Elasticsearch Search Feature Guide

Overview

This guide explains the Elasticsearch search functionality integrated into the Amazon-like e-commerce site.

Features Implemented

1. Elasticsearch Integration

- **Document Model**: `ProductDocument` - Elasticsearch representation of products
- **Repository**: `ProductSearchRepository` - Handles Elasticsearch operations
- **Service**: `ProductSearchService` - Business logic for search and indexing
- **Configuration**: Elasticsearch runs on `localhost:9200` (via Docker)

2. Search Capabilities

- **Full-text search** across product names and descriptions
- **Case-insensitive** matching
- **Category-based** filtering
- **Real-time** search results
- **40+ Products** across multiple categories for comprehensive testing

3. Product Categories

- Electronics (9 products)
- Books (5 products)
- Home & Kitchen (6 products)
- Sports & Outdoors (6 products)
- Fashion (6 products)
- Beauty & Personal Care (5 products)
- Gaming (4 products)

Starting the Application

Step 1: Start All Services

```
./start-all.sh
```

This starts:

- PostgreSQL database
- Elasticsearch
- Logstash

- Kibana

Step 2: Verify Elasticsearch is Running

```
curl http://localhost:9200
```

You should see Elasticsearch version information.

Step 3: Start the Spring Boot Application

```
./gradlew bootRun
```

The application will:

1. Create 40+ products in PostgreSQL
2. Automatically index all products to Elasticsearch
3. Start on `http://localhost:8081`

Testing Search Functionality

Web UI Search

1. Open `http://localhost:8081` in your browser
2. Enter a search term in the search bar (e.g., "Apple", "coffee", "gaming", "yoga")
3. Press Enter or click the Search button
4. See filtered results matching your query

Example Searches to Try

Search by Brand/Product Name

- `Apple` - Shows AirPods, Apple Watch
- `Nike` - Shows Nike Air Force 1 sneakers
- `Samsung` - Shows Galaxy Buds
- `PlayStation` - Shows PS5 console

Search by Product Type

- `coffee` - Shows Keurig coffee maker
- `headset` - Shows gaming headset
- `blender` - Shows Vitamix blender
- `vacuum` - Shows Roomba robot vacuum

Search by Category

- `gaming` - Shows PlayStation, Nintendo Switch, Xbox controller
- `yoga` - Shows yoga mat
- `fitness` - Shows fitness tracker
- `books` - Shows various books

Search by Features/Description

- `wireless` - Shows wireless earbuds, controllers, headsets
- `noise cancelling` - Shows AirPods Pro, Galaxy Buds
- `waterproof` - Shows Hydro Flask, YETI
- `electric` - Shows electric toothbrush, pressure cooker

API Endpoints

1. Elasticsearch Search

```
curl "http://localhost:8081/api/products/elasticsearch/search?
keyword=coffee"
```

2. Database Search (Legacy)

```
curl "http://localhost:8081/api/products/search?keyword=apple"
```

3. Get All Products

```
curl http://localhost:8081/api/products
```

4. Reindex Products (Manual)

```
curl -X POST http://localhost:8081/api/products/elasticsearch/reindex
```

5. Get Product by ASIN

```
curl http://localhost:8081/api/products/asin/B09B8V1M94
```

6. Get Products by Category

```
curl http://localhost:8081/api/products/category/Electronics
```

Verifying Elasticsearch Index

Check Products Index

```
curl http://localhost:9200/products/_search?pretty
```

Search Products in Elasticsearch Directly

```
curl -X GET "http://localhost:9200/products/_search?pretty" -H 'Content-
Type: application/json' -d'
{
  "query": {
    "multi_match": {
      "query": "apple",
      "fields": ["name", "description"]
    }
  }
}
'
```

Count Indexed Products

```
curl "http://localhost:9200/products/_count?pretty"
```

Should show approximately 40 products.

Kibana Dashboard

Access Kibana at `http://localhost:5601` to:

- Visualize search queries
- Monitor Elasticsearch health
- Create dashboards for product analytics
- Debug search queries

Troubleshooting

Elasticsearch Not Running

```
docker ps | grep elasticsearch
```

If not running:

```
docker-compose up -d elasticsearch
```

Products Not Indexed

Manually trigger reindexing:

```
curl -X POST http://localhost:8081/api/products/elasticsearch/reindex
```

Check Application Logs

```
tail -f logs/application.log
```

Look for:

- `✓ Successfully indexed all products to Elasticsearch`
- Any Elasticsearch connection errors

Clear and Rebuild Index

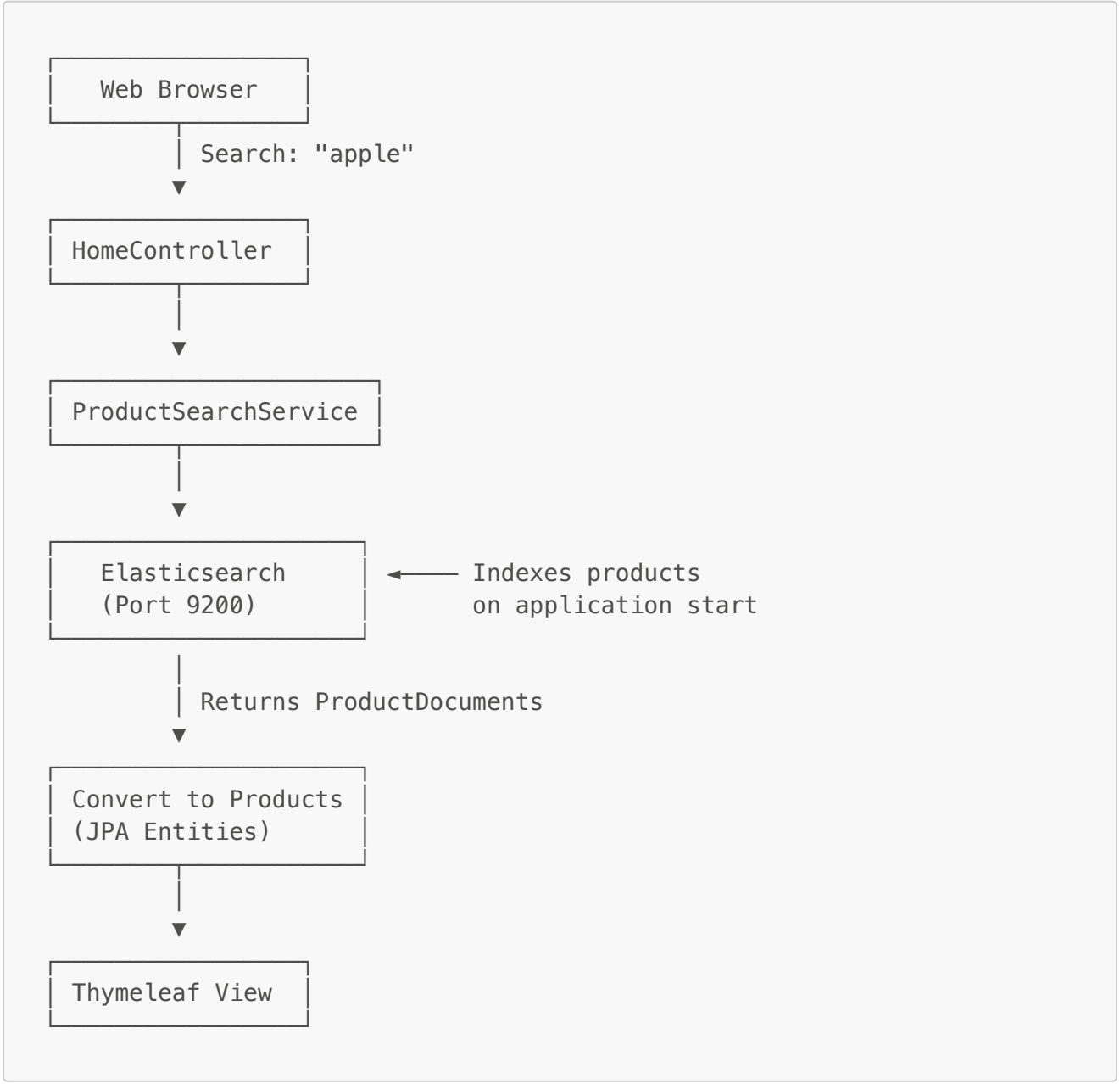
```
# Delete the index
curl -X DELETE http://localhost:9200/products

# Restart the application to recreate and reindex
./gradlew bootRun
```

Architecture

Data Flow

1. **Product Creation**: Products saved to PostgreSQL via JPA
2. **Indexing**: ProductSearchService indexes products to Elasticsearch
3. **Search Query**: User searches via web UI
4. **Query Processing**: HomeController routes to ProductSearchService
5. **Elasticsearch Query**: ProductSearchService queries Elasticsearch
6. **Result Mapping**: Elasticsearch documents mapped back to JPA entities
7. **Display**: Results rendered in Thymeleaf template

Key Components



Performance

Search Speed

- Elasticsearch provides near-instant search results
- Handles complex queries efficiently
- Scales well with large product catalogs

Indexing

- Initial indexing on application startup
- Can be manually triggered via API
- Automatic for new products (can be enhanced)

Future Enhancements

1. **Auto-sync**: Automatically index new products when created

2. **Faceted Search**: Add filters by price, category, availability
3. **Suggestions**: Implement autocomplete/search suggestions
4. **Relevance Tuning**: Adjust scoring and ranking
5. **Synonyms**: Add synonym support (e.g., "cellphone" → "smartphone")
6. **Fuzzy Matching**: Handle typos in search queries
7. **Aggregations**: Show category counts, price ranges

Stopping Services

```
./stop-all.sh
```

This stops all Docker containers (Elasticsearch, PostgreSQL, Logstash, Kibana).

Additional Resources

- Elasticsearch Documentation
- Spring Data Elasticsearch
- Project README: `README.md`
- Quick Start Guide: `QUICK_START.md`