🔐 Exercice Backend NestJS – API Utilisateurs + Authentification + Gestion de Produits

🎯 Objectif

Créer une API NestJS avec authentification JWT permettant aux utilisateurs de s'inscrire, se connecter, gérer leur profil et effectuer des opérations CRUD sur des produits qu'ils ont créés.

📅 Durée estimée : 3 jours

🧱 Modèle 1 : Utilisateur

Champ	Type	Contraintes
id	UUID	auto-généré
name	string	requis
email	string	unique, requis, format email
password	string	requis, min 6 caractères
role	string	`user` ou `admin` (par défaut: user)
createdAt	Date	auto-généré

📦 Modèle 2 : Produit

Champ	Type	Contraintes
id	UUID	auto-généré
name	string	requis, min 3 caractères
description	string	optionnel
price	number	requis, supérieur à 0
ownerId	UUID	lien vers l'utilisateur créateur
createdAt	Date	auto-généré

🔐 Fonctionnalités d’authentification

Méthode	Endpoint	Description
POST	`/auth/register`	Inscription (retourne un JWT)
POST	`/auth/login`	Connexion (retourne un JWT)

- Hachage du mot de passe avec `bcrypt`
- Token JWT avec `@nestjs/jwt`
- Guard `JwtAuthGuard` pour les routes protégées

👤 Fonctionnalités utilisateur

Méthode	Endpoint	Description	Accès
GET	`/users/profile`	Voir ses propres informations	Protégé
GET	`/users`	Voir tous les utilisateurs	Admin only

📦 Fonctionnalités produit (CRUD)

Méthode	Endpoint	Description	Accès
GET	`/products`	Liste des produits	Protégé
GET	`/products/:id`	Voir un produit en détail	Protégé
POST	`/products`	Créer un produit	Protégé
PUT	`/products/:id`	Modifier un produit (propriétaire)	Protégé
DELETE	`/products/:id`	Supprimer un produit (propriétaire)	Protégé

🛠️ Contraintes techniques

- **NestJS CLI**
- **JWT** pour sécuriser les endpoints
- **bcryptjs** pour le hachage
- **TypeORM** ou **Prisma** (avec SQLite ou PostgreSQL)
- **DTO + class-validator** pour la validation
- Architecture modulaire (`auth`, `users`, `products`)

🧪 Bonus à implémenter

- Guard `RolesGuard` pour sécuriser les routes admin
- Middleware pour logger les requêtes
- Swagger (`@nestjs/swagger`) pour documenter l'API
- Tests unitaires sur le `ProductService` ou `AuthService`

📦 Livrables attendus

1. Projet bien structuré (controllers, services, modules, DTO)
2. `README.md` avec :
    - Instructions pour lancer le projet
    - Exemple de requêtes HTTP (auth + produits)
3. `.env.example` avec les variables d'environnement
4. Code versionné sur GitHub

Envoyer un mail à l'adresse wuracareer@gmail.com en notifiant la fin de la tache et en insérant le lien vers le github

Dernier delai : vendredi 14 juillet 2025 à 20h , passer ce délai vous serez pénalisez