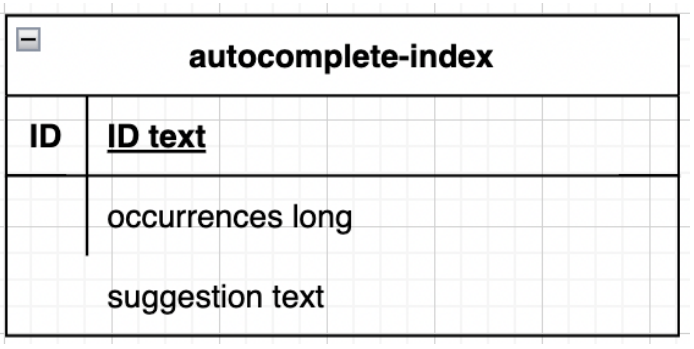
# Definition of an information retrieval system using Elasticsearch

## Document collaborators and roles

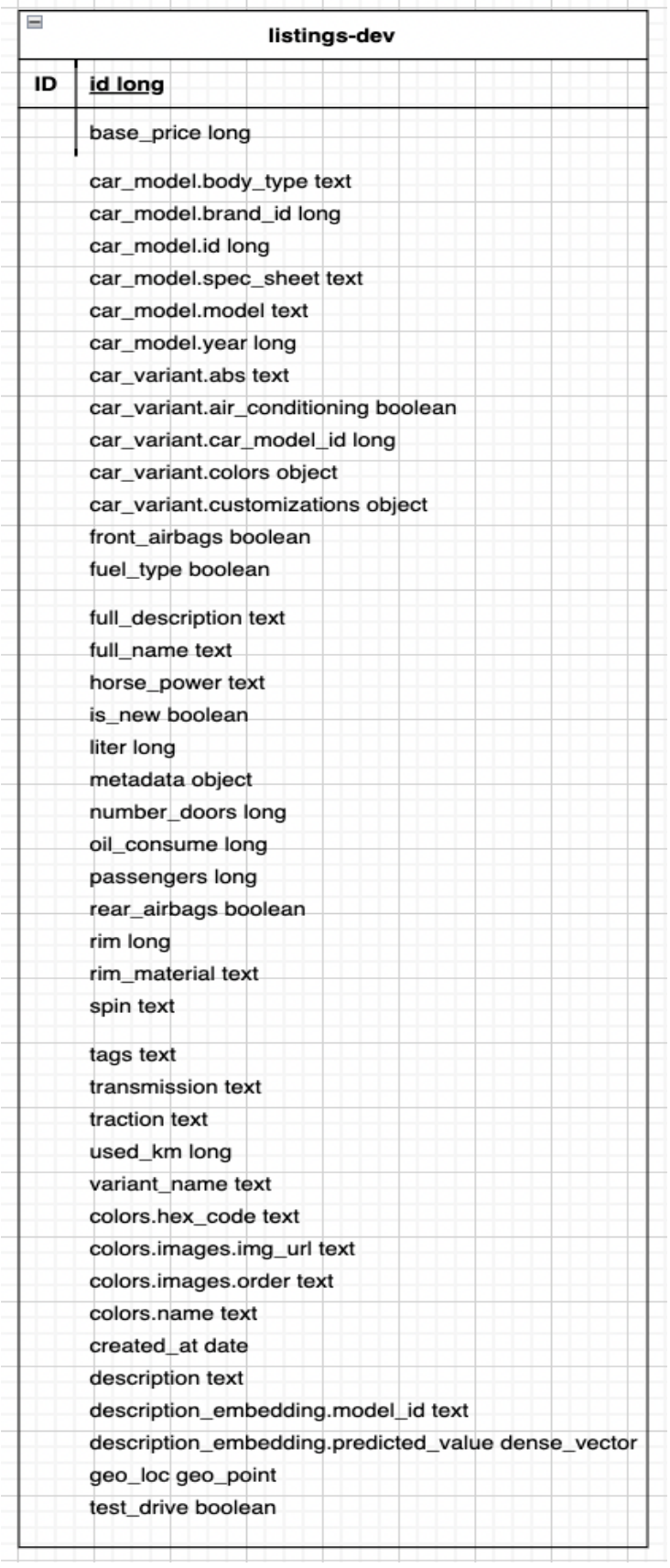
| Member | Role |
|---|---|
| Miguel Arriaga Velasco | Developer in the search team. Collaborated in the creation of the python script to create the indexes. |
| Stephan Guingor Falcón | Natural Language Processing module chief. |
| Pablo Rocha Ojeda | Developer in the search team. Collaborated in the creation of indexes. |
| Emilio Sibaja Villarreal | Collaborator |
| Ariadne Álvarez Reyes | Collaborator |
| Octavio Fenollosa Sosa | Collaborator |

## UML diagram

### Autocomplete-Index



## Listings-Dev



# Indexes

## Index: "listings-dev"

### Settings breakdown

#### Filters

- "lowercase": Converts all text to lowercase. This allows case-insensitive searches, ensuring that queries match indexed data regardless of capitalization.
- "asciifolding": Performs ASCII folding, which means it replaces accented characters and other special characters with their ASCII equivalents. This helps improve search accuracy by treating accented and non-accented characters as equivalent.
- "stop": Removes common words that are considered insignificant for search purposes, such as articles, prepositions, and conjunctions. These words are often excluded to reduce the index size and improve search performance.
- "custom_stemmer": Applies advanced stemming algorithms to reduce words to their base or root form. Stemming allows search queries to match different forms of the same word, such as singular and plural forms, verb conjugations, etc. The "custom_stemmer" likely refers to a specific implementation of a stemming algorithm.
    - "type": "snowball": This setting indicates the stemming algorithm used for the index. "Snowball" is a widely used stemming algorithm that supports multiple languages. It is designed to reduce words to their base form based on language-specific rules. In this case, the algorithm is configured for the Spanish language.
    - "language": "Spanish": Specifies the language of the indexed content. Elasticsearch uses language-specific analyzers to process text data, such as tokenizing, stemming, and applying language-specific rules. In this case, the index is configured for Spanish language content.

#### Replicas

- 3 replicas: Replicas are additional copies of index shards that provide redundancy and improve search performance by distributing data across multiple nodes. In this case, the index has three replicas, meaning each shard has three copies.

#### Shards

- 1 shard: Shards are smaller units that distribute the index across multiple nodes in a cluster, allowing for scalability and parallel processing. In this case, the index has only one shard, meaning all the data is stored on a single node.

Overall, these settings configure the "listings-dev" index to support Spanish language content with various text processing filters, including lowercase conversion, ASCII folding, stop word removal, and advanced stemming using the Snowball algorithm. The index has one shard and three replicas for redundancy and performance optimization.

**Properties of the documents in the "listings-dev" index**

- “id" : The ID of the listing
- "base_price": The starting price for the car being sold
- "car_model.body_type" :
- "car_model.brand_id" : The id of the brand selling the car
- "car_model.id" : A unique identifier for the car model
- "car_model.spec_sheet" : The specification of the car in text form
- "car_model.model" : The model of the car being sold
- "car_model.year": The manufacturing year of the car.
- "car_variant.abs" : A boolean representing true if the car has an abs system
- "car_variant.air_conditioning" : A boolean representing true if the car has air conditioning
- "car_variant.car_model_id" : A unique identifier for the car model
- "car variant.colors": An array of colors the car is available in
- "car variant.customizations" : An object containing the extra features the variant may modify
- "front_airbags" : A boolean representing true if the car has air bags
- "fuel_type" : Type of fuel that the car uses
- "full description" : The description of the car being sold
- "full_name" : The full name of the car.
- "horse_power": The horsepower of the car
- "is_new" : Boolean that represents the car is new or semi-new
- "liter" : Capacity in liters for fuel
- "metadata" : Object to store any additional information of the car being sold
- "number_doors" : Number of doors in the car
- "oil_consume" : Amount of oil consumed
- "passengers" : Number of passengers the car can have
- "rear_airbags" : A boolean representing true if the car has rear bags
- "rim" : Size of the rim
- "rim material" : Material of the rim.
- "spin" : A unique identifier used to retrieve images for 3D view
- "tags" : Array of values associated with the listing
- "transmission" : The transmission type of the car.
- "traction" : Traction of the car
- "used_km" : Number of km the car has go through
- "variant name" : The name of the car variant.
- "colors.hex_code": Hex code of the color
- "colors.images.img_url" : image url of car with the specified color
- "colors.images.order" : Order of the color
- "colors.name" : Name of the color
- "created at": The time when the listing was created
- "description": The description of the car being sold

- “description_embedding.model_id” : Id of the model that was used to create the text embeddings
- "description_embedding.predicted_value": A dense vector representing the text embedding for the listing description. This property is used to calculate similarity between documents using the cosine similarity metric.
- “geo_loc”: A field of type "geo_point" that stores the geographic location of the listing.
- “test_drive”: Boolean representing if the car is available for a test drive

## Index: "autocomplete"

**Properties of the documents in the "autocomplete" index**

- “suggestion”: The text value of the autocomplete suggestion, the field is of type “search_as_you_type” to optimize queries.
- “occurrences”: A numeric value that indicates the number of occurrences of the suggestion, in order to prioritize the most used suggestions.
- “id”: An md5 hash of the query

# Search overview

To create the best search experience we implemented a search endpoint that has two modes.

## Neural Search

The results for the users will be based on context, this is also known as semantic search, and we implemented it using text-embeddings in elastic search. The embeddings where generated by a model based on the pretrained model bert

## Keyword Search

This makes use of classic full text search, using elastic search. We use the user's query and full text search in different fields and rank the results based on the importance of the field and number of hits.

For both modes we can add filters on top, like ranges, or exact matches on fields, this is very important as users often want to limit the search results, we can also enable ‘near me‘ searches, which looks for cars that are being sold in less than a specific distance. ( 1000KM )

We also created an autocomplete endpoint, so that the user is given suggested queries as they type. For this endpoint, we store all the previous queries in the elastic search database, as well as a score for each one of them. The more a query is used, the higher its score will be and it will have more priority in the autocomplete recommendations.

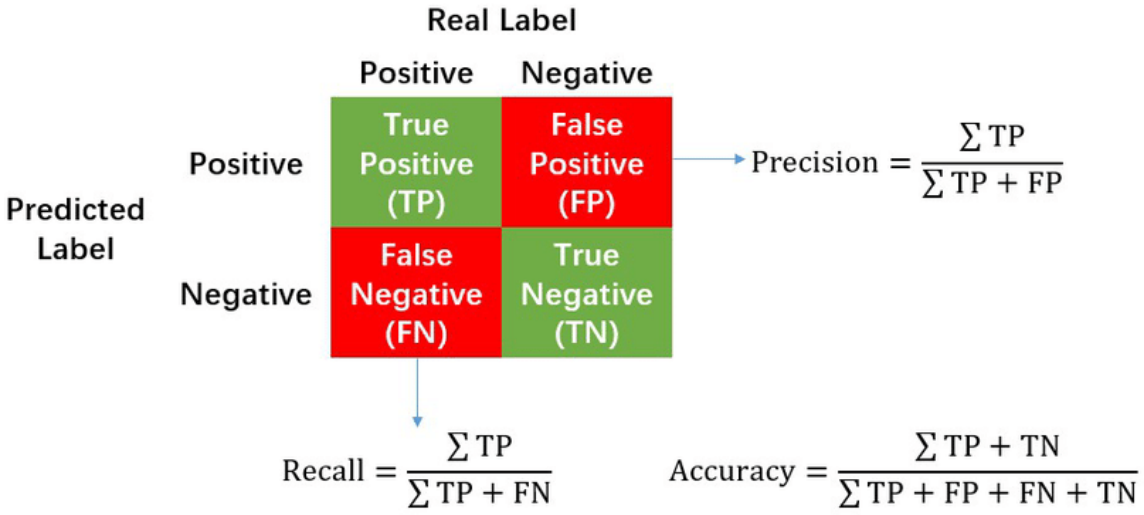
In addition if custom ner is enabled while searching, we will extract different entities from the plain text query. If the service from azure has a confidence score higher than 50% (should be higher but the model is not trained yet to give high scores) then we will add them to our filters and increase the efficiency of our results.

## Precision vs Recall Analysis & Test queries

We decided to obtain the precision and recall values for the keyword search mode, as it is going to be the main search method for the application (hard filters were also not used in the queries to fully test the natural language processing search). To do this, we made 20 test queries, created a precision vs. recall matrix for each one, as well as their overall precision value and recall value, and finally averaged the precision and recall for the 20 queries.

**It is important to keep in mind that we are only evaluating the first 10 results for each of the queries.**

The formulas used are shown below:




Where:

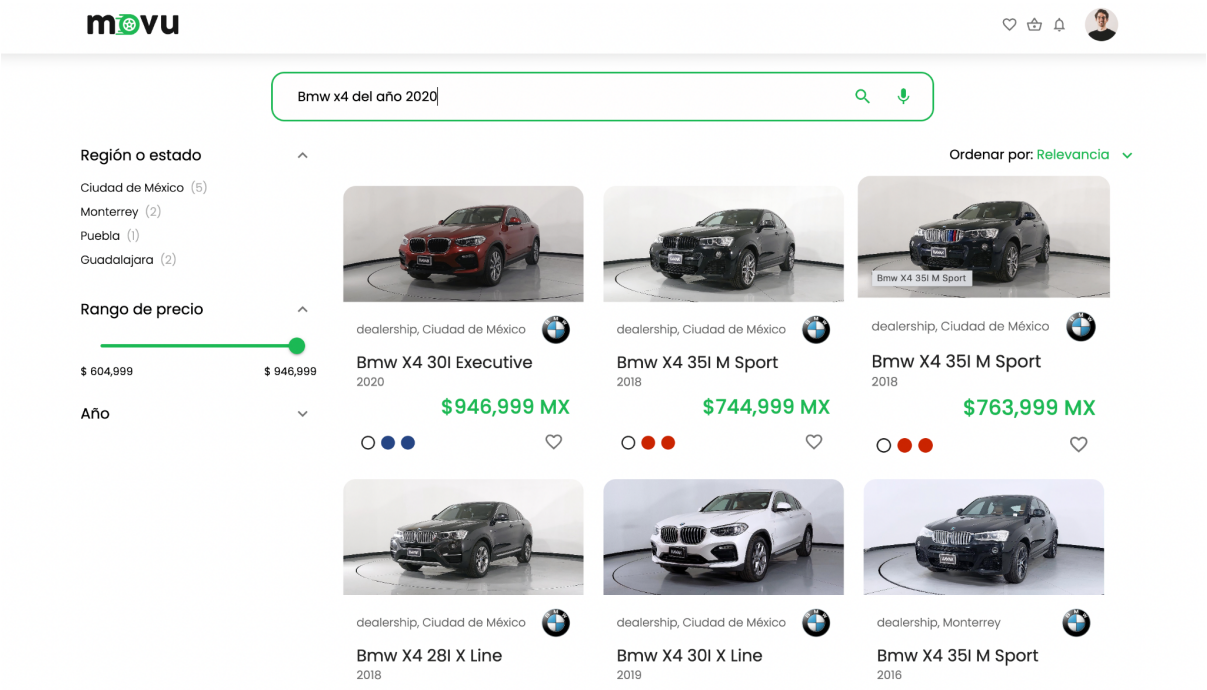
- **True Positive (TP)**: In the context of search results, a true positive represents a relevant document that is correctly retrieved by the search engine.
- **False Positive (FP)**: A false positive represents a non-relevant document that is mistakenly retrieved by the search engine, indicating an incorrect positive prediction.
- **False Negative (FN)**: A false negative represents a relevant document that is not retrieved by the search engine, indicating a missed relevant result.
- **True Negative (TN)**: In the context of search results evaluation, true negatives are not typically considered since they refer to non-relevant documents correctly identified as such (which are not typically tracked in this context).
- **Precision**: Precision measures the proportion of retrieved documents that are actually relevant. It is calculated as TP / (TP + FP). In the context of search results, it represents the accuracy of the retrieved documents.
- **Recall**: Recall measures the proportion of relevant documents that are successfully retrieved by the search engine. It is calculated as TP / (TP + FN). In the context of search results, it represents the comprehensiveness or coverage of the search engine.

# Query #1

**Query:**

Bmw x4 del año 2020

**Screenshot:**




**Precision Value:** 10%

**Recall Value:** 100%

**Results precision vs. recall matrix:**

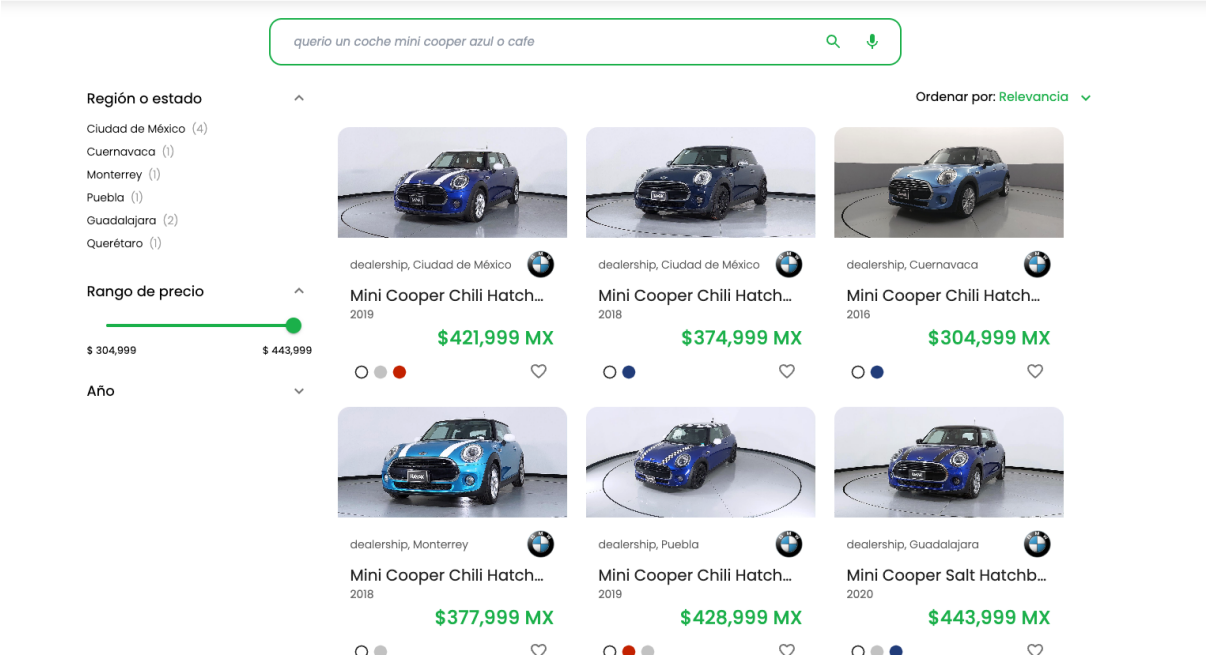
| True Positive (TP) | False Positive (FP) |
|---|---|
| 1 | 9 |
| **False Negative (FN)** | **True Negative (TN)** |
| 0 | 0 |

# Query #2

**Query:**

quiero un coche mini-cooper azul o cafe

**Screenshot:**




**Precision Value:** 50%

**Recall Value:** 45%

**Results precision vs. recall matrix:**

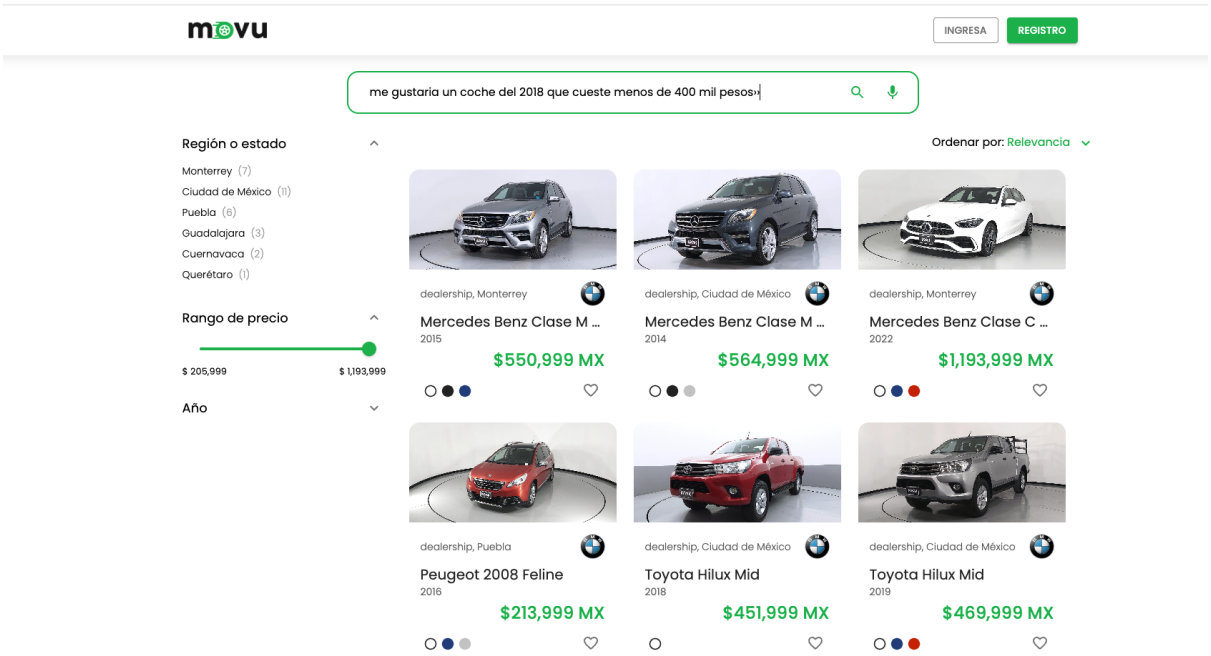
| True Positive (TP) | False Positive (FP) |
|---|---|
| 5 | 5 |
| **False Negative (FN)** | **True Negative (TN)** |
| 6 | 0 |

# Query #3

**Query:**

me gustaria un coche del 2018 que cueste menos de 400 mil pesos

**Screenshot:**




**Precision Value:** 0%

**Recall Value:** 0%

**Results precision vs. recall matrix:**

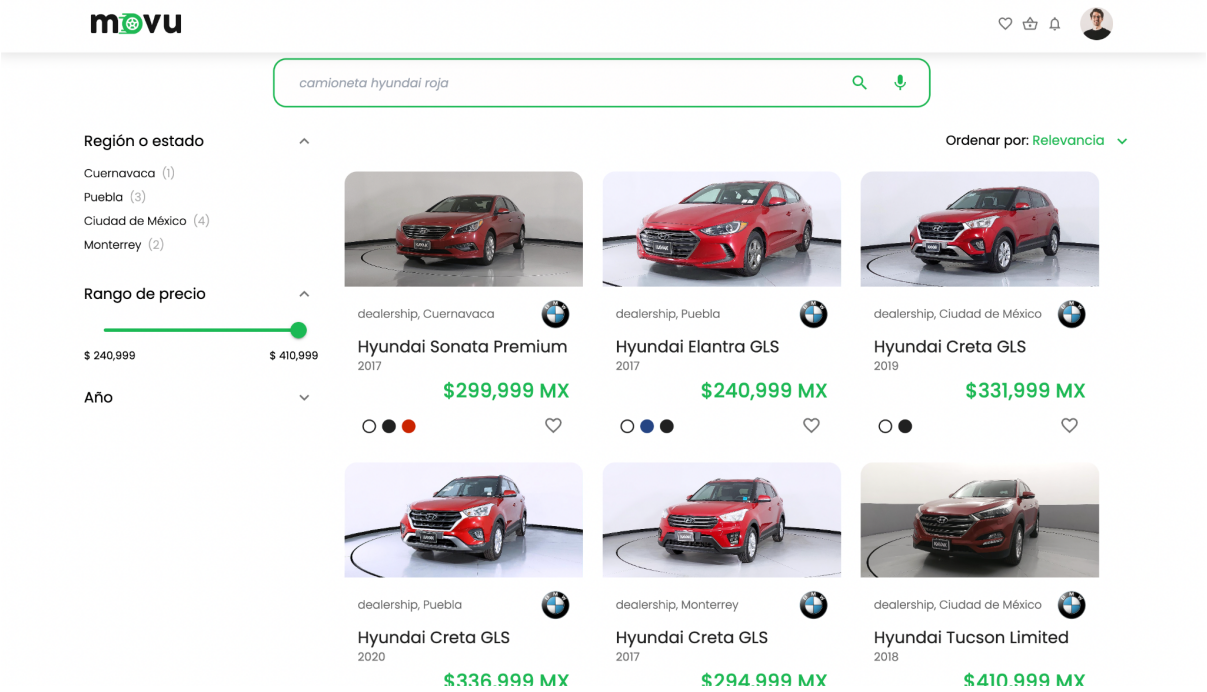
| True Positive (TP) | False Positive (FP) |
|---|---|
| 0 | 10 |
| **False Negative (FN)** | **True Negative (TN)** |
| 10 | 0 |

# Query #4

## Query:

camioneta hyundai roja

## Screenshot:




**Precision Value:** 60%

**Recall Value:** 16%

**Results precision vs. recall matrix:**

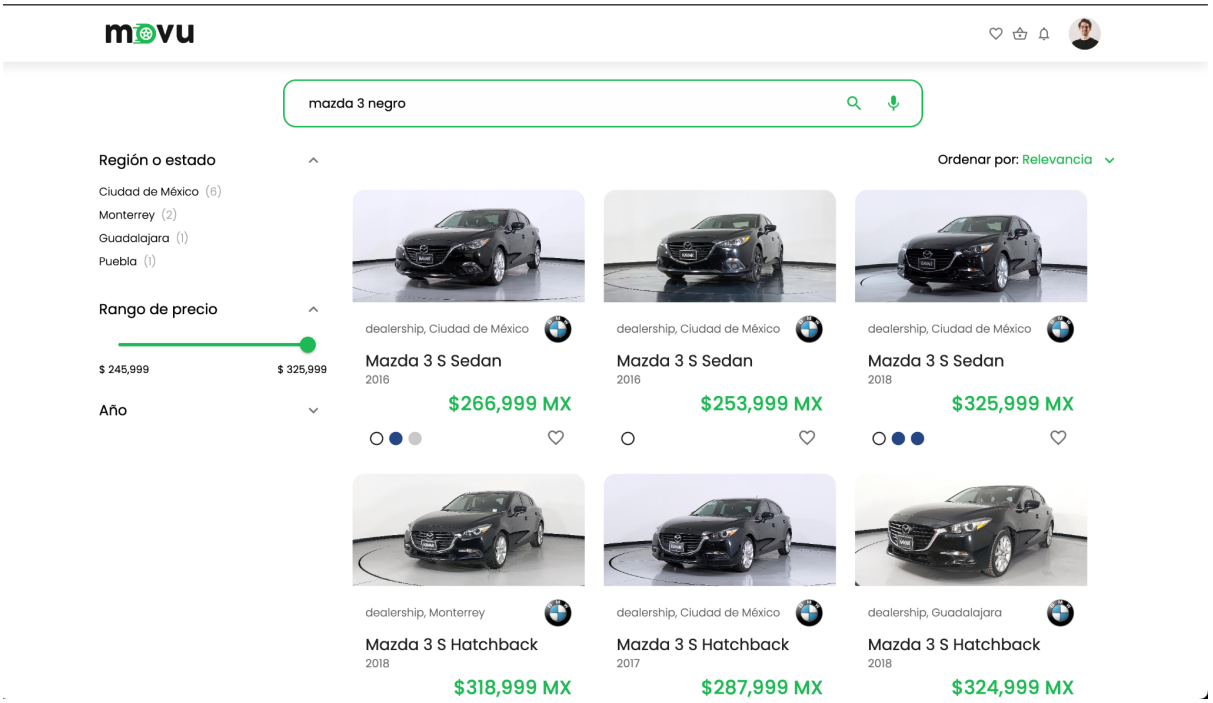
| True Positive (TP) | False Positive (FP) |
|---|---|
| 6 | 4 |
| **False Negative (FN)** | **True Negative (TN)** |
| 32 | 0 |

# Query #5

**Query:**

mazda 3 negro

**Screenshot:**




**Precision Value:** 20%

**Recall Value:** 1.1%

**Results precision vs. recall matrix:**

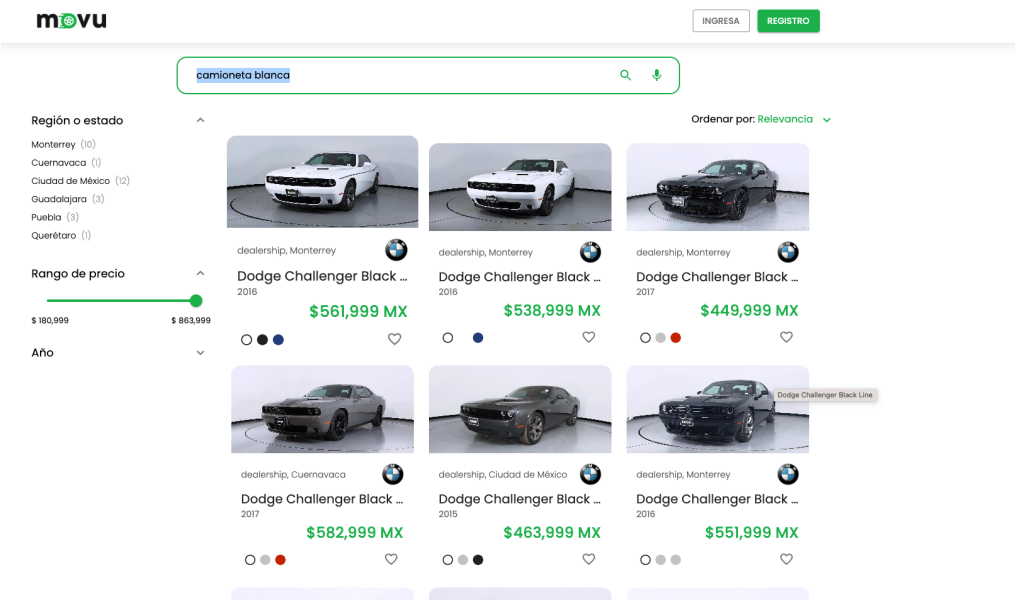
| True Positive (TP) | False Positive (FP) |
|---|---|
| 2 | 8 |
| **False Negative (FN)** | **True Negative (TN)** |
| 174 | 0 |

# Query #6

**Query:**

camioneta blanca

**Screenshot:**




**Precision Value:** 90%

**Recall Value:** 31%

**Results precision vs. recall matrix:**

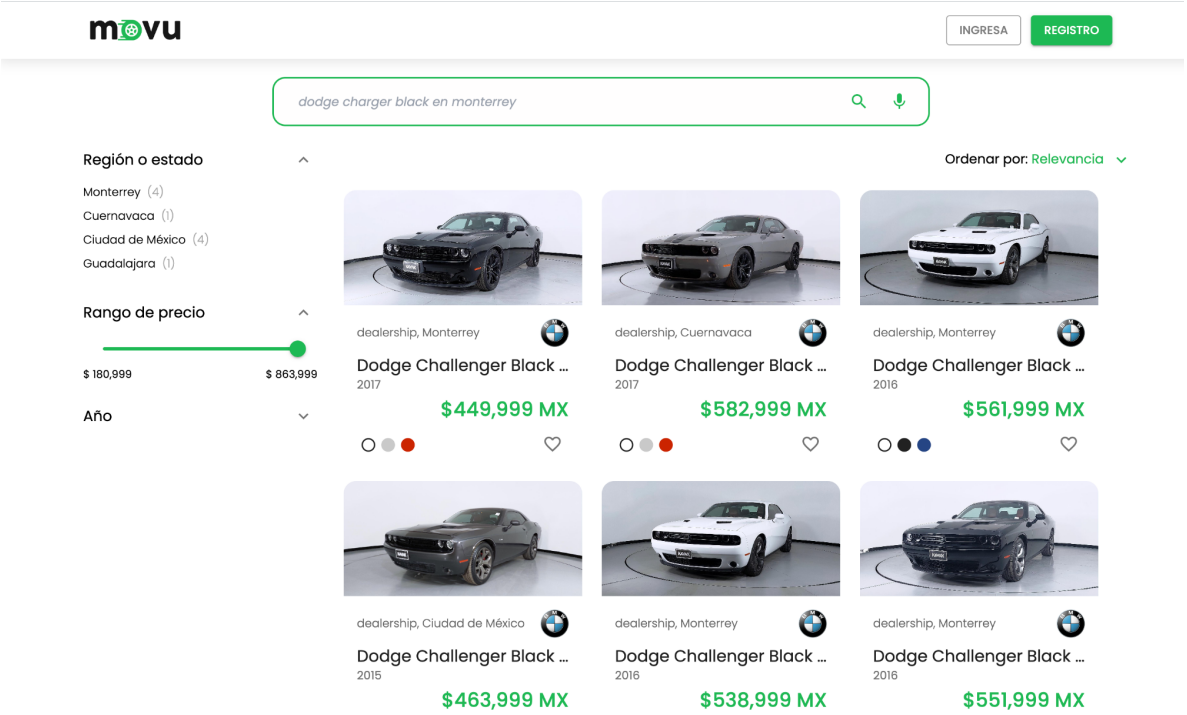
| True Positive (TP) | False Positive (FP) |
|---|---|
| 9 | 1 |
| **False Negative (FN)** | **True Negative (TN)** |
| 20 | 0 |

# Query #7

**Query:**

dodge charger black en monterrey

**Screenshot:**




**Precision Value:** 40%

**Recall Value:** 100%

**Results precision vs. recall matrix:**

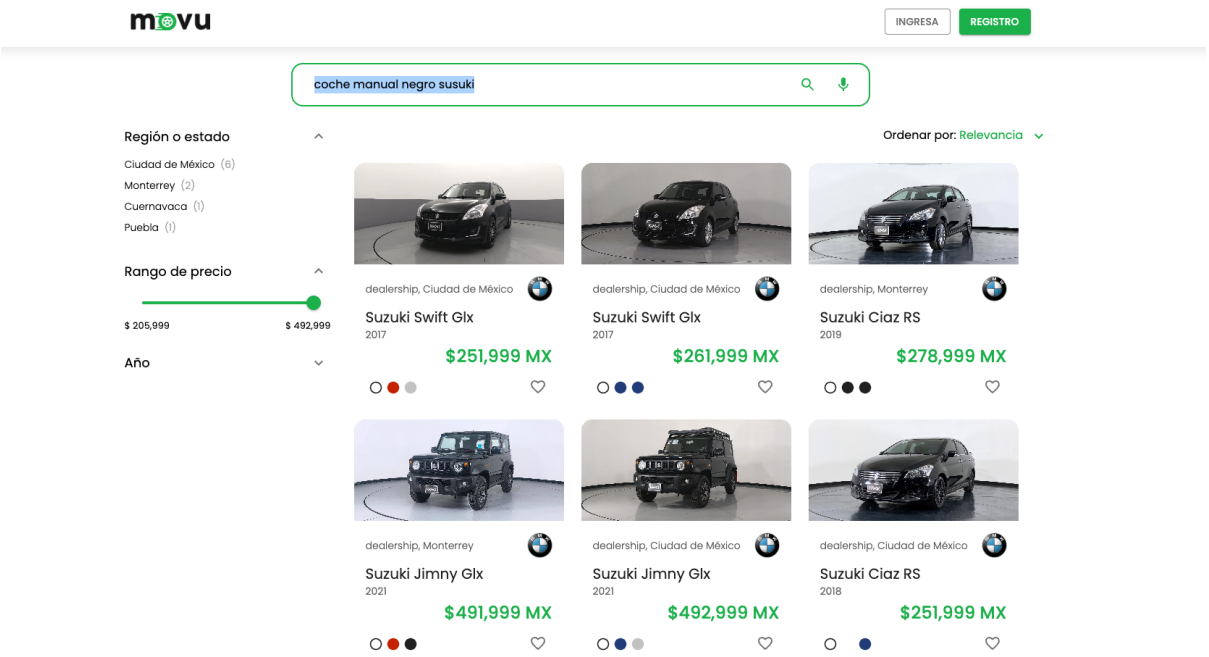
| True Positive (TP) | False Positive (FP) |
|---|---|
| 4 | 6 |
| **False Negative (FN)** | **True Negative (TN)** |
| 0 | 0 |

# Query #8

**Query:**

coche manual negro susuki

**Screenshot:**




**Precision Value:** 30%

**Recall Value:** 0.24%

**Results precision vs. recall matrix:**

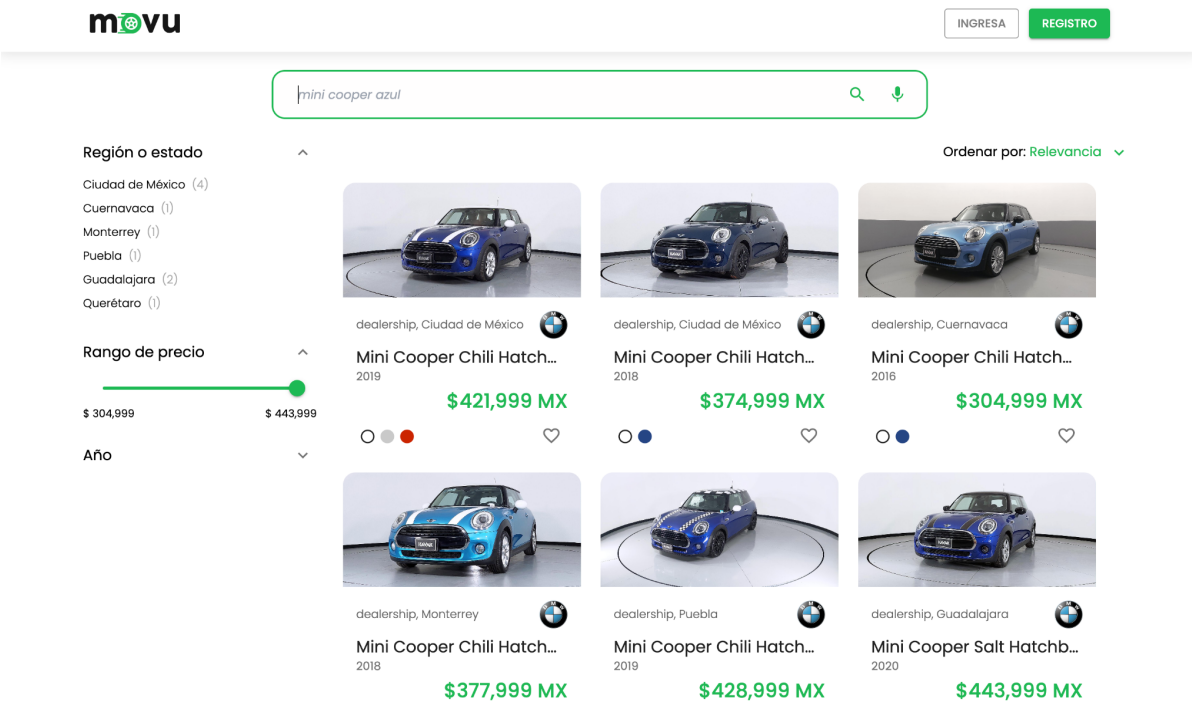
| True Positive (TP) | False Positive (FP) |
|---|---|
| 3 | 7 |
| **False Negative (FN)** | **True Negative (TN)** |
| 121 | 0 |

# Query #9

**Query:**

mini cooper azul

**Screenshot:**




**Precision Value:** 100%

**Recall Value:** 33%

**Results precision vs. recall matrix:**

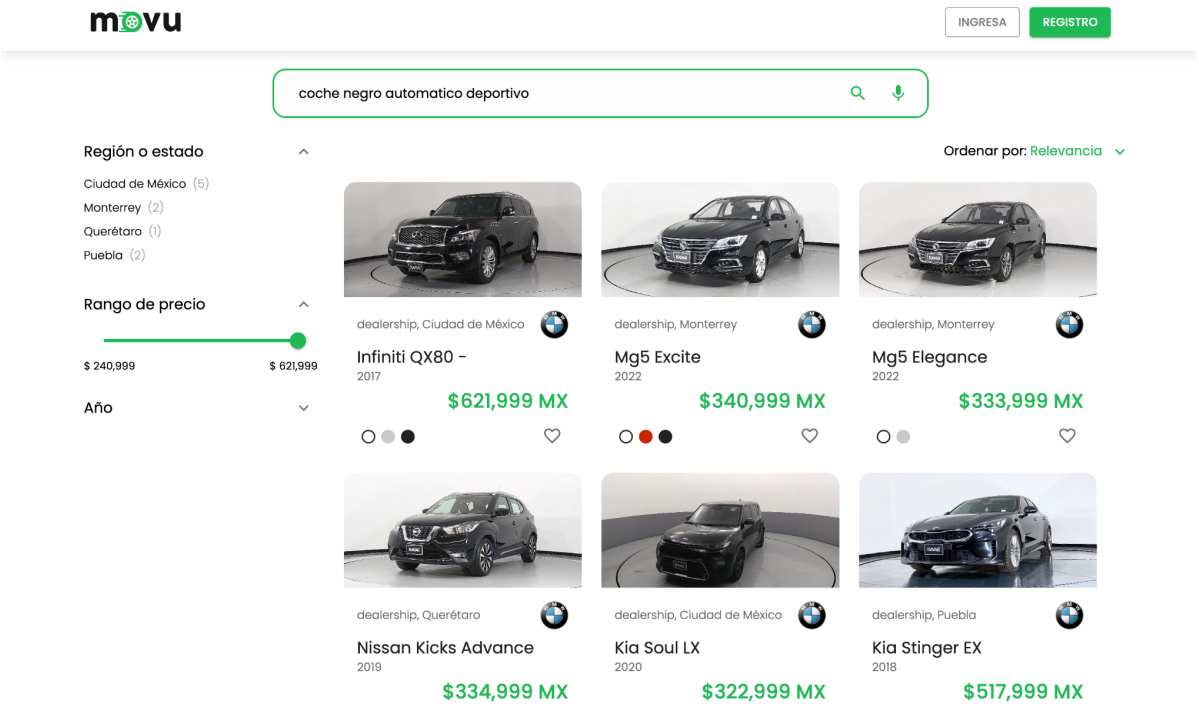
| True Positive (TP) | False Positive (FP) |
|---|---|
| 10 | 0 |
| **False Negative (FN)** | **True Negative (TN)** |
| 5 | 0 |

# Query #10

**Query:**

coche negro automatico deportivo

**Screenshot:**




**Precision Value:** 0%

**Recall Value:** 0%

**Results precision vs. recall matrix:**

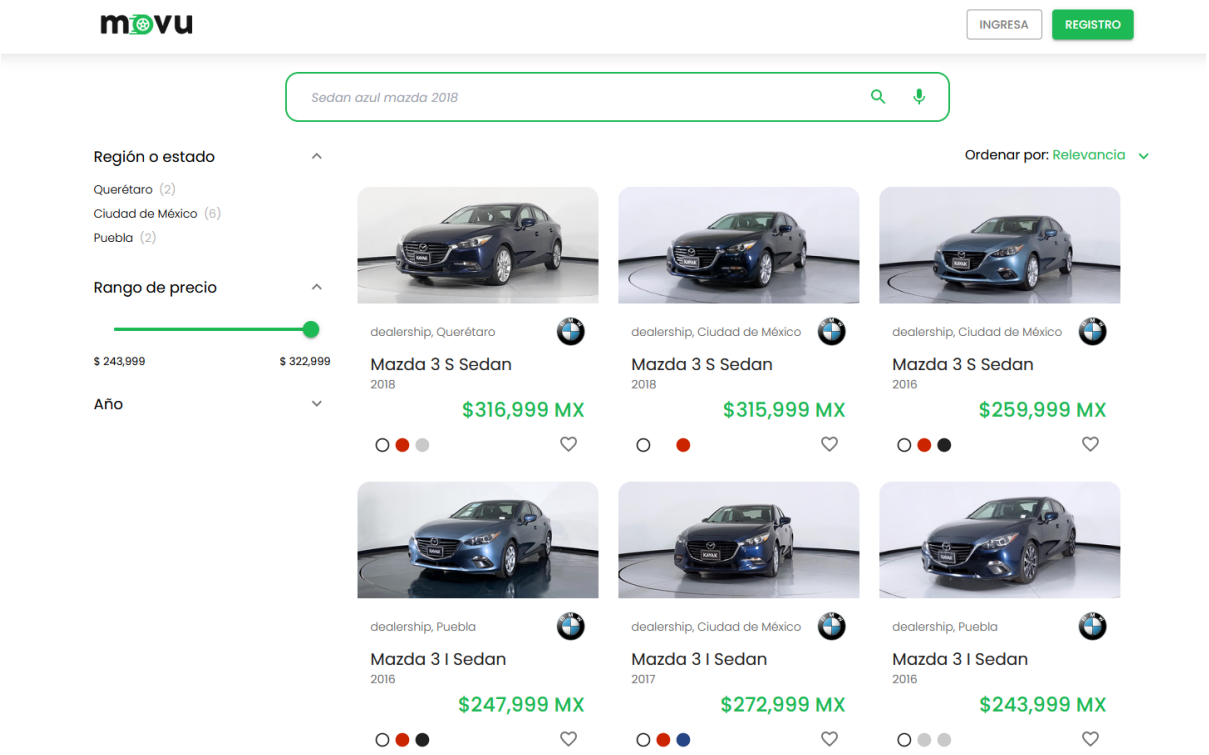
| True Positive (TP) | False Positive (FP) |
|---|---|
| 0 | 10 |
| **False Negative (FN)** | **True Negative (TN)** |
| 0 | 0 |

# Query #11

**Query:**

Sedan azul mazda 2018

**Screenshot:**




**Precision Value:** 40%

**Recall Value:** 1.3%

**Results precision vs. recall matrix:**

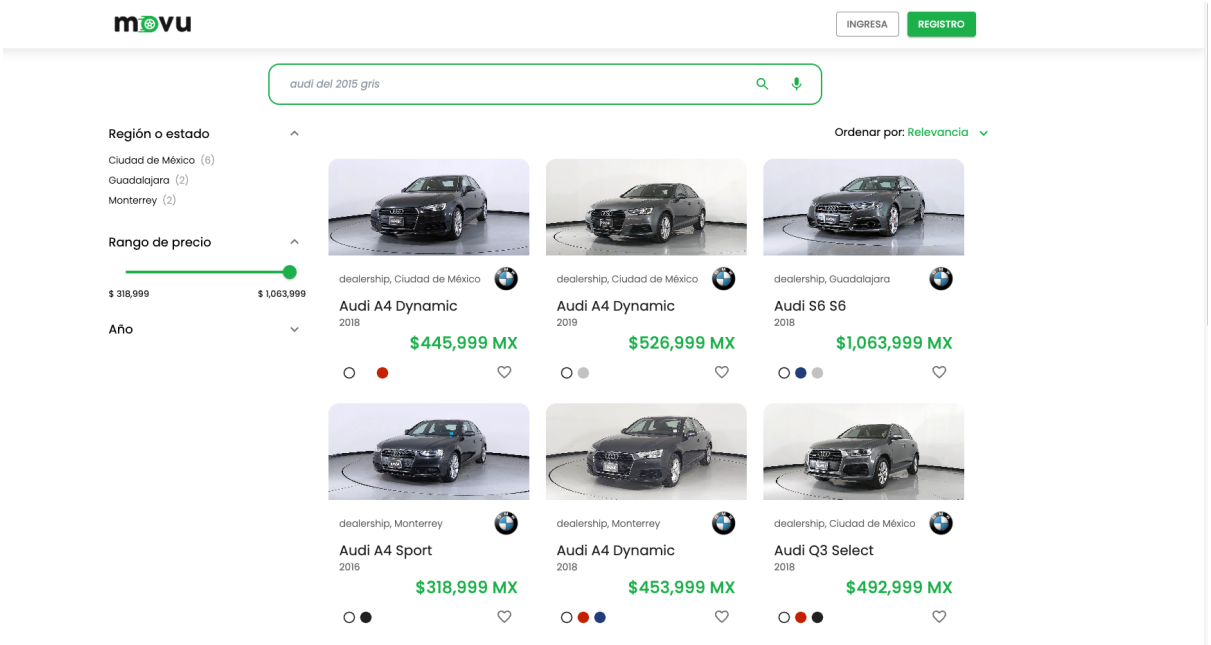
| True Positive (TP) | False Positive (FP) |
|---|---|
| 4 | 6 |
| **False Negative (FN)** | **True Negative (TN)** |
| 303 | 0 |

# Query #12

**Query:**

audi del 2015 gris

**Screenshot:**




**Precision Value:** 0%

**Recall Value:** 0%

**Results precision vs. recall matrix:**

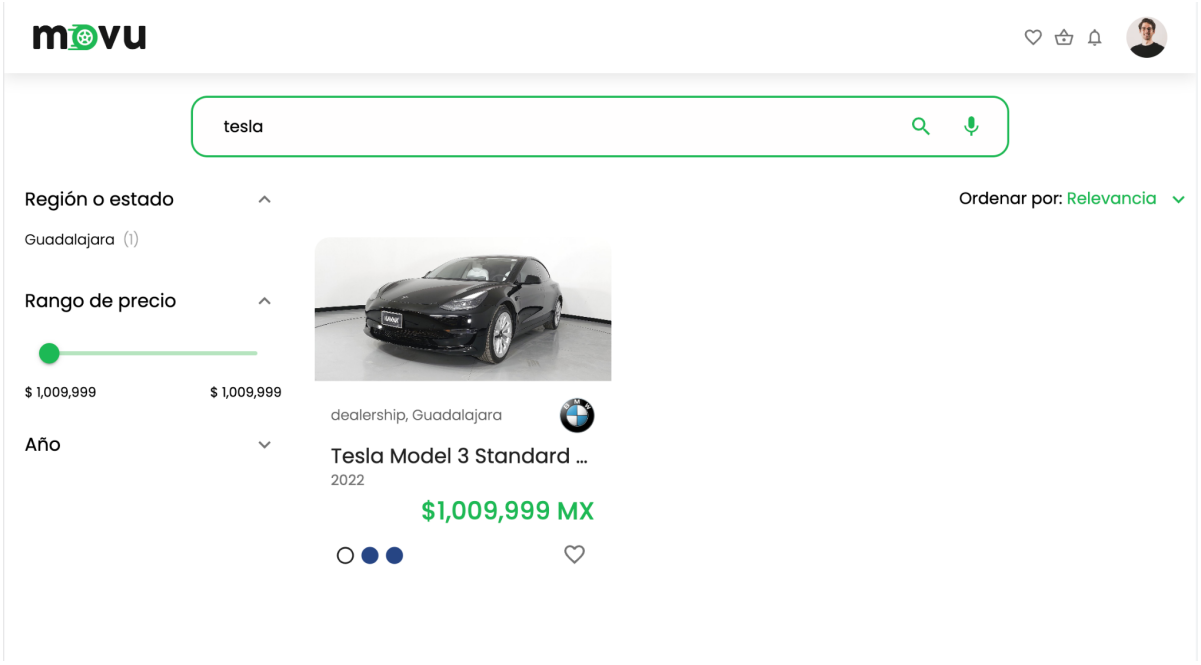
| True Positive (TP) | False Positive (FP) |
|---|---|
| 0 | 10 |
| **False Negative (FN)** | **True Negative (TN)** |
| 10 | 0 |

## Query #13

**Query:**

tesla

**Screenshot:**




**Precision Value:** 100%

**Recall Value:** 100%

**Results precision vs. recall matrix:**

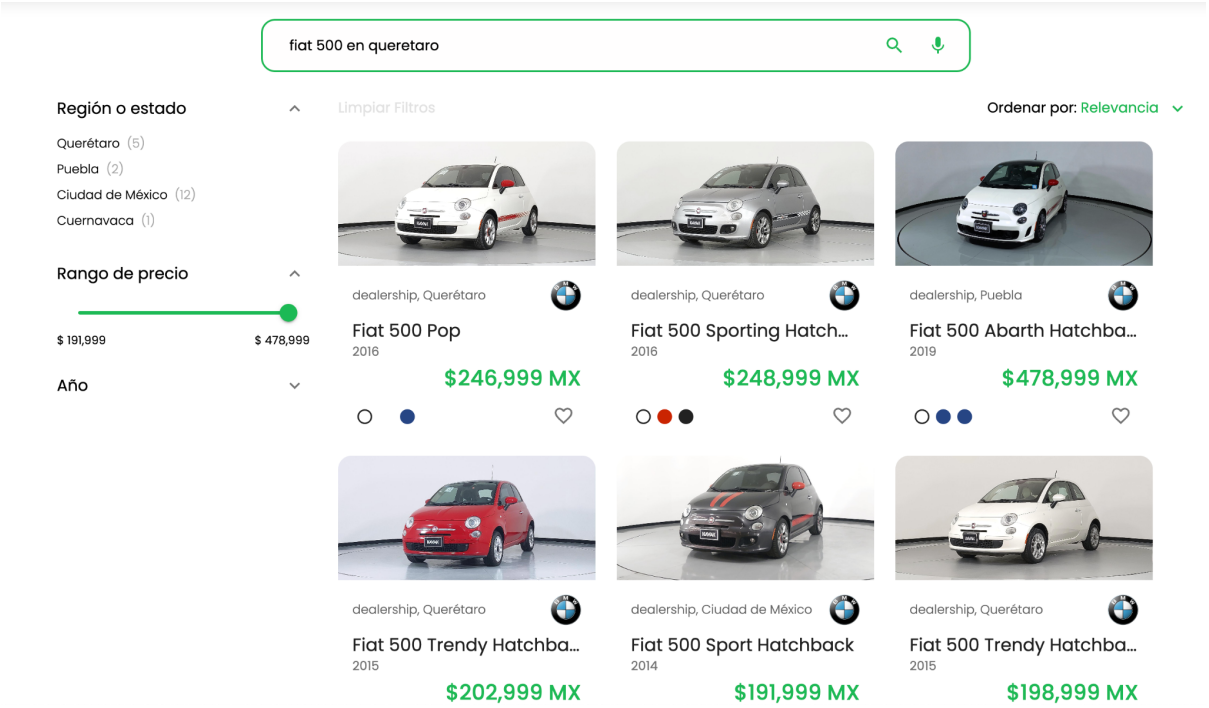
| True Positive (TP) | False Positive (FP) |
|---|---|
| 1 | 0 |
| **False Negative (FN)** | **True Negative (TN)** |
| 0 | 0 |

# Query #14

**Query:**

Fiat 500 en querétaro

**Screenshot:**




**Precision Value:** 50%

**Recall Value:** 100%

**Results precision vs. recall matrix:**

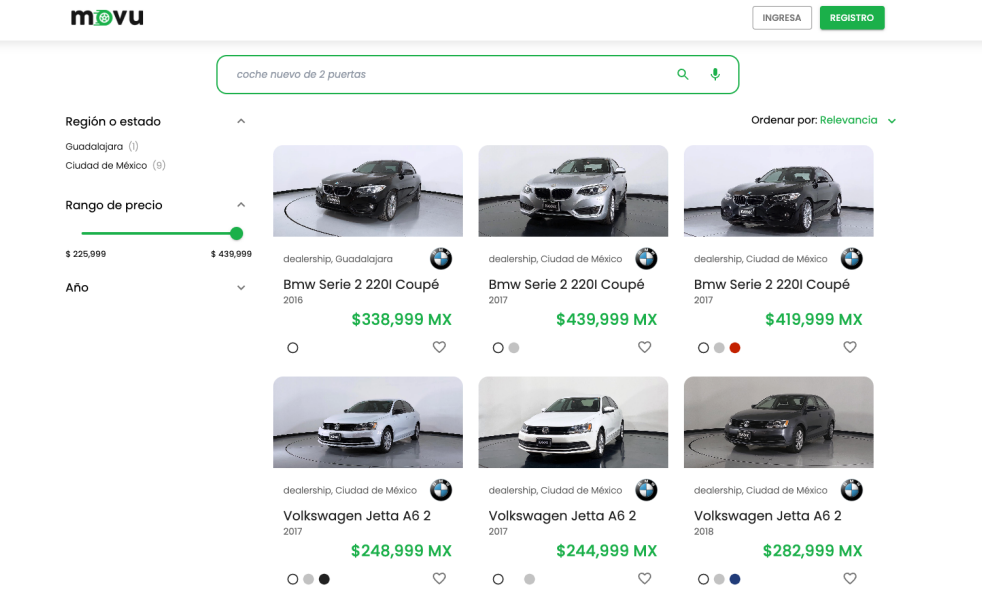
| True Positive (TP) | False Positive (FP) |
|---|---|
| 5 | 5 |
| **False Negative (FN)** | **True Negative (TN)** |
| 0 | 0 |

# Query #15

**Query:**

coche nuevo de 2 puertas

**Screenshot:**




**Precision Value:** 0%

**Recall Value:** 0%

**Results precision vs. recall matrix:**

| True Positive (TP) | False Positive (FP) |
|---|---|
| 0 | 10 |
| **False Negative (FN)** | **True Negative (TN)** |
| 15 | 0 |

## Query #16

**Query:**

jetta en la cdmx

**Screenshot:**

**Precision Value: 60%**

**Recall Value:  3.1 %**

**Results precision vs. recall matrix:**

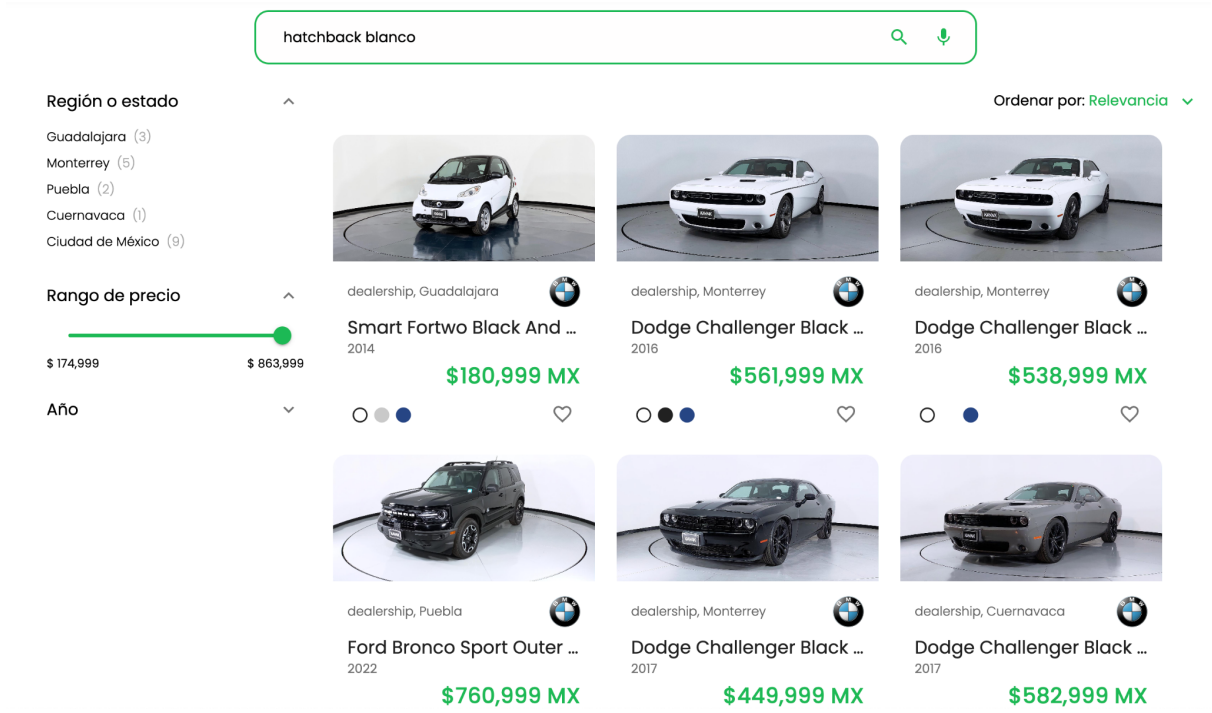
| True Positive (TP) | False Positive (FP) |
|---|---|
| 6 | 4 |
| **False Negative (FN)** | **True Negative (TN)** |
| 190 | 0 |

# Query #17

**Query:**

Hatchback blanco

**Screenshot:**




**Precision Value:** 0%

**Recall Value:** 0%

**Results precision vs. recall matrix:**

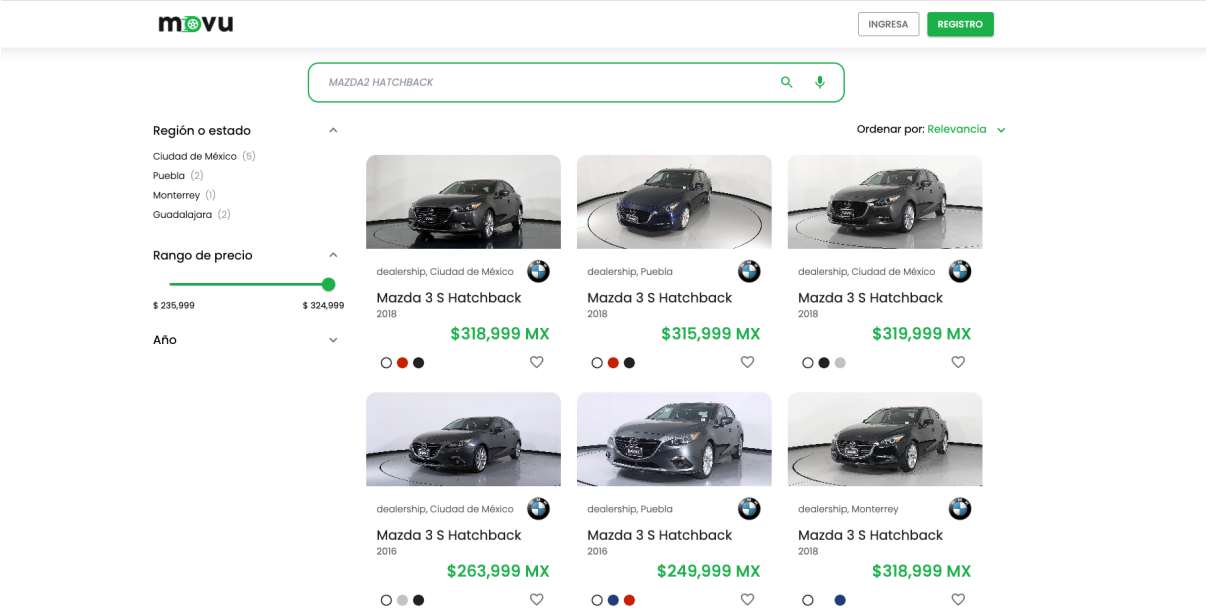
| True Positive (TP) | False Positive (FP) |
|---|---|
| 0 | 10 |
| **False Negative (FN)** | **True Negative (TN)** |
| 549 | 0 |

# Query #18

**Query:**

Mazda 2 Hatchback

**Screenshot:**




**Precision Value:** 10%

**Recall Value:** 14%

**Results precision vs. recall matrix:**

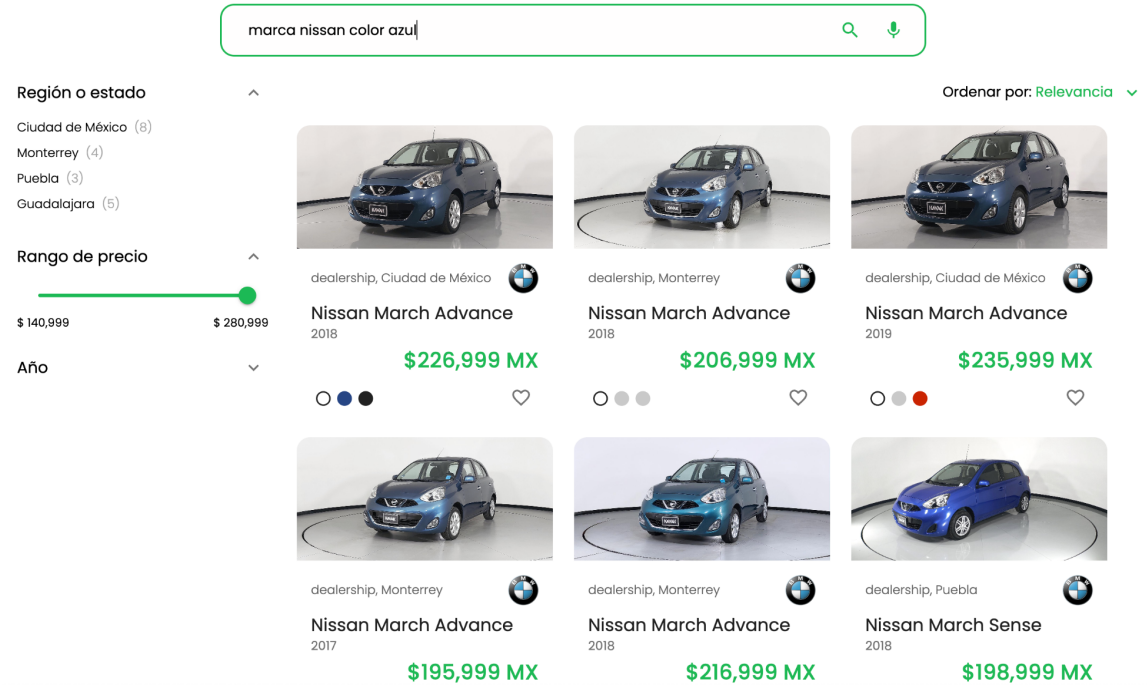
| True Positive (TP) | False Positive (FP) |
|---|---|
| 1 | 9 |
| **False Negative (FN)** | **True Negative (TN)** |
| 6 | 0 |

# Query #19

**Query:**

marca nissan color azul

**Screenshot:**




**Precision Value:** 50%

**Recall Value:** 0.98%

**Results precision vs. recall matrix:**

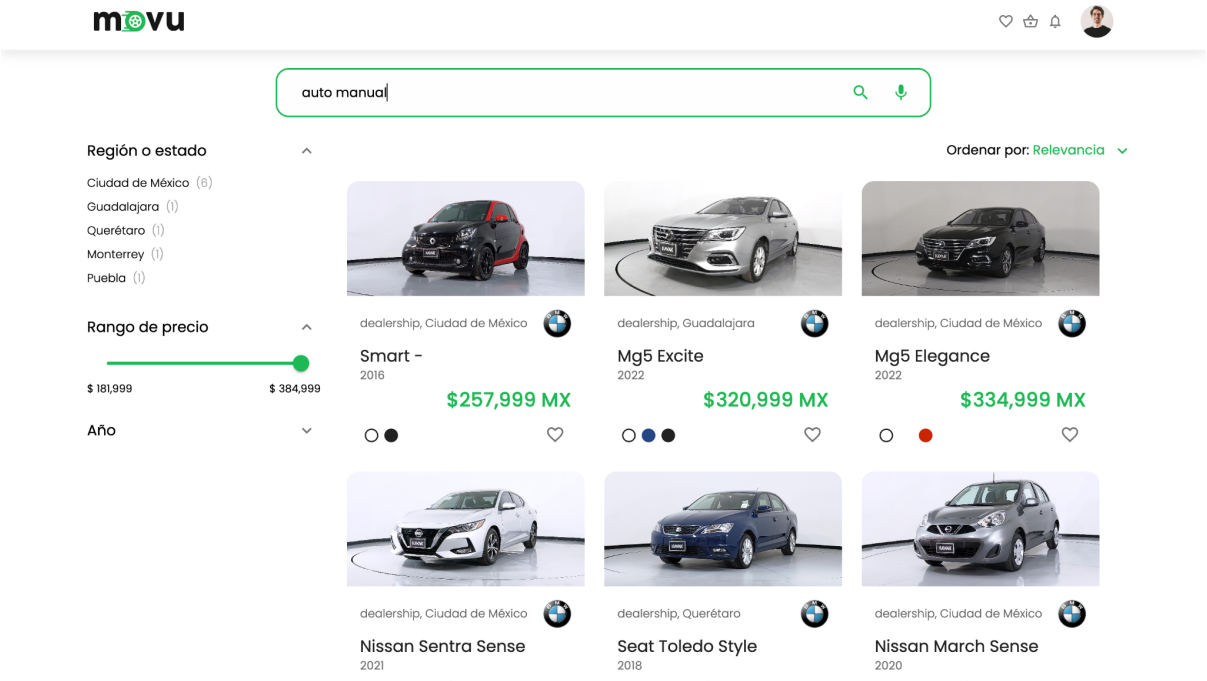
| True Positive (TP) | False Positive (FP) |
|---|---|
| 5 | 5 |
| **False Negative (FN)** | **True Negative (TN)** |
| 505 | 0 |

# Query #20

**Query:**

auto manual

**Screenshot:**




**Precision Value:** 100%

**Recall Value:** 0.0033%

**Results precision vs. recall matrix:**

| True Positive (TP) | False Positive (FP) |
|---|---|
| 10 | 0 |
| **False Negative (FN)** | **True Negative (TN)** |
| 3058 | 0 |

## Overall precision and recall values

Averaging the results of the 20 queries, we obtained the following values:

**Overall precision:** 40.5%

**Overall recall:** 27.28%

## Justification

Our data schema in Elasticsearch has been designed to enhance the precision and recall of the information retrieval system, achieving a balance between the two. Here's how our data schema contributes to obtaining more precise documents while maintaining a good recall:

- Descriptive Embeddings: The "description_embedding.predicted_value" property in the "listings-dev" index represents a dense vector that captures the textual representation of the listing description. By utilizing text embeddings and cosine similarity, we can calculate the similarity between documents. This approach allows us to retrieve more precise results by considering the semantic context of the search query.
- Keyword Search: In addition to neural search, we have implemented classic full-text search using Elasticsearch. By performing keyword-based searches across different fields, we leverage the power of Elasticsearch's search capabilities. This helps us retrieve relevant documents based on the user's query and rank them using the importance of the field and the number of hits. It improves precision by matching exact keywords and terms.
- Filters and Ranges: We provide users with the ability to apply filters, such as ranges or exact matches on specific fields, to refine their search results. By incorporating these filters, users can narrow down their search criteria and obtain more precise documents that meet their specific requirements.
- Autocomplete and User Behavior: We have implemented an auto-complete feature that suggests queries to users as they type. This feature not only improves the user experience but also contributes to precision by offering frequently used and highly scored queries. Storing previous queries and their respective scores in Elasticsearch allows us to prioritize popular queries and provide more accurate suggestions.
- Index settings: the settings that configure the "listings-dev" index are made to support Spanish language content with various text processing filters, including lowercase conversion, ASCII folding, stop word removal, and advanced stemming using the Snowball algorithm. The index has one shard and three replicas for redundancy and performance optimization.

By combining these strategies, we aim to strike a balance between precision and recall in our information retrieval system. Users can expect precise results based on the semantic context of their search query while ensuring a good recall rate by leveraging the power of full-text search and user behavior analysis.

## Integration of Elasticsearch into our e-commerce system

To incorporate Elasticsearch into our e-commerce system, we will integrate it as the primary search engine. Elasticsearch will handle the search queries from users and provide highly relevant and efficient results.

## Potential problems or areas of opportunity regarding information management and query handling

- Scalability: As our e-commerce platform grows and handles an increasing amount of data, scalability might become a concern. The size of the Elasticsearch cluster and the performance of search queries can be potential challenges. To mitigate this, we can implement sharding and replication strategies, optimize the Elasticsearch cluster configuration, and utilize hardware resources effectively.
- Relevance and Ranking: Ensuring the relevance and ranking of search results is an ongoing challenge. We may encounter situations where certain documents are ranked inaccurately or irrelevantly. Continuous monitoring, performance analysis, and feedback from users will be crucial in fine-tuning the relevance and ranking algorithms. Regular updates and improvements to our data schema and relevance models can help address these challenges.
- Query Understanding: Understanding user queries accurately and extracting relevant entities from plain text can be a complex task. Incorporating natural language processing techniques and leveraging external services, such as Azure's custom NER, can enhance the query understanding process. Regularly updating and fine-tuning these services based on user feedback and data analysis can further improve query understanding and accuracy.

## Measures of mitigation for query handling

- Continuous Monitoring and Analysis: Regular monitoring of search performance metrics, user feedback, and query logs will help identify any potential issues or areas for improvement. Analyzing search patterns, user behavior, and query performance will enable us to make data-driven decisions to enhance precision and recall.
- A/B Testing: Conducting A/B testing with different ranking algorithms, relevance models, or query processing techniques can provide insights into the effectiveness of different approaches. By comparing the performance of different variants, we can identify the most effective strategies and continuously iterate on them.
- User Feedback and Iterative Improvements: Actively soliciting and analyzing user feedback on search results and suggestions will help us understand user expectations and address any shortcomings. Incorporating user feedback into our iterative improvement process will enable us to enhance the precision and recall of the search system over time.
- Regular Maintenance and Updates: Keeping Elasticsearch and its associated components up to date with the latest versions, security patches, and performance enhancements will ensure a stable and efficient search system. Regularly reviewing and updating the data schema, relevance models, and indexing strategies will also contribute to improved query handling and information management.

By addressing these potential challenges and implementing the mitigation measures mentioned above, we aim to optimize the performance of our e-commerce search system, enhance precision and recall, and provide an excellent user experience.