Uzdevums: To-Do list

Uzdevuma apraksts:

Jūsu uzdevums ir izveidot vienkāršu **to-do list** mājaslapu, kas ļauj lietotājiem pievienot, rediģēt un dzēst uzdevumus. Mājaslapa būs jāizveido, izmantojot **Node.js** kā servera pusi un **MongoDB** kā datubāzi. Uzdevums palīdzēs apgūt darbību ar serveriem, datubāzēm un HTTP pieprasījumiem.

Uzdevuma prasības:

1. **Node.js serveris**:
   - Izveidot Node.js serveri, kas darbojas uz Express.js ietvara.
   - Iestatīt nepieciešamās bibliotēkas, piemēram, `express`, `mongoose` (MongoDB izmantošanai), `body-parser` (lūgumu ķeršanai).
2. **MongoDB datubāze**:
   - Izveidot MongoDB datubāzi ar kolekciju, kurā glabāsies uzdevumi. Katram uzdevumam jābūt šādiem laukiem:
     - `title` (uzdevuma nosaukums)
     - `description` (uzdevuma apraksts)
     - `status` (uzdevuma status, piemēram, "pabeigts" vai "nepabeigts")
     - `createdAt` (laiks, kad uzdevums tika pievienots)
     - `updatedAt` (laiks, kad uzdevums tika pēdējo reizi atjaunināts)
3. **API izveide**:
   - **GET /tasks** – atgriež visus uzdevumus no datubāzes.
   - **POST /tasks** – pievieno jaunu uzdevumu datubāzē.
   - **PUT /tasks/:id** – atjaunina esošu uzdevumu (piemēram, izmaina statusu vai aprakstu).
   - **DELETE /tasks/:id** – dzēš uzdevumu no datubāzes.
4. **Mājaslapas izveide (front-end)**:
   - Izveidot vienkāršu HTML/CSS front-endu, kas attēlo uzdevumu sarakstu.
   - Nodrošināt funkcionalitāti pievienot jaunu uzdevumu, mainīt tā statusu un dzēst uzdevumus, izmantojot API pieprasījumus (izmantojot `fetch` vai `axios`).
   - Lietotājs varēs redzēt visus uzdevumus, kā arī pievienot jaunus un rediģēt esošos.
5. **Papildus uzdevums (neobligāti)**:
   - Ievietot autentifikāciju, lai katrs lietotājs varētu piekļūt tikai saviem uzdevumiem.
   - Ievietot validāciju uzdevumu datiem gan serverī, gan front-endā.

Ieteicamās darbības:

1. **Node.js instalēšana un konfigurēšana**:
   - Instalējiet un inicializējiet Node.js projektu, izmantojot `npm init`.
   - Instalējiet nepieciešamās bibliotēkas (`express`, `mongoose`, `body-parser`, u.c.).
2. **MongoDB konfigurācija**:
   - Pieslēdzieties MongoDB datubāzei, izmantojot `mongoose`.
   - Izveidojiet uzdevumu modeli un veiciet datubāzes operācijas (CRUD).

3. **API maršrutu izveide**:
    - Izveidojiet maršrutus (`GET`, `POST`, `PUT`, `DELETE`), lai veiktu uzdevumu pārvaldību.
4. **Front-end izveide**:
    - Izveidojiet vienkāršu HTML/CSS lapu, kas attēlo uzdevumu sarakstu un ļauj mijiedarboties ar API.
5. **Testēšana**:
    - Testējiet API pieprasījumus, lai nodrošinātu, ka dati tiek pareizi ievadīti, atjaunināti un dzēsti.