Securing a REST API

Requirements :

First, ensure you have Node.js and npm (Node Package Manager) installed on your machine. You can download and install them from the official Node.js website.

First step :

Create a new directory for your project and navigate into it using your command line interface. The most frequent one are cd allowing to move with a path that you give.

```
cd C:\Users\dylan\OneDrive\Bureau\erasmus\SOA-master\SOA-master
```

This is what the command look like,

```
PS C:\Users\dylan\OneDrive\Bureau\Nouveau dossier\SOA_Node_Dylan>
```

and this what you should see if it works.

Set up :

You can now create a file named main-tutorial.js or simply paste the document provided, now be sure your shell is in the good directory and use the following command to download express and jsonwebtoken.

```
npm install express jsonwebtoken
```

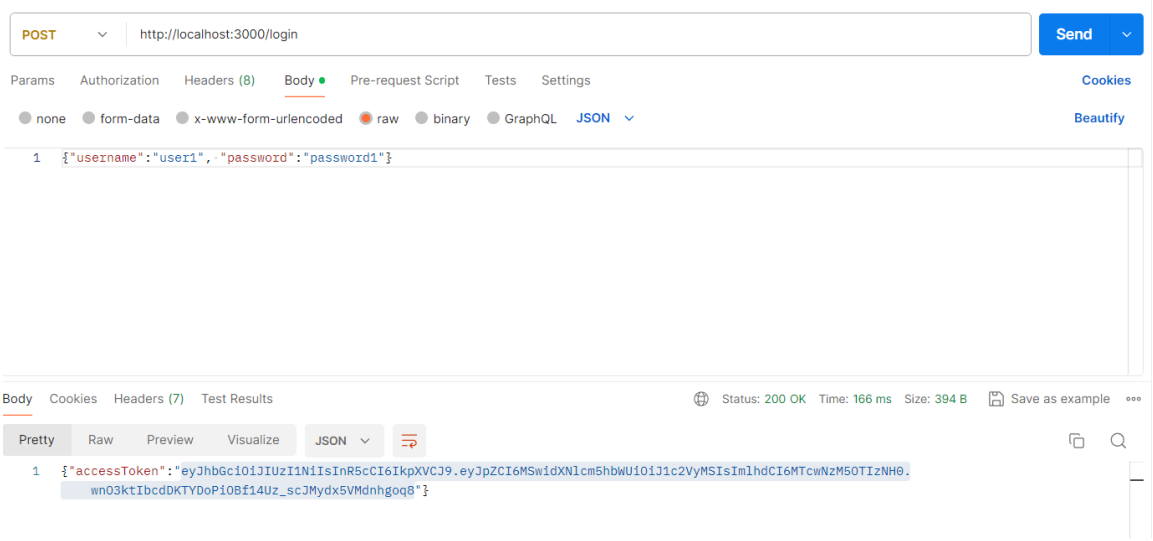
If you have error in your code install any missing library could have miss and verify your path.

Testing :

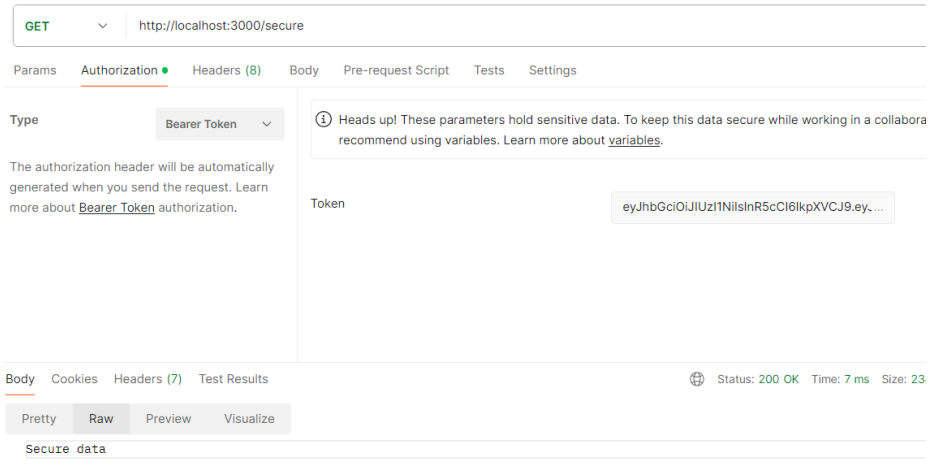
You can now test your API using tools like Postman or by sending HTTP requests programmatically from your frontend application, we recommend the use of postman as he is easier to visualize and modify header than other solution that’s why we’re gonna use it.

Here's how you can test the endpoints:

Send a POST request to http://localhost:3000/login with JSON body containing username and password. To do it with postman create a new HTTP request and put your localhost address which is http://localhost:3000/login then go to body and select raw for the data type, choose json for the language. You can now send it and get back your token at the end of your window.



We will now access the data create a new HTTP request and put your localhost but this time it will be http://localhost:3000/secure, then go to authorization and select bearer token in type, paste the token you get to the previous step and at the bottom of your window under raw you should see ‘secure data’



that the answer to this line confirming the token is valid

```
app.get('/secure', authenticateToken, (req, res) => {
  res.send('Secure data');
});
```

Conclusion :

Congratulations! You have successfully created a secure REST API you now need to make a web application for people to enter their log, use a database to store their log, make additional security and find something to do with your server.