Helpbuttons

Guía Completa de Instalación en Servidor Propio

Para personas que nunca han alquilado un servidor · Mayo 2026

Si tras ver esta guía se te hace imposible avanzar. Puedes escribirnos o saltar estos pasos y solicitar una red en helpbuttons.org . Crearemos la red por ti en la url que elijas.

¿Qué es Helpbuttons?

Software libre (licencia Mozilla Public License v2.0) para crear herramientas colaborativas en tu comunidad:

Compartir transporte, alimentación, apoyo mutuo, coordinación de emergencias.

Lo instalas en tu propio servidor y los datos son tuyos.

Repositorio principal: gitlab.com/helpbuttons/helpbuttons · Espejo: github.com/helpbuttons/helpbuttons

Comunidad: https://t.me/helpbuttonsgroup

Contacto: help@helpbuttons.org

Índice

Sección 0: (Opción A) Crear tu propio servidor

Recomendaciones de hardware y configuración del servidor local

Esta sección complementa la guía principal con información sobre qué ordenadores son adecuados para alojar Helpbuttons de forma local (en casa o en un local comunitario) y cómo preparar el sistema operativo paso a paso.

1. Ordenadores recomendados (últimos 15 años) y capacidad estimada de usuarios

La web oficial de Helpbuttons indica que un ordenador de 2010–2015 con 4 GB de RAM es suficiente para alojar una red con aproximadamente 3.500 usuarios registrados. La siguiente tabla recoge modelos populares ordenados por generación:

Ordenador / Familia	Año	CPU	RAM típica	Usuarios estimados	Notas
Intel Core 2 Duo (ej. iMac 2009)	2009–2011	Core 2 Duo E8400	2–4 GB	**500–1.000**	Límite de RAM; Docker puede ir justo
Dell OptiPlex 780/790	2010–2012	Core i3/i5 2ª gen	4–8 GB	**1.500–3.500**	Con 4 GB ya cubre el objetivo; muy común de segunda mano
Lenovo ThinkCentre M82	2012–2013	Core i5 3ª gen	4–16 GB	**3.500–6.000**	Ideal: silencioso, bajo consumo, fácil de ampliar RAM
Raspberry Pi 4 (4/8 GB)	2019–act.	ARM Cortex-A72	4–8 GB	**1.000–2.500**	ARM; Docker funciona, pero compilación más lenta
HP ProDesk 400 G3	2016–2017	Core i5 6ª gen	8–16 GB	**5.000–10.000**	Potencia cómoda; puede ampliar más allá del objetivo
Mac mini 2014	2014–2016	Core i5 4ª gen	4–16 GB	**3.500–7.000**	Compacto y silencioso; buen candidato para uso 24/7
Intel NUC 6ª–8ª gen	2016–2019	Core i3/i5/i7	8–32 GB	**10.000+**	Mini-PC de alto rendimiento; excede el objetivo
Laptop reciclado (genérico)	2012–2018	Core i3–i5 var.	4–8 GB	**2.000–4.000**	Batería actúa como UPS improvisado

✅ **Recomendación mínima: cualquier ordenador con procesador Intel Core i3/i5 de 2ª generación o posterior y al menos 4 GB de RAM (idealmente 8 GB). El almacenamiento puede ser HDD, aunque un SSD mejora notablemente los tiempos de respuesta.**

⚠️ **Nota sobre Raspberry Pi: aunque funciona, la arquitectura ARM puede ralentizar la compilación inicial de Docker. Si ya está compilado (imagen preconstruida), el rendimiento es aceptable para comunidades pequeñas.**

2. Configuración del ordenador como servidor

A continuación se detallan los pasos para convertir un ordenador básico en un servidor listo para Helpbuttons, desde la instalación del sistema operativo hasta la puesta en marcha en red.

Paso 1. Instalar Ubuntu Server (sin escritorio gráfico)

Descarga Ubuntu Server 22.04 LTS desde ubuntu.com/download/server. Grábalo en una USB con Balena Etcher. Arranca el ordenador desde la USB, sigue el instalador y elige 'Ubuntu Server' (sin escritorio) para ahorrar RAM. Crea un usuario con privilegios sudo y activa OpenSSH para gestión remota.

Paso 2. Actualizar el sistema y configurar el firewall

Al entrar por primera vez, actualiza todos los paquetes y abre solo los puertos necesarios:

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

sudo ufw allow OpenSSH

sudo ufw allow 80/tcp

sudo ufw allow 443/tcp

sudo ufw enable
```

Paso 3. Instalar Docker y docker-compose

Helpbuttons requiere Docker >= 24.0.7 y docker-compose >= 2.23.3:

```
sudo apt install -y docker.io docker-compose-plugin

sudo systemctl enable --now docker

sudo usermod -aG docker $USER
```

```
newgrp docker   # aplica el grupo sin cerrar sesión
```

Paso 4. Configurar IP estática o DDNS

Para que los usuarios accedan desde internet necesitas una dirección fija. Dos opciones:

- Opción A — IP estática: contrata con tu ISP una IP fija, abre los puertos 80 y 443 en el router y apunta tu dominio a esa IP.
- Opción B — DDNS: si tu ISP te da IP dinámica, usa No-IP o DuckDNS (gratuitos). Instala su cliente en el servidor para actualizar la IP automáticamente.

Paso 5. Instalar Nginx y HTTPS con Let's Encrypt

Nginx actúa como proxy inverso y certbot genera el certificado SSL gratuito:

```
sudo apt install -y nginx certbot python3-certbot-nginx
sudo certbot --nginx -d tudominio.com
sudo systemctl enable nginx
```

Paso 6. Optimizar el sistema para uso como servidor

Ajustes de rendimiento recomendados para un ordenador de gama baja:

```
# Deshabilitar interfaz gráfica si se instaló por error
sudo systemctl set-default multi-user.target
# Activar swap de 4 GB (seguridad extra si la RAM es justa)
sudo fallocate -l 4G /swapfile && sudo chmod 600 /swapfile
sudo mkswap /swapfile && sudo swapon /swapfile
echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab
```

Paso 7. Encender el servidor automáticamente tras un corte de luz

En la BIOS/UEFI del ordenador, busca la opción 'Power On After Power Loss' o 'Restore on AC Power Loss' y ponla en 'On' o 'Last State'. Así el servidor arranca solo si hay un corte de electricidad.

3. Requisitos mínimos de referencia rápida

Componente	Mínimo recomendado
Procesador	x86-64, 2 núcleos (Core i3/i5, 2ª gen o superior)
Memoria RAM	4 GB mínimo · 8 GB recomendado
Almacenamiento	20 GB libres (SSD preferible, HDD válido)
Sistema operativo	Ubuntu Server 22.04 LTS (sin escritorio)
Docker	≥ 24.0.7 con docker-compose ≥ 2.23.3
Conexión a internet	10 Mbps simétrico mínimo · IP fija o DDNS

Componente	Mínimo recomendado
Consumo eléctrico aprox.	15–65 W según el hardware (24/7)
Swap recomendada	4 GB si la RAM es de 4 GB

💡 **Consejo final: un portátil reciclado con la tapa cerrada puede ser un excelente servidor silencioso y bajo consumo. La batería actúa como SAI (UPS) improvisado, protegiéndote de cortes de luz breves sin necesidad de hardware adicional.**

Parte 1 — (Opción b) Elegir y contratar un servidor en la nube

Si nunca has alquilado un servidor, aquí va la explicación básica: estás alquilando un ordenador pequeño que está en un centro de datos, encendido las 24 horas del día. Tú te conectas a él desde tu propio ordenador a través de un programa llamado SSH (básicamente una ventana de terminal remota). Escribes comandos y las cosas ocurren en ese ordenador remoto.

Requisitos recomendado para Helpbuttons

- 2 GB de RAM (4 GB recomendado para uso cómodo)
- 2 vCPU
- 1 GB de disco (puede ser menos para empezar)
- Sistema operativo: Ubuntu 22.04 o 24.04 LTS

¿Qué sistema operativo elegir?

Siempre Ubuntu LTS. Cuando el proveedor te pregunte, elige Ubuntu 22.04 LTS o Ubuntu 24.04 LTS.

Nunca elijas 'Ubuntu latest' sin número — podría ser una versión en desarrollo.

LTS significa Long Term Support: 5 años de actualizaciones de seguridad garantizadas.

Opciones de proveedores

Si no tienes tu propio servidor, te recomendamos buscar un proveedor de tu zona, que siempre suele haber. Alguien de confianza al que puedas consultar. Así tendrás control real de la tecnología que usáis. Si no encuentras, otros proveedores de servidores son:

Proveedor	Precio/mes	RAM/CPU	Dónde	Para quién
DigitalOcean	$12–24/mes	2–4 GB / 1–2 vCPU	Global	⭐ Mejor para principiantes: interfaz clara, documentación excelente
Hetzner	€4–8/mes	2–4 GB / 2 vCPU	Alemania / Finlandia	Mejor precio. GDPR europeo. Recomendado si estás en Europa
Vultr	$12/mes	2 GB / 1 vCPU	Global	Similar a DigitalOcean, buena cobertura global
Linode/Akamai	$12/mes	2 GB / 1 vCPU	Global	Veterano y fiable. Buena documentación

OVHcloud	€3.50/mes	2 GB / 1 vCPU	Europa/Global	El más barato. Menos amigable para principiantes

Cómo contratar paso a paso (ejemplo con DigitalOcean)

Crear cuenta + servidor — tiempo total	⏱️ **20–30 minutos**

1. Ve a digitalocean.com y crea una cuenta. Tendrás $200 de crédito gratis durante 60 días.
2. Pulsa el botón verde 'Create' → 'Droplets' (así llaman a sus servidores virtuales).
3. En 'Choose an image': selecciona Ubuntu → 22.04 LTS.
4. En 'Choose a plan': selecciona 'Basic' → el plan de $12/mes (2 GB RAM / 1 vCPU) o $24/mes (4 GB / 2 vCPU).
5. En 'Choose a datacenter region': elige el más cercano a tus usuarios (Frankfurt para Europa, Nueva York o San Francisco para América).
6. En 'Authentication': selecciona 'Password' si es tu primera vez (más simple). Pon una contraseña fuerte.
7. Pulsa 'Create Droplet' y espera 1–2 minutos. Recibirás una IP como 134.209.45.12.

Nota para Hetzner (la opción más económica en Europa)

1. Ve a hetzner.com/cloud → Cloud Console → New Project → Add Server.

2. Elige ubicación: Falkenstein o Núremberg (Alemania) o Helsinki (Finlandia).

3. Imagen: Ubuntu 22.04.

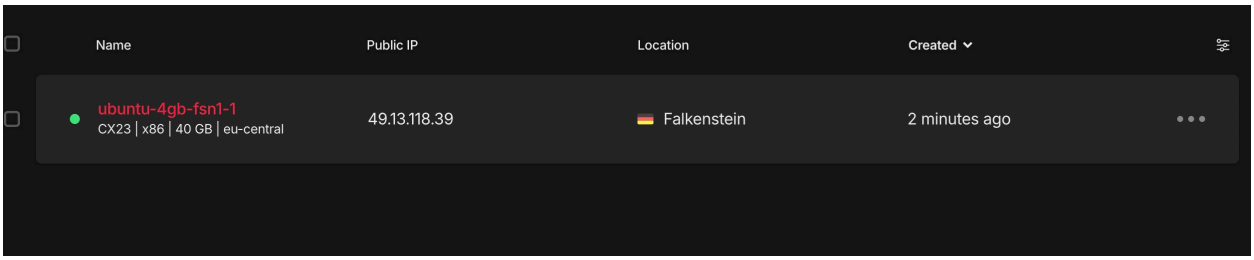
4. Tipo: CX23 (2 vCPU, 4 GB RAM, 40 GB disco) por ~€4.15/mes.

5. Autenticación: crea una clave SSH o usa contraseña.

6. Ignora los demás campos

6. Pulsa 'Create & Buy Now'.

Parte 2 — Conectarte a tu servidor por primera vez



(Vista de Hetzner donde se ve la IP)

Conexión inicial	⏱️ **5–10 minutos**

Una vez creado el servidor, tienes una dirección IP (algo como 134.209.45.12). Ahora abres una terminal (es una ventana para meter el código, se explica después) en TU ordenador y te conectas a ese servidor remoto.

¿Cómo abrir una terminal?

- Mac: Applications → Utilities → Terminal (o pulsa Cmd+Space y escribe 'Terminal').
- Windows: instala 'Windows Terminal' desde Microsoft Store, o usa PuTTY (putty.org). También puedes activar WSL (busca 'Windows Subsystem for Linux' en Configuración).
- Linux: ya sabes dónde está tu terminal.

Comando de conexión

Sustituye la IP por la tuya real:

```
ssh root@134.209.45.12
```

Si aparece un aviso que dice 'The authenticity of host can't be established' — escribe yes y pulsa Enter. Es normal la primera vez.

Si pusiste contraseña al crear el servidor, te la pedirá. (Hetzner te la envía al email) Escríbela y pulsa Enter. No verás los caracteres mientras escribes — eso es normal, es una medida de seguridad. Copia y pega. Hetzner te pedirá cambiar la contraseña. Pega la antigua primero, y te pedirá la nueva después.

Si todo va bien, verás algo como 'root@ubuntu-s-1vcpu-2gb:~#'. Estás dentro de tu

servidor. Todo lo que escribas ahora ocurre en el ordenador remoto.

Parte 3 — Preparar el servidor (instalación única)

Actualización del sistema + instalación de Docker	⏱️ 15–20 minutos

Estos comandos sólo hay que ejecutarlos una vez. Copia cada bloque, pégalo en el terminal, pulsa Enter y espera a que termine antes de pasar al siguiente.

Paso 3.1 — Actualizar el sistema

(Descarga actualizaciones de seguridad. Tarda 3–5 minutos.)

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

Paso 3.2 — Instalar Docker

Docker es el programa que gestiona los contenedores donde corre Helpbuttons. (Tarda 5–10 minutos.)

```
curl -fsSL https://get.docker.com -o get-docker.sh

sh get-docker.sh
```

Paso 3.3 — Instalar Docker Compose y Git

```
apt install -y docker-compose-plugin git
```

Paso 3.4 — Verificar que Docker funciona

Ambos comandos deben imprimir un número de versión. Si ves 'command not found', algo falló en la instalación.

```
docker --version

docker compose version
```

¿Qué es Docker?

Docker empaqueta la aplicación y todas sus dependencias en 'contenedores'.

Piensa en ellos como cajas selladas que contienen todo lo necesario para ejecutar el programa.

La ventaja: no tienes que instalar Node.js, PostgreSQL, Redis, etc. por separado.

Docker lo gestiona todo. Solo necesitas instalar Docker una vez.

Parte 4 — Descargar y configurar Helpbuttons

Descarga + configuración del archivo .env	⏱️ 15–20 minutos

Paso 4.1 — Descargar el código

```
git clone https://github.com/helpbuttons/helpbuttons.git

cd helpbuttons
```

Ahora estás dentro de la carpeta 'helpbuttons' en tu servidor.

Paso 4.2 — Crear el archivo de configuración

```
cp env.sample .env

nano .env
```

Esto abre un editor de texto básico dentro del terminal. Verás estas variables que debes rellenar:

```
POSTGRES_HOSTNAME=db
POSTGRES_DB=hb-db
POSTGRES_USER=postgres
POSTGRES_PASSWORD=changeme        ← CAMBIA ESTO por algo seguro
POSTGRES_PORT=5432

hostName=example.com              ← tu dominio o la IP de tu servidor
smtpHost=                         ← servidor de email (localhost si no tienes)
smtpPort=                         ← 1025 si no usas email
smtpUser=                         ← dummy si no usas email
smtpPass=                         ← dummy si no usas email
from=                             ← noreply@localhost si no usas email

jwtSecret=                        ← lo generamos en el siguiente paso
REDIS_HOST=redis
REDIS_PORT=6379
VERSION=main                      ← o una versión concreta como v5.5.1
WEB_URL=http://tu-ip:3000         ← URL donde accederán los usuarios
API_URL=http://api:3001
```

Lo más importante que debes cambiar

- POSTGRES_PASSWORD — usa algo como MiClave!Segura2025 (no uses 'changeme')
- hostName — el dominio que tengas, o la IP de tu servidor si no tienes dominio todavía
- WEB_URL — la URL completa: http://134.209.45.12:3000 (con tu IP real)

Para editar en nano: usa las flechas del teclado para moverte, escribe directamente para cambiar el texto.
Para guardar y salir: pulsa Ctrl+X, luego Y, luego Enter.

Si quieres email real (recomendado)

Brevo (brevo.com), Mailgun o SendGrid tienen capa gratuita.

Te dan credenciales SMTP reales. Sustitúyelas en el .env:

smtpHost=smtp-relay.brevo.com

smtpPort=587

smtpUser=tu@email.com

smtpPass=tu-clave-api

from=tu@email.com

Por qué aparece el error aunque la app funcione

El API intenta verificar la conexión SMTP al arrancar.

Si los campos están vacíos, intenta conectar a 127.0.0.1:587 que no existe.

Verás este mensaje en los logs (no es un error crítico, solo un aviso):

ERROR [MailerService] connect ECONNREFUSED 127.0.0.1:587

Con valores ficticios el aviso desaparece.

Paso 4.3 — Generar la clave secreta JWT

Esta clave es como una firma criptográfica para las sesiones de usuario. Se genera automáticamente:

```
docker compose run api yarn cli config:genjwt
```

Imprimirá una cadena larga de caracteres. Cópiala. Luego abre de nuevo el .env:

```
nano .env
```

Busca la línea 'jwtSecret=' y pega el valor generado después del signo igual. Guarda y sal (Ctrl+X, Y, Enter).

¿Qué es el jwtSecret?

Es una clave que el servidor usa para firmar los tokens de autenticación de los usuarios.

Si cambias este valor después de que haya usuarios registrados, todas las sesiones activas se invalidarán

y todos tendrán que volver a hacer login. Guárdalo en un lugar seguro.

Paso 4.4 — Añadir internal_network al contenedor web

Abre el docker-compose.yml:

```
nano docker-compose.yml
```

Busca la sección del servicio web. Cambia esto:

```
  web:
    image: helpbuttons/helpbuttons-web:${VERSION}
    env_file:
      - .env
    ports:
      - "3000:3000"
    depends_on:
      - api
    networks:
      - external_network
```

Por esto (solo cambia la parte de networks, añade internal_network):

```
  web:
    image: helpbuttons/helpbuttons-web:${VERSION}
    env_file:
      - .env
    ports:
      - "3000:3000"
    depends_on:
      - api
    networks:
      - external_network
      - internal_network
```

Paso 4.5 — Añadir internal_network al contenedor web

En el mismo archivo docker-compose.yml, busca la sección de redes al final del archivo. Cambia esto:

```
networks:

  external_network:

  internal_network:

    internal: true
```

Por esto:

```
networks:

  external_network:

  internal_network:
```

Guarda con Ctrl+X, Y, Enter.

Verificación

Después de reiniciar (ver más abajo), comprueba que el web puede ver al api:

```
docker compose exec web wget -qO- http://api:3001/networks/config
```

Debe devolver un JSON con hostName, databaseNumberMigrations y userCount.

```
docker compose exec api wget -qO- https://photon.komoot.io/api/?q=madrid
```

Debe devolver un JSON con datos de localización de Madrid. Si es así, la búsqueda en el mapa funcionará.

Parte 5 — Arrancar Helpbuttons

Arranque + primera verificación	⏱️ 10–20 minutos

Paso 5.1 — Arrancar todos los contenedores

La primera vez descarga las imágenes de Docker (puede tardar varios minutos según la conexión).

```
docker compose up -d
```

El '-d' significa que corre en segundo plano (detached). Verás un montón de texto y al final volverá el prompt.

Paso 5.2 — Crear la estructura de la base de datos

```
docker compose run api yarn migration:run
```

Esto configura todas las tablas que necesita la aplicación. Solo hay que hacerlo la primera vez (y cada vez que actualizas).

Paso 5.3 — Verificar que todo funciona

```
docker compose ps
```

Deberías ver cuatro contenedores listados: web, api, db, redis — todos con estado 'Up'. Si alguno dice 'Exit' o 'Restarting', algo fue mal (consulta la sección de problemas al final).

Paso 5.4 — Abrir la aplicación

Abre un navegador y ve a:

```
http://tu-ip-del-servidor:3000
```

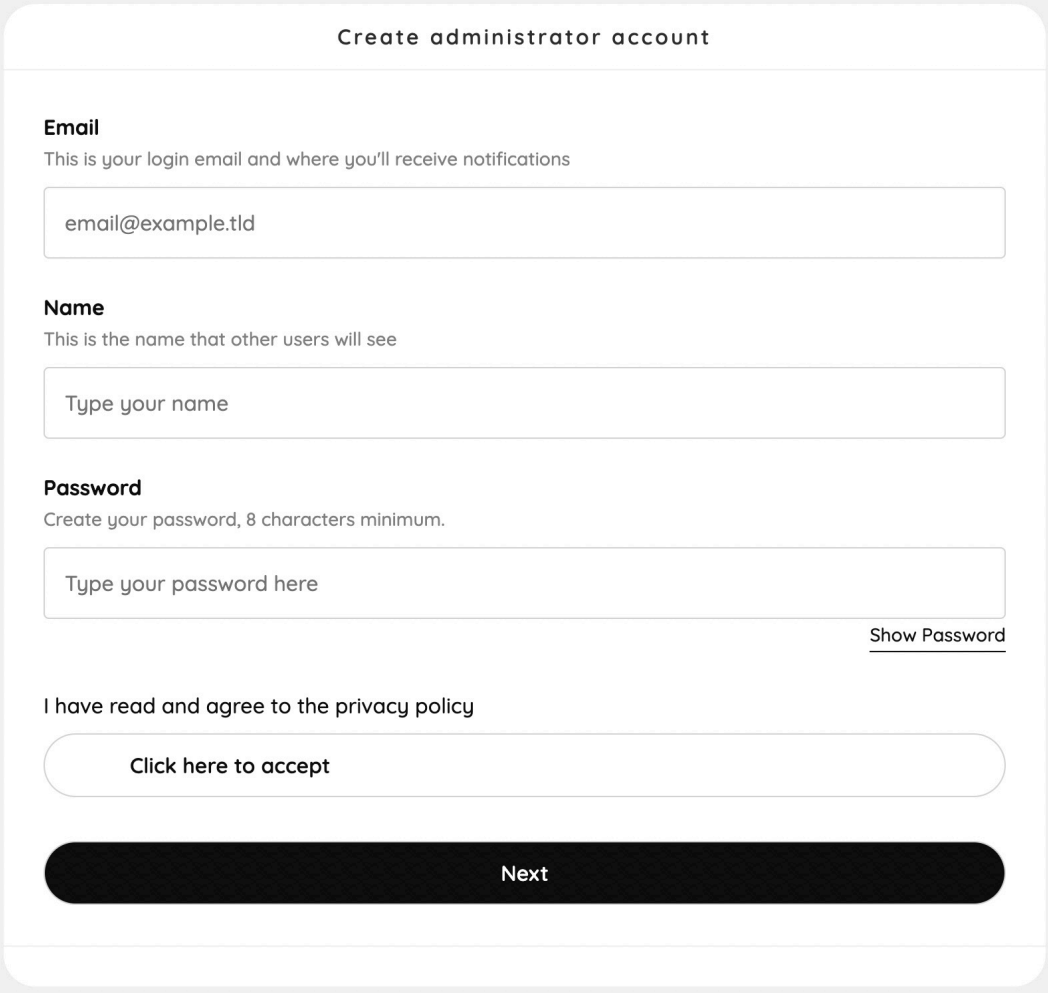
Deberías ver la interfaz de Helpbuttons. ¡Enhorabuena, la aplicación está funcionando!

Si no puedes acceder al puerto 3000

Algunos proveedores tienen un firewall activo por defecto. Abre el puerto ejecutando:

```
ufw allow 3000
ufw allow 22
ufw enable
```

En Hetzner puedes también gestionar el firewall desde el panel web (Firewall → Add Rule).



Paso 5.5 — Rellena los campos

Ahora la interfaz the pedirá que rellenes los campos para crear una cuenta de administrador y los datos de tu nueva Red. Cuando completes esos dos pasos ya tendrás una Red de Helpbuttons operativa. ¡Enhorabuena! Para consultar consejos acerca de cómo configurar tu red acude al siguiente enlace:

https://wofreedom.org/como-activar-tu-red-en-helpbuttons-guia-paso-a-paso/

Parte 6 — Dominio propio y HTTPS (para producción real)

Configuración completa con dominio + HTTPS	⏱️ **30–60 minutos + espera DNS**

Por ahora funciona en el puerto 3000, lo que es suficiente para probar. Para uso real con usuarios necesitas un dominio y el candado HTTPS.

Paso 6.1 — Escríbenos o Contrata un dominio

Te podemos ofrecer un dominio de tipo tudominio.botones.org o tudominio.helpbuttons.org de forma gratuita. Si quieres usar un dominio tuyo comprado (como tudominio.com) aquí tienes opciones.

- Namecheap (namecheap.com) — ~10€/año para .com, interfaz clara
- Cloudflare Registrar (cloudflare.com) — precio de coste sin margen, muy bueno
- Porkbun (porkbun.com) — barato, buena interfaz
- OVH (ovh.com) — opción europea económica

Después de comprarlo, ve a la configuración DNS de tu dominio y añade un registro A:

```
Tipo:  A

Nombre: @  (o 'helpbuttons' para subdominio)

Valor:  134.209.45.12  (tu IP real)

TTL:   3600
```

Los cambios DNS tardan entre 5 minutos y 48 horas en propagarse (normalmente menos de 30 minutos).

Paso 6.2 — Instalar Nginx como proxy inverso

```
apt install -y nginx
```

Paso 6.3 — Configurar Nginx

```
nano /etc/nginx/sites-available/helpbuttons
```

Pega esto (cambia tudominio.com por tu dominio real o tudominio.botones.org dependiendo de la opción que hayas elegido):

```
server {
    listen 80;
    server_name tudominio.com;

    location / {
        proxy_pass http://localhost:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_cache_bypass $http_upgrade;
    }
}
```

Guarda con Ctrl+X, Y, Enter.

Paso 6.4 — Activar la configuración

```
ln -s /etc/nginx/sites-available/helpbuttons /etc/nginx/sites-enabled/
nginx -t
systemctl reload nginx
```

El comando 'nginx -t' verifica que no haya errores. Si dice 'syntax is ok', todo está bien.

Paso 6.5 — Añadir HTTPS gratuito con Let's Encrypt

```
apt install -y certbot python3-certbot-nginx
certbot --nginx -d tudominio.com
```

Sigue los pasos: introduce tu email, acepta los términos, elige si redirigir HTTP a HTTPS

(elige sí). El certificado se renovará automáticamente cada 90 días.

Paso 6.6 — Actualizar el .env con la nueva URL

```
nano .env
```

Cambia la línea WEB_URL:

```
# Antes:
WEB_URL=http://tudominio.com:3000


# Después:
WEB_URL=https://tudominio.com
```

Luego reinicia la aplicación:

```
docker compose down
docker compose up -d
docker compose run api yarn migration:run
```

Parte 7 — Mantenimiento y actualizaciones

Ver los logs si algo falla

```
docker compose logs -f              # todos los contenedores
docker compose logs -f api          # solo la API
docker compose logs -f web          # solo el frontend
docker compose logs -f db           # solo la base de datos
```

Pulsa Ctrl+C para salir de la vista de logs.

Actualizar a una nueva versión

```
cd helpbuttons
git pull
nano .env           # cambia VERSION= al número de versión nuevo
docker compose pull
docker compose up -d
docker compose run api yarn migration:run
```

Hacer una copia de seguridad de la base de datos

```
docker compose exec db pg_dump -U postgres hb-db > backup_$(date +%Y%m%d).sql
```

Esto crea un archivo SQL con todos los datos. Descárgalo a tu ordenador con SCP:

```
scp root@tu-ip:~/helpbuttons/backup_20260501.sql ./
```

Restaurar desde una copia de seguridad

```
docker compose exec -T db psql -U postgres hb-db < backup_20260501.sql
```

Acceder directamente a la base de datos

```
docker compose exec db psql -U postgres hb-db
```

Reiniciar completamente (¡elimina todos los datos!)

> ⚠️ **ADVERTENCIA**
>
> El siguiente comando borra TODOS los datos de la base de datos. No hay vuelta atrás.
>
> Úsalo solo si quieres empezar desde cero o si la base de datos está corrupta.

```
docker compose down -v

docker compose up -d

docker compose run api yarn migration:run
```

Resumen de tiempos estimados

Estimaciones para alguien que nunca ha hecho esto antes:

Tarea	Principiante	Con experiencia
Elegir proveedor y crear cuenta	20–30 min	10 min
Crear el servidor y obtener la IP	10–15 min	5 min
Conectarse por SSH por primera vez	5–10 min	2 min
Actualizar sistema + instalar Docker	15–20 min	10 min
Descargar el código + configurar .env	15–20 min	5 min
Arrancar contenedores + migraciones	10–20 min	5 min
Primera vez que funciona la app	~2 horas total	35–40 min
Añadir dominio propio + HTTPS	+30–60 min	+15 min
Instalación completa en producción	2.5–4 horas	50–60 min

Una vez que lo has hecho una vez

La segunda instalación (en otro servidor o tras reinstalar) tarda 30–45 minutos.

Actualizar a una versión nueva tarda menos de 5 minutos.

Los backups diarios pueden automatizarse con un cron job.

Problemas frecuentes y soluciones

'Connection refused' al abrir la app en el navegador

- Los contenedores todavía están iniciando. Espera 1–2 minutos e inténtalo de nuevo.
- Ejecuta 'docker compose ps' para ver si todos están 'Up'.
- El firewall puede estar bloqueando el puerto 3000. Ejecuta: ufw allow 3000

La migración de la base de datos falla

Normalmente significa que la base de datos no está lista. Espera 30 segundos y vuelve a ejecutar:

```
docker compose run api yarn migration:run
```

Error 'Permission denied' con Docker

```
newgrp docker
```

O cierra sesión y vuelve a entrar con ssh.

El servidor se queda sin memoria (OOM)

Síntoma: contenedores se reinician solos, 'Killed' en los logs. Solución — añadir swap:

```
fallocate -l 2G /swapfile

chmod 600 /swapfile

mkswap /swapfile

swapon /swapfile
```

O mejor: sube el plan del servidor a 4 GB de RAM.

El email no funciona (notificaciones)

- Los campos SMTP en .env son opcionales. Sin ellos, la app funciona pero no envía emails.
- Para emails puedes usar servicios con capa gratuita: SendGrid, Mailgun, o Brevo.

- Te dan credenciales SMTP que pegas en el .env: smtpHost, smtpPort, smtpUser, smtpPass, from.

Quiero que la app arranque automáticamente si el servidor se reinicia

Crea un servicio systemd. Primero, crea el archivo:

```
nano /etc/systemd/system/helpbuttons.service
```

Pega esto:

```
[Unit]
Description=Helpbuttons
After=docker.service
Requires=docker.service

[Service]
Type=oneshot
RemainAfterExit=yes
WorkingDirectory=/root/helpbuttons
ExecStart=/usr/bin/docker compose up -d
ExecStop=/usr/bin/docker compose down

[Install]
WantedBy=multi-user.target
```

Actívalo:

```
systemctl enable helpbuttons
systemctl start helpbuttons
```

Comunidad y ayuda

Si te quedas atascado o algo no funciona como esperabas:

- Mastodon: fosstodon.org/@helpbuttonsorg
- Telegram: t.me/+ls0xkQlG8uBlZjZk
- Email: help@helpbuttons.org
- Issues (problemas técnicos): github.com/helpbuttons/helpbuttons/issues
- Documentación adicional: github.com/helpbuttons/hb-docs

¿Prefieres que lo gestionen ellos?

El equipo de Helpbuttons también ofrece instancias alojadas si no quieres gestionar un servidor.

Consulta helpbuttons.org para más información sobre esta opción.

El proyecto está financiado por Wof! (wofreedom.org), una organización sin ánimo de lucro.

Si quieres apoyar el proyecto: buy.stripe.com/9AQ5kl3CYaIvgRW6ou