**Creating a Web Application in Linux with Apache Web Server, PHP and MySQL**

(<http://php.net/manual/en/book.ibm-db2.php>)

PHP is designed to work with MySQL to maintain and retrieve data stored in a MySQL database.

The application created in this example creates a database with MySQL and uses PHP to manage login information and mailing address information for a mail carrier. The structure of the three tables in the database is included below. Based on a user’s access level, users can add, edit, and delete information about addresses, users, and provinces and states. Once logged in, the major interface appears as below.

**To create a database in MYSQL in Linux:**

Type mysql –u *root*

Press enter

You are now in the mysql environment. To leave this environment, type exit and press enter. Note that when you exit mysql, you are still in linux.

To create the database called mailman, the SQL code on the next two pages was copied from the file mailman.txt and then pasted into mysql. When it was pasted into mysql, it ran and created the database mailman with three tables and inserted data into the three tables. The three tables are users, address, and prov.

**Table Structures:**

**Users table structure**

|  |  |  |
| --- | --- | --- |
| Field name | Data type | Columns size |
| Userid | int |  |
| firstname | varchar | 25 |
| lastname | varchar | 30 |
| username | varchar | 15 |
| password | varchar | 10 |
| accesslevel | int |  |
| Phone | varchar | 14 |
| Email | varchar | 35 |

**Address table structure**

|  |  |  |
| --- | --- | --- |
| Field name | Data type | Columns size |
| addressid | int |  |
| firstname | varchar | 25 |
| lastname | varchar | 30 |
| Address1 | varchar | 35 |
| Address2 | varchar | 35 |
| City | varchar | 35 |
| Prov | char | 2 |
| Postal | varchar | 12 |
| Phone | varchar | 14 |
| Email | varchar | 20 |

**Prov table structure**

|  |  |  |
| --- | --- | --- |
| Field name | Data type | Columns size |
| Provid | int |  |
| provcode | char | 2 |
| provname | varchar | 35 |

**Selected contents of file named mailman.txt (used to create databases and tables):**

All the tables created will be in the root directory. Thus, a designation that uniquely identifies each database is needed. Thus, put your userid as a prefix to the database name of mailman. For example, ***uoas001mailman***. The rest of this documentation uses the short name of mailman with the prefix; however note that you will need to change it throughout the remainder of the document—including code as all the examples use just the name ***mailman***.

To create a database after getting into mysql, use the following—***these commands are in mailman.txt but you must change the database name***. These statements, after changing the database name (it is useridmainlman in the document), you can paste the code into mysql and it will create three tables and populate them with data.

CREATE DATABASE uoasxxmailman;

USE uoasxxxmailman;

**CREATE TABLE address**

 (addressid int not null primary key,

 firstname varchar(25),

 lastname varchar(30),

 address1 varchar(35),

 address2 varchar(35),

 city varchar(25),

 prov char(2),

 postal varchar(12),

 phone varchar(14),

 email varchar(20)

 );

INSERT INTO address

 (addressid, firstname, lastname, address1, address2, city, prov, postal, phone, email)

VALUES

 (1, 'Peter', 'MacIntyre', 'PO Box 2076', 'Station A', 'Stratford', 'PE',

 'C1A 7J6', '902-555-1234', 'peter\_maci@anymail.com') ;

.

.

INSERT INTO address

 (addressid, firstname, lastname, address1, address2, city, prov, postal, phone, email)

VALUES (5, 'Simon', 'MacIntyre', 'PO Box 2076', 'Station A', 'Stratford', 'PE',

 'C1A 7J6', '902-555-1114', 'simonsoccker@anymail.com') ;

**CREATE TABLE users (userid int not null primary key,**

firstname varchar(25),

lastname varchar(30),

username varchar(15),

password varchar(10),

accesslevel int not null,

phone varchar(14),

email varchar(35) );

INSERT INTO users (userid, firstname, lastname, username, password, accesslevel, phone, email)

Values (1, 'administrator', 'administrator', 'admin', 'entry', 1, '555-1234', ‘admin@admin.com');

INSERT INTO users (userid, firstname, lastname, username, password, accesslevel, phone, email)

Values (2, 'Peter', 'MacIntyre', 'pbmacintyre', 'happy', 1, '555-1234', 'peter\_maci@hotmail.com') ;

INSERT INTO users (userid, firstname, lastname, username, password, accesslevel, phone, email)

Values (3, 'Fred', 'Flintstone', 'fredflint', 'pebbles', 2, '111-1234', 'fred@bedrock.com') ;

**CREATE TABLE prov (provid int not null primary key,**

provcode char(2),

provname varchar(35) );

INSERT INTO prov (provid, provcode, provname) values(1, 'PE', 'Prince Edward Island');

INSERT INTO prov (provid, provcode, provname) values(2, 'NF', 'Newfoundland');

.

.

INSERT INTO prov (provid, provcode, provname) values(16, 'NV', 'Nevada');

INSERT INTO prov (provid, provcode, provname) values(17, 'NM', 'New Mexico');

**Using MySQL in Linux: The bold and underlined statements were typed.**

uoasxxx@linux01:~>**mysql -u root**

The mysql response is below:

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 121 to server version: 5.0.26

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>

To list the databases, use the **show databases** statement:

mysql> **show databases;**

+----------+

| Database |

+----------+

| mailman |

| mysql |

| test |

| test2 |

+----------+

4 rows in set (0.00 sec)

Before you can access a database you must invoke it via the **use** command

mysql> **use uoas3xxmailman;**

The mysql response is:

Reading table information for completion of table and column names

You can turn off this feature to get a quicker startup with -A

Database changed

After gaining access to the database, you can list the database tables using the **show tables** command.

mysql> **show tables;**

+-------------------+

| Tables\_in\_uoas3xxmailman |

+-------------------+

| address |

| prov |

| users |

+-------------------+

3 rows in set (0.00 sec)

You can use the SQL select statement to list the rows in each table in the database as shown below.

mysql> **select \* from address;**

+-----------+-----------+------------+-------------------+-----------+-----------+------+---------+--------------+----------------------+

| addressid | firstname | lastname | address1 | address2 | city | prov | postal | phone | email |

+-----------+-----------+------------+-------------------+-----------+-----------+------+---------+--------------+----------------------+

| 1 | Peter | MacIntyre | PO Box 2076 | Station A | Stratford | PE | C1A 7J6 | 902-555-1234 | peter\_maci@anymail.c |

| 2 | Dawn | Riley | 31 Marjorie Cres. | | Stratford | PE | C1B 1X4 | 902-555-1236 | testmail@anymail.com |

| 3 | Fred | Flintstone | PO Box 2076 | Station F | Bedrock | PE | C1A 7J6 | 902-555-4234 | fflint@anymail.com |

| 4 | Frodo | Baggins | PO Box 2276 | Station A | The Shire | ME | C1A 7J6 | 902-556-1134 | thering@anymail.com |

| 5 | Simon | MacIntyre | PO Box 2076 | Station A | Stratford | PE | C1A 7J6 | 902-555-1114 | simonsoccker@anymail |

+-----------+-----------+------------+-------------------+-----------+-----------+------+---------+--------------+----------------------+

5 rows in set (0.00 sec)

mysql> **select \* from prov;**

+--------+----------+----------------------+

| provid | provcode | provname |

+--------+----------+----------------------+

| 1 | PE | Prince Edward Island |

| 2 | NF | Newfoundland |

| 3 | NS | Nova Scotia |

| 4 | NB | New Brunswick |

| 5 | PQ | Quebec |

| 6 | ON | Ontario |

| 7 | MB | Manitoba |

| 8 | AB | Alberta |

| 9 | SK | Saskatechwan |

| 10 | BC | British Columbia |

| 11 | ME | Maine |

| 12 | NH | New Hampshire |

| 13 | VT | Vermont |

| 14 | NY | New York |

| 15 | CO | Colorado |

| 16 | NV | Nevada |

| 17 | NM | New Mexico |

+--------+----------+----------------------+

17 rows in set (0.00 sec)

mysql> **select \* from users;**

+--------+---------------+---------------+-------------+----------+-------------+----------+------------------------+

| userid | firstname | lastname | username | password | accesslevel | phone | email |

+--------+---------------+---------------+-------------+----------+-------------+----------+------------------------+

| 1 | administrator | administrator | admin | entry | 1 | 555-1234 | admin@admin.com |

| 2 | Peter | MacIntyre | pbmacintyre | happy | 1 | 555-1234 | peter\_maci@hotmail.com |

| 3 | Fred | Flintstone | fredflint | pebbles | 2 | 111-1234 | fred@bedrock.com |

+--------+---------------+---------------+-------------+----------+-------------+----------+------------------------+

3 rows in set (0.00 sec)

Use **exit** to exit mysql and return to Linux.

mysql> **exit;**

Bye

uoasxxx@linux01:~>

Now that we have a MySQL database created, the next step is to create an application that interacts with the database and manipulates data from the database. The application manages a user list and is scheduled to be expanded in the future to perform basic mailing list tasks such as recording mailing addresses.

**Design of the MailMan system:**

Three major components: Security logon, system maintenance, and mailing address management. Note the mailing address management section is not addressed in this handout.

Security logon – the gateway to the whole system. This component grants or denies system access. A typical logon screen that asks for the username and password appears. The system verifies the data entered against information stored in the database. If the user passes this security check, the other parts of the application become accessible.



Systems maintenance – Keeps the applications that support the data in the database up to date. You typically need to add information, edit existing information, or delete information. The data segments that users can change are addresses, lists of states and provinces and the user logon data itself.

Only Level 1 access can add users, edit users, and delete users. They can also perform the functions of a level 2 or level 3 user.

Level 2 access can add, edit, or delete address information and state/prov information.

Level 3 access can only add, edit, or delete state/prov information.

The diagram on the next page shows the layout of the mailing list application.

**Index of PHP and HTML files**

mailmandatabase.txt – was used to create the database and three tables

mailman.html – the main menu. Uses a form - the action is trylogon.php

trylogon.php – the action in mailman.html. If the logon fails, failedlogon.html displays a message. If the logon succeeds, mailman\_main.php is used.

failedlogon.html – displays message if logon fails.

mailman\_main.php – displays a menu of actions to perform if logon succeeds (shown below)

mailman\_header.inc – included in the mailman\_main.php file



**The links for the address information and provinces/states management are slated for future development.**

The links for the system user information (add system user, edit system user and delete system user) are working and included in this document.

* adduser.php- uses a form – action is saveuser.php
* search\_edit\_user.php:

includes mailman\_header.inc

 once the user is found in the table, uses edit\_user.php.

edit\_user.php includes mailman\_header.inc

edit\_user.php uses a form – action is commit\_edit\_user.php

 commit\_edit-user.php returns user to mailman\_main.php

* search\_delete\_user.php –

includes mailman\_header.inc

once the user is found in the tables, uses delete\_user.php

 delete\_user.php includes mailman\_header.inc.

 delete\_user.php uses a form – action is commit\_delete\_user.php

 commit\_delete\_user.php returns user to mailman\_main.php

This file contains the header and is used in several other files to maintain consistency.

**mailman\_header.inc**

<html>

<head>

 <title>Mailman Main Window</title>

</head>

<body bgcolor="silver">

<table cellSpacing=1 cellPadding=1 align=center border=0>

<tr>

<td>

 <h2 align=center>MailMan Mailing List </h2></td>

</tr>

<tr>

<td>

 <h2 align=center>Web Application</h2> </td>

</tr></table>

**Creating and Using a Logon Window**



**mailman.html – uses a form. The statement in bold below contains the action. The code for the action follows this code.**

<html>

 <head>

 <title> mailman logon window </title>

 </head>

 <body bgcolor="silver">

 <table cellspacing=1 cellpadding=1 align="center">

 <tr>

 <td>

 <p align="center"> welcome to the <br>&nbsp;</p>

 </td>

 </tr>

 <tr>

 <td>

 <h2 align="center">mailman mailing list</h2>

 </td>

 </tr>

 <tr>

 <td>

 <p align="center"> web application </p> </td>

 </tr>

 </table>

 <h4><center>

 please provide the requested information:</center></h4>

 <form action="trylogon.php" method="post">

 <table border=1 align="center" cellspacing=2 cellpadding=6>

 <tr>

 <td>enter user name:</td>

 <td><input size=15 name="username"></td>

 </tr>

 <tr>

 <td>enter password:</td>

 <td><input type="password" size=15 name="password"></td>

 </tr>

 <tr>

 <td> <p align="center"><input type="submit" value="login"

 name="submit"></p></td>

 <td> <p align="center"><input type="reset" value="clear"></p></td>

 </tr>

 </table>

 </form>

 </body>

</html>

**trylogon.php –** this is the action used in the form in the mailman.html file above. Lines 2-4 below are used to setup the database connection and are discussed on the next page.

If the connection to the database is successful, then the program checks to see if the username and password are in the database in the lines that follow. The $sql variable is given a string value of the SQL command that is passed to the MySQL engine once the valid connection is obtained. The program then uses the mysql\_query() function to execute that command from within PHP, and stores the result of that command in the variable called $result. The $result variable holds a result set of all the rows that were returned from the execution of the SQL command. If there are unique users in the database, $result should only have one row in it. Upon failure, it should contain no values. The if statement is included to determine if $result has data. If it is empty, then the number of rows will be zero and the failed logon message will appear. If it is not empty, a row matching the username and password was found and mailman\_main.php will appear.

**trylogon.php:**

**<?**

**$connection = mysql\_connect("localhost","root","");**

**$db = "uoasxxxmailman";**

**mysql\_select\_db($db, $connection) or die( "Could not open $db");**

**$sql = "Select \* from users where username = '$\_POST[username]' and password=**

**'$\_POST[password]'";**

**$result = mysql\_query($sql, $connection) or die( "Could not execute sql:**

**$sql");**

**$num\_rows = mysql\_num\_rows($result);**

**if ( $num\_rows > 0 ) {**

 **header( "Location: mailman\_main.php");**

 **} else {**

 **header( "Location: failedlogon.php" );**

 **}**

**?>**

**Failedlogon.html – referred to in trylogon.php. If the login and password are not in the table, then this file is displayed.**

<html>

<head>

 <title>Mailman Failed Login</title>

</head>

<body>

<h3> Sorry your login information was invalid ... please try again...</h3>

</body>

</html>



**Setting up Database Connections**

To use PHP with the MySQL mailman database, you need to connect a PHP script to the database. When users attempt to logon, for example, you tell PHP to connect to the mailman database and check to see if the username and password the user entered matches data in the user table. To interact with a database, you must first tell PHP to connect to the database engine itself. If you are connecting to a MySQL database, you use the PHP mysql\_connect function for this purpose. The syntax for this function is:

mysql\_connect(“localhost”, “root”, “password”)

The connection function returns a value that holds the connection object so that you can use it again for other database actions. A connection object is a memory handle that PHP can use to communicate with the database in subsequent accesses.

After you gain access to a database engine using the PHP mysql\_connect command to store the connection object’s handle in the $connection variable, you need to establish a connection to a specific database. The database we want to connect to is named uoasxxmailman. The PHP statement used is called mysql\_select\_db. The database name is first stored in a variable ($db in our example)

The syntax used in our programs is as follows:

$connection = mysql\_connect("localhost","root","");

$db = "uoasxxxmailman";

mysql\_select\_db($db, $connection) or die( "Could not open $db");

The mysql\_connect statement connects to the database engine.

The mysql\_select\_db statement connects to the specific database.

Once a user has supplied a valid logon and password in the screen above, mailman\_main.php becomes the header and the following menu appears:



mailman\_main.php – the PHP files in bold below exist; the other PHP files are slated for future developed. Notice the syntax of the lines in bold. This is the hyperlink with an embedded value that we have seen before.

<? include("mailman\_header.inc"); ?>

<h4><center>

Please select the process that you want to

perform</center></h4>

<h4>

<center>

<table border=1 align=center cellSpacing=2 cellPadding=6>

 <tr>

 <td>

 <h3 align=center>Address <br>Information</h3></td>

 <td>

 <h3 align=center>System User <br>Information</h3></td>

 <td>

 <h3 align=center>Provinces / States <br>Management</h3></td>

 </tr>

 <tr>

 <td><a href="addaddress.php">

 <p align=center>Add Addresses</a></p>

 </td>

 <td><a href="adduser.php">

 <p align=center>Add System User</a></p></td>

 <td><a href="addprov.php">

 <p align=center>Add Province / State</a></p></td>

 </tr>

 <tr>

 <td><a href="search\_edit\_address.php">

 <p align=center>Edit Addresses</a></p></td>

 <td><a href="search\_edit\_user.php">

 <p align=center>Edit System User</a></p></td>

 <td><a href="search\_edit\_prov.php">

 <p align=center>Edit Province / State</a></p></td>

 </tr>

 <tr>

 <td><a href="search\_delete\_address.php">

 <p align=center>Delete Addresses</a></p></td>

 <td><a href="search\_delete\_user.php">

 <p align=center>Delete System User</a></p></td>

 <td><a href="search\_delete\_prov.php">

 <p align=center>Delete&nbsp;Province / State</a></p></td>

 </tr>

</table>

</center></h4>

</body>

</html>

**Adding a user:**



**adduser.php**

<? include ("mailman\_header.inc"); ?>

<form action=saveuser.php method=post>

<table cellSpacing=2 cellPadding=6 align=center border=1>

 <tr>

 <td colSpan=4>

 <h3 align=center>Add System User&nbsp;Information</h3></td></tr>

 <tr>

 <td>First Name</td>

 <td><input name=fname></td>

 <td>Last Name</td>

 <td><input name=lname></td></tr>

 <tr>

 <td>User Name</td>

 <td><input name=username></td>

 <td> Password</td>

 <td><input name=password></td></tr>

 <tr>

 <td>Access Level</td>

 <td><select name=accesslevel>

 <option value=0 selected>select level

 <option value=1>1

 <option value=2>2

 <option value=3>3

 </option>

 </select></td>

 <td>Phone #</td>

 <td><input name=phone></td>

 </tr>

 <tr>

 <td>E-mail</td>

 <td><Input name=email></td>

 <td><Input type=submit value=Save></td>

 <td><input type=reset value=Reset></td>

 </tr>

</table>

</form>

</body>

</html>

**Saveuser.php:**

<?

$connection = mysql\_connect ("localhost","root","");

$db = "uoasxxxmailman";

mysql\_select\_db ($db, $connection ) or die ( "Could not open $db");

$sql = "select max(userid) as currentid from users";

$result = mysql\_query($sql, $connection) or die ( "Could not execute

sql:$sql");

$row = mysql\_fetch\_array($result);

$nextid = $row["currentid"];

$nextid++;

$sql = "INSERT INTO users (userid, firstname, lastname, username, password,

accesslevel, phone, email) ";

$sql = $sql . "VALUES ($nextid, '$\_POST[fname]', '$\_POST[lname]', '$\_POST[username]', '$\_POST[password]', $\_POST[accesslevel],";

$sql = $sql . "'$\_POST[phone]', '$\_POST[email]')";

$res = mysql\_query($sql, $connection) or die( "Could not execute sql:

$nextid: $sql");

if ( !$res ) {

 echo "problem adding to database";

 } else {

 header( "Location: mailman\_main.php");

 }

?>

**Editing a user:**

 

**Search\_edit\_user.php**

<?

include("mailman\_header.inc");

$connection = mysql\_connect("localhost","root","");

$db = "uoasxxxmailman";

mysql\_select\_db($db, $connection) or die( "Could not open $db");

$sql = "select \* from users";

$result = mysql\_query($sql, $connection) or die( "Could not execute sql:

$sql");

$num\_result = mysql\_num\_rows($result);

?>

<Table cellSpacing=2 cellPadding=6 align=center border=1>

<tr>

<td colspan=5>

<h3 align=center>Click on the user

record<br>&nbsp;you want to edit</h3></td></tr>

<tr>

<td> First Name </td>

<td> Last Name </td>

<td> User Name </td>

<td> Phone # </td>

<td> E-mail </td>

</tr>

<?

for ($i=0; $i < $num\_result; $i++) {

 $row = mysql\_fetch\_array($result);

 $id = $row["userid"];

 echo "<tr><td>";

 echo "<a href=\"edit\_user.php?userid=$id\">";

 echo $row ["firstname"];

 echo "</a></td><td>";

 echo $row ["lastname"];

 echo "</td><td>";

 echo $row ["username"];

 echo "</td><td>";

 echo $row ["phone"];

 echo "</td><td>";

 echo $row ["email"];

 echo "</td></tr>";

}

?>

</tr>

</table>

</body>

</html>

**Edit\_user.php**

<? include("mailman\_header.inc"); ?>

<?

$connection = mysql\_connect("localhost","root","");

$db = "uoasxxxmailman";

mysql\_select\_db($db, $connection) or die( "Could not open $db");

$sql = "select \* from users where userid = '$\_GET[userid]'";

$result = mysql\_query($sql, $connection) or die( "Could not execute sql:

$sql");

$row = mysql\_fetch\_array($result);

?>

<form action=commit\_edit\_user.php method=post>

<input type=hidden name=userid value="<?= $\_GET[userid] ?>" >

<table cellSpacing=2 cellPadding=6 align=center border=1>

<tr>

 <td colSpan=4>

 <h3 align=center>Update System User Information</h3></td></tr>

<tr>

 <td>First Name</td>

 <td><INPUT name=fname value="<?= $row["firstname"] ?>" ></td>

 <td> Last Name</td>

 <td><input name=lname value="<?= $row["lastname"] ?>"></td></tr>

<tr>

 <td>User Name</tr>

 <td><INPUT name=username value="<?= $row["username"] ?>"></td>

 <td> Password</td>

 <td><input name=password value="<?= $row["password"] ?>"></td></tr>

<tr>

 <td>Access Level</td>

 <TD>

<select name=accesslevel>

<? switch ($row["accesslevel"]) {

 case 1: ?>

 <option value = 1 selected>1

 <option value = 2>2

 <option value = 3>3

<? break;

 case 2: ?>

 <option value = 1>1

 <option value = 2 selected >2

 <option value = 3>3

<? break;

 case 3: ?>

 <option value = 1>1

 <option value = 2>2

 <option value = 3 selected>3

 <? } ?>

 </option>

 </select></td>

 <td>phone #</td>

 <td><input name=phone value="<?= $row["phone"] ?>"></td>

</tr>

<tr>

 <td>e-mail</td>

 <td><input name=email value="<?= $row["email"] ?>"></td>

 <td><input type=submit value=save></td>

 <td><input type=reset value=reset></td>

</tr>

</table>

</form>

</body>

</html>

**Commit\_edit\_user.php**

<?

$connection = mysql\_connect("localhost","root","");

$db = "uoasxxxmailman";

mysql\_select\_db($db, $connection) or die( "Could not open $db");

$sql = "UPDATE users SET firstname = '$\_POST[fname]', lastname = '$\_POST[lname]',

username = '$\_POST[username]',";

$sql = $sql . "password = '$\_POST[password]', accesslevel = $\_POST[accesslevel], phone =

'$\_POST[phone]', ";

$sql = $sql . "email = '$\_POST[email]' WHERE userid = $\_POST[userid]";

echo $sql;

$res = mysql\_query($sql, $connection) or die( "Could not execute sql: $nextid: $sql");

if( !$res) {

 echo "problem updating dtabase";

 } else {

 header( "Location: mailman\_main.php");

 }

 ?>

**Deleting users:**

****

**search\_delete\_user.php:**

<?

include("mailman\_header.inc");

$connection = mysql\_connect("localhost","root","");

$db = "uoasxxxmailman";

mysql\_select\_db($db, $connection) or die( "Could not open $db");

$sql = "SELECT \* FROM users";

$result = mysql\_query($sql, $connection) or die( "Could not execut sql: $sql");

$num\_result = mysql\_num\_rows($result);

?>

<table cellSpacing=2 cellPadding=6 align=center border=1>

 <tr>

 <td colspan=5>

 <h3 align=center>Click on the user

 record<BR>&nbsp;you want to delete</h3></td></tr>

 <tr>

 <td> First Name </td>

 <td> Last Name </td>

 <td> User Name </td>

 <td> Phone # </td>

 <td> E-Mail </td>

 </tr>

<?

 for ($i=0; $i < $num\_result; $i++) {

 $row = mysql\_fetch\_array($result);

 $id = $row["userid"];

 echo "<tr><td>";

 echo "<a href=\"delete\_user.php?userid=$id\">";

 echo $row["firstname"];

 echo "</a></td><td>";

 echo $row["lastname"];

 echo "</td><td>";

 echo $row["username"];

 echo "</td><td>";

 echo $row["phone"];

 echo "</td><td>";

 echo $row["email"];

 echo "</td></tr>";

 }

?>

</tr>

</table>

</body>

</html>

**Delete\_user.php**

<? include("mailman\_header.inc"); ?>

<?

$connection = mysql\_connect("localhost","root","");

$db = "uoasxxxmailman";

mysql\_select\_db($db, $connection) or die( "Could not open $db");

$sql = "select \* from users where userid= $\_GET[userid]";

$result = mysql\_query($sql, $connection) or die( "Could not execute sql:

$sql");

$row = mysql\_fetch\_array($result);

?>

<form action=commit\_delete\_user.php method=post>

<input type=hidden name=userid value="<?= $\_GET[userid] ?>" >

<table cellSpacing=2 cellPadding=6 align=center border=1>

<tr>

 <td colSpan=4>

 <h3 align=center>Confirm Deletion of System User

 Information</h3></td></tr>

<tr>

 <td>First Name</td>

 <td> <?= $row["firstname"] ?> </td>

 <td>Last Name</td>

 <td> <?= $row["lastname"] ?> </td></tr>

<tr>

 <td>User Name</td>

 <td> <?= $row["username"] ?> </td>

 <td> Password</td>

 <td> <?= $row["password"] ?> </td></tr>

<tr>

 <td>Access Level</td>

 <td>

 <? switch ($row["accesslevel"]) {

 case 1: ?>

 1

 <? break;

 case 2: ?>

 2

 <? break;

 case 3: ?>

 3

 <? } ?>

</td>

<td>Phone #</td>

<td> <?= $row["phone"] ?> </td></tr>

<tr>

 <td>E-Mail</td>

 <td> <?= $row["email"] ?> </td>

 <td><input type=submit value=Delete></td>

 <td><input type=reset value=Reset></td>

</tr>

</table>

</form>

</body>

</html>

**Commit\_delete\_user.php**

<?

$connection = mysql\_connect("localhost","root","");

$db = "uoasxxxmailman";

mysql\_select\_db($db, $connection) or die( "Could not open $db");

$sql = "DELETE FROM users WHERE userid = '$\_POST[userid]' ";

$res = mysql\_query($sql, $connection) or die( "Could not execut sql:

$nextid: $sql");

if ( !$res ) {

 echo "problem updating database";

 } else {

 header( "Location: mailman\_main.php" );

}

?>

**Including security**

Trylogon2.php – add the following three rows that are shown in bold. This will establish security through the use of cookies. Once the program finds the username and password, it retrieves the user’s access level from the table and stores it in the $accesslevel variable. Using the Setcookie() PHP function, the program sets a cookie that has the name of “access” and gives this cookie the retrieved value. Once the cookie is set in this code, it can be retrieved the same way an HTML variable is – you reference it by preceding its name with a dollar sign ($) character.

<?

$connection = mysql\_connect("localhost","root","");

mysql\_select\_db($db, $connection) or die( "Could not open $db");

$sql = "Select \* from users where username = '$\_POST[username]' and password=

'$\_POST[password]'";

$result = mysql\_query($sql, $connection) or die( "Could not execute sql:

$sql");

$num\_rows = mysql\_num\_rows($result);

if ( $num\_rows > 0 ) {

 **$row = mysql\_fetch\_array($result);**

 **$accesslevel = $row["accesslevel"] ;**

 **SetCookie("access", $accesslevel);**

 header( "Location: mailman\_main.php" );

} else {

 header( "Location: failedlogon.html" );

}

?>

Change mailman.html to use trylogon2.php instead of trylogon.php

Now that the access level is set for the particular session of using the Mailman application, you can check its value on every Web page within the application to ensure that the logged-on user has permission to go to certain pages. In this example, the access level of 1 is set to administration level meaning access to anywhere within the application. An access level of 3 can not add, edit, or delete users.

Look at mailman\_main.php. The files that are referenced for add, edit, and delete are adduser.php, search\_edit\_user.php, and search\_delete\_user.php. To use the cookie, the following code needs to be added to the beginning of each of these files. If the user’s access code is not 1, then they are presented with the unauthorized screen.

**<?**

**if( $access != 1 ) header ( "Location: unauthorized.html" );**

**?>**

Enter a userid in the database with a code of 3 and try to add, edit, and delete someone else.