**Everything about Rest API -** [Ram Potabatti](https://medium.com/@rampotabatti?source=post_page-----4ab4ee9ea2c8--------------------------------) Apr 2, 2021

In this tutorial, we would study what REST API is and also learn to create a REST API to use in our Projects.

Let’s go from basics, have a look at API. Application Programming Interface can be referred to as a bridge that communicates between Client and Server. It is a software intermediary that allows two applications to talk with each other. API is simply like a waiter in a hotel, which serves the customer’s request by providing the necessary dish or food, in this case, the Client asks for a Request, and a response is served through API. Indeed, this was an old scenario.

Today’s Modern Applications use the concept of REST API, REST API is nothing but the Architectural Style to access the resources through the different Restful Web Services. REST API nowadays, used almost all types of Applications, i.e. Web Applications and also Native Applications.

Now, we would look at some of the **Important Methods of REST API.**

**1. HTTP GET Method**

GET Method, is typically used to get or fetch all or specific resources among the webserver. This method is commonly used to show data to the client-side.

**2. HTTP POST Method**

POST Method is basically used to pass the data from Client to Server. Suppose, a user fills up a form with his/her credentials on any (REST API Tech implemented website), then that data would further save in the Database with the help of POST Method.

**3. HTTP PUT Method**

PUT method is the method to update the data present inside the Database, and here this method normally can be used to change values that are saved into the Database.

**4. HTTP DELETE Method**

As the name speaks, we use the HTTP DELETE method to delete specific or all resources among the Database.

Now, Let’s dive deep and understand how to **Create a Restful API using Node JS and Express -**

We would need some of the requirements before we go-ahead:

1. NPM (or Yarn) Installed — To create a Node Application.
2. Postman — Postman is basically an HTTP Client Software that helps to Create and Test the Rest of API Services.

Now, create a Directory called nodeAPI in C drive, go through this directory, and type:

code .

This command will open VS Code, through that particular folder-

Now, type as follows:

npm init

This command could create a Package.json file. Now, install Express JS into our Application using the following code.

npm i --save express nodemon

This command would bring up the node\_modules folder in our app. And nodemon is a package that is used to reload the server every time it sees updates in code.

Now, it's time to create an index.js file inside our root directory.

Our folder would look as follows:

node\_API - node\_modules  
 - index.js  
 - package.json  
 - package-lock.json

Now, copy the further code and paste it inside the index.js file:

const express = require('expres');const app = express();app.use(express.json());//HTTP GET Method, to fetch the data from Server to Client.app.get("/",(req,res) => {res.send("Hello Developers!")})app.listen(9000,() => { return "Server starts..."})

Now, let’s run the Server and check whether we are on the right path.

nodemon

And, this command runs the server, now, let's open up Postman to check whether our APIs are working properly or not.

Once, we have entered into Postman, we would go to the URL section and type the following:

http://localhost:9000

NOTE — We must use the same port number, which is specified inside the code.

Once, we have entered the above URL and hit send button, we would see a message — “Hello Developers!”. This is because we have sent a response of a string to “/” assuming when a user calls that API.

This is in fact a GET API, but now we would learn how to fetch data from API.

**GET API**

To fetch data, we would require information and so we would use this array of objects.

const users = [{ "username" : "Sundar Pichai", "id" : 1},{ "username" : "Elon Musk", "id" : 2},{ "username" : "Jeff Bezos", "id" : 3},{ "username" : "Bill Gates", "id" : 4},]

And also, create an API route -

app.get("/api/users", (req,res) => {res.send(users);})

Now, if we test this API using Postman, we would probably GET all the objects present inside the Array.

**POST API**

Inside the POST API call, we would get the username from the Client. The code can be specified as follows:

app.post("/api/users/post" , (req,res) => {username = req.body.usernameif(username){res.send({msg:"Success"})}return res.send({msg:"Error"})})

Inside the Postman, first, we need to switch the GET method to the POST method, and also turn the body as JSON format and add the following code:

{"username" : "Smith"}

If this returns a response as Success, then our POST API works well then fine.

**PUT API**

PUT API can be used to update the data, which is present inside the Database. Here, we pass two parameters from the client-side to the server, since we need to update the particular row in a table and hence we ask a client to enter (username) id and new-username to update the old data.

app.put("/api/user-update/:id" , (req,res) => {const input\_id = req.params.id;const input\_username = req.params.username;const userUpdate = users.find(n => n.id === parseInt(input\_id))if(!userUpdate){res.send("ID Invalid")}const {Error} = {msg:"Error"};if(Error){res.send({msg:"Error"})}else{users.username = input\_username;res.send({msg:"Success"})}})

Now, while you send a request from Postman, you have to add an id parameter and also pass a new username as a JSON payload to update the Data.

**DELETE API**

app.delete("/api/delete-user/:id", (req,res) => {const input\_id = users.find(n => n.id === parseInt(req.params.id))if(!input\_id){res.send("ID Invalid!")}const index = users.indexOf(users)users.splice(index,1)res.send(input\_id)})

In this case, we have to pass the id as a parameter through Postman, which directly deletes a record from the DB.

Refer to the Repo [here](https://github.com/balram1402/NodeJS_API.git).

This may be the biggest reason which makes a Restful API much simple and easy to use. Such APIs are used in day-to-day modern web apps as they render content fast and are also not that handy in coding.