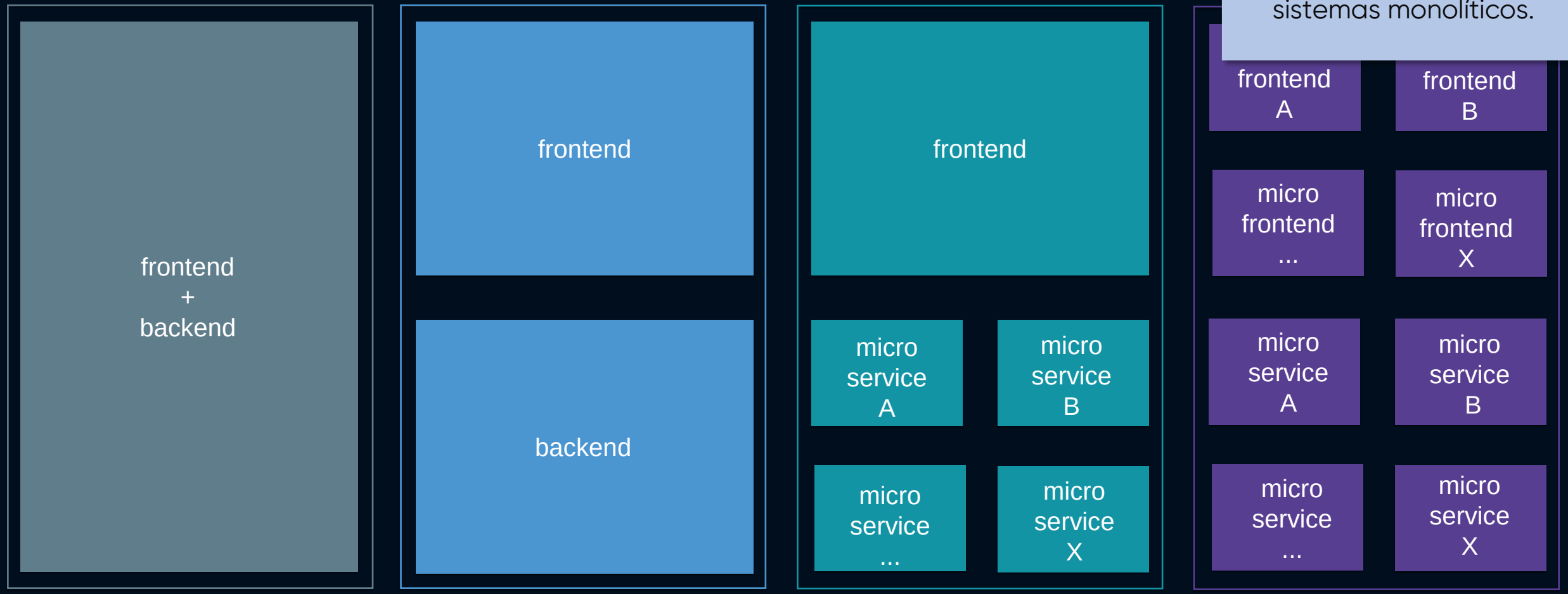
DECODER WEEK

Arquitetando o sistema Food4You

DIA 1

Evolução Arquitetural

Arquitetura de Microservices é uma solução as limitações dos sistemas monolíticos.



evolução arquitetural

COMPLEMENTANDO

Principais Benefícios dos Microservices

- Manutenibilidade
- Alta Disponibilidade
- Flexibilidade Tecnológica
- Independência das Equipes
- Melhor Performance
- Divisão da Complexidade do Negócio
- Isolamento a Falhas
- Alta Escalabilidade
- Baixo Acoplamento
- Melhor Testabilidade
- Agilidade nas Mudanças
- Isolamento Modelagem Dados
- Aumento Resiliência

DESCOMPLICANDO

#decoderweek

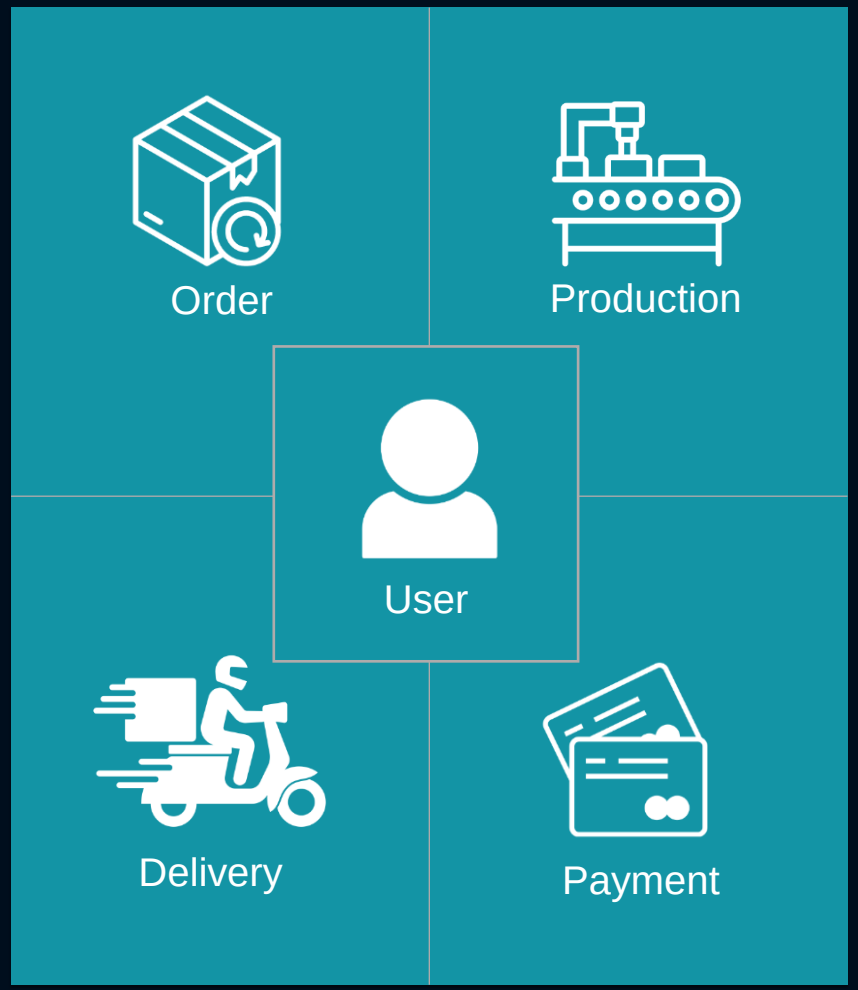
Arquitetura de Microservices é um modelo evolutivo ...

Modelo que mais facilmente se adapta as mudanças dos negócios e as modernizações tecnológicas.

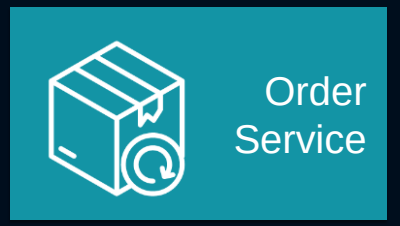
Boas práticas e Padrões na modelagem de uma Arquitetura de Microservices com Spring

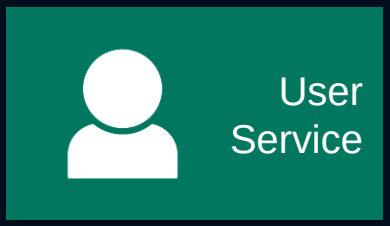
Modelagem na Prática

Divisão do Negócio em Microservices



Food4You domain





Granularidade dos Microservices

Order Service

User Service

Production Service

Payment Service

Delivery Service

Kitchen Service

PagSeguro Service

Preparation Service

Paypal Service

Packaging Service

É essencial o conhecimento profundo do negócio.

Arquitetura de Microservices – **Food4You App**

Microservices com Spring Boot

Spring Projects

Spring Boot

COMPLEMENTANDO

Spring Boot

COMPLEMENTANDO

Spring Projects

Spring Boot 

Spring Security 

Spring Cloud 

Spring Data 

Spring Web 

Spring AMQP 

Spring Hateoas 

Spring Batch 

Spring Framework 

Maturidade do ecossistema Spring

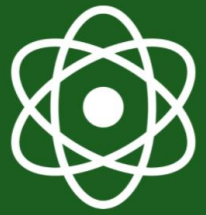
Microservices

Cloud

Event Driven

Batch

Reactive



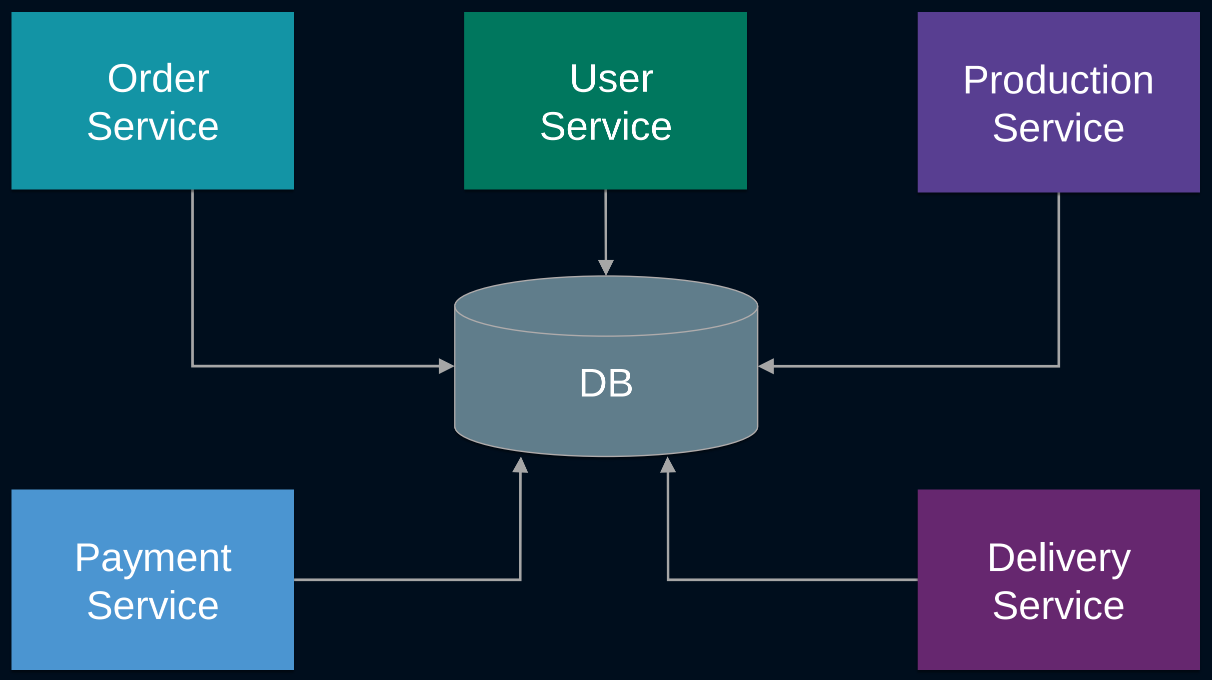
Database

Modelando Base de Dados em Microservices

Base de dados compartilhada	X	Base de dados por Microservice

Base de Dados Compartilhada

Arquitetura do Food4You com base de dados compartilhada

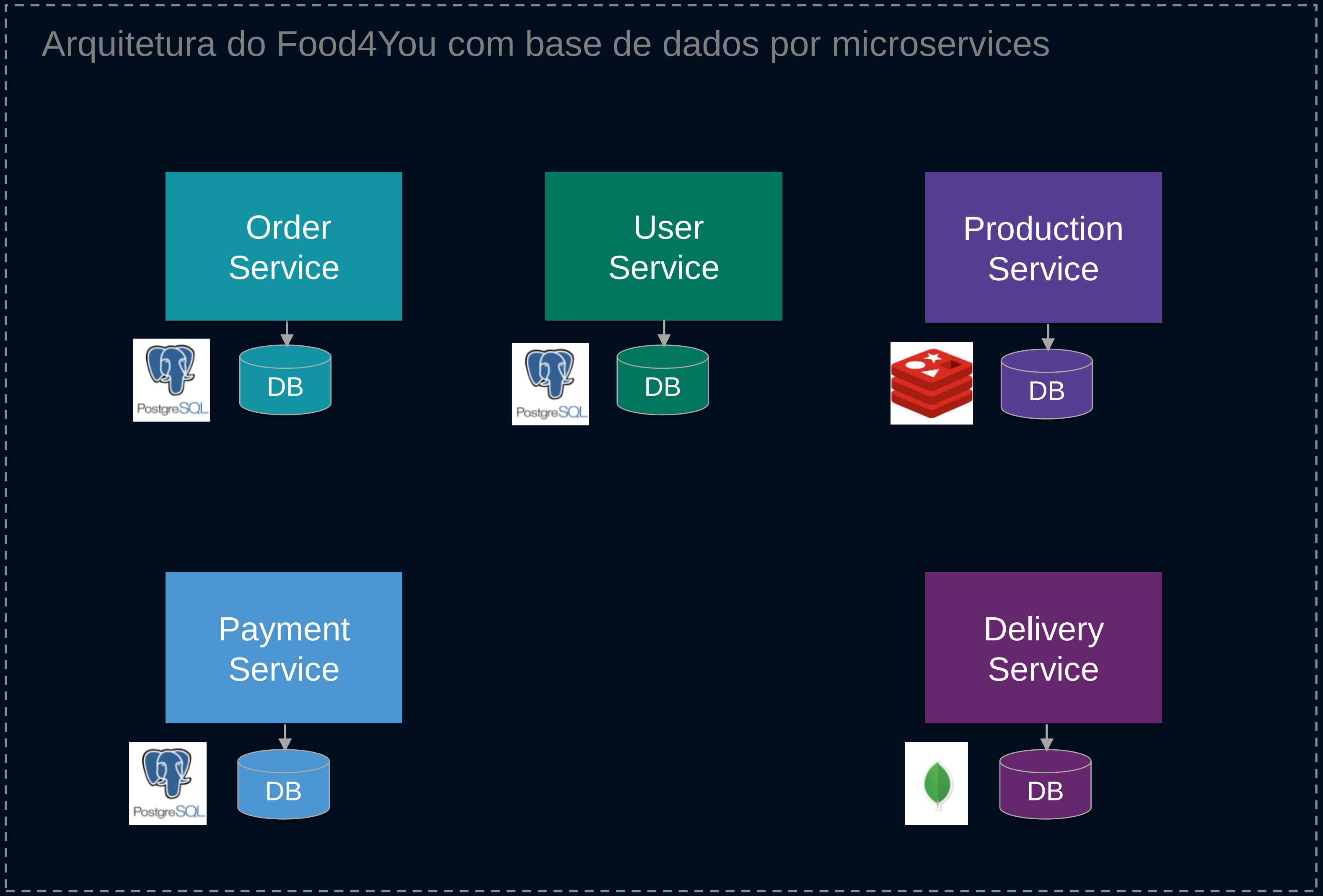


DESCOMPLICANDO

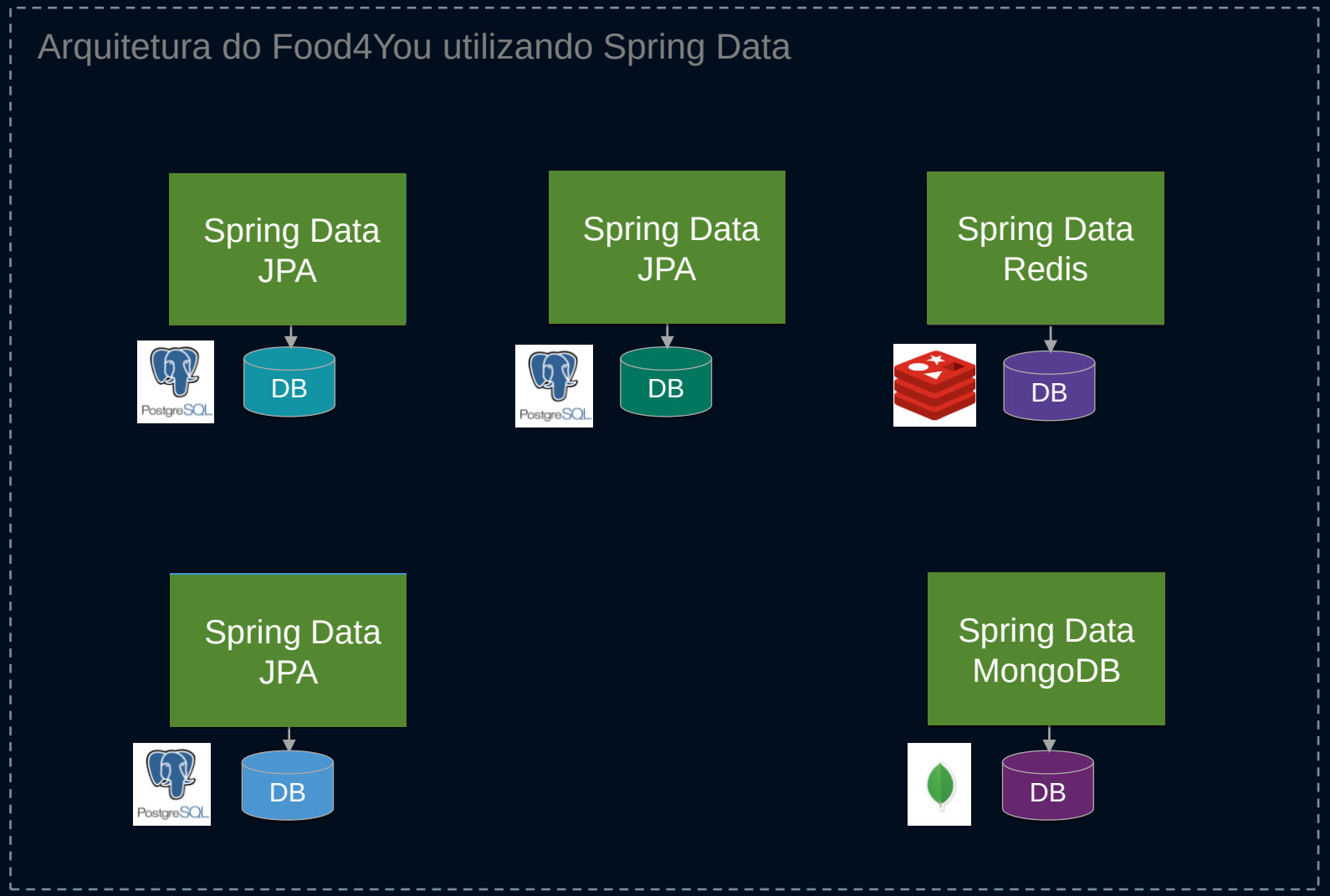
#decoderweek

Já temos várias vantagens de uma Arquitetura de Microservices mesmo com a base de dados compartilhada.

Base de Dados por Microservices



Base de Dados por Microservices com Spring Data



COMPLEMENTANDO

Spring Data 

- Spring Data JPA
- Spring Data MongoDB
- Spring Data JDBC
- Spring Data Redis
- Spring Data Cassandra
- Spring Data Elasticsearch

```xml
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
```

```xml
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-mongodb</artifactId>
</dependency>
```

```xml
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-mongodb-reactive</artifactId>
</dependency>
```

```xml
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jdbc</artifactId>
</dependency>
```

DESCOMPLICANDO

#decoderweek

Com base de dados por Microservices temos os dados distribuídos pela arquitetura.

Com os dados distribuídos entre os Microservices temos que lidar com a sincronia e replicação de dados, com o desafio de manter a consistência com a alta disponibilidade.

Dia 2

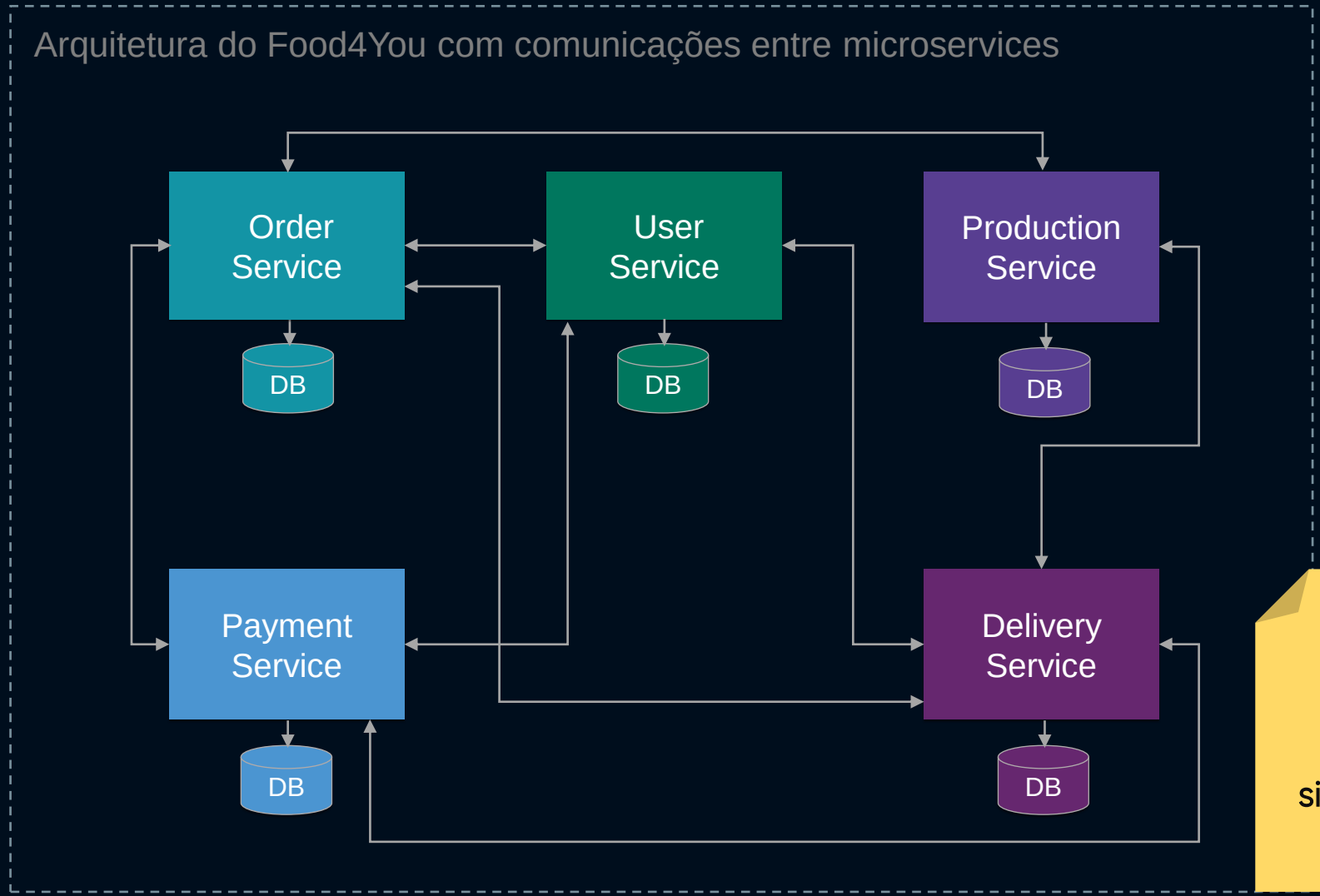
Gestão de dados distribuídos, teorema CAP...

DESCOMPLICANDO

#decoderweek

Precisamos utilizar de Identificadores Universais (UUIDs) para Arquiteturas Distribuídas.

Base de Dados por Microservices → Sincronia e Replicação dos Dados Distribuídos → Comunicação entre Microservices



Dia 2

Modelos de comunicações e sincronia e replicação de dados.

DESCOMPLICANDO

Não existe desacoplamento absoluto entre Microservices...

A ideia é buscar o maior desacoplamento possível.

O alto acoplamento nem sempre é um problema de modelagem arquitetural, mas sim pode ser inerente ao próprio negócio, pode ser uma regra do negócio.

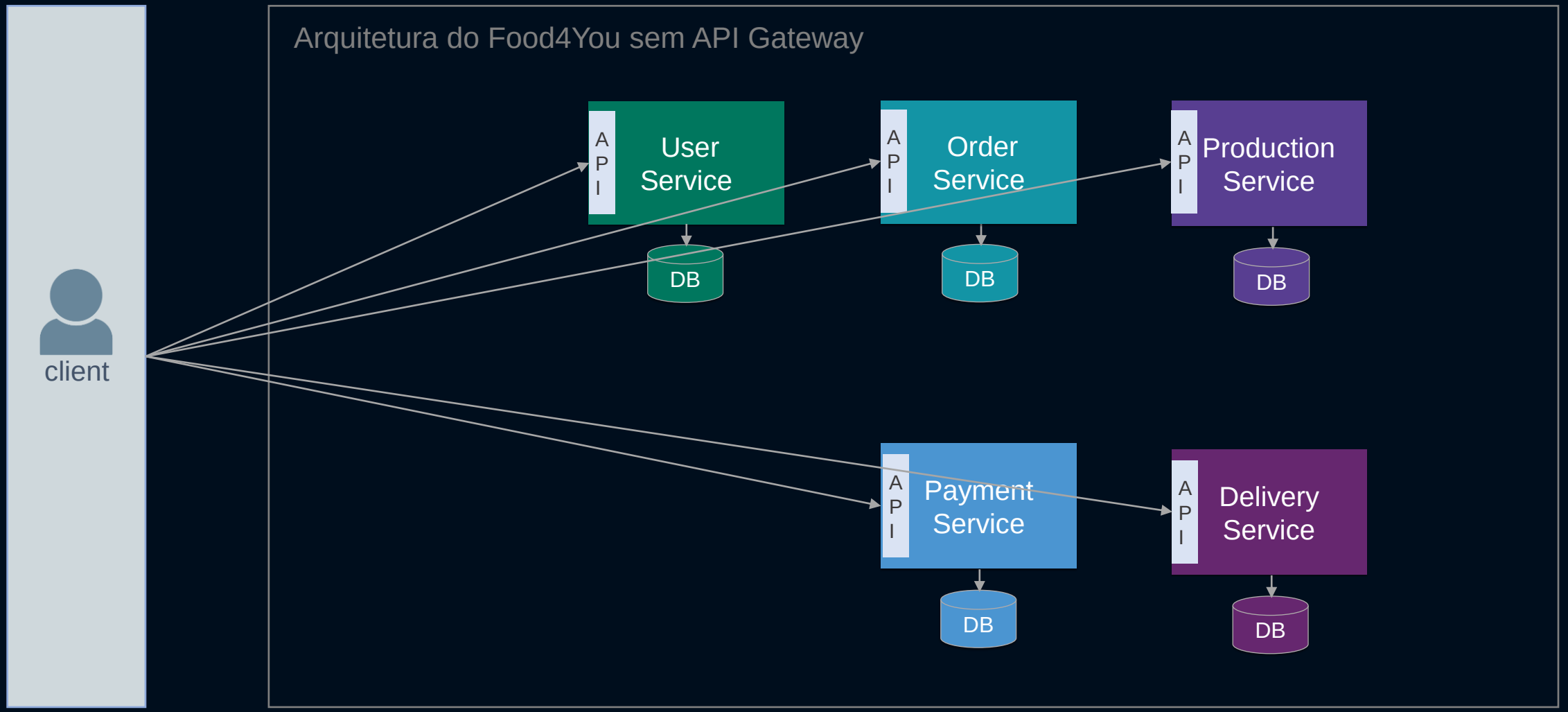
Microservices de Configurações

API
Gateway

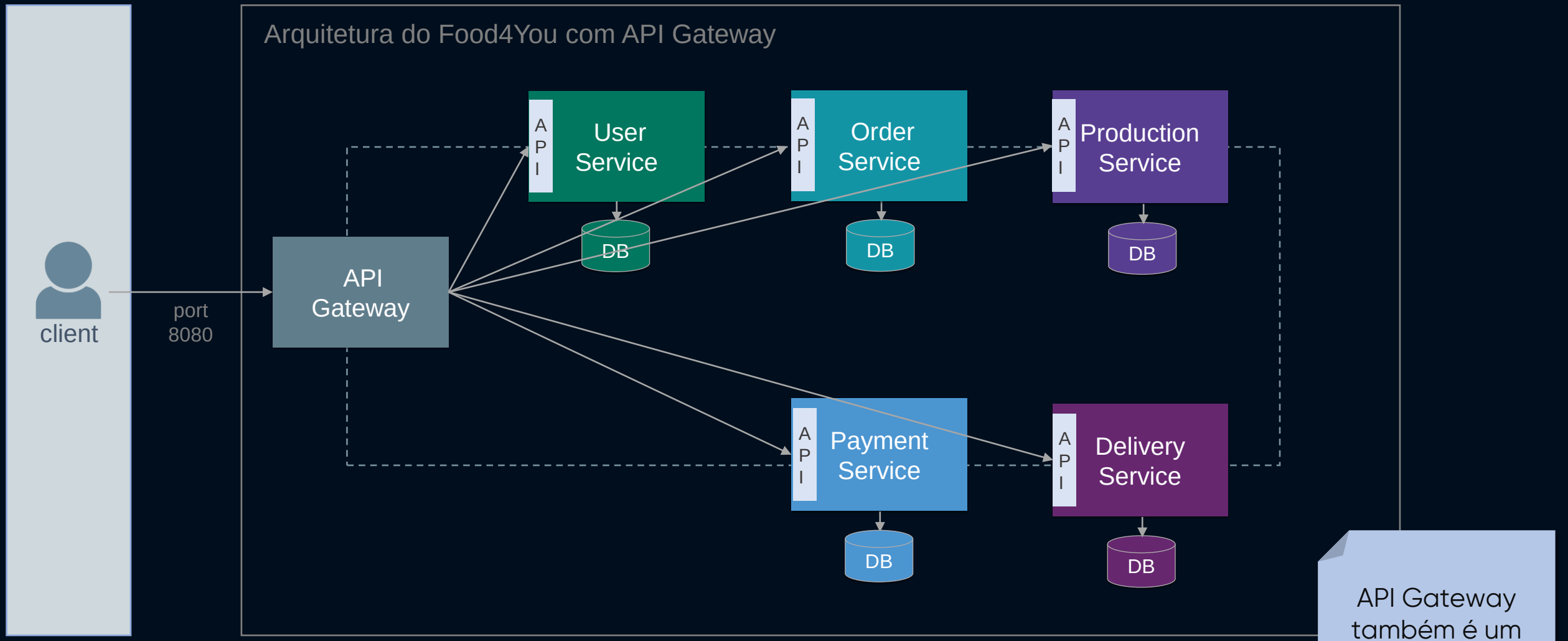
Service
Registry

Config
Server

Modelando API Gateway

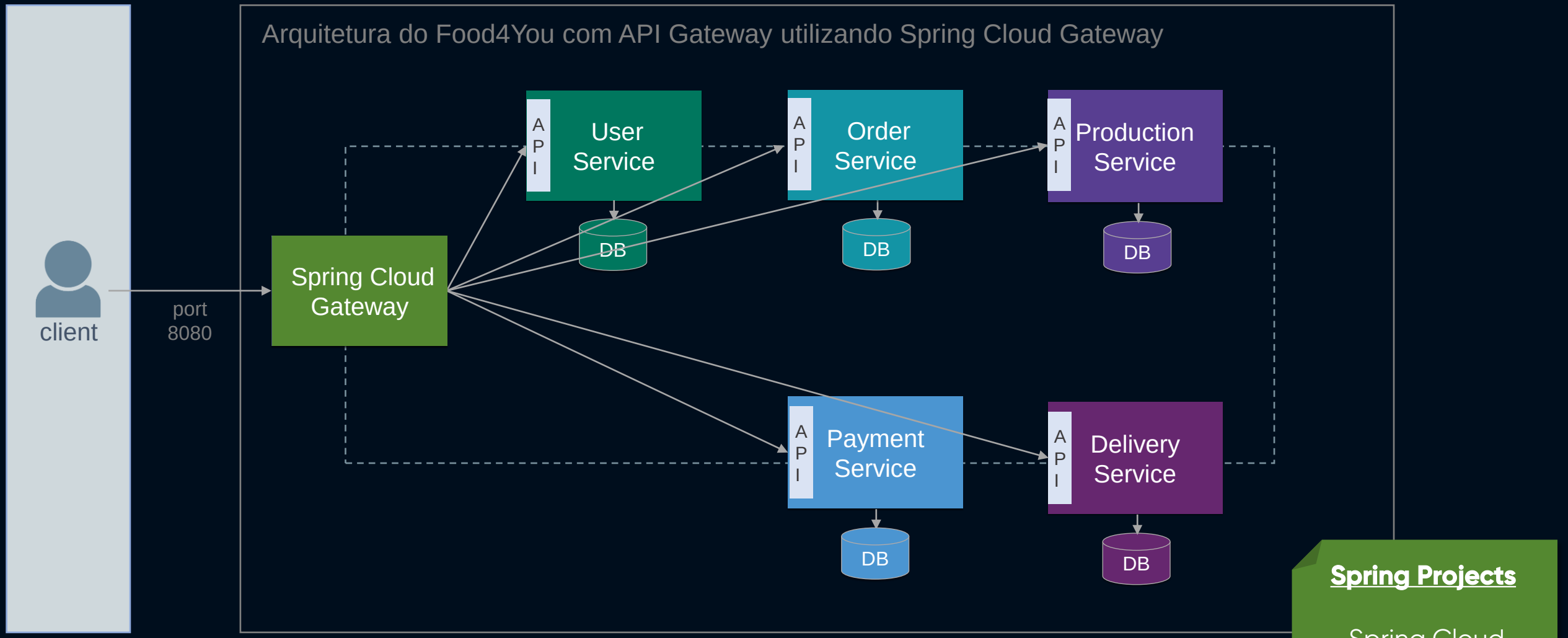


Modelando API Gateway



API Gateway também é um Microservice.

API Gateway com Spring Cloud Gateway



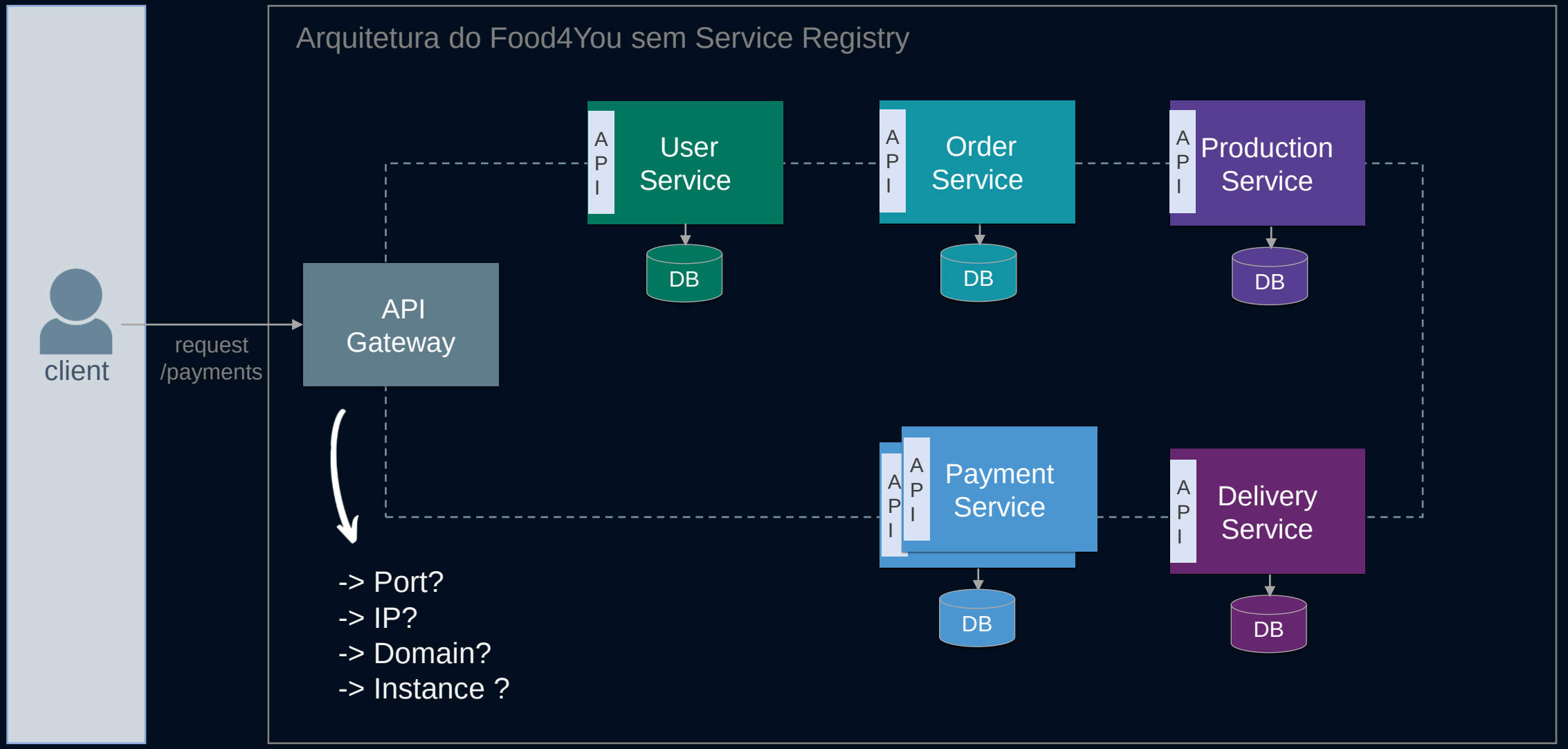
COMPLEMENTANDO

Spring Cloud Gateway

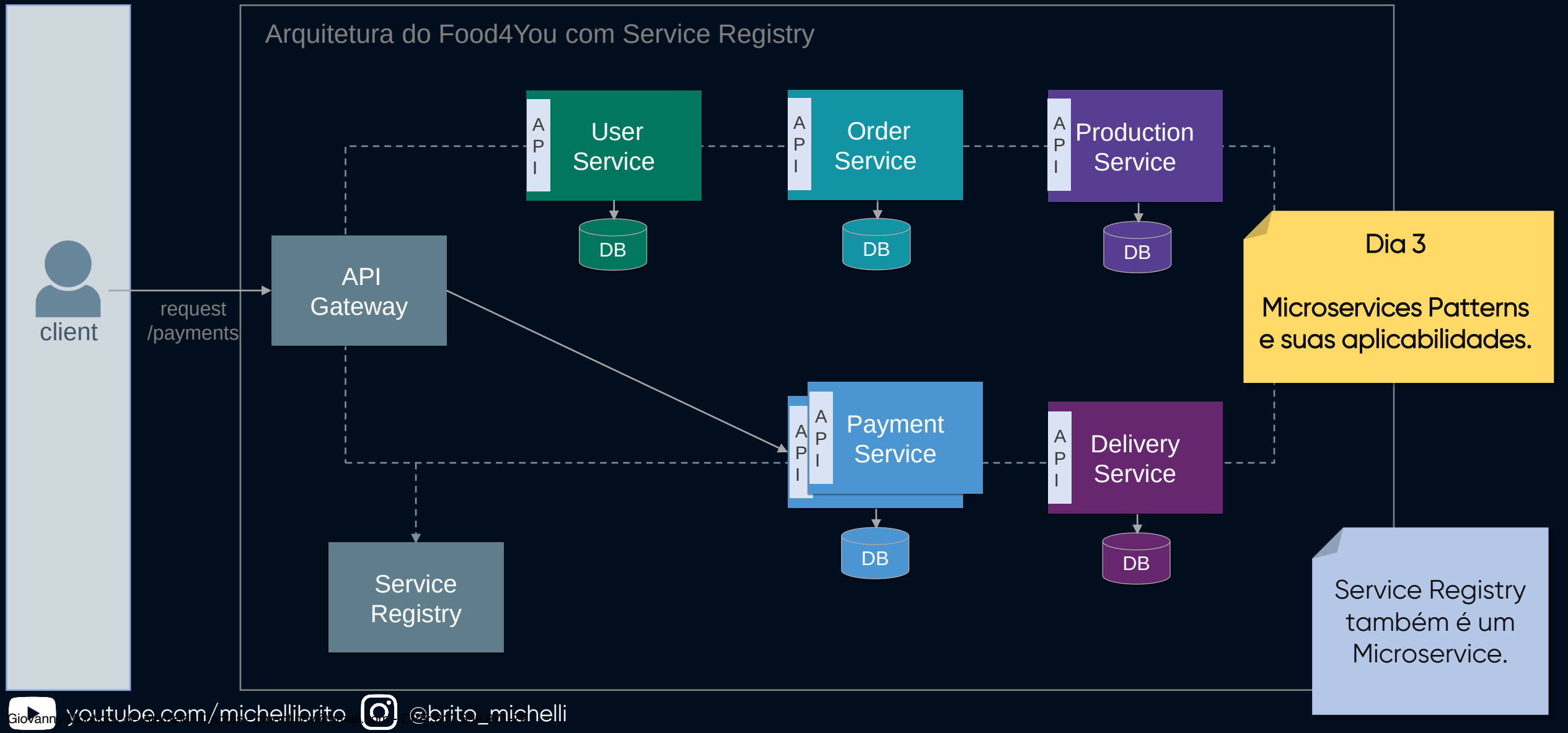
- Inserido na versão 5 do Spring Framework
- Baseado no projeto Reactor
- Suporte para programação Reativa
- Servidor Netty
- Roteamento para APIs, definição de filtros e predicados
- Integração com Service Registry para registro e descoberta de serviços

```
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-gateway</artifactId>
</dependency>
```

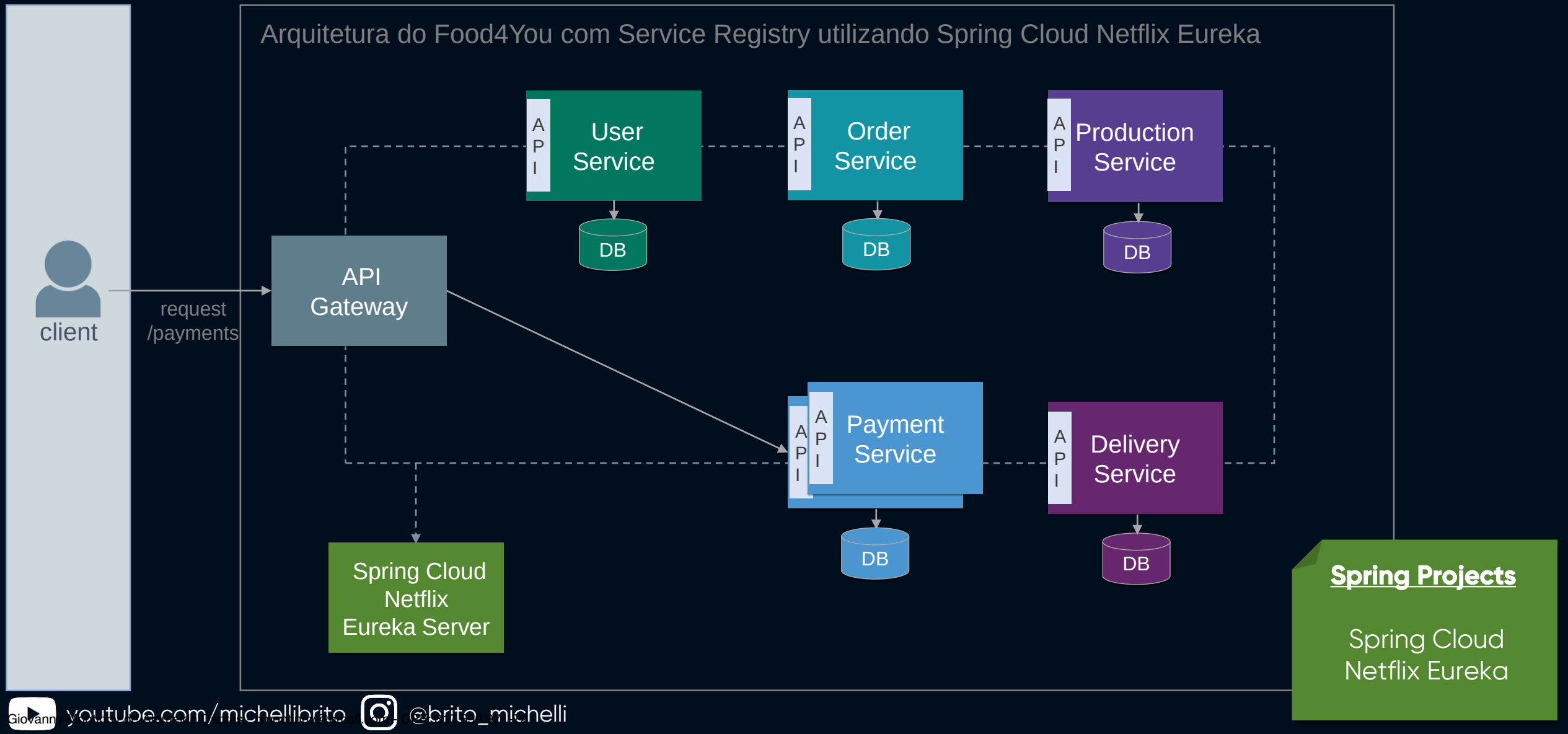
Modelando Service Registry / Discovery



Modelando Service Registry / Discovery



Registry Discovery com Spring Cloud Netflix Eureka

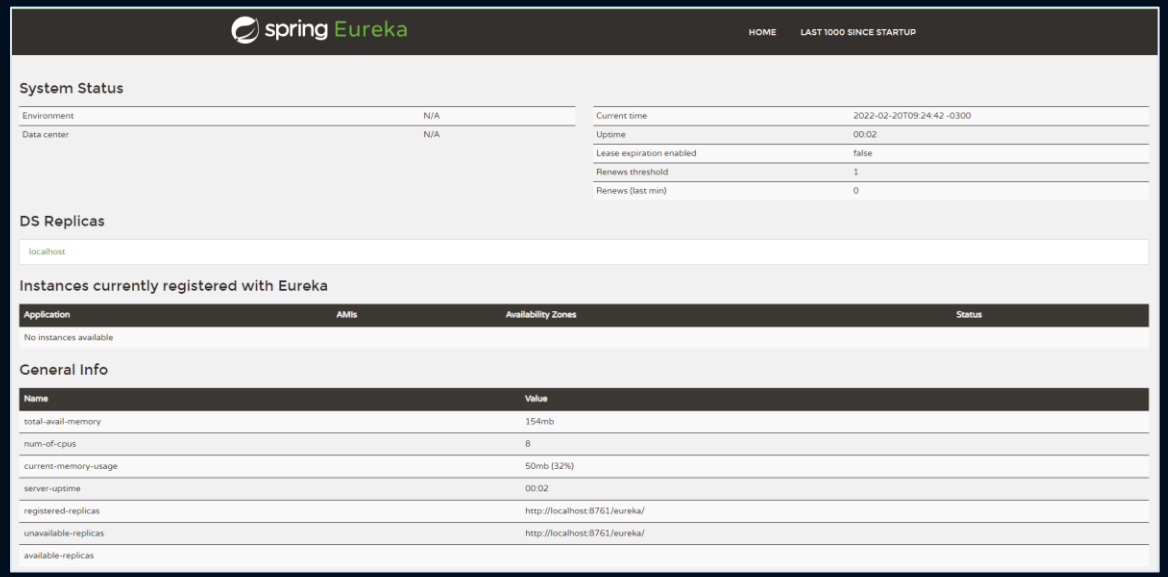


COMPLEMENTANDO

Spring Cloud Netflix Eureka

Instancias podem ser registradas em um servidor Eureka: Eureka Server

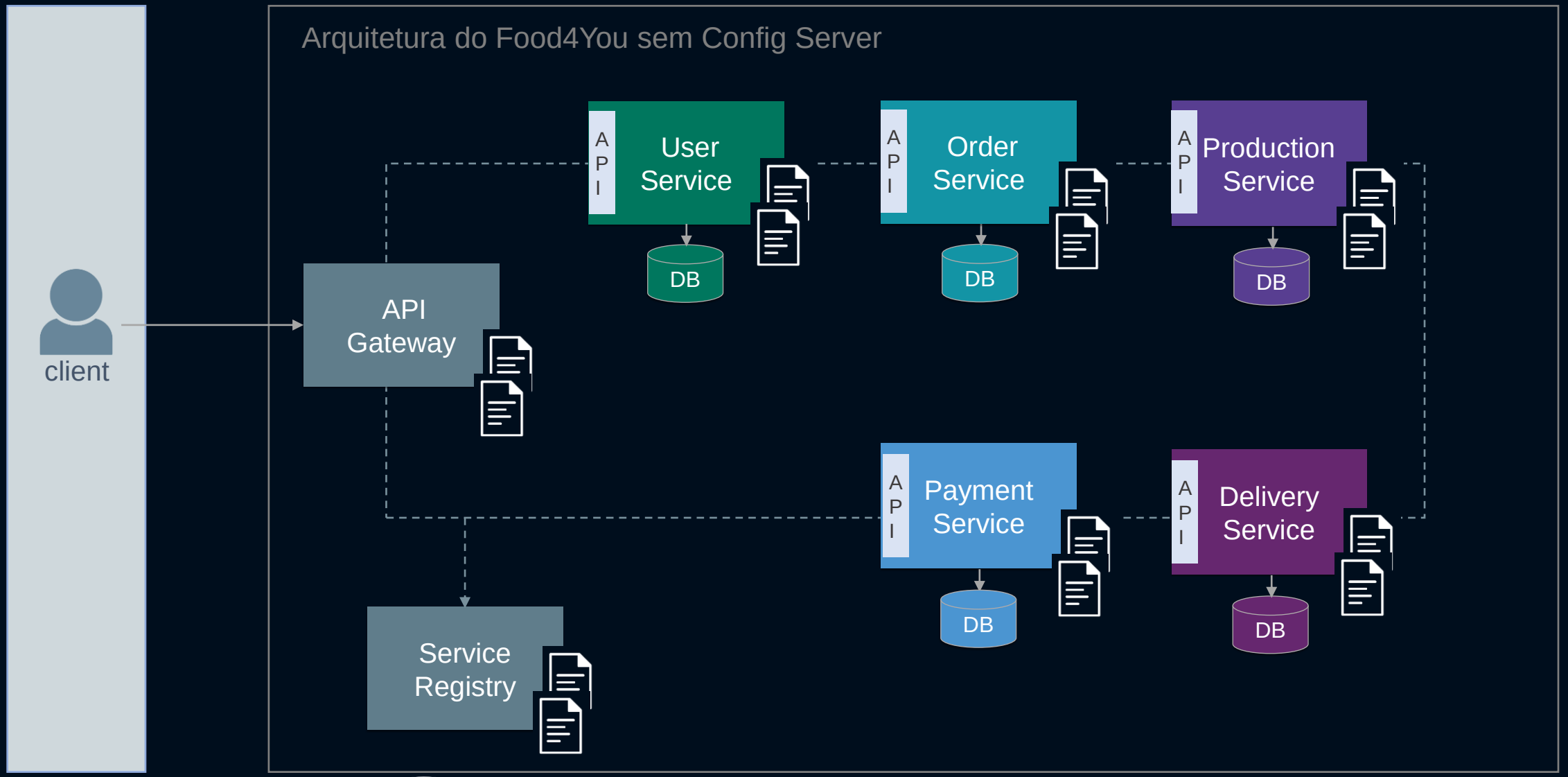
Os Eureka Clients podem descobrir essas instancias quando necessário



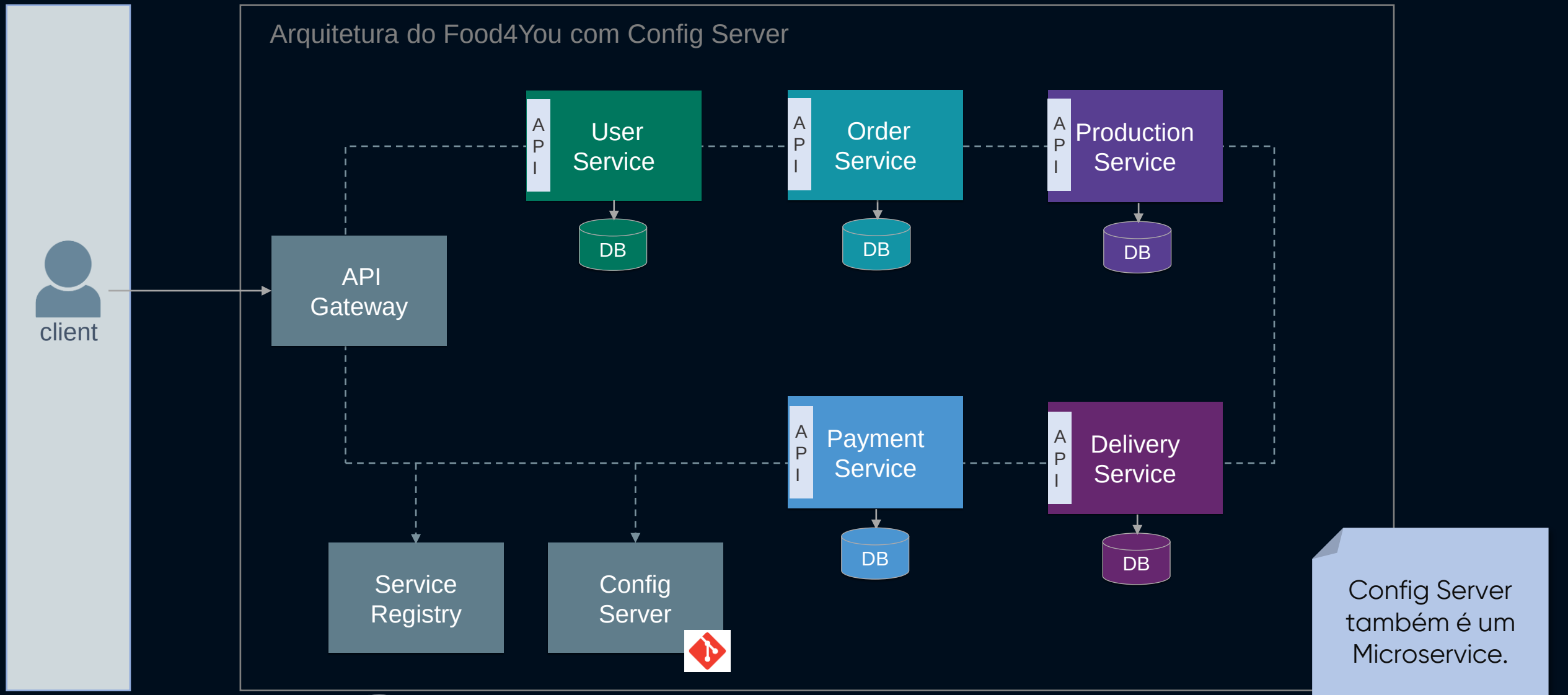

```xml
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>
</dependency>
```

```xml
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
</dependency>
```

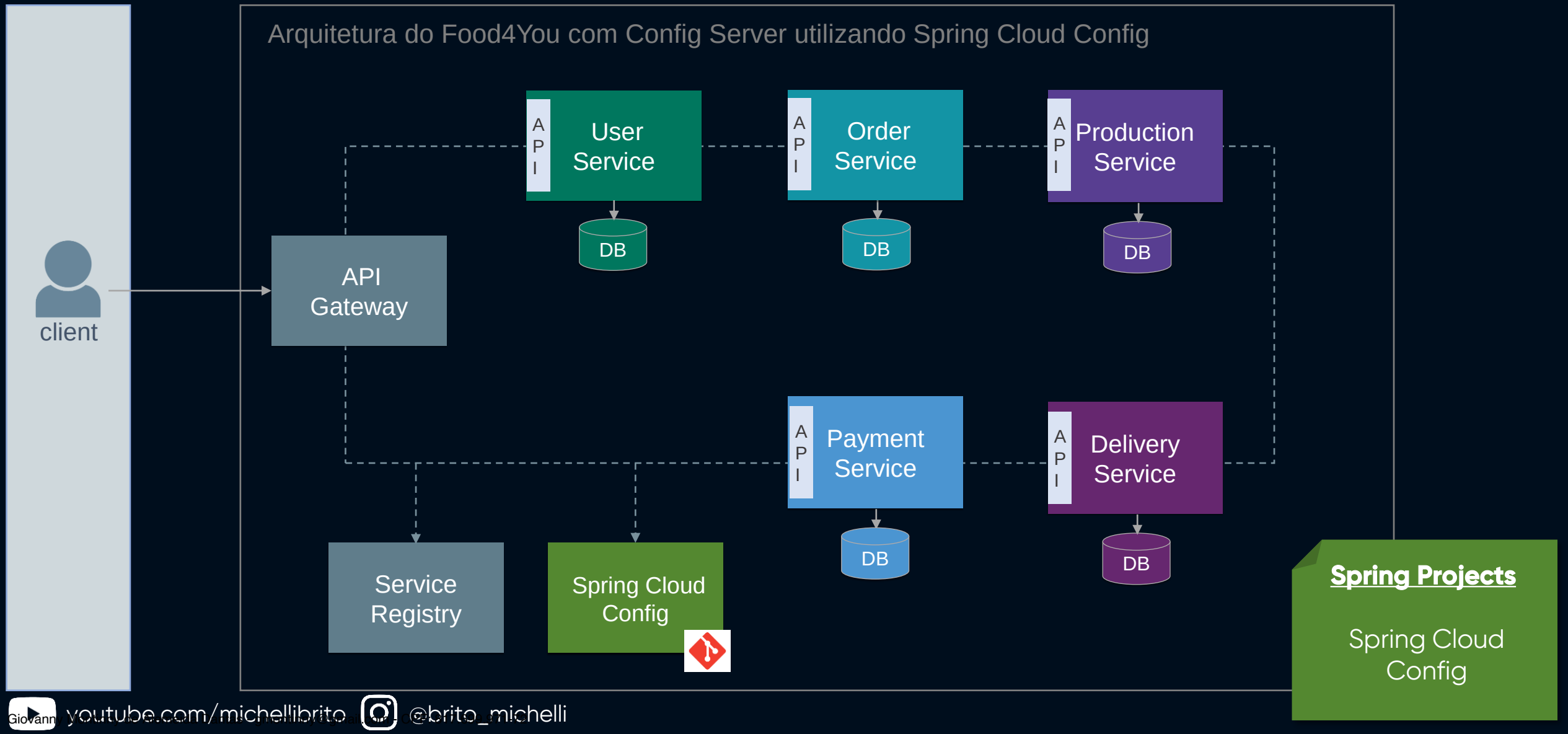
Modelando Config Server



Modelando Config Server



Config Server com Spring Cloud Config



COMPLEMENTANDO

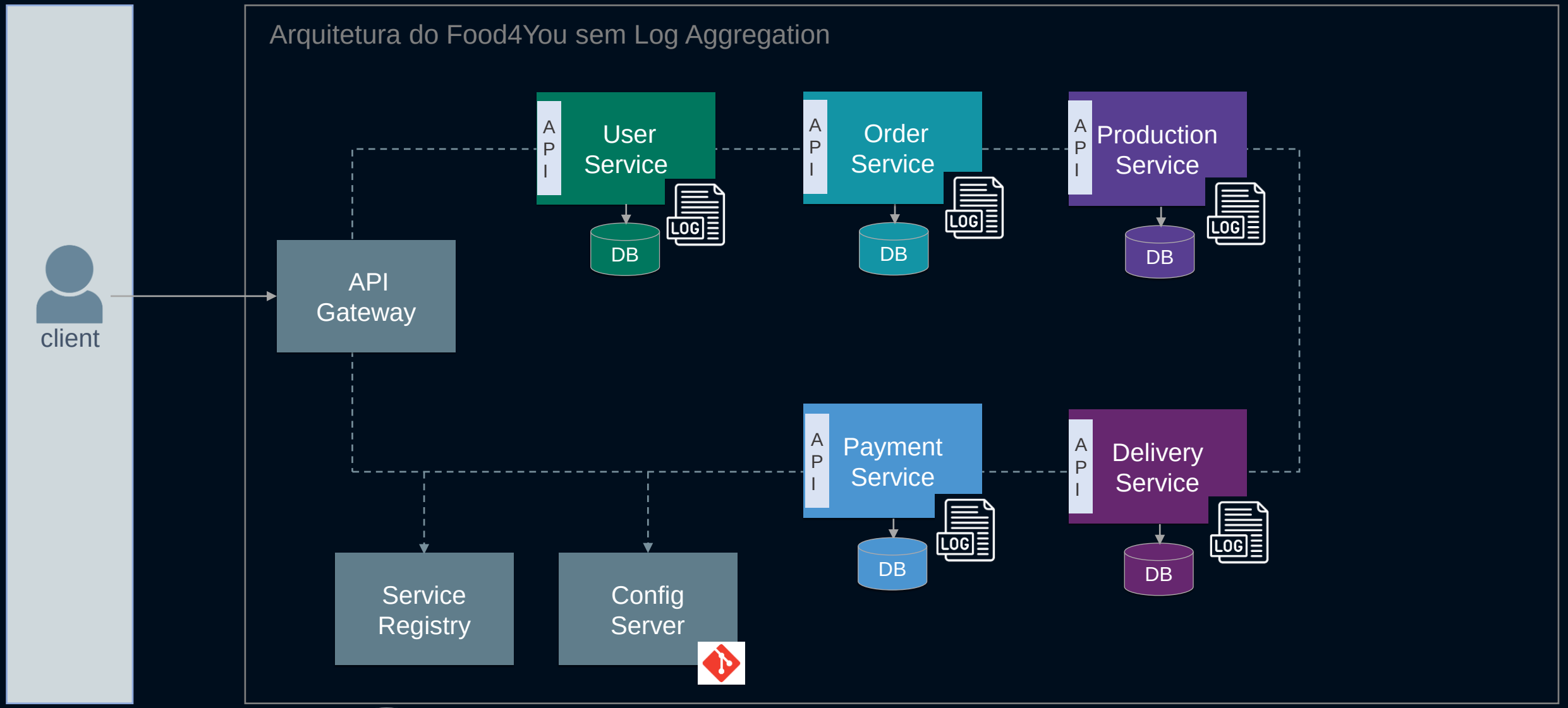
Spring Cloud Config

- Suporte para configuração externa em uma arquitetura de Microservices
- Config Server para gerenciar configurações de múltiplos ambientes
- Utiliza do Git para armazenamento e versionamento das propriedades
- APIs para consultas de propriedades
- Criptografia e Descriptografia de valores de propriedades

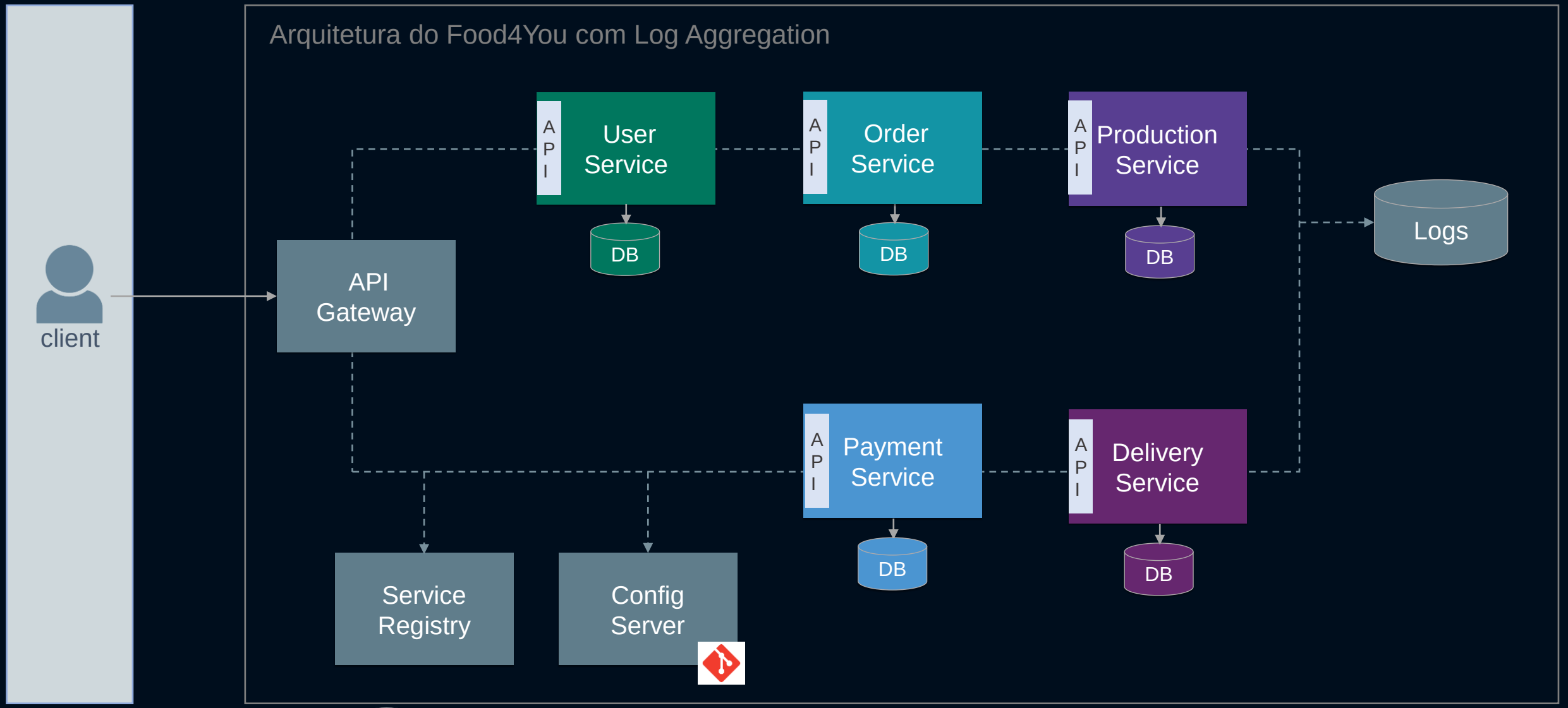
```xml
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-config-server</artifactId>
</dependency>
```

```xml
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-config</artifactId>
</dependency>
```

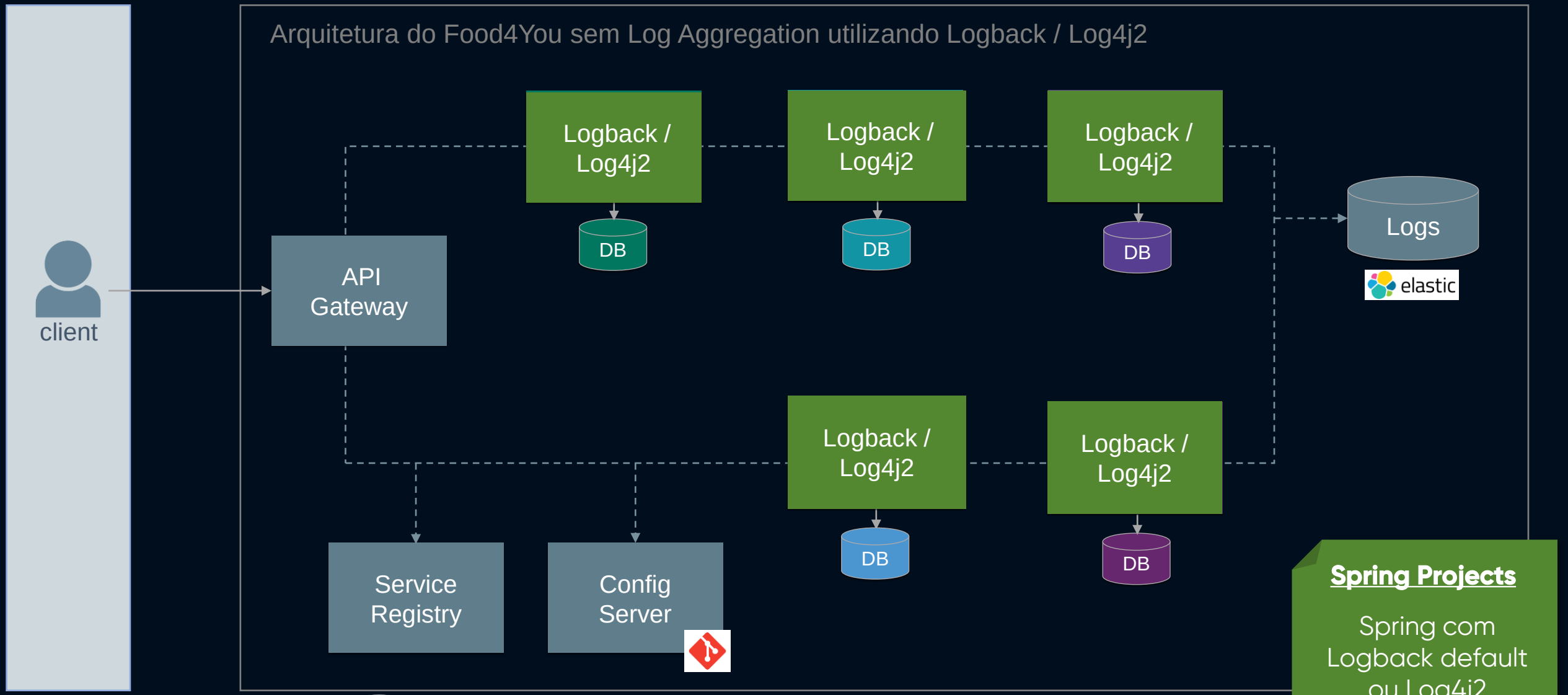
Modelando Observabilidade com Log Aggregation



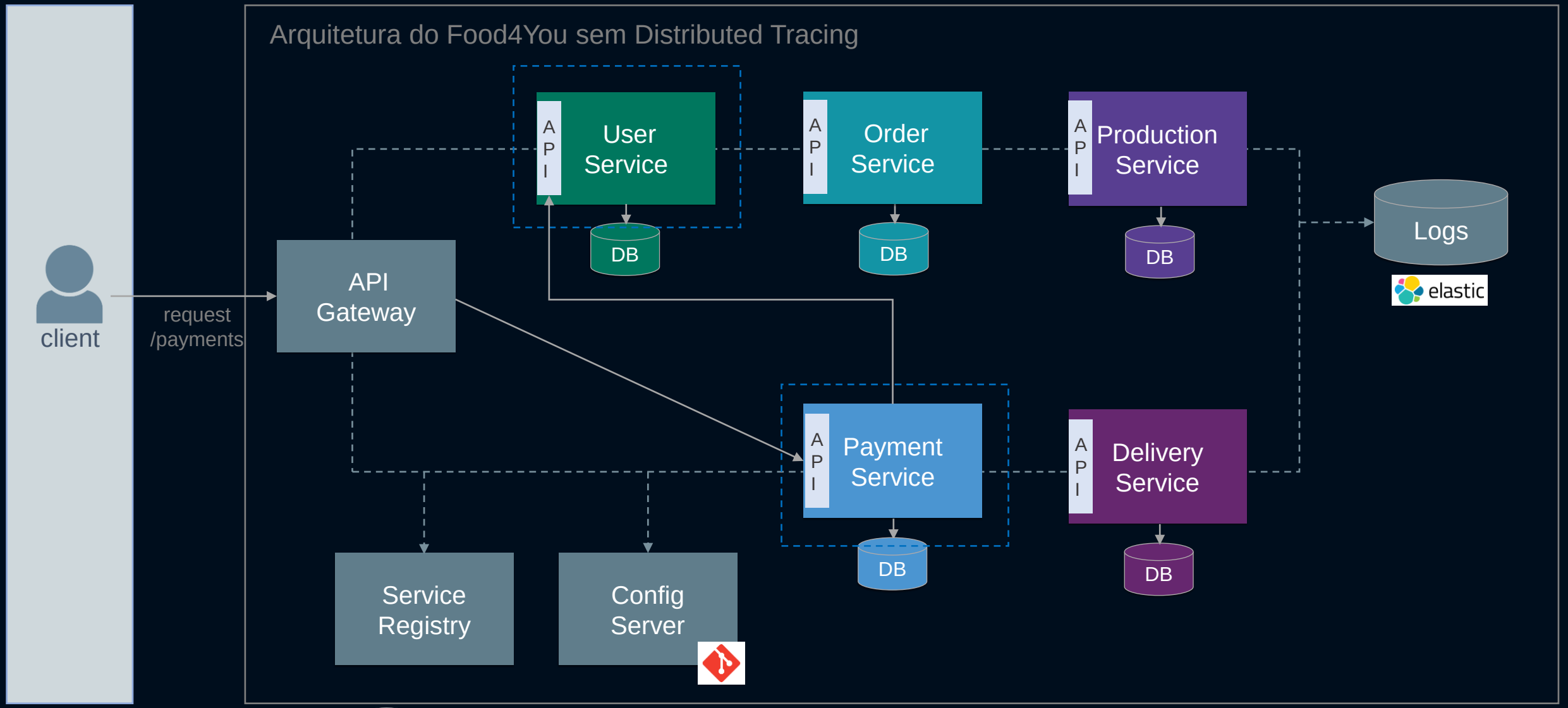
Modelando Observabilidade com Log Aggregation



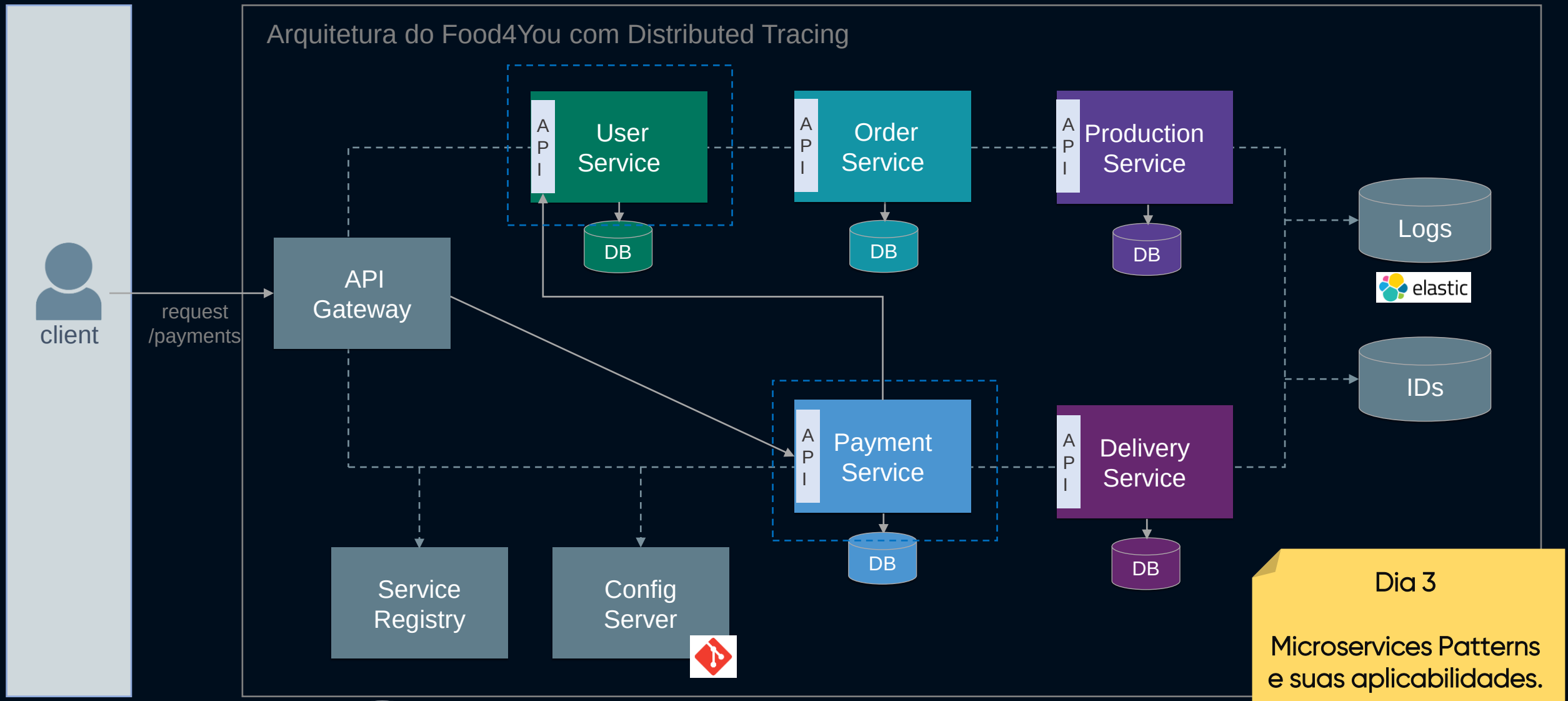
Log Aggregation com ELK e Logback/Log4j2



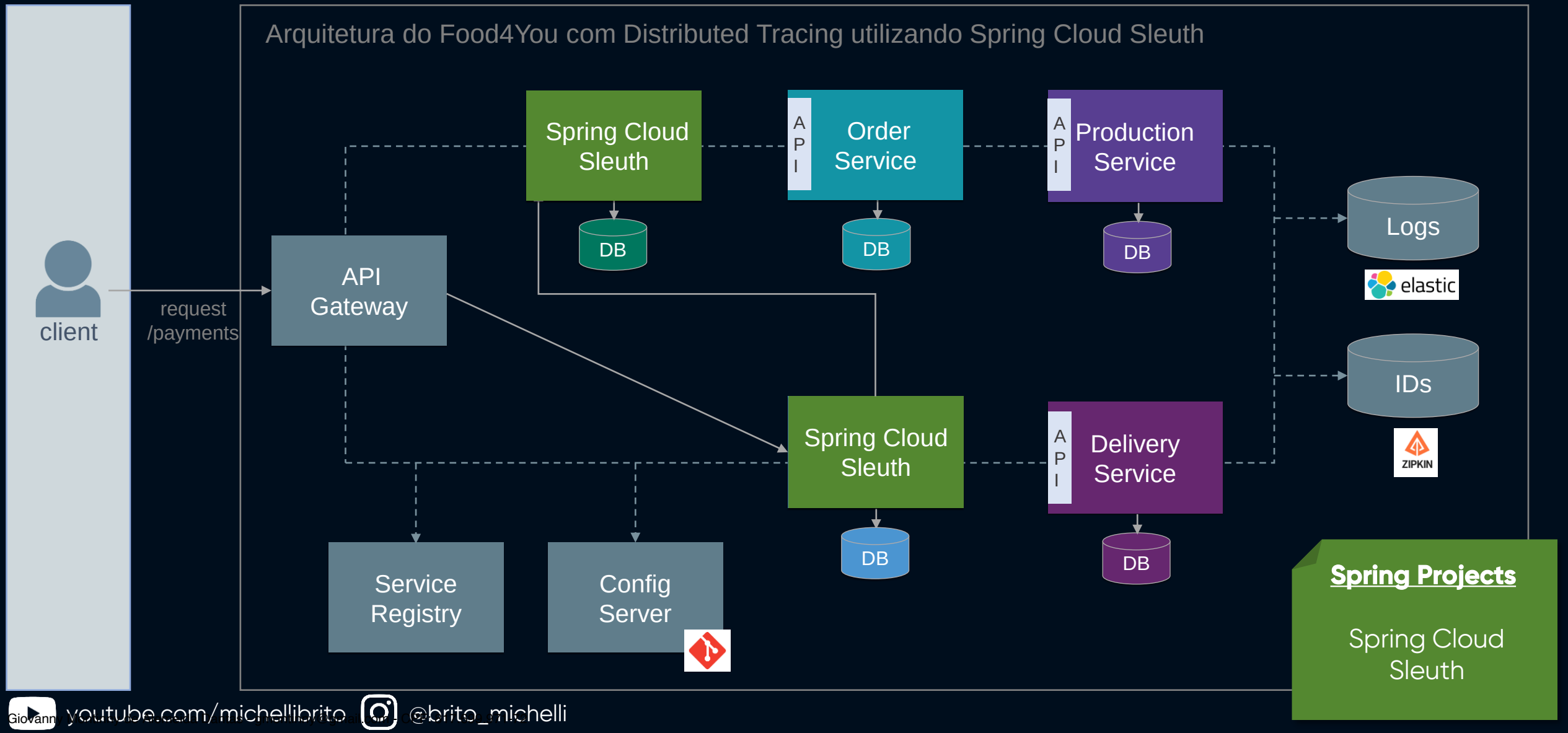
Modelando Observabilidade com Distributed Tracing



Modelando Observabilidade com Distributed Tracing



Distributed Tracing com Spring Cloud Sleuth



COMPLEMENTANDO

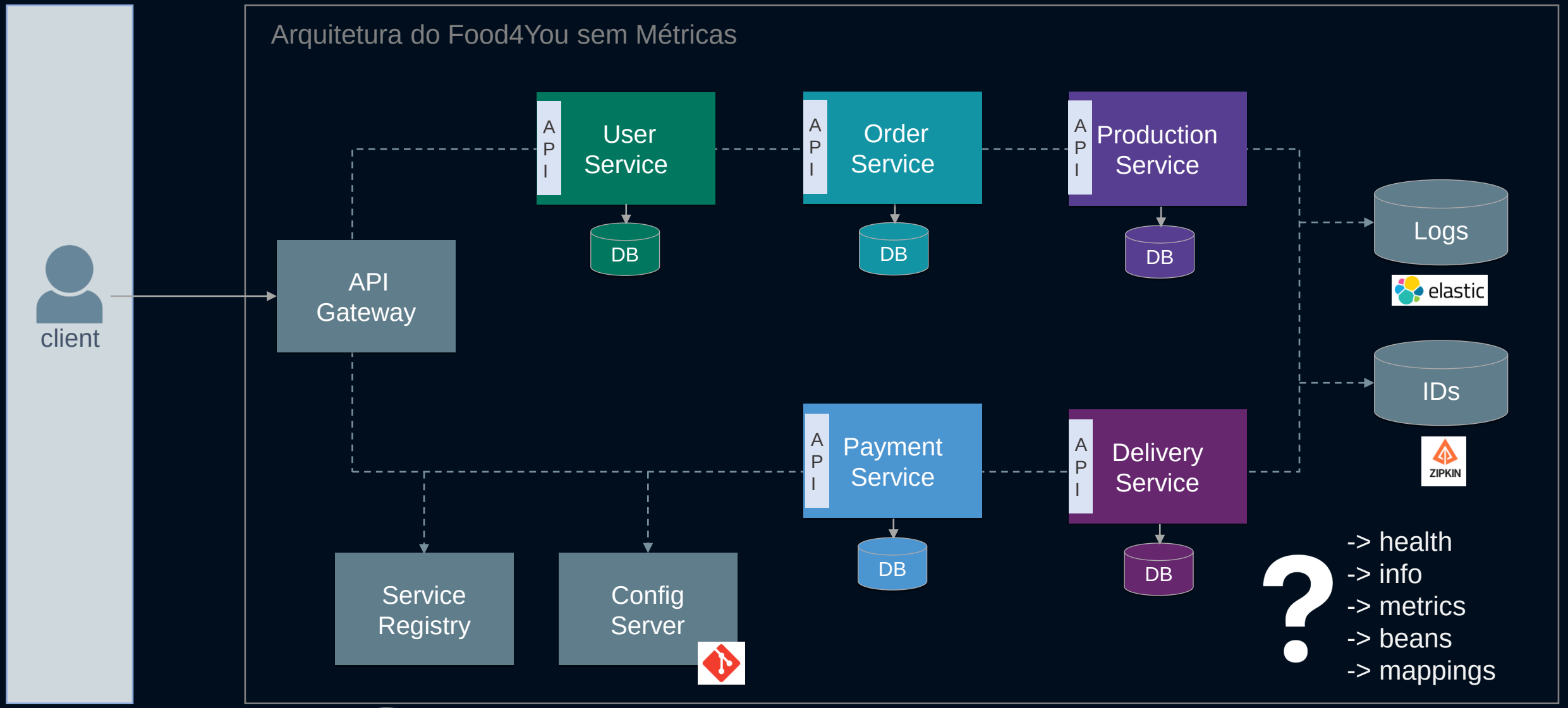
Spring Cloud Sleuth

- Suporte para rastreamento distribuído
- Geração de identificadores Trace ID (único entre todas as chamadas)
- Geração de identificadores Span ID (individuais para cada chamada)
- Integração com Zipkin para analise e interpretação dos ids de forma centralizada

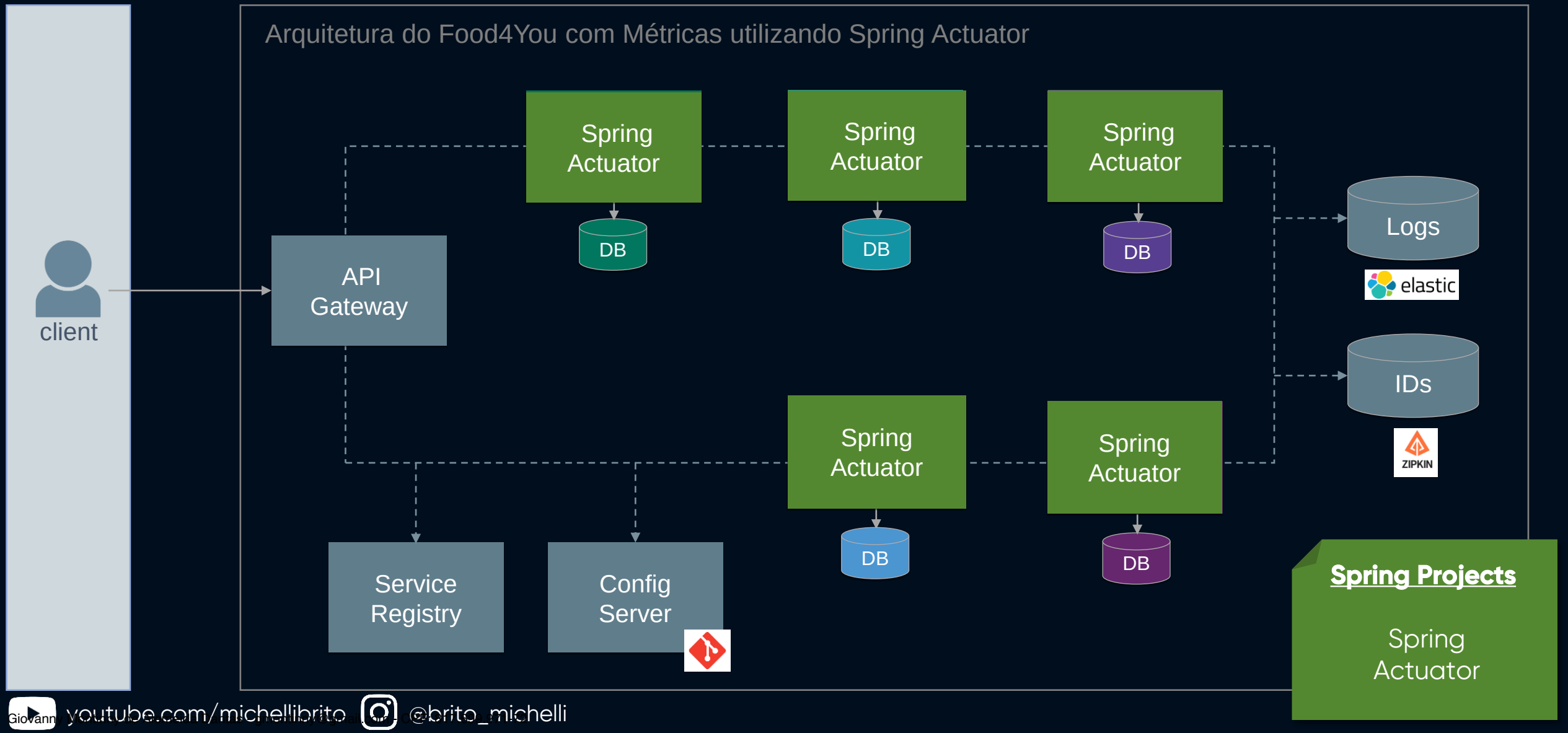
```xml
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-sleuth</artifactId>
</dependency>
```

```xml
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-sleuth-zipkin</artifactId>
</dependency>
```

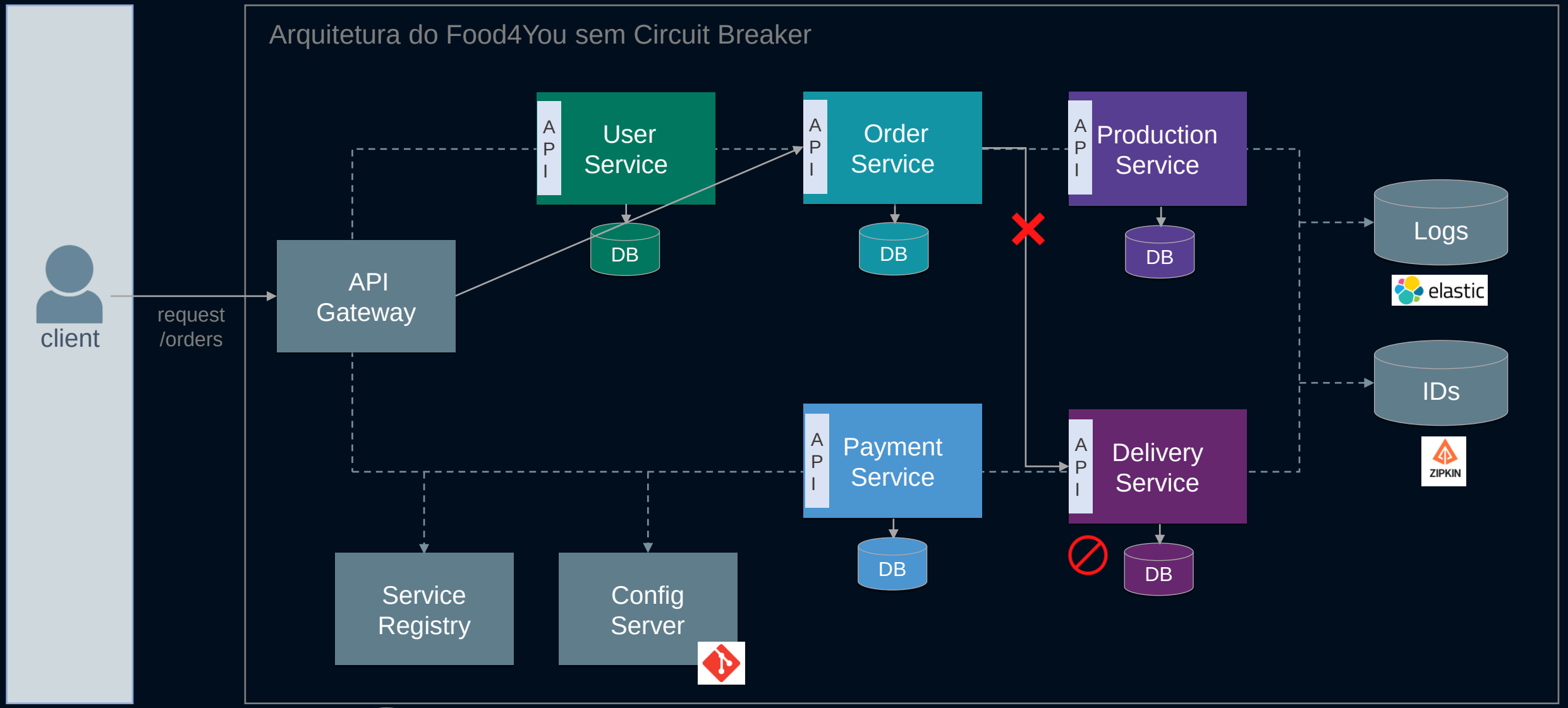
Modelando Observabilidade com Métricas



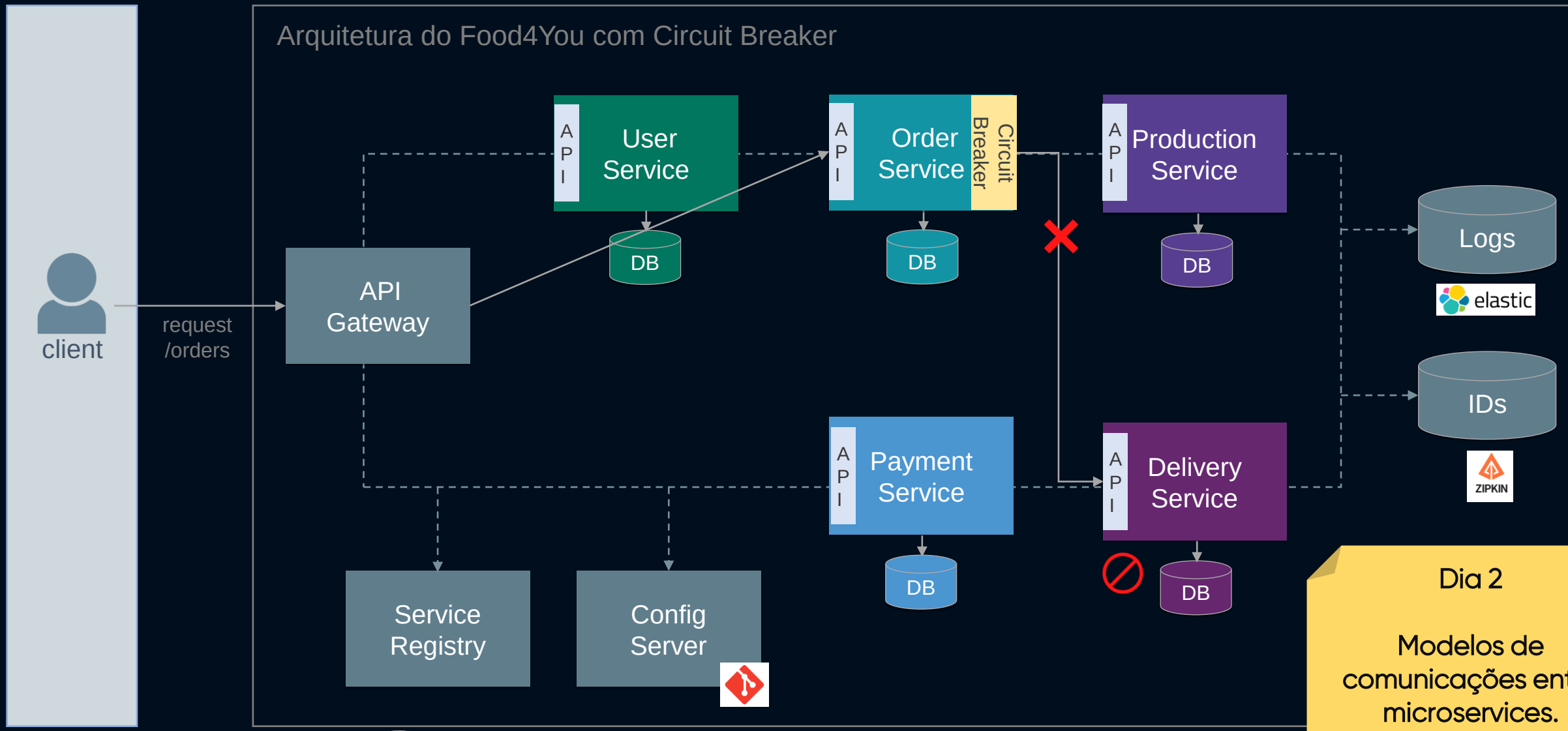
Modelando Métricas com Spring Actuator



Modelando Resiliência com Circuit Breaker



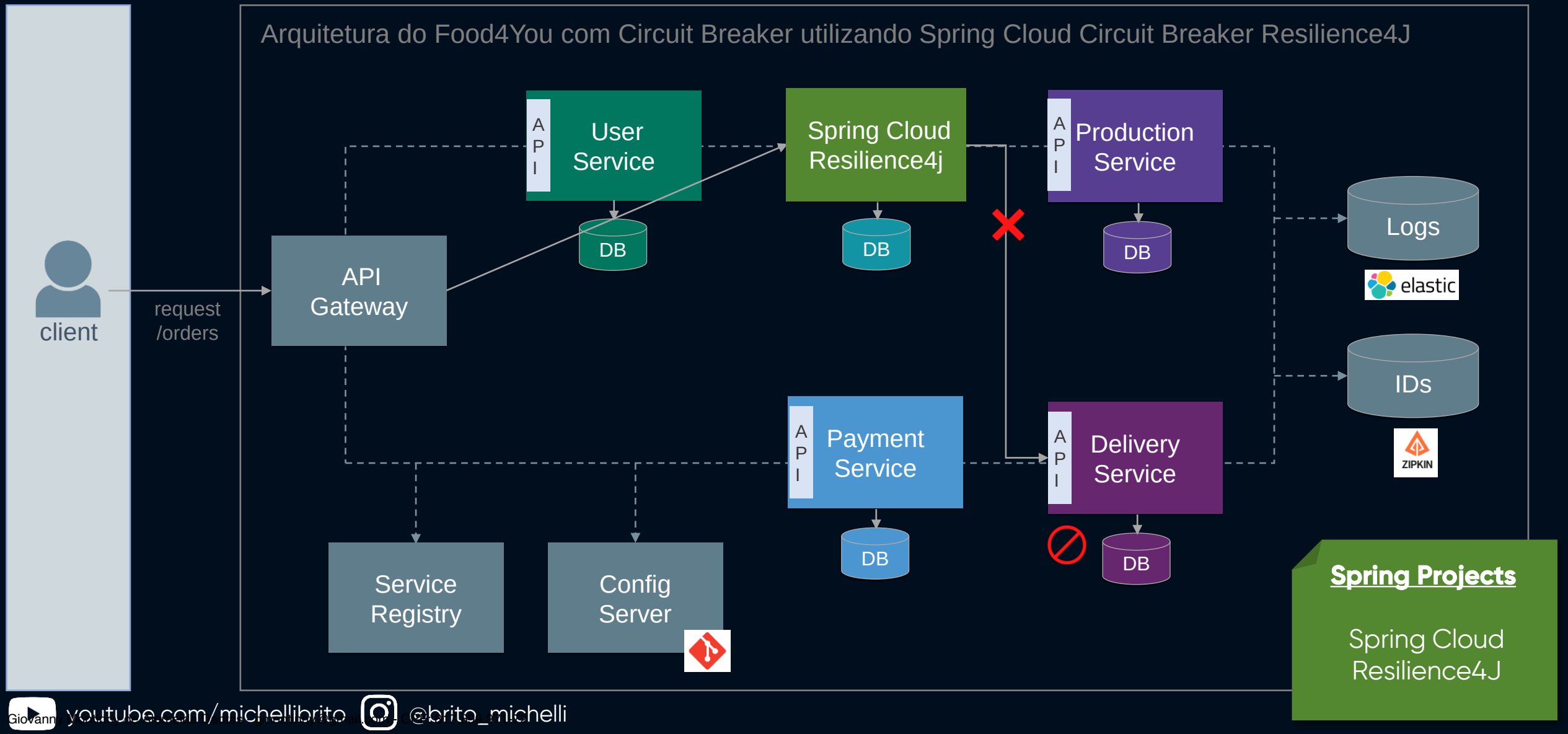
Modelando Resiliência com Circuit Breaker



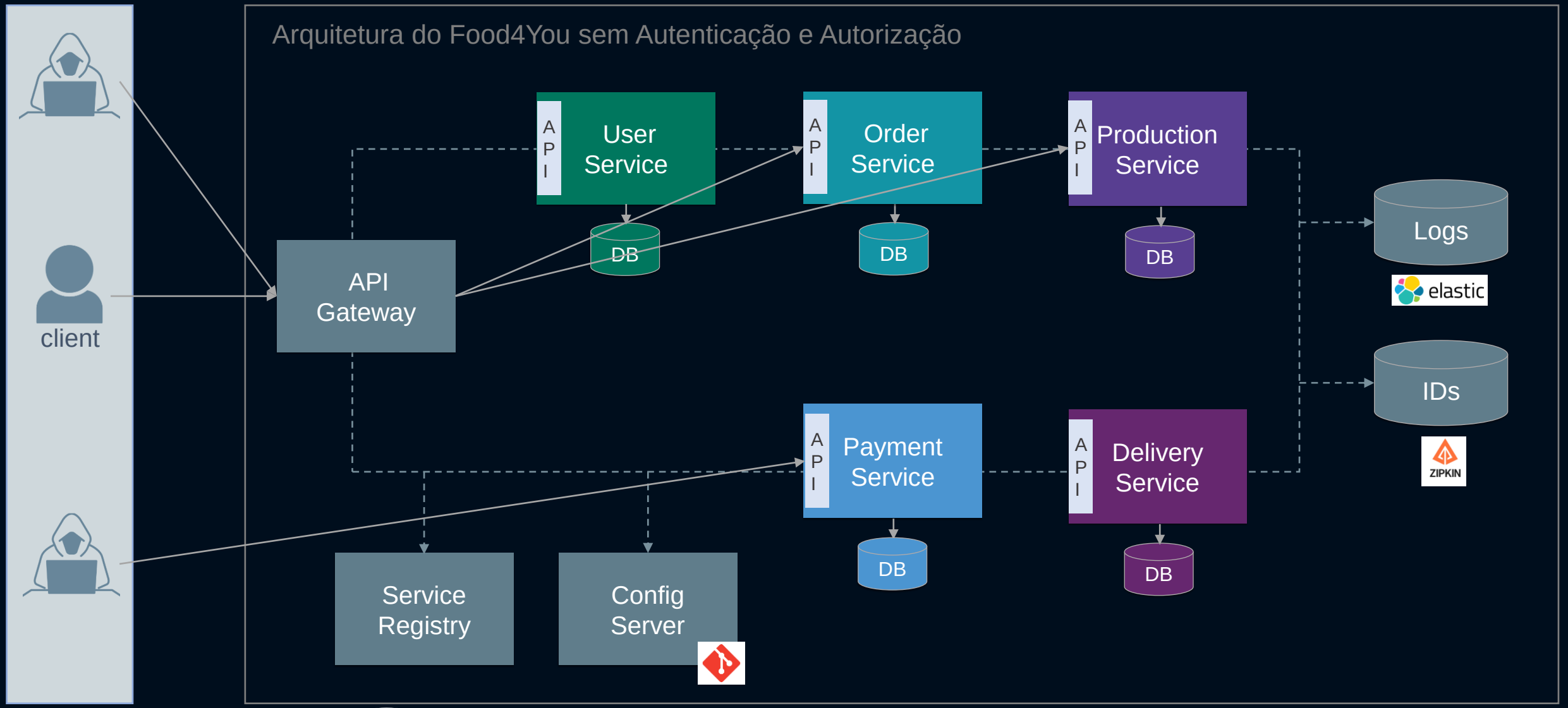
Dia 2

Modelos de comunicações entre microservices.

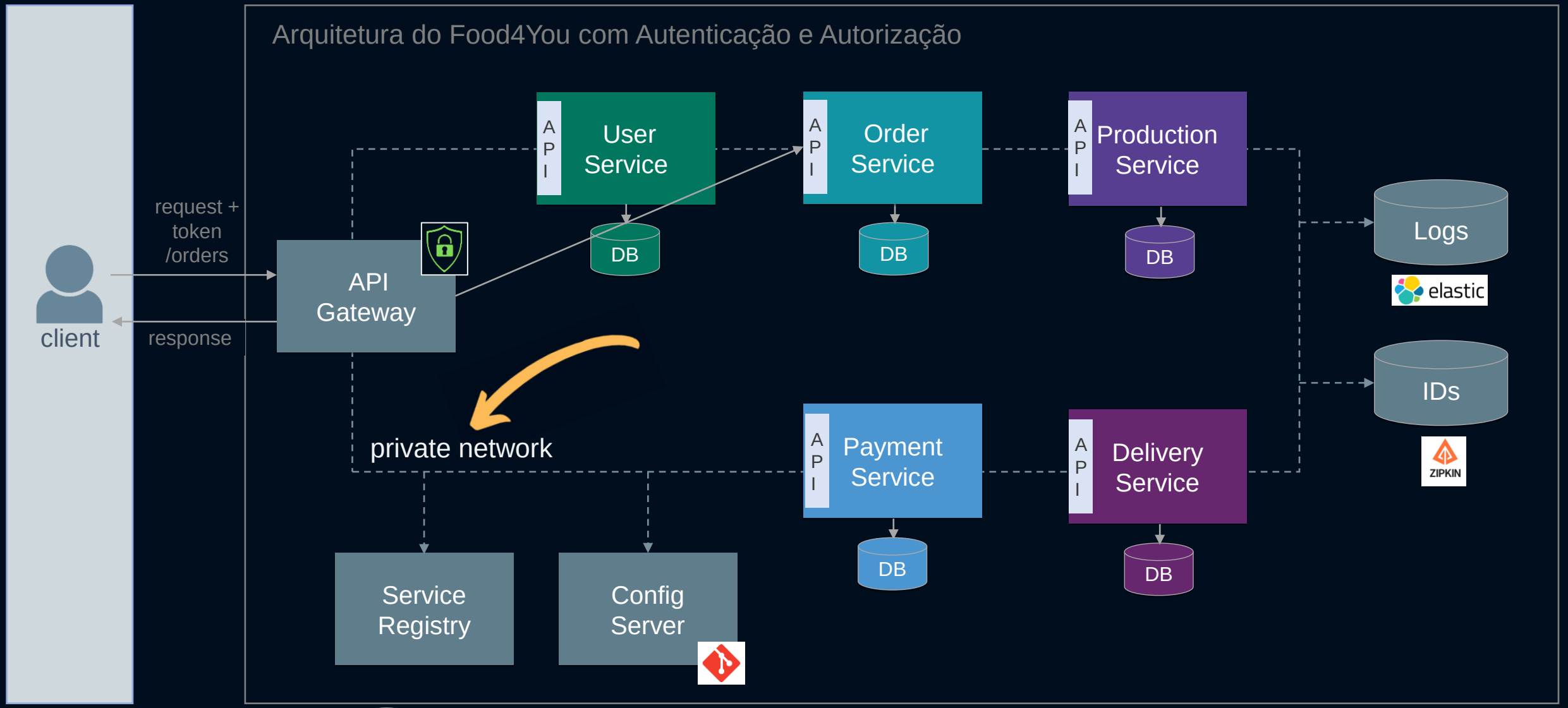
Circuit Breaker com Spring Cloud Resilience4J



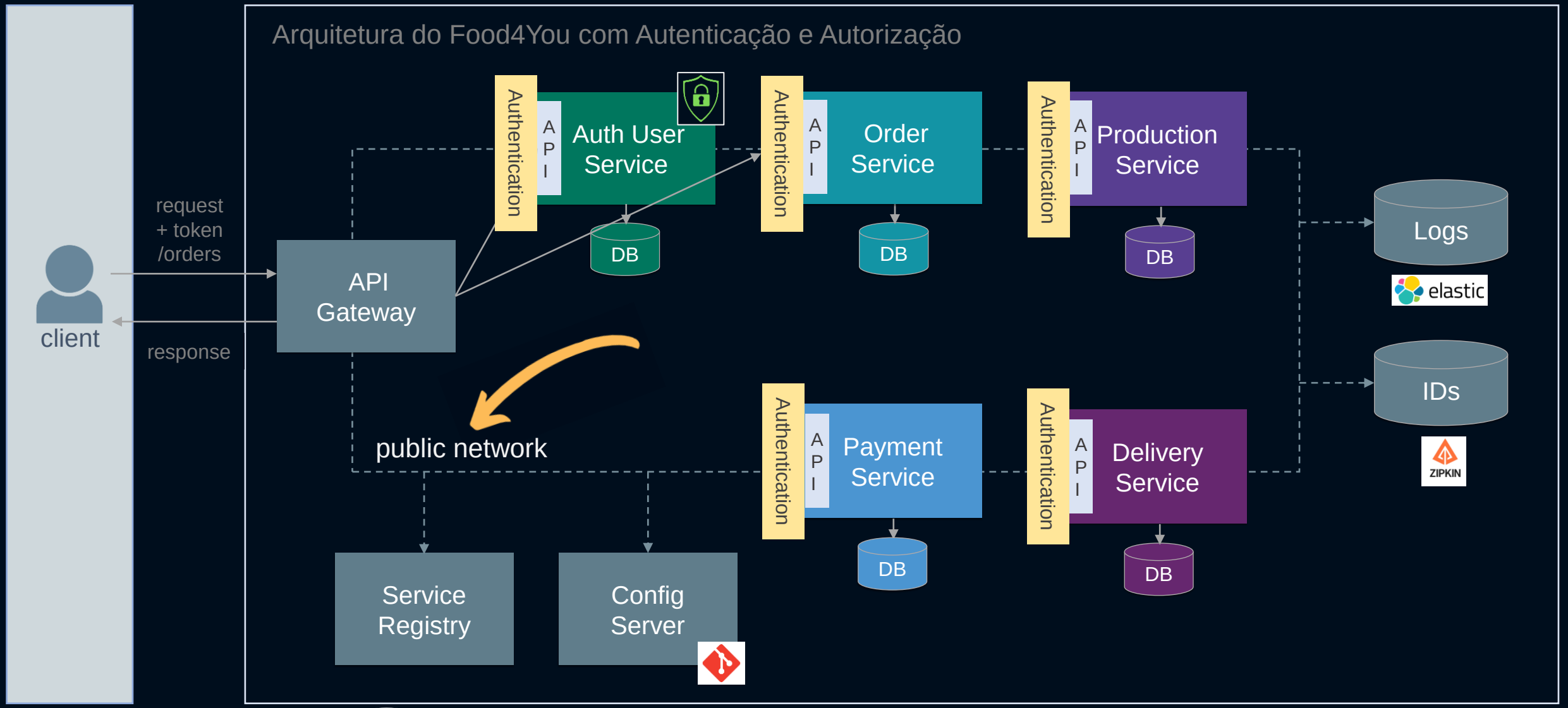
Modelando Autenticação e Autorização



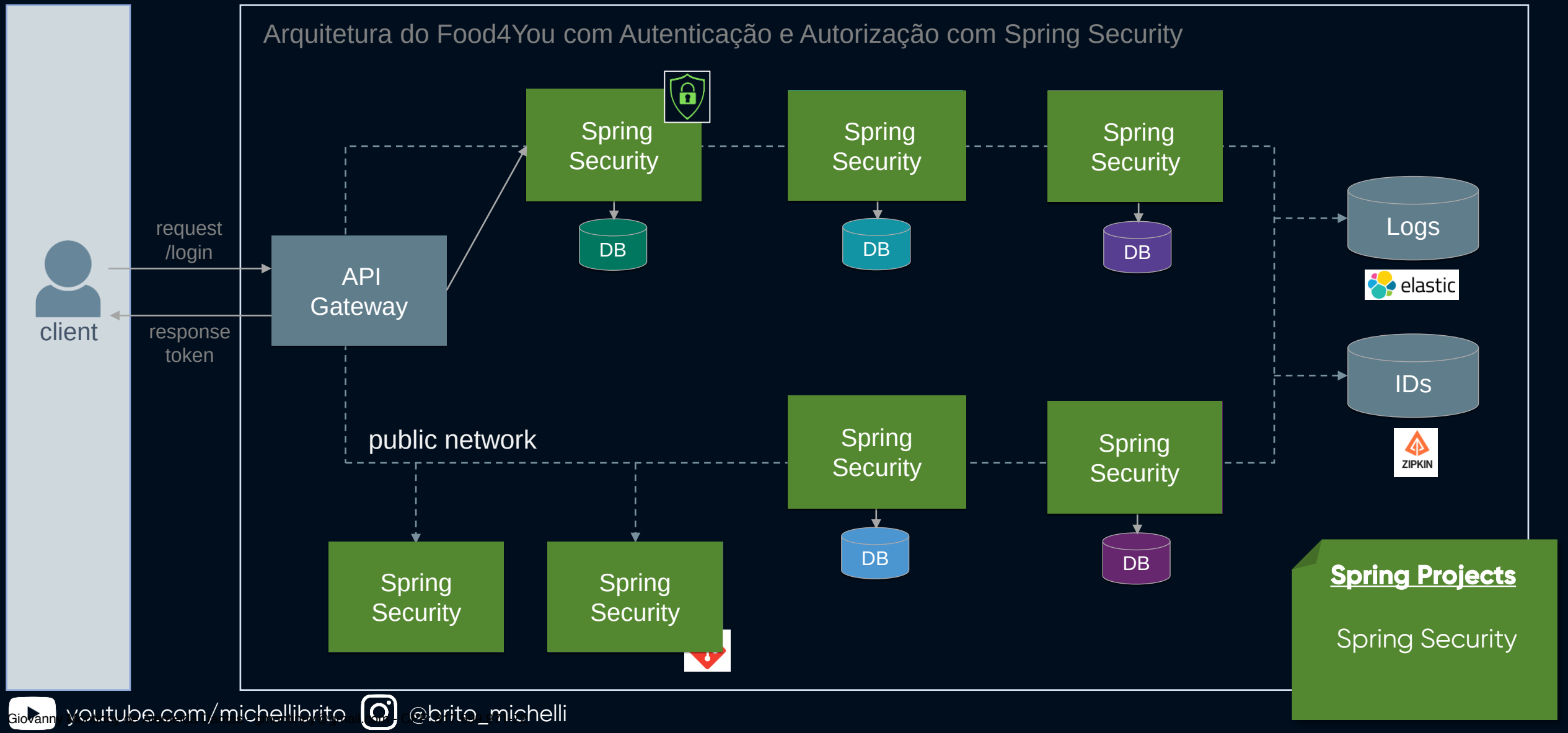
Modelo 1: Autenticação e Autorização no Gateway



Modelo 2: Autenticação e Autorização Distribuída



Autenticação e Autorização com Spring Security



COMPLEMENTANDO

Spring Security

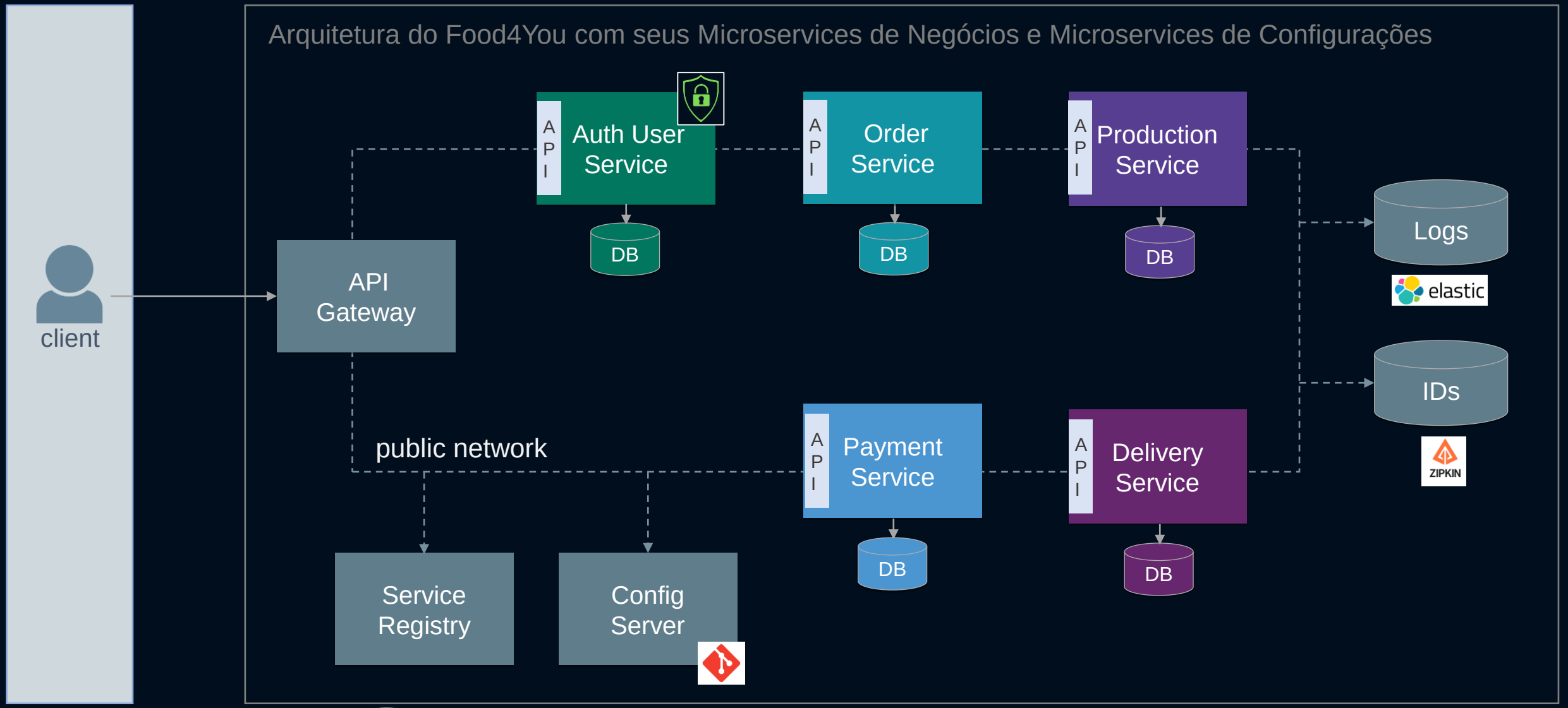
Projeto que fornece autenticação e autorização para aplicações Java

Suporte para implementação de Basic Authentication em memória e em database, JWT e OAuth

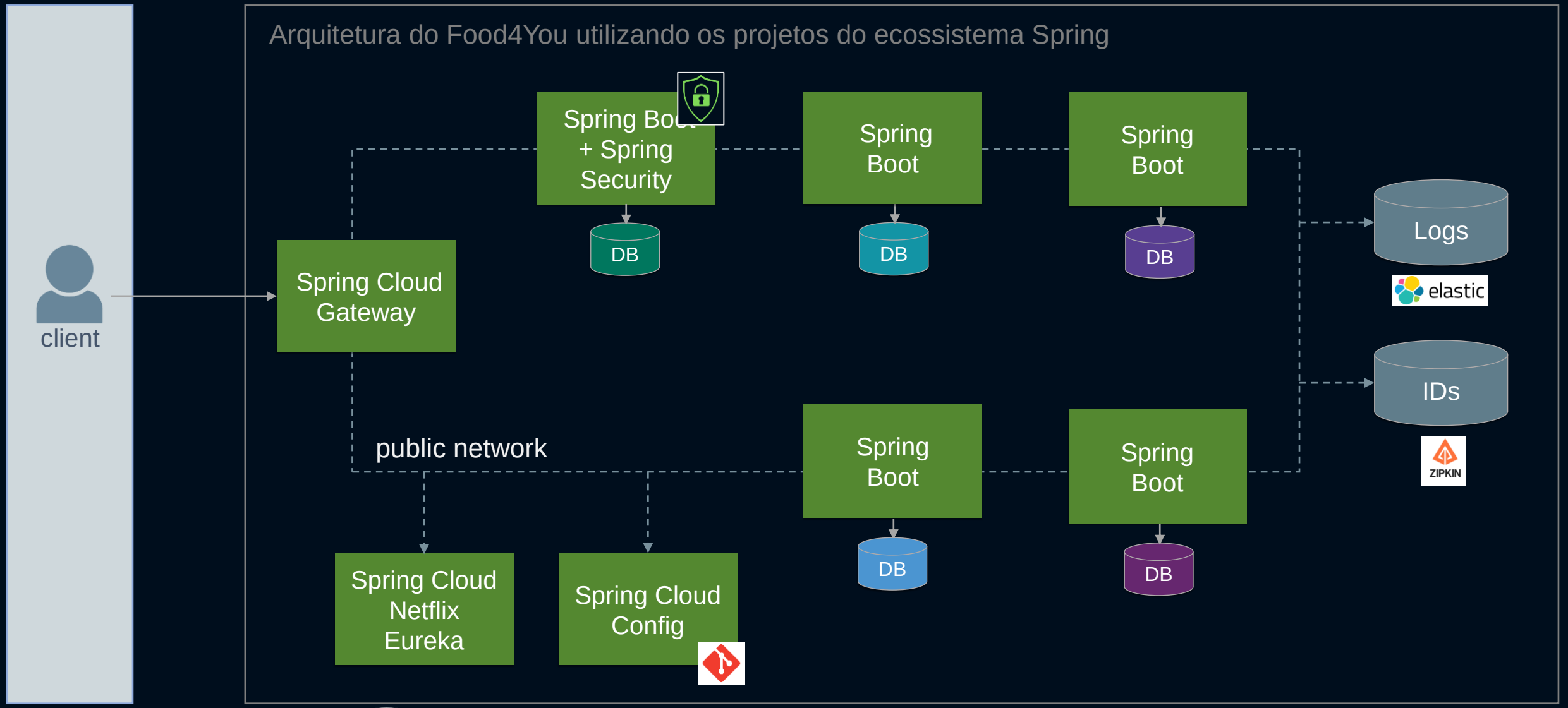
```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-security</artifactId>
</dependency>
```

Arquitetura Completa

Arquitetura completa do **Food4You**



Arquitetura completa do **Food4You**



O entendimento do negócio juntamente com as boas práticas e padrões são essenciais para você fazer as melhores escolhas e definir as melhores soluções e assim ser um profissional mais qualificado no mercado.

A discussão de Microservices não deve sobre o tamanho ou complexidade do negócio, mas principalmente sobre projetar sistemas sustentáveis.