Cahier des Charges Projet WE4A

Ébauche de Réseau Social

Objectif : Créer un site où des utilisateurs inscrits peuvent échanger des messages courts. La manière la plus simple de résumer étant de dire : « un Facebook ou un Twitter simplifié ».
Votre site doit aussi intégrer des comptes « admin » qui ont le pouvoir d'effectuer différentes tâches de modération (cacher un message offensant, envoyer un avertissement, bannir temporairement ou définitivement un utilisateur, etc...)

Comme vous le constaterez en lisant ce document, nous ne vous demandons pas de réaliser toutes les fonctionnalités d'un véritable réseau social.
Nous ne répliquerons que les fonctionnalités principales et directement visibles pour les utilisateurs et admins, sans nous préoccuper des aspects plus complexes d'un tel site.

Langages à utiliser/contraintes techniques

Le site doit **fonctionner sur un serveur Apache** (comme celui de XAMPP, qui est très fortement recommandé) et **n'utiliser que les langages de base du Web** : HTML, CSS, PHP, Javascript.

La base de données utilisée sera une base de données relationnelle « classique » MySQL. Comme PHP, elle nécessite un serveur, et ce dernier est aussi inclus dans les packages Web « tout-en-un » comme XAMPP.
Pour éviter de rendre la correction et les barèmes ingérables, Symfony et tous les autres frameworks sont interdits pour le projet WE4A. C'est en partie B que vous pourrez utiliser certains d'entre eux. La partie A vise à vous former sur les **langages fondamentaux du Web**, car ils sont indispensables **que ce soit avec ou sans framework.**

Ce projet étant en collaboration avec SI40, **la note attribuée pour WE4A ne tiendra pas compte de la qualité de la base de données** *(pertinence des clés étrangères, types de relations entre données, absence de redondances, etc, etc.).* **La qualité des requêtes SQL présentes dans votre code sera aussi évaluée en SI40, et ignorée en WE4A.**
Néanmoins, il est tout de même possible de perdre des points en WE4A pour les choses liées à la base de données. Et ce dans les cas suivants :

- **Si une requête échoue et que cela résulte en une erreur critique PHP**, la page ne pourra pas être affichée du tout. Cela empêchera alors le correcteur de voir la page sans modification coûteuse en temps, et résultera donc en un malus côté WE4A
- **Si le code PHP d'une page web ne vérifie pas que les données reçues de la requête sont correctes** (bon noms de variables, etc) et qu'il tente de les afficher, il va se passer des choses étranges sur la page (textes vides ou warnings PHP, selon config du serveur). Si votre site inclut une telle page, il y aura un malus.
- Dans l'esprit du cas précédent, **une page qui ne gère pas correctement le cas où la requête ne renvoie aucun résultat amènera un malus.** *(exemple : aucun produit ne correspond à la recherche, et la page n'affiche pas un message clair pour le communiquer)*

Dit autrement: *les problèmes de SQL qui amènent des anomalies visuellement détectables dans le site vous pénaliseront en WE4A. Si les problèmes ont lieu dans la BDD d'une manière qui n'affecte pas les visuels, alors ça vous pénalisera en SI40.*

Barême de notation

Le projet est noté sur 100 à la base. Cette note sera ensuite ramenée sur 20, arrondi au dixième.

Comme vous le verrez ensuite , ce cahier des charges donne un barème indicatif des points accordés **en WE4A** pour la réalisation de chaque fonctionnalité. **Mais attention : le barème écrit ici n'est pas totalement strict.**
Cette année, le sujet est noté par fonctionnalité (feature) plutôt que « par page ». Cela signifie donc que vous avez une certaine liberté sur comment afficher, sur une ou plusieurs pages, ces interactions et leurs effets. (mais attention, l'énoncé de certaines ne le permet pas)
Une implémentation de qualité exceptionnelle sur une feature pourrait se voir accorder légèrement plus de points que le maximum, et à l'inverse, une feature 100 % absente ou totalement exécrable pourrait amener un score légèrement négatif, en particulier si cela a des répercussions importantes dans le reste du site (mais rassurez vous, cela sera réservé aux cas extrêmes).

- **Rapport :** 20 points
- **Fonctionnalités Utilisateur :** 40 points
  - Écrire et afficher un post *(4 points)*
  - Posts avec des images *(2 points)*
  - S'inscrire, se connecter *(5 points)*
  - Avoir sa propre page, écrire des posts sur sa page *(4 points)*
  - Like/Unlike les posts *(3 points)*
  - Découvrir facilement des posts ou pages d'inconnus *(3 points)*
  - Voir la page d'un autre utilisateur *(1 point)*
  - Pouvoir poster des commentaires/réponses sur les messages *(4 points)*
  - Pouvoir « follower » quelqu'un *(2 points)*
  - Fil regroupant les messages des gens que l'on follow *(2 points)*
  - Gérer les gens que l'on follow *(2 points)*
  - Gérer son compte (changer avatar, info perso…) *(4 points)*
  - Page de notifications *(5 points)*
- **Fonctionnalités Admin :** 15 points
  - Existence des comptes admin *(2 points)*
  - Envoyer un avertissement à un utilisateur *(3 points)*
  - Marquer un post comme « sensible » *(4 points)*
  - Retirer un post offensant/problématique *(3 points)*
  - Bannir un utilisateur, temporairement ou non *(3 points)*
- **Visuels/UX:** 15 points
- **Innovation/Ajouts:** 10 points

Le Rapport et le crédit des sources

Dans le développement Web, il est très courant d'avoir recours à des tutoriels, des templates, des bouts de code et des librairies gratuites partagées sur le Web.
Bien avant l'arrivée de chatGPT pour les amalgamer via I.A., ces ressources faisaient déjà partie du *workflow* des développeurs Web depuis des années, et nous ne pouvons donc pas vous les interdire.
Mais néanmoins, nous voulons combattre la mauvaise utilisation de ces ressources, c'est à dire celle qui consiste à tout recopier, sans rien comprendre et sans rien créditer.

Nous vous demandons donc de citer toutes vos sources externes de contenu, y compris les générations de bouts de code via I.A, dans votre rapport. Le faire est, contrairement à ce que certains pourraient croire, valorisant et vous rapportera des points. C'est une attitude attendue d'un ingénieur vraiment professionnel.
A l'inverse, toute tentative d'utiliser un contenu externe sans en créditer la source sera pénalisée.

Outils et Librairies facultatifs

En dehors des technos requises et énumérées précédemment, il y a trois outils/librairies dont vous avez le droit de vous servir, malgré le fait que nous n'en avons pas parlé formellement en TD :

- **Flex :** cette extension de CSS, désormais relativement ancienne, est souvent considérée comme en faisant partie intégrante, désormais. Mais pour dissiper tout doute : oui, vous avez le droit d'utiliser toutes les commandes CSS-Flex dans votre projet.
- **Bootstrap :** Ensemble d'outil pour construire plus facilement les HTML/CSS/Javascript d'un site, en utilisant une logique de « grille ». Vous avez le droit de vous en servir, en particulier si vous souhaitez créer vos propres visuels et/ou vous éloigner davantage d'un template téléchargé.
- **JQuery :** Cette librairie Javascript, très couramment utilisée, est autorisée. Attention : elle contient tellement de fonctions « clé-en-main » que son utilisation modifie la manière de coder le Javascript de façon significative.

Attention : A cause de nos contraintes sur le temps, nous n'avons pas le temps d'apprendre ces outils en WE4A. Si vous souhaitez vous en servir, il faudra que vous vous lanciez dans un auto-apprentissage assez conséquent.
Le bon usage de ces outils sera récompensé par des points d'innovation (voir plus loin).
Mais en revanche, un mauvais usage qui amènerait des bugs vous ferait probablement perdre davantage de points que ce bonus, et serait globalement négatif pour votre note.

Il est donc déconseillé de vous lancer dans l'usage de ces outils si vous n'êtes pas parfaitement à l'aise avec les outils que nous avons vu ensemble *(à part pour Flex, qui est bien plus accessible et va probablement sortir dans vos recherches sur le CSS de toute façon)*

Réfléchissez donc bien avant d'avoir recours à ces outils. Selon votre « bagage », cela pourrait représenter une prise de risque qui n'en vaut pas forcément la peine.

Importance des visuels et de l'ergonomie dans le Web

Comme vous l'avez vu, il y a 15 points sur les 100 consacrés aux visuels et à la « user experience », c'est à dire l'ergonomie et la clarté visuelle de votre site.
Comme l'UTBM n'est pas une école de design ou d'arts appliqués, nous ne demandons pas non plus des œuvres CSS extraordinaires. **Mais nous sommes en 2024 et un site trop laid et/ou trop délicat à utiliser n'est juste plus acceptable dans aucune entreprise.** Consacrer une partie des points à cet aspect sert à refléter cette réalité et à vous la faire intégrer le plus tôt possible.

Heureusement, pour ceux qui n'ont pas trop de «sensibilités design», **vous avez le droit de télécharger un des nombreux templates HTML/CSS gratuits qui sont disponibles sur le web, et de vous en servir comme base pour créer votre site.**
Comme expliqué précédemment : créditez votre source ! Vu le temps limité que nous avons pu consacrer à HTML et CSS dans cette UV, il sera assez facile pour nous de détecter l'usage d'un template, et donc de vous pénaliser si vous l'avez utilisé sans citer la source.
Soyez donc des ingénieurs honnêtes et créditez les ressources que vous utilisez.
Ces 15 points «visuels/UX» ne sont pas totalement «artistiques» : ils incluent aussi des éléments purement techniques, à savoir la qualité du code HTML/CSS et votre usage intelligent de Javascript et de AJAX pour améliorer le confort utilisateur. Ce n'est donc pas juste une histoire de «beauté des pages».

Il est hélas difficile de donner un barème précis sur ces 15 points. Mais je peux vous donner un ordre d'idée :

- Il est assez facile d'avoir 5/15 ou plus en UX : il suffit de ne pas faire quelque chose d'anormalement frustrant ou désagréable.
- Si vous utilisez des Javascripts et AJAX pour réduire les rafraîchissements de page complets et/ou créer des éléments d'interface améliorés, vous aurez facilement plus de 10/15.

Mais souvenez-vous: ces chiffres sont juste indicatifs. Un site avec peu de Javascript mais très joli pourra tout à fait atteindre 10 points aussi, par exemple. Ce qui compte au final, c'est l'expérience utilisateur. Ce « feeling » que votre site est sympa et pratique !

Règles pour le Rendu

Vous devez rendre un fichier ZIP qui contient les éléments suivants :

- **Un dossier nommé avec les noms de famille** des 3 participants, que vous pouvez éventuellement abréger (ex : ATHOS-PORTHOS-ARAMIS), et qui contient **toutes les sources du site** : fichiers PHP, HTML, etc, etc.

A côté du dossier, les autres fichiers. C'est à dire :

- **Votre rapport format PDF.** Incluez aussi vos noms de famille dans son nom de fichier
- **Un fichier .SQL que je peux exécuter dans PHPMyAdmin** afin de générer une copie de votre base de données. Ce fichier sera noté côté SI40, mais si il ne marche pas, il y aura une petite pénalité côté WE4A aussi.
- **Il est préférable que le script crée la BDD** en plus de son contenu. Donnez à votre BDD un nom contenant **au moins un** de vos noms de famille.
- Il est préférable que vous fournissiez une BDD contenant déjà **une certaine quantité de données exemple** *(les utilisateurs fictifs que vous avez crée pour vos propres tests peuvent tout à fait faire l'affaire)*

Définitions des termes employés dans ce document

Pour éviter toute ambiguïté, les termes utilisés dans ce cahier des charges sont définis ici.
La définition de certains termes suffit à expliquer la quasi-intégralité de la feature correspondante. Auquel cas cela sera signalé dans l'énoncé plus détaillé des features que vous trouverez après cette partie consacrée aux définitions.

Les « posts » : terme utilisé pour désigner les messages que les utilisateurs s'échangent sur le site. Un post est associé à son auteur et la manière la plus simple de le voir est de se trouver sur la page de l'auteur en question. Mais ils peuvent aussi être vus à travers la fonction « follow » où à travers l' « exploration aléatoire » du site *(selon comment vous implémentez cette dernière)*.
Un post contient un texte, et peut aussi contenir une image facultative qui sera affichée avec le texte. La vidéo n'est pas demandée, mais vous pouvez l'implémenter pour marquer des « points innovation » *(privilégiez plutôt l'intégration de vidéo Youtube que l'upload de fichier vidéo)*.

Module Login/Compte : Un élément d'interface présent sur toutes les pages du site.
Si l'utilisateur n'est pas loggé, le module est en mode « login » et affiche un formulaire de login ainsi qu'un lien «Créer un Compte », dont la destination est explicite.
Si l'utilisateur est connecté, le module est en mode « compte » et affiche trois liens/boutons : « Mes follows», « Mon Compte » et « Se déconnecter ». La destination de ces liens est explicite.

Utilisateur déconnecté : Utilisateur qui ne s'est pas identifié avec un nom et mot de passe. Cet utilisateur ne peut pas créer de post ou laisser de commentaires sur les posts des autres, mais il peut explorer le site, lire les messages et voir les pages des autres utilisateurs.

Compte utilisateur : Le compte d'un utilisateur ordinaire, enregistré dans la base de données. Si un utilisateur se connecte avec un tel compte, il peut créer des posts nouveaux sur sa page et poster des réponses/commentaires sur les posts des autres.
Un compte utilisateur peut être crée librement par n'importe qui, en se rendant sur une page de création de compte. Un compte client contient au minimum les informations suivantes : email, password, nom, prénom, date de naissance, adresse.
Le login a lieu avec la combinaison email + password. Mais si vous le souhaitez, vous pouvez demander à l'utilisateur de saisir un « username », et utiliser la combinaison username+password

Compte admin : Un compte utilisateur pour les modérateurs du site. Si un utilisateur se connecte avec un tel compte, il pourra accéder à des pages et/ou des boutons supplémentaires afin de réaliser ses tâches de gestion du site.

Pour la création des comptes gérants, deux options s'offrent à vous :

1. Les comptes gérant sont pré-crées dans la BDD ***(indiquez les nous dans le rapport)***
2. Il y a un formulaire de création de compte gérant, situé sur une page secrète dont il faut connaître l'adresse pour y accéder (*aucun lien n'y mène).*
   Le formulaire en question peut aussi faire usage d'un password secret qui doit être saisi pour que l'ajout soit validé, afin de doublement s'assurer qu'un compte admin ne peut pas être crée par un intrus. *(dans ce cas, donnez nous le lien et le password dans le rapport)*

Utilisateur : Lorsque ce n'est pas précisé, nous parlons d'un utilisateur connecté avec un compte ordinaire.

Admin/modérateur : Utilisateur connecté avec un compte admin

Like : Marque d'appréciation qu'un utilisateur peut laisser sur un post. Le nombre total de likes est visible sur un post, à côté du pouce levé qui sert aussi de bouton pour donner/enlever son propre like.

Follow/Abonnement : Un utilisateur A peut choisir de follow un utilisateur B. Dans ce cas, lorsque l'utilisateur B crée un post, l'utilisateur A peut le voir dans sa « liste de follows ».
Une notification peut également être envoyé à A sur sa page de notifications. Mais c'est une option désactivée par défaut, que l'utilisateur devra donc activer manuellement dans sa page de gestion de compte si il désire ce comportement (voir plus loin).
Selon le thème et le ton de votre site, vous pouvez préférer le verbe « s'abonner » pour cette fonctionnalité, voire créer votre propre terme, si vous pensez que c'est plus adapté.

Follower/Abonné : Si on reprend l'exemple précédent où A follow B, alors A est aussi un follower de B. Le nombre de followers d'un utilisateur est indiqué sur sa page personnelle.
La possibilité pour un utilisateur de connaître l'identité de ceux qui sont ces followers n'est pas exigée. Mais si vous la pensez pertinente, vous pouvez l'implémenter pour quelques points d'innovation.
Comme pour le follow, le terme follower peut être remplacé par un équivalent de votre choix, selon le thème et le ton de votre site, si il y en a un.

Liste de follows : Expression officieuse que j'utilise ici pour désigner la possibilité de voir, sur une même page ou un même onglet, l'intégralité des messages des gens qu'un utilisateur follow.
Pour chacun d'eux, il a l'option d'aller directement sur leur page, ou de se « désabonner »

Commentaire/Réponse : Il est possible pour un utilisateur connecté de répondre/réagir à un post avec un de ses propres posts.
Cela signifie qu'un post a non seulement un auteur, mais aussi un « parent » facultatif : un autre post. Cela peut amener à la création de chaînes de posts assez longues et vous devez donc pensez à comment votre affichage HTML/CSS va s'adapter à cela.
Encore une fois, selon votre préférence ou l'éventuel thème particulier de votre site, vous pouvez choisir un autre terme pour parler de ceci dans votre projet (réaction, re-blabla, etc...)

Notification : Une notification est un message qui est généré à l'intention d'un utilisateur lorsqu'il se produit quelque chose qui demande son attention :

- une réponse a été postée sur un de ses posts (peut être désactivé. Activé par défaut)
- un follow a crée un nouveau post (peut être activé. Désactivé par défaut)
- une action admin a eu lieu sur un de ses posts (ne peut pas être désactivé)

Lorsqu'un utilisateur a au moins une notification non lue, il faut attirer l'attention sur le bouton/lien qui permet l'accès à la page des notifications. Le classique nombre indiquant le nombre de notifications non lues, sur un fond rouge pétant, fera très bien l'affaire.
Vous pouvez ajouter d'autres effets pour attirer l'attention si vous voulez, mais le nombre est demandé (n'en faites pas trop non plus)

Une notification doit avoir une existence dans la base de données. Un booléen est forcément nécessaire pour se souvenir si la notification a été lue ou non.
Les notifications lues peuvent être détruites manuellement par l'utilisateur, ou alors celles vieilles de plus de 15 jours sont éliminées automatiquement quand l'utilisateur ouvre sa page notification.
Ce n'est pas le plus optimum, mais c'est simple et clair avec le niveau en PHP de WE4A.

Si vous le souhaitez, vous pouvez améliorer le système d'effacement automatique des vieilles notifications lues, pour gagner des « points d'innovation ».

Remarque : Comme les intitulés des notification sont prédéterminés à quelques paramètres près, les entrées dans la table n'ont pas forcément à stocker des chaînes string pour ça.

Description des features exigées

Écrire et afficher un post - 4 points

- Avoir un HTML/CSS permettant d'afficher un post minimal : avec l'avatar et le nom de son auteur, et bien entendu le texte du post.
- Cliquer sur l'avatar ou le nom de l'auteur d'un post affiché amène sur sa page personnelle
- Étant donné la récurrence de l'affichage de post dans le site, placez bien le code nécessaire dans une fonction que vous ré-utiliserez. Pour éviter de faire trop de requêtes sur la BDD, le paramètre de cette fonction sera le résultat d'une requête réalisé en amont, avec toutes les données du post à afficher.
- Un formulaire de saisie de post est accessible à un utilisateur connecté. Ou bien en haut de sa propre page personnelle, ou bien en cliquant sur un bouton « répondre/réagir/nom custom » qui apparaît sur tous les posts (pour peut que l'on soit connecté bien sûr).
- Le formulaire de saisie, dans le cas d'une réponse, doit être invisible/absent dans un premier temps, et rendu visible/ajouté à la page via Javascript quand on clique « répondre » *(plus de détails dans la feature « poster des commentaires/réponses. Les points sont aussi là bas!)*
- Lorsqu'une conversation a lieu, une arborescence va donc se créer : post original, réaction, réaction à la réaction, etc. C'est, théoriquement, potentiellement infini. Ce qui ne va pas sans poser certains problèmes d'UX/interface.
- Votre capacité à rendre les conversations instinctives à utiliser comptera donc, de facto, pour une bonne partie des points consacrés à l'UX

Posts avec des images - 2 points

- Améliorer le formulaire de saisie de post pour que l'on puisse uploader une image si on le désire (elle n'a pas à être obligatoire)
- Placez des limites qui font sens sur la mémoire et les formats autorisés pour l'image. Elle peut aussi être redimensionnée via votre codePHP pour garantir une certaine largeur/taille
- Améliorer l'affichage des posts pour afficher les images lorsqu'il y en a une.
- Les images doivent être sauvegardées côté serveur, dans un dossier dédié, et recevoir chacune un identifiant unique qui garantit l'absence de conflit.
- La méthode consistant à sauvegarder le binaire de l'image dans la BDD (BLOB) n'est pas autorisée pour ce projet.

S’inscrire, se connecter - 5 points

- Consacrez un espace dans le design de vos pages pour un « module login », qui sera visible sur toutes vos pages.
- Si on n’est pas connecté, ce « module login » propose un formulaire avec champ « email » (ou « username ») et un champ « mot de passe », ainsi qu’un bouton « se connecter ». Cliquer ce bouton envoie le formulaire et tente de connecter l’utilisateur.
- Si on n’est pas connecté, ce « module login » a aussi un bouton « créer votre compte ». Le cliquer amène sur une page séparée pour la création de compte.
- La création de compte nécessite de saisir au moins les informations suivantes : email, password, nom, prénom, date de naissance, adresse.
- L’utilisateur doit aussi pouvoir uploader un avatar. Si il n’en uploade pas, une image par défaut lui sera attribuée dans tout le reste de l’application.
- Vous pouvez ajouter d’autres informations si vous jugez cela pertinent pour la communauté que vous cherchez à créer autour de votre site. Notamment, proposer aux utilisateurs de saisir un pseudo pour qu’il n’apparaissent pas sous leur vrai nom.
- Le login est à priori basé sur une logique « email+mot de passe ». Mais vous pouvez remplacer le mail par le pseudo si vous le préférez.
- Niveau BDD, il est important que les identifiants de login soient uniques. Une création de compte doit donc être refusée si un e-mail (ou autre élément utilisé pour login) déjà présent dans la BDD est utilisé dans le formulaire de création de compte.
- Envoyer un mail de vérification demande l’accès à un serveur SMTP. Ce n’est pas demandé pour ce projet. Quand le nouveau compte est crée avec succès, vous pouvez donc connecter immédiatement l’utilisateur pour lui éviter de se login manuellement.
- Après un login réussi (et donc une inscription réussie aussi, puisque cela le connecte automatiquement), l’utilisateur est automatiquement redirigé vers sa page personnelle.
- Lorsqu’un utilisateur est connecté, le « module login » visible sur chaque page se transforme en « module compte » qui occupe le même espace.
- Le « module compte » montrele nom de l’utilisateur connecté (ou le pseudo si vous avez pris cet angle), son avatar, ainsi que 2 boutons : « Notifications » et « Se déconnecter ».
- Cliquer l’avatar et/ou le nom conduit l’utilisateur sur sa page personnelle. Cliquer « Notifications » amène sur la page des notifications (ou modifie la page via AJAX pour les afficher). Cliquer « sé déconnecter » met fin à la session et l’utilisateur devient un utilisateur invité/non identifié.
- Lorsque l’utilisateur se déconnecte, essayez de concevoir votre code pour qu’il revienne sur la même page après l’opération. Si vous n’y arrivez pas, ramenez le à la page d’accueil (cela vaudra une petite fraction de point en moins, mais mieux vaut ça que ne pas pouvoir logout)

Avoir sa propre page, écrire des posts sur sa page - 4 points

- A chaque compte utilisateur crée est associé une page personnelle, un peu comme le mur facebook où ce qui s'affiche si on choisist un utilisateur spécifique sur Twitter/X.
- La page personnelle affiche toutes les « chaînes de posts » qui commencent par un post crée par cet utilisateur. Ils sont rangés du plus récent au plus ancien.
- Si vous le voulez, vous pouvez aussi afficher les posts crées « en réaction/commentaire » par cette personne sur le mur. Mais auquel cas, il faudra les distinguer visuellement de façon intelligente, afin que l'on ne les confonde pas avec les « chaînes » initiées par cette personne.
- Pour chaque post, on peut voir au moins les posts en commentaire/réaction DIRECTE en dessous de chacun. (c'est à dire qu'on ne voit pas les réactions aux réactions)
- Si un post de réaction a elle même un ou plusieurs posts de réaction, un petit bouton « Voir les réponses » permet de les charger, via AJAX. On évite ainsi de charger toute une conversation par défaut, mais on a l'option de le faire.
- Cette logique devient un laborieuse pour les longues conversations : si il y a 5 réponses enchaînées formant une conversation, je devrai cliquer 4 fois « voir les réponses » ! Essayez donc de proposer un système plus malin pour naviguer les chaînes de discussion de 3 posts ou plus. Cela vous donnera des points d'UX et/ou d'innovation.
- La page d'un utilisateur n'affiche qu'un nombre N de posts/chaînes par défaut (les N plus récents donc). Si il y a davantage de posts, un bouton « charger plus de posts » en bas de la page permet de charger les N suivants et de les ajouter à la page via AJAX.
- Si un utilisateur connecté se trouve sur sa propre page personelle, un formulaire pour créer un nouveau post « de début de chaîne» est alors présent en haut de la page, permettant de créer un nouveau post et potentiellement de nouvelles chaînes de conversation.

Like/Unlike des Posts - 2 points

- Ajoutez a votre affichage de post un bouton en forme de pouce, accompagné d'un nombre. Comme vous pouvez vous en douter, ils vont servir à ajouter/enlever un like sur le post, et à afficher combien de likes ont été donnés à ce post.
- Lorsque l'utilisateur clique le bouton sur un post qu'il n'a pas déjà liké une requête est envoyée en AJAX pour ajouter le like. Le AJAX renvoie une réponse de succès ou d'échec. En cas de succès, le pouce change de couleur/d'image et le nombre de like augmente de 1. cela doit se faire dynamiquement sans recharger la page.
- Un utilisateur ne peut liker un post qu'une seule fois. Dans la BDD, il est donc important qu'un like soit associé non seulement au post concerné, mais aussi à celui qui l'a donné.
- L'utilisateur peut annuler un like déjà donné en cliquant sur le pouce à nouveau. Dans ce cas, une requête est envoyée en AJAX pour retirer le like. Le AJAX renvoie une réponse de succès ou d'échec. En cas de succès, le pouce revient à l'état initial « pas de like » et le nombre de like diminue de 1. cela doit se faire dynamiquement sans recharger la page.
- Lorsque des posts déjà likés sont affichés à l'écran, le « bouton de pouce » doit, évidemment, être déjà dans l'état correspondant à un « like donné »

Découvrir des posts ou pages d’inconnus - 3 points

- Sur un réseau social, un des principaux intérêt est de pouvoir découvrir des informations et/ou d’autres utilisateurs intéressants. Et la logique complexe des algorithmes de recommandation est ce qui fait cela sur un site réel comme ceux que vous connaissez bien.
- Pour notre site simplifié, seules deux méthodes « d’exploration » sont demandées :
  - une page « Les derniers posts » qui affiche les posts les plus récents , rangés du plus récent au plus ancien, parmi l’intégralité des utilisateurs.
  - Une page « les posts les plus populaires », qui affiche les posts rangés en ordre décroissant sur le nombre de likes.
- Ces pages ont une logique assez proche de celle de la page personnelle d’un utilisateur : les X posts les plus récents sont affichés, et un bouton « charger plus de posts » permet de charger les X suivants.
- Ce système étant très basique, il n’est pas le plus pratique ou naturel. L’améliorer peut donc s’avérer un bon angle pour tenter d’obtenir une partie des 10 points d’innovation. Quelques pistes pour cela : permettre à l’utilisateur de choisir une catégorie pour ses posts, contraindre l’affichage à un intervalle de temps (aujourd’hui, dernière semaine, etc.), associer des mots-clés aux posts, une fonction de recherche, etc.

Voir la page d’un autre utilisateur - 1 point

- Un utilisateur, connecté ou non, peut librement regarder les pages personnelles de tous les autres utilisateurs. Il peut ainsi voir leurs posts et y réagir.
- Ces pages s’affichent, globalement de manière très similaire à ce qu’on voit sur sa propre page. Mais le formulaire pour poster un post « de début de chaîne » est absent.
- Si l’utilisateur est connecté, il peut aussi choisir de follow cette personne. Les détails (ainsi que les points) de cet aspect se trouvent plus loin.
- Rappel : la manière la plus commune d’arriver sur la page de quelqu’un est de cliquer sur son nom/pseudo dans un de ses posts.

Pouvoir poster des commentaires/réponses - 4 points

- Un utilisateur connecté a la possibilité de répondre/réagir/commenter à n’importe quel post (y compris les siens) avec un autre post. Vous pouvez utiliser le terme de votre choix dans votre site, voire créer le vôtre. Dans cette partie, j’utiliserai « réponse ».
- Pour peu que l’on soit connecté, chaque post doit donc venir avec un petit bouton «répondre ». Un formulaire pour écrire un nouveau post doit alors être ouvert dynamiquement en dessous de ce post.

- Pour le formulaire « ouvert dynamiquement », deux options :
  - chaque post est crée avec un formulaire, mais celui ci est caché (hidden) par défaut, et cliquer le bouton le fait apparaître.
  - L'autre option : le HTML du formulaire est ajouté dynamiquement après le post via Javascript.
- Les problèmes liés à l'affichage des réponses sur les posts ont été mentionnés plus tôt lorsque nous avons parlé des pages personnelles et de l'affichage des posts, car cela était pertinent. **MAIS TOUS LES POINTS POUR CE QUI EST LIE AUX RÉPONSES SONT DANS CETTE PARTIE.**

Pouvoir « follower » quelqu'un - 2 points

- Il est possible de « follow » un utilisateur, le but final étant de recevoir des notifications lorsque cet utilisateur crée un nouveau post (de début de chaîne)
- Au minimum, un bouton suivre/follow doit être visible sur la page personnelle de chaque utilisateur. ...A part soi même, si on est connecté.
- Cliquer le bouton enregistre le follow dans la BDD. Si on suit quelqu'un, le bouton est remplacé par un autre avec un texte de type « Vous suivez cette personne », et qu'on peut cliquer pour se désabonner.
- Si on n'est pas connecté , le bouton peut être invisible. Ou alors, il est visible mais change de comportement pour que, si on le clique, un message de type « vous devez être connecté pour pouvoir suivre quelqu'un » doit s'afficher.
- Lorsqu'on clique le bouton pour se désabonner, une confirmation est demandée via javascript avant de modfier la BDD pour enlever le follow.
- Les conséquences du follow, comme les notifications, sont considérées comme d'autres features.

Fil regroupant les messages des follow - 2 points

- Un utilisateur connecté a accès à un lien vers une page « Suivis ». Comme souvent, vous pouvez choisir un autre terme adapté pour désigner cette page. (« Que font mes amis ? » etc)
- Sur cette page, on peut voir les posts (début de chaîne seulement) qui ont été crées par les gens que l'on follow. Rangés, comme d'habitude, du plus récent au plus ancien.
- Cette page a un affichage assez similaire aux pages personnelles ou à la page où l'on voit « les posts les plus récents parmi tous les utilisateurs ».
- Comme sur ces autres pages, X posts sont affichés au maximum. Et on peut « charger plus de posts »

Gérer les gens que l’on follow - 2 points

- Sur la page « Suivis », il y a une section listant les noms de tous les gens que l’on follow (cela peut être un onglet, une colonne sur le côté, etc...)
- A côté de chaque nom, on peut trouver un bouton pour se désabonner/arrêter de follow cette personne (ils peuvent reprendre l’apparence de ceux sur les pages individuelles, mais ce n’est pas obligatoire)
- Ce n’est pas obligatoire, mais vous pouvez intégrer un bouton pour « follow/unfollow »un utilisateur directement dans les posts, et ce sur toutes les pages. Cela pose quelques difficultés (mettre à jour toutes les occurrences, etc.) donc cela vaudra des points en UX et/ou en innovation si c’est bien fait.

Gérer son compte - 4 points

- Cette page affiche les informations saisies lors de la création du compte : email, password, nom, prénom, date de naissance, adresse. Ainsi que l’image d’avatar.
- Si vous avez ajouté la saisie d’autres informations dans votre création de compte, elles doivent aussi être visibles sur cette page.
- Il est possible, depuis cette page, de modifier toutes ces informations. SAUF CELLE QUE VOUS UTILISEZ COMME CLE PRIMAIRE (normalement, c’est l’information que vous utilisez pour le login et qui n’est pas le mot de passe).
- Utilisez votre expérience de navigattion sur le Net pour rendre l’édition de ces informations aussi naturelle et pratique que possible (par exemple, en demandant de saisir deux fois un nouveau mot de passe pour s’assurer qu’il n’y a pas eu de typo)
- Lorsque l’avatar est changé, assurez vous que l’ancien fichier d’image est bien retiré du serveur.

Page de notifications - 5 points

- Cette page a pour rôle d’afficher des notifications à l’utilisateur, ou bien pour l’informer des nouveaux posts de ses follow, ou bien des décisions admin sur ses posts ou son compte (voir plus loin).
- La page de notifications est accessible via un lien VISIBLE SUR TOUTES LES PAGES, POURVU QUE L’ON SOIT CONNECTE.
- Si on a des notifications non lues, un numéro sur fond rouge apparaît à côté de ce lien, indiquant combien de notifications non lues qui s’y trouvent. Assurez vous que cette information attire bien l’attention et ne soit pas perdue dans un coin. Vous pouvez même introduire d’autres changements visuels en plus du numéro rouge pour cela.

- Une notification est une entité qui doit exister dans une table spécifique de la BDD. Elles doivent pouvoir contenir un message texte et être associées à un post (de façon facultative).
- Imaginons qu'un utilisateur A est abonné à (follow) l'utilisateur B. Lorsque B crée un nouveau post, A va recevoir une notification « B a crée un nouveau post ». A peut la voir sur sa page de notifications, et elle affiche une copie du post en question, permettant à A d'y répondre/réagir immédiatement si il le désire.
- Lorsque qu'une notification est affichée pour la première fois sur la page dédiée chez A, elle a un visuel différent pour notifier qu'elle n'avait pas encore été lue jusqu'à maintenant.
- Une notification affichée « non lue » sera modifiée immédiatement après pour être considérée « lue » lors des affichages ultérieurs, ce qui va affecter le « nombre d'alerte rouge » que A peut voir.
- Les notifications lues sont toujours affichées sur la page des notifications, mais ont un style différent des « affichées pour la première fois »
- Toutes les notifications ont un bouton permettant à l'utilisateur de les effacer. Une confirmation doit être demandée via Javascript avant de l'exécuter. Vous pouvez retirer ces notifications de la BDD, où les conserver en leur plaçant un marqueur indiquant qu'elles sont « à la corbeille » et ne doivent plus s'afficher.
- Lorsque l'utilisateur affiche la page des notifications, toutes les notifications LUES vieilles de plus de deux semaines sont automatiquement effacées avant affichage.
- La « page de notification » peut, techniquement, être une colonne, un onglet ou une sous fenêtre, et pas une page totalement séparée. Ces options qui permettent de rester sur la même page peuvent contribuer à améliorer l'UX et marquer plus de points sur cet aspect

---

Existence des comptes admin – 2 points

- La table sauvegardant les utilisateurs doit intégrer un booléen isAdmin permettant de savoir si un utilisateur est admin ou non.
- Les comptes admin sont ou bien pré-crées et fournis dans votre export de BDD, ou bien il existe une page secrète (= aucun lien n'y mène et il faut taper l'adresse manuellement) qui permet de créer des comptes admin.
- Dans les deux cas, fournissez bien les informations nécessaires VERS LE DÉBUT DE VOTRE RAPPORT (adresse de la page secrète, login des comptes pré-crées…)
- Sur la page personnelle et les posts crées par les comptes admins, on peut voir une icône qui informe les visiteurs que cette personne est un admin.

Envoyer un avertissement à un utilisateur - 4 points

- Lorsque l'on est connecté en temps qu'admin, les posts ont des boutons/options supplémentaires pour réaliser des actions de modération. Essayez de regrouper ces actions dans un menu contextuel/une pop-up qui va s'ouvrir en cliquant un bouton « actions admin »
- Une des options dans la « pop-up de modération » est « envoyer un avertissement à cet utilisateur ». Lorsqu'il est cliqué, un formulaire de création de notification apparaît, pré-rempli avec un texte d'avertissement par défaut, mais que l'admin peut changer avant de l'envoyer.
- Lorsque l'avertissement est envoyé, l'utilisateur concerné reçoit une notification avec un intitulé de type « Vous avez reçu un avertissement pour comportement inapproprié ».
- Lorsque cette notification est sélectionnée sur la page des notifications, l'utilisateur averti peut voir le message qui avait été saisi par l'admin lors de l'envoi.

Marquer un post comme « sensible » - 4 points

- Le second choix dans le « menu pop-up admin » est de marquer un post comme choquant/violent afin d'en cacher la vue. Les utilisateurs pourront décider de « voir quand même », mais l'idée est d'éviter que les gens voient ces posts par accident.
- Beaucoup d'effets de cette action sont communs avec l'avertissement : lorsque l'admin décide de marquer un post comme sensible, un formulaire apparaît pour saisir un texte de justification pour la décision. Un texte générique par défaut est déjà présent dans le champ input, mais l'admin peut le modifier/personnaliser avant de l'envoyer.
- L'utilisateur concerné reçoit une notification avec un intitulé de type « Un de vos posts a été marqué comme sensible par nos modérateurs ».
- Lorsque cette notification est sélectionnée sur la page des notifications, l'utilisateur peut voir non seulement le message custom saisi par l'admin, mais aussi une copie du post qui a été affecté par la décision.
- Lorsque le post marqué comme sensible est vu dans un autre contexte (sur la page personnelle, dans une notification de nouveau post, etc.), le contenu du post (texte et image) sont cachés et remplacés par un texte disant « ce post a été marqué comme sensible et potentiellement choquant par les administrateurs. Vous pouvez le voir quand même en cliquant ici. »
- Comme le texte l'indique, si on clique l'endroit approprié sur le « cache », alors le cache est retiré et on peut voir le post normalement. Cette opération doit absolument être faite par AJAX, sans recharger la page.
- Pour améliorer et gagner des points innovation, vous pouvez implémenter l'option pour l'admin de revenir sur sa décision, et rendre le post « normal » à nouveau, envoyant au passge une notification à l'auteur de post.

Retirer un post offensant/problématique - 3 points

- Le troisième choix dans le « menu pop-up admin » est pour les cas encore plus extrêmes de post inacceptable : Retirer ce post.
- Les posts « retirés » ne sont, en réalité, pas effacés de la base de données : ils sont marqués comme « retirés » via un booléen (ou avec une connection à d'autres tables si cela vous semble plus pertinent). Les réseaux sociaux font toujours ça pour garder une trace et contribuer à leur algorithmes de recommandation ou de publicité.
- Beaucoup d'effets de cette action sont communs avec les autres fonctions admin : lorsque l'admin décide de marquer un post comme sensible, un formulaire apparaît pour saisir un texte de justification pour la décision. Un texte générique par défaut est déjà présent dans le champ input, mais l'admin peut le modifier/personnaliser avant de l'envoyer.
- L'utilisateur concerné reçoit une notification avec un intitulé de type « Un de vos posts a été retiré par nos modérateurs ».
- Lorsque cette notification est sélectionnée sur la page des notifications, l'utilisateur peut voir non seulement le message custom saisi par l'admin, mais aussi une copie du post qui a été affecté par la décision (c'est aussi pour ça qu'un post « retiré » ne doit pas être vraiment effacé)
- A part dans la notification à la personne concernée, les posts marqués comme « retirés » ne s'affichent plus dans aucune autre page du site. Adaptez donc vos requêtes SQL pour qu'ils soient ignorés lors des recherches.
- Encore une fois, implémenter une manière de revenir en arrière n'est pas exigé, mais est une idée simple pour marquer quelques points d'innovation.

Bannir un utilisateur, temporairement ou non - 3 points

- Le dernier choix dans le « menu pop-up admin » est concu pour être utilisé en cas d'offenses répétées, et permet de bannir un utilisateur. Temporairement ou non.
- Les utilisateurs bannis définitivement ne doivent pas être retirés de la base de données. En effet, leurs messages resteront visibles malgré tout. Les messages d'un utilisateur banni doivent être visuellement différents et contenir un message visible en rouge informant de la durée de ban restante.
- Beaucoup d'effets de cette action sont communs avec les autres fonctions admin : lorsque l'admin décide de marquer un post comme sensible, un formulaire apparaît pour saisir un texte de justification pour la décision. Un texte générique par défaut est déjà présent dans le champ input, mais l'admin peut le modifier/personnaliser avant de l'envoyer.
- Cette fois-ci, un autre champ est disponible pour choisir la durée du ban : on peut saisir un nombre d'heures ou sélectionner une case à cocher « ban définitif ». Le ban étant une décision plus lourde en conséquences que les autres, le formulaire demande aussi une confirmation avant d'être envoyé.
- Lorsqu'un utilisateur est banni, sa page personnelle est modifiée pour montrer le statut de banni, en indiquant la durée restante. (Le plus : utiliser javascript pour que le temps restant change dynamiquement sous nos yeux)
- L'utilisateur banni reçoit bien entendu une notification « Vous avez été banni jusqu'au XX/XX/20XX à XXhXX », où il peut voir le texte de justification tapé par l'admin.
- Vous avez deux choix pour ce qui se passe lorsqu'un utilisateur banni se connecte au site :

    - Ou bien vous lui retirez simplement sa capacité à créer des posts (nouveaux ou réponses), auxquel cas les boutons et formulaires habituels seront remplacés par des messages « vous êtes banni et ne pouvez plus poster de message jusqu'à telle date »)
    - L'utilisateur ne peut voir d'autres pages que sa page personnelle et sa page de notifications. Si il tente de le faire, il est redirigé vers sa page personnelle.
- Le « ban » ne verrouillant pas totalement l'accès au site, même si il est définitif, vous pouvez utiliser un autre terme ou expression, comme par exemple « compte verrouillé »
- Comme pour les autres fonctionnalités admin, implémenter des manières de revenir en arrière et modifier/annuler un ban n'est pas exigé, mais est une manière de marquer des points d'innovation.