**Task: Implement a Search Functionality using Elasticsearch**

**Objective:**

Develop a .NET web application that integrates with Elasticsearch to provide a search functionality. The data to be searched will be stored in an SQL database. The goal of this task is to assess your skills in .NET backend development, including your ability to work with databases, data indexing, and search engines.

**Requirements:**

1. **Setup SQL Database:**
   * Create an SQL database with a table named **Products**.
   * The **Products** table should have the following columns:
     + **Id** (int, primary key)
     + **Name** (nvarchar(255))
     + **Description** (nvarchar(max))
     + **Price** (decimal)
     + **Category** (nvarchar(100))
2. **Data Population:**
   * Populate the **Products** table with at least 100 sample records.
3. **Elasticsearch Integration:**
   * Setup an Elasticsearch instance (local or cloud-based).
   * Create an index in Elasticsearch for the **Products** data.
4. **Data Indexing:**
   * Develop a .NET console application or a background service that reads data from the SQL database and indexes it into Elasticsearch.
5. **Search API:**
   * Create a .NET Core Web API with an endpoint **/api/search** that accepts a query parameter.
   * The endpoint should search the indexed data in Elasticsearch and return relevant results.
   * The search should be performed on the **ID,Name** and **Description** fields of the **Products**.

**Deliverables:**

1. Source code for the .NET Core web application and the data indexing service.
2. Instructions on how to setup and run the application, including any necessary configuration for Elasticsearch.

**Evaluation Criteria:**

1. **Code Quality:**
   * Clear and maintainable code with appropriate comments and documentation.
   * Proper use of .NET best practices and design patterns.
2. **Functionality:**
   * The application should meet all the specified requirements.
   * The search functionality should be efficient and return relevant results.
3. **Use of Elasticsearch:**
   * Proper indexing of data in Elasticsearch.
   * Efficient search queries that utilize Elasticsearch's capabilities.
4. **Performance:**
   * The application should handle a reasonable load and return search results quickly.

Please Procced for Next page for Solution.

**Package information:**

Package contains following:

1. Data Indexing Console App
   1. Used for creating index and placing data from SQL to ElasticSearch Service
2. Api used to perform following
   1. Fetching data based on ID, Name , Description from Elastic search Service
   2. Inserting Data into SQL
      1. On each insert api will update the index in elastic search
   3. Update Data into SQL
   4. Delete Data from SQL
3. DB scripts
   1. Execute the DB script file that will create the schema and dump 100 records in DB.

**Configurations**

There are two ways to configure the elastic search service

* On Prem
* Cloud based instance

The solution I have implemented in Cloud based instance.

However below are the on Prem instance details.

**On Prem:**

Download the elasticsearch service extract the file and go to bin folder and execute the Elasticsearch.bat

Once service executed open the config folder open elasticsearch.yml file in notepad and update the values of following attributes.

xpack.security.enabled: false

xpack.security.enrollment.enabled: false

close the console and start the elasticsearch.bat again.

**Cloud Based instance configuration:( solution has been implemented)**

<https://www.elastic.co/>

configure the account and publish the elastic search instance (No Action Required)

system will provide the endpoints and authentication key. (No Action Required)

We need to set these values in postman or in our code configuration. (No Action Required)

Get the db scripts from below path and execute on server

Open appsettings.json and update the DB Connection string.

Get the code from Below Path:

* Data Indexing and syncing Console app
  + <https://github.com/Fahhad-m/DataIndexingService.git>
* API Search
  + <https://github.com/Fahhad-m/ElasticSearchSimplias.git>
* DB Scripts
  + <https://github.com/Fahhad-m/DataIndexingService/tree/66a1aa64e360dfaf54980cf8f58d0bab3b58bb76/DbSchema>
* Read me Document
  + https://github.com/Fahhad-m/Readme-ElasticSearch.git

**execute the setup**

DataIndexingService.exe from DataIndexingService-> Bin->DataIndexingService.exe

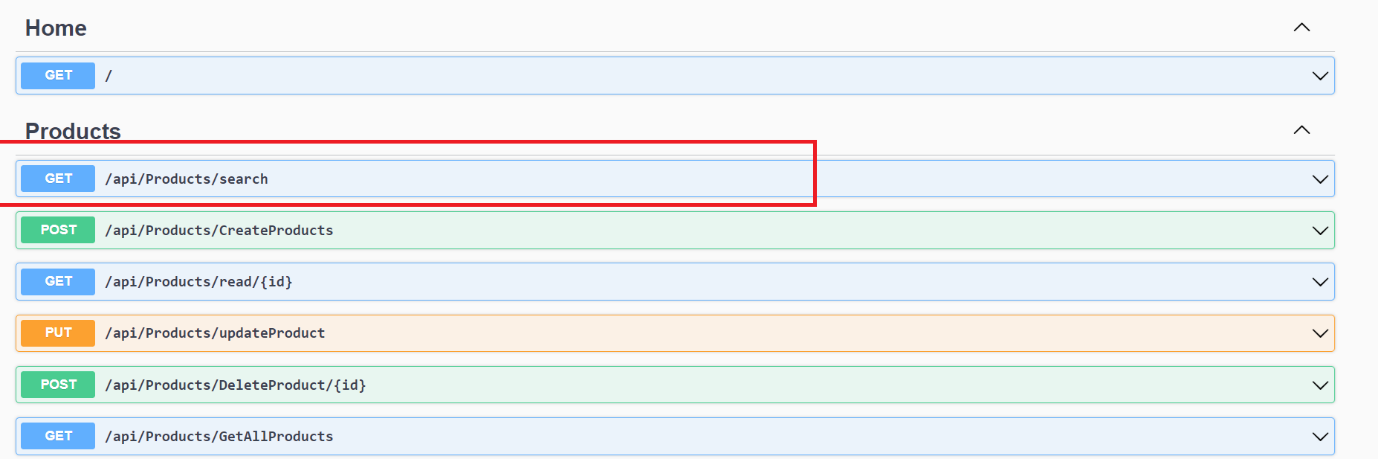
Or by opening solution and press ctrl+f5

In downloaded solution go to ..\DataIndexingService\DbSchema and open the scripts.sql in SQL management studio to create the table and oracle the data.

**Run Search API**

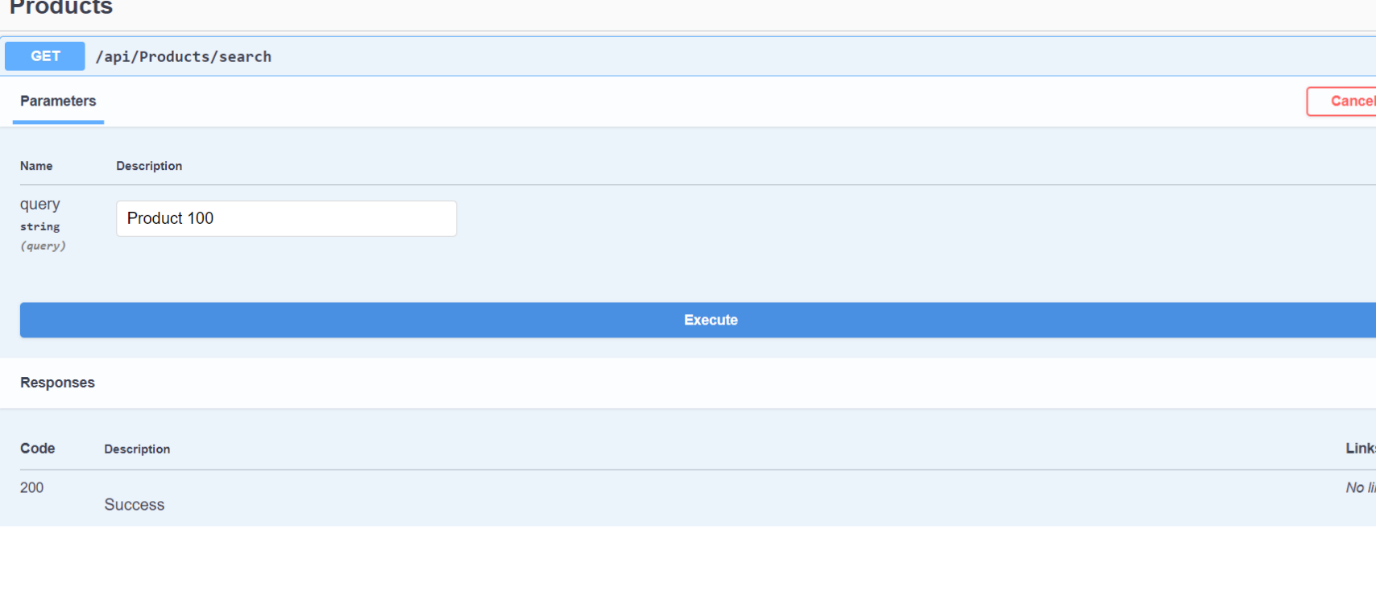
by opening solution and press ctrl+f5

go to the following section



Open the Get Product Search API

Search the “product 1” which is the name of product



System will search the data based on ID,index and descripton.

Via postman:

**Get Products from elastic search**

https://07220b9a9eee41038832df72f3571834.us-central1.gcp.cloud.es.io/products/\_search?q=Product 100

token information

Type:API Key

Key : Authorization

Value: ApiKey MFdmci1JOEJLUHRQSmRaYXNVVms6anh0MUw0MENUd2UtNW9zbFJYNFZPQQ==

Add to: Header

Create Product API:

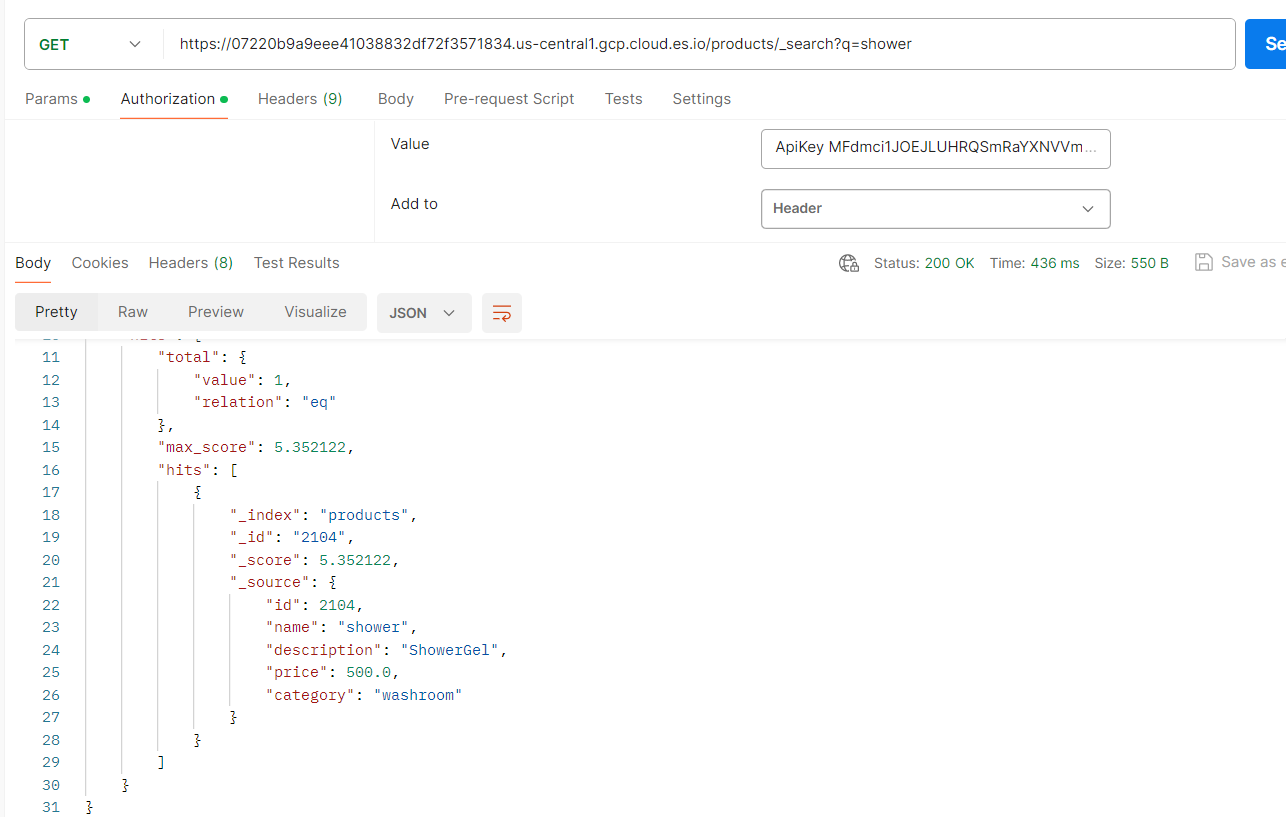
**https://localhost:44373/api/Products/CreateProducts**

call type : Post

Auth Auth is off.

..

Results- Post Man



Result via .net Core API

