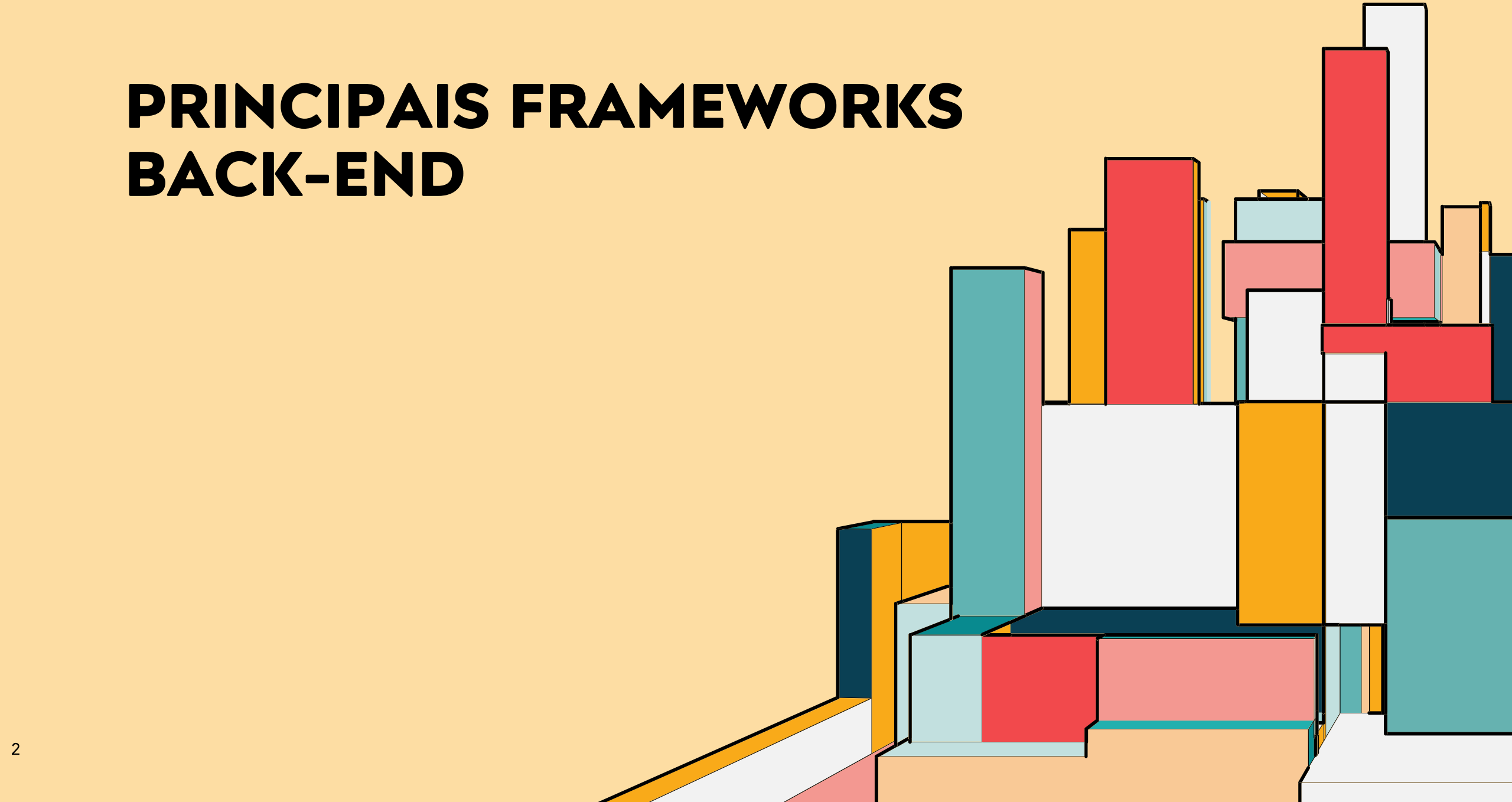
BACK-END FRAMEWORKS

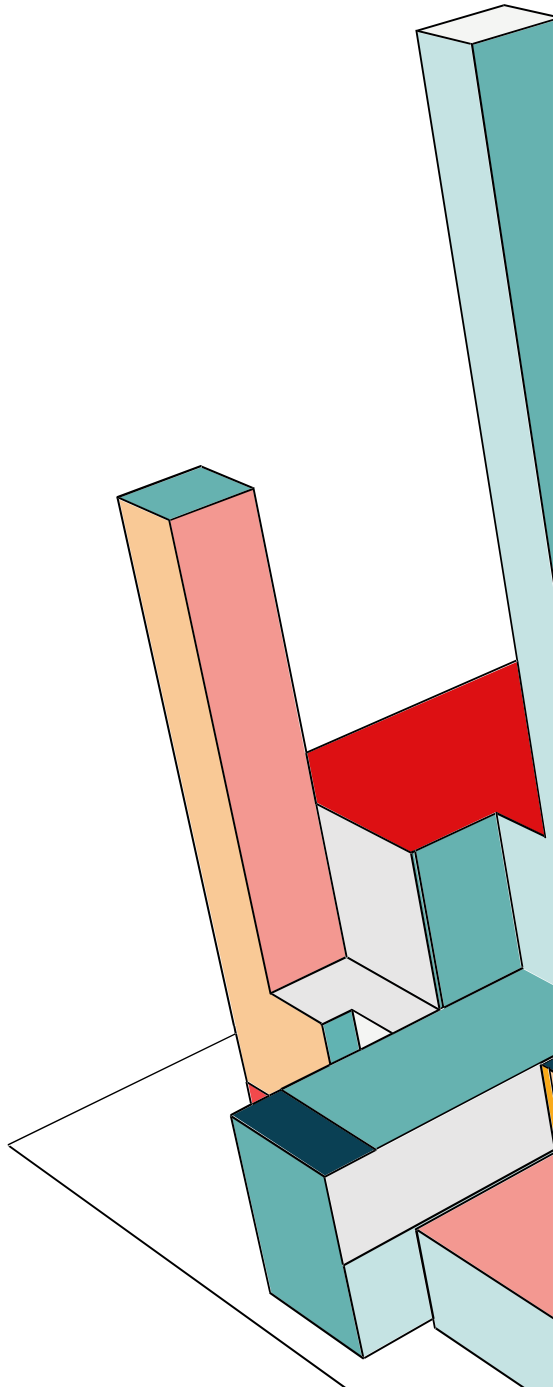
Prof. Jonas Bernardino

PRINCIPAIS FRAMEWORKS BACK-END



FRAMEWORKS POPULARES

- Spring Boot (Java)
- Django (Python)
- Express.js (Node.js)
- Laravel (PHP)



SPRING BOOT

- **Linguagem:** Java

- **Características:** Criação facilitada de APIs REST, suporte a microserviços, integração com JPA e Hibernate, fácil configuração inicial.

- JPA (Java Persistence API)
  - Especificação (as regras).
- Hibernate
  - Implementação (um dos motores que seguem essas regras).

DJANGO

- Linguagem: Python

- Características: Framework completo, ORM eficiente, administrador automático, segurança integrada, ideal para aplicações robustas.

EXPRESS.JS

- **Linguagem:** JavaScript (Node.js)

- **Características:** Minimalista, leve, flexível e altamente escalável. Amplamente usado em APIs e aplicações rápidas.

LARAVEL

- **Linguagem:** PHP

- **Características:** Sistema de rotas intuitivo, autenticação integrada, alta produtividade no desenvolvimento.

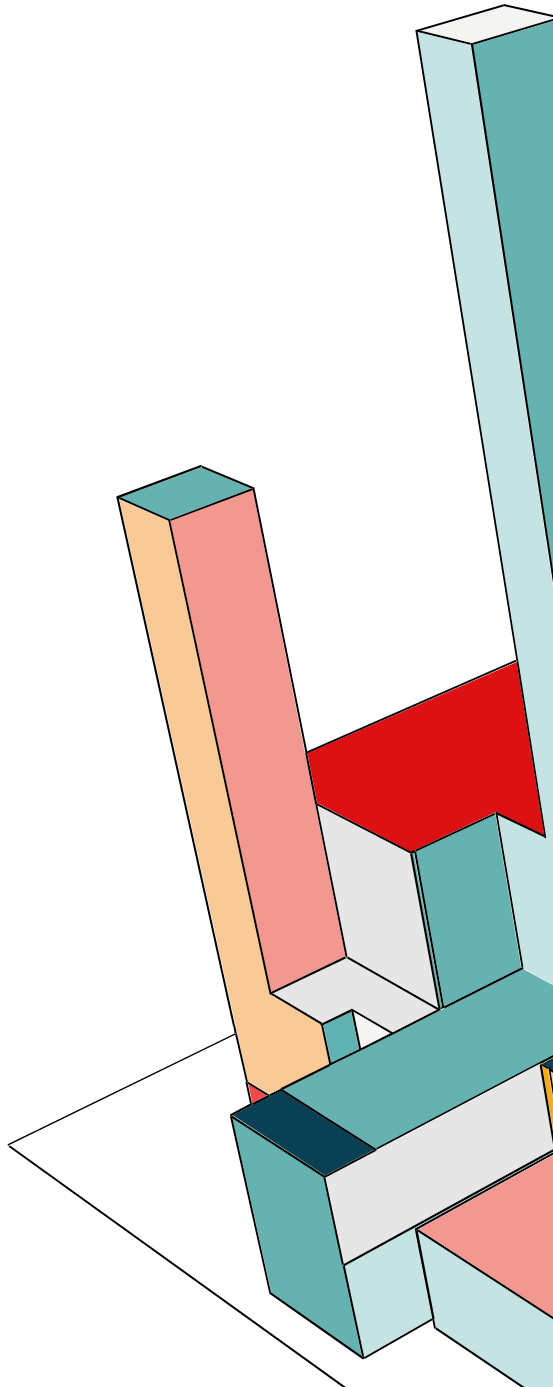
ARQUITETURA EM CAMADAS

INTRODUÇÃO À ARQUITETURA EM CAMADAS

- É uma forma de dividir uma aplicação em diferentes camadas com responsabilidades específicas.

- Garante maior organização, facilidade de manutenção e escalabilidade.

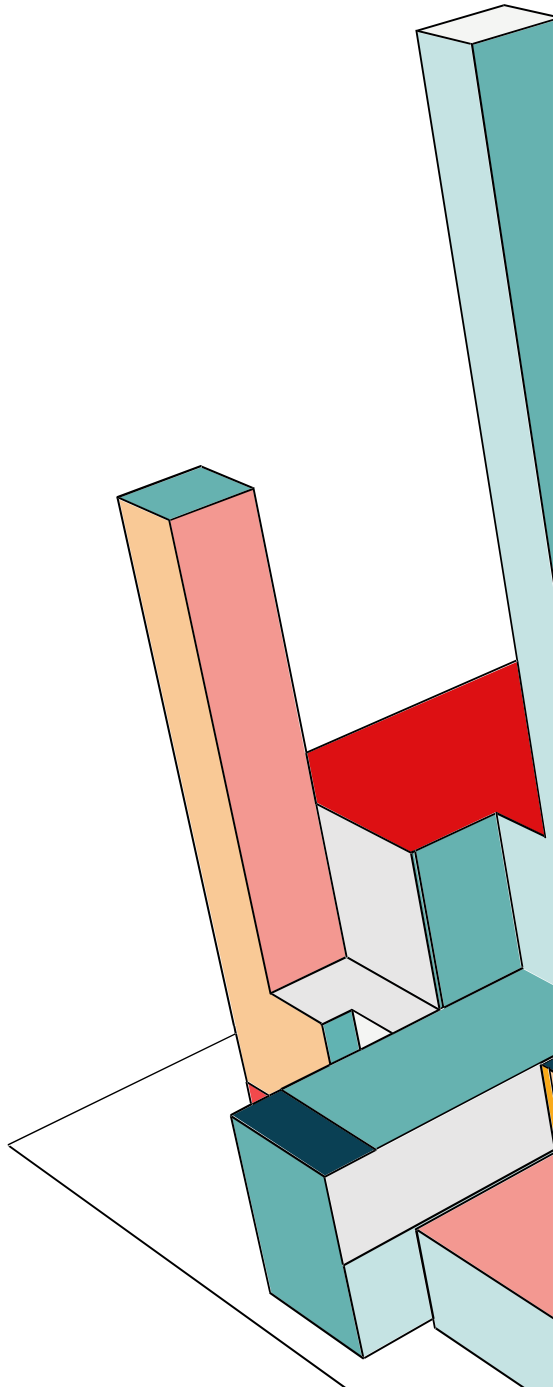
- Exemplo de camadas: Apresentação, Negócio e Dados.



CAMADA DE APRESENTAÇÃO

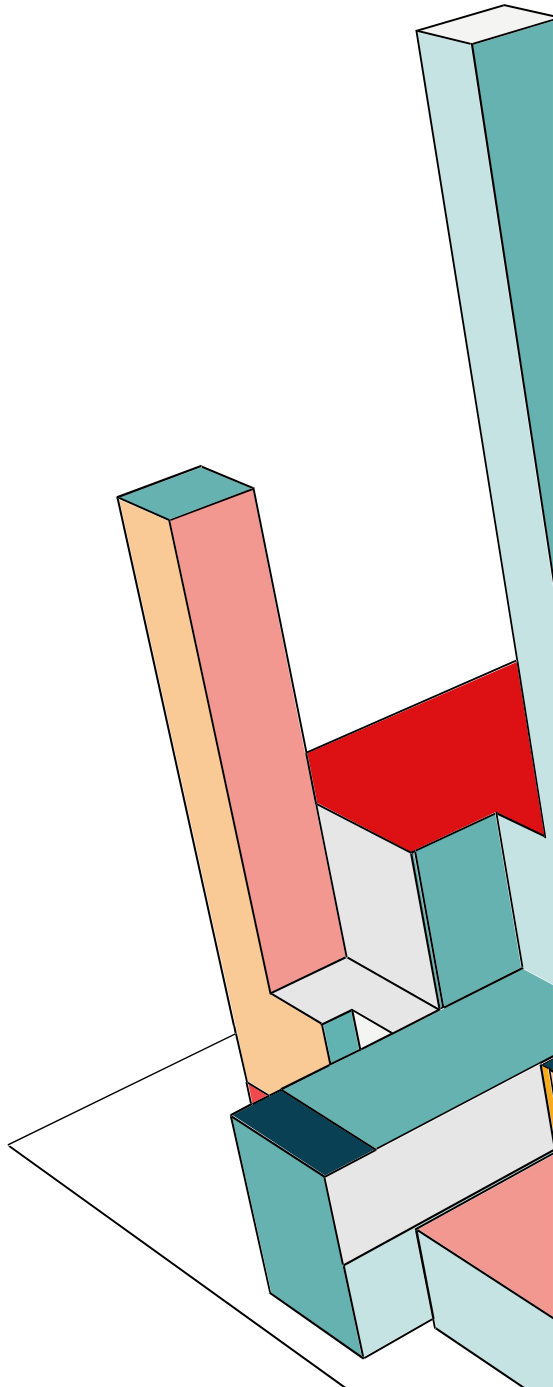
- Responsável pela interação direta com o usuário

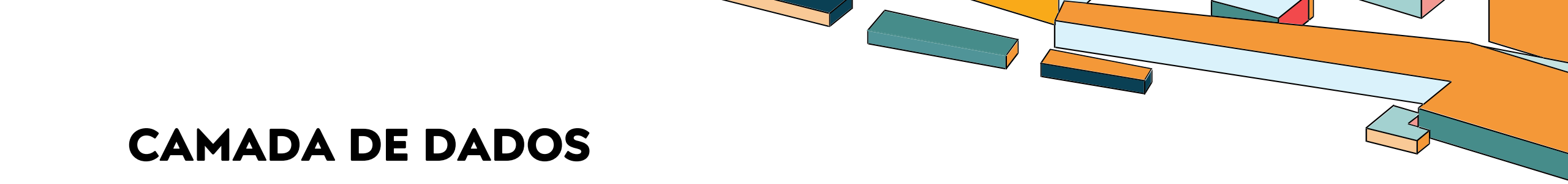
- Renderização de interfaces e coleta de informações dos usuários



CAMADA DE NEGÓCIO (LÓGICA)

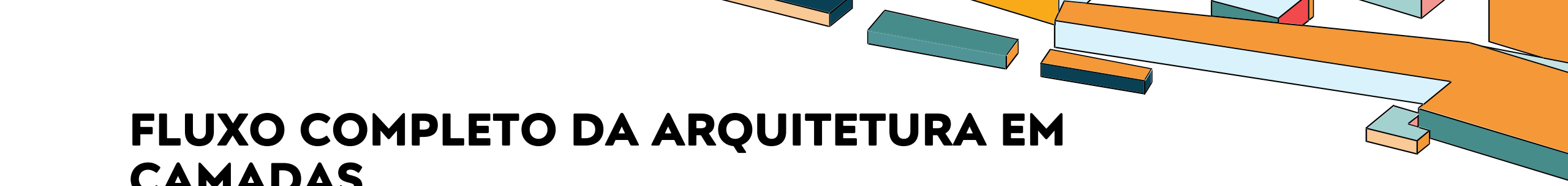
- Contém regras e validações da aplicação
- Responsável pelo processamento lógico das informações





CAMADA DE DADOS

- Responsável pela persistência dos dados
- Interação direta com o banco de dados
- Utiliza tecnologias como ORM ou consultas diretas ao banco (SQL)
- Exemplos: ORM (Hibernate, JPA)



FLUXO COMPLETO DA ARQUITETURA EM CAMADAS

Fluxo visual:

1.Cliente realiza uma requisição.

2.Camada de Apresentação recebe e encaminha ao Negócio.

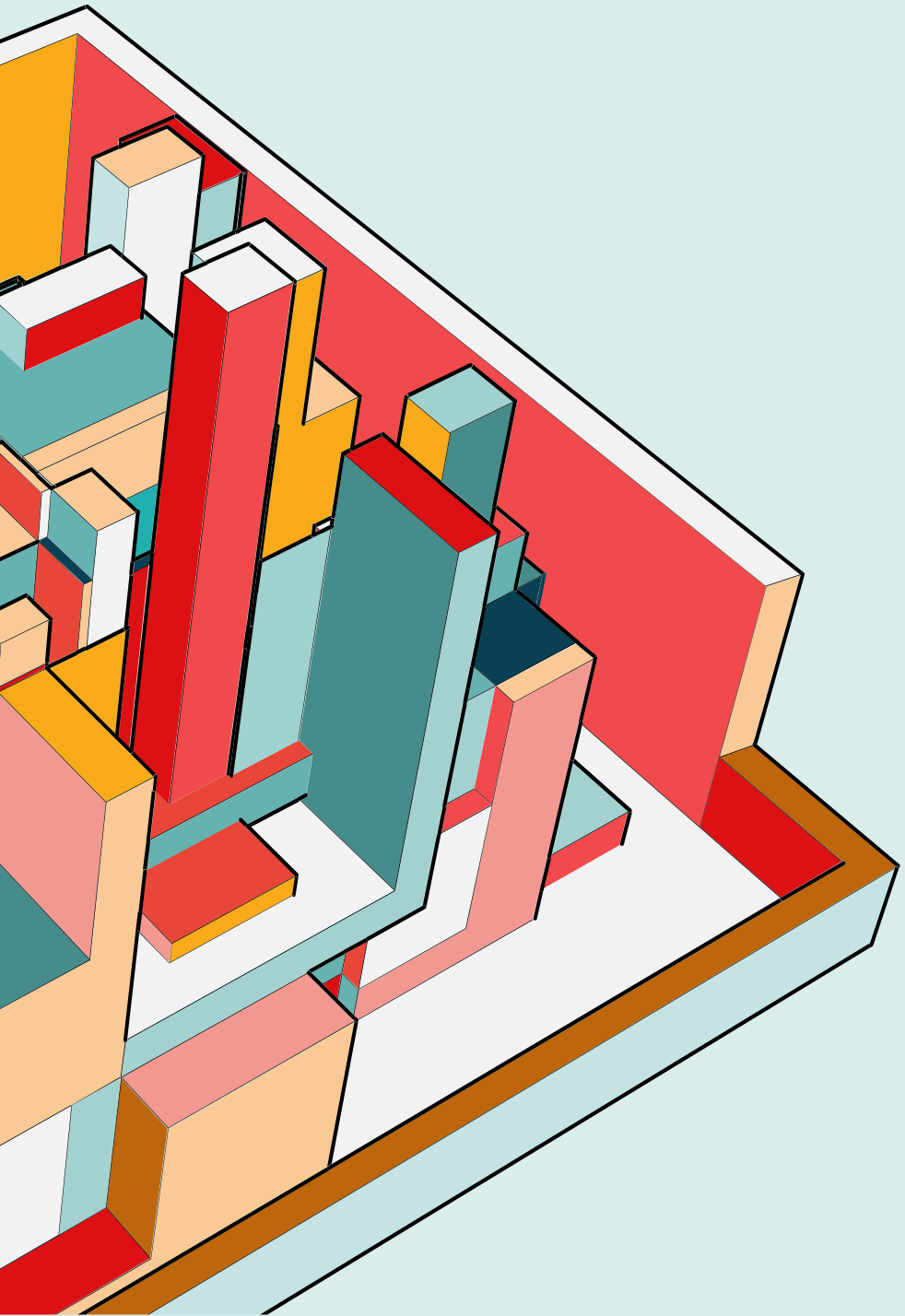
3.Negócio processa, valida e consulta a camada de Dados.

4.Dados acessa o banco e retorna informações.

5.Retorno chega até o Cliente via Apresentação.

APLICANDO ARQUITETURA EM CAMADAS COM FRAMEWORKS

- Em frameworks como Spring Boot:
  - **Controller (Apresentação)**
  - **Service (Negócio)**
  - **Repository (Dados)**

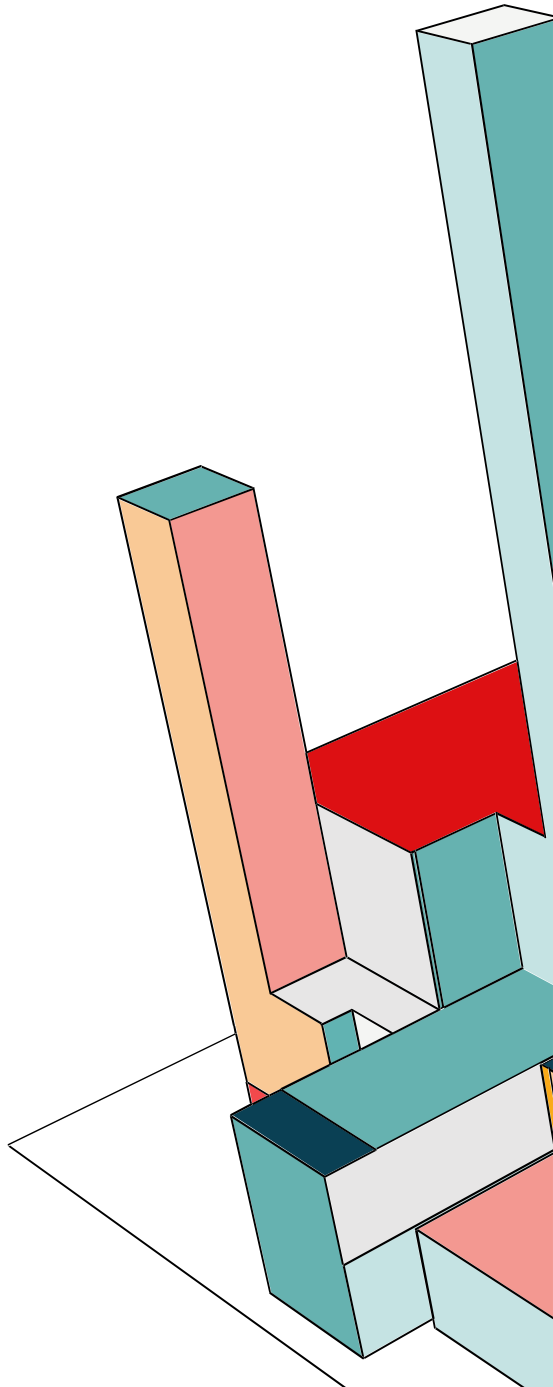


MODELO MVC

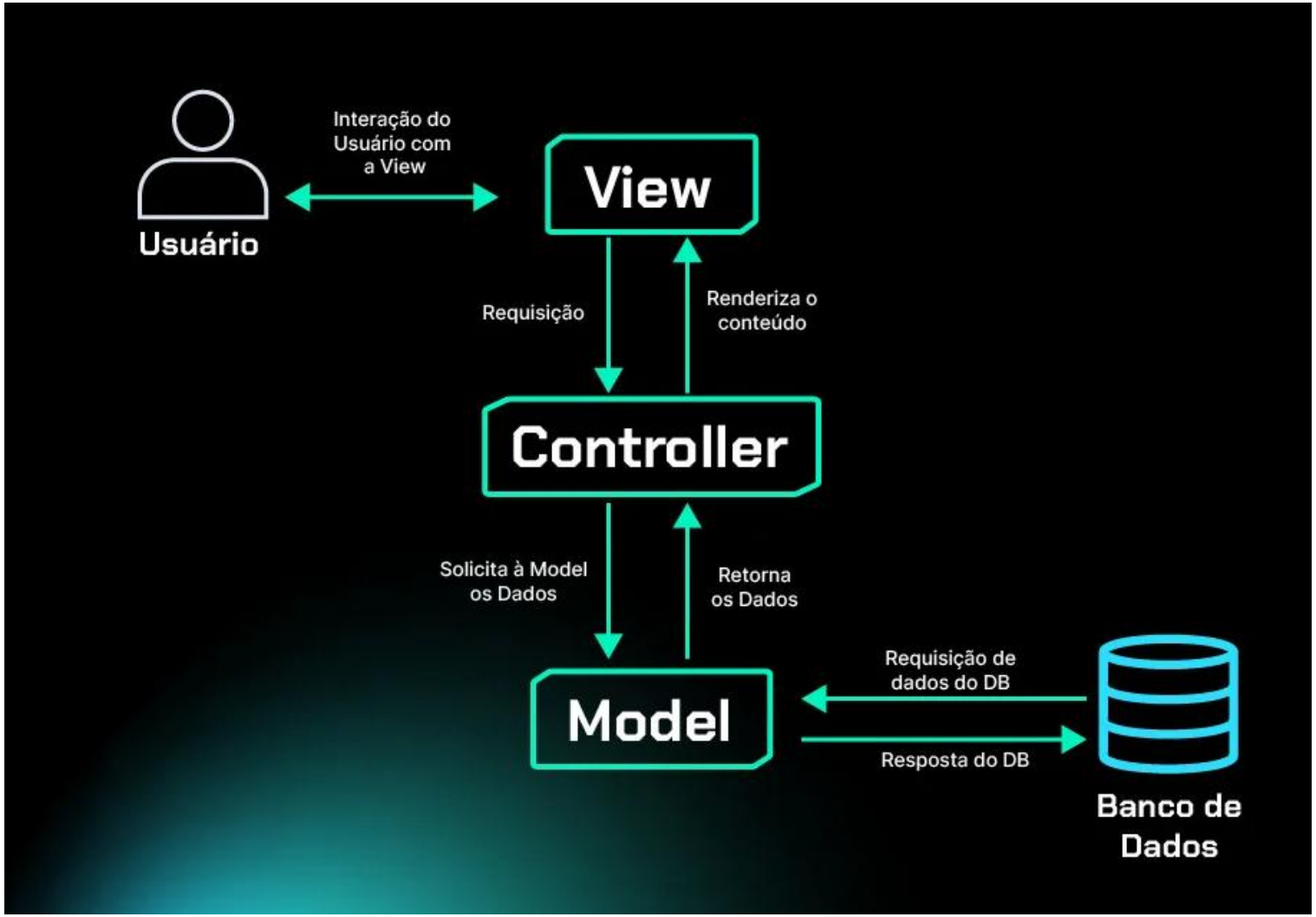
MVC – MODEL VIEW CONTROLLER

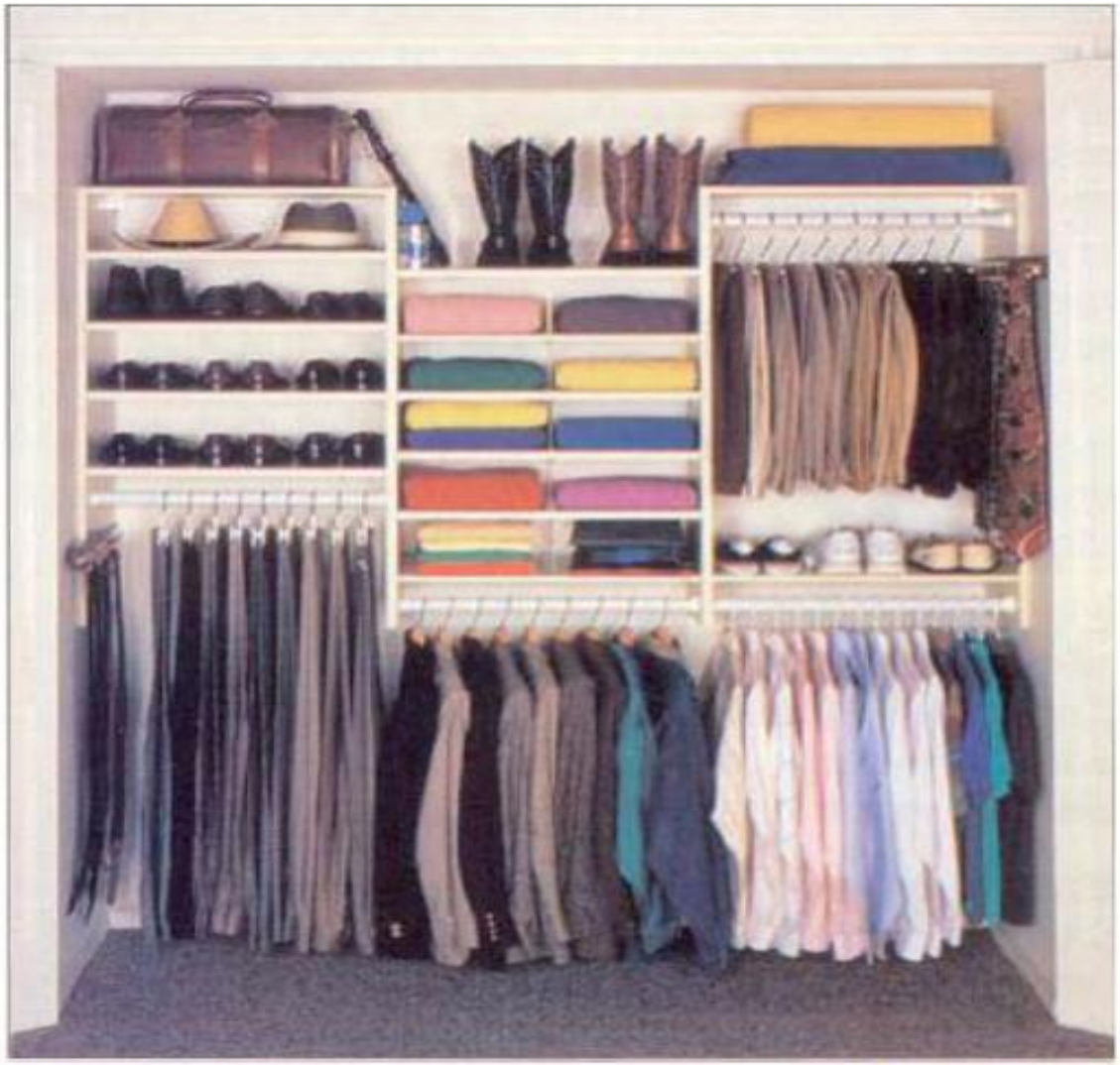
MVC significa **Model-View-Controller** (Modelo-Visão-Controlador), é um padrão arquitetural que separa uma aplicação em três componentes principais:

- **Model** (Modelo)
- **View** (Visão)
- **Controller** (Controlador)
- Este modelo ajuda na separação clara das responsabilidades dentro da aplicação.



MVC – MODEL VIEW CONTROLLER





ESTRUTURA GERAL DO MVC

- Model (Modelo):
  Camada responsável pela manipulação e gestão dos dados da aplicação.

- View (Visão): Responsável pela interface com o usuário. Ela exibe as informações e permite interação.

- Controller (Controlador): Faz a ligação entre as camadas Modelo e Visão, gerencia requisições e respostas, além de direcionar o fluxo da aplicação.

BENEFÍCIOS DA UTILIZAÇÃO DO MVC

- Separação de Responsabilidades
  Facilita manutenção, testes e desenvolvimento paralelo.

- Maior Produtividade
  Diferentes equipes podem trabalhar em camadas distintas simultaneamente.

- Reutilização de código
  Modelos ou controladores podem ser reaproveitados em diferentes partes da aplicação.

BACK-END FRAMEWORKS

Prof. Jonas Bernardino