

FIAP GRADUAÇÃO

SISTEMAS DE INFORMAÇÃO

MICROSERVICE AND WEB ENGINEERING

Prof^a. Aparecida Castello Rosa
profaparecida.rosa@fiap.com.br

■ Agenda

- Introdução ao microserviço (*microservice*) - conceitos
- Iniciando com Microserviços na prática!

Objetivo

- Entender os principais conceitos.
- Iniciar com microsserviços de Pagamentos.

URI x URL x URN



URI: *Uniform Resource Identifier*

URL: *Uniform Resource Locator* - Localizador de Recursos Universal

URN: *Uniform Resource Name* - Nome de Recursos Universal

URI: é o identificador do recurso. Pode ser uma imagem, uma página, etc, pois tudo o que está disponível na internet precisa de um **identificador único**.

URL: diz respeito ao local, o `Host` para acessar determinado recurso. O objetivo da **URL** é associar um endereço remoto com um nome de recurso na Internet.

URN: é o nome do recurso que será acessado e também fará parte da URI.

Resumindo:

A **URI** une o Protocolo (`https://`) a localização do recurso (**URL:** `fiap.com.br`) e o nome do recurso (**URN** - `/graduacao`).

AULA 29 – Microsserviços

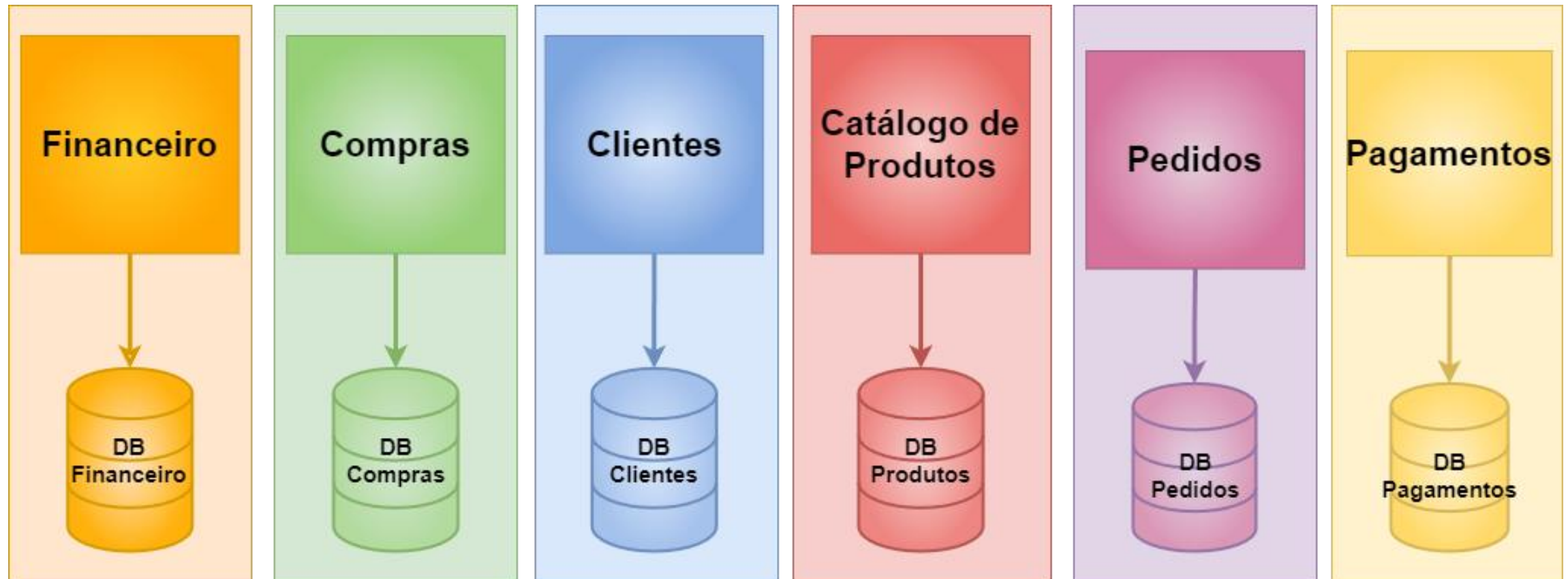
Aplicações Monolíticas



■ Aplicação Monolítica

- Todos os processos são altamente acoplados e executam como um único serviço.
- Várias responsabilidades em um único sistema – falta de coesão.
- Manutenções podem implicar em parar todo o sistema.
- Dificuldade de escalar somente uma parte da aplicação.
- Deploy da aplicação toda.
- Complexidade para adicionar ou aprimorar recursos.
- Geralmente, utiliza uma única linguagem.
- O software muito grande, leva mais tempo para ficar pronto.
- Quando o software é grande, a dependência também é enorme.
- Debug mais fácil.

Aplicações Microserviços



I Microserviço

- Determina um modelo de desenvolvimento.
- Também é um conjunto de práticas.
- Aumenta a velocidade de desenvolvimento.
- Permite escalabilidade
- Princípios e padrões de tecnologias.
- Orientado a produto.
- Coeso – responsabilidades reduzidas / especializado.
- Cada serviço realiza uma única função.
- Serviços são componentes independentes, não podem depender

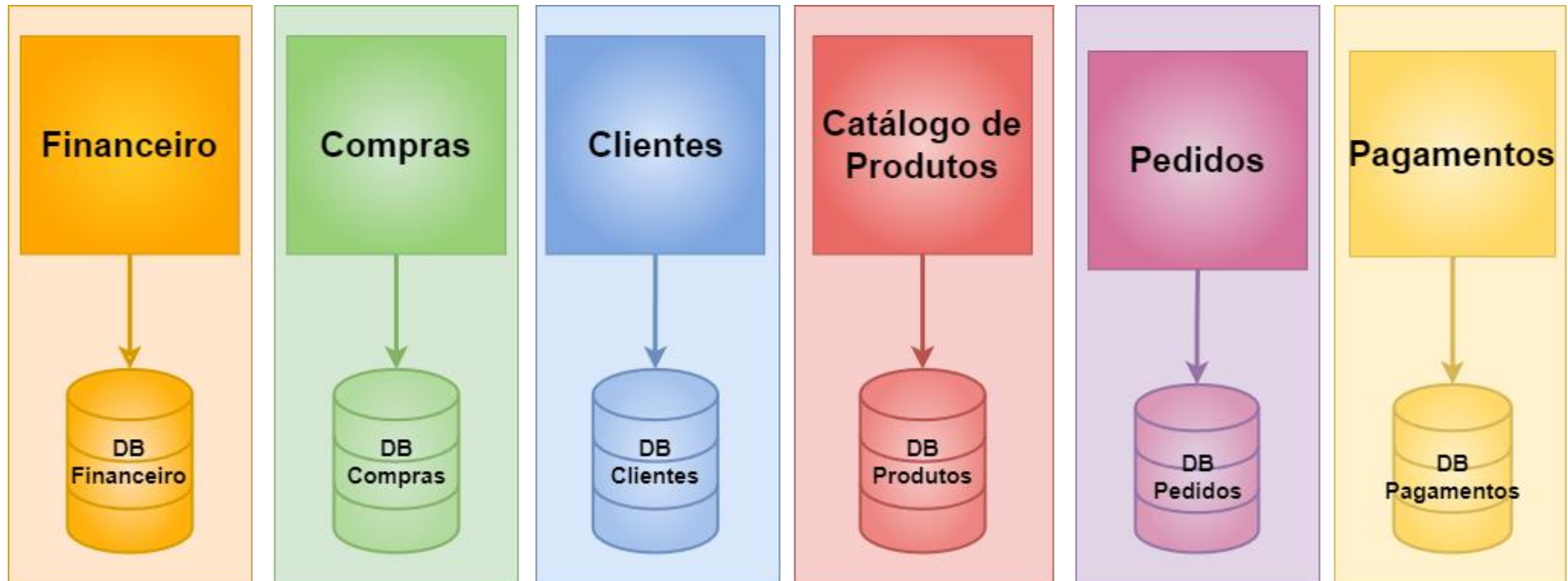
I Microserviço

- Desacoplado.
- Independentes:
 - Contexto
 - Banco de dados próprio.
 - Agnóstico de tecnologia - Tecnologias/linguagens próprias.
- Deploy menor.
- Cada serviço pode ser atualizado, implantado e escalado para atender a demanda de funções específicas de um aplicativo.
- Debug mais complicado.

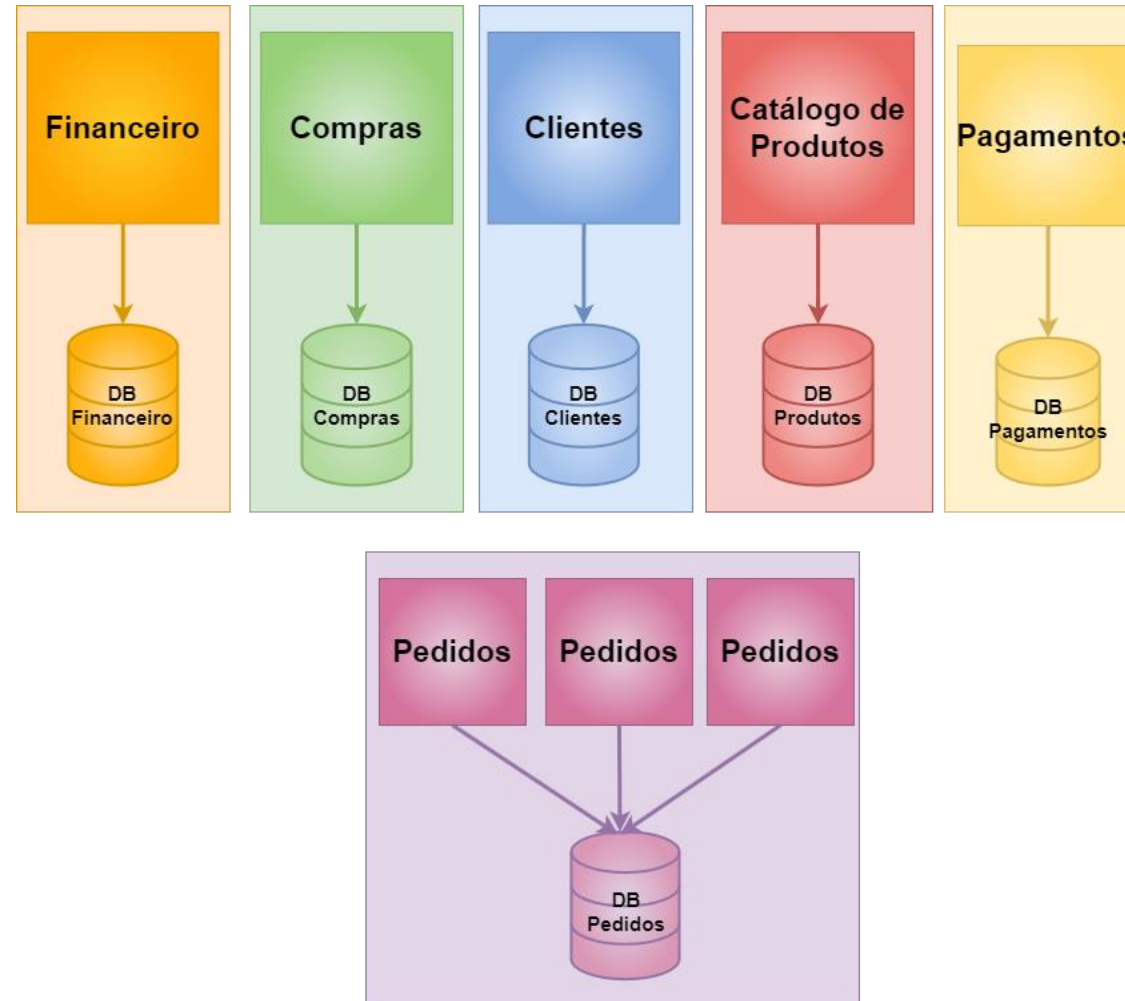
I Qual o Tamanho do Microserviço?

- Não dá para mensurar o tamanho.
- Faz apenas uma coisa bem-feita.
- Delimitação de negócio (***Bounded context***, DDD)
- Agir independentemente.
 - Incluir o *deploy*.
- É o ato de dividir o domínio em subdomínios;

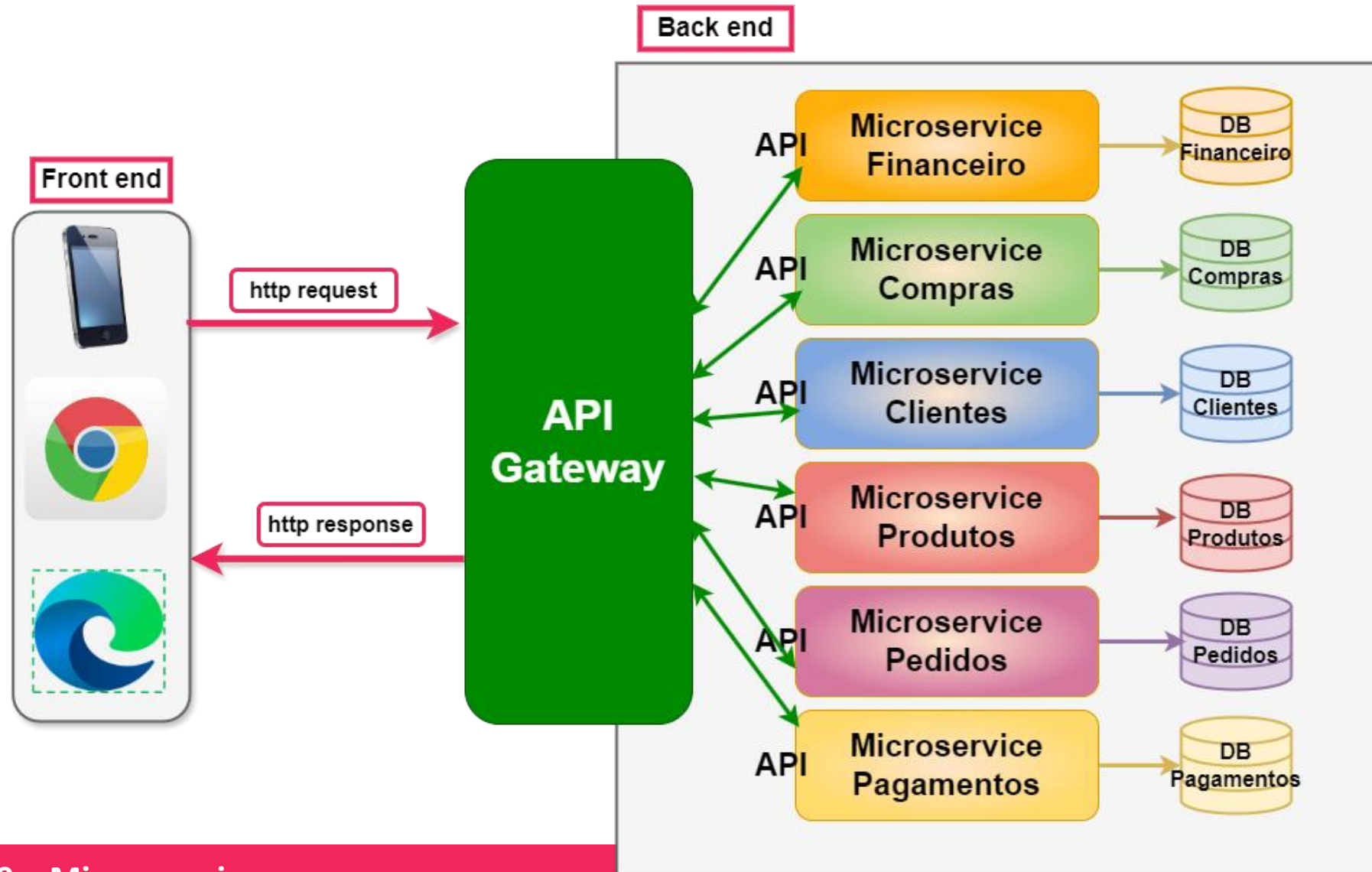
Aplicações Microserviços



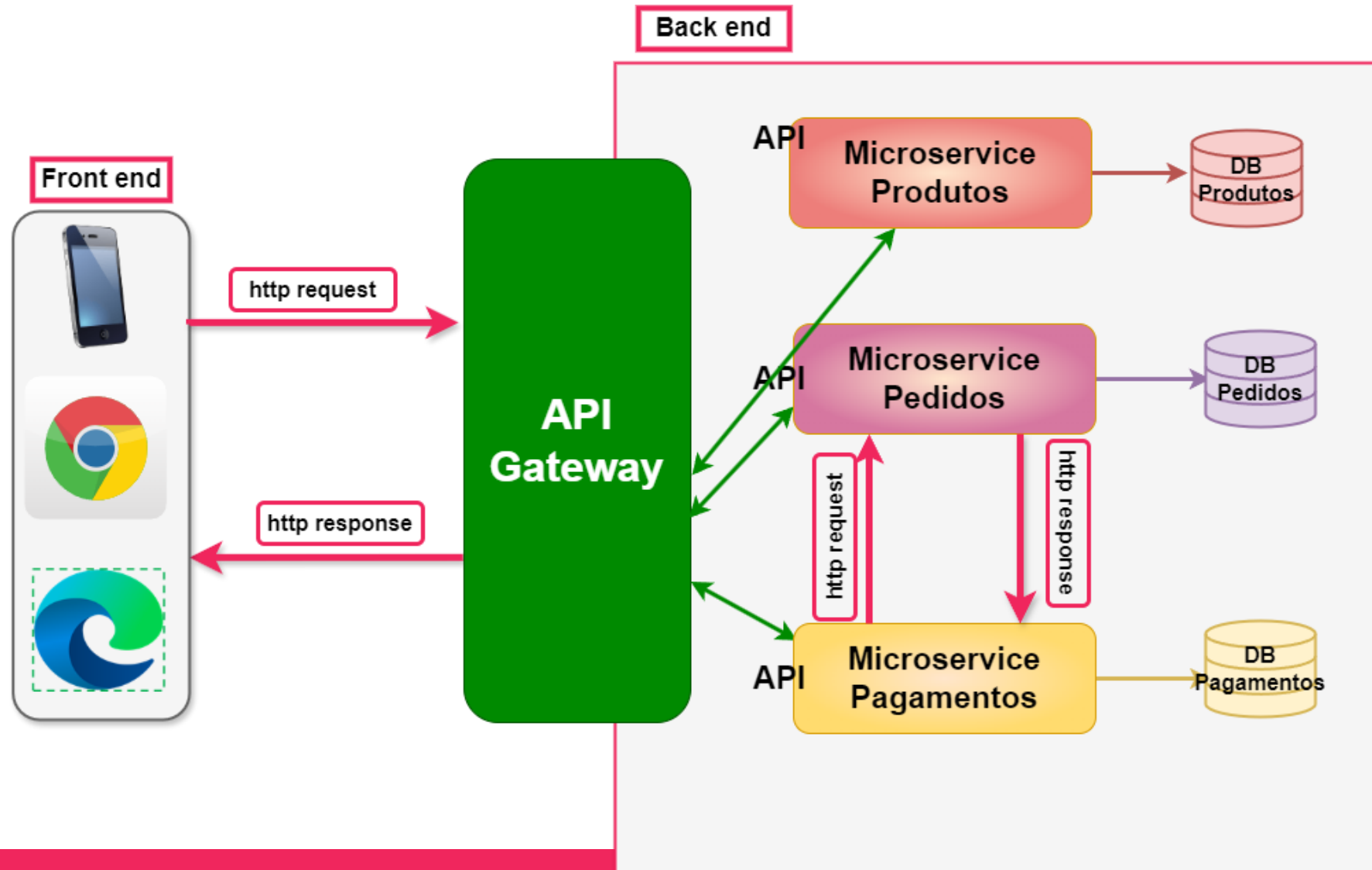
Aplicações Microserviços - Escalar



Aplicações Microserviços



Aplicações Microserviços



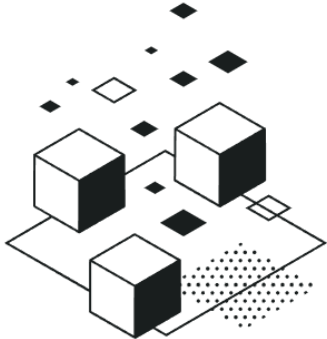
Ref. Spring Microservices

SpringOne 2024 will be virtual and free to [register! View the schedule >](#)

spring by VMware Tanzu Why Spring ▾ Learn ▾ Projects ▾ Academy ▾ Solutions ▾ Community ▾

Microservices

Microservice architectures are the 'new normal'. Building small, self-contained, ready to run applications can bring great flexibility and added resilience to your code. Spring Boot's many purpose-built features make it easy to build and run your microservices in production at scale. And don't forget, no microservice architecture is complete without [Spring Cloud](#) – easing administration and boosting your fault-tolerance.



spring by VMware Tanzu Why Spring ▾ Learn ▾ Projects ▾ Academy ▾ Solutions ▾ Co ▾

< ALL GUIDES

Spring Quickstart Guide

What you'll build	What you'll need
You will build a classic "Hello World!" endpoint which any browser can connect to. You can even tell it your name, and it will respond in a more friendly way.	An Integrated Developer Environment (IDE) Popular choices include IntelliJ IDEA , Spring Tools , Visual Studio Code , or Eclipse , and many more.
	A Java™ Development Kit (JDK) We recommend BellSoft Liberica JDK version 17.

- <https://spring.io/microservices>
- <https://spring.io/quickstart>

■ Ref. Arquitetura de Microserviços

- Microservices a definition of this new architectural term (inglês)
 - <https://martinfowler.com/articles/microservices.html>
- Resumo em Português
 - <https://www.thoughtworks.com/pt-br/insights/blog/microservices-nutshell>
 - <https://codigo35.com/2016/01/02/microservicos/>
- Medium Português
 - <https://medium.com/xp-inc/entendendo-a-arquitetura-de-microservices-cdab6b52d6ed>

- <https://cursos.alura.com.br/course/microservicos-padroes-projeto>
- <https://cursos.alura.com.br/course/fundamentos-microservicos-aprofundando-conceitos>
- <https://cursos.alura.com.br/course/microservicos-implementando-java-spring>
- <https://cursos.alura.com.br/course/spring-boot-3-desenvolva-api-rest-java>
- <https://cursos.alura.com.br/course/spring-boot-aplique-boas-praticas-proteja-api-rest>
- <https://cursos.alura.com.br/course/microservices-spring-cloud-service-registry-config-server>
- <https://cursos.alura.com.br/course/microservicos-pratica-iac-cdk-deploy-aws>


- <https://cursos.alura.com.br/course/kafka-spring-aplicacoes-fluxos-dados>
- <https://cursos.alura.com.br/course/microservicos-pratica-mensageria-rabbitmq>
- <https://cursos.alura.com.br/course/integracao-continua-testes-automatizados-pipeline-github-actions>

I Referências Adicionais

- Baeldung
 - <https://www.baeldung.com/>
- Michelli Brito
 - <https://www.michellibrito.com/>
 - <https://www.youtube.com/@MichelliBrito>
- Cida Castello
 - <https://www.youtube.com/c/CidaCastello>

Spring Boot - versão








SpringOne 2024 will be virtual and free to [register! View the schedule >](#)

 by VMware Tanzu [Why Spring](#) [Learn](#) [Projects](#) [Academy](#) [Solutions](#) [Community](#)

Spring Boot
Spring Framework
Spring Data >
Spring Cloud >
Spring Cloud Data Flow
Spring Security >
Spring Authorization Server
Spring for GraphQL
Spring Session >
Spring Integration
Spring HATEOAS
Spring Modulith
Spring REST Docs
Spring AI
Spring Batch

Spring Boot 3.3.2

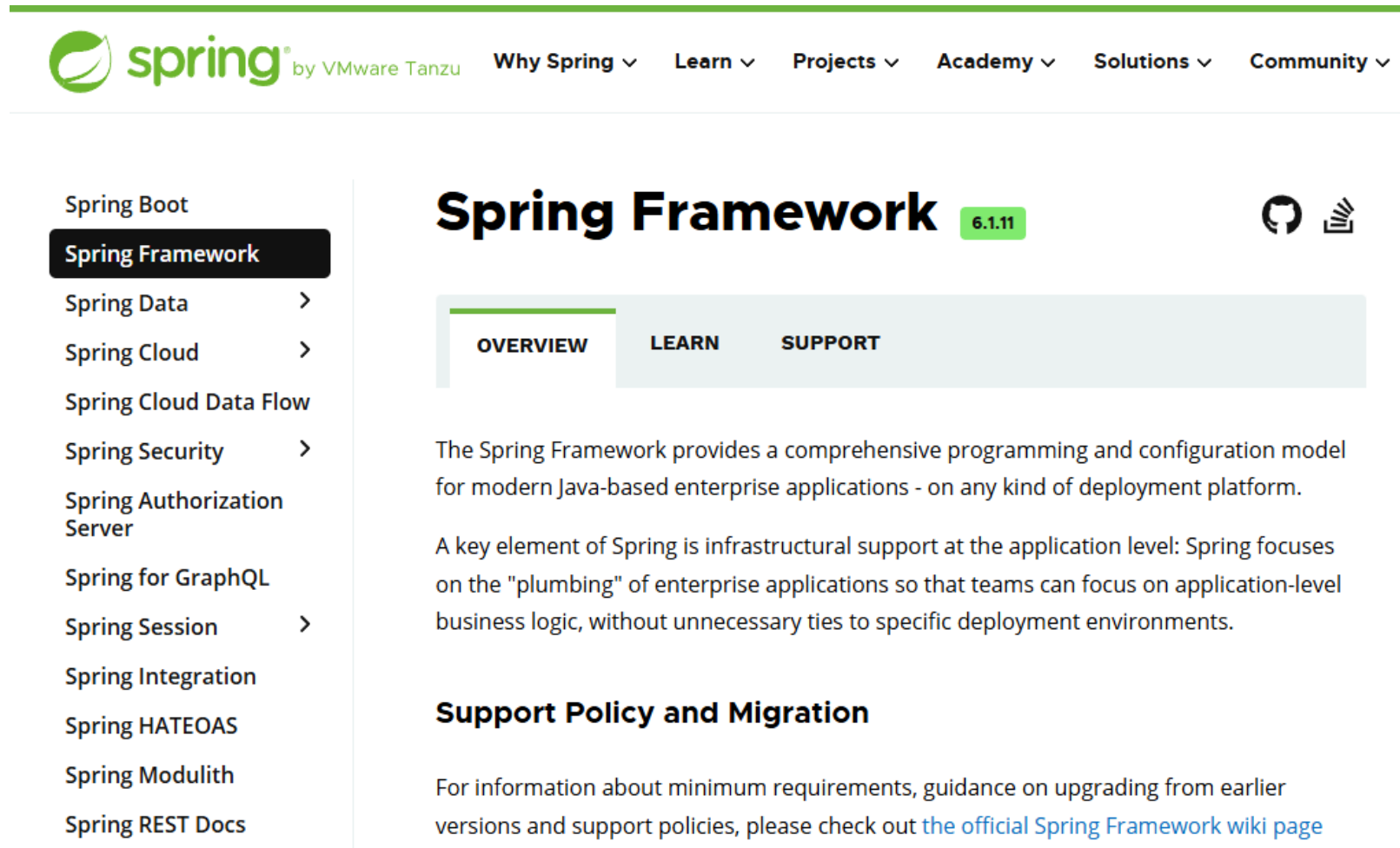
[OVERVIEW](#) [LEARN](#) [SUPPORT](#) [SAMPLES](#)

Branch	Initial Release	End of Support	End Enterprise Support *
 3.4.x	2024-11-21	2025-11-21	2027-02-21
 3.3.x	2024-05-23	2025-05-23	2026-08-23
 3.2.x	2023-11-23	2024-11-23	2026-02-23
 3.1.x	2023-05-18	2024-05-18	2025-08-18
 3.0.x	2022-11-24	2023-11-24	2025-02-24
 2.7.x	2022-05-19	2023-11-24	2025-08-24
 2.6.x	2021-11-17	2022-11-24	2024-02-24

[More >](#)

<https://spring.io/projects/spring-boot#support>

Spring Framework - versão



The screenshot shows the Spring Framework website. At the top, there's a navigation bar with the Spring logo (a green circle with a white 'S') and the text 'spring by VMware Tanzu'. To the right of the logo are links: 'Why Spring', 'Learn', 'Projects', 'Academy', 'Solutions', and 'Community', each followed by a downward arrow. Below the navigation bar is a sidebar on the left with a list of Spring projects: 'Spring Boot', 'Spring Framework' (highlighted with a dark background), 'Spring Data', 'Spring Cloud', 'Spring Cloud Data Flow', 'Spring Security', 'Spring Authorization Server', 'Spring for GraphQL', 'Spring Session', 'Spring Integration', 'Spring HATEOAS', 'Spring Modulith', and 'Spring REST Docs'. The main content area has the title 'Spring Framework' in large black font, followed by a green badge with '6.1.11'. To the right of the title are icons for GitHub and a document. Below the title is a horizontal tab bar with three tabs: 'OVERVIEW' (active, highlighted with a green underline), 'LEARN', and 'SUPPORT'. The 'OVERVIEW' tab contains two paragraphs of text. The first paragraph states: 'The Spring Framework provides a comprehensive programming and configuration model for modern Java-based enterprise applications - on any kind of deployment platform.' The second paragraph states: 'A key element of Spring is infrastructural support at the application level: Spring focuses on the "plumbing" of enterprise applications so that teams can focus on application-level business logic, without unnecessary ties to specific deployment environments.' Below the paragraphs is a section titled 'Support Policy and Migration' in bold. Under this section, it says: 'For information about minimum requirements, guidance on upgrading from earlier versions and support policies, please check out the official Spring Framework wiki page' with a blue link.

spring by VMware Tanzu

Why Spring ▾ Learn ▾ Projects ▾ Academy ▾ Solutions ▾ Community ▾

Spring Boot

Spring Framework

Spring Data >

Spring Cloud >

Spring Cloud Data Flow

Spring Security >

Spring Authorization Server

Spring for GraphQL

Spring Session >



Spring Integration

Spring HATEOAS

Spring Modulith

Spring REST Docs

Spring Framework 6.1.11

OVERVIEW LEARN SUPPORT

The Spring Framework provides a comprehensive programming and configuration model for modern Java-based enterprise applications - on any kind of deployment platform.

A key element of Spring is infrastructural support at the application level: Spring focuses on the "plumbing" of enterprise applications so that teams can focus on application-level business logic, without unnecessary ties to specific deployment environments.

Support Policy and Migration

For information about minimum requirements, guidance on upgrading from earlier versions and support policies, please check out [the official Spring Framework wiki page](#)

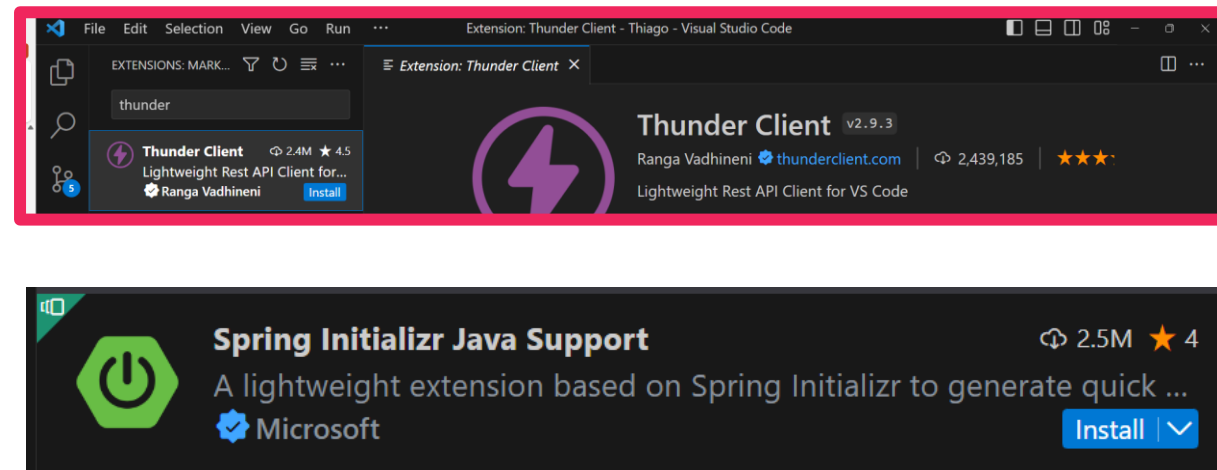
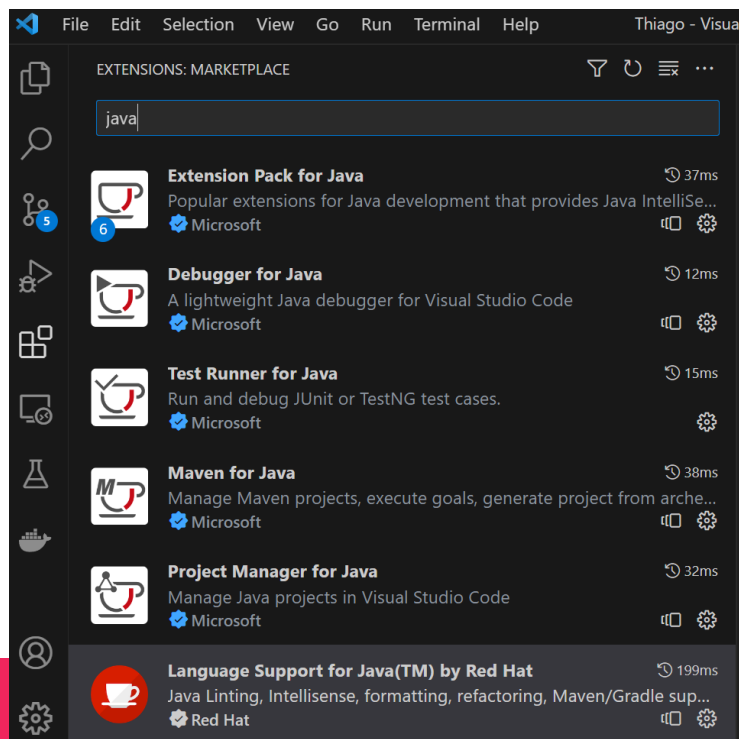
<https://spring.io/projects/spring-framework>

Praticando!

Implementando o MS Pagamentos

Preparando o Ambiente – API Clients

- Postman - <https://www.postman.com/>
- Insomnia - <https://insomnia.rest/>
- VS Code – extensão Thunder Client



I Lombok - Instalar

- Precisa fazer a instalação do Lombok no STS e outras IDE's.
- Referências:
- https://www.youtube.com/watch?v=W0ywxkvc4_M
- <https://projectlombok.org/setup/eclipse>
- Lombok no IntelliJ
- <https://plugins.jetbrains.com/plugin/6317-lombok>

■ Preparando o Ambiente - Recursos

- **Database**

- Database H2 (testes)
- MySQL, Postgres, Oracle (produção)

- **Docker**

- Docker Linux
- Docker Desktop – Windows, WSL2
- Docker for Mac

Material de Apoio no GitHub

- <https://github.com/cidacastello/FIAP-Microservice-2024>

■ Preparando o Ambiente - Recursos

- Java **JDK 17 LTS** ou superior
- **Spring Boot 3.2.8 ou superior; ou Spring Boot 3.3.2 ou superior.**
- **Spring Initializr** - Initializr generates spring boot project with just what you need to start quickly!

<https://start.spring.io/>

MS-Pagamento – *init*



Project

☐ Gradle - Groovy ☐ Gradle - Kotlin ☒ Java ☐ Kotlin ☐ Groovy
☒ Maven

Spring Boot

☐ 3.4.0 (SNAPSHOT) ☐ 3.4.0 (M1) ☐ 3.3.3 (SNAPSHOT) ☒ 3.3.2
☐ 3.2.9 (SNAPSHOT) ☐ 3.2.8

Project Metadata

Group
Artifact
Name
Description
Package name
Packaging ☒ Jar ☐ War
Java ☐ 22 ☐ 21 ☒ 17

Language

Dependencies

[ADD DEPENDENCIES...](#) CTRL + B

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Data JPA SQL

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

H2 Database SQL

Provides a fast in-memory database that supports JDBC API and R2DBC access, with a small (2mb) footprint. Supports embedded and server modes as well as a browser based console application.

Validation I/O

Bean Validation with Hibernate validator.

Lombok DEVELOPER TOOLS

Java annotation library which helps to reduce boilerplate code.

Spring Boot DevTools DEVELOPER TOOLS

Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

MS-Pagamento – *init* – GitHub



Project

☐ Gradle - Groovy ☐ Gradle - Kotlin ☒ Java ☐ Kotlin ☐ Groovy
☒ Maven

Spring Boot

☐ 3.4.0 (SNAPSHOT) ☐ 3.4.0 (M1) ☐ 3.3.3 (SNAPSHOT) ☒ 3.3.2
☐ 3.2.9 (SNAPSHOT) ☐ 3.2.8

Project Metadata

Group
Artifact
Name
Description
Package name
Packaging ☒ Jar ☐ War
Java ☐ 22 ☐ 21 ☒ 17

Language

Dependencies

[ADD DEPENDENCIES...](#) [CTI](#)

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Data JPA SQL

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

H2 Database SQL

Provides a fast in-memory database that supports JDBC API and R2DBC access, with a small (2mb) footprint. Supports embedded and server modes as well as a browser based console application.

Validation I/O

Bean Validation with Hibernate validator.

Lombok DEVELOPER TOOLS

Java annotation library which helps to reduce boilerplate code.

Spring Boot DevTools DEVELOPER TOOLS

Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

MS Pagamentos

MS Pagamentos

Pagamento

- id : Long
- valor : BigDecimal
- nome : String
- numeroDoCartao : String
- validade : String
- codigoDeSeguranca : String
- status : Status
- pedidoId : Long
- formaDePagamentoId : Long

<<enum>>

Status

- CRIADO : String
- CONFIRMADO : String
- CANCELADO : String

powered by Astah

I README.md

- Todos os projetos deverão ter um arquivo README.md no qual deve constar todas as informações do projeto, tais como, *stacks* utilizadas, *endpoints*, como compilar, executar, etc.
- Pesquise como fazer um README.md.

Implementando as classes model

- Adicionar o plugin do Maven para evitar falhas. (projeto anterior ou GitHub)

enum Status



The screenshot shows a code editor window titled 'Status.java'. The code defines a package and an enum. The enum 'Status' has three values: 'CRIADO', 'CONFIRMADO', and 'CANCELADO'. A red rectangle highlights the enum values.

```
1 package com.github.cidacastello.ms_pagamento.model;  
2  
3 public enum Status {  
4     CRIADO,  
5     CONFIRMADO,  
6     CANCELADO  
7 }  
8
```

Class Pagamento.java

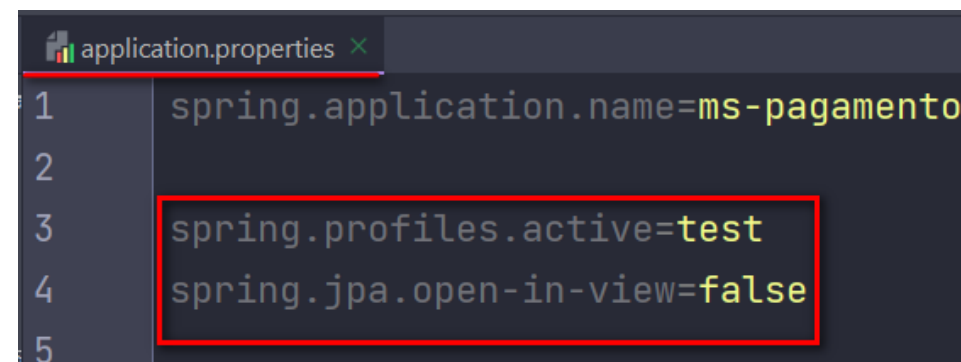
```
Pagamento.java x
1  package com.github.cidacastello.ms_pagamento.model;
2
3  import ...
10
11  @AllArgsConstructor
12  @NoArgsConstructor
13  @Getter
14  @Setter
15  public class Pagamento {
16
17      private Long id;
18      private BigDecimal valor;
19      private String nome; // Nome no cartão
20      private String numeroDoCartao; // XXXX XXXX XXXX XXXX
21      private String validade; // validade do cartão - MM/AA
22      private String codigoDeSeguranca; // código de segurança - XXX
23      private Status status; // Status do pagamento
24      private Long pedidoId; // Id do pedido
25      private Long formaDePagamentoId; // 1 - dinheiro | 2 - cartão | 3 - pix
26  }
```

Definindo Entidade - ORM

```
Pagamento.java x
22
23 @Entity
24 @Table(name = "tb_pagamento")
25 public class Pagamento {
26
27     @Id
28     @GeneratedValue(strategy = GenerationType.IDENTITY)
29     private Long id;
30     @Column(nullable = false)
31     private BigDecimal valor;
32     private String nome; // Nome no cartão
33     private String numeroDoCartao; // XXXX XXXX XXXX XXXX
34     private String validade; // validade do cartão - MM/AA
35     private String codigoDeSeguranca; // código de segurança - XXX
36     @Column(nullable = false)
37     @Enumerated(value = EnumType.STRING)
38     private Status status; // Status do pagamento
39     @Column(nullable = false)
40     private Long pedidoId; // Id do pedido
41     @Column(nullable = false)
42     private Long formaDePagamentoId; // 1 - dinheiro | 2 - cartão | 3 - pix
43 }
```

■ Configurar H2

- Implementar application-test.properties – (GitHub ou projeto anterior)
- Alterar application.properties



The screenshot shows a code editor window titled 'application.properties'. The content of the file is as follows:

```
1 spring.application.name=ms-pagamento
2
3 spring.profiles.active=test
4 spring.jpa.open-in-view=false
5
```

A red rectangular box highlights the lines containing 'spring.profiles.active=test' and 'spring.jpa.open-in-view=false'.

I Seed Database

- Criar o arquivo *import.sql* com 3 pelo menos registros

Class PagamentoRepository

- Implementar a interface *Repository*

```
PagamentoRepository.java x
1 package com.github.cidacastello.ms_pagamento.repository;
2
3
4 import com.github.cidacastello.ms_pagamento.model.Pagamento;
5 import org.springframework.data.jpa.repository.JpaRepository;
6
7 public interface PagamentoRepository extends JpaRepository<Pagamento, Long> {
8
9 }
```


Class StatusDTO

```
StatusDTO.java x
1  package com.github.cidacastello.ms_pagamento.dto;
2
3  import com.github.cidacastello.ms_pagamento.model.Status;
4  import lombok.Getter;
5
6  @Getter
7  public class StatusDTO {
8
9      private Status status;
10 }
```

Class PagamentoDTO

```
PagamentoDTO.java x
1  package com.github.cidacastello.ms_pagamento.dto;
2
3  import ...
15
16  @AllArgsConstructor
17  @NoArgsConstructor
18  @Getter
19  public class PagamentoDTO {
20
21      private Long id;
22      @NotNull(message = "Campo obrigatório")
23      @Positive(message = "0 valor deve ser positivo")
24      private BigDecimal valor;
25      @Size(max=100, message = "Máximo de 100 caracteres")
26      private String nome; // Nome no cartão
27      @Size(max = 19, message = "Número do cartão deve ter no máximo de 19 caracteres")
28      private String numeroDoCartao; // XXXX XXXX XXXX XXXX
29      @Size(min=5, max = 5, message = "Validade do cartão deve ter 5 caracteres")
30      private String validade; // validade do cartão - MM/AA
```

Class PagamentoDTO

```
PagamentoDTO.java x
31 @Size(min = 3, max = 3, message = "Código de segurança deve ter 3 caracteres")
32 private String codigoDeSeguranca; // código de segurança - XXX
33 @Enumerated(value = EnumType.STRING)
34 private Status status; // Status do pagamento
35 @NotNull(message = "Pedido ID é obrigatório")
36 private Long pedidoId; // Id do pedido
37 @NotNull(message = "Forma de pagamento ID é obrigatório")
38 private Long formaDePagamentoId; // 1 - dinheiro | 2 - cartão | 3 - pix
39
40 @ public PagamentoDTO(Pagamento entity) {
41     id = entity.getId();
42     valor = entity.getValor();
43     nome = entity.getNome();
44     numeroDoCartao = entity.getNumeroDoCartao();
45     validade = entity.getValidade();
46     codigoDeSeguranca = entity.getCodigoDeSeguranca();
47     status = entity.getStatus();
48     pedidoId = entity.getPedidoId();
49     formaDePagamentoId = entity.getFormaDePagamentoId();
50 }
51 }
```

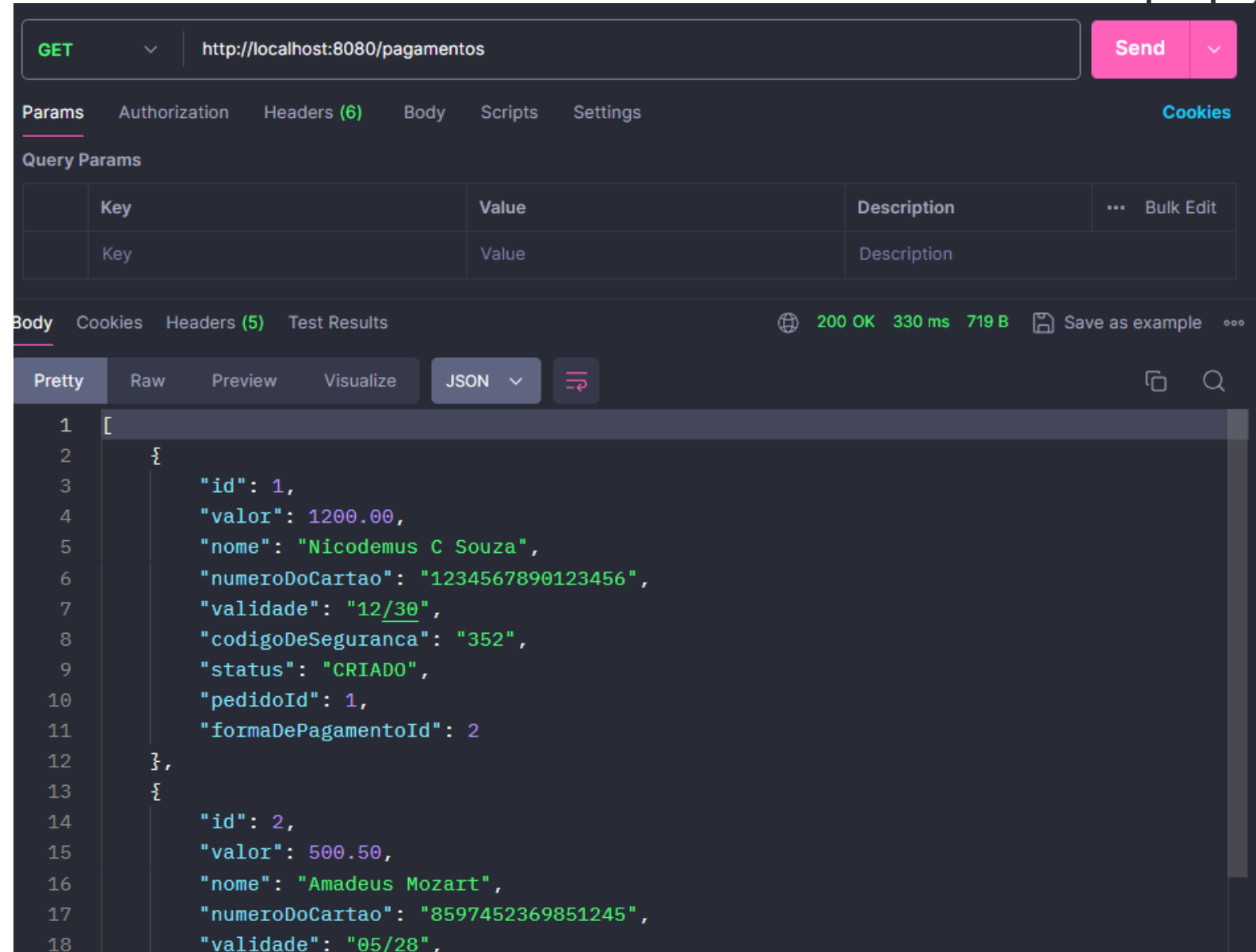
Class PagamentoService - findAll

```
PagamentoService.java x
1 package com.github.cidacastello.ms_pagamento.service;
2
3 import ...
4
14
15 @Service
16 public class PagamentoService {
17
18     @Autowired
19     private PagamentoRepository repository;
20
21     @Transactional(readonly = true)
22     public List<PagamentoDTO> findAll(){
23         List<Pagamento> list = repository.findAll();
24         return list.stream().map(PagamentoDTO::new).collect(Collectors.toList());
25     }
26 }
```

Class PagamentoController - findAll

```
PagamentoController.java x
1 package com.github.cidacastello.ms_pagamento.controller;
2
3 import ...
13
14 @RestController
15 @RequestMapping(value = "/pagamentos")
16 public class PagamentoController {
17
18     @Autowired
19     private PagamentoService service;
20
21     @GetMapping
22     public ResponseEntity<List<PagamentoDTO>> findAll(){
23         List<PagamentoDTO> dto = service.findAll();
24         return ResponseEntity.ok(dto);
25     }
}
```

Testar o MS



GET http://localhost:8080/pagamentos

Params Authorization Headers (6) Body Scripts Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (5) Test Results 200 OK 330 ms 719 B Save as example

Pretty Raw Preview Visualize JSON

```
[
  {
    "id": 1,
    "valor": 1200.00,
    "nome": "Nicodemus C Souza",
    "numeroDoCartao": "1234567890123456",
    "validade": "12/30",
    "codigoDeSeguranca": "352",
    "status": "CRIADO",
    "pedidoId": 1,
    "formaDePagamentoId": 2
  },
  {
    "id": 2,
    "valor": 500.50,
    "nome": "Amadeus Mozart",
    "numeroDoCartao": "8597452369851245",
    "validade": "05/28",
```

Class PagamentoService - findById

```
15  @Service
16  public class PagamentoService {
17
18      @Autowired
19      private PagamentoRepository repository;
20
21      @Transactional(readonly = true)
22      public List<PagamentoDTO> findAll() {...}
23
24
25
26
27      @Transactional(readonly = true)
28      public PagamentoDTO findById(long id) {
29          Pagamento entity = repository.findById(id).orElseThrow(
30              () -> new EntityNotFoundException("Recurso não encontrado")
31          );
32
33          return new PagamentoDTO(entity);
34      }
```

Class PagamentoController - findById

```
14 @RestController
15 @RequestMapping(value = "/pagamentos")
16 public class PagamentoController {
17
18     @Autowired
19     private PagamentoService service;
20
21     @GetMapping
22     public ResponseEntity<List<PagamentoDTO>> findAll(){...}
23
24
25
26
27     @GetMapping("/{id}")
28     public ResponseEntity<PagamentoDTO> findById(@PathVariable Long id){
29         PagamentoDTO dto = service.findById(id);
30         return ResponseEntity.ok(dto);
31     }
32 }
```


Testar o MS

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** http://localhost:8080/pagamentos/1
- Send Button:** A pink button labeled "Send" with a dropdown arrow.
- Tabs:** Params, Authorization, Headers (6), Body, Scripts, Settings, Cookies.
- Query Params Table:**

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		
- Body Tab:** Selected, showing a JSON response.
 - Visualize:** Pretty, Raw, Preview, Visualize
 - Format:** JSON
 - Status:** 200 OK, 34 ms, 350 B
 - Actions:** Save as example, ...
- JSON Response:**

```
1 {
2   "id": 1,
3   "valor": 1200.00,
4   "nome": "Nicodemus C Souza",
5   "numeroDoCartao": "1234567890123456",
6   "validade": "12/30",
7   "codigoDeSeguranca": "352",
8   "status": "CRIADO",
9   "pedidoId": 1,
10  "formaDePagamentoId": 2
11 }
```

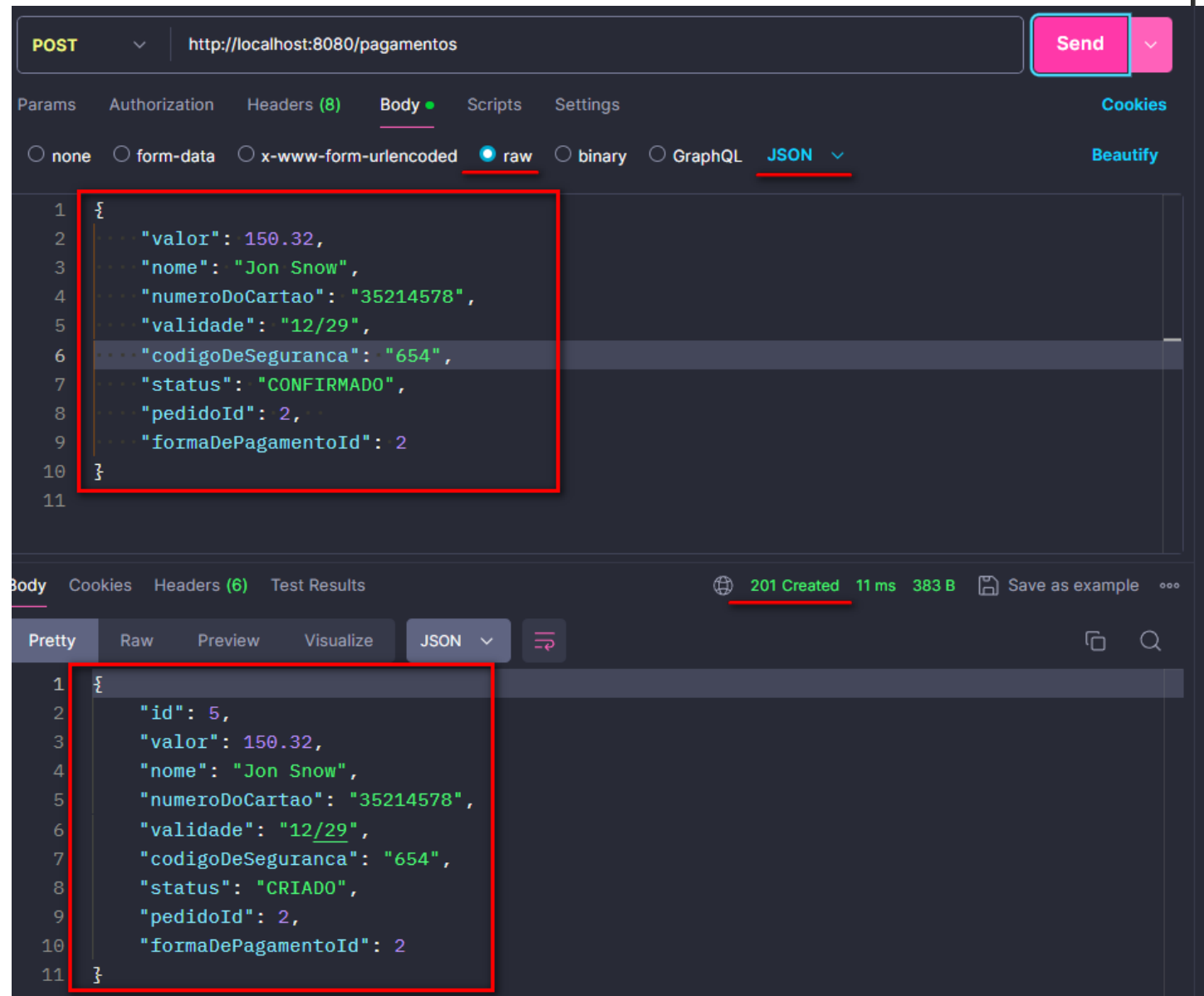
Class PagamentoService - insert

```
PagamentoService.java x
36  @Transactional
37  public PagamentoDTO insert(PagamentoDTO dto) {
38      Pagamento entity = new Pagamento();
39      copyDtoToEntity(dto, entity);
40      entity = repository.save(entity);
41      return new PagamentoDTO(entity);
42  }
43
44  @private void copyDtoToEntity(PagamentoDTO dto, Pagamento entity) {
45      entity.setValor(dto.getValor());
46      entity.setNome(dto.getNome());
47      entity.setNumeroDoCartao(dto.getNumeroDoCartao());
48      entity.setValidade(dto.getValidade());
49      entity.setCodigoDeSeguranca(dto.getCodigoDeSeguranca());
50      //quando incluimos o pagamento usamos o status CRIADO
51      entity.setStatus(Status.CRIADO);
52      entity.setPedidoId(dto.getPedidoId());
53      entity.setFormaDePagamentoId(dto.getFormaDePagamentoId());
54  }
55 }
```

Class PagamentoController - insert

```
PagamentoController.java x
27     @GetMapping("/{id}")
28     public ResponseEntity<PagamentoDTO> findById(@PathVariable Long id){...}
32
33     @PostMapping
34     public ResponseEntity<PagamentoDTO> insert(@RequestBody @Valid PagamentoDTO dto){
35         dto = service.insert(dto);
36         URI uri = ServletUriComponentsBuilder
37             .fromCurrentRequest()
38             .path("/{id}")
39             .buildAndExpand(dto.getId()).toUri();
40         return ResponseEntity.created(uri).body(dto);
41     }
42 }
```

Testar o MS



The screenshot displays the Postman interface for a POST request to `http://localhost:8080/pagamentos`. The request body is a JSON object with the following fields: `valor` (150.32), `nome` (Jon Snow), `numeroDoCartao` (35214578), `validade` (12/29), `codigoDeSeguranca` (654), `status` (CONFIRMADO), `pedidoId` (2), and `formaDePagamentoId` (2). The response status is 201 Created, with a response time of 11 ms and a body size of 383 B. The response body is a JSON object with the following fields: `id` (5), `valor` (150.32), `nome` (Jon Snow), `numeroDoCartao` (35214578), `validade` (12/29), `codigoDeSeguranca` (654), `status` (CRIADO), `pedidoId` (2), and `formaDePagamentoId` (2).

```
POST http://localhost:8080/pagamentos

{
  "valor": 150.32,
  "nome": "Jon Snow",
  "numeroDoCartao": "35214578",
  "validade": "12/29",
  "codigoDeSeguranca": "654",
  "status": "CONFIRMADO",
  "pedidoId": 2,
  "formaDePagamentoId": 2
}
```

201 Created 11 ms 383 B Save as example

```
{
  "id": 5,
  "valor": 150.32,
  "nome": "Jon Snow",
  "numeroDoCartao": "35214578",
  "validade": "12/29",
  "codigoDeSeguranca": "654",
  "status": "CRIADO",
  "pedidoId": 2,
  "formaDePagamentoId": 2
}
```

Class PagamentoService - delete

```
PagamentoService.java x
44 @Transactional
45 public void delete (Long id){
46     if(!repository.existsById(id)){
47         throw new EntityNotFoundException("Recurso não encontrado");
48     }
49     try{
50         repository.deleteById(id);
51     }catch (EntityNotFoundException e){//depois será alterado
52         throw new EntityNotFoundException("Recurso não encontrado");
53     }
54 }
```

Class PagamentoController - delete

```
PagamentoController.java x
33     @PostMapping
34     public ResponseEntity<PagamentoDTO> insert(@RequestBody @Valid PagamentoDTO dto){...}
42
43     @DeleteMapping("/{id}")
44     public ResponseEntity<Void> delete(@PathVariable Long id){
45         service.delete(id);
46         return ResponseEntity.noContent().build();
47     }
48 }
49
```

Testar o MS

DELETE ⌵ `http://localhost:8080/pagamentos/2` **Send** ⌵

Params Authorization Headers (6) Body Scripts Settings **Cookies**

Query Params

	Key	Value	Description	⋮ Bulk Edit
	Key	Value	Description	

Body Cookies Headers (3) Test Results 🌐 204 No Content 13 ms 112 B 💾 Save as example ⋮

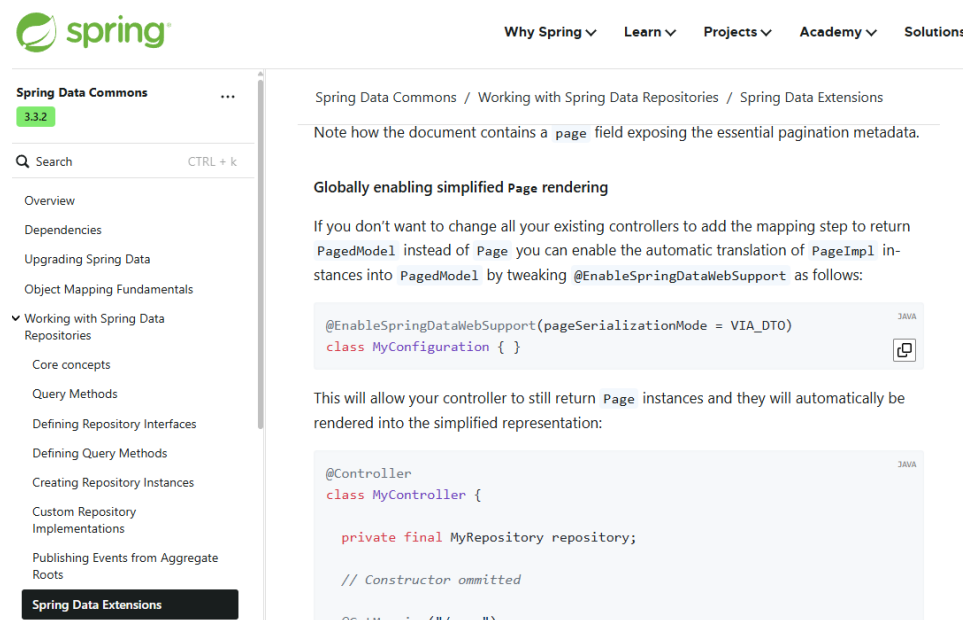
Pretty Raw Preview Visualize **Text** ⌵ 🔄 📄 🔍

1 |

Busca Paginada

- Para testar, vamos colocar mais registros no DB.
- Precisamos criar uma classe de configuração.

<https://docs.spring.io/spring-data/commons/reference/repositories/core-extensions.html#core.web.pageables>



Classe WebConfiguration

```
WebConfiguration.java x
1  package com.github.cidacastello.ms_pagamento.config;
2
3  import org.springframework.context.annotation.Configuration;
4  import org.springframework.data.web.config.EnableSpringDataWebSupport;
5  import org.springframework.web.servlet.config.annotation.EnableWebMvc;
6
7  import static org.springframework.data.web.config.
8      EnableSpringDataWebSupport.PageSerializationMode.VIA_DTO;
9
10 @Configuration
11 @EnableWebMvc
12 @EnableSpringDataWebSupport(pageSerializationMode = VIA_DTO)
13 public class WebConfiguration {
14
15 }
```

Alterar PagamentoService - findAll

```
WebConfiguration.java x PagamentoService.java x
18 public class PagamentoService {
19
20     @Autowired
21     private PagamentoRepository repository;
22
23     @Transactional(readonly = true)
24     public Page<PagamentoDTO> findAll(Pageable pageable) {
25         Page<Pagamento> page = repository.findAll(pageable);
26         return page.map(PagamentoDTO::new);
27     }
```

Alterar PagamentoController - findAll

```
PagamentoController.java x
19 public class PagamentoController {
20
21     @Autowired
22     private PagamentoService service;
23
24     @GetMapping
25     public ResponseEntity<Page<PagamentoDTO>> findAll(
26         @PageableDefault(size = 10)Pageable pageable){
27         Page<PagamentoDTO> dto = service.findAll(pageable);
28         return ResponseEntity.ok(dto);
29     }
```

Testar o MS

```
1 {
2   "content": [
3     {
4       "id": 1,
5       "valor": 1200.00,
6       "nome": "NICODEMUS C SOUZA",
7       "numeroDoCartao": "1234567890123456",
8       "validade": "12/30",
9       "codigoDeSeguranca": "352",
10      "status": "CRIADO",
11      "pedidoId": 1,
12      "formaDePagamentoId": 2
13    },
14    {
15      "id": 2,
16      "valor": 500.50,
17      "nome": "AMADEUS MOZART",
18      "numeroDoCartao": "8597452369851245",
19      "validade": "05/28",
20      "codigoDeSeguranca": "647",
21      "status": "CRIADO",
22      "pedidoId": 5,
23      "formaDePagamentoId": 2
24    }
25  ]
26 }
```

```
111 {
112   "pageable": {
113     "pageNumber": 0,
114     "pageSize": 10,
115     "sort": {
116       "empty": true,
117       "sorted": false,
118       "unsorted": true
119     }
120   },
121   "offset": 0,
122   "paged": true,
123   "unpaged": false
124 },
125 {
126   "last": false,
127   "totalPages": 2,
128   "totalElements": 12,
129   "size": 10,
130   "number": 0,
131   "sort": {
132     "empty": true,
133     "sorted": false,
134     "unsorted": true
135   },
136   "first": true,
137   "numberOfElements": 10,
138   "empty": false
139 }
```

I Testar o MS

Query Params

```
http://localhost:8080/pagamentos?size=10&page=1&sort=nome,desc
```

Testar o MS

FIAP_2024_02 / MS-Pagamentos / Pagamentos Copy

GET <http://localhost:8080/pagamentos?page=1> Send

Params • Authorization Headers (6) Body Scripts Settings Cookies

Query Params

<input checked="" type="checkbox"/>	Key	Value	Description	...	Bulk Edit
<input checked="" type="checkbox"/>	page	1			

Body Cookies Headers (5) Test Results 200 OK 13 ms 851 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "content": [
3     {
4       "id": 11,
5       "valor": 500.50,
6       "nome": "Amadeus Mozart",
7       "numeroDoCartao": "8597452369851245",
8       "validade": "05/28",
9       "codigoDeSeguranca": "647",
10      "status": "CRIADO",
11      "pedidoId": 5,
12      "formaDePagamentoId": 2
13    },
14    {
15      "id": 12,
16      "valor": 1200.00,
17      "nome": "Maria Joaquina",
18      "numeroDoCartao": "2457896547123654",
19      "validade": "01/25",
20      "codigoDeSeguranca": "543",
21      "status": "CRIADO",
```

GET <http://localhost:8080/pagamentos?size=10&page=0&sort=nome,desc> Send

Params • Authorization Headers (6) Body Scripts Settings Cookies

Query Params

<input checked="" type="checkbox"/>	Key	Value	Description	...	Bulk Edit
<input checked="" type="checkbox"/>	size	10			
<input checked="" type="checkbox"/>	page	0			
<input checked="" type="checkbox"/>	sort	nome,desc			

Body Cookies Headers (5) Test Results 200 OK 12 ms 2.28 KB Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "content": [
3     {
4       "id": 1,
5       "valor": 1200.00,
6       "nome": "Nicodemus C Souza",
7       "numeroDoCartao": "1234567890123456",
8       "validade": "12/30",
9       "codigoDeSeguranca": "352",
10      "status": "CRIADO",
11      "pedidoId": 1,
12      "formaDePagamentoId": 2
13    },
14    {
15      "id": 10,
16      "valor": 1200.00,
17      "nome": "Nicodemus C Souza",
```



Copyright © 2024
Prof^a. Aparecida de Fátima Castello Rosa

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).