<Medical Semantic Search Engine Program – 2016>

Medical Semantic Search Engine Program – 2016

</Medical Semantic Search Engine Program – 2016>

A Console Application

INFT8F2H2 - PROGRAMMING MEDICAL INFORMATICS SYSTEMS

ASSIGNMENT 03 : SOFTWARE : MEDICAL SEMANTIC SEARCH ENGINE PROGRAM - 2016

208501583 – Zakia Salod – zakia.salod@ukzn.ac.za

This document serves as documentation / an overview of the Software developed as part of Assignment 03 of this module. It includes an "About" of this program, the Technologies Used, Sample Run of Program, Source Code, & About the Programmer.

| *Faculty of Health Sciences* |

| *School of Nursing & Public Health* |

| *University of Kwa-Zulu Natal, Durban, South Africa* |

INFT8F2H2 - PROGRAMMING MEDICAL INFORMATICS SYSTEMS

ASSIGNMENT 03 : SOFTWARE : MEDICAL SEMANTIC SEARCH ENGINE PROGRAM - 2016

ABOUT : MEDICAL SEMANTIC SEARCH ENGINE PROGRAM – 2016 . . .

WHAT IS IT?

The Medical Semantic Search Engine Program - 2016 is a software used to tag keywords that are found in a text file, and to also search the database for files on the system with the medical term entered by the user. The program reads-in a text file, for example, a medical journal article, Doctor's notes and finds pre-defined keywords, and tags them. The program then writes this tagged information onto a new .TXT file and stores them onto a "medicaloutputfiles" directory. The pre-defined keywords are read from an (.OWL) Ontology file, which was created using Protege. Optionally, the user has a facility to also enter medical term/s to find all files (papers) on the system that has this medical term/s. The program does this by consulting the Ontology file, and then using the hierarchy from there, together with the search term entered by the user, and then searches the database for all relevant files. If the term is found on the database, then the program displays all file (paper) information, in the order of the most relevant - i.e. in terms of the date in which the file was loaded onto the system - in reverse chronological order (latest entries first).

This program was created by the programmer, as part of a Medical Informatics Programming course at the University of KwaZulu-Natal Nelson R Mandela School of Medicine campus in South Africa in the year 2016.

NOTE

INPUT :

Sample Medical Text File/s & (.OWL) Ontology file can be found here :

- medicalinputfiles\sampleinputtxtfile1.txt
- medicalinputfiles\sampleinputtxtfile2.txt
- medicalinputfiles\sampleinputtxtfile3.txt

Sample Medical Keywords Text File can be found here :

- medicalinputfiles\medicalterm.owl

OUTPUT :

ABSTRACT

This document serves as documentation / an overview of the Software developed as part of Assignment 03 of this module. It includes an "About" of this program, the Technologies Used, Sample Run of Program, Source Code, & About the Programmer.

FROM TAGGING :-

DIRECTORY :

o Output file/s are stored in the "medicaloutputfiles" directory.

o If the "medicaloutputfiles" directory does not exist, the program will create this directory - for the first run of this program.

o Subsequent runs of the program will store output files onto this directory.

FILE/S :

o The user is prompted for the output file name. If the file name already exists in the "medicaloutputfiles" directory, the program continues to prompt the user for a unique file name.

FROM USER SEARCH TERM ENTERED :-

CONSOLE :

o File (paper) information from the database, that satisfies the search term/s entered by the user.

CONTACT

o If you would like more info about the Medical Semantic Search Engine Program, or require free support for this program, please contact the programmer at http://www.zakiasalod.weebly.com or zakia.salod@gmail.com

---

TECHNOLOGIES USED . . .

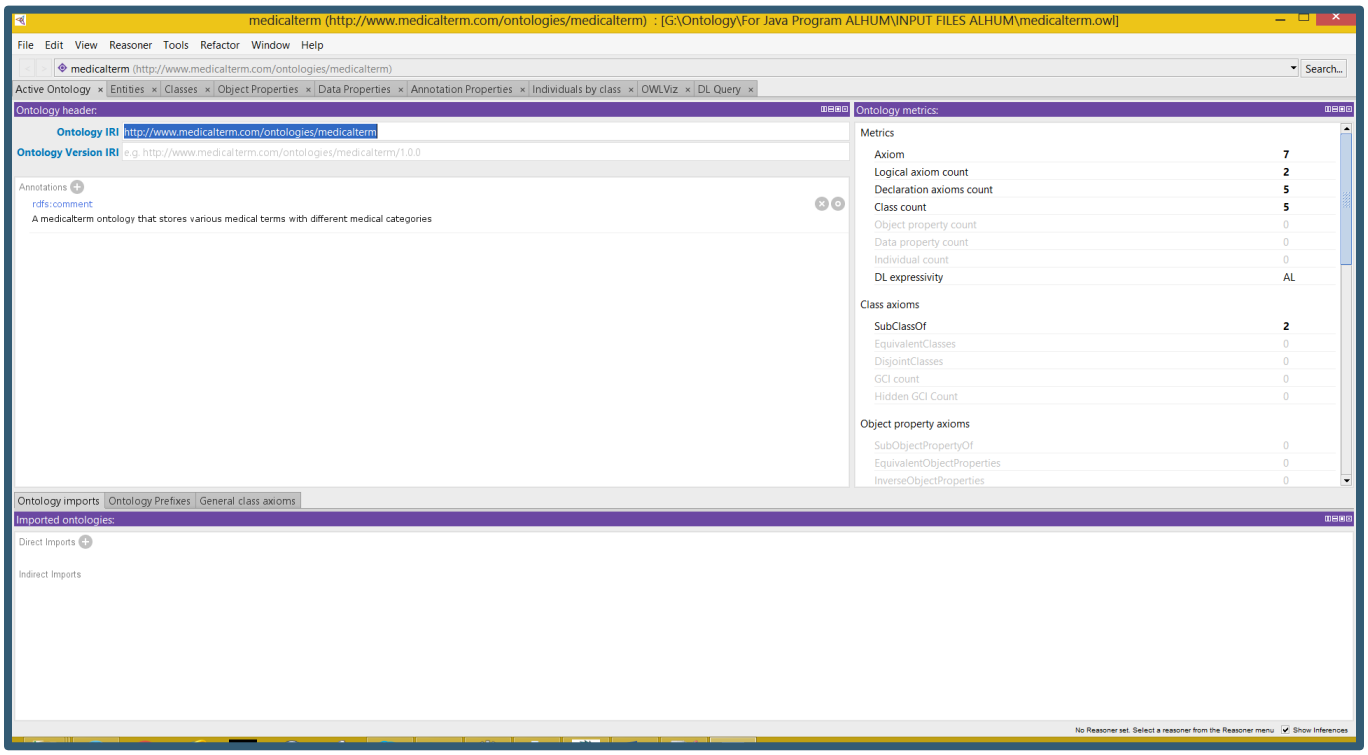
- ✓ Programming Language : Java
- ✓ Integrated Development Environment (IDE) : Eclipse LUNA
- ✓ JRE : 1.8.0_45
- ✓ JDK : 1.8.0_45
- ✓ Webservice : USBWebserver v8.6
- ✓ Database : MySQL
- ✓ Mysql-connector : mysql-connector-java-5.1.40-bin.jar
- ✓ Protégé 5.1.0
- ✓ JENA API Library

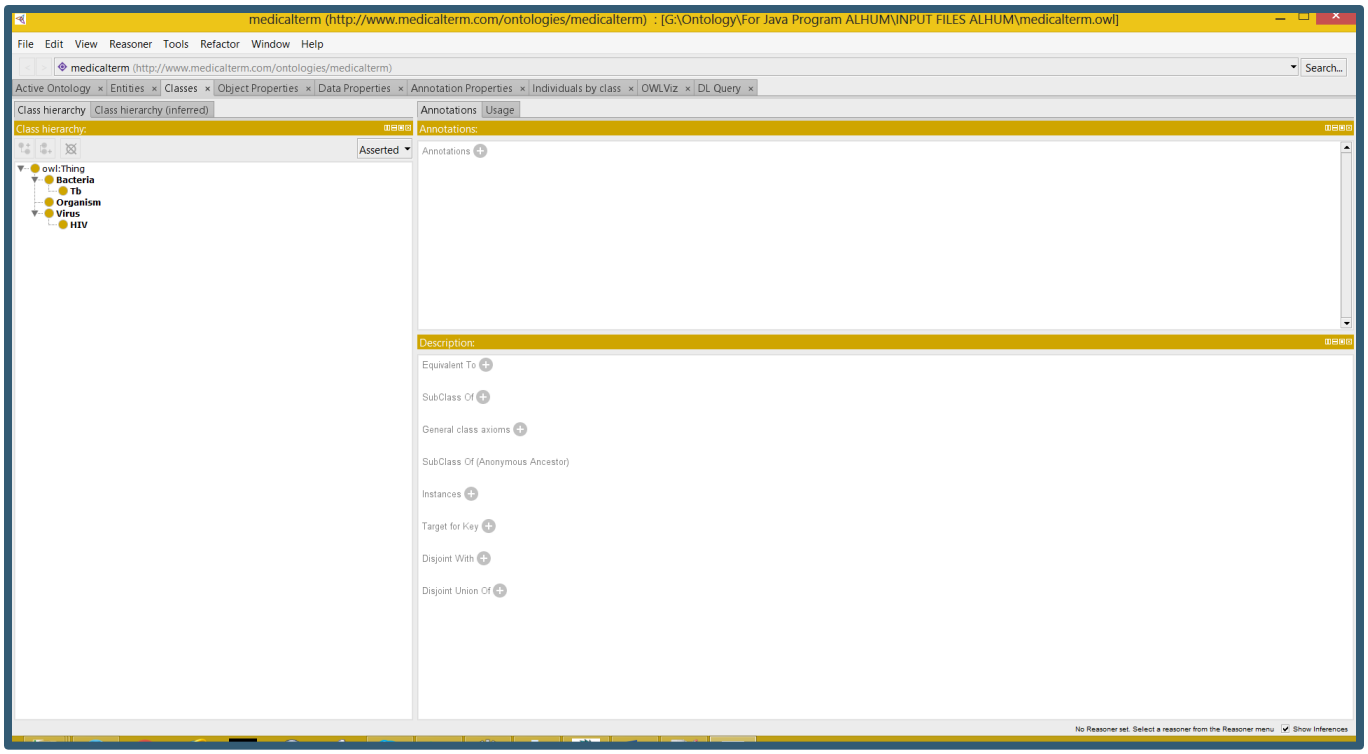
SAMPLE RUN OF PROGRAM . . .

INPUT

MEDICALINPUTFILES\MEDICALTERM.OWL

{Opened in Protégé}

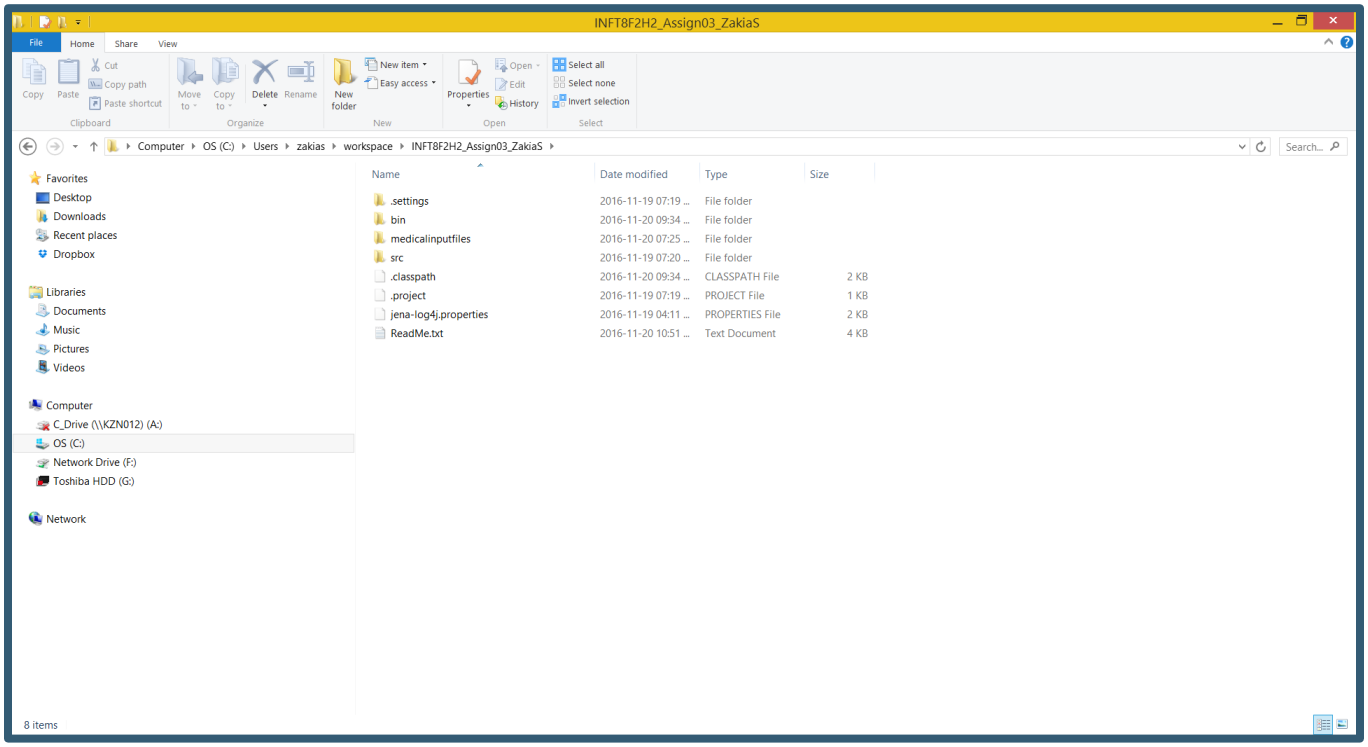




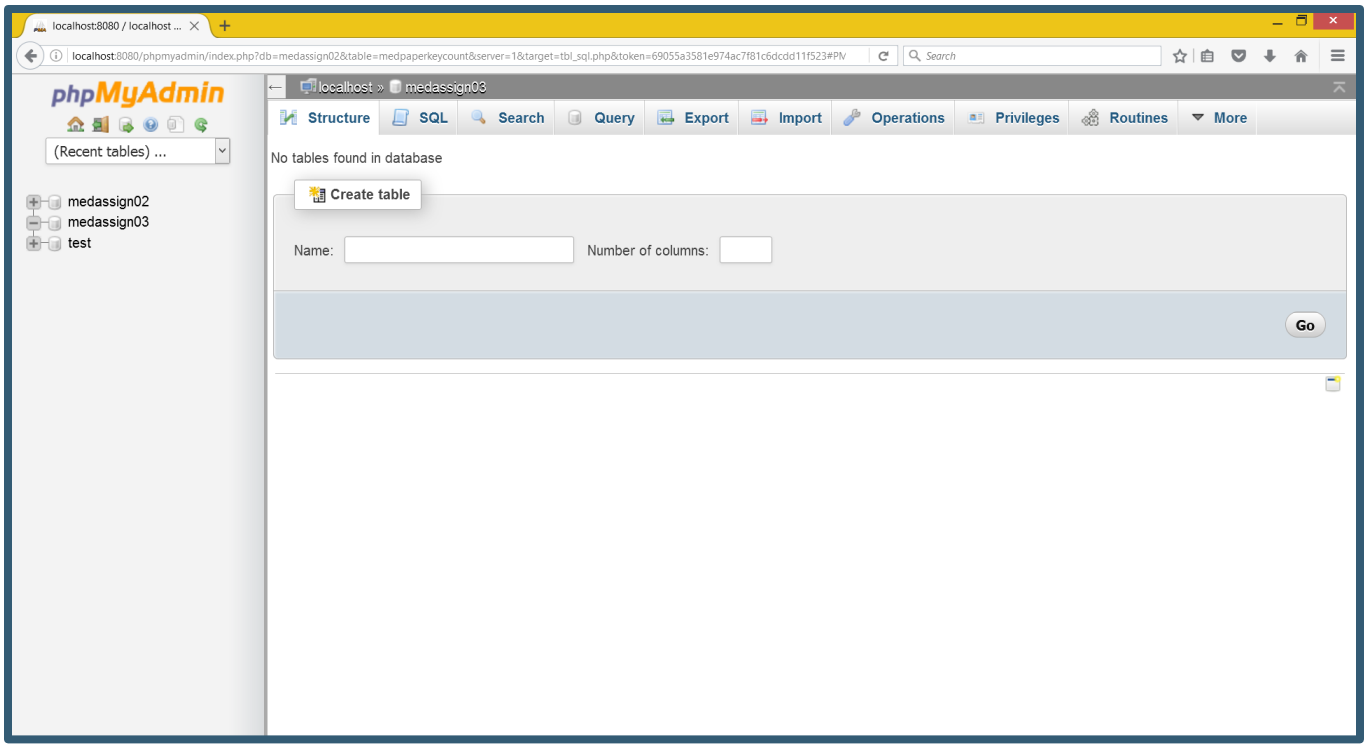
MEDICALINPUTFILES\SAMPLEINPUTTXTFILE1.TXT

Organism Bacteria Tb Virus HIV

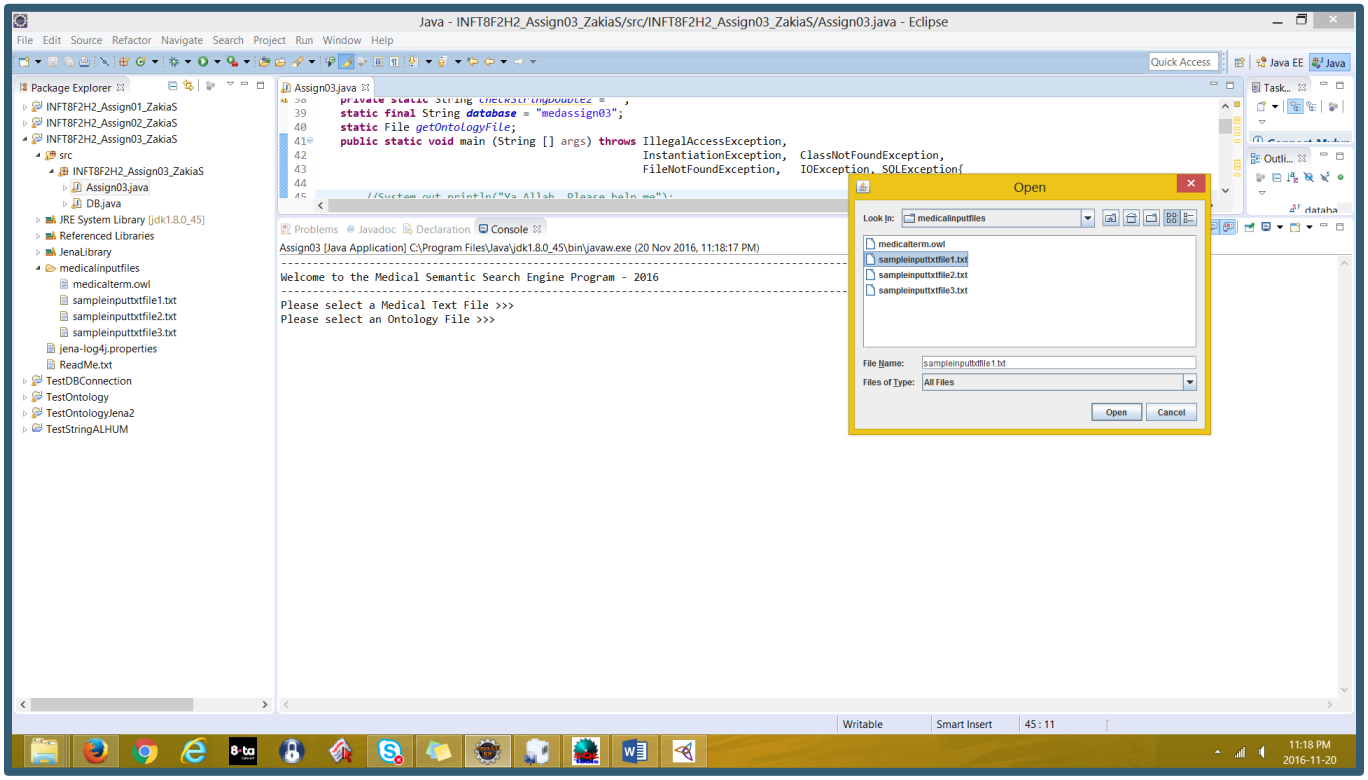
BEFORE



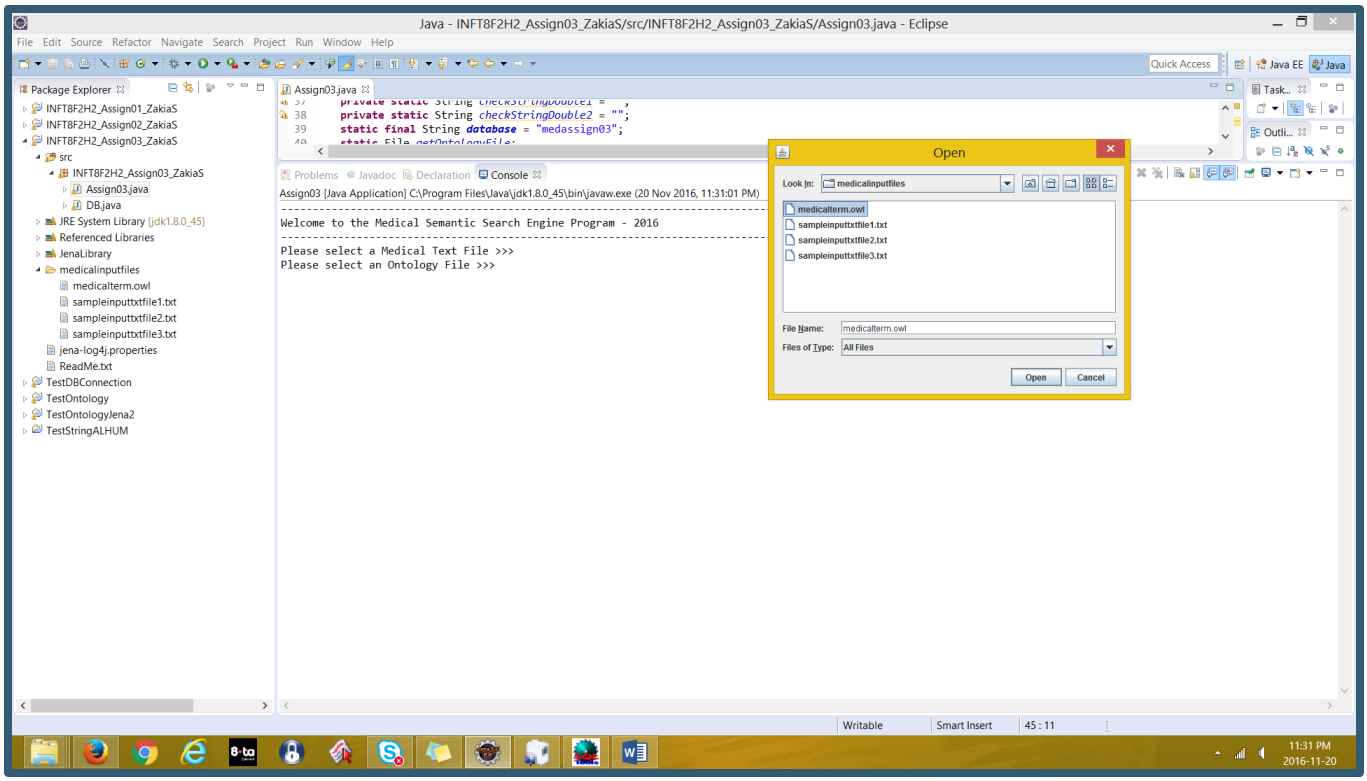
medontologylog Database Table Does NOT Exist in medassign03 Database (Note the medassign03 database node is expanded below, but it does not contain any tables currently)



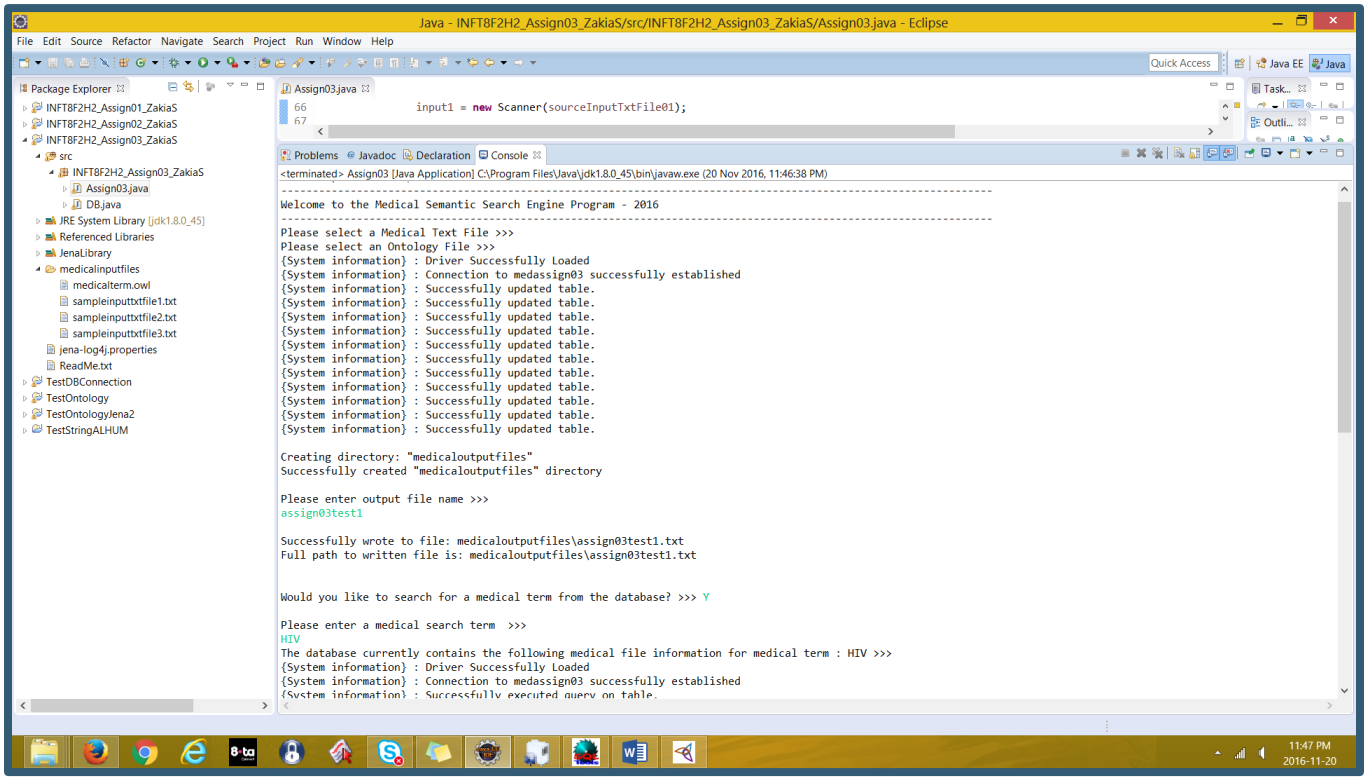
JOURNAL

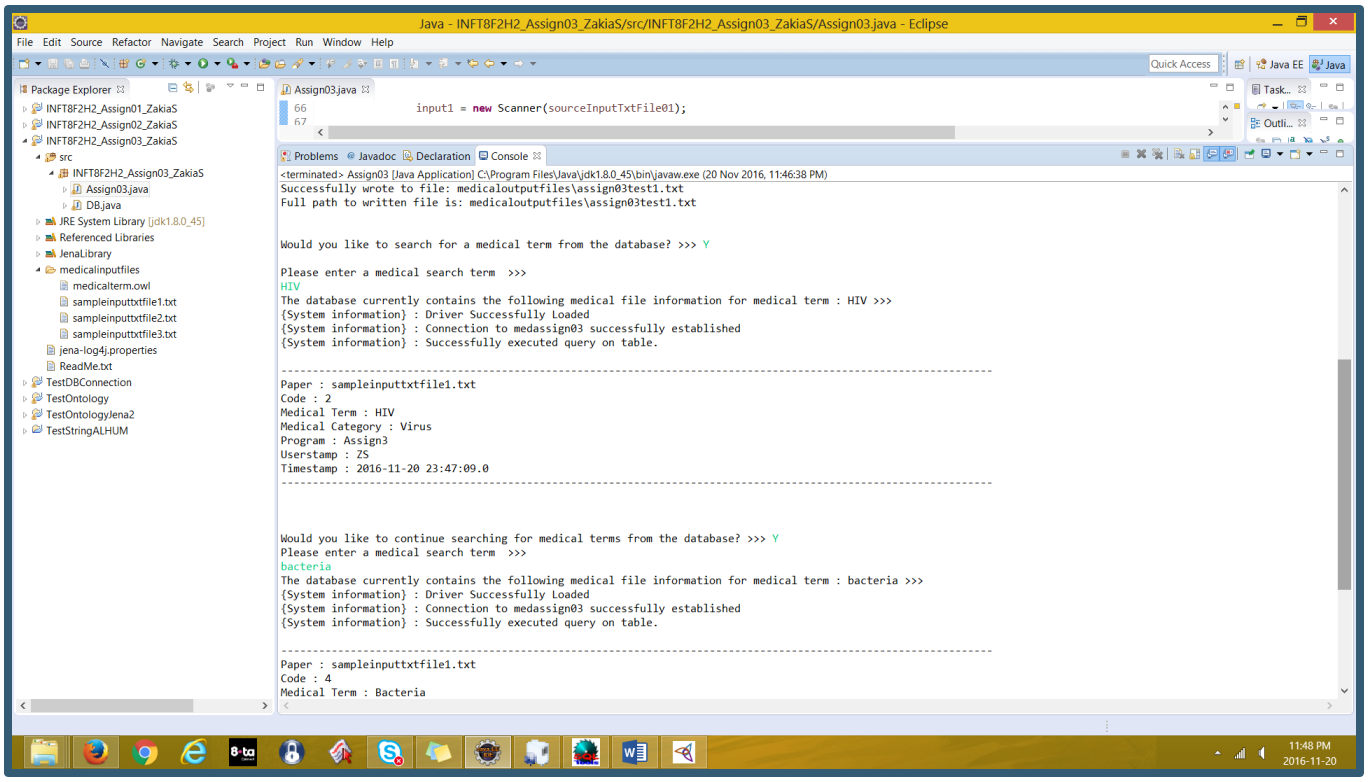


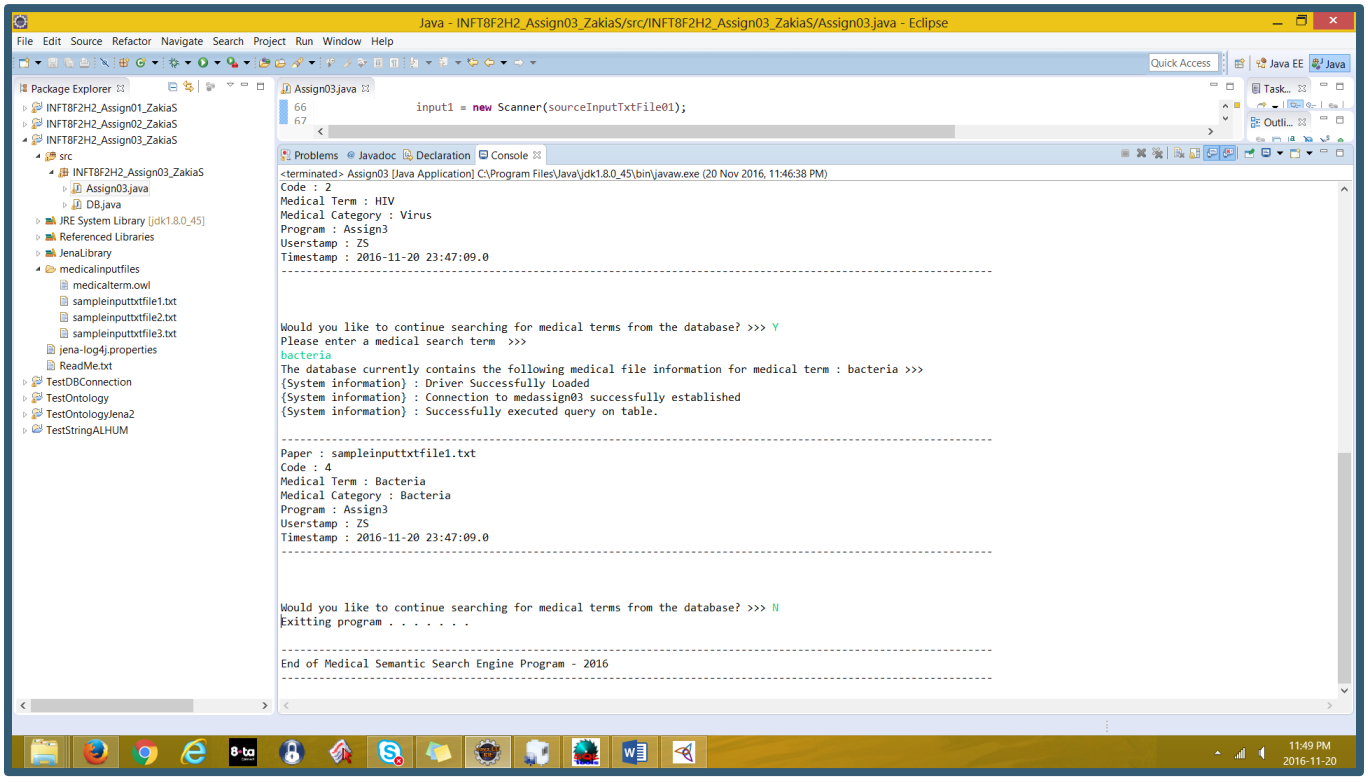
Click 'Open' on the 'Open' modal dialog box



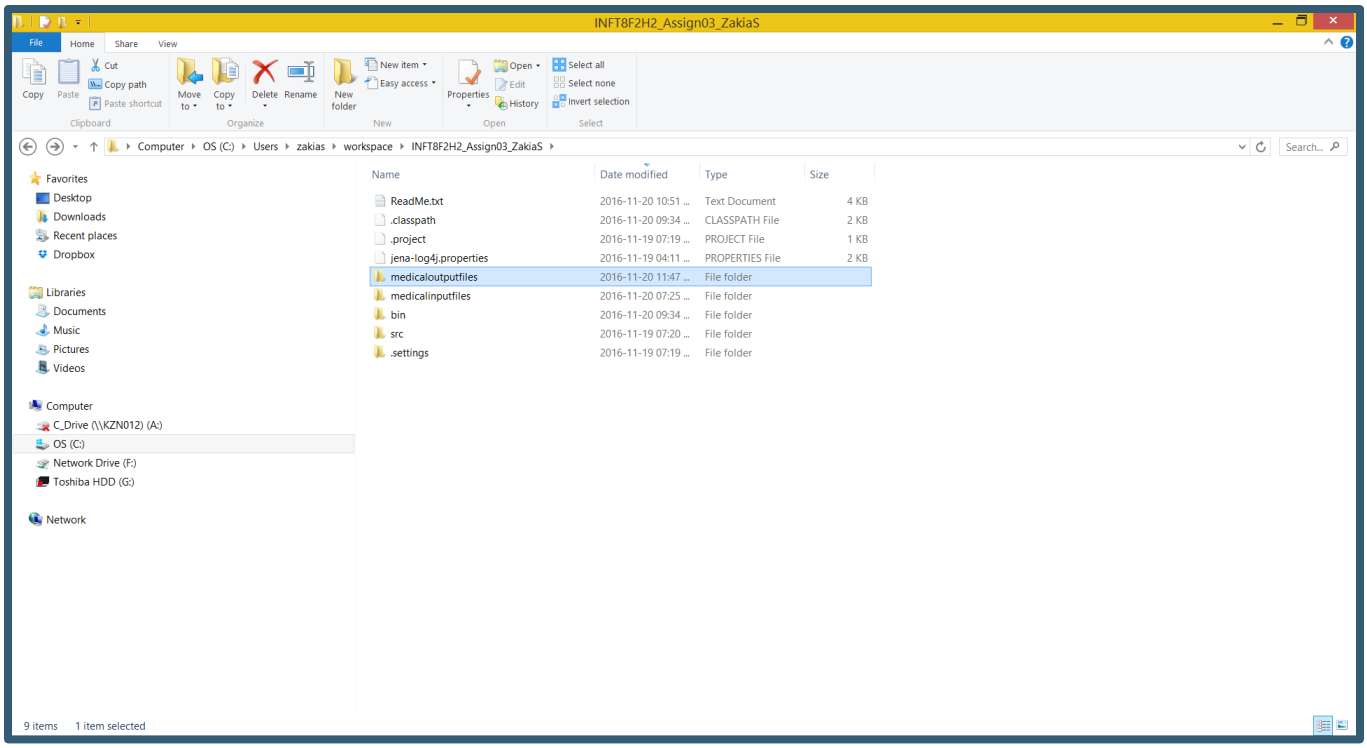
Click 'Open' on the 'Open' modal dialog box



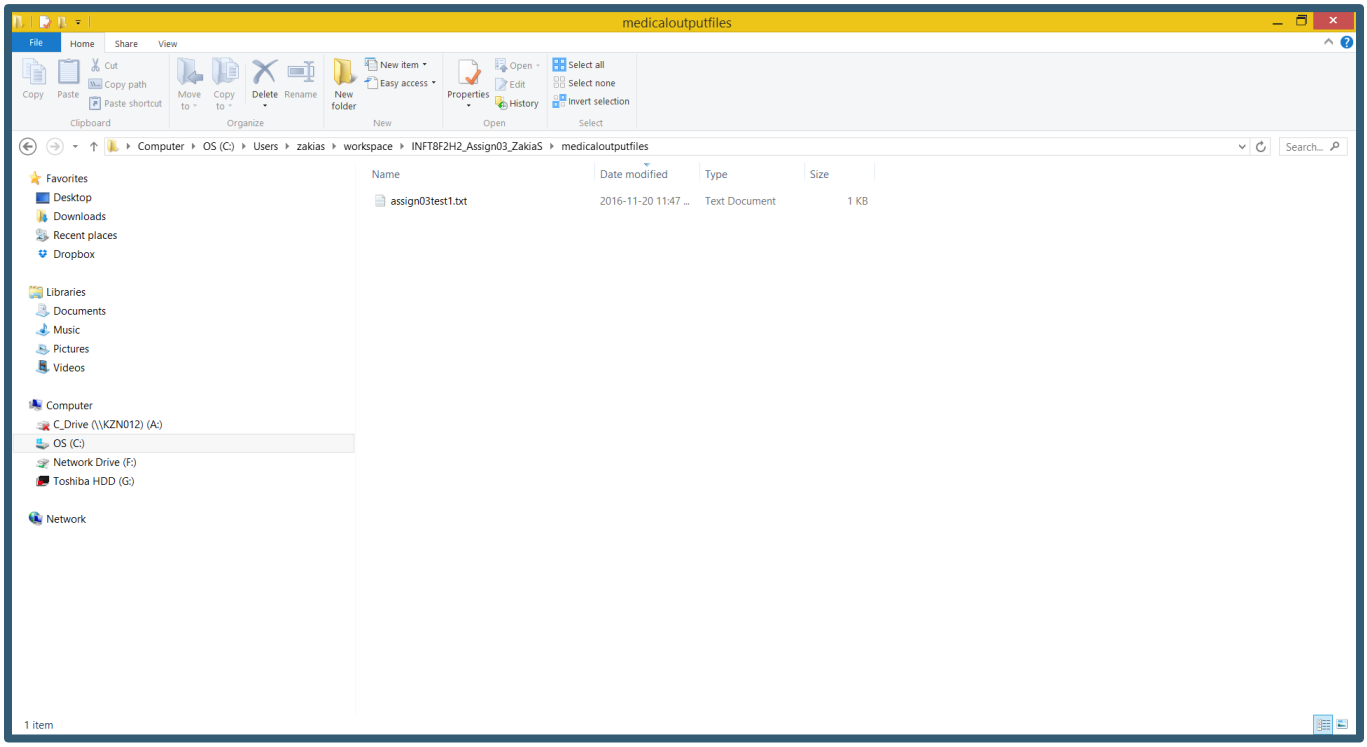

Java - INFT8F2H2_Assign03_ZakiaS/src/INFT8F2H2_Assign03_ZakiaS/Assign03.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
INFT8F2H2_Assign01_ZakiaS
INFT8F2H2_Assign02_ZakiaS
INFT8F2H2_Assign03_ZakiaS
src
INFT8F2H2_Assign03_ZakiaS
Assign03.java
DB.java
JRE System Library [jdk1.8.0_45]
Referenced Libraries
JenaLibrary
medicalinputfiles
medicalterm.owl
sampleinputtxtfile1.txt
sampleinputtxtfile2.txt
sampleinputtxtfile3.txt
jena-log4j.properties
ReadMe.txt
TestDBConnection
TestOntology
TestOntologyJena2
TestStringALHUM
Assign03.java
66 input1 = new Scanner(sourceInputTxtFile01);
67
Problems Javadoc Declaration Console
<terminated> Assign03 [Java Application] C:\Program Files\Java\jdk1.8.0_45\bin\javaw.exe (20 Nov 2016, 11:46:38 PM)
Successfully wrote to file: medicaloutputfiles\assign03test1.txt
Full path to written file is: medicaloutputfiles\assign03test1.txt
Would you like to search for a medical term from the database? >>> Y
Please enter a medical search term >>>
HIV
The database currently contains the following medical file information for medical term : HIV >>>
{System information} : Driver Successfully Loaded
{System information} : Connection to medassign03 successfully established
{System information} : Successfully executed query on table.
Paper : sampleinputtxtfile1.txt
Code : 2
Medical Term : HIV
Medical Category : Virus
Program : Assign3
Userstamp : ZS
Timestamp : 2016-11-20 23:47:09.0
Would you like to continue searching for medical terms from the database? >>> Y
Please enter a medical search term >>>
bacteria
The database currently contains the following medical file information for medical term : bacteria >>>
{System information} : Driver Successfully Loaded
{System information} : Connection to medassign03 successfully established
{System information} : Successfully executed query on table.
Paper : sampleinputtxtfile1.txt
Code : 4
Medical Term : Bacteria
11:48 PM
2016-11-20


Java - INFT8F2H2_Assign03_ZakiaS/src/INFT8F2H2_Assign03_ZakiaS/Assign03.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
INFT8F2H2_Assign01_ZakiaS
INFT8F2H2_Assign02_ZakiaS
INFT8F2H2_Assign03_ZakiaS
src
INFT8F2H2_Assign03_ZakiaS
Assign03.java
DB.java
JRE System Library [jdk1.8.0_45]
Referenced Libraries
JenaLibrary
medicalinputfiles
medicalterm.owl
sampleinputtxtfile1.txt
sampleinputtxtfile2.txt
sampleinputtxtfile3.txt
jena-log4j.properties
ReadMe.txt
TestDBConnection
TestOntology
TestOntologyJena2
TestStringALHUM
Assign03.java
66 input1 = new Scanner(sourceInputTxtFile01);
67
Problems Javadoc Declaration Console
<terminated> Assign03 [Java Application] C:\Program Files\Java\jdk1.8.0_45\bin\javaw.exe (20 Nov 2016, 11:46:38 PM)
Code : 2
Medical Term : HIV
Medical Category : Virus
Program : Assign3
Userstamp : ZS
Timestamp : 2016-11-20 23:47:09.0
Would you like to continue searching for medical terms from the database? >>> Y
Please enter a medical search term >>>
bacteria
The database currently contains the following medical file information for medical term : bacteria >>>
{System information} : Driver Successfully Loaded
{System information} : Connection to medassign03 successfully established
{System information} : Successfully executed query on table.
Paper : sampleinputtxtfile1.txt
Code : 4
Medical Term : Bacteria
Medical Category : Bacteria
Program : Assign3
Userstamp : ZS
Timestamp : 2016-11-20 23:47:09.0
Would you like to continue searching for medical terms from the database? >>> N
Exitting program . . . . . .
End of Medical Semantic Search Engine Program - 2016
11:49 PM
2016-11-20

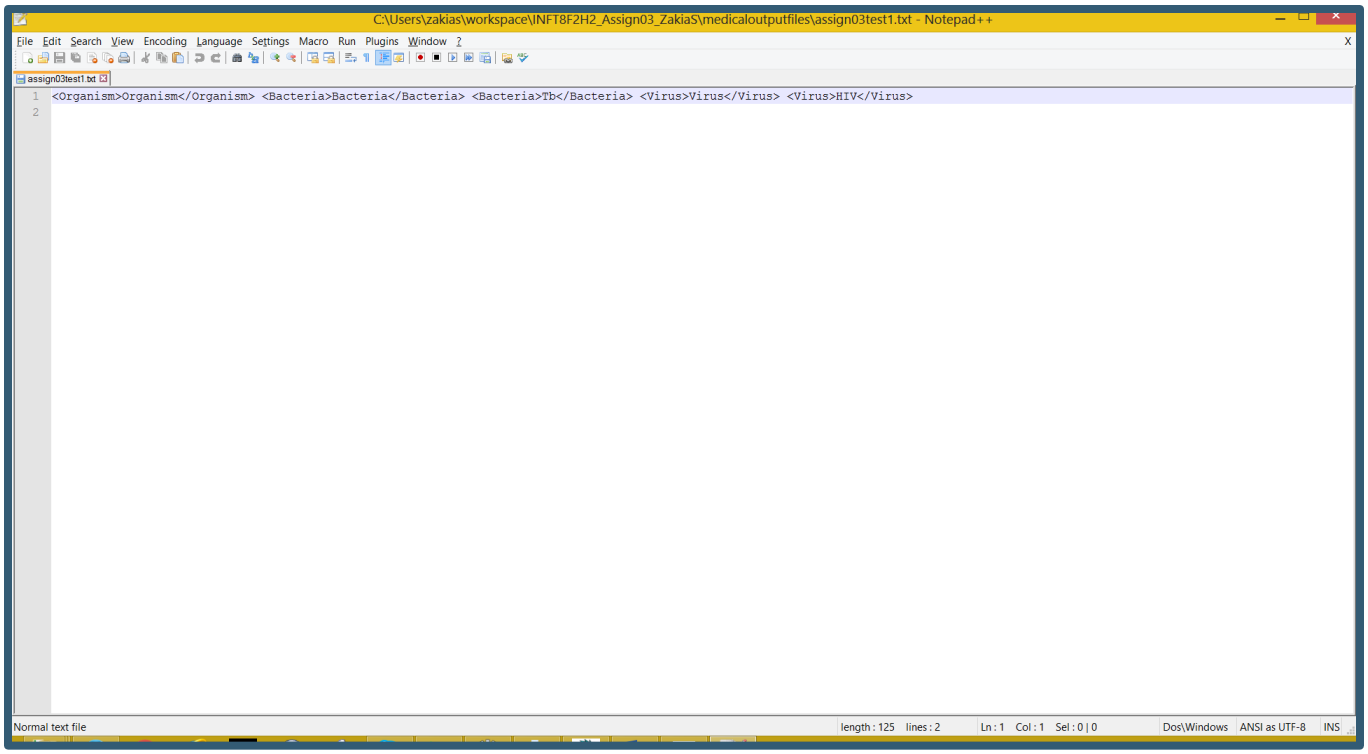
AFTER



Click 'medicaloutputfiles'



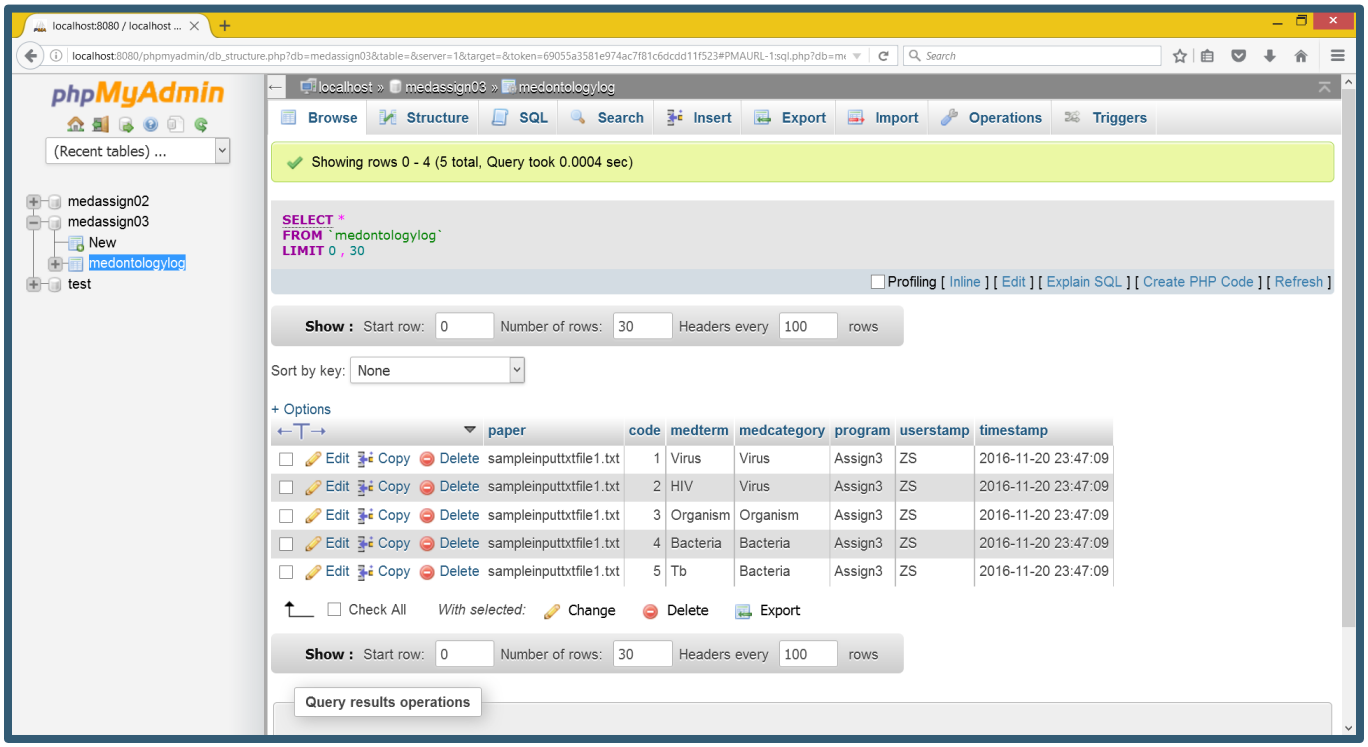
Open 'assign03test1.txt'



Full-text copy-pasted from above file :

<Organism>Organism</Organism> <Bacteria>Bacteria</Bacteria> <Bacteria>Tb</Bacteria> <Virus>Virus</Virus> <Virus>HIV</Virus>

medontologylog Database Table DOES Exist in medassign03 Database



SOURCE CODE . . .

ASSIGN03.JAVA

```
/*****************************************************************************
 *Bismillahir Rahmaanir Raheem
 *Almadadh Ya Gause Radi Allahu Ta'alah Anh - Ameen
 *Student Number : 208501583
 *Name : Zakia Salod
 *Course : INFT8F2H2
 *Assignment : 03
 *Masters of Medical Science - Medical Informatics
 *Year : 2016
 *****************************************************************************/

package INFT8F2H2_Assign03_ZakiaS;
import java.io.*;
import java.net.URI;
import javax.swing.JFileChooser;
import org.apache.commons.logging.LogFactory;
import org.apache.jena.atlas.logging.Log;
import org.apache.jena.atlas.logging.LogCtl;
import org.apache.jena.ontology.*;
import org.apache.jena.rdf.model.ModelFactory;
import org.apache.jena.util.*;
import org.apache.jena.util.iterator.ExtendedIterator;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.sql.SQLException;
import java.sql.Timestamp;
import java.nio.channels.FileChannel;
import java.nio.channels.FileLock;
import java.nio.channels.OverlappingFileLockException;
import javax.swing.JFileChooser;
import java.sql.*;
import java.util.*;

public class Assign03 {

      private static String strInput1 = "";
      private static String checkStringDouble1 = "";
      private static String checkStringDouble2 = "";
    static final String database = "medassign03";
    static File getOntologyFile;
      public static void main (String [] args) throws IllegalAccessException,
                                                      InstantiationException,
ClassNotFoundException,
                                                      FileNotFoundException,   IOException,
SQLException{

            LogCtl.setCmdLogging();

            System.out.println("------------------------------------------------------------
-----------------------------------------------------");
          System.out.println("Welcome to the Medical Semantic Search Engine Program -
2016");
```

```
          System.out.println("-----------------------------------------------------------------
----------------------------------------------------");

          System.out.println("Please select a Medical Text File >>>");
            System.out.println("Please select an Ontology File >>>");
          JFileChooser fileChooser1 = new JFileChooser();
          JFileChooser fileChooser2 = new JFileChooser();
          Scanner input1 = null;

         //>>>>>>>>>>>>>>>>>>>>>>>>  Prompt User To Choose Medical Text File & an Ontology
File >>>>>>>>>>>>>>>>>>>>>>>>
          if(fileChooser1.showOpenDialog(null) == JFileChooser.APPROVE_OPTION &&
fileChooser2.showOpenDialog(null) == JFileChooser.APPROVE_OPTION){

                    File sourceInputTxtFile01 = fileChooser1.getSelectedFile();
                    getOntologyFile = fileChooser2.getSelectedFile();


                    try {
                           input1 = new Scanner(sourceInputTxtFile01);

                        if (input1.hasNextLine()){
                               strInput1 = input1.nextLine();
                        }

                           OntModel inf =
ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM);
                           inf.read(getOntologyFile.toString(), "");
                           String URI = "http://www.organism.com/ontologies/organism.owl#";

                           ExtendedIterator classes = inf.listClasses();

                           boolean printToTargetFile = false;

                           if (!sourceInputTxtFile01.exists()){
                                        System.out.println("Source text file " +
sourceInputTxtFile01 + " does not exist");
                                        System.exit(0);
                           }//end if

                           else if (!getOntologyFile.exists()){
                                        System.out.println("Ontology file " +
getOntologyFile + " does not exist");
                                        System.exit(0);
                           }//end else if

                           //>>>>>>>>>>>>>>>>>>>>>>>>  Read-In Medical Keywords Text File &
Tag Medical Text File >>>>>>>>>>>>>>>>>>>>>>>>
                           //>>>>>>>>>>>>>>>>>>>>>>>>  Also insert into table as well too
>>>>>>>>>>>>>>>>>>>>>>>>
                           else {
                                    try{
                                             DB connectToAssign03Db = new DB(database);
                                                  createDbTable(connectToAssign03Db);
                                        int code = 0;

                                                  //iterate through the Ontology file and
Tag the Medical input file
```

```
                                                    while (classes.hasNext()) {
                                                            OntClass ontologyClass =
(OntClass) classes.next();
                                                            String ontologyClassStr =
ontologyClass.getLocalName().toString().trim();

                                                            if
(ontologyClass.hasSubClass()) {
                                                                    if
(strInput1.toLowerCase().contains(ontologyClassStr.toLowerCase())){
                                                                            //tag medical
input file

tagMedicalText(ontologyClassStr, ontologyClassStr);
                                                                            //delete then
insert latest information into database table
                                                                            code+=1;


deletemedontologylogentry(connectToAssign03Db, "medontologylog",
sourceInputTxtFile01.getName(), code, ontologyClassStr, ontologyClassStr);

insertmedontologylogentry(connectToAssign03Db, "medontologylog",
sourceInputTxtFile01.getName(), code, ontologyClassStr, ontologyClassStr);
                                                                    }//end if
                                                                      OntClass cla =
inf.getOntClass(URI + ontologyClassStr);
                                                            for (Iterator i =
cla.listSubClasses(); i.hasNext();) {
                                                                OntClass c = (OntClass)
i.next();
                                                                if
(strInput1.toLowerCase().contains(c.getLocalName().toString().toLowerCase())){

tagMedicalText(c.getLocalName().toString(), ontologyClassStr);
                                                                        //delete then insert
latest information into database table
                                                                        code+=1;


deletemedontologylogentry(connectToAssign03Db, "medontologylog",
sourceInputTxtFile01.getName(), code, c.getLocalName().toString(), ontologyClassStr);

insertmedontologylogentry(connectToAssign03Db, "medontologylog",
sourceInputTxtFile01.getName(), code, c.getLocalName().toString(), ontologyClassStr);
                                                                }//end if
                                                             }//end for
                                                            }//end if
                                                            else
if(!ontologyClass.hasSubClass() && !ontologyClass.hasSuperClass()){
                                                                    if
(strInput1.toLowerCase().contains(ontologyClassStr.toLowerCase())){

tagMedicalText(ontologyClassStr, ontologyClassStr);
                                                                            //delete then insert
latest information into database table
                                                                            code+=1;
```

```
deletemedontologylogentry(connectToAssign03Db, "medontologylog",
sourceInputTxtFile01.getName(), code, ontologyClassStr, ontologyClassStr);

insertmedontologylogentry(connectToAssign03Db, "medontologylog",
sourceInputTxtFile01.getName(), code, ontologyClassStr, ontologyClassStr);
                                                                    }//end if
                                                                }
                                                            }//end while

                                                                printToTargetFile = true;
                                                    }//end try
                                                    catch(Exception  e){
                                                          e.printStackTrace();
                                                     }//end catch

                                        }//end else

                                        //>>>>>>>>>>>>>>>>>>>>>>>  Create Medical Output Directory If
Does Not Exist >>>>>>>>>>>>>>>>>>>>>>>>
                                        File medicalOutputDir = new File("medicaloutputfiles");
                                        if (!medicalOutputDir.exists()){
                                              System.out.println("\nCreating directory: "+ '"' +
"medicaloutputfiles" + '"');
                                              medicalOutputDir.mkdir();
                                              System.out.println("Successfully created " + '"' +
"medicaloutputfiles" + '"' + " directory");
                                        }
                                        else{
                                              System.out.println("Output file/s will be written to
directory: "+ '"' + "medicaloutputfiles" + '"');
                                        }

                                        //>>>>>>>>>>>>>>>>>>>>>>>  Write to File >>>>>>>>>>>>>>>>>>>>>>>
                                        if(printToTargetFile == true && medicalOutputDir.exists() &&
medicalOutputDir.isDirectory()){
                                              String outputFileName = "";
                                              System.out.println("\nPlease enter output file name >>>
");
                                          Scanner getOutputFileName = new Scanner(System.in);
                                          outputFileName = getOutputFileName.nextLine();
                                             File targetOutputTxtFile;
                                          boolean uniqueOutputFileName = true;

                                             targetOutputTxtFile = new
File(medicalOutputDir+"/"+outputFileName.trim()+".txt");

                                           while (targetOutputTxtFile.exists()){
                                                uniqueOutputFileName = false;
                                                System.out.println("Target text file: " +
targetOutputTxtFile + " already exists");
                                                      System.out.println("Please enter a different
output file name >>> ");
                                                outputFileName = getOutputFileName.nextLine();
                                                  targetOutputTxtFile = new
File(medicalOutputDir+"/"+outputFileName.trim()+".txt");

                                                if(!targetOutputTxtFile.exists()){
```

```
                                        uniqueOutputFileName = true;
                                    }//end if
                                }//end while

                            if (uniqueOutputFileName == true) {
                                PrintWriter output = null;

                                try{
                                        output = new PrintWriter(targetOutputTxtFile);
                                        output.println(strInput1);

                                        System.out.println("\nSuccessfully wrote to file: "
+ targetOutputTxtFile);
                                        System.out.println("Full path to written file is: "
+ targetOutputTxtFile);
                                        System.out.println();
                                }//end try
                                        catch(FileNotFoundException e){
                                                e.printStackTrace();
                                        }//end catch
                                finally{
                                            //close the file
                                                    if (output !=null)
                                                            output.close();
                                }//end finally
                                }//end if
                            }//end if
                        }//end try
                        catch(FileNotFoundException e){
                                e.printStackTrace();
                        }//end catch1
                catch (Exception e) {
                    System.out.println(e.getMessage());
                }//end catch2
                        finally{
                                //close the medical input file
                                if (input1 !=null)
                                   input1.close();
                        }//end finally
                }//end if

                    //handles searching of medical term/s entered by the user, and searches and
displays files in the database with this medical term/s
                    userSearchTerm();

                    System.out.println("Exitting program . . . . . . . ");

                    System.out.println();
                    System.out.println("-----------------------------------------------------------------
-----------------------------------------------------");
                       System.out.println("End of Medical Semantic Search Engine Program - 2016");
                       System.out.println("-----------------------------------------------------------
-----------------------------------------------------");

            }//end main()

                /**pass-in the same variable (superclass string) for both parameters - for Superclass
Ontology term tagging
```

```
	 * pass-in Subclass Ontology term for "ontologyTerm" and Superclass Ontology term for
"ontologyTagTerm" for Subclass Ontology term tagging
	 */
	private static void tagMedicalText(String ontologyTerm, String ontologyTagTerm){
			strInput1 = strInput1.replace(ontologyTerm, "<"+ontologyTagTerm+">" +
ontologyTerm + "</"+ontologyTagTerm+">");
	}

	/**creates database table medontologylog**/
	private static void createDbTable(DB connectToAssign03Db){

		try{
		    String createMedOntologyLog = "CREATE TABLE medontologylog(paper
varchar(250) NOT NULL, "+
                                                "code int(11) NOT NULL, medterm
varchar(255) NOT NULL, medcategory varchar(255) NOT NULL, "+
                                                 "program varchar(15) NOT NULL,
userstamp varchar(15) NOT NULL, timestamp datetime NOT NULL, "+
                                                "PRIMARY KEY(paper, code, medterm,
medcategory))";

		    connectToAssign03Db.updateTbl(createMedOntologyLog);
		}
		catch(SQLException e){
			System.out.println("Table already exists in the " + database + "
database");
		}
	}//end createDbTable()


	/**deletes records from the medontologylog database table, according to the specified
Primary Key values**/
	private static void deletemedontologylogentry(DB connectToAssign03Db, String
tableName, String paper, int code, String medterm, String medcategory){

		try{
			connectToAssign03Db.updateTbl("DELETE FROM " + tableName +" WHERE
paper='" + paper +"' AND code=" + code + " AND medterm='" + medterm + "' AND medcategory
='" + medcategory+"'");
		}
		catch(SQLException e){
			e.printStackTrace();
		}
	}//end deletemedontologylogentry()


	/**inserts records onto the medontologylog database table**/
	private static void insertmedontologylogentry(DB connectToAssign03Db, String
tableName, String paper, int code, String medterm, String medcategory){

		try{
			String program   = "Assign3";
			String userstamp = "ZS";
			java.util.Date dt = new java.util.Date();
			java.text.SimpleDateFormat sdf = new java.text.SimpleDateFormat("yyyy-
MM-dd HH:mm:ss");
			String currentTime = sdf.format(dt);
```

```
                  connectToAssign03Db.updateTbl("INSERT INTO " + tableName +"(paper, code,
medterm, medcategory, program, userstamp, timestamp) "+
                                                "VALUES('"+paper+"'," + code + ",'" +
medterm + "','" + medcategory+"','" + program +"','" + userstamp + "','" + currentTime +
"'"  +")");
            }
            catch(SQLException e){
                  e.printStackTrace();
            }
      }//end insertmedontologylogentry()


      /**prompts the user on whether or not he/she would like to search the database for a
medical term
       * to retrieve file information for this term/s from the database
       */
      private static void userSearchTerm(){
            String promptUser = "";
            Scanner getpromptUser;
            System.out.print("\nWould you like to search for a medical term from the
database? >>> ");
          getpromptUser = new Scanner(System.in);
          promptUser = getpromptUser.nextLine();

          System.out.println();
          //user wants to proceed to search the database
          while (promptUser.equalsIgnoreCase("Y") || promptUser.equalsIgnoreCase("Yes") ){
            String searchTerm ="";
            System.out.println("Please enter a medical search term  >>> ");
                Scanner getUserMedicalSearchTerm = new Scanner(System.in);
                searchTerm = getUserMedicalSearchTerm.nextLine();
                  System.out.println("The database currently contains the following
medical file information for medical term : " + searchTerm + " >>>");
                searchUserTermAndOntology(searchTerm);

                  System.out.print("\nWould you like to continue searching for medical
terms from the database? >>> ");
                getpromptUser = new Scanner(System.in);
                promptUser = getpromptUser.nextLine();
          }
      }//end userSearchTerm()


      /**search the Ontology, by first looking at the user's search term - (searchTerm
passed-in as parameter)**/
      private static void searchUserTermAndOntology(String searchTerm){
            OntModel inf = ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM);

            inf.read(getOntologyFile.toString(), "");
            String URI = "http://www.organism.com/ontologies/organism.owl#";

            ExtendedIterator classes = inf.listClasses();
            boolean foundUserMedicalTermInOnt = false;

            //iterate through the Ontology file and look for the Medical term entered by
the user
            while (classes.hasNext()) {
                  foundUserMedicalTermInOnt = true;
```

```
                    OntClass ontologyClass = (OntClass) classes.next();
                    String ontologyClassStr =
ontologyClass.getLocalName().toString().trim();
                    if (ontologyClass.hasSubClass()) {
                                    if
(searchTerm.toLowerCase().contains(ontologyClassStr.toLowerCase())){
                                        //search database using medterm = ontologyClassStr
and medcategory = ontologyClassStr
                                        try{
                                           searchDBAndDisplay(ontologyClassStr,
ontologyClassStr);
                                        }//end try
                                            catch(SQLException sqlException){
                                                    System.out.println("Error");
                                                    sqlException.printStackTrace();
                                            }//end catch
                                    }//end if
                                   OntClass cla = inf.getOntClass(URI + ontologyClassStr);
                for (Iterator i = cla.listSubClasses(); i.hasNext();) {
                    OntClass c = (OntClass) i.next();
                                   if
(searchTerm.toLowerCase().contains(c.getLocalName().toString().toLowerCase())){
                                        //search database using medterm =
c.getLocalName().toString() and medcategory = ontologyClassStr
                                        try{

searchDBAndDisplay(c.getLocalName().toString(), ontologyClassStr);
                                        }//end try
                                        catch(SQLException sqlException){
                                                System.out.println("Error");
                                                sqlException.printStackTrace();
                                        }//end catch
                                    }//end if
                }//end for
                    }//end if
                    else if(!ontologyClass.hasSubClass() &&
!ontologyClass.hasSuperClass()){
                                if
(searchTerm.toLowerCase().contains(ontologyClassStr.toLowerCase())){
                                  //search database using medterm = ontologyClassStr and
medcategory = ontologyClassStr
                                   try{
                                     searchDBAndDisplay(ontologyClassStr, ontologyClassStr);
                                   }//end try
                                       catch(SQLException sqlException){
                                              System.out.println("Error");
                                              sqlException.printStackTrace();
                                       }//end catch
                                }//end if
             }
            }//end while

             //Medical Term not found in Ontology
             if(!foundUserMedicalTermInOnt){
                   System.out.println("Sorry, there are currently no medical files on the
system with medical term : " + searchTerm);
             }
       }//end searchUserTermAndOntology
```

```
	/**searches the database and displays the output from the database for the specified medical term, after consulting the Ontology**/
	private static void searchDBAndDisplay(String medterm, String medcategory) throws SQLException{
		try{
			DB connectToAssign03Db = new DB(database);
				ResultSet resultSet = connectToAssign03Db.queryTbl("SELECT * FROM medontologylog WHERE medterm = '" + medterm + "'" +
												" AND medcategory = '" + medcategory + "'"  + "ORDER BY timestamp DESC");
				ResultSetMetaData metaData = resultSet.getMetaData();
				int numberOfColumns = metaData.getColumnCount();
				System.out.println();

				boolean foundUserMedicalTerm = false;
				while(resultSet.next()){
					foundUserMedicalTerm = true;
					System.out.println("------------------------------------------------------------------------------------");
					System.out.println("Paper : " + resultSet.getString("paper"));
					System.out.println("Code : " + resultSet.getString("code"));
					System.out.println("Medical Term : " + resultSet.getString("medterm"));
					System.out.println("Medical Category : " + resultSet.getString("medcategory"));
					System.out.println("Program : " + resultSet.getString("program"));
					System.out.println("Userstamp : " + resultSet.getString("userstamp"));
					System.out.println("Timestamp : " + resultSet.getString("timestamp"));
					System.out.println("------------------------------------------------------------------------------------");
					System.out.println();
				}

				//Medical Term not found in Database
				 if(!foundUserMedicalTerm){
					System.out.println("Sorry, there are currently no medical files on the database with medical term : " + medterm);
				}

				System.out.println();
		}
		catch(SQLException sqlException){
			System.out.println("Error");
			sqlException.printStackTrace();
		}
	}//end searchDBAndDisplay
}//end class Assign03
```

DB.JAVA

```
/**************************************************************************
 *Bismillahir Rahmaanir Raheem
 *Almadadh Ya Gause Radi Allahu Ta'alah Anh - Ameen
 *Student Number : 208501583
 *Name : Zakia Salod
 *Course : INFT8F2H2
 *Assignment : 03
 *Masters of Medical Science - Medical Informatics
 *Year : 2016
 **************************************************************************/

package INFT8F2H2_Assign03_ZakiaS;

import java.io.*;
import java.sql.*;

public class DB {
	Connection conn;

	DB(String database){
		final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
		final String DATABASE_URL = "jdbc:mysql://localhost:3307/"+database;

		//connect to mysql driver
		try{
			Class.forName(JDBC_DRIVER);
			System.out.println("{System information} : Driver Successfully
Loaded");
		}//end try
		catch(ClassNotFoundException e){
			System.out.println("{System information} : Unable to connect");
			System.exit(1);
		}//end catch


		try{
			conn = DriverManager.getConnection(DATABASE_URL, "root", "usbw");
			System.out.println("{System information} : Connection to " + database +
" successfully established" );
		}//end try
		catch(Exception e){
			System.out.println(e.getMessage());
		}//end catch
	}//end DB() Constructor



	//Method executes SQL queries, input as string argument
	ResultSet queryTbl(String sqlStmt) throws SQLException{
		Statement stmt = conn.createStatement();
		ResultSet rs = stmt.executeQuery(sqlStmt); // select * from a table
		System.out.println("{System information} : Successfully executed query on
table.");
		return rs;
	}
```

```
	void updateTbl(String update) throws SQLException{
		Statement stmt = conn.createStatement();
		stmt.executeUpdate(update);
		System.out.println("{System information} : Successfully updated table.");
		stmt.close();
	}



	void closeDB() throws SQLException{
		conn.close();
		System.out.println("{System information} : Successfully closed table.");
	}
}//end DB class
```

ABOUT THE PROGRAMMER . . .

Zakia Salod was born on December 24th 1989 in Durban. She is currently studying full-time towards her Masters in Medical Science Medical Informatics degree at the faculty of Health Sciences at UKZN. She is also working full-time as a Software Developer at a software company, 2Cana Solutions in La Lucia Ridge, Durban – on the Momentum medical aid system.

She graduated with a BSc in Computer Science and IS&T at UKZN in 2010. She had also graduated with a BCom IT Honours (Cum Laude) degree at UKZN in 2011, with first position in her degree from both Westville and Pietermaritzburg campuses.

END OF DOCUMENT