CNT 4714: Enterprise Computing Fall 2009

Introduction to PHP – Part 3

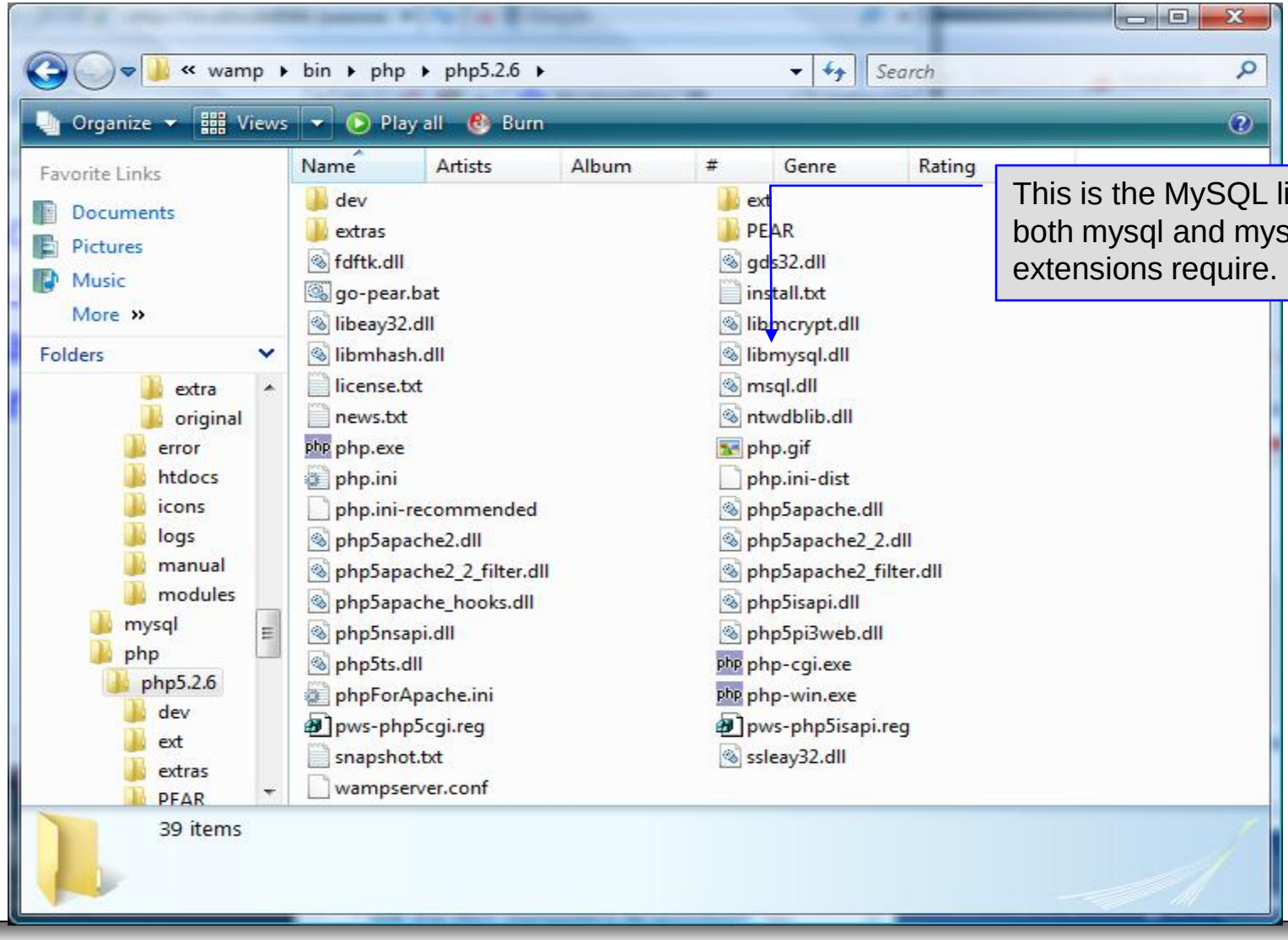
Instructor : Dr. Mark Llewellyn
markl@cs.ucf.edu
HEC 236, 407-823-2790
http://www.cs.ucf.edu/courses/cnt4714/fall2009

School of Electrical Engineering and Computer Science
University of Central Florida

PHP and Database Connectivity

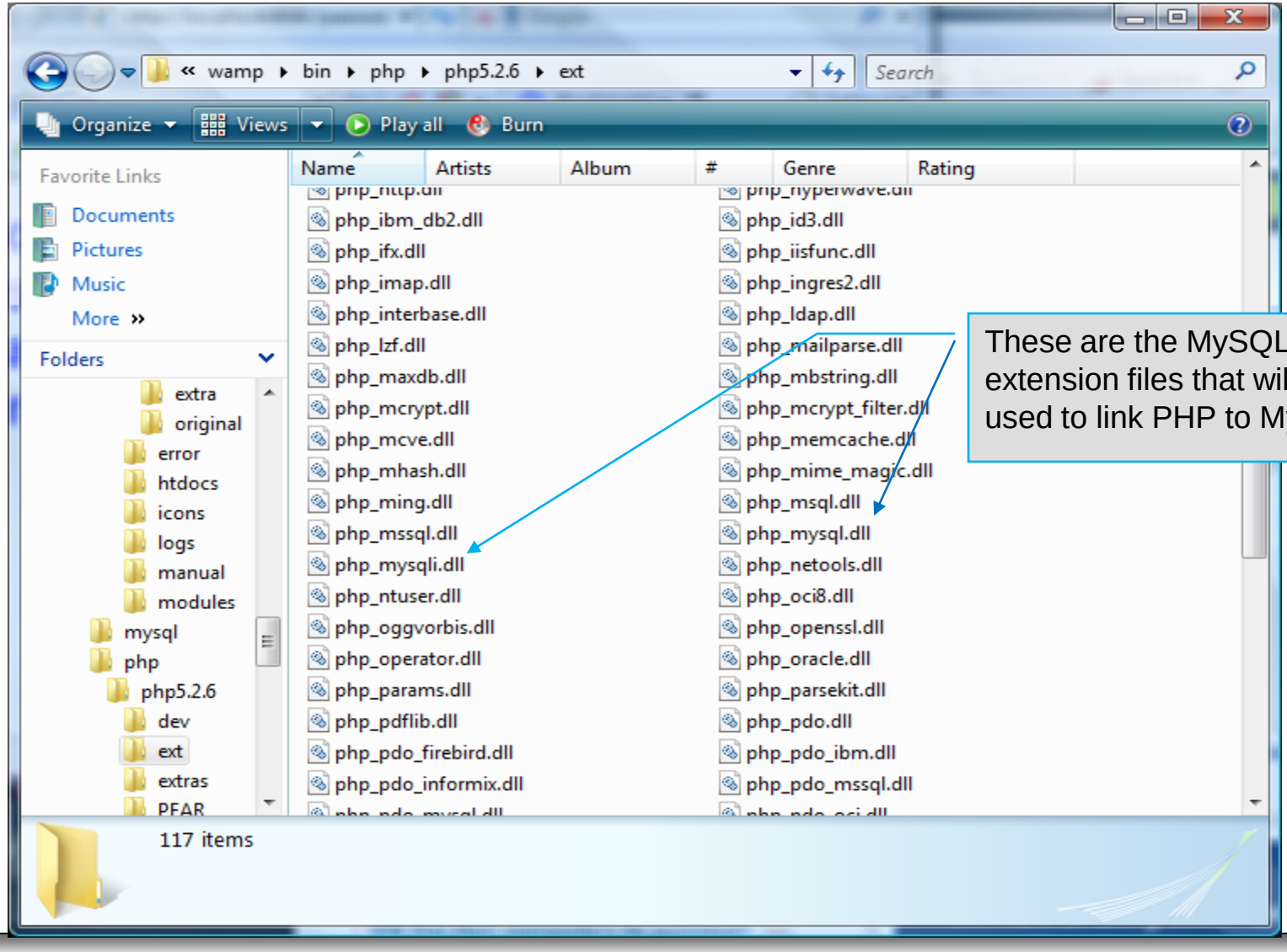
- PHP offers built-in support for a wide variety of database systems from Unix DBM through relational systems such as MySQL to full size commercial systems like Oracle.
- We'll continue to use MySQL as the underlying database system so that you can easily compare the work we've done with MySQL using Java servlets and JSPs.
- Before you go any further in these notes you must configure PHP to access MySQL databases. Beginning with PHP 5, MySQL is not enabled by default in PHP, nor is the MySQL library bundled with PHP.
  - Versions of MySQL greater than 4.1.0 use MySQLi extensions.
  - Versions of MySQL less than 4.1.0 use MySQL extensions.

PHP and Database Connectivity (cont.)

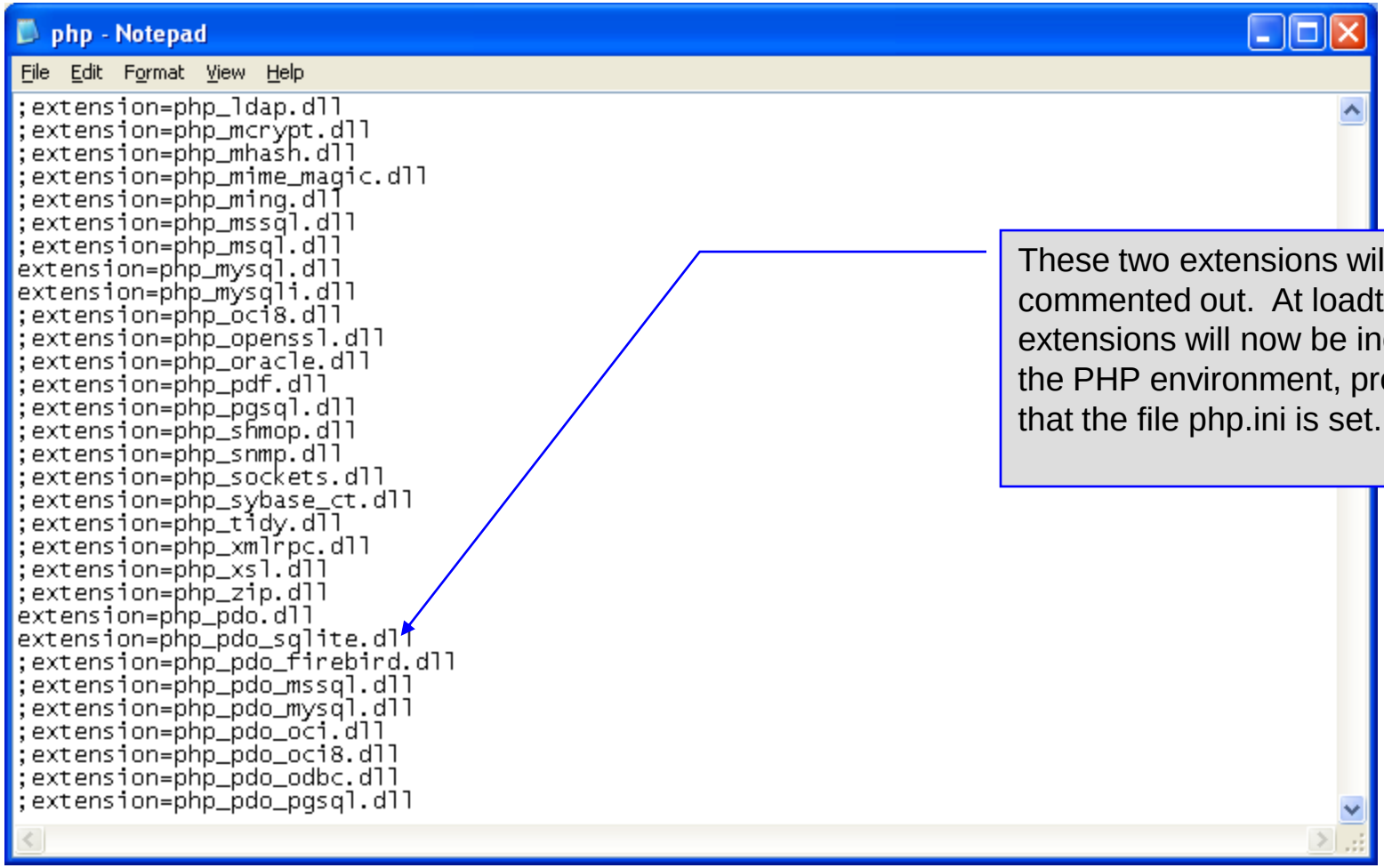


This is the MySQL library that both mysql and mysqli extensions require.

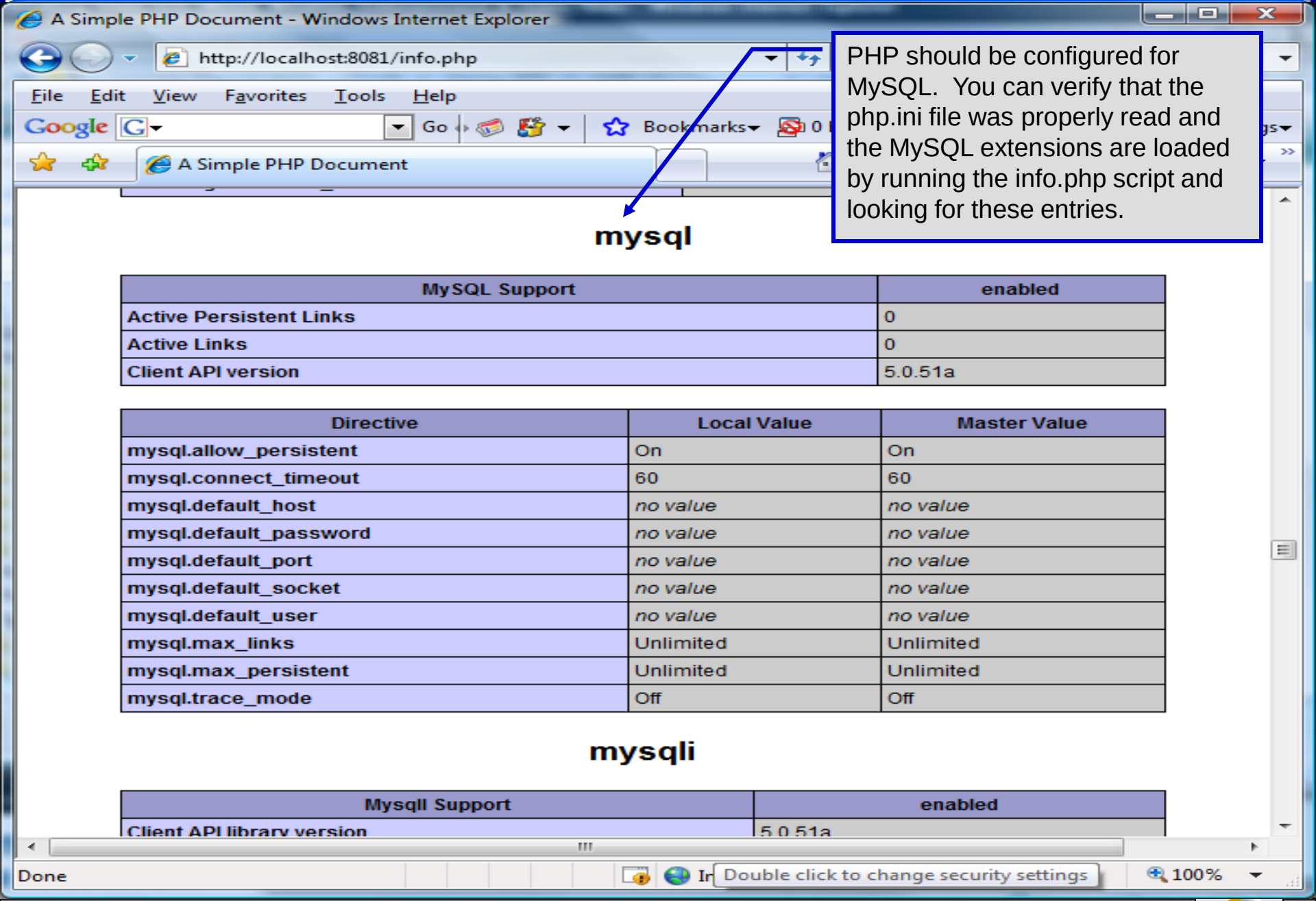
PHP and Database Connectivity (cont.)



PHP and Database Connectivity (cont.)



These two extensions will not be commented out. At loadtime, these extensions will now be included in the PHP environment, provided that the file php.ini is set..


A Simple PHP Document - Windows Internet Explorer
http://localhost:8081/info.php
File Edit View Favorites Tools Help
Google Go Bookmarks
A Simple PHP Document
PHP should be configured for MySQL. You can verify that the php.ini file was properly read and the MySQL extensions are loaded by running the info.php script and looking for these entries.
mysql
MySQL Support enabled
Active Persistent Links 0
Active Links 0
Client API version 5.0.51a
Directive Local Value Master Value
mysql.allow_persistent On On
mysql.connect_timeout 60 60
mysql.default_host no value no value
mysql.default_password no value no value
mysql.default_port no value no value
mysql.default_socket no value no value
mysql.default_user no value no value
mysql.max_links Unlimited Unlimited
mysql.max_persistent Unlimited Unlimited
mysql.trace_mode Off Off
mysqli
Mysqli Support enabled
Client API library version 5.0.51a
Done
Double click to change security settings
100%

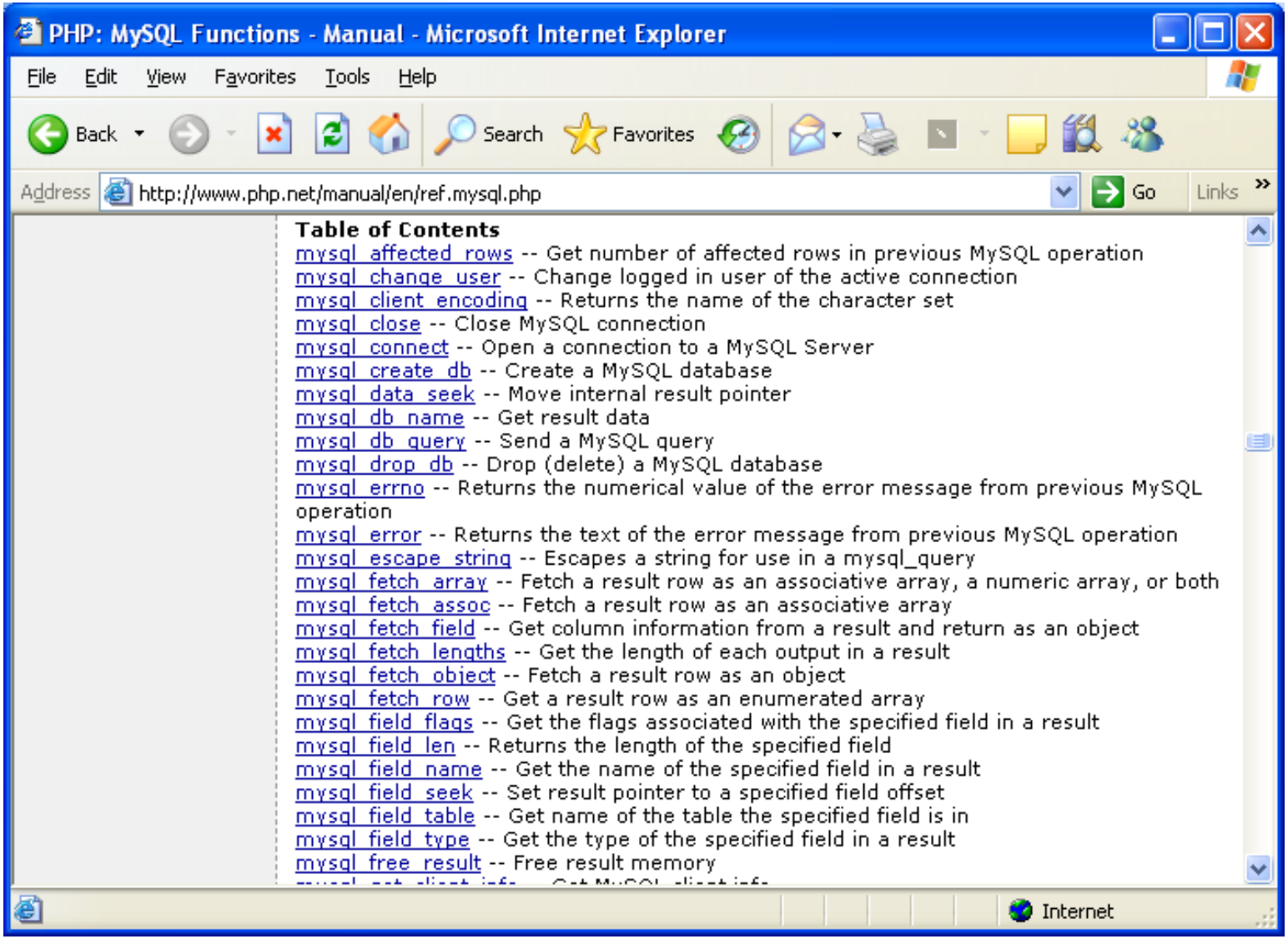
PHP and Database Connectivity (cont.)

- PHP contains a fairly extensive set of commands that can be used to access and manipulate MySQL databases.
- A very brief listing of some of these commands appears on the next page.
- For a complete listing see:

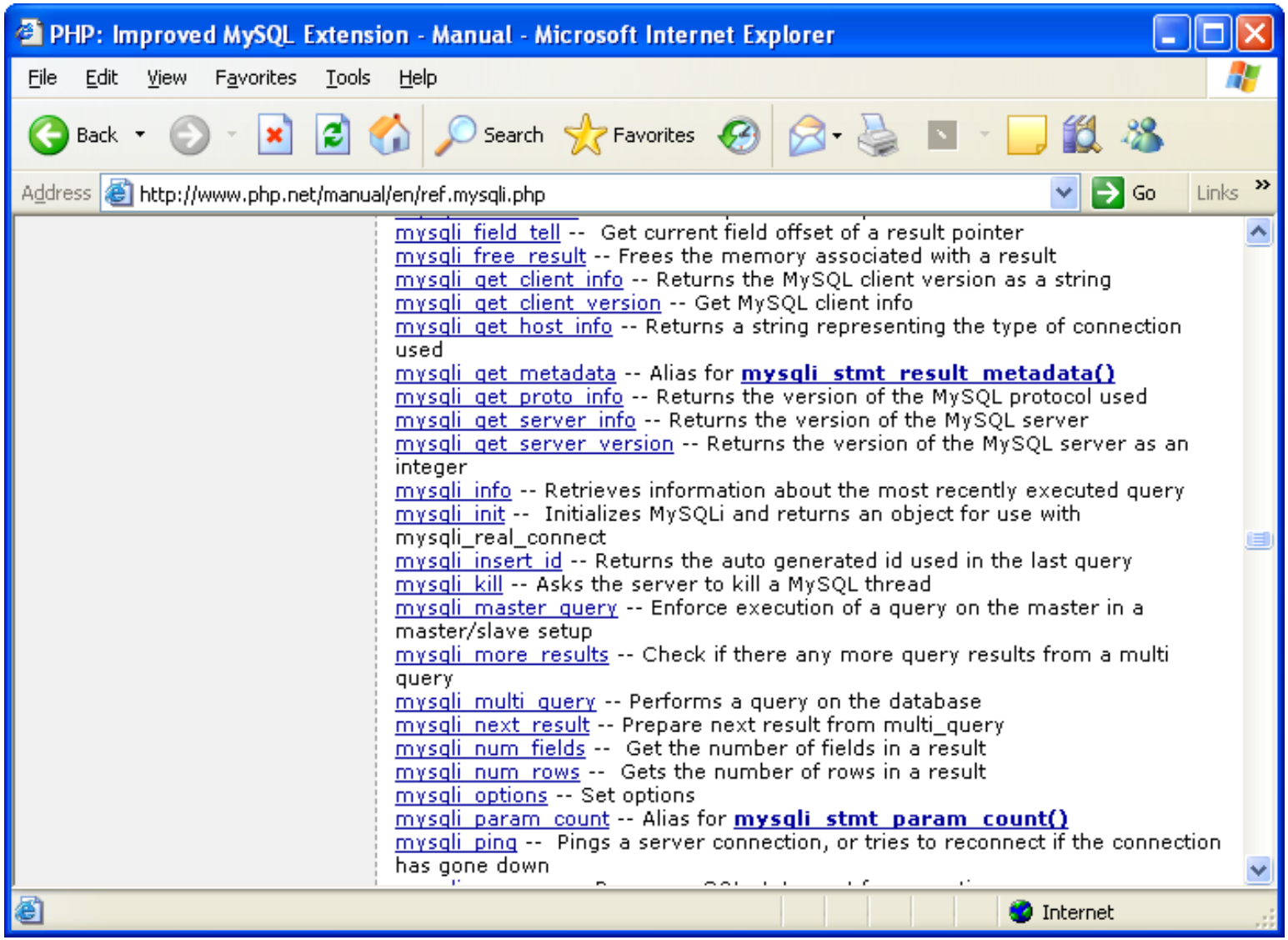
  http://us2.php.net/manual/en/print/ref.mysql.php.

  http://us2.php.net/manual/en/print/ref.mysqli.php.

Portion of `mysql.dll` Extension



Portion of `mysqli.dll` Extension



PHP and Database Connectivity (cont.)

- Now that you have PHP set to accept MySQL extensions, let's connect to the bike database that we used for examples with Java servlets and JSPs.
- The following example is a simple database connection process in PHP where the client interacts with the database from an XHTML form that simply asks them to select which attributes from the bikes table that they would like to display. This is done through the `data.html` file.
- When the client clicks the submit query button, the `database.php` script executes by connecting to the database, posting the query, retrieving the results, and displaying them to the client.

data.html

Client side

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<!-- data.html     -->
<!-- Querying a MySQL Database From a PHP Script -->

<html xmlns = "http://www.w3.org/1999/xhtml">
  <head>      <title>Sample Database Query From PHP</title>   </head>
  <body style = "background-color: #545454" background=image1.jpg >
    <h2 style = "font-family: arial color: blue"> Querying a MySQL database from a PHP Script. </h2>
    <form method = "post" action = "database.php">
      <p>Select a field to display:
        <!-- add a select box containing options for SELECT query   -->
        <select name = "select">
          <option selected = "selected">*</option>
          <option>bikename</option>
          <option>size</option>
          <option>color</option>
          <option>cost</option>
          <option>purchased</option>
          <option>mileage</option>
        </select>
      </p>
      <input type = "submit" value = "Send Query"   style = "background-color: blue;
        color: yellow; font-weight: bold" />
    </form>
  </body>   </html>
```

database.php

Server side

Page 1

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<!-- database.php       -->
<!-- Program to query a database and send results to the client.     -->

<html xmlns = "http://www.w3.org/1999/xhtml">
  <head>     <title>Database Search Results</title>   </head>

  <body style = "font-family: arial, sans-serif"
    style = "background-color: #4A766E" background=image1.jpg link=blue vlink=blue>
    <?php

      extract( $_POST );

      // build SELECT query
      $query = "SELECT " . $select . " FROM bikes";

      // Connect to MySQL
      if ( !( $database = mysqli_connect( "localhost",
        "root", "root“, bikedb ) ) )
        die( "Could not connect to database" );
```

Default query is to select the attributes chosen by the client for use in a SELECT query.

Connect to MySQL database. URL, username, password, and database all specified.

database.php

Server side

Page 2

```
   // query bikedb database
   if ( !( $result = mysql_query( $database, $query ) ) ) {
      print( "Could not execute query! <br />" );
      die( mysql_error() );
   }
?>

<h3 style = "color: blue">
Database Search Results</h3>
<table border = "1" cellpadding = "3" cellspacing = "3"
   style = "background-color: #00FFFF">  <!-- ADD8E6  -->

   <?php
         // fetch meta-data
         $metadata = mysqli_fetch_fields( $result);
         print("<tr>");
         for ($i=0; $i<count($metadata); $i++){
                  print("<td>");
                  printf("%s",$metadata[$i]->name);
                  print("</td>");
         }
         print("</tr>");
```

Get metadata for the query

Display metadata in the top row of the table

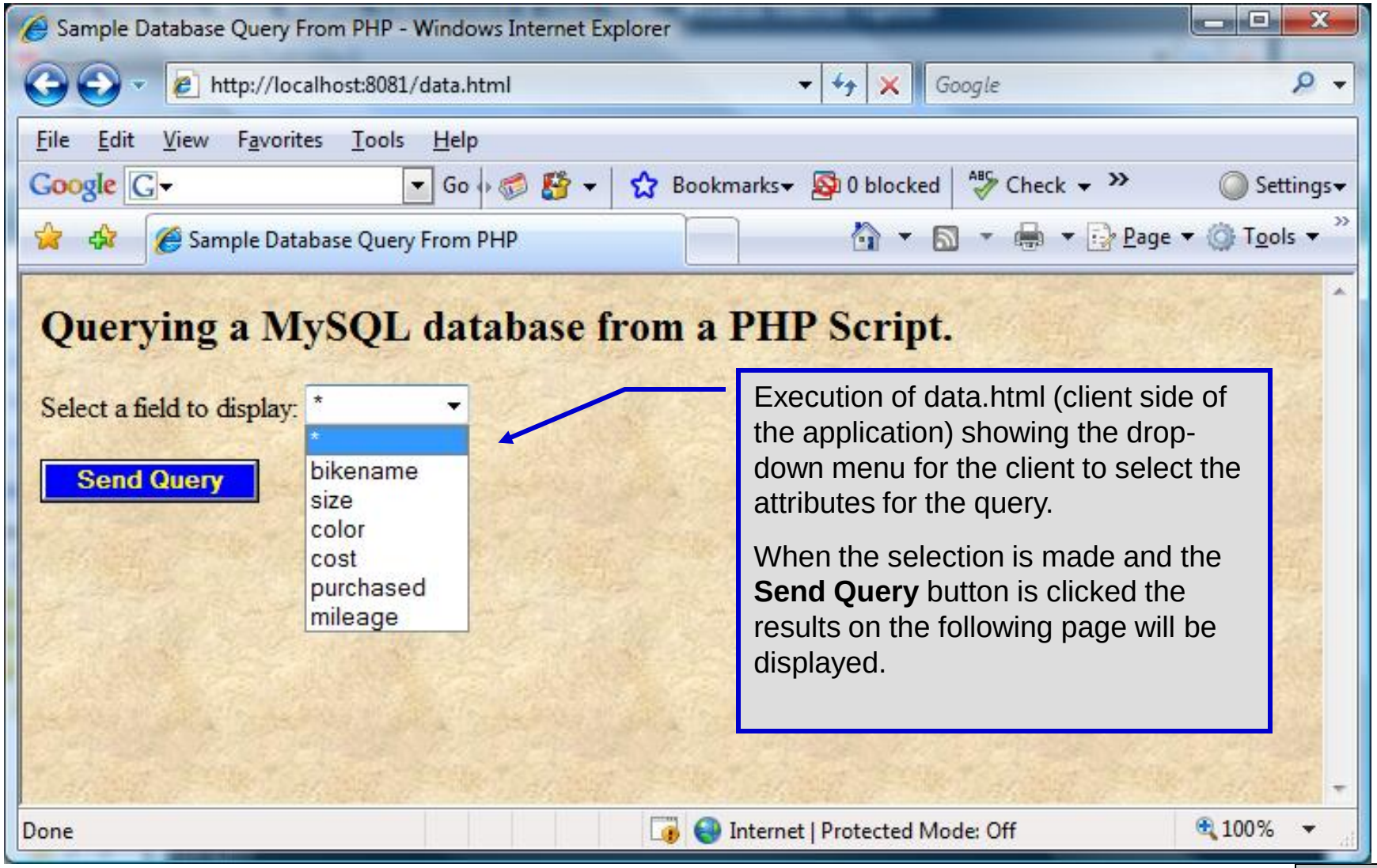
database.php

Server side

Page 3

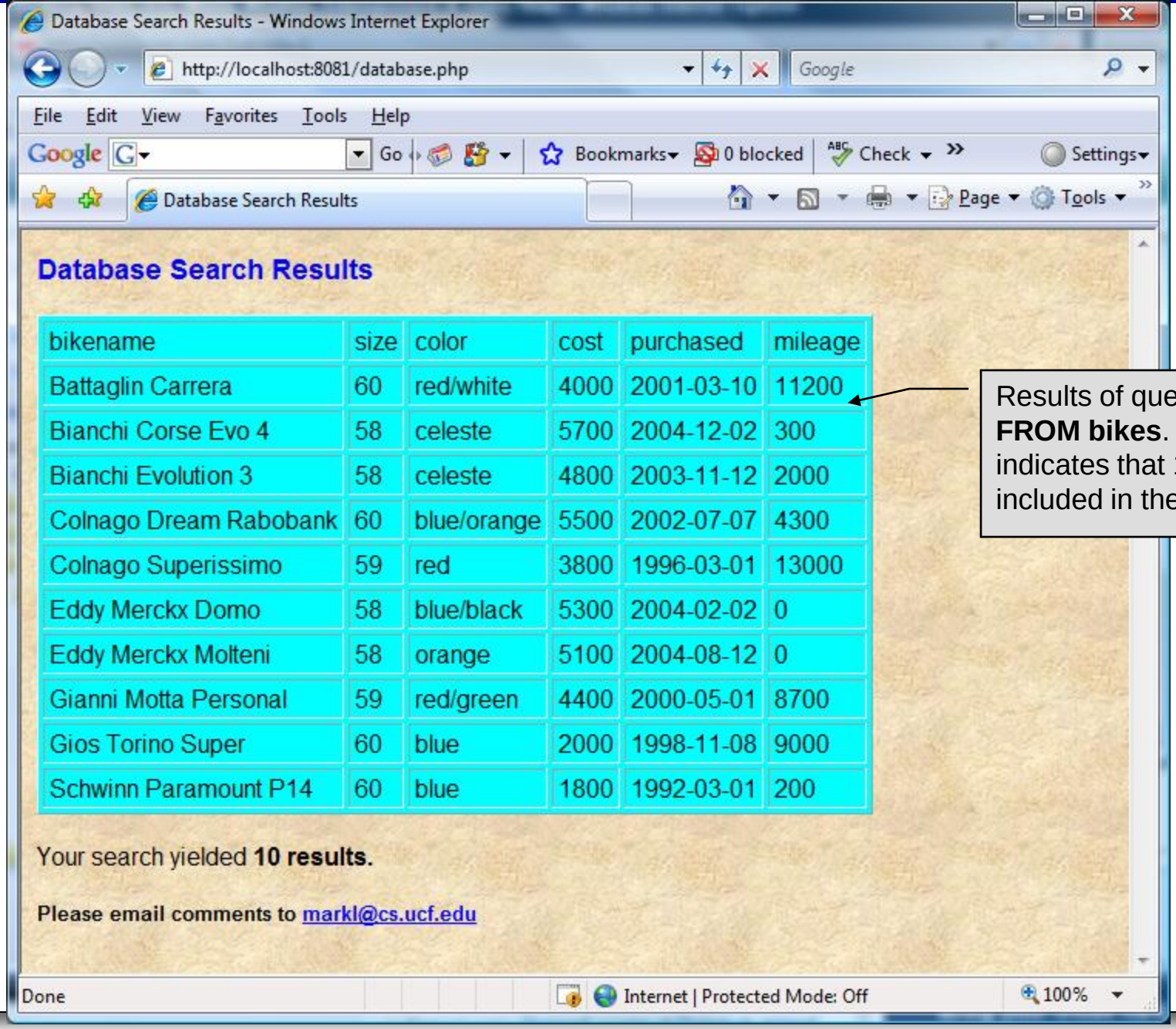
```
      // fetch each record in result set
         for ( $counter = 0;
            $row = mysql_fetch_row( $result );
            $counter++ ){
            // build table to display results
            print( "<tr>" );
            foreach ( $row as $key => $value )
               print( "<td>$value</td>" );
            print( "</tr>" );
         }
         mysql_close( $database );
      ?>
   </table>
   <br />Your search yielded <strong>
         <?php print( "$counter" ) ?> results.<br /><br /></strong>
         <h5>Please email comments to
         <a href = "mailto:markl@cs.ucf.edu">
                              markl@cs.ucf.edu
          </a>
        </h5>
</body></html>
```

Execution of data.html – Client side



Execution of data.html (client side of the application) showing the drop-down menu for the client to select the attributes for the query.

When the selection is made and the **Send Query** button is clicked the results on the following page will be displayed.



Results of query **SELECT * FROM bikes**. Display indicates that 10 rows were included in the result.

Cookies

- A cookie is a text file that a Web site stores on a client's computer to maintain information about the client during and between browsing sessions.
- A Web site can store a cookie on a client's computer to record user preferences and other information that the Web site can retrieve during the client's subsequent visits. For example, many Web sites use cookies to store client's zipcodes. The Web site can retrieve the zipcode from the cookie and provide weather reports and news updates tailored to the user's region.
- Web sites also use cookies to track information about client activity. Analysis of information collected via cookies can reveal the popularity of Web sites or products.

Cookies (cont.)

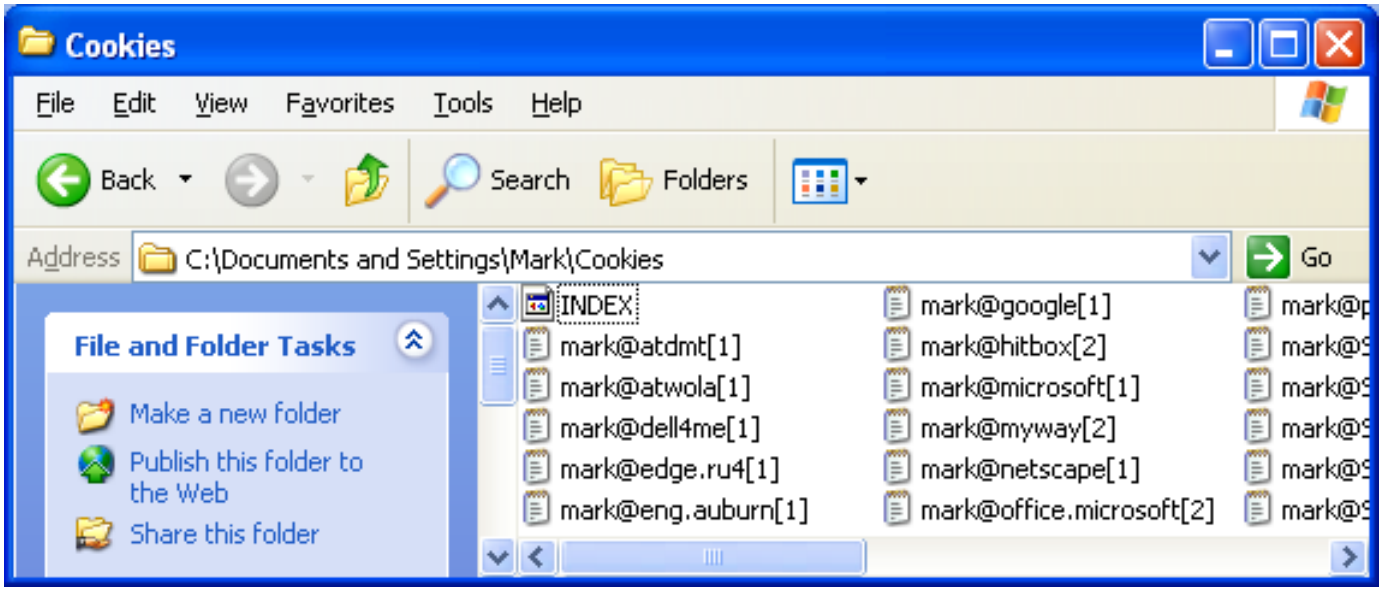
- Marketers use cookies to determine the effectiveness of advertising campaigns.
- Web sites store cookies on users' hard drives, which raises issues regarding security and privacy. Web sites should not store critical information, such as credit-card numbers or passwords, in cookies, because cookies are just text files that anyone can read.
- Several cookie features address security and privacy concerns. A server can access only the cookies that it has placed on the client.
- A cookies has an expiration date, after which the Web browser deletes it.

Cookies (cont.)

- Users who are concerned about the privacy and security implications of cookies can disable them in their Web browsers. However, the disabling of cookies can make it impossible for the user to interact with Web sites that rely on cookies to function properly.
- Information stored in the cookie is sent to the Web server from which it originated whenever the user requests a Web page from that particular server. The Web server can send the client XHTML output that reflects the preferences or information that is stored in the cookie.
- The location of the cookie file varies from browser to browser. Internet Explorer places cookies in the Cookies directory located at `C:\Documents and Settings\...\Cookies`

Cookies (cont.)

- After a cookie is created, a text file is added to this directory. While the name of the file will vary from user to user a typical example is shown below.



- The contents of a cookie are shown on page 74.

Cookies (cont.)

- Now let's create the code necessary to create our own cookie.
- In this example, a PHP script is invoked from a client-side HTML document. The HTML document creates a form for the user to enter the information that will be stored in the cookie. (Often the information that is stored in a cookie will be extracted from several different areas and may involved tracking the client's actions at the Web site.)
- Once the user has entered their information, when they click the Write Cookie button, the `cookies.php` script executes.
- The XHTML document and the PHP script are shown on the next pages. The XHTML document `cookies.html` is on page 36 and the PHP script `cookies.php` appears on page 37.

cookies.html – page 1

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<!-- cookies.html -->
<!-- Writing a Cookie        -->

<html xmlns = "http://www.w3.org/1999/xhtml">
  <head> <title>Writing a cookie to the client computer</title>   </head>

  <body style = "font-family: arial, sans-serif;
    background-color: #856363"  background=image1.jpg>
    <h2>Click Write Cookie to save your cookie data.</h2>

    <form method = "post" action = "cookies.php" style = "font-size: 10pt"
            background-color: #856363">
      <strong>Name:</strong><br />
      <input type = "text" name = "NAME" /><br />
      <strong>Height:</strong><br />
      <input type = "text" name = "HEIGHT" /><br />
      <strong>Favorite Color:</strong><br />
      <input type = "text" name = "COLOR" /><br />
      <p>
        <input type = "submit" value = "Write Cookie"  style = "background-color: #0000FF;
          color: yellow;  font-weight: bold" /></p>
    </form>
  </body>  </html>
```

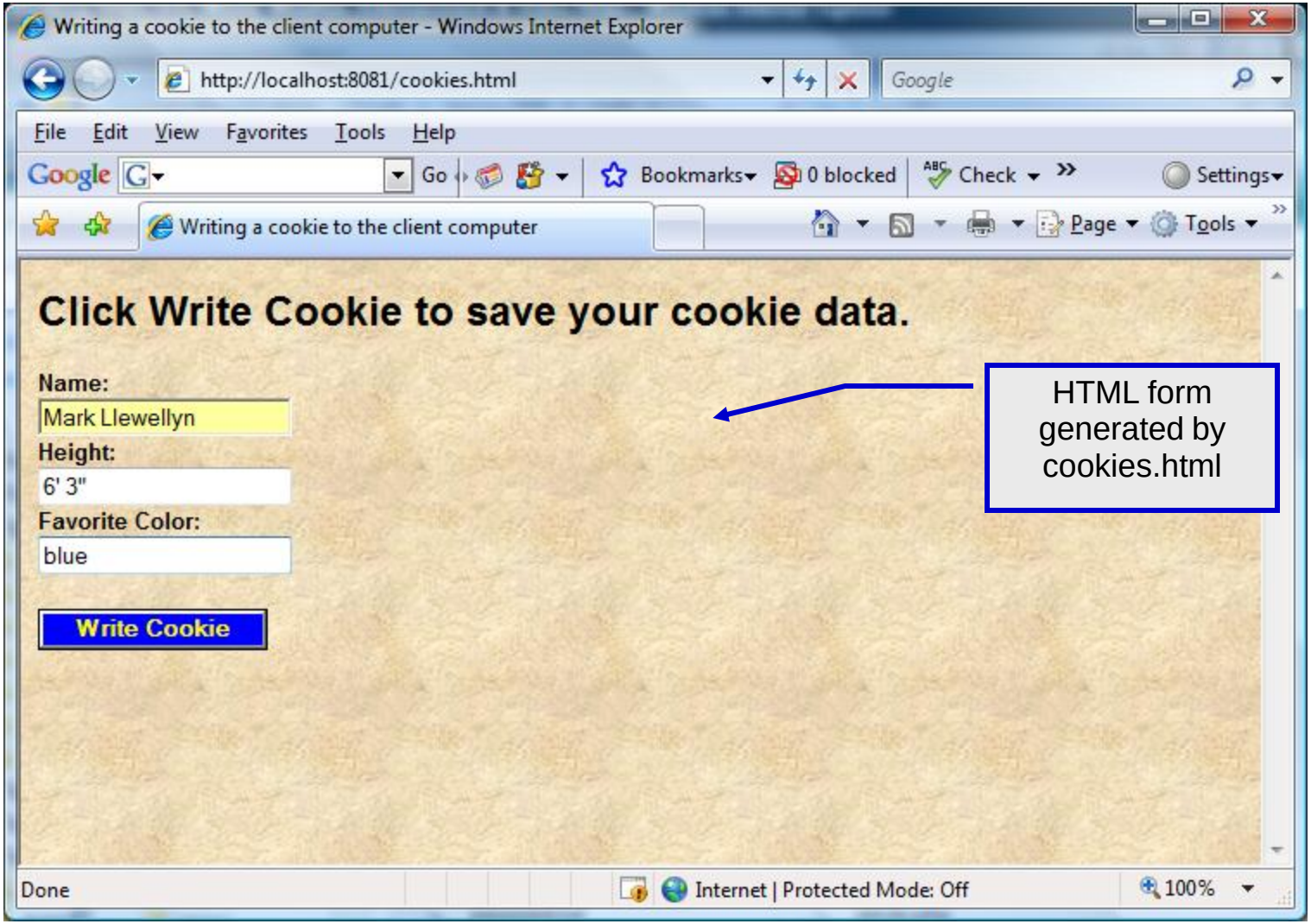
cookies.php – page 1

```
<?php
   // cookies.php
   // Program to write a cookie to a client's machine
    extract( $_POST );
   // write each form field's value to a cookie and set the
   // cookie's expiration date
   setcookie( "Name", $NAME, time() + 60 * 60 * 24 * 5 );
   setcookie( "Height", $HEIGHT, time() + 60 * 60 * 24 * 5 );
   setcookie( "Color", $COLOR, time() + 60 * 60 * 24 * 5 );
?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>      <title>Cookie Saved</title>   </head>
   <body style = "font-family: arial, sans-serif", background=image1.jpg>
      <p><b>The cookie has been set with the following data:</b></p>
      <!-- print each form field's value -->
      <br /><span style = "color: blue">Name:</span>
         <?php print( $NAME ) ?><br />
      <span style = "color: blue">Height:</span>
         <?php print( $HEIGHT ) ?><br />
      <span style = "color: blue">Favorite Color:</span>
      <span style = "color: <?php print( "$COLOR\">$COLOR" ) ?>
      </span><br />
      <p>Click <a href = "readCookies.php">here</a>  to read the saved cookie.</p>
   </body>  </html>
```

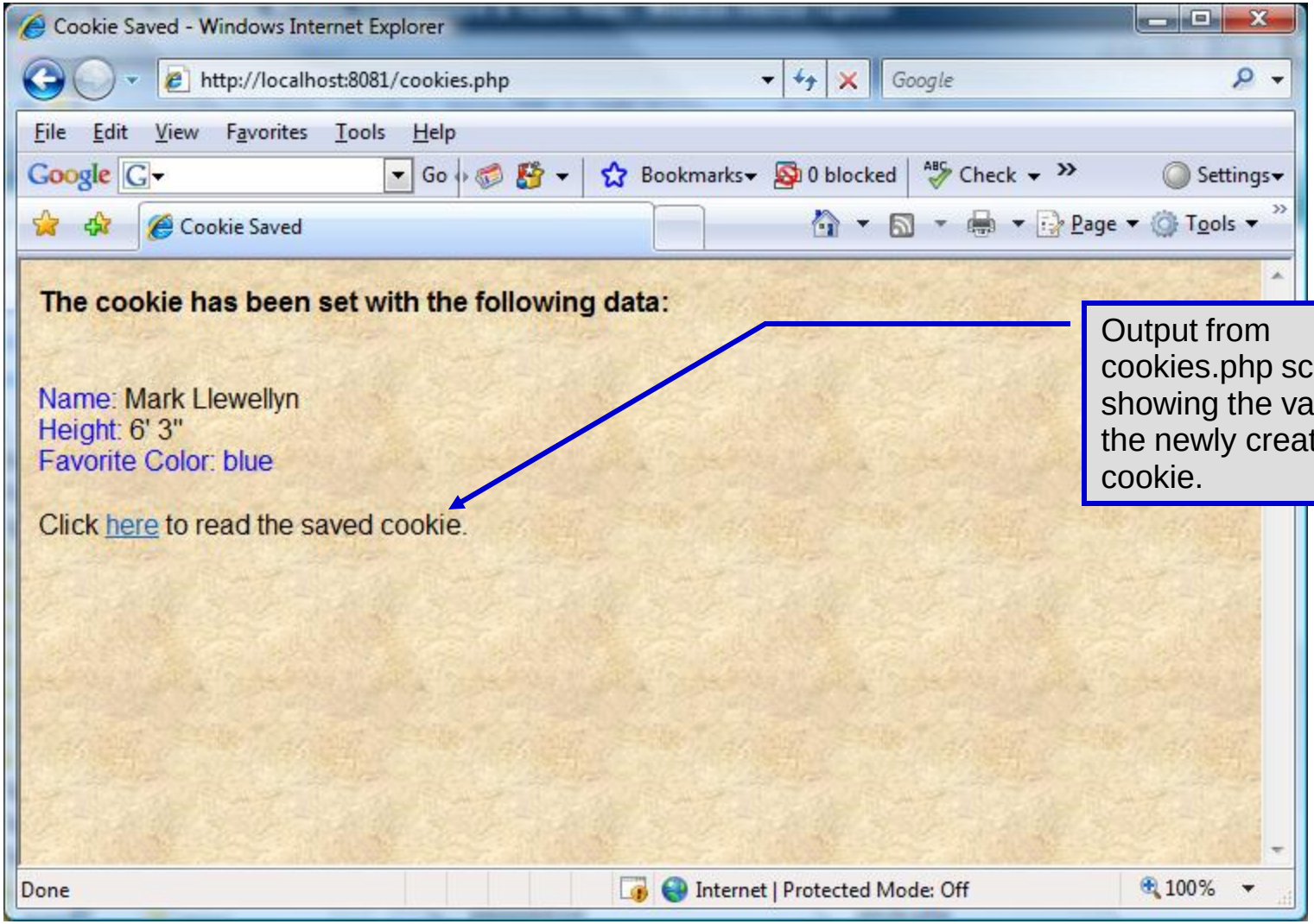
Function setcookie sets the cookies to the values passed from the cookies.html form. Function setcookie prints XHTML header information and therefore it needs to be called before any other XHTML (including comments) is printed.

The third argument to setcookie is optional and indicates the expiration date of the cookie. In this case it is set to expire 5 days from the current time. Function time returns the current time and then we add to this the number of seconds after which the cookie is to expire.

Cookies (cont.)



Cookies (cont.)



Output from cookies.php script showing the values in the newly created cookie.

Cookies (cont.)

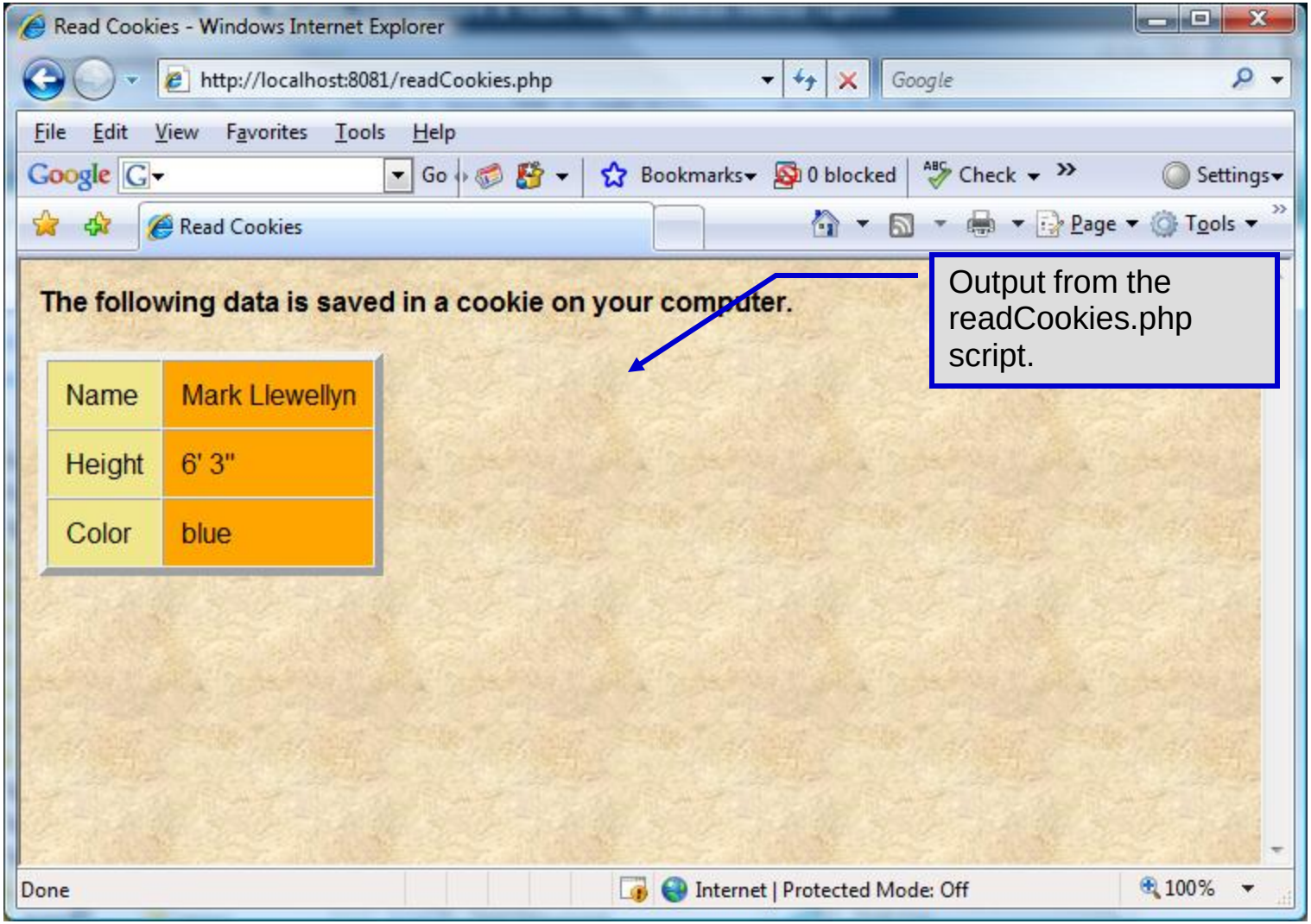
- Once the cookie has been created, the cookies.php script gives the user the chance to view the newly created cookie by invoking the readCookies.php script from within the cookies.php script by clicking on the link.
- The readCookies.php script code is illustrated on the next page followed by the output from the execution of this PHP script.

readCookies.php – page 1

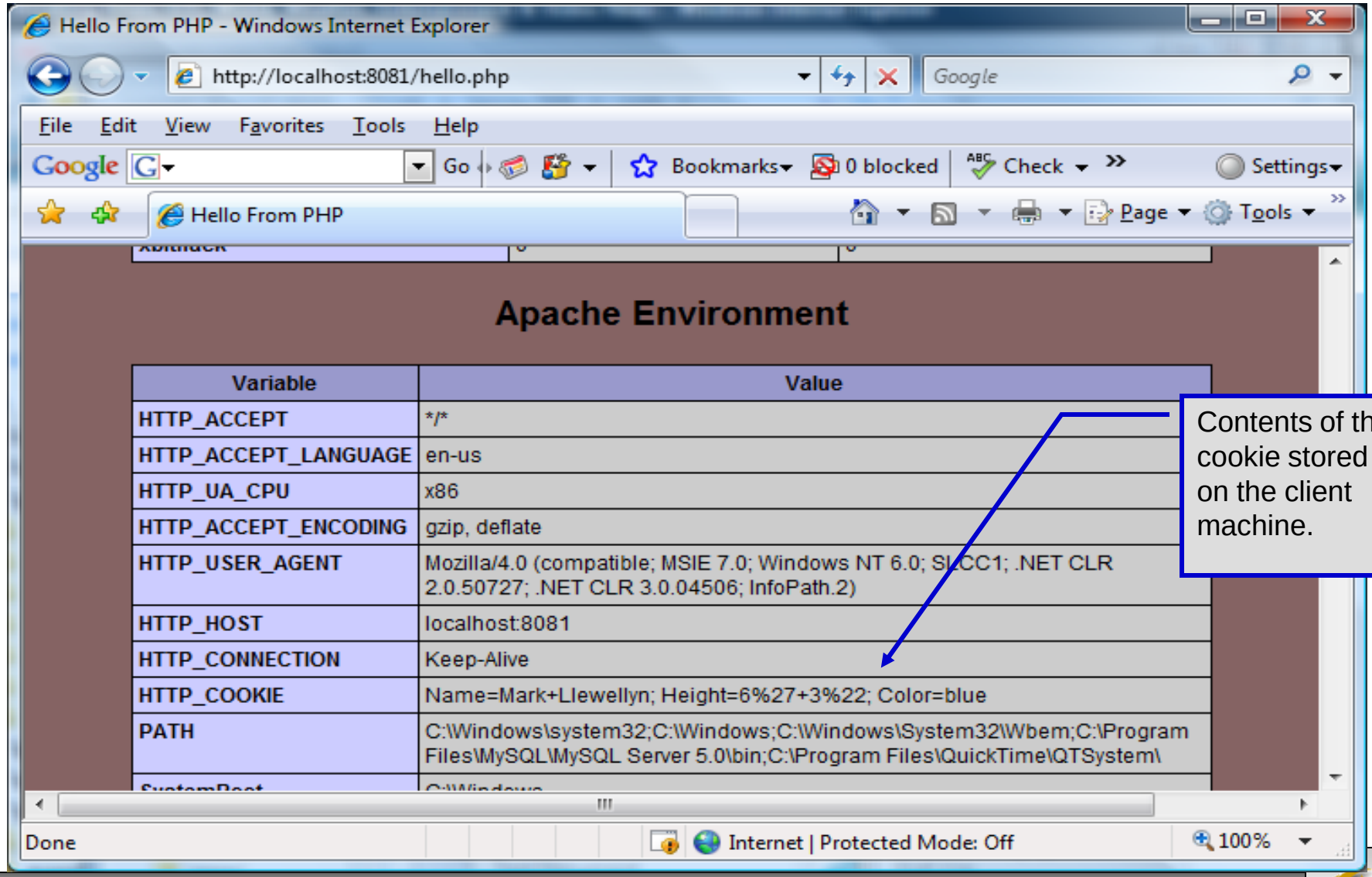
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<!-- readCookies.php                    -->
<!-- Program to read cookies from the client's computer -->

<html xmlns = "http://www.w3.org/1999/xhtml">
   <head><title>Read Cookies</title></head>

   <body style = "font-family: arial, sans-serif" background=image1.jpg>
      <p>
         <strong>   The following data is saved in a cookie on your computer.
        </strong>
      </p>
      <table border = "5" cellspacing = "0" cellpadding = "10">

         <?php
            // iterate through array $_COOKIE and print
            // name and value of each cookie
            foreach ( $_COOKIE as $key => $value )
               print( "<tr>
                  <td bgcolor=\"#F0E68C\">$key</td>
                  <td bgcolor=\"#FFA500\">$value</td>
                  </tr>" );
         ?>
      </table>
   </body>  </html>
```

Superglobal array holding cookie.

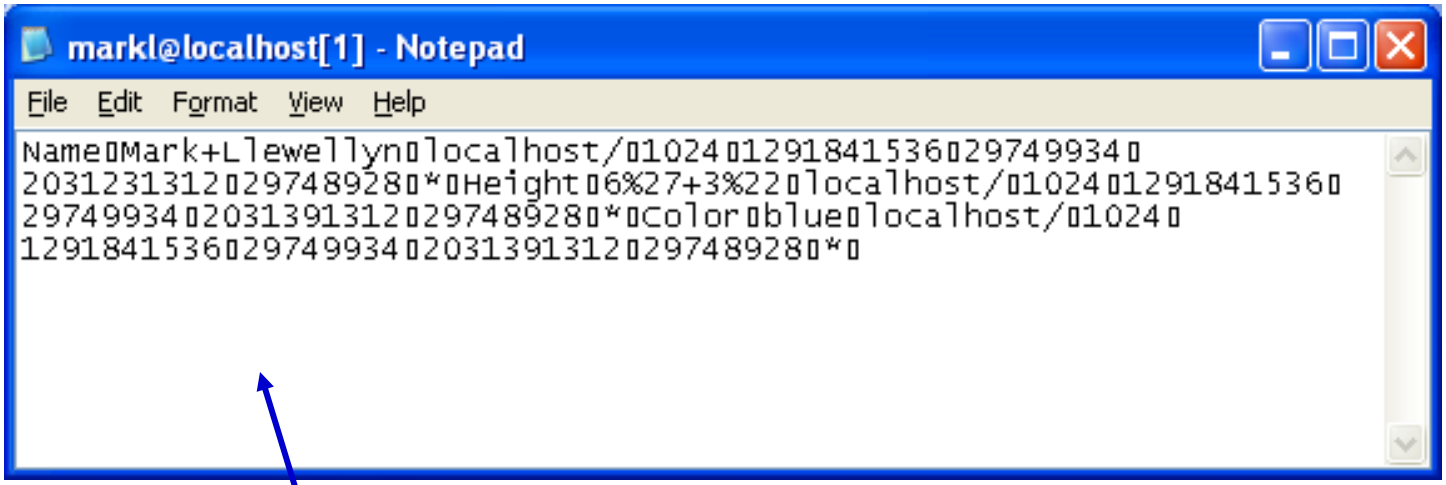
Cookies (cont.)



Cookies (cont.)



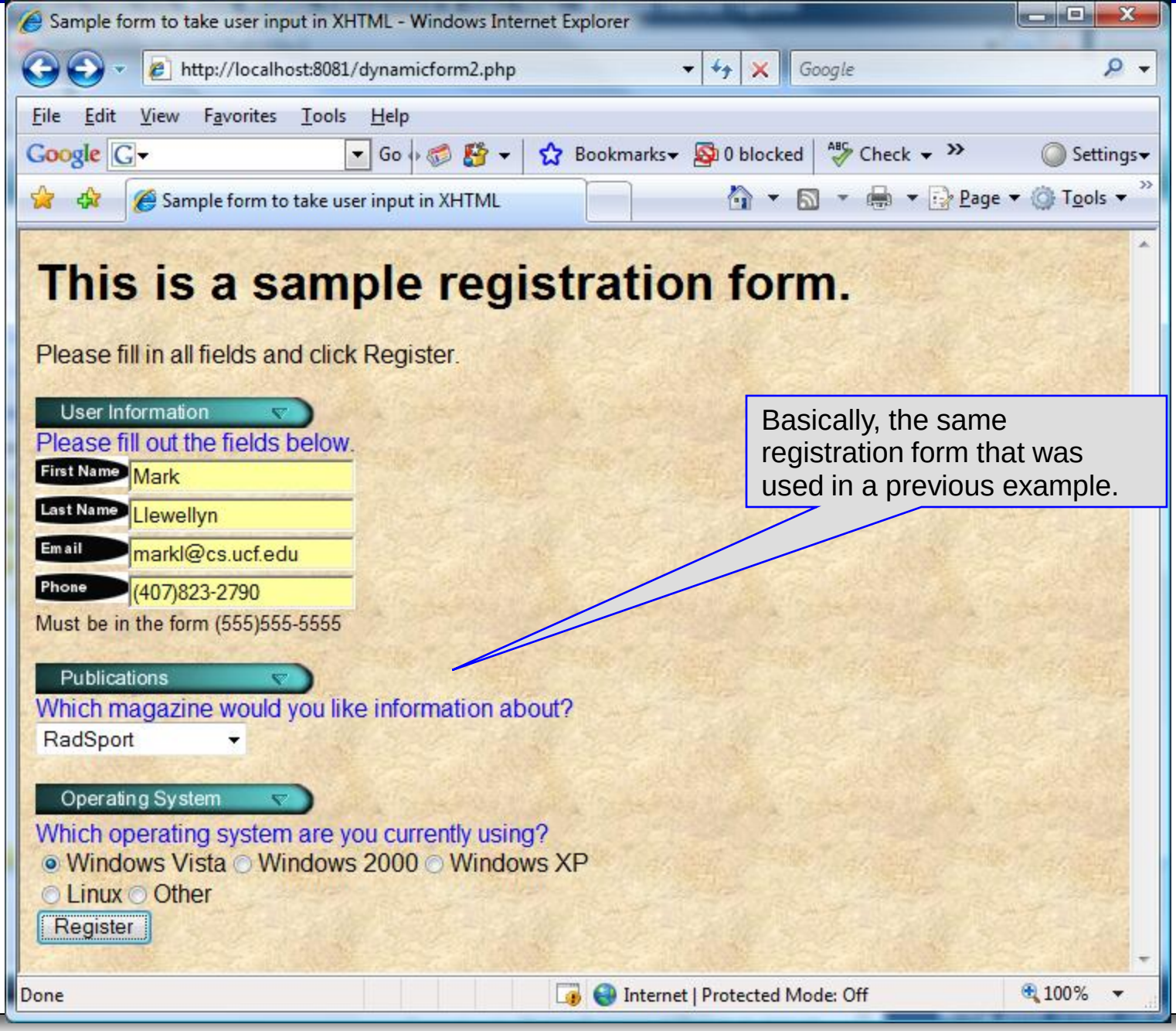
Cookies (cont.)

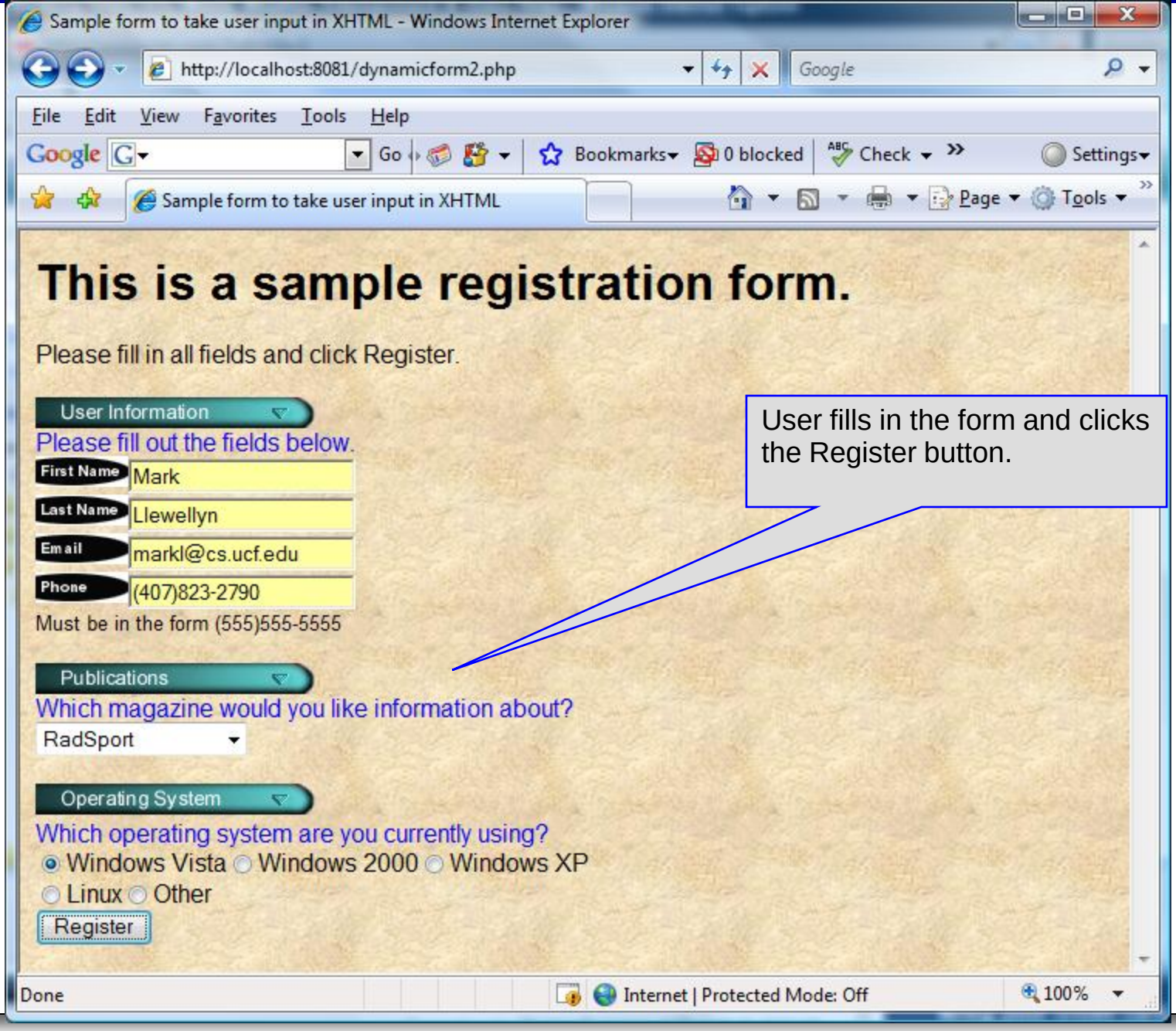


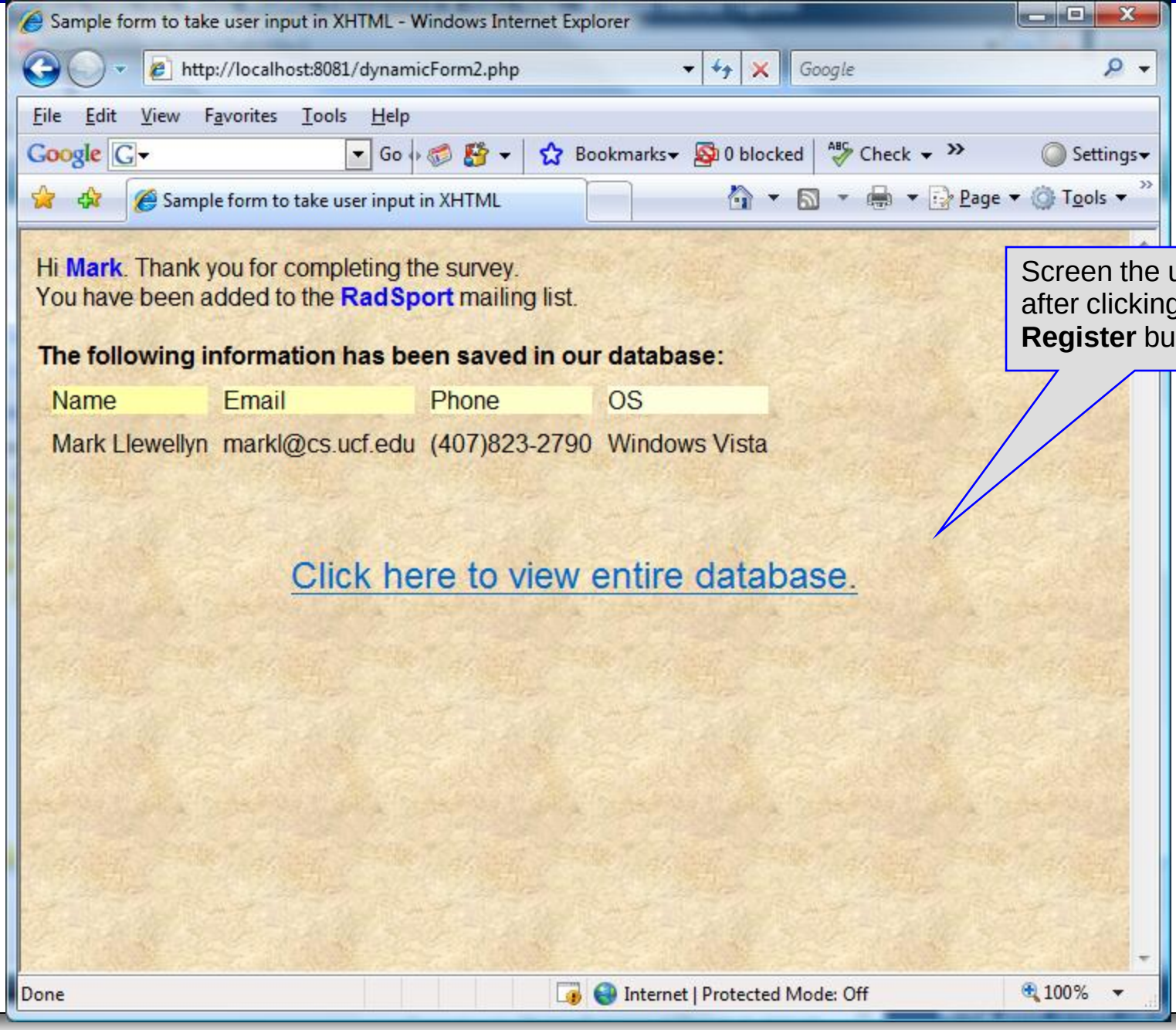
Actual text file holding cookie data for the cookie that was created in this example.

Dynamic Content in PHP

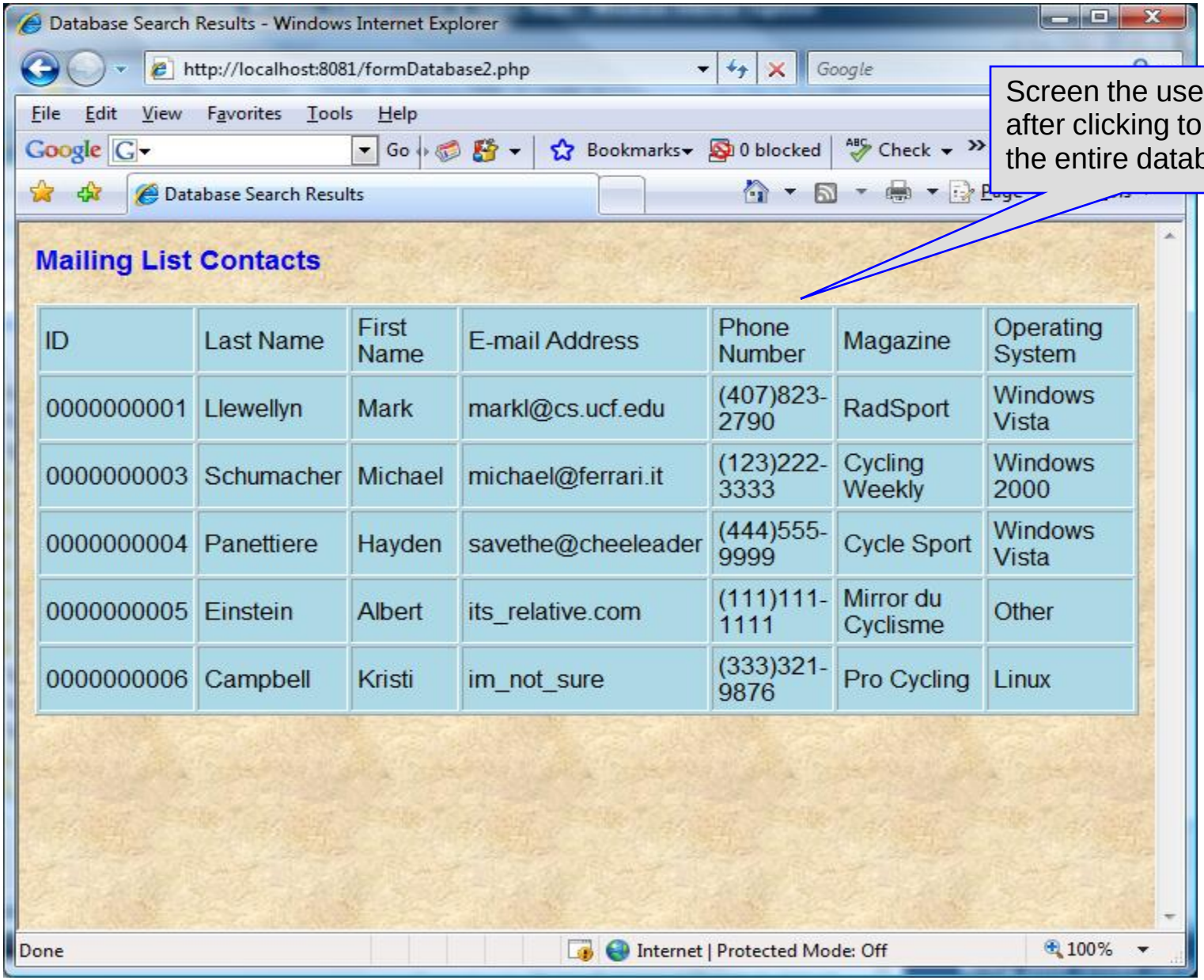
- Of all the strengths PHP exhibits as a server-side scripting language, perhaps its greatest strength lies in its ability to dynamically change XHTML output based on user input.
- In this final section of notes, we'll build on the examples we've constructed in the previous two sets of notes by combining `form.html` and `form.php` into one dynamic PHP document named `dynamicForm2.php`.
- We'll add error checking to the user input fields and inform the user of invalid entries on the form itself, rather than on an error page. If an error exists, the script maintains the previously submitted values in each form element.
- Finally, after the form has been successfully completed, we'll store the input from the user in a MySQL database.


Sample form to take user input in XHTML - Windows Internet Explorer
http://localhost:8081/dynamicform2.php
Google
File Edit View Favorites Tools Help
Google Go Bookmarks 0 blocked Check Settings
Sample form to take user input in XHTML
Page Tools
This is a sample registration form.
Please fill in all fields and click Register.
User Information
Please fill out the fields below.
First Name Mark
Last Name Llewellyn
Email markl@cs.ucf.edu
Phone (407)823-2790
Must be in the form (555)555-5555
Publications
Which magazine would you like information about?
RadSport
Operating System
Which operating system are you currently using?
Windows Vista Windows 2000 Windows XP
Linux Other
Register
Basically, the same registration form that was used in a previous example.
Done
Internet | Protected Mode: Off
100%

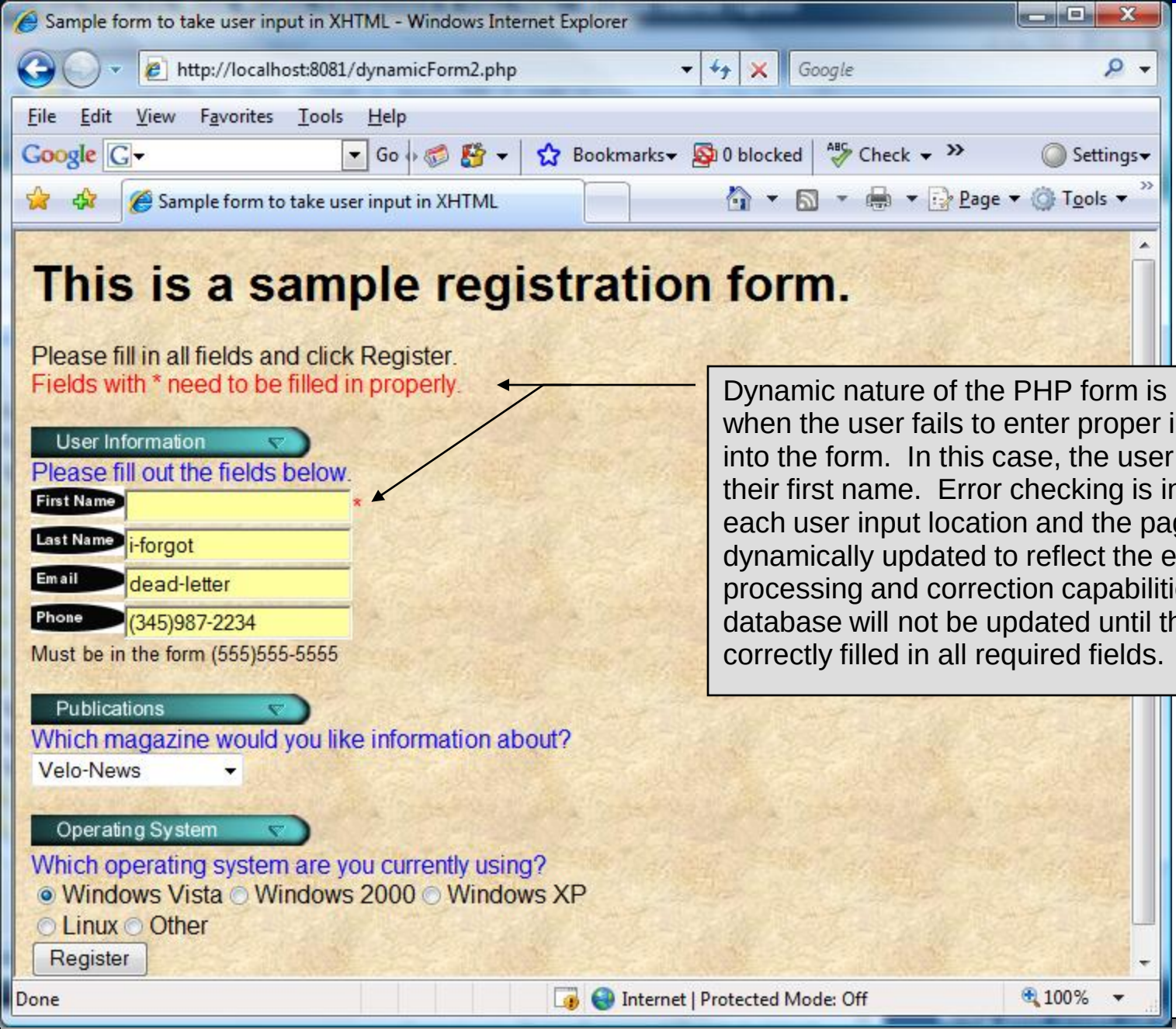

Sample form to take user input in XHTML - Windows Internet Explorer
http://localhost:8081/dynamicform2.php
Google
File Edit View Favorites Tools Help
Google Go Bookmarks 0 blocked Check Settings
Sample form to take user input in XHTML
Page Tools
This is a sample registration form.
Please fill in all fields and click Register.
User Information
Please fill out the fields below.
First Name Mark
Last Name Llewellyn
Email markl@cs.ucf.edu
Phone (407)823-2790
Must be in the form (555)555-5555
Publications
Which magazine would you like information about?
RadSport
Operating System
Which operating system are you currently using?
Windows Vista Windows 2000 Windows XP
Linux Other
Register
User fills in the form and clicks the Register button.
Done
Internet | Protected Mode: Off
100%



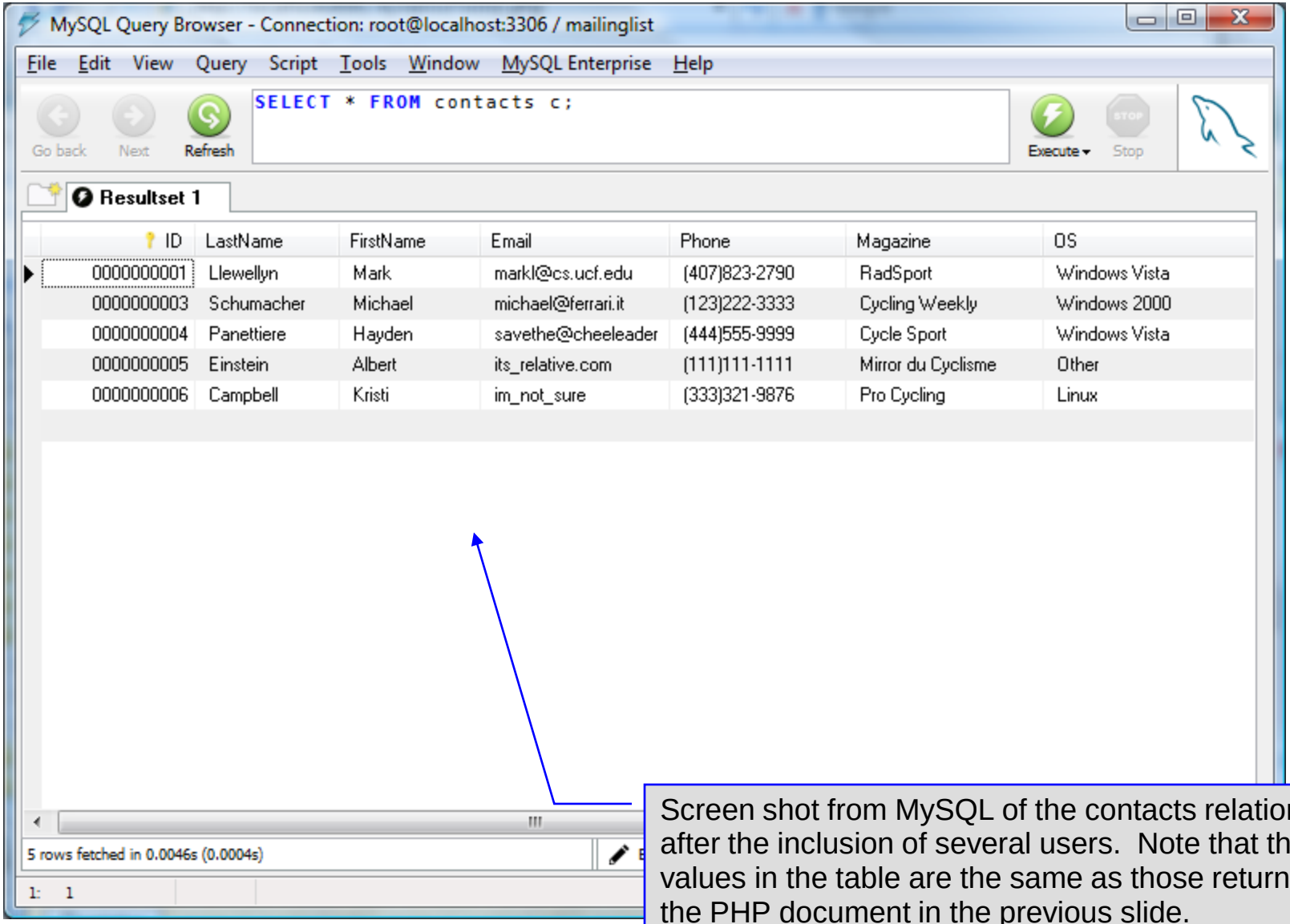
Screen the user sees after clicking the **Register** button.



Screen the user sees after clicking to see the entire database.



Dynamic nature of the PHP form is illustrated when the user fails to enter proper information into the form. In this case, the user forgot to enter their first name. Error checking is in place on each user input location and the page is dynamically updated to reflect the error processing and correction capabilities. The database will not be updated until the user has correctly filled in all required fields.



Screen shot from MySQL of the contacts relation after the inclusion of several users. Note that the values in the table are the same as those returned to the PHP document in the previous slide.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<!-- dynamicForm2.php          -->

<html xmlns = "http://www.w3.org/1999/xhtml">
  <head>
    <title>Sample form to take user input in XHTML</title>
</head>
<body style = "font-family: arial, sans-serif;  background-color: #856363"
background=background.jpg>
  <?php
    extract ( $_POST );
    $iserror = false;
    // array of magazine titles
    $maglist = array( "Velo-News",
      "Cycling Weekly",
      "Pro Cycling",
      "Cycle Sport",
              "RadSport",
              "Mirror du Cyclisme" );
    // array of possible operating systems
    $systemlist = array( "Windows XP",
      "Windows 2000",
      "Windows 98",
      "Linux",
      "Other");
```

```
// array of name and alt values for the text input fields
      $inputlist = array( "fname" => "First Name",
         "lname" => "Last Name",
         "email" => "Email",
         "phone" => "Phone" );
      if ( isset ( $submit ) ) {
         if ( $fname == "" ) {
            $formerrors[ "fnameerror" ] = true;
            $iserror = true;
         }
         if ( $lname == "" ) {
            $formerrors[ "lnameerror" ] = true;
            $iserror = true;
         }
         if ( $email == "" ) {
            $formerrors[ "emailerror" ] = true;
            $iserror = true;
         }
         if ( !ereg( "^\([0-9]{3}\)[0-9]{3}-[0-9]{4}$", $phone ) ) {
            $formerrors[ "phoneerror" ] = true;
            $iserror = true;
         }
         if ( !$iserror ) {
            // build INSERT query
            $query = "INSERT INTO contacts " .
               "(ID, LastName, FirstName, Email, Phone, Magazine, OS ) " .
               "VALUES (null, '$lname', '$fname', '$email', " .  "'" . quotemeta( $phone ) . "', '$mag', '$os' )";
```

```
// Connect to MySQL
 if ( !( $database = mysql_connect( "localhost",
    "root", "root" ) ) )
    die( "Could not connect to database" );

 // open MailingList database
 if ( !mysql_select_db( "MailingList", $database ) )
    die( "Could not open MailingList database" );

 // execute query in MailingList database
 if ( !( $result = mysql_query( $query, $database ) ) ) {
    print( "Could not execute query! <br />" );
    die( mysql_error() );
 }
 print( "<p>Hi
    <span style = 'color: blue'> <strong>$fname</strong></span>.
    Thank you for completing the survey.<br />
    You have been added to the <span style = 'color: blue'>
    <strong>$mag</strong></span> mailing list.              </p>
    <strong>The following information has been saved in our database:</strong><br />

    <table border = '0' cellpadding = '0' cellspacing = '10'>
    <tr>
    <td bgcolor = '#ffffaa'>Name </td>
    <td bgcolor = '#ffffbb'>Email</td>
    <td bgcolor = '#ffffcc'>Phone</td>
    <td bgcolor = '#ffffdd'>OS</td>
    </tr>
    <tr>
```

```
        <!-- print each form field's value -->
        <td>$fname $lname</td>
        <td>$email</td>
        <td>$phone</td>
        <td>$os</td>
        </tr></table>
        <br /><br /><br />
        <div style = 'font-size : 10pt; text-align: center'>
                                        <div style = 'font-size : 18pt'>
                                        <a href = 'formDatabase2.php'>
                                        Click here to view entire database.</a>
                                        </div>
          </div></body></html>" );
      die();
    }
}
print( "<h1>This is a sample registration form.</h1>
   Please fill in all fields and click Register." );
if ( $iserror ) {
   print( "<br /><span style = 'color : red'>
      Fields with * need to be filled in properly.</span>" );
}
print( "<!-- post form data to dynamicForm2.php -->
   <form method = 'post' action = 'dynamicForm2.php'>
   <img src = 'images/user.gif' alt = 'User' /><br />
   <span style = 'color: blue'>
   Please fill out the fields below.<br />
   </span>
```

Invoke PHP script to see contents of entire database if user clicks this link. Code begins on page 14.

The form created is self-submitting (i.e., it posts to itself). This is done by setting the action to dynamicForm2.php

```
<!-- create four text boxes for user input -->" );
 foreach ( $inputlist as $inputname => $inputalt ) {
    $inputtext = $inputvalues[ $inputname ];

    print( "<img src = 'images/$inputname.gif'
       alt = '$inputalt' /><input type = 'text'  name = '$inputname' value = '" . $$inputname . "' />" );
    if ( $formerrors[ ( $inputname )."error" ] == true )
       print( "<span style = 'color : red'>*</span>" );
    print( "<br />" );
 }
 print( "<span style = 'font-size : 10pt" );
 if ( $formerrors[ "phoneerror" ] )  print( "; color : red" );
 print( "'>Must be in the form (555)555-5555
    </span><br /><br />
    <img src = 'images/downloads.gif'
    alt = 'Publications' /><br />
    <span style = 'color: blue'>
    Which magazine would you like information about?
    </span><br />
    <!-- create drop-down list containing magazine names -->
    <select name = 'mag'>" );
 foreach ( $maglist as $currmag ) {
    print( "<option" );
    if ( ( $currmag == $mag ) )
       print( " selected = 'true'" );
    print( ">$currmag</option>" );
 }
```

The $$variable notation specifies variable variables. PHP permits the use of variable variables to allow developers to reference variables dynamically. The expression $$variable could also be written as ${$variable} for added clarity.

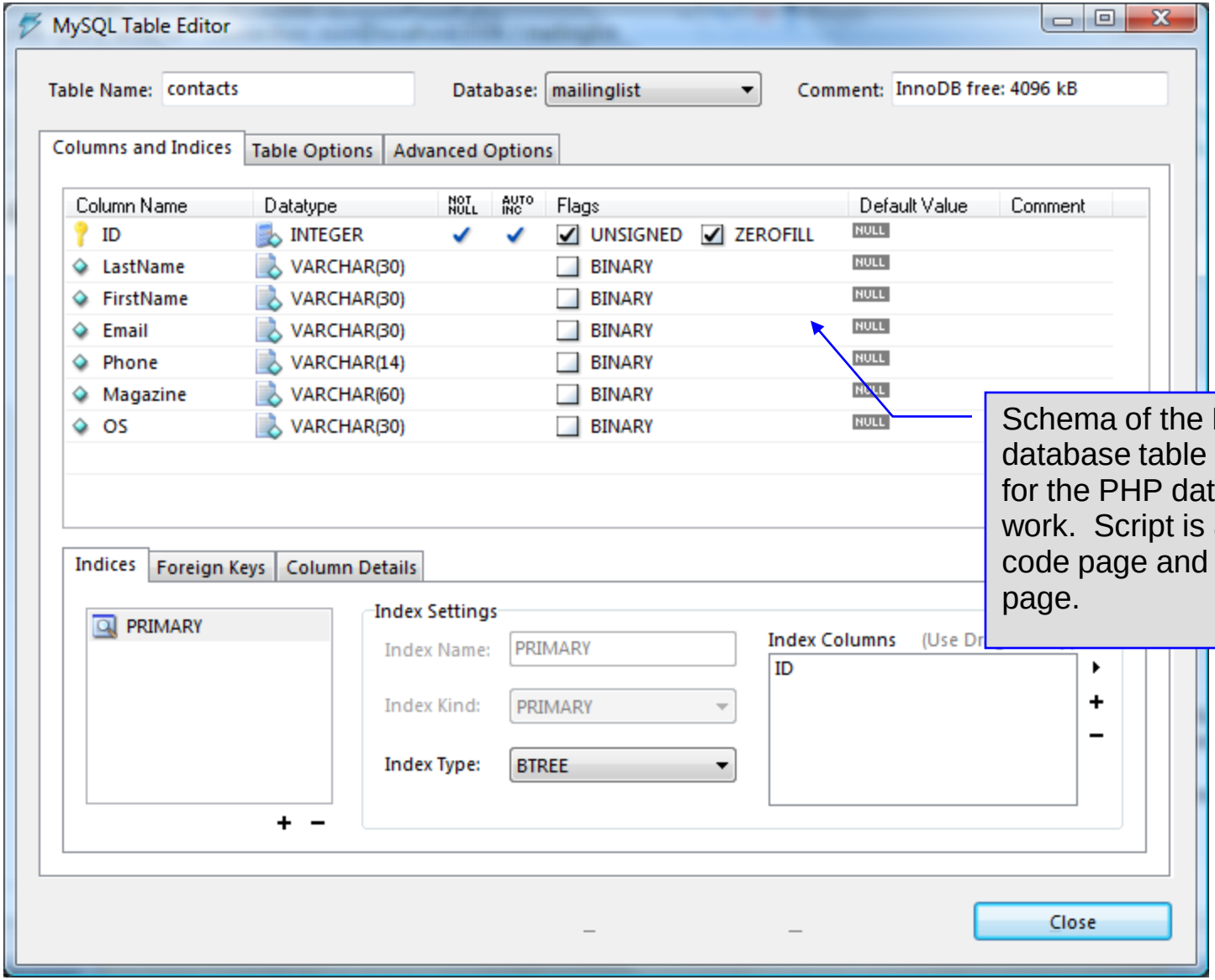
```
      print( "</select><br /><br />
      <img src = 'images/os.gif' alt = 'Operating System' />
      <br /><span style = 'color: blue'>
      Which operating system are you currently using?
      <br /></span>

      <!-- create five radio buttons -->" );

   $counter = 0;

   foreach ( $systemlist as $currsystem ) {
      print( "<input type = 'radio' name = 'os'
         value = '$currsystem'" );

      if ( $currsystem == $os ) print( "checked = 'checked'" );
      if ( iserror && $counter == 0 ) print( "checked = 'checked'" );

      print( " />$currsystem" );

      if ( $counter == 2 ) print( "<br />" );
      $counter++;
   }

   print( "<!-- create a submit button -->
      <br />
      <input type = 'submit' name = 'submit' value = 'Register' />
      </form></body></html>" );
?>
```

formDatabase2.php – page 1

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<!– formDatabase2.php        -->
<!-- Program to query a database and send results to the client.     -->

<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>      <title>Database Search Results</title>   </head>
   <body style = "font-family: arial, sans-serif"
      style = "background-color: #F0E68C" background=image1.jpg>
      <?php
         extract( $_POST );
         // build SELECT query
         $query = "SELECT * FROM contacts";
                // Connect to MySQL
         if ( !( $database = mysqli_connect( "localhost",  "root", "root“, MailingList ) ) )
            die( "Could not connect to database" );
          // query MailingList database
         if ( !( $result = mysqli_query( $database, $query ) ) ) {
            print( "Could not execute query! <br />" );
            die( mysqli_error() );
         }
      ?>
      <h3 style = "color: blue">
      Mailing List Contacts</h3>
```

formDatabase2.php – page 2

```
<table border = "1" cellpadding = "3" cellspacing = "2"
 style = "background-color: #ADD8E6">
 <tr>
    <td>ID</td>
    <td>Last Name</td>
    <td>First Name</td>
    <td>E-mail Address</td>
    <td>Phone Number</td>
    <td>Magazine</td>
    <td>Operating System</td>
 </tr>
 <?php
    // fetch each record in result set
    for ( $counter = 0;
       $row = mysqli_fetch_row( $result );
       $counter++ ){
       // build table to display results
       print( "<tr>" );
       foreach ( $row as $key => $value )
          print( "<td>$value</td>" );
       print( "</tr>" );
    }
    mysqli_close( $database );
 ?>

    </table>
  </body>
</html>
```



Schema of the MailingList database table contacts required for the PHP database example to work. Script is available on the code page and shown on the next page.

*C:\Courses\CNT 4714 - Enterprise Computing\Fall 2009\code\mailing list script.sql - Notepad++

File Edit Search View Format Language Settings Macro Run TextFX Plugins Window ?

MySQLServlet.java | SQLFormatter.java | mailing list script.sql

```
1  # SQL commands to create and populate the MySQL database for
2  # CNT 4714 - Fall 2009
3  #
4  # delete the database if it already exists
5  drop database if exists mailinglist;
6
7  #create a new database named mailinglist
8  create database mailinglist;
9
10 #switch to the new database
11 use mailinglist;
12
13 #create the schemas for the four relations in this database
14 create table contacts (
15     ID integer unsigned zerofill auto_increment not null,
16     LastName varchar(30),
17     FirstName varchar(30),
18     Email varchar(30),
19     Phone varchar(14),
20     Magazine varchar(60),
21     OS varchar(30),
22     primary key (ID)
23 );
```

Structured Query Langua | nb char : 622 | nb line : 26 | Ln : 2 | Col : 23 | Sel : 0 | Dos\Windows | ANSI | INS