



Département de Génie Informatique et Communication
Department of Information and Communication Engineering

Topic : TP09

Subject : Internet Programming

Teacher : Mr. Hok Tin

Name : YORNG TONGHY

ID : e20191313

Group : I4-GIC-C

Year : 2022-2023

Table of Contents

GitHub Link:	3
Exercise 1:	3
-Result of Exercise1:.....	3
- Connect to Mongo DB.....	3
-Find specific username	4
-List data in mongoDB.....	4
TP09.2: Nodejs (Authentication Implementation)	5
Result of Ex2:.....	5
Encrypted user accounts	5
-Request valid Email	5
-Request valid Password	5
TP09.3: NodeJS (Token, Cookie)	6
Result of Ex3:.....	6
- Register User Account.....	6
-Register user account if user already existed	7
- login to user account just input _id:	7
- login already signed in	8
-Logout	9
-Re – logout	9

GitHub Link:

https://github.com/yorng-tonghy/TP08_IP-TP09_IP/tree/TP09

Exercise 1:

TP9.1: Node JS (Mongo Database)

Ex1: For the previous authentication APIs, replace your user.json storage by a Nosql database, Mongoose

-Result of Exercise1:

- Connect to Mongo DB

The screenshot shows the Robo 3T MongoDB interface. On the left, the sidebar lists 'Open Connections' (localhost_1) and 'Samples' (es6+ samples for beginners, MongoDB Basic CRUD Operations, MongoDB Basic Query Operators, NoSQLBooster Tutorial). The main area shows a connection to 'localhost_1' with a database 'yorngtonghy'. A query window displays the following code:

```
1 db.user.find({})
2   .projection({})
3   .sort({_id:-1})
4   .limit(100)
```

The results table shows the 'user' collection with 7 documents. One document is selected, showing its details:

Key	Value	Type
updatedAt	5/28/2023, 7:38:27 AM - 3 days ago	Date
__v	0	Int32
(2) 6471926a017aabba63ed02e1a7cb	{ email : "chanvotey@gmail.com" } (9 fields)	Document
(3) 64719061ace404c0a8b7ce6	{ email : "SokVately@gmail.com" } (9 fields)	Document
(4) 6471926a5819ba802ce92d8d	{ email : "channararoth@gmail.com" } (9 fields)	Document
(5) 64719ddc5819ba802ce92d58	{ email : "Guest@gmail.com" } (9 fields)	Document
(6) 64718aa25819ba802ce92d3b	{ email : "admin@gmail.com" } (9 fields)	Document
(7) 647171a5819ba802ce92cf9	{ email : "yorngtonghy@gmail.com" } (9 fields)	Document
_id	647171a5819ba802ce92cf9	ObjectID
username	YORNGTONGHY12345	String
firstname	YORNG	String
lastname	TONGHY	String
email	yorngtonghy@gmail.com	String
password	\$2a\$10\$jmWklZf2qyCATwmU2VGu93c1Dvb/BuWiqvYybc0LxPjU2Coy.Fy	String
createdAt	5/27/2023, 10:55:06 AM - 4 days ago	Date
updatedAt	5/27/2023, 10:55:06 AM - 4 days ago	Date
__v	0	Int32

-Find specific username

The screenshot shows the NoSQLBooster IDE interface. In the top navigation bar, the connection is set to "yorngtonghy@localhost_2". The main area displays a MongoDB query:

```

1 db.user.find({username:"YORNGTONGHY12345"})
2   .projection({})
3   .sort({_id:-1})
4   .limit(100)

```

Below the query, the results are shown in a table:

Key	Value	Type
<code>(1) 64717f1a5819ba802ce92cf9</code>	{ email : "yorngtonghy@gmail.com" } (9 fields)	Document
<code>_id</code>	64717f1a5819ba802ce92cf9	ObjectID
<code>username</code>	YORNGTONGHY12345	String
<code>firstname</code>	YORNG	String
<code>lastname</code>	TONGHY	String
<code>email</code>	yorngtonghy@gmail.com	String
<code>password</code>	\$2a\$10\$jMw/kZf2qyCAtwmU2VGu93c1Dvb/BuWiqvYybc0LxPjU2Coy.Fy	String
<code>createdAt</code>	5/27/2023, 10:55:06 AM - 4 days ago	Date
<code>updatedAt</code>	5/27/2023, 10:55:06 AM - 4 days ago	Date
<code>_v</code>	0	Int32

-List data in mongoDB

The screenshot shows the NoSQLBooster IDE interface. In the left sidebar, the connection is set to "localhost_1" and the database is "user". The main area displays a MongoDB query:

```

1 db.user.find({})
2   .projection({})
3   .sort({_id:-1})
4   .limit(100)

```

Below the query, the results are shown in a table:

Key	Value	Type
<code>updatedAt</code>	5/28/2023, 7:38:27 AM - 3 days ago	Date
<code>_v</code>	0	Int32
<code>(2) 6472a017aabba63e02e1a7cb</code>	{ email : "chanvotey@gmail.com" } (9 fields)	Document
<code>(3) 6471c90e1ace404cd0a8b7ce6</code>	{ email : "SokVatey@gmail.com" } (9 fields)	Document
<code>(4) 6471926a5819ba802ce92d8d</code>	{ email : "channaracoth@gmail.com" } (9 fields)	Document
<code>(5) 64718da5819ba802ce92d58</code>	{ email : "Guest@gmail.com" } (9 fields)	Document
<code>(6) 64718a25819ba802ce92d3b</code>	{ email : "admin@gmail.com" } (9 fields)	Document
<code>(7) 64717f1a5819ba802ce92cf9</code>	{ email : "yorngtonghy@gmail.com" } (9 fields)	Document
<code>_id</code>	64717f1a5819ba802ce92cf9	ObjectID
<code>username</code>	YORNGTONGHY12345	String
<code>firstname</code>	YORNG	String
<code>lastname</code>	TONGHY	String
<code>email</code>	yorngtonghy@gmail.com	String
<code>password</code>	\$2a\$10\$jMw/kZf2qyCAtwmU2VGu93c1Dvb/BuWiqvYybc0LxPjU2Coy.Fy	String
<code>createdAt</code>	5/27/2023, 10:55:06 AM - 4 days ago	Date
<code>updatedAt</code>	5/27/2023, 10:55:06 AM - 4 days ago	Date
<code>_v</code>	0	Int32

TP09.2: Nodejs (Authentication Implementation)

Ex2: Implement the authentication with the password encryption and request validation middleware

Result of Ex2:

Encrypted user accounts

Key	Value	Type
64717f1a5819ba802ce92cf9	{ email : "yorngtonghy@gmail.com" } (9 fields)	Document
_id	64717f1a5819ba802ce92cf9	ObjectId
username	YORNGTONGHY12345	String
firstname	YORNG	String
lastname	TONGHY	String
email	yorngtonghy@gmail.com	String
password	\$2a\$10\$jMw/ktZf2qyCAtwmU2VGu83c1Dvb/BuWiqvYybc0LxPjU2Coy.Fy	String
createdAt	5/27/2023, 10:55:06 AM - 4 days ago	Date
updatedAt	5/27/2023, 10:55:06 AM - 4 days ago	Date
v	0	Int32

-Request valid Email

```

POST http://localhost:3001/login
Body:
{
  "email": "tonghy123@.com",
  "password": "123456"
}

Body:
{
  "success": false,
  "error": {
    "_original": {
      "email": "tonghy123@.com",
      "password": "123456"
    },
    "details": [
      {
        "message": "\"email\" must be a valid email",
        "path": [
          "email"
        ],
        "type": "string.email",
        "context": {
          "value": "tonghy123@.com",
          "invalids": [
            "tonghy123@.com"
          ],
          "label": "email"
        }
      }
    ]
  }
}
  
```

-Request valid Password

```

1
2 "email": "tonghy123@gmail.com"
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
  
```

Status: 200 OK Time: 7 ms Size: 509 B Save as Example

```

1
2 "success": false,
3 "error": {
4     "_original": {
5         "email": "tonghy123@gmail.com"
6     },
7     "details": [
8         {
9             "message": "\"password\" is required",
10            "path": [
11                "password"
12            ],
13            "type": "any.required",
14            "context": {
15                "label": "password",
16                "key": "password"
17            }
18        }
19    ]
20 }
  
```

TP09.3: NodeJS (Token, Cookie)

Ex3: Create a token of an authenticated user and store it in the cookie for ensuring the authorized user. The token validation method must be a middleware function.

Result of Ex3:

- Register User Account

```

1
2 ...
3 ...
4 ...
5 ...
6 ...
7 ...
8 ...
9 ...
10 ...
  
```

Status: 200 OK Time: 194 ms Size: 699 B Save as Example

```

1
2 ...
3 ...
4 ...
5 ...
6 ...
7 ...
8 ...
9 ...
10 ...
  
```

```

1
2 ...
3 ...
4 ...
5 ...
6 ...
7 ...
8 ...
9 ...
10 ...
  
```

-Register user account if user already existed

POST http://localhost:3001/register

Body (JSON)

```

1
2   "email": "tonghy999@gmail.com",
3   "username": "YorngTonghy",
4   "firstname": "Tonghy",
5   "lastname": "Tonghy",
6   "password": "123",
7   "repeat_password": "123"
8
9
10

```

Status: 200 OK Time: 31 ms Size: 542 B Save as Example

Pretty Raw Preview Visualize JSON

```

1
2   "success": false,
3   "error": "User is already existed~"
4

```

- login to user account just input _id:

POST http://localhost:3001/register

Body (JSON)

```

1
2   "email": "tonghy999@gmail.com",
3   "username": "YorngTonghy",
4   "firstname": "Tonghy",
5   "lastname": "Tonghy",
6   "password": "123",
7   "repeat_password": "123"
8
9
10

```

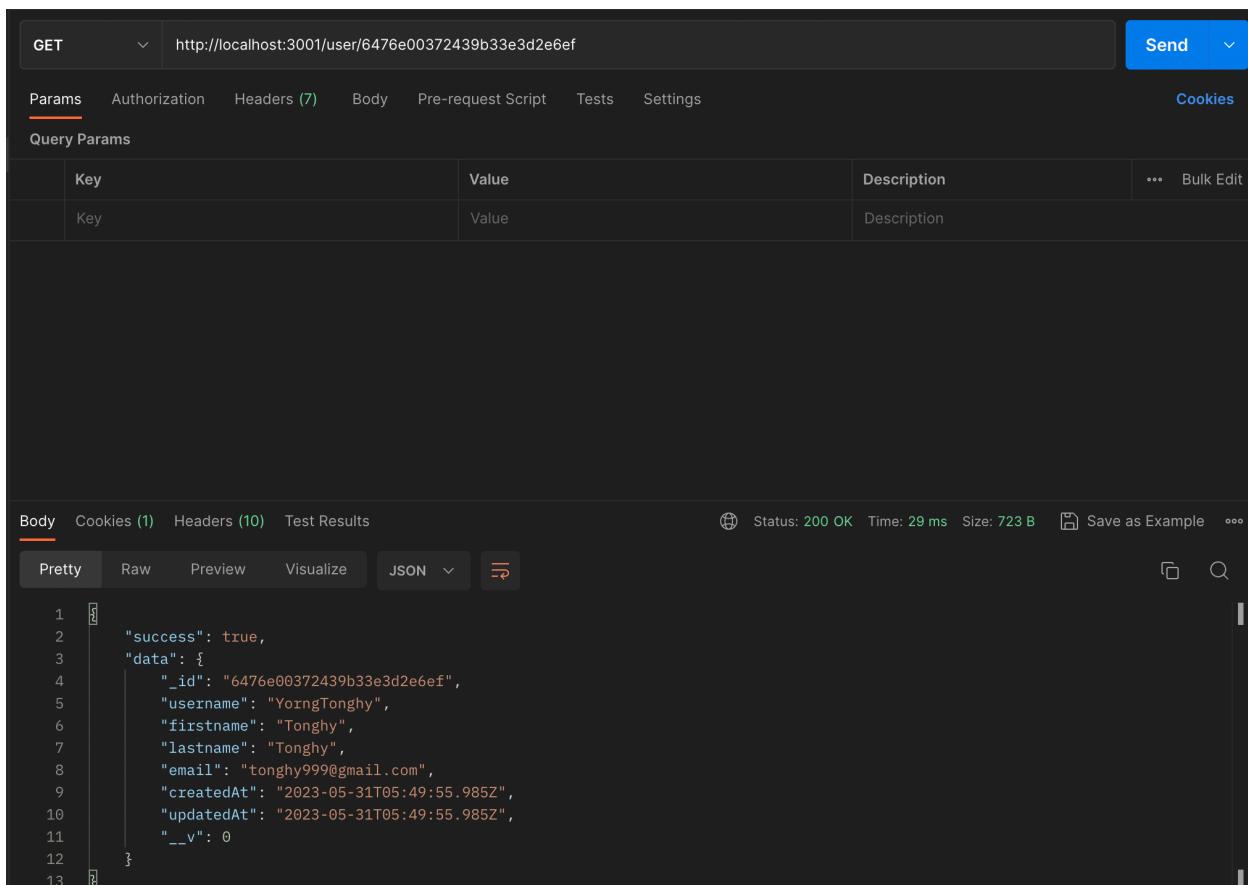
Status: 200 OK Time: 31 ms Size: 542 B Save as Example

Pretty Raw Preview Visualize JSON

```

1
2   "success": false,
3   "error": "User is already existed~"
4

```



GET http://localhost:3001/user/6476e00372439b33e3d2e6ef

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description	...	

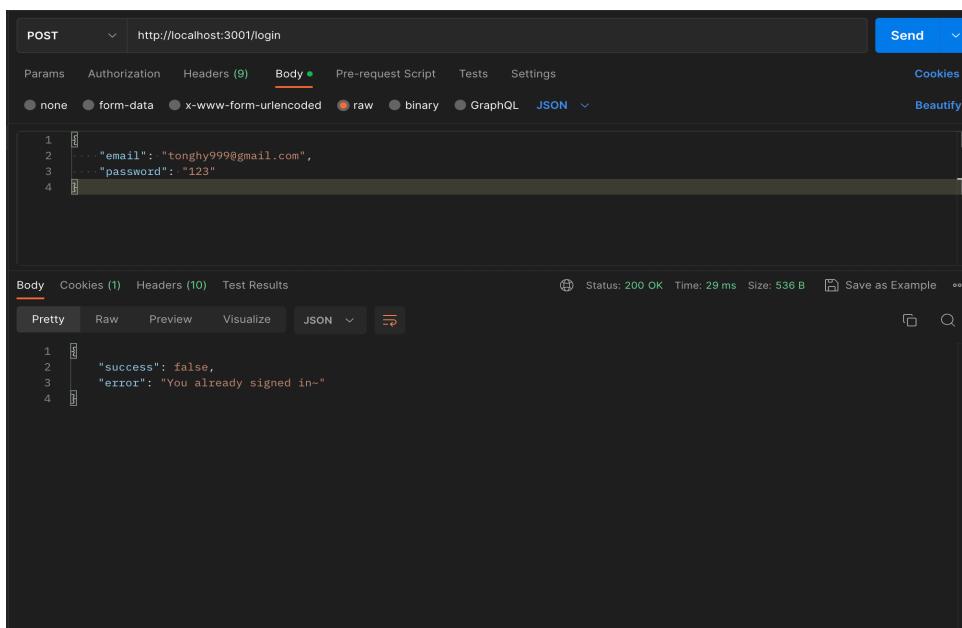
Body Cookies (1) Headers (10) Test Results Status: 200 OK Time: 29 ms Size: 723 B Save as Example

```

1 "success": true,
2 "data": {
3     "_id": "6476e00372439b33e3d2e6ef",
4     "username": "YorngTonghy",
5     "firstname": "Tonghy",
6     "lastname": "Tonghy",
7     "email": "tonghy999@gmail.com",
8     "createdAt": "2023-05-31T05:49:55.985Z",
9     "updatedAt": "2023-05-31T05:49:55.985Z",
10    "__v": 0
11 }
12
13

```

- login already signed in



POST http://localhost:3001/login

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON

Body Cookies (1) Headers (10) Test Results Status: 200 OK Time: 29 ms Size: 536 B Save as Example

```

1 "email": "tonghy999@gmail.com",
2 ...
3 ...
4

```

Body Cookies (1) Headers (10) Test Results Status: 200 OK Time: 29 ms Size: 536 B Save as Example

```

1 "success": false,
2 "error": "You already signed in~"
3
4

```

-Logout

The screenshot shows a POST request to `http://localhost:3001/logout`. The Body tab is selected, showing a JSON payload with `"email": "tonghy999@gmail.com"` and `"password": "123"`. The response status is `200 OK`, time is `24 ms`, size is `317 B`. The response body is `{"success": true}`.

```

POST http://localhost:3001/logout
Body
1 {"email": "tonghy999@gmail.com",
2     "password": "123"
3 }
4
Pretty Raw Preview Visualize JSON
1 {"success": true}
2
3

```

-Re – logout

The screenshot shows a POST request to `http://localhost:3001/logout`. The Body tab is selected, showing a JSON payload with `"email": "tonghy999@gmail.com"` and `"password": "123"`. The response status is `200 OK`, time is `52 ms`, size is `529 B`. The response body is `{"success": false, "error": "You must sign In~"}`.

```

POST http://localhost:3001/logout
Body
1 {"email": "tonghy999@gmail.com",
2     "password": "123"
3 }
4
Pretty Raw Preview Visualize JSON
1 {"success": false,
2     "error": "You must sign In~"
3 }
4

```