ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
Université du Québec

ÉTS
Engineering for Industry

Optimisation des performances des systèmes IdO basés sur les microservices grâce aux meilleures pratiques : revue et étude empirique

Présenté par :
Yahia El FELLAH

Sous la supervision de :
Prof. Naouel Moha
Prof. Julien Gascon-Samson
Prof. Yann-Gaël Guéhéneuc

Membre de jury:
Prof. Lokman Sboui
Prof. Ghizlane El Boussaidi

Soutenance de mémoire le
12 avril 2024

Plan de présentation

01 Contexte

02 Contribution 1 : revue de la littérature académique

03 Contribution 2 : revue de la littérature grise

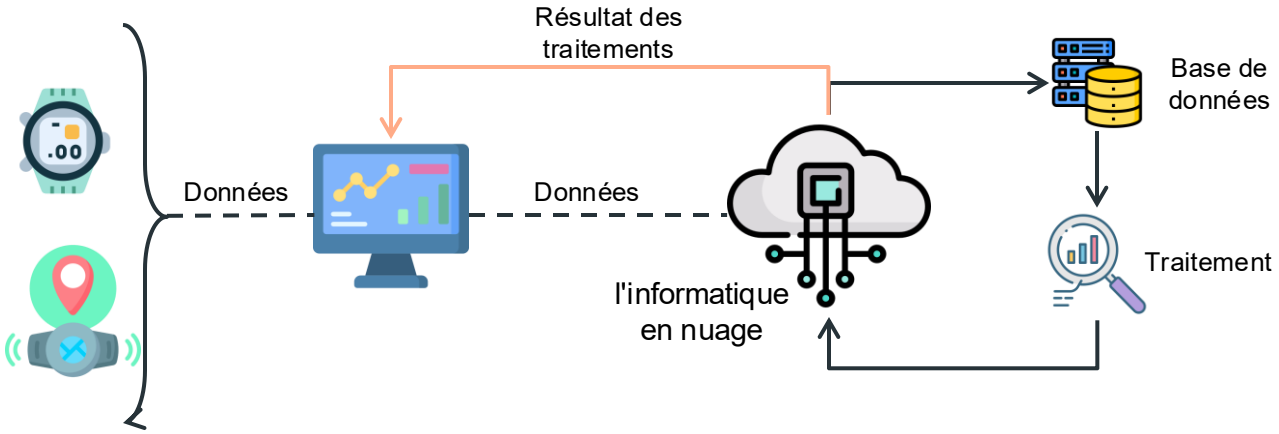
04 Contribution 3 : étude empirique

05 Menaces à la validité

06 Conclusion et travaux futurs

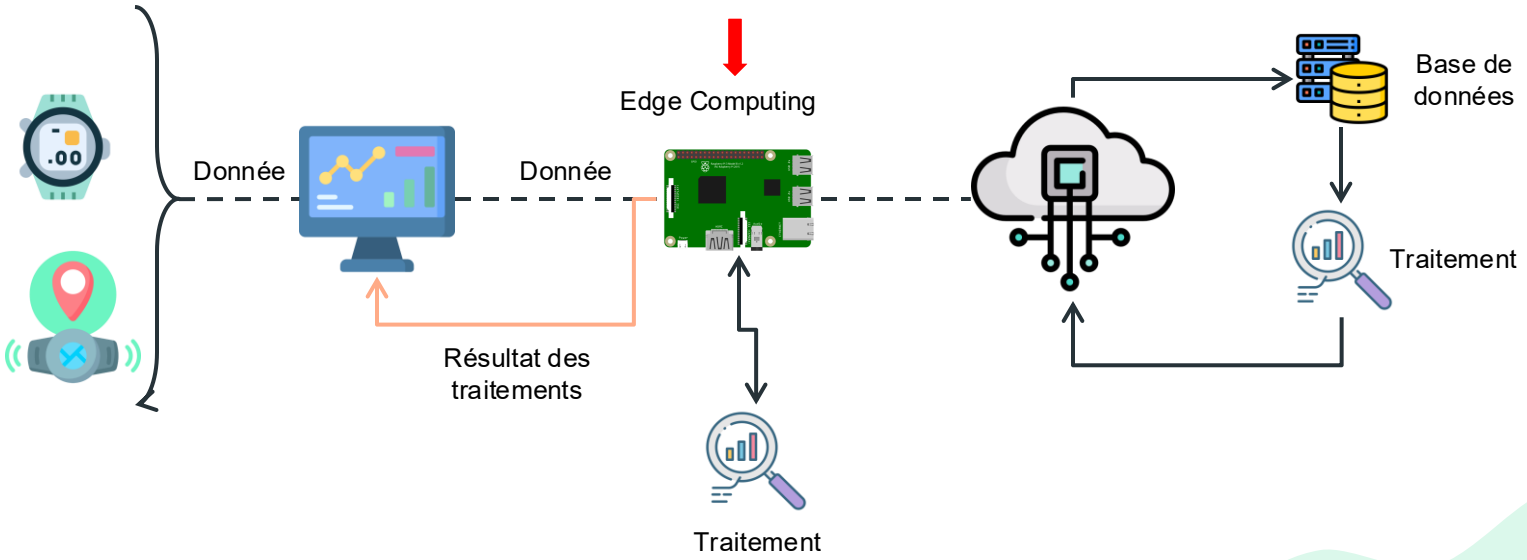
Contexte (1/4) : Internet des objets (IoT)

L'IoT est un concept dans lequel des dispositifs, des logiciels et des technologies de réseau sont interconnectés pour échanger des données avec divers appareils et systèmes via Internet

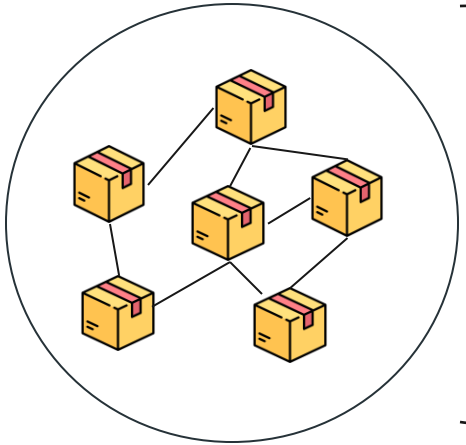


Contexte (2/4) : Edge Computing

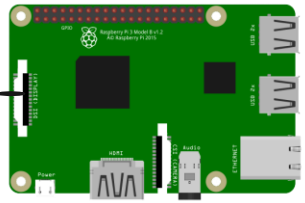
L'Edge Computing est une architecture informatique où le traitement des données est effectué aussi près que possible de la source des données



Contexte (3/4) : Microservice et IoT



Microservices



Environnement avec ressources limitées

Performance et optimisation des ressources

Est-ce qu'on peut adresser ce défi en utilisant les bonnes pratiques logicielles ?

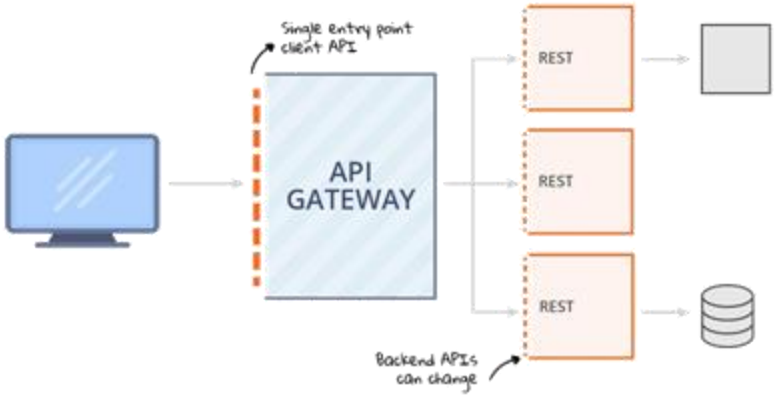
Exemple : Conteneurisation des microservices

Les conteneurs offrent plusieurs avantages pour les performances :

- **Utilisation efficace des ressources**
- **Portabilité**
- **Isolation légère…**

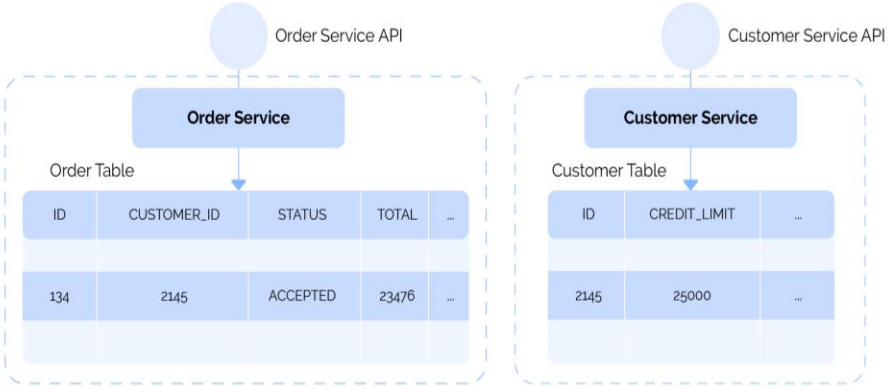
Contexte (4/4) : bonnes pratiques de génie logiciel

Passerelle d'API



Source : https://www.crmsoftwareblog.com/2021/06/how-to-make-an-api-gateway-work-to-your-advantage-as-a-crm-admin/

Base de données par service



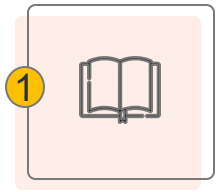
Source : https://www.sayonetech.com/blog/microservices-database-management-what-you-should-know/

Problématique

Il n'existe actuellement aucune étude démontrant l'efficacité des bonnes pratiques logicielles pour optimiser les performances et l'utilisation des ressources des systèmes IoT dans les environnements contraints du edge computing

Contributions

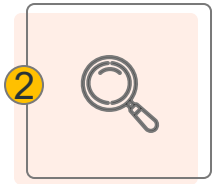
QR1 : Quels sont les défis rencontrés par les développeurs de systèmes IoT lors du déploiement de systèmes IoT basés sur les microservices ?



Revue de la littérature académique

Défis et approches

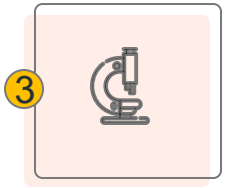
QR2 : Quelles sont les bonnes pratiques logicielles qui pourraient optimiser les performances et l'utilisation des ressources des systèmes IoT basés sur les microservices en périphérie ?



Revue de la littérature grise

Catalogue de pratiques

QR3: Quel est l'impact sur les performances des bonnes pratiques logicielles identifiées pour les systèmes IoT basés sur les microservices déployés en périphérie ?

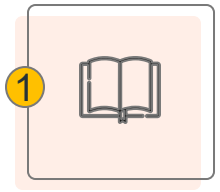


Étude empirique

Évaluation comparative des performances des systèmes

Contributions

QR1 : Quels sont les défis rencontrés par les développeurs de systèmes IoT lors du déploiement de systèmes IoT basés sur les microservices ?

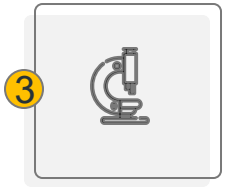


Revue de la littérature académique

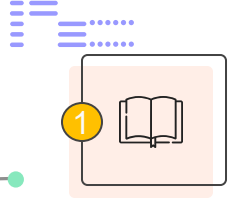
QR2

Revue de la littérature grise

QR3



Étude empirique



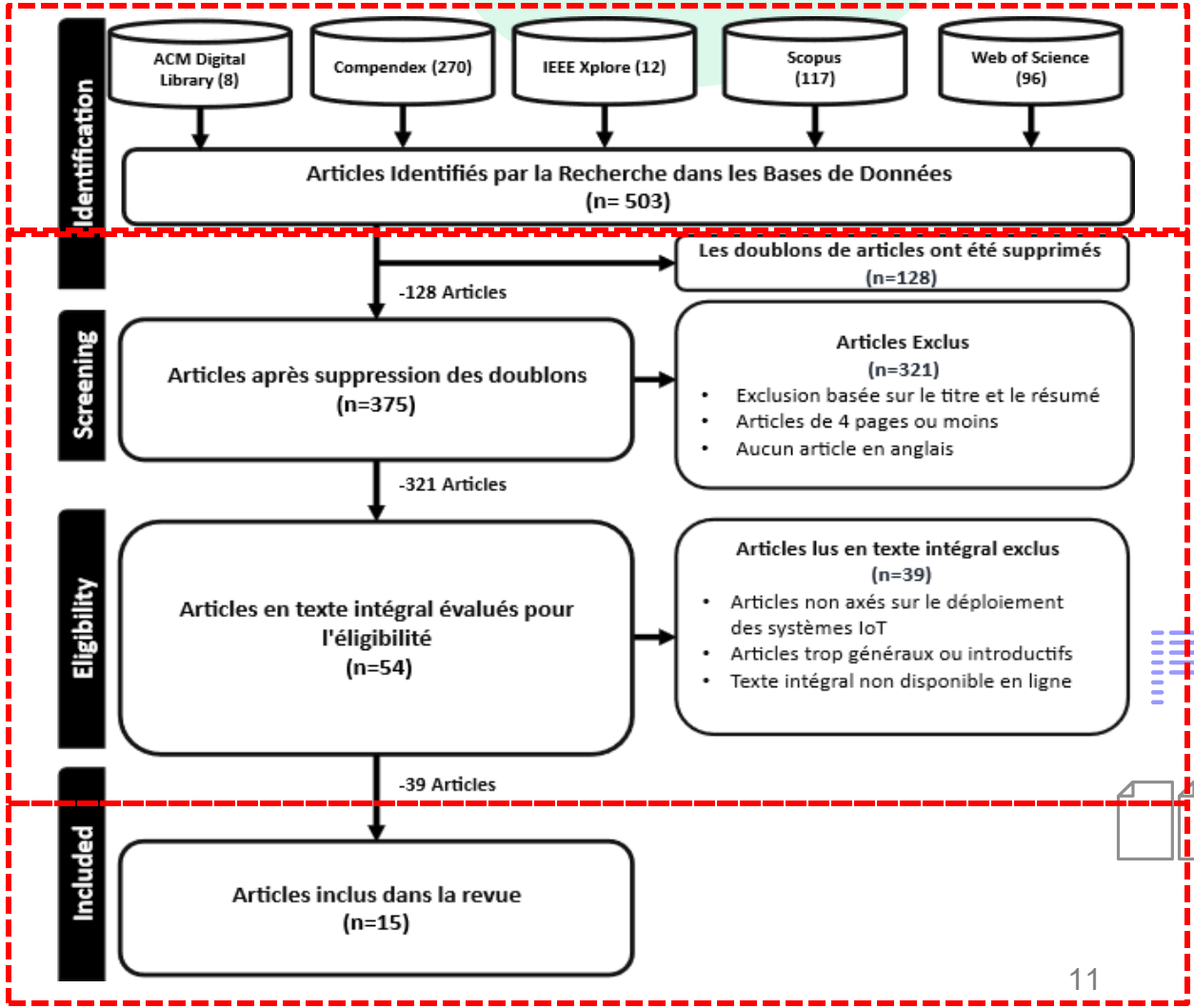
Revue de la littérature académique

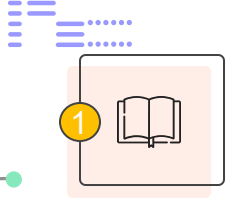
- La SLR est guidée par le protocole **PRISMA**
- Notre requête de recherche a été structurée selon le modèle **PICO**

1. Identification des termes clés ("microservices", "IoT"….)
2. Identification des termes alternatifs
3. Faire la combinaison avec OU / ET

(deploy **OR** install) **AND** (IoT **OR** Edge **OR** 'Internet of Things') **AND** (microservice)

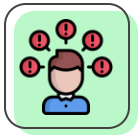
Revue de la littérature académique





Revue de la littérature académique

Défis et approches



Problème de placement

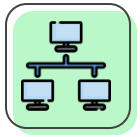
Approches mathématiques, les heuristiques…

7 articles

Équilibrage de charge dynamique

Approches d'apprentissage automatique ….

3 articles



Réseau

Approches pour répartir efficacement les charges réseau

1 articles

La complexité de programmation

Cadre de travail de microservices pour simplifier le développement d'applications IoT

1 articles

Orchestration et gestion des ressources

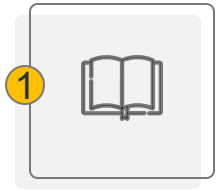
La mise en œuvre de services légers auto-déployables…

4 articles

Contributions

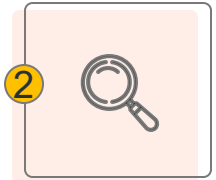
QR2 : Quelles sont les bonnes pratiques logicielles qui pourraient optimiser les performances et l'utilisation des ressources des systèmes IoT basés sur les microservices en périphérie ?

QR1



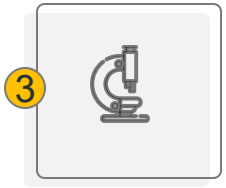
Revue de la littérature académique

Défis et approches



Revue de la littérature grise

QR 3



Étude empirique

Il existe une lacune dans la recherche qui intègre les pratiques du génie logiciel pour adresser ces défis

Revue de la littérature grise

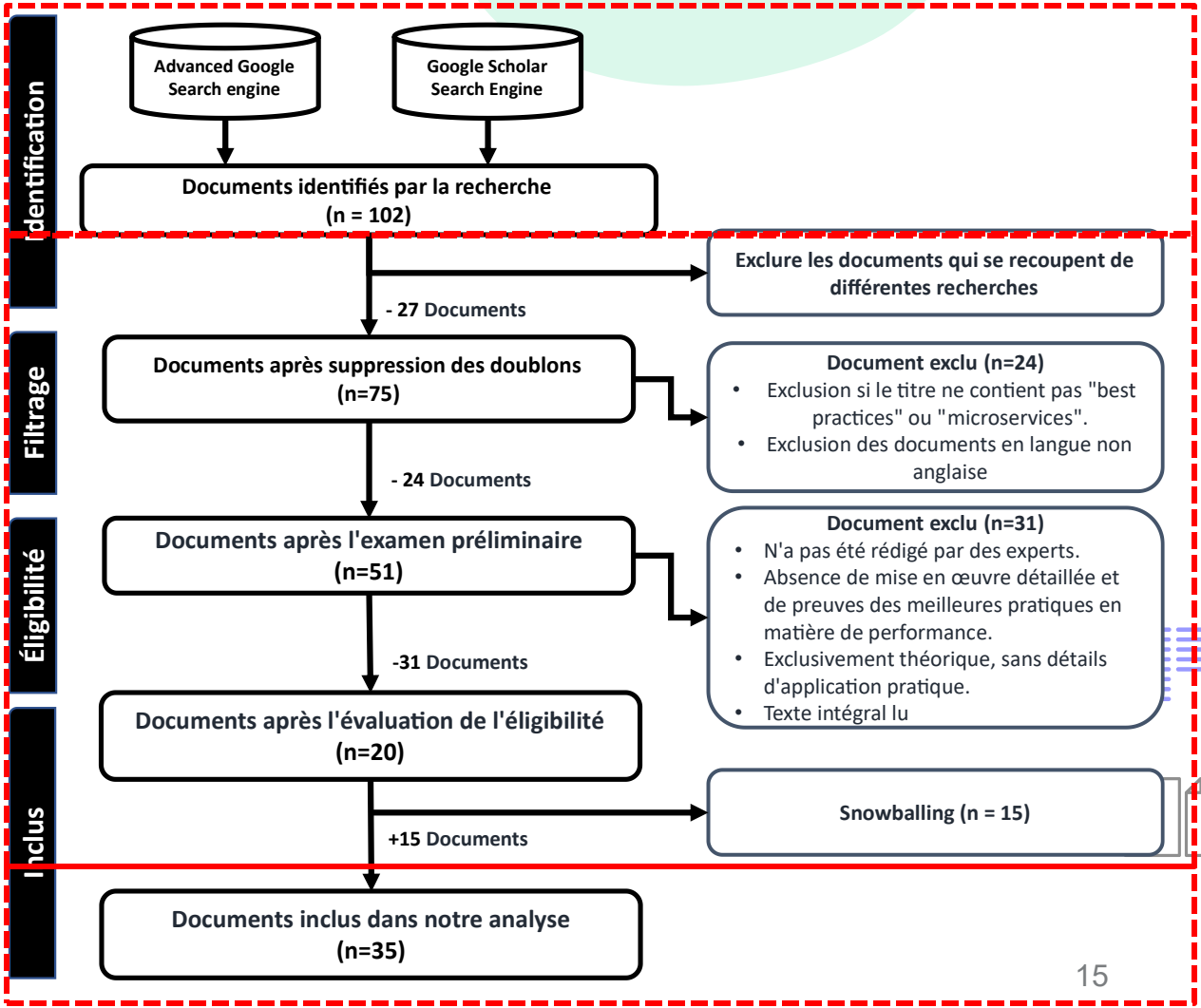
Pour garantir la rigueur et la reproductibilité, notre revue de la littérature grise est guidée par le protocole **PRISMA**

La requête de recherche a été structurée pour combiner les termes « performance », «microservices » et « meilleures pratiques »

(Best practices for microservices **AND** Performance) **OR** (Microservices deployment best practices **AND** Performance)

2

Revue de la littérature grise



Revue de la littérature grise

Comment extraire le catalogue de pratiques?

Méthode utilisée

1. Évaluation des documents
2. Vérification de l'expertise de l'auteur
3. Sélection des pratiques pertinentes

Points de difficulté

1. Extraction des recommandations
2. Variation des expressions et formulations

Revue de la littérature grise

Catalogue de pratiques

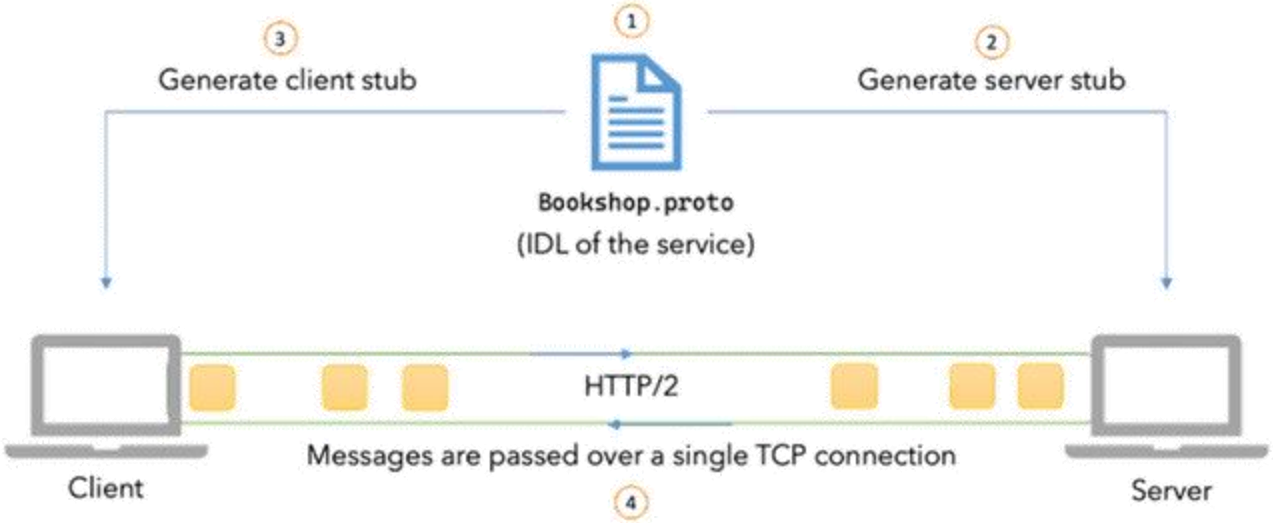
#	Titre des pratiques	N.de.refs
1	Maintenir un code clair et un versionnement	6
2	Utiliser une passerelle API	7
3	Utiliser des protocoles de communication efficaces et des motifs de communication	3
4	Utiliser la gestion des requêtes (par exemple, optimiser la charge utile, équilibrage de charge)	3
5	Utiliser des bases de données spécialisées comme NoSQL ou des bases de données en mémoire	3
6	Utiliser une base de données par microservice pour maintenir l'isolation des données	6
7	Mettre en œuvre des vérifications de l'état de santé et utiliser des outils de surveillance	3
8	Utiliser la conteneurisation pour les microservices pour un déploiement léger, cohérent et portable	6
9	Utiliser l'intégration continue (CI)	4
10	Utiliser des pipelines de déploiement continu (CD) et des pipelines d'automatisation	9
11	Surveiller les performances des microservices en temps réel pour identifier et résoudre les goulets d'étranglement potentiels	3
12	Utiliser le regroupement et la minification	2

Revue de la littérature grise

Exemple de pratiques

#	Titre des pratiques	N.de.refs
1	Maintenir un code clair et un versionnement	6
2	Utiliser une passerelle API	7
3	Utiliser des protocoles de communication efficaces et des motifs de communication	3
4	Utiliser la gestion des requêtes (par exemple, optimiser la charge utile, équilibrage de charge)	3
5	Utiliser des bases de données spécialisées comme NoSQL ou des bases de données en mémoire	3
6	Utiliser une base de données par microservice pour maintenir l'isolation des données	6
7	Mettre en œuvre des vérifications de l'état de santé et utiliser des outils de surveillance	3
8	Utiliser la conteneurisation pour les microservices pour un déploiement léger, cohérent et portable	6
9	Utiliser l'intégration continue (CI)	4
10	Utiliser des pipelines de déploiement continu (CD) et des pipelines d'automatisation	9
11	Surveiller les performances des microservices en temps réel pour identifier et résoudre les goulets d'étranglement potentiels	3
12	Utiliser le regroupement et la minification	2

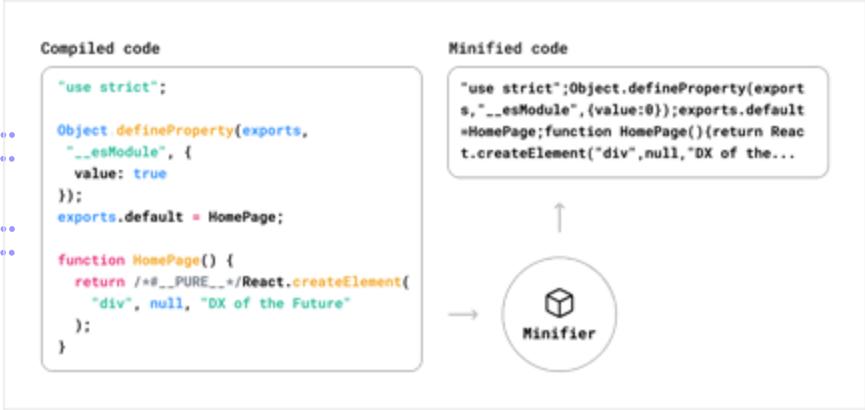
Exemple de pratiques : Protocole de communication RPC



Source : https://www.linkedin.com/pulse/introduction-implementing-grpc-net-creating-server-client-souhail-dtlkc

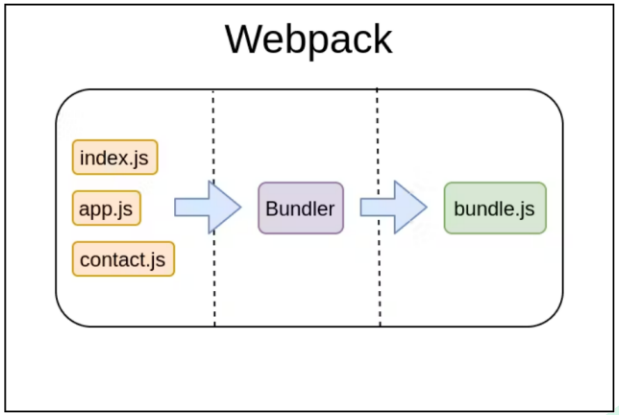
Exemple de pratiques : Minification et regroupement

- **Minification :** La minification consiste à réduire la taille des fichiers source



Source : https://wpdeveloper.com/how-to-minify-css-wordpress/

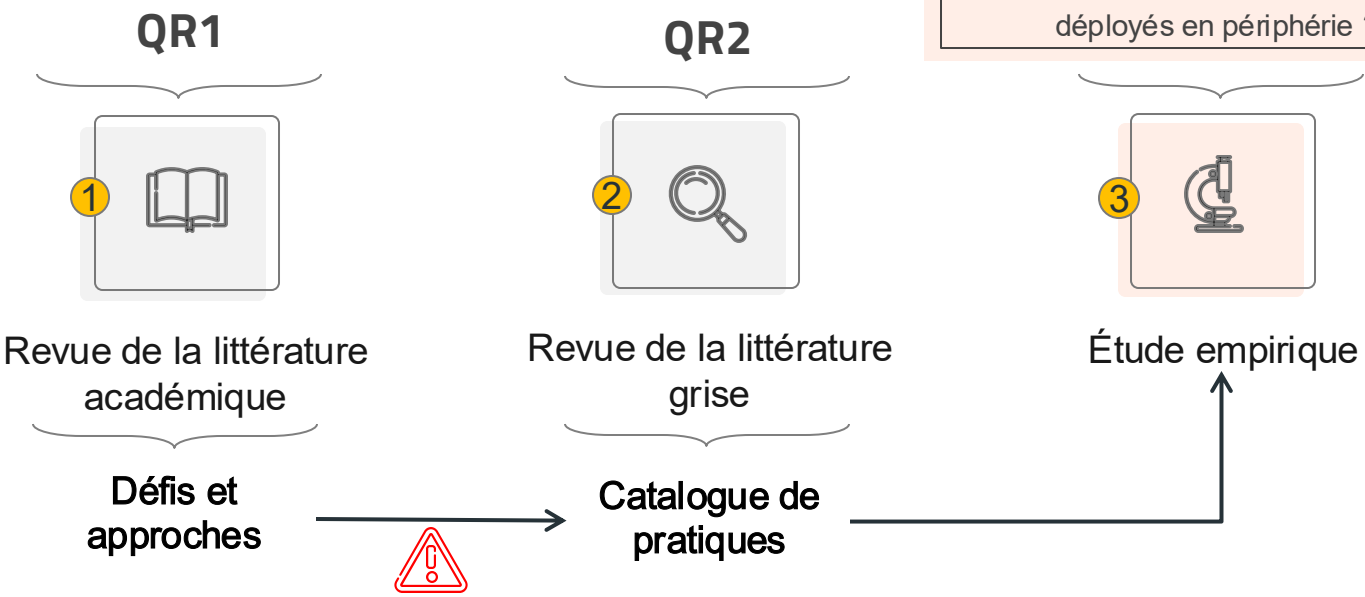
- **Regroupement** : Le regroupement est le processus de regroupement de plusieurs fichiers source en un seul fichier



Source: https://blog.skay.dev/a-basic-introduction-to-webpack

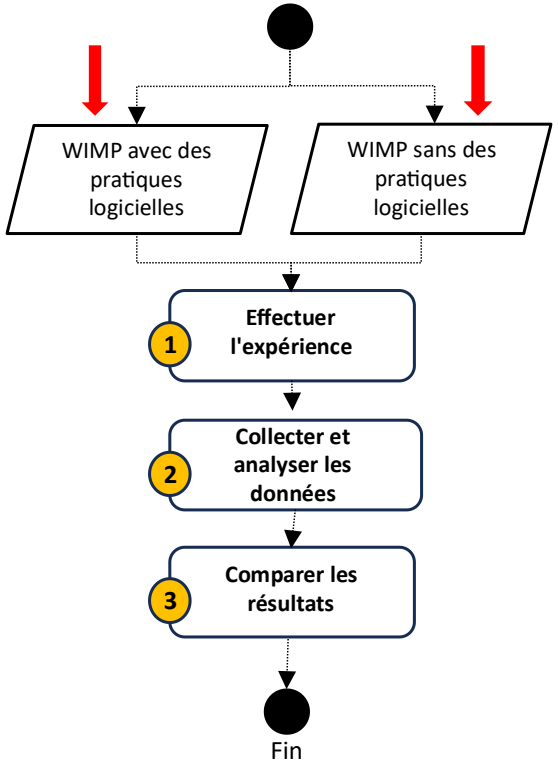
Contributions

QR3: Quel est l'impact sur les performances des bonnes pratiques logicielles identifiées pour les systèmes IoT basés sur les microservices déployés en périphérie ?



Conception de l'expérience

- Adopter la méthodologie « *Expériences à un facteur, deux niveaux* »
- Evaluer l'impact collectif des pratiques logicielles identifiées



3

Étude empirique

Cas d'étude

Le projet WIMP pour "Where Is My Professor?" est une preuve de concept qui permet aux étudiants de connaître la disponibilité de leurs enseignants en temps réel

WIMP

Fonctionnalités

1. Gestion des utilisateurs internes
2. Analyse et suivi de la disponibilité des professeurs

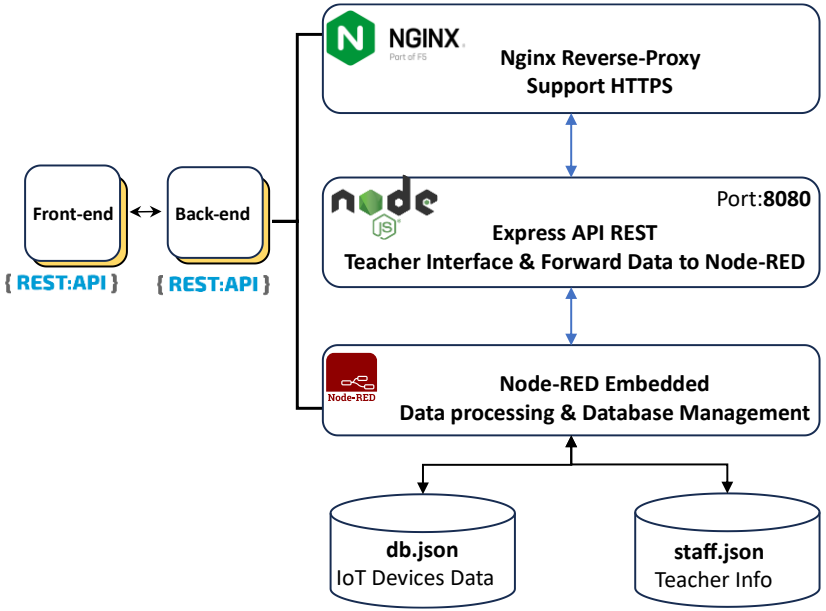
Technologies

1. Javascript
2. Node-RED

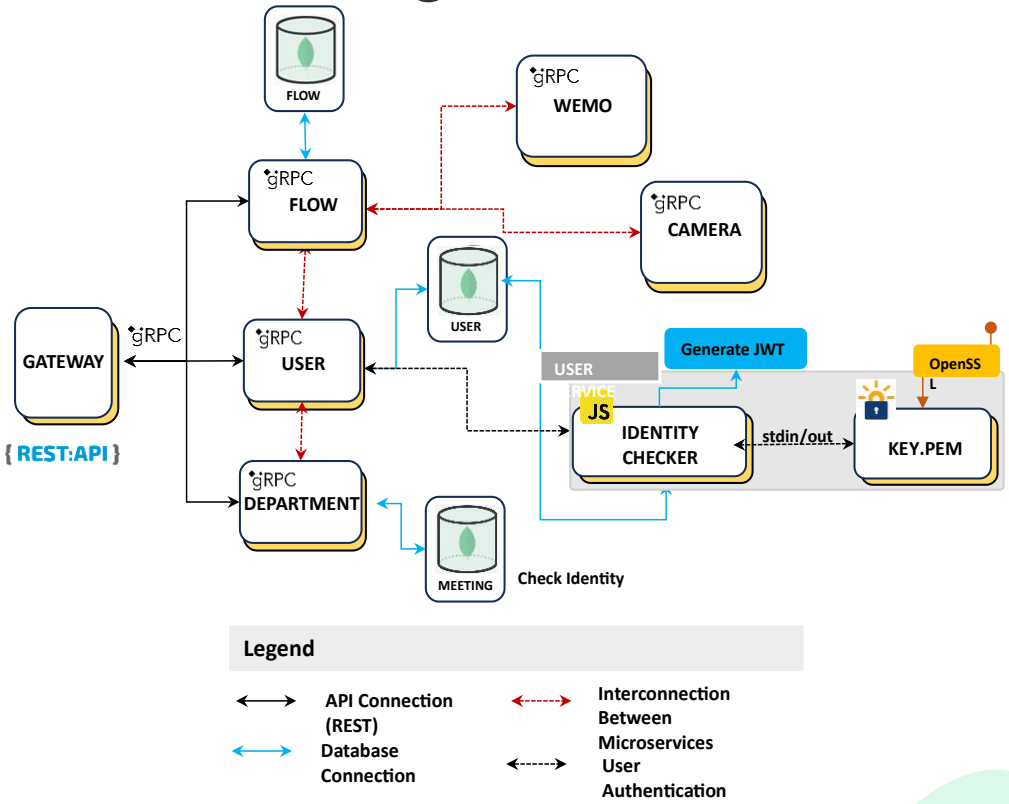
Capteurs et appareils IoT

1. Prise intelligente WEMO
2. Montre Fitbit

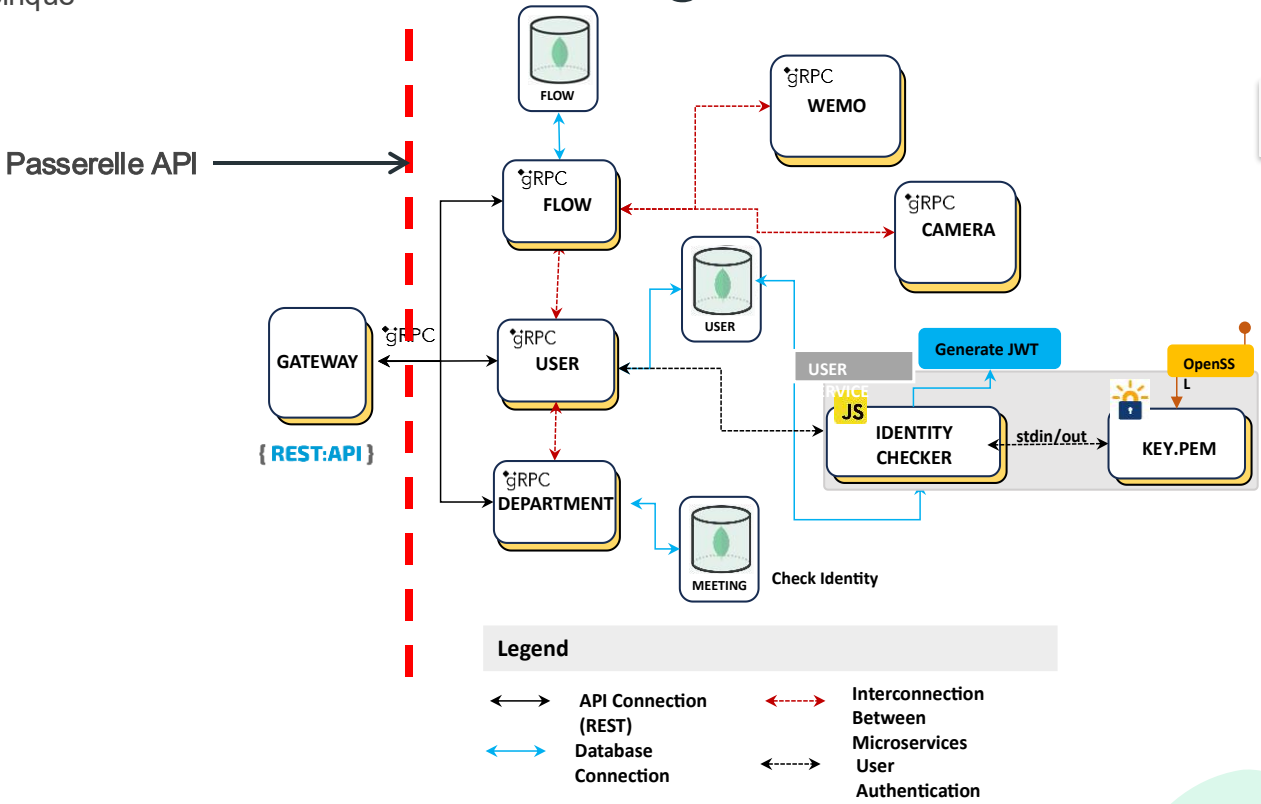
WIMP sans les pratiques logicielles



WIMP avec les pratiques logicielles



WIMP avec les pratiques logicielles



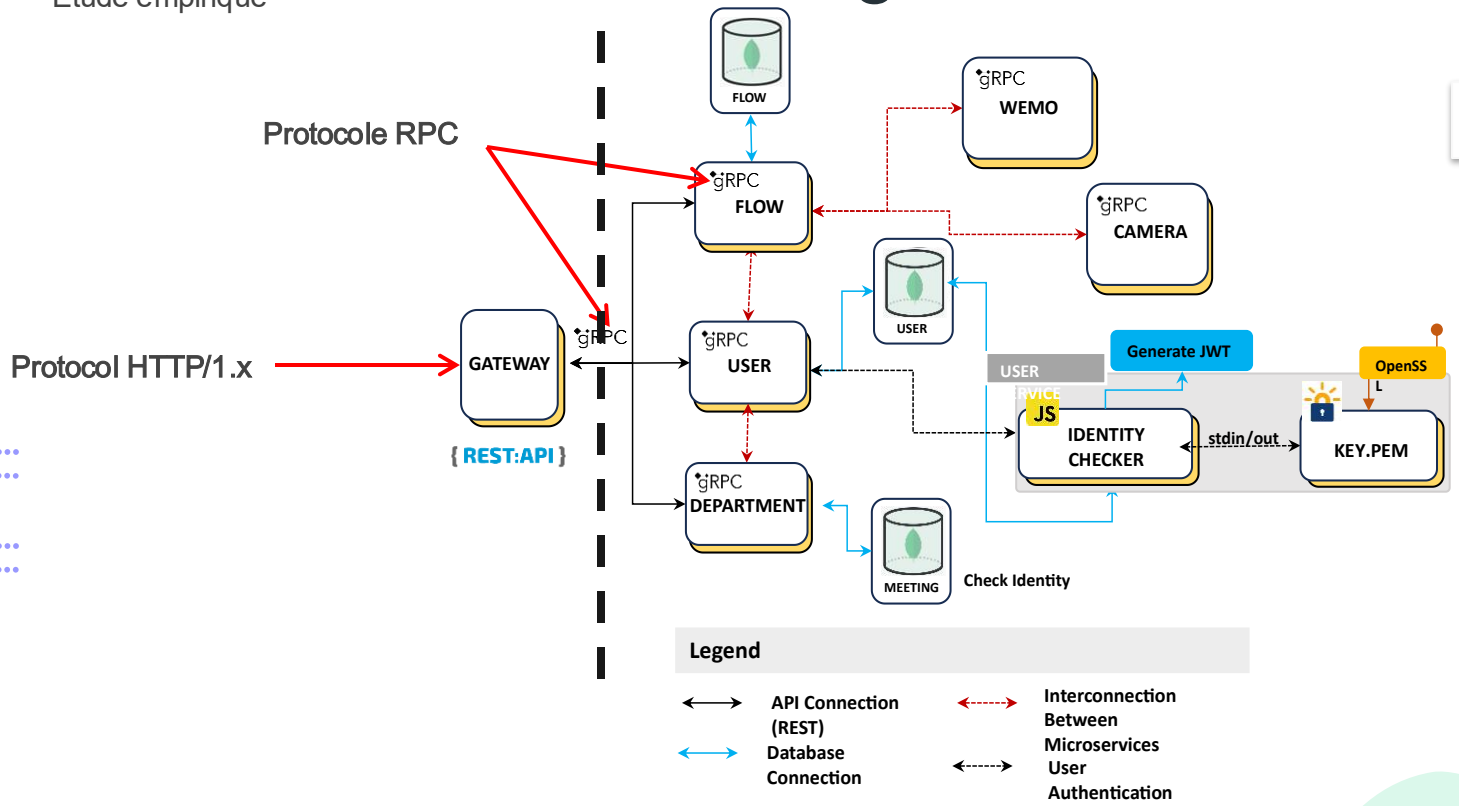
Pratique s'applique à tous les microservices

Minification et regroupement

docker

Conteneurisation des microservices

WIMP avec les pratiques logicielles



Pratique s'applique à tous les microservices

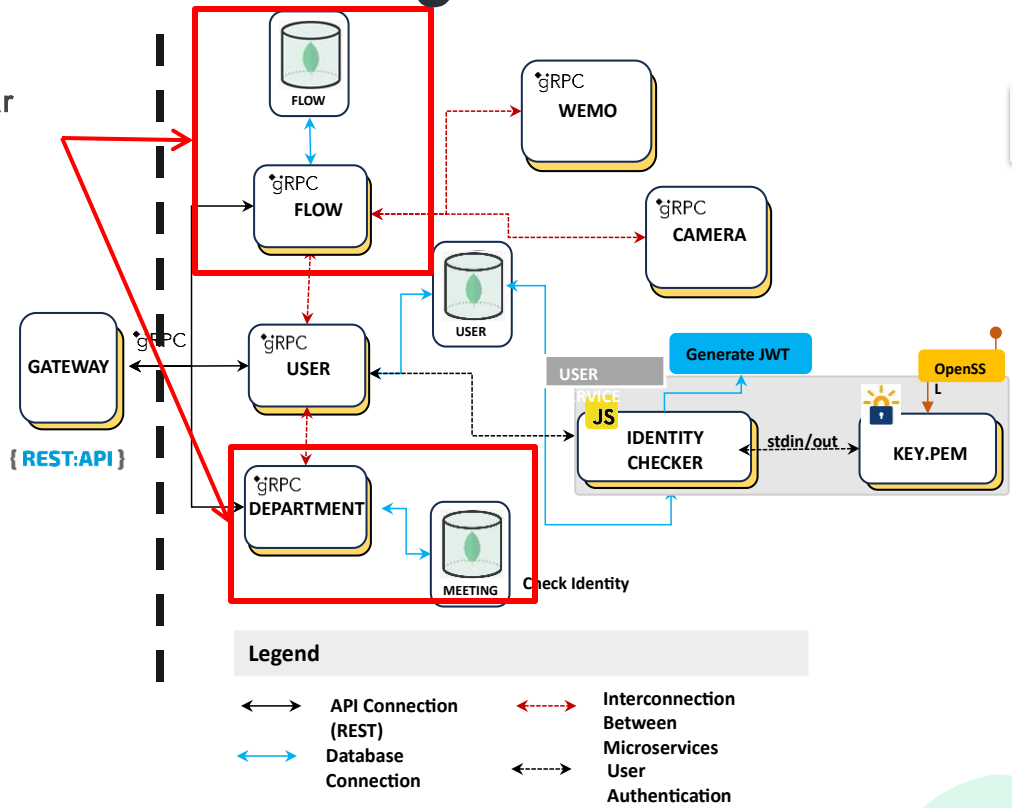
Minification et regroupement

docker

Conteneurisation des microservices

WIMP avec les pratiques logicielles

Base de données par service

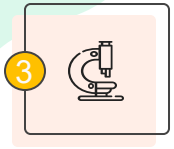


Pratique s'applique à tous les microservices

Minification et regroupement

docker

Conteneurisation des microservices



Étude empirique

Application de la pratique : Protocole RPC

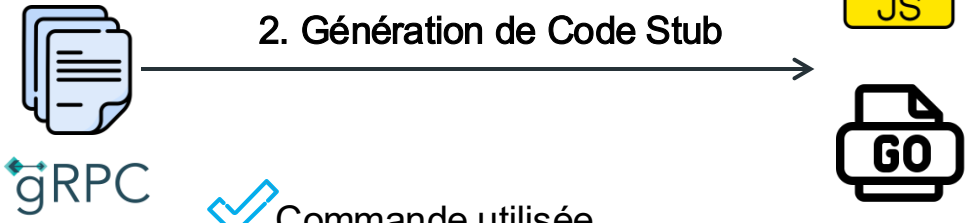
1 . Définition de User.proto

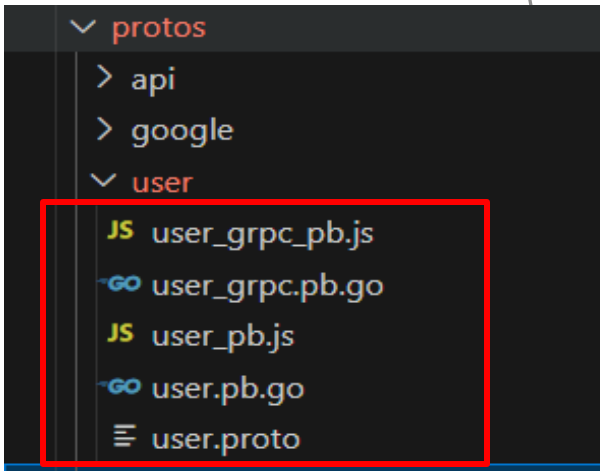
Service

Message

```proto
syntax = "proto3";
package protos.user;
option go_package =
"github.com/ptidejteam/WIMPv2/packages/protos/user";
// Le service qui définit la méthode RPC
service UserSvc {
    rpc Authenticate(LoginRequest) returns (LoginResponse);
    // autre methode ......
}
// Le message utilisé pour les requêtes et les réponses
message LoginRequest {
    string username = 1;
    string password = 2;
}
message LoginResponse {
    string access_token = 1;
    string refresh_token = 2;
}
```

Application de la pratique : Protocole RPC



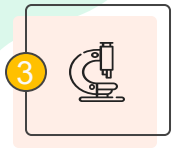


Commande utilisée

Exemple de commande : protoc --go_out=plugins=grpc:. user.proto

Points de difficulté

La gestion de différentes générations de types de code



Étude empirique

Application de la pratique : Protocole RPC

3. Implémentation de serveur pour microservice *User*

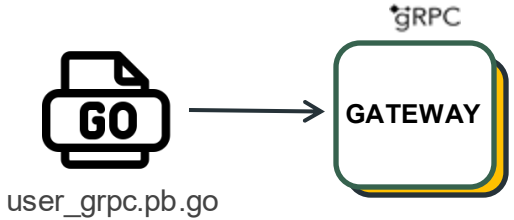
```
const grpc = require("@grpc/grpc-js");
const services = require("./routes/proto/user_grpc_pb");
const API = require("./routes/api");

async function main() {
  let server = new grpc.Server();
  let api = new API(grpc);
  server.addService(services.UserSvcService, {
    authenticate: api.authenticate,
  });
  let address = process.env.HOST + ":" + process.env.PORT;
// Spécifiez l'adresse et le port pour le serveur
  server.bindAsync(address, grpc.ServerCredentials.createInsecure(), () => {
    server.start();
    console.log("Server running at " + address);
  });
}

main();
```

Application de la pratique : Protocole RPC

4. Configuration des requêtes au niveau de la passerelle API



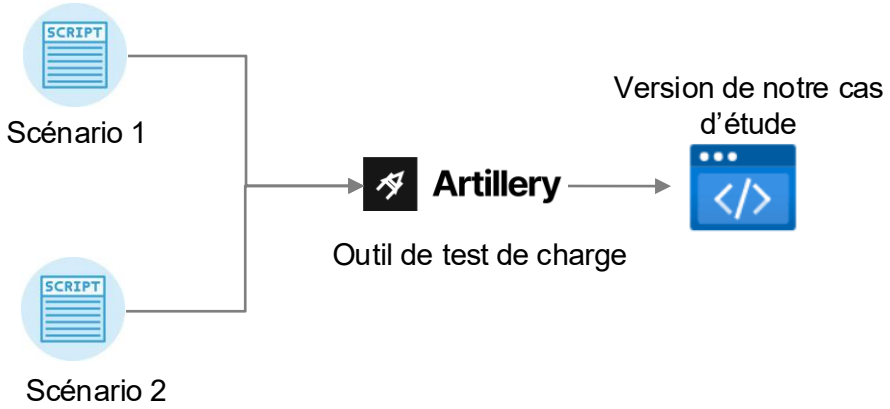

```go
func main() {
    mux := runtime.NewServeMux()
    opts := []grpc.DialOption{grpc.WithInsecure()}
    err := pb.RegisterUserSvcHandlerFromEndpoint(ctx, mux,
"localhost:50051", opts)
    if err != nil {
        log.Fatalf("Could not register service UserSvc: %v", err)
    }
    if err := http.ListenAndServe(":8080", mux); err != nil {
        log.Fatalf("Failed to serve: %v", err)
    }
}
```

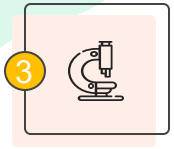
Point de difficulté

La gestion de différentes configurations des microservices

Scénarios de test

- **Scénario 1** : Envoi de 10 000 demandes d'authentification dans le système IoT
- **Scénario 2** : Envoi de 10 000 demandes pour récupérer la disponibilité du professeur





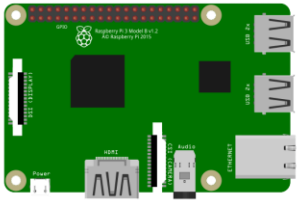
Étude empirique

Environnement de test

WIMP avec les pratiques logicielles

WIMP sans les pratiques logicielles

Déploiement



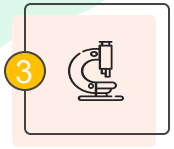
Raspberry Pi 4

Monitoring

Prometheus

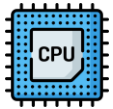
Test

Artillery



Étude empirique

Métriques



Performance du processeur

Performance de la mémoire

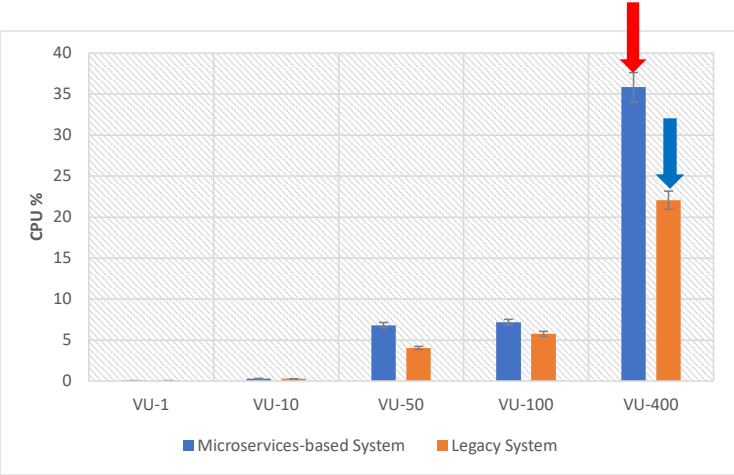
Performance du réseau: latence et débit

Prometheus pour collecter les données tout au long des deux scénarios de test de charge

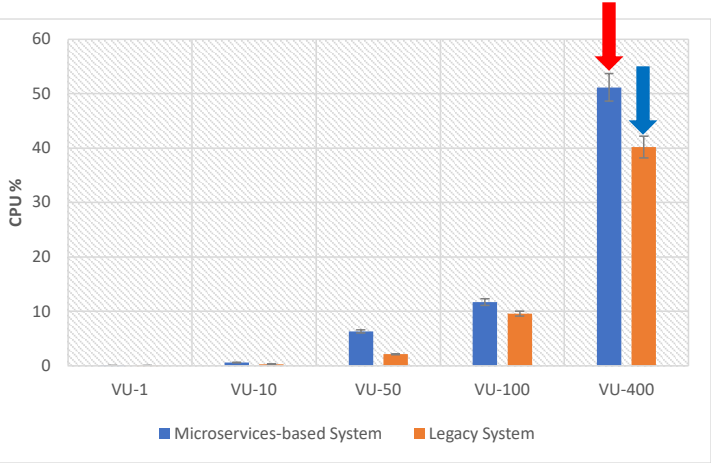
Rapport généré par l'outil de test de charge Artillery.io

Étude empirique

Résultats:
Utilisation du processeur



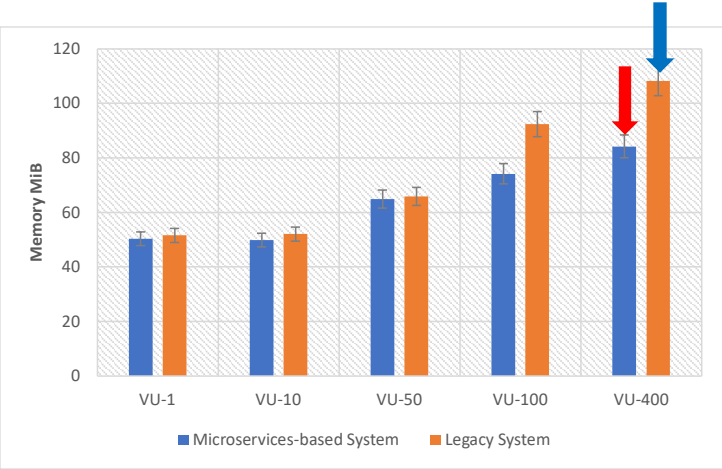
Utilisation du processeur – Scénario 1



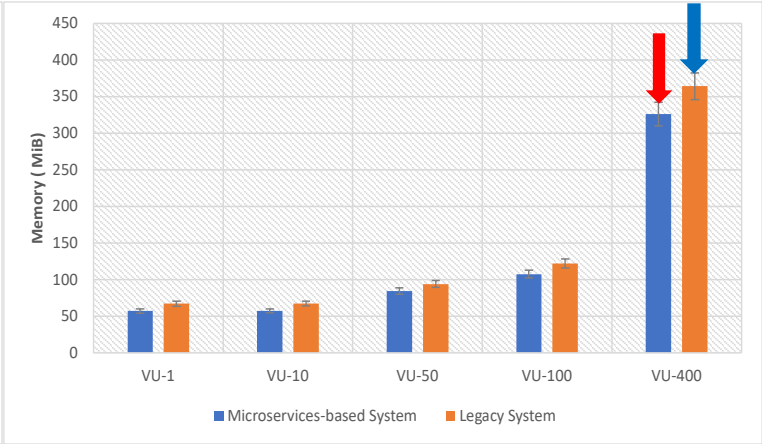
Utilisation du processeur – Scénario 2

Le système IoT basé sur les microservices requiert davantage de ressources CPU (~ 1.14%). Cela est dû à l'utilisation de plusieurs processus pour exécuter les différentes parties fonctionnelles pour exécuter les deux scenarios

Résultats: Utilisation de mémoire



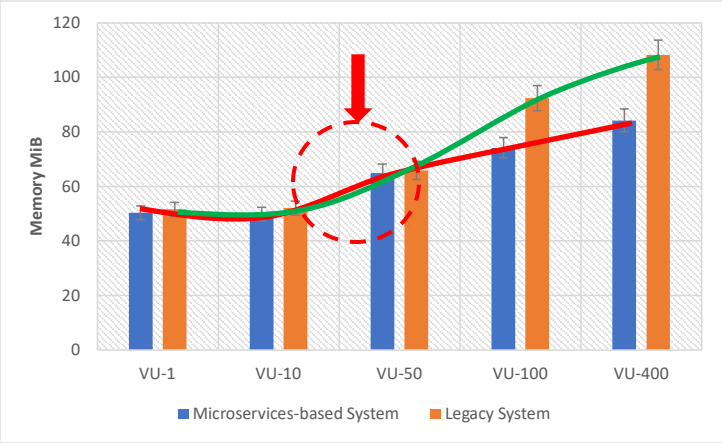
Utilisation de mémoire – Scénario 1



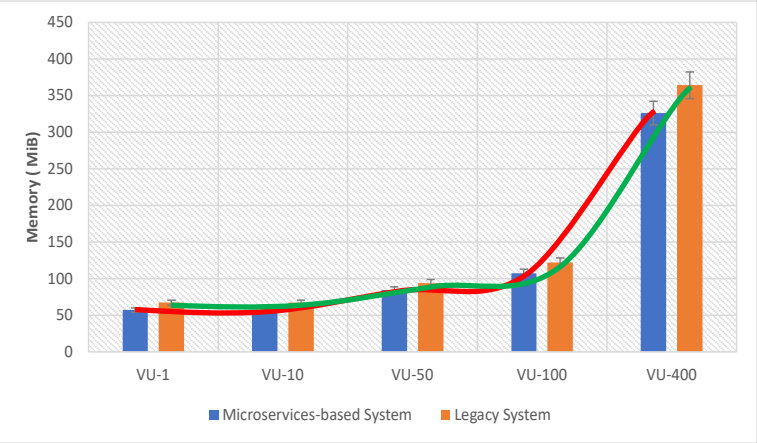
Utilisation de mémoire – Scénario 2

Le système basé sur les microservices s'avère plus économique en mémoire (~ 12%). . Cela est probablement dû à une optimisation plus poussée et à une empreinte mémoire réduite de chaque microservice

Résultats:
Utilisation de mémoire



Utilisation de mémoire – Scénario 1

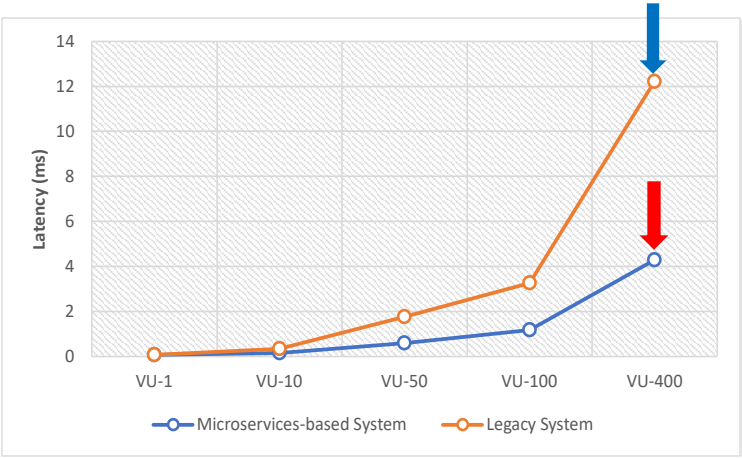


Utilisation de mémoire – Scénario 2

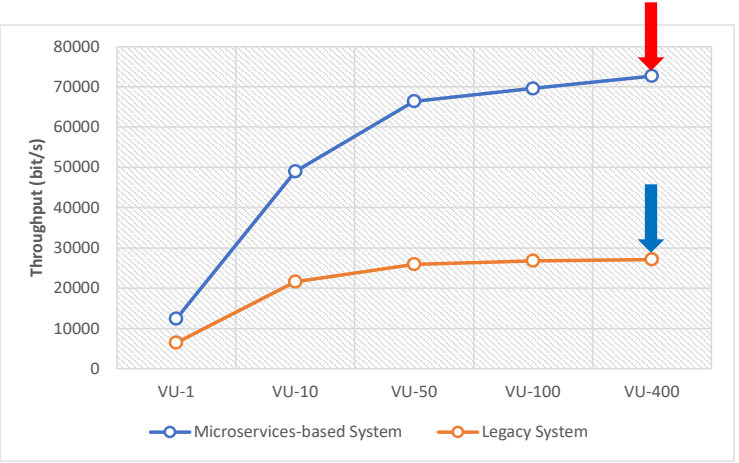
On remarque des fluctuations au niveau de l'utilisation de mémoire

Étude empirique

Résultats: Performance réseau



Latence moyenne

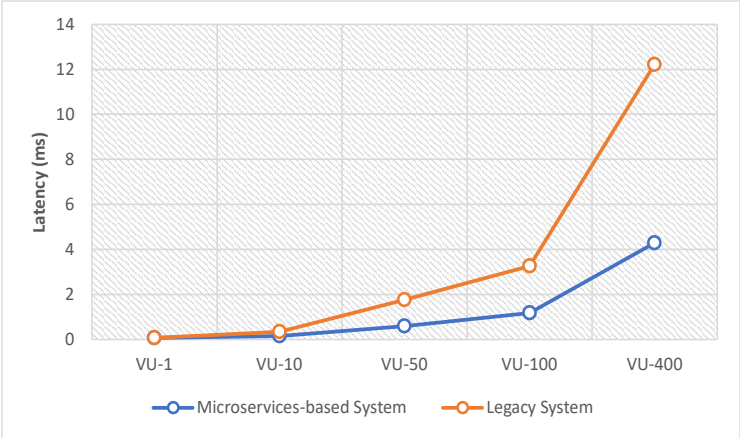


Débit de transfert moyen

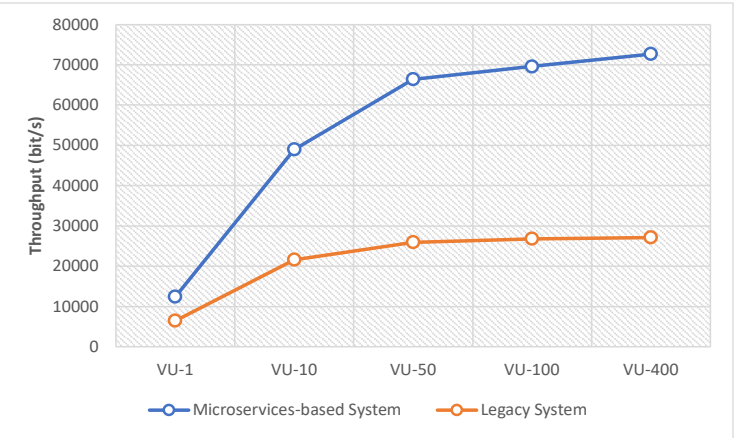
Le système IoT basé sur les microservices a montré une amélioration du débit et une réduction de la latence par rapport au système IoT existant

Étude empirique

Résultats: Performance réseau



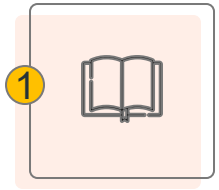
Latence moyenne



Débit de transfert moyen

Le débit plus élevé en kbit/s implique que le système basé sur les microservices peut traiter plus de requêtes par seconde que le système hérité pour la même charge de travail

Menaces à la validité

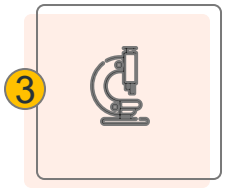


Revue de la littérature académique

Sélection et portée des articles

Revue de la littérature grise

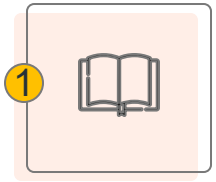
Risque de qualité des documents



Étude empirique

Utilisation d'un seul système d'étude de cas
Expansion des tests

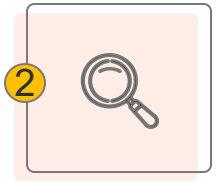
Conclusion et travaux futurs



Revue de la littérature académique

Défis et approches

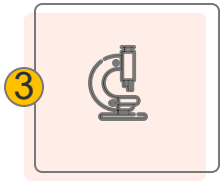
Exemple : Problème de placement et les approches mathématiques pour résoudre ce problème



Revue de la littérature grise

Catalogue de pratiques

12 pratiques telles que la conteneurisation, la passerelle API...



Étude empirique

Évaluation comparative des performances des systèmes

Améliorations significatives des performances et de l'efficacité du système, notamment en ce qui concerne le débit, la latence et l'utilisation de la mémoire

Travaux futurs

- Étude des impacts individuels de ces pratiques et évaluation de leurs effets sur des systèmes IoT plus vastes et plus complexes