

Global Standards



Princípios Fundamentais

Hierarquia de Prioridades

1. **SEGURANÇA:** Nunca comprometer segurança por velocidade
2. **QUALIDADE:** Código limpo e testado acima de funcionalidade rápida
3. **MANUTENIBILIDADE:** Arquitetura limpa e DDD como base
4. **PERFORMANCE:** Otimização sem sacrificar legibilidade
5. **ENTREGA:** Funcionalidade completa e documentada

Princípios SOLID + DDD

Single Responsibility Principle (SRP)

- Uma classe/função = uma responsabilidade
- Se precisa de “e” para descrever, está violando SRP
-  `class UserAuthenticator`
-  `class UserAuthenticatorAndValidator`

Open/Closed Principle (OCP)

- Aberto para extensão, fechado para modificação
- Usar interfaces e composition
- Exemplo: Strategy pattern para payment processors

Liskov Substitution Principle (LSP)

- Subtipos substituíveis pelos tipos base
- Testes devem passar com qualquer implementação
- Todas as implementações de Repository devem ser intercambiáveis

Interface Segregation Principle (ISP)

- Interfaces específicas > interfaces gerais
- Múltiplas interfaces pequenas
- `IReadable` , `IWritable` ao invés de `IStorage`

Dependency Inversion Principle (DIP)

- Depender de abstrações, não concreções
- Injeção de dependência obrigatória
- `constructor(private readonly userRepo: IUserRepository)`

Padrões DDD

Value Objects

- Imutáveis e com validação interna
- Exemplos: `Email` , `Money` , `PhoneNumber`

Entities

- Identidade única e ciclo de vida claro
- Exemplos: `User` , `Order` , `Product`

Aggregates

- Consistência transacional e invariantes
- Exemplo: `Order` com `OrderItems`

Domain Services

- Operações que não pertencem a entidades
- Exemplos: `PricingService` , `TaxCalculator`

Repositories

- Abstração para persistência
- Exemplos: `IUserRepository` , `IOrderRepository`

Domain Events

- Comunicação assíncrona entre bounded contexts
- Exemplos: `OrderPlacedEvent` , `UserRegisteredEvent`

Gestão de Contexto e Tokens

Otimização de Contexto

- **Max Context Window:** 200k tokens
- **Optimal Usage:** < 100k tokens para manter velocidade

Hierarquia de Prioridade

1. **Priority 1** (2000 tokens): Current TODO item + acceptance criteria
2. **Priority 2** (3000 tokens): Directly related ADRs
3. **Priority 3** (5000 tokens): Direct dependencies (imports/exports)
4. **Priority 4** (3000 tokens): Test files for current module
5. **Priority 5** (2000 tokens): Related configuration files

Estratégias de Poda

- Remover código não relacionado à tarefa atual
- Comprimir discussões anteriores
- Usar aliases para paths longos
- Agrupar imports similares

Otimização de Tokens

- **Responses:** Max 500 tokens de explicação
- **Code Comments:** Apenas para lógica complexa
- **References:** Linkar docs ao invés de explicar
- **Diffs:** Unified diff para mudanças grandes

Controles Operacionais

Limites Patch-Only

- **Max Changes:** 5 arquivos, 200 linhas por arquivo, 500 linhas total
- **Forbidden:** Mass refactoring, bulk renaming, architecture changes
- **Safe Operations:** Bug fix, feature addition, test creation, documentation update

Exception Process

- **Trigger:** Need exceeds limits
- **Procedure:** Create refactoring ADR
- **Approval:** Tech lead required
- **Execution:** Phased approach mandatory

Referências

- @ref:naming-conventions
- @ref:error-handling
- @docs:<https://docs.cursor.com/en/context/rules>