

## Lesson 05 Demo 05

### Implementing AJAX Calls

**Objective:** To demonstrate the implementation of AJAX calls using XMLHttpRequest and Fetch API for real-time data retrieval and efficient asynchronous operations using promises in web development

**Tools required:** Visual Studio Code and Node.js

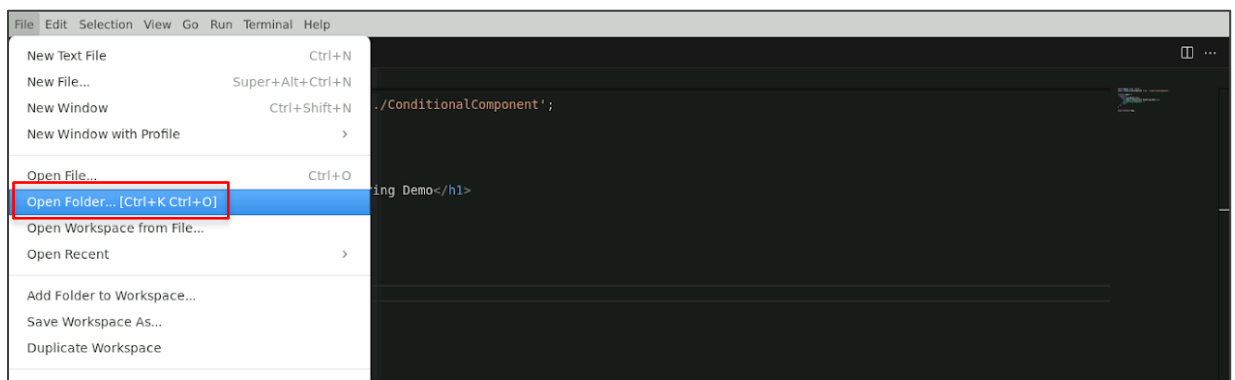
**Prerequisites:** None

Steps to be followed:

1. Write code for AJAX
2. Execute and verify the working of AJAX calls

#### Step 1: Write code for AJAX

- 1.1 Navigate to Visual Studio Code, right-click on the **File** menu of the code editor, and select the **Open Folder** option



- 1.2 Click on the **src** folder of the project, select the **New File** option, and enter the filename as **ajax\_demo1.html**



1.3 Write the given code in `ajax_demo1.html`:

```
<html>
<body>
  <header>
    <h1>MEAN Stack</h1>
    <p> Ajax without Fetch and Promise</p>
  </header>
  <script language="javascript" type="text/javascript">
    function ajax_call_demo( ){
      if (window.XMLHttpRequest ) {
        {
          xhttp = new XMLHttpRequest();
        }
      } else {
        alert("Your browser does not support XMLHttpRequest...!");
      }
      xhttp.open("GET", "https://images.pexels.com/photos/853168/pexels-photo-853168.jpeg?auto=compress&cs=tinysrgb&dpr=1&w=500", true); // Make sure file is
in same server
      xhttp.overrideMimeType('text/plain; charset=x-user-defined');
      xhttp.send(null);
      xhttp.onreadystatechange = function() {
        if (xhttp.readyState == 4){
          if ((xhttp.status == 200) || (xhttp.status == 0)){
            var image = document.getElementById("get_img");
            image.src = "data:image/gif;base64," + encode64(xhttp.responseText);
          }else{
            alert("Something misconfiguration : " +
              "\nError Code : " + xhttp.status +
              "\nError Message : " + xhttp.responseText);
          }
        }
      };
    }
    function encode64(inputStr){
      var b64 =
      "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=";
```

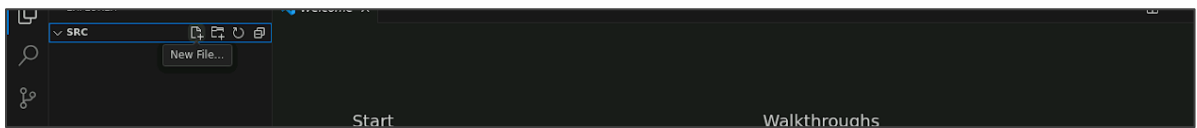
```

var outputStr = "";
var i = 0;
while (i<inputStr.length){
    var byte1 = inputStr.charCodeAt(i++) & 0xff;
    var byte2 = inputStr.charCodeAt(i++) & 0xff;
    var byte3 = inputStr.charCodeAt(i++) & 0xff;
    var enc1 = byte1 >> 2;
    var enc2 = ((byte1 & 3) << 4) | (byte2 >> 4);
    var enc3, enc4;
    if (isNaN(byte2)){
        enc3 = enc4 = 64;
    } else{
        enc3 = ((byte2 & 15) << 2) | (byte3 >> 6);
        if (isNaN(byte3)){
            enc4 = 64;
        } else {
            enc4 = byte3 & 63;
        }
    }
    outputStr += b64.charAt(enc1) + b64.charAt(enc2) + b64.charAt(enc3) +
b64.charAt(enc4);
}
return outputStr;
}
</script>
<button onClick="ajax_call_demo()">Click here to get an image</button><br />
<img id="get_img" />
</body>
</html>

```

```
ajax_demo1.html X
ajax_demo1.html > ...
1 <html>
2 <body>
3
4     <header>
5         <h1>MEAN Stack</h1>
6         <p> Ajax without Fetch and Promise</p>
7     </header>
8
9     <script language="javascript" type="text/javascript">
10         function ajax_call_demo() {
11             if (window.XMLHttpRequest) {
12                 {
13                     xhttp = new XMLHttpRequest();
14                 }
15             } else {
16                 alert("Your browser does not support XMLHttpRequest...!");
17             }
18
19             xhttp.open("GET", "https://images.pexels.com/photos/853168/pexels-photo-853168.jpeg?auto=compress&cs=tinysrgb&dpr=1&w=500", true); // Ma
20             xhttp.overrideMimeType('text/plain; charset=x-user-defined');
21             xhttp.send(null);
22
23             xhttp.onreadystatechange = function() {
24                 if (xhttp.readyState == 4){
25                     if ((xhttp.status == 200) || (xhttp.status == 0)){
26                         var image = document.getElementById("get_img");
27                         image.src = "data:image/gif;base64 " + encode64(xhttp.responseText);
```

1.4 Click on the **src** folder of the project, select the **New File** option, enter the filename as **index.js**, and write the code shown below:



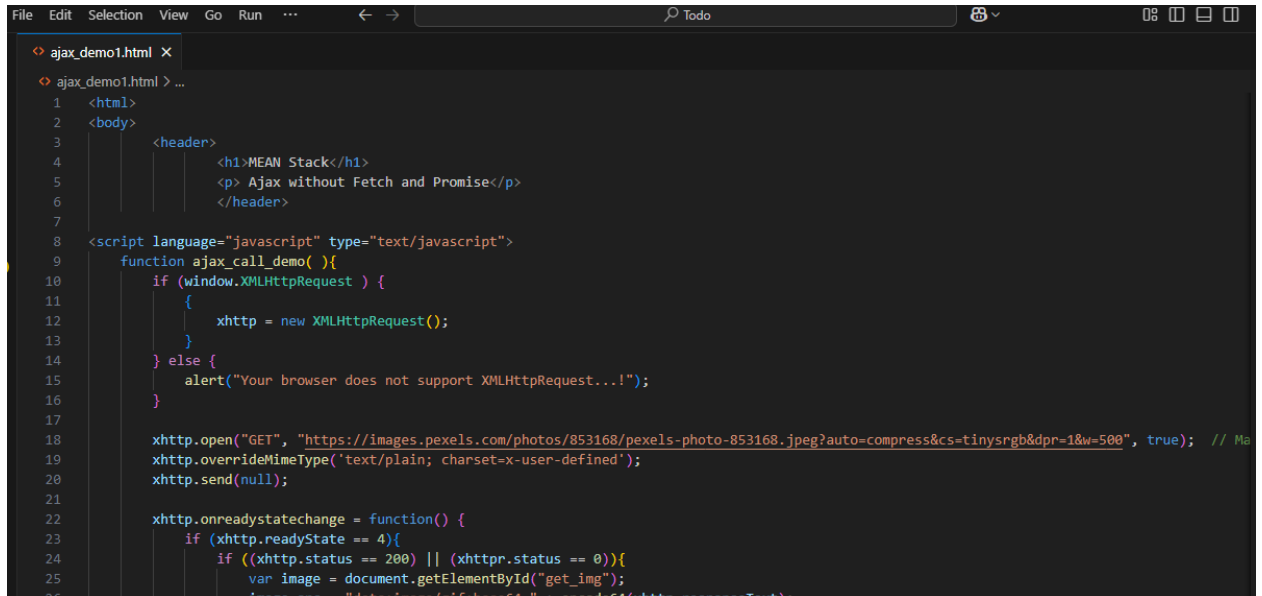
1.5 Write the code shown below in **index.js**:

```
const URL = "https://reqres.in/api/users";
fetch(URL)
.then((response) => response.json())
.then(res => console.log(res.data))
```

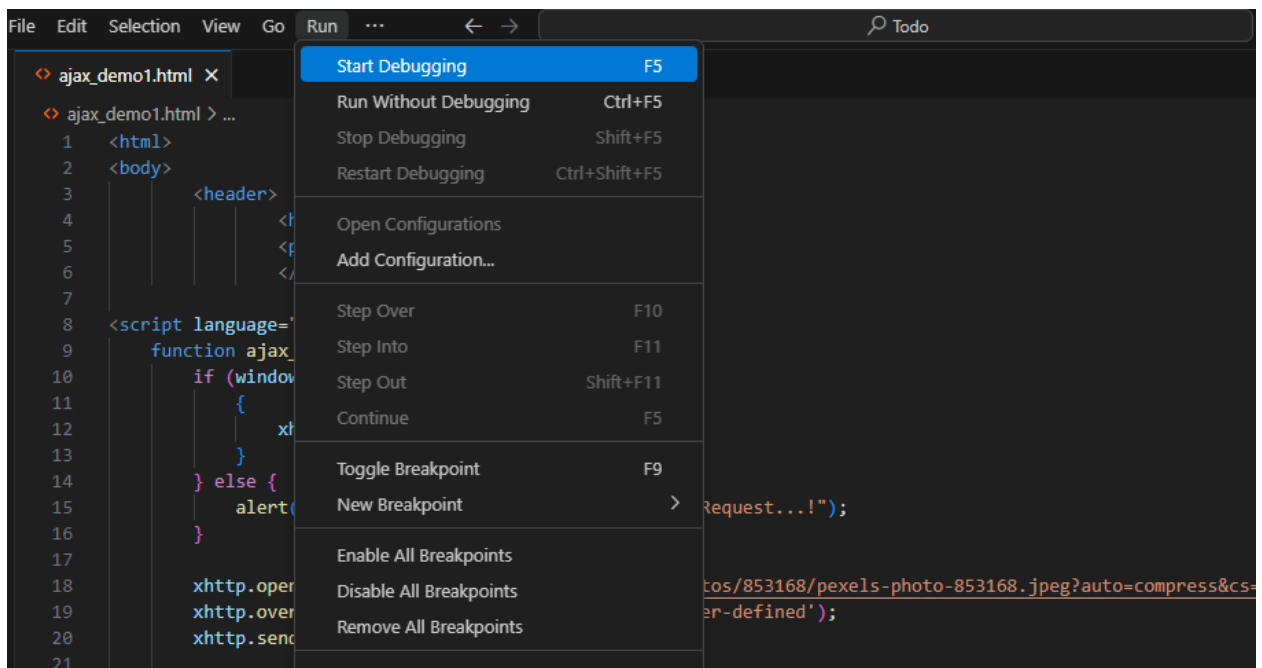
```
EXPLORER
DEMO 8
src
  ajax_demo1.html
  JS indexjs
src > JS indexjs > ...
1 const URL = "https://reqres.in/api/users";
2
3 fetch(URL)
4 .then((response) => response.json())
5 .then(res => console.log(res.data))
6
7 |
```

## Step 2: Execute and verify the working of AJAX calls

### 2.1 Click on **Run** and then on **Start Debugging** to execute the JavaScript file



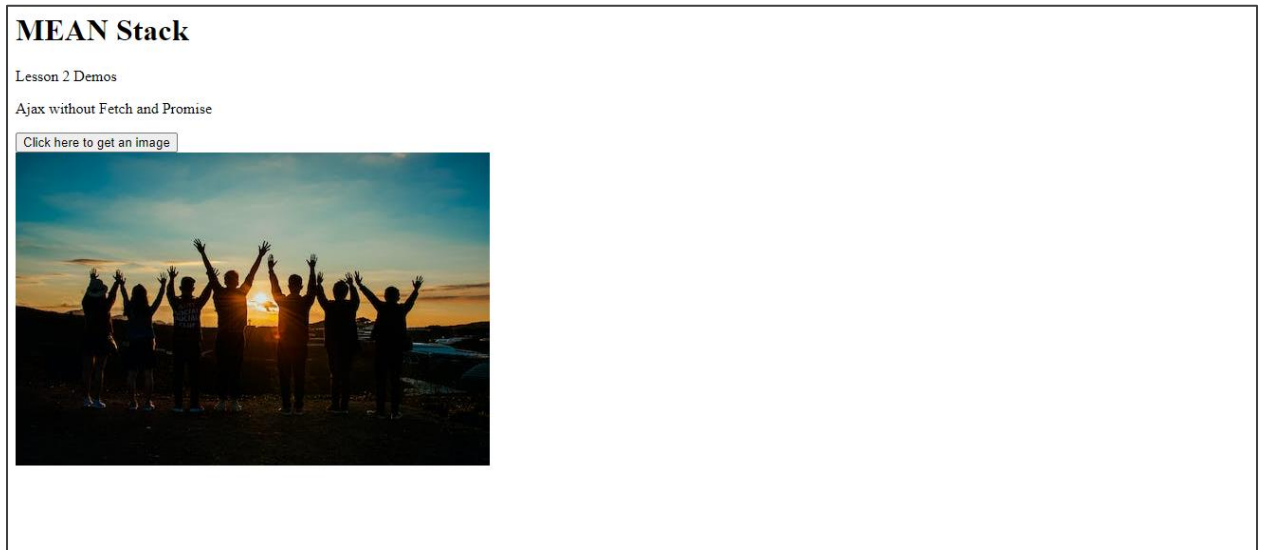
```
File Edit Selection View Go Run ... < -> Todo
ajax_demo1.html X
ajax_demo1.html > ...
1 <html>
2 <body>
3   <header>
4     <h1>MEAN Stack</h1>
5     <p> Ajax without Fetch and Promise</p>
6   </header>
7
8   <script language="javascript" type="text/javascript">
9     function ajax_call_demo() {
10      if (window.XMLHttpRequest) {
11        {
12          xmlhttp = new XMLHttpRequest();
13        }
14      } else {
15        alert("Your browser does not support XMLHttpRequest...!");
16      }
17
18      xmlhttp.open("GET", "https://images.pexels.com/photos/853168/pexels-photo-853168.jpeg?auto=compress&cs=tinysrgb&dpr=1&w=500", true); // Ma
19      xmlhttp.overrideMimeType('text/plain; charset=x-user-defined');
20      xmlhttp.send(null);
21
22      xmlhttp.onreadystatechange = function() {
23        if (xmlhttp.readyState == 4) {
24          if ((xmlhttp.status == 200) || (xmlhttp.status == 0)) {
25            var image = document.getElementById("get_img");
26            image.src = "data:image/gif;base64 " + encode64(xmlhttp.responseText);
```



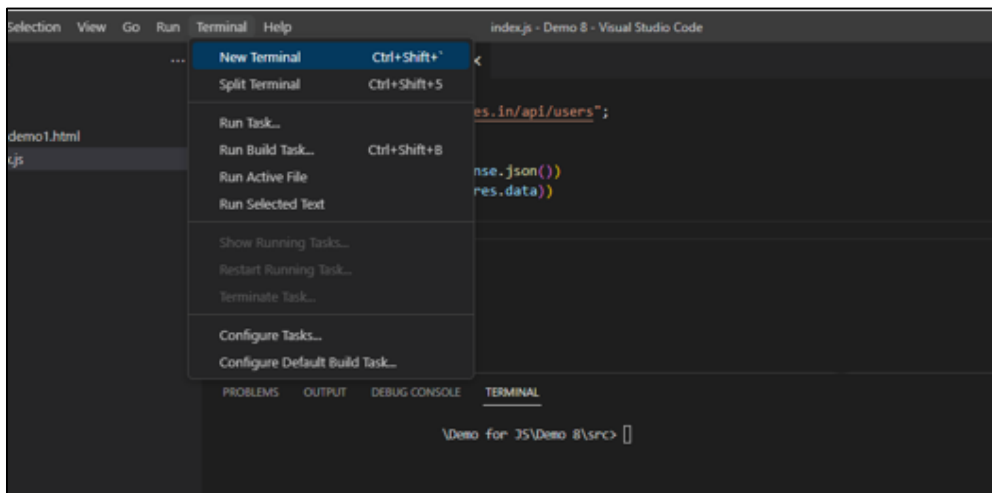
```
File Edit Selection View Go Run ... < -> Todo
ajax_demo1.html X
ajax_demo1.html > ...
1 <html>
2 <body>
3   <header>
4     <h1>MEAN Stack</h1>
5     <p> Ajax without Fetch and Promise</p>
6   </header>
7
8   <script language="
9     function ajax
10      if (window
11        {
12          xh
13        }
14      } else {
15        alert(
16      }
17
18      xmlhttp.oper
19      xmlhttp.over
20      xmlhttp.send
21
22      xmlhttp.onrea
23        if (xmlhttp.readyState == 4) {
24          if ((xmlhttp.status == 200) || (xmlhttp.status == 0)) {
25            var image = document.getElementById("get_img");
26            image.src = "data:image/gif;base64 " + encode64(xmlhttp.responseText);

Run and Debug menu:
Start Debugging F5
Run Without Debugging Ctrl+F5
Stop Debugging Shift+F5
Restart Debugging Ctrl+Shift+F5
Open Configurations
Add Configuration...
Step Over F10
Step Into F11
Step Out Shift+F11
Continue F5
Toggle Breakpoint F9
New Breakpoint >
Enable All Breakpoints
Disable All Breakpoints
Remove All Breakpoints
```

2.2 When the server starts running, right-click and select the **Inspect Element** option and click on the **Console** tab




2.3 Click on **Terminal** and select the **New Terminal** option



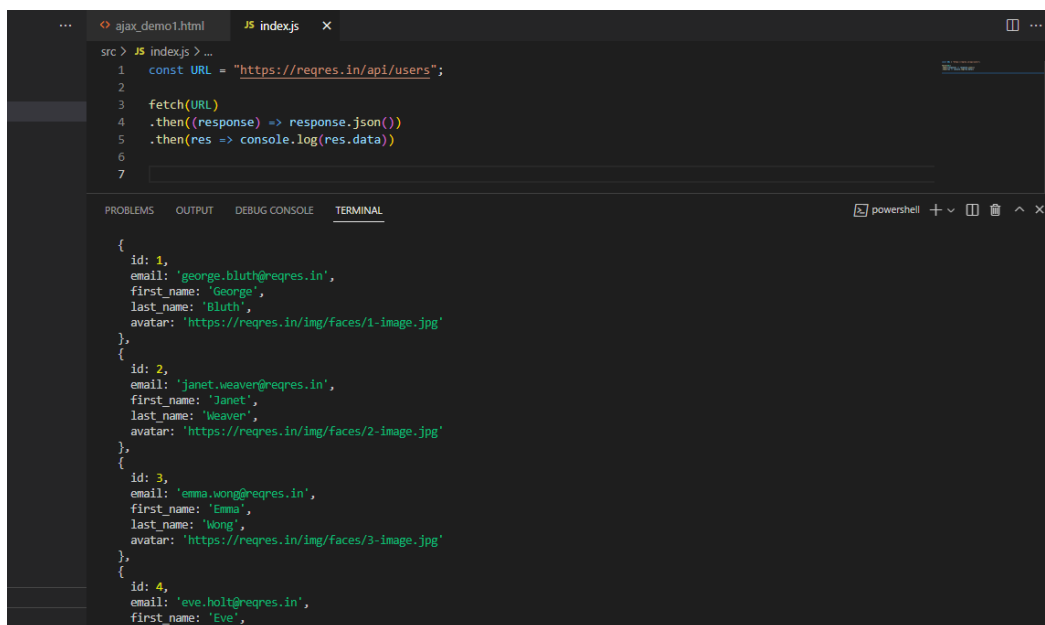
2.4 Run the following command to execute index.js:

**node index.js**



```
labuser@ip-172-31-19-65:~/Desktop/src$ node index.js
```

2.5 Click on enter, and the following data will be shown:



```
src > JS indexjs > ...
1  const URL = "https://reqres.in/api/users";
2
3  fetch(URL)
4    .then(response => response.json())
5    .then(res => console.log(res.data))
6
7
{
  id: 1,
  email: 'george.bluth@reqres.in',
  first_name: 'George',
  last_name: 'Bluth',
  avatar: 'https://reqres.in/img/faces/1-image.jpg'
},
{
  id: 2,
  email: 'janet.weaver@reqres.in',
  first_name: 'Janet',
  last_name: 'Weaver',
  avatar: 'https://reqres.in/img/faces/2-image.jpg'
},
{
  id: 3,
  email: 'emma.wong@reqres.in',
  first_name: 'Emma',
  last_name: 'Wong',
  avatar: 'https://reqres.in/img/faces/3-image.jpg'
},
{
  id: 4,
  email: 'eve.holt@reqres.in',
  first_name: 'Eve',
```

By following the above steps, you have successfully implemented AJAX calls using XMLHttpRequest and Fetch API, demonstrating real-time data retrieval techniques and the integration of promises for efficient asynchronous operations in a web development framework.