



Projet pluridisciplinaire.

2ème année de second cycle.

Développement d'une solution e-commerce à base de micro services pour une boutique de vente de meubles en ligne.



Rapport général.

Encadreur

- M. Mahammed Nadir.

Équipe de développement

- Bellaoudj Ismail.
- Dermi Malika.
- Gaffour Abderrahmene
- Habib Kaouther.

Introduction générale	4
Chapitre I : Division des services d'E-meuble.	6
Architecture des microservices:	6
Services d'E-meuble:	6
Conclusion:	7
Chapitre II : Conception et analyse de l'application E-meuble.	8
1. Ingénierie des besoins	8
1.1. Diagrammes des cas d'utilisation	8
1.1.1 Diagramme du Cas Authentications d'administrateurs:	8
1.1.2 Diagramme du Cas fonctions d'administrateurs:	9
1.1.3 Diagramme du Cas fonctions du client:	9
1.2. Diagrammes des séquences:	10
1.2.1 Diagramme de séquence de "Fonctions d'administrateurs":	10
1.2.2 Diagramme de séquence de "Fonctions du client":	11
Figure 07: diagramme du cas fonctions du client.	11
1.3. Diagrammes des classes:	12
1.4. Diagrammes d'activités:	13
1.4.1 Diagramme d'activité de "processus du panier":	13
1.4.2 Diagramme d'activité de "processus du commande":	14
1.5. Diagramme de déploiement:	15
2. Conclusion:	15
Chapitre III : Implémentation.	16
Les Principes d'implémentation:	16
1.1 EDGE Microservices:	16
1.1.1 EUREKA SERVER: La découvrabilité:	17
1.1.1 SPRING CLOUD GATEWAY:	17
1.2 Microservices:	18
1.2.1 MS-AUTH:	18
1.2.2 MS-CLIENT:	18
1.2.3 MS-ORDER:	19
1.2.4 MS-ADMIN:	19
Les interfaces d'e-meuble:	20
2.1 Page d'accueil:	20
2.1 Page détails du produit et commande:	21
2.1 Checkout Page (Panier):	22
Conclusion:	23
Chapitre IV : Les Outils de travail.	24
Langages et frameworks:	24
1.1 SPRING BOOT framework :	24
1.2 Angular:	24
1.3 FLUTTER:	25
1.4 MYSQL:	25

Outils de travail:	25
2.1 GITHUB:	25
2.2 TRELLO:	26
2.3 ASTAH UML:	26
Conclusion générale:	27

Introduction générale

Le commerce en ligne, la vente en ligne, la vente à distance ou encore cybercommerce est l'échange onéreux de biens et de services par le biais des réseaux informatiques, généralement internet.

Ces dernières années, le secteur du commerce électronique s'est inscrit en forte évolution. Il représente actuellement une composante très dynamique de l'économie numérique.

L'augmentation du nombre d'internautes, ainsi que la variété et la qualité des produits et services proposés en ligne ont favorisé ce développement. La dynamique du secteur et les opportunités qu'offrent les cybercommerçants aux cyberconsommateurs renforcent les investissements dans ce secteur.

Notre projet consiste à réaliser une solution informatique pour le commerce du secteur des meubles. On aborde une problématique à laquelle on propose une solution conceptuelle et une réalisation d'un outil qui satisfait les besoins d'un client et d'un commerçant.

Problématique et objectifs

La réalisation de telle application nous permet d'atteindre certains objectifs qu'on cite:

- Assurer au gérant de ce business (L'admin) une interface simple à manipuler ou il peut superviser ses différentes annexes de son magasin, et tous ses produits.
- Présenter au client un catalogue de produits riche et clair en exposant les détails de chaque produit.
- La facilité de filtrer les produits par catégorie.
- Permettre au client de voir les commentaires des gens sur n'importe quel produit.
- La possibilité pour le client de commander en ligne, de choisir la livraison à domicile et de donner son avis en toute transparence.

Alors pour accomplir ces derniers une problématique se pose:

Comment peut-on vraiment intégrer l'architecture des microservices pour une telle application? Existe-t-il un moyen pour réaliser une application qui

répond aux besoins du commerçant et du client et de satisfaire leurs attentes en termes de fonctionnalités et d'ergonomie ? Si c'est le cas, comment et quelle est la méthode ou processus de développement à suivre pour réaliser un tel système?

Pour répondre à ces problèmes, On vous propose ce modeste travail constitué de trois chapitres :

- **Chapitre I** : Division des services de E-meuble.

Dans ce chapitre on va définir l'architecture des microservices utilisés pour ce projet, et comment on a divisé les différentes fonctions en services.

- **Chapitre II** : Conception et analyse de l'application E-meuble.

Ce chapitre expose tous les diagrammes réalisés lors de la conception et l'analyse de l'application.

- **Chapitre III**: Implémentation

Exposition des différentes pages et interfaces web et mobile implémentées.

- **Chapitre IV** : Les outils de travail.

Dans ce chapitre on démontre les outils utilisés pour ce projet.

- **Conclusion générale.**

Chapitre I : Division des services d'E-meuble.

1. Architecture des microservices:

Il existe un certain nombre de définitions des microservices. La plus commune et généralisée reste celle de Martin Fowler qui dit :

« Le style architectural des microservices est une approche permettant de développer une application unique sous la forme d'une suite logicielle intégrant plusieurs services. Ces services sont construits autour des capacités de l'entreprise et peuvent être déployés de façon indépendante. »

Concrètement, les microservices sont une méthode développement logiciel utilisée pour concevoir une application comme un ensemble de services modulaires. Chaque module répond à un objectif métier spécifique et communique avec les autres modules.¹

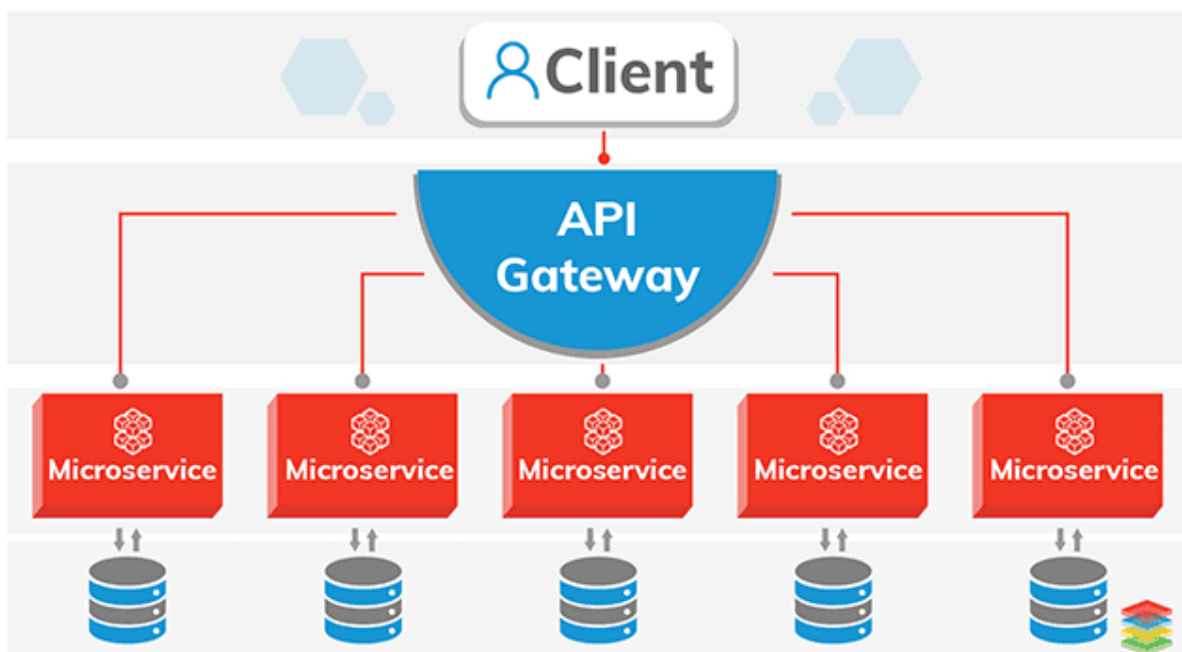


Figure 01: l'architecture des microservices.

2. Services d'E-meuble:

Pour diviser n'importe quelle application en microservices, il existe plusieurs possibilités de découpage:

- Par « **business capability** » : Liés à l'activité économique de l'entreprise
- Par « **Bounded Context** » Contexte Bornés

¹ <https://www.talend.com/fr/resources/guide-microservices/>

On a utilisé le contexte borné, il consiste à regrouper plusieurs **microservices** ayant en commun un contexte fonctionnel.

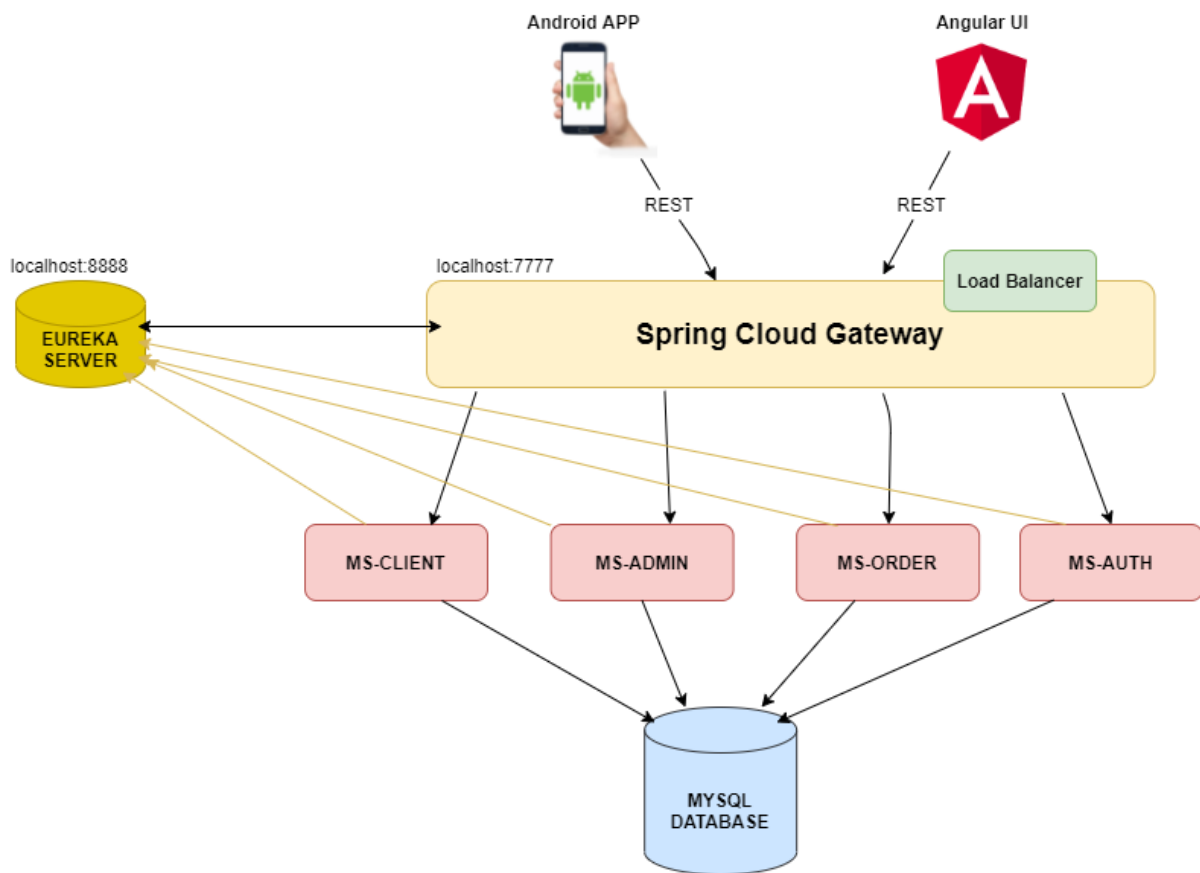


Figure 02: Services d'e-meuble.

La figure ci-dessus montre les différents services pour notre application qu'on va détailler en chapitre 3 : Implémentation.

3. Conclusion:

Dans ce chapitre on a illustré l'architecture utilisée pour ce projet et l'approche de découpage des services de l'application.

Chapitre II : Conception et analyse de l'application E-meuble.

1. Ingénierie des besoins

L'ingénierie des besoins vise à définir les services attendus par le client d'un système et les contraintes sous lesquelles il s'exécutera et sera développé, en se basant sur des pratiques du génie logiciel afin de garantir un logiciel de qualité à moindre coût.

1.1. Diagrammes des cas d'utilisation

Les diagrammes de cas d'utilisation décrivent les fonctions générales et la portée d'un système. Ces diagrammes identifient également les interactions entre le système et ses acteurs. Les cas d'utilisation et les acteurs dans les diagrammes de cas d'utilisation décrivent ce que le système fait et comment les acteurs l'utilisent, mais ne montrent pas comment le système fonctionne en interne.

1.1.1 Diagramme du Cas Authentications d'administrateurs:

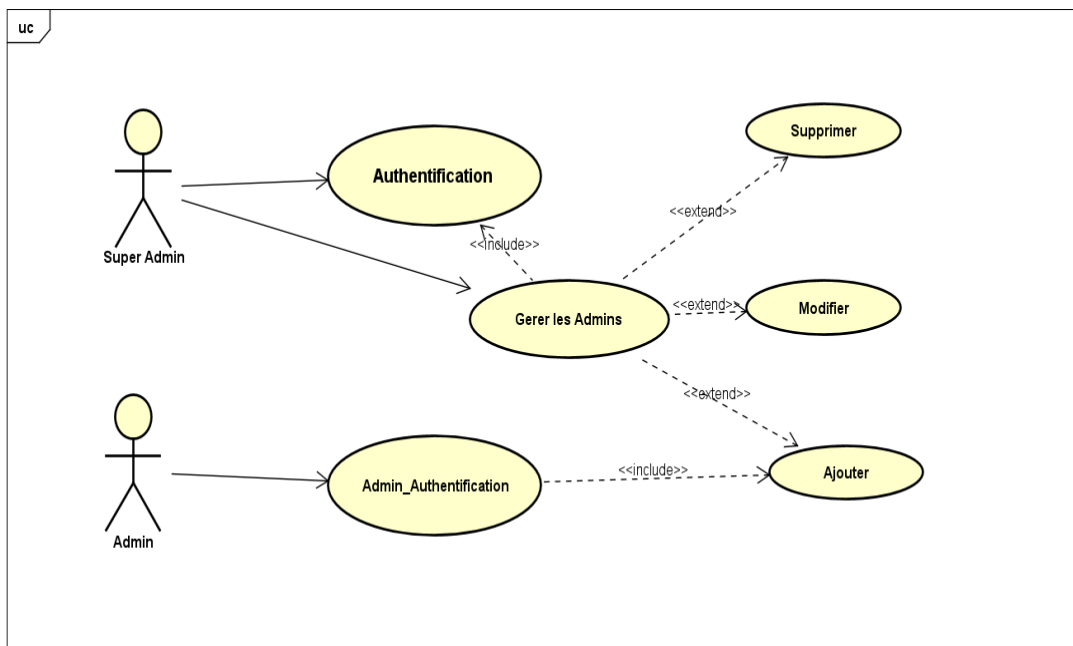


Figure 03: diagramme du cas authentification d'administrateurs.

1.1.2 Diagramme du Cas fonctions d'administrateurs:

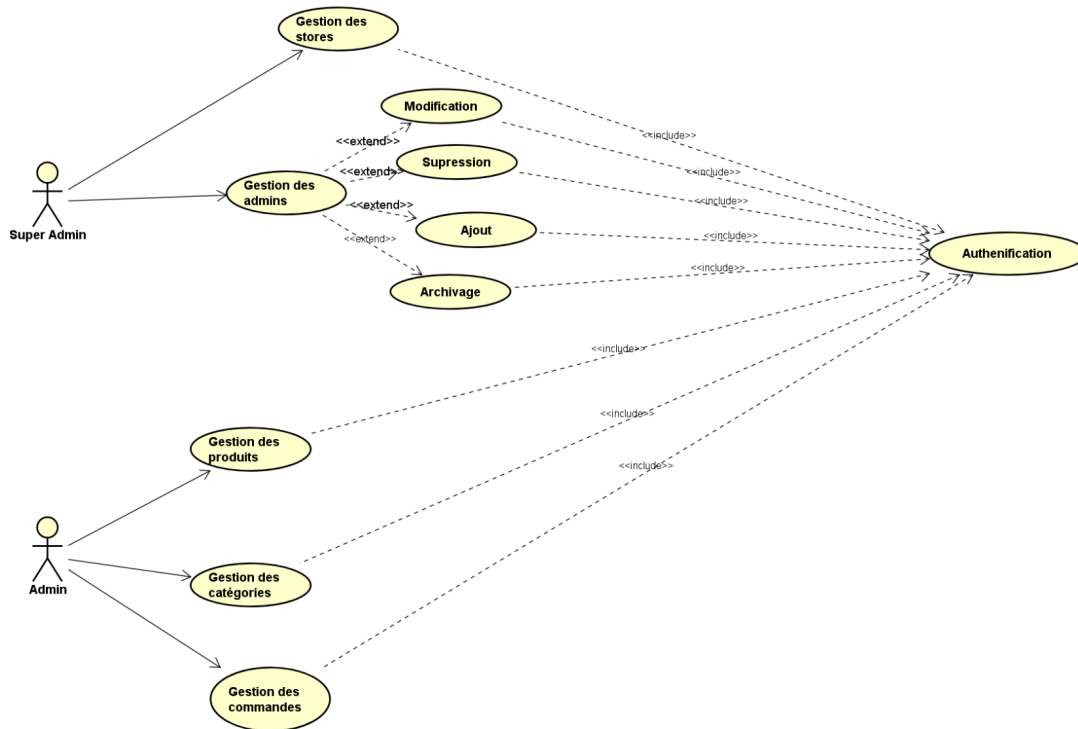


Figure 04: diagramme du cas fonctions d'admin.

1.1.3 Diagramme du Cas fonctions du client:

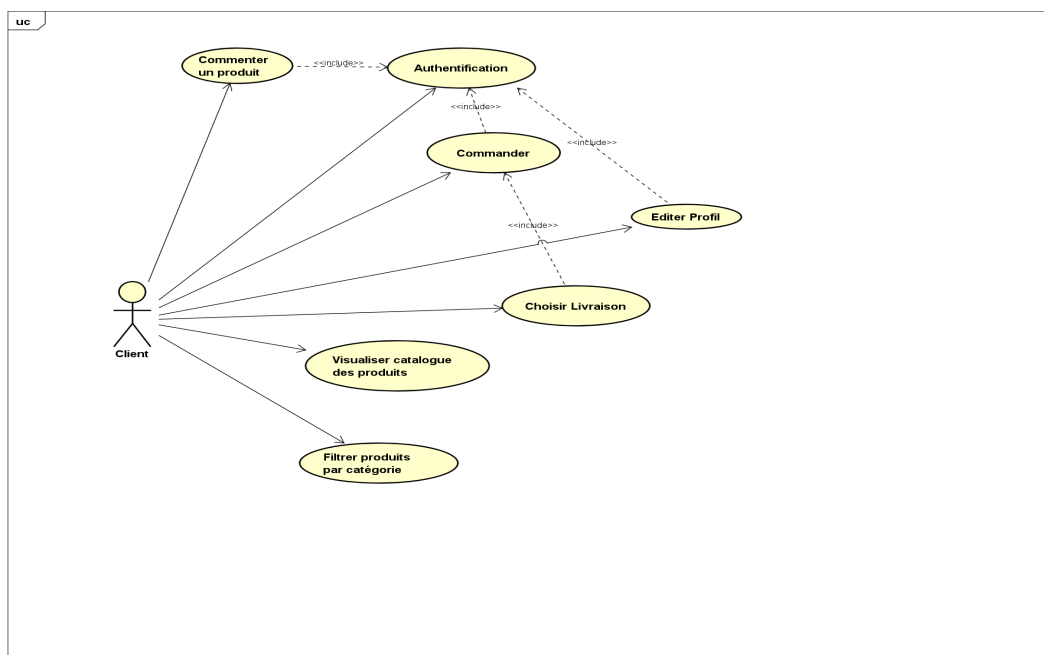


Figure 05: diagramme du cas fonctions du client.

1.2. Diagrammes des séquences:

Les diagrammes de séquences permettent de décrire COMMENT les éléments du système interagissent entre eux et avec les acteurs :

- Les objets au cœur d'un système interagissent en s'échangeant des messages.
- Les acteurs interagissent avec le système au moyen d'IHM (Interfaces Homme-Machine)

1.2.1 Diagramme de séquence de "Fonctions d'administrateurs":

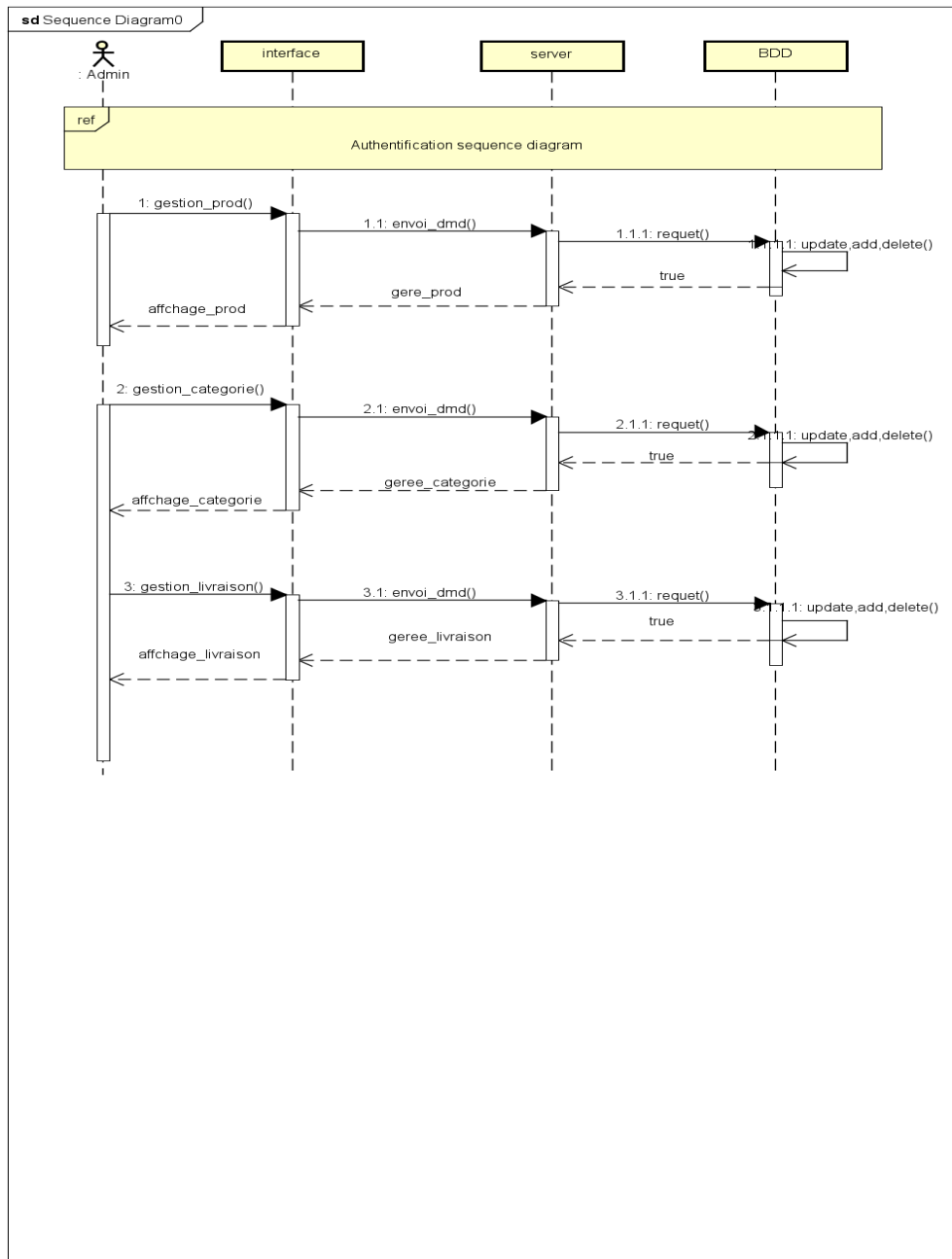


Figure 06: diagramme de séquences fonctions d'admin.

1.2.2 Diagramme de séquence de "Fonctions du client":

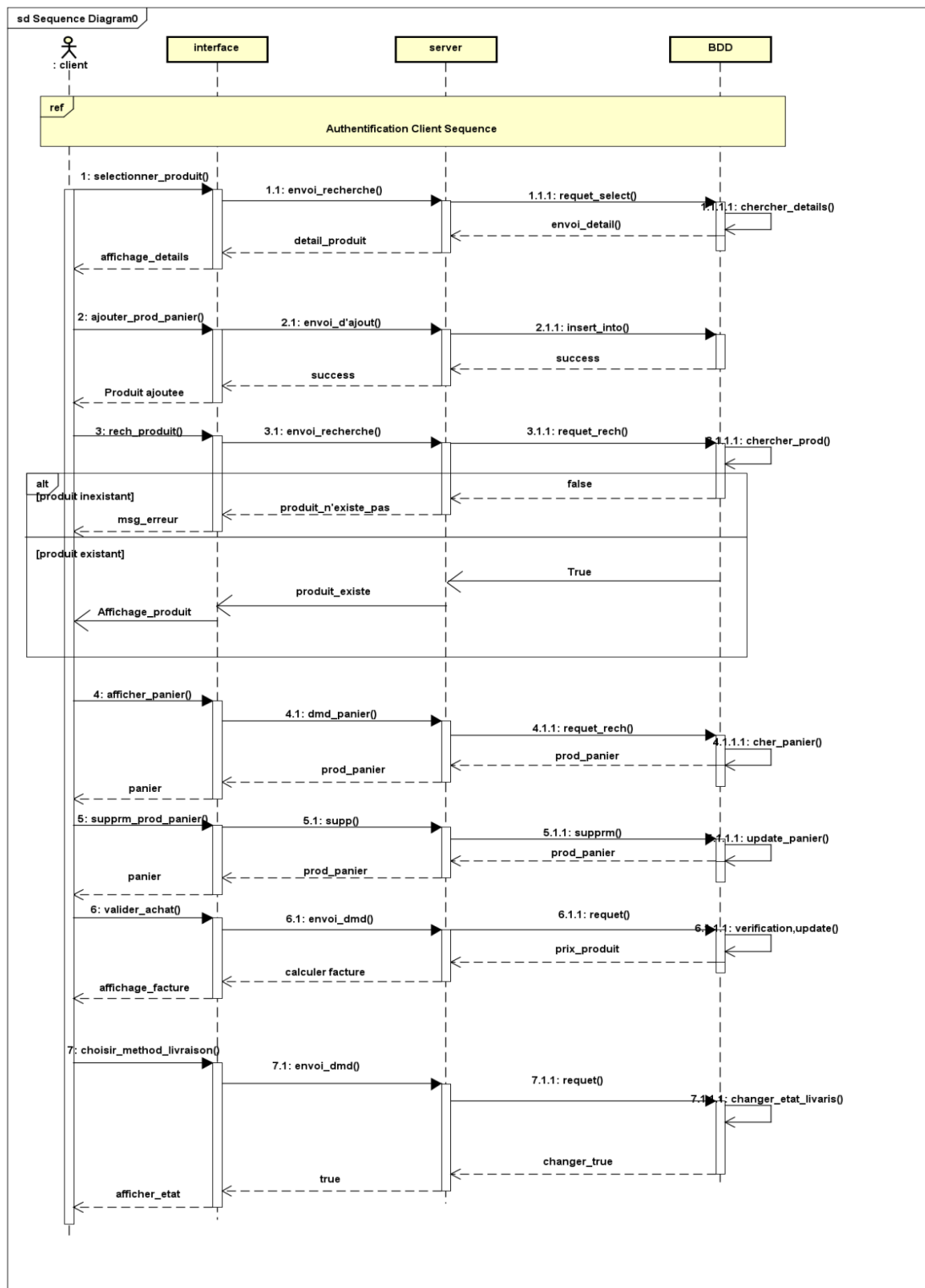


Figure 07: diagramme du cas fonctions du client.

1.3. Diagrammes des classes:

Les diagrammes de classes sont l'un des types de diagrammes UML les plus utiles, car ils décrivent clairement la structure d'un système particulier en modélisant ses classes, ses attributs, ses opérations et les relations entre ses objets.

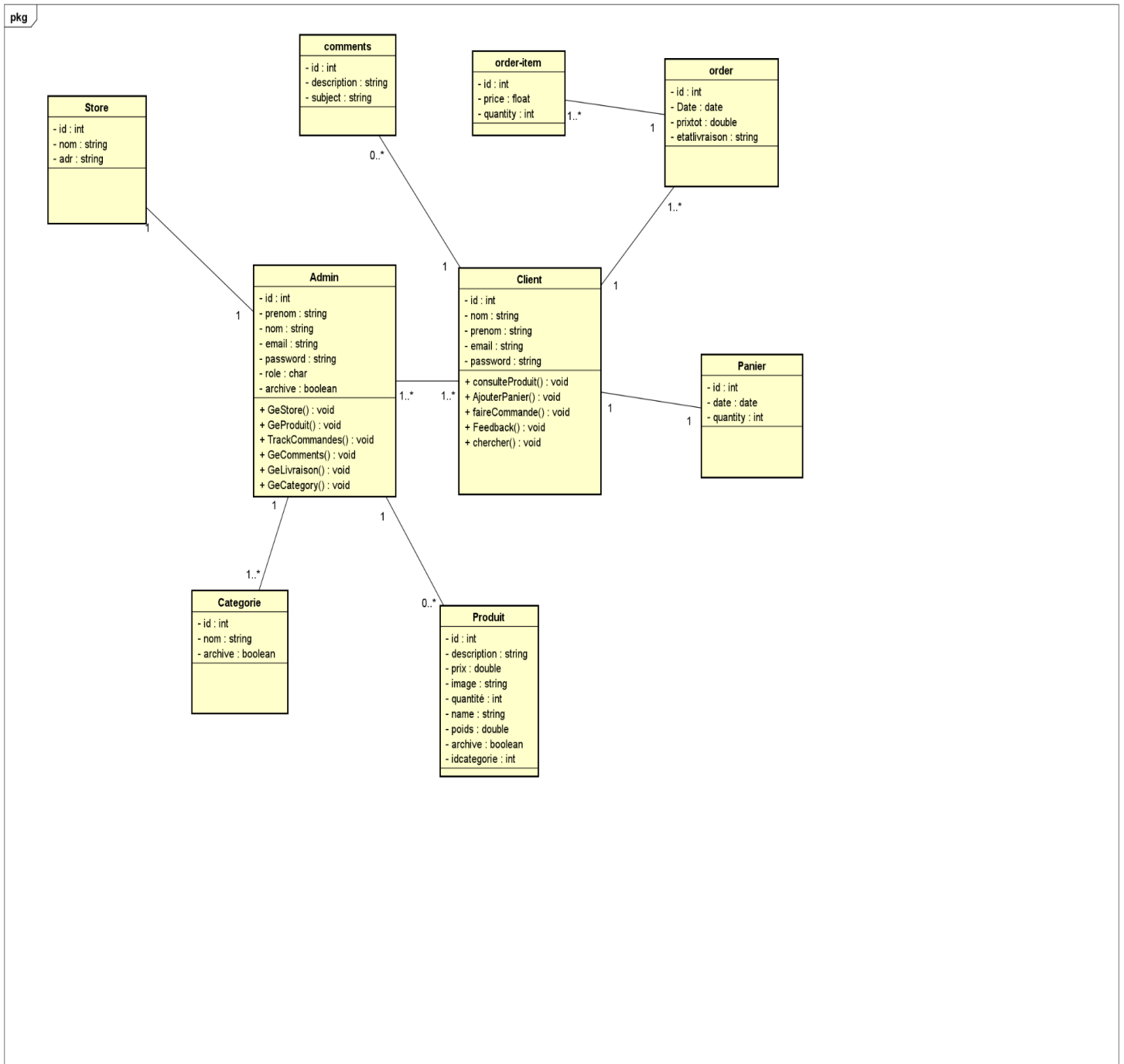


Figure 08: diagramme des classes.

1.4. Diagrammes d'activités:

Le diagramme d'activité est un diagramme comportemental d'UML, permettant de représenter le déclenchement d'événements en fonction des états du système et de modéliser des comportements parallélisables. Le diagramme d'activité est également utilisé pour décrire un flux de travail.

1.4.1 Diagramme d'activité de “processus du panier”:

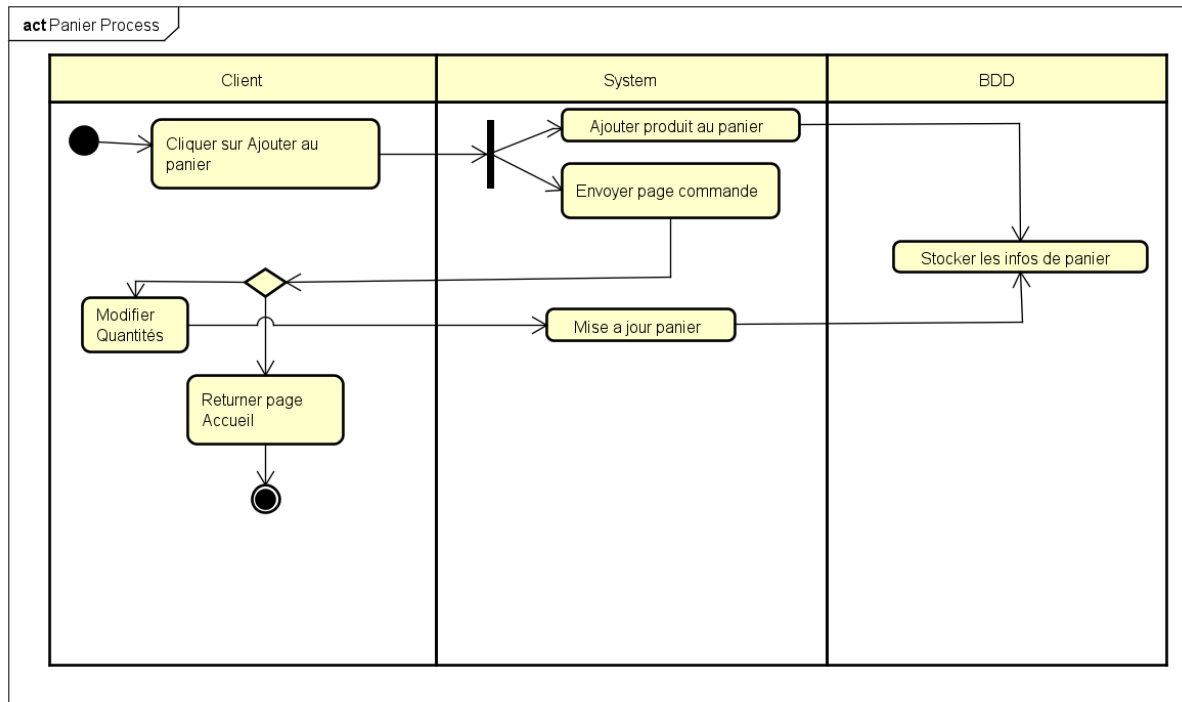


Figure 09: diagramme d'activité “processus panier”.

1.4.2 Diagramme d'activité de "processus du commande":

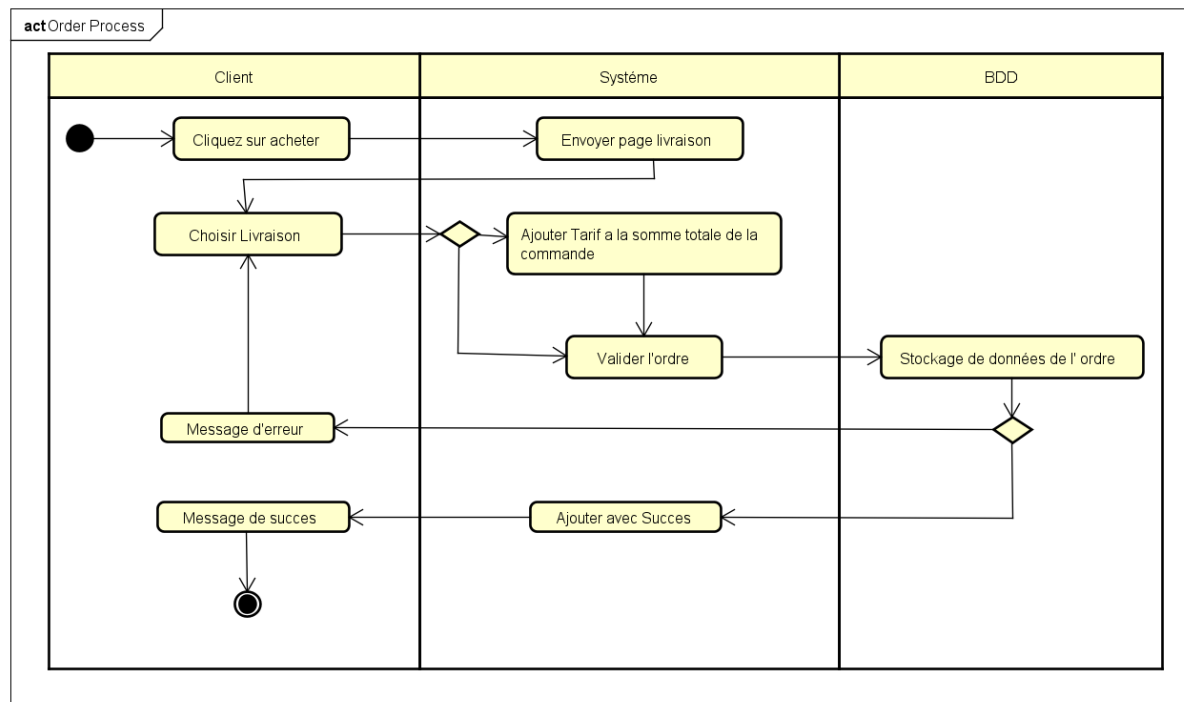


Figure 10: diagramme d'activité "processus commande".

1.5. Diagramme de déploiement:

Le diagramme de déploiement suivant montre comment l'interface d'utilisateur interagit avec le système en envoyant une requête HTTP au serveur de l'application.

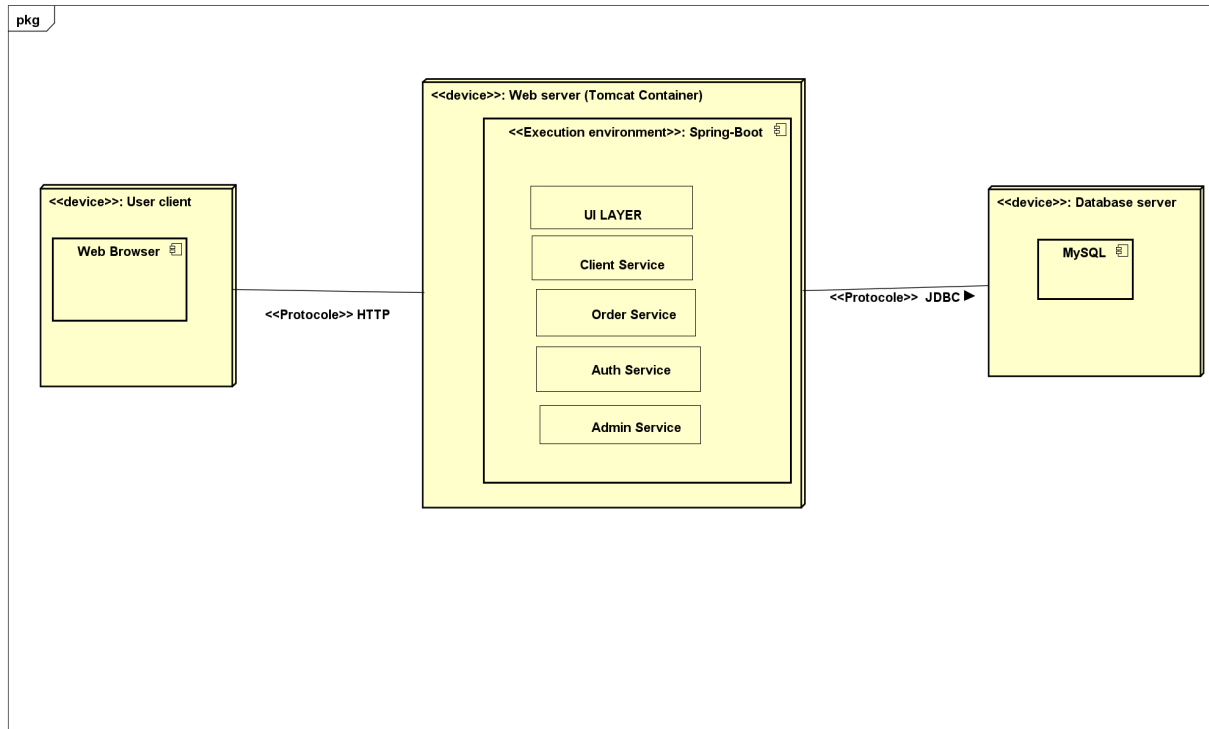


Figure 11 : diagramme du déploiement.

2. Conclusion:

Dans ce chapitre, on a identifié la partie d'analyse et conception et ses diagrammes UML de l'application E-meuble.

Chapitre III : Implémentation.

1. Les Principes d'implémentation:

Comme mentionné auparavant l'application est implémentée en se basant sur l'architecture des microservices et découpée en utilisant l'approche du BOUNDED CONTEXT et implémentée avec le framework SPRING BOOT.

1.1 EDGE Microservices:

Les **Edge Microservices** sont des microservices spécialisés dans l'orchestration des Microservices centraux responsables de la logique de l'application.

Les Edge Microservices les plus populaires sont ceux publiés et utilisés par Netflix pour sa plateforme. Ils sont en grande partie regroupés sous Spring Cloud et disposent de fonctionnalités pour fonctionner nativement ensemble.²

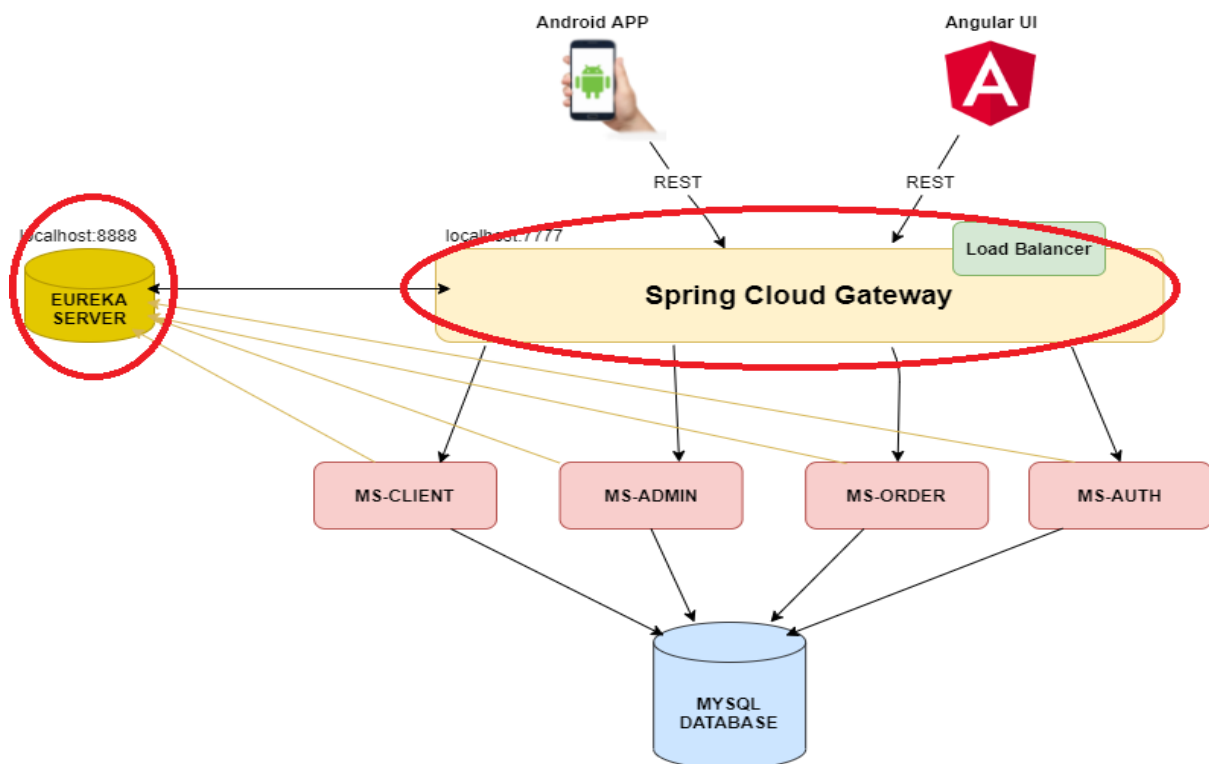


Figure 12 : l'architecture des microservices d'e-meuble

1.1.1 EUREKA SERVER: La découvrabilité:

En cas de grande charge sur notre application, nous souhaitons ajouter plusieurs instances d'un Microservice.

=> **Comment garder la trace des URL des différentes instances disponibles, ainsi que leurs états ?**

Pour répondre à ce problème, Eureka se propose en naming server. Grâce à une simple annotation, toute nouvelle instance de votre Microservice va être enregistrée auprès d'Eureka.

Notre client n'a plus qu'à consulter ce registre pour trouver les instances disponibles d'un Micro Service donné.

Eureka s'occupe également de vérifier régulièrement si chaque instance enregistrée est toujours disponible afin de mettre à jour son registre en éliminant celles qui n'existent plus.

1.1.1 SPRING CLOUD GATEWAY:

Le développement d'une telle application professionnelle d'e-commerce, la réalisation d'une commande en ligne par exemple reposera sur plusieurs microservices:

- Microservice d'authentification
- Microservice du client.
- Microservice d'order.

Donc il faut faire appel à tous ces Microservices. Cela implique de les identifier, de repérer leurs instances, de s'authentifier auprès de chacun d'entre eux pour accéder aux ressources, de transformer le résultat reçu pour qu'il soit adapté au type d'appareil, etc en gardant tout ça invisible pour le client.

=> **Comment résoudre cette complexité ?**

La solution est d'installer un point d'entrée unique vers les Microservices. C'est ce qu'on appelle une API Gateway. Ce micro service aide à relier et de passer les données d'un micro service à un autre car on a spécifié la route qui mène à chaque microservice.

```
public RouteLocator routes(RouteLocatorBuilder builder) {  
    return builder.routes()  
        .route(id: "user-service", r -> r.path( ...patterns: "/users/**")  
            .filters(f -> f.filter(filter))  
            .uri("lb://ms-client"))  
        .route(id: "auth-service", r -> r.path( ...patterns: "/auth/**")  
            .filters(f -> f.filter(filter))  
            .uri("lb://ms-auth"))  
        .route(id: "order-service", r -> r.path( ...patterns: "/order/**")  
            .filters(f -> f.filter(filter))  
            .uri("lb://ms-order"))  
        .route(id: "panier-service", r -> r.path( ...patterns: "/panier/**")  
            .filters(f -> f.filter(filter))  
            .uri("lb://ms-order"))  
        .build();  
}
```

Figure 13 : Fonction de la redirection des routes en Gateway.

En gateway on a également défini les routes qui nécessitent ou pas une authentification.

```
public class RouterValidator {  
    public static final List<String> openApiEndpoints = List.of(  
        "/auth/register",  
        "/auth/login",  
        "/auth/sendMailClient",  
        "/auth/forgotpsw",  
        "/auth/resetpsw",  
        "/users/login"  
    );  
  
    public Predicate<ServerHttpRequest> isSecured =  
        request -> openApiEndpoints  
            .stream()  
            .noneMatch(uri -> request.getURI().getPath().contains(uri));  
}
```

Figure 14 : Fonction de redirection des routes nécessitant ou pas une authentification.

1.2 Microservices:

1.2.1 MS-AUTH:

C'est le micro service responsable de l'authentification et la vérification d'éligibilité de l'utilisateur.

Son rôle principal est la génération du TOKEN ou jeton qui est utilisé pour désigner un identificateur de session pour identifier une session lors d'une communication réseau.

1.2.2 MS-CLIENT:

Toutes les fonctions qui concernent le client sont implémentés au sein de ce micro service comme :

- Affichage du catalogue des produits.
- L'inscription et la connexion qui communiquent avec le service d'authentification 'MS-AUTH'
- La fonction des commentaires.
- L'affichage et l'édition du profil.
- Filtrage des produits par catégorie.

1.2.3 MS-ORDER:

Ce service concerne toutes les fonctions responsables des commandes en ligne et de la livraison.

- Remplissage du panier.
- Commander en ligne.
- Choix de la livraison.

En communiquant bien sûr avec le service client.

1.2.4 MS-ADMIN:

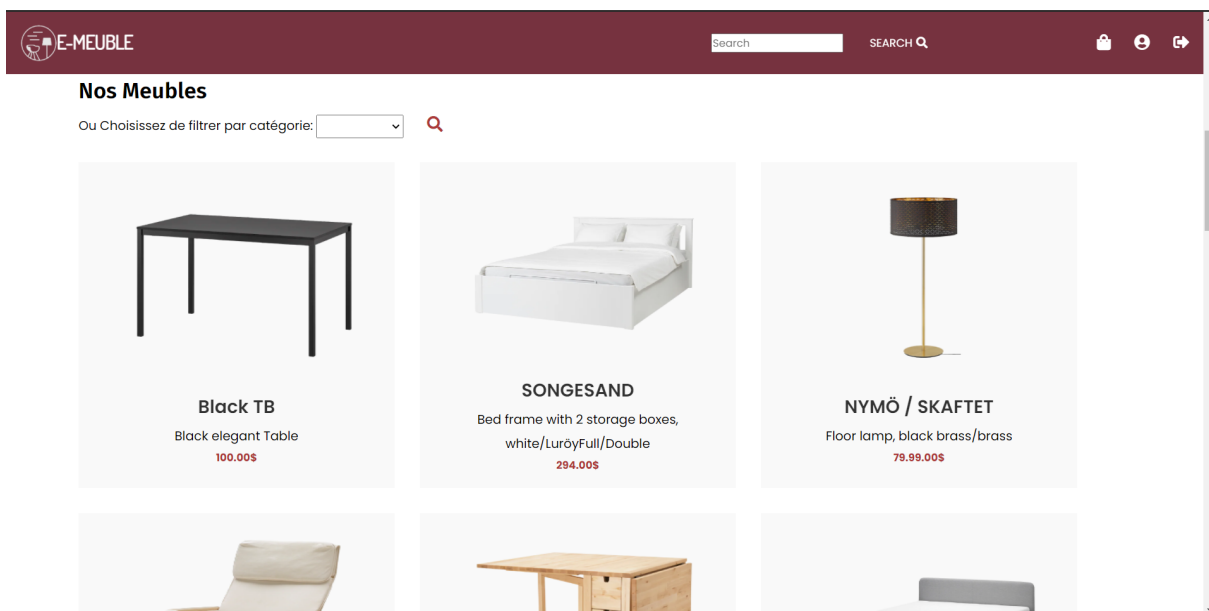
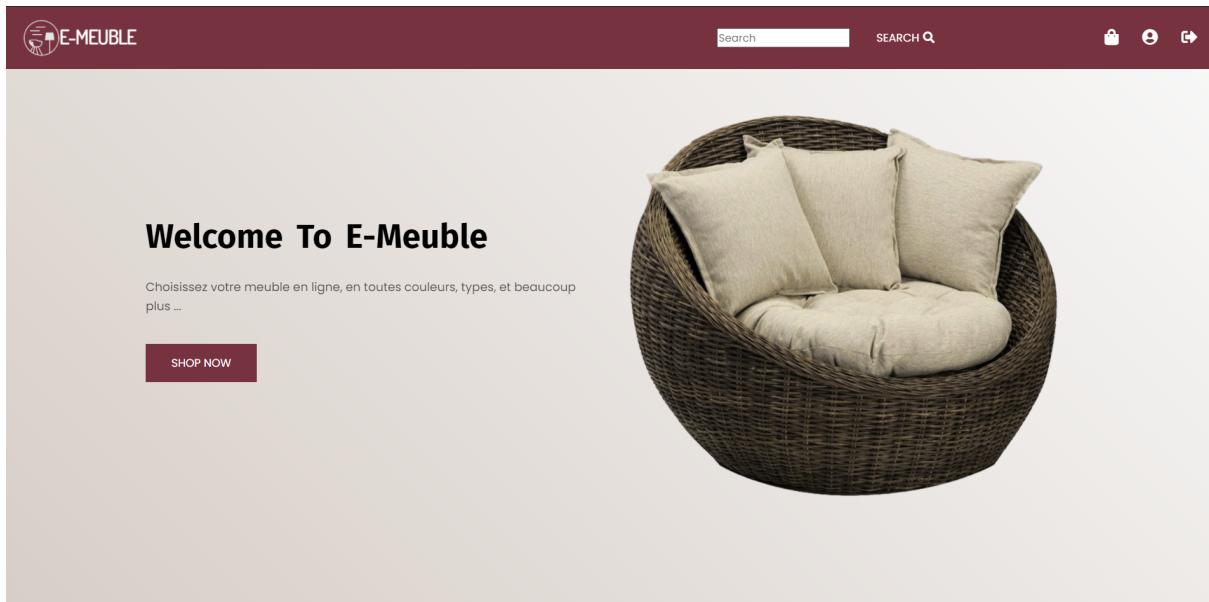
L'admin et le superadmin ont également leurs fonctions séparées en microservice.

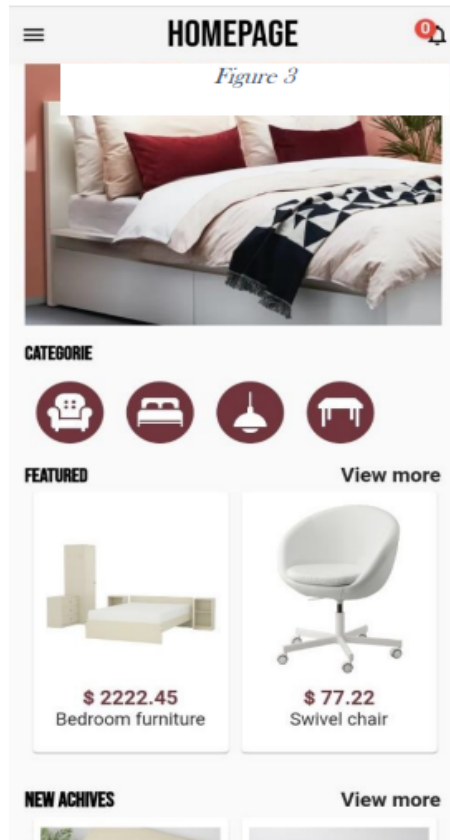
- Le super admin a comme fonctions: l'ajout des boutiques et l'admin qui la supervise.
- L'admin supervise la boutique, gère les clients, les produits, les commandes,etc

2. Les interfaces d'e-meuble:

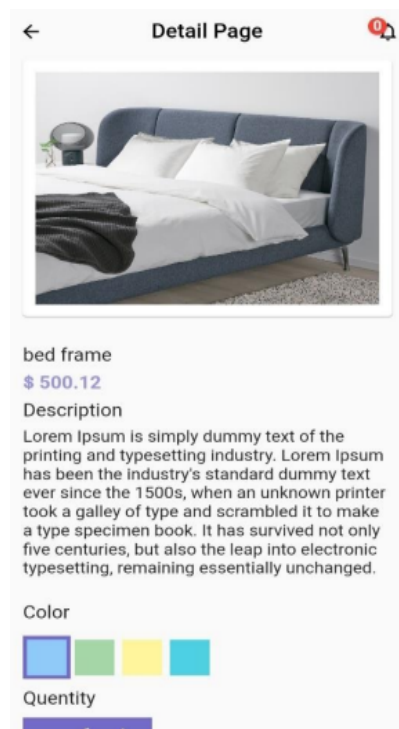
Quelques photos qui montrent les interfaces importantes de l'application web et mobile:

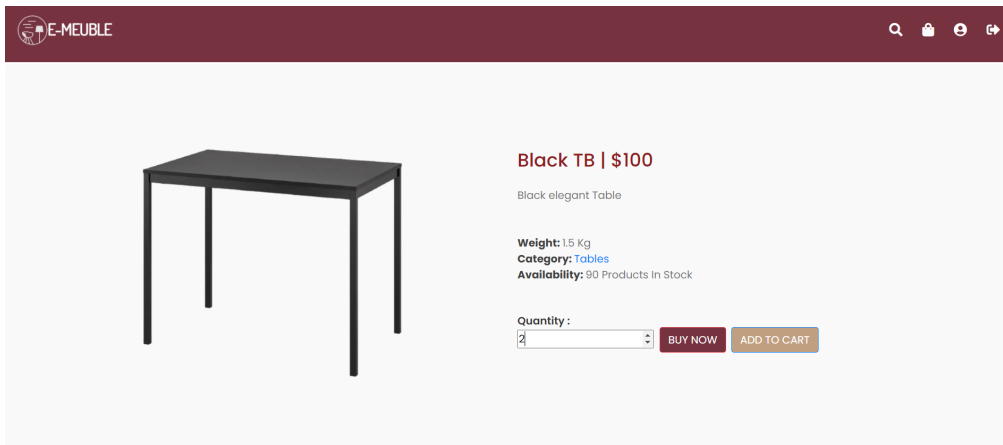
2.1 Page d'accueil:






2.1 Page détails du produit et commande:





2.1 Checkout Page (Panier):



Products	Price	Quantity	Total
 Black TB	100	2	\$200
CONTINUE SHOPPING		UPDATE CART	

Choose Delivery

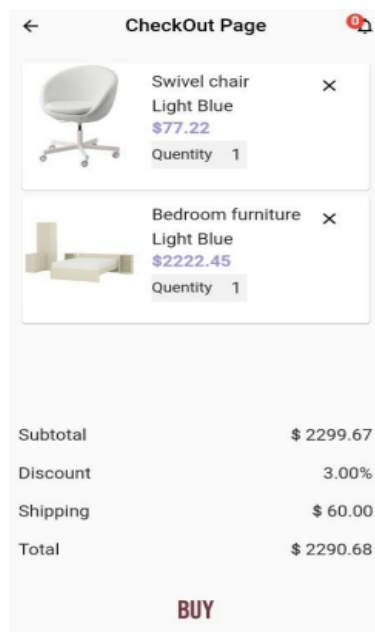
If you choose delivery extra price will be added to your total cost

Delivery price: 700 DA

Cart Total

Subtotal	200
Total	200

PROCEED TO CHECKOUT



3. Conclusion:

Dans ce chapitre on a expliqué les principes suivis pour implémenter notre application en se basant sur l'architecture des microservices, on a également exposé les interfaces importantes de notre application.

Chapitre IV : Les Outils de travail.

1. Langages et frameworks:

Pour l'application web on a utilisé le framework SPRING BOOT côté backend et Angular en frontend, et on a utilisé le kit de développement FLUTTER en langage DART, et pour la base de données MYSQL.

1.1 SPRING BOOT framework :

Spring Boot est un framework de développement applicatif JAVA open Source. Il est particulièrement recommandé pour le développement d'API et pour les applications basées sur l'architecture des microservices.



1.2 Angular:

Développé par Google, Angular est un Framework open source écrit en JavaScript qui permet la création d'applications Web et plus particulièrement de ce qu'on appelle des « Single Page Applications » : des applications web accessibles via une page web unique qui permet de fluidifier l'expérience utilisateur et d'éviter les chargements de pages à chaque nouvelle action. Le Framework est basé sur une architecture du type MVC et permet donc de séparer les données, le visuel et les actions pour une meilleure gestion des responsabilités. Un type d'architecture qui a largement fait ses preuves et qui permet une forte maintenabilité et une amélioration du travail collaboratif.



1.3 FLUTTER:

Flutter est un kit de développement de logiciel (SDK) d'interface utilisateur open-source créé par Google. Il est utilisé pour développer des applications pour Android, iOS, Linux, Mac, Windows, Google Fuchsia et le web à partir d'une seule base de code.



1.4 MYSQL:

MySQL est un système de gestion de bases de données relationnelles SQL open source développé et supporté par Oracle.



2. Outils de travail:

2.1 GITHUB:

A un haut niveau, GitHub est un site web et un service de cloud qui aide les développeurs à stocker et à gérer leur code, ainsi qu'à suivre et contrôler les modifications qui lui sont apportées.



2.2 TRELLO:

Inspiré par la méthode agile Kanban, Trello s'articule historiquement autour d'un tableau digital de gestion de projet permettant de répartir les tâches, sous forme de cartes, au sein de colonnes (ou listes dans le langage de Trello) se déclinant par exemple en "A faire", "En cours" et "Fait". ³



2.3 ASTAH UML:

C'est un logiciel pour la modélisation des différents diagrammes en langages UML, et il est disponible en version d'étudiant gratuite.



Conclusion générale:

Ce projet pluridisciplinaire consiste à concevoir une application web et mobile d'e-commerce du meuble en se basant sur l'architecture des microservices.

Pour concevoir ce travail nous avons présenté premièrement l'évolution du secteur du commerce en ligne, et comment on a divisé les services de notre application pour l'adapter selon l'architecture des microservices.

En second lieu nous avons réalisé les diagrammes d'analyse : Diagramme de cas d'utilisation, de séquence et d'activité en utilisant le langage UML, ainsi que les diagrammes de conception : Diagramme de composants et déploiement .

Par ailleurs, on a présenté les différentes pages réalisées de notre application web et mobile en expliquant leur implémentation.

Finalement, nous avons défini les langages et les logiciels utilisés pour le développement.