Clear Messages

This lesson covers how to delete all messages from the database which in turn erases them from your chat stream.

We'll cover the following

- Set Up your Clear Button
- Make an Event Listener
- Place the Firestore Query Into the Click Event
- Styling Elements
- The Chat Application

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.

```
<!DOCTYPE html>
<html lang="en">
<head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <meta http-equiv="X-UA-Compatible" content="ie=edge">
        <title>Document</title>
        <script src="https://www.gstatic.com/firebasejs/6.3.0/firebase-app.js"></script>
    <script src="https://www.gstatic.com/firebasejs/6.3.0/firebase-firestore.js"></script>
</head>
<body>
        <div id="container">
                <div id="user-options">
                        <div>Chatting as: <span id="name"></span></div>
                        <div id="change-name">change name</div>
                </div>

                <form id="message-form">
                        <input type="text" id="message-input" placeholder="message" required>
                        <button class="orange-button">send</button>
                </form>

                <div id="messages"></div>

                <button id="clear" class="purple-button">Clear All Messages</button>
        </div>
</body>
```

```
</html>
```

HTML

Make an Event Listener

Since it’s a button, the event we want to listen for is *click*.

```
document.querySelector('#clear').addEventListener('click', () => {

})
```

JavaScript

Place the Firestore Query Into the Click Event

To delete, we have to do a few things. The code you see below may look overwhelming but after I walk you through it you will see it’s easy to understand.

The steps are as follows:

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

If you are unfamiliar with JavaScript promises to learn more go here: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise

```
document.querySelector('#clear').addEventListener('click', () => {
    db.collection('messages')
    .get()
    .then(snapshot => {
        snapshot.forEach(doc => {
                        db.collection('messages').doc(doc.id).delete()
            .then(() => {
                                console.log('Document successfully deleted!')
                        })
            .catch(error => {
```

```
                                console.error(`Error removing document: ${error}`)
                        })
        })

    })
    .catch(error => {
        console.log(`Error getting documents: ${error}`)
    })
})
```

JavaScript

Styling Elements

To style the clear button add this CSS.

```
.purple-button{
    background-color: #aa96da ;
        box-shadow: 0 4px #7765a0;
}

.purple-button:hover{
    box-shadow: 0 2px #7765a0;
}

.purple-button:active{
    box-shadow: 0 0 #7765a0;
}
```

The Chat Application

This code requires the following API keys to execute:

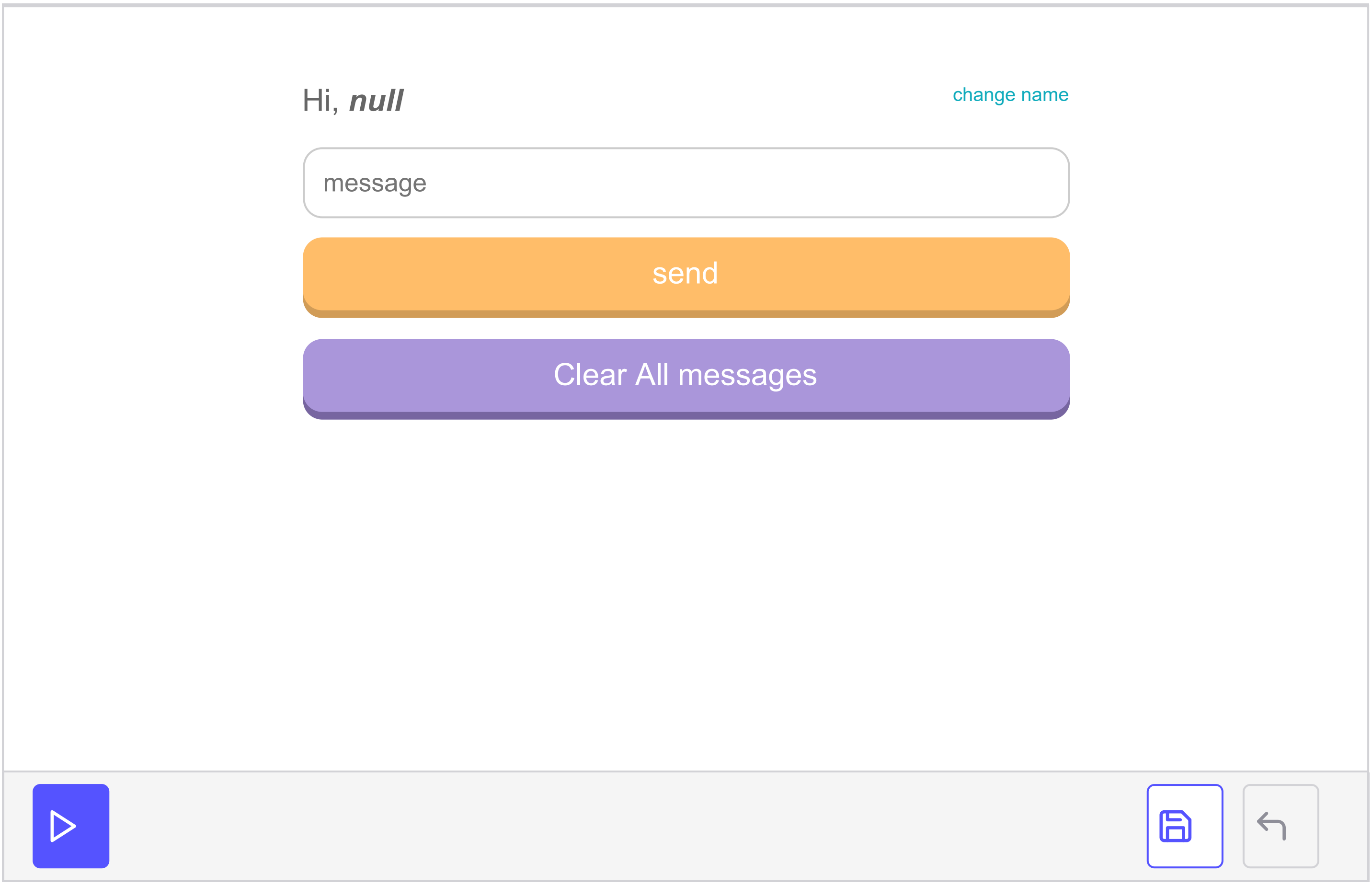
apiKey	Not Specified...
authDomain	Not Specified...
databaseURL	Not Specified...
projectId	Not Specified...
storageBucket	Not Specified...
messagingSenderId	Not Specified...
appId	Not Specified...

Output

JavaScript

HTML

CSS (SCSS)



Now you should be able to add documents and remove entire collections. Feels good, right?

Ready to test your knowledge? Join me in the next section for a quiz on Firebase Firestore.