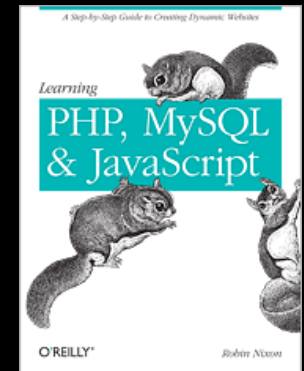


Cookies, Sessions, and Authentication

Chapter 13
Dr. Charles Severance

To be used in association with the book:
PHP, MySQL, and JavaScript by Robin Nixon



open.michigan

Unless otherwise noted, the content of this course material is licensed under a Creative Commons Attribution 3.0 License.

<http://creativecommons.org/licenses/by/3.0/>.

Copyright 2011, Charles Severance



High Level Summary

- The web is “stateless” - the browser does not maintain a connection to the server while you are looking at a page. You may never come back to the same server - or it may be a long time - or it may be one second later
- So we need a way for servers to know “which browser is this?”
 - In the browser state is stored in “Cookies”
 - In the server state is stored in “Sessions”


twitter

Home Find & Follow Public Timeline Settings Help Sign out

What are you doing?

140

Hi, [your profile](#)

 drchuck

update

Recent Replies Archive



dkeats Spending the morning with the eLearning team innovation, and participation in a community of practice Energizing. 3 minutes ago from web ☆ ↻



drchuck Working on lecture slides - talking about cook sessions. 5 minutes ago from web ☆ 🗑



MattH Ordered Urban Fortunes from Amazon because I saying it said things I wasn't at all agreeing with. Right. ago from [twiterrific](#) ☆ ↻



microcline Back from walking the dogs. Cleaned them can, but they don't get to share the futon tonight. 31 m from web ☆ ↻

flickr LOVES YOU

Signed in as dr-chuck (1 new) H

Home You Organize Contacts Groups Explore

Search everyone's photos



Ni hao dr-chuck!

Now you know how to greet people in Mandarin!

- You have [1 new message](#).
- [Find your friends](#)

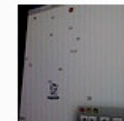
Flickr News

31 Mar 08 - Flickr is more fun with friends, but it is a big busy place, and sometimes it can be hard to find the people you know. The new Find Your... [read more news](#)

» [Flickr Blog](#) Great photos & latest news, daily!

» [Upload Photos](#) (Or, look at our uploading [tools...](#))

» [Your Photos](#) ([Recent activity](#) / [Comments you've made](#))



» [Photos from your Contacts](#)



From [Lance70](#)



From [Lance70](#)

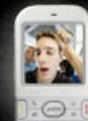


From [Lance70](#)



From [Lance70](#)

ADVERTISEMENT



Other Web sites always seem to know who you are!

Multi-User

- When a server is interacting with many different browsers at the same time, the server needs to know *which* browser a particular request came from
- Request / Response initially was stateless - all browsers looked identical - this was really really bad and did not last very long at all.

Web Cookies to the Rescue

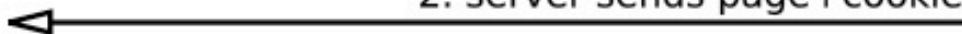
Technically, cookies are arbitrary pieces of data chosen by the Web server and sent to the browser. The browser returns them unchanged to the server, introducing a state (memory of previous events) into otherwise stateless HTTP transactions. Without cookies, each retrieval of a Web page or component of a Web page is an isolated event, mostly unrelated to all other views of the pages of the same site.

http://en.wikipedia.org/wiki/HTTP_cookie

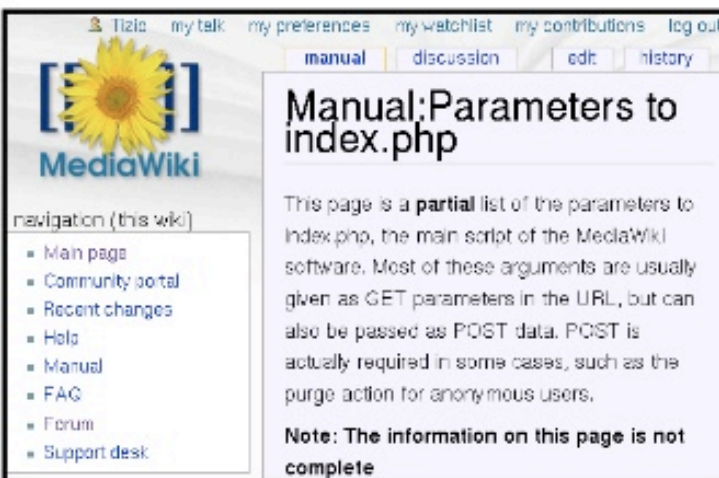
1. browser requests a Web page



2. server sends page+cookie



cookie



server

Web
browser

3. browser requests another page



cookie

http://en.wikipedia.org/wiki/HTTP_cookie

Web browser request headers

1
GET /index.html
HTTP/1.1 Host:
www.webserver.com

3
GET /news.html HTTP/1.1
Host: www.webserver.com
Cookie: name=value

Headers returned by *webserver.com*

2
HTTP/1.1 200 OK
Content-type: text/html
Set-Cookie: name=value
...Contents of web page

4
HTTP/1.1 200 OK
Content-type: text/html
...Contents of web page

Cookies In the Browser

- Cookies are marked as to the web addresses they come from - the browser only sends back cookies that were originally set by the same web server
- Cookies have an expiration date - some last for years - others are short-term and go away as soon as the browser is closed

Sessions

In The Server - Sessions

- In most server applications, as soon as we meet a new browser - we create a session
- We set a session cookie to be stored in the browser which indicates the session id in use
- The creation and destruction of sessions is handled by a web framework or some utility code that we just use to manage the sessions

Session Identifier

- A large, random number that we place in a browser cookie the first time we encounter a browser.
- This number is used to pick from the many sessions that the server has active at any one time.
- Server software stores data in the session which it wants to have from one request to another from the same browser.
- Shopping cart or login information is stored in the session in the server

PHP Sessions

PHP Sessions

- We can establish / initialize a PHP Session by calling `session_start()` before any output has come out
- If the user has cookies set, we can use the array `$_SESSION` to store data from one request to the next with a particular browser
- We have a bit of data that persists from one request to the next

session_start

(PHP 4, PHP 5)

session_start — Initialize session data

Description

[Report a bug](#)

```
bool session_start ( void )
```

session_start() creates a session or resumes the current one based on a session identifier passed via a GET or POST request, or passed via a cookie.

To use a named session, call [session_name\(\)](#) before calling **session_start()**.

When [session.use_trans_sid](#) is enabled, the **session_start()** function will register an internal output handler for URL rewriting.

If a user uses *ob_gzhandler* or similar with [ob_start\(\)](#), the function order is important for proper output. For example, *ob_gzhandler* must be registered before starting the session.

<http://php.net/manual/en/function.session-start.php>

```
<?php
    session_start();
    if ( ! isset($_SESSION['value']) ) {
        echo("<p>Session is empty</p>\n");
    } else {
        echo("<pre>\n");
        print_r($_SESSION);
        echo("<pre>\n");
    }
    $_SESSION['value'] = $_SESSION['value'] + 1;
?>
<p><a href="sess1.php">Click Me!</a></p>
```

```
http://localhost/si572/php-12/sess1.php
http://localhost/si572/php-12/sess1.php
Google

Session is empty
Click Me!
```

```
http://localhost/si572/php-12/sess1.php
http://localhost/si572/php-12/sess1.php
Google

Array
(
    [value] => 1
)
Click Me!
```

```
http://localhost/si572/php-12/sess1.php
http://localhost/si572/php-12/sess1.php
Google

Array
(
    [value] => 2
)
Click Me!
```

```
http://localhost/si572/php-12/sess1.php
http://localhost/si572/php-12/sess1.php
Google

Array
(
    [value] => 10
)
Click Me!
```

Login / Logout

- Having a session is not the same as being logged in.
- Generally you have a session the instant you connect to a web site
- The Session ID cookie is set when the first page is delivered
- Login puts user information in the session (stored in the server)
- Logout removes user information from the session

login.php

```
<?php
session_start();
require_once "db.php";

if ( isset($_POST['email']) && isset($_POST['password']) ) {
    $e = mysql_real_escape_string($_POST['email']);
    $p = mysql_real_escape_string($_POST['password']);
    $sql = "SELECT name FROM users
           WHERE email = '$e' AND password='$p'";
    echo "<!--\n$sql\n-->\n";
    $result = mysql_query($sql);
    if ( $result === FALSE ) {
        echo "<p>Login incorrect.</p>\n";
        unset($_SESSION['name']);
    } else {
        $row = mysql_fetch_row($result);
        echo "<p>Login success.</p>\n";
        $_SESSION['name'] = $row[0];
    }
}
```

login.php

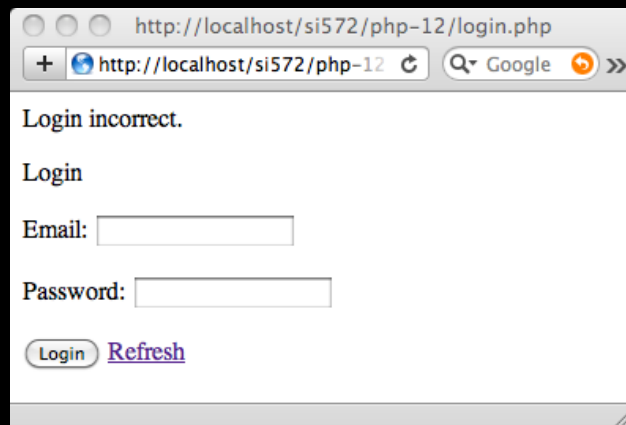
```
if ( $_SESSION['name'] ) {
    echo ('<p>Hello ' . $_SESSION['name'] . '</p>' . "\n");
    echo ('<p><a href="logout.php">Logout</a></p>' . "\n");
    return;
}
?>
<p>Login</p>
<form method="post">
<p>Email:
<input type="text" name="email"></p>
<p>Password:
<input type="text" name="password"></p>
<p><input type="submit" value="Login"/>
<a href="login.php">Refresh</a></p>
</form>
```

```
mysql> select * from users;
```

id	name	email	password
1	Chuck	csev@umich.edu	123
2	Glenn	gg@umich.edu	456
4	Sam	sam@umich.edu	zzz
15	Smith', 'sneaky') --	smith@umich.edu	secret
17	Sarah	sarah@umich.edu	123

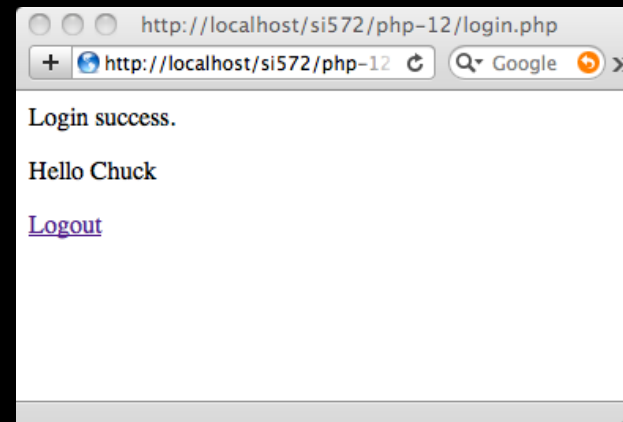
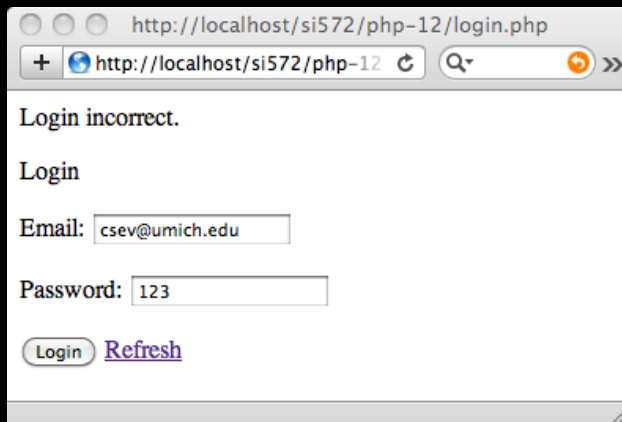
A screenshot of a web browser window at the URL `http://localhost/si572/php-12/login.php`. The page title is "Login". The "Email:" field contains the text `csev@umich.edu`. The "Password:" field contains the text `wrong`. Below the fields are two buttons: "Login" and "Refresh".

A screenshot of a web browser window at the URL `http://localhost/si572/php-12/login.php`. The page displays the message "Login incorrect." above the "Login" form. The "Email:" and "Password:" fields are empty. Below the fields are two buttons: "Login" and "Refresh".



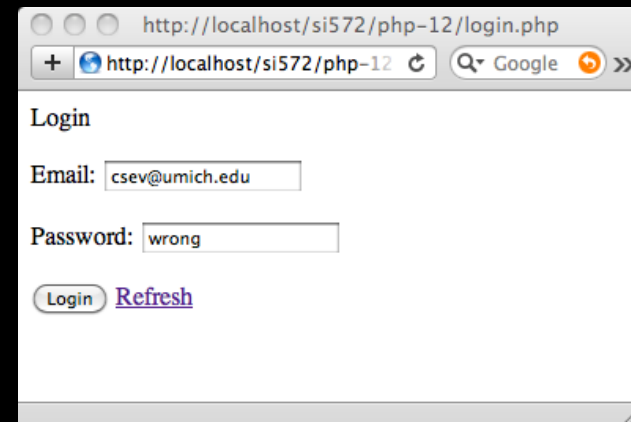
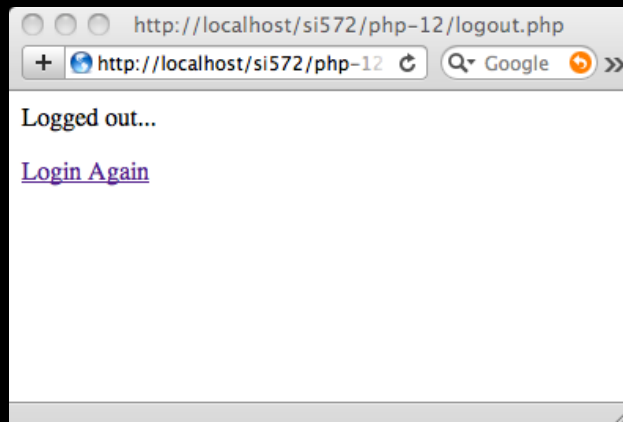
```
<!--  
SELECT name FROM users  
          WHERE email = 'csev@umich.edu' AND password='wrong'  
-->  
<p>Login incorrect.</p>  
<p>Login</p>  
<form method="post">
```

```
if ( $result === FALSE ) {
    echo "<p>Login incorrect.</p>\n";
    unset($_SESSION['name']);
} else {
    $row = mysql_fetch_row($result);
    echo "<p>Login success.</p>\n";
    $_SESSION['name'] = $row[0];
}
```



logout.php

```
<?php
session_start();
unset($_SESSION['name']);
?>
<p>Logged out...</p>
<p><a href="login.php">Login Again</a></p>
```



CRUD, POST, Redirect, Oh My!

POST / GET Gesture

- Once you do a POST, if you do refresh, the browser will re-send the POST data a second time
- The user gets a popup that tries to explain what is about to happen

http://localhost/si572/php-10/users/

Chuck	csev@umich.edu	123	Edit / Delete
Glenn	gg@umich.edu	456	Edit / Delete
Sam	sam@umcih	pp	Edit / Delete
Smith', 'sneaky') --	smith@umich.edu	secret	Edit / Delete
Fredy	fredy@umich.edu	secret	Edit / Delete

[Add New](#)

http://localhost/si572/php-10/users/add.php

Add A New User

Name:

Email:

Password:

[Cancel](#)

http://localhost/si572/php-10/users/add.php

```
INSERT INTO users (name, email, password)
VALUES ('Sarah', 'sarah@umich.edu', '123')
```

Success - [Continue...](#)

http://localhost/si572/php-10/users/index.php

Chuck	csev@umich.edu	123	Edit / Delete
Glenn	gg@umich.edu	456	Edit / Delete
Sam	sam@umcih	pp	Edit / Delete
Smith', 'sneaky') --	smith@umich.edu	secret	Edit / Delete
Sarah	sarah@umich.edu	123	Edit / Delete
Fredy	fredy@umich.edu	secret	Edit / Delete

[Add New](#)

http://localhost/si572/php-12/users/index.php

Chuck	csev@umich.edu	123	Edit / Delete
Glenn	gg@umich.edu	456	Edit / Delete
Sam	sam@umich.edu	zzz	Edit / Delete
Sarah	sarah@umich.edu	123	Edit / Delete

[Add New](#)



http://localhost/si572/php-12/users/add.php

Add A New User

Name:

Email:

Password:


[Cancel](#)

http://localhost/si572/php-12/users/add.php

```
INSERT INTO users (name, email, password)
VALUES ('Barb', 'barb@umich.edu', '123')
```

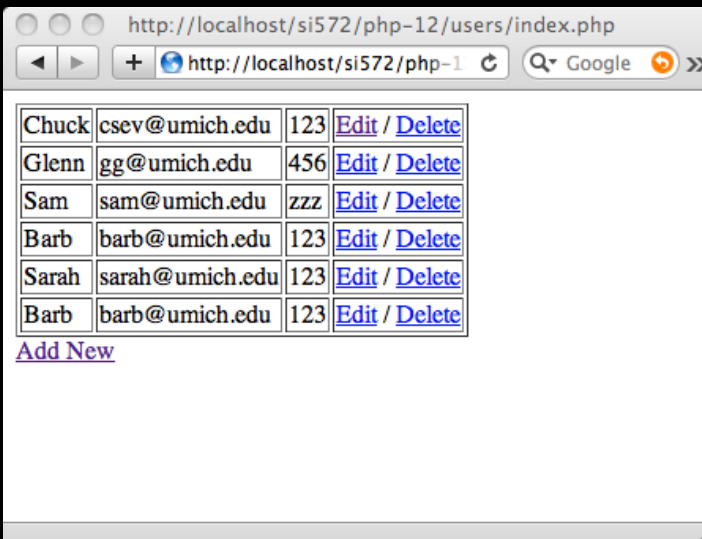
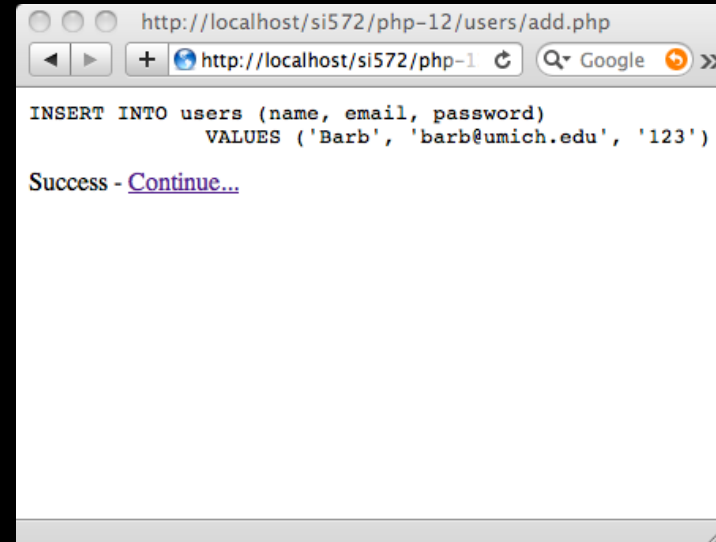
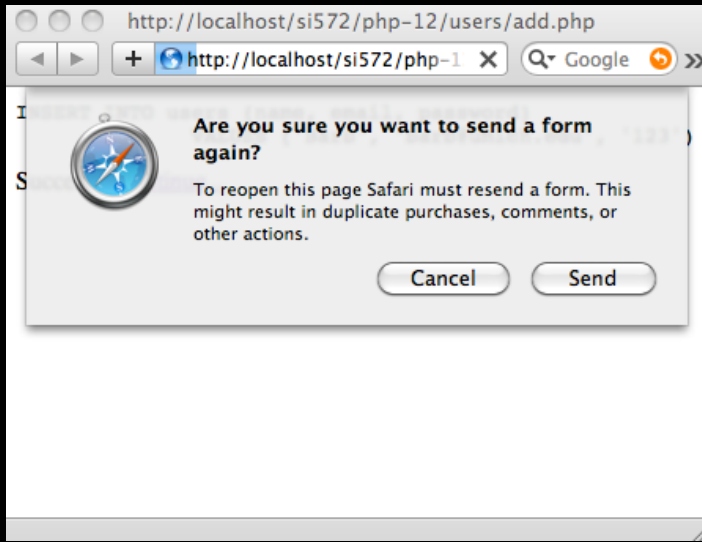
Success - [Continue...](#)

http://localhost/si572/php-12/users/add.php



Are you sure you want to send a form again?

To reopen this page Safari must resend a form. This might result in duplicate purchases, comments, or other actions.




Redirect

- If your application has not yet sent any data, it can send a special header as part of the HTTP Response
- The redirect header includes a URL that the browser is supposed to forward itself to
- It was originally used for web sites that moved from one URL to another

http://en.wikipedia.org/wiki/URL_redirection

POST / Redirect

- When our scripts receive POST data, update database tables, put out a success or error message, and put out a link to the next screen
- We can automatically redirect to another page after processing the post but we need to handle messages

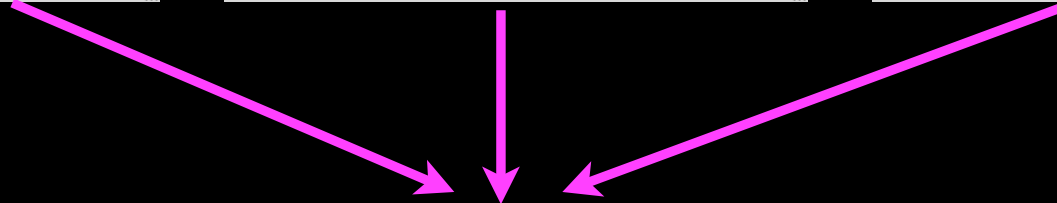


```
http://localhost/si572/php-12/users/add.php
http://localhost/si572/php-1
INSERT INTO users (name, email, password)
VALUES ('Barb', 'barb@umich.edu', '123')
Success - Continue...
```

```
http://localhost/si572/php-10/users/add.php
INSERT INTO users (name, email, password)
VALUES ('Sarah', 'sarah@umich.edu', '123')
Success - Continue...
```

```
http://localhost/si572/ph...0/users/delete.php?id=13
DELETE FROM users WHERE id = 13
Success - Continue...
```

```
http://localhost/si572/php-10/users/edit.php?id=4
UPDATE users SET name='Sam', email='sam@umich.edu',
password='zzz' WHERE id='4'
Updated - Continue...
```



```
http://localhost/si572/php-10/users/
```

Chuck	csev@umich.edu	123	Edit / Delete
Glenn	gg@umich.edu	456	Edit / Delete
Sam	sam@umcih	pp	Edit / Delete
Smith', 'sneaky') --	smith@umich.edu	secret	Edit / Delete
Fredy	fredy@umich.edu	secret	Edit / Delete

[Add New](#)

We come back to index.php from three different places.

We put error messages and success messages in \$_SESSION

index.php

```
<?php
require_once "db.php";
session_start();
if ( isset($_SESSION['error'] ) {
    echo '<p style="color:red">'. $_SESSION['error'] . "</p>\n";
    unset($_SESSION['error'] );
}
if ( isset($_SESSION['success'] ) {
    echo '<p style="color:green">'. $_SESSION['success'] . "</p>\n";
    unset($_SESSION['success'] );
}
echo '<table border="1">'. "\n";
$result = mysql_query("SELECT name, email, password, id FROM users");
while ( $row = mysql_fetch_row($result) ) {
    echo "<tr><td>";
    echo(htmlentities($row[0]));
    echo("</td><td>");
    echo(htmlentities($row[1]));
    echo("</td><td>");
    echo(htmlentities($row[2]));
    echo("</td><td>\n");
    echo('<a href="edit.php?id=' . htmlentities($row[3]) . '">Edit</a> / ');
    echo('<a href="delete.php?id=' . htmlentities($row[3]) . '">Delete</a>');
    echo("</td></tr>\n");
}
?>
</table>
<a href="add.php">Add New</a>
```

index.php

```
<?php
require_once "db.php";
session_start();
if ( isset($_SESSION['error'] ) ) {
    echo '<p style="color:red">' . $_SESSION['error'] . "</p>\n";
    unset($_SESSION['error']);
}
if ( isset($_SESSION['success'] ) ) {
    echo '<p style="color:green">' . $_SESSION['success'] . "</p>\n";
    unset($_SESSION['success']);
}
echo '<table border="1">' . "\n";
```

```

<?php
require_once "db.php";
session_start();
if ( isset($_POST['name']) && isset($_POST['email'])
    && isset($_POST['password'])) {
    $n = mysql_real_escape_string($_POST['name']);
    $e = mysql_real_escape_string($_POST['email']);
    $p = mysql_real_escape_string($_POST['password']);
    $sql = "INSERT INTO users (name, email, password)
        VALUES ('$n', '$e', '$p')";
    mysql_query($sql);
    $_SESSION['success'] = 'Record Added';
    header( 'Location: index.php' ) ;
    return;
}
?>
<p>Add A New User</p>
<form method="post">
<p>Name:
<input type="text" name="name"></p>
<p>Email:
<input type="text" name="email"></p>
<p>Password:
<input type="password" name="password"></p>
<p><input type="submit" value="Add New" />
<a href="index.php">Cancel</a></p>
</form>

```

add.php

http://localhost/si572/php-12/users/add.php

Add A New User

Name:

Email:

Password:

[Cancel](#)

http://localhost/si572/php-12/users/index.php

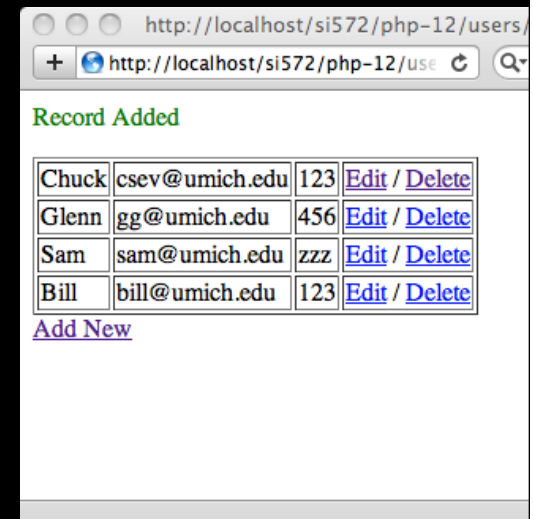
Record Added

Chuck	csev@umich.edu	123	Edit / Delete
Glenn	gg@umich.edu	456	Edit / Delete
Sam	sam@umich.edu	zzz	Edit / Delete
Bill	bill@umich.edu	123	Edit / Delete

[Add New](#)

index.php

```
<?php
require_once "db.php";
session_start();
if ( isset($_SESSION['error'] ) ) {
    echo '<p style="color:red">'. $_SESSION['error'] . "</p>\n";
    unset($_SESSION['error'] );
}
if ( isset($_SESSION['success'] ) ) {
    echo '<p style="color:green">'. $_SESSION['success'] . "</p>\n";
    unset($_SESSION['success'] );
}
echo '<table border="1">'. "\n";
$result = mysql_query("SELECT name, email, password, id FROM users");
while ( $row = mysql_fetch_row($result) ) {
    echo "<tr><td>";
    echo(htmlentities($row[0]));
    echo("</td><td>");
    echo(htmlentities($row[1]));
    echo("</td><td>");
    echo(htmlentities($row[2]));
    echo("</td><td>\n");
    echo('<a href="edit.php?id=' . htmlentities($row[3]) . '">Edit</a> / ');
    echo('<a href="delete.php?id=' . htmlentities($row[3]) . '">Delete</a>');
    echo("</td></tr>\n");
}
?>
</table>
<a href="add.php">Add New</a>
```



Summary

- Cookies
- Sessions
- Sessions in PHP
- Login / Logout
- POST / Redirect Pattern