Atlas Search is a full-text search service provided by MongoDB, designed to perform advanced search operations on data stored in MongoDB databases.

Now lets see the process step by step

**1. Index Creation:**

The first step in using Atlas Search is to create a search index on the collection that contains the data you want to search. This index defines which fields to index and how to analyze them during search operations.

For my website we created index over keywords,abstract and title fields

**2. Data Ingestion:**

After creating indexes as mentioned above we deployed around 820 documents on to the database.

**3. Query Parsing:**

When user search something using search feature In the UI that search will be sent as query using $search query operator for using the atlas search to the database from backend server,

At the mongodb server Search parses the query to understand the user's intent and requirements. This parsing involves breaking down the query into tokens, identifying keywords, and potentially extracting entities or features from the query.

**4. NLP Processing:**

Natural Language Processing (NLP) algorithms come into play here to enhance the search experience. These algorithms include techniques such as tokenization, lemmatization, part-of-speech tagging, named entity recognition (NER), sentiment analysis, and semantic analysis.

* Tokenization: Breaking down the query into individual words or tokens.
* Lemmatization: Reducing words to their base or dictionary form to handle variations like plurals or verb tenses.
* Part-of-speech tagging: Identifying the grammatical parts of words (noun, verb, adjective, etc.).
* Named Entity Recognition (NER): Identifying entities such as people, organizations, locations, etc., mentioned in the query.
* Sentiment Analysis: Determining the sentiment or mood expressed in the query.
* Semantic Analysis: Understanding the meaning and context of the query beyond just the individual words.

**5. Query Execution:**

After parsing and processing the query, Atlas Search executes the search operation using the index created earlier. It matches the query against the indexed fields and documents in the collection.

Atlas Search utilizes various search algorithms and techniques, including inverted indexes, TF-IDF (Term Frequency-Inverse Document Frequency) algorithm.

**TF-IDF (Term Frequency-Inverse Document Frequency) algorithm:**

The Term Frequency-Inverse Document Frequency (TF-IDF) algorithm is a widely used technique in information retrieval and text mining for evaluating the importance of a term within a document relative to a collection of documents. TF-IDF is commonly used in search engines, including Atlas Search, to rank documents based on their relevance to a given query. Let's break down the TF-IDF algorithm:

**1. Term Frequency (TF):**

Term Frequency measures how frequently a term appears in a document. It indicates the importance of a term within the document.

Term Frequency for a term t in a document d is calculated as the ratio of the number of times term t appears in the document to the total number of terms in the document.

Mathematically, TF is calculated as:

TF(*t*,*d*)=Total number of terms in document *d/*Number of times term *t* appears in document *d*​

**2. Inverse Document Frequency (IDF):**

* Inverse Document Frequency measures the importance of a term across a collection of documents. It helps to identify rare terms that may have more discriminatory power.
* IDF for a term t is calculated as the logarithm of the ratio of the total number of documents in the collection to the number of documents containing the term t.

Mathematically, IDF is calculated as:

IDF(*t*)=log(Number of documents containing term *t/*Total number of documents​)

**3. TF-IDF Weighting:**

* TF-IDF is calculated by multiplying the Term Frequency (TF) of a term in a document by its Inverse Document Frequency (IDF) across the entire collection.
* The resulting TF-IDF score reflects the importance of a term within a document relative to its importance across the entire collection.

Mathematically, TF-IDF is calculated as:

TF-IDF(*t*,*d*,*D*)=TF(*t*,*d*)×IDF(*t*)

where D represents the collection of documents.

**4. Ranking Documents:**

* Once the TF-IDF scores are calculated for all terms in a document, documents are ranked based on their overall TF-IDF scores for the query terms.
* Documents with higher TF-IDF scores are considered more relevant to the query.

**6. Presentation of Results:**

Finally the results are returned by mongodb server to the website backend server and backend server will send the results to frontend where result publicaions will be displayed.