Web Programming
#9 Cookie & Session

Herman Kabetta, M.T.

What is Cookie?

- Small file with the maximum size of 4KB that the web server stores on the client computer.

- HTTP is a stateless protocol; cookies allow us to track the state of the application.

- Personalizing the user experience using cookies.

- A cookie can only be read from the domain that it has been issued from.

- Cookie must be started before any HTML tags.

1) Cookie Enabled Page Request
- User access index.php
- index.php tracks user IP Addresses

2) Server Sets Cookie
- server stores the IP address in the cookie

3) Other Page Requests
- server returns cookie name and value on all page requests

Creating Cookies

```php
<?php
    setcookie('language', 'english');
    echo 'the cookie has been set';
?>

<?php
    setcookie("username", "admin", time()+ 60,'/');
    // expires after 60 seconds
    echo 'the cookie has been set for 60 seconds';
?>
```

	Elements	Console	Sources	Network	Performance	Memory	Application	Security	Audits

Web SQL
▼ Cookies
  🍪 http://localhost

Cache

C Filter

Name	Value	Domain	Path	Expires / Max-Age
_ga	GA1.1....	localhost	/	2020-03-06T01:39:49.000Z
language	english	localhost	/	Session

```php
1  <?php
2      print_r($_COOKIE);
3      //output the contents of the cookie array variable
4      echo $_COOKIE['language'];
5  ?>
```

Set the expiry time to a time that has already passed to delete

```php
1  <?php
2      setcookie("language", "english", time() - 60,'/');
3  ?>
```

Securing Cookies

```php
<?php

    function signing_checksum($string) {
        $salt = "r4j4H03t4n"; // makes process hard to guess
        return hash('sha1', $string . $salt);
    }

    function sign_string($string) {
        return $string . '--' . signing_checksum($string);
    }

    function signed_string_is_valid($signed_string) {
        $array = explode('--', $signed_string);
        // if not 2 parts it is malformed or not signed
        if(count($array) != 2) { return false; }

        $new_checksum = signing_checksum($array[0]);
        return ($new_checksum === $array[1]);
    }

    setcookie("language", sign_string("english"));
    echo "cookie has been set";

?>
                echo signed_string_is_valid($_COOKIE['language']);
```

Session

- A session is a global variable stored on the server.

- Each session is assigned a unique id which is used to retrieve stored values.

- Whenever a session is created, a cookie containing the unique session id is stored on the user's computer and returned with every request to the server. If the client browser does not support cookies, the unique php session id is displayed in the URL

- Sessions have the capacity to store relatively large data compared to cookies.

- The session values are automatically deleted when the browser is closed. If you want to store the values permanently, then you should store them in the database.

Session

- Just like the $_COOKIE array variable, session variables are stored in the $_SESSION array variable.

- Just like cookies, the session must be started before any HTML tags.

- Store important information such as the user id more securely on the server where malicious users cannot temper with them.

- Alternative to cookies on browsers that do not support cookies.

- Store global variables in an efficient and more secure way compared to passing them in the URL

- Good for developing an application such as a shopping cart that has to temporary store information with a capacity larger than 4KB.

```php
<?php
    session_start();
    $_SESSION['user_id'] = 42;
    echo $_SESSION['user_id'];
?>
```

```php
<?php

    session_start(); //start the PHP_session function

    if(isset($_SESSION['page_count'])) {
        $_SESSION['page_count'] += 1;
    } else {
        $_SESSION['page_count'] = 1;
    }
    echo 'You are visitor number ' . $_SESSION['page_count'];

?>
```
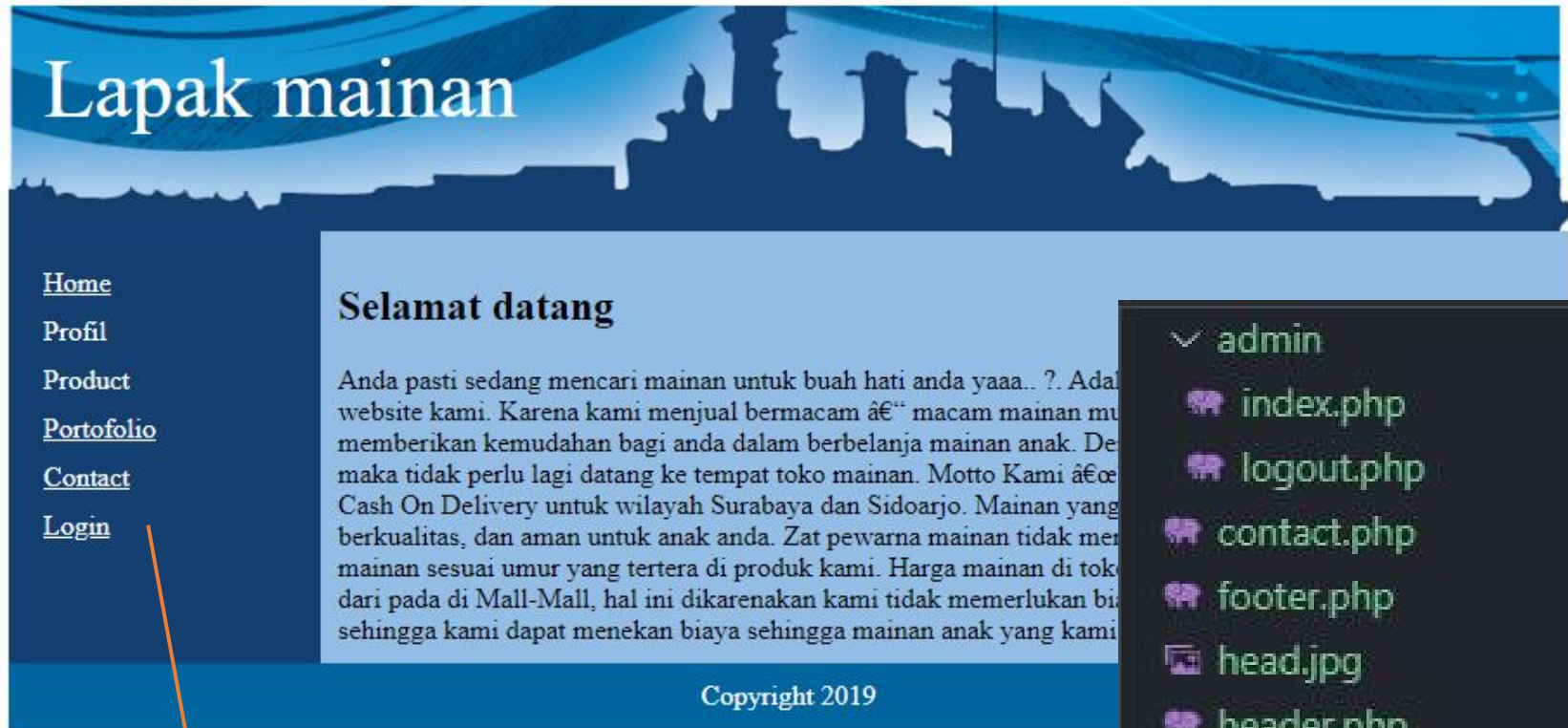
```php
<?php
    session_destroy();
    //destroy entire session
    unset($_SESSION['page_count']);
    //destroy page_count session
?>
```
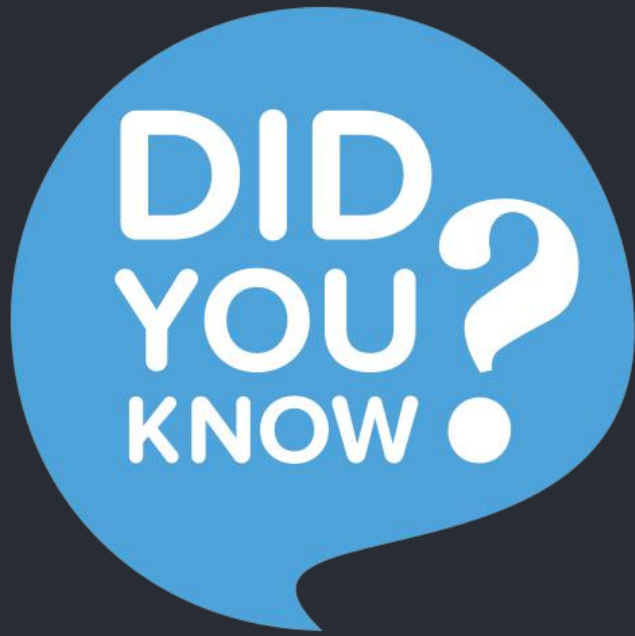
Make a login link to admin directory
Login

The **index.php** is a PHP file that is the entry point of any website or application.

```
http://localhost/goodapp == http://localhost/goodapp/index.php
```

Login Form Using Session

```html
1   <!DOCTYPE HTML>
2   <html>
3   <head>
4       <title>Admin Page</title>
5   </head>
6   <body>
7     <div class="login" style="margin :0 auto; width: 600px;">
8     <h2 class="header">Login Here...</h2>
9     <form method="post">
10      <input type="text" name="user" placeholder="Username...">
11      <input type="password" name="pass" placeholder="Password...">
12      <input type="submit" name="login" value="Log In">
13    </form>
14    </div>
15  </body>
16  </html>
```

```php
8    <?php
9        session_start();
10       if (!isset($_SESSION['user'])) {
11   ?>

18   </form>
19   <?php
20     if(isset($_POST['user'])&&isset($_POST['pass'])){
21         if ($_POST['user']=='admin' && $_POST['pass']=='admin123') {
22             $_SESSION['user'] = $_POST['user'];
23             header('location:http://localhost/admin/index.php');
24         } else {
25             echo "<br>Wrong username or password";
26         }
27     }
28   ?>
```

```php
30    <?php
31        } else {
32            echo "<div style=\"margin :0 auto; width: 600px;\">";
33            echo "Selamat Datang ".$_SESSION['user']."!<br>";
34            echo "<a href='logout.php'>LOGOUT</a>";
35            echo "</div>";
36        }
37    ?>
38  </body>
```

admin > logout.php

```php
1    <?php
2        session_start();
3        session_destroy();
4        header('location:index.php');
5    ?>
```

Protect Page Using Session

admin > 🐷 profil.php

```php
<?php
    session_start();
    if (isset($_SESSION['is_loggedin'])) {
        echo "<div style=\"margin :0 auto; width: 600px;\">";
        echo "<h2>This is Profil Page</h2>";
        echo "User  : ".$_SESSION['is_loggedin'];
        echo "<br>Name  : Admin Web App";
        echo "<br>Email : admin@myweb.app";
        echo "</div>";
    } else {
        header("Location:http://localhost");
    }
?>
```
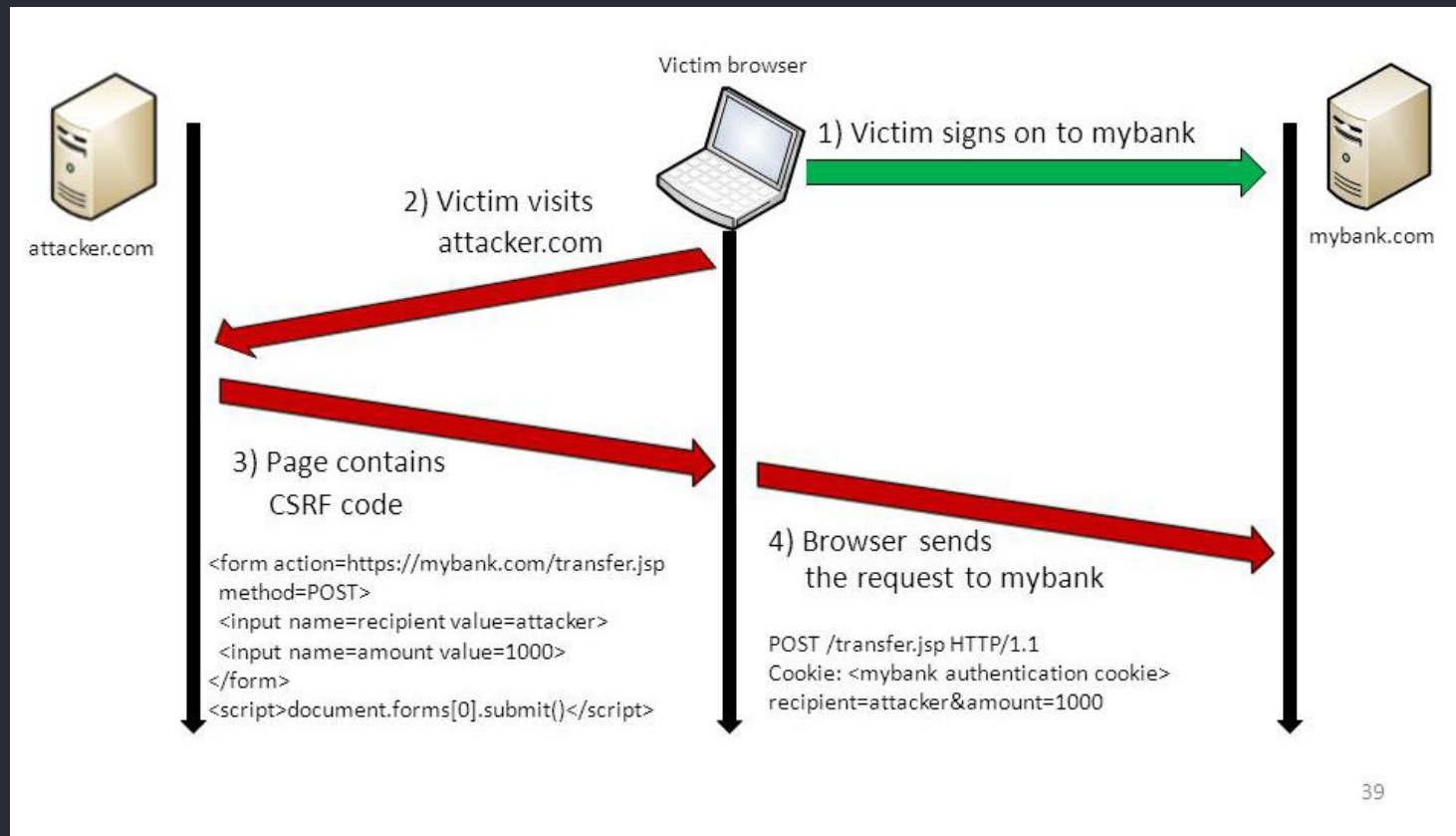
Cross Site Request Forgery

"Cross-Site": originates on one site but performs an action on another

"Request Forgery": it is not a genuine user request

respon.php

```php
1    <?php
2        echo $_POST['amount'];
3    ?>
```

Different Server

fake.php

```html
1    <html>
2      <head>
3        <title>Fake Form</title>
4      </head>
5      <body onload="document.bank_form.submit()">
6        <form action="http://localhost:8080/respon.php" method="POST"
7              name="bank_form" style="display: none;">
8          <input type="text" name="amount" value="10000" />
9          <input type="text" name="to_account" value="2468013579" />
10       </form>
11     </body>
12   </html>
```

🐷 trueform.php

```php
<?php
    session_start();

    function csrf_token() {
        return bin2hex(rand(100000, 999999));
        // in PHP 7 use random_bytes(64);
    }

    function create_csrf_token() {
        $token = csrf_token();
        $_SESSION['csrf_token'] = $token;
        $_SESSION['csrf_token_time'] = time();
        return $token;
    }

    function csrf_token_tag() {
        $token = create_csrf_token();
        return '<input type="hidden" name="csrf_token" value="' . $token . '">';
    }
?>
```

Prevent CSRF Attack

```php
trueform.php
21    <html>
22      <head>
23        <title>True Form</title>
24      </head>
25      <body>
26        <form action="respon.php" method="POST">
27          <input type="text" name="amount" value="10000" />
28          <?php echo csrf_token_tag(); ?>
29          <input type="text" name="to_account" value="2468013579" />
30          <input type="submit" value="submit" />
31        </form>
32      </body>
33    </html>
```

Prevent CSRF Attack

```php
respon.php
1   <?php
2       session_start();
3       function csrf_token_is_valid() {
4           if(!isset($_POST['csrf_token'])) { return false; }
5           if(!isset($_SESSION['csrf_token'])) { return false; }
6           return ($_POST['csrf_token'] === $_SESSION['csrf_token']);
7       }
8
9
10      if(csrf_token_is_valid()) {
11          echo $_POST['amount'];
12      } else {
13          echo "Invalid request!";
14      }
15  ?>
```