

# ChatGPT - Hero

Esta é uma cópia de uma conversa entre ChatGPT e Anônimo.



Crie um arquivo md com uma checklist com uma lista de checagem para publicação de uma api, adicione todas as verificações necessárias para que o projeto seja auditado de modo que se torne seguro a publicação desta api de nível industrial.

Claro! Aqui está um arquivo `API_PUBLISH_CHECKLIST.md` com uma checklist completa para auditoria e verificação de uma API antes da publicação em ambiente de produção. O foco é garantir segurança, performance, conformidade e estabilidade para nível industrial:

markdown

## # Checklist de Publicação de API (Nível Industrial)

Este documento fornece uma lista completa de verificação para garantir que uma API esteja pronta para ser publicada com segurança e robustez em ambientes de produção.

---

## ## Segurança

- [ ] **\*\*Autenticação forte implementada\*\*** (ex: OAuth2, JWT, API Key, etc)
- [ ] **\*\*Autorização por escopo e RBAC/ABAC\*\***
- [ ] **\*\*Proteção contra ataques XSS, CSRF, SSRF, RCE, SQLi e DoS\*\***
- [ ] **\*\*Rate limiting e throttling aplicados\*\***
- [ ] **\*\*CORS configurado corretamente\*\***
- [ ] **\*\*HTTPS obrigatório e certificado válido\*\***
- [ ] **\*\*Headers de segurança configurados** (`helmet`, `strict-transport-security`, etc)
- [ ] **\*\*Logs de acesso e segurança armazenados com integridade\*\***
- [ ] **\*\*Validação e sanitização de entradas com esquemas tipados** (Zod, Joi, etc)
- [ ] **\*\*Exposição controlada de erros** (sem stack traces nem dados sensíveis)
- [ ] **\*\*Revalidação de tokens e revogação implementadas** (blacklist/rotatividade)
- [ ] **\*\*Mecanismos de proteção a brute-force e bots\*\***
- [ ] **\*\*Verificação de dependências com falhas conhecidas** (ex: `npm audit`, `snyk`)
- [ ] **\*\*Variáveis de ambiente sensíveis protegidas e não versionadas\*\***
- [ ] **\*\*Backups e políticas de recuperação definidos\*\***

---

## ## Arquitetura e Infraestrutura

- [ ] **\*\*Estrutura modular e desacoplada** (por domínios ou contexto)
- [ ] **\*\*Separação clara entre Core, Módulos, Plugins e Entryoints\*\***
- [ ] **\*\*CI/CD implementado com deploy automático seguro\*\***
- [ ] **\*\*Infraestrutura como código** (IaC) versionada (ex: Terraform, Pulumi)
- [ ] **\*\*Build reproduzível** (sem side-effects)
- [ ] **\*\*Health-checks para HTTP, WebSocket e DB\*\***
- [ ] **\*\*Ambientes configurados corretamente** (dev, staging, prod)
- [ ] **\*\*Monitoramento ativo** (ex: Prometheus, Grafana, Datadog, etc)
- [ ] **\*\*Alertas configurados para falhas críticas\*\***
- [ ] **\*\*Versão semântica e changelog atualizado\*\***
- [ ] **\*\*Sistema de log centralizado** (ELK, Loki, etc)
- [ ] **\*\*Orquestração** (Docker, K8s, etc) documentada e estável

---

## ## Performance e Escalabilidade

- [ ] **\*\*Cache implementado** (HTTP, Redis, etc)
- [ ] **\*\*Consultas otimizadas e índices no banco revisados\*\***

```
- [ ] **Uso de CDN para conteúdos estáticos**
- [ ] **Compressão de payload (gzip, brotli) ativada**
- [ ] **Testes de estresse e carga realizados**
- [ ] **Fila de tarefas assíncronas (ex: BullMQ, RabbitMQ) usada quando necessário**
- [ ] **Limites e paginação em listas**
- [ ] **Lazy loading e streaming implementado onde necessário**
- [ ] **Uso de workers, pools ou threads onde apropriado**

---
```

## ## 📦 Qualidade de Código

```
- [ ] **Cobertura de testes mínima garantida (>80%)**
- [ ] **Testes unitários, integrados e E2E automatizados**
- [ ] **Linting e formatação automatizados (eslint, prettier)**
- [ ] **Padrões e estilos de código definidos e aplicados**
- [ ] **Documentação de código (JSDoc ou TypeDoc)**
- [ ] **Sem warnings ou erros de compilação**
- [ ] **Sem commits com console.log, debugger ou `any` solto**
- [ ] **Typescript configurado com `strict: true`**
- [ ] **Estrutura de arquivos consistente e escalável**
- [ ] **Uso de base classes e injeção de dependência padronizado**

---
```

## ## 📖 Documentação

```
- [ ] **OpenAPI (Swagger) atualizado e acessível**
- [ ] **Exemplos de uso e payloads nos endpoints**
- [ ] **README.md atualizado com instruções de uso**
- [ ] **Política de versionamento explicada**
- [ ] **Política de depreciação documentada**
- [ ] **FAQs e troubleshooting básico**
- [ ] **Arquitetura e diagrama de componentes incluídos**
- [ ] **Licença e termos de uso informados**
- [ ] **Checklist de primeiros passos para devs**

---
```

## ## 📋 Conformidade e Legal

```
- [ ] **LGPD/GDPR aplicados (uso mínimo de dados pessoais)**
- [ ] **Política de privacidade publicada**
- [ ] **Logs com anonimização, se necessário**
- [ ] **Consentimento explícito para dados sensíveis**
- [ ] **Tempo de retenção de dados definido**
- [ ] **Auditorias e rastreabilidade de ações (logs por usuário, IP, etc)**

---
```

## ## 🏁 Finalização

```
- [ ] **Revisão manual e peer review dos PRs**
- [ ] **Branch protegida e releases assinadas/tagueadas**
- [ ] **Deploy canário ou gradual configurado**
- [ ] **Reversão rápida possível (rollback seguro)**
- [ ] **Tudo validado em staging idêntico ao ambiente de produção**

---
```

> ✅ Após todas as verificações acima, sua API estará pronta para publicação segura e confiável em ambiente de produção.

Se quiser, posso transformar isso em um script que valide parte dessas verificações automaticamente ou integrar à sua CLI da Hero API. Deseja isso?