

Guía Visual de Configuración en Coolify

Capturas de Pantalla de Referencia

Según las imágenes proporcionadas, aquí está la guía paso a paso de cómo configurar EscalaFin en Coolify.

Paso 1: Dashboard de Coolify

Cuando accedas a <https://adm.escalafin.com>, verás el dashboard principal:

```
Coolify v4.0.0-beta.428
├── Dashboard
├── Projects ← EMPEZAR AQUÍ
├── Servers
├── Sources
├── Destinations
├── S3 Storages
├── Shared Variables
├── Notifications
├── Keys & Tokens
├── Tags
├── Terminal
├── Profile
├── Teams
└── Settings
```

Acción:

1. Click en **“Projects”** en el sidebar izquierdo
2. Verás “My first project” (puedes usarlo o crear uno nuevo)
3. Click en **“+ Add”** para crear nuevo proyecto

Paso 2: Crear Proyecto

Formulario de Proyecto:

```
Nombre: escalafin-mvp
Descripción: Sistema de gestión de préstamos y créditos
```

Click en **“Create”**

Paso 3: Conectar GitHub (Si no lo has hecho)

IMPORTANTE: Según tu captura, ya tienes Coolify pero necesitas conectar GitHub.

En la pantalla mostrada (dok.jpg):

1. Ve a la sección **"Github"** en el menú superior
2. Verás:
 - **Propietario:** qhosting ✓
 - **Repositorio:** escalafin-mvp ✓
 - **Rama:** main ✓
 - **Ruta de compilación:** / ← DEJAR VACÍO o poner /
3. ⚠ **CRÍTICO** - El campo vacío "Ruta de compilación":
NO LLENAR - Coolify lo detectará automáticamente

Si GitHub NO está conectado aún:

1. Click en **"Sources"** en sidebar
2. Click **" + Add Source "**
3. Selecciona **"Github"**
4. Autoriza Coolify en GitHub
5. Selecciona organización: qhosting
6. Selecciona repositorio: escalafin-mvp



Paso 4: Configurar Tipo de Build

Según captura dok2.jpg:

En la pantalla **"Compilación"**, verás 3 opciones:

- Dockerfile
Usa el comando "docker build" (docs)
- Buildpacks
Elija sus buildpacks deseados
- Nixpacks
Nueva forma de crear aplicaciones desde Railway (documentación)

Acción:

✓ **Seleccionar: "Dockerfile"**


Esto habilitará un campo donde debes especificar:

Dockerfile: Dockerfile.**production**

⚠ **MUY IMPORTANTE:** No uses Dockerfile o Dockerfile.coolify, usa específicamente **Docker-file.production**

Paso 5: Configurar Aplicación

Configuración Básica (en escal.jpg se ve el proyecto creado):

```
Name: escalafin_mvp
Type: Application
Status:  Not deployed yet
```

En la sección “General”:

```
Build Type: Dockerfile
Dockerfile Name: Dockerfile.production ← CRÍTICO
Branch: main
Build Directory: /
Port: 3000
```

En la sección “Build”:

```
Dockerfile: Dockerfile.production
Context: .
Build Arguments: (ninguno necesario)
```

Paso 6: Variables de Entorno

Click en “**Environment**” o “**Variables**” en el menú de tu app.

Agregar cada variable:

```
# Base de datos
DATABASE_URL=postgresql://escalafin:PASSWORD@db:5432/escalafin
POSTGRES_USER=escalafin
POSTGRES_PASSWORD=[GENERAR PASSWORD SEGURO]
POSTGRES_DB=escalafin
DATABASE_HOST=db
DATABASE_PORT=5432

# NextAuth
NEXTAUTH_URL=https://app.escalafin.com
NEXTAUTH_SECRET=[generar: openssl rand -base64 32]

# OpenPay
OPENPAY_MERCHANT_ID=
OPENPAY_PRIVATE_KEY=
OPENPAY_PUBLIC_KEY=
OPENPAY_BASE_URL=https://sandbox-api.openpay.mx

# AWS S3
AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_BUCKET_NAME=escalafin-files
AWS_REGION=us-east-1
AWS_FOLDER_PREFIX=production/

# Evolution API
EVOLUTION_API_URL=
EVOLUTION_API_TOKEN=
EVOLUTION_INSTANCE_NAME=escalafin

# Configuración de app
NODE_ENV=production
NEXT_TELEMETRY_DISABLED=1
PORT=3000
HOSTNAME=0.0.0.0
SKIP_MIGRATIONS=false
RUN_SEED=false
```

Tip: Usa el botón **" + Add "** para cada variable y marca las sensibles como **"Secret"** (🔒)

Paso 7: Crear Servicio PostgreSQL

En tu proyecto escalafin-mvp:

1. Click en **" + Servicio "** (botón verde en la barra superior)
2. Selecciona **"PostgreSQL"**

Configuración:

```
Name: escalafin-db
Version: 14
Username: escalafin
Password: [mismo que en POSTGRES_PASSWORD]
Database: escalafin
Port: 5432
```

Persistencia:

```
Volume Mount Path: /var/lib/postgresql/data
```

Click en **“Create”**

⚠ IMPORTANTE: Espera a que el servicio esté “Running” (verde) antes de desplegar la app.



Paso 8: Configurar Dominio y SSL

En la sección “Domains” de tu app:

1. Click en **“+ Add Domain”**
2. Ingresa: `app.escalafin.com`
3. Habilita:
 - ☒ Generate SSL Certificate (Let's Encrypt)
 - ☒ Force HTTPS
 - ☒ WWW Redirect (opcional)
4. Click en **“Save”**

Configurar DNS (en tu proveedor - Cloudflare, etc.):

```
Type: A
Name: app.escalafin
Value: [IP de tu servidor Coolify]
TTL: Auto
Proxy: ☒ (deshabilitado para primera configuración)
```

Tip: Para verificar la IP de tu servidor, ve a “Servers” en Coolify.



Paso 9: ¡Desplegar!

Botón de Deploy:

En la esquina superior derecha de tu aplicación, verás un botón grande:

Deploy

← Click aquí

Lo que sucederá:

1. **Pull del código** desde GitHub
2. **Build de la imagen** Docker (5-10 minutos)
3. **Push de la imagen** al registry
4. **Start del contenedor**
5. **Health checks**

Seguir el progreso:

- Ve a la pestaña **“Logs”**
- Verás el output en tiempo real
- Busca mensajes como:

☒ Build completed
☒ Container started
☒ Health check passed





👁️ Paso 10: Monitorear Deployment

Logs en tiempo real:

[Logs Tab]

- └─ Build Logs (construcción de imagen)
- └─ Application Logs (logs de tu app)
- └─ System Logs (logs del sistema)

Estados posibles:

-  **Building**: Construyendo imagen
-  **Starting**: Iniciando contenedor
-  **Running**: ¡Todo funciona!
-  **Failed**: Error (revisar logs)

Verificar que todo funciona:

```
# 1. Health check
curl https://app.escalafin.com/api/health

# 2. Verificar respuesta
# Debe retornar:
{
  "status": "healthy",
  "timestamp": "2025-10-01T...",
  "version": "1.0.0",
  "environment": "production",
  "database": "connected",
  "uptime": 123.45
}
```

Paso 11: Configurar Auto-Deploy (Opcional)

En la sección “General” de tu app:

- ☒ Automatic Deployment
Deploy automatically when code is pushed to main branch

Esto activará:

- Webhook en GitHub
- Deploy automático en cada `git push origin main`
- Sin intervención manual

Checklist Visual de Coolify

Use este checklist mientras configuras:

Preparación

- ☐ GitHub conectado en “Sources”
- ☐ Repositorio `ghosting/escalafin-mvp` visible
- ☐ Proyecto `escalafin-mvp` creado

Configuración de App

- ☐ Build type: **Dockerfile** ☒
- ☐ Dockerfile name: **Dockerfile.production** ☒
- ☐ Branch: **main** ☒
- ☐ Port: **3000** ☒
- ☐ Variables de entorno configuradas (ver lista)

Servicios

- ☐ PostgreSQL creado y **Running** (●)
- ☐ Conexión DB verificada

Dominio

- ☐ Dominio agregado: `app.escalafin.com`
- ☐ DNS configurado (A record)
- ☐ SSL habilitado (Let's Encrypt)

Deployment

- ☐ Primer deploy completado
- ☐ Health check pasa (●)
- ☐ App accesible en el dominio
- ☐ Login/registro funcionan



Troubleshooting Visual

Si ves “Error de validación” (como en dok.jpg):

“Ruta de compilación: Required”

Solución:

- En “Ruta de compilación”, pon: `/`
- O déjalo vacío (Coolify lo detectará)

Si el build falla:

1. **Ve a “Logs”** → “Build Logs”

2. Busca errores como:

`ERROR: Could not find Dockerfile`

Solución: Verifica que pusiste `Dockerfile.production`

3. Si ves:

`ERROR: failed to solve: failed to read dockerfile`

Solución: El archivo no está en el repo. Ejecuta `./deploy-to-github.sh`

Si el contenedor no inicia:

1. **Ve a “Logs”** → “Application Logs”

2. Busca errores de env vars:

`Missing environment variables: NEXTAUTH_SECRET`

Solución: Agrega la variable faltante en “Environment”

3. Si ves error de base de datos:

`Database connection refused`

Solución: Verifica que PostgreSQL esté “Running” (●)




Dashboard después de Deploy Exitoso

Tu dashboard de Coolify mostrará:

escalafin_mvp

APP

Status:  Running

Uptime: 2h 34m

CPU: 12%

Memory: 256MB / 2GB

Domain: app.escalafin.com

[Restart]


[Stop]

[Logs]

[Settings]

escalafin-db

DB

Status:  Running

Type: PostgreSQL 14

Port: 5432

Size: 45MB

[Backup]

[Logs]

[Settings]

✓ Verificación Final

Test desde tu navegador:

1. Ve a: <https://app.escalafin.com>
2. Deberías ver la página de login de EscalaFin
3. Crea una cuenta de prueba
4. Verifica que puedes acceder al dashboard

Test desde terminal:

```
# Health check
curl -I https://app.escalafin.com/api/health

# Debe retornar:
HTTP/2 200
content-type: application/json
...
```

🎓 Tips Pro

1. Ver logs en tiempo real desde UI:

- Click en tu app → “Logs”
- Toggle “Auto-scroll” para seguir el output
- Usa filtros para buscar errores específicos

2. Rollback rápido:

- Si algo falla, ve a “Deployments”
- Click en el deployment anterior que funcionaba
- Click en “Redeploy”

3. Configurar notificaciones:

- Ve a “Notifications” en sidebar
- Agrega tu email o webhook
- Recibe alertas de:
 - Deploy exitoso
 - Deploy fallido
 - Downtime detectado

4. Backups automáticos:

- Click en tu servicio PostgreSQL
- Ve a “Backups”
- Configura:

Frequency: Daily

Time: 02:00 AM

Retention: 7 days



Recursos Adicionales

- **Coolify Docs:** <https://coolify.io/docs>
 - **Guía completa:** `EASYPANEL-COMPLETE-GUIDE.md`
 - **Troubleshooting:** Sección de troubleshooting en la guía completa
-



¡Listo!

Si todo salió bien, ahora tienes:

- ✓ **EscalaFin desplegado** en Coolify
- ✓ **SSL configurado** automáticamente
- ✓ **Base de datos** funcionando
- ✓ **Auto-deploy** desde GitHub
- ✓ **Monitoring** activo
- ✓ **Backups** configurados

¡Tu aplicación está en producción! 🚀

Última actualización: Octubre 1, 2025

Basado en: Coolify v4.0.0-beta.428

Screenshots: dok.jpg, dok2.jpg, escal.jpg