MÓDULO 2 - EXTRA
MENSAGERIA COM KAFKA e QUARKUS

UNIPDS
EDUCAÇÃO

MENSAGERIA

É CHATO ALUGAR IMÓVEL, NÉ?

MAS JÁ FOI MAIS CHATO...

Antigamente

Comunicação Síncrona

Hoje

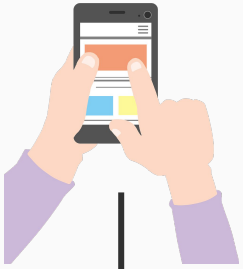
Intermediário

Comunicação Assíncrona

MAS O QUE ISSO TEM A VER COM SOFTWARE?

INTEGRAÇÃO DE SISTEMAS

Florinda Eats



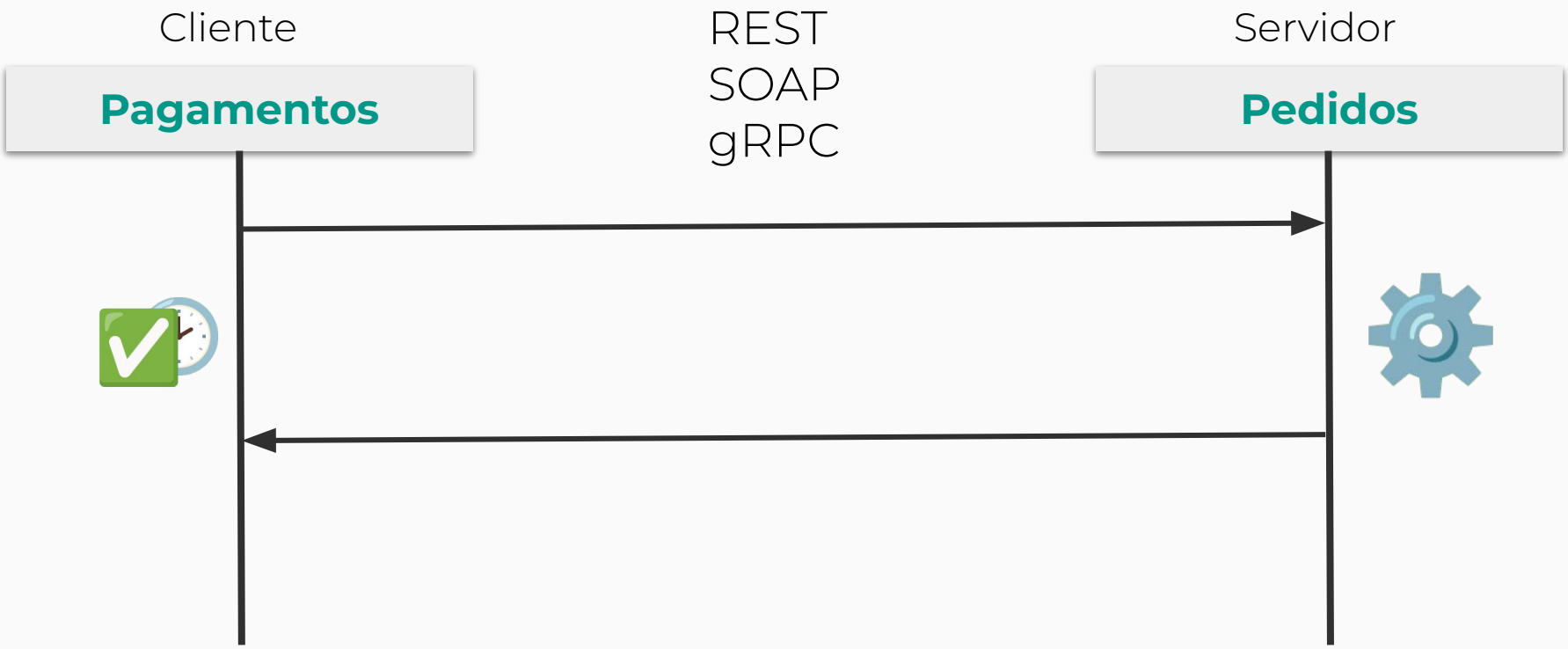
Pagamentos

?

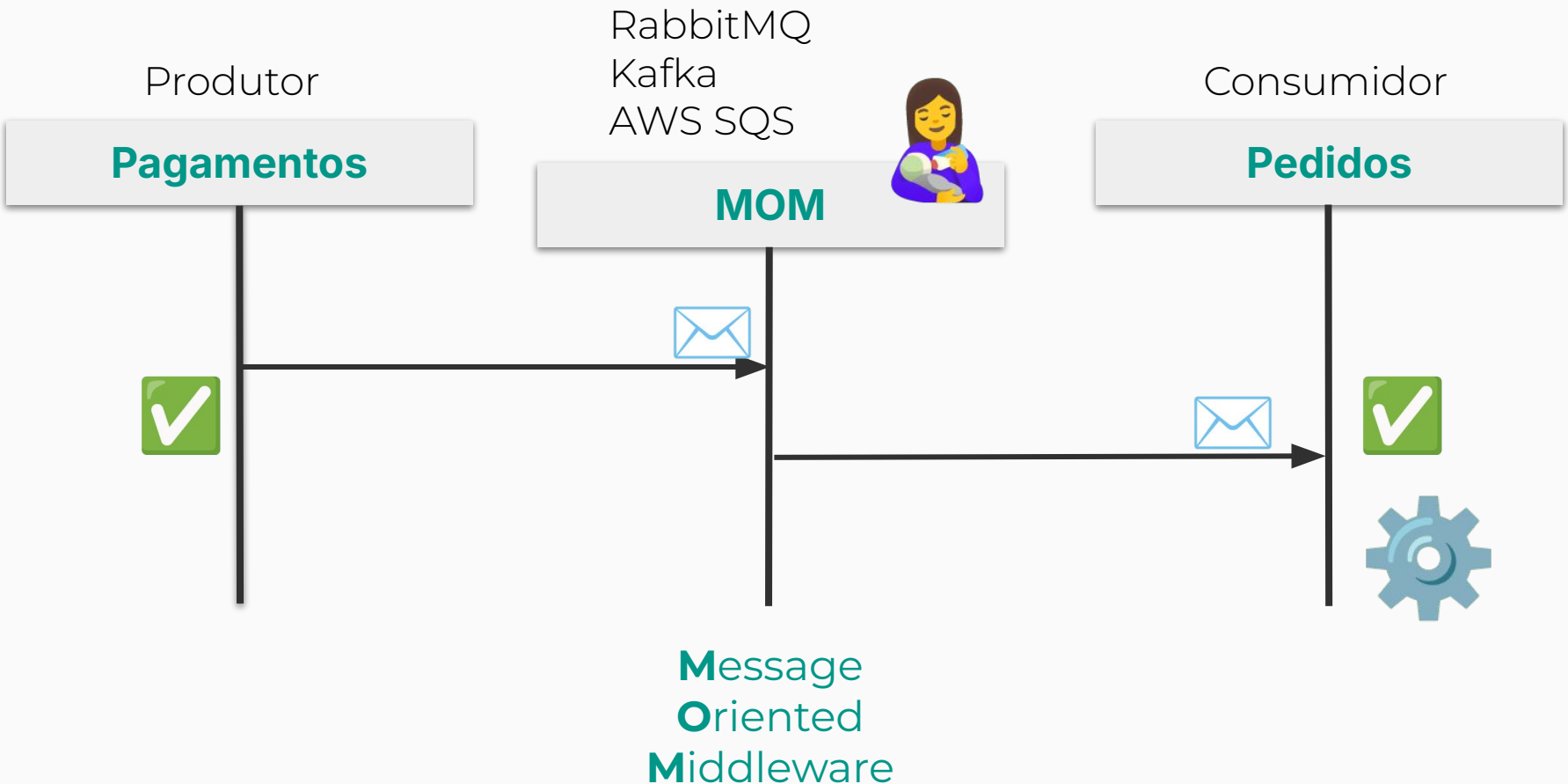
Nota Fiscal

Pedidos

Integração Síncrona



Integração Assíncrona



MENSAGENS?

Mensagem

Documento

- Usada para transmitir dados entre aplicações

Comando

- Invocação de um método em outra aplicação
- Apenas a requisição, sem uma resposta

Evento

- Notificação que algo aconteceu

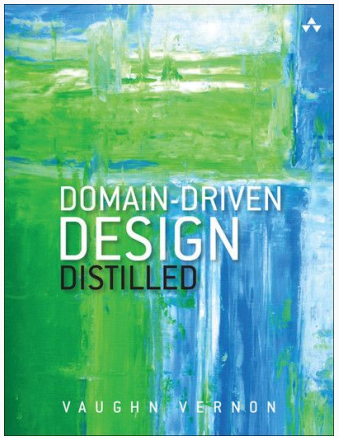
Event-Driven Architecture

Eventos de Domínio

> **Domain Event** é uma ocorrência significativa em termos de negócio em um determinado Contexto Delimitado.

Vaughn Vernon

Domain-Driven Design Distilled (2016)



Eventos de Domínio

ProdutoCriado

ReleaseAgendada

PedidoRealizado

PagamentoConfirmado

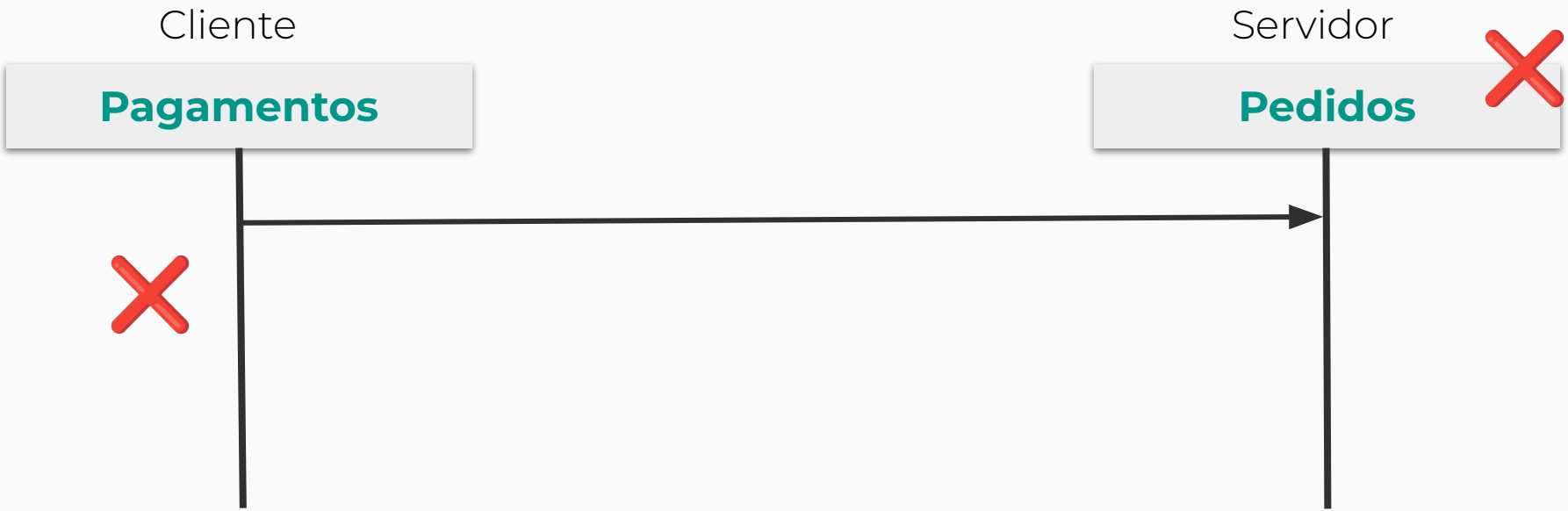
NotaFiscalEmitida

EnderecoAlterado

CARACTERÍSTICAS ARQUITETURAIS

INTEGRAÇÃO SÍNCRONA vs ASSÍNCRONA

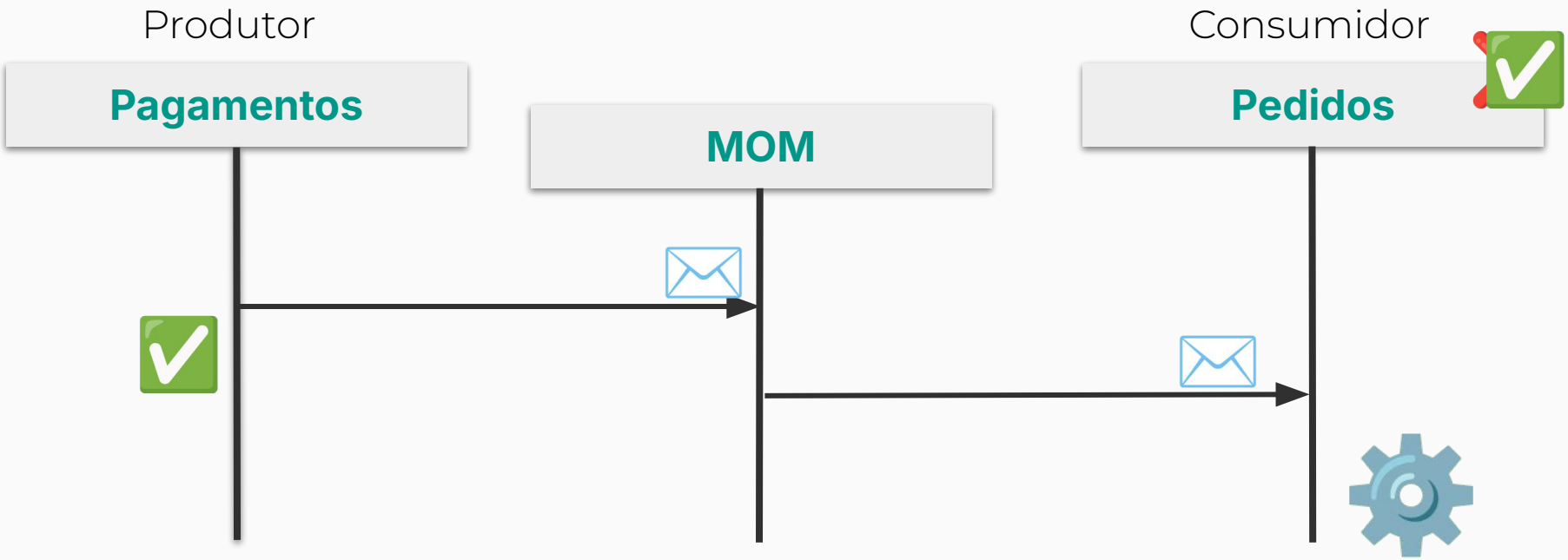
Integração Síncrona



falta de **Disponibilidade**

Possível Solução: Redundância e Balanceamento de Carga

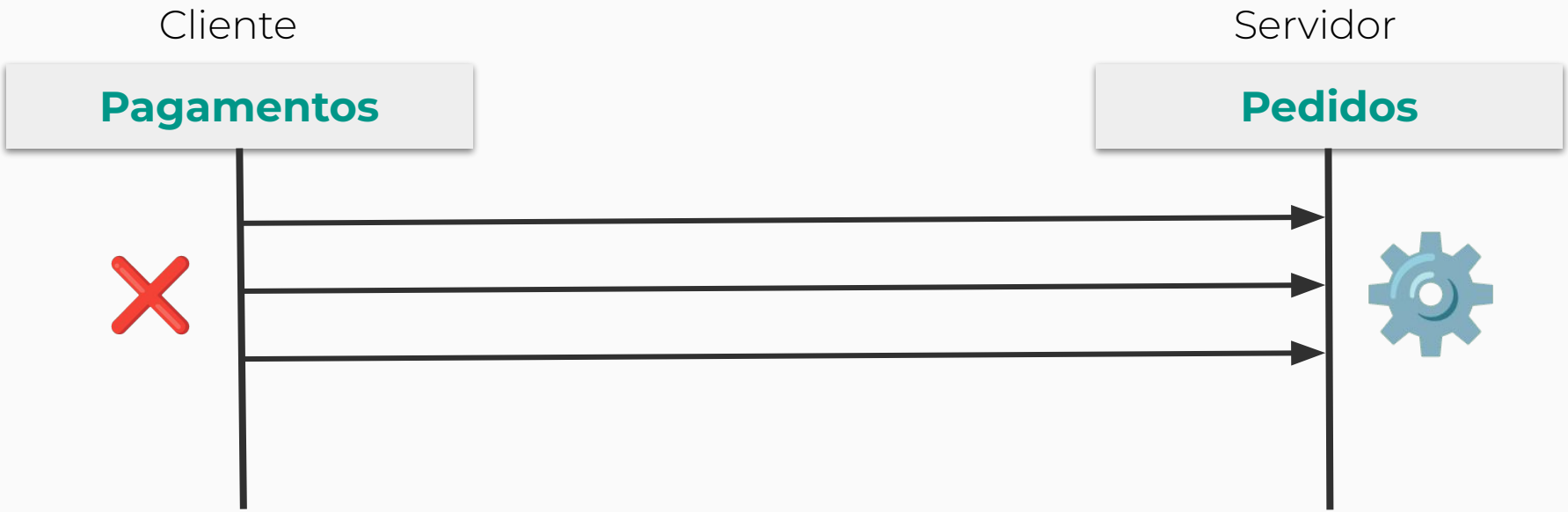
Integração Assíncrona



Produtor envia mensagens mesmo com consumidor fora do ar

Disponibilidade

Integração Síncrona

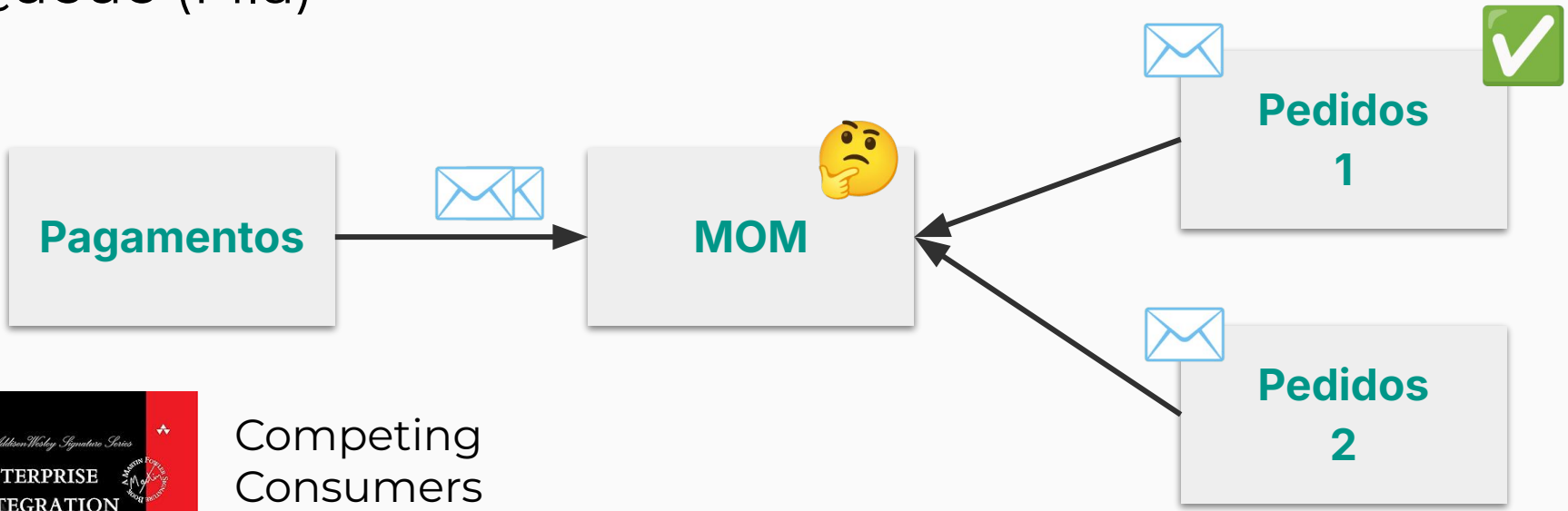


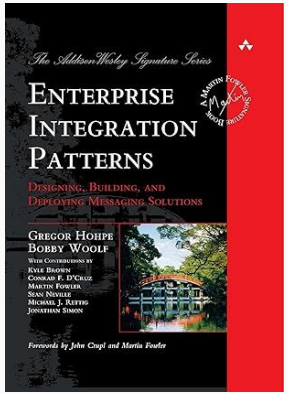
falta de **Escalabilidade**

Possível Solução: Redundância e Balanceamento de Carga

Point-to-Point Channel

Queue (Fila)





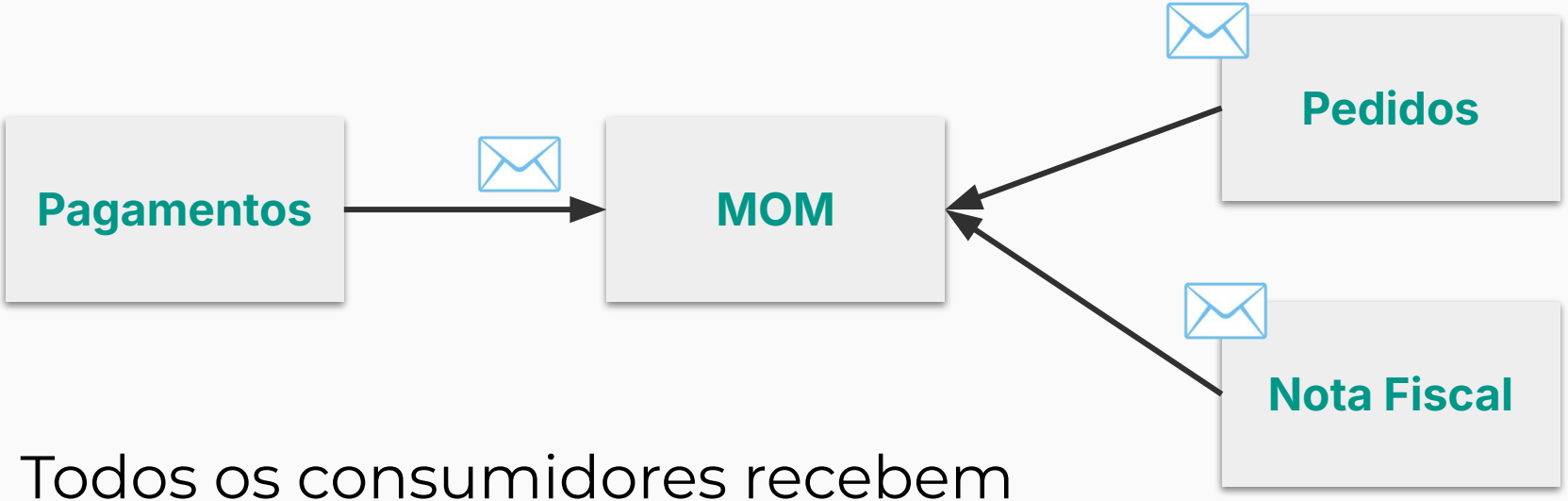
Competing
Consumers
Pattern

Escalabilidade Horizontal

Publisher-Subscriber

Channel

Topic (Tópico)



Todos os consumidores recebem todas as mensagens

Desacoplamento

KAFKA

Conceitos fundamentais

Tópicos log

Partições

Mesma chave, mesma partição

Consumer Groups

record, não mensagem

Executando Kafka via Container

docker-compose.yml

```yaml
services:
  kafka:
    image: 'bitnami/kafka:latest'
    ports:
      - '9094:9094'
    networks:
      - florinda-eats-network
    environment:
      - KAFKA_CFG_NODE_ID=0
      - KAFKA_CFG_PROCESS_ROLES=controller,broker
      - KAFKA_CFG_LISTENERS=PLAINTEXT://:9092,CONTROLLER://:9093,EXTERNAL://:9094
      - KAFKA_CFG_ADVERTISED_LISTENERS=PLAINTEXT://kafka:9092,EXTERNAL://localhost:9094
      - KAFKA_CFG_LISTENER_SECURITY_PROTOCOL_MAP=CONTROLLER:PLAINTEXT,EXTERNAL:PLAINTEXT,PLAINTEXT:PLAINTEXT
      - KAFKA_CFG_CONTROLLER_QUORUM_VOTERS=0@kafka:9093
      - KAFKA_CFG_CONTROLLER_LISTENER_NAMES=CONTROLLER
networks:
  florinda-eats-network:
```

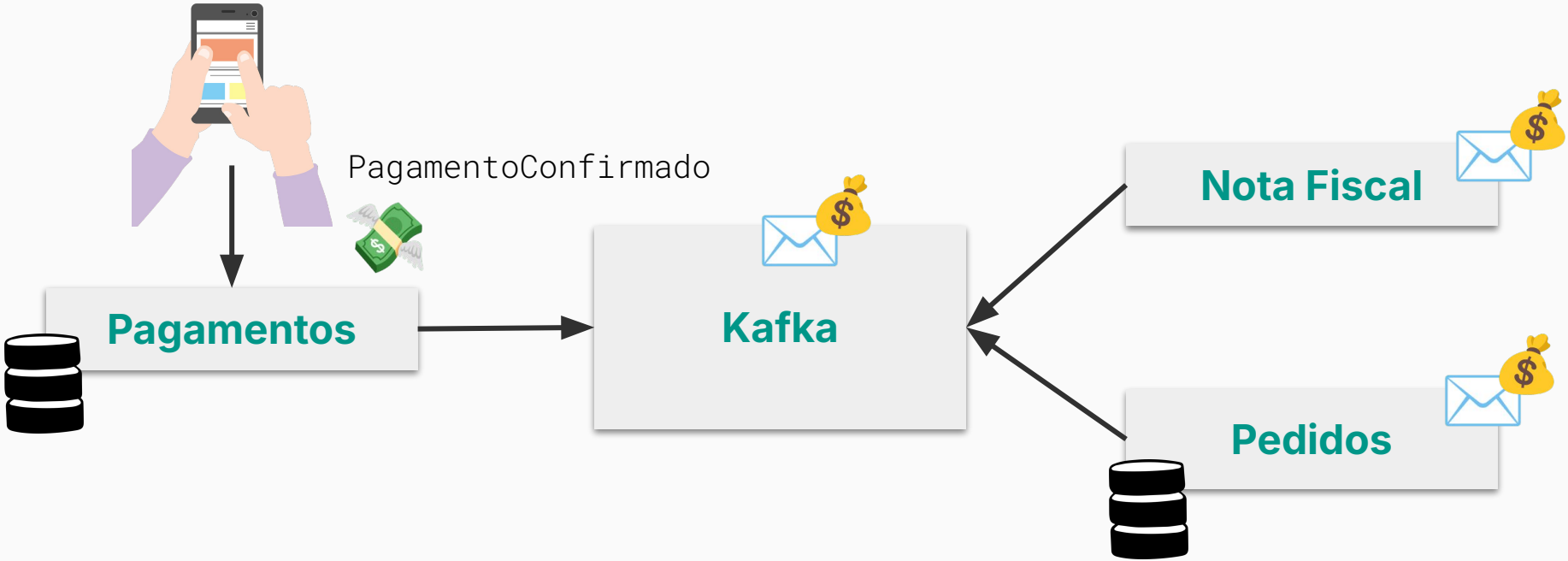
https://github.com/unipds-projetos/florinda-eats/blob/main/docker-compose.yml

Executando Kafka via Container

docker compose up

```
✔ Network florinda-eats-microservices_florinda-eats-network  Created                    0.1s
 ⠟ Container florinda-eats-microservices-kafka-1             Created                    0.0s
Attaching to kafka-1
kafka-1  | kafka 16:13:02.42 INFO  ==> Welcome to the Bitnami kafka container
...
kafka-1  | kafka 16:13:02.42 INFO  ==> ** Starting Kafka setup **
...
kafka-1  | kafka 16:13:05.24 INFO  ==> Initializing KRaft storage metadata
...
kafka-1  | kafka 16:13:12.12 INFO  ==> ** Starting Kafka **
...
kafka-1  | [2024-10-27 16:13:19,826] INFO Kafka version: 3.7.0 (org.apache.kafka.common.utils.AppInfoParser)
kafka-1  | [2024-10-27 16:13:19,826] INFO Kafka commitId: 2ae524ed625438c5
(org.apache.kafka.common.utils.AppInfoParser)
kafka-1  | [2024-10-27 16:13:19,826] INFO Kafka startTimeMs: 1730045599825
(org.apache.kafka.common.utils.AppInfoParser)
kafka-1  | [2024-10-27 16:13:19,828] INFO [KafkaRaftServer nodeId=0] Kafka Server started
(kafka.server.KafkaRaftServer)
```

Florinda Eats



Sistemas reagindo ao evento

Criando um tópico (via CLI)

```
docker exec -it florinda-eats-kafka-1 kafka-topics.sh
--bootstrap-server localhost:9092
--create --partitions 2 --topic pagamentosConfirmados

Created topic pagamentosConfirmados.
```

```
docker exec -it florinda-eats-kafka-1 kafka-topics.sh
--bootstrap-server localhost:9092
--describe --topic pagamentosConfirmados

Topic: pagamentosConfirmados  TopicId: dMu4BDHmTaOOWBoYASo1DQ
       PartitionCount: 2  ReplicationFactor: 1  Configs:
  Topic: pagamentosConfirmadosPartition: 0 Leader: 0    Replicas: 0  Isr: 0
  Topic: pagamentosConfirmadosPartition: 1 Leader: 0    Replicas: 0  Isr: 0
```

Produzindo uma mensagem (via CLI)

```
docker exec -it florinda-eats-kafka-1 kafka-console-producer.sh
--bootstrap-server localhost:9092 --topic pagamentosConfirmados --property
“parse.key=true”--property “key.separator=;”
```

```
> 1; {“pagamentoId”: 1, “pedidoId”: 1}
```

Consumindo uma mensagem (via CLI)

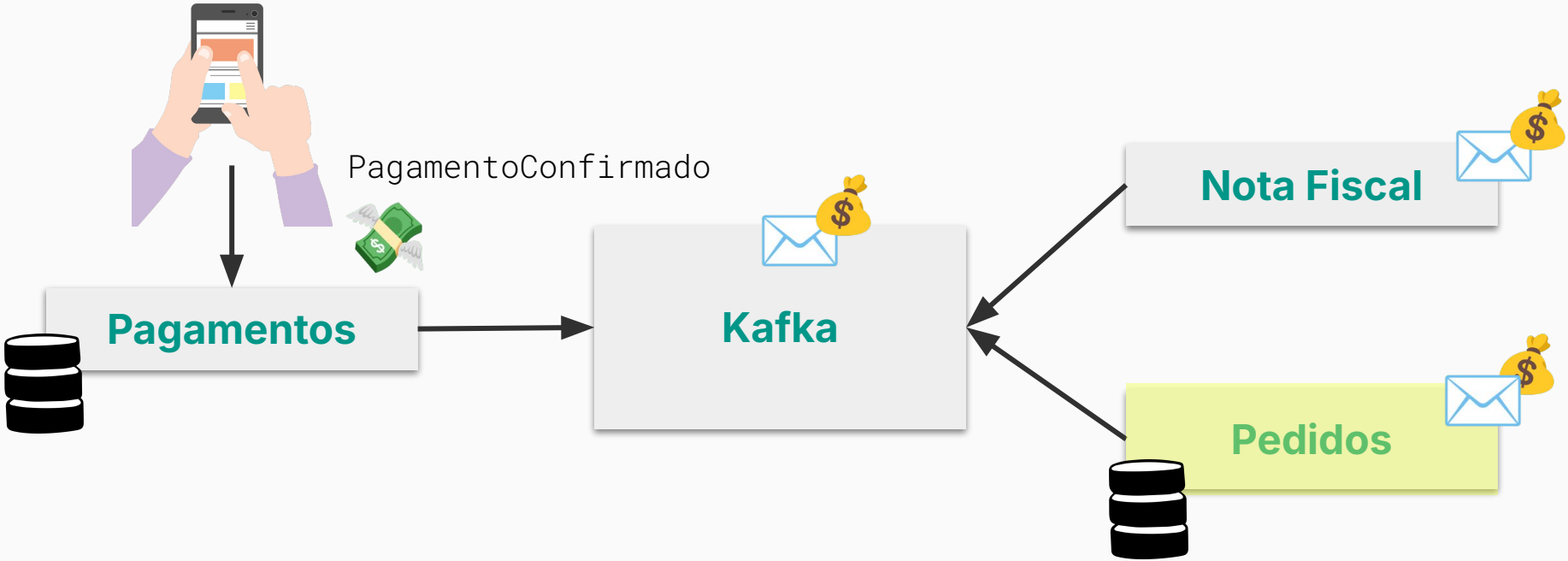
```
docker exec -it florinda-eats-kafka-1 kafka-console-consumer.sh
--bootstrap-server localhost:9092 --topic pagamentosConfirmados --from-beginning
--group teste

{“pagamentoId”: 1, “pedidoId”: 1}
```

Verificando consumer groups (via CLI)

```
docker exec -it florinda-eats-kafka-1 kafka-consumer-groups.sh
--bootstrap-server localhost:9092 --all-groups --describe


GROUP           TOPIC                   PARTITION  CURRENT-OFFSET  LOG-END-OFFSET
LAG             CONSUMER-ID
HOST            CLIENT-ID

teste           pagamentosConfirmados 0          0               0
0               console-consumer-f9f08bdc-188b-44a1-a223-96359a9245d0
/172.19.0.2     console-consumer

teste           pagamentosConfirmados 1          2               2
0               console-consumer-f9f08bdc-188b-44a1-a223-96359a9245d0
/172.19.0.2     console-consumer
```

QUARKUS REACTIVE MESSAGING

Florinda Eats

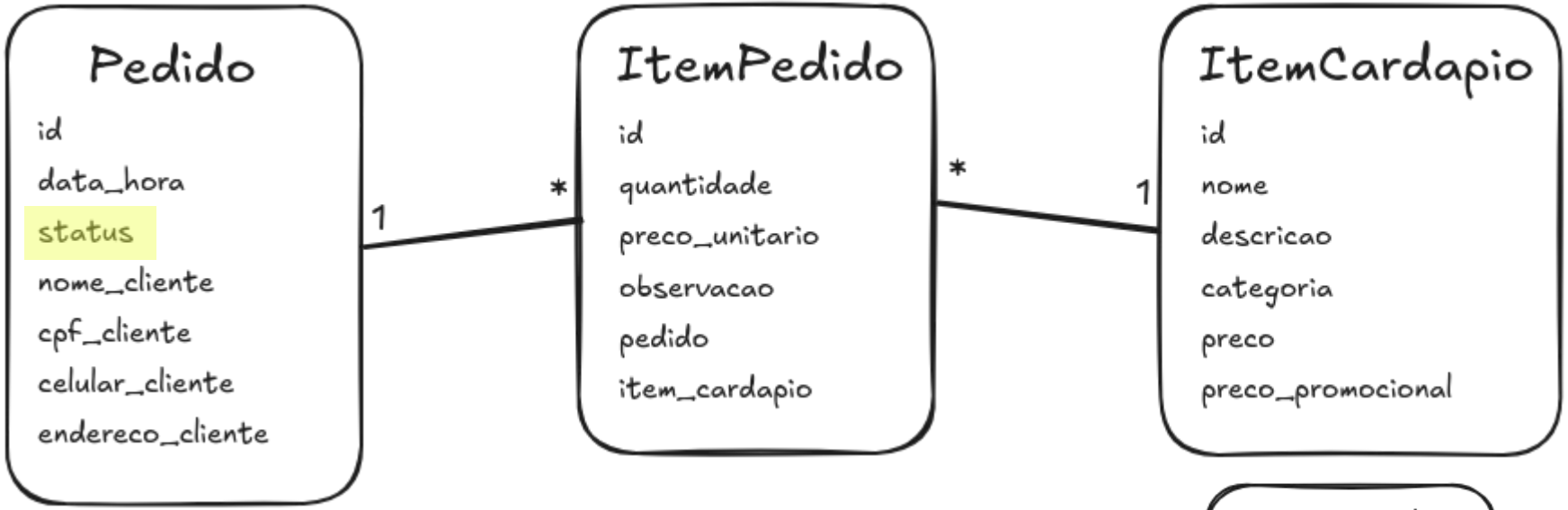


https://github.com/unipds-projetos/florinda-eats

CODANDO UM CONSUMIDOR

PEDIDOS

Pedidos - modelo de dados



StatusPedido

REALIZADO
PAGO
CONFIRMADO
PRONTO
SAIU_PRA_ENTREGA
ENTREGUE

Atualizar **status do Pedido** para **PAGO** quando o evento **PagamentoConfirmado** chegar.

CategoriaCardapio

ENTRADAS
PRATOS PRINCIPAIS
BEBIDAS
SOBREMESA

Pedidos - extensão Kafka

```
quarkus ext add messaging-kafka
```

Consumer Group é o **artifactId** (`florinda-eats-pedidos`)

```
<dependency>
    <groupId>io.quarkus</groupId>
    <artifactId>quarkus-rest-jackson</artifactId>
</dependency>
<dependency>
    <groupId>io.quarkus</groupId>
    <artifactId>quarkus-messaging-kafka</artifactId>
</dependency>
<dependency>
```

Apontando para o Kafka correto e desligando container automático do Kafka (DevServices)

`application.properties`

```
kafka.bootstrap.servers=localhost:9094
```

Pedidos - objeto do evento

```
package mx.florinda.pedido;

public class PagamentoConfirmadoEvent {

  public Long pagamentoId;
  public Long pedidoId;

  @Override
  public String toString() {
    // ...
  }
}
```

Pedidos - consumidor

```
package mx.florinda.pedido;

import io.quarkus.hibernate.reactive.panache.Panache;
import io.smallrye.mutiny.Uni;
import jakarta.enterprise.context.ApplicationScoped;
import org.eclipse.microprofile.reactive.messaging.Incoming;

@ApplicationScoped
public class PagamentoConfirmadoConsumer {

  @Incoming("pagamentosConfirmados")
  public Uni<Void> consome(PagamentoConfirmadoEvent evento) {
    return Panache.withTransaction(() ->
            Pedido.<Pedido>findById(evento.pedidoId)
                    .onItem().ifNotNull().invoke(pedido -> {
                      pedido.status = StatusPedido.PAGO;
                    })).replaceWithVoid();
  }

}
```

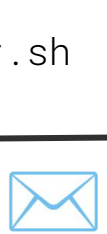
Pedidos - executando a aplicação

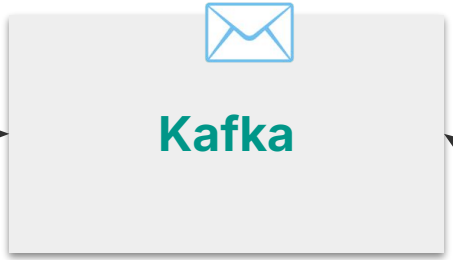
```
quarkus dev

2024-10-27 17:53:10,555 INFO  [io.qua.sma.dep.processor] (build-39)
Configuring the channel 'pagamentosConfirmados' to be managed by the connector
'smallrye-kafka'
2024-10-27 17:53:10,560 INFO  [io.qua.sma.dep.processor] (build-4)
 Generating Jackson deserializer for type mx.florinda.pedido.PagamentoConfirmadoEvent
2024-10-27 17:57:55,327 INFO  [io.sma.rea.mes.kafka] (Quarkus Main Thread) SRMSG18229:
Configured topics for channel 'pagamentosConfirmados': [pagamentosConfirmados]
2024-10-27 17:57:55,334 INFO  [io.sma.rea.mes.kafka](Quarkus Main Thread) SRMSG18214:
Key deserializer omitted, using String as default
2024-10-27 17:57:55,593 INFO  [io.sma.rea.mes.kafka]
(smallrye-kafka-consumer-thread-0) SRMSG18257: Kafka consumer
kafka-consumer-pagamentosConfirmados, connected to Kafka brokers 'localhost:9094',
belongs to the 'florinda-eats-pedidos' consumer group and is configured to poll
records from [pagamentosConfirmados]
2024-10-27 18:40:01,208 INFO  [io.sma.rea.mes.kafka] (vert.x-eventloop-thread-4)
SRMSG18256: Initialize record store for topic-partition 'pagamentosConfirmados-1' at
position 1.
```

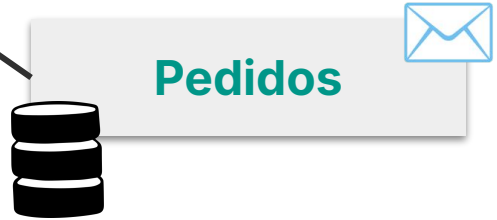
Pedidos - testando evento

kafka-console-producer.sh

```
> {“pagamentoId”: 1,
   “pedidoId”: 1,
   “Valor”: 9.48 }
```

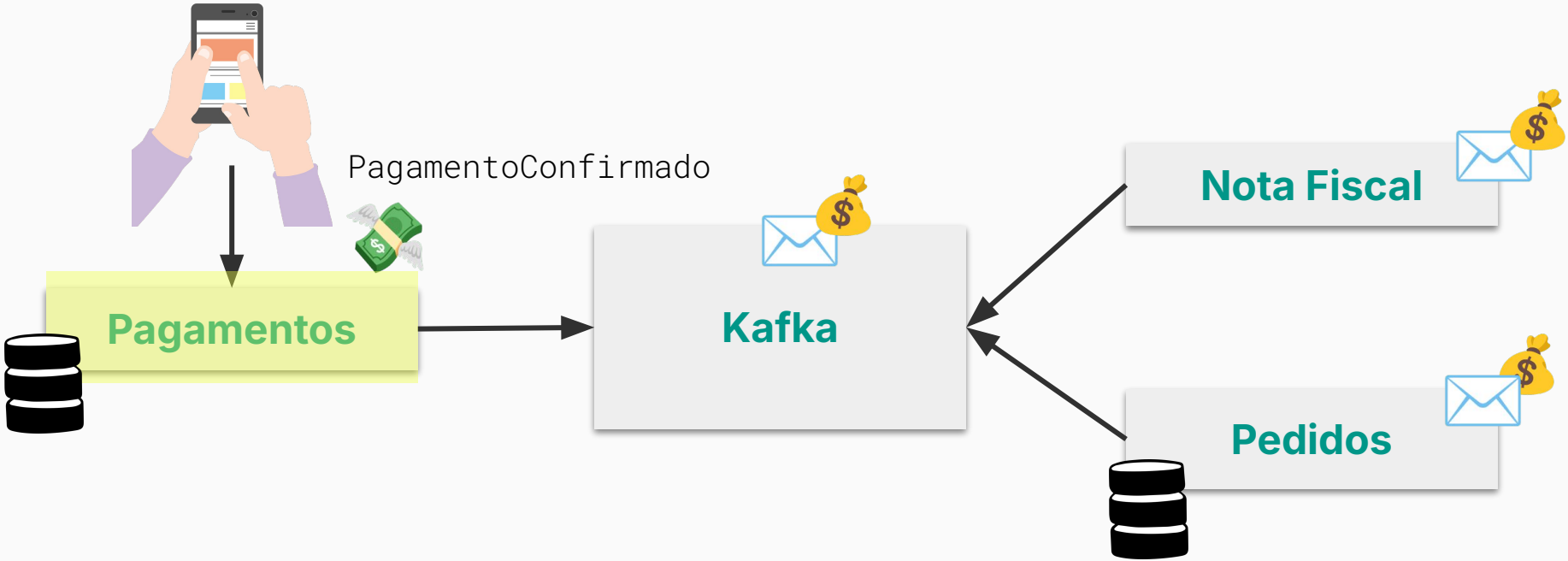






https://github.com/unipds-projetos/florinda-eats/tree/1-consumer-pedidos

Florinda Eats



CODANDO UM PRODUTOR

PAGAMENTOS

Pagamentos - extensão Kafka

```
quarkus ext add messaging-kafka
```

Consumer Group é o **artifactId** (`florinda-eats-pagamentos`)

```
<dependency>
    <groupId>io.quarkus</groupId>
    <artifactId>quarkus-rest-jackson</artifactId>
</dependency>
<dependency>
    <groupId>io.quarkus</groupId>
    <artifactId>quarkus-messaging-kafka</artifactId>
</dependency>
<dependency>
```

Apontando para o Kafka correto

`application.properties`

```
kafka.bootstrap.servers=localhost:9094
```

Pagamentos - objeto do evento

```
package mx.florinda.pedido;

import java.math.BigDecimal;

public class PagamentoConfirmadoEvent {

  public Long pagamentoId;
  public Long pedidoId;
  public BigDecimal valor;

  public PagamentoConfirmadoEvent(Long pagamentoId, Long pedidoId, BigDecimal valor) {
    this.pagamentoId = pagamentoId;
    this.pedidoId = pedidoId;
    this.valor = valor;
 }

  @Override
  public String toString() {
    // ...
  }
}
```

Pagamentos - produtor

```
package mx.florinda.pagamento;

// imports...
import org.eclipse.microprofile.reactive.messaging.Channel;
import org.eclipse.microprofile.reactive.messaging.Emitter;

@Path("/pagamentos")
public class PagamentoResource {

  @Inject
  @Channel("pagamentosConfirmados")
  Emitter<PagamentoConfirmadoEvent> emitter;

  @PUT
  @Path("/{id}")
  public  Uni<Pagamento> confirma(Long id) {
    return Panache.withTransaction(() ->
            Pagamento.<Pagamento>findById(id)
                    .onItem().ifNotNull().invoke(pagamento -> {
                      pagamento.status = StatusPagamento.CONFIRMADO;
                    }));
  }

}
```

Pagamentos - produtor (cont.)

```
package mx.florinda.pagamento;

// imports...

@Path("/pagamentos")
public class PagamentoResource {

 // inject do Emitter...

  @PUT
  @Path("/{id}")
  public  Uni<Pagamento> confirma(Long id) {
    return Panache.withTransaction(() ->
            Pagamento.<Pagamento>findById(id)
                    .onItem().ifNotNull().invoke(pagamento -> {
                      pagamento.status = StatusPagamento.CONFIRMADO;

                      PagamentoConfirmadoEvent evento =
                          new PagamentoConfirmadoEvent(pagamento.id, pagamento.pedidoId,
                                                       pagamento.valor);

                      emitter.send(evento);

                    }));
  }
```

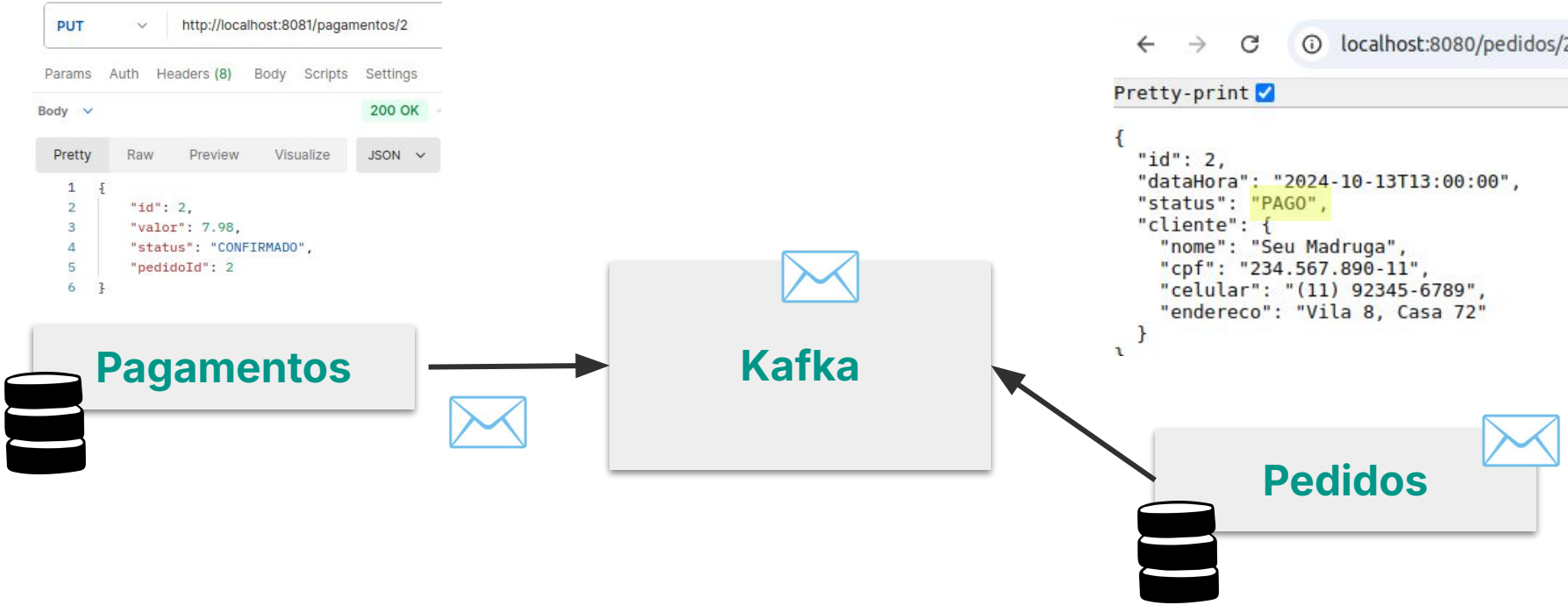
Alternativa:

```
@Outgoing("pagamentosConfirmados")
```

Pagamentos - executando a aplicação

```
quarkus dev

2024-10-27 21:09:46,142 INFO  [io.qua.sma.dep.processor] (build-19)
Configuring the channel 'pagamentosConfirmados' to be managed by the
connector 'smallrye-kafka'
2024-10-27 21:09:46,148 INFO  [io.qua.sma.dep.processor] (build-45)
Generating Jackson serializer for type
mx.florinda.pagamento.PagamentoConfirmadoEvent

2024-10-27 21:10:17,967 INFO  [io.sma.rea.mes.kafka]
(smallrye-kafka-producer-thread-0) SRMSG18258:
Kafka producer kafka-producer-pagamentosConfirmados, connected to Kafka
brokers 'localhost:9094',
is configured to write records to 'pagamentosConfirmados'
```

Pagamentos - testando evento



https://github.com/unipds-projetos/florinda-eats/tree/2-producer-pagamentos

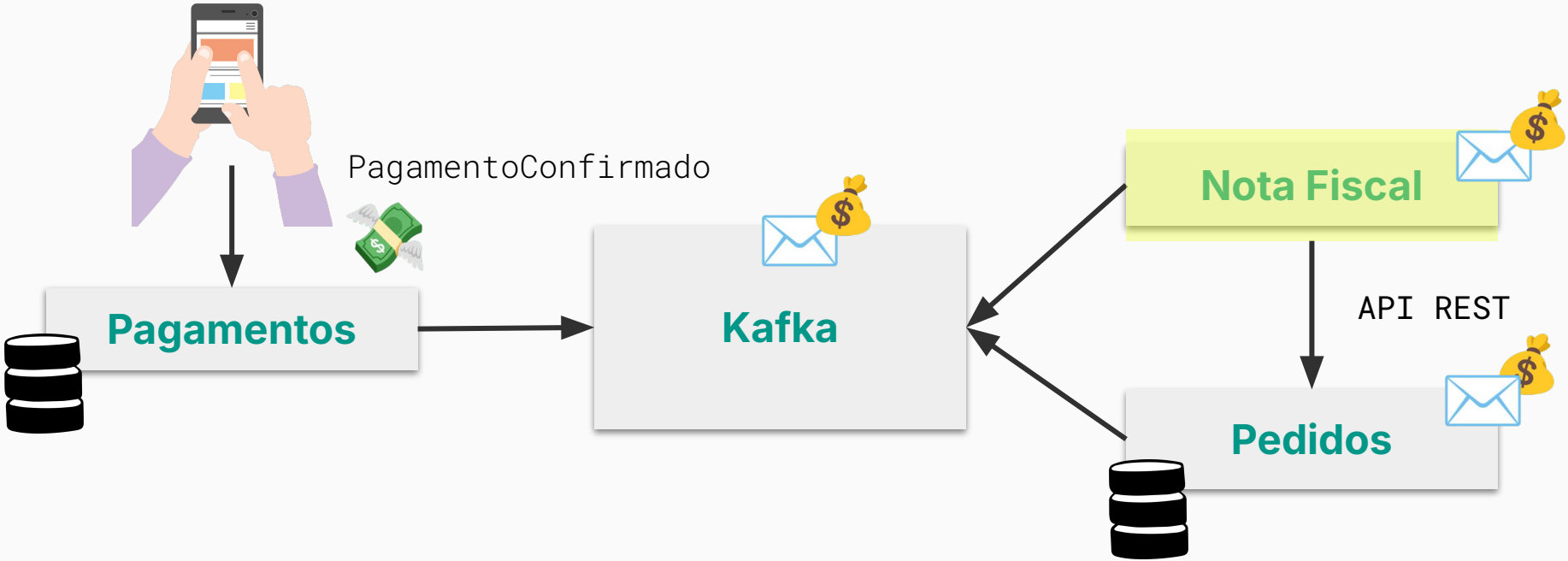
EXERCÍCIO PRÁTICO

Consumidor no serviço de Notas Fiscais.

Quando houver um novo evento de pagamento confirmado, deve ser gerada uma nota fiscal com os dados do cliente.

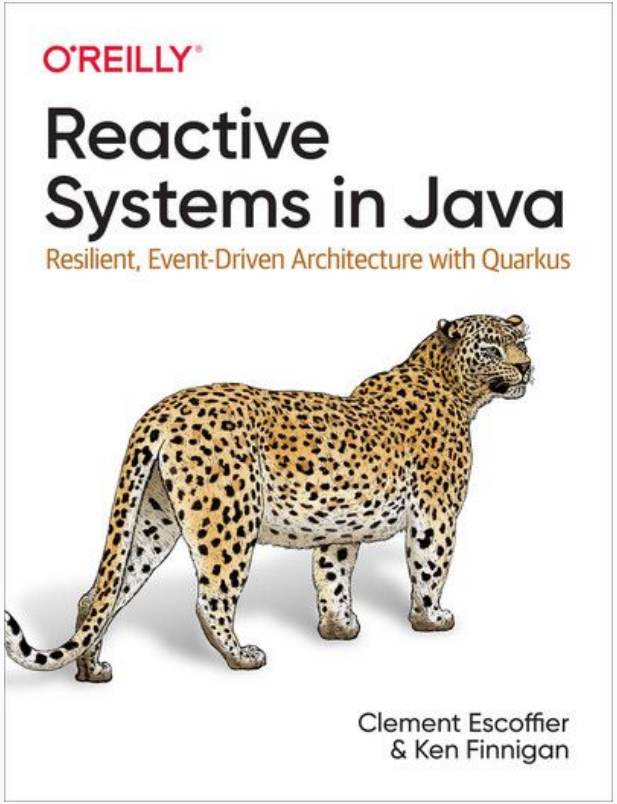
```
<xml>
    <valor>8.49</valor>
    <cliente>
        <nome>Jaiminho</nome>
        <cpf>345.890.123-22</cpf>
        ...
    </cliente>
</xml>
```

Florinda Eats



https://github.com/unipds-projetos/florinda-eats/tree/3-notas-fiscais

LEITURA COMPLEMENTAR



Reactive Systems in Java

Resilient, Event-Driven Architecture with Quarkus

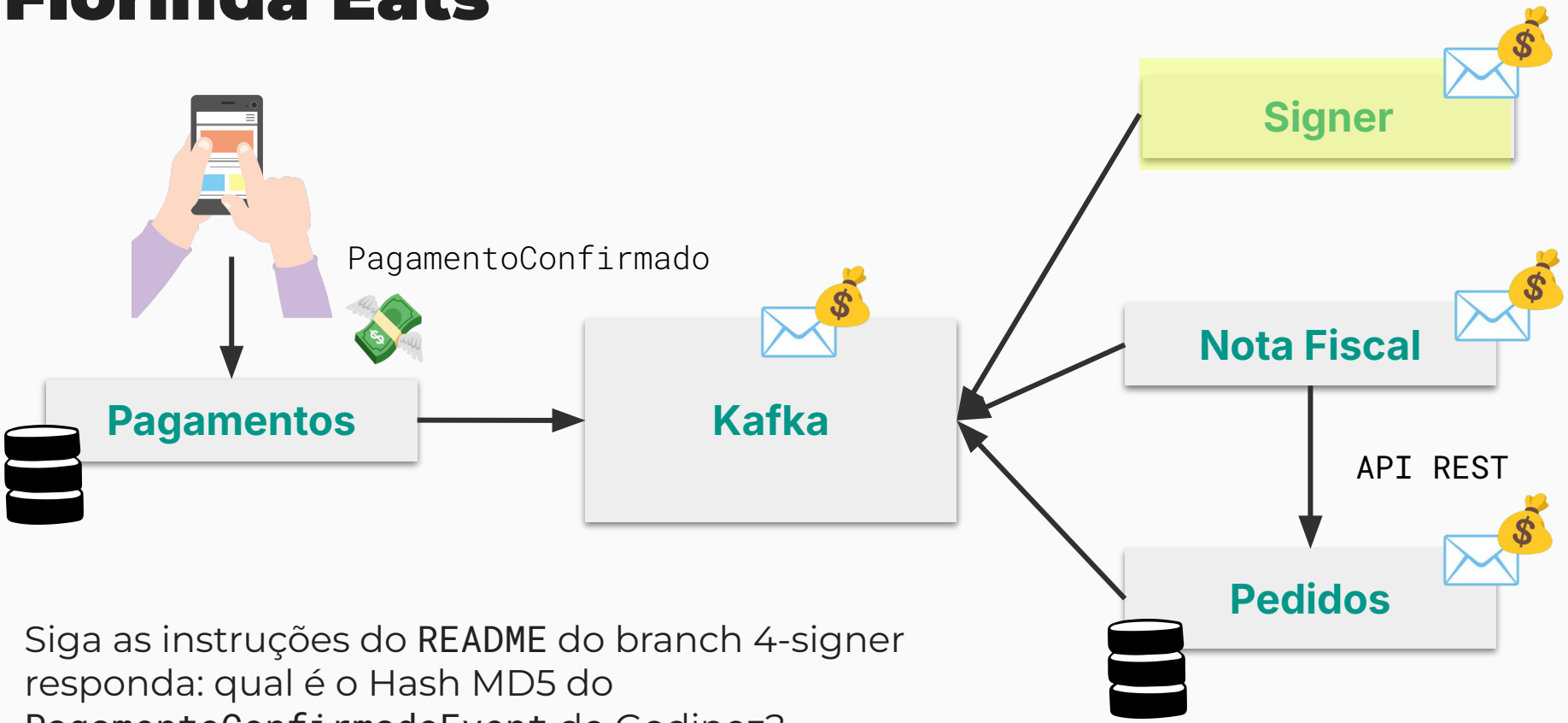
Novembro/2021

Clement Escoffier, Reactive Architect na Red Hat, contribui no código do Vert.x e do Quarkus.

Ken Finnigan, Senior Principal Software Engineer na Red Hat, líder do SmallRye e parte do time do Quarkus.

DESAFIO

Florinda Eats



Siga as instruções do `README` do branch 4-signer
responda: qual é o Hash MD5 do
`PagamentoConfirmadoEvent` de Godinez?

https://github.com/unipds-projetos/florinda-eats/tree/4-signer