AJAX examples

The provided code handles the submission of a form and makes an asynchronous request to a specified URL using different AJAX techniques (jQuery, Axios, Fetch, and plain JavaScript). It also displays the response received from the server.

```
document.addEventListener("DOMContentLoaded",
function(event) {

document.querySelector('input[name="sender"]').addEvent
Listener('click', makeRequest);
});

var method = document.getElementById('method');
method.addEventListener('change', function() {
    var myForm = document.getElementById('myForm');
    myForm.style.display = method.value === 'POST' ?
'block' : 'none';
});

function makeRequest() {
    var formd = new
FormData(document.getElementById('myForm'));
    var resource =
document.getElementById('resource').value;
    var url =
document.querySelector('input[name="url"]').value;
```

```
    switch (resource) {
        case 'JQuery':
            ajaxJQ(url, formd);
            break;
        case 'Axios':
            ajaxAx(url, formd);
            break;
        case 'Fetch':
            ajaxFe(url, formd);
            break;
        default:
            ajaxJS(url, formd);
    }
}

function ajaxJS(url, formd) {
    var xhr = new XMLHttpRequest();
    xhr.onreadystatechange = function() {
        if (xhr.readyState === 4) {
            var responseStatus = xhr.status;
            var responseText = xhr.responseText;
            var message = 'JS error';

            if (responseStatus === 200) {
                message = 'JS response';
            } else if (responseStatus === 201) {
                message = 'JS added';
            }
```

```
            output(responseText, responseStatus,
message);
        }
    };

    xhr.open(method.value, url, true);

    if (method.value === 'POST') {
        xhr.setRequestHeader('Content-Type',
'application/x-www-form-urlencoded');
        xhr.send(new
URLSearchParams(formd).toString());
    } else {
        xhr.send();
    }
}

function ajaxJQ(url, formd) {
    $.ajax({
        url: url,
        type: method.value,
        data: formd,
        success: function(data) {},
    }).done(function(response, textStatus, xhr) {
        output(response, xhr.status, 'JQ success');
    }).fail(function(response, textStatus, xhr) {
        output(response, xhr.status, 'JQ error');
    });
}
```

```
function ajaxAx(url, formd) {
    axios({
        method: method.value,
        url: url,
        data: formd,
    }).then(function(response) {
        output(response.data, response.status, 'Axios
success');
    }).catch(function(error) {
        console.log(error);
        output(error.response.data,
error.response.status, 'Axios error');
    });
}

function ajaxFe(url, formd) {
    var options = {
        method: method.value,
        headers: { 'Content-Type':
'application/x-www-form-urlencoded' },
    };

    if (method.value === 'POST') {
        options.body = new
URLSearchParams(formd).toString();
    }

    fetch(url, options)
        .then(function(response) {
            return response.json();
```

```
        })
        .catch(function(error) {
            return output(error, 0, 'Fetch failed');
        })
        .then(function(response) {
            return output(response, 200, 'Fetch
success');
        });
}

function output(response, status, message) {
    console.log(response);
    var output = document.getElementById('output');
    output.innerHTML = '';
    response = typeof response === 'string' ?
JSON.parse(response) : response;
    output.innerHTML += '<div class="a">' +
JSON.stringify(response) + '</div>';
    output.innerHTML += '<div class="b">' + status +
'</div>';
    output.innerHTML += '<div class="c">' + message +
'</div>';
}

// Add an event listener for form submission
document.getElementById('myForm').addEventListener('sub
mit', function(event) {
    event.preventDefault(); // Prevent form submission

    makeRequest();
```

```
});
```

In the above code, we added an event listener to the form element with the id "myForm". The event listener is triggered when the form is submitted. We used the preventDefault() method to prevent the default form submission behavior, which allows us to handle the submission through the makeRequest() function.

Now, when the user submits the form, the makeRequest() function will be called, and the request will be made based on the selected resource (JQuery, Axios, Fetch, or plain JavaScript), the chosen HTTP method, and the entered URL.

Please note that this assumes you have an HTML form with the id "myForm" and appropriate input elements like "sender", "method", "resource", "url", etc., in your HTML markup. You may need to modify the HTML and JavaScript code to match your specific requirements.