Table of Contents

**[Purpose of this document:](#_Toc14617114)** [2](#_Toc14617114)

[**Problem Statement Question 2:** 2](#_Toc14617115)

[**Expected Result:** 2](#_Toc14617116)

[**Overview of the System:** 2](#_Toc14617117)

[**Assumptions:** 2](#_Toc14617118)

[**Architecture:** 3](#_Toc14617119)

[**Step by Step Algorithm Flow:** 3](#_Toc14617120)

[**Improvements:** 4](#_Toc14617121)

[**Process Diagram:** 4](#_Toc14617122)

## **Purpose of this document:**

This document contains the high level design details of Categories. It includes the architecture of the system and step by step process of algorithm to solve the problem.

## **Problem Statement Question 2:**

Assume a database of categories, and keywords associated with a category, with an N-to-N

relation between them. You need to analyse a file attachment (such as .pdf, .doc, .xml), and

find the most fitting category for this file, based on the file's contents, and number of

keyword occurrences per category.

Categories & Keywords for example:

Category: Computers; Keywords: Networking, Keyboard, Mouse, Processor, RAM

Category: Clothing; Keywords: Pants, Shoes, T-Shirt, Dress Shirt, Socks

Please propose the software design(architecture) plus algorithm to solve the problem above.

Please make sure to have detailed explanation about each step.

Example of category object:

{

"\_id": "catrgoey\_dfdfdf72924003be162c340370ecfb",

"\_type": "category",

“title”:”Computers”,

“Keywords”:[

“Networking”,” Keyboard”, “Mouse”, ‘Processor”, “RAM ”

],

"creation\_date": 1485344457000

}

Expected result:

## **Expected Result:**

Process diagram, description of the process and reasoning.

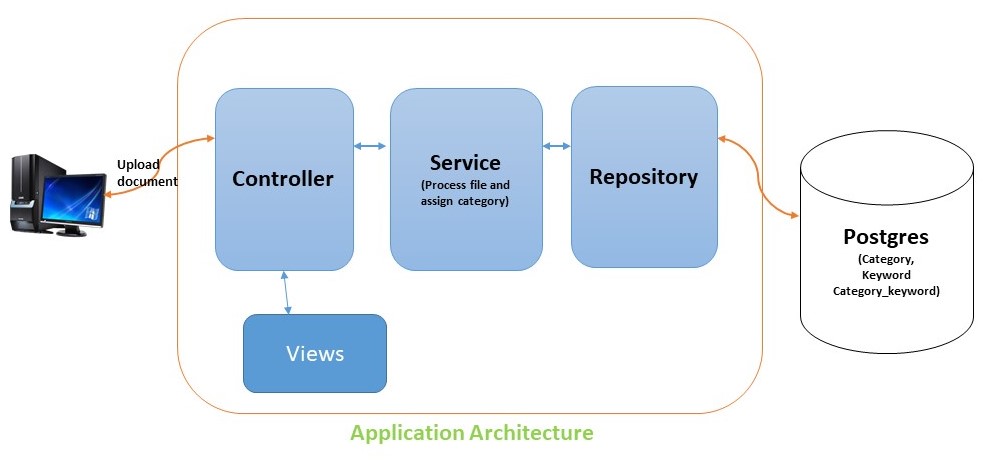
## **Overview of the System:**

System reads the file and analyse each word and assign the appropriate category to it based on the keywords in the document.

## **Assumptions:**

1. Category and Keyword **tables** already there in the database.
2. Category and Keyword has N-N relationship by using another table **category\_keywords.**

## **Architecture:**



## **Step by Step Algorithm Flow:**

**Step 1:**

Start

**Step 2:**

Fetch category contents from DB and assign them to a list

**Step 3:**

Get the Keywords from DB object list and put them in a Map (Key as Keyword and count 0)

**Step 4:**

Read File contents from disk using BufferReader

**Ex: BufferedReader br = new BufferedReader (new FileReader("file"));**

**Step 5:**

Read line from bufferreader

**Step 6:**

Split the line content with space ( )

**Step 7:**

Validate the split keywords with DB category keywords map

**Step 8:**

If the keyword contains in map increase the count by one to the corresponding keyword in map

**Step 9:**

Repeat the step 7 and 8

**Step 10:**

Sort the map by value using custom Comparator

**Step 11:**

Iterate the final Map and find the largest value count

**Step 12:**

Repeat step 11 and find for second largest

**Step 13:**

If both value counts are same repeat step 11 for third largest

**Step 14:**

Search for title in category objects using the keywords (top two keywords), assign title to a variable

**Step 15:**

Save the file the corresponding category

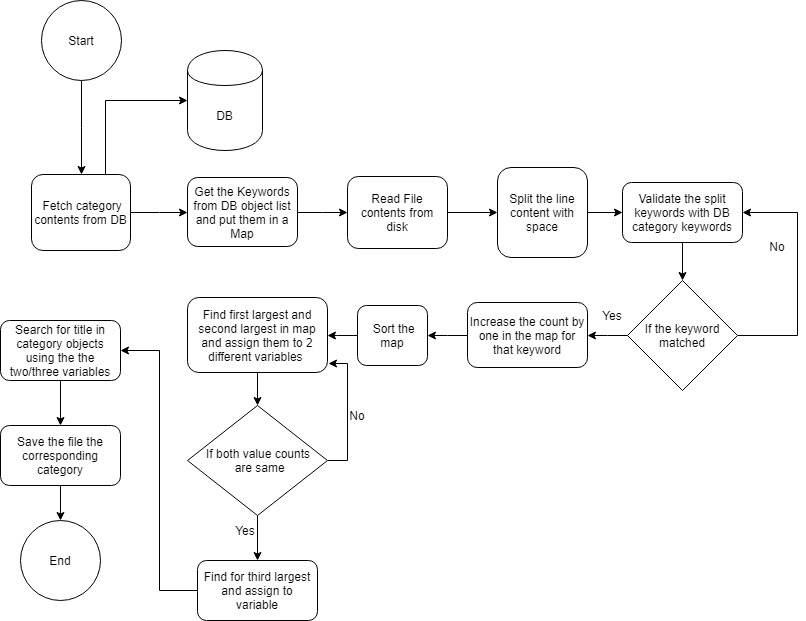
**Step 16:**

End

## **Improvements:**

1. If the file very large, we can use **Multi-Threading** process. For each line we can use one thread to process (step 6 – step 9).
2. By using Multi-Threading we can reduce the processing time.
3. We can use the cache to hold the category and its keywords from database.

## **Process Diagram:**

****