



DOUGLAS COLLEGE

COMMERCE AND BUSINESS ADMINISTRATION  
**CSIS 3380: Full Stack Development with JavaScript**

**Assignment 2**

**(5 % towards your Final Grade)**

**Total Marks: 40**

**In the first part of this assignment, you are going to integrate content into web page from an API using Ajax.**

**Introduction:**

Ajax is a technology that allows developers to make asynchronous HTTP requests without the need for a full-page reload.

**Using Fetch**

The Fetch API provides an interface for fetching resources. The `fetch()` method takes one mandatory argument, the path to the resource you want to fetch, and returns a promise that resolves to the response to that request (successful or not). Fetch API is built into the window object, and therefore doesn't need to be installed as a dependency or imported in client-side code. A basic fetch request looks like this. You can also use the

```
fetch(`http://example.com/task.json`)  
  .then(response=>response.json())  
  .then(data => console.log(data))
```

**Using Axios**

Axios is a Promise-based HTTP client for the browser and Node. It is similar to the Fetch API and returns a JavaScript Promise object but also includes many powerful features.

**Using Async/Await**

Another popular approach for handling promises is to create an async function. Instead of waiting for the results of a promise to resolve and handling it with a chain of then functions, async functions can be told to wait for the promise to resolve before further executing any code found in the function. The `await` keyword is used before promise calls.

## Part 1 (This is purely client side; there is no need to create a server for this part.)

Demonstrate the use of **Fetch API** and **Axios** to fetch data from a third-party API and update the part of the HTML page. You are provided with HTML template named “*imageGallery.html*”. Use pixabay API <https://pixabay.com/api/docs/> to bring the desired images and image details on your web page (You will have to generate API key to use the API.).

Create two **scripts** “**fetchapi.js**” (using **fetch API**) and “**axios.js**” (using **Axios**) - you can use **async/await** or **then** function. You must link **both the javascript files** to the *imageGallery.html* file and comment one out. I will uncomment it to see if it is working or not. Both the scripts should accomplish the following task:


When the user enters the search keyword for the image and the number of images to be shown, and hits “**Click to see results**” button, the images and corresponding details for each image like **Downloads**, **LargeImageURL**, **Image Likes**, **Tags**, **Image Type**, **User Name**, **URL** are displayed. Make sure

- the output looks similar to what is shown in Excerpt below. You are free to use your own colors and font styles for displaying the relevant details.
- you clear the previous results before pasting the new ones.
- raise an alert if the search box for keyword is empty or the number of images is less than 2

The following excerpt shows a portion of the resulting webpage after the user enters the search keyword “**Christmas Tree**” and number of images to be 5.

### The Image Gallery

Click to see results



#### Image Details

Downloads: 522386

**LargeImageURL:**  
<https://pixabay.com/get/g7f08e461ab72f464295cfd33a992465bdc191c3e89a32bc41353bc4f7b1e43ab2966e400ecf2fbcecc17e7d950deb9286a5143846098285ce9445e1b7489>


**Image Likes:** 1231

**Tags:** christmas, christmas tree, presents

**Image Type:** photo

**User Name:** Pexels

**URL:**  
<https://pixabay.com/get/g9ddb3dbd94f52b6c633b9b03c2ed72ccf495a9eaa1524445889bd036ce61b005340f63416e19bd0c1b8d222bfcabdfc15f22c4c734989f232bed49e52c3ed5>



#### Image Details

Downloads: 289461

**LargeImageURL:**  
[https://pixabay.com/get/g34d7eeacdb755010017b751188e569036e9fe4e928e04087ed3f932c1c763fd0b4f4de4079ff363d02380f46d03bf216e0ae07426f73ac3ca39f97ef61ccd73\\_](https://pixabay.com/get/g34d7eeacdb755010017b751188e569036e9fe4e928e04087ed3f932c1c763fd0b4f4de4079ff363d02380f46d03bf216e0ae07426f73ac3ca39f97ef61ccd73_)

**Image Likes:** 730

**Tags:** christmas tree, ornaments, decorations

**Image Type:** photo

**User Name:** Pexels

**URL:**  
<https://pixabay.com/get/gdd9efa87f90a4d4aab36778f1e3ee6bf1802ab28ab3778d8120114be38e3a7d1162c659c50a927a51b660f0501e08f8a5512dafd355687b58fba31f3ba26f>

**In the second part, build a server using Node/Express and use Express to render a website with the API data.**

## Part 2:

Complete the following five tasks:

- a. [2 Marks] Handle a GET request from the client at the root route `localhost:3000/`. Send back the following HTML file named “*imageGallery2.html*” (not provided by me; you can create this on your own) to the client as the response:

### The Image Gallery

- b. [2 Marks] After the client populates the required form fields (search keyword and the number of images to be displayed), parse the POST request.

### The Image Gallery

Use the parsed data to structure the URL to fetch the image data from an external server.

- Perform validation: if the search box for keyword is empty or the number of images is less than 2, then send an appropriate status code back to the client. Otherwise, proceed with the next step.
- c. [8 Marks] Next, make a GET request to the external Server (at Pixabay API) with the Fetch and Axios (async/await or then functions). You should comment one out, but both modules should work. I will uncomment the code to see if it works fine for both Fetch and Axios. Collect the dynamic image data from the API based on what user typed into the input fields.
- d. [8 Marks] Create a new template file `imageGallery3.ejs` or `imageGallery3.pug` that renders the output as shown in the previous excerpt. Use `res.render('imageGallery3.ejs', {what data you need to pass})` to render this file from your server. You can use EJS or PUG.

## Submission:

- You are provided with HTML page template that you would use to accomplish the task.
- The assignment is to be completed individually. Any form of cheating or sharing of work may have serious consequences.
- Compress/zip your Assignment\_2 folder as `AS2_LastName_StudentID` and upload it to the Blackboard. NO LATE SUBMISSION will be allowed.

## Rubrics for grading:

**Part 1:** 20 Marks    fetch API 10 + Axios 10

**Part 2:** 20 Marks