

## Lab3 - Express server

### GREENS ARE BONUS

Implement a todo app where users can login and do simple CRUD operations on their todos.

- A Todo must have title, status, tags (optional) and the creation and update time.
- A User must have username, password, firstName and date of birth (optional).
- A User can sign up, login and add, edit, view or delete his own todos.
- Todo has the status of 'new' by default and users can update into 'inProgress' or 'done'.

1- Create two models (user, todo)

User : {

username: String, required, unique, min 8

~~password: String, required,~~

firstName: String, required, min length 3, max length 15

lastName: String, required, min length 3, max length 15

dob: Date, optional

}

Todo {

userId: the ObjectId of the user,

title: String, required, min 5, max 20,

status: String, optional, default is "to-do" ['to-do', 'in progress', 'done']

tags:[String], optional, max length for each tag is 10

**createdAt: Date, timeStamp,**

**updatedAt: Date, timeStamp**

}

**Use autoIncremental id instead of mongo id**

3 - Implement the following end points.

HTTP Method	route	Description
post	/users	- Register a user with the following <b>required</b> attributes Username,password , firstName, lastName Notes: <del>Return registered user with token if success</del> - Handle validation errors returned from mongo
Post	/users/login	<del>Return (user with token )-)</del> Don't return password  <del>If the the authentication failed</del>

		<del>Return error with 401 status code</del>
GET	/users	Return the first name of registered users
DELETE	/users/:id	Delete the user with selected id
PATCH	/users/:id	- Edit the user with the selected id - Return ({message:"user was edited successfully", user: theUserAfterEdit}) if success - Handle validation errors returned from mongo
POST	/toods	Create new todo ({title,tags}) You must save it with userId of the logged in user Return the new todo to the user
PATCH	/todos/:id	Edit todo
DELETE	/todos/:id	Delete todo
GET	/todos/:userId	Return the todos of specific user
GET	/todos?limit=10&skip=0	Return the posts with specific required filters (defaults are limit 10 skip 0 )

~~4 Protect all endpoints with a proper authentication layer. (except for registration and login of course!)~~

## READ THE DOCUMENTATION

<https://mongoosejs.com/docs/>

## Useful reads:

[Handle errors in express](#) **(IMPORTANT)**

[How the Web works](#)

[CORS](#)

-----  
[How bcryptjs works](#)

[How to Deploy a MERN Application to Heroku Using MongoDB Atlas](#)  
[promisify](#)