NestJS - Fil Rouge - SPA Animals

Repo Github Loïc : Github Repo Tf Webapps 2023 - Nest js

Phase 1 - Création projet + crud de base sur objet user
Installation NestJs -> npm i @nestjs/cli
Création de projet -> nest new projectName
Lancement du serveur -> npm run start:dev

Explication générale de nest module + controller + service (voir draw.io) annexe

Phase 1 - Exercise
Sur base du projet que vous avez créé, le client nous demande de préparer les endpoint possibles pour sa SPA, la partie user et admin sera vue en tant que fil rouge du suivi de formation.
Le client demande :

- 1 : On peut faire rentrer un animal dans la spa, mais également le faire sortir ! c'est le but...
- 2 : Que l'on puisse récupérer tous les animaux ou un seul
- 3 : que l'on puisse récupérer les animaux d'une tel sorte (chat chien suffiront)
- 4 : que l'on puisse mettre à jour les vaccins de pupuce le chat (flag VaccineUpdate à true suffira)
- 5 : malheureusement, on doit pouvoir indiquer un décès de pupuce le chat.
- 6 : fort heureusement, nous sommes en 2050 et pupuce peut revivre grâce à chatGPT...

Phase 2 - Les filtres (exception), les DTO, Les pipes (les validations)
Création des types DTO du user
Validations des entrées de requête et application des filtres
pour appliquer les class validators dans les DTO il faudra installer les deux package suivant
→ npm i --save class-validator class-transformer

Phase 2 - Exercise
Appliquer les nouvelles choses apprise sur les DTO et les validations !,
Pour la validation global ne pas oublier d'aller check le main.ts ! :)

pour les curieux, vous pouvez faire de la simplification de DTO avec @nestjs/mapped-types - npm (npmjs.com) -> les omission, les pick, et les partial de mapped-type :) bonne chance.. → doc :Introducing Mapped Types for NestJS - Trilon Consulting

Phase 3 - TypeOrm
après la configuration des entités et de type orm niveau appmodule et module for feature on va pouvoir injecter notre repo
voici la doc de la class Repository fournie par typeorm : Repository API | TypeORM
La doc des spécifications des find : Find Options | TypeORM

Autre doc indépendante super pratique : [Select using Query Builder | TypeORM Docs (biunav.com)]
Nous avons vu les méthode CRUD complète
la majorité des méthodes TypeOrm on été utilisée OU lue au minium

Phase 3 - TypeOrm - Exos
Compléter votre crud incoming, nous le ferons ensemble aussi !!!! pour bien corrigé avec les validator et dto

Phase 4 : Relational et lifecycle
mise en place des relation entre user et donateur de croquette + crud adapté

- Mto
- Otm

Après midi - mise en place des lifecycle DeletedAt CreatedAt UpdatedAt + méthodes liée, remove, softdel, del etc.. (cru**B)**

Phase 4 Exos - TypeOrm - relational et lifecycle
Compléter votre crud incoming, nous le ferons ensemble aussi !!!! pour bien corrigé avec les validator et dto

- Ajouter une simple relation entre specie et animal (logique en même temps...)
- adaptez votre crud dessus ...

Si tout est ok, tenter de créer aussi, un OTO de animal vers Vaccine (`isVaccine`)
Si tout est ok , prenez le temps de splitter animalCtrl et SpecieCtrl et VaccineCtrl vers 3 modules complets.
PS : avant la théorie sur les lifecycle, ne toucher plus au méthode delete des animaux...

Phase 5 - Authentification JWT
Phase 6 - Les middlewares CheckUser
Phase 7 - Swagger OpenAI - Documentation Web api Rest