**NODE EXPRESS EXAMPLE-2**

In this Project, we will use Express.js to create a web server and then create API calls. These API calls will handle CRUD operations. CRUD stands for create, read, update, and Delete.

Let’s get started!

1.

Create an empty directory named node-express-CRUD and run:

npm init

2.

Fill out the required information to initialize your project. There should be package.json file created. (Yo may press the “Enter” key to leave the default settings as they are)

3.

Use the command below to install Express:

npm install express --save

4.

Create a file named index.js — this file will house the code for our application.

5.

- First require the express package.

- Create a variable app and assign it to express() method.

- At the end of the page, listen app for any incoming requests and console a message for server running port

Run your code with “node index.js” and you can see response data in console.

6.

express.json() is a middleware function that parses JSON payload, if any, in the incoming API requests. Use it.

app.use(express.json());

7.

Import all the routes that we will create in another directory

app.use("/api/users", require("./routes/api/users"));

8.

Then, create “UsersData.js” file under the project folder.

This file stores all user data in the user management system. Since this is a small project, we are not using a database here, and only using a list to store the users’ information.

const users = [

  {

    id: 1,

    name: "John",

    email: "john@gmail.com",

  },

  {

    id: 2,

    name: "Smith",

    email: "smith@gmail.com",

  },

  {

    id: 3,

    name: "Chris",

    email: "chris@gmail.com",

  },

  {

    id: 4,

    name: "Jack",

    email: "jack@gmail.com",

  },

];

module.exports = users;

9.

Then, create a folder named “routes”, and inside that, create another folder named “api”. After creating the folders, add a new file and name it users.js.

In this file, we will add the API routes that we want to include in the application.

10.

Inside users.js file;

Import “express” to create API routes,

Import “uuid” to generate ids for new users.

Import users data,

Create a variable by using this line of code: const router=express.Router();

11.

Now, create a GET request “/” to router and send json data of your users.

12.

Create a GET request (in other words define a route) “/:id” to search for a user’s data using their ID. It should returns the data if **found**; otherwise, it sends an error message.

Filter your users data and send a json data of your user if found.

HINT: use some or filter methods.

users.some((user) => user.id === parseInt(req.params.id));

13.

Create POST request to ‘/’.

Inside it, create a newuser object as model (takes data from req) and push this object to our users data.

Send json data of your users.

HINT:

const newUser = {

    id: uuid.v4(),

    name: req.body.name,

    email: req.body.email,

  };

14.

Create PUT request “/:id” to update the data of particular user if **found**; otherwise, it sends an error message.

HINT:

foundUser.name = req.body.name ? req.body.name : foundUser.name;

foundUser.email = req.body.email ? req.body.email : foundUser.email;

res.json({ msg: "User updated", user });

15.

Finally, create the last route for our Node.js application that takes in a user’s ID as input and deletes that user if **found**; otherwise, it sends an error message

HINT:

users = users.filter((user) => user.id !== parseInt(req.params.id));

res.json({msg: "User deleted",  users });

Do not forget to export your modüle.

Run your code and see results in your browser.

To test all routes you need to run postman.

Congragulations…