Query Completition

by Manuel Papst, 01130635, Technische Universität Graz, Austria

1. Introduction

Scenario: A user starts to type in something…

What should happen next is, that suggestions according to the typed words should appear. In this report i will discribe two approaches for this problem.

2. What problem are you working on?

As already mentioned, a user types in some characters and it should suggest completitions according to the characters already entered. So for example if you type in „flag of Austr“ one of the suggestions should be „flag of Austria“.

3. Why did you choose this approach?

After doing a little research about this topic i decided to try two different approaches.

a.) The first approach i used is an Autosuggest API (Bing Autosuggest API)
b.) The second approach was to download a SQL data dump and work with that data

I have decided to try two different approaches so a could do some comparision between the two.

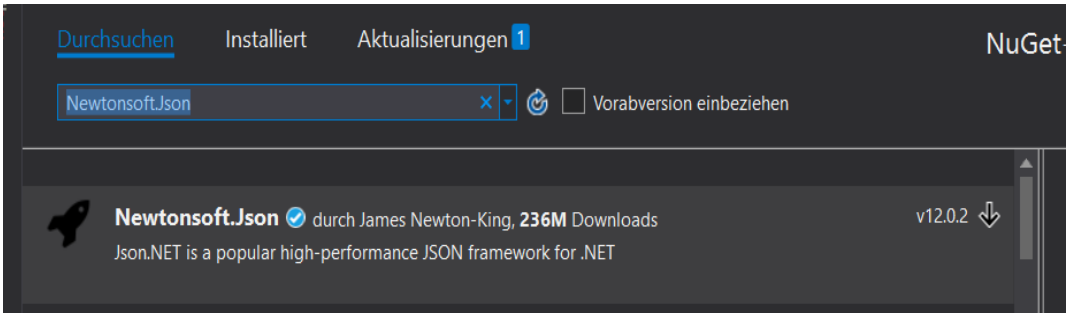
4. How have you tackled the problem?

For both approaches i used C#, .NET and Visual Studio and XAMPP to host my databases.

a.) First i had to get an API key. With that key i could use the API[1]. Then i started a new UWP Project in Visual Studio, added just a Textblock and the Textbox for the queries. I also found a pretty good tutorial how to use this API[2]. When implemented the user could type in some words in the textbox and it would suggest some completitions according to what was typed. When the user starts typing something an HTTP Request is sent to the API which returns an JSON object with the best result.

[1] https://azure.microsoft.com/de-de/try/cognitive-services/?api=autosuggest-api
[2] https://docs.microsoft.com/en-us/azure/cognitive-services/Bing-Autosuggest/quickstarts/csharp

To work with JSON objects i had to add the following package to my Project:



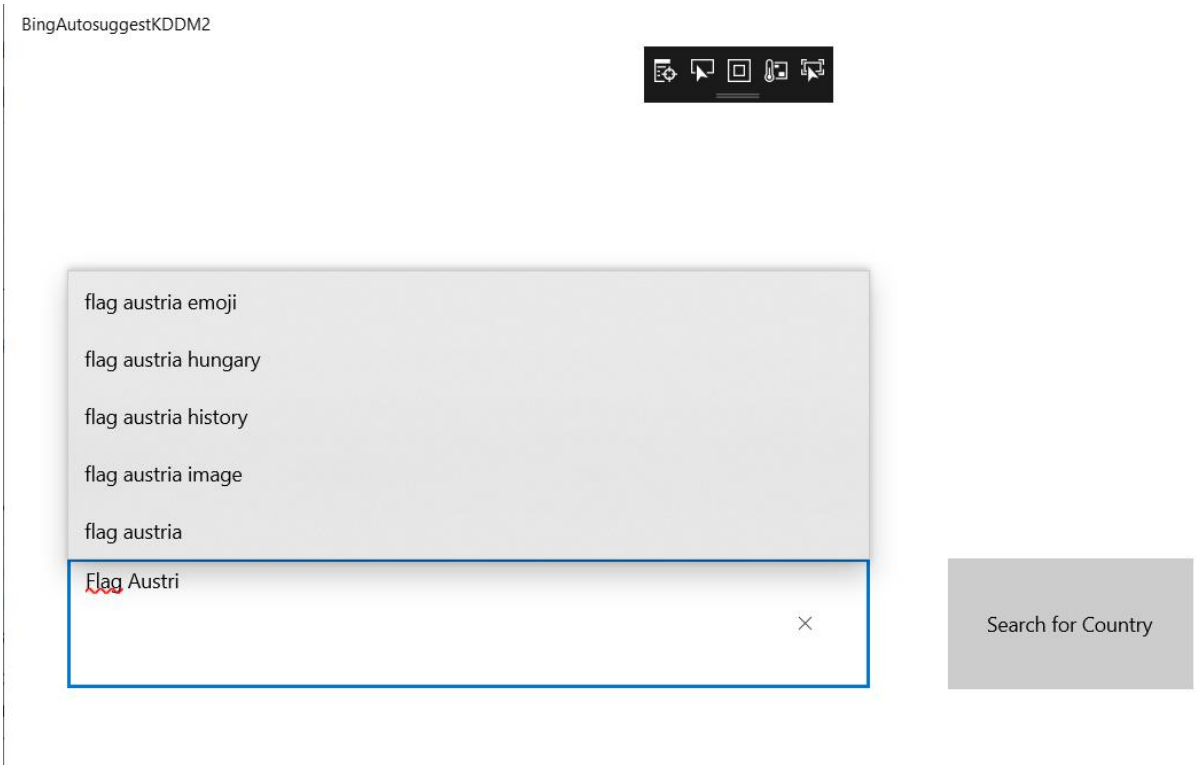
After adding this package i could work with the JSON respones from the API which look like this:


```
{
  "_type": "Suggestions",
  "instrumentation": null,
  "queryContext": {
    "originalQuery": "Flag Austria"
  },
  "suggestionGroups": [
    {
      "name": "Web",
      "searchSuggestions": [
        {
          "url": "https://www.bing.com/search?q=flag+austria&FORM=USBAPI",
          "urlPingSuffix": null,
          "displayText": "flag austria",
          "query": "flag austria",
          "searchKind": "WebSearch"
        },
        {
          "url": "https://www.bing.com/search?q=flag+australia&FORM=USBAPI",
          "urlPingSuffix": null,
          "displayText": "flag australia",
          "query": "flag australia",
          "searchKind": "WebSearch"
        },
```


As you can see it takes the original query and does a websearch and returns some suggestion groups with the best results. E.g. when you type in „Flag of aust“ it would suggest you „Flag of Austria“ or „Flag of Australia“ etc.

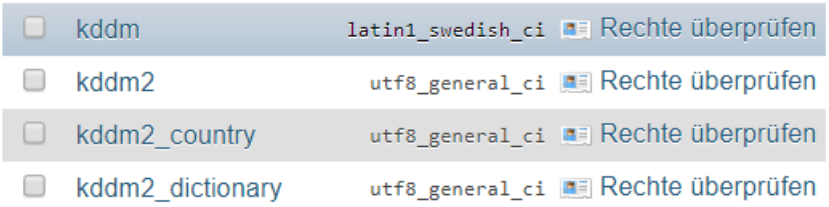
This already worked fine, but what to do with all that? So i decided that i would add a country search which directly lead you tot he wikipedia page of that country.
To accomplish that i downloaded a country sql dump[3] and now everytime a user types in a query it checks in the database if a country name is in that query and if so i takes you directly to the wikipedia page oft hat country.

[3] http://27.org/isocountrylist/



So if you would query „Flag of Austria“ and click the button it would open your Chrome browser and take you to the wikipedia page of Austria.

b.) For the second approach i wanted to download a large database dump and work with that. So i find a SQL dump of patreon on this site[4] and downloaded it. The dump had about 15GB. I started importing the database on my machine and it took about 36 hours to finish.



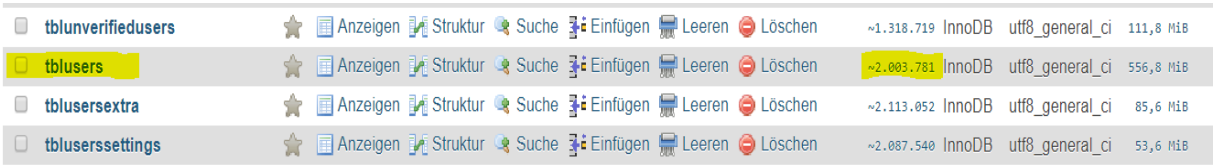
I loaded the patreon dump into the „kddm2“ database and the „kddm2_country“ database was used for the first approach.

After that i look through all the data and thought about what i should to with it. The database included full names, adresses etc. I had to think about data protection aswell so i decided i would just go with usernames (online aliases) for this one.

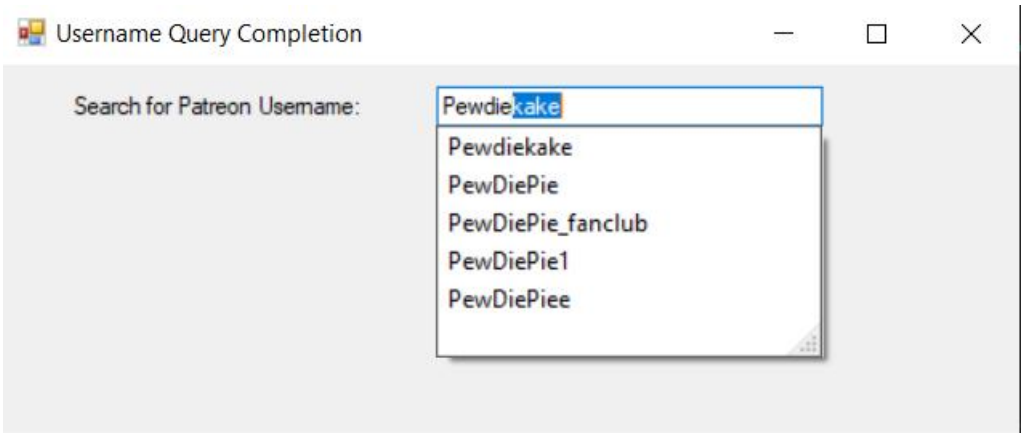
I created a new Project, added the searchbox and everything and then implemented a preloading from the database to that searchbox. So when the

[4] https://databases.today/search.php

application starts it preloads all usernames from the table (which has about 2 Mio. entries)



And when the user starts typing it would suggest completions for existing usernames.
Like when you type in „PewDie“ one of the suggestions would obviously be „PewDiePie“.



And when you hit Enter it will take you to the patreon page of the user you searched for.

5. What are your evaluation results (is the problem solved)?

Both approaches solved the problem but in different ways. The first one was based on an Websearch API which is more dynamic as it changes with time and you could search for everything and i would suggest completions for it.

The second approach also solved also solved the problem but in a more static why. When the application was started the data was loaded, so if the database would change and a username was added to is it wouldn’t suggest it. But even though it has about 2 Mio. entries the application started pretty fast.

6. What have you learnt (new insights)?

This was the first time i worked with sql dumps and i read a lot about search engines and search APIs like the one i used. I improved my knowledge in working with search APIs and also working with large DB files. Also it was the first time i actually had tot hink about data protection.

7. Did something unexpected happen?

The most unexpected thing that happened was that it took my machine about 36 hours to import the patreon SQL dump. And also that i would actually think about data protection was really unexpected for me.

8. Would the solution apply to other scenarios (and how well)?

Both approaches would apply to other scenarios. E.g. the websearch API could easily be used in own website or projects to help the user get suggestions. Another option would be to take other APIs for special purposes and implement them in your projects.

The second approach can also be used for projects where data does not changed constantly. If data is added or deleted from the database within seconds this approach wouldn’t work so well.

9. Conclusion

After reading about different approaches i decided to try out two different ones myself. The websearch API was easy to understand and implement and can be used for further tasks.

And i also wanted to work with large DB files so i decided to take a different approach aswell. It was really unexpected that i took so long to import the database to my machine and also how hard it is to find the right data in all this. Also when working with large amounts of (personal) data it is important to think about data protection aswell.