Membuat Halaman Login dengan PHP

Halaman *login* sering kali digunakan untuk otentifikasi *user* yang memasuki sebuah halaman web. Apabila Anda berniat membuat halaman login dengan PHP, tentu Anda memerlukan *session* dan *cookies* dalam pembuatan sistem otentifikasi pengunjung situs web Anda. Mungkin Anda pemula PHP yang ingin belajar session dan cookies dan tertarik ingin belajar membuat halaman login seperti ini?

Tidak bisa dipungkiri bahwa keamanan akses di sebuah halaman web merupakan suatu hal yang sangat penting untuk diperhatikan. Data yang terdapat dalam suatu halaman web seringkali merupakan data penting yang tidak boleh diperlihatkan sembarangan kepada orang yang tidak berhak. Contohnya tidak mungkin seluruh data yang menyangkut privasi seseorang dalam suatu halaman web langsung ditampilkan kepada semua pengunjung yang membuka situs web tertentu.

Sedangkan Internet merupakan dunia yang luas di mana setiap orang dapat dengan mudah mengakses situs Anda. Untuk itu, hak akses user terhadap data yang terdapat dalam situs web perlu dibatasi. Cara yang biasa digunakan untuk membatasi hak akses seseorang terhadap isi sebuah website, yaitu dengan melakukan proses login terlebih dahulu. Ketika Anda memasuki sebuah halaman web yang seperti ini, sebelum dapat mencari data yang diperlukan dari website tersebut Anda diminta untuk memasukkan *user id* yang merupakan pengenal Anda dan password yang hanya diketahui oleh Anda sendiri, sehingga hanya user yang telah memiliki haklah yang diizinkan untuk mengakses situs tersebut.

Untuk dapat mengakses situs ini, user yang login harus memberikan user id dan password dengan benar. User id dan password yang telah diberikan akan divalidasi dengan database yang sudah ada di server. Apabila user id terdaftar dan password yang diberikan juga cocok dengan data yang terdapat dalam database, maka user akan diberi wewenang untuk masuk ke dalam sebuah halaman web.

Mungkin Anda ingin mencoba membuat halaman login seperti ini. Kita akan membahas bagaimana cara membuat halaman login seperti ini, sekaligus kita dapat melihat perbandingan proses login dengan session dibandingkan menggunakan cookies yang juga merupakan kemampuan bahasa script PHP. Tulisan sebenarnya ini lebih ditujukan bagi Anda yang mungkin masih asing dan belum mengenal session dan cookies.

Saat ini, PHP merupakan bahasa *script open source* yang masih seringkali digunakan untuk menciptakan halaman web yang dinamis terutama di lingkungan berbasiskan Linux. Dengan didukung oleh apache server dan database MySQL, PHP dapat menghasilkan halaman web dinamis yang cukup *powerful*. Nantinya database yang diperlukan untuk otentifikasi user id dan password akan kita buat dengan menggunakan database MySQL.

Sekarang kita akan membuat database yang akan digunakan untuk menyimpan user id dan password yang dapat digunakan untuk login dalam sebuah halaman web. Untuk membuat database dan table yang diperlukan, maka Anda dapat menjalankan mysql sebagai user root dan memberikan perintah-perintah SQL berikut:

```
mysql> CREATE DATABASE login ;

mysql>CONNECT login ;

mysql> CREATE TABLE user_data (
  name VARCHAR(40) NOT NULL,
  user_id VARCHAR(20) NOT NULL,
  password VARCHAR(20) NOT NULL,
  PRIMARY KEY(user_id)
  ) ;
```

Saat ini tabel user_data masih kosong. Sekarang kita masukkan data user yang nantinya akan Anda berikan untuk mencoba untuk login dalam halaman login yang nantinya Anda buat.

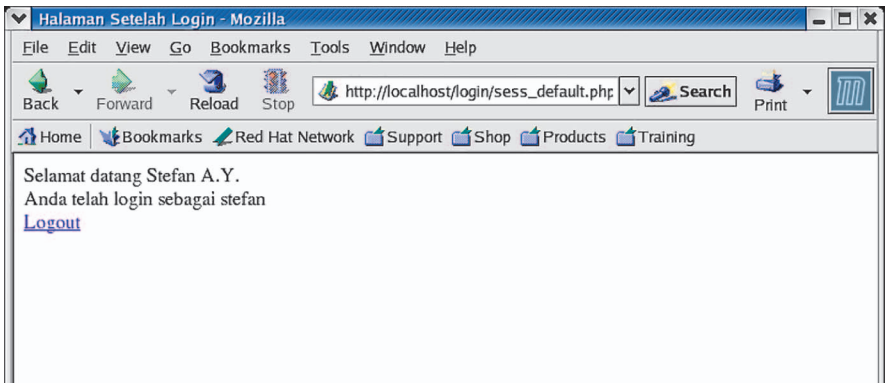
```
mysql> insert into user_data values
('Stefan A.Y.', 'stefan', 'sembarang');
```

Semua script PHP yang kita buat nantinya akan terkoneksi ke database login dalam mysql sebagai user 'php'. Berikut ini adalah perintah-perintah untuk memberikan wewenang bagi database user 'php' untuk mengakses tabel yang terdapat dalam database login.

```
mysql> GRANT ALL PRIVILEGES ON
login.* TO php@localhost identified by
'php' ;
```

Sekarang database sudah siap dan kita tinggal membuat script-script PHP untuk menghasilkan sebuah halaman login. Pertama-tama untuk mempermudah kita dalam mengakses database, buatlah sebuah file yang bernama 'connect.php' yang berisi variabel yang akan digunakan dalam koneksi dengan database. Tujuan kita membuat file connect.php, yaitu supaya apabila sewaktu-waktu terjadi perubahan informasi mengenai database dapat dengan mudah memperbaiki informasi program dengan mengganti isi variabel yang terdapat dalam connect.php.

```
// file connect.php
<?php
        $host="localhost"; // server dimana
        database disimpan
        $db_username="php"; // nama user
        yang dipakai untuk connect ke
        database
```



Halaman sesudah login.

```
        $db_password = "php"; // password
        database
        $db_name = "login"; // database
        yang digunakan
?>
```

Simpanlah script php ini ke dalam sebuah file yang bernama 'connect.php'

Menggunakan session

Sekarang kita akan membuat *form login* menggunakan *session*. Form ini yang akan digunakan user untuk login, di dalamnya terdapat script PHP yang bertugas untuk menjalankan proses login. Apabila proses login lancar maka akan ditampilkan script file sess_default.php.

Berikut ini adalah script form login. Kita misalkan file sess_login.php sebagai form login:

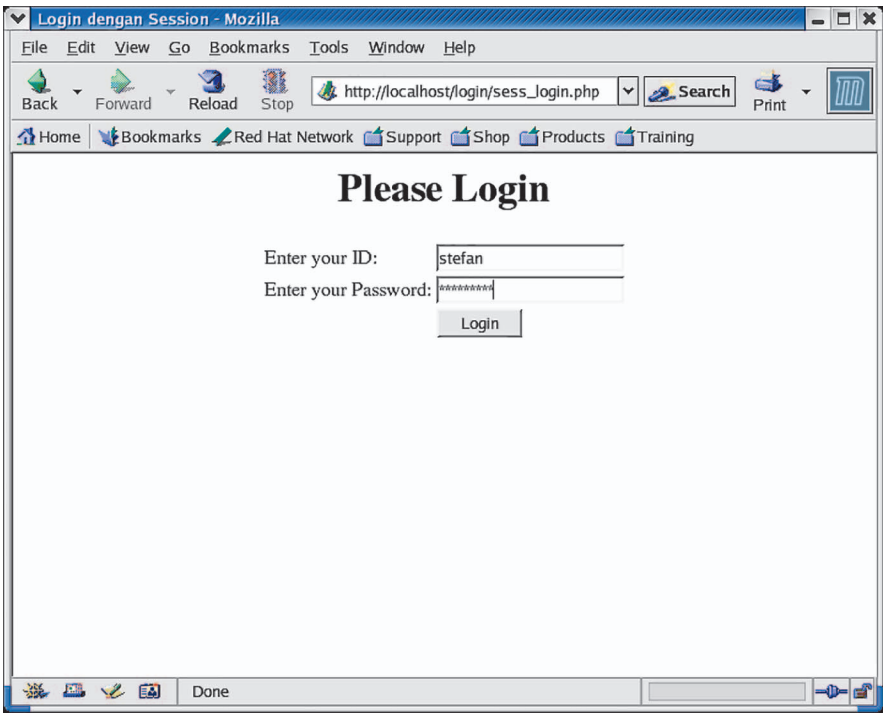
```
// file sess_login.php
<?php
if($action == "login")
{
  session_start();
  session_register("reg_username");
  session_register("reg_userid");
  session_register("reg_userpassword");
  require "connect.php";

  if(!($link = mysql_pconnect($host,
  $db_username,$db_password)))
  {
    printf("%s\n",mysql_error());
    exit();
  }
  if(!($r = mysql_db_query("$db_name",
  "select * from user_data  where
  user_id = '$user_id'")))
  {
    printf("Error %d:%s\n",mysql_errno(),
    mysql_error());
    exit();
  }
  if(($row = mysql_fetch_array($r)) &&
  ($user_password == $row[2] &&
  $user_password! = ""))
  {
    $reg_username = $row[0];
    $reg_userid = $user_id;
    $reg_userpassword = $user_password;
    header("location: sess_default.php");
  } else
  {
    session_unset();
    session_destroy();
    echo "Login gagal! Periksa user ID dan
    pasword anda<br>";
    echo "<a href=\"sess_login.php\">
    Back to login</a>";
    exit();
  }
}
?>
<html>
<head>
<title>Login dengan Session</title>
</head>
<body>
<h1 align="center">Please Login</h1>
<form method="post" action="sess_
login.php?action=login">
<table align="center">
<tr>
<td>Enter your ID:</td>
<td><input name="user_id" type=
"text"></td>
</tr>
<tr>
<td>Enter your Password:</td>
<td><input name="user_password"
type="password"></td>
</tr>
<tr>
<td></td>
<td><input name="submit" type=
"submit" value="Login"></td>
</tr>
</table>
</form>
</body>
</html>
```

Script PHP pada form ini bertugas untuk mencocokkan data yang dimasukkan user dengan data yang terdapat dalam database. Setelah user menekan tombol *submit*, maka halaman sess_login akan dimuat sekali lagi dengan memberikan perintah login pada script php. Barulah script PHP yang terdapat pada awal program dilaksanakan. Bila otentifikasi berhasil, maka akan dipanggil halaman yang bernama sess_default.php, sedangkan bila gagal akan dicetak pesan kesalahan dalam memasukkan user id atau password.

Pada awal script dilaksanakan perintah session_start(), perintah ini menandakan bahwa kita memulai sesi baru. Dilanjutkan dengan mendaftarkan variabel-variabel session. Variabel ini nantinya bisa digunakan dalam halaman web Anda yang sebenarnya seperti informasi mengenai nama user, umur, dan lain-lain yang sedang masuk ke dalam situs web Anda.

Dalam pembahasan kali ini, kita hanya menggunakan tiga buah variabel dan tiga kolom pada tabel yang terdapat dalam database mysql, namun pada penerapan yang sebenarnya Anda dapat menambahkan variabel lain yang berisi data pribadi user yang sedang login.

Tentu saja setelah otentifikasi form login telah berhasil, kita akan mempersilakan user untuk mengakses halaman yang



Login dengan session.

sesungguhnya akan ditampilkan padanya. Kita misalkan halaman ini diberi nama sess_default.php. Berikut ini contoh halaman web yang dituju setelah proses login:

```
// file sess_default.php
<?php
session_start();
if(!session_is_registered('reg_userid'))
{
  header("location: sess_login.php");
  exit();
}
?>
<html>
<head>
<title>Halaman Setelah Login</title>
<body>
Selamat datang <?echo "$reg_username";
?><br>
Anda telah login sebagai <?echo
"$reg_userid";?><br>
<a href="sess_logout.php">Logout</a>
</body>
</html>
```

Tag yang terdapat di dalam tag <body> dapat Anda ganti dengan halaman html Anda sendiri. Variabel session yang telah terdaftar seperti reg_username maupun reg_userid dapat Anda gunakan untuk ditampilkan pada halaman html Anda. Dari halaman sess_default.php inilah link-link ke halaman yang lain akan dimulai. Namun ada kemungkinan user yang iseng langsung menambahkan teks: default.php ke dalam url sehingga halaman tersebut dapat tampil tanpa harus login terlebih dahulu.

Maka dari itu, untuk menjaga keamanan halaman web tersebut sebaiknya pada setiap halaman web yang ingin disembunyikan terhadap user yang belum login pada script diatas kita memeriksa apakah user id sudah terdaftar sebagai variabel session.

```
<?php
session_start();
if(!session_is_registered('reg_userid'))
{
  header("location: sess_login.php");
  exit();
}
?>
```

Script ini akan melakukan pengecekan terhadap variabel session 'reg_userid'. Apabila variabel session belum terdaftar maka user diminta untuk melakukan prosedur login melalui halaman sess_login.php.

Setelah user dapat memasuki halaman web melalui proses login dan ingin mengakhiri pekerjaannya, user harus melalui proses *logout* terlebih dahulu. Proses logout ini penting karena melalui proses ini data mengenai user akan dilupakan oleh *browser*. Apabila Anda lupa untuk melakukan logout, dan ada orang lain lain yang kebetulan menggunakan komputer yang sama kemungkinan dapat menyamar sebagai Anda untuk memasuki halaman tersebut dengan menggunakan user id dan password Anda. Bukankah hal ini cukup berbahaya.

Berikut ini empat baris script php yang digunakan untuk menghapus semua variabel session yang telah Anda gunakan:

```
// File sess_logout.php
<?php
  session_start();
  session_unset();
  session_destroy();
  header("location: sess_login.php");
?>
```

Dengan begini semua variabel session yang telah ada akan dihapus dari ingatan browser yang Anda gunakan. Apabila sewaktu-waktu Anda ingin memberi kesempatan pada user untuk melakukan logout Anda tinggal menambahkan *hyperlink* pada halaman web Anda yang mengarah kepada halaman sess_logout.php ini.

Sekarang proses login yang telah Anda buat sudah selesai. Cobalah untuk membuka halaman sess_login.php pada browser kesayangan Anda. Untuk saat ini Anda diminta untuk memasukkan user id 'stefan' dan user password-nya 'sembarang'. Maka akan terbuka halaman baru yang bernama sess_default. Di sini akan Anda tempatkan isi dari halaman web Anda. Halaman lainnya dapat dituju dengan menempatkan hyperlink-nya pada halaman ini.

Menggunakan cookies

Setelah mengetahui pembuatan login menggunakan session, sekarang kita membuat halaman login menggunakan cookies. Sebelum membuat sebuah halaman login menggunakan cookies, Anda tetap perlu melakukan perintah-perintah pembuatan database dan tabel user_data pada MySQL seperti di atas. Salah satu perbedaan menggunakan session diban-

dingkan dengan menggunakan cookies yaitu karena variabel session juga disimpan di server sedangkan variabel cookies hanya disimpan di komputer client.

Sebelum kita mulai membuat halaman login, kita akan mempersiapkan *function* yang akan dipergunakan oleh script PHP selanjutnya. Kita akan membuat sebuah file function.php yang berisi function autentifikasiUser yang nantinya berguna untuk mencocokkan data cookies dengan database user yang ada.

Di dalam file function.php juga terdapat function untuk menghapus cookies yang telah ada. Nantinya function hapusCookies akan digunakan dalam proses logout. Function yang terdapat dalam file function.php dapat digunakan dengan terlebih dahulu dengan memanggilnya melalui satu baris script berikut. Sehingga kita tidak perlu mengetikkannya berulang-ulang yang akan menyebabkan source code program menjadi panjang.

```
require "function.php";
```

Berikut ini adalah script php yang terdapat dalam file function.php:

```
// File function.php
<?php
include "connect.php";
function autentifikasiUser($user,
$password)
{
  global $host, $db_username,
  $db_password, $db_name;
  if(!($conn=mysql_pconnect($db_host,
  $db_username,$db_password))){
    printf ("Error %s\n",mysql_error());
    return 0;
  }
  if(!($result=mysql_db_query
  ("$db_name","select * from user_data
  where user_id='$user'")))
  {
    printf ("Error %s\n",mysql_error());
    return 0;
  }
  if(($row=mysql_fetch_array($result)) &&
  ($password==$row[2] && $password
  !="")) {
    setcookie("cookie_username",
    "$row[0]");
    setcookie("cookie_userid","$row[1]");
    setcookie("cookie_password",
    "$row[2]");
    return 1;
  } else
    return 0;
}
function hapusCookies()
{
  setcookie("cookie_username","");
  setcookie("cookie_userid","");
  setcookie("cookie_password","");
}
?>
```

Function autentifikasiUser akan mengembalikan nilai 1 bila data user terdapat dalam database sekaligus mengisikannya dalam variabel cookies dengan perintah setcookie() namun akan mengembalikan nilai 0 bila proses membandingkan data tidak berhasil dengan baik.

Sekarang saatnya kita akan membuat form login dengan cookies. Berikut ini adalah contoh script-script yang bisa digunakan:

```
// file cookie_login.php
<?php
if($action==="login")
{
  require "function.php";
  hapusCookies();
  if(autentifikasiUser($user_id,$user_
  password)) {
    header("location: cookie_default.php");
    exit();
  } else {
    echo "Login gagal! Periksa user ID dan
    password anda<br>";
    echo "<a href=\"cookie_login.php\
    ">Back to login</a>";
    exit();
  }
}
?>
<html>
<head>
<title>Login dengan cookies</title>
</head>
<body>
<h1 align="center">Please Login</h1>
<form method="post" action="cookie_
login.php?action=login">
<table align="center">
<tr>
<td>Enter your ID:</td>
<td><input name="user_id" type=
"text"></td>
</tr>
<tr>
<td>Enter your Password:</td>
<td><input name="user_password"
type="password"></td>
</tr>
<tr>
```

```
<td></td>
<td><input name="submit" type=
"submit" value=" Login "></td>
</tr>
</table>
</form>
</body>
</html>
```

Sama halnya seperti script yang kita buat dengan session, apabila user memasukkan user id dan password-nya dengan benar (user id 'stefan' dan user password 'sembarang'), maka perjalanan akan dilanjutkan menuju halaman yang sesungguhnya (dalam hal ini cookie_default.php).

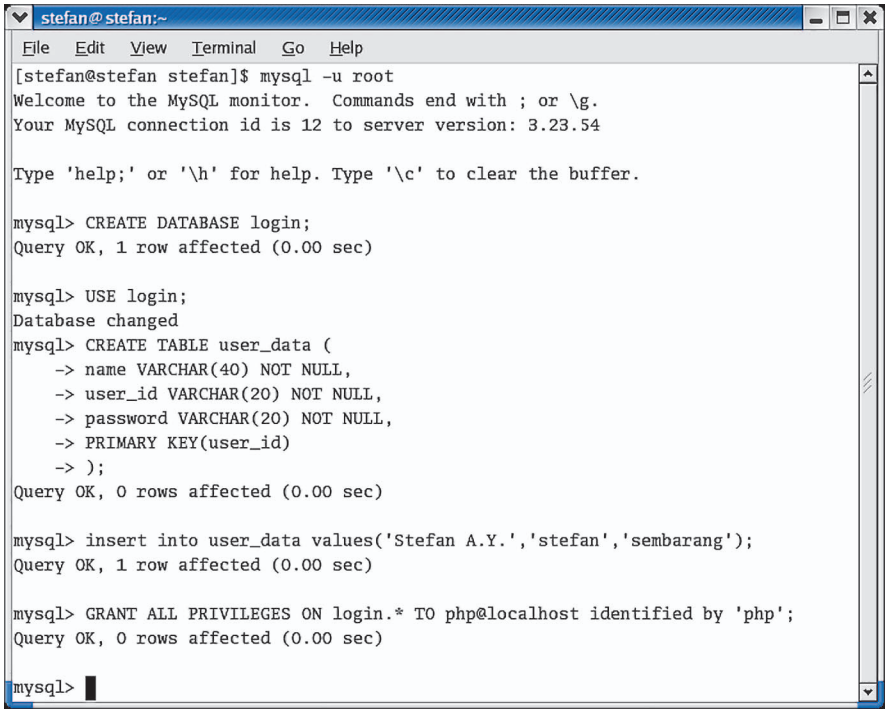
Berikut ini adalah contoh script cookie_default.php:

```
// File cookie_default.php
<?php
require "function.php";
if(!autentifikasiUser($cookie_userid,$cookie_password))
{
  header("location: cookie_login.php");
  exit();
}
?>
<html>
<head>
<title>Halaman Setelah Login</title>
<body>
Selamat datang <?echo "$cookie_
username";?><br>
Anda telah login sebagai <?echo
"$cookie_userid";?><br>
<a href="cookie_logout.php">Logout
</a>
</body>
</html>
```

File cookie_default.php sama kegunaannya dengan file sess_default.php, di sinilah awal halaman web yang sesungguhnya yang ingin ditampilkan kali pertama pada user yang telah otentifikasi.

Namun sebelum tag html yang sebenarnya perlu dilakukan pencocokan kembali antara data yang terdapat dalam cookies dengan data yang terdapat dalam database. Sama halnya dengan session.

```
<?php
require "function.php";
if(!autentifikasiUser($cookie_userid,
$cookie_password))
```



Mempersiapkan database.

```
{
  header("location: cookie_login.php");
  exit();
}
?>
```

Script ini perlu dicantumkan pada awal setiap halaman yang ingin dilindungi. Dengan demikian, halaman tersebut tidak dapat dibuka oleh user dengan mengetikkannya secara langsung dengan menambahkannya pada url tanpa melalui login terlebih dahulu.

Untuk proses logout menggunakan cookies kita cukup menghapus isi vaiabel cookies dengan menggunakan perintah yang sama dengan waktu menciptakannya, yaitu dengan perintah setcookies().

Berikut ini merupakan proses logout halaman otentifikasi yang menggunakan cookies:

```
// File cookie_logout.php
<?php
  require "function.php";
  hapusCookies();
  header("location: cookie_login.php");
?>
```

Mungkin dari benak Anda muncul pertanyaan cara manakah yang lebih baik untuk digunakan dalam membuat halaman login pada web. Kedua cara di atas tidak ada yang baik maupun yang buruk, keduanya dapat digunakan sesuai dengan kebutuhan. Apabila dibutuhkan untuk memeriksa data user setiap kali memasuki halaman web yang dilindungi, Anda dapat menggunakan variabel cookies. Namun bila Anda menginginkan kemudahan atau tidak memerlukan validasi dengan database di setiap halaman yang dilindungi, Anda dapat menggunakan session dalam proses autentifikasi.

Saat ini, baru terdapat satu user di dalam database login. Untuk menambahkan data user baru Anda dapat membuat form baru menggunakan script PHP, semua proses memasukkan data ke dalam tabel di database dilakukan oleh script php sehingga Anda tidak perlu mengetikkannya langsung pada command line MySQL seperti saat kita memasukkan user 'stefan' ke dalam database login.

Kita telah merasakan bahwa membuat halaman login dengan session maupun cookies bukan hal yang sulit. Dengan berbekal pengetahuan tentang session dan cookies, Anda dapat membuat login untuk website Anda sendiri. Selamat mencoba!

Stefan *(stefan_ay@plasa.com)*