

中国矿业大学计算机学院

2017 级本科生课程设计报告

课程名称 Web 应用开发技术

报告时间 2020 年 5 月 2 日

学生姓名 陆玺文

学 号 03170908

专 业 计算机科学与技术

任课教师 赵莹

《Web 应用开发技术》评分表

编号	毕业要求	课程教学目标	考查方式与考查点	占比	得分
1	1.3	目标 1: 掌握 Web 应用程序的工作原理, CSS3 的语法, 应用 CSS3+DIV 技术实现页面美化与布局, JavaScript 内置核心语言对象以及基本语法 JSP 页面的基本构成, 掌握 Servlet 的工作原理、技术特点、分类等基础的 Web 应用程序开发的基础知识, 了解通过 Web 技术解决复杂工程问题的基本方法。	1) 针对具体要求, 运用 HTML、CSS、JavaScript、JSP 内置对象设计并开发静态页面和动态页面。 2) 能够针对实际问题, 合理地分析和选择 Web 前端技术与 Web 服务器端开发技术 (JSP、Servlet) 进行 Web 系统设计。	50%	
2	3.4	目标 2: 熟悉目前几种主流的 Web 开发技术, 掌握 4 种内置对象属性范围的区别与应用原则, 掌握 MVCII 模式的原理与特点, 使用该模式进行 Web 应用程序开发。能够针对具体复杂应用问题, 在开发平台上设计实施方案, 完成程序的编码、调试和运行, 并能对结果进行初步的分析和评价。	1) 能够针对实际问题, 合理地分析和选择 Html, 第三方 JavaScript 库解决客户端设计问题, 选择合适的开发模式进行 Web 系统开发。 2) 结合具体较为复杂工程需求, 设计与编程实现、调试, 并能对结果进行初步分析与评价。	50%	
总分				100%	

评阅人:

日期:

摘 要

在本次 Web 应用开发课程设计中，进行了 4 个实验。

➤ 实验一，静态 Web 页面设计。

初次接触 html 与 CSS，模仿了一些知名的新闻门户网站（如百度新闻），做出了“齐心抗疫，共克时艰”专题网站。首页的 JQuery 库实现的轮播图是实验一的一大特色。实验一重点关注在 hmtl 界面与 js 的使用上，使用 js 为注册界面实现了简单验证，同时为登录成功与注册成功页面实现跳转。

➤ 实验二，动态 Web 页面设计。

在实验一的基础上，将部分静态的 html 页面，改成了动态的 jsp 页面，实现了访客数统计，**JavaBean 注入**，session 对象控制**登录注销**。此外，实现了利用 javabean 与 jsp 表单的映射实现了注册验证。

➤ 实验三，Web 数据库实验。

实验三中，实现了 **JDBC** 访问连接数据库，filter 实现权限控制、页面分发，同时利用数据库的增删改查技术打造了小型系统“研招信息网”的纯净版。

➤ 实验四，Web 综合实验。

实验四中，在实验三基础上，进一步完善扩充了“研招信息网”系统，增设留言板、意见反馈模块，主体仍然围绕数据库的增删改查。同时实践了 Web 项目的部署发布。此外，投入更多的精力，学习钻研了 **JeeSite4.1** 开源 ERP 框架，深入探索了代码自动生成模块，成功实践。

关键词: 轮播图 JavaBean 注入 登录注销 研招信息网 JDBC JeeSite4.1

目 录

1、 实验一 静态 WEB 页面设计	1
1.1 实验说明	1
1.1.1 实验目的	1
1.1.2 实验要求	1
1.2 实验内容	1
1.2.1 页面设计	1
1.2.2 CSS 布局与美化	2
1.2.3 JavaScript 验证	5
1.2.4 JavaScript 第三方库的使用	8
1.3 设计心得	10
2、 实验二 动态 WEB 页面设计实验	11
2.1 实验说明	11
2.1.1 实验目的	11
2.1.2 实验要求	11
2.2 实验内容	11
2.2.1 动态页面	11
2.2.2 JavaBean 实现注册验证	12
2.2.3 Session 实现登录注销	14
2.2.4 客户端服务器端界面跳转	15
2.2.5 内置对象作用域	15
2.3 其他展示	16
2.4 设计心得	17
3、 实验三 WEB 数据库实验	18
3.1 实验说明	18
3.1.1 实验目的	18
3.1.2 实验要求	18
3.2 实验内容	18
3.2.1 JDBC 技术连接与访问数据库	18
3.2.2 Filter 实现登录权限控制	20
3.3 小型系统：研招信息网（DAO 模式）	23
3.3.1 数据表设计	23
3.3.2 登录注册（Users 表查询增加）	24
3.3.3 信息展示增加（Info 表增加显示）	29
3.4 其他展示	36
3.5 设计心得	37
4、 实验四 “研招信息网”设计与实现	38
4.1 实验说明	38
4.1.1 实验目的	38

4.1.2 实验要求.....	38
4.2 选题说明	38
4.2.1 背景介绍.....	38
4.2.2 选题.....	38
4.3 需求分析	38
4.4 概要设计	39
4.4.1 层次示意.....	39
4.4.2 静态建模.....	40
4.5 详细设计	43
4.5.1 夏令营信息.....	43
4.5.2 留言板.....	44
4.5.3 意见反馈.....	44
4.6 编码	45
4.6.1 夏令营信息.....	45
4.6.2 留言板.....	45
4.6.3 意见反馈.....	51
4.7 测试	51
4.7.1 单元测试（类测试）	52
4.7.2 集成测试（功能测试）	53
4.8 部署发布	54
4.8.1 安装 JDK	54
4.8.2 安装 Tomcat.....	54
4.8.3 安装 MySQL.....	56
4.8.4 部署项目.....	57
4.8.5 系统演示.....	57
4.9 JEE SITE 4.1 框架学习	58
4.9.1 配置管理定义	58
4.9.2 数据表自动创建.....	58
4.9.3 实体类映射	61
4.9.4 创建菜单.....	62
4.9.5 整体功能.....	63
4.9.6 代码自动生成原理.....	64
4.10 设计心得	65
5、 结课体会	66

1、实验一 静态 Web 页面设计

1.1 实验说明

1.1.1 实验目的

1. 配置 Web(TOMCAT)服务器，了解 Web 工作原理。
2. 熟悉常用 HTML 5 标记的含义，能够熟练使用这些标记设计静态 Web 页面，实现 Web 页面上各种元素的合理布局，如表单、表格、图片以及框架等标记的使用。
3. 了解 CSS 样式表的定义和使用方法，能够使用 CSS 美化和布局 Web 页面。
4. 掌握 JavaScript 脚本语言的基本语法，能够使用 JavaScript 与浏览器对象进行交互。
5. 能够使用 JavaScript 处理表单和表单元素事件。

1.1.2 实验要求

1. 使用 HTML 5 开发 Web 静态页面。按照 HTML 5 的规范设计与开发网站。
2. 练习 HTML 5 的新 HTML5 新特性和效果。
3. 练习使用 Web 页面开发工具(MyEclipse、Dreamwear、VS.NET 或其它)
4. 练习第三方 JavaScript 库的使用。
5. 完成实验报告和实验成果。

1.2 实验内容

1.2.1 页面设计

第一次实验主题为当前的热点话题“抗击新冠肺炎疫情”，因而网站主题定位“齐心抗疫，共克时艰”。围绕这一主题共设计主页、登录页、注册页、注册成功页、登陆成功页共计 5 个 html 页面，同时包含注册验证 js 文件以及特效 js 文件若干。文件目录结构如图 1-2(b)所示。

1.2.1.1 首页 index.html

首页在 (1019×1163 分辨率，缩放 60%) 视图下的整体效果如图 1-1 所示。其中所有链接爬取自百度新闻，可以实现点击跳转。轮播图的思想借鉴于百度新闻。



图 1-1 首页 index.html

1.2.2 CSS 布局与美化

在登录页面 login.html 中，主体部分使用了如图 1-2(a)下所示的结构示意布局。在使用 CSS 美化之后的整日效果如图 1-3 所示。



图 1-2 a.登录页主体布局 b.文件目录（5 个 html 页面）

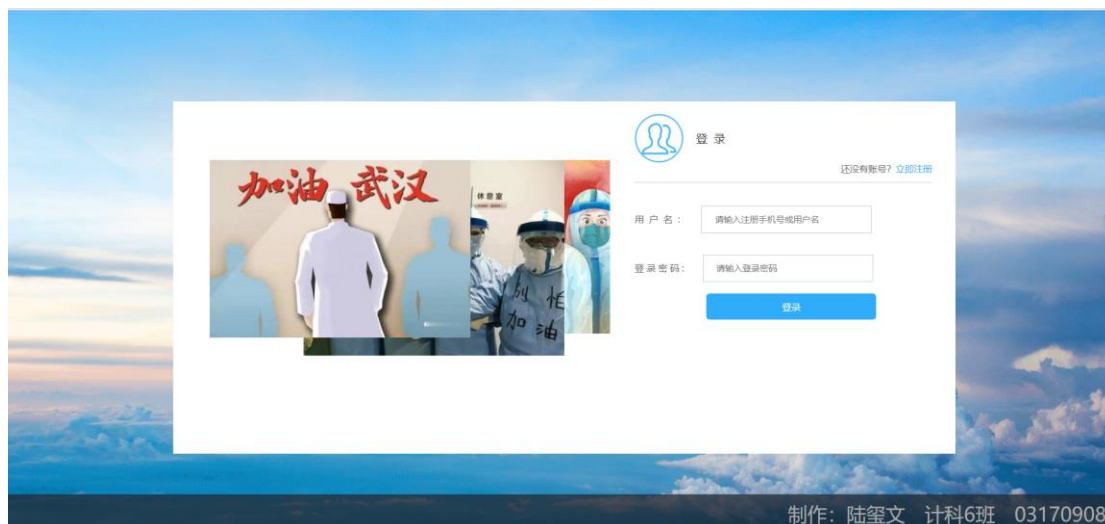


图 1-3 登录页 login.html

其 html 代码与 CSS 布局代码分别如代码 1-1 登录页 HTML 代码代码 1-1, 代码 1-2 所示。

代码 1-1 登录页 HTML 代码

```

1. <body>
2. <div class="main">
3.   <div class="main0">
4.     <div class="main_left">
5.       
6.       
7.       
8.     </div>
9.     <div class="main_right">
10.      <div class="main_r_up">
11.        
12.        <div class="pp">登录</div>
13.      </div>
14.      <div class="sub"><p>还没有账号? <a href="zhuce.html"><span
class="blue">立即注册</span></a></p></div>
15.      <div class="txt">
16.        <span style="letter-spacing:8px;">用户名:</span>
17.        <input name="" type="text" class="txtphone" placeholder="
请输入注册手机号或用户名"/>
18.      </div>
19.      <div class="txt">
20.        <span style="letter-spacing:4px;">登录密码:</span>
21.        <input name="" type="text" class="txtphone" placeholder="

```



```

请输入登录密码"/>
22.         </div>
23.
24.         <div class="xiayibu">登录</div>
25.     </div>
26. </div>
27. </div>
28.
29. <footer>
30.     <div class="footer0">
31.         制作: 陆玺文 计科6班 03170908
32.     </div>
33. </footer>
34. </body>

```

代码 1-2 登录页 CSS 代码

```

1.  *{ margin:0; padding:0}
2.  body{ font-family:"微软雅黑";width:100%; min-width:1200px;
    overflow:hidden}
3.  ul li{ list-style:none}
4.  .blue{ color:#31acfb}
5.  a{ text-decoration:none}
6.
7.
8.  .main{ width:100%; background:url(../images/bk-Login.png) no-repeat;
    background-size:100% 100%; overflow:hidden}
9.  .main0{ width:1200px; height:540px; background:#fff; margin:0 auto;
    margin-top:140px;}
10. .main_left{ float:left; width:647px; position:relative}
11. .main_left img{ position:absolute}
12. .theimg{ top:90px; left:270px}
13. .secimg{top:90px; left:200px}
14. .firing{top:90px; left:56px}
15.
16. .main_right{ width:456px; height:386px; float:right; padding-
    right:36px;}
17. .main_r_up{ height:74px;padding-top:20px;}
18. .main_r_up img{ float:left}
19. .main_r_up .pp{ float:left;height:74px; line-height:74px; font-
    size:18px; color:#333; padding-left:20px;letter-spacing:9px;}
20. .sub{ width:456px; height:30px; font-size:14px; color:#666; border-
    bottom:1px solid #cccccc;}
21. .sub p{ float:right}
22. .txt{ width:456px; height:40px; margin-top:35px}
23. .txt span{ width:90px; font-size:14px; color:#666; height:40px; line-

```

```

height:40px;}
24. .txt input{height:40px; border:1px solid #ccc; padding-left:20px;
margin-left:20px}
25. .txt input.txtphone{ width:240px; }
26. .xiayibu{ width:260px; height:40px; line-height:40px; text-
align:center; color:#fff; background:#30adfa; font-size:14px; border-
radius:5px; margin-left:110px; margin-top:20px; cursor:pointer}
27.
28. footer{ width:100%; height:60px; line-height:60px; position:fixed;
bottom:0; background:url(../images/footerBg.png);background-color:
transparent;}
29. .footer0{ width:1200px; height:60px; margin:0 auto; font-size:20px; c
olor:#adacac}

```

注册页与登录页面相类似，最终的完成效果如图 1-4 所示。

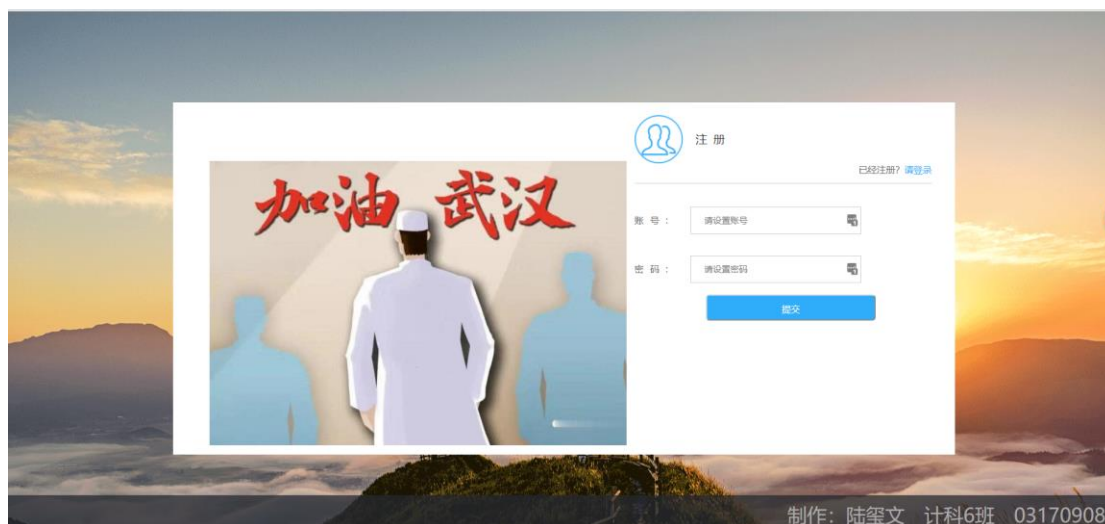


图 1-4 注册页 zhuze.html

1.2.3 JavaScript 验证

在注册页面，规定账号必须仅包含数字和字母，否则注册视为不合法，并通过弹窗提醒用户的注册是否成功，其中注册页中表单代码如代码 1-3 所示，JS 验证代码如代码 1-4 所示。弹窗提醒如图 1-5 所示。

代码 1-3 注册页表单代码

```

1. <form action="" method="post" name="LoginForm" class="info">
2. <div class="main">
3.   <div class="main0">
4.     <div class="main_left">
5.       

```

```
6.     </div>
7.     <div class="main_right">
8.         <div class="main_r_up">
9.             
10.            <div class="pp">注册</div>
11.        </div>
12.        <div class="sub"><p>已经注册? <a href="index.html"><span
class="blue">请登录</span></a></p></div>
13.        <div class="txt">
14.            <span style="letter-spacing:10px;">账号:</span>
15.            <input name="ID" type="text" class="txtphone"
placeholder="请设置账号"/>
16.        </div>
17.        <div class="txt">
18.            <span style="letter-spacing:10px;">密码:</span>
19.            <input name="password" type="password" class="txtphone"
placeholder="请设置密码"/>
20.        </div>
21.        <input type="button" value="提交" class="xiayibu"
onclick="validate(loginForm)">
22.    </div>
23. </div>
24. </div>
25. </form>
```

代码 1-4 注册判断 JS 代码

```
1.  /**用于注册验证
2.   * 满足要求时弹出“注册成功”
3.   */
4.  function validate(f)
5.  {
6.      var name=f.ID.value;
7.      var flag=1;
8.      var RegName=/^[w\d]*$/;
9.      if(!RegName.test(name))
10.     {
11.         alert("用户名只能是数字和字母");
12.         f.ID.focus();
13.         f.ID.select();
14.         flag=-1;
15.     }
16.     if(f.password.value.length<6)
17.     {
18.         alert("密码必须大于6位");
```

```
19.     flag=-1;
20. }
21. if(flag==1)
22. {
23.     alert("注册失败");
24. }
25. else
26.     alert("注册成功");location.href="./zhuceSucc.html";
27. }
```

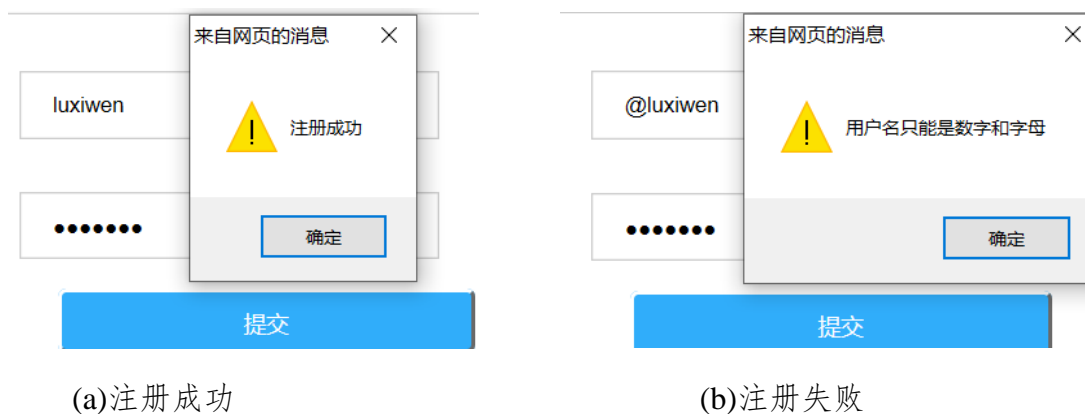


图 1-5 注册成功与失败时弹窗提醒

如代码 1-4 中第 26 行所示，在判断到注册成功之后，整个界面也会跳转到 zhuceSucc.html 界面，如所示。在注册成功页，不执行具体操作，主要用于对用户进行一次提示，同时跳转回首页。关键代码如代码 1-5 所示。



图 1-6 注册成功页 zhuceSucc.html

代码 1-5 注册成功页关键代码

```
1. <div class="main">
2.   <div class="main0">
3.     <div class="font24">注册成功，即将自动跳转
      到首页! </div>
4.       <div class="main_Left">
5.         
6.       </div>
7.     </div>
8.   </div>
9.
10. <script type="text/javascript">
11.   setTimeout(go, 3000);
12.   function go(){
13.     location.href="../index.html";
14.   }
15. </script>
```

登录成功页与注册成功页功能类似，在登录之后进行一次 js 实现跳转，回到首页，其效果如图 1-7 所示。



图 1-7 登陆成功页 loginSucc.html

1.2.4 JavaScript 第三方库的使用

如图 1-8 所示，除了自己所编写的用于注册校验的 JS 代码 loginTest.js 之外，主要使用了 jquery 第三方库，用于完成首页轮播图的展示，如图 1-9 所示，其代码见代码 1-6。

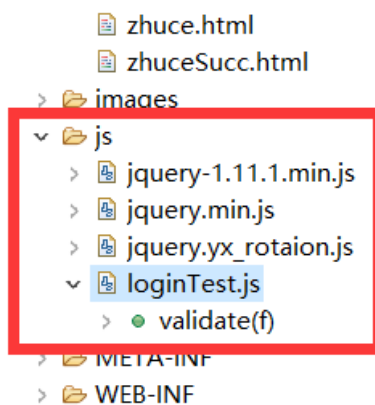


图 1-8 第三方库使用情况



图 1-9 首页轮播图动态效果 (jQuery 库使用)

代码 1-6 轮播图代码

```
1. <!-- 代码 开始 -->
2. <!--UL标签class属性必需，图片alt属性值即标题文字-->
3. <div class="yx-rotation">
4.     <ul class="rotation_list">
5.         <li><a href="#"></a></li>
6.         <li><a href="#"></a></li>
7.         <li><a href="#"></a></li>
```



```
8.         <li><a href="#"></a></li>
9.         <li><a href="#"></a></li>
10.    </ul>
11. <script type="text/javascript" src="js/jquery.min.js"></script>
12. <script type="text/javascript"
    src="js/jquery.yx_rotaion.js"></script>
13. <script type="text/javascript">
14. $(".yx-rotaion").yx_rotaion({auto:true});
15. </script>
16. <!-- 代码 结束 -->
17. </div>
```

1.3 设计心得

在上一学期稍有接触过 HTML，这一次的 Web 课使我更加系统地学习了开发 Web 应用的常用技术，此次实验很好的巩固了通过 HTML 标记语言来搭建网站同时使用 CSS 进行美化的技术。客户端脚本语言 JavaScript 也做了一些简单的尝试。

实验过程中发现了在 CSS 美化这一部分仍然有很大欠缺。在学习借鉴其他互联网资源时，发现他们的 CSS 都做得十分精细，而我美工的能力还是有很大的欠缺，这一点在这次实验中也有所体现。之后的学习过程中，要增加操练，不断加强自己去运用的能力，更好的掌握 Web 开发的技术。能够搭建出更加美观的站点。

2、实验二 动态 Web 页面设计实验

2.1 实验说明

2.1.1 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。

2.1.2 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。

2.2 实验内容

2.2.1 动态页面

将实验一中的部分页面修改为 jsp 动态页面，修改后的目录结构如下如图 2-1 所示。其中首页顶部菜单栏中使用 jsp，增加了网站访问量的统计，效果如图 2-2 所示。相关代码如代码 2-1 所示。

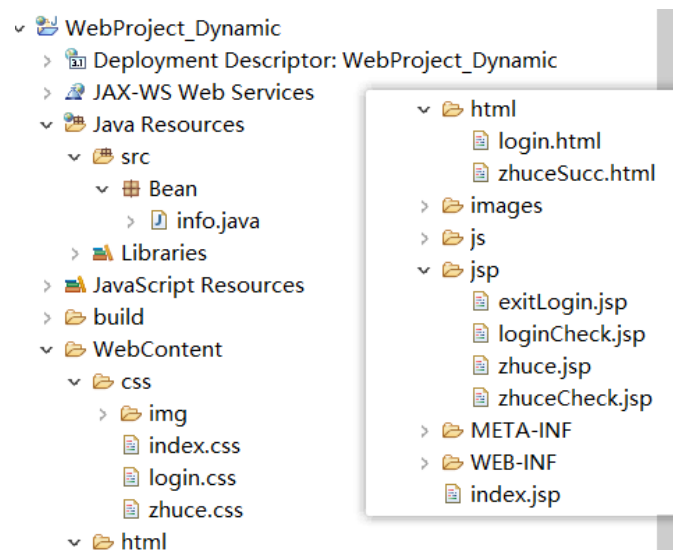


图 2-1 实验二目录结构

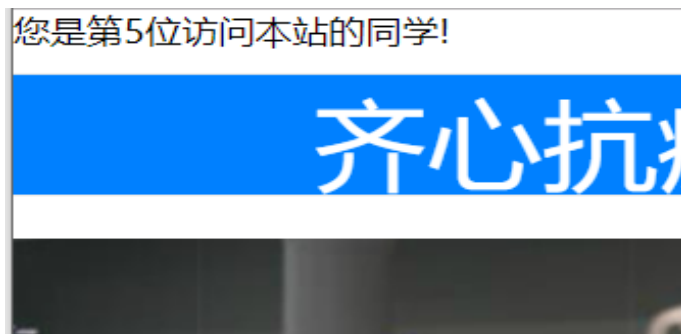


图 2-2 首页访问量统计效果图

代码 2-1 首页 Index.jsp 欢迎词相关代码

```
1. <%!int number = 1;%>
2. <header>
3.     您是第<%=number++%>位访问本站的同学!
4.     <jsp:useBean id="simple" class="Bean.info" scope="session"/>
5.
6.     <%!String str; %>
7.     <%str=simple.getID(); %>
8.     <%if(str!=null){out.print("欢迎您,"+ str+","); %>
9.     <a href="jsp/exitLogin.jsp">注销 </a>
10.    <%} %>
```

2.2.2 JavaBean 实现注册验证

在实验一中，注册校验使用了 javascript 脚本文件进行（代码 1-4），实验二中利用动态 Web 页面的 JavaBean 技术，通过自定义 Bean 类（Bean.info.java），在类中定义注册合法性的校验函数。Info 类代码如代码 2-2 所示。

代码 2-2 info.java

```
1. public class info {
2.     private String ID;
3.     private String password;
4.     private Map<String, String> errors = new HashMap<String,
String>();
5.
6.     public String getID() {
7.         return ID;
8.     }
9.     public String getPassword() {
10.        return password;
11.    }
```

```

12.     public void setID(String id) {
13.         ID = id;
14.     }
15.     public void setPassword(String password) {
16.         this.password = password;
17.     }
18.     public String getErrorMsg(String key) {
19.         return this.errors.get(key);
20.     }
21.     public void setErrors(Map<String, String> errors) {
22.         this.errors = errors;
23.     }
24.     public Map<String, String> getErrors() {
25.         return errors;
26.     }
27.     public boolean isValidate() {
28.         boolean flag = true;
29.         System.out.println("进入Bean,开始验证");
30.         if(!this.ID.matches("\\w{6,15}")) {
31.             flag = false;
32.             this.ID="";
33.             this.errors.put("errID","用户名是6~15位字母或数字");
34.         }
35.         if(!this.password.matches("\\w{6,15}")) {
36.             flag = false;
37.             this.password = "";
38.             this.errors.put("errpass","密码是6~15位字母或数字");
39.         }
40.         System.out.println("进入Bean,结束 验证");
41.         return flag;
42.     }
43. }

```

在注册页的表单中，定义与 info 类属性相同名称的两个表单 Input 元素，分别为 ID 和 password，表单提交后转到 zhuceCheck.jsp 界面进行注册是否成功的判断。在判断界面中，使用的 JavaBean 语句如代码 2-3 所示。

代码 2-3 zhuceCheck.jsp 中 JavaBean

```

1. <jsp:useBean id="simple" scope="session" class="Bean.info"/>
2. <jsp:setProperty property="*" name="simple"/>

```

zhuceCheck.jsp 页面在获得一个 Info 类的实例对象 simple 后，通过调用对象自身的 isValidate() 方法来进行注册有效性的判断，具体代码如代码 2-4 所示。这一步完全替代了实验一中使用 JavaScript 脚本语句进行验证判断（代码 1-4）。

代码 2-4 zhuzeCheck.jsp 中验证调用

```
1. <%  
2. if(simple.isValidate()){  
3. %>  
4. <script type="text/javascript">  
5. alert("Succ");  
6. </script>  
7. <jsp:forward page="../html/zhuzeSucc.html"/>  
8. <%  
9. }else{  
10. %>  
11. <jsp:forward page="../jsp/zhuze.jsp"/>  
12. <%  
13. }  
14. %>
```

2.2.3 Session 实现登录注销

2.2.3.1 登录

在 2.2.2 注册验证通过之后, simple 对象被保存在了 session 的作用域内, 在代码 2-4 中第 7 行, 跳转回主页。同时, 在代码 2-1 中第 8 行, 主页取得了一个有效的 simple 对象的 ID, 进而即可以显示出该用户的姓名。效果如图 2-3 所示。

您是第4位访问本站的同学! 欢迎您,luxiwen, [注销](#)

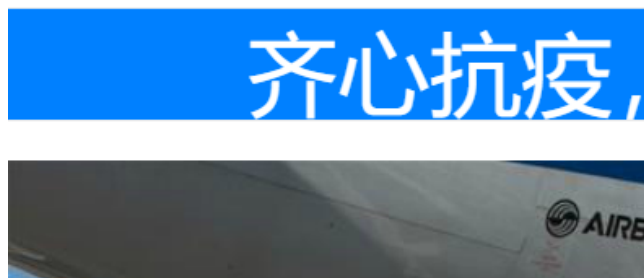


图 2-3 首页欢迎词效果图

类似的, 当登录时, 校验页 loginCheck.jsp 中同样使用了保存在 session 中的 Bean 对象, 并跳转到主页, 也能够在首页显示如图 2-3 所示的效果。其使用代码与代码 2-3 zhuzeCheck.jsp 中 JavaBean 相同。

2.2.3.2 注销

在中可以看到提供了注销超链接, 这一点在代码 2-1 中第 9 行有所体现 (注销), 即当所取得的 simple 对象 ID 有效, 即处于登录状态时, 可以通过 exitLogin.jsp 页面来注销登录。该页面的关键代码如代码 2-5 所示。

代码 2-5 exitLogin.jsp 中关键代码

```
1. <body>
2. <jsp:useBean id="simple" class="Bean.info" scope="session"/>
3. <%session.removeAttribute("simple"); %>
4. <%response.sendRedirect("../index.jsp"); %>
5. </body>
```

首先是获取了 info 类的 simple 对象，接着调用 session 的 removeAttribute() 方法移除该对象，接着再跳转到主页。这样但回到主页的时候，所获得的 simple 对象的 ID 属性值一定是无效的，从而仅显示图 2-2 效果，亦即实现了注销功能。

2.2.4 客户端服务器端界面跳转

2.2.4.1 服务器端跳转

在注册与登录的校验页面，成功或失败之后，均使用了服务器端跳转<jsp:forward>，服务器端跳转的特点在于不会产生新的 request，response。在同一个请求内，可以用 request 来转递参数。

同时，<jsp:forward>后面的语句不会被执行也不会继续发送到客户端。执行速度比较快。

代码 2-4 中第 7、11 行使用了服务器端跳转，均在页面末尾已获得较快响应速度。效果如图 2-4 所示。

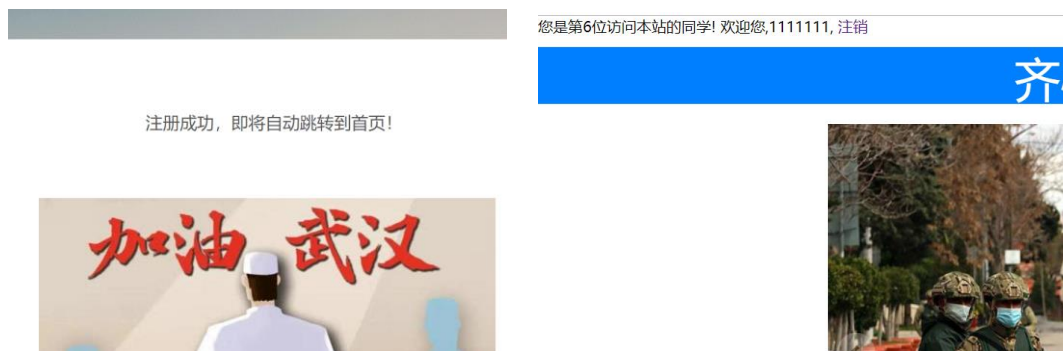


图 2-4 跳转前后效果图

2.2.4.2 客户端跳转

response.sendRedirect()方法即客户端重定向方法，在代码 2-5 第 4 行中，选择了使用客户端跳转。原因在于在删除了 simple 对象后，希望浏览器能够根据 URL 重新发起请求，产生新的 request 和 response。

2.2.5 内置对象作用域

JSP 动态页面中，涉及属性范围的内置对象共有 4 个，分别是 request、pag

e、session、application。其比较如表 2-1 所示。

表 2-1 JSP 的 4 种属性范围内置对象

Page	只能在创建对象的页面中访问
Request	作用在一次请求中
Session	每次会话中，两种跳转都可以获取
Application	保存在整个服务器运作期间

在实验二的开发过程中，因为在首页使用到了欢迎词的功能，同时默认注册成功之后即同时登陆成功，故所用设计 **JavaBean** 的 **Scope** 时，均选用了 **session** 对象的范围，方便于相互之间共同去调用 **simple** 对象。

2.3 其他展示

最初版本的注册在验证失败时，之后跳转回注册界面，缺少提示信息。为了对用户更加友好，在参考了老师的示例程序后，通过在 **info** 类中增加 **errors** 属性，使用 **JavaBean** 技术来设置和获取错误类型，从而在跳转回注册界面时，给与用户相应的提示信息。

效果如图 2-5 所示。在 **zhuce.jsp** 中，用于判断并提示错误信息的 **jsp** 代码段如代码 2-6 所示。



图 2-5 注册错误提示信息

代码 2-6 zhuce.jsp 中错误提醒相关代码

```
1. <%String str=simple.getErrorMsg("errID");if(str!=null){ %>
2.     <script type="text/javascript" lang="javascript">
    alert("<%=str%>"); </script>
3. <%} %>
```

```
4. <%String pass=simple.getErrMsg("errpass");if(pass!=null){ %>
5.     <script type="text/javascript" lang="javascript">
    alert("<%=pass%>"); </script>
6. <%} %>
```

在代码 2-2 的 `isValidate()` 函数中，当与设定条件不符合时，会自动存储相应的错误信息 Map 数据对，当在跳转到 `zhuce.jsp` 页面时，通过判断执行相应的 JavaScript 语句。相关代码见代码 2-6。

2.4 设计心得

这次实验很好的练习了 **JavaBean** 的使用，通过构造 **Bean** 类，可以方便的将表单信息传输到所需要的对象并完成设置。同时，熟悉了客户端与服务器端的跳转，通过一些简单的登录注销，以及访问量统计进一步熟悉了动态网页的搭建构造。

这一次实验，在登录验证部分选择了全部放行，仅为了学习跳转，在接下来的数据库操作实验中期待进一步通过访问数据库，更好的去进行登陆判断。之后的学习过程中，要增加操练，不断加强自己去运用的能力，更好的掌握 **Web** 开发的技术。能够搭建出更加美观的站点。

3、实验三 Web 数据库实验

3.1 实验说明

3.1.1 实验目的

1. 掌握 Servlet 与 Filter 的开发、配置；
2. 熟悉 JDBC 以及 DAO 的概念及工作原理；
3. 能够熟练运用 Servlet+DAO 模式对数据库进行访问，实现数据查询、添加、修改等常用操作；
4. 能够熟练运用 MVC 模式开发 Web 应用程序，实现数据查询、添加、修改等常用操作。

3.1.2 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. 完成实验报告和试验成果。

3.2 实验内容

3.2.1 JDBC 技术连接与访问数据库

使用 JDBC 技术连接访问数据库有如下几个步骤：

1. 加载数据库驱动程序；
2. 连接数据库；
3. 执行操作；
4. 关闭数据库。

在实验三中，为了与 DAO 开发模式相结合，将数据库的相关操作整合在 dbc 包下的 DatabaseConnection.java 类中，如代码 3-1 所示。

代码 3-1 DatabaseConnection.java

```
1. public class DatabaseConnection {
2.     //定义MySQL的数据库驱动程序
3.     private static final String DBDRIVER="com.mysql.cj.jdbc.Driver";
4.     //定义MySQL数据库的连接地址
5.     private static final String
        DBURL="jdbc:mysql://localhost:3306/masterschool?serverTimezone=GMT";
6.     // MySQL数据库的连接用户名
7.     private static final String DBUSER="EricLu";
8.     // MySQL数据库的连接密码
9.     private static final String DBPASS="*****";
```

```

10     private Connection conn;
11
12     public DatabaseConnection() throws Exception{    //无参构造函数
13         Class.forName(DBDRIVER);
14         this.conn=DriverManager.getConnection(DBURL, DBUSER,
15         DBPASS);
16     }
17     public Connection getConn() { //get 方法
18         return conn;
19     }
20     public void setConn(Connection conn) { //set 方法
21         this.conn = conn;
22     }
23     public void close() throws Exception { //close方法
24         if(this.conn != null){
25             try{
26                 this.conn.close() ;
27             }catch(Exception e){
28                 throw e ;
29             }
30         }
31     }

```

在这里需要注意的是，代码 3-1 中的第 3 行，即 MySQL 驱动程序加载时，需要根据所使用的数据库驱动程序 jar 包，找到对应的 Driver 类的路径，将其加载进入，如图 3-1 所示。

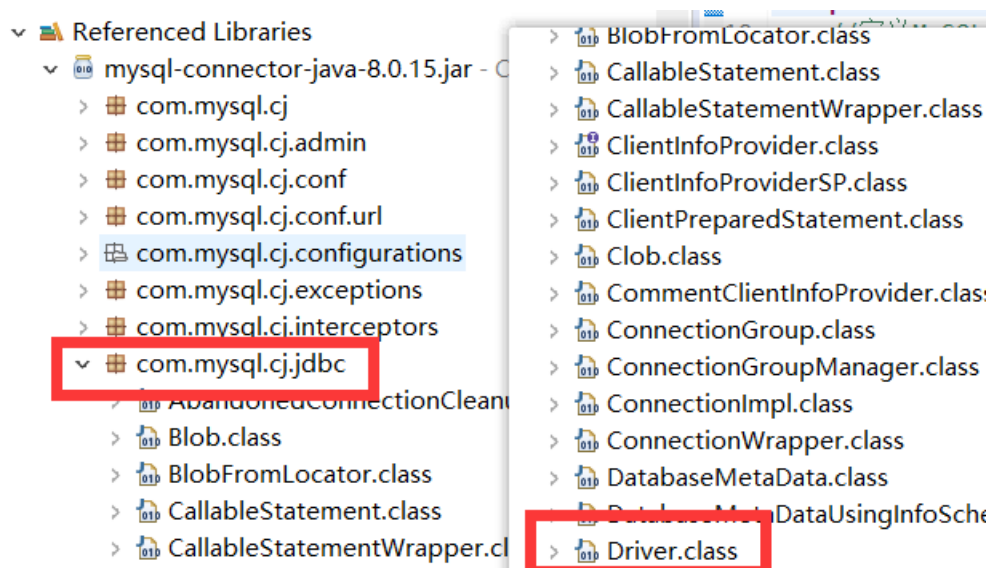


图 3-1 数据库驱动路径

在实验三中，因为使用了最新的驱动程序，故需要在代码 3-1 中的第 5 行 URL 中增加对于时区的说明。编写测试类 testJDBC.java，在其中编写测试方法 testJDBCconnect() 获取连接并打印输出，结果如图 3-2 所示，即连接成功。

```
static void testJDBCconnect() throws Exception {  
    Connection conn = new DatabaseConnection().getConn();  
    System.out.println(conn);  
}
```

<terminated> testJDBC [Java Application] D:\Learns\Java\jre1.8.0_231\bin\j
com.mysql.cj.jdbc.ConnectionImpl@5204062d

图 3-2 测试 JDBC 连接

3.2.2 Filter 实现登录权限控制

Filter 是一种小型的 Web 组件，它们拦截请求和响应，以便查看、提取或以某种方式操作客户机和服务器之间交换的数据。

在实验三中，首页面为 index.jsp 页面，如图 3-3 所示。其中点击访问主页 home.jsp 时，Filter 过滤器会工作，拦截请求，查看当前 session 会话中是否包含有 users.java 类对象 user。



图 3-3 首页 index.jsp

Users.java 类对应数据库中的 Users 表，是用户信息表单，存储已注册过的所有用户。在登录或是注册时，均会向当前的 session 会话中，添加 user 对象，以便于告知过滤器，该用户已登录。过滤器中的拦截代码如下

代码 3-2 所示。

代码 3-2 isLoginFilter 代码

```

1.    public void doFilter(ServletRequest servletRequest, ServletResponse
      servletResponse, FilterChain filterChain) throws IOException,
      ServletException {
2.        // TODO Auto-generated method stub
3.        // place your code here
4.        System.out.println("过滤器开始执行过滤");
5.        HttpServletRequest request = (HttpServletRequest)
      servletRequest;
6.        HttpServletResponse response = (HttpServletResponse)
      servletResponse;
7.
8.        HttpSession session = request.getSession();
9.        String currPath = request.getRequestURI();    //当前请求的URL
10.       System.out.println("当前URL为: "+currPath);
11.       Users user = (Users)session.getAttribute("user");
12.       System.out.println("当前session对象中user对象的ID值为:
      "+user.getID());
13.       if (user.getID() != null) {
14.           String role = user.getRole();
15.           System.out.println("用户已登录, 权限为"+role);
16.           if(role.equals("guest"))response.sendRedirect("homeGuest.jsp");
17. //           response.sendRedirect("../home.html");不能加这句!! 会无限跳转
      报senderror
18.       } else {
19.           System.out.println("还没有登录");
20.           response.sendRedirect("../html/jumplogin.html");
21.       }
22.       // pass the request along the filter chain
23.       filterChain.doFilter(request, response);
24.   }

```

在

代码 3-2 中同时展现了权限控制的逻辑。当用户没有登录时, 会显示如图 3-6 所示的提示跳转到登录界面 jumpLogin.html 界面。在 3 秒之后, 即会自动跳转到登录界面。

当用户登录有效, 即 ID 不为空时, 会对其 role 属性进行判断, 若为宾客”guest”, 则将当前页面跳转到宾客对应的主页 homeGuest.jsp (图 3-4), 否则不跳转则正常进入管理员”admin”对应的主页 home.jsp 中 (图 3-5)。可以看到两者不仅在背景颜色上具有差异, 而且宾客界面没有增加按钮无法增加相应表单信息,

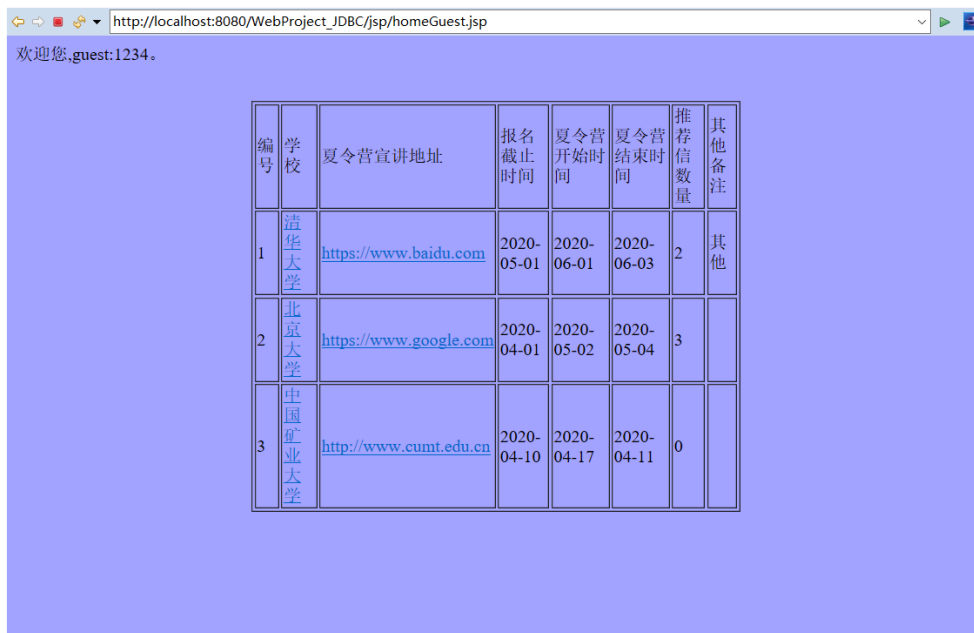


图 3-4 guest 宾客主页

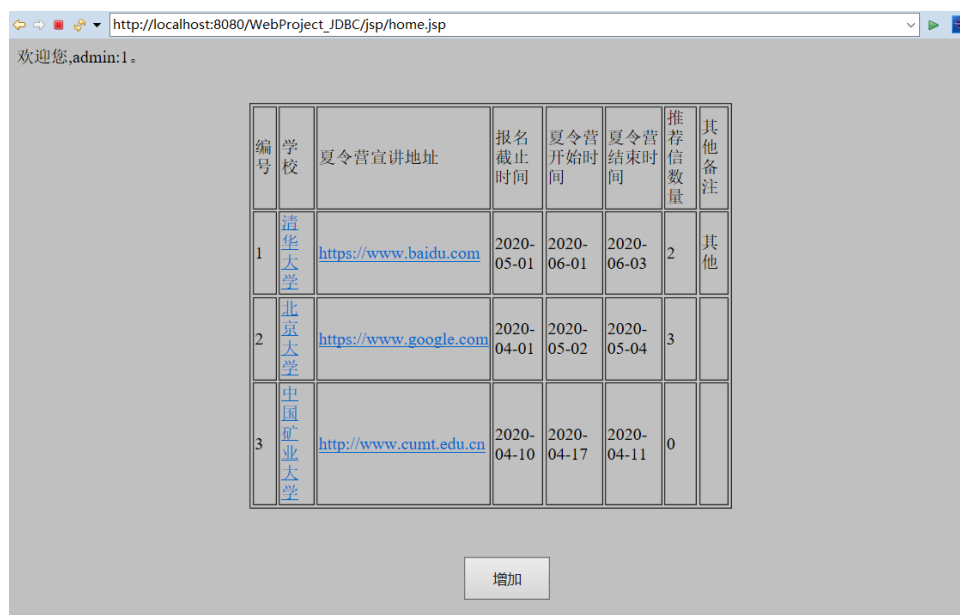


图 3-5 admin 管理员主页

同时，在主页中，使用了实验二中提到过的 **JavaBean** 技术进行带姓名提示的欢迎词，界面对用户更加友好。



图 3-6 jumpLogin 界面

3.3 小型系统：研招信息网（DAO 模式）

在实验三中，为了进一步练习连接访问数据库以及进行相应的数据库操作，结合当下的热点，研究生招生信息，建立一个简单的信息访问查询页面。

3.3.1 数据表设计

在“研招信息网”这一小型系统中，需要一张用户表（Users 表，图 3-7）进行权限控制以及账密管理，一张学校信息表（Info 表，图 3-8）用于存储各个学校的招生信息。

	ID	password	role	integral
▶	1	1	admin	100
	1233	1233	guest	0
	1234	1234	guest	0
	12341	12341	guest	0
	12345	12345	guest	0
	1234567	1234567	guest	0

图 3-7 Users 表

	NO	School	SchoolHomePage	SummerApplyFinishTime	SummerActivityStart	SummerActivityFinish	RecommendationLetterNum	PreachLink	Others
▶	1	清华大学	https://www.tsinghua.edu.cn/	2020-05-01	2020-06-01	2020-06-03	2	https://www.baidu.com	其他
	2	北京大学	https://www.pku.edu.cn/	2020-04-01	2020-05-02	2020-05-04	3	https://www.google.com	
	3	中国矿业大学	http://www.cumt.edu.cn	2020-04-10	2020-04-17	2020-04-11	0	http://www.cumt.edu.cn	

图 3-8 Info 表

其中 Users 表的 integral 属性指代该用户的积分。Info 表的各属性从左至右分别表示学校名称、学校主页、夏令营申请截止时间、夏令营开始时间、夏令营

截止时间、所需推荐信数量、夏令营宣讲地址、其他。

3.3.2 登录注册（Users 表查询增加）

根据 DAO 开发模式，将数据表操作分为 VO、DAO、Impl、Proxy、Factory 五大包。其具体负责模块功能如表 3-1 所示。

表 3-1 DAO 模式代码划分

VO	定义与表字段对应的 JavaBean
DAO	操作的接口（定义数据表原子操作）
Impl	DAO 的真实实现类，完成具体操作
Proxy	代理实现类。数据库连接的打开和关闭
Factory	取得一个 DAO 实例对象

在最终实现的过程中，最终 java 代码结构如图 1-1 所示。

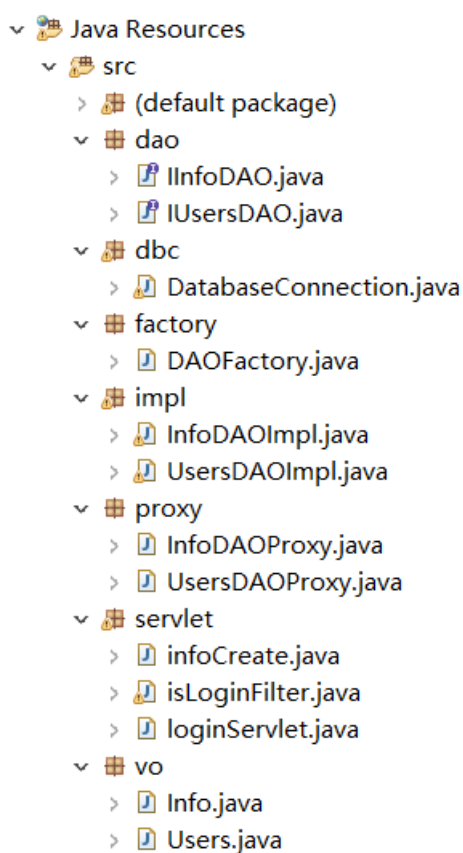


图 3-9 Java 代码结构

3.3.2.1 代码介绍

首先是 VO 包下的 Users.java，定义了与数据表结构对应的 java 类，同时可以提供作为一个 JavaBean 类，其代码与代码 2-2 接近，为各个属性提供了 setter() 和 getter()，同时加入了验证函数。

Dao 包下的 IUsersDAO.java 定义了对该数据表的原子性操作查找和增加，具体代码如代码 3-3 所示。

代码 3-3 IUsersDAO.java

```
1 public interface IUsersDAO {  
2     public boolean doCreate(Users users) throws Exception;  
3     public Users findById(String uid) throws Exception;  
4 }
```

Impl 包下的 UsersDAOImpl.java 给出了根据 ID 查找字段和增加记录的具体实现方法，代码如代码 3-4 所示。其中 doCreate() 方法默认新创建的每一个用户其身份皆为“guest”。

代码 3-4 UsersDAOImpl.java

```
1. public class UsersDAOImpl implements IUsersDAO {  
2.     private Connection conn=null;  
3.     private PreparedStatement pstmt=null;  
4.  
5.     public UsersDAOImpl(Connection conn){  
6.         this.conn = conn;  
7.     }  
8.  
9.     @Override  
10.    public Users findById(String uid) throws Exception {  
11.        Users user = null;  
12.        String sql = "SELECT * FROM Users WHERE ID=?" ;  
13.        this.pstmt = this.conn.prepareStatement(sql);  
14.        this.pstmt.setString(1, uid);  
15.        ResultSet rs = this.pstmt.executeQuery();  
16.        while(rs.next()) {  
17.            user = new Users();  
18.            user.setID(rs.getString(1));  
19.            user.setPassword(rs.getString(2));  
20.            user.setRole(rs.getString(3));  
21.            user.setIntegral(rs.getInt(4));  
22.        }  
23.        this.pstmt.close();  
24.        return user;  
}
```

```

25.     }
26.
27.     public boolean doCreate(Users users) throws Exception{
28.         boolean flag = false;
29.         String sql = "INSERT INTO Users VALUES(?,?,?,?);";
30.         this.pstmt = this.conn.prepareStatement(sql);
31.         this.pstmt.setString(1, users.getID());
32.         this.pstmt.setString(2, users.getPassword());
33.         this.pstmt.setString(3, "guest");
34.         this.pstmt.setInt(4, 0);
35.
36.         if(this.pstmt.executeUpdate()>0) {
37.             flag = true;
38.         }
39.         this.pstmt.close();
40.         return flag;
41.     }
42. }

```

Proxy 包下的 UsersDAOProxy.java 通过实现 IUsersDAO，调用 UsersDAOImpl 并打开关闭数据库，来完成相应的操作，其代码如代码 3-5 所示。

代码 3-5 UsersDAOProxy.java

```

1. public class UsersDAOProxy implements IUsersDAO{
2.     private DatabaseConnection dbc = null;
3.     private IUsersDAO dao = null;
4.
5.     public UsersDAOProxy() throws Exception {
6.         this.dbc = new DatabaseConnection();
7.         this.dao = new UsersDAOImpl(this.dbc.getConn());
8.     }
9.
10.    public boolean doCreate(Users users) throws Exception {
11.        boolean flag = false;
12.        try {
13.            flag = this.dao.doCreate(users);
14.        } catch (Exception e) {
15.            System.out.println("在创建新的Users表单时出现了问题");
16.            e.printStackTrace();
17.        }finally {
18.            this.dbc.close();
19.        }
20.        return flag;
21.    }
22. }

```

```

23.     @Override
24.     public Users findByID(String uid) throws Exception {
25.         Users user = null;
26.         user = this.dao.findByID(uid);
27.         if (user==null) {
28.             System.out.println("查找用户失败, 根据id : "+uid);
29.         }
30.         this.dbc.close();
31.         return user;
32.     }
33. }

```

3.3.2.2 登录

在首页（图 3-3）点击登录之后，即进入登录界面（效果类似图 1-3），提交表单信息之后，使用了 loginServlet 来接收表单信息，同时根据表单所填写 ID，调用代码 3-4 中 findByID()，查找数据库中该用户，若有效则根据其 role 进行相应的跳转，若无效则显示如图 3-10 所示，如代码 3-6 所示。

代码 3-6 loginServlet.java

```

1. @WebServlet(description = "sendURLmatchUser", urlPatterns =
   { "/loginServlet" })
2. public class loginServlet extends HttpServlet {
3.     private static final long serialVersionUID = 1L;
4.
5.     /**
6.      * @see HttpServlet#HttpServlet()
7.      */
8.     public loginServlet() {
9.         super();
10.    }
11.    /**
12.     * @see HttpServlet#doGet(HttpServletRequest request,
13.     HttpServletResponse response)
14.     */
15.     protected void doGet(HttpServletRequest request,
16.     HttpServletResponse response) throws ServletException, IOException {
17.         // TODO Auto-generated method stub
18.         response.getWriter().append("Served at:
19. ").append(request.getContextPath());
20.         Users user = new Users();
21.         String id = request.getParameter("ID");
22.         System.out.println("request参数中获取到的登录用户名为: "+id);
23.         HttpSession session = request.getSession();
24.         try {
25.             user = DAOFactory.getIUsersDAOInstance().findByID(id);

```

```

22.         session.setAttribute("user", user);
23.         if(user==null) {
24.             System.out.println("登录id无效");
25.             response.getWriter().print("\n Please login again!");
26.         }else if (user.getRole().equals("admin")) {
27.             System.out.println("欢迎进入管理员界面");
28.             response.sendRedirect("jsp/home.jsp");
29.         }else if(user.getRole().equals("guest")) {
30.             System.out.println("欢迎进入宾客界面");
31.             response.sendRedirect("jsp/homeGuest.jsp");
32.         }
33.
34.     } catch (Exception e) {
35.         // TODO Auto-generated catch block
36.         e.printStackTrace();
37.         response.sendRedirect("html/login.html");
38.     }
39. }
40. }

```



图 3-10 登录无效提醒界面

3.3.2.3注册

在首页（图 3-3）点击注册之后，即进入注册界面（效果类似图 1-4），提交表单信息之后，转向 zhuceCheck.jsp 页面，该页面主要利用 JavaBean 技术自动注入表单中的各项属性信息，同时调用代码 3-4 中 doCreate()，向数据库中创建该用户，如代码 3-7 所示。

代码 3-7 zhuceCheck.jsp

```

1. <jsp:useBean id="user" scope="session" class="vo.Users"/>
2. <jsp:setProperty property="*" name="user"/>
3.
4. <body>
5. <%if(user.isValidate()){ %>
6. <%
7. //开始向数据库写入该组有效注册
8. try{

```

```

9.      if(DAOFactory.getIUsersDAOInstance().doCreate(user)){
10.          System.out.println("注册用户加入数据库成功");
11.      }
12.  }catch(Exception e){
13.          System.out.println("注册用户加入数据库失败");
14.  }
15.  %>
16.  <script type="text/javascript">
17.  alert("注册成功");
18.  </script>
19.
20.  <jsp:setProperty property="role" name="user" value="guest"
    />
21.  <jsp:forward page="../html/zhuceSucc.html"/>
22.  <%}else{ %>
23.
24.  <jsp:forward page="../jsp/zhuce.jsp"/>
25.  <%} %>
26.  </body>

```

在注册成功之后，即跳转向 zhuceSucc.html 提示用户注册成功，3 秒之后自动转向首页，此时欢迎页利用 session 中保存的 user 对象即可以显示出对应的用户 ID，如图 3-11 所示。



图 3-11 注册成功后首页

3.3.3 信息展示增加（Info 表增加显示）

当用户成功注册或是以 guest 身份成功登陆后，即可以自动跳转向 homeGuest.jsp 界面（图 3-4），在此界面上会自动通过 jsp 获取 Info 表的 DAO 实例化对象，完成数据表的全部查询。

3.3.3.1 代码介绍

在 VO 包下的 Info.java 与 Users.java 相类似，同样完成了从数据表到 Bean 类的映射，所有属性名均与数据表中属性名相同。如代码 3-8 所示。

代码 3-8 Info.java

```
1.  public class Info {
2.      private int NO;
3.      private String School;
4.      private String SchoolHomePage;
5.      private Date SummerApplyFinishTime;
6.      private Date SummerActivityStart;
7.      private Date SummerActivityFinish;
8.      private int RecommendationLetterNum;
9.      private String PreachLink;
10.     private String Others;
11.     public int getNO() {
12.         return NO;
13.     }
14.     public String getSchool() {
15.         return School;
16.     }
17.     public Date getSummerApplyFinishTime() {
18.         return SummerApplyFinishTime;
19.     }
20.     public Date getSummerActivityStart() {
21.         return SummerActivityStart;
22.     }
23.     public Date getSummerActivityFinish() {
24.         return SummerActivityFinish;
25.     }
26.     public int getRecommendationLetterNum() {
27.         return RecommendationLetterNum;
28.     }
29.     public String getPreachLink() {
30.         return PreachLink;
31.     }
32.     public String getOthers() {
33.         return Others;
34.     }
35.     public String getSchoolHomePage() {
36.         return SchoolHomePage;
37.     }
38.     public void setSchoolHomePage(String schoolHomePage) {
39.         SchoolHomePage = schoolHomePage;
```

```

40.     }
41.     public void setNO(int n0) {
42.         NO = n0;
43.     }
44.     public void setSchool(String school) {
45.         School = school;
46.     }
47.     public void setSummerApplyFinishTime(Date
summerApplyFinishTime) {
48.         SummerApplyFinishTime = summerApplyFinishTime;
49.     }
50.     public void setSummerActivityStart(Date
summerActivityStart) {
51.         SummerActivityStart = summerActivityStart;
52.     }
53.     public void setSummerActivityFinish(Date
summerActivityFinish) {
54.         SummerActivityFinish = summerActivityFinish;
55.     }
56.     public void setRecommendationLetterNum(int
recommendationLetterNum) {
57.         RecommendationLetterNum = recommendationLetterNum;
58.     }
59.     public void setPreachLink(String preachLink) {
60.         PreachLink = preachLink;
61.     }
62.     public void setOthers(String others) {
63.         Others = others;
64.     }
65. }

```

在 DAO 下的 IInfoDAO.java 给出了增加和查询所有信息的两种操作声明，供 Impl 以及 Proxy 中类去具体实现。如代码 3-9 所示。

代码 3-9 IInfoDAO.java

```

1. public interface IInfoDAO {
2.     public boolean doCreate(Info info) throws Exception ;//数据插入
3.     public List<Info> findAll() throws Exception ;//完成数据 的查询操作
4. }

```

在 Impl 包下 InfoDAOImpl.java 类具体的实现了增加信息和查询所有信息的数据表操作，如代码 3-10 所示。同时，在第 11 行可以看到，为了实现编号的自动加一，创建数据表时执行了一次记录计数，保证每条记录的编号值唯一递增。

代码 3-10 InfoDAOImpl.java

```

1. public class InfoDAOImpl implements IInfoDAO {

```

```
2.     private Connection conn=null;
3.     private PreparedStatement pstmt=null;
4.
5.     public InfoDAOImpl(Connection conn){
6.         this.conn = conn;
7.     }
8.     @Override
9.     public List<Info> findAll() throws Exception {
10.        List<Info> all = new ArrayList<Info>();
11.
12.        String sql = "SELECT * FROM Info";
13.        this.pstmt = this.conn.prepareStatement(sql);
14.        ResultSet rs = this.pstmt.executeQuery();
15.
16.        Info info=null;
17.        while(rs.next()) {
18.            info=new Info();
19.            info.setNO(rs.getInt(1));
20.            info.setSchool(rs.getString(2));
21.            info.setSchoolHomePage(rs.getString(3));
22.            info.setSummerApplyFinishTime(rs.getDate(4));
23.            info.setSummerActivityStart(rs.getDate(5));
24.            info.setSummerActivityFinish(rs.getDate(6));
25.            info.setRecommendationLetterNum(rs.getInt(7));
26.            info.setPreachLink(rs.getString(8));
27.            info.setOthers(rs.getString(9));
28.            all.add(info);
29.        }
30.        return all;
31.    }
32.    @Override
33.    public boolean doCreate(Info info) throws Exception {
34.        boolean flag = false;
35.        String sql = "INSERT INTO Info VALUES(?,?,?,?,?,?,?,?,?)";
36.        //为了实现编号自动加一
37.        String sql_count = "SELECT count(*) num from info;";
38.        Statement stmt = this.conn.createStatement();
39.        ResultSet rs_count = stmt.executeQuery(sql_count);
40.        rs_count.next();
41.        int row_count = rs_count.getInt("num");
42.        stmt.close();
43.
44.        this.pstmt = this.conn.prepareStatement(sql);
45.        this.pstmt.setInt(1, row_count+1);
```

```

46.         this.pstmt.setString(2, info.getSchool());
47.         this.pstmt.setString(3, info.getSchoolHomePage());
48.         this.pstmt.setDate(4, info.getSummerApplyFinishTime());
49.         this.pstmt.setDate(5, info.getSummerActivityStart());
50.         this.pstmt.setDate(6, info.getSummerActivityFinish());
51.         this.pstmt.setInt(7, info.getRecommendationLetterNum());
52.         this.pstmt.setString(8, info.getPreachLink());
53.         this.pstmt.setString(9, info.getOthers());
54.
55.         if(this.pstmt.executeUpdate()>0) {
56.             flag = true;
57.         }
58.         this.pstmt.close();
59.         return flag;
60.     }

```

Proxy 包中对于 Info 类的代理类 InfoDAOProxy.java 同 UsersDAOProxy.java 相类似，继承了 IInfoDAO 接口，调用了 InfoDAOImpl，负责了数据库的关闭和打开。

最后的 Factory 包中，DAOFactory.java 包含两个函数，分别与取得一个 DAO 的实例对象，也是最终在 jsp 页面中去调用的函数。如代码 3-11 所示。

代码 3-11 DAOFactory.java

```

1. import dao.*;
2. import proxy.*;
3.
4. public class DAOFactory {
5.     public static IInfoDAO getIInfoDAOInstance() throws Exception{
6.         return new InfoDAOProxy();
7.     }
8.
9.     public static IUsersDAO getIUsersDAOInstance() throws Exception {
10.        return new UsersDAOProxy();
11.    }
12. }

```

3.3.3.2 信息展示

homeguest.jsp 中用于取得所有信息并罗列的代码如代码 3-12 所示，通过 DAOFactory 获得一个实例化的对象之后，通过调用对象的 FindAll()方法完成所有信息的查找，最后使用迭代器循环输出 table 格式的信息，在表单中展现。最后的显示效果如图 3-4 所示。

代码 3-12 homeGuest.java

```
1.  <%
2.  List<Info> all = DAOFactory.getIInfoDAOInstance().findAll();
3.  Info info = null;
4.  Iterator<Info> iter = all.iterator();
5.  %>
6.
7.  <form action="stu_list.jsp" method="post" style="position:
   absolute;left: 50%;top: 60px;transform: translate(-50%,0);">
8.  <table border="1">
9.  <tr>
10.
11.  <td>编号</td>
12.  <td width="30">学校</td>
13.  <td>夏令营宣讲地址</td>
14.  <td width="60">报名截止时间</td>
15.  <td>夏令营开始时间</td>
16.  <td>夏令营结束时间</td>
17.  <td>推荐信数量</td><td>其他备注</td>
18.  </tr>
19.
20.  <%
21.  while(iter.hasNext()){
22.  Info info2 = iter.next() ;
23.  %>
24.  <tr>
25.
26.  <td><%=info2.getNO()%></td>
27.  <td><a
   href=<%=info2.getSchoolHomePage()%><%=info2.getSchool()%></a></td>
28.  <td><a
   href=<%=info2.getPreachLink()%><%=info2.getPreachLink()%></a></td>
29.  <td><%=info2.getSummerApplyFinishTime()%></td>
30.  <td><%=info2.getSummerActivityStart()%></td>
31.  <td><%=info2.getSummerActivityFinish()%></td>
32.  <td><%=info2.getRecommendationLetterNum()%></td>
33.  <td><%=info2.getOthers()%></td>
34.  </tr>
35.  <%
36.  }
37.  %>
38.  </table>
```

39. </form>

3.3.3.3增加信息

在 home.jsp 页面（图 3-5）中，提供了一个增加的 button 以供用户增加相应的表单信息，在单击了该按钮之后，会导向 insertInfo.html 界面（图 3-12），所有表单属性的名称基本与 Info 类的属性名相对应，更加便捷的完成信息添加功能。

学校:	<input type="text" value="中国矿业大学"/>
夏令营链接:	<input type="text"/>
夏令营报名截止时间:	<input type="text" value="年 / 月 / 日"/>
夏令营开始时间:	<input type="text" value="年 / 月 / 日"/>
夏令营截止时间:	<input type="text" value="年 / 月 / 日"/>
需要推荐信数量:	<input type="text" value="0"/>
研招官网地址:	<input type="text"/>
备注:	<input type="text"/>
<input type="button" value="提交"/>	

图 3-12 insertInfo.html 界面

insertInfo.html 界面最终提交向 infoCreate，这一专门用于增加 info 表单信息的 Servlet 中，其 doGet()方法如代码 3-13 所示。

代码 3-13 infoCreate 中 doGet()

```
1. protected void doGet(HttpServletRequest request, HttpServletResponse
   response) throws ServletException, IOException {
2.     response.getWriter().append("Served at:
   ").append(request.getContextPath());
3.
4.     Info info = new Info();
5.     System.out.print("request参数中获得的School值: ");
6.     System.out.println(request.getParameter("School"));
7.     info.setSchool(new
   String(request.getParameter("School").getBytes("iso-8859-1"), "utf-
   8"));
8.     System.out.print("request参数中获得的School值（修改编码后）: ");
9.     System.out.println(request.getParameter("School"));
10.
```

```
11.         info.setSchoolHomePage(request.getParameter("SchoolHomePage"));
12.
13.
14.         info.setSummerApplyFinishTime(java.sql.Date.valueOf(request.getPara
meter("SummerApplyFinishTime")));
15.
16.         info.setSummerActivityStart(java.sql.Date.valueOf(request.getParame
ter("SummerActivityStart")));
17.
18.         info.setSummerActivityFinish(java.sql.Date.valueOf(request.getParam
eter("SummerActivityFinish")));
19.
20.         info.setRecommendationLetterNum(Integer.parseInt(request.getParamet
er("RecommendationLetterNum")));
21.
22.         info.setPreachLink(request.getParameter("PreachLink"));
23.         info.setOthers(new
String(request.getParameter("Others").getBytes("iso-8859-1"), "utf-
8"));
24.         System.out.println("读取新info表单信息成功");
25.         try {
26.             if (DAOFactory.getIInfoDAOInstance().doCreate(info)) {
27.                 System.out.println("添加入数据库成功");
28.             }
29.             response.sendRedirect("jsp/home.jsp");
30.         } catch (Exception e) {
31.             e.printStackTrace();
32.             response.sendRedirect("html/insertInfo.html");
33.         }
```

信息添加成功即自动跳转向 home.jsp, 此时仍然会触发过滤器, 以便分发界面, 如果添加失败, 则会重新回到 insertInfo.html 的表单界面, 重新填写并提交表单信息。

3.4 其他展示

JDBC 的连接访问中经常会出现许多由 java 代码引发的问题, 可能是数据库语句定义的不规范或是 java 语句相互调用过程中异常抛出所引发的问题。此时维护一个好的服务器端日志输出是非常重要的, 有助于快速定位错误快速纠正错误。

在这一次的实验中，每个类中的关键语句中，我均嵌入了相应的系统输出语句，以便在调试台 **Consoles** 中及时查看服务器的整体运行情况。尤其是在获取类实例对象属性值时，观察是否为空，以及具体属性值如何。图 3-13 展示了用户一开始以 ID “12” 进行登录，数据库查找失败的错误信息。之后该用户进行注册，注册成功。

```
过滤器开始执行过滤
当前URL为: /WebProject_JDBC/jsp/home.jsp
还没有登录
request参数中获取到的登录用户名为: 12
查找用户失败, 根据id : 12
登录id无效
进入Bean, 开始验证
进入Bean, 结束 验证
注册用户加入数据库成功
```

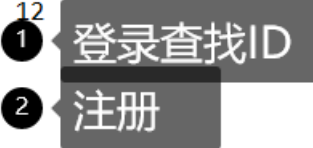


图 3-13 部分后台调试输出语句

类似的服务器端输出调试还有许多，大大便捷了整体的开发过程。

3.5 设计心得

这次实验花费了比较多的精力，部分课堂上没有及时掌握的内容，通过课后再一次的消化吸收理解回放内容，在实践中进一步练习了 **JDBC** 的访问和使用，同时对于 **Servlet** 的运用和 **Filter** 的运用也有了很好地实践和体会。

过滤器确实很大程度上节省了编写重复代码，**DAO** 模式的引用的确是很好地将 **jsp** 页面中的 **java** 语句与 **html** 语句相分离，整个设计都十分精妙。

在实验中，考虑到信息的安全性，没有添加删除操作，相信在能够访问数据库的基础上，具体的语句并不具有难度。在接下来的综合实验中期待进一步完善“研招信息网”这个项目。之后的学习过程中，要增加操练，不断加强自己去运用的能力，更好的掌握 **Web** 开发的技术。能够搭建出更加美观的站点。

4、实验四 “研招信息网”设计与实现

4.1 实验说明

4.1.1 实验目的

1. 深入了解 B/S 模式的基本原理，能够合理划分浏览器端和服务端的功能；
2. 能够运用软件工程的方法，分析设计 Web 应用程序结构；
3. 能够运用 JSP 技术与 MVC 设计模式，依据特定的应用背景，实现步骤 2 中设计的 Web 应用程序；
4. 步骤 3 中实现的 Web 应用程序能够正常部署及发布。

4.1.2 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；

4.2 选题说明

4.2.1 背景介绍

每一年的研究生招生人数都在不断上升，而做为可能是绝大多数人最后一段求学生涯的研究生时光，选择一所适合自己的学校，一位适合自己的导师则显得更加重要。但是在大多数的时候，碍于一些信息的不透明，各所学校信息发布的分散性，导致学生选校选导师过程中不可避免的存在信息不对称成本，这也是直接或间接导致许多研究生后悔甚至于想要退学的原因。

针对这一情形，网络上已经有“保研论坛”、“后保研”等交流场所，然而其针对范围较为广泛，其中所含的信息非常之多，导致在期中搜寻切合自己的信息仍然不够便捷。

4.2.2 选题

故针对我们目前的具体情况，即矿大计算机 17 级的保研学子们，特定制一个信息发布交流的场所，以便大家能够更好地针对性的选择合适的夏令营，选择合适的导师。

所以，第四个实验选择制作面向矿大 17 计算机有志于国内深造同学的“研招信息网”交流平台。

4.3 需求分析

系统功能主要在于满足大家的一些特定类型发布获取以及一些心得体会的交流，所以主要分为“研招信息页”与“留言板”两块。

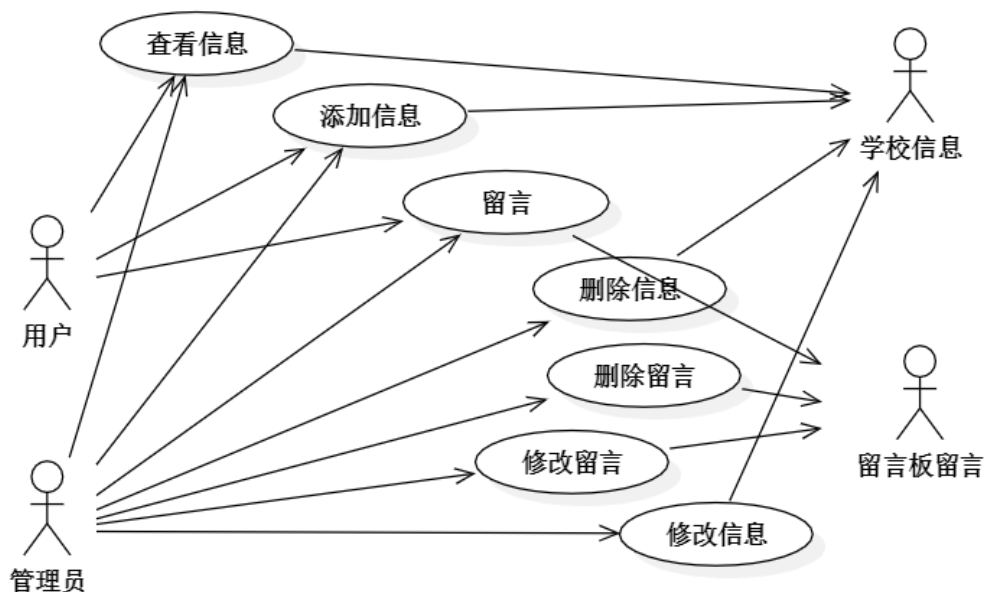


图 4-1 用例图-研招信息网

图 4-1 为系统的整体用例图，展示了在系统中交互的角色以及具体可以获得的功能。

其角色主要为：宾客、管理员。

相关功能主要有：查看信息、增加信息、修改信息、删除信息、留言、删除留言。

4.4 概要设计

采用面向对象的软件设计方法，使用 MVC 架构进行程序编写，每一芯片为一个模型（Model），实现对业务逻辑的处理；液晶屏与键盘代表视图（View），分别用于显示数据和接受用户请求与输入；出租车和计价器类为控制器（Controller），连接模型和视图，调用不同的模型处理用户请求，选择不同视图显示系统的信息。

4.4.1 层次示意

在实验四中，选用了 JSP+Servlet+JavaBean 的 Model2 级开发模式，其示意图如图 4-2 所示。作为一个 Model2 级小型系统，在实验三的基础上，增添留言板交流功能，供大家交流申请心得，吐槽学习生活。此外增添联系作者模块，以便于促进交流反馈。

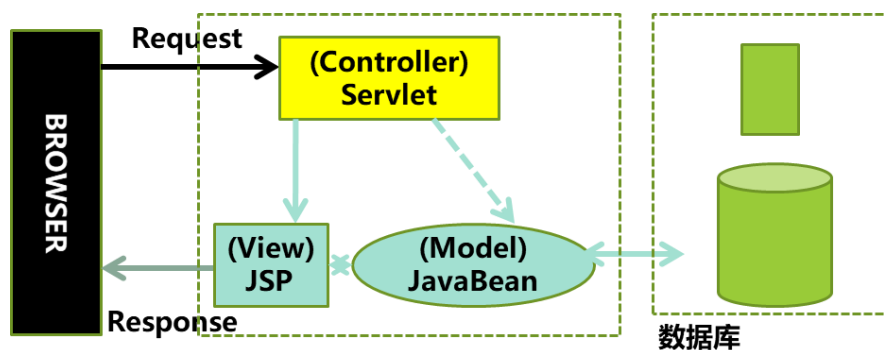


图 4-2 开发模式 JSP+Servlet+JavaBean



图 4-3 系统层次图

4.4.2 静态建模

在 Model2 模型的指导下，程序主体仍旧采用 DAO 模式进行开发，其代码包共包含有：dao、dbc、factory、impl、proxy、servlet、vo。下文展示在静态建模时各个包内的类图设计。

4.4.2.1 DAO 包

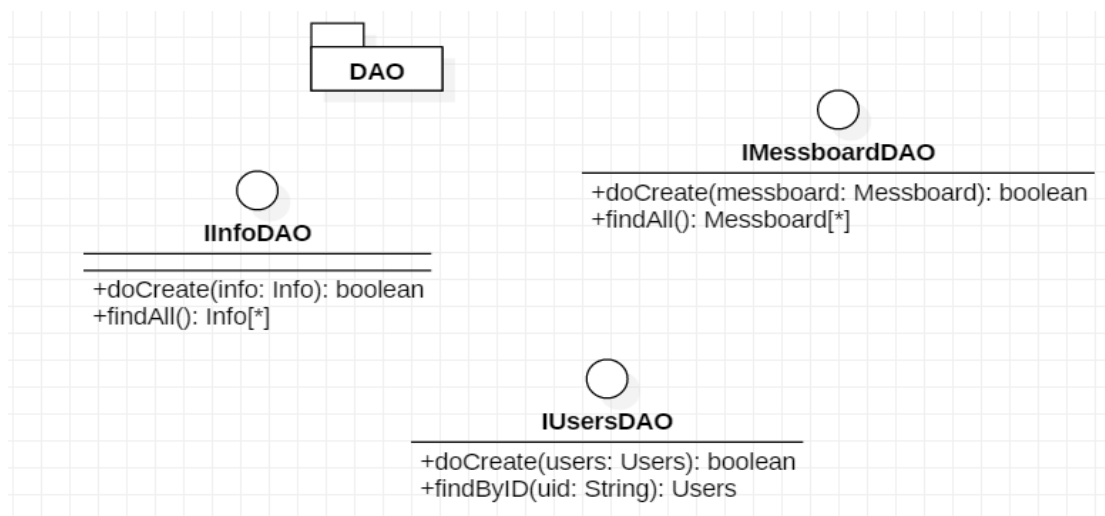


图 4-4 DAO 包类图

4.4.2.2 Proxy 包

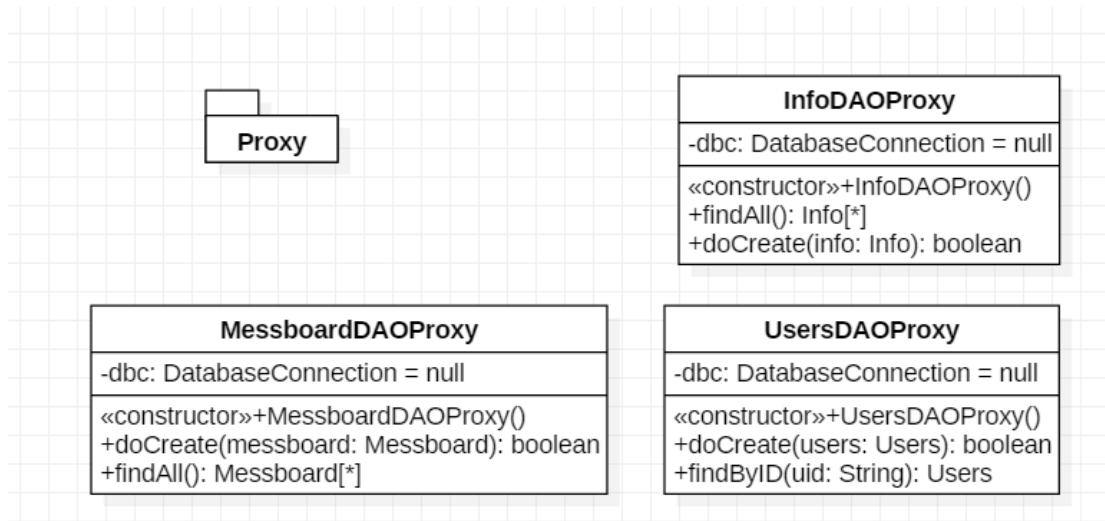


图 4-5 Proxy 包类图

4.4.2.3 Servlet 包

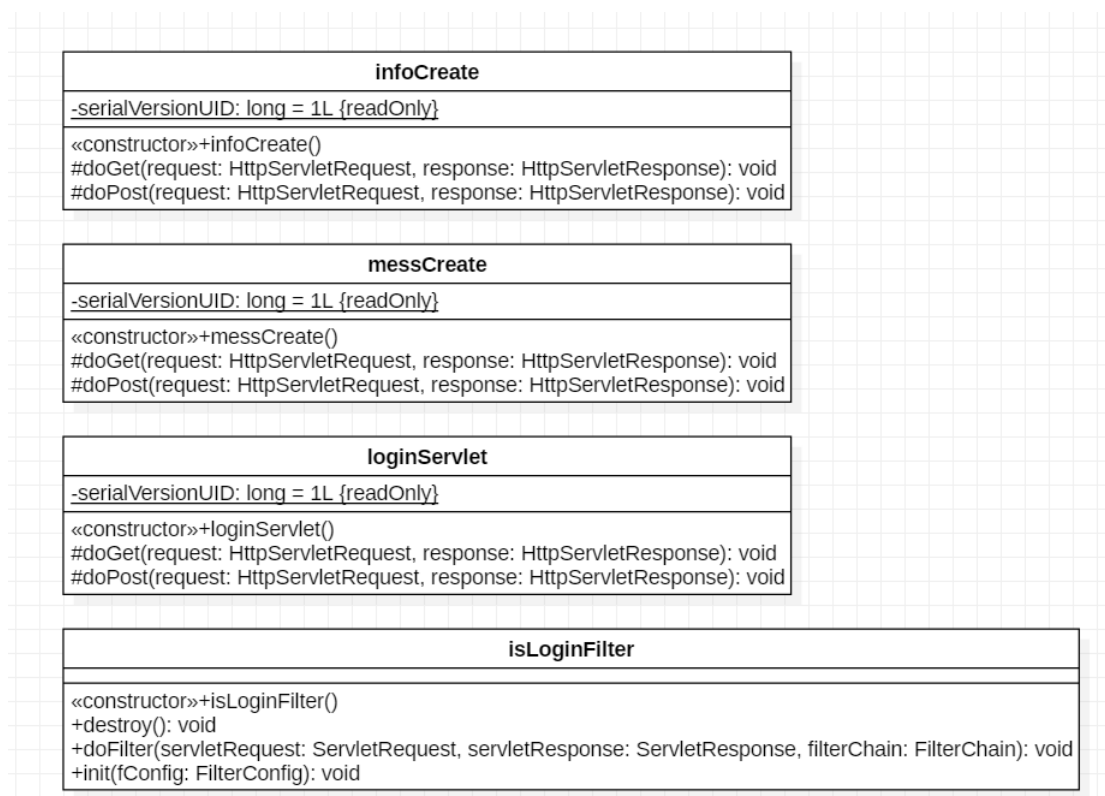


图 4-6 Servlet 包类图

4.4.2.4 VO 包

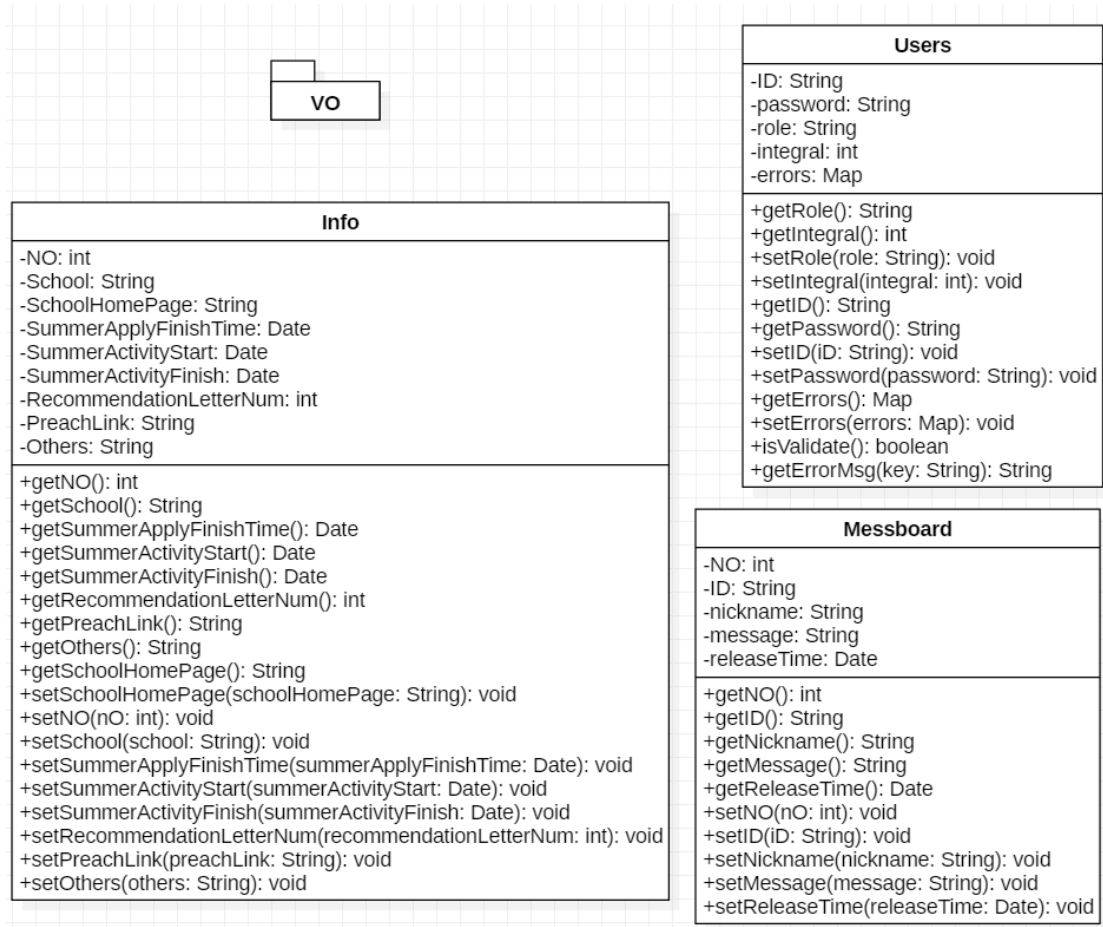


图 4-7 VO 包类图

4.4.2.5 Impl 包

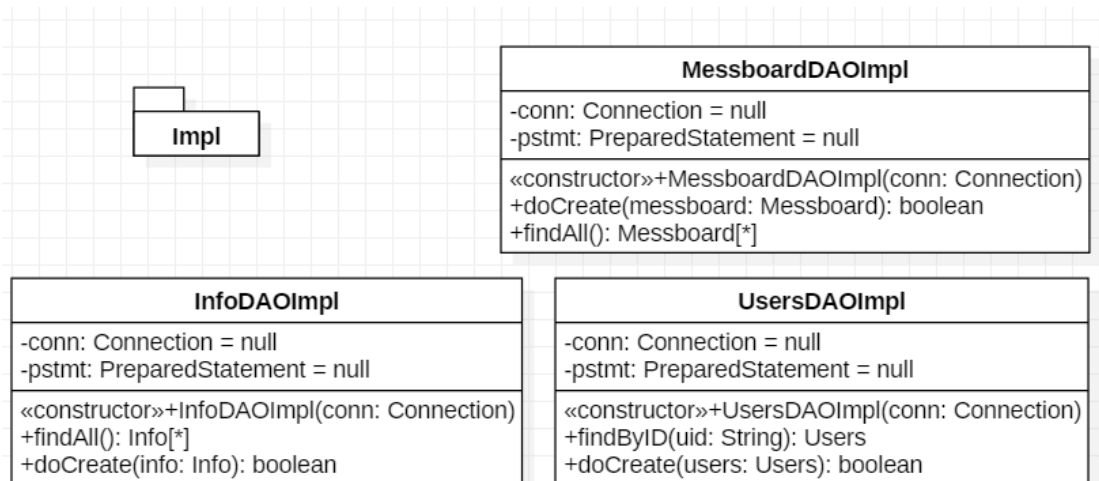


图 4-8 Impl 包类图

4.4.2.6 Factory 包

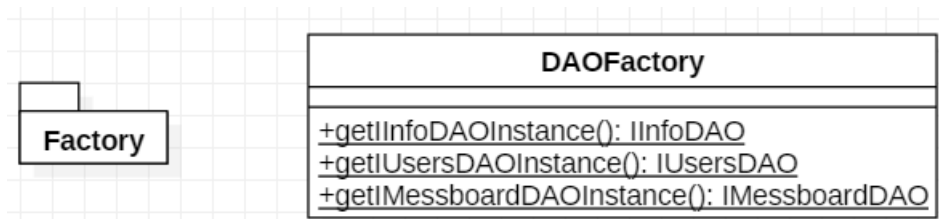


图 4-9 factory 包类图

4.4.2.7 DBC 包

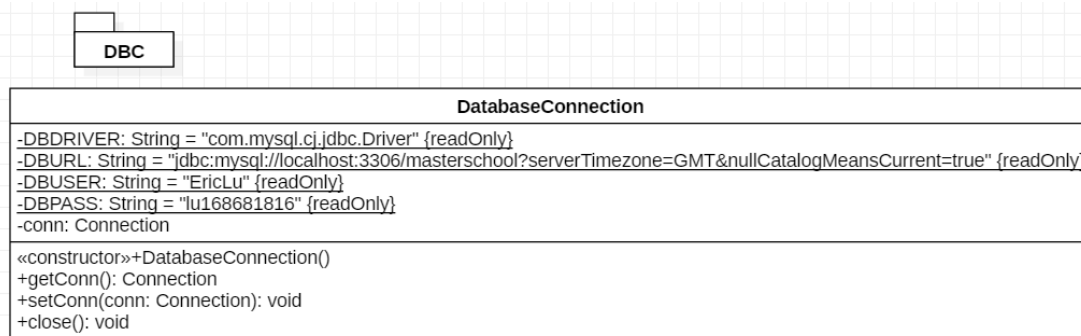


图 4-10 DBC 包类图

4.5 详细设计

在类图的指导下，根据系统层次图（图 4-3），分为 3 大模块进行详细设计。

4.5.1 夏令营信息

在夏令营信息部分，主题上沿用实验三中设计，主要完成夏令营列表展示的安全性鲁棒性。在逻辑判断用户角色身份时，其权限控制流程图如图 4-11 所示。

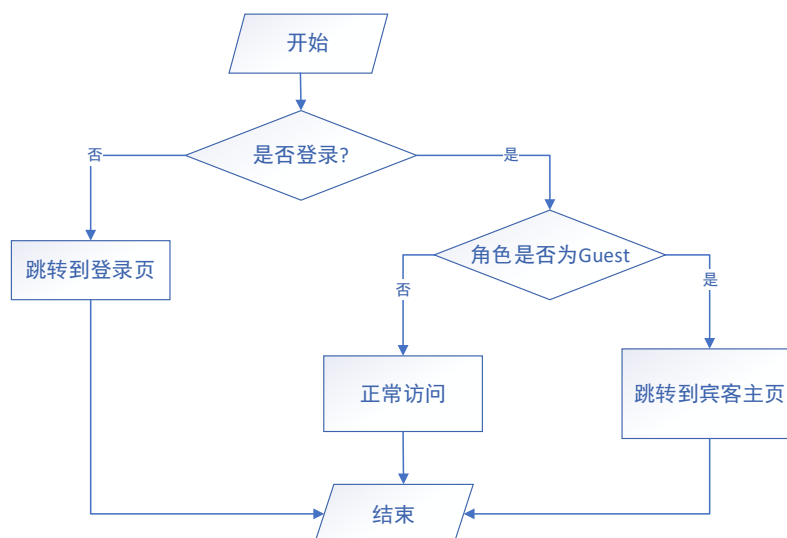


图 4-11 权限控制流程图

4.5.2 留言板

留言板部分的主题逻辑也是对于数据表的增查,在总体逻辑上与 3.3.3 相近,重点需要关注其相互之间的调用协作关系,如图 4-12 所示。

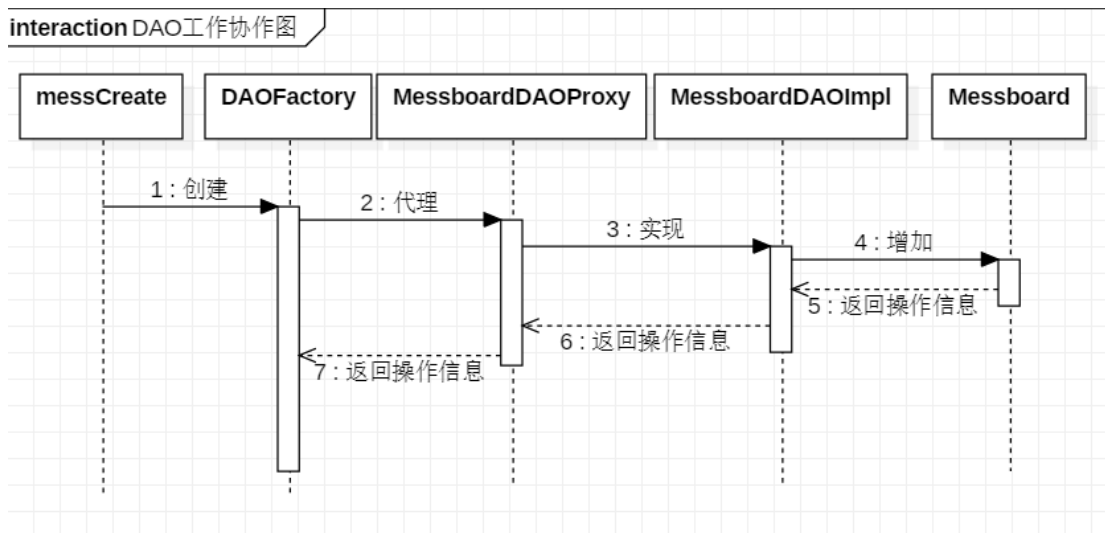


图 4-12 留言板增添时工作协作图

4.5.3 意见反馈

考虑到作为初学者,软件开发与使用过程中难免存在诸多不到位欠缺之处,故增加意见反馈模块供用户在使用过程中及时反馈程序缺漏,以便及时改正。

参考了许多资料后,在软件工程时效性经济性思想的指导下,选择采用“腾讯邮箱邮我”API 接口。接口获取方式可以从“QQ 邮箱-设置-邮我”中获得,如图 4-13 所示。



图 4-13 邮我 API 获取

4.6 编码

整体代码在编写过程中，同样遵循“高耦合低内聚”的开发原则，分模块进行开发设计。

4.6.1 夏令营信息

夏令营信息部分依照 3.3.3 进行编码设计，总体功能架构未做改变，最终展示页效果如图 4-14 所示。



图 4-14 研招夏令营信息

4.6.2 留言板

留言板为新增功能，在代码设计上严格遵照了 Model2 标准，使用 Servlet 作为控制层，DAO 模式作为持久层，jsp 页面作为展示层。

在 VO 包下的 Messboard.java 与 Users.java 相类似，同样完成了从数据表到 Bean 类的映射，所有属性名均与数据表中属性名相同。如代码 4-1 所示。

代码 4-1 Info.java

```

1. public class Messboard {
2.     private int NO;
3.     private String ID;
4.     private String nickname;
5.     private String message;
6.     private Date releaseTime;
7.     public int getNO() {
8.         return NO;
9.     }
10.    public String getID() {
11.        return ID;
12.    }

```

```
13.     public String getNickname() {
14.         return nickname;
15.     }
16.     public String getMessage() {
17.         return message;
18.     }
19.     public Date getReleaseTime() {
20.         return releaseTime;
21.     }
22.     public void setNO(int n0) {
23.         NO = n0;
24.     }
25.     public void setID(String iD) {
26.         ID = iD;
27.     }
28.     public void setNickname(String nickname) {
29.         this.nickname = nickname;
30.     }
31.     public void setMessage(String message) {
32.         this.message = message;
33.     }
34.     public void setReleaseTime(Date releaseTime) {
35.         this.releaseTime = releaseTime;
36.     }
37. }
```

在 DAO 下的 IMessboardDAO.java 给出了增加和查询所有信息的两种操作声明，供 Impl 以及 Proxy 中类去具体实现。如代码 4-2 所示。

代码 4-2 IInfoDAO.java

```
1. public interface IMessboardDAO {
2.     public boolean doCreate(Messboard messboard) throws Exception ;//
    执行数据的插入操作
3.     public List<Messboard> findAll() throws Exception ;//完成数据 的查
    询操作
4. }
```

在 Impl 包下 MessboardDAOImpl.java 类具体的实现了增加信息和查询所有信息的数据表操作，如代码 4-3 所示。同时，在第 11 行可以看到，messboard 表中 NO 属性列使用了自动增加，故在创建新记录时，直接赋值 DEFAULT 即可，

保证每条记录的编号值唯一递增。

代码 4-3 InfoDAOImpl.java

```
1. public class MessboardDAOImpl implements IMessboardDAO {
2.     private Connection conn=null;
3.     private PreparedStatement pstmt=null;
4.
5.     public MessboardDAOImpl(Connection conn) {
6.         this.conn = conn;
7.     }
8.     @Override
9.     public boolean doCreate(Messboard messboard) throws Exception {
10.        boolean flag = false;
11.        String sql = "INSERT INTO Messboard VALUES(DEFAULT,?,?,?,?)";
12.        this.pstmt = this.conn.prepareStatement(sql);
13.        this.pstmt.setString(1, messboard.getID());
14.        this.pstmt.setString(2, messboard.getNickname());
15.        this.pstmt.setString(3, messboard.getMessage());
16.        this.pstmt.setDate(4, messboard.getReleaseTime());
17.
18.        if(this.pstmt.executeUpdate()>0) {
19.            flag = true;
20.        }
21.        this.pstmt.close();
22.        return flag;
23.    }
24.    @Override
25.    public List<Messboard> findAll() throws Exception {
26.        List<Messboard> all = new ArrayList<Messboard>();
27.        String sql = "SELECT * FROM Messboard";
28.        this.pstmt = this.conn.prepareStatement(sql);
29.
30.        ResultSet rs = this.pstmt.executeQuery();
31.        Messboard messboard = null;
32.        while (rs.next()) {
33.            messboard = new Messboard();
34.            messboard.setNO(rs.getInt(1));
35.            messboard.setID(rs.getString(2));
36.            messboard.setNickname(rs.getString(3));
37.            messboard.setMessage(rs.getString(4));
38.            messboard.setReleaseTime(rs.getDate(5));
39.            all.add(messboard);
40.        }
41.        return all;
}
```

```

42.     }
43. }

```

Proxy 包中对于 Messboard 类的代理类 MessboardDAOProxy.java 同 UsersDAOProxy.java 相类似, 继承了 IMessboardDAO 接口, 调用了 MessboardDAOImpl, 负责了数据库的关闭和打开。

最后的 Factory 包中, DAOFactory.java 包含两个函数, 分别与取得一个 DAO 的实例对象, 也是最终在 jsp 页面中去调用的函数。如代码 4-4 所示。

代码 4-4 DAOFactory.java

```

1. public class DAOFactory {
2.     public static IInfoDAO getIInfoDAOInstance() throws Exception{
3.         return new InfoDAOProxy();
4.     }
5.     public static IUsersDAO getIUsersDAOInstance() throws Exception {
6.         return new UsersDAOProxy();
7.     }
8.     public static IMessboardDAO getIMessboardDAOInstance() throws
Exception{
9.         return new MessboardDAOProxy();
10.    }
11. }

```

4.6.2.1 主界面展示

messBoard.jsp 中用于取得所有信息并罗列的代码如

代码 4-5 所示, 通过 DAOFactory 获得一个实例化的对象之后, 通过调用对象的 FindAll()方法完成所有信息的查找, 最后使用迭代器循环输出 table 格式的信息, 在表单中展现。最后的显示效果如图 4-15 所示。

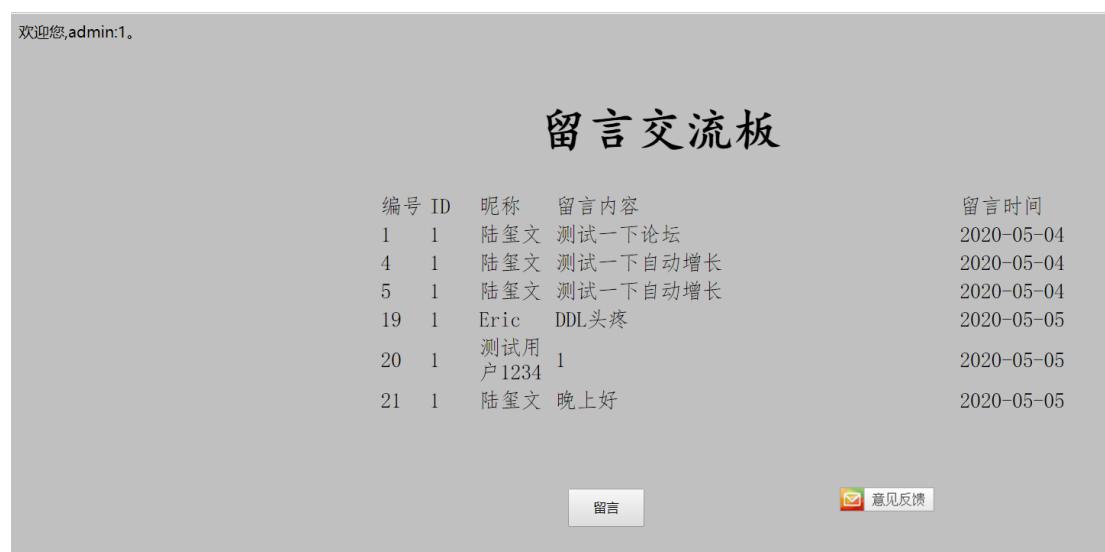


图 4-15 留言板显示图

代码 4-5 homeGuest.java

```
1. <table border="0" autocapitalize="on" style="font-  
    family:fangsong;font-size:22px;border-spacing: 3px">  
2. <tr>  
3. <td width="60">编号</td>  
4. <td width="60">ID</td>  
5. <td width="90">昵称</td>  
6. <td width="600">留言内容</td>  
7. <td width="200">留言时间</td>  
8. </tr>  
9.  
10. <%  
11. while(iter.hasNext()){  
12. Messboard mess2 = iter.next() ;  
13. %>  
14. <tr>  
15.  
16. <td><%=mess2.getNO()%></td>  
17. <td><%=mess2.getID()%></td>  
18. <td><%=mess2.getNickname()%></td>  
19. <td><%=mess2.getMessage()%></td>  
20. <td><%=mess2.getReleaseTime()%></td>  
21. </tr>  
22. <%  
23. }  
24. %>  
25. </table>
```

4.6.2.2增加留言

在 messBoard.jsp 页面（图 3-5）中，提供了一个留言的 button 以供用户增加相应的表单信息，在单击了该按钮之后，会导向 insertMess.html 界面（图 4-16），所有表单属性的名称基本与 Info 类的属性名相对应，更加便捷的完成信息添加功能。

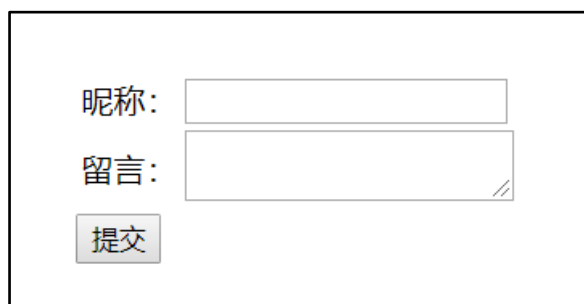


图 4-16 insertMess.html 界面

insertMess.html 界面最终提交向 messCreate，这一专门用于增加 Messboard 表单信息的 Servlet 中，其 doGet()方法如代码 4-6 所示。

代码 4-6 messCreate 中 doGet()

```
1.      protected void doGet(HttpServletRequest request,
2.      HttpServletResponse response) throws ServletException, IOException {
3.
4.          Messboard mess= new Messboard();
5.          mess.setNickname(new
6.      String(request.getParameter("nickname").getBytes("iso-8859-1"),"utf-
7.      8"));
8.          mess.setMessage(new
9.      String(request.getParameter("message").getBytes("iso-8859-1"),"utf-
10.     8"));
11.         mess.setReleaseTime(new java.sql.Date(new
12.     java.util.Date().getTime()));
13.         HttpSession session = request.getSession();
14.         Users user = (Users)session.getAttribute("user");
15.         System.out.println("当前session对象中user对象的ID值为:
16.     "+user.getID());
17.         if (user.getID()==null) {
18.             mess.setID("guest");
19.         }else {
20.             mess.setID(user.getID());
21.         }
22.         try {
23.             if(DAOFactory.getIMessboardDAOInstance().doCreate(mess))
24.                 System.out.println("添加新的留言成功");
25.         }
```

```
21.         response.sendRedirect("jsp/messBoard.jsp");
22.     } catch (Exception e) {
23.         e.printStackTrace();
24.         response.sendRedirect("html/insertMess.html");
25.     }
26. }
```

信息添加成功即自动跳转向 messBoard.jsp，如果添加失败，则会重新回到 insertMess.html 的表单界面，重新填写并提交表单信息。

4.6.3 意见反馈

邮我 API 的具体使用代码如代码 4-7 所示，通过点击超链图标，将会导向如图 4-17 所示的界面，比较符合功能需求，界面同时也满足了简洁的效果。经测试发送，可以正常收到邮件

代码 4-7 邮我代码

```
<a target="_blank" href="http://mail.qq.com/cgi-bin/qm_share?t=qm_mailme&email=t9vCz97A0tmGjo6098bGmdTY2q" style="text-decoration:none;margin-left: 200px;"></a>
```



The screenshot shows the 'Mail' QQ mailbox interface. At the top, there is a dark blue header with the 'Mail' logo, 'QQ 邮箱', and 'mail.qq.com'. To the right of the header, the user's email address '1159706522' and a link to '更换账号' (Change Account) are displayed. Below the header, the recipient's email address 'luxuwen1999@qq.com' is shown. The form has two main sections: '主题' (Subject) with a text input field containing 'test', and '正文' (Body) with a large text area containing '测试意见反馈'. At the bottom right of the form, there is a green button labeled '发送' (Send).

图 4-17 邮我实际效果界面

4.7 测试

在代码的编写过程中，以及编写完成后，均严格执行了功能测试。在测试过程中不断修改与完善“研招信息网”这一系统。

4.7.1 单元测试（类测试）

逐一编写各个模块类，并在缺省包下编写 test 静态类进行测试。在模块类的关键方法中嵌入控制台输出，进而即使出现异常错误也能及时指导问题的发生时序。以数据库打的各项测试为例，如代码 4-8 所示。

其中所有测试方法均为静态方法，便于在 main()函数中直接执行调用，以 java application 的方式运行。testJDBCconnect()为了测试 JDBC 连接是否正常，通常可以解决 Driver 驱动错误等问题；testMessboarddoCreate()测试添加留言板信息是否正常，是数据库增删改查操作测试中的一部分；testMessboardFindAll()功能测试留言板信息查找全部是否正常，用这一种方式可以根据 exception 的具体暴露位置去精准定位错误发生地。

代码 4-8 测试用类

```

1.      public static void main(String[] args) throws Exception{
2.          //      testJDBCconnect();
3.          //      testMessboarddoCreate();
4.          System.out.println(new java.sql.Date(new
           java.util.Date().getTime()));
5.      }
6.      static void testMessboardFindAll() {
7.          try {
8.              List<Messboard> all =
                DAOFactory.getIMessboardDAOInstance().findAll();
9.              Messboard mess = null;
10.             Iterator<Messboard> iter = all.iterator();
11.             while(iter.hasNext()) {
12.                 Messboard mess2 = iter.next() ;
13.                 System.out.println(mess2.getNO());
14.                 System.out.println(mess2.getID());
15.                 System.out.println(mess2.getNickname());
16.                 System.out.println(mess2.getMessage());
17.                 System.out.println(mess2.getReleaseTime());
18.             }
19.         }catch (Exception e) {
20.             System.out.println("出现了异常");
21.         }
22.     }
23.     static void testMessboarddoCreate() throws Exception {
24.         Messboard mess=new Messboard();
25.         mess.setID("1234");

```

```

26.         mess.setNickname("test");
27.         mess.setMessage("数据库加入测试");
28.         mess.setReleaseTime(java.sql.Date.valueOf("2020-5-4"));
29.         boolean flag =
DAOFactory.getIMessboardDAOInstance().doCreate(mess);
30.         System.out.println(flag);
31.     }
32.     static void testInfoFindAll() {
33.         try {
34.             List<Info> all =
DAOFactory.getIInfoDAOInstance().findAll();
35.             Info info = null;
36.             Iterator<Info> iter = all.iterator();
37.             while(iter.hasNext()) {
38.                 Info info2 = iter.next();
39.                 System.out.println(info2.getSchool());
40.                 System.out.println(info2.getSummerApplyFinishTime());
41.                 System.out.println(info2.getSummerActivityStart());
42.                 System.out.println(info2.getSummerActivityFinish());
43.                 System.out.println(info2.getRecommendationLetterNum());
44.                 System.out.println(info2.getPreachLink());
45.                 System.out.println(info2.getOthers());
46.             }
47.         } catch (Exception e) {
48.             System.out.println("出现了异常");
49.         }
50.     }
51.     static void testJDBCconnect() throws Exception {
52.         Connection conn = new DatabaseConnection().getConn();
53.         System.out.println(conn);
54.     }
55. }

```

4.7.2 集成测试（功能测试）

在各模块实现的基础上，结合系统功能需求测试整个系统，测试用例如表 4-1 所示。

表 4-1 单元测试用例

序号	模块	测试操作	期望结果	测试结果
1	夏令营信息	查询所有	正常显示	✓
2		增加信息	数据库加入正常	✓
3		跳转留言板	正常跳转	✓
4	留言板	查询信息	显示所有信息	✓
5		添加留言	正常添加	✓

6	系统流程	不登录访问主页	显示无权限，跳转登录	✓
7		登录 ID 错误	提示，重新登录	✓
8		登录 ID 成功	根据角色转入主页	✓
9		注册，验证错误	提示，重新注册	✓
10		注册，符合要求	注册成功，转入主页	✓
11	权限	管理员	进入管理员主页	✓
12		宾客	进入宾客主页	✓
13	意见反馈	点击邮我	正常跳转链接	✓
14		发送	内置邮箱收到反馈信件	✓
18		等待	刷新点阵	✓

4.8 部署发布

为了进一步发挥系统的实际使用价值，在腾讯云服务器上部署整个系统。地址为：<http://123.207.227.230:8080/MasterAdmissionNet/index.jsp>。

部署环境：腾讯云 CentOS 7。

项目名称：MasterAdmission（研招信息网）。

4.8.1 安装 JDK

4.8.1.1 安装

```
yum -y install java-1.8.0-openjdk java-1.8.0-openjdk-devel
```

4.8.1.2 配置环境变量

```
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64
export CLASSPATH=.:$JAVA_HOME/jre/lib/rt.jar:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
export PATH=$PATH:$JAVA_HOME/bin
```

当输入 `java -version` 成功出现版本信息后即可。

4.8.2 安装 Tomcat

4.8.2.1 下载 Tomcat

镜像地址会改变，Tomcat 版本也会不断升级。如果下载链接失效，请您到 [Tomcat 官网](https://tomcat.apache.org/download-80.cgi) 选择合适的安装包地址。

```
wget http://mirrors.tuna.tsinghua.edu.cn/apache/tomcat/tomcat-8/v8.5.39/bin/apache-tomcat-8.5.39.tar.gz
tar -xzf apache-tomcat-8.5.39.tar.gz
mv apache-tomcat-8.5.39 /usr/local/tomcat/
```

4.8.2.2添加用户

```
useradd www
mkdir -p /data/wwwroot/default
echo Hello Tomcat! > /data/wwwroot/default/index.jsp
chown -R www.www /data/wwwroot
```

4.8.2.3设置 JVM 内存参数

```
vi /usr/local/tomcat/bin/setenv.sh
JAVA_OPTS='-Djava.security.egd=file:/dev/./urandom -server -Xms256m -Xmx496m -Dfile.encoding=UTF-8'
```

4.8.2.4配置 server.xml

```
cd /usr/local/tomcat/conf/
mv server.xml server_default.xml
vi server.xml
```

添加如下内容：

```
<?xml version="1.0" encoding="UTF-8"?>
<Server port="8006" shutdown="SHUTDOWN">
  <Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener"/>
  <Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener"/>
  <Listener className="org.apache.catalina.core.ThreadLocalLeakPreventionListener"/>
  <Listener className="org.apache.catalina.core.AprLifecycleListener"/>
  <GlobalNamingResources>
    <Resource name="UserDatabase" auth="Container"
      type="org.apache.catalina.UserDatabase"
      description="User database that can be updated and saved"
      factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
      pathname="conf/tomcat-users.xml"/>
  </GlobalNamingResources>
  <Service name="Catalina">
    <Connector port="8080"
      protocol="HTTP/1.1"
```

```

connectionTimeout="20000"
redirectPort="8443"
maxThreads="1000"
minSpareThreads="20"
acceptCount="1000"
maxHttpHeaderSize="65536"
debug="0"
disableUploadTimeout="true"
useBodyEncodingForURI="true"
enableLookups="false"
URIEncoding="UTF-8"/>
<Engine name="Catalina" defaultHost="localhost">
<Realm className="org.apache.catalina.realm.LockOutRealm">
<Realm className="org.apache.catalina.realm.UserDatabaseRealm"
resourceName="UserDatabase"/>
</Realm>
<Host name="localhost" appBase="/data/wwwroot/default" unpackWARs="true" autoDeploy="true">
<Context path="" docBase="/data/wwwroot/default" debug="0" reloadable="false" crossContext="true"/>
<Valve className="org.apache.catalina.valves.AccessLogValve" directory="logs"
prefix="localhost_access_log." suffix=".txt" pattern="%h %l %u %t &quot;%r&quot; %s %b" />
</Host>
</Engine>
</Service>
</Server>

```

4.8.3 安装 MySQL

4.8.3.1 下载 MySQL

```

wget -i -c http://dev.mysql.com/get/mysql57-community-release-el7-10.noarch.rpm
yum -y install mysql57-community-release-el7-10.noarch.rpm

```

4.8.3.2 安装

```
yum -y install mysql-community-server
```

4.8.3.3 启动 MySQL

```
systemctl start mysqld.service
```

4.8.3.4 导入 Web 数据库

4.8.4 部署项目

4.8.4.1 导出 war 包

在 Eclipse 下，右击工程名，即可 export 出 war 包，如图 4-18 所示。

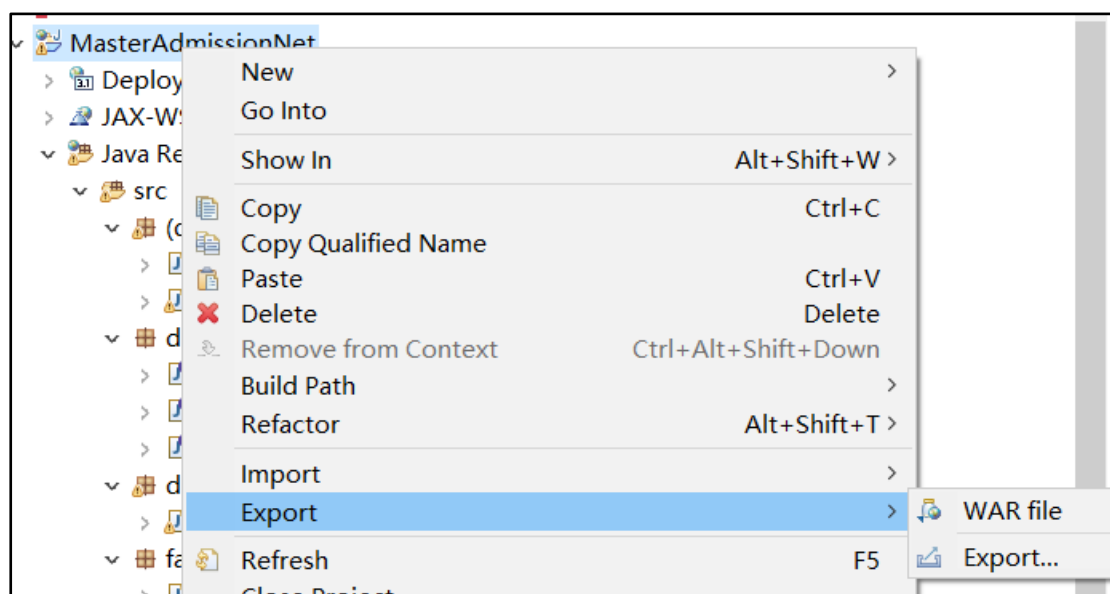


图 4-18 导出 WAR 包

4.8.4.2 部署启动

将 WAR 文件包导入 Tomcat 的缺省目录下后，当启动 Tomcat 会自动解压该 WAR 文件。启动 Tomcat 方式如下。

```
cd /usr/local/tomcat/bin  
./startup.sh
```

4.8.5 系统演示

整体系统演示视频如下：https://v.youku.com/v_show/id_XNDY2MTAzNjI0OA==.html。

4.9 JeeSite 4.1 框架学习

“研招信息网”的总体功能上相对简单，主体依托实验三，在数据库的增删改查上做了新的工作。为了更加进一步的美化系统，并充实系统，选择了目前较为新的 JeeSite4.1 框架进行部署学习，并尝试扩充整体项目。

JeeSite4.1 是一个 Java 企业信息化快速开发平台，基于经典技术组合，方便的在线代码生成功能。**开源！**

在这一次的实验中，也是重点学习了其代码在线生成模块，并加以了实践。

4.9.1 配置管理定义

首先是进入项目在线页的后台管理员界面，在左侧菜单栏中的研发栏中选择代码生成工具，在配置管理中新增所要定义的表说明，可以是单表，可以是一对多，也可以多对多。



图 4-19 表配置管理

4.9.2 数据表自动创建

为了进一步减少手工工作量，借助于 Eclipse 中的 ERMaster 插件进行自动生成创建数据表。ERMaster 主界面如图 4-20 所示。

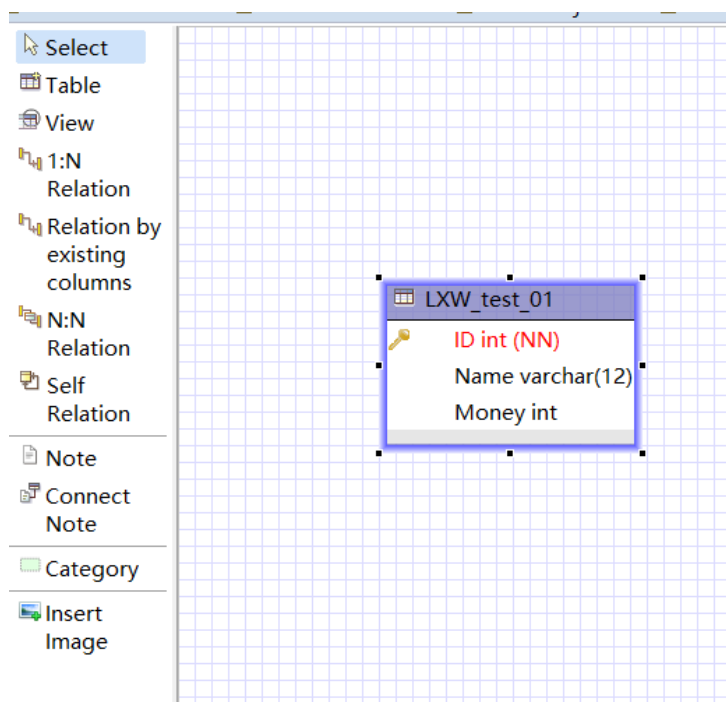


图 4-20 ERMaster 主界面

右击该表,可以快速编辑属性名称和类型,同时 ERMaster 支持内置字段组,在重复定义时可以快速添加进入,更进一步减少建表时间,如图 4-21 所示。

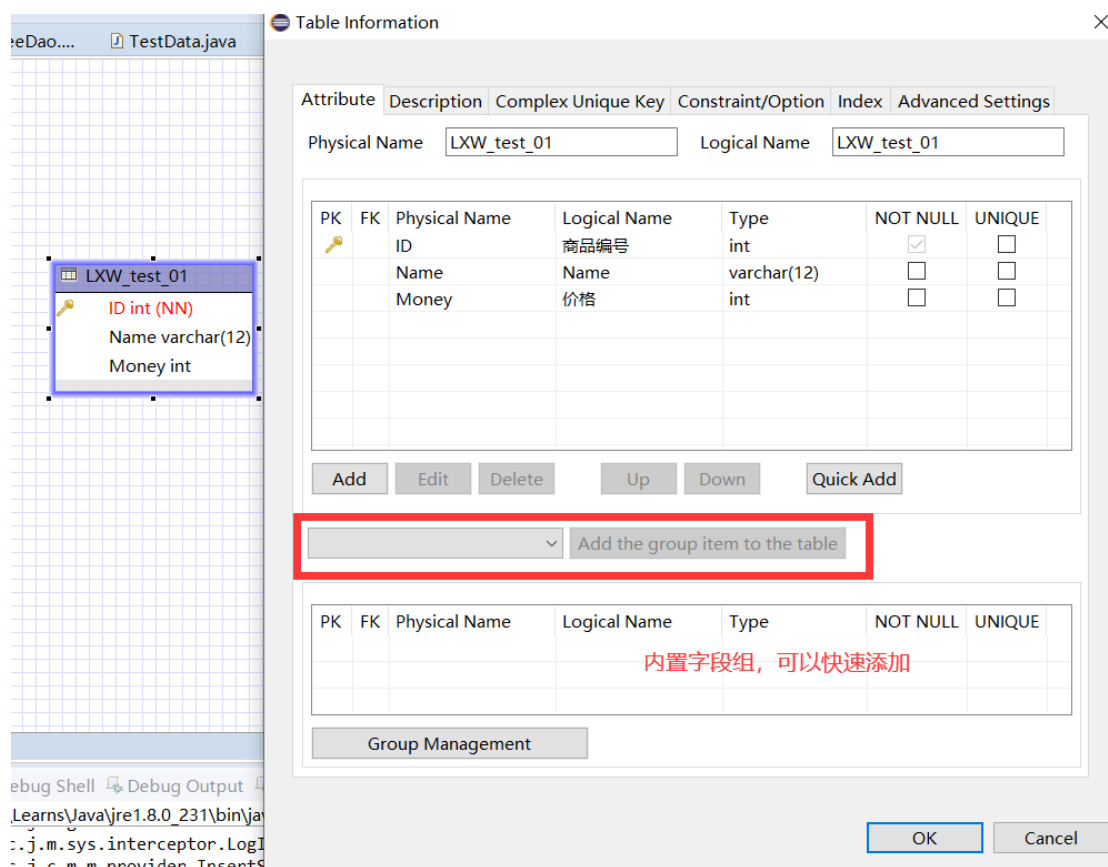


图 4-21 ERMaster 快速定义属性名称类型

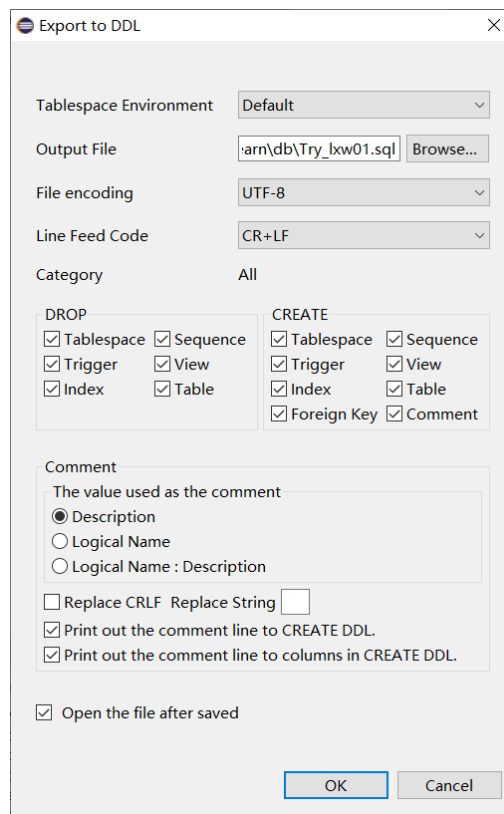


图 4-22 导出 DDL

Export to DDL 是 ERMaster 的又一大亮点，可以快速导出常用数据库的 DDL，不用自己编写 DDL 语句，简直神器！界面如图 4-22 所示。所导出的 DDL 语句如图 4-23 所示。

```
Connection profile
Type: [ ] Name: [ ]

1 SET SESSION FOREIGN_KEY_CHECKS=0;
2
3 /* Drop Tables */
4
5 DROP TABLE IF EXISTS LXW_test_01;
6
7
8
9
10 /* Create Tables */
11
12 CREATE TABLE LXW_test_01
13 (
14     ID int NOT NULL,
15     Name varchar(12),
16     Money int,
17     PRIMARY KEY (ID)
18 );
19 |
```

图 4-23 导出的 DDL 示例

4.9.3 实体类映射

这一步操作，为了对上一步中生成的数据表进行操作，通过内置模板，自动编写出对应的实体类，即我们常说的 **Bean** 类。此处我选择了在上文中新建的 **lxw_test_01** 表，如图 4-24 所示。

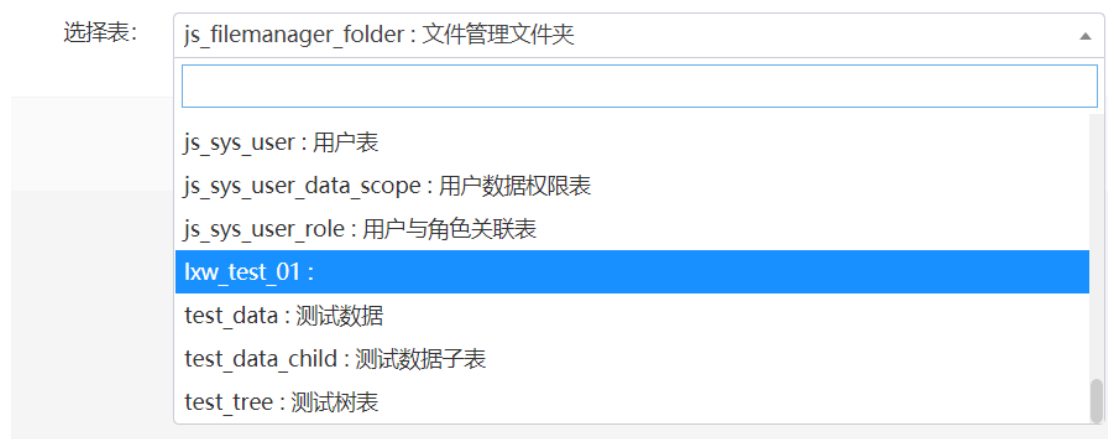


图 4-24 选择操作的数据表

在选定所需要的表之后,即可具体配置想要的映射关系了,如图 4-25 所示。



图 4-25 映射到实体类

接着即可具体配置增删改查操作中支持的类型了!这一步直接节省了大量的重复增删改查编写时间!如图 4-26 所示。

列名	列说明	字段类型	属性类型	属性名称	主键	插入	更新	列表	查询	匹配方式	编辑	必填	控件类型	栅格	新行	字典类型
1	id	int(11)	Integer	id	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	=	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	单行文本框	12/2/5...	<input checked="" type="checkbox"/>	
2	name	varchar(12)	String	name	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Like	<input checked="" type="checkbox"/>	<input type="checkbox"/>	单行文本框	12/2/5...	<input checked="" type="checkbox"/>	
3	money	int(11)	Integer	money	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	>=	<input checked="" type="checkbox"/>	<input type="checkbox"/>	单行文本框	12/2/5...	<input checked="" type="checkbox"/>	

图 4-26 定义支持的操作

最后配置生成路径。如图 4-27 所示。

* 生成模板: 单表/主子表 (增删改查)

* 生成包路径: com.jeesite.modules

* 生成模块名: lxw 存储路径

* 生成功能名: lxw_test_01

* 功能名 (简写): lxw第一次测试

生成基础路径: E:\JeeSites4-Programs\ManageSystem 最近路径快速选择

其它选项

是否有停用启用: ☐ 是 ☒ 否

是否有删除功能: ☒ 是 ☐ 否

是否可上传图片: ☐ 是 ☒ 否

是否可上传附件: ☐ 是 ☒ 否

☒ 保存并编译 ☒ 保存并生成代码 ☐ 关闭 ☐ 是否替换现有文件

图 4-27 定义实体类存储路径

4.9.4 创建菜单

在后台的数据表、映射实体类都配置完成之后，则可以自动生成前端的操作界面了，这一步，JeeSite 也支持自动生成！

在“创建菜单”这一研发工具中，可以细致的配置名称，目录，类型，图标等等！如图 4-28 所示。

仪表盘 代码生成工具 创建菜单

基本信息

上级菜单: 我的工作

菜单名称: 销售信息

链接(Href): /lxw/lxwTest01/list

排序(升序): 660

菜单图标:

页签标题:

菜单权重: 二级管理员

菜单类型: ☒ 菜单 ☐ 权限

归属模块: ☐ 文件管理 ☐ 内容管理 ☒ 核心模块

目标(Target):

权限标识: lxw:lxwTest01:view,lxw:lxwTest01:edit

字体颜色:

可见: ☒ 显示 ☐ 隐藏

其它信息

备注:

扩展字段

保存 关闭

图 4-28 创建菜单

最后的菜单创建效果如图 4-29 所示。



图 4-29 创建菜单效果

4.9.5 整体功能

在全部创建完成后,可以测试一下其增删改查的功能效果如何。这里添加了一条手机字段,如图 4-30 所示。

基本信息

* id: 1

商品名称: 手机

金额: 5000

✓ 保存 ✕ 关闭

图 4-30 添加字段“手机”

可以看到该字段添加成功，如图 4-31 所示。

lxw第一次测试管理



	商品名称	金额	操作
1	手机	5000	 

图 4-31 添加成功

4.9.6 代码自动生成原理

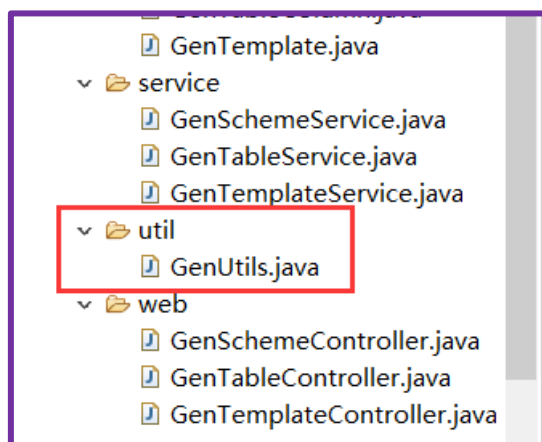


图 4-32 代码生成类 GenUtils.java

代码自动生成确实非常神奇而且给力！但并不神秘，当我仔细分析其源代码时，发现在 `util` 包下，给出了控制代码生成的类，`GenUtils.java`。其中定义了获取模板，XML 文件转换为对象等等工具，如图 4-33 所示。

```

/**
 * 获取模板路径
 * @return
 */
public static String getTemplatePath(){
    try{
        File file = new DefaultResourceLoader().getResource("").getFile();
        if(file != null){
            return file.getAbsolutePath() + File.separator + String.join(
                new String[]{"util."+GenUtils.class.getSimpleName()
            }
        }
    }catch(Exception e){
        logger.error("{} ", e);
    }

    return "";
}

/**
 * XML文件转换为对象
 * @param fileName
 * @param clazz
 * @return
 */
@SuppressWarnings("unchecked")
public static <T> T fileToObject(String fileName, Class<?> clazz){
    try {

```

图 4-33 GenUtils.java 部分内容

所以，自动生成增删改查的原理即：1.获取对应模板；2.配置 XML；3.创建对象；4.获取文件内容，写入对应目录。

4.10 设计心得

第四次实验在完成过程中，比对过许多种方案，最终也是选择了从自身以及同学们的实际使用需求角度出发，设计了“研招信息网”这么一个有现实意义的系统。当然在实际开发过程中，仍旧主体围绕数据库的增删改查，做了扩充。前端的美工方面实在是太过于欠缺了！

有个遗憾在于，最初想直接依托现成框架开发一款大型网站，然而时间有限，所得框架不是太过庞大难于吸收，就是陈旧难以运用。JeeSite 是我在搜寻了众多资料后最最中意的一款 ERP 框架，尤其是其对于数据库增删改查功能的自动生成！我也花费了大量的精力去学习它，并最终成功创建了 lxw_test 表模块，增添了菜单信息成功。然而碍于时间和能力，其他模块诸如国际化、报表分析、论坛等还没能细致掌握，实在不好直接整个作为自己的实验成果！故在报告中同时展示了相对简陋的实验四，以及在 JeeSite 框架上的学习。

5、结课体会

5.1 整体演示

4 次实验的整体演示视频地址：https://v.youku.com/v_show/id_XNDY2MTAzNjI0OA==.html。

5.2 心得体会

《Web 应用开发技术》使我收获颇丰：一是对于上学期所学 java 课程有了非常全面的练习实操，从 web 角度；二是对于完整系统网站的设计实现能力有了很大增长；三是结合同期所学《软件工程》，在设计与分析系统，撰写报告陈述思路方面得到了很好的实践；四是查阅资料，尤其是阅读 Web 项目开发的能力也有了很大的提升。

受限于课程的时间，觉得仍然有一小部分的遗憾，最后的实验没有能够去完成一个真正有广泛应用价值的大型网站；没能够将 JeeSite 框架全面掌握，实现基于 JeeSite 框架搭建“研招信息网”；Web 中还有许许多多的技术没能去体验尝试；前端的美工还是太弱……

在整个课程周期中，我始终保持着贯注的状态，基础的理论十分扎实，正如老师所说，这些理论基础正是阅读最新技术的强有力武器。这一点，在我独自阅读钻研 JeeSite 框架中，有了很好的体现，对于其自动生成数据库操作类能够比较快的理解并成功实现运用。因而，有信心在之后的学习工作中，技术不断提高，能力不断进步。

总体上，十分难忘整个过程！