Pre-Index Search data using Events.

# Status

*Draft*

# Context

The Search Service maintains a Search Index used for searching meals based on multiple criteria like Nutritional value, meal-type, locality, and other factors. The output needs to be ranked to maximize the probability of placing the order and is based on parameters such as location, availability, purchase history and dietary information. For an optimal user experience, the service needs to respond to queries within 3 seconds.

All this information needs to be published to the Search Service as Application events in an Event Bus.

# Decision

We will use an Event Bus to post the required events needed by Search Service to index the search data.

For search service to respond to user queries within the stipulated time, the data needs to be pre-processed into a Search Index. Having dependencies while searching on other services would introduce indeterminism in the result while making the search time dependent on the SLA of these services. Hence to remove the coupling with other services, Search service should post the query only to the Search index and no additional requests should be invoked.

There are multiple independent services which master the data that is needed to create the Search Index. These data would become available based on the operations happening in the system. The operations from these services should be posted to the Search Service as Events. The search service would be listening to the Event Bus and index the data based on the published events. Depending upon the type and complexity of the event, the time and processing power needed to index the database might vary. E.g. an event like the addition of a new region might trigger indexing a considerable portion of the database. We need to protect the Search service from getting choked in such scenarios. Hence, the Search service will be allowed to consume events from the Event Bus based on the availability of memory and processing power. This would also ensure that the Search Index would be able to respond to queries without any increase in latency while it’s getting re-indexed.

# Consequence

Data in Search Index would be eventually consistent with the global inventory snapshot. Updates in the Inventory like the addition of new meals, meal availability or addition of new smart fridges wouldn’t be immediately available for users to query.

List the exact points needed by Search Service; Should we mention the performance limit of Search Service? Do we need an Event Bus, or a simple push API will suffice?