

A

AUNNA
innovation technologies

MANUAL DE IMPLANTACIÓN

Presentado a:



Murcia, a 15 de mayo de 2024

© AUNNA IT – Reservados todos los derechos.



www.aunnait.es



Título documento	Plan de implantación		
Destinado a:	Diputación de Alicante		
Preparado por:	AUNNA IT	Fecha de versión	15/05/2024
Revisado por:	AUNNA IT	Fecha de revisión	24/07/2024

Versión	Fecha versión	Autor	Descripción
1.0	15/05/2024	AUNNA IT	Versión inicial del documento.
2.0	28/05/2024	AUNNA IT	Se añaden dos nuevos comandos a la instalación de PHP
3.0	04/06/2024	AUNNA IT	Se añaden aclaraciones según la solicitud realizada por sistemas de Suma.
4.0	07/06/2024	AUNNA IT	Se añaden varias aclaraciones como el uso de root y bash. Se incluye un ejemplo completo de "mysql_secure_installation".
5.0.	28/06/2024	AUNNA IT	Se añade en apartado 5.1. descomprimir plugin-rasa.zip. Se añaden comandos "composer install", php artisan db:seeders" y "npm install" en el apartado 5.3.
6.0.	24/07/2024	AUNNA IT	Revisión general del documento aplicando las mejoras necesarias para la correcta instalación y configuración del proyecto.

Contenido

Contenido	3
1. INTRODUCCIÓN	4
2. REQUISITOS PREVIOS	4
2.1. Requisitos del sistema	4
2.2. Pre-instalaciones necesarias	8
2.3. Instalación inicial Rasa	10
4. CONFIGURACIÓN DEL ENTORNO	10
4.1. Composer	10
4.2. PHP	11
4.3. Node.js – NPM	12
4.4. MySQL	13
4.5. Instalación de Apache	15
5. CONFIGURACIÓN DEL BACKEND / FRONTEND	16
5.1. Descomprimir instalación de Laravel	16
5.2. Configuración del archivo .env	17
5.3. Migración de la base de datos	18
6. IMPLEMENTACIÓN RASA	19
6.1. Instalación y configuración de Rasa	19
6.2. Configuración de apache	20
6.3. Creación del archivo de configuración y su certificado	21
6.4. SUPERVISOR	25
6. Configuración del CRON JOB	26
7. Ejecutar el proyecto	26





1. INTRODUCCIÓN

Este documento detalla el plan de implementación del proyecto del Chatbot para la Diputación de Alicante. El Chatbot ha sido diseñado como una herramienta esencial para proporcionar un canal de comunicación ágil y disponible las 24 horas del día, los 7 días de la semana. Su función es responder consultas, guiar en trámites y ofrecer información relevante de manera automatizada.

El objetivo principal de su implementación es mejorar la experiencia del usuario al interactuar con los servicios públicos, reducir los costos operativos asociados a la atención al cliente y facilitar el acceso a la información y servicios públicos digitales de manera eficiente.

La tecnología subyacente para este proyecto incluye el uso de Rasa para el desarrollo del Chatbot, un backend/frontend en Laravel para la integración y gestión de la información, y diversas pre-instalaciones necesarias en el servidor para asegurar un entorno robusto y seguro. La combinación de estas tecnologías permitirá una implementación que se adapte a las necesidades específicas de la Diputación de Alicante.

Este documento está estructurado para proporcionar una guía completa y detallada sobre todos los aspectos relacionados con la implementación del Chatbot. Se describen los requisitos previos, la configuración del entorno, la instalación y configuración de Rasa, la implementación del backend y frontend en Laravel, y las pruebas y validación necesarias para garantizar un funcionamiento óptimo.

A lo largo de esta guía, se abordarán los pasos necesarios para llevar a cabo la implantación del Chatbot y se describirán las tareas clave involucradas.

2. REQUISITOS PREVIOS

En esta sección, detallaremos los pasos requeridos para la instalación de las diversas tecnologías empleadas, las cuales han sido fundamentales en el desarrollo del producto.

La arquitectura está preparada para dar soporte a un entorno multi-organismo con un único servidor, una única instancia del servidor de aplicaciones y una única instancia de base de datos.

2.1. Requisitos del sistema

Para la implementación del Chatbot de la Diputación de Alicante, se requiere un servidor con las siguientes especificaciones mínimas:

- **Sistema operativo.** Ubuntu 20.04 LTS o superior. Para este proyecto se ha utilizado la versión 22.04.3 LTS, la última versión LTS ("Long Term Support", Soporte a Largo Plazo), lanzada el 23 de abril de 2022. Al ser una versión LTS, recibirá actualizaciones de seguridad y soporte técnico durante un período de 5 años, hasta abril de 2027.

```
ubuntu@vps-c5d78b61:/proc$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:   Ubuntu 22.04.4 LTS
Release:       22.04
Codename:      jammy
```

- **Procesador.** CPU de al menos 2 núcleos. Importante que el procesador sea capaz de soportar instrucciones AVX. A continuación se muestra la información de uno de los procesadores utilizados:

```

ubuntu@vps-c5d78b61:~$ cat /proc/cpuinfo
processor       : 0
vendor_id      : GenuineIntel
cpu family     : 6
model          : 60
model name     : Intel Core Processor (Haswell, no TSX)
stepping       : 1
microcode      : 0x1
cpu MHz        : 2399.998
cache size     : 16384 KB
physical id    : 0
siblings       : 1
core id        : 0
cpu cores      : 1
apicid         : 0
initial apicid : 0
fpu            : yes
fpu_exception  : yes
cpuid level    : 13
wp             : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 syscall nx rdtscp lm constant_tsc rep_good nopl xtopolog
y cpuid tsc_known_freq pni pclmuldq vmx ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand hypervisor lahf_lm abm cpuid_fau
lt invpcid_single pti tpr_shadow vmmi flexpriority ept vpid ept_ad fsgsbase bmi1 avx2 smep bmi2 erms invpcid xsaveopt arat md_clear
vmx flags       : vmmi preemption_timer posted_intr invvpid ept_x_only ept_ad ept_lgb flexpriority apicv tsc_offset vtptr mtf vpic ept vpid unrestricted_guest vpic_reg vid
shadow_vmcs     : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swaps itlb_multihit srbds mmio_unknown bhi
bogomips       : 4799.99
clflush size    : 64
cache alignment : 64
address sizes   : 40 bits physical, 48 bits virtual
power management:

```

- **Memoria RAM.** Un mínimo de 4 GB de RAM, recomendándose 8 GB o más para un mejor rendimiento, considerando que cada instancia del Chatbot desarrollado con Rasa ocupa aproximadamente 1 GB de RAM. Cada idioma del Chatbot se corresponde con una de estas instancias, de modo que un Chatbot con los tres idiomas (castellano, inglés y valenciano) estaría consumiendo 3 instancias.
- **Espacio en Disco:** Al menos 20 GB de espacio libre en disco, preferiblemente en un SSD para mejorar la velocidad de acceso y procesamiento.

A continuación, incluimos más información sobre el servidor utilizado obtenida con el comando “dmidecode”:

```

# dmidecode 3.3
Getting SMBIOS data from sysfs.
SMBIOS 2.8 present.
11 structures occupying 533 bytes.
Table at 0x000F6820.

Handle 0x0000, DMI type 0, 24 bytes
BIOS Information
    Vendor: SeaBIOS
    Version: 2:1.10.2-58953eb7
    Release Date: 04/01/2014
    Address: 0xE8000
    Runtime Size: 96 kB
    ROM Size: 64 kB
    Characteristics:
        BIOS characteristics not supported
        Targeted content distribution is supported
    BIOS Revision: 0.0

```



Handle 0x0100, DMI type 1, 27 bytes

System Information

Manufacturer: OpenStack Foundation

Product Name: OpenStack Nova

Version: 19.3.2

Serial Number: 0d3ac571-fb13-4f09-b946-510bae600102

UUID: 0d3ac571-fb13-4f09-b946-510bae600102

Wake-up Type: Power Switch

SKU Number: Not Specified

Family: Virtual Machine

Handle 0x0300, DMI type 3, 22 bytes

Chassis Information

Manufacturer: QEMU

Type: Other

Lock: Not Present

Version: pc-i440fx-focal

Serial Number: Not Specified

Asset Tag: Not Specified

Boot-up State: Safe

Power Supply State: Safe

Thermal State: Safe

Security Status: Unknown

OEM Information: 0x00000000

Height: Unspecified

Number Of Power Cords: Unspecified

Contained Elements: 0

SKU Number: Not Specified

Handle 0x0400, DMI type 4, 42 bytes
Processor Information

Socket Designation: CPU 0
Type: Central Processor
Family: Other
Manufacturer: QEMU
ID: C1 06 03 00 FF FB 8B 07
Version: pc-i440fx-focal
Voltage: Unknown
External Clock: Unknown
Max Speed: 2000 MHz
Current Speed: 2000 MHz
Status: Populated, Enabled
Upgrade: Other
L1 Cache Handle: Not Provided
L2 Cache Handle: Not Provided
L3 Cache Handle: Not Provided
Serial Number: Not Specified
Asset Tag: Not Specified
Part Number: Not Specified
Core Count: 1
Core Enabled: 1
Thread Count: 1
Characteristics: None



```
Handle 0x1000, DMI type 16, 23 bytes
Physical Memory Array
    Location: Other
    Use: System Memory
    Error Correction Type: Multi-bit ECC
    Maximum Capacity: 4000 MB
    Error Information Handle: Not Provided
    Number Of Devices: 1

Handle 0x1100, DMI type 17, 40 bytes
Memory Device
    Array Handle: 0x1000
    Error Information Handle: Not Provided
    Total Width: Unknown
    Data Width: Unknown
    Size: 4000 MB
    Form Factor: DIMM
    Set: None
    Locator: DIMM 0
    Bank Locator: Not Specified
    Type: RAM
    Type Detail: Other
    Speed: Unknown
    Manufacturer: QEMU
    Serial Number: Not Specified
    Asset Tag: Not Specified
    Part Number: Not Specified
    Rank: Unknown
    Configured Memory Speed: Unknown
    Minimum Voltage: Unknown
    Maximum Voltage: Unknown
    Configured Voltage: Unknown
```

2.2. Pre-instalaciones necesarias

Antes de proceder con la instalación y configuración del Chatbot, se deben realizar las siguientes pre-instalaciones y configuraciones en el servidor Ubuntu.

- **Actualización del sistema.**

```
sudo apt update && sudo apt upgrade -y
```


** Aceptar los parámetros por defecto que puedan surgir.*

- **Instalación de Python.** Se requiere Python 3.8 o superior.

```
sudo apt install python3 python3-venv python3-pip -y
```

** Aceptar los parámetros por defecto que puedan surgir.*

- **Instalación de Dependencias Adicionales.** Instalación de dependencias necesarias para el funcionamiento de Rasa.

```
sudo apt install build-essential libssl-dev libffi-dev python3-dev -y
```

** Aceptar los parámetros por defecto que puedan surgir.*

- **Configuración del Firewall:** Asegurar que los puertos necesarios estén abiertos para la comunicación del Chatbot.

```
sudo ufw allow 5005 # Puerto por defecto de Rasa
sudo ufw allow 6001
sudo ufw allow 6002
sudo ufw allow http # HTTP
sudo ufw allow https # HTTPS
sudo ufw allow ssh # SSH
sudo ufw enable
sudo ufw status verbose #VERIFICAR EL ESTADO DE LA CONFIGURACIÓN
```

```
ubuntu@vps-c5d78b61:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
```

To	Action	From
--	-----	----
5005	ALLOW IN	Anywhere
6001	ALLOW IN	Anywhere
6002	ALLOW IN	Anywhere
80/tcp	ALLOW IN	Anywhere
443	ALLOW IN	Anywhere
22/tcp	ALLOW IN	Anywhere
5005 (v6)	ALLOW IN	Anywhere (v6)
6001 (v6)	ALLOW IN	Anywhere (v6)
6002 (v6)	ALLOW IN	Anywhere (v6)
80/tcp (v6)	ALLOW IN	Anywhere (v6)
443 (v6)	ALLOW IN	Anywhere (v6)
22/tcp (v6)	ALLOW IN	Anywhere (v6)

Estas pre-instalaciones y configuraciones aseguran que el entorno del servidor esté preparado para la

instalación de Rasa y el despliegue del Chatbot, garantizando un rendimiento óptimo.

2.3. Instalación inicial Rasa

Es importante asegurarse de que Python esté instalado previamente, tal y como se indicó en los pasos anteriores. La instalación de Rasa y su configuración implica la creación de un entorno virtual y la instalación de paquetes específicos, además de la configuración de permisos para el usuario `www-data`, que será el mismo usuario que la web y que interactuará con Rasa en la creación, eliminación y gestión de distintos elementos.

1. Crear directorio para Rasa:

```
sudo mkdir -p /opt/rasa
```

2. Crear un entorno virtual:

```
sudo chown -R $(whoami):$(whoami) /opt/rasa
```

```
python3 -m venv /opt/rasa
```

3. Instalar Rasa e idiomas:

Se ejecutan sin sudo

```
/opt/rasa/bin/pip install --upgrade pip
/opt/rasa/bin/pip install rasa rasa[spacy]
/opt/rasa/bin/python -m spacy download es_core_news_md
/opt/rasa/bin/python -m spacy download en_core_web_md
/opt/rasa/bin/python -m spacy download ca_core_news_md
```

4. CONFIGURACIÓN DEL ENTORNO

En este apartado se describen los pasos necesarios para instalar las diferentes tecnologías requeridas para el correcto funcionamiento del sistema de Chatbot, orientado al sistema operativo Ubuntu mencionado anteriormente, 22.04.3 LTS. A continuación, se detallan las tecnologías necesarias y el procedimiento para su instalación.

4.1. Composer

Composer es una herramienta de administración de dependencias para PHP. Permite declarar y administrar las bibliotecas y dependencias que necesita un proyecto. Composer simplifica el proceso de integración de bibliotecas externas en un proyecto PHP, facilitando así el desarrollo de aplicaciones complejas. Estas bibliotecas pueden especificarse en un archivo llamado "composer.json" ubicado en la raíz de su proyecto. Este archivo describe las dependencias del proyecto, incluyendo el nombre de la biblioteca, la versión requerida y la fuente del paquete.

Versión. La versión utilizada es la 2.2.6

Instalación.

1. El primer paso es actualizar la lista de paquetes disponibles desde los repositorios configurados en el sistema. La lista de repositorios debe estar configurada en `/etc/apt/sources.list`. Para ello, ejecutamos el siguiente comando:

```
sudo apt update
```

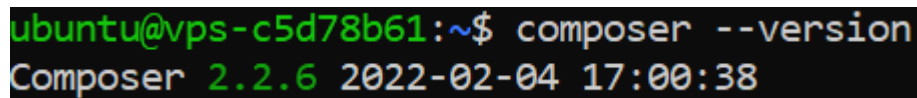
Este comando comprueba si hay nuevas versiones de los paquetes disponibles. No actualiza los paquetes en sí, solo actualiza la lista de paquetes disponibles en los repositorios. Después de ejecutar "apt update", se podría usar "apt upgrade" para actualizar los paquetes a las versiones más recientes disponibles. Es importante ejecutar "apt update" regularmente para asegurarse de que tu sistema esté al tanto de las últimas versiones de los paquetes disponibles.

2. El siguiente paso sería instalar Composer a partir del siguiente comando:

```
sudo apt install composer -y
```

** Aceptar los parámetros por defecto que puedan surgir.*

VERIFICAR INSTALACIÓN: `composer --version`



```
ubuntu@vps-c5d78b61:~$ composer --version
Composer 2.2.6 2022-02-04 17:00:38
```

***APT** (Advanced Package Tool) es el sistema de gestión de paquetes predeterminado en las distribuciones basadas en Debian, como Ubuntu. Se utiliza para instalar, actualizar y administrar paquetes de software en el sistema operativo. APT automatiza el proceso de descarga, instalación y configuración de software, facilitando así la gestión del sistema y la instalación de nuevas aplicaciones.

4.2. PHP

El sistema de Chatbot requiere PHP junto con una serie de extensiones para su funcionamiento adecuado.

Versión. La versión utilizada es la 8.2.

Instalación

1. Instalar el paquete base de PHP con la versión necesaria. Por defecto, Composer instala la versión 8.1.2. Para comprobar las versiones que se encuentran instaladas debemos ejecutar `apt list --installed | grep php`. Para conocer la versión que se está utilizando escribimos `php -v`. A continuación ejecutamos los siguientes comandos:

- `sudo add-apt-repository ppa:ondrej/php`
- `sudo apt update`
- `sudo apt install php8.2 -y` (con esta instrucción también queda instalado el servicio de apache)

** Aceptar los parámetros por defecto que puedan surgir.*

2. Instalación de las diferentes extensiones de PHP necesarias y utilizadas en este proyecto. (copiar comando completo)

- `sudo apt install php8.2-cli php8.2-dom php8.2-common php8.2-mysql php8.2-zip php8.2-gd php8.2-mbstring php8.2-curl php8.2-xml php8.2-bcmath libapache2-mod-php8.2 php8.2-fpm -y`

3. Indicamos al sistema operativo la versión por defecto de PHP a utilizar (8.2). Ejecutamos el siguiente comando y escribimos el número que corresponda a la versión 8.2.

```
sudo update-alternatives --config php
```

```
ubuntu@vps-c5d78b61:~$ sudo update-alternatives --config php
There are 3 choices for the alternative php (providing /usr/bin/php).

  Selection    Path                        Priority  Status
  -----
* 0            /usr/bin/php.default        100      auto mode
  1            /usr/bin/php.default        100      manual mode
  2            /usr/bin/php8.1             81       manual mode
  3            /usr/bin/php8.2             82       manual mode

Press <enter> to keep the current choice[*], or type selection number: 3
```

```
ubuntu@vps-c5d78b61:~$ php -v
PHP 8.2.21 (cli) (built: Jul  4 2024 16:26:28) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.2.21, Copyright (c) Zend Technologies
    with Zend OPcache v8.2.21, Copyright (c), by Zend Technologies
```

4.3. Node.js – NPM

Node.js y NPM, ambas tecnologías son utilizadas únicamente para la gestión y uso de librerías de JavaScript y Vue.js.

NVM (Node Version Manager) es una herramienta de línea de comandos que permite facilitar la instalación de Node.js, así como la conmutación entre diferentes versiones, lo que ayuda a evitar conflictos y problemas de compatibilidad. Además, NVM permite administrar las versiones de Node.js de forma aislada, lo que significa que cada proyecto puede tener su propia versión de Node.js sin afectar a otros proyectos en el mismo sistema.

Al instalar Node.js a través de NVM, también se instala automáticamente NPM (Node Package Manager). NPM es el administrador de paquetes de Node.js y se utiliza para instalar, actualizar y gestionar las dependencias de los proyectos de Node.js. Al instalar Node.js con NVM, obtenemos tanto Node.js como NPM.

Versión. La versión de Node.js utilizada es 18.20.4.

Instalación.

1. Descargamos el software NVM del repositorio correspondiente.

```
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash
```

** En el paso anterior se pierde la conexión, se debe volver a conectar con el servidor y ejecutar de nuevo el comando.*

```
source ~/.bashrc
```

```
sudo apt-get install -y nodejs=18.20.4-1nodesource1
```

2. Finalmente, NPM se instalará de forma automática junto con Node.js.

```
ubuntu@vps-c5d78b61:~$ node -v
v18.20.4
```

4.4. MySQL

MariaDB es el sistema de gestión de bases de datos relacional utilizado en este proyecto para almacenar los datos del Chatbot municipal.

Versión. La versión utilizada es la 15.1, distribución 10.6.16-MariaDB

Instalación. La instalación de MariaDB se realiza con el siguiente comando.

- `sudo apt install mariadb-server`
- `sudo mysql_secure_installation`

A continuación, se incluye un ejemplo completo del comando:

“mysql_secure_installation”.

```
root@server:~# mysql_secure_installation
```

```
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
```

```
SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!
```

```
In order to log into MariaDB to secure it, we'll need the current password for the root user. If you've just installed MariaDB, and haven't set the root password yet, you should just press enter here.
```

```
Enter current password for root (enter for none):
```

```
** Se debe introducir una contraseña **
```

```
OK, successfully used password, moving on...
```

```
Setting the root password or using the unix_socket ensures that nobody can log into the MariaDB root user without the proper authorisation.
```

```
You already have your root account protected, so you can safely answer 'n'.
```

```
Switch to unix_socket authentication [Y/n] n
```

```
... skipping.
```

```
You already have your root account protected, so you can safely answer 'n'.
```

```
Change the root password? [Y/n] y
```

```
New password: ** Nueva contraseña para root sólo en la base de datos, anotarla para después **
```

```
Re-enter new password: ** Repetir la nueva contraseña para root en la base de datos **
```

```
Password updated successfully!
```

```
Reloading privilege tables..
```

```
... Success!
```



By default, a MariaDB installation has an anonymous user, allowing anyone to log into MariaDB without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a production environment.

Remove anonymous users? [Y/n] **Y**

... Success!

Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] **Y**

... Success!

By default, MariaDB comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment.

Remove test database and access to it? [Y/n] **Y**

- Dropping test database...

... Success!

- Removing privileges on test database...

... Success!

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

Reload privilege tables now? [Y/n] **Y**

... Success!

Cleaning up...

All done! If you've completed all of the above steps, your MariaDB installation should now be secure.

Thanks for using MariaDB!

Iniciar el servicio MySQL

Nos aseguramos que el servicio MySQL está en funcionamiento:

- `sudo systemctl status mysql`

En caso de no estar activo:

- `sudo systemctl start mysql`

Activamos para que se inicie automáticamente durante el arranque de la máquina:

- `sudo systemctl enable mysql`

Acceder a MySQL

Inicia sesión en el cliente de MySQL con el usuario root:

- `mysql -u root -p`

Te pedirá la contraseña indicada durante el proceso de instalación.

Crear una base de datos

Dentro del cliente MySQL, crea una nueva base de datos:

- `CREATE DATABASE nombre_de_tu_base_de_datos;`

Salir del cliente MySQL

- `EXIT;`

4.5. Instalación de Apache

Apache es el servidor web que se utilizará para servir la aplicación Laravel. Laravel es el Framework de desarrollo web PHP utilizado en el desarrollo del proyecto. Es de código abierto. Ofrece una amplia gama de características y herramientas, incluyendo un sistema de enrutamiento intuitivo, una capa de abstracción de bases de datos y un motor de plantillas Blade. Además, Laravel proporciona un conjunto de herramientas integradas para la autenticación, el almacenamiento en caché, la gestión de sesiones, entre otras.

Con la instalación anterior de PHP apache queda instalado. En caso de no estar instalado, ejecutar el siguiente comando.

- `sudo apt install apache2`

Habilitar los siguientes módulos:

- `sudo a2enmod proxy`
- `sudo a2enmod proxy_fcgi`
- `sudo a2enmod proxy_html`
- `sudo a2enmod proxy_wstunnel`
- `sudo a2enmod ssl`
- `sudo a2enmod headers`

Reiniciamos servicio de apache:

- `sudo systemctl restart apache2`



5. CONFIGURACIÓN DEL BACKEND / FRONTEND

En este apartado, se describen los pasos necesarios para descomprimir, configurar e instalar Laravel, así como los comandos necesarios para preparar la base de datos y asegurar la ejecución de las colas. También se incluyen las instrucciones de configuración de Apache para que sirva la aplicación Laravel.

5.1. Descomprimir instalación de Laravel

La instalación de Laravel se pasará como un archivo adjunto en formato .zip. Se recomienda descomprimir este archivo en el directorio `/var/www/chatbot` como ejemplo:

- `sudo mkdir -p /var/www/chatbot`
- `cd /var/www/chatbot`

Enviamos desde nuestra máquina local al servidor el siguiente archivo:

- `scp tu_archivo.zip usuario@ip_servidor:/tmp`

Estando en el directorio `/var/www/chatbot` en el servidor ejecutamos estos comandos:

- `sudo mv /tmp/tu_archivo.zip .`
- `sudo unzip tu_archivo.zip -d .`
- Copiamos el contenido de la carpeta en `/var/www/chatbot`
`sudo rsync -a tu_archivo_descomprimido/ .`

```
ubuntu@vps-c5d78b61:/var/www/chatbot$ ls -la
total 668
drwxrwxr-x 12 root root  4096 Jul 24 09:41 .
drwxr-xr-x  4 root root  4096 Jul 24 09:36 ..
-rw-rw-r--  1 root root   283 Jul 12 19:34 .editorconfig
-rw-rw-r--  1 root root  1366 Jul 22 03:18 .env.example
-rw-rw-r--  1 root root   186 Jul 12 19:34 .gitattributes
-rw-rw-r--  1 root root   243 Jul 12 19:34 .gitignore
-rw-rw-r--  1 root root   761 Jul 12 19:34 .gitlab-ci.yml
-rw-rw-r--  1 root root  4106 Jul 12 19:34 README.md
drwxrwxr-x 10 root root  4096 Jul 12 19:34 app
-rwxr-xr-x  1 root root  1686 Jul 12 19:34 artisan
drwxrwxr-x  3 root root  4096 Jul 12 19:34 bootstrap
-rw-rw-r--  1 root root  2392 Jul 18 16:20 composer.json
-rw-rw-r--  1 root root 420898 Jul 18 16:14 composer.lock
drwxrwxr-x  2 root root  4096 Jul 12 19:34 config
drwxrwxr-x  5 root root  4096 Jul 12 19:34 database
-rw-rw-r--  1 root root 155842 Jul 12 19:34 package-lock.json
-rw-rw-r--  1 root root  1478 Jul 12 19:34 package.json
-rw-rw-r--  1 root root   1134 Jul 12 19:34 phpunit.xml
drwxrwxr-x  8 root root  4096 Jul 12 19:34 public
-rw-rw-r--  1 root root    17 Jul 12 19:34 requirements.txt
drwxrwxr-x  9 root root  4096 Jul 12 19:34 resources
drwxrwxr-x  2 root root  4096 Jul 12 19:34 routes
drwxrwxr-x  6 root root  4096 Jul 12 19:34 storage
drwxrwxr-x  4 root root  4096 Jul 12 19:34 tests
drwxrwxr-x  3 root root  4096 Jul 18 16:19 vendor
-rw-rw-r--  1 root root   929 Jul 12 19:34 vite.config.js
```

- `sudo rm -fr tu_archivo_descomprimido`
- `sudo rm tu_archivo.zip`

- `sudo chown -R www-data:www-data /var/www/chatbot`
- `sudo chmod -R 775 /var/www/chatbot/`

5.2. Configuración del archivo .env

El archivo .env se incluye en la instalación de Laravel y contiene varias configuraciones esenciales para el funcionamiento de la aplicación.

Parámetro RASA_URL. Este parámetro indica la ubicación del servidor donde se encuentra Rasa. Generalmente, Rasa está situado en el mismo servidor que la aplicación Laravel. Este parámetro se usa de forma interna por la aplicación para comunicarse con Rasa. Dado que el acceso se realiza a nivel local por parte del servidor, el valor puede ser `http://localhost`. Esto significa que Rasa y Laravel interactúan entre sí a través de la red local del servidor, sin necesidad de acceso externo desde el cliente o navegador del cliente.

Configuración del archivo .env para la base de datos y Rasa:

- `sudo cp .env.example .env`
- `sudo nano .env`

Así debe quedar tu archivo .env, recuerda cambiar los datos en **negrita** en base a los tuyos:

```
APP_NAME=nombre_descriptivo_proyecto
APP_ENV=local
APP_KEY= (Va en blanco y se genera automáticamente cuando se ejecuta
el comando "php artisan key:generate" ver punto 5.3)
APP_DEBUG=true
APP_URL=http://simalicante1987.aunnaitapp.es (Reemplazar url por la
que corresponda)

LOG_CHANNEL=daily
LOG_DEPRECATIONS_CHANNEL=null
LOG_LEVEL=debug

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=nombre_base_de_datos
DB_USERNAME=usuario_base_de_datos
DB_PASSWORD="contraseña_base_de_datos"
RASA_URL=http://localhost

BROADCAST_DRIVER=pusher
CACHE_DRIVER=file
FILESYSTEM_DISK=local
QUEUE_CONNECTION=database
SESSION_DRIVER=file
SESSION_LIFETIME=3600

MEMCACHED_HOST=127.0.0.1
```

```

REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379

MAIL_MAILER=smtp
MAIL_HOST=mail
MAIL_PORT=1025
MAIL_USERNAME=null
MAIL_PASSWORD=null
MAIL_ENCRYPTION=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

PUSHER_APP_ID=5645623463
PUSHER_APP_KEY=79k6h4r190jky67
PUSHER_APP_SECRET=2kl4g59d4r3p45623
PUSHER_HOST=
PUSHER_PORT=443
PUSHER_SCHEME=https
PUSHER_APP_CLUSTER=mt1

WEBSOCKET_HOST=simalicante1987.aunnaitapp.es (reemplazar por el que
corresponda)
WEBSOCKET_SCHEME=http
WEBSOCKET_PORT=6001

VITE_APP_NAME="${APP_NAME}"
VITE_PUSHER_APP_KEY="${PUSHER_APP_KEY}"
VITE_PUSHER_HOST="${PUSHER_HOST}"
VITE_PUSHER_PORT="${PUSHER_PORT}"
VITE_PUSHER_SCHEME="${PUSHER_SCHEME}"
VITE_PUSHER_APP_CLUSTER="${PUSHER_APP_CLUSTER}"

VITE_WEBSOCKET_HOST="${WEBSOCKET_HOST}"
VITE_WEBSOCKET_SCHEME="${WEBSOCKET_SCHEME}"
VITE_WEBSOCKET_PORT=6002
VITE_FORCE_TLS=true

```

5.3. Migración de la base de datos

Se debe acceder al `php.ini`, reemplaza la versión de `php` por la que tengas instalada

- `sudo nano /etc/php/8.2/cli/php.ini`

Debemos asegurar que la línea que carga la extensión `intl` esté descomentada (sin punto y coma al inicio): Buscar `extension=intl`, descomentarlo, guardar y salir (**CTRL +O**, **ENTER** y **CTRL + X**).

- `sudo systemctl restart apache2`
- `sudo apt install php8.2-intl`
- `sudo systemctl restart apache2`

Por último ejecutamos los siguientes comandos:

- `sudo composer update`
- `sudo npm install`
- `sudo php artisan key:generate`
- `sudo php artisan migrate`
- `sudo php artisan db:seed`

6. IMPLEMENTACIÓN RASA

En este apartado, se describen los pasos de la estructura de archivos Rasa requerida para el proyecto.

Cuando se hace referencia a la carpeta `scripts/...`, se está hablando de la carpeta que contiene el plugin de Rasa en Laravel, ubicada en el directorio donde esté instalado Laravel, específicamente en `laravel_root/vendor/aunnait/rasa-alicante/scripts/...`. Este paso se debe ejecutar una vez disponemos de la instalación de Laravel, en el punto 5.

6.1. Instalación y configuración de Rasa

Los pasos para la instalación y configuración de rasa en la raíz del servidor fuer a de nuestro proyecto con lo siguiente.

1. Instalar servidor de acciones:

- Crear directorio para las acciones:

```
mkdir -p /opt/rasa/action
```

- Inicializar Rasa en el directorio de acciones:

```
/opt/rasa/bin/python -m rasa init --no-prompt --init-dir
/opt/rasa/action
```

- Copiar los archivos de configuración y scripts necesarios:

```
cp /var/www/chatbot/vendor/aunnait/rasa-
alicante/scripts/templates/actions.py
/opt/rasa/action/actions
```

```
cp /var/www/chatbot/vendor/aunnait/rasa-
alicante/scripts/templates/run-action.sh /opt/rasa/action
```

```
sudo cp /var/www/chatbot/vendor/aunnait/rasa-
alicante/scripts/templates/rasa-action.service
/etc/systemd/system
```

2. Copiar archivos de configuración adicionales:

```
sudo cp /var/www/chatbot/vendor/aunnait/rasa-
alicante/scripts/templates/sudo.rasa /etc/sudoers.d/rasa

sudo cp /var/www/chatbot/vendor/aunnait/rasa-
alicante/scripts/templates/rasa@.service /etc/systemd/system
```

Le damos permisos al usuario www-data:

```
sudo chown www-data:www-data -R /opt/rasa

sudo chmod -R 775 /opt/rasa
```

Recargar los demonios del sistema y habilitar los servicios:

```
sudo systemctl daemon-reload

sudo systemctl enable rasa-action

sudo systemctl start rasa-action
```

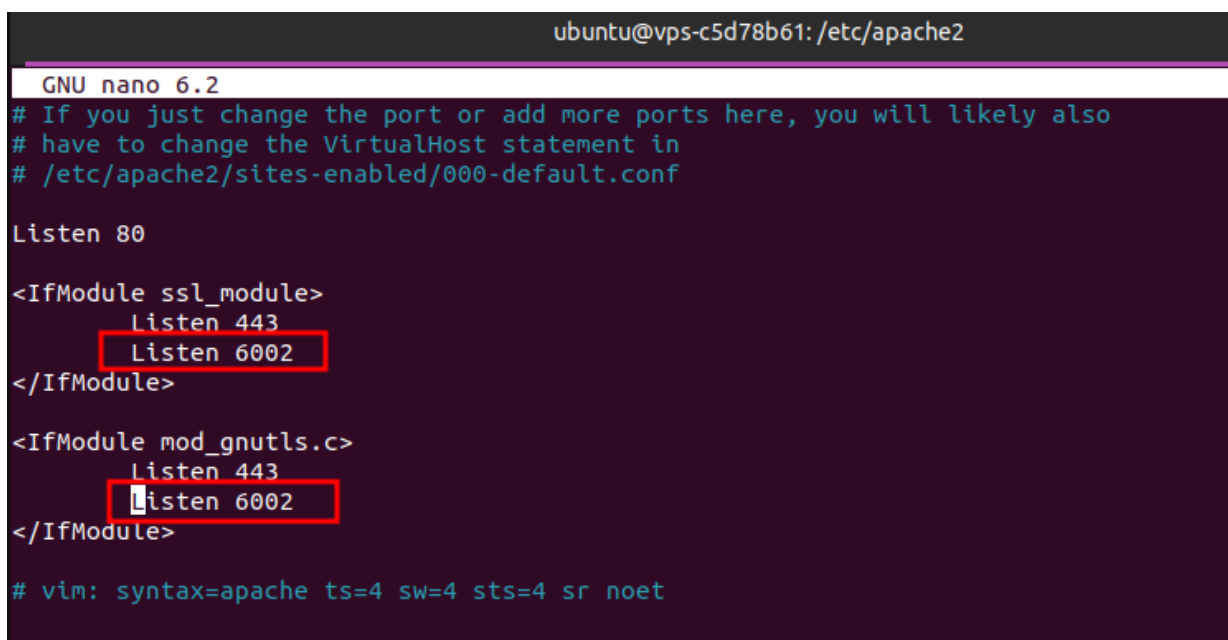
Estos pasos aseguran que Rasa esté correctamente instalado y configurado en el servidor, con los permisos adecuados para el usuario que interactuará con el sistema. Además, se configuran los servicios necesarios para la ejecución de las acciones del Chatbot, asegurando un entorno operativo robusto y funcional.

6.2. Configuración de apache

Se debe modificar el archivo ports.conf que se encuentra en el directorio **/etc/apache2** que describe los puertos en los que apache va a escuchar las conexiones. En este caso es importante puesto que se está escuchando en un puerto no estándar como es el 6002 para los websockets. Para acceder al archivo ejecuta el siguiente comando:

- `sudo nano /etc/apache2/ports.conf`

El archivo ports.conf debe quedar de la siguiente manera (**ver imagen**).



```
ubuntu@vps-c5d78b61: /etc/apache2
GNU nano 6.2
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80

<IfModule ssl_module>
    Listen 443
    Listen 6002
</IfModule>

<IfModule mod_gnutls.c>
    Listen 443
    Listen 6002
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

6.3. Creación del archivo de configuración y su certificado

El proyecto requiere la instalación de un archivo de configuración y su certificado correspondiente. Para ello, debemos editar la configuración de Apache.

Habilitamos los módulos necesarios.

- `sudo a2enmod rewrite`

Reiniciar apache para aplicar cambios.

- `sudo systemctl restart apache2`

Se debe crear un nuevo archivo de configuración (por ejemplo, `mi-sitio-puerto-XXXX.conf`) en el directorio `/etc/apache2/sites-available/` de la siguiente manera:

- `cd /etc/apache2/sites-available/` (entrar al directorio)
- `sudo nano nombre-sitio-web.conf` (crear el archivo)

Esta es la estructura que debería tener el archivo que se creó:

```
<VirtualHost *:80>
    ServerName    simalicante1987.aunnaitapp.es    (cambiar por el dominio
correspondiente)
    ServerAdmin   webmaster@localhost
    DocumentRoot  /var/www/chatbot/public
    ErrorLog      ${APACHE_LOG_DIR}/error.log
    CustomLog     ${APACHE_LOG_DIR}/access.log combined

    RewriteEngine on
    RewriteCond   %{SERVER_NAME} =simalicante1987.aunnaitapp.es    (cambiar por el
dominio correspondiente)
    RewriteRule   ^ https://%{SERVER_NAME}%{REQUEST_URI} [END,NE,R=permanent]
</VirtualHost>
```

Deshabilitar el sitio predeterminado de Apache:

- `sudo a2dissite 000-default.conf`

Este comando deshabilita el sitio predeterminado de Apache, que se configura en el archivo `000-default.conf`. Al deshabilitarlo, Apache dejará de servir el sitio configurado en este archivo.

Habilitar un nuevo sitio:

- `sudo a2ensite nombre-sitio-web.conf`

Una vez vemos que tenemos bien configurado el archivo `.conf` de nuestro proyecto:

- `sudo apache2ctl configtest`

Deberíamos obtener `Syntax OK` como respuesta, para verificar que la configuración del archivo `nombre-sitio-web.conf`, es correcta.

Recargamos servicio de apache:

- `sudo systemctl reload apache2`

A continuación, lanzamos la instalación de nuestro certificado a través de una entidad certificadora, para este documento usaremos la entidad certificadora de Let's Encrypt siguiendo estos pasos:

- `sudo apt install certbot python3-certbot-apache -y`

Ejecutamos el siguiente comando para la certificación del dominio:

- `sudo certbot --apache`

Output

```
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator apache, Installer apache
Enter email address (used for urgent renewal and security notices) (Enter 'c' to
cancel): you@your_domain
```

```
-----
Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.2-November-15-2017.pdf. You must
agree in order to register with the ACME server at
https://acme-v02.api.letsencrypt.org/directory
-----
(A)gree/(C)ancel: A
```

```
-----
Would you be willing to share your email address with the Electronic Frontier
Foundation, a founding partner of the Let's Encrypt project and the non-profit
organization that develops Certbot? We'd like to send you email about our work
encrypting the web, EFF news, campaigns, and ways to support digital freedom.
-----
(Y)es/(N)o: N
```

```
Which names would you like to activate HTTPS for?
-----
```

```
1: your_domain
2: www.your_domain
-----
```

```
Select the appropriate numbers separated by commas and/or spaces, or leave input
blank to select all options shown (Enter 'c' to cancel):
```

```

Obtaining a new certificate
Performing the following challenges:
http-01 challenge for your_domain
http-01 challenge for www.your_domain
Enabled Apache rewrite module
Waiting for verification...
Cleaning up challenges
Created an SSL vhost at /etc/apache2/sites-available/ your_domain -le-ssl.conf
Enabled Apache socache_shmcb module
Enabled Apache ssl module
Deploying Certificate to VirtualHost /etc/apache2/sites-available/ your_domain -le-ssl.conf
Enabling available site: /etc/apache2/sites-available/ your_domain -le-ssl.conf
Deploying Certificate to VirtualHost /etc/apache2/sites-available/ your_domain -le-ssl.conf

```

Please choose whether or not to redirect HTTP traffic to HTTPS, removing HTTP access.

-
- 1: No redirect - Make no further changes to the webserver configuration.
 - 2: Redirect - Make all requests redirect to secure HTTPS access. Choose this for new sites, or if you're confident your site works on HTTPS. You can undo this change by editing your web server's configuration.
-

Select the appropriate number [1-2] then [enter] (press 'c' to cancel): 2

Seleccione 2 para habilitar el redireccionamiento

Verifica que el archivo **nombre-sitio-web-le-ssl.conf** tenga una estructura similar. Es posible que falte la sección "Directory":

```

<IfModule mod_ssl.c>
<VirtualHost *:443>
    ServerName simalicante1987.aunnaitapp.es (cambiar por el dominio correspondiente)
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/chatbot/public
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
    <Directory /var/www/>
        Header set Access-Control-Allow-Origin "*"
        Header set Access-Control-Allow-Headers "*"
        Options Indexes FollowSymLinks MultiViews
        AllowOverride All
        Order allow,deny
        allow from all
        Require all granted
    </Directory>
    RewriteEngine on
    # Some rewrite rules in this file were disabled on your HTTPS site,
    # because they have the potential to create redirection loops.
    # RewriteCond %{SERVER_NAME} =simalicante1987.aunnaitapp.es (cambiar por el dominio correspondiente)
    # RewriteRule ^ https://%{SERVER_NAME}%{REQUEST_URI} [END,NE,R=permanent]

```

```

SSLCertificateFile
/etc/letsencrypt/live/simalicante1987.aunnaitapp.es/fullchain.pem (cambiar
por el dominio correspondiente)
SSLCertificateKeyFile
/etc/letsencrypt/live/simalicante1987.aunnaitapp.es/privkey.pem (cambiar por
el dominio correspondiente)
Include /etc/letsencrypt/options-ssl-apache.conf
</VirtualHost>
</IfModule>

```

En este mismo directorio (**/etc/apache2/sites-available**) se debe crear el archivo **ws-proxy.conf** cuyo contenido de ejemplo ha sido proporcionado con esta guía. Debe ser configurado de la siguiente manera:

- `sudo nano ws-proxy.conf`
- Escuchando en el puerto 6002 es el puerto seguro de conexión de websockets para Laravel. Aquí deberán igualmente cambiar algunos campos como:
- Servername al dominio que le corresponda.
- Todos los campos relacionados con los certificados.

Debería quedar así:

```

<IfModule mod_ssl.c>
<VirtualHost *:6002>
    ServerName simalicante1987.aunnaitapp.es (cambiar por el dominio
correspondiente)

    SSLEngine on
    SSLProxyEngine on
    #SSLProxyVerify none
    #SSLProxyCheckPeerName off
    #SSLVerifyClient require
    # Configuración de Proxy para WebSockets
    ProxyPass / ws://127.0.0.1:6001/
    ProxyPassReverse / ws://127.0.0.1:6001/
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
    SSLCertificateFile
/etc/letsencrypt/live/simalicante1987.aunnaitapp.es/fullchain.pem
(cambiar por el dominio correspondiente)
    SSLCertificateKeyFile
/etc/letsencrypt/live/simalicante1987.aunnaitapp.es/privkey.pem
(cambiar por el dominio correspondiente)
    Include /etc/letsencrypt/options-ssl-apache.conf
</VirtualHost>
</IfModule>

```



```

GNU nano 6.2 ws-proxy.conf
<IfModule mod_ssl.c>
<VirtualHost *:6002>
    ServerName simalicante1976.aunnaitapp.es
    SSLEngine on
    SSLProxyEngine on
    #SSLProxyVerify none
    #SSLProxyCheckPeerName off
    #SSLVerifyClient require
    # Configuración de Proxy para WebSockets
    ProxyPass / ws://127.0.0.1:6001/
    ProxyPassReverse / ws://127.0.0.1:6001/
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
    SSLCertificateFile /etc/letsencrypt/live/simalicante1976.aunnaitapp.es/fullchain.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/simalicante1976.aunnaitapp.es/privkey.pem
    SSLCertificateChainFile /etc/letsencrypt/live/simalicante1976.aunnaitapp.es/chain.pem
    Include /etc/letsencrypt/options-ssl-apache.conf
</VirtualHost>
</IfModule>

```

Habilitar un nuevo sitio:

- `sudo a2ensite ws-proxy.conf`

Reiniciar Apache:

- `sudo systemctl reload apache2`

Estos pasos aseguran que Laravel esté correctamente instalado, configurado y que su aplicación esté lista para funcionar, sirviendo contenido desde el directorio público especificado y con la configuración de base de datos adecuada. Además, se garantiza que las colas de trabajo de Laravel se ejecuten correctamente.

6.4. SUPERVISOR

Este software es necesario para el uso de las colas y los websockets en Laravel.

```
sudo apt install supervisor
```

Una vez instalado hay que crear los archivos en la carpeta **conf.d** del directorio **/etc/supervisor/conf.d**. Principalmente se deben tener en cuenta todos los campos que hacen referencia a la ruta donde se encuentra la aplicación web. En este caso en concreto han de modificar la directiva ***directory*** únicamente.

Se crea el archivo **websockets.conf** de la siguiente manera:

- `sudo nano /etc/supervisor/conf.d/websockets.conf`

Y agregamos las siguientes líneas:

```
[program:websocket]
command=php artisan websocket:serve --port=6001
directory=/var/www/chatbot
autostart=true
autorestart=true
redirect_stderr=true
stdout_logfile=/var/log/apache2/websocket.log

[program:queues]
;process_name=%(program_name)s_%(process_num)02d
;process_name=%(program_name)s_%(process_num)02d
command=php artisan queue:work --sleep=1 --max-jobs=3000 --tries=3
;numprocs=1
directory=/var/www/chatbot
autostart=true
autorestart=true
redirect_stderr=true
stdout_logfile=/var/log/apache2/queues.log
```

Luego ejecutamos el siguiente comando para reiniciar **supervisor**:

- `sudo supervisorctl reread`
- `sudo supervisorctl update`
- `sudo systemctl restart supervisor`

Verifica el estado de supervisor

- `systemctl status supervisor`

6. Configuración del CRON JOB

Ingresamos a **/etc/cron.d**, creamos el archivo laravel de la siguiente manera:

- `cd /etc/cron.d`
- `sudo nano laravel`

y agregamos la siguiente línea:

```
* * * * * www-data    cd /var/www/chatbot && php artisan schedule:run >> /dev/null 2>&1
```

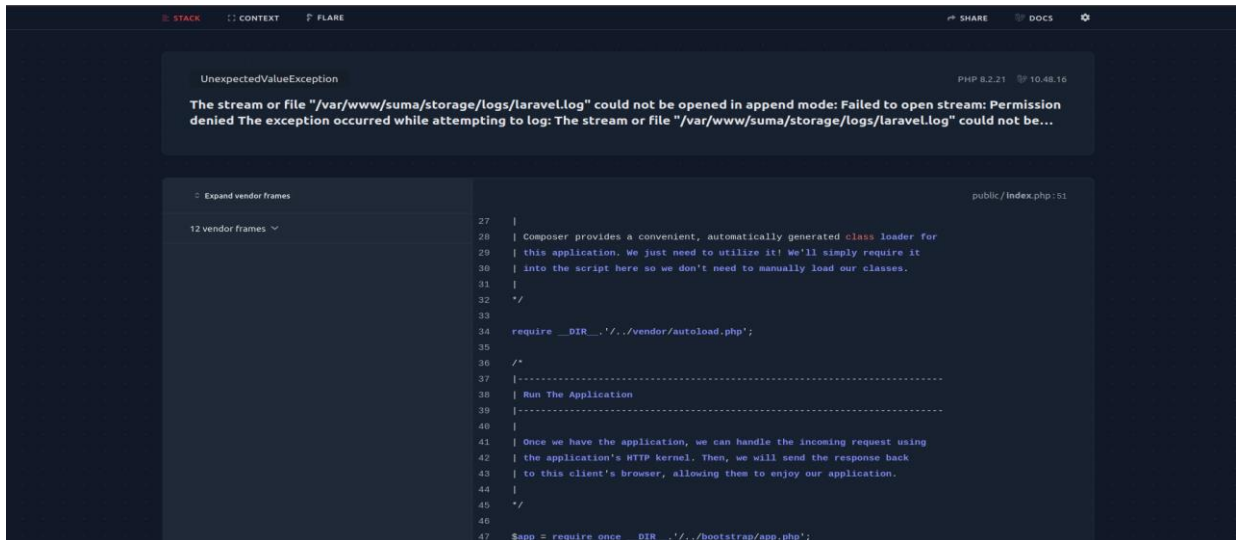
7. Ejecutar el proyecto

Ejecutar en la raíz del proyecto **/var/www/chatbot**:

- `cd /var/www/chatbot`
- `sudo composer dump-autoload`
- `sudo php artisan optimize`
- `sudo php artisan optimize:clear`
- `sudo npm run build`

- `sudo chown -R www-data:www-data /var/www/chatbot/`
- `sudo chmod -R 775 /var/www/chatbot/`

Abrir la aplicación en un navegador web, si se percibe esto:



Abrimos nuevamente el proyecto en el navegador. Debería salir la pantalla de login, con esto damos por finalizado el despliegue del proyecto.

En caso de realizar cambios sobre el proyecto se deben ejecutar los siguientes comandos:

```
sudo npm install
sudo npm run build
sudo rm -rf node_modules/* .git Dockerfile docker-compose.yml 000-
default.conf
sudo composer update
sudo composer install
sudo php artisan key:generate
sudo php artisan lang:publish
sudo php artisan cache:clear
sudo php artisan optimize
sudo php artisan storage:link
sudo supervisorctl restart websocket
sudo supervisorctl restart queues
```