1. Backend\server.ts  
   **Import de modules** : Vous avez correctement importé les modules nécessaires **express** et **cors**. Assurez-vous que ces modules sont bien installés en tant que dépendances dans votre fichier **package.json**.
2. **Middleware CORS** : Vous avez configuré le middleware CORS pour gérer les requêtes cross-origin. Cela permet à votre application front-end Angular de communiquer avec votre serveur backend. Assurez-vous que l'origine spécifiée (**http://localhost:4200**) est correcte pour votre application front-end.
3. **Routes API** : Vous avez défini plusieurs routes pour votre API, notamment pour récupérer la liste complète de "sesamiens", rechercher par nom et récupérer par ID. Cela semble correct en termes de fonctionnalité.
4. **Port d'écoute** : Vous avez configuré le serveur pour écouter sur le port 5000. Assurez-vous que ce port est disponible et ne sera pas en conflit avec d'autres applications en cours d'exécution sur votre système.
5. **Log de démarrage** : Vous avez ajouté un message de log pour indiquer quand le serveur démarre. C'est une bonne pratique pour le débogage.
6. **Gestion des erreurs** : Il pourrait être utile d'ajouter une gestion des erreurs pour gérer les cas où une route demandée n'existe pas ou lorsque les données ne sont pas trouvées (par exemple, en renvoyant une réponse 404).
7. **Sécurité** : Assurez-vous de mettre en œuvre des pratiques de sécurité appropriées, telles que la validation des entrées utilisateur et la protection contre les attaques par injection SQL.
8. **Modularisation** : À mesure que votre application backend devient plus complexe, envisagez d'organiser votre code en modules distincts (par exemple, pour les routes, les contrôleurs, les modèles) pour maintenir une meilleure lisibilité et une meilleure gestion du code.
9. **Validation des paramètres d'URL** : Pour éviter les erreurs de conversion, assurez-vous de valider et de sécuriser les paramètres d'URL (par exemple, **sesmienId**) avant de les utiliser dans des opérations.
10. **Tests unitaires** : Envisagez d'ajouter des tests unitaires pour vos routes et fonctions pour garantir que votre application fonctionne correctement.
11. **Documentation** : Si ce n'est pas déjà fait, envisagez de documenter votre API pour que les autres développeurs puissent facilement comprendre comment l'utiliser.