CAHIER DE CHARGE BACKEND

Projet : Win Academy

Type : API REST centralisée
Framework : NestJS

---

1 OBJECTIF DU BACKEND

Le backend doit :

- Exposer une API REST sécurisée
- Servir deux applications :
    - App Utilisateur (étudiants)
    - App Admin
- Gérer toute la logique métier
- Gérer authentification + autorisation
- Gérer génération de certificats
- Garantir intégrité des données
- Être scalable et maintenable

---

2 ARCHITECTURE GLOBALE

Architecture modulaire NestJS :

```
src/
│
├── main.ts
├── app.module.ts
│
├── prisma/
├── auth/
├── users/
├── formations/
├── categories/
├── modules/
├── contents/
├── enrollments/
```

```
├── evaluations/
├── attempts/
├── certificates/
├── departments/
├── communes/
│
└── common/
```

---

3 RÔLES SYSTÈME

USER

- S’inscrire
- Se connecter
- S’inscrire à une formation
- Suivre modules
- Passer évaluations
- Télécharger certificat

ADMIN

- Gérer utilisateurs
- Gérer formations
- Gérer modules
- Gérer contenus
- Gérer évaluations
- Voir statistiques
- Valider ou bloquer comptes

---

4 SÉCURITÉ

- Global Validation Pipe
- DTO + class-validator
- Global Exception Filter
- Rate limiting
- CORS configuré
- Helmet
- Sanitisation données

---

6 DOCUMENTATION API

- Swagger auto via Nest
- Routes groupées par module
- Auth protégée

---

7 RÈGLES MÉTIER CRITIQUES

- Email unique
- Impossible supprimer formation si users inscrits
- Impossible générer certificat sans validation
- Seuls admins peuvent publier formation
- Progression calculée automatiquement
- Score calculé côté serveur uniquement

---

8 PERFORMANCE

- Pagination obligatoire
- Index DB sur :
    - email
    - formationId
    - userId
- Transactions Prisma pour opérations critiques

---

9 ENVIRONNEMENTS

- Dev
- Staging
- Production

Variables .env :

- DATABASE_URL
- JWT_SECRET

- JWT_REFRESH_SECRET
- PORT

🔟 LIVRABLES ATTENDUS DE L'ÉQUIPE

1. API complète fonctionnelle
2. Documentation Swagger prête
3. Fichier Prisma schema propre
4. Scripts migration
5. Seed initial admin
6. Gestion erreurs propre
7. Structure modulaire respectée
8. Tests basiques sur auth + evaluation

RÉSULTAT ATTENDU

Une API :

- Stable
- Sécurisée
- Modulaire
- Scalable
- Prête à connecter :
    - Frontend User
    - Frontend Admin

Si tu veux, prochaine étape :

Je peux te faire :

- 📊 Le schema Prisma complet
- 🔄 Le diagramme relationnel
- 🧱 L'ordre optimal de développement module par module
- 🔐 La stratégie d'auth avancée (access + refresh rotation)

On construit ça comme une banque, pas comme un cybercafé 🚀