<Medical Distributed Search Engine Program – 2016>

Medical Distributed Search Engine Program – 2016

</Medical Distributed Search Engine Program – 2016>

A Console Application

INFT8F2H2 - PROGRAMMING MEDICAL INFORMATICS SYSTEMS

ASSIGNMENT 04 : SOFTWARE : MEDICAL DISTRIBUTED 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 04 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 04 : SOFTWARE : MEDICAL DISTRIBUTED SEARCH ENGINE PROGRAM - 2016

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

WHAT IS IT?

The Medical Distributed Search Engine Program - 2016 is a software used to search the database for files on the system with the

medical term entered by the user. 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 (this is done by the Medical Query Agent), and then searches the database for all

relevant files (this is done by the Medical Search Agent). 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).

APPROACH :

Software Agents are used as query agents ("Medical Query Agent") and search agents ("Medical Search Agent") to perform the relevant

operations, using the Java Agent Development Environment (JADE). The query agent prompts the user for a query. The query agent then

consults the Ontology to discover what term (semantically) needs to be sent to the search agent to query. This is then directed

to the Yellow Pages Service of JADE, by the use of an agent called the Directory Facilitator (DF). The DF then directs the query to

the search agent. The search engine then receives the query from the query agent and searches the database. All results that satisfies

the search from the database, is output to the console.

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 (.OWL) Ontology file can be found here :

- medicalinputfiles\medicalterm.owl

ABSTRACT

This document serves as documentation / an overview of the Software developed as part of Assignment 04 of this module. It includes an "About" of this program, the Technologies Used, Sample Run of Program, Source Code, & About the Programmer.

OUTPUT :

FROM USER SEARCH TERM ENTERED :-

CONSOLE :

o File (paper) information from the database, that satisfies the search term/s entered by the user (which is first directed to the Ontology file to check for, semantically, which term/s need to be sent to the search agent to search its database for).

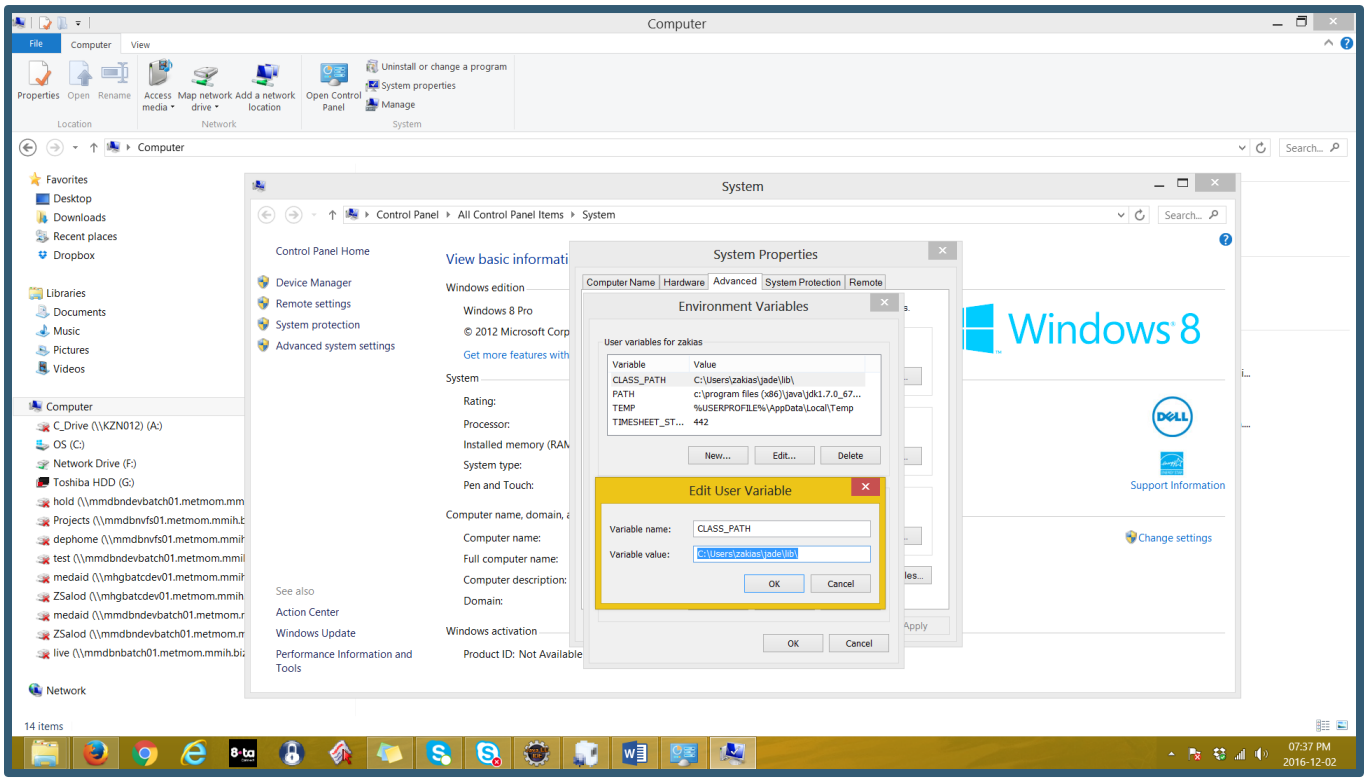
CONTACT

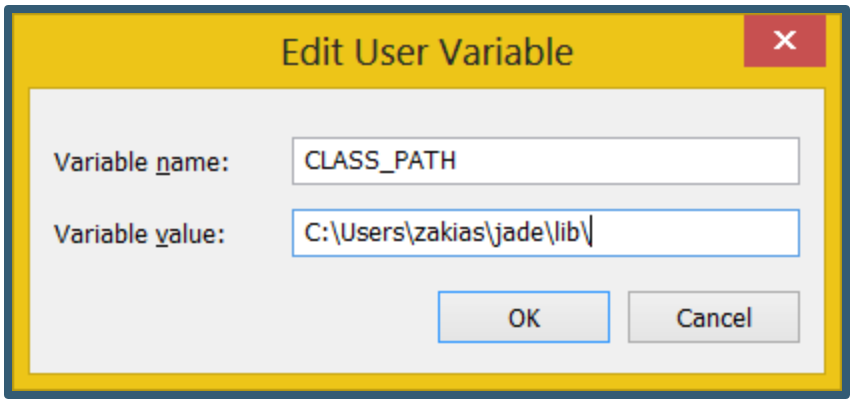
o If you would like more info about the Medical Distributed Search Engine Program, or require free support for this program, please contact the programmer at http://www.zakiasalod.weebly.com or zakia.salod@gmail.com

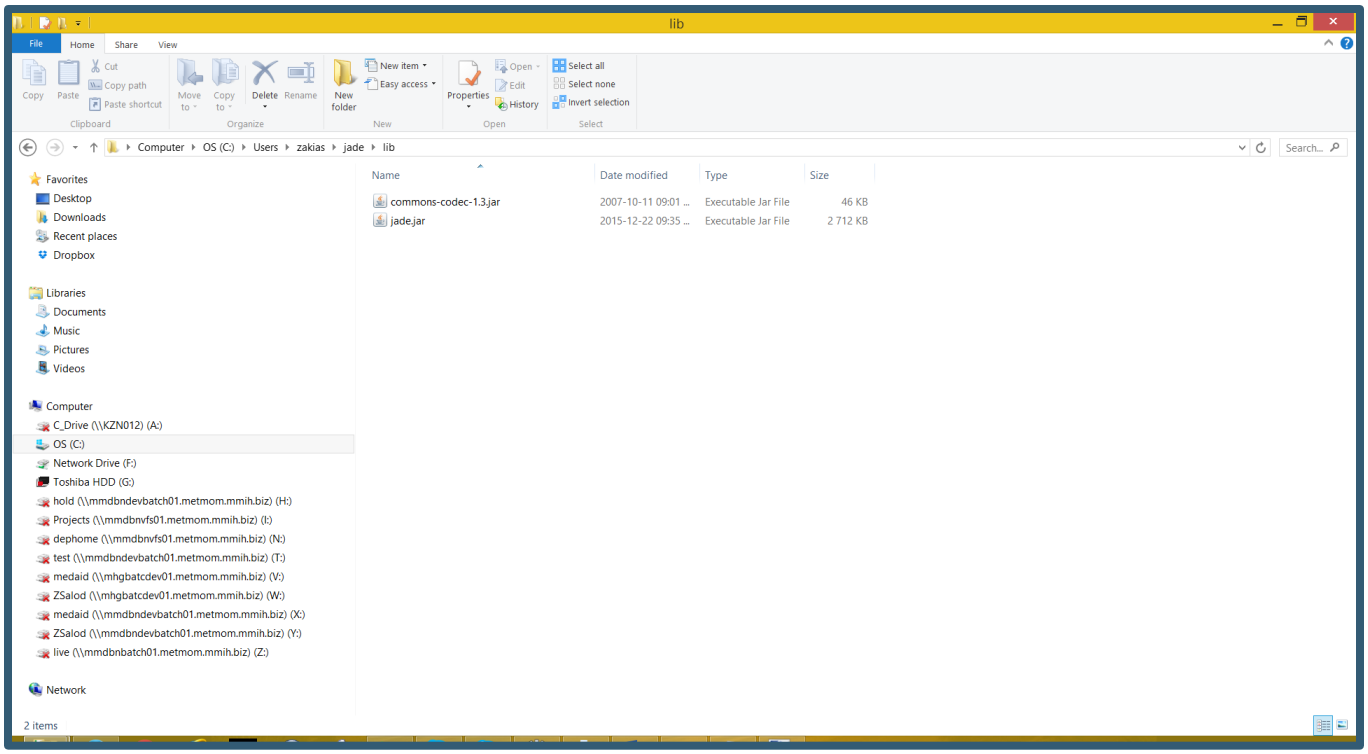
---

TECHNOLOGIES USED . . .

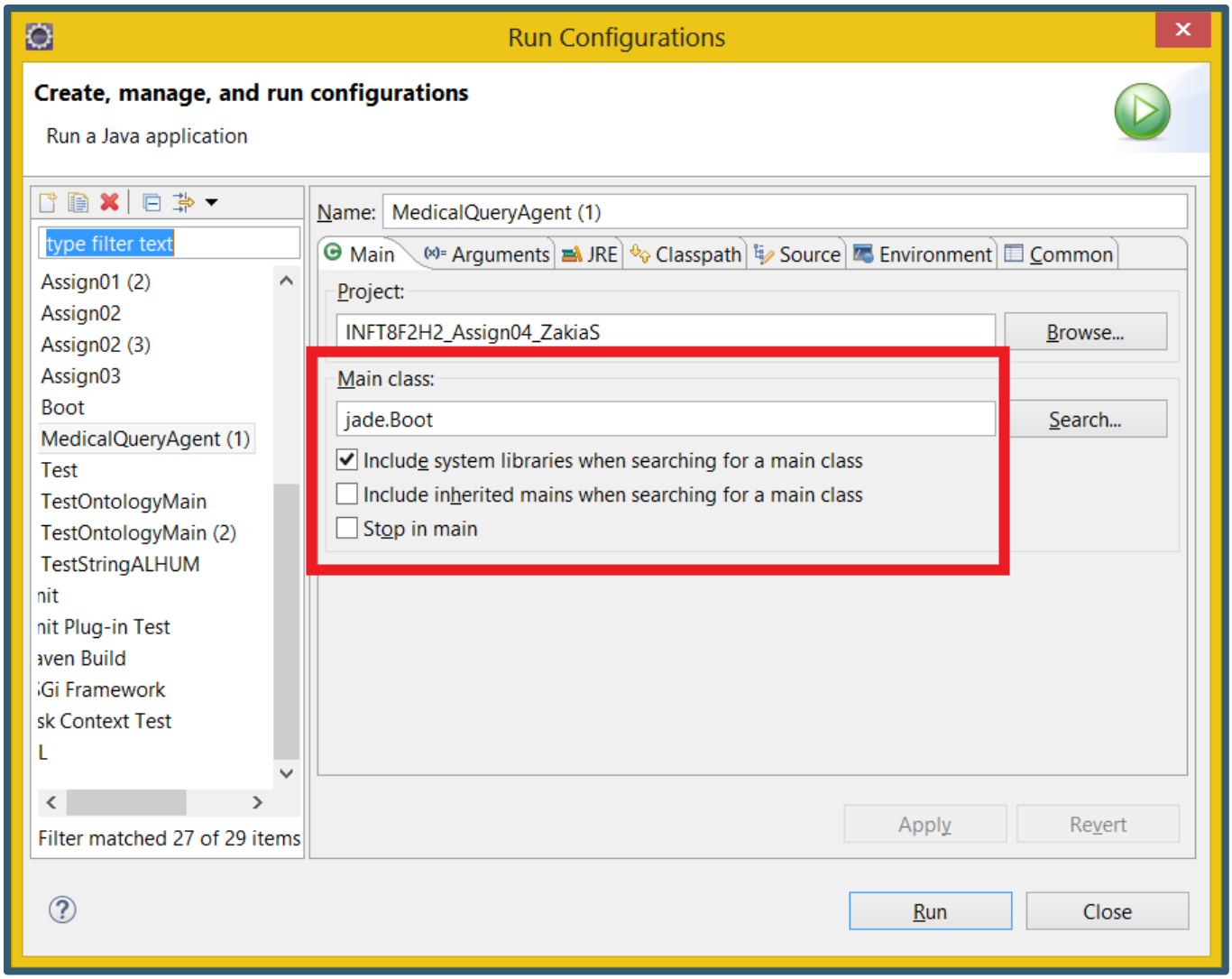
- ✓ Programming Language : Java
- ✓ Integrated Development Environment (IDE) : Eclipse LUNA
- ✓ JRE : 1.8.0_45
- ✓ JDK : 1.8.0_45
- ✓ Webservice : USBWebserver v8.6
- ✓ Database : MySQL
- ✓ Mysql-connector : mysql-connector-java-5.1.40-bin.jar
- ✓ Protégé 5.1.0
- ✓ JENA API Library
- ✓ JADE 4.4.0 → jade.jar file
- ✓ JADE 4.4.0 → commons-codec-1.3.jar
- ✓ JADE 4.4.0 → The Yellow Pages Service → The DFService Class Facility
- ✓ System (client-level) environment variable *(Set-Ups)*:

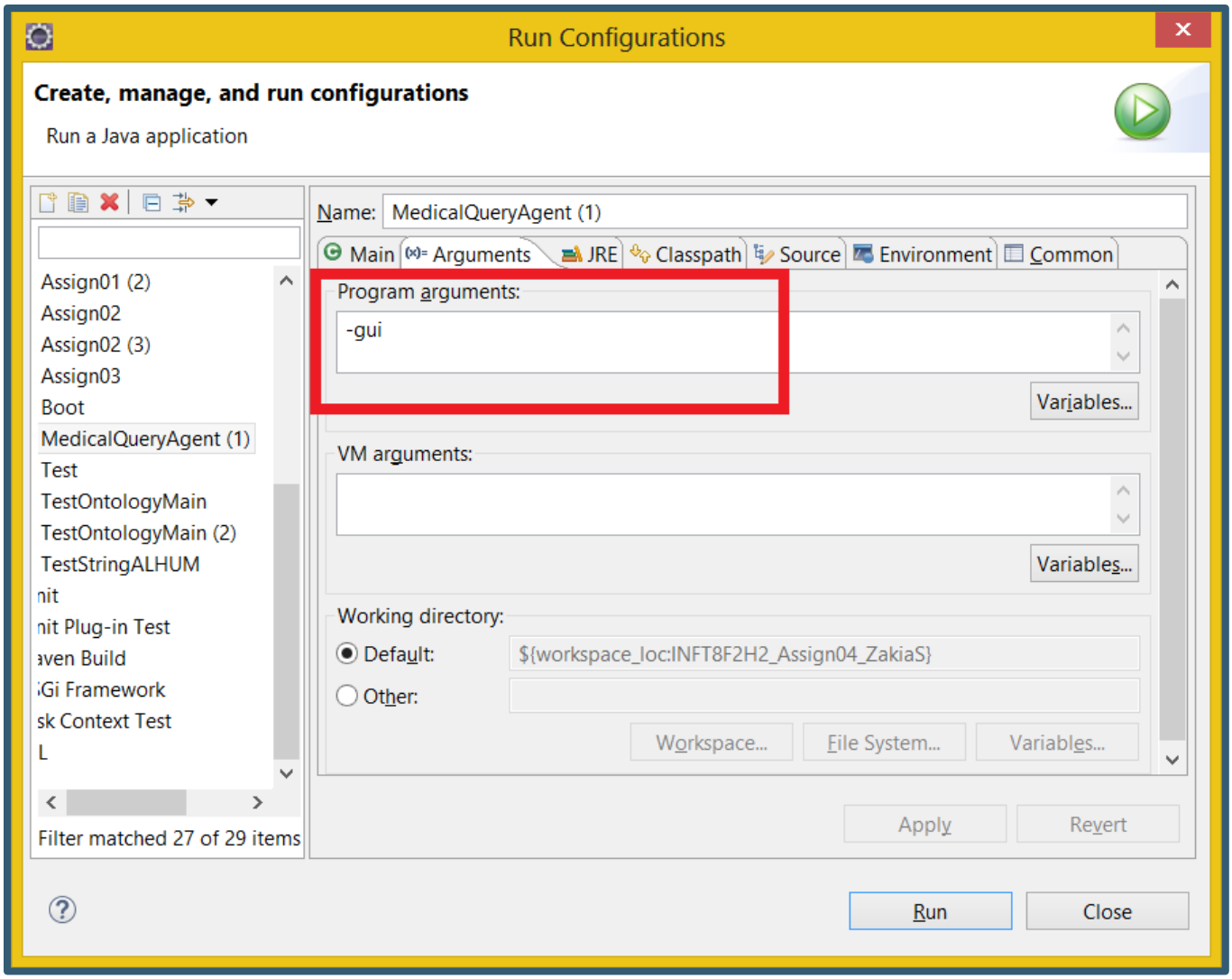






✓ Eclipse Set-Ups *(Project → Run → Run Configurations)*


Run Configurations
Create, manage, and run configurations
Run a Java application
type filter text
Assign01 (2)
Assign02
Assign02 (3)
Assign03
Boot
MedicalQueryAgent (1)
Test
TestOntologyMain
TestOntologyMain (2)
TestStringALHUM
nit
nit Plug-in Test
aven Build
iGi Framework
sk Context Test
L
Filter matched 27 of 29 items
Name: MedicalQueryAgent (1)
Main
Arguments
JRE
Classpath
Source
Environment
Common
Project:
INFT8F2H2_Assign04_ZakiaS
Browse...
Main class:
jade.Boot
Search...
Include system libraries when searching for a main class
Include inherited mains when searching for a main class
Stop in main
Apply
Revert
Run
Close

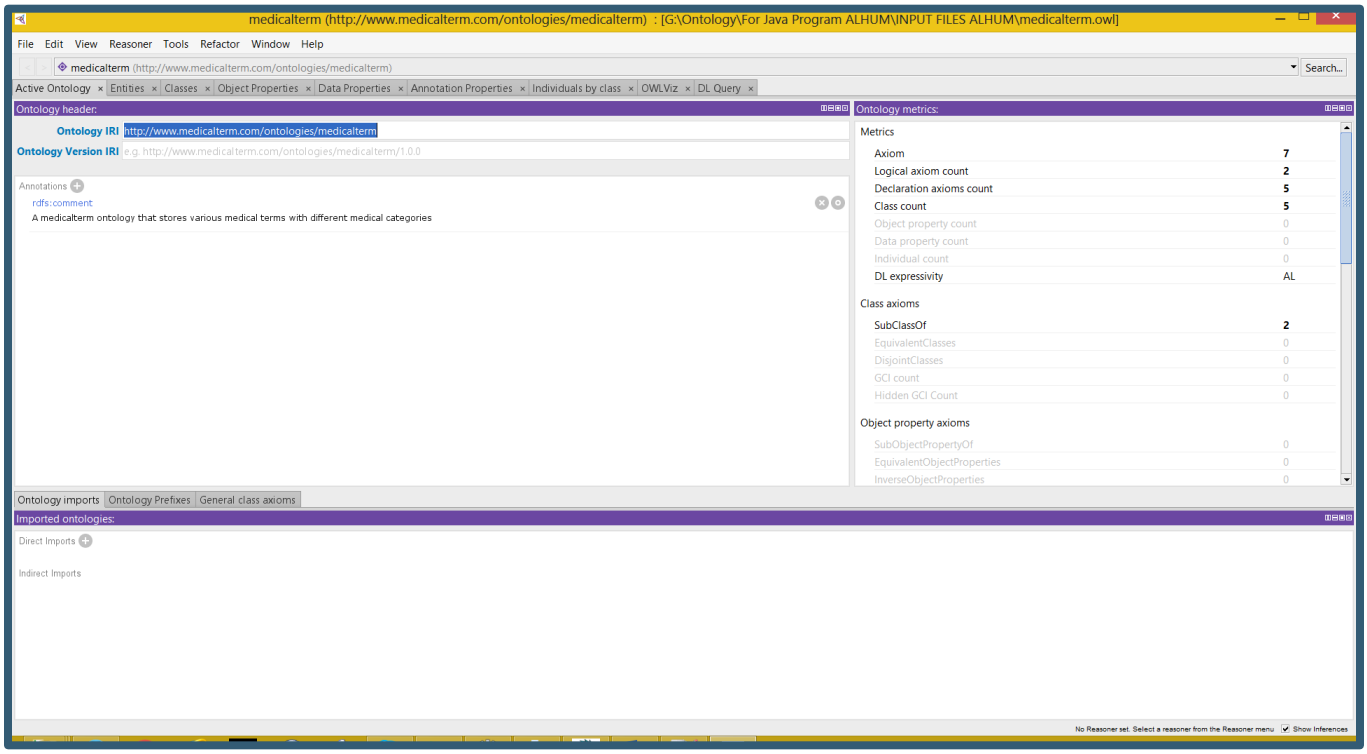

Run Configurations
Create, manage, and run configurations
Run a Java application
Assign01 (2)
Assign02
Assign02 (3)
Assign03
Boot
MedicalQueryAgent (1)
Test
TestOntologyMain
TestOntologyMain (2)
TestStringALHUM
Filter matched 27 of 29 items
Name: MedicalQueryAgent (1)
Main
Arguments
JRE
Classpath
Source
Environment
Common
Program arguments:
-gui
Variables...
VM arguments:
Variables...
Working directory:
Default:
${workspace_loc:INFT8F2H2_Assign04_ZakiaS}
Other:
Workspace...
File System...
Variables...
Apply
Revert
Run
Close

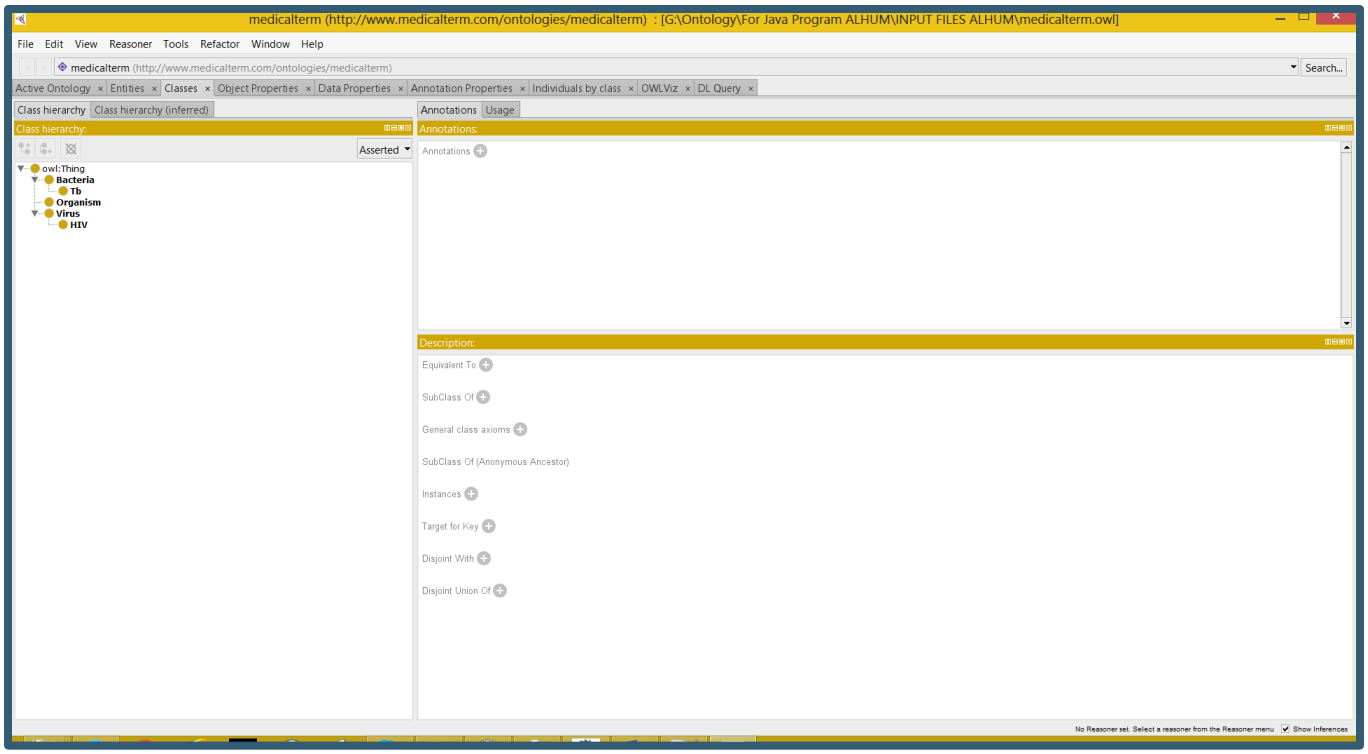
SAMPLE RUN OF PROGRAM . . .

INPUT

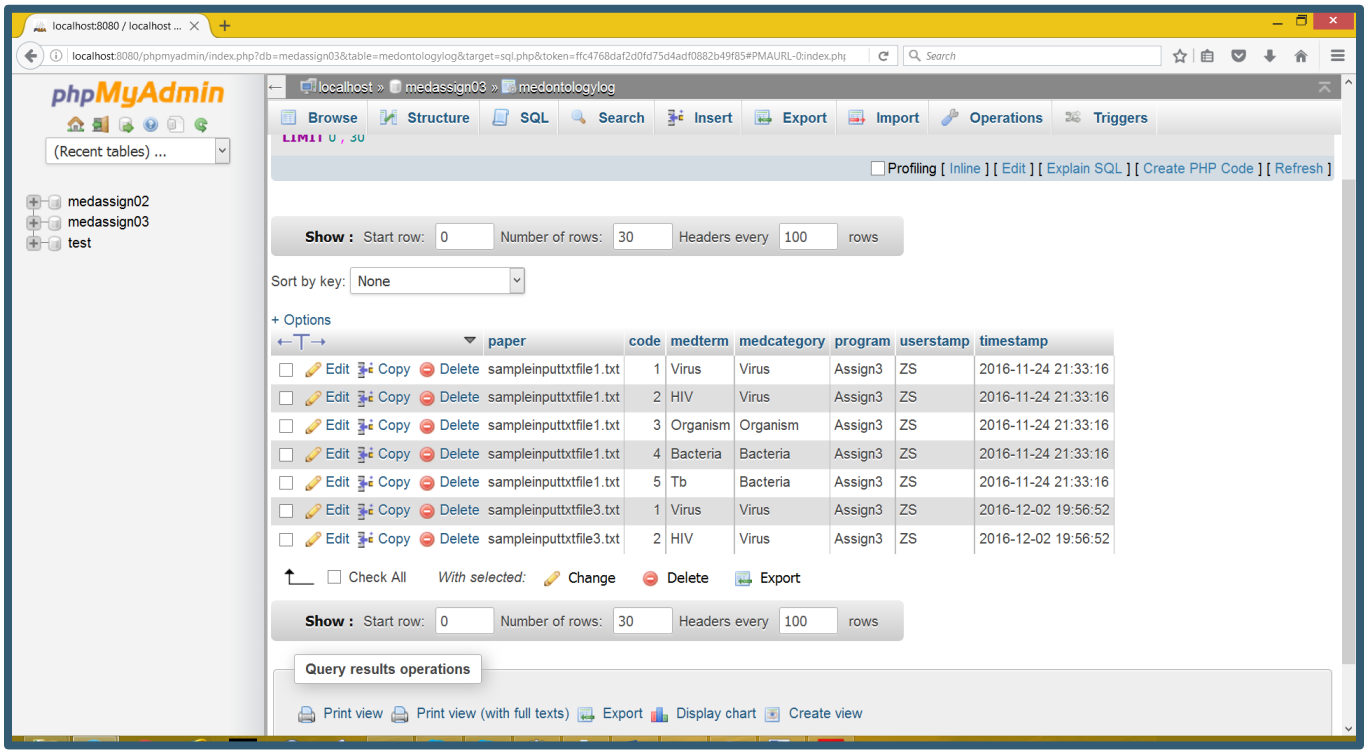
MEDICALINPUTFILES\MEDICALTERM.OWL

{Opened in Protégé}





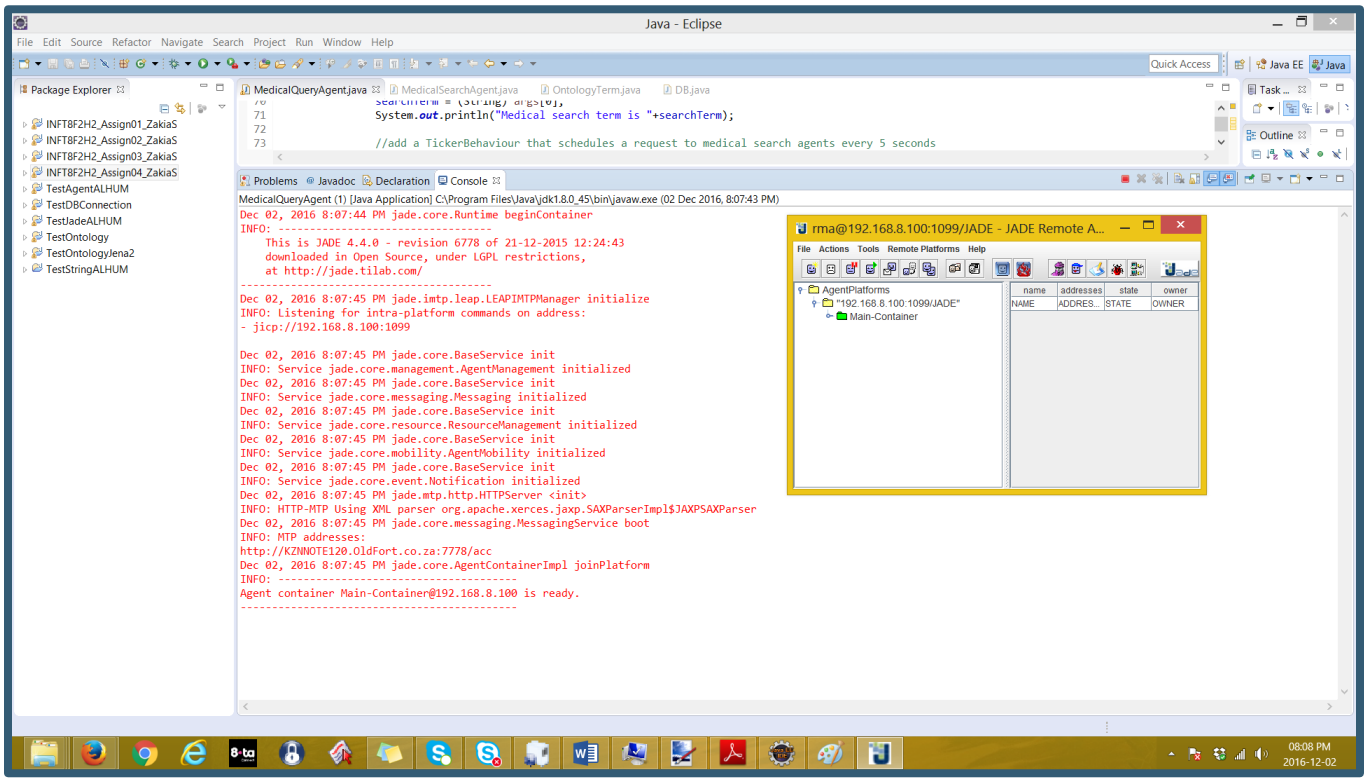
TEST DATA (FROM DATABASE)

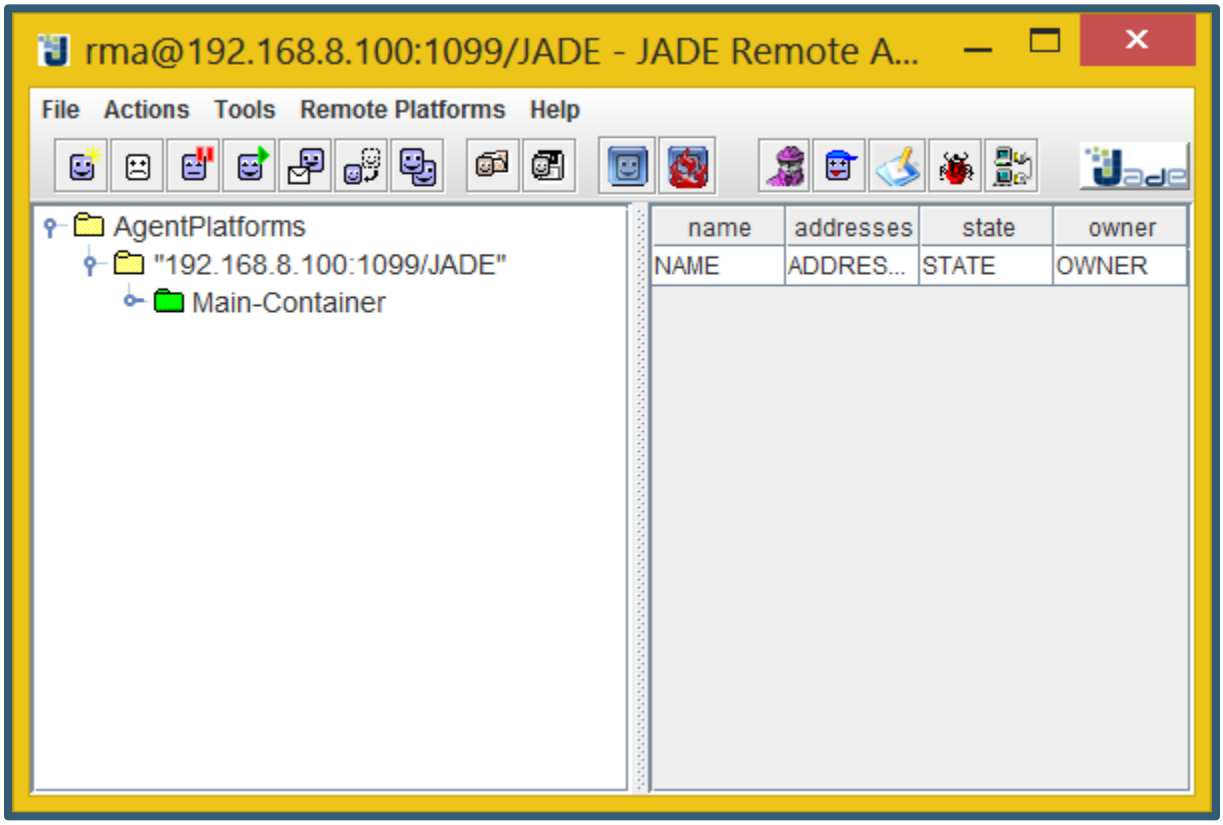


paper	code	medterm	medcategory	program	userstamp	timestamp
sampleinputtxtfile1.txt	1	Virus	Virus	Assign3	ZS	2016-11-24 21:33:16
sampleinputtxtfile1.txt	2	HIV	Virus	Assign3	ZS	2016-11-24 21:33:16
sampleinputtxtfile1.txt	3	Organism	Organism	Assign3	ZS	2016-11-24 21:33:16
sampleinputtxtfile1.txt	4	Bacteria	Bacteria	Assign3	ZS	2016-11-24 21:33:16
sampleinputtxtfile1.txt	5	Tb	Bacteria	Assign3	ZS	2016-11-24 21:33:16
sampleinputtxtfile3.txt	1	Virus	Virus	Assign3	ZS	2016-12-02 19:56:52
sampleinputtxtfile3.txt	2	HIV	Virus	Assign3	ZS	2016-12-02 19:56:52

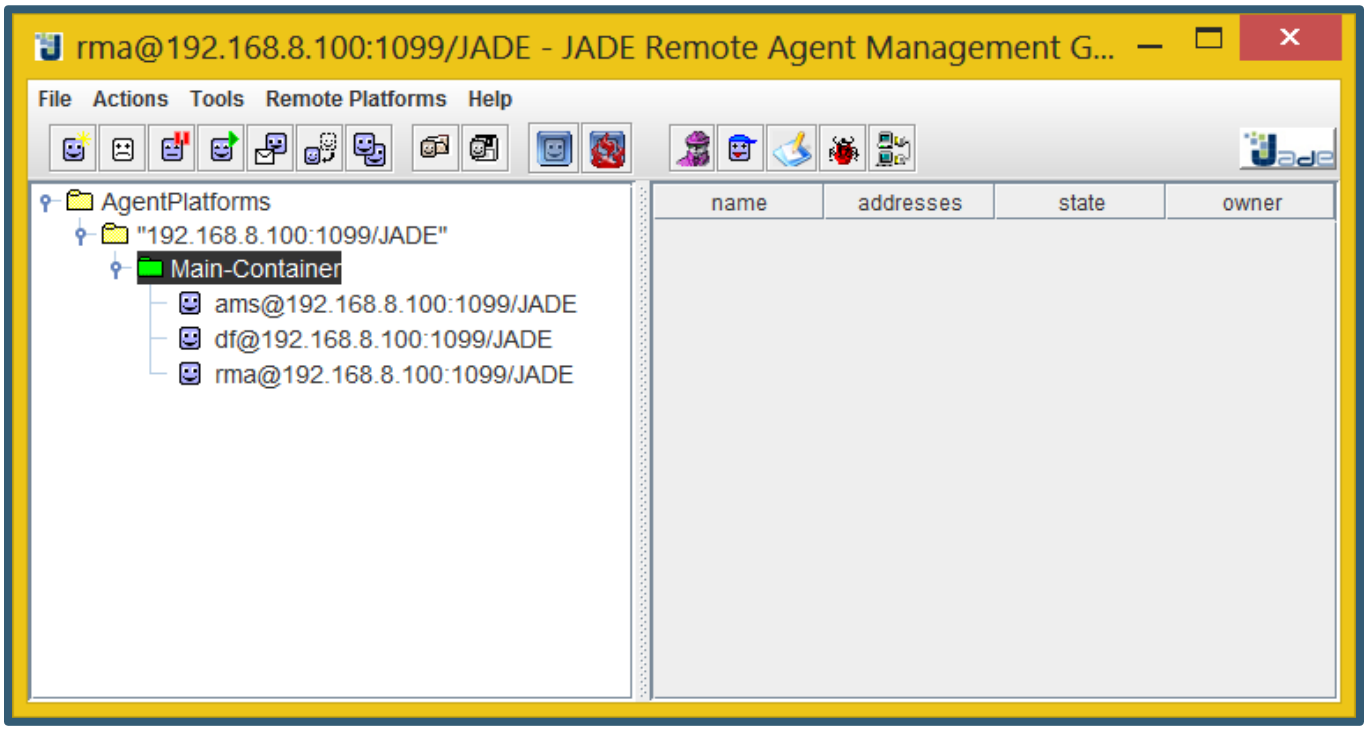
JOURNAL

Run → Run Configurations

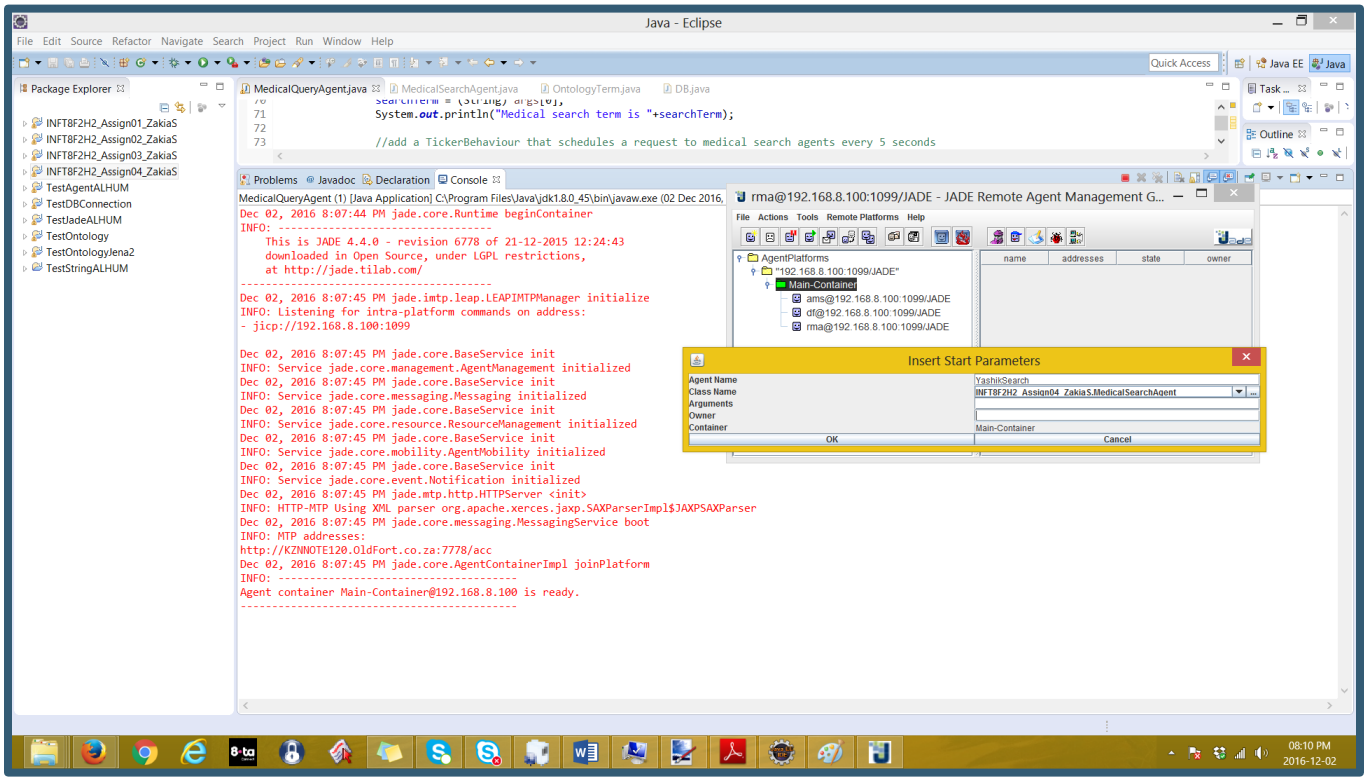


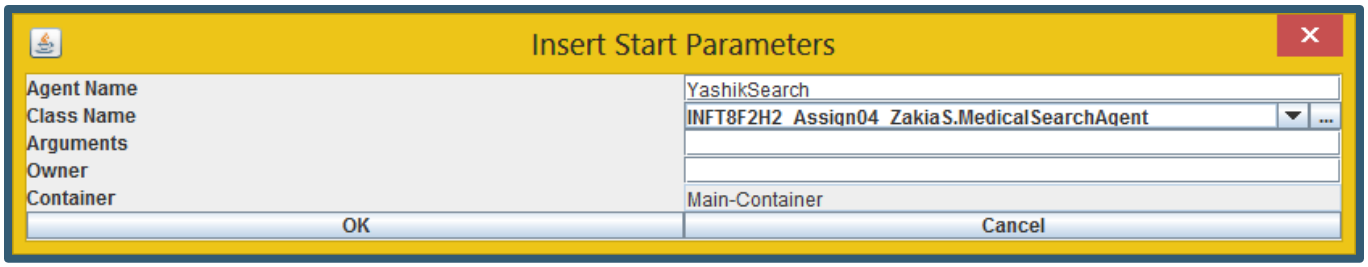


Creating Medical Search Agent Instance →

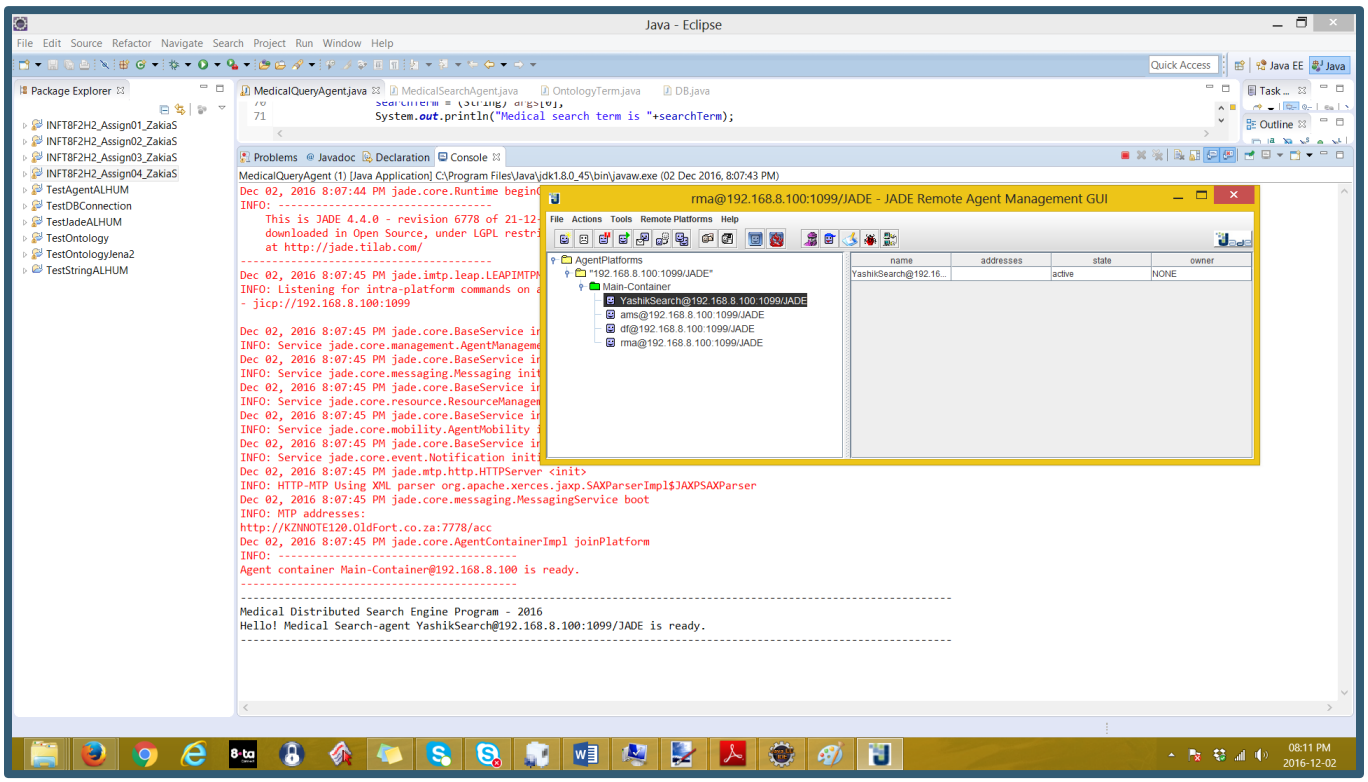


Right-Click 'Main-Container' → Start New Agent → Full-In Details of Agent

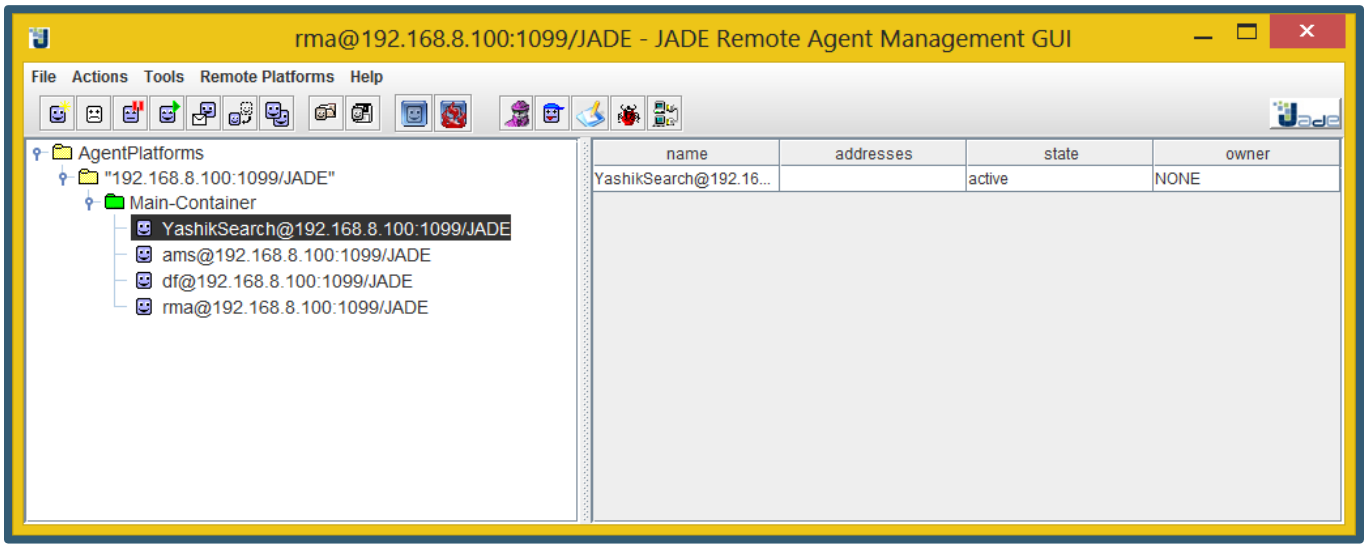



Insert Start Parameters
Agent Name
YashikSearch
Class Name
INFT8F2H2_Assign04_ZakiaS.MedicalSearchAgent
Arguments
Owner
Container
Main-Container
OK
Cancel

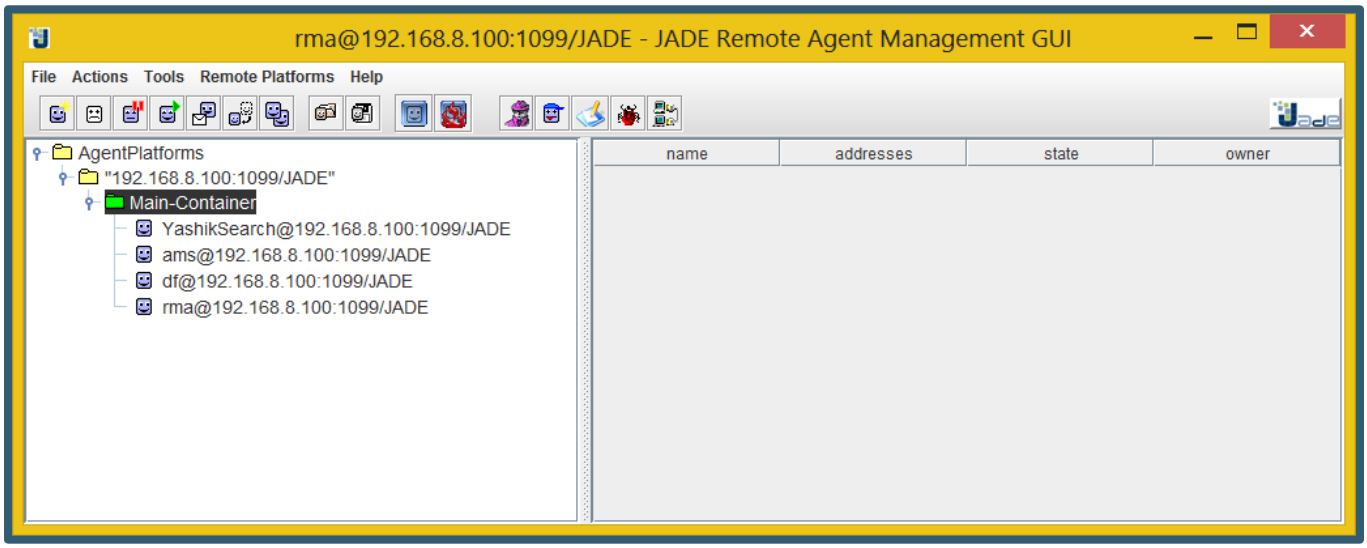
Click OK


Java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
INFT8F2H2_Assign01_ZakiaS
INFT8F2H2_Assign02_ZakiaS
INFT8F2H2_Assign03_ZakiaS
INFT8F2H2_Assign04_ZakiaS
TestAgentALHUM
TestDBConnection
TestJadeALHUM
TestOntology
TestOntologyJena2
TestStringALHUM
MedicalQueryAgent.java
MedicalSearchAgent.java
OntologyTerm.java
DB.java
System.out.println("Medical search term is "+searchTerm);
Problems
Javadoc
Declaration
Console
MedicalQueryAgent (1) [Java Application] C:\Program Files\Java\jdk1.8.0_45\bin\javaw.exe (02 Dec 2016, 8:07:43 PM)
Dec 02, 2016 8:07:44 PM jade.core.Runtime begin
INFO: ----------
This is JADE 4.4.0 - revision 6778 of 21-12-
downloaded in Open Source, under LGPL restri
at http://jade.tilab.com/
Dec 02, 2016 8:07:45 PM jade.imtp.leap.LEAPIMTP
INFO: Listening for intra-platform commands on
- jicp://192.168.8.100:1099
Dec 02, 2016 8:07:45 PM jade.core.BaseService in
INFO: Service jade.core.management.AgentManageme
Dec 02, 2016 8:07:45 PM jade.core.BaseService in
INFO: Service jade.core.messaging.Messaging init
Dec 02, 2016 8:07:45 PM jade.core.BaseService in
INFO: Service jade.core.resource.ResourceManagem
Dec 02, 2016 8:07:45 PM jade.core.BaseService in
INFO: Service jade.core.mobility.AgentMobility i
Dec 02, 2016 8:07:45 PM jade.core.BaseService in
INFO: Service jade.core.event.Notification init
Dec 02, 2016 8:07:45 PM jade.mtp.http.HTTPServer <init>
INFO: HTTP-MTP Using XML parser org.apache.xerces.jaxp.SAXParserImpl$JAXPSAXParser
Dec 02, 2016 8:07:45 PM jade.core.messaging.MessagingService boot
INFO: MTP addresses:
http://KZNNOTE120.OldFort.co.za:7778/acc
Dec 02, 2016 8:07:45 PM jade.core.AgentContainerImpl joinPlatform
INFO: ----------
Agent container Main-Container@192.168.8.100 is ready.

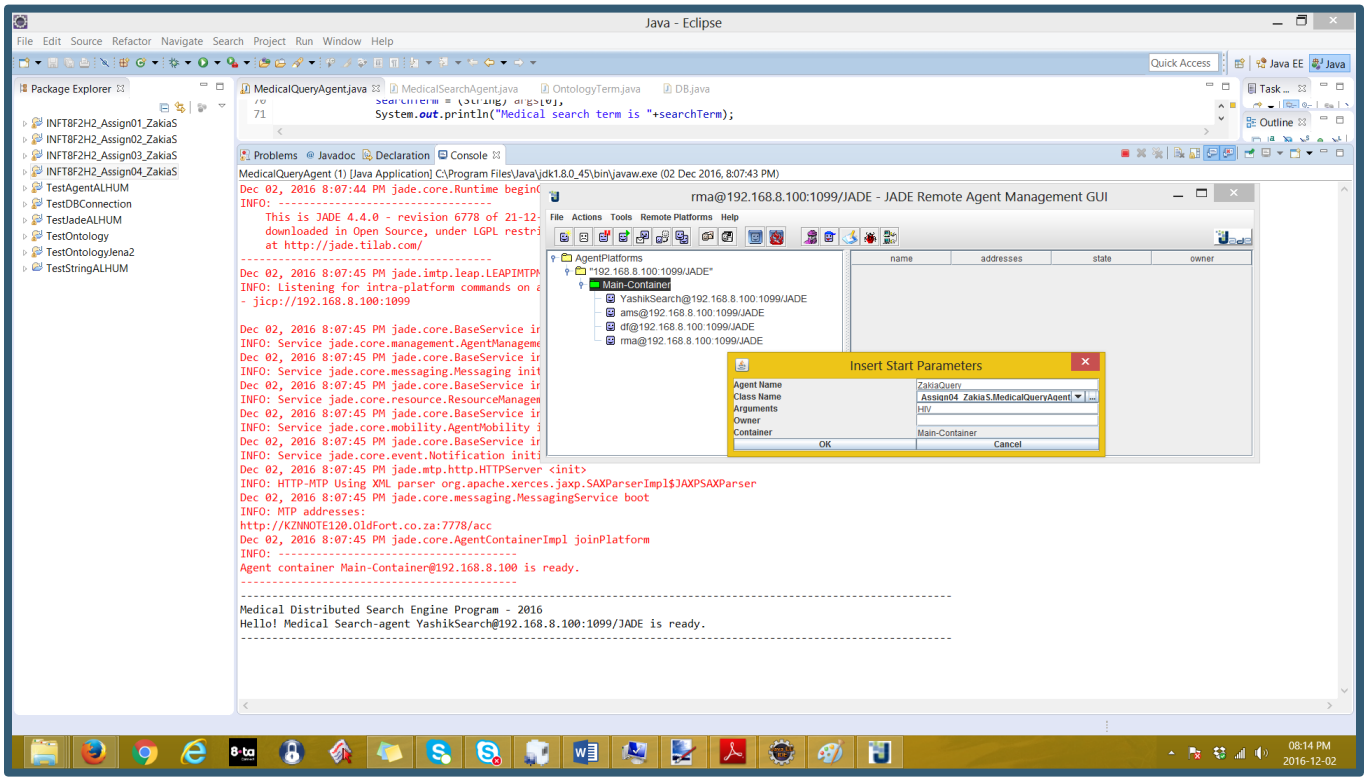
Medical Distributed Search Engine Program - 2016
Hello! Medical Search-agent YashikSearch@192.168.8.100:1099/JADE is ready.
rma@192.168.8.100:1099/JADE - JADE Remote Agent Management GUI
File Actions Tools Remote Platforms Help
AgentPlatforms
"192.168.8.100:1099/JADE"
Main-Container
YashikSearch@192.168.8.100:1099/JADE
ams@192.168.8.100:1099/JADE
df@192.168.8.100:1099/JADE
rma@192.168.8.100:1099/JADE
name
addresses
state
owner
YashikSearch@192.16...
active
NONE
08:11 PM
2016-12-02

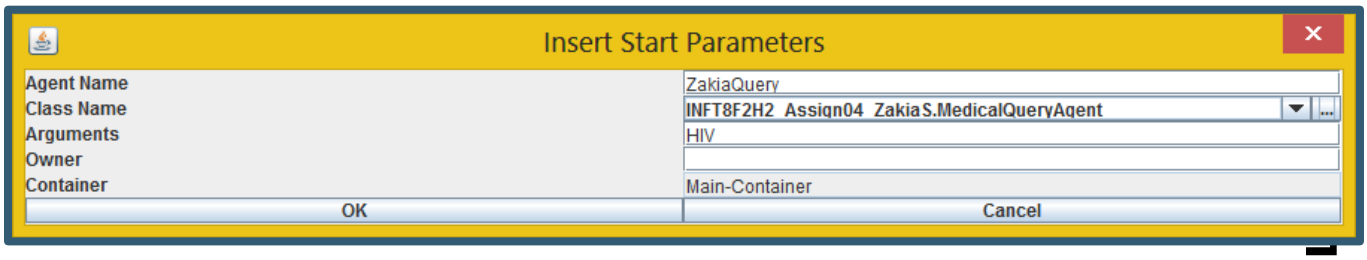

rma@192.168.8.100:1099/JADE - JADE Remote Agent Management GUI
File Actions Tools Remote Platforms Help
AgentPlatforms
"192.168.8.100:1099/JADE"
Main-Container
YashikSearch@192.168.8.100:1099/JADE
ams@192.168.8.100:1099/JADE
df@192.168.8.100:1099/JADE
rma@192.168.8.100:1099/JADE
name
addresses
state
owner
YashikSearch@192.16...
active
NONE

Creating Medical Query Agent Instance →

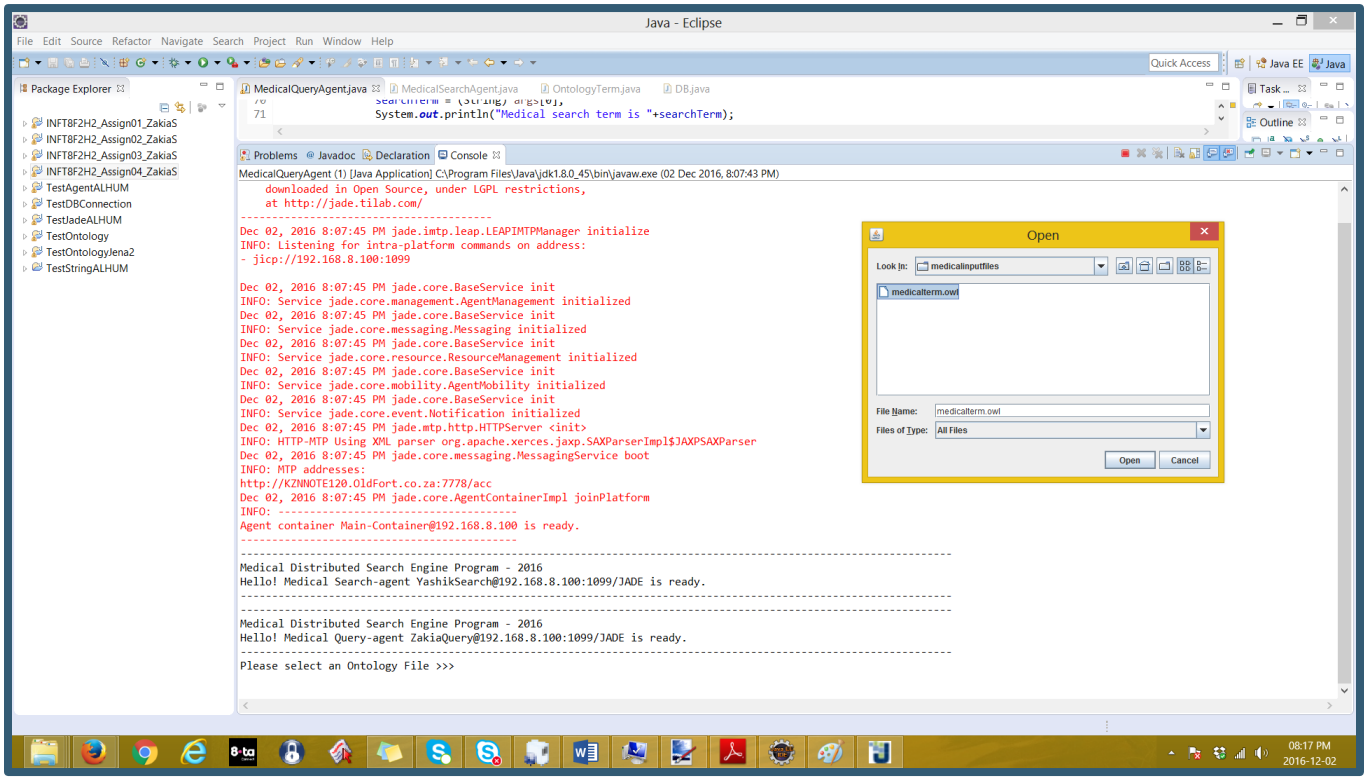


Right-Click 'Main-Container' → Start New Agent → Full-In Details of Agent

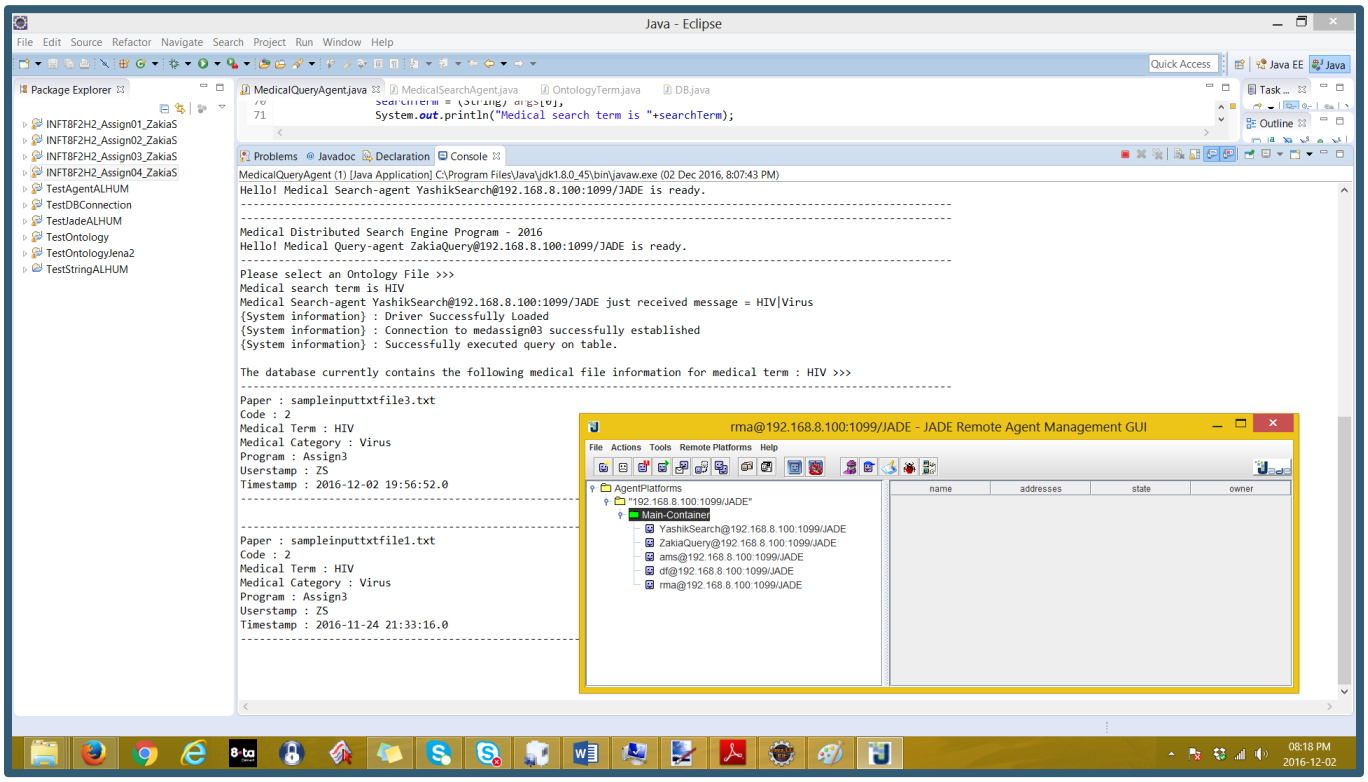




Click OK



Select the .OWL Ontology File & Click 'Open' on the Modal Dialog Above

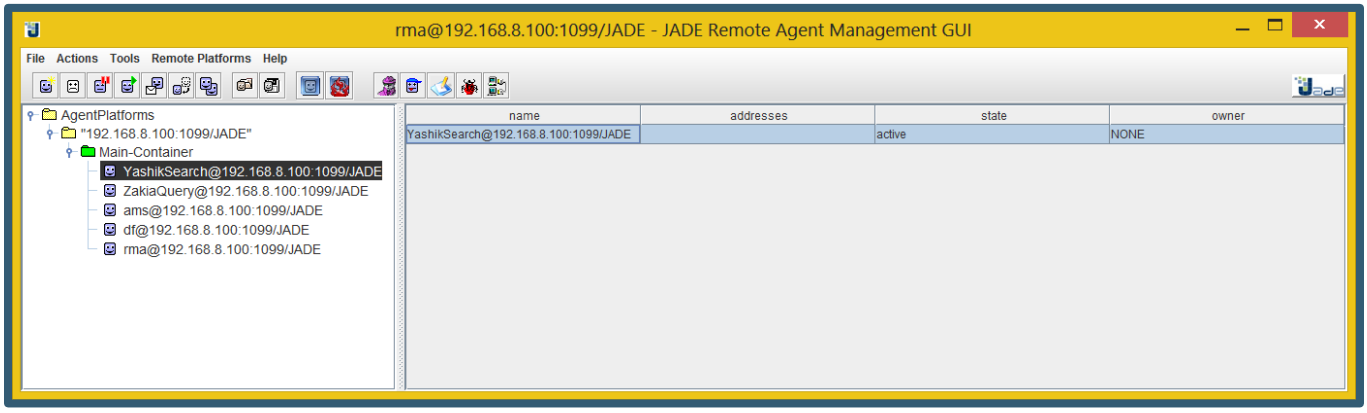


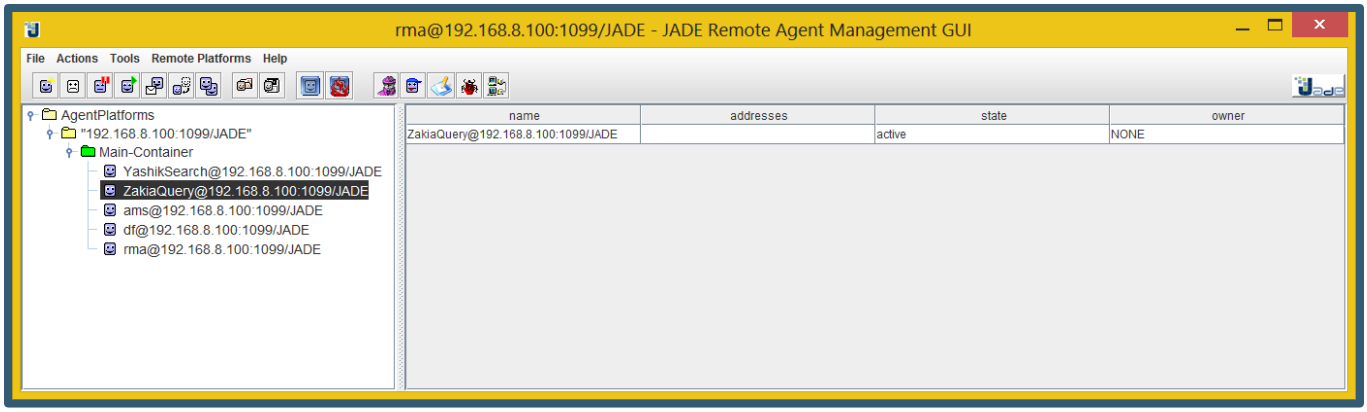
Entire Output on Console As Follows →

```
Dec 02, 2016 8:07:44 PM jade.core.Runtime beginContainer
INFO: ----------------------------------
    This is JADE 4.4.0 - revision 6778 of 21-12-2015 12:24:43
    downloaded in Open Source, under LGPL restrictions,
    at http://jade.tilab.com/
----------------------------------------
Dec 02, 2016 8:07:45 PM jade.imtp.leap.LEAPIMTPManager initialize
INFO: Listening for intra-platform commands on address:
- jicp://192.168.8.100:1099

Dec 02, 2016 8:07:45 PM jade.core.BaseService init
INFO: Service jade.core.management.AgentManagement initialized
Dec 02, 2016 8:07:45 PM jade.core.BaseService init
INFO: Service jade.core.messaging.Messaging initialized
Dec 02, 2016 8:07:45 PM jade.core.BaseService init
INFO: Service jade.core.resource.ResourceManagement initialized
Dec 02, 2016 8:07:45 PM jade.core.BaseService init
INFO: Service jade.core.mobility.AgentMobility initialized
Dec 02, 2016 8:07:45 PM jade.core.BaseService init
INFO: Service jade.core.event.Notification initialized
Dec 02, 2016 8:07:45 PM jade.mtp.http.HTTPServer <init>
INFO: HTTP-MTP Using XML parser org.apache.xerces.jaxp.SAXParserImpl$JAXPSAXParser
Dec 02, 2016 8:07:45 PM jade.core.messaging.MessagingService boot
INFO: MTP addresses:
http://KZNNOTE120.OldFort.co.za:7778/acc
Dec 02, 2016 8:07:45 PM jade.core.AgentContainerImpl joinPlatform
INFO: ---------------------------------------
Agent container Main-Container@192.168.8.100 is ready.
--------------------------------------------
------------------------------------------------------------------------------------------
----------------------
Medical Distributed Search Engine Program - 2016
Hello! Medical Search-agent YashikSearch@192.168.8.100:1099/JADE is ready.
------------------------------------------------------------------------------------------
----------------------
------------------------------------------------------------------------------------------
----------------------
Medical Distributed Search Engine Program - 2016
Hello! Medical Query-agent ZakiaQuery@192.168.8.100:1099/JADE is ready.
------------------------------------------------------------------------------------------
----------------------
Please select an Ontology File >>>
Medical search term is HIV
Medical Search-agent YashikSearch@192.168.8.100:1099/JADE just received message = HIV|Virus
{System information} : Driver Successfully Loaded
{System information} : Connection to medassign03 successfully established
{System information} : Successfully executed query on table.

The database currently contains the following medical file information for medical term :
HIV >>>
------------------------------------------------------------------------------------------
----------------------
Paper : sampleinputtxtfile3.txt
Code : 2
Medical Term : HIV
Medical Category : Virus
```

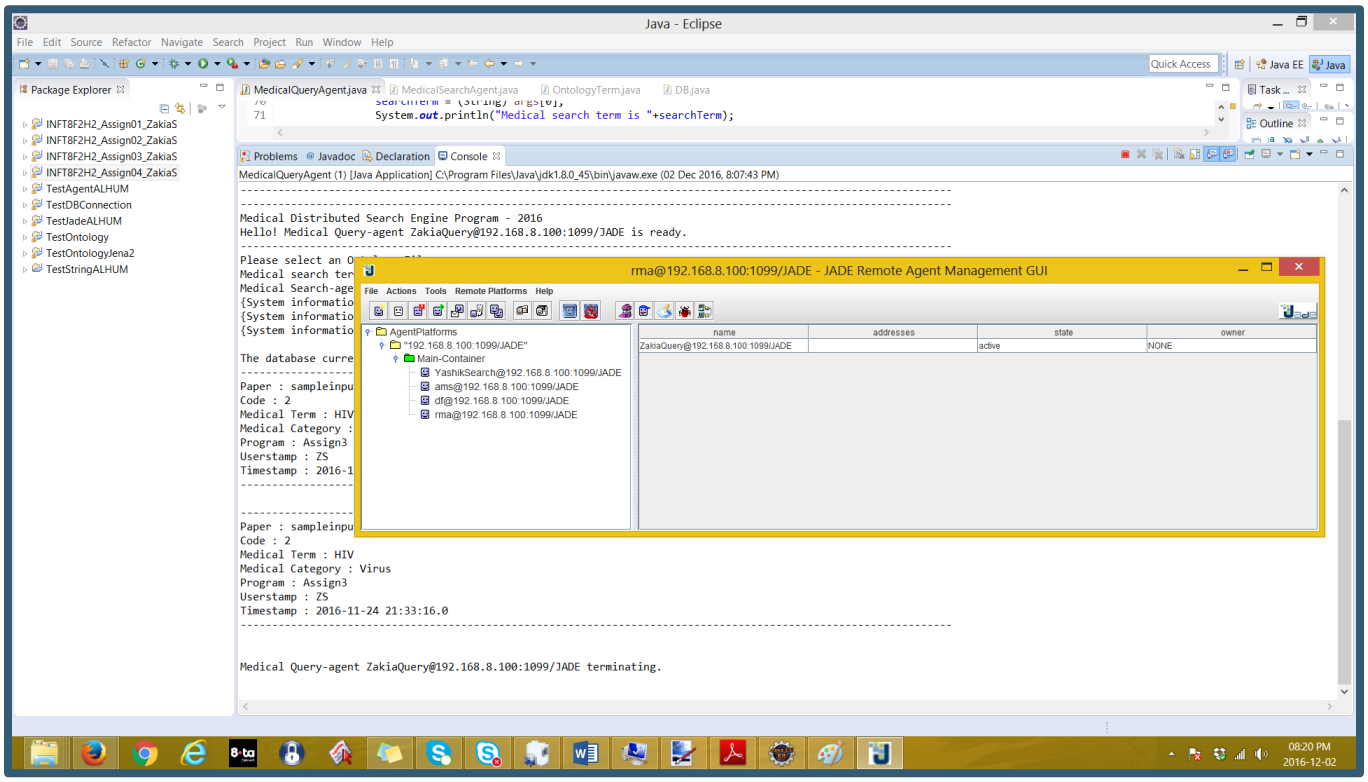
```
Program : Assign3
Userstamp : ZS
Timestamp : 2016-12-02 19:56:52.0
-------------------------------------------------------------------------------------------
----------------------

-------------------------------------------------------------------------------------------
----------------------
Paper : sampleinputtxtfile1.txt
Code : 2
Medical Term : HIV
Medical Category : Virus
Program : Assign3
Userstamp : ZS
Timestamp : 2016-11-24 21:33:16.0
-------------------------------------------------------------------------------------------
----------------------
```

Status of Agents →

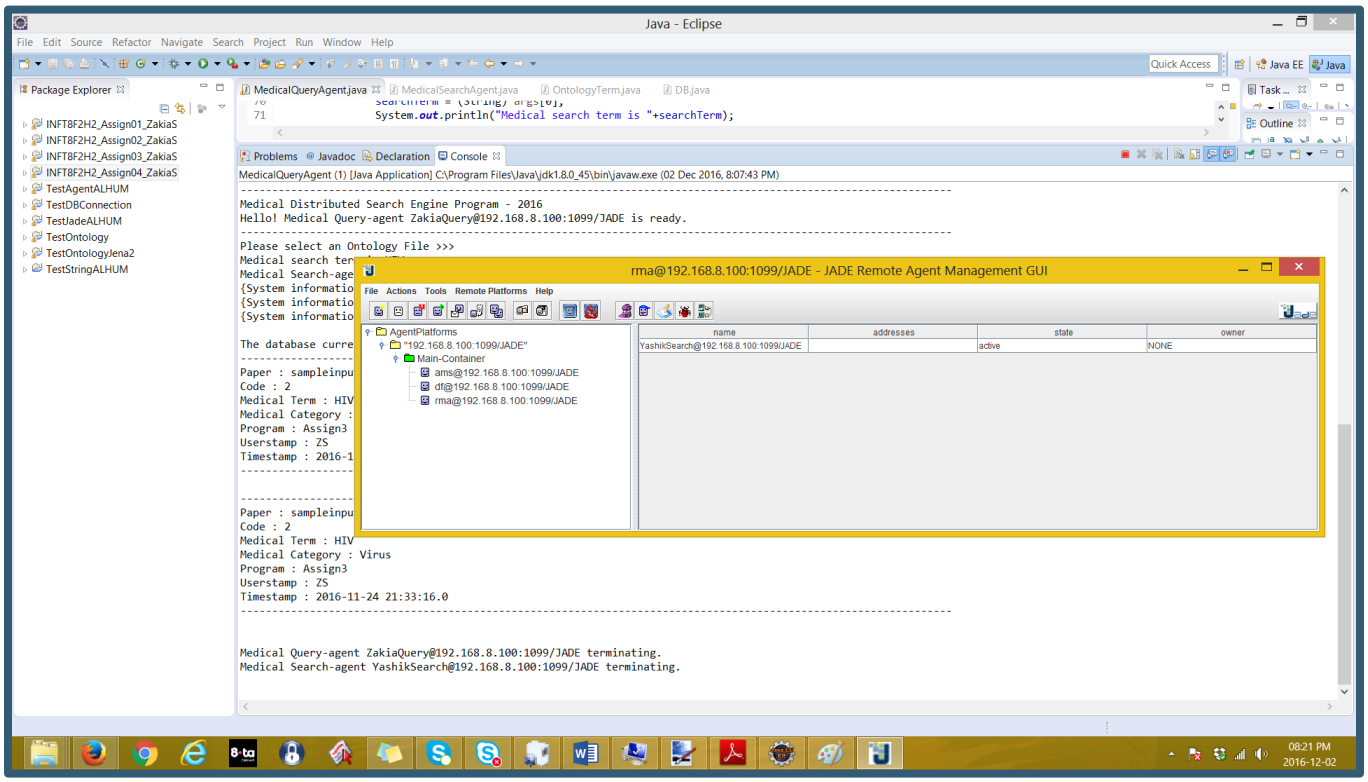




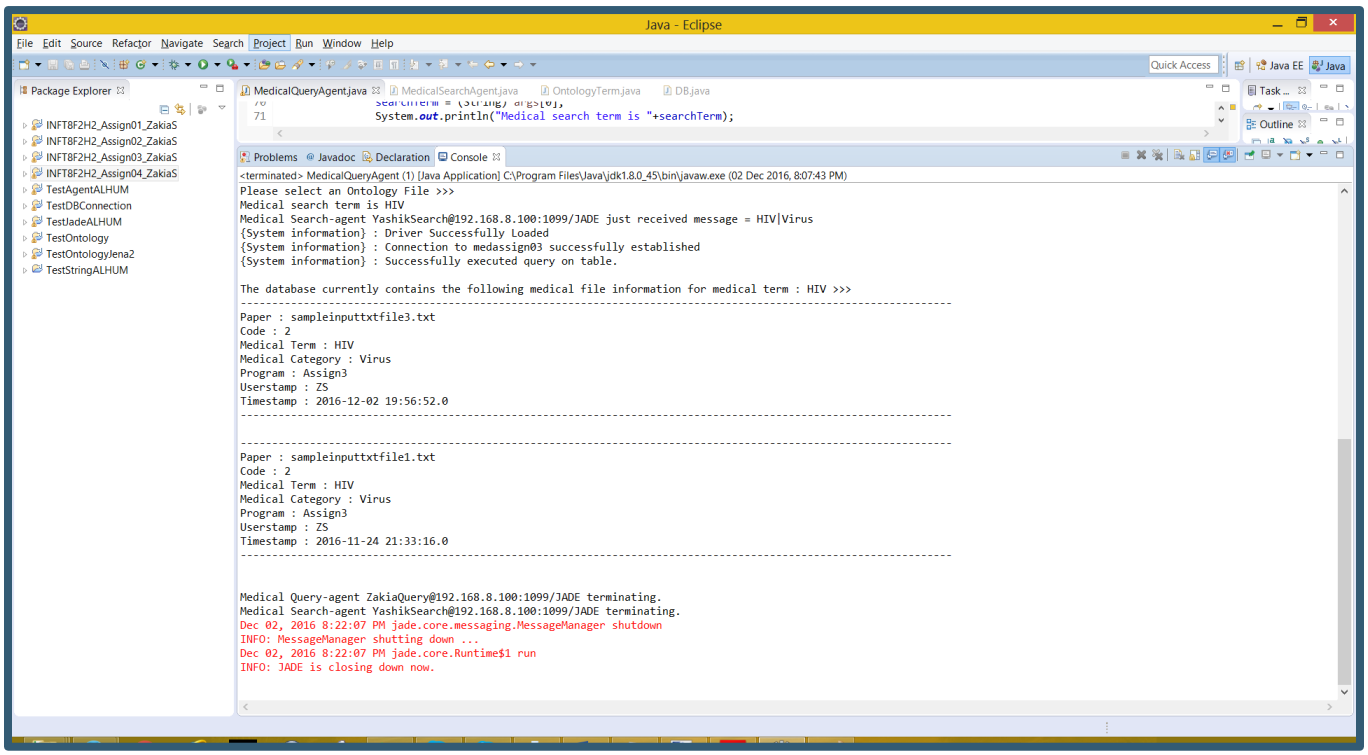
Kill ZakiaQuery Agent →



Kill YashikSearch Agent →



Kill Main-Container Platform



MEDICALQUERYAGENT.JAVA

```
/*************************************************************************
 *Bismillahir Rahmaanir Raheem
 *Almadadh Ya Gause Radi Allahu Ta'alah Anh - Ameen
 *Student Number : 208501583
 *Name : Zakia Salod
 *Course : INFT8F2H2
 *Assignment : 04
 *Masters of Medical Science - Medical Informatics
 *Year : 2016
 *************************************************************************/

package INFT8F2H2_Assign04_ZakiaS;

import jade.core.AID;
import jade.core.Agent;
import jade.lang.acl.ACLMessage;
import jade.lang.acl.MessageTemplate;
import jade.domain.DFService;
import jade.domain.FIPAException;
import jade.domain.FIPAAgentManagement.DFAgentDescription;
import jade.domain.FIPAAgentManagement.ServiceDescription;
import jade.core.AID.*;
import jade.core.behaviours.Behaviour;
import jade.core.behaviours.CyclicBehaviour;
import jade.core.behaviours.TickerBehaviour;
import jade.core.*;
import jade.lang.acl.ACLMessage;
import java.io.File;
import java.sql.SQLException;
import java.util.Iterator;
import javax.swing.JFileChooser;
import org.apache.jena.atlas.logging.LogCtl;
import org.apache.jena.ontology.OntClass;
import org.apache.jena.ontology.OntModel;
import org.apache.jena.ontology.OntModelSpec;
import org.apache.jena.rdf.model.ModelFactory;
import org.apache.jena.util.iterator.ExtendedIterator;

public class MedicalQueryAgent extends Agent {
    static final String database = "medassign03";
    static File getOntologyFile;
      //the medical search term to be searched
      private String searchTerm;
      //the list of dynamic search agents
      private AID[] searchAgents;
      private MessageTemplate mt;

      //agent initializations here
      protected void setup()
    {

            //printout a welcome message
```

```
            System.out.println("-------------------------------------------------------------
-------------------------------------------------------------");
          System.out.println("Medical Distributed Search Engine Program - 2016");
          System.out.println("Hello! Medical Query-agent "+getAID().getName()+" is
ready.");
          System.out.println("-------------------------------------------------------------
-------------------------------------------------------------");

            System.out.println("Ya Allah, Please help me");
            LogCtl.setCmdLogging();

            System.out.println("Please select an Ontology File >>>");
            JFileChooser fileChooser = new JFileChooser();

            //get the medical search term to be searched as a start-up argument
            Object[] args = getArguments();

            if (args != null && args.length > 0) {
                  if (fileChooser.showOpenDialog(null) == JFileChooser.APPROVE_OPTION){
                        getOntologyFile = fileChooser.getSelectedFile();
                        searchTerm = (String) args[0];
                        System.out.println("Medical search term is "+searchTerm);

                        //add a TickerBehaviour that schedules a request to medical
search agents every 5 seconds
                        addBehaviour(new TickerBehaviour(this, 5000){
                              protected void onTick(){
                                    DFAgentDescription template = new
DFAgentDescription();
                                    ServiceDescription sd = new ServiceDescription();
                                    sd.setType("medical-searching");
                                    template.addServices(sd);
                                    try{
                                          DFAgentDescription [] result =
DFService.search(myAgent, template);
                                          searchAgents = new AID[result.length];
                                          for(int i = 0; i<result.length; i++){
                                                searchAgents[i] = result[i].getName();
                                          }
                                    }//end try
                                    catch(FIPAException fe){
                                          fe.printStackTrace();
                                    }//end catch

                                    ACLMessage msg = new ACLMessage(ACLMessage.INFORM);
                                    for(int i=0; i<searchAgents.length; i++){
                                        msg.addReceiver(searchAgents[i]);
                                    }

                                    OntologyTerm getOntologyTerm =
searchUserTermAndOntology(searchTerm);
                                    msg.setContent((getOntologyTerm.getMedTerm().trim()
+ "|" +getOntologyTerm.getMedCategory().trim()).trim());
                                    msg.setConversationId("medical-searching");

      msg.setReplyWith("msg"+System.currentTimeMillis());//unique value
                                    myAgent.send(msg);
```

```
                                        mt =
MessageTemplate.and(MessageTemplate.MatchConversationId("medical-searching"),
MessageTemplate.MatchInReplyTo(msg.getReplyWith()));
                                    }
                            });
                   }//end if (fileChooser.showOpenDialog(null) == JFileChooser.APPROVE_OPTION)
            }//end if (args != null && args.length > 0)

            else {
                  // Make the agent terminate
                  System.out.println("No target medical search term specified");
                  doDelete();
            }
    }//end setup()



      //agent clean-up operations here
      protected void takeDown() {
            //printout a dismissal message
            System.out.println("Medical Query-agent "+getAID().getName()+" terminating.");
      }//end takeDown()



      //search the Ontology, by first looking at the user's search term - (searchTerm
passed-in as parameter)
      private static OntologyTerm 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())){
                                        try{
                                            //send-out medterm = ontologyClassStr and
medcategory = ontologyClassStr
                                               return new OntologyTerm(ontologyClassStr,
ontologyClassStr);
                                        }//end try
                                            catch(Exception e){
                                                    System.out.println("Error");
                                                    e.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())){
                                                try{
                                                    //send-out medterm =
c.getLocalName().toString() and medcategory = ontologyClassStr
                                                      return new
OntologyTerm(c.getLocalName().toString(), ontologyClassStr);
                                                }//end try
                                                catch(Exception e){
                                                        System.out.println("Error");
                                                        e.printStackTrace();
                                                }//end catch
                                    }//end if
                    }//end for
                        }//end if
                        else if(!ontologyClass.hasSubClass() &&
!ontologyClass.hasSuperClass()){
                                if
(searchTerm.toLowerCase().contains(ontologyClassStr.toLowerCase())){
                                    try{
                                        //send-out medterm = ontologyClassStr and medcategory
= ontologyClassStr
                                          return new OntologyTerm(ontologyClassStr,
ontologyClassStr);
                                    }//end try
                                        catch(Exception e){
                                               System.out.println("Error");
                                               e.printStackTrace();
                                        }//end catch
                                }//end if
                }
              }//end while

               //Medical Term not found in Ontology
               if(!foundUserMedicalTermInOnt){
                      return new OntologyTerm("", "");
               }

               //default return
               return new OntologyTerm("", "");
       }//end searchUserTermAndOntology
}//end MedicalQueryAgent class
```

MEDICALSEARCHAGENT.JAVA

```
/*********************************************************************
 *Bismillahir Rahmaanir Raheem
 *Almadadh Ya Gause Radi Allahu Ta'alah Anh - Ameen
 *Student Number : 208501583
 *Name : Zakia Salod
 *Course : INFT8F2H2
 *Assignment : 04
 *Masters of Medical Science - Medical Informatics
 *Year : 2016
 *********************************************************************/

package INFT8F2H2_Assign04_ZakiaS;

import java.sql.*;
import jade.core.AID;
import jade.core.Agent;
import jade.domain.DFService;
import jade.domain.FIPAException;
import jade.domain.FIPAAgentManagement.DFAgentDescription;
import jade.domain.FIPAAgentManagement.ServiceDescription;
import jade.lang.acl.ACLMessage;

public class MedicalSearchAgent extends Agent{
      //private AID[] searchAgents = {new AID("AllahYay", AID.ISLOCALNAME)};
    static final String database = "medassign03";

    //agent initializations here
      protected void setup()
    {

            //printout a welcome message
        System.out.println("-------------------------------------------------------------
-------------------------------------------------------");
          System.out.println("Medical Distributed Search Engine Program - 2016");
          System.out.println("Hello! Medical Search-agent "+getAID().getName()+" is
ready.");
          System.out.println("-------------------------------------------------------------
-------------------------------------------------------");


        DFAgentDescription dfd = new DFAgentDescription();
        dfd.setName(getAID());
        ServiceDescription sd = new ServiceDescription();
        sd.setType("medical-searching");
        sd.setName("JADE-medical-searching");
        dfd.addServices(sd);

        try{
             DFService.register(this, dfd);
        }//end try
        catch(FIPAException fe){
             fe.printStackTrace();
        }//end catch


        ACLMessage msg = null;
```

```
        msg = blockingReceive();
        System.out.println("Medical Search-agent "+getAID().getName() + " just received
message = " + msg.getContent());

             try{
                   //search the database and display results
                   searchDBAndDisplay(msg.getContent().substring(0,
msg.getContent().indexOf("|")),
msg.getContent().substring(msg.getContent().indexOf("|")+1));
             }//end try
                 catch(SQLException sqlException){
                        System.out.println("Error");
                        sqlException.printStackTrace();
                 }//end catch
    }//end setup()



      //agent clean-up operations here
      protected void takeDown() {
             //de-register from the yellow pages
             try{
                   DFService.deregister(this);
             }
             catch(FIPAException fe){
                   fe.printStackTrace();
             }

             //print-out a dismissal message
             System.out.println("Medical Search-agent "+getAID().getName()+"
terminating.");
      }//end takeDown()



      //searches the database and displays the output from the database for the specified
medical term
      private static void searchDBAndDisplay(String medterm, String medcategory) throws
SQLException{
                 int countTracker = 1;
                 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()){
                                   if (countTracker ==1){
                                     System.out.println("The database currently contains the
following medical file information for medical term : " + medterm + " >>>");
                                   }
                                   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();

                                countTracker +=1;
                            }//end while

                           //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);
                            }//end if

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

ONTOLOGYTERM.JAVA

```java
/*********************************************************************
 *Bismillahir Rahmaanir Raheem
 *Almadadh Ya Gause Radi Allahu Ta'alah Anh - Ameen
 *Student Number : 208501583
 *Name : Zakia Salod
 *Course : INFT8F2H2
 *Assignment : 04
 *Masters of Medical Science - Medical Informatics
 *Year : 2016
 *********************************************************************/

package INFT8F2H2_Assign04_ZakiaS;

public final class OntologyTerm {
       private final String medTerm;
    private final String medCategory;

    public OntologyTerm(String medTerm, String medCategory) {
        this.medTerm  = medTerm;
        this.medCategory = medCategory;
    }//end OntologyTerm() Constructor

    public String getMedTerm() {
        return medTerm;
    }

    public String getMedCategory() {
        return medCategory;
    }
}//end OntologyTerm class
```

DB.JAVA

```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;
	}
```

```java
        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