# Description des diagrammes

Alexis LAURENT, Louis LABORIE, Jean MARCILLAC, Noa SILLARD, Shana CASCARRA, Tony FAGES

# Description rapide de notre projet :

Le projet vise à créer une application mobile et un site Internet dédiés à lutter contre la prolifération de la désinformation et des fausses nouvelles.

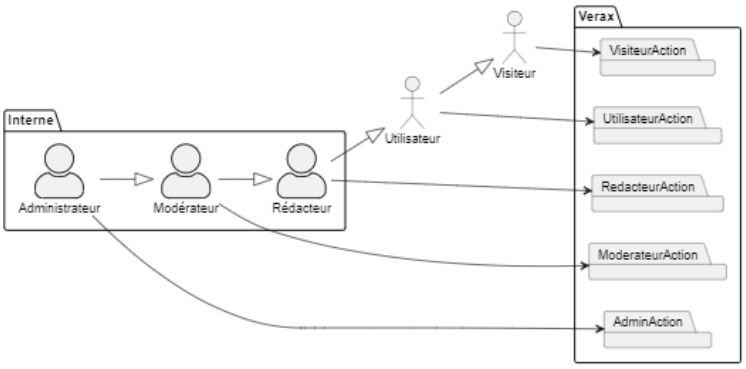
Il s'agira donc de traiter des articles et informations erronés, voir mensongers, qui, une fois soumis à notre équipe de rédacteurs, seront démontés par le biais de différentes ressources telles que des articles, des vidéos, des podcasts et des critiques d'images, qui expliqueront en détail chaque infox, en déconstruisant ses éléments faux, et présentant des faits et des preuves pour éclairer la vérité.

Une caractéristique importante sera la classification thématique des infox. Cette organisation permettra aux utilisateurs de naviguer facilement parmi les différents sujets et contribuera à identifier les tendances en matière de désinformation dans des domaines spécifiques tels que la politique, la santé, la technologie, etc.

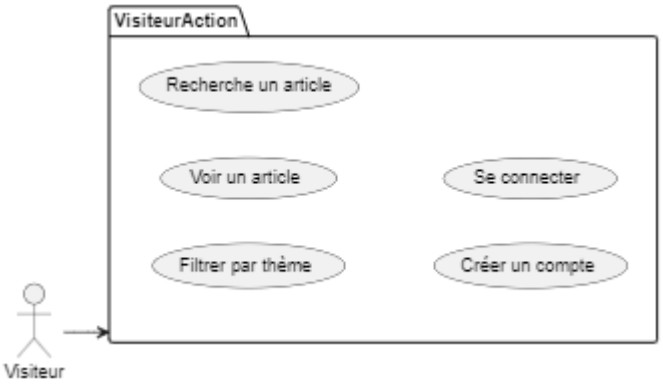
Pour garantir le bon fonctionnement de la plateforme, il y aura différents niveaux d'accréditation, que ce soit celui de simple lecteur, de rédacteur autorisé à la publication des différentes ressources, de modérateur pour la gestion de la communauté ou d'administrateur pour la gestion globale de la plateforme.

En résumé, cette plateforme a pour ambition de se positionner en tant que bouclier contre la désinformation en fournissant aux personnes les outils nécessaires pour distinguer la vérité au milieu du flux continu d'infox, par le biais d'articles postés sur la plateforme.

# Diagramme de cas d'utilisation :

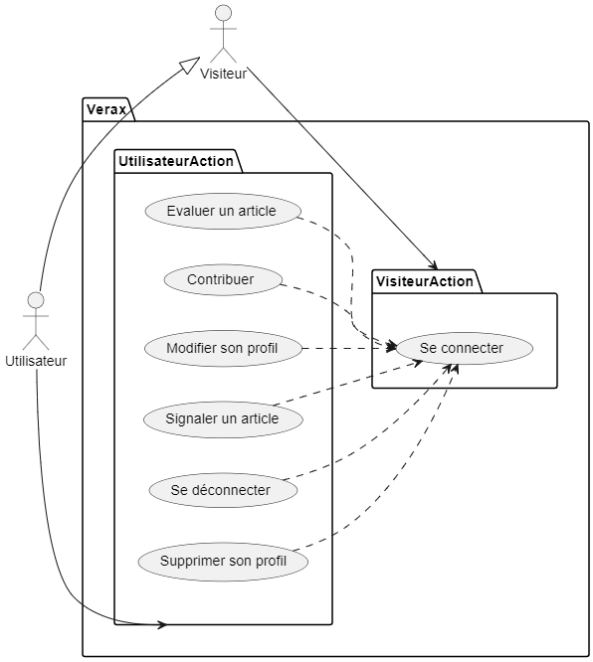


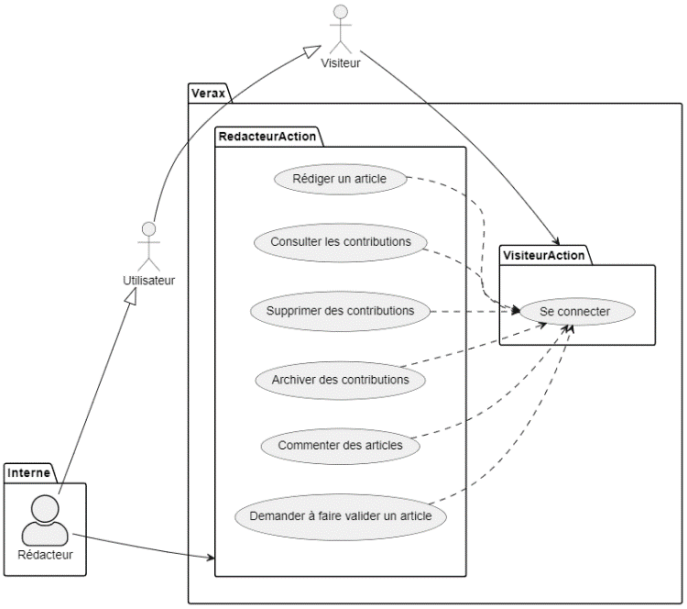

Notre diagramme de cas d'utilisation illustre les différents acteurs et leurs interactions avec la plateforme Verax. Les acteurs sont les visiteurs, utilisateurs, rédacteurs, modérateurs et administrateurs. Le diagramme est découpé en plusieurs paquets qui regroupent les actions spécifiques pour chaque type d'utilisateur. Nous allons détailler l'intérieur des paquets dans la suite de la description, nous avons choisi de faire de cette manière afin de faciliter la compréhension mais le diagramme global est disponible à la fin de cette description.




Ici, vous pouvez voir les actions possibles par un simple visiteur, c'est-à-dire quelqu'un qui ne s'est pas connecté. Un visiteur peut effectuer différentes actions telles que filtrer par thème, voir un article, rechercher un article, créer un compte et se connecter.

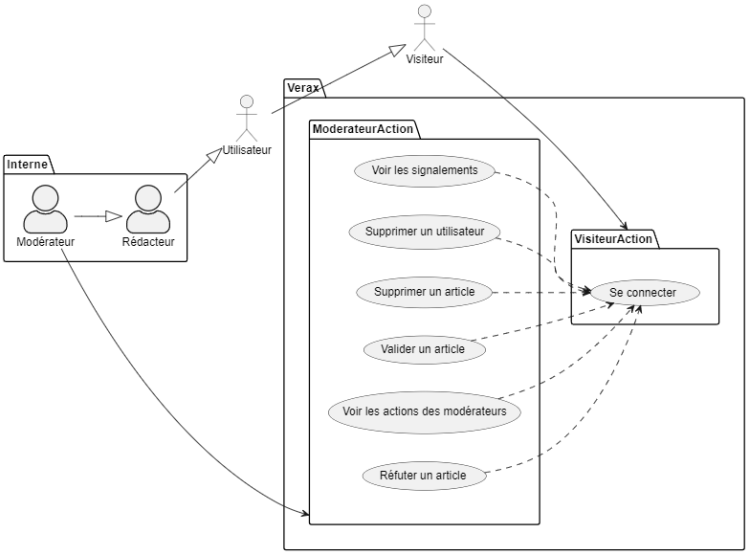
Cette partie du diagramme représente les actions disponibles pour un utilisateur sur la plateforme Verax. Un utilisateur est un Visiteur qui s'est connecté, il a donc la possibilité de faire les mêmes actions qu'un Visiteur et d'autre encore. En effet, il peut évaluer des articles, contribuer en soumettant du contenu, modifier son profil, signaler des articles problématiques, se déconnecter et supprimer son profil. Toutes ces actions sont accessibles une fois que l'utilisateur est connecté à la plateforme.



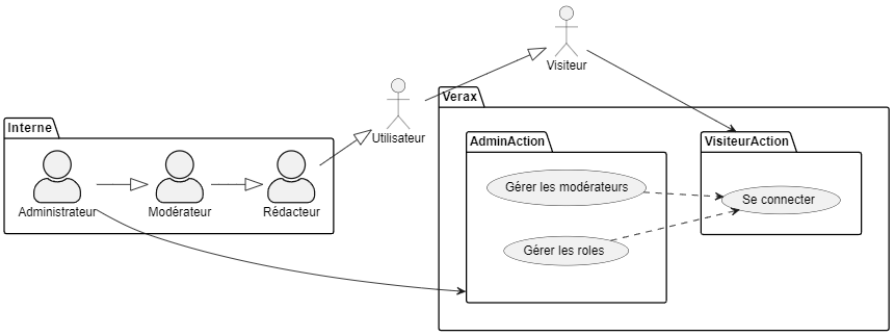

Cette partie décrit les actions disponibles pour un Rédacteur sur la plateforme Verax. Les Rédacteurs peuvent rédiger des articles, consulter leurs contributions, supprimer ou archiver des contributions précédentes, commenter des articles et demander la validation d'un nouvel article. Toutes ces actions sont accessibles une fois que le Rédacteur est connecté à la plateforme après s'être initialement connecté via l'action "Se connecter" du package "VisiteurAction".

Un Rédacteur a aussi la possibilité de faire toutes les actions que possède l'Utilisateur puisqu'il en hérite.




Cette partie du diagramme illustre les actions accessibles pour un Modérateur. Les Modérateurs héritent des actions des Rédacteurs, ils ont donc accès aux mêmes fonctionnalités qu'eux avec d'autres en plus. Une fois connectés, les Modérateurs peuvent voir les signalements d'articles, supprimer des utilisateurs ou des articles, valider des articles soumis par les rédacteurs, consulter les actions effectuées par d'autres modérateurs et réfuter des articles si nécessaire.

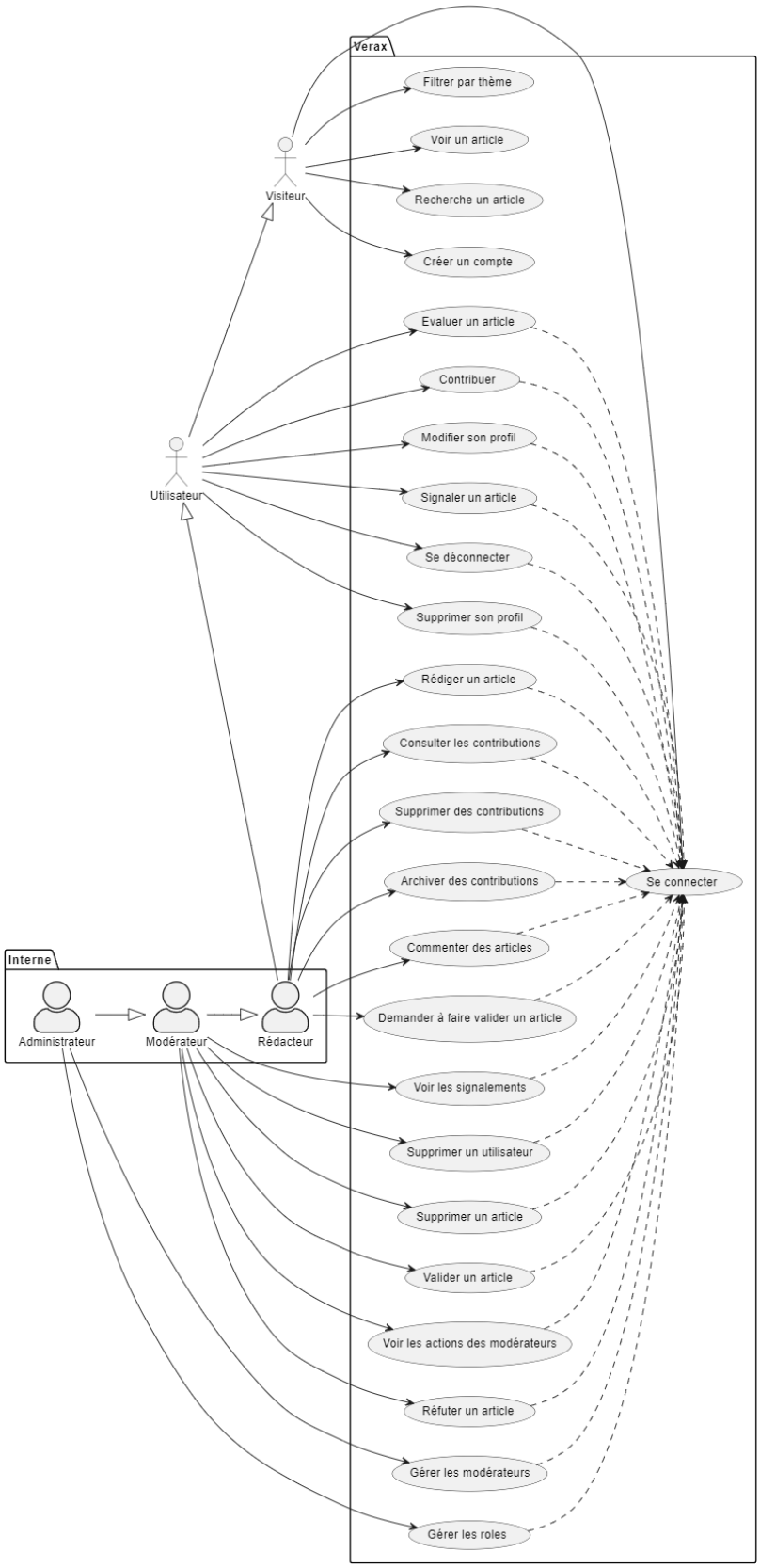
Ici, vous pouvez voir les actions accessibles pour un administrateur. Les administrateurs ont la capacité de gérer les modérateurs et les rôles sur la plateforme.



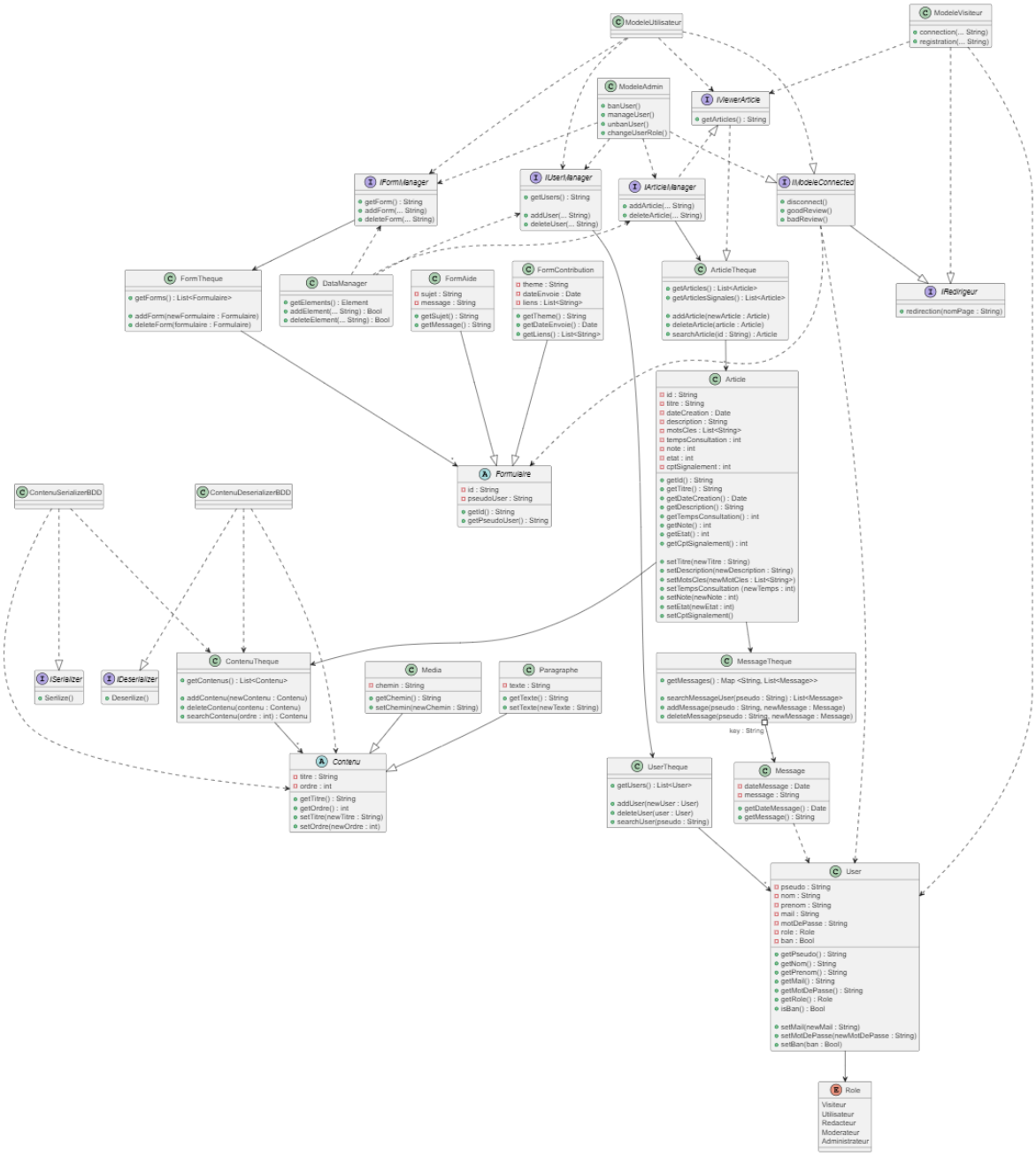

Ces actions sont accessibles une fois que l'administrateur est connecté à la plateforme après s'être initialement connecté via l'action "Se connecter" du package "VisiteurAction". Les Administrateurs héritent des actions du Modérateur, on peut donc dire que l'Admin a la possibilité de faire toutes les actions décrites précédemment.

Haut du formulaire

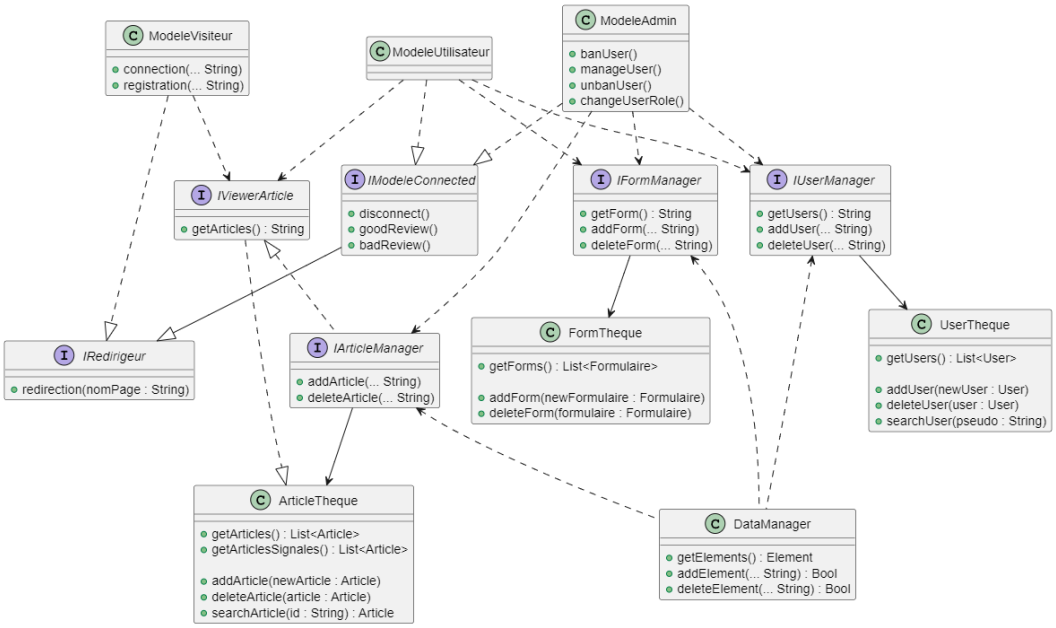
Voici le diagramme dans sa globalité :



# Diagramme de classe :




Notre diagramme de classe contient beaucoup de classe pour faire marcher notre application. Pour la description, nous allons encore une fois la découper en plusieurs parties afin de faciliter la compréhension.




Cette partie du diagramme de classe illustre l'organisation des différentes fonctionnalités et dépendances entre les classes et interfaces avec des modules spécifiques pour différents rôles utilisateurs. Le Redirigeur est une interface responsable de la redirection des pages, la page est reçue en paramètre de la fonction *redirection*.

Le ModeleVisiteur implémente IRedirigeur car ces fonctionnalités sont accessibles à tous les utilisateurs, la classe fournit aussi des fonctionnalités de connexion et d'inscription et dépend de IViewerArticle pour afficher les articles.

Le ModeleConnected est une interface centrale pour les utilisateurs connectés, dérivée de IModeleConnected, héritée par ModeleUtilisateur et ModeleAdmin car ces deux types d'utilisateurs ont besoin de ces fonctionnalités une fois connectées.

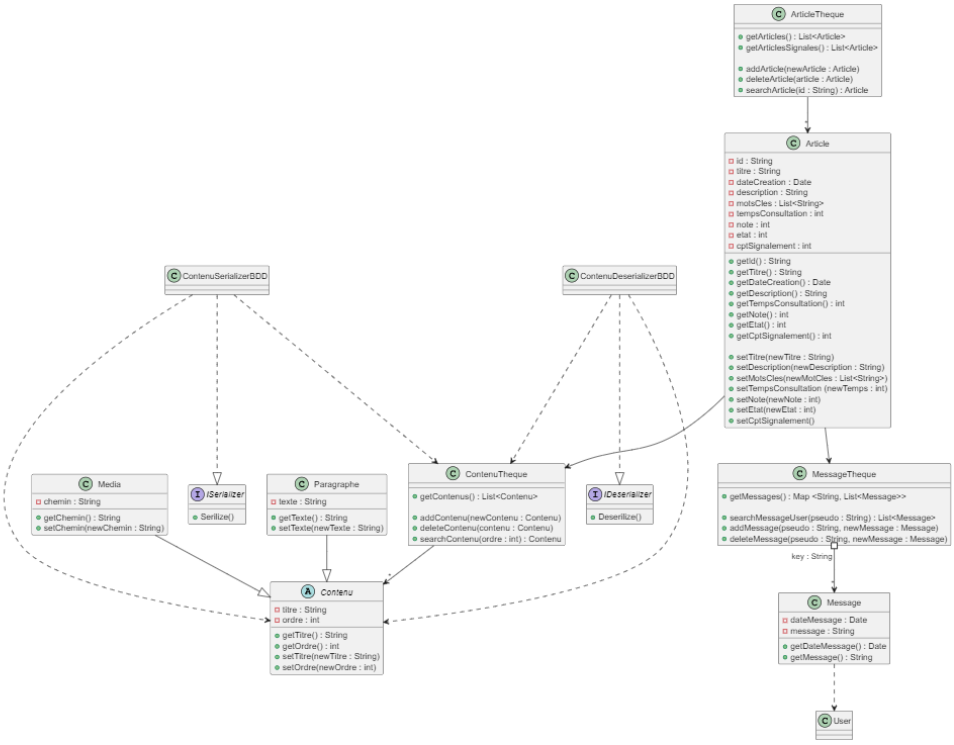
Le ModeleUtilisateur partage des fonctionnalités similaires à ModeleVisiteur mais avec des fonctionnalités supplémentaires comme l'accès à IFormManager pour permettre d'ajouter des formulaires et IUserManager pour la gestion du compte utilisateur.

Le ModeleAdmin implémente IModeleConnected car il a accès aux mêmes fonctionnalités que les personnes connectées. Ce modèle a accès à toutes les autres interfaces qui permettent de gérer les utilisateurs, les articles et les formulaires.

Les interfaces de gestion IFormManager, IUserManager, IArticleManager sont des interfaces utilisées par les différents modèles pour gérer les formulaires, les utilisateurs et les articles respectivement.

Il y a aussi les classes de gestion FormTheque, UserTheque, ArticleTheque qui permettent la gestion de leurs objets respectifs.

Pour finir, le DataManager est une interface générique utilisée pour manipuler les éléments de données.



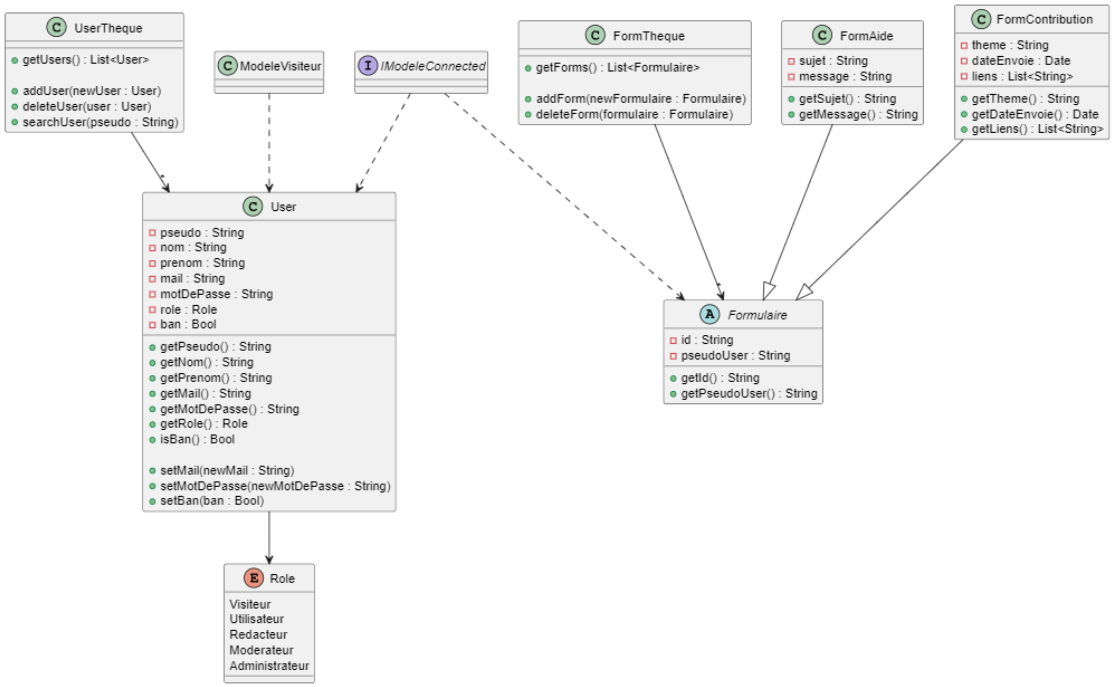

Cette partie du diagramme de classe représente l'endroit où l'on trouve la gestion des contenus, des articles et des messages, ainsi que des opérations de sérialisation/désérialisation des contenus.

La classe ContenuTheque gère une collection de contenus et fournit des méthodes pour récupérer, ajouter, supprimer et rechercher des contenus. La classe Contenu est une classe abstraite représentant un contenu générique avec des propriétés communes telles que le titre et l'ordre, avec des méthodes pour les récupérer et les modifier. Les Médias et les Paragraphes sont tous deux des classes qui représentent différents types de contenus spécifiques. Elles héritent de la classe abstraite Contenu.

La classe Article représente un article avec des attributs tels que l'ID, le titre, la date de création, etc. La classe ArticleTheque gère une collection d'articles et fournit des méthodes pour gérer ces articles.

La classe MessageTheque gère les messages utilisateurs sous forme de dictionnaire avec des listes de messages pour chaque utilisateur. La classe Message représente un message avec une date et un contenu.

Les interfaces de sérialisation ISerializer et IDeserializer représentent des interfaces pour la sérialisation et la désérialisation des contenus, les classes ContenuSerializerBDD et ContenuDeserializerBDD sont associées à ces interfaces, indiquant leur utilisation pour stocker des contenus en base de données.




Ce segment du diagramme représente la gestion des utilisateurs, des rôles et des formulaires.

La classe User représente un utilisateur avec des attributs tels que le pseudo, le nom, le prénom, l'email, le mot de passe, le rôle et un indicateur de bannissement. Il a des méthodes pour récupérer et modifier ses attributs.

La UserTheque permet de gérer une collection d'utilisateurs et fournit des méthodes pour ajouter, supprimer et rechercher des utilisateurs.

L'énumération Role représente les différents rôles qu'un utilisateur peut avoir dans le système : Visiteur, Utilisateur, Rédacteur, Modérateur, Administrateur.

Classe abstraite Formulaire représente un formulaire générique avec des attributs communs tels que l'ID et le pseudo de l'utilisateur créateur, avec des méthodes pour récupérer ces attributs. Les classes FormAide et FormContribution sont concrètes, elles représentent des types spécifiques de formulaires, l'un pour demander de l'aider ou de signaler un bug, et l'autre pour les contributions, avec des attributs spécifiques à chaque type et des méthodes pour les récupérer.