TP4 : JEE

Tags	
status	sent

MVC

> L’objectif de ce TP est de se familiariser avec le model MVC, le découpage de l’application en 3 couches :
> Web, Métier et Donnée, et l’implémentation du model MVC dans la couche Web.

Exercice 1 : Création d’une application du calcul de l’IMC (Calcul_IMC)

Couche Metier :

1. Création de l’interface :

IImcMetier.java

```java
package metier;

public interface IImcMetier {
    public double calculerImc(int poids, double taille);
}
```

2. Implémentation de l’interface :

```java
package metier;

public class ImcMetierImpl implements IImcMetier {
    @Override
    public double calculerImc(int poids, double taille) {
        double imc = poids/(taille*taille);
        // TODO Auto-generated method stub
        return imc;
    }
}
```

3. Test et validation du traitement métier

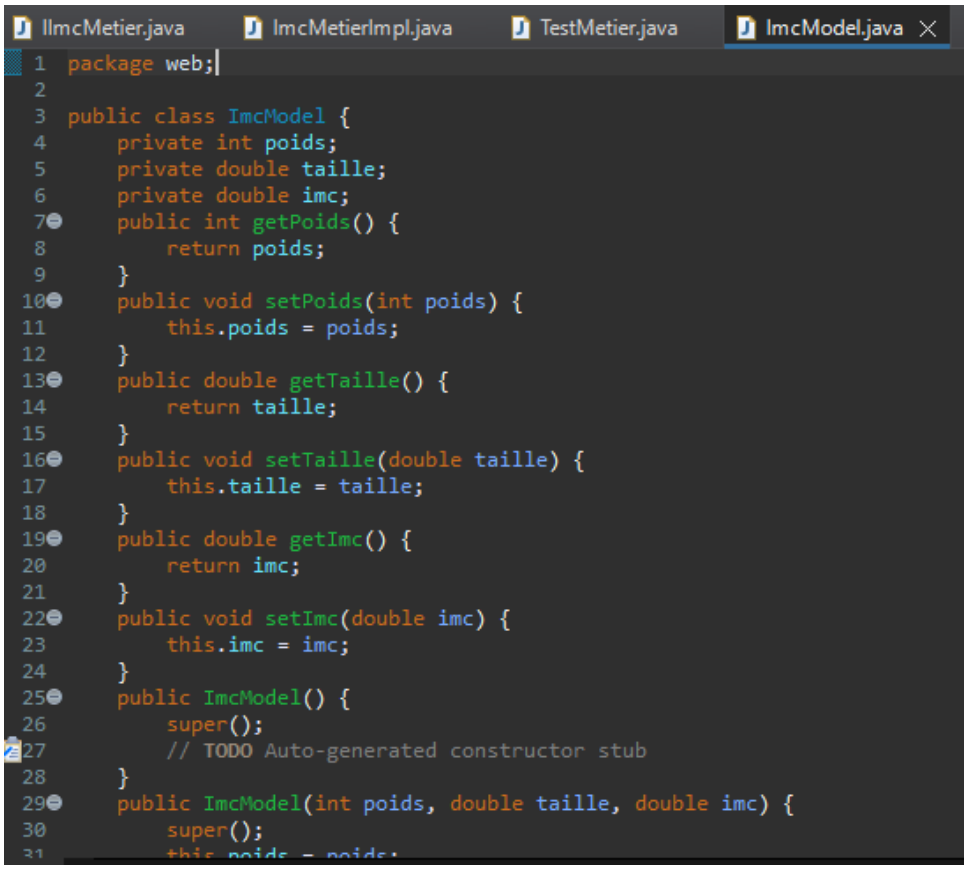
```java
package metier;

public class TestMetier {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        ImcMetierImpl metier=new ImcMetierImpl();
        double taille=1.2;
        int poids=65;
        double imc=metier.calculerImc(poids, taille);
        System.out.println(imc);
    }
}
```

```
Problems  Servers  Terminal  Data Source Explorer  Properties  Console
<terminated> TestMetier [Java Application] D:\Users\ibtissam\Documents\eclipse-jee-2023-09-R-win32-x86_6
45.13888888888889
```

Couche Web :

1. Création de model:



2. Création du contrôleur

```java
package web;
import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import metier.ImcMetierImpl;

public class ControleurServlet extends HttpServlet{
    private ImcMetierImpl metier;
    @Override
    public void init() throws ServletException {
        metier=new ImcMetierImpl();
    }
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse rep) throws
        req.setAttribute("ImcModel", new ImcModel());
        req.getRequestDispatcher("VueImc.jsp").forward(req,rep);
    }
    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse rep) throw
        /*lire les données de la req*/
        int poids = Integer.parseInt(req.getParameter("poids"));
        double taille = Double.parseDouble(req.getParameter("taille"));

        /*stocker dans le model*/
        ImcModel model= new ImcModel();
        model.setPoids(poids);
        model.setTaille(taille);
```

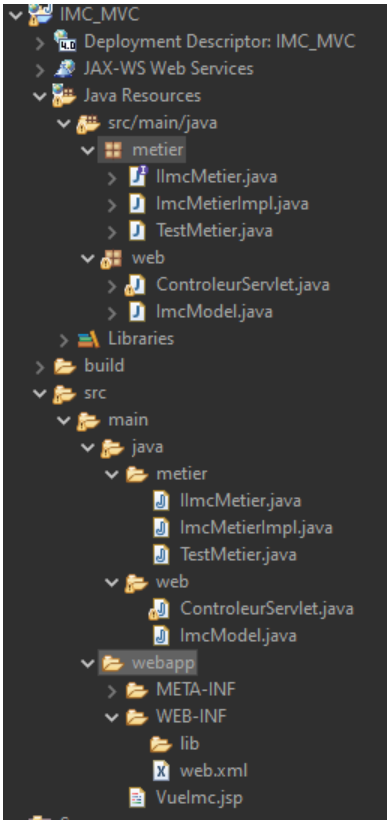
3. Edition Descripteur de déploiement de la servlet : web.xml

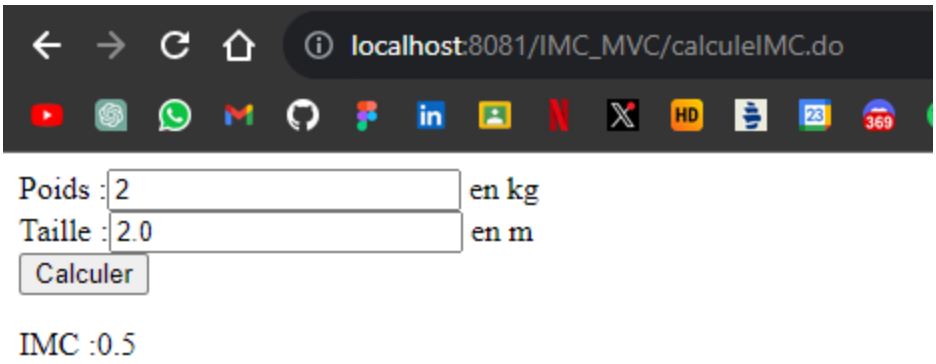
```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xml
  <display-name>IMC_MVC</display-name>
  <servlet>
       <servlet-name>cs</servlet-name>
       <servlet-class>web.ControleurServlet</servlet-class>
   </servlet>
   <servlet-mapping>
       <servlet-name>cs</servlet-name>
       <url-pattern>*.do</url-pattern>
   </servlet-mapping>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.jsp</welcome-file>
    <welcome-file>default.htm</welcome-file>
  </welcome-file-list>
</web-app>
```

4. Création de la vue :

ImcModel.java | ControleurServlet.java | web.xml | VueImc.jsp

```jsp
<%@page import="web.ImcModel"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%
ImcModel model = (ImcModel) request.getAttribute("ImcModel");
%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>IMC</title>
</head>
<body>
    <div>
    <form action="calculeIMC.do" method="post">
        Poids :<input type="text" name="poids" value="<%=model.getPoids()%>"> en
        Taille :<input type="text" name="taille" value="<%=model.getTaille()%>">
        <input type="submit" value="Calculer">
    </form>
    </div>
    <p></p>
    <div>
        IMC :<%=model.getImc()%>
    </div>
</body>
</html>
```





Exercice 2 : Création d'une application de gestion des notes (Gestion_Note)

> Nous souhaitons créer une application web J2EE qui permet de gérer les notes des étudiants en respectant le model MVC. L'application doit permettre de :
> − Saisir le numéro d'inscription d'un étudiant,
> − En validant le formulaire, afficher les notes de cet étudiant en affichant la moyenne de ses notes.

A. Couche Donnée :

1. Créer la base de données Scolarité

2. Créer la table Notes :

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	Id_Note	int			No	None			Change Drop More
☐	2	Num_Ins	varchar(20)	utf8mb4_0900_ai_ci		No	None			Change Drop More
☐	3	Matiere	varchar(20)	utf8mb4_0900_ai_ci		No	None			Change Drop More
☐	4	Note	float			No	None			Change Drop More

	Id_Note	Num_Ins	Matiere	Note
Edit Copy Delete	12	123	JEE	18
Edit Copy Delete	13	123455	English	16

B. Couche Métier :

1. Créer le package mod.sco

2. Créer le model : Classe Note.java

Note.java

```java
package mod.sco;

public class Note {
    private int id_note;
    private String num_Ins;
    private String matiere;
    private float note;
    public int getId_note() {
        return id_note;
    }
    public void setId_note(int id_note) {
        this.id_note = id_note;
    }
    public String getNum_Ins() {
        return num_Ins;
    }
    public void setNum_Ins(String num_Ins) {
        this.num_Ins = num_Ins;
    }
    public String getMetiere() {
        return matiere;
    }
    public Note(int id_note, String num_Ins, String metiere, float note) {
        super();
        this.id_note = id_note;
        this.num_Ins = num_Ins;
        this.matiere = metiere;
        this.note = note;
    }
    public void setMetiere(String metiere) {
        this.matiere = metiere;
    }
    public float getNote() {
        return note;
    }
    public void setNote(float note) {
        this.note = note;
    }
}

```

3. Créer la classe DatabaseConn

Note.java DatabaseConn.java

```java
package mod.sco;

import java.sql.Connection;
import java.sql.DriverManager;

public class DatabaseConn {
    private final String URL = "jdbc:mysql://localhost:3306/scolarité";
    private final String USERNAME = "root";
    private final String PASSWORD = "";
    private Connection conn;
    public DatabaseConn() {
        try {
            conn = DriverManager.getConnection(URL, USERNAME, PASSWORD);
        }catch(Exception e){
            System.out.println("Error While connecting to db");
            e.printStackTrace();
        }
    }
    public Connection getConn() {
        return conn;
    }

```

4.Créer la classe Scolarite.java . Cette classe dispose d'une seule fonction Vector getNotes(String num_Ins) qui permet de retourner les notes d'un étudiant dont le num d'inscription est passé en paramètre.

Note.java DatabaseConn.java Scolarite.java

```java
package mod.sco;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

import java.util.Vector;

public class Scolarite {
    public Vector<Note> getNotes(String num_Ins){
        Vector<Note> vectNotes = new Vector<Note>();
        try {
            DatabaseConn DBConn = new DatabaseConn();
            Connection conn = DBConn.getConn();
            String sql = "SELECT * FROM notes WHERE Num_Ins = ?";
            PreparedStatement preparedStatement = conn.prepareStatement(sql);
            preparedStatement.setString(1, num_Ins);
            ResultSet resultSet = preparedStatement.executeQuery();
            while (resultSet.next()) {
                int Id_note = resultSet.getInt("Id_Note");
                String Num_Ins = resultSet.getString("Num_Ins");
                String Matiere = resultSet.getString("Matiere");
                float note = resultSet.getFloat("Note");
                Note noteObj = new Note(Id_note,Num_Ins,Matiere,note);
                vectNotes.add(noteObj);
            }

        }catch(Exception e) {
            System.out.println("error while getting data from database");
            e.fillInStackTrace();
        }
        return vectNotes;
```

5.Créer une classe TestMetier. Java qui permet de tester la validité de couche métier

Note.java DatabaseConn.java Scolarite.java TestMetier.java

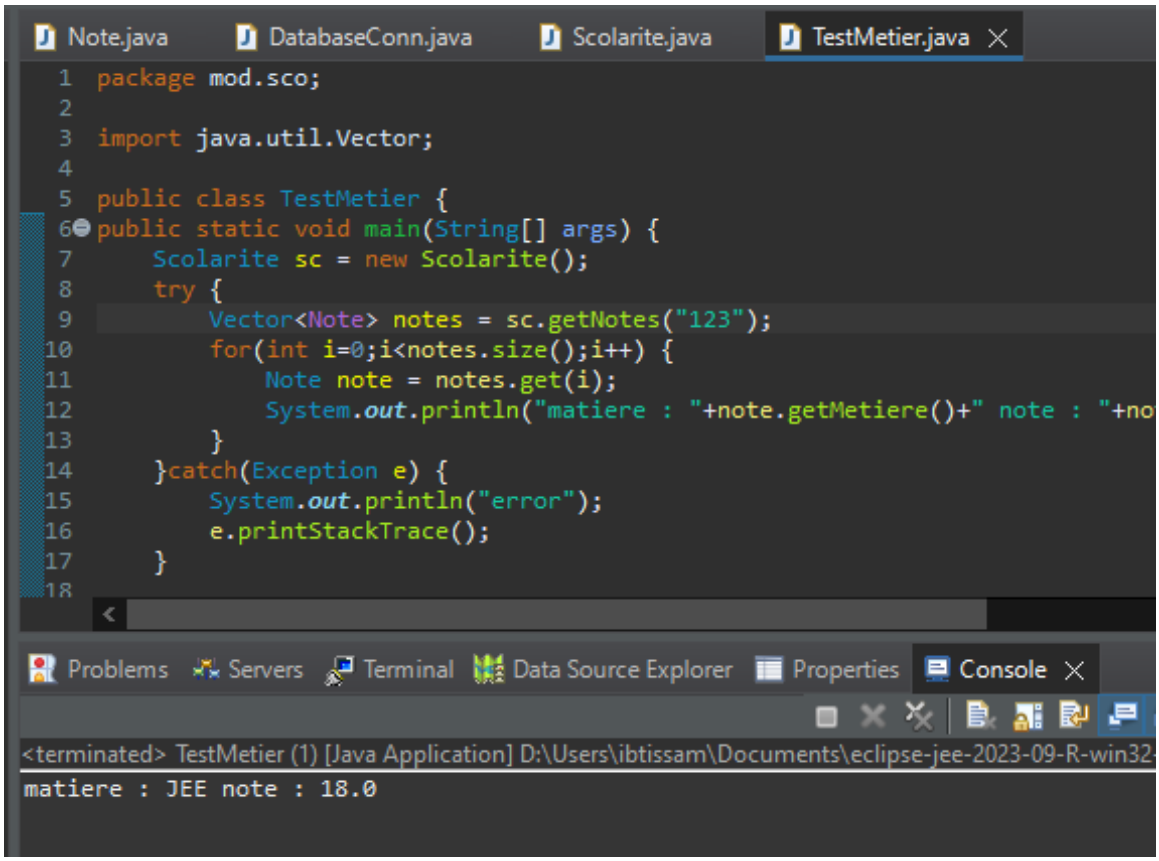
```java
package mod.sco;

import java.util.Vector;

public class TestMetier {
public static void main(String[] args) {
    Scolarite sc = new Scolarite();
    try {
        Vector<Note> notes = sc.getNotes("12");
        for(int i=0;i<notes.size();i++) {
            Note note = notes.get(i);
            System.out.println("matiere : "+note.getMetiere()+" note : "+note.getNote()+"\n");
        }
    }catch(Exception e) {
        System.out.println("error");
        e.printStackTrace();
    }


}
}

```



C. Couche Web :

1. Créer la package form.beans;

2.Créer la classe NoteForm.java qui permet de stocker le N° d'inscription saisie par l'utilisateur et les notes récupérer de la BD.

DatabaseConn... Scolarite.java TestMetier.java *NoteForm.java

```java
package form.beans;
import java.util.Vector;
import mod.sco.Note;

public class NoteForm {
    private String num_Ins;
    private Vector<Note> notes;
    public String getNum_Ins() {
        return num_Ins;
    }
    public void setNum_Ins(String num_Ins) {
        this.num_Ins = num_Ins;
    }
    public Vector<Note> getLesNotes() {
        return notes;
    }
    public void setLesNotes(Vector<Note> notes) {
        this.notes = notes;
    }

}
```

3. Créer le contrôleur ControleurServlet.java

4. Modifier la methode doPost()

```java
package mod;

import java.io.IOException;

public class ControleurServlet extends HttpServlet{
    /**
     *
     */
    private static final long serialVersionUID = 1L;

    @Override
    public void doPost(HttpServletRequest req, HttpServletResponse rep)
            throws ServletException, IOException{
        Scolarite sco = new Scolarite();
        NoteForm nf = new NoteForm();

        String num_Ins = req.getParameter("num_Ins");
        System.out.println(num_Ins);
        nf.setNum_Ins(num_Ins);
        Vector<Note> Notes = sco.getNotes(num_Ins);
        nf.setLesNotes(Notes);

        HttpSession session = req.getSession();
        session.setAttribute("nf", nf);
        rep.sendRedirect("Notes.jsp");
    }

}
```

5. Editer le fichierweb.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="htt
  <display-name>GestionNotes</display-name>
<servlet>
    <servlet-name>cnt</servlet-name>
    <servlet-class>mod.ControleurServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>cnt</servlet-name>
    <url-pattern>*.php</url-pattern>
</servlet-mapping>
</web-app>
```

6. Créer la vue Notes.jsp et ajouter les scriptlets nécessaires pour calculer et afficher la moyenne

Note.java | ControleurS... | web.xml | Notes.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@ page import="java.util.Vector" %>
<%@ page import="mod.sco.Note" %>
<%@ page import="form.beans.NoteForm" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<%
NoteForm nf = (NoteForm)session.getAttribute("nf");
Vector<Note> dataVect=null;
if(nf!=null){
    dataVect = nf.getLesNotes() ;
}
%>
<form action="fetchNote.php" method = "post">
Num Inscription : <input type="text" name="num_Ins" >
<input type="submit" value="OK">
</form>
<table>
    <tr>
        <th>Matière</th>
        <th>Note</th>
    </tr>
    <%
    if(dataVect!=null){
    for (Note note : dataVect){
    %>
        <tr>
            <td>${note.matiere}</td>
            <td>${note.note}</td>
        </tr>
    <%}}%>
</table>
</body>
</html>
```