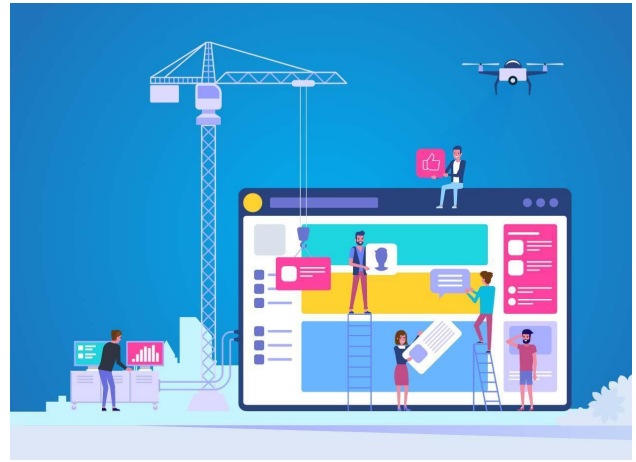


REDES SOCIALES



¿Cuál es nuestro objetivo para este módulo?

Aprendimos a añadir la funcionalidad de Me gusta a un mensaje.

¿Qué logramos en la clase hoy?

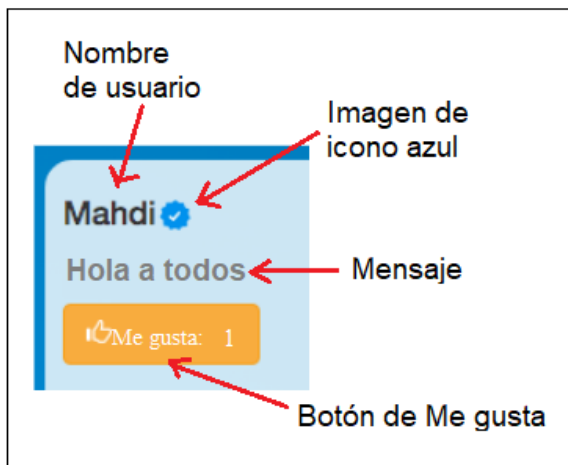
- Escribimos el código JS para actualizar y mostrar el número de Me gusta de un mensaje.
- Completamos la aplicación web de Kwitter.

¿Qué conceptos y bloques de código vimos hoy?

- Terminamos de programar la función **getData()**.
- Escribimos la función **updateLike()**.

¿Cómo hicimos las actividades?

Agregamos los siguientes elementos a la página de mensajes de Kwitter:



Lo siguiente se añade en cada mensaje.

El alumno debe seguir trabajando en el archivo **kwitter_page.js** que estuvo utilizando en la clase pasada.

Como puedes ver, el archivo **kwitter_page.js** ya contiene la función **getData()**.

- Código de **getData()**.

```
function getData() { firebase.database().ref("/"+room_name).on('value', function(snapshot) { document.getElementById("output")
    firebase_message_id = childKey;
    message_data = childData;
    //Inicia código
    //Termina código
  } }); }); }
getData();
```

- Este código se utiliza para obtener todos los datos de Firebase, no tienes que entender el código por ahora, ya que sólo lo usamos para encontrar los documentos fácilmente.
- Este código sólo es para obtener el mensaje, el nombre de usuario y los Me gusta. Escribimos código para mostrar el mensaje, el nombre de usuario y los Me gusta en un elemento HTML que habíamos definido en la clase anterior, dentro de **kwitter_page.html**.

Tú debes programar entre las dos líneas de comentario **Inicia código** y **Termina código**.

Código completo de **getData()**:

```
function getData() { firebase.database().ref("/"+room_name).on('value', function(snapshot) { document.getElementById("output").inner
    firebase_message_id = childKey;
    message_data = childData;
    //Inicia código
    console.log(firebase_message_id);
    console.log(message_data);
    name = message_data['name'];
    message = message_data['message'];
    like = message_data['like'];
    name_with_tag = "<h4> " + name + "<img class='user_tick' src='tick.png'></h4>";
    message_with_tag = "<h4 class='message_h4'>" + message + "</h4>";
    like_button = "<button class='btn btn-warning' id='"+firebase_message_id+" value='+like+' onclick='updateLike(this.id)'>";
    span_with_tag = "<span class='glyphicon glyphicon-thumbs-up'>Like: " + like + "</span></button><hr>";

    row = name_with_tag + message_with_tag + like_button + span_with_tag;
    document.getElementById("output").innerHTML += row;
    //Termina código
    } }); }); }
    getData();
```

El alumno debe programar esto.

1. Utiliza la variable **firebase_message_id** para contener todos los id únicos de los mensajes generados por Firebase.
2. Utiliza la variable **message_data** para guardar todos los mensajes, los Me gusta y el nombre de usuario de cada mensaje.
 - Utiliza estas dos variables para mostrar el nombre de usuario, los Me gusta y los mensajes en el elemento HTML.

```
function getData() { firebase.database().ref("/"+room_name).on('value', function(snapshot) { document.getElementById("output").inner
    firebase_message_id = childKey;
    message_data = childData;
```

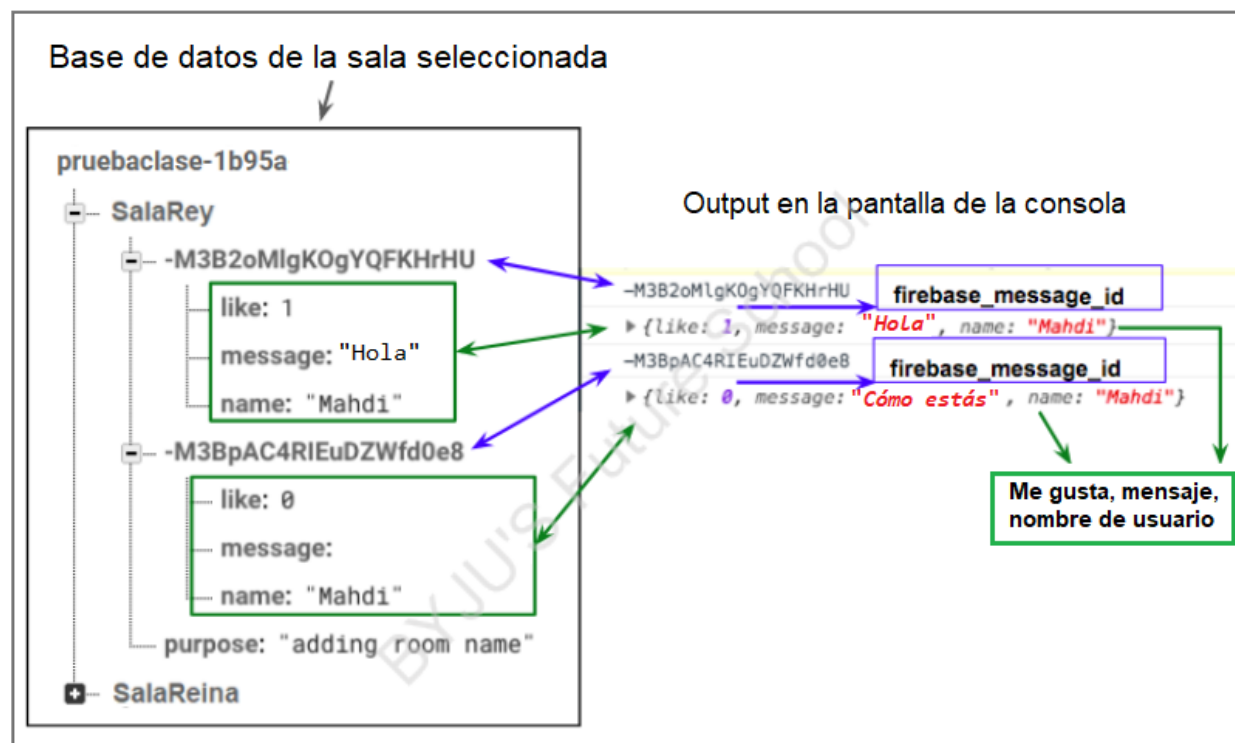
3. Ahora vamos a mostrar en la consola estas dos variables.

```
function getData() { firebase.database().ref("/"+room_name).on('value', function(snapshot) { document.getElementById("output").inner
    firebase_message_id = childKey;
    message_data = childData;
    //Inicia código
    console.log(firebase_message_id);
    console.log(message_data);
```

- Output en la pantalla de la consola:

-M3B2oMlgK0gYQFKHrHU	kwiter_page.js:33
▶ {like: 1, message: "HoLa" , name: "Madhi"}	kwiter_page.js:34
-M3BpAC4RIEuDZWfd0e8	kwiter_page.js:33
▶ {like: 0, message: "Cómo estás" , name: "Madhi"}	kwiter_page.js:34

Explicación de las variables `firebase_message_id` y `message_data`:



Los elementos marcados son las **Claves**

-M3B2oMlgK0gYQFKHrHU	kwiter_page.js:33
▶ {like: 1, message: "HoLa" , name: "Mahdi"}	kwiter_page.js:34
-M3BpAC4RIEuDZWfd0e8	kwiter_page.js:33
▶ {like: 0, message: "Cómo estás" , name: "Mahdi"}	kwiter_page.js:34

Los elementos marcados son los **Valores**

```
-M3B2oMlgK0gYQFKHrHU                                twitter_page.js:33
▶ {like: 1, message: "Hola", name: "Mahdi"}              twitter_page.js:34
-M3BpAC4RIEuDZWfd0e8                                twitter_page.js:33
▶ {like: 0, message: "Cómo estás", name: "Mahdi"}        twitter_page.js:34
```

Ahora, obtén uno por uno estos valores de la variable **message_data** utilizando las claves y almacénalos en variables.

- Primero obtén el nombre de la variable **message_data** usando la clave **name**. Y guárdalo dentro de la variable **name**.

```
console.log(firebase_message_id);
console.log(message_data);
name = message_data['name'];
```

- name** es nuestra variable.
- message_data['name']** - **message_data** contiene todos los valores de los datos del mensaje (que son los Me gusta, el nombre y el mensaje) y las claves correspondientes.
 - name** es la clave para poder tomar el nombre.

- Obtén el mensaje de la variable **message_data** usando la clave **message**. Y guárdalo dentro de la variable **message**.

```
console.log(firebase_message_id);
console.log(message_data);
name = message_data['name'];
message = message_data['message'];
```

- message** es nuestra variable.
- message_data['message']** - **message_data** contiene todos los valores del

mensaje (que son los Me gusta, el nombre y el mensaje) y las claves correspondientes.

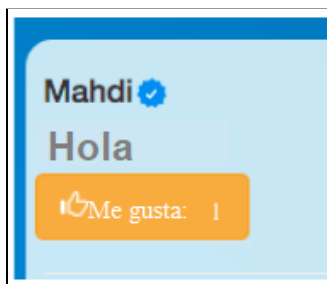
- **message** es la clave para poder tomar el mensaje.

6. Obtén el número de la variable **message_data** usando la clave **like**. Y guárdalo dentro de la variable **like**.

```
console.log(firebase_message_id);
console.log(message_data);
name = message_data['name'];
message = message_data['message'];
like = message_data['like'];
```

- **like** es nuestra variable
- **message_data['like']** - **message_data** contiene todos los valores del mensaje (que son los Me gusta, el nombre y el mensaje) y las claves correspondientes.
- **like** es la clave para poder tomar el número de Me gusta del mensaje.

Ahora creamos las etiquetas HTML para mostrar el nombre de usuario, la imagen del icono, el mensaje y el botón de Me gusta, así:



1. Creamos una etiqueta HTML para mostrar el nombre de usuario y la imagen del icono, y la almacenaremos dentro de una variable.

```
name = message_data['name'];
message = message_data['message'];
like = message_data['like'];
name_with_tag = "<h4> " + name + "<img class='user_tick' src='tick.png'></h4>";
```

Explicación del código anterior:

- **name_with_tag** es nuestra variable, aquí vamos a almacenar el nombre de

usuario y la imagen del icono.

```
name_with_tag =
```

- Definimos la etiqueta **h4** entre comillas dobles.

```
"<h4> "
```

- Luego ponemos un signo "+" para unir el nombre con la etiqueta **h4**.

```
"<h4> "+
```

- Luego tenemos la variable **name**, que contiene el nombre de usuario.

```
"<h4> "+ name
```

- Luego tenemos un signo "+" para unir la etiqueta **img**.

```
"<h4> "+ name +"
```

- Definimos la etiqueta **img** entre comillas dobles.

```
"<h4> "+ name +"<img
```

- Definimos **class** para la etiqueta **img** entre comillas simples.

```
"<h4> "+ name +"<img class='user_tick'
```

- Utilizamos **class = 'user_tick'** solo porque añadimos algunas propiedades CSS a esta clase en **style.css**.
- Ponemos **src** a esta etiqueta **img** entre comillas simples.

```
"<h4> "+ name +"<img class='user_tick' src='tick.png'>
```

- Cerraremos la etiqueta **h4** después de la etiqueta **img**.

```
name_with_tag = "<h4> " + name + "<img class='user_tick' src='tick.png'></h4>";
```

- El código anterior dará como resultado lo siguiente:



2. Creamos una etiqueta HTML para mostrar los mensajes y la guardamos dentro de una variable.

```
name = message_data['name'];
message = message_data['message'];
like = message_data['like'];
name_with_tag = "<h4> " + name + "<img class='user_tick' src='tick.png'></h4>";
message_with_tag = "<h4 class='message_h4'>" + message + "</h4>";
```

Explicación del código anterior:

- **message_with_tag** es nuestra variable, aquí vamos a almacenar el mensaje.

```
message_with_tag =
```

- Añadimos la etiqueta **h4** entre comillas dobles.

```
"<h4
```

- Definimos **class** para la **etiqueta h4** entre comillas simples.

```
"<h4 class='message_h4'>"
```

- Utilizamos **class = 'message_h4'** solo porque añadimos algunas propiedades CSS a esta clase en **style.css**.

- Luego tenemos un signo **"+"** para unir la etiqueta **message**.

```
"<h4 class='message_h4'>" +
```


- Luego tenemos la variable **message**, que contiene el mensaje.

```
"<h4 class='message_h4'>" + message
```

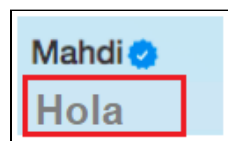
- Luego ponemos otro signo "+" para unir la etiqueta **h4** de cierre.

```
"<h4 class='message_h4'>" + message +
```

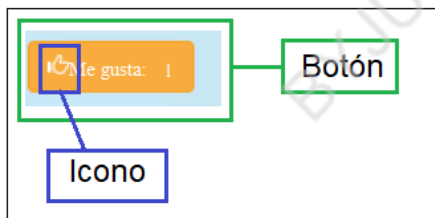
- Cerramos la etiqueta h4 dentro de comillas dobles.

```
"<h4 class='message_h4'>" + message + "</h4>";
```

- El código anterior dará como resultado lo siguiente:



Escribimos una etiqueta HTML para mostrar el botón de Me gusta. Hacemos esto en dos partes, primero añadimos botón y luego el icono de Me gusta al lado.



3. Añadimos la etiqueta de botón y la almacenamos dentro de la variable.

```
like = message_data['like'];
name_with_tag = "<h4> " + name + "<img class='user_tick' src='tick.png'></h4>";
message_with_tag = "<h4 class='message_h4'>" + message + "</h4>";
like_button = "<button class='btn btn-warning' id='"+firebase_message_id+"' value="+like+" onclick='updateLike(this.id)'">";
```

Explicación del código anterior:

- **like_button** es nuestra variable, aquí vamos a almacenar la etiqueta **button**.

```
like_button =
```

- Primero añadimos una etiqueta **button** entre comillas dobles.

```
"<button "
```

- Luego ponemos **class** a esta etiqueta **button** entre comillas simples.

```
"<button class='btn btn-warning'"
```

- Utiliza **class = 'btn btn-warning'** estas son clases de bootstrap que ya conoces.
- Ahora le ponemos el **id = "+firebase_message_id"** al botón. Hacemos esto porque cada botón debe tener una identificación única.
 - **firebase_message_id** tendrá el id del mensaje específico.

```
"<button class='btn btn-warning' id="+firebase_message_id+""
```

- Cerramos las comillas dobles y añadimos el signo de más.
 - Escribimos el nombre de la variable y volvemos a añadir el signo de más.
 - Luego cerramos las comillas dobles.
 - Esto se hace porque las variables **no** se pueden escribir dentro de comillas dobles.

Asegúrate de que no hay ningún error en esto, de lo contrario, no funcionará.

Por ejemplo:

- Cuando el código se ejecute y si el valor de la variable **firebase_message_id** es **-M3B2oMlgKOgYQFKHrHU**.
 - Entonces `<button class=btn btn-warning id = "+firebase_message_id+"` el resultado será:
 - `<div class='room_name' id = -M3B2oMlgKOgYQFKHrHU`
- Cuando el código se ejecute y si el valor de la variable **firebase_message_id** es **-M3BpAC4RIEuDZWfd0e8**.
 - Entonces `<button class=btn btn-warning id = "+firebase_message_id+"` el

resultado será:

- `<div class='room_name' id = -M3BpAC4RIEuDZWfd0e8`

- Ahora le damos valor al botón.

```
"<button class='btn btn-warning' id="+firebase_message_id+" value="+like+">"
```

- Cerramos las comillas dobles y añadimos el signo de más.
- Escribimos el nombre de la variable y volvemos a añadir el signo de más.
- Luego cerramos las comillas dobles.
 - Esto se hace porque las variables **no** se pueden escribir dentro de comillas dobles.

Asegúrate de que no hay ningún error en esto, de lo contrario, no funcionará.

Por ejemplo:

- Cuando el código se ejecute y si el valor de la variable **like** es **1**.
 - Entonces `<button class=btn btn-warning id = "+firebase_message_id+" valor ="+ like +"` el resultado será:
 - `<div class='room_name' id = -M3B2oMlgKOGYQFKHrHU valor ="1"`
- Y cuando el código se ejecute y si el valor de la variable **like** es **2**.
 - Entonces `<button class=btn btn-warning id = "+firebase_message_id+" valor ="+ 2 +"` el resultado será:
 - `<div class='room_name' id = -M3BpAC4RIEuDZWfd0e8 valor ="1"`

Le damos el número de Me gusta del mensaje como valor al botón.

EXPLICACIÓN: hacemos esto porque cuando se hace clic en el botón de Me gusta, tomaremos el valor de ese botón, lo aumentaremos en 1 y actualizaremos los Me gusta de ese mensaje en la base de datos.

- Ahora añadimos la función onclick al botón.

```
"<button class='btn btn-warning' id="+firebase_message_id+" value="+like+" onclick='updateLike(this.id)'>"
```

- **onclick='updateLike(this.id)'**
 - **onclick** ya lo conocemos.
 - La función **updateLike** la definimos más adelante.
 - **this.id** significa que siempre que se llame a la función **updateLike**, se pasará el valor del id actual del elemento dentro de la función **updateLike()**.

Por ejemplo:

El id correspondiente se estableció en el paso anterior.

- Así que si el id=-M3B2oMlgKOGYQFKHrHU cuando el código se ejecute.
 - Entonces **updateLike(this.id)** será:
 - **updateLike(-M3B2oMlgKOGYQFKHrHU)**
- Y si el id=-M3BpAC4RIEuDZWfd0e8 cuando el código se ejecute.
 - Entonces **updateLike(this.id)** será:
 - **updateLike(-M3BpAC4RIEuDZWfd0e8)**

Pasamos indirectamente el id de Firebase dentro de la función **updateLike()**.

EXPLICACIÓN: hacemos porque cuando se hace clic en el botón Me gusta queremos que aumente el número de Me gusta del mensaje. Así que para identificar de qué mensaje queremos aumentar los Me gusta, tenemos que pasar el **firebase_message_id** dentro de la función **updateLike()**.

4. Para añadir el icono dentro del botón de Me gusta, vamos a escribir:

```
like = message_data['like'];
name_with_tag = "<h4> " + name + "<img class='user_tick' src='tick.png'></h4>";
message_with_tag = "<h4 class='message_h4'>" + message + "</h4>";
like_button = "<button class='btn btn-warning' id='"+firebase_message_id+"' value="+like+" onclick='updateLike(this.id)'">";
span_with_tag = "<span class='glyphicon glyphicon-thumbs-up'>Like: "+ like + "</span></button><hr>";
```

AMPLIACIÓN DEL CÓDIGO:

```
span_with_tag = "<span class='glyphicon glyphicon-thumbs-up'>Like: "+ like + "</span></button><hr>";
```

Explicación del código anterior:

- **span_with_tag** es nuestra variable, aquí almacenamos la etiqueta del botón.

```
span_with_tag =
```

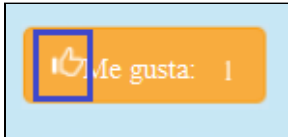
- Definimos la **etiqueta span** entre comillas dobles.

```
"<span
```

- Luego ponemos **class** a esta etiqueta **span** entre comillas simples.

```
"<span class='glyphicon glyphicon-thumbs-up'>|
```

- Utiliza class = 'glyphicon glyphicon-thumbs-up' estas son clases de bootstrap para añadir el icono:



- Añadimos el texto de Me gusta dentro de esta etiqueta span, así:



```
span_with_tag = "<span class='glyphicon glyphicon-thumbs-up'>Like: "
```

- Luego tenemos la variable **like**. El resultado es este:



```
span_with_tag = "<span class='glyphicon glyphicon-thumbs-up'>Like: "+ like +"
```

- Ponemos comillas dobles y el signo de más.
- Escribimos el nombre de la variable y volvemos a añadir el signo de más.
- Luego cerramos las comillas dobles.
 - Esto se hace porque las variables **no** se pueden escribir dentro de comillas dobles.

Asegúrate de que no hay ningún error en esto, de lo contrario, no funcionará.

EXPLICACIÓN: hacemos esto para mostrar el número de Me gusta de ese mensaje en específico.

5. Ponemos todas esas variables dentro de una sola.

```
span_with_tag = "<span class='glyphicon glyphicon-thumbs-up'>Like: "+ like + "</span></button><hr>";  
row = name_with_tag + message_with_tag + like_button + span_with_tag;
```

6. Actualizamos el elemento HTML que tiene el **id= 'output'**, con esta variable **row**, que contiene el elemento HTML con el **nombre de usuario, la imagen del icono, el mensaje, el botón y el icono de Me gusta.**

```
row = name with tag + message with tag + like button + span with tag;  
document.getElementById("output").innerHTML += row;
```

- Primero obtenemos el elemento HTML que tiene **id= 'output'**, ya definimos este elemento HTML en **twitter_room.html**.
- Usamos **innerHTML**, esto ayudará a actualizar el elemento HTML seleccionado.
- Si sólo escribimos = que normalmente escribimos para actualizar los elementos HTML, entonces sólo se mostrará un nombre de sala. Pero puede haber muchos nombres de salas, y queremos mostrarlos todos dentro de un solo elemento HTML, por eso usamos += y luego escribimos la variable **row**.

NOTA DE DEPURACIÓN: revisa si los paréntesis() y las llaves{} están cerrados correctamente para la función **addRoom()**. Si se produce algún error, abre la consola para encontrarlo y corregirlo.

NOTA DE DEPURACIÓN: REVISAMOS QUE EL CÓDIGO TENGA LAS COMILLAS DOBLES " " Y SIMPLES' CERRADAS CORRECTAMENTE.

Ahora escribimos la función **updateLike()**.

Código completo:

```
function updateLike(message_id)
{
  console.log("Botón Me gusta pulsado - " + message_id);
  button_id = message_id;
  likes = document.getElementById(button_id).value;
  updated_likes = Number(likes) + 1;
  console.log(updated_likes);

  firebase.database().ref(room_name).child(message_id).update({
    like : updated_likes
  });
}
```

Explicación del código:

7. Definimos la función y le pasamos el mensaje según la explicación anterior.

```
function updateLike(message_id)
{
```

- Este **message_id** es el mismo **firebase_message_id** que habíamos pasado al pulsar el botón **like**.

Propósito de **message_id**:

Propósito 1:

- Ya que pusimos **message_id** como la identificación única para el botón like, Y colocamos el número de Me gusta como valor al botón like.
 - Entonces, cuando se haga clic en el botón, utilizamos el **message_id** para identificar al botón y tomar su valor, que es el número de Me gusta y lo incrementamos.

Propósito 2:

- Este **message_id** es la identificación única para cada mensaje en la base de datos.
 - Usamos **message_id** para identificar el mensaje en la base de datos y actualizamos los nuevos **valores incrementados de Me gusta**.
8. Mostramos en la consola el **message_id** para verificar si se muestra o no.

```
function updateLike(message_id)
{
  console.log("Botón Me gusta pulsado - " + message_id);
```

- Haz clic en el botón de Me gusta.



- Ve la pantalla de la consola.

-M3B2oMlgK0gYQFKHrHU	twitter_page.js:33
▶ {like: 0, message: "Hola", name: }	twitter_page.js:34
-M3BpAC4RIEuDZWfd0e8	twitter_page.js:33
▶ {like: 0, message: "Cómo estás", name: }	twitter_page.js:34
Botón Me gusta pulsado - -M3B2oMlgK0gYQFKHrHU	twitter_page.js:51

- Notarás que el **message_id** aparece correctamente. Esto significa que podemos usar **message_id** para actualizar los Me gusta de los mensajes.

9. Ahora asigna el valor de **message_id** a una nueva variable **button_id**.

```
console.log("Botón Me gusta pulsado - " + message_id);
button_id = message_id;
```

10. Según el **propósito 1** que definimos, tomamos el valor del botón **like** y lo incrementamos en 1.

```
likes = document.getElementById(button_id).value;
```

- **likes** es la variable. Ahí almacenamos el valor de los Me gusta que obtenemos

del botón.

- `document.getElementById(button_id).value`, aquí pasamos el **button_id** que contiene el **message_id**, que es la identificación única del botón **like**.
- Y tomamos el **valor** de este botón escribiendo **.value**. El valor del botón es el número de Me gusta del mensaje.

11. Ahora incrementamos este número de Me gusta en 1.

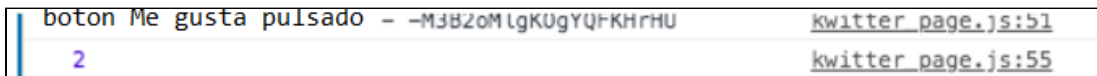
```
updated_likes = Number(likes) + 1;
```

- **updated_likes** es la variable en la que almacenaremos el valor incrementado de Me gusta.
- **Number(likes) + 1**: **Number()** es una función de JS para convertir cadenas en números.
 - Así que convertimos el valor de los Me gusta de cadena a numérico, hacemos esto porque si "likes" = 2 y se le agrega 1:
 - Se convertiría en 21 y no en 3, por eso convertimos el número de likes a numérico, para obtener el valor correcto que es 3.
 - Dentro de **Number(likes)** pasamos la variable **likes** porque queremos convertir su valor en numérico.
 - **+1** - significa que añadirá 1 a la variable **likes**. Entonces, si la variable **likes** es 2, el **+1** hará que se convierta en 3.

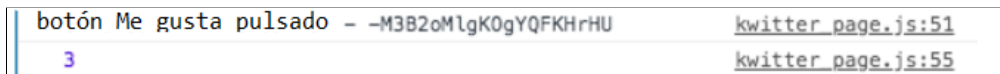
12. Ahora vamos a la consola a ver **updated_likes**.

```
console.log(updated_likes);
```

- Pulsa el botón de Me gusta y ve la pantalla de la consola:



- Presiona de nuevo el botón de Me gusta y ve la pantalla de la consola:



- Al pulsar el botón de Me gusta, el número de output en la pantalla de la consola aumenta en 1.

13. Actualizamos los Me gusta del mensaje usando **message_id** (que es la identificación única de los mensajes en la base de datos) con el valor de like incrementado.

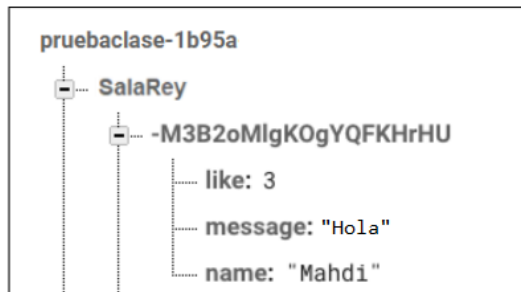
```
firebase.database().ref(room_name).child(message_id).update({  
  like : updated_likes  
});
```

- Firebase se utiliza para establecer la referencia para añadir datos a la base.
- **database()** es para añadir datos a la base de datos.
- **ref(room_name)**:
 - **ref()** es la referencia, es decir, la sala donde está el mensaje concreto para el que queremos actualizar los Me gusta.
 - **"room_name"** tiene el nombre de la sala en la que está el mensaje. Por eso pasamos la variable **room_name** dentro de **ref()**, para poder localizar el mensaje.
- **child(message_id)**:
 - La función **child()** se utiliza para buscar el mensaje concreto en la sala seleccionada.
 - **message_id** tiene la identificación del mensaje.
- **update** es la función de firebase que se utiliza para actualizar la base de datos con los valores.
- Para actualizar los Me gusta dentro del mensaje utilizamos la clave like. Y el valor que queremos actualizar es el valor incrementado de like, por eso escribimos la variable **update_likes** (que tiene el valor incrementado de los Me gusta).

```
like : updated_likes
```

Ahora presiona el botón de Me gusta de cualquier mensaje y abre la base de datos, verás que el like de ese mensaje se actualiza. También se mostrará el like actualizado dentro del botón de Me gusta del mensaje:





14. Finalmente, añadiremos la función **logout()**, la misma que la función **logout()** de la página de la sala de Kwitter. Abrimos el archivo **kwitter_room.js** copiamos la función **logout()** y la pegamos en **kwitter_page.js**.

```
function logout() {
  localStorage.removeItem("user_name");
  localStorage.removeItem("room_name");
  window.location.replace("kwitter.html");
}
```

¿Qué sigue?

Empezaremos a construir una aplicación web de selfie controlada por voz. Aprenderemos a convertir la voz en texto.

AMPLÍA TUS CONOCIMIENTOS

Aquí tienes algunas de las mejores referencias que hemos recopilado para ampliar tus conocimientos y comprensión de los conceptos que aprendimos hoy. Esto te ayudará a convertirte en un profesional de la programación y la creación de productos tecnológicos de calidad industrial.


Videos cortos: Mira estos videos para entender la utilización de los conceptos aprendidos en clase en aplicaciones del mundo real.


1. Tutorial de la aplicación web Firebase:
<https://www.youtube.com/watch?v=nUUPedePJ4o>
2. Cómo usar la función `window.location.replace()` en JS:
<https://www.youtube.com/watch?v=d2UUD-jSnFA>

3. Bucle ForEach en JS: <https://www.youtube.com/watch?v=upP06o5ZPIs>

Patio de programación: Prueba estos ejemplos de código para practicar la creación de páginas web y aplicaciones listas para la Play Store.

1.  Retrieving Data | Firebase ...
firebase.google.com
:
<https://firebase.google.com/docs/database/admin/retrieve-data>

2.  Location replace() Method
w3schools.com
:
https://www.w3schools.com/jsref/met_loc_replace.asp

3.  JavaScript Array forEach(...)
w3schools.com
:
https://www.w3schools.com/jsref/jsref_foreach.asp