Experiment 9 : To study AJAX

Name of Student	**Spandan Deb**
Class Roll No	**13**
D.O.P.	
D.O.S.	
Sign and Grade	

Aim: To study AJAX

Theory:-

1. How do Synchronous and Asynchronous Requests differ?

Synchronous and Asynchronous requests are two different methods used in web development to communicate with a server, especially when using technologies like **AJAX (Asynchronous JavaScript and XML)**.

Synchronous Requests:

- In a synchronous request, the **browser waits** for the server to respond before continuing to execute the rest of the JavaScript code.
- The entire user interface (UI) **freezes or becomes unresponsive** until the request is complete.
- This approach can lead to a **poor user experience**, especially if the server takes a long time to respond.

Example:
If a synchronous request is made to load data from a server, the user cannot interact with the webpage (like clicking buttons or typing) until the response is received.

Asynchronous Requests:

- In an asynchronous request, the browser **does not wait** for the response.
- The remaining JavaScript code continues to execute, and the browser stays responsive.
- Once the server responds, a **callback function** (or event listener) is triggered to handle the response.

Example:
Fetching live search results without reloading the page. The user continues typing while results update in real time.

Key Differences:

Feature	Synchronous	Asynchronous
Browser Behavior	Waits for response	Does not wait
User Experience	Freezes UI	UI remains responsive
Implementation	Simpler, but less efficient	Requires callbacks or promises
Usage	Rare in modern web apps	Common and preferred

2. Describe various properties and methods used in XMLHttpRequest Object

The XMLHttpRequest object is a built-in JavaScript object used to interact with servers and fetch data without reloading the webpage. It is a core part of AJAX-based applications.

Commonly Used Properties:

Property	Description
readyStat e	Returns the current state of the request (0 to 4).
status	Returns the HTTP status code (e.g., 200 for OK, 404 for Not Found).
statusTex t	Returns the textual status (e.g., "OK", "Not Found").
responseT ext	Returns the response data as a string.
responseX ML	Returns the response data as an XML document (if the response is XML).

readyState Values:

Value	State	Description
0	UNSENT	Request not initialized
1	OPENED	open() has been called
2	HEADERS_RECEIVED	send() has been called
3	LOADING	Receiving the response
4	DONE	Request finished and response is ready

Commonly Used Methods:

Method	Description
open(method, url, async)	Initializes a request. method is "GET" or "POST", async is a boolean.
send()	Sends the request to the server.
setRequestHeader(head er, value)	Sets a request header (e.g., Content-Type). Used after open(), before send().
abort()	Cancels an ongoing request.

Example:

Javascript

```
let xhr = new XMLHttpRequest();
xhr.open("GET", "data.json", true);  // Asynchronous GET request
xhr.onreadystatechange = function() {
  if (xhr.readyState === 4 && xhr.status === 200) {
    console.log(xhr.responseText);   // Handle response
  }
};
xhr.send();
```

Problem Statement:

Create a registration page having fields like Name, College, Username and Password (read password twice).
Validate the form by checking for
1. Usernameis not same as existing entries
2. Name field is not empty
3. Retyped password is matching with the earlier one. Prompt a message is
And also auto suggest college names.
Show the message "Successfully Registered" on the same page below the submit button, on Successfully registration. Let all the updations on the page be Asynchronously loaded. Implement the same using XMLHttpRequest Object.

CODE:-

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Registration Form</title>
   <style>
      body {
         font-family: Arial, sans-serif;
         max-width: 600px;
         margin: 0 auto;
         padding: 20px;
      }
      .form-group {
         margin-bottom: 15px;
      }
      label {
         display: block;
         margin-bottom: 5px;
         font-weight: bold;
      }
      input, select {
         width: 100%;
         padding: 8px;
         box-sizing: border-box;
         border: 1px solid #ddd;
         border-radius: 4px;
      }
      button {
         background-color: #4CAF50;
         color: white;
         padding: 10px 15px;
         border: none;
         border-radius: 4px;
         cursor: pointer;
```

```
            font-size: 16px;
        }
        button:hover {
            background-color: #45a049;
        }
        .error {
            color: red;
            font-size: 14px;
            margin-top: 5px;
        }
        .success-text {
            color: green;
            font-size: 14px;
            margin-top: 5px;
        }
        .success-message {
            color: green;
            font-weight: bold;
            margin-top: 15px;
            display: none;
            text-align: center;
            padding: 10px;
            background-color: #f0fff0;
            border: 1px solid #80c080;
            border-radius: 4px;
        }
        #collegeOptions {
            position: absolute;
            background: white;
            border: 1px solid #ddd;
            max-height: 150px;
            overflow-y: auto;
            width: calc(100% - 40px);
            z-index: 100;
            display: none;
        }
        .college-option {
            padding: 8px;
            cursor: pointer;
        }
        .college-option:hover {
            background-color: #f1f1f1;
        }
    </style>
</head>
<body>
    <h1>Registration Form</h1>

    <form id="registrationForm">
```

```
        <div class="form-group">
            <label for="name">Name*</label>
            <input type="text" id="name" name="name" required>
            <div id="nameError" class="error"></div>
        </div>

        <div class="form-group">
            <label for="college">College*</label>
            <input type="text" id="college" name="college" autocomplete="off" required>
            <div id="collegeOptions"></div>
        </div>

        <div class="form-group">
            <label for="username">Username*</label>
            <input type="text" id="username" name="username" required>
            <div id="usernameError" class="error"></div>
            <div id="usernameSuccess" class="success-text"></div>
        </div>

        <div class="form-group">
            <label for="password">Password*</label>
            <input type="password" id="password" name="password" required>
        </div>

        <div class="form-group">
            <label for="confirmPassword">Confirm Password*</label>
            <input type="password" id="confirmPassword" name="confirmPassword"
required>
            <div id="passwordError" class="error"></div>
        </div>

        <button type="submit">Register</button>
    </form>

    <div id="successMessage" class="success-message">Successfully
Registered!</div>

    <script>
        // Sample data for existing usernames
        const existingUsernames = ['john123', 'alice456', 'bob789', 'user1', 'admin'];

        // Sample data for college suggestions
        const colleges = [
            'VESIT',
            'SPIT,Andheri',
            'VJTI',
            'VIT,Pune',
            'Princeton University',
            'Yale University',
```

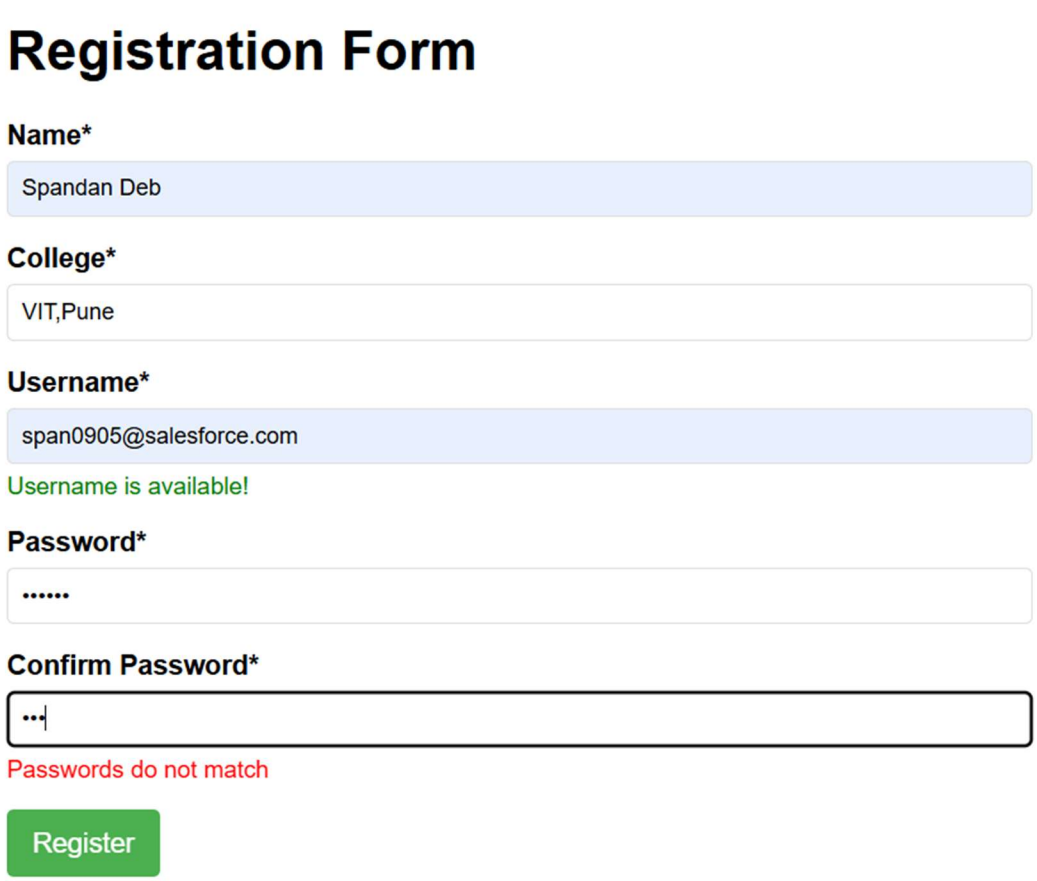
```
    'Columbia University',
    'University of Chicago',
    'University of Pennsylvania',
    'Cornell University',
    'Duke University',
    'Johns Hopkins University',
    'Northwestern University',
    'Brown University',
    'University of California, Berkeley',
    'University of California, Los Angeles',
    'University of Michigan',
    'New York University',
    'Boston University',
    'Carnegie Mellon University'
];

const form = document.getElementById('registrationForm');
const nameInput = document.getElementById('name');
const collegeInput = document.getElementById('college');
const usernameInput = document.getElementById('username');
const passwordInput = document.getElementById('password');
const confirmPasswordInput = document.getElementById('confirmPassword');
const collegeOptions = document.getElementById('collegeOptions');
const successMessage = document.getElementById('successMessage');
const nameError = document.getElementById('nameError');
const usernameError = document.getElementById('usernameError');
const usernameSuccess = document.getElementById('usernameSuccess');
const passwordError = document.getElementById('passwordError');

// Real-time name validation
nameInput.addEventListener('input', function() {
    if (this.value.trim() === '') {
        nameError.textContent = 'Name is required';
    } else {
        nameError.textContent = '';
    }
});

// College autocomplete functionality
collegeInput.addEventListener('input', function() {
    const value = this.value.toLowerCase();
    collegeOptions.innerHTML = '';

    if (value.length < 2) {
        collegeOptions.style.display = 'none';
        return;
    }

    const matchingColleges = colleges.filter(college =>
```

```
            college.toLowerCase().includes(value)
        );

        if (matchingColleges.length > 0) {
            matchingColleges.forEach(college => {
                const option = document.createElement('div');
                option.className = 'college-option';
                option.textContent = college;
                option.addEventListener('click', function() {
                    collegeInput.value = college;
                    collegeOptions.style.display = 'none';
                });
                collegeOptions.appendChild(option);
            });
            collegeOptions.style.display = 'block';
        } else {
            collegeOptions.style.display = 'none';
        }
    });

    // Hide college options when clicking elsewhere
    document.addEventListener('click', function(event) {
        if (event.target !== collegeInput) {
            collegeOptions.style.display = 'none';
        }
    });

    // Real-time username validation
    usernameInput.addEventListener('input', function() {
        checkUsername();
    });

    function checkUsername() {
        const username = usernameInput.value.trim();

        if (username === '') {
            usernameError.textContent = 'Username is required';
            usernameSuccess.textContent = '';
            return;
        }

        // Use XMLHttpRequest to check if username exists
        const xhr = new XMLHttpRequest();
        xhr.open('POST', 'validate-username.php', true);
        xhr.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');

        xhr.onreadystatechange = function() {
            if (xhr.readyState === 4) {
                // For demo purposes, we'll simulate the server response
```

```
                if (existingUsernames.includes(username)) {
                    usernameError.textContent = 'Username already exists. Please choose another one.';
                    usernameSuccess.textContent = '';
                } else {
                    usernameError.textContent = '';
                    usernameSuccess.textContent = 'Username is available!';
                }
            }
        };

        xhr.send('username=' + encodeURIComponent(username));
    }

    // Real-time password matching validation
    confirmPasswordInput.addEventListener('input', function() {
        checkPasswordsMatch();
    });

    passwordInput.addEventListener('input', function() {
        if (confirmPasswordInput.value !== '') {
            checkPasswordsMatch();
        }
    });

    function checkPasswordsMatch() {
        if (passwordInput.value !== confirmPasswordInput.value) {
            passwordError.textContent = 'Passwords do not match';
        } else {
            passwordError.textContent = '';
        }
    }

    // Form submission
    form.addEventListener('submit', function(event) {
        event.preventDefault();

        let isValid = true;

        // Validate name (not empty)
        if (nameInput.value.trim() === '') {
            nameError.textContent = 'Name is required';
            isValid = false;
        }

        // Validate username
        if (usernameInput.value.trim() === '') {
            usernameError.textContent = 'Username is required';
            isValid = false;
```

```
        } else if (existingUsernames.includes(usernameInput.value)) {
            usernameError.textContent = 'Username already exists. Please choose
another one.';
            isValid = false;
        }

        // Validate password match
        if (passwordInput.value !== confirmPasswordInput.value) {
            passwordError.textContent = 'Passwords do not match';
            isValid = false;
        }

        if (isValid) {
            // Simulate form submission using XMLHttpRequest
            const xhr = new XMLHttpRequest();
            xhr.open('POST', 'register.php', true);
            xhr.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');

            xhr.onreadystatechange = function() {
                if (xhr.readyState === 4) {
                    // In a real app, you would check the response from the server
                    // For this demo, we'll simulate a successful registration
                    successMessage.style.display = 'block';
                    form.reset();
                    usernameSuccess.textContent = '';

                    // Hide success message after 5 seconds
                    setTimeout(function() {
                        successMessage.style.display = 'none';
                    }, 5000);
                }
            };

            const formData = new FormData(form);
            const urlEncodedData = new URLSearchParams(formData).toString();
            xhr.send(urlEncodedData);
        }
    });
  </script>
</body>
</html>
```

OUTPUT:


Registration Form
Name*
Spandan Deb
College*
VIT,Pune
Username*
span0905@salesforce.com
Username is available!
Password*
Confirm Password*
Passwords do not match
Register

Registration Form

Name*

Spandan Deb

College*

VIT,Pune

Username*

span0905@salesforce.com

Username is available!

Password*

••••••

Confirm Password*

••••••

Register

Registration Form

Name*

Spandan Deb

College*

VI

VIT,Pune

VIT,Vellore

Password*

Confirm Password*

Register

Successfully Registered!