EXAMEN JEE TEST 01/XX
WWW.PANDACODEUR.COM

Proposé par: GROUPE GENIUS
par: Joël Yk

Exercice 1:

1) Un serveur Http: permet la gestion des requêtes et réponses Http suivant l'architecture Client-Serveur.
exemple: nginx, Apache

2) a) Rôle du serveur Http: Le serveur Http reçoit la requête l'interprète et genere alors une page Web qu'il enverra au client par le biais d'une reponse Http.

b) Rôle du navigateur:

- Il envoie tout d'abord une requête Http au serveur pour lui demander la page correspondante.

- Il reçoit la réponse du serveur, via cette réponse il affiche la page web finale à l'utilisateur.

3) Les éléments constitutifs d'un serveur d'application:

- Le serveur HTTP
- Le Conteneur Web // conteneur de Servlet

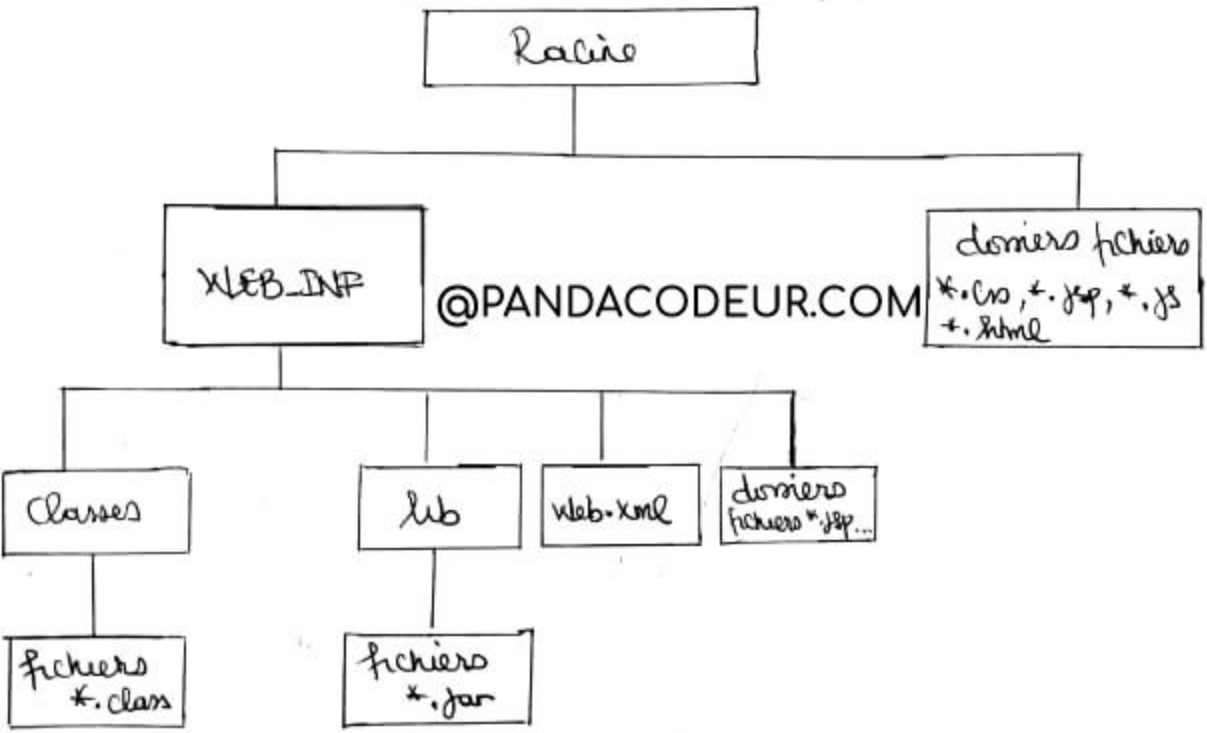
4) Un Framework: est un ensemble de composants qui servent à créer l'architecture et les grandes lignes de notre application.

exemple : Spring, Hibernate, JSF, Struts

5) Un EDI : C'est un ensemble d'outils qui permet d'augmenter la productivité des programmeurs qui developpent des logiciels.

EDI = Environnement Developpement Intégré

6) Structure des fichiers d'une application Web JEE.



Exercice 2 :

1) Le rôle d'un serveur web est de pouvoir traiter une requête HTTP.

Le rôle d'un serveur d'application est de fournir une infrastructure de services pour l'execution d'applications.

2) Elements logiciels constitutifs :

→ Serveur Http
→ Le Serveur d'application
→ Le Conteneur JSP.
→ Conteneur de Servlet
→ La plate forme JEE.

3) Difference entre les commandes GET & POST :

La difference fondamentale entre les méthodes GET & POST est que selon un critère de bonne pratique la méthode POST doit être utilisée pour réaliser les opérations qui ont un effet sur la ressource, alors que la méthode GET est la méthode utilisé par le client pour récuperer une ressource Web au Serveur via une URL.

4) Elle est traité par la méthode `doPost()`

Sa signature :

doPost(HttpServletRequest, HttpServletResponse)

5) Le type de l'objet fourni par une Servlet qui porte la reponse du client est : HttpServletResponse.

6) Le type de flot de sortie est : PrintWriter.

Problème

PARTIE A : Couche accès aux données

1) Créer les classes entités dans le package "ma.projet.entity"

2) Client.java.

```
package ma.projet.entity;

import java.util.Date;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;

@Entity
public class Client extends User
    @GeneratedValue
        private String nom;
        private String prenom;

    @Temporal (TemporalType.DATE)
        private Date datenaissance;

    public client() { }
public Client(String nom, String prenom, Date datedenaissance,
  String email, String password, int etat, String code) {
    super(email, password, etat, code);
```

PARTIE A : Couche Accès aux Données

1) b- Entité : Employer

```
package ma.projet.entity;
import javax.persistence.Entity;
@Entity
public class Employer extends User {
 public Employer() {
   }
}
```

1) c- Entité : User

```
package ma.projet.entity;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.Inheritance;
import javax.persistence.InheritanceType;
@Entity
@Inheritance(strategy = InheritanceType.SINGLE_TABLE)
public class User {
    @Id
   @GeneratedValue
   protected int id;
   @Column(unique = true)
   protected String email;
   protected String password;
   protected int etat;
   protected String code;
```

```
    public User() {
    }

    public User(String email, String password, int etat, String code) {
        this.email = email;
        this.password = password;
        this.etat = etat;
        this.code = code;
    }
    public int getId() {
        return id;
    }
   public void setId(int id) {
        this.id = id;
    }
    public String getEmail() {
        return email;
    }
  public void setEmail(String email) {
        this.email = email;
    }
public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public int getEtat() {
        return etat;
    }

    public void setEtat(int etat) {
```

```
        this.etat = etat;
    }

    public String getCode() {
        return code;
    }

    public void setCode(String code) {
        this.code = code;
    }
}
```

2-Créer le fichier de Configuration HIBERNATE

```
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration
DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
  <session-factory>
    <property
name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
    <property
name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
    <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/sessiontp?zeroDateTi
meBehavior=convertToNull</property>
    <property name="hibernate.connection.username">root</property>
    <property name="hibernate.hbm2ddl.auto">update</property>
    <property name="hibernate.show_sql">true</property>
    <property name="hibernate.format_sql">true</property>
    <mapping class="ma.projet.entity.User"/>
    <mapping class="ma.projet.entity.Client"/>
    <mapping class="ma.projet.entity.Employer"/>
  </session-factory>
</hibernate-configuration>
```

```
    }
}
```

3) Création du Fichier HibernateUtil

```
package ma.projet.util;

import org.hibernate.cfg.AnnotationConfiguration;
import org.hibernate.SessionFactory;

public class HibernateUtil {

   private static final SessionFactory sessionFactory;

   static {
      try {
         // Create the SessionFactory from standard (hibernate.cfg.xml)
         // config file.
         sessionFactory = new
AnnotationConfiguration().configure().buildSessionFactory();
      } catch (Throwable ex) {
         // Log the exception.
         System.err.println("Initial SessionFactory creation failed." + ex);
         throw new ExceptionInInitializerError(ex);
      }
   }

   public static SessionFactory getSessionFactory() {
      return sessionFactory; }}
```

4) Création Interface IDao :

```
package IDao;


import java.util.List;
public interface IDao<T> {


   boolean create(T o);
   boolean delete(T o);
   boolean update(T o);
   T getById(int id);
   List<T> getAll();
}
```

PARTIE B : Couche Service

1) Création des Classes :

a- ClientService

```
package ma.projet.service;


import java.util.List;
import ma.projet.util.HibernateUtil;
import ma.projet.entity.Client;
import org.hibernate.Session;


public class ClientService implements IDao.IDao<Client> {


   @Override
   public boolean create(Client o) {
      Session session = HibernateUtil.getSessionFactory().openSession();
```

```
    session.beginTransaction();
    session.save(o);
    session.getTransaction().commit();
    return true;
}

@Override
public boolean delete(Client o) {
   Session session = HibernateUtil.getSessionFactory().openSession();
    session.beginTransaction();
    session.delete(o);
    session.getTransaction().commit();
    return true;
}

@Override
public boolean update(Client o) {
     Session session = HibernateUtil.getSessionFactory().openSession();
    session.beginTransaction();
    session.update(o);
    session.getTransaction().commit();
    return true;
}

@Override
public Client getById(int id) {
    Client client = null;
    Session session = HibernateUtil.getSessionFactory().openSession();
    session.beginTransaction();
    client = (Client) session.get(Client.class, id);
    session.getTransaction().commit();
    return client;
}
```

```
    @Override
    public List<Client> getAll() {
        List<Client> clients = null;
        Session session = HibernateUtil.getSessionFactory().openSession();
        session.beginTransaction();
        clients = session.createQuery("from Client").list();
        session.getTransaction().commit();
        return clients;
    }

    public Client getByEmail(String email) {
        Client c = null;
        Session session = HibernateUtil.getSessionFactory().openSession();
        session.beginTransaction();
        c = (Client) session.createQuery("from Client where email = ?").setParameter(0,
email).uniqueResult();
        session.getTransaction().commit();
        return c;
    }

}
```

b- EmployeService

```
package ma.projet.service;

import java.util.List;
import ma.projet.util.HibernateUtil;
import ma.projet.entity.Employer;
import org.hibernate.Session;

public class EmployeService implements IDao.IDao<Employer> {

    @Override
```

```
public boolean create(Employer o) {
    Session session = HibernateUtil.getSessionFactory().openSession();
    session.beginTransaction();
    session.save(o);
    session.getTransaction().commit();
    return true;
}

@Override
public boolean delete(Employer o) {
    Session session = HibernateUtil.getSessionFactory().openSession();
    session.beginTransaction();
    session.update(o);
    session.getTransaction().commit();
    return true;
}

@Override
public boolean update(Employer o) {
    Session session = HibernateUtil.getSessionFactory().openSession();
    session.beginTransaction();
    session.delete(o);
    session.getTransaction().commit();
    return true;
}

@Override
public Employer getById(int id) {
    Employer employer = null;
    Session session = HibernateUtil.getSessionFactory().openSession();
    session.beginTransaction();
    employer = (Employer) session.get(Employer.class, id);
    session.getTransaction().commit();
    return employer;
```

```
    }

    @Override
    public List<Employer> getAll() {
        List<Employer> employes = null;
        Session session = HibernateUtil.getSessionFactory().openSession();
        session.beginTransaction();
        employes = session.createQuery("from Employer").list();
        session.getTransaction().commit();
        return employes;
    }

}
```

2)Classe de Test !

```
package ma.projet.test;

import ma.projet.entity.Employer;
import ma.projet.service.ServiceEmployer;
import ma.projet.util.HibernateUtil;

public class Test {
    public static void main(String[ ] args) {
        HibernateUtil.getSessionFactory().openSession();
        } }
```

PARTIE C : Couche Présentation

1) Création Page D'inscription d'un Client :

```
package ma.projet.controlleur;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.Date;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import ma.projet.entity.Client;
import ma.projet.service.ClientService;
import ma.projet.service.sendMail;
import ma.projet.util.Util;

@WebServlet(name = "Inscription", urlPatterns = {"/Inscription"})
public class Inscription extends HttpServlet {

   protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
           throws ServletException, IOException {
       String nom = request.getParameter("nom");
       String prenom = request.getParameter("prenom");
       String email = request.getParameter("email");
       String date = request.getParameter("date").replace("-", "/");
       String password = Util.md5(request.getParameter("password"));

       ClientService cs = new ClientService();
       cs.create(new Client(nom, prenom, new Date(date), email, password,0,null));
```

```
        response.sendRedirect("Authentification.jsp?email="+email);
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the +
sign on the left to edit the code.">
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

2)Création Page D'authentification:

```
package ma.projet.controlleur;

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import ma.projet.entity.Client;
import ma.projet.service.ClientService;
import ma.projet.util.Util;

@WebServlet(name = "Authentification", urlPatterns = {"/Authentification"})
public class Authentification extends HttpServlet {

   protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
         throws ServletException, IOException {
      String email = request.getParameter("email");
      String passworde = request.getParameter("password");
      ClientService cl = new ClientService();
      Client c = cl.getByEmail(email);

      if (c != null) {
         if (c.getPassword().equals(Util.md5(passworde))) {
            HttpSession session = request.getSession();
            session.setAttribute("client", c);
            c.setEtat(1);
            cl.update(c);
            response.sendRedirect("welcome.jsp");
         } else {
            response.sendRedirect("Authentification.jsp?msg=mot de passe incorrect");
         }
      } else {
         response.sendRedirect("Authentification.jsp?msg=Email introvable");
      }
   }
```

3) Maitrisez Encore Plus les Controlleurs :

a) Verifier

```
package ma.projet.controlleur;

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import ma.projet.entity.Client;
import ma.projet.service.ClientService;

@WebServlet(name = "Verfier", urlPatterns = {"/Verfier"})
public class Verfier extends HttpServlet {

   protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
         throws ServletException, IOException {
      response.setContentType("text/html;charset=UTF-8");
      int code = Integer.parseInt(request.getParameter("code"));
      ClientService cl = new ClientService ();
      HttpSession session = request.getSession();
      Client c = (Client) session.getAttribute("client");
      if (c != null) {
         if (Integer.parseInt(c.getCode())==code) {
            response.sendRedirect("updatemotdepasse.jsp");
         } else {
            response.sendRedirect("virfier.jsp?msg= le code est incorrect Zutte!! ");
         }
      } else {
```

```
            response.sendRedirect("virfier.jsp?msg= session vide!! ");
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the +
sign on the left to edit the code.">
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }


    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

b)update

```
package ma.projet.controlleur;

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
```

```
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import ma.projet.entity.Client;
import ma.projet.service.ClientService;
import ma.projet.util.Util;

@WebServlet(name = "updatemotdepasse", urlPatterns = {"/updatemotdepasse"})
public class updatemotdepasse extends HttpServlet {
protected void processRequest(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
    String password=request.getParameter("password");
    String passwordcnf=request.getParameter("passwordcnf");
    ClientService cl = new ClientService();
    if(password.equals(passwordcnf)){
     HttpSession session = request.getSession();
     Client c = (Client) session.getAttribute("client");
     c.setPassword(Util.md5(password));
     cl.update(c);
      response.sendRedirect("Authentification.jsp?email="+c.getEmail());
    }
    else{
    response.sendRedirect("updatemotdepasse.jsp?email=mot de passe incorrect");
    }
  }

  // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the +
sign on the left to edit the code.">

  @Override
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
```

```
    }


    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

Fait Par Joël_yk .