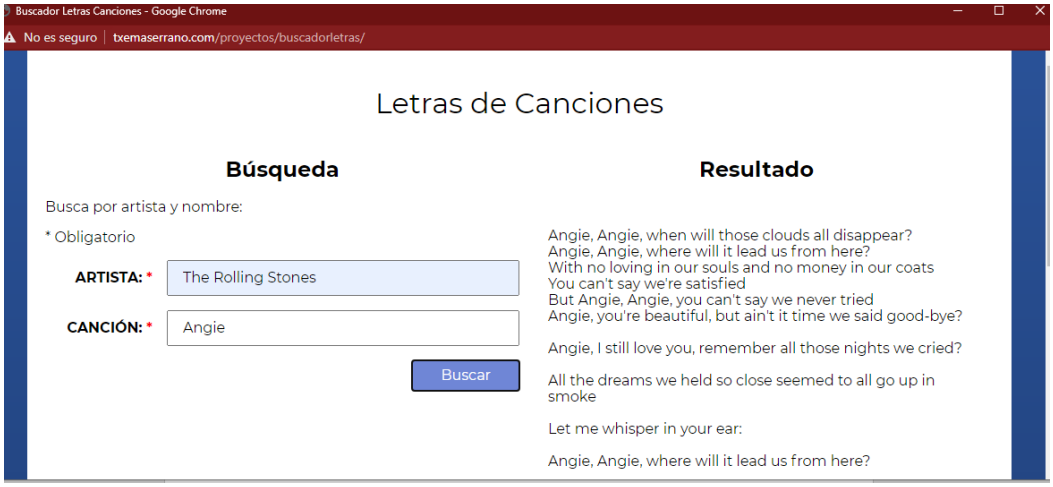
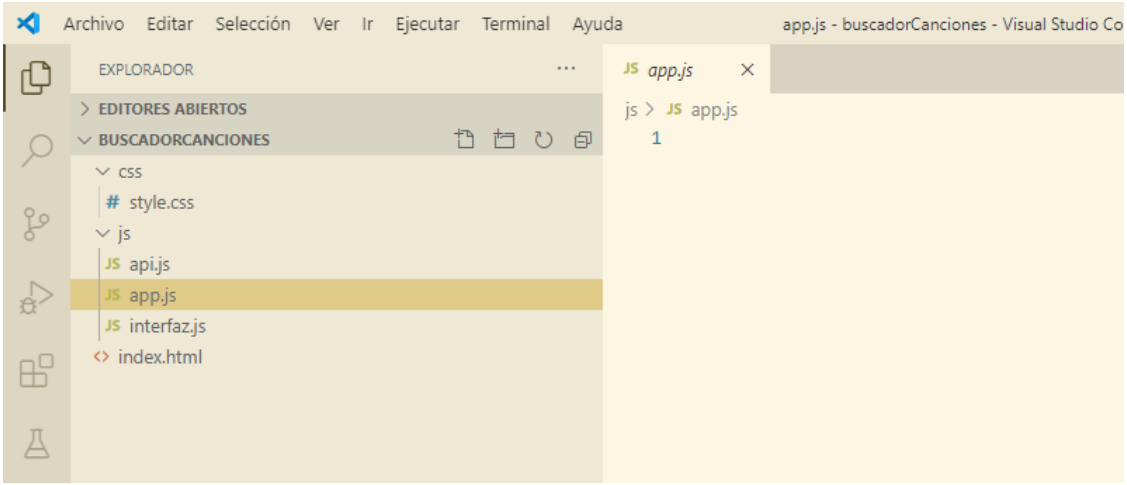
Buscador de canciones con Fetch-API

Si rellenamos el formulario:



Paso 1. Descargar plantilla:



Como vamos a utilizar módulos declaramos el script de tipo “module” en index.html:

```
    <script src="js/app.js" type="module"></script>
</body>
</html>
```

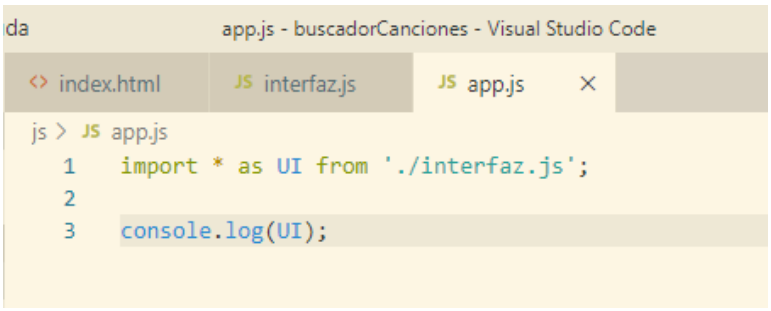
Paso 2. Crear la interfaz.

El primer fichero que trabajaremos será interfaz.js, para colocar una serie de selectores:

index.html | interfaz.js

```
export const formularioBuscar = document.querySelector("#formulario-buscar"),
             // document.getElementById("formulario-buscar");
    divBuscar = document.querySelector("#buscar"),
    divMensajes = document.querySelector("#mensajes"),
    divResultado = document.querySelector("#resultado"),
    headingResultado = document.querySelector(".letra-resultado h2");
```

En **app.js** importaremos esta interfaz:



Salida:

```
Live reload enabled.                                    index.html:77
▼Module                                                 app.js:3
    divBuscar: (...)
    divMensajes: (...)
    divResultado: (...)
    formularioBuscar: (...)
    headingResultado: (...)
    Symbol(Symbol.toStringTag): "Module"
```

Asociamos una función al formulario para cuando se pulse el botón y extraemos los parámetros de la búsqueda: artista y canción:

index.html | interfaz.js | api.js | app.js

js > app.js > buscarCancion

```
1   import * as UI from './interfaz.js';
2
3   console.log(UI);
4
5   // Utilizamos el formulario buscar:
6
7   UI.formularioBuscar.addEventListener('submit', buscarCancion);
8
9   function buscarCancion(e) {
10      e.preventDefault();
11
12      const artista = document.querySelector("#artista").value;
13      const cancion = document.querySelector("#cancion").value;
14
15      // Validamos los datos
16
17      if (artista == "" || cancion == "") {
18          // El usuario dejó un campo vacío, mostrar mensaje de error
19
20          UI.divMensajes.textContent = "Error: todos los campos son obligatorios";
21          UI.divMensajes.classList.add("error");
22      }
23  }
```

Comprobamos que no estén vacíos:

Para hacer que el mensaje se visualice durante 3 segundos únicamente:

```
    UI.divMensajes.textContent = "Error: todos los campos son obligatorios";
    UI.divMensajes.classList.add("error");

    setTimeout(() => {
        UI.divMensajes.innerHTML = '';
        UI.divMensajes.classList.remove('error');
    }, 3000);
}
```

Paso 3. Conectar con el servicio REST.

Si no hay error, consultamos la API que nos descarga las canciones desde el servicio REST lyrics.com

index.html | interfaz.js | api.js | app.js

js > app.js > buscarCancion

```
6
7   UI.formularioBuscar.addEventListener('submit', buscarCancion);
8
9   function buscarCancion(e) {
10      e.preventDefault();
11
12      const artista = document.querySelector("#artista").value;
13      const cancion = document.querySelector("#cancion").value;
14
15      // Validamos los datos
16
17      if (artista == "" || cancion == "") {
18          // El usuario dejó un campo vacío, mostrar mensaje de error
19
20          UI.divMensajes.textContent = "Error: todos los campos son obligatorios";
21          UI.divMensajes.classList.add("error");
22
23          setTimeout(() => {
24              UI.divMensajes.innerHTML = '';
25              UI.divMensajes.classList.remove('error');
26          }, 3000);
27      } else {
28
29          // Consultamos la API
30          console.log('Consultando la API');
31      }
32  }
```

El esquema de nuestra API es:

index.html | interfaz.js | api.js | app.js

js > api.js > API

```
1   // Importamos la interfaz
2
3   import * as UI from './interfaz.js';
4
5   class API {
6
7   }
8
9   export default API;
```

Creamos el constructor de la clase:

```
4
5   class API {
6
7       constructor(artista, cancion){
8           this.artista = artista;
9           this.cancion = cancion;
10      }
11
12  }
13
14  export default API;
```

Esta clase hay que importarla en app.js:

index.html | interfaz.js | api.js | app.js

```
1   import * as UI from './interfaz.js';
2   import API from './api.js';
3
4   console.log(UI);
5
6   // Utilizamos el formulario buscar:
7
8   UI.formularioBuscar.addEventListener('submit', buscarCancion);
```

Y para realizar la búsqueda con los elementos del formulario, creamos una función **consultarAPI()** en la API:

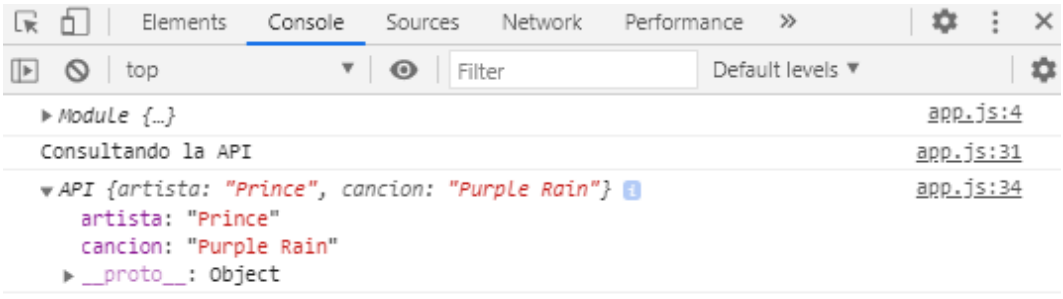
index.html | interfaz.js | api.js | app.js

```
1   // Importamos la interfaz
2
3   import * as UI from './interfaz.js';
4
5   class API {
6
7       constructor(artista, cancion) {
8           this.artista = artista;
9           this.cancion = cancion;
10      }
11
12      consultarAPI() {
13          console.log('Consultando la API...');
14      }
15
16  }
17
18  export default API;
```

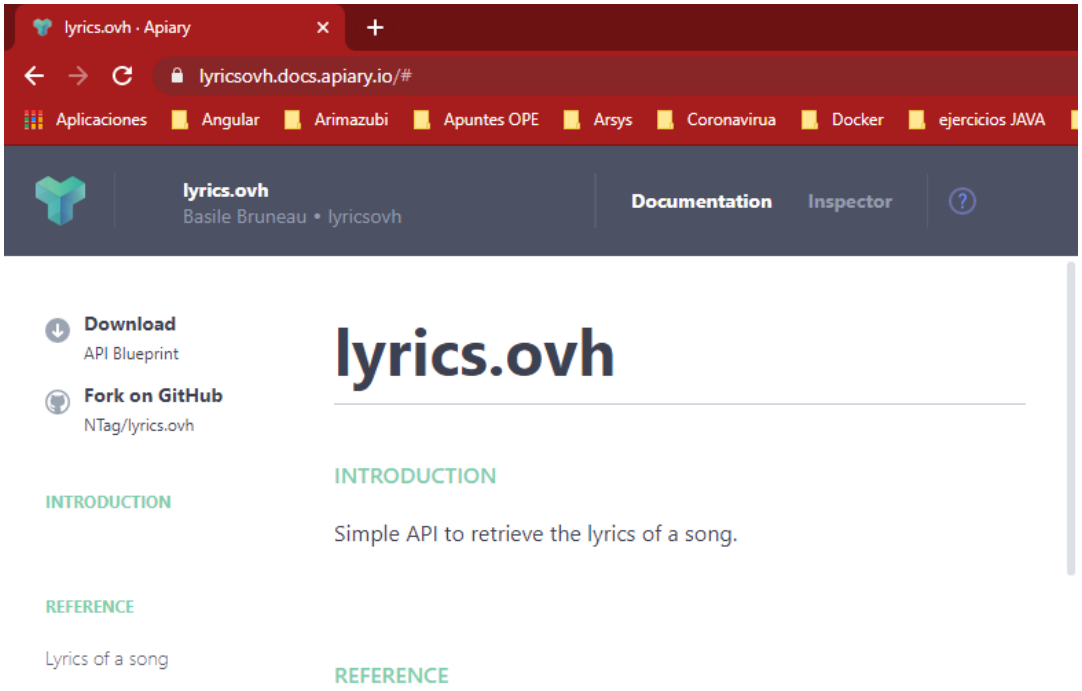
La invocamos con nuestro objeto de búsqueda desde app.js:

```
27          }, 3000);
28      } else {
29
30          // Consultamos la API
31          console.log('Consultando la API');
32          const busqueda = new API(artista, cancion);
33
34          busqueda.consultarAPI();
35          console.log(busqueda);
36      }
37  }
```

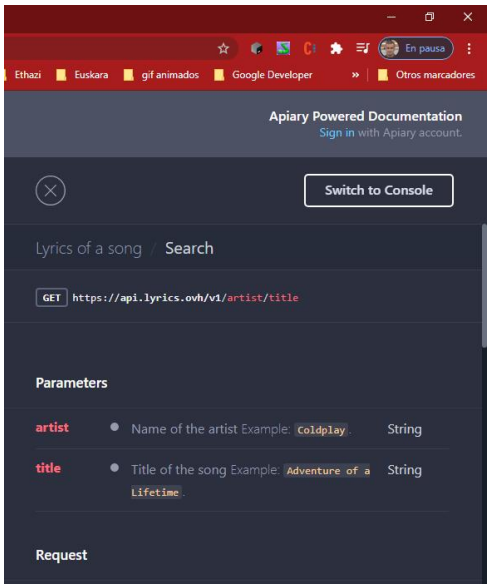
La salida que se obtiene:

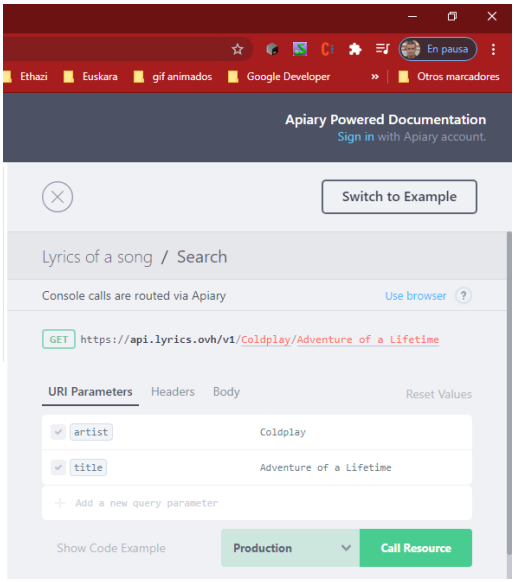


Para retornar las letras de las canciones, utilizamos el siguiente servicio REST:



Cuya forma de invocar:





https://api.lyrics.ovh/v1/artist/title

Parámetros:

- Artista (string): nombre del artista, ejemplo: Coldplay.
- Title (string): nombre de la canción, ejemplo: Adventure of a Lifetime.

Si lo invocamos:

Request

GEThttps://private-anon-1e370afdc9-lyricsovh.apiary-proxy.com/v1/Coldplay/Adventure%20of%20a%20Lifetime

Response

200

Response Headers

```
content-type: application/json; charset=utf-8
content-length: 1361
x-powered-by: express
access-control-allow-origin: *
etag: w/"551-eftq7nnlbwxj35m+zvvbbloqgxu"
….
```

Response Body

```
{
  "lyrics": "Turn your magic on, Umi she'd say
Everything you want's a dream away
We are legends, every day
That's what she told me
Turn your magic on, to me she'd say
……..
```

Paso 4. Reconfiguramos nuestra API con los parámetros del formulario.

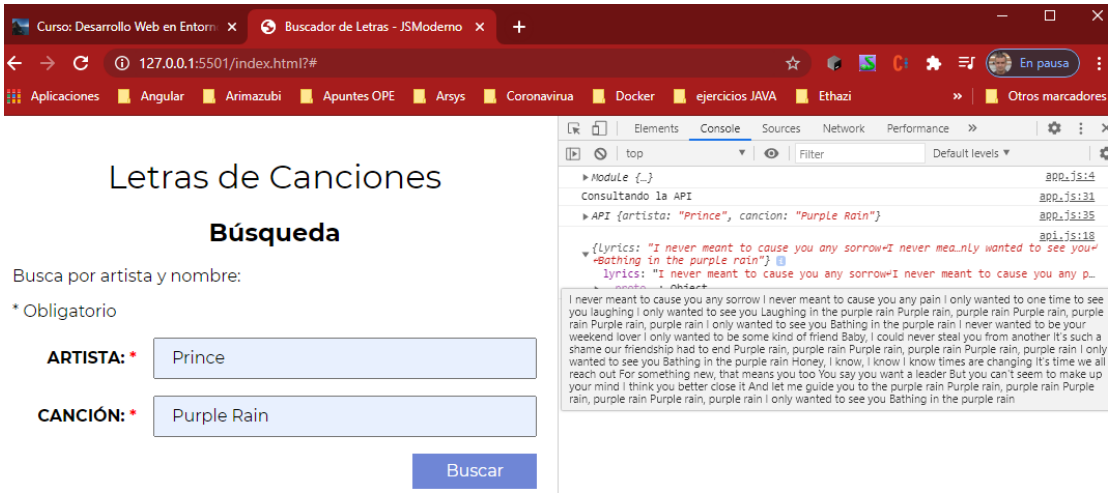
Copiamos la url en nuestra API:

```
5   class API {
6
7       constructor(artista, cancion) {
8           this.artista = artista;
9           this.cancion = cancion;
10      }
11
12      consultarAPI() {
13          const url = `https://api.lyrics.ovh/v1/artist/title`;
14
15          fetch(url)
16              .then(respuesta => respuesta.json());
17      }
18
19  }
```

Sustituimos artista y title por sus valores en la API:

```
3   import * as UI from './interfaz.js';
4
5   class API {
6
7       constructor(artista, cancion) {
8           this.artista = artista;
9           this.cancion = cancion;
10      }
11
12      consultarAPI() {
13          const url = `https://api.lyrics.ovh/v1/${this.artista}/${this.cancion}`
14
15          fetch(url)
16              .then(respuesta => respuesta.json())
17              .then(resultado => {
18                  console.log(resultado);
19              });
20      }
21
22  }
23
```

La salida:

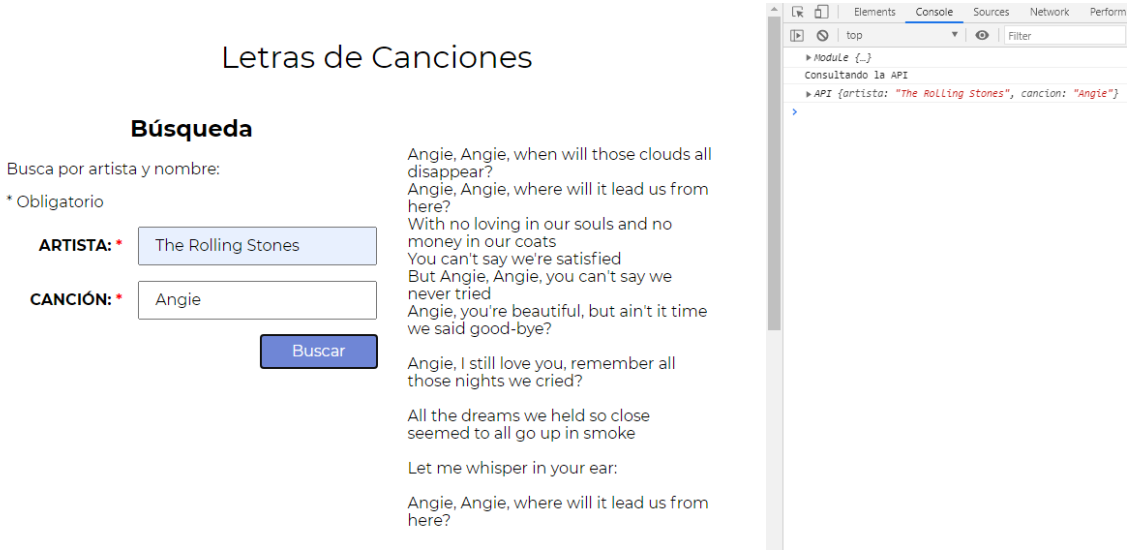


Paso 5: mostrar el resultado.

Esta salida la mostraremos en un div:

```
consultarAPI() {
    const url = `https://api.lyrics.ovh/v1/${this.artista}/${this.cancion}`;

    fetch(url)
        .then(respuesta => respuesta.json())
        .then(resultado => {

            const { lyrics } = resultado;

            UI.divResultado.textContent = lyrics;

        });
}
```

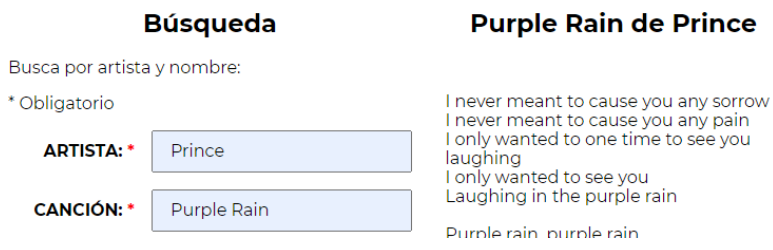
La salida:



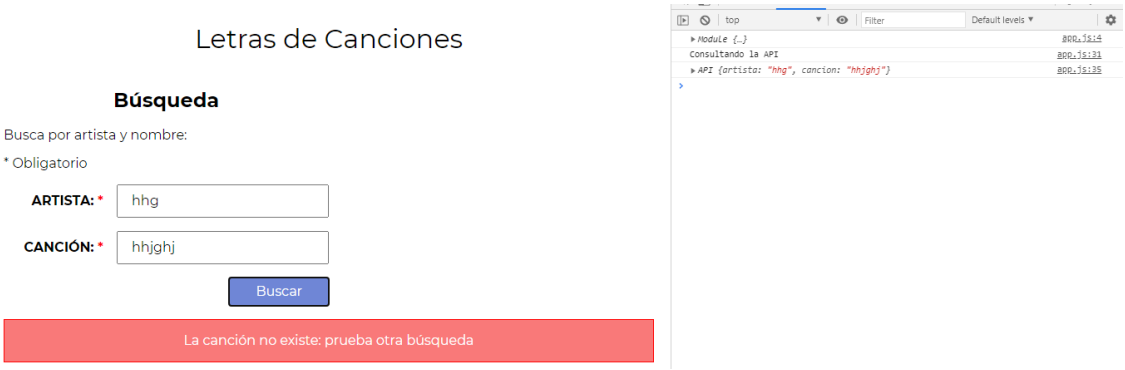
Si queremos poner un título:

```
    const { lyrics } = resultado;

    UI.divResultado.textContent = lyrics;
    UI.headingResultado.textContent = `${this.cancion} de ${this.artista}`;
```

Salida:



Si ponemos una canción que no exista:



Le ponemos un timeOut para mostrar el mensaje sólo 3 segundos y desaparecerá el mensaje limpiando el formulario.