|  |
| --- |
|  |
| Spotify for Developers  Playlists |
| |  |  |  | | --- | --- | --- | |  |  |  | |

## Get a Playlist

Get a playlist owned by a Spotify user.

|  |  |
| --- | --- |
| https://api.spotify.com/v1/playlists/{playlist\_id} | |
| Header | |
| Authorization | Valid Access Token from Spotify Accounts service |
| Path Parameter | |
| playlist\_id | Spotify Id for the playlist |
| Query Parameter | |
| fields | Filters for the query: comma-separated list of fields to return e.g. description, uri |
| market | ISO 3166-1 alpha-2 country code e.g. “GB”. Provide to apply Track Relinking |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | Playlist Object |
| Error | |
| Error Code | Error Object |

### Step 1

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | In **Windows 10** choose **Start**, and then from the **Start Menu** find and select **Visual Studio 2019** |
|  | Once done, from the **Get started** screen for **Visual Studio 2019** select **Open a project or solution** |
| A screenshot of a social media post  Description automatically generated | 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**, **Authorisation Guide**, **Search & Browse**, **Playlists & Artists**, **Albums & Tracks**, **Episodes & Shows** and **Follow** |

### 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** below the **method** for public async Task<IActionResult> OnPostUnfollowPlaylistAsync() { ... } enter the following **method**:

|  |
| --- |
| public async Task<IActionResult> OnPostGetPlaylistAsync(string value)  {  LoadToken();  var result = await Api.GetPlaylistAsync(value);  if (result != null)  {  Results = new List<Result> { new Result()  {  Id = result.Id,  Name = result.Name,  Image = result?.Images?.FirstOrDefault()?.Url  }};  }  return Page();  } |

The **method** for OnPostGetPlaylistAsync is used to get a **playlist** by **Playlist Id** on Spotify with the value and populate the **property** for Results accordingly.

### Step 5

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

### Step 6

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

### Step 7

Once in the **Code View** for **Index.cshtml** above <!-- Spotify Web API User Authorisation --> enter the following:

|  |
| --- |
| <ul class="list-group mb-2">  <li class="list-group-item list-group-item-primary">  <h5 class="list-group-item-heading">Playlists</h5>  </li>  <li class="list-group-item">  <form **asp-page-handler**="GetPlaylist" method="post">  <input **asp-for**="Value" placeholder="Playlist Id" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Get a Playlist  </button>  </form>  </li>  <!-- Playlists -->  </ul> |

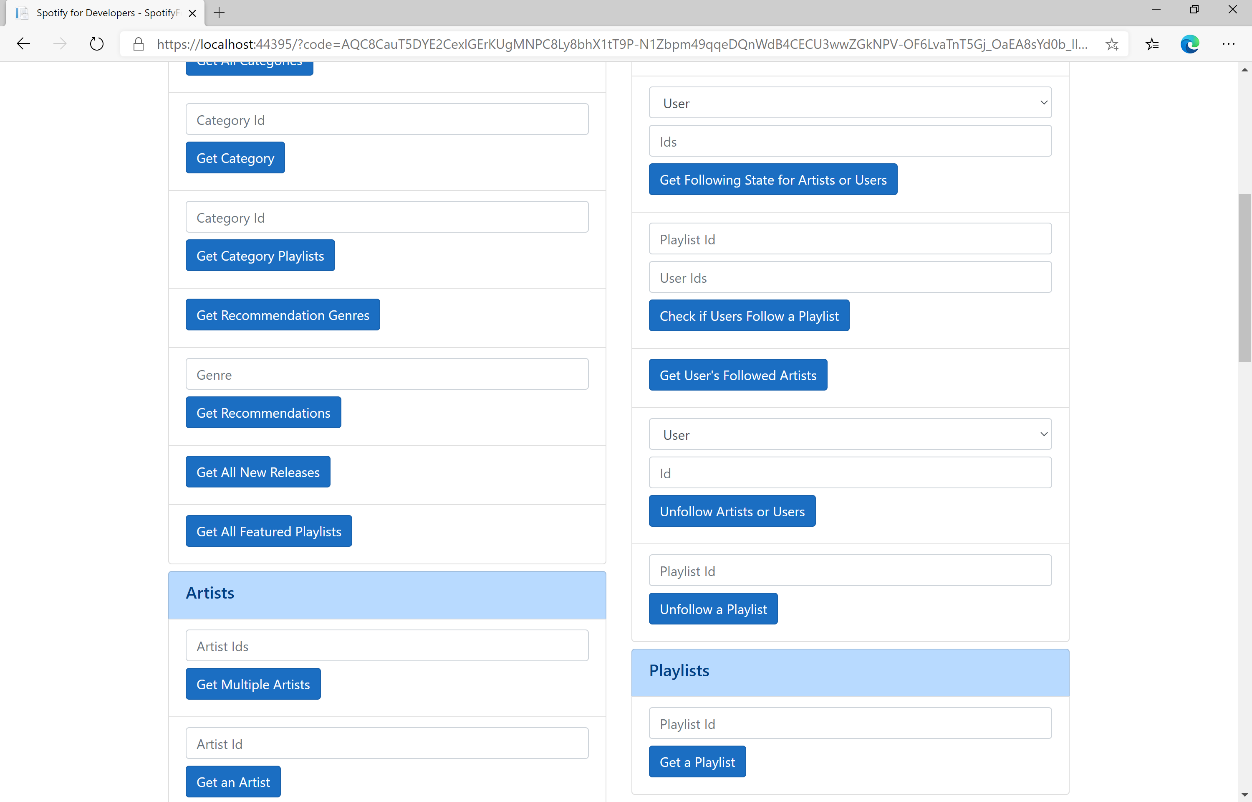
This form will **post** to the **method** for OnPostGetPlaylistAsync with the Value as the **Playlist Id** and will output to the **Results**. This form could also be used with **Spotify Web API App Authorisation** as it does not require a **User Token**

### Step 8

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

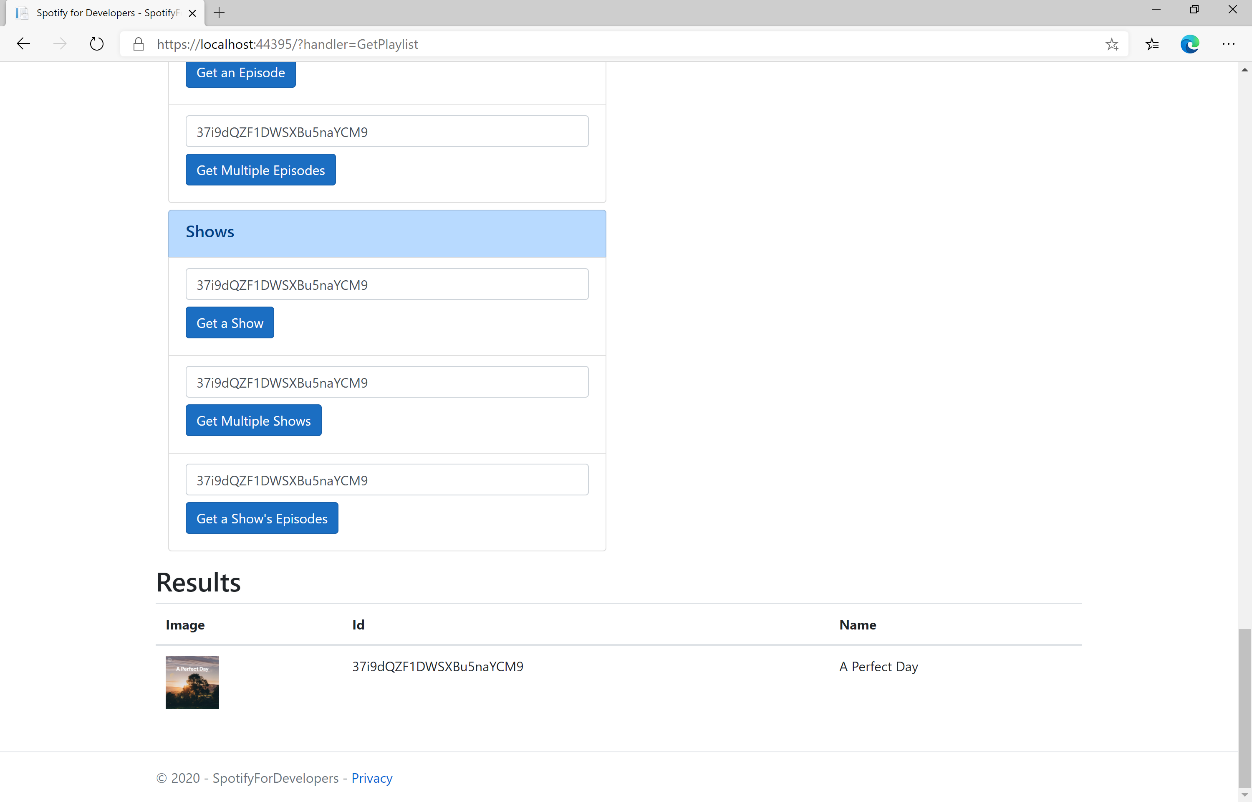
### Step 9

Once the **Web Application** is running and you select **Authorisation Code Flow Login** or **Implicit Grant Flow Login** and scroll down you should see something like the following:



### Step 10

You can then enter a **Playlist Id** from **Get All Featured Playlists** and select **Get a Playlist** and scroll down to view **Results** like the following:



### Step 11

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

### Step 12

|  |  |
| --- | --- |
|  | You can choose to exit **Visual Studio 2019** by selecting the **Close** button in the top right of the **application** as that completes thispart of the workshop |

## Get a Playlist’s Tracks

Get full details of the tracks of a playlist owned by a Spotify user.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/playlists/{playlist\_id}/tracks | |
| Header | |
| Authorization | Valid Access Token from Spotify Accounts service |
| Path Parameter | |
| playlist\_id | Spotify Id for the playlist |
| Query Parameter | |
| fields | Filters for the query: comma-separated list of fields to return e.g. total, limit |
| limit | Maximum number of results to return |
| offset | Index of first result to return |
| market | ISO 3166-1 alpha-2 country code e.g. “GB”. Provide to apply Track Relinking |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | Array of Playlist Track Objects wrapped in a Paging Object |
| Error | |
| Error Code | Error Object |

### Step 1

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | If you chose to close **Visual Studio 2019** previously, in **Windows 10** choose **Start**, and then from the **Start Menu** find and select **Visual Studio 2019** |
|  | Once done, from the **Get started** screen for **Visual Studio 2019** select **Open a project or solution** |
| A screenshot of a social media post  Description automatically generated | 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**, **Authorisation Guide**, **Search & Browse**, **Playlists & Artists**, **Albums & Tracks**, **Episodes & Shows**, **Follow** and **Playlists** |

### 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** below the **method** for public async Task<IActionResult> OnPostGetPlaylistAsync(...) { ... } enter the following **method**:

|  |
| --- |
| public async Task<IActionResult> OnPostGetPlaylistTracksAsync(string value)  {  LoadToken();  var results = await Api.GetPlaylistTracksAsync(value);  if (results?.Items != null)  {  Results = results.Items.Select(result => new Result()  {  Id = result?.Track.Id,  Name = result?.Track.Name,  Image = result?.Track.Album?.Images?.FirstOrDefault()?.Url,  Inner = new Result()  {  Id = result?.Track?.Artists?.FirstOrDefault()?.Id,  Name = result?.Track?.Artists?.FirstOrDefault()?.Name  }  });  }  return Page();  } |

The **method** for OnPostGetPlaylistTracksAsync is used to get the **tracks** for a **playlist** by **Playlist Id** on Spotify with the value and populate the **property** for Results accordingly.

### Step 5

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

### Step 6

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

### Step 7

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

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="GetPlaylistTracks" method="post">  <input **asp-for**="Value" placeholder="Playlist Id" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Get a Playlist's Tracks  </button>  </form>  </li> |

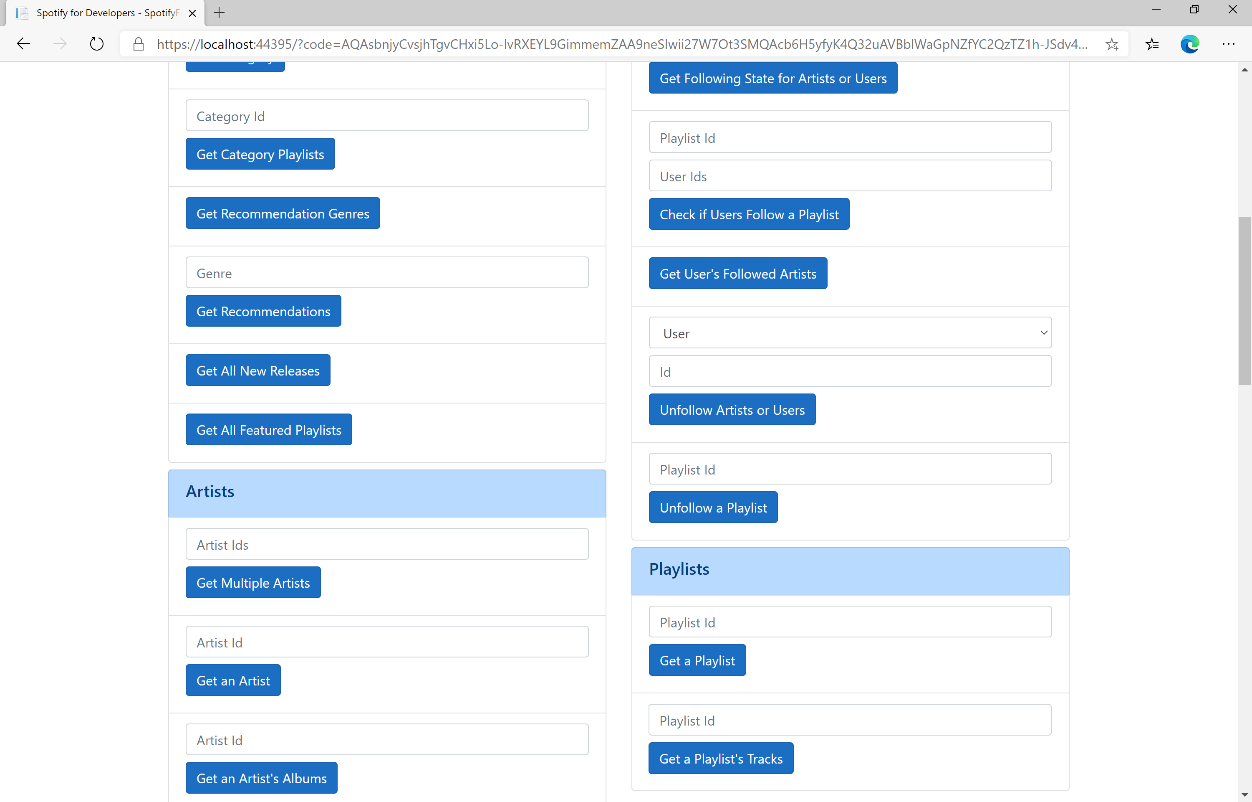
This form will **post** to the **method** for OnPostGetPlaylistTracksAsync with the Value as the **Playlist Id** and will output to the **Results**. This form could also be used with **Spotify Web API App Authorisation** as it does not require a **User Token**.

### Step 8

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

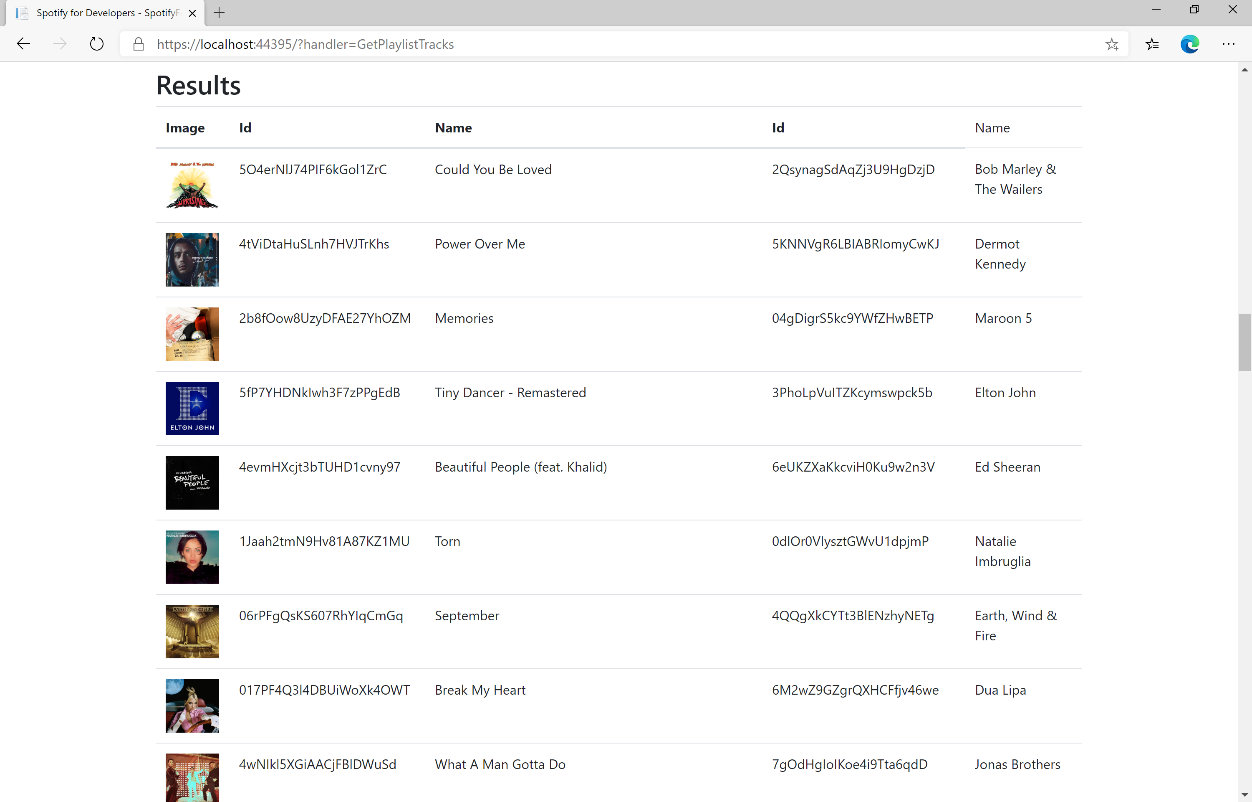
### Step 9

Once the **Web Application** is running and you select **Authorisation Code Flow Login** or **Implicit Grant Flow Login** and scroll down you should see something like the following:



### Step 10

You can then enter a **Playlist Id** from **Get All Featured Playlists** and select **Get Playlist Tracks** and scroll down to view **Results** like the following:



### Step 11

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

### Step 12

|  |  |
| --- | --- |
|  | You can choose to exit **Visual Studio 2019** by selecting the **Close** button in the top right of the **application** as that completes thispart of the workshop |

## Create a Playlist

Create a playlist for a Spotify user

|  |  |
| --- | --- |
| POST https://api.spotify.com/v1/users/{user\_id}/playlists | |
| Header | |
| Authorization | User Token from Spotify Accounts service with **playlist-modify-public** scope for publicly followed playlist and **playlist-modify-private** for a privately followed playlist |
| Path Parameter | |
| user\_id | User’s Spotify User Id |
| Body Parameter | |
| name | Name for the new playlist |
| public | If true, the playlist will be public, if false it will be private |
| collaborative | If true, the playlist will be collaborative and other users will be able to modify the playlist |
| description | Playlist description |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK |  |
| Http Status 201  Created |  |
| Error | |
| HTTP Status 403  Forbidden | Returned when not Authorised |
| Error Code | Error Object |

### Step 1

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | If you chose to close **Visual Studio 2019** previously, in **Windows 10** choose **Start**, and then from the **Start Menu** find and select **Visual Studio 2019** |
|  | Once done, from the **Get started** screen for **Visual Studio 2019** select **Open a project or solution** |
| A screenshot of a social media post  Description automatically generated | 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**, **Authorisation Guide**, **Search & Browse**, **Playlists & Artists**, **Albums & Tracks**, **Episodes & Shows**, **Follow** and **Playlists** |

### 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** below the **method** for public async Task<IActionResult> OnPostGetPlaylistTracksAsync(...) { ... } enter the following **method**:

|  |
| --- |
| public async Task<IActionResult> OnPostCreatePlaylistAsync(string value, string option)  {  LoadToken();  var result = await Api.CreatePlaylistAsync(value, option);  if (result != null)  {  Results = new List<Result> { new Result()  {  Id = result.Id,  Name = result.Name  }};  }  return Page();  } |

The **method** for OnPostCreatePlaylistAsync is used to **create** a **playlist** for a **user** by **User Id** with the value and option with a given **Name** on Spotify and populate the **property** for Results accordingly.

### Step 5

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

### Step 6

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

### Step 7

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

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="CreatePlaylist" method="post">  <input **asp-for**="Value" placeholder="User Id" class="form-control mb-2" />  <input **asp-for**="Option" placeholder="Name" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Create a Playlist  </button>  </form>  </li> |

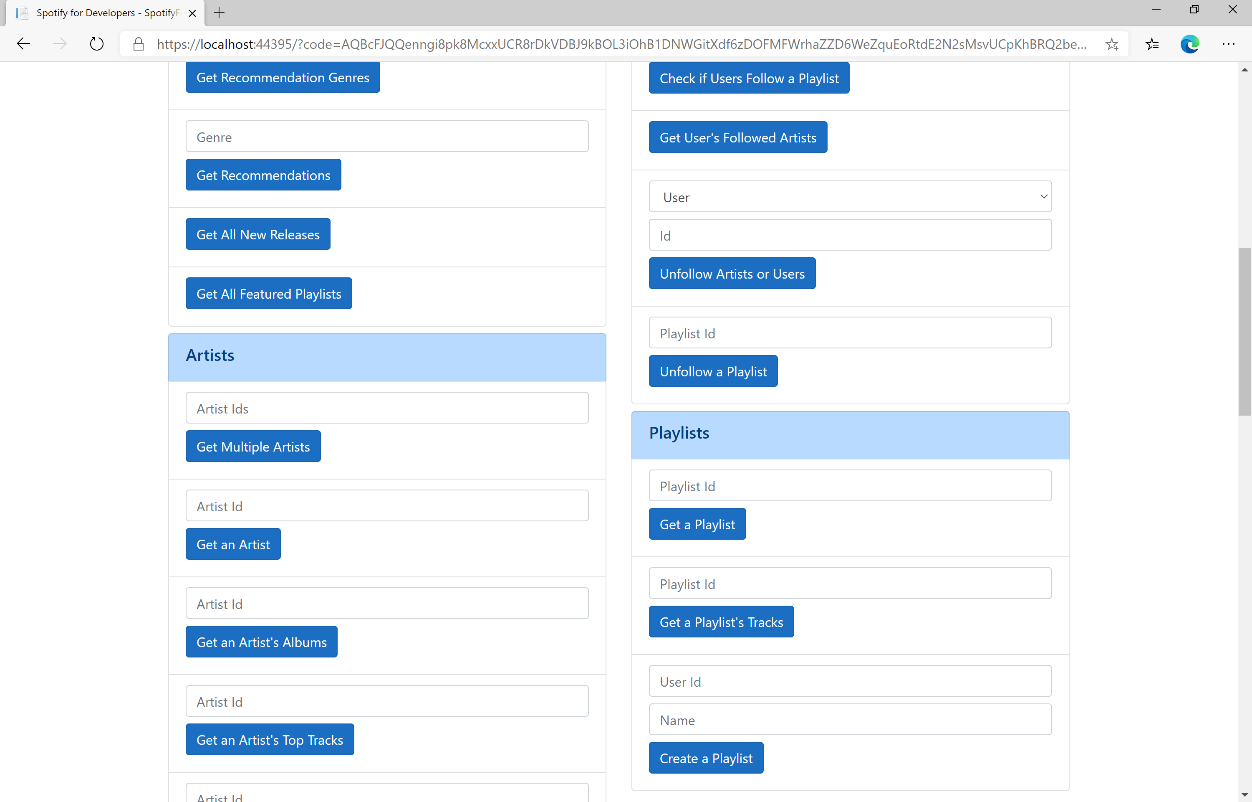
This form will **post** to the **method** for CreatePlaylist with the Value of the **User Id** and the Option of the **Name** and will output to the **Results**.

### Step 8

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

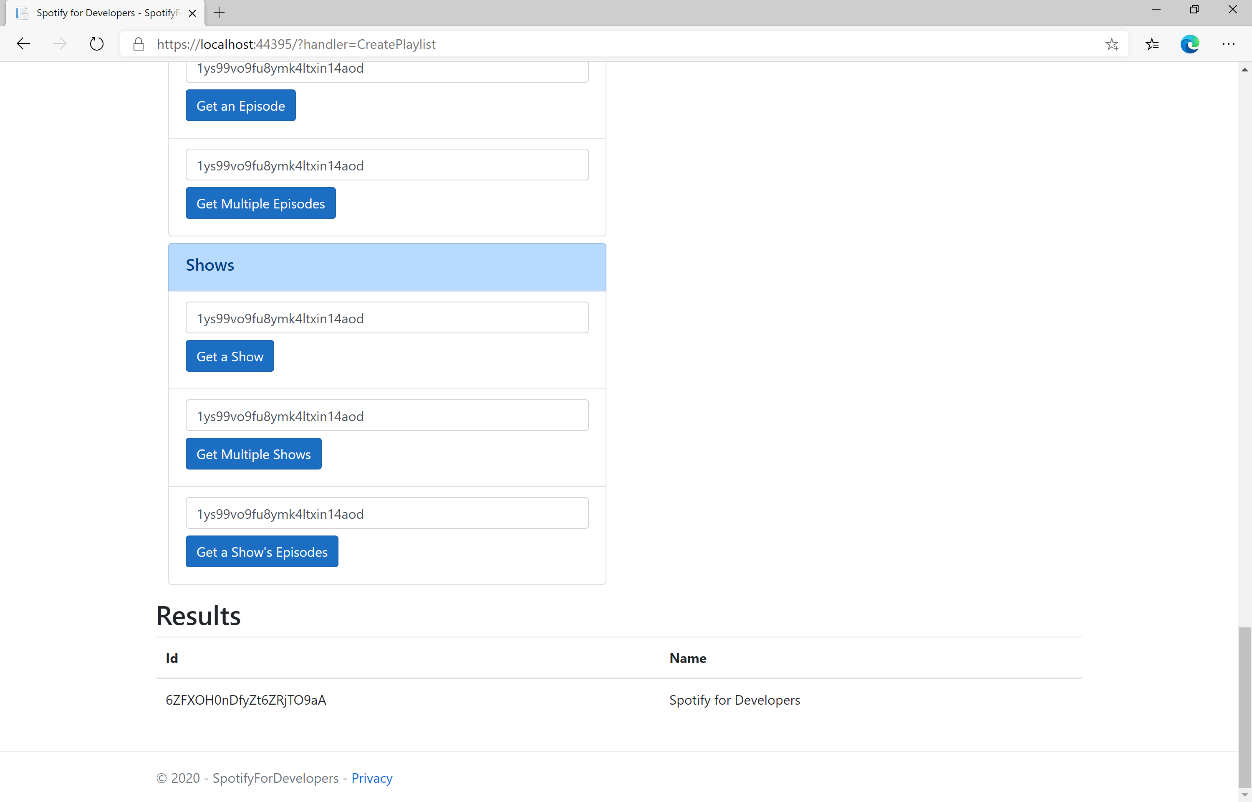
### Step 9

Once the **Web Application** is running and you select **Authorisation Code Flow Login** or **Implicit Grant Flow Login** and scroll down you should see something like the following:



### Step 10

You can then enter above **Create a Playlist** your **User Id**, to get this in **Spotify** select your **Username** then **...** and **Copy Spotify URI** your **User Id** will be after **spotify:user:**, and a **Name** then select **Create a Playlist** and scroll down to view **Results** like the following and copy the **Id** for use later:



### Step 11

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

### Step 12

|  |  |
| --- | --- |
|  | You can choose to exit **Visual Studio 2019** by selecting the **Close** button in the top right of the **application** as that completes thispart of the workshop |

## Get a List of Current User's Playlists

Get a list of the playlists owned or followed by the current Spotify user.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/me/playlists | |
| Header | |
| Authorization | User Token from Spotify Accounts service with **playlist-read-private** scope for playlists for the current user and **playlist-read-collaborative** for collaborative playlists |
| Query Parameter | |
| limit | Maximum number of playlists to return. Default: 20. Minimum: 1. Maximum: 50 |
| offset | Index of the first playlist to return. Default: 0 (the first object). Maximum offset: 100,000. Use with limit to get the next set of playlists. |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | Array of Simplified Playlist Object wrapped in a Paging Object |
| Error | |
| Error Code | Error Object |

### Step 1

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | If you chose to close **Visual Studio 2019** previously, in **Windows 10** choose **Start**, and then from the **Start Menu** find and select **Visual Studio 2019** |
|  | Once done, from the **Get started** screen for **Visual Studio 2019** select **Open a project or solution** |
| A screenshot of a social media post  Description automatically generated | 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**, **Authorisation Guide**, **Search & Browse**, **Playlists & Artists**, **Albums & Tracks**, **Episodes & Shows**, **Follow** and **Playlists** |

### 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** below the **method** for public async Task<IActionResult> OnPostCreatePlaylistAsync(...) { ... } enter the following **method**:

|  |
| --- |
| public async Task<IActionResult> OnPostGetCurrentUserPlaylistsAsync()  {  LoadToken();  var results = await Api.GetUserPlaylistsAsync();  if (results?.Items != null)  {  Results = results.Items.Select(result => new Result()  {  Id = result.Id,  Name = result.Name,  Image = result?.Images?.FirstOrDefault()?.Url  });  }  return Page();  } |

The **method** for OnPostGetCurrentUserPlaylistsAsync is used to list the **playlists** for the logged in **user** on Spotify and populate the **property** for Results accordingly.

### Step 5

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

### Step 6

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

### Step 7

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

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="GetCurrentUserPlaylists" method="post">  <button class="btn btn-primary mb-2">  Get a List of Current User's Playlists  </button>  </form>  </li> |

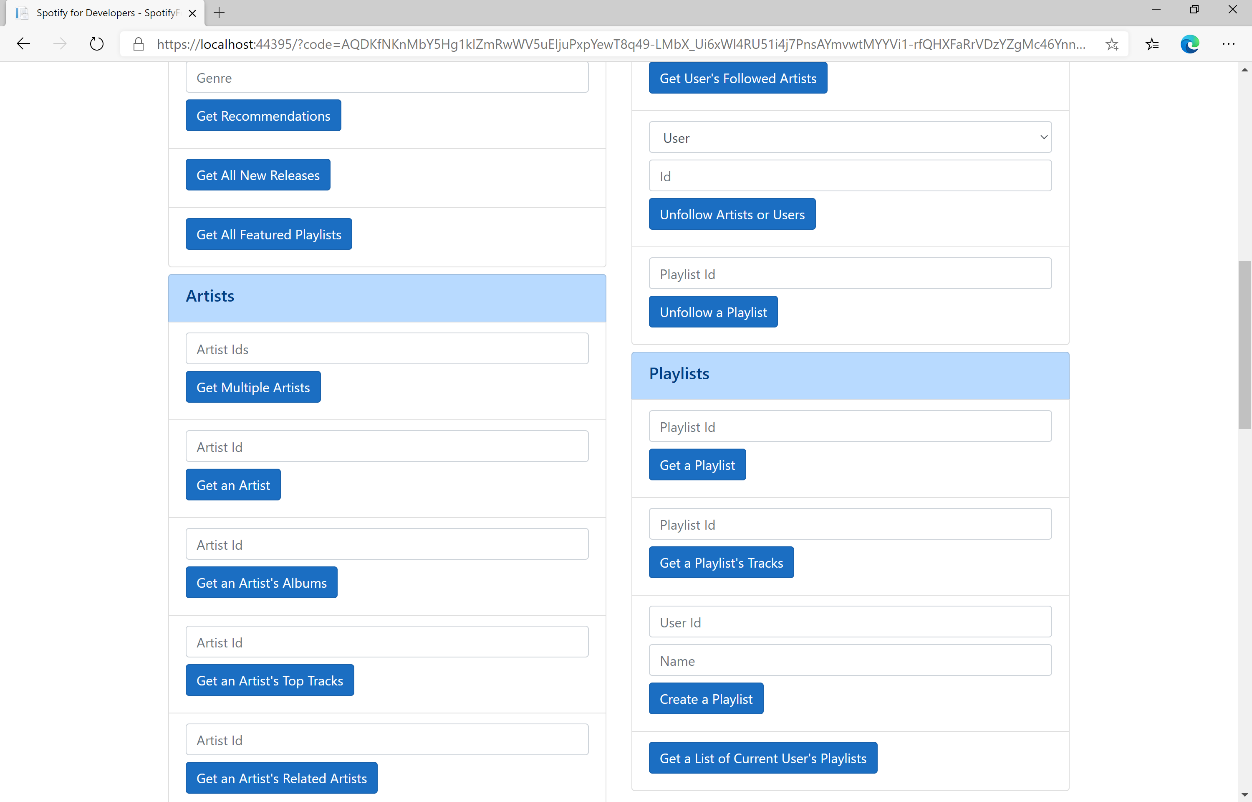
This form will **post** to the **method** for GetCurrentUserPlaylists and will output to the **Results**.

### Step 8

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

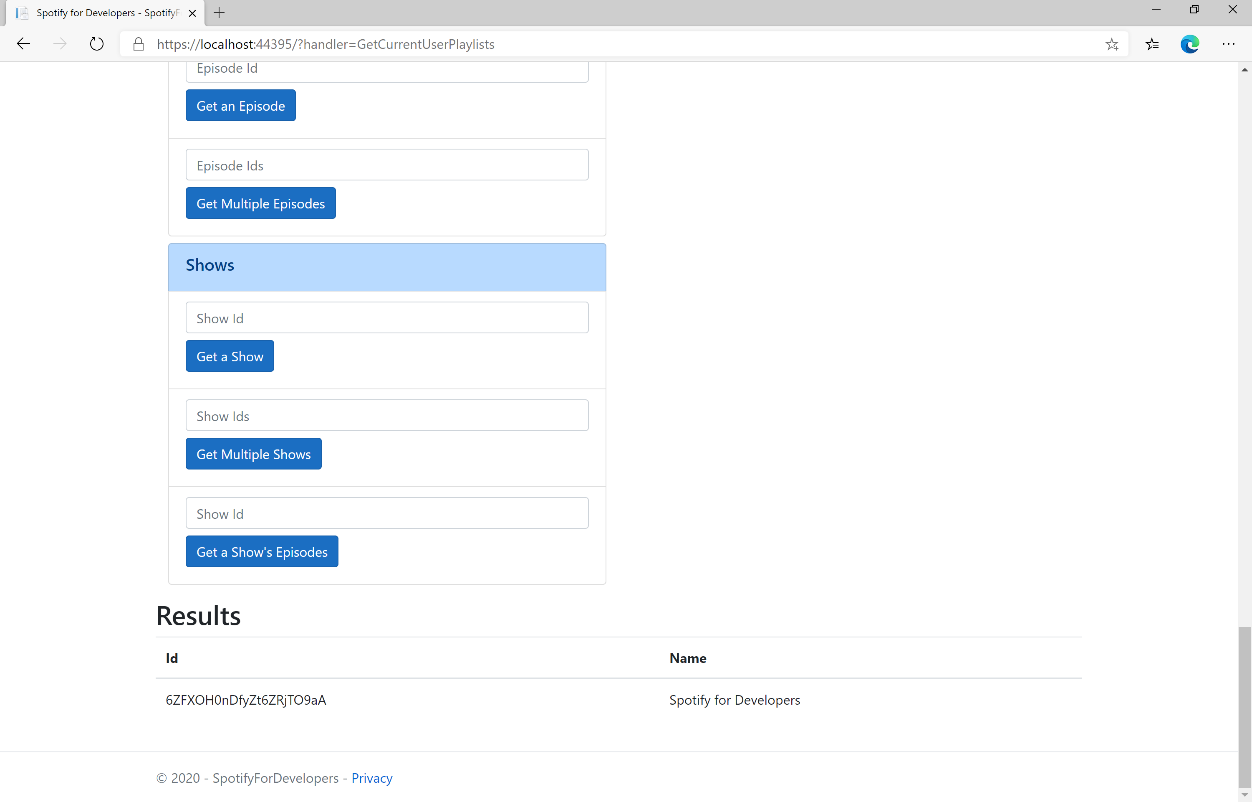
### Step 9

Once the **Web Application** is running and you select **Authorisation Code Flow Login** or **Implicit Grant Flow Login** and scroll down you should see something like the following:



### Step 10

You can then select **Get a List of Current User's Playlists** and scroll down to view **Results** like the following:



### Step 11

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

### Step 12

|  |  |
| --- | --- |
|  | You can choose to exit **Visual Studio 2019** by selecting the **Close** button in the top right of the **application** as that completes thispart of the workshop |

## Get a List of a User's Playlists

Get a list of the playlists owned or followed by a Spotify user.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/users/{user\_id}/playlists | |
| Header | |
| Authorization | User Token from Spotify Accounts service with **playlist-read-private** scope for playlists for the current user and **playlist-read-collaborative** for collaborative playlists |
| Path Parameter | |
| user\_id | User’s Spotify User Id |
| Query Parameter | |
| limit | Maximum number of playlists to return. Default: 20. Minimum: 1. Maximum: 50 |
| offset | Index of the first playlist to return. Default: 0 (the first object). Maximum offset: 100,000. Use with limit to get the next set of playlists. |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | Array of Simplified Playlist Object wrapped in a Paging Object |
| Error | |
| Error Code | Error Object |

### Step 1

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | If you chose to close **Visual Studio 2019** previously, in **Windows 10** choose **Start**, and then from the **Start Menu** find and select **Visual Studio 2019** |
|  | Once done, from the **Get started** screen for **Visual Studio 2019** select **Open a project or solution** |
| A screenshot of a social media post  Description automatically generated | 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**, **Authorisation Guide**, **Search & Browse**, **Playlists & Artists**, **Albums & Tracks**, **Episodes & Shows**, **Follow** and **Playlists** |

### 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** below the **method** for public async Task<IActionResult> OnPostGetCurrentUserPlaylistsAsync() { ... } enter the following **method**:

|  |
| --- |
| public async Task<IActionResult> OnPostGetUserPlaylistsAsync(string value)  {  LoadToken();  var results = await Api.GetUserPlaylistsAsync(value);  if (results?.Items != null)  {  Results = results.Items.Select(result => new Result()  {  Id = result.Id,  Name = result.Name,  Image = result?.Images?.FirstOrDefault()?.Url  });  }  return Page();  } |

The **method** for OnPostGetUserPlaylistsAsync is used to list the **playlists** for a **user** by **User Id** on Spotify and populate the **property** for Results accordingly.

### Step 5

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

### Step 6

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

### Step 7

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

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="GetUserPlaylists" method="post">  <input **asp-for**="Value" placeholder="User Id" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Get a List of a User's Playlists  </button>  </form>  </li> |

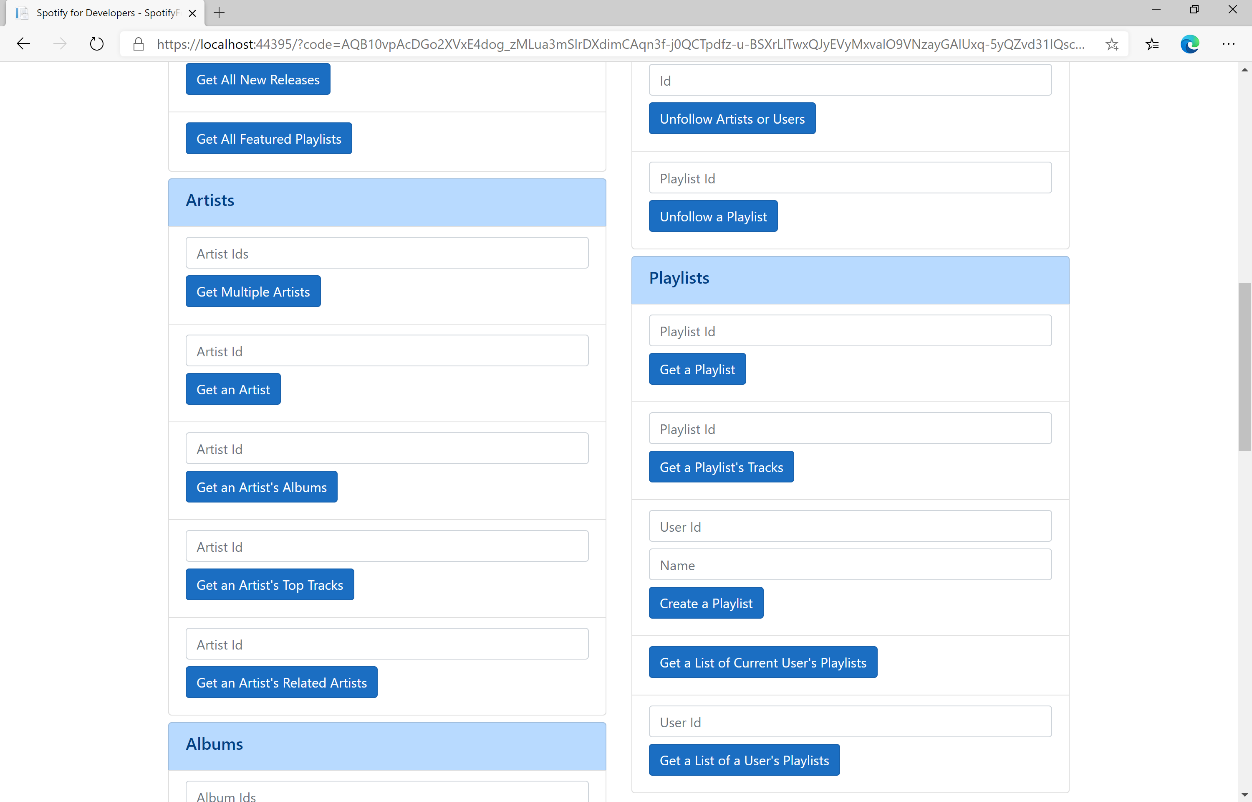
This form will **post** to the **method** for GetUserPlaylists with the Value of the **User Id** and will output to the **Results**.

### Step 8

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

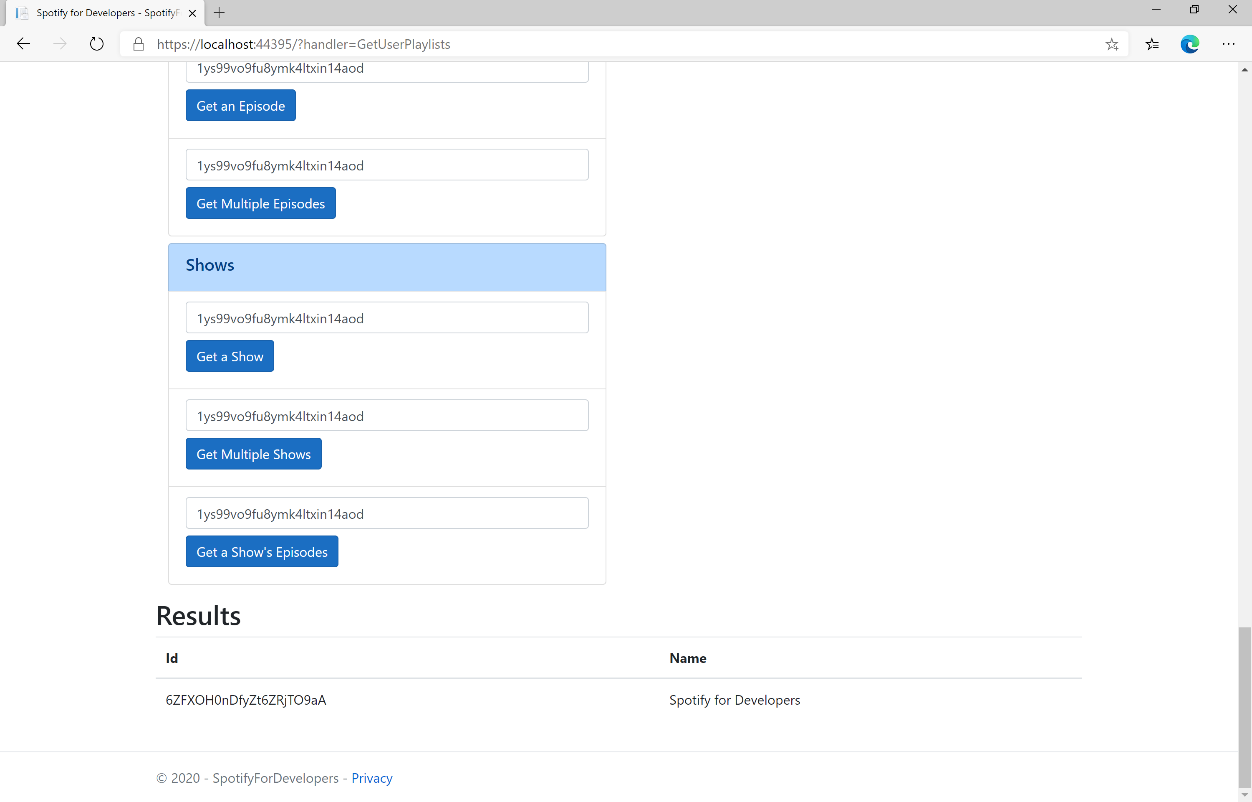
### Step 9

Once the **Web Application** is running and you select **Authorisation Code Flow Login** or **Implicit Grant Flow Login** and scroll down you should see something like the following:



### Step 10

You can then above **Get a List of a User's Playlists** enter your own **User Id**, to get this in **Spotify** select your **Username** then **...** and **Copy Spotify URI** your **User Id** will be after **spotify:user**:, and then select **Get a List of a User's Playlists** and scroll down to view **Results** like the following:



### Step 11

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

### Step 12

|  |  |
| --- | --- |
|  | You can choose to exit **Visual Studio 2019** by selecting the **Close** button in the top right of the **application** as that completes thispart of the workshop |

## Add Tracks to a Playlist

Add one or more tracks to a user’s playlist.

|  |  |
| --- | --- |
| POST https://api.spotify.com/v1/playlists/{playlist\_id}/tracks | |
| Header | |
| Authorization | User Token from Spotify Accounts service with **playlist-modify-public** scope for publicly followed playlist and **playlist-modify-private** for a privately followed playlist |
| Path Parameter | |
| playlist\_id | Spotify Id for the Playlist |
| Query Parameter | |
| uri | Comma-separated list of Spotify Track URIs to add – maximum 100 |
| position | Position to insert the tracks, a zero-based index. |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 201 Created | snapshot\_id |
| Error | |
| HTTP Status 403  Forbidden | Returned when not Authorised or more than 10,000 tracks in Playlist |
| Error Code | Error Object |

### Step 1

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | If you chose to close **Visual Studio 2019** previously, in **Windows 10** choose **Start**, and then from the **Start Menu** find and select **Visual Studio 2019** |
|  | Once done, from the **Get started** screen for **Visual Studio 2019** select **Open a project or solution** |
| A screenshot of a social media post  Description automatically generated | 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**, **Authorisation Guide**, **Search & Browse**, **Playlists & Artists**, **Albums & Tracks**, **Episodes & Shows**, **Follow** and **Playlists** |

### 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** below the **method** for public async Task<IActionResult> OnPostGetUserPlaylistsAsync(...) { ... } enter the following **method**:

|  |
| --- |
| public async Task<IActionResult> OnPostAddTracksToPlaylistAsync(string value, string option)  {  LoadToken();  var values = value.Split(",").Select(value => $"spotify:track:{value}").ToList();  var result = await Api.AddTracksToPlaylistAsync(option, values);  if (result != null)  {  Results = new List<Result>() { new Result()  {  Id = result.SnapshotId,  Name = result.Success.ToString()  }};  }  return Page();  } |

The **method** for OnPostAddTracksToPlaylistAsync is used add **tracks** by **Track Ids** to a **playlist** by **Playlist Id** on Spotify and populate the **property** for Results accordingly.

### Step 5

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

### Step 6

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

### Step 7

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

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="AddTracksToPlaylist" method="post">  <input **asp-for**="Value" placeholder="Track Ids" class="form-control mb-2" />  <input **asp-for**="Option" placeholder="Playlist Id" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Add Tracks to a Playlist  </button>  </form>  </li> |

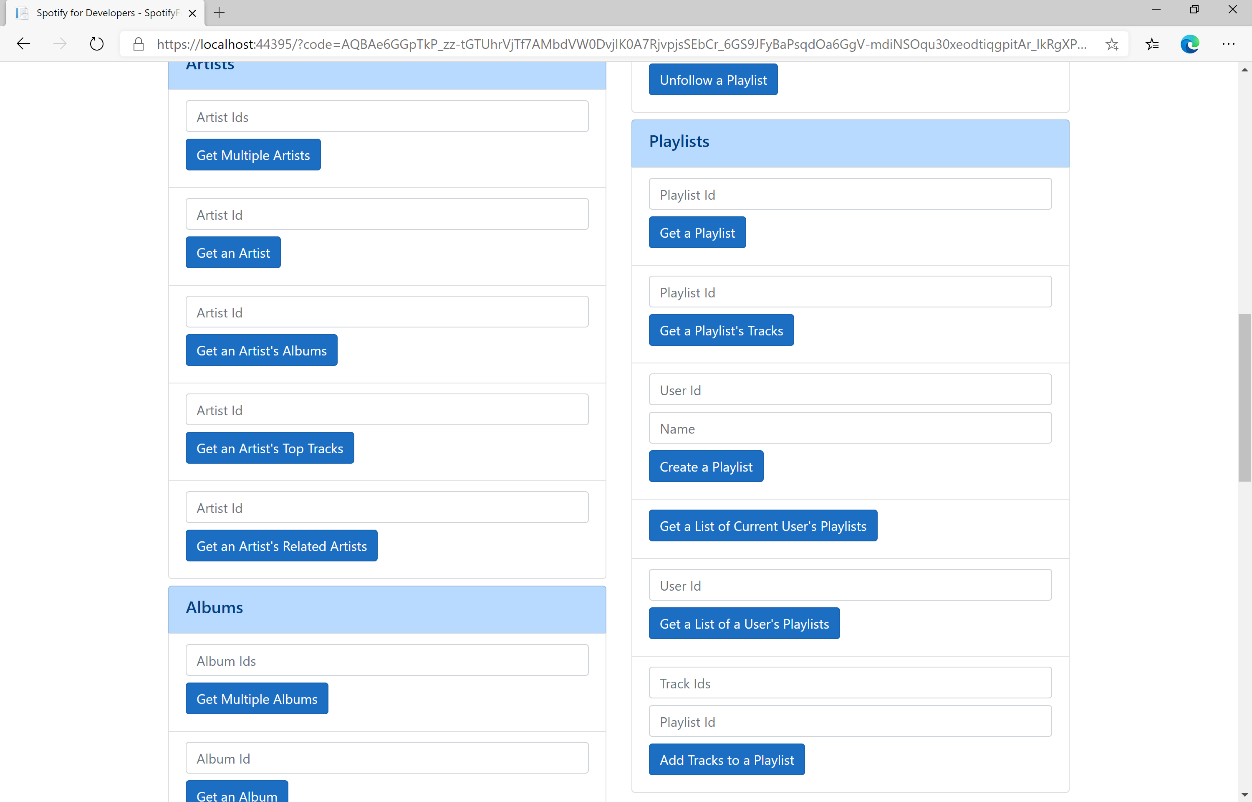
This form will **post** to the **method** for AddTracksToPlaylist with the Value of the **Track Ids** and the Option of the **Playlist Id** and will output to the **Results**.

### Step 8

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

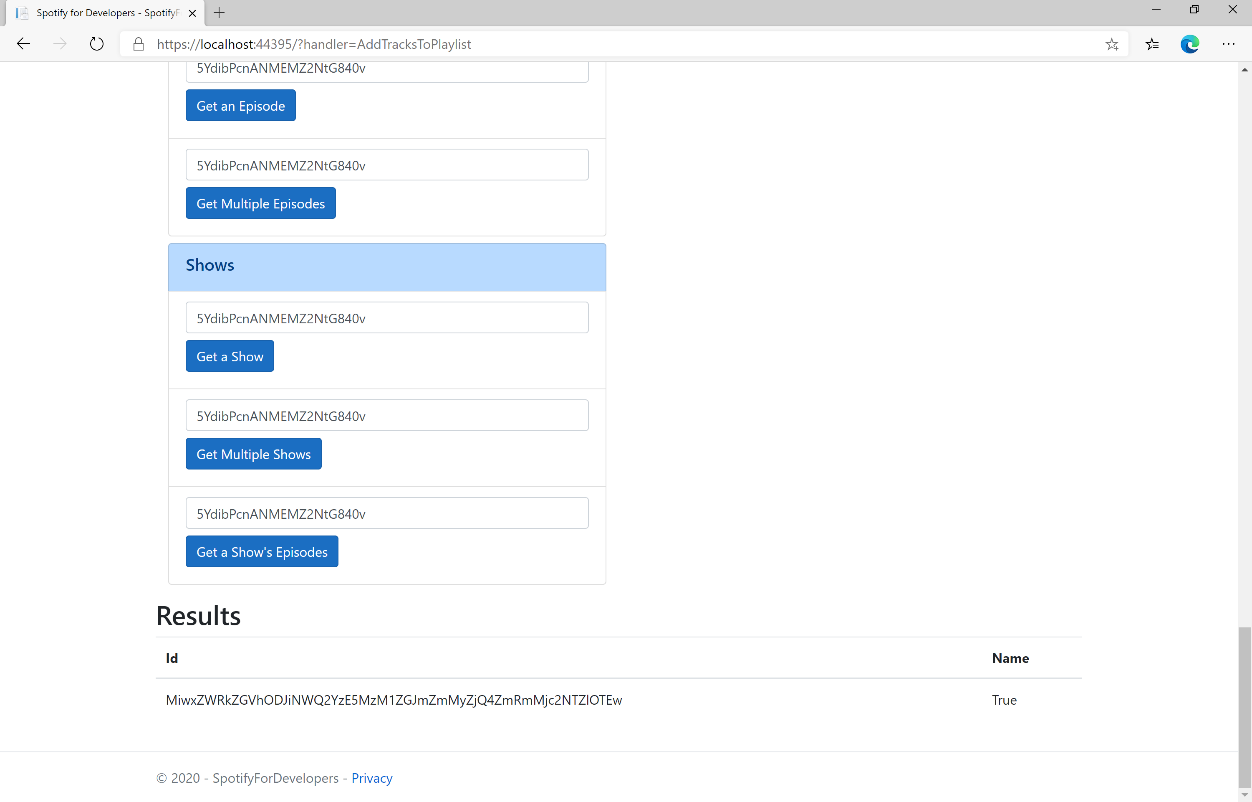
### Step 9

Once the **Web Application** is running and you select **Authorisation Code Flow Login** or **Implicit Grant Flow Login** and scroll down you should see something like the following:



### Step 10

You can then enter some **Track Ids** from an **Album Id** from **Get All New Releases** using **Get an Album's Tracks** into **Track Ids** above **Add Tracks to a Playlist** and the **Playlist Id** from **Create a Playlist** then select **Add Tracks to a Playlist** and scroll down to view **Results** like the following:



### Step 11

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

### Step 12

|  |  |
| --- | --- |
|  | You can choose to exit **Visual Studio 2019** by selecting the **Close** button in the top right of the **application** as that completes thispart of the workshop |

## Remove Tracks from a Playlist

Remove one or more tracks from a user’s playlist.

|  |  |
| --- | --- |
| DELETE https://api.spotify.com/v1/playlists/{playlist\_id}/tracks | |
| Header | |
| Authorization | User Token from Spotify Accounts service with **playlist-modify-public** scope for publicly followed playlist and **playlist-modify-private** for a privately followed playlist |
| Path Parameter | |
| playlist\_id | Spotify Id for the Playlist |
| Query Parameter | |
| tracks | Comma-separated list of Spotify Track URIs to add – maximum 100 |
| snapshot\_id | Playlist’s snapshot Id against which you want to make the changes |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | snapshot\_id |
| Error | |
| HTTP Status 403  Forbidden | Returned when not Authorised |
| HTTP Status 400  Bad Request | Client Errors including specified invalid positions |
| Error Code | Error Object |

### Step 1

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | If you chose to close **Visual Studio 2019** previously, in **Windows 10** choose **Start**, and then from the **Start Menu** find and select **Visual Studio 2019** |
|  | Once done, from the **Get started** screen for **Visual Studio 2019** select **Open a project or solution** |
| A screenshot of a social media post  Description automatically generated | 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**, **Authorisation Guide**, **Search & Browse**, **Playlists & Artists**, **Albums & Tracks**, **Episodes & Shows**, **Follow** and **Playlists** |

### 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** below the **method** for public async Task<IActionResult> OnPostAddTracksToPlaylistAsync(...) { ... } enter the following **method**:

|  |
| --- |
| public async Task<IActionResult> OnPostRemoveTracksFromPlaylistAsync(string value, string option)  {  LoadToken();  var values = value.Split(",").Select(value => $"spotify:track:{value}").ToList();  var result = await Api.RemoveTracksFromPlaylistAsync(option, values);  if (result != null)  {  Results = new List<Result>() { new Result()  {  Id = result.SnapshotId,  Name = result.Success.ToString()  }};  }  return Page();  } |

The **method** for OnPostRemoveTracksFromPlaylistAsync is used remove **tracks** by **Track Ids** from a **playlist** by **Playlist Id** on Spotify and populate the **property** for Results of the **success** of the operation accordingly.

### Step 5

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

### Step 6

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

### Step 7

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

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="RemoveTracksFromPlaylist" method="post">  <input **asp-for**="Value" placeholder="Track Ids" class="form-control mb-2" />  <input **asp-for**="Option" placeholder="Playlist Id" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Remove Tracks from a Playlist  </button>  </form>  </li> |

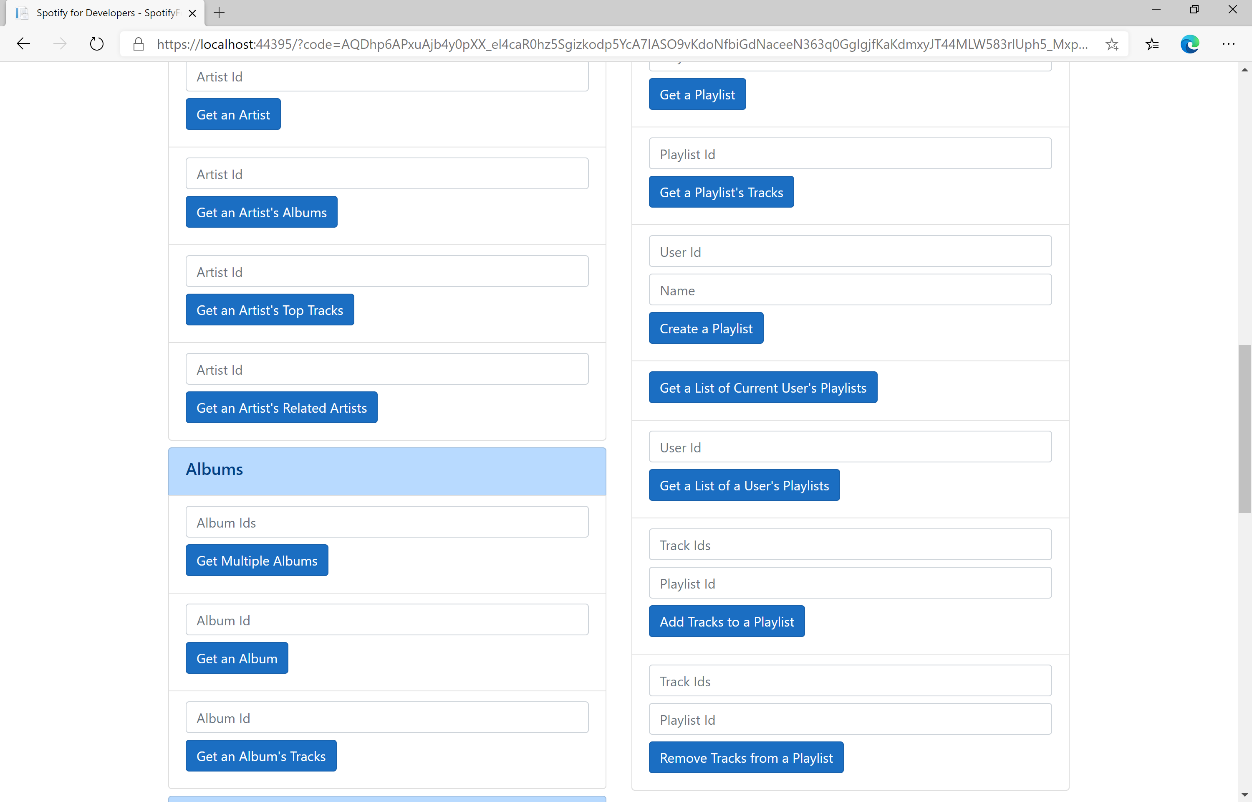
This form will **post** to the **method** for RemoveTracksFromPlaylist with the Value of the **Track Ids** and the Option of the **Playlist Id** and will output to the **Results**.

### Step 8

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

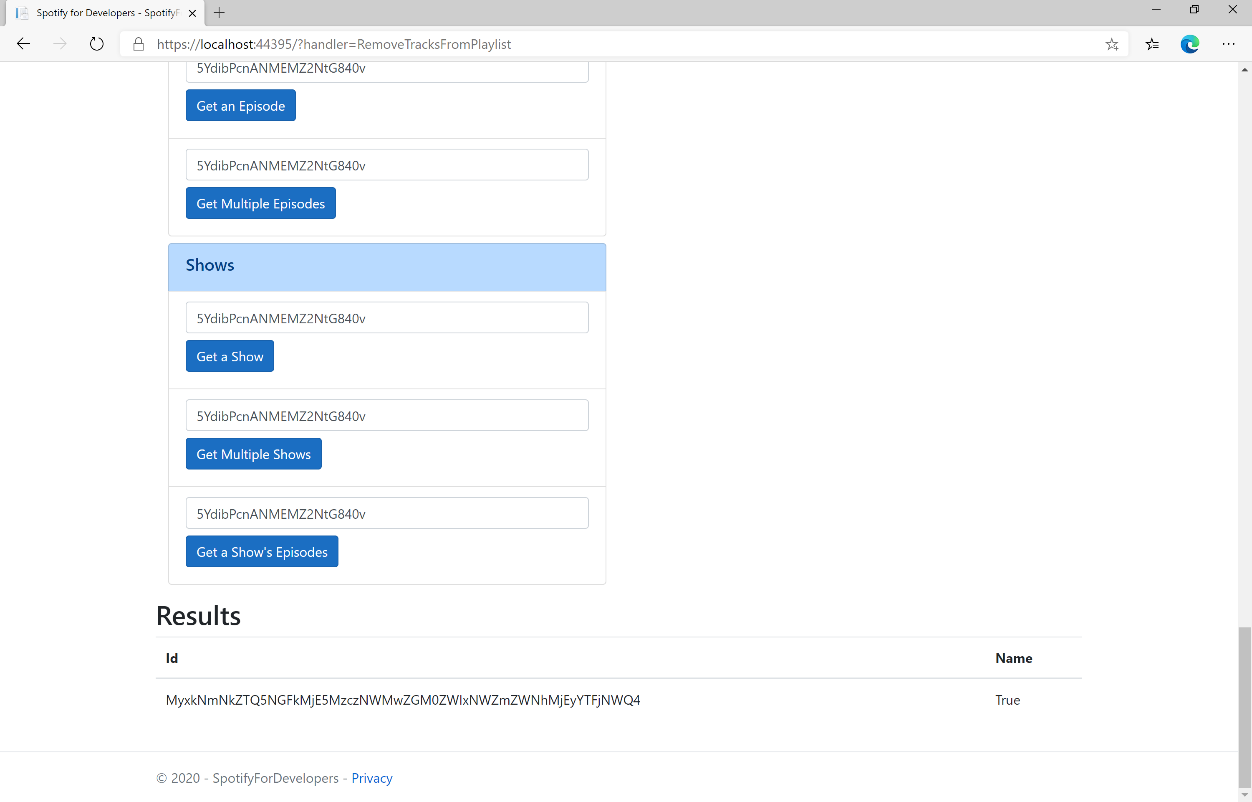
### Step 9

Once the **Web Application** is running and you select **Authorisation Code Flow Login** or **Implicit Grant Flow Login** and scroll down you should see something like the following:



### Step 10

You can use **Get a Playlist's Tracks** and the **Playlist Id** from **Create a Playlist** to get the **Track Ids** to enter into **Track Ids** above **Remove Tracks from a Playlist** and the **Playlist Id** then select **Remove Tracks from a Playlist** and scroll down to view **Results** like the following:



### Step 11

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

### Step 12

|  |  |
| --- | --- |
|  | You can choose to exit **Visual Studio 2019** by selecting the **Close** button in the top right of the **application** as that completes thispart of the workshop |

## Upload a Custom Playlist Cover Image

Replace the image used to represent a specific playlist.

|  |  |
| --- | --- |
| PUT https://api.spotify.com/v1/playlists/{playlist\_id}/images | |
| Header | |
| Authorization | User Token from Spotify Accounts service with **ugc-image-upload**, **playlist-modify-public** scope for publicly followed playlist and **playlist-modify-private** for a privately followed playlist |
| Path Parameter | |
| playlist\_id | Spotify Id for the Playlist |
| Body | |
|  | Base 64 Encoded JPEG with a maximum size of 256KB |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK |  |
| Error | |
| HTTP Status 429  Too Many Requests | Returned when have sent too many requests |
| Error Code | Error Object |

### Step 1

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | If you chose to close **Visual Studio 2019** previously, in **Windows 10** choose **Start**, and then from the **Start Menu** find and select **Visual Studio 2019** |
|  | Once done, from the **Get started** screen for **Visual Studio 2019** select **Open a project or solution** |
| A screenshot of a social media post  Description automatically generated | 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**, **Authorisation Guide**, **Search & Browse**, **Playlists & Artists**, **Albums & Tracks**, **Episodes & Shows**, **Follow** and **Playlists** |

### 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** below the **method** for public async Task<IActionResult> OnPostRemoveTracksFromPlaylistAsync(...) { ... } enter the following **method**:

|  |
| --- |
| public async Task<IActionResult> OnPostUploadCustomPlaylistCoverImageAsync(string value)  {  LoadToken();  System.IO.MemoryStream stream = new System.IO.MemoryStream();  Upload.OpenReadStream().CopyTo(stream);  var file = stream.ToArray();  var result = await Api.UploadCustomPlaylistCoverImageAsync(value, file);  if (result != null)  {  Results = new List<Result>() { new Result()  {  Id = result.Code.ToString(),  Name = result.Success.ToString()  }};  }  return Page();  } |

The **method** for OnPostUploadCustomPlaylistCoverImageAsync is **upload** an **image** with Upload for a **Playlist** by **Playlist Id** with Value on Spotify and populate the **property** for Results of the **success** of the operation accordingly.

### Step 5

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

### Step 6

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

### Step 7

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

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="UploadCustomPlaylistCoverImage"  enctype="multipart/form-data" method="post">  <input **asp-for**="Value" placeholder="Playlist Id" class="form-control mb-2" />  <input **asp-for**="Upload" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Upload a Custom Playlist Cover Image  </button>  </form>  </li> |

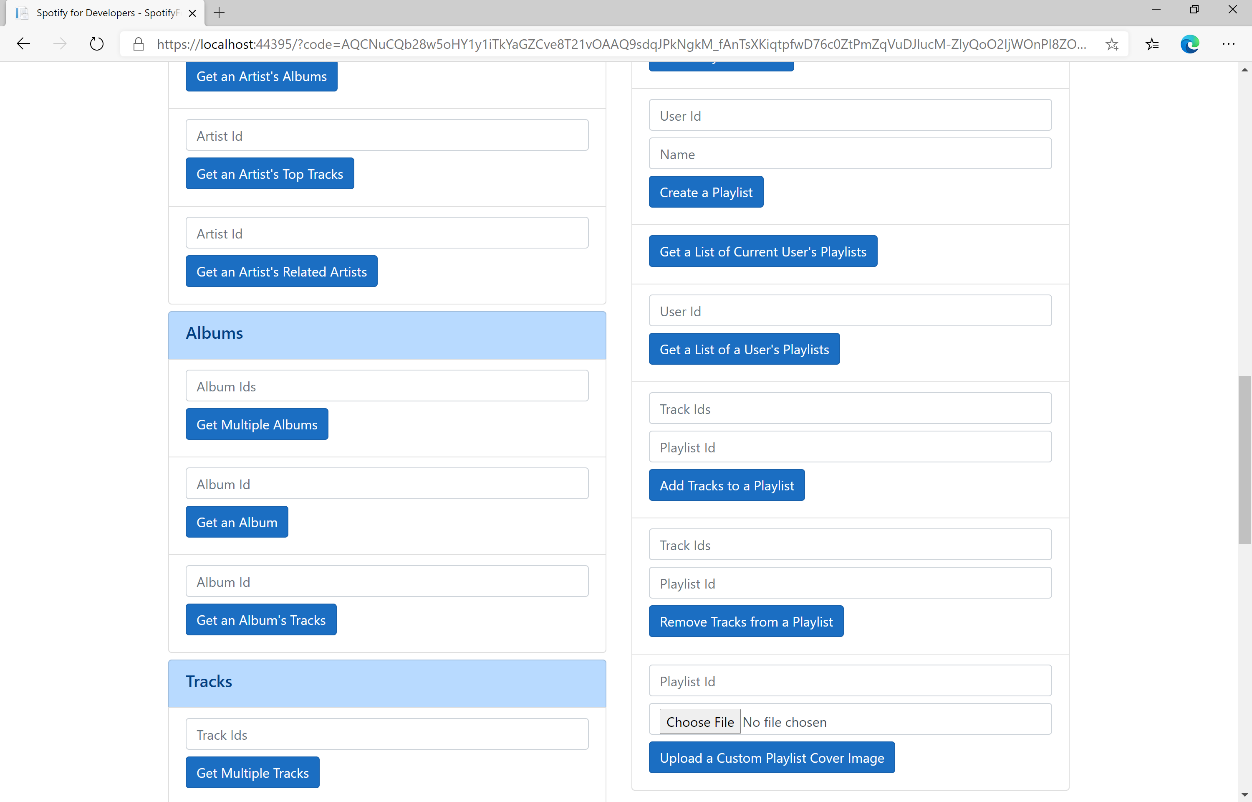
This form will **post** to the **method** for UploadCustomPlaylistCoverImage with the Value of the **Playlist Id** and the Upload of the **File** and will output to the **Results**.

### Step 8

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

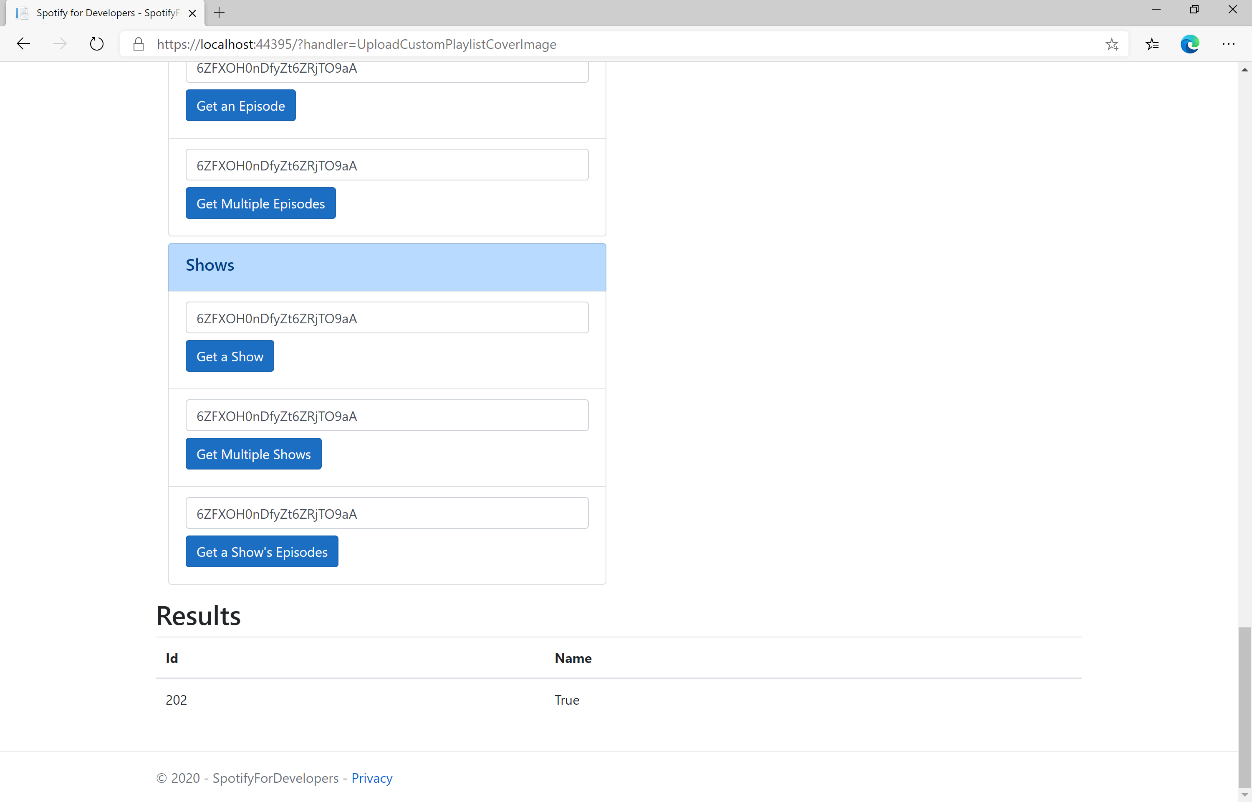
### Step 9

Once the **Web Application** is running and you select **Authorisation Code Flow Login** or **Implicit Grant Flow Login** and scroll down you should see something like the following:



### Step 10

You can then enter the **Playlist Id** from **Create a Playlist** into **Playlist Id** above **Upload a Custom Playlist Cover Image** and choose a **JPEG** file of **256KB** or smaller to upload and **Playlist Id** from **Create a Playlist** then select **Upload a Custom Playlist Cover Image** and scroll down to view **Results** like the following:



### Step 11

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

### Step 12

|  |  |
| --- | --- |
|  | You can choose to exit **Visual Studio 2019** by selecting the **Close** button in the top right of the **application** as that completes thispart of the workshop |

## Get a Playlist Cover Image

Get the current image associated with a specific playlist.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/playlists/{playlist\_id}/images | |
| Header | |
| Authorization | User Token from Spotify Accounts service |
| Path Parameter | |
| playlist\_id | Spotify Id for the Playlist |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | List of Image Objects |
| Error | |
| Error Code | Error Object |

### Step 1

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | If you chose to close **Visual Studio 2019** previously, in **Windows 10** choose **Start**, and then from the **Start Menu** find and select **Visual Studio 2019** |
|  | Once done, from the **Get started** screen for **Visual Studio 2019** select **Open a project or solution** |
| A screenshot of a social media post  Description automatically generated | 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**, **Authorisation Guide**, **Search & Browse**, **Playlists & Artists**, **Albums & Tracks**, **Episodes & Shows**, **Follow** and **Playlists** |

### 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** below the **method** for public async Task<IActionResult> OnPostUploadCustomPlaylistCoverImageAsync(...) { ... } enter the following **method**:

|  |
| --- |
| public async Task<IActionResult> OnPostGetPlaylistCoverImageAsync(string value)  {  LoadToken();  var results = await Api.GetPlaylistCoverImageAsync(value);  if (results != null)  {  Results = results.Select(result => new Result()  {  Image = result.Url  });  }  return Page();  } |

The **method** for OnPostGetPlaylistCoverImageAsync gets an **image** for a **Playlist** by **Playlist Id** with Value on Spotify and populate the **property** for Results accordingly.

### Step 5

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

### Step 6

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

### Step 7

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

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="GetPlaylistCoverImage" method="post">  <input **asp-for**="Value" placeholder="Playlist Id" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Get a Playlist Cover Image  </button>  </form>  </li> |

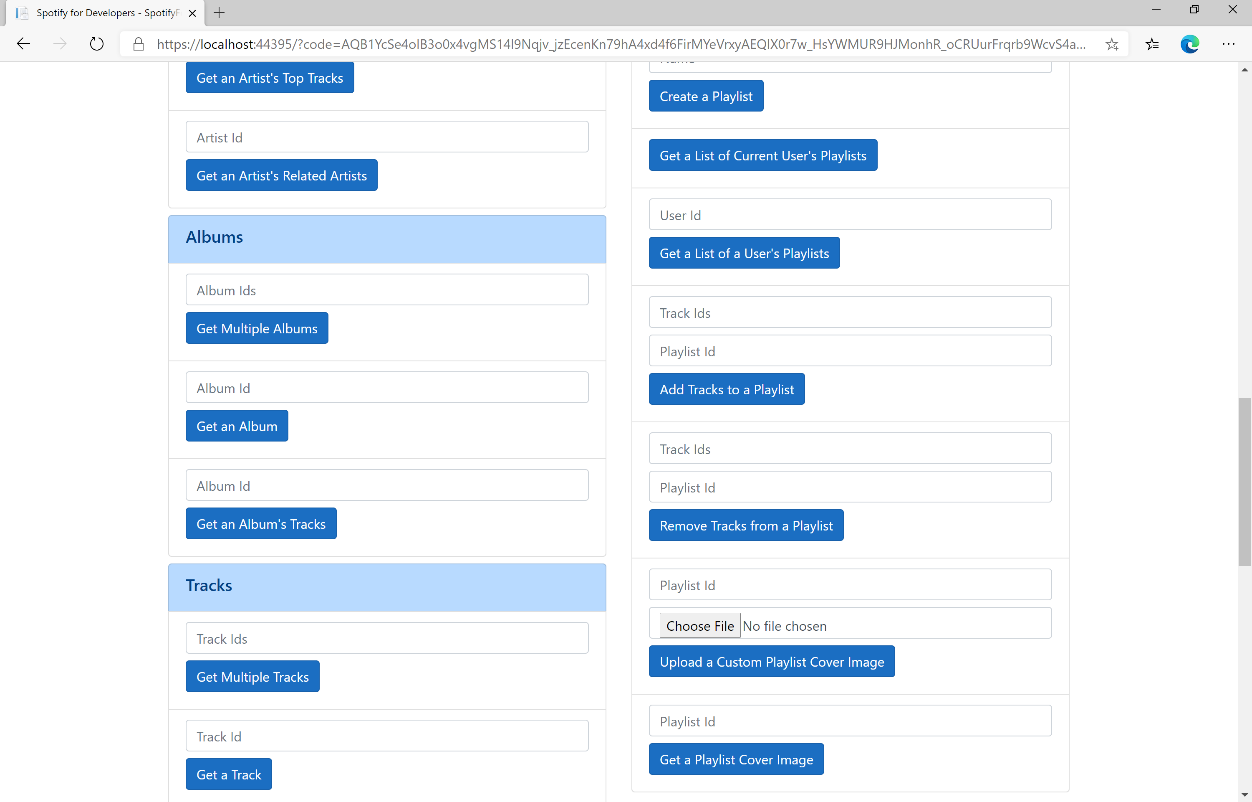
This form will **post** to the **method** for GetPlaylistCoverImage with the Value of the **Playlist Id** and the Upload of the **File** and will output to the **Results**.

### Step 8

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

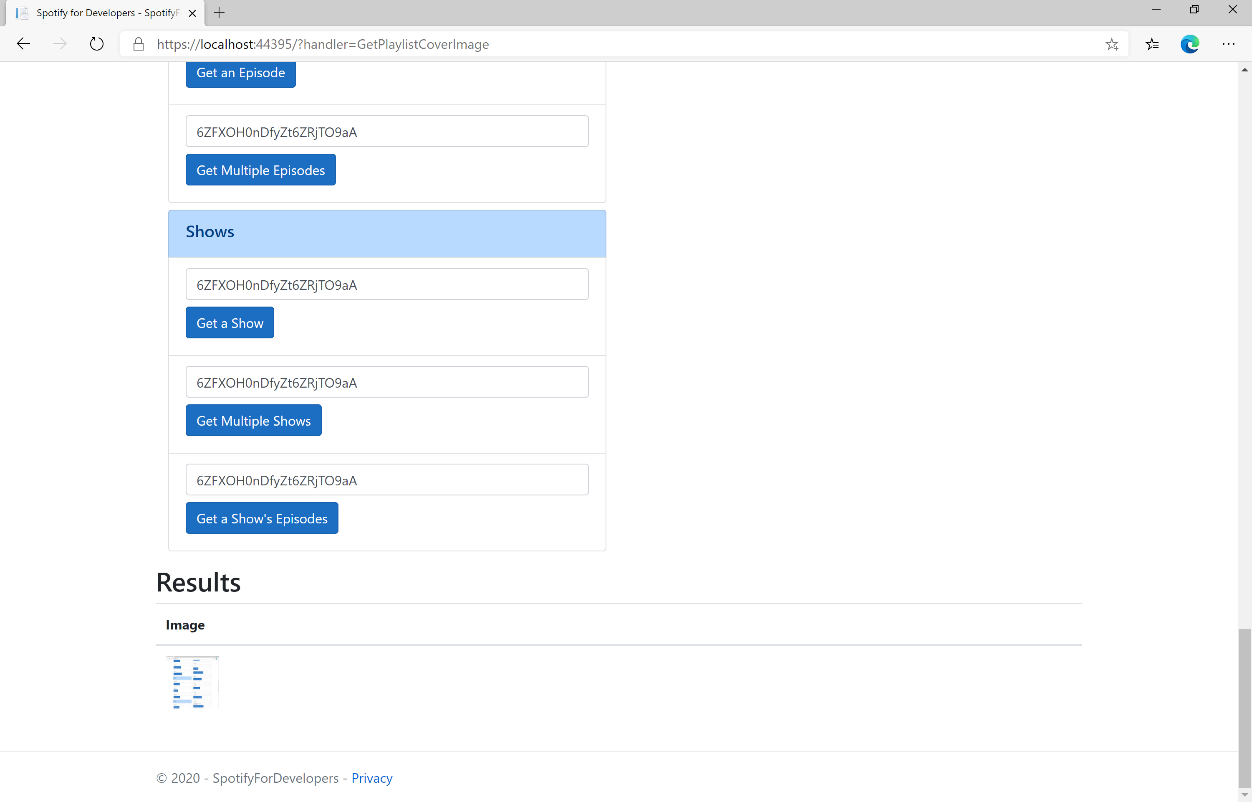
### Step 9

Once the **Web Application** is running and you select **Authorisation Code Flow Login** or **Implicit Grant Flow Login** and scroll down you should see something like the following:



### Step 10

You can then enter the **Playlist Id** from **Create a Playlist** into **Playlist Id** above **Get a Playlist Cover Image** then select **Get a Playlist Cover Image** and scroll down to view **Results** like the following:



### Step 11

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

### Step 12

|  |  |
| --- | --- |
|  | You can choose to exit **Visual Studio 2019** by selecting the **Close** button in the top right of the **application** as that completes thispart of the workshop |

## Change a Playlist's Details

Change a playlist’s name and public/private state of a playlist owned by a user

|  |  |
| --- | --- |
| PUT https://api.spotify.com/v1/playlists/{playlist\_id} | |
| Header | |
| Authorization | User Token from Spotify Accounts service with **playlist-modify-public** scope for publicly followed playlist and **playlist-modify-private** for a privately followed playlist |
| Path Parameter | |
| playlist\_id | Spotify Id for the Playlist |
| Body Parameter | |
| name | New name for the playlist |
| public | If true, the playlist will be public, if false it will be private |
| collaborative | If true, the playlist will be collaborative and other users will be able to modify the playlist |
| description | Playlist description |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK |  |
| Error | |
| HTTP Status 403  Forbidden | Returned when not Authorised |
| Error Code | Error Object |

### Step 1

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | If you chose to close **Visual Studio 2019** previously, in **Windows 10** choose **Start**, and then from the **Start Menu** find and select **Visual Studio 2019** |
|  | Once done, from the **Get started** screen for **Visual Studio 2019** select **Open a project or solution** |
| A screenshot of a social media post  Description automatically generated | 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**, **Authorisation Guide**, **Search & Browse**, **Playlists & Artists**, **Albums & Tracks**, **Episodes & Shows**, **Follow** and **Playlists** |

### 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** below the **method** for public async Task<IActionResult> OnPostGetPlaylistCoverImageAsync(...) { ... } enter the following **method**:

|  |
| --- |
| public async Task<IActionResult> OnPostChangePlaylistDetailsAsync(string value, string option)  {  LoadToken();  var result = await Api.ChangePlaylistDetailsAsync(value, option);  if (result != null)  {  Results = new List<Result>() { new Result()  {  Id = result.Code.ToString(),  Name = result.Success.ToString()  }};  }  return Page();  } |

The **method** for OnPostChangePlaylistDetailsAsync sets the **Name** for a **Playlist** of Option by **Playlist Id** with Value on Spotify and populate the **property** for Results of the **success** of the operation accordingly.

### Step 5

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

### Step 6

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

### Step 7

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

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="ChangePlaylistDetails" method="post">  <input **asp-for**="Value" placeholder="Playlist Id" class="form-control mb-2" />  <input **asp-for**="Option" placeholder="Name" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Change a Playlist's Details  </button>  </form>  </li> |

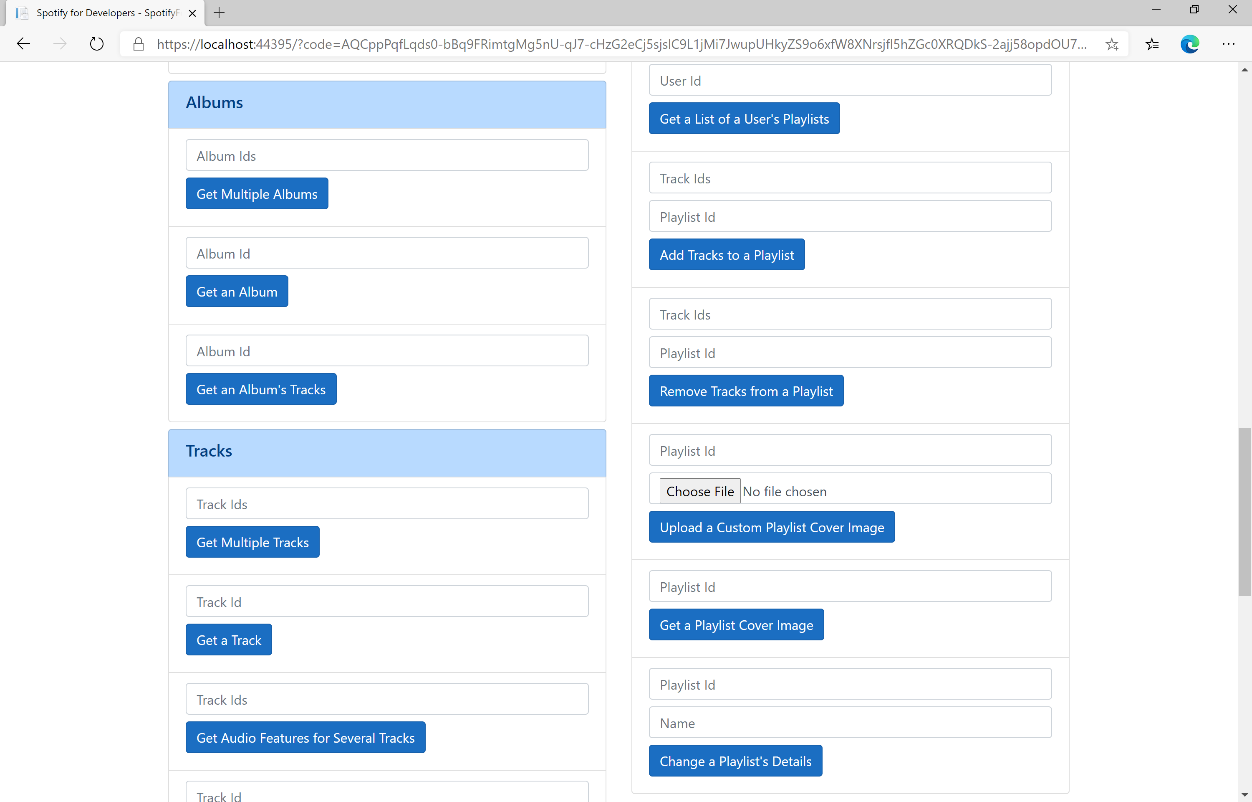
This form will **post** to the **method** for ChangePlaylistDetails with the Value of the **Playlist Id** and the Option of the **Name** and will output to the **Results**.

### Step 8

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

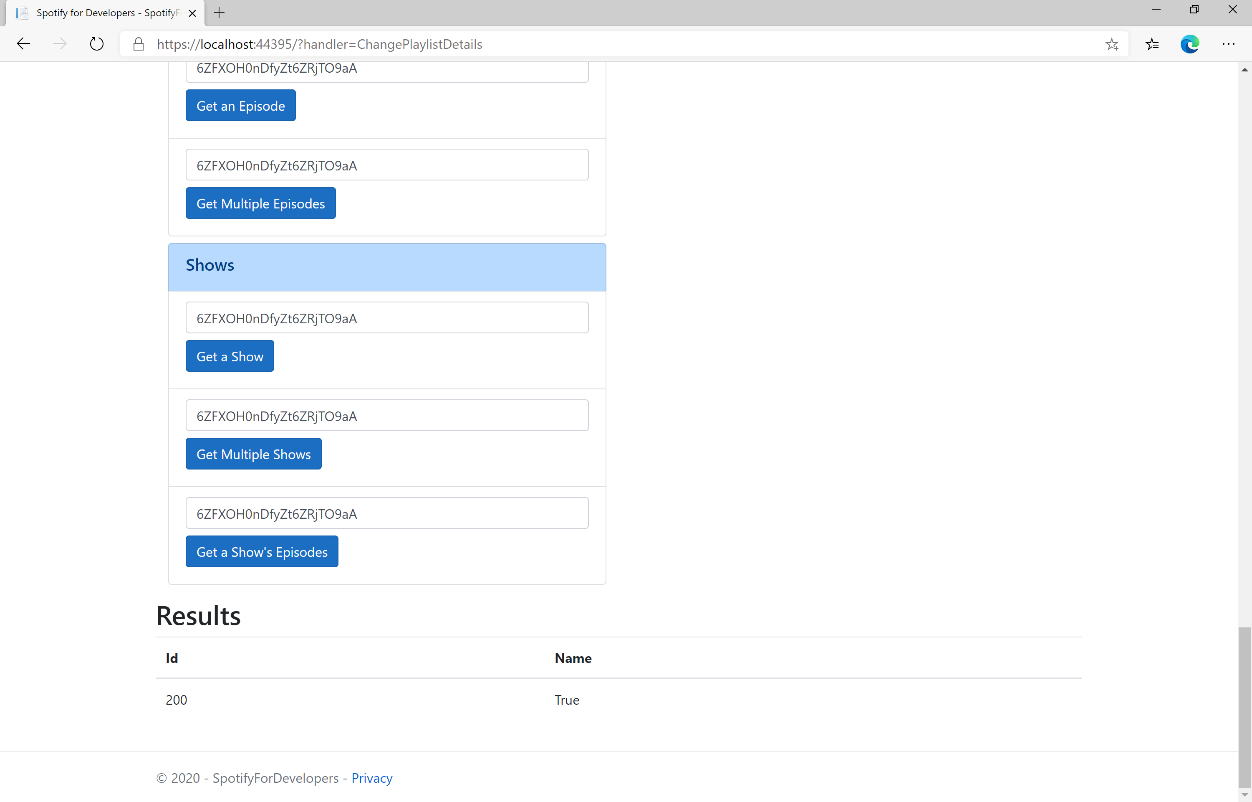
### Step 9

Once the **Web Application** is running and you select **Authorisation Code Flow Login** or **Implicit Grant Flow Login** and scroll down you should see something like the following:



### Step 10

You can then enter the **Playlist Id** from **Create a Playlist** into **Playlist Id** above **Change a Playlist's Details** and a new **Name** then select **Change a Playlist's Details** and scroll down to view **Results** like the following:



### Step 11

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

### Step 12

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

## Replace a Playlist's Tracks

Replace all the tracks in a playlist, overwriting its existing tracks.

|  |  |
| --- | --- |
| PUT https://api.spotify.com/v1/playlists/{playlist\_id}/tracks | |
| Header | |
| Authorization | User Token from Spotify Accounts service with **playlist-modify-public** scope for publicly followed playlist and **playlist-modify-private** for a privately followed playlist |
| Path Parameter | |
| playlist\_id | Spotify Id for the playlist |
| Query Parameter | |
| uris | Comma-separated list of Spotify track URIs to set – maximum of 100 |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| Http Status 201  Created |  |
| Error | |
| HTTP Status 403  Forbidden | Returned when not Authorised |
| Error Code | Error Object |

### Step 1

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | If you chose to close **Visual Studio 2019** previously, in **Windows 10** choose **Start**, and then from the **Start Menu** find and select **Visual Studio 2019** |
|  | Once done, from the **Get started** screen for **Visual Studio 2019** select **Open a project or solution** |
| A screenshot of a social media post  Description automatically generated | 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**, **Authorisation Guide**, **Search & Browse**, **Playlists & Artists**, **Albums & Tracks**, **Episodes & Shows**, **Follow** and **Playlists** |

### 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** below the **method** for public async Task<IActionResult> OnPostChangePlaylistDetailsAsync(...) { ... } enter the following **method**:

|  |
| --- |
| public async Task<IActionResult> OnPostReplacePlaylistTracksAsync(string value, string option)  {  LoadToken();  var values = value.Split(",").Select(value => $"spotify:track:{value}").ToList();  var result = await Api.ReplacePlaylistTracksAsync(option, values);  if (result != null)  {  Results = new List<Result>() { new Result()  {  Id = result.Code.ToString(),  Name = result.Success.ToString()  }};  }  return Page();  } |

The **method** for OnPostReplacePlaylistTracksAsync replaces the **tracks** for a **Playlist** with Value of the **Track Ids** and the **Playlist Id** of Option on Spotify and populate the **property** for Results of the **success** of the operation accordingly.

### Step 5

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

### Step 6

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

### Step 7

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

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="ReplacePlaylistTracks" method="post">  <input **asp-for**="Value" placeholder="Track Ids" class="form-control mb-2" />  <input **asp-for**="Option" placeholder="Playlist Id" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Replace a Playlist's Tracks  </button>  </form>  </li> |

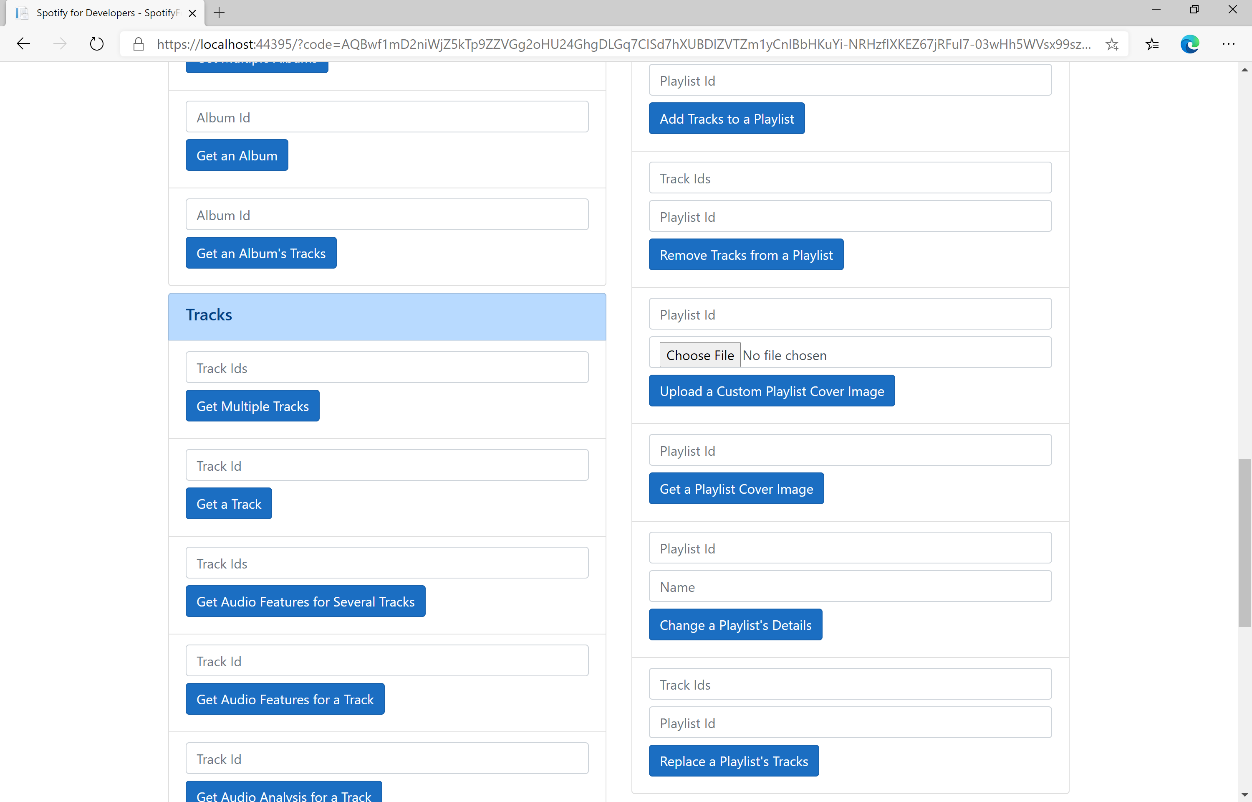
This form will **post** to the **method** for ReplacePlaylistTracks with the Value of the **Track Ids** and the Option of the **Playlist Id** and will output to the **Results**.

### Step 8

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

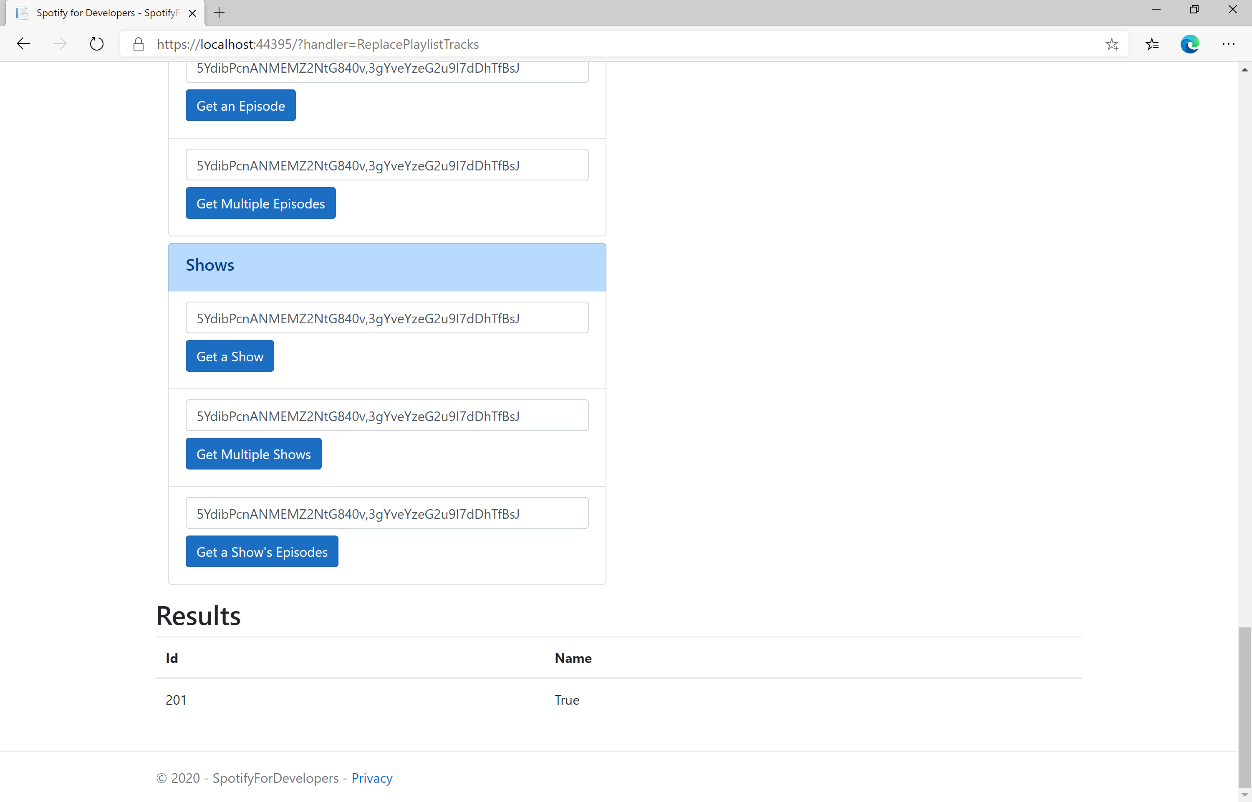
### Step 9

Once the **Web Application** is running and you select **Authorisation Code Flow Login** or **Implicit Grant Flow Login** and scroll down you should see something like the following:



### Step 10

You can then enter some **Track Ids** from an **Album Id** from **Get All New Releases** using **Get an Album's Tracks** into **Track Ids** above **Replace a Playlist's Tracks** and enter the **Playlist Id** from **Create a Playlist** into **Playlist Id** and then select **Replace a Playlist's Tracks** and scroll down to view **Results** like the following:



### Step 11

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

### Step 12

|  |  |
| --- | --- |
|  | You can choose to exit **Visual Studio 2019** by selecting the **Close** button in the top right of the **application** as that completes thispart of the workshop |

## Reorder a Playlist's Tracks

Reorder a track or a group of tracks in a playlist.

|  |  |
| --- | --- |
| PUT https://api.spotify.com/v1/playlists/{playlist\_id}/tracks | |
| Header | |
| Authorization | User Token from Spotify Accounts service with **playlist-modify-public** scope for publicly followed playlist and **playlist-modify-private** for a privately followed playlist |
| Path Parameter | |
| playlist\_id | Spotify Id for the playlist |
| Body Parameter | |
| range\_start | Position of the first track to be reordered |
| insert\_before | Position where the tracks should be inserted |
| range\_length | Number of tracks to be reordered. Defaults to 1 if not set. |
| snapshot\_id | Playlist’s snapshot ID against which you want to make the changes |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| Http Status 200 OK |  |
| Error | |
| Error Code | Error Object |

### Step 1

|  |  |
| --- | --- |
| A screenshot of a cell phone  Description automatically generated | If you chose to close **Visual Studio 2019** previously, in **Windows 10** choose **Start**, and then from the **Start Menu** find and select **Visual Studio 2019** |
|  | Once done, from the **Get started** screen for **Visual Studio 2019** select **Open a project or solution** |
| A screenshot of a social media post  Description automatically generated | 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**, **Authorisation Guide**, **Search & Browse**, **Playlists & Artists**, **Albums & Tracks**, **Episodes & Shows**, **Follow** and **Playlists** |

### 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** below the **method** for public async Task<IActionResult> OnPostReplacePlaylistTracksAsync(...) { ... } enter the following **method**:

|  |
| --- |
| public async Task<IActionResult> OnPostReorderPlaylistTracksAsync(string value, string option)  {  LoadToken();  var index = int.Parse(option);  var result = await Api.ReorderPlaylistTracksAsync(value, index, 0, 1);  if (result != null)  {  Results = new List<Result>() { new Result()  {  Id = result.SnapshotId.ToString(),  Name = result.Success.ToString()  }};  }  return Page();  } |

The **method** for OnPostReorderPlaylistTracksAsync **reorders** the **tracks** in a **Playlist** with the Value of the **Playlist Id** on Spotify and the Option is the **Index** of the **track** to move to the first position and populate the **property** for Results of the **success** of the operation accordingly.

### Step 5

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

### Step 6

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

### Step 7

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

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="ReorderPlaylistTracks" method="post">  <input **asp-for**="Value" placeholder="Playlist Id" class="form-control mb-2" />  <input **asp-for**="Option" placeholder="Index" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Reorder a Playlist's Tracks  </button>  </form>  </li> |

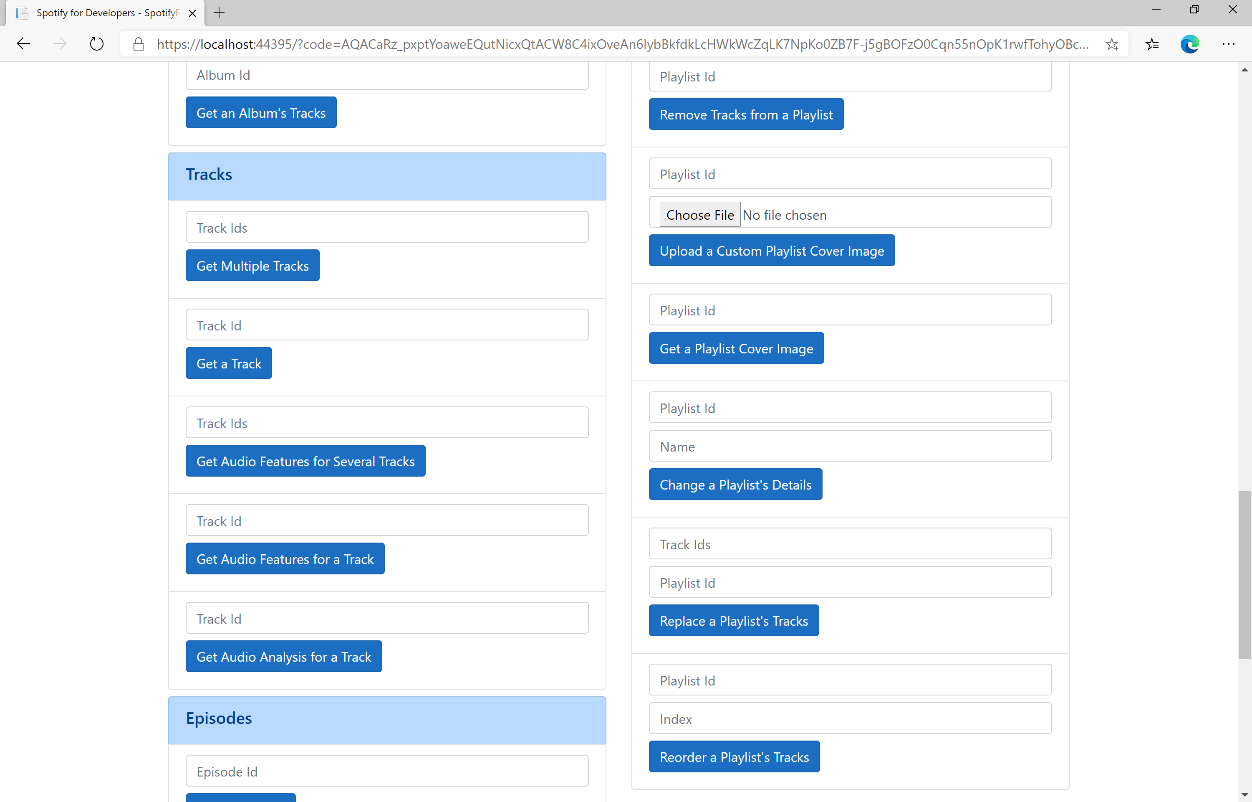
This form will **post** to the **method** for ReorderPlaylistTracks with the Value of the **Playlist Id** and the Option of **Index** and will output to the **Results**.

### Step 8

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

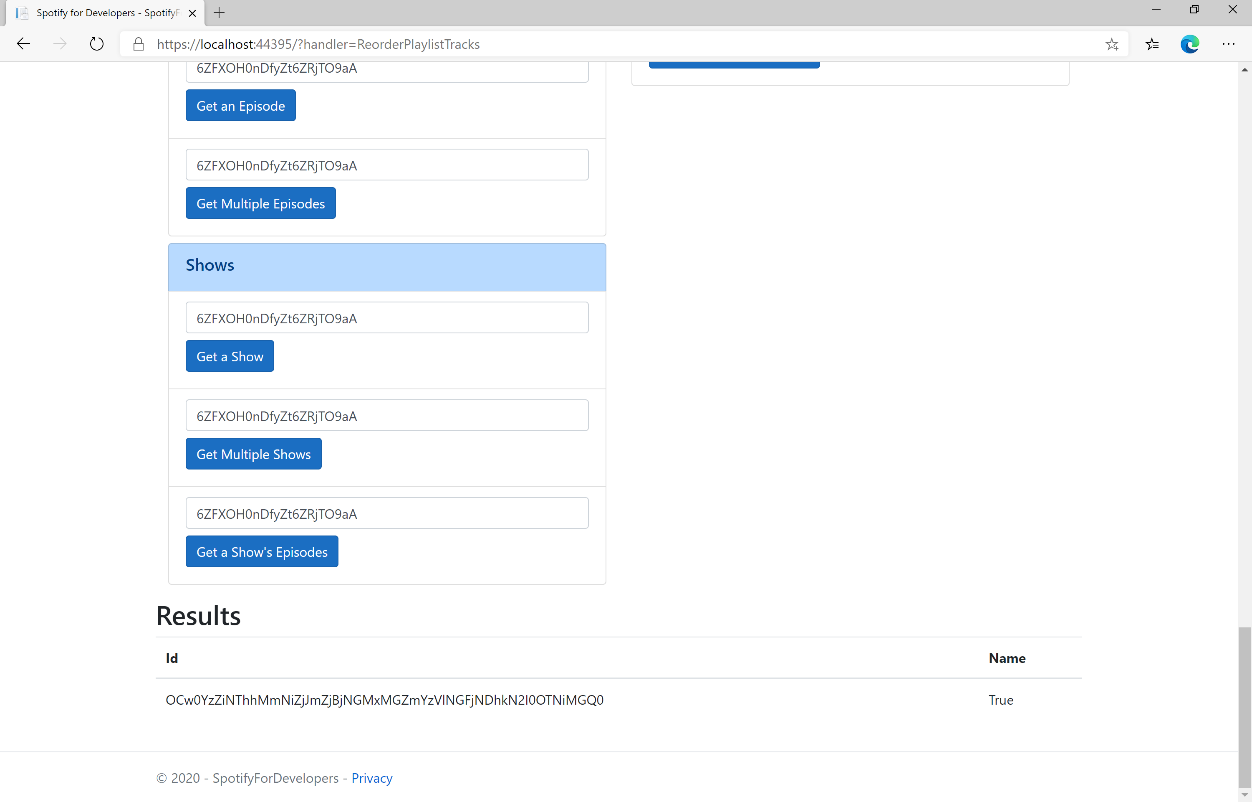
### Step 9

Once the **Web Application** is running and you select **Authorisation Code Flow Login** or **Implicit Grant Flow Login** and scroll down you should see something like the following:



### Step 10

You can use **Get a Playlist's Tracks** and **Playlist Id** from **Create a Playlists** to check the **Order** of the **Tracks** in the **Playlist**. You can enter above **Reorder a Playlist's Tracks** the **Playlist Id** from **Create a Playlist** into **Playlist Id**, then enter an **Index** e.g. **1** and select **Reorder a Playlist's Tracks** and scroll down to view **Results** like the following:



### Step 11

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

### Step 12

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