UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

ES46A - Arquitetura de Software

Arquitetura em Camadas

DACOM – Engenharia de Computação
Diego Addan

UTFPR - 2023

Para hoje..

Aviso: Notas do projeto 1

Modelagem/Implementação

Arquitetura em camadas

Arquitetura de aplicações Web

Arquitetura em camadas

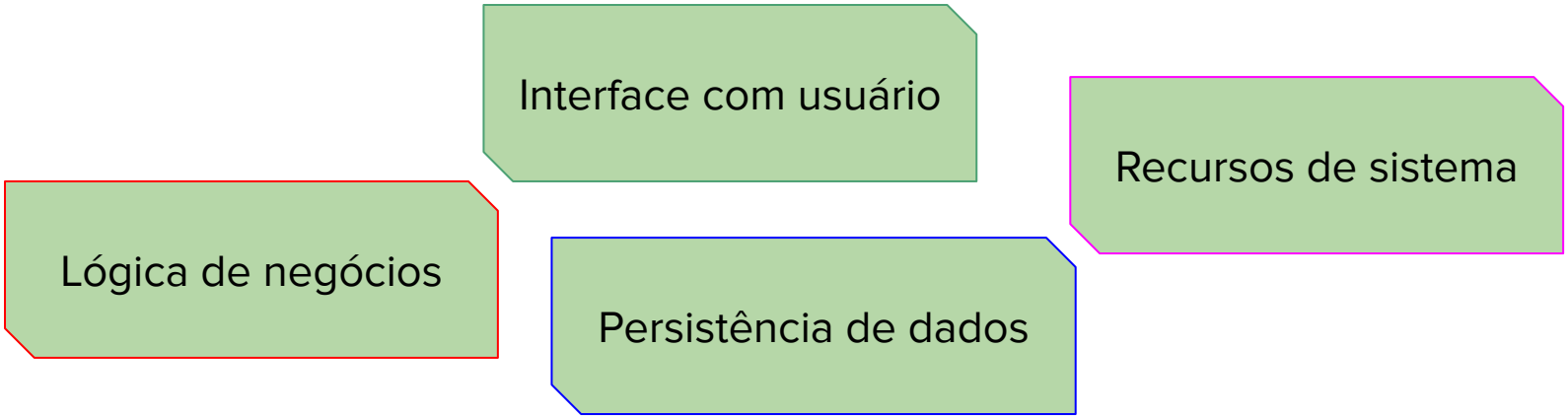
Recomendado que se adote uma arquitetura no projeto

- Manutenção
- Compreensão do projeto
- Sprints e entregas / Projeção do sistema

Pode variar muito do escopo e tipo de aplicação

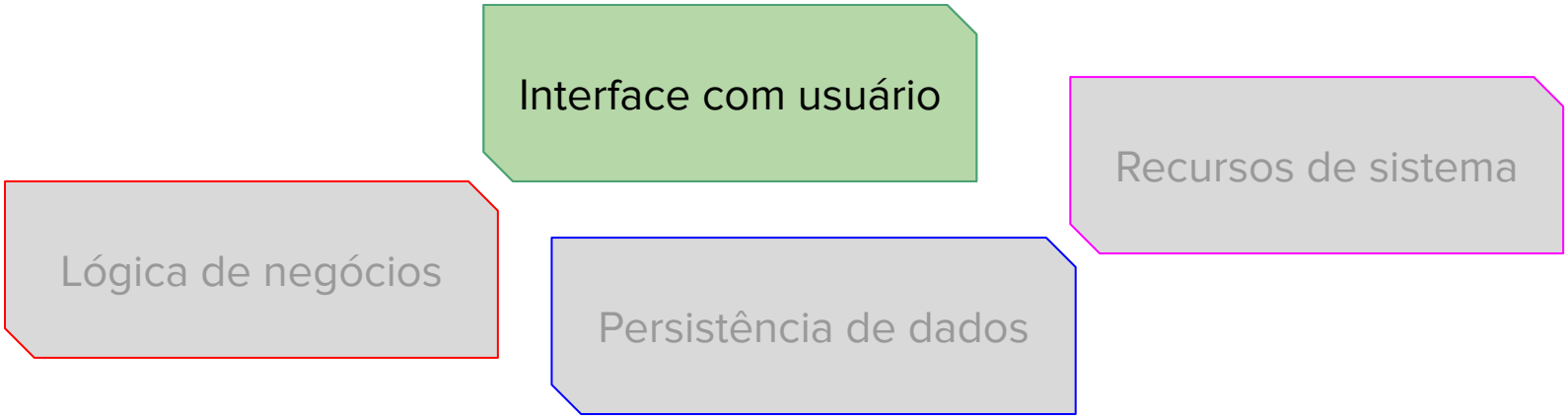
Arquitetura em camadas

Tem como base a organização de componentes lógicos por meio de camadas/categorias de funcionalidades



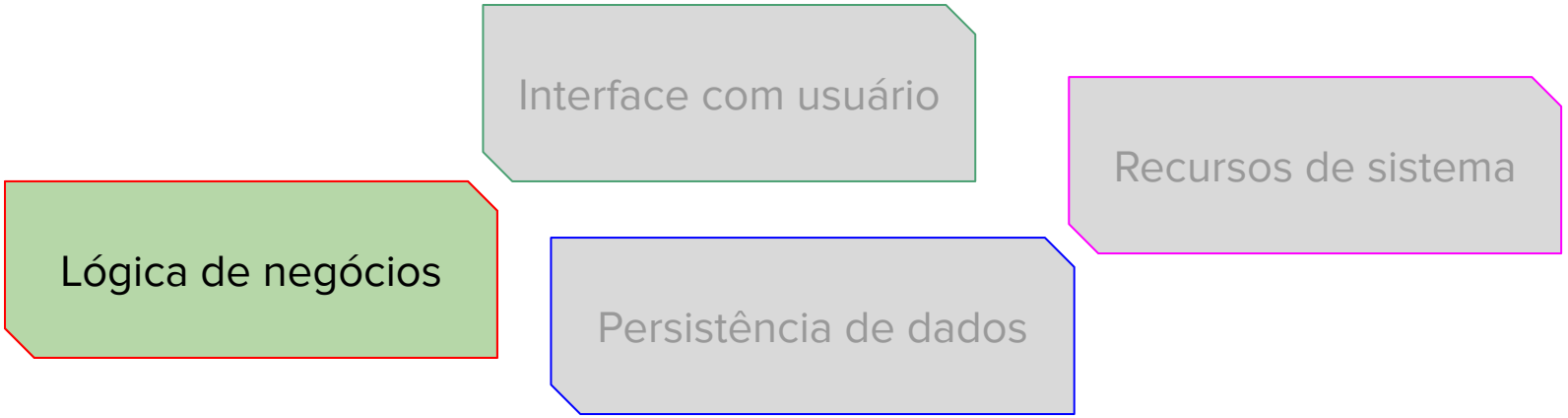
Arquitetura em camadas

Elementos de interação com o usuário: **UI, UX, View, I/O**



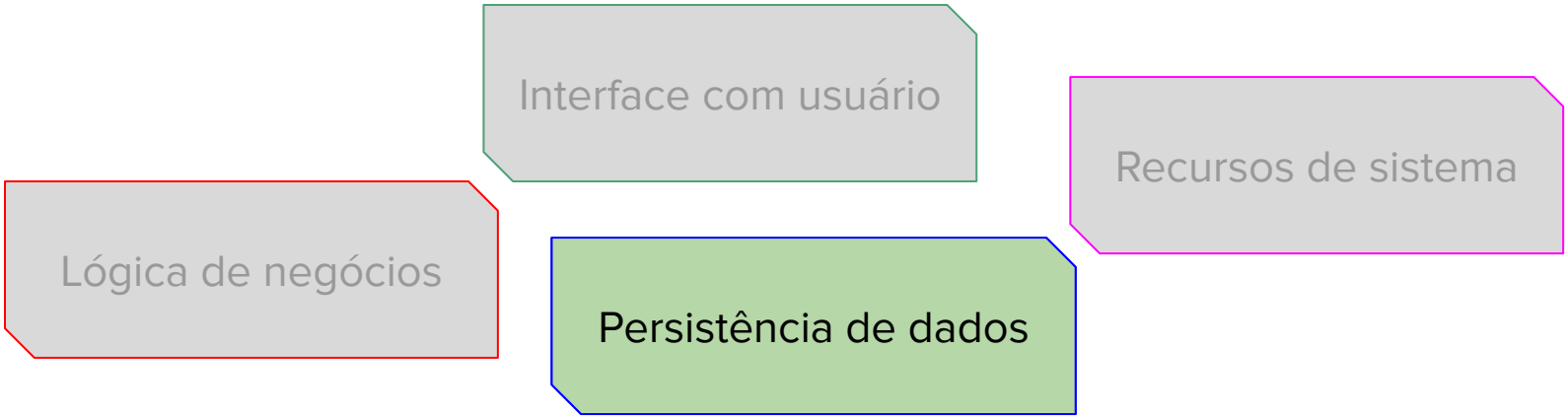
Arquitetura em camadas

Classes e componentes que implementam: **Regras, restrições, fórmulas, procedimentos de negócio**



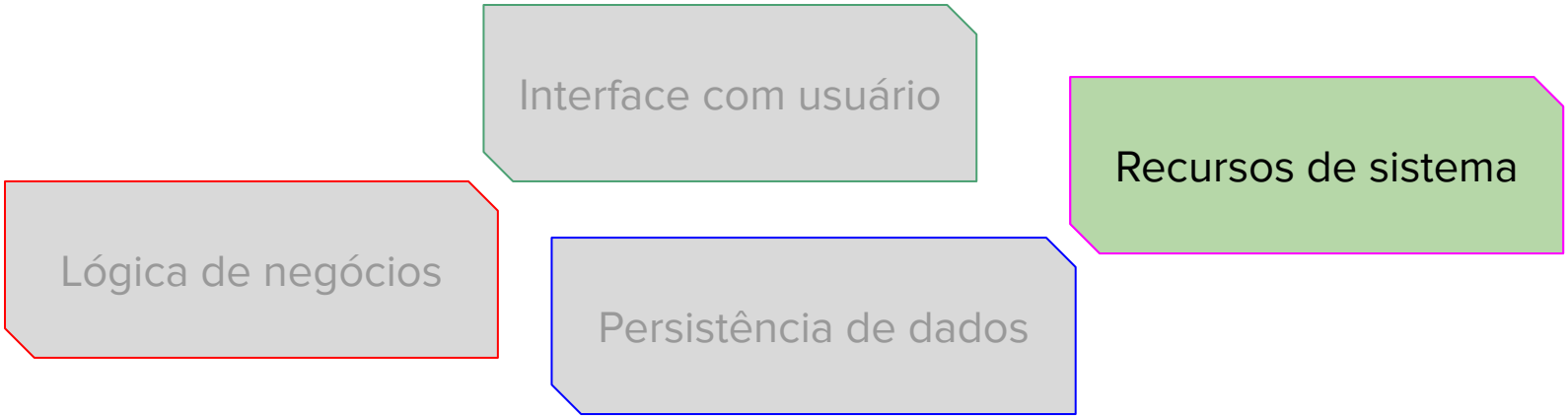
Arquitetura em camadas

Classes que permitem comunicação, armazenamento e manipulação de dados: **serviços, axios, sgbd, noSQL**



Arquitetura em camadas

Funções, ou módulos que viabilizam a comunicação de recursos distribuídos:
Rede, arquivos de configuração, comunicação entre aplicações

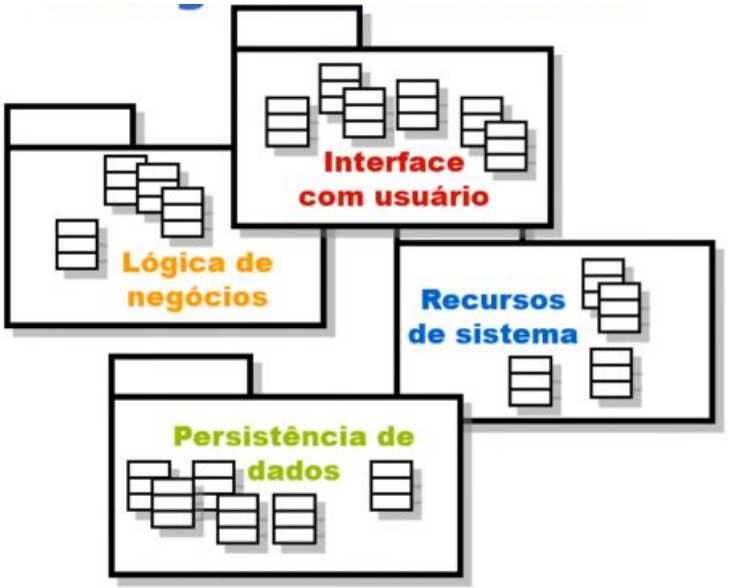


Arquitetura em camadas

Cada categoria é uma camada:

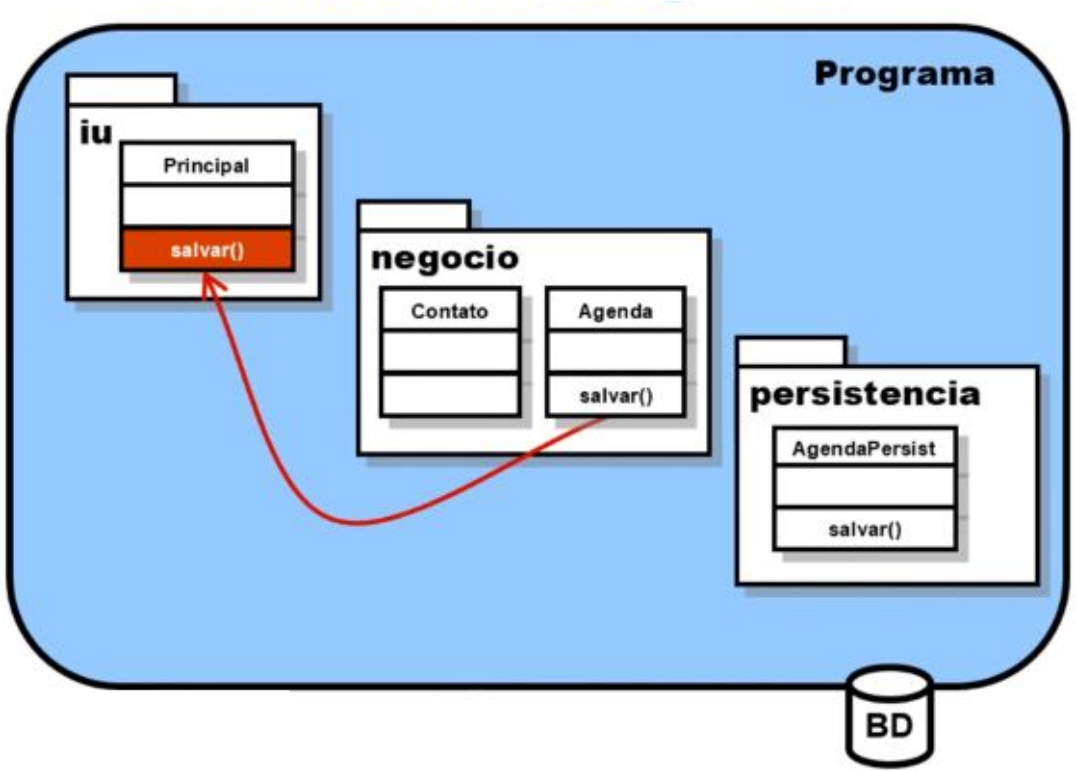
Estes artefatos são chamados de pacotes, (um pacote de classes, por exemplo, será Responsável pela UI)

As camadas interagem entre si através de processos



Arquitetura em camadas

A interação pode ser eventual



Arquitetura em camadas

```
package agenda.iu;

import agenda.negocio.Agenda;
import agenda.negocio.Contato;

public class Principal {

    // programa principal com telas, botões, etc.
    public static void main(String[] args) {
        Agenda agenda = new Agenda();
        Contato contato = new Contato();
        // Click! O usuário selecionou um botão com a opção
        // salvar contato! O método adicionarContato() "sabe"
        // como adicionar um contato na agenda.
        agenda.adicionarContato( contato );
    }
}
```

Arquitetura em camadas

```
package agenda.negocio;

import agenda.persistencia.AgendaPersist;

public class Agenda {

    private ArrayList contatos = new ArrayList();
    private AgendaPersist AgendaPersist = new AgendaPersist();

    public void adicionarContato(Contato contato) {
        contatos.add( contato );
        // Agora que a agenda possui um novo contato
        // precisa ser armazenada no banco de dados. Já que não sei
        // como salvar a agenda no BD vou pedir para o objeto
        // AgendaPersist fazê-lo.
        AgendaPersist.salvar(this);
    }
```

Arquitetura em camadas

```
package agenda.persistencia;

public class AgendaPersist {

    // conexão com o banco de dados, etc.

    public boolean salvar(Agenda agenda) {
        // Ei! Eu sou especialista em salvar a agenda no banco
        // de dados! Deixe eu salvá-la agora!
    }
}
```

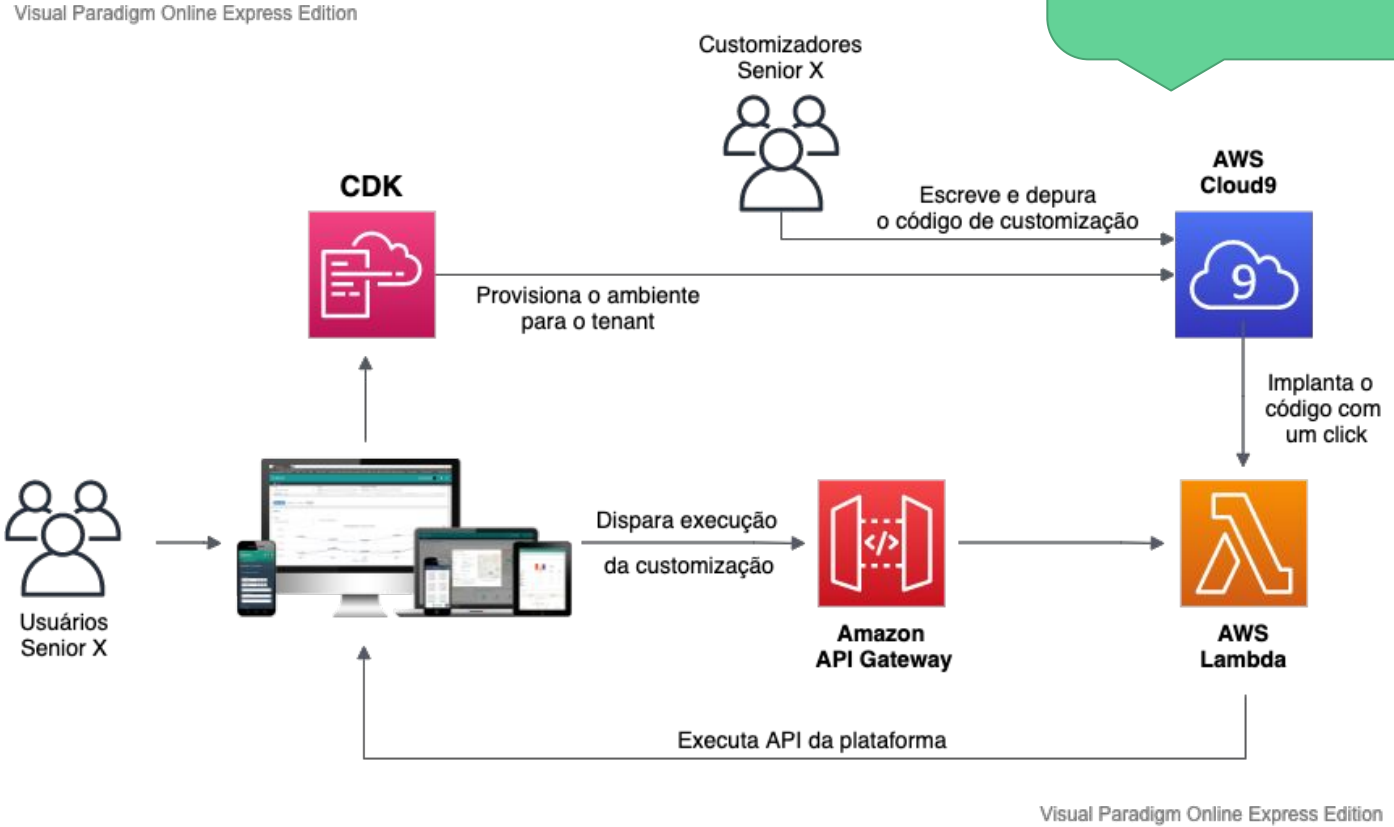
Arquitetura em camadas

- **Cada camada implementa serviços (métodos) para a camada superior.**
- **Uma camada somente deve chamar métodos de objetos da camada imediatamente inferior a sua**
- **O modelo em camadas cria maior independência entre classes de cada camada**

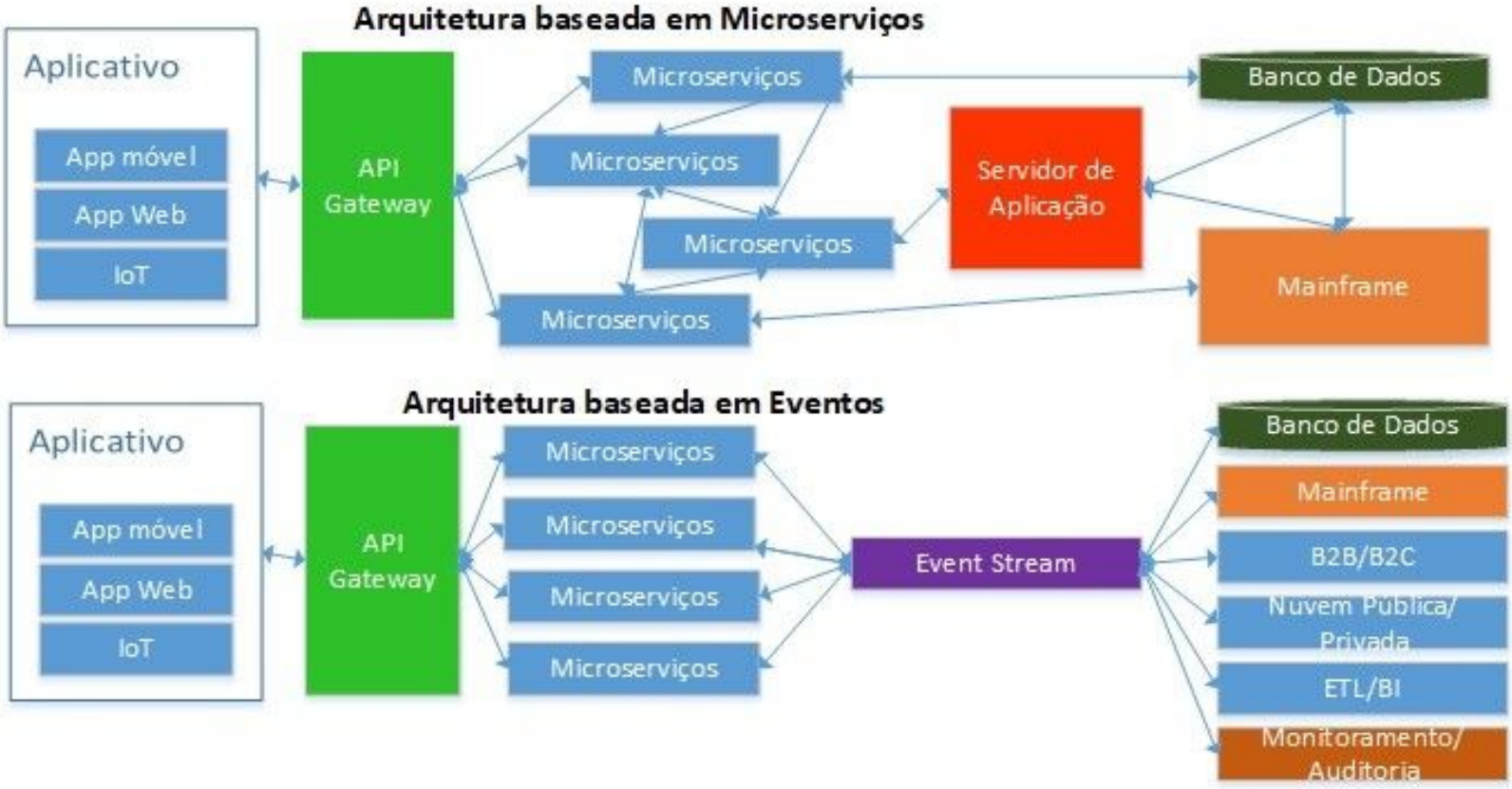
  Ex. para incluir um novo BD, basta uma nova camada de persistência.

Arquitetura em camadas

Cada camada tem uma série de classes e componentes

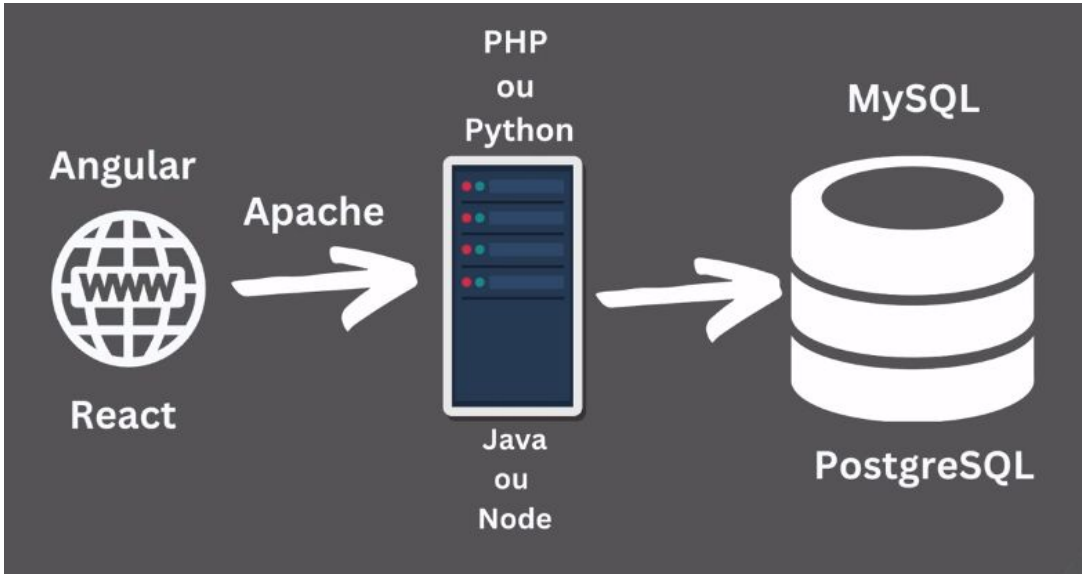


Arquitetura em camadas



Arquitetura Monolítica

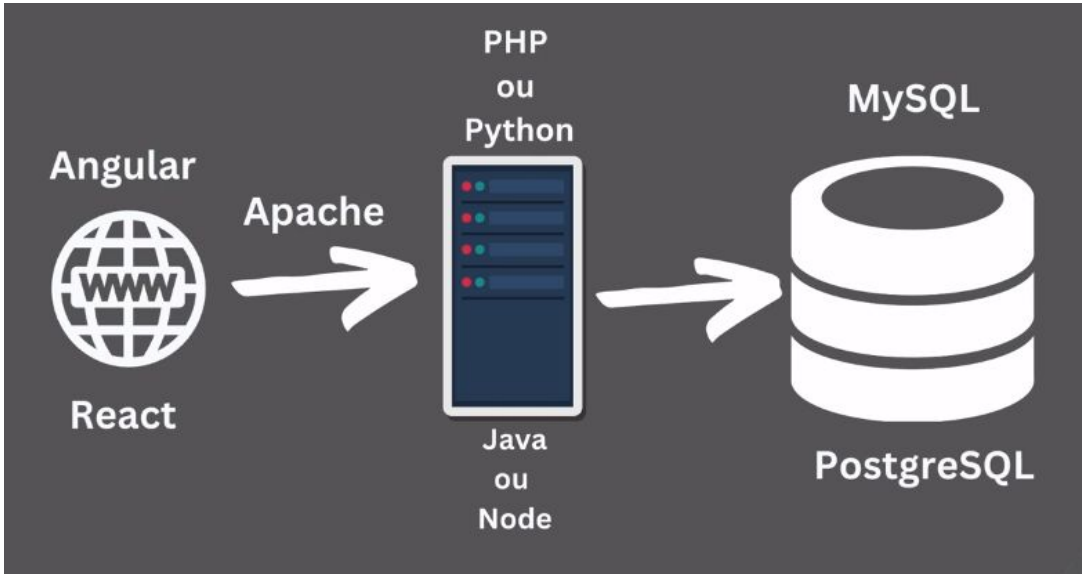
Normalmente é construído em uma única aplicação Cliente que unifica os processos em uma aplicação robusta



Arquitetura Monolítica

Frontend e Backend construídos juntos

É possivel escalar!



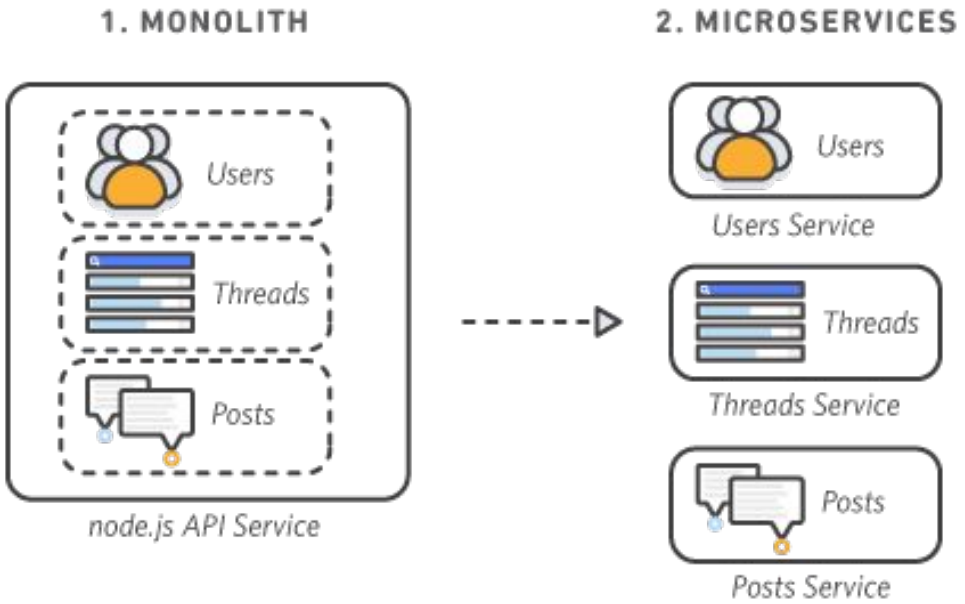
Arquitetura em Camadas: Alternativas

Facilita a manutenção e escalabilidade
da aplicação

Permite uma integração menos
Custosa

Existem outros modelos:

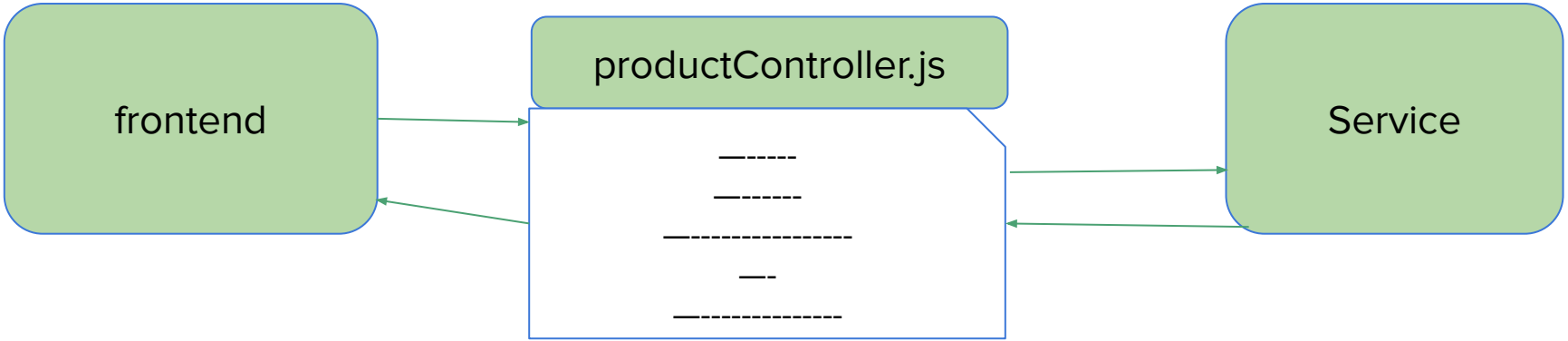
MSC, MVC



Arquitetura MSC

Model, Service e Controller: comumente utilizado no Backend.

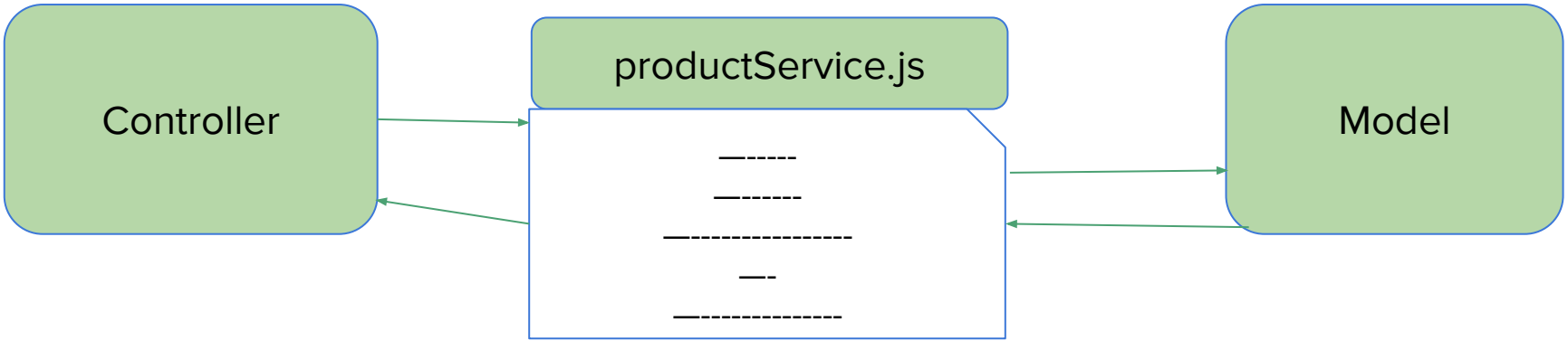
CONTROLLER: Camada responsável pela interação com o cliente. Recebe requisições (request) fazendo a chamada da camada de service, passando os dados necessários e retornando o que foi solicitado (response).



Arquitetura MSC

Model, Service e Controller: comumente utilizado no Backend.

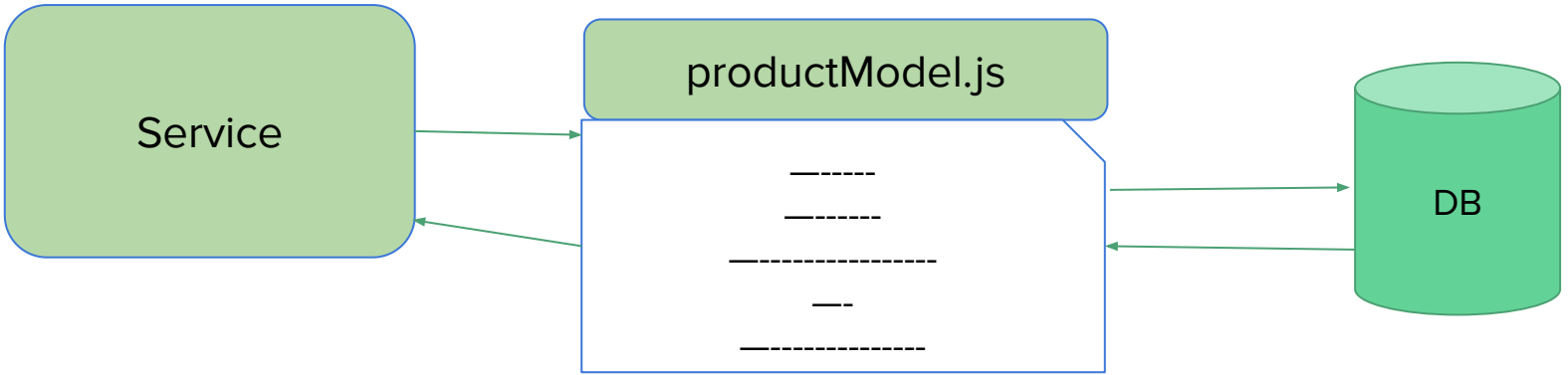
SERVICE: Camada responsável pelas regras de negócio da sua aplicação. É chamada pela camada Controller, recebendo ou não parâmetros. Faz a chamada para a camada de Model, para fazer a interação com o sistema de armazenamento dos dados.



Arquitetura MSC

Model, Service e Controller: comumente utilizado no Backend.

MODEL: Camada responsável por interações com o banco de dados ou com a ferramenta responsável pelo armazenamento.



Concluindo

Model, Service e Controller: comumente utilizado no Backend.

Arquiteturas em camadas:

Monolítica e Microsserviços

MSC

Referências

Fowler, Martin. UML Essencial. 2o Edição. - Porto Alegre: Bookman, 2000.

Eckel, Bruce. Pensando em JAVA. Prentice-Hall, 2010

Deitel, Harvey. Java - Como programar. 6a Edição. - Porto Alegre: Bookman, 2006.