Objectif de formation :

Développement d'une application Full stack utilisant les technologies MongoDB / Mysql, Node.js, Nest.js et Angular

- Angular
  - Bases (components, directives)
  - Data services
  - Rxjs
- Node.js / Nest.js
  - Models / DB
  - Services
  - Controllers

Application à développer : "ImageBook"

L'objectif de l'application est de collectionner des images et de permettre à tous les utilisateurs de les commenter.

Spécifications de base :

1. Le book organise les images par catégories, chaque image possède donc une catégorie.
2. Chaque image possède également au minimum les attributs suivants :
   1. Un titre
   2. Une url
   3. Une date de réalisation
   4. Une date d'ajout
   5. Une description
   6. Un flag (booléen) indiquant si elle a été supprimée

User Stories

Chacune des stories a pour objectif d'assimiler une partie du framework Angular.

1. *Principes de base des composants :*
   L'utilisateur doit pouvoir lister toutes les images. Cette liste doit au minimum afficher le titre de l'image et la date de réalisation. Elle doit afficher une vignette de l'image.
2. *Directives :*
   Par défaut les images indiquées comme supprimées ne doivent pas apparaitre dans la liste. Un bouton doit permettre à l'utilisateur d'également afficher, dans la liste, les images supprimées.
3. Composant+Directive : l'utilisateur doit pouvoir "consulter" les détails d'une image. A partir de la liste, en sélectionnant une image, le détail de l'image est affiché en grand, à côté de la liste (ou en dessous, la partie css n'est pas importante ici). Dans cette vue en détail, l'image doit être affichée en grand, et tous les attributs doivent être affichés. Si l'on sélectionne une autre image, les informations sont adaptées.
4. Forms :
   Dans la vue détaillée d'une image, un petit formulaire permet à l'utilisateur d'ajouter un commentaire dans l'image. Il faut donc également adapter le modèle de données d'une image pour y conserver plusieurs commentaires. Tous les commentaires sont visibles l'un en dessous de l'autre. En dernier le formulaire permet d'en ajouter un.
5. Forms : validation
   Il faudrait que chaque nouveau commentaire ajouté se termine par une ponctuation de fin de phrase. Le formulaire doit donc contrôler que le contenu du commentaire se termine par l'un des droits caractères suivants : . ? !
   Si le message n'est pas validé, le bouton pour enregistrer le commentaire ne doit pas être actif et un message indiquant qu'il manque une ponctuation en fin de message doit être présent en dessous de l'input du commentaire.

Tâches

1. Initialiser un projet back-end Nest.Js
   1. Base de données MongoDB / MySQL
   2. Modèle "business" pour la resource "Image"
   3. Schéma DB correspondant pour MongoDB / MySQL
   4. CRUD complet pour la resource "Image"
      1. Un service doit s'occuper de toute la logique Business ↔️ DB
      2. Un controller s'occupe des routes CRUD avec des modèles DTO et la communication vers le service
      3. Idéalement implémenter un mapper entre les modèles DB ↔️ model business ↔️ DTO
2. Tester tous les appels CRUD à l'aide du programme Insomnia
3. Initialiser un projet Angular (version > 10)
4. Implémenter les users stories 1 à 1 (voir ci-dessus)

Comment démarrer

- Nest.JS : https://docs.nestjs.com/first-steps
- MongoDB : https://docs.mongodb.com/manual/administration/install-community/

- MySQL (avec Nest.JS et TypeORM) : https://docs.nestjs.com/techniques/database
- Insomnia Core :
  - https://insomnia.rest/products/core/
  - https://support.insomnia.rest/article/11-getting-started
- Angular :
  - Angular : https://angular.io/tutorial/toh-pt0
  - Material : https://material.angular.io/components/categories
  - livre ng-book2 (Dossier partagé "Tous")

Comments