ESILV
ENGINEERING SCHOOL
DE VINCI PARIS

IoT -- Cloud

Thème 6 : API management et exemples

Jérémy GOLDSCHILD, Arnaud MAGARIAN, Nathan BOON, Hugo STEPHAN, Maël BALLET

ESILV
ENGINEERING SCHOOL
DE VINCI PARIS

Plan du cours

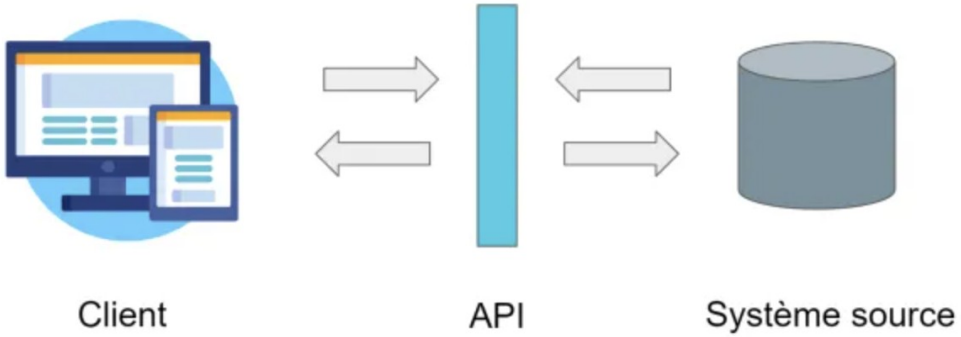
1. Introduction
2. Intérêt des API
3. Gestion d'accès à une API
4. Principe de token/clé, révocation, authentification
5. Orchestration et la répartition de charge
6. Différents acteurs et options
7. Monitoring
8. Conclusion

Qu'est-ce qu'une API?

- Une API est un ensemble de définitions et de protocoles qui facilite la création et l'intégration de logiciels d'applications.
- API est un acronyme anglais qui signifie « Application Programming Interface », que l'on traduit par interface de programmation d’application.
- C’est un facilitateur d’échange.
- Elle permet d’intégrer des services tierces ou de communiquer avec des bases de données.

Intérêt des APIs

L'API se situe en amont d’un système quel qu'il soit, et c’est elle que l’on va interroger lorsqu'on veut accéder à des informations et non plus au système directement.



Elle permet de pouvoir interagir avec un système sans se soucier de sa complexité et de son fonctionnement. Une Api est "*user friendly*" et est bien documentée.

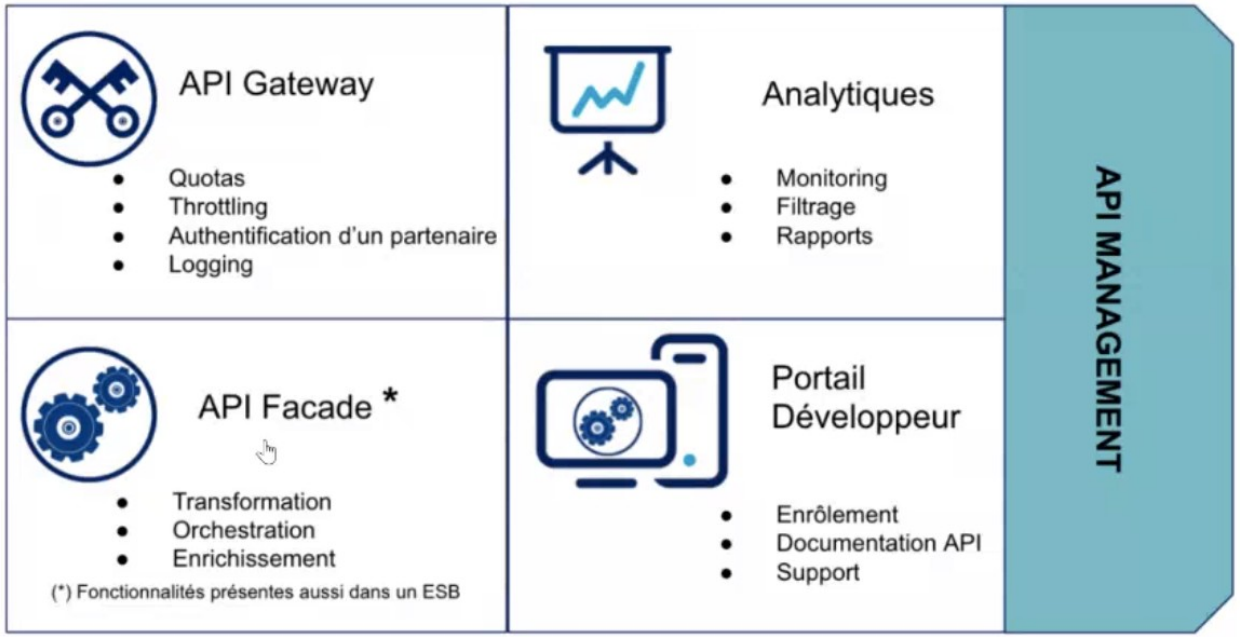
Une API est souvent spécialisée dans un domaine et sur un "*use case*" particulier ce qui simplifie son utilisation, sa compréhension et sa sécurisation

Une API permet également de facilement mettre à disposition à des tiers, des fonctionnalités sans ouvrir l’accès aux systèmes sources. Il est beaucoup plus simple de sécuriser l’accès à une api qu’à tout son système interne.

Principe du management d'API

- Discipline qui consiste à exploiter au mieux les API ans mettre en péril le système d'information et sans affecter l'expérience utilisateur

- L'objectif étant d'avoir une vision macroscopique de toutes les API



§ Centralisation du contrôle des API

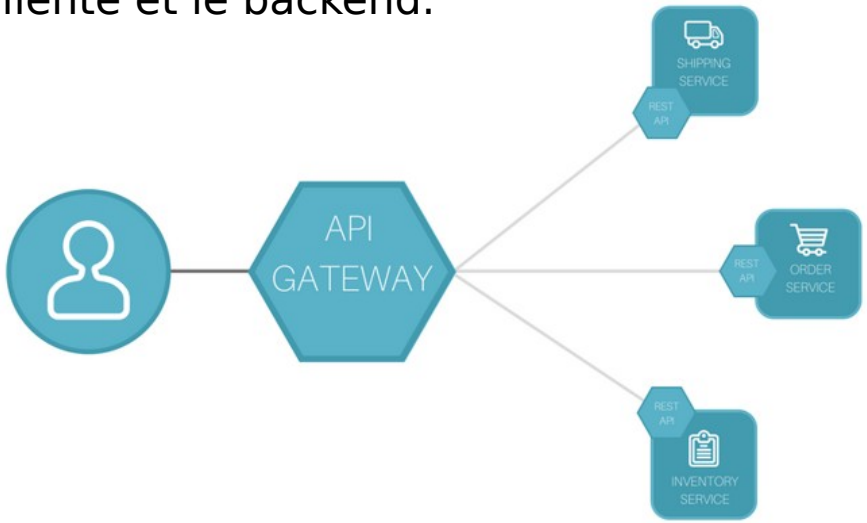
- Contrôle d'accès et sécurité
- Analyse des données
- Contrôle de la consommation
- Monétisation
- Gérer le cycle de vie des API
- Workflow de dev.

§ Harmonisation des services

API Gateway

- L'API Gateway est l'unique point d'entrée pour les API et les micro-services et redistribue les requêtes vers les différents micro-services

- Sans API Gateway, les applications doivent connaître précisément tous les services pour pouvoir les appeler. Cela augmente potentiellement la latence réseau en obligeant la multiplication des aller-retours entre l'application cliente et le backend.



§ Sans API Gateway, les micro-services se retrouvent publiquement exposés augmentant le risque d'attaque.

§ Comme ces fonctionnalités doivent être prévues pour tous les micro-services, cela complexifie fortement l'implémentation et la maintenabilité sur le long terme.

§ Utiliser une API Gateway permettra de simplifier les évolutions d'architecture ou d'application puisqu'elle fait abstraction de l'implémentation réelle du ou des micro-services.

Gestion des accès à une API

API Gateway

L'API Gateway :

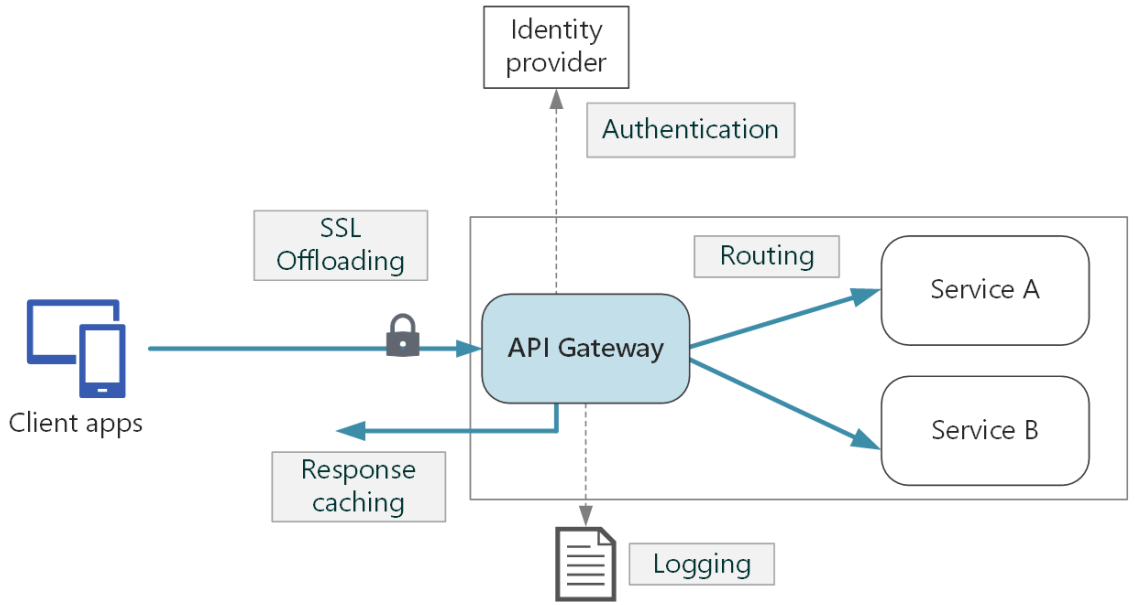
- Est "entre" l'application et l'API
- Route les requêtes vers le bon service
- Route les réponses vers la bonne application
- Enregistre toute l'activité de l'API. Toutes les requêtes et réponses passent par le Gateway.
- Contrôle les droits et accès

L'accès

L'application essayant de se connecter est-elle autorisée à se connecter à l'API ?

Les droits

A quelles services l'application effectuant la requête a-t-elle le droit de se connecter ?



https://docs.microsoft.com/en-us/azure/architecture/microservices/design/gateway

Gestion des accès à une API

Les limitations physiques

Notre API est stockée en physique sur des machines reliés à internet qui soulève deux questions :

1. Combien d'appel peut gérer notre backend par unité de temps ?
2. Combien d'opération peut gérer notre backend par unité de temps ?

Comment y répondre ?

- Quel est notre bande passante ascendante ? Descendante ?
- De quelle puissance de calcul disposons nous ?

Tous ces limites ainsi que toutes les méthodes de contrôle et limitation dont nous allons parler doivent être répertoriées dans un SLA (Service Level Agreement). Ce document, fournit par toutes les API, décrit les niveaux acceptables de disponibilité du service et de performance pour un utilisateur.

https://nevatech.com/blog/post/Service-Level-Agreements-in-API-Management-Basic-Setup-part-1

Pour plus d'informations sur le SLA :

Gestion des accès à une API

Gérer de l'accès à une API synonyme de limiter l'accès à une API.

Dans cette partie nous allons aborder :

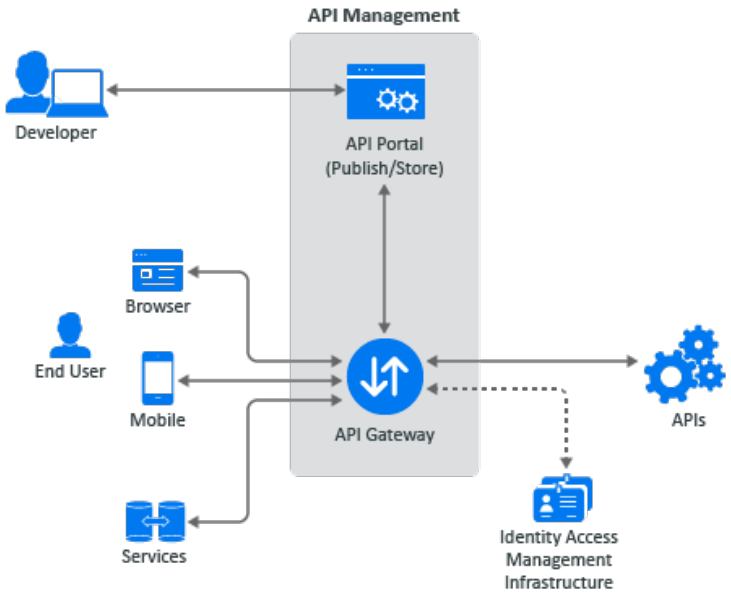
API rate limiting

API throttling

API burst

API quotas

API key



https://www.netiq.com/documentation/secure-api-manager-10/secure-api-manager-install/data/t4727csee7d5.html

API rate limiting / throttling

Envoie des requêtes :

- À un rythme limité
- En quantité limitée

Retour des réponses :
En quantité limitée

API burst

Donner à des clients des autorisations occasionnelles pour :

- Envoyer plus de requêtes
- Recevoir plus de réponses

Dans quels cas ?

1. Pour des clients qui ne peuvent prévoir le nombre d'appels qui vont être fait sur l'API. Dans le cas d'un pique sur l'application du client, l'API doit permettre à ce dernier de toujours fonctionner.

2. Lorsque les systèmes sont au ralenti et que les ressources n'ont pas besoin d'être réparties entre les différents utilisateurs.

API quotas

Méthode la plus connue, elle est très utilisée dans le cadre d'API à but commercial

Ces quotas définis par utilisateur limitent :

- Le nombre de requêtes par unité de temps autorisé
- La puissance de calcul allouée
- Le nombre de réponses ou bien la bande passante descendante allouée

Ces quotas peuvent être définis par secondes, jours ou mois et par application, utilisateur ou organisation.
Les fournisseurs d'API proposent généralement différents plans aux clients que ces derniers choisissent fonction de leurs besoins.

Différence API rate limiting et API quotas ?

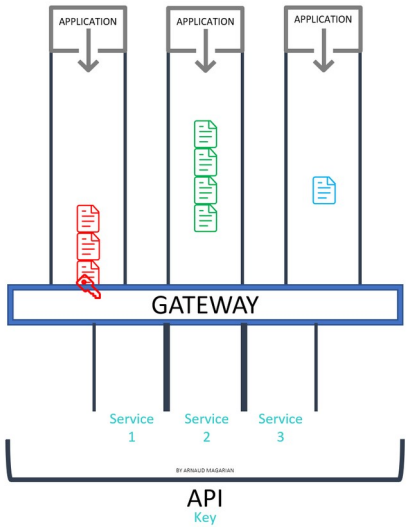
Dans le premier cas il s'agit d'une répartition optimisée ou 100% des ressources sont réparties entre 100% des utilisateurs
Dans le second cas, la répartition est faite par les gestionnaires de l'API en fonction des différents plans que ces derniers souhaitent offrir.
L'API throttling s'applique dans les deux cas.

API key

Dans presque 100% des API (gratuites ou non) l’accès est contrôlé par des API Key. Ces grâce à ces clés que l’API Gateway « reconnait » l’application et « sait » quels sont les services souscrits et les limitations

Il y a les clés :

- Application
- Adr



Sources et plus d'informations sur :

Le Gateway :

- Lien 1

Le Service Level Agreement (SLA) :

- Lien 2
- Lien 3

Les stratégies et techniques de limitations des API :

- Lien 4

L'API throttling et les quotas :

- Lien 5

Les Limites et quotas :

- Lien 6

Gestion d'API :

- Lien 7

Un peu de tout :

- Lien 8

Principe AAA

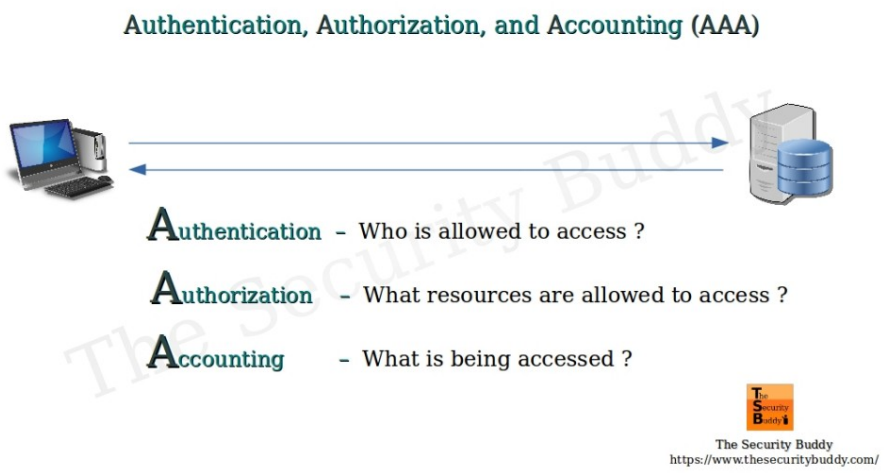
Principe AAA (Authentification, Authorization, Accounting)

- Authentification – Qui appelle notre API?
  Utilisation d’un token d’accès/clé api dans l’header.
  Nom d’utilisateur/Mot de passe encrypté avec hashage.

- Autorisation – Quels droits?
  Une fois authentifié qu’est-ce que la personne peut faire dans vo
  A-t-elle le droit de consommer cette ressource en particulier?

- Traçabilité – Qui Quoi Quand?
  Savoir qui a fait quoi, quand et avec quelles ressources?
  Pour chaque appel HTTP il faut savoir.
  Quoi? La ressource consommée, le verbe HTTP appelé, le code HTTP répondu.
  Qui? L’utilisateur final à l’origine de l’appel, l’application qu’utilise cet utilisateur.
  Quand? Le moment où a été exécutée la requête.

- Révocation
  Une telle traçabilité de vos données pourra vous permettre une révocation d’accès,



Plus d'infos

Authentification – Clé API

Trois types d’authentification API

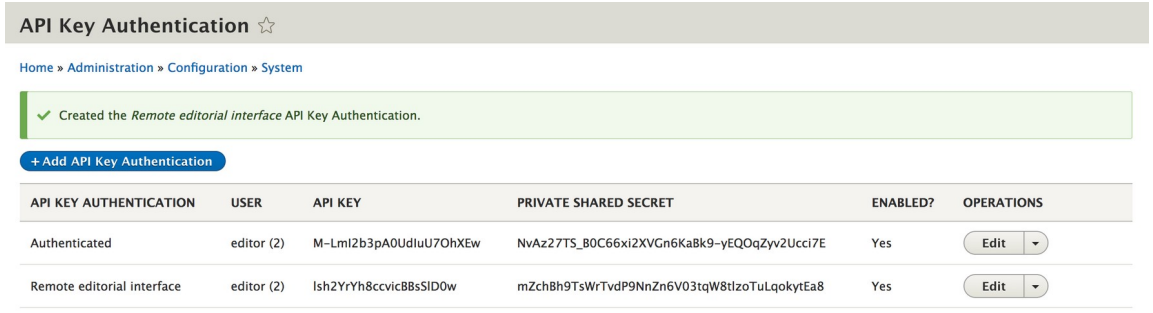
- API Key
- Oauth Tokens
- Json Web Tokens (JWT)

API Key

Une clé API est un identifiant unique qui authentifie un utilisateur d’API
Elles permettent à une application d’accéder à une ressource (API).
Il s’agit du mécanisme le plus simple de sécurité d’API.

Cependant, la simplicité s’accompagne à la fois d’inconvénients de sécurité et d’expérience utilisateur.

- La clé API peut être compromise (si elle est stockée dans une application publique)
- Elle permet d’identifier que l’application cliente, et non pas un utilisateur précis.
- Elle n’a pas de date d’expiration



Authentification/Autorisation Oauth Tokens

Oauth Tokens

Les standards de l'industrie pour l'authentification et l'autorisation sont OAuth2 et OpenID Connect : OAuth2 est dédié à la délégation de droits ("moi, utilisateur, autorise cette application à lire mes mails en mon nom") et OIDC, à la gestion d'identités. Comparer à la clé d'API les Oauth tokens peuvent d'expirer, le refresh token permet de récupérer un nouveau token d'accès lorsque celui-ci vient à expiration.

Access token: Comme une Clé, il permet à l'application d'accéder aux données d'une API, il peut expirer (Optionnel)
Refresh token: Optionnel, Tokens de refresh récupère un access token lorsqu'il a expiré.

Authentification JWT

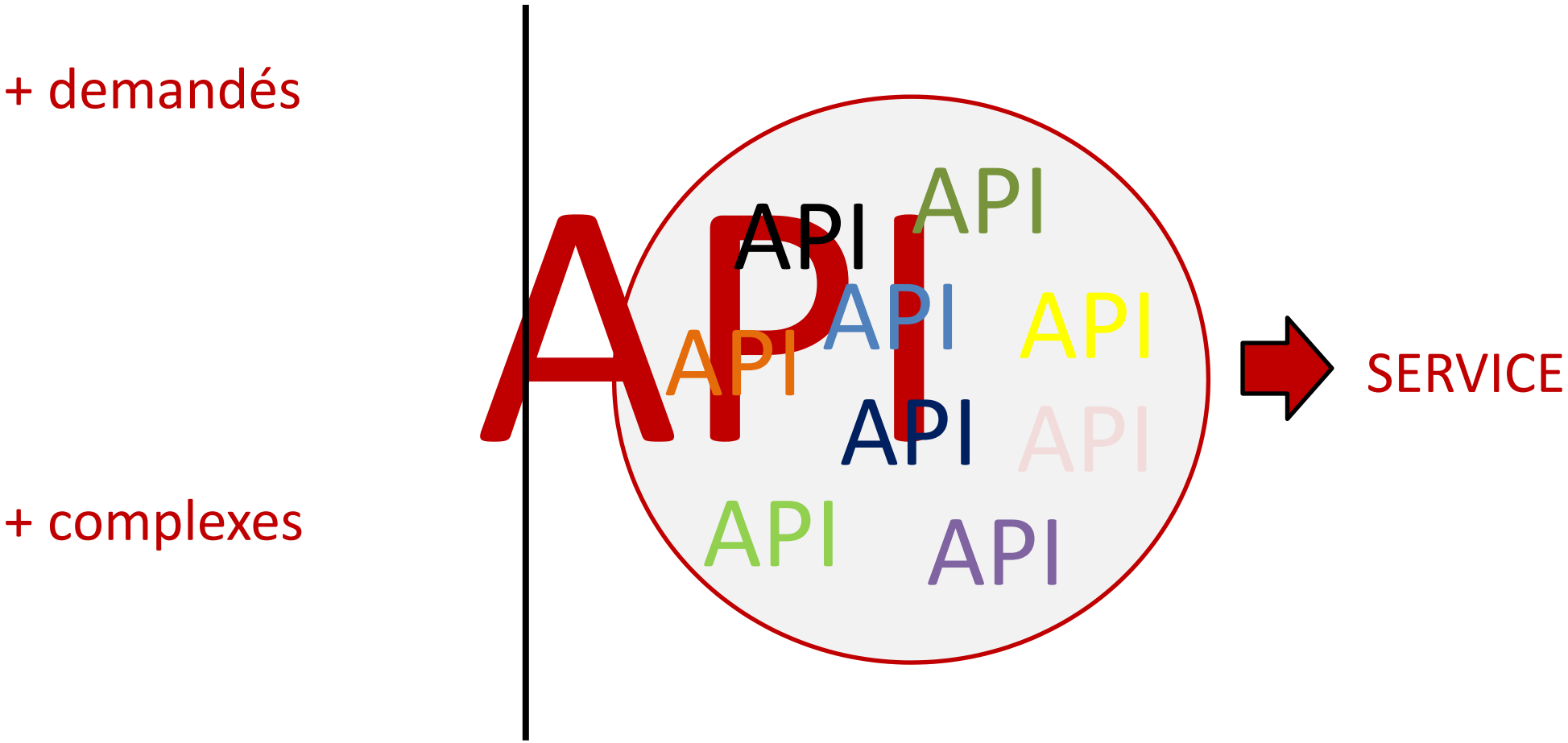
Json Web Tokens (JWT)

Alors que les clés API et les jetons OAuth sont toujours utilisés pour accéder aux API, les jetons Web JSON (JWT) peuvent être utilisés dans de nombreux scénarios différents.

En fait, JWT peut stocker tout type de données, c'est là où il excelle en combinaison avec OAuth. Avec un jeton d'accès JWT, beaucoup moins d’accès à la base de données sont nécessaires sans pour autant compromettre la sécurité.

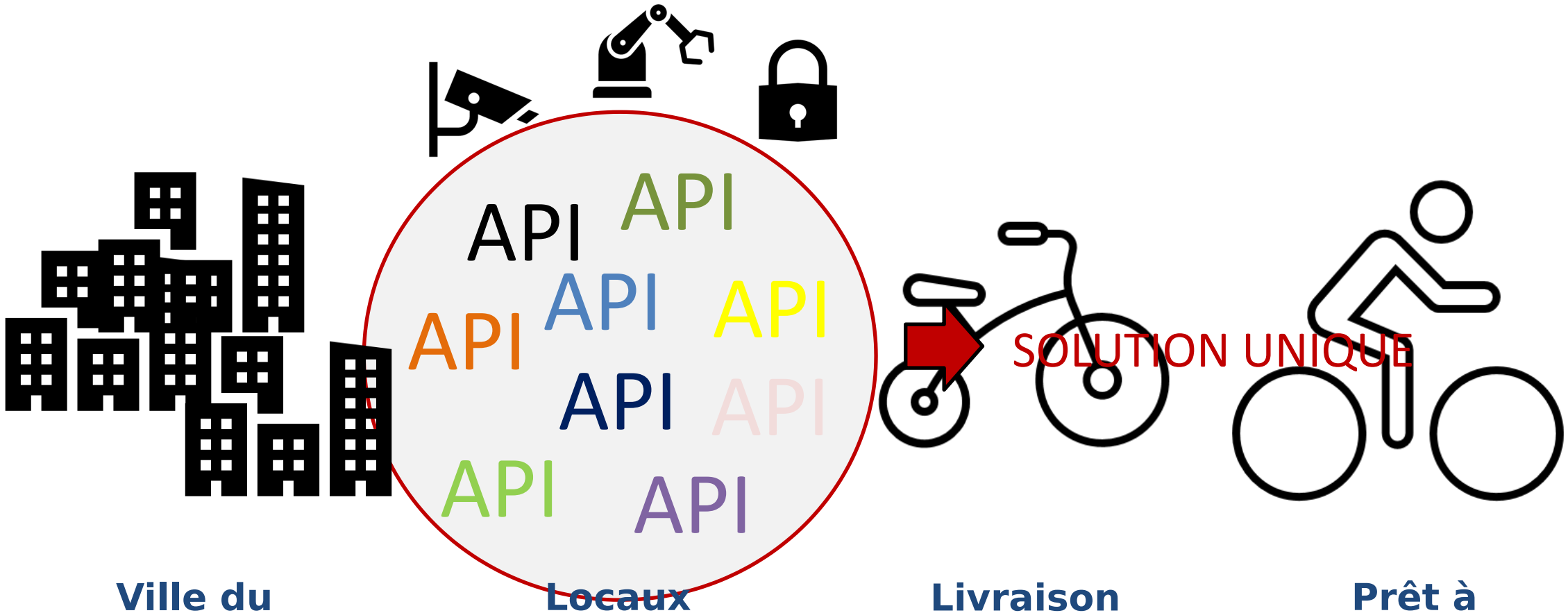
Plus d'infos Key/Token/JWT

Orchestration et repartition de charge



Orchestration

Qu’est-ce ? Comment ça marche ?



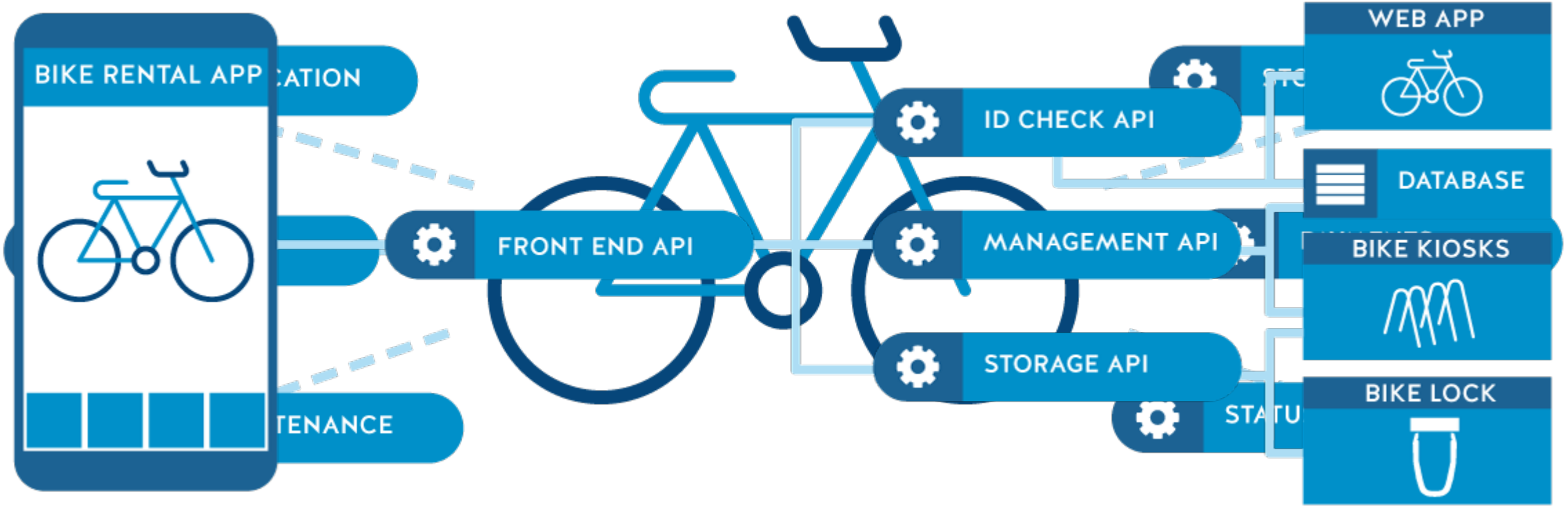
Ville du futur

Locaux dédiés

Livraison autonome

Prêt à rouler

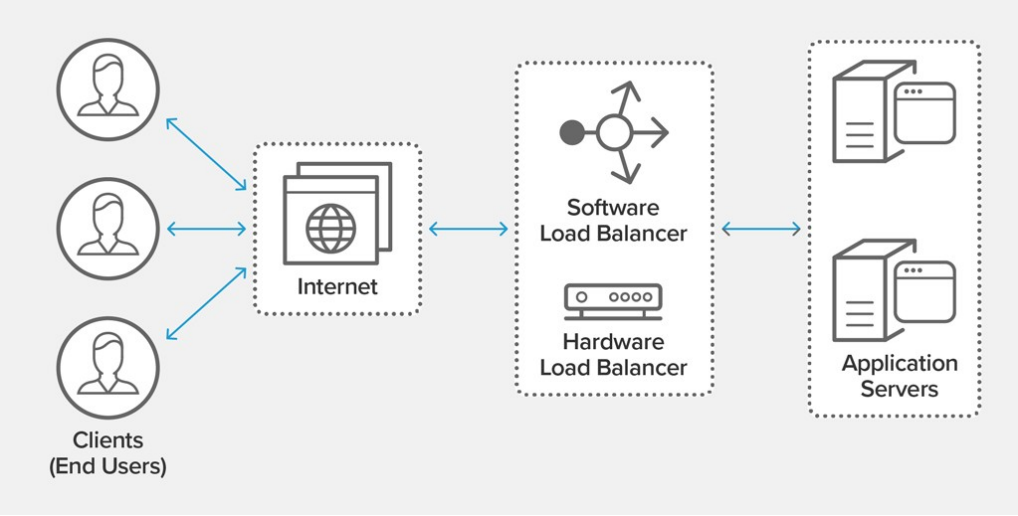
Orchestration

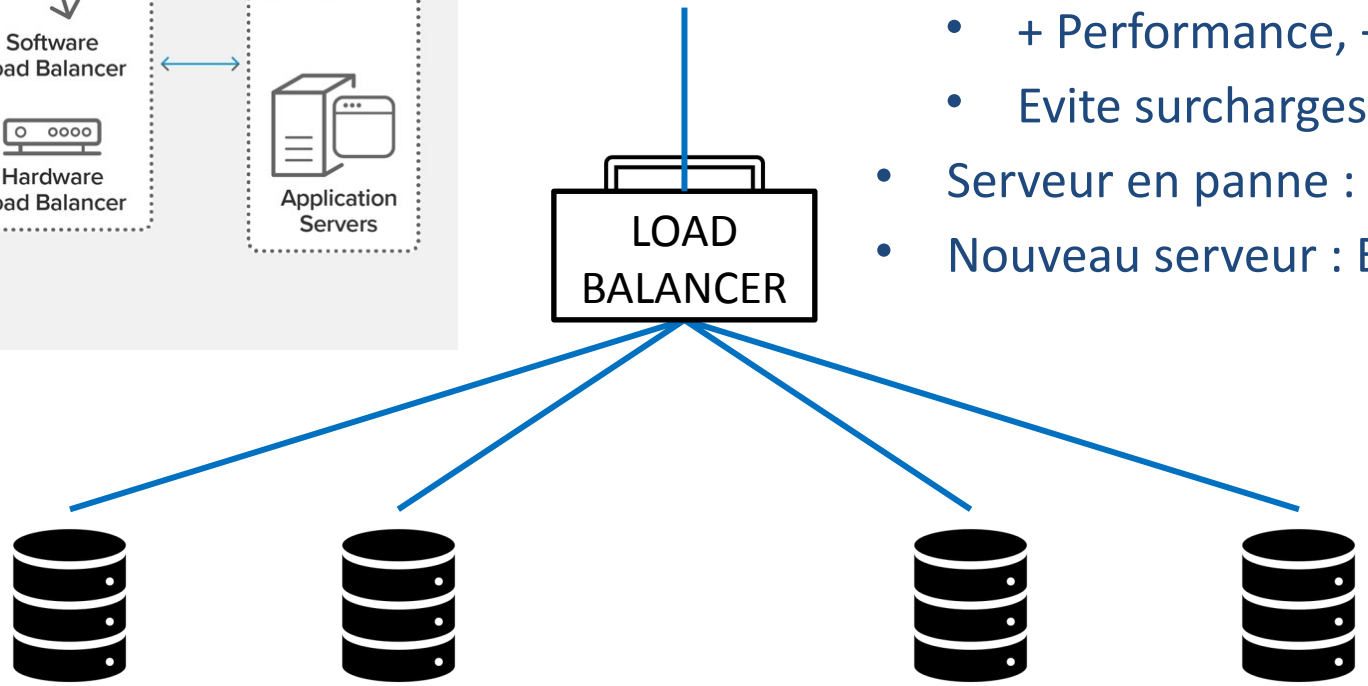


- **FEDERATION :** Création d'un token de confiance, évite les étapes systématiques d'identification

Répartition de charge

Qu’est-ce que le Load Balancing ?





- Surveille serveurs
- Redirige requêtes aux serveur aptes
  - + Performance, + Vitesse
  - Evite surcharges
- Serveur en panne : Redirige trafic
- Nouveau serveur : Envoi de requêtes

Répartition de charge

Comment ça fonctionne ? Quels composants ?

```
+-------------------------------------------------------------+
|                   +--------------------+                    |
|                   |  Highly Available  |                    |
|                   |     IP address     |                    |
|                   +---------+--+-------+                    |
|                             |  |                            |
|                             |  |                            |
|                             |  |                            |
|                  traffic    |  |                            |
|                     +       |  |                            |
|               health checks |  | health checks              |
|                             |  |                            |
|       +-----------+         |  |      +-----------+         |
|       | Master LB |<--------+  +----->| Backup LB |         |
|       +-----+-----+                   +-----------+         |
|             |                                               |
|             |                                               |
|             |           traffic & health checks             |
|     +-------+------+--------------+--------------+          |
|     |              |              |              |          |
|     |              |              |              |          |
+-----|--------------|--------------|--------------|----------+
      |              |              |              |
      |              |              |              |
+-----v----+   +-----v----+   +-----v----+   +-----v----+
| Backend  |   | Backend  |   |          |   | Backend  |
| Server 1 |   | Server 2 |   |    ...   |   | Server N |
+----------+   +----------+   +----------+   +----------+
```

API Testing et API Monitoring

API testing pour la préproduction :

- Tests intégrés tout au long du processus de développement pour minimiser les risques.
- Valider que les performances et les fonctionnalités n'ont pas été affectées par les mises à jour.

API monitoring pour la post-production :

- Vérifier que les API sont fonctionnelles, sécurisées et rapides.
- Permet d'identifier et résoudre les pannes et les erreurs

Moniteurs API basiques :

Les moniteurs API basiques testent un seul appel d'API.

Moniteurs API multi-étapes :

La surveillance API en plusieurs étapes vérifie les interactions API complètes.

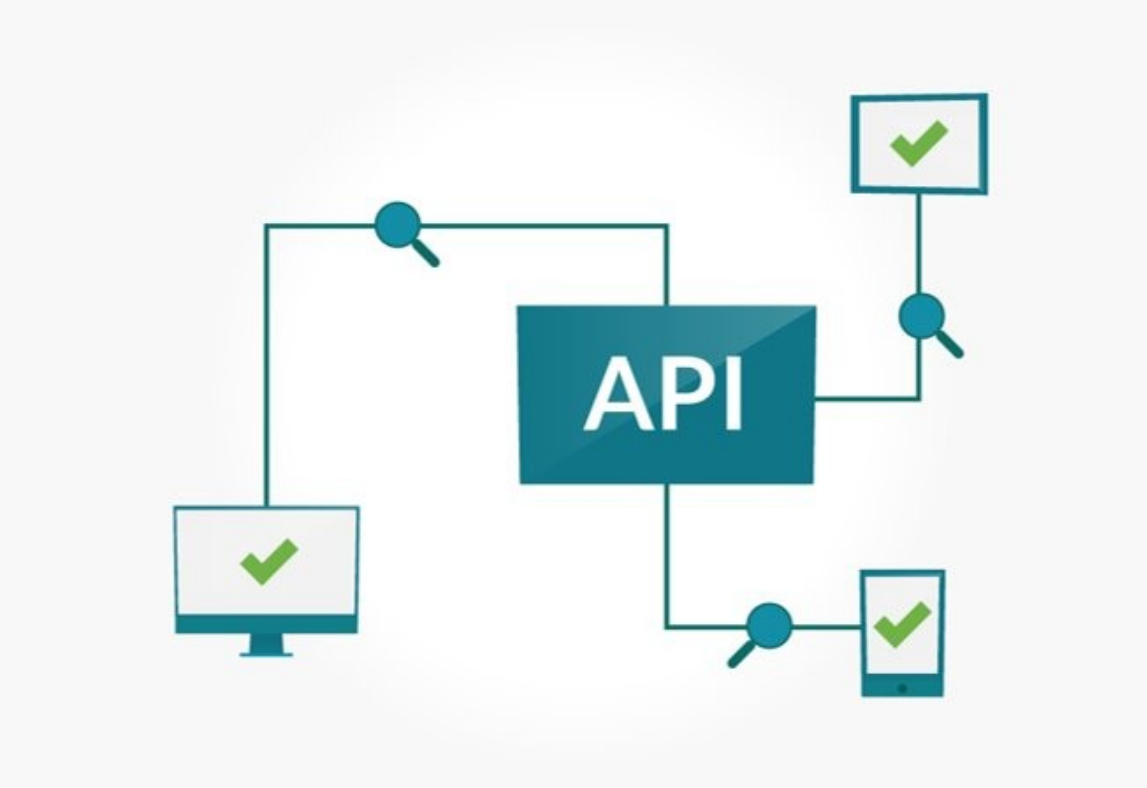
Pourquoi tester une API ?

Tester les jonctions et les jalons clés:

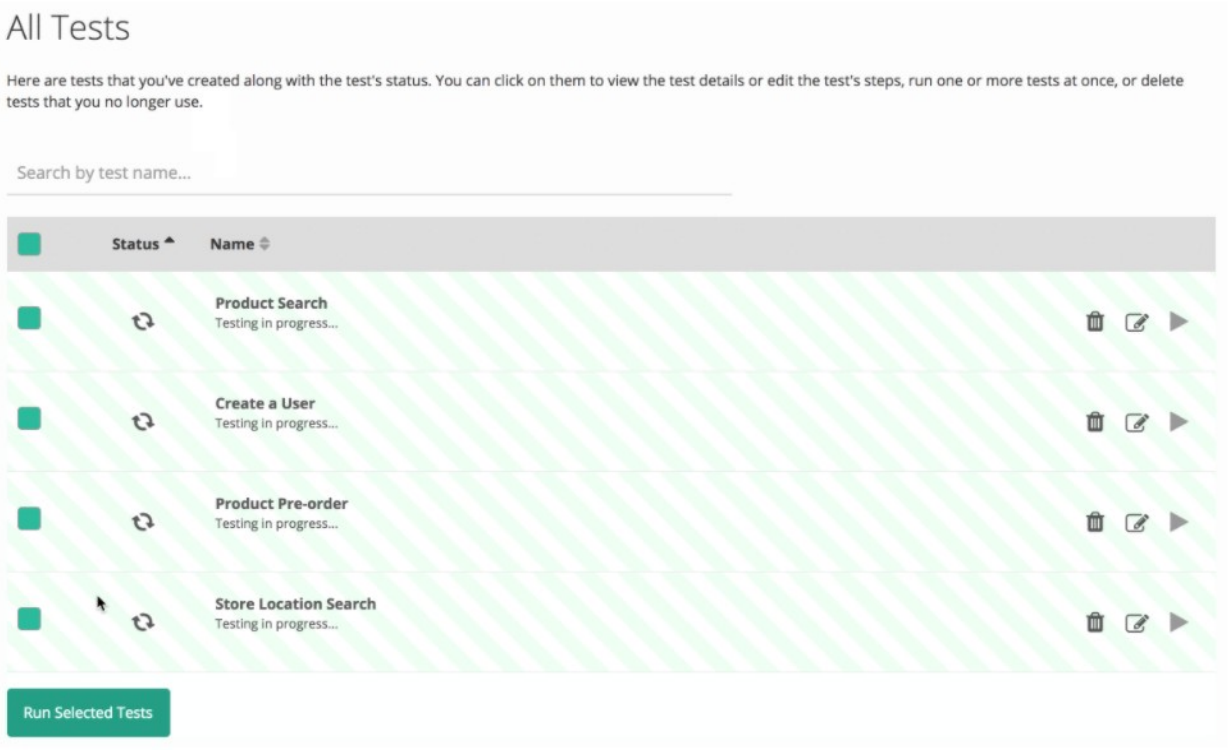
- Ne pas déployer de bugs lors de la mise en production
- Savoir immédiatement si quelque chose ne fonctionne pas.

Ce que l'on teste :

- Disponibilité : vérifier si les ressources sont disponibles ou non.
- Temps de réponse : à quelle vitesse l'API renvoie-t-elle les réponses ?
- Validation des données : assurer que les systèmes reçoivent ce dont ils ont besoin pour effectuer des tâches.
- Processus : enregistrement et réutilisation des variables d'une requête à l'autre, authentification.
- Tester les intégrations et modifications de fonctionnalités : assurer la compatibilité avec les



API Testing


All Tests
Here are tests that you've created along with the test's status. You can click on them to view the test details or edit the test's steps, run one or more tests at once, or delete tests that you no longer use.
Search by test name...
Status
Name
Product Search
Testing in progress...
Create a User
Testing in progress...
Product Pre-order
Testing in progress...
Store Location Search
Testing in progress...
Run Selected Tests

Status
Name
Test Passed
Product Search
Last run Today 1:07:32 PM UTC
PASSED
Test Passed
Create a User
Last run Today 1:07:32 PM UTC
PASSED
Test Failed
Product Pre-order
Last run Today 1:07:32 PM UTC
[Step 1] contact Pre-order API failed
Test Passed
Store Location Search
Last run Today 1:07:32 PM UTC
PASSED

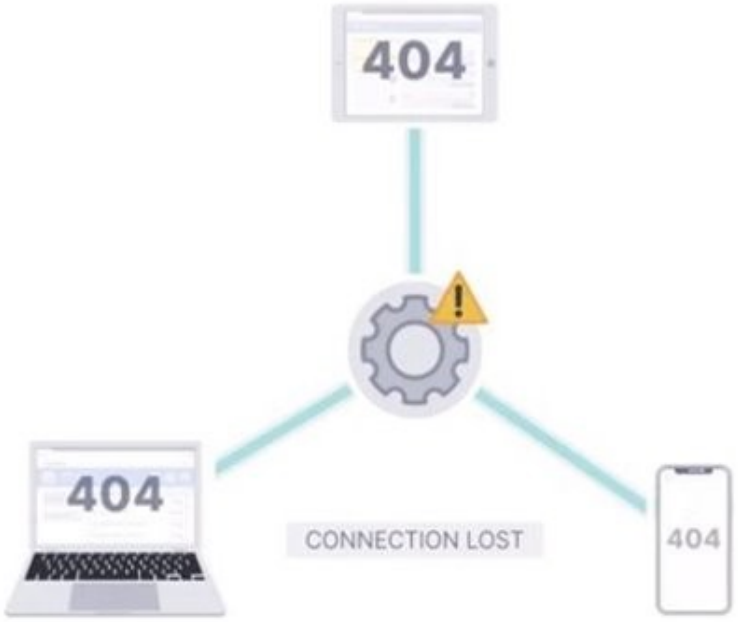
Pourquoi monitorer une API ?

- Assurer les performances
  Garantir les performances et la disponibilité continues des API.

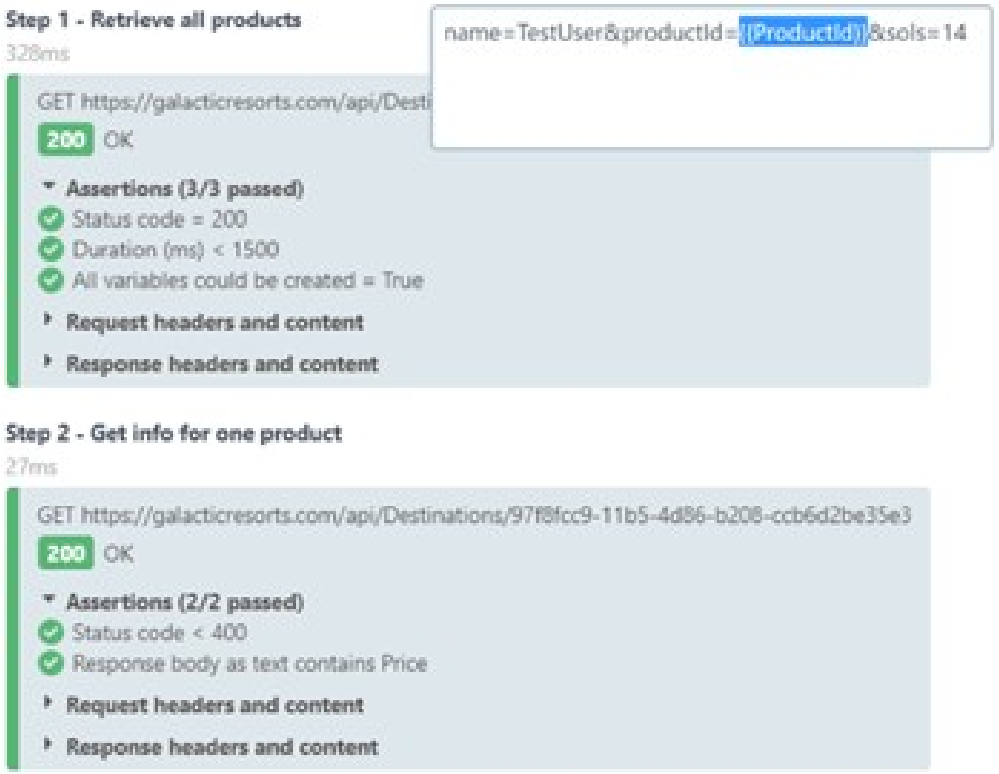
- Surveiller les performances
  Vérifier si les performances s'améliorent ou se dégradent.

Ce que l'on surveille :

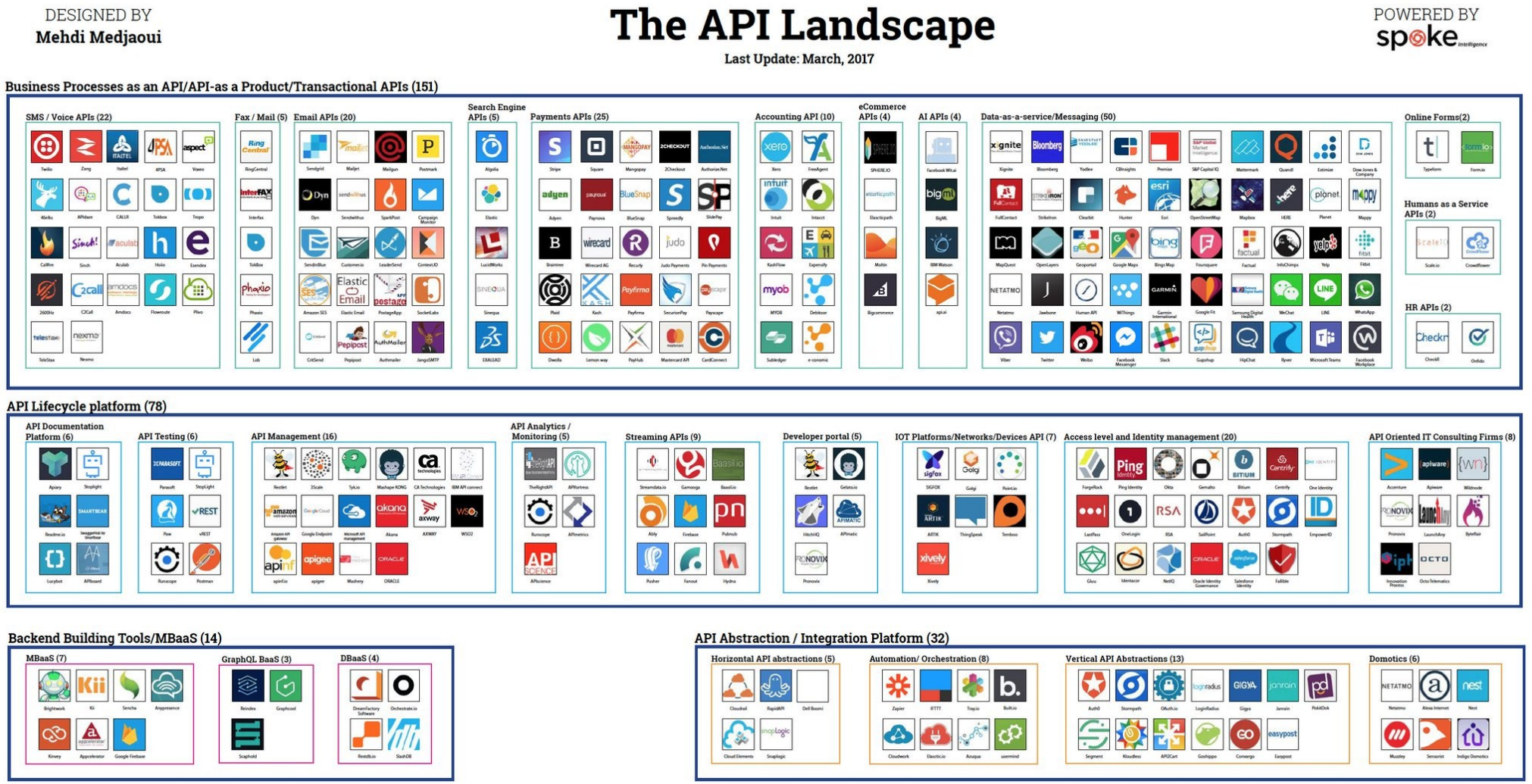
- Disponibilité : Vérifier que les ressources soient disponibles et ne renvoient pas d'erreurs à tout moment.
- Temps de réponse : comparer le temps de réponse par rapport à la préproduction et les variations.
- Validation des données : Vérifier que les données renvoyées soient correctes et au bon format.
- Processus



API Monitoring



API - Différents acteurs



https://seinecle.github.io/mk99/generated-html/apis-and-their-business-relevance-fr.html

ESILV
ENGINEERING SCHOOL
DE VINCI PARIS

IoT -- Cloud

Merci de votre attention !

Jérémy GOLDSCHILD, Arnaud MAGARIAN, Nathan BOON, Hugo STEPHAN, Maël BALLET