Node Assignment 1:

Build a RESTful API using Node.js and Express

Submitted By :
Chaitali Mahato

Github: https://github.com/chaitali9497/restful-api

PROJECT OVERVIEW :

This project involves developing a simple RESTful API using Node.js and Express for managing user information. The application allows performing basic CRUD operations such as creating, retrieving, updating, and deleting users. It demonstrates core backend development concepts including RESTful routing, middleware implementation, handling different HTTP methods, proper use of status codes, structured error handling, and in-memory data storage for managing user data efficiently.

Technologies Used:

- ***Node.js***
- ***Express.js***
- ***Thunder Client (for API testing)***
- ***JavaScript***

Project Setup

- Initialized a Node.js project using `npm init`.
- Installed Express framework.
- Created `server.js` as the main entry file.
- Configured middleware for JSON parsing, logging, and validation.

API Endpoints Implemented:

Method	***Endpoint***	***Description***
GET	***/users***	***Fetch all users***
GET	***/users/:id***	***Fetch user by ID***
POST	***/user***	***Add a new user***
PUT	***/user/:id***	***Update an existing user***
DELETE	***/user/:id***	***Delete a user***

Middleware Implementation

1.Logging Middleware

- Logs request method and URL for every API call.
- Helps in monitoring incoming requests.

2. **Validation Middleware**

- Validates required fields (`firstName`, `lastName`, `hobby`) in POST and PUT requests.
- Returns 400 Bad Requests if any field is missing.

Error Handling

The API uses proper HTTP status codes and meaningful error messages:

Scenario	***Status Code***
Successful Request	*200*
User Created	*201*
User Not Found	*404*
Invalid Input	*400*

Test Results :

1) GET all users

Request:
Method: GET
URL: http://localhost:3000/users

Response:

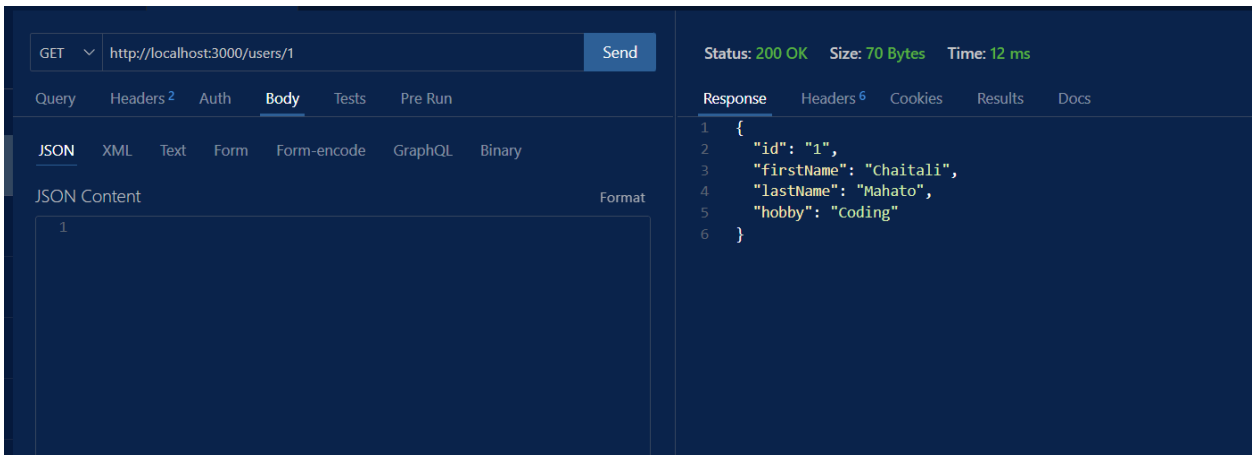

```
{
    "id": "1",
    "firstName": "Chaitali",
    "lastName": "Mahato",
    "hobby": "Coding"
 }
```


2) GET user by ID

Request:
Method: GET
URL: http://localhost:3000/users/1

Response:


```
{
   "id": "1",
   "firstName": "Chaitali",
   "lastName": "Mahato",
   "hobby": "Coding"
 }
```

3) POST add user

Request:

Method: POST

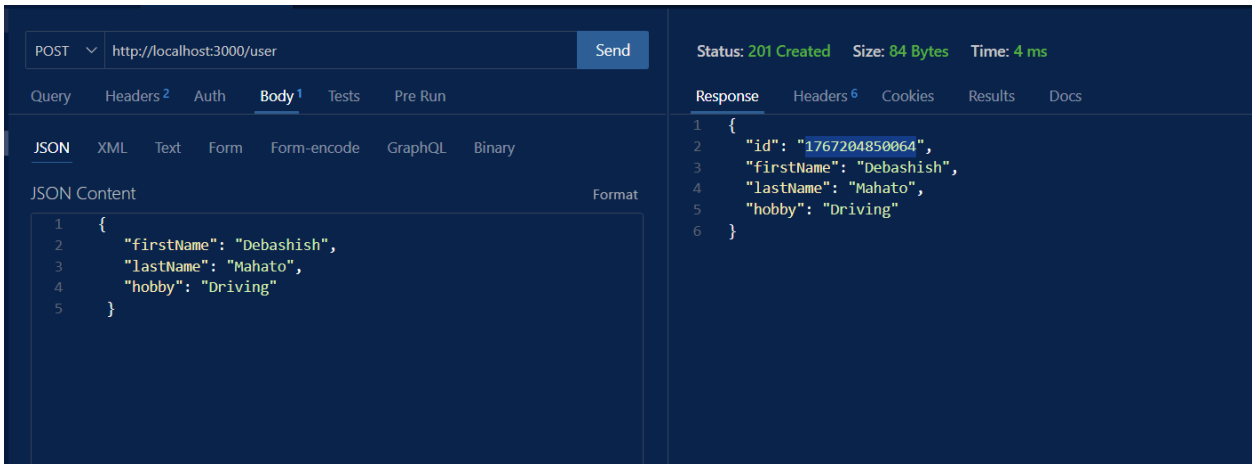
URL: http://localhost:3000/user

Body:

```
{
   "firstName": "Debashish",
   "lastName": "Mahato",
   "hobby": "Driving"
}
```

Response:

```
{
   "id": "1767204850064",
   "firstName": "Debashish",
   "lastName": "Mahato",
   "hobby": "Coding"
}
```



4) PUT update user

Request:

Method: PUT

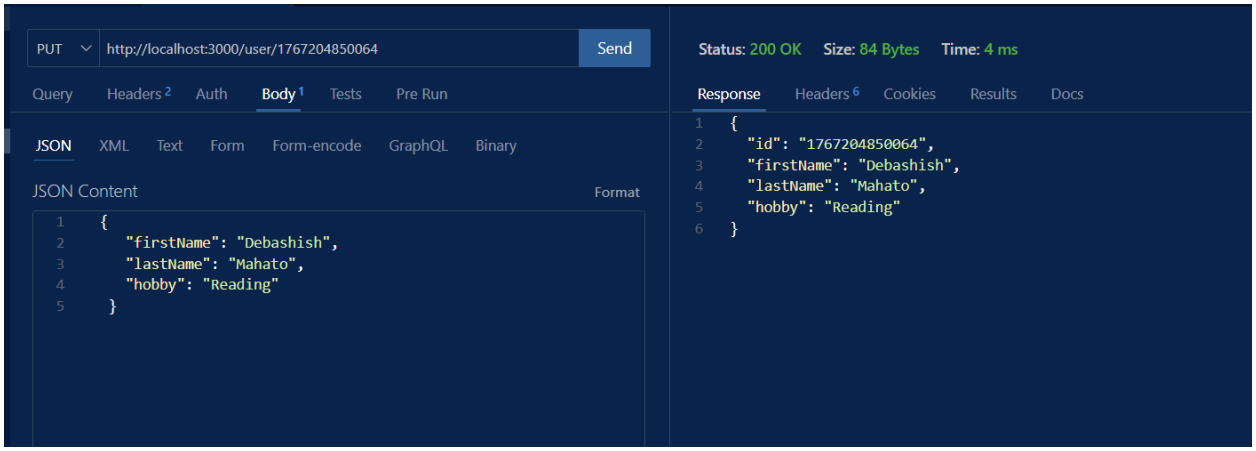
URL: http://localhost:3000/user/1767204850064

Body:

```
{
   "firstName": "Debashish",
   "lastName": "Mahato",
   "hobby": "Reading"
}
```

Response:

```
{
   "id": "1767204850064",
   "firstName": "Debashish",
   "lastName": "Mahato",
   "hobby": "Reading"
}
```

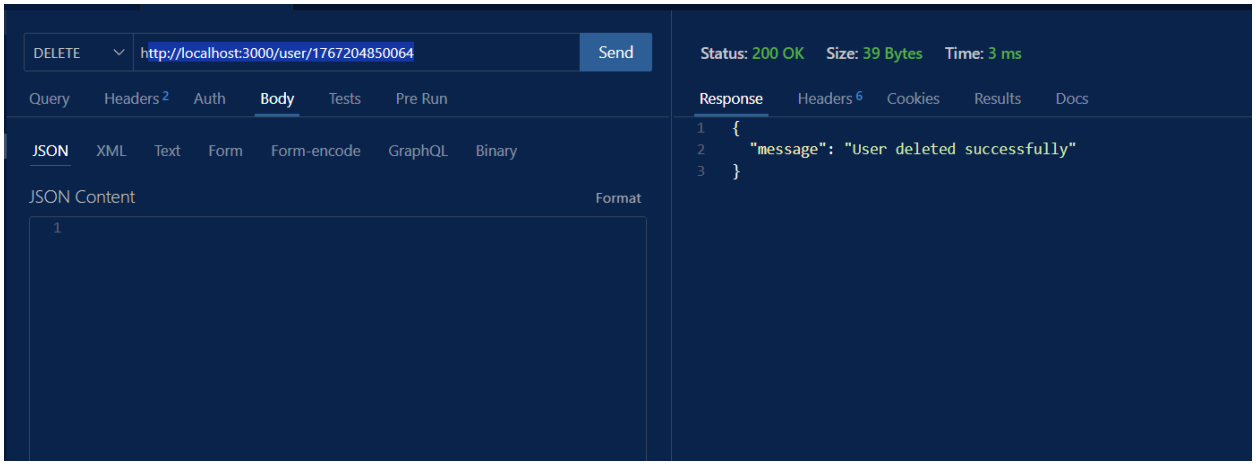


5) DELETE user

Request:
Method: DELETE
URL: http://localhost:3000/user/1767204850064

Response:

```
{
   “message” : “User deleted successfully”
 }
```



—---------- THANK YOU —----------