

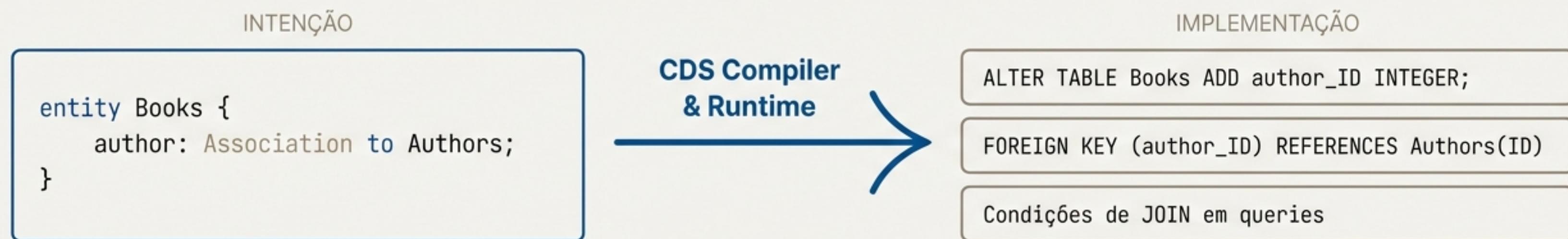
Desvendando a Modelagem de Domínio com SAP CAP

Transformando a Intenção de Negócio em
Aplicações Robustas com CDS

Baseado no CAP Cookbook 2: Domain Modeling.

O Princípio Fundamental: Foco na Intenção

Capture a Intenção, Não a Implementação. (O Quê, não Como).



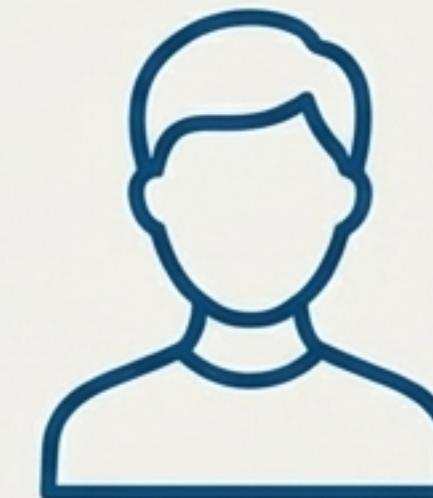
- Modelos de domínio em CDS devem ser **concisos, comprehensíveis por humanos e focados no que o negócio precisa.**
- A complexidade da implementação (o 'como') é delegada a **implementações genéricas e otimizadas** fornecidas pelo framework CAP.
- Isso não só mantém os modelos limpos, mas também permite que o CAP forneça as melhores implementações possíveis para cada cenário.

Nosso Ponto de Partida: Uma Livraria

Queremos criar uma livraria que permita aos usuários navegar por **Livros** e **Autores**.

Os usuários devem poder navegar de Livros para seus Autores e vice-versa.

Os livros são classificados por **Gênero**.



O Primeiro Rascunho: Traduzindo Requisitos em Entidades

O primeiro passo é mapear os substantivos do nosso domínio para entidades no CDS. Definimos as estruturas de dados essenciais.

```
// No arquivo db/schema.cds
using { cuid } from '@sap/cds/common';

entity Books : cuid {
    title : String;
    descr : String;
    genre : Genre;
    // Como conectar ao autor?
}

entity Authors : cuid {
    name : String;
}

type Genre : String enum {
    Mystery; Fiction; Drama;
}
```

Criando Conexões com Associações Gerenciadas

O Elo Perdido: "Como um Livro se relaciona com seu Autor?"

Em vez de gerenciar chaves estrangeiras manualmente, declaramos a relação conceitual. O CAP cuida dos detalhes da implementação, como a criação da coluna author_ID e a gestão das condições de junção (join).

Antes

```
// No arquivo db/schema.cds
using { cuid } from '@sap/cds/common';

entity Books : cuid {
    title : String;
    descr : String;
    genre : Genre;
    // Como conectar ao autor?
}

entity Authors : cuid {
    name : String;
}
```

Depois

```
// Modelo Evoluído
entity Books : cuid {
    title : String;
    descr : String;
    genre : Genre;
    author : Association to Authors;
}

entity Authors : cuid {
    name : String;
    books : Association to many Books on books.author = $self;
}
```

Uma linha para definir
uma relação complexa.
Esta é a 'intenção'.

O Poder dos Aspectos: Resolvendo Problemas Comuns

Aspectos são blocos de construção reutilizáveis que encapsulam funcionalidades comuns. Eles mantêm nossos modelos de domínio limpos e focados.

O Jeito Manual

Problema: Preciso de chaves primárias únicas e universais e campos de auditoria.

```
entity Books {  
    key ID      : UUID;        // Manual  
    createdAt   : Timestamp;   // Manual  
    createdBy   : User;       // Manual  
    modifiedAt  : Timestamp;   // Manual  
    modifiedBy  : User;       // Manual  
    title       : String;  
    // ...  
}
```

Com Aspectos CAP

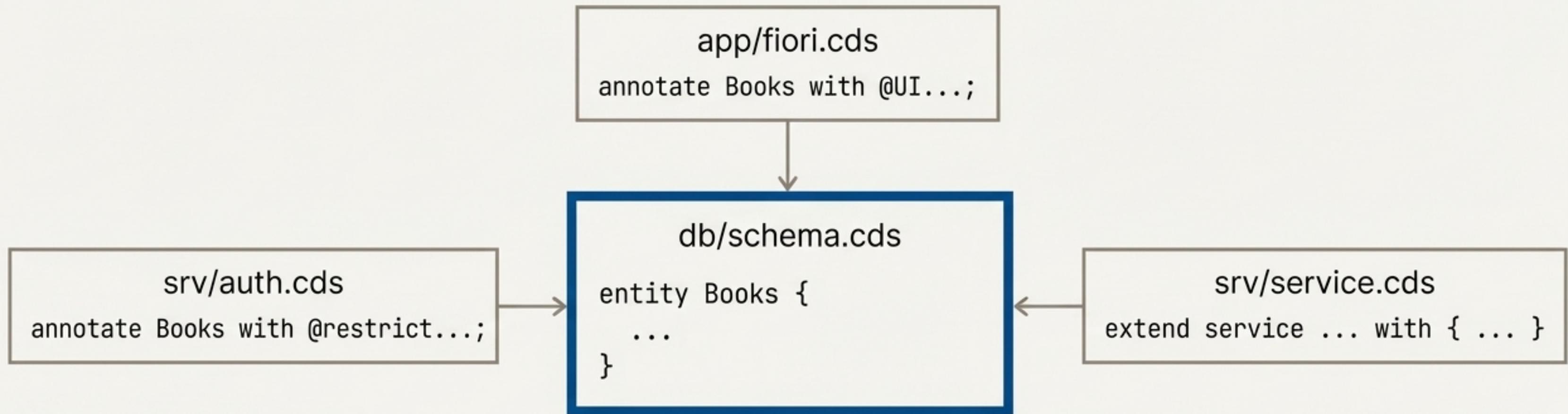
Solução: Use `cuid` para chaves e `managed` para auditoria.

```
using { cuid, managed } from '@sap/cds/common';  
  
entity Books : cuid, managed {  
    title      : String;  
    // ...  
}
```

Uma única linha (`:cuid, managed`) substitui cinco linhas de código repetitivo e propenso a erros, além de adicionar comportamento automático.

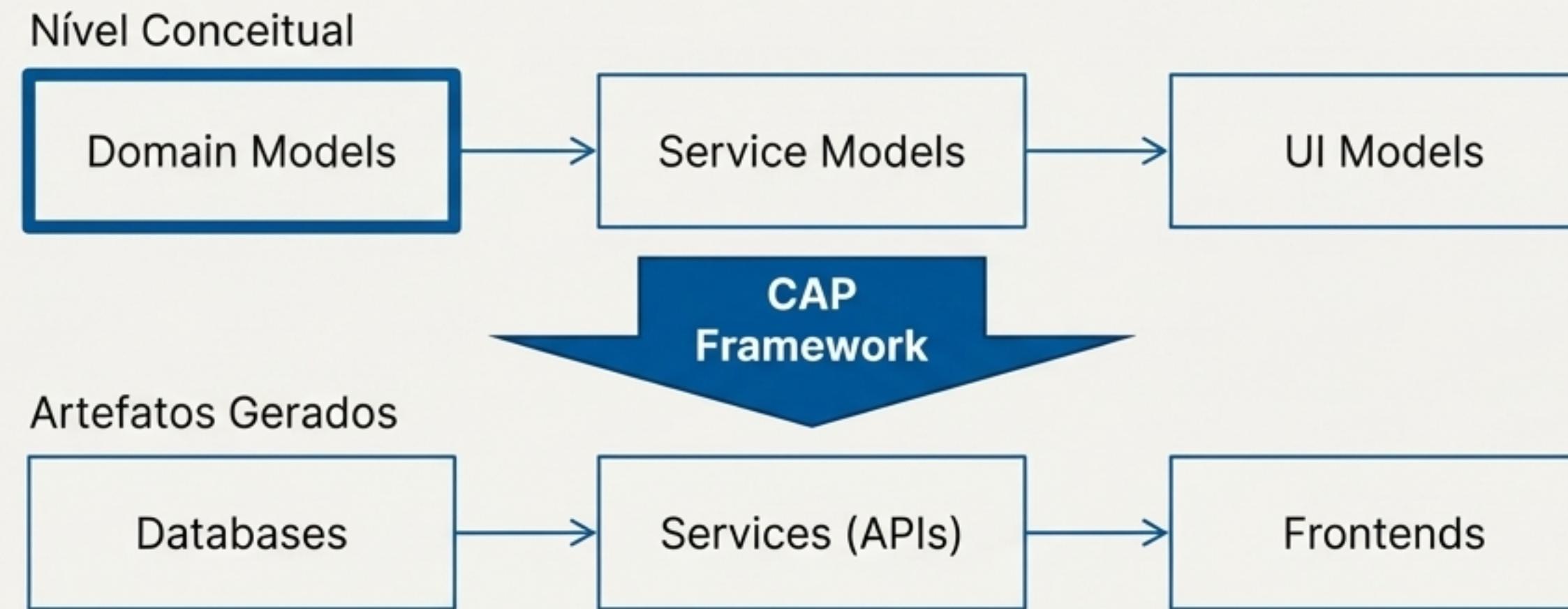
Arquitetura Limpa com Separação de Responsabilidades

Aspectos são a chave para decompor modelos. O domínio principal permanece puro, enquanto preocupações secundárias como anotações de UI, autorização e extensões de serviço são aplicadas em arquivos separados. Isso melhora a manutenibilidade e a colaboração.



O modelo de domínio principal fica intocado. Outras camadas adicionam seus próprios metadados e comportamentos de forma não invasiva.

O Modelo de Domínio como a Fonte da Verdade



O modelo de domínio em CDS não é apenas documentação. É um artefato executável que o framework CAP utiliza para gerar automaticamente a persistência do banco de dados, as definições de serviço base e as anotações para a UI.

*“O modelo que você define é o modelo que você executa.
Menos lacunas, menos código de cola, mais consistência.”*

Aprofundando: Entidades e Chaves Primárias

Vamos formalizar os blocos de construção.

Entidades

Estruturas com elementos nomeados e tipados que possuem uma identidade única.

```
entity Books : cuid {  
    title : String;  
    stock : Integer;  
}
```

Chaves Primárias

Elementos marcados com a palavra-chave `key` que identificam unicamente uma instância da entidade.

Melhores Práticas

- **Prefira chaves simples e técnicas:** Evite chaves compostas por múltiplos campos de negócio.
- **Use chaves canônicas:** key ID: UUID; é o padrão ouro. O aspecto cuidado de @sap/cds/common simplifica isso.
- **Imutabilidade:** Chaves primárias nunca devem mudar após a criação.

```
using { cuid } from '@sap/cds/common';  
entity Authors : cuid { ... }
```

Aprofundando: A Cola do Modelo - Associação vs. Composição

A escolha entre `Association` e `Composition` define o ciclo de vida e a posse dos dados, impactando diretamente o comportamento da aplicação (ex: exclusões em cascata).

Associação

Conceito: Uma relação entre entidades com ciclos de vida independentes. Uma pode existir sem a outra.

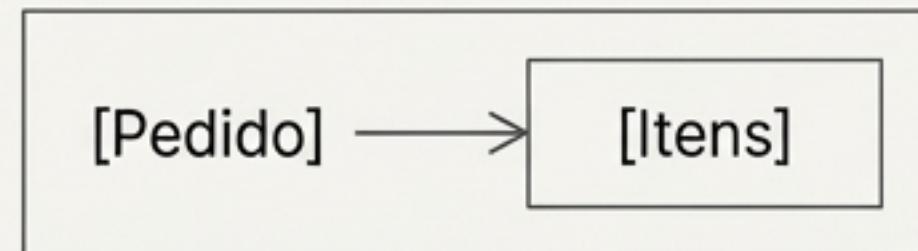
Exemplo: Um `Livro` está **associado** a um `Autor`. Se o livro for deletado, o autor continua existindo.



Composição

Conceito: Uma relação de posse. As partes pertencem ao todo e não existem sem ele.

Exemplo: Os `Itens` de um `Pedido` são uma **composição** do pedido. Se o pedido for deletado, seus itens também são.



Seu braço é uma **composição** do seu corpo. Seu celular é uma **associação** com você, pois amanhã ele pode pertencer a outra pessoa.

Criando Modelos Elegantes: Melhores Práticas de um Mestre

Dicas do *Domain Modeling Cookbook* para criar modelos que são robustos, fáceis de manter e compreender.

KISS (Keep It Simple, Stupid)

Prefira modelos planos e evite estruturas aninhadas desnecessariamente.

 **Bom:** entity Contacts { firstname: String; lastname: String; }

 **Ruim:** entity Contacts { personData: { name: { first: String; last: String; } } }

Convenções de Nomenclatura

- Nomes de entidades em Plural e Capitalizados: `entity Books`.
- Nomes de elementos em singular e minúsculos: `element title`.

Nomes Concisos

- Evite redundância: Use `Authors.name` em vez de `Authors.authorName`.
- Prefira uma palavra: `address` em vez de `addressInformation`.

Nosso Modelo Final: Juntando Todas as Peças

Este é o nosso modelo de domínio da livraria, aplicando todos os princípios: entidades, tipos, associações, aspectos e melhores práticas.

(1) Aspectos: Entidades com chaves canônicas (`cuid`) e auditoria (`managed`).

(2) Localização: Elementos marcados como `localized` para internacionalização.

(3) Associação: Relação gerenciada para uma conexão limpa entre `Books` e `Authors`.

(4) Backlink: Navegação bidirecional, de `Authors` de volta para `Books`.

(5) Reuso: Entidade simples (`Genres`) usada como uma lista de códigos.

```
// db/schema.cds
namespace my.bookshop;
using { cuid, managed } from '@sap/cds/common';

// (1)
entity Books : cuid, managed {
    // (2)
    title : localized String(111);
    descr : localized String(1111);
    // (3)
    author : Association to Authors;
    stock : Integer;
    price : Decimal(9,2);
    genre : Association to Genres;
}

entity Authors : cuid, managed {
    name : String(100);
    // (4)
    books : Association to many Books on books.author = $self;
}

// (5)
entity Genres : cuid {
    name : String;
    descr : String;
}
```

O Ecossistema em Ação: Onde o Domínio Encontra a Função

Um modelo de domínio robusto é o alicerce que simplifica drasticamente a implementação de funcionalidades complexas. O framework CAP utiliza os metadados do seu modelo para automatizar tarefas.



Tradução: A palavra-chave `localized` gera automaticamente tabelas `.texts` e lógica de fallback de idioma. (Fonte: Cookbook 10)

UI Fiori: Anotações `@UI` estendem o modelo para gerar UIs ricas e consistentes sem código de frontend. (Fonte: Cookbook 7)

Autorização: Anotações `@restrict` no modelo definem regras de acesso a dados que são aplicadas automaticamente. (Fonte: Cookbook 2)

Lições Essenciais da Modelagem de Domínio com CAP

Quatro princípios para levar com você em sua jornada de desenvolvimento com CAP.

1. Foco na Intenção, não na Implementação

Sempre se pergunte “O quê?” e não “Como?”. Deixe o framework otimizar. A concisão do seu modelo é um indicador de sucesso.

2. Aspectos são Seus Melhores Aliados

Use `cuid`, `managed` e crie seus próprios `aspectos` para eliminar código repetitivo, manter seu domínio principal limpo e garantir consistência.

3. O Domínio é a Única Fonte da Verdade

Trate seu modelo CDS como o coração da sua aplicação. Ele alimenta a base de dados, os serviços e a interface do usuário, garantindo alinhamento total.

4. Adote as Melhores Práticas desde o Início

Modelos planos, nomenclatura consistente e chaves canônicas não são apenas sugestões, são a base para aplicações escaláveis e de fácil manutenção.

O Modelo de Domínio é o Coração da Sua Aplicação



Investir tempo em um **modelo de domínio limpo, expressivo e focado na intenção** não é um luxo, é a **decisão arquitetônica mais impactante** que você pode tomar em um projeto CAP.

Ele **paga dividendos** em cada etapa do ciclo de vida do desenvolvimento, da persistência de dados à experiência do usuário final.

Pense no Domínio. O Resto se Seguirá.