# ASSIGNMNET 1B

## Aim:- Write a JavaScript Program to get the user registration data and push to array/local storage with AJAX POST method and data list in new page.

AJAX stands for Asynchronous JavaScript and XML. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and Java Script.

Ajax uses XHTML for content, CSS for presentation, along with Document Object Model and JavaScript for dynamic content display.

Conventional web applications transmit information to and from the sever using synchronous requests. It means you fill out a form, hit submit, and get directed to a new page with new information from the server.

With AJAX, when you hit submit, JavaScript will make a request to the server, interpret the

results, and update the current screen. In the purest sense, the user would never know that anything was even transmitted to the server. XML is commonly used as the format for receiving server data, although any format, including plain text, can be used. AJAX is a web browser technology independent

of web server software.

A user can continue to use the application while the client program requests information from

the server in the background. Intuitive and natural user interaction. Clicking is not required, mouse movement is a sufficient event trigger. Data-driven as opposed to page-driven.

AJAX is based on the following open standards −

- Browser-based presentation using HTML and Cascading Style Sheets (CSS).
- Data is stored in XML format and fetched from the server.
- Behind-the-scenes data fetches using XMLHttpRequest objects in the browser.
- JavaScript to make everything happen

AJAX cannot work independently. It is used in combination with other

technologies to create interactive webpages.

• JavaScript

– Loosely typed scripting language.

– JavaScript function is called when an event occurs in a page.

– Glue for the whole AJAX operation.

• DOM

– API for accessing and manipulating structured documents.

– Represents the structure of XML and HTML documents.

• CSS

– Allows for a clear separation of the presentation style from the content

and may be changed programmatically by JavaScript

• XMLHttpRequest

– JavaScript object that performs asynchronous interaction with the server.

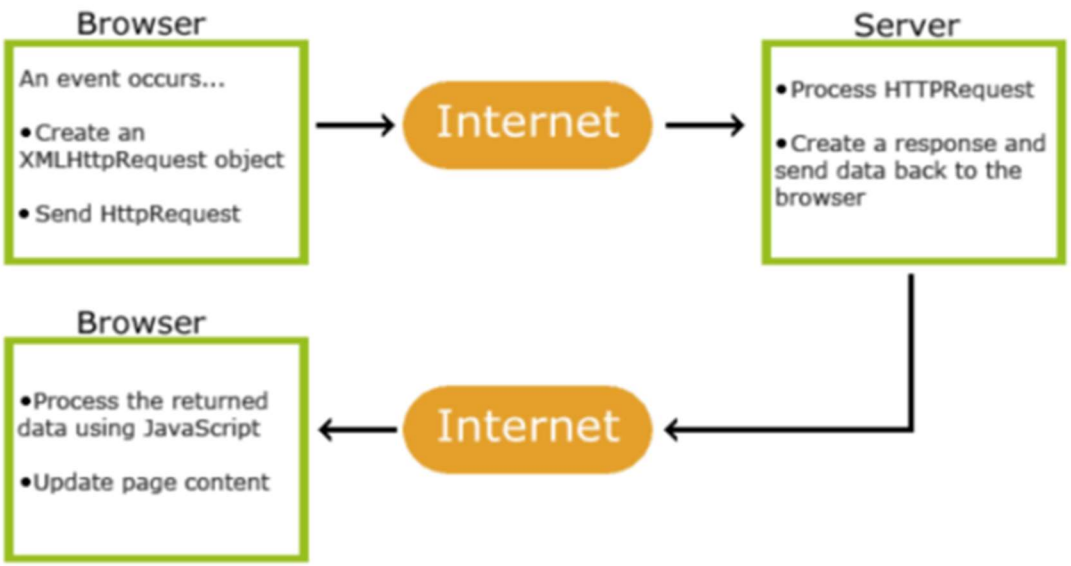


Fig 1. How AJAX works

## AJAX – Events : onreadystatechange Event Properties

| Property | Description |
|---|---|
| onReadyStateChange | It is called whenever readystate attribute changes. It must not be used with synchronous requests. |

| | |
|---|---|
| readyState | represents the state of the request. It ranges from 0 to 4.<br><br>• 0: request not initialized (open() is not called.)<br>• 1: server connection established (open is called but send() is not called.)<br>• 2: request received (send() is called, and headers and status are available.)<br>• 3: processing request (Downloading data; responseText holds the data.)<br>• 4: request finished and response is ready (The operation is completed fully.) |
| status | 200: "OK"<br>403: "Forbidden"<br>404: "Page not found" |

XMLHttpRequest object properties

| **Property** | **Description** |
|---|---|
| readyState | An integer from 0. . .4. (0 means the call is uninitialized, 4 means<br><br>that the call is complete.) |
| onreadystatechange | Determines the function called when the objects readyState changes. |
| responseText | Data returned from the server as a text string (read-only). |
| responseXML | Data returned from the server as an XML document object (read-only). |
| status | HTTP status code returned by the server |

| | |
|---|---|
| statusText | HTTP status phrase returned by the server |

XMLHttpRequest object methods

| **Method** | **Description** |
|---|---|
| open('method', 'URL', asyn) | Specifies the HTTP method to be used (GET or POST as a string, the target URL, and whether or not the request should be handled asynchronously (asyn should be true or false, if omitted, true is assumed). |
| send(content) | Sends the data for a POST request and starts the request, if GET is used you hould call send(null). |
| setRequestHeader('x','y') | Sets a parameter and value pair x=y and assigns it to the header to be sent with the request. |
| getAllResponseHeaders() | Returns all headers as a string. |
| getResponseHeader(x) | Returns header x as a string. |
| abort() | Stops the current operation. |

## //code

## //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">
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
```

```
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p" crossorigin="anonymous"></script>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
  <script src="/main.js"></script>
  <title>Document</title>
</head>
<body>
  <form class="row g-3 needs-validation" novalidate>
    <div class="col-md-4">
      <label for="validationCustom01" class="form-label">First name</label>
      <input type="text" class="form-control" id="firstName" value="Mark" required>
      <div class="valid-feedback">
        Looks good!
      </div>
    </div>
    <div class="col-md-4">
      <label for="validationCustom02" class="form-label">Last name</label>
      <input type="text" class="form-control" id="lastName" value="Otto" required>
      <div class="valid-feedback">
        Looks good!
      </div>
    </div>
    <div class="col-md-4">
      <label for="validationCustomUsername" class="form-label">Username</label>
      <div class="input-group has-validation">
        <span class="input-group-text" >@</span>
        <input type="text" class="form-control" id="username" aria-describedby="inputGroupPrepend" required>
        <div class="invalid-feedback">
          Please choose a username.
        </div>
      </div>
    </div>
    <div class="col-md-6">
      <label for="validationCustom03" class="form-label">City</label>
      <input type="text" class="form-control" id="city" required>
      <div class="invalid-feedback">
        Please provide a valid city.
      </div>
    </div>
    <div class="col-md-3">
      <label for="validationCustom04" class="form-label">State</label>
      <select class="form-select" id="state" required>
        <option selected disabled value="">Choose...</option>
        <option>...</option>
      </select>
      <div class="invalid-feedback">
        Please select a valid state.
      </div>
    </div>
    <div class="col-md-3">
```

```
        <label for="validationCustom05" class="form-label">Zip</label>
        <input type="text" class="form-control" id="zipcode" required>
        <div class="invalid-feedback">
          Please provide a valid zip.
        </div>
      </div>
      <div class="col-12">
        <div class="form-check">
          <input class="form-check-input" type="checkbox" value="" id="invalidCheck" required>
          <label class="form-check-label" for="invalidCheck">
            Agree to terms and conditions
          </label>
          <div class="invalid-feedback">
            You must agree before submitting.
          </div>
        </div>
      </div>
      <div class="col-12">
        <button class="btn btn-primary" type="button" id="submit-btn">Submit form</button>
      </div>
    </form>
</body>
<script>
   // Example starter JavaScript for disabling form submissions if there are invalid fields
(function () {
  'use strict'

  // Fetch all the forms we want to apply custom Bootstrap validation styles to
  var forms = document.querySelectorAll('.needs-validation')

  // Loop over them and prevent submission
  Array.prototype.slice.call(forms)
    .forEach(function (form) {
      form.addEventListener('submit', function (event) {
        if (!form.checkValidity()) {
          event.preventDefault()
          event.stopPropagation()
        }

        form.classList.add('was-validated')
      }, false)
    })
})();
</script>
</html>
```

## //Main.js

```
$(document).ready(function()
{
   $("#submit-btn").click(function(event)
   {
```

```
        var formdata = {
            fname : $("#firstName").val(),
            lname : $("#lastName").val(),
            username : $("#username").val(),
            city : $("#city").val(),
            state : $("#state").val(),
            zipcode : $("#zipcode").val(),
        };

        storeDataToLocalStorage(formdata);

        // console.log(formdata);
        event.preventDefault();
       sendData(formdata);

        window.location.href='displayData.html'



    })
})

function storeDataToLocalStorage(formdata)
 {
   //The getItem() method of the Storage interface, when passed a key name, will return that key's value,
   //or null if the key does not exist, in the given Storage object.
      console.log(formdata);
        if (!localStorage.getItem("formdata")) {
            localStorage.setItem("formdata", JSON.stringify(formdata));
        } else {
          localStorage.removeItem("formdata");
          localStorage.setItem("formdata", JSON.stringify(formdata));

        }
 }

 function sendData(formdata) {

    console.log(formdata);
   var studentData = JSON.stringify(formdata);
   let xhr = new XMLHttpRequest();

   xhr.open("POST", "http://localhost:5501",true);
   xhr.setRequestHeader("Content-Type", "application/json");
   xhr.send(studentData);

 }

//DisplayData.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">

   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-
ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
crossorigin="anonymous"></script>
   <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
   <script src="displayData.js"></script>
   <title>Document</title>
</head>
<body>

   <table>
      <tr>
         <th>First Name</th>
         <th>Last Name</th>
         <th>Username</th>
         <th>City</th>
         <th>State</th>
         <th>Zipcode</th>
      </tr>
      <tr>
         <td id="firstName"></td>
         <td id="lastsName"></td>
         <td id="userName"></td>
         <td id="city"></td>
         <td id="state"></td>
         <td id="zipcode"></td>
      </tr>

   </table>
</body>
</html>

//DisplayData.js

$(document).ready(function () {
   getData();
 });
 // Get data from local storage and disply on console and on next page
 function getData() {
   let localStorageData = localStorage.getItem("formdata");
   let studentObj = JSON.parse(localStorageData);
   console.log(studentObj);

   $("#firstName").text(studentObj.fname);
   $("#lastName").text(studentObj.lname);
```

```
    $("#userName").text(studentObj.username);
    $("#city").text(studentObj.city);
    $("#state").text(studentObj.state);
    $("#zipcode").text(studentObj.zipcode);

  }
```

# Output