

Desarrollo Web en Entorno a Cliente: Spotify API

Inicio

1. Crea un proyecto público en Github para gestionar las tareas de nuestra app de Spotify.....	2
Postman.....	3
2. Sigue las instrucciones de la web para desarrolladores de Spotify para obtener un access token y hacer tu primera llamada a la API.....	3
1. Inicia sesión en el panel de control usando tu cuenta de Spotify.....	3
2. Crea una aplicación y selecciona "Web API" en la pregunta sobre qué API planeas usar. Una vez que hayas creado tu aplicación, tendrás acceso a las credenciales de la aplicación. Estas serán necesarias para la autorización de API para obtener un token de acceso.....	4
3. Utilice el token de acceso en sus solicitudes de API	5
3. Usa gitignore para no subir nunca el access token a tu repositorio.....	5
4. Proyecto.....	6
index.html.....	6
Propósito de index.html.....	6
Funciones de index.html.....	6
Flujo de interacción de index.html.....	7
auth.js.....	8
Propósito de auth.js.....	8
Funciones de auth.js.....	8
Flujo de interacción de auth.js.....	8
token.php.....	10
Propósito de token.php.....	10
Funciones de token.php.....	10
Flujo de interacción de token.php.....	10
Interacción final.....	12
Ruta y archivos finales.....	12
index.html.....	13
Redirección Spotify, login y obtención de code.....	13
Reenviamos token + code.....	14
access_token.....	15
token_type.....	15
expires_in.....	15
refresh_token.....	15
scope.....	15
Postman.....	16

1. Crea un proyecto público en Github para gestionar las tareas de nuestra app de Spotify

URL: <https://github.com/FabiolaYon/DesWebEntCliente/tree/practicaDos>

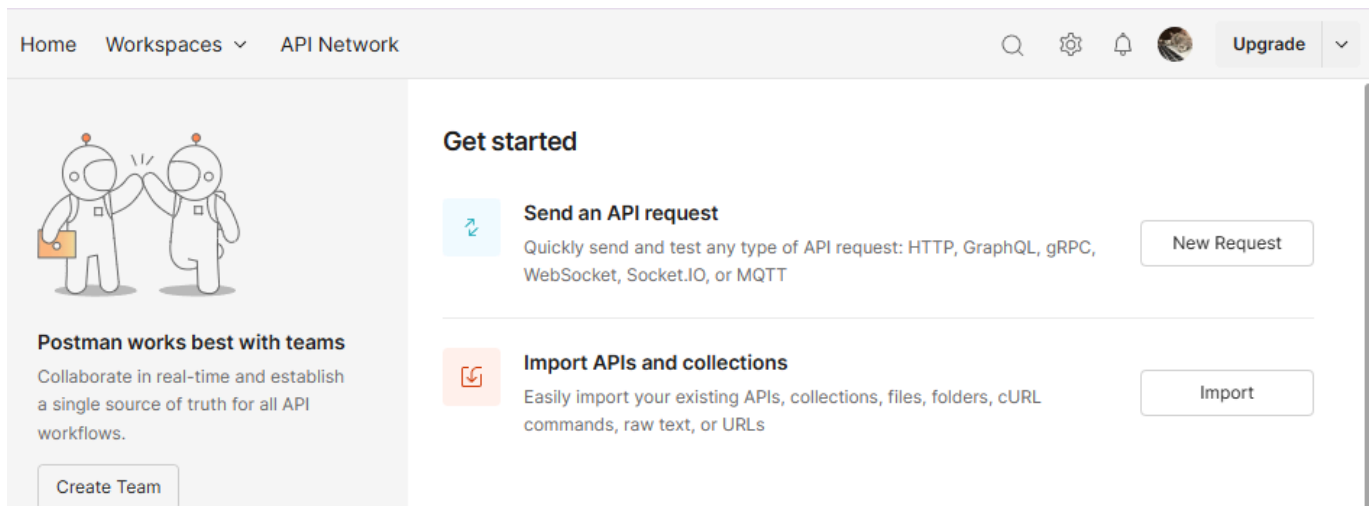
Proyecto: project_SpotifyAPI

En el proyecto he colocado las tareas principales, iré agregando más.

The screenshot shows a GitHub repository named 'DesWebEntCliente' by user 'FabiolaYon'. The repository is public and has 3 branches and 0 tags. The main branch is 'practicaDos', which is 1 commit ahead of and 14 commits behind the 'main' branch. The repository contains two files: 'readme.md' and 'test.txt', both committed 6 minutes ago. The README file is displayed, showing the title 'DesWebEntCliente', the subtitle 'DAW 2 - Desarrollo de Aplicaciones Web en Entorno Cliente', and the section 'Practica 2 - SpotifyAPI'. Under 'Objetivo', there are three numbered tasks: 1. Crea un proyecto público en Github para gestionar las tareas de nuestra app de Spotify, 2. Sigue las instrucciones de la web para desarrolladores de Spotify para obtener un access token y hacer tu primera llamada a la API, and 3. Usa gitignore para no subir nunca el access token a tu repositorio. A note at the bottom states: 'La entrega será la ruta hacia tu proyecto (tanto el proyecto como el repositorio deberán ser públicos)'. The right sidebar shows the repository's 'About' section, 'Releases' (no releases published), and 'Packages' (no packages published).

The screenshot shows a Trello board for the project 'project_SpotifyAPI'. The board is organized into three columns: 'Todo' (2/5 items), 'In Progress' (1/5 items), and 'Done' (0 items). The 'Todo' column contains two tasks: 'DesWebEntCliente #4' (Sigue las instrucciones de la web para desarrolladores de Spotify para obtener un access token y hacer tu primera llamada a la API) and 'DesWebEntCliente #5' (Usa gitignore para no subir nunca el access token a tu repositorio). The 'In Progress' column contains one task: 'DesWebEntCliente #3' (Crea un proyecto público en Github para gestionar las tareas de nuestra app de Spotify). The 'Done' column is currently empty. The board also features a 'Filter by keyword or by field' option and a '+ New view' button.

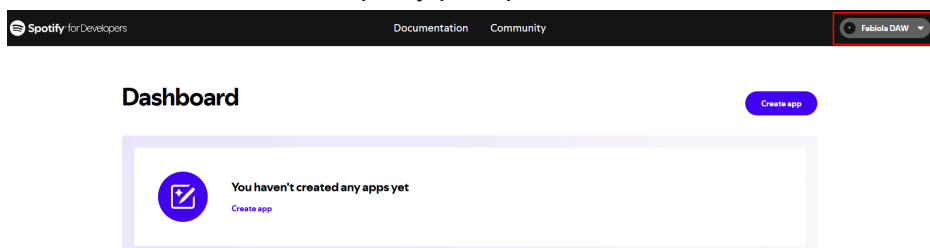
Postman



2. Sigue las instrucciones de la [web para desarrolladores de Spotify](#) para obtener un access token y hacer tu primera llamada a la API

1. Inicia sesión en el panel de control usando tu cuenta de Spotify.

He creado una cuenta de Spotify para poder continuar FabiolaDAW



Dashboard > DesWebEntCliente_Fabiola Home

Settings

D

Home

All Stats

Active Users

Endpoints

Locations

Daily Active Users

2

1

0

Sun Dec 15 2024

Sun Dec 22 2024

Sun Dec 29 2024

Sun Jan 05 2025

Monthly Active Users

2

1

0

Sun Dec 15 2024

Sun Dec 22 2024

Sun Dec 29 2024

Sun Jan 05 2025

3. Utilice el token de acceso en sus solicitudes de API .

Client ID: 47c200312a434a9db75bec9b963c3b03

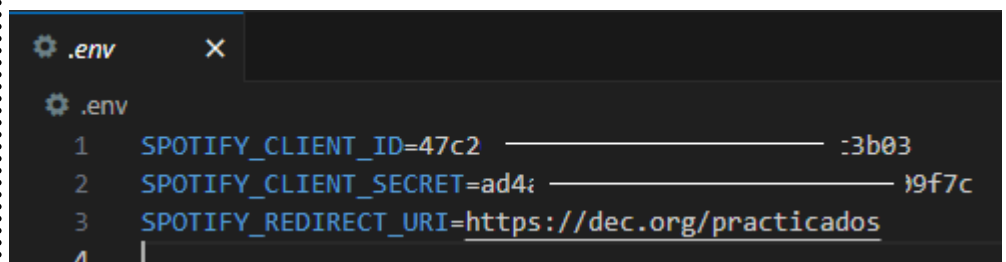
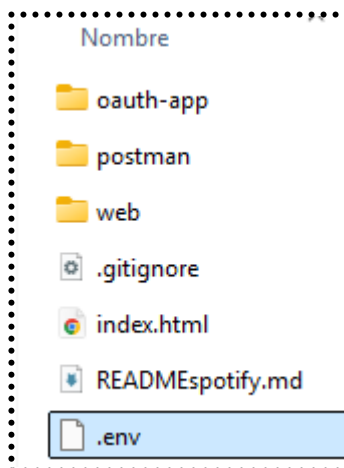
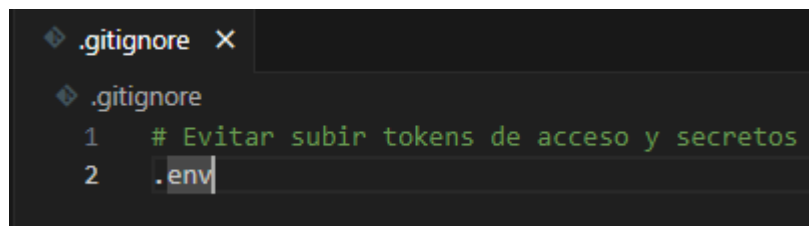
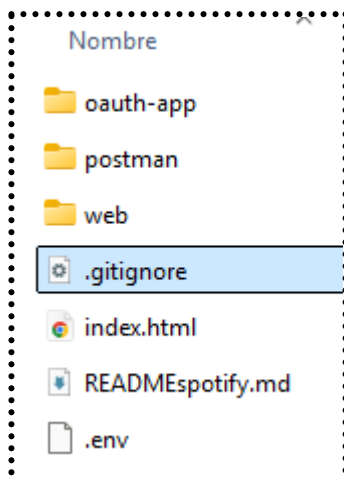
Client secret: ad4a53a6c7fe4a0283fcddbcd6c09f7c

Basic Information

Basic Information User Management Extension Requests

Client ID	App Status
47c200312a434a9db75bec9b963c3b03	Development mode
Client secret	
ad4a53a6c7fe4a0283fcddbcd6c09f7c	
Hide client secret	Rotate client secret

3. Usa gitignore para no subir nunca el access token a tu repositorio.



No subiré al repositorio el archivo .env

Editamos Redirect URIs para que conecte con nuestro proyecto de forma local.

Redirect URIs

- <https://dec.org/practicados>

Bundle IDs

Android packages

APIs used

- Web API

Edit

Redirect URIs *

<https://dec.org/practicados> **Remove**

Add

URIs where users can be redirected after authentication success or failure

Redirect URIs

- <http://localhost/FABIOLA/spotify-app-main>
- <https://dec.org/practicados>

4. Proyecto

index.html

Propósito de index.html

- Proporciona un botón que permite iniciar sesión con Spotify.
- Se encarga de detectar el code en la URL cuando Spotify redirige de vuelta a tu aplicación.
- Llama a las funciones de auth.js para manejar el flujo de autenticación.

Funciones de index.html

- `<button id="spotify-login">Spotify login</button>` : Botón que inicia el proceso de autenticación con Spotify.
- `if (window.location.search.includes("code")) { handleSpotifyRedirect();}` : Cuando Spotify redirige al usuario después de un inicio de sesión exitoso, incluye un code en la URL de la aplicación
- `redirectToSpotifyLogin`: Gestiona la redirección a Spotify.

- `handleSpotifyRedirect`: Gestiona la redirección de vuelta desde Spotify, extrae el code de la URL y llama a `token.php` para intercambiarlo por un `access_token`.

Flujo de interacción de `index.html`

```
[Usuario carga index.html] --> [Hace clic en el botón de login]
--> [Se redirige a Spotify para iniciar sesión]
--> [Spotify redirige con un code]
--> [index.html detecta el code y lo procesa usando auth.js]
--> [access_token es obtenido y almacenado]
```

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Spotify API Login</title>
</head>
<body>
  <h1>Spotify for Developers: DesWebEntCliente_Fabiola</h1>
  <button id="spotify-login">Spotify login</button>

  <script type="module">
    import { redirectToSpotifyLogin, handleSpotifyRedirect } from
'./src/auth.js';

    // Asigna la funcionalidad al botón de login
    const loginButton = document.getElementById('spotify-login');
    loginButton.addEventListener('click', redirectToSpotifyLogin);

    // Maneja el redireccionamiento al cargar la página
    if (window.location.search.includes("code")) {
      handleSpotifyRedirect();
    }
  </script>
</body>
</html>
```

auth.js

Propósito de auth.js

- Redirigir al usuario a la página de inicio de sesión de Spotify.
- Capturar el code que Spotify devuelve tras un inicio de sesión exitoso.
- Enviar el code a tu servidor backend (como token.php) para intercambiarlo por un access_token.
- Limpiar la URL después de procesar el código para evitar problemas de reutilización.

Funciones de auth.js

- redirectToSpotifyLogin: Redirigir al usuario a la página de inicio de sesión de Spotify con los parámetros necesario
- handleSpotifyRedirect: Procesar el code que Spotify devuelve tras el inicio de sesión exitoso.
- getUserProfile: Propósito: Realizar solicitudes autenticadas a la API de Spotify usando el access_token.

Flujo de interacción de auth.js

- Usuario hace clic en "Iniciar sesión con Spotify":
 - redirectToSpotifyLogin redirige al usuario a la página de inicio de sesión de Spotify.
 - Spotify pide al usuario que inicie sesión y otorgue permisos.
- Spotify redirige de vuelta a tu aplicación:
 - El navegador es redirigido a tu redirect_uri con un parámetro code en la URL.
- auth.js procesa el code:
 - handleSpotifyRedirect detecta el code en la URL.
 - Envía el code al backend (token.php) para intercambiarlo por un access_token.
- El backend responde con el access_token:
 - auth.js guarda el access_token en localStorage.
- Realizar solicitudes autenticadas:
 - Usando el access_token, puedes realizar solicitudes a la API de Spotify para obtener datos del usuario o interactuar con sus playlists.

```
[Usuario hace clic] --> [auth.js redirige a Spotify para login]
--> [Spotify devuelve un "code" a tu aplicación]
--> [auth.js envía el "code" a token.php]
--> [token.php responde con un "access_token"]
--> [auth.js guarda el "access_token"]
--> [Realizas solicitudes autenticadas con el token]
```

```
const clientId = "...";
const redirectUri = "http://localhost/...";
const scopes = [
  "user-read-private",
  "user-read-email",
  "playlist-read-private",
  "playlist-read-collaborative"
];
```



```

function getSpotifyAuthUrl() {
  var authUrl = "https://accounts.spotify.com/authorize";
  var url = authUrl +
    "?response_type=code" +
    "&client_id=" + encodeURIComponent(clientId) +
    "&scope=" + encodeURIComponent(scopes.join(" ")) +
    "&redirect_uri=" + encodeURIComponent(redirectUri);
  return url;
}

function redirectToSpotifyLogin() {
  var authUrl = getSpotifyAuthUrl();
  window.location.href = authUrl;
}

async function handleSpotifyRedirect() {
  var params = new URLSearchParams(window.location.search);
  var code = params.get("code");

  if (code) {
    try {
      // Envía el código al servidor para intercambiarlo por un token
      var response = await fetch("/web/token.php?code=" + code);
      var data = await response.json();

      if (data.access_token) {
        console.log("Token de acceso: " + data.access_token);
        localStorage.setItem("spotify_access_token",
data.access_token);
        alert("Se ha iniciado sesión exitosamente");

        // Limpia la URL para evitar reutilizar el código
        window.history.replaceState({}, document.title,
"/FABIOLA/spotify-app-main");
      } else {
        console.error("Error en el token:", data);
      }
    } catch (error) {
      console.error("Error:", error);
    }
  }
}

// Exporta las funciones para usarlas en el HTML
export { redirectToSpotifyLogin, handleSpotifyRedirect };

```

token.php

Propósito de token.php

- Recibe el code enviado por Spotify a través de tu aplicación frontend (auth.js).
- Envía este code al endpoint de Spotify <https://accounts.spotify.com/api/token> junto con otros parámetros necesarios (como el client_id y el client_secret).
- Recibe la respuesta de Spotify, que incluye el access_token, refresh_token, y otros datos importantes.
- Devuelve esta respuesta al frontend como un JSON.

Funciones de token.php

- if (isset(\$_GET['code'])) { ... : Verifica si el parámetro code fue enviado como parte de la solicitud GET. Si no se proporciona, devuelve un error 400 Bad Request indicando que falta el código.
- \$data = ['grant_type' => 'authorization_code', ... : Configura los parámetros necesarios para solicitar un access_token a Spotify.
- \$options = ['http' => ['header' => "Content-Type: application/x-www-form-urlencoded\r\n", 'method' => 'POST', ... : Envía una solicitud POST al endpoint <https://accounts.spotify.com/api/token> para obtener el access_token.
- if (\$response === false) { http_response_code(500); ... : Si la solicitud a Spotify falla, devuelve un error 500 Internal Server Error al frontend con un mensaje de error.
- echo \$response; : Devuelve el JSON completo de Spotify al frontend.

Flujo de interacción de token.php

- El frontend llama a token.php:
 - El code es enviado como parte de una solicitud GET al backend : <http://localhost/FABIOLA/spotify-app-main/web/token.php?code=AQAC...>
- token.php verifica el code:
 - Si el code no está presente, devuelve un error.
 - Si el code está presente, continúa con el flujo.
- Solicitud al endpoint de Spotify:
 - token.php envía una solicitud POST a <https://accounts.spotify.com/api/token> con los datos necesarios.
 - Spotify responde con un JSON que incluye el access_token.
- Respuesta al frontend:
 - token.php devuelve el JSON recibido de Spotify al frontend.
 - El frontend usa esta información para almacenar el access_token y realizar solicitudes autenticadas.

```
{
  "access_token": "BQD1zkj...",
  "token_type": "Bearer",
  "expires_in": 3600,
  "refresh_token": "AQ CZ_6kZniheQTZ9RqyR...",
  "scope": "playlist-read-private playlist-read-collaborative user-read-email user-read-private"
}
```

```
<?php

header('Content-Type: application/json');

// Obtén el código de la URL
if (isset($_GET['code'])) {
    $code = $_GET['code'];
} else {
    $code = null;
}

if (!$code) {
    http_response_code(400);
    echo json_encode(["error" => "Falta el parámetro 'code'"]);
    exit;
}

// Configuración de la aplicación Spotify
$clientId = ' '; //Client ID
$clientSecret = ' '; // Client Secret
$redirectUri = 'http://localhost/...'; // Redirect URI

// Datos para la solicitud de token
$data = [
    'grant_type' => 'authorization_code',
    'code' => $code,
    'redirect_uri' => $redirectUri,
    'client_id' => $clientId,
    'client_secret' => $clientSecret
];

// Configuración de la solicitud HTTP
$options = [
    'http' => [
        'header' => "Content-Type: application/x-www-form-urlencoded\r\n",
        'method' => 'POST',
        'content' => http_build_query($data)
    ]
];

// Realiza la solicitud a Spotify
$context = stream_context_create($options);
$response = file_get_contents('https://accounts.spotify.com/api/token', false,
    $context);
```

```
if ($response === false) {  
    http_response_code(500);  
    echo json_encode(["error" => "Error al solicitar el token"]);  
    exit;  
}  
  
// Devuelve la respuesta de Spotify  
echo $response;  
?>
```

Interacción final

Ruta y archivos finales

🔄 🖥️ > ... xampp > htdocs > FABIOLA > spotify-app-main >

📄 📁 📄 📄 🗑️ | 🔍 Ordenar ▾ ☰ Ver ▾ ...

Nombre

📁 src
📁 web
📄 .env
⚙️ .gitignore
🌐 index.html

🔄 🖥️ > ... xampp > htdocs > FABIOLA > spotify-app-main > src

📄 📄 📄 📄 🗑️ | 🔍 Ordenar ▾ ☰ Ver ▾ ...

Nombre

📄 auth.js

🔄 🖥️ > ... xampp > htdocs > FABIOLA > spotify-app-main > web

📄 📄 📄 📄 🗑️ | 🔍 Ordenar ▾ ☰ Ver ▾ ...

Nombre

📄 token.php

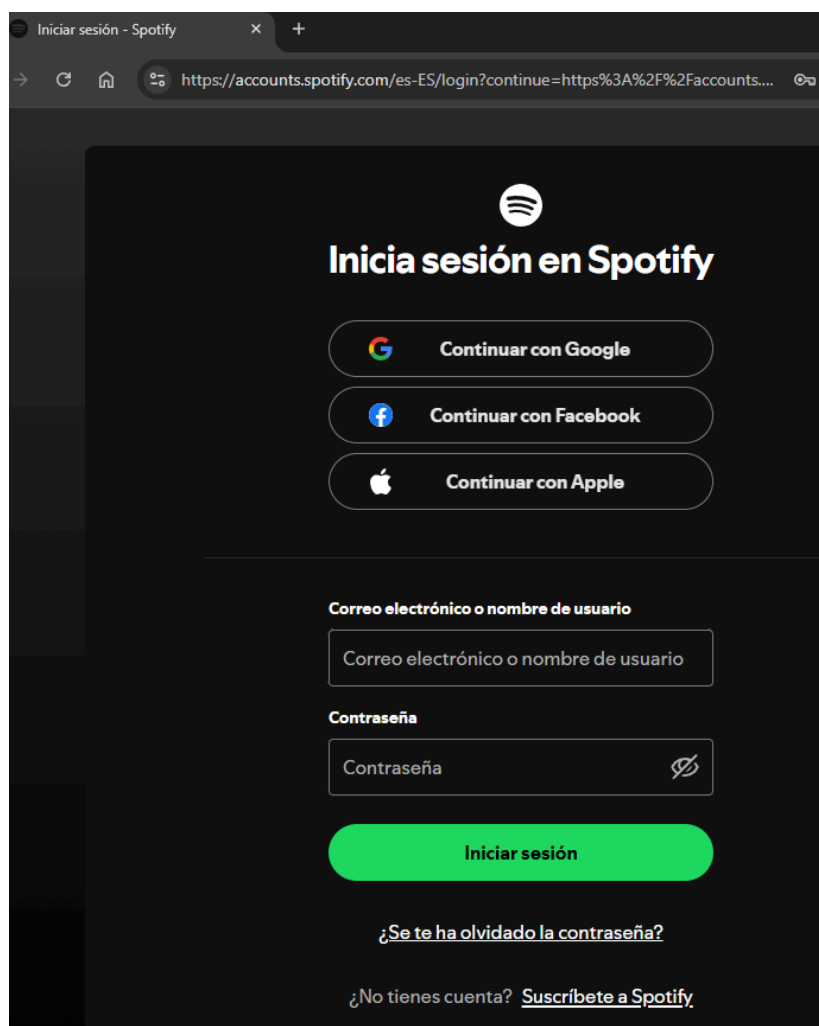
index.html

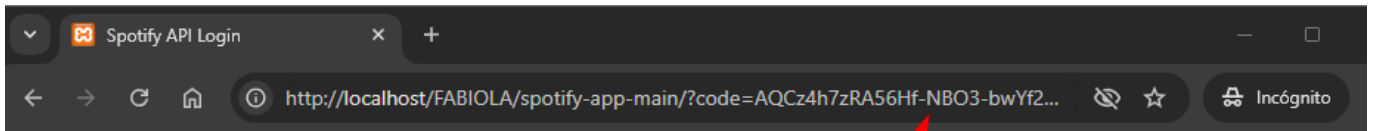
Spotify for Developers: DesWebEntCliente_Fabiola

Spotify login

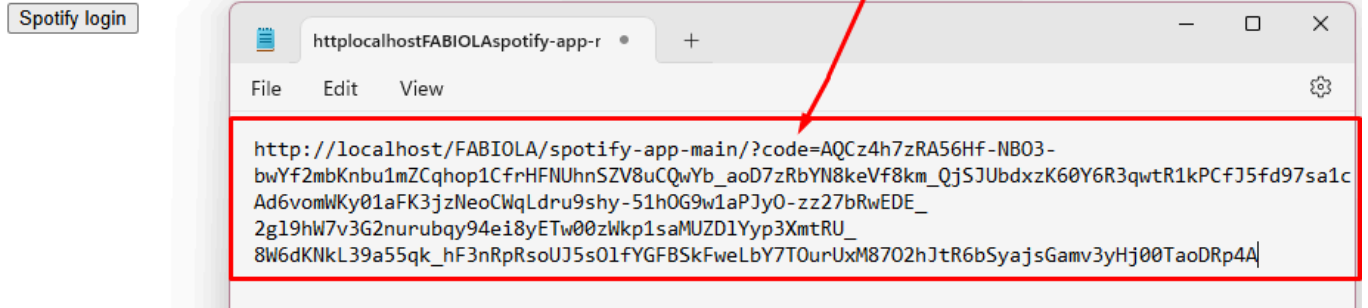


Redirección Spotify, login y obtención de code



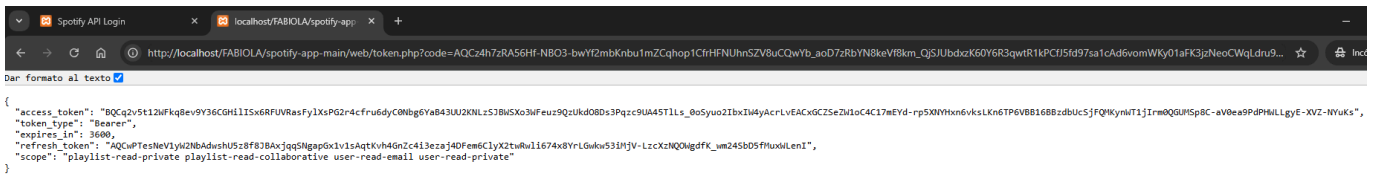


Spotify for Developers: DesWebEntCliente_Fabiola



Reenviamos token + code

http://localhost/FABIOLA/spotify-app-main/web/token.php?code=AQCz4h7zRA56Hf-NBO3-bwYf2mbKnbu1mZCqhop1CfrHFNUhnSZV8uCQwYb_aoD7zRbYN8keVf8km_QjSJUbdxzK60Y6R3qwtR1kPCfJ5fd97sa1cAd6vomWKy01aFK3jzNeoCWqLdru9shy-51hOG9w1aPJyO-zz27bRwEDE_2gl9hW7v3G2nurubqy94ei8yETw00zWkp1saMUZD1Yyp3XmtRU_8W6dKNkL39a55qk_hF3nRpRsoUJ5s01fYGFBSkFweLbY7T0urUxM87O2hJtR6bSyajsGamv3yHj00TaoDRp4A



```
{
  "access_token": "BQCq2v5t12WFkq8ev9Y36CGHil...",
  "token_type": "Bearer",
  "expires_in": 3600,
  "refresh_token": "AQcWPTesNeV1yW2NbAdws...",
  "scope": "playlist-read-private playlist-read-collaborative user-read-email user-read-private"
}
```

access_token

Llave temporal para autenticar solicitudes (válido por 1 hora).

token_type

Tipo de token (siempre "Bearer").

expires_in

Tiempo de validez del access_token (en segundos).

refresh_token

Llave persistente para obtener nuevos access_token.

scope

Permisos otorgados por el usuario a la aplicación.

Postman

The screenshot shows the Postman interface with a GET request to `https://api.spotify.com/v1/me`. The request is configured with a Bearer token in the Authorization header. The response is a JSON object containing user information.

Request Details:

- Method: GET
- URL: `https://api.spotify.com/v1/me`
- Headers (8):

Key	Value	Description
Authorization	Bearer AQCz4h7zRA56Hf-NBO3-bwYf2mbKn...	
Key	Value	Description

Response Body (JSON):

```
1 {
2   "country": "ES",
3   "display_name": "Fabiola DAW",
4   "email": "fabimaqvirtual@gmail.com",
5   "explicit_content": {
6     "filter_enabled": false,
7     "filter_locked": false
8   },
9   "external_urls": {
10    "spotify": "https://open.spotify.com/user/31g5ximxebzxnyobggx23yc3jgre"
11  },
12  "followers": {
13    "href": null,
14    "total": 0
15  },
16  "href": "https://api.spotify.com/v1/users/31g5ximxebzxnyobggx23yc3jgre",
```

Status: 200 OK
Time: 73 ms
Size: 1.29 KB