

## Método.2. En cualquier aplicación wordpress

### 1.Actualizamos el servidor

Desde el servidor ejecutamos `apt update`, y éste actualizará el servidor. También realizamos la instalación de nano con el comando `apt install nano`, esta aplicación nos permitirá editar ficheros de texto con facilidad.

```
root@srv1:~# apt update
```

Figura 54: comando de actualización en servidor.

```
root@srv1:~/proyecto# apt install nano
```

Figura 55: comando de instalación de aplicación nano.

### 2. Directorio para el proyecto

Generamos una carpeta en el interior de este equipo con el comando `mkdir proyecto`, donde guardaremos los archivos y ficheros correspondientes además de realizar las correspondientes operaciones. Accederemos a este mismo con el comando `cd proyecto`.

```
root@srv1:~# mkdir proyecto
root@srv1:~# cd proyecto/
root@srv1:~/proyecto#
```

Figura 56: creación de carpeta proyecto.

### 3. Instalación de docker

Entonces continuamos con el comando de instalación de docker, `apt install docker docker-compose docker.io`

```
root@srv1:~/proyecto# apt install docker docker-compose docker.io
```

Figura 57: comando de instalación de aplicación docker

### 4. Ejecutamos contenedor Portainer

Por si fuera necesario, instalaremos el contenedor Portainer. Este nos ofrece una manera de visualizar los contenedores, imágenes y más detalles de la ejecución de docker.

Para ello implementaremos el siguiente comando, con el cual logramos la ejecución de Portainer en el puerto 9000.

```
docker run -d -p 8100:9000 --name=portainer --restart=always -v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-ce:latest
```

```
root@srv1:~/proyecto# docker run -d -p 8100:9000 --name=portainer --restart=always -v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-ce:latest
```

Figura 58: comando de creación y ejecución container Portainer

### 5. Ejecución de docker-compose.yml

Con la documentación que nos ofrece la imagen oficial de wordpress generamos el `docker-compose.yml`, este será necesario para generar los contenedores.

```

version: '3.1'

services:
  wordpress:
    image: wordpress
    restart: always
    ports:
      - 8080:80
    environment:
      WORDPRESS_DB_HOST: db
      WORDPRESS_DB_USER: exampleuser
      WORDPRESS_DB_PASSWORD: examplepass
      WORDPRESS_DB_NAME: exampledb
    volumes:
      - wordpress:/var/www/html

  db:
    image: mysql:5.7
    restart: always
    environment:
      MYSQL_DATABASE: exampledb
      MYSQL_USER: exampleuser
      MYSQL_PASSWORD: examplepass
      MYSQL_RANDOM_ROOT_PASSWORD: '1'
    volumes:
      - db:/var/lib/mysql

volumes:
  wordpress:
  db:

```

Figura 61: documento docker-compose.yml método 2

Para generar éste manualmente, con el comando `nano docker-compose.yml` directamente, y nos pondrá a editarlo. Introducimos la información correspondiente, y lo guardamos pulsando Ctrl + O.

A continuación se ejecuta el documento docker-compose que generará un contenedor Wordpress en el puerto 8080 y un contenedor mysql. El comando correspondiente que debe ser usado es `docker-compose -f {nombre del fichero} up` en el directorio donde se encuentre.

```

root@srv1:~/proyecto# docker-compose -f docker-compose-propio.yml up -d
Creating network "proyecto_default" with the default driver
Creating volume "proyecto_wordpress" with default driver
Creating volume "proyecto_db" with default driver
Creating proyecto_wordpress_1 ... done
Creating proyecto_db_1 ... done

```

## 6. Inicio de la aplicación Wordpress

Completado el paso anterior, se podrá acceder al contenedor Wordpress con el enlace `{servidor o equipo local}:8080/` nada más acceder a este link nos enviará a la configuración correspondiente de Wordpress. Estas configuraciones no tendrán mucha importancia, pues se sobrescribirá más adelante.

Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Do not worry, you can always change these settings later.

Site Title: Heiko Shop 2

Username: alberto  
Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

Password: p@ssw0rd  
Very weak  
Important: You will need this password to log in. Please store it in a secure location.

Confirm Password: ☒ Confirm use of weak password

Your Email: alberto.cantero@iesjulianmaria  
Double-check your email address before continuing.

Search engine visibility: ☒ Discourage search engines from indexing this site  
It is up to search engines to honor this request.

Install WordPress

Figura 62: *formulario configuración Wordpress*

## 7. Instalación del Plugin All-in-One WP Migration

Tras completar la configuración correspondiente e ingresar el usuario y contraseña que se ha indicado, estaremos en la interfaz de Wordpress, en el menú de la izquierda se encontrará una opción llamada Plugins, en ésta seleccionamos Add New / Añadir Nuevo.

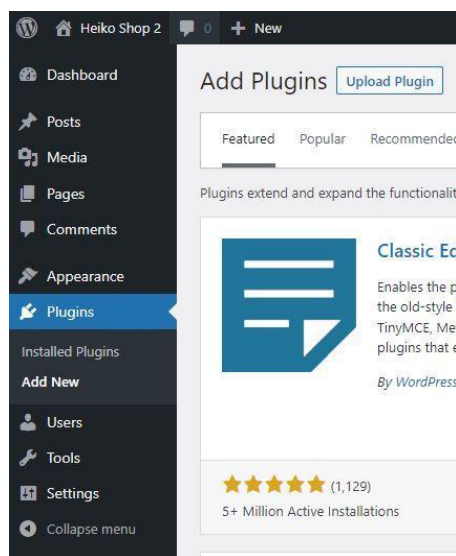


Figura 63: *barra lateral izquierda Wordpress*

En el buscador de la página en la que nos encontramos escribimos "All-in-One WP Migration" y seleccionamos el correspondiente cuyo autor es "ServMask", como se ve en la figura 23.

Después de instalarlo y activarlo recargamos la página, y en el menú de la izquierda se habrá añadido una nueva opción para operar con el plugin, la cual tendrá una opción de importar.

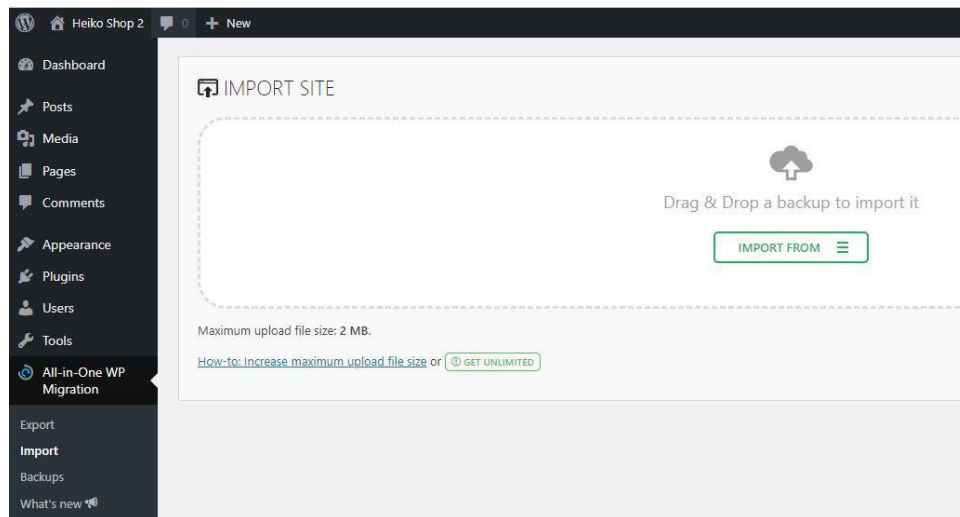


Figura 64: interfaz plugin All-In-One

Antes de continuar con la importación, hay un paso más que se debe llevar a cabo, pues el archivo a importar es de un tamaño mayor a el permitido por Wordpress, como se puede ver el máximo de capacidad que permite es de 2 MB.

### 3. Configuración capacidad máxima de Wordpress

Para lograr que Wordpress nos permita importarlo tendremos que realizar un cambio en un documento que se encuentra en el contenedor de éste, el cambio permitirá aumentar el máximo a 500MB.

Para lograrlo, primero accederemos al contenedor, necesitaremos el ID que se generó para el contenedor. Para saber cual es el ID, se ejecutará `docker ps` y este comando nos mostrará los contenedores en ejecución y los IDs de los mismos. Entonces implementamos el ID o nombre del contenedor Wordpress en el comando siguiente `docker exec -it {nombre o id del contenedor} /bin/bash`. Tras la ejecución de este comando, estaremos en el interior del contenedor docker de Wordpress.

```
root@srv1:~/proyecto# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
7580d9fee805   mysql:5.7 "docker-entrypoint.s..." 7 minutes ago Up 7 minutes   3306/tcp, 33060/tcp               proyecto_db_1
52a9dd475a0b   wordpress "docker-entrypoint.s..." 7 minutes ago Up 7 minutes   0.0.0.0:8080->80/tcp              proyecto_wordpress_1
root@srv1:~/proyecto# docker exec -it proyecto_wordpress_1 /bin/bash
root@52a9dd475a0b:/var/www/html#
```

Figura 65: listado de contenedores y acceso exec

Desde aquí realizaremos nuevamente el comando `apt update`, para actualizar el contenedor, y tras ello realizaremos `apt install nano`, esto nos permitirá poder editar el interior del fichero. Este documento se encuentra en la carpeta `var/www/html/` que posiblemente será en la que estemos nada más acceder.

```
root@52a9dd475a0b:/var/www/html# apt update
```

Figura 66: comando de actualización en contenedor.

Entonces se realizan las modificaciones correspondientes con el comando nano y el documento .htaccess (el punto es necesario). En el interior de este insertamos las siguientes líneas de código, las cuales nos aumentaran la capacidad de los archivos que Wordpress permite importar, y guardamos con Ctrl + O .

```
php_value upload_max_filesize 500M
php_value post_max_size 500M
php_value memory_limit 500M
php_value max_execution_time 0
php_value max_input_time 0
```

```
GNU nano 5.4 .htaccess
# BEGIN WordPress
# The directives (lines) between "BEGIN WordPress" and
# dynamically generated, and should only be modified via
# Any changes to the directives between these markers will
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteRule .* - [E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]
RewriteBase /
RewriteRule ^index\.php$ - [L]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule . /index.php [L]
</IfModule>

php_value upload_max_filesize 500M
php_value post_max_size 500M
php_value memory_limit 500M
php_value max_execution_time 0
php_value max_input_time 0

# END WordPress
```

El resultado es un aumento del máximo permitido a 500 MB

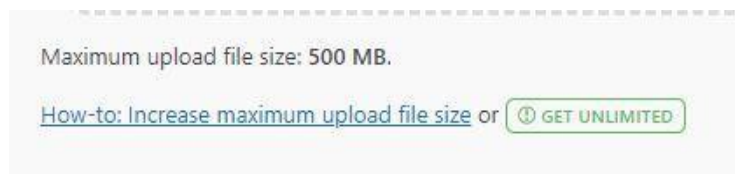


Figura 68: resultado de la modificación realizada.

## 8. Importación final de la página

Desde la página descrita anteriormente del plugin All-in-One WP Migration, se realiza la importación con el archivo que se encuentra en (ProyectoHeikoShop/Instalacion/metodo2\_wordpress/CopiaTienda.wpress)

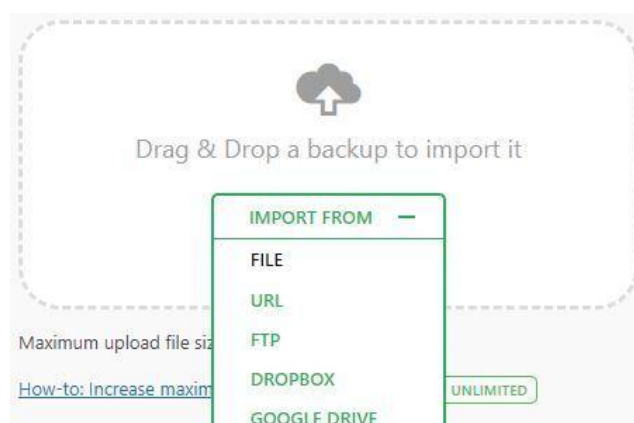



Figura 69: interfaz de importación de All-In-One

▼ hoy (1)

 CopiaTienda.wordpress	18/05/2023 19:38	Archivo WPRESS	338,272 KB
---	------------------	----------------	------------

> al principio de esta semana (1)

Figura 70: *Fichero copia de proyecto*

Tras esto se comenzará a importar y saldrá un aviso que advertirá que la página actual se sobrescribirá con la que estamos importando, seleccionamos proceder.

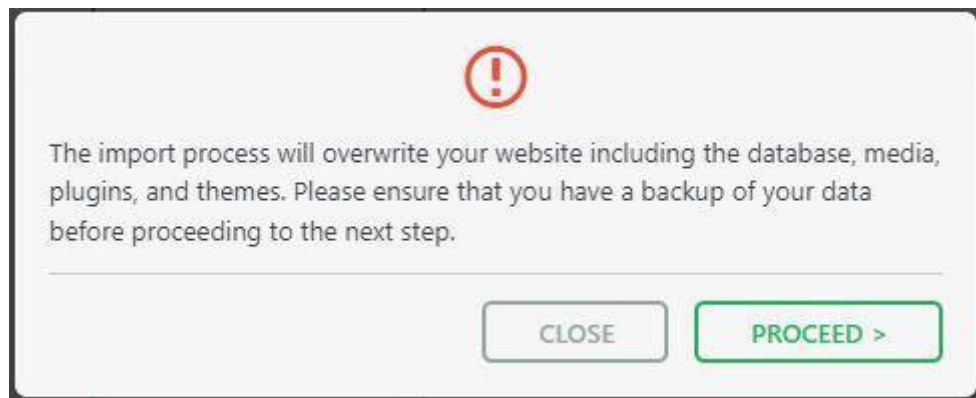


Figura 71: *advertencia plugin All-In-One*

Se finaliza el proceso, logrando así una copia exacta de la web. Si se quiere acceder al wp-admin de esta página, será necesario el usuario y contraseña de la página original (usuario:allberto,contraseña:ProyectoAlbertoDaw05).