

TP 2 :

Développement d'une application Gestion des tâches (To-Do List) avec MEAN Stack

Objectifs :

1. Comprendre et implémenter les fonctionnalités de base d'une application MEAN.
2. Configurer un backend avec Node.js, Express.js, et MongoDB.
3. Développer un frontend interactif avec Angular pour consommer les APIs du backend.
4. Utiliser des requêtes HTTP (GET, POST, PUT, DELETE) pour communiquer entre le frontend et le backend.

Contexte :

Vous êtes chargé de développer une application web pour gérer une liste de tâches (*To-Do List*) en utilisant le MEAN Stack (MongoDB, Express.js, Angular, Node.js). L'application doit permettre aux utilisateurs d'effectuer les opérations suivantes :

- Ajouter une tâche.
- Consulter la liste des tâches.
- Modifier une tâche (marquer comme terminée ou changer son contenu).
- Supprimer une tâche.

Objectifs pédagogiques :

1. Comprendre le fonctionnement et l'architecture du MEAN Stack.
2. Apprendre à configurer un serveur backend avec Node.js et Express.js.
3. Manipuler MongoDB pour le stockage et la gestion des données.
4. Développer une interface utilisateur réactive avec Angular et consommer les APIs RESTful.
5. Utiliser des requêtes HTTP pour intégrer le frontend et le backend.

Exigences fonctionnelles :

1. **Création de tâches :**
 - Les utilisateurs doivent pouvoir ajouter une tâche avec un titre via un formulaire.
2. **Lecture de la liste des tâches :**
 - Affichez toutes les tâches dans une liste, incluant leur statut (terminée ou non).
3. **Mise à jour des tâches :**
 - Les utilisateurs doivent pouvoir :
 - Modifier le contenu d'une tâche.
 - Marquer une tâche comme terminée ou non terminée.
4. **Suppression de tâches :**
 - Les utilisateurs doivent pouvoir supprimer une tâche.

5. Interface utilisateur :

- L'application doit avoir une interface simple avec une liste de tâches et un formulaire pour ajouter de nouvelles tâches.

Exigences techniques :**1. Backend (Node.js et Express.js) :**

- Créez un serveur RESTful pour gérer les opérations CRUD.
- Utilisez MongoDB pour stocker les données des tâches.
- Implémentez des routes pour les opérations suivantes :
 - GET /tasks : Récupérer toutes les tâches.
 - POST /tasks : Ajouter une nouvelle tâche.
 - PUT /tasks/:id : Mettre à jour une tâche existante.
 - DELETE /tasks/:id : Supprimer une tâche.

2. Frontend (Angular) :

- Développez une interface utilisateur pour consommer les APIs REST.
- Implémentez les fonctionnalités CRUD.
- Utilisez Angular Forms pour gérer les formulaires.

3. Base de données :

- Configurez MongoDB pour stocker les tâches avec les champs suivants :
 - title : le titre de la tâche (texte).
 - completed : statut de la tâche (booléen, par défaut false).

Livrables :

1. Le code complet de l'application :
 - Dossier backend contenant le serveur Node.js et les modèles MongoDB.
 - Dossier frontend contenant le code Angular.
2. Documentation simple incluant :
 - Instructions pour lancer le backend et le frontend.
 - Explications des principales fonctionnalités implémentées.

Contraintes :

1. Le frontend et le backend doivent communiquer via des APIs REST.
2. Les données doivent être stockées dans MongoDB.
3. Le code doit être bien structuré et commenté.

Étapes suggérées :

1. **Mise en place de l'environnement de travail :**
 - Installez Node.js, Angular CLI et MongoDB.
 - Initialisez les projets backend et frontend.
2. **Développement du backend :**
 - Configurez le serveur Express.js.
 - Implémentez les routes CRUD.
 - Connectez le backend à MongoDB.
3. **Développement du frontend :**
 - Créez une interface avec Angular.
 - Implémentez un service pour consommer les APIs REST.

- Ajoutez un composant pour afficher et gérer les tâches.
4. **Test de l'application :**
- Vérifiez que toutes les fonctionnalités sont opérationnelles.
 - Corrigez les éventuels bugs.

Critères d'évaluation :

1. Respect des exigences fonctionnelles et techniques.
2. Qualité du code (structuration, lisibilité, commentaires).
3. Fonctionnement correct des APIs et de l'interface utilisateur.
4. Documentation claire et instructions d'utilisation.

Solution attendue de l'application de gestion de tâche (voir figure ci-dessous) :