



INSTRUCCIONES PARA REBUILD EN EASYPANEL

Fecha: 29 de Octubre 2025

Proyecto: EscalaFin MVP

URL Deploy: <https://demo.escalafin.com>



ESTADO ACTUAL - VERIFICADO

Commit actual en GitHub: `c1dbdb4`

Última actualización: Validación completa de sincronización (29 Oct 2025)

Features incluidos:

- ☒ Configuración de Chatwoot mediante BD y variables de entorno
- ☒ Scripts de inicio mejorados (adaptados de CitaPlanner)
- ☒ Setup automático de usuarios de prueba
- ☒ Dockerfile optimizado para Node 18 + NPM



PASOS PARA REBUILD EN EASYPANEL



1 Acceder a EasyPanel

URL: <https://panel.qhosting.com> (o tu URL de EasyPanel)



2 Localizar el Proyecto

1. Login en EasyPanel
2. Navegar a "Projects"
3. Seleccionar: "escalafin" o "demo-escalafin"



3 Verificar Configuración de GitHub

Settings → Source:


- Repository: <https://github.com/qhosting/escalafin>
- Branch: main
- Dockerfile: Dockerfile (en la raíz del repo)



4 Limpiar Cache de Build

Build Settings:

1. Click en "Build Settings" o "Settings"
2. Buscar opción "Clear Build Cache" o "Delete Cache"
3. Click en el botón para limpiar cache
4. Confirmar la acción

 **IMPORTANTE:** Este paso es CRÍTICO. Sin limpiar el cache, EasyPanel puede usar archivos antiguos del build anterior.

5 Hacer Rebuild

1. Click en "Rebuild" o "Deploy"
2. Esperar a que el build complete
3. Monitorear los logs de build

6 Monitorear Logs de Build

Durante el build, deberías ver:

```
# Fase de Dependencias
📦 Instalando dependencias con NPM...
✅ XXX paquetes instalados

# Fase de Build
🏗 Building Next.js...
✅ Compiled successfully

# Fase de Runtime
🚀 Iniciando ESCALAFIN (versión mejorada)...
🔍 Verificando conexión a base de datos...
✅ DATABASE_URL configurada
🔄 Sincronizando esquema con base de datos...
✅ Esquema sincronizado exitosamente
🌱 Verificando necesidad de configurar usuarios...
👤 Usuarios en DB: X
🚀 INICIANDO SERVIDOR NEXT.JS
```

CHECKLIST DE VERIFICACIÓN

Durante el Build

- ☐ Build inicia sin errores
- ☐ Dependencias se instalan correctamente
- ☐ Next.js compila sin errores
- ☐ Build completa exitosamente
- ☐ Container inicia correctamente

Después del Build

- ☐ Aplicación responde en <https://demo.escalafin.com>
- ☐ Health check funciona: `/api/health`
- ☐ Login accesible: `/auth/login`
- ☐ Panel de admin accesible después de login

VERIFICACIÓN POST-DEPLOY

1. Health Check

```
curl https://demo.escalafin.com/api/health
```

Respuesta esperada:

```
{
  "status": "ok",
  "message": "EscalaFin API is running"
}
```

2. Verificar Login

URL: <https://demo.escalafin.com/auth/login>

Credenciales de prueba:

- Admin: admin@escalafin.com / admin123
- Asesor: asesor@escalafin.com / asesor123
- Cliente: cliente@escalafin.com / cliente123

3. Verificar Dashboard

Después del login exitoso:

- Admin debe redirigir a: </app/admin/dashboard>
- Asesor debe redirigir a: </app/asesor/dashboard>
- Cliente debe redirigir a: </app/cliente/dashboard>

4. Verificar Configuración de Chatwoot

URL: <https://demo.escalafin.com/app/admin/chatwoot>

Funcionalidad:

- ☐ Panel de configuración visible
- ☐ Campos para Website Token y Base URL
- ☐ Botón de activar/desactivar widget
- ☐ Botón de test de conexión

LOGS A REVISAR

Logs de Inicio (Application Logs)

Buscar estos indicadores:

- ✓ INDICADORES DE ÉXITO:
 - 🚀 Iniciando ESCALAFIN (versión mejorada)...
 - ✓ DATABASE_URL configurada
 - ✓ Esquema sincronizado exitosamente
 - ✓ DB ya inicializada con usuarios (o configurando usuarios)
 - 🚀 INICIANDO SERVIDOR NEXT.JS
- ✗ INDICADORES DE ERROR:
 - ✗ ERROR: prisma/schema.prisma no encontrado
 - ✗ ERROR: db push falló
 - ✗ DATABASE_URL no configurada
 - ✗ ERROR: server.js no encontrado

Logs de Build

- ✓ INDICADORES DE ÉXITO:
 - 📦 Instalando dependencias con NPM...
 - ✓ XXX paquetes instalados
 - 🔨 Building Next.js...
 - ✓ Compiled successfully
 - 📦 Copying standalone output
 - ✓ Build completed
- ✗ INDICADORES DE ERROR:
 - npm ERR!
 - Error: Cannot find module
 - Build failed
 - ENOENT: no such file or directory

TROUBLESHOOTING

Problema 1: “Build Fails - Cannot find module”

Síntomas:

```
npm ERR! Cannot find module '@prisma/client'
```

Solución:

1. Verificar que package-lock.json está en el repo
2. Limpiar cache de build
3. Rebuild

Problema 2: “Application not responding after build”

Síntomas:

- Build completa OK
- Container inicia
- Pero la aplicación no responde en el puerto

Solución:

1. Revisar logs de aplicación (Runtime Logs)
2. Verificar que DATABASE_URL está configurada
3. Verificar que no hay errores de Prisma
4. Verificar que el puerto es 3000 (default de Next.js)

Problema 3: “Database schema not synchronized”

Síntomas:

ERROR: db push falló

Solución:

1. Verificar que DATABASE_URL es correcta
2. Verificar que la base de datos está accesible desde el container
3. Verificar que las credenciales son correctas
4. Intentar conexión manual a la base de datos

Problema 4: “Users not created automatically”

Síntomas:

⚠️ setup-users-production.js no encontrado

Verificación:

```
# El archivo DEBE estar en GitHub en:
app/scripts/setup-users-production.js

# Si no está, verificar commit:
git log --oneline -10

# Debe aparecer commit e65a995 o c1dbdb4
```

Solución:

```
# Si el archivo no está en GitHub:
cd /home/ubuntu/escalafin_mvp
git add app/scripts/setup-users-production.js
git commit -m "fix: Add missing setup-users-production.js"
git push origin main

# Luego rebuild en EasyPanel
```



VARIABLES DE ENTORNO REQUERIDAS

Verificar en EasyPanel → Settings → Environment Variables:

Críticas (Obligatorias)

```
DATABASE_URL=postgresql://user:password@host:5432/dbname
NEXTAUTH_SECRET=<secret-generado-seguro>
NEXTAUTH_URL=https://demo.escalafin.com
```

Opcionales (Pero recomendadas)

```
# Openpay (Pagos)
OPENPAY_ID=<merchant-id>
OPENPAY_PRIVATE_KEY=<private-key>
OPENPAY_PUBLIC_KEY=<public-key>

# AWS S3 (Storage)
AWS_ACCESS_KEY_ID=<access-key>
AWS_SECRET_ACCESS_KEY=<secret-key>
AWS_BUCKET_NAME=<bucket-name>
AWS_REGION=us-east-1

# Evolution API (WhatsApp)
EVOLUTION_API_URL=<api-url>
EVOLUTION_API_KEY=<api-key>
```

Nuevas (Para Chatwoot)

```
# Estas se pueden configurar desde el panel de admin también
CHATWOOT_WEBSITE_TOKEN=<token>
CHATWOOT_BASE_URL=https://app.chatwoot.com
CHATWOOT_ENABLED=true
```

ORDEN DE EJECUCIÓN RECOMENDADO

1. **Verificar Variables de Entorno** ← Empezar aquí
 2. **Limpiar Cache de Build** ← Muy importante
 3. **Verificar Source Settings** (GitHub repo y branch)
 4. **Hacer Rebuild**
 5. **Monitorear Logs de Build**
 6. **Monitorear Logs de Runtime**
 7. **Verificar Health Check**
 8. **Verificar Login**
 9. **Verificar Dashboard**
 10. **Configurar Chatwoot** (si aplica)
-

SOPORTE

Si después de seguir estos pasos la aplicación no funciona:

1. Exportar logs completos:

- Build logs (completos)
- Runtime logs (últimas 200 líneas)

2. Verificar información:








- Commit de GitHub que está siendo usado
- Variables de entorno configuradas (sin valores sensibles)
- Errores específicos en logs

3. Compartir información:

- Proporcionar logs y configuración para diagnóstico

ÉXITO ESPERADO

Después de completar estos pasos exitosamente, deberías ver:

-  Build completo sin errores
-  Aplicación iniciada correctamente
-  Health check responde OK
-  Login funciona con usuarios de prueba
-  Dashboards accesibles según rol
-  Configuración de Chatwoot disponible
-  Toda la funcionalidad operativa

Última actualización: 29 de Octubre 2025, 07:30 UTC

Commit de GitHub: c1dbdb4

Features: Chatwoot configurable + Scripts mejorados + Usuarios automáticos