

# Guía de despliegue

---

Esta aplicación tiene un backend Node.js/Express y un frontend React (Vite). Puedes desplegarla de dos formas:

- A) Monolítica en cPanel (backend y frontend juntos)
- B) Separada: frontend en cPanel (<https://cotizador.aysafi.com>) y backend en un VPS (<https://emqx.aysafi.com>)

A continuación están ambos flujos. Si buscas el escenario "separado", ve directo a la sección B.

---

## A) Monolítico en cPanel (todo junto)

En cPanel podemos usar Application Manager (Node.js/Passenger) para correr el backend y servir la SPA construida.

### 1) Preparar build local y subir archivos

- Configura tu `.env` local con valores reales (ver "Variables de entorno" más abajo).
- (Recomendado) Lint rápido: `npm run lint`
- Construye el frontend: `npm run frontend:build`
- Verifica que existan: `frontend/dist/`, carpeta `backend/` y carpeta `outputs/`.
- Sube el proyecto a cPanel (FTP o Administrador de Archivos). No subas `node_modules/`.

### 2) Crear aplicación Node.js en cPanel

- Setup Node.js App → Application root: carpeta del proyecto (ej. `/home/usuario/cotizador`)
- Application URL: tu subdominio (ej. `https://cotizador.tudominio.cl`)
- Startup file: `backend/server.js`
- Node.js 18 o 20 (recomendado 20) → Crear

### 3) Variables de entorno (en cPanel)

- `JWT_SECRET`: secreto aleatorio y único (requerido para login admin)
- `ADMIN_PASSWORD`: contraseña admin para emitir el token (opcional pero útil)
- `FRONTEND_URL`: URL pública del frontend (ej. `https://cotizador.tudominio.cl`)
- `PUBLIC_API_BASE`: déjala vacía cuando frontend y backend están en el mismo dominio; si usas otro host, pon la URL absoluta del backend
- `SMTP_HOST`, `SMTP_PORT`, `SMTP_USER`, `SMTP_PASS` (si enviarás correos)
- `OUTPUTS_DIR` (opcional): ruta absoluta para guardar PDFs/QR/JSON si no quieres usar `outputs/` dentro del proyecto
- `PORT` (opcional): cPanel/Passenger puede ignorarlo; no estorba

### 4) Instalar dependencias en el servidor

- Terminal de la app o SSH en el root del proyecto:

- `npm ci`
- `cd frontend && npm ci && cd ..`
- (Si prefieres construir en servidor) `npm run build`

## 5) Estáticos y SPA

- El backend sirve `frontend/dist` y hace fallback SPA para rutas no `/api` ni `/outputs`.
- El backend expone `/config.js` con `window.__APP_CONFIG__` para configurar el frontend en tiempo de ejecución (no requiere rebuild). Cambia env vars y reinicia.

## 6) Comprobación

- Reinicia la app y prueba:
  - `GET /` → `{ ok: true }`
  - Navega a la raíz → SPA carga
  - `/admin/config` → muestra valores de runtime y `GET /api/config`

## 7) Permisos y seguridad

- Asegura que `outputs/` sea escribible.
- No subas `.env` al repo; usa variables en cPanel.
- Rota `JWT_SECRET`/credenciales SMTP tras pruebas.

## 8) SMTP troubleshooting (Nodemailer)

- `ETIMEDOUT/Greeting never received`: revisa host/puerto/seguridad; prueba `npm run verify:smtp` y `npm run send:test-email`.

---

## B) Separado: frontend en cPanel y backend en VPS

Objetivo: servir el frontend estático en `https://cotizador.aysafi.com` (cPanel) y el backend en `https://emqx.aysafi.com` (VPS). El frontend llamará al backend vía `PUBLIC_API_BASE` usando `window.__APP_CONFIG__` cargado desde `config.js`.

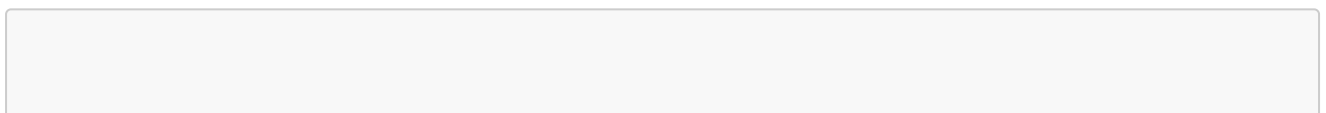
### 1) Backend en VPS (<https://emqx.aysafi.com>)

Requisitos en el VPS (Ubuntu/Debian típico):

- DNS del dominio `emqx.aysafi.com` apuntando a la IP del VPS
- Node.js 20 y npm
- Nginx como reverse proxy con SSL (Let's Encrypt)
- Acceso SSH y permisos para crear un servicio (systemd)

Pasos:

1. Clonar e instalar dependencias



```
sudo mkdir -p /opt/cotizador && sudo chown $USER:$USER /opt/cotizador
cd /opt/cotizador
git clone https://github.com/asdrubalfuentes/cotizador .
npm ci
```

#### 1. Variables de entorno del backend (archivo `.env` en `/opt/cotizador`)

```
# URL públicas
FRONTEND_URL=https://cotizador.aysafi.com
PUBLIC_API_BASE=https://emqx.aysafi.com

# Seguridad
JWT_SECRET=pon-aqui-un-secreto-largo-unico
ADMIN_PASSWORD=elige-una-clave-admin

# SMTP (si enviarás correos)
SMTP_HOST=smtp.tu-proveedor.com
SMTP_PORT=587
SMTP_USER=usuario@tu-dominio.com
SMTP_PASS=tu-pass

# Otros (opcionales)
OUTPUTS_DIR=/var/lib/cotizador/outputs
MORGAN_FORMAT=combined
```

#### 1. Directorio de salidas (si usas ruta externa)

```
sudo mkdir -p /var/lib/cotizador/outputs
sudo chown -R $USER:$USER /var/lib/cotizador
```

#### 1. Servicio systemd (opcional recomendado)

Archivo: `/etc/systemd/system/cotizador.service`

```
[Unit]
Description=Cotizador Backend
After=network.target

[Service]
Type=simple
WorkingDirectory=/opt/cotizador
Environment=NODE_ENV=production
ExecStart=/usr/bin/node backend/server.js
Restart=always
RestartSec=5
```

```
# User=cotizador ; si creas un usuario de servicio
```

[Install]

```
WantedBy=multi-user.target
```

Activar e iniciar:

```
sudo systemctl daemon-reload  
sudo systemctl enable cotizador --now
```

## 1. Nginx reverse proxy con SSL y SSE

Archivo: `/etc/nginx/sites-available/cotizador-backend`

```
server {  
    listen 80;  
    server_name emqx.aysafi.com;  
    location /.well-known/acme-challenge/ { root /var/www/html; }  
    location / { return 301 https://$host$request_uri; }  
}  
  
server {  
    listen 443 ssl http2;  
    server_name emqx.aysafi.com;  
  
    ssl_certificate /etc/letsencrypt/live/emqx.aysafi.com/fullchain.pem;  
    ssl_certificate_key /etc/letsencrypt/live/emqx.aysafi.com/privkey.pem;  
  
    client_max_body_size 10m; # subir logos (la app limita 5MB)  
  
    location / {  
        proxy_pass http://127.0.0.1:3000; # puerto donde corre Node  
        proxy_http_version 1.1;  
        proxy_set_header Host $host;  
        proxy_set_header X-Real-IP $remote_addr;  
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;  
        proxy_set_header X-Forwarded-Proto $scheme;  
  
        # SSE (EventSource): sin buffer y con timeouts amplios  
        proxy_read_timeout 3600s;  
        proxy_send_timeout 3600s;  
        proxy_buffering off;  
    }  
}
```

Activar sitio y recargar:

```
sudo ln -s /etc/nginx/sites-available/cotizador-backend /etc/nginx/sites-enabled/  
sudo nginx -t && sudo systemctl reload nginx
```

### 1. Comprobaciones rápidas del backend

- `curl https://emqx.aysafi.com/` → JSON { ok: true, ... }
- `curl -I https://emqx.aysafi.com/api/events` → Content-Type: text/event-stream

### Notas CORS/SSE:

- CORS: el backend usa `cors()` abierto; para restringir: `cors({ origin: 'https://cotizador.aysafi.com' })`.
- SSE: con `proxy_buffering off` y timeouts altos, EventSource funciona estable tras proxies.

## 2) Frontend estático en cPanel (<https://cotizador.aysafi.com>)

### 1. Build local y subida

- Ejecuta: `npm run frontend:build`
- En cPanel, crea el subdominio `cotizador.aysafi.com` apuntando a un docroot (ej. `/home/usuario/public_html/cotizador`).
- Sube todo el contenido de `frontend/dist/` al docroot.
  - Si usaste `npm run package:cpanel`, dentro de `release/cpanel-.../frontend/` encontrarás dos utilidades:
    - `config.js.template`: renómbralo a `config.js`, edita las URLs y súbelo al docroot.
    - `.htaccess`: archivo listo para SPA fallback en Apache/cPanel; súbelo al docroot si aún no existe.

### 1. Crear `/config.js` en el docroot (enganche al backend)

### Contenido del archivo:

```
window.__APP_CONFIG__ = {  
  API_BASE: 'https://emqx.aysafi.com',  
  FRONTEND_URL: 'https://cotizador.aysafi.com'  
};
```

Nuestro `index.html` ya incluye `<script src="/config.js"></script>`. Para cambiar de backend en el futuro, solo edita este archivo (no requiere rebuild).

### 1. Habilitar SPA fallback con `.htaccess` (rutas profundas como `/admin/config`)

```
RewriteEngine On  
RewriteBase /  
RewriteCond %{REQUEST_FILENAME} !-f
```

```
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule . /index.html [L]
```

#### 1. SSL del subdominio

- Activa AutoSSL/Let's Encrypt en cPanel para `cotizador.aysafi.com`.

#### 1. Verificación end-to-end

- Abre `https://cotizador.aysafi.com/` y en la consola ejecuta `window.__APP_CONFIG__` (debe mostrar `API_BASE`/`FRONTEND_URL` correctos).
- En la UI, entra a "Config runtime" (`/admin/config`) para revisar `window.__APP_CONFIG__` y probar `/api/config`.
- Crea/edita registros para confirmar llamadas a `https://emqx.aysafi.com/*`.

---

## Variables de entorno (resumen y propósito)

- `PUBLIC_API_BASE`: Base pública del backend para el frontend.
  - Split: `https://emqx.aysafi.com`
  - Monolítico: vacío (mismo dominio)
- `FRONTEND_URL`: URL pública del frontend (correos/QR), ej. `https://cotizador.aysafi.com`.
- `JWT_SECRET`: firma del token admin; único y largo.
- `ADMIN_PASSWORD`: contraseña para obtener token admin (`POST /api/admin/login`).
- `SMTP_HOST`, `SMTP_PORT`, `SMTP_USER`, `SMTP_PASS`: correo saliente.
- `OUTPUTS_DIR` (opcional): ruta de PDFs/QR/JSON cuando no quieras usar `outputs/` interno.
- `MORGAN_FORMAT` (opcional): formato de logs HTTP.

---

## Cómo "enganchar" frontend y backend en dominios distintos (versión simple)

1. Asegúrate de que el backend responda en `https://emqx.aysafi.com` y permita CORS.
2. En cPanel, crea/edita `/config.js` con `API_BASE = 'https://emqx.aysafi.com'` y `FRONTEND_URL = 'https://cotizador.aysafi.com'`.
3. Para cambios futuros de URL, edita solo `config.js` y recarga el navegador (no necesitas rebuild).
4. Verifica en `/admin/config` que la configuración runtime sea correcta.