

Dockerfile v8.10 - Captura de Output Completo

Problema en v8.9

El build estaba fallando con **exit code 1**, pero los logs del error **NO se estaban mostrando** completamente en EasyPanel.

```
ERROR: exit code: 1
```

Qué faltaba: Ver el error REAL del build de Next.js.

Solución en v8.10

Script Simplificado con Output Completo

```
RUN echo "=== INICIANDO BUILD NEXT.JS ===" && \
  npm run build 2>&1 | tee /tmp/build-output.log; \
  if [ ! -f .next/BUILD_ID ]; then \
    echo "===== " && \
    echo "❌ BUILD DE NEXT.JS FALLÓ" && \
    echo "===== " && \
    echo "=== OUTPUT COMPLETO DEL BUILD ===" && \
    cat /tmp/build-output.log && \
    echo "=== ESTRUCTURA DE ARCHIVOS ===" && \
    ls -la && \
    echo "=== PRISMA CLIENT ===" && \
    ls -la node_modules/@prisma/client 2>/dev/null || echo "❌ @prisma/client NO ENCONTRADO" && \
    echo "=== CONTENIDO .next ===" && \
    ls -laR .next/ 2>/dev/null || echo "❌ .next NO EXISTE" && \
    exit 1; \
  fi && \
  echo "✅ Build de Next.js completado exitosamente"
```




Cambios Principales

v8.9 → v8.10




Aspecto	v8.9	v8.10
Captura	<code>tail -100</code>	<code>cat</code> (todo)
Output	Últimas 100 líneas	Completo
Lógica	<code>set -o pipefail + OR</code>	Simple if/else
Separadores	Básicos	Visuales claros
Claridad	Media	Alta

Por Qué “cat” en Lugar de “tail”

tail -100:

-  Muestra últimas 100 líneas
-  Puede perder contexto importante
-  Si el error está en línea 50, no lo vemos

cat:

-  Muestra TODO el output
 -  No pierde nada
 -  Vemos todo desde el inicio del build
-



Output Esperado

Si el Build Falla

```

=== INICIANDO BUILD NEXT.JS ===

> app@0.1.0 build
> next build

Creating an optimized production build...
[... todo el proceso de build ...]

Failed to compile.

./app/lib/db.ts:1:0
Module not found: Can't resolve '@prisma/client'

   1 | import { PrismaClient } from '@prisma/client'
     |                               ^
   2 |
   3 | const prisma = new PrismaClient()

=====
❌ BUILD DE NEXT.JS FALLÓ
=====

=== OUTPUT COMPLETO DEL BUILD ===
[... TODO el output desde el inicio ...]
Failed to compile.
./app/lib/db.ts:1:0
Module not found: Can't resolve '@prisma/client'
[... stack trace completo ...]

=== ESTRUCTURA DE ARCHIVOS ===
total 248
drwxr-xr-x  6 root  root      4096 Oct  8 02:55 .
drwxr-xr-x 18 root  root      4096 Oct  8 02:50 ..
drwxr-xr-x  8 root  root      4096 Oct  8 02:53 app
drwxr-xr-x 523 root  root    20480 Oct  8 02:51 node_modules
-rw-r--r--  1 root  root     1234 Oct  8 02:50 package.json
drwxr-xr-x  2 root  root      4096 Oct  8 02:50 prisma

=== PRISMA CLIENT ===
drwxr-xr-x  4 root  root      4096 Oct  8 02:51 .
drwxr-xr-x 523 root  root    20480 Oct  8 02:51 ..
drwxr-xr-x  2 root  root      4096 Oct  8 02:51 runtime
-rw-r--r--  1 root  root     12345 Oct  8 02:51 index.js
[... o ...]
❌ @prisma/client NO ENCONTRADO

=== CONTENIDO .next ===
❌ .next NO EXISTE

```



Lo Que Podremos Ver

Con v8.10 obtendremos información completa sobre:

1. Error Exacto del Build

```
Failed to compile.

./app/lib/db.ts:1:0
Module not found: Can't resolve '@prisma/client'
```

→ **Identificamos:** Falta @prisma/client

2. Stack Trace Completo

```
> 1 | import { PrismaClient } from '@prisma/client'
    |                               ^
    2 |
    3 | const prisma = new PrismaClient()

Error: Cannot find module '@prisma/client'
```

→ **Identificamos:** Exactamente dónde falla

3. Estado de Prisma

```
=== PRISMA CLIENT ===
drwxr-xr-x @prisma/client/
```

→ **Confirmamos:** Si Prisma Client fue generado

4. Estructura de .next

```
=== CONTENIDO .next ===
.next/
  cache/
  BUILD_ID (NO EXISTE)
```

→ **Confirmamos:** Qué partes del build se completaron

Posibles Errores y Diagnóstico

Error 1: @prisma/client No Encontrado

Logs mostrarán:

```
Module not found: Can't resolve '@prisma/client'
...
❌ @prisma/client NO ENCONTRADO
```

Causa: Prisma generate falló o no se ejecutó

Fix v8.11:

- Verificar stage de Prisma

- Asegurar que generate se complete
- Verificar que client esté en node_modules

Error 2: Error de TypeScript

Logs mostrarán:

```
Type error: Property 'xyz' does not exist on type 'User'
at app/components/UserCard.tsx:15:5
```

Causa: Error de tipos en el código

Fix v8.11:

- Corregir tipos
- O agregar `ignoreBuildErrors: true` temporalmente

Error 3: Variable de Entorno Requerida

Logs mostrarán:

```
Error: DATABASE_URL is required for build
```

Causa: Next.js requiere variable en build time

Fix v8.11:

- Verificar ENV vars en Dockerfile
- Ajustar next.config.js

Error 4: Dependencia Faltante

Logs mostrarán:

```
Cannot find module 'some-package'
```








Causa: Package no instalado

Fix v8.11:

- Agregar a package.json
- Rebuild

Información Que Veremos

v8.10 proporciona información completa para diagnóstico:

1.  **Output completo del build** (todo, no solo últimas líneas)
2.  **Mensaje de error exacto**
3.  **Stack trace completo**
4.  **Archivo y línea del error**
5.  **Estructura de archivos** (verificar que todo existe)
6.  **Estado de Prisma Client** (generado o no)
7.  **Contenido de .next** (qué se generó parcialmente)

Por Qué Este Enfoque

Simplicidad

```
# Más simple, más claro
if [ ! -f .next/BUILD_ID ]; then
  cat /tmp/build-output.log
  exit 1
fi
```

vs v8.9:

```
# Más complejo
set -o pipefail && ... || (...) && ... || (...)
```

Completitud

cat muestra TODO el output, sin truncar.

Debugging

Separadores claros (`=====`) hacen fácil encontrar cada sección.

Próximos Pasos

1. Rebuild con v8.10

Commit: 0135929

Versión: 8.10 (full output)

2. Copiar TODO el Output

CRÍTICO: Cuando el build falle, busca en los logs:

```
=====
X BUILD DE NEXT.JS FALLÓ
=====
```

Y copia TODO desde ahí hasta el final, incluyendo:

- Output completo del build
- Estructura de archivos
- Estado de Prisma
- Contenido .next

3. Con Ese Output Completo

1. ☒ Identificaré el error exacto
2. ☒ Determinaré la causa raíz
3. ☒ Aplicaré fix específico en v8.11
4. ☒ Build completará exitosamente

✓ Mejoras en v8.10

Técnicas

- ✓ Script más simple y directo
- ✓ `cat` en lugar de `tail` (output completo)
- ✓ Lógica if/else clara
- ✓ Separadores visuales

Funcionales

- ✓ Muestra TODO el error
- ✓ No pierde información
- ✓ Más fácil de debuggear
- ✓ Output más organizado

Diagnóstico

- ✓ Error exacto visible
- ✓ Stack trace completo
- ✓ Contexto de archivos
- ✓ Estado de dependencias

Conclusión

v8.10 está diseñado para:

- ✓ Mostrar el error COMPLETO
- ✓ No perder ninguna línea importante
- ✓ Facilitar diagnóstico preciso
- ✓ Aplicar fix correcto en la próxima versión

Con el output completo de v8.10, podré:

1. Ver exactamente qué está fallando
2. Identificar la causa raíz
3. Aplicar el fix específico
4. Resolver el problema definitivamente

Versión: 8.10

Fecha: 2025-10-08 02:55 GMT

Estado: 🔍 FULL OUTPUT CAPTURE

Cambio principal:

- cat en lugar de tail
- Output completo del build
- Separadores visuales claros
- Diagnóstico más fácil

Objetivo: Ver el error COMPLETO de Next.js sin perder información.

¡Rebuild y copia TODO el bloque de error! 🚀