Alternativas Open Source para API Gateway/Manager

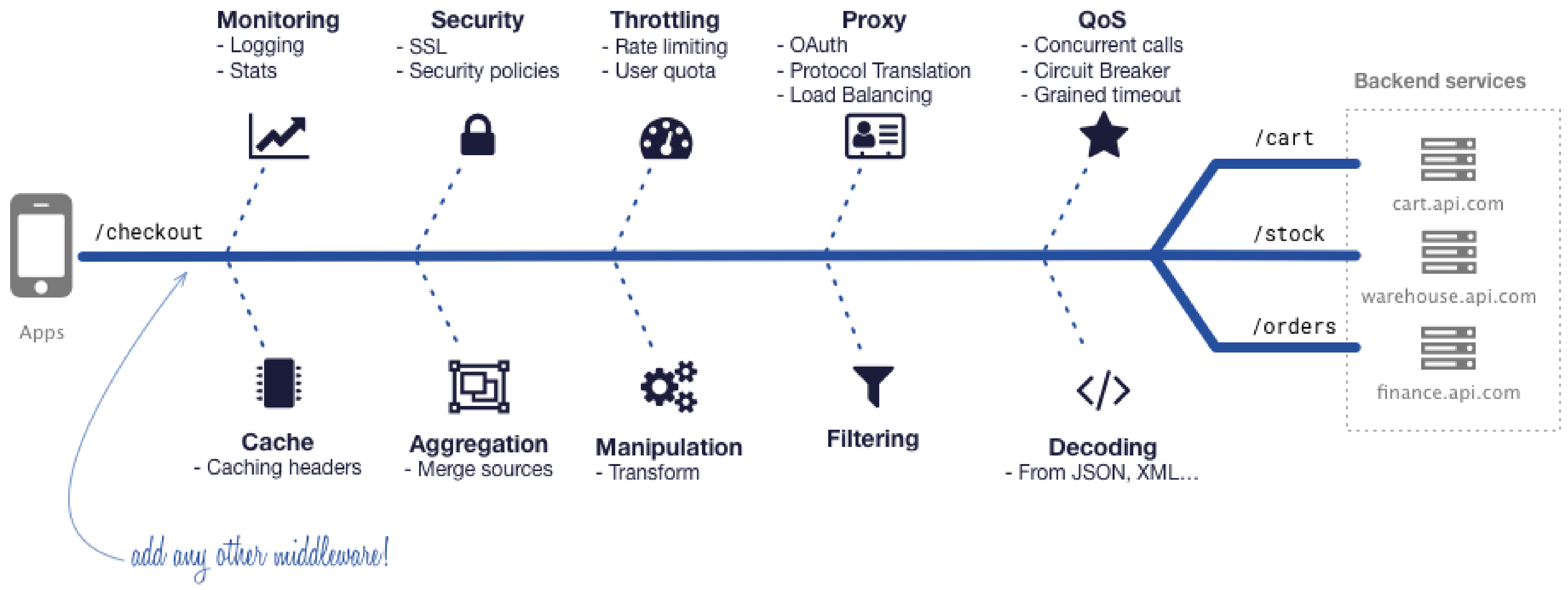
Boas Práticas e Hands On

Edgar Silva
QriarLabs, Co-Founder
edgar.silva@qriarlabs.com

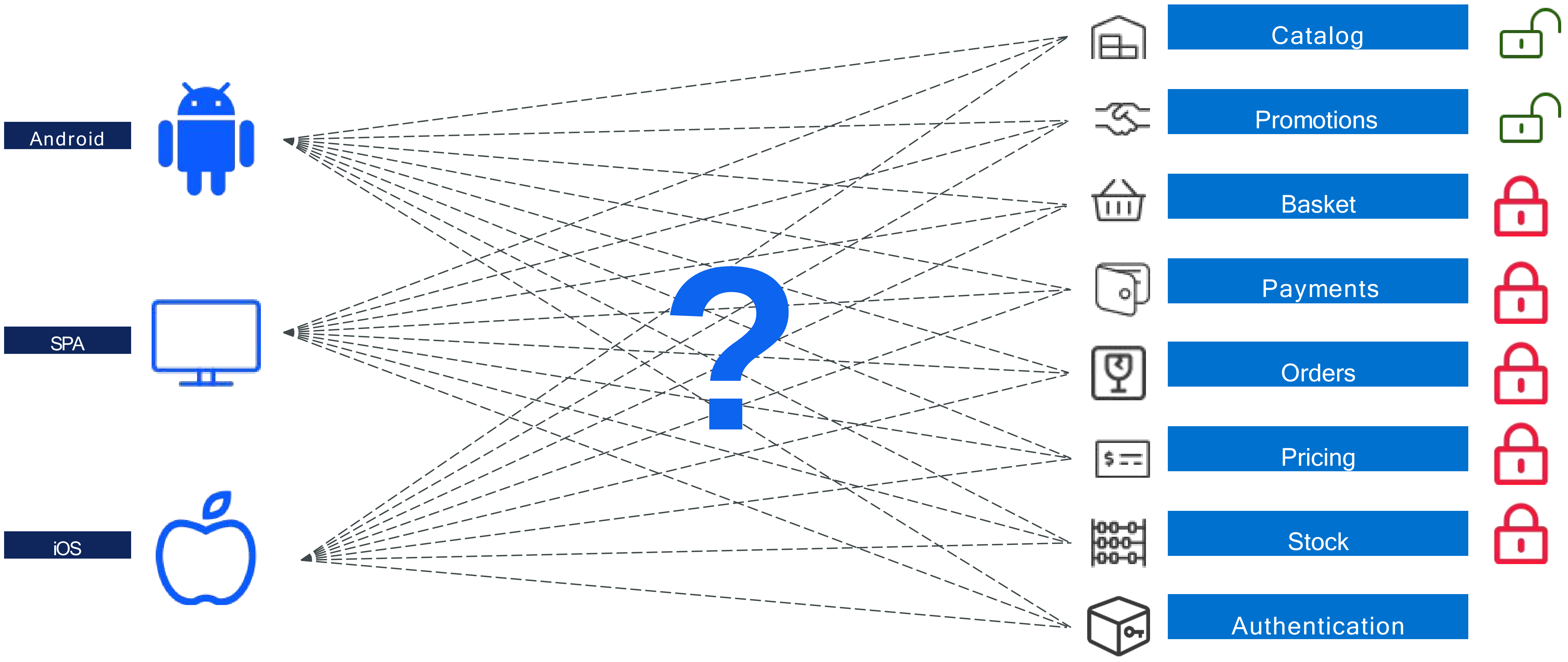
krakenD

https://www.krakend.io

qriarlabs

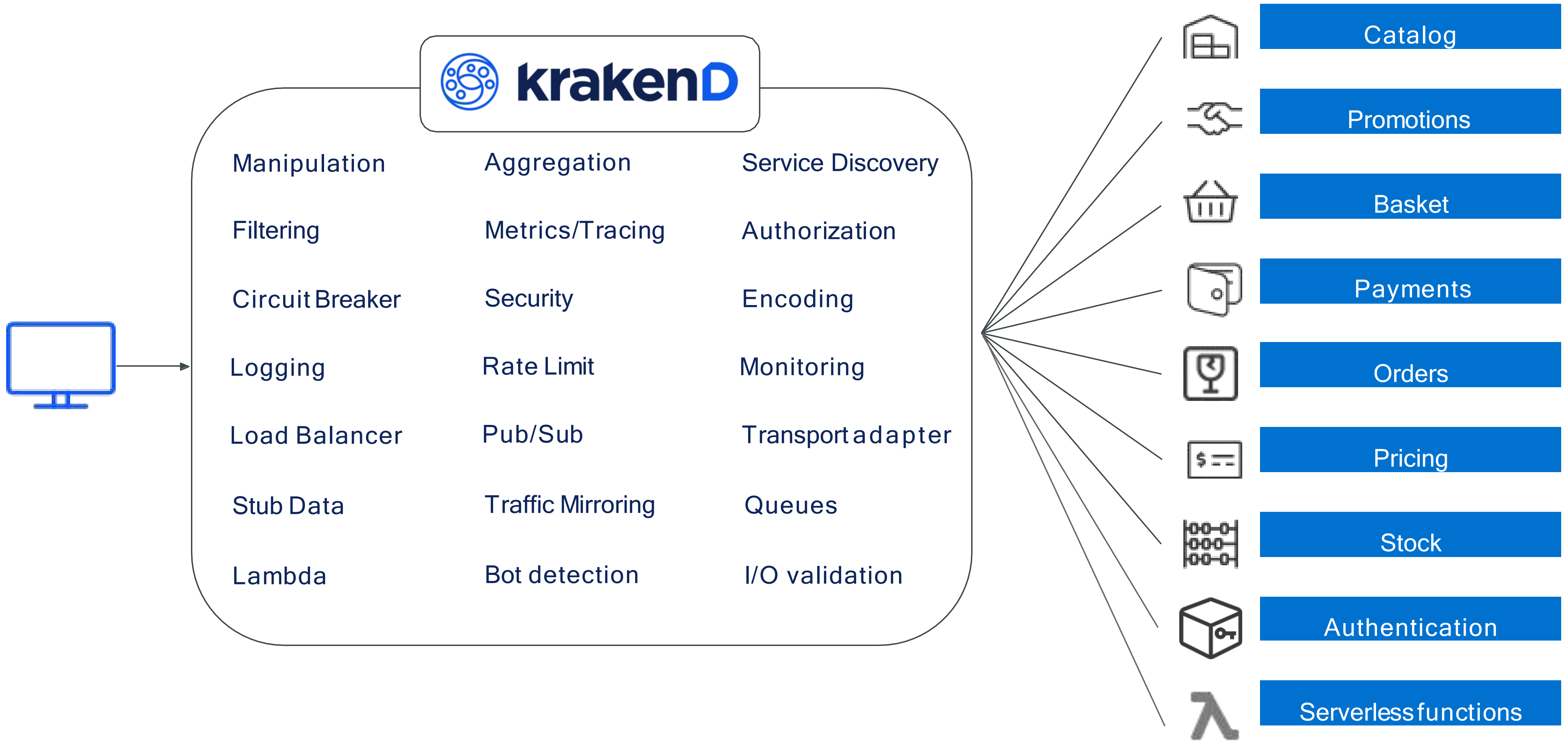

Monitoring
- Logging
- Stats
Security
- SSL
- Security policies
Throttling
- Rate limiting
- User quota
Proxy
- OAuth
- Protocol Translation
- Load Balancing
QoS
- Concurrent calls
- Circuit Breaker
- Grained timeout
Backend services
/cart
cart.api.com
/stock
warehouse.api.com
/orders
finance.api.com
/checkout
Apps
Cache
- Caching headers
Aggregation
- Merge sources
Manipulation
- Transform
Filtering
Decoding
- From JSON, XML...
add any other middleware!

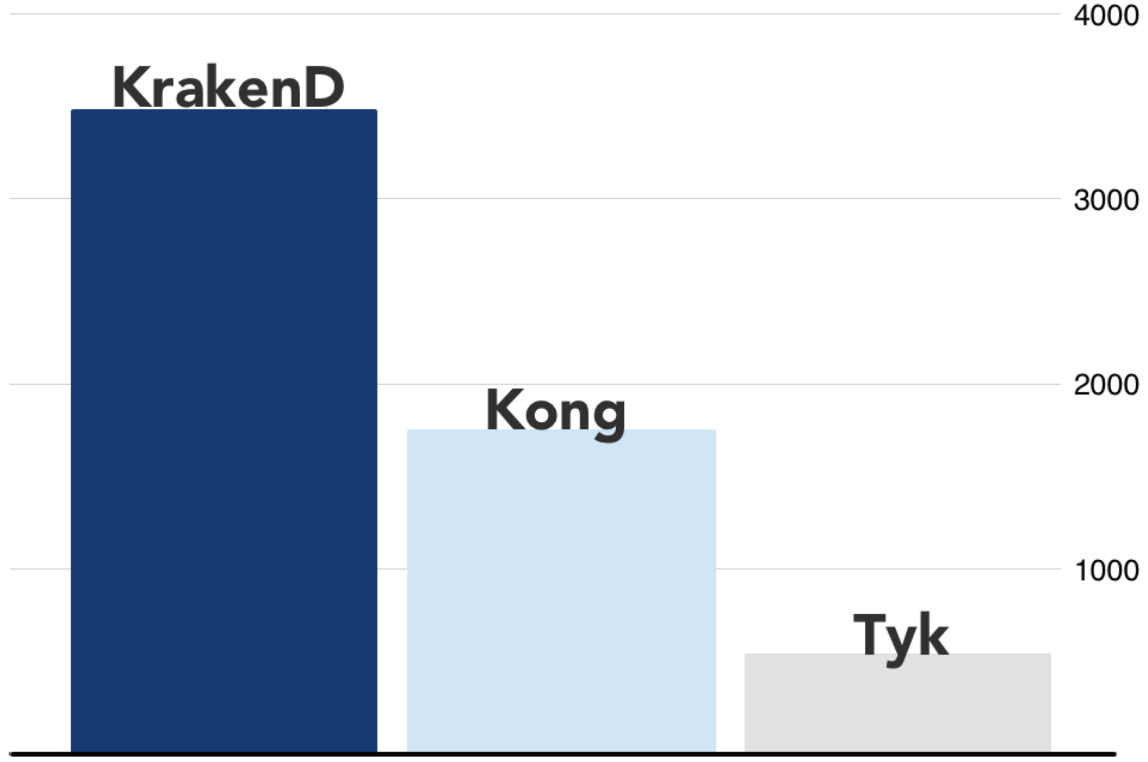
qriarlabs


Android
SPA
iOS
Catalog
Promotions
Basket
Payments
Orders
Pricing
Stock
Authentication

qriar labs

KrakenD offloads shared needs





+70,000 requests/second on commodity hardware

GO

A gateway is not the new monolith

KrakenD

Stateless

- ★ No node coordination
- ★ No synchronization
- ★ Zero complexity
- ★ No challenges for Multi-region
- ★ Declarative configuration
- ★ Immutable infrastructure

LINEAR SCALABILITY

Other gateways

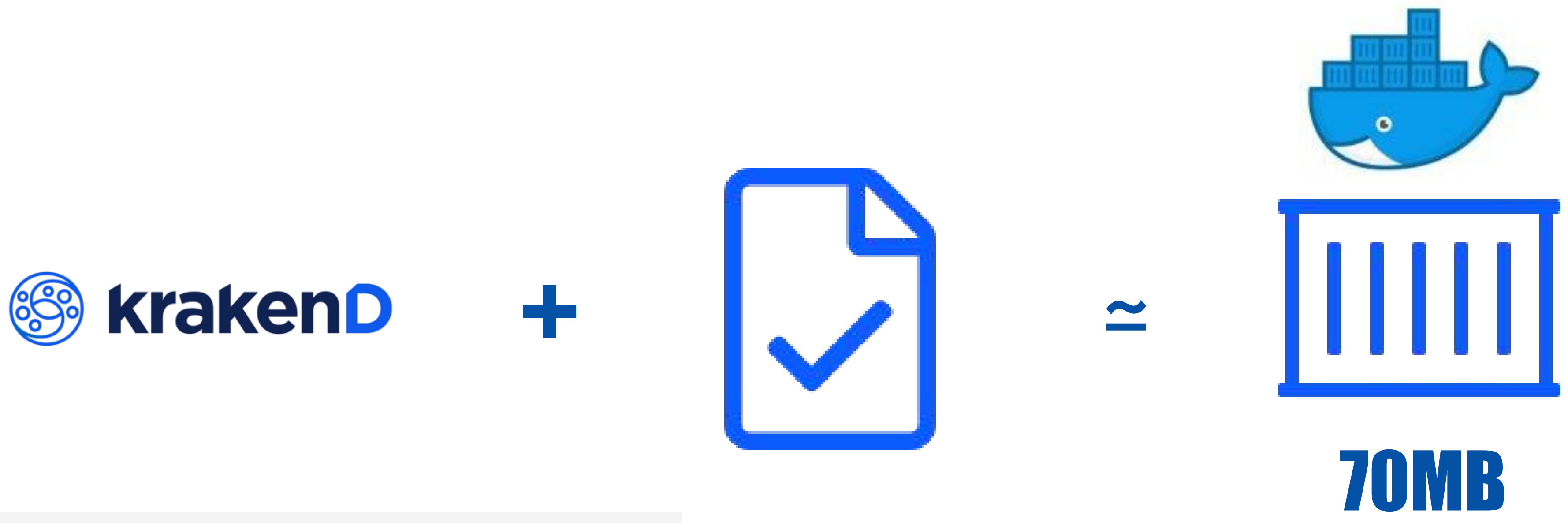
Stateful

- ★ Coordination required
- ★ Data synchronization
- ★ Datastore as source of truth
- ★ Complexity
- ★ Multi-region lag
- ★ Mutable configuration

NON-LINEAR SCALABILITY

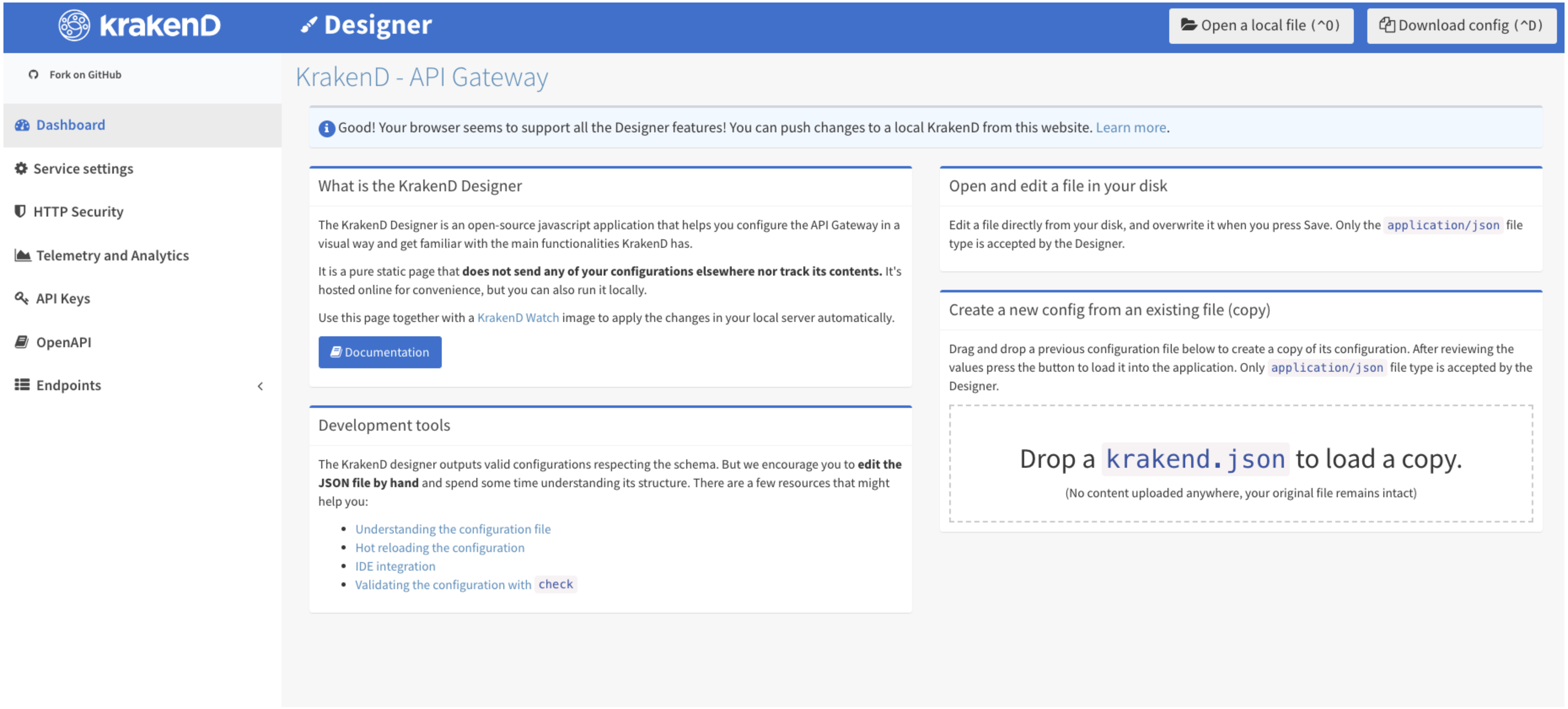
DB-LESS != stateless

Simple deployment (stateless)



Dockerfile

```
FROM devopsfaith/krakend


COPY krakend.json \
/etc/krakend/krakend.json
```



KrakenD - API Gateway

Good! Your browser seems to support all the Designer features! You can push changes to a local KrakenD from this website. Learn more.

What is the KrakenD Designer

The KrakenD Designer is an open-source javascript application that helps you configure the API Gateway in a visual way and get familiar with the main functionalities KrakenD has.

It is a pure static page that **does not send any of your configurations elsewhere nor track its contents.** It's hosted online for convenience, but you can also run it locally.

Use this page together with a KrakenD Watch image to apply the changes in your local server automatically.

Development tools

The KrakenD designer outputs valid configurations respecting the schema. But we encourage you to **edit the JSON file by hand** and spend some time understanding its structure. There are a few resources that might help you:

- Understanding the configuration file
- Hot reloading the configuration
- IDE integration
- Validating the configuration with `check`

Open and edit a file in your disk

Edit a file directly from your disk, and overwrite it when you press Save. Only the `application/json` file type is accepted by the Designer.

Create a new config from an existing file (copy)

Drag and drop a previous configuration file below to create a copy of its configuration. After reviewing the values press the button to load it into the application. Only `application/json` file type is accepted by the Designer.

Drop a `krakend.json` to load a copy.

(No content uploaded anywhere, your original file remains intact)

qriar labs

Executando via Docker

```
docker run -p "3890:3890" -v $PWD:/etc/krakend/ devopsfaith/krakend:2.4.3 run -c krakend.json
```

```
$ http http://localhost:3890/v1/github-proxy
```

KrakenD applies **zero-trust** criteria to incoming requests. Unless explicitly added below, no query strings, headers, or cookies are forwarded to the backend service.

Path Param

```
{
"endpoint": "/v1/github/{user}",
"method": "GET",
"output_encoding": "json",
"backend": [
{
"url_pattern": "/users/{user}",
"encoding": "json",
"sd": "static",
"method": "GET",
"host": [
"https://api.github.com"
],
"disable_host_sanitize": false
}
]
}
```

qriar labs

Query Param

```
{
"endpoint": "/v1/issues-from-repo/{user}/{reponame}",
"method": "GET",
"output_encoding": "json",
"backend": [
{
"url_pattern": "/repos/{user}/{reponame}",
"encoding": "json",
"sd": "static",
"method": "GET",
"host": [
"https://api.github.com"
],
"disable_host_sanitize": false
}
],
"input_query_strings": [
"state"
]
}
```

Rate Limit

```
"endpoint": "/v1/limited-requests",
        "extra_config": {
                "qos/ratelimit/router": {
                        "@comment": "Client rate limit of 5 every 5m",
                        "client_max_rate": 5,
                        "every": "5m",
                        "strategy": "ip"
}
},
```

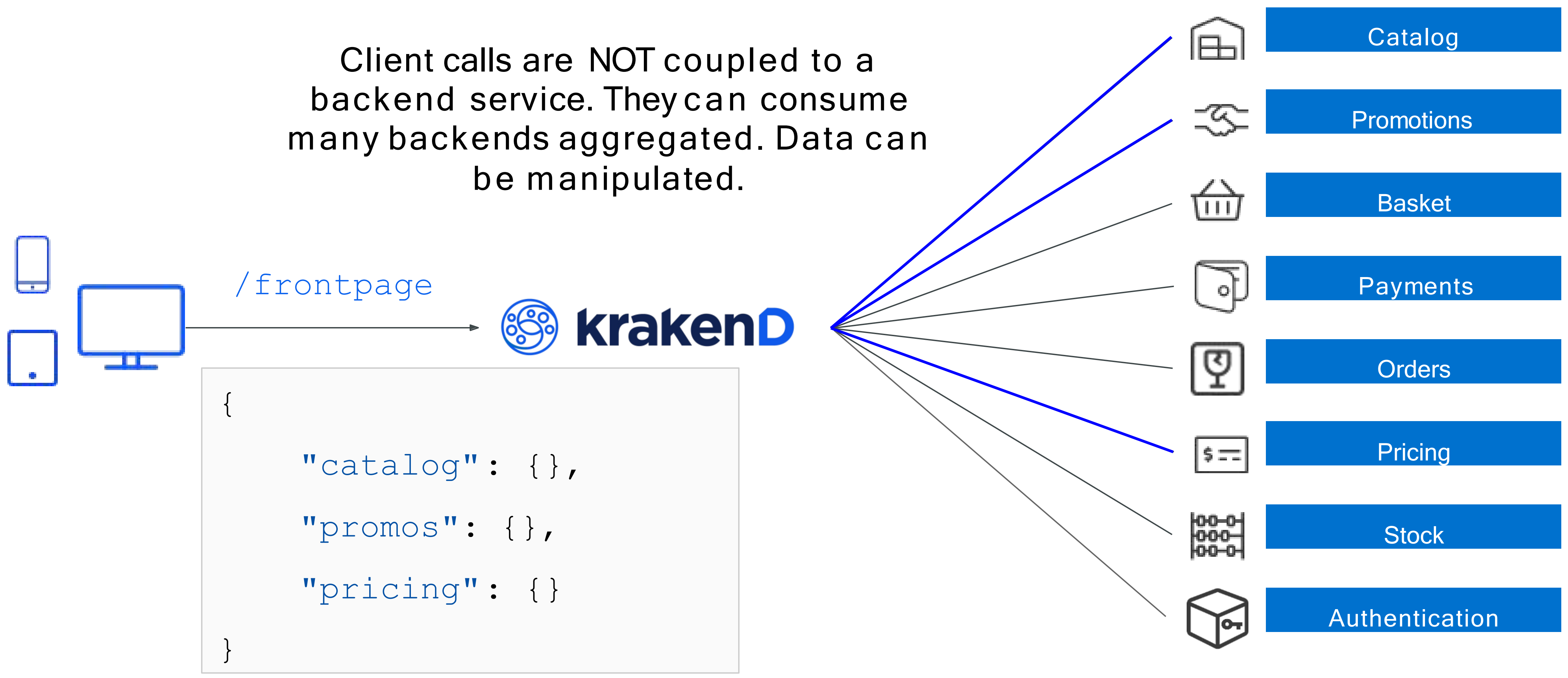
qriar labs

Headers

```
"endpoint": "/v1/limited-requests",
        "input_headers": [
                "Authorization"
        ],
"extra_config": {
"qos/ratelimit/router": {
"client_max_rate": 5,
"every": "5m",
"strategy": "ip"
}
},
```

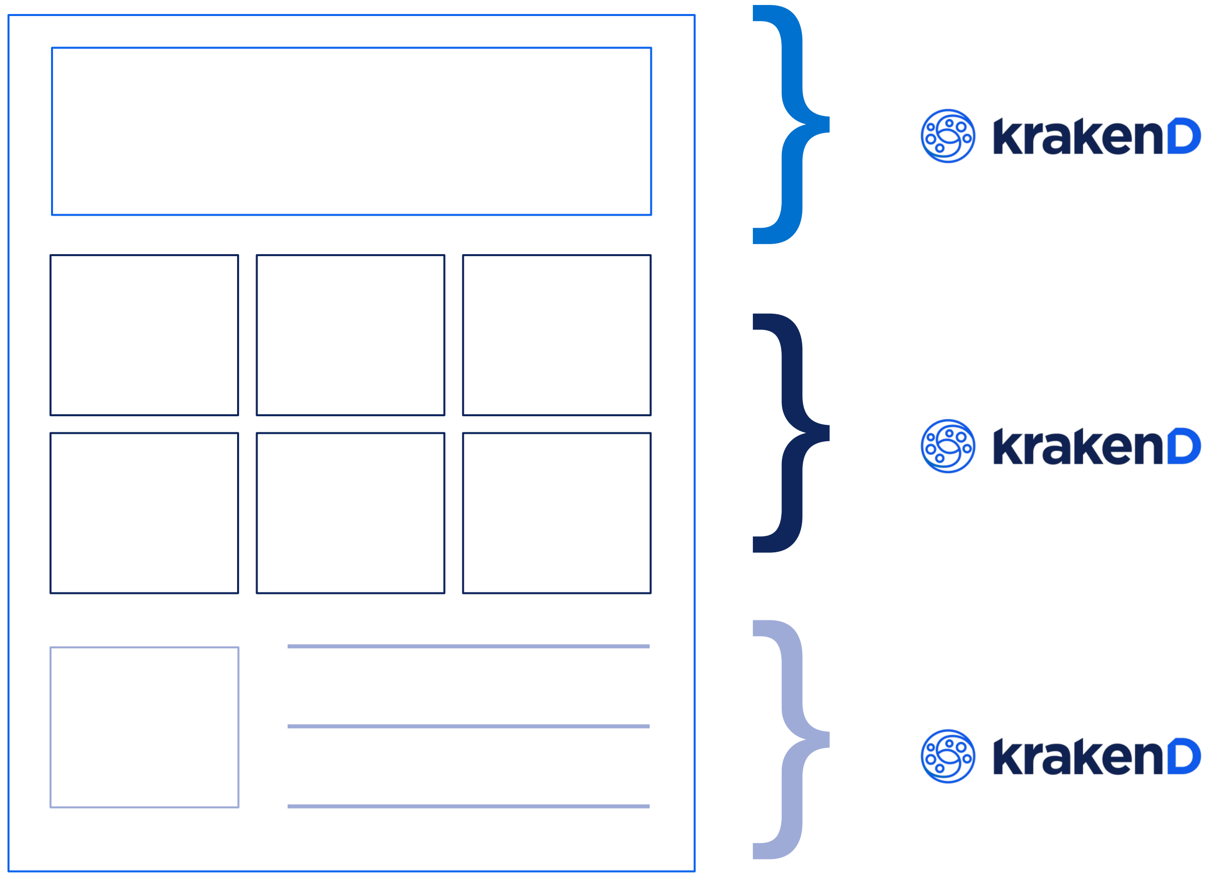
KrakenD API Gateway with Backend for Frontend

Client calls are NOT coupled to a backend service. They can consume many backends aggregated. Data can be manipulated.

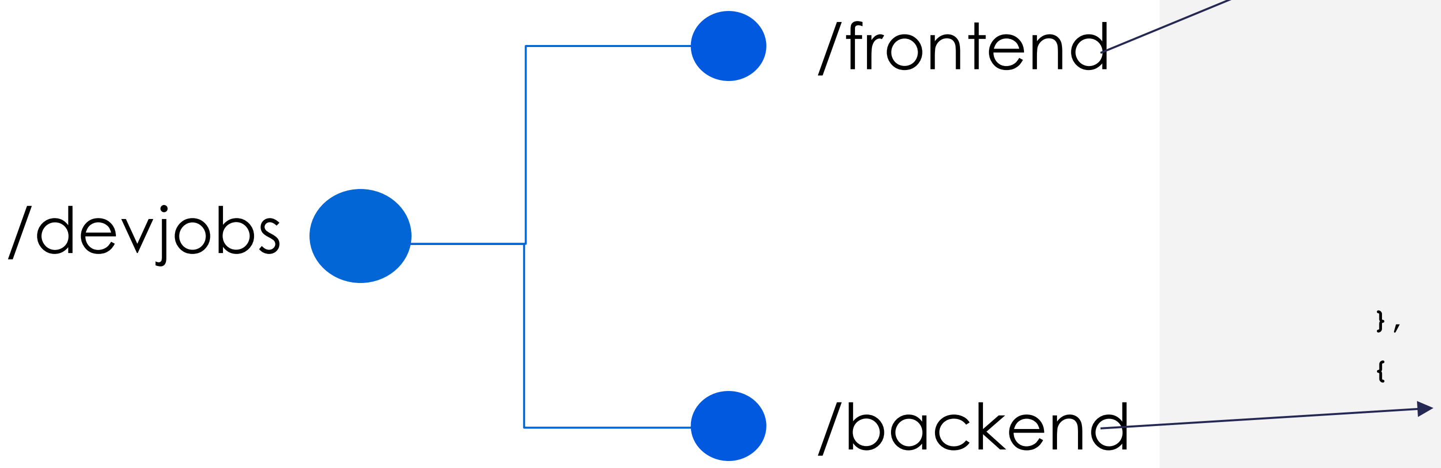


```
{
    "catalog": {},
    "promos": {},
    "pricing": {}
}
```

Assign a KrakenD to each team (micro frontends)



Merge



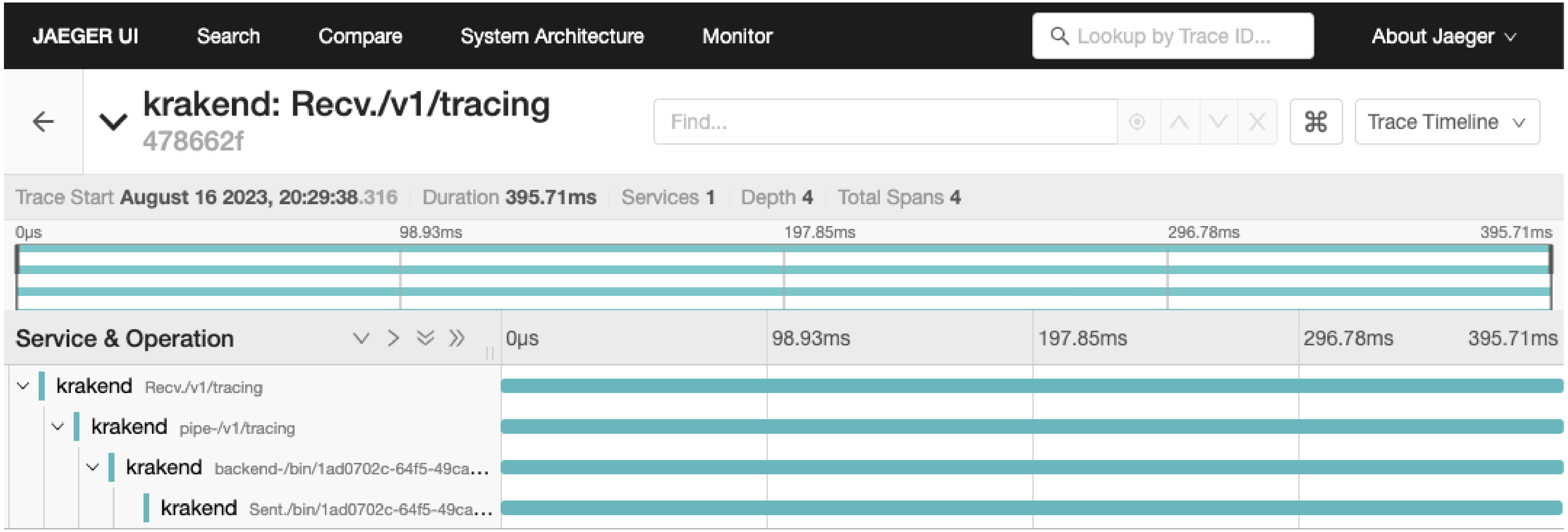
```
{
  "endpoint": "/devjobs",
  "backend": [{
          "url_pattern": "/repos/frontendbr/vagas/issues",
          "is_collection": true,
          "mapping": {
              "collection": "frontends"
          },
          "host": [
              "https://api.github.com"
          ]
      },
      {
          "url_pattern": "/repos/backend-br/vagas/issues",
          "is_collection": true,
          "mapping": {
              "collection": "backends"
          },
          "host": [
              "https://api.github.com"
          ]
      }

  ]
}
```

qriar labs

Telemetry

- https://mockbin.org/bin/1ad0702c-64f5-49ca-a12d-009659897249/view

```
version: '3'
services:
krakend:
image: devopsfaith/krakend
ports:
- "3890:3890"
volumes:
- /Users/edgar/OneDrive/skalena/2023/qriar/eventos/devops-bootcamp-01/krakend-06.json:/etc/krakend/krakend.json
jaeger:
image: jaegertracing/all-in-one:latest
ports:
- "3335:16686" # Jaeger UI
- "6831:6831" # Agent UDP Thrift
- 14268:14268 # http
environment:
- COLLECTOR_ZIPKIN_HTTP_PORT=9411

jaeger-ui:
image: jaegertracing/all-in-one
ports:
- "16686:16686"
environment:
- COLLECTOR_ZIPKIN_HTTP_PORT=9411
```


JAEGER UI Search Compare System Architecture Monitor Lookup by Trace ID... About Jaeger

krakend: Recv./v1/tracing
478662f

Find... Trace Timeline

Trace Start **August 16 2023, 20:29:38**.316 | Duration **395.71ms** | Services **1** | Depth **4** | Total Spans **4**

0µs 98.93ms 197.85ms 296.78ms 395.71ms

Service & Operation | 0µs | 98.93ms | 197.85ms | 296.78ms | 395.71ms

krakend Recv./v1/tracing
krakend pipe-/v1/tracing
krakend backend-/bin/1ad0702c-64f5-49ca…
krakend Sent./bin/1ad0702c-64f5-49ca…

qriarlabs

KrakenD

Segurança

- mTLS
- OAuth2 / JWT
  - Azure
  - Cognito
  - Firebase
  - Auth0/Okta
  - WSO2
  - Keycloak

```
"endpoint": "/v1/documentos-corp",
"method": "GET",
"output_encoding": "negotiate",
"input_headers": [
       "clientId",
       ”x-requested_nasph_uri",
       ”x-billing"
],
"extra_config": {
       "auth/validator": {
              "alg": "RS256",
              "jwk_url":
"http://keycloak:8080/auth/realms/api-
manager/protocol/openid-connect/certs",
              "disable_jwk_security": true,
              "operation_debug": true,
              "propagate_claims": [
                            [
                           "clientId",
                           "clientId"
                           ]
              ]
} ...
```

KrakenD

Async APIs

- Saga Pattern
    - O padrão saga é um padrão de design de transações distribuídas que coordena um processo que precisa ser realizado através de múltiplos serviços ou microserviços. Em vez de uma transação ACID tradicional, uma saga é composta por várias etapas ou transações locais.
- Event sourcing
    - Padrão de design em que as mudanças no estado de uma aplicação são armazenadas como uma sequência de eventos, em vez de apenas representar o estado em um determinado momento

```
{
"version": 3,
"async_agent": [
{
"name": "cool-agent",
"connection": {
"max_retries": 10,
"backoff_strategy": "exponential-jitter",
"health_interval": "10s"
},
"consumer": {
"topic": "*",
"workers": 1,
"timeout": "150ms",
"max_rate": 0.5
},
"backend": [
{
"host": [
"http://127.0.0.1:8080"
],
"url_pattern": "/__debug/"
}
],
....
```

```
...
"extra_config": {
"async/amqp": {
"host": "amqp://guest:guest@localhost:5672/",
"name": "krakend",
"exchange": "foo",
"durable": true,
"delete": false,
"exclusive": false,
"no_wait": true,
"prefetch_count": 5,
"auto_ack": false,
"no_local": true
}
}
}
]
}
```

qriar labs

Extensões (plugins)

- Lua Scripts
- Golang

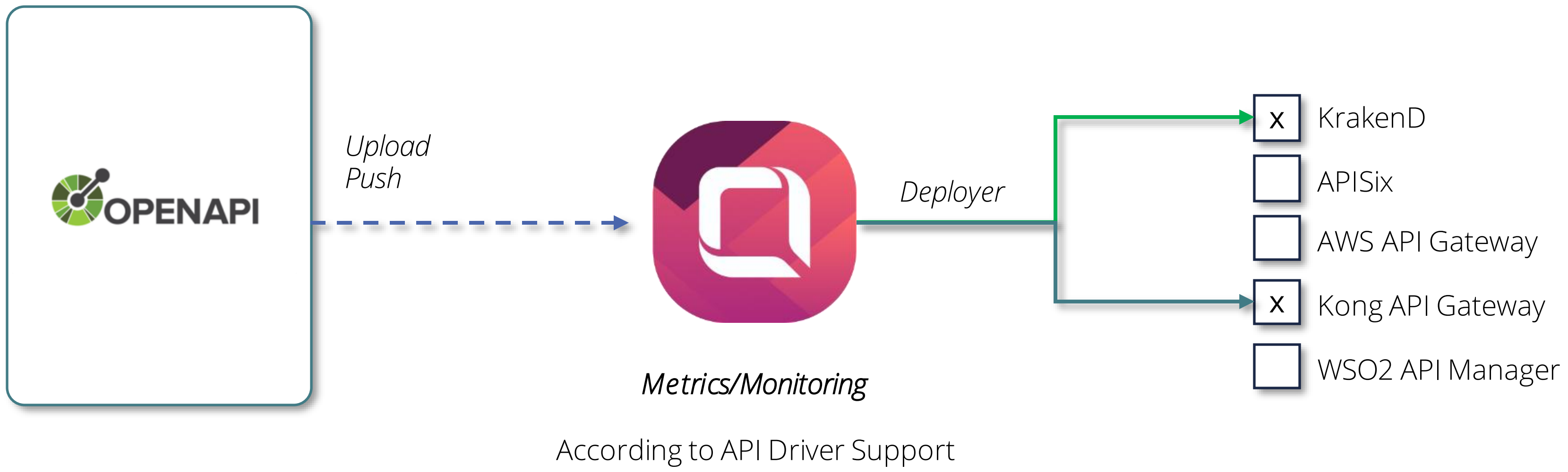
```
package main

import (
"context"
"fmt"
"net/http"
)
var HandlerRegisterer = registerer("billing")

type registerer string

func (r registerer) RegisterHandlers(f func(
name string,
handler func(context.Context, map[string]interface{}, http.Handler) (http.Handler,
error),
)) {
f(string(r), r.registerHandlers)
}

func (r registerer) registerHandlers(ctx context.Context, extra
map[string]interface{}, h http.Handler) (http.Handler, error) {

return http.HandlerFunc(func(w http.ResponseWriter, req *http.Request) {

req.Header.Set("X-requested_nasph_uri", req.URL.Path)

w.Header().Set("X-requested_nasph_uri", req.URL.Path)


….
```

```
"plugin": {
"pattern": ".so",
"folder": "./plugins-extensions/"
},
"extra_config": {
"plugin/http-server": {
"name": ["billing"]
}
},
```

qriar labs

API Orchestrator

Manage Multiple API Gateways/Managers



Nova tendência para o mercado de produtos de API

qriar labs

Edgar Silva

https://qriarlabs.com