|  |
| --- |
|  |
| Spotify for Developers  Personalisation  & User Profile |
| |  |  |  | | --- | --- | --- | |  |  |  | |

## Get a User's Top Artists and Tracks

Get the current user’s top artists based on calculated affinity.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/me/top/ | |
| Header | |
| Authorization | User Token from Spotify Accounts service with **user-top-read** scope |
| Path Parameter | |
| type | The type of entity to return. Valid values: artists or tracks |
| Query Parameter | |
| limit | Number of entities to return. Default: 20. Minimum: 1. Maximum: 50 |
| offset | Index of the first entity to return. Default: 0 |
| time\_range | Over what time frame the affinities are computed. long\_term - calculated from several years of data and including all new data as it becomes available, medium\_term - approximately last 6 months and short\_term - approximately last 4 weeks. Default: medium\_term |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | List of Artists or Tracks wrapped in a Paging 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**, **Follow**, **Playlists**, **Library** and **Player** |

### 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> OnPostUserPlaybackTransferAsync() { ... } enter the following **method**:

|  |
| --- |
| public async Task<IActionResult> OnPostGetUserTopArtistsAndTracksAsync(string option)  {  LoadToken();  if(option.Equals("Artists"))  {  var results = await Api.GetUserTopArtistsAsync();  if (results?.Items != null)  {  Results = results.Items.Select(result => new Result()  {  Id = result.Id,  Name = result.Name,  Image = result?.Images?.FirstOrDefault()?.Url  });  }  }  else  {  var results = await Api.GetUserTopTracksAsync();  if (results?.Items != null)  {  Results = results.Items.Select(result => new Result()  {  Id = result.Id,  Name = result.Name,  Image = result?.Album?.Images?.FirstOrDefault()?.Url,  Inner = new Result()  {  Id = result?.Album?.Artists?.FirstOrDefault()?.Id,  Name = result?.Album?.Artists?.FirstOrDefault()?.Name  }  });  }  }  return Page();  } |

The **method** for OnPostGetUserTopArtistsAndTracksAsync is used to get the **top** played **artists** or **tracks** using the Option of the **Artists** and **Tracks** 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">Personalisation</h5>  </li>  <li class="list-group-item">  <form **asp-page-handler**="GetUserTopArtistsAndTracks" method="post">  <select **asp-for**="Option" class="form-control mb-2">  <option>Artists</option>  <option>Tracks</option>  </select>  <button class="btn btn-primary mb-2">  Get a User's Top Artists and Tracks  </button>  </form>  </li>  </ul> |

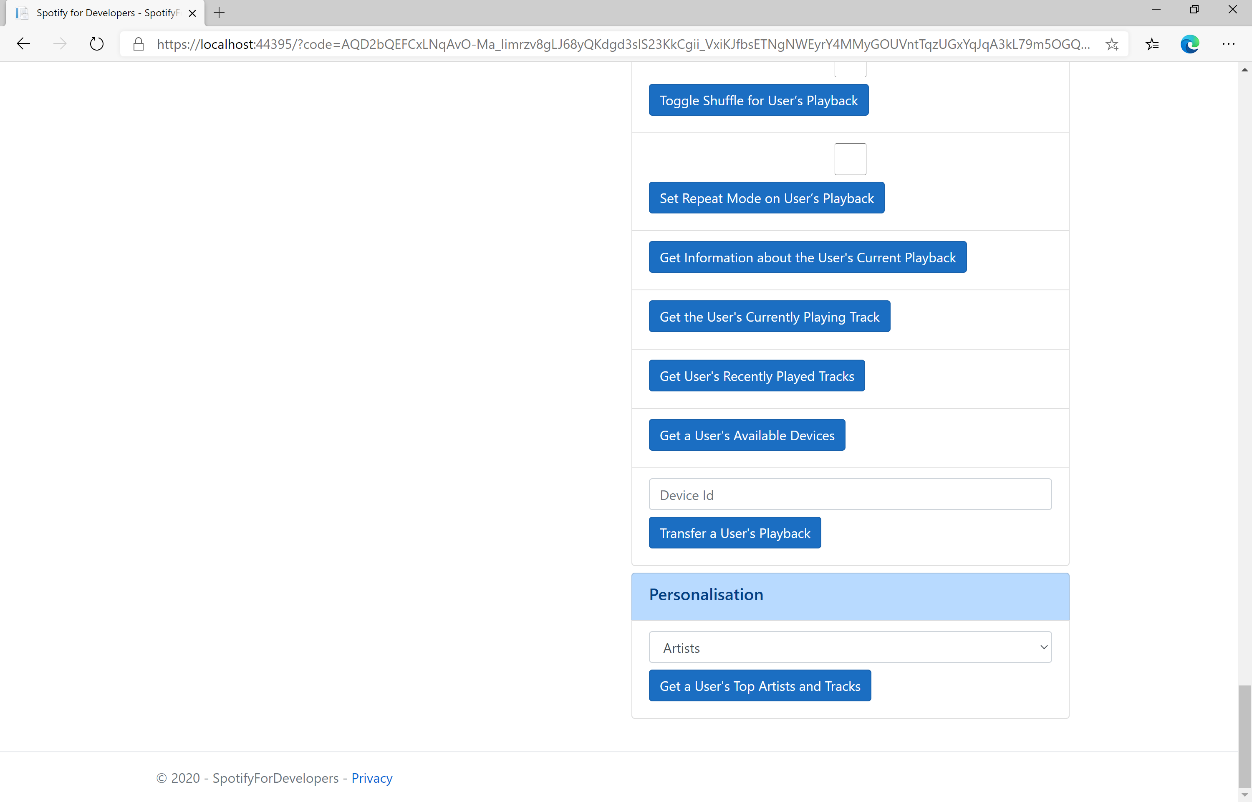
This form will **post** to the **method** for GetUserTopArtistsAndTracks with the Option of **Artists** and **Tracks** and will output to the **Results**.

### Step 8

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

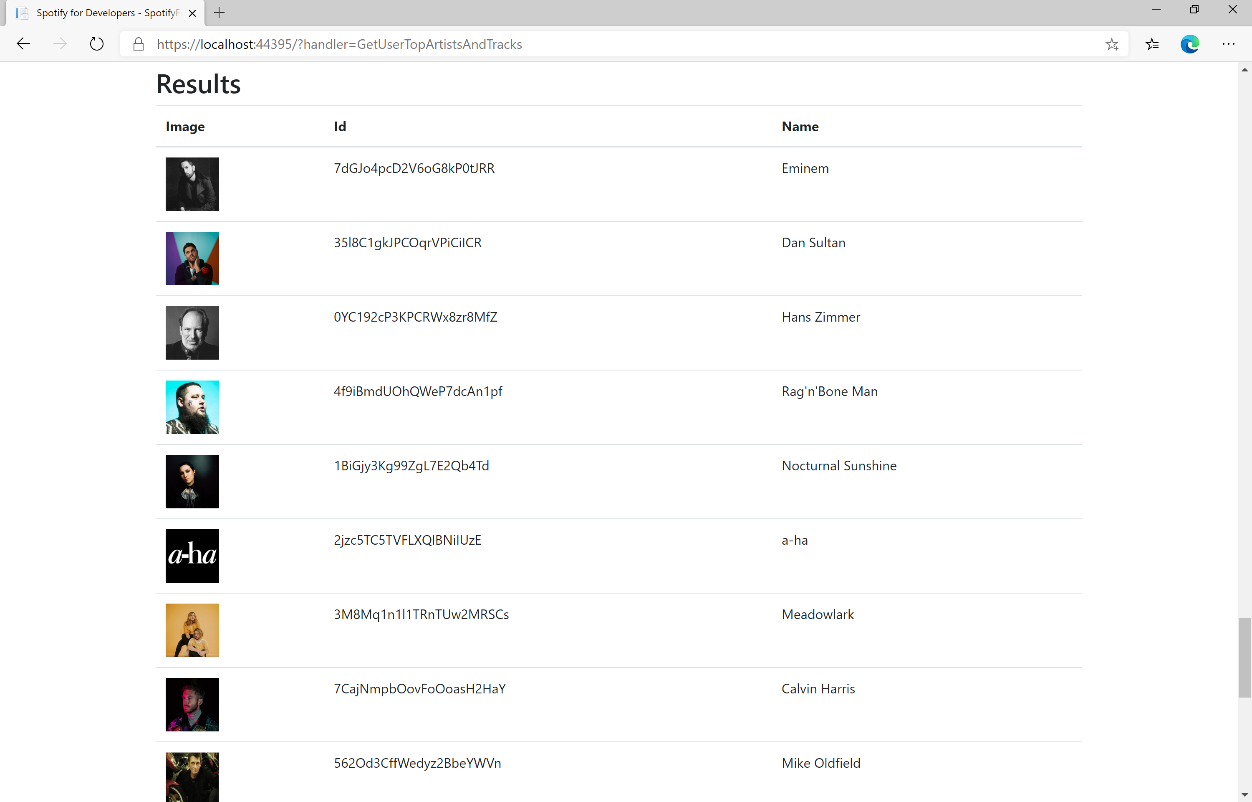
### Step 9

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 either **Artists** or **Tracks** above **Get a User's Top Artists and Tracks** and then select **Get a User's Top Artists and 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 |

## Get a User's Profile

Get public profile information about a Spotify user.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/users/{user\_id} | |
| Header | |
| Authorization | User Token from Spotify Accounts service |
| Path Parameter | |
| user\_id | The user’s Spotify User Id |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | List of Tracks wrapped in a Paging Object |
| Error | |
| HTTP Status 404  Not Found | User does not exist |
| 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**, **Playlists**, **Library**, **Player** and **Personalisation & User Profile** |

### 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> OnPostGetUserTopArtistsAndTracksAsync(...) { ... } enter the following **method**:

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

The **method** for OnPostGetUserProfileAsync is used to get the **profile** of a **user** using the Value of the **User Id** 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">User Profile</h5>  </li>  <li class="list-group-item">  <form **asp-page-handler**="GetUserProfile" method="post">  <input **asp-for**="Value" placeholder="User Id" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Get a User's Profile  </button>  </form>  </li>  <!-- User Profile -->  </ul> |

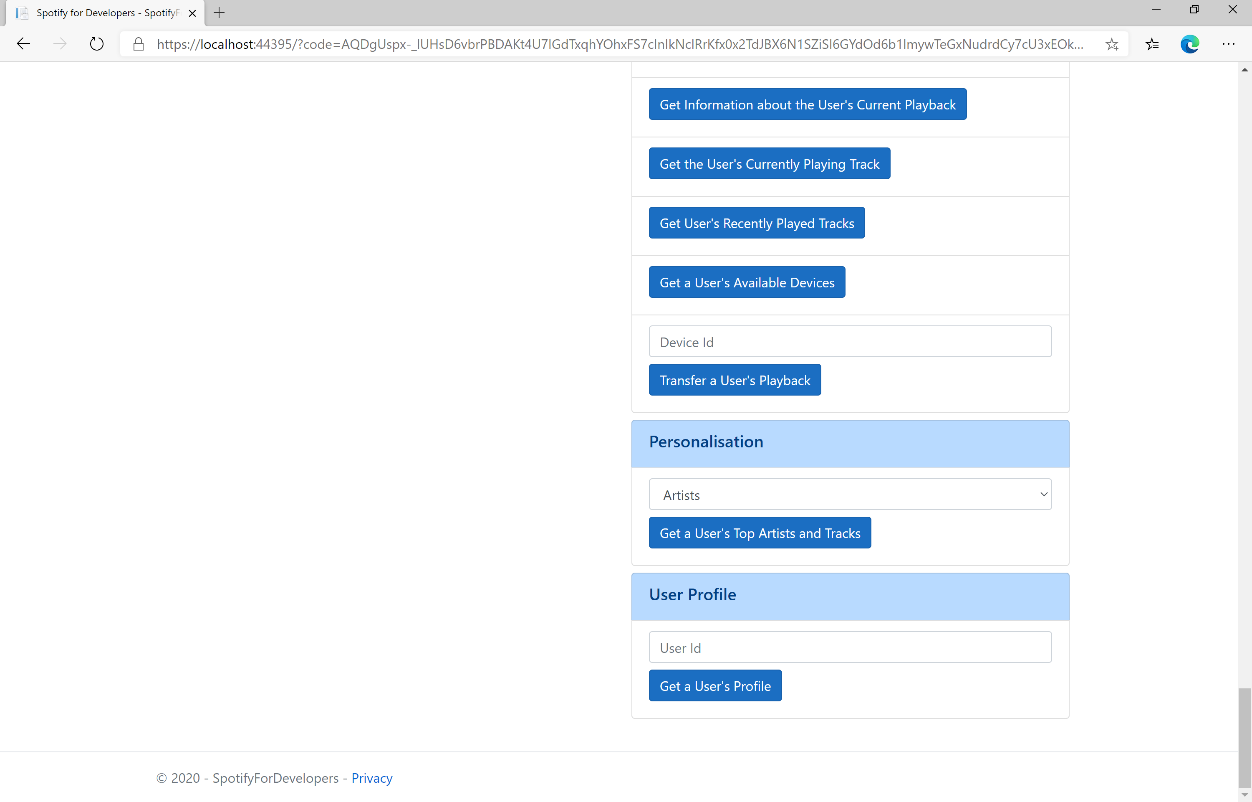
This form will **post** to the **method** for GetUserProfile with the Value of **User Id** and will output to the **Results**.

### Step 8

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

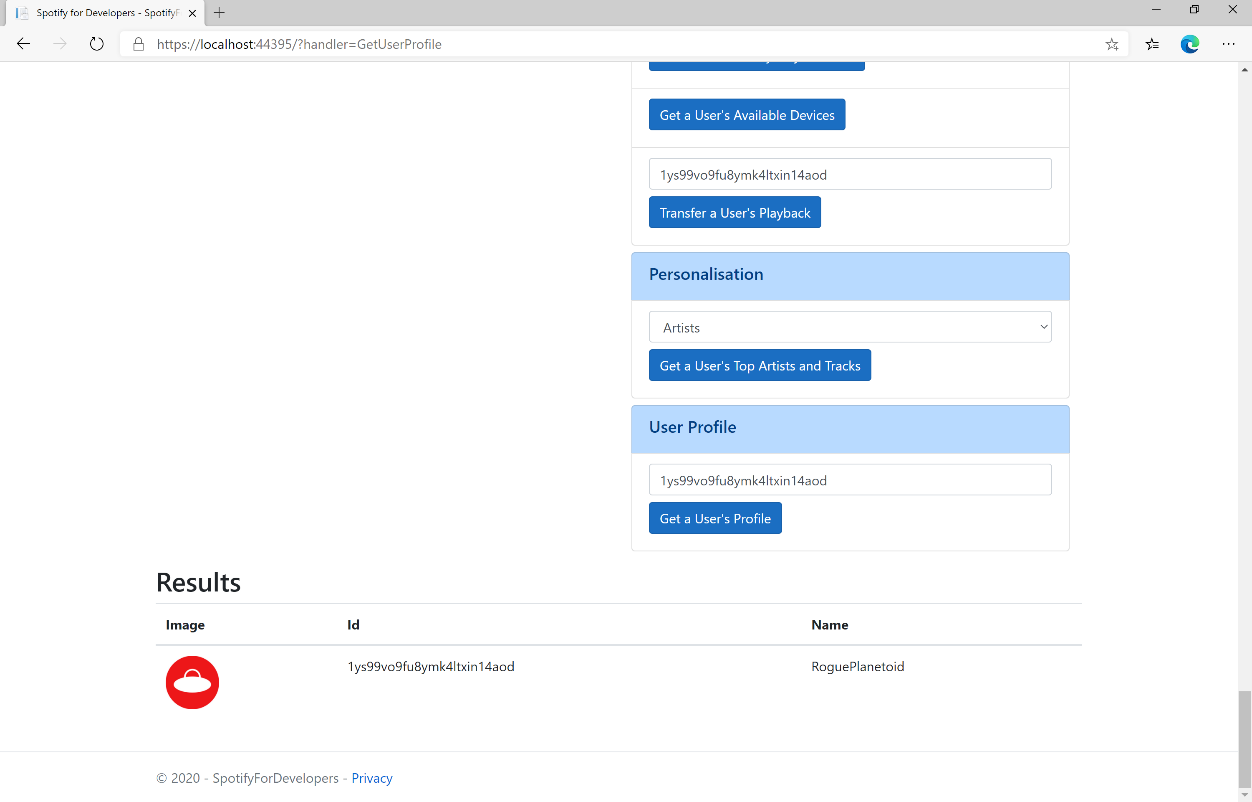
### Step 9

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 enter above **Get a User's Profile** 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 then select **Get a User's Profile** 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 Current User's Profile

Get detailed profile information about the current user including the current user’s username.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/me | |
| Header | |
| Authorization | User Token from Spotify Accounts service. Reading the user’s email address requires the **user-read-email** scope or reading country and product subscription level requires the **user-read-private scope** |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | List of Tracks wrapped in a Paging Object |
| Error | |
| HTTP Status 403  Forbidden | When requesting fields that you don’t have the user’s authorisation to access |
| 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**, **Playlists**, **Library**, **Player** and **Personalisation & User Profile** |

### 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> OnPostGetUserProfileAsync(...) { ... } enter the following **method**:

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

The **method** for OnPostGetCurrentUserProfileAsync is used to get the **profile** of the **current user** 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 <!-- User Profile --> enter the following:

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

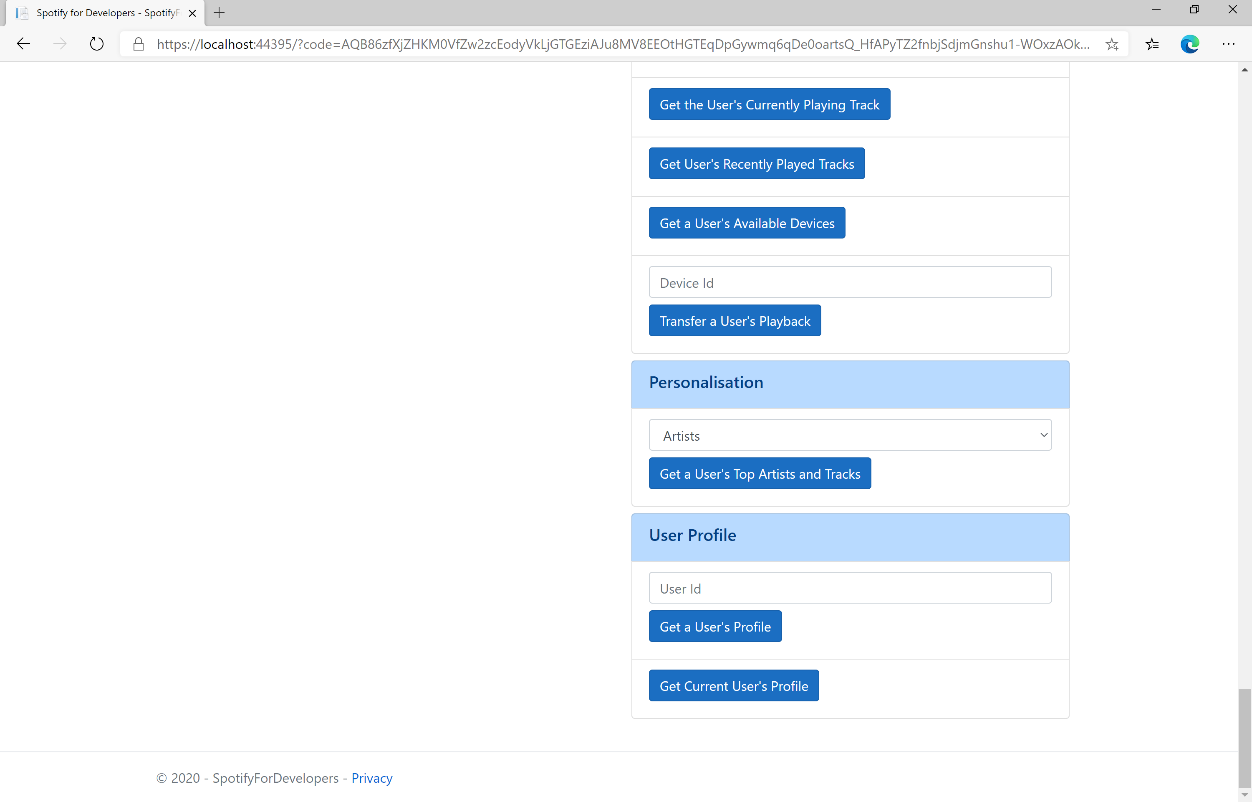
This form will **post** to the **method** for GetCurrentUserProfile and will output to the **Results**.

### Step 8

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

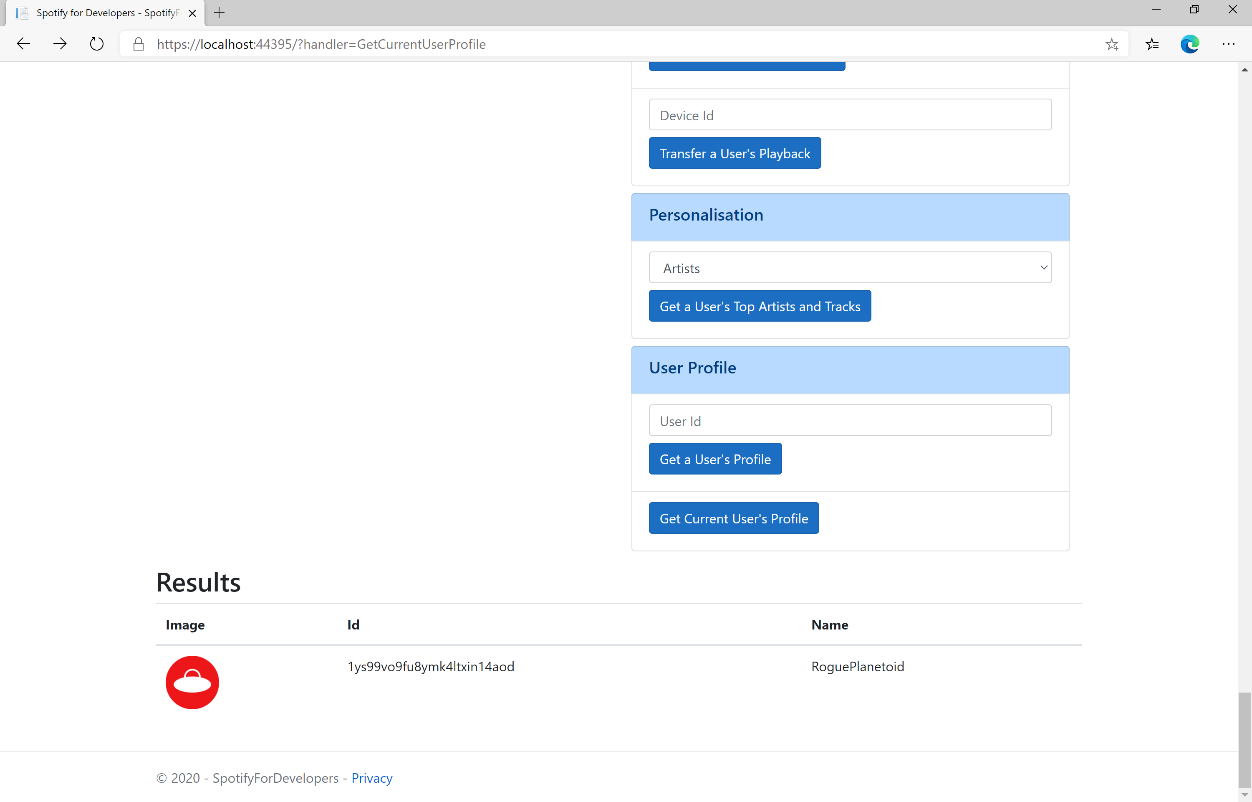
### Step 9

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 Current User's Profile** 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 thisworkshop |