Name : Prajyot Shinde **Class: D15A** **Roll No. 55**

Experiment – 6: AJAX

Aim: To study AJAX

Theory:

1) How do Synchronous and Asynchronous Requests differ?

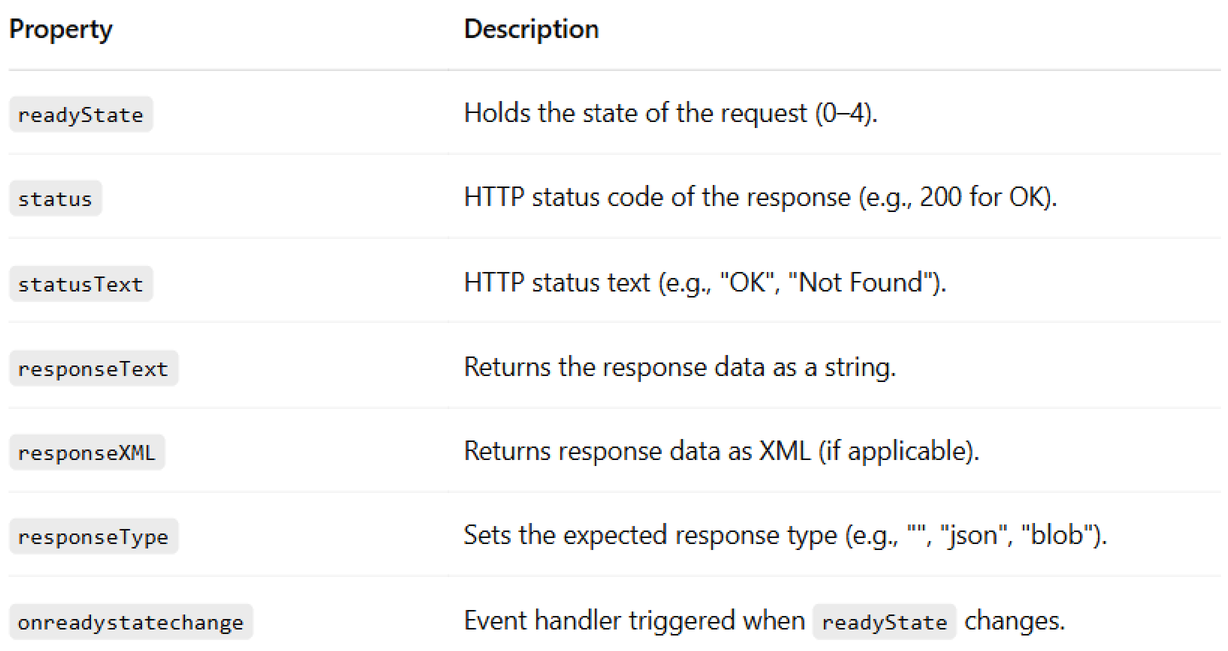
Ans:

In a Synchronous request, the request sent from the client receives the response in the same HTTP connection. Examples are responses from Internet gateway, phone calls, and video meetings.

Whereas for Asynchronous requests, multiple requests can be sent from clients and their responses can be received in subsequent connections. Examples are collaborative documents like assessments, online queries, emails, and online forums.

2) Describe various properties and methods used in XMLHttpRequest Object

Ans:



Property	Description
`readyState`	Holds the state of the request (0–4).
`status`	HTTP status code of the response (e.g., 200 for OK).
`statusText`	HTTP status text (e.g., "OK", "Not Found").
`responseText`	Returns the response data as a string.
`responseXML`	Returns response data as XML (if applicable).
`responseType`	Sets the expected response type (e.g., "", "json", "blob").
`onreadystatechange`	Event handler triggered when `readyState` changes.

Problem Statement:

Create a registration page having fields like Name, College, Username and Password (read password twice).

Validate the form by checking for

1. Username is not same as existing entries

2 Retyped password is matching with the earlier one. Prompt a message is

3 Auto suggest college names.

Name : Prajyot Shinde Class: D15A Roll No. 55

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.

Output:

register.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Registration Form</title>
  <style>
    body {
      font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
      background-color: #f4f7f8;
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
      margin: 0;
    }

    .form-container {
      background-color: #ffffff;
      padding: 30px 40px;
      border-radius: 12px;
      box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);
      width: 100%;
      max-width: 400px;
    }

    h2 {
      text-align: center;
      margin-bottom: 25px;
      color: #333;
    }

    input {
      width: 100%;
      padding: 12px 14px;
      margin-bottom: 15px;
      border: 1px solid #ccc;
      border-radius: 8px;
      font-size: 15px;
      transition: border 0.3s ease;
    }

    input:focus {
      border-color: #007bff;
      outline: none;
    }

    button {
      width: 100%;
      padding: 12px;
      background-color: #007bff;
      color: #fff;
      border: none;
      font-size: 16px;
      border-radius: 8px;
      cursor: pointer;
      transition: background-color 0.3s ease;
    }

    button:hover {
      background-color: #0056b3;
    }

    #message {
      text-align: center;
      margin-top: 15px;
      font-weight: bold;
      font-size: 15px;
    }
  </style>
</head>
<body>

  <div class="form-container">
    <h2>Register</h2>

    <form id="registerForm">
      <input type="text" id="name" placeholder="Full Name" required />

      <input type="text" id="college" placeholder="College" list="collegeList" autocomplete="off"
required />
```

```
      <datalist id="collegeList"></datalist>

      <input type="text" id="username" placeholder="Username" required />

      <input type="password" id="password" placeholder="Password" required />
      <input type="password" id="confirmPassword" placeholder="Confirm Password" required />

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

    <div id="message"></div>
  </div>

  <script src="script.js"></script>
</body>
</html>
```

colleges.json

```
["VESIT", "DJ Sanghvi", "SPIT", "KJ Somaiya", "Thakur College"]
```

usernames.json

```
["prajyot", "tejas", "varun"]
```

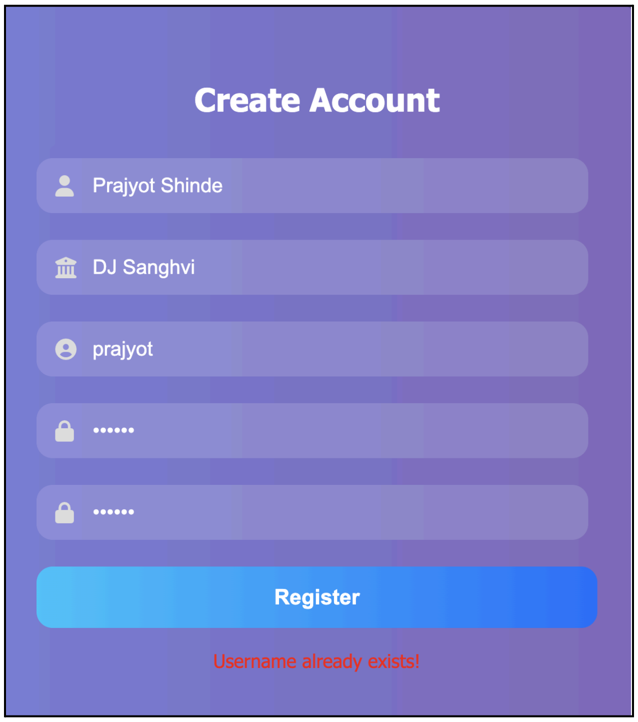
script.js

```
// Fetch colleges and populate datalist
function loadColleges() {
    const xhr = new XMLHttpRequest();
    xhr.open('GET', 'colleges.json', true);
    xhr.onload = function() {
      if (xhr.status === 200) {
        const colleges = JSON.parse(xhr.responseText);
        const list = document.getElementById('collegeList');
        list.innerHTML = "";
        colleges.forEach(college => {
          const option = document.createElement('option');
          option.value = college;
          list.appendChild(option);
        });
      }
    };
    xhr.send();
  }

  // Check if username exists
```

```
function checkUsernameExists(username, callback) {
  const xhr = new XMLHttpRequest();
  xhr.open('GET', 'usernames.json', true);
  xhr.onload = function() {
    if (xhr.status === 200) {
      const existingUsernames = JSON.parse(xhr.responseText);
      callback(existingUsernames.includes(username));
    }
  };
  xhr.send();
}

// Handle form submission
document.getElementById('registerForm').addEventListener('submit', function(e) {
  e.preventDefault();

  const name = document.getElementById('name').value;
  const college = document.getElementById('college').value;
  const username = document.getElementById('username').value;
  const password = document.getElementById('password').value;
  const confirmPassword = document.getElementById('confirmPassword').value;
  const messageBox = document.getElementById('message');

  if (password !== confirmPassword) {
    messageBox.style.color = 'red';
    messageBox.textContent = "Passwords do not match!";
    return;
  }

  checkUsernameExists(username, function(exists) {
    if (exists) {
      messageBox.style.color = 'red';
      messageBox.textContent = "Username already exists!";
    } else {
      // Simulate registration
      messageBox.style.color = 'green';
      messageBox.textContent = "Successfully Registered";
    }
  });
});

// Initial load
loadColleges();
```

Name : Prajyot Shinde **Class: D15A** **Roll No. 55**

1) Username already exists



Create Account

Prajyot Shinde

DJ Sanghvi

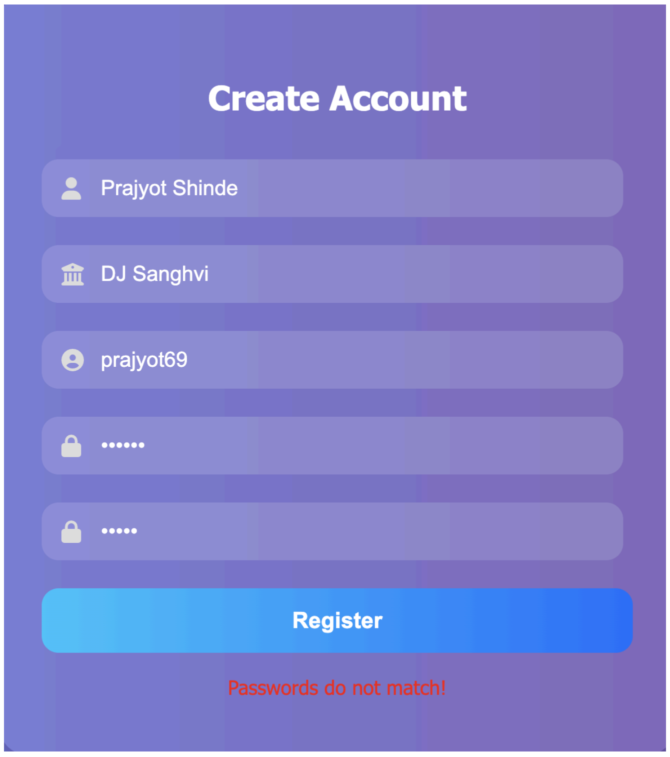
prajyot

••••••

••••••

Username already exists!

2) Passwords do not match



Create Account

Prajyot Shinde

DJ Sanghvi

prajyot69

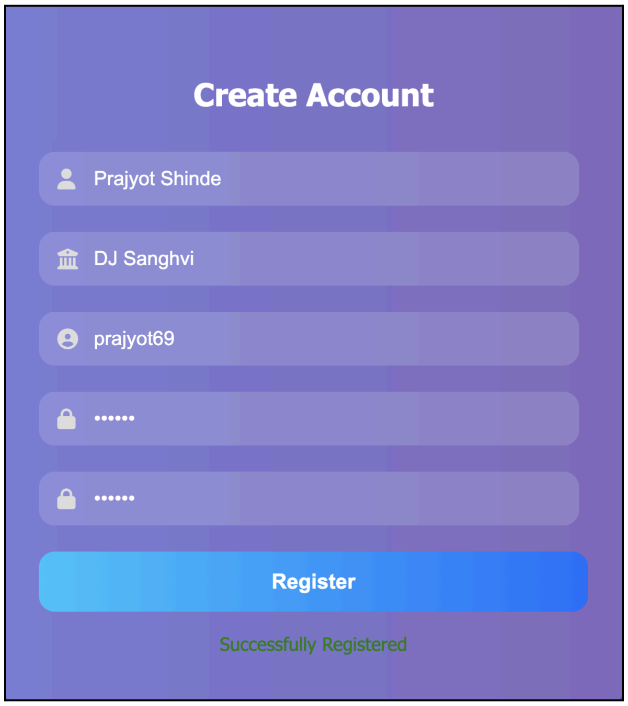
••••••

•••••

Passwords do not match!

Name : Prajyot Shinde **Class: D15A** **Roll No. 55**

3) Successfully registered



Create Account

Prajyot Shinde

DJ Sanghvi

prajyot69

••••••

••••••

Successfully Registered

Conclusion:
This experiment helped in understanding how to create a dynamic registration form using XMLHttpRequest for asynchronous operations. It demonstrated validation techniques such as password matching and checking for existing usernames. College name autosuggestion improved user experience through real-time data loading.