1. **Develop a web server with following functionalities:**

**- Serve static resources.**

const http = require('http');

const url = require('url');

const static= require('node-static');

const fileserver = new static.Server('./server.js')

var server = http.createServer(function(req, res){

    req.addListener('end', function(){

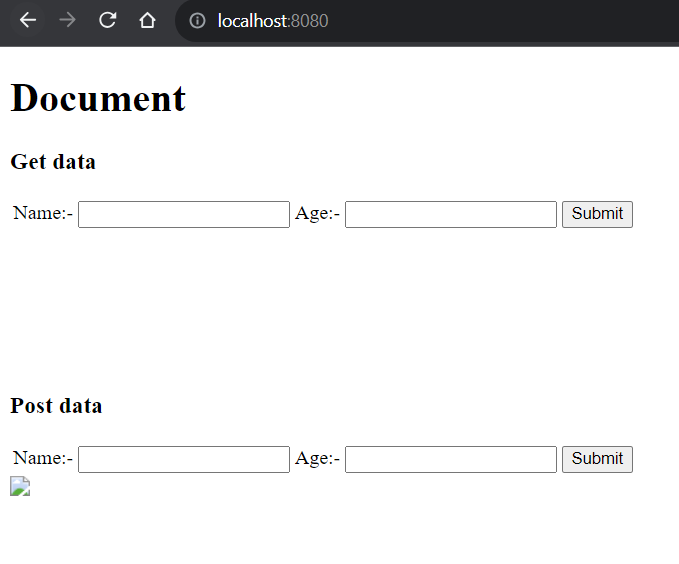
        fileserver.serve(req, res);

    }).resume();

}).listen(8080);

console.log("Listening on port number 1010");

Output : -



**-Handle GET request.**

var http = require('http')

var fs = require('fs')

const url = require('url');

var server = http.createServer(function(req,res){

    console.log("recived url" + req.url);

var u1 = url.parse(req.url,true);

    if(req.url=="/")

    {

        res.write("hello");

        res.write("hello1");

        res.end();

    }

    else if(req.url=="/list")

    {

        res.write("List");

        res.end();

    }

    else if (u1.pathname=="/process" && req.method === 'GET')

    {

        res.write(u1.query.name+" "+u1.query.age)

        res.end();

    }

    else if(req.url=="/index2.html" && req.method== 'GET')

    {

        var filename = "./index2.html";

        fs.readFile(filename,function(err,data){

            if (err) {

                res.writeHead(404,{'Content-type' : 'text/html'});

                return res.end("404 not found");

            }

            res.writeHead(200,{'Content-type' : 'text/html'});

            res.write(data);

            return res.end();

        });

    }

    else {

        res.write("other pages");

        res.end();

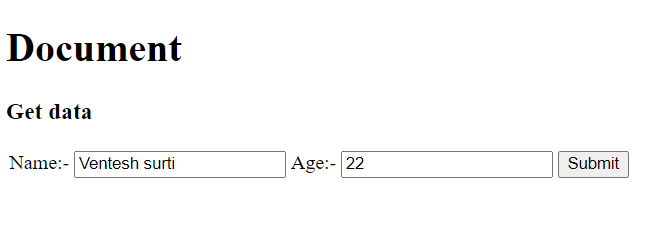
    }

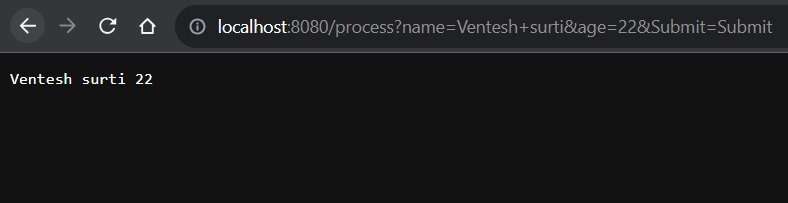
});

server.listen(8080);

console.log("server listing on 8080:");

Output : -





**- Handle POST request.**

var http = require('http')

var fs = require('fs')

const url = require('url');

var server = http.createServer(function(req,res){

    console.log("recived url" + req.url);

    var u1 = url.parse(req.url,true);

    if(req.url=="/")

    {

        res.write("hello");

        res.write("hello1");

        res.end();

    }

    else if(req.url=="/list")

    {

        res.write("List");

        res.end();

    }

    else if(req.url=="/index2.html" && req.method== 'GET')

    {

        var filename = "./index2.html";

        fs.readFile(filename,function(err,data){

            if (err) {

                res.writeHead(404,{'Content-type' : 'text/html'});

                return res.end("404 not found");

            }

            res.writeHead(200,{'Content-type' : 'text/html'});

            res.write(data);

            return res.end();

        });

    }

    else if(req.url=="/process" && req.method == 'POST')

    {

        let body = '';

        req.on('data', chunk => {

            body+= chunk.toString();

        });

        req.on('end', () => {

            console.log(body);

            res.end("ok => "+body);

        });

    }

    else {

        res.write("other pages");

        res.end();

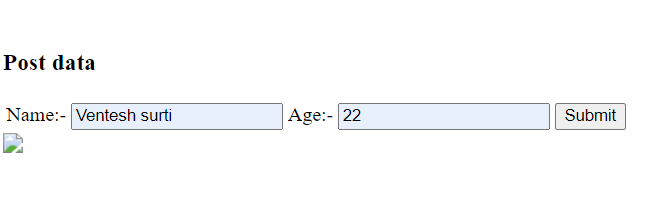
    }

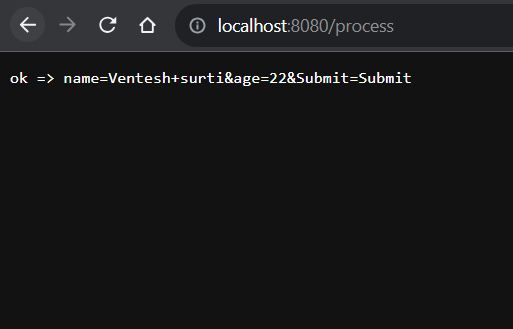
});

server.listen(8080);

console.log("server listing on 8080:");

Output : -





1. **Develop nodejs application with following requirements:**

* **Develop a route "/gethello" with GET method. It displays "Hello NodeJS!!" as response.**

const express = require('express');

const app = express();

const path = require('path');

*// Set up a route for "/gethello" with GET method*

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

  res.send('Hello NodeJS!!');

});

*// Serve the HTML file*

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

  res.sendFile(path.join(\_\_dirname, 'index.html'));

});

*// Start the server*

const port = 8080;

app.listen(port, () => {

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

});

**- Make an HTML page and display.**

**- Call "/gethello" route from HTML page using AJAX call. (Any frontend AJAX call API can be**

**used.)**

<!DOCTYPE html>

<html>

<head>

  <title>AJAX Call Demo</title>

</head>

<body>

  <h1>AJAX Call Demo</h1>

  <button onclick="getHello()">Get Hello</button>

  <div id="result"></div>

  <script>

    function getHello() {

*// Make an AJAX call to the "/gethello" route*

      fetch('/gethello')

        .then(response => response.text())

        .then(data => {

*// Display the response in the "result" div*

          document.getElementById('result').textContent = data;

        })

        .catch(error => {

          console.error('Error:', error);

        });

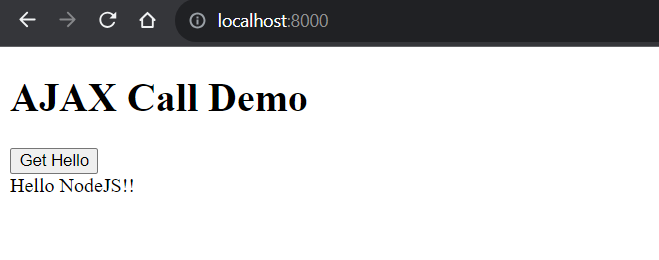
    }

  </script>

</body>

</html>

Output :-



**3. Develop a module for domain specific chatbot and use it in a command line application.**

**APP.js :-**

const { CLIENT\_RENEG\_LIMIT } = require("tls");

var Chatbot = require("./chatbot");

var readline = require("readline");

var rt = readline.createInterface(process.stdin, process.stdout);

rt.setPrompt("You==>");

rt.prompt();

rt.on('line', function (message) {

    console.log('Bot ==> ' + Chatbot.chatbotReply(message));

    rt.prompt();

}).on('close', function () {

    process.exit(0);

});

**Chatbot.js :-**

module.exports.chatbotReply = function(message){

*this*.Bot\_age = 22;

*this*.Bot\_Name = "name1";

*this*.Bot\_Universtiy = "VNSGU";

*this*.Bot\_Contry ="India";

    message = message.toLowerCase()

    if(message.indexOf("hi")  > -1 ||

                indexOf("hello") > -1 ||

                indexOf("welcome") > -1)

    {

                    return "Hi..!";

    } else if(message.indexOf("age") > -1 &&

              message.indexOf("your"))

    {

        return "I'm " + *this*.Bot\_age;

    }

    else if(message.indexOf("how") > -1 &&

            message.indexOf("are") &&

            message.indexOf("you"))

    {

        return "I'm fine ^\_^";

    }

    else if(message.indexOf("where") > -1 &&

            message.indexOf("live") &&

            message.indexOf("you"))

    {

        return "I live in  " + *this*.Bot\_Contry;

    }

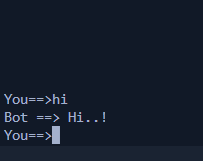
    else{

        return " Sorry, I did't get it :( ";

    }

}

Output :-



**4. Use above chatbot module in web based chatting of websocket.**

const WebSocket = require('ws')

var http = require('http')

var url = require('url')

var st = require('node-static')

var fileserver = new st.Server('./public')

var httpserver = http.createServer(function(request, response){

    request.on('end',function(){

        var get = url.parse(request.url, true).query;

        fileserver.serve(request,response);

    }).resume();

}).listen(8080,function(){

    console.log((new Date()) +

       'server listening on port 8080');

});

const wss = new WebSocket.Server({server: httpserver});

wss.on('connection', function(ws){

    ws.send('hello client')

    ws.on('message', message => {

    console.log('Received message => ${message}')

    ws.send('I received:' + message)

    })

})

-:> index.html page:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Web Socket</title>

    <script>

        var ws = new WebSocket('ws://localhost:8080');

        ws.addEventListener('message', function(e){

            var msg = e.data;

            document.getElementById('chatlog').innerHTML+='<br> Server: '+msg;

        });

        function sendMessage(){

            var message = document.getElementById('message').value;

            document.getElementById('chatlog').innerHTML+='<br> Me: '+ message;

            ws.send(message);

            document.getElementById('message').value = '';

        }

    </script>

</head>

<body>

    <h2>Data from Server</h2>

     <div id="chatlog"></div>

     <hr />

    <h2>Data from client</h2>

     <input type="text" id="message" />

     <input type="button" id="b1" onclick="sendMessage()" value="send" />

</body>

</html>

**Output :-**

****

**5. Write a program to create a compressed zip file for a folder.**

var fs = require('fs')

var zlib = require('zlib')

fs.createReadStream('./text1.txt')

    .pipe(zlib.createGzip())

    .pipe(fs.createWriteStream('./text.txt.gz'));

console.log('File compressed..!!');

**Output :-**

****

****

**6. Write a program to extract a zip file.**

var fs = require('fs')

var unzip = require('zlib')

fs.createReadStream('./text.txt.gz')

    .pipe(unzip.createGunzip())

    .pipe(fs.createWriteStream('./text1.txt'));

console.log('File Decompressed..!!');

**Output :-**





**7. Write a program to promisify fs.unlink function and call it.**

var fs = require('fs/promises')

function readFile(fpath)

{

    return new Promise(function(success,fail)

    {

        fs.unlink(fpath,(err,data) =>

        {

            if(err)

                fail(err)

            else

                success(data)

        })

    })

}

readFile('./text1.txt').then((data)=>{

    console.log(data)

}).catch((err)=>{

    console.log(err)

})

**Output :-**





****

**8. Fetch data of google page using note-fetch using async-await model.**

*//var fetch = require('node-fetch')*

const fetch = (...args) => import('node-fetch').then(({default: fetch}) => fetch(...args));

async function asyncajaxawait()

{

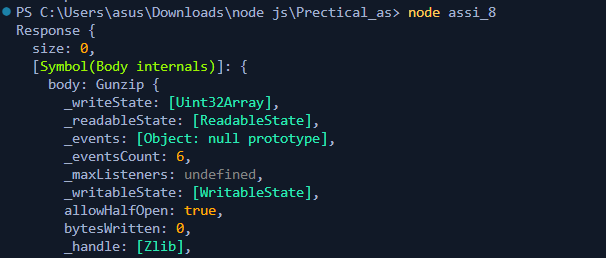
  const res = await fetch('https://www.google.com/')

  console.log(res);

}

asyncajaxawait();

**Output :-**

****

**9. Write a program that connect Mysql database, Insert a record in employee table and**

**display all records in employee table using promise based approach.**

const mysql = require('nodejs-mysql').default;

const config = {

    host: "localhost",

    user: "root",

    password: "",

    database: "employee "

}

const db = mysql.getInstance(config)

db.connect()

    .then(function(){

        console.log("Connected!");

        var sql = "INSERT INTO employe (username, password, firstname, lastname, email) VALUES ('USER', 'pass','fname1','lname1','a@b.com')";

        return db.exec(sql);

    }).then(function(res){

        console.log(res);

        return db.exec("SELECT \* FROM users");

    }).then(function(result){

        for (var i in result) {

            console.log('Username: ', result[i].username + " " +result[i].password);

        }

        process.exit(0);

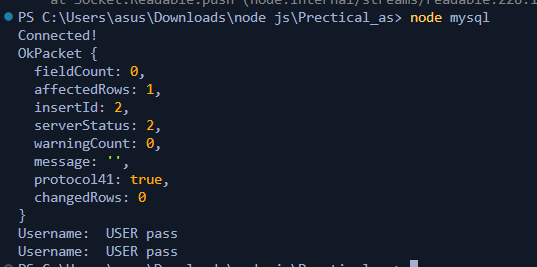
    }).catch(function(err){

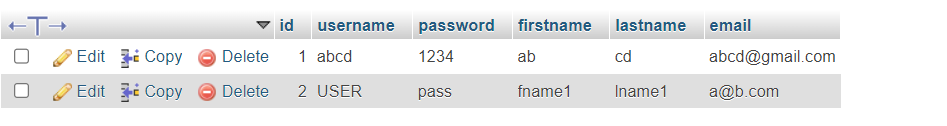
        console.log("Error");

        process.exit(0);

    })

**Output :-**

****

****

**10. Set a server script, a test script and 3 user defined scripts in package.json file in your nodejs**

**application.**

{

  "name": "node-js",

  "version": "1.0.0",

  "description": "",

  "main": "index.js",

  "scripts": {

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

    "start": "node server.js",

    "script1" : "node assi\_6.js",

    "script2" : "node assi\_7.js",

    "script3" : "node assi\_8.js"

  },

  "author": "",

  "license": "ISC",

  "dependencies": {

    "cheerio": "^1.0.0-rc.12",

    "cors": "^2.8.5",

    "express": "^4.18.2",

    "http": "^0.0.1-security",

    "node-fetch": "^3.3.2",

    "node-static": "^0.7.11",

    "nodejs-mysql": "^0.1.3",

    "request": "^2.88.2",

    "url": "^0.11.1",

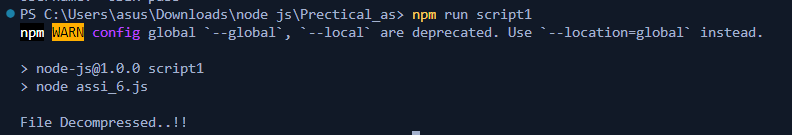
    "websocket": "^1.0.34",

    "ws": "^8.13.0"

  }

}

**Output :-**

****

**11. Develop an application to show live cricket score.**

const express = require('express');

const axios = require('axios');

const app = express();

const port = 3000;

const apiKey = 'd4594015-7cc4-4cd1-9817-610c4768246e';

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

  const cricapiUrl = `https://api.cricapi.com/v1/currentMatches?apikey=d4594015-7cc4-4cd1-9817-610c4768246e&offset=0`;

  axios.get(cricapiUrl)

    .then(response => {

      const liveMatches = response.data.matches.filter(match => match.matchStarted);

      const liveScores = liveMatches.map(match => {

        return {

          id: match.id,

          date: match.date,

          score: match.score,

        };

      });

      res.json(liveScores);

    })

    .catch(error => {

      console.error('Error fetching live scores:', error.message);

      res.status(500).send('Error fetching live scores.');

    });

});

app.listen(port, () => {

  console.log(`Live cricket score app is running on http://localhost:${port}`);

});