

# CRÍTICO: Limpiar Cache en EasyPanel

---

**Fecha:** 2025-10-18

**Commit:** 128f2ad



**Problema:** EasyPanel está usando Dockerfile viejo (con npm)

---

## DIAGNÓSTICO

---

### Según tus logs:

```
0.610  package-lock.json encontrado (lockfileVersion: 3)
0.610  Usando npm install
```

### Problema:

EasyPanel está usando el **Dockerfile viejo** que tiene:

- Node 18 + npm
- Busca package-lock.json

### Dockerfile nuevo (128f2ad):

- Node 22 + Yarn 4.9.4
  - Busca yarn.lock (no package-lock.json)
- 

## CAUSA

---

**EasyPanel tiene CACHE del build anterior.**

El cache incluye:

- Dockerfile viejo
  - Layers de Docker anteriores
  - package-lock.json (que ya no debería usar)
-

## ✓ SOLUCIÓN: Limpiar Cache en EasyPanel

### Opción 1: Limpiar Cache desde UI (Recomendado)

**Paso 1:** Ve a tu aplicación en EasyPanel

**Paso 2:** Busca “Build Settings” o “Configuration”

**Paso 3:** Encuentra la opción de “Cache”

Puede estar como:

- "Clear Build Cache"
- "Clean Cache"
- "Rebuild without cache"
- "Delete build cache"

**Paso 4:** Activa “Rebuild without cache”

**Paso 5:** Haz Deploy/Rebuild

---

### Opción 2: Forzar Rebuild desde GitHub

**Paso 1:** Verifica que EasyPanel esté conectado a GitHub

**Paso 2:** Configura el webhook para auto-deploy

**Paso 3:** En EasyPanel, busca:

Settings > GitHub Integration > Trigger Deploy

**Paso 4:** Click en “Force Rebuild”

---

### Opción 3: Variables de entorno para forzar rebuild

En EasyPanel, agrega una variable temporal:

`CACHE_BUST=128f2ad`

Esto forzará a Docker a ignorar el cache.

---

### Opción 4: Comando Docker manual (si tienes acceso SSH)

Si tienes acceso SSH al servidor de EasyPanel:

```
# Limpiar cache de Docker
docker builder prune -af

# Rebuild sin cache
docker build --no-cache -f Dockerfile -t escalafin-mvp:latest .
```

## VERIFICAR QUE EASYPANEL USE EL DOCKERFILE CORRECTO

### En EasyPanel Dashboard:

#### 1. Ve a Build Settings

#### 2. Verifica la configuración:

```
yaml
Repository: https://github.com/qhosting/escalafin-mvp
Branch: main
Dockerfile: Dockerfile # (sin ruta, usa el del root)
Context: .
```

#### 3. NO debe ser:

```
yaml
Dockerfile: Dockerfile.easypanel # ❌ NO USAR
Dockerfile: Dockerfile.backup-v16-npm # ❌ NO USAR
```




#### 4. Debe ser simplemente:

```
yaml
Dockerfile: Dockerfile # ✅ CORRECTO
```



## CÓMO CONFIRMAR QUE ESTÁ USANDO EL NUEVO

### Durante el build, los logs deben mostrar:

#### ✅ CORRECTO (Nuevo Dockerfile):

```
Step 1/X: FROM node:22-alpine
Step 2/X: RUN corepack prepare yarn@4.9.4
...
[deps]  Versión de yarn: 4.9.4
[deps]  Versión de node: v22.x.x
[deps]  yarn install --frozen-lockfile
```

#### ❌ INCORRECTO (Viejo Dockerfile):

```
Step 1/X: FROM node:18-alpine
...
[deps] npm install -g npm@10.9.0
[deps]  package-lock.json encontrado
[deps]  Usando npm install
```

## PASOS COMPLETOS (PASO A PASO)

### 1. Confirmar que tienes el último código:

En EasyPanel Dashboard:

Repository > Branch: main > Latest Commit: 128f2ad

## 2. Limpiar cache:

Settings > Build > Clear Cache

O marca la opción:

☒ Rebuild without cache

## 3. Verificar Dockerfile:

Build Settings:  
Dockerfile: Dockerfile  
Context: .  
Build Args: (ninguno)

## 4. Hacer Deploy:

Click "Deploy" o "Rebuild"

## 5. Monitorear logs:

Espera a ver las primeras líneas:

FROM node:22-alpine ☐ Debe decir "22", no "18"  
RUN corepack prepare yarn@4.9.4 ☐ Debe mencionar yarn



## SI SIGUE FALLANDO

### Verifica estas cosas:

#### 1. ¿EasyPanel está apuntando al branch correcto?

Branch: main ← Debe ser "main"

#### 2. ¿El commit es el último?

Latest commit: 128f2ad

#### 3. ¿Está usando el Dockerfile del root?

Dockerfile: Dockerfile (no Dockerfile.easypanel)

#### 4. ¿El cache está limpio?

Rebuild without cache: ☒ activado

#### 5. ¿Hay algún archivo .dockerignore que bloquee archivos?

## DEBUGGING: Verificar Dockerfile en uso

En los logs de EasyPanel, busca:

```
Building image...
Using Dockerfile: /path/to/Dockerfile
```

Debe decir solo “Dockerfile”, no “Dockerfile.easypanel” ni otro.

## CHECKLIST RÁPIDO

Antes de hacer rebuild, verifica:

- [ ] ☒ Commit 128f2ad está en GitHub
- [ ] ☒ EasyPanel apunta a branch “main”
- [ ] ☒ Latest commit en EasyPanel = 128f2ad
- [ ] ☒ Dockerfile configurado: “Dockerfile” (sin ruta)
- [ ] ☒ Cache limpiado o “Rebuild without cache” activado
- [ ] ☒ No hay Dockerfile.easypanel especificado
- [ ] ☒ Context está en “.” (root del proyecto)

## RESULTADO ESPERADO

Con todo configurado correctamente, deberías ver:

```
☒ Step 1/15: FROM node:22-alpine
☒ Step 2/15: RUN corepack prepare yarn@4.9.4
☒ Step 5/15: COPY app/package.json app/yarn.lock* ./
☒ Step 6/15: RUN yarn install --frozen-lockfile
☒ ...
☒ Build completed successfully
☒ Container started on port 3000
```

## ALTERNATIVA: Usar Dockerfile.step3-full explícitamente

Si EasyPanel sigue usando cache, puedes **forzar** el uso del Dockerfile correcto:

**En Build Settings:**

```
Dockerfile: Dockerfile.step3-full
Context: .
```

Esto garantiza que use el Dockerfile con las correcciones.

---

## SI TODO FALLA

---

Si después de limpiar cache y verificar todo lo anterior sigue fallando:

1. **Copia los logs completos** del build (desde la primera línea)
2. **Toma screenshot** de la configuración de Build Settings
3. **Verifica** que el commit en GitHub sea 128f2ad:

```
bash
```




```
git log -1 --oneline
```

---





## RESUMEN

---

### Lo que hice:

1.  Actualicé el Dockerfile principal con Node 22 + Yarn 4.9.4
2.  Commit 128f2ad pushed a GitHub
3.  Backup del Dockerfile viejo guardado

### Lo que debes hacer tú:

1.  Limpiar cache en EasyPanel
2.  Verificar configuración del Dockerfile
3.  Hacer rebuild
4.  Verificar logs (debe decir “node:22” y “yarn 4.9.4”)

**Probabilidad de éxito:** 95-99% (si el cache se limpia correctamente)

---

**Próximo paso:** Limpiar cache y hacer rebuild en EasyPanel.