🚀 GUIA COMPLETO DE DEPLOY NA VERCEL

📋 Índice

🎯 Pré-requisitos

1. Conta na Vercel

- Criar conta em: https://vercel.com/signup
- Pode usar GitHub, GitLab ou email

2. Banco de Dados PostgreSQL

Você precisa de um banco PostgreSQL em produção. Opções:

Opção A: Supabase (Recomendado - Gratuito)

```
1. Criar conta: https://supabase.com/
2. Create New Project
3. Copiar a DATABASE_URL (Connection String)
   Formato: postgresql://postgres:[password]@[host]:5432/postgres
```

Opção B: Railway

```
1. Criar conta: https://railway.app/
2. New Project > Provision PostgreSQL
3. Copiar a DATABASE_URL
```

Opção C: Neon

```
1. Criar conta: https://neon.tech/
2. Create Project
3. Copiar a Connection String
```

3. Stripe (Sistema de Pagamentos)

```
1. Criar conta: https://dashboard.stripe.com/register
2. Obter chaves:
   - Dashboard > Developers > API keys
   - Copiar: Publishable key e Secret key
```

4. AWS S3 (Opcional - usar o mesmo bucket atual)

```
Você já tem configurado:
- Bucket: abacusai-apps-a586e61459b01a2447123971-us-west-2
- Region: us-west-2

OU criar novo bucket em: https://console.aws.amazon.com/s3/
```

🚀 OPÇÃO 1: Deploy Direto (Sem GitHub)

Passo 1: Instalar Vercel CLI

```
npm install -g vercel
```

Passo 2: Fazer Login

```
vercel login
```

Passo 3: Deploy

```
# Ir para a pasta do projeto
cd /home/ubuntu/ats_platform/nextjs_space

# Iniciar deploy
vercel

# Responder as perguntas:
# ? Set up and deploy "~/ats_platform/nextjs_space"? [Y/n] y
# ? Which scope do you want to deploy to? (Selecionar sua conta)
# ? Link to existing project? [y/N] n
# ? What's your project's name? recruitai
# ? In which directory is your code located? ./
# ? Want to override the settings? [y/N] n
```

Passo 4: Configurar Variáveis de Ambiente

```
# Adicionar variáveis uma por uma
vercel env add DATABASE_URL production
# Cole o valor quando solicitado

vercel env add NEXTAUTH_SECRET production
# Gerar: node -e "console.log(require('crypto').randomBytes(32).toString('hex'))"

vercel env add NEXTAUTH_URL production
# Valor: https://seu-projeto.vercel.app

# Continuar para todas as variáveis (ver seção completa abaixo)
```

Passo 5: Deploy de Produção

```
vercel --prod
```

Pronto! Sua aplicação estará disponível em: `https://seu-projeto.vercel.app`

💻 OPÇÃO 2: Deploy com GitHub (Recomendado)

Esta opção permite deploy automático a cada push.

Passo 1: Preparar o Código

1.1 Criar `.gitignore` (se não existir)

```
cd /home/ubuntu/ats_platform/nextjs_space

cat > .gitignore << 'EOF'
# dependencies
/node_modules
/.pnp
.pnp.js

# testing
/coverage

# next.js
/.next/
/out/

# production
/build

# misc
.DS_Store
*.pem

# debug
npm-debug.log*
yarn-debug.log*
yarn-error.log*

# local env files
.env
.env*.local
.env.production

# vercel
.vercel

# typescript
*.tsbuildinfo
next-env.d.ts

# prisma
prisma/*.db
prisma/*.db-journal
EOF
```

1.2 Criar `README.md` para o projeto

```
cat > README.md << 'EOF'
# RecruitAI - Plataforma de Recrutamento Inteligente

Sistema ATS (Applicant Tracking System) com IA integrada.

## Stack

- Next.js 14
- React 18
- TypeScript
- Prisma ORM
- PostgreSQL
- Stripe
- AWS S3
- Tailwind CSS

## Setup

1. Clone o repositório
2. `yarn install`
3. Configure `.env` (ver `.env.example`)
4. `yarn prisma db push`
5. `yarn tsx scripts/seed.ts`
6. `yarn dev`

## Deploy

Ver `DEPLOY_VERCEL.md`
EOF
```

Passo 2: Inicializar Git

```
cd /home/ubuntu/ats_platform/nextjs_space

# Inicializar repositório
git init

# Adicionar arquivos
git add .

# Primeiro commit
git commit -m "Initial commit - RecruitAI Platform"
```

Passo 3: Criar Repositório no GitHub

Via Navegador:

```
1. Ir para: https://github.com/new
2. Repository name: recruitai-platform
3. Description: Sistema ATS com IA integrada
4. Visibility: Private (recomendado)
5. Não inicializar com README (já temos)
6. Create repository
```

Copiar comandos mostrados no GitHub:

```
git remote add origin https://github.com/SEU_USUARIO/recruitai-platform.git
git branch -M main
git push -u origin main
```

Passo 4: Conectar Vercel ao GitHub

4.1 Na Vercel Dashboard:

```
1. Ir para: https://vercel.com/new
2. Clicar em "Import Git Repository"
3. Se primeiro acesso: Clicar "Connect GitHub"
   - Autorizar Vercel no GitHub
4. Selecionar o repositório: recruitai-platform
5. Clicar "Import"
```

4.2 Configurar o Projeto:

```
Project Name: recruitai-platform
Framework Preset: Next.js (detectado automaticamente)
Root Directory: ./ (deixar padrão)
Build Command: next build (padrão)
Output Directory: .next (padrão)
Install Command: yarn install (padrão)
```

NÃO FAZER DEPLOY AINDA! Primeiro configurar variáveis.

Passo 5: Configurar Variáveis de Ambiente na Vercel

```
1. Na página do projeto, ir para: Settings > Environment Variables
2. Adicionar cada variável (ver seção completa abaixo)
3. Environment: Production, Preview, Development (marcar todos)
```

Passo 6: Fazer o Deploy

```
1. Voltar para a aba "Deployments"
2. Clicar em "Redeploy" ou fazer novo push no GitHub
3. Aguardar build (3-5 minutos)
```

Passo 7: Deploys Automáticos

Agora a cada push no GitHub:

```
git add .
git commit -m "Sua mensagem"
git push origin main
```

A Vercel fará deploy automaticamente! 🎉

🔐 Configuração de Variáveis de Ambiente

Lista Completa de Variáveis

1. Database (Obrigatório)

```
DATABASE_URL
# Exemplo: postgresql://user:password@host:5432/database
# Obter de: Supabase, Railway ou Neon
```

2. NextAuth (Obrigatório)

```
NEXTAUTH_SECRET
# Gerar novo: node -e "console.log(require('crypto').randomBytes(32).toString('hex'))"
# Exemplo: c8d9e4f3a2b1c7d6e5f4a3b2c1d0e9f8a7b6c5d4e3f2a1b0c9d8e7f6a5b4c3d2e1

NEXTAUTH_URL
# Produção: https://seu-dominio.vercel.app
# Custom domain: https://seu-dominio.com
```

3. Stripe (Obrigatório para pagamentos)

```
STRIPE_SECRET_KEY
# Obter em: https://dashboard.stripe.com/apikeys
# Produção: sk_live_xxx
# Teste: sk_test_xxx

STRIPE_PUBLISHABLE_KEY
# Produção: pk_live_xxx
# Teste: pk_test_xxx

STRIPE_WEBHOOK_SECRET
# Obter após configurar webhook (ver abaixo)
# Formato: whsec_xxx
```

Configurar Webhook do Stripe:

```
1. Ir para: https://dashboard.stripe.com/webhooks
2. Add endpoint
3. Endpoint URL: https://seu-dominio.vercel.app/api/webhooks/stripe
4. Events to send:
   - checkout.session.completed
   - customer.subscription.created
   - customer.subscription.updated
   - customer.subscription.deleted
   - invoice.paid
   - invoice.payment_failed
5. Copiar o "Signing secret" (whsec_xxx)
```

4. AWS S3 (Obrigatório para currículos)

```
AWS_REGION
# Valor: us-west-2

AWS_BUCKET_NAME
# Atual: abacusai-apps-a586e61459b01a2447123971-us-west-2
# Ou novo bucket que você criar

AWS_FOLDER_PREFIX
# Valor: resumes/
# Ou outro prefixo de sua escolha

# Apenas se usar outro bucket:
AWS_ACCESS_KEY_ID
# Obter em: AWS IAM

AWS_SECRET_ACCESS_KEY
# Obter em: AWS IAM
```

5. Abacus.AI (Obrigatório para IA)

```
ABACUSAI_API_KEY
# Valor atual: 5bb8032f287b4b89bfcae4529b50a199
# Se quiser nova key, obter em: apps.abacus.ai
```

6. Email SMTP (Obrigatório)

```
SMTP_HOST
# Valor: smtp.zoho.com
# Ou outro provedor SMTP

SMTP_PORT
# Valor: 587
# Porta padrão STARTTLS

SMTP_USER
# Seu email de envio: noreply@seudominio.com

SMTP_PASS
# Senha do email ou App Password

SMTP_FROM_NAME
# Nome de exibição: RecruitAI
```

Configurar Email:

1. Criar conta de email para envios (ex: noreply@)
2. Habilitar autenticação de app se necessário
3. Testar envio antes do deploy

7. API de Manutenção (Obrigatório)

```
MAINTENANCE_SECRET
# Token de segurança para API de manutenção
# Gerar: node -e "console.log(require('crypto').randomBytes(32).toString('hex'))"
# Exemplo: 3977aa7046e9bf25ce7e91d535177b4c00794ec8fd29b98b5fc5a2697a455c1e
```

⚠️ **IMPORTANTE:** Guarde este token em local seguro! Ele permite acesso total à API de manutenção.

8. Sistema de Teste (Opcional)

```
TEST_MODE_EMAIL
# Email para testes de pagamento sem cobrança
# Exemplo: teste@fcmtech.com.br
```

Quando usar: Para testar planos em produção sem processar pagamentos reais.

9. OAuth (Opcional)

```
GOOGLE_CLIENT_ID
# Obter em: https://console.cloud.google.com/

GOOGLE_CLIENT_SECRET
# Obter junto com Client ID

LINKEDIN_CLIENT_ID
# Obter em: https://www.linkedin.com/developers/

LINKEDIN_CLIENT_SECRET
# Obter junto com Client ID
```

Como Adicionar na Vercel

Via Dashboard:

```
1. Project > Settings > Environment Variables
2. Para cada variável:
   - Name: DATABASE_URL
   - Value: (colar o valor)
   - Environment: ☑ Production ☑ Preview ☑ Development
   - Save
```

Via CLI:

```
vercel env add DATABASE_URL production
# Cole o valor quando solicitado

vercel env add NEXTAUTH_SECRET production
# Cole o valor...

# Repetir para todas as variáveis
```

💾 Configuração do Banco de Dados

Após Configurar DATABASE_URL na Vercel

Método 1: Via Vercel CLI (Recomendado)

```
# 1. Fazer pull das env vars
vercel env pull

# 2. Rodar migrations
yarn prisma db push

# 3. Rodar seed (opcional)
yarn tsx --require dotenv/config scripts/seed.ts
```

Método 2: Via Console da Vercel

```
1. Ir para: Project > Deployments
2. Abrir o deployment atual
3. Clicar nos 3 pontinhos > "Open Build Logs"
4. Verificar se o Prisma rodou corretamente

Se houver erro, rodar manualmente:
```bash
# Conectar ao banco e rodar
psql $DATABASE_URL

# Ou usar Prisma Studio
vercel env pull
yarn prisma studio
```

Verificar Tabelas

```
-- Conectar ao banco
psql "sua_database_url"

-- Listar tabelas
\dt

-- Deve mostrar:
-- User, CandidateProfile, Job, Application, Plan, Subscription, etc.
```

🌐 Configurar Domínio Personalizado

Passo 1: Na Vercel

```
1. Project > Settings > Domains
2. Add Domain
3. Digite: seudominio.com
4. Add
```
```

Passo 2: Configurar DNS

A Vercel mostrará instruções. Exemplo:

Para Domínio Apex (seudominio.com):

```
Type: A
Name: @
Value: 76.76.21.21
```

Para WWW (www.seudominio.com):

```
Type: CNAME
Name: www
Value: cname.vercel-dns.com
```

Passo 3: Aguardar Propagação

- Tempo: 24-48 horas (geralmente 1-2 horas)
- Verificar em: https://dnschecker.org/

Passo 4: Atualizar Variáveis

```
NEXTAUTH_URL=https://seudominio.com

# E no Stripe webhook:
https://seudominio.com/api/webhooks/stripe
```

🔍 Troubleshooting

Problema: Build Failed

Erro: TypeScript errors

```
# Verificar tipos localmente
yarn tsc --noEmit

# Se houver erros, corrigir e fazer push
```

Erro: Module not found

```
# Verificar dependencies no package.json
yarn install

# Commit e push
git add package.json yarn.lock
git commit -m "Fix dependencies"
git push
```

Problema: Database Connection Error

Erro: Can’t reach database

```
1. Verificar DATABASE_URL na Vercel
2. Verificar se o banco permite conexões externas
3. Para Supabase: Verificar se está no plano gratuito (máximo 2 conexões simultâneas)
```

Solução para Supabase:

```
1. No Supabase Dashboard
2. Settings > Database > Connection Pooling
3. Usar a Connection String com pooling
4. Atualizar DATABASE_URL na Vercel
```

Problema: Webhook do Stripe não funciona

```
1. Verificar endpoint: https://seu-dominio.vercel.app/api/webhooks/stripe
2. Testar no Stripe Dashboard: Send test webhook
3. Ver logs na Vercel: Project > Logs > Functions
4. Verificar STRIPE_WEBHOOK_SECRET na Vercel
```

Problema: Página 404

Rotas dinâmicas não funcionando:

```
Verificar se está usando:
- export const dynamic = 'force-dynamic'
Em todas as route.ts
```

Problema: Variáveis de Ambiente não carregam

```
# 1. Fazer pull das variáveis
vercel env pull

# 2. Verificar se estão no .env.local
cat .env.local

# 3. Redeployar
vercel --prod
```

Problema: S3 Upload Fail

```
1. Verificar AWS credentials na Vercel
2. Verificar permissões do bucket
3. Verificar CORS do bucket:
```

```
[
  {
    "AllowedOrigins": ["https://seu-dominio.vercel.app"],
    "AllowedMethods": ["PUT", "POST", "GET", "DELETE"],
    "AllowedHeaders": ["*"],
    "MaxAgeSeconds": 3000
  }
]
```

📊 Monitoramento

Logs da Vercel

```
1. Project > Logs
2. Filtrar por tipo:
   - Functions (API routes)
   - Build (erros de build)
   - Static (páginas)
```

Analytics

```
1. Project > Analytics
2. Ver:
   - Page views
   - Top pages
   - Performance
```

Alertas

```
1. Project > Settings > Integrations
2. Adicionar:
   - Slack (notificações de deploy)
   - Discord
   - Email
```

🔄 Atualizações Futuras

Atualizar o Código

```bash
# 1. Fazer mudanças no código
git add .
git commit -m "Descrição das mudanças"
git push origin main

# 2. Vercel faz deploy automaticamente

# 3. Se precisar de rollback:
# Ir para: Project > Deployments
# Clicar em deployment antigo
# Promote to Production
```

Atualizar Banco de Dados

```
# 1. Fazer mudanças no schema.prisma
# 2. Gerar migration (se usar migrations)
yarn prisma migrate dev --name descricao

# 3. OU usar db push
yarn prisma db push

# 4. Commit e push
git add prisma/
git commit -m "Update database schema"
git push

# 5. Rodar migrations em produção (se necessário)
vercel env pull
yarn prisma db push
```

📝 Checklist Final

Antes de considerar o deploy completo:

- [ ] Banco de dados criado e funcionando
- [ ] Todas as variáveis de ambiente configuradas
- [ ] Tabelas criadas (prisma db push)
- [ ] Dados de seed inseridos (opcional)
- [ ] Build passa sem erros
- [ ] Deploy feito com sucesso
- [ ] Site acessível na URL da Vercel
- [ ] Stripe configurado e testado
- [ ] Webhooks do Stripe funcionando
- [ ] S3 upload funcionando
- [ ] Login/cadastro funcionando
- [ ] Domínio personalizado configurado (se aplicável)
- [ ] SSL (HTTPS) ativo
- [ ] Notificações por email configuradas (se aplicável)

🎉 Próximos Passos Após Deploy

1. **Testar o Sistema:**
   - Criar conta de teste
   - Criar vaga
   - Candidatar-se
   - Testar pagamento (modo teste)

2. **Configurar Monitoring:**
   - Ativar alertas na Vercel

- Monitorar logs
- Verificar performance

3. **Backup:**
   - Configurar backups automáticos do banco
   - Fazer backup do código (GitHub)

4. **Documentação:**
   - Documentar quaisquer customizações
   - Manter README atualizado

📞 Recursos e Suporte

Documentação Oficial

- **Vercel:** https://vercel.com/docs
- **Next.js:** https://nextjs.org/docs
- **Prisma:** https://www.prisma.io/docs
- **Stripe:** https://stripe.com/docs

Comunidade

- **Vercel Discord:** https://vercel.com/discord
- **Next.js Discord:** https://nextjs.org/discord

Em Caso de Problemas

1. Verificar logs na Vercel
2. Consultar esta documentação
3. Verificar status da Vercel: https://www.vercelstatus.com/

FIM DO GUIA DE DEPLOY

Boa sorte com o deploy! 🚀