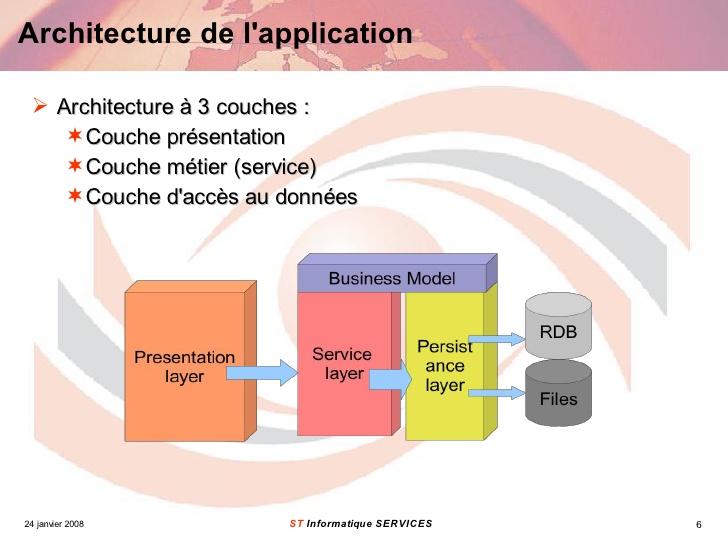
Le modele en couche

Application PLANNING

SLAM 2

Matthieu Planchot



La couche présentation ou interface graphique qui contient le formulaire est responsable de l’affichage des données à l’utilisateur.

Ces données sont obtenues grâce à la couche logique métier. De même, l’ajout de nouvelles données transite par cette couche.

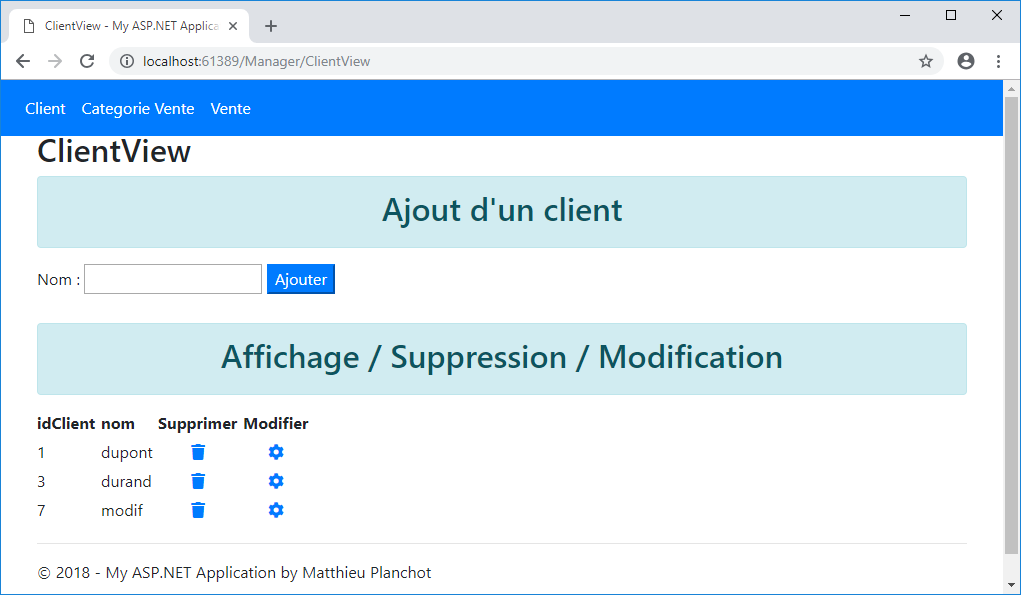
La couche métier est responsable du traitement de ces données. Elle contient les objets métiers qui sont utilisés dans l’application.

La couche d’accès aux données est responsable de la création des objets métiers à partir de la base de données (select) et de sa mise à jour (insert, delete, update) à partir de ses objets métiers. Chaque classe est spécifique d’une table de la base de données.

J’ai mis mon projet sur github à l’adresse suivante :

<https://github.com/konki76/ModeleEnCouche>

# Vues



@{

ViewBag.Title = "ClientView";

}

<h2>ClientView</h2>

<div class="alert alert-info text-center">

<h2>Ajout d'un client</h2>

</div>

<form action="ClientView" method="post">

<label for="nom">Nom :</label>

<input type="text" name="nom" required />

<input type="submit" name="Ajouter" class="btn-primary" value="Ajouter">

</form>

<br />

<div class="alert alert-info text-center">

<h2>Affichage / Suppression / Modification</h2>

</div>

<table>

<tr>

<th>idClient</th>

<th>nom</th>

<th>Supprimer</th>

<th>Modifier</th>

</tr>

@foreach (ModeleEnCouchePlanning.Models.Client client in ViewData["Clients"] as List<ModeleEnCouchePlanning.Models.Client>)

{

<tr>

<td>@client.idClient</td>

<td>@client.nom</td>

<td class="text-center"><a href='ClientView?idClient=@client.idClient'><i class="fas fa-trash"></i></a></td>

<td class="text-center"><a href='ClientModifView?idClient=@client.idClient'><i class="fas fa-cog"></i></a></td>

</tr>

}

</table>

Le foreach récupère chaque objet de la liste ViewData["Clients"] définie dans ManagerController et il utilise l’objet métier Client.

Les vues CategVenteView et Vente ressemble à ClientView.

@{

ViewBag.Title = "CategVenteView";

}

<h2>CategVenteView</h2>

<div class="alert alert-info text-center">

<h2>Ajout d'une catégorie</h2>

</div>

<form action="CategVenteView" method="post">

<label for="libelleCategVente">libelle :</label>

<input type="text" name="libelleCategVente" required />

<input type="submit" name="Ajouter" class="btn-primary" value="Ajouter">

</form>

<br />

<div class="alert alert-info text-center">

<h2>Affichage / Suppression / Modification</h2>

</div>

<table>

<tr>

<th>idCategVente</th>

<th>libelleCategVente</th>

<th>Supprimer</th>

<th>Modifier</th>

</tr>

@foreach (ModeleEnCouchePlanning.Models.CategVente categ in ViewData["CategVentes"] as List<ModeleEnCouchePlanning.Models.CategVente>)

{

<tr>

<td>@categ.idCategVente</td>

<td>@categ.libelleCategVente</td>

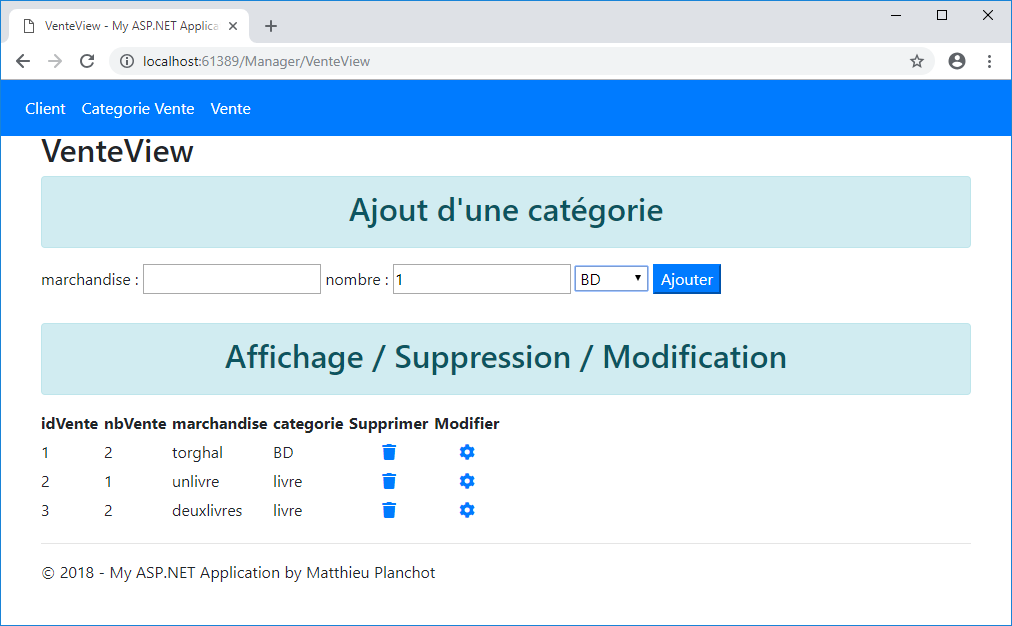
<td class="text-center"><a href='CategVenteView?idClient=@categ.idCategVente'><i class="fas fa-trash"></i></a></td>

<td class="text-center"><a href='CategVenteModifView?idClient=@categ.idCategVente'><i class="fas fa-cog"></i></a></td>

</tr>

}

</table>



@{

ViewBag.Title = "VenteView";

}

<h2>VenteView</h2>

<div class="alert alert-info text-center">

<h2>Ajout d'une catégorie</h2>

</div>

<form action="VenteView" method="post">

<label for="marchandise">marchandise :</label>

<input type="text" name="marchandise" required />

<select name="idCategVente">

@foreach (ModeleEnCouchePlanning.Models.CategVente categ in ViewData["CategVentes"] as List<ModeleEnCouchePlanning.Models.CategVente>)

{

<option value="@categ.idCategVente">@categ.libelleCategVente</option>

}

</select>

<input type="submit" name="Ajouter" class="btn-primary" value="Ajouter">

</form>

<br />

<div class="alert alert-info text-center">

<h2>Affichage / Suppression / Modification</h2>

</div>

<table>

<tr>

<th>idVente</th>

<th>nbVente</th>

<th>marchandise</th>

<th>categorie</th>

<th>Supprimer</th>

<th>Modifier</th>

</tr>

@foreach (ModeleEnCouchePlanning.Models.Vente vente in ViewData["Ventes"] as List<ModeleEnCouchePlanning.Models.Vente>)

{

<tr>

<td>@vente.idVente</td>

<td>@vente.nbVente</td>

<td>@vente.marchandise</td>

<td>@vente.idCategVente</td>

<td class="text-center"><a href='VenteView?idVente=@vente.idVente'><i class="fas fa-trash"></i></a></td>

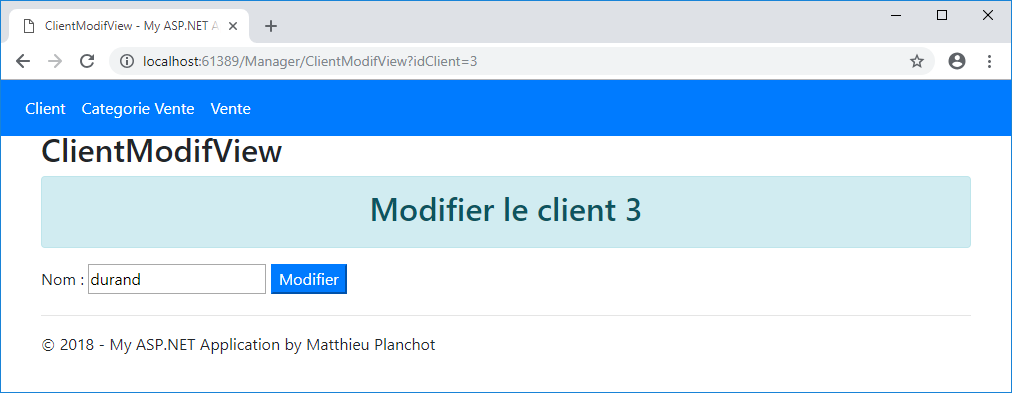
<td class="text-center"><a href='VenteModifView?idVente=@vente.idVente'><i class="fas fa-cog"></i></a></td>

</tr>

}

</table>

# Vues des modifications



@{

ViewBag.Title = "ClientModifView";

}

<h2>ClientModifView</h2>

<div class="alert alert-info text-center">

<h2>Modifier le client @ViewData["idClient"]</h2>

</div>

<form action="ClientView" method="post">

<label for="nom">Nom :</label>

<input type="hidden" name="idClient" value="@ViewData["idClient"]" />

<input type="text" name="nom" value="@ViewData["nomClient"]" required />

<input type="submit" name="Modifier" class="btn-primary" value="Modifier">

</form>

@{

ViewBag.Title = "CategVenteModifView";

}

<h2>CategVenteModifView</h2>

<div class="alert alert-info text-center">

<h2>Modifier la catégorie @ViewData["idCategVente"]</h2>

</div>

<form action="CategVenteView" method="post">

<label for="idCategVente">Libelle :</label>

<input type="hidden" name="idCategVente" value="@ViewData["idCategVente"]" />

<input type="text" name="libelleCategVente" value="@ViewData["libelleCategVente"]" required /><br />

<input type="submit" name="Modifier" class="btn-primary" value="Modifier">

</form>

@{

ViewBag.Title = "VenteModifView";

}

<h2>VenteModifView</h2>

<div class="alert alert-info text-center">

<h2>Modifier la marchandise @ViewData["idVente"]</h2>

</div>

<form action="VenteView" method="post">

<input type="hidden" name="idVente" value="@ViewData["idVente"]" />

<label for="marchandise">Marchandise :</label>

<input type="text" name="marchandise" value="@ViewData["marchandise"]" required />

<label for="nbVente">Nombre de vente : </label>

<input type="text" name="nbVente" value="@ViewData["nbVente"]" required />

<label for="idCategVente">id Catégorie : </label>

<input type="text" name="idCategVente" value="@ViewData["idCategVente"]" required />

<input type="submit" name="Modifier" class="btn-primary" value="Modifier">

</form>

# Bibliothèque de classe

La classe AccesBD permet de se connecter à la base de données, c’est la classe mère des classes ClientDAO, CategVente et VenteDAO.

public class AccesBD

{

private static string connect = "Server=127.0.0.1;Database=planning;Uid=root;Pwd=;";

protected MySqlConnection co = new MySqlConnection(connect);

protected void Initialize()

{

try { co.Open(); }

catch (Exception e) { e.ToString(); }

}

protected void nonQuery(string query)

{

try

{

// Ouverture de la connexion à la base de données

co.Open();

// Exécution d’une instruction en base de données

MySqlCommand cmd = new MySqlCommand(query, co);

cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

Console.WriteLine("Connexion impossible, raisons : " + ex);

}

finally

{

// Fermeture de la base de données

try { co.Close(); }

catch (MySqlException e) { Console.WriteLine(e); }

}

}

}

Chaque classe de la couche d’accès aux donnés hérite de la classe AccesBD et permet la lecture, l’ajout, la modification et la suppression de la table du même nom.

public class ClientDAO : AccesBD

{

public List<Client> listClient()

{

string query = "SELECT idClient, nom FROM client;";

// initialisation de la connexion

Initialize();

// création de la liste à retourner

List<Client> client = new List<Client>();

MySqlCommand cmd = new MySqlCommand(query, co);

//crée un data reader et exécute la commande

MySqlDataReader dataReader = cmd.ExecuteReader();

if (dataReader.HasRows)

{

//lit la liste et stocke les données dans la liste

while (dataReader.Read())

{

Client a = new Client();

a.idClient = dataReader["idClient"] == DBNull.Value ? string.Empty : dataReader["idClient"].ToString();

a.nom = dataReader["nom"] == DBNull.Value ? string.Empty : dataReader["nom"].ToString();

client.Add(a);

}

}

dataReader.Close();

co.Close();

//retourne la liste à afficher

return client;

}

public Client idClient(string idClient)

{

string query = "SELECT idClient, nom FROM client WHERE idClient = "+ idClient;

Initialize();

Client client = new Client();

MySqlCommand cmd = new MySqlCommand(query, co);

MySqlDataReader dataReader = cmd.ExecuteReader();

if (dataReader.HasRows)

{

while (dataReader.Read())

{

client.idClient = dataReader["idClient"] == DBNull.Value ? string.Empty : dataReader["idClient"].ToString();

client.nom = dataReader["nom"] == DBNull.Value ? string.Empty : dataReader["nom"].ToString();

}

}

dataReader.Close();

co.Close();

return client;

}

public void clientDelete(string idClient)

{

string query = "DELETE FROM client WHERE idClient = " + idClient;

nonQuery(query);

}

public void clientInsert(Client client)

{

string query = "INSERT INTO client(nom) VALUES ('" + client.nom + "');";

nonQuery(query);

}

public void clientUpdate(Client client)

{

string query = "UPDATE client SET nom = '" + client.nom + "' WHERE idClient = " + client.idClient;

nonQuery(query);

}

}

public class CategVenteDAO : AccesBD

{

public List<CategVente> listCategVente()

{

string query = "SELECT idCategVente, libelleCategVente FROM categvente;";

// initialisation de la connexion

Initialize();

// création de la liste à retourner

List<CategVente> categVente = new List<CategVente>();

MySqlCommand cmd = new MySqlCommand(query, co);

//crée un data reader et exécute la commande

MySqlDataReader dataReader = cmd.ExecuteReader();

if (dataReader.HasRows)

{

//lit la liste et stocke les données dans la liste

while (dataReader.Read())

{

CategVente a = new CategVente();

a.idCategVente = dataReader["idCategVente"] == DBNull.Value ? string.Empty : dataReader["idCategVente"].ToString();

a.libelleCategVente = dataReader["libelleCategVente"] == DBNull.Value ? string.Empty : dataReader["libelleCategVente"].ToString();

categVente.Add(a);

}

}

dataReader.Close();

co.Close();

//retourne la liste à afficher

return categVente;

}

public CategVente idCategVente(string idCategVente)

{

string query = "SELECT idCategVente, libelleCategVente FROM categvente WHERE idCategVente = " + idCategVente;

Initialize();

CategVente categVente = new CategVente();

MySqlCommand cmd = new MySqlCommand(query, co);

MySqlDataReader dataReader = cmd.ExecuteReader();

if (dataReader.HasRows)

{

while (dataReader.Read())

{

categVente.idCategVente = dataReader["idCategVente"] == DBNull.Value ? string.Empty : dataReader["idCategVente"].ToString();

categVente.libelleCategVente = dataReader["libelleCategVente"] == DBNull.Value ? string.Empty : dataReader["libelleCategVente"].ToString();

}

}

dataReader.Close();

co.Close();

return categVente;

}

public void categVenteDelete(string idCategVente)

{

string query = "DELETE FROM categvente WHERE idCategVente = " + idCategVente;

nonQuery(query);

}

public void categVenteInsert(CategVente categVente)

{

string query = "INSERT INTO categvente(libelleCategVente) VALUES ('" + categVente.libelleCategVente + "');";

nonQuery(query);

}

public void categVenteUpdate(CategVente categVente)

{

string query = "UPDATE categvente SET libelleCategVente = '" + categVente.libelleCategVente + "' WHERE idCategVente = " + categVente.idCategVente;

nonQuery(query);

}

}

public class VenteDAO : AccesBD

{

public List<Vente> listVente()

{

string query = "SELECT idVente, marchandise, nbVente, C.libelleCategVente AS idCategVente FROM vente V INNER JOIN categvente C ON(V.idCategVente = C.idCategvente);";

// initialisation de la connexion

Initialize();

// création de la liste à retourner

List<Vente> vente = new List<Vente>();

MySqlCommand cmd = new MySqlCommand(query, co);

//crée un data reader et exécute la commande

MySqlDataReader dataReader = cmd.ExecuteReader();

if (dataReader.HasRows)

{

//lit la liste et stocke les données dans la liste

while (dataReader.Read())

{

Vente a = new Vente();

a.idVente = dataReader["idVente"] == DBNull.Value ? string.Empty : dataReader["idVente"].ToString();

a.marchandise = dataReader["marchandise"] == DBNull.Value ? string.Empty : dataReader["marchandise"].ToString();

a.nbVente = dataReader["nbVente"] == DBNull.Value ? string.Empty : dataReader["nbVente"].ToString();

a.idCategVente = dataReader["idCategVente"] == DBNull.Value ? string.Empty : dataReader["idCategVente"].ToString();

vente.Add(a);

}

}

dataReader.Close();

co.Close();

//retourne la liste à afficher

return vente;

}

public Vente idVente(string idVente)

{

string query = "SELECT idVente, marchandise, nbvente, idCategVente FROM vente WHERE idVente = " + idVente;

Initialize();

Vente vente = new Vente();

MySqlCommand cmd = new MySqlCommand(query, co);

MySqlDataReader dataReader = cmd.ExecuteReader();

if (dataReader.HasRows)

{

while (dataReader.Read())

{

vente.idVente = dataReader["idVente"] == DBNull.Value ? string.Empty : dataReader["idVente"].ToString();

vente.marchandise = dataReader["marchandise"] == DBNull.Value ? string.Empty : dataReader["marchandise"].ToString();

vente.nbVente = dataReader["nbVente"] == DBNull.Value ? string.Empty : dataReader["nbVente"].ToString();

vente.idCategVente = dataReader["idCategVente"] == DBNull.Value ? string.Empty : dataReader["idCategVente"].ToString();

}

}

dataReader.Close();

co.Close();

return vente;

}

public void venteDelete(string idVente)

{

string query = "DELETE FROM vente WHERE idVente = " + idVente;

nonQuery(query);

}

public void venteInsert(Vente vente)

{

string query = "INSERT INTO vente(marchandise, nbVente, idCategVente) VALUES ('" + vente.marchandise + "', '"+vente.nbVente+"', '"+vente.idCategVente+"');";

nonQuery(query);

}

public void venteUpdate(Vente vente)

{

string query = "UPDATE vente SET marchandise = '" + vente.marchandise + "', nbVente = '"+vente.nbVente+ "', '" + vente.idCategVente + "' WHERE idVente = " + vente.idVente;

nonQuery(query);

}

}

# Bibliothèque de classes

Une table correspond dans cet exemple à une classe.

public class Client

{

public string idClient { get; set; }

public string nom { get; set; }

public Client() { }

public Client(string idClient, string nom)

{

this.idClient = idClient;

this.nom = nom;

}

public Client(string nom)

{

this.nom = nom;

}

}

public class CategVente

{

public string idCategVente { get; set; }

public string libelleCategVente { get; set; }

public CategVente() { }

public CategVente(string idCategVente, string libelleCategVente)

{

this.idCategVente = idCategVente;

this.libelleCategVente = libelleCategVente;

}

public CategVente(string libelleCategVente)

{

this.libelleCategVente = libelleCategVente;

}

}

public class Vente

{

public string idVente { get; set; }

public string marchandise { get; set; }

public string nbVente { get; set; }

public string idCategVente { get; set; }

public Vente() { }

public Vente(string idVente, string marchandise, string nbVente, string idCategVente)

{

this.idVente = idVente;

this.marchandise = marchandise;

this.nbVente = nbVente;

this.idCategVente = idCategVente;

}

public Vente (string marchandise, string nbVente, string idCategVente)

{

this.marchandise = marchandise;

this.nbVente = nbVente;

this.idCategVente = idCategVente;

}

}

# Classe Manager

Le controller qui correspond à la couche logique métier fait la liaison entre la couche d’accès aux données et les vues en utilisant la bibliothèque de classes.

public class ManagerController : Controller

{

// GET: Manager

public ActionResult Index()

{

return View();

}

public ActionResult ClientView()

{

Models.ClientDAO c = new Models.ClientDAO();

// supression d'un client

if (Request.QueryString["idClient"] != null)

{

string idClient = String.Format("{0}", Request.QueryString["idClient"]);

c.clientDelete(idClient);

}

// insertion d'un client

if (Request.Form["Ajouter"] == "Ajouter" && Request.Form["nom"] != null)

{

string nom = String.Format("{0}", Request.Form["nom"]);

Models.Client client = new Models.Client(nom);

c.clientInsert(client);

}

// modification d'un client

if (Request.Form["Modifier"] == "Modifier" && Request.Form["idClient"] != null && Request.Form["nom"] != null)

{

string idClient = String.Format("{0}", Request.Form["idClient"]);

string nom = String.Format("{0}", Request.Form["nom"]);

Models.Client client = new Models.Client(idClient, nom);

c.clientUpdate(client);

}

ViewData["Clients"] = c.listClient();

return View();

}

public ActionResult ClientModifView()

{

if (Request.QueryString["idClient"] != null)

{

string idClient = Request.QueryString["idClient"];

Models.Client client = new Models.Client();

Models.ClientDAO c = new Models.ClientDAO();

client = c.idClient(idClient);

ViewData["idClient"] = client.idClient;

ViewData["nomClient"] = client.nom;

}

return View();

}

public ActionResult CategVenteView()

{

Models.CategVenteDAO c = new Models.CategVenteDAO();

// supression d'une categVente

if (Request.QueryString["idCategVente"] != null)

{

string idCategVente = String.Format("{0}", Request.QueryString["idCategVente"]);

c.categVenteDelete(idCategVente);

}

// insertion d'une categVente

if (Request.Form["Ajouter"] == "Ajouter" && Request.Form["libelleCategVente"] != null)

{

string libelleCategVente = String.Format("{0}", Request.Form["libelleCategVente"]);

Models.CategVente categVente = new Models.CategVente(libelleCategVente);

c.categVenteInsert(categVente);

}

// modification d'une categVente

if (Request.Form["Modifier"] == "Modifier" && Request.Form["idCategVente"] != null && Request.Form["libelleCategVente"] != null)

{

string idCategVente = String.Format("{0}", Request.Form["idCategVente"]);

string libelleCategVente = String.Format("{0}", Request.Form["libelleCategVente"]);

Models.CategVente categVente = new Models.CategVente(idCategVente, libelleCategVente);

c.categVenteUpdate(categVente);

}

ViewData["CategVentes"] = c.listCategVente();

return View();

}

public ActionResult CategVenteModifView()

{

if (Request.QueryString["idCategVente"] != null)

{

string idCategVente = Request.QueryString["idCategVente"];

Models.CategVente vente = new Models.CategVente();

Models.CategVenteDAO c = new Models.CategVenteDAO();

vente = c.idCategVente(idCategVente);

ViewData["idCategVente"] = vente.idCategVente;

ViewData["libelleCategVente"] = vente.libelleCategVente;

}

return View();

}

public ActionResult VenteView()

{

Models.VenteDAO c = new Models.VenteDAO();

Models.CategVenteDAO cv = new Models.CategVenteDAO();

// supression d'une Vente

if (Request.QueryString["idVente"] != null)

{

string idVente = String.Format("{0}", Request.QueryString["idVente"]);

c.venteDelete(idVente);

}

// insertion d'une Vente

if (Request.Form["Ajouter"] == "Ajouter" && Request.Form["marchandise"] != null)

{

string marchandise = String.Format("{0}", Request.Form["marchandise"]);

string nbVente = String.Format("{0}", Request.Form["nbVente"]);

string idCategVente = String.Format("{0}", Request.Form["idCategVente"]);

Models.Vente Vente = new Models.Vente(marchandise, nbVente, idCategVente);

c.venteInsert(Vente);

}

// modification d'une Vente

if (Request.Form["Modifier"] == "Modifier" && Request.Form["idVente"] != null && Request.Form["marchandise"] != null)

{

string idVente = String.Format("{0}", Request.Form["idVente"]);

string marchandise = String.Format("{0}", Request.Form["marchandise"]);

string nbVente = String.Format("{0}", Request.Form["nbVente"]);

string idCategVente = String.Format("{0}", Request.Form["idCategVente"]);

Models.Vente Vente = new Models.Vente(idVente, marchandise, nbVente, idCategVente);

c.venteUpdate(Vente);

}

ViewData["CategVentes"] = cv.listCategVente();

ViewData["Ventes"] = c.listVente();

return View();

}

public ActionResult VenteModifView()

{

if (Request.QueryString["idVente"] != null)

{

string idVente = Request.QueryString["idVente"];

Models.Vente vente = new Models.Vente();

Models.VenteDAO c = new Models.VenteDAO();

vente = c.idVente(idVente);

ViewData["idVente"] = vente.idVente;

ViewData["marchandise"] = vente.marchandise;

ViewData["nbVente"] = vente.nbVente;

ViewData["idCategVente"] = vente.idCategVente;

}

return View();

}

}