Axios Library

- Axios is a Promise-based HTTP client for JavaScript which can be used in your front-end application and in your Node.js backend.
- Axios is a lightweight HTTP client based on the XMLHttpRequests service. It is similar to the Fetch API and is used to perform HTTP requests
- Axios is a promise-based HTTP client that works both in the browser and in a Node.js environment.
- Axios is a client HTTP API based on the XMLHttpRequest interface provided by browsers.
- The Axios library wraps the complex XHR syntax and provides an abstract and declarative way to make requests from the browser as well as in a node environment.
- Axios is similar to the Fetch API and is used to perform HTTP requests
- By using Axios it is easy to send asynchronous HTTP request to REST endpoints and perform CRUD operations.
- The Axios library can be used in your plain JavaScript application or can be used together with more advanced frameworks like Vue.js, ReactJS, AngularJS etc.
- The first version of Axios was released around 5 years ago, and its open-source code is available on GitHub. Axios has multiple contributors that have contributed to each version of Axios.
- Axios is one of those recommenced libraries that used with ReactJs.
- Axios supports all modern browsers, including support for IE8, edge and higher.
- Axios is also free and open-source.

Features of Axios Library

- Make XMLHttpRequests from the browser.
- Make http requests from node.js.
- Supports the Promise API.
- Intercept request and response.
- Transform request and response data.
- It has the ability to cancel requests.
- Automatic transforms for JSON data.
- It enables client-side protection against XSRF.
- It has built-in support for download progress.

Advantages of Axios Library

- Request and response interception.
- Streamlined error handling.
- Protection against XSRF or CSRF.
- Support for upload progress.
- Axios has a way to set a response timeout.
- The ability to cancel requests.
- Support for older browsers.
- Automatic JSON data transformation.
- Lightweight library.

Axios Library Vs Fetch API

Axios Library

- Axios is a 3rd party library.
- Axios performs automatic JSON data transformation.
- Axios supported by older as well as newer browsers.
- Axios provides Simple & Cleaner Syntax.
- Built-in CSRF (or, XSRF) protection.
- Axios' response data contains the object.
- Axios request is ok when status is 200 and statusText is 'OK'.
- "Axios" uses the "data" property in order to send the data.
- Axios allows cancelling request and request timeout.
- Axios, will reject the request promise if 4xx or 5xx series error status codes is returned.
- Better error handling.

Fetch API

- Fetch is built into most modern browsers; no installation is required as such.
- Fetch uses a two-step process when dealing with JSON data.
- Fetch is supported by modern web browsers as it has modern JavaScript syntax.
- Fetch API makes your code complex.
- Fetch API has no protection feature.
- Fetch's response body has to be stringified.
- Fetch request is ok when response object contains the ok property.
- "fetch()" uses the "body" property in order to send the data.
- Fetch doesn't have cancelling request and request timeout feature.
- When using Fetch, if the server returns a 4xx or 5xx series error, your catch() method callback won't be called / invoked / triggered.
- No error handling process.

Axios Library Vs Fetch API Syntax

Fetch API

```
fetch('/', {
  // configuration
})
.then(response => response.json())
.then(response => {
   // do something with data
})
```

Axios Library

```
axios({
   url: '/',
   // configuration
})
.then(response => {
  // do something with JSON response data
})
```

Axios Library Alternatives

We have following alternative to axios that includes :-

- JavaScript "XMLHttpRequest" Object.
- JavaScript "fetch()" Method.
- JQuery "$.ajax()" Method.

Axios Environmental Setup

Axios does not come as a native JavaScript API, so we will have to manually import into our project. To get started, we have following ways to do this :-

a) By Using CDN

Axios CDN Link :
<script src="https://cdnjs.cloudflare.com/ajax/libs/axios/0.21.1/axios.min.js"></script>

b) By Using NPM

Command to install "Axios" Using "NPM" --> npm install axios

c) By Using Yarn

Command to install "Axios" Using "Yarn" --> yarn add axios

d) VSCode Extensions

We have following extensions related to axios in VSCode editor :-

1) Axios Snippets --> Yggdrasill-7C9
2) Axios Snippets --> presidentma
3) Axios Snippets --> Loyalpotato

note :-

- We cannot import axios library using ES6 "import" statement because axios doesn't provide an export statement in "axios.min.js / axios.js" module.
- Use, "Axios Library" with "CDN links".

Example -- Linking from "node_modules" folder

This method is useful when we install "axios" using "npm / yarn".

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>Document</title>
</head>
<body>
        <script src="./node_modules/axios/dist/axios.min.js" ></script>
    <script src="app.js"></script>
</body>
</html>
```

app.js

```
axios.get('https://jsonplaceholder.typicode.com/posts/1')
  .then(response => {
    // Handle the successful response here
    console.log('Response data:', response.data);
  })
  .catch(error => {
    // Handle any errors that occurred during the request
    console.error('Error:', error);
  });
```

Example -- Linking directly from "cdnjs"

This method is useful when we use "axios" from "cdnjs".

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>Document</title>
</head>
<body>
   <script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>
   <script src="app.js"></script>
</body>
</html>
```

app.js

```
axios.get('https://jsonplaceholder.typicode.com/posts/1')
 .then(response => {
   // Handle the successful response here
   console.log('Response data:', response.data);
 })
 .catch(error => {
   // Handle any errors that occurred during the request
   console.error('Error:', error);
 });
```

Axios Library Syntax

Syntax 1 -- Without "Async-Await" Syntax

```
let config = {
    method : 'get',                       //use method : ‘post’ to post data
    url : 'https://jsonplaceholder.typicode.com/posts/1', // API endpoint
    data : {name:'tushar'},               // Payload to send
    responseType : 'json'                 // Response type (optional)
}

let axiosData = axios(config)

axiosData.then((res) => {
    console.log(res.data)
}).catch((err) => {
    console.log(err)
})
```

Syntax 2 -- With "Async-Await" Syntax

```
async function sendData() {
  try {
    const config = {
      method: 'get',                      //use method : ‘post’ to post data
      url: 'https://jsonplaceholder.typicode.com/posts/1', // API endpoint
      data: { name:'tushar'},             // Payload to send
      responseType: 'json'                // Response type (optional)
    };
    const response = await axios(config); // Await the Axios request
    console.log(response.data); // Log the response data
  } catch (error) {
    console.error(error.message); // Log the error message
  }
}
sendData();
```

Axios Library Shorthand Methods & Properties

Axios also provides more functions to make other network requests as well, matching the HTTP verbs that you wish to execute, such as :-

- axios.request(config)
- axios.get(url[, config])
- axios.delete(url[, config])
- axios.head(url[, config])
- axios.options(url[, config])
- axios.post(url[, data[, config]])
- axios.put(url[, data[, config]])
- axios.patch(url[, data[, config]])

Also, the response object from a request contains the following information: -

- response.data --> The response provided by the server.
- response.status --> The HTTP status code from the server response.
- response.statusText --> HTTP status message from the server response.
- response.headers --> The headers that the server responded with.
- response.config --> The config that was provided to axios for the request.
- response.request --> The request that generated this response.

Examples of Axios Library

a). Axios with "get()" request

Example -- Fetching data from a text file

```
let config = {
    method : 'get',
    url : 'axiosData.txt'
}

let axiosData = axios(config)
axiosData.then((res) => {
    console.log(res)
    console.log(res.data)
}).catch((err) => {
    console.log(err)
})
```

Example -- Fetching data from a Remote API

```
let config = {
    method : 'get',
    url : 'https://api.github.com/users'
}

let axiosData = axios(config)
axiosData.then((res) => {
    console.log(res.data[0])
}).catch((err) => {
    console.log(err)
})
```

Example -- Using the shorthand method with axios

```
let axiosData = axios.get('axiosData.txt').then((res) => {
   console.log(res.data)
}).catch((err) => {
   console.log(err)
})
```

Example -- Axios by default called with "get()" method

```
let axiosData = axios('axiosData.txt').then((res) => {
   console.log(res.data)
}).catch((err) => {
   console.log(err)
})
```

Example -- Using Parameters in "Axios" Request

```
let axiosData = axios('axiosData.txt',{method : 'get'})
.then((res) => {
   console.log(res.data)
}).catch((err) => {
   console.log(err)
})
```

Example -- Using Parameters in "Axios" Request

```
let config = {
   method : 'get'
}
let axiosData = axios('axiosData.txt',config)
.then((res) => {
   console.log(res.data)
}).catch((err) => {
   console.log(err)
})
```

Example -- Error Handling in Axios

```
let axiosData = axios.get('axiosData.txt')
.then((res) => {
   console.log(res.data)
}).catch((err) => {
   console.log(err)
}).then(() => {
   console.log("Promise Settled")
})
```

Example -- Axios with "async-await"

```
async function asyncAwait(){
      let config = {
      url : 'https://api.github.com/users',
      method : 'get'
   }
   let response = await axios(config)
   console.log(response)
}
asyncAwait()
```

Example -- Axios with "async-await"

```
async function asyncAwait(){
   let response = await axios.get('axiosData.txt')
   console.log(response)
}
asyncAwait()
```

Example -- Axios with "async-await" & Error Handling

```
async function asyncAwait(){
     try{
        let response = await axios.get('axiosData.txt')
        console.log(response)
    }catch(error){
        console.log(error)
    }
  }
asyncAwait()
```

Example -- Printing data on webpage using json file

```
async function asyncAwait(){
    try{
        let response = await axios.get('axiosData.json')
        let prntData = document.getElementById('abc');
        prntData.innerHTML = `Hello ${response.data.name} your age is
${response.data.age}`
    }catch(error){
        console.log(error)
    }
}
asyncAwait()
```

// for the above code, a <div> element with id="abc" is required in the html file.

Example -- Axios with "JSONPlaceholder"

```
async function asyncAwait(){
       try{
        let response = await
axios.get('https://jsonplaceholder.typicode.com/posts/1')
        console.log(response)
        console.log(response.data)
    }catch(error){
        console.log(error)
    }
  }
asyncAwait()
```

Example -- Axios with "JSONPlaceholder" With "params" Property

```
function asyncAwait(){
   let config = {
    method : 'get',
    url : 'https://jsonplaceholder.typicode.com/posts/',
    params : {id : 1}
  }
  axios(config).then((res) => {
    console.log("Response Data : ",res.data)
    console.log("Response Data Title : ",res.data[0].title)
    console.log("Response Data Body : ",res.data[0].body)
  }).catch((err) => {
    console.log(err)
  })
 }
asyncAwait()
```

Example -- Axios with "JSONPlaceholder"

```
async function asyncAwait(){
  try{
    let response = await axios.get('https://jsonplaceholder.typicode.com/posts')
    console.log(response)
    console.log(response.data[0].id)
    console.log(response.data[0].title)
    console.log(response.data[0].body)
  }catch(error){
    console.log(error)
  }
}
asyncAwait()
```

Example -- Handling Multiple Data using "forEach()" Loop

```
function asyncAwait(){
    let config = {
        method : 'get',
        url : 'https://jsonplaceholder.typicode.com/posts/',
    }
    axios(config).then((res) => {
       res.data.forEach((item) => {
           console.log("ID : ",item.id);
           console.log("TITLE : ",item.title);
           console.log("BODY : ",item.body)
       })
    }).catch((err) => {
        console.log(err)
    })
}
asyncAwait()
```

Example -- Handling Multiple Data using "map()" Method

```
function asyncAwait(){
    let config = {
        method : 'get',
        url : 'https://jsonplaceholder.typicode.com/posts',
    }
    axios(config).then((res) => {
       res.data.map((item) => {
            console.log("ID : ",item.id)
            console.log("TITLE : ",item.title)
            console.log("BODY : ",item.body)
       })
    }).catch((err) => {
        console.log(err)
    })
}
asyncAwait()
```

Example -- Showing Multiple Data using "map()" Method

```
function asyncAwait(){
      let config = {
        method : 'get',
        url : 'https://jsonplaceholder.typicode.com/posts',
    }
    axios(config).then((res) => {
       res.data.map((item) => {
            let prntData = document.getElementById('abc');
            prntData.innerHTML += `
                <p>${item.id}</p>
                <p>${item.title}</p>
                <p>${item.body}</p>
                <hr/>
            `
       })
    }).catch((err) => {
        console.log(err)
    })
  }
asyncAwait()
```

Axios project using 'get()' request

a) Example -- Using GitHub API

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>Document</title>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-
bootstrap/4.6.0/css/bootstrap.min.css" />
</head>
<body>
    <div class="container">
        <h2 class="mt-4 text-center">Axios Project</h2>
        <button type="button" id="get" class="mt-4 btn btn-primary btn-
block">Send Get Request</button>
        <div id="response"></div>
    </div>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/twitter-
bootstrap/4.6.0/js/bootstrap.min.js"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/axios/0.21.1/axios.min.js"></script>
    <script src="app.js"></script>
</body>
</html>
```

main.js

```
document.getElementById('get').addEventListener('click',getData);
function getData(){
    let config = {
        method : 'get',
        url : 'https://api.github.com/users'
    }
    axios(config).then(res => {
        showData(res)
        console.log(res)
    }).catch(err => {
        showError(err)
    })
}

// Function to Show API Response
function showData(response){
    document.getElementById('response').innerHTML = `
                <div class="mt-5 card">
                    <div class="card-header">
                        <h3>Response Status : ${response.status}</h3>
                    </div>
                    <div class="card-body">`
                    +
                    response.data.map(item => `
                    <div class="card mb-3" >
                    <div class="card-header">
                        <h5>Github User - ${item.id}</h5>
                    </div>
                    <div class="row g-0">
                        <div class="col-md-4">
                            <img src="${item.avatar_url}" class="mx-2 my-2 img-
thumbnail">
                        </div>
                        <div class="col-md-8">
                            <div class="card-body">
```

```
                                <span><span class="font-weight-bold">Username :
</span>${item.login}</span>
                           </div>
                        </div>
                        </div>
                     </div>
                    `)
                    +
                    `
                    </div>
                </div>
    `;
}
function showError(error){
    document.getElementById('response').innerHTML = `
    <div class="mt-4 alert alert-danger" role="alert">
            ${error}
    </div>`
}
```

b) Example -- Using JSONPlaceholder

index.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>
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.0/dist/css/bootstrap.min.css">
</head>
<body>
        <div class="container">
            <h1 class="mt-4 text-center">AXIOS PROJECT</h1>
                <div class="mt-4">
                    <center>
                        <button type="button" class="btn btn-primary btn-block"
id="get">Send GET Request</button>
                    </center>
                </div>
                <div id="response"></div>
        </div>
        <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.0/dist/js/bootstrap.bundle.min.j
s" ></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/axios/0.21.1/axios.min.js"></script>
    <script src="app.js"></script>
</body>
</html>
```

main.js -- Without Shorthand

```
document.getElementById('get').addEventListener('click',getData);
// Functions Applied On Buttons
function getData(){
    let config = {
        method : 'get',
        url : 'https://jsonplaceholder.typicode.com/users',
        params:{
            _limit:5
        }
    }
    axios(config).then(res => {
       showData(res)
    }).catch(err => {
       showError(err)
    })
}
// Function Applied On Response
function showData(response){
   document.getElementById('response').innerHTML = `
        <div class="card mt-5">
            <div class="card-header">
                <h5>RESPONSE IN HTML WITH STATUS : ${response.status}</h5>
            </div>
            <div class="card-body">`
            +
                response.data.map(item => `
                <div class="card">
                   <div class="card-body">
                       <span class="card-text"><span class="font-weight-
bold">Name</span> : ${item.name}</span><br>
                       <span class="card-text"><span class="font-weight-
bold">Username</span> : ${item.username}</span><br>
                       <span class="card-text"><span class="font-weight-
bold">Email</span> : ${item.email}</span><br>
```

```
                        <span class="card-text"><span class="font-weight-
bold">Address</span> : ${item.address.street + ","+
item.address.city}</span><br>
                        <span class="card-text"><span class="font-weight-
bold">Company</span> : ${item.company.name}</span><br>
                        <span class="card-text"><span class="font-weight-
bold">Website</span> : http://www.${item.website}</span><br>
                    </div>
                 </div>
                 `)
            +
            `</div>
        </div>
   `
}
```

// Function Applied On Error

```
function showError(error){
    document.getElementById('response').innerHTML = `
        <div class="alert alert-danger mt-4" role="alert">
            ${error}
        </div>
    `
}
```

Or, we can use

main.js -- With Shorthand

```
document.getElementById('get').addEventListener('click',getData);
// Functions Applied On Buttons
function getData(){
   axios.get('https://jsonplaceholder.typicode.com/users?_limit=5')
   .then(res => {
     showData(res)
   }).catch(err => {
     showError(err)
   })
}
// Function Applied On Response
function showData(response){
  document.getElementById('response').innerHTML = `
       <div class="card mt-5">
          <div class="card-header">
             <h5>RESPONSE IN HTML WITH STATUS : ${response.status}</h5>
          </div>
          <div class="card-body">`
          +
             response.data.map(item => `

             <div class="card">
               <div class="card-body">
                  <span class="card-text"><span class="font-weight-
bold">Name</span> : ${item.name}</span><br>
                  <span class="card-text"><span class="font-weight-
bold">Username</span> : ${item.username}</span><br>
                  <span class="card-text"><span class="font-weight-
bold">Email</span> : ${item.email}</span><br>
                  <span class="card-text"><span class="font-weight-
bold">Address</span> : ${item.address.street + ","+
item.address.city}</span><br>
```

```
                        <span class="card-text"><span class="font-weight-
bold">Company</span> : ${item.company.name}</span><br>
                        <span class="card-text"><span class="font-weight-
bold">Website</span> : http://www.${item.website}</span><br>
                    </div>
                 </div>
                 `)
            +
            `</div>
        </div>
   `
}
```

// Function Applied On Error

```
function showError(error){
    document.getElementById('response').innerHTML = `
        <div class="alert alert-danger mt-4" role="alert">
            ${error}
        </div>
    `
}
```