

# 《软件安全》实验报告

姓名：王众

学号：2313211

## 实验名称：

WEB开发实践

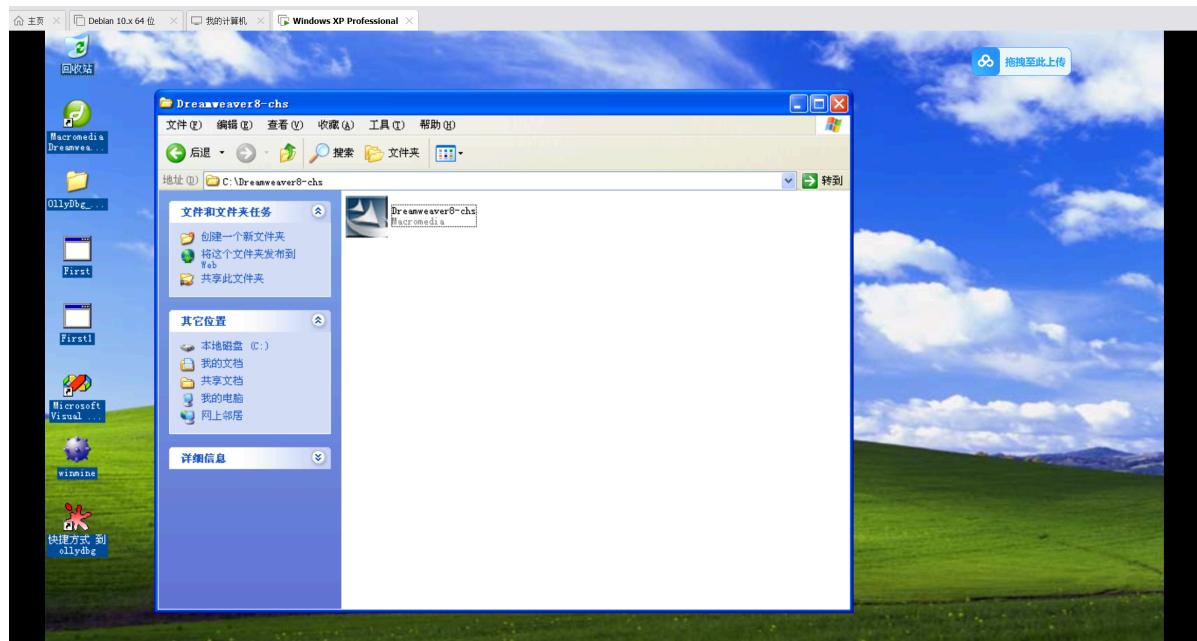
## 实验要求：

复现课本第十章的实验三(10.3.5节):利用php, 编写简单的数据库插入、查询和删除操作的示例。基于课本的完整的例子，进一步了解WEB开发的细节。

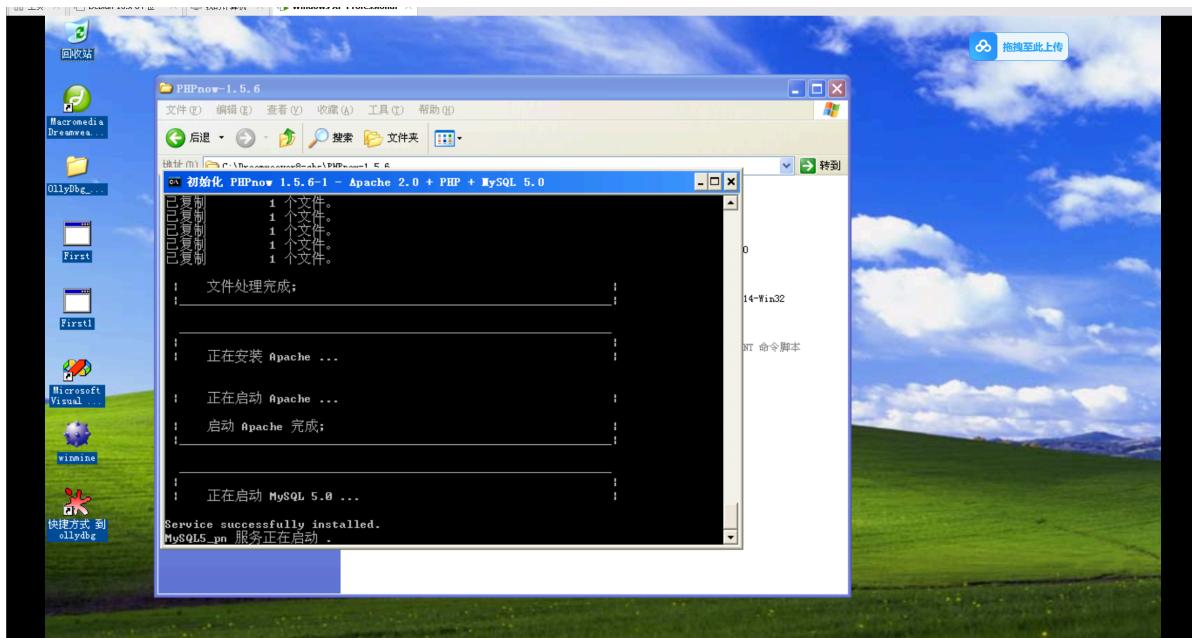
## 实验过程：

### 1 配置环境

我们使用的是xp系统。首先，我们使用资料库中提供的资源，进行解压缩，存放在我们的课程作业文件夹中，将其修改为了C盘内的路径，这样就不会出问题了。



安装完 Dreamweaver 之后，我们继续安装下一个软件：PHPnow。我们按照流程进行安装。



接下来我们设置 `root` 的密码为000000，之后就跳转到了网址的页面。

The screenshot shows the Microsoft Internet Explorer browser displaying the PHPnow Works! homepage. The address bar shows `http://127.0.0.1/`. The page content includes:

- Server Information** table:

SERVER_NAME	127.0.0.1
SERVER_ADDR:PORT	127.0.0.1:80
SERVER_SOFTWARE	Apache/2.0.63 (Win32) PHP/5.2.14
PHP_SAPI	apache2handler
php.ini	C:\Dreamweaver8-chs\PHPnow-1.5.6\php-5.2.14-Win32\php-apache2handler.ini
网站主目录	C:/Dreamweaver8-chs/PHPnow-1.5.6/htdocs
Server Date / Time	2025-05-16 17:24:07 (+08:00)
Other Links	<a href="#">phpinfo()</a>   <a href="#">phpMyAdmin</a>
- PHP 组件支持** table:

Zend Optimizer	Yes / 3.3.3
MySQL 支持	Yes / client lib version 5.0.90
GD library	Yes / bundled (2.0.34 compatible)
eAccelerator	No
- MySQL 连接测试** (This section is currently empty.)

可以看到我们成功的安装了 `PHPnow`，进入了专用的本地IP `127.0.0.1`！我们输入我们的密码 `123456`，就可以发现成功的连接了我们的数据库。

The screenshot shows the Microsoft Internet Explorer browser window with the title "PHPnow Works! - Microsoft Internet Explorer". The address bar contains the URL "http://127.0.0.1/index.php". The page content is a PHP configuration report:

- Server Information**

SERVER_NAME	127.0.0.1
SERVER_ADDR:PORT	127.0.0.1:80
SERVER_SOFTWARE	Apache/2.0.63 (Win32) PHP/5.2.14
PHP_SAPI	apache2handler
php.ini	C:\Dreamweaver8-chs\PHPnow-1.5.6\php-5.2.14-Win32\php-apache2handler.ini
网站主目录	C:/Dreamweaver8-chs/PHPnow-1.5.6/htdocs
Server Date / Time	2025-05-16 17:25:41 (+08:00)
Other Links	<a href="#">phpinfo()</a>   <a href="#">phpMyAdmin</a>
- PHP 组件支持**

Zend Optimizer	Yes / 3.3.3
MySQL 支持	Yes / client lib version 5.0.90
GD library	Yes / bundled (2.0.34 compatible)
eAccelerator	No
- MySQL 连接测试**

MySQL 服务器	localhost	MySQL 数据库名	test
MySQL 用户名	root	MySQL 用户密码	
<input type="button" value="连接"/>			
- MySQL 测试结果**

服务器 localhost	OK (5.0.90-community-nt)
数据库 test	OK

这样的话，本次实验所需要的环境就都配置完毕了！接下来我们就可以开始本次实验了。

## 2 实验复现

下面我们复现课本第十章的实验三(10.3.5节)：利用 `php`，编写简单的数据库插入、查询和删除操作的示例。这个实验我们要完成的是利用 `php` 完成对一个数据库中的信息进行一些常规操作，我们可以归纳为以下的步骤：

1. 建立数据库
2. 编写 `login.htm` 文件
3. 编写 `loginok.php` 文件
4. 编写 `sys.php` 文件
5. 编写 `add.php` 文件
6. 编写 `del.php` 文件
7. 编写 `index.php` 文件
8. 编写 `news.php` 文件

下面我们就一步一步来进行实现。

### 2.1 建立数据库

我们首先需要建立自己的数据库。我们先打开 `phpMyAdmin`，输入账号和密码来进行登录。

MySQL

服务器: localhost via TCP/IP  
服务器版本: 5.0.90-community-nt  
协议版本: 10  
用户: root@localhost  
MySQL 字符集: UTF-8 Unicode (utf8)

网站服务器

Apache/2.0.63 (Win32) PHP/5.2.14  
MySQL 客户端版本: 5.0.90  
PHP 扩展: mysql

phpMyAdmin

版本信息: 3.3.7  
文档  
维基 (Wiki) (外链, 英文)  
官方主页 (外链, 英文)  
[ChangeLog] [Git] [Lists]

我们需要完成的是数据库的新建。

数据库:

表 1: News (newsid, topic, content)

表 2: userinfo (username, password)

首先我们新建，获得数据库。

结构 SQL 搜索 查询 导出 导入 操作 权限 删除

创建数据库 test2 成功。  
CREATE DATABASE `test2` ;

在数据库 test2 中新建一个数据表

名字: newsid  
字段数: 3

执行

然后我们在新建的数据库中新建数据表，分别是 News 和 userinfo。

**userinfo**

CREATE TABLE `testdb`.`userinfo` (  
`username` VARCHAR( 30 ) CHARACTER SET latin1 COLLATE latin1\_swedish\_ci NOT NULL ,  
`password` VARCHAR( 30 ) CHARACTER SET latin1 COLLATE latin1\_swedish\_ci NOT NULL  
) ENGINE = MYISAM ;

**字段** **类型** **整理** **属性** **空** **默认** **额外** **操作**

username	varchar(30)	latin1_swedish_ci	否	无			
password	varchar(30)	latin1_swedish_ci	否	无			

**已用空间**

类型	已用
数据	0 字节
索引	1,024 字节
总计	1,024 字节

**新闻**

CREATE TABLE `testdb`.`News` (  
`newsid` INT( 11 ) NOT NULL ,  
`topic` VARCHAR( 30 ) CHARACTER SET latin1 COLLATE latin1\_swedish\_ci NOT NULL ,  
`content` TEXT CHARACTER SET latin1 COLLATE latin1\_swedish\_ci NOT NULL  
) ENGINE = MYISAM ;

**字段** **类型** **整理** **属性** **空** **默认** **额外** **操作**

newsid	int(11)		否	无			
topic	varchar(30)	latin1_swedish_ci	否	无			
content	text	latin1_swedish_ci	否	无			

**已用空间**

类型	已用
数据	0 字节
索引	0 字节
总计	0 字节

这样我们就完成了数据表的新建！

## 2.2 编写 login.html 文件

下面我们开始进入到 php 文件的编写部分，首先我们编写登录的部分代码，根据源代码，我们写入如下的程序即可。

```
<html>
<body>
<form id="form1" name="form1" method="post" action="loginok.php">
<table width="900" border="0" cellspacing="0" cellpadding="0">
<tr>
```

```

<td height="20">姓名</td>
<td height="20"><label>
<input name="username" type="text" id="username" />
</label></td>
</tr>
<tr>
<td height="20">口令</td>
<td height="20"><label>
<input name="pwd" type="password" id="pwd" />
</label></td>
</tr>
<tr>
<td height="20"> </td>
<td height="20"><label>
<input type="submit" name="Submit" value="提交" />
</label></td>
</tr>
</table>
</form>
</body>
</html>

```

我们只需要将文件放到 PHPnow 的 htdoc 目录下，创建属于自己的 web 库即可。这样我们就完成了该部分的代码创建，我们进行一下测试：打开 html 网页，我们发现出现了以下的界面。



说明该部分的代码编写成功了，在上面的表单中，定义了接受表单输入的处理文件为“`loginok.php`”，该文件是检查用户输入的帐号密码是否正确。我们继续编写后面的代码，即用来检测前一部分的输入是否正确。

## 2.3 编写 `Loginok.php` 文件

本文件的目的是验证在 `login.html` 文件打开的网页上所输入的帐号和密码是否在数据库的 `userinfo` 表中，从而来处理用户的登录，即成功的话载入下一个界面，不成功的话返回登录界面重新登陆。为此，我们先在 `userinfo` 表中插入一条帐号为 `admin`，密码为 `123456` 的记录：

The screenshot shows the phpMyAdmin interface for the testdb database. In the 'userinfo' table, a new row has been inserted with the values 'admin' for 'username' and '123456' for 'password'. The '执行' (Execute) button has been clicked, and the message '以新行插入 然后' (Insert into new row then) is displayed above the input fields.

我们插入完之后进行检查，确实在数据库中出现了一条记录。里面存放着我们的账号和密码。

The screenshot shows the phpMyAdmin interface displaying the results of a SQL query: 'SELECT \* FROM `userinfo` LIMIT 0 , 30'. The result set shows one row with 'username' as 'admin' and 'password' as '123456'. The '执行' (Execute) button has been clicked, and the message '显示行 0-0 (总计, 查询花费 0.0012 秒)' is displayed above the results.

然后编写 `loginok.php` 文件如下所示。

```

<?php
$loginok=0;
$conn=mysql_connect("localhost", "root", "123456");
$username = $_POST['username'];
$pwd = $_POST['pwd'];
$SQLStr = "SELECT * FROM userinfo where username='$username' and
password='$pwd'";
echo $SQLStr;
$result=mysql_db_query("testDB", $SQLStr, $conn);
if ($row=mysql_fetch_array($result))//通过循环读取数据内容
{

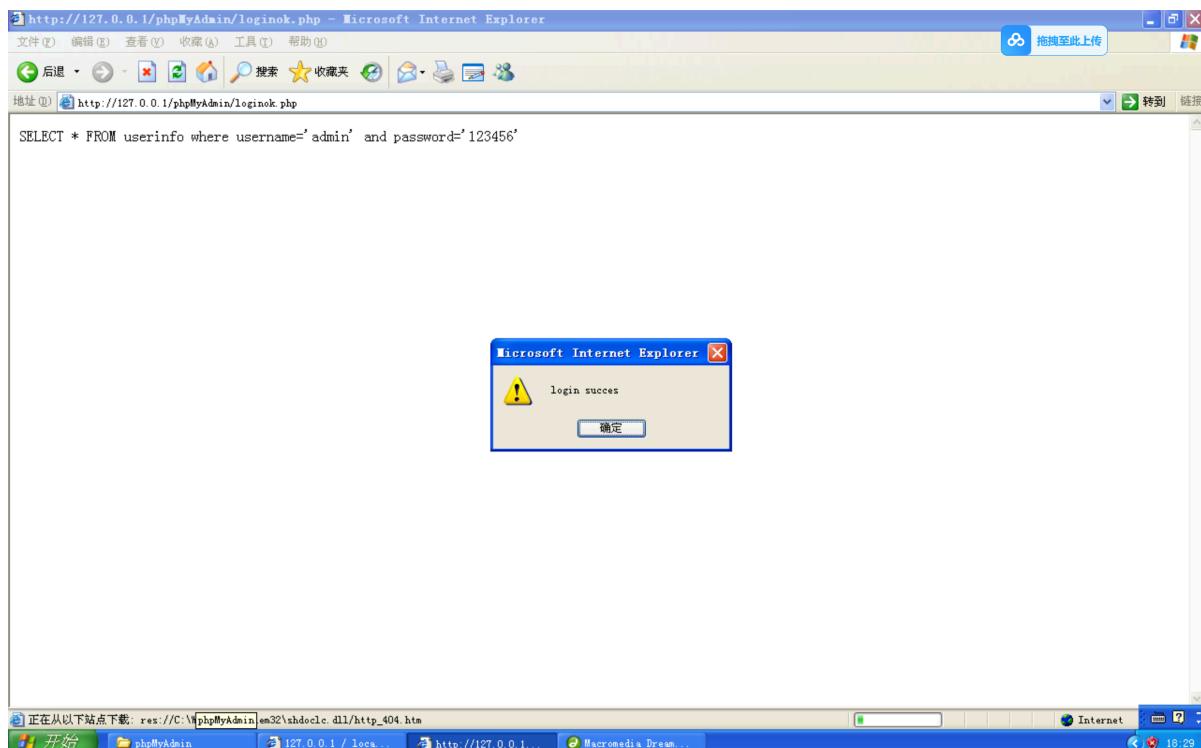
```

```

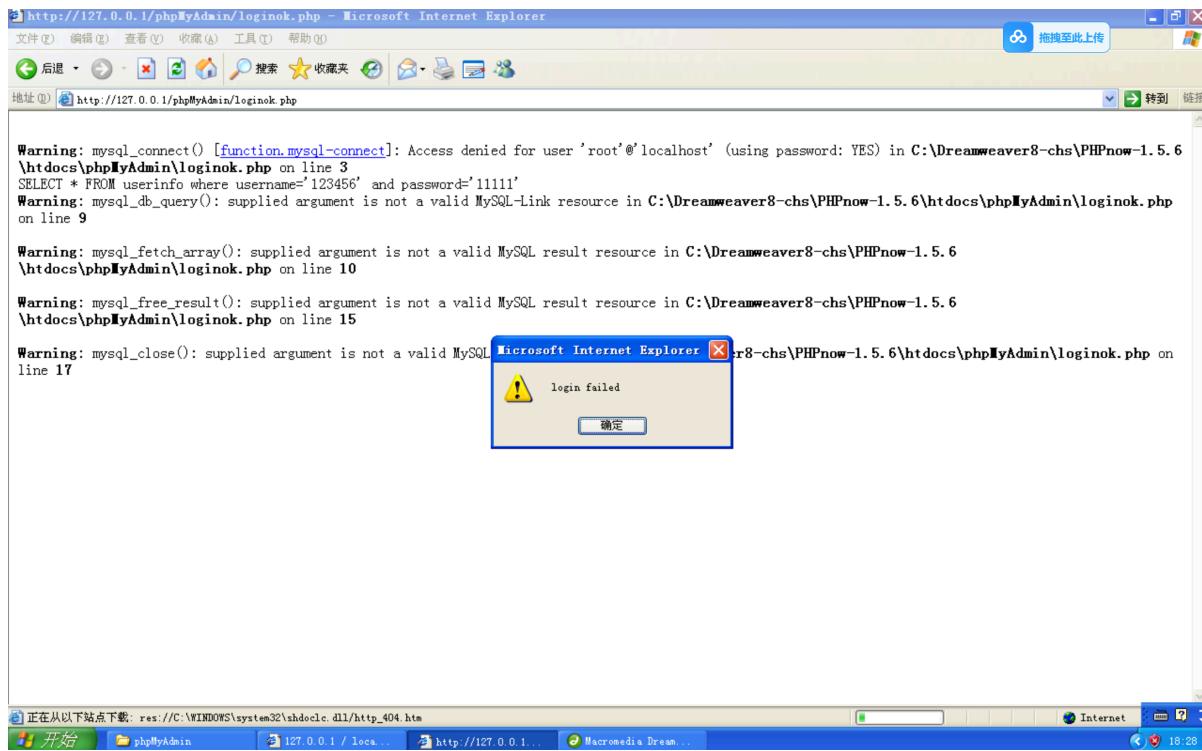
$loginok=1;
}
// 释放资源
mysql_free_result($result);
// 关闭连接
mysql_close($conn);
if ($loginok==1)
{
?>
<script>
alert("login succes");
window.location.href="sys.php";
</script>
<?php
}
else{
?>
<script>
alert("login failed");
history.back();
</script>
<?php
}
?>

```

我们对其进行验证，发现当输入正确的账号和密码时，会输出登陆成功的提示页面。



当我们输入错误的密码时，会提示密码错误，说明我们程序的逻辑是没有问题的。



说明我们本部分的代码也编写成功了！

## 2.4 编写 sys.php 文件

该文件用于我们成功登录后，自主添加新闻信息，包括新闻的标题、内容等，同时还可以选择删除新闻。源码如下：

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>主页</title>
</head>
<?php
$conn=mysql_connect("localhost", "root", "123456");
?>
<body>
<div align="center">
<table width="900" border="0" cellspacing="0" cellpadding="0">
<tr>
<td height="40"><form id="form1" name="form1" method="post" action="add.php">
<div align="right">新闻标题:<br/>
<input name="topic" type="text" id="topic" size="50" />
<BR>
新闻内容:<br/>
<textarea name="content" cols="60" rows="8" id="content"></textarea><br/>
<input type="submit" name="Submit" value="添加" />
</div>
</form>
</td>
</tr>
<tr>
<td><hr /></td>
</tr>
<tr>
```

```

<td height="300" align="center" valign="top"><table width="600" border="0"
cellspacing="0"
cellpadding="0">
<tr>
<td width="100" height="30"><div align="center">新闻序号</div></td>
<td><div align="center">新闻标题</div></td>
<td><div align="center">删除</div></td>
</tr>
<?php
$SQLStr = "select * from news";
$result=mysql_db_query("testDB", $SQLStr, $conn);
if ($row=mysql_fetch_array($result))//通过循环读取数据内容
{
// 定位到第一条记录
mysql_data_seek($result, 0);
// 循环取出记录
while ($row=mysql_fetch_row($result))
{
?>
<tr>
<td height="30"><div align="center"> <?php echo $row[0] ?> </div></td>
<td width="400"> <div align="center"> <?php echo $row[1] ?> </div></td>
<td><div align="center"><a href="del.php?newsid=<?php echo $row[0] ?> " > 删除
</a>
</div></td>
</tr>
<?php
}
}
?>
</table></td>
</tr>
</table>
</div>
</body>
</html>
<?php
// 释放资源
mysql_free_result($result);
// 关闭连接
mysql_close($conn);
?>

```

在页面的主体部分，首先创建了一个用于添加新闻的表单，该表单包含两个输入字段（新闻标题和新闻内容）和一个提交按钮。当用户填写新闻标题和内容后，点击“添加”按钮，将通过 POST 方法将数据发送到 “add.php” 文件进行处理。之后，页面上有一个表格用于显示新闻列表。这个列表从数据库的 “news” 表中获取所有新闻记录，并将每一条新闻的序号，标题和一个删除链接显示出来。每个新闻的删除链接会指向 “del.php” 文件，并带上新闻的 id 作为参数。这样我们就完成了该部分框架的构建。效果图如下所示：

新闻标题:	<input type="text"/>
新闻内容:	<div style="height: 100px; border: 1px solid #ccc;"></div>
<input type="button" value="添加"/>	

---

**Warning:** mysql\_db\_query(): supplied argument is not a valid MySQL-Link resource in C:\Dreamweaver8-chs\PHPnow-1.5.6  
\htdocs\phpMyAdmin\sys.php on line 38

**Warning:** mysql\_fetch\_array(): supplied argument is not a valid MySQL result resource in C:\Dreamweaver8-chs\PHPnow-1.5.6  
\htdocs\phpMyAdmin\sys.php on line 39

新闻序号	新闻标题	删除
------	------	----

## 2.5 编写 add.php 文件

该文件用于对从 sys.php 文件传来的新闻标题以及内容进行添加，添加的位置是我们本地数据库 TestDB 中，添加之后可以在我们的 sys.php 界面显示。源代码如下所示：

```

<?php
$conn=mysql_connect("localhost", "root", "123456");
mysql_select_db("TestDB");
$topic = $_POST['topic'];
$content = $_POST['content'];
$SQLStr = "insert into news(topic, content) values('$topic', '$content')";
echo $SQLStr;
$result=mysql_query($SQLStr);

// 关闭连接
mysql_close($conn);
if ($result)
{
?
<script>
alert("insert succes");
window.location.href="sys.php";
</script>
<?php
}
else{
?
<script>
alert("insert failed");
history.back();
</script>
<?php
}

?>

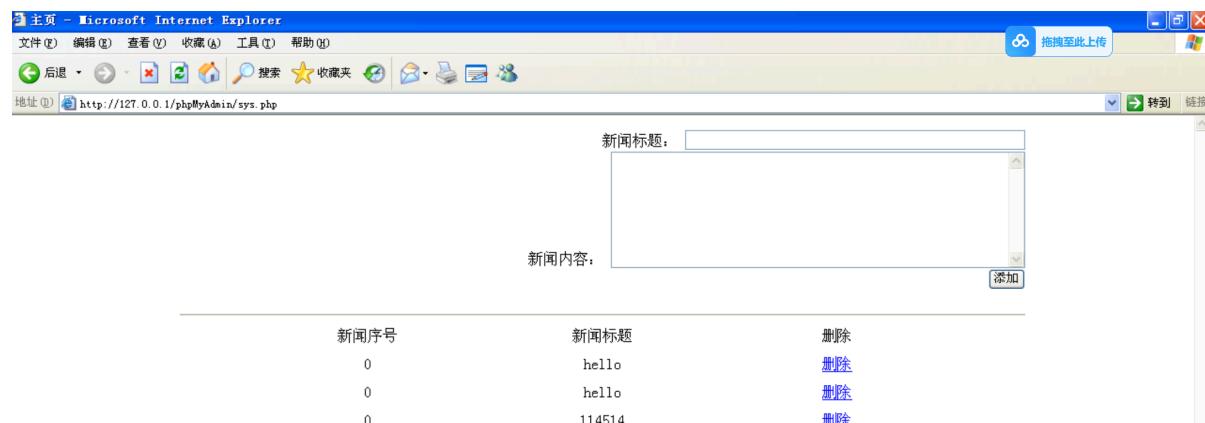
```

首先，代码连接到本地的 MySQL 数据库，并选择 "TestDB" 数据库。然后，从 POST 请求中获取 'topic' 和 'content' 字段的值，这些值是用户在 HTML 表单中提交的新闻标题和内容；根据 SQL 语句的执行结果决定下一步行动。如果 SQL 语句执行成功，代码将弹出一个显示 "insert success" 的提示框，然后页面会跳转到 "sys.php"。如果 SQL 语句执行失败，代码将弹出一个显示 "insert failed" 的提示框，并让页面返回到前一个页面。在 add.php 中数据库访问操作是通过 `mysql_select_db("TestDB")` 连接数据库之后，使用 `$result=mysql_query($SQLstr)` 进行更新和删除语句的执行，可见 `mysql_query` 比较适合更新和删除语句，其返回值是 `boolean`，可以校验执行成功与失败。

我们在其中插入一条信息，观察是否正常存入 sys.php 中。



输入信息后，我们点击添加，在 sys.php 界面就可以发现成功的插入了该条信息。

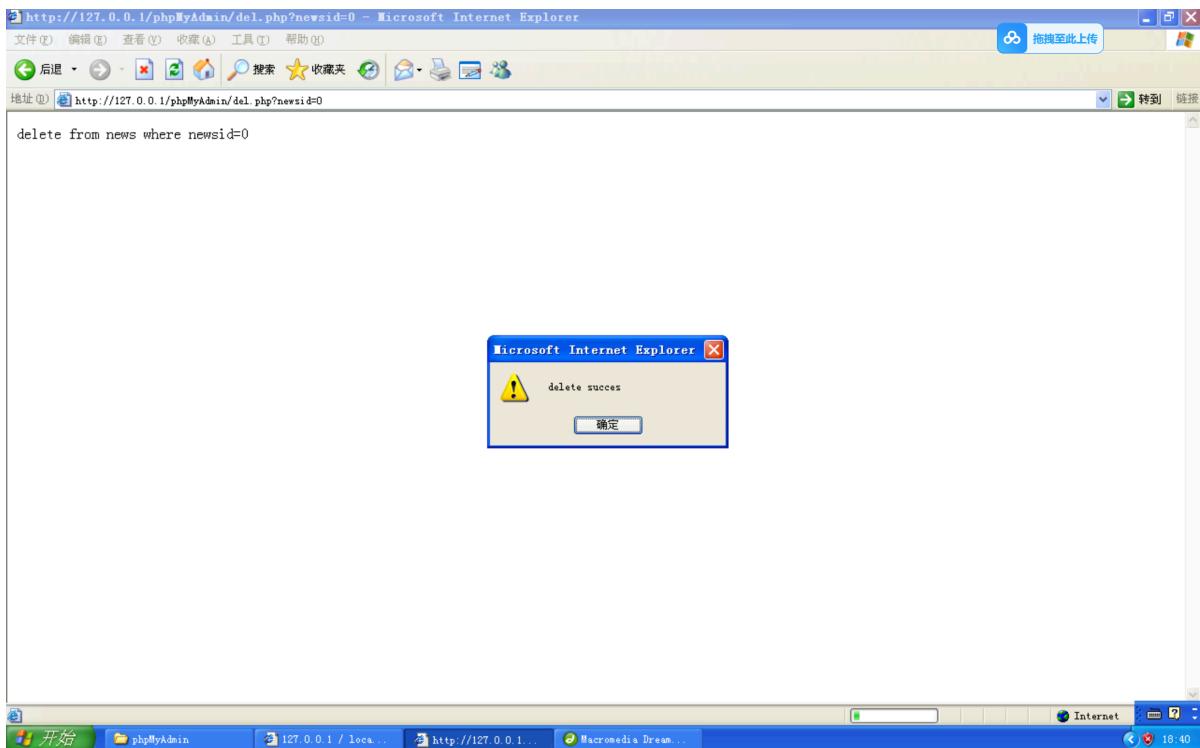


## 2.6 编写 del.php 文件

该文件用于对在 sys.php 界面中点击“删除”的新闻进行删除，从 TestDB 数据库中将其删除。源代码如下：

```
<?php
$conn=mysql_connect("localhost", "root", "123456");
mysql_select_db("TestDB");
$newsid = $_GET['newsid'];
$SQLStr = "delete from news where newsid=$newsid";
echo $SQLStr;
$result=mysql_query($SQLStr);
// 关闭连接
mysql_close($conn);
if ($result)
{
?>
<script>
alert("delete succes");
window.location.href="sys.php";
</script>
<?php
}
else{
?>
<script>
alert("delete failed");
history.back();
</script>
<?php
}
?>
```

删除的原理与添加的原理基本类似，获取从 sys.php 界面传来的 newsid，然后进行删除处理，我们进行测试，得出结果。



我们对现有的信息进行删除操作，删除第二条信息，就可以发现删除成功！

## 2.7 编写 index.php 文件

该文件主要是实现让登录者可以查看当前已经有的新闻，同时可以点击连接去查看当前已经有的新闻的内容。源代码如下所示：

```

<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>主页</title>
</head>
<?php
$conn=mysql_connect("localhost", "root", "123456");

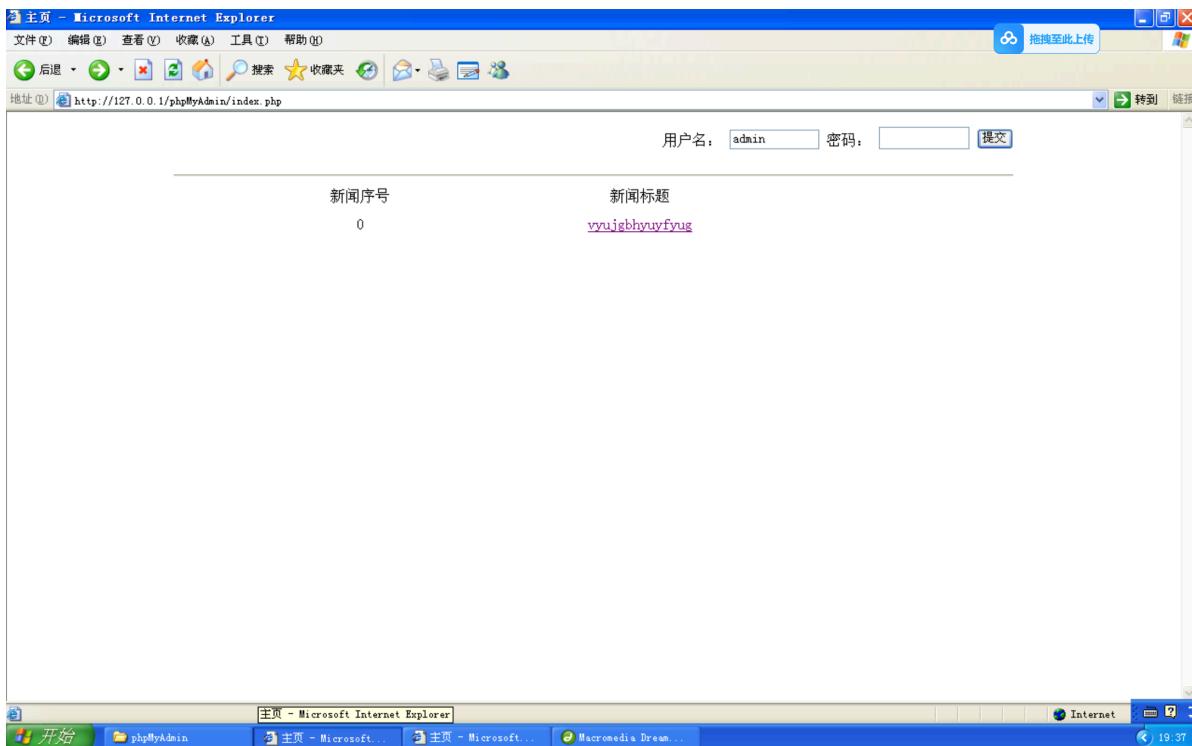
```

```
?>
<body>
<div align="center">
<table width="900" border="0" cellspacing="0" cellpadding="0">
<tr>
<td height="40"><form id="form1" name="form1" method="post" action="loginok.php">
<div align="right">用户名:
<input name="username" type="text" id="username" size="12" /> 密码:
<input name="pwd" type="password" id="pwd" size="12" />
<input type="submit" name="Submit" value="提交" />
</div>
</form>
</td>
</tr>
<tr>
<td><hr /></td>
</tr>
<tr>
<td height="300" align="center" valign="top"><table width="600" border="0"
cellspacing="0" cellpadding="0">
<tr>
<td width="100" height="30"><div align="center">新闻序号</div></td>
<td><div align="center">新闻标题</div></td>
</tr>
<?php
$SQLStr = "select * from news";
$result=mysql_db_query("testDB", $SQLStr, $conn);
if ($row=mysql_fetch_array($result))//通过循环读取数据内容
{
// 定位到第一条记录
mysql_data_seek($result, 0);
// 循环取出记录
while ($row=mysql_fetch_row($result))
{
?>
<tr>
<td height="30"><div align="center"> <?php echo $row[0] ?> </div></td>
<td> <div align="center"> <a href="news.php?newsid=<?php echo $row[0] ?> " > <?
php echo
$row[1] ?> </a> </div></td>
</tr>
<?php
}
}
?>
</table></td>
</tr>
</table>
</div>
</body>
</html>
<?php
// 释放资源
mysql_free_result($result);
// 关闭连接
mysql_close($conn);
```

```
?>
```

在该部分代码中，我们通过一个**循环**来显示当前存在的所有新闻的序号以及标题，同时对每个连接的标题我们设置了一个链接，点击该链接就可以查看新闻的内容了。

我们测试一下效果，发现在登录后确实可以看到所有的新闻消息。



## 2.8 编写 news.php 文件

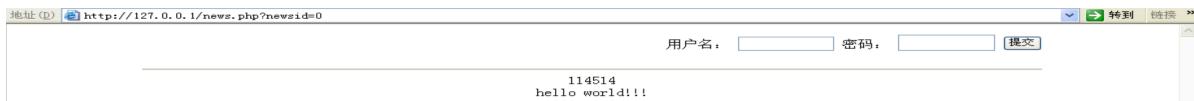
该文件的目的就是在 index.php 中点击新闻标题后能够显示出来新闻的内容。源代码如下所示：

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>主页</title>
</head>
<body>
<div align="center">
<table width="900" border="0" cellspacing="0" cellpadding="0">
<tr>
<td height="40"><form id="form1" name="form1" method="post" action="loginok.php">
<div align="right">用户名:<br/>
<input name="username" type="text" id="username" size="12" /> 密码:<br/>
<input name="password" type="password" id="password" size="12" />
<input type="submit" name="Submit" value="提交" />
</div>
</form>
</td>
</tr>
<tr>
<td><hr /></td>
</tr>
<tr>
<td>
```

```

<td height="300" align="center" valign="top"><p>&ampnbsp</p>
<?php
$conn=mysql_connect("localhost", "root", "123456");
$newsid = $_GET['newsid'];
$SQLStr = "select * from news where newsid=$newsid";
$result=mysql_db_query("testDB", $SQLStr, $conn);
if ($row=mysql_fetch_array($result))//通过循环读取数据内容
{
// 定位到第一条记录
mysql_data_seek($result, 0);
// 循环取出记录
while ($row=mysql_fetch_row($result))
{
echo "$row[1]<br>"; echo "$row[2]<br>";
}
}
// 释放资源
mysql_free_result($result);
// 关闭连接
mysql_close($conn);
?>
</td>
</tr>
</table>
</div>
</body>
</html>

```



同样，我们进行了测试，发现输出正确，所以我们的WEB开发就到此全部完成了！！！

## 心得体会：

在本次实验中，第一次接触 `php` 和 `html` 语言，我通过实验，对这两种语言有了更深的了解。我学会了如何创建自己的一个网站，学会了如何使用 `php` 语言与数据库的表进行连接，学会了如何创建自己的 `web` 开发，学习了一些常见的开发方法，受益匪浅。

通过此次实验，我还更好地理解了前端与后端的交互过程。我学会了如何在前端获取用户的输入，然后通过 `POST` 或 `GET` 方法发送到后端。在后端，我学会了如何接收这些输入，处理它们，并将结果发回前端。