Guia de Deploy – Sistema de Doações com Coolify em VPS

> **Assista à demonstração em vídeo do processo de deploy:**youtube

1. Pré-requisitos

- Este guia apresenta um fluxo simplificado e pressupõe que o usuário já possui conhecimento básico sobre Coolify (instalação, navegação e conceitos principais). Caso não tenha, recomenda-se fortemente a leitura da documentação oficial do Coolify, que é bastante completa e cobre todos os detalhes necessários para operar a plataforma.
- VPS (Ubuntu recomendado, acesso root ou sudo)
- Docker e Docker Compose instalados
- Domínio configurado (DNS apontando para o IP da VPS)
- Coolify instalado na VPS (documentação oficial)
- Certificados TLS (Let's Encrypt, gerenciado pelo Coolify)
- Acesso ao repositório GitHub do projeto
- Registry Docker privado (opcional): pode ser criado no Coolify ou utilizar um registry já existente, como DockerHub, GitHub Container Registry, ou outro. O uso do registry do Coolify é recomendado para facilitar o deploy, mas não é obrigatório.

2. Estrutura do Sistema

- **Frontend:** Vue.js, servido via Nginx em container
- **Backend:** Java Spring Boot, containerizado
- **Banco de Dados:** PostgreSQL, containerizado com volume persistente
- **Uploads:** Volume dedicado para arquivos enviados
- **Proxy reverso:** Nginx ou Traefik, gerenciado pelo Coolify
- Demo pública (aplicação frontend): `https://doacoes.vitorweirich.com/`

3. Passos para Deploy

3.1. Instalar Coolify

> Alguns providers de VPS disponibilizam imagens que já vem com o coolify instalado

```
curl -fsSL https://get.coolify.io | bash
```

- Acesse o painel web do Coolify (porta 3000 por padrão) e finalize a configuração inicial.

3.2. Criar e Configurar o Banco de Dados

Você pode optar por:

- Criar um **Database** no Coolify (PostgreSQL recomendado)
    - No painel do Coolify, crie um novo serviço do tipo **Database** e selecione PostgreSQL
    - Após criado, copie as credenciais de acesso (host, porta, usuário, senha, nome do banco)
- Utilizar um banco de dados já existente externamente (ex: RDS, serviço gerenciado, outro servidor PostgreSQL)
    - Certifique-se de ter as credenciais de acesso

Em ambos os casos, configure as variáveis de ambiente da aplicação (backend) para apontar para o banco de dados correto:

- Exemplo: `DB_HOST`, `DB_PORT`, `DB_USER`, `DB_PASSWORD`, `DB_NAME`
- No Coolify, adicione essas variáveis nas configurações do serviço do backend

3.3. Criar e Configurar o Registry Docker (Opcional)

Esta etapa é opcional. Você pode:

- Criar um registry privado no Coolify (recomendado para integração direta)
- Utilizar um registry já existente, como DockerHub, GitHub Container Registry, ou outro serviço compatível

Se optar por criar um registry no Coolify:

- No painel do Coolify, crie um novo **Service** do tipo **Docker Registry**
- Após criado, copie o endereço do registry (ex: `registry.seudominio.com`), usuário e senha
- Configure o registry para permitir push/pull das imagens do backend e frontend

Se optar por usar um registry existente:

- Certifique-se de ter as credenciais de acesso (usuário/senha ou token)
- Ajuste as configurações do workflow e do Coolify para apontar para o registry escolhido

3.4. Configurar Projeto no Coolify

3.4.1. Setup Simplificado (Manual)

Este método é indicado para testes rápidos ou ambientes de desenvolvimento.

- Crie um novo "Application" para o **backend**:
    - Tipo: Docker Compose ou Build from Dockerfile
    - Fonte: Repositório GitHub (`sistema-doacoes-completo/backend`)
    - Variáveis de ambiente: Configure segredos (DB, JWT, SMTP, etc.)
- Crie um novo "Application" para o **frontend**:
    - Tipo: Docker Compose ou Build from Dockerfile
    - Fonte: Repositório GitHub (`sistema-doacoes-completo/frontend`)
    - Build args: `MODE=production`
    - Configure domínio customizado

- Configure volumes para **uploads** (arquivos enviados pelo backend)

3.4.2. Setup Automático (Recomendado - Produção)

Este método utiliza imagens Docker já buildadas e publicadas no registry, integrando com o workflow do GitHub Actions para deploy automático.

- Crie um novo "Application" para o **backend**:
    - Tipo: Docker Image
    - Informe o endereço da imagem do backend no registry privado (ex: `registry.seudominio.com/nome-backend:latest`)
    - Configure variáveis de ambiente (DB, JWT, SMTP, etc.)
- Crie um novo "Application" para o **frontend**:
    - Tipo: Docker Image
    - Informe o endereço da imagem do frontend no registry privado (ex: `registry.seudominio.com/nome-frontend:latest`)
    - Configure domínio customizado
- Configure volumes para **uploads** (arquivos enviados pelo backend)

Faça a configuração dos webhooks para deploys automaticos.

Veja mais detalhes sobre integração automática e webhooks na documentação oficial do Coolify: Automatic commit deployments with webhooks (optional)

3.5. Integração com GitHub Actions e Deploy Automático (Opcional)

- No repositório, configure os secrets do workflow:
    - `REGISTRY_USERNAME`, `REGISTRY_PASSWORD`: credenciais do registry Docker
    - `COOLIFY_BACKEND_WEBHOOK`, `COOLIFY_FRONTEND_WEBHOOK`: URLs dos webhooks de deploy do Coolify
    - `COOLIFY_TOKEN`: token de autenticação do webhook (se necessário)
- O workflow `.github/workflows/build-on-pr-merge.yml` está configurado para:
    1. Buildar as imagens Docker do backend e frontend usando Docker Buildx
    2. Fazer push das imagens para o registry privado
    3. Acionar os webhooks do Coolify para atualizar os containers automaticamente
    4. O deploy é disparado apenas quando o commit na branch `master` contém `#deploy` na mensagem

Exemplo de fluxo:

1. Desenvolvedor faz push/merge na branch `master` com mensagem `feat: nova feature #deploy`
2. GitHub Actions executa o workflow:
    - Builda imagens
    - Faz push para o registry

    - Chama os webhooks do Coolify
3. Coolify faz pull das novas imagens e atualiza os containers
4. Sistema entra em produção automaticamente

3.7. Deploy e Atualizações

- O deploy é totalmente automatizado via workflow e webhooks:
    - Push/merge com `#deploy` → build → push → webhook → atualização automática
- Acompanhe logs e status pelo painel do Coolify (containers, builds, histórico de deploys)

3.8. Boas práticas: Backup e Segurança

- Configure backup periódico do banco de dados (dump do Postgres)
- Faça backup dos volumes de uploads
- Mantenha firewall restrito (apenas portas 22, 80, 443)
- Use variáveis de ambiente para segredos (nunca commitados)
- Atualize SO e dependências regularmente