**Custom Search with Elastic Search**

* Custom search allows you to modify and enhance the way search works to meet specific needs.
* Elastic Search is commonly used with Magento 2 to handle complex search functions quickly and accurately

**When to Use Each**

* **Use Elastic Search:** if you need a reliable, scalable, and out-of-the-box solution for fast searching across standard catalog attributes.
* **Use Custom Search:** if your store requires special filters, unique business rules, or specific search logic that Elastic Search doesn’t support directly.

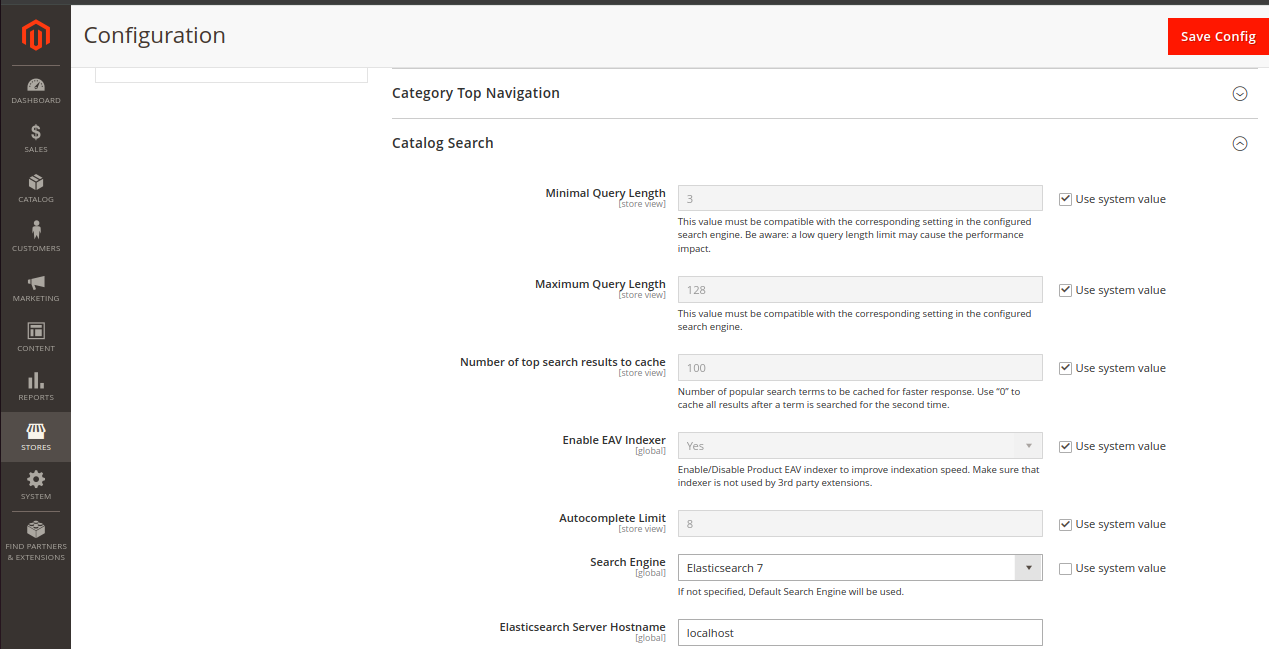
**Custom Search with Elastic Search - Workflow**

1. **Data Indexing**: Elastic Search creates a structured index of all product data, allowing fast retrieval of results when a search query is made.
2. **Query Modification**: When a custom search is run, Magento sends the query to Elastic Search, where any custom filters (e.g., by specific attribute like "brand") are applied before the results are fetched.
3. **Fast Results**: Elastic Search returns result very quickly, making it highly efficient even for large catalogs, as it’s optimized for searching big datasets.

**Custom Search without ElasticSearch - Workflow**

1. **Database Querying**: Without ElasticSearch, Magento queries its internal MySQL database directly for search requests, which may slow down with larger catalogs.
2. **Custom Code Logic**: Custom filters and search rules are added directly into Magento’s database query logic, requiring more effort to maintain and customize.
3. **Limited Performance**: As the database grows, search speeds decrease since Magento’s database isn’t optimized for complex, large-scale searches like ElasticSearch.

|  |  |  |
| --- | --- | --- |
| Feature | Custom Search with ElasticSearch | Custom Search without ElasticSearch |
|  |  |  |
| Speed | Very fast due to optimized indexing and retrieval | Slower, as it relies on direct database queries |
| Scalability | Efficient for large catalogs; handles high data volume well | Performance decreases as catalog size grows |
| Customization Complexity | Easier to apply custom filters within ElasticSearch queries | More complex to modify database queries directly |



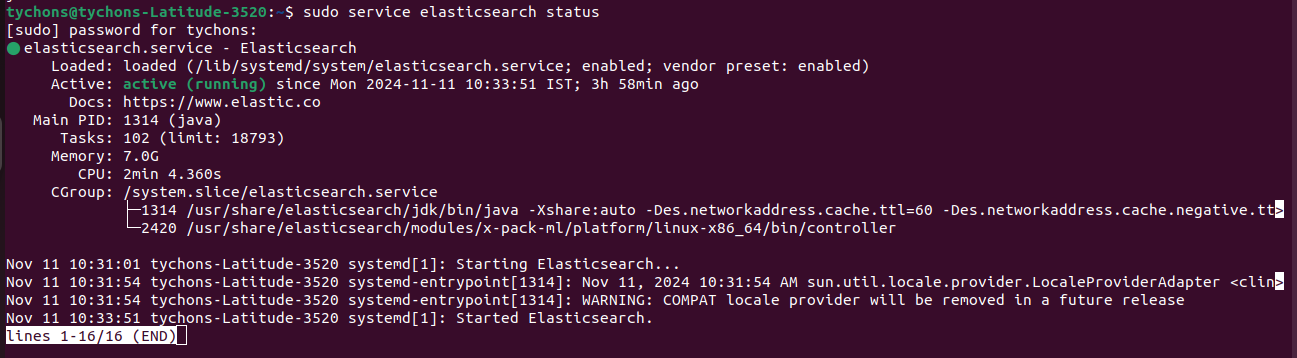
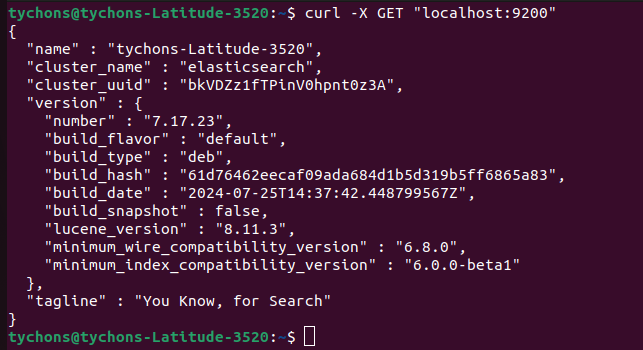
In search engine is elastic search

**Setting Up Custom Search with Elasticsearch in Magento 2**

Elasticsearch is a powerful, fast search engine that improves search performance and provides more relevant search results for your Magento 2 store. By setting up **Elasticsearch** for custom search functionality, you can make your search much faster and more flexible.

Here’s a simple step-by-step guide to setting up **Custom Search with Elasticsearch** in Magento 2, with a real-time example.

**1. Check Elasticsearch Version**

Before configuring Elasticsearch, ensure it is installed and running. You can check the version of Elasticsearch by running the following command in your terminal:  
  


**Install Elasticsearch (if not already installed)**

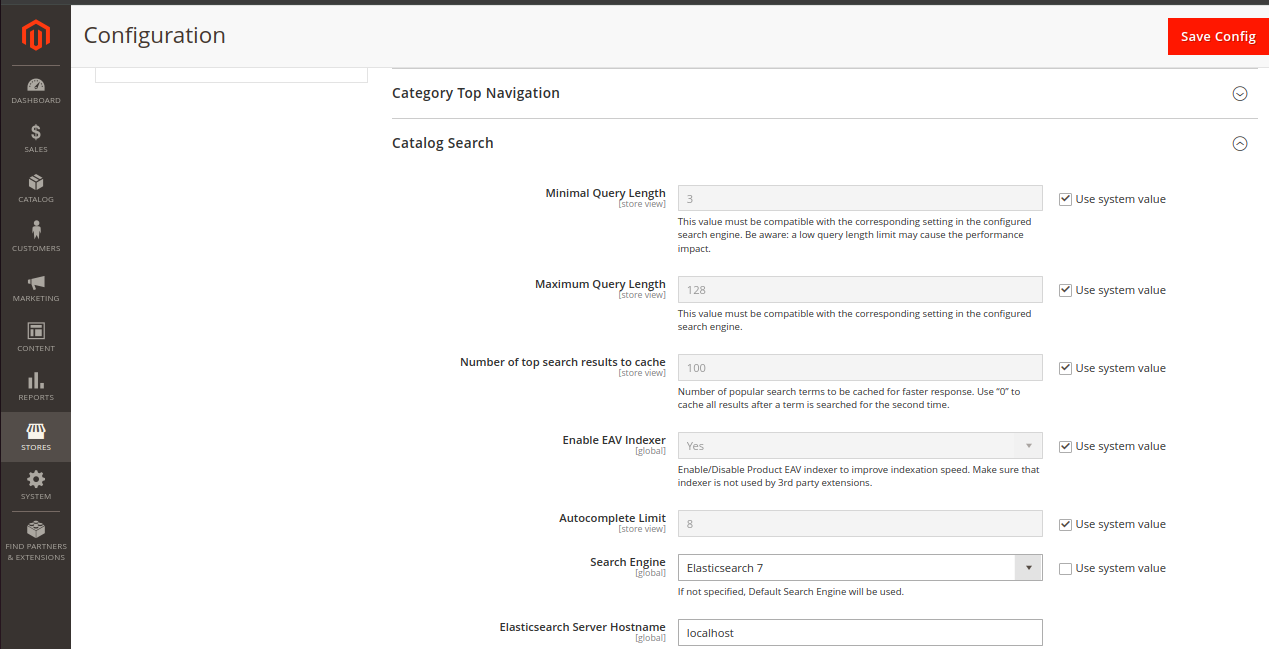
sudo apt update

sudo apt install elasticsearch

sudo systemctl start elasticsearch

**Configure Elasticsearch in Magento 2 Admin Panel**

* **Login to Magento Admin Panel**.
* Go to **Stores > Configuration > Catalog > Catalog Search**.
* In the **Search Engine** dropdown, select **Elasticsearch**.
* Set the **Elasticsearch Server Hostname** to localhost (if you're running Elasticsearch on the same server) or the IP address of your Elasticsearch server.
* Set the **Port** to 9200 (default for Elasticsearch).
* You can enable **Search Suggestions** if you want suggestions to appear as customers type their queries in the search box.
* Click **Save Config**.



**Index Product Data in Elasticsearch**

Once you've selected Elasticsearch as your search engine, Magento will index your product data into Elasticsearch. To do this:

1. Go to **System > Tools > Index Management**.
2. Select **Reindex Data**. This will trigger Magento to update the Elasticsearch index with your product information.

**Set Up Searchable Product Attributes**

To improve your search results, make sure the correct product attributes are indexed for search. For example, if you want to make **Product Name** and **SKU** searchable, ensure those attributes are included in the search index.

Here’s how to configure searchable product attributes:

1. Go to **Stores > Attributes > Product**.
2. Select the attribute you want to include in search (for example, **Name**, **SKU**, or **Description**).
3. Make sure the **Use in Search** option is set to **Yes**.
4. Click **Save Attribute**.

**How Mapping Works with Elasticsearch in Magento 2**

* In **Magento 2**, **Elasticsearch** is used as the search engine, and **mapping** is crucial to define how product attributes (fields) are indexed and searched.
* Mapping defines the structure of the data in Elasticsearch and helps Elasticsearch understand how each attribute of your product should be treated during search and filtering operations.

**What is Mapping in Elasticsearch?**

**Mapping** is like a schema definition in a database. It tells Elasticsearch how to index and store fields of data so they can be efficiently searched.

* **Field Type**: Specifies whether a field should be treated as a **text**, **keyword**, **integer**, **date**, etc.
* **Indexing Rules**: Controls whether the field should be searchable (indexed), and how the data in the field should be analyzed (for example, for full-text search).
* **Search Behavior**: Determines how Elasticsearch should handle different types of searches—whether the field should support full-text search, exact matching, or be used for filtering and sorting

**Mapping in Magento 2**

* Magento 2 uses Elasticsearch for various purposes, such as searching for products in the catalog, filtering products, and performing full-text searches.
* The **mapping configuration** in Elasticsearch for Magento 2 determines how product attributes are indexed and how they interact with the search process.
* Magento creates **default mappings** for common product attributes (e.g., name, sku, price) when Elasticsearch is set up as the search engine.
* If you have custom attributes (e.g., color, brand, bag), you need to ensure that these are correctly mapped to work with Elasticsearch.

**Types of Fields in Elasticsearch Mapping**

When defining mappings in Elasticsearch, there are several types of fields you can use:

1. **Text**: Used for full-text search (e.g., product names or descriptions).
2. **Keyword**: Used for exact match or filtering (e.g., product categories, product status).
3. **Integer/Long**: Used for numerical values like prices or quantities.
4. **Date**: Used for date-based attributes like release dates.
5. **Boolean**: For true/false values (e.g., is the product visible?).

Magento automatically uses these field types for default attributes like name, sku, and price, but you may need to explicitly define these mappings for custom attributes.

**Custom Attributes and Mapping in Magento 2**

When you create custom product attributes in Magento 2 (for example, bag, brand, color), Elasticsearch needs to know how to index and search those attributes. This is done through **mapping**.

**Example: Mapping a Custom Attribute bag**

Let's say you want to create a custom product attribute called **"bag"** to filter products by their type (e.g., "backpack", "handbag").

In Elasticsearch, this attribute needs to be mapped properly to ensure correct indexing and efficient searching. Here’s an example of how the mapping might look

{

"properties": {

"bag": {

"type": "keyword" // Use "keyword" for exact matches or filtering

}

}

}

* **type: keyword**: This tells Elasticsearch to treat the bag attribute as a **keyword** field, which is ideal for filtering (e.g., selecting products where bag = backpack).
* **type: text**: If you want the attribute to support full-text search (e.g., searching for the word "leather bag"), you could use text instead of keyword.

**How Magento 2 Handles Elasticsearch Mapping**

Magento 2 automatically handles the mapping for default attributes when Elasticsearch is set as the search engine. However, for custom attributes, you may need to adjust or ensure proper mapping.

* **Product Attribute Configuration**: In Magento Admin, you define product attributes (like color, size, bag, etc.). You can set attributes to be **used in search** or **used for filtering**.
* **Mapping Custom Attributes**: Custom attributes are mapped in Elasticsearch automatically, but you may need to customize this mapping if you need specific behavior. For example, you can change the mapping for custom attributes like bag to be **exact match** (using keyword) or **full-text search** (using text).

**How Elasticsearch Mapping Affects Search Behavior**

1. **Exact Matching**: When you map an attribute as keyword, Elasticsearch performs exact matches for filtering and faceting. For example, searching for a product with bag = backpack will return products with the bag attribute exactly matching "backpack".
2. **Full-Text Search**: When you map an attribute as text, Elasticsearch can perform **full-text search** on that field. This is useful for attributes like description or name where the search should look for matches to parts of the text.
3. **Faster Filtering**: If you map attributes that you filter by (e.g., price, brand, bag) as keyword, Elasticsearch can quickly filter results based on those exact values.

In Magento 2, the **Exact Matching** option (using **keyword mapping**) is the most frequently used.

**How Data is fetched from elastic search**

* Data is fetched from Elasticsearch in Magento 2 by sending a structured query to Elasticsearch, which then retrieves matching documents based on indexed attributes and returns the results to Magento.

**Main use of mapping**

* to define how product attributes are indexed in Elasticsearch, optimizing search accuracy and filtering speed for different attribute types.

**Mapping with Elasticsearch**

* defines specific data types and indexing rules for each attribute to enhance search and filtering

**Mapping without Elasticsearch**

* relies on MySQL, which lacks advanced search optimization and typically results in slower, less accurate searches for large catalogs.