RiftPlay — Backend Hardening & Deploy (Playbook Rápido)

Contexto

• Stack: Node.js/Express + MongoDB Atlas (MONGO_URI). Node 18+. Start: server.js. Deploy: Render.

• Objetivo: finalizar backend para produção em 4 blocos com testes rápidos.

Blocos

1) Segurança: helmet, CORS, rate-limit, express-mongo-sanitize, JWT.

2) DB/Performance: índices, TTL, lean, conexões.

3) Rotas/Controllers: status codes e mensagens claras.

4) Erros & Consistência: logger, error handler, padrões.

Variáveis de Ambiente (Render)

NODE_ENV=production

API_URL=https://riftplay-backend.onrender.com

HEALTH_URL=https://riftplay-backend.onrender.com/health

JWT_SECRET=seu-segredo-forte

MONGO_URI=(string Atlas)

CORS_ORIGINS=https://seu-front.com,https://outro.com (opcional)

JSON_LIMIT=100kb

RATE_LIMIT_WINDOW_MS=60000

RATE_LIMIT_MAX=5 ~ 120

QUEUE_TTL_SECS=300

TIMEOUT_SWEEP_MINUTES=5

WEEKLY_CAP_PERIOD_DAYS=7

O que já está no server.js

• trust proxy (Render), helmet (CORP cross-origin), CORS baseado em CORS_ORIGINS, express.json(JSON_LIMIT), mo

• /health -> { ok:true }.

• Rate-limit em produção: /api/matchmaking/status com publicLimiter.

Testes rápidos (PowerShell)

1) Healthcheck:

```
Invoke-WebRequest -UseBasicParsing -Uri "https://riftplay-backend.onrender.com/health"
```

2) Registro → Login → /api/me:

```
$body=@{nickname="user123";email="user123@example.com";senha="123456";cpf="12345678900"}|ConvertTo-J
Invoke-RestMethod -Method Post -Uri "https://riftplay-backend.onrender.com/api/register" -ContentTyp
$creds=@{email="user123@example.com";senha="123456"}|ConvertTo-Json
$resp=Invoke-RestMethod -Method Post -Uri "https://riftplay-backend.onrender.com/api/login" -Content
$token=$resp.token
Invoke-RestMethod -Method Get -Uri "https://riftplay-backend.onrender.com/api/me" -Headers @{Authori
```

3) Rate-limit – Sem token (se a rota exigir auth voltará 401):

```
for($i=1;$i -le 12;$i++){try{$r=Invoke-WebRequest -UseBasicParsing -Method GET -Uri "https://riftpla
```

3B) Rate-limit – Com token:

```
$creds=@{email="user123@example.com";senha="123456"}|ConvertTo-Json
$resp=Invoke-RestMethod -Method Post -Uri "https://riftplay-backend.onrender.com/api/login" -Content
$token=$resp.token
for($i=1;$i -le 12;$i++){try{$r=Invoke-WebRequest -UseBasicParsing -Method GET -Uri "https://riftpla
```

Checklist de Diagnóstico

• 401 antes de 429: rota protegida; use login e Bearer.
• Para ver 429 rápido: RATE_LIMIT_MAX=5 e RATE_LIMIT_WINDOW_MS=60000 → redeploy → rodar script com token.
• Observe headers RateLimit-Remaining/Reset.

Blackbox Pro e Próximos passos

Blackbox Pro — como conduzir

1) Manual ON e Add Context: backend/ completo (server.js, routes/, controllers/, middleware/, models/, config/, package.js
2) Mensagem: “Bloco 1 – Segurança. Diagnóstico rápido → patch em unified diff → comandos de teste”.
3) Aplicar patch, rodar testes. Depois Bloco 2 (DB), Bloco 3 (Rotas), Bloco 4 (Erros).

Passos para o Frontend (depois)

• API_URL do front → https://riftplay-backend.onrender.com
• Implementar chamadas (login, /api/me, matchmaking/status) + estados de loading/erro.
• Build e deploy (Vercel/Netlify ou estático no Render).

Erros comuns & Dicas

• Envs digitados errado (ex.: RATE_LIMIT_WINDO). Revise.
• Testar rota protegida sem token → 401 (não é bug de rate-limit).
• Sempre aguarde redeploy após mudar envs.
• Em dev, publicLimiter só aplica em NODE_ENV=production.