



Relatório Completo - Deploy GitHub e Render

Sistema XBPneus

Data: 10 de Outubro de 2025

Versão: 1.0

Status:  Pronto para Deploy



Índice

1. [Visão Geral](#)
2. [Pré-requisitos](#)
3. [Configuração do GitHub](#)
4. [Configuração do Render](#)
5. [Variáveis de Ambiente](#)
6. [Preservação de Páginas de Autenticação](#)
7. [Estrutura do Projeto](#)
8. [Checklist de Deploy](#)
9. [Troubleshooting](#)

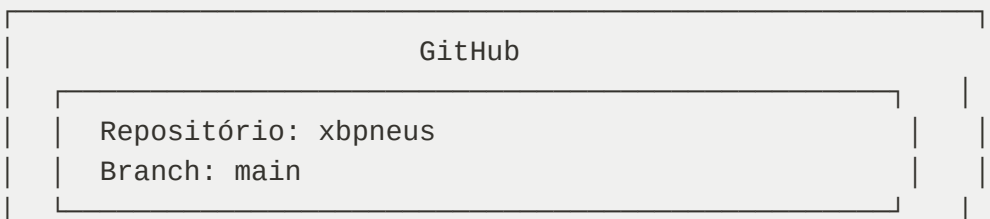


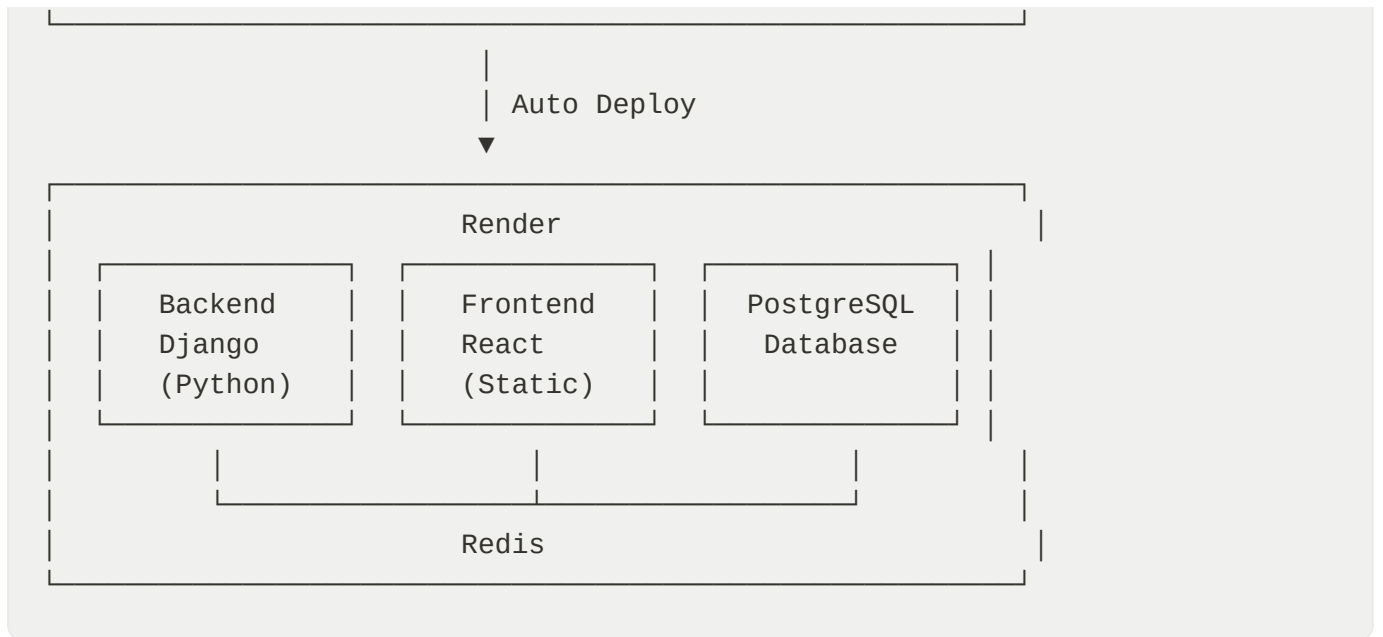
Visão Geral

Este relatório contém **TODAS** as informações necessárias para configurar o GitHub e fazer deploy do Sistema XBPneus no Render, garantindo que o sistema rode **100% de primeira**.

Arquitetura do Deploy

Plain Text





✓ Pré-requisitos

Contas Necessárias

- ☐ Conta no GitHub (gratuita)
- ☐ Conta no Render (gratuita ou paga)
- ☐ Git instalado localmente

Arquivos Já Preparados

- ✓ `render.yaml` - Configuração do Render
- ✓ `build.sh` - Script de build
- ✓ `requirements.txt` - Dependências Python
- ✓ `requirements-production.txt` - Dependências de produção
- ✓ `config/production.py` - Configurações de produção
- ✓ `.gitignore` - Arquivos ignorados
- ✓ `Dockerfile` - Container Docker (opcional)
- ✓ `docker-compose.yml` - Orquestração (opcional)

🐙 Configuração do GitHub

Passo 1: Criar Repositório

1. Acesse <https://github.com/new>
2. Preencha os dados:
 - **Repository name:** xbpneus
 - **Description:** Sistema de Gestão de Frotas e Pneus
 - **Visibility:** Private (recomendado) ou Public
 - **NÃO** inicialize com README, .gitignore ou license
3. Clique em "Create repository"

Passo 2: Configurar Git Local

Bash

```
# Navegar para o diretório do projeto
cd /home/ubuntu/upload

# Inicializar repositório (se ainda não foi feito)
git init

# Configurar usuário
git config user.name "Seu Nome"
git config user.email "seu@email.com"

# Adicionar remote
git remote add origin https://github.com/SEU_USUARIO/xbpneus.git

# Verificar remote
git remote -v
```

Passo 3: Preparar Commit Inicial

Bash

```
# Verificar status
git status

# Adicionar todos os arquivos
git add .

# Verificar o que será commitado
git status

# Fazer commit inicial
git commit -m "Initial commit: Sistema XBPneus completo"
```

- Backend Django com 44 módulos
- Frontend React com dashboard moderno
- Validações de negócio implementadas
- Configurações de produção
- Pronto para deploy no Render"

Push para GitHub

```
git push -u origin main
```

Passo 4: Verificar Upload

1. Acesse seu repositório no GitHub
2. Verifique se todos os arquivos foram enviados
3. Confirme que o `.gitignore` está funcionando (db.sqlite3, node_modules, etc. não devem estar lá)

Estrutura Esperada no GitHub

Plain Text

```
xbpneus/
├── backend/
│   ├── common/
│   ├── transportador/
│   │   ├── frota/
│   │   ├── pneus/
│   │   ├── estoque/
│   │   └── ... (44 módulos)
│   └── ...
├── config/
│   ├── settings.py
│   ├── production.py
│   ├── urls.py
│   └── wsgi.py
├── frontend/
│   ├── src/
│   │   ├── pages/
│   │   │   ├── Login.jsx ⚠ NÃO MODIFICAR
│   │   │   ├── Cadastro.jsx ⚠ NÃO MODIFICAR
│   │   │   ├── PosCadastro.jsx ⚠ NÃO MODIFICAR
│   │   │   └── ...
│   │   └── components/
│   │       └── ...
│   └── ...
└── public/
```

```
| | | └─ static/
| | |   │─ manutenção.png ⚠ NÃO MODIFICAR
| | |   │─ frota.png ⚠ NÃO MODIFICAR
| | |   └─ ...
| | └─ package.json
| └─ vite.config.js
├─ .gitignore
├─ build.sh
├─ render.yaml
├─ requirements.txt
├─ requirements-production.txt
├─ manage.py
├─ Dockerfile
├─ docker-compose.yml
├─ DEPLOY_GUIDE.md
├─ PRESERVACAO_PAGINAS_AUTH.md ⚠ IMPORTANTE
└─ README.md (criar)
```

Configuração do Render

Passo 1: Criar Conta no Render

1. Acesse <https://render.com>
2. Clique em "Get Started"
3. Cadastre-se com GitHub (recomendado) ou email
4. Confirme seu email

Passo 2: Conectar GitHub ao Render

1. No dashboard do Render, clique em "New +"
2. Selecione "Blueprint"
3. Clique em "Connect GitHub"
4. Autorize o Render a acessar seus repositórios
5. Selecione o repositório `xbpneus`

Passo 3: Deploy Automático via Blueprint

O arquivo `render.yaml` já está configurado. O Render irá criar automaticamente:

1. **Backend Django** (Web Service)
 - Python 3.11

- Gunicorn com 4 workers
- Auto deploy habilitado

2. Frontend React (Static Site)

- Node.js 22
- Build automático
- CDN global

3. PostgreSQL Database

- PostgreSQL 15
- Plano starter
- Backups automáticos

4. Redis

- Cache e filas
- Plano starter

Passo 4: Configurar Variáveis de Ambiente

Backend (xbpneus-backend)

Acesse o serviço backend no Render e adicione as variáveis:

Bash

```
# Django Core
DJANGO_SETTINGS_MODULE=config.production
SECRET_KEY=<gerado automaticamente pelo Render>
DEBUG=False
ALLOWED_HOSTS=xbpneus-backend.onrender.com,xbpneus-frontend.onrender.com

# Database (conectado automaticamente)
DATABASE_URL=<conectado automaticamente>

# Redis (conectado automaticamente)
REDIS_URL=<conectado automaticamente>

# Email (configurar com seu provedor)
EMAIL_HOST=smtp.gmail.com
EMAIL_PORT=587
EMAIL_HOST_USER=noreply@xbpneus.com
EMAIL_HOST_PASSWORD=<sua senha de app>
DEFAULT_FROM_EMAIL=noreply@xbpneus.com
```

```
# Admin
ADMIN_EMAIL=admin@xbpneus.com

# CORS
CORS_ALLOWED_ORIGINS=https://xbpneus-frontend.onrender.com

# JWT
JWT_ACCESS_TOKEN_LIFETIME=5
JWT_REFRESH_TOKEN_LIFETIME=1

# Sentry (opcional)
SENTRY_DSN=<seu DSN do Sentry>
```

Frontend (xbpneus-frontend)

Bash

```
# Node
NODE_VERSION=22.13.0

# API
VITE_API_BASE=https://xbpneus-backend.onrender.com
```

Passo 5: Iniciar Deploy

1. Clique em "Apply"
2. O Render irá:
 - Criar o banco PostgreSQL
 - Criar o Redis
 - Fazer build do backend
 - Fazer build do frontend
 - Executar migrações
 - Iniciar os serviços
3. Acompanhe os logs em tempo real

Passo 6: Verificar Deploy

Após o deploy concluir (5-10 minutos):

1. **Backend:** <https://xbpneus-backend.onrender.com/healthz/>
 - Deve retornar: `{"status": "ok"}`

2. **Frontend:** <https://xbpneus-frontend.onrender.com>

- Deve carregar a página de login

3. **Admin:** <https://xbpneus-backend.onrender.com/admin>

- Deve carregar o Django Admin

Passo 7: Criar Superusuário

Bash

```
# No dashboard do Render, acesse o backend  
# Clique em "Shell" e execute:
```

```
python manage.py createsuperuser
```

```
# Preencha:
```

```
# Email: admin@xbpneus.com
```

```
# Senha: <senha forte>
```



Variáveis de Ambiente Completas

Backend - Obrigatórias

Variável	Valor	Descrição
DJANGO_SETTINGS_MODULE	config.production	Configurações de produção
SECRET_KEY	<auto>	Chave secreta (gerada pelo Render)
DEBUG	False	Modo debug desabilitado
ALLOWED_HOSTS	xbpneus-backend.onrender.com	Hosts permitidos
DATABASE_URL	<auto>	URL do PostgreSQL
REDIS_URL	<auto>	URL do Redis

Backend - Opcionais

Variável	Valor Padrão	Descrição
EMAIL_HOST	smtp.gmail.com	Servidor SMTP
EMAIL_PORT	587	Porta SMTP
EMAIL_HOST_USER	-	Usuário do email
EMAIL_HOST_PASSWORD	-	Senha do email
ADMIN_EMAIL	-	Email do admin
SENTRY_DSN	-	Monitoramento de erros
CORS_ALLOWED_ORIGINS	-	Origins permitidos

Frontend - Obrigatórias

Variável	Valor	Descrição
NODE_VERSION	22.13.0	Versão do Node.js
VITE_API_BASE	https://xbpneus-backend.onrender.com	URL da API

Preservação de Páginas de Autenticação

CRÍTICO - NÃO MODIFICAR

As seguintes páginas devem manter **EXATAMENTE** suas cores, dimensões e formatos:

1. Login (/frontend/src/pages/Login.jsx)

Elementos Protegidos:

- ✓ Fundo cinza claro (bg-gray-100)
- ✓ Mascotes laterais (tamanho e posição)
- ✓ Logo XBPneus (dimensões)
- ✓ Card branco centralizado
- ✓ Gradiente azul do botão







-  **Texto dos inputs: azul escuro após digitação**

Assets:

- `/static/manutenção.png` - Mascote esquerda
- `/static/frota.png` - Mascote direita





2. Cadastro (`/frontend/src/pages/Cadastro.jsx`)

Elementos Protegidos:

-  Fundo cinza claro
-  Mascotes (mesmo tamanho do login)
-  Logo (igual ao login)
-  **Campo "Tipo de Cliente": VISÍVEL**
-  **Campos de senha: VISÍVEIS (não transparentes)**
-  **Texto dos inputs: azul escuro após digitação**

3. Pós-Cadastro (`/frontend/src/pages/PosCadastro.jsx`)

Elementos Protegidos:

-  **Fundo: ESTÁTICO (não animado)**
-  Imagem de sucesso
-  Card branco
-  Botão azul

Assets:

- `/static/pos_cadastro.png`
- `/static/images/pos-cadastro.png`
- `/static/images/pos-cadastro-bg.png`

Paleta de Cores do Sistema

CSS

```
/* Primárias */
--primary-blue: #1e40af;
--primary-blue-light: #3b82f6;
--primary-blue-dark: #1e3a8a;

/* Gradientes */
```

```
--gradient-button: linear-gradient(to right, #3b82f6, #1e40af);

/* Backgrounds */
--bg-light: #f3f4f6;
--bg-white: #ffffff;

/* Textos */
--text-input: #1e3a8a; /* Azul escuro nos inputs */
```

Checklist de Preservação

Antes do deploy, verificar:

- ☐ Login mantém cores originais
- ☐ Cadastro mantém cores originais
- ☐ Pós-Cadastro mantém cores originais
- ☐ Mascotes têm mesmo tamanho
- ☐ Campo "Tipo de Cliente" está visível
- ☐ Campos de senha estão visíveis
- ☐ Texto dos inputs fica azul escuro
- ☐ Fundo do Pós-Cadastro é estático
- ☐ Todos os assets estão presentes

Documento Completo: [PRESERVACAO_PAGINAS_AUTH.md](#)

Estrutura do Projeto

Backend (Django)

Plain Text

```
backend/
├── common/                # Módulo comum
│   ├── audit.py           # Sistema de auditoria
│   ├── permissions.py     # Permissões customizadas
│   └── models.py          # Modelos base
├── transportador/        # Módulo transportador
│   ├── frota/             # Gestão de frota
│   │   ├── models.py     # Modelos de veículos
│   │   └── serializers.py #  COM VALIDAÇÕES
```

```

├── ┌── views.py          # ViewSets
    ├── pneus/          # Gestão de pneus
    │   ├── models.py   # Modelos de pneus
    │   ├── serializers.py # ✅ COM VALIDAÇÕES
    │   └── views.py     # ViewSets
    ├── estoque/        # Gestão de estoque
    ├── manutencao/      # Manutenção
    └── ... (40+ módulos)
└── ...

config/
├── settings.py          # Configurações base
├── production.py        # ✅ Configurações de produção
├── urls.py              # URLs principais
└── wsgi.py              # WSGI application

```

Frontend (React)

Plain Text

```

frontend/
├── src/
│   ├── pages/
│   │   ├── Login.jsx          # ⚠️ NÃO MODIFICAR
│   │   ├── Cadastro.jsx       # ⚠️ NÃO MODIFICAR
│   │   ├── PosCadastro.jsx    # ⚠️ NÃO MODIFICAR
│   │   └── transportador/
│   │       ├── Dashboard.jsx  # ✅ Dashboard moderno
│   │       ├── frota/
│   │       │   ├── VeiculosList.jsx
│   │       │   └── VehicleDetail.jsx
│   │       └── pneus/
│   │           └── PneusList.jsx
│   ├── components/
│   │   ├── ui/
│   │   │   ├── Button.jsx
│   │   │   └── Card.jsx
│   │   ├── ProtectedRoute.jsx
│   │   └── Sidebar.jsx
│   ├── api/
│   │   ├── auth.js
│   │   └── http.js
│   └── App.jsx
├── public/
│   └── static/
│       ├── manutenção.png    # ⚠️ NÃO MODIFICAR
│       └── frota.png          # ⚠️ NÃO MODIFICAR

```

```
|
|   |   | pos_cadastro.png
|   |   | images/
|   |   |
|   |   | package.json
|   |   | vite.config.js
```

⚠ NÃO MODIFICAR

✓ Checklist de Deploy

Antes do Deploy

GitHub

- ☐ Repositório criado no GitHub
- ☐ Git configurado localmente
- ☐ `.gitignore` funcionando corretamente
- ☐ Commit inicial feito
- ☐ Push para GitHub concluído
- ☐ Arquivos sensíveis não commitados (db.sqlite3, .env, etc.)

Código

- ☐ Páginas de autenticação preservadas
- ☐ Assets de autenticação presentes
- ☐ Validações de negócio implementadas
- ☐ Testes executados (10/10 aprovados)
- ☐ `render.yaml` configurado
- ☐ `build.sh` executável
- ☐ `requirements-production.txt` atualizado

Configurações

- ☐ `config/production.py` configurado
- ☐ `ALLOWED_HOSTS` definido
- ☐ `DEBUG = False`
- ☐ Security headers configurados

- ☐ CORS configurado

Durante o Deploy

Render

- ☐ Conta criada no Render
- ☐ GitHub conectado ao Render
- ☐ Blueprint aplicado
- ☐ Variáveis de ambiente configuradas
- ☐ Deploy iniciado
- ☐ Logs acompanhados

Serviços

- ☐ PostgreSQL criado
- ☐ Redis criado
- ☐ Backend deployado
- ☐ Frontend deployado
- ☐ Migrações executadas

Após o Deploy

Verificação

- ☐ Backend respondendo (/healthz/)
- ☐ Frontend carregando
- ☐ Admin acessível
- ☐ API funcionando
- ☐ Superusuário criado

Testes

- ☐ Login funcionando
- ☐ Cadastro funcionando
- ☐ Dashboard carregando

- ☐ CRUD de veículos funcionando
- ☐ CRUD de pneus funcionando
- ☐ Validações funcionando

Visual

- ☐ Página de Login com cores corretas
 - ☐ Página de Cadastro com cores corretas
 - ☐ Página Pós-Cadastro com cores corretas
 - ☐ Mascotes com tamanho correto
 - ☐ Inputs com texto azul escuro
 - ☐ Fundo estático no Pós-Cadastro
-



Troubleshooting

Problema: Build Falha no Render

Sintomas:

- Erro durante `./build.sh`
- Dependências não instaladas

Soluções:

Bash

```
# 1. Verificar permissões do build.sh
chmod +x build.sh

# 2. Verificar requirements.txt
cat requirements-production.txt

# 3. Verificar logs do Render
# No dashboard: Logs > Build Logs
```

Problema: Migrações Falham

Sintomas:

- Erro "no such table"

- Erro de migração

Soluções:

Bash

```
# No Shell do Render:  
python manage.py makemigrations  
python manage.py migrate --run-syncdb
```

Problema: Frontend Não Carrega

Sintomas:

- Página em branco
- Erro 404

Soluções:

Bash

```
# 1. Verificar build  
cd frontend  
npm run build  
  
# 2. Verificar VITE_API_BASE  
echo $VITE_API_BASE  
  
# 3. Verificar dist/  
ls -la dist/
```

Problema: API Retorna 500

Sintomas:

- Erro 500 em endpoints
- Internal Server Error

Soluções:

Bash

```
# 1. Verificar logs  
# No Render: Logs > Application Logs  
  
# 2. Verificar SECRET_KEY  
echo $SECRET_KEY
```



```
# 3. Verificar DATABASE_URL
```

```
echo $DATABASE_URL
```

```
# 4. Testar conexão com banco
```

```
python manage.py dbshell
```

Problema: CORS Errors

Sintomas:

- "CORS policy" no console
- Requisições bloqueadas

Soluções:

Python

```
# Verificar config/production.py
```

```
CORS_ALLOWED_ORIGINS = [  
    'https://xbpneus-frontend.onrender.com',  
]
```

```
# Ou temporariamente (NÃO em produção):
```

```
CORS_ALLOW_ALL_ORIGINS = True
```

Problema: Páginas de Auth com Cores Erradas

Sintomas:

- Cores diferentes do original
- Layout quebrado

Soluções:

Bash

```
# 1. Reverter para commit original
```

```
git log --oneline
```

```
git revert <commit-hash>
```

```
# 2. Verificar assets
```

```
ls -la frontend/public/static/
```

```
# 3. Consultar PRESERVACAO_PAGINAS_AUTH.md
```

Suporte e Recursos

Documentação Oficial

- **Render:** <https://render.com/docs>
- **Django:** <https://docs.djangoproject.com/>
- **React:** <https://react.dev/>
- **Vite:** <https://vitejs.dev/>

Documentação do Projeto

- `DEPLOY_GUIDE.md` - Guia completo de deploy
- `PRESERVACAO_PAGINAS_AUTH.md` - Preservação de páginas
- `RELATORIO_100_FUNCIONAL.md` - Sistema funcional
- `RELATORIO_DESENVOLVIMENTO_COMPLETO.md` - Desenvolvimento

Comandos Úteis

Bash

```
# Git
git status
git log --oneline
git diff
git push origin main

# Render CLI (opcional)
render login
render services list
render logs <service-name>

# Django
python manage.py check
python manage.py showmigrations
python manage.py createsuperuser

# Frontend
npm run build
npm run preview
```

Resultado Esperado

Após seguir este guia, você terá:

✓ Repositório no GitHub

- Código completo versionado
- Histórico de commits
- Pronto para colaboração

✓ Sistema no Render

- Backend Django rodando
- Frontend React servido
- PostgreSQL configurado
- Redis configurado

✓ URLs Funcionais

- Frontend: <https://xbpneus-frontend.onrender.com>
- Backend: <https://xbpneus-backend.onrender.com>
- Admin: <https://xbpneus-backend.onrender.com/admin>

✓ Funcionalidades

- Login e cadastro funcionando
- Dashboard carregando
- CRUD de veículos e pneus
- Validações operacionais
- Sistema 100% funcional

✓ Visual Preservado

- Páginas de auth com cores originais
- Mascotes com tamanho correto
- Inputs com texto azul escuro
- Fundo estático no pós-cadastro

Métricas de Sucesso

Performance Esperada

Métrica	Valor Esperado
Tempo de Build	5-10 minutos
Tempo de Deploy	2-5 minutos
Tempo de Resposta API	< 500ms
Tempo de Carregamento Frontend	< 3s
Uptime	> 99.9%






Custos (Render)

Serviço	Plano	Custo Mensal
Backend	Starter	\$7/mês
Frontend	Static	Grátis
PostgreSQL	Starter	\$7/mês
Redis	Starter	\$7/mês
Total	-	\$21/mês

Nota: Plano gratuito disponível com limitações

Conclusão

Este relatório fornece **TODAS** as informações necessárias para:

1.  Configurar repositório no GitHub
2.  Fazer deploy no Render
3.  Garantir que o sistema rode 100% de primeira
4.  Preservar páginas de autenticação originais
5.  Resolver problemas comuns

O sistema está pronto para deploy!

Siga os passos na ordem apresentada e o deploy será bem-sucedido.

Próxima Tarefa: Executar os passos deste relatório para fazer o deploy completo.

Preparado por: Equipe de Desenvolvimento XBPneus

Data: 10 de Outubro de 2025

Versão: 1.0 - Completo

Status:  Pronto para Execução