

# Postman Results:

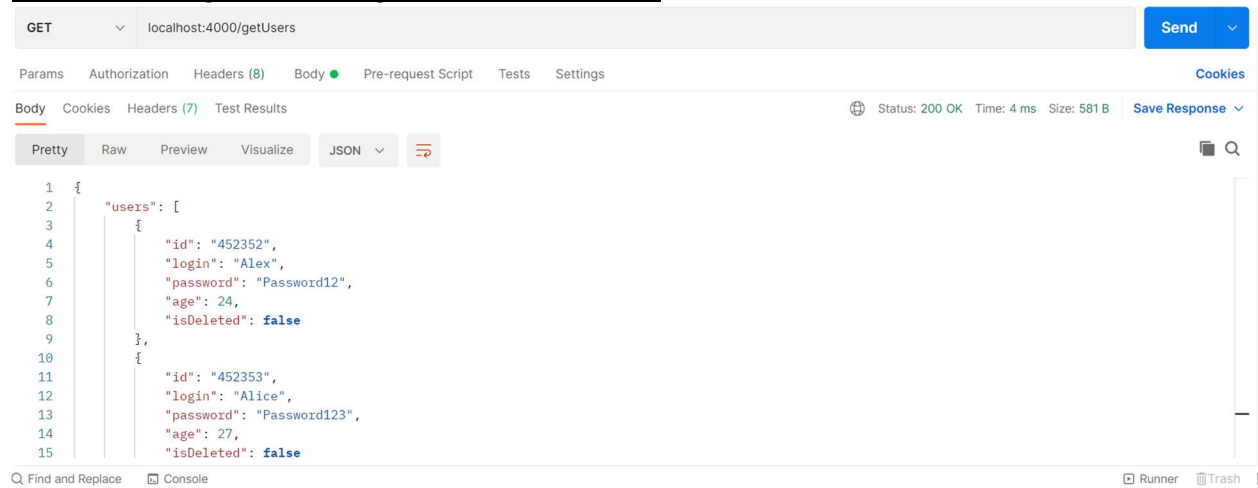
## Vscode terminal output when we start the server

```
PS C:\Users\Md_Liyaqath_Ali\Desktop\NodeTask2> npm start

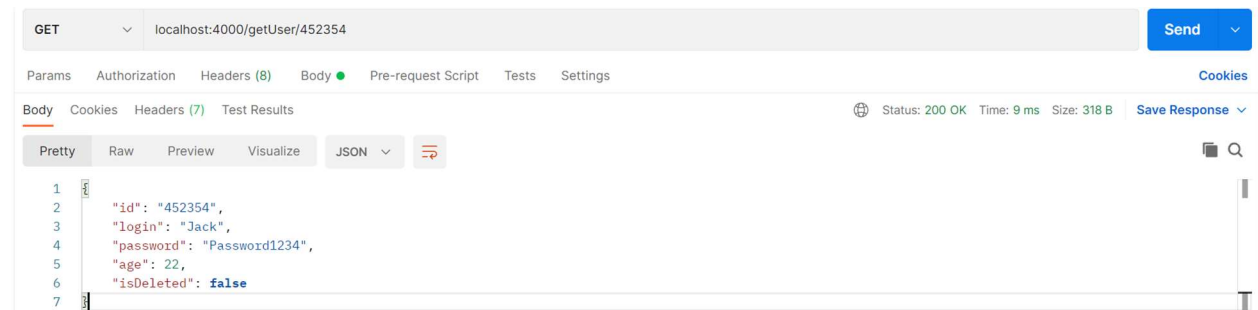
> express-starter@1.0.0 start
> SET PORT=4000 && nodemon index.js

[nodemon] 2.0.16
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
The server is running on port 4000 To access it visit Postman
localhost:4000 /getUsers - To get all the Users
localhost:4000 /getUser/id - To get User with specific id
localhost:4000 /createUser - To create a new User
localhost:4000 /deleteUser/id - To delete a User with specific id
localhost:4000 /updateUser/id - To update User with specific id
localhost:4000 /getAutoSuggestUsers/loginSubstring/limit - To get an AutoSuggest list in a sorted order with specific limit and substring
```

## localhost:4000/getUsers – To get all the user records:



## localhost:4000/getUser/id – To get a particular user with specific id:



# Postman Results:

## localhost:4000/createUser – To create a User (Success case Passed Joi validation):

POST localhost:4000/createUser

Params Authorization Headers (8) Body Pre-request Script Tests Settings

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

```
1 {
2   "id": "452365",
3   "login": "John",
4   "password": "Password9863",
5   "age": 65,
6   "isDeleted": false
7 }
```

Body Cookies Headers (7) Test Results

Status: 200 OK Time: 11 ms Size: 275 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Successfully created a user"
3 }
```

## Checking the User is created or not – Its created successfully

GET localhost:4000/getUser/452365

Params Authorization Headers (8) Body Pre-request Script Tests Settings

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

Body Cookies Headers (7) Test Results

Status: 200 OK Time: 8 ms Size: 318 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": "452365",
3   "login": "John",
4   "password": "Password9863",
5   "age": 65,
6   "isDeleted": false
7 }
```

## localhost:4000/createUser – (Failed case age less than 4 Joi validation case)

POST localhost:4000/createUser

Params Authorization Headers (8) Body Pre-request Script Tests Settings

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

```
1 {
2   "id": "4523100",
3   "login": "Adam",
4   "password": "Password2343",
5   "age": 2,
6   "isDeleted": false
7 }
```

Body Cookies Headers (7) Test Results

Status: 404 Not Found Time: 8 ms Size: 411 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "errorMessage": [
3     {
4       "message": "\"age\" must be greater than or equal to 4",
5       "path": [
6         "age"
7       ],
8       "type": "number.min",
9       "context": {
10        "limit": 4,
11        "value": 2,
12        "label": "age",
13        "key": "age"
14      }
15    }
16  ]
17 }
```

# Postman Results:

## localhost:4000/createUser– (Failed case Missed the Required field “login” Joi validation case):

This screenshot shows a POST request to `localhost:4000/createUser` in Postman. The request body is a JSON object with the following fields: `id` (4523100), `password` (Password2343), `age` (2), and `isDeleted` (false). The response status is 404 Not Found, with a time of 9 ms and a size of 378 B. The response body is a JSON object with an `errorMessage` array containing a single object: `{ "message": "\"login\" is required", "path": [ "login" ], "type": "any.required", "context": { "label": "login", "key": "login" } }`.

```
1 {
2   "id": "4523100",
3   "password": "Password2343",
4   "age": 2,
5   "isDeleted": false
6 }
7
```

Status: 404 Not Found Time: 9 ms Size: 378 B Save Response

```
1 {
2   "errorMessage": [
3     {
4       "message": "\"login\" is required",
5       "path": [
6         "login"
7       ],
8       "type": "any.required",
9       "context": {
10        "label": "login",
11        "key": "login"
12      }
13     }
14   ]
15 }
```

## localhost:4000/updateUser/id – To update a particular user with id (Success case Joi validation):

This screenshot shows a PUT request to `localhost:4000/updateUser/452354` in Postman. The request body is a JSON object with the following fields: `id` (452354), `login` (Jack), `password` (Password1234), `age` (54), and `isDeleted` (false). The response status is 200 OK, with a time of 16 ms and a size of 281 B. The response body is a JSON object with a `message` field: `"Successfully Updated User's data"`.

```
1 {
2   "id": "452354",
3   "login": "Jack",
4   "password": "Password1234",
5   "age": 54,
6   "isDeleted": false
7 }
```

Status: 200 OK Time: 16 ms Size: 281 B Save Response

```
1 {
2   "message": "Successfully Updated User's data"
3 }
```

## Checking the user is updated or not – its successfully updated

This screenshot shows a GET request to `localhost:4000/getUser/452354` in Postman. The response status is 200 OK, with a time of 4 ms and a size of 318 B. The response body is a JSON object with the following fields: `id` (452354), `login` (Jack), `password` (Password1234), `age` (54), and `isDeleted` (false).

Status: 200 OK Time: 4 ms Size: 318 B Save Response

```
1 {
2   "id": "452354",
3   "login": "Jack",
4   "password": "Password1234",
5   "age": 54,
6   "isDeleted": false
7 }
```

# Postman Results:

## localhost:4000/updateUser/id – (Failed case the age must be in between 4-5 Joi validation case):

POST localhost:4000/updateUser/452354 Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL ☒ JSON Beautify

```
1 {
2   "id": "452354",
3   "login": "Jack",
4   "password": "Password1234",
5   "age": 3,
6   "isDeleted": false
7 }
```

**Body** Cookies Headers (7) Test Results Status: 404 Not Found Time: 10 ms Size: 411 B Save Response

Pretty Raw Preview Visualize JSON 🔍

```
1 {
2   "errorMessage": [
3     {
4       "message": "\"age\" must be greater than or equal to 4",
5       "path": [
6         "age"
7       ],
8       "type": "number.min",
9       "context": {
10        "limit": 4,
11        "value": 3,
12        "label": "age",
13        "key": "age"
14      }
15    }
16  ]
17 }
```

## localhost:4000/updateUser/id – (Failed case missing the required field id Joi validation):

POST localhost:4000/updateUser/452354 Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL ☒ JSON Beautify

```
1 {
2   "login": "Jack",
3   "password": "Password1234",
4   "age": 31,
5   "isDeleted": false
6 }
```

**Body** Cookies Headers (7) Test Results Status: 404 Not Found Time: 10 ms Size: 366 B Save Response

Pretty Raw Preview Visualize JSON 🔍

```
1 {
2   "errorMessage": [
3     {
4       "message": "\"id\" is required",
5       "path": [
6         "id"
7       ],
8       "type": "any.required",
9       "context": {
10        "label": "id",
11        "key": "id"
12      }
13    }
14  ]
15 }
```

# Postman Results:

**localhost:4000/getAutoSuggestUsers/substring/limit – To get a list to sorted users with substring and limit:**

GET localhost:4000/getAutoSuggestUsers/AI/2

Status: 200 OK Time: 5 ms Size: 445 B Save Response

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "The Suggested Users",
3   "Users": [
4     {
5       "id": "452352",
6       "login": "Alex",
7       "password": "Password12",
8       "age": 24,
9       "isDeleted": false
10    },
11    {
12      "id": "452353",
13      "login": "Alice",
14      "password": "Password123",
15      "age": 27
```

**localhost:4000/deleteUser/id – To delete a particular user with specific id**

DELETE localhost:4000/deleteUser/452352

Status: 200 OK Time: 18 ms Size: 279 B Save Response

Body Cookies Headers (7) Test Results

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

1

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Successfully Deleted some data"
3 }
```

**checking whether the isDeleted of particular id is changed to true or not after deleting – its changed**

GET localhost:4000/getUser/452352

Status: 200 OK Time: 5 ms Size: 315 B Save Response

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": "452352",
3   "login": "Alex",
4   "password": "Password12",
5   "age": 24,
6   "isDeleted": true
7 }
```

# Postman Results:

## Validation case if we delete already deleted user or the user which is not in the record



DELETE localhost:4000/deleteUser/452355 Send

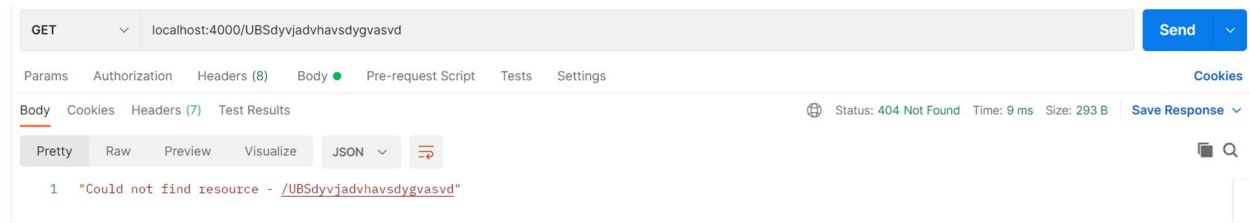
Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (7) Test Results Status: 404 Not Found Time: 7 ms Size: 329 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Unable to find the user to delete OR Already the user is set to be Deleted"
3 }
```

## Validation case if any random url is entered



GET localhost:4000/UBSdyvjadvhavsdygvasvd Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (7) Test Results Status: 404 Not Found Time: 9 ms Size: 293 B Save Response

Pretty Raw Preview Visualize JSON

```
1 "Could not find resource - /UBSdyvjadvhavsdygvasvd"
```

\*\*\*\*\*