```java
package DBLayer;
import ModelLayer.*;

import java.sql.*;

/**
 * @author DM79_group 3
 *  @version December 2012.
 */


public class DBBiteDisease implements IFDBBiteDisease
{
   private Connection con;

   public DBBiteDisease()
   {
         con = DbConnection.getInstance().getDBcon();
   }

   @Override
   public BiteDisease searchBiteDiseaseById(int diseaseId,
   boolean retriveAssociation)
   {
         String  wClause = "diseaseId = '" + diseaseId + "'";
         System.out.println("SearchBiteDisease " + wClause);
         return singleWhere(wClause, retriveAssociation);
   }


   @Override
   // find bite disease using the name.
   public BiteDisease searchBiteDiseaseByName(String
   diseaseName, boolean retriveAssociation)
   {
         String wClause = "diseaseName like '%" + diseaseName +
         "%'";
       System.out.println("Search BiteDisease " + wClause);
       return singleWhere(wClause, retriveAssociation);
   }


   @Override
   public int updateBiteDisease(BiteDisease bitedis)
   {
         BiteDisease biteObj  = bitedis;
         int rc=-1;

         String query="UPDATE mfBiteDisease SET "+
                     "diseaseName ='"+ biteObj.getDiseaseName() + "' " +
                     "WHERE diseaseId ='"+ biteObj.getDiseaseId()+"'";

       System.out.println("Update query:" + query);
         try
         {
               //update mfBiteDisease.
               Statement stmt = con.createStatement();
               stmt.setQueryTimeout(5);
               rc = stmt.executeUpdate(query);
               stmt.close();
         }//end try
         catch(Exception ex)
         {
               System.out.println("Update exception in BiteDisease DB: "+
```

```
65                ex);
66          }
67
68          return(rc);
69      }
70
71
72      @Override
73      public int insertBiteDisease(BiteDisease bitedis)
74       {
75          int rc = -1;
76          PreparedStatement pstmt = null;
77          String insert = "INSERT INTO mfBiteDisease (
78          diseaseId, diseaseName)" + "values (?,?)";
79          System.out.println(insert);
80          try
81          {
82           pstmt = con.prepareStatement(insert);
83              pstmt.setInt   ( 1, bitedis.getDiseaseId() );
84              pstmt.setString( 2, bitedis.getDiseaseName() );
85              rc = pstmt.executeUpdate();
86          }
87           catch(SQLException sqlE)
88           {
89              System.out.println("SQL Error");
90              System.out.println(sqlE.getMessage());
91           }
92          catch(Exception e)
93          {
94              e.getMessage();
95          }
96
97          return rc;
98       }
99
100
101     @Override
102     public int deleteBiteDiseaseWithId(int diseaseId)
103       {
104          int rc = -1;
105          PreparedStatement pstmt = null;
106          String delete = "DELETE FROM mfBiteDisease WHERE
107          diseaseId = ?";
108          System.out.println(delete);
109          try
110          {
111                pstmt = con.prepareStatement(delete);
112              pstmt.setInt(1,diseaseId);
113              rc = pstmt.executeUpdate();
114          }
115          catch(SQLException sqlE)
116          {
117              System.out.println("SQL Error");
118              System.out.println(sqlE.getMessage());
119          }
120          catch(Exception e)
121          {
122              e.getMessage();
123          }
124
125          return rc;
126       }
127
128
```

```
129   //Singelwhere is used when we only select one BiteDisease.
130   private BiteDisease singleWhere(String wClause, boolean
131   retrieveAssociation)
132   {
133         ResultSet results;
134
135         String query =  buildQuery(wClause);
136         System.out.println(query);
137         try
138         {
139               //read the disease from the database.
140               Statement stmt = con.createStatement();
141               stmt.setQueryTimeout(5);
142               results = stmt.executeQuery(query);
143
144               if( results.next() )
145               {
146                     BiteDisease bitedisObj = buildBiteDisease(results);
147                     stmt.close();
148                     return bitedisObj;
149               }
150               else
151               {
152                     //no Bite disease was found.
153                     return null;
154               }
155
156         }
157         //end try.
158         catch(Exception e){
159               System.out.println("Query exception: "+e);
160         }
161
162         return null;
163   }
164
165
166   //method to build the query
167   private String buildQuery(String wClause)
168   {
169         String query = "SELECT diseaseId, diseaseName FROM
170         mfBiteDisease";
171         if (wClause.length() > 0)
172               query = query + " WHERE "+ wClause;
173         return query;
174   }
175
176
177   //method to build a bite disease object.
178   private BiteDisease buildBiteDisease(ResultSet results)
179   {
180         try
181         {
182               //use columns from mfBiteDisease table.
183               BiteDisease bitedisObj = new BiteDisease(results.getInt(
184               "diseaseId"),results.getString("diseaseName"));
185               return bitedisObj;
186
187               // Alternativ way to do this.
188               //return new BiteDisease(results.getInt("diseaseId"),
189               results.getString("diseaseName"));
190         }
191         catch (Exception e)
192         {
```

```
193                 System.out.println("Error building the Disease Bite
194                 object.");
195         }
196 
197         return null;
198   }
199 
200 }// end of DBBiteDisease class.
```