PROJET P3 **Mission - Pilotez le développement d'une solution informatique DATASHARE**

SEMAINE 1 : TRAVAIL EFFECTUE

ACTIONS EFFECTUEES :

ACTION 1 : SUIVRE LES COURS

J'ai suivi 2 cours :

1) Devenez un expert de Git et GitHub.
2) Passez au Full Stack avec Node.js, Express et MongoDB

ACTION 2 : ETUDE PRELIMINAIRE

1) Prendre en compte la mission, les recommandations
2) Etudier la spécification

ACTION 3 : REALISATION D'ETAPE 1: Architecture technique du projet DATASHARE

1- Objectif du projet

Développer un prototype de plateforme de transfert de fichiers sécurisé, modulaire et maintenable, en pilotant l'implémentation avec un copilote IA et en assurant la qualité du code (tests, documentation, supervision).

2- Architecture technique retenue

Composant	Choix	Justification
Back-end	**NestJS (Node.js)**	Framework modulaire, structuré, orienté architecture propre. Idéal pour piloter un projet avec un copilote IA. Support natif TypeScript, DI, tests, validation, sécurité.
Front-end	**Angular**	Conforme aux maquettes du projet, robuste, parfait pour un MVP structuré.
Base de données	**PostgreSQL**	Système relationnel mature, fiable, performant. Très adapté aux contraintes d'intégrité (ex : email unique).
ORM	**TypeORM**	Intégration native avec NestJS, migrations, relations explicites, cohérence avec PostgreSQL.
Stockage fichiers	**Local (filesystem)**	Suffisant pour un MVP. Simple, rapide à mettre en place. Évolutif vers S3 ou équivalent plus tard.
Conteneurisation	**Docker + Docker Compose**	Isolation complète des services, reproductibilité, environnement maîtrisé, idéal pour CI/CD et pour piloter un projet multi-composants.
Gestion de versions	**Git + GitHub**	Suivi propre des évolutions, collaboration facilitée, intégration possible avec actions CI/CD.

Ajout: Dockerisation complète du backend

Avec un Docker

- **Reproductibilité totale** : même environnement pour moi, ton mentor, et la production.
- **Isolation** : PostgreSQL, backend NestJS et futur frontend Angular tournent dans des conteneurs séparés.
- **Déploiement simplifié** : un seul docker-compose up lance tout le stack.
- **Standard**: indispensable dans un contexte DevOps / SRE Data.
- **Gestion propre des dépendances** : plus de conflits Node/PostgreSQL sur la machine locale.

Structure Docker actuelle

- **Service** backend
    - Build multi-stage (builder + production)
    - Installation des dépendances
    - Compilation NestJS
    - Exécution des migrations TypeORM
    - Lancement de l'application
- **Service** postgres
    - Volume persistant
    - Configuration via variables d'environnement
    - Démarrage automatique avant le backend
- **Réseau Docker dédié**
    - Communication sécurisée entre services
    - Isolation du reste du système

Puis j'ai commencé à étudier le projet P3 dont la mission consistant à piloter le développement d'une solution informatique DATASHARE

Lecture de du projet "Développement du prototype de transfert de fichier"

a) Concevoir l’architecture et le modèle de données.

b) Piloter l’implémentation des fonctionnalités clés, assigner des tâches de code à un copilote IA, puis superviser et revoir ce code.

c) Assurer la qualité et la maintenabilité du code avec des tests, du débogage et de la documentation.

J’ai donc commencé à voir quelle l’architecture technique est adaptée du projet DATASHARE