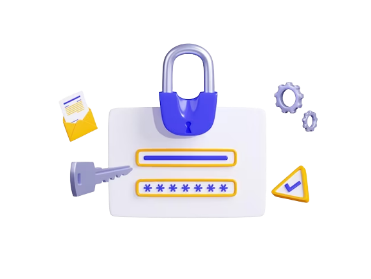
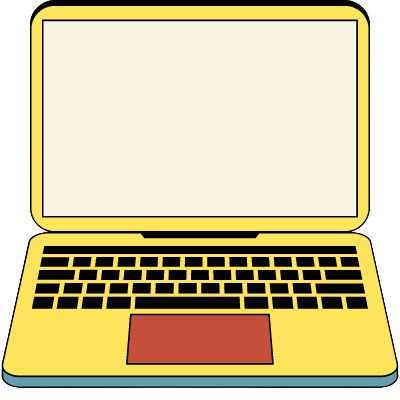
Subir sitio de login

**Christian Millán Soria**





- Curso: 2023-2024

- Clase: 2º DAW Tarde

- Profesor: David Hormigo Ramírez

- Módulo: Despliegue de Aplicaciones Web

- Fecha: 07/11/2023

**1. Código de login** 3

**1.1. Comprobación y modificación de la base de datos** 3

**1.2. Lógica del código** 5

**1.2.1. Estructura del login** 5

**1.2.2. login.html** 5

**1.2.3. login.php** 6

**1.2.4. sign-out.php** 6

**1.2.5. logged-in.php** 6

**1.2.6. style.css** 6

**1.3. Subida de archivos mediante FTP** 7

**1.3.1. Recolocar los archivos** 8

**2. Dominio DNS** 9

**3. Creación del sitio** 9

**4. Comprobación de funcionamiento** 10

En esta práctica se mostrará el proceso de creación de un sistema básico de login en PHP haciendo uso de nuestra instancia y base de datos en AWS.

**1. Código de login**

**1.1. Comprobación y modificación de la base de datos**

Lo primero que vamos a hacer es crear la lógica de programación que validará el inicio de sesión para esta práctica.

Según se explica en la documentación de la misma, podemos realizarlo de **3 maneras diferentes**:

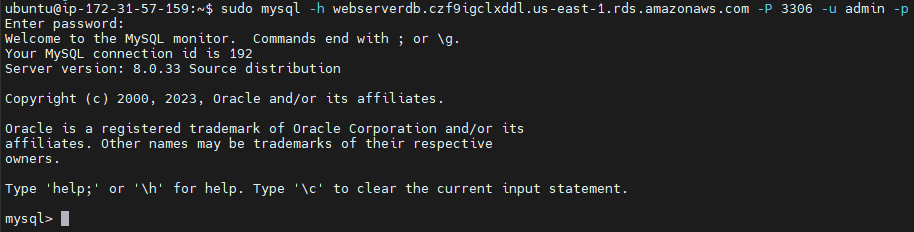
- **Localmente:** Se utilizan arrays para establecer un usuario con contraseña ficticios y se valida el login si se introducen las credenciales deseadas.

- **Acceso a base de datos:** Se realiza una conexión real a una base de datos (en nuestro caso la RDS de AWS) y se valida el login sólo si el usuario existe en la tabla creada en [**la práctica dedicada a ello**🔗](https://github.com/cmilsor245/daweb/blob/main/04-finalizar_pila_lamp/Finalizar%20instalaci%C3%B3n%20-%20Pila%20LAMP.pdf) y su contraseña es correcta.

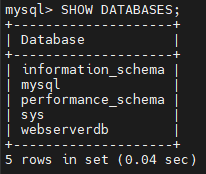
- **Ambos:** Se presenta un botón de tipo [“**radio**”🔗](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/radio) en el código HTML que permite elegir qué método de login se va a utilizar.

Vamos a comprobar que la base de datos y la tabla ya existen en RDS. Nos conectamos mediante SSH a nuestra instancia.

Ejecutamos el comando **sudo mysql -h webserverdb.czf9igclxddl.us-east-1.rds.amazonaws.com -P 3306 -u admin -p** y entramos al cliente de la base de datos.

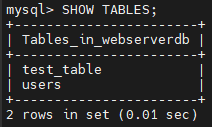


Con el comando “**SHOW DATABASES;**” vemos la lista de bases de datos de nuestro sistema.



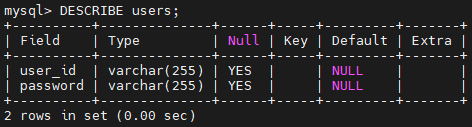
Utilizamos el comando **USE webserverdb;** para seleccionar la nuestra.

Una vez seleccionada, con el comando **SHOW TABLES;** accedemos a la lista de tablas, donde podemos ver que ya existe la tabla que creamos con anterioridad.



\* Es la tabla “**users**”.

Por último, con el comando **DESCRIBE users;** vemos la estructura de la misma.

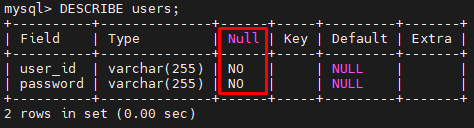


Vamos a realizar un cambio con el fin de que ninguno de los dos campos pueda estar nunca vacío para ningún usuario, lo cual conseguimos con el siguiente comando:

**ALTER TABLE users**

**MODIFY user\_id varchar(255) NOT NULL,**

**MODIFY password varchar(255) NOT NULL;**



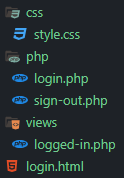
**1.2. Lógica del código**

El propósito de la práctica es realizar el código para el programa de forma local y subir el trabajo terminado a la instancia haciendo uso de FTP, como vimos en [**la anterior práctica🔗**](https://github.com/cmilsor245/daweb/blob/main/05-instalacion_servidor_ftp/Instalaci%C3%B3n%20de%20Servidor%20FTP.pdf).

Cabe destacar que cada vez que en nuestro código abramos una etiqueta de PHP, debe escribirse como “**<?php**” y no “**<?**”, ya que la versión del servidor no permite la nueva forma para esto.

**1.2.1. Estructura del login**

La estructura que seguirá nuestro mini-proyecto de login será la siguiente:



Vamos a hacer uso de la **tercera opción con respecto a la forma de loguearse**, lo que significa que contaremos con un botón de tipo “radio” que nos permitirá cambiar entre un tipo de login y otro.

**1.2.2. login.html**

En este archivo se mostrará un formulario donde introducir el usuario y contraseña con el que se quiere realizar el inicio de sesión, junto con dos botones radio para elegir el método de login.

[**Código del archivo.🔗**](https://github.com/cmilsor245/daweb/blob/main/06-subir_sitio_login/code/login.html)

Se puede observar cómo el formulario no redirecciona a un archivo concreto de forma directa, sino que cuenta con un enlace a un script de JavaScript externo que realizará la conversión.

**1.2.3. login.php**

Este es el controlador principal de nuestro sistema de login. Los datos introducidos en el formulario de la página principal se envían aquí, donde se comprueba que tipo de conexión se ha elegido y se realiza el login según dicha elección.

Si el login se valida, se redirige a la página de contenido para los usuarios de la base de datos.

Las contraseñas se gestionan de forma encriptada siempre.

[**Código del archivo.🔗**](https://github.com/cmilsor245/daweb/blob/main/06-subir_sitio_login/code/php/login.php)

Mediante el uso de [**PDO🔗**](https://www.php.net/manual/en/ref.pdo-mysql.php)realizamos una conexión con la base de datos en RDS, estableciendo las credenciales para la misma. Más adelante se realiza un **SELECT** en la tabla **users**, en los campos **user\_id** y **password** para comprobar el usuario introducido por teclado en la primera página.

**1.2.4. sign-out.php**

En él se encuentra el código necesario para destruir la sesión creada en el archivo anterior para volver a ser capaces de iniciar una sesión diferente.

[**Código del archivo.🔗**](https://github.com/cmilsor245/daweb/blob/main/06-subir_sitio_login/code/php/sign-out.php)

**1.2.5. logged-in.php**

Es la página que muestra la bienvenida al usuario una vez ha iniciado sesión.

[**Código del archivo.🔗**](https://github.com/cmilsor245/daweb/blob/main/06-subir_sitio_login/code/views/logged-in.php)

**1.2.6. style.css**

Es el código CSS elegido para la página, para que no sea tan simple como por defecto.

[**Código del archivo.🔗**](https://github.com/cmilsor245/daweb/blob/main/06-subir_sitio_login/code/css/style.css)

**1.3. Subida de archivos mediante FTP**

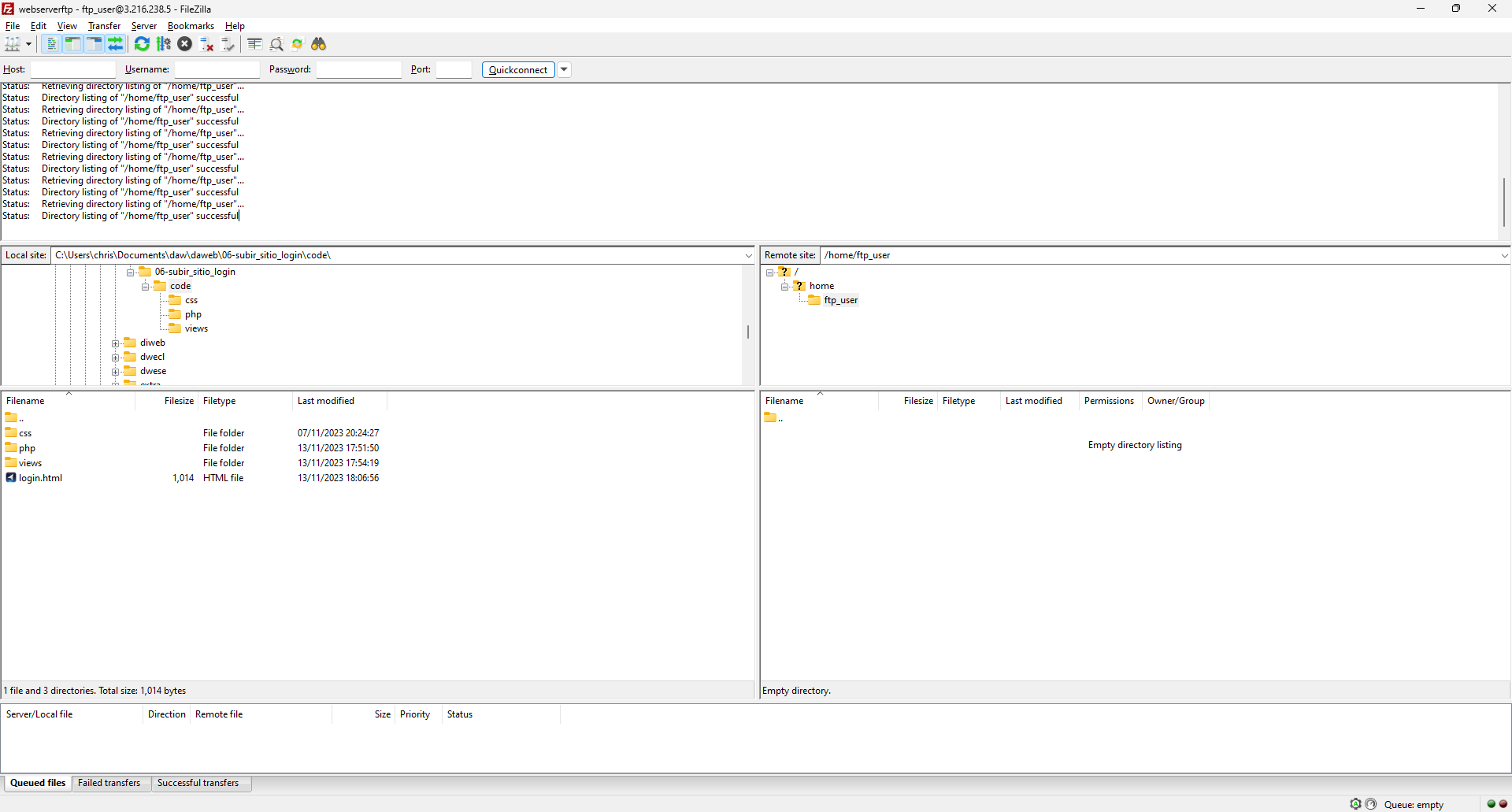
Vamos a proceder a subir los archivos al servidor haciendo uso de nuestro cliente FTP, FileZilla.

Como hemos visto, hemos realizado el código de programación en nuestra máquina local, por lo que ahora hay que subir dicho código a la instancia de AWS.

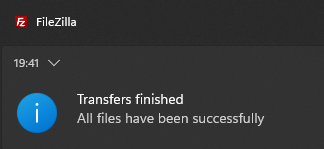
Para ello utilizaremos el usuario de FTP que creamos en [**la anterior práctica🔗**](https://github.com/cmilsor245/daweb/blob/main/05-instalacion_servidor_ftp/Instalaci%C3%B3n%20de%20Servidor%20FTP.pdf).

Accedemos al cliente en nuestra máquina local. Entramos en la conexión mediante el sitio “**webserverftp**”.

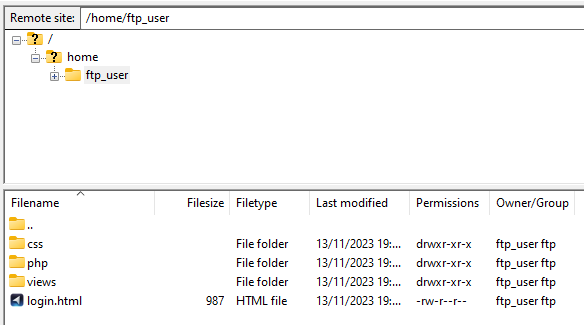
Una vez conectados, nos colocamos en el directorio local donde tenemos el mini-proyecto.



Arrastramos los archivos hacia la derecha, al directorio remoto de nuestra instancia en AWS. Se muestra una notificación en nuestro sistema que avisa de que todos los archivos se han transferido con éxito.



Podemos ver el nuevo directorio remoto con los archivos.



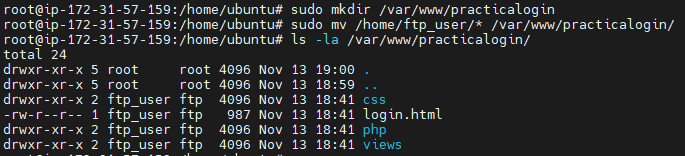
**1.3.1. Recolocar los archivos**

Como se puede ver, únicamente hemos podido subir estos archivos al directorio home del usuario “**ftp\_user**”. Esto es debido a que anteriormente limitamos el acceso únicamente a este usuario y establecimos que sólo pudiera navegar en su home.

Para solucionar esto, vamos a dirigirnos a la línea de comandos de nuestra instancia.

Aquí ejecutaremos, primero, el comando **sudo mkdir /var/www/practicalogin** para crear el directorio que alojará nuestro nuevo sitio, donde se encontrará todo nuestro código.

Una vez creado el directorio, con el comando **sudo mv /home/ftp\_user/\* /var/www/practicalogin/** movemos todos los archivos y carpetas del home del usuario al directorio deseado.



**2. Dominio DNS**

El objetivo de la práctica es que, ya que no contamos con un servidor DNS todavía, creemos un nuevo dominio DNS gratuito para redireccionar a la dirección IP de nuestra máquina en AWS.

Para poder hacer esto, vamos a utilizar [**Duck DNS🔗**](https://www.duckdns.org/), un servicio gratuito que permite realizar lo anterior mencionado.

Accedemos a la web de Duck DNS y nos creamos una cuenta. Una vez logueados, podemos acceder a crear dominios gratuitos y redireccionarlos a la dirección que deseemos.

Creamos uno con nuestro nombre y apellidos y actualizamos su IP para que apunte a nuestro servidor de AWS.



El nuevo dominio cuenta con una URL similar a la siguiente:

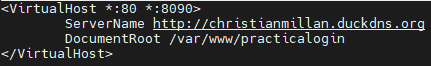
**http://christianmillan.duckdns.org**

**3. Creación del sitio**

En la documentación de la práctica se especifica que debemos crear un sitio nuevo en Apache para la configuración de este login.

Para ello, con el comando **sudo cp /etc/apache2/sites-available/misitio.conf /etc/apache2/sites-available/practicalogin.conf** hacemos una copia de un sitio existente y con nano entramos a editarlo.

Dentro cambiamos la directiva **ServerName** para que apunte al dominio de Duck DNS que hemos creado. Hacemos que el directorio que aloja el sitio sea el que hemos creado anteriormente también.

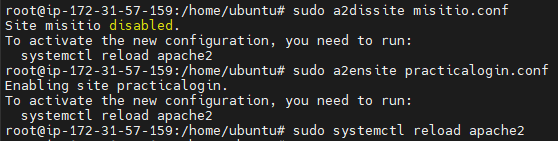


Guardamos, salimos y recargamos el servicio con el comando **sudo systemctl reload apache2**.

Además, necesitamos configurar el módulo **dir** para que busque primeramente un archivo llamado **login.html**, que es como se llama nuestro archivo principal de login.

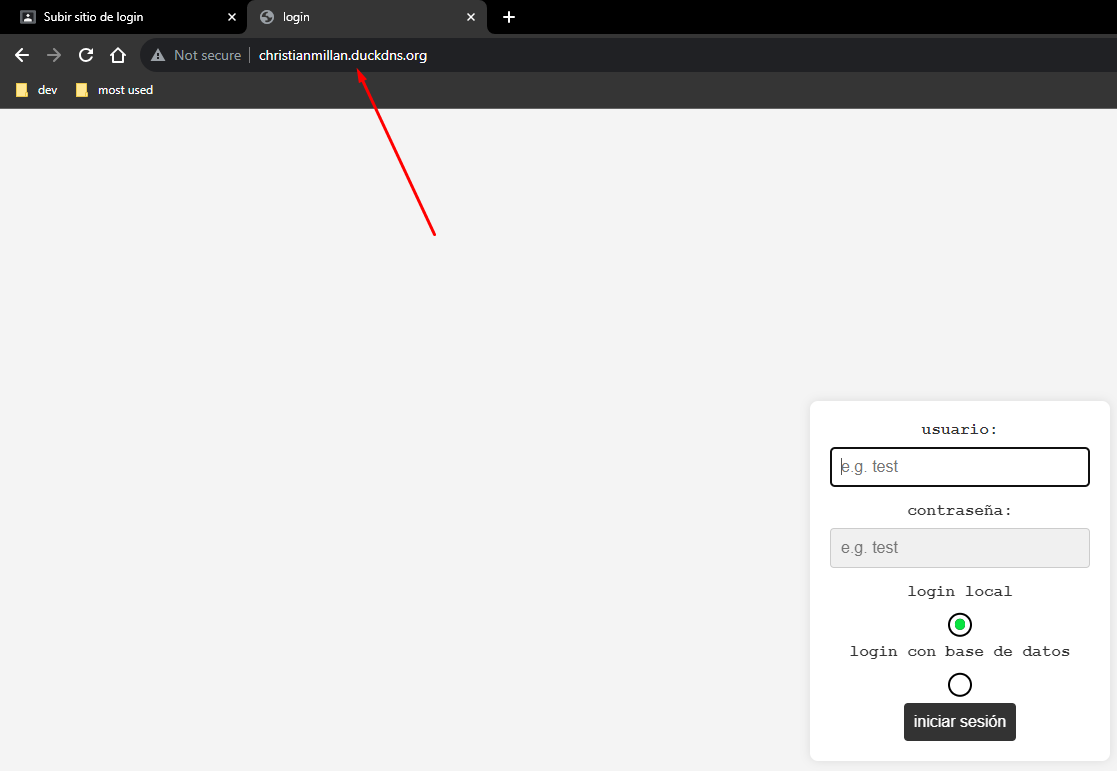


Deshabilitamos el sitio de las prácticas anteriores y habilitamos el actual.



**4. Comprobación de funcionamiento**

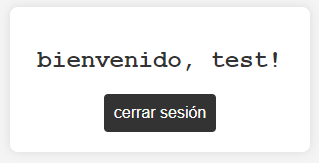
Si introducimos el dominio que creamos en Duck DNS en un navegador web se nos mostrará directamente la página de login.



Introducimos un usuario existente con su contraseña, como se puede ver en los placeholders.



Primero probaremos a iniciar sesión mediante el método local (arrays ficticios). Hacemos clic en el botón de abajo.



Hacemos clic en el botón para cerrar sesión y volver a la página principal. Esta vez marcamos la otra opción de inicio de sesión.



Podemos comprobar que el inicio de sesión se realiza con éxito utilizando los dos métodos.

