```
1
2    //****************************Homework 8*************************************** */
3    // Function to handle the AJAX process for loading CD data
4    // The following function to load CD data using the Fetch API - Failed to work!
5    /*function loadCDData() {
6        // Making a GET request to the specified URL
7        fetch("AjaxAssignment/cd_catalog.xml")
8        .then(response => {
9            // Checking if the response is successful
10           if (!response.ok) {
11               throw new Error('Network response was not ok');
12           }
13           // Parsing the response as text
14           return response.text();
15       })
16       .then(str => {
17           // Parsing the string response to an XML Document
18           const parser = new DOMParser();
19           const xmlDoc = parser.parseFromString(str, "application/xml");
20           // Updating the table with data from the XML
21           updateTable(xmlDoc);
22       })
23       .catch(error => {
24           console.error('There has been a problem with your fetch operation:', error);
25
26           // Selecting the element where the error message will be displayed
27           const errorMessageElement = document.getElementById("demo");
28
29           // Setting the inner HTML to the error message
30           errorMessageElement.innerHTML = 'Failed to load CD data.';
31
32           // Changing the style of the element to have white text
33           errorMessageElement.style.color = 'white';
34           // Additional styling can be added here if needed
35       });
36   }
37
38   Function to update the table with CD data
39   function updateTable(xmlDoc) {
40       // Getting all CD elements from the XML document
41       const cds = xmlDoc.getElementsByTagName("CD");
42       // Starting the table HTML with headers
43       let table = "<tr><th>Artist</th><th>Year</th></tr>";
44
45       // Looping through each CD element and adding its data to the table
46       for (let cd of cds) {
47           // Getting the artist and year from the CD element
48           const artist = cd.getElementsByTagName("ARTIST")[0].textContent;
49           const year = cd.getElementsByTagName("YEAR")[0].textContent;
50           // Appending a table row for each CD
51           table += `<tr><td>${artist}</td><td>${year}</td></tr>`;
52       }
53
54       // Setting the inner HTML of the table element to the constructed table
55       document.getElementById("demo").innerHTML = table;
56   }
57   */
58
59   //Following Code Copied from given ajaxassignment code  making changes in year from title
60   function loadDoc() {
61       const xhttp = new XMLHttpRequest();
62       xhttp.onload = function() {
63       myFunction(this);
64       }
65       xhttp.open("GET", "AjaxAssignment/cd_catalog.xml");
66       xhttp.send();
67   }
68   function myFunction(xml) {
69       const xmlDoc = xml.responseXML;
```

```
 70          const x = xmlDoc.getElementsByTagName("CD");
 71          let table="<tr><th>Artist</th><th>Year</th></tr>";
 72          for (let i = 0; i <x.length; i++) {
 73          table += "<tr><td>" +
 74          x[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +
 75          "</td><td>" +
 76          x[i].getElementsByTagName("YEAR")[0].childNodes[0].nodeValue +
 77          "</td></tr>";
 78          }
 79          document.getElementById("demo").innerHTML = table;
 80      }
 81
 82
 83      // Function to create and display an ordered list of process steps
 84      document.getElementById('showProcess').addEventListener('click', function() {
 85          // Define the process steps as an array of strings
 86          const processSteps = [
 87              "Making the Order (Sending the AJAX Request): <br>You decide you're hungry for
                 some data. This is like picking up your phone and dialing the pizza place to
                 order pizza.",
 88              "Waiting for the Pizza (Waiting for the Response): <br>After placing your order,
                 you go back to your activities. Meanwhile, the pizza place prepares your order.",
 89              "Pizza is Ready! (Receiving the Response): <br>The pizza place lets you know
                 that your pizza is ready. You check to make sure it's the right order.",
 90              "Unboxing the Pizza (Processing the Data): <br>You open the pizza box and check
                 out each slice. This is like processing the XML data.",
 91              "Enjoying Your Meal (Displaying the Data): <br>You arrange the pizza slices on
                 plates, similar to how the data is arranged in a readable format on the
                 webpage.",
 92              "Sharing the Pizza (Updating the Webpage): <br>You invite everyone to eat. The
                 webpage is updated with the new data.",
 93              "Handling Pizza Mishaps (Error Handling): <br>If something goes wrong, like
                 receiving the wrong pizza, you know how to handle it. This is like error
                 checking in your code."
 94          ];
 95
 96          // Create an ordered list element
 97          const processList = document.createElement('ol');
 98
 99          // For each step, create a list item, set its innerHTML, and append it to the list
100          processSteps.forEach(step => {
101              const listItem = document.createElement('li');
102              listItem.innerHTML = step;
103              processList.appendChild(listItem);
104          });
105
106          // Get the container element and set its content to the process list
107          const container = document.getElementById('processContainer');
108          container.innerHTML = ''; // Clear any previous content
109          container.appendChild(processList); // Add the process list to the container
110      });
111
112
113      //*****************************************REST API*****************************************
         */
114
115      // Adding an event listener to the button with the ID 'showReport'
116      document.getElementById('showReport').addEventListener('click', function() {
117          // Defining the content of the report as a string literal
118          const reportContent = `
119          <p><strong>Enhancing Heart Pizza's Online System with CalorieNinjas API</p>
120          <p>Purpose:</strong><br>We are poised to take Heart Pizza's online ordering system
             to the next level by integrating the CalorieNinjas API. This tool provides essential
             nutritional information, including calorie counts, for all our pizza ingredients,
             enriching our customers' experience with valuable health insights.</p>
121          <p><strong>JSON Response Format:</strong><br>The API offers data in JSON format,
             ensuring compatibility with modern web technologies and facilitating easy data
             handling in our web application.</p>
122          <p><strong>API Key Requirement:</strong><br>Access to the API is secured with an API
```

```
key, providing a layer of protection and ensuring that our data transactions are
safe and authenticated.</p>
123    <p><strong>Cost-Effectiveness:</strong><br>The API's flexible pricing structure
allows us to start with basic, free features, with the option to scale up as our
needs grow and evolve.</p>
124    <p><strong>Comprehensive Documentation:</strong><br>Detailed <a
href='https://calorieninjas.com' target='_blank'>documentation</a> is available to
guide our developers through seamless integration and effective utilization of the
API.</p>
125    <p>By adopting the CalorieNinjas API, we are not just improving our service
offerings but also moving towards a more informed and health-conscious business
model.</p>
126  `;
127
128    // Inserting the report content into the element with the ID 'reportContainer'
129    document.getElementById('reportContainer').innerHTML = reportContent;
130  });
131
132  // Adding an event listener to the button with the ID 'calculateCalories'
133  document.getElementById('calculateCalories').addEventListener('click', function() {
134    // Retrieving the value of the input field with the ID 'ingredients'
135    const ingredients = document.getElementById('ingredients').value;
136
137    // Fetching data from the CalorieNinjas API using the ingredients provided by the
user
138
       fetch(`https://api.calorieninjas.com/v1/nutrition?query=${encodeURIComponent(ingredie
nts)}`, {
139        headers: { 'X-Api-Key': 'Wp8RZrtU4h5nv4LxhQrxTg==ourU0D694CVXILNx' } //
CalorieNinjas API key
140    })
141    .then(response => response.json()) // Parsing the response to JSON format
142    .then(data => {
143        // Calculating the total calories by summing up the calories of each ingredient
144        const calories = data.items.reduce((total, item) => total + item.calories, 0);
145
146        // Displaying the total calories in the element with the ID 'calorieResult',
dynamically showing the pizza type
147        document.getElementById('calorieResult').innerText = `Total Calories for
${ingredients}: ${calories.toFixed(2)}`;
148    })
149    .catch(error => {
150        // Logging any errors to the console and displaying an error message to the user
151        console.error('Error:', error);
152        document.getElementById('calorieResult').innerText = 'Failed to calculate
calories.';
153    });
154  });
155
```