

Polypet : Rendu final



Valentin Campello, Yann Martin d'Escienne, Lucie Morant, Yohann Tognetti, Tigran Neressian

I. Analyse du besoin

Notre rôle principal sur le site de Polypet est tout d'abord de permettre l'affichage de tous les produits que vendent la société et ses partenaires commerciaux sous forme de catalogue. Le catalogue doit donc être un élément toujours disponible pour les clients. Il peut apparaître sous une forme non détaillée de chaque produit.

Lorsque le client clique sur un produit, il faut qu'une fiche détaillée apparaisse, contenant les différentes caractéristiques (prix, dimensions, ingrédients, description du produit, etc, par exemple), ce qui est pertinent lorsqu'il s'agit d'un produit pharmaceutique.

Il peut être possible d'avoir une section "Nouveaux produits" dans le catalogue afin qu'ils y mettent les nouveautés et que celles-ci soient mises en valeur par rapport aux produits présents depuis plus longtemps. Le critère de base serait la nouveauté du produit mais cela peut par la suite évoluer en prenant également en compte sa pertinence selon Polypet.

Le client aura la possibilité d'ajouter les différents produits qui lui plaisent à son panier et commander ainsi directement sur le site. Lorsqu'il souhaite finaliser sa commande, il faut qu'il rentre son adresse, nom, prénom, mail et sa carte bancaire.

Le paiement pourra se faire selon différentes solutions : la banque de Polypet propose différentes interfaces bancaires, le client n'aura donc qu'à choisir celle qui lui convient le mieux pour ensuite passer par elle pour effectuer son virement. Dans le cas d'une transaction bancaire en attente (mais pas refusée), Polypet souhaite avertir le client que sa commande lui sera confirmée après validation de la transaction, le délai de validation maximum donné par les banques étant d'une heure.

Une fois l'achat d'un produit effectué, le client doit pouvoir regarder le statut de sa commande ainsi que son récapitulatif avec ses informations de livraison. Pour cela, un client doit pouvoir s'identifier et se connecter à son espace personnel. De même les partenaires et employés de Polypet doivent se connecter à un espace membre afin d'accéder aux différents services du site réserver pour ceux-ci.

Si le produit provient de Polypet, l'entreprise délègue la livraison à une société de livraison par drone. Il faut donc que celle-ci puisse prévenir Polypet de la date de livraison estimée. En revanche, si le produit vient d'un partenaire et prend en charge la livraison, celui-ci doit pouvoir accéder aux informations de livraison du destinataire et de la commande afin de procéder à la livraison.

Il faut de plus pouvoir transmettre au service vente & marketing des informations sur les produits (prix, caractéristiques, s'il y a ou a eu des promotions) et permettre de les modifier.

L'équipe vente & marketing désire également les différentes activités enregistrées sur le site (le nombre de visiteurs total sur les différentes pages, quel est le produit le plus consulté, quel est le produit le plus acheté par les consommateurs). Cela passera par des statistiques.

Pour ajouter de leur nouveaux produits en ligne, les partenaires de Polypet doivent transmettre un catalogue des articles qu'ils souhaitent mettre en vente par l'intermédiaire de requêtes à l'équipe Marketing. L'équipe s'occupe ensuite de valider ou non les requêtes selon leur respect des critères de Polypet.

Le site internet doit être capable de gérer de nombreux clients simultanément, notamment sur l'achat et la consultation des détails des produits, en raison de la forte augmentation de la vente de produits pour les NAC. Un client qui ne peut pas payer ou accéder à ce qu'il veut sous les 2 minutes est un client perdu.

Il faut qu'en cas de problème, le site de Polypet ait un backup de chacune de ses bases de données afin de ne perdre aucune information sur les clients ou les produits. Le site doit être résilient et pouvoir fonctionner en mode dégradé dans le cas où un sous ensemble du système viendrait à tomber en panne.

Polypet souhaite proposer des codes promotions à ses clients (ou des réductions sur certains produits) afin de rendre ceux-ci plus attractifs. Une IA pour des propositions plus personnalisées pour le client est également un axe d'évolution de Polypet.

II. Personnas

Véronique, 45 ans, est entourée d'animaux depuis qu'elle est petite. Elle possède aujourd'hui un yorkshire, dont elle prend le plus grand soin. Celui-ci a souvent des problèmes de santé et doit consommer des croquettes spéciales, que Véronique trouve seulement chez les partenaires de Polypet. Mais elle n'a pas souvent le temps de se rendre en magasin et souhaiterait ainsi retrouver les produits-médicaments qu'elle souhaite pour son chien sur le site de Polypet, avec des descriptions des ingrédients pour son chien sensible et pouvoir les commander et les recevoir chez elle.

Jean, 30 ans, possède un chat. Celui-ci est très joueur et aime beaucoup découvrir de nouveaux jouets. Jean, qui ne peut pas lui résister, en cherche souvent. Mais, à force, il est difficile d'en trouver de nouveaux et de fouiller sur Internet. Pour faciliter cette recherche, Jean aimerait un site où il pourrait voir les nouveaux produits proposés pour les animaux et mettre ainsi sa main sur les nouveautés et rendre son chat heureux.

Charlotte, 35 ans, fait partie du service vente & marketing de Polypet. Pour comprendre les besoins de leurs clients et les produits qui leur plaisent le plus (et booster leurs ventes), elle souhaiterait avoir des statistiques récurrentes sur le(s) produit(s) le(s) plus consulté(s) et le(s) produit(s) le(s) plus acheté(s) par les consommateurs. De plus, pour savoir si leur site est attractif, elle souhaiterait recevoir le nombre de visiteurs que le site reçoit en une journée et d'autres statistiques pertinentes.

Jean-Baptiste est un partenaire industriel de Polypet. Il souhaite promouvoir ses produits à travers le nouveau site de Polypet afin de se faire connaître d'un maximum de clients. Il reçoit régulièrement de nouveaux produits qu'il mettra dès que possible en ligne. La mise en ligne rapide et la visibilité des nouveaux ajouts est un critère important pour lui.

III. User stories

(Dans le contexte du site internet de Polypet)

En tant que Véronique

Je veux consulter les détails des médicaments vendus par les partenaires de Polypet pour pouvoir ensuite les commander

Afin de choisir des croquettes spéciales pour mon chien qui a souvent des problèmes de santé.

En tant que Jean,

Je veux pouvoir consulter les nouveautés de Polypet, passer commande et avoir un récapitulatif de livraison,

Afin de faire découvrir de nouveaux jouets à mon chat et de vérifier que mon adresse de livraison est bien celle de chez moi.

En tant que Charlotte,

Je veux pouvoir avoir les différentes statistiques de notre site comme le nombre de visites total, le produit le plus consulté ect... ,

Afin de mieux promouvoir les prochaines nouveautés de Polypet et de concentrer mes efforts sur ce qui marche le mieux.

En tant que Jean-Baptiste,

Je veux pouvoir ajouter facilement mes nouveaux produits sur le site de Polypet et vérifier que ceux-ci sont bien visibles,

Afin de donner plus de visibilité à mes produits et faire connaître mon commerce à un maximum de clients.

IV. Scénarios principaux (MVP)

Quelqu'un achète un article mais le paiement est mis en attente, il reçoit la confirmation en différé

1. Je me connecte à mon espace marketplace
2. Je remarque que le panier est vide dans "Shopping Cart"
3. J'accède à la liste des produits disponibles dans "Catalog"
4. En sélectionnant un produit, je peux voir sa fiche détaillée
5. Je peux choisir une quantité de produit et l'ajouter à mon panier
6. Je visualise mon panier dans "Shopping Cart" et vois que les produits sont bien ajoutés
7. Je rentre mes infos de paiement et essaye de payer
8. Je peux voir que ma commande est payée sur "Order"
9. Le partenaire peut accéder aux informations de livraison sur "Partner Panel"
10. Le partenaire rentre la date de livraison sur la vue détaillée
11. Je vois le statut changer dans "Order" et la date de livraison apparaître
12. Je peux également accéder à un récapitulatif de la commande avec les informations de livraison

Je suis une société externe, je veux ajouter un article

1. Je me connecte à mon espace marketplace
2. Je fais une demande d'ajout d'un article avec les bons arguments (nom, prix ...)
3. Un membre de l'équipe PolyPet peut voir les demandes d'ajout de produit sur "Employee Panel"
4. Un membre de l'équipe PolyPet valide la requête d'ajout
5. On constate que l'article est bien ajouté sur "Catalog"
6. Je peux voir le produit dans la liste des nouveaux articles sur "PolyPet"

Quelqu'un achète un article mais le paiement est refusé

1. Je me connecte à mon espace marketplace
2. Je remarque que le panier est vide dans "Shopping Cart"
3. J'accède à la liste des produits disponibles dans "Catalog"
4. En sélectionnant un produit, je peux voir sa fiche détaillée
5. Je rajoute plus de produits dans mon panier que je ne peux en payer
6. Je visualise mon panier dans "Shopping Cart" et vois que les produits sont bien ajoutés
7. Je rentre mes infos de paiement et essaye de payer
8. Je peux voir que mon paiement est refusé sur "Order"

V. Scénarios secondaires

Je suis un membre de Polypet, je veux ajouter un article

1. Je me connecte à mon espace marketplace
2. J'ajoute un article avec les bons arguments (nom, prix, ...)
3. Je vérifie que je peux voir le produit dans la liste des articles de la section "nouveau"

Je suis un membre de l'équipe marketing de Polypet et je veux consulter les statistiques de notre site

1. Je me connecte à mon espace marketplace
2. Je récupère le nombre de connexion totale sur le site
3. Je le compare au nombre de connexions actuelles sur le site (client ou partenaire)
4. Je regarde le produit le plus consulté en ce moment
5. Je regarde si le produit le plus consulté est également le produit le plus acheté
6. Je vérifie si ce produit est présent dans la liste des articles de la section "nouveau" (pour savoir si la pub de ce nouveau produit a été efficace)

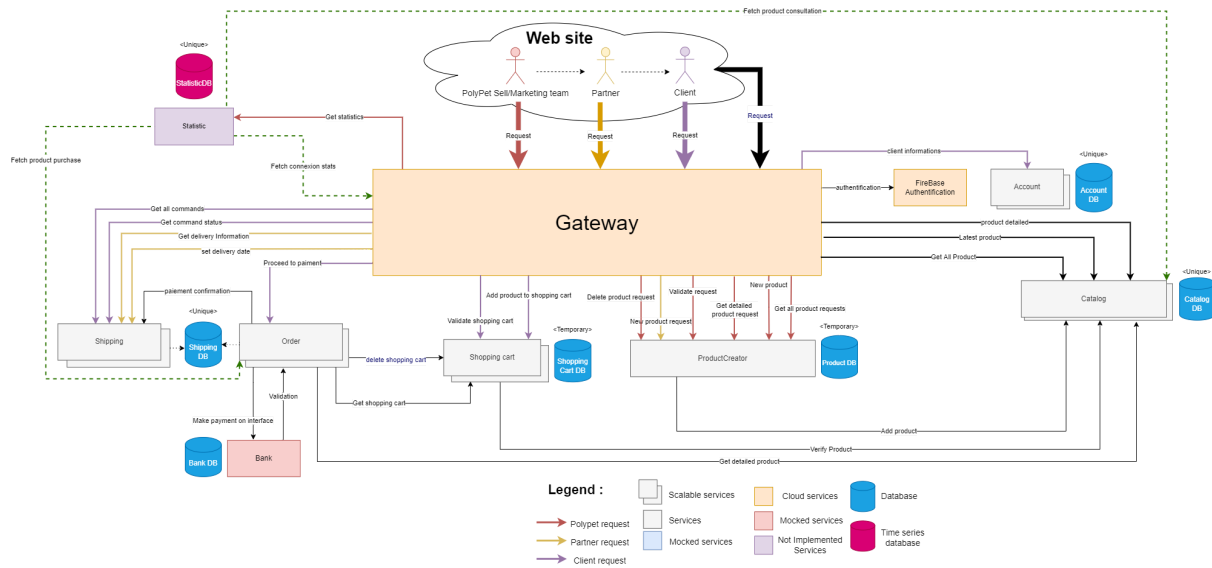
VI. Définition du périmètre MVP

- Pouvoir ajouter des articles dans un panier, passer à la commande des produits et procéder au paiement (avec les informations de livraison) ,
- Afficher des produits avec des fiches détaillées (prix, caractéristiques...),
- Afficher une section des nouveautés dans le catalogue du site,
- Pouvoir payer sur une interface bancaire de Polypet et gérer le cas d'une attente de confirmation de paiement (le panier n'est pas perdu si le paiement est refusé),
- Une fois la validation du paiement faite, pouvoir consulter sa commande effectuée avec un récapitulatif d'achat et informations de livraison,
- Faire en sorte qu'un partenaire puisse effectuer une demande d'ajout des produits au catalogue et que cela soit validé ou non par un employé de Polypet,
- Les employés de Polypet doivent pouvoir ajouter des produits directement,
- Les clients et partenaires se connectent au site (via création de compte ou compte Google) pour effectuer leurs actions,
- Les partenaires et employés de Polypet doivent pouvoir accéder aux informations de livraison des commandes.
- Les partenaires et employés de Polypet doivent pouvoir renseigner une date de livraison une fois la commande payée.

VII. Simplifications dans le MVP

- La gestion du paiement, confirmation, etc... ne sera pas directement géré par nous mais par un système bancaire ultra minimaliste (service bouchonné),
- Les nouveaux produits envoyés par les partenaires industriels et sociétés pharmaceutiques sont déjà dans le format utilisable par notre site,
- L'interface de notre site internet sera très simpliste et servira seulement à démontrer nos fonctionnalités,
- Même espace marketplace pour les clients, les partenaires et les employés,
- Pas d'envoi de confirmation de commande par mail aux clients, le client doit voir lui même l'état de sa commande,
- La base de donnée est commune à nos services (mais les tables sont différentes, ce qui permettra une séparation assez simple),
- Pas de gestion du stock des produits. Par exemple, lorsqu'un client achète un article, il n'est pas décompté du stock total,
- Il n'y aura pas de gestion d'ajout ou suppression des produits directement depuis le panier,
- Il n'y aura pas de gestion de média (image, son, vidéo) pour les description de nos produits,
- Il n'y aura pas de tri mis en place sur nos produits ou sur l'historique des commandes,
- L'implémentation des statistiques sera remise à plus tard, mais leur conception doit être faite,
- Nous ne gardons pas d'historiques des requêtes d'ajout de produit des partenaires,
- Actuellement le partenaire ne peut accéder qu'aux commandes qu'il a lui-même faites (il n'y a pas encore de différenciation de compte, il est considéré comme un client).

VIII. Diagramme d'architecture logicielle



(Pour plus de lisibilité, se référer à l'image dans le repository)

Notre diagramme d'architecture comporte plusieurs éléments :

1. Gateway

La gateway permet une implémentation rapide de la partie frontend sans se soucier de la structure des services côté serveur pour le client. Étant le seul point d'entrée, il est facile de monitorer et filtrer les différentes requêtes qui vont être faites ainsi que d'établir des statistiques. De plus, il nous permettrait plus tard de pouvoir rediriger les requêtes vers des services particuliers selon le pays en mettant en place des CDN ou bien même de restreindre l'accès en fonction du niveau de permission, avec par exemple des services accessibles seulement par des partenaires ou des employés de PolyPet.

2. Account

Ce service stocke les différents comptes ainsi que les informations clients (adresses mail, nom, prénoms). De même, il stocke les comptes des membres de Polypet et des partenaires. Lors de leur première inscription, il leur fournissait un ID mais à laisser cette fonctionnalité à Firebase. Par la suite, des informations complémentaires pourront être stockées pour chaque client, partenaire ou membre de Polypet.

NB: en raison de la non nécessité des informations clientes, ce service n'est pas forcément incorporé dans l'architecture de notre MVP. Il prend cependant son importance dans les évolutions futures de Polypet. Son modèle de données est d'ailleurs retravaillé pour sa nouvelle fonction (voir Modèle de données).

3. Product Creator

Ce service permet aux partenaires d'effectuer une demande d'ajout de leurs nouveaux produits au site internet de Polypet. L'équipe Marketing de Polypet pourra ensuite valider et ainsi

ajouter ceux-ci dans le catalogue. Cette validation a pour but d'assurer une cohérence dans la gamme des produits proposés. Il stocke temporairement les demandes d'ajout de produits dans sa base de données avant de les supprimer lorsque les demandes sont validées / refusées.

4. Catalog

Ce service stocke tous les produits actuellement présents dans le catalogue du site. Il fournit les détails des produits ainsi que la liste des nouveaux articles. Pour ne pas surcharger les navigateurs des clients, il envoie tout d'abord les produits sous forme non détaillée (nom, prix, description). Il dispose de plusieurs instances pour supporter la charge mais celles-ci sont connectées sur la même base de données CatalogDB afin de garder la cohérence des données des produits malgré les différentes instances. Il s'avère donc être le service le plus sollicité et le plus critique de l'architecture. Il est le point de départ de l'échange entre nos services car il fournit les informations pour le service Order et permet de vérifier la validité des produits pour Shopping Cart. Il participe évidemment à l'enrichissement des statistiques concernant le nombre de consultations des différents produits.

5. Shopping Cart

Ce service représente un panier de commandes. Il dispose également de plusieurs instances pour supporter la charge. Il utilise une base de données qui stocke temporairement les données simplifiées des produits et du panier avant de les transmettre au service Order. Une fois le panier payé et validé, le service Order demande à Shopping Cart de supprimer les informations du panier.

6. Order

Ce service permet de procéder au paiement. Pour ce faire, il récupère le panier auprès de Shopping Cart ainsi que les prix et quantités de chaque produit qu'il va alors stocker dans une base de données qu'il partage avec Shipping. Le client lui envoie également les informations de livraison et de paiement qu'il transmet ensuite à la banque.

- En cas d'échec de paiement, il va proposer au client de renseigner de nouveau les informations en marquant la commande avec le statut "Paiement refusé".
- En cas de succès, il va alors prévenir le service Shipping afin que celui prenne le relais et va demander à Shopping Cart de supprimer ce panier de sa base de données.
- En cas de réponse délayée, il stocke l'état de la commande comme étant "En attente de paiement". Le client va pouvoir consulter l'état de la commande et voir son évolution depuis le service Shipping.

Il partage donc la base de données avec le service Shipping vu qu'il se base sur le même modèle métier. Il peut lui aussi posséder plusieurs instances afin de gérer la charge. Cela est d'autant plus important pour ce service car un échec de paiement ou une impossibilité d'accéder au service peut notamment inciter le client à se rendre chez un commerce concurrent. Pour des raisons de sécurité et de protection des données, les informations de paiement ne sont pas mémorisées en base de données. Il participe à l'enrichissement des statistiques notamment sur le nombre de produits achetés.

7. Shipping

Ce service gère la confirmation des commandes données par Order ainsi que le statut des commandes client. Il s'occupe également de fournir les informations de livraison. Celles-ci sont utilisées par les partenaires pour effectuer l'expédition de leurs produits aux clients. C'est également par ce service que les partenaires vont informer Polypet de la date estimée de livraison. Il peut posséder plusieurs instances afin de gérer la charge et d'être toujours disponible.

8. Statistic

Ce service gère les statistiques du site. Il reçoit la plupart de ces données par la Gateway lors d'une nouvelle connexion ou consultation mais également des services Order et Catalog. Il stocke les statistiques dans sa propre base de données et les traite afin d'en construire de plus complexes comme le top 10 des produits les plus consultés, les catégories de produits les plus consultées en fonction de temps, etc...

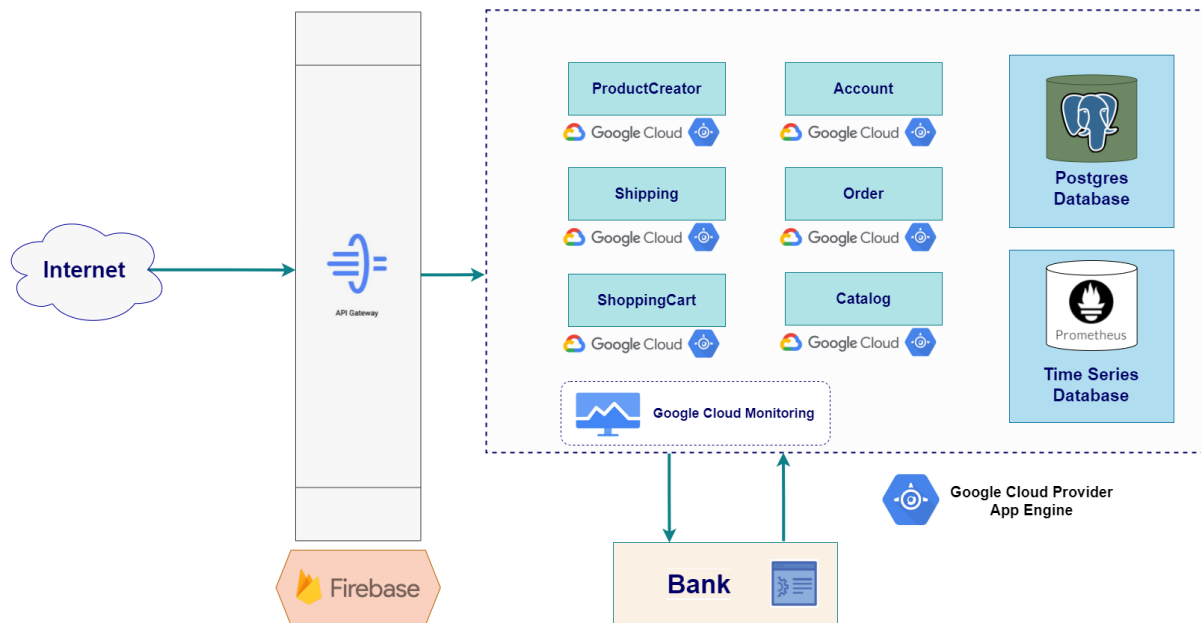
9. Bank

Ce service est bouchonné et imite le comportement de l'interface d'une banque. Il nous permet de vérifier si les informations bancaires du client sont bonnes et, si elles le sont, d'autoriser sa transaction. Sa base de données nous permet de stocker les données bancaires bouchonnées d'un client, que l'on enregistre avant à l'aide de ses différentes routes. Il effectue tout de même une vérification basique sur la transaction bancaire et répond en fonction de la possibilité de l'achat.

10. Firebase Authentication

Son utilité est de permettre à nos différents acteurs de s'identifier et s'authentifier sur notre site à l'aide d'un identifiant et d'un mot de passe (ou d'un compte Google). Sans compte, il n'est pas possible d'avoir un panier et d'y ajouter des produits présents sur Polypet mais nous autorisons cependant la consultation du catalogue et de ses différents produits. Il marche évidemment avec la Gateway.

IX. Diagramme d'infrastructure cloud et choix technologiques



Le provider cloud dans ce projet n'était pas au choix, nous étions dans l'obligation de choisir **GCP** (Google Cloud Provider).

Nous avons donc décidé de partir sur **App Engine de GCP**, qui est donc du PaaS (Platform as a Service), pour la mise en place de nos services sur le Cloud. Nous avons choisi de prendre du PaaS pour diminuer le coût en ressources humaines dans tout ce qui est déploiement de la solution sur le cloud, quitte à payer un peu plus cher qu'avec du CaaS (Container as a Service). De plus, nous avons choisi comme technologie Node JS (Framework Nest JS) 14, qui est une version récente et devrait donc être maintenue et disponible sur GCP encore un long moment.

Durant notre développement, nous avions un budget de 250\$. Nous sommes donc partis sur du F1 avec le paramétrage par défaut (ce qui revient ici au moins cher possible). Dans la mise en place définitive de notre architecture sur le cloud, nous envisageons de garder certains services en F1 comme Shipping, Account, Order et Product Creator en raison de leur faible coût en performance (voir la partie Coût Prévisionnel). Mais le service Catalog va lui passer en F4 en raison de sa monopolisation en ressources constantes. En effet, n'importe qui peut accéder au catalogue et celui-ci est le point central du site.

Nous avons choisi d'utiliser la gateway intégrée de Google : API Gateway pour des raisons de simplicité d'intégration et prise en charge. Elle possède également une HA (High Availability) ainsi qu'une résilience élevée. Couplée à FireBase, également intégré à la gateway et servant à faire de l'authentification, il nous est donc facilement possible de mettre en place un système de compte et de connexion pour les clients de Polypet ainsi que les partenaires et employés.

Pour la base de données, nous avons voulu rester sur un modèle relationnel avec Postgresql. Une base de données relationnelle nous permet de retrouver facilement et rapidement un utilisateur ou autre avec son ID.

Pour les statistiques de notre site, nous avons choisi l'utilisation d'un Time Series Database avec Prometheus. Cela permet de récupérer périodiquement des données auprès de nos services et d'en construire de plus complexes.

Les données seront ensuite traitées dans le service Stats avec Grafana afin de faciliter la visualisation de données. La gateway nous permet également de récupérer des statistiques plus générales sur le site comme le nombre de connexion, la localisation des clients, etc...

La partie Monitoring du cloud est détaillée dans le chapitre *XII. Présentation de la supervision du système*.

NB: Le service bouchonné Bank est lui aussi sur App Engine en F1. Il n'est là que dans le cadre du POC.

X. Modèles de données

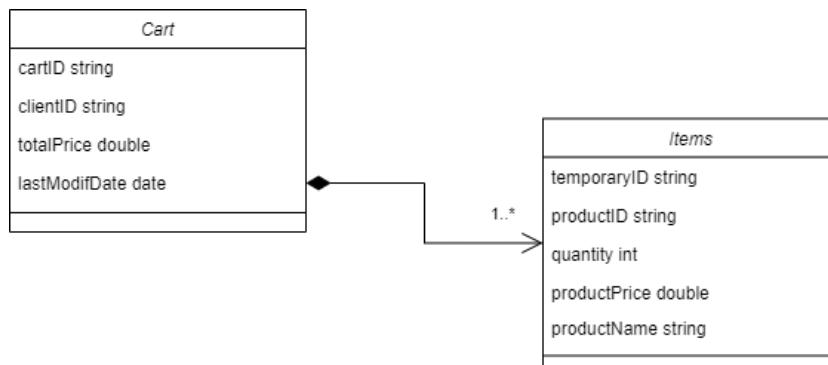
Nous sommes partis sur plusieurs bases de données relationnelles en raison des différentes liaisons entre les données de nos services.

AccountDB																						
<div><div><table><tr><th>Customer</th></tr><tr><td>id string</td></tr><tr><td>name string</td></tr><tr><td>surname string</td></tr><tr><td>address string</td></tr><tr><td>mail string</td></tr><tr><td></td></tr></table></div><div><table><tr><th>Account</th></tr><tr><td>username string</td></tr><tr><td>id string</td></tr><tr><td>password string</td></tr><tr><td></td></tr></table></div><div><table><tr><th>Employee</th></tr><tr><td>id string</td></tr><tr><td>name string</td></tr><tr><td>surname string</td></tr><tr><td></td></tr></table></div><div><table><tr><th>Partner</th></tr><tr><td>id string</td></tr><tr><td>name string</td></tr><tr><td></td></tr></table></div></div>		Customer	id string	name string	surname string	address string	mail string		Account	username string	id string	password string		Employee	id string	name string	surname string		Partner	id string	name string	
Customer																						
id string																						
name string																						
surname string																						
address string																						
mail string																						
Account																						
username string																						
id string																						
password string																						
Employee																						
id string																						
name string																						
surname string																						
Partner																						
id string																						
name string																						
<p>AccountDB contient 4 tables différentes afin de différencier les acteurs qui accèdent à notre site. Tous les regrouper dans une même table nous semblait être compliqué et n’apportait pas la précision que l’on souhaitait. De plus, faire de cette façon aurait signifié que, lorsque nous essayons de retrouver les informations d’un client, nous aurions dû à chaque requête vérifier l’id de toutes les personnes, même si celles-ci n’étaient pas catégorisées comme étant un client, ce qui n’est donc pas cohérent au niveau des données. La table Account nous permet d’avoir une trace d’un compte d’un tiers et l’id qui y est stocké est celui de la personne concernée.</p>																						
ProductDB	CatalogDB																					
<div><table><tr><th>ProductRequest</th></tr><tr><td>id number</td></tr><tr><td>name string</td></tr><tr><td>price double</td></tr><tr><td>category string</td></tr><tr><td>description string</td></tr><tr><td>partner_id string</td></tr><tr><td>ingredient string</td></tr><tr><td>dimension string</td></tr><tr><td></td></tr></table></div>	ProductRequest	id number	name string	price double	category string	description string	partner_id string	ingredient string	dimension string		<div><table><tr><th>Product</th></tr><tr><td>product_id string</td></tr><tr><td>name string</td></tr><tr><td>price double</td></tr><tr><td>category string</td></tr><tr><td>description string</td></tr><tr><td>addedDate date</td></tr><tr><td>partner_id string</td></tr><tr><td>ingredient string</td></tr><tr><td>dimension string</td></tr><tr><td></td></tr></table></div>	Product	product_id string	name string	price double	category string	description string	addedDate date	partner_id string	ingredient string	dimension string	
ProductRequest																						
id number																						
name string																						
price double																						
category string																						
description string																						
partner_id string																						
ingredient string																						
dimension string																						
Product																						
product_id string																						
name string																						
price double																						
category string																						
description string																						
addedDate date																						
partner_id string																						
ingredient string																						
dimension string																						
<p>Nous avons dans cette base de données différentes informations sur un produit qui souhaite être ajouté au catalogue. Ici, l’id n’est pas celui qu’il aurait</p>	<p>CatalogDB contient tous les produits achetables sur Polypet et les informations à leur sujet qui sont intéressantes pour l’utilisateur : nom, prix, catégorie, description, ingrédients... Le champ partner_id est</p>																					

potentiellement dans le catalogue mais simplement un id interne. Il ne sert qu'à accéder rapidement à la donnée si une requête de validation ou de suppression est faite. Lorsqu'un produit est validé, il est envoyé au catalogue et supprimé de la base de données.

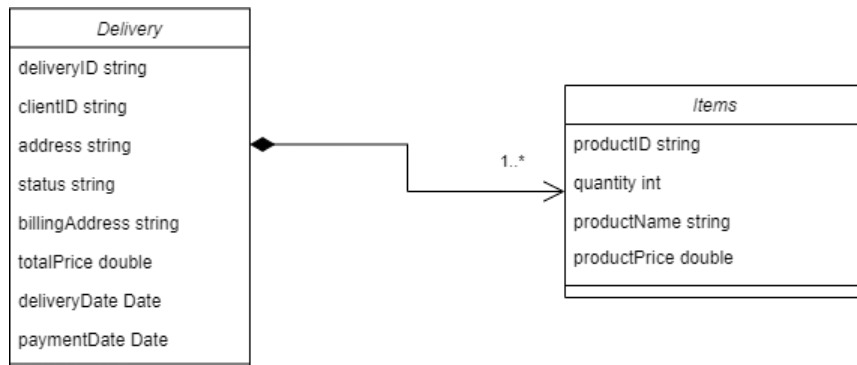
mal nommé et devrait plutôt s'appeler "partner_name" car on y stocke le nom de l'entreprise vendant le produit, plus que son identifiant.

ShoppingCartDB



ShoppingCartDB contient toutes les informations relatives au panier d'un client. Chaque panier peut contenir un ou plusieurs produits présents dans le catalogue de Polypet. Les produits présentés dans la table Items sont dans un format simplifié ne contenant que les informations utiles au panier.

ShippingDB



ShippingDB a comme fonction de stocker les commandes faites à travers Polypet et de pouvoir suivre leur statut : si celles-ci ont eu un paiement accepté ou refusé, si elles sont en attente ou en cours de livraison... Nous ne gérons pas la livraison mais il est possible qu'un employé rentre la date de livraison d'un colis afin que le client soit au courant que sa commande a été prise en charge. Cette base de données est partagée entre les services Shipping et Order car ceux-ci partagent le même modèle de données.

StatisticsDB

Voici la liste des différentes métriques métier qui seront relevées dans les statistiques :

- Nombre de connexion au site

- Nombre d'appels de la route pour voir un produit détaillé (par produit)
- Nombre d'appels sur la route pour acheter un produit (par l'intermédiaire du panier)
- Localisation des utilisateurs utilisant le site
- Prix moyen des paniers achetés (à l'aide de la time serie database)
- Nombre de produits vendus par partenaire
- Prix moyen des produits vendus par partenaire
- Moyenne de temps de livraison par partenaire
- Nombre de connexion moyenne des clients par semaine
- Nombre de requêtes de produits faites par mois
- ...

BankDB

Card
id string
cardID string
account string
amount double

Cette base de données est liée à notre service Bank, qui est bouchonné. Elle ne contient donc que les informations que l'on souhaite pour notre site afin de mocker un paiement lors de la validation d'un panier. Au départ, celle-ci n'existait pas et cela nous posait donc des problèmes lorsque nous voulions tester car la donnée restait exploitable qu'une minute.

XI. Coûts prévisionnels

Pour les App Engine (PaaS)

Pour le développement nous utilisons un App Engine F1. Mais pour la production et la mise en place du site, nous penchons plus sur l'utilisation de F1 pour les services moins sollicités et en F4 pour le service Catalog. Il sera possible par la suite de passer à des F2 pour les services en F1 si le succès de Polypet retentit. Actuellement, 25h par mois sont offertes sur les F1, ce qui est avantageux.

Nous estimons que le service Catalog mis à l'échelle utilise 2 instances par heure en moyenne. Nous arrivons donc au coût suivant :

App Engine standard environment instances		
Belgium		
Instance Type: F4		
Instance Hours: 5,840 per month		
EUR 214.69		
Belgium		
Instance Type: F1		
Instance Hours: 2,920 per month		
EUR 89.02		

Pour la Gateway

Il ne semble pas y avoir de coût pour la gateway dans le calculateur utilisé. Mais entre 0 et 2 millions d'appels par mois, la gateway de google est gratuite. Nous estimons donc être en dessous de ce seuil.

Nombre d'appels d'API par mois et par compte de facturation	Coût par million d'appels d'API
Entre 0 et 2 millions	0,00 \$
Entre 2 millions et 1 milliard	3,00 \$
Plus de 1 milliard	1,50 \$

Pour la partie Firestore (Firebase)

Nous estimons 10 000 logins par jour, 500 inscriptions, 20 désinscriptions. Nous sommes donc en dessous de la limite des 50 000 lecture, 20 000 ecriture et 20 000 suppressions de documents et il n'y a pas de frais.

Firestore







A portion of your estimate fits within the Firestore free tier. Please [click here](#) for more detail.

Pour les bases de données Postgresql

La base la plus importante est celle du service Catalog, d'où le fait que celle-ci est plus de stockage.

La base de données du service Product Creator avec laquelle les employés et partenaires interagissent ne sera disponible que 12h par jour étant donné des heures de travail (modulo les fuseaux horaires dans la zone ciblée) afin de réduire les coûts.

Pour les bases de données, nous arrivons donc au total de frais suivant :

CATALOG		
# of instances: 1		
Instance type: db-standard-2		
Commitment term: 1 Year		
Location: Iowa		
730.0 total hours per month		
SSD Storage: 50.0 GiB		
Backup: 200.0 GiB		
EUR 84.76		
PRODUCT CREATOR		
# of instances: 1		
Instance type: db-lightweight-1		
Commitment term: 1 Year		
Location: Belgium		
730.0 total hours per month		
SSD Storage: 10.0 GiB		
Backup: 0.0 GiB		
EUR 33.30		
ORDER + SHIPPING		
# of instances: 1		
Instance type: db-lightweight-2		
Commitment term: 1 Year		
Location: Belgium		
730.0 total hours per month		
SSD Storage: 20.0 GiB		
Backup: 20.0 GiB		
EUR 55.61		
Total Estimated Cost: EUR 477.37 per 1 month		

Ainsi, le coût mensuel total du cloud pour le site de Polypet sur **une zone donnée** s'élève donc à **477,37 \$ par mois** chez **Google Cloud Provider**, soit **5 728,44 \$ à l'année**. Ici, nous nous sommes basés sur des data center situés en Belgique, le coût peut donc varier d'une zone à l'autre.

Après une estimation similaire des coûts chez AWS, nous arrivons au résultat suivant :

Services (5)		
Amazon EC2 Description: other Region: EU (Paris)	Edit	Action ▼
Quick estimate		
Operating system (Linux), Quantity (4), Pricing strategy (EC2 Instance Savings Plans 1 Year No Upfront), Storage amount (10 GB), Instance type (t4g.small)		Monthly: 39.39 USD
Amazon EC2 Region: EU (Paris)	Edit	Action ▼
Quick estimate		
Operating system (Linux), Quantity (2), Pricing strategy (EC2 Instance Savings Plans 1 Year No Upfront), Storage amount (10 GB), Instance type (t4g.xlarge)		Monthly: 140.87 USD
Amazon Aurora PostgreSQL-Compatible DB Region: EU (Paris)	Edit	Action ▼
Aurora PostgreSQL-Compatible Edition		
Quantity (1), Instance type (db.t3.medium), Pricing strategy (OnDemand), Storage amount (50 GB)		Monthly: 78.66 USD
Amazon Aurora PostgreSQL-Compatible DB Description: catalog + order + product creator Region: EU (Paris)	Edit	Action ▼
Aurora PostgreSQL-Compatible Edition		
Quantity (2), Instance type (db.t3.medium), Pricing strategy (OnDemand), Storage amount (20 GB)		Monthly: 153.71 USD
Amazon API Gateway Description: auth gateway Region: EU (Paris)	Edit	Action ▼
HTTP API requests units (millions), Average size of each request (500 KB), REST API request units (millions), Cache memory size (GB) (None), WebSocket message units (thousands), Average message size (32 KB), Requests (2 per month)		Monthly: 2.34 USD
Estimate summary Info		
Upfront cost 0.00 USD	Monthly cost 414.97 USD	Total 12 months cost 4,979.64 USD

Le prix total est donc de **414.97 \$ par mois**, soit **4979.67 \$ par an**. Il faut également noter que sur le papier, AWS propose de meilleures performances techniques que GCP, notamment sur les bases de données, pour un prix inférieur.

Que ce soit GCP ou AWS, nous sommes partis sur une durée d'engagement de 1 an afin de réduire les coûts. Il ne nous semblait pas pertinent de ne pas s'engager du tout ou de s'engager sur une plus grande période aux vues de l'évolution de Polypet encore incertaine.

XII. Supervision du système

Concernant la supervision de notre système, nous avons décidé de partir sur de la supervision active de nos services car nous n'en possédons pas beaucoup pour le site de Polypet. Pour cela, nous avons donc prioriser DataDog (<https://www.datadoghq.com/about/leadership/>) pour des raisons de facilité d'intégration avec le reste de notre architecture cloud GCP. Après configuration de DataDog sur nos App Engine, la solution offre une mise en place très facile de test de disponibilité ainsi que des alertes et notifications. Ces mêmes alertes pourront notifier une potentielle équipe de maintenance d'astreinte en cas de problèmes hors horaires de travail.

Il est ensuite possible de créer un tableau de bord personnalisable avec de nombreux graphiques. Cela convient parfaitement à nos besoins car le faible nombre de services permet de créer un tableau de bord qui affiche simplement l'état de chacune des instances de nos services d'un simple coup d'œil.

Datadog permet également une récupération des logs de nos services, mais ne nous semble pas pertinent dans le cadre du site Polypet actuellement, bien qu'il peut s'avérer utile pour le futur.

Il ne nous a pas été possible d'estimer le coût de cette solution mais dans le cas d'un tarif hors de la portée de polypet, il nous est toujours possible d'utiliser le monitor offert directement par GCP, bien que celui-ci soit plus difficilement exploitable en l'état. Il fournit lui aussi les logs de nos services ainsi que le nombre d'instances actuellement utilisées :

Charge actuelle					
URI	Requêtes/minute actuellement	Requêtes dernières 24 heures	Mégacycles d'exécution dernière heure	Latence moyenne dernière heure	Traces dernières 24 heures
/shopping-cart/cart	1,6	65	1 818	2 346,121 ms	Afficher les traces
/catalog/get-all-products	0,4	51	545	535,409 ms	Afficher les traces
/shopping-cart/product	0,4	51	759	1 042,862 ms	Afficher les traces
/product-request/all-product-requests	0,4	50	2 000	2 940,833 ms	Afficher les traces
/catalog/get-detailed-product	0,4	46	105	119 ms	Afficher les traces
/client-command	0	42	1 429	974,429 ms	Afficher les traces
/catalog/get-latest-products	0	28	615	754,231 ms	Afficher les traces
/catalog/verify-product	0	9	0	126 ms	Afficher les traces
/get-command-status	0	8	0	ms	Afficher les traces
/favicon.ico	0	8	0	59,857 ms	Afficher les traces
/product-request/delete-product-request	0	7	0	ms	Afficher les traces
/delivery-information	0	6	0	ms	Afficher les traces
/balance	0	6	3 000	3 570,75 ms	Afficher les traces
/	0	6	0	89,6 ms	Afficher les traces
/runtime.a00a55635868ef99.js	0	5	0	60,4 ms	Afficher les traces
/order/proceed-to-payment	0	4	0	ms	Afficher les traces
/main.25fdb2f0de4fbc3.js	0	4	0	0 ms	Afficher les traces
/product-request/add-new-product	0	4	0	ms	Afficher les traces
/catalog/add-product	0	3	0	ms	Afficher les traces
/polyfills.0ede5bb3e80d2e5f.js	0	3	0	0 ms	Afficher les traces

Lignes par page: 50 ▼ 1 – 20 sur 20 < >

XIII. Evolutions envisagées dans notre architecture

Notre projet devant être MVP, nous avons décidé de faire certaines simplifications par question de temps. Nous avons néanmoins des idées des différentes évolutions, ainsi que la résolution des problèmes que nous avons envisagées.

Tout d'abord, au niveau du service Catalog, les premiers problèmes qui risquent d'arriver sont un nombre de requêtes élevées pour consulter les différents produits. Cela s'explique notamment par un problème de centralisation des requêtes. Il serait donc intéressant de mettre en place un CDN afin de séparer les requêtes par zone géographique. Pour ce faire, il nous suffit d'ajouter un service global qui aurait une base de données centralisant tous les produits de Polypet. Les différents articles seraient ensuite récupérés dans cette base et dupliqués dans les différents services Catalog de chaque zone afin de réduire la latence. Cela permet également de gérer le cas de produits présents dans certaines régions du monde, il suffit alors de ne pas les récupérer de la base de données centrale.

Par la suite, nous envisageons aussi de continuer notre travail sur l'authentification des utilisateurs, pour que ceux-ci n'aient accès qu'aux parties de l'application qui leurs sont autorisées. Cette amélioration pourrait aussi nous permettre d'améliorer notre IHM en ne faisant apparaître que les onglets utilisables par les types d'utilisateurs (clients, partenaires, employés).

Pour des raisons de sécurité, il nous sera nécessaire de bloquer l'accès aux services pour les requêtes ne passant pas par la gateway et le service d'authentification. Ceci est configurable sur le tableau de bord de App Engine.

Il nous est également possible d'utiliser un Secure Web Gateway afin de vérifier encore plus la légitimité des appels moyennant un coût d'entretien plus élevé.

Comme dit précédemment, nous pouvons par la suite aussi continuer d'améliorer notre IHM, pour qu'elle devienne autre chose qu'un simple outil de démonstration avec la possibilité de se déconnecter par un onglet. Cela devra s'accompagner d'ajouts de fonctionnalités métiers au niveau de nos services sur ce que nous avons négligé lors de notre MVP, par exemple sur la gestion des objets une fois que ceux-ci sont dans le panier, la gestion des stocks de notre catalogue, etc... Il faudra bien sûr donner aux partenaires l'accès à toutes les commandes clientes et non seulement les leurs (voir simplification) et de n'y afficher que les commandes payées.

Dans les fonctionnalités métiers, il faudra que nous nous penchions sur l'implémentation des statistiques. Nous avons mis cette partie de côté car la prise en main de la technologie que nous voulions utiliser nous aurait pris trop de temps et nous aurait empêché de développer des parties plus importantes pour ce MVP. En effet, nous avons choisi d'utiliser Prometheus pour implémenter notre récolte de statistique, et aucun de nous n'avait d'expérience dans l'utilisation de Prometheus ou même dans les Time Series databases. L'implémentation des statistiques serait également accompagnée des données offertes par la gateway de GCP. A l'aide de Grafana, les statistiques seraient alors présentées visuellement sous forme de graphiques regroupant les données.

Il serait intéressant pour Polypet d'ajouter un service avec une IA s'alimentant des données des statistiques afin de proposer du contenu personnalisé aux clients (à l'aide des cookies également)

ainsi que des promotions qui seraient ensuite envoyées par mail au client. Ce service aurait potentiellement une instance cloud réservée car constamment monopolisée avec une consommation CPU et RAM importante de part ses calculs.

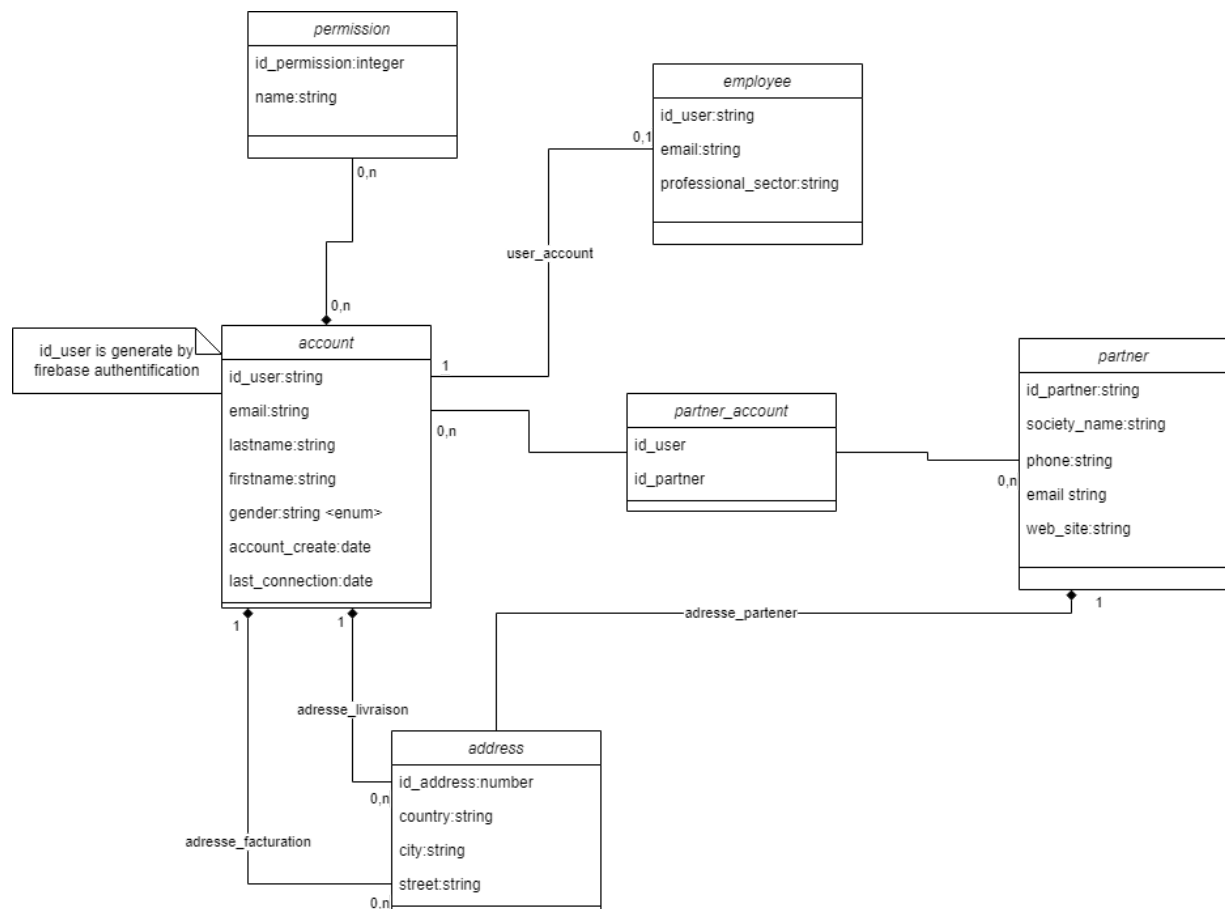
Dans la suite de ce projet, nous devrons aussi réfléchir à l'implémentation d'interface réelle avec la banque, qui pour l'instant n'est qu'un composant bouchonné. Il nous faudrait alors implémenter différentes interfaces afin d'interagir avec les différentes banques de chaque pays et zone du monde.

Un service de notification pourrait également être mis en place afin de prévenir un client d'un changement de statut de sa commande (son email étant dans Account) ou pour le prévenir de nouveaux produits sur le catalogue. Ce service servirait également à avertir des partenaires d'une nouvelle livraison et des employés de Polypet d'une nouvelle requête d'ajout de produit. Enfin, il serait possible de contacter l'équipe de maintenance par ce service lorsque le monitoring cloud détectera des anomalies ou problèmes d'indisponibilités sur les différents services.

Il est envisageable de mettre un système de connexion anonyme valide uniquement sur une période donnée. Cela nous permettrait de mettre en place un système de panier fonctionnel sans se connecter sur le site.

(suite sur l'autre page)

Nous souhaitons également faire évoluer nos modèles de données comme ci-dessous :

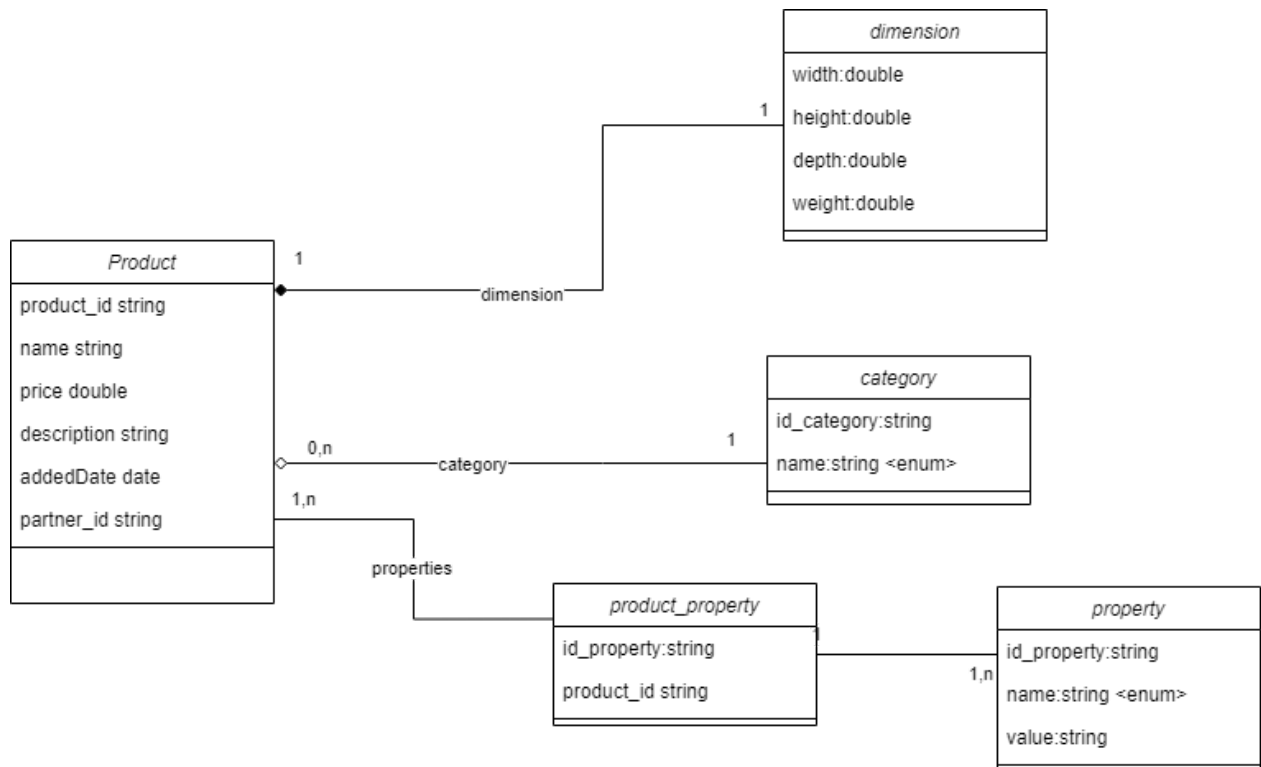


La base de données précédente avait des défauts, comme très peu de données et aucune relation entre les différentes tables.

Elle permettait donc de répondre à un premier MVP mais l'évolution de celui-ci nous semble nécessaire pour les prochaines versions du site. Le nouveau modèle ci-dessus offre bien plus de possibilités. Tout d'abord, au niveau des adresses, cela permet à un client de les avoir liées à son compte. De plus, la table Partner, qui représente une société partenaire, a la possibilité d'avoir plusieurs comptes (un par employé de la société). Il en va de même pour les employés de Polypet et la table Employee, qui est un compte particulier qui a la possibilité d'avoir des informations supplémentaires.

Au niveau de la sécurité, ce modèle offre la possibilité d'avoir différentes permissions afin de pouvoir donner accès uniquement aux pages qui leurs sont autorisées.

Catalog DB



Pour les produits, nous avons décidé de revoir le modèle mais nous avons décidé de garder la table Product avec les informations principales du produit.

Ensuite, nous avons découpé la dimension d'un produit et créé une table spéciale pour cette donnée en séparant et typant les champs (afin de manipuler un objet à part entière et non une string).

Pour rendre l'utilisation des filtres plus simple, la catégorie du produit a été détachée et mise dans une table à part afin de le gérer comme un objet et faciliter son utilisation dans le code. Un produit ne peut cependant n'avoir qu'une et une seule catégorie.

Au niveau des différents filtres que nous pouvons mettre en place sur un produit, nous avons décidé de partir sur une table de "property". Cette table permet à un produit d'avoir plusieurs propriétés avec un nom "name" fixé et géré en interne par une énumération. Ces propriétés peuvent être par exemple les ingrédients dont le produit est composé (par exemple, un produit peut avoir plusieurs fois la même propriété mais avec des valeurs différentes).

Grâce à cela, il est facilement envisageable de mettre un système de filtre et de comparaison des différents produits de notre site pour aider l'utilisateur à faire son choix.