```
/**
* This is my code! ItâM-^@M-^Ys goal is to ....
* CS 312 - Assignment 9
* @Michael Higgins
* Version 1.0
*/


import java.util.HashMap;
import java.util.Set;
import java.util.HashSet;


public class SearchEngine extends HashMap<String,Set<String>>{

    HashMap<String,Set<String>> reverseIndex;


    public SearchEngine(Setup s){

        reverseIndex = s.returnIndexAsMap();

    }
    public String search(String search){

        String[] WordList = search.split(" ");

        Set<String> docs = reverseIndex.getOrDefault(WordList[0],null);

            for(int i = 1;i<WordList.length;i++){

                if(reverseIndex.containsKey(WordList[i])){

                    docs.retainAll(reverseIndex.get(WordList[i]));
                }
        if(!reverseIndex.containsKey(WordList[i])){

            return "Words not present in list";

        }

           }
       return docs.toString();
    }
}
```



```
/**
 * This is my code! ItÃ¢âM-^B¬âM-^D¢s goal is to ....
 * CS 312 - Assignment 9
 * @Michael Higgins
 * version 1.1
 */

import java.util.HashMap;
import java.util.HashSet;
import java.util.PriorityQueue;
import java.util.Scanner;
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Set;
public class Setup {

    HashMap<String,Set<String>> map;

    HashSet<String> stopList;

    public Setup(PriorityQueue<String> docList, String stopAdr) throws FileNotFo
undException{

        stopList = setupStoplist(stopAdr);

        map = setupFile(docList);

    }

    public HashSet<String> setupStoplist(String stopAdr) throws FileNotFoundExce
ption{

        File stopFile = new File(stopAdr);

        Scanner myStopReader = new Scanner(stopFile);

        String data = "";

        HashSet<String> localSet = new HashSet<String>();

        while (myStopReader.hasNextLine()) {


            data = myStopReader.nextLine();

            localSet.add(data);

        }
        myStopReader.close();

        return localSet;
    }



    public HashMap<String,Set<String>>  setupFile(PriorityQueue<String> docQueue
) throws FileNotFoundException{

        HashMap<String,Set<String>> map = new HashMap<String, Set<String>>();

        while(docQueue.peek() != null){

            File doc = new File(docQueue.poll());

            Scanner myReader = new Scanner(doc);



            while(myReader.hasNextLine()) {
```

```
                String[] priority = myReader.nextLine().split(" ");

                for (String data : priority) {

                    data = data.toLowerCase();


                    if (checkStopList(data)) {

                        if (!map.containsKey(data)) {

                            HashSet<String> Collection = newCollection(doc.getNa
me());

                            map.put(data, Collection);
                        } else {
                            map.get(data).add(doc.getName());


                        }

                    }

                }

            }
        }
        return map;
    }
    public boolean checkStopList(String word){

        return !stopList.contains(word);

    }
    public String returnIndexasString(){

        return map.toString();

    }
    public HashSet<String> newCollection(String adr){

        HashSet<String> set = new HashSet<String>();

        set.add(adr);

        return set;


    }
    public HashMap<String,Set<String>> returnIndexAsMap(){

        return map;

    }
}
```



```
/**
* This is my code! ItâM-^@M-^Ys goal is to ....
* CS 312 - Assignment 9
* @Michael Higgins
* Version 1.0
*/

import java.io.FileNotFoundException;
import java.util.Scanner;
import java.util.PriorityQueue;
public class CLI {
    public static void main(String[] args) throws FileNotFoundException{

    long startTime = System.currentTimeMillis();

    String stopListPath="";

    PriorityQueue<String> docQueue = new PriorityQueue<String>();

    if(!args[0].equals("-d")){

        stopListPath = args[0];

        for(int i = 1; i < args.length; i++){

            docQueue.add(args[i]);



        }
    }

    if(args[0].equals("-d")){

        stopListPath = args[1];

        for(int i = 2;i<args.length;i++){

            docQueue.add(args[i]);


        }

    }

    Setup s = new Setup(docQueue,stopListPath);

    long stopTime = System.currentTimeMillis();

    long elapsedTime = stopTime - startTime;

    System.out.println("@@ build time " + elapsedTime + "ms");

    Scanner sc = new Scanner(System.in);

    String input = sc.nextLine();

    long startTime2 = System.currentTimeMillis();

    SearchEngine se = new SearchEngine(s);

    System.out.println(se.search(input));

    if(args[0].equals("-d")){

        System.out.print(s.returnIndexasString());
    }
    long stopTime2 = System.currentTimeMillis();
```

```
    long elapsedTime2= stopTime2- startTime2;

    System.out.println("finding the current word took"+elapsedTime2+"ms");

    }
}
```