Anagrams

Requirement

Given an input file which contains one word per line, as an output construct a list of all anagrams from that input file. Print those words to the console, where all words that are an anagram should each other should be on the same line.

Technologies

The solution uses following technologies:

Spring Boot (via Spring Initializr)
- Dependencies: Lombok

Spring Boot removes a lot of configuration work while also embedding necessary dependencies for this project.
Lombok keeps the code clean by replacing getters/setters/builder etc. with annotations.

Process logic

When executing the program, the given argument is checked. It must be a plain text file, or else the program will abort.

The text file is parsed into a list of words, which is then converted to 'Anagram' objects.
Besides the word, these contain the alphabetically ordered lower-case characters of the word as a separate String. For example the word "Hello" receive the String "ehllo".

The objects are written into a map, grouping the words by the String of ordered characters. So for each key, there is a list of matching words.
The map is then filtered, removing all entries where the word list has fewer than 2 entries.
It is also sorted by words (for readability).

The result is then printed to the console, joining the words for each line.

Performance & Scalability

As this is a single application, it can only scale vertically. Processing billions of words would cause severe performance and memory issues, requiring a very powerful and expensive machine.
To solve this it is needed to convert the application to a micro service. This allows the creation of multiple instances, each processing a part of the words.

To gather the words in one place a database is needed, so each instance can put its processed words into the DB. For many parallel inputs a NoSQL DB should be chosen like MongoDB.

The performance can then be boosted vertically by giving each instance more power and horizontally by creating more instances.

After the inputs are resolved, a separate application can query this data from the MongoDB.

For this, the I/O should be done via web based communication like REST.