Clear Images

This lesson will teach you how to remove all the photo references from Cloud Firestore.

We'll cover the following

Set up Your Clear Button

We start by creating an HTML button. We will add an `id` to it, so we can attach an event listener to it with our JavaScript. Since it's a button, the event we want to listen for is a click. We will get to that in a moment but as you can see from the code below, the HTML is pretty straight forward.

We will be using all the code from the previous lessons as we build in the additional functionality of clearing the photos.

```
<div class="buttons-grid">
    <div>
        <input type="file" id="upload-button">
    </div>
    <div>
        <button id="clear" class="blue-button">Remove all photos</button>
    </div>
</div>
```

HTML

Button styling

We added a class of `blue-button` only so I can give it some decent styling instead of default HTML buttons. Use the CSS below to style it.

instead of default HTML buttons. Use the CSS below to style it.

```css
button{
    color: #fff;
    width: 100%;
    text-align: center;
    border: 0px;
    border-radius: 10px;
    padding: 10px 0px 10px 0px;
    font-size: 16px;
    outline: 0;
    cursor: pointer;
}

.blue-button{
    background-color: #a5dee5 ;
    box-shadow: 0 6px #7eabb1;
}

.blue-button:hover{
    box-shadow: 0 4px #7eabb1;
}

.blue-button:active{
    box-shadow: 0 0 #7eabb1;
}
```

Make an Event Listener for Your New Button

In order to get access to the HTML button, I used the `querySelector`. By implementing `document.querySelector('#clear')`, our JavaScript is aware of the button. Now we can attach the event listener by adding `.addEventListener('click', () => {})`. We will place the code we want to execute on click in between the `{}`.

```javascript
document.querySelector('#clear').addEventListener('click', () => {
        // Code to exexute
    // In our case it will be a Firebase query
})
```

JavaScript

Place the Firestore Query Into the Click Event

To delete, we have to do a few things. At first, the code you see below may look overwhelming but after I walk you through it, you will see it's easy to

look overwhelming but after I walk you through it, you will see it's easy to understand.

The steps we will take are:

1. Request the entire **photos** collection from Firestore. That's the code that looks like this `db.collection("photos").get()` (lines 3-4).
2. Create a loop so we can access the ID of each document inside the **photos** collection (line 8).
3. Use that documents ID to make another request to Firebase (line 9). This is the delete request for that specific document. Each document is a separate message.
4. It's going to return as a promise so we can access it if it was successful with a `.then` or not with a `.catch()` (lines 11-17).

```
document.querySelector('#clear').addEventListener('click', () => {
        // Step 1
        firebase.firestore().collection("images")
        .get()
        // Step 2 (if success)
    .then(function(snapshot) {
                // Step 3
        snapshot.forEach(function(doc) {
                        firebase.firestore().collection("images").doc(doc.id).delete()
                        // Step 4 (if success)
                        .then(function() {
                                console.log("Document successfully deleted!");
                        })
                        // Step 4 (if error)
                        .catch(function(error) {
                                console.error("Error removing document: ", error);
                        });
        });
        })
        // Step 2 (if error)
    .catch(function(error) {
        console.log("Error getting documents: ", error);
    });
})
```

JavaScript

You will notice right away that the images are removed from the output because the docs that reference them have been removed from Firestore. The images are still in your **Storage** which can be viewed from your Firebase console. You would need a little extra functionality to remove them from storage at the same time as you removed the reference from Firestore. For now, you can go into **Storage** from the console and remove them manually as

needed.

The Final Photo Sharing Application!

After pressing the **Run** button, upload 2-3 photos to play around with the functionality we just implemented. Then press the remove button to clear the uploaded photos from the output.

This code requires the following keys to execute:

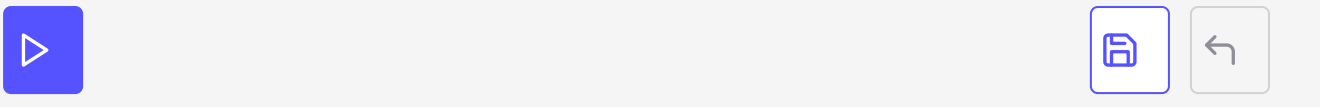
Key:	Value:
apiKey	Not Specified...
authDomain	Not Specified...
databaseURL	Not Specified...
projectId	Not Specified...
storageBucket	Not Specified...
messagingSenderId	Not Specified...
appId	Not Specified...

Output

JavaScript

HTML

CSS (SCSS)



Ready to test your knowledge of Firebase Storage? Join me in the next section for a quick quiz.