《数据库》课设作业

微信通讯录系统

学　号:　16231015

姓　名:　阮海航

专　业:　计算机科学与技术

学　院:　计算机与信息技术学院

目录

第一章 系统需求分析

随着市场经济的飞速发展和人们生活水平的不断提高，计算机科学技术逐渐成熟,其强大的功能已为人们深刻认识,并且在代替和延伸脑力劳动方面发挥越来越重要的作用。作为计算机应用的一部分,使用计算机对各项信息进行管理，具有着手工管理所无法比拟的优点。例如:检索迅速、查找方便、可靠性高、存储量大、保密性好等。这些优点能够极大地提高工作的效率,也是企业的科学化、正规化管理与世界接轨的重要条件。

本次课程设计要求建立一个对微信通讯录进行电子化管理的通讯录系统，用 mysql 数据库来实现其功能。在本通讯录系统中，可以将用户的账号、密码、微信号、备注等资料保存在数据库中，并可以随时进行登陆、退出、查看、添加、修改、删除、模糊查询、排序、个人设置等，在十足人性化的同时，提供一定的安全机制，是使该系统具有方便性、系统性、规划性、完备性和普遍性的性质。

1.1 涉及部门和人员分析

此微信通讯录系统包含了用户，可实现添加好友功能，添加了好友后，可对好友进行权限设置，不给对方看我的朋友圈，添加备注标签，拉入黑名单等等。管理员信息表包含管理员的账号密码，密码采用 md5 加密，投诉表包含被投诉人的账号，以及投诉原因，投诉信息表包含投诉人的账号，以及被投诉的次数。

1.2 涉及事件分析

（1）本人可查看自己的通讯录（好友）

（2）有添加好友、删除好友、查找已有好友的操作

（3）为每个好友设置、修改备注

（4）每个好友可以进行标签设置

（5）把好友添加到或移出黑名单

（6）好友朋友圈权限（不让对方看我的朋友圈、我不看对方朋友圈）

（7）投诉好友功能，投诉次数大于 5 的系统会请用户们注意

（8）管理员邀请码功能，必须有邀请码才可以注册

（9）所有密码采用 md5 加密

第二章 基本数据信息描述

2.1 数据元素

数据元素(data element)是计算机科学术语。它是数据的基本单位，在计算机程序中通常作为一个整体进行考虑和处理。

本系统的数据元素有：

用户的 id，微信号，密码，邮箱

用户的 id，微信号、是否为好友、备注、标签、黑名单

被投诉的名单，投诉的内容

被投诉的名单，投诉的次数

管理员 id，账号，邮箱，密码

邀请码，邀请人，被邀请人

2.2 数据组织

数据组织 data Organization :按照一定的方式和规则对数据进

行归并、存储、处理的过程。

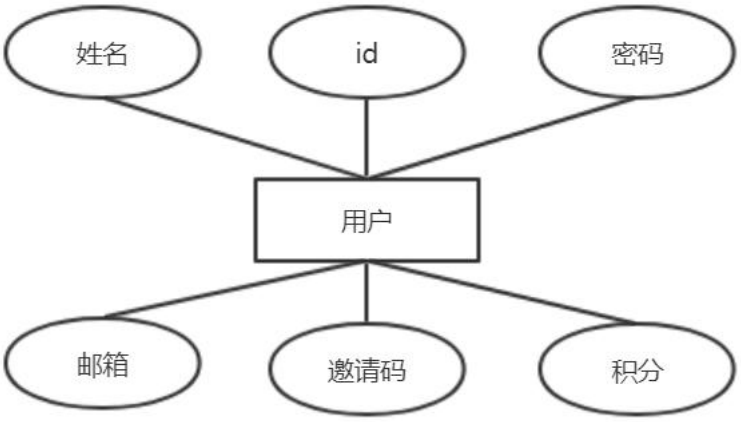
第三章 概要设计

3.1 ER 图

E-R 图也称实体-联系图(Entity Relationship Diagram)，提供了表示实体类型、属性和联系的方法，用来描述现实世界的概念模型。

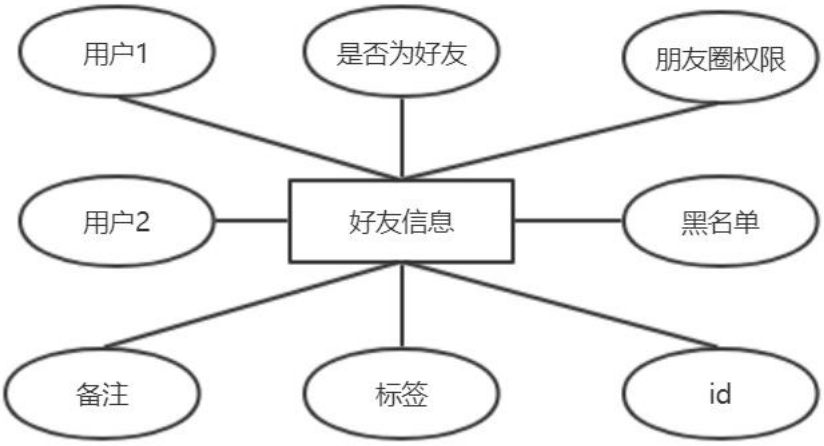
用户 ER 图

用户信息包括 id，姓名，密码，邮箱，邀请码，积分六部分



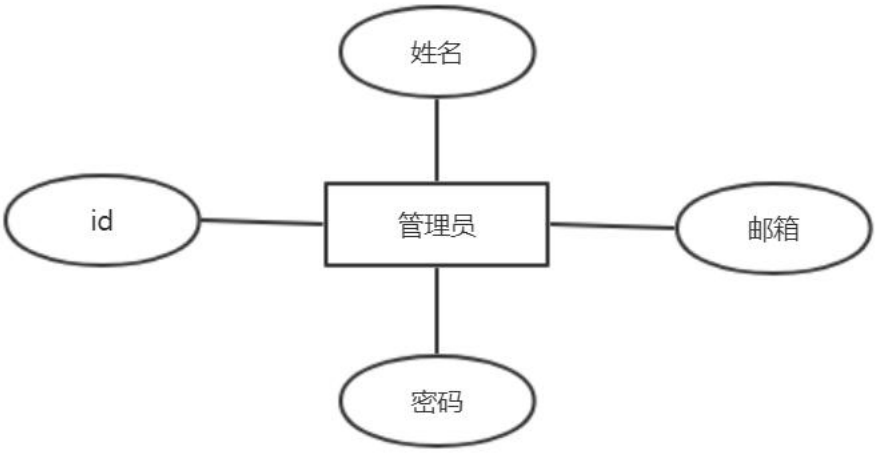
好友信息 ER 图

好友信息主要记录用户 1 对用户 2 的操作，有记录是否为好友，朋友圈权限，黑名单，备注，标签，id 等属性



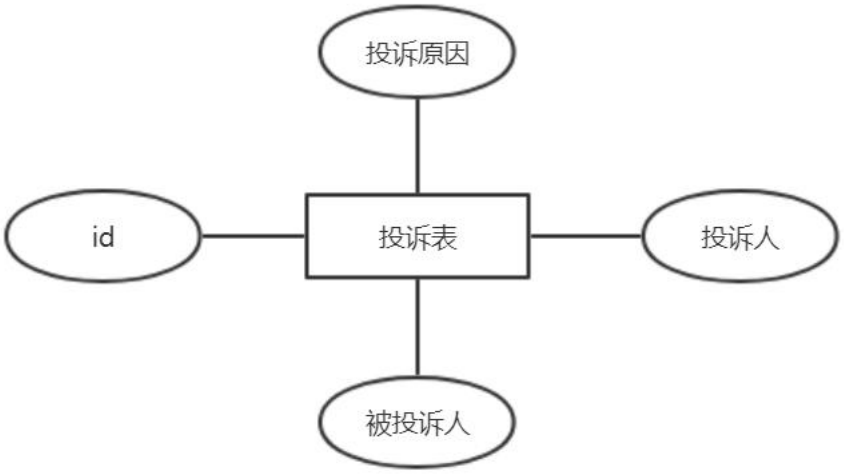
管理员 ER 图

管理员信息包括管理员 id，姓名，邮箱，密码等



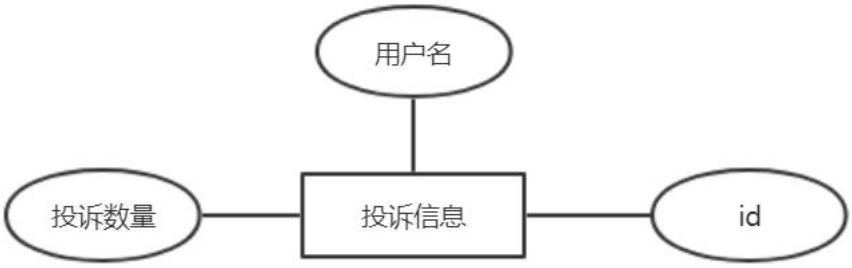
投诉表 ER 图

投诉表包括 id，投诉人，被投诉人，投诉原因



投诉信息记录 ER 图

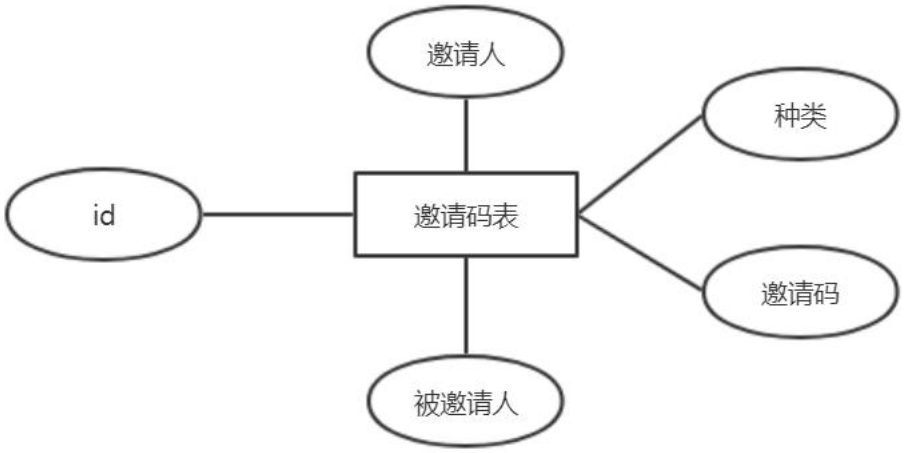
投诉信息记录包括 id，用户名，投诉数量的等



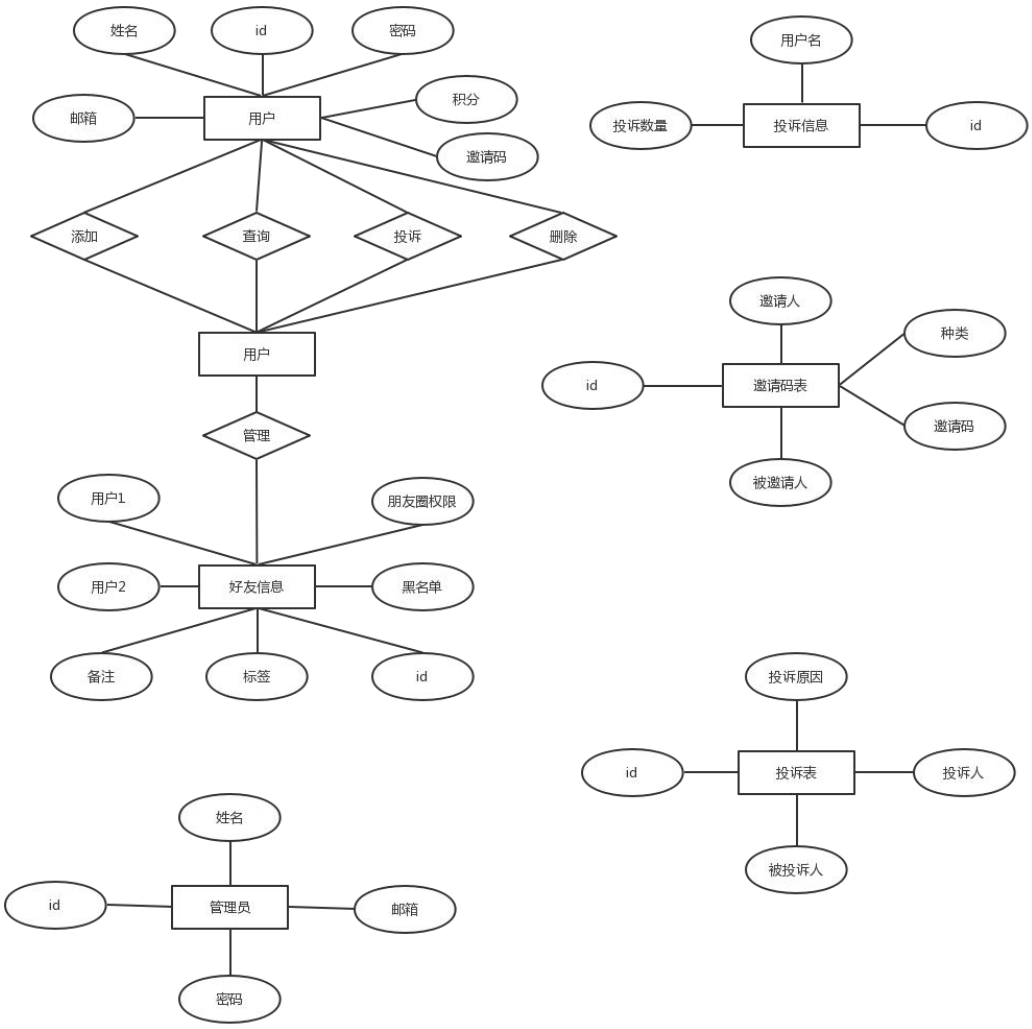
邀请码 ER 图

邀请码码包括 id，邀请码，ER 图，邀请人，被邀请人，种类，

邀请码



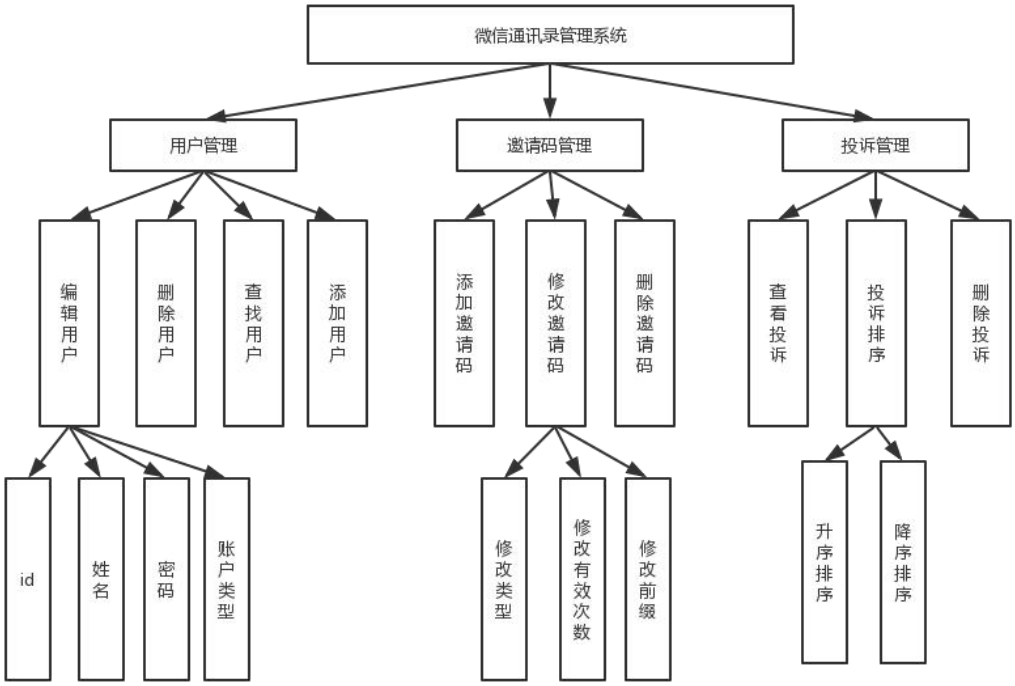
全局 ER 图



3.2 系统功能结构图

我们将系统分为俩部分，分为管理员系统和普通用户系统，为了使图示清晰我们为俩个部分分别绘制了系统功能结构图，分别如下所示。

管理员的功能结构图



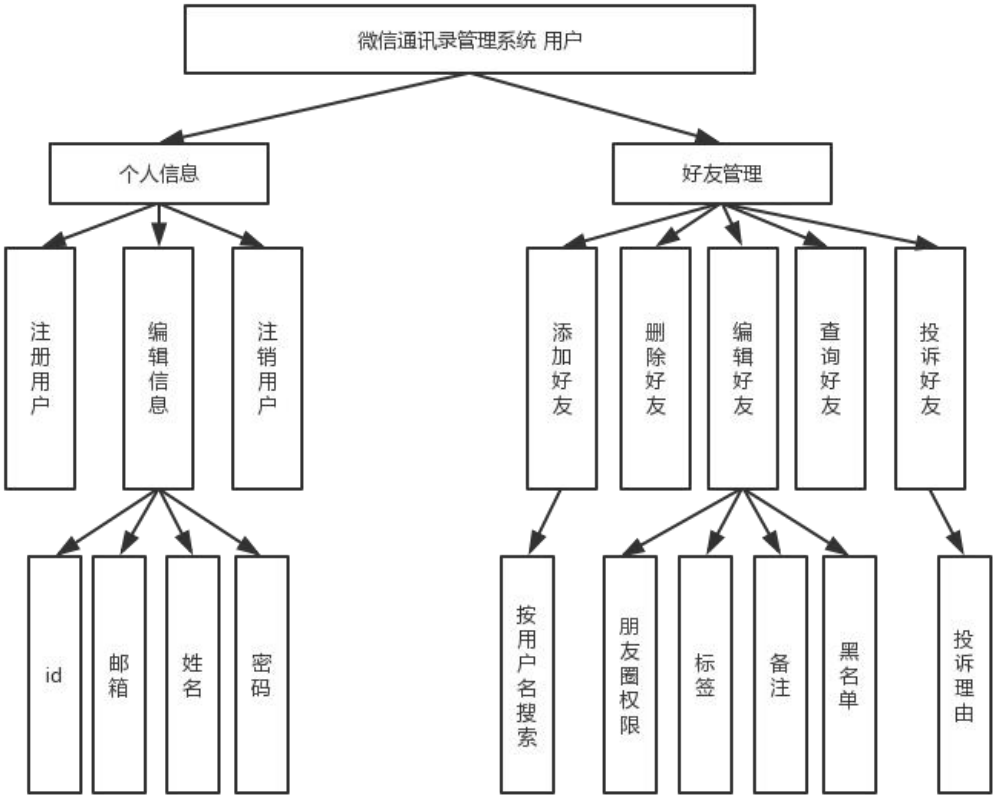
从上图可以看出，管理员的功能共有三类，用户管理，邀请码管理，投诉管理。

用户管理，管理员可以编辑用户，删除用户，查找用户，添加用户等，编辑用户可以用来编辑 id，姓名，密码，账户类型等操作。

邀请码管理，管理员可以添加，修改，删除邀请码，对邀请码可以做修改类型，修改有效次数，修改前缀的操作。

投诉管理，管理员可以查看投诉，对投诉信息进行排序，以及删除某个用户的投诉，对于投诉的数量，可以用来进行升序排序或者降序排序。

用户的功能结构图



在用户功能模块，分别俩部分，一个是个人信息的管理，一个是好友信息的管理。

个人信息包括，注册用户，编辑个人信息，注销用户三个功能，编辑信息可以对自己的 id，邮箱，姓名，密码进行修改。

好友管理包括添加好友，删除好友，编辑好友，查询好友，投诉好友五个功能，其中添加好友是按照用户名搜索，用户名在修改时是必须唯一的，编辑好友可修改好友的朋友圈权限，标签，备注，黑名单等，投诉好友时选择输入投诉进行投诉，投诉时，管理员可在系统进行投诉管理。

第四章 详细设计

4.1 数据库设计

4.1.1 数据库中表的设计

本数据库中共有 6 张表，分别如下

1 用户表

数据项名	数据项含义	类型	长度	完整性约束
id	用户 id	int	15	主键
user_name	用户名	varchar	128	
email	邮箱	varchar	128	
pass	密码	varchar	128	
invite_num	邀请码	varchar	128	

2 管理员表

数据项名	数据项含义	类型	长度	完整性约束
id	管理员 id	int	15	主键
admin_name	用户名	varchar	128	
email	邮箱	varchar	128	
pass	密码	varchar	128	

3 好友信息表

数据项名	数据项含义	类型	长度	完整性约束
id	好友信息 id	int	15	主键
node_name	用户 1 名	varchar	128	外键

node_server	备注	varchar	128	
node_type	是否为好友	bool	10	
node_method	用户 2 名	varchar	128	
node_info	朋友圈权限	bool	10	
node_status	标签	varchar	128	
node_order	黑名单	bool	10	

4 投诉表

数据项名	数据项含义	类型	长度	完整性约束
id	被投诉人 id	int	15	主键
node_name	用户名	varchar	128	
node_complaint	投诉理由	varchar	128	

5 投诉统计表

数据项名	数据项含义	类型	长度	完整性约束
user_name	用户名	varchar	128	主键，外键
countl	投诉数量	Int	32	

6 邀请码表

数据项名	数据项含义	类型	长度	完整性约束
id	邀请码 id	int	15	主键
code	邀请码	varchar	128	
user	被邀请人	varchar	128	

4.1.2 触发器的设计

1 在对 user 表进行插入修改操作时，判断 id 是否正确

```
CREATE TRIGGER user_insert
BEFORE INSERT
ON user FOR EACH ROW
if new.id>0
    insert int xxxx values(1);
end if
```

2 编辑好友信息朋友圈权限时，判断是否为 0 或 1

```
CREATE TRIGGER ifbool
BEFORE INSERT
ON ss_node  FOR EACH ROW
if node_info == 0 or new.node_info==1
    update node_info = new.node_info
end if
```

3 投诉某个人时，相应的投诉表的投诉数量+1

```
CREATE TRIGGER `hhdg`
AFTER INSERT ON `complaints`
 FOR EACH ROW UPDATE complaints_users
SET countl=countl+1 WHERE new.node_name =
complaints_users.user_name
```

4.1.3 存储过程的设计

1 查询信息的存储过程

```
drop procedure if exists sl;
-- 如果存在就 drop
delimiter //
-- 定义为碰到//才表示语句结束
create procedure sl(name varchar)
begin
     select * from user where user_name > name;
end //
delimiter ;
```

2 插入信息的存储过程

```
create procedure sp1(id int,name varchar(128),pass
varchar(128))
begin
    insert into user values(id,name,pass);
     select * from user;
end //
delimiter ;
```

3 数据修改的存储过程

```
delimiter //
-- 定义为碰到//才表示语句结束
create procedure up(name varchar(128),id int(15))
begin
       UPDATE user  SET id = id WHERE user_name = name;
       select * from user;
end //
delimiter ;
-- 定义为碰到;表示语句结束
```

4.1.4 数据库视图的设计

SN	字段名称	描述	类型
1	user_name	用户名	varchar(128)
2	id	id	varchar(128)
3	node_server	备注	bool(10)
4	node_type	是否为好友	bool(10)
5	node_status	标签	bool(10)

4.1.5 数据库函数的设计

1 根据投诉情况情况自动生成投诉者 id 编号

```
create function rid () returns varchar(4)
as begin declare @id varchar(4)
select @id = (select 'r' + right('000' +
cast(max(cast(right(rtrim([user-id]),3)as int))+1
as varchar(3)),3)from users)
return @id
end
```

2、根据用户注册自动生成的 id

```
create function bid () returns varchar(4)
as begin declare @id varchar(4)
select @id = (select 'b'+ right('000'+
cast(max(cast(right(rtrim([ss_node-id]),3)as int))+1
as varchar(3)),3)from ss_node)
return @id
End
```

4.2 详细功能设计

4.2.1 用户功能

1 网站首页

浏览器输入端输入 fly-me.cn:99 即可访问，公网可访问，目前该项目已经部署上服务器。

网站首页包含五个功能模块，分别为网站首页，用户登录，用户注册，用户中心，更多推荐五个功能。默认进入的是网站首页模块。

2 登录注册

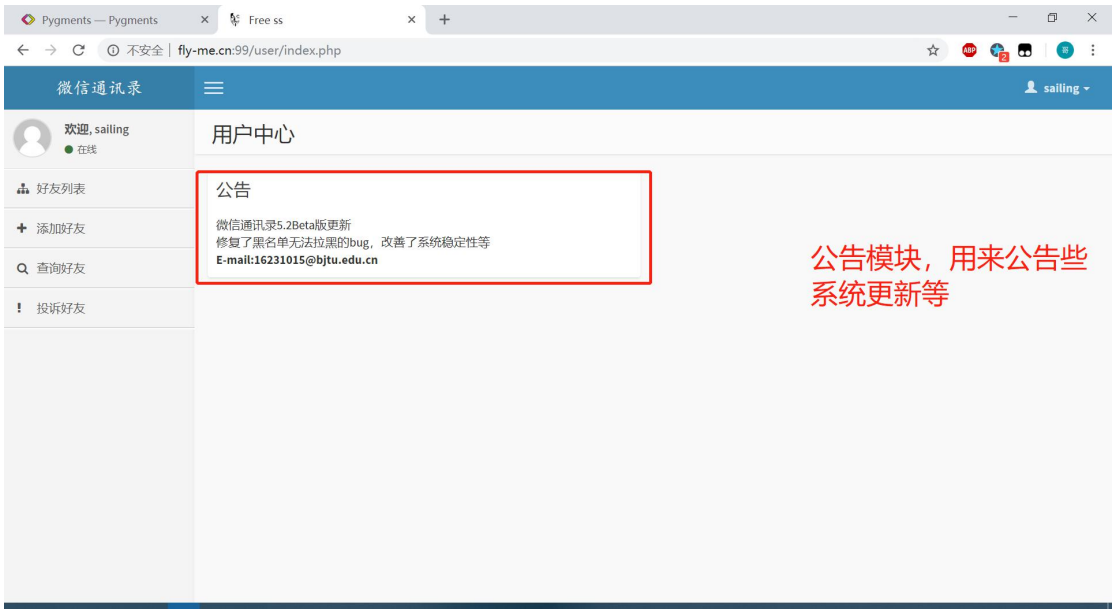
我们在首页点击用户注册，即可注册个用户，需要填写邀请码，邀请码由管理员生成，填写邀请码后可注册。注册完用户后，进入登录界面，就可以进入微信通讯系统了，


Free ss注册
用户名
密码
重复密码
邮箱
邀请码
注册
首页 登录


登录
sailing
保存Cookie7天
登录
首页 注册

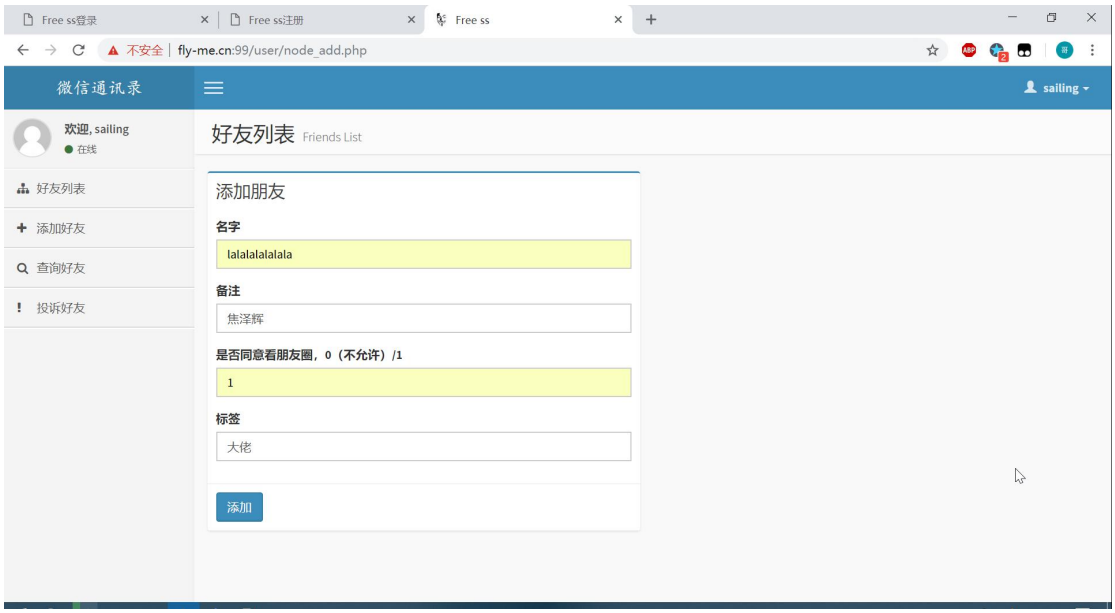
3 进入系统

进入系统后，就是用户中心，用户中心目前显示的系统公告，版本更新要点，以及管理员的联系邮箱

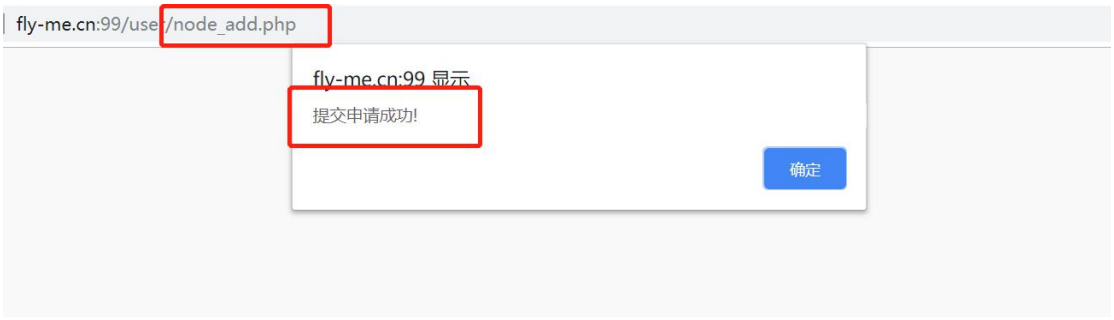


4 添加好友

新用户进入系统后，是没有好友的，所以我们进入添加好友界面，添加好友，在添加好友的时候可以提前给用户设置信息，备注，是否同意看朋友圈，以及标签。

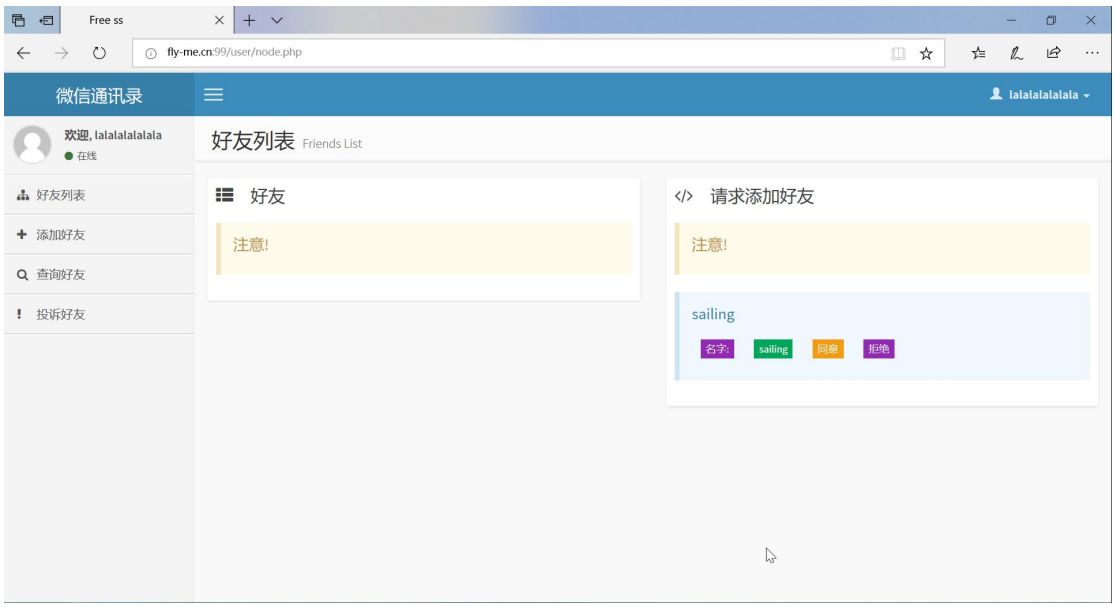


提交申请成功后，点击确定。



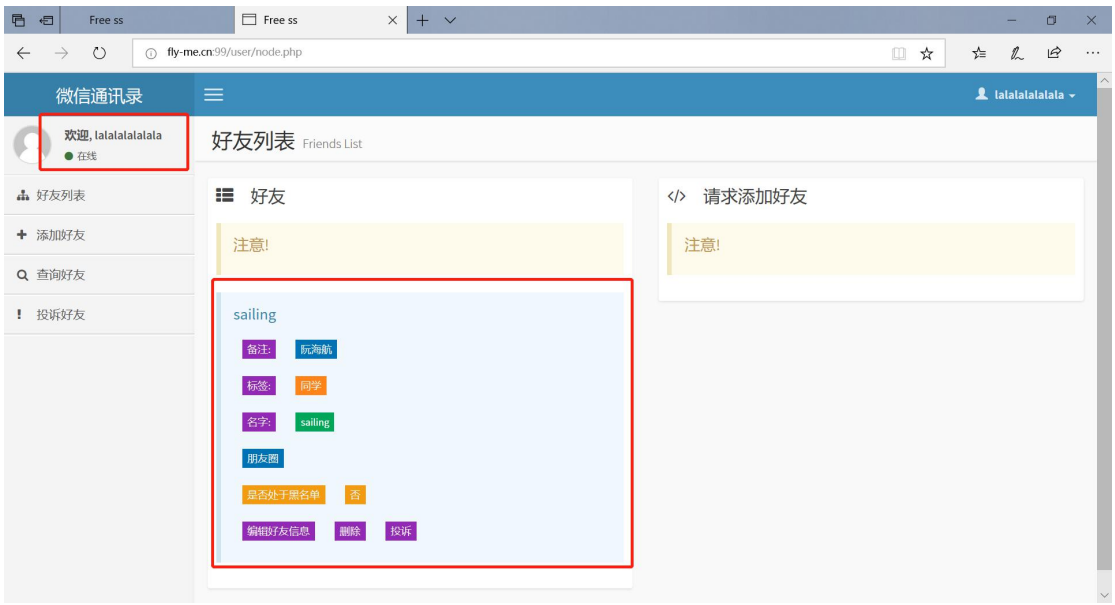
点开自己的好友列表，发下还未加到好友，说明要等对方同意才可以添加好友。

进入焦泽辉的界面，可以看到，sailing 提交了好友申请。焦泽辉可以选择同意或者拒绝

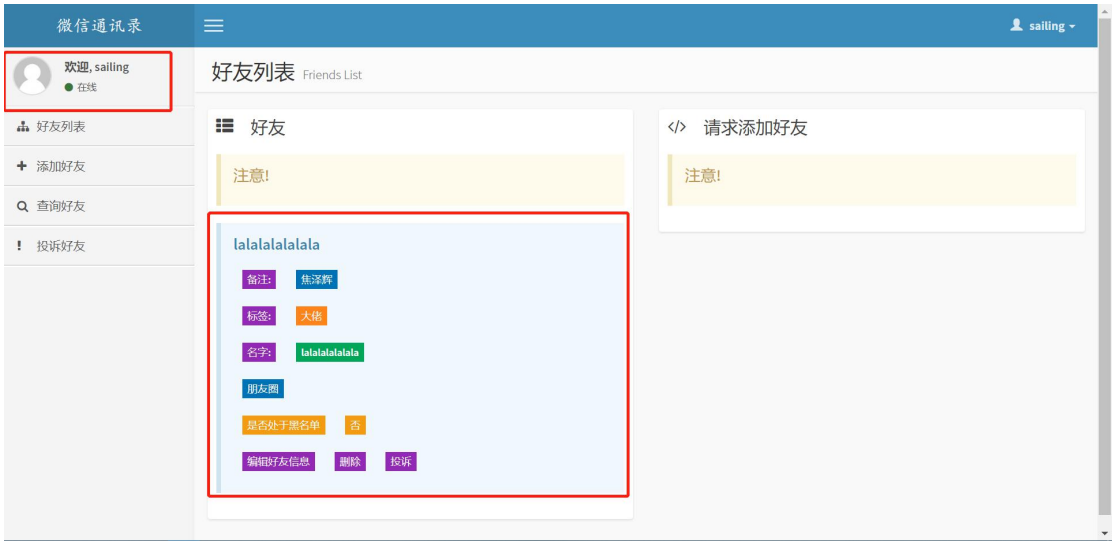


点同意即进入设置好友信息界面，在此可以设置好友的信息，点击修改即进入好友列表界面。

焦泽辉成功添加阮海航为好友，在好友列表中显示出来了。



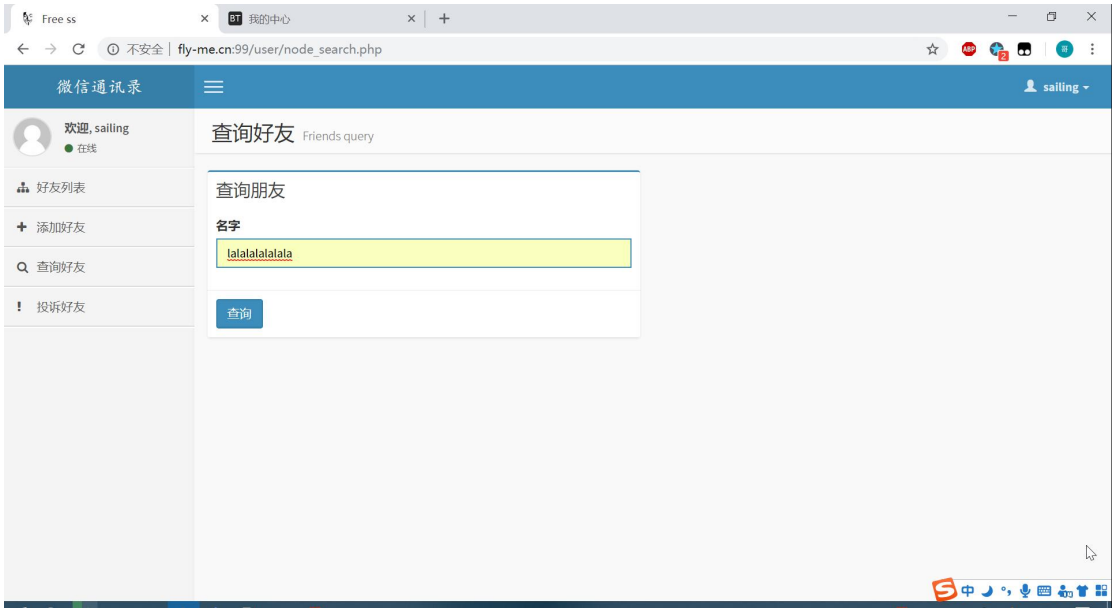
焦泽辉同意添加好友后，然后 sailing 刷新自己的好有列表就可以看出添加好友成功，看到焦泽辉了

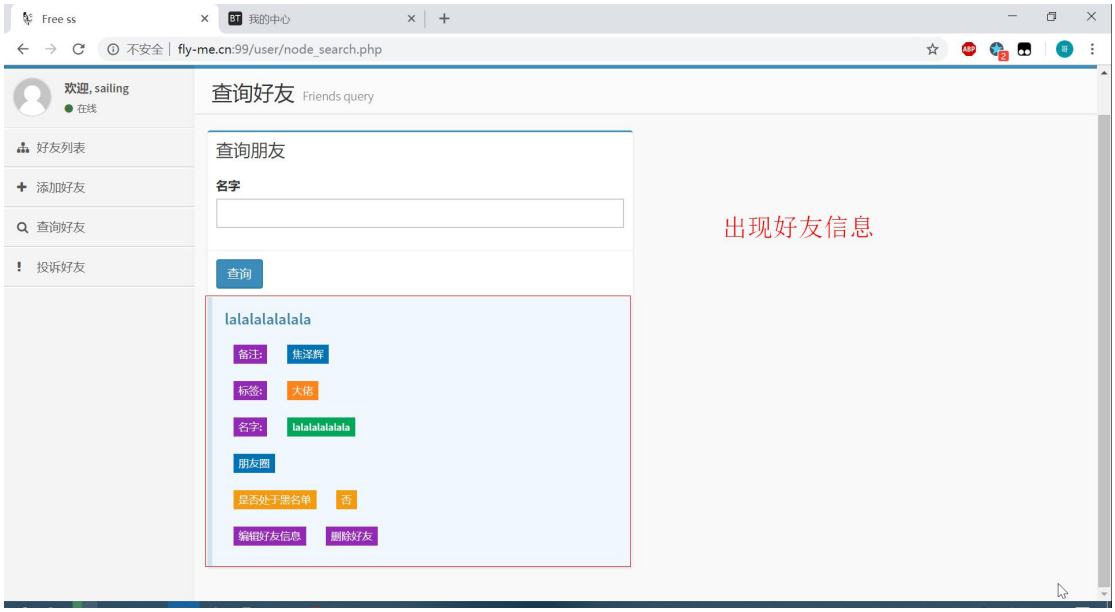


这样双方就添加好友成功了

5 查询好友功能

进入查询界面，输入好友的名字，点击查询，然后系统就会出现好友的信息。在查询到的界面，依然可以对好友信息进行操作。

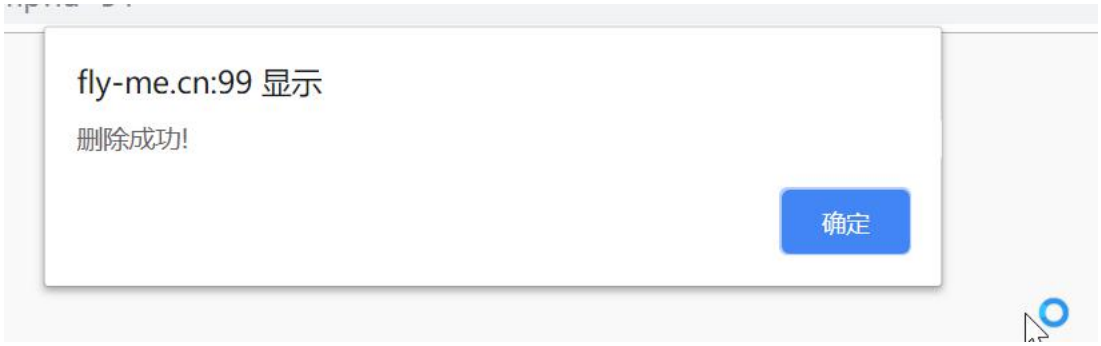




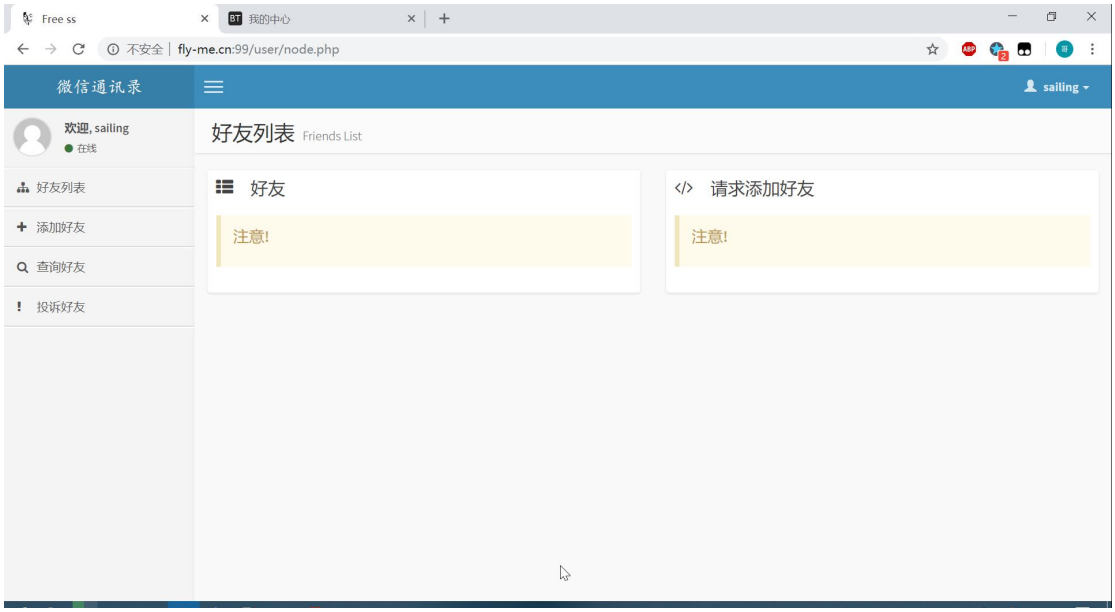
6 删除好友功能

点击删除，即可删除好友

系统显示删除成功

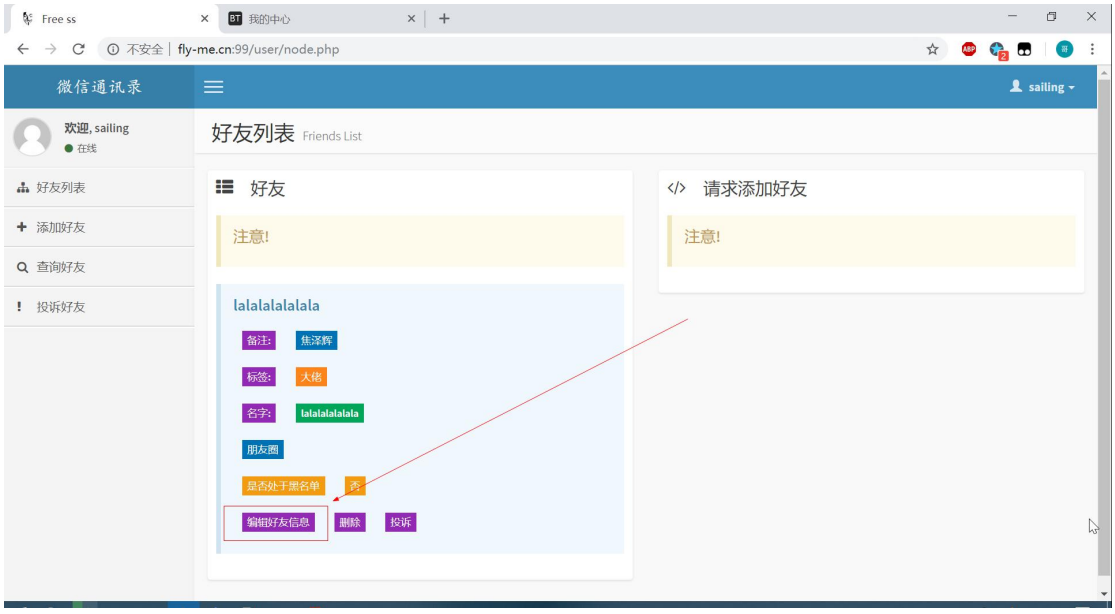


好友列表界面没有那个好友，即删除成功



7 编辑好友信息

在好友列表，可以看到好友的备注，标签，朋友圈权限，是否处于黑名单等，点击编辑好友信息，即可进入好友信息编辑界面。



可以给好友设置备注，标签，朋友圈权限，黑名单等

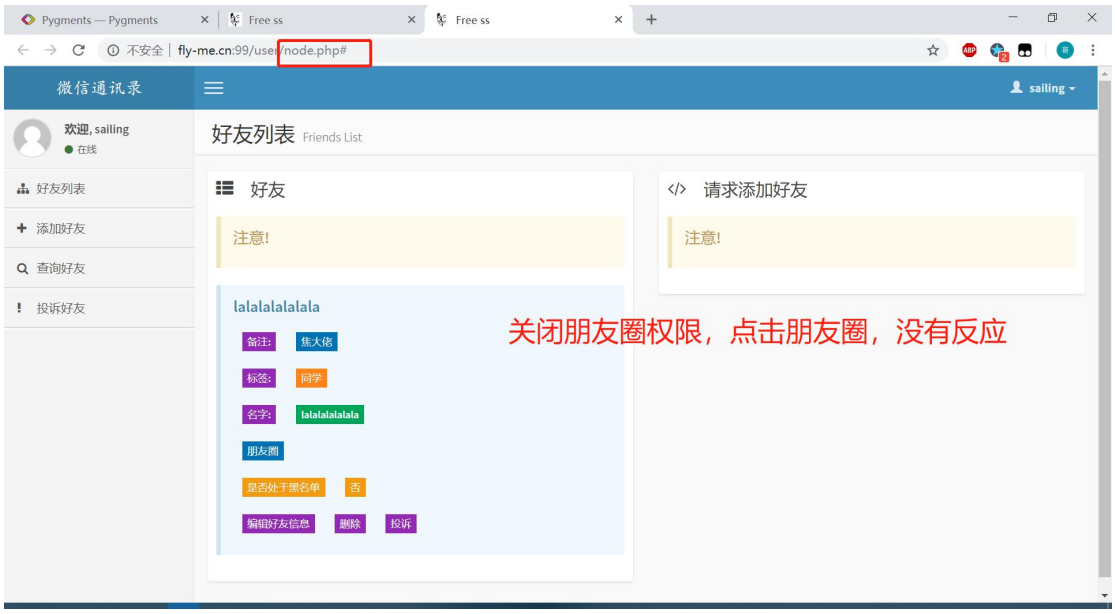
再次我们将备注修改为焦大佬，标签修改为同学。

然后跳转到好友列表界面，可以看出好友的备注和标签被修改了。

设置朋友圈权限，如果可以看得话，点击朋友圈跳转到百度首页，如果朋友圈权限关的话，就无法跳转页面，保持原页面不变。

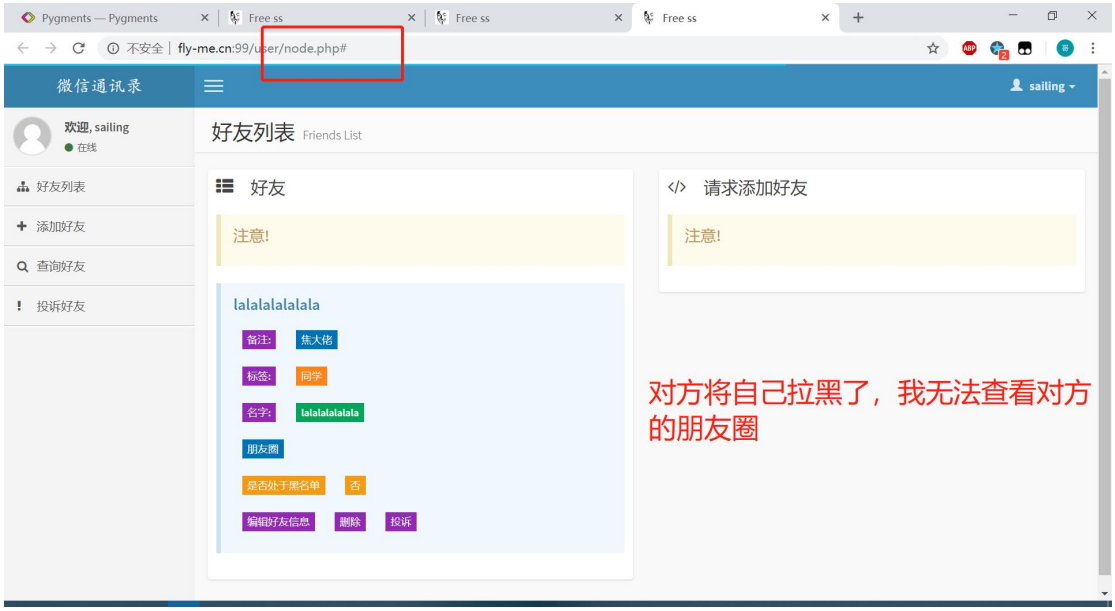

微信通讯录
sailing
欢迎, sailing
在线
好友列表
添加好友
查询好友
投诉好友
好友列表 Friends List
好友
注意!
请求添加好友
注意!
lalalalalalala
备注: 焦大佬
标签: 同学
名字: lalalalalalala
朋友圈
是否处于黑名单 否
编辑好友信息 删除 投诉
此处设置开放朋友圈，点击朋友圈，就跳转到百度首页
正在等待 secure.gravatar.com 的响应...


微信通讯录
sailing
欢迎, sailing
在线
好友列表
添加好友
查询好友
投诉好友
好友列表 Friends List
编辑好友信息
备注
焦大佬
是否看对方朋友圈，0（不允许）/1
0
标签
同学
是否加入黑名单 0为不加入，1为加入
0
修改
此处关闭朋友圈权限


Pygments — Pygments
Free ss
Free ss
不安全 fly-me.cn:99/user/node.php#
微信通讯录
sailing
欢迎, sailing
在线
好友列表
添加好友
查询好友
投诉好友
好友列表 Friends List
好友
注意!
请求添加好友
注意!
lalalalalalala
备注: 焦大佬
标签: 同学
名字: lalalalalalala
朋友圈
是否处于黑名单 否
编辑好友信息 删除 投诉
关闭朋友圈权限，点击朋友圈，没有反应

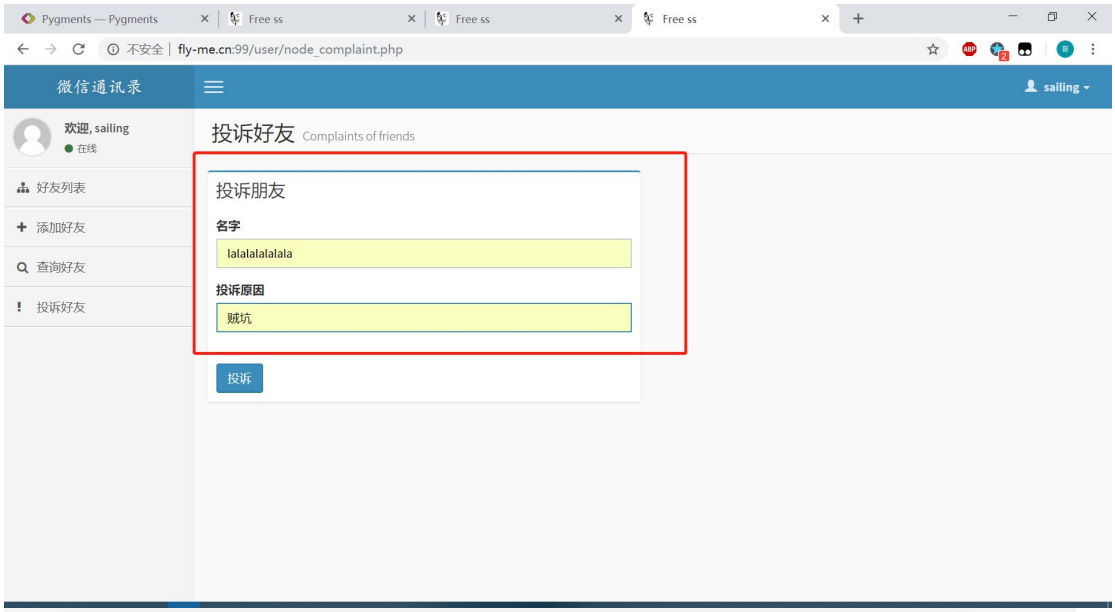
就算开放了朋友圈权限，对方把自己拉黑了，我也看不了对方的朋友圈的。

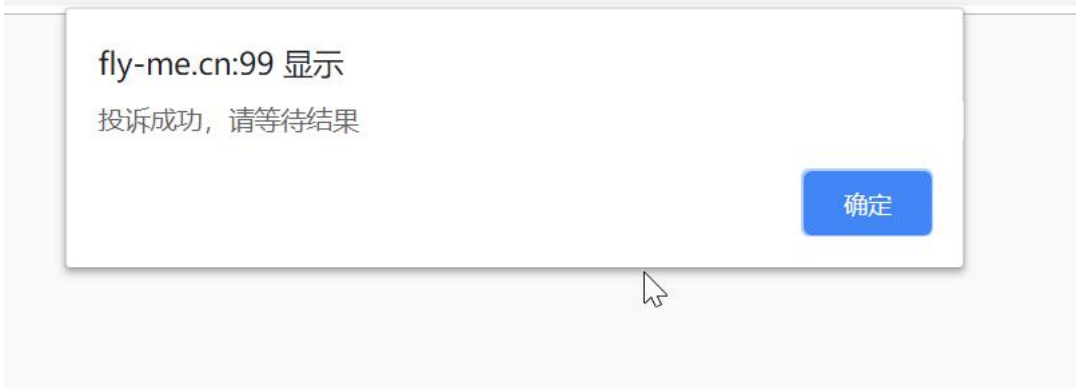
阮海航在自己的系统是访问不了焦泽辉的朋友圈的



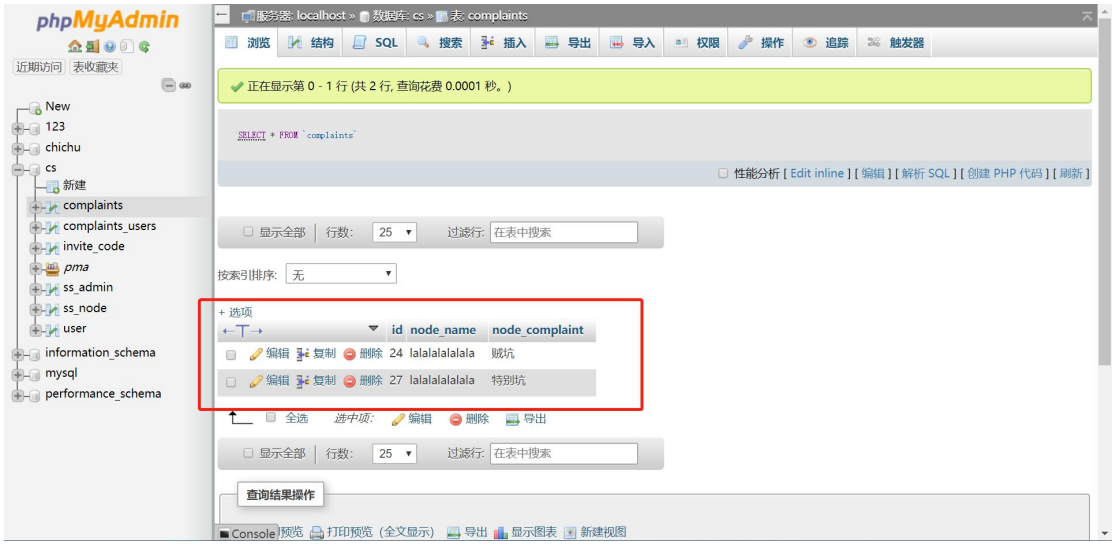
8 投诉好友

点击投诉，然后输入投诉理由，即可提交投诉，然后等待系统审核

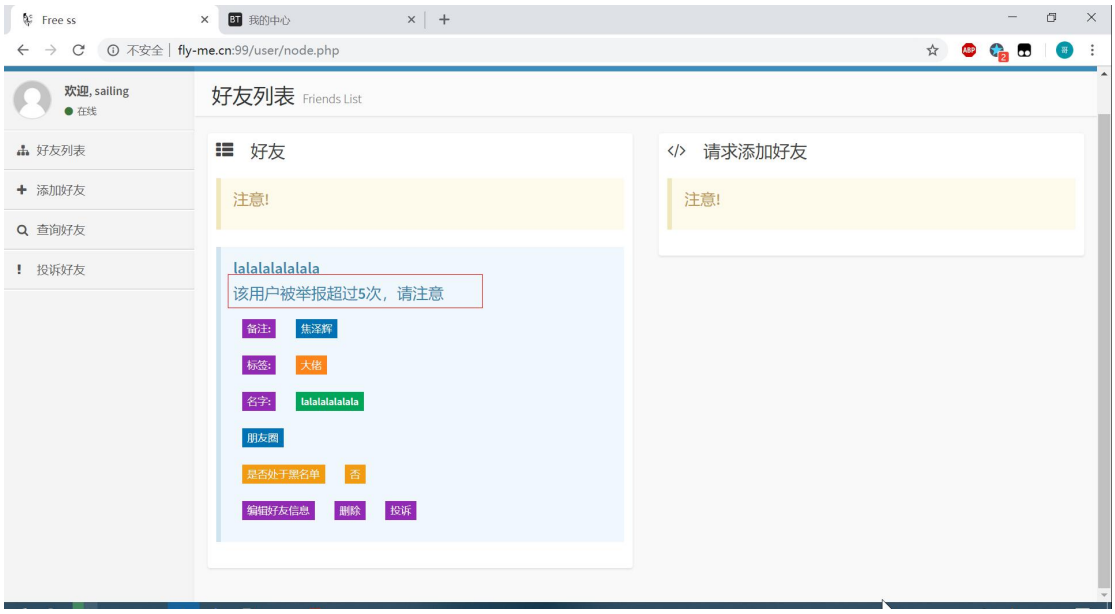




在数据库表中，可以看到具体的投诉信息。

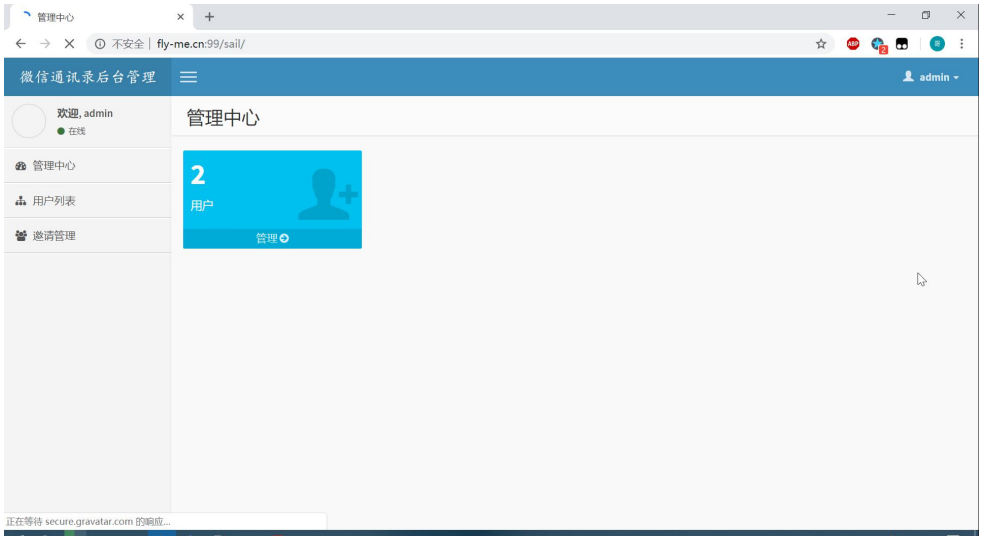


如果一个好友投诉超过 5 次，他的好友列表就会显示该好友投诉次数超过 5 次，请注意



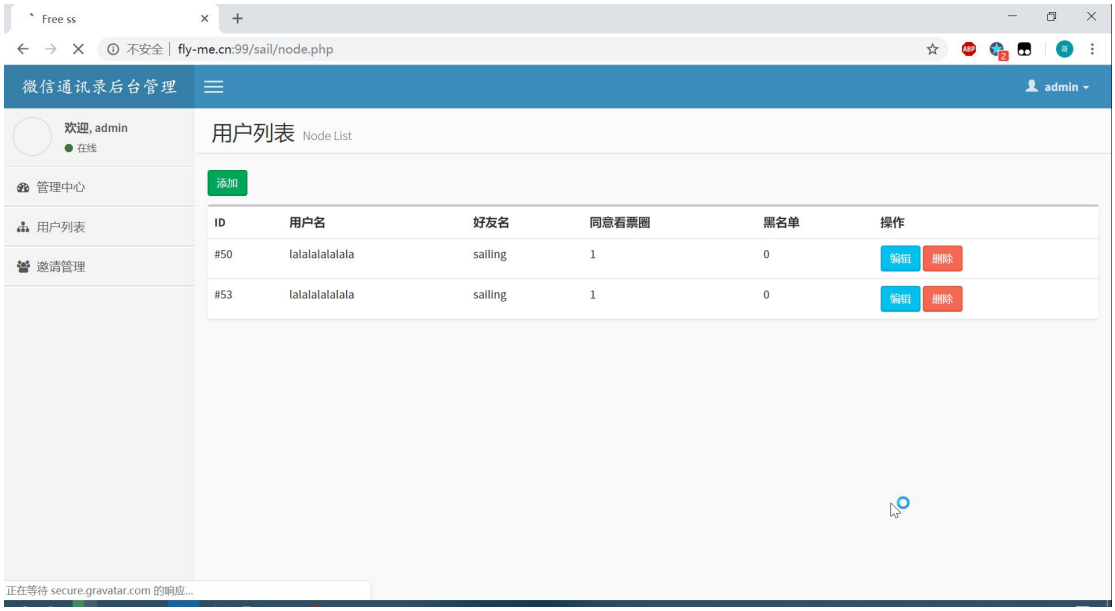
4.2.2 管理员功能

浏览器框输入 http://fly-me.cn:99/sail/即可进入管理员系统，输入账号密码后登陆



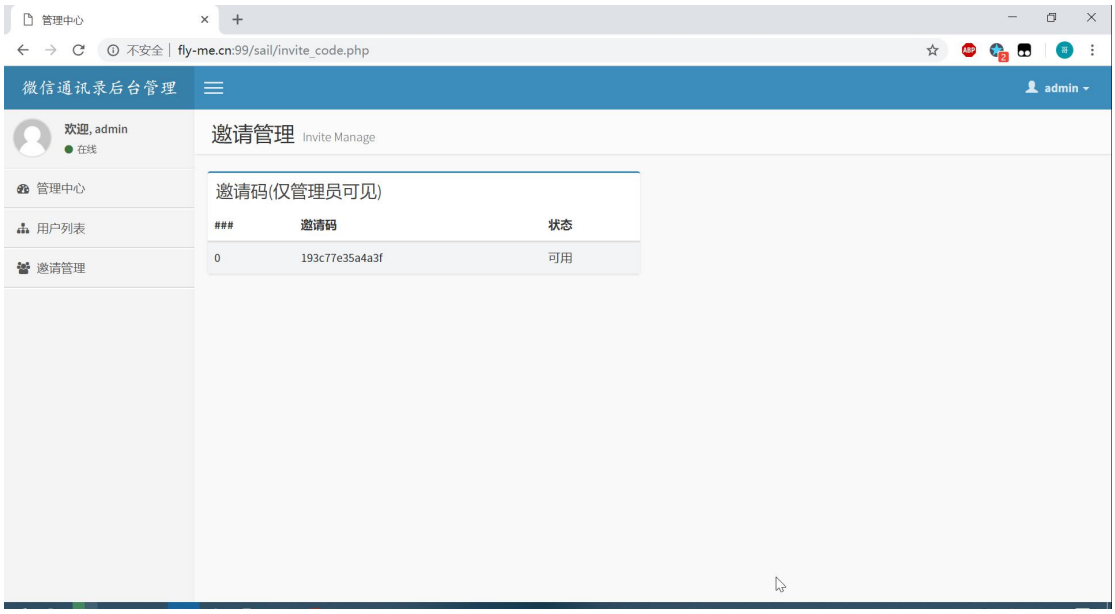
1 用户管理

管理员可在后台看到用户列表，可对用户进行编辑删除信息等操作



2 邀请码管理

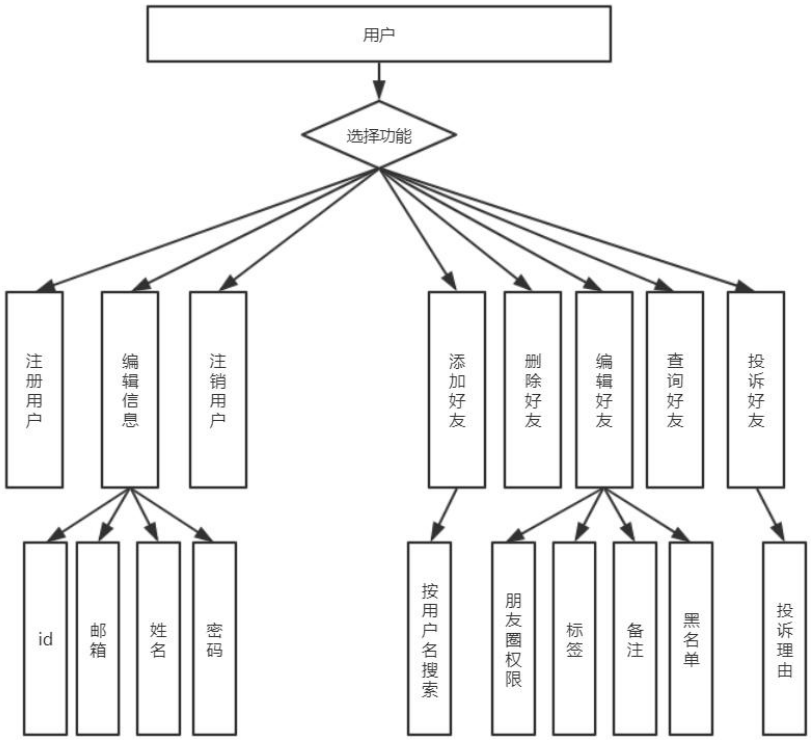
邀请码为用户注册所必须要填写的，由管理员生成，管理可生成三中类型的邀请码，0 为公开，1 为管理员可见，2 位已注册用户可见

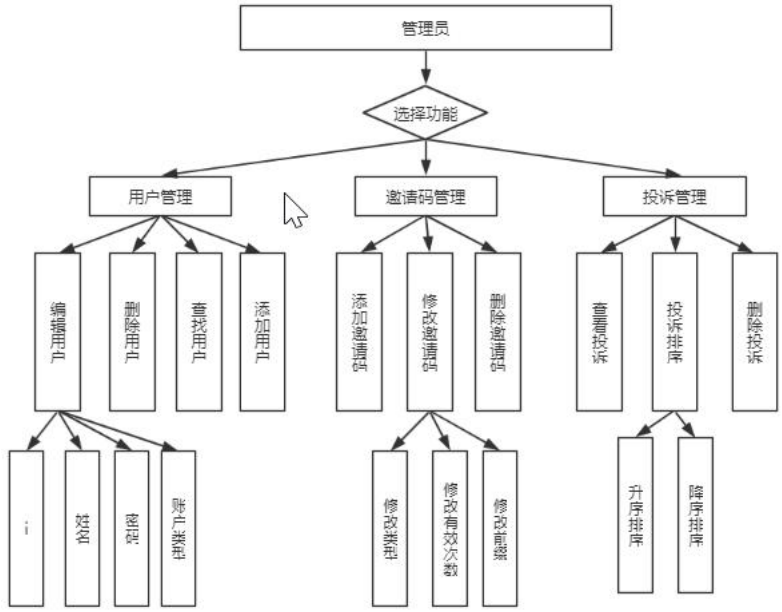


第五章 系统实现

5.1 程序框图

进入本系统前首先需要进行登录，由于整个程序框图较大，为显示清晰，我们将其分解为俩个部分进行展示，分别为登录部分的程序框图、管理员的程序框图及读者的程序框图





5.2 关键技术

5.2.1 Md5 加密算法

```
$pwd    = md5($_POST['password']); //md5 加密
```

5.2.2 邮箱特殊字符校验

```
//对电子邮件的验证
//定义 email 正则表达式
        var email_reg =
/^([a-zA-Z0-9]+[_|\_|\.]?)*[a-zA-Z0-9]+@([a-zA-Z0-9]+[_|\_|\.]
?)*[a-zA-Z0-9]+\.[a-zA-Z]{2,3}$/;

        if(!email_reg.test(document.reg.email.value)) {
            alert('提示\n\n 请输入有效的 E-mail! ');
            document.reg.email.focus();
            return false;
        }

        // 特殊字符检验
        var no_allow_txt = new
RegExp("[\\*,\\&,\\\\,\\/,\\?,\\|,\\:,\\<,\\>,\"]");
        if(no_allow_txt.test(document.reg.username.value)){
            alert("用户名不允许含特殊字符");
            document.reg.username.focus();
            return false;
        }
```

5.2.3 获取对方数据库值判断是否将自己拉黑

对方将自己拉黑，我是无法查看对方的朋友圈的，而拉黑权限时数据表里面 node_order 属性，所以需要查询对方的数据库 node_order 对自己是否设置为 1, 1 为拉黑

```
<div class="box-body">
                    <div class="callout callout-warning">
                        <h4>注意!</h4>
                        <p></p>
                    </div><?php
                    $abc=$_COOKIE['user_name'];
                    $sql ="SELECT * FROM `ss_node` WHERE `node_type` =
'0' AND `node_name` ='$abc' ORDER BY node_order ";//node_type = 0 表示
双方都是好友。=1 表示请求添加为好友
                    $query =  $dbc->query($sql);
                    while ( $rs = $query->fetch_array()){
                    $abcd=$rs['node_method'];
```

```
                    $sql ="SELECT * FROM `ss_node` WHERE `node_method`
= '$abc' AND `node_name` ='$abcd'";
                    $queryy =  $dbc->query($sql);
                    $rss = $queryy->fetch_array()
                    ?>
                        <div class="callout callout-info">
                            <h4><?php echo $rs['node_method'];
                              ?></h4>
                            <h4><?php $a = $rs['node_method'];
                                    $sql ="SELECT * FROM
`complaints_users` WHERE `user_name` = '$a'";
                                    $q =  $dbc->query($sql);
                                    $acf = $q->fetch_array();
                                    if($acf['countl'] >5) echo "该用户被举
报超过 5 次，请注意"; ?></h4>
                                <p>
                                    <a class="btn btn-xs bg-purple btn-flat
margin" href="#">备注:</a>
                                    <a class="btn btn-xs bg-blue btn-flat
margin" ><?php echo $rs['node_server']; ?></a><br>
                                    <a class="btn btn-xs bg-purple btn-flat
margin" href="#">标签:</a>
                                    <a class="btn btn-xs bg-orange btn-flat
margin" href="#"><?php echo $rs['node_status']; ?></a><br>
                                    <a class="btn btn-xs bg-purple btn-flat
margin" href="#">名字:</a>
                                    <a class="btn btn-xs bg-green btn-flat
margin" href="#"><?php echo $rs['node_method']; ?></a><br>

                                    <a class="btn btn-xs bg-blue btn-flat
margin" target="_blank" href="<?php if($rs['node_info'] ==1 &&
$rss['node_order'] ==0) echo 'http://www.baidu.com';else echo '#'; ?>">
朋友圈</a><br>
                                    <a class="btn btn-xs bg-yellow btn-flat
margin"  href="#">是否处于黑名单<!--你仍然可以在列表中看到他,但是他无法再添加你
为好友你也无法看他的--></a>
                                    <a class="btn btn-xs bg-yellow btn-flat
margin"   href="#"><?php if($rs['node_order']==1)echo "是";else echo "
否"; ?></a><br>
                                    <a class="btn btn-xs bg-purple btn-flat
margin" href="node_edit.php?id=<?php echo "$rs[id]" ?>">编辑好友信息</a>
                                    <a class="btn btn-xs bg-purple btn-flat
margin" href="node_del.php?id=<?php echo "$rs[id]" ?>">删除</a>
```

```
                                <a class="btn btn-xs bg-purple btn-flat
margin" href="node_complaint.php?id=<?php echo "$rs[id]" ?>">投诉</a>
                            </p>
                        </div>
                    <?php }?>
                </div><!-- /.box-body -->
```

5.2.4 检测是否登录

```
//检测是否登录，若没登录则转向登录界面
if(!isset($_COOKIE['user_name'])||!isset($_COOKIE['user_uid'])
||!isset($_COOKIE['user_pwd'])){
    header("Location:login.php");
    exit();
}else{
    //co
    $uid = $_COOKIE['user_uid'];
    $user_name = $_COOKIE['user_name'];
    $user_pwd  = $_COOKIE['user_pwd'];
    $user_email = get_user_email($uid);

    //验证 cookie
    $pw = get_user_pass($uid);
    $pw = co_pw($pw);
    if($pw != $user_pwd){
        header("Location:login.php");
    }
}
```

第六章 系统维护

6.1 恢复数据库的方法

对于事物内部故障

恢复时要在不影响其他事务运行的情况下，强行回滚该事务，即撤销该事务已经做出的任何的对数据库的修改；

对于系统故障

一方面，在系统重新启动时让所有非正常终止的事务滚回，强行撤销所有未完成事务；另一方面，把已完成的事务提交的结果重新写入数据库；

对于介质故障

在故障发生前对数据库进行转储，即使用数据库镜像功能，根据实际情况要求自动把整个数据库或其中的关键数据复制到另一个磁盘上，防患于未然。

6.2 数据库恢复实现技术

采用转储的方法定期地将整个数据库复制到磁带或另一个磁盘上保存起来，称为后备副本或后援副本。（动态转储和静态转储）；

通过登记日志文件，进行事务故障恢复和系统故障恢复，并协助后备副本进行介质故障恢复。

6.3 数据库恢复策略

事务故障的恢复

（1）反向扫描日志文件，查找该事务的更新操作；

（2）对该事务的更新操作执行逆操作，即将日志记录中“更新前的值”写入数据库；

（3）继续反向扫描日志文件，查找该事物的其他操作，做同样处理，直到事物开始标志为止。

系统故障的恢复

（1）正向扫描日志文件，找出在故障发生前已经提交的事务，

将其事务标识记入重做队列。同时找出故障发生时尚未完成的事务，将其事务标识记入撤销队列。

（2）反向扫描日志文件，对每个 UNDO 事务的更新操作执行逆操作，即将日志记录中“更新前的值”写入数据库。

（3）正向扫描日志文件，对每个 REDO 事务重新执行日志文件登记操作。即将日志记录中“更新前的值”写入数据库。

介质故障的恢复：重装数据库

（1）装入最新的数据库后备副本，使数据库恢复到最近一次转储时的一致性状态。

（2）装入相应的日志文件副本，重做已完成的事务。

第七章 系统设计总结

我觉得每一次有难度的课程设计对于我来说都像一次挑战，一次历练。本次系统设计实现对数据库视图、存储过程、触发器、规范化要求、索引等方面的设计，对需求方案，系统稳定性分析、系统维护等方面进行了全面的阐述和实现。我们采用 php+mysql 的形式开发系统，最后将系统部署到服务器上，实现访问。在完成基础功能，附加功能的基础上，额外完成了登录注册，md5 加密，投诉超过一定数量系统提示，邀请码等功能。本次数据库系统设计让我更加的了解了后端开发，对以后开发出更加完善的系统有着重要的帮助，让我受益匪浅。