Creando MauCap

Mauricio Nicolas Adame

Lic. Informática

Comercio Electrónico

Entorno de desarrollo

Creación de carpeta con laravel y breeze

Requisitos previos

Antes de comenzar, **asegúrate** de tener instalados los siguientes requisitos en tu máquina:

- Actualizar **paquetes.**

  ```
  sudo apt update
  ```

- **PHP**: Versión 8.0 o **superior**.

  ```
  sudo apt install -y software-properties-common

  sudo add-apt-repository ppa:ondrej/php

  sudo apt update

  sudo apt update sudo apt install -y php8.0
  ```

- **Composer**: Un manejador de dependencias para **PHP**.

  ```
  sudo apt install -y curl php-cli php-mbstring unzip

  curl -sS https://getcomposer.org/installer | php

  sudo mv composer.phar /usr/local/bin/composer
  ```

- **Node.js** y **NPM**: Para manejar dependencias de frontend.

  ```
  curl -sL https://deb.nodesource.com/setup_16.x | sudo -E bash -

  sudo apt install -y nodejs
  ```

- **MySQL** o cualquier base de datos compatible.

  ```
  sudo apt install -y mysql-server

  sudo mysql_secure_installation
  ```

Instalar Laravel

Instalar Laravel usando **Composer**:

```
composer create-project --prefer-dist laravel/laravel ecommerce_caps
```

Accede al **directorio** del **proyecto**:

```
cd /Documentos/ecommerce_caps
```

1: Configurar tu entorno

1.1.Configura el archivo **.env**

```
cp .env.example .env
```

1.2. Genera una clave de **aplicación**

```
php artisan key:generate
```

1.3. Configura la **base** de **datos**:

Abre el archivo .env y configura las credenciales de tu base de datos:

```
DB_CONNECTION=mysql
DB_HOST=db
DB_PORT=3306
DB_DATABASE=ecommerce_caps
DB_USERNAME=root
DB_PASSWORD=root
```

1.1: Instalar Laravel Breeze

1.1.1 Ejecuta el siguiente comando para instalar **Breeze**:

```
composer require laravel/breeze --dev
```

1.1.2 Ejecuta el instalador de **Breeze**:

Instala las vistas de autenticación de Breeze usando el siguiente comando:

```
php artisan breeze:install
```

*Laravel Breeze instalará las rutas, controladores, vistas y archivos necesarios para la autenticación básica.

1.1.3 Instala las dependencias de **NPM**:

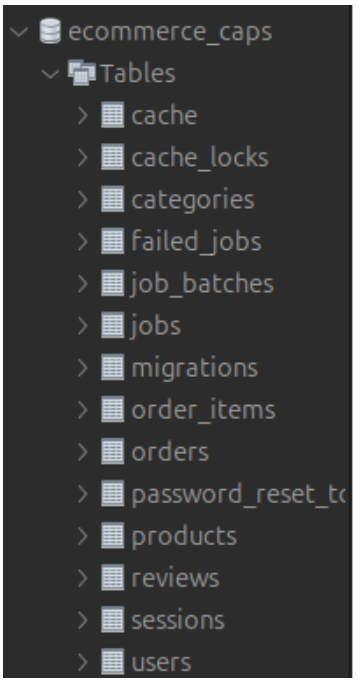
Ahora, instala las dependencias de NPM y construye los activos:

```
npm install && npm run dev
```

1.2: Migrar la base de datos

Ejecuta las **migraciones** para crear las **tablas** necesarias en tu **base** de **datos**:

```
php artisan migrate
```



Si el archivo .en v esta bien configurado para conectarse a la base de datos veríamos que se crearon nuestras tablas

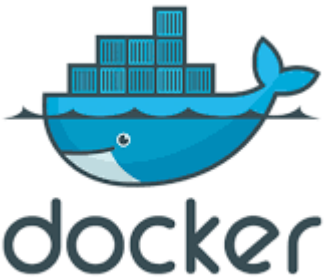
Posibles errores

Asegurate de configurar bien tu archivo .env como se menciona en el paso **1.3** Configura la **base** de **datos**: ya que esto servirá para controlar todas las inyecciones a la base de datos

En caso de querer conectarse mediante la consola puedes usar el siguiente **comando** usando las credenciales de acceso de tu archivo .env

```
mysql -h localhost -u root -proot ecommerce_caps
```

2: Ejecutar el servidor de desarrollo



En este paso, utilizaremos **Docker** para crear un **entorno** de **desarrollo** para tu proyecto **Laravel**. **Docker** te permitirá **encapsular** tu aplicación junto con todas sus dependencias en un **contenedor**, facilitando su despliegue y evitando problemas de configuración en diferentes entornos.

Si necesitas una guía más completa pera usar docker y Laravel consulta este video : Instalar y Ejecutar LARAVEL usando DOCKER MUY FÁCIL

2.1. Crear el archivo `docker-compose.yml`

El archivo `docker-compose.yml` define los servicios que conforman tu aplicación, como el servidor web y la base de datos.

```
version: '3.8'

services:
 web:
   build:
     context: .
     dockerfile: Dockerfile
   ports:
     - "8005:80"
   volumes:
     -
/home/maunick/Documentos/ecommerce_caps/public:/var/www/html/public
     -
/home/maunick/Documentos/ecommerce_caps/resources:/var/www/html/resourc
es
     -
/home/maunick/Documentos/ecommerce_caps/node_modules:/var/www/html/node
_modules
     - /home/maunick/Documentos/ecommerce_caps:/var/www/html
   networks:
     - app-network

 db:
   image: mysql:5.7
   environment:
     MYSQL_ROOT_PASSWORD: root
     MYSQL_DATABASE: ecommerce_caps
   ports:
     - "3306:3306"
   volumes:
     - db-data:/var/lib/mysql
   networks:
     - app-network
networks:
 app-network:
   driver: bridge

volumes:
 db-data:
```

2.2. Crear el archivo `Dockerfile`

```
# Usa una imagen base de PHP con Apache
FROM php:8.2-apache

# Instalar dependencias de PHP
RUN apt-get update && apt-get install -y \
    libpng-dev libjpeg-dev libfreetype6-dev \
    libzip-dev git unzip libpq-dev libxml2-dev curl \
    && docker-php-ext-install dom xml

# Instalar Node.js y npm
RUN curl -fsSL https://deb.nodesource.com/setup_16.x | bash - && \
    apt-get install -y nodejs

    # Instalar Composer
RUN curl -sS https://getcomposer.org/installer | php --
--install-dir=/usr/local/bin --filename=composer

# Configurar extensiones de PHP
RUN docker-php-ext-configure gd --with-freetype --with-jpeg && \
    docker-php-ext-install gd pdo pdo_mysql zip

# Habilitar mod_rewrite
RUN a2enmod rewrite

# Establecer el directorio de trabajo
WORKDIR /var/www/html

# Copiar el código fuente desde el contexto de construcción
COPY . /var/www/html

# Instalar dependencias de Node.js
RUN npm install

# Compilar assets usando Vite
RUN npm run build

# Exponer el puerto 80 (por defecto para Apache)
EXPOSE 80

# Comando por defecto
CMD ["apache2-foreground"]
```

2.3. Construir y ejecutar los contenedores de Docker

Ahora que tienes configurados el `docker-compose.yml` y el `Dockerfile`, puedes construir y ejecutar tus contenedores Docker:

2.3.1 Navega al directorio del proyecto en tu terminal donde se encuentran los archivos `docker-compose.yml` y `Dockerfile`.

2.3.2 Ejecuta el siguiente comando para construir y ejecutar los contenedores:

```
docker-compose up -d --build
```

2.3.3 Verifica que los **contenedores** estén en **ejecución** con el siguiente comando:

```
docker-compose ps
```

Deberías ver los servicios web y db en ejecución.

```
NAME                    IMAGE                COMMAND                  SERVICE
ecommerce_caps-db-1     mysql:5.7            "docker-entrypoint.s…"   db
ecommerce_caps-web-1    ecommerce_caps-web   "docker-php-entrypoi…"   web
```

3. Acceder a la aplicación Laravel

Una vez que los contenedores estén en ejecución, puedes acceder a tu aplicación Laravel en tu navegador web en la dirección:

http://localhost:8005/public/index.php

Comandos adicionales útiles

Reiniciar los contenedores:

```
docker-compose restart
```

Ver los **logs** de los contenedores:

```
docker-compose logs -f
```

Detener los contenedores:

```
docker-compose down
```