



Guia de Instalação Local - Sistema Plano de Aquisição TIC

Resumo Rápido

Para instalar este projeto localmente, você precisará de:

1. **Node.js 18+** e **Yarn**
 2. **PostgreSQL** (banco de dados)
 3. **Configurar variáveis de ambiente**
 4. **Instalar dependências**
 5. **Configurar banco de dados**
-



Passo a Passo Detalhado

1. Instalar Pré-requisitos

Node.js (versão 18 ou superior)

```
# Verificar se já tem instalado
node --version
npm --version

# Se não tiver, baixar em: https://nodejs.org/
```

Yarn (gerenciador de pacotes)

```
# Instalar Yarn globalmente
npm install -g yarn

# Verificar instalação
yarn --version
```

PostgreSQL

```
# No Ubuntu/Debian
sudo apt update
sudo apt install postgresql postgresql-contrib

# No macOS (com Homebrew)
brew install postgresql

# No Windows, baixar em: https://www.postgresql.org/download/
```

2. Configurar PostgreSQL

```
# Acessar PostgreSQL
sudo -u postgres psql

# Criar banco de dados
CREATE DATABASE plano_aquisicao_tic;

# Criar usuário (opcional)
CREATE USER seu_usuario WITH PASSWORD 'sua_senha';
GRANT ALL PRIVILEGES ON DATABASE plano_aquisicao_tic TO seu_usuario;

# Sair
\q
```

3. Clonar e Configurar o Projeto

```
# Navegar para a pasta do app
cd plano_aquisicao_tic/app

# Instalar dependências
yarn install
```

4. Configurar Variáveis de Ambiente

Copie o arquivo de exemplo:


```
cp .env.example .env
```

Edite o arquivo `.env` com seus dados:

```
# Substitua pelos seus dados do PostgreSQL
DATABASE_URL="postgresql://seu_usuario:sua_senha@localhost:5432/plano_aquisicao_tic"

# Gere uma chave secreta aleatória
NEXTAUTH_SECRET="cole_aqui_uma_string_aleatoria_de_32_caracteres"

# URL do seu ambiente local
NEXTAUTH_URL="http://localhost:3000"
```

 **Dica:** Para gerar uma chave secreta segura:

```
openssl rand -base64 32
```

5. Configurar Banco de Dados

```
# Gerar cliente Prisma
npx prisma generate

# Aplicar migrations
npx prisma db push

# (Opcional) Popular com dados iniciais
npx prisma db seed
```

6. Executar a Aplicação

```
# Modo desenvolvimento
yarn dev

# A aplicação estará disponível em: http://localhost:3000
```

7. Build para Produção (opcional)

```
# Fazer build
yarn build

# Executar em produção
yarn start
```



Comandos Úteis

Prisma (Banco de Dados)

```
# Ver dados no navegador
npx prisma studio

# Reset completo do banco (CUIDADO!)
npx prisma db reset

# Aplicar mudanças no schema
npx prisma db push

# Gerar cliente após mudanças
npx prisma generate
```

Next.js

```
# Desenvolvimento
yarn dev

# Build
yarn build

# Produção
yarn start

# Lint (verificar código)
yarn lint
```

! Solução de Problemas Comuns

✗ Erro de conexão com PostgreSQL

```
# Verificar se PostgreSQL está rodando
sudo service postgresql status

# Iniciar PostgreSQL
sudo service postgresql start

# Testar conexão
psql -h localhost -U seu_usuario -d plano_aquisicao_tic
```

✗ Erro “prisma client not generated”

```
npx prisma generate
```

✗ Erro de dependências

```
# Limpar e reinstalar
rm -rf node_modules
rm yarn.lock
yarn install
```

✗ Porta 3000 já está em uso

```
# Descobrir qual processo está usando
lsof -i :3000

# Matar o processo (substitua PID)
kill -9 PID

# Ou usar outra porta
yarn dev -p 3001
```



Estrutura do Banco de Dados

O sistema possui as seguintes entidades principais:

- **Usuários** - Controle de acesso e permissões
 - **Departamentos** - Estrutura organizacional
 - **Categorias TIC** - Classificação de itens
 - **Itens TIC** - Catálogo de produtos/serviços
 - **Solicitações** - Pedidos de aquisição
 - **Aprovações** - Workflow de aprovação
 - **Exclusões** - Lista de itens excluídos
-



Precisa de Ajuda?

1. **Verificar logs de erro** no terminal
2. **Checar arquivo** `.env` - credenciais corretas?
3. **PostgreSQL rodando?** - `sudo service postgresql status`
4. **Dependências instaladas?** - `yarn install`
5. **Prisma configurado?** - `npx prisma generate`

Se ainda tiver problemas, compartilhe o erro específico que está acontecendo!