Guia de Instalacion

Posibles problemas a la hora de instalar el proyecto despues de haberlo clonado en git.

Para estas pruebas utilizaremos una maquina virtual con **Ubuntu 22.04 LTS**, en este entorno se encuentra instalado **XAMPP**. En el cual alojaremos nuestro proyecto.

Tambien debemos disponer de NodeJS, preferiblemente usando NVM para poder trabajar con la carpeta cliente. Asi como tambien debemos tener PHP y Laravel instalados para poder trabajar con el servidor.

A continuacion se detallan los pasos para instalar todo en un ordenador con nuestras caracteristicas.

ANGULAR

Para instalar nvm usamos:

Download and install NVM
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.3/install.sh | bash

Cerrar la terminal y al empezar una nueva podemos comprobar que funciona con:
nvm -v

Procedemos luego a instalar NodeJS, usando nvm basta escribir el siguiente comando:
nvm install –lts

Luego debemos instalar el packet manager de Node, esto se hace con:
sudo apt install npm

Ya con todo instalado podemos hacer uso de su manager para instalar Angular.

El comando es el siguiente:
sudo npm install -g @angular/cli

LARAVEL

Al igual que Angular, Laravel tambien requiere de Node, si no lo tienes instalado sigue los pasos anteriores.

Para instalar Laravel debemos de tener instalada la ultima version de PHP, para esto podemos usar los comandos:

sudo apt update

sudo apt install php php-cli php-mbstring php-xml php-curl php-zip php-bcmath php-json php-mysql

Tambien debemos instalar **composer**, esto lo logramos con:

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

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

Con todo instalado, hacemos un clone a nuestro repositorio:

Por defecto lo que veras en pantalla es la rama **main** del repositorio, si quieres empezar a desarrollar o realizar algun cambio debes tener tambien la rama develop, esta la puedes conseguir usando:

git fetch –all
git checkout develop

Esto te dara acceso a nuestra rama de desarrollo.

CLIENTE

Ya dentro del repositorio puedes navegar a la carpeta Client, alli debes tambien instalar las dependencias de Node, ya que estas al pesar mucho no vienen incluidas y estan puestas en el fichero .gitignore de nuestro repositorio. Esto se hace usando el comando:

npm install (IMPORTANTE, ejecutar el comando estando dentro de Client, si no, no funciona)

Podemos probar si esta todo operativo abriendo un servidor usando el comando:
ng serve -o

En nuestra maquina de pruebas dio un error y fue:

Node.js version v12.22.9 detected.
The Angular CLI requires a minimum Node.js version of v18.19.

Please update your Node.js version or visit https://nodejs.org/ for additional instructions.

Para corregirlo se puede instalar otra version de node usando nvm:

nvm install 18
nvm use 18
nvm -v (para comprobar)

EXITO! Si abrimos el servidor servido por node que por defecto es **http://localhost:4200/** podemos ver que nuestra app de Angular esta servida.

SERVIDOR

Ahora verificamos las funcionalidad del servidor, importante, antes de nada verificar que XAMPP este sirviendo la base de datos MySQL como Apache.

Ya con nuestro repositorio descargado navega a la carpeta app/Server, en esta debes descargar las dependencias usando:

composer install

En nuestro caso nos encontramos que nuestra version de PHP no es soportada por composer, para actualizarla debemos usar otro repositorio que contenga una version mas reciente:

sudo add-apt-repository ppa:ondrej/php
sudo apt update

Luego instalamos todas las dependencias de php:

sudo apt install php8.3

sudo apt install php8.3-cli php8.3-fpm php8.3-mysql php8.3-xml php8.3-mbstring php8.3-curl php8.3-zip php8.3-bcmath

sudo update-alternatives --set php /usr/bin/php8.3

Luego de seguir estos pasos, reiniciamos nuestro servidor.

En caso de no tenerlo, crear un documento .env estableciendo la conexión y el tipo de base de datos en este.

```
APP_NAME=Laravel
```

```
APP_ENV=local
APP_KEY=base64:WT5df+eLrwMnzarY267Eu2ryviKLV0aqwB98k+Rv0cA=
APP_DEBUG=true
APP_TIMEZONE=UTC
APP_URL=http://localhost

APP_LOCALE=en
APP_FALLBACK_LOCALE=en
APP_FAKER_LOCALE=en_US

APP_MAINTENANCE_DRIVER=file
# APP_MAINTENANCE_STORE=database

PHP_CLI_SERVER_WORKERS=4

BCRYPT_ROUNDS=12

LOG_CHANNEL=stack
LOG_STACK=single
LOG_DEPRECATIONS_CHANNEL=null
LOG_LEVEL=debug

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=server
DB_USERNAME=root
DB_PASSWORD=

SESSION_DRIVER=database
SESSION_LIFETIME=120
SESSION_ENCRYPT=false
SESSION_PATH=/
SESSION_DOMAIN=null

BROADCAST_CONNECTION=log
FILESYSTEM_DISK=local
QUEUE_CONNECTION=database

CACHE_STORE=database
CACHE_PREFIX=

MEMCACHED_HOST=127.0.0.1

REDIS_CLIENT=phpredis
REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379
```

```
MAIL_MAILER=log
MAIL_SCHEME=null
MAIL_HOST=127.0.0.1
MAIL_PORT=2525
MAIL_USERNAME=null
MAIL_PASSWORD=null
MAIL_FROM_ADDRESS="hello@example.com"
MAIL_FROM_NAME="${APP_NAME}"

AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_DEFAULT_REGION=us-east-1
AWS_BUCKET=
AWS_USE_PATH_STYLE_ENDPOINT=false

VITE_APP_NAME="${APP_NAME}"
```

Para poder hacer uso de la base de datos, recordar hacer un **php artisan migrate.**

Podemos comprobar que se ha creado la base de datos satisfactoriamente. Si a la hora de acceder a localhost nos diese algun problema, verificar los permisos a los ficheros de server.

sudo chmod -R 775 /opt/lampp/htdocs/VoluntApp/app/Server/storage
sudo chown -R daemon:daemon /opt/lampp/htdocs/VoluntApp/app/Server/storage

sudo chown -R daemon:daemon
/opt/lampp/htdocs/VoluntApp/app/Server/bootstrap/cache
sudo chmod -R 775 /opt/lampp/htdocs/VoluntApp/app/Server/bootstrap/cache