Filière :	Développement des Systèmes d'Information	Durée :	4 Heures
Épreuve :	Développement des Applications informatiques DAI	Coefficient :	45

ÉTUDE DE CAS : GESTION DE RECYCLAGE DES COMPOSANTS ÉLECTRONIQUES

DOSSIER 1 : Système d'information de recyclage de la NAT-AMI *(12 pts)*

1.

1.1
```
public class Composant {
    private String reference;
```
(0,5 pt = 0,25 pt + 0,25 pt)

1.2
```
public Composant(String ref) { this.reference=ref;}
}
```
(0,5 pt)

2.

2.1
```
public class CarteMere extends Composant {
    private String designation;
```
(0,5 pt = 0,25 pt + 0,25pt)

2.2
```
public CarteMere(String ref, String desg) { super(ref);  this.designation=desg;}
}
```
(0,5 pt)

3.

3.1
```
public class Procede implements Serializable {
    private String reference;
    private double prix;
```
(0,5 pt = 0,25 pt + 0,25 pt)

3.2 + 3.3 *(1,5 pt = 1 pt (Constructeur)+ 0,5 pt (Exception))*
```
public Procede(String ref, double pr) throws ErreurPrix
{       if(pr<0) throw new ErreurPrix("Erreur prix : "+pr);
        this.reference=ref;  this.prix=pr;}
```

3.4
```
public String getReference(){ return this.reference;}
}
//-----------------------------------------------------------------------------------------------
public class ErreurPrix extends Exception{
public ErreurPrix(String msg){ super(msg);}}
```
(0,5 pt)

4.

4.1
```
public class Traitement{
        private Date dateDebut, dateFin;
        private Procede prd;
        private Composant comp;
```
(0,5 pt = 0,25 pt + 0,25pt)

4.2
```
public Traitement (Date dd, Date df, Procede p, Composant cmp)
{       this.dateDebut=dd;  this.dateFin=df;    this.prd=p;      this.comp=cmp;}
```
(0,5 pt)

4.3
```
@Override
public String toString(){
        SimpleDateFormat f= new SimpleDateFormat("dd/MM/yyyy");
        return "Date début : "+ f.format (dateDebut)+", date fin : "+f.Format(dateFin) + ", Procédé : "+
        prd+"Composant : "+comp;}
        }
```
(0,5 pt)

5.

5.1

```
public class Laboratoire{
private int idLab;
private Sting nom;
private ArrayList<Procede> liste;
```

(0,5 pt = 0,25 pt + 0,25 pt)

5.2 *(0,5 pt)*

```
public Laboratoire (int id,String nom)
{        this.idLab=id;    this.nom=nom;    liste=new ArrayList();}
```

5.3 @Override

```
public String toString(){
        String m="Id Labo : "+idLab+", Nom : "+nom+"\n";
        for(Procede p:liste) m+=p+"\n";
        return m;
        }
}
```

(0,5 pt)

5.4

a.) *(0,5 pt)*

```
public boolean addProcede(Procede p)
    { return liste.add(p);}
```

b.) *(0,5 pt)*

```
public Procede delProcede(int index)
    { try{ return liste.remove(index);}
    catch (Exception e)
            { return null;}}
```

c.) *(1,5 pt)*

```
public Procede searchProcede(String ref)
  { for(Procede p:liste) if (p.getReference().equals(ref)) return p;
 return null;
}
```

d.)

(1 pt)

```
public boolean enregistrer(String f)
 { try { ObjectOuputStream out=new ObjectOutputStream(new FileOutputStream(f));
out.writeObject(liste);  out.flush();  out.close(); return true;
}
 catch(Exception e) { return false;}
}
```

e.

(1 pt)

```
public boolean charger(String f)
 { try { ObjectInputStream in=new ObjectInputStream(new FileInputStream(f));
    liste=(ArrayList)in.readObject();  in.close(); return true;
}   catch(Exception e) { return false;}
}
}
```

DOSSIER 2 : Consultation des mapping des adresses IP par pays *(10 pts)*

➢ **Architecture :**

1. Il s'agit d'une architecture 3 tiers, en effet, il existe trois intervenants : Client, Serveur de base de données et serveur d'application. *(1 pt)*

2. Modèle Gartner : *(1 pt)*

➢ **Côté Serveur :**

3. Classe de connexion à la base de données :

```
public class SingletonConnection{
private Connection conn=null;
private static SingletonConnection  objConn=null;
3.1 private SingletonConnection(){                                    (1 pt)
      try{conn=DriverManager.getConnection("jdbc:mysql://10.50.49.10:3306/Suivi_acces",
            "admin","123456");}
      catch(SQLException e){e.printStackTrace();}}

      public static SingletonConnection getConn()
      {  if(objConn==null)
            objConn=new SingletonConnection();
            return objConn;
      }

3.2 public ResultSet  lire(String req)                                (1 pt)
      { try{Statement st=conn.createStatement();
            return st.executeQuery(req);}
            catch(SQLException e){return null;}
      }
3.3  public int  ecrire(String req)                                   (1 pt)
      { try{Statement st=conn.create Statement();
            return st.executeUpdate(req);}
            catch(SQLException e){return -1;}
            }
      }
```

4. Classe du serveur

```
public class Serveur{
private ServerSocket  serv;
private Socket ss;
ObjectOutputStream  out;
BufferedReader in;

4.1   public Serveur()                                                (1 pt)
{try{serv=new ServerSocket(4000);}  catch(Exception e){e.printStackTrace();}}
```

4.2 public void **start**() *(2 pts)*

```
{
        try{ss=serv.accept();
        out=new ObjectOutputStream(ss.getOutputStream());
        in=new BufferedReader(new InputStreamReader(ss.getInputStream()));
        String c; ArrayList<String> T;
        while(true)
                {c=in.readLine();
                T=new ArrayList();
ResultSet rs= SingletonConnection.getConn().lire("select * from Adresses_IP where code_pays='"+c+"'");
        while(rs.next()) T.add(rs.getString(1)+":"+rs.getString(2));
        out.writeObject(T); out.flush();
        }
        } catch(Exception e){e.printStackTrace();}
} }
```

5. Côté client

```
public  class Client {
        private String IP;
        private int  Port;
        private  Socket s=null;
        private  PrintWriter   pw=null;
        private ObjectInputStream  ois=null;
        public  Client   (String IP, int Port){
                this.IP=IP;
                this.Port=Port;
        }
```

5.1 private boolean **setconnexion()** *(1 pt)*

```
        { try{
        s= new Socket(IP,Port);
        pw=new PrintWriter(new OutputStreamWriter(s.getOutputStream()),true);
        ois =new ObjectInputStream(s.getInputStream());
           return true;
             }catch(Exception e){return false;}
    }
```

5.2 private ArrayList<String> demandeAdressesIP(String code) *(1 pt)*

```
    {
         try{
           pw.println(code);
           return (ArrayList) ois.readObject();
            }catch(Exception e){ return null;}
        }
        }
```

DOSSIER 3 : INVENTAIRE DES LOGICIELS INSTALLÉS *(10pts)*

1. Public con as SqlConnection

Public Function Connexion() as Boolean *(1,5pt)*

```
Try
    con= new SqlConnection("initial catalog=BD_Installation;data source=Serv1_Nat;
    integrated security=true;")
    con.open()
    return True
Catch
     return False
End Try
End Function
```

2. La procédure « **Afficher_Liste** » *(2 pts)*

```
Sub Afficher_Liste(ByVal Du As Date, ByVal Au As Date)
    dim req as String
    req="select Num_Installation as [N° Installation], DateInstallation as [Date Installation],
      NumPost as [N° Poste], Marque, Modele as [Modèle], Libellé as [Libellé Logiciel],
      Propriétaire, Version, Remarque from installation i, poste p, logiciel l
      where i.NumPoste=p.NumPoste and i.codeLogiciel=l.CodeLogiciel and
      DateInstallation between Du and Au"
    dim cmd as new SqlCommand(req,con)
    Dim rd as SqlDataReader=cmd.ExecuteReader()
    Dim T As new DataTable
    T.load(rd)
    rd.close()
    DGVListe.DataSource=T
End Sub
```

« **DGVListe**» :

	N° Installation	Date Installation	N° Poste	Marque	Modèle	Libellé Logiciel	Proporiétaire	Version	Remarque
▶*									

3. La procédure « **Supprimer_Installation** » *(1,5 pt)*

```
Private Sub Supprimer_Installation(ByVal Num As String)
   Dim cmd As New SqlCommand("delete from Installation where Num_Installation='"+Num+"'",con)
   cmd.ExecuteNonQuery()
End Sub
```

4. La fonction « **Generer_Num_Installation** » *(2 pts)*

```
Private Function Generer_Num_Installation() As String
   Dim d as DateTime = DateTime.Now
   Dim cmd As New SqlCommand("select count(*) from Installation
                        where year(DateInstallation)='"+d.Year+"'",con)
    Dim n As Integer=cmd.ExecuteScalar()
    n=n+1
    Dim m as String=format(n,"0000")+"/"+d.Year
   return m
End Function
```

5. La procédure « **Lister_Logiciel** » *(1 pt)*

```
Sub Lister_Logiciel()
      Dim cmd As New SqlCommand("select * from Logiciel",con)
      Dim rd As SqlDataReader = cmd.ExcuteReader()
      Dim T As New DataTable
     T.load(rd)
    rd.Close()
    CmbLogiciel.DisplayMember="Libellé"
    CmbLogiciel.ValueMember="CodeLogiciel"
    CmbLogiciel.DataSource=T
End Sub
```

6. La procédure « **Statistique** » *(2 pts)*

```
Sub Statistique( )
   dim req as String
   req="select Libellé As [Libellé Logiciel], count(NumInstallation) As [Nombre d'Installations]
         from installation i, logiciel l where i.codeLogiciel=l.CodeLogiciel group By i.codeLogiciel"
   dim cmd as new SqlCommand(req,con)
   Dim rd as SqlDataReader=cmd.ExecuteReader()
   Dim T As new DataTable
  T.load(rd)
  rd.close()
  DGV_Statistiques.DataSource=T
End Sub
```

DOSSIER 4 : VALIDATION DES DEMANDES DE RAMASSAGE DES DÉCHETS *(10 pts)*

1. La fonction **`getConnexion ()`** *(1,5 pt)*

```
function getConnexion()  {
    $con =new mysqli("10.50.49.10","admin","123456","DB_Dechets");          (1 pt)
    if(mysqli_connect_errno())                                              (0,5 pt)
   {
       die(mysqli_connect_error());
   }
      return $con ;
  }
```

2. La fonction **`getInValidDemandes`** ( ) *(2 pts)*

```
function getInValidDemandes()
{
    $con= getConnexion();
    $res=$con->query("select * from demandes where statut='invalide'");
     While ($row=$res->fetch_row())   {
      echo "<tr><td>$row[1]</td><td>$row[3]</td><th><a>
     href='detailsDemandes.php?id=$row[0]'></a></th><td>$row[4]</td></tr>" ;
   }
}
```

Le code de cette page web est le suivant :

```
<meta charset="utf8">
    <link rel="stylesheet" type="text/css" href="style.css">
<?php
    include( 'fonctions.php');
    $id=$_GET['id'];   // ID de la demande récupéré depuis la page d'accueil
    echo "<h1>Détails de la demande Numéro: $id</h1>";
    echo "<table>";
    echo "<tr><th>Article</th><th>Quantité</th></tr>";
    getdetailsDemandes($id);
    echo "</table>";
    echo "<a href='traitement.php?id=$id'>Valider la demande</a>";
    echo "<br><a href='demandes.php'>Retour vers les demandes</a>";
?>
```

3. La fonction `getdetailsDemandes($id)` *(2 pts)*

```
function getdetailsDemandes($id)
{
    $con= getConnexion();
     $res=$con->query("select LibelleArticle,qte from lignedemandes l,article a
                        where l.idArticle=a.idArticle and l.idDemande=$id");
   while($row=$res->fetch_row())  {
       echo "<tr><td>$row[0]</td><td>$row[1]</td></tr>";
   }
}
```

Un clic sur le lien <<**Valider la demande**>> permet d'ouvrir la page <<**traitement.php**>>. Elle contient le script suivant :

```
1-      <meta charset="utf8">
2-      <?php
3-          include 'fonctions.php';
4-         $id=$_GET['id'] ;
5-          updateDemande($id);
6-          echo "<script> .....</script>";   // à compléter
7-           echo "<script>document.location.href='index.php';</script>";
8-      ?>
```

4. La fonction **validerDemande($id)** *(1,5 pt)*

```
function validerDemande($id)
{
    $con= getConnexion();
   $con->query("update  demandes  set statut='valide' where idDemande=$id");
}
```

5. `echo "<script>alert('Demande validée avec succès');</script>";` *(1 pt)*