
Práctica 1.6

Implantación de Aplicaciones Web

Curso 2025/2026

Índice general

1	Instalación de WordPress en una instancia EC2 de AWS	1
1.1	¿Qué es un Sistema de Gestión de Contenidos (CMS - <i>Content Management System</i>)?	1
1.2	¿Qué es WordPress?	1
1.3	Instalación de WordPress en el directorio raíz	1
1.4	Instalación de WordPress en su propio directorio	4
1.5	Configuración de las <i>security keys</i> de WordPress	8
1.6	Tareas a realizar	9
1.7	Entregables	9
1.7.1	Documento técnico	10
1.7.2	Scripts de Bash	10
2	Referencias	11
3	Licencia	12

1 Instalación de WordPress en una instancia EC2 de AWS

En esta práctica tendremos que realizar la instalación de [WordPress](#) en una instancia EC2 de [Amazon Web Services \(AWS\)](#).

[Amazon Web Services \(AWS\)](#) es una colección de servicios de computación en la nube pública que en conjunto forman una plataforma de computación en la nube, ofrecidas a través de Internet por [Amazon](#).

1.1. ¿Qué es un Sistema de Gestión de Contenidos (CMS - *Content Management System*)?

Un **Sistema de Gestión de Contenidos** (CMS) es un software que permite a los usuarios crear, editar y gestionar de una forma sencilla el contenido de un sitio web. Algunos ejemplos de CMS son: [WordPress](#), [Joomla](#) y [Drupal](#).

1.2. ¿Qué es WordPress?

[WordPress](#) es un sistema de gestión de contenidos (CMS) muy utilizado para crear sitios webs y blogs. Es software libre y gratuito, y está desarrollado en PHP. Además cuenta con una gran cantidad de plugins y temas que permiten añadir nuevas sus funcionalidades y personalizar su diseño de una forma muy sencilla.

1.3. Instalación de WordPress en el directorio raíz

En esta sección se explican los pasos que hay que llevar a cabo para instalar WordPress en directorio raíz de Apache. Por ejemplo: `/var/www/html`.

1. Descargamos la última versión de WordPress con el comando `wget`.

```
1 wget https://wordpress.org/latest.tar.gz -P /tmp
```

El parámetro `-P` indica la ruta donde se guardará el archivo.

2. Descomprimos el archivo `.tar.gz` que acabamos de descargar con el comando `tar`.

```
1 tar -xzvf /tmp/latest.tar.gz -C /tmp
```

Utilizamos los siguientes parámetros:

- `-x`: Indica que queremos extraer el contenido del archivo.
- `-z`: Indica que queremos descomprimir el archivo.
- `-v`: Habilita el modo verboso para mostrar por pantalla el proceso de descompresión.
- `-f`: Se utiliza para indicar cuál es el nombre del archivo de entrada.
- `-C`: Se utiliza para indicar cuál es el directorio destino.

4. El contenido se ha descomprimido en una carpeta que se llama `wordpress`. Ahora, movemos el contenido de `/tmp/wordpress` a `/var/www/html`.

```
1 mv -f /tmp/wordpress/* /var/www/html
```

5. Creamos la base de datos y el usuario para WordPress.

```
1 mysql -u root <<< "DROP DATABASE IF EXISTS $WORDPRESS_DB_NAME"
2 mysql -u root <<< "CREATE DATABASE $WORDPRESS_DB_NAME"
3 mysql -u root <<< "DROP USER IF EXISTS $WORDPRESS_DB_USER@$IP_CLIENTE_MYSQL"
4 mysql -u root <<< "CREATE USER $WORDPRESS_DB_USER@$IP_CLIENTE_MYSQL IDENTIFIED
  BY '$WORDPRESS_DB_PASSWORD'"
5 mysql -u root <<< "GRANT ALL PRIVILEGES ON $WORDPRESS_DB_NAME.* TO
  $WORDPRESS_DB_USER@$IP_CLIENTE_MYSQL"
```

Tenga en cuenta que las variables `$WORDPRESS_DB_NAME`, `$WORDPRESS_DB_USER`, `$WORDPRESS_DB_PASSWORD` y `$IP_CLIENTE_MYSQL` estarán definidas en el archivo `.env`.

El valor de la variable `$IP_CLIENTE_MYSQL` puede ser:

- `localhost`: Para permitir que el usuario sólo puede conectarse desde el servidor MySQL.
- `%`: Para permitir que el usuario pueda conectarse desde cualquier dirección IP.
- Una dirección IP concreta. Para permitir que el usuario pueda conectarse desde una dirección IP concreta. Ejemplo: `172.31.80.67`.
- Una dirección IP con un comodín. Para permitir que el usuario pueda conectarse desde rango de direcciones IPs. Ejemplo: `172.31.%`.

6. Creamos un archivo de configuración `wp-config.php` a partir del archivo de ejemplo `wp-config-sample.php`.

```
1 cp /var/www/html/wp-config-sample.php /var/www/html/wp-config.php
```

7. En este paso tenemos que configurar las variables de configuración del archivo de configuración de WordPress. El contenido original del archivo `wp-config.php` será similar a este:

```
1 // ** Database settings - You can get this info from your web host ** //
2 /** The name of the database for WordPress */
3 define( 'DB_NAME', 'database_name_here' );
4
5 /** Database username */
6 define( 'DB_USER', 'username_here' );
```

```

7
8 /** Database password */
9 define( 'DB_PASSWORD', 'password_here' );
10
11 /** Database hostname */
12 define( 'DB_HOST', 'localhost' );
13
14 /** Database charset to use in creating database tables. */
15 define( 'DB_CHARSET', 'utf8' );
16
17 /** The database collate type. Don't change this if in doubt. */
18 define( 'DB_COLLATE', '' );

```

Por lo tanto, lo que haremos será reemplazar el texto `database_name_here`, `username_here`, `password_here` y `localhost` por los valores de las variables `$WORDPRESS_DB_NAME`, `$WORDPRESS_DB_USER`, `$WORDPRESS_DB_PASSWORD` y `$WORDPRESS_DB_HOST` respectivamente.

Para realizar este paso utilizaremos el comando `sed`.

```

1 sed -i "s/database_name_here/$WORDPRESS_DB_NAME/" /var/www/html/wp-config.php
2 sed -i "s/username_here/$WORDPRESS_DB_USER/" /var/www/html/wp-config.php
3 sed -i "s/password_here/$WORDPRESS_DB_PASSWORD/" /var/www/html/wp-config.php
4 sed -i "s/localhost/$WORDPRESS_DB_HOST/" /var/www/html/wp-config.php

```

Tenga en cuenta que las variables `$WORDPRESS_DB_NAME`, `$WORDPRESS_DB_USER`, `$WORDPRESS_DB_PASSWORD` y `$WORDPRESS_DB_HOST` estarán definidas en el archivo `.env`.

8. Cambiamos el propietario y el grupo al directorio `/var/www/html`.

```
1 chown -R www-data:www-data /var/www/html/
```

9. Preparamos la configuración para los enlaces permanentes de WordPress. En este paso tendremos que crear un archivo `.htaccess` en el directorio `/var/www/html` con un contenido similar a este.

```

1 # BEGIN WordPress
2 <IfModule mod_rewrite.c>
3 RewriteEngine On
4 RewriteBase /
5 RewriteRule ^index\.php$ - [L]
6 RewriteCond %{REQUEST_FILENAME} !-f
7 RewriteCond %{REQUEST_FILENAME} !-d
8 RewriteRule . /index.php [L]
9 </IfModule>
10 # END WordPress

```

A continuación, se explica cada una de las directivas que se han utilizado:

- `RewriteEngine On`: Habilita el motor de reescritura de URLs.
- `RewriteBase /`: Define la ruta base que se utilizará para las reglas de reescritura.
- `RewriteRule ^index\.php$ - [L]`: Esta regla indica que si la petición que se está evaluando coincide con `index.php` entonces no se realiza ninguna acción. El carácter `-` indica que no se hará ninguna acción y el *flag* `[L]` (*Last*) indica que será la última regla que se aplique en esta evaluación.

- `RewriteCond` `%{REQUEST_FILENAME} !-f`: Esta condición comprueba si la petición que se está evaluando no es un archivo. Si no lo es, entonces se aplica la siguiente regla de reescritura.
- `RewriteCond` `%{REQUEST_FILENAME} !-d`: Esta condición comprueba si la petición que se está evaluando no es un directorio. Si no lo es, entonces se aplica la siguiente regla de reescritura.
- `RewriteRule` `. /index.php [L]`: Esta regla se aplicará cuando se cumplan las dos condiciones anteriores. Si la petición no es un archivo y no es un directorio, entonces se hace una redirección a `index.php`.

Por lo tanto, este archivo hará que todas las peticiones que lleguen a este directorio, si no son un archivo o un directorio entonces se dirigen a `index.php`.

Nota: Tenga en cuenta que para que el servidor web Apache pueda leer el archivo `.htaccess` tendremos que configurar la directiva `AllowOverride` como `AllowOverride All`. Esta configuración se tendrá que realizar en el archivo de configuración del VirtualHost de Apache.

10. Habilitamos el módulo `mod_rewrite` de Apache.

```
1 a2enmod rewrite
```

11. Después de habilitar el módulo deberá reiniciar el servicio de Apache.

```
1 sudo systemctl restart apache2
```

1.4. Instalación de WordPress en su propio directorio

En esta sección se explica los pasos que hay que llevar a cabo para instalar WordPress en su propio directorio. Por ejemplo: `/var/www/html/wordpress`.

1. Descargamos la última versión de WordPress con el comando `wget`.

```
1 wget http://wordpress.org/latest.tar.gz -P /tmp
```

El parámetro `-P` indica la ruta donde se guardará el archivo.

2. Descomprimos el archivo `.tar.gz` que acabamos de descargar con el comando `tar`.

```
1 tar -xzf /tmp/latest.tar.gz -C /tmp
```

Utilizamos los siguientes parámetros:

- `-x`: Indica que queremos extraer el contenido del archivo.
 - `-z`: Indica que queremos descomprimir el archivo.
 - `-v`: Habilita el modo verboso para mostrar por pantalla el proceso de descompresión.
 - `-f`: Se utiliza para indicar cuál es el nombre del archivo de entrada.
 - `-C`: Se utiliza para indicar cuál es el directorio destino.
4. El contenido se ha descomprimido en una carpeta que se llama `wordpress`. Ahora, movemos el contenido de `/tmp/wordpress` a `/var/www/html`.

```
1 mv -f /tmp/wordpress /var/www/html
```

5. Creamos la base de datos y el usuario para WordPress.

```
1 mysql -u root <<< "DROP DATABASE IF EXISTS $WORDPRESS_DB_NAME"
2 mysql -u root <<< "CREATE DATABASE $WORDPRESS_DB_NAME"
3 mysql -u root <<< "DROP USER IF EXISTS $WORDPRESS_DB_USER@$IP_CLIENTE_MYSQL"
4 mysql -u root <<< "CREATE USER $WORDPRESS_DB_USER@$IP_CLIENTE_MYSQL IDENTIFIED
  BY '$WORDPRESS_DB_PASSWORD'"
5 mysql -u root <<< "GRANT ALL PRIVILEGES ON $WORDPRESS_DB_NAME.* TO
  $WORDPRESS_DB_USER@$IP_CLIENTE_MYSQL"
```

Tenga en cuenta que las variables `$WORDPRESS_DB_NAME`, `$WORDPRESS_DB_USER`, `$WORDPRESS_DB_PASSWORD` y `$IP_CLIENTE_MYSQL` estarán definidas en el archivo `.env`.

El valor de la variable `$IP_CLIENTE_MYSQL` puede ser:

- `localhost`: Para permitir que el usuario sólo puede conectarse desde el servidor MySQL.
- `%`: Para permitir que el usuario pueda conectarse desde cualquier dirección IP.
- Una dirección IP concreta. Para permitir que el usuario pueda conectarse desde una dirección IP concreta. Ejemplo: `172.31.80.67`.
- Una dirección IP con un comodín. Para permitir que el usuario pueda conectarse desde rango de direcciones IPs. Ejemplo: `172.31.%`.

6. Creamos un archivo de configuración `wp-config.php` a partir del archivo de ejemplo `wp-config-sample.php`.

```
1 cp /var/www/html/wordpress/wp-config-sample.php /var/www/html/wordpress/wp-config.php
```

7. En este paso tenemos que configurar las variables de configuración del archivo de configuración de WordPress. El contenido original del archivo `wp-config.php` será similar a este:

```
1 // ** Database settings - You can get this info from your web host ** //
2 /** The name of the database for WordPress */
3 define( 'DB_NAME', 'database_name_here' );
4
5 /** Database username */
6 define( 'DB_USER', 'username_here' );
7
8 /** Database password */
9 define( 'DB_PASSWORD', 'password_here' );
10
11 /** Database hostname */
12 define( 'DB_HOST', 'localhost' );
13
14 /** Database charset to use in creating database tables. */
15 define( 'DB_CHARSET', 'utf8' );
16
17 /** The database collate type. Don't change this if in doubt. */
18 define( 'DB_COLLATE', '' );
```

Por lo tanto, lo que haremos será reemplazar el texto `database_name_here`, `username_here`, `password_here` y `localhost` por los valores de las variables `$WORDPRESS_DB_NAME`, `$WORDPRESS_DB_USER`, `$WORDPRESS_DB_PASSWORD` y `$WORDPRESS_DB_HOST` respectivamente.

Para realizar este paso utilizaremos el comando `sed`.

```
1 sed -i "s/database_name_here/$WORDPRESS_DB_NAME/" /var/www/html/wordpress/wp-config.php
2 sed -i "s/username_here/$WORDPRESS_DB_USER/" /var/www/html/wordpress/wp-config.php
3 sed -i "s/password_here/$WORDPRESS_DB_PASSWORD/" /var/www/html/wordpress/wp-config.php
4 sed -i "s/localhost/$WORDPRESS_DB_HOST/" /var/www/html/wordpress/wp-config.php
```

Tenga en cuenta que las variables `$WORDPRESS_DB_NAME`, `$WORDPRESS_DB_USER`, `$WORDPRESS_DB_PASSWORD` y `$WORDPRESS_DB_HOST` estarán definidas en el archivo `.env`.

8. Cuando realizamos la instalación de WordPress en su propio directorio, es necesario configurar estas dos variables de configuración:

- **Dirección de WordPress (`WP_SITEURL`):** Es la URL que incluye el directorio donde está instalado el código fuente de WordPress.
- **Dirección del sitio (`WP_HOME`):** Es la URL que queremos que usen los usuarios para acceder a WordPress.

Las variables de **Dirección de WordPress (`WP_SITEURL`)** y **Dirección del sitio (`WP_HOME`)** se pueden configurar:

- Desde la sección de **Ajustes -> Generales** del **panel de administración** de WordPress.
- Desde el archivo de configuración `wp-config.php`.

Si hemos realizado la instalación de **WordPress** en el directorio `wordpress` tendremos que asignarles los siguientes valores:

- **Dirección de WordPress (`WP_SITEURL`):** `https://NOMBRE_DE_DOMINIO/wordpress`
- **Dirección del sitio (`WP_HOME`):** `https://NOMBRE_DE_DOMINIO`

Donde `NOMBRE_DE_DOMINIO` será el nombre de dominio que ha reservado para su sitio web. Observe que hemos utilizado `https`, porque su sitio web tendrá que utilizar un certificado HTTPS.

Si quiere realizar pruebas sin tener un nombre de dominio, puede hacerlas utilizando en su lugar la dirección IP pública del servidor web.

- **Dirección de WordPress (`WP_SITEURL`):** `http://IP_PUBLICA_SERVIDOR_WEB/wordpress`
- **Dirección del sitio (`WP_HOME`):** `http://IP_PUBLICA_SERVIDOR_WEB`

Para automatizar la configuración de las variables `WP_SITEURL` y `WP_HOME` podemos utilizar el comando `sed`

```
1 sed -i "/DB_COLLATE/a define('WP_SITEURL', 'https://$CERTIFICATE_DOMAIN/wordpress');" /var/www/html/wordpress/wp-config.php
2 sed -i "/WP_SITEURL/a define('WP_HOME', 'https://$CERTIFICATE_DOMAIN');" /var/www/html/wordpress/wp-config.php
```

Tenga en cuenta que la variable `$CERTIFICATE_DOMAIN` estará definida en el archivo `.env`.

9. Copiamos el archivo `/var/www/html/wordpress/index.php` a `/var/www/html`.

```
1 cp /var/www/html/wordpress/index.php /var/www/html
```

10. El contenido original del archivo `index.php` será similar a este:

```
1 /** Loads the WordPress Environment and Template */
2 require( dirname( __FILE__ ) . '/wp-blog-header.php' );
```

Y tenemos que reemplazarlo por este otro:

```
1 /** Loads the WordPress Environment and Template */
2 require( dirname( __FILE__ ) . '/wordpress/wp-blog-header.php' );
```

Donde `wordpress` es el directorio donde se encuentra el código fuente de WordPress que hemos descomprimido en pasos anteriores.

Por lo tanto, para realizar este paso utilizaremos el comando `sed`.

```
1 sed -i "s#wp-blog-header.php#wordpress/wp-blog-header.php#" /var/www/html/index.php
```

11. Preparamos la configuración para los enlaces permanentes de WordPress. En este paso tendremos que crear un archivo `.htaccess` en el directorio `/var/www/html` con un contenido similar a este.

```
1 # BEGIN WordPress
2 <IfModule mod_rewrite.c>
3 RewriteEngine On
4 RewriteBase /
5 RewriteRule ^index\.php$ - [L]
6 RewriteCond %{REQUEST_FILENAME} !-f
7 RewriteCond %{REQUEST_FILENAME} !-d
8 RewriteRule . /index.php [L]
9 </IfModule>
10 # END WordPress
```

A continuación, se explica cada una de las directivas que se han utilizado:

- `RewriteEngine On`: Habilita el motor de reescritura de URLs.
- `RewriteBase /`: Define la ruta base que se utilizará para las reglas de reescritura.
- `RewriteRule ^index\.php$ - [L]`: Esta regla indica que si la petición que se está evaluando coincide con `index.php` entonces no se realiza ninguna acción. El carácter `-` indica que no se hará ninguna acción y el *flag* `[L]` (*Last*) indica que será la última regla que se aplique en esta evaluación.
- `RewriteCond %{REQUEST_FILENAME} !-f`: Esta condición comprueba si la petición que se está evaluando no es un archivo. Si no lo es, entonces se aplica la siguiente regla de reescritura.
- `RewriteCond %{REQUEST_FILENAME} !-d`: Esta condición comprueba si la petición que se está evaluando no es un directorio. Si no lo es, entonces se aplica la siguiente regla de reescritura.
- `RewriteRule . /index.php [L]`: Esta regla se aplicará cuando se cumplan las dos condiciones anteriores. Si la petición no es un archivo y no es un directorio, entonces se hace una redirección a `index.php`.

Por lo tanto, este archivo hará que todas las peticiones que lleguen a este directorio, si no son un archivo o un directorio entonces se redirigen a `index.php`.

Nota: Tenga en cuenta que para que el servidor web Apache pueda leer el archivo `.htaccess` tendremos que configurar la directiva `AllowOverride` como `AllowOverride All`. Esta configuración se tendrá que realizar en el archivo de configuración del VirtualHost de Apache.

12. Habilitamos el módulo `mod_rewrite` de Apache.

```
1 a2enmod rewrite
```

13. Después de habilitar el módulo deberá reiniciar el servicio de Apache.

```
1 sudo systemctl restart apache2
```

1.5. Configuración de las *security keys* de WordPress

El archivo de configuración `wp-config.php` de WordPress incluye una sección para configurar unas *security keys* que se utilizan para mejorar la seguridad de las sesiones y generar cookies seguras.

Se recomienda que cada instalación genere unas claves que sean únicas para esa instalación.

1. En primer lugar, eliminamos los valores por defecto de las *security keys* que aparecen en el archivo de configuración.

```
1 sed -i "/AUTH_KEY/d" /var/www/html/wordpress/wp-config.php
2 sed -i "/SECURE_AUTH_KEY/d" /var/www/html/wordpress/wp-config.php
3 sed -i "/LOGGED_IN_KEY/d" /var/www/html/wordpress/wp-config.php
4 sed -i "/NONCE_KEY/d" /var/www/html/wordpress/wp-config.php
5 sed -i "/AUTH_SALT/d" /var/www/html/wordpress/wp-config.php
6 sed -i "/SECURE_AUTH_SALT/d" /var/www/html/wordpress/wp-config.php
7 sed -i "/LOGGED_IN_SALT/d" /var/www/html/wordpress/wp-config.php
8 sed -i "/NONCE_SALT/d" /var/www/html/wordpress/wp-config.php
```

2. Vamos a hacer uso de la API que nos ofrece WordPress para generar unas *security keys*. La URL de la API es:

- <https://api.wordpress.org/secret-key/1.1/salt/>

Hacemos una llamada a la API de wordpress para obtener las *security keys* y almacenamos el resultado en una variable de entorno.

```
1 SECURITY_KEYS=$(curl https://api.wordpress.org/secret-key/1.1/salt/)
```

3. Las nuevas *security keys* que acabamos de generar pueden contener el carácter `/` y este carácter puede darnos problemas a la hora de utilizar el comando `sed` para añadirlas al archivo de configuración.

Para evitar posibles problemas con el carácter `/` vamos a reemplazarlo por el carácter `_`.

```
1 SECURITY_KEYS=$(echo $SECURITY_KEYS | tr / _)
```

4. Añadimos las security keys al archivo de configuración.

```
1 sed -i "/@-/a $SECURITY_KEYS" /var/www/html/wordpress/wp-config.php
```

5. Cambiamos el propietario y el grupo al directorio `/var/www/html`.

```
1 chown -R www-data:www-data /var/www/html/
```

1.6. Tareas a realizar

En esta práctica tendremos que realizar la instalación de [WordPress](#) en una instancia EC2 de [Amazon Web Services \(AWS\)](#).

A continuación se describen **muy brevemente** algunas de las tareas que tendrá que realizar.

1. Crea una instancia EC2 en AWS.
2. La **Amazon Machine Image (AMI)** que vamos a seleccionar para esta práctica será una **Community AMI** con la última versión de **Ubuntu Server**.
3. Cuando esté creando la instancia deberá configurar los puertos que estarán abiertos para poder conectarnos por SSH y para poder acceder por HTTP/HTTPS.
 - SSH (TCP)
 - HTTP (TCP)
 - HTTPS (TCP)
4. Crea un par de claves (pública y privada) para conectar por SSH con la instancia. También puedes hacer uso de las claves que te proporciona AWS Academy (*vockey.pem*).
5. Crea una dirección **IP elástica** y asígnala a la instancia EC2.
6. Una vez que haya iniciado su instancia deberá hacer uso de los **scripts de bash** que diseñó en las prácticas anteriores para automatizar la instalación de la pila LAMP.
7. Automatice la instalación de [WordPress](#).
8. Busque cuál es la dirección IP elástica de su instancia y compruebe que puede acceder a ella desde un navegador web.

1.7. Entregables

Deberá crear un repositorio en [GitHub](#) con el nombre de la práctica y añadir al profesor como colaborador.

El repositorio debe tener el siguiente contenido:

- Un **documento técnico** con la descripción de todos los pasos que se han llevado a cabo.
- Los **scripts de Bash** que se han utilizado para automatizar la instalación y configuración de la pila LAMP, así como de [WordPress](#).

Además del contenido anterior puede ser necesario crear otros archivos de configuración. A continuación se muestra un ejemplo de cómo puede ser la estructura del repositorio:

```
1  .|—
2  README.md|—
3  conf|
4    └─ 000-default.conf|—
5  htaccess|
6    └─ .htaccess|—
7  scripts|—
8    .env|—
9    install_lamp.sh|—
10   setup_letsencrypt_https.sh|—
11   deploy_wordpress_root_directory.sh|—
12   deploy_wordpress_own_directory.sh
```

1.7.1. Documento técnico

El documento técnico `README.md` tiene que estar escrito en [Markdown](#) y debe incluir **como mínimo** los siguientes contenidos:

- Descripción del proceso de instalación de la instalación de [WordPress](#).

1.7.2. Scripts de Bash

El directorio `scripts` debe incluir los siguientes archivos:

- `.env`: Este archivo contiene todas las variables de configuración que se utilizarán en los scripts de Bash.
- `install_lamp.sh`: Script de Bash con la automatización del proceso de instalación de la pila LAMP.
- `setup_letsencrypt_https.sh`: Script de Bash con la automatización del proceso de solicitar un certificado SSL/TLS de Let's Encrypt y configurarlo en el servidor web Apache.
- `deploy_wordpress_root_directory.sh`: Script de Bash con la automatización del proceso de instalación de WordPress sobre el directorio raíz `/var/www/html`.
- `deploy_wordpress_own_directory.sh`: Script de Bash con la automatización del proceso de instalación de WordPress sobre su propio directorio. Por ejemplo: `/var/www/html/wordpress`.

2 Referencias

- [Amazon Web Services](#)
- [Amazon Web Services en Wikipedia](#)
- [¿Qué es Amazon EC2?](#)
- [SSH to an EC2 instance from VS Code.](#)
- [Remote Development using SSH with Visual Studio Code.](#)
- [Sistema de Gestión de Contenidos \(CMS - Content Management System\)](#)
- [Giving WordPress its own directory](#)
- [Breve guía de introducción a WordPress](#)

3 Licencia

Esta página forma parte del curso Implantación de Aplicaciones Web de José Juan Sánchez Hernández y su contenido se distribuye bajo una licencia Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional.