UNIDAD 5: INSTALACIÓN

1. MARCOS PHP

Un marco es una herramienta que proporciona una serie de módulos que ayudan a organizar y desarrollar un producto de software.
En el caso concreto de los frameworks PHP, la mayoría de ellos proporcionan una serie de comandos o herramientas para crear proyectos con una estructura determinada (normalmente, siguiendo el patrón MVC que veremos más adelante), de forma que ya dan una base de trabajo realizado, y facilidades para poder crear el modelo de datos, la conexión a la base de datos, las rutas de las diferentes secciones de la aplicación, etc.

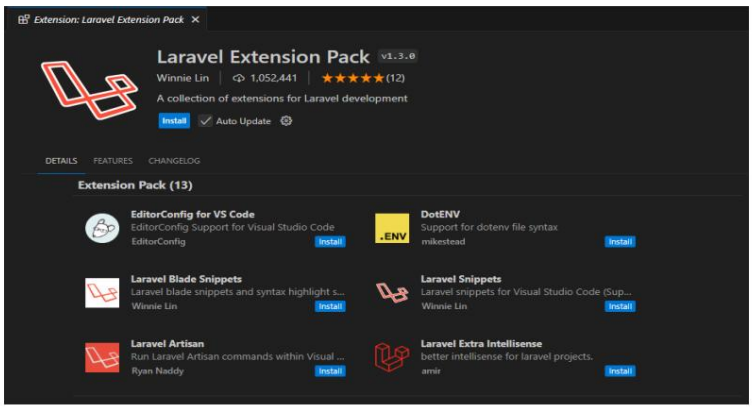
1.1 Recursos previos

Al trabajar con Laravel, necesitamos tener previamente instalados en nuestro sistema una serie de recursos software, como son:
1. Un IDE (entorno de desarrollo) con el que poder editar el código de nuestros proyectos. Nosotros utilizaremos Visual Studio Code en estas notas, aunque existen otras alternativas similares, como PHPStorm, Sublime Texto, etc.
2. Un servidor web que admita PHP.
3. Un servidor de base de datos donde almacenar la información de nuestras aplicaciones. Utilizaremos un servidor MariaDB/MySQL.
4. PHP
5. El framework Laravel en sí.
6. Además, necesitaremos el administrador de paquetes 'npm' para instalar las dependencias del lado del cliente en Laravel proyectos.

En este documento tienes los pasos para instalar cada uno de estos elementos mediante un docker.

1.2. Visual Studio Code: extensiones útiles

Cuando utilizamos Visual Studio Code para desarrollar aplicaciones en una determinada tecnología (Laravel, Node, etc.), puede ser conveniente instalar algún plugin o extensión que facilite el desarrollo de dichas aplicaciones.
A través de estas extensiones podemos, por ejemplo, resaltar la sintaxis de los archivos que editamos, ayudarnos a autocompletar código, etc.
En el caso de Laravel, podemos ir a la sección Extensiones del menú izquierdo de Visual Studio Code, y buscar la extensión Laravel Extension Pack. Luego, hacer clic en el botón Instalar para instalar esta extensión, que a su vez contiene un paquete de extensiones útiles para desarrollar aplicaciones Laravel.



En concreto, podemos destacar la sintaxis de las vistas HTML que realizamos con el motor de plantillas Blade, autocompletar determinadas partes del código, autocorregir determinados errores de código, etc.

También instale las extensiones: Dev Containers y Docker si desea administrar Docker desde Visual Studio Código

2. PREPARACIÓN DEL ENTORNO (opcional)

1. Cree una máquina virtual Ubuntu 22: ubuntu-22.04.2-desktop-amd64 2. Instale Docker Engine

Puedes descargar una máquina virtual Ubuntu 22 desde Teams con Docker Engine instalado. (usuario:marta y contraseña: 123). Así que puedes saltarte este paso e ir directamente a 3. INSTALACIÓN DE LARAVEL DOCKER Y CREACIÓN DE CONTENEDORES

2.1. Métodos de instalación

Instalar Docker Engine en Ubuntu | Documentación de Docker

Puedes instalar Docker Engine de diferentes maneras, según tus necesidades:

1. Docker Engine viene incluido con Docker Desktop para Linux. Esta es la forma más fácil y rápida. Manera de empezar.
2. Configure e instale Docker Engine desde Docker apto repositorio.
3. Instalarlo manualmente y gestionar las actualizaciones manualmente.
4. Utilice un guión de conveniencia. Sólo recomendado para entornos de prueba y desarrollo.

Utilizaremos el método 2n.

2.1.1. Instalación mediante el repositorio apt

Instalar Docker Engine en Ubuntu | Documentación de Docker

Antes de instalar Docker Engine por primera vez en una nueva máquina host, debe configurar el repositorio de Docker. Luego, puede instalar y actualizar Docker desde el repositorio.

Configurar el repositorio

• Actualice el índice de paquetes de apt e instale los paquetes para permitir que apt use un repositorio HTTPS:

```
$ sudo apt-get actualización

$ sudo apt-get install certificados ca curl gnupg
```

• Agregue la clave GPG oficial de Docker:

```
$ sudo install -m 0755 -d /etc/apt/keyrings

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg
--dearmor -o /etc/apt/keyrings/docker.gpg

$ sudo chmod a+r /etc/apt/keyrings/docker.gpg
```

• Utilice el siguiente comando para configurar el repositorio:

```
$ eco \
    "deb [arch="$(dpkg --print-architecture)" firmado-
por=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
    "$(. /etc/os-release && echo "$VERSION_CODENAME")" estable" | \
    sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Nota

Si usa una distribución derivada de Ubuntu, como Linux Mint, es posible que necesite usar UBUNTU_CODENAME en lugar de VERSION_CODENAME.

Instalar Docker Engine

• Actualizar el índice del paquete apt :

```
$ sudo apt-get actualización
```

• Instalar Docker Engine, contenedord y Docker Compose.

Para instalar la última versión, ejecute:

```
$ sudo apt-get install docker-ce docker-ce-cli contenedord.io docker-buildx-
complemento docker-compose-plugin
```

- Verifique que la instalación de Docker Engine sea exitosa ejecutando la imagen hello-world .

```
$ sudo docker run hola-mundo
```

Este comando descarga una imagen de prueba y la ejecuta en un contenedor. Cuando el contenedor se ejecuta, imprime un mensaje de confirmación y sale.
Ahora ha instalado e iniciado Docker Engine correctamente.

Postinstalación

Pasos posteriores a la instalación de Docker Engine en Linux | Documentación de Docker

¿Recibes errores al intentar ejecutar sin root? Si ejecutas “docker ps”… No tienes permiso

El grupo de usuarios de Docker existe, pero no contiene usuarios, por lo que se requiere que utilice sudo para ejecutar comandos de Docker. Continúe permitiendo que los usuarios sin privilegios ejecuten comandos de Docker y otros pasos de configuración opcionales.

Para crear el grupo de Docker y agregar su usuario:

- Crear el grupo de Docker .

```
$ sudo groupadd docker (Este paso no es necesario porque el grupo tiene
ya fue creado)
```

- Agregue su usuario al grupo Docker .

```
$ sudo usermod -aG docker $USUARIO
```

- Cierre sesión y vuelva a iniciarla para que se vuelva a evaluar su membresía del grupo.

  Si está ejecutando Linux en una máquina virtual, puede ser necesario reiniciar la máquina virtual para que los cambios surtan efecto.

También puede ejecutar el siguiente comando para activar los cambios en los grupos:

```
$ nuevo grupo docker
```

- Verifique que pueda ejecutar comandos de Docker sin sudo.

```
$ docker run hola-mundo
```

3. INSTALACIÓN DE LARAVEL DOCKER Y CREACIÓN DE CONTENEDORES:

Crea la carpeta laravel en /home/user y copia los archivos de Docker:

carpeta de configuración con

Información de docker-compose:

• db: network: 172.21.0.2 Este
es el contenedor con los datos en la base de datos.

• myapp: network: 172.21.0.3 Este
es el contenedor con el proyecto Laravel. Será
accesible en el puerto http://localhost:8010

• phpmyadmin: red: 172.21.0.4 Este es el
contenedor con phpMyadmin como sistema de gestión de base de datos. Será
accesible en el puerto http://localhost:8005

Abre una terminal 1, ve a la carpeta laravel y ejecuta:

```
$ docker compone
```

Deja de “intentar conectarse a la base de datos”
Acceso denegado para el usuario 'marta'@'172.21.0.3' (usando contraseña: NO) significa que la
máquina Laravel va a utilizar el usuario marta para conectarse a la máquina mysql.

Da un error de acceso para el usuario marta, así que vamos a solucionarlo.

No cierre la terminal 1.

Abrir una nueva terminal 2,

Vaya a la carpeta de laravel e ingrese al contenedor de laravel en modo interactivo ejecutando:

```
$ docker exec -it laravel-db-1 bash
```

y luego....

```
bash-4.2#mysql -u raíz -p
```

contraseña= '' (como en db.env)

mysql>crear usuario marta@172.21.0.3 identificado por ''; //172.21.0.3 ip de docker

mysql>conceder todos los privilegios en *.* a marta@172.21.0.3;

mysql>flush privilegios;

Cerrar terminal 2

Regresa a la terminal 1 presiona Ctrl + C para detener los contenedores.

Escribe de nuevo:

$ docker compone

No cierres esta terminal mientras estés trabajando con tu proyecto laravel

Nuestro proyecto Laravel está funcionando en http://localhost:8010/
Pero con un error:

Esto se debe a que tenemos que configurar la base de datos.

Abra la carpeta de su proyecto “my-project” (dentro de laravel) con Visual Studio Code.

En config/database.php
'default' => env('DB_CONNECTION', 'mysql'),

En .env
CONEXIÓN_BD=mysql
HOST_BD=172.21.0.2
PUERTO_BD=3306
BASE_DATOS_BD=BD_myapp
NOMBRE_USUARIO_BD=marta
CONTRASEÑA_BD=

Desde otra terminal: docker exec -it laravel-myapp-1 bash php artisan migrants

Y puedes cerrar la terminal

Entonces tu proyecto está funcionando:

Para acceder a phpmyadmin: http://localhost:8005/

Servidor: 172.21.0.2
Nombre de usuario:
root Contraseña: "

Si abrimos la carpeta laravel, podemos ver que hay 2 nuevas carpetas: datos y my-project. Son los volúmenes de la base de datos y del proyecto Laravel respectivamente.

Para detener el proyecto, desde la terminal abierta:

```
$ docker componer abajo
```

Cambiar el nombre de la carpeta my-project a biblioteca

Crea un duplicado de la carpeta biblioteca y nómbralo blog

Trabajaremos con dos proyectos durante el curso:

- Biblioteca: Proyecto sobre libros y autores. Este proyecto lo haremos juntos. Os haré un vídeo con todas las prácticas y lo repetiréis en casa.
- blog: Este será el proyecto que entregarás cada quince días.
  IMPORTANTE: Y tendrás que traerlo para el examen, ya que será una práctica de validación obligatoria en el examen.

Para trabajar con ambos proyectos, debes cambiar el nombre del proyecto con el que quieres trabajar en docker-compose.yml. Por ejemplo, si quieres trabajar con la biblioteca:

En docker-compose.yml cambie:

```
volúmenes:
    - './mi-proyecto:/aplicación'
```

Para:

```
volúmenes:
    - './biblioteca:/aplicación'
```