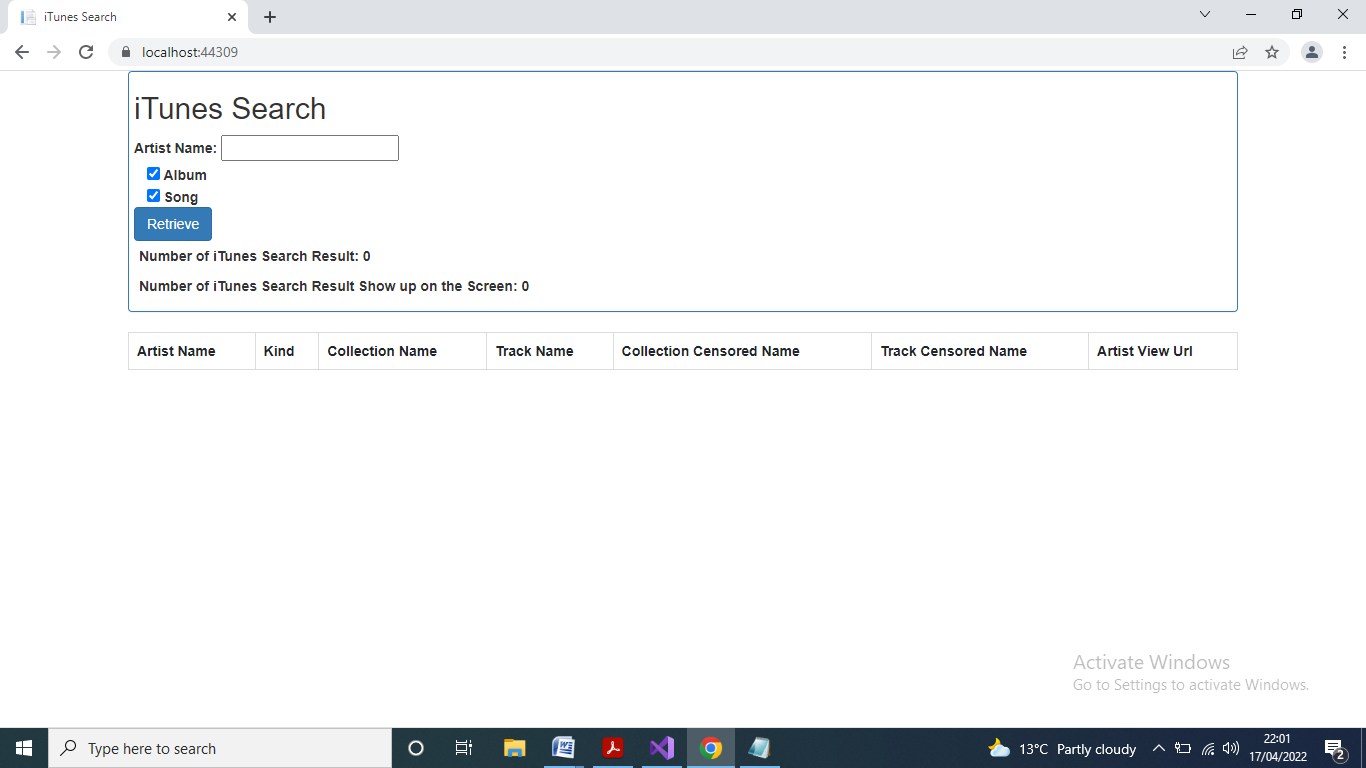
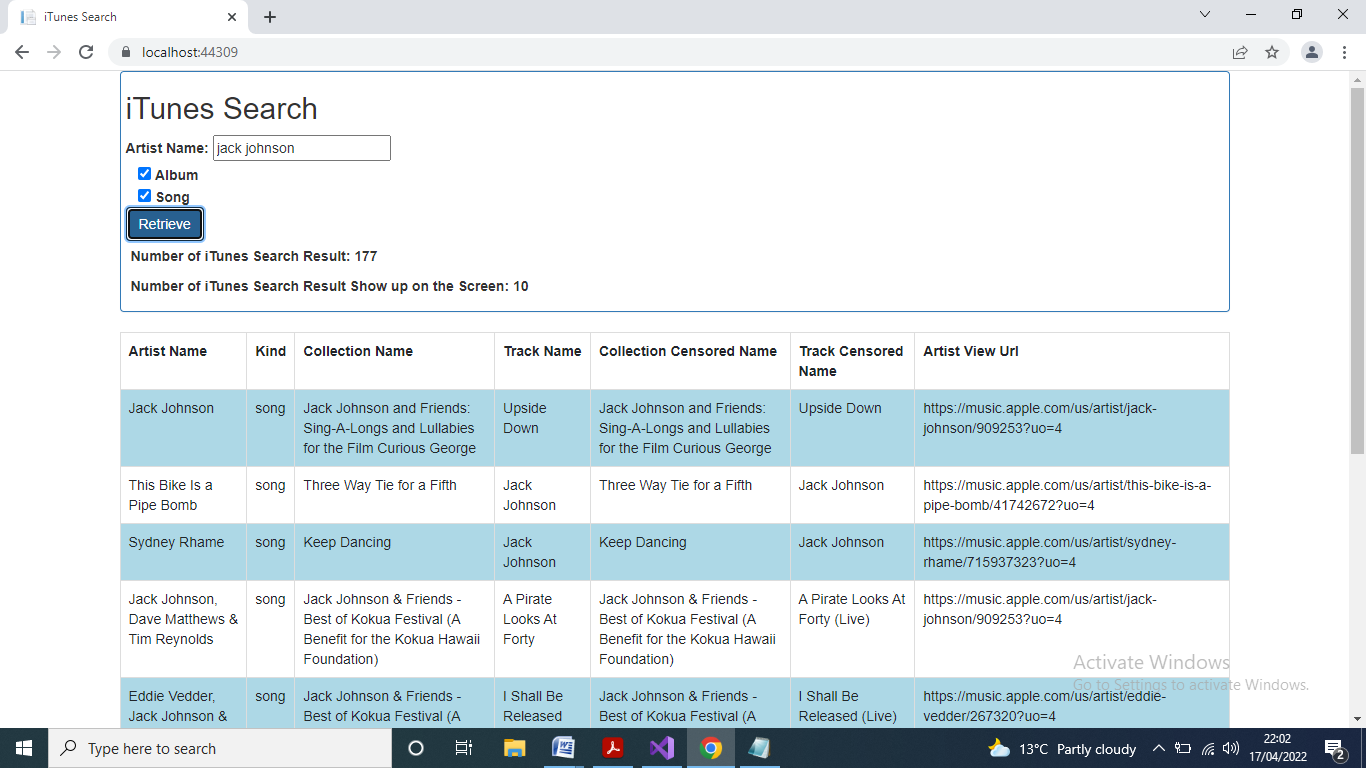
**iTunes Search User Manual**



In the above screen, input any search string in the field “Artist Name:” and tick the necessary fields in the “Album” and “Song” being filtered out. If all of these fields of “Album” and “Song” are un-ticked, it will be filtered out all of its album and song. Then, press the button “Retrieve” to connect the iTunes server.



In the above screen, after pressed the button “Retrieve”, the number of iTunes search result and its being shown up on the screen (by default 10 items) will be updated accordingly. If there is no result being found, the error message will be shown up. During window scrolling down, it will be updated the number of iTunes search result being shown up on the screen by another 10 items.



In the above screen, after scrolling certain distance, it will be shown up the button “Top” being clicked and back to the top of the screen.

**iTunes Search System Architecture**

Based on the unit testing approach, the iTunes API call has details how to connect to their server by different parameters and return json format to discover how its result within this json data. Then, create UI HTML5 and data binding/font-end operations for each component testing in JavaScript/TypeScript.

The project file of “iTunesSearch.zip” is just unzipped to able build and run with Visual Studio 2019.

In the project folder “...\Views\Home”, the file of “Index.cshtml” is its front-end UI HTML5 rendering data for display.

In the project folder “...\Scripts”, the file of “TypeScript.ts” is used to wrap up some classes being called.

In the project folder “...\wwwroot\js”, the file of “Model.js” is used for separating from UI HTML5 from the front-end operations.

In the project folder “...\Controllers” and “...\Models”, the files of “HomeController.cs” and “Retrospective.cs” are respectively just sample of C# coding being used in ASP.NET Core MVC for Web API and business model/session variables. They are not related to this iTunes Search.