

Fast & Foodious!

Grupo 34

Felipe Maximiliano da Rosa <felipemax.suporteti@gmail.com>

Rodrigo Ottero <ottero@gmail.com>

Renato Rodrigues Silva <spamcares-fiap@yahoo.com>



Fast & Foodious!

Agenda

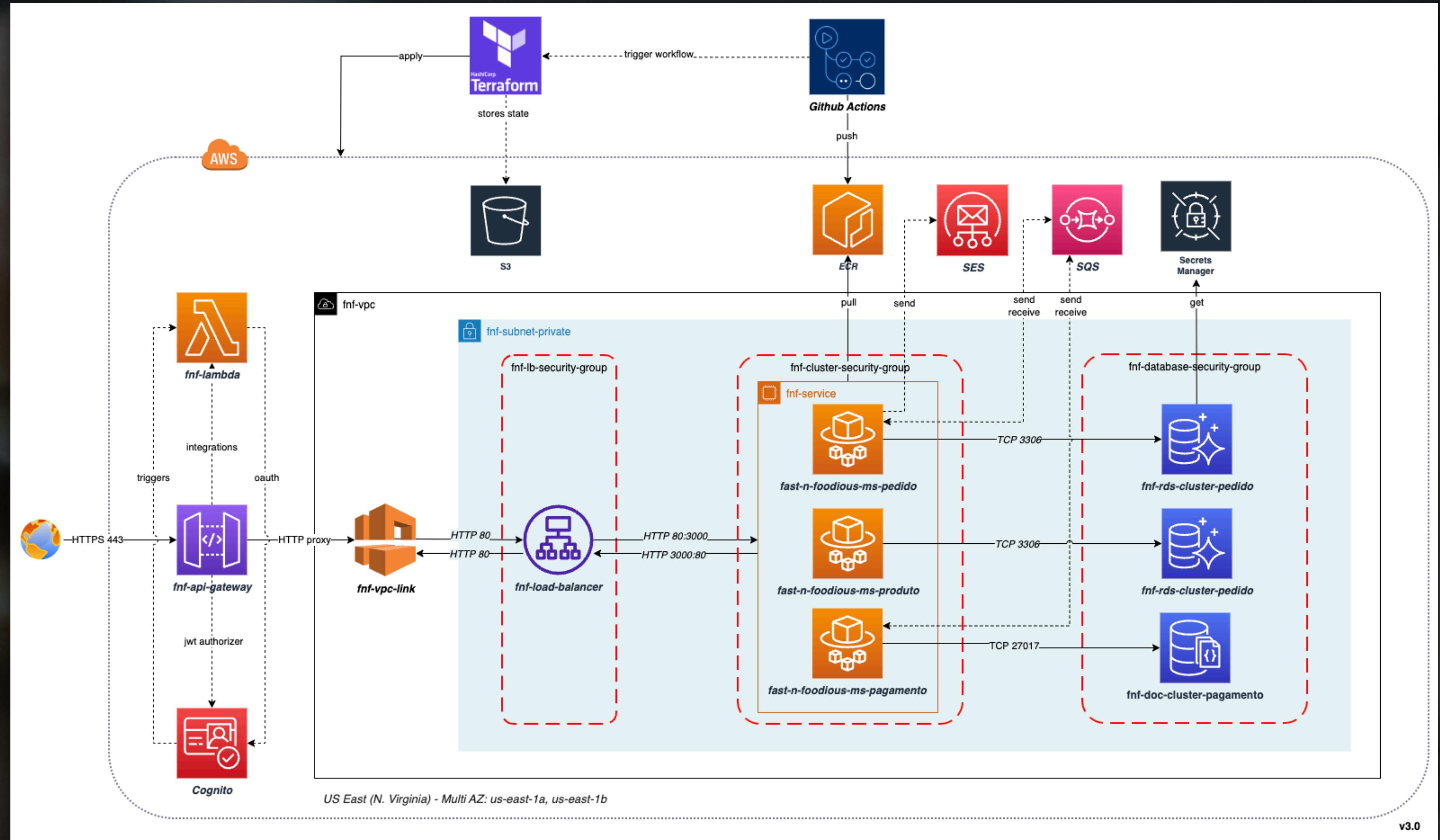
- Arquitetura Cloud AWS
- Transições de Estados
- SAGAs
 - Justificativa: Coreografia
 - Checkout: Solicitação de Pagamento
 - Webhook: Pagamento Confirmado
 - Webhook: Pagamento Rejeitado
- Demo
 - AWS Console: overview dos principais componentes
 - Postman
 - Execução do Cadastro de Cliente
 - Execução do Processo de Realização de Pedidos (Confirmado/Rejeitado)
 - Exibição de Notificações de Pagamento
 - Execução de Deleção de dados do Cliente
 - Exibição de Filas de Processamento
 - Logs CloudWatch



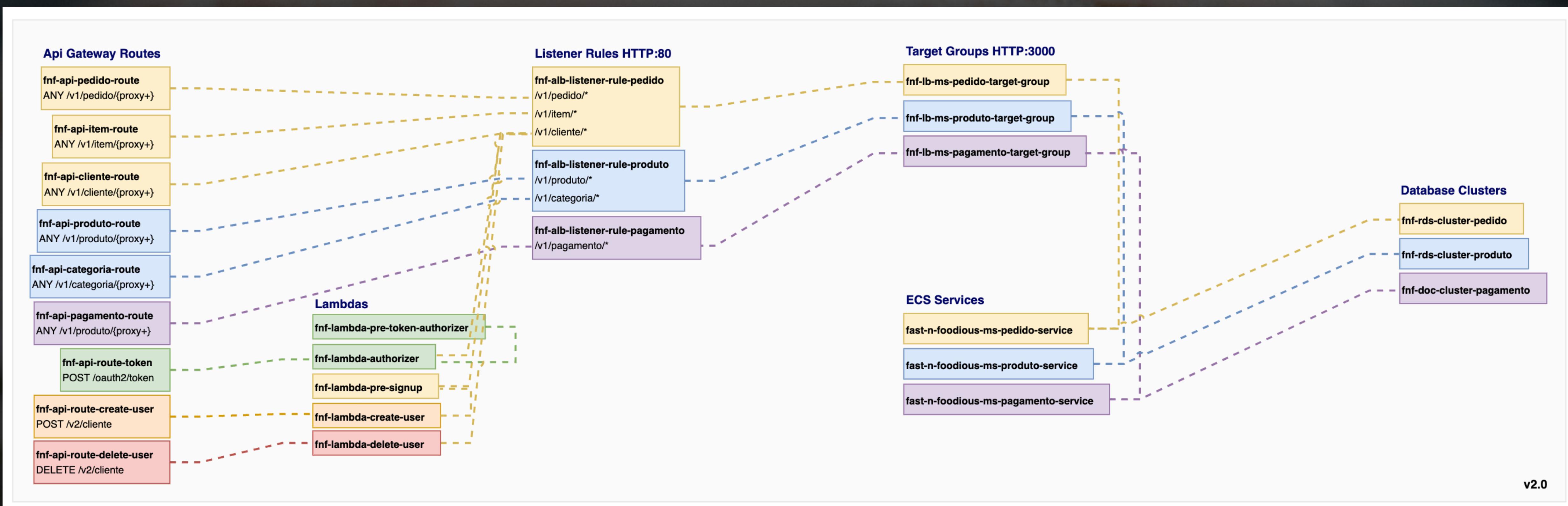
ARQUITETURA



Lambda
Api Gateway
Cognito
S3
VPC
Security Groups
Load Balancer
ECS
Fargate
ECR
Secrets Manager
Aurora RDS
DocumentDB
SES
SQS

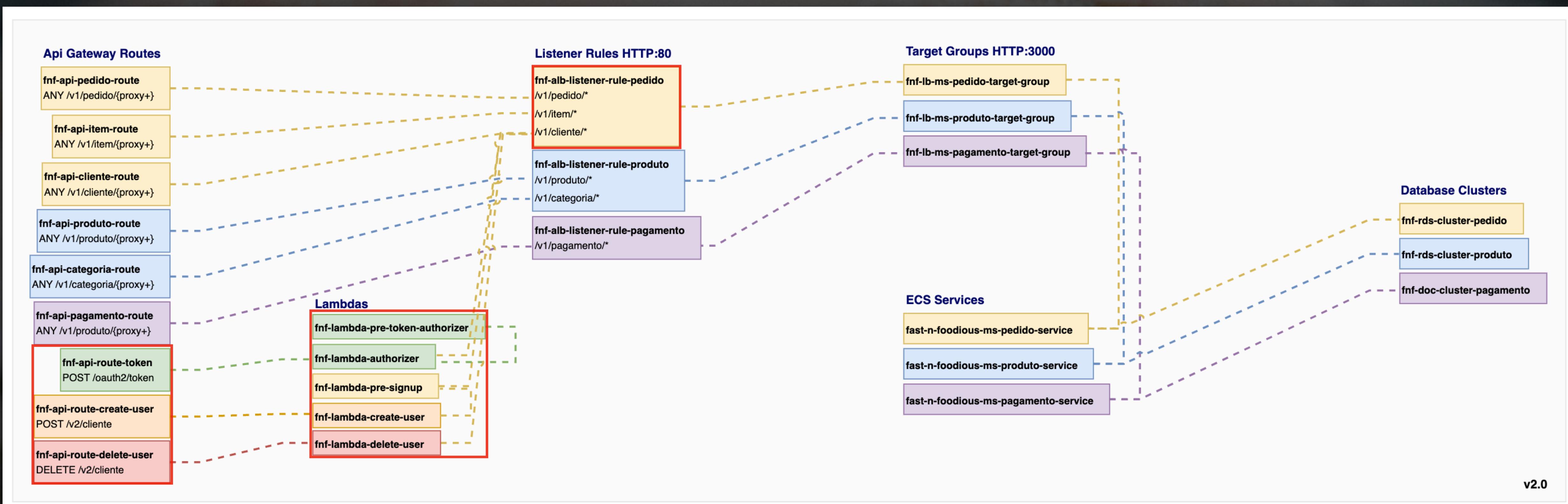


Overview da iteração entre os principais recursos de infraestrutura (network, compute, storage)



v2.0

Overview da iteração entre os principais recursos de infraestrutura (network, compute, storage)

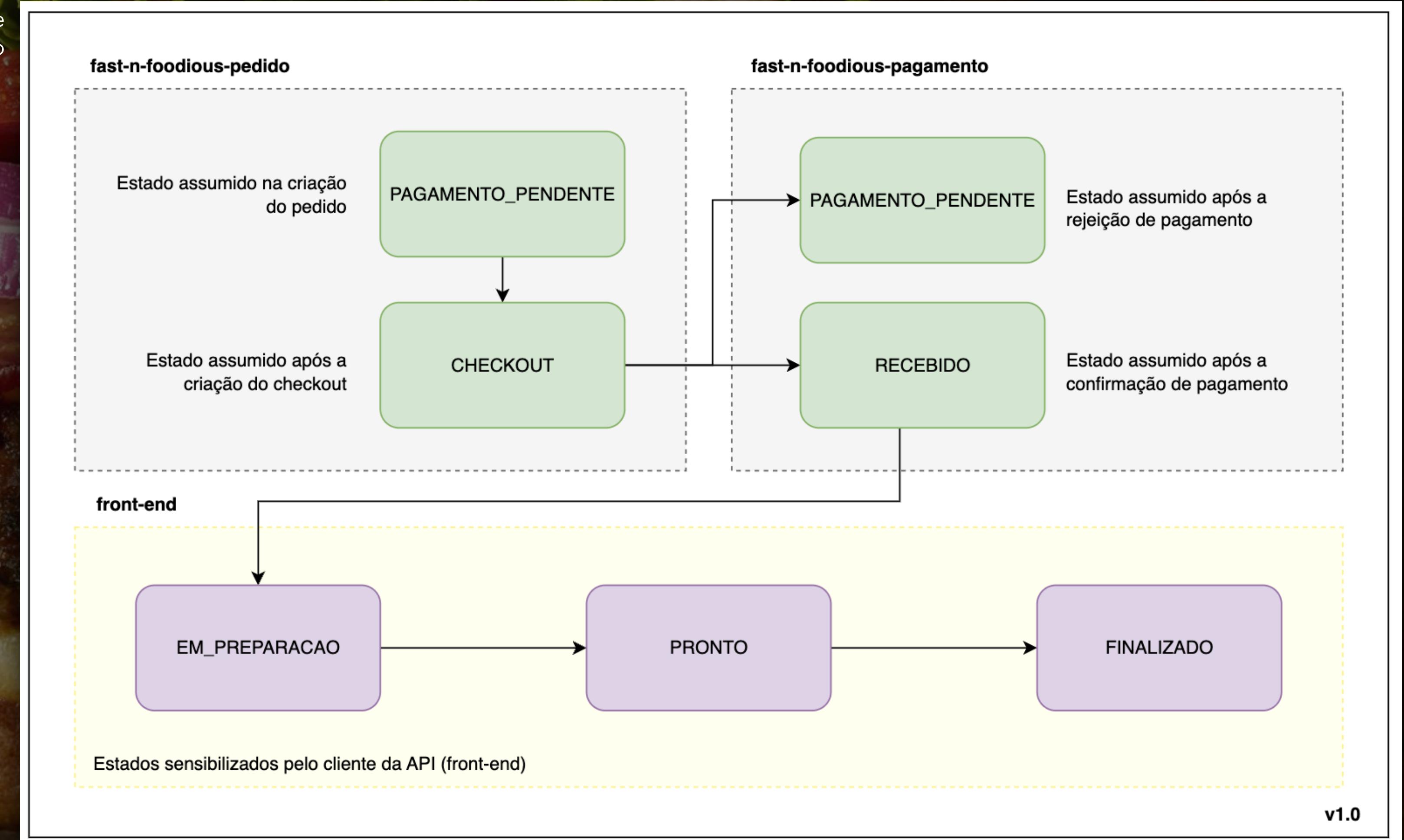


v2.0

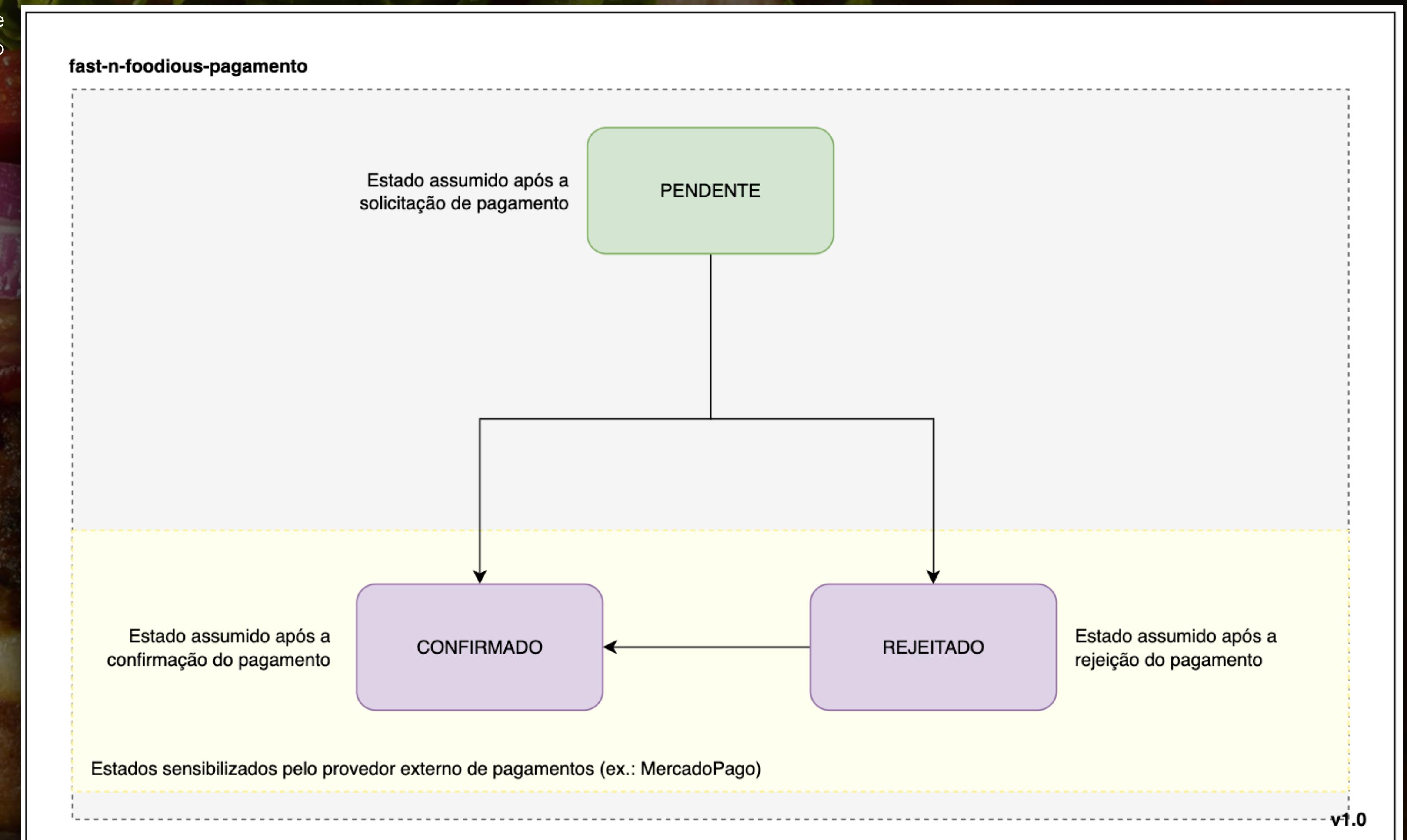
Transições de Estados



Componentes que sensibilizam as mudanças de estados do pedido



Componentes que sensibilizam as mudanças de estados do pagamento





SAGAS



Facilidade de implementação no cloud provider

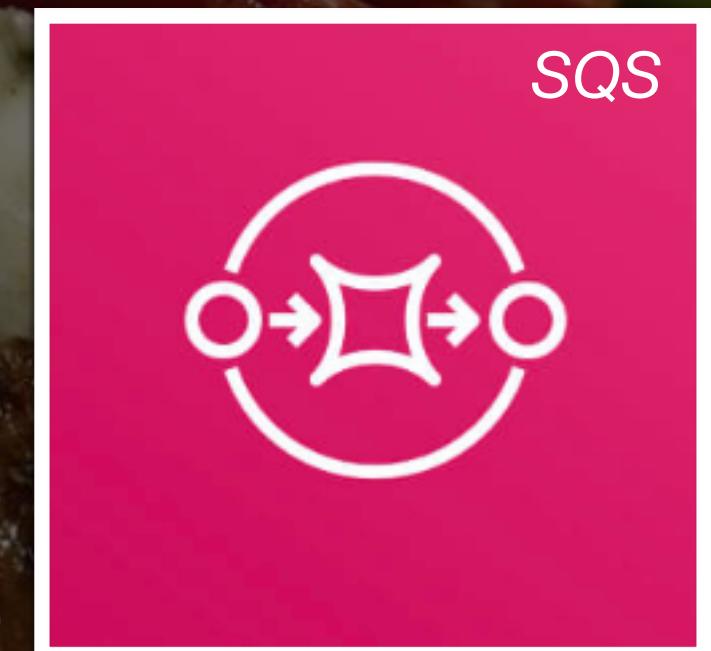
Sem adição de complexidade adicional

Coordenação direta entre micro serviços participantes

Desacoplamento e evolução independente

Uso de filas SQS de forma desacoplada

Tolerância a falhas através de retries & DLQs



C.o.r.e.o.g.r.a.f.i.a

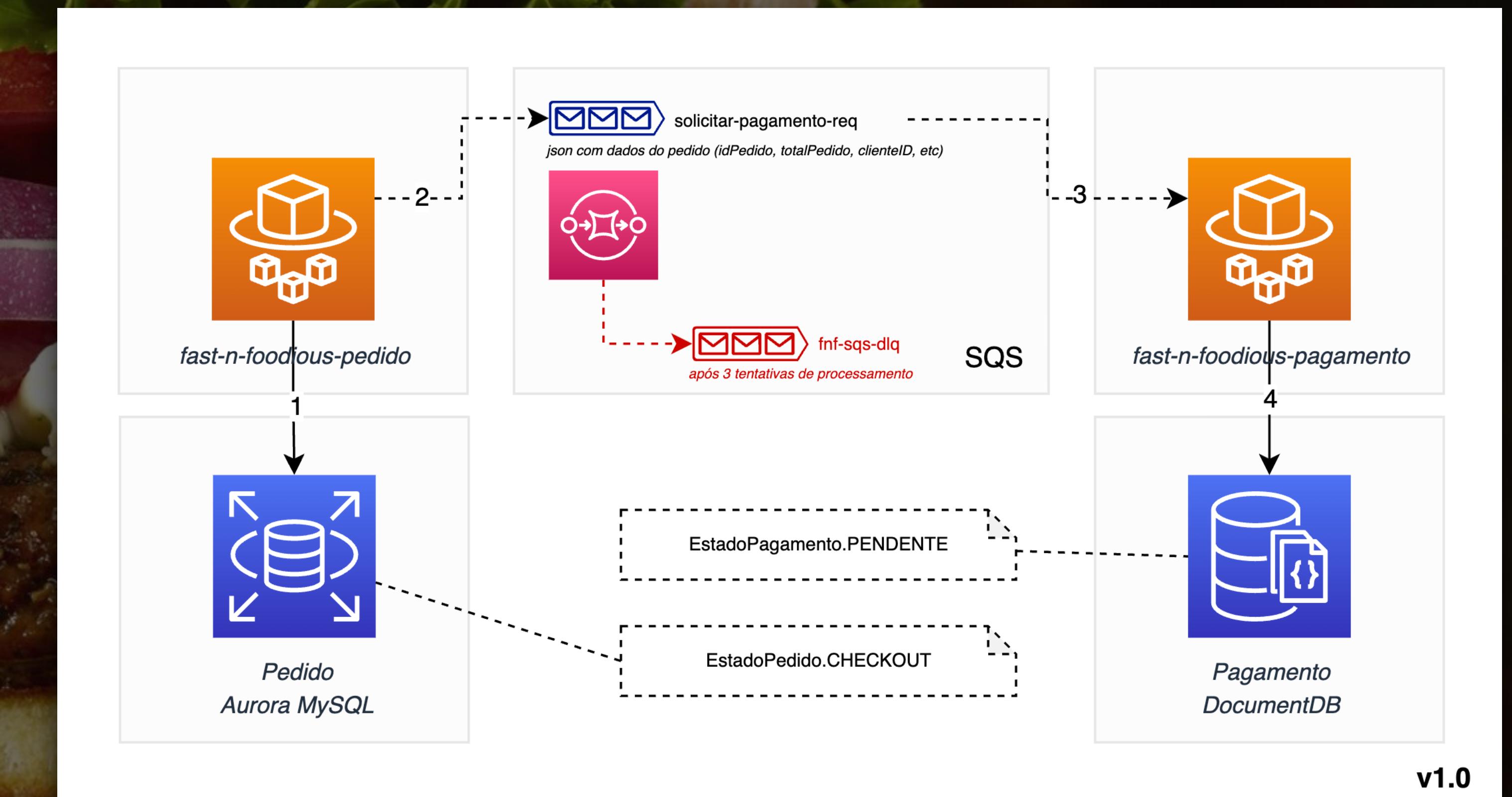
Integração entre micro serviços de pedido e pagamento, no processo de checkout

1-Atualização de estado de pedido para
'CHECKOUT' no Aurora Mysql

2-Publicação de mensagem com dados do pedido
(*idPedido*, *totalPedido*, *clienteID*, etc) na fila 'solicitar-
pagamento-req'

3-Consumo da fila 'solicitar-pagamento-req'

4-Atualização de estado do pagamento para
'PENDENTE' no DocumentDB



v1.0

1-Notificação do provider de pagamento, com o status de pagamento confirmado



Integração entre micro serviços de pagamento e pedido, no processo notificação de pagamento confirmado

2-Atualização de estado do pagamento para `CONFIRMADO` no DocumentDB



3-Publicação de mensagem com dados do pagamento (*idPagamento, transacaoId, etc*) na fila `webhook-pagamento-confirmado-res`



4-Consumo da fila `webhook-pagamento-confirmado-res`



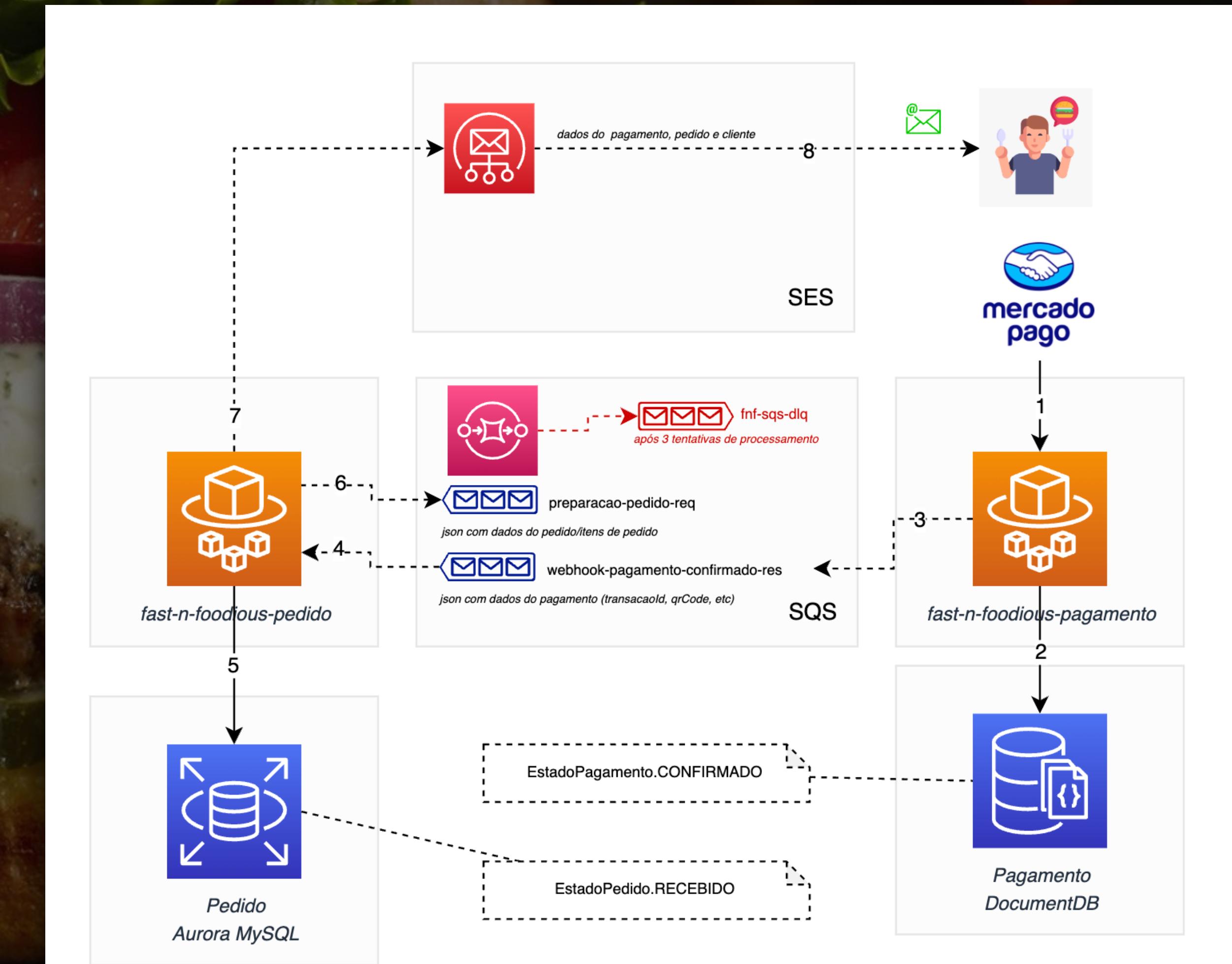
5-Atualização de estado de pedido para `RECEBIDO` no Aurora Mysql



6-Publicação de mensagem com dados do pedido (*pedidoid, itens, etc*) na fila `preparacao-pedido-req`



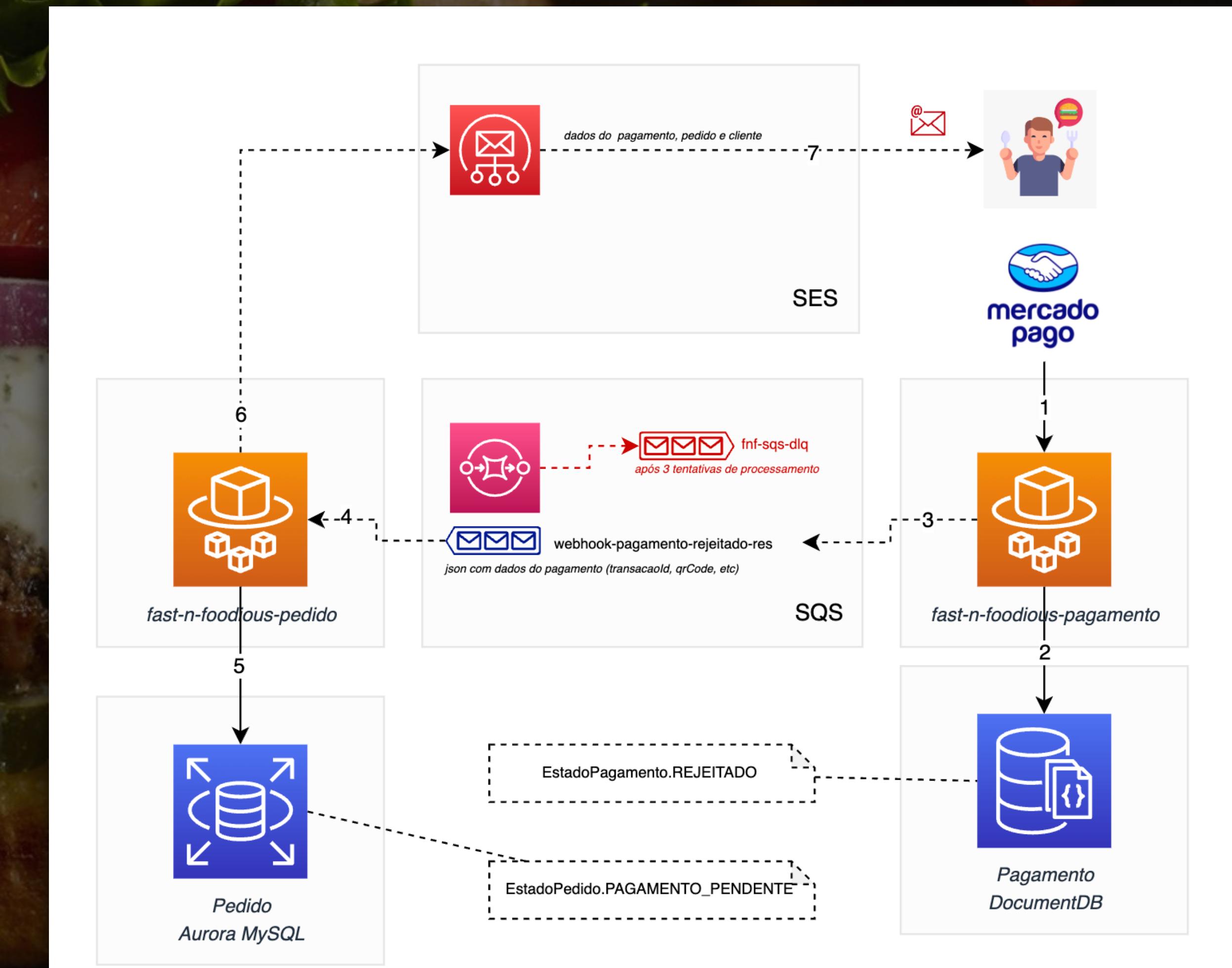
7/8-Envio de mensagem ao cliente sobre pagamento confirmado e início da preparação do pedido.



- 1-Notificação do provider de pagamento, com o status de pagamento rejeitado
- 2-Atualização de estado do pagamento para 'REJEITADO' no DocumentDB
- 3-Publicação de mensagem com dados do pagamento (`idPagamento`, `transacaoId`, etc) na fila `webhook-pagamento-rejeitado-res`
- 4-Consumo da fila `webhook-pagamento-rejeitado-res`
- 5-Atualização de estado de pedido para 'PAGAMENTO_PENDENTE' no Aurora Mysql
- 6/7-Envio de mensagem ao cliente sobre pagamento rejeitado.



Integração entre micro serviços de pagamento e pedido, no processo notificação de pagamento rejeitado



- AWS Console: overview dos principais componentes
- Postman
 - Execução do Cadastro de Cliente
 - Execução do Processo de Realização de Pedidos (Confirmado/Rejeitado)
 - Exibição de Notificações de Pagamento
 - Execução de Deleção de dados do Cliente
- Exibição de Filas de Processamento
- Logs CloudWatch

DEMO

