IUT du Limousin / Département MMI / BUT1

SAE 2.03 | Concevoir un site avec une source de données

Compétences ciblées

Développer pour le web et les médias numériques

Objectifs et problématique professionnelle

Objectifs :

- Combiner les ressources liées au développement web et à la gestion des bases de données et comprendre comment elles s'articulent pour automatiser la production de pages Web ;
- Approfondir les ressources liées à l'intégration en lien avec les normes et les bonnes pratiques pour produire des pages fluides, valides et accessibles ;
- Découvrir le développement front et approfondir la connaissance du CSS en mettant en place des animations et interactions simples ;
- Découvrir l'administration d'un hébergement web en mettant en ligne un projet web et en veillant à la sécurité des données.

En tant que développeurs web juniors, les étudiants doivent concevoir un site web relié à une base de données comme par exemple un catalogue de produits organisé par catégories ou mots-clés. Ce site doit présenter de manière lisible et ergonomique les données stockées et permettre de naviguer au sein de celles-ci. Il doit permettre aux utilisateurs d'ajouter des contenus par exemple en postant des commentaires ou en réservant un produit. Les étudiants doivent répondre à la question : Comment consulter de manière fluide des contenus stockés dans une base de données et permettre aux utilisateurs d'interagir avec ces contenus ?

Apprentissages critiques

AC14.01 | Exploiter de manière autonome un environnement de développement efficace et productif
AC14.02 | Produire des pages web incluant un balisage sémantique efficace et des interactions simples
AC14.03 | Générer des pages web à partir de données structurées
AC14.04 | Mettre en ligne une application web en utilisant une solution d'hébergement standard
AC14.05 | Modéliser les données d'une application web

Ressources mobilisées et combinées

R2.12 | Intégration
R2.13 | Développement Web
R2.14 | Système d'information
R2.15 | Hébergement

1. Organisation de la SAé

QUI ?

Cette SAé est individuelle. Les livrables demandés le sont pour chacun d'entre vous.

QUAND ?

Les dates à bien conserver en tête pour cette SAé sont les suivantes :
De mercredi 22 mars au mardi 4 avril :
Réalisation successive des 7 itérations décrites dans ce document.
Le mercredi 5 avril :
Mise en ligne de votre site sur son hébergement "de production".
Du mercredi 5 avril au vendredi 7 avril :
Réalisation d'une itération surprise ! Elle vous sera communiquée le mercredi 5 avril.

QUOI ?

Il s'agit de développer une application de type Netflix, Prime vidéo, etc… Votre application sera subdivisée en 2 sous-applications complémentaires :

- un site vitrine" destiné à tous les utilisateurs
- un back office destiné aux administrateurs

Ces 2 parties feront l'objet d'un développement Front. L'une comme l'autre fonctionneront grâce à une même application serveur PHP/MySQL qui sera elle l'objet d'un développement Back.

Notez que cette structuration est identique aux TP de R2.13 réalisés sur le thème des "menus de la semaine".

COMMENT ?

Vous procéderez par itération. Une itération est une version incomplète mais fonctionnelle du travail demandé. C'est-à-dire qu'on ne va pas développer toutes les fonctionnalités de l'application d'un seul coup. Chaque itération se concentre sur une ou deux fonctionnalités seulement. Mais on ne passe pas à l'itération suivante tant que la précédente n'est pas aboutie à 100%. D'itération en itération, vous enrichirez votre application de fonctionnalités supplémentaires.

Donc, il ne s'agit pas de faire d'abord la base de données, puis les scripts PHP, puis les sites vitrine et admin, il vous faudra à chaque itération intervenir sur toutes les parties qui le

nécessitent. Toutes les itérations vous sont détaillées en fin de document. Voici déjà un résumé :

- **Itération 1**
  Voir les films disponibles dans la base de données via un site vitrine.
  Ajouter des films dans la base de données via une interface d'administration
- **Itération 2**
  Regarder le trailer d'un film sur le site vitrine.
- **Itération 3**
  Pouvoir filtrer l'affichage des films selon leur catégorie.
- **Itération 4**
  Avoir des profils utilisateur.
- **Itération 5**
  Avoir une liste de lecture par profil utilisateur
- **Itération 6**
  Pouvoir retirer un film de la liste de lecture d'un profil utilisateur
- **Itération 7**
  Pouvoir supprimer un profil utilisateur depuis le back office.

EVALUATION

Vos itérations (hébergées) seront appréciées comme le ferait un véritable client :

- **Convaincant et votre base d'évaluation sera 15.**
  C'est convaincant si le résultat est de qualité professionnelle en tout point. Si des défauts existent, ils sont mineurs et aisément rectifiables.

- **Mitigé et votre base d'évaluation sera 10.**
  C'est mitigé si le résultat est intéressant mais avec des défauts majeurs qui ne sont pas acceptables dans une optique professionnelle. Un défaut majeur reste rectifiable mais demandera un travail significatif.

- **Insuffisant et votre base d'évaluation sera 5.**
  C'est insuffisant si le résultat n'est tout simplement pas utilisable et/ou ne répond pas à la demande. Les défauts sont alors critiques, trop de mauvais choix ou d'erreurs ont été faites pour envisager rectifier le tir sans reprendre le projet de zéro.

Cette base d'évaluation sera ensuite modulée en appréciant séparément le niveau d'acquisition de tous les apprentissages critiques (voir au début du document) impliqués dans cette SAE.

2. Descriptif détaillé des itérations

Itération 1

Voir les films disponibles dans la base de données via un site vitrine.
Ajouter des films dans la base de données via une interface d'administration

FRONT (HTML / CSS / Javascript)

Intégration (site vitrine & site admin) :
A partir du projet Figma fourni, sélectionner et intégrer les éléments dont vous avez besoin pour réaliser les interfaces demandées. Procédez ainsi pour chaque itération, chaque fois que vous avez besoin d'éléments supplémentaires. Des informations complémentaires vous sont données dans la dernière partie de ce document.

Site vitrine :
Une page qui possède un menu et une zone d'affichage.
Dans le menu se trouve un bouton (ou équivalent) "All". Lorsque l'utilisateur clique sur "All", tous les films présents dans la base de données apparaissent dans la zone d'affichage.
*Indications : cliquer sur le bouton "All" déclenche une requête HTTP de type fetch en Javascript ciblant l'url **script.php?action=getmovies** sur le serveur. La réponse du serveur à la requête contient au format JSON tous les films disponibles dans la base de données. A l'aide d'un template, tous les films sont formatés et ajoutés dans la zone d'affichage.*

Site administrateur (back office) :
Une page avec un formulaire qui permet d'ajouter un film dans la base de données. L'envoi des données de formulaire sera laissé à la charge du navigateur. On utilisera donc un bouton de type "submit" pour le déclencher. On prendra soin de nommer ce bouton "action" et de lui donner la valeur "addmovie" en cohérence avec la description de la partie "Scripts serveur" ci-après. Le formulaire sera configuré pour envoyer les données à script.php sur le serveur. Gardez en tête que c'est le navigateur qui fera la requête HTTP et ajoutera automatiquement les données du formulaire en paramètre de la requête.

BACK (PHP / MySQL)

Scripts serveur :
Notes : On suppose que le script PHP chargé de répondre aux requêtes HTTP des applications Front se nomme script.php. Les noms des paramètres dans les requêtes HTTP sont donnés à titre d'exemple. Vous pouvez choisir les mêmes ou d'autres, ça ne change rien à la façon dont doit répondre le serveur.
La partie serveur doit savoir répondre aux requêtes HTTP suivantes :

`script.php?action=getmovies`
Si la requête HTTP est accompagnée du paramètre "action" et que sa valeur est "getmovies" alors le script serveur retournera dans sa réponse HTTP tous les films présents dans la base de données. Les données seront transmises au format JSON.

`script.php?action=addmovie&title=Interstellar&direction=christopher_nolan&...`
Si la requête HTTP est accompagnée du paramètre “action” et que sa valeur est “addmovie” alors le script serveur ajoutera à la base de données un nouveau film dont les caractéristiques (titre, réalisateur, etc...) seront les valeurs des autres paramètres de la requête HTTP. Voir la section base de données ci-après pour la liste complète des caractéristiques à prendre en compte.

Base de données :
Elle se résume à une table Movies pour stocker tous les films. Dans cette table, un film sera décrit par :

- un identifiant unique
- son titre
- son réalisateur
- l’année de sa sortie
- l’url d’une image de l’affiche du film
- l’url embed youtube du trailer du film

Pour débuter vous saisirez (possiblement via le back office) 4 à 6 films dans la table Movies.

Itération 2

Regarder le trailer d’un film sur le site vitrine.

FRONT (HTML / CSS / Javascript)

Intégration (site vitrine & site admin) :
A partir du projet Figma fourni, sélectionner et intégrer les éléments dont vous avez besoin pour réaliser les interfaces demandées. Procédez ainsi pour chaque itération, chaque fois que vous avez besoin d’éléments supplémentaires.

Site vitrine :
Lorsque l’on clique sur un des films présents dans la zone d’affichage, celle-ci est mise à jour pour faire apparaître le trailer du film sur lequel on a cliqué. Pour faire à nouveau apparaître tous les films dans la zone d’affichage, on cliquera sur “All” comme décrit lors de la précédente itération.
Indication : Plusieurs solutions sont possibles. La plus simple : Lorsque l’on clique sur l’élément HTML qui contient l’affichage d’un film, cela déclenche une requête de type fetch en Javascript ciblant l’url ***script.php?action=getmovie&idmovie=12*** *qui demande au serveur de renvoyer toutes les données (en JSON) du film sur lequel on a cliqué (dans notre exemple, celui d’identifiant 12). Lorsqu’on les reçoit, on utilise un (autre) template pour formater le player youtube à l’aide des données reçues. Et on insère le tout dans la zone d’affichage à la place de la liste de tous les films. Cette solution nécessitera d’adapter le formatage réalisé dans la précédente itération afin que chaque clique déclenche la bonne requête avec le bon identifiant de film.*

Site administrateur (back office) :
Pas d’évolution du back office.

BACK (PHP / MySQL)

Scripts serveur :
Notes : On suppose que le script PHP chargé de répondre aux requêtes HTTP des applications Front se nomme script.php. Les noms des paramètres dans les requêtes HTTP sont donnés à titre d'exemple. Vous pouvez choisir les mêmes ou d'autres, ça ne change rien à la façon dont doit répondre le serveur.
La partie serveur doit savoir répondre aux requêtes HTTP supplémentaires suivantes :

`script.php?action=getmovie&idmovie=45`
Si la requête HTTP est accompagnée du paramètre "action" et que sa valeur est "getmovie" (notez bien getmovie et non getmovie**S** comme dans la précédente itération) alors le script serveur retournera dans sa réponse HTTP les données du film dont l'identifiant est égal à la valeur du paramètre idmovie (dans notre exemple idmovie vaut 45 mais chaque film possède un identifiant différent bien sûr). Les données seront transmises au format JSON.

Base de données :
La base de données ne nécessite aucune modification.

Itération 3

Pouvoir filtrer l'affichage des films selon leur catégorie.

FRONT (HTML / CSS / Javascript)

Intégration (site vitrine & site admin) :
A partir du projet Figma fourni, sélectionner et intégrer les éléments dont vous avez besoin pour réaliser les interfaces demandées. Procédez ainsi pour chaque itération, chaque fois que vous avez besoin d'éléments supplémentaires.

Site vitrine :
Le menu permet désormais de choisir si l'on veut afficher tous les films ou bien uniquement les films d'une catégorie donnée (par exemple : Fantasy, Comedy, Sci-Fi, Animation, etc...) Vous pouvez par exemple ajouter des boutons ou bien utiliser un élément <select> pour que votre menu ne soit pas dépendant du nombre de catégories. Cette solution vous est recommandée. L'option "All" fait partie des options de votre <select> si l'utilisateur ne souhaite pas filtrer. Et c'est l'option par défaut.
*Indication : le passage d'un <button> sur lequel on clique à un <select> dont on modifie la valeur fait que l'on ne **clique** plus sur un bouton mais que l'on **change** la valeur d'une liste à choix multiples. Par conséquent, on ne va plus réagir à un **onclick**, mais à un **onchange**.*
*Ensuite, si l'on sélectionne tous les films, on est dans le cas de l'itération 1 et l'on déclenche toujours une requête HTTP de type fetch en Javascript qui ciblera l'url **script.php?action=getmovies**. Si une category a été choisie par l'utilisateur, alors l'url cible devient **script.php?action=getmovies&idcategory=3** pour demander au serveur ne retourner que les films de la catégorie d'identifiant 3 (3 est une valeur exemple, correspondant à l'identifiant d'une catégorie de film). Pour former votre requête il vous faudra récupérer la valeur de l'option sélectionnée. Vous ferez en sorte*

votre <select> affiche les noms des catégories mais que la valeur de chaque option soit l'identifiant cette catégorie dans la base de donnée.
Notez que cette itération est analogue au TP sur "Les menus de la semaine" lorsque vous avez dû ajouter la saisie d'un numéro de semaine.

*Indication++ : on acceptera que le <select> ainsi que ses options soient écrits "en dur" directement en HTML. Mais on appréciera que les options du <select> soient formatées dynamiquement en utilisant les catégories présentes dans la base de données. Ce qui implique de faire une requête HTTP (par exemple **script.php?action=getcategories**) au chargement du site pour récupérer les catégories de la bdd au format JSON et formater les options à l'aide d'un template, options que l'on placera ensuite dans le <select>. Considérez cette solution comme un bonus.*

Site administrateur (back office) :
Le formulaire pour ajouter un film à la base de données sera complété afin de permettre à l'administrateur de renseigner la catégorie d'un film au moment de son ajout.
Note : pour les films déjà présents dans votre base avant cette itération, vous ajouterez "à la main" leur catégorie via phpMyAdmin.

BACK (PHP / MySQL)

Scripts serveur :

Notes : On suppose que le script PHP chargé de répondre aux requêtes HTTP des applications Front se nomme script.php. Les noms des paramètres dans les requêtes HTTP sont donnés à titre d'exemple. Vous pouvez choisir les mêmes ou d'autres, ça ne change rien à la façon dont doit répondre le serveur.
La partie serveur doit savoir répondre aux requêtes HTTP supplémentaires suivantes :

`script.php?action=getmovies&idcategory=45`
Il s'agit d'une adaptation du traitement de la première requête réalisé à la première itération. Soit la requête ne comporte pas de paramètre idcategory et dans ce cas, la réponse demeure l'ensemble des films de la base de données au format JSON. Par contre, si un paramètre idcategory est présent, alors on sélectionne uniquement les films qui sont de cette catégorie. Les données sont toujours transmises au format JSON.

`script.php?action=addmovie&idcategory=7&title=Interstellar&direction=christopher_nolan&...`
Il s'agit aussi d'une adaptation du traitement de la seconde requête réalisé à la première itération. Quand l'administrateur enregistre un nouveau film depuis le back office, le navigateur va transmettre, en plus des informations déjà traitées, un identifiant de catégorie. Vous devez en tenir compte et faire en sorte que ce dernier soit correctement enregistré dans la base de données.

Base de données :
La base de données doit être modifiée pour que chaque film soit associé à une (et une seule) catégorie. Via phpMyAdmin, vous ajouterez une table Category. Chaque catégorie sera caractérisée par :

- un identifiant unique
- un nom de catégorie

Toujours via phpMyAdmin, vous ajouterez quelques catégories dans cette table, à minima celles qui correspondent aux films déjà enregistrés dans la table Movie.Vous ajouterez ensuite une colonne id_category à la table Movie. Cette colonne fait référence à un identifiant de catégorie présent dans la table Category.

Itération 4

Avoir des profils utilisateur.

Un profil utilisateur n'est pas exactement un compte utilisateur car on ne demandera pas d'authentification par login et mot de passe. Mais l'idée est bien de permettre ensuite d'associer des informations à un profil afin de permettre un affichage personnalisé au niveau du site vitrine.

FRONT (HTML / CSS / Javascript)

Intégration (site vitrine & site admin) :
A partir du projet Figma fourni, sélectionner et intégrer les éléments dont vous avez besoin pour réaliser les interfaces demandées. Procédez ainsi pour chaque itération, chaque fois que vous avez besoin d'éléments supplémentaires.

Site vitrine :
Un (autre) élément <select> permet de choisir un des profils utilisateur présents dans la base de données. Le profil sélectionné sera le profil "actif" (voir itération 5).
Indication : on acceptera que ce <select> soit directement codé en dur en HTML. Mais on appréciera qu'il soit formaté dynamiquement après avoir été récupérer la liste des profils disponibles dans la base de données.

Site administrateur (back office) :
Un second formulaire permet à l'administrateur d'ajouter un profil utilisateur à la base de données. Vous pouvez ajouter ce second formulaire sur la même page ou si vous préférez, sur une seconde. Dans ce dernier cas, vous prévoyez un petit menu de navigation basique (de simples liens <a>) pour aller d'un formulaire à un autre.

BACK (PHP / MySQL)

Scripts serveur :

Notes : On suppose que le script PHP chargé de répondre aux requêtes HTTP des applications Front se nomme script.php. Les noms des paramètres dans les requêtes HTTP sont donnés à titre d'exemple. Vous pouvez choisir les mêmes ou d'autres, ça ne change rien à la façon dont doit répondre le serveur.
La partie serveur doit savoir répondre aux requêtes HTTP suivantes :

`script.php?action=getprofiles`
Si la requête HTTP est accompagnée d'un paramètre "action" dont la valeur est "getprofiles", alors le script serveur sélectionne tous les profils disponibles dans la base de données et les inclut au format JSON dans sa réponse HTTP.
Indication : Le traitement de cette requête est nécessaire seulement si vous avez choisi de générer dynamiquement la liste des profils sur le site vitrine. Sinon ce n'est pas utile.

`script.php?action=addprofile&name=bob`
Si la requête HTTP est accompagnée d'un paramètre "action" dont la valeur est "addprofile", alors le script serveur ajoutera un nouveau profile dans la base de donnée dont le nom sera la valeur du paramètre "name" (dans notre exemple, la valeur de "name" est "bob"). Le script serveur inclura dans la réponse HTTP en message simple pour dire si l'opération s'est bien passée (ou pas).

Base de données :
La base de données doit être modifiée pour stocker des profils d'utilisateur. Via phpMyAdmin, vous ajouterez une table UserProfile dans ce but. Un profil utilisateur sera caractérisé par :

- un identifiant unique
- un nom de profil

Au besoin vous pourrez ajouter quelques profils directement via phpMyAdmin pour tester. Mais le mieux serait de le faire via votre site d'administration.

Itération 5

Avoir une liste de lecture par profil utilisateur

Une liste de lecture c'est un peu comme un panier sur un site marchand. On peut mettre des films dedans, pas pour les acheter mais pour les regarder plus tard. La liste de lecture d'un profil utilisateur est sauvegardée dans la base de données. Ce qui permet de la restaurer lorsqu'on revient sur le site et que l'on sélectionne son profil. L'utilisateur peut voir le contenu de sa liste de lecture et, s'il clique sur l'un des films présents, voir son trailer.

Cette itération est volontairement moins détaillée. Les itérations précédentes vous ont déjà donné toutes les clés. Et vous aurez certainement commencé à comprendre qu'ajouter une fonctionnalité c'est peu ou prou tout le temps la même chose, peu importe la fonctionnalité.

FRONT (HTML / CSS / Javascript)

Intégration (site vitrine & site admin) :
A partir du projet Figma fourni, sélectionner et intégrer les éléments dont vous avez besoin pour réaliser les interfaces demandées. Procédez ainsi pour chaque itération, chaque fois que vous avez besoin d'éléments supplémentaires.

Site vitrine :
Première modification : Lorsque l'interface présente des films, faites apparaître pour chacun d'eux un symbole "+" (ou autre). Cliquer sur "+" ajoute un film à la playlist de l'utilisateur dont le profil est actif (sélectionné).
Indications : Lorsque l'utilisateur clique sur "+", cela déclenche une requête de type fetch en Javascript ciblant l'url ***script.php?action=addtoplaylist?idmovie=56&idprofile=2*** *pour demander à ce que le film*

d'identifiant 56 (c'est un exemple) soit ajouté à la playlist du profil d'identifiant 2 (c'est encore un exemple).
Faites en sorte que votre "+" ne se superpose pas à l'élément qui contient les informations du film. Sinon un clic sur un "+" serait aussi un clic sur le film et déclencherait son trailer. Ce type de problème se gère, mais pas à niveau BUT1. Donc contourner le en plaçant votre "+" juste à côté.
Enfin notez que l'ajout d'un film à une playlist n'oblige pas à traiter la réponse du serveur.

Seconde modification : Votre interface doit présenter un menu supplémentaire que l'on appellera "Playlist". Lorsque l'on clique dessus, les films que l'on a déjà ajoutés à sa playlist apparaissent dans la zone d'affichage. La présentation peut être la même que pour afficher n'importe quel film sauf qu'il n'y a plus de symbole "+" (on ne va pas ajouter une playlist des films qui s'y trouvent déjà...)
Indications : Lorsque l'utilisateur clique sur "-", cela déclenche une requête de type fetch en Javascript ciblant l'url ***script.php?action=getplaylist?idprofile=2*** *pour demander au serveur de retourner tous les films (en JSON) de la playlist du profil utilisateur d'identifiant 2 (c'est toujours un exemple).*

Site administrateur (back office) :
Le back office ne nécessite pas d'évolution.

BACK (PHP / MySQL)

Scripts serveur :
Notes : On suppose que le script PHP chargé de répondre aux requêtes HTTP des applications Front se nomme script.php. Les noms des paramètres dans les requêtes HTTP sont donnés à titre d'exemple. Vous pouvez choisir les mêmes ou d'autres, ça ne change rien à la façon dont doit répondre le serveur.
La partie serveur doit savoir répondre aux requêtes HTTP suivantes :

`script.php?action=addtoplaylist&idmovie=23&idprofile=7`
Si la requête HTTP est accompagnée du paramètre "action" et que sa valeur est "addtoplaylist" alors le script enregistre dans la base de données la présence du film d'identifiant 23 (c'est un exemple) dans la playlist de l'utilisateur d'identifiant 7 (c'est aussi un exemple).

`script.php?action=getplaylist&idprofile=9`
Si la requête HTTP est accompagnée du paramètre "action" et que sa valeur est "getplaylist" alors le script sélectionne dans la base de données tous les films contenu dans la playlist du profil d'identifiant 9 (c'est un exemple). Comme à chaque fois que l'on renvoie des films, les données seront transmises au format JSON dans la réponse HTTP.

Base de données :
Vous devez modifier votre base de données afin de pouvoir enregistrer les données d'une playlist pour chaque profil utilisateur. Vous êtes dans le cas "classique" d'une association entre la table UserProfile et la table Movie. Cette association peut être nommée Playlist. Elle associe un identifiant de profil à un identifiant de film. Playlist est une association "Many-To-Many" au sens où un profil peut être associé à plusieurs films (c'est mieux si on veut plus d'un film dans sa playlist) et un film peut être associé à plusieurs profils (c'est mieux si on veut qu'un film puisse se trouver plusieurs playlist.

Pour matérialiser cette association, créez une table Playlist avec une colonne id_profile qui référence un

identifiant de profil et une colonne id_movie qui référence un identifiant de film. N'oubliez pas de définir sa clé primaire.

Itération 6

Pouvoir retirer un film de la liste de lecture d'un *profil utilisateur*

FRONT (HTML / CSS / Javascript)

Intégration (site vitrine & site admin) :
A partir du projet Figma fourni, sélectionner et intégrer les éléments dont vous avez besoin pour réaliser les interfaces demandées. Procédez ainsi pour chaque itération, chaque fois que vous avez besoin d'éléments supplémentaires.

Site vitrine :
Lorsqu'on consulte les films d'une playlist, il n'y a pas de symbole "+" mais à la place un symbole "-". Et si l'on clique sur "-" cela retire le film de la playlist.
Indications : Lorsque l'utilisateur clique sur "-", cela déclenche une requête de type fetch en Javascript ciblant l'url ***script.php?action=removefromplaylist?idmovie=56&idprofile=2*** *pour demander à ce que le film d'identifiant 56 (c'est un exemple) soit supprimé de la playlist du profil d'identifiant 2 (c'est toujours un exemple). Notez qui si vous voulez que l'affichage de la playlist soit rafraichi (pour faire disparaître le film que l'on a retiré) il faudra refaire une requête HTTP pour recharger la playlist à jour depuis le serveur et la ré-afficher.*

Site administrateur (back office) :
Le back office ne nécessite pas d'évolution.

BACK (PHP / MySQL)

Scripts serveur :
Notes : On suppose que le script PHP chargé de répondre aux requêtes HTTP des applications Front se nomme script.php. Les noms des paramètres dans les requêtes HTTP sont donnés à titre d'exemple. Vous pouvez choisir les mêmes ou d'autres, ça ne change rien à la façon dont doit répondre le serveur.
La partie serveur doit savoir répondre aux requêtes HTTP suivantes :

`script.php?action=removefromplaylist&idmovie=13&idprofile=4`
Si la requête HTTP est accompagnée du paramètre "action" et que sa valeur est "removefromplaylist" alors le script supprime de la base de données l'association du film d'identifiant 13 (c'est une exemple) à la playlist du profile d'identifiant 7 (par exemple)

Base de données :
La base de données ne nécessite pas de modification.

Itération 7

Pouvoir supprimer un profil utilisateur depuis le back office.

FRONT (HTML / CSS / Javascript)

Intégration (site vitrine & site admin) :
A partir du projet Figma fourni, sélectionner et intégrer les éléments dont vous avez besoin pour réaliser les interfaces demandées. Procédez ainsi pour chaque itération, chaque fois que vous avez besoin d'éléments supplémentaires.

Site vitrine :
Le site vitrine ne nécessite pas de modification.

Site administrateur (back office) :
Le back office présente un formulaire supplémentaire. Celui-ci inclut une liste déroulante (un <select>) qui permet de sélectionner par son nom l'un des profils utilisateur présent dans la base de données. Un bouton de type "submit" permet d'envoyer au serveur la demande de suppression du profil.

BACK (PHP / MySQL)

Scripts serveur :
Notes : On suppose que le script PHP chargé de répondre aux requêtes HTTP des applications Front se nomme script.php. Les noms des paramètres dans les requêtes HTTP sont donnés à titre d'exemple. Vous pouvez choisir les mêmes ou d'autres, ça ne change rien à la façon dont doit répondre le serveur.
La partie serveur doit savoir répondre aux requêtes HTTP suivantes :

`script.php?action=deleteprofile&idprofile=11`
Si la requête HTTP est accompagnée du paramètre "action" et que sa valeur est "deleteprofile" alors le script supprime de la base de données le profile d'identifiant 11 (c'est un exemple) de la table UserProfile ainsi que le contenu de sa playlist.

Base de données :
La base de données ne nécessite pas de modification.

---

3. Partie Intégration : Faire son Framework CSS

Tout au long de la SAé, vous allez construire votre propre framework CSS à partir du projet Figma SAé203. -

Les frameworks CSS sont des ensembles d'outils et de conventions qui facilitent la conception et le développement d'un site web en fournissant un ensemble de règles CSS prédéfinies et réutilisables. Les développeurs peuvent alors combiner ces classes prédéfinies pour créer la mise en forme de leur site web, ce qui leur permet de gagner du temps et d'améliorer leur productivité.

Même si cela n'est pas toujours conseillé, nous allons diviser notre CSS en plusieurs fichiers distincts selon leur fonction : nous allons créer un fichier pour les variables, un fichier pour les classes utilitaires, un fichier pour les classes de composants et un fichier pour les classes de layout. Nous verrons l'an prochain comment utiliser un préprocesseur CSS pour compiler tous ces fichiers en un seul.
Voici quelques pistes pour créer votre framework CSS :
- reset
- variables
- **classes utilitaires (Atomic CSS)** : couleurs, espacements, taille de police,
  Les classes utilitaires sont utilisées pour ajouter des styles spécifiques à un élément sans avoir à créer de nouvelles classes CSS. Les classes utilitaires (ou atomiques) sont des classes destinées à contenir une seule déclaration CSS.
  À chaque classe correspond une seule fonction, ce qui offre une complète séparation entre la structure HTML des données de présentation CSS.
- **classes de layout flexbox, grid (Atomic CSS)**
  Les classes de layout sont elles aussi des classes utilitaires, mais utilisées pour définir la structure de base d'une page web, comme la mise en page de la grille, la position des éléments et la taille de la page.
- **classes des composants (BEM)** : boutons, formulaires, menus, etc.
  Les classes de composants sont utilisées pour définir le style des éléments de base d'une page web, comme les boutons, les formulaires, les menus, etc. (c'est cette méthode que nous avons principalement utilisée jusqu'à présent).

Documentez chaque classe et fonctionnalité de votre framework pour faciliter la compréhension et la maintenance du code.
Concentrez-vous sur la simplicité et la réutilisabilité de vos classes.
https://www.alsacreations.com/actu/lire/1823-quels-framework-methodologie-css-choisir.html

Votre framework évoluera tout au long de la SAé, en fonction de vos besoins ou des contraintes qui vous seront données.