TP4 : JEE

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.

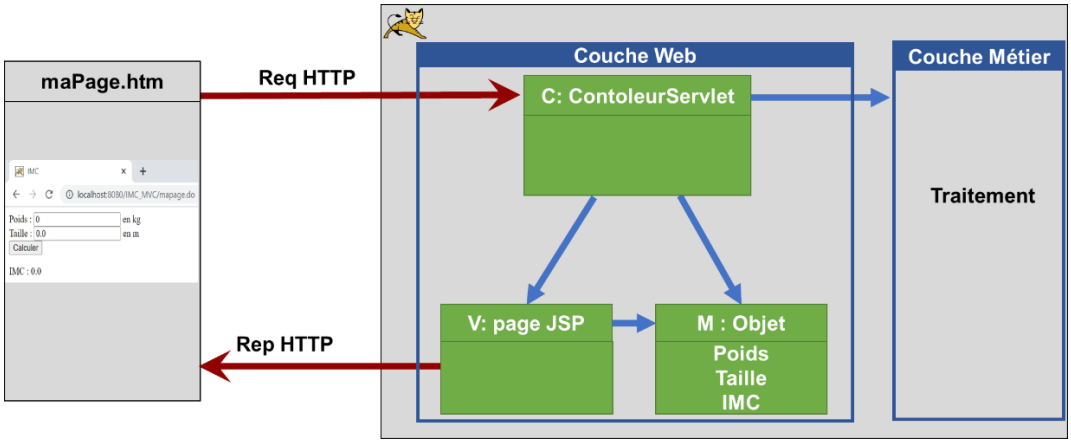
Exercice 1 : Création d'une application du calcul de l'IMC (Calcul_IMC)

Nous souhaitons de créer une application JEE qui permet de calculer l'IMC d'un utilisateur en se basant sur le model MVC.

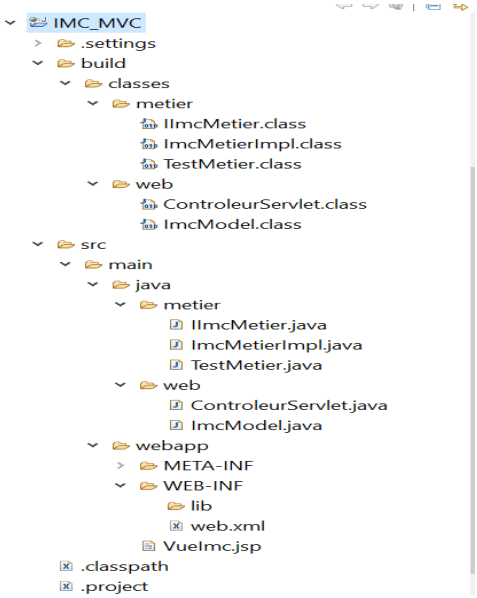
L'IMC est calculé sur la base de la taille en m et du poids en kg.

La formule de calcul est comme suit : imc= poids / taille².

L'architecture de l'application est illustrée sur le schéma suivant :



La structure globale du projet est comme suit :



A. Couche Metier :

1. Création de l'interface :

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

2. Implémentation de l'interface :

```
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

```
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);
	}
}
```

B. Couche Web

1. Création de model:

```
package web;

      public class ImcModel {
	private int poids;
	private double taille;
	private double imc;

	public int getPoids() {
		return poids;
	}
	public void setPoids(int poids) {
		this.poids = poids;
	}
	public double getTaille() {
		return taille;
	}
	public void setTaille(double taille) {
		this.taille = taille;
	}
	public double getImc() {
		return imc;
	}
	public void setImc(double imc) {
		this.imc = imc;
	}
	public ImcModel() {
		super();
		// TODO Auto-generated constructor stub
	}
	public ImcModel(int poids, double taille, double imc) {
		super();
```

```
		this.poids = poids;
		this.taille = taille;
		this.imc = imc;
	}
}
```

2. Création du contrôleur

```
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 ServletException, IOException {
		req.setAttribute("ImcModel", new ImcModel());
		req.getRequestDispatcher("VueImc.jsp").forward(req,rep);
   }

	@Override
	protected void doPost(HttpServletRequest req, HttpServletResponse rep)
			throws ServletException, IOException {

		/*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);


		/*faire le trait (appel metier)*/
		double res=metier.calculerImc(poids, taille);


		/*stocker le res dans le model */
		model.setImc(res);

		/*stocker le model dans l'objet req*/

		req.setAttribute("ImcModel", model);

		/*faire le forward vers la vue jsp*/

		req.getRequestDispatcher("VueImc.jsp").forward(req, rep);
	}
}
```

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

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
id="WebApp_ID" version="4.0">

  <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>
```

4. Création de la vue :

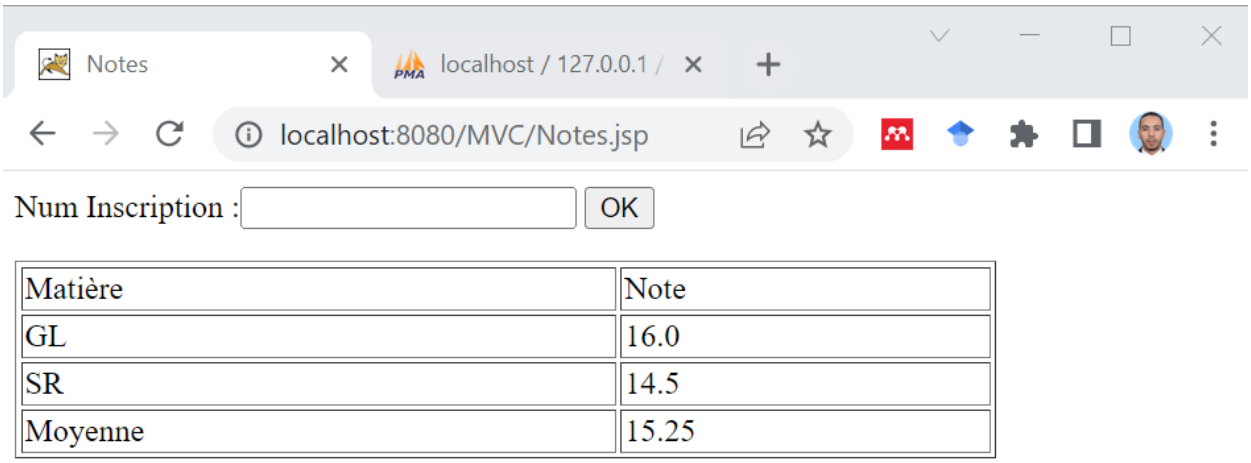
```
<%@page import="web.ImcModel"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
 pageEncoding="ISO-8859-1"%>


<%
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 kg <br>
Taille :<input type="text" name="taille" value="<%=model.getTaille()%>"> en m<br>
  <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)

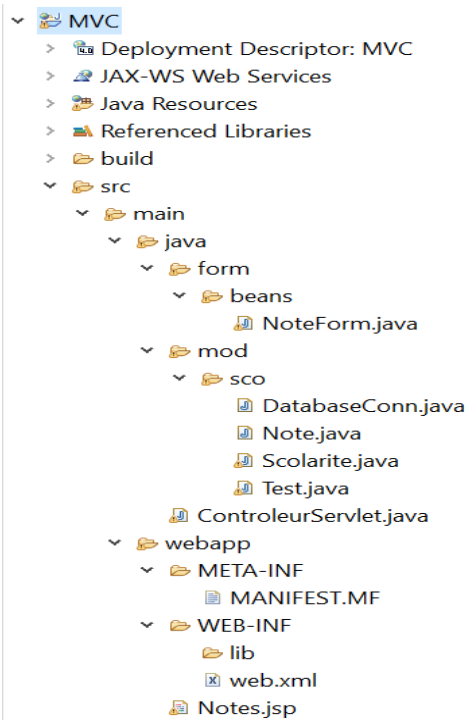
Nous souhaitons créer une application web J2EE qui permet de gérer les notes des étudiants en respectant le model MVC. L'application doit permettre de :

- 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.

La figure suivante présente un aperçu de la vue à réaliser :



La structure globale du projet est comme suit :



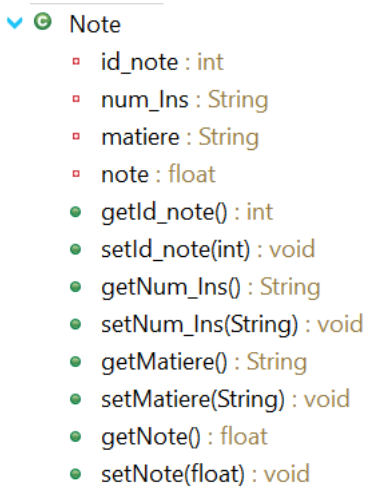
A. Couche Donnée :

1. Créer la base de données Scolarité
2. Créer la table Notes :

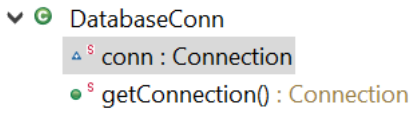
#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra
1	Id_Note	int(11)			Non	*Aucun(e)*		AUTO_INCREMENT
2	Num_Ins	varchar(20)	utf8mb4_general_ci		Non	*Aucun(e)*		
3	Matiere	varchar(20)	utf8mb4_general_ci		Non	*Aucun(e)*		
4	Note	float			Non	*Aucun(e)*		

B. Couche Métier :

1. Créer le package mod.sco
2. Créer le model : Classe Note.java



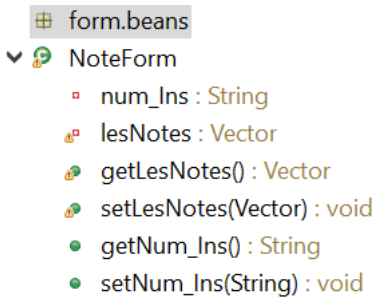
3. Créer la classe DatabaseConn



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

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.
3. Créer le contrôleur ControleurServlet.java
4. Modifier la methode doPost() comme suit :




```
public void doPost(
            HttpServletRequest request,
            HttpServletResponse response)
            throws ServletException, IOException {

      Scolarite sco=new Scolarite();
      NoteForm nf=new NoteForm();

      nf.setNum_Ins(request.getParameter("num_Ins"));
      nf.setLesNotes(sco.getNotes(nf.getNum_Ins()));

      HttpSession session=request.getSession();
      session.setAttribute("nf",nf);
      response.sendRedirect("Notes.jsp");
   }
```

5. Editer le fichierweb.xml

```
<servlet>
      <servlet-name>cnt</servlet-name>
      <servlet-class>ControleurServlet</servlet-class>
</servlet>
<servlet-mapping>
      <servlet-name>cnt</servlet-name>
      <url-pattern>*.php</url-pattern>
  </servlet-mapping>
```

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