Story 1

Provides a search function API

Purpose:

Provides an API for other modules to call, searching the specified index table, and return results.

Scenarios:

Other modules search API through OKAPI call, parameters including index, COLs, sort, size, from, and Conditions. After the module accepts parameters, parameter verification is carried out, including the authentication of legality and validity. The parameters are then sorted into ES retrieval parameters and forwarded to ES server. Forward the return result from the ES server to the caller.

Story 2

Provides data import function API

Purpose:

Enables users to import indexes and data into ElasticSearcch through database configuration

Scenarios:

Scenario 1:

Configure indexes for ES. Users can read the configuration through the configuration of the database and import the configured index information, including the index name, index type, whether to store it or not, and the type of separator, into the ElasticSearcch through the program.

Scenario 2:

Manual data import. The user passes the relevant configuration of the database. Get the SQL statement saved in the database, through the SQL statement and related database connection data, from the database to get the data that needs to be converted to ES. Then, according to the source field saved in the database and the corresponding ES conversion field, the obtained data is converted into JSON conforming to the ES standard and sent to Kafka. Finally, Logstash outputs JSON to the ES by consuming messages from the Kafka message queue.

Scenario 3:

Real-time data import. The program USES Logstash to listen for changes in the database, thus synchronizing the changed data in the database into ES in quasi-real-time.

Scenario 4:

Timing data import. The program periodically executes SQL statements to send the changed data to the Kafka message queue, and then Logstash outputs JSON data to the ES by consuming the messages in the Kafka message queue.