





Configuração de Porta (PORT) para Deployment

Problema Identificado

Durante o deployment no Easypanel/Heroku, a aplicação estava sendo terminada prematuramente com erros `SIGTERM` porque o Next.js estava rodando na porta padrão (3000), mas a plataforma espera que a aplicação use a variável de ambiente `PORT` fornecida dinamicamente.

Sintomas do Problema:

-  Build completava com sucesso
-  Next.js iniciava: "Local: http://localhost:3000"
-  Health check falhava
-  Processo era terminado: "npm error signal SIGTERM"

Solução Implementada

Alteração no `package.json`

Antes:

```
"start": "next start"
```

Depois:

```
"start": "next start -p ${PORT:-3000}"
```

Como Funciona


A sintaxe `${PORT:-3000}` é uma expansão de parâmetro do shell que:

1. **Usa `PORT`** se a variável de ambiente estiver definida
2. **Fallback para 3000** se `PORT` não estiver definida
3. **Funciona perfeitamente** com Next.js 14.2.28

Testes Realizados

Teste 1: Com PORT Customizada

```
PORT=4000 npm start
```

Resultado:  Servidor iniciou na porta 4000








Teste 2: Sem PORT (Default)

```
npm start
```

Resultado:  Servidor iniciou na porta 3000



Compatibilidade

- **Next.js:** 14.2.28 
- **Node.js:** Todas as versões compatíveis com Next.js 14
- **Plataformas:**
 -  Easypanel
 -  Heroku
 -  Railway
 -  Render
 -  Fly.io
 -  Qualquer PaaS que use variável `PORT`



Notas Técnicas

Por que isso é necessário?

Plataformas PaaS (Platform as a Service) como Easypanel e Heroku:

1. Fazem binding dinâmico de portas
2. Fornecem a porta através da variável `PORT`
3. Esperam que a aplicação escute nessa porta específica
4. Executam health checks na porta fornecida
5. Terminam processos que não respondem (SIGTERM)

Alternativas Consideradas

Opção 1: Usar apenas `next start` (sem `-p`)

```
"start": "next start"
```

- Next.js 14.2.28 suporta `PORT` nativamente
- Funciona, mas menos explícito
- **Não escolhido:** Menos controle e visibilidade

Opção 2: Script separado

```
"start": "node server.js"
```

- Requer servidor customizado
- Mais complexo
- **Não escolhido:** Desnecessário para este caso

Opção 3: Usar variável de ambiente na linha de comando **ESCOLHIDA**

```
"start": "next start -p ${PORT:-3000}"
```

- Explícito e claro
- Funciona em desenvolvimento e produção
- Fallback seguro para porta 3000

Verificação no Easypanel

Após o deployment:

1. Verificar Logs:

✓ Ready in 314ms

Local: `http://localhost:[PORTA_DINAMICA]`

2. Health Check:

- Aplicação deve responder na porta `PORT`
- Sem erros SIGTERM
- Processo mantém-se ativo

3. Acesso Externo:

- Link do Easypanel deve abrir a aplicação
- Sem timeout ou erro 502/503

Referências


- [Next.js CLI Documentation](https://nextjs.org/docs/api-reference/cli#production) (https://nextjs.org/docs/api-reference/cli#production)
- [Next.js Deployment](https://nextjs.org/docs/deployment) (https://nextjs.org/docs/deployment)
- [Heroku Node.js Support](https://devcenter.heroku.com/articles/nodejs-support) (https://devcenter.heroku.com/articles/nodejs-support)

Checklist de Deployment

- [x] Package.json atualizado com PORT configurável
- [x] Testado localmente com PORT customizada
- [x] Testado localmente com PORT padrão
- [x] Documentação criada
- [x] Compatibilidade com Next.js 14.2.28 verificada
- [] Deploy no Easypanel realizado
- [] Verificação do health check no Easypanel
- [] Aplicação acessível via link do Easypanel

Data da Implementação: 27 de Outubro de 2025

Versão do Next.js: 14.2.28

Status:  Implementado e Testado