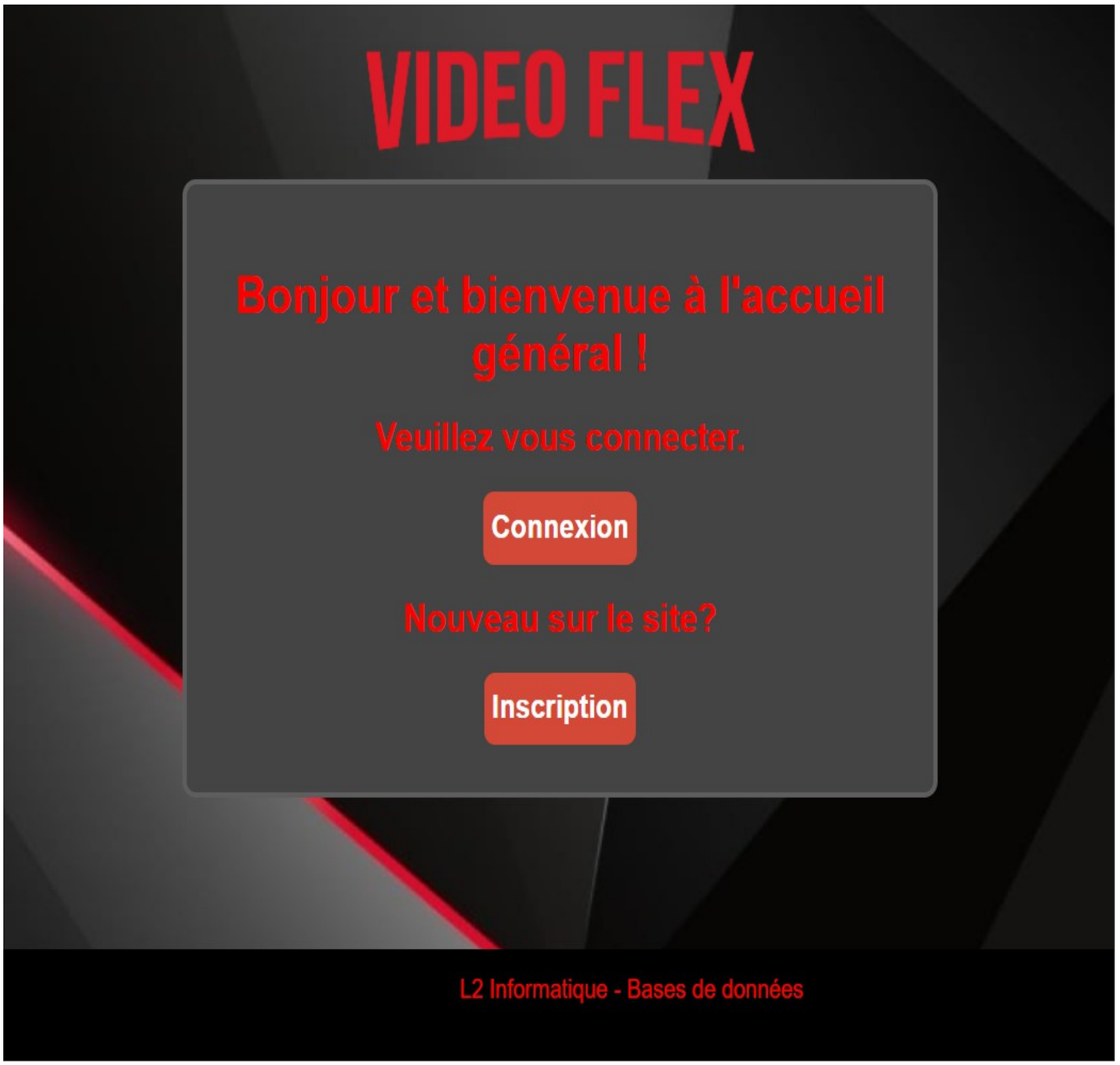
VidéoFlex



Lien du site: *https://etudiant.u-pem.fr/~bdiagn01/Projet/WebFinal/accueil.php*

Présentation

Qu'est-ce que c'est ?

VidéoFlex est un site de streaming qui propose des films et séries. Son but est d'offrir l'expérience la plus ergonomique et agréable possible pour tout ses clients.

Ses fonctions

Pour cela, il liste toutes les informations sur tout les films et séries qu'il propose, possède une mécanique de sauvegarde qui permet à l'utilisateur de reprendre son visionnage à là où il s'est arrêté et enfin, il établit également une système de notation avec les labels que les clients peuvent attribuer à un film, ce qui permet au site de pouvoir suggérer des films à ses clients en visant les labels qui correspondent le mieux aux centres d'intérêt de ce client.

Guide de l'utilisateur

Vous devez d'abord vous identifier en saisissant votre nom d'utilisateur ainsi que le mot de passe. Si vous n'avez toujours pas de compte, vous pouvez toujours vous inscrire en cliquant sur le bouton « Inscription ».

Une fois connecté, le site vous renvoie sur la page d'accueil qui affiche les nouveautés ainsi que les suggestions proposées par le site. Vous avez le choix entre rechercher un autre film/série pour le regarder, lui assigner un label/noter, ou alors reprendre votre visionnage si vous avez déjà entamé la simulation.

Si vous souhaitez quitter le site, n'oubliez pas de vous déconnecter.

Partie du développeur

Schéma entité-association

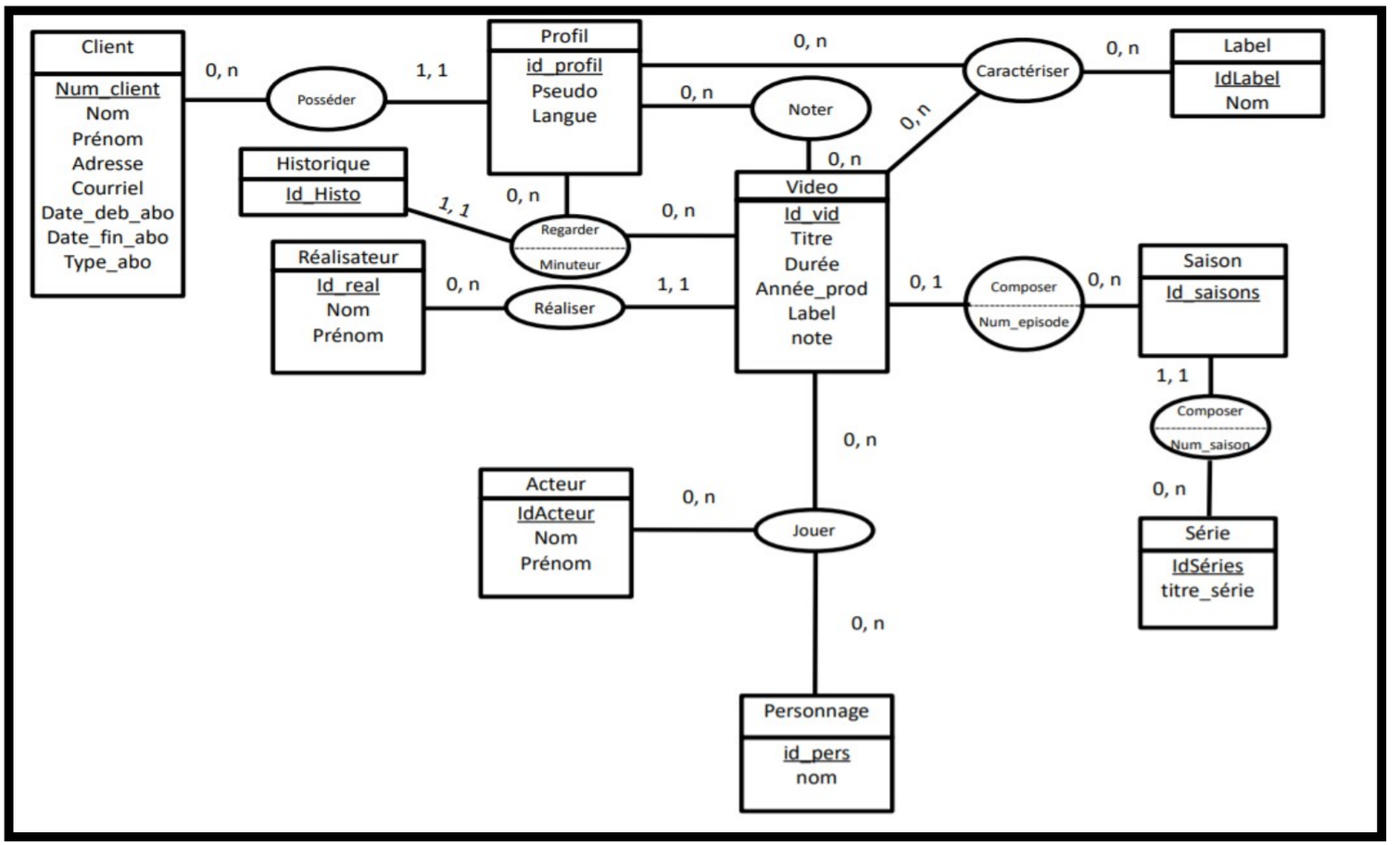


Schéma relationnel

Client (Num_client, Nom, Prénom, Adresse, Courriel, Date_deb_ao, Date_fin_abo, Type_abo)
Profil (id_profil, Num_client, Pseudo, Langue)
Caractériser (id_profil, idLabel, Id_vid)
Noter (Id_profil, Id_vid)
Label (idLabel, Nom)
Video (Id_vid, Id_Real, Titre, Durée, Année_prod, Label, note, Id_saisons, Num_episode)
Historique(Id_histo, Id_vid, Id_profil, Minuteur)
Realisateur (Id_real, Nom, Prénom)
Saison (Id_saisons, IdSeries, Num_saison)
Serie (IdSeries, titre_serie)

Jouer (Id_vid, IdActeur, Id_pers)
Acteur (IdActeur, Nom, Prenom)
Personnage (Id_pers, nom)

Clés étrangères (FK) :
Profil(Num_client) fait référence à Client(Num_client)
Caracteriser(Id_profil) fait référence à Profil(Id_profil)
Caracteriser(IdLabel) fait référence à Label(IdLabel)
Caracteriser(Id_vid) fait référence à Vidéo(Id_vid)
Noter(Id_profil) fait référence à Profil(Id_profil)
Noter(Id_vid) fait référence à Vidéo(Id_vid)
Histo(id_profil) fait référence à Profil(Id_profil)
Histo(Id_vid) fait référence à Vidéo(Id_vid)
Video(Id_saisons) fait référence à Saison(Id_saisons)
Saison(IdSeries) fait référence à Série(IdSéries)
Jouer(Id_vid) fait référence à Vidéo(Id_vid)
Jouer(IdActeur) fait référence à Acteur(IdActeur)
Jouer(Id_pers) fait référence à Personnage(Id_pers)

Types de données & contraintes

Pour toutes les clés primaires on a choisi le type unsigned Integer, nous donnant ainsi une valeur maximum élevée et positive toutes valeurs de type numérique sont elles aussi signées afin d'éviter qu'elles soient négative.

Table Client:

On a attribué la contrainte NOT NULL aux attributs prenom, nom et adresse. Ces derniers sont en effet obligatoires pour toute personne souhaitant s'enregistrer sur notre futur site (elles permettront de vérifier l'identité d'un client en cas de problème comme par exemple le piratage d'un compte). On a ensuite attribué une contrainte sur l'attribut type_abo, ce dernier doit être soit « premium » soit « normaux » car ce sont les deux types d'abonnement disponible, cet attribut peut avoir la valeur NULL car un client peut très bien créer un compte sans avoir d'abonnement ou un client dont l'abonnement se termine n'a pas à être effacé de la base. On aussi contraint l'attribut «courriel» à contenir un «@» et un «.fr» ou un «.com», nous savons que d'autres types d'adresse mail existent cependant on se limitera à ces derniers. (Le reste des attributs a un type qui semble évident vu ce qu'ils représentent).

Nous avons aussi rajouté un attribut « mot de passe » non nul (tout les profils doivent avoir un mot de passe) qui n'était pas présent sur le schéma entité-association de base. En effet, cet attribut va être vital pour le site car elle va renforcer la sécurité du profil en limitant son accès. De plus, ce mot de passe est crypté par la fonction md5().

Table Video:

Pour l'attribut « titre » on a choisi un type varchar(255) qui peut paraître assez long mais des films comme « Night of the Day of the Dawn of the Son of the Bride of the Return of the Revenge of the Terror of the Attack of the Evil, Mutant, Alien, Flesh Eating, Hellbound, Zombified Living Dead Part 2 : In Shocking 2- D » de James Riffel ont un titre très long. Pour l'attribut "note", on vérifie si ce dernier est positif et supérieur à 0 car les notes des vidéos vont jusqu'à 10. Pour le numéro d'épisode on a choisi le type smallint car le nombre d'épisodes d'une série ne dépasse pas 32767 (même Les Feux de l'Amour ne possède que 11 585 épisodes), on a aussi ajouté une contrainte vérifiant si ce nombre est positif.

Table Label:

L'attribut "nom" doit être unique afin de ne pas créer de redondance inutile dans la table.

Table Profil :

Cette table est également liée à la table Client pour pouvoir faire lien entre le client et ses différents profils.

Les autres tables n'ont rien de particulier, à part le fait de reprendre les principes cités précédemment.

Autres:

Un problème a été rencontré pour la note des films, on a choisi une note arbitraire ne dépendant pas de la note attribuée par les différents profils. Il faudrait en réalité, faire les moyennes des notes présentes dans la table «noter» et attribuer ces dernières à la vidéo correspondante. Il en est de même pour le label des vidéos.

Un autre problème est la gestion du serial : Toutes nos tables ont un id ne possédant pas ce type, nous sommes donc forcés d'écrire les id manuellement. Modifier toutes les tables en remplacant le type de l'id par un serial n'est pas possible avec un simple alter column car serial est un pseudo type et nous sommes obligés de créer une séquence pour rendre ça possible. Tout re-modifer prendra beaucoup de temps donc on a décidé de le passer et d'utiliser DEFAULT pour les insertions de valeurs à la place.

Le site

Partie I: Authentification

On commence d'abord par l'authentification de l'utilisateur. Ce dernier a le choix entre se connecter et s'inscrire.

Inscription

On va d'abord prendre les valeurs entrées par l'utilisateur sur l'inscription à l'aide de la méthode POST, puis nous ajoutons ces valeurs dans la base de données grâce à cette requête :

$resultat = $cnx->prepare("INSERT INTO client(num_cli,nom,prenom,adresse,courriel,mdp) VALUES(DEFAULT,?,?,?,?,?);");

$resultat->execute(array($nom,$prenom,$adresse,$email,$mdp));

Ici, on prépare la requête en mettant des marqueurs sur les valeurs pour ainsi la compléter avec les valeurs récupérés par le post à l'aide de la méthode execute.

Pour l'abonnement, on clique sur le type d'abonnement qu'on souhaite payer. En faisant cela, on va stocker la valeur représentant l'abonnement dans une session pour les futurs tâches une fois l'inscription fini.

Enfin, le paiement n'a pas d'influence sur la base de données, elle a été crée juste pour l'immersion. Elle était censé faire parti d'une des attributs de la table client, mais nous avons voulu éviter le superflu de valeurs (car elle allait rester dans la table sans être utilisé une fois).

Connexion

La requête principale qui va nous permettre de se connecter sera la suivante:
$cnx->query("SELECT * FROM client WHERE courriel='".$email."'");
En entrant son adresse mail et son mot de passe pour se connecter, le code va récuperer les valeurs saisies via la méthode post, puis à l'aide de ces valeurs (qu'on stocke à des variables comme $email qui représente l'adresse mail), nous effectuons une requete SQL qui va sélectionner les informations du compte possédant cette adresse email. Par la suite, nous décomposons les résultats de la requete par un tableau avec fetchAll(), nous ciblons la valeur du mot de passe dans le tableau et enfin nous vérifions si elle est la même que le mot de passe saisie.

Partie II: Le menu

Une fois connecté à son profil, l'utilisateur a plusieurs possibilités :

Accueil

Dans l'accueil figure les nouveautés, qui sont les films de la base de données qui datent de moins de 30 jours. Pour cela, nous regardons pour chaque film sa date et comparons sa différence avec date actuelle donné par l'ordinateur.
On a ensuite la rubrique des suggestions qui possèdent des labels visant votre centre d'interêt. Pour ça, on éxécute de nouveau une requête SQL qui va nous permettre de chercher les séries qu'il a regardé. La requête va fouiller dans l'historique du profil connecté les vidéos qu'il a regardé. On selectionne ensuite le label de ces vidéos regardés et on selectionne les vidéos possédant ce même label.
On avait originellement prévu d'intégrer une autre rubrique « Au hasard » qui allait selectionner aléatoirement les vidéos présentés par le site. Au final, on ne gérait pas bien la fonction array_rand pour avoir ce qu'on voulait. On a donc abandonné l'idée.

Compte

Cliquer sur cet onglet nous ramène sur les paramètres du compte. Ici, on va juste extraire les données sauvegardés dans la session puis la mettre ici. Les paramètres nous donne l'option de modifier nos informations et aussi de créer ou supprimer des profils si nous sommes des abonnés premiums. Pour la modification d'information, nous reprenons la même méthode que pour l'inscription tandis que pour l'ajout et la suppression des profils, nous effectuons principalement ces requetes :

Ajout:
$cnx->exec("INSERT INTO profil VALUES (DEFAULT, '".$pseudo."',".
$_SESSION['login'].")");
Suppression :
$cnx->exec("DELETE FROM profil WHERE pseudo = '".$pseudo."'");

Films/Séries

Cette rubrique va lister toutes les films/séries que le site offre. Pour afficher les différents films/séries, nous allons effectuer une requête sur la base de données qui va prendre toutes les données de la table vidéo, ensuite décomposer ces données pour les mettre dans un tableau, puis parcourir ce tableau pour afficher son nom suivant d'un radiobutton (pour le choisir) en alliant le php avec le html. Pareil pour le choix des saisons/épisodes avec les séries.

Après que le choix du film/série a été faite, nous avons le choix entre simuler un visionnage, labeliser ou noter.

Simulation de visionnage

D'abord, on vérifie si le profil a déjà regardé cet épisode dans l'historique. Comme pour la liste des séries, nous effectuons une requête dans la table historique qui vérifie si le profil actuellement connecté et l'id de la vidéo figurent dans la même table.
Dans le cas ou c'est vrai, nous demandons à l'utilisateur s'il souhaite reprendre. Dans le cas contraire, on demande à l'utilisateur de rentrer la durée du visionnage qu'on souhaite

simuler. Ici, on va établir une limite qui va être de 00:00 à la durée du film/épisode en heures. On extrait cette dernière en effectuant une requête qui va afficher toutes les informations du film selectionné, y compris sa durée. La valeur rentrée va alors apparaître dans la table historique.

Label

Pour labeliser un(e) film/série, nous allons mettre en place un input de type texte. La valeur saisie sur cette barre de soumission va apparaître sur la table label qui va lister tout les labels. Pour caractériser ce label nouvellement ajoutée à la vidéo, nous devrons appeler cette requête :

// Pour créer le label
$resultat = $cnx->prepare("INSERT INTO label VALUES(DEFAULT,?)");
$resultat->execute(array(Label_nom));
// Pour attribuer le label à la vidéo
$resultat = $cnx->prepare("INSERT INTO caracteriser VALUES(?,?,?)");
$resultat->execute(array(id_profil, id_vid, id_label);

Caractériser étant la table qui fait le lien entre le label, le profil attribuant le label et la vidéo.

Note

Enfin, on peut noter la vidéo. Pour cela, on crée une boucle qui va afficher les notes de 0 à 10 sous forme de bouton radio , mais avant cela, on vérifie si le profil a déjà noté cette épisode ou pas en éxécutant une requête qui va selectionner le nombre d'occurences de l'id du profil auquel on est connecté couplé avec l'id de la vidéo sur la table noter, et on vérifie si elle est nulle ou pas.

Si elle est nulle

Le profil lui a attribué aucune note. On affiche les notes et l'utilisateur n'aura qu'à choisir.

Si elle n'est pas nulle

Le profil lui a forcément déjà attribué une note. On annule alors l'affichage.

Organisation du travail

Puisque nous sommes pas dans le même groupe de TD/TP sur tout les autres matières autre que le TP de BDD et à cause d'une part de la grève nationale, la communication et les prises de rendez-vous furent assez compliqué et on a au final pu se voir en dehors des cours qu'une fois. Nous avons communiqué tout nos problèmes via Whatsapp et envoyé toutes nos traces de progression par mail.
En ce qui concerne la répartition du travail, le premier rendu était assez homogénisé sur la répartition étant donné qu'on faisait les mêmes tâches ensemble et qu'on se communiquait nos trouvailles. Durant le deuxième rendu, Bruny s'est focalisé sur la création de la base de données tandis que Ben s'est focalisé sur le remplissage de la base.
Pour le dernier rendu, Bruny a consacré une grande part sur la partie PHP du site et Ben s'est occupé de son design ainsi que quelques autres pages PHP.

Modifications faites après la soutenance

-Nous sommes tombés sur une erreur durant la soutenance avec l'ajout et la suppression de profils. En effet, si l'utilisateur ajoute, puis supprime ce même profil ajouté, une erreur nous informant une violation de clé étrangère appairaissait car la clé étrangère n'avait pas la contrainte ON DELETE qui permettait d'effacer les autres clés référencés par l'id du profil en effacant l'id. La contrainte ON DELETE CASCADE a ainsi été ajoutée.

-Nous avons vidé toute les tables client et profil pour éviter le superflu.