

Sylvain Tran et Kevin Thai

Groupe de TP : 7

# Rapport : VidéoFlex



Lien permettant d'accéder à l'accueil du site : <https://etudiant.u-pem.fr/~sylvain.tran/accueil.php>

\*Adresse e-mail et mot de passe pour tester le site (lors de l'authentification) :

-Adresse mail : Bilolo@gmail.com

-Mot de passe : aaa

\*Et voici éventuellement quelque série et film pour tester la barre de recherche :

-Shrek, Venom, Alibaba, FBI

## SOMMAIRE:

### 1) Introduction

### 2) Manuel utilisateur

### 3) Présentation des étapes 1 et 2 du projet et modifications apportées

### 4) Description technique des fonctionnalités du site

### 5) Eventuelles pistes d'améliorations et problèmes rencontrés

### 6) Organisation et répartitions des tâches

### 1) Introduction

Nous devons implémenter un site de streaming de film et série nommé VidéoFlex (comme Netflix par exemple) en utilisant une base de données par le biais de PDO et PHP. (Pour accéder au site, lien ci-dessus)

### 2) Manuel utilisateur

Une fois sur la page d'accueil du site, l'utilisateur aura la possibilité de s'inscrire ou de s'identifier s'il possède déjà un compte. Pour s'identifier, l'utilisateur devra entrer une adresse mail avec le domaine @gmail.com ainsi qu'un mot de passe.

Une fois identifié, celui-ci aura accès à une page « Mon compte » où il pourra accéder à ses informations personnelles, supprimer, ajouter un profil et accéder à la page de ses différents profils s'il possède un compte premium (jusqu'à 4 profils) ou à son unique profil s'il possède un compte normal.

Sur la page d'un profil, l'utilisateur pourra reprendre la lecture d'une vidéo à l'endroit précis où il s'était arrêté. Des suggestions de vidéos lui seront proposées en fonction de ses précédents visionnages ainsi qu'un top des films du moment. Il pourra également chercher une série ou un film depuis la barre de recherche.

Sur la page d'une vidéo, celui-ci aura accès aux informations concernant la vidéo (à quelle série elle appartient si c'est un épisode, son réalisateur, les acteurs qui ont joué dedans, etc.). Sur la même page, l'utilisateur pourra visionner, reprendre, noter la vidéo (s'il ne l'a pas déjà fait) et lui attribuer des labels.

De plus, des boutons « retour » seront placés en haut à gauche de toutes les pages (sauf à l'accueil et dans la page Mon compte) lui permettant de revenir aux pages précédentes et un bouton déconnecter pour se déconnecter.

### 3) Présentation des étapes 1 et 2 du projet et modifications apportées

Dans l'étape 1 et 2 du projet, nous devons réaliser un schéma Entité-Association répondant au cahier des charges du client puis transformer ce schéma en modèle relationnel.

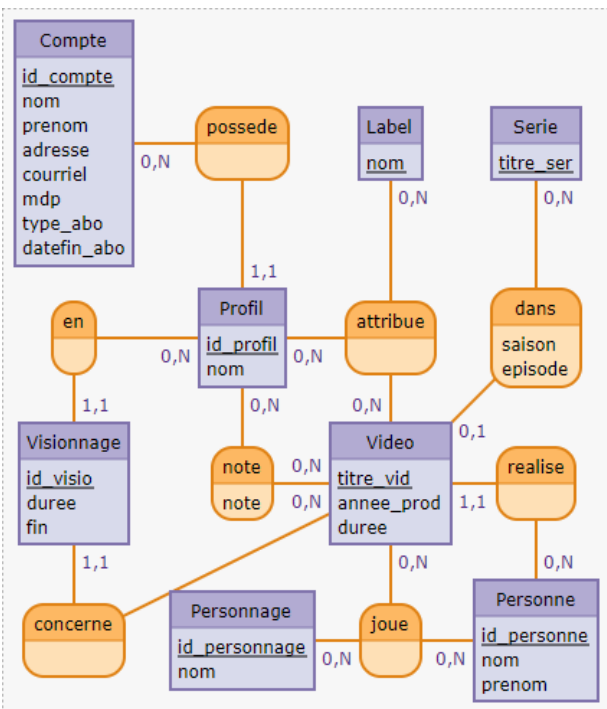
Pour la conception du site, nous avons décidé de changer deux tables : Compte et Profil en ajoutant 1 attribut à chacun : l'attribut « mdp » dans Compte et « nom » dans Profil.

On a jugé que c'était mieux pour l'authentification d'avoir un mot de passe et que chaque profil ait un nom que l'utilisateur peut choisir.

On a également enlevé les tables Serie et Label dans le fichier SQL où on a créé les tables car elles étaient inutiles dans le sens où elles ne contenaient qu'un seul attribut et que cet attribut était dans une autre table en tant que clé étrangère.

De plus, on a aussi changé le type de données de 2 attributs : « type\_abo » de la table Compte et « fin » de la table Visionnage en Integer pour avoir des données plus compactes et faciles à écrire.

Donc, voici le schéma Entité-Association et le modèle relationnel final :



**COMPTE** ( id\_compte, nom, prenom, adresse, courriel, mdp, type\_abo, datefin\_abo )  
**PROFIL** ( id\_profil, nom, *id\_compte* )  
**ATTRIBUE** ( id\_profil, nom, titre\_vid )  
**VISIONNAGE** ( id\_visio, duree, fin, *id\_profil*, *titre\_vid* )  
**NOTE** ( id\_profil, titre\_vid, note )  
**VIDEO** ( titre\_vid, annee\_prod, duree, *titre\_ser*, saison, episode, *id\_personne* )  
**PERSONNAGE** ( id\_personnage, nom )  
**JOUE** ( id\_personne, id\_personnage, titre\_vid )  
**PERSONNE** ( id\_personne, nom, prenom )

#### 4) Description technique des fonctionnalités du site

Voici les principales fonctionnalités du site :

- 1) **S'inscrire** Bouton qui permet à l'utilisateur de s'inscrire si celui-ci ne possède pas de compte
- 2) **S'identifier** Bouton qui permet à l'utilisateur de s'identifier et donc d'accéder à son compte
- 3) **déconnexion** Bouton pour se déconnecter (fait revenir à la page d'accueil)
- 4) **Information sur le compte** Bouton qui permet d'accéder aux différentes informations de l'utilisateur c'est-à-dire : nom, prénom, adresse, etc.

- 5) Nom du profil:   
 Ajouter Supprimer  
 Permet de supprimer ou d'ajouter un profil

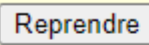
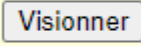
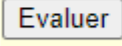
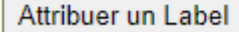
- 6) **retour** Bouton permettant de revenir aux pages précédentes

Barre de recherche pour pouvoir chercher une série ou un film

- 7) Search:  Recherchez une série ou un film

Aucun résultat pour Pikachu dans l'espace

(Un exemple de retour si la série ou le film entré n'existe pas)

- 8)  Bouton qui permet de reprendre la lecture d'une vidéo au moment où il s'était arrêté la dernière fois
- 9)  Bouton pour visionner la vidéo (la vidéo part du début)
- 10)  Bouton qui permet de noter la vidéo (entre 0 et 10)
- 11)  Bouton pour attribuer un label à la vidéo

Voici les fonctions et requêtes les plus intéressantes que nous avons utilisé :

1)

Nous avons utilisé la fonction mktime et la fonction date pour la date de fin d'abonnement qui est 1 ans après l'inscription :

```
//date de fin d'abonnement => 1 ans après
$annee = mktime(0, 0, 0, date("m"), date("d"), date("Y")+1);
$d_abo = date("Y-m-d", $annee);
```

Avec date (« m »), date (« d »), Date (« Y ») +1, on obtient la date courante avec l'année + 1 et date (« Y-m-d ») permet d'avoir une date avec le format Année-Mois-Jour (2022-12-05 par exemple).

2)

Pour générer des clés primaires différentes (ou id) pour l'insertion de données dans la table profil par exemple, on a utilisé cette requête :

```
$last_id = $cnx->query("SELECT max(id_profil) FROM profil");
$val = $last_id->fetch();
```

C'est la méthode qu'on a trouvée, on prend à chaque fois la valeur maximum avec l'agrégat max() qu'on incrémente par la suite de 1.

```
$exec = $prep->execute(array(':idp' => $val["max"]+1, ':nom' => $nom, ":idc" => $id_compte));
```

Comme ça, on n'enfreint jamais une contrainte de clé primaire déjà existante.

3)

Une des requêtes les plus technique qu'on a utilisé est celle-ci :

```
//label le plus fréquent
$label = $cnx->query("SELECT nom, count(nom) FROM attribue WHERE id_profil = $id_profil GROUP BY nom ORDER BY count(nom) DESC LIMIT 1");
```

Par label, on récupère le nombre de fois où il apparait dans la table attribue où le profil est \$id\_profil. On les trie ensuite par ordre décroissant et on limite le résultat de la recherche à la première ligne.

C'est une requête qui permet de récupérer le label le plus fréquent parmi les films, séries regardées par l'utilisateur (dont l'id du profil est donné par \$id\_profil).

## 2 Remarque :

Pour la lecture des vidéos, nous avons utilisé la balise <video> :

```
<video width='500' height='500' controls>
  <source src='video.mp4#t=".$val["duree"]."' type='video/mp4'>
  La video ne peut pas être lu
</video>
```

Nous avons décidé de prendre 1 vidéo (le fichier video.mp4) pour représenter toutes les vidéos de notre base de données. On a fait cela car sinon il y aurait trop de vidéo à importer, ça ferait lourd. Mais si on l'avait fait, on aurait pour chaque fichier .mp4 nommé celui-ci avec le titre de la vidéo

(Par exemple, Venom.mp4 pour le film Venom) et donc au lieu de faire ça :

```
<source src='video.mp4#t=".$val["duree"]."' type='video/mp4'>
```

On aurait fait ça :

```
<source src='".$titre_vid.".mp4#t=".$val["duree"]."' type='video/mp4'>
```

Où \$titre\_vid représente le titre de la vidéo.

De plus, pour faire en sorte que la vidéo commence à un instant T précis (pour les utilisateurs qui veulent reprendre leur vidéo là où il s'était arrêté), nous avons mis un #t juste après le nom du fichier .mp4. Et donc par exemple #t00:05:00 permet de démarrer la vidéo à 5 minutes.

Nous avons ajouté des boutons « retour » aux pages car nous avons beaucoup de formulaire qui renvoie des données sur la même page et cela fait que quand on appuis sur la flèche gauche en haut à gauche du navigateur pour revenir en arrière, on a souvent des messages d'erreurs du type : « Confirmer le nouvel envoi du formulaire ».

Et donc en ajoutant ce bouton, on a réglé ce problème.

## 5) Eventuelles pistes d'améliorations et problèmes rencontrés

### 1)

Nous n'avons pas réussi à récupérer le temps courant d'une vidéo. Quand l'utilisateur arrête une vidéo à un instant T précis, on ne s'est pas comment récupérer ce temps. Et donc cela fait qu'à chaque fois qu'un utilisateur veut reprendre son visionnage là où il s'était arrêté, il ne peut pas. Soit il repart du tout début, soit au temps qu'on a initialement mis dans la base de données.

```
INSERT INTO Visionnage VALUES (2, '00:01:05', 0, 4, 'Prendre les armes');
```

C'est-à-dire ici à 1 minute 5 pour la vidéo dont le titre est « Prendre les armes » pour le profil 4. S'il reprend son visionnage et s'arrête par exemple à 1 minute 50, qu'il quitte la page pour ensuite revenir. La vidéo repart à 1 minute 5 et pas à 1 minute 50.

2)

Pour les adresses mail lors de l'authentification, on a gardé que le domaine @gmail.com alors qu'il en existe des centaines : @yahoo.fr, @hotmail.fr, etc. On ne sait pas comment faire pour vérifier tous les domaines (sans faire des centaines de condition if).

D'ailleurs, pour vérifier si le domaine @gmail.com est présent, on a utilisé la fonction strpos :

```
$cond = strpos($mail, '@gmail.com');
```

Qui renvoie False si la chaîne '@gmail.com' n'est pas présent dans l'adresse mail \$mail.

Donc, ce n'est pas très optimisé et on force un peu le client à se créer un compte gmail (dans le cas où il n'en possède pas un) pour pouvoir s'inscrire à VideoFlex.

3)

(Par manque de temps, on n'a pas eu le temps de faire un site un peu plus beau. On a juste mis un logo dans la page d'accueil.)

## 6) Organisation et répartitions des tâches

Nous avons fait tous ensemble en vocal sur discord :

Mais Sylvain s'est plus occupé de la gestion des vidéo (barre de recherche, séries, lecture de vidéo, note, labels)

Et Kevin de l'accueil, l'authentification, l'inscription, la gestion des comptes et des profils.