Sujet de TP – Développement Back-End Complet

Ce projet peut vous permettre de justifier des compétences clés sur votre Bachelor, de vous vendre à des entreprises, de frimer devant des collègues de l'école, ou simplement d'être fiers de votre taff.

Vous êtes des concepteurs. Vous ne faites pas “juste du code” : vous créez, vous imaginez et vous faites naître un produit complet. Un bon développeur ne pense pas seulement à ce qu’il écrit, mais à **comment** il le construit, **pour qui**, et **avec qui**.

Vous montez une solution qui doit respecter vos collaborateurs, vos utilisateurs, et les standards professionnels du marché. Vous devez être réellement satisfaits de votre travail, et cela passe par du sérieux, de la rigueur et une vraie collaboration au sein de votre binôme. Testez vos fonctionnalités, documentez votre API, conceptualisez l’architecture, réfléchissez à vos choix techniques.

Le “développement” pur n’est qu’une petite partie du cycle de création d’une application et ici d’un back-end complet. Ce TP a pour but de vous faire vivre cette démarche de conception et de compréhension globale, pas simplement d’écrire du code.

Vous n'êtes pas des machines, vous êtes des créatifs.

Vous n’êtes pas des exécutants, vous êtes des architectes.

Vous n’êtes pas ici pour recopier, mais pour penser, pour concevoir, pour résoudre, pour améliorer.

Vous avez les outils, vous avez les bases, maintenant vous devez montrer votre capacité à transformer des idées en un produit solide, cohérent et professionnel.

Ce travail, c’est le vôtre. Appropriez-vous-le. Faisons les choses sérieusement, et faisons-les bien.

Organisation du travail

- Le projet est **obligatoirement réalisé en groupe de 2 personnes**.
- Indiquez **les noms et prénoms** des deux membres du groupe en première page du PDF.
- Chaque membre doit **participer activement** au développement, à la documentation et à l’analyse.
- Vous devez être capables d’**expliquer tout le projet**, même les parties codées par votre binôme.

Objectif général

Vous devez concevoir, développer et documenter **un back-end complet** sur la thématique de votre choix, par exemple :

- 🎮 Steam / Jeux vidéo

- 🎬 Cinéma / Films
- 📺 Netflix / Streaming
- 👥 Réseau social
- 🎓 myEfrei / Gestion de cours
- 🍽️ Restaurants / Réservations
- 💪 Application sportive
- … ou tout autre sujet validé avec moi.

Contraintes techniques obligatoires

Architecture & Code

Votre projet doit impérativement utiliser :

- **Express.js** organisé en **REST + MVC + POO**
- **Séparation claire des couches** :
  - Routes
  - Controllers
  - Services / Business Logic
  - Models (SQL + NoSQL)
  - Middlewares
- **CORS configuré**
- **Rate Limiter** (limitation du nombre de requêtes)
- **JWT Auth complet** :
  - Inscription
  - Login
  - Refresh token
  - Gestion de rôles (admin/user)

Bases de données

- **PostgreSQL** → Base relationnelle principale
- **MongoDB** → Base complémentaire (logs, historiques, données non structurées…)

Sécurité

- Gestion stricte des tokens JWT
- Hashage des mots de passe
- Validation des données
- Rate limiting
- CORS
- Vérification des permissions et rôles
- M'offrir une pinte
- Gestion des erreurs centralisée

Tests

- **Tests d'intégration complets**
  - Utilisation de supertest ou équivalent
  - Tests sur routes publiques
  - Tests sur routes protégées
  - Mock DB ou base de test dédiée

Documentation

- **Swagger** complet (généré automatiquement)
- Définition de tous les endpoints
- Description des bodies, params, headers
- Codes de réponses et erreurs

Options bonus (non obligatoires, si vous désirez aller plus loin)

- API secondaire en **GraphQL**
- Utilisation de **Prisma** avec PostgreSQL
- Pipeline CI/CD simple
- Utilisation de Docker (en dev)

Travail attendu (livrable)

Les livrables devront absolument être soignés, avec un niveau de langue adapté à un environnement d'entreprise.

Votre PDF suivra impérativement la nomenclature suivante (sinon perte de points) : NOM1-NOM2-BEND-EFREI-DEV3-202511-LIV.pdf

Vous devez rendre :

- **Votre projet complet sur un repository Git public avec des commits clairs et une gestion de branches correcte** (vous aurez également un README sur le repository qui décrit l'utilisation de votre application)
- **Un Rapport format PDF complet avant vendredi 10 novembre à 12h40**. Il contiendra au minimum les éléments ci-après.

Expression de besoin

- Contexte du projet
- Objectifs
- Acteurs / rôles
- Scénarios utilisateur (au moins 5)
- Cas d'usage clés

Analyse des sécurités mises en place

Décrivez précisément :

- CORS
- Rate Limiting
- Token JWT
- Rôle / permission
- Hashing
- Middleware de validation
- Gestion des erreurs
- Politique de refresh
- Justification des choix

Choix des technologies (avec schéma)

Pour chaque techno :

- Pourquoi cette technologie ?
- Dans quel rôle est-elle utilisée ?
- Avantages / limites
- Intégration technique
- Schéma d'architecture complet :
    - Client → API → Services → Bases de données

MCD et MLD

- **MCD Merise** : entités, relations, cardinalités
- **MLD** : tables, PK, FK, types
- Justification de la répartition SQL vs NoSQL

Explications de code

Incluez des extraits pertinents :

- Route
- Controller
- Model PostgreSQL
- Model MongoDB
- Middleware JWT
- Rate limiter
- Tests d'intégration
- Swagger

Expliquez les extraits avec vos **propres mots**.

Schéma des échanges entre API et bases

Un diagramme montrant :

- la requête client
- le passage dans les middlewares
- le controller
- l'accès SQL ou NoSQL
- la réponse

- les erreurs possibles

Documentation Swagger

- Capture d’écran
- Explication de la structure
- Lien (local) vers la documentation

Vulgarisation des technologies**

Pour chaque technologie :

- Express.js
- REST
- POO
- MVC
- PostgreSQL
- MongoDB
- CORS
- Rate Limiter
- JWT
- Swagger
- (optionnel) GraphQL
- (optionnel) Prisma

Faites une **explication simplifiée en 3 à 6 lignes**, avec vos mots.