UIT2602 WEB PROGRAMMING

Ex. No 07: AJAX Integration

**Name:** Sabarish Sankaran B

**Class:** IT-B

**Roll No:** 3122215002087

# Aim:

To demonstrate the integration of AJAX (Asynchronous JavaScript and XML) into a web application, enabling dynamic and interactive features such as live search, dynamic form submissions, and updating parts of the page without refreshing.

# Procedure and Code:

Step 1: Setting Up the Files Create the following files in your project directory:

* search.html
* search.php
* submit.php
* dynamic-content.php

Step 2: Include the jQuery Library

<script

src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

Step 3: Live Search: Create input

<h2>Live Search</h2>

<!-- Step 3: Live Search - Create input and search results container -->

<input type="text" id="searchInput" placeholder="Search...">

<div id="searchResults"></div>

Step 4: Attach an event listener to the input field to trigger an AJAX request whenever the user types in the search box

// Submit search form

$('#searchInput').on('keyup', function(){ var searchText = $(this).val(); if(searchText !== ''){

$.ajax({

url: 'search.php', method: 'POST',

data: {query: searchText}, dataType: 'json',

success: function(response){ displayResults(response);

}

});

} else {

$('#searchResults').html('');

}

});

Step 5: Dynamic Form Submission

<h2>Dynamic Form Submission</h2>

<!-- Step 5: Dynamic Form Submission - Create form -->

<form id="myForm">

<input type="text" name="username" placeholder="Username">

<input type="email" name="email" placeholder="Email">

<button type="submit">Submit</button>

</form>

Step 6: Intercept the form submission using AJAX to prevent the default behavior, then submit the form data asynchronously.

// Step 6: Intercept the form submission using AJAX to prevent the default behavior, then submit the form data asynchronously

$('#myForm').submit(function(event) {

event.preventDefault(); // Prevent default form submission var formData = $(this).serialize();

$.ajax({

url: 'submit.php', // URL to your server-side script handling form

submission

method: 'POST', data: formData,

success: function(response) {

// Handle success response console.log(response);

$('#submitResponse').html(response); // Display the response

},

error: function(xhr, status, error) {

// Handle error response console.error(error);

}

});

});

Step 7: Updating Parts of the Page Without Refreshing

<!-- Step 7: Updating Parts of the Page Without Refreshing -->

<h2>Dynamic Content</h2>

<div id="dynamicContent"></div>

Step 8: Use AJAX to fetch new content from the server and update the container periodically.

// Step 8: Use AJAX to fetch new content from the server and update the container periodically

function updateDynamicContent() {

$.ajax({

url: 'dynamic-content.php', // URL to your server-side script returning dynamic content

method: 'GET',

success: function(response) {

$('#dynamicContent').html(response);

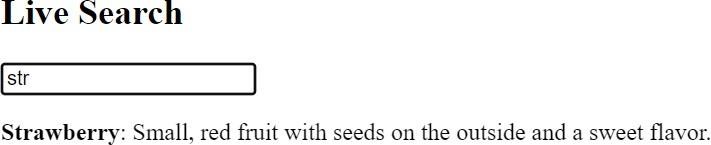
}

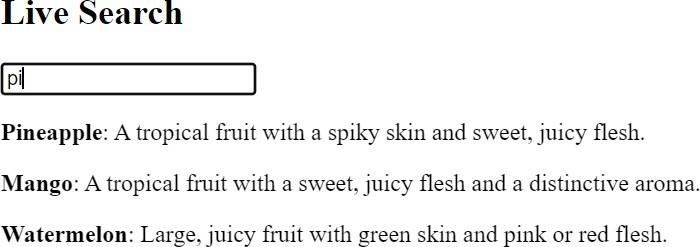
});

}

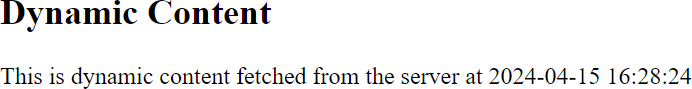
# Output:

1. **Live Search**





1. **Dynamic Content**



# Conclusion:

Incorporating AJAX into web applications allows for enhanced user experience by enabling dynamic and interactive features. AJAX facilitates smoother communication between the client and server, resulting in a more responsive and engaging user interface.