目录

1. 引言

1.1 实习目的和要求

(1) 实习目的：

专业生产实习是本科教学计划中非常重要的实践性教学环节，是专业培养目标和教学计划、课程设置的有机组成部分，是理论教学的完善和补充。通过生产实习，使学生通过实践了解和掌握计算机的应用的开发、设计、运行等环节。通过生产实习培养学生掌握从事计算机系统开发的技能，加深对已经学习过的专业理论知识的理解和认识，提高在生产实践中调查研究、观察问题分析问题以及解决问题的能力和方法。为学生后续学习打下基础。

(2) 实习要求：

邀请专家做专题报告，了解计算机专业各领域发展方向和前景，主要研究内容和应掌握的理论与技术。

参观东软集团，了解东软主要发展历程，企业发展方向。学习他们的创业、敬业精神。

初步掌握 MyEclipse、Tomcat、sql 等开发工具，巩固有关数据库基础理论，熟悉基于网络的信息系统的开发过程，开发一个管理信息系统。

要求学生运用 JSP 的相关软件，用 JSP 语言进行一次基于 MVC 的 Java Web 开发，实现友好的界面管理与人性化的操作。网站要具有健壮性以及易于更新和维护的功能。

1.2 实习内容

实习内容如下表所示：

表 1-1 实习内容

实习时间	实习地点	实习内容
		实习动员及集中培训
		学习 JSP 开发技术，设计微博系统，完成数据库设计，数据库配置与连接，实现注册、登陆、微博管理、好友管理、推荐等模块功能。
		完善细节、撰写实习报告

1.3 实习单位概况

东北大学计算机学院实验教学中心简介：东北大学计算机学院实验教学中心承担着计算机学院计算机类专业和电子信息类专业的实验教学任务。此外，还承担各类课程设计、生产实习和毕业实习等实践教学任务。2015 年东北大学计算机实验教学中心入选国家级实验教学示范中心。

实验教学中心设有面向计算机科学与技术、物联网工程、通信工程和电子信息工程 4 个专业的 10 个实验室。其中，有面向计算机硬件的计算机网络、计算机组成原理、接口技术和嵌入式的 2 个硬件实验室，面向计算机软件的 2 个实验室，面向通信工程专业的 3 个实验室，面向电子信息工程的 2 个实验室（微控制器/微处理器实验室），面向物联网专业的 1 个实验室。

实验教学示范中心现有实验用房约 2000㎡，仪器设备 2277 台套，其中台式计算机 250 余套。中心现有实验技术人员 17 人，其中，高级实验师 4 名，实验师 8 名；具有硕士学位的 5 人，学士学位的 13 人。实验课程设置紧密围绕本科生培养计划制定的课程体系，以培养学生复杂工程问题的解决能力和提高动手能力为主旨的实验课程项目体系建设。

2. 基于 JSP 的微博系统设计与实现

2.1 系统需求分析

2.1.1 开发背景

过去很多人都喜欢写文章写随笔以求实现相互间的沟通、展现自己的才华和让别人了解自己的想法观点。现在的网络已经成为人们生活中不可或缺的一个元素，所以自然而然诞生了微博这样一个新兴事物，它不仅仅能取代前面所说的功能，还能加入图片，而且使得作者更能无所拘束地、生动地写出自己想写的，旁人也能非常便捷地阅读并且加以评论，并且它还能作为展示个人个性的窗户。微博现在已经成为很多人生活中必不可少的一个部分，方便了人与人之间的沟通和交流。具有传统交流媒介以及博客不具备的优势。

2.1.2 系统实现目标概述

基于微博以上的特点，本系统要实现微博的主要基本功能有注册、登录、退出、微博用户发表微博（心情）、分页浏览微博和评论、加好友（关注）等。当然由于微博的网络流行特点以及个人个性的展示，还适当要求界面比较漂亮轻快，直观便捷，操作方式简单以及人性化。

2.1.3 用例分析

微博系统用例图如下图图 2.1.3-1 所示：

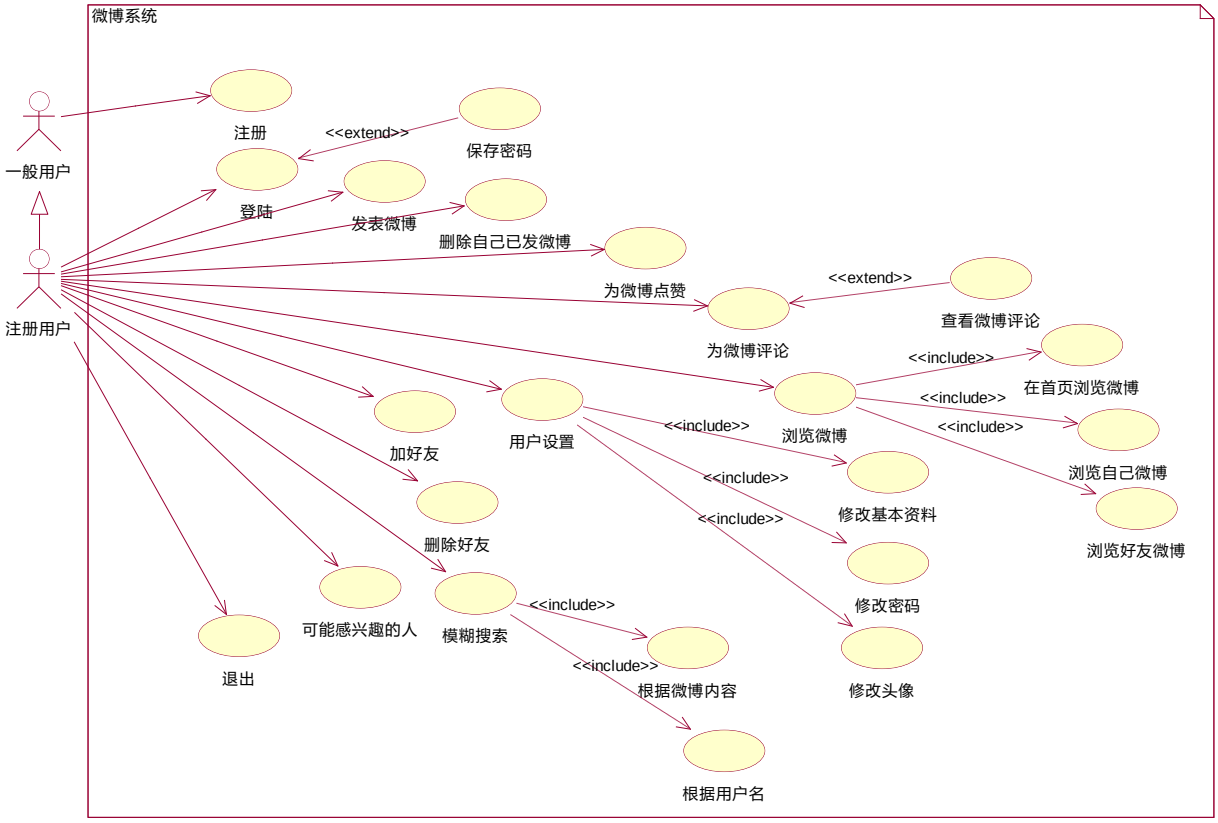

图 2.1.3-1 微博系统用例图

2.1.4 数据逻辑模型

微博系统的 E-R 图如下图图 2.1.4-1 所示：

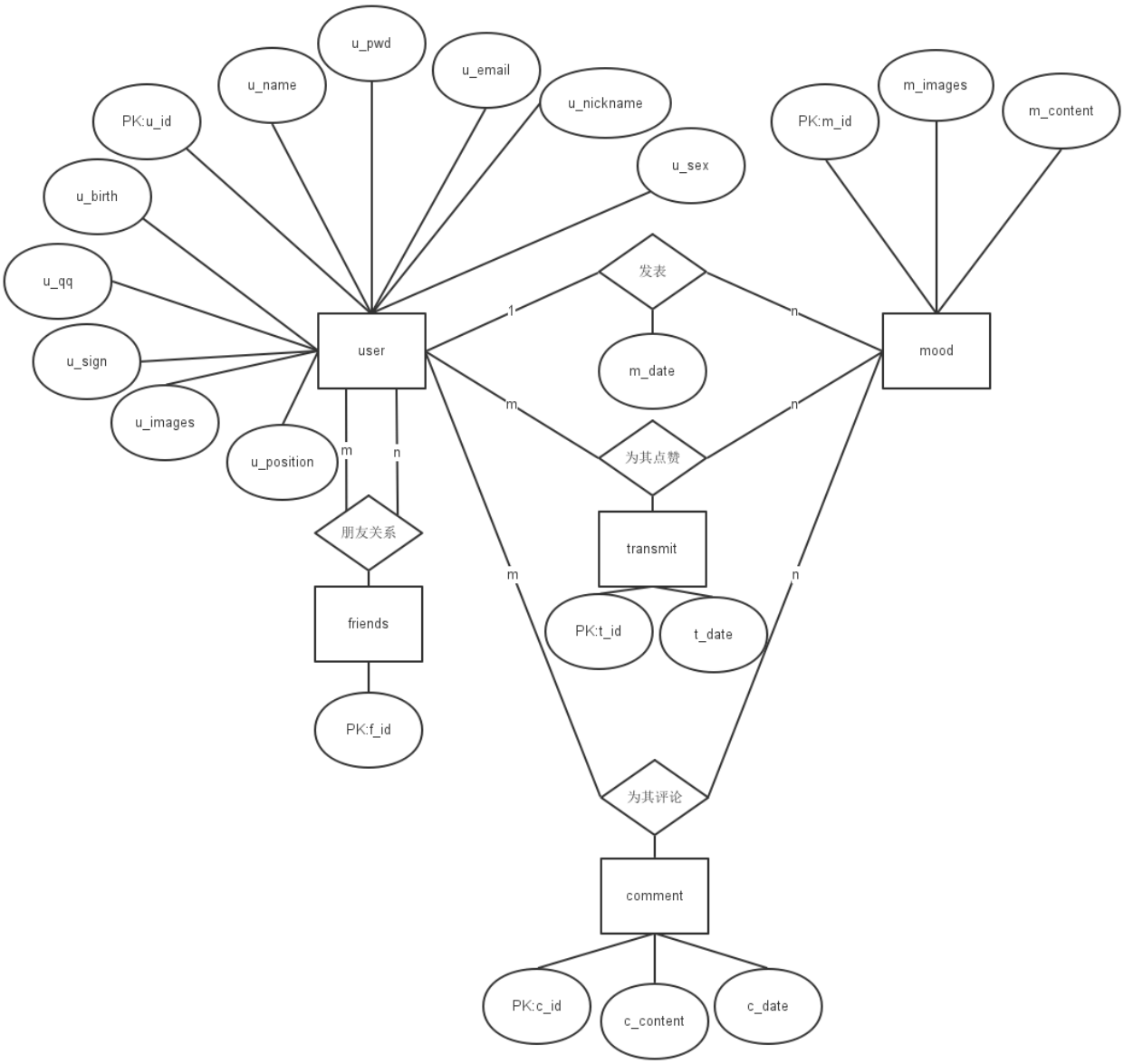

图 2.1.4-1 微博系统 E-R 图

如上图所示，微博系统主要的表为注册用户、微博。这两个是最基本信息。好友、评论、点赞信息是在网站运行中必要的信息。

2.2 系统总体设计

微博系统的功能主要可分成两部分，一个是前台，一个是后台，两个分别有不同的主页，其中功能有相关联的地方，有共同实现的功能。

前台功能模块主要包括登录注册退出模块。后台功能模块主要包括登录的注册用户进行微博操作、用户设置、好友操作、迷糊搜索以及推荐模块。

微博系统功能图如下图图 2.2-1 所示：

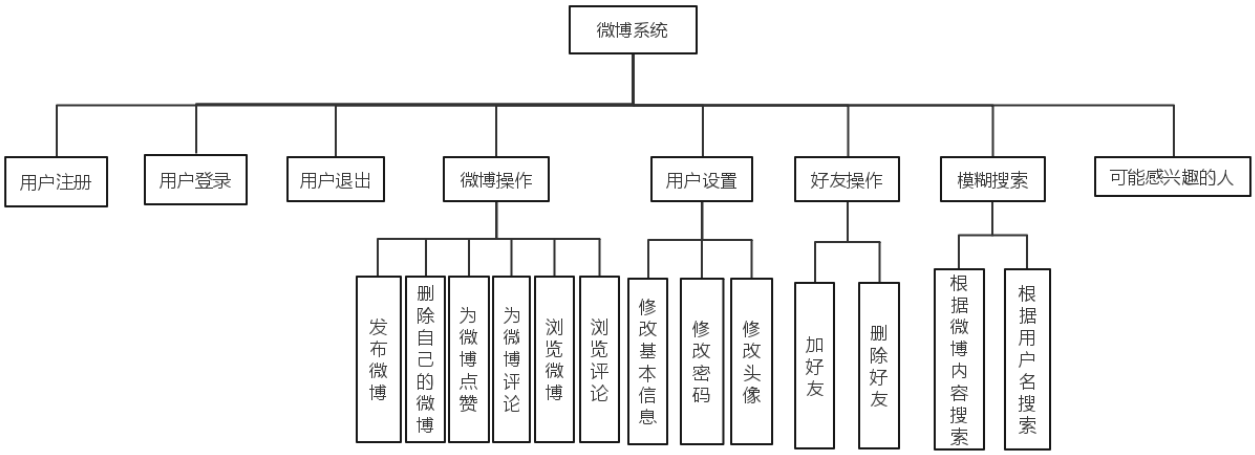

图 2.2-1 微博系统功能图

2.3 系统数据库设计

根据前面 E-R 图设计出实现微博系统功能所需要的数据库表的结构，系统数据库共 5 个表，下面进行一一详细介绍。

user 表为注册用户表：

表 2.3-1 user 表结构

字段名	数据类型	长度	主键	外键	可空	说明
u_id	int	11	是	\	not null	默认主键，每加一条记录自动加一
u_name	varchar	20	\	\	not null	用户名
u_pwd	varchar	20	\	\	not null	密码
u_email	varchar	20	\	\	not null	注册邮箱
u_nickname	varchar	30	\	\	not null	昵称
u_sex	varchar	8	\	\	not null	性别
u_position	varchar	50	\	\	not null	所在地
u_images	varchar	100	\	\	not null	上传后头像后的路径
u_sign	varchar	200	\	\	default null	签名
u_qq	int	10	\	\	default null	QQ 号
u_birth	datetime	\	\	\	default null	生日

mood 表为发布微博表：

表 2.3-2 mood 表结构

字段名	数据类型	长度	主键	外键	可空	说明
m_id	int	11	是	\	not null	默认主键，每加一条记录自动加一
u_id	int	11	\	是	default null	发本条微博的用户主键
m_date	varchar	30	\	\	not null	发本条微博时的时间
m_images	varchar	100	\	\	not null	本条微博的配图上传后的路径
m_content	varchar	2000	\	\	not null	本条微博的文字内容

comment 表为为微博评论表：

表 2.3-3 comment 表结构

字段名	数据类型	长度	主键	外键	可空	说明
c_id	int	11	是	\	not null	默认主键，每加一条记录自动加一
u_id	int	11	\	是	default null	发本条评论的用户主键
m_id	int	11	\	是	default null	本条评论所属微博的主键
c_content	varchar	45	\	\	not null	本条评论的内容
c_date	varchar	45	\	\	not null	发本条评论的时间

friends 表为朋友关系表：

表 2.3-4 friends 表结构

字段名	数据类型	长度	主键	外键	可空	说明
f_id	int	11	是	\	not null	默认主键，每加一条记录自动加一
u_id	int	11	\	是	default null	用户 id
f_user_id	int	11	\	是	default null	该用户加的好友的 id

transmit 表为为微博点赞表：

表 2.3-5 transmit 表结构

字段名	数据类型	长度	主键	外键	可空	说明
t_id	int	11	是	\	not null	默认主键，每加一条记录自动加一
u_id	int	11	\	是	default null	点赞的用户主键
m_id	int	11	\	是	default null	此赞所属微博的主键
t_date	varchar	45	\	\	not null	点赞的时间

2.4 系统详细设计

下面分模块来介绍本微博系统的详细设计。

2.4.1 用户注册模块

表 2.4.1-1 用户注册模块

输入	处理	输出
个人信息	查询 user 表判断用户名是否可用；可用则把记录添加到 user 表	注册成功跳转到 index.jsp 网站首页；否则跳转到 error.jsp

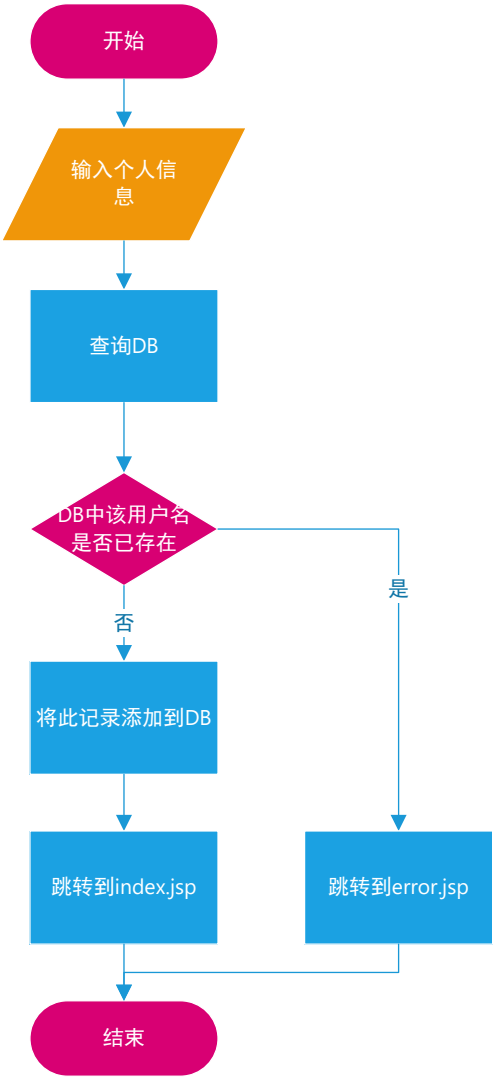

图 2.4.1-1 用户注册模块

2.4.2 用户设置模块

表 2.4.2-1 用户设置模块

编号	输入	处理	输出
1	用户 id 和想要修改的新基本信息	修改 user 表	输出更新是否成功
2	用户 id 和想要修改的新密码	修改 user 表	输出更新是否成功
3	用户 id 和想要修改的新头像	修改 user 表	输出更新是否成功

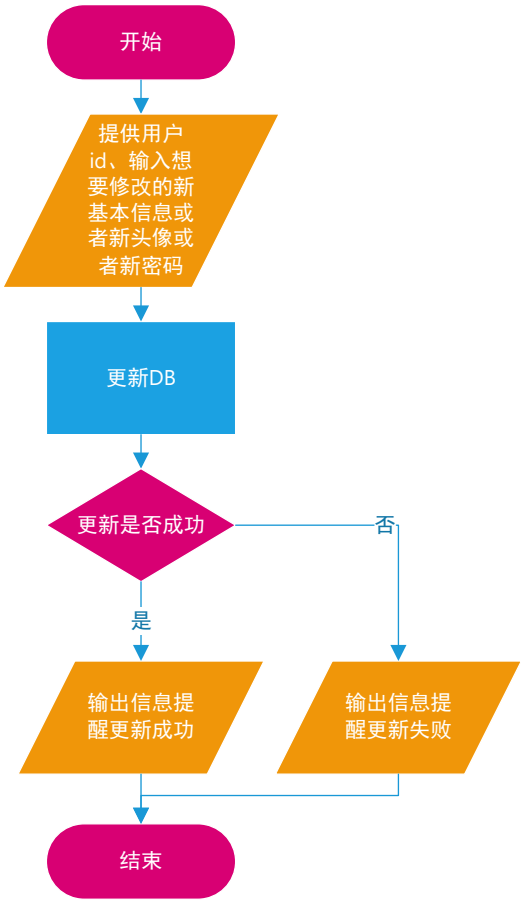

图 2.4.2-1 用户设置模块

2.4.3 用户登录模块

表 2.4.3-1 用户登录模块

输入	处理	输出
用户账号和密码	查询 user 表	验证成功跳转到用户主页 home.jsp；否则跳转到 error.jsp

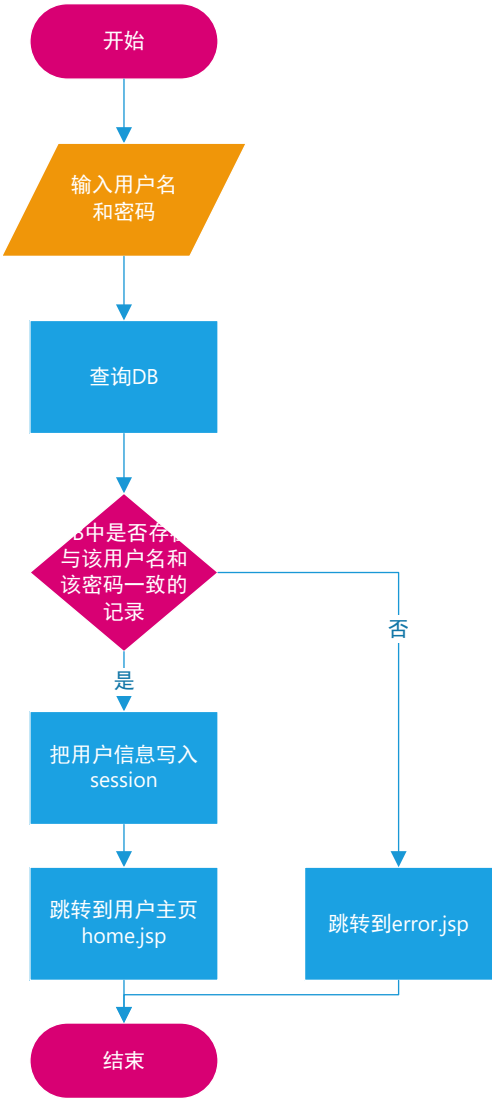

图 2.4.3-1 用户登录模块

2.4.4 用户退出模块

表 2.4.4-1 用户退出模块

输入	处理	输出
\	注销 session	跳转到 index.jsp 网站首页

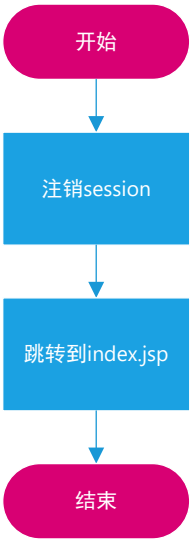

图 2.4.4-1 用户退出模块

2.4.5 微博相关操作模块

表 2.4.5-1 微博相关操作模块

编号	输入	处理	输出
1	用户 id 和微博文字内容（可选）以及微博配图（必选）	微博内容符合要求则添加到 mood 表	输出发表微博成功与否
2	要删除的微博 id	从 mood 表中把要删除的记录删除	跳转到“我的微博”profile.jsp，方便继续删除已发微博
3	点赞的用户 id 和被点赞的微博 id	将记录添加到 transmit 表	输出点赞是否成功
4	评论的用户 id 和被评论的微博 id	将记录添加到 transmit 表	输出评论是否成功

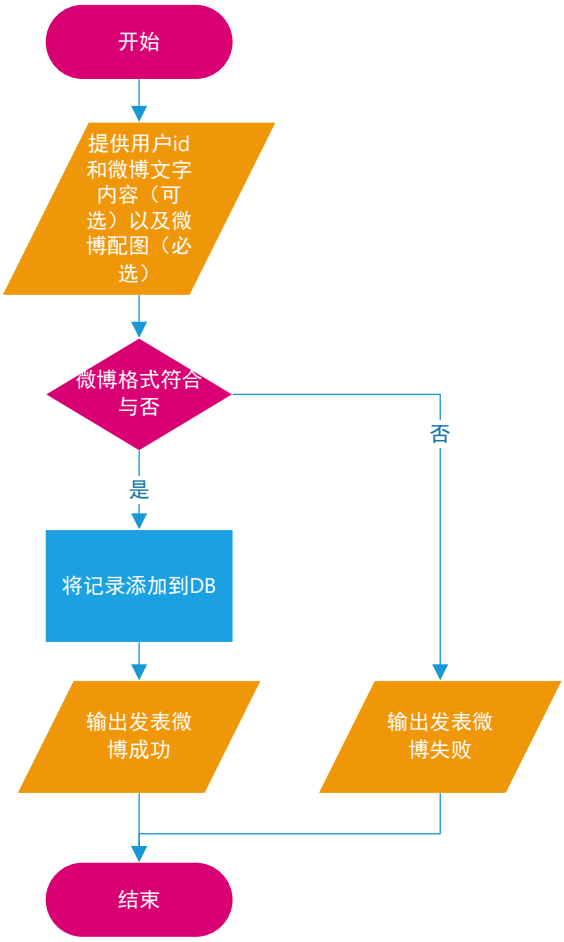

图 2.4.5-1 发布微博

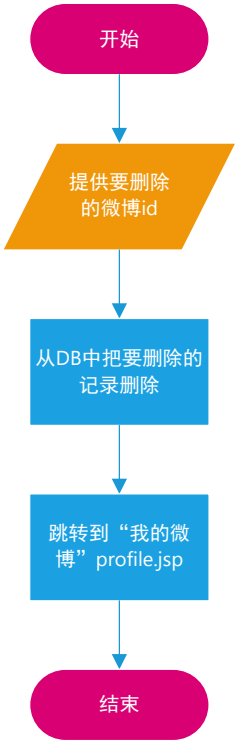

图 2.4.5-2 删除自己的微博

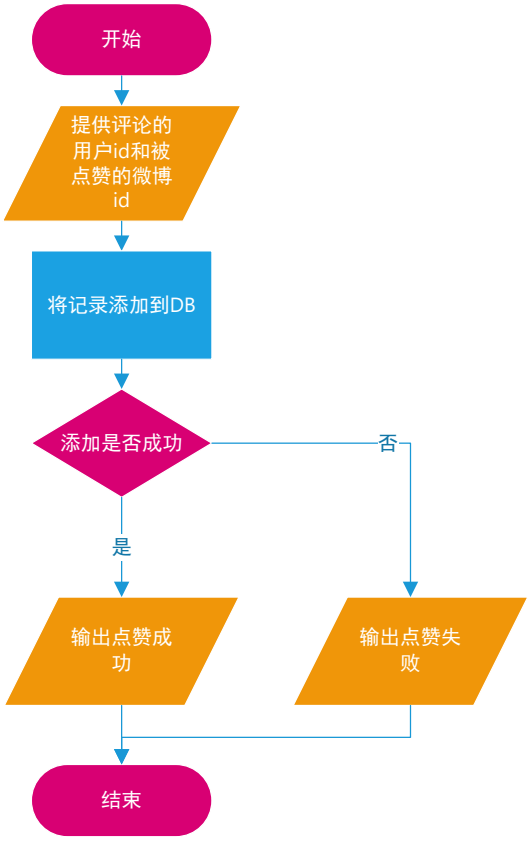
开始
提供评论的用户id和被点赞的微博id
将记录添加到DB
添加是否成功
否
是
输出点赞成功
输出点赞失败
结束

图 2.4.5-3 为微博点赞

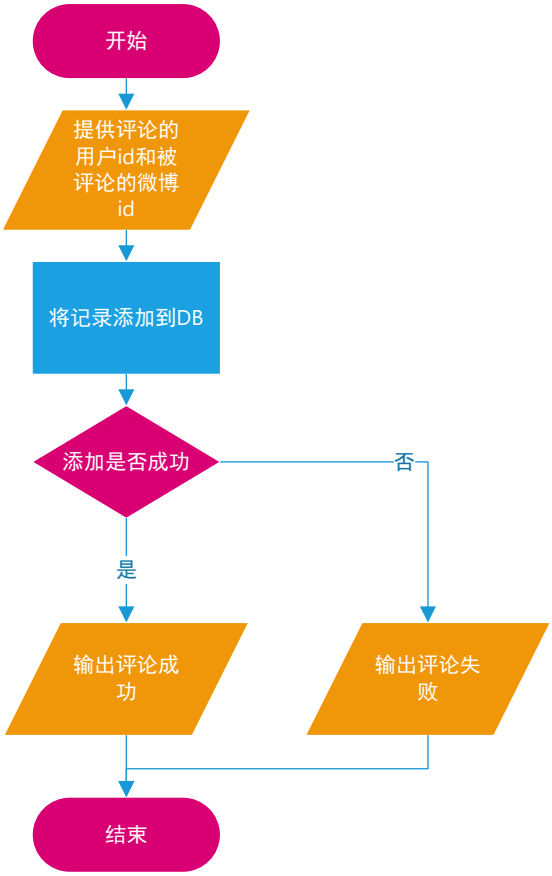

图 2.4.5-4 为微博评论

2.4.6 好友相关操作模块

表 2.4.6-1 好友相关操作模块

编号	输入	处理	输出
1	用户 id 和想要关注的好友的 id	将记录添加到 friends 表	输出添加好友是否成功
2	用户 id 和想要取消关注的好友的 id	从 friends 表中把要删除的记录删除	跳转到“我的朋友”friends.jsp，方便继续删除已关好友

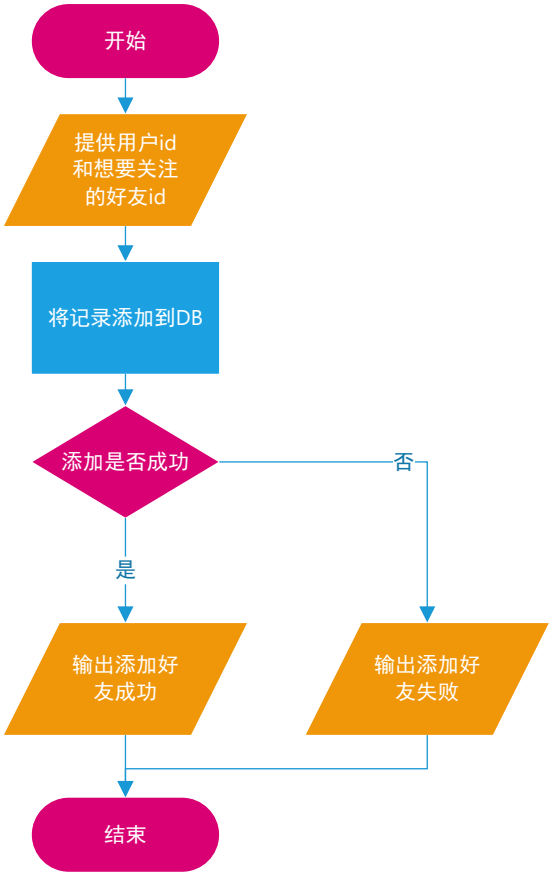

图 2.4.6-1 加关注

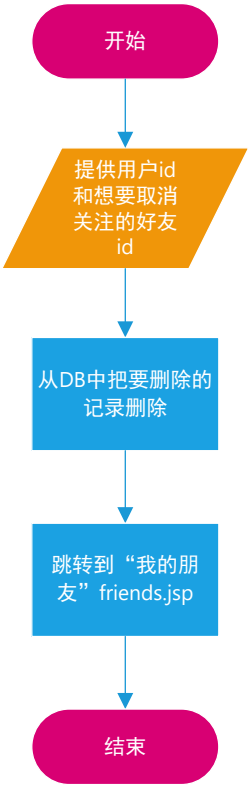

图 2.4.6-2 取消关注

2.4.7 模糊搜索模块

表 2.4.7-1 模糊搜索模块

编号	输入	处理	输出
1	微博文字内容关键词	查询 mood 表	跳转到显示查询显示页 showsearchbycontent.jsp
2	用户名关键词	查询 user 表	跳转到显示查询显示页 showsearchbyuser.jsp

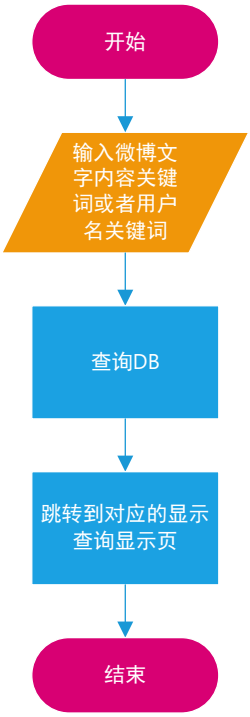

图 2.4.7-1 模糊搜索模块

2.4.8 可能感兴趣的人模块

表 2.4.8-1 可能感兴趣的人模块

输入	处理	输出
用户 id	查询 user 表	输出用户的好友的好友（除去用户已关注的好友和用户自己）并随机显示

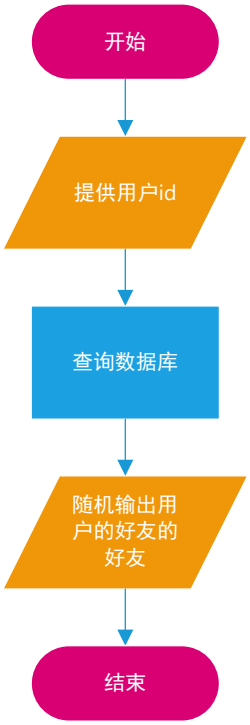

图 2.4.8-1 可能感兴趣的人模块

2.4.9 实体类设计

实体类有三个，实体类类图如下，每个用户可以写 0——n 篇微博，每个用户可以写 0——n 篇评论，每篇微博下面可以有 0——n 篇评论：

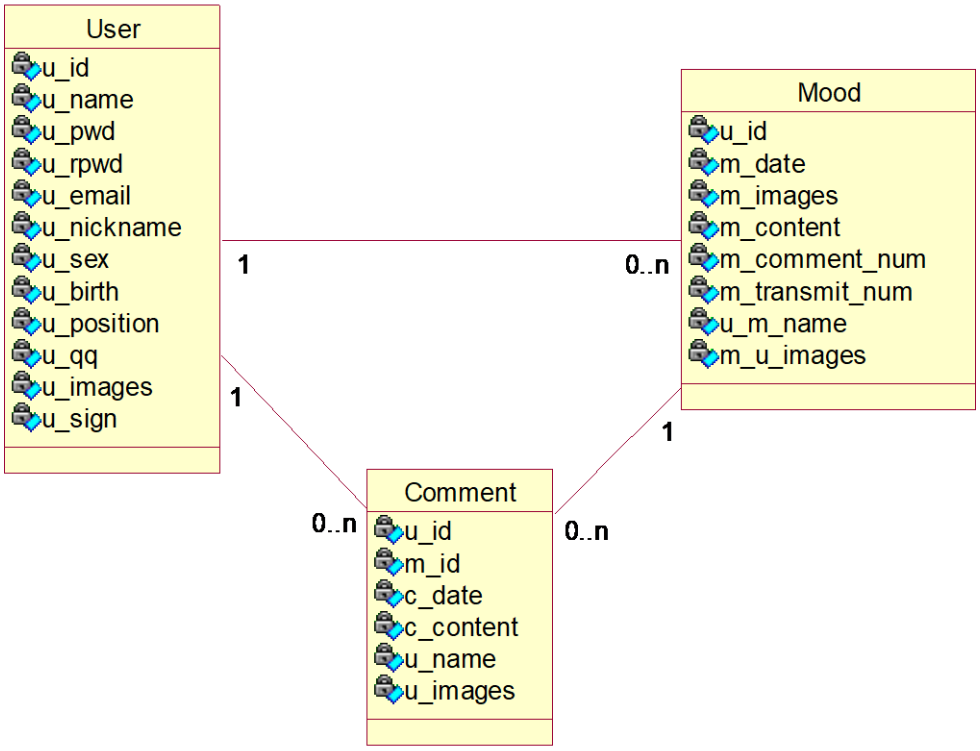

图 2.4.9-1 实体类类图

2.4.10 数据库模块

数据库模块核心代码：

```
package com.ice.dbutil;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
public class DBConn {
    //三属性、四方法

    //三大核心接口
    private Connection conn = null;
    private PreparedStatement pstmt = null;
    private ResultSet rs = null;

    //四个方法
    //method1: 创建数据库的连接
    private void getConntion(){
        try {
```

```
            //1: 加载连接驱动，Java反射原理
            Class.forName(Config.CLASS_NAME);
            //2:创建Connection接口对象，用于获取MySQL数据库的连接对象。三个参数：url连接字符串    账号  密码
            String url =
Config.DATABASE_URL+"://"+Config.SERVER_IP+":"+Config.SERVER_PORT+"/"
+Config.DATABASE_SID;
            conn =
DriverManager.getConnection(url,Config.USERNAME,Config.PASSWORD);
        } catch (ClassNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }


    //method2: 关闭数据库的方法
    public void closeConn(){
        if(rs!=null){
            try {
                rs.close();
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
        if(pstmt!=null){
            try {
                pstmt.close();
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
        if(conn!=null){
            try {
                conn.close();
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
```

```
        }
    }


    //method3: 专门用于发送 增删改 语句的方法
    //执行操作数据库的过程是：通过Connection连接对象获取Statement对象，再通过Statement对象执行操作
    public int execOther(final String strSQL,final Object[] params ){
        //1、调用方法1，获取数据库连接
        getConntion();
        //2、预先打印出即将执行的SQL语句，便于项目测试
        System.out.println("SQL:> "+strSQL);
        try {
            //3、创建Statement接口对象
            pstmt = conn.prepareStatement(strSQL);
            //4、动态为pstmt对象赋值
            for(int i=0; i< params.length ;i++){
                pstmt.setObject(i+1, params[i]);
            }
            //5、使用Statement对象发送SQL语句
            int affectedRows = pstmt.executeUpdate();
            //6、返回结果
            return affectedRows;
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            return -1;
        }
    }


    //method4: 专门用于发送查询语句
    public ResultSet execQuery(final String strSQL,final Object[] params){
        //1、获取数据库连接
        getConntion();
        //2、预先打印出即将执行的SQL语句，便于项目测试
        System.out.println("SQL:> "+strSQL);
        try {
            //3、创建PreparedStatement接口对象
            pstmt = conn.prepareStatement(strSQL);
            //4、动态为pstmt对象赋值
            for(int i=0; i< params.length ;i++){
                pstmt.setObject(i+1, params[i]);
```

```
            }
            //5、使用Statement对象发送SQL语句
            rs = pstmt.executeQuery();
            //6、返回结果
            return rs;
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
            return null;
        }
    }

}
```

2.4.11 页面设计

页面设计采取 html 语言的设计方法，将每一个模块分别写出，但是每一个模块都对应着自己的 jsp 页面从而实现动态。在做 jsp 网页时采用了 css 样式。

以下是用户主页 home. jsp 代码：

```
<%@ page language="java"
import="java.util.*,com.ice.dao.*,com.ice.po.*" pageEncoding="UTF-
8"%>
<%@ taglib uri="http://java.sun.com/jstl/core_rt" prefix="c"%>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<link rel="stylesheet" type="text/css" href="css/global.css" />
<link rel="stylesheet" type="text/css" href="css/home.css" />
<title>微博 - ${u_nickname}的主页</title>
<script type="text/javascript" src="script/home.js"></script>
</head>
<body>
<%
    User user=(User)session.getAttribute("user");
    //关注
    SearchFocusDao dao = new SearchFocusDao();
    int SFCount =dao.SFDao(user.getU_id());
    pageContext.setAttribute("SFCount",SFCount);
    //粉丝
    SearchFunNumberDao sfdao = new SearchFunNumberDao();
    int SFunCount =sfdao.FunDao(user.getU_id());
    pageContext.setAttribute("SFunCount",SFunCount);
```

```
    //微博
    FindMoodNumDao wbdao = new FindMoodNumDao();
    int SMNCount = wbdao.getMyNum(user.getU_id());
    pageContext.setAttribute("SMNCount",SMNCount);
    //分页
    FindMoodNumDao findOrder=new FindMoodNumDao();

    int pageNo=0;
    if(request.getParameter("No")==""){
        pageNo=1;//默认是第一页
    }
    else{
        pageNo=Integer.parseInt(request.getParameter("No"));
    }
    //要显示的微博
    ShowMoodDao showmooddao=new ShowMoodDao();
    List<Mood>
lstMod=showmooddao.showAllMood(user.getU_id(),pageNo,8);
    pageContext.setAttribute("lstMod",lstMod);
    //要推荐的好友，为好友的好友
    RandomShowFavUser rsfu=new RandomShowFavUser();
    List<User>lstuser=rsfu.ShowFavUser(user.getU_id());
    List<User>countLstUser = rsfu.RandomUser(lstuser,2);
    pageContext.setAttribute("countLstUser",countLstUser);
%>

<%
  if(request.getParameter("msg")!=null)
  {
   int res=Integer.parseInt(request.getParameter("msg"));
   switch(res)
   {
     case 5:
     out.print("<script>alert('资料更新成功！'); </script>");
     break;
     case 6:
     out.print("<script>alert('资料更新失败，请确保基本资料全部填写！');
</script>");
     break;
     case 1:
     out.print("<script>alert('赞成功！'); </script>");
     break;
     case 2:
     out.print("<script>alert('赞失败！'); </script>");
```

```
    break;

   }
  }
%>
<!-- header开始-->
<table id="header" align="center" border="0" cellspacing="0"
cellpadding="0">
  <tr>
       <td width="20%" align="center"><img src="images/logo.png"
width="178" height="62" /></td>
       <td width="55%" align="right">
         <table border="0" align="right" cellpadding="0"
cellspacing="0" id="daohang">
           <tr>
             <td width="20%"><a href="home.jsp?No=1">我的首页</a></td>
             <td width="20%"><a href="profile.jsp?No=1">我的微博
</a></td>
             <td width="20%"><a href="friend.jsp?No=1">我的关注
</a></td>
             <td width="20%"><a href="servlet/LogoutServlet">[ 退
出 ]</a></td>
           </tr>
         </table>
       </td>
    </tr>
</table>
<!-- header结束-->

<!-- container 开始-->
<table border="0" align="center" cellpadding="0" cellspacing="0"
id="container" >
    <tr>
       <td width="670" height="600" valign="top">
          <form action="servlet/PublishMoodServlet" method="post"
name="myform" enctype="multipart/form-data">
             <input type="hidden" id="update_id" name="u_id"
value="${u_id }" />
             <input type="hidden" value="<%=pageNo %>"
name="No"/><%-- 见“分页” --%>
             <table width="100%" border="0" cellpadding="0"
cellspacing="0" id="input">
               <tr>
                 <td width="160" height="48"> </td>
```

```
                    <td width="479" align="right"> <label
class="inputnum" id="chLeft">250</label></td>
                    <td width="31"> </td>
                  </tr>
                  <tr>
                    <td height="84"> </td>
                    <td><textarea id="inputbox" name="m_content"
cols="45" rows="5" onkeyup=checkLength(this)></textarea></td><%--
onkeyup属性在用户（在键盘上）释放按键时触发  --%>
                    <td> </td>
                  </tr>
                  <tr>
                    <td> </td>
                    <td valign="middle">请选择要上传的图片
  <input type="file"  name="m_images"
id="fileField"/>    <input type="image"
src="images/btn_input.png" onclick="document.myform.submit()"/></td>
                  </tr>
                </table>
            </form>
            <table width="100%" border="0" align="center"
cellpadding="0" cellspacing="0" id="menu">
              <tr>
                <td>模糊搜索：</td>
                <td>
                    <form id="form1" name="search" method="post"
action="servlet/SearchServlet">
                      <select name="subsql">
                              <option value="mb_content">微博内容
</option>
                              <option value="mb_user">用户名称</option>
                      </select>
                      <input name="textfield" type="text" class="input"
id="textfield" />
                      <input name="u_id" value="${u_id }"
type="hidden"/>
                      <input name="button" type="submit"
class="btnsearch" id="button" value="搜索" />
                    </form>
                </td>
              </tr>
            </table>
            <!-- weibo 开始-->
            <c:forEach items="${lstMod}" var="mood">
```

```
            <table id="weibo" width="90%" border="0" align="center"
cellpadding="3" cellspacing="0">
                <tr>
                    <td rowspan="3" align="center" valign="top"><img
src="${mood.m_u_images }" width="50" height="50" /></td>
                    <td width="88%"
class="content">${mood.u_m_name}<img src="icon/v.gif" width="11"
height="10" align="middle" />: ${mood.m_content}</td>
                </tr>
                <tr>
                    <td>
                        <c:if test="${mood.m_images!=null}">
                        <img src="${mood.m_images}"  width="120"
height="120"/>
                        </c:if>
                    </td>
                </tr>
                <tr>
                    <td height="25">
                        <table width="100%" border="0" cellpadding="0"
cellspacing="0" id="weibo_status">
                          <tr>
                            <td>${mood.m_date}</td>
                            <td align="right">
                            <c:if test="${mood.u_id==u_id}">
                            <a href =
"servlet/RemoveMoodServlet?m_id=${mood.m_id}">删除
</a>   |
                            </c:if>
                            <a
href="servlet/DealTransmit?u_id=${u_id}&m_id=${mood.m_id}">赞
</a>(${mood.m_transmit_num})</td>
                          </tr>
                        </table>
                    </td>
                </tr>
                <tr>
                    <td width="50"></td>
                    <td>
                        <form id="formcomment" name="formcomment"
method="post" action="servlet/DealCommentServlet">
                          <input name="textfield" type="text"
id="textfield" style="width:456px"/>
                          <input name="u_id" value="${u_id }"
```

```
type="hidden"/>
                        <input name="m_id" value="${mood.m_id }"
type="hidden"/>
                        <input name="button6" type="submit"
id="button6" value="评论" />
                    </form>
                    <form id="formshowcomment"
name="formshowcomment" method="post"
action="servlet/getCommentServlet?m_id=${mood.m_id }">

                        <input name="button7" type="submit"
id="button7" value="显示评论" />
                    </form>
                </td>

            </tr>
        </table>
        </c:forEach>

        <% int all=(findOrder.getFriendsNum(user.getU_id())-
1)/8+1; %>
        <table align="right">
            <tr>
                <td colspan="2" align="center">共<%=all %>页
  当前位于第<%=pageNo %>页
                <% if(pageNo>1){ %>
                    <a href="home.jsp?u_id=<%=
user.getU_id() %>&No=<%=pageNo-1 %>">上一页</a>
                <%}
                if(pageNo<all){%>
                    <a href="home.jsp?u_id=<%=
user.getU_id() %>&No=<%=pageNo+1 %>">下一页</a>
                <%}%>
                </td>
            </tr>
        </table>
        <!-- weibo 结束-->
    </td>
    <td width="280" align="center" valign="top" class="pageright">
        <!-- userinfo 开始-->
        <table align="center" id="userinfo">
            <tr>
                <td width="25%" rowspan="2"><img src="${u_images}"
width="50" height="50" /></td>
```

```
                <td width="75%">${u_nickname}</td>
            </tr>
            <tr>
                <td valign="top">${u_sign} </td>
            </tr>
            <tr>
                <td colspan="2" class="split1"><a
href="userinfo.jsp">用户设置</a></td>
            </tr>
            <tr>
                <td colspan="2" align="left">
                    <table width="80%" border="0" align="left"
cellpadding="3" cellspacing="0">
                        <tr>
                            <td align="center" class="split2">关注
<br>${SFCount}</td>
                            <td align="center" class="split2">粉丝
<br>${SFunCount}</td>
                            <td align="center" class="split2">微博
<br>${SMNCount}</td>
                        </tr>
                    </table>
                </td>
            </tr>
        </table>
        <table border="0" align="center" cellpadding="0"
cellspacing="0" id="userlist">
            <tr>
                <td class="title" height="29">可能感兴趣的人</td>
                <td align="right" class="title"><a
href="home.jsp?No=<%=pageNo %>">[换一换]</a></td>
            </tr>
            <c:forEach items="${countLstUser}" var="lstuser">
            <form
action="servlet/AddFriendsServlet?u_id=${u_id}&f_user_id=${lstuser.u_
id}" method="post" name="myform">
                <tr>
                    <td colspan="2">
                        <table border="0" cellpadding="0"
cellspacing="0" class="userdetail">
                            <tr>
                                <td width="26%"><img
src="${lstuser.u_images}" width="50" height="50" border="0" /></td>
                                <td width="74%">用户:
```

```
${lstuser.u_name}<input name="button3" type="submit"
class="btnguanzhu" id="button3" value="+关注"/>
                                    <br />昵称：${lstuser.u_nickname}
                                    <br />性别：${lstuser.u_sex}
                                    <br />所在地：${lstuser.u_position}
                                    <c:if
test="${lstuser.u_sign!=null}">
                                    <br />签名：${lstuser.u_sign}
                                    </c:if>
                                  </td>
                                </tr>
                              </table>
                            </td>
                          </tr>
                        </form>
                        </c:forEach>
                    </table>
                    <!-- userinfo 结束-->
                </td>
            </tr>
</table>
<!-- container 结束-->

<!--footer开始-->
<table id="footer" border="0" align="center" cellpadding="3"
cellspacing="0">
  <tr>
    <td width="447" align="center">Copyright &copy; 2016 刘皓冰. All
Rights Reserved</td>
  </tr>
</table>
<!--footer结束-->
</body>
</html>
```

以下是 home.css 代码：

```
/* CSS Document */
body {font:12px/1.75 Tahoma,Arial;background:#d3edfa
url(../images/bg.jpg) no-repeat center top;
background-attachment:fixed;
color:#333333;
line-height:180%;
word-wrap:break-word;
word-break:normal;
}
```

```
/*header*/
#header{
    width:950px;
    height:80px;
}
#daohang{
    width:350px;
    height:35px;
    background-image:url(../images/home_daohang_bg.png);
    background-repeat:no-repeat;
    text-align:center;
    font-size:14px;
}
#daohang a:link{
    text-decoration:none;
    color:#DFF;
}
#daohang a:visited{
    text-decoration:none;
    color:#DFF;
}
#daohang a:hover{
    text-decoration:none;
    color:#F90;
}
/* container */
.uploadbtn {width:200px;}
#priviewbtn {display:none;color:#0066cc;margin-
top:6px;float:left;margin-left:30px;_margin-left:15px;}
#uploading {display:none;color:#999;margin-top:5px;float:left;margin-
left:30px;_margin-left:15px;}
.clearline {display:block;height:0;line-height:0;font-
size:0;overflow:hidden;clear:both;}
#priviewpoic {padding:3px;margin-left:30px}

#container{
    width:950px;
    background-repeat:no-repeat;
}
#input{
    width:670px;
    height:175px;
    background-image:url(../images/input.jpg);
    background-repeat:no-repeat;
```

```
    color:#0066CC;
}
.inputnum{
    font-size:20px;
    color:#0066CC;
    font-weight:bold;
    padding:80px 90px 0px 0px;
}
#input textarea{
    width:470px;
    height:70px;
    font-size:13px;
    color:#333;
    margin-left:5px;
    border:none;
}
#input img{
    margin-right:10px;
    margin-left:10px;
    border:0px;
}

#menu{
    width:90%;
    height:35px;
    background-image:url(../images/menu_bg.gif);
    margin-top:10px;
    color:#09F;
}
#menu .input{
    width:200px;
    height:18px;
    border:solid 1px #CCC;
    color:#666;
}
#menu .btnsearch{
    width:60px;
    height:23px;
    border:solid 1px #CCC;
    color:#666;
}
#menu a{
    color:#09C;
    text-decoration:none;
```

```
}

#weibo{
    margin-top:10px;
    border-bottom:dotted 1px #CCC;
}
#weibo img{
    margin:5px 10px 0px 0px;
    padding:2px;
    border:solid 1px #CCC;
}
#weibo .content{
    line-height:25px;
    font-size:14px;
}
#weibo .content a{
    color:#09C;
    text-decoration:none;
}
#weibo .content img{
    border:none;
    vertical-align:middle;
    margin:0px;
    padding:0px;
}
#weibo_status{
    color:#09C;
}
/*page*/
#page{
    width:90%;
    height:60px;
    margin-bottom:20px;
}
#page .nextpage{
    border:solid 1px #999;
    background-color:#EFEFEF;
}

#userinfo{
    width:90%;
    height:180px;
    margin-top:10px;
    font-size:14px;
```

```
    color:#666;
    text-align:left;
}
#userinfo img{
    border:1px #CCC solid;
    padding:2px;
}
#userinfo a{
    color:#09C;
    text-decoration:none;
    font-size:14px;
}
#userinfo .split1{
    border-top:dotted 1px #CCC;
}
#userinfo .split2{
    border-right:solid 1px #CCC;
}
/*userlist*/
#userlist{
    width:90%;
    color:#666;
    margin-top:10px;
    text-align:left;
}

#usrelist a{
    color:#09C;
}
#userlist .title{
    background-image:url(../images/menu_bg.gif);
    padding-left:5px;
    font-size:14px;
    font-weight:bold;
}
#userlist .title a{
    color:#09C;
    text-decoration:none;
}
.userdetail{
    width:100%;
    border-bottom:1px dotted #999999;
}
.userdetail a{
```

```
    font-size:14px;
    color:#09C;
    text-decoration:none;
}
.userdetail img{
    border:solid 1px #CCC;
    padding:1px;
    margin:10px;
}
.btnguanzhu{
    border:1px solid #CCC;
    background-color:#EFEFEF;
    color:#F60;
    float:right;
}
/*footer*/
#footer{
    width:950px;
    color:#999;
    margin-top:2px;
    padding:30px 30px 30px 30px;
}
#footer a{
    color:#09C;
    text-decoration:none;
}
```

home.jsp 效果图如下：

图 2.4.9-1 home.jsp 效果图 1

图 2.4.9-2 home.jsp 效果图 2

profile.jsp 效果图如下：

图 2.4.9-3 profile.jsp 效果图

friends.jsp 效果图如下：

图 2.4.9-4 friends.jsp 效果图

index.jsp 效果图如下：

图 2.4.9-5 index.jsp 效果图

register.jsp 效果图如下：

图 2.4.9-6 register.jsp 效果图

2.5 系统实现

下面分模块来介绍本微博系统的实现。

2.5.1 用户注册模块

图 2.5.1-1 用户注册模块实现结果截图

部分关键代码如下：

```
        //获得要注册用户的各个属性
        String u_email = user.getU_email();
        String u_pwd = user.getU_pwd();
        String u_nickname = user.getU_nickname();
        String u_sex = user.getU_sex();
        String u_position = user.getU_position();
        String strSQL = "insert into user
values(null,?,?,?,?,?,?,'/MicroBlogbeta/face/default.jpg',null,null,n
ull)";
        //建立数据库连接
        DBConn dbConn = new DBConn();
        //执行数据库操作并保存结果集
        int affectedRows = dbConn.execOther(strSQL, new Object[]
{ u_email,u_pwd, u_email, u_nickname, u_sex, u_position});
        dbConn.closeConn();
        return affectedRows > 0 ? true : false;
```

2.5.2 用户设置模块

图 2.5.2-1 修改头像实现结果截图

更改基本资料部分关键代码如下，修改密码和修改头像与其相似，不再赘述：

```
        //得到更新的用户资料中的各个信息
        int u_id = user.getU_id();
        String u_nickname = user.getU_nickname();
        String u_sex = user.getU_sex();
        String u_birth = user.getU_birth();
        String u_position = user.getU_position();
        String u_qq = user.getU_qq();
        String u_sign = user.getU_sign();
       //执行数据库操作
        String strSQL = "update user set
u_nickname=?,u_position=?,u_sex=?,u_birth=?,u_qq=?,u_sign=? where
u_id = ?";
        DBConn dbconn = new DBConn();
        int affectRows = dbconn.execOther(strSQL, new Object[]
{ u_nickname,u_position, u_sex, u_birth, u_qq, u_sign,u_id });
        dbconn.closeConn();
        return affectRows > 0 ? true : false;
```

2.5.3 用户登录模块

图 2.5.3-1 用户登录实现结果截图

部分关键代码如下：

```
//登陆验证
public int checkUser(final String u_name, final String u_pwd) {
    int res=0;
    String strSQL = "select * from user where u_name = ? and u_pwd
= ?";
    //建立数据库连接
    DBConn dbConn = new DBConn();
    //保存结果集
    ResultSet rs = dbConn.execQuery(strSQL, new Object[] { u_name,
u_pwd });
    try {
        //最初rest的游标位于第一行之前，因此第一次调用next()，将把游标置于第
一行上，使它成为当前行。随着每次调用 next()，导致游标向下移动一行，按照从上至下的
次序获取 ResultSet行。
        //如果登陆成功，返回用户id
        while(rs.next()){      //这里必须循环遍历
            res=rs.getInt("u_id");//返回一条记录
        }
        return res;
    } catch (SQLException e) {
        e.printStackTrace();
        //登陆失败返回-1
        return -1;
    } finally {
        dbConn.closeConn();
    }
}
```

2.5.4 用户退出模块

图 2.5.4-1 用户退出实现结果截图

部分关键代码如下：

```
//注销Session
HttpSession session = request.getSession();
session.invalidate();//将session设置为失效,注销session 同时浏览器会立即创建一个新的session
response.sendRedirect("../index.jsp");
```

2.5.5 微博相关操作模块

图 2.5.5-1 发布微博实现结果截图

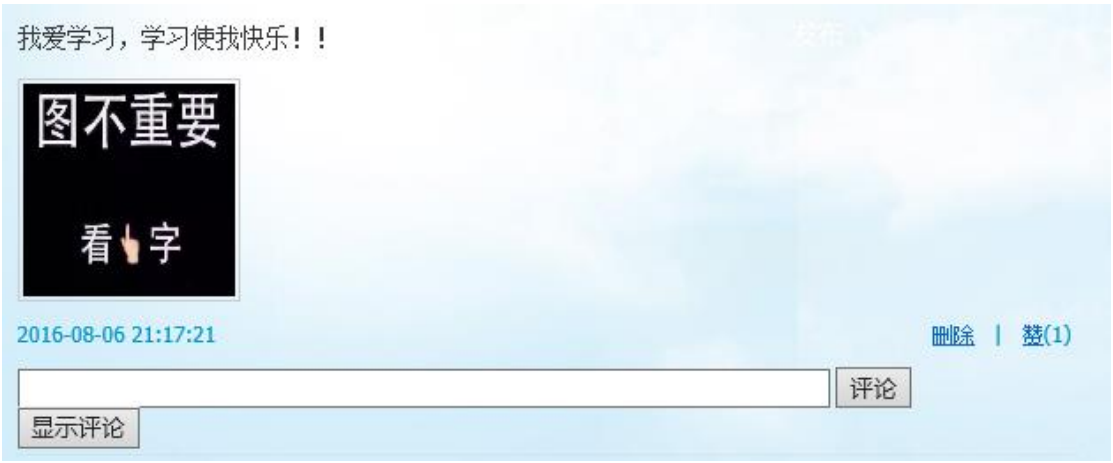

图 2.5.5-2 已发布的微博样例截图，可以进行删除、点赞或评论操作

图 2.5.5-3 好友发布的微博样例截图，可以进行点赞或评论操作

发布微博部分关键代码如下：

```
//得到用户发表的微博的各个属性
int u_id=mood.getU_id();
String m_date=mood.getM_date();
String m_images=mood.getM_images();
String m_content=mood.getM_content();
//建立数据库连接
DBConn dbconn=new DBConn();
String strSQL="insert into mood values ( null,?,?,?,?)";
//执行数据库操作
int affectedRows=dbconn.execOther(strSQL, new
Object[]{u_id,m_date,m_images,m_content});
boolean flag=(affectedRows>0)?true:false;
dbconn.closeConn();
return flag;
```

删除自己的微博部分关键代码如下：

```
public boolean removeDao(int m_id) {
    String strSQL = "delete from mood where m_id = ? ";
    //建立数据库连接
    DBConn dbConn = new DBConn();
    try {
        //执行数据库操作
        int affectedRows = dbConn.execOther(strSQL, new Object[]
{ m_id });
        boolean flag = (affectedRows > 0) ? true : false;
        return flag;
    } catch (Exception e) {
        e.printStackTrace();
```

```
            return false;
        } finally {
            dbConn.closeConn();
        }
    }
```

为微博点赞部分关键代码如下：

```
    public boolean dealTransmit(int m_id,int u_id) {
    //m_id和u_id是通过接收页面传递过来的，之后的t_date通过如下得到
    Date t_date_now=new Date();
        SimpleDateFormat format=new SimpleDateFormat("yyyy-MM-dd
HH:mm:ss",java.util.Locale.CHINA);
        String t_date=format.format(t_date_now);
    //建立数据库操作
    DBConn dbConn = new DBConn();
    String sql_Transmit = "insert into transmit values
(null,?,?,?)";
    int flag = dbConn.execOther(sql_Transmit, new
Object[]{m_id,u_id,t_date});
    dbConn.closeConn();
    return flag>0?true:false;
    }
```

为微博评论部分关键代码如下

```
        //获得一个日期对象并且赋值给c_date
    Date t_date_now=new Date();
        SimpleDateFormat format=new SimpleDateFormat("yyyy-MM-dd
HH:mm:ss",java.util.Locale.CHINA);//
        String c_date=format.format(t_date_now);
        int flag =0;
    //建立数据库操作
    DBConn dbConn = new DBConn();
    String sql_Comment = "insert into comment values
(null,?,?,?,?)";
        try {
        //执行数据库操作
       flag = dbConn.execOther(sql_Comment, new
Object[]{u_id,m_id,c_content,c_date});
        } catch (Exception e) {
            e.printStackTrace();
        }
    dbConn.closeConn();
    return flag>0?true:false;
```

```
    }
```

2.5.6 好友相关操作模块

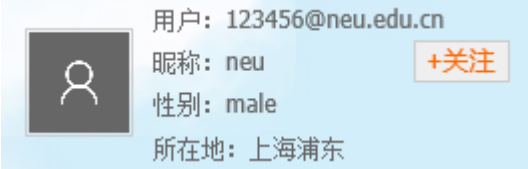

图 2.5.6-1 可以对其他注册用户进行关注

图 2.5.6-2 可以对好友进行取消关注

与微博的增、删操作关键代码相似，故不再赘述。

2.5.7 模糊搜索模块

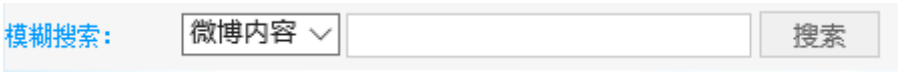

图 2.5.7-1 模糊搜索实现结果截图

模糊搜索微博文字内容部分关键代码如下，搜索好友与其相似，不再赘述：

```
public List<Mood> ShowByContentDao(String searc)
{
    //建立一个list
    List<Mood> lstMoodf=new ArrayList<Mood>();
    //模糊查询语句
    String strSQL="select * from mood where m_content like
'%"+searc+"%'";
    DBConn dbconn=new DBConn();
    //保存结果集
    ResultSet rs=dbconn.execQuery(strSQL, new Object[]{});
```

```
        try {
            //循环将结果添加到lstMoodf中
            while(rs.next())
            {
                Mood mood=new Mood();

                int m_id=rs.getInt("m_id");
                mood.setM_id(m_id);
                mood.setM_date(rs.getString("m_date"));
                mood.setM_images(rs.getString("m_images"));
                mood.setM_content(rs.getString("m_content"));
                mood.setM_comment_num(rs.getInt("m_comment_num"));
                int u_id=rs.getInt("u_id");
                mood.setU_id(u_id);
                String u_m_name=showMoodName(u_id);
                mood.setU_m_name(u_m_name);//获取每一条微博的用户名，相当于
两个表连接了，实现在下面
                FaceDao facedao=new FaceDao();
                String getFaceUrl=facedao.getFaceUrl(u_id);
                mood.setM_u_images(getFaceUrl);

                DealTransmitDao transmit=new DealTransmitDao();
                int m_transmit_num=transmit.countTransmit(m_id);
                mood.setM_transmit_num(m_transmit_num);
                lstMoodf.add(mood);
            }
            return lstMoodf;
        } catch (Exception e) {
            e.printStackTrace();
            return null;
        }
        finally{
            dbconn.closeConn();
        }
    }
```

2.5.8 可能感兴趣的人模块

图 2.5.8-1 可能感兴趣的人实现结果截图

部分关键代码如下：

```
List<User> lstuser=new ArrayList<User>();
//选择好友的好友用于随机显示，别忘了除去自己已关注的好友和自己~ 由于mysql不支持except，用left join实现
String strSQL="select * from (select * from user where u_id in (select f_user_id from friends where u_id in (select f_user_id from friends where u_id= ? ))) u1  left join  (select * from user where u_id in (select f_user_id from friends where u_id= ? ) union select * from user where u_id= ? ) u2 on u1.u_id=u2.u_id where u2.u_id is null";
//链接数据库
DBConn dbconn=new DBConn();
ResultSet rs=dbconn.execQuery(strSQL, new Object[]{u_id,u_id,u_id});
```

2.6 系统测试

本次实习所用的开发环境为

开发技术：JSP、CSS、JavaScript、MYSQL、JavaBean；

开发工具：Myeclipse 8.5、Photoshop、MYSQL；

服务器：Tomcat 6；

数据库：MYSQL；

2.6.1 测试项目

1、测试网站能够被哪个浏览器正常浏览；

2、测试数据库能否顺利连接；

3、测试网站各个功能能否正常进行。

2.6.2 测试结果

一 在 Microsoft Edge 浏览器上运行网站

网站的首页分布正常，功能都可以正常的运行。首页截图如下图所示：

图 2.6.2-1 Microsoft Edge 运行首页状况

二 在 IE11 浏览器上运行网站

在 IE11 上运行网站情况与 Microsoft Edge 相同：

图 2.6.2-2 IE11 运行状况

在其他浏览器上也均能正常地运行在此不再一一列举截图。

3. 实习总结

3.1 实习内容的复杂性评价

本次实习的工程复杂性评价如表 3.1 所示：

表 3.1 实习内容的工程复杂性评价

	复杂工程问题特征	问题描述及解决方案（方法）
1	必须运用深入的工程原理，经过分析才可能得到解决	理解 java web 的工作原理和 MVC 的设计模式。通过阅读书籍和上网查找资料，理解了 java web 原理，并运用了 MVC 模式解决实际项目中遇到的问题
2	涉及多方面的技术、工程和其它因素，并可能相互有一定冲突	综合运用了数据库技术、html+css 前端技术、java web 后端技术来完成项目
3	需要通过建立合适的抽象模型才能解决，在建模过程中需要体现出创造性	设计数据库时，建立了 E-R 实体联系模型
4	不是仅靠常用方法就可以完全解决的	提前检查账户是否可用、提前检查用户所填信息是否为空、提前检查两次密码输入是否一致
5	问题中涉及的因素可能没有完全包含在专业工程实践的标准和规范中	暂无
6	问题相关各方利益不完全一致	暂无
7	具有较高的综合性，包含多个相互关联的子问题	本项目的复杂性适中

3.2 实习体会、收获与建议

4. 参考文献

[1] 张斌，郭军. 软件工程及应用. 沈阳:东北大学出版社,2007.

[2] 王珊，萨师煊.数据库系统概论. 北京:高等教育出版社,2006.

[3] 潘国荣.Java&Jsp 应用程序实例开发. 北京:电子工业出版社,2014

[4] 百度学术