**Implementation Code**

**WebDisplay.html:**

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Revive</title>

</head>

<body>

<p> <h1>

<centre>Revive yourself from daily work with movies and books</centre>

</h1></p>

<form id=*"Movie Details"* name=*"Movie Details"*>

<label>Specify the Genre:</label>

<select id=*"Genre"*>

<option value=*"select one from below"*>Select one option</option>

<option value=*"Adventure"*>Adventure</option>

<option value=*"War"*>War</option>

<option value=*"Action"*>Action</option>

<option value=*"Sci-Fi"*>Sci-Fi</option>

<option value=*"Biography"*>Biography</option>

<option value=*"Comedy"*>Comedy</option>

<option value=*"Drama"*>Drama</option>

<option value=*"Music"*>Music</option>

</select>

<button id=*"search"*>Search</button>

<br />

</form>

<p>Movies of the specified Genre</p>

<div id=*"results1"*> <div>

<p>Recommended Books of the specified Genre</p>

<div id=*"results"*></div>

</body><

<script src=*"http://ajax.googleapis.com/ajax/libs/jquery/1.4.2/jquery.min.js"*></script>

<script>

**function** on\_data(data) {

$('#results1').empty();

**var** users = data.response.docs;

$.each(users, **function**(id,title) {

**var** endlist = 'Recommended Movies'+users;

$('#results1').prepend('<div>' + endlist + '</div>');

}

}

**function** on\_search() {

**var** query = $('#Genre').val();

**if** (query.length == 0) {

**return**;

}

**var** url='http://134.193.136.127:8983/solr/collection1\_shard1\_replica1/select?q=\*query\*&wt=json&indent=true';

$.getJSON(url, **function** on\_data(data) {

$('#results1').empty();

**var** users = data.response.docs;

$.each(users, **function**(id,title) {

**var** endlist = 'Recommended Movies'+users;

$('#results1').prepend('<div>' + endlist + '</div>');

}

});

}

**function** on\_ready() {

$('#search').click(on\_search);

$('body').keypress(**function**(kp) {

**if** (kp.keyCode == '13') {

on\_search();

}

});

}

$(document).ready(on\_ready);

</script>

</html>

**Main.Java:**

**import** java.io.BufferedReader;

**import** java.io.BufferedWriter;

**import** java.io.FileNotFoundException;

**import** java.io.FileReader;

**import** java.io.FileWriter;

**import** java.io.IOException;

**import** java.util.ArrayList;

**import** java.util.HashMap;

**import** java.util.Iterator;

**import** java.util.List;

**import** java.util.Map;

**public** **class** Main {

**public** **static** **void** main(String[] args) **throws** IOException {

// **TODO** Auto-generated method stub

**int** temp=0;/\*List<String> sl= new ArrayList<String>();\*/

Map<String, String> map = **new** HashMap<>();

//String s="0000417::Le voyage dans la lune (1902)::Short|Adventure|Fantasy|Short";

BufferedReader reader = **new** BufferedReader(**new** FileReader("/home/cloudera/Desktop/Data/movies/movies.dat"));

String line = **null**;

**while** ((line = reader.readLine()) != **null**) {

//System.out.println(line);

temp=0;List<String> sl= **new** ArrayList<String>();

String[] sp=line.split("::",3 );

String last=sp[2];

String movie=sp[1];

//System.out.println(last);

**for**(**int** i=0;i<last.length();i++)

{

**if**(last.charAt(i)=='|')

{

**if**(!sl.contains(last.substring(temp, i))) {

sl.add(last.substring(temp, i));

}

temp=i+1;

}

}

**if**(!sl.contains(last.substring(temp, last.length()))) {

sl.add(last.substring(temp, last.length()));

}

Iterator<String> iterator = sl.iterator();

**while** (iterator.hasNext()) {

String g=iterator.next();

**if**(g.compareTo("")==0)

g="General";

//System.out.println(g);

**if**(!map.isEmpty())

{

**if**(map.containsKey(g))

{

//System.out.println(g);

//System.out.println(map.get(g));

String content=map.get(g)+","+movie;

map.put(g, content);

}

**else**

{

map.put(g, movie);

}

}

**else**

{

//System.out.println(g);

map.put(g, movie);

}

}

// ...

}

BufferedWriter reader1 = **new** BufferedWriter(**new** FileWriter("/home/cloudera/Desktop/Data/movies/output1.json"));

System.*out*.println(map.size()+"\n");

Iterator<Map.Entry<String, String>> entries = map.entrySet().iterator();

reader1.write("{");

**while** (entries.hasNext()) {

Map.Entry<String, String> entry = entries.next();

**if**(entry.getKey().compareTo("Drama")==0)

System.*out*.println("Key = " + entry.getKey() + ", Value = " + entry.getValue());

String entireline="\"" + entry.getKey() +"\":{\""+ "movie"+"\":"+"\""+ entry.getValue()+"\""+"},"+"\n";

reader1.write(entireline);

}

reader1.write("}");

reader1.close();reader.close();

}

/\* (non-Java-doc)

\* @see java.lang.Object#Object()

\*/

**public** Main() {

**super**();

}

}

**Recommendation:**

MyRecommender.java:

package com.bookrecommender;

import java.io.File;

import java.io.FileNotFoundException;

import java.util.List;

import java.io.IOException;

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import org.apache.commons.cli2.OptionException;

import org.apache.mahout.cf.taste.common.TasteException;

import org.apache.mahout.cf.taste.impl.m.file.FileDataModel;

import org.apache.mahout.cf.taste.impl.recommender.CachingRecommender;

import org.apache.mahout.cf.taste.impl.recommender.slopeone.SlopeOneRecommender;

import org.apache.mahout.cf.taste.m.DataModel;

import org.apache.mahout.cf.taste.recommender.RecommendedItem;

import org.apache.mahout.cf.taste.impl.common.LongPrimitiveIterator;

public class MyBookRecommender {

public static void main(String... args) throws FileNotFoundException, TasteException, IOException, OptionException {

// create data source (m) - from the csv file

File rate = new File("/home/cloudera/Books\_Ratings.csv");

DataModel m = new FileDataModel(rate);

// create a simple recommender on our data

CachingRecommender cr = new CachingRecommender(new SlopeOneRecommender(m));

// for all users

for (LongPrimitiveIterator it = m.getUserIDs(); it.hasNext();){

long id = it.nextLong();

// get the recommendations for the user

List<RecommendedItem> recommendations = cr.recommend(id, 10);

// if empty write something

if (recommendations.size() == 0){

System.out.print("User ");System.out.print(id);System.out.println(": no recommendations");

}

// print the list of recommendations for each

for (RecommendedItem recommendedItem : recommendations) {

System.out.print("User ");

System.out.print(id);

System.out.print(": ");

System.out.println(recommendedItem);

}

}

}

}

TestMyApp.java:

package com.unresyst;

import junit.framework.Test;

import junit.framework.TestCase;

import junit.framework.TestSuite;

/\*\*

\* Unit test for simple App.

\*/

public class TestMyApp

extends TestCase

{

/\*\*

\* Create the test case

\*

\* @param testName name of the test case

\*/

public TestMyApp( String testName )

{

super( testName );

}

/\*\*

\* @return the suite of tests being tested

\*/

public static Test suite()

{

return new TestSuite( TestMyApp.class );

}

/\*\*

\* Rigourous Test :-)

\*/

public void testApp()

{

assertTrue( true );

}

}