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 :

Du mercredi 22 avril au jeudi 7 mai :

Temps imparti pour réaliser la SAÉ. Si vous n'êtes pas en cours, vous êtes en SAÉ. Le temps programmé et réservé pour se consacrer à la SAÉ est de **95h (la moitié en séances encadrées + l'autre moitié en autonomie).** C'est énorme. Et il faut que ça se voit dans votre travail à la fin.

Le vendredi 30 avril :

Première mise en ligne de votre projet (l'itération de plus haut rang que vous aurez terminée à cette date).

Du mardi 5 mai au jeudi 7 mai :

Réalisation d'une itération surprise ! Elle vous sera communiquée le mardi 5 mai midi.

Le jeudi 7 mai :

Mise en ligne de votre site sur son hébergement "de production".

QUOI ?

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

- APP : une application destinée aux utilisateurs de la plateforme
- ADMIN : un back office destiné aux administrateurs de la plateforme

Pour réaliser ce projet, on respectera **scrupuleusement** la même structure que celle travaillée en R2.13 avec les "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.

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.

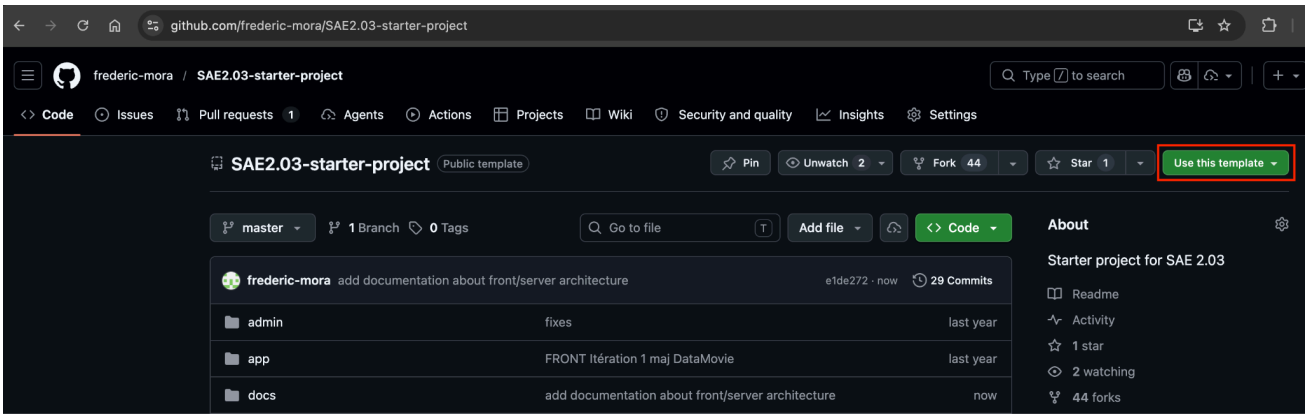
Cette base d'évaluation sera ensuite modulée en appréciant séparément le niveau d'acquisition de tous les apprentissages critiques (voir au début du document) impliqués dans cette SAE.

2. Mise en place de l'environnement de développement

Vous allez utiliser le serveur mmi.unilim.fr pour votre application. Afin d'éviter de devoir sans cesse transférer vos fichiers de votre machine vers le serveur mmi.unilim.fr (ce que vous faisiez en TP), il est possible d'installer et d'utiliser l'extension Remote SSH pour VSCode. Elle vous permettra de connecter VSCode au serveur mmi.unilim.fr de sorte à éditer vos fichiers directement sur le serveur.

Pour mettre en place l'environnement de développement, vous devez au préalable créer un compte Github.

1. Créez votre projet à l'aide du *template project* disponible ici :
   https://github.com/frederic-mora/SAE2.03-starter-project



   Au passage renommez le repository SAE2.03-*votreNom*

   Note : un template project est une base de code pour un projet. Cela vous permet de débuter sans pour autant partir de zéro. Le template project fourni est identique aux architectures travaillées en R2.13. Vous trouverez dans le dossier docs les explications sur ces architectures front et server, déjà présentes dans les derniers TP de R2.13..

2. Ouvrez Visual Studio Code et via l'extension Remote SSH, connectez-vous à mmi.unilim.fr :
   a. Cliquez sur l'icône de l'extension Remonte SSH dans la barre latérale gauche
   b. Au niveau du menu SSH, cliquez sur +
   c. Faites une première connnexion à mmi.unilim.fr en saisissant la commande :
      ssh votreLoginUnilim@mmi.unilim.fr
   d. Sélectionnez le fichier ssh/config pour que Remote SSH mémorise le serveur mmi.unilim.fr
   e. Après saisie de votre mot de passe Unilim vous devez voir en bas à gauche de la fenêtre que vous êtes connecté
   f. Remote ssh a mémorisé le serveur mmi.unilim.fr, vous le retrouverez sous le menu SSH pour y accéder plus rapidement les prochaines fois

3. Ouvrez votre répertoire public_html

4. Ouvrez un terminal et clonez votre fork :
   git clone https://github.com/votre-username-github/SAE2.03-votreNom

5. Suivez ensuite les indications contenues dans le fichier README.md pour finir la configuration du projet.

Il est **important de sauvegarder votre travail régulièrement** (minimum une fois par jour) en synchronisant vos modifications avec votre repository Github (SAE2.03-votreNom).
Vous allez faire des modifications de votre code sur mmi.unilim.fr. Et vous allez vouloir que ces modifications soient sauvegardées dans votre repository SAE2.03-votreNom sur votre espace Github. Par conséquent, il faut que mmi.unilim.fr puisse se connecter à votre compte Github, ce qui nécessite une authentification.
Il existe plusieurs méthodes pour s'authentifier, nous vous en proposons 2 en vous conseillant la première pour commencer.

Solution 1 : Utilisez l'extension Github Pull Request :
Recherchez et installez cette extension. En soit, vous ne vous en servirez pas. Mais lors de son installation, vous allez vous connecter à votre compte Github et bénéficierez de cette connexion pour pouvoir synchroniser vos modifications et votre repository.

Solution 2 : Créez des clés SSH
A priori vous connaissez déjà le principe (https). L'authentification par clé SSH est une méthode sécurisée pour se connecter à un serveur ou à un service, comme GitHub, sans utiliser de mot de passe. Elle repose sur un couple de clés : une clé privée, qui reste sur votre machine (mmi.unilim.fr) et doit être gardée secrète, et une clé publique, qui est partagée avec l'autre machine (Github). Lorsque vous tentez de vous connecter, Github utilisera la clé publique pour vérifier que vous possédez la clé privée correspondante. Si elles correspondent, l'accès est accordé. Cette méthode est plus sécurisée que les mots de passe, car elle utilise un chiffrement robuste et élimine le risque de vol de mot de passe.

A. Générez une nouvelle clé SSH

Depuis un terminal ouvert sur mmi.unilim.fr, générez une nouvelle clé avec la commande suivante :

```
ssh-keygen -t ed25519 -C "votre_email@example.com"
```

- Remplacez votre_email@example.com par l'adresse e-mail associée à votre compte GitHub.
- Appuyez sur Entrée pour accepter l'emplacement par défaut (~/.ssh/id_ed25519).
- Si vous le souhaitez, ajoutez une passphrase pour plus de sécurité (optionnel)
- Note : ed25519 est l'algorithme utilisé pour crypter les clés.

B. Ajoutez la clé SSH à l'agent SSH

Démarrez l'agent SSH et ajoutez votre clé privée :

```
eval "$(ssh-agent -s)"

ssh-add ~/.ssh/id_ed25519
```

C. Ajoutez la clé publique à votre compte GitHub

- Copiez le contenu de votre clé publique (~/.ssh/id_ed25519.pub) dans le presse-papiers :
- Connectez-vous à votre compte GitHub.
- Allez dans Settings > SSH and GPG keys > New SSH key.

- Collez la clé publique dans le champ "Key" et donnez-lui un titre (par exemple, "mmi.unilim.fr").
- Cliquez sur Add SSH key.

L'avantage de cette seconde solution est qu'elle est plus fiable et évite de devoir saisir son mot de passe à chaque nouvelle session de travail. Notez que l'opération est à renouveler sur chaque machine que vous souhaitez utiliser pour travailler.

Comment faire un "commit" ?
Vous êtes à présent prêt pour pouvoir synchroniser votre travail et votre repository sur Github. Voici la démarche à suivre depuis un terminal :

1. Assurez vous de vous trouver dans le répertoire de votre projet (SAE2.03-votreNom)

2. Exécutez la commande : *git add .*
   Dis à git qu'on ajoute le répertoire courant (.) à la liste des modifications à prendre en compte

3. Exécutez la commande : *git commit -m "message sur les modifications apportées"*
   Dans le jargon git, un commit est une sauvegarde à l'instant t du code. Git conserve un historique de tous vos commits et il est possible de revenir à la version de code d'un commit si besoin était : Git est un outil de versionning. C'est pour cela qu'il est important de préciser avec un message les modifications car ce sera la seule information vous permettant de retrouver une version donnée de votre projet.

4. Exécutez la commande : *git push*
   Cette commande synchronise votre dossier local (sur mmi.unilim.fr) avec votre repository Github (sur votre compte Github.com). Cela permet de garder à jour une version de votre travail sur Github.com et vous protège d'erreurs de manipulation qui conduiraient à la perte totale, définitive et déprimante de votre projet. Git (outil local) et Github (plateforme en ligne) sont 2 choses distinctes mais sont prévues pour fonctionner en symbiose.

Note : il existe aussi des extensions VSCode pour gérer vos commits autrement qu'en ligne de commande. Ca n'apporte pas grand chose compte tenu de l'usage de base que vous aurez de Git. Après c'est comme vous voulez tant que les commits sont faits.

> Pour rappel, le serveur mmi.unilim.fr est également accessible depuis l'extérieur de l'Université si vous passez par son VPN.

3. Descriptif détaillé des itérations

Pour réaliser les itérations décrites ci-après, vous travaillerez avec Visual Studio Code et utiliserez le serveur web et de base de données dont vous disposez sur mmi.unilim.fr.

N'effacez pas ce que vous avez dans votre base de données, ce n'est pas grave si des tables liées à des applications différentes coexistent dans la même base tant que vous ne vous mélangez les

pinceaux ! Une pratique courante consiste à préfixer toutes les tables pour une même application. Par exemple vous pouvez préfixer vos tables avec SAE203_****.

ITÉRATION 1

[APP] Consulter la liste des films proposés par la plateforme

USER STORY

En tant que visiteur de l'application,
je veux pouvoir consulter la liste des films disponibles sur la plateforme,
afin de découvrir les films proposés et choisir ceux qui m'intéressent.

Critères d'acceptation :

1. La liste des films doit afficher au minimum :
    - Le titre du film.
    - Une affiche ou une image représentative.

2. Les films doivent être présentés sous forme de cartes ou de liste visuellement attrayante.
3. La liste doit être accessible depuis la page d'accueil de l'application.
4. Si aucun film n'est disponible, un message clair doit être affiché (ex. : "Aucun film disponible pour le moment.").

FRONT (HTML / CSS / Javascript) - Indications

app / data /
Prévoir un module *dataMovie.js* qui sera chargé de toutes les requêtes HTTP à destination de la partie serveur. Ici vous aurez besoin d'une fonction DataMovie.requestMovies (par exemple) qui interrogera le serveur afin d'obtenir la liste de tous les films disponibles.

app / component /
Prévoir un composant Movie pour afficher un film conformément aux critères d'acceptation.
Vous aurez plusieurs films à afficher, il serait bon que la fonction Movie.format accepte un tableau de films en paramètre afin de formater un à plusieurs films.

BACK (PHP / MySQL) - Indications

BDD MySQL :
Via phpMyAdmin sur mmi.unilim.fr, importez le fichier SAE203.sql fourni avec le starter project. Il

contient un noyau de base de données avec une dizaine de films pour vous aider à démarrer.

server/
Votre serveur doit permettre de retourner en JSON la liste des films disponibles en base. Comme d'habitude, on se basera sur la valeur d'un paramètre 'todo' pour interpréter la requête HTTP.
Par exemple : https://mmi.unilim.fr/.../script.php?todo=readmovies
Respectez bien le schéma habituel : *script.php / controller.php / model.php*

ATTENTION
Renvoyer les informations détaillées de tous les films de la base de données pourraient représenter beaucoup beaucoup (trop) d'informations. Aussi on ne renverra que le strict nécessaire : le titre, son affiche et l'identifiant. Vous n'allez pas utiliser l'identifiant de suite mais ça vous sera utile plus tard.

ITÉRATION 2

[ADMIN] Ajouter des films dans la base de données

USER STORY

En tant qu'administrateur,
je veux avoir un formulaire pour ajouter des films dans la base de données,
afin d'enrichir le catalogue de films disponibles sur la plateforme.

Critères d'acceptation :

1. Le formulaire doit permettre de saisir les informations suivantes :
    - Titre du film.
    - Réalisateur du film
    - Année de sortie
    - Durée en minutes
    - Description ou synopsis.
    - Catégorie (Drame, Aventure, Animation...).
    - nom du fichier de l'affiche ou image du film.
    - URL du trailer.
    - Restrictions d'âge.
2. Une fois le formulaire soumis :
    - Les données doivent être validées (ex. : champs obligatoires remplis, âge valide).
    - Le film doit être ajouté à la base de données.
    - Un message de confirmation doit être renvoyé par le serveur (ex. : "Le film a été ajouté avec succès.").
3. En cas d'erreur (ex. : champ manquant ou problème de connexion), un message d'erreur clair doit être affiché.

FRONT (HTML / CSS / Javascript) - Indications

admin / data /
Prévoir un module *dataMovie.js* qui sera chargé de toutes les requêtes HTTP à destination de la partie serveur. Même rôle que celui de app/data, Mais les requêtes à prendre en charge ne seront pas toujours les mêmes. Ici vous devez commencer par prévoir une fonction DataMovie.add (par exemple) à même d'envoyer les données du formulaire MovieForm (voir ci-après) au serveur. Pour rappel, s'agissant de l'envoie de données de formulaire, la requête HTTP est à faire en POST.

admin / component /
Prévoir un composant MovieForm permettant la saisie d'un film via un formulaire conformément aux critères d'acceptation. Notez bien qu'on ne vous demande pas d'uploader les images via le formulaire, seulement le nom des fichiers. On acceptera de placer "manuellement" les images dans le dossier "server / images".

admin / component / Log
Le composant Log (le même qu'en TP) est déjà intégré à l'interface. Faites en bon usage pour afficher les messages en provenance du serveur ou le statut des opérations en cours.

admin / index.html
De manière analogue aux exercices de TP, prévoyez ici une fonction C.handlerAddMovie (par exemple) qui devra s'exécuter lorsque l'administrateur cliquera sur le bouton de validation du formulaire.

BACK (PHP / MySQL) - Indications

server/
Votre serveur savoir traiter des requêtes HTTP contenant les données d'un film à enregistrer dans votre base de données. Comme d'habitude, on se basera sur la valeur d'un paramètre 'todo' pour interpréter la requête HTTP.
Par exemple : https://mmi.unilim.fr/.../script.php?todo=addMovie
Respectez bien le schéma habituel : *script.php / controller.php / model.php*
Le serveur renverra un message indiquant si le film a bien été enregistré ou non.

ITÉRATION 3

[APP] Consulter les informations détaillées d'un film ainsi que son trailer

USER STORY

En tant qu'utilisateur de l'application,
je veux pouvoir consulter les informations détaillées d'un film ainsi que visionner son trailer,
afin de décider si je souhaite le regarder.

Critères d'acceptation :

1. Lorsqu'un utilisateur clique sur un film dans la liste, une page ou une section dédiée doit apparaître avec les informations suivantes :
    - Le titre du film.
    - Une affiche ou une image représentative.
    - Une description ou un synopsis.
    - Le réalisateur
    - L'année de sortie
    - La catégorie du film.
    - Les restrictions d'âge.
    - Le trailer du film.
2. Le trailer doit être visionnable directement via un lecteur intégré (ex. : iframe YouTube).
3. On doit pouvoir revenir à la liste des films via la barre de navigation

FRONT (HTML / CSS / Javascript) - Indications

app / data /
Prévoir dans le module *dataMovie.js* une fonction DataMovie.requestMovieDetails (par exemple) qui sera chargée d'interroger le serveur pour obtenir tous les détails d'un film en transmettant son identifiant en paramètre.

app / component / MovieDetail
Prévoir un composant MovieDetail permettant de présenter les détails d'un film conformément aux critères d'acceptation.

app / index.html
Prévoir une fonction C.handlerDetail (par exemple) qui prendra en paramètre l'identifiant d'un film, obtiendra du serveur ses informations détaillées puis les affichera dans l'application à la place de la liste des films

app / component / Movie
Un composant Movie doit être cliquable à présent. Cliquer sur film doit provoquer l'affichage détaillé du film. Pour ce vous devez faire en sorte de formater chaque composant Movie avec, sur un élément HTML donné, onclick="C.handlerDetail(X)" où X est l'identifiant du film formaté (donc ce sera différent pour chaque film).

app / component / NavBar
"Modifier le composant NavBar à votre convenance pour que l'on puisse en cliquant sur un menu ré-afficher la liste des films. Vous aurez certainement aussi à éditer ou ajouter un handler correspondant dans index.html

BACK (PHP / MySQL) - Indications

server/
Le serveur doit prendre en charge un nouveau type de requête HTTP pour renvoyer les informations détaillées d'un film. Par exemple :
https://mmi.unilim.fr/.../script.php?todo=readMovieDetail&id=23
Ce pourrait être la requête HTTP demandant les détails du film d'identifiant 23.
Respectez bien le schéma habituel : *script.php / controller.php / model.php*

ITÉRATION 4

[APP] Afficher les films en les regroupant par catégorie

USER STORY

En tant qu'utilisateur de l'application,
je veux voir les films regroupés par catégorie,
afin de naviguer facilement et trouver des films correspondant à mes préférences.

Critères d'acceptation :

1. Les catégories doivent être clairement affichées (ex. : Action, Comédie, Drame, etc.).
2. Chaque catégorie doit afficher une liste de films correspondants sous forme de cartes ou de vignettes.
3. Les films doivent inclure au minimum :
    - Le titre.
    - Une affiche ou une image représentative.
4. L'utilisateur doit pouvoir naviguer facilement entre les différentes catégories.

FRONT (HTML / CSS / Javascript) - Indications

app / component / MovieCategory
Créez un composant MovieCategory pour afficher le nom d'une catégorie ainsi qu'une liste de films appartenant à cette catégorie. En toute logique, vous importerez dans MovieCategory le composant Movie pour afficher cette liste.

app / index.html
Au lieu d'afficher une liste unique de films, on affiche désormais les films par catégorie. Donc on utilisera le composant MovieCategory autant de fois que de catégorie disponible.

BACK (PHP / MySQL) - Indications

server/
Il faut apporter les modifications nécessaires pour que la réponse à une requête telle que :
https://mmi.unilim.fr/.../script.php?todo=readMovies
comprend la catégorie de chaque film. Éventuellement (voir ci-après) les films sont déjà groupés par catégorie.

ATTENTION
A un moment ou à un autre, vous allez devoir regrouper les films par catégorie pour pouvoir les afficher comme demandé. Ca ne peut plus marcher juste avec un ensemble de films "en vrac", il faut faire "des paquets de films" par catégorie. Techniquement, cela peut se faire côté Front juste après réception des films ; mais aussi côté serveur web, au niveau du contrôleur ou même du modèle ; et aussi sur le serveur MySQL si vous faites la ou les bonnes requêtes ! Les 3 solutions pourront fonctionner. Si on prend en compte le facteur performance, on essaiera d'éviter de le faire côté Front pour éviter d'impacter la réactivité de l'interface.

ITÉRATION 5

[ADMIN] Avoir un formulaire pour ajouter des profils utilisateur

USER STORY

En tant qu'administrateur,
je veux avoir un formulaire pour ajouter des profils utilisateur,
afin de permettre aux utilisateurs de personnaliser leur expérience sur la plateforme.

Critères d'acceptation :

1. Le formulaire doit permettre de saisir les informations suivantes :
    - Nom du profil.
    - Avatar ou image associée (facultatif).
    - Restrictions d'âge (ex. : 0 (Tous publics), 12, 16, etc.).
2. Une fois le formulaire soumis :

- Les données doivent être validées (ex. : champs obligatoires remplis, format correct).
- Le profil doit être ajouté à la base de données.
- Un message de confirmation doit être affiché (ex. : "Le profil a été ajouté avec succès.").

3. En cas d'erreur (ex. : champ manquant ou problème de connexion), un message d'erreur clair doit être affiché.

Remarque : un profil utilisateur n'est pas un compte utilisateur. Choisir plus tard un profil utilisateur n'impliquera pas une étape d'authentification.

FRONT (HTML / CSS / Javascript) - Indications

admin / data
Créez un nouveau module dataProfile.js qui regroupera toutes les requêtes HTTP adressées au serveur et concernant les profils utilisateur. Prévoyez une fonction (par exemple DataProfile.add) qui envoie au serveur les données utiles à l'enregistrement d'un nouveau profil utilisateur.
Rappel : s'agissant de l'envoi de données de formulaire, c'est une requête HTTP à faire en utilisant la méthode POST.

admin / component / ProfileForm
Créez un composant ProfileForm permettant la saisie des informations nécessaires à la création d'un nouveau profil utilisateur.

admin / index.html
Intégrer le composant ProfileForm à l'interface et ajouter une fonction (par exemple C.handlerAddProfile) pour gérer l'envoie des données de formulaire en cas de validation.

BACK (PHP / MySQL) - Indications

BDD
Il vous faut ajouter une table dans votre base de données pour y stocker les profils utilisateur
Attention à bien choisir les types de données et privilégiez un identifiant numérique en auto-incrément pour la clef primaire de votre table.

server/
Il faut prendre en charge un nouveau type de requête HTTP pour enregistrer un nouveau profil utilisateur dans la base. Par exemple
https://mmi.unilim.fr/.../script.php?todo=addProfile
Respectez comme d'habitude la structuration en script.php / controller.php / model.php

---

ITÉRATION 6

[APP] Pouvoir choisir un profil utilisateur

USER STORY

En tant qu'utilisateur de l'application,
je veux pouvoir sélectionner un profil utilisateur parmi ceux disponibles,
afin de personnaliser mon expérience en fonction de mes préférences et restrictions.

Critères d'acceptation :

1. Une liste des profils disponibles doit être affichée à l'ouverture de l'application (ou lors de la déconnexion d'un profil actif).
2. Chaque profil doit afficher au minimum :
    - Le nom du profil.
    - Une image ou un avatar associé (facultatif)
3. Une fois un profil sélectionné, l'application doit charger les paramètres et préférences associés à ce profil (sera traité dans les prochaines itérations).

FRONT (HTML / CSS / Javascript) - Indications

app / data
Créez un nouveau module dataProfile.js qui regroupera toutes les requêtes HTTP adressées au serveur et concernant les profils utilisateur. Vous prévoirez une fonction DataProfile.read (par exemple) qui interroge le serveur pour obtenir la liste des profils disponibles.

app / component / NavBar
Modifiez le composant NavBar pour y ajouter la possibilité de sélectionner un profile.

app / index.html
Vous aurez sans doute à modifier également le handler associé à la NavBar. Pour la suite, il pourra être pratique de conserver dans une variable globale l'identifiant du profil actif (actuellement sélectionné)

BACK (PHP / MySQL) - Indications

server/
Il faut prendre en charge un nouveau type de requête HTTP pour renvoyer les profils disponibles en base de données. Par exemple
https://mmi.unilim.fr/.../script.php?todo=readProfiles
Respectez comme d'habitude la structuration en script.php / controller.php / model.php

ITÉRATION 7

[APP] Filtrer les contenus selon l'âge du profil sélectionné

USER STORY

En tant qu'utilisateur de l'application,
je veux que les contenus affichés soient filtrés en fonction des restrictions d'âge associées à mon profil,
afin de ne voir que les films adaptés à mon âge.

Critères d'acceptation :

1. Lorsqu'un profil utilisateur est sélectionné, les restrictions d'âge associées doivent être appliquées automatiquement.
2. Les films non conformes aux restrictions d'âge du profil ne doivent pas être affichés dans la liste des films ou les catégories.
3. Si aucun film n'est disponible après application du filtre, un message clair doit être affiché (ex. : "Aucun film disponible pour votre tranche d'âge.").

FRONT (HTML / CSS / Javascript) - Indications

app / data
Dans le module dataMovie.js, modifiez votre fonction DataMovies.requestMovies afin de prendre en paramètre un âge limite.

app / index.html
Là où vous utilisez DataMovie.requestMovies il vous faut passer en paramètre l'âge du profil actuellement sélectionné. Si aucun profil n'est sélectionné on passera 0 en paramètre

BACK (PHP / MySQL) - Indications

server/
La requête HTTP pour demander les films comprend désormais un âge minimum. Par exemple :
https://mmi.unilim.fr/.../script.php?todo=readMovies&age=7
le Modifiez la chaîne de traitement de la requête pour prendre en compte ce nouveau paramètre et in fine, sélectionner dans la base de données uniquement les films dont l'âge conseillé est supérieur ou égal à la valeur du paramètre "age".

ATTENTION
Techniquement, il est aussi possible de récupérer tous les films côté Front puis de les filtrer avant affichage. Mais cela veut dire faire transiter des données qui ne serviront pas (i.e. tous les films au dessus de la limite d'âge). Donc ce n'est pas la bonne option en terme de consommation de la bande passante, pensez au forfait data de l'utilisateur !

ITÉRATION 8

[ADMIN] Pouvoir modifier un profil utilisateur

USER STORY

En tant qu'administrateur,
je veux pouvoir modifier les informations d'un profil utilisateur,
afin de mettre à jour ses données ou corriger des erreurs.

Critères d'acceptation :

1. Une interface doit permettre de sélectionner un profil utilisateur existant à modifier.
2. Les informations modifiables doivent inclure :
    - Le nom du profil.
    - L'avatar ou l'image associée (facultatif)
    - Les restrictions d'âge.
3. Une fois les modifications effectuées et validées :
    - Les données doivent être mises à jour dans la base de données.
    - Un message de confirmation doit être affiché (ex. : "Le profil a été modifié avec succès.").
4. En cas d'erreur (ex. : champ manquant ou problème de connexion), un message d'erreur clair doit être affiché.

FRONT (HTML / CSS / Javascript) - Indications

Basez-vous sur les TP et le formulaire permettant d'ajouter ou de modifier un menu.
Techniquement il n'y a pas de différence. Juste que ce sont ici des films au lieu de menus.

BACK (PHP / MySQL) - Indications

Même chose que pour le Front. Il faut juste adapter votre requête SQL pour qu'elle ajoute un profil lorsqu'il n'existe pas ou bien qu'elle le remplace si il existe déjà..

ITÉRATION 9

[APP] Pouvoir ajouter des films à une liste de favoris par profil utilisateur

USER STORY

En tant qu'utilisateur de l'application,
je veux pouvoir ajouter des films à une liste de favoris pour mon profil,
afin de retrouver facilement les films que je souhaite regarder plus tard.

Critères d'acceptation :

1. Chaque film doit avoir une option (ex. : un bouton ou une icône) pour l'ajouter à la liste de favoris.
2. Une fois un film ajouté :
    - Il doit apparaître dans la liste de favoris associée au profil utilisateur actif.
    - Un message de confirmation doit être affiché (ex. : "Le film a été ajouté à vos favoris.").
3. Si un film est déjà dans la liste de favoris, l'option d'ajout doit être désactivée ou remplacée par une option de suppression (la suppression sera réalisée dans une prochaine itération).
4. La liste de favoris doit être accessible depuis le profil utilisateur.

FRONT (HTML / CSS / Javascript) - Indications

app / data
Soon...

BACK (PHP / MySQL) - Indications

server /
Soon...

ITÉRATION 10

[APP] Pouvoir retirer un film de sa liste de favoris

USER STORY

En tant qu'utilisateur de l'application,
je veux pouvoir retirer des films de ma liste de favoris,
afin de gérer ma liste et supprimer les films que je ne souhaite plus conserver.

Critères d'acceptation :

1. Chaque film dans la liste de favoris doit avoir une option (ex. : un bouton ou une icône) pour le retirer.
2. Une fois un film retiré :
    - Il doit disparaître de la liste de favoris associée au profil utilisateur actif.
    - Un message de confirmation doit être affiché (ex. : "Le film a été retiré de vos favoris.").
3. Si la liste de favoris devient vide après suppression, un message clair doit être affiché (ex. : "Votre liste de favoris est vide.").

FRONT (HTML / CSS / Javascript) - Indications

app / data
Soon...

BACK (PHP / MySQL) - Indications

server /
Soon...

ITÉRATION 11

[APP] Avoir des films mis en avant

USER STORY

En tant qu'utilisateur de l'application,
je veux voir une section dédiée aux films "mis en avant",
afin de découvrir les films recommandés ou populaires.

Critères d'acceptation :

1. Une section "Films mis en avant" doit être visible sur la page d'accueil de l'application.
2. Les films "mis en avant" doivent inclure au minimum :
    - Le titre.
    - Une affiche ou une image représentative.
    - Une courte description ou synopsis.
3. Les films affichés dans cette section doivent être marqués comme "mis en avant" dans la base de données.
4. Si aucun film n'est marqué comme "mis en avant", un message clair doit être affiché (ex. : "Aucun film mis en avant pour le moment.").

FRONT (HTML / CSS / Javascript) - Indications

app / data
Soon...

BACK (PHP / MySQL) - Indications

server /
Soon...

ITÉRATION 12

[APP] Pouvoir consulter des statistiques

USER STORY

En tant qu'utilisateur de la plateforme,
je veux pouvoir consulter les statistiques d'utilisation du site,
afin de déterminer quels sont les films les mieux notés, combien de profils existent, etc.

Critères d'acceptation :

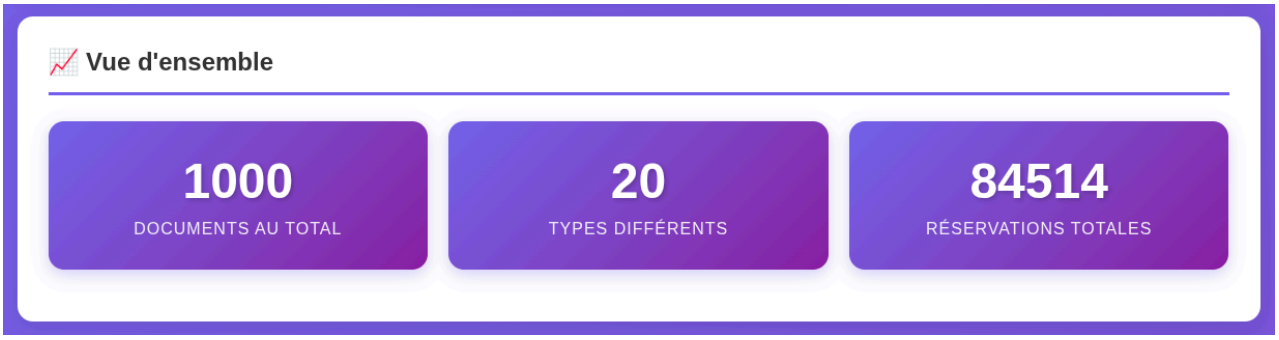
Une page dédiée et accessible depuis le menu doit permettre aux visiteurs de consulter différentes statistiques (vous pouvez en rajouter d'autres si vous avez des idées) :

1. Statistiques utilisateur & engagement
    - Nombre total de profils créés (Itération 5 & 6)
    - Nombre moyen de films par profil dans les favoris (Itération 9 & 10)

2. Statistiques films & catalogue
    - Nombre total de films dans la base
    - Film le plus ajouté aux favoris (Itération 9)
    - Catégorie la plus populaire (nombre de films favoris par catégorie — Itération 4)

FRONT (HTML / CSS / Javascript) - Indications

Au niveau du front c'est à vous de choisir comment faire apparaître ces données statistiques, à titre d'exemple voici le type d'affichage qu'on pourrait voir sur le site d'une médiathèque :



Bien entendu cette page doit s'intégrer graphiquement avec le reste du site, et l'idée est que les données affichées changent en fonction des films ajoutés, des notes données par les utilisateurs, etc.

BACK (PHP / MySQL) - Indications

Cette partie implique de lancer des requêtes vers la base de données adaptées à chaque donnée statistique souhaitée, par exemple pour calculer le nombre de films vous aurez sûrement côté serveur du code qui ressemble à :

```
// Dans model.php
function getTotalFilms() {
        $sql = "SELECT COUNT(*) AS total_films FROM SAE203_films";
        $stmt = $pdo->prepare($sql);
        $stmt->execute();
        return $stmt->fetch(PDO::FETCH_ASSOC);
}
```

ITÉRATION 13

[APP] Avoir un champ de recherche

USER STORY

En tant qu'utilisateur de l'application,
je veux pouvoir rechercher un film par son titre (obligatoire) ou d'autres critères (optionnel),
afin de trouver rapidement un film spécifique dans le catalogue.

Critères d'acceptation :

1. Une barre de recherche doit être disponible sur la page d'accueil.
2. L'utilisateur doit pouvoir rechercher un film en saisissant :
    - Le titre du film (partiel ou complet).
    - Optionnellement, d'autres critères comme la catégorie ou l'année de sortie.
3. Les résultats de la recherche doivent afficher les films correspondants sous forme de liste ou de cartes, incluant :
    - Le titre.
    - Une affiche ou une image représentative.
4. Si aucun film ne correspond à la recherche, un message clair doit être affiché (ex. : "Aucun film ne correspond à votre recherche.").

FRONT (HTML / CSS / Javascript) - Indications

app / component
Intégrez un champ de recherche (input) à votre NavBar. Possiblement, cela peut faire l'objet d'un sous-composant. Il y a 3 façons de déclencher une requête pour rechercher sur le serveur les films dont le titre contient le (ou les) mot clé saisi par l'utilisateur :

1. avec un bouton (onclick) en plus du champ de recherche
2. lorsque la valeur du champ est modifiée (onchange)
3. à chaque fois qu'une touche est "frappée" au clavier (onkeyup)

BACK (PHP / MySQL) - Indications

BDD
Vous aurez à un moment besoin de sélectionner les films qui contiennent un mot clé dans leur titre. Documentez-vous sur l'instruction SQL "LIKE" pour vous aider.

ITÉRATION 14

[ADMIN] Rechercher un ou des films pour gérer leur statut "mis en avant"

USER STORY

En tant qu'administrateur,
je veux pouvoir rechercher un film ou des films et leur attribuer ou retirer le statut "mis en avant",
afin de gérer les films recommandés visibles par les utilisateurs.

Critères d'acceptation :

1. Une interface de recherche doit permettre de rechercher un ou plusieurs films en fonction de critères comme :
    - Le titre du film (partiel ou complet).
    - La catégorie ou le genre (optionnel)
    - L'année de sortie (optionnel)
2. Les résultats de la recherche doivent afficher les films correspondants avec les informations suivantes :
    - Le titre.
    - La catégorie
    - Le statut actuel ("mis en avant" ou non).
3. L'administrateur doit pouvoir :
    - Attribuer le statut "mis en avant" à un film.
    - Retirer le statut "mis en avant" d'un film.
4. Une fois les modifications effectuées, un message de confirmation doit être affiché (ex. : "Le statut du film a été mis à jour avec succès.").
5. Si aucun film ne correspond à la recherche, un message clair doit être affiché (ex. : "Aucun film ne correspond à votre recherche.").

FRONT (HTML / CSS / Javascript) - Indications

admin / data / dataMovie.js
On doit pouvoir réutiliser (copier/coller) la même fonction réalisée pour la partie app dans l'itération précédente.

Et il faudra aussi pour faire une requête qui permet de demander au serveur de modifier le statut (mis en avant ou pas) d'un film.

admin / component
On pourra créer un composant avec un champ pour rechercher des films et les présenter dans un formulaire en incluant le statut mis en avant (ou pas) du film.

BACK (PHP / MySQL) - Indications

server /
On doit aussi pouvoir réutiliser le point d'entrée créé dans l'itération précédente pour rechercher des films sur la base d'un mot-clé (éventuellement en incluant dans la réponse du serveur l'information de statut (mis en avant ou non) du film.

ITÉRATION 15

[APP] Avoir un système de notation

USER STORY

En tant qu'utilisateur de l'application,
je veux pouvoir attribuer une note aux films que j'ai regardés,
afin de partager mon avis et aider les autres utilisateurs à choisir un film.

Critères d'acceptation :

1. Chaque film doit afficher une interface de notation (ex. : étoiles, score sur 10, etc.).
2. L'utilisateur doit pouvoir attribuer une note uniquement s'il est connecté à un profil.
3. Une fois une note attribuée :
    - Elle doit être enregistrée dans la base de données et associée au profil utilisateur.
    - Un message de confirmation doit être affiché (ex. : "Votre note a été enregistrée.").
4. Si un utilisateur a déjà noté un film, il ne peut pas le noter à nouveau
5. La note moyenne des utilisateurs doit être affichée pour chaque film.

FRONT (HTML / CSS / Javascript) - Indications

app / data
Soon...

BACK (PHP / MySQL) - Indications

server /
Soon...

ITÉRATION 16

[APP] Avoir un système de commentaires

USER STORY

En tant qu'utilisateur de l'application,
je veux pouvoir laisser des commentaires sur les films,
afin de partager mon avis et interagir avec d'autres utilisateurs.

Critères d'acceptation :

1. Chaque film doit afficher une section dédiée aux commentaires sur sa page détaillée.
2. L'utilisateur doit pouvoir :
    - Ajouter un commentaire (texte uniquement).
    - Voir les commentaires laissés par d'autres utilisateurs.
3. Les commentaires doivent inclure :
    - Le texte du commentaire.
    - Le nom ou pseudonyme du profil utilisateur ayant laissé le commentaire.
    - La date et l'heure du commentaire.
4. Un utilisateur doit être connecté à un profil pour pouvoir laisser un commentaire.
5. Si aucun commentaire n'a encore été laissé pour un film, un message clair doit être affiché (ex. : "Aucun commentaire pour ce film. Soyez le premier à en laisser un !").

FRONT (HTML / CSS / Javascript) - Indications

app / data
Soon...

BACK (PHP / MySQL) - Indications

server /
Soon...

ITÉRATION 17

[ADMIN] Avoir un système de modération des commentaires

USER STORY

En tant qu'administrateur,
je veux pouvoir modérer les commentaires laissés par les utilisateurs,
afin de garantir que seuls les commentaires appropriés soient visibles sur la plateforme.

Critères d'acceptation :

1. Une interface doit permettre aux administrateurs de :
    - Voir tous les nouveaux commentaires non modérés.
    - Marquer un commentaire comme "approuvé" pour qu'il soit visible par les utilisateurs.
    - Supprimer un commentaire inapproprié.
2. Les nouveaux commentaires doivent être visibles uniquement par les administrateurs jusqu'à ce qu'ils soient approuvés.
3. Chaque commentaire doit afficher les informations suivantes :
    - Le texte du commentaire.
    - Le nom ou pseudonyme du profil utilisateur ayant laissé le commentaire.
    - La date et l'heure du commentaire.
    - Le statut actuel du commentaire (ex. : "En attente", "Approuvé").
4. Une fois une action effectuée (approbation ou suppression), un message de confirmation doit être affiché (ex. : "Le commentaire a été approuvé avec succès.").
5. Si aucun commentaire n'est en attente de modération, un message clair doit être affiché (ex. : "Aucun commentaire à modérer pour le moment.").

FRONT (HTML / CSS / Javascript) - Indications

app / data
Soon...

BACK (PHP / MySQL) - Indications

server /
Soon...

ITÉRATION 18

[APP] Signaler les films récemment ajoutés à la plateforme

USER STORY

En tant qu'utilisateur de la plateforme,
Je veux que les films récemment ajoutés soient signalés avec un tag "new",
Afin de pouvoir identifier facilement les nouveautés disponibles.

Critères d'acceptation :

1. Les films ajoutés dans les 7 derniers jours doivent être marqués avec un tag "new".
2. Le tag "new" doit être visible sur la page de liste des films et sur la page de détails d'un film.
3. Le tag doit disparaître automatiquement une fois que le film n'est plus considéré comme "récent" (après 7 jours).
4. Si aucun film récent n'est disponible, aucun tag "new" ne doit apparaître.

FRONT (HTML / CSS / Javascript) - Indications

app / data
Soon...

BACK (PHP / MySQL) - Indications

server /
Soon...

ITÉRATION 19

[APP] Pouvoir consulter (plus) de statistiques

USER STORY

En tant qu'utilisateur de la plateforme,
je veux pouvoir consulter davantage de statistiques d'utilisation du site,
afin de déterminer quels sont les films les mieux notés, combien de profils existent, etc.

Critères d'acceptation :

La page de statistiques doit être plus complètes et inclure :

1. Statistiques utilisateur & engagement
    - Profil le plus actif (le plus de films ajoutés aux favoris ou de notes attribuées)
    - Nombre de commentaires publiés (Itération 16) — avec statut : en attente / approuvés (Itération 17)
2. Statistiques films & catalogue
    - Film le mieux noté (Itération 15) — avec moyenne des notes
    - Film le plus récent (Itération 18 — tag "new")

FRONT (HTML / CSS / Javascript) - Indications

Essayez de conserver une seule vue d'ensemble des différentes statistiques. Donc plutôt compléter les précédentes statistiques que de créer de nouvelles "pages"

BACK (PHP / MySQL) - Indications

soon...

---

4. Partie Intégration : Faire son Framework CSS

Méthodologie de travail pour chaque itération de la SAé

Tout au long de la SAé, chaque itération suivra un processus en deux étapes :

1. **Conception du prototype Figma** : Réalisez une maquette répondant à l'user story de l'itération en utilisant les composants déjà disponibles dans le projet Figma partagé. Si un composant nécessaire est manquant, créez-le et ajoutez-le à la bibliothèque du projet.
2. **Intégration HTML/CSS :** Chaque composant doit être intégré de manière autonome et réutilisable.

Remarque importante : Durant ces premières phases de la SAé, aucune identité visuelle particulière n'est attendue. **Conservez un style volontairement minimal, de type wireframe ou basse fidélité** : **concentrez-vous sur la structure et le comportement de vos composants, pas sur leur apparence**. La définition d'une charte graphique vous sera imposée lors de l'itération surprise, il sera alors temps d'habiller vos composants avec l'identité visuelle complète donnée.

Pourquoi construire son framework Css

Cette approche, basée sur le découpage, la décomposition de vos interfaces en petits composants vous permettra de construire votre propre framework CSS. 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.

Constituants du framework Css

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 au S3 comment utiliser un préprocesseur CSS comme Sass pour compiler automatiquement tous ces fichiers en un seul, ce qui est la méthode préférable en production.

Voici quelques pistes pour créer votre framework CSS :
- **reset** : neutralise les styles par défaut des navigateurs
- **variables** : approche par tokens - couleurs, typographie, espacements, breakpoints ...
- **classes utilitaires (Atomic 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.
ex : .mt-4, .text-sm, .bg-primary, .font-bold, .rounded ...
- **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.
ex: structure de page.flex, .grid, .grid-cols-3, .justify-center, .gap-4 ...
- **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).

Utilisez la **règle css "@layer**", cette règle css est utilisée pour déclarer une couche de cascade et peut également être utilisée afin de définir l'ordre de précédence lorsqu'il y a plusieurs couches de cascade.

Concentrez-vous sur la simplicité et la réutilisabilité de vos classes**.**

Documentez chaque classe et fonctionnalité de votre framework pour faciliter la compréhension et la maintenance du code.

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

Testez les performances de votre site avec Lighthouse :

Google Lighthouse est un outil d'audit de performances de sites Web développé par Google accessible depuis l'inspecteur de Chrome. Il analyse différentes métriques telles que le temps de chargement de la page, la performance de la mise en page, l'accessibilité, les bonnes pratiques de codage et le référencement naturel (SEO).
Une fois que l'audit est terminé, Lighthouse fournit un rapport détaillé sur les performances du site Web et des suggestions d'amélioration pour chaque métrique. Ces suggestions sont basées sur les meilleures pratiques de Google et peuvent aider les utilisateurs à optimiser la performance de leur site Web.
En utilisant les recommandations de Lighthouse, les développeurs de sites Web peuvent améliorer l'expérience utilisateur, augmenter le temps de visite et améliorer le référencement naturel de leur site.

Autre outils :
https://www.projectwallace.com/
https://csspeeper.com/

5. Partie Base de Données

Pour cette partie vous devez créer un sous-dossier **DocumentationBD/** contenant :

- un export de votre base de données au format .sql depuis PhpMyAdmin (**uniquement les tables qui concernent la SAé)**
- votre modèle complet créé avec Looping (fichier .loo)
- un fichier README.md qui explique, :
    - **pour chaque itération** : les requêtes SQL que vous avez utilisées dans votre code PHP, et si vous avez dû faire des modifications sur la base, en donnant une justification de vos choix concernant l'organisation des tables, les associations, les types de données, longueurs, clefs primaires, etc
    - une capture d'écran de la vue Looping
    - une explication rapide concernant les cardinalités (1, 1), (1, n), etc associées aux relations liant les tables

6. Livrables

Un dépôt Moodle vous permettra de déposer votre travail.

Le dépôt sera ouvert le jeudi 7 mai de 14:00 à 23:59.

Le lien du dépôt est https://community-iut.unilim.fr/course/view.php?id=3522

Vous devrez compléter un formulaire en ligne où il faudra (entre autre) renseigner :

- Le lien vers une landing page permettant d'accéder à vos différents sites hébergés (Itération de plus haut rang, itérations surprise, app et admin dans les 2 cas).
- Les accès administrateur (login/pwd) pour la partie admin.
- Le lien vers votre repository Github.

Vous devrez également déposer une archive. La taille de l'archive est limitée à 500 Mo et vous n'avez qu'une tentative. Dans cette archive, nous devons trouver :

- un document pdf qui montre l'analyse produite par Lighthouse sur vos sites et éventuellement une brève synthèse des modifications que vous avez pu apporter pour améliorer vos "métriques".
- l'intégralité de vos codes sources (admin, app, serveur). Si vous avez des difficultés à passer sous la barre des 500 Mo, n'incluez pas les images.