# O que é Domain-Driven Design

* Abordagem/Guia para desenvolver softwares onde o foco é modelar e entender os problemas do mundo real, e transformar isso em código.
* É um conjunto de princípios, um guia. Não é uma arquitetura.
* Facilita uma possível expansão de projetos.
* Facilita manutenção.

## Duas Ideias Chaves do DDD

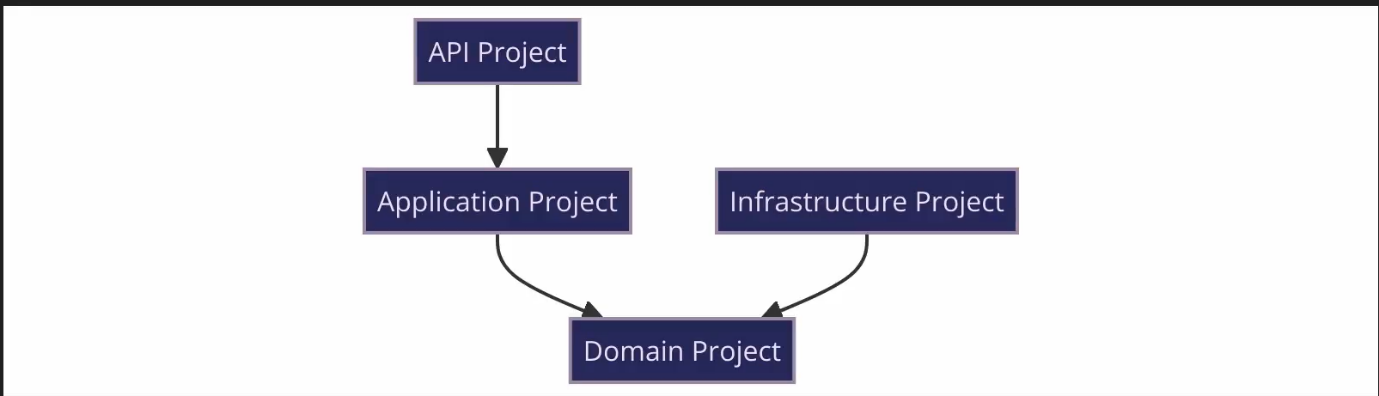
### Linguagem Ubíqua

* Garante uma linguagem comum a todos os envolvidos no projeto, sejam eles os Devs, PO, Cliente, etc.
* Visa eliminar ambiguidades e garantir que todos entendam os termos e conceitos do negócio da mesma forma.

### Contexto Delimitado

* O código deve ser dividido em áreas. Cada área terá regras e responsabilidades específicas.
* Permite que o projeto seja mais gerenciável a nível de código.

# Visão Geral de Projetos .NET com DDD



As setas entre os projetos significam: “Consegue enxergar / Pode utilizar”.

* O projeto de API recebe uma requisição e repassa os dados para o projeto de Application, onde estão implementadas as regras de negócio.
* A regra de negócio solicita implementações que sigam os contratos (Interfaces contidas no Domain) para realizar as operações, como registrar o usuário.
* O serviço de injeção de dependência fornece a implementação para a regra de negócio.
* A regra de negócio repassa a responsabilidade de registrar o usuário no banco para o projeto de Infrastructure.

## API Project

* Contém os Controllers e os endpoints.
* Enxerga os projetos de Application, Communication e Exception.

## Application Project

* Contém as regras de negócio.
* Validações de nomes, senhas e cadastros.
* Utiliza os contratos (Interfaces) em Domain.
* Enxerga os projetos de Domain, Communication e Exception.

## Infrastructure Project

* Contém os códigos que se comunicam com serviços externos.
* Responsável por banco de dados, envio de e-mails e integração com serviços de terceiros.
* Precisa de acesso ao Domain para implementar contratos e acessar entidades para persistência.

## Domain Project

* Contém entidades com propriedades fornecidas pelos usuários.
* Contém os contratos (Interfaces) que definem os métodos.
* Este projeto não precisa enxergar outros projetos.

# DDD Melhorado

No DDD melhorado, são adicionados dois projetos:

Diagrama

O conteúdo gerado por IA pode estar incorreto.

## Communication Project

* Evita duplicidade de código.
* Contém todas as Requests e Responses.
* Organizado com pastas, como “Request”, centralizando as definições.

## Exception Project

* Centraliza todas as exceções que precisam ser lançadas.
* Evita o uso excessivo de try e catch espalhados pelo código.
* Permite a criação de exceções customizadas e devolução de status codes corretos.

# IMPORTANTE

O projeto de API precisa enxergar o projeto de Infrastructure apenas para configurar as injeções de dependência.