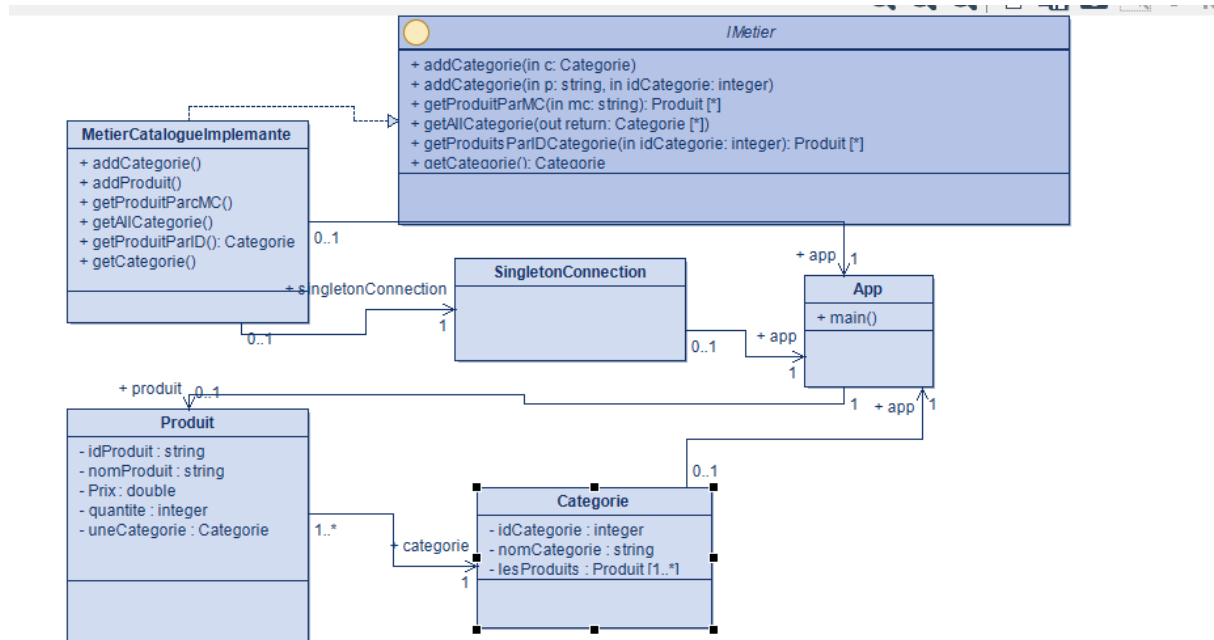


TP 3 JDBC & ORM & SWING

1. Établir un diagramme de classe de la couche métier.



3. Créer la base de données Mysql.

The screenshot shows the phpMyAdmin interface for a MySQL database named 'catalogue1'. The 'Structure' tab is active, displaying the following tables:

Table	Action	Lignes	Type	Interclassement	Taille	Perte
categorie	Parcourir Structure Rechercher Insérer Vider Supprimer	2	InnoDB	utf8mb4_general_ci	16,0 kio	-
produit	Parcourir Structure Rechercher Insérer Vider Supprimer	3	InnoDB	utf8mb4_general_ci	32,0 kio	-
2 tables	Somme	5	InnoDB	utf8mb4_general_ci	48,0 kio	0 0

Below the table list, the 'Structure de table' for 'categorie' is shown, detailing its columns:

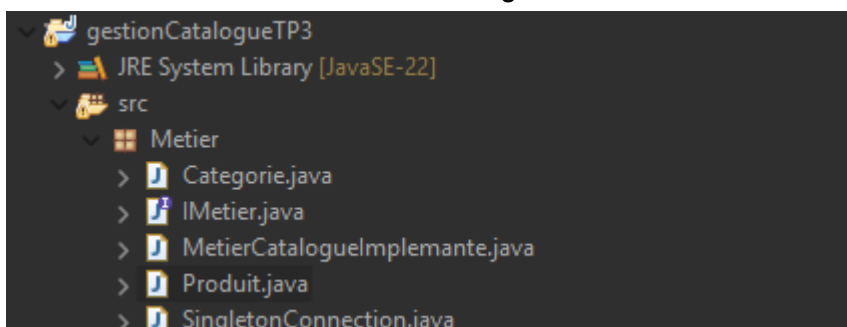
#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
1	idCategorie	int(11)			Non	Aucun(e)			Modifier Supprimer Plus
2	nomCategorie	varchar(50)	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus

The interface also includes navigation links like 'Parcourir', 'Structure', 'SQL', 'Rechercher', 'Insérer', 'Exporter', 'Importer', 'Privileges', 'Operations', 'Suivi', and 'Déclencheurs'.

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
<input type="checkbox"/>	1 idProduit	varchar(11)	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	2 nomProduit	varchar(50)	utf8mb4_general_ci		Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	3 prix	double			Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	4 quantite	int(11)			Non	Aucun(e)			Modifier Supprimer Plus
<input type="checkbox"/>	5 idCategorie	int(11)			Oui	NULL			Modifier Supprimer Plus

4. Créer les classes Persistantes Produit et Categorie.
 Avant de créer ces classes nous allons créer un nouveau répertoire.

Nous créer les classes Produit et Categorie:



Contenue de la classe “Categorie”:

```
package Metier;
public class Produit {
private String idProduit;
private String nomProduit;
private double prix;
private int quantite;
private Categorie categorie;
public Produit(String idProduit, String nomProduit, double prix, int quantite) {
    super();
    this.idProduit = idProduit;
    this.nomProduit = nomProduit;
    this.prix = prix;
    this.quantite = quantite;
}
public Produit() {
}
public String getIdProduit() {
    return idProduit;
}
public void setIdProduit(String idProduit) {
    this.idProduit = idProduit;
}
```

```

}
public String getNomProduit() {
    return nomProduit;
}
public void setNomProduit(String nomProduit) {
    this.nomProduit = nomProduit;
}
public double getPrix() {
    return prix;
}
public void setPrix(double prix) {
    this.prix = prix;
}
public int getQuantite() {
    return quantite;
}
public void setQuantite(int quantite) {
    this.quantite = quantite;
}
public Categorie getCategorie() {
    return categorie;
}
public void setCategorie(Categorie categorie) {
    this.categorie = categorie;
}
@Override
public String toString() {
    return "Produit [idProduit=" + idProduit + ", nomProduit=" + nomProduit + ",
prix=" + prix + ", quantite="
        + quantite + ", categorie=" + categorie + "]";
}
}

```

Contenue de la classe “Produit”:

```

“package Metier;
public class Produit {
    private String idProduit;
    private String nomProduit;
    private double prix;
    private int quantite;
    private Categorie categorie;
    public Produit(String idProduit, String nomProduit, double prix, int quantite) {
        super();
    }
}

```

```

        this.idProduit = idProduit;
        this.nomProduit = nomProduit;
        this.prix = prix;
        this.quantite = quantite;
    }

    public Produit() {
    }

    public String getIdProduit() {
        return idProduit;
    }

    public void setIdProduit(String idProduit) {
        this.idProduit = idProduit;
    }

    public String getNomProduit() {
        return nomProduit;
    }

    public void setNomProduit(String nomProduit) {
        this.nomProduit = nomProduit;
    }

    public double getPrix() {
        return prix;
    }

    public void setPrix(double prix) {
        this.prix = prix;
    }

    public int getQuantite() {
        return quantite;
    }

    public void setQuantite(int quantite) {
        this.quantite = quantite;
    }

    public Categorie getCategorie() {
        return categorie;
    }

    public void setCategorie(Categorie categorie) {
        this.categorie = categorie;
    }

    @Override
    public String toString() {
        return "Produit [idProduit=" + idProduit + ", nomProduit=" + nomProduit + ",
        prix=" + prix + ", quantite="
        + quantite + ", categorie=" + categorie + "]";
    }
}

```

5. Créer l'interface IMetier déclarant les méthodes à implémenter.
Ensuite nous allons créer un nouveau fichier interfaces :
Le code introduit est le suivant :

```
package Metier;
import java.util.List;
public interface IMetier {
    //Ajouter une nouvelle Catégorie
    public void addCategorie(Categorie c);
    //Ajouter un produit appartenant à une catégorie
    public void addProduit(Produit p, int idCategorie);
    //Lister les produits en fonction du mot clé
    public List<Produit> getProduitParMC(String mc);
    //Lister les produits en fonction de l'ID de catégorie
    public List<Produit> getProduitsParIDCategorie(int idCategorie);
    //Lister toutes les catégories
    public List<Categorie> getAllCategorie();
    //Récupérer une catégorie en sachant son idCategorie
    public Categorie getCategorie(int idCategorie);
}
```

6. Créer une classe SingletonConnection contenant une méthode statique getConnection qui permet de retourner une Connection unique vers la base de données.

Le code ci-joint va retourner une connexion à ma base de données nommée ici "catalogue1";

```
package Metier;
import java.sql.Connection;
import java.sql.DriverManager;
public class SingletonConnection {
    private static Connection connection;
    static {
        try {
            Class.forName("com.mysql.jdbc.Driver");

            connection=DriverManager.getConnection("jdbc:mysql://localhost:3306/catalogue1",
            "root", "");

            System.out.println("Création d'une connection");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    public static Connection getConnection() {
```

```

        return connection;
    }
}

```

7. Créer une implémentation de l'interface IMetier.

Ensuite, je vais créer un code pour implémenter l'interface Métier:

```

package Metier;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.List;
public class MetierCatalogueImplemante implements IMetier {
    @Override
    public void addCategorie(Categorie c) {
        Connection conn = SingletonConnection.getConnection();
        try{
            PreparedStatement ps =
conn.prepareStatement("INSERT INTO categorie (nomCategorie) VALUES (?)");
            ps.setString(1, c.getNomCategorie());
            ps.executeUpdate();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    @Override
    public void addProduit(Produit p, int idCategorie) {
        Connection conn = SingletonConnection.getConnection();
        try {
            PreparedStatement ps = conn.prepareStatement("INSERT INTO
produit (idProduit, nomProduit, prix, quantite, idCategorie) VALUES (?, ?, ?, ?, ?)");
            ps.setString(1, p.getIdProduit());
            ps.setString(2, p.getNomProduit());
            ps.setDouble(3, p.getPrix());
            ps.setInt(4, p.getQuantite());
            ps.setInt(5, idCategorie);
            ps.executeUpdate();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    @Override

```

```

public List<Produit> getProduitParMC(String mc) {
    List<Produit> lesproduits = new ArrayList<>();
    Connection conn = SingletonConnection.getConnection();
    try {
        PreparedStatement ps = conn.prepareStatement("SELECT *
FROM produit WHERE nomProduit LIKE ?");
        ps.setString(1, "%" + mc + "%");
        try (ResultSet rs = ps.executeQuery()) {
            while (rs.next()) {
                Produit p = new Produit();
                p.setIdProduit(rs.getString("idProduit"));
                p.setNomProduit(rs.getString("nomProduit"));
                p.setPrix(rs.getDouble("prix"));
                p.setQuantite(rs.getInt("quantite"));
                int idCategorie = rs.getInt("idCategorie");
                try (PreparedStatement ps2 =
conn.prepareStatement("SELECT * FROM categorie WHERE idCategorie = ?")) {
                    ps2.setInt(1, idCategorie);
                    try (ResultSet rs2 = ps2.executeQuery()) {
                        if (rs2.next()) {
                            Categorie cat = new
Categorie();
cat.setIdCategorie(rs2.getInt("idCategorie"));
cat.setNomCategorie(rs2.getString("nomCategorie"));
p.setCategorie(cat);
                        }
                    }
                }
                lesproduits.add(p);
            }
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
    return lesproduits;
}

@Override
public List<Produit> getProduitsParIDCategorie(int idCategorie) {
    List<Produit> produits = new ArrayList<Produit>();
    Connection conn = SingletonConnection.getConnection();
    try {

```

```

        PreparedStatement ps=conn.prepareStatement("select * from
produit where idCategorie =?");
        ps.setInt(1, idCategorie);
        ResultSet rs=ps.executeQuery();
        while (rs.next()) {
            Produit p = new Produit();
            p.setIdProduit(rs.getString("idProduit"));
            p.setNomProduit(rs.getString("nomProduit"));
            p.setPrix(rs.getDouble("prix"));
            p.setQuantite(rs.getInt("quantite"));
            idCategorie=rs.getInt("idCategorie");
            PreparedStatement ps2=conn.prepareStatement("select *
from categorie where idCategorie= ?");
            ps2.setInt(1, idCategorie);
            ResultSet rs2=ps2.executeQuery();
            if(rs2.next()) {
                Categorie cat=new Categorie();
                cat.setIdCategorie(rs2.getInt(1));
                cat.setNomCategorie(rs2.getString(2));
                p.setCategorie(cat);
            }
            produits.add(p);
            ps2.close();
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
    return produits;
}

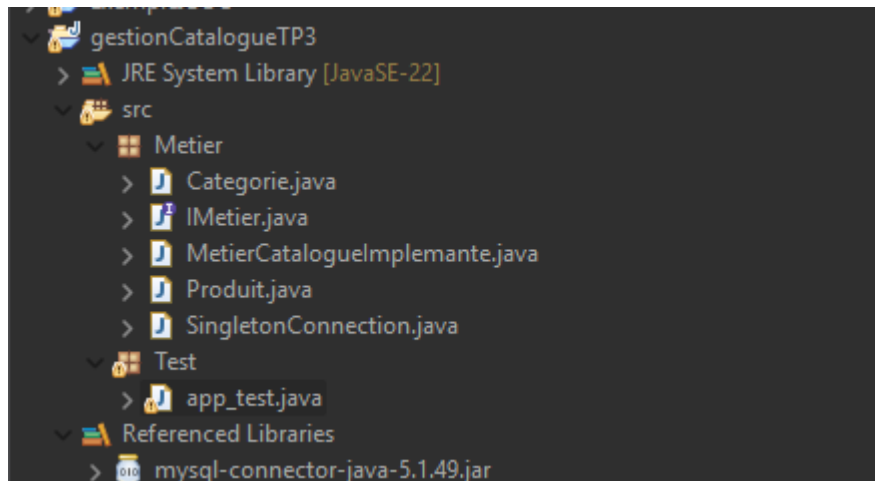
@Override
public List<Categorie> getAllCategorie() {
    // Implémentation nécessaire
    return null;
}

@Override
public Categorie getCategorie(int idCategorie) {
    // Implémentation nécessaire
    return null;
}
}

```


8. Créer une application java qui permet de tester l'ensemble des méthodes de la couche métier. (En mode console).

Puis une fois cela fait, je vais tester le code via une app dans la couche métier, pour cela, je crée un nouveau dossier depuis le répertoire principal que j'appelle "Test" et le fichier "App-test.java", ensuite, j'introduis le code suivant.



```
package Test;
import java.sql.Connection;
import Metier.Categorie;
import Metier.MetierCatalogueImplemante;
import Metier.SingletonConnection;
public class app_test {
    public static void main(String[] args) {
        Connection conn1=SingletonConnection.getConnection();
        MetierCatalogueImplemante metier=new MetierCatalogueImplemante();
        metier.addCategorie(new Categorie("Ecran"));
        //metier.addCategorie(new Categorie("CLE USB"));
        //metier.addCategorie(new Categorie("PC"));
        //metier.addCategorie(new Categorie("Clavier"));
        //metier.addProduit(new Produit("pp1", "Dell200", 2000,40), 4);
        //metier.addProduit(new Produit("pp2", "HP200", 1500,50), 4);
        //metier.addProduit(new Produit("pp3", "Acer39", 1200,90), 4);
    }
}
```

Voici le résultat au niveau de la console:

```
Help
D DemandeAbsen... D Salarie.java D DateFr.java D Point.java D Compte.java D Article.java D VendableKil... D VendablePiec... D Solde.java D app_test.java X AppAWT.java »33

package test;

import java.sql.Connection;

public class app_test {
    public static void main(String[] args) {
        Connection conn1=SingletonConnection.getConnection();
        Connection conn2=SingletonConnection.getConnection();
        Connection conn3=SingletonConnection.getConnection();
        Connection conn4=SingletonConnection.getConnection();

        MetierCatalogueImplemante metier=new MetierCatalogueImplemante();
        // List<Categorie>cat=metier.getAllCategorie();
        // for(Categorie c:cat) {
        //     System.out.println(c.getNomCategorie());
        // }
        // Categorie cat1=metier.getCategorie(4);
        // System.out.println(cat1);
        List<Produit> produits=metier.getProduitsParIDCategorie(4);
        for(Produit p:produits) {
            System.out.println(p);
        }
        // metier.addCategorie(new Categorie("Console"));
        // metier.addCategorie(new Categorie("PC"));
        // metier.addCategorie(new Categorie("Clavier"));
        // metier.addProduit(new Produit("pp5", "PS4", 300,20, 4);
        // metier.addProduit(new Produit("pp4", "PS5", 500,30, 4);
        // metier.addProduit(new Produit("pp1", "Dell200", 2000,40, 4);
        // metier.addProduit(new Produit("pp2", "HP200", 1500,50, 4);
        // metier.addProduit(new Produit("pp3", "Acer39", 1200,90, 4);
    }
}

Problems Javadoc Declaration Console X
<terminated> app_test [Java Application] C:\Users\udem\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.23.0.0.v20240919-1706\jre\bin\javaw.exe (4 déc. 2024, 19:36:37 - 19:36:37) [pid: 16208]

Création d'une connexion
Produit [idProduit=pp1, nomProduit=Dell200, prix=2000.0, quantite=40, categorie=Categorie [idCategorie=4, nomCategorie=Clavier, lesProduits=null]]
Produit [idProduit=pp2, nomProduit=HP200, prix=1500.0, quantite=50, categorie=Categorie [idCategorie=4, nomCategorie=Clavier, lesProduits=null]]
Produit [idProduit=pp3, nomProduit=Acer39, prix=1200.0, quantite=90, categorie=Categorie [idCategorie=4, nomCategorie=Clavier, lesProduits=null]]
Produit [idProduit=pp4, nomProduit=PS5, prix=500.0, quantite=30, categorie=Categorie [idCategorie=4, nomCategorie=Clavier, lesProduits=null]]
Produit [idProduit=pp5, nomProduit=PS4, prix=300.0, quantite=20, categorie=Categorie [idCategorie=4, nomCategorie=Clavier, lesProduits=null]]
```