



Gateway Webchat

Procedimiento de instalación y validación del software base

Documento	Procedimiento técnico de instalación
Versión	0.01
Fecha	1 de agosto de 2026
Estado	Instalación ejecutada y validada
Plataforma	Laravel 13 + Workerman + cliente MQTT asíncrono
Directorio	/var/www/html/gateway.axynia.smart-admin.biz
Dominio	gateway.axynia.smart-admin.biz

Resultado: El servidor quedó preparado para alojar el futuro Gateway WebSocket en PHP/Laravel. El proceso de red será persistente, ejecutado por Supervisor y basado en Workerman. La implementación funcional del Gateway, su endpoint, la conexión MQTT productiva y la configuración final de Supervisor quedan fuera del alcance de este documento.

Documento integrante del proyecto del Gateway. Ubicación prevista:

docs/setup/Axynia_Gateway_Setup_Plataforma_v0.01.pdf

1. Objetivo y alcance

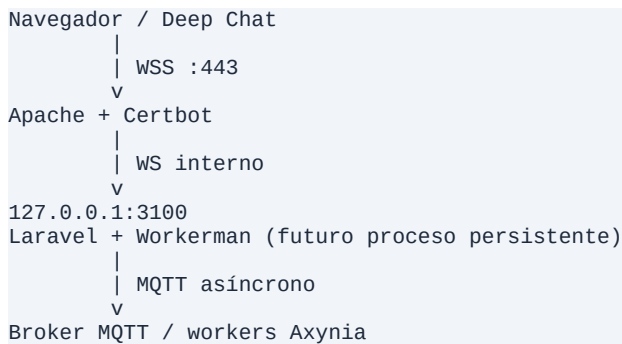
Este documento registra el procedimiento ejecutado para preparar la plataforma base del Gateway Webchat de Axynia sobre PHP/Laravel. La instalación se realizó directamente en el directorio definitivo del proyecto, luego de eliminar la implementación anterior y recrear el directorio de trabajo.

- Verificación del sistema operativo, PHP CLI, Composer, Git, Supervisor y extensiones requeridas.
- Creación de un proyecto Laravel 13 en la ruta definitiva.
- Instalación de Workerman y del cliente MQTT asíncrono workerman/mqtt.
- Configuración inicial de entorno, directorios documentales y permisos.
- Validación como root y como el usuario de servicio axynia-gateway.
- Corrección de soporte SQLite, protección de .env y controles finales de plataforma.

1.1 Fuera de alcance

- Implementación del servidor WebSocket y del protocolo de mensajes.
- Creación del endpoint persistente gateway.php o comando Artisan definitivo.
- Configuración del programa Supervisor que ejecutará el Gateway.
- Conexión real al broker MQTT, ACL, topics, QoS y credenciales.
- Cambios en Apache, TLS o reverse proxy, ya disponibles en el servidor.
- Pruebas funcionales del Gateway contra Deep Chat.

2. Arquitectura de plataforma resultante



Laravel aporta la configuración, el contenedor de servicios, logging, pruebas y organización del proyecto. Workerman será el motor de red que mantendrá el listener WebSocket y el event loop. El paquete workerman/mqtt permitirá incorporar la publicación y suscripción MQTT dentro del mismo flujo asíncrono, sin crear un loop bloqueante separado.

3. Plataforma validada

Componente	Versión / estado final	Uso
Sistema operativo	Debian GNU/Linux 13.6 (trixie)	Servidor base
PHP CLI	8.4.24 NTS	Runtime persistente del Gateway
Composer	2.8.8	Gestión de dependencias PHP
Git	2.47.3	Control de versiones
Laravel Framework	13.23.0	Framework de aplicación
Workerman	5.2.2	WebSocket y event loop
workerman/mqtt	2.2	Cliente MQTT asíncrono
Supervisor	Activo	Administración futura del proceso

Extensión event	No instalada	Opcional; se evaluará con carga
-----------------	--------------	---------------------------------

Aclaración de versión PHP: La verificación inicial mostró PHP 8.4.23. Durante la instalación de paquetes del sistema el patch release quedó actualizado; la versión final validada y que debe documentarse es PHP 8.4.24.

4. Preparación del servidor

4.1 Verificación inicial

```
cat /etc/os-release
php -v
php --ini
composer --version
git --version
systemctl is-active supervisor
supervisorctl status
ss -ltnp | grep ':3100 ' || true
df -h /var/www/html
```

Se verificó Debian 13.6, PHP CLI disponible, Composer y Git instalados, Supervisor activo, espacio libre suficiente y disponibilidad del puerto 3100 para la futura implementación PHP.

4.2 Extensiones PHP requeridas

```
php -r '
$required = [
    "ctype", "curl", "dom", "fileinfo", "filter", "hash",
    "mbstring", "openssl", "pcre", "pdo", "session",
    "tokenizer", "xml", "pcntl", "posix", "sockets"
];
foreach ($required as $extension) {
    printf("%-12s %s\n", $extension,
        extension_loaded($extension) ? "OK" : "FALTA");
}
'
```

Todas las extensiones requeridas quedaron disponibles. En particular, pcntl, posix y sockets son necesarias para el modelo de proceso persistente de Workerman en Linux.

```
php -r 'echo extension_loaded("event")
? "event: INSTALADA\n"
: "event: NO INSTALADA\n";'
```

La extensión event no fue instalada. Es una optimización opcional y no constituye un requisito para la primera etapa.

4.3 Usuario de servicio

```
id axynia-gateway
```

Se validó el usuario de sistema existente:

```
uid=988(axynia-gateway) gid=988(axynia-gateway) groups=988(axynia-gateway)
```

5. Creación del proyecto Laravel

5.1 Recreación del directorio definitivo

Precaución: Este procedimiento supone que la implementación anterior ya fue respaldada o descartada. El comando `rm -rf` elimina completamente el contenido de la ruta indicada.

```
cd /var/www/html
rm -rf gateway.axynia.smart-admin.biz/

BUILD_DIR="/var/www/html/gateway.axynia.smart-admin.biz"
```

5.2 Creación de Laravel 13

```
COMPOSER_ALLOW_SUPERUSER=1 composer create-project laravel/laravel:^13.0 "$BUILD_DIR" --
prefer-dist --no-interaction
```

La instalación creó Laravel 13 y generó automáticamente el archivo .env y la clave de aplicación. La versión final del framework validada fue 13.23.0.

5.3 Advertencia inicial de SQLite

Durante el post-install de Laravel se intentó ejecutar la migración inicial usando SQLite. El primer intento informó "could not find driver". La instalación del framework continuó correctamente. Más adelante se instaló php8.4-sqlite3 y se ejecutó la migración con éxito.

6. Dependencias del Gateway

6.1 Instalación de Workerman y MQTT

```
cd /var/www/html/gateway.axynia.smart-admin.biz

COMPOSER_ALLOW_SUPERUSER=1 composer require workerman/workerman workerman/mqtt --with-all-
dependencies --no-interaction
```

6.2 Versiones instaladas

```
COMPOSER_ALLOW_SUPERUSER=1 composer show workerman/workerman
COMPOSER_ALLOW_SUPERUSER=1 composer show workerman/mqtt
```

Paquete	Versión validada
workerman/workerman	v5.2.2
workerman/mqtt	v2.2
workerman/coroutine	v1.1.6, dependencia instalada

6.3 Validación del autoload

```
php -r '
require __DIR__ . "/vendor/autoload.php";
$classes = [
    Workerman\Worker::class,
    Workerman\Connection\TcpConnection::class,
    Workerman\Mqtt\Client::class,
];
foreach ($classes as $class) {
    printf("%-45s %s\n", $class,
        class_exists($class) ? "OK" : "FALTA");
}
'
```

```
Workerman\Worker                OK
Workerman\Connection\TcpConnection  OK
Workerman\Mqtt\Client           OK
```

7. Configuración inicial del proyecto

7.1 Variables generales de Laravel

```

APP_NAME="Axynia Gateway"
APP_ENV=production
APP_DEBUG=false
APP_URL=https://gateway.axynia.smart-admin.biz

LOG_CHANNEL=daily
LOG_LEVEL=info
CACHE_STORE=file
SESSION_DRIVER=file
QUEUE_CONNECTION=sync

```

7.2 Variables propias del Gateway

```

GATEWAY_ID=gateway-01
GATEWAY_HOST=127.0.0.1
GATEWAY_PORT=3100
GATEWAY_PATH=/ws
GATEWAY_WORKER_COUNT=1
GATEWAY_ALLOWED_ORIGINS=https://webchat.axynia.smart-admin.biz
GATEWAY_HEARTBEAT_SECONDS=25
GATEWAY_MAX_FRAME_BYTES=65536
GATEWAY_MAX_MESSAGE_CHARS=8000

MQTT_ENABLED=false
MQTT_HOST=
MQTT_PORT=1883
MQTT_USERNAME=
MQTT_PASSWORD=
MQTT_CLIENT_ID=gateway-01
MQTT_QOS=1

```

Las mismas claves no secretas fueron agregadas a .env.example. MQTT permanece deshabilitado hasta que se implemente y apruebe la fase correspondiente.

7.3 Clave y limpieza de cachés

```

php artisan key:generate --force
php artisan optimize:clear

```

8. Directorios documentales

```

mkdir -p docs/setup
mkdir -p docs/architecture
mkdir -p docs/specifications

```

Ruta	Contenido previsto
docs/setup/	Procedimientos de instalación y operación
docs/architecture/	Diagramas y decisiones de arquitectura
docs/specifications/	Especificaciones funcionales y de protocolo

9. Permisos del proyecto

El código queda administrado por root y accesible al grupo axynia-gateway. El proceso de servicio solamente necesita escritura en storage y bootstrap/cache.

```
cd /var/www/html
chown -R root:axynia-gateway gateway.axynia.smart-admin.biz
chmod -R g+rX,o-rwx gateway.axynia.smart-admin.biz
chown -R axynia-gateway:axynia-gateway gateway.axynia.smart-admin.biz/storage
gateway.axynia.smart-admin.biz/bootstrap/cache
chmod -R u+rwX,g+rwX,o-rwx gateway.axynia.smart-admin.biz/storage gateway.axynia.smart-admin.biz/bootstrap/cache
```

9.1 Protección de .env

```
cd /var/www/html/gateway.axynia.smart-admin.biz
chown root:axynia-gateway .env
chmod 640 .env

stat -c '%A %a %U:%G %n' .env
-rw-r----- 640 root:axynia-gateway .env
```

Seguridad: .env.example puede quedar en 0644 porque no debe contener secretos. .env debe permanecer fuera de Git y legible únicamente por root y el usuario/grupo del servicio.

10. Soporte SQLite del Laravel base

Aunque el Gateway no utilizará la base de datos como parte de su responsabilidad de transporte, se dejó consistente la instalación estándar de Laravel para ejecutar sus migraciones y pruebas base.

```
apt update
apt install -y php8.4-sqlite3

php -m | grep -E 'pdo_sqlite|sqlite3'

cd /var/www/html/gateway.axynia.smart-admin.biz
php artisan migrate --force
```

Regla arquitectónica: La disponibilidad de SQLite no habilita al Gateway a persistir sesiones, conversaciones o historial. Esa información seguirá perteneciendo al backend de Axynia.

11. Validaciones ejecutadas

11.1 Validación de Laravel

```
cd /var/www/html/gateway.axynia.smart-admin.biz
php artisan about
php artisan test
```

Control	Resultado
Aplicación	Axynia Gateway
Entorno	production
Debug	OFF
APP_URL	https://gateway.axynia.smart-admin.biz
Laravel	13.23.0
PHP	8.4.24

Tests base	2 pruebas, 2 assertions, PASS
------------	-------------------------------

11.2 Requisitos y auditoría Composer

```
COMPOSER_ALLOW_SUPERUSER=1 composer check-platform-reqs
COMPOSER_ALLOW_SUPERUSER=1 composer audit
```

Todos los requisitos de plataforma informaron success y Composer no detectó avisos de vulnerabilidades en las dependencias instaladas.

11.3 Validación como usuario de servicio

```
runuser -u axynia-gateway -- bash -c '
cd /var/www/html/gateway.axynia.smart-admin.biz &&
php artisan about
'

runuser -u axynia-gateway -- bash -c '
cd /var/www/html/gateway.axynia.smart-admin.biz &&
php artisan optimize:clear
'
```

El usuario axynia-gateway pudo cargar Laravel y limpiar las cachés, confirmando permisos correctos sobre storage y bootstrap/cache.

11.4 Controles de residuos y puerto

```
supervisorctl status

grep -RniE 'gateway\.axynia\.smart-admin\.biz|dist/index\.js|node .*gateway'
/etc/supervisor/conf.d /etc/supervisor/supervisord.conf || true

ss -ltnp | grep ':3100 ' || echo "Puerto 3100 disponible"
```

Se confirmó que no quedan procesos anteriores asociados al Gateway y que el puerto 3100 queda disponible hasta que el equipo implemente el endpoint PHP definitivo.

12. Estado final del proyecto

```
/var/www/html/gateway.axynia.smart-admin.biz/
├── app/
├── bootstrap/
├── config/
├── database/
├── docs/
│   ├── architecture/
│   ├── setup/
│   └── specifications/
├── public/
├── resources/
├── routes/
├── storage/
├── tests/
├── vendor/
├── .env
├── .env.example
├── artisan
├── composer.json
└── composer.lock
```

Elemento	Estado
Directorio definitivo	Creado y operativo
Laravel	Instalado y validado
Workerman	Instalado y cargable
Cliente MQTT	Instalado y cargable

Usuario de servicio	Validado
Permisos de escritura	Limitados a storage y bootstrap/cache
.env	root:axynia-gateway, modo 0640
SQLite base	Driver y migraciones disponibles
Supervisor	Servicio activo; programa del Gateway aún pendiente
Puerto 3100	Disponible para la futura implementación
Apache / TLS	Existente; no modificado por este setup

13. Próximo hito

- Definir la estructura final del código luego de la ronda de revisión de la especificación v0.02.
- Crear el endpoint persistente Laravel/Workerman o comando Artisan equivalente.
- Implementar el listener WebSocket en 127.0.0.1:3100 y el endpoint /ws.
- Reproducir primero el flujo de eco compatible con Deep Chat.
- Crear la configuración Supervisor con user=axynia-gateway.
- Validar el recorrido WSS -> Apache -> WS -> Gateway PHP.
- Incorporar MQTT y el protocolo de sesión en fases posteriores.

14. Lista de verificación reproducible

- [x] Debian, PHP, Composer, Git y Supervisor verificados.
- [x] pcntl, posix y sockets disponibles en PHP CLI.
- [x] Laravel 13 creado en el directorio definitivo.
- [x] Workerman y workerman/mqtt instalados mediante Composer.
- [x] Clases principales cargan desde vendor/autoload.php.
- [x] .env configurado para producción y protegido con 0640.
- [x] .env.example actualizado sin secretos.
- [x] storage y bootstrap/cache escribibles por axynia-gateway.
- [x] php8.4-sqlite3 instalado y migraciones base ejecutadas.
- [x] php artisan test finaliza correctamente.
- [x] composer check-platform-reqs finaliza sin errores.
- [x] composer audit no informa vulnerabilidades.
- [x] El usuario axynia-gateway puede iniciar Laravel y limpiar cachés.
- [x] El puerto 3100 queda libre hasta instalar el Gateway.

15. Registro documental

Versión	Fecha	Estado	Descripción
0.01	01/08/2026	Validado	Instalación base de Laravel 13, Workerman, workerman/mqtt, permisos y controles de plataforma.