



CONFIGURACIÓN VISUAL PARA EASYPANEL



Basado en tus capturas de pantalla

He analizado las imágenes que subiste (dok.jpg , dok2.jpg , escal.jpg) y aquí está la configuración EXACTA que debes usar.



IMAGEN 1: Configuración de GitHub (dok.jpg)



PROBLEMA DETECTADO:

En la captura se ve **“Error de validación”** y el campo **“Ruta de compilación”** está vacío.



SOLUCIÓN:

Pestaña: GitHub

Propietario *	
ghosting	
Nombre de usuario u organización de Github	
Repositorio *	
escalafin-mvp	
Nombre del repositorio de Github	
github.com / ghosting / escalafin-mvp	
Rama *	
main	
Esta debe ser una rama válida	
Ruta de compilación *	
/app	⚠ AQUÍ!
Required	

[Guardar]

⚠ **CRÍTICO:** El campo “Ruta de compilación” NO puede estar vacío. Debe ser: `/app`

📄 IMAGEN 2: Selección de Método de Compilación (dok2.jpg)

Opciones Disponibles:

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

✅ SELECCIONAR:

☒ Dockerfile ← ESTA OPCIÓN

Después de seleccionar Dockerfile, configurar:

Dockerfile to use

Dockerfile.step3-full

Build context

.

📄 IMAGEN 3: Panel de Coolify (escal.jpg)

Esta imagen muestra Coolify, pero has dicho que **NO usarás Coolify**, así que **IGNORAR**.

🔧 CONFIGURACIÓN COMPLETA PASO A PASO

PASO 1: Crear PostgreSQL

1. En EasyPanel, ir a “**Services**” → “+ **Add Service**”

2. Seleccionar **“PostgreSQL”**

3. Configurar:

```
Name: escalafin-db
Version: 16 (latest)
Database Name: escalafin
Username: escalafin_user
Password: [Generar password seguro]
```

1. Click **“Create”**

2. **ESPERAR** a que el servicio esté “Running” (1-2 minutos)

PASO 2: Anotar DATABASE_URL

Después de crear PostgreSQL, EasyPanel te dará el connection string:

```
postgresql://escalafin_user:TU_PASSWORD@escalafin-db:5432/escalafin?schema=public
```


COPIAR ESTO - lo necesitarás en el siguiente paso.

PASO 3: Crear App desde GitHub

1. En EasyPanel, ir a **“Apps”** → **“+ Add App”**

2. Seleccionar **“GitHub”**

Pestaña: GitHub

```
Propietario: qhosting
Repositorio: escalafin-mvp
Rama: main
Ruta de compilación: /app  NO OLVIDAR
```

Pestaña: Build

```
Método: Dockerfile (seleccionar el radio button)
Dockerfile: Dockerfile.step3-full
Build Context: .
```

Pestaña: Environment

Agregar estas variables (click **“+ Add Variable”** para cada una):

```
# === DATABASE ===
DATABASE_URL=postgresql://escalafin_user:TU_PASSWORD@escalafin-db:5432/escalafin?
schema=public

# === NEXTAUTH ===
NEXTAUTH_URL=https://escalafin-mvp.tu-dominio.com
NEXTAUTH_SECRET=[generar con: openssl rand -base64 32]

# === AWS S3 ===
AWS_BUCKET_NAME=tu-bucket-s3
AWS_FOLDER_PREFIX=escalafin/
AWS_REGION=us-east-1
AWS_ACCESS_KEY_ID=tu-access-key
AWS_SECRET_ACCESS_KEY=tu-secret-key

# === OPENPAY ===
OPENPAY_MERCHANT_ID=tu-merchant-id
OPENPAY_PRIVATE_KEY=tu-private-key
OPENPAY_PUBLIC_KEY=tu-public-key
OPENPAY_API_ENDPOINT=https://sandbox-api.openpay.mx
OPENPAY_IS_SANDBOX=true

# === EVOLUTIONAPI ===
EVOLUTION_API_URL=tu-url-evolutionapi
EVOLUTION_API_KEY=tu-api-key
EVOLUTION_API_INSTANCE=tu-instance

# === BUILD ===
NODE_ENV=production
NEXT_OUTPUT_MODE=standalone
SKIP_ENV_VALIDATION=1
NEXT_TELEMETRY_DISABLED=1
PORT=3000
```

Pestaña: Domain

Custom Domain: escalafin-mvp.tu-dominio.com

Pestaña: Resources

CPU: 1-2 cores
Memory: 2GB minimum (4GB recomendado)
Replicas: 1

PASO 4: Deploy

1. Click **“Create App”**
2. EasyPanel comenzará el build automáticamente
3. Ir a **“Logs”** para monitorear el progreso

LOGS QUE DEBES VER (Éxito)

```
Step 1/25 : FROM node:18-alpine AS base
--> [image id]
Step 2/25 : RUN apk add --no-cache...
--> [image id]

...
=== 📦 Instalando dependencias ===
✅ Dependencias instaladas

...
=== 🛠 Generando Prisma Client ===
✅ Prisma Client generado

...
=== 🏗 Building Next.js ===
✅ Build completado
✅ Standalone verificado

...
Successfully built [image id]
Successfully tagged escalafin-mvp:latest
```

Y después del deploy:

```
✅ Esperando PostgreSQL...
PostgreSQL está disponible
✅ Aplicando migraciones Prisma...
✅ Migraciones aplicadas exitosamente
✅ Ejecutando seed inicial...
✅ Seed completado
▲ Next.js 14.x.x
- Local:      http://0.0.0.0:3000
✓ Ready in XXXms
```

ERRORES COMUNES Y SOLUCIONES

❌ “Build Path required”

Causa: Campo “Ruta de compilación” vacío (como en tu captura dok.jpg)

Solución: Poner `/app`

❌ “Dockerfile not found”

Causa: Dockerfile especificado incorrectamente

Solución: Usar `Dockerfile.step3-full` (no solo “Dockerfile”)

❌ “Cannot connect to database”

Causa: PostgreSQL no está running o DATABASE_URL incorrecta

Solución:

1. Verificar que escalafin-db está “Running”
2. Verificar DATABASE_URL tiene el formato correcto
3. Esperar 1-2 minutos y reintentar

✗ “NEXTAUTH_SECRET is not set”

Causa: Variable de entorno faltante

Solución: Generar con `openssl rand -base64 32` y agregar

✗ “server.js not found”

Causa: Standalone build no se generó

Solución: Verificar que NEXT_OUTPUT_MODE=standalone está configurado

✓ CHECKLIST VISUAL

Antes de hacer click en “Create App”, verificar:

- ☒ PostgreSQL "escalafin-db" está Running
- ☒ DATABASE_URL copiado y pegado en Environment
- ☒ NEXTAUTH_SECRET generado y configurado
- ☒ Campo "Ruta de compilación": /app (NO VACÍO)
- ☒ Método de compilación: Dockerfile (radio button seleccionado)
- ☒ Dockerfile especificado: Dockerfile.step3-full
- ☒ Todas las variables de entorno configuradas
- ☒ Dominio configurado
- ☒ Recursos asignados (mínimo 2GB RAM)

🎯 FLUJO COMPLETO

1. Crear PostgreSQL
↓
2. Esperar a que esté Running
↓
3. Copiar DATABASE_URL
↓
4. Crear App desde GitHub
↓
5. Configurar:
 - GitHub: qhosting/escalafin-mvp, main, /app
 - Build: Dockerfile, Dockerfile.step3-full
 - Environment: Todas las variables
 ↓
6. Create App
↓
7. Monitorear Logs
↓
8. Success

📞 SI ALGO FALLA

NO TOCAR NADA MÁS.

Tomar captura de pantalla de:

1. Configuración completa (GitHub tab)
2. Logs del build
3. Variables de entorno

Y mostrar las capturas para identificar el problema exacto.

¡Sigue estos pasos EXACTAMENTE y el deploy funcionará! 🚀