".equals(zhiweileibie))，通过这句条件语句分别判断下这个值是否为空，当不为空时则通过StingBuffer来叠加组装查询的sql语句，最后当sql语句组装好了之后，通过zhiweiDao.selectBeanList方法来调用数据库的查询方法来查到符合条件的记录，在通过request.setAttribute方法把查询到的记录传到显示的页面。同时通过this.setUrl("zhiwei/zhiweilist.jsp");方法指定传回到具体指定的是哪个JSP页面。通过这样实现对数据查询功能的实现，同时在该岗位列表页面定义了对岗位信息的增加，修改，删除等操作。