Tutorialr.com

Spotify for Developers Authorisation Guide

Authorisation guide will show how to authorise your application to get data or to allow an end user to approve your application to access their Spotify data or features. Authorised requests to Spotify require permission to be granted to access data. In accordance with OAuth 2.0 the parties involved in the authorisation process are the End User, Application Client and the Spotify Server. Your application can be authorised in one of two ways. **Spotify API App Authorisation** where you authorise your app to access the Spotify Platform e.g. APIs, SDKs and Widgets and **Spotify API User Authorisation** where you grant your app permission to access or modify the user's own data.

Authorisation Scopes

Authorisation scopes allow your application to access specific API endpoints on behalf of a user. The set of scopes you can pass from your application include **Listening History**, **Library**, **Follow**, **Playlists**, **User & Images**, **Spotify Connect** and **Playback** for iOS, Android and Web Playback SDKs.

Listening History

user-top-read	
Endpoints	
Get a User's Top Artists / Get a User's Top Tracks	
Description	**Visible to users**
Read access to a user's top artists and tracks	Read your top artists and content.

user-read-playback-position	
Endpoints	
Get an Episodes, Get Multiple Episodes, Get a Show, Get Multiple Shows and Get a Show's Episodes	
Description	**Visible to users**
Read access to a user's playback position in a content.	Read your position in content you have played.

user-read-recently-played	
Endpoints	
Get Current User's Recently Played Tracks	
Description	**Visible to users**
Read access to a user's recently played tracks.	Access your recently played items.

Library

user-library-modify	
Endpoints	
Remove Albums for Current User, Remove User's Saved Tracks, Remove User's Saved Shows, Save Albums for Current User, Save Tracks for User and Save Shows for Current User	
Description	**Visible to users**
Write/delete access to a user's "Your Music" library.	Manage your saved content.

user-library-read	
Endpoints	
Check User's Saved Albums, Check User's Saved Tracks, Check User's Saved Shows, Get Current User's Saved Albums, Get a User's Saved Tracks and Get Users Saved Shows	
Description	**Visible to users**
Read access to a user's "Your Music" library.	Access your saved content.

Follow

user-follow-read	
Endpoints	
Get Following State for Artists/Users and Get User's Followed Artists	
Description	**Visible to users**
Read access to the list of artists and other users that the user follows.	Access your followers and who you are following.

user-follow-modify	
Endpoints	
Follow Artists or Users and Unfollow Artists or Users	
Description	**Visible to users**
Write/delete access to the list of artists and other users that the user follows.	Manage who you are following.

Playlists

playlist-read-collaborative	
Endpoints	
Get a List of Current User's Playlists and Get a List of a User's Playlists	
Description	**Visible to users**
Include collaborative playlists when requesting a user's playlists.	Access your collaborative playlists.

playlist-modify-public	
Endpoints	
Follow a Playlist, Unfollow a Playlist, Add Tracks to a Playlist, Change a Playlist's Details, Create a Playlist, Remove Tracks from a Playlist, Reorder a Playlist's Tracks, Replace a Playlist's Tracks and Upload a Custom Playlist Cover Image	
Description	**Visible to users**
Write access to a user's public playlists.	Manage your public playlists.

playlist-read-private	
Endpoints	
Check if Users Follow a Playlist, Get a List of Current User's Playlists and Get a List of a User's Playlists	
Description	**Visible to users**
Read access to user's private playlists.	Access your private playlists.

playlist-modify-private	
Endpoints	
Follow a Playlist, Unfollow a Playlist, Add Tracks to a Playlist, Change a Playlist's Details, Create a Playlist, Remove Tracks from a Playlist, Reorder a Playlist's Tracks, Replace a Playlist's Tracks and Upload a Custom Playlist Cover Image	
Description	**Visible to users**
Write access to a user's private playlists.	Manage your private playlists.

Users & Images

user-follow-read	
Endpoints	
Get Following State for Artists / Users and Get User's Followed Artists	
Description	**Visible to users**
Read access to the list of artists and other users that the user follows.	Access your followers and who you are following.

user-read-private	
Endpoints	
Search for an Item, Get Current User's Profile	
Description	**Visible to users**
Read access to user’s subscription details (type of user account).	Access your subscription details.

ugc-image-upload	
Endpoints	
Upload a Custom Playlist Cover Image	
Description	**Visible to users**
Write access to user-provided images.	Upload images to Spotify on your behalf.

Spotify Connect

user-read-playback-state	
Endpoints	
Get a User's Available Devices, Get Information About The User's Current Playback and Get the User's Currently Playing Track	
Description	**Visible to users**
Read access to a user’s player state.	Read your currently playing content and Spotify Connect devices information.

user-modify-playback-state	
Endpoints	
Pause a User's Playback, Seek To Position In Currently Playing Track, Set Repeat Mode On User’s Playback, Set Volume For User's Playback, Skip User’s Playback To Next Track, Skip User’s Playback To Previous Track, Start/Resume a User's Playback, Toggle Shuffle For User’s Playback, Transfer a User's Playback and Add an Item To User's Current Playback Queue	
Description	**Visible to users**
Write access to a user’s playback state	Control playback on your Spotify clients and Spotify Connect devices.

user-read-currently-playing	
Endpoints	
Get the User's Currently Playing Track	
Description	**Visible to users**
Read access to a user’s currently playing content.	Read your currently playing content.

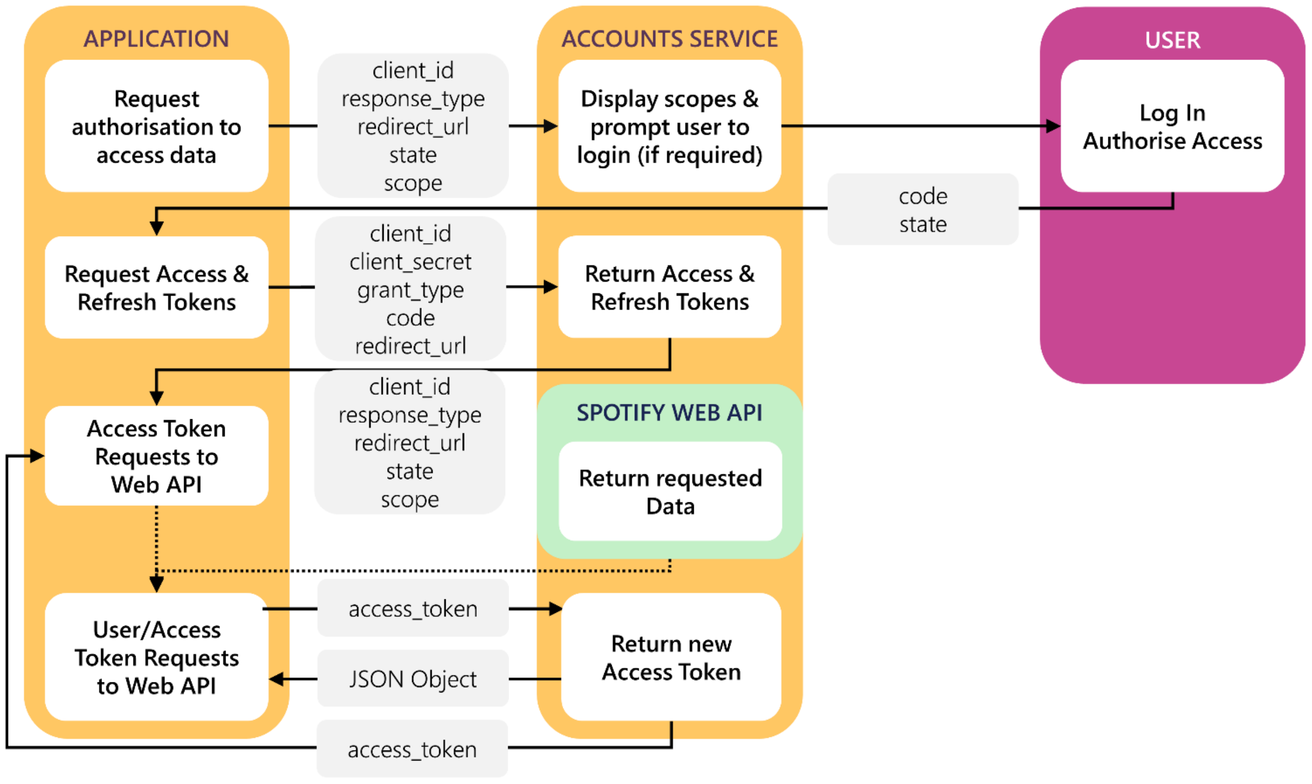
Playback

streaming	
Endpoints	
Web Playback SDK	
Description	**Visible to users**
Control playback of a Spotify track. This scope is currently available to the Web Playback SDK. The user must have a Spotify Premium account.	Play content and control playback on your other devices.

app-remote-control	
Endpoints	
iOS SDK and Android SDK	
Description	**Visible to users**
Remote control playback of Spotify. This scope is currently available to Spotify iOS and Android SDKs.	Communicate with the Spotify app on your device.

Authorisation Code

Authorisation Code flow is designed for long-running applications where the user grants permission once and provides a Token that can be refreshed where there is an initial request from application to the accounts service, then the user can authorise access to the application followed by any requests to the API to return requested data or new access tokens.



Access Code Request

User logs in and authorises access which includes required parameters including the **Client ID** obtained from the **Spotify Dashboard**, the **Redirect URI** which is the URI to redirect to and you can also provide a **State** value.

Query Parameter	
client_id	Client ID provided to you by Spotify when you registered your application
response_type	Set to "code"
redirect_uri	URI to redirect to after the user grants/denies permission. This URI needs to be entered in the URI whitelist that you specify when you registered your application
state	Strongly recommended as state can be useful for correlating requests and responses
scope	A space-separated list of scopes
show_dialog	Whether or not to force user to approve the app again if they've already done so

Example
https://accounts.spotify.com/authorize?client_id=5fe01282e44241328a84e7c5cc169165&response_type=code&redirect_uri=https%3A%2F%2Fexample.com%2Fcallback&scope=user-read-private%20user-read-email&state=STATE

Access Code Response

If the user accepts the request, you'll be redirected to your **Redirect URI** with the **Access Code** parameter. If the user denies the request or there's a problem, then you'll get an error returned – both will include the **State** value provided.

https://example.com/callback?code=NApCCg..BkWtQ&state=STATE	
User Accepts Request	
code	Authorisation code that can be exchanged for Token
state	Value of the state parameter supplied in the request

https://example.com/callback?error=access_denied&state=STATE	
User Denies Request or Error Occurred	
code	Authorisation code that can be exchanged for Token
state	Reason authorisation failed, for example: "access_denied"

Authorisation Code Request

Your application can request a **Refresh Token** and **Access Token** where you provide a base-64 encoded string containing the **Client ID** and **Client Secret** and **Code** from a previous **Access Code Response** and **Redirect URI** by performing a **POST** request with the following **Header** and **Body** parameters.

POST https://accounts.spotify.com/api/token	
Header	
Authorization	Base 64 encoded string containing Client ID and Client Secret **Authorization: Basic <base64 encoded client_id:client_secret>**
Body Parameter	
grant_type	Set to "authorization_code"
code	Authorisation code returned from Access Code Request
redirect_uri	URI to redirect to after the user grants/denies permission. This URI needs to be entered in the URI whitelist that you specify when you registered your application

Authorisation Code Response

If successful you will get an **Access Token** to use in subsequent calls to the **Spotify API** and a **Refresh Token**.

Result	
access_token	Access token that can be provided in subsequent calls e.g. Spotify Web API services
token_type	How the access token may be used - always "Bearer"
scope	Space-separated list of scopes which have been granted for this access_token
expires_in	Time period (in seconds) for which the access token is valid.
refresh_token	Token that can be sent to Spotify Accounts service in place of authorisation code

Refresh Token Request

You can request a refreshed **Access Token** by providing the **Refresh Token** returned from a previous **Authorisation Code Response** by performing a **POST** request with the following **Header** and **Body** parameters.

POST https://accounts.spotify.com/api/token	
Header	
Authorization	Base 64 encoded string containing Client ID and Client Secret **Authorization: Basic <base64 encoded client_id:client_secret>**
Body Parameter	
grant_type	Set to "refresh_token"
refresh_token	Refresh Token returned from the Authorisation Code exchange

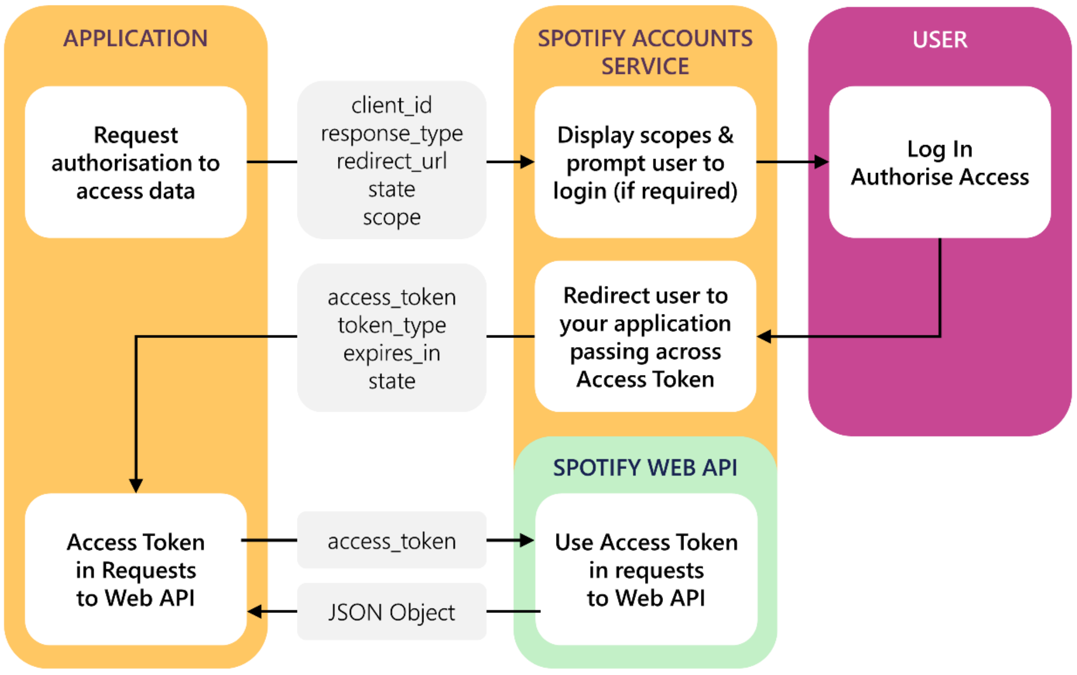
Refresh Token Response

If successful you will get an **Access Token** to use in subsequent calls to the **Spotify API** and can re-use a previously obtained **Refresh Token**.

Result	
access_token	Access token that can be provided in subsequent calls e.g. Spotify Web API services
token_type	How the access token may be used - always "Bearer"
scope	Space-separated list of scopes which have been granted for this access_token
expires_in	Time period (in seconds) for which the access token is valid.

Implicit Grant

Implicit Grant Flow is designed for clients that are implemented such as those using JavaScript and run in a web browser. There is the initial request from the application to the accounts service, then the user can log in and authorise access passing across a short-lived token followed by any requests to the API to return requested data using the token.


APPLICATION
Request authorisation to access data
client_id
response_type
redirect_url
state
scope
SPOTIFY ACCOUNTS SERVICE
Display scopes & prompt user to login (if required)
USER
Log In
Authorise Access
access_token
token_type
expires_in
state
Redirect user to your application passing across Access Token
SPOTIFY WEB API
Access Token in Requests to Web API
access_token
JSON Object
Use Access Token in requests to Web API

Implicit Grant Request

User logs in and authorises access which includes required parameters including the **Client ID** obtained from the **Spotify Dashboard**, the **Redirect URI** which is the URI to redirect to and you can also provide a **State** value.

Query Parameter	
client_id	Client ID provided to you by Spotify when you registered your application
response_type	Set to "token"
redirect_uri	URI to redirect to after the user grants/denies permission. This URI needs to be entered in the URI whitelist that you specify when you registered your application
state	Strongly recommended as state can be useful for correlating requests and responses
scope	A space-separated list of scopes
show_dialog	Whether or not to force user to approve the app again if they've already done so

Example
https://accounts.spotify.com/authorize?client_id=5fe01282e44241328a84e7c5cc169165&response_type=code&redirect_uri=https%3A%2F%2Fexample.com%2Fcallback&scope=user-read-private%20user-read-email&state=STATE

Implicit Grant Response

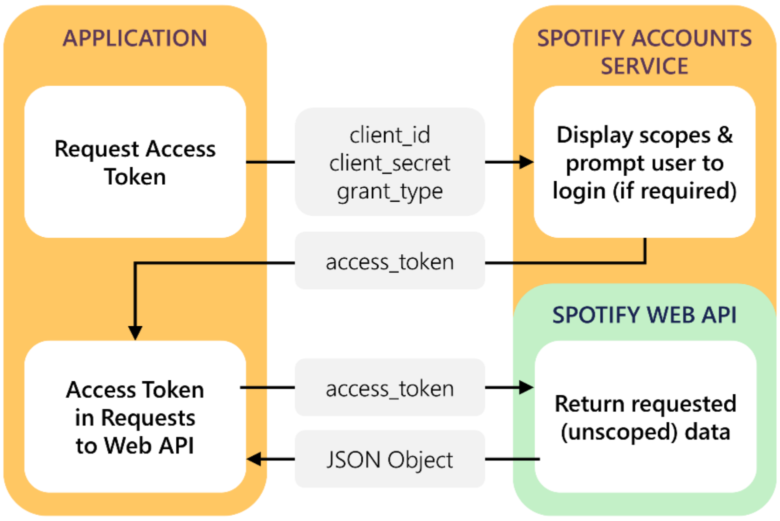
If the user grants access the URI will contain a hash fragment with data encoded as a query string including the **Access Token**, if the user doesn't grant access or there's a problem an error will be returned as a normal query string – both will also include the **State** value provided.

https://example.com/callback#access_token=Nw...2Tk&token_type=Bearer&expires_in=3600&state=STATE	
User Grants Access	
access_token	Token that can be provided in subsequent calls
token_type	Set to "Bearer"
expires_in	The time period (in seconds) for which the access token is valid
state	The value of the state parameter supplied in the request

https://example.com/callback?error=access_denied&state=STATE	
User Denies Access or Error Occurred	
error	Reason authorisation failed, for example: "access_denied"
state	Value of the state parameter supplied in the request

Client Credentials

Client Credentials Flow is designed for server-to-server authentication however only endpoints that do not access user information can be accessed, there is an initial request to obtain a token, you can optionally request the user log in to obtain a higher rate limit, followed by any requests to the API to return requested data using the token.



Client Credentials Request

You can request an **Access Token** by performing a **POST** request with the following **Header** and **Body** parameters.

POST https://accounts.spotify.com/api/token	
Header	
Authorization	Base 64 encoded string containing Client ID and Client Secret **Authorization: Basic <base64 encoded client_id:client_secret>**
Body Parameter	
grant_type	Set to “client_credentials”

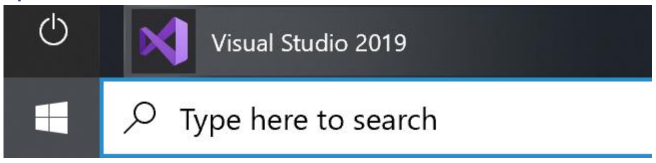
Client Credentials Response

If successful you will get an **Access Token** to use in subsequent calls to the **Spotify API**

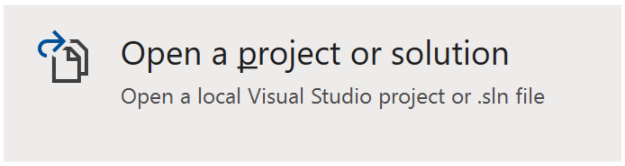
Result	
access_token	Access token that can be provided in subsequent calls e.g. Spotify Web API services
token_type	How the access token may be used - always “bearer”
expires_in	Time period (in seconds) for which the access token is valid.

Authorisation Flows

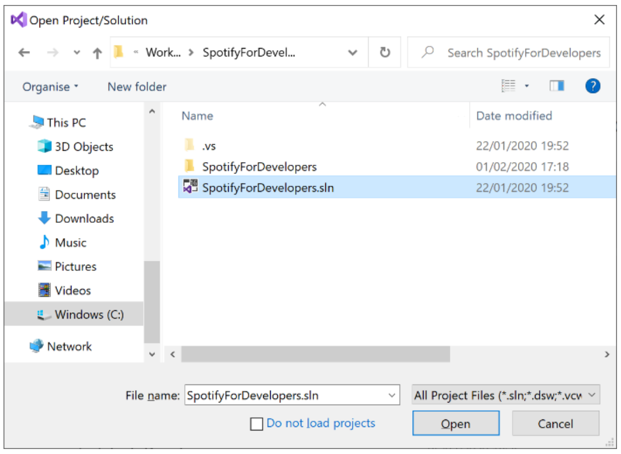
Step 1



In **Windows 10** choose **Start**, and then from the **Start Menu** find and select **Visual Studio 2019**

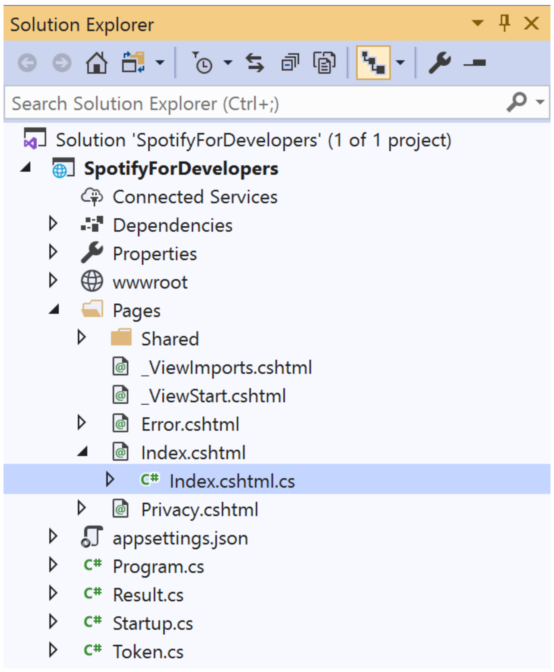

Open a project or solution
Open a local Visual Studio project or .sln file

Once done, from the **Get started** screen for **Visual Studio 2019** select **Open a project or solution**


Open Project/Solution
Work... > SpotifyForDevel...
Search SpotifyForDevelopers
Organise
New folder
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Windows (C:)
Network
Name
Date modified
.vs 22/01/2020 19:52
SpotifyForDevelopers 01/02/2020 17:18
SpotifyForDevelopers.sln 22/01/2020 19:52
File name: SpotifyForDevelopers.sln
All Project Files (*.sln;*.dsw;*.vcw
Do not load projects
Open
Cancel

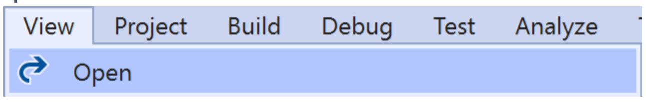
Then locate and select **SpotifyForDevelopers.sln** and select **Open** if you don't have this file already then please follow the previous parts of the workshop including **Getting Started**

Step 2



Once opened, in the **Solution Explorer** open the **Pages** section, then open the **Index.cshtml** section and select **Index.cshtml.cs**

Step 3



Then from the **Menu** choose **View** and then **Open**

Step 4

In the **Code View** for **Index.cshtml.cs** remove the existing `public void OnGet() {}` **method** which will appear like the following:

```
public void OnGet()
{

}
```

Then once removed, replace it with the following **method**:

```
public async Task<IActionResult> OnGetAsync(
    string code = null, string access_token = null)
{
    LoadToken();
    if (code != null)
        Token = new Token(await Api.GetAuthorisationCodeAuthTokenAsync(
            CurrentUri, RedirectUri, state));
    if (access_token != null)
        Token = new Token(Api.GetImplicitGrantAuthToken(
            CurrentUri, RedirectUri, state));
    SaveToken();
    return Page();
}
```

This **method** will be used with the responses from the **Authorisation Code Flow** and **Implicit Grant Flow** it also will read an existing **token** with the `LoadToken()` **method** and write a **token** with the `SaveToken()` **method**.

Step 5

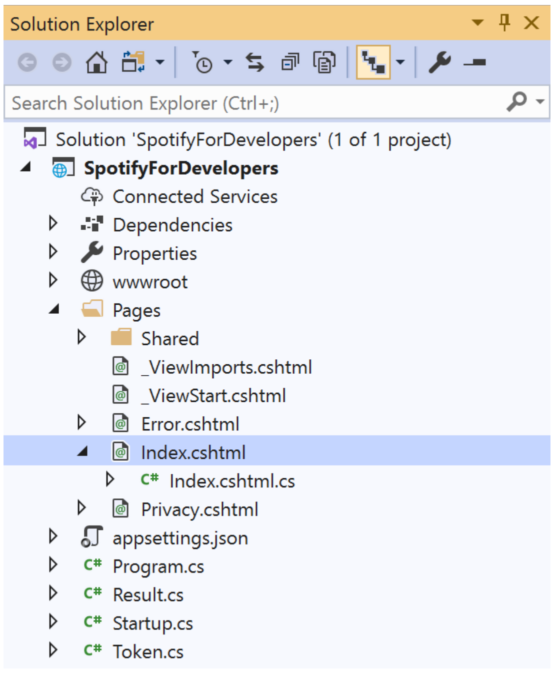
While still in the **Code View** for **Index.cshtml.cs** below the `public async Task<IActionResult> OnGetAsync(...) { ... }` **method** add the following **methods**:

```
public IActionResult OnPostAuthorisationCode() =>
    Redirect(Api.GetAuthorisationCodeAuthUri(
        RedirectUri, state, Scope.AllPermissions).ToString());

public IActionResult OnPostImplicitGrant() =>
    Redirect(Api.GetImplicitGrantAuthUri(
        RedirectUri, state, Scope.AllPermissions).ToString());
```

These **methods** will handle the requests for the **Authorisation Code Flow**, **Implicit Grant Flow** and the whole process for the **Client Credentials Flow**.

Step 6



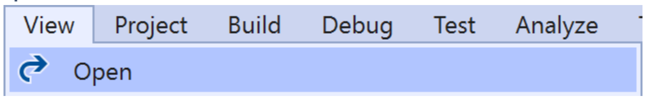
Solution Explorer

Search Solution Explorer (Ctrl+;)

```
Solution 'SpotifyForDevelopers' (1 of 1 project)
◢ SpotifyForDevelopers
    Connected Services
  ▷ Dependencies
  ▷ Properties
  ▷ wwwroot
  ◢ Pages
    ▷ Shared
      _ViewImports.cshtml
      _ViewStart.cshtml
    ▷ Error.cshtml
    ◢ Index.cshtml
      ▷ Index.cshtml.cs
    ▷ Privacy.cshtml
  ▷ appsettings.json
  ▷ Program.cs
  ▷ Result.cs
  ▷ Startup.cs
  ▷ Token.cs
```

In the **Solution Explorer** in the **Pages** section select **Index.cshtml**

Step 7



Then from the **Menu** choose **View** and then **Open**

Step 8

Once in the **Code View** for **Index.cshtml** above `<!-- Authorisation Guide -->` enter the following:

```
<h2 class="text-center">Authorisation</h2>
<ul class="list-group list-group-horizontal">
    <li class="list-group-item">
        <form asp-page-handler="authorisationcode" method="post">
            <button class="btn btn-success m-md-1">
                Authorisation Code Flow Login
            </button>
        </form>
    </li>
    <li class="list-group-item">
        <form asp-page-handler="implicitgrant" method="post">
            <button class="btn btn-success m-md-1">
                Implicit Grant Flow Login
            </button>
        </form>
    </li>
    <li class="list-group-item">
        <form asp-page-handler="clientcredentials" method="post">
            <button class="btn btn-success m-md-1">
                Client Credentials Flow Login
            </button>
        </form>
    </li>
</ul>
```

The `button` for each of these will trigger the various **Authorisation Flows** for **Authorisation Code**, **Implicit Grant** and **Client Credentials**.

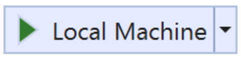
Step 9

While still in the **Code View** at the bottom of **Index.cshtml** enter the following `@section` and `script`:

```
@section Scripts
{
    <script language="javascript">
        $(window).ready(function () {
            var href = $(location).attr("href");
            if (href.indexOf('#') >= 0) {
                href = href.replace('#', '?');
                window.location.replace(href);
            }
        });
    </script>
}
```

The `script` will handle the response from the **Implicit Grant Response** which returns a **hash fragment** and turn this into a **query string** for use with the **web application**.

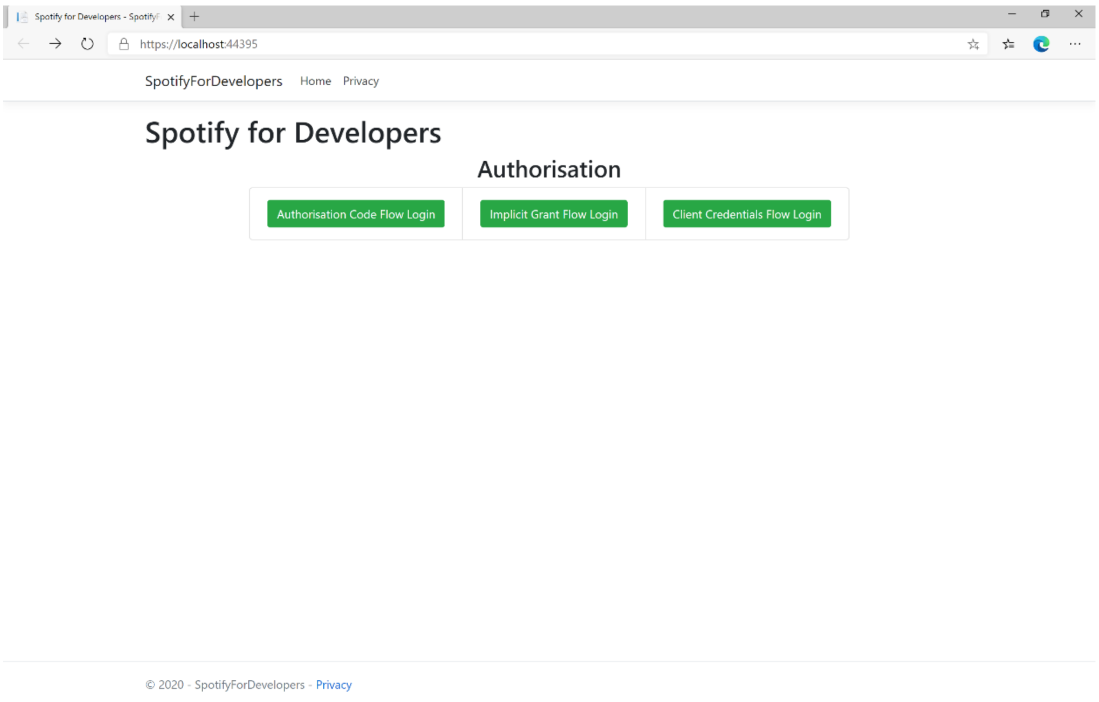
Step 10



Finally, in **Visual Studio 2019** select **IIS Application** to run the **Web Application**

Step 11

Once the **Web Application** is running you should see something like the following:


Spotify for Developers - Spotify
https://localhost:44395
SpotifyForDevelopers
Home
Privacy

Spotify for Developers

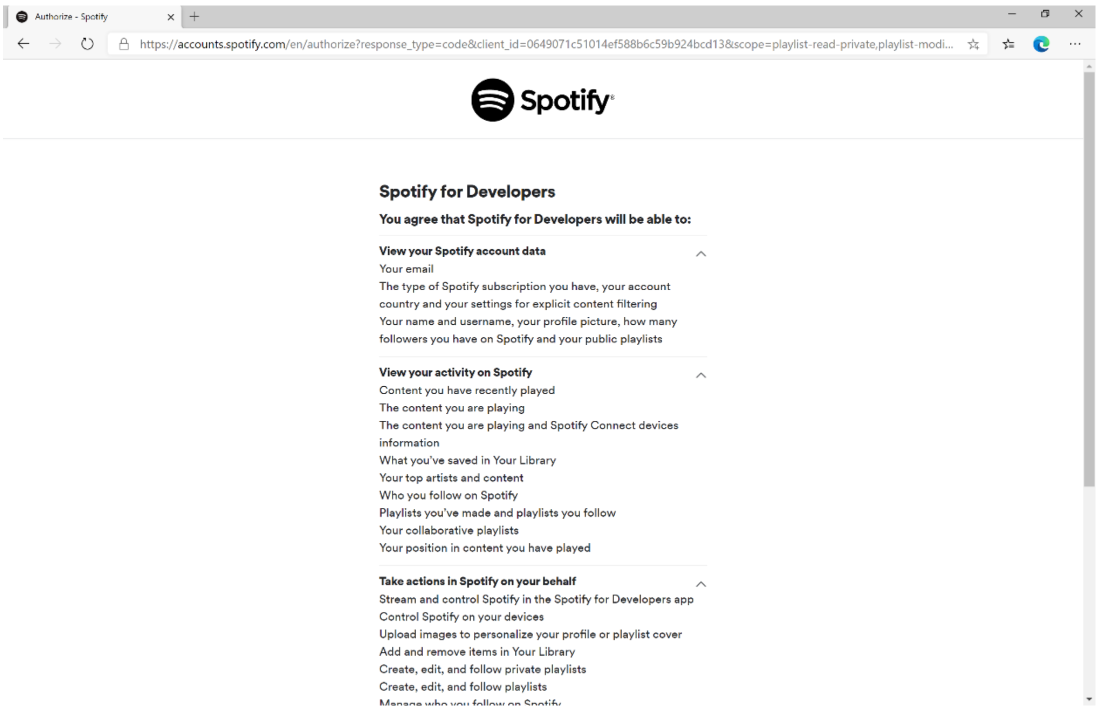
Authorisation

Authorisation Code Flow Login
Implicit Grant Flow Login
Client Credentials Flow Login

© 2020 - SpotifyForDevelopers - Privacy

Step 12

Then in the **Web Application** you can select the **Authorisation Code Flow Login** or the **Implicit Grant Flow Login** button then choose **Agree**:



Authorize - Spotify

https://accounts.spotify.com/en/authorize?response_type=code&client_id=0649071c51014ef588b6c59b924bcd13&scope=playlist-read-private,playlist-modi...

Spotify for Developers

You agree that Spotify for Developers will be able to:

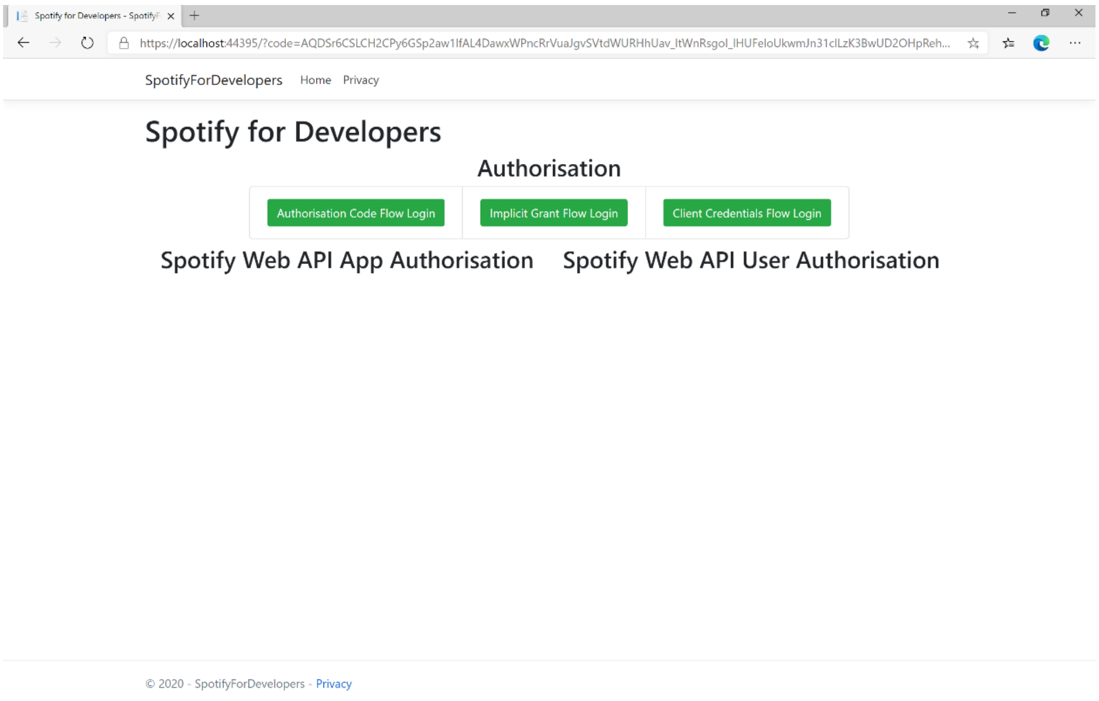
View your Spotify account data
Your email
The type of Spotify subscription you have, your account country and your settings for explicit content filtering
Your name and username, your profile picture, how many followers you have on Spotify and your public playlists

View your activity on Spotify
Content you have recently played
The content you are playing
The content you are playing and Spotify Connect devices information
What you've saved in Your Library
Your top artists and content
Who you follow on Spotify
Playlists you've made and playlists you follow
Your collaborative playlists
Your position in content you have played

Take actions in Spotify on your behalf
Stream and control Spotify in the Spotify for Developers app
Control Spotify on your devices
Upload images to personalize your profile or playlist cover
Add and remove items in Your Library
Create, edit, and follow private playlists
Create, edit, and follow playlists
Manage who you follow on Spotify

Step 13

Once agreed the **web application** if **Authorisation Code Flow Login** or **Implicit Grant Flow Login** was selected should appear like the following:



Spotify for Developers - SpotifyF

https://localhost:44395/?code=AQDSr6CSLCH2CPy6GSp2aw1IfAL4DawxWPncRrVuaJgvSVtdWURHhUav_ItWnRsgol_IHUFeloUkwmJn31clLzK3BwUD2OHpReh...

SpotifyForDevelopers Home Privacy

Spotify for Developers

Authorisation

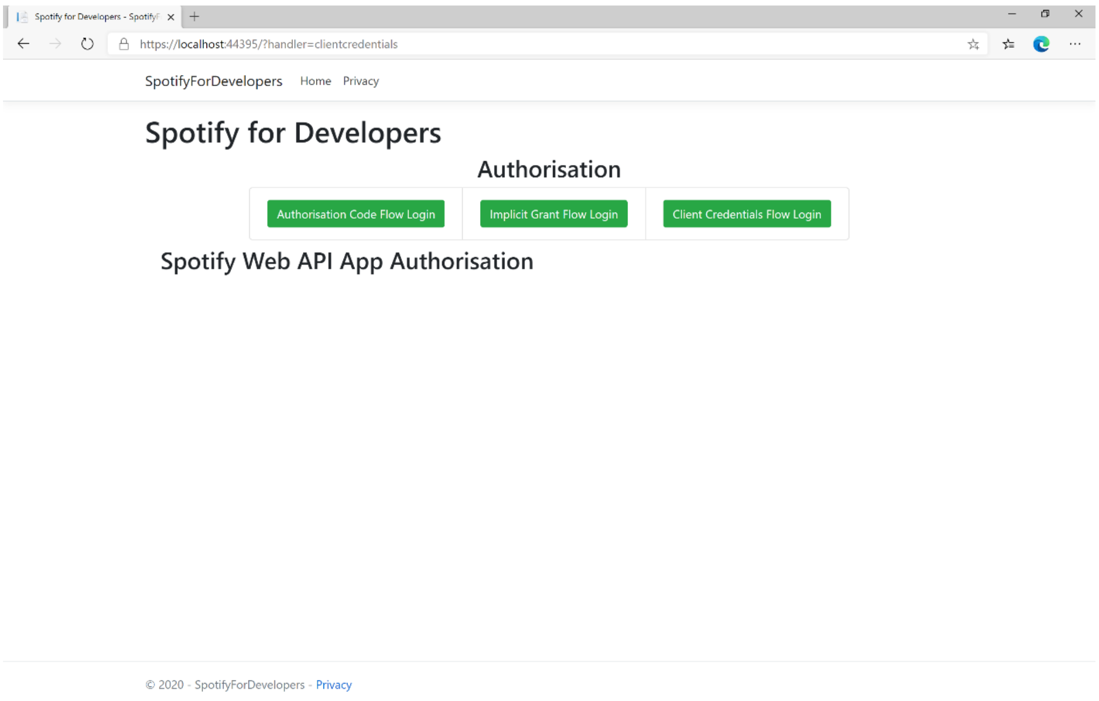
Authorisation Code Flow Login | Implicit Grant Flow Login | Client Credentials Flow Login

Spotify Web API App Authorisation   Spotify Web API User Authorisation

© 2020 - SpotifyForDevelopers - Privacy

Step 14

If the **Client Credentials Flow Login** was selected the **web application** should appear like the following:


Spotify for Developers - SpotifyF
https://localhost:44395/?handler=clientcredentials
SpotifyForDevelopers
Home
Privacy
Authorisation Code Flow Login
Implicit Grant Flow Login
Client Credentials Flow Login
Privacy

Spotify for Developers

Authorisation

Spotify Web API App Authorisation

© 2020 - SpotifyForDevelopers - Privacy

Step 15

You can stop the **web application** in **Visual Studio 2019** by selecting the **Stop debugging** button

Step 16

You can exit **Visual Studio 2019** by selecting the **Close** button in the top right of the **application** as that completes this part of the workshop