CACS 205: Web Technology
Web Server

PREPARED BY KRISHNA PD. ACHARYA

(MECHI MULTIPLE CAMPUS)

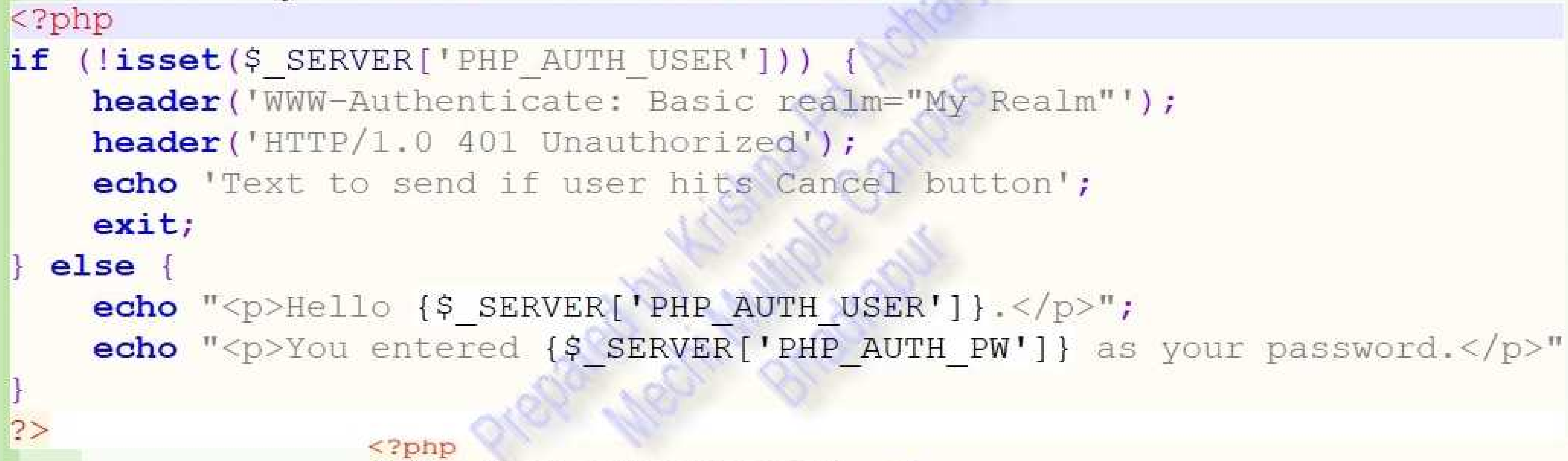
The Server tier 8 Hrs

- **Authentication**: Authentication is the process of determining the identity of a user based on the user's credentials. The user's credentials are usually in the form of user ID and password, which is checked against any credentials' store such as database. If the credentials provided by the user are valid, then the user is considered an authenticated user.
- **Authorization**: After successful authentication, deciding which resources a user can access based on their identity and checking whether the authenticated user has sufficient rights to access the requested resource is authorization.
- **Impersonation:** Impersonation is a process in which user accesses the resources(Ex:Files,DB) by using the identity of another user.
- There are four different kinds of Windows authentication options available that can be configured in IIS:
- **Anonymous Authentication**: IIS runs all the users' requests using the identity of the IUSR_machine name account which is created by IIS. This is an example of Impersonation wherein user accesses the resources by using the identity of the another user. IIS doesn't perform any authentication check. IIS allows any user to access the ASP .NET application.

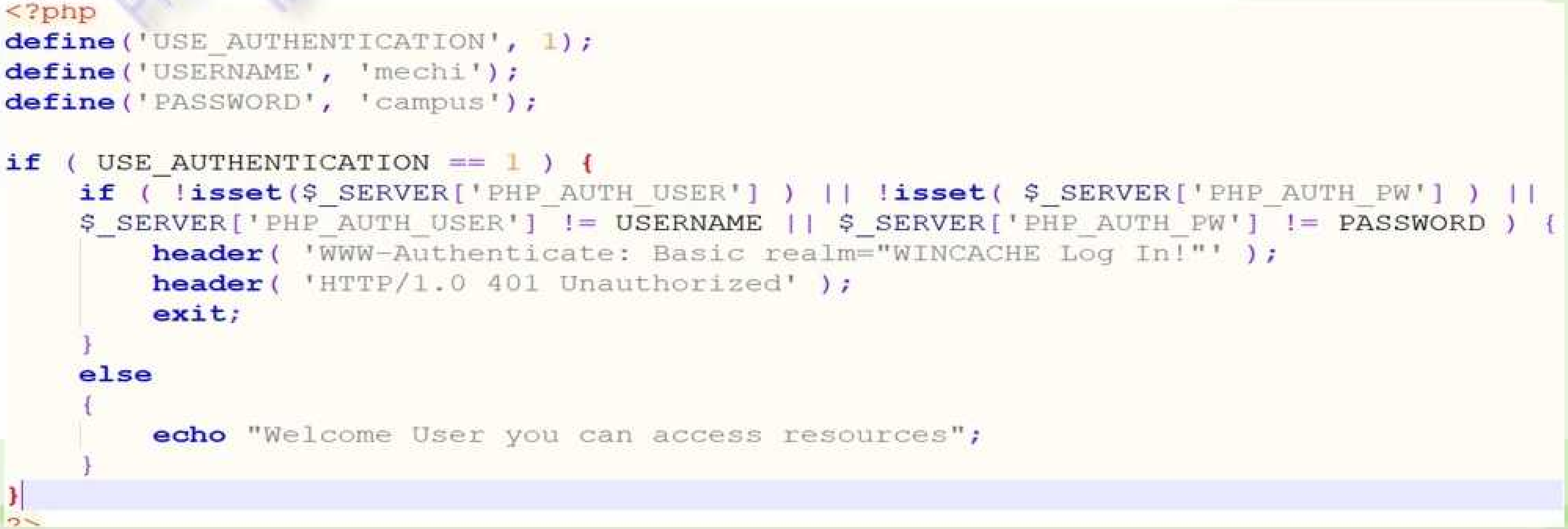
The Server tier 8 Hrs

- **Basic Authentication**: For this kind of authentication, a Windows user name and password have to be provided to connect. However, this information is sent over the network in plain text and hence this is an insecure kind of authentication. Basic Authentication is mostly supported by older, non-IE browsers.
- **Digest Authentication**: It is same as Basic Authentication but for the fact that the password is hashed before it is sent across the network. However, to be using Digest Authentication, we must use IE 5.0 or above.
- **Integrated Windows Authentication**: In this kind of authentication technique, passwords are not sent across the network. The application here uses either the kerberos or challenge/response protocols to authenticate users. Kerberos, a network authentication protocol, is designed to provide strong authentication for client-server applications. It provides the tools of authentication and strong cryptography over the network to help to secure information in systems across entire enterprise.

Anonymous Authentication



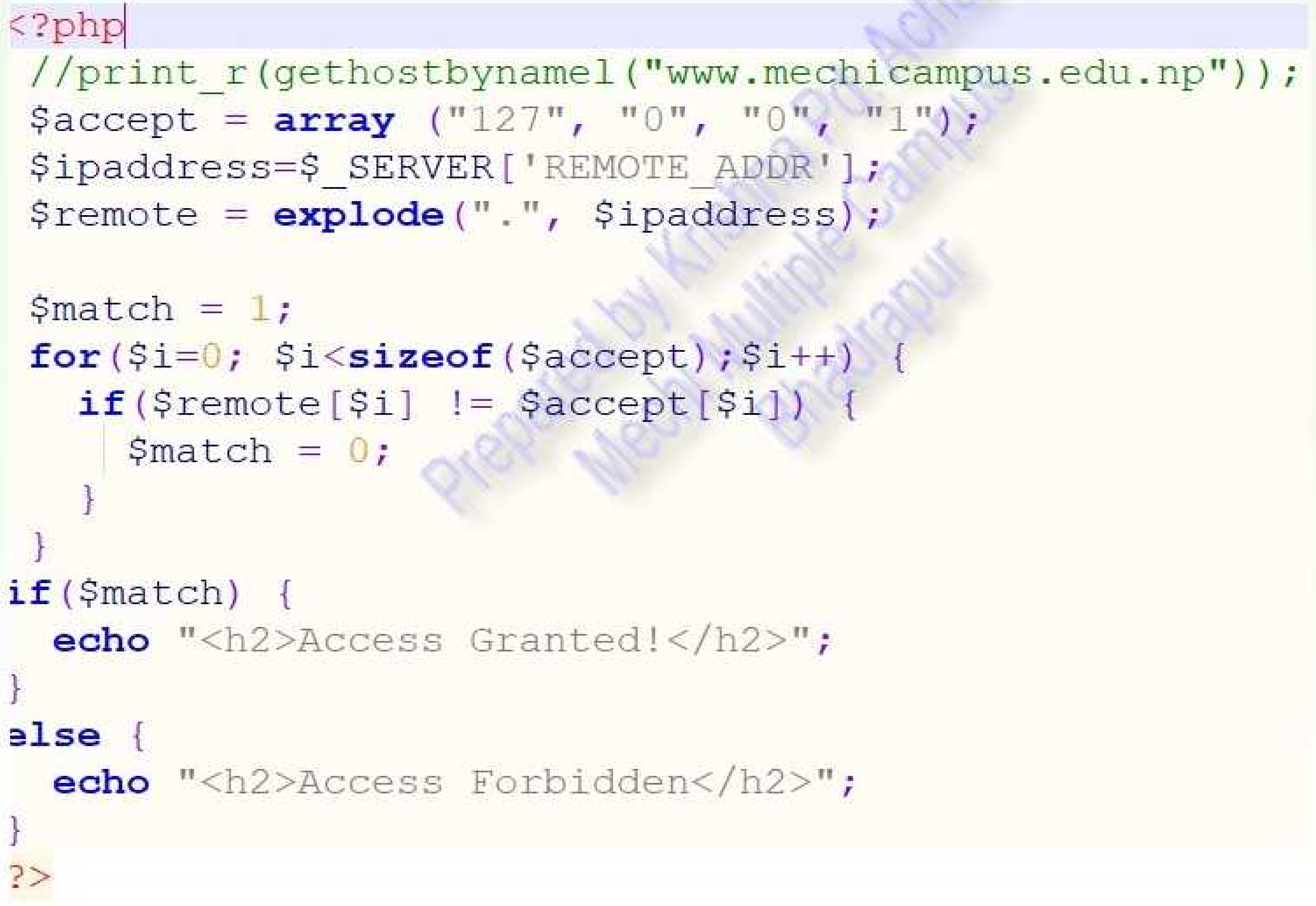
```php
<?php
if (!isset($_SERVER['PHP_AUTH_USER'])) {
    header('WWW-Authenticate: Basic realm="My Realm"');
    header('HTTP/1.0 401 Unauthorized');
    echo 'Text to send if user hits Cancel button';
    exit;
} else {
    echo "<p>Hello {$_SERVER['PHP_AUTH_USER']}.</p>";
    echo "<p>You entered {$_SERVER['PHP_AUTH_PW']} as your password.</p>"
}
?>
```



```php
<?php
define('USE_AUTHENTICATION', 1);
define('USERNAME', 'mechi');
define('PASSWORD', 'campus');

if ( USE_AUTHENTICATION == 1 ) {
    if ( !isset($_SERVER['PHP_AUTH_USER'] ) || !isset( $_SERVER['PHP_AUTH_PW'] ) ||
    $_SERVER['PHP_AUTH_USER'] != USERNAME || $_SERVER['PHP_AUTH_PW'] != PASSWORD ) {
        header( 'WWW-Authenticate: Basic realm="WINCACHE Log In!"' );
        header( 'HTTP/1.0 401 Unauthorized' );
        exit;
    }
    else
    {
        echo "Welcome User you can access resources";
    }
}
```

Authentication by IP Address and Domain



```php
<?php
 //print_r(gethostbyname1("www.mechicampus.edu.np"));
 $accept = array ("127", "0", "0", "1");
 $ipaddress=$_SERVER['REMOTE_ADDR'];
 $remote = explode(".", $ipaddress);

 $match = 1;
 for($i=0; $i<sizeof($accept);$i++) {
   if($remote[$i] != $accept[$i]) {
     $match = 0;
   }
 }
if($match) {
  echo "<h2>Access Granted!</h2>";
}
else {
  echo "<h2>Access Forbidden</h2>";
}
?>
```

Cookie

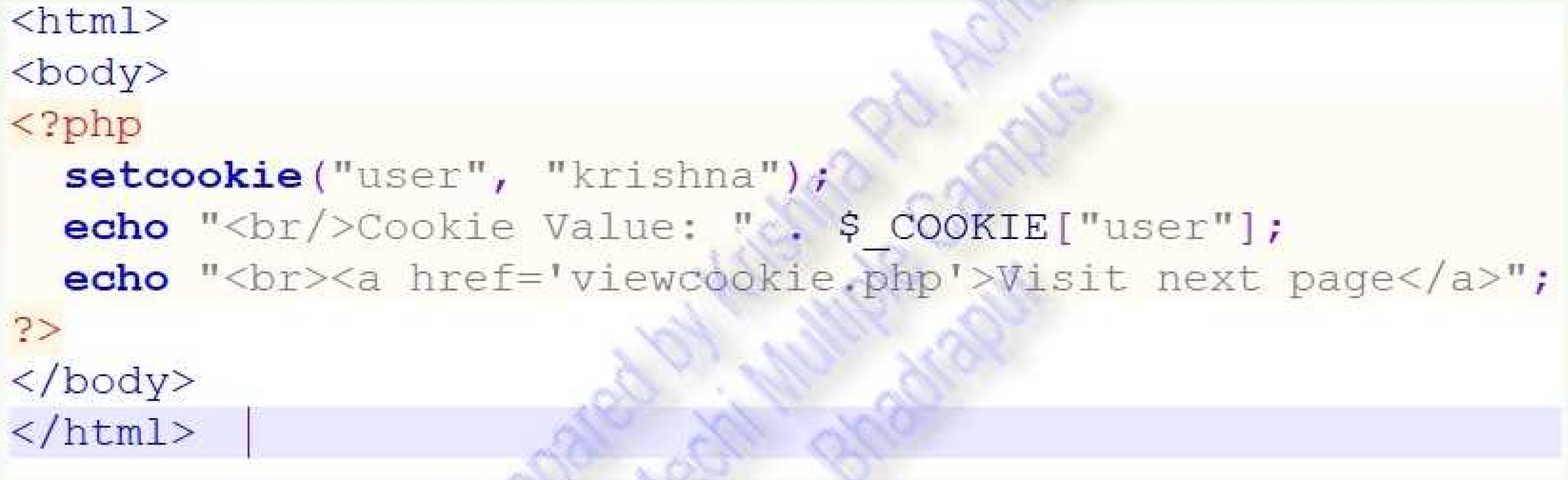
- A cookie is a small text file that lets us store a small amount of data (nearly 4KB) on the user's computer. They are typically used to keeping track of information such as username that the site can retrieve to personalize the page when user visit the website next time.
- Each time the browser requests a page to the server, all the data in the cookie is automatically sent to the server within the request.

```
setcookie(name, value, expire, path, domain, secure, httponly);
```

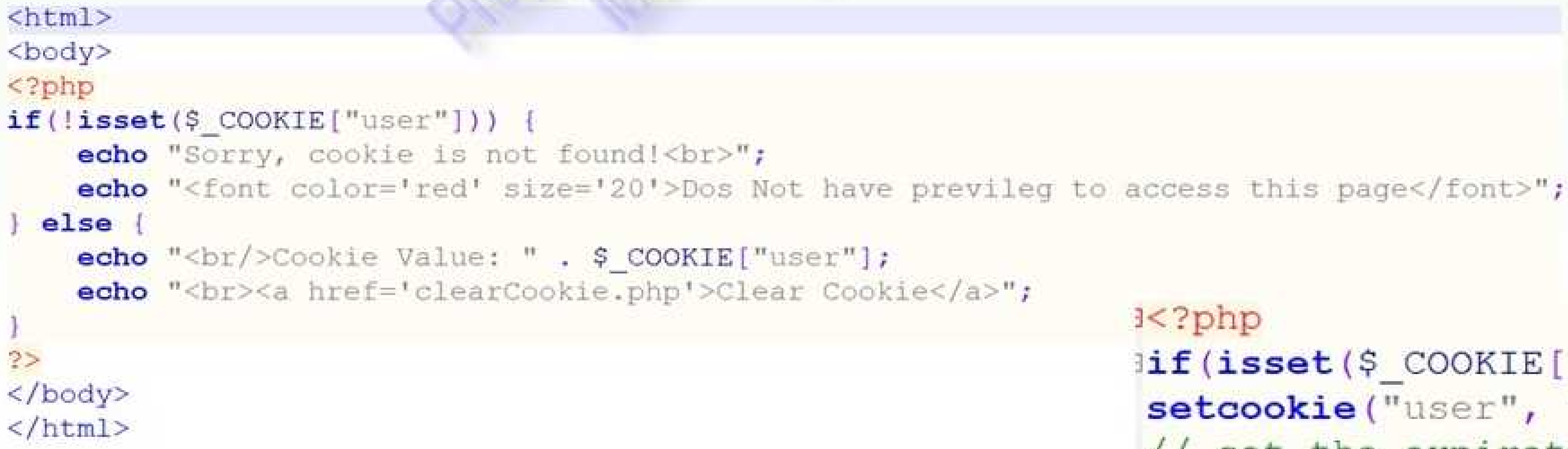
```
setcookie("CookieName", "CookieValue");/* defining name and value only*/
setcookie("CookieName", "CookieValue", time()+1*60*60);//using expiry in 1 hour(1*60*60 seconds or 3600 sec
setcookie("CookieName", "CookieValue", time()+1*60*60, "/mypath/", "mydomain.com", 1);
```

```
setcookie("user", "Administrator", time()+30*24*60*60);
```

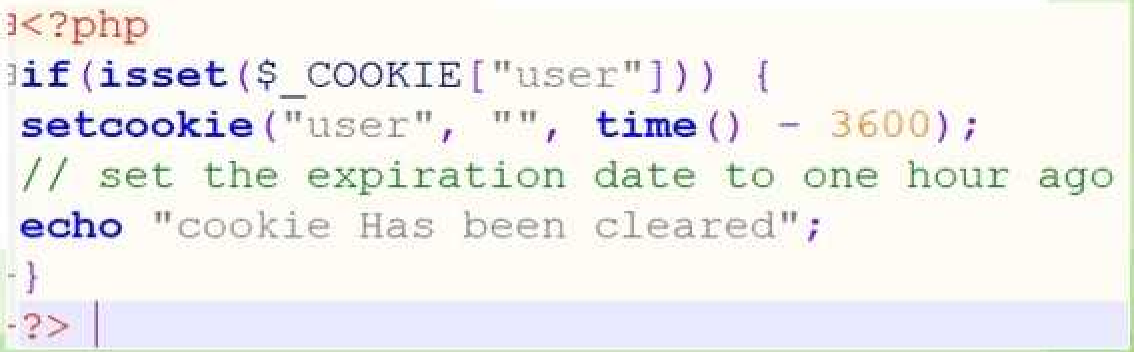
Cookie



```
<html>
<body>
<?php
  setcookie("user", "krishna");
  echo "<br/>Cookie Value: " . $_COOKIE["user"];
  echo "<br><a href='viewcookie.php'>Visit next page</a>";
?>
</body>
</html>
```



```
<html>
<body>
<?php
if(!isset($_COOKIE["user"])) {
    echo "Sorry, cookie is not found!<br>";
    echo "<font color='red' size='20'>Dos Not have previleg to access this page</font>";
} else {
    echo "<br/>Cookie Value: " . $_COOKIE["user"];
    echo "<br><a href='clearCookie.php'>Clear Cookie</a>";
}
?>
</body>
</html>
```



```
<?php
if(isset($_COOKIE["user"])) {
setcookie("user", "", time() - 3600);
// set the expiration date to one hour ago
echo "cookie Has been cleared";
}
?>
```

Cookie

Advantages of using cookies.

- Cookies are simple to use and implement.
- Occupies less memory, do not require any server resources and are stored on the user's computer so no extra burden on server.
- We can configure cookies to expire when the browser session ends (session cookies) or they can exist for a specified length of time on the client's computer (persistent cookies).
- Cookies persist a much longer period of time than Session state.

Disadvantages of using cookies

- As mentioned previously, cookies are not secure as they are stored in clear text they may pose a possible security risk as anyone can open and tamper with cookies. You can manually encrypt and decrypt cookies, but it requires extra coding and can affect application performance because of the time that is required for encryption and decryption
- Several limitations exist on the size of the cookie text(4kb in general), number of cookies(20 per site in general), etc.
- User has the option of disabling cookies on his computer from browser's setting .
- Cookies will not work if the security level is set to high in the browser.
- Users can delete a cookies.
- Users browser can refuse cookies, so your code has to anticipate that possibility.
- Complex type of data not allowed (e.g. dataset etc). It allows only plain text (i.e. cookie allows only string content)

Cookie Vs Session

Cookies vs Sessions

- Both cookies and sessions are used for storing persistent data. But there are few differences.
- Sessions are stored on server side. Cookies are on the client side.
- Sessions are closed when the user closes his browser. For cookies, you can set time that when it will be expired.
- Sessions are safer than cookies. Because, since stored on client's computer, there are ways to modify or manipulate cookies.

Application:

Session management

- Logins, shopping carts, game scores, or anything else the server should remember.

Personalization

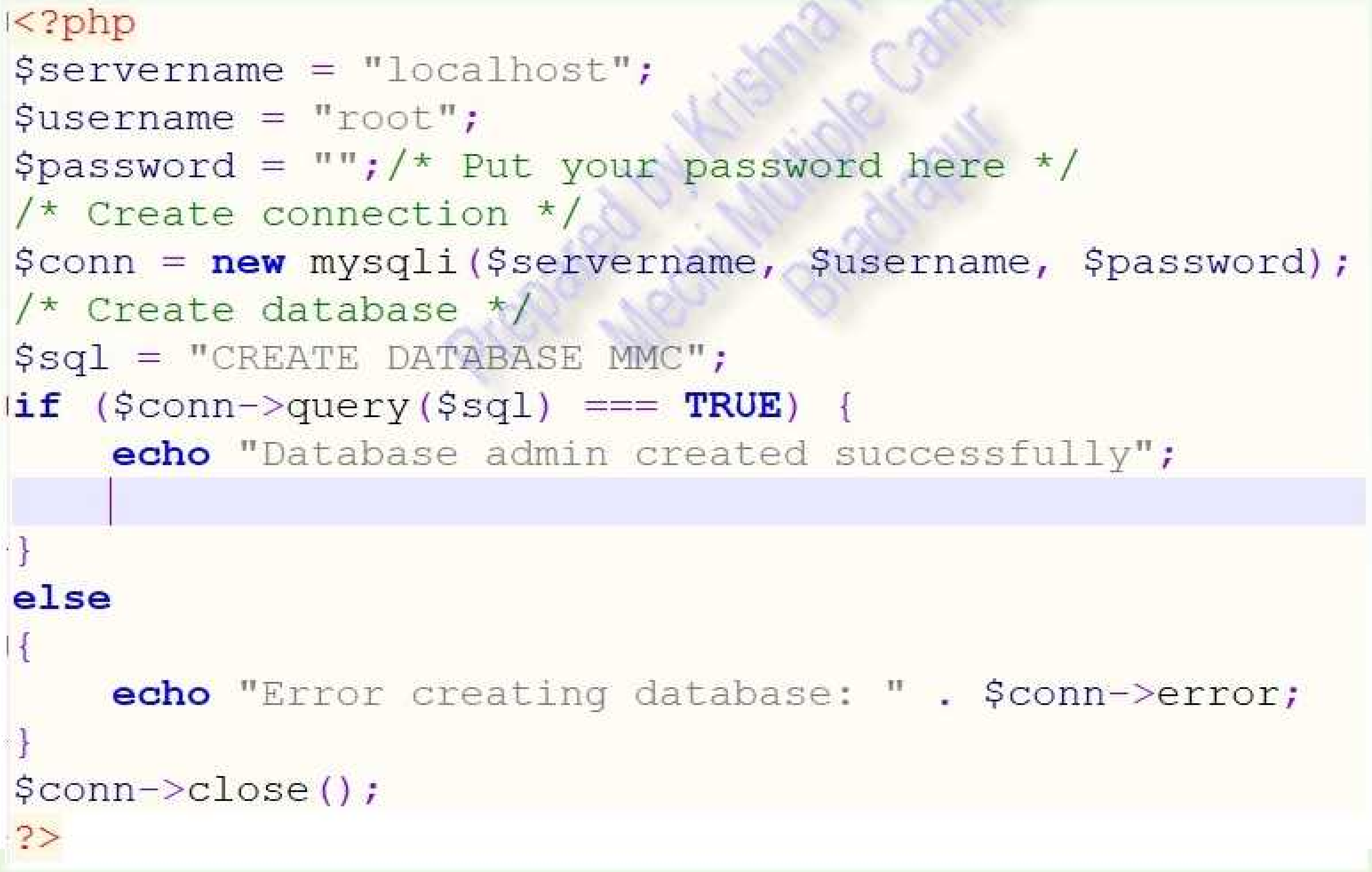
- User preferences, themes, and other settings.

Tracking

- Recording and analyzing user behavior.

Database Connectivity

Syntax: CREATE DATABASE <dbname>;
CREATE DATABASE mmc;



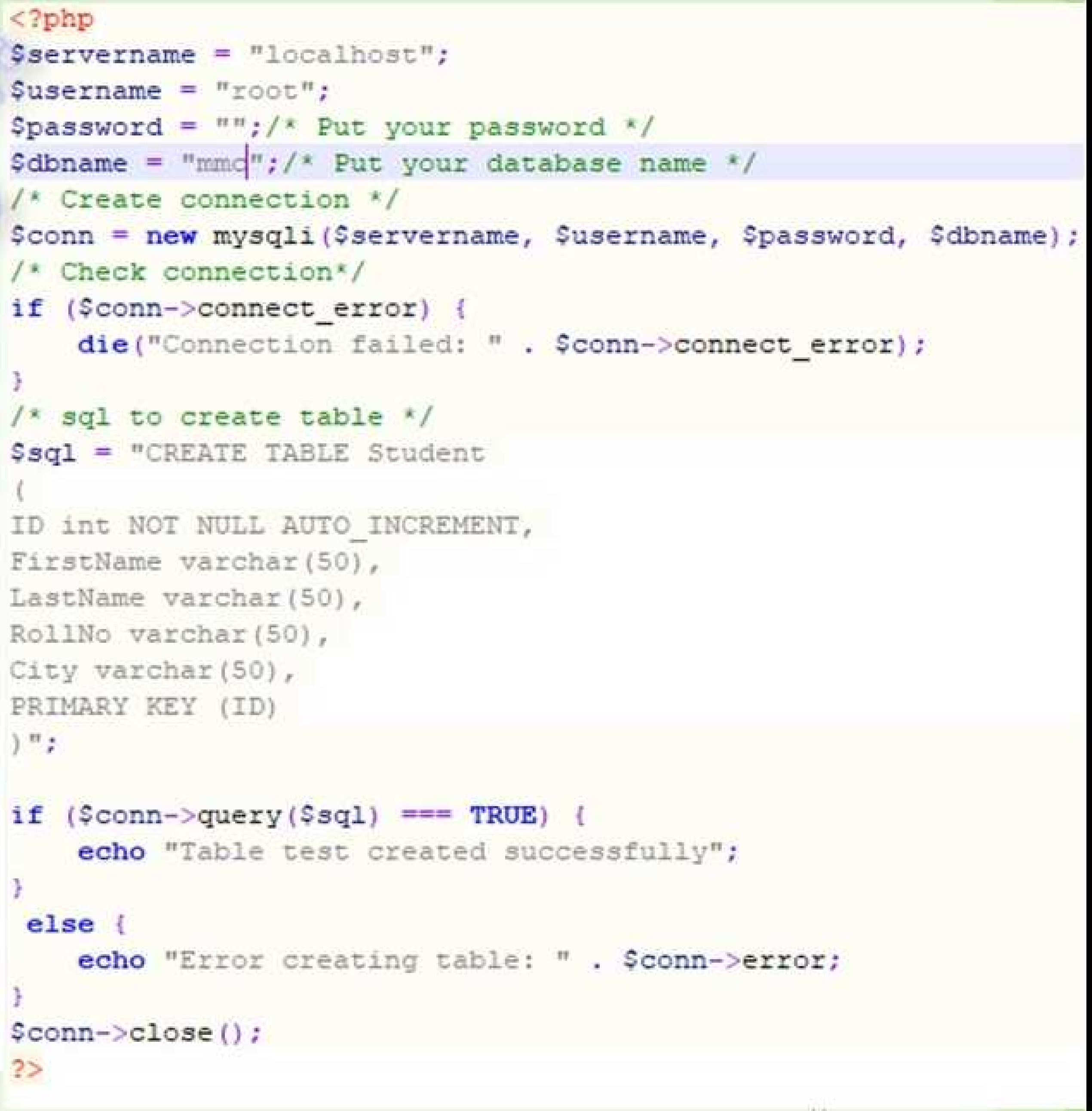
```php
<?php
$servername = "localhost";
$username = "root";
$password = "";/* Put your password here */
/* Create connection */
$conn = new mysqli($servername, $username, $password);
/* Create database */
$sql = "CREATE DATABASE MMC";
if ($conn->query($sql) === TRUE) {
    echo "Database admin created successfully";
    
}
else
{
    echo "Error creating database: " . $conn->error;
}
$conn->close();
?>
```

Database Connectivity

Student

ID	FirstName	LastName	Rollno	City
1	Ram	Rai	101	KTM

```
CREATE TABLE table_name (
    column1_name data_type constraints,
    column2_name data_type constraints,
    ....
);
```



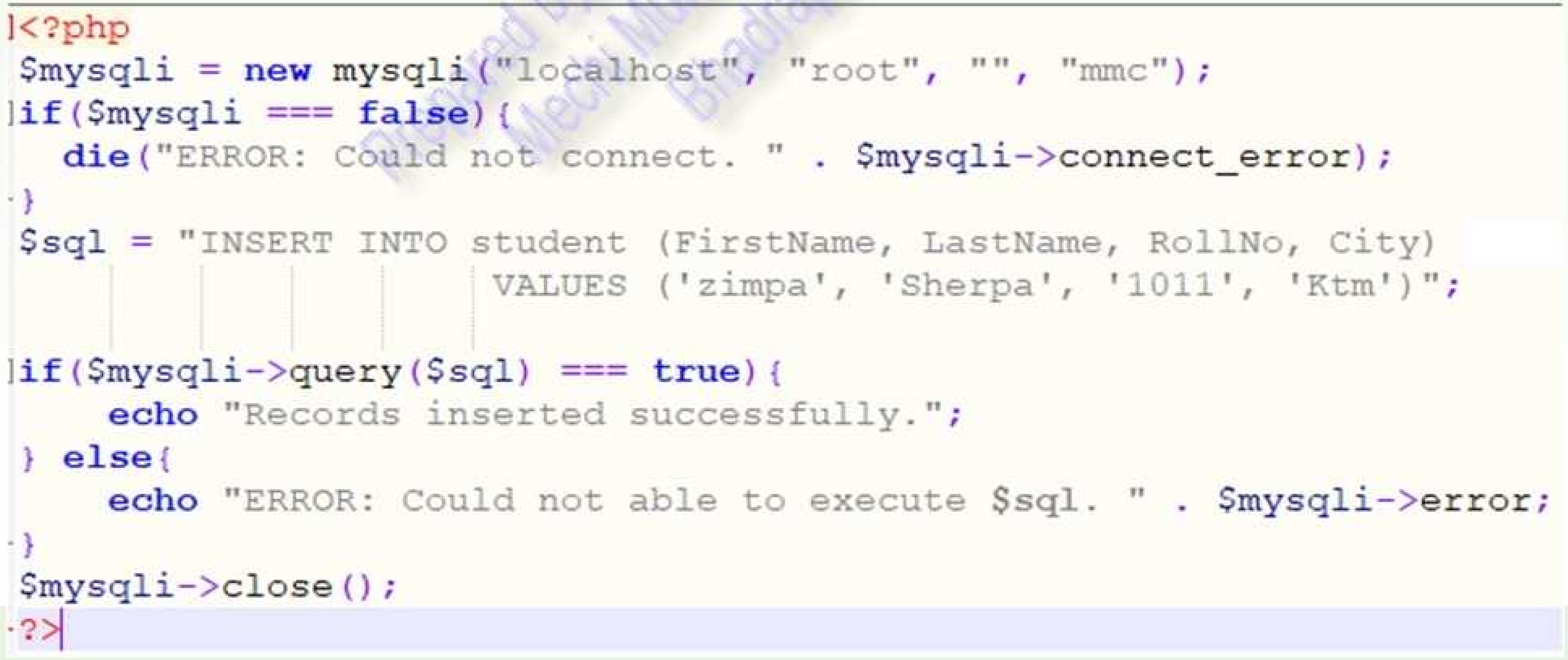
```
<?php
$servername = "localhost";
$username = "root";
$password = "";/* Put your password */
$dbname = "mmc";/* Put your database name */
/* Create connection */
$conn = new mysqli($servername, $username, $password, $dbname);
/* Check connection*/
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
/* sql to create table */
$sql = "CREATE TABLE Student
(
ID int NOT NULL AUTO_INCREMENT,
FirstName varchar(50),
LastName varchar(50),
RollNo varchar(50),
City varchar(50),
PRIMARY KEY (ID)
)";

if ($conn->query($sql) === TRUE) {
    echo "Table test created successfully";
}
 else {
    echo "Error creating table: " . $conn->error;
}
$conn->close();
?>
```

Database Connectivity

Student				
ID	FirstName	LastName	Rollno	City
1	Zimpa	Sherpa	1011	Ktm

INSERT INTO table_name (column1,column2,...)
VALUES (value1,value2,...);



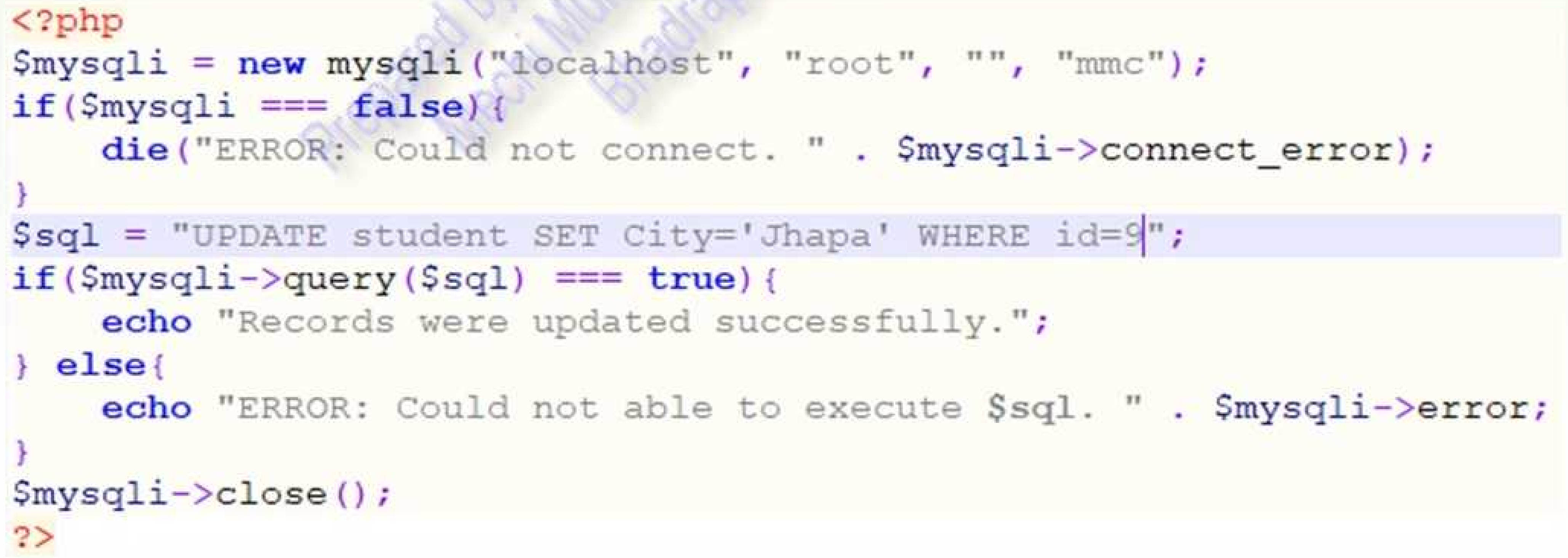
```
<?php
$mysqli = new mysqli("localhost", "root", "", "mmc");
if($mysqli === false){
    die("ERROR: Could not connect. " . $mysqli->connect_error);
}
$sql = "INSERT INTO student (FirstName, LastName, RollNo, City)
                    VALUES ('zimpa', 'Sherpa', '1011', 'Ktm')";

if($mysqli->query($sql) === true){
    echo "Records inserted successfully.";
} else{
    echo "ERROR: Could not able to execute $sql. " . $mysqli->error;
}
$mysqli->close();
?>
```

```
UPDATE table_name SET column1=value,
column2=value2,... WHERE column_name=some_value
```

Student

ID	FirstName	LastName	Rollno	City
1	Zimpa	Sherpa	1011	Jhapa

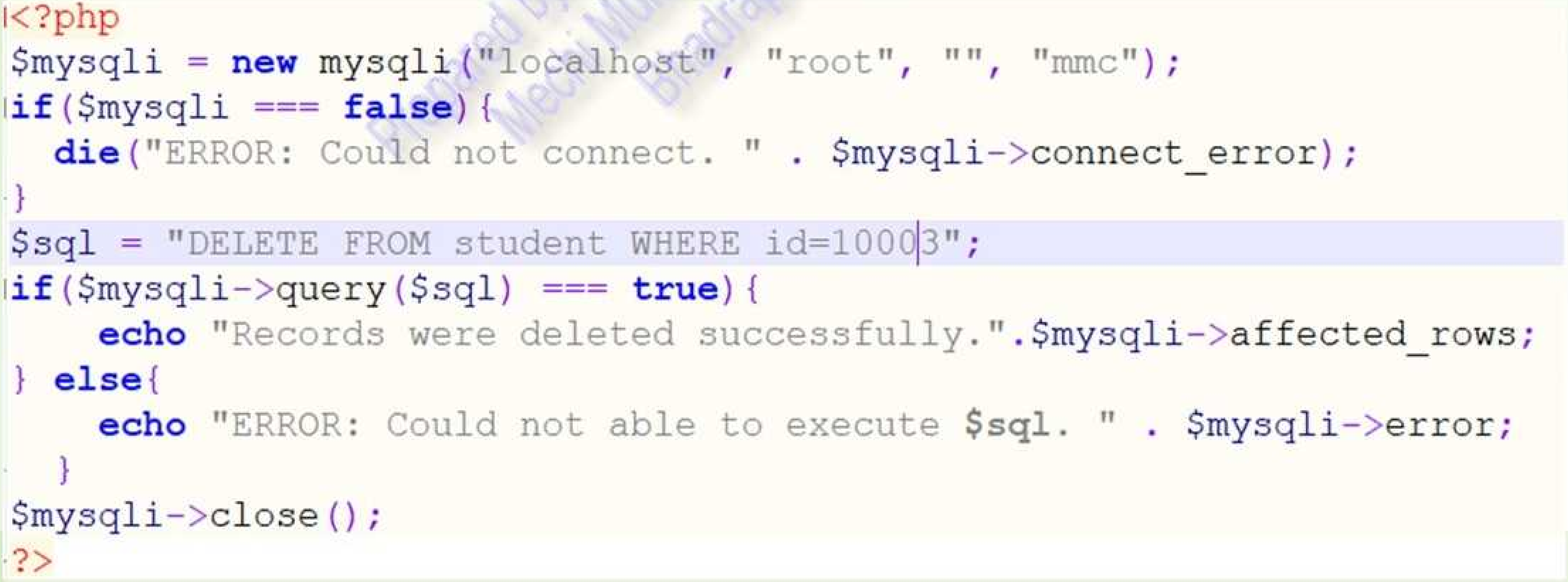


```php
<?php
$mysqli = new mysqli("localhost", "root", "", "mmc");
if($mysqli === false){
    die("ERROR: Could not connect. " . $mysqli->connect_error);
}
$sql = "UPDATE student SET City='Jhapa' WHERE id=9";
if($mysqli->query($sql) === true){
    echo "Records were updated successfully.";
} else{
    echo "ERROR: Could not able to execute $sql. " . $mysqli->error;
}
$mysqli->close();
?>
```

Database Connectivity

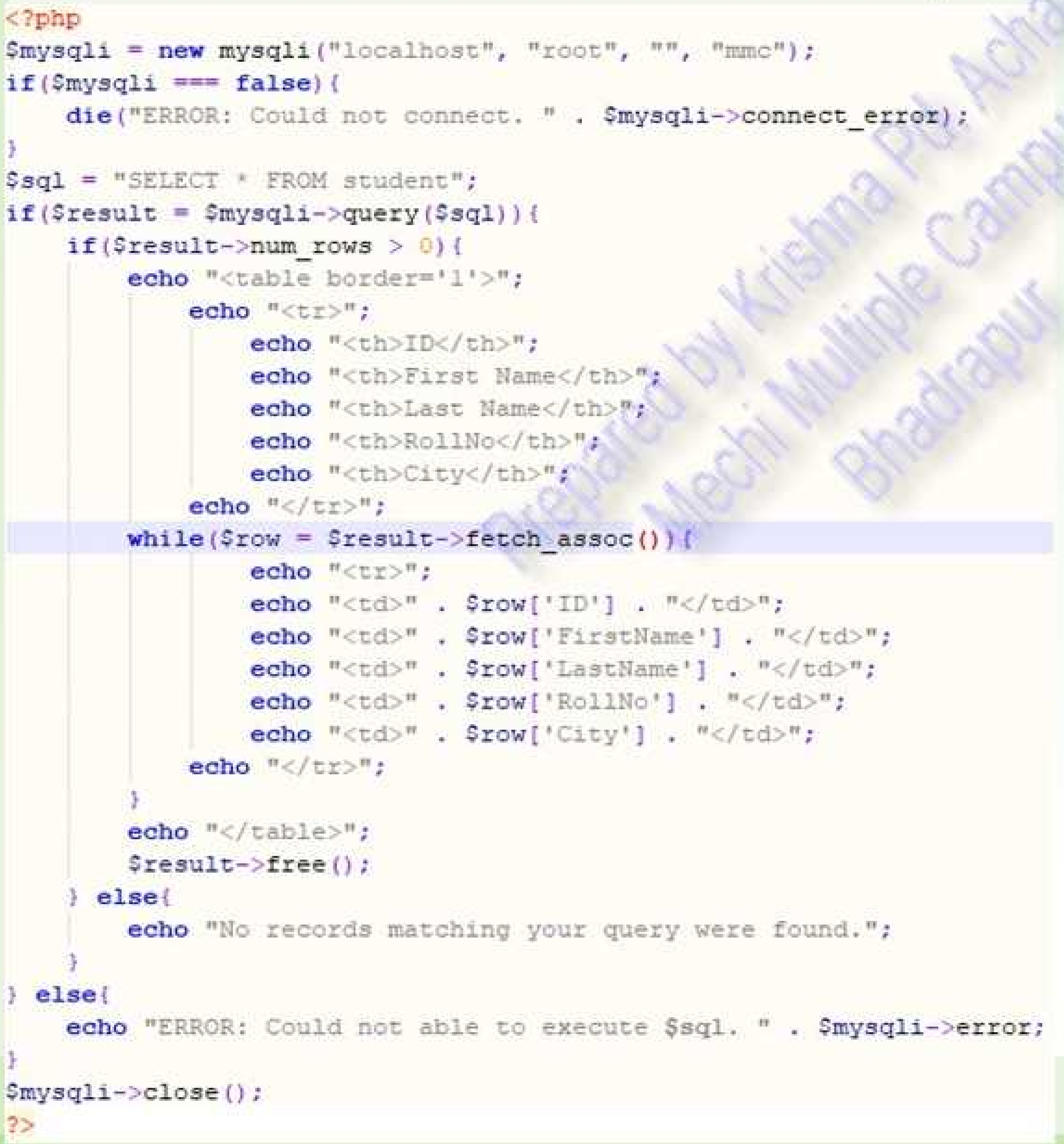
Student				
ID	FirstName	LastName	Rollno	City
1	Zimpa	Sherpa	1011	Jhapa

DELETE FROM table_name WHERE column_name=some_value



```php
<?php
$mysqli = new mysqli("localhost", "root", "", "mmc");
if($mysqli === false){
  die("ERROR: Could not connect. " . $mysqli->connect_error);
}
$sql = "DELETE FROM student WHERE id=10003";
if($mysqli->query($sql) === true){
    echo "Records were deleted successfully.".$mysqli->affected_rows;
} else{
    echo "ERROR: Could not able to execute $sql. " . $mysqli->error;
  }
$mysqli->close();
?>
```

Database Connectivity



```
<?php
$mysqli = new mysqli("localhost", "root", "", "mmc");
if($mysqli === false){
    die("ERROR: Could not connect. " . $mysqli->connect_error);
}
$sql = "SELECT * FROM student";
if($result = $mysqli->query($sql)){
    if($result->num_rows > 0){
        echo "<table border='1'>";
            echo "<tr>";
                echo "<th>ID</th>";
                echo "<th>First Name</th>";
                echo "<th>Last Name</th>";
                echo "<th>RollNo</th>";
                echo "<th>City</th>";
            echo "</tr>";
        while($row = $result->fetch_assoc()){
                echo "<tr>";
                echo "<td>" . $row['ID'] . "</td>";
                echo "<td>" . $row['FirstName'] . "</td>";
                echo "<td>" . $row['LastName'] . "</td>";
                echo "<td>" . $row['RollNo'] . "</td>";
                echo "<td>" . $row['City'] . "</td>";
            echo "</tr>";
        }
        echo "</table>";
        $result->free();
    } else{
        echo "No records matching your query were found.";
    }
} else{
    echo "ERROR: Could not able to execute $sql. " . $mysqli->error;
}
$mysqli->close();
?>
```

SELECT column1_name, column2_name, columnN_name FROM table_name;

SELECT column_name(s) FROM table_name WHERE column_name=operator value

SELECT column_name(s) FROM table_name LIMIT row_offset, row_count;

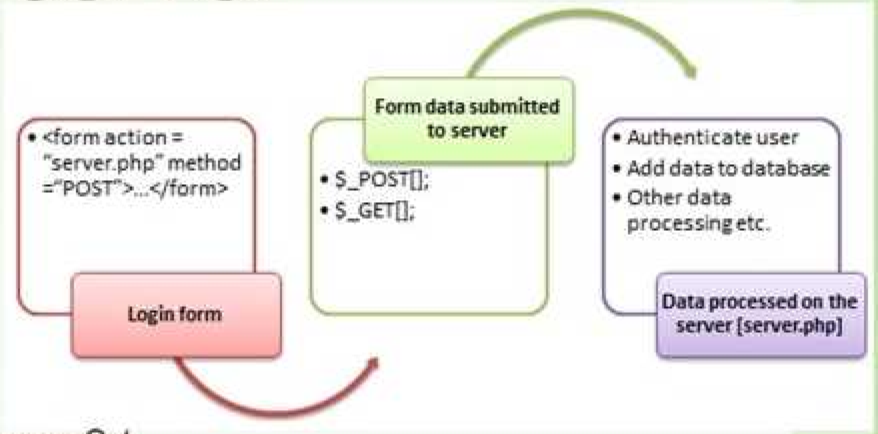
SELECT * FROM persons LIMIT 3;

SELECT * FROM persons LIMIT 1, 3;

SELECT column_name(s) FROM table_name ORDER BY column_name(s) ASC | DESC

SELECT * FROM persons ORDER BY first_name DESC

Methods of Sending Information to Server



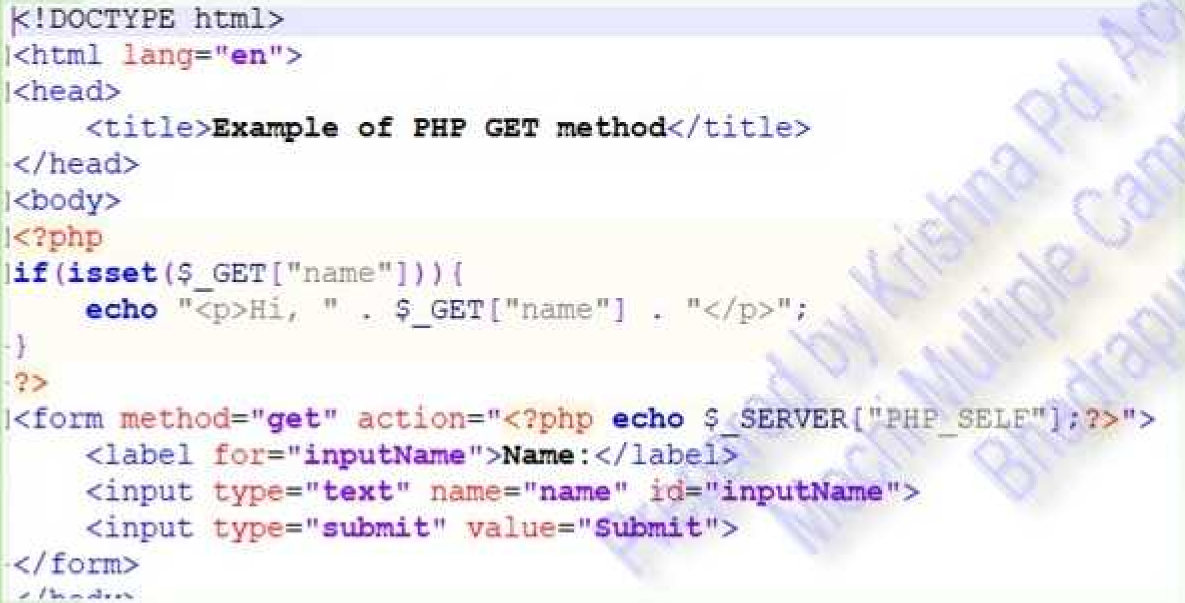
• <form action = "server.php" method ="POST">...</form>
Login form
Form data submitted to server
• $_POST[];
• $_GET[];
• Authenticate user
• Add data to database
• Other data processing etc.
Data processed on the server [server.php]

A web browser communicates with the server typically using one of the two HTTP (Hypertext Transfer Protocol) methods; GET and POST. In GET method the data is sent as URL parameters that are usually strings of name and value pairs separated by ampersands (&).

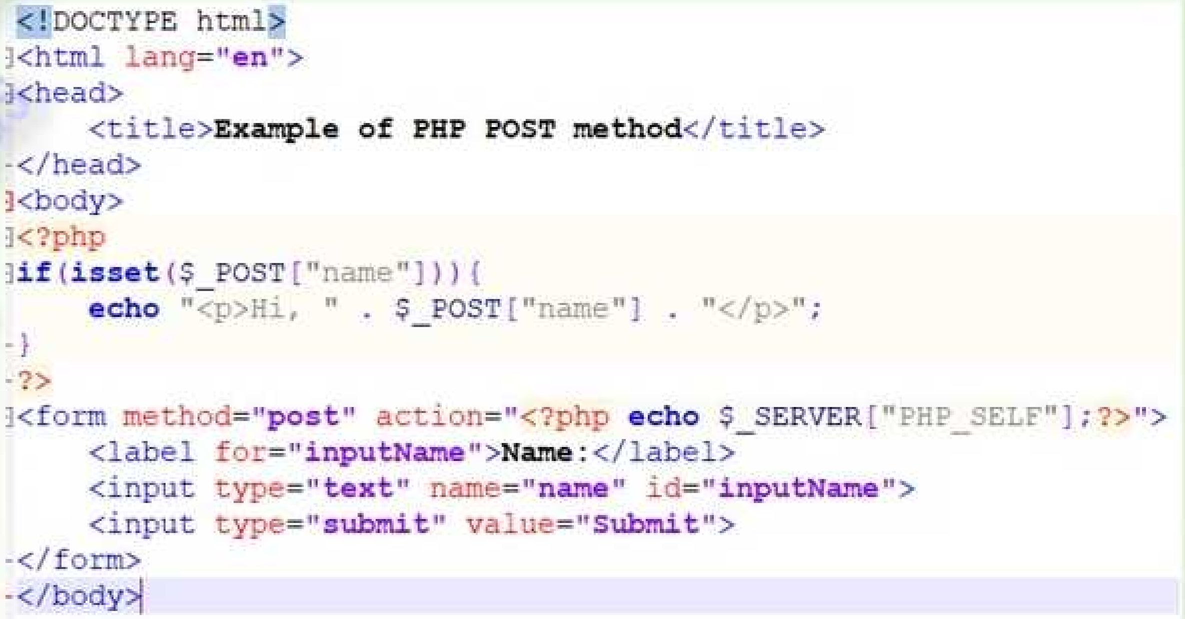
http://www.mechicampus.edu.np/action.php?name=mohan&age=24

- Since the data sent by the GET method are displayed in the URL, it is possible to bookmark the page with specific query string values.
- The GET method is not suitable for passing sensitive information such as the username and password, because these are fully visible in the URL query string as well as potentially stored in the client browser's memory as a visited page.
- Because the GET method assigns data to a server environment variable, the length of the URL is limited. So, there is a limitation for the total data to be sent.

Methods of Sending Information to Server



```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Example of PHP GET method</title>
</head>
<body>
<?php
if(isset($_GET["name"])){
    echo "<p>Hi, " . $_GET["name"] . "</p>";
}
?>
<form method="get" action="<?php echo $_SERVER["PHP_SELF"];?>">
    <label for="inputName">Name:</label>
    <input type="text" name="name" id="inputName">
    <input type="submit" value="Submit">
</form>
```



```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Example of PHP POST method</title>
</head>
<body>
<?php
if(isset($_POST["name"])){
    echo "<p>Hi, " . $_POST["name"] . "</p>";
}
?>
<form method="post" action="<?php echo $_SERVER["PHP_SELF"];?>">
    <label for="inputName">Name:</label>
    <input type="text" name="name" id="inputName">
    <input type="submit" value="Submit">
</form>
</body>
```

$_REQUEST["name"]

PHP Basics of File Handling

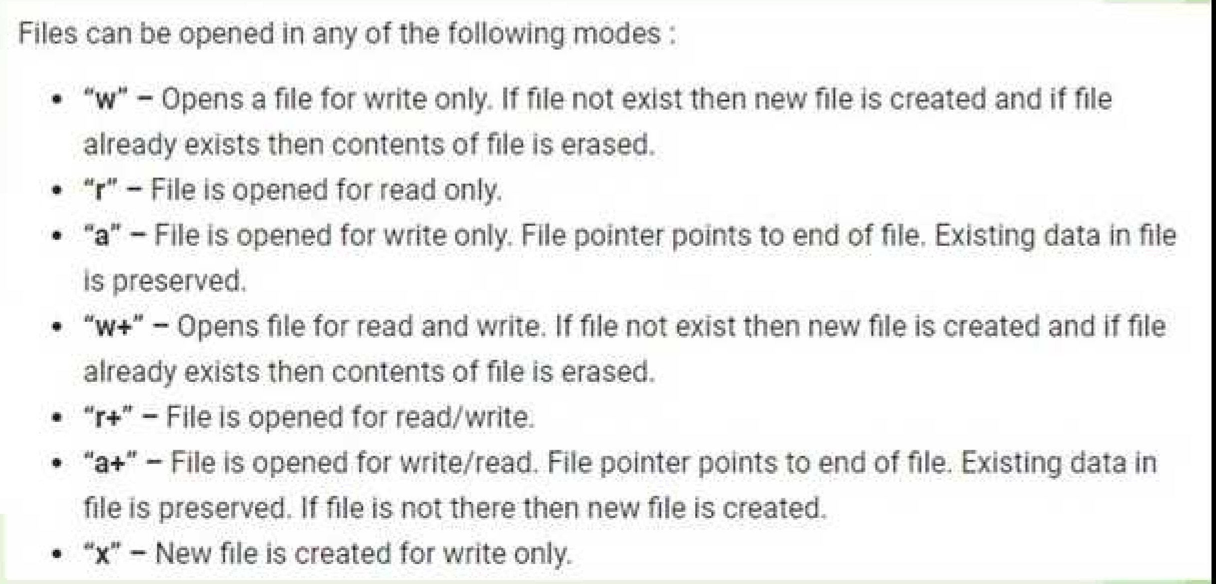
- File handling is needed for any application. For some tasks to be done file needs to be processed. File handling in PHP is similar as file handling is done by using any programming language like C. PHP has many functions to work with normal files. Those functions are:

1) fopen() – PHP fopen() function is used to open a file. First parameter of fopen() contains name of the file which is to be opened and second parameter tells about mode in which file needs to be opened, e.g.,

```
<?php
$file = fopen("demo.txt",'w');
?>
```

Read:

```
<?php
$filename = "demo.txt";
$file = fopen( $filename, 'r' );
$size = filesize( $filename );
$filedata = fread( $file, $size );
?>
```



Files can be opened in any of the following modes :

- **"w"** – Opens a file for write only. If file not exist then new file is created and if file already exists then contents of file is erased.
- **"r"** – File is opened for read only.
- **"a"** – File is opened for write only. File pointer points to end of file. Existing data in file is preserved.
- **"w+"** – Opens file for read and write. If file not exist then new file is created and if file already exists then contents of file is erased.
- **"r+"** – File is opened for read/write.
- **"a+"** – File is opened for write/read. File pointer points to end of file. Existing data in file is preserved. If file is not there then new file is created.
- **"x"** – New file is created for write only.

PHP Basics of File Handling

2) **fread() —** After file is opened using fopen() the contents of data are read using fread(). It takes two arguments. One is file pointer and another is file size in bytes, e.g.

3) **fwrite()** – New file can be created or text can be appended to an existing file using fwrite() function. Arguments for fwrite() function are file pointer and text that is to written to file. It can contain optional third argument where length of text to written is specified.

```
<?php
$file = fopen("demo.txt", 'w');
$text = "Hello world\n";
fwrite($file, $text);
?>
```

4) **fclose() –** file is closed using fclose() function. Its argument is file which needs to be closed.

```
<?php
$file = fopen("demo.txt", 'r');
//some code to be executed
fclose($file);
?>
```