Building REST API with node.js and Express

API: application programming interface. the developer create API in the server & allows the client to talk to it.

REST: Determines how the API looks like . it stands for REpresentational State Transfer.

CRUD :crud is an Acronym for Create Read Update Delete.

Almost every website is CRUD in sense. whenever we have the data base of users we need to be able to create user accounts get/read the profile details, update the details & finally delete an account.

If we want build our own API, we need to have node.js, Express, POSTman, code editor called VS code.

Node.js: node js a javascript runtime environment built on chrome V8 engine of javascript.it allows the javascript to be run and places otherthan the chrome and other browsers. we can run in our console over in our terminal. We can use to create server side javascript applications.

Express: fast unopinionated minimalistic web framework for node js. It help us to create different routes for API.

POSTman: help us make different request to ur API’s so that we can test all our end points.

To initialize backend javascript application we need to run npm init-y.

Npm install –save express.

Node –v.

import bodyParser from 'body-parser'; //this allows to take in incoming post request bodies.

We need to initialize our express application const app.

We call express as function express();

const app = express();

now the whole application lies in the little variable called app

we can specify the post of our application.

const PORT  = 5000;

then we can initialize bodyParser middle ware

Even if a small change in the code we have to stop the engine and start again ctrl+c.

npm install --save-dev nodemon as dev dependency.

Npm start : as the npm start command.

How to we run our application with nodemon. "scripts": {

    "start": "node Index.js",

    "dev": "nodemon index"

  },

npm start

node and express primarily whole about Routing.

app.get('/',(req,res)=>{

       console.log('[Testing the server]!');

       res.send('<h1>Admit your mistake before somenone exaggerates<h1/>')

})

Get() method has the two parameters to pass in parenthesis.1)path '/' to the home page 2)callback function it accepts two parameters (req,res)=>{

}

GET /users find all users

POST /users creates a user

GET /users/id: find user details

DELETE /users/:id delete a user

PATCH /users/:id update a user

Json:javascript object notation. It is common format for sending and requesting data through rest api. A json object looks like a javascript object. In Jason ecach property and value are wrapped in double quotation marks instead of single quotation marks.

Browser can only make get request.

//this is index.js

import express from 'express';

//const express = require('express')

import bodyParser from 'body-parser';

const app = express();

const PORT  = 5000;

app.use(bodyParser.json()); //we  call it as function

app.get('/',(req,res)=>{

       console.log('[Testing the server]!');

       res.send('<h1>Admit your mistake before somenone exaggerates<h1/>')

})

app.get('/user',(req,res)=>{

    console.log('[test@@]');

    res.send('<h1>Give yourself to permission to be imperfect<h1>');

})

app.listen(PORT,()=>{

    console.log(`server running on port: http://localhost:${PORT}`);

})

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Creates routes to get request from browser

//this is index.js

import express from 'express';

//const express = require('express')

import bodyParser from 'body-parser';

const app = express();

const PORT  = 5000;

import usersRoutes from './routes/users.js'

app.use(bodyParser.json()); //we  call it as function

app.use('/users',usersRoutes);

app.listen(PORT,()=>{

    console.log(`server running on port: http://localhost:${PORT}`);

})

//this is users.js

import express from 'express';

const router = express.Router();

const users = [

    {

        firstName:"karthik yadav",

        lastName:"Barla",

        qualification:"MSc botany",

        year:2018

    },

{

    "first name":"Dhivya",

    "lastNmae":"kakatapalli",

    "qualification":"Btech",

    "year":2019

},

{

    "firstname":"uma dhevi",

    "lastName":"pilaka",

    "qualification":"Btech",

    "year":2020

}

]

router.get('/', (req,res)=>{

    console.log(users);

    //res.send(`<h1>you'll fail,don't give up in the process<h1>`);

    res.send(users)

});

export default router;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//post request..

//this is index.js

import express from 'express';

//const express = require('express')

import bodyParser from 'body-parser';

const app = express();

const PORT  = 5000;

import usersRoutes from './routes/users.js'

app.use(bodyParser.json()); //we  call it as function

app.use('/users',usersRoutes);

app.listen(PORT,()=>{

    console.log(`server running on port: http://localhost:${PORT}`);

})

//this is users.js

import express from 'express';

const router = express.Router();

const users = [

    {

        firstName:"karthik yadav",

        lastName:"Barla",

        qualification:"MSc botany",

        year:2018

    },

{

    "first name":"Dhivya",

    "lastNmae":"kakatapalli",

    "qualification":"Btech",

    "year":2019

},

{

    "firstname":"uma dhevi",

    "lastName":"pilaka",

    "qualification":"Btech",

    "year":2020

}

]

router.post('/', (req,res)=>{

    //console.log('dispaly the post in console');

    const user = req.body;

    users.push(user);

    //console.log(req.body);

    //res.send(`<h1>you'll fail,don't give up in the process<h1>`);

    //res.send('posted');

    //res.send(`User with name ${user.firstName} added to the database`)

    res.send(users);

});

export default router;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//getting single users with among users..

//this is index.js

import express from 'express';

//const express = require('express')

import bodyParser from 'body-parser';

const app = express();

const PORT  = 5000;

import usersRoutes from './routes/users.js'

app.use(bodyParser.json()); //we  call it as function

app.use('/users',usersRoutes);

app.listen(PORT,()=>{

    console.log(`server running on port: http://localhost:${PORT}`);

})

//this is users.js

import express from 'express';

const router = express.Router();

const users = [

    {

        id:101,

        genericName:"Hibiscus",

        specificName:"rosa-sinensis",

        family:"Malvaceae",

        class:"Dicot"

    },

    {

        id:102,

        genericName:"Abhelmoschus",

        specificName:"esculentus",

        family:"Malvaceae",

        class:"Dicot"

    },

    {

        id:103,

        genericName:"Chlitoris",

        specificName:"ternata",

        family:"fabaceae",

        calss:"Dicot"

    },

    {

        id:104,

        genericName:"Cymbopogan",

        specificName:"citratus",

        family:"cyperaceae",

        class:"Monocot"

    },

    {

        id:105,

        genericName:"Cocos",

        specificName:"nucifera",

        family:"Anacardiacea",

        class:"monocot"

    }

]

router.get('/:id',(req,res)=>{

    console.log('i got users in json format');

    const user = users.find(bin=>bin.id === parseInt(req.params.id));

    if(!user) res.status(404).send('the course with the given id was not found');

  res.send(user);

    //res.send(users)

});

export default router;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//deleting a user from data.

//this is index.js

import express from 'express';

//const express = require('express')

import bodyParser from 'body-parser';

const app = express();

const PORT  = 5000;

import usersRoutes from './routes/users.js'

app.use(bodyParser.json()); //we  call it as function

app.use('/users',usersRoutes);

app.listen(PORT,()=>{

    console.log(`server running on port: http://localhost:${PORT}`);

})

//this is users.js

import express from 'express';

const router = express.Router();

let users = [

    {

        id:101,

        genericName:"Hibiscus",

        specificName:"rosa-sinensis",

        family:"Malvaceae",

        class:"Dicot"

    },

    {

        id:102,

        genericName:"Abhelmoschus",

        specificName:"esculentus",

        family:"Malvaceae",

        class:"Dicot"

    },

    {

        id:103,

        genericName:"Chlitoris",

        specificName:"ternata",

        family:"fabaceae",

        calss:"Dicot"

    },

    {

        id:104,

        genericName:"Cymbopogan",

        specificName:"citratus",

        family:"cyperaceae",

        class:"Monocot"

    },

    {

        id:105,

        genericName:"Cocos",

        specificName:"nucifera",

        family:"Anacardiacea",

        class:"monocot"

    }

]

router.delete('/:id',(req,res)=>{

    console.log('i got users in json format');

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

    if(!user) res.status(404).send(`user with thje id ${id} has deleted from the database`);

  res.send(user);

    //res.send(users)

});

export default router;

What is express??

Express is a minimal and flexible node.js web application framework that provides robust set of features for web and mobile applications.

It is an open source framework developed and maintained by node.js foundation.

Express is web application framework that provides with a simple middleware APIs to build websites, web apps and backs ends.

With epress JS, you need not worry about low level protocols , processes ,etc.

It is flexible as there are numerous modules available in on npm, which can directly plugged into Express.

Express.js maintained by the node.js foundation an numerous open source contributors.

Express is a fast, un-opinionated and minimalist web framework for node.js.

Express is a “server-side” or “back-end” framework.it is not comparable to client side framework like React, Angular & Vue. It can be used in combination with those framework to build full stack application.

Why express??

Easy, robust, and quick way to set up backend APIs

All batteries included.

Express is a fast, un-opinionated and minimalist web framework for node.js.

It is very flexible and pluggable.

Installation of Express.js

We should have node and npm(node package manager) installed in our mechine..node –version npm –version.

We need to continuously change / and update the code –we need auto monitor and /hot reload. npm install –D nodemon

Initialize the npm using below command. npm init

Install express. npm install express --save

var express = require('express'); //importing the express.js library into ou application.

var app = express(); //initialising the application using express

var port = 3000;

app.get('/',(req,res)=>{

    res.send("<h1>Ficus bengalensis</h1>");

})

/\*app.listen(port,()=>{

  console.log(`listening at ${port}`);

})\*/

app.listen(3000);

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

What is an API??

Application program interface

Postman??

Cool tool to do all API testings.

We can make requests via different HTTP methods.

We can view responses and headers.

We can set authentication headers.

We can create collection for grouping API requests.

We can create export collections.

We can create documentation for API.

CRUD Operations

Create – create new source. Adding new e-mail, new task, new user

Read – Read resource from server. reading re-sources from server.

Update – Update resource. Updating existing record at the server.

Delete – Delete a resource. Deleting the record at server side.

Creating a user, reading the user, updating the user & deleting the user.

HTTP Methods

Crude corresponds to HTTP methods

Post posting data from ui to server.hidden way secured way to create new resource.

Post method requests server accept the data.

Get requesting data from the server.

Put putting certain data for certain conditions for update and modifications..

Index.js

/\*const express = require('express')

const app = express()

const port = 80

app.get('/', (req, res) => {

  res.send('world!')

})

app.listen(port, () => {

  console.log(`Example app listening at http://localhost:${port}`)

})\*/

/\*var express = require('express');

var app = express();

var port = 3000;

app.get('/',(req,res)=>{

    res.send("<h1>Ficus bengalensis</h1>");

})

/\*app.listen(port,()=>{

  console.log(`listening at ${port}`);

})\*/

//app.listen(3000);