NS - Architecture Documentation

Date: 24 abr 2023

Team:

Objective:

The objective of this natural search team was to make a simple search experience that doesn't rely on filters.

User Stories:

As a **user** I want to search for cars without the need of filters or advanced details to be able to have an easy experience in searching my car.

Concepts:
Text-Embedding:
Custom NER:
Elastic Search:
IP Geolocation:

Overview:

Docs for API

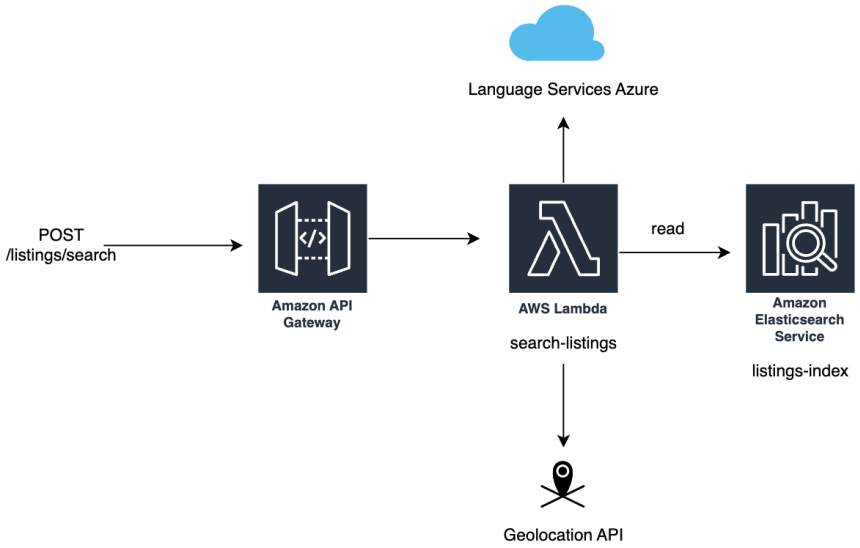
To create the best search experience we implemented a search endpoint that has two modes.

- 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)

Architecture diagram:

Search Diagram:



Search-Listings Lambda: This service is in charge of determining the queries mode, if it needs to call the geolocation api and build the query that will be sent to elastic-search.

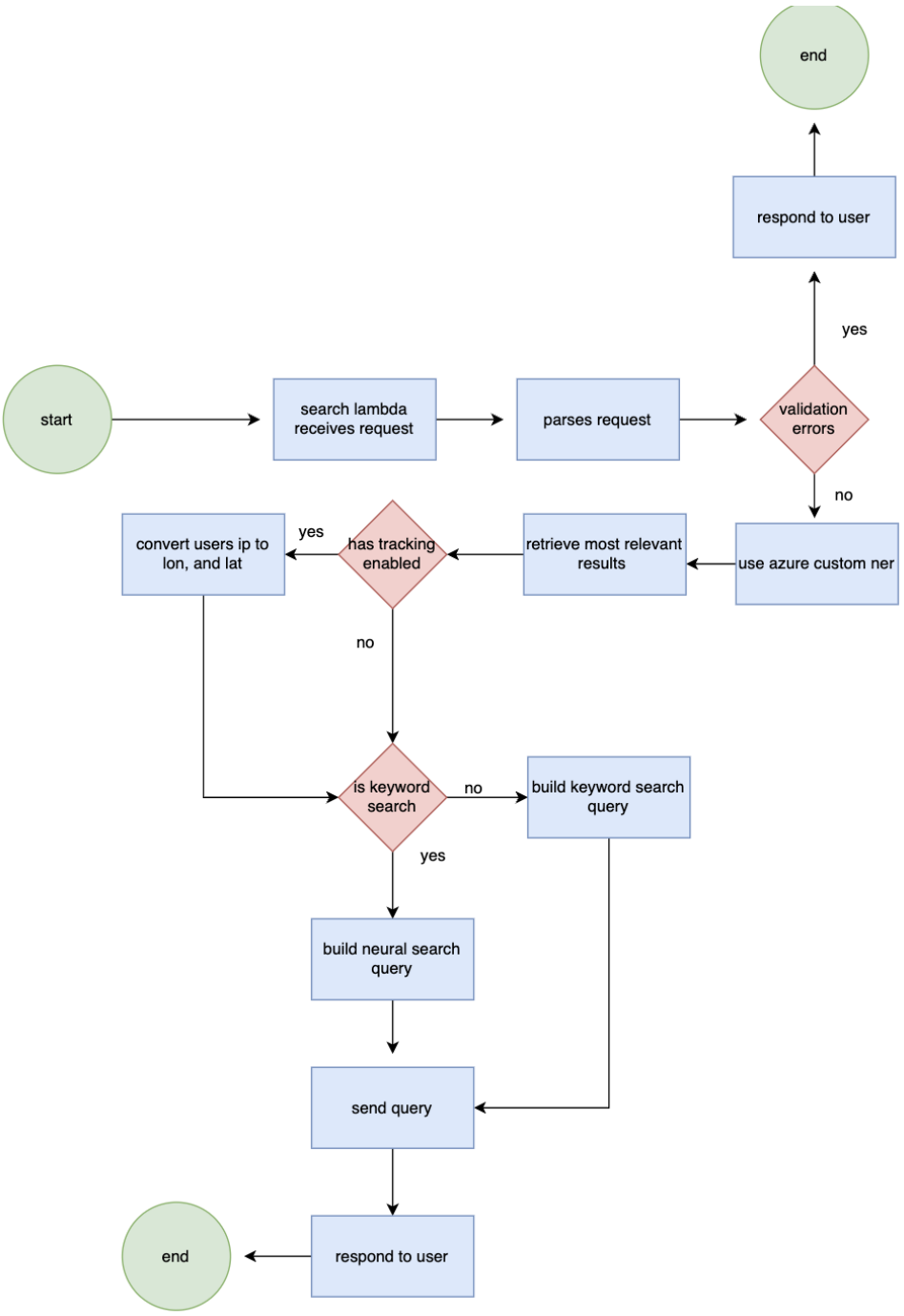
Language Services Azure: Custom NER model was trained and deployed using Azure Language Services, the service is in charge of receiving the user's query and returning a list of entities related to the car, for example the make, model, year. This in order to reduce the search space, without the user specifying the filters.

Geolocation API: If enable_tracking parameter is passed in to the search endpoint then this service will receive the users ip address and convert it to a coordinate, then we can use the coordinate to filter the nearest hits from elasticsearch.

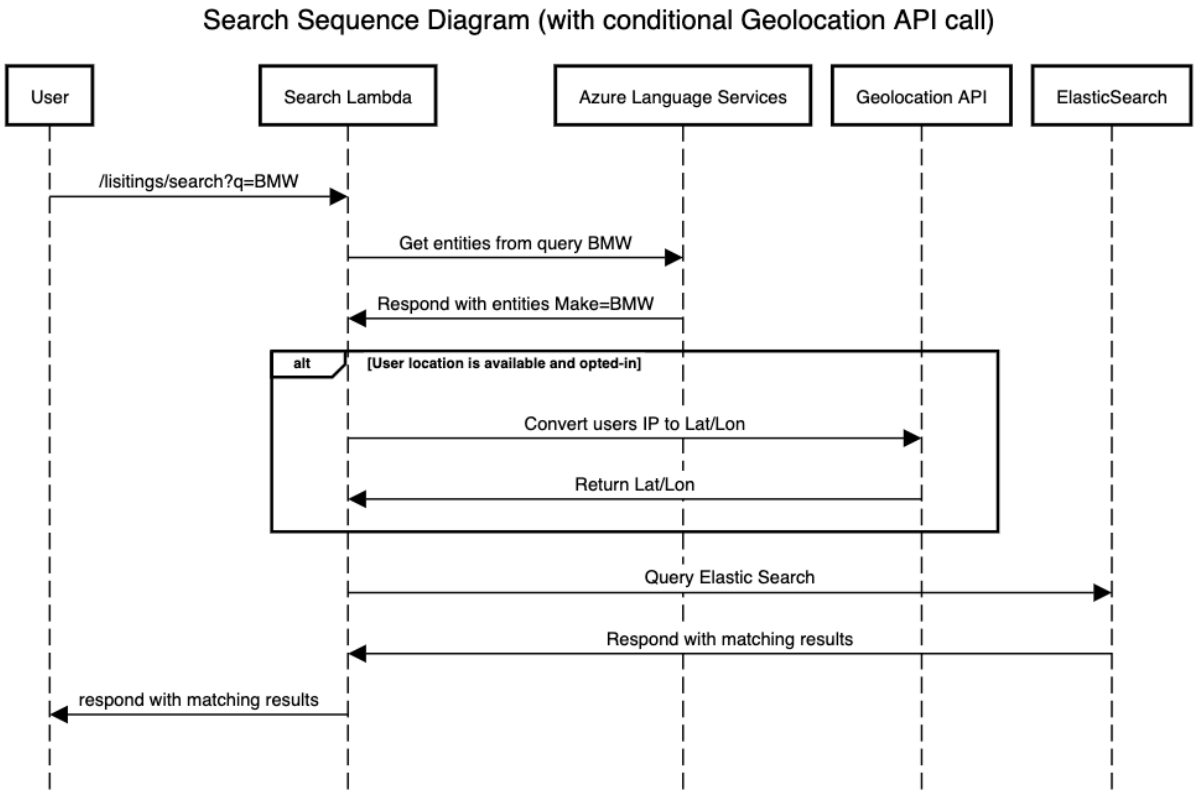
Elasticsearch: This is a NoSQL DB, and search engine we use to retrieve our information, we have setup an ingestion pipeline that when used creates a text-embedding out of the description field, and when we query data we also create an embedding for the query in order for us to look in the database for contextually similar results. It is used for all of our queries, full text search, finding results that are geographically close and contextually.

Control Flow Diagram:

Search Listings Flow Diagram:



Sequence Diagrams:



Security Considerations:

Risks:

Future Improvements: