|  |
| --- |
|  |
| Spotify for Developers  Search & Browse |
| |  |  |  | | --- | --- | --- | |  |  |  | |

## Search for an Item

Get Spotify Catalogue information about albums, artists, playlists, tracks, shows or episodes that match a query string.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/search?q=tania%20bowra&type=artist | |
| Header | |
| Authorization | Valid Access Token from Spotify Accounts service |
| Query Parameter | |
| q | Query keywords, optional field filters and operators |
| type | A comma-separated list of item types to search across. Valid types are: album, artist, playlist, track, show and episode |
| market | ISO 3166-1 alpha-2 country code |
| limit | Maximum number of results to return |
| offset | Index of first result to return |
| include\_external | “audio” includes any relevant audio content that is hosted externally |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | Array of Artist Objects / Simplified Album Objects / Track Objects / Simplified Show Object / Simplified Episode Object 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** and **Authorisation Guide** |

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

|  |
| --- |
| public async Task<IActionResult> OnPostSearchForAnItemAsync(string value, string option)  {  LoadToken();  var results = new List<Result>();  var search = await Api.SearchForItemAsync(value,  new SearchType()  {  Album = option.Contains("Album"),  Track = option.Contains("Track"),  Artist = option.Contains("Artist"),  Playlist = option.Contains("Playlist"),  Show = option.Contains("Show"),  Episode = option.Contains("Episode")  }, country);  if (search?.Albums?.Items != null)  {  results.AddRange(search.Albums.Items.Select(result => new Result(  result.Id, result.Name, result?.Images?.FirstOrDefault()?.Url,  new Result(result?.Artists[0]?.Id, result?.Artists[0]?.Name))));  }  if (search?.Tracks?.Items != null)  {  results.AddRange(search.Tracks.Items.Select(result => new Result(  result.Id, result.Name, result.Album?.Images?.FirstOrDefault()?.Url,  new Result(result?.Artists[0]?.Id, result?.Artists[0]?.Name))));  }  if (search?.Artists?.Items != null)  {  results.AddRange(search.Artists.Items.Select(result => new Result(  result.Id, result.Name, result?.Images?.FirstOrDefault()?.Url)));  }  if (search?.Playlists?.Items != null)  {  results.AddRange(search.Playlists.Items.Select(result => new Result(  result.Id, result.Name, result?.Images?.FirstOrDefault()?.Url)));  }  if (search?.Shows?.Items != null)  {  results.AddRange(search.Shows.Items.Select(result => new Result(  result.Id, result.Name, result?.Images?.FirstOrDefault()?.Url)));  }  if (search?.Episodes?.Items != null)  {  results.AddRange(search.Episodes.Items.Select(result => new Result(  result.Id, result.Name, result?.Images?.FirstOrDefault()?.Url)));  }  if (results.Any())  Results = results;  return Page();  } |

The **method** for OnPostSearchForAnItemAsync is used to find any **track**, **artist**, **album** or **playlist** and populates **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 App 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">Search</h5>  </li>  <li class="list-group-item">  <form **asp-page-handler**="SearchForAnItem" method="post">  <select **asp-for**="Option" class="form-control mb-2">  <option>Album</option>  <option>Track</option>  <option>Artist</option>  <option>Playlist</option>  <option>Show</option>  <option>Episode</option>  </select>  <input **asp-for**="Value" placeholder="Query" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Search For An Item  </button>  </form>  </li>  </ul> |

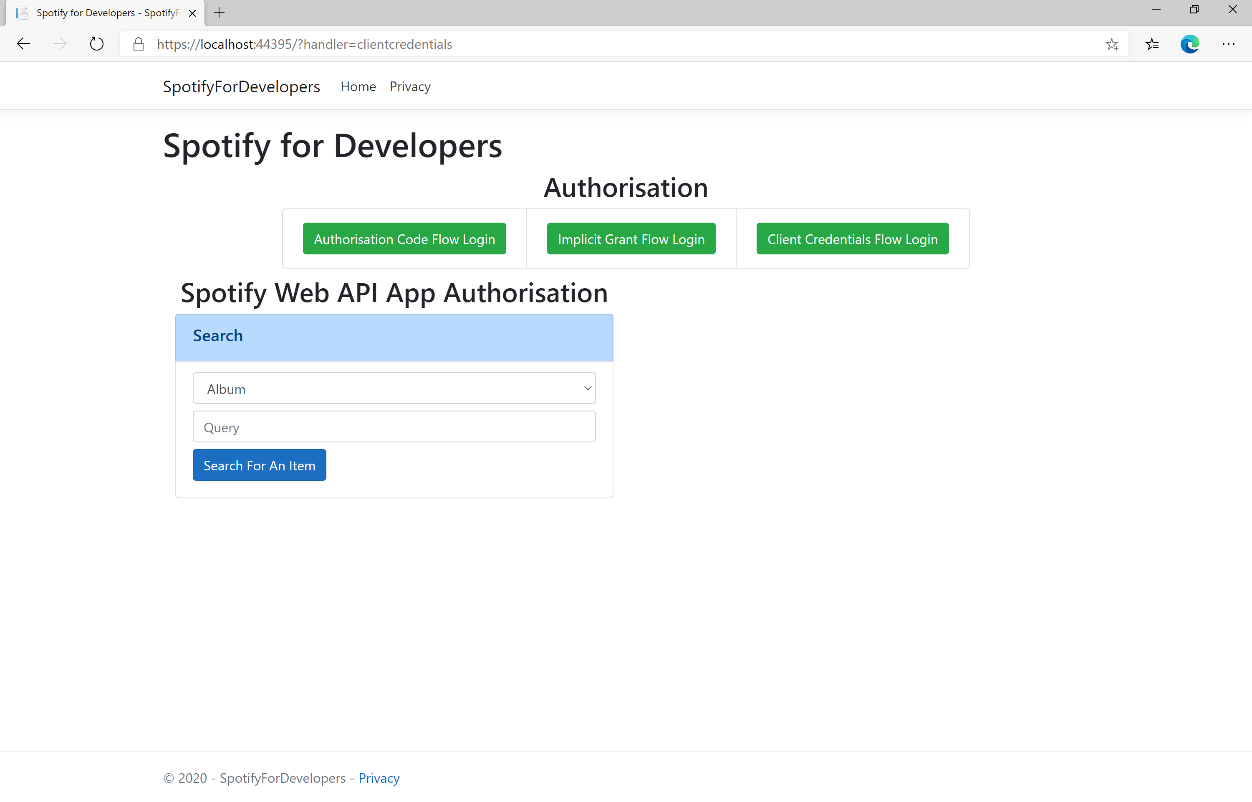
This form will **post** to the **method** for SearchForAnItemAsync and has a select list to choose an option to search for and an input for the Query.

### Step 8

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

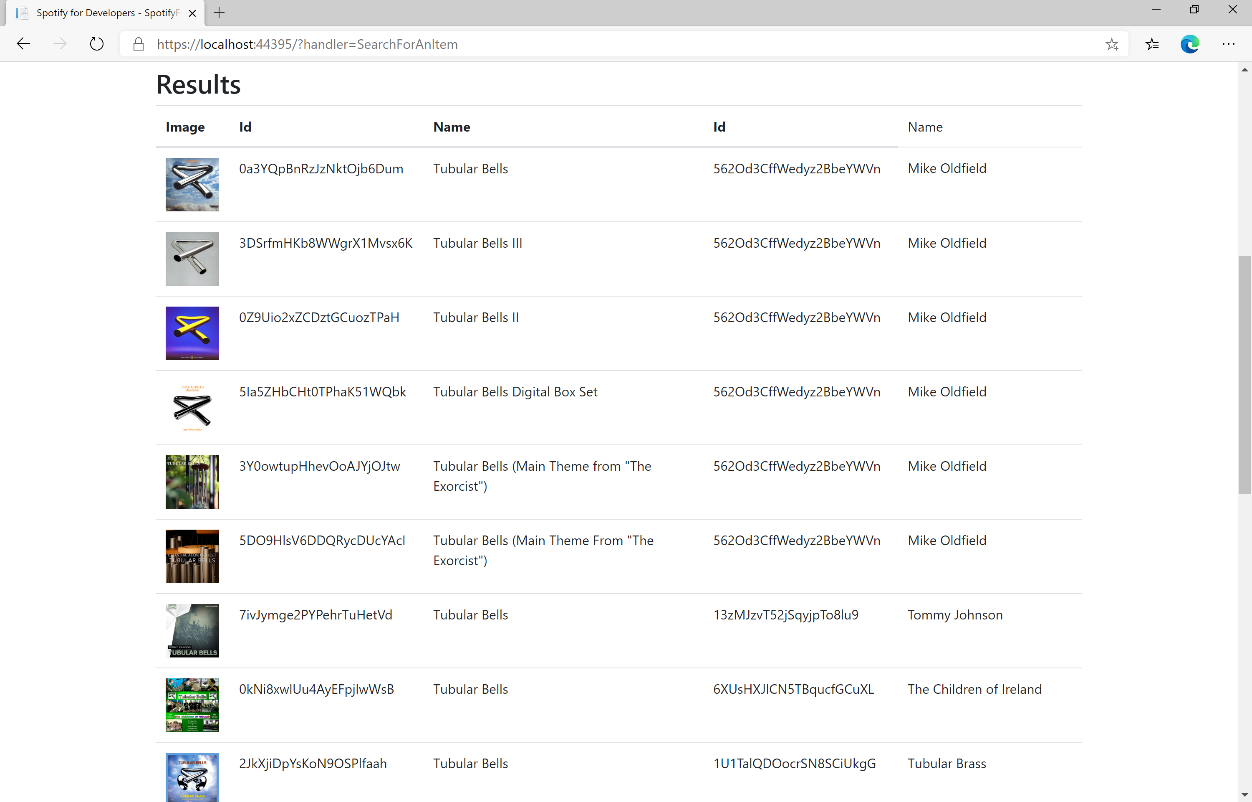
### Step 9

Once the **Web Application** is running and you select **Client Credentials Flow Login** you should see something like the following:



### Step 10

You can then search for something, such as an **Album** and select **Search For An Item** then 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 All Categories

Get a list of categories used to tag items in Spotify, as shown in the Spotify player’s “Browse” tab.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/browse/categories | |
| Header | |
| Authorization | Valid Access Token from Spotify Accounts service |
| Query Parameter | |
| country | ISO 3166-1 alpha-2 country code e.g. “GB”. If omitted globally relevant items returned |
| locale | Desired language, consisting of ISO 639-1 language code and ISO 3166-1 alpha-2 country code, joined by an underscore e.g. “en\_GB”. If omitted results returned in American English |
| limit | Maximum number of results to return |
| offset | Index of first result to return |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | Categories field with an array of Category 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** and **Search** **&** **Browse** |

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

|  |
| --- |
| public async Task<IActionResult> OnPostGetAllCategoriesAsync()  {  LoadToken();  var results = await Api.GetAllCategoriesAsync(country);  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 OnPostGetAllCategoriesAsync is used to get all the **categories** 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 <!-- Spotify Web API App 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">Browse</h5>  </li>  <li class="list-group-item">  <form **asp-page-handler**="GetAllCategories" method="post">  <button class="btn btn-primary mb-2">  Get All Categories  </button>  </form>  </li>  <!-- Browse -->  </ul> |

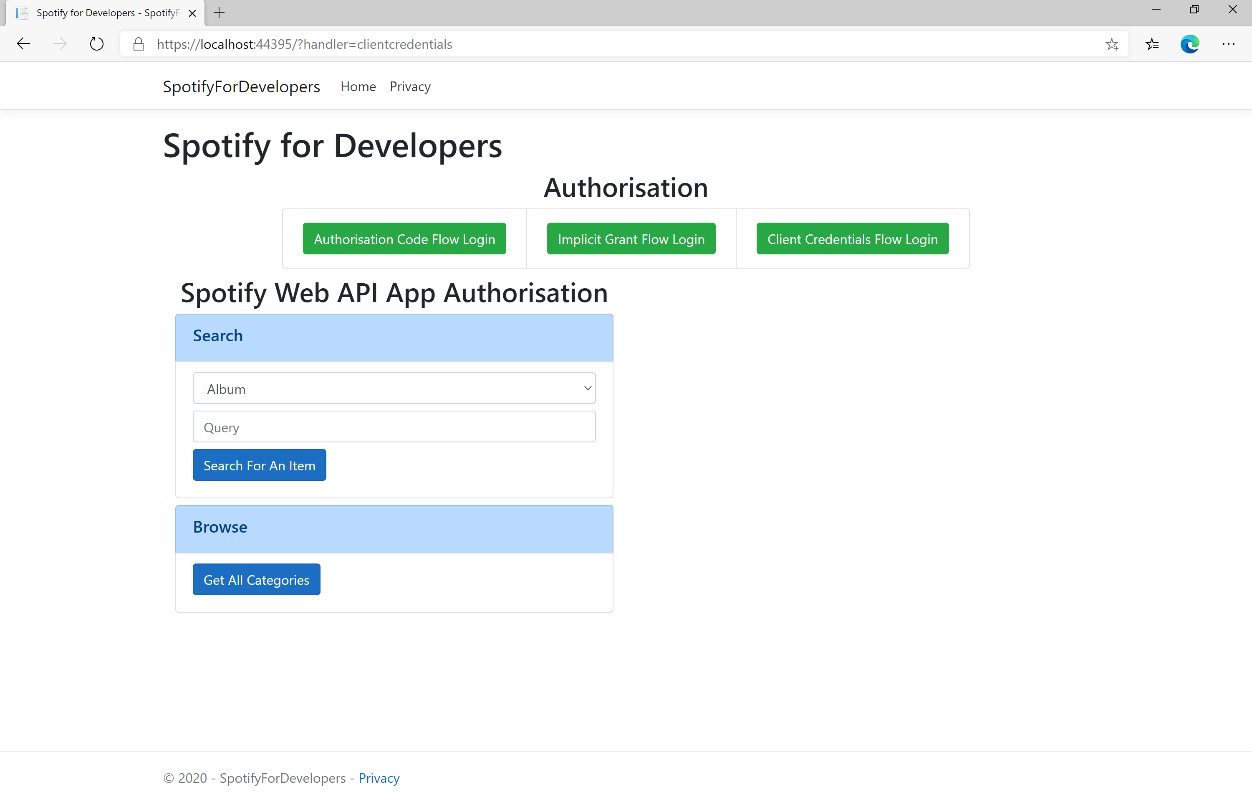
This form will **post** to the **method** for GetAllCategoriesAsync and will output to the **Results**.

### Step 8

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

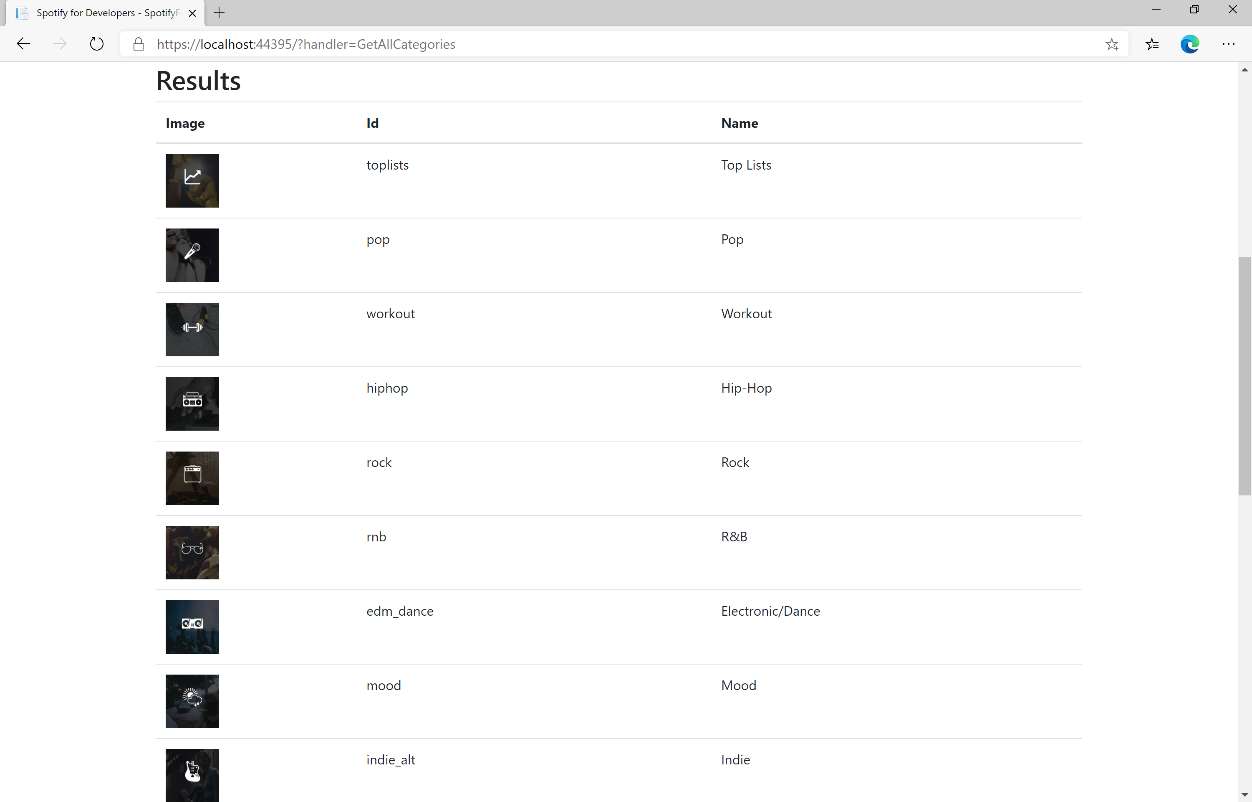
### Step 9

Once the **Web Application** is running and you select **Client Credentials Flow Login** you should see something like the following:



### Step 10

You can then select **Get All Categories** 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 Category

Get a single category used to tag items in Spotify, as shown in the Spotify player’s “Browse” tab.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/browse/categories/{category\_id} | |
| Header | |
| Authorization | Valid Access Token from Spotify Accounts service |
| Path Parameter | |
| category\_id | Spotify category ID for the category |
| Query Parameter | |
| country | ISO 3166-1 alpha-2 country code e.g. “GB”. If omitted globally relevant items returned |
| locale | Desired language, consisting of ISO 639-1 language code and ISO 3166-1 alpha-2 country code joined by an underscore e.g. “en\_GB”. If omitted results returned in American English |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | Category 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** and **Search** **&** **Browse** |

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

|  |
| --- |
| public async Task<IActionResult> OnPostGetCategoryAsync(string value)  {  LoadToken();  var result = await Api.GetCategoryAsync(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 OnPostGetCategoryAsync is used to get a **category** by **Category 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 <!-- Browse --> enter the following:

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

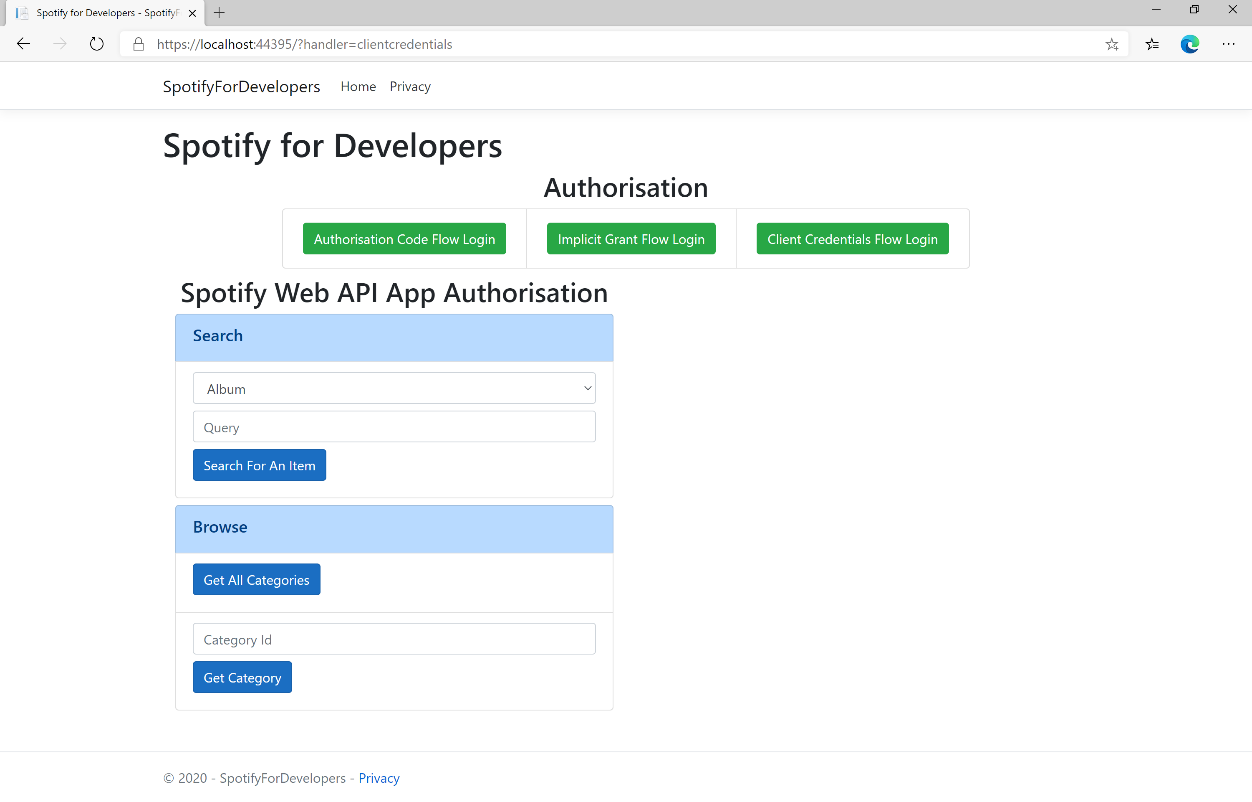
This form will **post** to the **method** for OnPostGetCategoryAsync with the Value as the **Category Id** and will output to the **Results**.

### Step 8

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

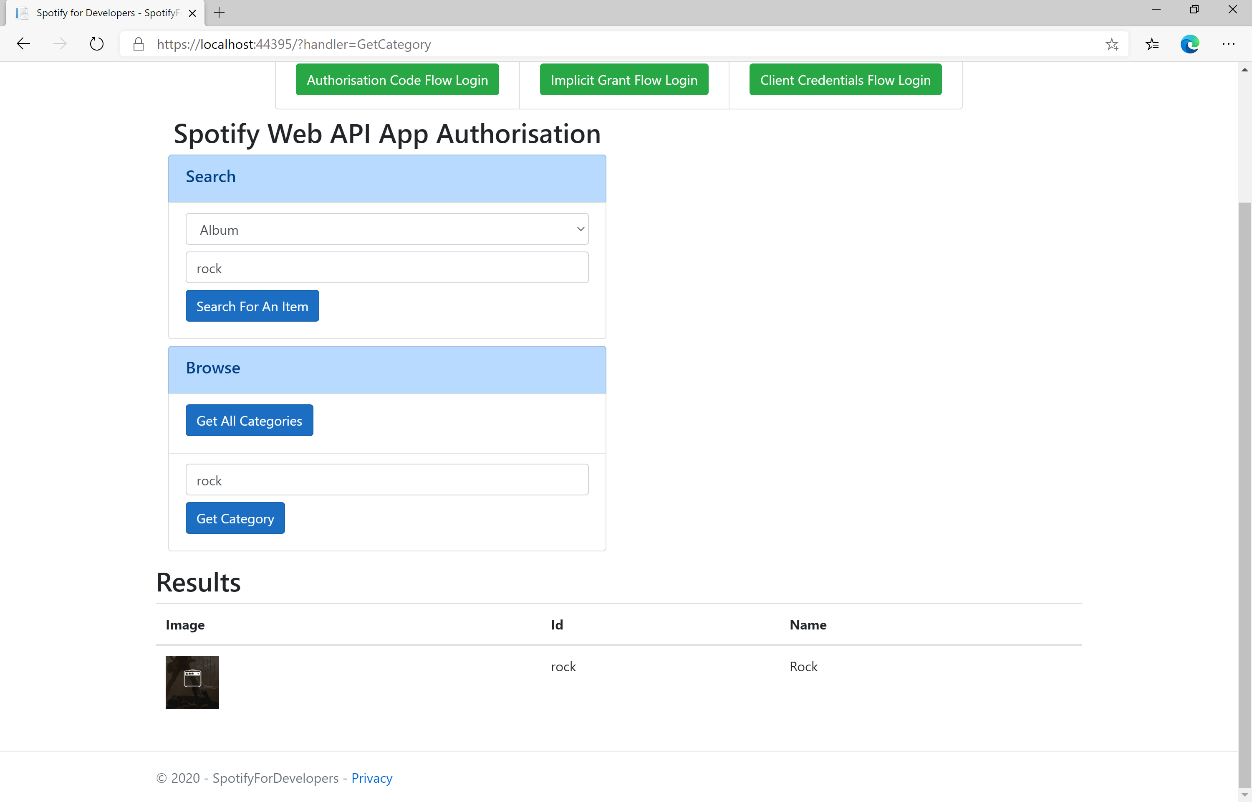
### Step 9

Once the **Web Application** is running and you select **Client Credentials Flow Login** you should see something like the following:



### Step 10

You can then enter a **Category Id** from **Get All Categories** and select **Get Category** 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 Category's Playlists

Get a list of Spotify playlists tagged with a category.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/browse/categories/{category\_id}/playlists | |
| Header | |
| Authorization | Valid Access Token from Spotify Accounts service |
| Path Parameter | |
| category\_id | Spotify category ID for the category |
| Query Parameter | |
| country | ISO 3166-1 alpha-2 country code e.g. “GB”. If omitted globally relevant items returned |
| limit | Maximum number of results to return |
| offset | Index of first result to return |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | Array of Simplified Playlist 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** and **Search** **&** **Browse** |

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

|  |
| --- |
| public async Task<IActionResult> OnPostGetCategoryPlaylistsAsync(string value)  {  LoadToken();  var results = await Api.GetCategoryPlaylistsAsync(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 OnPostGetCategoryPlaylistsAsync is used to get the **playlists** for a **category** by **Category 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 <!-- Browse --> enter the following:

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="GetCategoryPlaylists" method="post">  <input **asp-for**="Value" placeholder="Category Id" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Get Category Playlists  </button>  </form>  </li> |

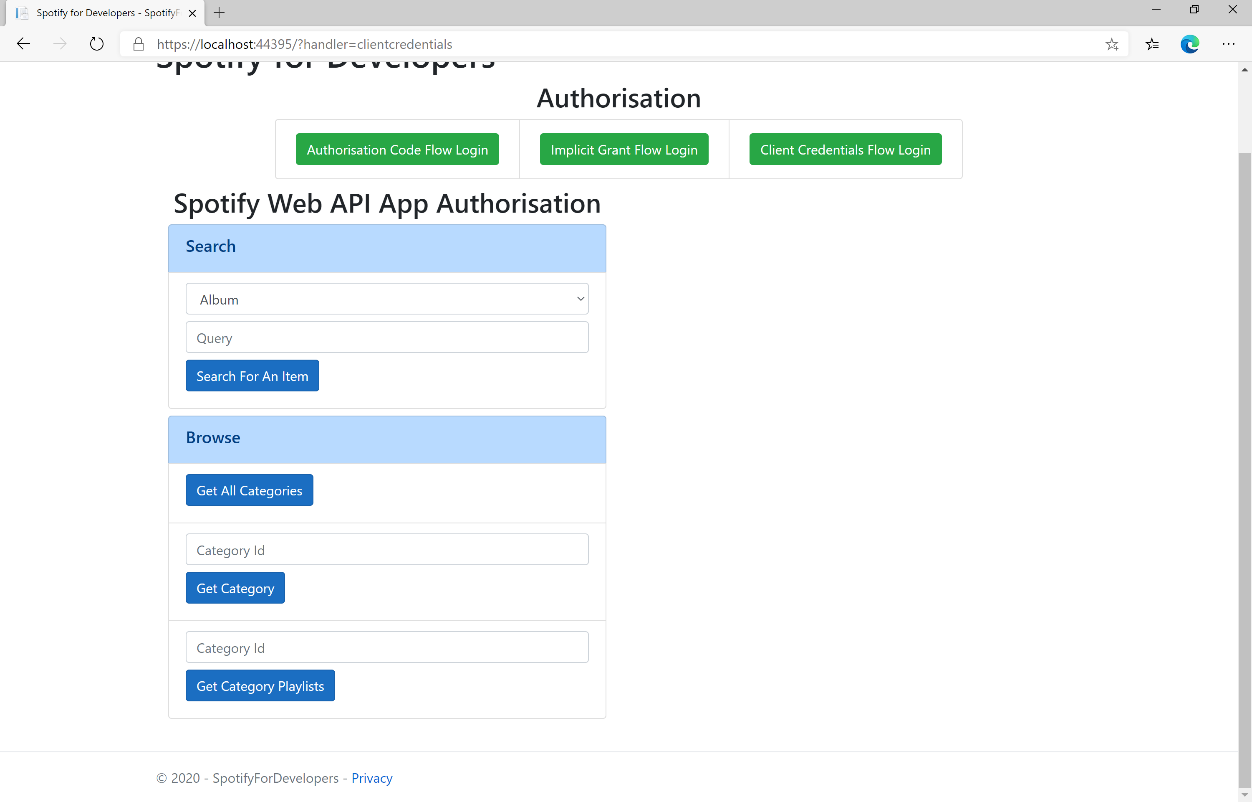
This form will **post** to the **method** for OnPostGetCategoryPlaylistsAsync with the Value as the **Category Id** and will output to the **Results**.

### Step 8

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

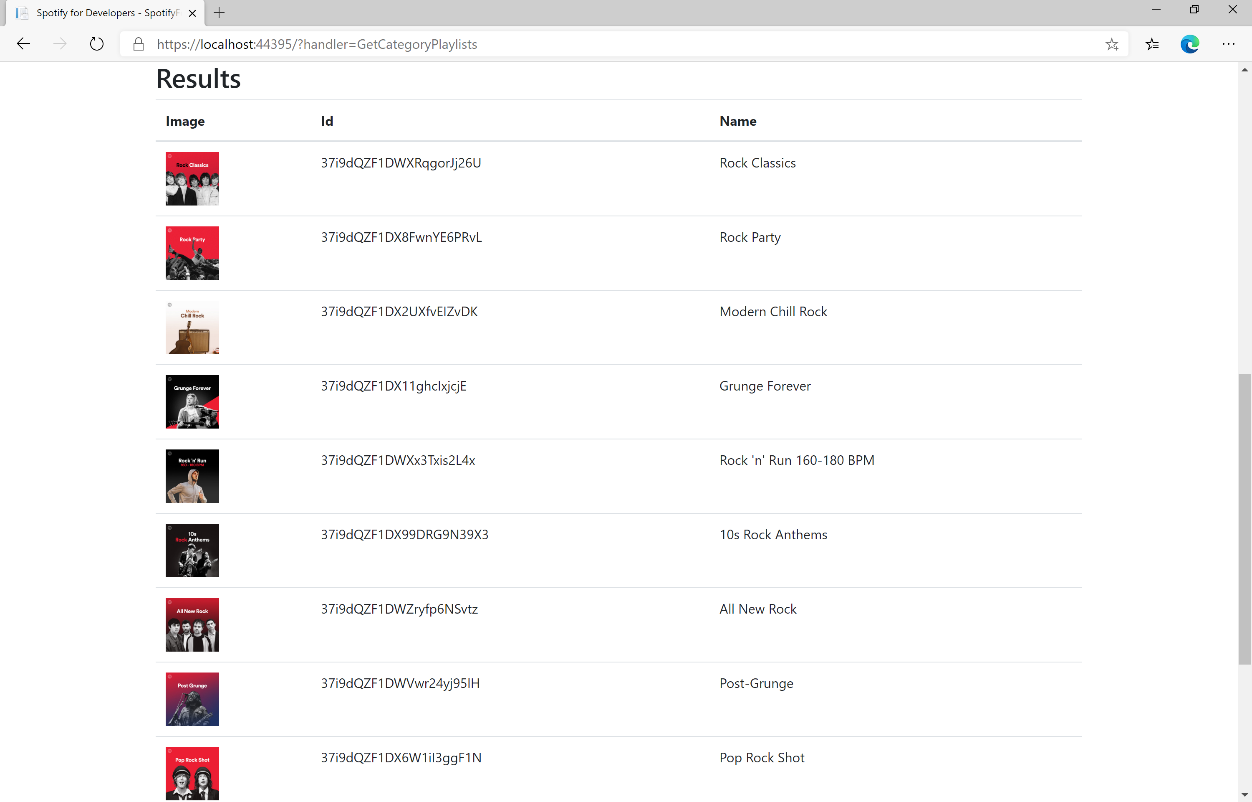
### Step 9

Once the **Web Application** is running and you select **Client Credentials Flow Login** and scroll down you should see something like the following:



### Step 10

You can then enter a **Category Id** from **Get All Categories** and select **Get Category 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 Recommendation Genres

Retrieve a list of available genres seed parameter values for recommendations.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/recommendations/available-genre-seeds | |
| Header | |
| Authorization | Valid Access Token from Spotify Accounts service |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | Array of Genre Seeds |
| 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** and **Search** **&** **Browse** |

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

|  |
| --- |
| public async Task<IActionResult> OnPostGetRecommendationGenresAsync()  {  LoadToken();  var results = await Api.GetRecommendationGenresAsync();  if (results?.Genres != null)  {  Results = results.Genres.Select(result => new Result()  {  Id = result  });  }  return Page(); } |

The **method** for OnPostGetRecommendationGenresAsync is used to get the **genres** for **recoomendations** 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 <!-- Browse --> enter the following:

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="GetRecommendationGenres" method="post">  <button class="btn btn-primary mb-2">  Get Recommendation Genres  </button>  </form> </li> |

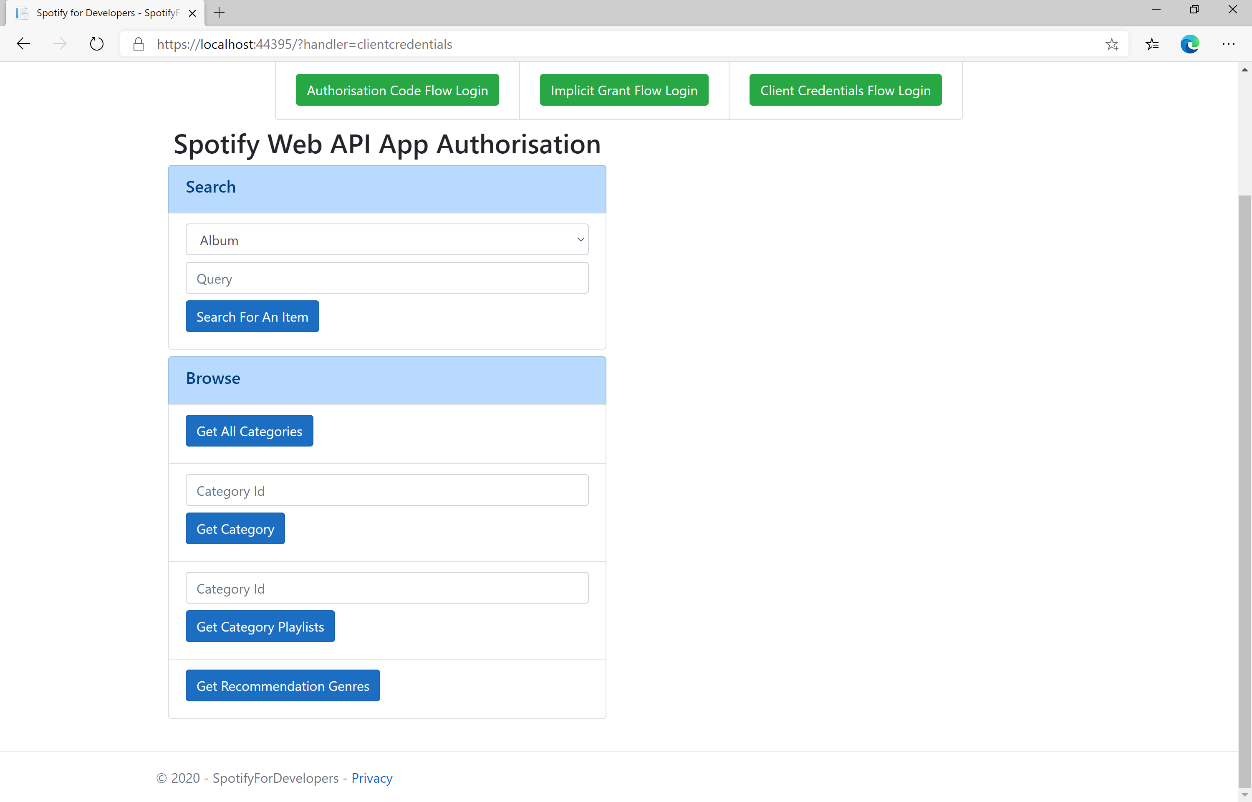
This form will **post** to the **method** for OnPostGetRecommendationGenresAsync and will output to the **Results**.

### Step 8

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

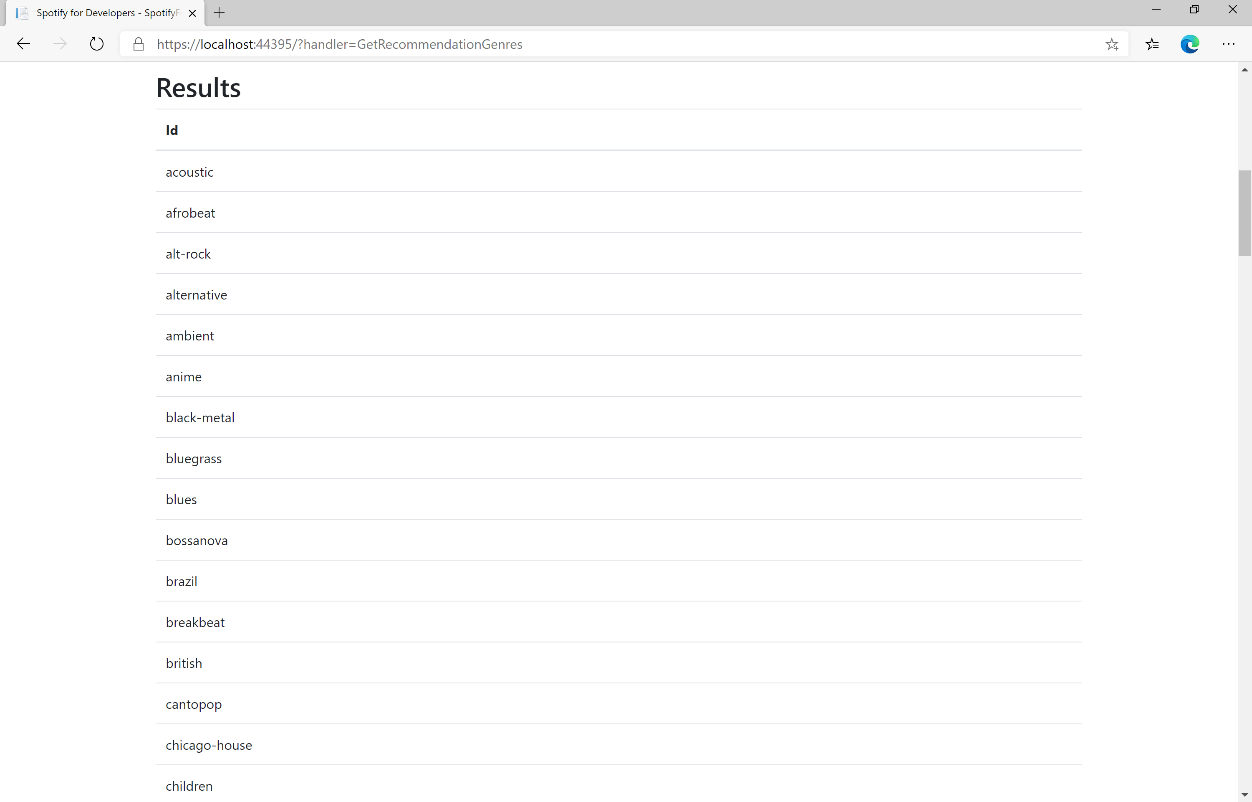
### Step 9

Once the **Web Application** is running and you select **Client Credentials Flow Login** and scroll down you should see something like the following:



### Step 10

You can select **Get Recommendation Genres** 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 Recommendations

Recommendations are generated based on the available information for a given seed entity and matched against similar artists and tracks. If there is enough information about the provided seeds, a list of tracks will be returned together with pool size details.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/recommendations | |
| Header | |
| Authorization | Valid Access Token from Spotify Accounts service |
| Query Parameter | |
| seed\_artists | Comma separated list of Spotify IDs for seed artists. Up to 5 seed values may be provided in any combination of seed\_artists, seed\_tracks and seed\_genres |
| seed\_genres | Comma separated list of any genres in the set of available genre seeds. Up to 5 seed values may be provided in any combination of seed\_artists, seed\_tracks and seed\_genres |
| seed\_tracks | Comma separated list of Spotify IDs for a seed track. Up to 5 seed values may be provided in any combination of seed\_artists, seed\_tracks and seed\_genres |
| limit | Maximum number of results to return |
| market | ISO 3166-1 alpha-2 country code e.g. “GB”. Provide to enable Track Relinking |
| min\_\* | For each Tuneable Track attribute, a hard floor on the selected track attribute’s value can be provided |
| max\_\* | For each Tuneable Track attribute, a hard ceiling on the selected track attribute’s value can be provided |
| target\_\* | For each of the Tuneable Track attributes a target value may be provided. Tracks with the attribute values nearest to the target values will be preferred |

|  |  |
| --- | --- |
| Tuneable Track Attributes | |
| acousticness | A confidence measure from 0.0 to 1.0 of whether the track is acoustic. |
| danceability | Describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity |
| duration\_ms | Duration of the track in milliseconds |
| energy | Measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity |
| instrumentalness | Predicts whether a track contains no vocal |
| key | Key the track is in. Integers map to pitches using standard Pitch Class notation |
| liveness | Detects presence of an audience in the recording |
| loudness | Overall loudness of a track in decibels (dB) |
| mode | Mode indicates the modality of a track (major or minor) |
| popularity | Popularity of track, between 0 and 100, with 100 being the most popular |
| speechiness | Detects the presence of spoken words in a track |
| tempo | Estimated tempo of a track in beats per minute (BPM) |
| time\_signature | Estimated overall time signature of a track |
| valence | Measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | Recommendations Response 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** and **Search** **&** **Browse** |

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

|  |
| --- |
| public async Task<IActionResult> OnPostGetRecommendationsAsync(string value)  {  LoadToken();  var results = await Api.GetRecommendationsAsync(  seedGenres: new List<string> { value });  if (results?.Tracks != null)  {  Results = results.Tracks.Select(result => new Result()  {  Id = result.Id,  Name = result.Name,  Inner = new Result()  {  Id = result?.Artists?.FirstOrDefault()?.Id,  Name = result?.Artists?.FirstOrDefault()?.Name  }  });  }  return Page();  } |

The **method** for OnPostGetRecommendationsAsync is used to get **recommendations** by **genre** 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 <!-- Browse --> enter the following:

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="GetRecommendations" method="post">  <input **asp-for**="Value" placeholder="Genre" class="form-control mb-2" />  <button class="btn btn-primary mb-2">  Get Recommendations  </button>  </form>  </li> |

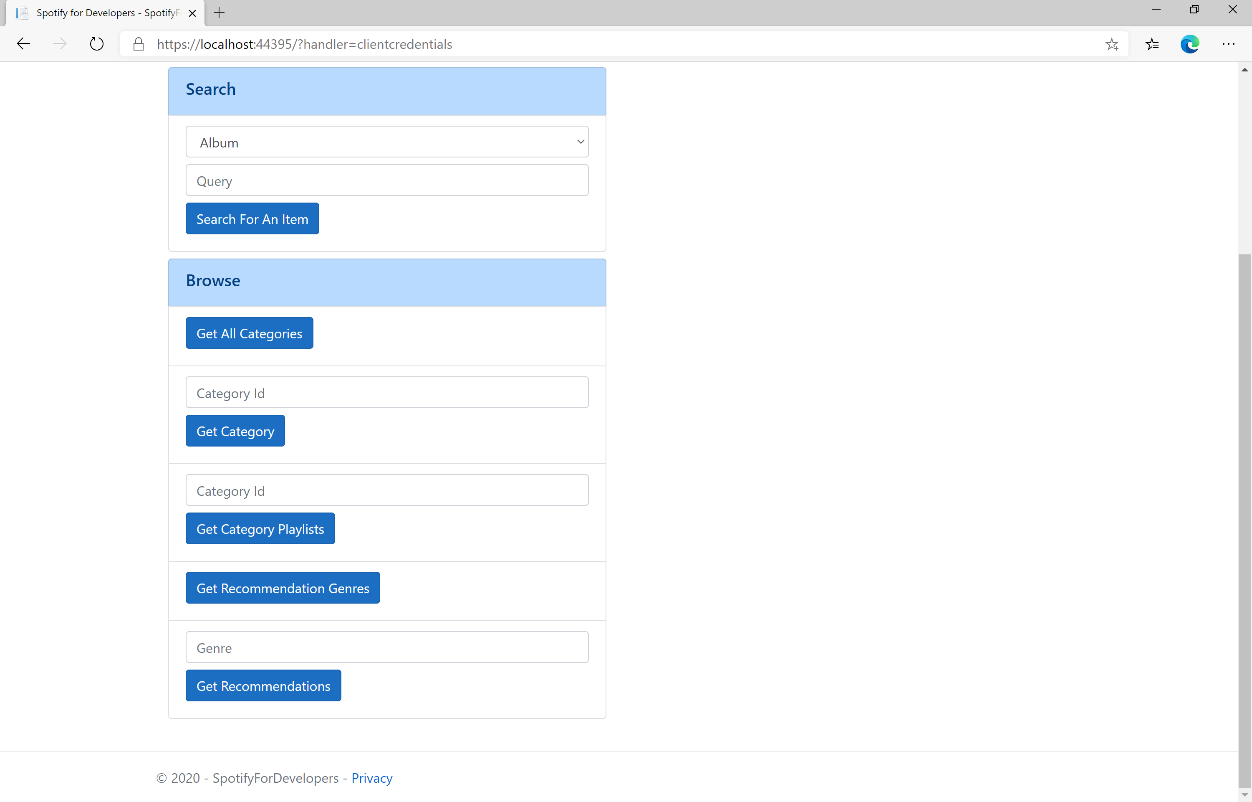
This form will **post** to the **method** for OnPostGetRecommendationsAsync with the Value as the **Genre** and will output to the **Results**.

### Step 8

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

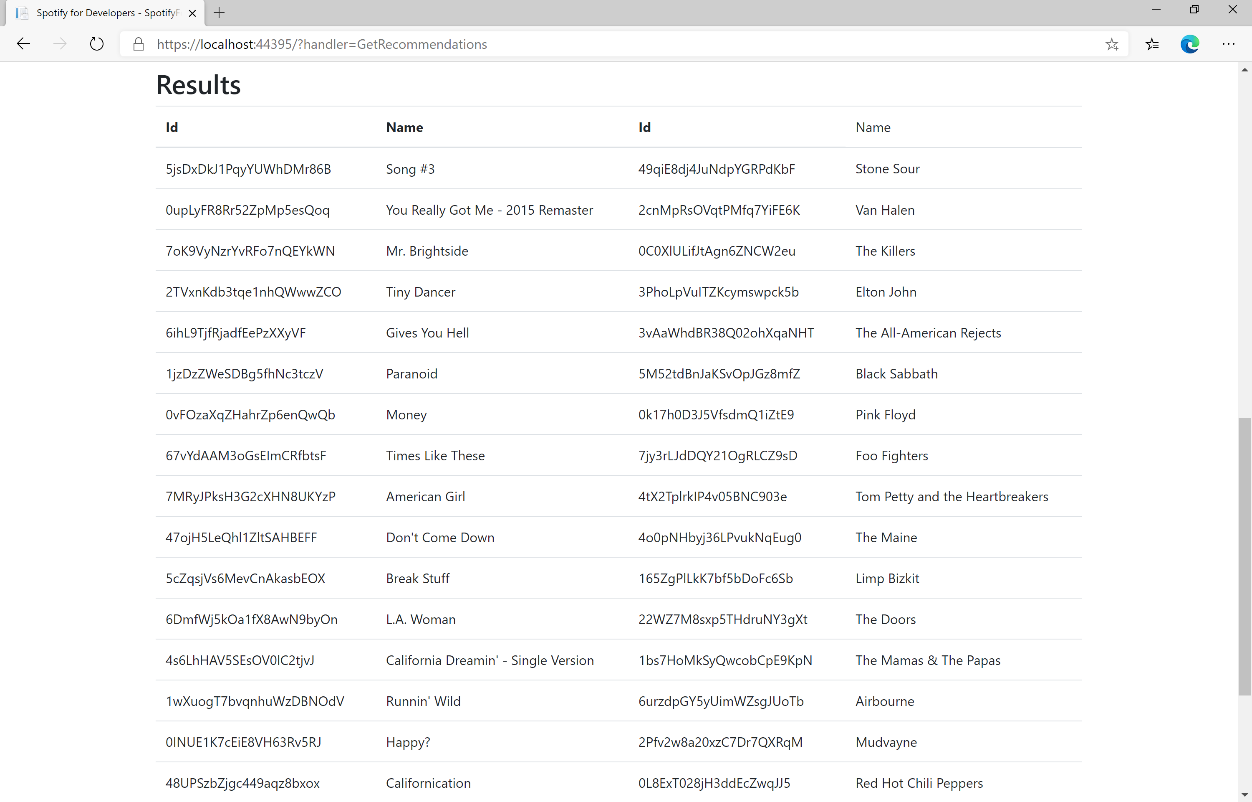
### Step 9

Once the **Web Application** is running and you select **Client Credentials Flow Login** and scroll down you should see something like the following:



### Step 10

You can then enter a **Genre** from **Get Recommendation Genres** and select **Get Recommendations** 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 All New Releases

Get a list of new album releases featured in Spotify as shown in the Spotify player’s “Browse” tab.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/browse/new-releases | |
| Header | |
| Authorization | Valid Access Token from Spotify Accounts service |
| Query Parameter | |
| country | ISO 3166-1 alpha-2 country code e.g. “GB”. If omitted globally relevant items returned |
| limit | Maximum number of results to return |
| offset | Index of first result to return |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | Array of Simplified Album 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** and **Search** **&** **Browse** |

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

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

The **method** for OnPostGetAllNewReleasesAsync is used to get **all new releases** by country 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 <!-- Browse --> enter the following:

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="GetAllNewReleases" method="post">  <button class="btn btn-primary mb-2">  Get All New Releases  </button>  </form>  </li> |

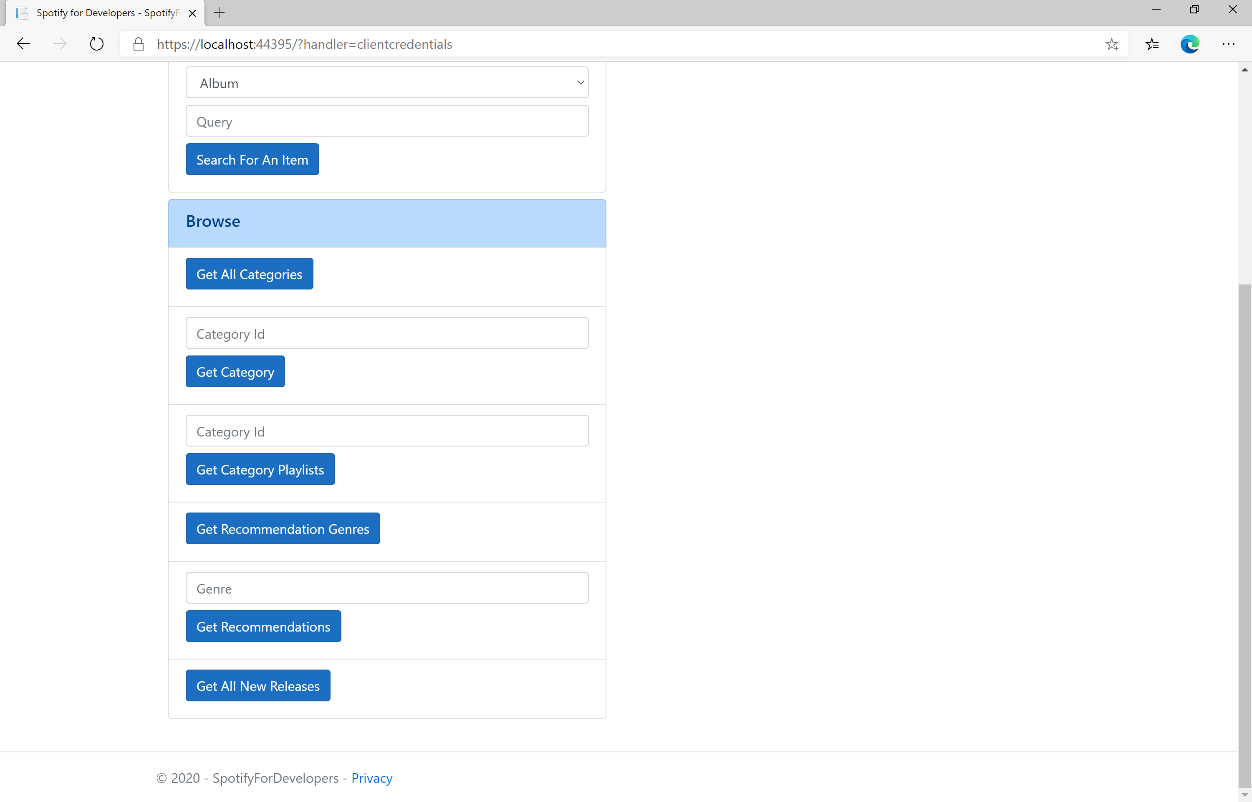
This form will **post** to the **method** for OnPostGetAllNewReleasesAsync and will output to the **Results**.

### Step 8

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

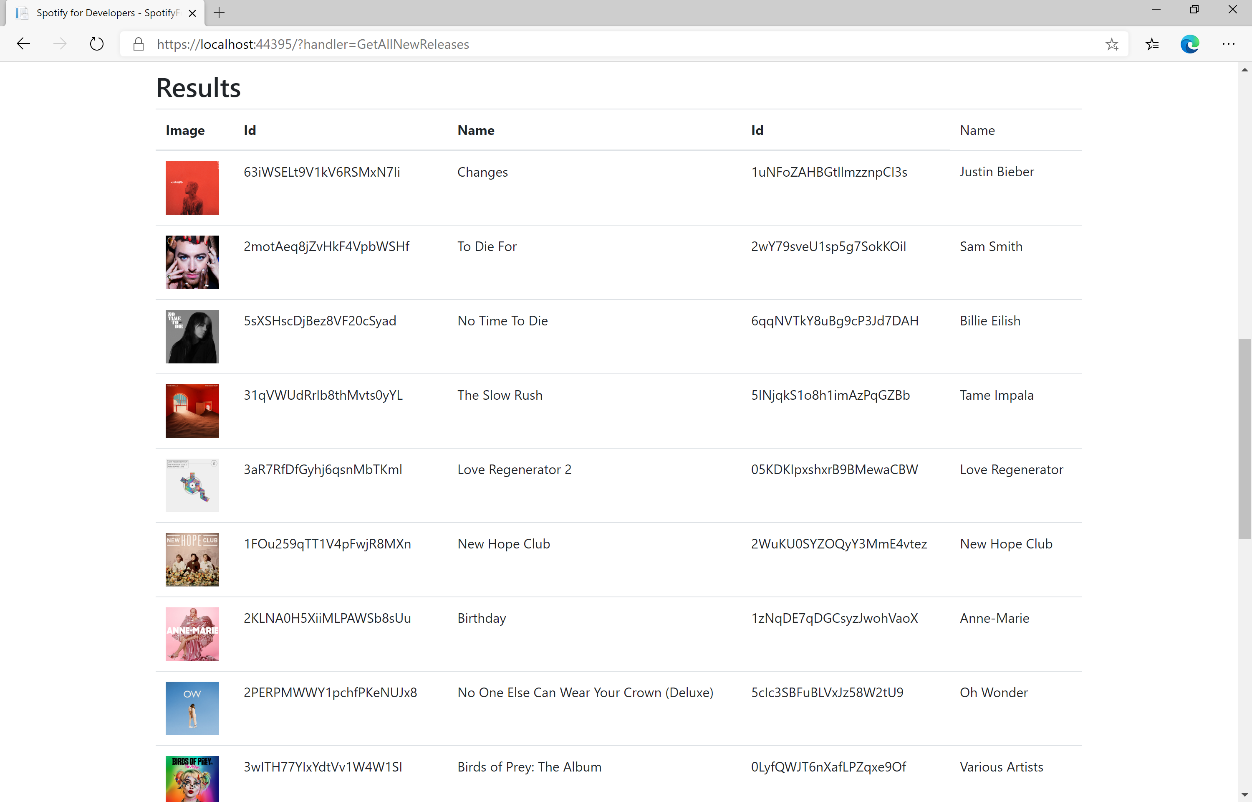
### Step 9

Once the **Web Application** is running and you select **Client Credentials Flow Login** and scroll down you should see something like the following:



### Step 10

You can then select **Get All New Releases** 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 All Featured Playlists

Get a list of Spotify featured playlists as shown in the Spotify player’s “Browse” tab.

|  |  |
| --- | --- |
| GET https://api.spotify.com/v1/browse/featured-playlists | |
| Header | |
| Authorization | Valid Access Token from Spotify Accounts service |
| Query Parameter | |
| country | ISO 3166-1 alpha-2 country code e.g. “GB”. If omitted globally relevant items returned |
| locale | Desired language, consisting of ISO 639-1 language code and ISO 3166-1 alpha-2 country code, joined by an underscore e.g. “en\_GB”. If omitted results returned in American English |
| timestamp | Timestamp in ISO 8601 format: yyyy-MM-ddTHH:mm:ss. Use this parameter to specify the user’s local time to get results tailored for that specific date and time in the day |
| limit | Maximum number of results to return |
| offset | Index of first result to return |

|  |  |
| --- | --- |
| Header | Response |
| Success | |
| HTTP Status 200 OK | Playlists Object contains an Array of Simplified Playlist 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** and **Search** **&** **Browse** |

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

|  |
| --- |
| public async Task<IActionResult> OnPostGetAllFeaturedPlaylistsAsync()  {  LoadToken();  var results = await Api.GetAllFeaturedPlaylistsAsync(country);  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 OnPostGetAllFeaturedPlaylistsAsync is used to get **all featured playlists** by country 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 <!-- Browse --> enter the following:

|  |
| --- |
| <li class="list-group-item">  <form **asp-page-handler**="GetAllFeaturedPlaylists" method="post">  <button class="btn btn-primary mb-2">  Get All Featured Playlists  </button>  </form>  </li> |

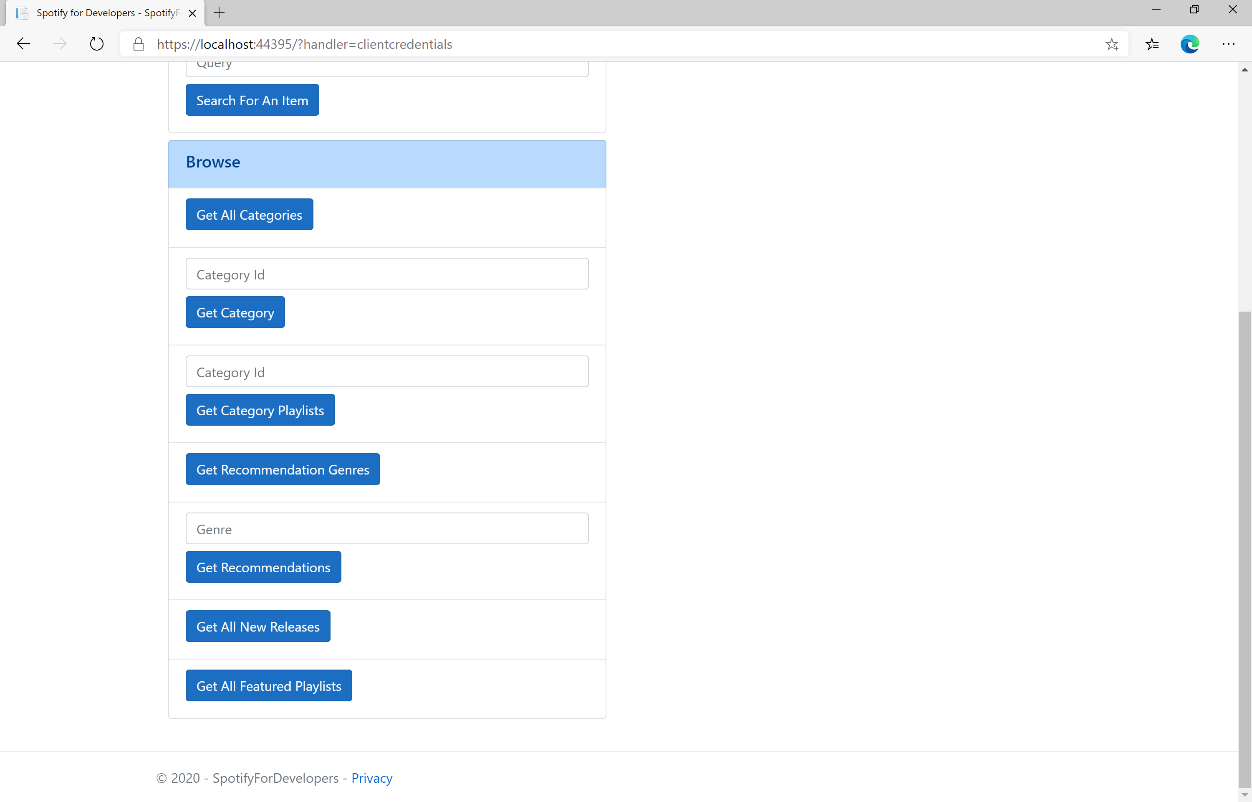
This form will **post** to the **method** for OnPostGetAllFeaturedPlaylistsAsync and will output to the **Results**.

### Step 8

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

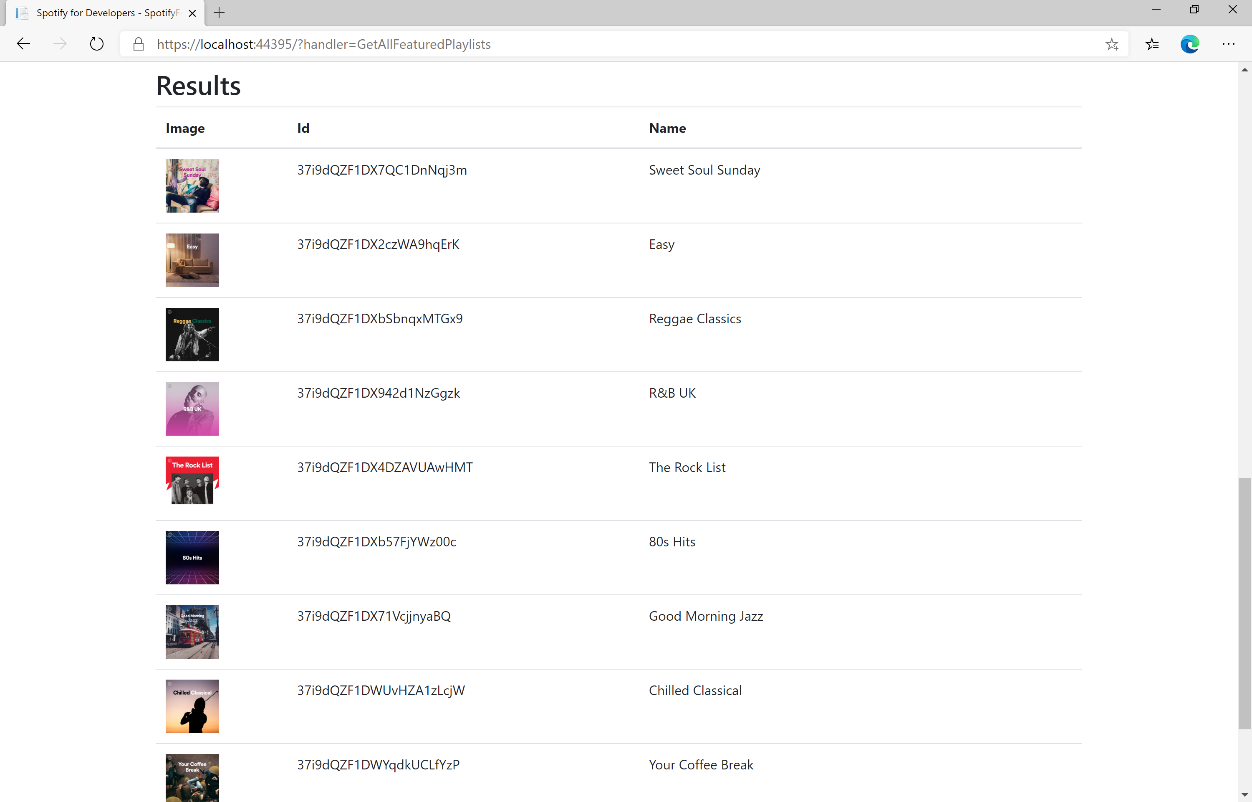
### Step 9

Once the **Web Application** is running and you select **Client Credentials Flow Login** and scroll down you should see something like the following:



### Step 10

You can then select **Get All Featured 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 exit **Visual Studio 2019** by selecting the **Close** button in the top right of the **application** as that completes thispart of the workshop |