

# Markdown PDF

## Arquitetura Docker para Sistema PACS - BitPacs

### Esclarecendo a Arquitetura

CI/CD lento NÃO afeta carregamento de imagens

O CI/CD lento é apenas no momento do **build/deploy** (quando você atualiza o código). Uma vez rodando, **não há diferença de performance** entre frontend dentro ou fora do Docker.

CI/CD lento = rebuild da imagem Docker (minutos)

↓

Acontece só quando você faz deploy de nova versão

↓


NÃO afeta usuários visualizando DICOM

### Onde ficam as imagens DICOM?

Componente	O que armazena	Tamanho
PostgreSQL	Metadados (PatientID, StudyDate, etc)	~KB por estudo
Orthanc (filesystem)	Arquivos .dcm reais, pixels	~GB por estudo

Volumes Persistentes (Fora dos containers!)

- `orthanc_data` - Arquivos DICOM (CT, MR, CR)
- `postgres_data` - Índices e metadados

 **Volumes sobrevivem a:** restart, rebuild, docker compose down, atualizações

### Performance: Frontend no Docker vs Fora

Aspecto	Dentro do Docker	Fora do Docker
Latência de rede	~0.1ms	~1-5ms
Carregamento DICOM	⚡ Idêntico	⚡ Idêntico
Consumo RAM	+50-100MB	0 (se em CDN)
Risco de perda de dados	❌ ZERO	❌ ZERO

### Arquitetura Recomendada para Produção

## SERVIDOR

- └─ Docker Compose
  - └─ Nginx + React (:80)
  - └─ OHIF Viewer (:3000)
  - └─ Orthanc PACS (:8042)
  - └─ PostgreSQL (:5432)
- └─ DISCO SSD (Sistema + PostgreSQL)
  - └─ /var/lib/docker/volumes/postgres\_data (~10-50GB)
- └─ DISCO HDD/NAS (DICOM Storage)
  - └─ /mnt/dicom-storage/orthanc\_data (~1TB - 100TB+)
- └─ BACKUP EXTERNO
  - └─ NAS / Cloud (S3, Azure Blob)

---

## Resposta Final

Pergunta	Resposta
CI/CD lento afeta visualização?	<b>NÃO</b>
PostgreSQL guarda imagens pesadas?	<b>NÃO</b> - só metadados
Risco de perder imagens no Docker?	<b>NÃO</b> - volumes persistentes
Frontend no Docker perde performance?	<b>NÃO</b>
Compensa frontend no Docker?	<b>SIM</b> para produção

## Recomendação

- **PRODUÇÃO:** Todos os 4 containers no Docker
- **DESENVOLVIMENTO:** Backend no Docker + Frontend local

**O gargalo em sistemas PACS é I/O de disco e rede, não containers.**