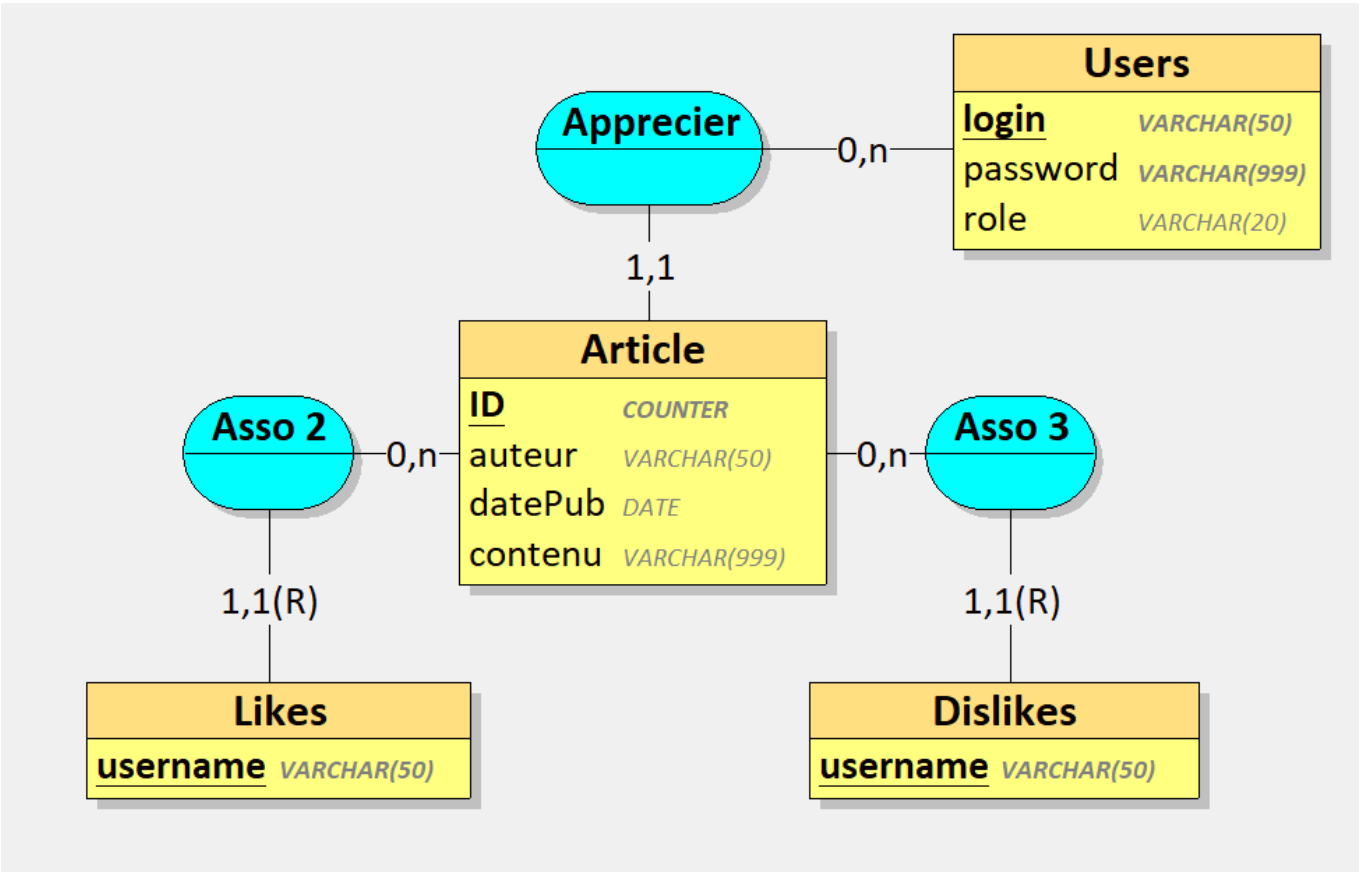
Gestion d'article de blog

APIREST



Procédure de configuration

Téléchargez le projet

Mettre le dossier dans xwamp/htdocs de votre serveur

Créer une base de données gestionarticle_rest dans MySQL

Importer le fichier gestionarticle_rest.sql

Modifier ensuite les paramètres de connexion pour qu'ils correspondent au votre

Vous pourrez accéder à l'API sur http://localhost:8080/APIREST-article-de-blog/controller/api.php

Ainsi que l'authentification sur http://localhost:8080/APIREST-article-de-blog/controller/auth.php

Lien Github : https://github.com/DiAmina/APIREST-article-de-blog

Présentation des User Story :

En tant que Publisher, on peut :

- Publier des articles
- Modifier/supprimer ses propres articles
- Liker/disliker les articles des autres utilisateurs

En tant que Moderator, on peut :

- Voir le nom, la date et le contenu d'un article
- Voir le nombre de like/dislike d'un article
- L'API permet de lister les articles publiés
- L'API permet de lister les articles publiés par un utilisateur

En tant que visiteur, on peut :

- Voir le nom, la date et le contenu d'un article

Nous avons utilisé la structure MVC dans notre projet, on retrouve donc un répertoire Controller, View et Model.

Pour l'authentification on a une classe qui est la représentation de la table user en sql et article de la table article en sql.

Puis 2 fichiers dans model pour les requêtes de l'article et d'user.

Dans la classe User on a les méthodes :

- **_construct** qui prend en entrée les paramètres $login, $password et $role. Cette méthode permet de créer un login, un password ainsi qu'un rôle.

- **getLogin** qui retourne le nom d'utilisateur.

- **getPassword** qui retourne le mot de passe.

- **getRole** qui retourne le rôle de l'utilisateur

- **str** est un string qui permet la concaténation du login, du password ainsi que du rôle

- **toUser est de type User et prend en entrée un tableau $user. Cette méthode** affecte un login, un password et un rôle à un utilisateur

- **isModerator est un booleen qui vérifie si le rôle est bien moderator**

- **isPublisher est un booleen qui vérifie si le rôle est bien publisher**

- **isAnonymous est un booléen qui vérifie si le rôle est bien anonymous**

Dans la classe Article on a les méthodes :

- _construct qui prend en entrée les paramètres $id, $auteur, $datePub, $contenu et $login. Cette méthode permet de créer un identifiant, un auteur, une date de publication, un contenu ainsi qu'un login.

- getId qui retourne l'id de l'article.

- getAuteur qui retourne l'auteur de l'article.

- getDatePub qui retourne la date de publication de l'article.

- getContenu qui retourne le contenu de l'article.

- getLogin qui retourne le login de l'article.

- **str** est un string qui permet la concaténation de l'id, de l'auteur, de la date de publication et du contenu.

- setContenu qui prend en entrée $contenu permet d'assigner du contenu.

- isAuteur est un booléen qui prend en entrée $auteur de type User et $article de type Article et qui retourne vrai si l'utilisateur est l'auteur de l'article.

Les différentes méthodes retournées :

- Si on veut vérifier que le token soit présent, en sortie on aura la méthode get_bearer_token.

- Si on veut vérifier que le token soit valide, en sortie on aura la méthode is_jwt_valid.

- Si on veut retrouver le role, en sortie on aura la méthode getPayload.

- La méthode getArticleID permet de retourner les informations d'un article donné. Si on rentre un id, en sortie on aura un article, sinon on a un code erreur 200.

- La méthode getLikes permet d'afficher le nombre de likes.

Si l'utilisateur est un moderateur, il peut voir qui a mis un like sinon il affiche le nombre de likes.

- La méthode getDislikes permet d'afficher le nombre de dislikes.

Si l'utilisateur est un moderateur, il peut voir qui a mis un dislike sinon il affiche le nombre de dislikes.

- La méthode getArticles permet de faire la liste des articles.

- La méthode postArticles permet de créer des articles.

Si on est un publisher, on peut créer un article sinon code erreur 401.

- Si on veut mettre à jour un article, on aura la méthode putArticle.

- La méthode deleteLike permet de supprimer un like.

Si l'utilisateur a mis un like, il peut le supprimer.

- La méthode deleteDislike permet de supprimer un dislike.

Si l'utilisateur a mis un dislike, il peut le supprimer.

- La méthode deleteArticle permet de supprimer un article.

Si l'utilisateur est l'auteur de l'article, il peut le supprimer sinon code erreur 401.

- Si on veut afficher la réponse, on aura la méthode deliverResponse.

- La méthode getUser permet d'afficher les informations de l'utilisateur.

Si on veut afficher les informations de l'utilisateur, on doit rentrer un login et un password.

- La méthode getRole permet d'afficher le rôle de l'utilisateur.

- Si on veut générer le token jwt, on aura la méthode generate_jwt.

POST Publisher	http://localhost:8080/APIREST-article-de-blog/controller/auth.php
POST Moderator	http://localhost:8080/APIREST-article-de-blog/controller/auth.php
GET Article (all)	http://localhost:8080/APIREST-article-de-blog/controller/api.php
GED Article (ID)	http://localhost:8080/APIREST-article-de-blog/controller/api.php?id=12
POST Article	http://localhost:8080/APIREST-article-de-blog/controller/api.php
POST Like	http://localhost:8080/APIREST-article-de-blog/controller/api.php?id=12
POST Dislike	http://localhost:8080/APIREST-article-de-blog/controller/api.php?id=12
DELETE like	http://localhost:8080/APIREST-article-de-blog/controller/api.php?id=12&username=crococarl
DELETE Dislike	http://localhost:8080/APIREST-article-de-blog/controller/api.php?id=1&username=crococarl
DELETE Article	http://localhost:8080/APIREST-article-de-blog/controller/api.php?id=12
PUT Article	http://localhost:8080/APIREST-article-de-blog/controller/api.php?id=12
PATCH Like	
PATCH Dislike	

La liste des utilisateurs authentifié

Moderators

Login	Mot de passe
amina	iutinfo
clement	iutclement
prof	$iutinfo


```
{
    "username":"amina",
    "password":"iutinfo"
}
{
    "username":"clement",
    "password":"iutclement"
}
```


Publishers

Login	Mot de passe
crococarl	hihihiha
vinki	inki000
patrick	patoche
Johnny	hallyday
viallet	viallet456


```
{
    "username":"vinki",
    "password":"vinki000"
}
{
    "username":"crococarl",
    "password":"hihihiha"
}
```


Pour mettre à jour un contenu


```
{
"contenu":"Teste 2.0"
}
```


Pour ajouter un article (à noter : l' username doit être égal à l’auteur)

```
{
    "auteur":"crococarl",
    "contenu":"Bonjour, passez une excellente journée"
}
```


Pour ajouter un like/dislike à un article autre que le sien


```
{
    "id":14
}
```


Présenter les différentes url pour chaque ressource (spécifier l'action « GET ») suivie d'un screen

L'action du GET sur la méthode getArticleAll


```
"status": 200,
    "status_message": "Liste des articles",
    "data": [
        {
            "ID": "2",
            "auteur": "vinki",
            "datePub": "2023-03-24",
            "contenu": "Contempler le coucher de soleil, fait parti des mes activités favoris"
        },
        {
            "ID": "3",
            "auteur": "crococarl",
            "datePub": "2023-03-17",
            "contenu": "Singapour !"
        },
…
```