BackEnd Development

Node.js

Node js allows to talk with computer hardware and minimise use of GUI.

1]$pwd: Print Working Directory

* Show Current Directory Work
* Eg:/Users/angelaYu
* Directory starts with root.
* Users is the folder inside Root.
* angelaYu is the folder inside Users.

2]$cd Desktop/

* Change Directory.
* Moves towards Desktop Folder
* Tab then autocomplete the folder.
* hp@DESKTOP-U46E5PJ MINGW64 ~
* $ cd OneDrive;
* hp@DESKTOP-U46E5PJ MINGW64 ~/OneDrive
* $ cd Desktop;
* hp@DESKTOP-U46E5PJ MINGW64 ~/OneDrive/Desktop
* Currently we are in Desktop Folder.

hp@DESKTOP-U46E5PJ MINGW64 ~/OneDrive/Desktop

$ pwd;

/c/Users/hp/OneDrive/Desktop

Show us the current Directory where we are in.

We can also us ls(list files)

$ls

These will print all Files inside Desktop.

3]$mkdir(Make Directory)

* Used to a make a new directory and folder.
* hp@DESKTOP-U46E5PJ MINGW64 ~/OneDrive/Desktop

$ ls

05092022.txt Atom.lnk\* 'GitHub Desktop.lnk'\* 'Java Programs MIT'/ 'New Text Document.txt' react-ultimate-resume-develop/ Try.html "Tushar's Resume.pdf" YouTube.lnk\*

ABC\_ID.txt desktop.ini Hyper.lnk\* 'Microsoft Teams.lnk'\* nOTES.txt SlideToShutDown.exe.lnk\* Tushar/ 'Visual Studio Code.lnk'\* Zoom.lnk\*

hp@DESKTOP-U46E5PJ MINGW64 ~/OneDrive/Desktop

$ mkdir Node.js;

hp@DESKTOP-U46E5PJ MINGW64 ~/OneDrive/Desktop

$ ls

05092022.txt desktop.ini 'Java Programs MIT'/ **Node.js/**  SlideToShutDown.exe.lnk\* "Tushar's Resume.pdf" Zoom.lnk\*

ABC\_ID.txt 'GitHub Desktop.lnk'\* 'Microsoft Teams.lnk'\* nOTES.txt Try.html 'Visual Studio Code.lnk'\*

Atom.lnk\* Hyper.lnk\* 'New Text Document.txt' react-ultimate-resume-develop/ Tushar/ YouTube.lnk\*

As you can see Node.js folder is added on desktop.

4]$touch

* Create the new file.
* $ touch index.js

Creates the file name index.js in Node.js Folder

5]$node index.js

* Helps to run the .js file on command line prompt.
* $node index.js

Hello World!

6]Repl(Read Evaluation Print Loop)

* Repl is installed when we download the node.js application.
* We can do operations same as Chrome DeveloperTool.
* Console+”Two Tabs Click” give options.
* .exit to exit from the repl node.
* Crtl+c Two times exit from the terminal.

Text

Description automatically generated with low confidence

**Chapter 2: Native Node Modules.**

<https://node.js.org/api/fs.html>

Documentation to use method for file system module and many more modules.

const fs = require("fs");

fs.copyFileSync("file1.txt", "file2.txt");

Here fs is file system.

Here we are try to import file system and use its method.

File1 has some text and then file2 has same text then after changing text in file2,then we rerun the code same content of file1 is replaced in file2 as before.

**Chapter 3:Imp NPM**

NPM is Node Program Manager.

npm is two things: first and foremost, it is an online repository for the publishing of open-source Node.js projects; second, it is a command-line utility for interacting with said repository that aids in package installation, version management, and dependency management.

<https://www.npmjs.com>

$ npm init;

This utility will walk you through creating a package.json file.

It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields

and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and

save it as a dependency in the package.json file.

Press ^C at any time to quit.

package name: (node)

version: (1.0.0)

git repository:

keywords:

license: (ISC)

About to write to C:\Users\hp\Onedrive\Desktop\Node.js\package.json:

{

"name": "node",

"version": "1.0.0",

"description": "This intro to node.js",

"main": "index.js",

"scripts": {

"test": "echo \"Error: no test specified\" && exit 1"

},

"author": "Tushar Bhansali",

"license": "ISC",

"dependencies": {

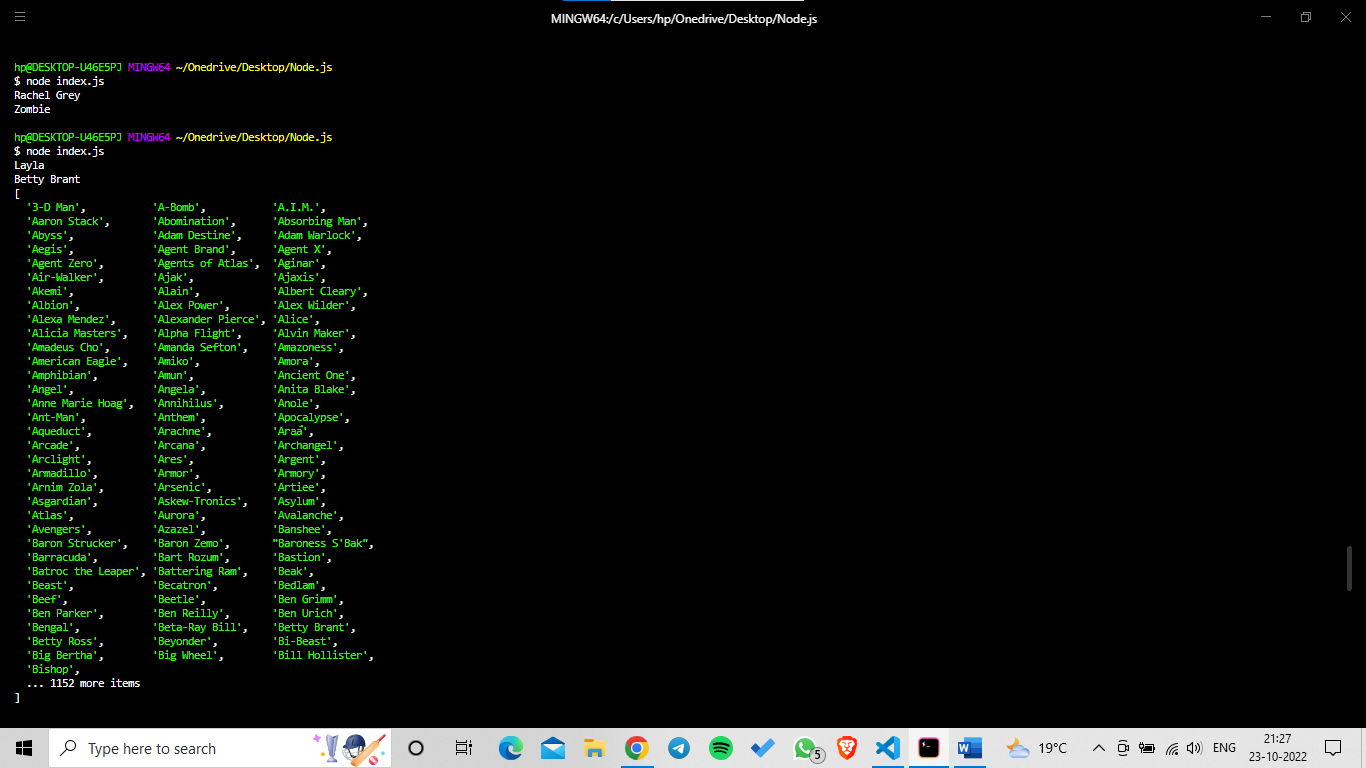
"marvel-characters": "^1.0.3",

"superheroes": "^3.0.0"

},

"devDependencies": {}

}



Package.json is created:

{

  "name": "node",

  "version": "1.0.0",

  "description": "This intro to node.js",

  "main": "index.js",

  "scripts": {

    "test": "echo \"Error: no test specified\" && exit 1"

  },

  "author": "Tushar Bhansali",

  "license": "ISC",

  "dependencies": {

    "marvel-characters": "^1.0.3",

    "superheroes": "^3.0.0"

  }

}

Package-lock.json is also created

npm install marvel-characters

npm install superheroes

{

  "name": "node",

  "version": "1.0.0",

  "lockfileVersion": 2,

  "requires": true,

  "packages": {

    "": {

      "name": "node",

      "version": "1.0.0",

      "license": "ISC",

      "dependencies": {

        "marvel-characters": "^1.0.3",

        "superheroes": "^3.0.0"

      },

      "devDependencies": {}

    },

    "node\_modules/marvel-characters": {

      "version": "1.0.3",

      "resolved": "https://registry.npmjs.org/marvel-characters/-/marvel-characters-1.0.3.tgz",

      "integrity": "sha512-79/Iox3UNuXYVnvAjO1XxJUE4+IbJSgojjdDl1OC48s/j5LW9ZZ1LoHnCBBuok/zB8Jhzbj8PmLCkoEnmNrg/Q==",

      "dependencies": {

        "minimist": "^1.1.1",

        "unique-random-array": "^1.0.0"

      },

      "bin": {

        "marvel-characters": "cli.js"

      }

    },

    "node\_modules/marvel-characters/node\_modules/unique-random": {

      "version": "1.0.0",

      "resolved": "https://registry.npmjs.org/unique-random/-/unique-random-1.0.0.tgz",

      "integrity": "sha512-K1sUkPf9EXCZFNIlMCoX4icAqcvkR4FMPH4Z61HbyiWhQl1ZGo0zYeV2bJmocK8Cp6tnKYrCnpkeKGebXZoRTQ==",

      "engines": {

        "node": ">=0.10.0"

      }

    },

    "node\_modules/marvel-characters/node\_modules/unique-random-array": {

      "version": "1.0.1",

      "resolved": "https://registry.npmjs.org/unique-random-array/-/unique-random-array-1.0.1.tgz",

      "integrity": "sha512-z9J/SV8CUIhIRROcHe9YUoAT6XthUJt0oUyLGgobiXJprDP9O9dsErNevvSaAv5BkhwFEVPn6nIEOKeNE6Ck1Q==",

      "dependencies": {

        "unique-random": "^1.0.0"

      },

      "engines": {

        "node": ">=0.10.0"

      }

    },

    "node\_modules/minimist": {

      "version": "1.2.7",

      "resolved": "https://registry.npmjs.org/minimist/-/minimist-1.2.7.tgz",

      "integrity": "sha512-bzfL1YUZsP41gmu/qjrEk0Q6i2ix/cVeAhbCbqH9u3zYutS1cLg00qhrD0M2MVdCcx4Sc0UpP2eBWo9rotpq6g==",

      "funding": {

        "url": "https://github.com/sponsors/ljharb"

      }

    },

    "node\_modules/superheroes": {

      "version": "3.0.0",

      "resolved": "https://registry.npmjs.org/superheroes/-/superheroes-3.0.0.tgz",

      "integrity": "sha512-XXXzeKHMnf0rmZItYkGU803JlqYpoxvxzKFoe6k8C4bolcKfLZH716Rm4DyNJhxPTurbzDEB/QC7TXGsoei+ew==",

      "dependencies": {

        "unique-random-array": "^2.0.0"

      },

      "engines": {

        "node": ">=8"

      }

    },

    "node\_modules/unique-random": {

      "version": "2.1.0",

      "resolved": "https://registry.npmjs.org/unique-random/-/unique-random-2.1.0.tgz",

      "integrity": "sha512-iQ1ZgWac3b8YxGThecQFRQiqgk6xFERRwHZIWeVVsqlbmgCRl0PY13R4mUkodNgctmg5b5odG1nyW/IbOxQTqg==",

      "engines": {

        "node": ">=6"

      }

    },

    "node\_modules/unique-random-array": {

      "version": "2.0.0",

      "resolved": "https://registry.npmjs.org/unique-random-array/-/unique-random-array-2.0.0.tgz",

      "integrity": "sha512-xR87O95fZ7hljw84J8r1YDXrvffPLWN513BNOP4Bv0KcgG5dyEUrHwsvP7mVAOKg4Y80uqRbpUk0GKr8il70qg==",

      "dependencies": {

        "unique-random": "^2.1.0"

      },

      "engines": {

        "node": ">=8"

      }

    }

  },

  "dependencies": {

    "marvel-characters": {

      "version": "1.0.3",

      "resolved": "https://registry.npmjs.org/marvel-characters/-/marvel-characters-1.0.3.tgz",

      "integrity": "sha512-79/Iox3UNuXYVnvAjO1XxJUE4+IbJSgojjdDl1OC48s/j5LW9ZZ1LoHnCBBuok/zB8Jhzbj8PmLCkoEnmNrg/Q==",

      "requires": {

        "minimist": "^1.1.1",

        "unique-random-array": "^1.0.0"

      },

      "dependencies": {

        "unique-random": {

          "version": "1.0.0",

          "resolved": "https://registry.npmjs.org/unique-random/-/unique-random-1.0.0.tgz",

          "integrity": "sha512-K1sUkPf9EXCZFNIlMCoX4icAqcvkR4FMPH4Z61HbyiWhQl1ZGo0zYeV2bJmocK8Cp6tnKYrCnpkeKGebXZoRTQ=="

        },

        "unique-random-array": {

          "version": "1.0.1",

          "resolved": "https://registry.npmjs.org/unique-random-array/-/unique-random-array-1.0.1.tgz",

          "integrity": "sha512-z9J/SV8CUIhIRROcHe9YUoAT6XthUJt0oUyLGgobiXJprDP9O9dsErNevvSaAv5BkhwFEVPn6nIEOKeNE6Ck1Q==",

          "requires": {

            "unique-random": "^1.0.0"

          }

        }

      }

    },

    "minimist": {

      "version": "1.2.7",

      "resolved": "https://registry.npmjs.org/minimist/-/minimist-1.2.7.tgz",

      "integrity": "sha512-bzfL1YUZsP41gmu/qjrEk0Q6i2ix/cVeAhbCbqH9u3zYutS1cLg00qhrD0M2MVdCcx4Sc0UpP2eBWo9rotpq6g=="

    },

    "superheroes": {

      "version": "3.0.0",

      "resolved": "https://registry.npmjs.org/superheroes/-/superheroes-3.0.0.tgz",

      "integrity": "sha512-XXXzeKHMnf0rmZItYkGU803JlqYpoxvxzKFoe6k8C4bolcKfLZH716Rm4DyNJhxPTurbzDEB/QC7TXGsoei+ew==",

      "requires": {

        "unique-random-array": "^2.0.0"

      }

    },

    "unique-random": {

      "version": "2.1.0",

      "resolved": "https://registry.npmjs.org/unique-random/-/unique-random-2.1.0.tgz",

      "integrity": "sha512-iQ1ZgWac3b8YxGThecQFRQiqgk6xFERRwHZIWeVVsqlbmgCRl0PY13R4mUkodNgctmg5b5odG1nyW/IbOxQTqg=="

    },

    "unique-random-array": {

      "version": "2.0.0",

      "resolved": "https://registry.npmjs.org/unique-random-array/-/unique-random-array-2.0.0.tgz",

      "integrity": "sha512-xR87O95fZ7hljw84J8r1YDXrvffPLWN513BNOP4Bv0KcgG5dyEUrHwsvP7mVAOKg4Y80uqRbpUk0GKr8il70qg==",

      "requires": {

        "unique-random": "^2.1.0"

      }

    }

  }

}

Node.js framework Express.js

**Chapter 4: Express.js**

**4.1 Creating out first local Server using Express:**

Npm install express;

Creating First Sever

//jshint  esversion:6

const express = require("express");

const app = express();

const port = 3000;

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

  res.send("<h1>Hello World</h1>"); // sends data to browser.// we can use html tags

  console.log("Hello World");

  //   console.log(req);

  //   console.log(res);

});

///localhost:3000/

//|              |

// route        names of the further pages.

app.listen(port, () => {

  console.log("Example app listening on port 3000 ");

});

//used port is created properly or not

**WORKING OF THE ROUTE:**

 app.get("/contact", (req, res) => {

//here get is the higher order function with parameters action(pages),callback function

  res.send("My email id:tushar@gmail.com");

});

//localhost:3000/contact

// |              |

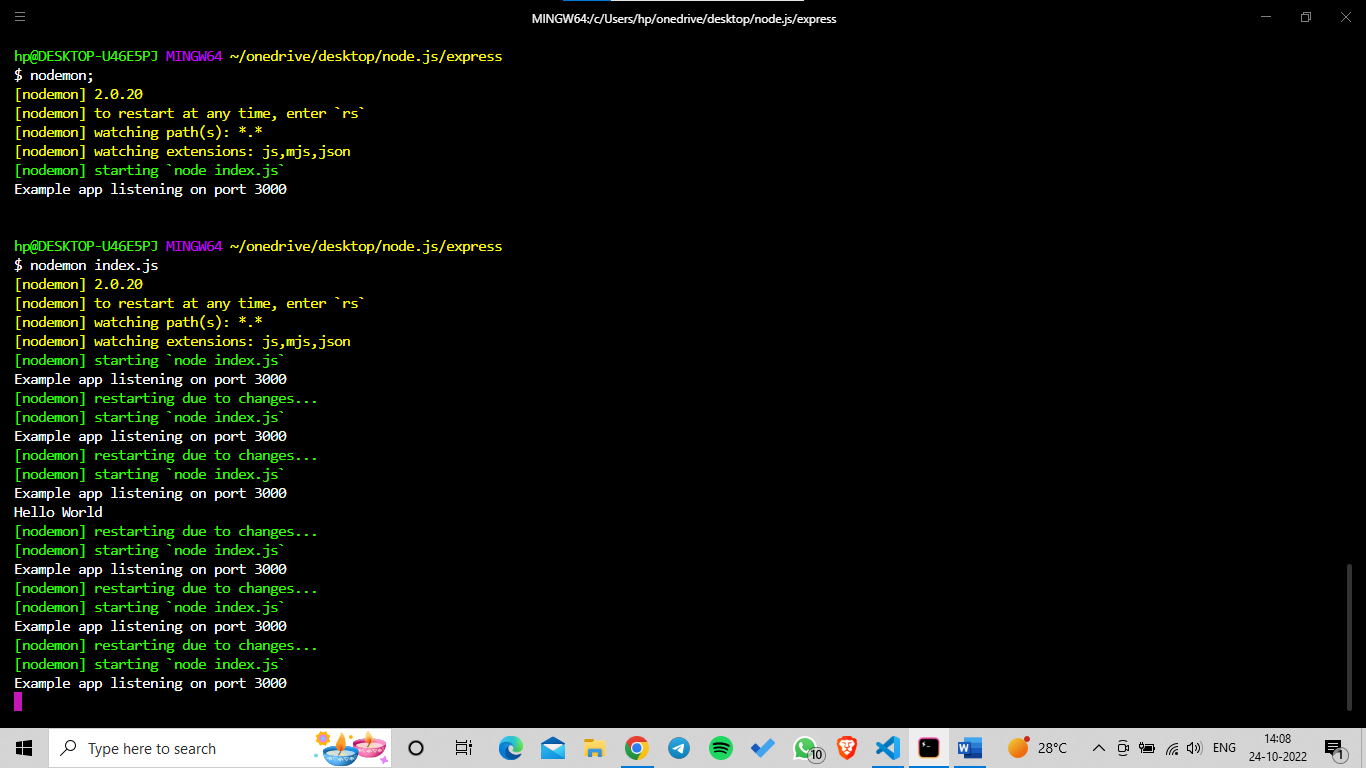
// root           names of the further pages.

**Graphical user interface, text, application

Description automatically generated**

Nodemon:

Helps to restart server automatically whenever there change in the code of index.js



Calculator.html

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

  </head>

  <body>

    <h1>Try to make Calculator</h1>

    <form action="/" method="post">

      <!--/ indicates localhost:3000 -->

      <input type="text1" name="num1" placeholder="First number" />

      <input type="text2" name="num2" placeholder="Second number" />

      <input type="text3" name="num3" placeholder="+,-,/,%" />

      <button type="submit" name="submit">Calculate</button>

    </form>

  </body>

</html>

Calculator.js

// const { application } = require("express");

const calci = require("express");

const bodyParser = require("body-parser");

const app = calci();

app.use(

  bodyParser.urlencoded({

    extended: true,

  })

);

const port = 3000;

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

  res.sendFile(\_\_dirname + "/index.html");

  //                  |

  //c:\Users\hp\OneDrive\Desktop\Node.js\Express\Calculator\index.html

  //console.log(\_\_dirname);

  // \_\_dirname stores the directory on the sever to refer the html file for access the web application.

  //if we just put filename then server doesn't know where the file is located

});

app.post("/", function (req, res) {

  var num1 = parseInt(req.body.num1);

  var num2 = parseInt(req.body.num2);

  var num3 = req.body.num3;

  switch (num3) {

    case "+":

      res.send("The result is: " + parseInt(num1 + num2));

      break;

    case "-":

      res.send("The result is: " + parseInt(num1 - num2));

      break;

    case "/":

      res.send("The result is: " + parseInt(num1 / num2));

      break;

    case "%":

      res.send("The result is: " + parseInt(num1 % num2));

      break;

    case "log":

      var result = Math.log(parseInt(num1));

      res.send("The result is: " + result);

      break;

    default:

      res.send("Wrong Option");

  }

});

app.listen(port, () => {

  console.log("Port is working");

});

//to check port is connected and working.

//npm body parser is used to fetch the data.

BMI is pending.

API’S:

* EndPoints
* Path
* Parameters
* Authentication.

API works as fetch the data from a particular organization and use for create the a project for the Software developer and the external terms.

Example:**Joke API**

<https://v2.jokeapi.dev/joke/Programming?blacklistFlags=explicit&type=single&contains=Java>

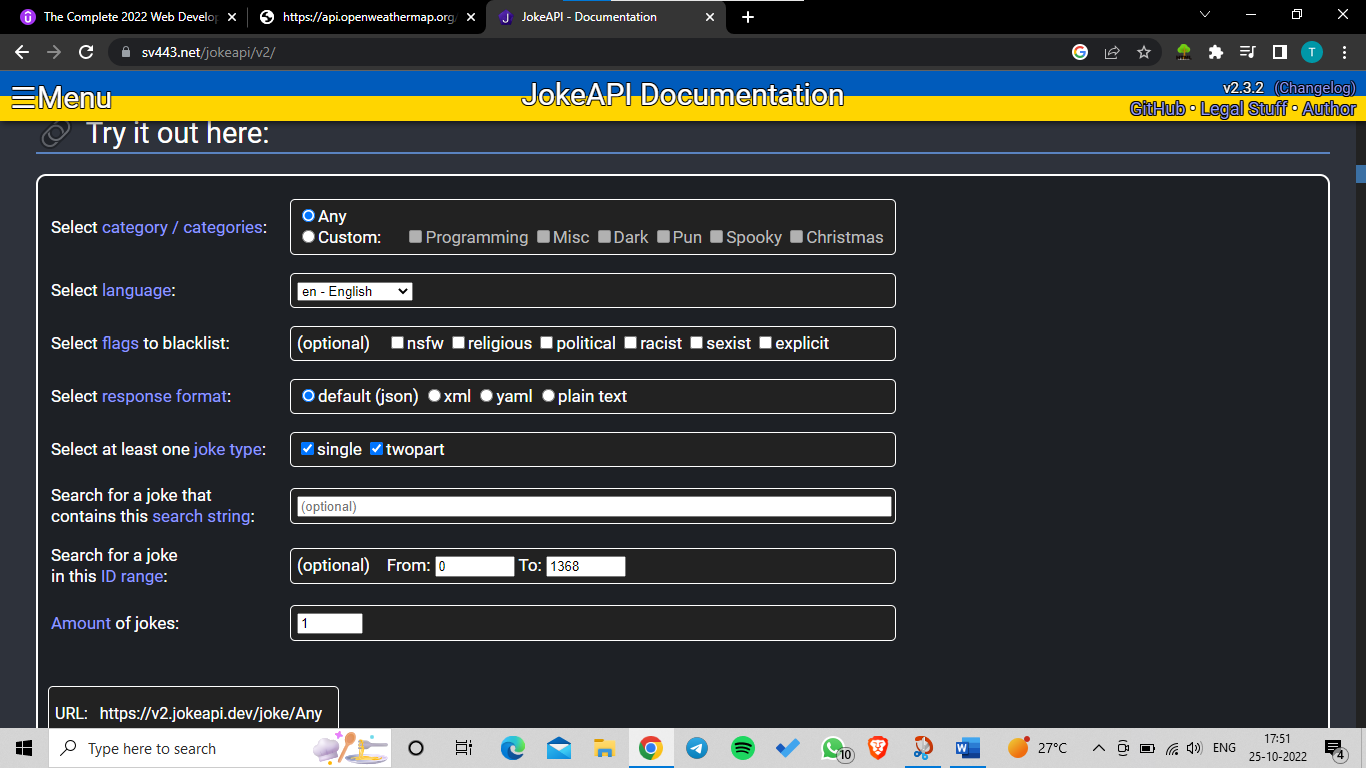
https://v2.jokeapi.dev/joke/ -> defines the endpoints

/Programming defines the path

? indicates further is the parameter

If there are two or more parameters they are seperated by &(ampersand)

It we check both conditions and then provide the joke related to it



Graphical user interface, text, application

Description automatically generated

**API Weather:**

[https://api.openweathermap.org/data/2.5/weather?q=Pune&appid=706a659514f1c8e26247274e21412ad1Graphical user interface, text, application, Word

Description automatically generated](https://api.openweathermap.org/data/2.5/weather?q=Pune&appid=706a659514f1c8e26247274e21412ad1)

**POSTMAN API:**

Postman: Postman is an API(application programming interface) development tool which **helps to build, test and modify APIs.**

POSTMAN Interface:

Key are used to apply condition.

Send that request to openweatherapi and fetching the data

**Graphical user interface, text, application, email

Description automatically generated**

O/P above is in JSON Format.so lets understand the JSON format.

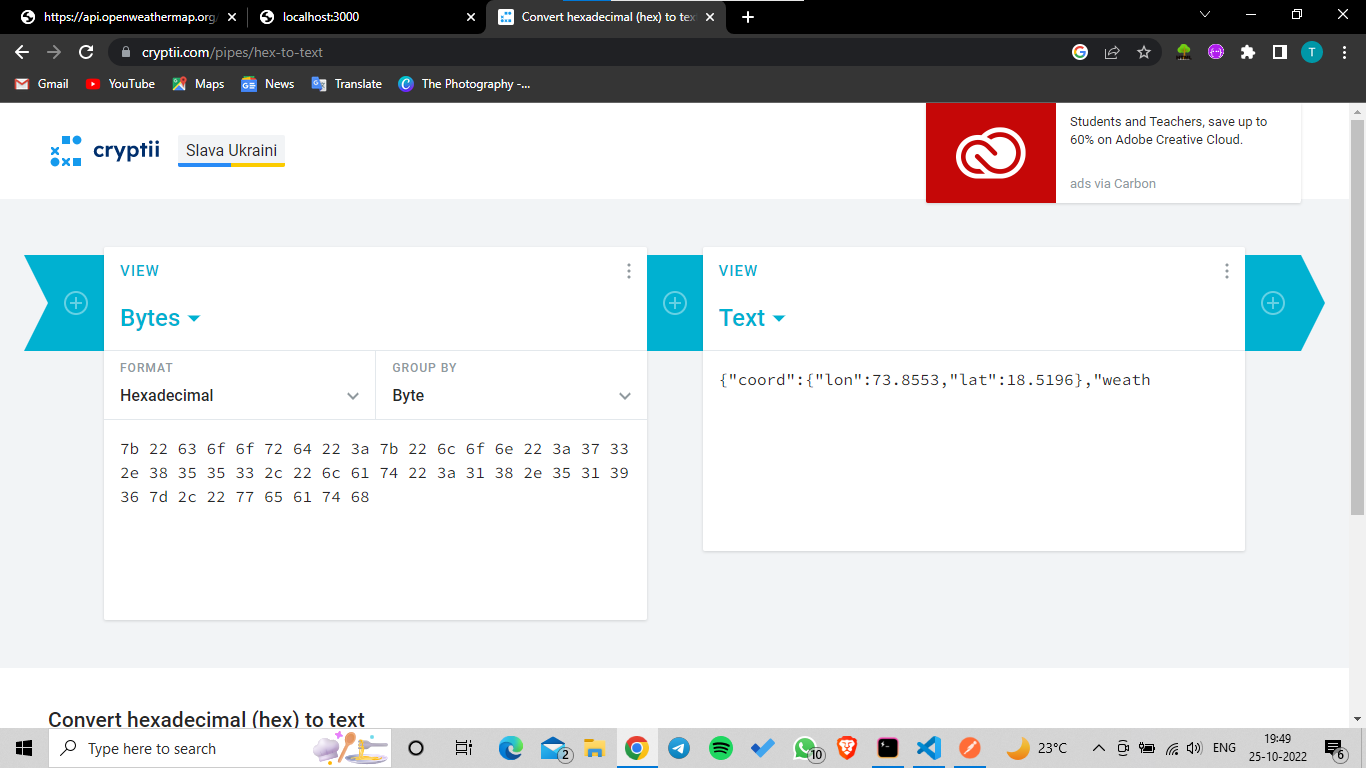
**JSON:(JavaScipt Object Notation)**

JSON is light weighted and easily understand through JavaScript objects.

Due to extension JSON Viewer we see the data in proper and understandable format.

Weather API

The data we got in hex code which we can understand and to it is converted text.



App.js For Weather Detection

const express = require("express");

const app = express();

const https = require("https");

const port = 3000;

app.get("/", function (req, res) {

  https.get(

    "https://api.openweathermap.org/data/2.5/weather?q=Pune&appid=706a659514f1c8e26247274e21412ad1&units=metric",

    (response) => {

      console.log(response.statusCode);

      response.on("data", (data) => {

        const weatherData = JSON.parse(data); //converts the data in format of JSON object

        //A common use of JSON is to exchange data to / from a web server.

        // When receiving data from a web server, the data is always a string.

        // Parse the data with JSON.parse(), and the data becomes a JavaScript object.

        console.log(weatherData);

        const temp1 = JSON.stringify(weatherData);

console.log(temp1); //JSON Java Object is

converted into String //typed later on update

        const temp = weatherData.main.temp;

        console.log(temp);

        const feelTemperature = weatherData.main.feels\_like;

        console.log(feelTemperature);

        const WeatherDescription = weatherData.weather[0].description;

        console.log(WeatherDescription);

      });

    }

  ); // if we have any typing mistake in endpoints it will return 404 .

  //if we have any typing mistake in appid parameters will return 401 i.e unauthorized user accessed  .

  res.send("Server is running ");

});

// app.use();

app.listen(port, (req, res) => {

  console.log("Server is running at port " + port);

});

O/P:

Text

Description automatically generated

For taking I/P From the html form and any other.

const bodyParser = require(“body-parser”);

app.use(bodyParser.urlencoder({extended:true}));

const https = require(“https”);

https allow use fetch the data from the external server i.e external server API.

https.get(“url”,function(req,res){

all content as per above example});

JSON gives the data in JavaScript Object format.

JSON.parse() Changes the data in the JSON Format i.e Object Oriented Format.

JSON.stringfy():Changes the JSON formatted data into String.or raw formated

res.sendFile(\_\_dirname,”/htmlFilename”);//get the data

res.send(): can we used one time.

Res.post() is used to fetch the data from method post in the form.

Const a = req.body.nameAttributeInInput.

console.log(response.statusCode);

to get the status that it is working properly or not.

Response.on() research

App.js

const express = require("express");

const app = express();

const bodyParser = require("body-parser");

app.use(bodyParser.urlencoded({ extended: true }));

const https = require("https");

const port = 3000;

app.get("/", function (req, res) {

  res.sendFile(\_\_dirname + "/index.html");

});

//for taking value from html elements with name attribute

app.post("/", (req, res) => {

  const query = req.body.CityName;

  const appid = "706a659514f1c8e26247274e21412ad1";

  const units = "metric";

  https.get(

    "https://api.openweathermap.org/data/2.5/weather?q=" +

      query +

      "&appid=" +

      appid +

      "&units=" +

      units,

    (response) => {

      console.log(response.statusCode);

      // if we have any typing mistake in endpoints it will return 404 .

      // if we have any typing mistake in appid parameters will return 401 i.e unauthorized user accessed  .

      response.on("data", (data) => {

        const weatherData = JSON.parse(data); //converts the data in format of JSON object

        // A common use of JSON is to exchange data to / from a web server.

        // When receiving data from a web server, the data is always a string.

        // Parse the data with JSON.parse(), and the data becomes a JavaScript object.

        //console.log(weatherData);

        const temp1 = JSON.stringify(weatherData); //JSON Java Object is converted into String.

        //console.log(temp1);

        const temp = weatherData.main.temp;

        //console.log(temp);

        const feelTemperature = weatherData.main.feels\_like;

        //console.log(feelTemperature);

        const WeatherDescription = weatherData.weather[0].description;

        //console.log(WeatherDescription);

        const name = weatherData.name;

        const icon = weatherData.weather[0].icon;

        const url = "http://openweathermap.org/img/wn/" + icon + "@2x.png";

        // res.write("<p1>" + temp1 + "</p1>");

        res.write(

          "<h1>" +

            "The temperature at " +

            name +

            " is " +

            temp +

            " Degree Celcius " +

            " and feels like " +

            feelTemperature +

            " Degree Celcius " +

            " weather is like " +

            WeatherDescription +

            "</h1>"

        );

        res.write("<img src=" + url + ">");

        res.send();

      });

    }

  );

});

app.listen(port, (req, res) => {

  console.log("Server is running at port " + port);

});

// const query = "Amaravati";

// const appid = "706a659514f1c8e26247274e21412ad1";

// const units = "metric";

// https.get(

//   "https://api.openweathermap.org/data/2.5/weather?q=" +

//     query +

//     "&appid=" +

//     appid +

//     "&units=" +

//     units,

//   (response) => {

//     console.log(response.statusCode);

// if we have any typing mistake in endpoints it will return 404 .

//if we have any typing mistake in appid parameters will return 401 i.e unauthorized user accessed  .

//     response.on("data", (data) => {

//       const weatherData = JSON.parse(data); //converts the data in format of JSON object

//A common use of JSON is to exchange data to / from a web server.

// When receiving data from a web server, the data is always a string.

// Parse the data with JSON.parse(), and the data becomes a JavaScript object.

//       console.log(weatherData);

//       const temp1 = JSON.stringify(weatherData); //JSON Java Object is converted into String.

// console.log(temp1);

//       const temp = weatherData.main.temp;

// console.log(temp);

//       const feelTemperature = weatherData.main.feels\_like;

// console.log(feelTemperature);

//       const WeatherDescription = weatherData.weather[0].description;

// console.log(WeatherDescription);

//       const name = weatherData.name;

//       const icon = weatherData.weather[0].icon;

//       const url = "http://openweathermap.org/img/wn/" + icon + "@2x.png";

//       // res.write("<p1>" + temp1 + "</p1>");

//       res.write(

//         "<h1>" +

//           "The temperature at " +

//           name +

//           " is " +

//           temp +

//           " Degree Celcius " +

//           " and feels like " +

//           feelTemperature +

//           " Degree Celcius " +

//           " weather is like " +

//           WeatherDescription +

//           "</h1>"

//       );

//       res.write("<img src=" + url + ">");

//       res.send();

//     });

//   }

// );

// app.use();

Index.html

|  |  |
| --- | --- |
|  | <!DOCTYPE html> |
|  | <html lang="en"> |
|  | <head> |
|  | <meta charset="UTF-8" /> |
|  | <meta http-equiv="X-UA-Compatible" content="IE=edge" /> |
|  | <meta name="viewport" content="width=device-width, initial-scale=1.0" /> |
|  | <title>Weather Project</title> |
|  | </head> |
|  | <body> |
|  | <form action="/" method="post"> |
|  | <label for="cityInput">City Name: </label |
|  | ><input id="cityInput" type="text" name="CityName" /> |
|  | <!-- <label for="units">Units:</label> |
|  | <input id="units" type="text" name="metric" /> --> |
|  | <button type="submit" value="submit">Evaluate</button> |
|  | </form> |
|  | </body> |
|  | </html> |
|  |  |