

## M09-UF-1: Implantació d'Aplicacions Web



# Índex

<b>M09-UF-1.....</b>	<b>1</b>
<b>1-Apache 14/09.....</b>	<b>2</b>
<b>2-PHP.....</b>	<b>3</b>
<b>3- Trozo de código en PHP.....</b>	<b>4</b>
<b>4- Bases de Datos y Forms.....</b>	<b>5</b>
<b>4.1 Login Register y sus parsers:.....</b>	<b>5</b>
<b>5- Sesiones.....</b>	<b>6</b>

## 1-Apache 14/09

Apache fue el primer gran servidor web en los años 90, fue sacado de linux, se creó en el año 91 cuando se creó también WWW el world wide web.



Next es una empresa de Steve Jobs que creo cuando lo echaron de Apple, lo echaron porque iba a llevar la empresa a la quiebra, tenía ordenadores Unix, osea ordenadores muy avanzados para la época, eso llevaba a que fueran los únicos ordenadores de la época que podría soportar un servidor web, en el año 91 sale la primera versión del núcleo de Linux y dos años después en el **93** avanza tanto que sale una versión de linux para crear servidores y esta es **apache**. Básicamente, gracias a Linux y apache eran los únicos que tenían interés en este proyecto y una vez lo sacaron adelante todas las grandes empresas se aprovecharon para crear su dominio. Del 93 al 96 se le llamaba web 0 porque era de solo lectura, era de texto porque no había nada más que texto plano. Un poco más tarde en el 96 apache añadió una nueva tecnología que se llamaba **CGI** que era es un **método por el cual un servidor web puede interactuar con programas externos de generación de contenido**, este duro hasta más o menos 2004 y le llamaremos web 1.0 y gracias a esto hace el primer **"BOOM"** creando así varios foros tiendas entre más cosas.

Más o menos en 2015 se crea la **web 2.0** que es la web de las redes sociales, aparece nueva información sin necesidad de actualizar página y fue aquí donde empezó el verdadero **"BOOM"** de internet, a esta web se la llama la **web del cambio continuo o web semántica**.

Ahora mismo seguimos en la web 2.0, ya que la web 3 es una historia de crypto bros. En el 2004 apareció el boom del .com muchas empresas apostaron por los dominios .com y hubo un boom parecido al del bitcoin por mucha inversión fallida y mucha estafa.

El .cat existe, ya que aunque Cataluña no sea un país, representa al idioma, hubo polémica, ya que mucha gente quería el .cat por gato en inglés, pero después el gobierno de Cataluña reclamo el dominio .cat e hizo que muchas webs de todo el mundo tuvieron que poner a Cataluña como idioma en la web.

## 2-PHP

**PHP** es un lenguaje de programación web, cuando hablamos de web se habla de front end que sería **CSS HTML y JS** entre otros y back end sería **PHP**. PHP es interpretado por el servidor como apache.

En resumen, quiere decir que frontend es como la capa visual o sea la del cliente y el backend es la segunda capa la que está en el servidor.



Aparte de backend y frontend existe el Full Stack que quiere decir que se trabaja tanto cliente como con servidor.

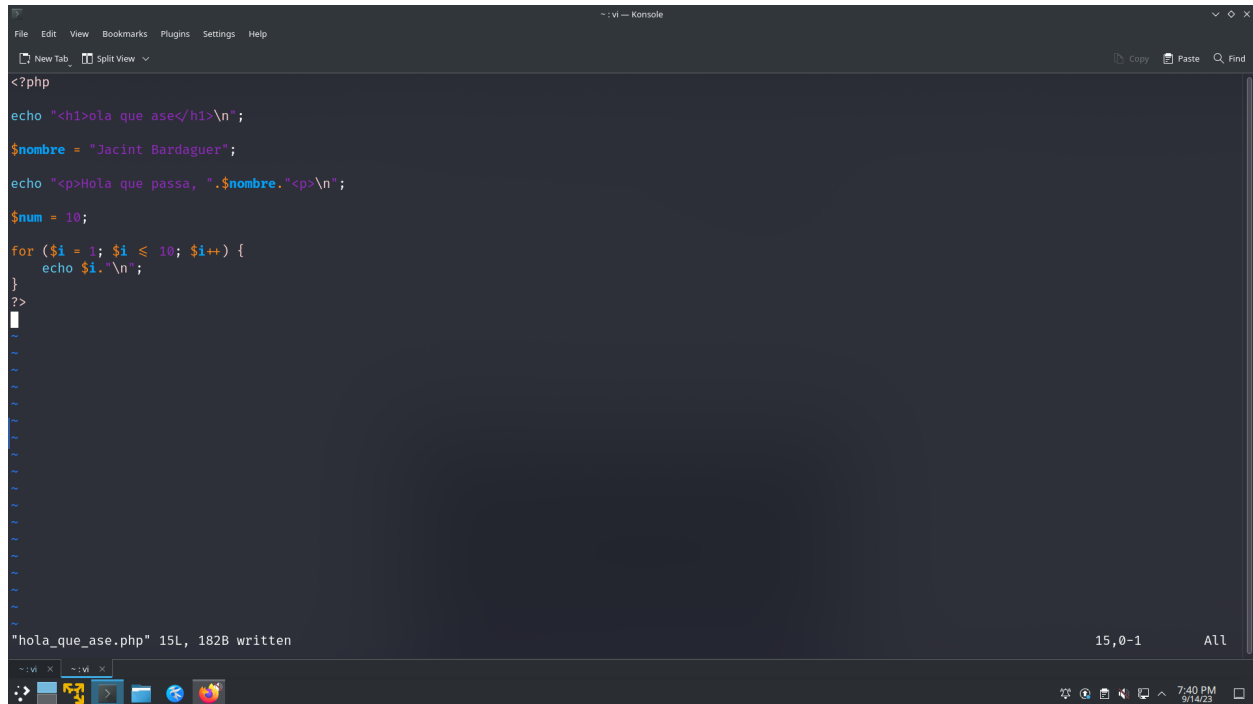
En diferencia salarial, es decir, lo que mejor se paga mejor en frontend, pero es más difícil que en backend, frontend es más artístico y backend está también muy bien, pero se sufre más, ya que debes estar mucho más pendiente de que el funcionamiento del servidor sea el correcto, vamos que es más sufrido.

Antes de PHP el backend se programaba en **C** y también con otro lenguaje que se llama **perl**.



Usaremos **PHP** para **procesar** los datos que nos lleguen del cliente para poder después enviar una respuesta para al cliente.

### 3- Trozo de código en PHP



```
<?php
echo "<h1>ola que ase</h1>\n";

$nombre = "Jacint Bardaguer";

echo "<p>Hola que passa, ".$nombre."<p>\n";

$num = 10;

for ($i = 1; $i <= 10; $i++) {
    echo $i."\n";
}

?>
```

The screenshot shows a VS Code editor window with a dark theme. The file name 'hola\_que\_ase.php' is visible in the bottom left. The code is a PHP script that outputs an HTML heading, a paragraph with a name, and a loop printing numbers 1 to 10. The status bar at the bottom right shows '15,0-1' and 'All'.

Para hacer una **print/echo** es como en bash, pero debemos cerrar con el “;” : **echo “hola”;**

Para crear una variable debemos escribir el “\$” + **NombreVariable**: “**\$NuevaVariable = “10”**”

Para llamar a una variable debemos hacer: “**echo “hola”.\$NuevaVariable**””

Para hacer un bucle for es de esta forma:

```
for ($i = 1; $i <= 10; $i++) {
echo $i."\n";
}
```

En este caso este for printa esto:



```
1
2
3
4
5
6
7
8
9
10
```

The screenshot shows the output of the PHP script, which is a list of numbers from 1 to 10, each on a new line.

## 4- Bases de Datos y Forms

Comenzamos a crear nuestro formulario en un archivo .php donde pondremos dentro el html para luego enviarlo a una base de datos host (nuestra ip) con mysql.

Otra característica de PHP es que no muestra el código de error en la bash normal, por eso si queremos activar-lo debemos acceder a /etc/php/7.4/apache2 i cambiar la opción de Display erros en ON, ya que por default estará en OFF

```
root@lluc:/etc/php/7.4/apache2# pwd
/etc/php/7.4/apache2
root@lluc:/etc/php/7.4/apache2# ls
conf.d  php.ini
root@lluc:/etc/php/7.4/apache2#
```

**Echo <<<EOD:** es como una señal que marca el inicio de un bloque largo de texto, llamado cadena heredoc. Este tipo de cadena puede extenderse por varias líneas. El <<<EOD debe estar solo en su propia línea, seguido de un punto y coma (;). Esto es útil para definir bloques de texto extensos, como HTML, XML u otros tipos de contenido que abarquen varias líneas.

**Str\_replace:** es una función que busca y reemplaza partes específicas de un texto con otras cosas. Por ejemplo, podrías decirle: "Encuentra la palabra 'manzana' y cámbiala por 'naranja' en este párrafo".

Ahora, respecto a las contraseñas, str\_replace no es una buena opción. Imagina que usas str\_replace para cambiar una parte de tu contraseña. El problema es que, aunque parezca seguro, no lo es. No protege tu contraseña de manera sólida.

### 4.1 Login Register y sus parsers:

Hemos creado un login y un register para nuestra página web y también sus respectivos parsers, que podríamos decir que es lo que valida y filtra la información que nos llega desde el login.

Ejemplo de parte del código del **login\_parser.php**:

```
<?php

if (!isset($_POST["email"]) or !isset($_POST["password"])) {
    die("Error 1: Envía el formulario completo");
}

$email = $_POST["email"];
$password = $_POST["password"];

if (!filter_var($_POST["email"], FILTER_VALIDATE_EMAIL)) {
    die("Error 2: El correo electrónico no es válido");
}

if (strlen($_POST["password"]) < 6) {
    die("Error 3: La contraseña debe tener al menos 6 caracteres");
}
```

## 5- Sesiones

Las sesiones en PHP son como notas que se guardan en tu navegador o en el servidor para recordar si has iniciado sesión o no. Es como una especie de etiqueta que dice "estoy dentro" o "estoy fuera". Cuando inicias sesión, se crea una nota vacía automáticamente.

La función **session\_start** es como el botón de inicio. Cuando lo presionas al principio, comienzas a llevar un registro de tus cosas. Es la primera línea que necesitas poner para que todo funcione correctamente. Como cuando abres tu cuaderno antes de tomar apuntes en clase.

En resumen, las sesiones son como recordatorios que te dicen si estás conectado o no, y **session\_start** es como el primer paso para activar esos recordatorios.

Ejemplo de código de sesiones:

```
<?php

function openHTML ($title)
{
    $login_url = "login.php";
    $login_text = "Login";
    if (isset($_SESSION["id_user"])){
        $login_url = "logout.php";
        $login_text = "Logout";
    }
}
```

**session\_start();** Es como apretar el botón para iniciar todo. Ahora, después de haber llenado los espacios en el formulario de login, si tu nombre y contraseña coinciden con lo que está guardado en la lista del servidor en mariadb, obtienes tu pase para entrar.

**fetch\_assoc()** es un truco que te da una fila de información de la lista en un formato especial llamado array asociativo. Este tipo de lista es como una lista de contactos donde cada nombre (clave) está vinculado a un número de teléfono (valor). Así, si tienes una lista de resultados llamada \$result, y usas fetch\_assoc(\$result), obtienes una fila de información donde cada detalle tiene un nombre y un valor. Por ejemplo, podrías tener algo así como \$row = fetch\_assoc(\$result).