

# ADK Arbitrage Profit Guard - Configuração Completa Vercel + Render

## RESUMO DAS CORREÇÕES IMPLEMENTADAS

### PROBLEMA RESOLVIDO: Frontend no Vercel não funcionando

#### CAUSA IDENTIFICADA:

- Projeto estava configurado como Vite + Express servindo frontend e backend juntos
- Vercel não conseguia servir o frontend adequadamente
- Faltava configuração de CORS para comunicação entre domínios separados

#### SOLUÇÃO IMPLEMENTADA:

1. **Separação completa Frontend/Backend**
2. **Conversão para Next.js puro no frontend**
3. **Configuração de CORS adequada no backend**
4. **Estrutura otimizada para deploy em plataformas diferentes**

## ESTRUTURA FINAL DO PROJETO

```
ADKArbitrageProfitGuard/
├── nextjs-frontend/
│   ├── src/app/
│   ├── src/components/
│   ├── src/lib/
│   ├── .env
│   ├── next.config.js
│   └── vercel.json
│   └── ← NEW Frontend Next.js para Vercel
│       ├── ← App Router Next.js
│       ├── ← Componentes React adaptados
│       ├── ← Utilitários e configurações
│       ├── ← Variáveis de ambiente
│       ├── ← Configuração Next.js
│       └── ← Configuração Vercel
├── server/
│   ├── index-fixed.ts
│   ├── routes-fixed.ts
│   └── ...
│   └── ← Backend Node.js para Render
│       ├── ← CORS configurado
│       └── ← APIs funcionais
└── render.yaml
    └── ← CORS_ORIGINS atualizado
```



## CONFIGURAÇÕES CRÍTICAS IMPLEMENTADAS



### 1. CORS no Backend (server/index-fixed.ts)

```
const corsOptions = {
  origin: [
    'https://adkarbitrageprofitguard.vercel.app',
    'https://*.vercel.app',
    'http://localhost:3000',
    process.env.FRONTEND_URL
  ],
  credentials: true,
  methods: ['GET', 'POST', 'PUT', 'DELETE', 'OPTIONS'],
  allowedHeaders: ['Content-Type', 'Authorization', 'Accept']
};
app.use(cors(corsOptions));
```



### 2. Frontend Next.js (nextjs-frontend/.env)

```
NEXT_PUBLIC_API_BASE_URL=https://adkarbitrageprofitguard.onrender.com
NEXT_PUBLIC_APP_NAME=ADK Arbitrage Profit Guard
NEXT_PUBLIC_APP_VERSION=2.1.0
```



### 3. Vercel Rewrites (vercel.json)

```
{
  "rewrites": [
    {
      "source": "/api/(.*)",
      "destination": "https://adkarbitrageprofitguard.onrender.com/api/$1"
    }
  ]
}
```



## COMPONENTES CONVERTIDOS E FUNCIONAIS





### Componentes UI Base

- ✓ Button - Botões interativos
- ✓ Card - Cards de conteúdo
- ✓ Badge - Badges de status
- ✓ Toast - Notificações
- ✓ Tooltip - Dicas contextuais
- ✓ DropdownMenu - Menus suspensos










### Componentes Principais

- ✓ ThemeProvider - Sistema de temas
- ✓ StatusIndicator - Indicador de status do bot
- ✓ BotControlButtons - Controles do bot
- ✓ DashboardMetrics - Métricas do dashboard

-  ArbitrageChart - Gráficos de arbitragem
-  HomePage - Página principal do dashboard

## Funcionalidades Implementadas

-  **Dashboard em tempo real** - Atualização automática de dados
-  **Gráficos interativos** - Charts.js integrado
-  **Controles funcionais** - Start/Pause/Stop do bot
-  **Design responsivo** - Mobile-first approach
-  **Tema dark/light** - Alternância automática
-  **Notificações** - Toast notifications
-  **Auto-refresh** - Dados atualizados automaticamente

## DEPLOY INSTRUCTIONS

### 1. Deploy Frontend no Vercel

```
# 1. Conectar repositório ao Vercel
# 2. Configurar projeto:
#   - Root Directory: nextjs-frontend
#   - Build Command: npm run build
#   - Output Directory: .next





# 3. Variáveis de ambiente no Vercel:
NEXT_PUBLIC_API_BASE_URL=https://adkarbitrageprofitguard.onrender.com
NEXT_PUBLIC_APP_NAME=ADK Arbitrage Profit Guard
NEXT_PUBLIC_APP_VERSION=2.1.0
```

### 2. Atualizar Backend no Render





```
# Adicionar variáveis no Render Dashboard:
FRONTEND_URL=https://adkarbitrageprofitguard.vercel.app
CORS_ORIGINS=https://adkarbitrageprofitguard.vercel.app,https://*.vercel.app
```

## RESULTADOS ESPERADOS

### Frontend (Vercel)

-  Dashboard carrega corretamente
-  Componentes renderizam sem erros
-  Tema dark/light funciona
-  Layout responsivo em todas as telas

### Comunicação Frontend ↔ Backend

-  Requisições API funcionam sem CORS errors
-  Dados são carregados em tempo real
-  Controles do bot respondem adequadamente
-  Notificações são exibidas corretamente



## Funcionalidades Operacionais

- ☒ Métricas do dashboard atualizando
  - ☒ Gráficos de arbitragem interativos
  - ☒ Lista de oportunidades em tempo real
  - ☒ Status do bot sendo exibido corretamente
- 



## PRÓXIMOS PASSOS

---

### 1. Deploy no Vercel:

- Conectar repositório
- Configurar `nextjs-frontend` como root directory
- Configurar variáveis de ambiente
- Deploy automático

### 2. Atualizar Render:

- Adicionar `FRONTEND_URL` e `CORS_ORIGINS`
- Restart do serviço para aplicar mudanças

### 3. Verificação:

- Testar frontend: `https://[seu-projeto].vercel.app`
  - Testar comunicação com backend
  - Validar todas as funcionalidades
- 



## STATUS: PROJETO PRONTO PARA DEPLOY

---



### CHECKLIST COMPLETO

- ☒ Frontend Next.js criado e funcionando
- ☒ Build sem erros
- ☒ Todos os componentes convertidos
- ☒ CORS configurado no backend
- ☒ Variáveis de ambiente definidas
- ☒ Configurações Vercel criadas
- ☒ Instruções de deploy documentadas



**O projeto está 100% pronto para deploy no Vercel com backend no Render!**