1_certificate-manager-fresh-install

Guia de Instalação: CertificateManager em Ubuntu Limpo

📋 Pré-requisitos

- Ubuntu 20.04 LTS ou superior recém instalado
- Acesso root ou sudo
- Conexão com internet

🚀 Instalação Completa

Passo 1: Atualizar Sistema e Instalar Dependências Básicas

```
# Atualizar repositórios e sistema
sudo apt update && sudo apt upgrade -y

# Instalar ferramentas essenciais
sudo apt install -y curl git wget build-essential
```

Passo 2: Instalar Node.js 20

```
# Adicionar repositório do Node.js 20
curl -fsSL https://deb.nodesource.com/setup_20.x | sudo -E bash -

# Instalar Node.js e npm
sudo apt-get install -y nodejs

# Verificar instalação
node --version  # Deve mostrar v20.x.x
npm --version   # Deve mostrar 10.x.x
```

Passo 3: Instalar PostgreSQL

```
# Instalar PostgreSQL e ferramentas
sudo apt install -y postgresql postgresql-contrib

# Iniciar e habilitar PostgreSQL
sudo systemctl start postgresql
sudo systemctl enable postgresql

# Verificar status
sudo systemctl status postgresql
```

Passo 4: Configurar Banco de Dados

```
# Acessar PostgreSQL como superusuário
sudo -u postgres psql
```

Execute no console do PostgreSQL:

```
-- Criar usuário da aplicação
CREATE USER appuser WITH PASSWORD 'strongpassword123';

-- Criar banco de dados
CREATE DATABASE tenant_management_db OWNER appuser;

-- Conceder permissões
GRANT ALL PRIVILEGES ON DATABASE tenant_management_db TO appuser;

-- Sair
\q
```

Passo 5: Clonar e Configurar Projeto

```
# Ir para diretório home
cd ~

# Clonar repositório
git clone https://github.com/mcsafx/CertificateManager.git

# Entrar na pasta
cd CertificateManager

# Instalar dependências do projeto
npm install

# Instalar dependências adicionais necessárias
npm install dotenv pg
npm install --save-dev @types/pg
```

Passo 6: Configurar Arquivos do Projeto

6.1 Criar arquivo de variáveis de ambiente

```
nano .env
```

Adicionar o conteúdo:

```
# Database Configuration
DATABASE_URL="postgresql://appuser:strongpassword123@localhost:5432/tenant_management_db"
```

```
# Application Configuration
NODE_ENV=development
PORT=5000

# Session Configuration
SESSION_SECRET="your-super-secret-session-key-change-this-in-production-abc123xyz789"

# Application URLs
VITE_API_URL=http://localhost:5000
```

6.2 Modificar server/db.ts para PostgreSQL local

```
nano server/db.ts
```

Substituir TODO o conteúdo por:

```
import { drizzle } from 'drizzle-orm/node-postgres';
import pg from 'pg';
import * as schema from "@shared/schema";
import dotenv from 'dotenv';

// Carregar variáveis de ambiente
dotenv.config();

if (!process.env.DATABASE_URL) {
  throw new Error(
    "DATABASE_URL deve ser configurado. Verifique as variáveis de ambiente."
  );
}

// Criando pool de conexões para PostgreSQL local
export const pool = new pg.Pool({
  connectionString: process.env.DATABASE_URL,
});

// Criando instância do Drizzle ORM
export const db = drizzle(pool, { schema });
```

6.3 Adicionar dotenv no server/index.ts

```
nano server/index.ts
```

Adicionar no INÍCIO do arquivo (antes de outros imports):

```
import dotenv from 'dotenv';
dotenv.config();
```

Passo 7: Preparar Banco de Dados

```
# Executar migrações
npm run db:push

# Se o comando acima funcionar, pule para o Passo 8
# Caso contrário, execute manualmente:
```

Se necessário, criar tabelas manualmente:

```
psql -h localhost -U appuser -d tenant_management_db
```

Cole o SQL do arquivo DEPLOY_LOCALHOST.md (seção de criação de tabelas)

Passo 8: Popular Dados Iniciais

```
# Conectar ao banco
psql -h localhost -U appuser -d tenant_management_db
```

Execute:

```
-- Inserir planos
INSERT INTO plans (code, name, description, price, storage_limit, max_users) VALUES
('A', 'Plano Básico', 'Funcionalidades essenciais', 99.90, 1000, 5),
('B', 'Plano Intermediário', 'Funcionalidades avançadas', 199.90, 5000, 15),
('C', 'Plano Completo', 'Todas as funcionalidades', 399.90, 20000, 50)
ON CONFLICT (code) DO NOTHING;

-- Inserir módulos
INSERT INTO modules (code, name, description, is_core) VALUES
('core', 'Módulo Core', 'Funcionalidades essenciais do sistema', true),
('products', 'Módulo Produtos', 'Gestão de produtos e inventário', false),
('certificates', 'Módulo Certificados', 'Emissão de certificados básicos', false),
('certificates_advanced', 'Certificados Avançados', 'Funcionalidades avançadas de certificados', false),
('multi_user', 'Multi-usuário', 'Gestão de múltiplos usuários', false)
ON CONFLICT (code) DO NOTHING;

-- Sair
\q
```

Passo 9: Executar Aplicação

```
# Iniciar em modo desenvolvimento
npm run dev
```

Passo 10: Acessar Sistema

1. Abrir navegador
2. Acessar: http://localhost:5000
3. O sistema criará automaticamente um usuário admin
4. Verificar credenciais nos logs do terminal

✅ Verificação Final

- [ ] PostgreSQL rodando: `sudo systemctl status postgresql`
- [ ] Aplicação rodando: Terminal mostra “serving on port 5000”
- [ ] Interface acessível: http://localhost:5000 carrega
- [ ] Login funcionando: Consegue entrar com usuário admin

🔧 Troubleshooting

Erro de porta em uso

```
sudo lsof -i :5000
sudo kill -9 [PID]
```

Erro de conexão PostgreSQL

```
sudo systemctl restart postgresql
```

Erro de permissão no banco

Recriar usuário e permissões (Passo 4)

Tempo estimado: 15-20 minutos
Dificuldade: Intermediária