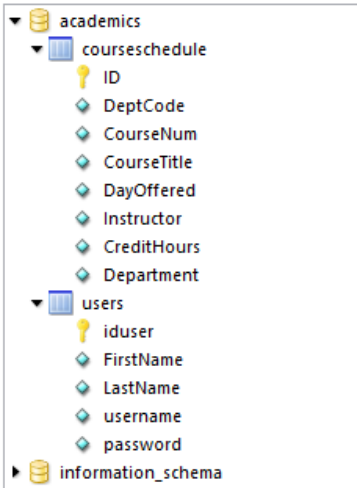
TP 2 des architectures logicielles
Séance 2 : Architecture n-tiers JEE Basic

1 Web dynamique MVC 1 « Servlet, JSP et javaBean, Service DAO » n-tiers basic avec connectivité JDBC

1. Préparation de la base de données : Utiliser MySQL Query Browser pour ajouter une nouvelle table users (voir la figure ci-après), ces attributs seront utilisés pour l'ajout d'une couche transverse de sécurité.



2 Dans cette application, l'utilisateur devra entrer son nom d'utilisateur et mot de passe. Tout d'abord, nous avons besoin d'une JSP qui demande à l'utilisateur d'entrer son nom d'utilisateur et mot de passe dans les champs correspondants.

   a. Créer un nouveau projet Dynamic Web Project : WebnTiers
   b. Créer un nouveau fichier LoginPage.jsp.
   c. Placer le code suivant :

invalidLogin.jsp | LoginPage.jsp | LoginServlet.java | UserBean.java | LoginServletController.java

```jsp
<%@ page language="java"
    contentType="text/html; charset=windows-1256"
    pageEncoding="windows-1256"
%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
                                <meta http-equiv="Content-Type" content="text/html; charset=windows-1256">
        <title>Login Page</title>
    </head>

    <body>
        <form action="LoginServlet">

            Please enter your username
            <input type="text" name="un"/><br>

            Please enter your password
            <input type="text" name="pw"/>

            <input type="submit" value="submit">

        </form>
    </body>
</html>

```

3. Créer le LoginServletController
   La servlet instancie un Bean qui est de type "UserBean", et appelle ensuite la DAO nommé "UserDAO".

   Notre UserBean est une classe représentant la table de l'utilisateur dans notre base de données (où chaque colonne dans la table users a une variable d'instance correspondante avec un setter et un getter).

   Le DAO, comme dit précédemment, contient des méthodes nécessaires pour communiquer avec la source de données. Dans notre exemple, la seule méthode nécessaire est la méthode login() qui vérifie si le nom d'utilisateur et mot de passe entrés par l'utilisateur sont valides ou non.

   a. Dans le dossier "src", créer un nouveau package "ma.ac.uir.controllers"
   b. Dans le ce "Package", créer une nouvelle Servlet "LoginServletController"
   c. Placer ce code

```java
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

/**
 * Servlet implementation class LoginServlet
 */
public class LoginServlet extends HttpServlet {


/**
     *
     */
    private static final long serialVersionUID = 1L;

public void doGet(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, java.io.IOException {

try
{

     UserBean user = new UserBean();
     user.setUserName(request.getParameter("un"));
     user.setPassword(request.getParameter("pw"));

     user = UserDAO.login(user);

    if (user.isValid())
    {

         HttpSession session = request.getSession(true);
         session.setAttribute("currentSessionUser",user);
         response.sendRedirect("userLogged.jsp"); //logged-in page
    }

    else
         response.sendRedirect("invalidLogin.jsp"); //error page
}


catch (Throwable theException)
{
    System.out.println(theException);
}
      }
}
```

4. Implémenter le Bean « UserBean » dans la package ma.ac.uir.models

```java
public class UserBean {

	private String username;
	private String password;
	private String firstName;
	private String lastName;
	public boolean valid;


	public String getFirstName() {
	   return firstName;
	}

	public void setFirstName(String newFirstName) {
	   firstName = newFirstName;
	}


	public String getLastName() {
	   return lastName;
	      }

	public void setLastName(String newLastName) {
	   lastName = newLastName;
	      }


	public String getPassword() {
	   return password;
	}

	public void setPassword(String newPassword) {
	   password = newPassword;
	}
	public String getUsername() {
	   return username;
	      }

	public void setUserName(String newUsername) {
	   username = newUsername;
	      }


	public boolean isValid() {
	   return valid;
	}

	public void setValid(boolean newValid) {
	   valid = newValid;
	}
}
```

5. Nous allons maintenant créer le service DAO correspondant à la classe UserBean. Pour cela, il faut ajouter :
    a. un nouveau package ma.ac.uir.dao
    b. une classe métier UserDAO

```java
import java.text.*;
import java.util.*;
import java.sql.*;

   public class UserDAO
   {
      static Connection currentCon = null;
      static ResultSet rs = null;



      public static UserBean login(UserBean bean) {

         //preparing some objects for connection
         Statement stmt = null;

         String username = bean.getUsername();
         String password = bean.getPassword();

         String searchQuery =
               "select * from users where username='"
                        + username
                        + "' AND password='"
                        + password
                        + "'";

      // "System.out.println" prints in the console; Normally used to trace the process
      System.out.println("Your user name is " + username);
      System.out.println("Your password is " + password);
      System.out.println("Query: "+searchQuery);

      try
      {
         //connect to DB
         currentCon = ConnectionManager.getConnection();
         stmt=currentCon.createStatement();
         rs = stmt.executeQuery(searchQuery);
         boolean more = rs.next();

         // if user does not exist set the isValid variable to false
         if (!more)
         {
            System.out.println("Sorry, you are not a registered user! Please sign up first");
            bean.setValid(false);
         }

         //if user exists set the isValid variable to true
         else if (more)
         {
            String firstName = rs.getString("FirstName");
            String lastName = rs.getString("LastName");

            System.out.println("Welcome " + firstName);
            bean.setFirstName(firstName);
            bean.setLastName(lastName);
            bean.setValid(true);
         }
      }

      catch (Exception ex)
      {
         System.out.println("Log In failed: An Exception has occurred! " + ex);
      }
```

```
68        //some exception handling
69        finally
70        {
71           if (rs != null)    {
72              try {
73                 rs.close();
74              } catch (Exception e) {}
75                 rs = null;
76              }
77
78           if (stmt != null) {
79              try {
80                 stmt.close();
81              } catch (Exception e) {}
82                 stmt = null;
83              }
84
85           if (currentCon != null) {
86              try {
87                 currentCon.close();
88              } catch (Exception e) {
89              }
90
91              currentCon = null;
92           }
93        }
94
95  return bean;
96
97        }
98     }
99
```

6. Ajouter une nouvelle page jsp « userLogged.jsp » permettant d'afficher un message pour la réussite de l'authentification.

invalidLogin.jsp | userLogged.jsp | LoginPage.jsp | LoginServlet.java | UserBean.java | LoginServlet

```
1  <%@ page language="java"
2           contentType="text/html; charset=windows-1256"
3           pageEncoding="windows-1256"
4           import="ExamplePackage.UserBean"
5     %>
6
7     <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
8     "http://www.w3.org/TR/html4/loose.dtd">
9
10    <html>
11
12       <head>
13          <meta http-equiv="Content-Type"
14             content="text/html; charset=windows-1256">
15          <title>   User Logged Successfully   </title>
16       </head>
17
18       <body>
19
20          <center>
21             <% UserBean currentUser = (UserBean) (session.getAttribute("currentSessionUser"));%>
22
23             Welcome <%= currentUser.getFirstName() + " " + currentUser.getLastName() %>
24          </center>
25
26       </body>
27
28    </html>
```

7. Ajouter une nouvelle page jsp « invalidUser » permettant d'afficher l'echec de l'authentification.

invalidLogin.jsp | LoginPage.jsp | LoginServlet.java | UserBean.java

```jsp
<%@ page language="java"
    contentType="text/html; charset=windows-1256"
    pageEncoding="windows-1256"
%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
    "http://www.w3.org/TR/html4/loose.dtd">

<html>

    <head>
        <meta http-equiv="Content-Type"
            content="text/html; charset=windows-1256">
        <title>Invalid Login</title>
    </head>

    <body>
        <center>
            Sorry, you are not a registered user! Please sign up first
        </center>
    </body>

</html>
```

8. Déployer l'application sur le serveur Tomcat pour voir le résultat d'exécution.
9. En adoptant le même modèle de conception MVC, vous devez refaire l'application du TP 1 (Architecture 3-tiers) avec une vue n-tiers.
10. Déployer la nouvelle application sur le serveur Tomcat pour voir le résultat d'exécution. Toutes les ressources de l'application doivent être sécurisées.