1.4 BUILD YOUR FIRST WEB SERVER

uilding your first web server is an exciting way to dive into web development. Here's a simple guide to help you set up a basic web server. For simplicity, we'll use **Node.js** and its built-in HTTP module, as it provides an easy way to get started.

Prerequisites

1. **Install Node.js**: Download and install Node.js. This will give you access to both node and npm (Node Package Manager).
2. **Code Editor**: Use an editor like VS Code

Steps to Create a Basic Web Server

1. **Create Your Project Directory**:

```
mkdir my-web-server

cd my-web-server
```

2. **Initialize a Node.js Project**:
   npm init -y

   This will create a package.json file with default settings.

3. **Create a JavaScript File**: Create a file named server.js in your project directory.

4. **Write Basic Server Code**: Open server.js in your editor and add the following code:

```
const http = require('http');


const hostname = '127.0.0.1'; // Localhost

const port = 3000;          // Port number


const server = http.createServer((req, res) => {
```

```
res.statusCode = 200; // HTTP status: OK
res.setHeader('Content-Type', 'text/html');
res.end('<h1>Hello, World!</h1>');
});

server.listen(port, hostname, () => {
console.log(`Server running at http://${hostname}:${port}/`);
});
```

5. **Run Your Server**: In your terminal, run

node server.js

Server running at http://127.0.0.1:3000/

6. **Access the Server**: Open your browser and go to http://127.0.0.1:3000/.

You should see "Hello, World!" displayed

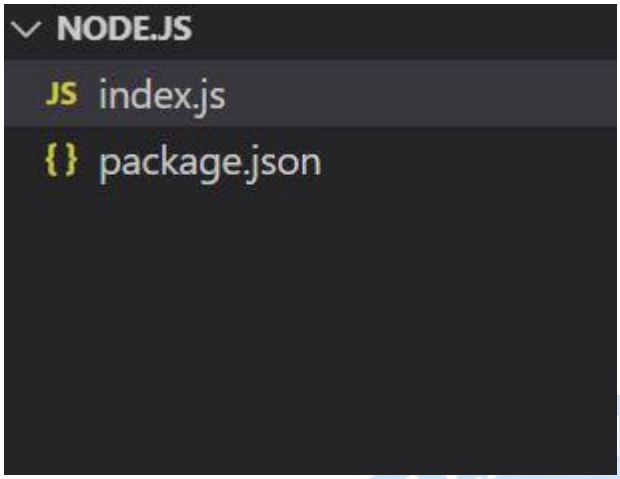
Using Built-in HTTP module

HTTP and HTTPS, these two inbuilt modules are used to create a simple server. The HTTPS module provides the feature of the encryption of communication with the help of the secure layer feature of this module. Whereas the HTTP module doesn't provide the encryption of the data.

Approach

Building a simple Node.js web server with the http module by using http.createServer(), which listens for requests, sends responses, and is ideal for understanding core server functionality.

Project structure: It will look like this.



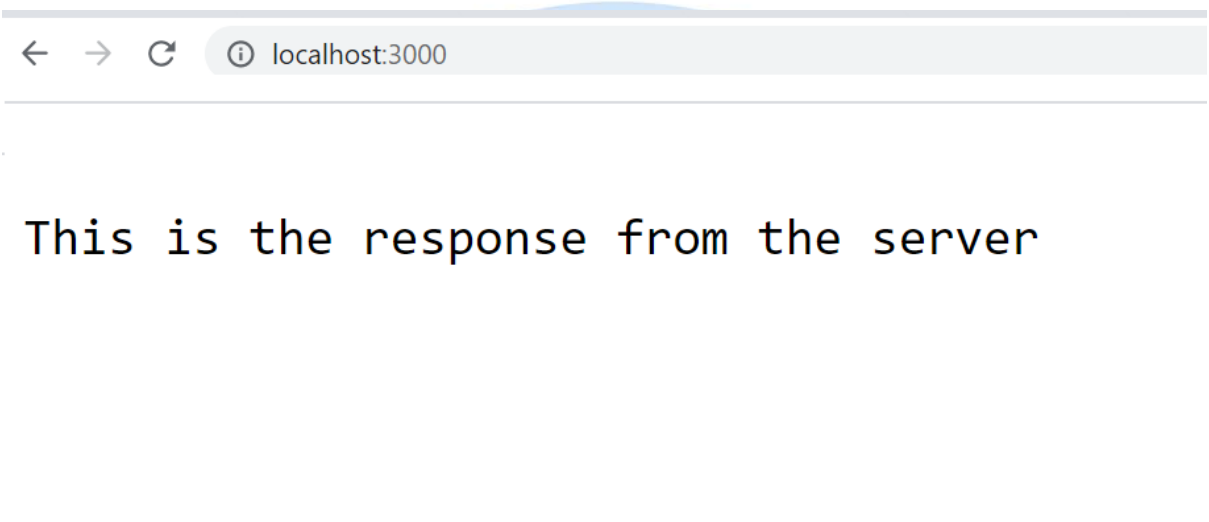

```
// Filename - index.js

// Importing the http module
const http = require("http")

// Creating server
const server = http.createServer((req, res) => {
    // Sending the response
    res.write("This is the response from the server")
    res.end();
})

// Server listening to port 3000
server.listen((3000), () => {
    console.log("Server is Running");
})
```

Run **index.js** file using below command:

```
node index.js
```

```
lenovo@LAPTOP-OBEPNKMU MINGW64 ~/Desktop/Node.js
$ node index.js
Server is Running
```

utput: Now open your browser and go to *http://localhost:3000/*, you will see the following output:



Using Express Module

The *express.js* is one of the most powerful frameworks of the node.js that works on the upper layer of the http module. The main advantage of using *express.js* server is filtering the incoming requests by clients.

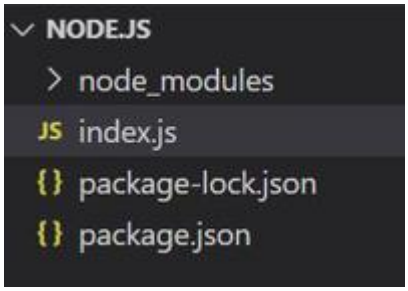
Approach

To create a web server with Express initialize an app with express(), defining routes with app.get(), and sending responses using res.send(). Express simplifies development with built-in features and middleware.

Installing module: Install the required module using the following command.

npm install express

Project structure: It will look like this.



Example:This example demonstrates creating a simple web server using express.js

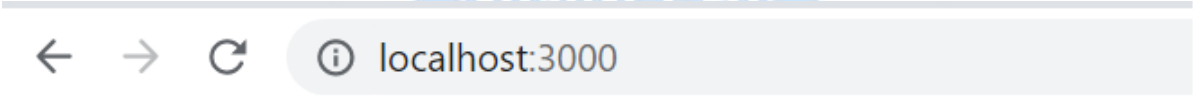
```
// Filename - index.js


// Importing express module

const express = require("express")

const app = express()


// Handling GET / request

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

    res.send("This is the express server")

})


// Handling GET /hello request

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

    res.send("This is the hello response");

})


// Server setup

app.listen(3000, () => {
```

```
    console.log("Server is Running")
})
```

Run the index.js file using the below command:

node index.js

Output: Now open your browser and go to *http://localhost:3000/*, you will see the following output:



This is the express server