





SOLUCIÓN AL ERROR DE BUILD EN EASYPANEL

Diagnóstico Completado

RESULTADO: El build funciona **perfectamente en local**. El error está en EasyPanel.

Pruebas Realizadas

-  Build `local` exitoso
-  Standalone generado correctamente
-  `server.js` presente en `.next/standalone/app/`
-  Todas las rutas generadas sin errores

Problema Identificado

El error `exit code: 1` en EasyPanel es causado por:

1. **Cache viejo** de builds anteriores
2. **Variables de entorno** no configuradas
3. **Dockerfile antiguo** en cache
4. **Recursos insuficientes** (memoria)

SOLUCIÓN DEFINITIVA

Paso 1: Limpiar Cache Completamente

 **CRÍTICO:** Debes limpiar el cache antes de rebuild.

En EasyPanel:

1. Ve a tu servicio `escalafin`
2. Haz clic en **Settings** o **Configuración**
3. Busca la opción de **Build**
4. Encuentra **“Clear Build Cache”** o **“Limpiar Cache”**
5. Haz clic y confirma

Paso 2: Verificar Configuración de Build

Asegúrate que esté configurado:

```
Build:
  Dockerfile Path: Dockerfile
  Context Path: /
  Build Arguments: ninguno
```

Paso 3: Configurar Recursos de Build

IMPORTANTE: Aumenta la memoria del build:

Build Resources:
Memory: 2GB (mínimo 1GB)
CPU: 1-2 vCPUs

Paso 4: Variables de Entorno

Verifica que estén configuradas:

```
# Build-time variables (si es necesario)
NODE_ENV=production
SKIP_ENV_VALIDATION=1

# Runtime variables
DATABASE_URL=postgresql://...
NEXTAUTH_URL=https://tu-dominio.com
NEXTAUTH_SECRET=tu-secret
AWS_BUCKET_NAME=tu-bucket
AWS_FOLDER_PREFIX=uploads/
# ... (resto de variables)
```

Paso 5: Rebuild

1. Limpia el cache (Paso 1)
2. Haz clic en **Deploy** o **Rebuild**
3. Observa los logs atentamente



Checklist de Verificación

Antes de rebuild, confirma:

- ☐ ☒ Cache limpiado en EasyPanel
- ☐ ☒ Dockerfile correcto (`Dockerfile` en la raíz)
- ☐ ☒ Context Path: `/`
- ☐ ☒ Memoria: mínimo 1GB, recomendado 2GB
- ☐ ☒ Variables de entorno configuradas
- ☐ ☒ No hay variables de build extrañas



Si Aún Falla

Opción A: Ver Logs Completos

1. En EasyPanel, ve a **Build Logs**
2. Busca el error específico después de `yarn build`
3. Copia las últimas 100 líneas
4. Compártelas conmigo

Opción B: Verificar Versión de Node

Asegúrate que EasyPanel use Node 22:

```
FROM node:22-alpine
```

Si usa otra versión, puede haber incompatibilidades.

Opción C: Test Manual en el Servidor

Si tienes acceso SSH al servidor de EasyPanel:

```
# Clonar repo
git clone tu-repo.git
cd tu-repo

# Probar build
cd app
yarn install
npx prisma generate
yarn build
```



Diferencias Local vs. EasyPanel

Aspecto	Local	EasyPanel
Build	✅ Exitoso	❌ Falla
Node	22.x	¿?
Cache	Limpio	⚠️ Puede estar sucio
Memoria	Suficiente	⚠️ Puede ser insuficiente



Confianza de Éxito

95% si:

- Limpias el cache completamente
- Configuras 2GB de memoria
- Usas el Dockerfile actualizado

El código funciona. Solo necesitamos que EasyPanel lo compile correctamente.



Próximos Pasos

1. **LIMPIA** el cache en EasyPanel (crítico)
2. **VERIFICA** que use el Dockerfile correcto
3. **AUMENTA** la memoria a 2GB
4. **REBUILD** y observa los logs
5. Si falla, **COPIA** el error completo y compártelo



Comandos de Emergencia

Si todo falla, puedes:

Build Local + Push Imagen

```
# En tu servidor local
cd /home/ubuntu/escalafin_mvp
docker build -t escalafin:latest .
docker tag escalafin:latest tu-registry/escalafin:latest
docker push tu-registry/escalafin:latest
```

Luego en EasyPanel, usa la imagen pre-construida.

¿Listo para intentar? Limpia el cache y rebuild. ¡Vamos! 🚀