TD-1 Sécurité et Authentification dans l'Application de Gestion de Tâches

1. Mesures de Sécurité pour Protéger les Données des Utilisateurs

La protection des données des utilisateurs est un aspect crucial du projet. Voici les principales mesures à mettre en œuvre pour assurer une sécurité optimale :

Chiffrement des Données Sensibles

- **Chiffrement des mots de passe** : Lors de la création d'un compte, les mots de passe des utilisateurs doivent être **hachés** (non stockés en texte clair) à l'aide d'algorithmes de hachage sécurisé tels que **bcrypt** ou **Argon2**. Cela garantit que même en cas de fuite de données, les mots de passe ne seront pas utilisables.
- **Chiffrement des données sensibles en transit** : Toutes les communications entre le client (navigateur ou application mobile) et le serveur doivent être protégées par le protocole **HTTPS** via des certificats SSL/TLS. Cela empêche les attaques de type **man-in-the-middle**, où les attaquants pourraient intercepter les informations sensibles (comme les identifiants de connexion ou les données des projets).

Protection contre les Attaques Courantes

- **Protection contre les injections SQL** : Toutes les requêtes vers la base de données doivent être préparées à l'aide de requêtes préparées ou d'ORM (Object Relational Mapping) pour éviter les injections SQL, où un attaquant pourrait insérer du code malveillant dans une requête SQL.
- **Protection contre le Cross-Site Scripting (XSS)** : Pour éviter que des scripts malveillants ne soient injectés dans les formulaires ou champs de texte (comme les descriptions de tâches), toutes les entrées utilisateur doivent être soigneusement validées et échappées.
- **Protection contre les attaques Cross-Site Request Forgery (CSRF)** : Un **token CSRF** sera inclus dans chaque formulaire de l’application pour vérifier que les requêtes faites par les utilisateurs proviennent bien du site et non d'une source externe malveillante.

Stockage Sécurisé des Données

- **Base de données sécurisée** : L'accès à la base de données doit être limité aux applications et utilisateurs autorisés, en utilisant des politiques d’accès strictes.
- **Sauvegarde régulière** : Des **sauvegardes chiffrées** des données doivent être effectuées régulièrement pour prévenir les pertes de données en cas de problème matériel ou de cyberattaque.

Authentification à Deux Facteurs (2FA) *(Optionnel)*

- Pour renforcer la sécurité, l'**authentification à deux facteurs (2FA)** peut être proposée. Après avoir entré le mot de passe, un code envoyé par e-mail ou par une application d'authentification serait requis pour accéder à l'application.

---

2. Authentification des Utilisateurs

L’authentification des utilisateurs est essentielle pour assurer que seuls les utilisateurs autorisés puissent accéder à leurs projets et données. Voici comment cette authentification sera mise en œuvre :

Système d'Authentification Basé sur des Jetons (JWT)

- **JSON Web Tokens (JWT)** : L’authentification des utilisateurs sera gérée via des **tokens JWT**. Lorsqu'un utilisateur se connecte, le serveur lui renvoie un token JWT signé qui contient des informations sur son identité (par exemple, son ID utilisateur et son niveau d'autorisation).
    - Ce token est stocké côté client (généralement dans un **cookie sécurisé** ou dans le **localStorage**) et est envoyé avec chaque requête pour prouver l’identité de l’utilisateur.
    - Le token est signé avec une clé secrète, garantissant qu'il ne peut pas être falsifié.
    - **Expiration du token** : Les tokens JWT doivent avoir une durée de vie limitée pour réduire les risques d’utilisation abusive en cas de vol. Des tokens à durée courte (par exemple, 15-30 minutes) et des **tokens de rafraîchissement** peuvent être utilisés pour prolonger la session de manière sécurisée.

Processus de Connexion et de Déconnexion

- **Connexion** : Lors de la connexion, l'utilisateur soumet son e-mail et son mot de passe (chiffré). Le serveur vérifie ces informations et, si elles sont correctes, génère un token JWT qui est renvoyé au client.

- **Déconnexion** : Lorsque l'utilisateur se déconnecte, le token JWT est invalidé du côté serveur (ou simplement supprimé du côté client). Un système de **liste noire (blacklist)** peut être mis en place pour empêcher les tokens révoqués d’être utilisés à nouveau.

Sécurisation de la Session Utilisateur

- **Cookies sécurisés** : Si des cookies sont utilisés pour stocker les tokens JWT, ils doivent être marqués comme **HTTP-only** et **secure**, empêchant l'accès via JavaScript (protection contre les attaques XSS) et assurant qu'ils ne sont envoyés que via des connexions sécurisées (HTTPS).
- **Contrôle des sessions** : Pour éviter les détournements de session, chaque utilisateur doit pouvoir **gérer ses sessions actives** et déconnecter les appareils inconnus ou non utilisés depuis l'application.

---

3. Gestion des Autorisations d'Accès aux Listes de Tâches

Il est essentiel de bien gérer les autorisations d'accès pour garantir que les utilisateurs n'aient accès qu'aux projets et tâches auxquels ils sont associés, avec les droits appropriés. Voici comment cette gestion sera effectuée dans l'application :

Système de Permissions par Projet

Chaque projet aura un propriétaire, qui pourra inviter d'autres utilisateurs avec différents **niveaux de permissions** :

- **Lecteur** : Peut uniquement **visualiser** le projet, les groupes de tâches et les tâches. Il n'a pas le droit d'apporter des modifications.
- **Éditeur** : Peut **modifier** le projet, ajouter/supprimer des groupes de tâches, et modifier les tâches existantes.

Ces permissions sont stockées dans la base de données dans la table `member_of`, qui associe chaque utilisateur à un projet et à son niveau de permission.

Gestion des Accès aux Données

Lorsqu’un utilisateur effectue une requête (par exemple, pour afficher les tâches d’un projet), le système doit :

1. **Vérifier son identité** via le token JWT fourni.

2. **Vérifier ses permissions** en consultant la table `member_of` pour voir s'il a accès au projet et quel est son niveau d'autorisation.
3. Si l'utilisateur a les droits requis, la requête est autorisée, sinon, une erreur 403 (Forbidden) est retournée.

Attribution des Tâches et Groupes de Tâches

Les utilisateurs peuvent être affectés à des **groupes de tâches** ou à des **tâches individuelles**. Lorsqu'une tâche est attribuée à un utilisateur, celui-ci doit avoir les permissions nécessaires pour interagir avec elle (selon qu'il soit lecteur ou éditeur du projet global).

Exemple d'Autorisation

- **Utilisateur A** est propriétaire d'un projet et invite **Utilisateur B** en tant que lecteur.
  - Utilisateur B peut voir les tâches et suivre l'évolution du projet, mais il ne peut pas modifier les tâches ou en créer de nouvelles.
  - Si **Utilisateur A** accorde à **Utilisateur B** les droits d'éditeur, ce dernier pourra alors modifier les tâches existantes, créer de nouvelles tâches, et réorganiser les groupes de tâches.

Vérification d'Accès Avant Chaque Action

Avant toute action critique (comme la suppression d'une tâche ou la modification d'un projet), l'application doit vérifier :

1. Si l'utilisateur a les **permissions nécessaires**.
2. Si la ressource (tâche ou projet) est bien accessible à cet utilisateur (via un contrôle sur les permissions d'accès aux projets).

Cela garantit une gestion fine des accès et des droits pour chaque utilisateur, permettant une collaboration sécurisée sur les projets.

Conclusion

En résumé, la **sécurité** et l'**authentification** de l'application de gestion de tâches sont assurées grâce à un ensemble de mesures robustes, incluant le chiffrement des données sensibles, l'utilisation de tokens JWT pour l'authentification, et des contrôles d'accès rigoureux basés sur les permissions pour garantir que seuls les utilisateurs autorisés puissent interagir avec les projets et les tâches. Des mécanismes de protection contre les

attaques courantes (XSS, CSRF, injections SQL) sont également mis en place pour sécuriser l’ensemble du système.