TodoAPIDocumentation

This is an Api to help you keep track of tasks that need to be done. In this Api, you can save the taskt to the database, view you tasks, update them, and delete them when done. The Api requires you to create a user by signing up, them logging in to acces the todo functionality

Authenticaiton

These request are used to signup to the Api, and logging in so that the functionality becomes available.

POST Sign up new user

http://localhost:3000/signup

The signup request takes three parameters in the body of the request:

- Name
- Email
- Password

[illegible]

Body raw (json)

json

```json
{
    "name": "example",
    "email": "example@gmail.com",
    "password": "examplePassword"
}
```

POST Login user to acces functionality

http://localhost:3000/login

The login requrest takes two parameters in the request body:

- Email
- Password

This requrest will return a token in the response body. Copy this token and place it as a Bearer token in the authorization tab to use the todo-functionality.

Below are a successfull and a negative example for this request.

Body raw (json)

json

```json
{
    "email": "example@gmail.com",
    "password": "examplePassword"
}
```

Basic Todo Operations

Here are examples for the todo-request.

To access all these, take the token from the login request, and paste it as a bearer token in the autorizations tab.

GET Get all todo items

http://localhost:3000/todo

This Get request return a list of all items that have been logged in

This GET request return a list of all items that have been logged in the database. If there are no items available, the request is still successfull, but message will prompt you that there are no items to view.

Below are a successfull and a negative example for this request.

AUTHORIZATION Bearer Token

Token	<token>

POST Create new todo item

http://localhost:3000/todo

This POST request creates a new item and saves it to tha database. The requrest takes two parameters in the requrest body:

- Name - string
- Category - integer

Below are a successfull and a negative example for this request.

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

json

```json
{
    "name": "Dusting",
    "category": 1
}
```

PUT Update todo item by name

http://localhost:3000/todo

This PUT request updates one item that is already in the database. This request takes two parameters in the request body.

- oldName - Name of item you wish to update
- newName - New name for the updated item.

Below are a successfull and a negative example for this request.

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

json

```json
{
    "oldName": "Dusting",
    "newName": "Cleaning"
}
```

DELETE Delete todo item

http://localhost:3000/todo

This DELETE request deletes one item. The requrest takes a name of the item you wish to delete as parameter in the request body.

- Name

Below are a successfull and a negative example for this request.

AUTHORIZATION Bearer Token

Token	<token>

Body raw (json)

json

```json
{
    "name": "Cleaning"
}
```