CAPI Gateway

Light API Gateway implemented with Apache Camel

in/rodrigoserracoelho

/rodrigoserracoelho

Motivation

- Easy to customize API Gateway (since we have several customers, with a wide scope of needs and technologies, we need to be able to easily build custom components on top of an API Gateway core.)

  1 - Customize processors

  2 - Integrate with externals IdP

  3 - Implement custom API Manager Interface
- Light and easy to install.
- Scale **UP/DOWN**
- Decoupled components:

  1 - Observability

  2 - Metrics

  3 - Database

  4 - API Manager Interface

  5 - Authorization

Motivation

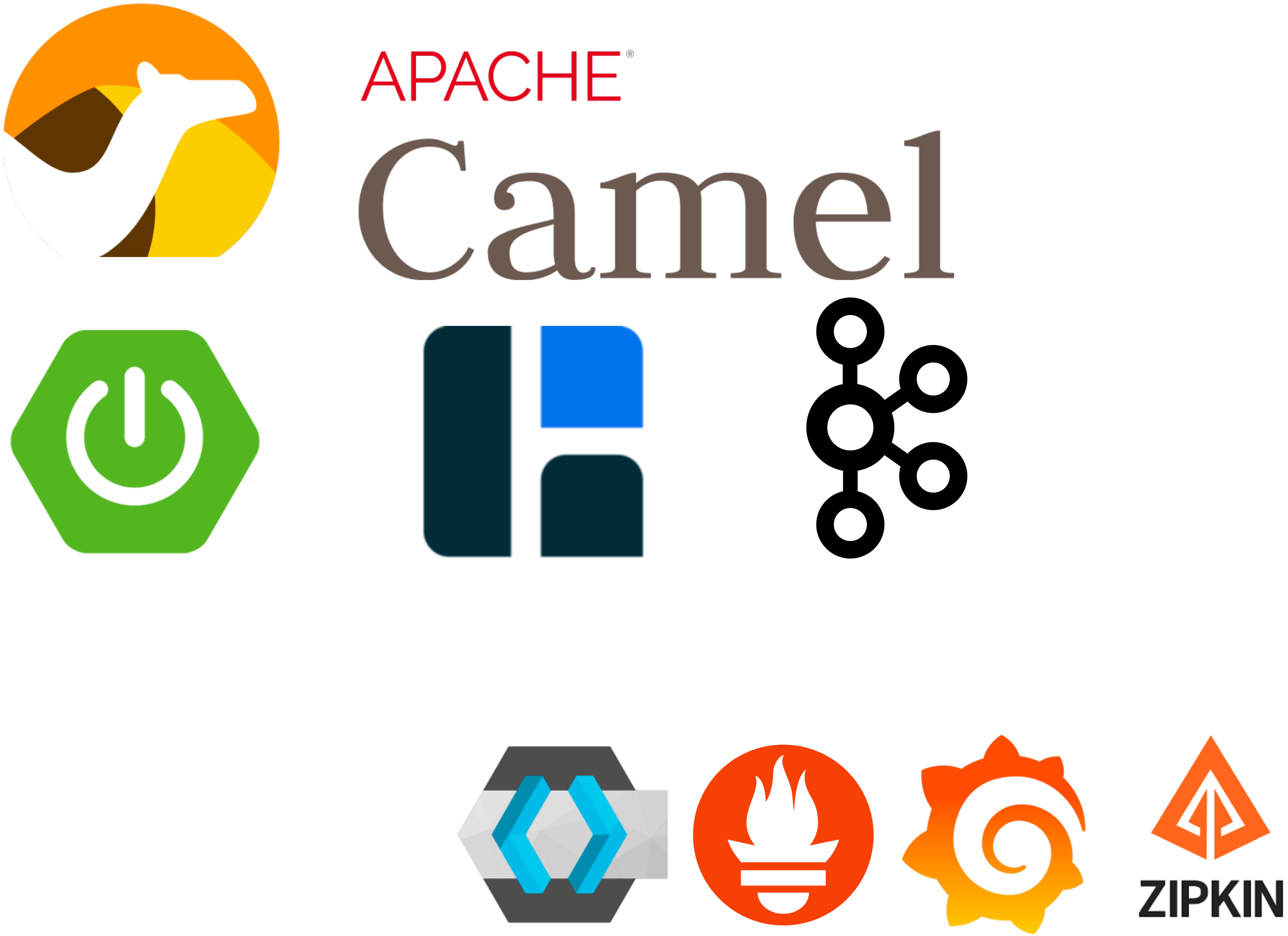
- Traffic management
- Automated traceability
- Automated Metrics
- Automated Dashboards
- Block APIs after failed attempts.
- REST endpoints to monitor and manage your API’s and Certificates.
- Authorization integration with Keycloak.

Why Camel?

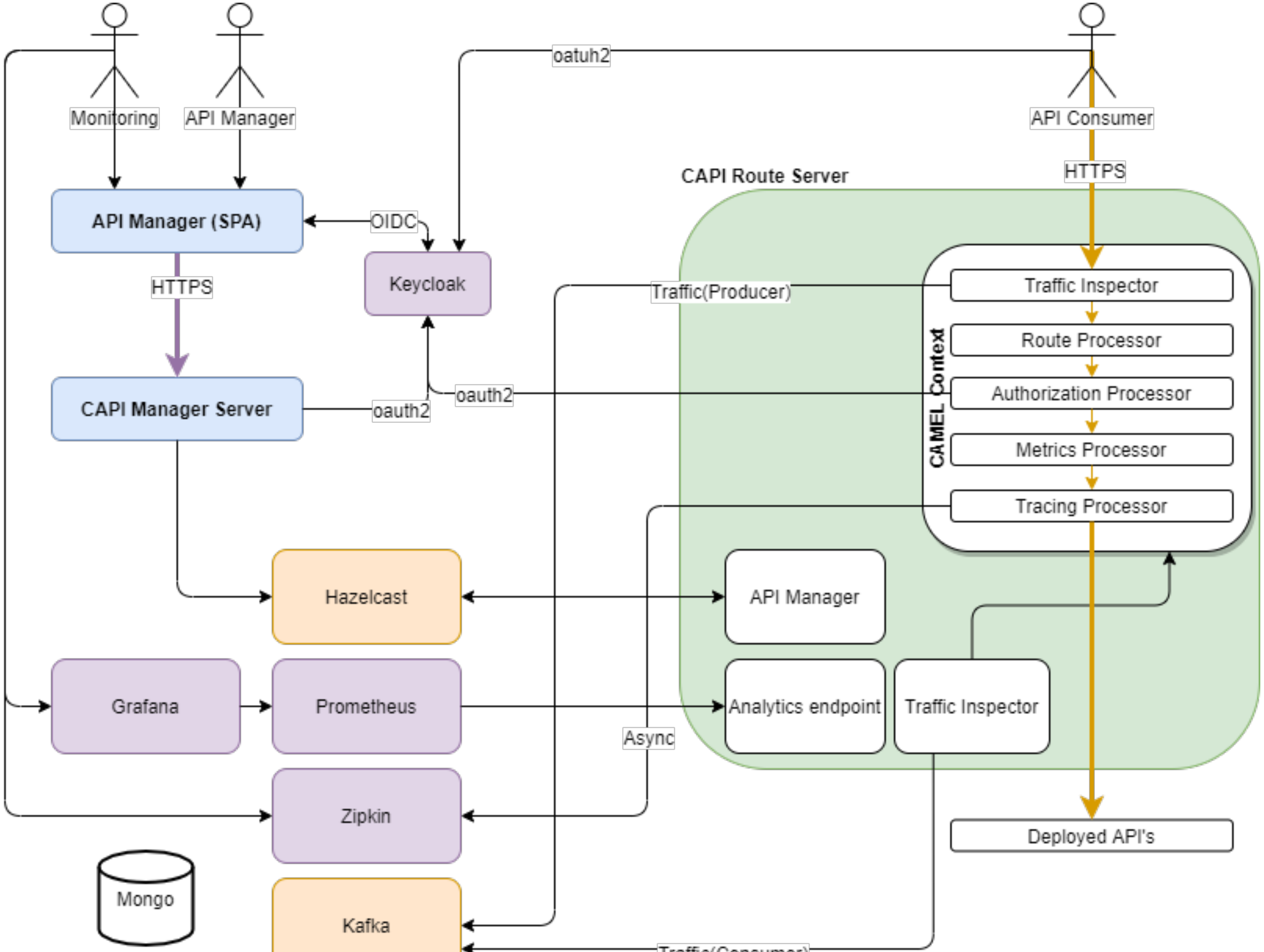
- Open Source.
- Big and active community.
- Fast and reliable.
- Deep integration with Spring Boot.
- DSL Language.
- Know-how in the team.
- It supports all the integration patterns that you can remember.

- Camek K
- gRPC
- Web sockets

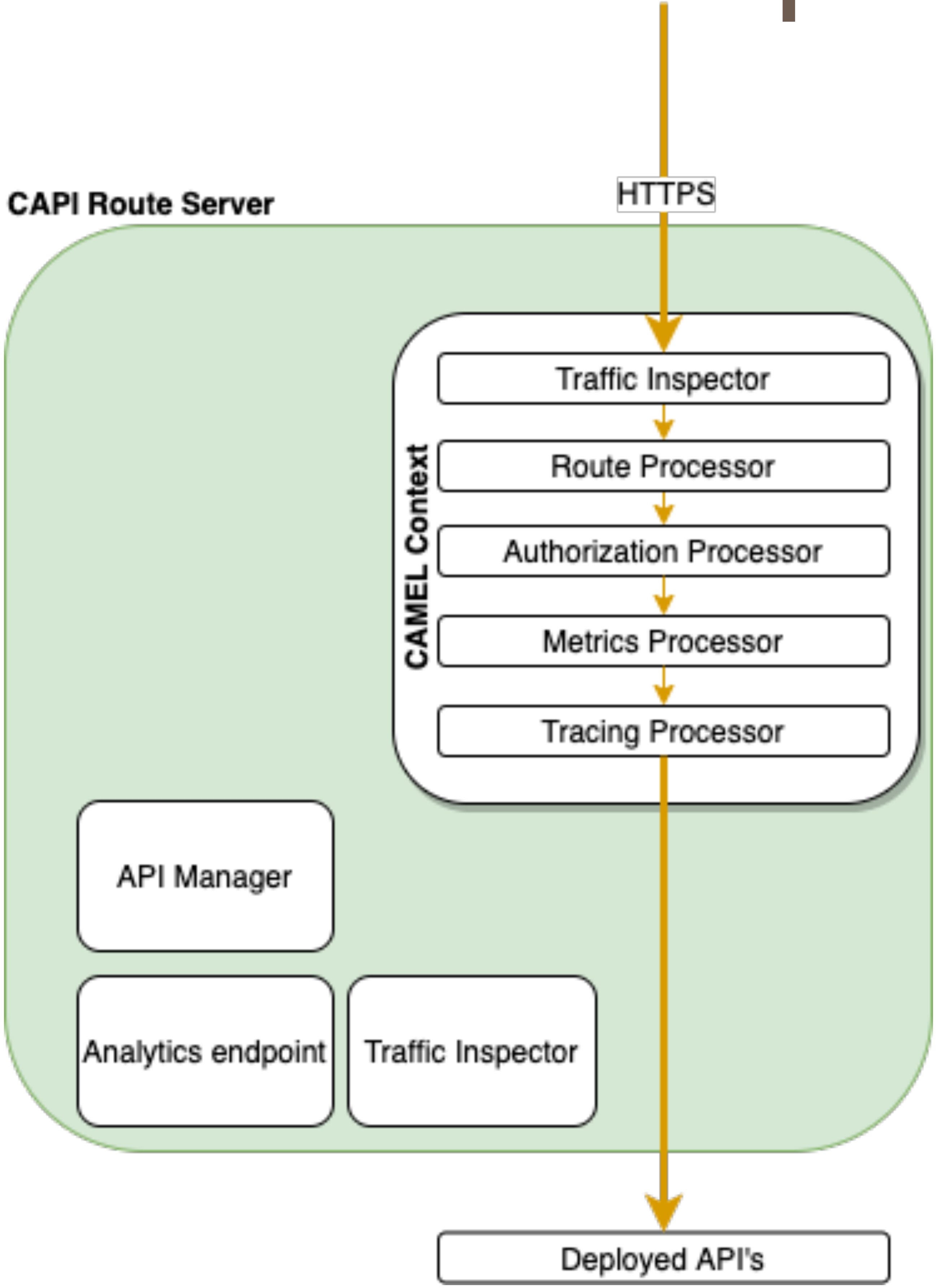
CAPI Gateway



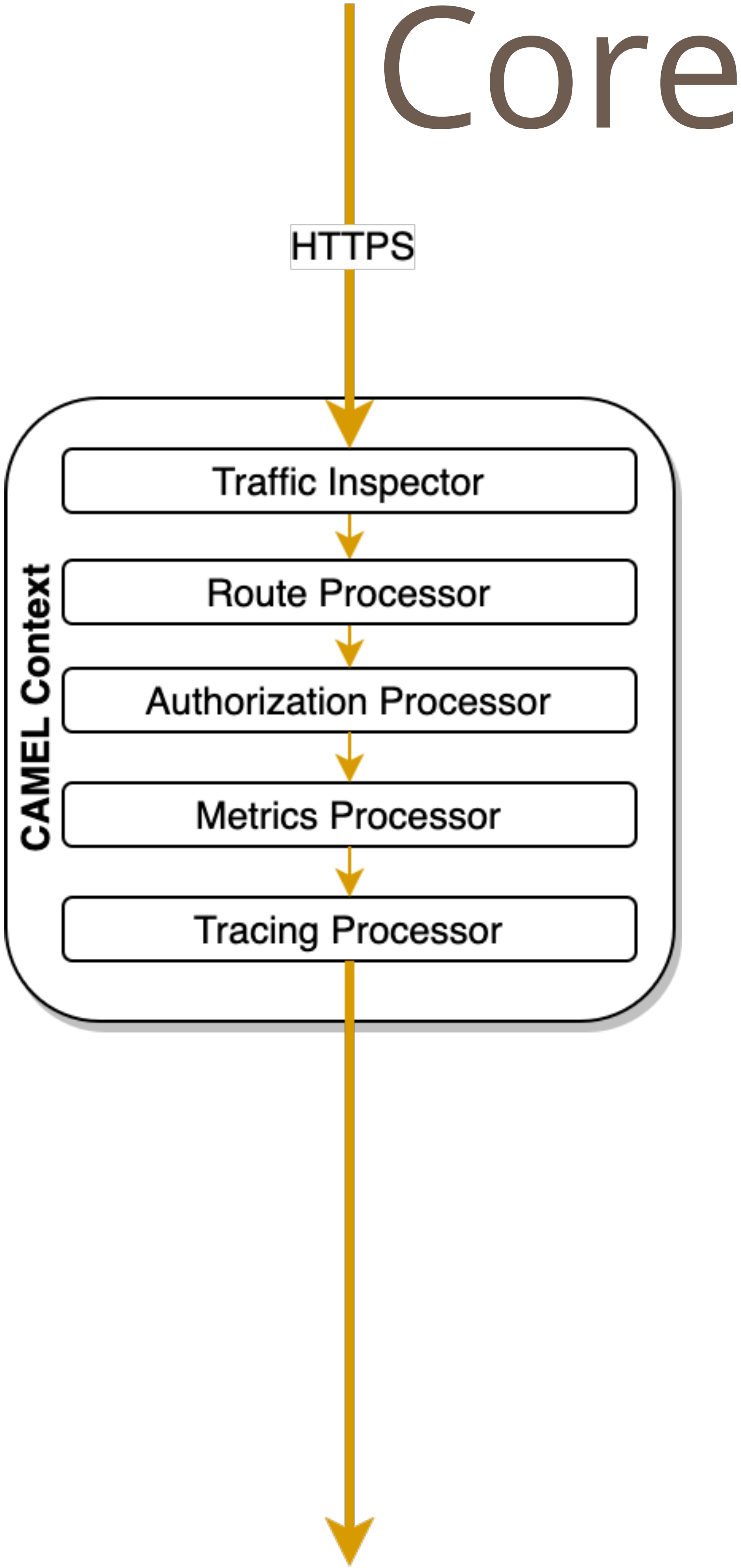
Core Concepts

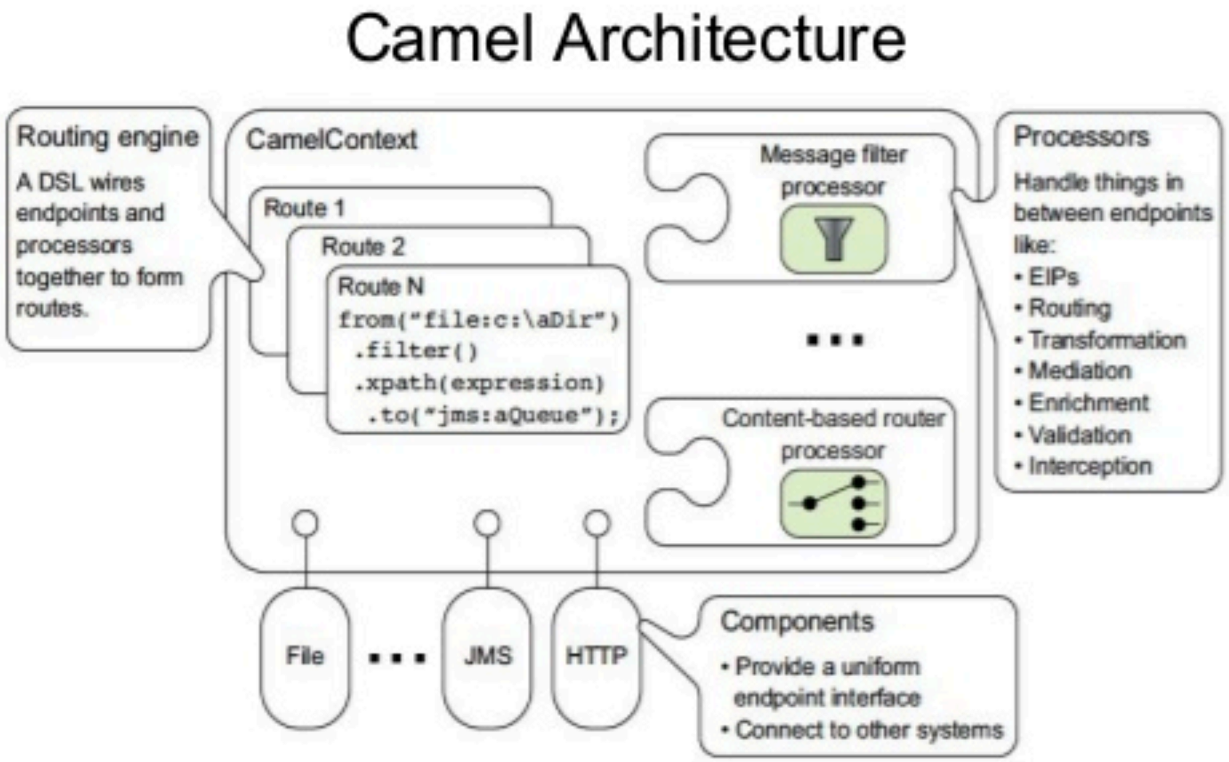


Core Concepts



Core Concepts





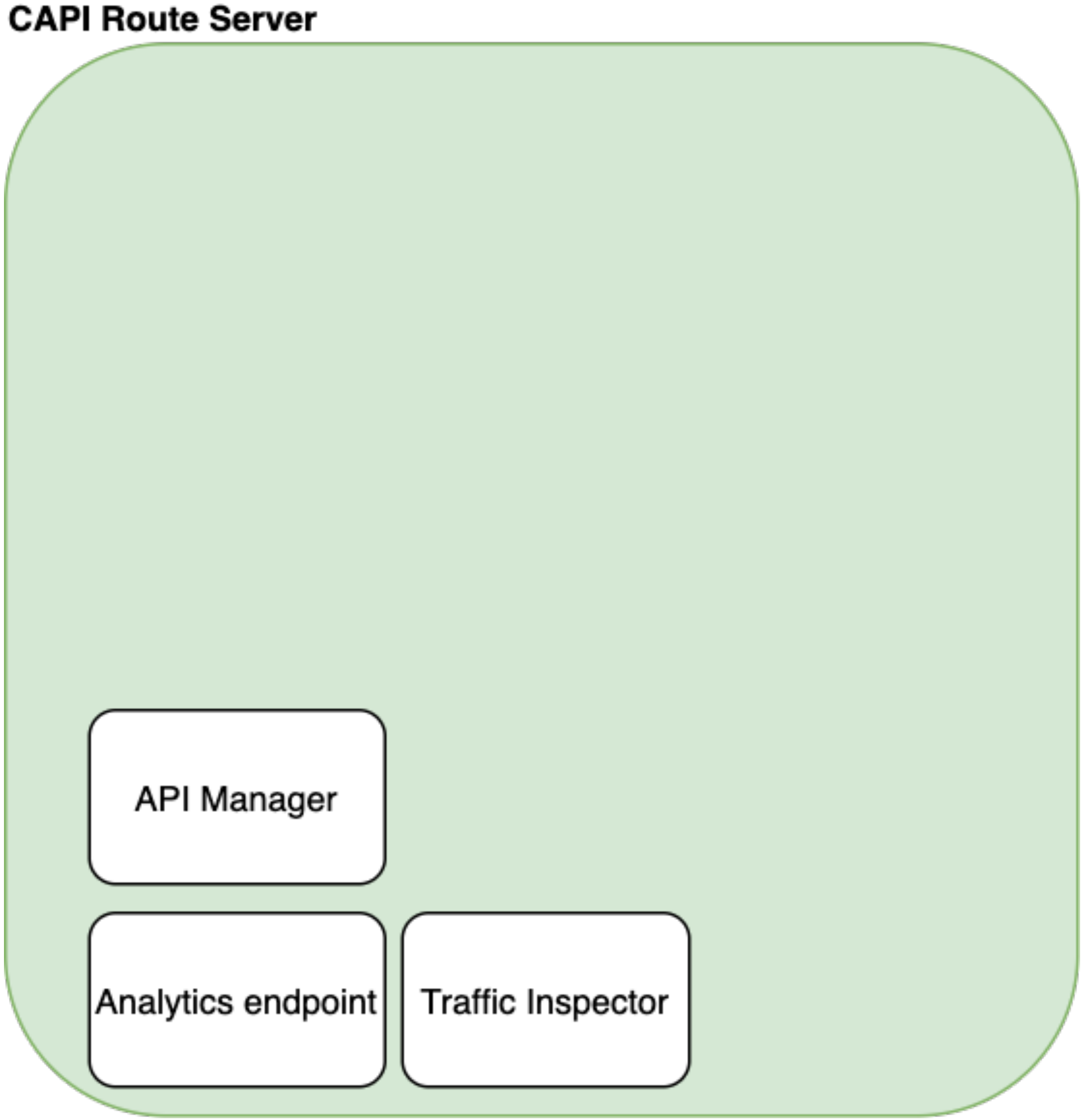
A DSL wires endpoints and processors together to form routes.

```
from("file:c:\aDir")
 .filter()
 .xpath(expression)
  .to("jms:aQueue");
```

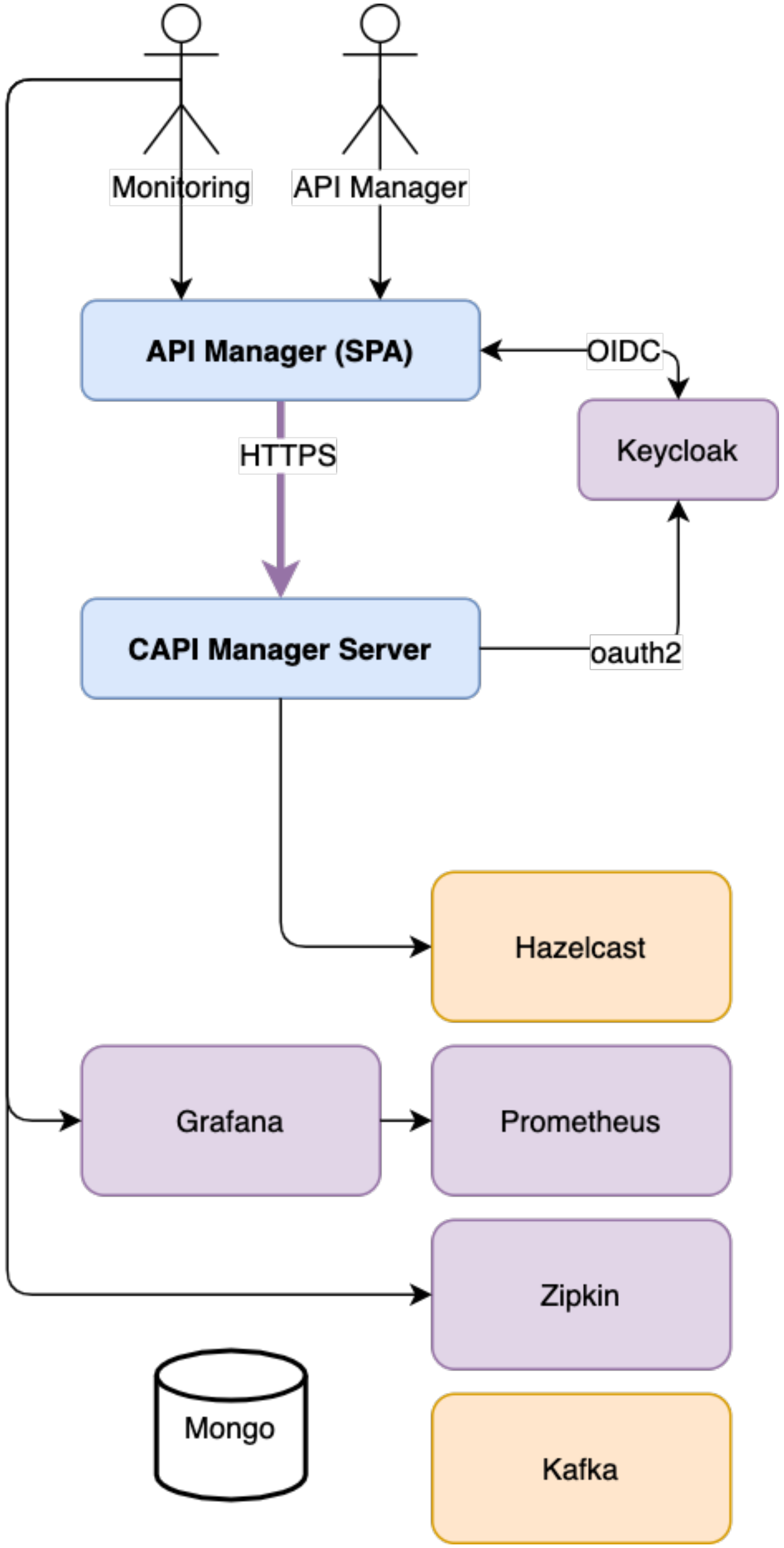
Handle things in between endpoints like:
- EIPs
- Routing
- Transformation
- Mediation
- Enrichment
- Validation
- Interception

- Provide a uniform endpoint interface
- Connect to other systems

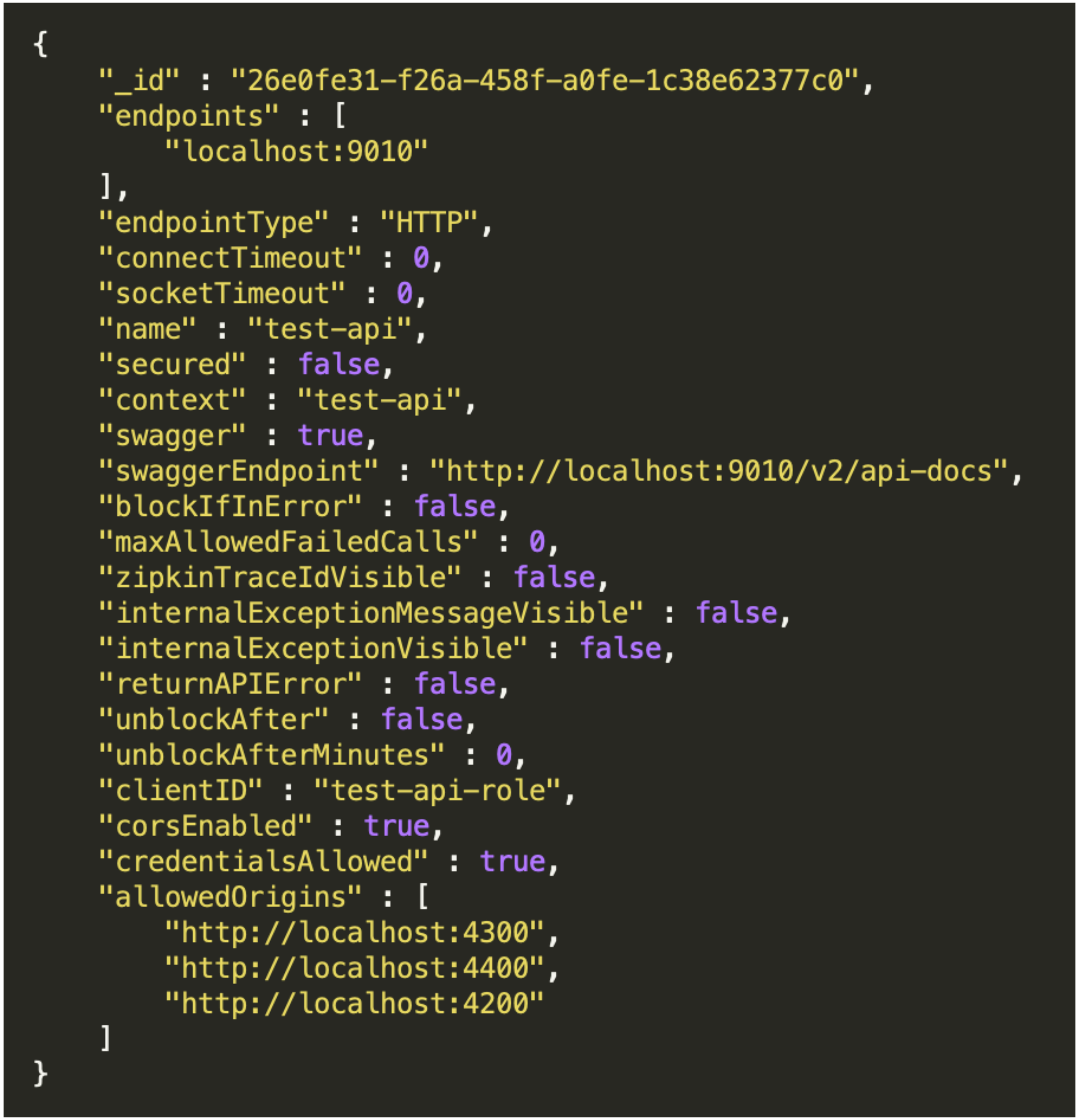
Core Concepts



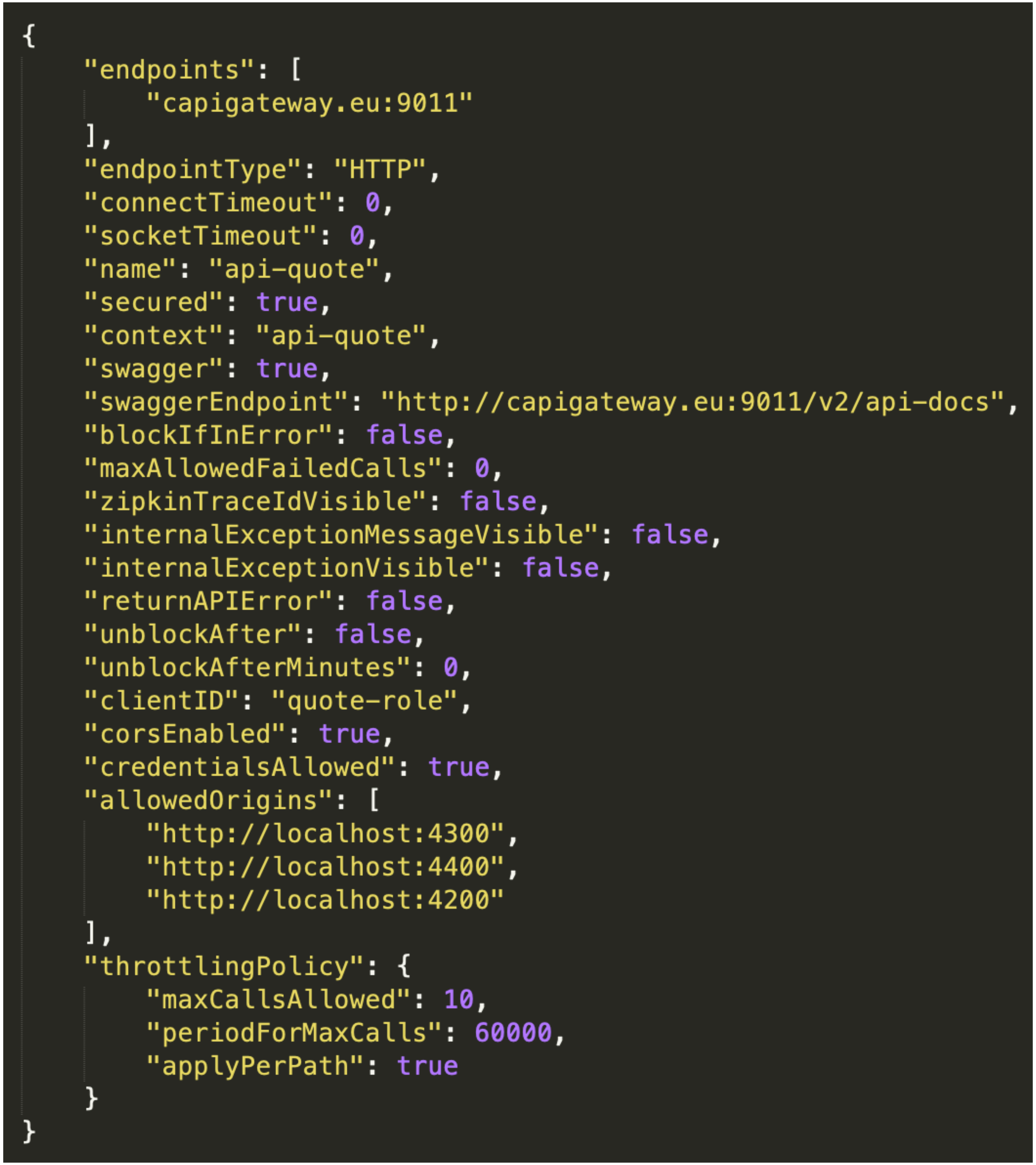
Core Concepts



API Model



```
{
    "_id" : "26e0fe31-f26a-458f-a0fe-1c38e62377c0",
    "endpoints" : [
        "localhost:9010"
    ],
    "endpointType" : "HTTP",
    "connectTimeout" : 0,
    "socketTimeout" : 0,
    "name" : "test-api",
    "secured" : false,
    "context" : "test-api",
    "swagger" : true,
    "swaggerEndpoint" : "http://localhost:9010/v2/api-docs",
    "blockIfInError" : false,
    "maxAllowedFailedCalls" : 0,
    "zipkinTraceIdVisible" : false,
    "internalExceptionMessageVisible" : false,
    "internalExceptionVisible" : false,
    "returnAPIError" : false,
    "unblockAfter" : false,
    "unblockAfterMinutes" : 0,
    "clientID" : "test-api-role",
    "corsEnabled" : true,
    "credentialsAllowed" : true,
    "allowedOrigins" : [
        "http://localhost:4300",
        "http://localhost:4400",
        "http://localhost:4200"
    ]
}
```



```
{
    "endpoints": [
        "capigateway.eu:9011"
    ],
    "endpointType": "HTTP",
    "connectTimeout": 0,
    "socketTimeout": 0,
    "name": "api-quote",
    "secured": true,
    "context": "api-quote",
    "swagger": true,
    "swaggerEndpoint": "http://capigateway.eu:9011/v2/api-docs",
    "blockIfInError": false,
    "maxAllowedFailedCalls": 0,
    "zipkinTraceIdVisible": false,
    "internalExceptionMessageVisible": false,
    "internalExceptionVisible": false,
    "returnAPIError": false,
    "unblockAfter": false,
    "unblockAfterMinutes": 0,
    "clientID": "quote-role",
    "corsEnabled": true,
    "credentialsAllowed": true,
    "allowedOrigins": [
        "http://localhost:4300",
        "http://localhost:4400",
        "http://localhost:4200"
    ],
    "throttlingPolicy": {
        "maxCallsAllowed": 10,
        "periodForMaxCalls": 60000,
        "applyPerPath": true
    }
}
```

Demo!

- Start 2 remote services
- Request the first access token
- Deploy the first: License Service (unprotected)
- Call the service **without** a bearer token
- Deploy the second: Quote Service (protected) (throttled)
- Call the service **without** a bearer token
- Request an access token
- Call the service **with** a bearer token

Demo!

- Create the role and add the role to the client ID
- Request a new access token
- Call the service **with** a bearer token
- Call the service more than 10X minute
- Observe the API being blocked
- Check Zipkin
- Check Prometheus
- Check Grafana

?