

Lesson 01 Demo 02

Creating HTML Elements for JSON Server

Objective: To create an HTML file that displays a table with data from a JSON server

Tools required: Visual Studio Code and json-server

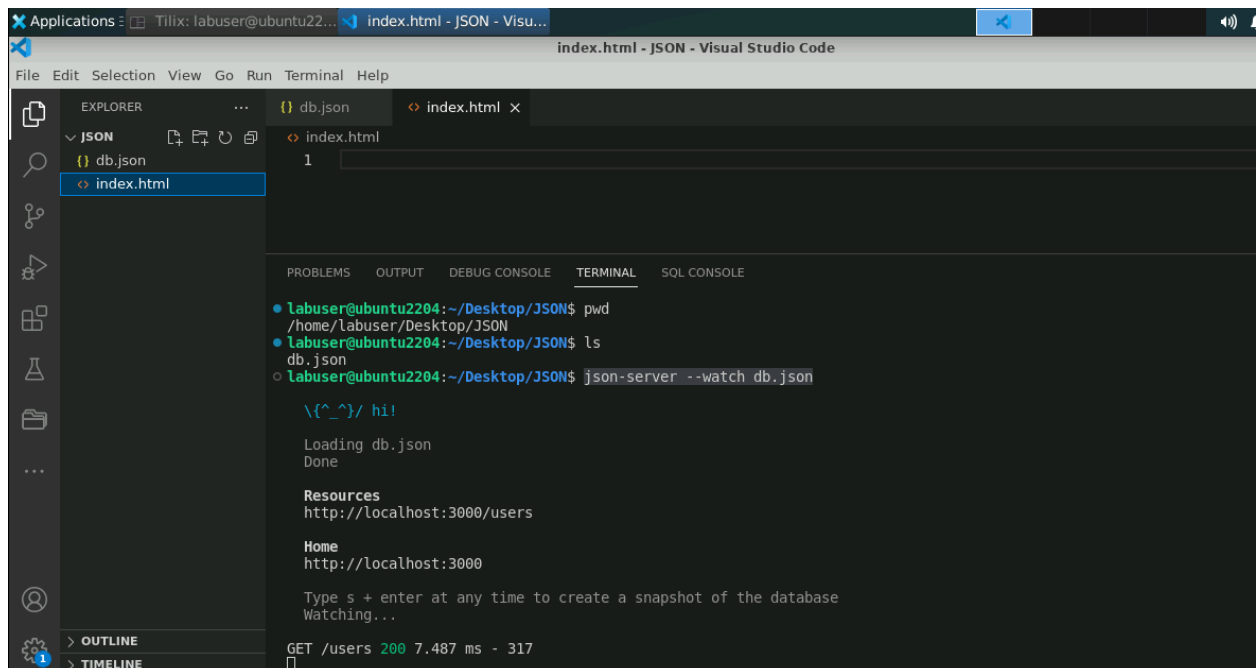
Prerequisites: Perform Lesson 01 Demo 01 and ensure **db.json** is active and running on **localhost:3000**

Steps to be followed:

1. Create an HTML file

Step 1: Create an HTML file

1.1 Open the Visual Studio Code editor and create a file named **index.html**



The screenshot shows the Visual Studio Code editor interface. The Explorer panel on the left shows a project named 'JSON' with two files: 'db.json' and 'index.html'. The 'index.html' file is selected and open in the editor, showing a single line of code: `1`. The Terminal panel at the bottom shows the following commands and output:

```
labuser@ubuntu2204:~/Desktop/JSON$ pwd
/home/labuser/Desktop/JSON
labuser@ubuntu2204:~/Desktop/JSON$ ls
db.json
labuser@ubuntu2204:~/Desktop/JSON$ json-server -w db.json

\(^_)/ hi!

Loading db.json
Done

Resources
http://localhost:3000/users

Home
http://localhost:3000

Type s + enter at any time to create a snapshot of the database
Watching...

GET /users 200 7.487 ms - 317
```

1.2 Add the following HTML and CSS code to the file:

```
<!DOCTYPE html>
<html>
<head>
  <title>JSON Server Demo</title>
  <style>
    table, th, td {
      border: 1px solid black;
      border-collapse: collapse;
    }
    th, td {
      padding: 10px;
    }
  </style>
</head>
<body>
  <h1>JSON Server Demo</h1>
  <table id="user-table">
    <thead>
      <tr>
        <th>ID</th>
        <th>Name</th>
        <th>Email</th>
        <th>Phone</th>
      </tr>
    </thead>
    <tbody>
      <!-- The table body will be populated with the data from the JSON server -->
    </tbody>
  </table>
<script>
</script>
</body>
</html>
```

```

db.json  index.html x
index.html > html
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>JSON Server Demo</title>
5      <style>
6          table,
7          th,
8          td {
9              border: 1px solid black;
10             border-collapse: collapse;
11         }
12
13         th,
14         td {
15             padding: 10px;
16         }
17     </style>
18 </head>
19
20 <body>
21     <h1>JSON Server Demo</h1>
22     <table id="user-table">
23         <thead>
24             <tr>
25                 <th>ID</th>
26                 <th>Name</th>
27                 <th>Email</th>
28                 <th>Phone</th>
29             </tr>
30         <tbody>
31             <!-- The table body will be populated with the data from the JSON server -->
32         </tbody>
33     </table>
34     <script>
35     </script>
36 </body>
37 </html>

```

1.3 Add the following JavaScript code within the `<script>` tag in the file:

```

<script>
    // Use the fetch API to get the data from the JSON server
    fetch('http://localhost:3000/users')
    .then(response => response.json()) // Convert the response to JSON
    .then(data => {
        // Get the table body element
        let tbody = document.getElementById('user-
table').getElementsByTagName('tbody')[0];
        // Loop through the data array
        for (let user of data) {
            // Create a new table row element
            let tr = document.createElement('tr');

```

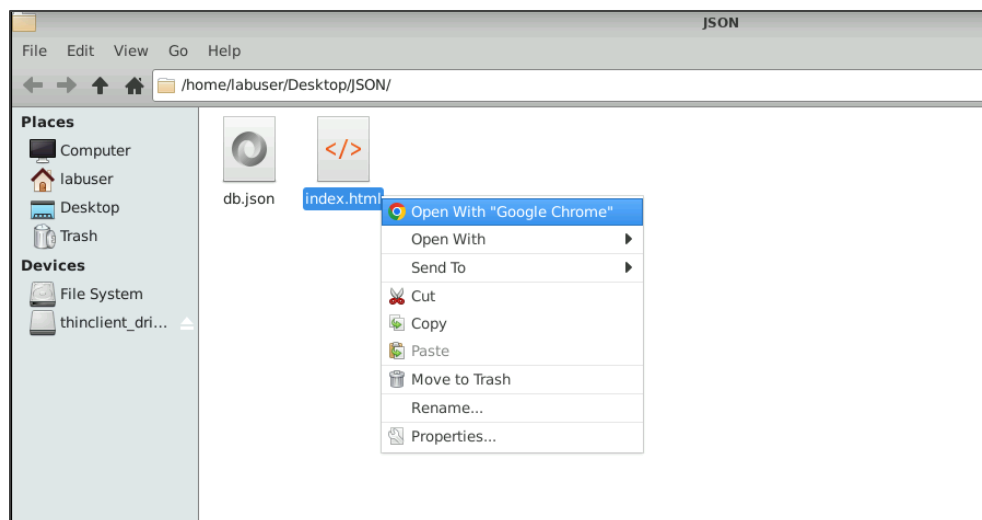
```
// Create four table cell elements for each user property
let td_id = document.createElement('td');
let td_name = document.createElement('td');
let td_email = document.createElement('td');
let td_phone = document.createElement('td');
// Set the text content of each table cell to the user property value
td_id.textContent = user.id;
td_name.textContent = user.name;
td_email.textContent = user.email;
td_phone.textContent = user.phone;
// Append the table cells to the table row
tr.appendChild(td_id);
tr.appendChild(td_name);
tr.appendChild(td_email);
tr.appendChild(td_phone);
// Append the table row to the table body
tbody.appendChild(tr);
}
})
.catch(error => {
  // Handle any errors
  console.error(error);
});
</script>
```

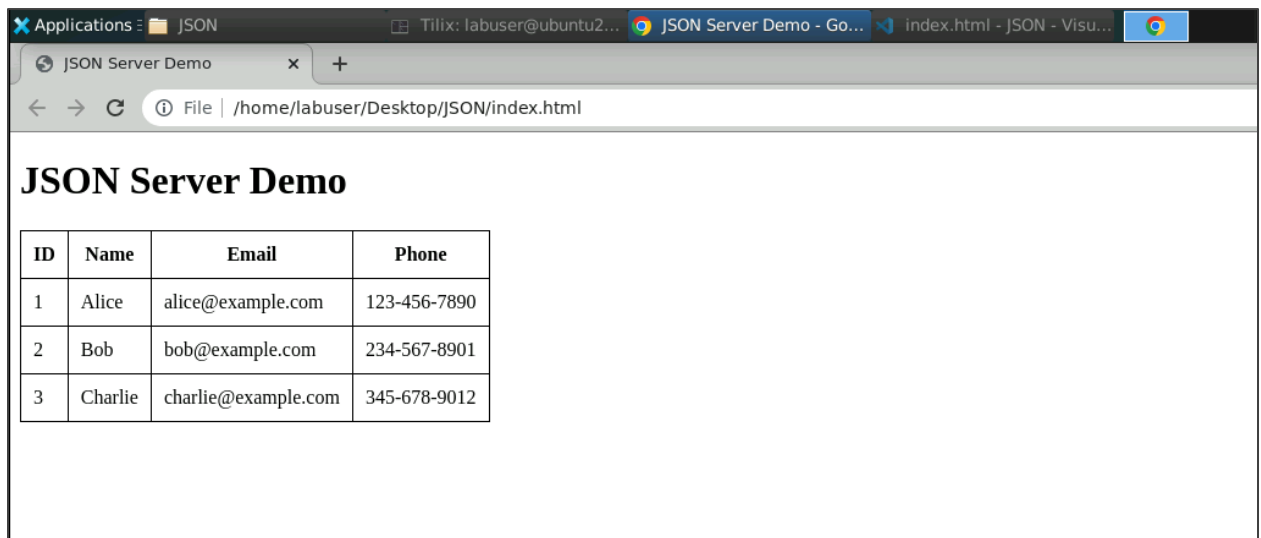
```

db.json  index.html X
index.html > html > body > script > then() callback
34 <script>
35 // Use the fetch API to get the data from the JSON server
36 fetch('http://localhost:3000/users')
37 .then(response => response.json()) // Convert the response to JSON
38 .then(data => {
39 // Get the table body element
40 let tbody = document.getElementById('user-table').getElementsByTagName('tbody')[0];
41 // Loop through the data array
42 for (let user of data) {
43 // Create a new table row element
44 let tr = document.createElement('tr');
45 // Create four table cell elements for each user property
46 let td_id = document.createElement('td');
47 let td_name = document.createElement('td');
48 let td_email = document.createElement('td');
49 let td_phone = document.createElement('td');
50 // Set the text content of each table cell to the user property value
51 td_id.textContent = user.id;
52 td_name.textContent = user.name;
53 td_email.textContent = user.email;
54 td_phone.textContent = user.phone;
55 // Append the table cells to the table row
56 tr.appendChild(td_id);
57 tr.appendChild(td_name);
58 tr.appendChild(td_email);
59 tr.appendChild(td_phone);
60 // Append the table row to the table body
61 tbody.appendChild(tr);
62 }
63 }
64 .catch(error => {
65 // Handle any errors
66 console.error(error);
67 });
68 </script>
69 </body>
70 </html>

```

1.4 Save and open the file in a web browser to see the result





You should see a table with data fetched from the JSON server.

By following these steps, you have successfully created an HTML file that fetches and displays user data from a JSON server in a table format, combining HTML, CSS, and JavaScript to interact with a REST API.