# Tutorial: Using Laravel with ElasticSearch

Prerequisite: Docker is not required in this project.

However, ElasticSearch must be installed.

## Integrate Elasticsearch into Laravel

Next, we will use existing project to integrate to ElasticSearch. Materials included in this tutorial contains sql file with schema and data named **sakila**

Install this package inside Laravel

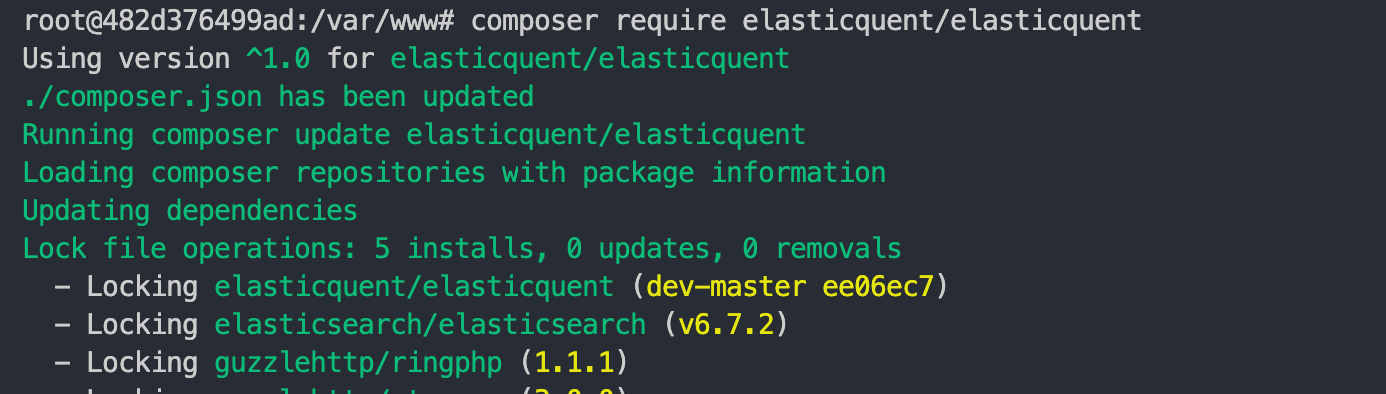


Figure 1: Install elasticquent to work with elasticsearch

Add ElasticquentServiceProvider to config/app.php for modifying Elasticsearch connection

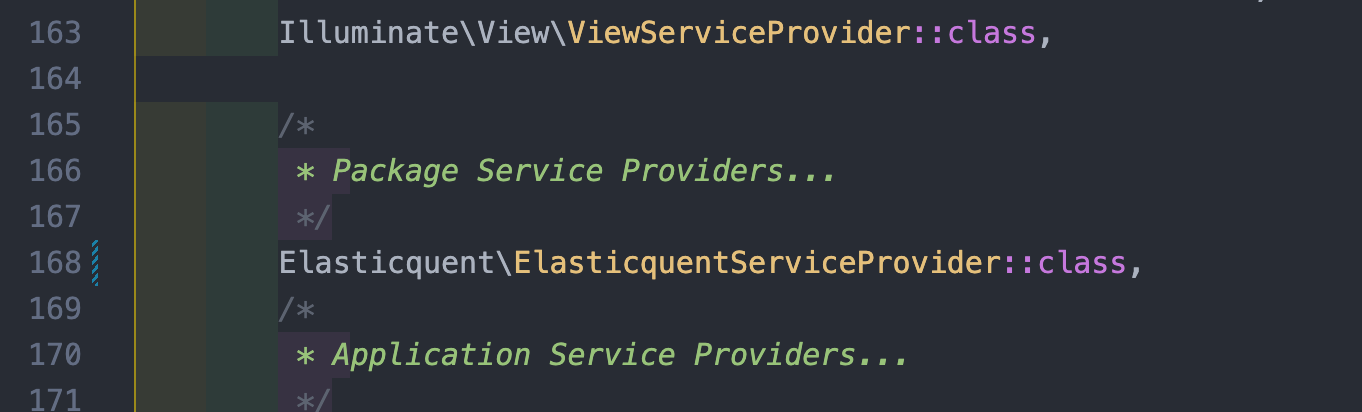


Figure 2: ElasticquentServiceProvider

Publishing config in ElasticSearch



Figure 3: Publish config Elasticsearch

Next, change default index to <your\_index>, such as **sakila**

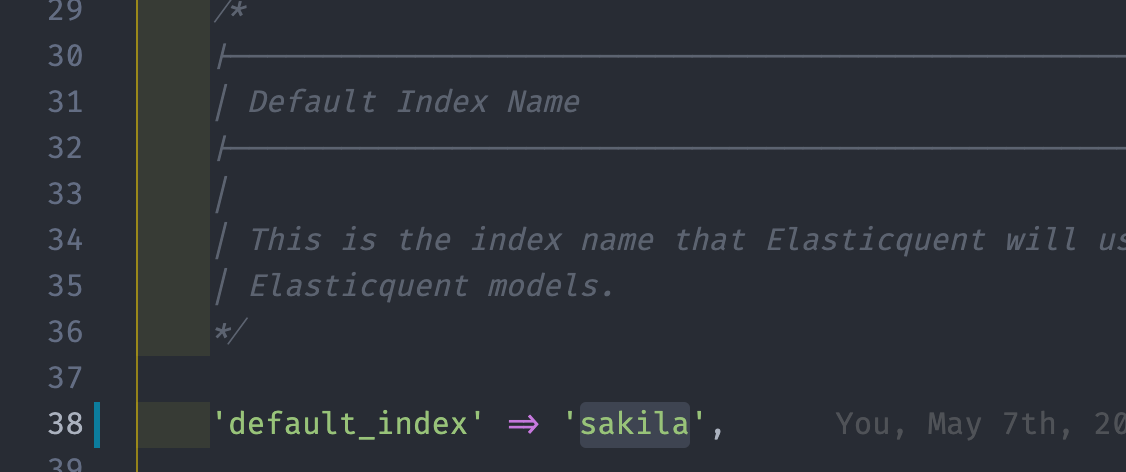


Figure 4: Change default index

This index will be used throughout the application, each model will correspond to type in Elasticsearch

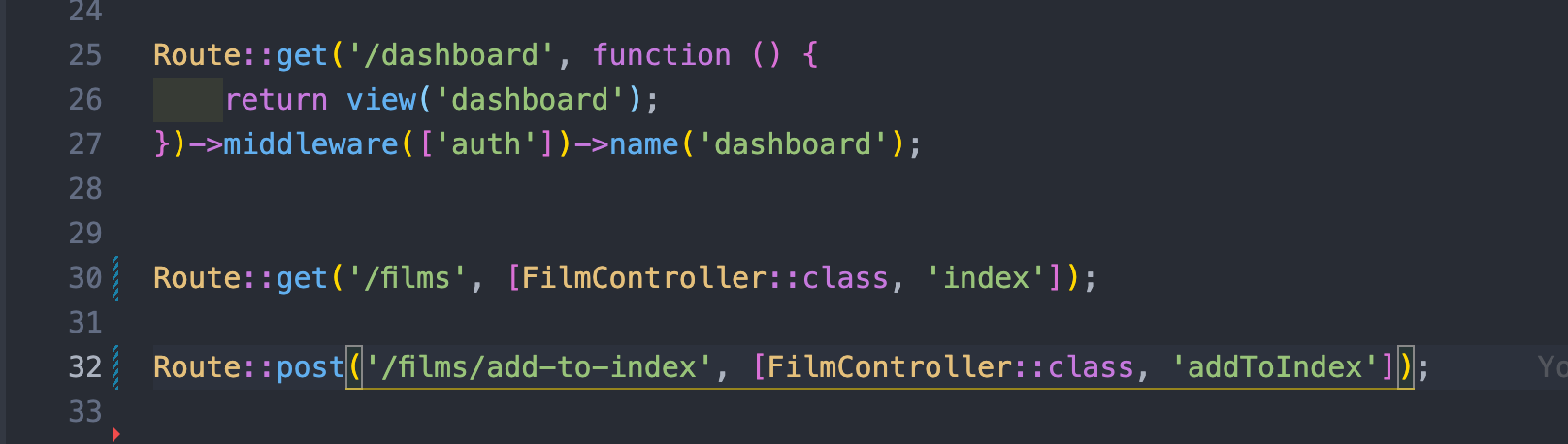


Figure 5: Route/web.php

Remember, in order to search for something in ES, we need to index it. The first step is to build index for model in ES.

To simplify things, add new button and create controller action to handle this

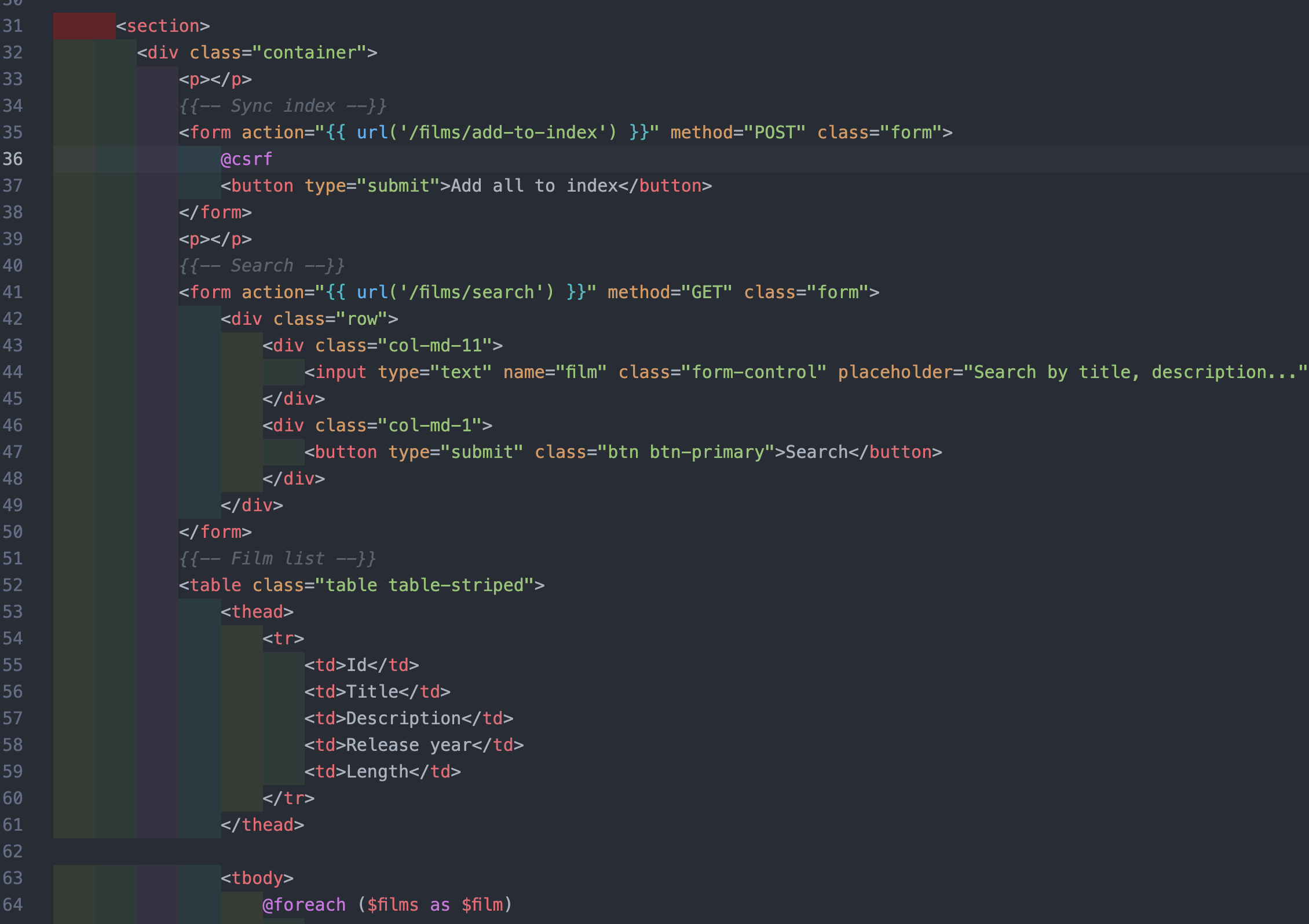


Figure 6: Create add to index

Output

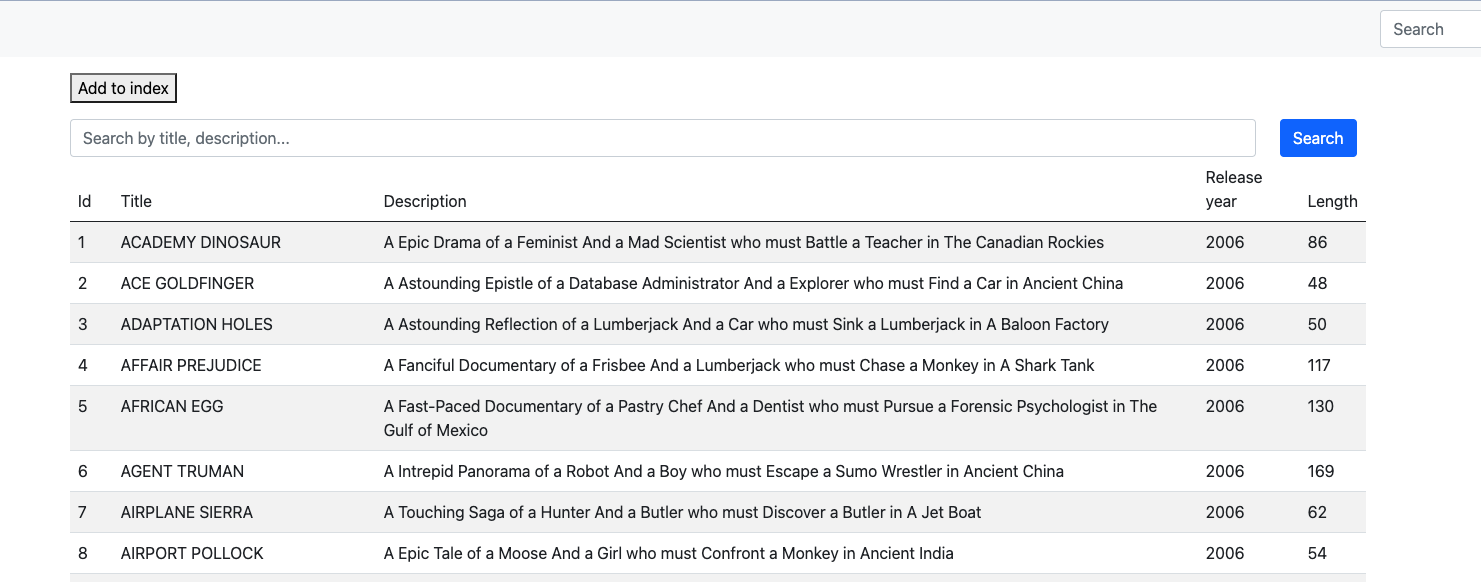


Figure 7: Add to index

## Sync films data to ElasticSearch

In class FilmController

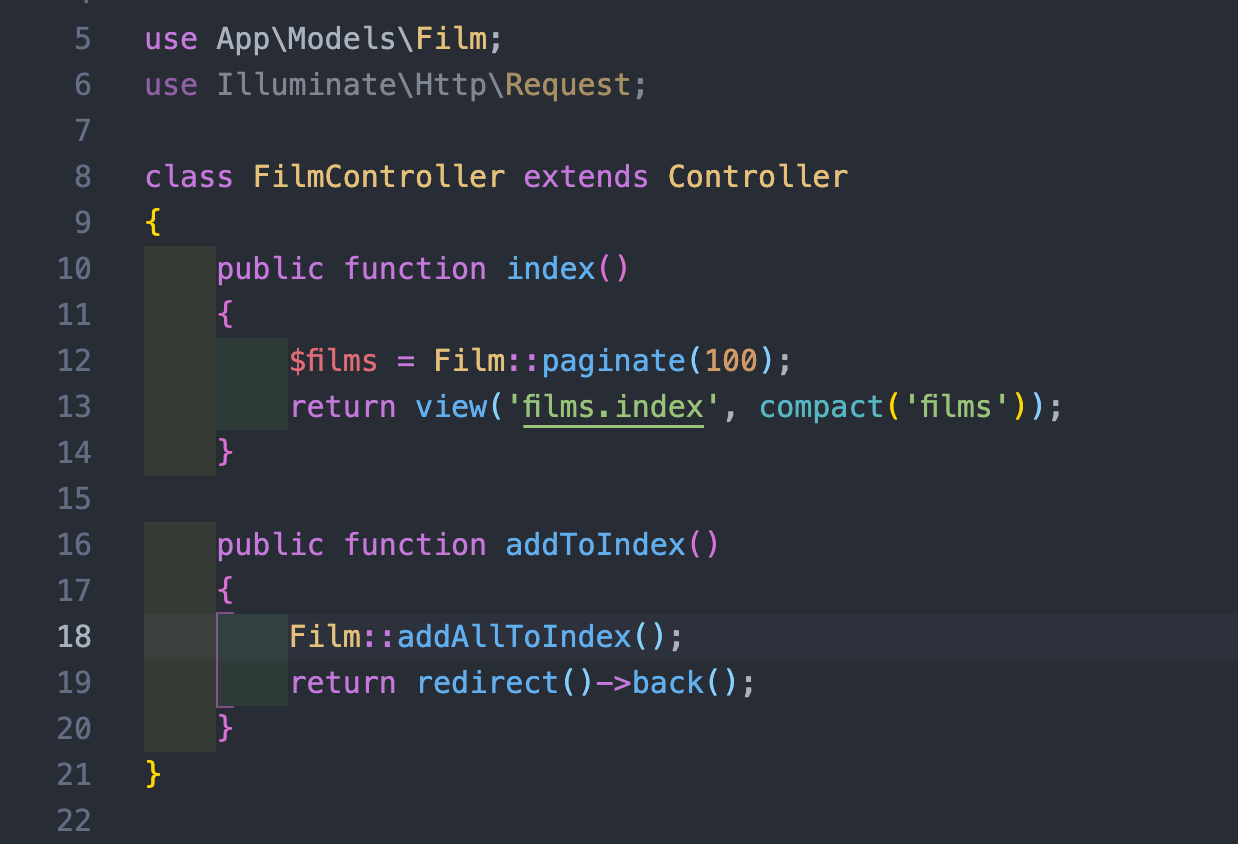


Figure 8: Add all to index

Method addAllToIndex() is for user for add every records to elasticsearch. In order to do that, make sure trait ElasticquentTrait is added to model

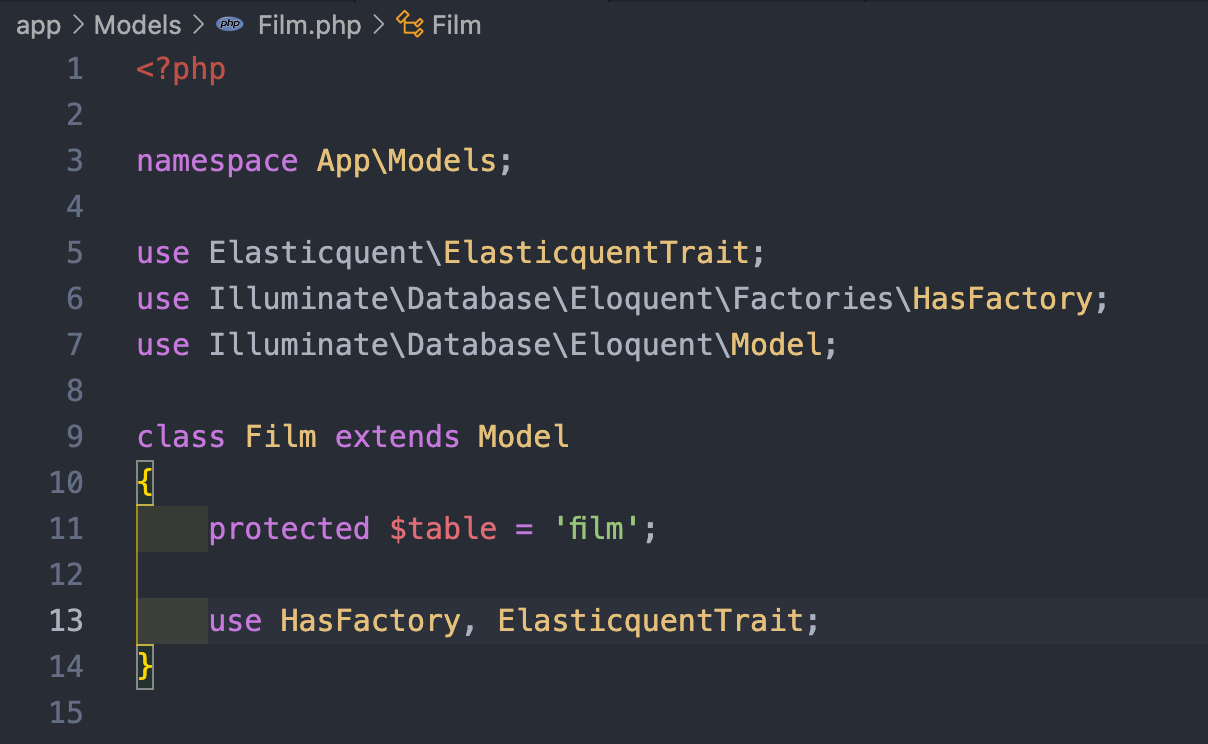


Figure 9: Add ElasticquentTrait to elasticsearch

Result



Figure 10: 1000 records are indexed, with type = films (line 19) and index sakila (Line 18)

Methods to add to index are here:

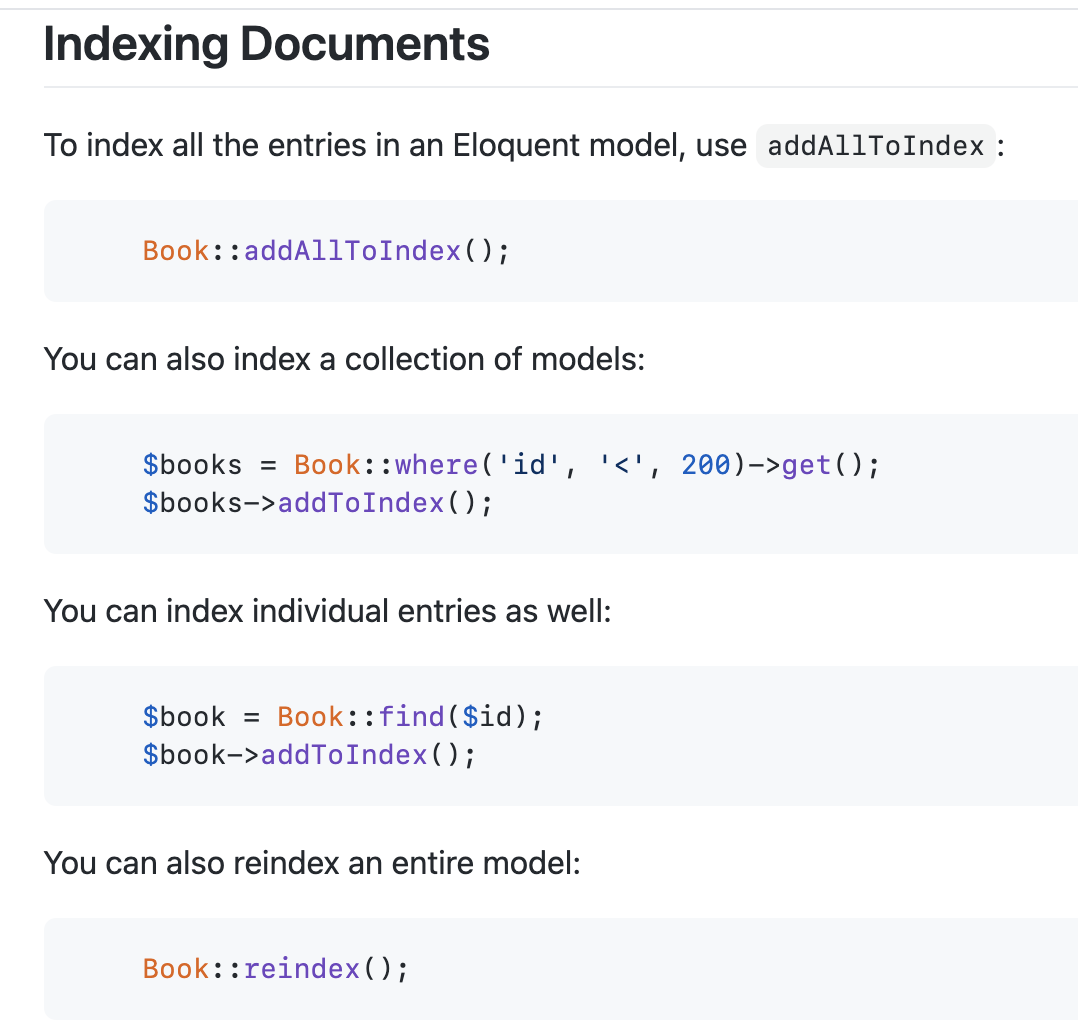


Figure 11: Book model in document to add index to ES.

## Searching in ElasticSearch

Building index to list all record inside database

Promoted UI will be

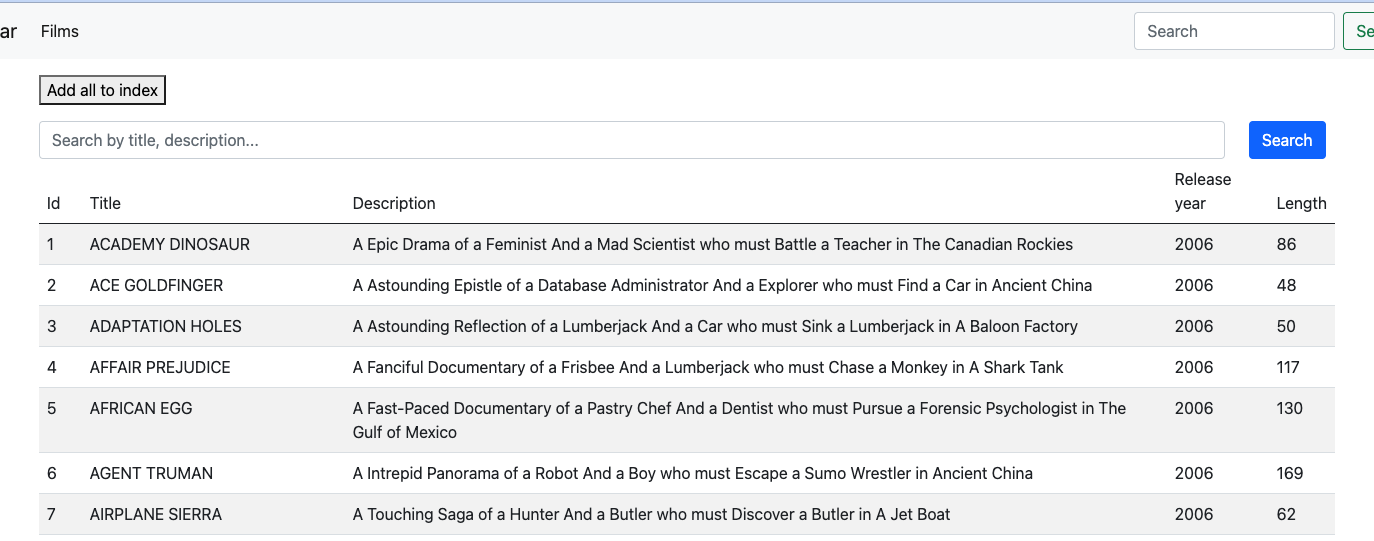


Figure 12: UI for searching

In package Elasticquent, there is an option for searchByQuery. Here, I use normal **match query** to filter result based on matching criteria

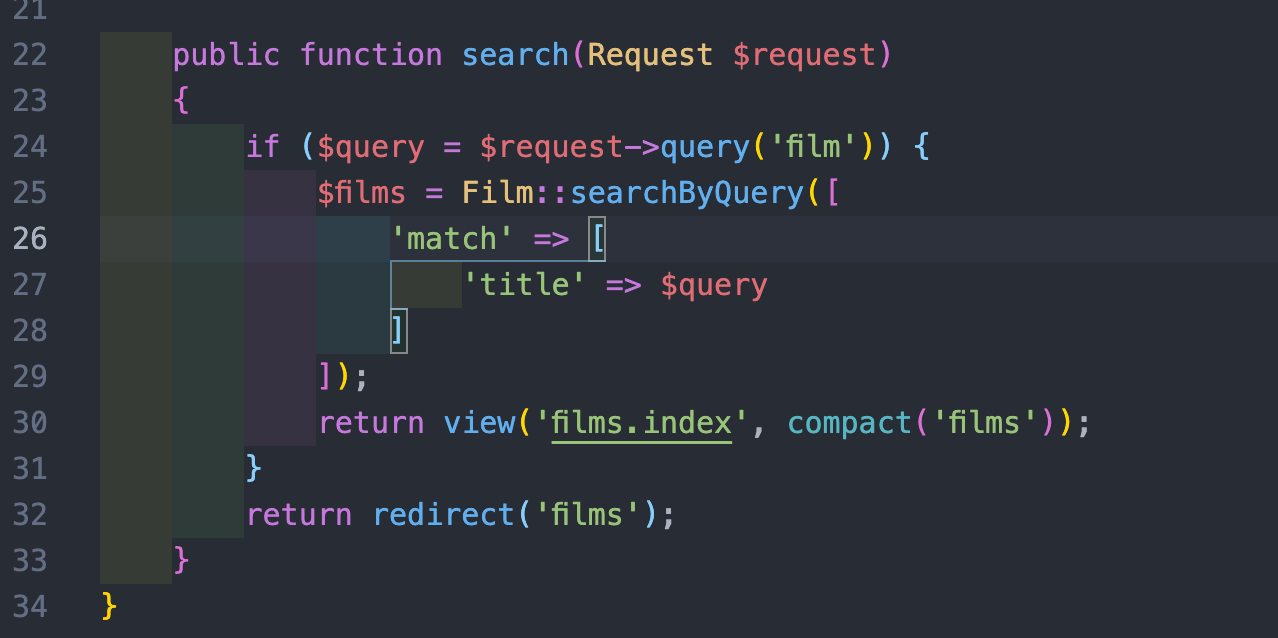


Figure 13: Normal search query using "match" criteria

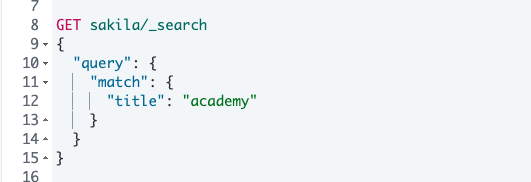


Figure 14: Normal search query in ElasticSearch

Result in ElasticSearch



Figure 15: Elasticsearch result

In Laravel, it will be

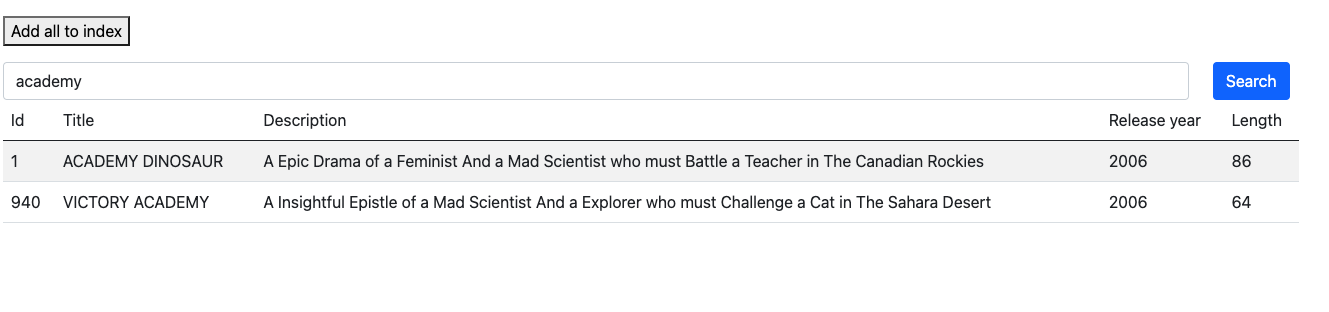


Figure 16: Laravel search result

## Multi-match search

Now it’s time to extends search for searching both title and description, using **multi\_match** in ES is designated for this situation

Setup multi\_match in ES is similar to using Kibana

Figure 17: Multi\_match in code

Total hits when searching drama is also shown in result, using totalHits() method

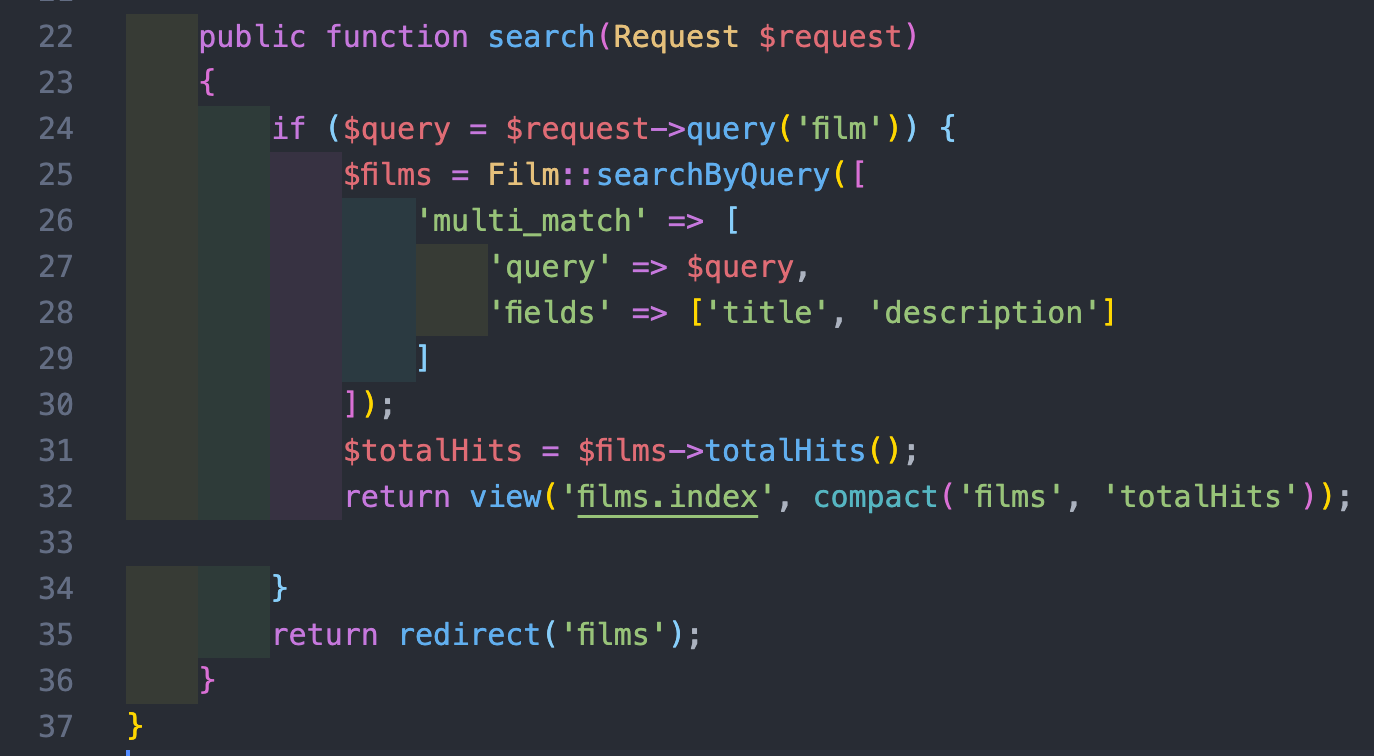


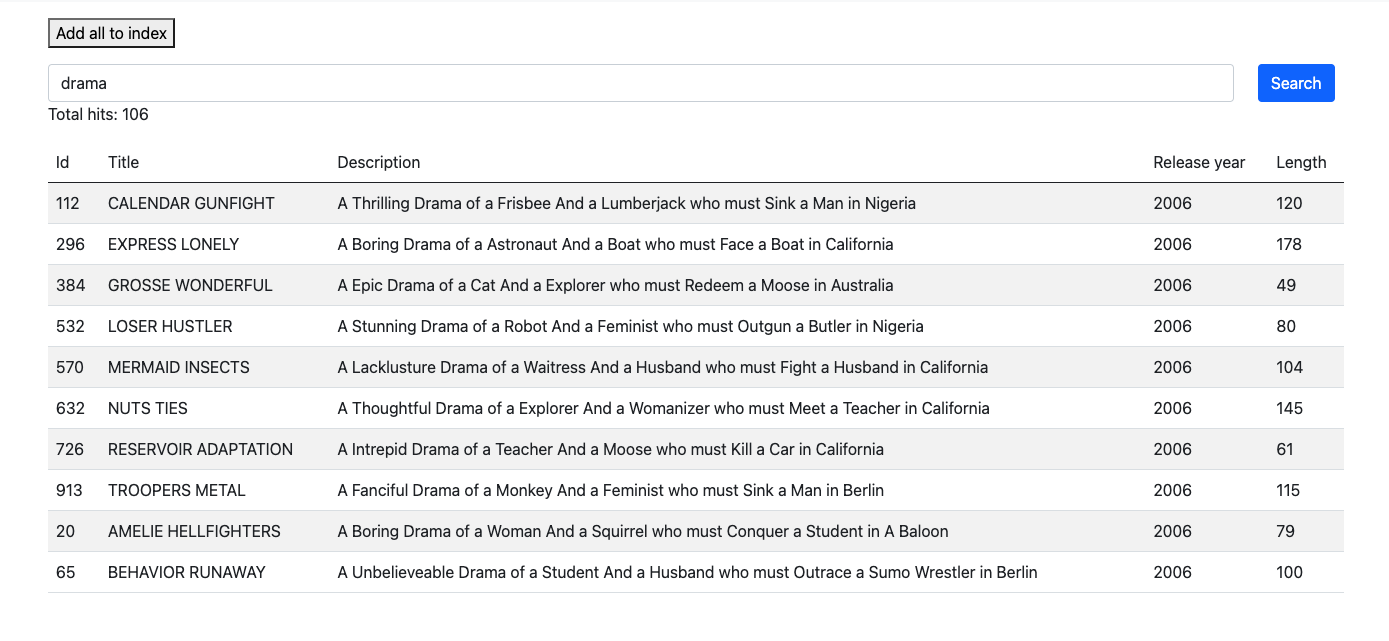
Figure 18: searching with totalHits()

Equivalent query in Elasticsearch



Figure 19: Elasticsearch query equivalent

Result after running



## Exercise

Based on tables provided in tutorial. Perform the following tasks:

* Create new film using your data of choice. Practice using addToIndex() in Elasticsearch to create new document and put in ElasticSearch

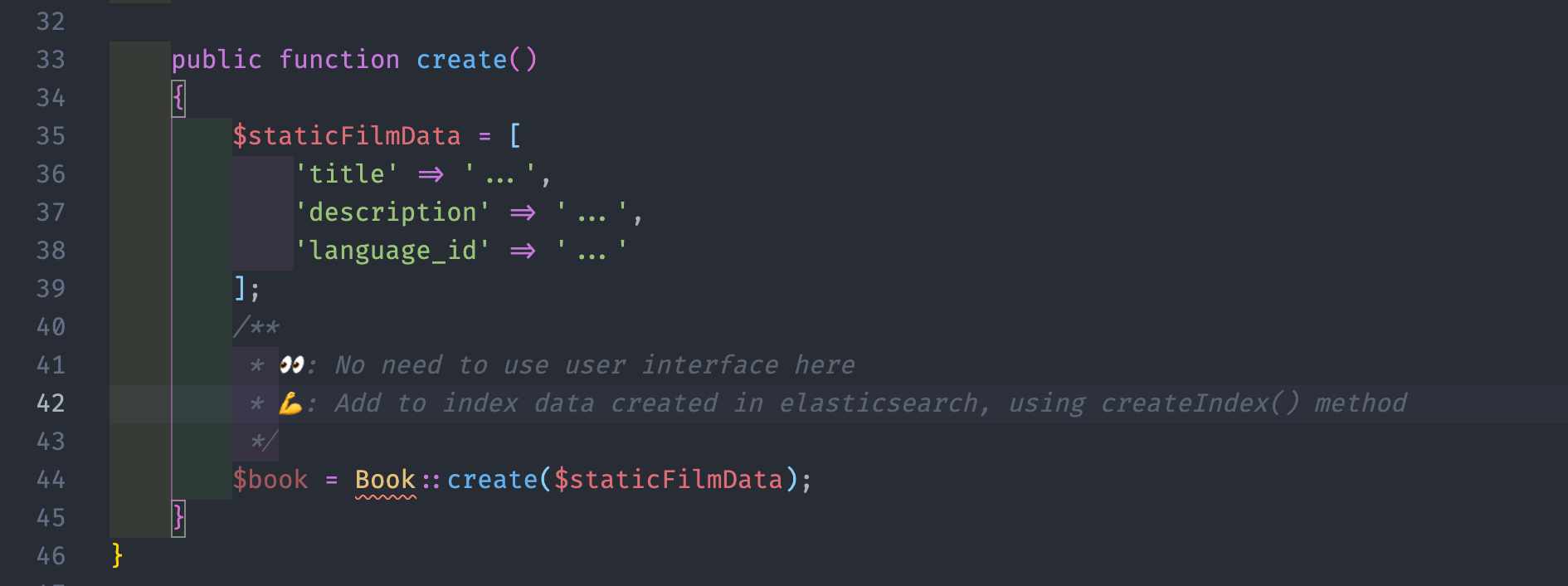


Figure 20: Add to database static data, and using createIndex() to add that record to Elasticsearch

* Using existing code, provide match query that if **title** and **description** match, title will be boosted by 2. Using **multi\_match** query with boost by 2 to title option to this

By default, ES result return 10 records (10 out of 42 for example). Specify limit property to increase this number

**Notes:** You can take a look more on method of ElasticquentTrait by going to your model 🡺 CTRL (on Windows) or COMMAND (on Mac) and click ElasticQuentTrait for more info about properties and methods can be used in Laravel model. Or lookup in doc

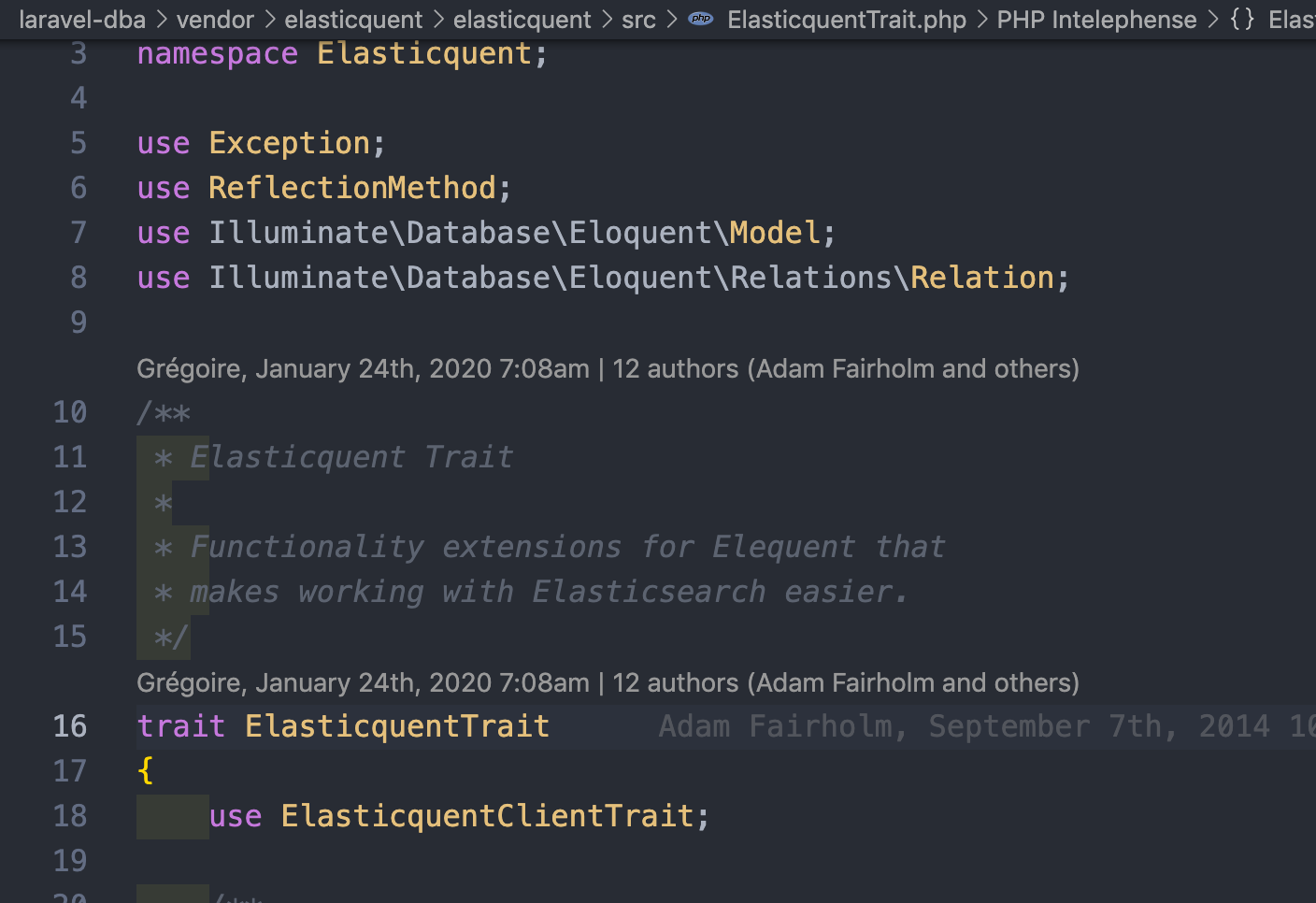


Figure 22: ElasticQuentTrait

Go to <https://github.com/elasticquent/Elasticquent> for docs.