Axios Project Using 'Get' Request

Example -- Sending Data Using Axios to a Remote Resource

```
function asyncAwait(){
      let config = {
        method : 'POST',
        url : 'https://reqres.in/api/users',
        headers : {
            'Content-Type' : 'application/json'
        },
        data : '{"name":"Monika Singh","age":27}'
    }
    axios(config).then((res) => {
       console.log(res.data)
    }).catch((err) => {
        console.log(err)
    })

}
asyncAwait()
```

Example -- Sending Data Using Axios & Async-Await to a Remote Resource

```
async function asyncAwait(){
   try{
    let config = {
      method : 'POST',
      url : 'https://reqres.in/api/users',
      headers : {
        'Content-Type' : 'application/json'
      },
      data : '{"name":"Rohit Singh","age":27}'
    }
    const res = await axios(config);
    console.log(res.data)
  }catch(error){
    console.log(error)
  }
 }
asyncAwait()
```

Example -- Sending HTML Form Data Using Axios 'POST' Method

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>
    <form method="post" >
        Enter Name: <input type="text" name="name" id="name"><br><br>
        Enter Age: <input type="text" name="age" id="age"><br><br>
        <input type="submit" value="Submit" id="btn">
    </form>
        <script src="./node_modules/axios/dist/axios.min.js" ></script>
    <script src="app.js"></script>
</body>
</html>
```

app.js

```
 document.getElementById('btn').addEventListener('click',asyncAwait)
  function asyncAwait(e){
    e.preventDefault();
    let getName = document.getElementById('name').value;
    let getAge = document.getElementById('age').value;
        let config = {
            method : 'POST',
            url : 'https://reqres.in/api/users',
            headers : {
                'Content-Type' : 'application/json'
            },
            data : JSON.stringify({name:getName,age:getAge})
        }
        axios(config).then((res) => {
            console.log(res.data)
        }).catch((error) => {
            console.log(error)
        })
  }
```

Or,

app.js

```
document.getElementById('btn').addEventListener('click',asyncAwait)
 async function asyncAwait(e){
    e.preventDefault();
    try{
        let getName = document.getElementById('name').value;
        let getAge = document.getElementById('age').value;
         let config = {
            method : 'POST',
            url : 'https://reqres.in/api/users',
            headers : {
                'Content-Type' : 'application/json'
            },
            data : JSON.stringify({name:getName,age:getAge})
        }
           let res = await axios(config);
        console.log(res.data)
    }catch(error){
        console.log(error)
    }
}
```

Axios Project Using 'Post' Request

index.html

```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>
        <center>
            <button type="button" id="post" class="mt-4 btn btn-primary btn-block">Send Post Request</button>
        </center>
        <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="main.js"></script>
</body>
</html>
```

main.js

```
document.getElementById('post').addEventListener('click',postData);
// Functions Applied On Buttons
function postData(){
      let config = {
        method : 'post',
        url : 'https://jsonplaceholder.typicode.com/comments',
        data : {
            name : "Rahul Singh",
            email : 'rahul@gmail.com',
            body : 'Hey this is new Comment'
        }
    }
    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 : ${response.status}</h5>
            </div>
            <div class="card-body">
                <span class="font-weight-bold">Name : </span>${response.data.name}<br>
                <span class="font-weight-bold">Email : </span>${response.data.email}<br>
                <span class="font-weight-bold">Comment :
</span>${response.data.body}<br>
                <span class="font-weight-bold">PostId :
</span>${response.data.postId}<br>
            </div>
        </div>
   `
}
// Function Applied On Error
function showError(error){
    document.getElementById('response').innerHTML = `
        <div class="alert alert-danger mt-4" role="alert">
            ${error}
        </div>
    `
}
```

Axios Project Using 'Put' Request

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>
                <center>
            <button type="button" id="put" class="mt-4 btn btn-success btn-block">Send PUT Request</button>
        </center>
        <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="main.js"></script>
</body>
</html>
```

main.js

```
document.getElementById('put').addEventListener('click',updateData);
// Functions Applied On Buttons
function updateData(){
  let config = {
      method : 'put',
      url : 'https://jsonplaceholder.typicode.com/comments/1',
      data : {
         name : "Rahul Singh",
         email : 'rahul@gmail.com',
         body : 'Hey this is new Comment'
      }
   }
   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 : ${response.status}</h5>
         </div>
         <div class="card-body">
            <span class="font-weight-bold">Name : </span>${response.data.name}<br>
            <span class="font-weight-bold">Email : </span>${response.data.email}<br>
            <span class="font-weight-bold">Comment :
</span>${response.data.body}<br>
            <span class="font-weight-bold">PostId :
</span>${response.data.postId}<br>
         </div>
      </div>
  `}
```

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

Axios Project Using 'Patch' Request

index.html

```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>
        <center>
            <button type="button" id="patch" class="mt-4 btn btn-success btn-block">Send PATCH Request</button>
        </center>
        <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="main.js"></script>
</body>
</html>
```

main.js

```
document.getElementById('patch').addEventListener('click',updateData);
// Functions Applied On Buttons
function updateData(){
    let config = {
        method : 'patch',
        url : 'https://jsonplaceholder.typicode.com/comments/1',
        data : {
            name : "Rahul Singh",
            email : 'rahul@gmail.com',
            body : 'Hey this is new Comment'
        }
    }
    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 : ${response.status}</h5>
            </div>
            <div class="card-body">
                <span class="font-weight-bold">Name : </span>${response.data.name}<br>
                <span class="font-weight-bold">Email : </span>${response.data.email}<br>
                <span class="font-weight-bold">Comment :
</span>${response.data.body}<br>
                <span class="font-weight-bold">PostId :
</span>${response.data.postId}<br>
            </div>
        </div>   `         }
```

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

Axios Project Using 'Delete' Request

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>
        <center>
            <button type="button" id="delete" class="mt-4 btn btn-danger btn-block">Send DELETE Request</button>
        </center>
        <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="main.js"></script>
</body>
</html>
```

main.js

```
document.getElementById('delete').addEventListener('click',deleteData);
// Functions Applied On Buttons
function deleteData(){
    let config = {
        method : 'delete',
        url : 'https://jsonplaceholder.typicode.com/comments/1',
    }
    axios(config)
    .then(res => {
       showData(res)
       console.log(new String(res.data))
    }).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">
                <span class="font-weight-bold">Data Deleted : </span>`
                +
                    (JSON.stringify(response.data) ==  '{}' ? 'Yes Data Deleted' : 'Data Not Deleted')
                +
                `<br>
            </div>
        </div>
   `
}
```

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

Axios Interceptors

- Axios has one powerful feature called Interceptors.
- Interceptors is a feature that allows an application to intercept requests or responses before they are handled by the ".then()" or the ".catch()" methods.
- Axios interceptors allows you to run your code or modify the request and/or response before the request and/or response reaches their destination.
- Axios interceptors allows you to run your code or modify the request and/or response before the request and/or response is started.
- Axios Request Interceptor is a method in the promise-based HTTP client that allow you to run your code or modify the request before the actual call to the endpoint is made.
- In Simple Words,
    - it allows you to write or execute a piece of your code before the request gets sent.
    - It allows you to write or execute a piece of your code before response reaches the calling end.
- We can use interceptors to transform the request before Axios sends it, or transform the response before Axios returns the response to your code.
- Interceptors can be very helpful, if used properly as it allows us to make checks and even changes even before the main method is executed.

Types of Axios Interceptors

a) Request Interceptors

- Axios calls request interceptors before sending the request, so you can use request interceptors to modify the request.
- This is called before the actual call to the endpoint is made.

Syntax of Request Interceptor

```
axios.interceptors.request.use(req => {
  // Do something before request is sent
  return req;
});
```

Or,

```
axios.interceptors.request.use(function (req) {
    // Do something before request is sent
    return req;
  }, function (error) {
    // Do something with request error
    return Promise.reject(error);
});
```

b) Response Interceptors

- Axios calls response interceptors after it sends the request and receives a response.
- This is called before the promise is completed and the data is received by the then callback.

Syntax of Response Interceptor

```
axios.interceptors.response.use(res => {
  // Do something with response data
});
```

Or,

```
axios.interceptors.response.use(function (res) {
   // Do something with response data
   return res;
}, function (error) {
   // Do something with response error
   return Promise.reject(error);
});
```

Examples Based on Axios Interceptors

a) Related to Request Interceptor

Example -- Simple Example

```
function sendReq(){
    axios.get('https://jsonplaceholder.typicode.com/posts').then(res => {
        console.log(res)
    }).catch(err => {
        console.log(err)
    })
}

axios.interceptors.request.use(req => {
    console.log(req.method.toUpperCase())
    console.log(req.url)
    return req
})

sendReq()
```

Example -- Request Interceptor With Error Handling

```
function sendReq(){
    axios.get('https://jsonplaceholder.typicode.com/posts').then(res => {
        console.log(res)
    }).catch(err => {
        console.log(err)
    })
}

axios.interceptors.request.use(req => {
    console.log(req.method.toUpperCase())
    console.log(req.url)
    return req
},
error => {
    return Promise.reject(error)
})

sendReq()
```

Example -- Updating Request Params Using Interceptor

```
function sendReq(){
    axios.get('https://jsonplaceholder.typicode.com/posts').then(res => {
        console.log(res)
    }).catch(err => {
        console.log(err)
    })
}

axios.interceptors.request.use(req => {
    console.log(req.method)
    console.log(req.url)
    req.method = 'post'
    req.url = 'https://jsonplaceholder.typicode.com/todos'
    req.data = {
        title : "New Todo Data",
        completed : true
    }
    console.log(req.method)
    console.log(req.url)
    return req
})

sendReq()
```

b) Related to Response Interceptor

Example -- Simple Example

```
function sendReq(){
    axios.get('https://jsonplaceholder.typicode.com/posts').then(res => {
        console.log("Hello Response Data",res)
    }).catch(err => {
        console.log(err)
    })
}

axios.interceptors.response.use(res => {
    console.log(res.data)
    console.log(res.status)
    return res
})

sendReq()
```

Example -- Response Interceptor With Error Handling

```
function sendReq(){
    axios.get('https://jsonplaceholder.typicode.com/posts').then(res => {
        console.log(res)
    }).catch(err => {
        console.log(err)
    })
}

axios.interceptors.response.use(res => {
    console.log(res.data)
    console.log(res.status)
    return res
},
error => {
    return Promise.reject(error)
})

sendReq()
```

Example -- Updating Response Params Using Interceptor

```
function sendReq(){
    axios.get('https://jsonplaceholder.typicode.com/posts').then(res => {
        console.log(res)
    }).catch(err => {
        console.log(err)
    })
}

axios.interceptors.response.use(res => {
    console.log(res.data)
    console.log(res.status)
    console.log(res.statusText)

    res.data = "New Data"
    res.statusText = "Data Fetched Successfully"
    console.log(res.data)
    console.log(res.statusText)
    return res
})

sendReq()
```

Example - Combining both 'Request' & 'Response' Interceptors

```
function sendReq(){
    axios.get('https://jsonplaceholder.typicode.com/posts').then(res => {
        console.log(res)
    }).catch(err => {
        console.log(err)
    })
}
axios.interceptors.request.use(req => {
    console.log(req.method)
    console.log(req.url)
    req.method = 'post'
    req.url = 'https://jsonplaceholder.typicode.com/todos'
    req.data = {
        title : "New Todo Data",
        completed : true
    }
    console.log("Updates Request Method is : ",req.method)
    console.log("Updates Request URL is : ",req.url)
    return req
})
axios.interceptors.response.use(res => {
    console.log(res.data)
    console.log(res.status)
    console.log(res.statusText)
    res.data = "New Data"
    res.statusText = "Data Fetched Successfully"
    console.log("Updates Response Data is : ",res.data)
    console.log("Updates Response Statustext is : ",res.statusText)
    return res
})
sendReq()
```

Example - Removing Interceptors

We can use "eject()" method in order to Remove Interceptor.

```
function sendReq(){
    axios.get('https://jsonplaceholder.typicode.com/posts').then(res => {
        console.log(res)
    }).catch(err => {
        console.log(err)
    })
}

let myInterceptor = axios.interceptors.request.use(req => {
    console.log(req.method.toUpperCase())
    console.log(req.url)
    return req
})

axios.interceptors.request.eject(myInterceptor);

sendReq()
```

Creating Custom Axios Instance

Axios instance will work exactly the same way the global Axios instance works.
With axios instance we don't need to add base path explicitly in every request we make, we can also define timeout, default headers, in this instance etc.
If we have multiple api calls, to the same endpoint and adding the the full url doesn't seem right, plus what if later on the endpoint is changed, the changes just seem too much of manual rework. So, axios instance solves this problem.

Syntax to create "axios instance"

```
        let instance = axios.create()
```

Example -- Creating Simple Axios Instance

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>
    <button onclick="asyncAwait()">Click Me</button>
       <script src="./node_modules/axios/dist/axios.min.js" ></script>
    <script src="app.js"></script>
</body>
</html>
```

app.js

```
let ai = axios.create({
    baseURL : 'https://reqres.in/api',
    timeout : 3000
})

function asyncAwait(){

   ai('/users').then((res) => {
        res.data.data.map((item) => {
            console.log(item)
        })
   }).catch((error) => {
       console.log(error)
   })

}
```