public class  BusinessLogic  
{

IFileType interfaceObj;

FileFactory factoryObj = new FileFactory();

public BusinessLogic(string InputFileType)

{

this.interfaceObj = factoryObj.getFile(InputFileType)

}

public static void main(string[] args)

{

BusinessLogic mainClassObj1= new BusinessLogic(“yaml”),

mainClassObj1.importFile();

BusinessLogic mainClassObj2= new BusinessLogic(“json”),

mainClassObj2.importFile();

/\* we can write a getGile() method to iterate files in the mentioned directory path      i.e /feed-products to get extension of files and create respective class Object in If Else condition block.\*/

}

}

}

Public Class FileFactory{

public static IFileType getFile(string type){

if (type==json){

return new JsonFile();

}

else If(type==yaml){

return new YamlFile();

}

}

}

public interface IFileType

{

void importFile();

}

public class JsonFile: IFileType

{ IDictionary data = new Hashtable();

public void importFile()

{

//way to read and import yaml File and

data.add(“tags”, value);

data.add(“name”, value);

data.add(“twitter, value”)

//pass value to DBFunction

DBFunction DBObj= new DBFunction();

DBObj.ImportFileinDB(data);

}

}

public class yamlFile: IFileType

{ IDictionary data = new Hashtable();

public void importFile()

{

//way to read and import yaml File and

data.add(“categories”, value);

data.add(“title”, value);

data.add(“twitter, value”)

//pass value to DBFunction

DBFunction DBObj= new DBFunction();

DBObj.ImportFileinDB(data);

}

}

Public Class DatabaseConnection

{

Private static DBConnection(){

try

{

//get Database connection

return connection;

}

catch(Exception e){

log.Error(er);

return null;

}

}

public DataTable ExecuteQuery(SQLCommand cmd, String action)

{

try

{

//use connection

Switch(action)

{

// switch cases for INSERT, UPDATE, SELECT etc queries

case “SELECT”:

//open connetion,

//execute Query

 return dataTable;

case “INSERT”:

//open connetion

 //execute Query

  break;

}

}

finally

{

//close connection

}

}

public class DatabaseFunction:DatabaseConnection

{

/\*contains methods where we are writing query and passing them as parameters to call method ExecuteQuery() in Parent Class

/return ExecuteQuery(cmd, “INSERT”); \*/

}