Experiment – 9: AJAX

Name of Student	**Sneha Patra**
Class Roll No	**D15A_40**
D.O.P.	**03/04/2025**
D.O.S.	**10/04/2025**
Sign and Grade	

Aim: To study AJAX

Theory:

1. **How do Synchronous and Asynchronous Requests differ?**

Feature	Synchronous	Asynchronous
Blocking	Blocks code execution until request completes	Doesn't block code execution
Performance	Slower, UI may freeze	Faster, smoother user experience
Usage	Not recommended in modern web apps	Preferred for web development
Example	`xhr.open("GET", url, false);`	`xhr.open("GET", url, true`

2. **Describe various properties and methods used in XMLHttpRequest Object**

Properties:

- `xhr.readyState` – Status of the request (0 to 4)
- `xhr.status` – HTTP status code (e.g., 200 = OK)
- `xhr.responseText` – Response data as text
- `xhr.responseXML` – Response data as XML (if available)

🛠️ **Methods:**

1) `xhr.open(method, url, async)` – Initializes a request

2) `xhr.send(data)` – Sends the request

3) `xhr.setRequestHeader(header, value)` – Sets custom request headers

4) `xhr.abort()` – Cancels the reques

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.

Output:

Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>AJAX Registration Form</title>
  <style>
    body {
      font-family: 'Segoe UI', sans-serif;
```

```
  background: #f3f4f6;
  margin: 0;
  padding: 20px;
}
.container {
  background: white;
  max-width: 500px;
  margin: 50px auto;
  padding: 30px;
  border-radius: 10px;
  box-shadow: 0 8px 16px rgba(0,0,0,0.1);
}
h2 {
  text-align: center;
  color: #0d47a1;
}
label {
  font-weight: bold;
  margin-top: 15px;
  display: block;
}
input[type="text"],
input[type="password"],
```

```
input[list] {
  width: 100%;
  padding: 10px;
  margin-top: 6px;
  border: 1px solid #ccc;
  border-radius: 5px;
  box-sizing: border-box;
}
button {
  margin-top: 20px;
  width: 100%;
  padding: 12px;
  background-color: #1976d2;
  color: white;
  border: none;
  border-radius: 5px;
  font-size: 16px;
  cursor: pointer;
}
button:hover {
  background-color: #1565c0;
}
.feedback {
```

```
      font-size: 13px;
      color: #d32f2f;
      margin-top: 3px;
    }
    .success {
      color: #388e3c;
      margin-top: 20px;
      text-align: center;
    }
  </style>
</head>
<body>
  <div class="container">
    <h2>Register</h2>
    <form id="registrationForm" onsubmit="return false;">
      <label for="name">Name:</label>
      <input type="text" id="name">
      <div id="nameFeedback" class="feedback"></div>

      <label for="college">College:</label>
      <input list="colleges" id="college">
      <datalist id="colleges">
        <option value="VESIT">
```

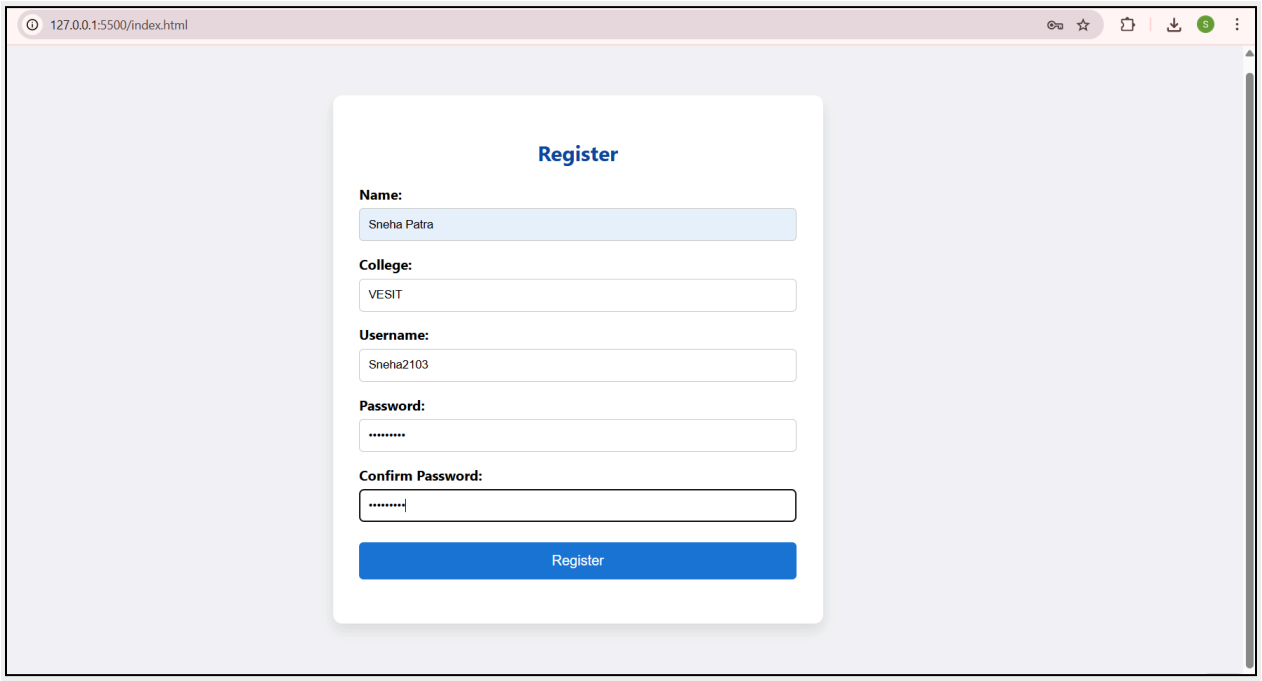
```
      <option value="IIT Bombay">
      <option value="VIT">
      <option value="MIT">
      <option value="BITS Pilani">
      <option value="University of Mumbai">
    </datalist>

    <label for="username">Username:</label>
    <input type="text" id="username">
    <div id="usernameFeedback" class="feedback"></div>

    <label for="password">Password:</label>
    <input type="password" id="password">

    <label for="confirmPassword">Confirm Password:</label>
    <input type="password" id="confirmPassword">
    <div id="passwordFeedback" class="feedback"></div>

    <button type="button" onclick="submitForm()">Register</button>
  </form>
  <div id="successMessage" class="success"></div>
</div>
```

```
<script>
  const existingUsernames = ["sneha123", "admin", "testuser"];

  function checkUsernameAsync(username, callback) {
    // Simulate an AJAX call with a small delay
    setTimeout(() => {
      const lower = username.toLowerCase();
      const exists = existingUsernames.some(u => u.toLowerCase() === lower);
      callback(!exists);
    }, 300); // simulate network delay
  }

  function submitForm() {
    const name = document.getElementById("name").value.trim();
    const username = document.getElementById("username").value.trim();
    const password = document.getElementById("password").value;
    const confirmPassword = document.getElementById("confirmPassword").value;

    const nameFeedback = document.getElementById("nameFeedback");
    const usernameFeedback = document.getElementById("usernameFeedback");
    const passwordFeedback = document.getElementById("passwordFeedback");
    const successMessage = document.getElementById("successMessage");
```

```
// Clear all previous messages
nameFeedback.textContent = "";
usernameFeedback.textContent = "";
passwordFeedback.textContent = "";
successMessage.textContent = "";

let valid = true;

if (name === "") {
  nameFeedback.textContent = "Name cannot be empty.";
  valid = false;
}

if (password !== confirmPassword) {
  passwordFeedback.textContent = "Passwords do not match.";
  valid = false;
}

if (!valid) return;

// Async check for username availability
checkUsernameAsync(username, function (isAvailable) {
  if (!isAvailable) {
```

```
        usernameFeedback.textContent = "Username already taken.";
        successMessage.textContent = "";
      } else {
        usernameFeedback.textContent = "";
        successMessage.textContent = "Successfully Registered!";
      }
    });
  }
</script>

</body>
</html>
```

Screenshot of Output



Register

Name:

Name cannot be empty.

College:

VESIT

Username:

sneha_123

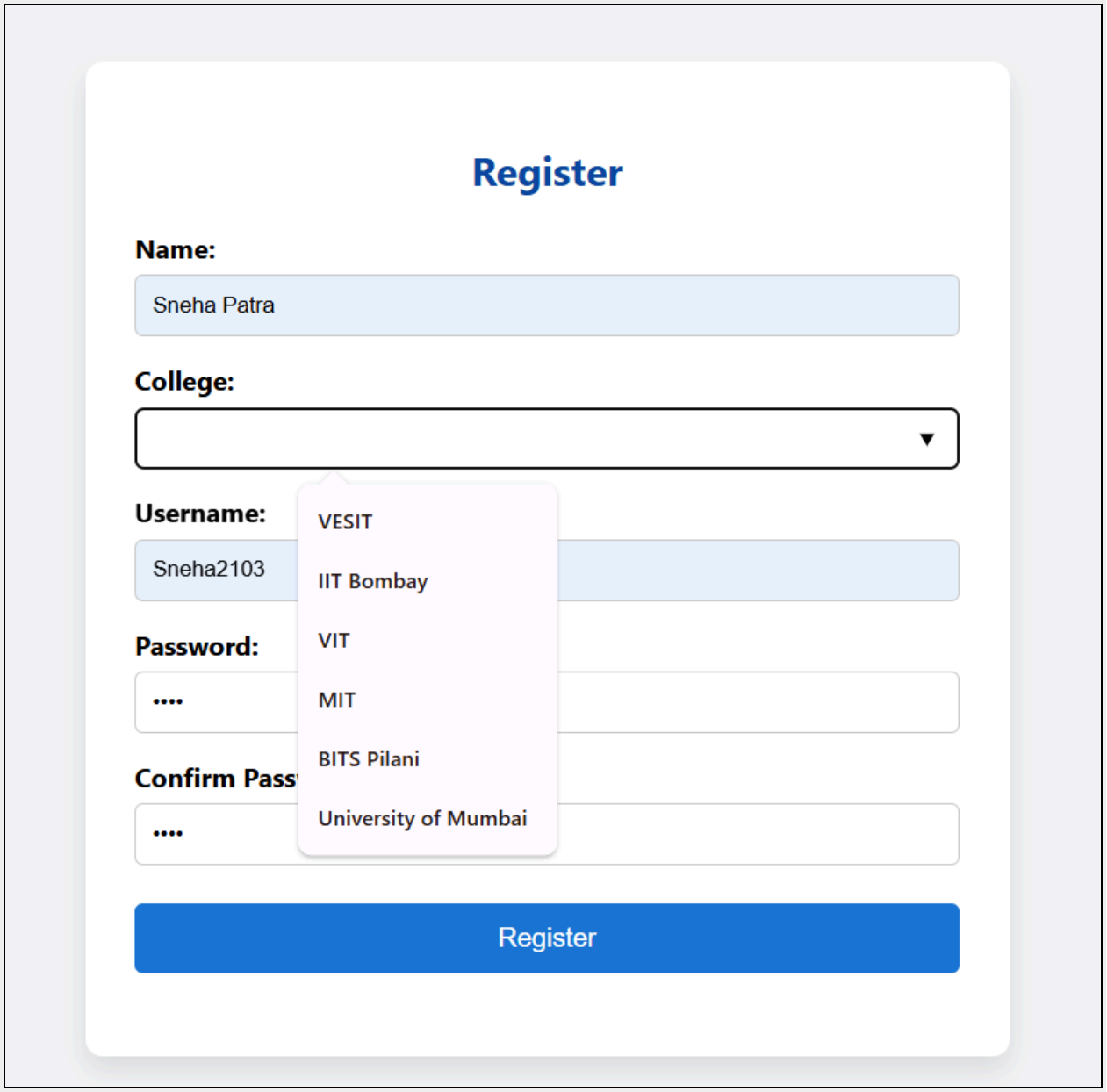
Password:

••••

Confirm Password:

••••

Register


Register
Name:
Sneha Patra
College:
Username:
Sneha2103
Password:
Confirm Pass
VESIT
IIT Bombay
VIT
MIT
BITS Pilani
University of Mumbai
Register

Register

Name:

Sneha Patra

College:

VESIT

Username:

sneha123

Username already taken.

Password:

•••

Confirm Password:

•••

Register

Register

Name:

Sneha Patra

College:

VESIT

Username:

sneha123

Password:

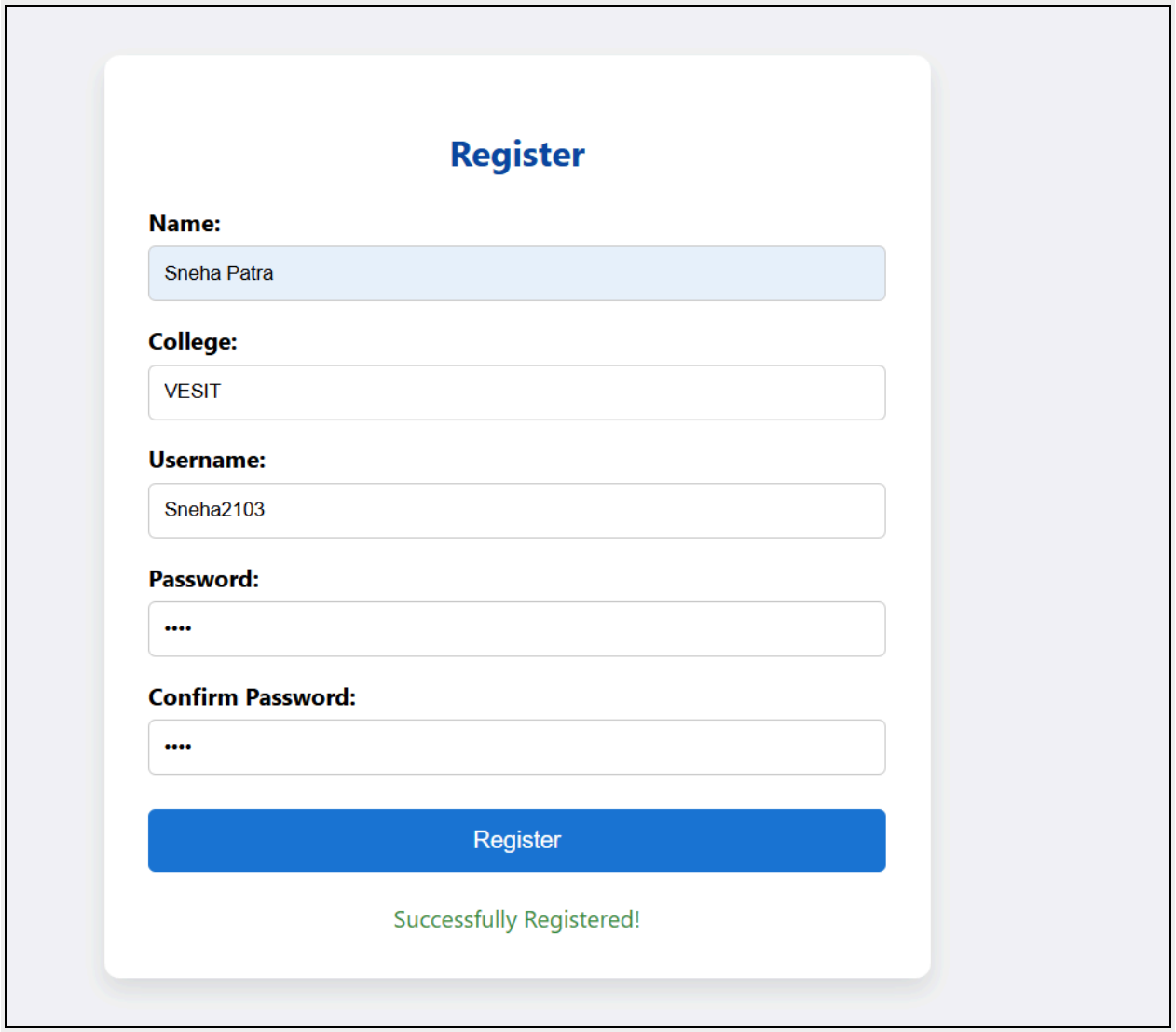
••••

Confirm Password:

•••

Passwords do not match.

Register



Conclusion:

Synchronous requests block the browser, while asynchronous requests run in the background without interrupting the user. `XMLHttpRequest` provides methods to make these requests and properties to handle responses, forming the base of AJAX functionality.