Travail pratique #05

Objectifs

1. Se familiariser avec les failles les plus répandues dans les applications web
2. Assurer une protection contre les attaques web les plus courantes
3. Programmer une application web
4. Mettre en pratique la matière vue en classe

Règles importantes

1. Veuillez commenter votre code *intelligemment*
2. Votre code doit être de qualité exemplaire
3. Votre site doit être fonctionnel pour qu'il puisse être corrigé. Je ne pourrai pas accorder de points pour les fonctionnalités manquantes ou pour les erreurs m'empêchant de tester votre site.

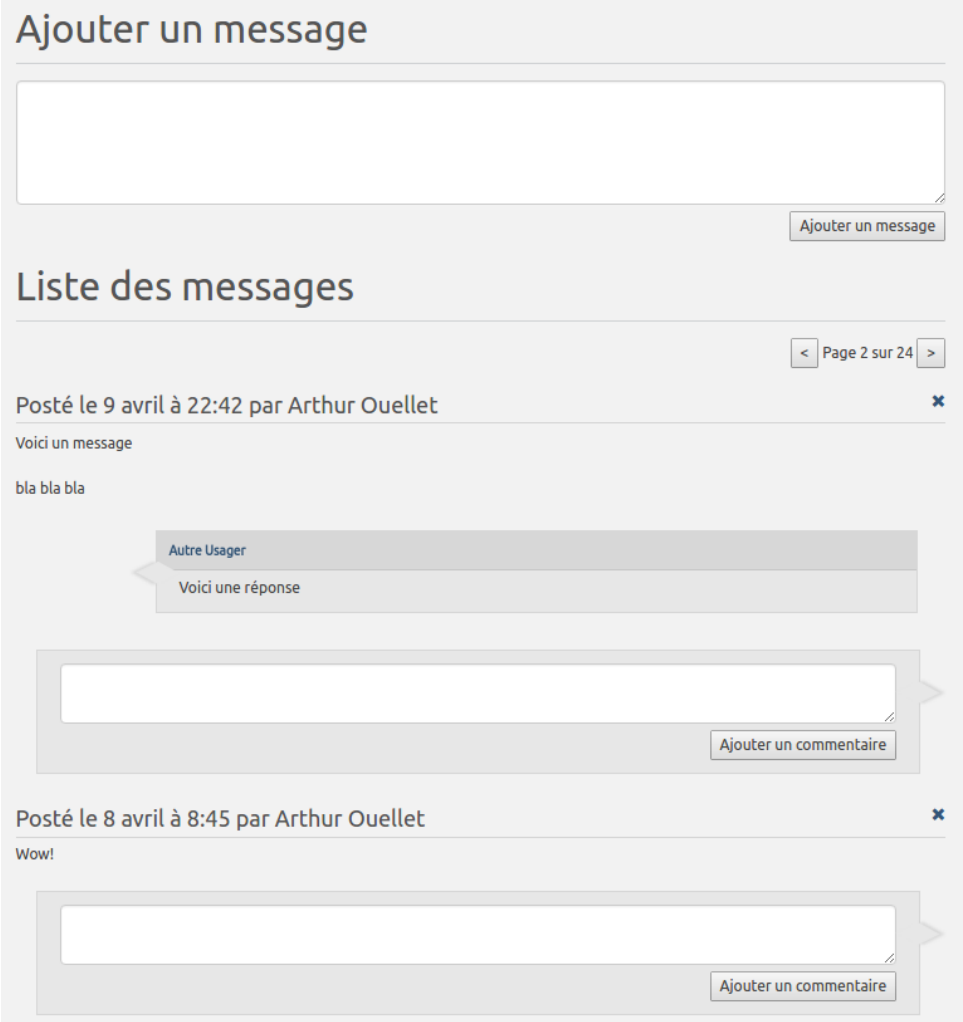
Instructions

En utilisant **le langage, le framework et le serveur de base de donnée de votre choix**, veuillez concevoir une application web **sécuritaire**. Le langage choisi devra permettre de faire tout ce qui est demandé dans cet énoncé.

Petite contrainte quant aux langages que vous pouvez choisir :

1. Si vous avez la possibilité d'héberger vous-même votre site web (votre propre serveur, hébergement gratuit en ligne, votre ordinateur chez vous) et **maintenir le serveur online et accessible publiquement pour une durée minimale de 2 semaines après la remise de ce travail, 24 heures sur 24, 7 jours sur 7**, il n'y a aucunes restrictions pour le langage que vous pouvez choisir et le type de base de données utilisé. En plus des fichiers source de votre site, vous devrez aussi me remettre un **fichier texte nommé "url.txt" contenant l'adresse de votre site web**.
2. Si par contre vous ne désirez pas ou ne pouvez pas héberger votre site web en ligne, celui-ci devra être fait avec l'environnement suivant :
   (a) **Environnement .Net Core/Framework** : Pour tester et corriger, je démarrerai votre projet dans Visual Studio en appuyant sur le bouton *Play* et tout devra s'initialiser correctement (incluant la base de donnée). Il est de votre responsabilité que ce soit le cas, veuillez le tester avant pour être certain.

Votre site web doit permettre d'afficher une série de messages et d'en poster de nouveaux. Il s'agit d'une application de type *forum* (en version simplifiée) qui affiche une succession de messages sur un *mur*.



Vous devez au minimum implémenter les fonctionnalités suivantes :

1. Toutes les informations et les données nécessaire au bon fonctionnement de votre site doivent être stockées dans une base de donnée
2. Avoir un formulaire de connexion permettant à l'usager d'entrer son nom d'usager et son mot de passe. Seuls les usagers avant un login valide pourront utiliser votre application.
3. Une fois connecté, l'usager voit un formulaire lui permettant d'ajouter un message sur le mur
4. Juste dessous le formulaire d'ajout doit se trouver la liste des messages, triée par date d'ajout en ordre décroissant (du plus récent au plus vieux).
5. Chaque message affiché doit montrer le **nom complet** de l'usager ayant envoyé le message, la **date de création** du message et le **contenu** du message.
6. Il doit être possible pour les utilisateurs de créer des messages sur plusieurs ligne et d'ajouter des retour chariots (*enter*) qui devront être préservés lors de l'affichage dans la liste des messages. Les messages peuvent contenir n'importe quels caractères et le contenu affiché dans la liste des messages doit être identique à ce que l'usager a entré (même <, >, ", etc).
7. Chaque message peut recevoir des commentaires, qui seront affichés du plus ancien au plus récent immédiatement en dessous de chaque message. Pour chaque commentaire, afficher seulement le nom complet de l'usager ayant fait le commentaire et le contenu du commentaire.

8. Un maximum de 5 messages doit être affiché par page, il doit être possible de changer de page (avec un bouton, un lien, ou autre) pour afficher les 5 messages suivants (ou précédents lorsque applicable) et ainsi de suite pour parcourir tout les messsages.
9. Les usagers peuvent être de 2 types :
   (a) Utilisateur normal : Peut envoyer des messages, répondre à des messages et effacer ses propres messages et réponses.
   (b) Administrateur : Peut envoyer des messages, répondre à des messages et effacer n'importe quel message et réponse.

Vous devez vous assurer de gérer correctement les éléments de sécurité suivants :

1. Injections dans la base de données et accès non-controlé aux données (OWASP 2019 : A3)
2. Gestion sécuritaire de la session des usagers (httponly, pas d'information sensible dans les cookies, token de session fiable, etc, OWASP 2019 : A7)
3. Gestion des accès et privilèges (OWASP 2019 : A1+A2)
4. XSS (OWASP 2019 : A3)
5. Stockage de mots de passe sécuritaire (salt, hash, ...)

Assurez-vous de tester votre site avec les versions récentes de Edge, Chrome et Firefox

Bien que cela serait essentiel si votre application web était utilisée dans un contexte réel, vous n'avez pas à implémenter des mesures de sécurité qui dépassent strictement la partie programmation de votre site web (certificat, redirection https, firewall, ...).

Références

1. `https://www.owasp.org/index.php/Main_Page`
2. `http://code.tutsplus.com/tutorials/http-the-protocol-every-web-developer-must-know-part-1--net-31177`
3. `https://www.owasp.org/index.php/Session_Management_Cheat_Sheet`

Remise

Vous devez me remettre sur vortex le code source complet de votre site web, ainsi que tout autre fichier nécessaires (voir en début de document pour les fichiers supplémentaires à remettre selon que vous ayez choisi d'héberger votre site vous-même ou non).

Vous devez aussi me remettre une liste de 3 combinaisons d'usager/mot de passe valides permettant d'utiliser votre site web (1 usager ayant les doit administrateur et 2 usagers normaux)