Documentație API Todo

API-ul Todo oferă funcționalități pentru crearea, preluarea și gestionarea sarcinilor (todo-urilor). Este construit folosind Node.js și Express și include autentificare bazată pe JSON Web Tokens (JWT) pentru acces securizat.

Am folosit Postman pentru testarea API-ului.

Rularea aplicației:

1. Clonați repository-ul GitHub în directorul local dorit
2. Deschideți un terminal în directorul proiectului
3. Rulați `npm install` pentru a instala dependențele necesare
4. Rulați aplicația folosind comanda `node app.js`. Serverul va fi pe portul 3000 sau pe portul definit în variabila de mediu `PORT`

Endpoints:

Înregistrare utilizatori:

POST `/register`

Body:

```
{
    "username": "numeUtilizator",
    "password": "parolaSecreta"
}
```

Răspunsuri posibile:

- 201 Created: `User created`
- 500 Internal Server Error

Autentificare utilizatori:

POST `/login`

Body:

```
{
    "username": "numeUtilizator",
    "password": "parolaSecreta"
}
```

Răspunsuri posibile:

200 OK: `{ "accessToken": "<token_jwt>" }`

400 Bad Request: `Cannot find user`

500 Internal Server Error

Creare todo:

POST `/todos`(necesită autentificare)

Body:

```
{
    "title": "Titlu todo",
    "completed": false
}
```

Răspunsuri posibile:

201 Created: `<obj_todo>`

400 Bad Request

401 Unauthorized

Preluare toate todo-urile:

GET `/todos`(necesită autentificare)

Răspunsuri posibile:

200 OK: Array de todo-uri

401 Unauthorized

Preluare todo după ID:

GET `/todos/:id`(necesită autentificare)

Răspunsuri posibile:

200 OK: `<obj_todo>`

404 Not Found

401 Unauthorized

Explicație stări HTTP de răspuns:

200 OK: Cererea a reușit.

201 Created: O resursă nouă (de ex., todo) a fost creată.

400 Bad Request: Cerere invalidă din cauza unui client error.

401 Unauthorized: Cererea necesită autentificare (token JWT necesar).

403 Forbidden: Utilizatorul nu are drepturi pentru resursa cerută.

404 Not Found: Resursa cerută nu a fost găsită.

500 Internal Server Error: Eroare generică de server.

Contextul API-ului de todo-uri:

Înregistrarea și Autentificarea Utilizatorilor: Succesul se traduce prin stările 201 (utilizator creat) și 200 (autentificare reușită, token JWT oferit). Erorile în procesul de autentificare sau cereri invalide rezultă în stările 400, 401 sau 500.

Gestionarea Todo-urilor: Crearea, preluarea sau actualizarea todo-urilor necesită un token JWT valid (401 pentru lipsă sau invalid). Crearea reușită returnează 201. Erorile de validare sau problemele de server duc la stările 400 sau 500. Încercarea de a accesa un todo inexistent întoarce 404.

Testarea aplicației:

După ce am deschis consola in VS Code, am introdus comanda `node app.js` și am primit mesajul Server is running on port 3000, ceea ce inseamnă că totul este în regulă și pot începe testarea.

În Postman, am creat o nouă colecție numită Test, unde am creat un POST Register, POST Login, GET All Todos, POST Add a new Todo, GET Get Todo by id.

În **POST Register**(http://localhost:3000/register), în Body, după ce am ales `raw`, am selectat formatul de date JSON și am introdus:

```
{
    "username": "beni",
    "password": "1234"
}
```

Când am dat Send, am primit mesajul User created(201 Created)

In **POST Login**(http://localhost:3000/login) am introdus in Body:

```
{
    "username": "beni",
```

"password": "1234"

}

Apoi mi-a fost returnat Token-ul pe care trebuie să îl copiez de acum încolo, pentru a putea să mă conectez.

În **GET All Todos**, am accesat Authorization, am selectat Bearer Token, și am introdus în secțiunea destinată Token-ul. Dacă nu fac asta, primesc mesajul Unauthorized.

Totuși dacă Token-ul este introdus corect, mi se va returna un vector gol atunci cand apas Send.

În **POST Add a new Todo**(http://localhost:3000/todos) introduc același Token în aceeași secțiune destinată pentru a putea continua cu adăugarea de Todo-uri. În Body, am selectat `raw` și am introdus

{

"title": "sport",

"completed": true

}

True trebuie introdus fără "", pentru a reprezenta un tip de date boolean.

Adăugăm și Token-ul și apasăm Send. Vom primi mesajul:

{

"id": 1,

"title": "sport",

"completed": true

}

Observăm că totul funcționează perfect, id-ul fiind setat la 1, așa cum trebuie. Ca să terminam cu adăugarea de todo-uri, mai introducem încă un todo în aceeași manieră. Editățm Body-ul, introducând:

{

"title": "citit",

"completed": false

}

Și primim:

{

"id": 2,

"title": "citit",

"completed": false

}

Ceea ce ne arată că totul este așa cum ne așteptam.

În **GET Get Todo by id**, putem adăuga /1 sau /2 la sfârșitul adresei: http://localhost:3000/todos în cazul nostru pentru a returna sarcina cu id-ul respectiv. Token-ul trebuie introdus ca până acum.

Dacă am urmat pașii corect până aici, ar trebui să obținem:

{

"id": 1,

"title": "sport",

"completed": true

}

Pentru id-ul 1, și:

{

"id": 2,

"title": "citit",

"completed": false

}

Pentru id-ul 2

Putem reveni la **GET Get All Todos** pentru a afișa toate todo-urile, deoarece acum avem 2 todo-uri înregistrate și le putem afișa pe ambele deodată.

Totul a mers perfect și aplicația noastră îndeplinește toate cerințele problemei.