

Proyecto UF3

Cristina Claver

link GitHub: <https://github.com/faisandelviento/proyecto-final.git>

índice

- En qué consiste mi proyecto
- Justificación y motivación del proyecto
- Esquema arquitectura
- Explicación detallada del código
 - como conectarme a las APIS
 - controladores
 - vistas
 - css
- Propuestas de mejora

GITHUB link: <https://github.com/faisandelviento/proyecto-final.git>

En qué consiste mi proyecto

Mi idea es hacer una página web que te recomiende una canción aleatoria al entrar.

Right on time

by Metronomy

0:03 / 0:29

Comprar album en [Amazon](#)

Escuchar en [Spotify](#)

Cada vez que se actualiza la página da una canción nueva





Justificación y motivación

La primera API que encontré que me llamó la atención fue la de Amazon porque es fácil de utilizar y solo entrar a la página se entiende bien qué es lo que ofrece la API, pero no quería hacer una tienda online.

Después de darle unas vueltas pensé en hacer algo relacionado con la música y llegue a la idea de una página web que te recomienda canciones. Para relacionarlo con la API de amazon pensé que la página podría contener un link de compra al álbum de la canción que se recomienda.

Personalmente me gusta buscar playlist hechas por usuarios aleatorios y ver que musica escuchan para encontrar nuevas canciones, así que hacer una pagina con esa temática me pareció interesante

Explicación detallada del código

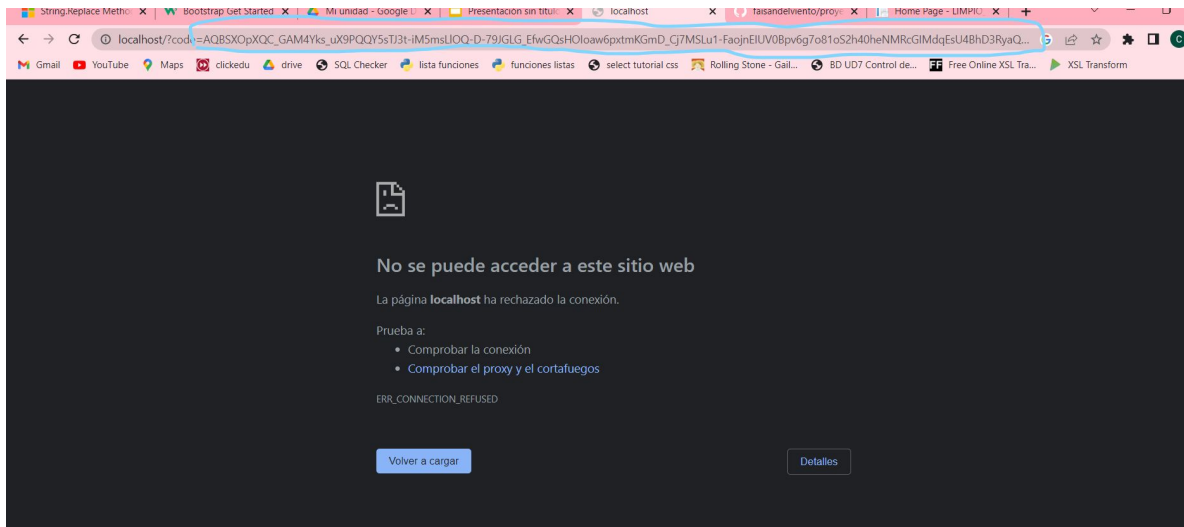
Que hacen en general las APIS utilizadas

- **Spotify** (da los datos de una playlist aka:canciones, foto del album, link canción etc)
[Link](#)
- **Amazon** (haces una búsqueda por ejemplo memory+cards y te da una lista con los resultados de amazon) [link](#)
- **Immaga** (le das una imagen y te devuelve los colores de esta foto con el porcentaje
[Link](#) [Try API](#))

Conectar Spotify API

Ingresar el siguiente link en el navegador:

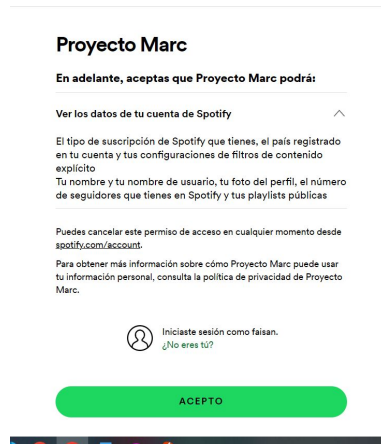
https://accounts.spotify.com/authorize?client_id=e4e9fd2135f24282ba620fa373f2a8be&response_type=code&redirect_uri=http://localhost/&scope=user-read-private



El navegador dará esta respuesta y es necesario copiar lo que va después de “code=” de la URL

Si te pide entrar a Spotify

Mi usuario:
cristina.claver@estudiant.fjaveri
anas.com
Micontraseña: ProyectoMarc



Conectar Spotify API

En el repositorio de github hay un archivo llamado 'ProgramaParaAPISpoty'.

Este programa solo se tiene que ejecutar cuando se necesita el código, no es parte del entorno de la aplicación.

¡Entonces, hay que pegar el "code" en la variable authorizationCode y darle a ejecutar, el acces token saldrá pintado por consola, tiene que copiarlo.

```
0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        string forCode = "https://accounts.spotify.com/authorize?client_id=e4e9fd2135f24282ba620fa373f2a8be&response_type=code&redirect_ur";

        // ingresar el resultado de code= en la variable de abajo.
        string authorizationCode = "AQDi_gxHYFYOE9vqAJey4UgpEL55RlSRaagjNzu1kcYJ23T_puO8RPngxAVQ7cg4KkSzDEjQAin5ywtBTdzJdpeGv6ioCsT0EKxEJ";

        string clientId = "e4e9fd2135f24282ba620fa373f2a8be";
        string clientSecret = "8ae21df3d2f54c53a0e1e780f18cb948";
    }
}
```

```
D:\Users\alumn\Desktop\spotifyharta\Program.cs(59,29): warning SYSLIB0014: 'WebClient.WebClient()' está obsoleto: 'WebRequest, HttpWebRequest, ServicePoint, and WebClient are obsolete. Use HttpClient instead.' [D:\Users\alumn\Desktop\spotifyharta\spotifyharta.csproj]
```

```
1
Respuesta del servidor:
```

```
{"access_token":"BQBxvGCLBxj_Df4iUm0i1lWMF9NM3QxYMfukTOwvP7kDhXh6fmZeJwtkhhB0yuPQLE4cubwNug5mSAk1-yXnxkcEcSBPtt1bRnn4ovK2eG9Z_Zp-qv3gm5nKAV7M49rnC9SnmrgFc0g5aJYvwew2QmBSkAIUuBvcNfzU3chBTGesjEmHm2_QAYdvRWIru3zFCYwF09zRxwLx92mRNQNw","token_type":"Bearer","expires_in":3600,"refresh_token":"AQbf8zAXy-uZPmE8CZVUcsLz7_sCGwbjhwPnW4iz56VHNRMETNFkNCuXLQNNwSWJ3RpcX8NkpqQnLi3mGY5F8XXfxGCz-Tdqg8r-63hqGBleHLQJQmheKIA9sAfTHzYHPw","scope":"user-read-private"}
```

```
PS D:\Users\alumn\Desktop\spotifyharta>
```

Conectar Spotify API

Pegar el token en el proyecto, Controllers, HomeController Linea 29

```
18 public IActionResult Index()  
19 {  
20  
21     var rand = new Random();  
22  
23     //////////conectar API Spotify GetPlaylist  
24     SpotifyGetplaylist datosSpotify = null; //declarar variable aqui para poder acceder fuera del if  
25  
26     const string linkSpotify = "https://api.spotify.com/v1/playlists/37i9dQZF1DXb2RUUTjIk3t";  
27  
28     ///ACCESS TOKEN  
29     const string accessToken = "BQDOefT9zx2ptXkn5J8KEmBUPMiibkcyQnKWEMu-2QCPaDkU_x_pg5FhcMMY_aveSdMnsTJ_1hRH-86547DVUxPD9Vkl";  
30     var clientSpoti = new HttpClient();  
31     clientSpoti.DefaultRequestHeaders.Add("Authorization", "Bearer " + accessToken); //autorizacion para conectar API  
32     var responseSpoty = clientSpoti.GetAsync(linkSpotify).Result;  
33 }
```

Listo :D

La conexión durará 1 hora

Conectar AMAZON API

La parte del proyecto que utiliza la api de amazon está comentada porque mi cuenta ha llegado al límite de request y aun creando una nueva cuenta no me permite utilizar la API, pero el código funciona.

Es posible que desde un ordenador diferente o Ip diferente la API deje hacer request (no lo he podido comprobar)

```
60 // //////////////////////////////////Conectar API AMAZON Search
61
62
63 // string linkSearchAmazon = "https://api.rainforestapi.com/request?api_key=BAB510A2176344DE97D5BBB6AE3D7B2C&type=search&amazon_domain=
64 // var clientSearch = new HttpClient();
65 // var responseSearch = clientSearch.GetAsync(linkSearchAmazon).Result;
66 // var contentSearch = responseSearch.Content.ReadAsStringAsync().Result;
67 // SearchAmazon busquedaAmazon = JsonConvert.DeserializeObject<SearchAmazon>(contentSearch);
68 // //datos amazon
69 // ViewBag.LinkAlbumAmazon = busquedaAmazon.search_results[0].link;/// el link lo paso con ViewBag p
70
```

Requests:

100 of 100 left

[Refresh credit usage](#)

 Choose a Plan

Close Account

controladores

Spotify

1. La Api de Spotify requería ir con un encabezado que llevaba el 'token' de acceso, cada 1 hora este token caduca y hay que actualizarlo
2. En esta API compruebo si la respuesta de la API es correcta y si lo es paso los datos a Json

```
////////conectar API Spotify GetPlaylist
SpotifyGetplaylist datosSpotify = null; //declarar variable aqui para poder acceder fuera del if

const string linkSpotify = "https://api.spotify.com/v1/playlists/37i9dQZF1DXb2RUUTjIk3t";
const string accessToken = "BQDu3gIDEzrrQsBYjsNPzdm3xgKrbgPTsOhsbLYEQb6pIVJUBTPrY6kTay1IyX1oKp_R7WuHNHbkd2knqKTS9Qh";
var clientSpoti = new HttpClient();
1 clientSpoti.DefaultRequestHeaders.Add("Authorization", "Bearer " + accessToken); //autorizacion para conectar API
var responseSpoty = clientSpoti.GetAsync(linkSpotify).Result;

2 if (responseSpoty.IsSuccessStatusCode) //comprobar que api funciona
{
    string contentSpoti = responseSpoty.Content.ReadAsStringAsync().Result;
    Console.WriteLine("SOLICITUD EXITOSA!!!!!!!!");
    //accedemos a los datos de una playlist de Spotify
    datosSpotify = JsonConvert.DeserializeObject<SpotifyGetplaylist>(contentSpoti);
}
else
{
    Console.WriteLine("La solicitud NO FUE EXITOSA. Código de estado: " + responseSpoty.StatusCode);
}

//Escogemos una cancion de la plylist aleatoriamente y nos guardamos
```

Spotify (manejo de datos)

Los datos que se necesitan para las otras APIs se extraen individualmente y el resto se pasa por la View.

1. Escogemos los datos de una canción aleatoria de la playlist mediante un random y lo guardamos en una variable. Pasaremos esta variable por el View en vez del Json completo para que la canción sea la misma fuera de la View
2. Necesitamos (título álbum+artista) para poder hacer la búsqueda con URL en la API de Amazon con cierta precisión.
Tiene que tener un formato específico. EJ “nirvana+nevermind”

```
//Escogemos una cancion de la plylist aleatoriamente y nos guardamos
// 1 titulo album+nombre artista para buscar el album el Amazon
// 2 la informacion del Json de la cancion para pasarla por la View
List<Item> cancionesPlaylist = datosSpotify.tracks.items; //lista de las canciones de la playlist

int indiceAleatorio= rand.Next(cancionesPlaylist.Count()); //hacemos un random con la 'length' de la playlist
var datosCancion= datosSpotify.tracks.items[indiceAleatorio]; //guardamos datos de cancion aleatoria, esto lo pasaremos a la view
string tituloAlbum =datosCancion.track.album.name;
string artistaAlbum=datosCancion.track.artists[0].name;

string tituloArtista= tituloAlbum+" "+artistaAlbum; //un tamos titulo y artista para busqueda mas precisa
string albumArtistaLink= tituloArtista.Replace(" ", "+"); //modificamos el titulo para poder añadirlo al link de Amazon
```

Amazon

1. Formar el link para la API de amazon añadiendo el resultado de la API de spotify
2. Conectamos a la API
3. No paso el Json por el View, solo necesito 1 dato de esta API así que lo paso mediante ViewBag

```
/////////Conectar API AMAZON Search
```

1

```
string linkSearchAmazon = "https://api.rainforestapi.com/request?api_key=BAB510A2176344DE97D5BBB6AE3D7B2C&type=search&amazon_domain=amazon.com&search_term="+albumArtistal;
var clientSearch = new HttpClient();
var responseSearch = clientSearch.GetAsync(linkSearchAmazon).Result;
var contentSearch = responseSearch.Content.ReadAsStringAsync().Result;
SearchAmazon busquedaAmazon = JsonConvert.DeserializeObject<SearchAmazon>(contentSearch);
//datos amazon
ViewBag.LinkAlbumAmazon = busquedaAmazon.search_results[0].link;/// el link lo paso con ViewBag pq solo necesito 1 dato de esta api y es mas facil que pasar los 2 Json
```

3

Imagga

Para conectar a esta API no utilizo HttpClient, utilizó 'RestSharp, el ejemplo en c# para conectar a la API de la documentación de imagga utilizaba eso así que copie pegue. Aunque modifique algunas cosas que daban error.

1. La foto que le paso a la API la sacó de la api de spotify y es la portada del álbum

Guardo algunos colores de la foto para estilar, la posición 0 es el color predominante y la última es el color menos predominante (1% de la foto mínimo). He determinado que posición cojones a ojo.

```
// //Conectar api colores
string apiKey = "acc_6cc2350e33b8254";
string apiSecret = "fd03cc394de5c46d52758a000c683a7d";
string imageUrl = datosCancion.track.album.images[0].url;

string basicAuthValue = "YwNjXzZjYzIzNTBlMzNiODI1NDpmZDAzY2MzOTRkZTVjNDZkNTI3NTNhMDAwYzY4M2E3ZA==";

var client = new RestClient("https://api.imagga.com/v2/colors");
var request = new RestRequest();
request.AddParameter("image_url", imageUrl);
request.AddHeader("Authorization", String.Format("Basic {0}", basicAuthValue));
var response = client.Execute(request);

Root datosColor = JsonConvert.DeserializeObject<Root>(response.Content);

ViewBag.colorMayoritario=datosColor.result.colors.image_colors[0].html_code;
var listaColores = datosColor.result.colors.image_colors;
int numeroColores= listaColores.Count();
if (numeroColores>=5){
    numeroColores=5;
}
ViewBag.colordetalles = datosColor.result.colors.image_colors[numeroColores-1].html_code;
if (numeroColores%2!= 0){
    numeroColores= numeroColores-1;
}
ViewBag.colordetalles2 = datosColor.result.colors.image_colors[numeroColores/2].closest_palette_color_parent;
```

vistas

Solo utilizó una vista que se llama 'index'.

```
<div class="container">
  <div class="row">
    <div class="col-sm">
      <h1 class="display-4" style="color:@ViewBag.colorMayoritario">@Model.track.name</h1>
      <p style="color:@ViewBag.colordetalles"> by @Model.track.artists[0].name</p>
      <audio controls autoplay> @*el autoplay a veces no funciona *@
        <source src=@Model.track.preview_url type="audio/mpeg">
        <source src=@Model.track.preview_url type="audio/ogg">
      </audio>
      <div class="row" style="width:90%;">
        <p>Comprar album en <a style="color:@ViewBag.colordetalles" href= @ViewBag.LinkAlbumAmazon>Amazon</a></p>
        <p>Escuchar en <a style="color:@ViewBag.colordetalles" href=@Model.track.album.external_urls.spotify>Spotify</a></p>
      </div>
    </div>
  </div>
```

1 columna

- Utilizo @Model para los datos de Spotify y @ViewBag para el resto
- He tenido que estilar los colores desde el html ya que no he encontrado una manera de pasarle los valores que me da la API al css
- contiene un reproductor de audio que recibe el audio de la canción de la API de spotify

Index

2 columna

1. Hay un container que tiene 2 fotos, si las clicas te lleva al link del álbum de spotify.
2. La imagen del disco sobresale del container hacia la izquierda un 40% y la imagen del álbum se sobrepone a la del disco siempre. (explicado en detalle en la parte de CSS)

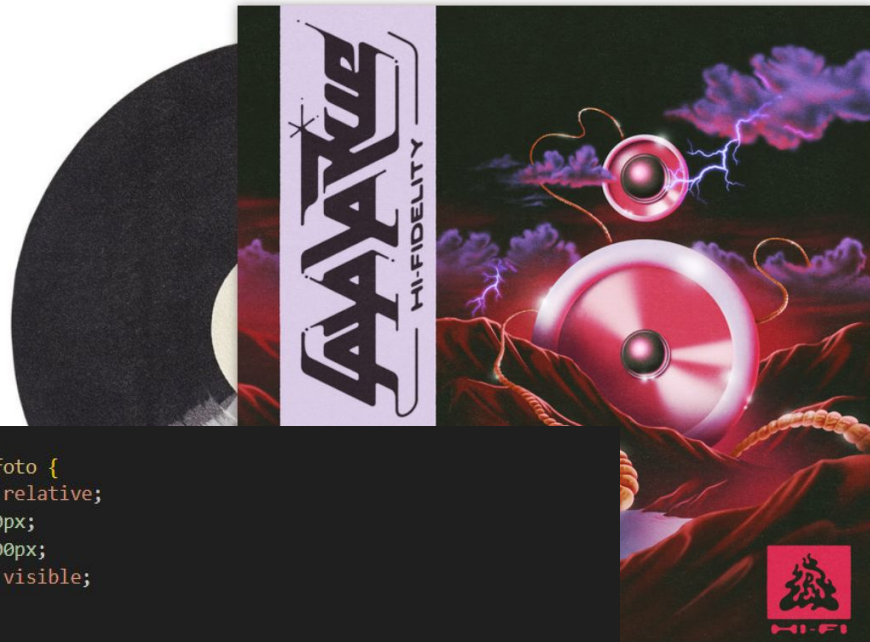


```
<div class="col-sm">  
  <div id="container-foto">  
    <a href=@Model.track.album.external_urls.spotify>1  
       @* queda mal??  
      <img src=@Model.track.album.images[0].url alt="Cover Album" class="sombra" id="second-image">  
    </a>  
  </div>  
</div>
```

CSS

1. El contenedor que almacena las imágenes es del mismo tamaño que las imágenes para que sea más fácil de manejar.
2. el container permite 'overflow', que la imagen se vea si se sale del tamaño del container
3. la primera imagen se mueve a la izquierda un 40% de la posición normal "right:40%"
4. y hay un tag sombra que crea una pequeña sombra, esta esta en la segunda imagen

```
#container-foto {  
  position: relative;  
  width: 600px;  
  height: 600px;  
  overflow: visible;  
}  
  
#first-image {  
  position: absolute;  
  top: 0;  
  right: 40%;  
  z-index: -1;  
}  
  
#second-image {  
  width: 100%;  
}  
  
.sombra {  
  box-shadow: 0px 0px 10px rgba(51, 40, 40, 0.5);  
}
```



Propuestas de mejora

Podría haber otra ruta que en vez de recomendar canciones recomiende en álbum entero y en la misma página salgan todas las pistas del álbum.

Se podrían hacer las recomendaciones por géneros musicales

Critical Mind Dump EP

de DEZ DARE




- ▶ 1. CMD OK! 03:37
- ▶ 2. Uncanny Velocity 01:18
- ▶ 3. Phase Transitions 00:37
- ▶ 4. Weasel Breath 04:26



Propuestas de mejora

Quería añadir fotos de como eran los CDs físicos mediante la API de Amazon, el problema fue que el número de fotos de los productos de amazon varía según el producto y que muchas veces no sale la foto del CD físico.



QUEEN
GREATEST HITS II

Haz clic para obtener una vista ampliada

Greatest Hits II
Imported ed.
Remasterizado, Importación
Queen (Artista) | Formato: CD de audio
4,8 ★★★★★ 5.575 valoraciones

9⁹⁹ €


Devoluciones GRATIS
Los precios incluyen IVA. El precio final a pagar al finalizar:
Ver detalles
> Ver los 24 formatos y ediciones

Streaming Unlimited	MP3 8,99 €	CD de audio 9,99 €
------------------------	---------------	-------------------------------

Escucha con
Aplicación gratuita

6 De 2ª mano desde 6,67 €
18 Nuevo desde 9,99 €
1 De coleccionista desde 7,90 €

AutoRip >>



HAMILTON
AN AMERICAN MUSICAL
BY LIN-MANUEL MIRANDA
ORIGINAL BROADWAY CAST RECORDING

Hamilton (2 CD's)
2 CDs
Original Broadway Cast Of Hamilton (Artista, Colaborado)
4,9 ★★★★★ 12.942 valoraciones

15⁷⁶ €

Devoluciones GRATIS
Los precios incluyen IVA. El precio final a pagar al finalizar:
Ver detalles
> Ver los 4 formatos y ediciones

Streaming Unlimited	MP3 14,99 €	CD de audio 15,76 €
------------------------	----------------	--------------------------------

Escucha con
Aplicación gratuita

5 De 2ª mano desde 13,00 €
9 Nuevo desde 15,76 €

Propuestas de mejora

Me hubiera gustado tener más habilidades para hacer un mejor estilado en la página.

Quería crear mi propio botón para activar el audio pero requería JS y el que encontré solo con Html no funcionaba bien así que tuve que dejar la idea.

Bocetos que hice de la página

