Solution IYARD

Solution technique et fonctionnelle

Email : even@pouyadou.fr
Site : https://IYARD.fr/

POUYADOU Even
Date : Juin 2019

Solution

Version 1.0

Table des matières

Introduction

Objet du document

Ce document définit la solution technique et fonctionnelle adoptée pour la conception de l'applications web nécessaire à la création d'un site pour une entreprise de jeu vidéo.

Contexte

Ce document découle du cahier des charge suivant :

Une entreprise de jeu vidéo souhaites créer une application web permettant aux usagers de pouvoir se tenir au courant de leur dernière avancé, acheter des abonnements et des objets à leur jeu, discuter ensemble grâce à un forum. Ce système comprendra dans sa version finale :

- Le site d'accueil (en responsive design) accessible aux usagers et permettant :
  - De lire des news.
  - (Option) De lire des informations liées au jeu (histoire, univers, wiki, ...).
  - (Option - spéciale) De rechercher des objets / histoire / news.
  - (Option) De pouvoir changer de langue.
- Un site boutique (en responsive design) accessible aux usagers et permettant :
  - D'acheter des abonnements ou/et des objets de jeu.
  - (Option - spéciale) Rechercher ces objets grâce à des catégories ou nom
- Un site administratif (en responsive design) accessible uniquement par le personnel qui suivant les permissions de l'usager pourra :
  - Écrire des news et programmer sa publication
  - (Option) Écrire des histoires
  - (Option) Ajouter des objets au wiki
  - Ajouter des éléments dans la boutique
- (Option) Un site forum (en responsive design) accessible aux usagers et permettant :
  - D'écrire, lire et supprimer des threads
  - D'écrire, lire, supprimer et répondre à des post
- (Option) Un site support (en responsive design) accessible aux usagers et permettant :
  - Créer des tickets à propos de bug en jeu, sur un des sites
  - Créer des tickets à propos d'un achat (SAV)
- (Option) Un site support / SAV pour le personnelle permettant :
  - De lire et répondre à des tickets envoyer par les usagers
- (Option) Plusieurs batch, à savoir :
  - Un batch pour les newsletters
  - Un batch pour les notifications du forum
  - Un batch pour les notifications de la boutique

Option – spéciale : La recherche, le trie sera fait grâce à AJAX, n'ayant pas encore les connaissances requises. Cela devient une option

Toutes les fonctionnalités précédé d' « (Option) » sont des fonctions qui seront implémenté définitivement mais pas forcément lors de la première version.

La première version de ce site contiendra donc :

- Un site principal qui permettra :
    - D'afficher des news
    - De créer un compte
    - De se connecter
    - D'éditer son profil
- Un site boutique qui permettre :
    - D'afficher des produits
    - Se connecter
    - D'ajouter au panier des articles
    - De passer une commande fictive
- Un site d'administration qui permettra :
    - D'afficher, d'ajouter, de modifier ou de supprimer des news
    - D'afficher, d'ajouter, de modifier ou de supprimer des produits
    - D'afficher, d'ajouter, de modifier ou de supprimer des utilisateurs

Contrainte

Chaque micro service doivent être indépendant au cas où il y en aurait un qui pose problème.
Les différent site doivent pouvoir accueillir beaucoup de personnes, et être rapide à charger.
La configuration doit être simple (version 2 et + : externalisé).

Objectif

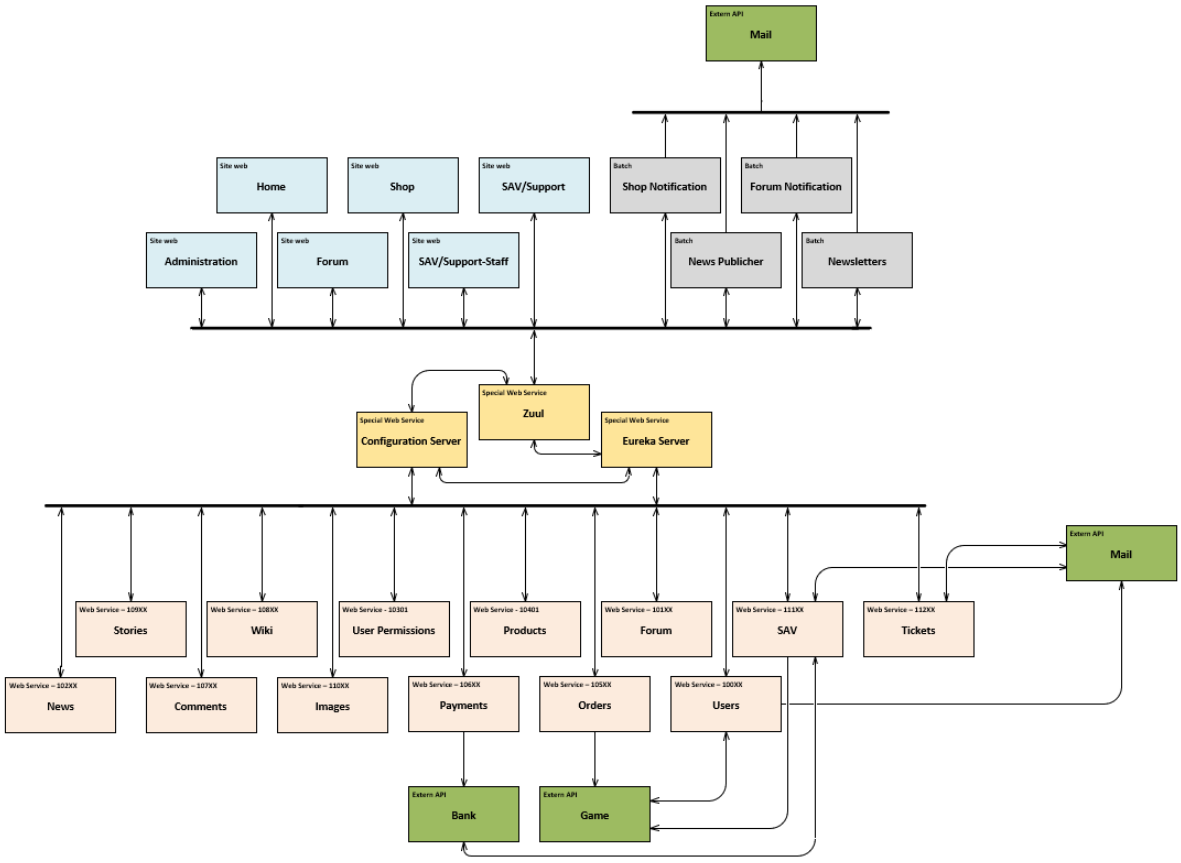
Dans l'optique de répondre au mieux aux attentes du client et de ses clients, ce document sera une aide précieuse pour les développeurs qui leur permettra de comprendre au mieux le fonctionnement de cette application.

Description Générale de la solution

D'après l'analyse des besoins de la société et des clients, nous en sommes arrivés à l'architecture suivante.

Les principes de fonctionnement

Le schéma ci-dessous, présente le découpage de l'application web.



Les acteurs

Différents acteurs interviennent sur ce projet. Le projet étant grand, nous nous contenterons de les lister et d'expliquer leurs rôles. Les différents acteurs sont les suivants :

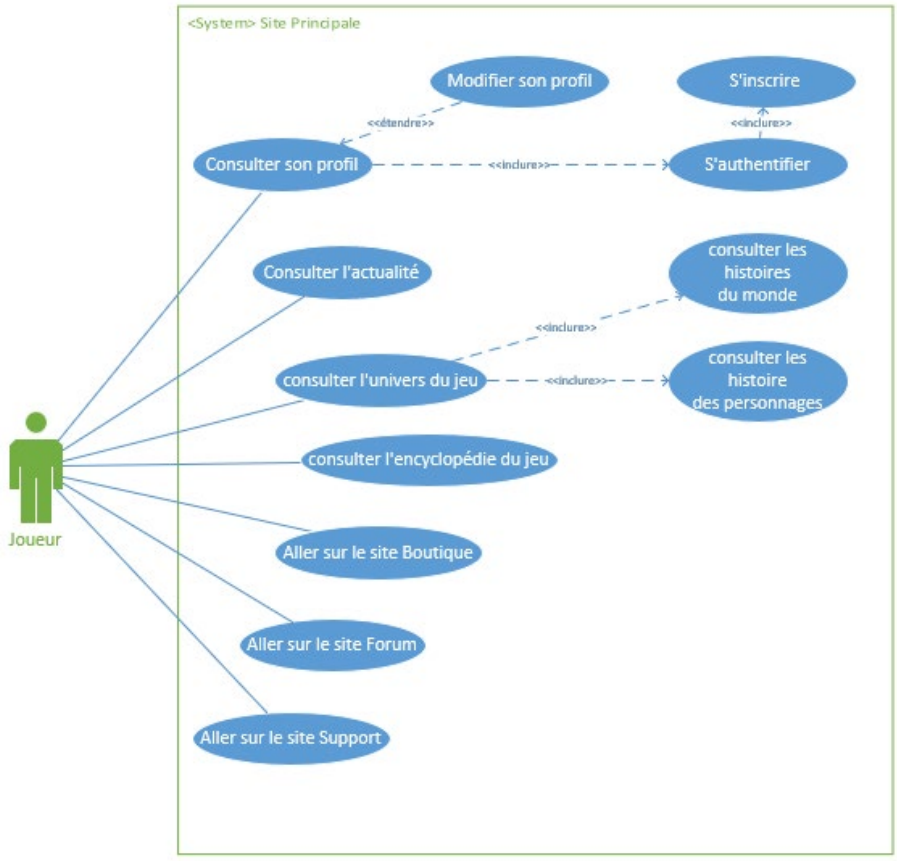
- Le client : peut se rendre sur les sites suivants :
  - Site d'accueil (Home), sur lequel il va pouvoir :
    - Lire l'actualité (les news), les histoires et le wiki s'il cherche des informations sur le jeu.
    - S'inscrire ou se connecter
    - Changer la langue du site
  - Site boutique (Shop), sur lequel il va pouvoir :
    - Regarder la liste des articles
    - Acheter un article
    - Rechercher un article
  - Site de support (SAV/Support), sur lequel il va pouvoir :
    - Créer un ticket en précisant le type de problème et en le décrivant.
  - Site Forum (Forum), sur lequel il va pouvoir :
    - Créer un thread (un sujet)
    - Répondre à un post (message d'un thread)

        - Supprimer un thread ou un post qui lui appartient
        - Signalé un thread ou un post offensant

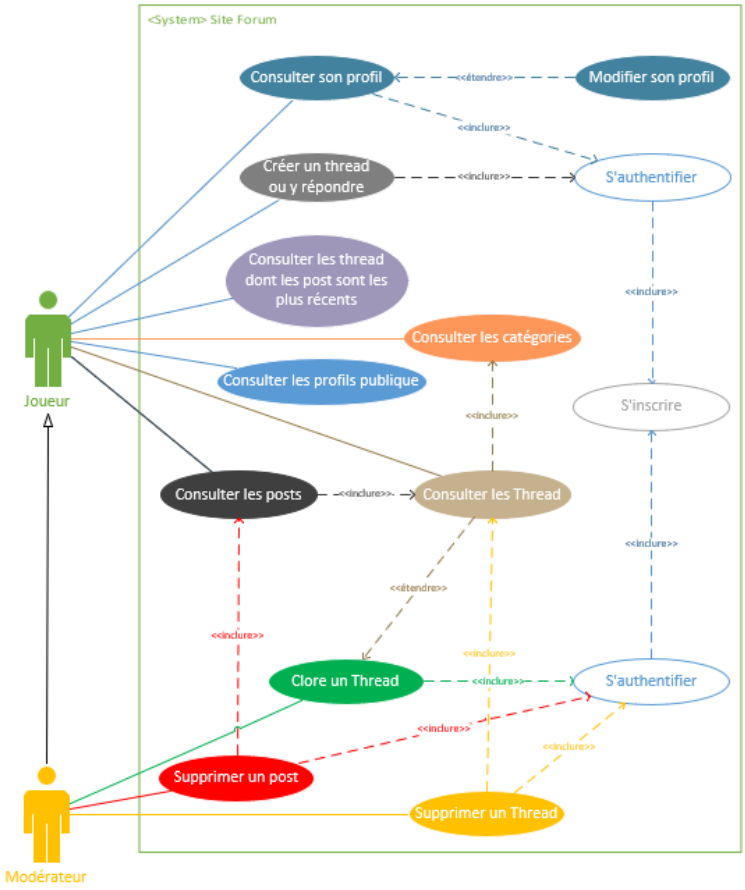
- Le modérateur : peut se rendre sur le site Administration. Cela lui permettra :
    - D'ajouter, de supprimer et de modifier :
        - Des news (les news peuvent être programmable pour qu'elle sorte à une heure précise.
        - Des histoires sur le jeu, son univers, ou des histoire parallèle
        - Des éléments au wiki (objets, statistique personnage …)
        - Des articles en vente sur la boutique
    - De modifier :
        - Le prix des articles en vente
    - Créer :
        - Des promotions sur un article, un type d'article, ou sur toute la boutique
- Le personnel du support peut se rendre sur le site SAV/Support-Staff. Cela lui permettra :
    - De lire et répondre au ticket reçu
    - Accédé aux informations utilisateur (Profil, Achat, …) (Ne vois pas les données personnelles & confidentielles)
    - Remboursé un achat
    - Donner les articles acheter qui n'aurait pas été donné dû à un bug.

Les cas d'utilisation

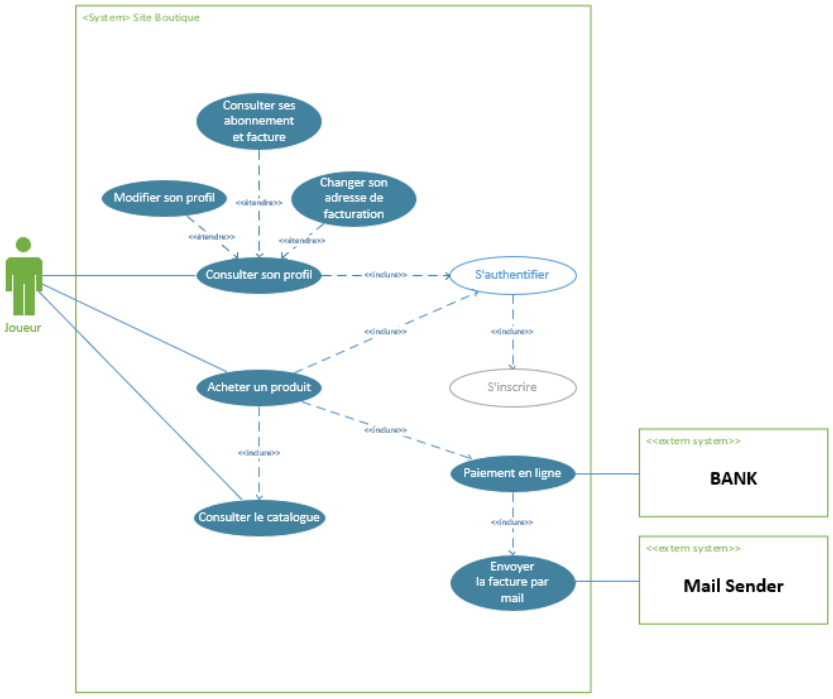
Ci-dessous sont présenté les différents cas d'utilisation liés à l'application. Ces derniers feront l'objet d'une description plus approfondis plus tard dans ce document



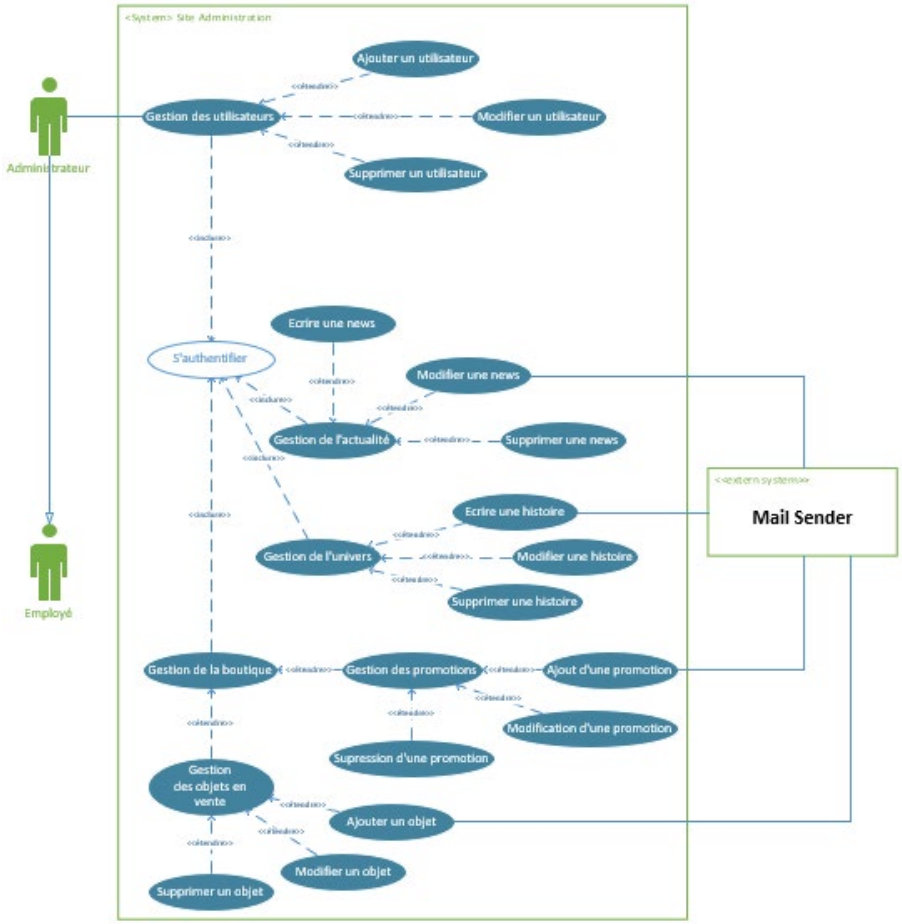
1. Cas d'utilisation sur le site principal



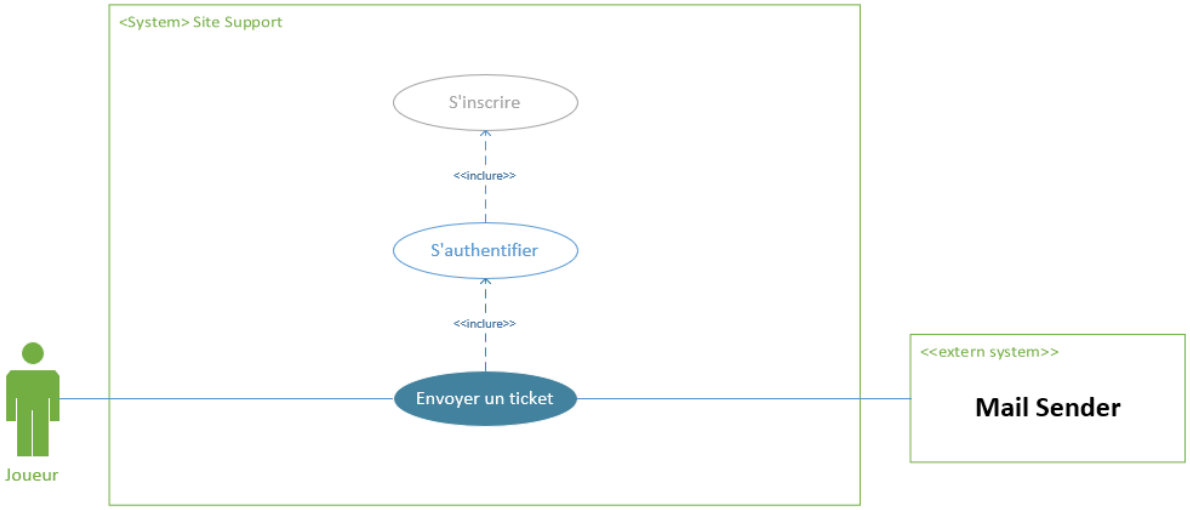
2. Cas d'utilisation sur le site « Forum »



3. Cas d'utilisation sur le site « Boutique »



4. Cas d'utilisation sur le site « Administration »



5. Cas d'utilisation sur le site « Support »

Domaine Fonctionnel

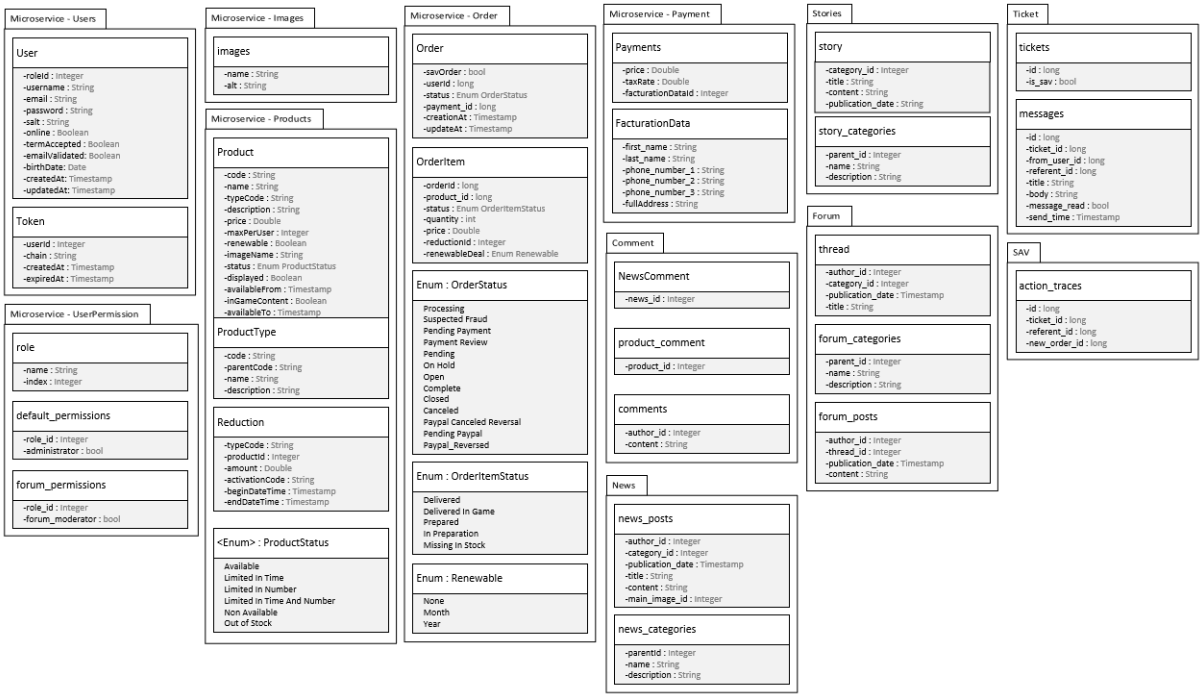
Les paragraphes suivants ont pour objectif de présenter l'organisation et l'utilisation des données par l'application.

Référentiel

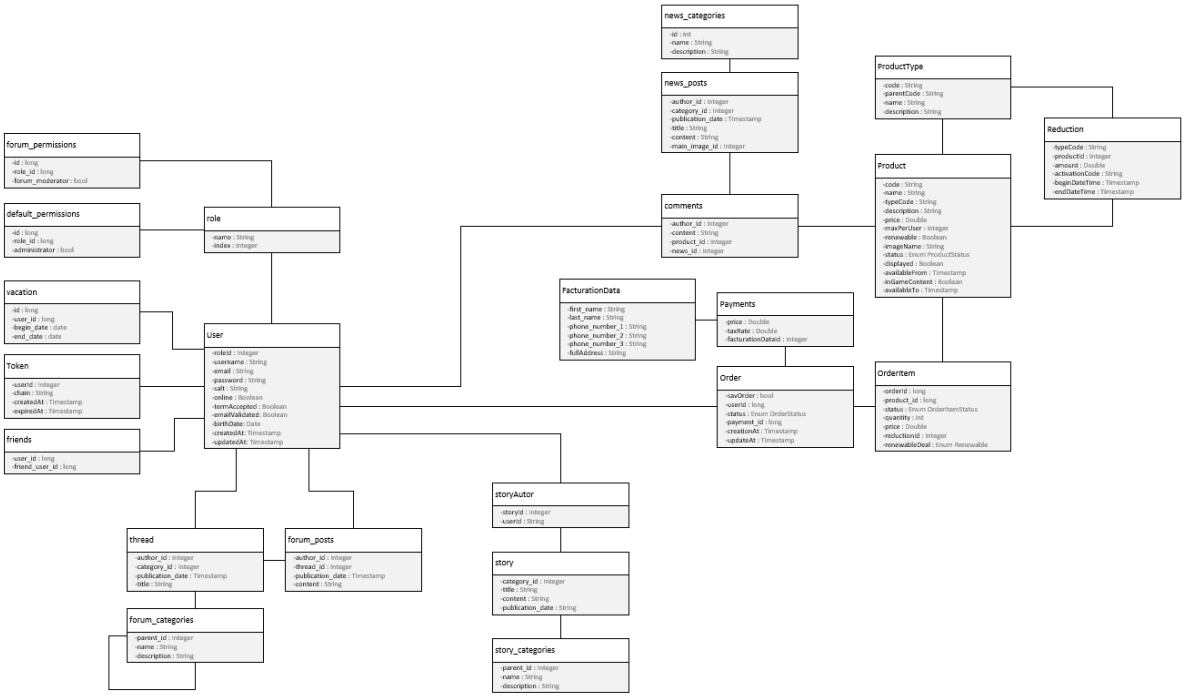
Diagrammes

La mise en place de cette application en micro-services nécessite de savoir comment sont réparti les modèles objets. Cette mise en place nous servira de point de départ pour la construction de cette application mais aussi de repère lors de future modification. À noter que ces schémas ne sont pas de bonne qualité dû au format de la page ; cependant vous pourrez les trouver dans le dossier ci-joint à ce document au format « .svg » ou « .jpeg ».

Le diagramme suivant présente la répartition des classes objets dans les différents micro-services.



Maintenant que nous avons vu la répartition entre les objets, nous allons nous intéresser aux liens entre ces derniers. Le schéma suivant présente donc ces liens.



Description des classes

Les objets présentés précédemment seront décrit de la manière suivante ; trier par micro-service puis par nom.

Micro-service – Users

User

User est la classe qui définit un utilisateur sur l'application. Ce compte utilisateur servira aussi à la connexion sur les jeux de notre client.

Variables	Description
roleId	Définit l'id du rôle au demander au micro-service « user_permissions » pour obtenir les permissions attribuées à ce dernier.
username	Définit l'identifiant de l'utilisateur.
email	Définit l'adresse email utilisée pour enregistrer ce compte.
emailValidated	Définit si oui ou non l'adresse email a été validé.
birthDate	Définit la date de naissance de l'utilisateur. L'utilisateur étant destiné à jouer à des jeux. Notre client se doit de vérifier l'âge de chaque personne jouant à ces derniers.
password	Définit le mot de passe « hashé » puis chiffré pour utiliser pour enregistrer ce compte.
Salt	Définit la clé de chiffrage du mot de passe.
termsAccepted	Définit si oui ou non l'utilisateur a pris connaissance des règles d'utilisation du site.
createdAt	Définit l'heure et la date à laquelle l'utilisateur a été créer dans la base de données.
updatedAt	Définit l'heure et la date à laquelle l'utilisateur a été mis à jour dans la base de données.

Token

Token est la classe qui permet à un utilisateur de se connecter automatiquement sur le site (client uniquement). Évidemment, l'utilisateur ne peux se connecter via ce token, uniquement s'il en a un attribué et que ce dernier n'ait pas expiré.

Variables	Description
user_id	Définit l'id de l'utilisateur associé au token
chain	Définit une chaine sécurisée permettant la connexion d'un utilisateur.
createdAt	Définit la date à laquelle le token est créé
expiredAt	Définit la date et l'heure à laquelle le token expire

Vacation

Vacation est une classe qui sert à savoir quand est-ce qu'un employé est en vacances. Ses tâches de travaille, suivant l'importance pourrons être redistribué a d'autre pour que les clients de mon client n'attendent pas la fin des vacances de ce dernier pour avoir une réponse.

Variables	Description
user_id	Définit l'id de l'utilisateur associé aux dates
begin_date	Définit la date de départ en vacances. Date Inclus (premier jour de vacance)
end_date	Définit la date de retour de vacance. Date Exclus (premier jour de travail après les vacances)

Friends

Friends n'est pas une classe normale mais une classe d'association. Elle associe à un utilisateur une liste d'utilisateur. Ces utilisateurs seront considérés comme les amis de cette personne. Cependant, si la variable « black_list » est égal à « true » cette personne ne pourra plus contacter cette autre personne.

Variables	Description
user_id	Définit l'id de l'utilisateur associé à un autre utilisateur correspondant à l'id « friend_user_id ».
friend_user_id	Définit l'id de l'utilisateur associé à l'utilisateur correspondant à l'id « user_id »
black_list	Définit si les deux utilisateurs peuvent exister entre eux (s'ils peuvent communiquer, se voir, être ami, ...)

Micro-service – UserPermissions

Les permissions ne sont pas encore toutes décider. Il se peut que des permissions s'ajoute au cours du temps.

Role

Role est une classe qui définit le lien entre les différents types de permissions.

Variables	Description
name	Définit le nom du rôle.
index	Définit la position dans la hiérarchie

Default_permissions

Default_permissions est une classe qui définit les permissions de base associées à un rôle.

Variables	Description
role_id	Définit l'id du rôle auquel appartiennent les permissions suivantes.
administrator	Définit si l'utilisateur à toutes les permissions.

Forum_permissions

Forum_permissions est une classe qui définit les permissions du forum associé à un rôle.

Variables	Description
role_id	Définit l'id du rôle auquel appartiennent les permissions suivantes.
forum_moderator	Définit si l'utilisateur à toutes les permissions pour manager le forum.

News

News_post

News_post est une classe qui définit une news. Ces news sont triés par catégories.

Variables	Description
authorId	Définit l'id de l'utilisateur qui a écrit la news. Tout le monde ne peut pas écrire une news. Cela est réservé uniquement aux personnes ayant les permissions et uniquement à partir du site d'administration.
categoryId	Définit la catégorie dans laquelle se trouve la news.
publicationDate	Définit la date de parution de la news. Elle peut aussi servir à programmer la sortie de la news.
title	Définit le titre de la news
content	Définit le contenu de la news
mainImageId	Définit l'id de l'image principale de la news.

News_categories

News_categories est une classe qui définit le nom et la description d'une catégorie. Une catégorie peut être sous-catégorie d'une autre.

Variables	Description
name	Définit le nom de la catégorie.
description	Définit la description de la catégorie.
parentId	Définit l'id de la catégorie parent.

Stories

story

Story est une classe qui définit une histoire. Ces histoires sont triées par catégories.

Variables	Description
categoryId	Définit la catégorie dans laquelle se trouve l'histoire.
publicationDate	Définit la date de parution de l'histoire. Elle peut aussi servir à programmer la sortie de l'histoire.
title	Définit le titre de l'histoire
content	Définit le contenu de l'histoire

Story_categories

Story_categories est une classe qui définit le nom et la description d'une catégorie. Une catégorie peut être sous-catégorie d'une autre.

Variables	Description
name	Définit le nom de la catégorie.
description	Définit la description de la catégorie.
parentId	Définit l'id de la catégorie parent.

Forum

Thread

Thead est une classe qui définit un sujet ouvert sur le forum.

Variables	Description
authorId	Définit l'id de l'utilisateur qui a écrit le sujet.
categoryId	Définit la catégorie dans laquelle se trouve le thread.
publicationDate	Définit la date de parution du sujet.
title	Définit le titre du sujet.

Forum_posts

Forum_posts est une classe qui définit un message envoyé dans un des sujets sur le forum.

Variables	Description
authorId	Définit l'id de l'utilisateur qui a écrit le post.
threadId	Définit le thread dans lequel se trouve le post.
publicationDate	Définit la date de création du post.
content	Définit le contenu du sujet

Forum_categories

Forum_categories est une classe qui définit le nom et la description d'une catégorie. Une catégorie peut être sous-catégorie d'une autre.

Variables	Description
name	Définit le nom de la catégorie.
description	Définit la description de la catégorie.
parentId	Définit l'id de la catégorie parent.

Images

Images

Images est une classe qui définit le nom et la description pour les mal-voyant des images enregistrées

Variables	Description
name	Définit le nom de l'image.
alt	Définit la description de l'image.

Products

Products

Products est une classe qui définit l'ensemble des informations nécessaire à la description de celui-ci.

Variables	Description
code	Définit le code produit qui sera utilisé pour générer les factures et la recherche de produit avec le SAV.
name	Définit le nom du produit.
typeCode	Définit le code du type de produit. Cela sert à faire des promotions sur un ensemble de produit du même type.
price	Définit le prix du produit.
maxPerUser	Définit le nombre de produit max achetable par l'utilisateur.
renewable	Définit si c'est un produit renouvelable (un abonnement renouvelable chaque mois par exemple)
imageName	Définit l'image du produit
status	Définit le statut du produit (Disponible, rupture, ...)
displayed	Définit si oui ou non le produit est affiché en boutique
inGameContent	Définit si oui ou non le produit est lié au jeu.
availableFrom	Définit la date et l'heure où le produit est ajouté à la vente
availableTo	Définit la date et l'heure où le produit est retiré de la vente

Product_types

Product_types est une classe qui définit le type d'un produit.

Variables	Description
Code	Définit le code du type de produit.
parentCode	Définit le code du type de produit parent
name	Définit le nom du type de produit
desciption	Définit la description du type de produit

Reductions

Reductions est une classe qui définit le montant et le code de réduction.

Variables	Description
type_code	Définit le code du type de produit en réduction.
productId	Définit le produit en réduction
amount	Définit le pourcentage de réduction
activationCode	Définit le code d'activation de la promotion
beginDateTime	Définit la date et l'heure du début de la réduction
endDateTime	Définit la date et l'heure de fin de la réduction

Orders

Orders

Orders est une classe qui définit une commande.

Variables	Description
savOrder	Définit si oui ou non cette commande a été créé par un membre du personnel du SAV.
userId	Définit l'id de l'utilisateur créateur de la commande
status	Définit l'id utilisateur à qui cette commande est dû
paymentId	Définit l'id du paiement lié à la commande.
createdAt	Définit la date et l'heure à laquelle la commande a été créée.
updateAt	Définit la date et l'heure à laquelle la commande a été mis à jour

OrderItem

OrderItem est une classe qui définit un type produit et le nombre qu'il y en a dans la commande liée.

Variables	Description
orderId	Définit l'id de la commande associée.
productId	Définit l'id du produit qui est commandé.
status	Définit l'état du produit commandé (en préparation, prêt à être livré, livré, ...)
quantity	Définit la quantité du produit commandé.
price	Définit le prix unitaire du produit au moment de la commande.
reductionId	Définit l'Id de la réduction appliquée.
renewableDeal	Définit à quel fréquence le produit doit-il être renouvelé.

Payments

Payments

Payments

Payments est une classe qui définit le montant de la commande et la taxe appliqué au moment de l'achat.

Variables	Description
price	Définit le prix à payer par l'utilisateur par la commande.
taxRate	Définit le pourcentage de taxe prélevé sur la commande
facturationDataId	Définit le lien vers les informations de facturation utilisateur.

FacturationData

FacturationData est une classe qui définit les informations de paiement nécessaire à la validation de la commande.

Variables	Description
first_name	Définit le prénom de l'acheteur.
last_name	Définit le nom de l'acheteur.
phone_number_1	Définit le numéro de téléphone de l'acheteur.
phone_number_2	Définit le numéro de téléphone de l'acheteur.
phone_number_3	Définit le numéro de téléphone de l'acheteur.
fullAddress	Définit le l'adresse de facturation de la commande

Tickets

Tickets

Tickets est une classe qui définit à quel service est destiné la demande.

Variables	Description
is_sav	Définit si oui ou non ce ticket est destiné au sav.

Messages

Messages est une classe qui définit peut recevoir un membre du personnel du SAV.

Variables	Description
ticket_id	Définit l'id du ticket associé.
from_user_id	Définit l'id utilisateur de l'envoyeur.
referent_id	Définit l'id utilisateur du membre du personnel du SAV à qui a été donner ce ticket.
title	Définit le titre du message
body	définit le contenu du message
message_read	Définit si le message a été lu ou non.
send_time	Définit la date et l'heure à laquelle a été envoyé le message.

SAV

Action_traces

Action_traces est une classes d'association qui lie un ticket avec un membre du personnel du SAV et la commande traitée.

Variables	Description
ticket_id	Définit l'id du ticket traité.
referent_id	Définit l'id utilisateur du personnel du SAV
new_order_id	Définit l'id de la commande traitée.

Comments

Product_comments

Product_comments est une classe qui définit un commentaire pour un produit.

Variables	Description
product_id	Définit l'id du produit commenté.
author_id	Définit l'id utilisateur qui a écrit le commentaire.
content	Définit le contenue du commentaire.

News_comments

News_comments est une classe qui définit un commentaire pour une news.

Variables	Description
news_id	Définit l'id de la news commentée.
author_id	Définit l'id utilisateur qui a écrit le commentaire.
content	Définit le contenue du commentaire.

Règles de gestion

Les règles de gestions sont les suivante (triées par acteur) :

Client / Joueur

- Un utilisateur doit pour se connecter :
    - Entrer son email ou son pseudonyme.
    - Entrer son mot de passe.
    - Valider un « anti-bot » en cas d'un nombre d'essai fixé dépasser.
- Un utilisateur souhaitant s'inscrire doit :
    - Entrer un email non utilisé.
    - Entrer une deuxième fois son email pour le confirmer.
    - Entrer un pseudonyme non utilisé.
    - Entrer sa date de naissance.
    - Entrer un mot de passe.
    - Entrer une deuxième fois son mot de passe pour le confirmer.
    - Accepter les conditions d'utilisation.
    - Cocher si oui ou non il souhaite s'inscrire à la newsletter.
    - Que l'utilisateur n'est pas un « bot ».
- Un utilisateur doit pour commander :
    - Avoir un moyen de paiement accepté par le site, et valide. (PayPal, Carte Visa)
    - Avoir un compte.
    - Certifier avoir lu et accepté le contrat de vente.
    - Fournir l'adresse de facturation, nom et prénom, numéro de téléphone.
    - Accepter le fait qu'un produit numérique destiné à un jeu (autre abonnement) de donne droit à aucun remboursement.
- Un utilisateur ayant commandé :
    - Reçoit le ou les produits numériques commandé après la validation du paiement par la banque. En cas des refus ces objets ne sont pas donnés, ou retirer suivant s'ils ont déjà été donné ou pas.
    - Reçoit un mail et une facture par email. La facture doit rester consultable sur l'application.
    - S'il le souhaite, et suivant la nature du produit commandé, il peut demander un remboursement.
- Un utilisateur souhaitant créer un thread sur le forum doit :
    - Être connecté.
    - Être dans une catégorie.
    - Entrer un titre.
    - Entrer le contenu du thread.
- Un utilisateur souhaitant créer un post sur un thread du forum doit :
    - Être connecté.
    - Entrer le contenu du post.
- Un utilisateur souhaitant commenter un news doit :
    - Être connecté.
    - Avoir entrer le contenu du commentaire.
- Un utilisateur souhaitant commenter un produit de la boutique doit :
    - Être connecté.
    - Avoir déjà acheter le produit.
    - Avoir entrer le contenu du commentaire.
- Un utilisateur souhaitant créer un ticket doit :
    - Être connecté
    - Avoir entrer un titre.
    - Avoir entrer le contenu du ticket

Membre du staff

- L'utilisateur souhaitant ajouter une news ou modifier (programmer ou non) doit :
    - Être connecté.
    - Être sur le site de l'administration

    - Avoir les permissions
    - Avoir entrer le titre
    - Avoir ajouter l'image principale
    - Avoir entrer le contenu
    - Avoir entrer la catégorie
    - S'il souhaite la programmer l'utilisateur :
        - Doit avoir entré la date de parution

- L'utilisateur souhaitant supprimer un news (programmer ou non) doit :
    - Être connecté
    - Être sur le site de l'administration
    - Avoir les permissions
- L'utilisateur souhaitant ajouter, modifier un produit doit :
    - Être connecté
    - Être sur le site de l'administration
    - Avoir les permissions
    - Avoir remplis les informations sur le produit
- L'utilisateur souhaitant supprimer un produit doit :
    - Être connecté
    - Être sur le site de l'administration
    - Avoir les permissions
- L'utilisateur souhaitant répondre à un ticket (support) reçu doit :
    - Être connecté
    - Être sur le site du support
    - Avoir les permissions
- L'utilisateur souhaitant répondre à un ticket (SAV) reçu doit :
    - Être connecté
    - Être sur le site du SAV
    - Avoir les permissions
- L'utilisateur souhaitant ajouter, modifier une histoire doit :
    - Être connecté
    - Être sur le site de l'administration
    - Avoir les permissions
    - Avoir remplis le titre
    - Avoir remplis le contenu de l'histoire
- L'utilisateur souhaitant supprimer une histoire doit :
    - Être connecté
    - Être sur le site de l'administration
    - Avoir les permissions

Application Web

Les acteurs

Les différents sites de cette application ont chacun des acteurs qui leurs sont propre. Cependant, il faut savoir que tous les utilisateurs du site (hors système) ont un compte utilisateurs une fois inscrit, même les employés. Voici les listes des acteurs qui joue sur ces sites.

(À noter que les différents rôles possibles d'avoir sur ces applications sont définissables et par conséquent il nous est impossible de les connaître, trois rôles sont définis par défaut : Super admin qui est réservé aux tests et en cas de secours, Administrateur à tous les droits, et Joueur sont les utilisateurs les plus restreint. Pour l'exemple disons qu'il existe le rôle Modérateur qui peut agir sur le Forum pour des actions de gestion).

Site principale (site d'accueil) :

Premièrement, le site principal n'a pas vraiment d'acteurs précis. Le seul changement d'état qui nous ajoute des possibilités est de passer d'un client lambda à un client connecté qui nous permet de consulter son profil. La liste des acteurs de ce site est donc :

- Joueur : Nous attendons sur ce site des joueurs. Il a donc été choisi de les nommer Joueur dans le schéma du cas d'utilisation. Les actions faisables par cet acteur sont simples :
    - Consulter l'actualité : Pour voir toutes les nouveautés du jeu et de son développement, de la boutique et du site en générale.
    - Consulter l'univers du jeu : Il a été demandé à ce que les joueurs puissent lire différentes histoires à propos du monde et des personnages du jeu.
    - Consulter l'encyclopédie du jeu : Le joueur à aussi la possibilité de consulter tout un tas de description des éléments du jeu ; leurs utilisations, les différentes recettes possibles avec ces dernier (dans le cas d'un matériau), …
    - Passer sur un autre site de l'application :
        - Boutique
        - Forum
        - Support
- Joueur Authentifié : Comme dit, se connecté permet de consulter son profil.

Forum :

Le forum est un site qui permet au joueur d'échanger des messages sur un certain sujet défini par des catégories. La liste des acteurs de ce site est :

- Joueur : Le joueur, ici, peut :
    - Consulter les profils publics d'autre joueur.
    - Consulter différents thread (= sujet de discussion)
        - Ceux dont les posts sont les plus récents sont mis en avant.
    - Consulter les posts d'un thread : Les posts sont les messages que s'échange les joueurs dans un thread. L'ensemble de ces derniers forment un thread.
- Joueur Authentifié : le joueur authentifié peut :
    - Consulter son profil
    - Créer un thread
        - La création d'un thread inclut la création d'un post
    - Créer un post pour répondre à un autre.
- Modérateur : le modérateur peut :
    - Supprimer un thread
    - Supprimer un post
    - Clore un thread : Ne supprime pas le thread mais empêche tout nouveau message dans ce dernier.

Site boutique :

La boutique s'adresse aux joueurs est ne peut être gérer côté site client. Cependant des acteurs externes joues sur ce site :

- Joueur : Le joueur peut :
    - Consulter le catalogue de produit vendu.
- Joueur authentifié : le joueur authentifié peut :
    - Consulter son profil
    - Acheter un produit
    - Payer en ligne
- Bank : Ce service externe permet aux joueurs en train de commandé de pouvoir payer. Bank permet de valider un paiement. Ce service représente tous les moyens de paiement capable de valider un paiement. Ce peut-être aussi bien PayPal comme le service d'une banque.
- Mail Sender : Ce service externe permet d'envoyer la facture d'une commande au joueur concerné.

(J'ai dispensé le fait de rajouter l'acteur externe « Game API » car elle n'existe pas pour le moment et n'est pas compris dans le développement de cette première version du site)

Site support :

Pour pouvoir faire n'importe quelle action sur ce site, il est essentiel d'être authentifié.

- Joueur authentifié : Cette utilisateur à une unique action sur ce site qui correspond à envoyer un ticket en spécifiant la catégorie de son / ses problèmes pour être aidé en retour par la personne qui convient par mail.

Site Administration :

Ce site n'est atteignable qu'à partir de l'entreprise. C'est un site réservé aux employés de cette dernière. Bien évidement pour pouvoir faire n'importe quelle action, chaque utilisateur doit être authentifié.

(À noter que les droits des employés varient suivant leur rôle, mais pour des raisons de présentation, les permissions ne seront pas représentées)

- Employé authentifié : Un employé peut :
    - Gérer l'univers : écrire, modifier ou supprimer une histoire
    - Gérer la boutique
        - Gérer les articles
            - La date d'apparition et de disparition d'un article peut être programmable.
        - Gérer les promotions : comme les articles les promotions ont une durée où elles sont actives et d'autres où elles ne le sont pas.
            - Par type
            - Par article
    - Gérer l'actualité : écrire, modifier ou supprimer un news.
        - Les news sont programmables pour sortir à une heure précise.
- Administrateur :
    - Gérer les employés : ajouter, modifier les permissions / les rôles, supprimer un utilisateur.

Les cas d'utilisations

IDENTIFIANT	Visite du site principale
Description	Un utilisateur non connecté visite le site principal et s'inscrit puis vérifie son profil
Préconditions	L'utilisateur ne doit pas être connecté ou inscrit
Donnée en entrée	L'utilisateur demande la page « index »
Scénario nominal	1. Le système charge le dernier news 2. Le système affiche la page avec la dernière news ajoutée 3. L'utilisateur appuis sur le bouton « connexion »

	4. Le système charge les données nécessaires et affiche la page de connexion. 5. L'utilisateur n'ayant pas de compte appuis sur le bouton « Sign up now » 6. Le système charge la donnée nécessaire et affiche la page d'inscription 7. L'utilisateur entre ses informations et valide son inscription. 8. Le système enregistre en base les données de l'utilisateur et renvois l'utilisateur sur la page de connexion 9. L'utilisateur entre ses informations de connexion et valide 10. L'utilisateur est connecté par le système et créer une session avec les informations de l'utilisateur. Le système redirige l'utilisateur vers la page d'accueil. 11. L'utilisateur connecté appuis sur le bouton avec son nom de compte. 12. Le système redirige l'utilisateur vers son profil 13. L'utilisateur s'étant trompé dans son pseudonyme décide de l'éditer et décide d'appuyer sur le bouton « edit profile » 14. Le système affiche la page 15. L'utilisateur modifie son pseudonyme et valide 16. Le système enregistre les informations en base et redirige l'utilisateur vers son profil.
Scénario alternatif	5. L'utilisateur ne souhaite pas s'inscrire et revient en arrière ou ferme le site 7. L'utilisateur ne souhaite pas s'inscrire et revient en arrière ou ferme le site 9. L'utilisateur se trompe dans le format de la saisie de ses informations et le système lui signale ce qu'il ne va pas. 9.b L'utilisateur a déjà un compte à cette adresse email. Il ne Peut donc pas s'en servir pour un nouveau compte. 9.c Le pseudonyme de l'utilisateur est déjà pris, il ne peut donc Pas s'en servir pour un nouveau compte. 15. L'utilisateur souhaite finalement le garder et retournes en Arrière
Résultats	L'utilisateur détient un compte et peut s'en servir pour modifier ses informations privées.
Erreur	En scénario alternatif quand l'utilisateur renonce à faire une action (5, 7, 15)

IDENTIFIANT	**Visite du site Boutique**
Description	Un utilisateur visite la boutique
Préconditions	L'utilisateur est connecté
Donnée en entrée	L'utilisateur demande la page « index » du site boutique
Scénario nominal	1. Le système charge les 10 dernier produit 2. L'utilisateur clique sur un produit 3. Le système charge un produit 4. Le système charge les informations relatives à se produit et affiche la page 5. L'utilisateur qui aime le produit l'ajoute à son panier 6. Le système ajoute au panier de l'utilisateur le produit affiche le panier 7. L'utilisateur souhaite commander et appuis sur le bouton

	8. Le système prépare la commande et affiche le récapitulatif avant le paiement pour l'utilisateur. 9. L'utilisateur appuis sur « payer » 10. Le système sécurise la page et affiche les différents moyens de paiement ainsi que leur interface respective. 11. L'utilisateur entre ses informations de paiement et valide sa commande. 12. Le système vérifie la validité du paiement puis redirige l'utilisateur sur la facture.
Scénario alternatif	8. et 9. Et 11. : L'utilisateur décide de renoncer à son achat.
Résultats	L'utilisateur vient de passer un achat
Erreur	En scénario alternatif quand l'utilisateur renonce à son achat

IDENTIFIANT	**Visite du site d'administration**
Description	Un administrateur visite le site de l'administration
Préconditions	L'utilisateur est connecté en tant qu'Administrateur
Donnée en entrée	L'utilisateur demande la page « news » du site administration
Scénario nominal	1. Le système affiche la page « news » 2. L'utilisateur souhaite créer une nouvelle actualité et clique sur « nouveau » 3. Le système charge la page « new news ». 4. L'utilisateur entre les informations et valide la création 5. Le système enregistre la news et redirige l'utilisateur vers la page « news » 6. L'utilisateur a fait une faute et décide de modifier la nouvelle news et clique sur le bouton « modifier ». 7. Le système charge les informations de la news et affiche la page « modify news ». 8. L'utilisateur corrige son erreur et valide la modification. 9. Le système enregistre les modifications et redirige l'utilisateur sur la page « news » 10. L'utilisateur souhaite supprimer une ancienne news et clique sur le bouton supprimer de la news en question. 11. Le système demande confirmation du choix de l'utilisateur. 12. L'utilisateurs valide. 13. Le système supprime la news
Scénario alternatif	4. et 8. : L'utilisateurs annule ou reviens en arrière 12. : L'utilisateurs annule.
Résultats	L'utilisateur vient de gérer la section news
Erreur	En scénario alternatif quand l'utilisateur décide d'annuler une de ses actions

Solution Technique

Logiciel utilisé

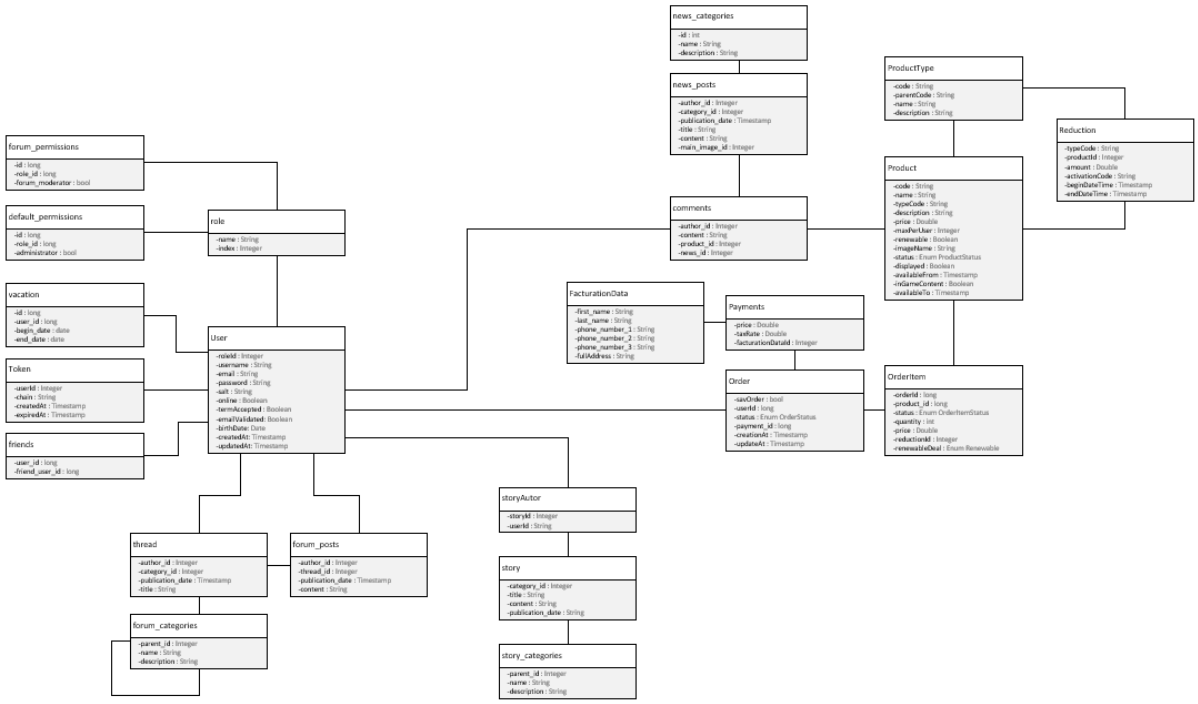
La pile logicielle est la suivante :

- Application J2EE (JDK version 1.8)
- Serveur d'application Apache Tomcat (version 8)
- Framework Spring Boot, Spring Data

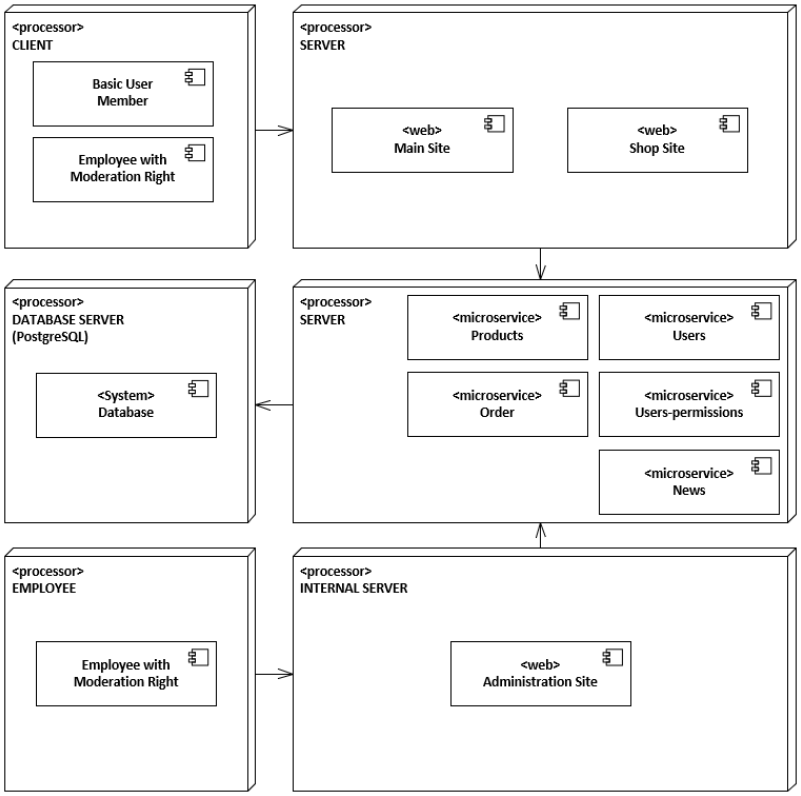
Base de données

Le système de gestion de base de données automatique utilisé sera **PostgreSQL**. En plus d'êtes rapide dans le traitement des requêtes SQL, il est compatible avec **Spring boot - Data**.

Modèle physique de données (MPD)



Architecture de déploiement



La solution envisagée prend la forme d'une application web à partir de laquelle les utilisateurs pourront effectuer les différentes actions prévues. Chaque micro-service peut être hébergé sur différente URL. Il en va de même pour les sites et les bases de données tant que vous pensez à changer la configuration dans les fichiers.

Serveur des bases de données

Les bases de données de l'application IYARD Project devront être déployer sur un ou plusieurs serveur Linux avec postgreSQL version 11 minimum.

Serveur de déploiement

L'application IYARD Project sera déployée sur un serveur Linux. L'application peut-être packager WAR ou JAR suivant les besoins. Dans le cas du WAR, il faudra que le projet soit lancer à partir de Tomcat. Quant au JAR, il contient Tomcat intégré par défaut avec Spring boot.

Architecture Logicielle

Principes généraux

Les sources et version de ce projet sont gérer par Git et GitHub, les dépendances et le packaging par Apache Maven.

Les couches

L'architecture des micro-services se compose de la sorte :

- **Configurations** (quand il y en a) : contient toutes les classes de configuration
- **Controllers** : contient toutes les classes qui s'occupe de recevoir des requêtes des clients et lancer les méthodes correspondantes pour donner en réponse un résultat aux clients.
- **Exceptions** : contient toutes les classes exceptions.
- **Models** : contient toutes les classes de données, dont leurs seuls rôles et de faire servir de conteneur pour des données. Le plus souvent on y trouve les mêmes objets que ceux dans la base de données.
- **Repository** : contient les classes communiquant avec la base de données et les row mappers.

L'architecture des sites se compose de la sorte :

- **Beans** : contient toutes les classes models des micro-services que nécessite le site.
- **Configurations** (quand il y en a) : contient toutes les classes de configuration
- **Controllers** : contient toutes les classes qui s'occupe de recevoir des requêtes des clients et lancer les méthodes correspondantes pour donner en réponse un résultat aux dispatcher pour qu'il puisse construire une view.
- **Exceptions** : contient toutes les classes exceptions.
- **Models** : contient toutes les classes de données, nécessaire au fonctionnement du site.
- **Proxies** : contient toutes les interfaces permettant de communiquer avec les micro-services grâce à open feign.

Points particuliers

Gestion des Logs

La gestion des logs se fait avec apache Log4J (version intégrée dans Spring boot). C'est une des bibliothèques d'Apache Logging. Apache Logging est un projet de la fondation Apache visant à fournir des bibliothèques de journalisation pour différents langages de programmation, ainsi que des applications pour les exploiter.

Fichiers de configuration

Les différents projets de l'application (micro-services et sites) ont un fichiers application.properties dans le répertoire resources. Ce fichier sert notamment à définir les accès à la base de données et d'autres propriétés comme la configuration des messages d'intentionnalisassions. Les micro-services quant à eux ont aussi un deuxième fichier application.properties dans le répertoire resources dans le dossier de test, qui permet la même chose que le premier mais pour les test.

Environnement de développement

Le développement de l'application se fait via un IDE. Dans notre cas se sera IDEA IntelliJ.

Préparer le projet / Configuration

Ce projet tendant à devenir un exemple pour tous, je vais expliquer comment le configurer et comment préparer le projet.

Préparer le projet

Bien sûr ce projet peut être ouvert avec n'importe quel IDE Java, néanmoins, ayant utilisé IntelliJ IDEA pour ce projet, je vais le faire avec celui-ci.

- Commencer par cloner le ripository github ou extraire de l'archive téléchargé les fichiers du projet
- Ensuite créer un nouveau projet Java maven simple
- Supprimer le fichier src
- Dans les propriétés du projet, dans modules, importer chaque sous projet (microservice-users, microservice-products, ...).
- Un fois tous les modules importés, vérifier bien que tous les dossiers soit marqué correctement (Fichier source, fichier ressources, Fichier source test, ...)

Configuration

Dans chaque fichier properities des microservices, vous pouvez configurer :

- L'url de la base de données
- L'utilisateur et son mot de passe
- Et le driver utiliser
  (Attention ce système utilise PostgreSQL, la compatibilité avec d'autre base de données n'est pas supportée)