AKILABANA MOKILI
Joseph
BTS SIO 2

Compte Rendu

Sommaire

Introduction

L'objectif de ce TP était de développer une application Java permettant de gérer un répertoire de contacts en combinant JavaFX pour l'interface graphique et JDBC pour la persistance des données dans une base MySQL. L'application devait permettre d'afficher tous les contacts existants, d'ajouter de nouveaux contacts dans le répertoire et dans la base de données, de naviguer entre les contacts et de quitter l'application

Base de données

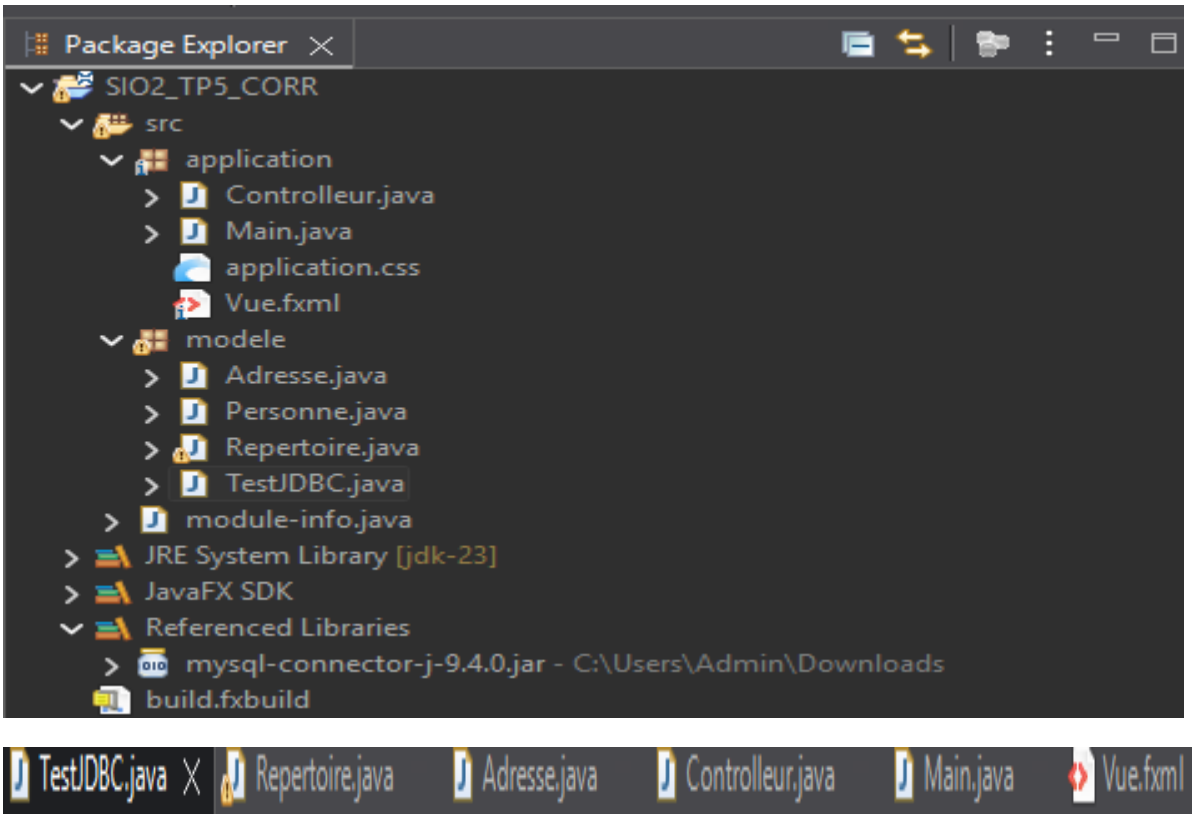
La base utilisée est le répertoire (avec accent, un problème que j'avais du mal à remarquer), contenant une table contact avec les colonnes suivantes : nom, prénom, tel, adresse, cp et ville. La structure de la table a été adaptée pour correspondre aux champs de l'application afin d'éviter les erreurs SQL comme Column not found car j'avais cette au début du TP. L'URL JDBC respecte le nom exact de la base, ce qui permet de se connecter correctement à MySQL.

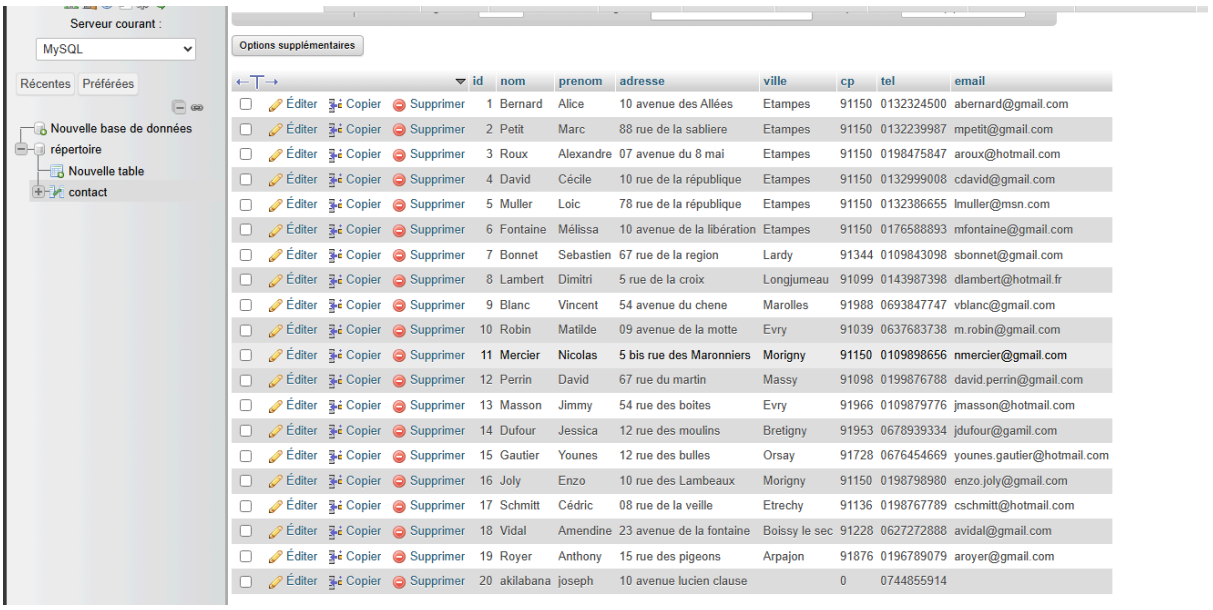
Voici le les élément de table contact

```
    public static void addContact(Personne p) {

        String sql = "INSERT INTO contact(nom, prenom, tel, adresse, cp, ville) VALUES(?,?,?,?,?,?)";

        try (
            Connection conn = DriverManager.getConnection(URL, USER, PASS);
            PreparedStatement ps = conn.prepareStatement(sql);
        ) {
            ps.setString(1, p.getNom());
            ps.setString(2, p.getPrenom());
            ps.setString(3, p.getTelephone());
            ps.setString(4, p.getAdresse().getRue());
            ps.setString(5, p.getAdresse().getCodePostal());
            ps.setString(6, p.getAdresse().getVille());

            ps.executeUpdate();

        } catch(SQLException e) {
            System.err.println("Erreur d'insertion : " + e);
        }
    }
}
```

Architecture du projet

Le projet est organisé en deux packages : **modele** et application. Le package modele contient les classes **Personne** et **Adresse**, la classe **Repertoire** qui gère la liste des contacts et la classe **TestJDBC** qui assure les opérations SQL (lecture et insertion de contacts). Le package **application** contient le contrôleur JavaFX Controlleur, qui initialise le **répertoire** depuis la base de données, gère l'ajout de contacts et permet de naviguer dans la liste via les boutons Précédent et Suivant.





	id	nom	prenom	adresse	ville	cp	tel	email
Éditer Copier Supprimer	1	Bernard	Alice	10 avenue des Allées	Etampes	91150	0132324500	abernard@gmail.com
Éditer Copier Supprimer	2	Petit	Marc	88 rue de la sabliere	Etampes	91150	0132239987	mpetit@gmail.com
Éditer Copier Supprimer	3	Roux	Alexandre	07 avenue du 8 mai	Etampes	91150	0198475847	aroux@hotmail.com
Éditer Copier Supprimer	4	David	Cécile	10 rue de la république	Etampes	91150	0132999008	cdavid@gmail.com
Éditer Copier Supprimer	5	Muller	Loic	78 rue de la république	Etampes	91150	0132386655	lmuller@msn.com
Éditer Copier Supprimer	6	Fontaine	Mélissa	10 avenue de la libération	Etampes	91150	0176588893	mfontaine@gmail.com
Éditer Copier Supprimer	7	Bonnet	Sebastien	67 rue de la region	Lardy	91344	0109843098	sbonnet@gmail.com
Éditer Copier Supprimer	8	Lambert	Dimitri	5 rue de la croix	Longjumeau	91099	0143987398	dlambert@hotmail.fr
Éditer Copier Supprimer	9	Blanc	Vincent	54 avenue du chene	Marolles	91988	0693847747	vblanc@gmail.com
Éditer Copier Supprimer	10	Robin	Matilde	09 avenue de la motte	Evry	91039	0637683738	m.robin@gmail.com
Éditer Copier Supprimer	11	Mercier	Nicolas	5 bis rue des Maronniers	Morigny	91150	0109898656	nmercier@gmail.com
Éditer Copier Supprimer	12	Perrin	David	67 rue du martin	Massy	91098	0199876788	david.perrin@gmail.com
Éditer Copier Supprimer	13	Masson	Jimmy	54 rue des boites	Evry	91966	0109879776	jmasson@hotmail.com
Éditer Copier Supprimer	14	Dufour	Jessica	12 rue des moulins	Bretigny	91953	0678939334	jdufour@gamil.com
Éditer Copier Supprimer	15	Gautier	Younes	12 rue des bulles	Orsay	91728	0676454669	younes.gautier@hotmail.com
Éditer Copier Supprimer	16	Joly	Enzo	10 rue des Lambeaux	Morigny	91150	0198798980	enzo.joly@gmail.com
Éditer Copier Supprimer	17	Schmitt	Cédric	08 rue de la veille	Etrechy	91136	0198767789	cschmitt@hotmail.com
Éditer Copier Supprimer	18	Vidal	Amendine	23 avenue de la fontaine	Boissy le sec	91228	0627272888	avidal@gmail.com
Éditer Copier Supprimer	19	Royer	Anthony	15 rue des pigeons	Arpajon	91876	0196789079	aroyer@gmail.com
Éditer Copier Supprimer	20	akilabana	joseph	10 avenue lucien clause		0	0744855914	

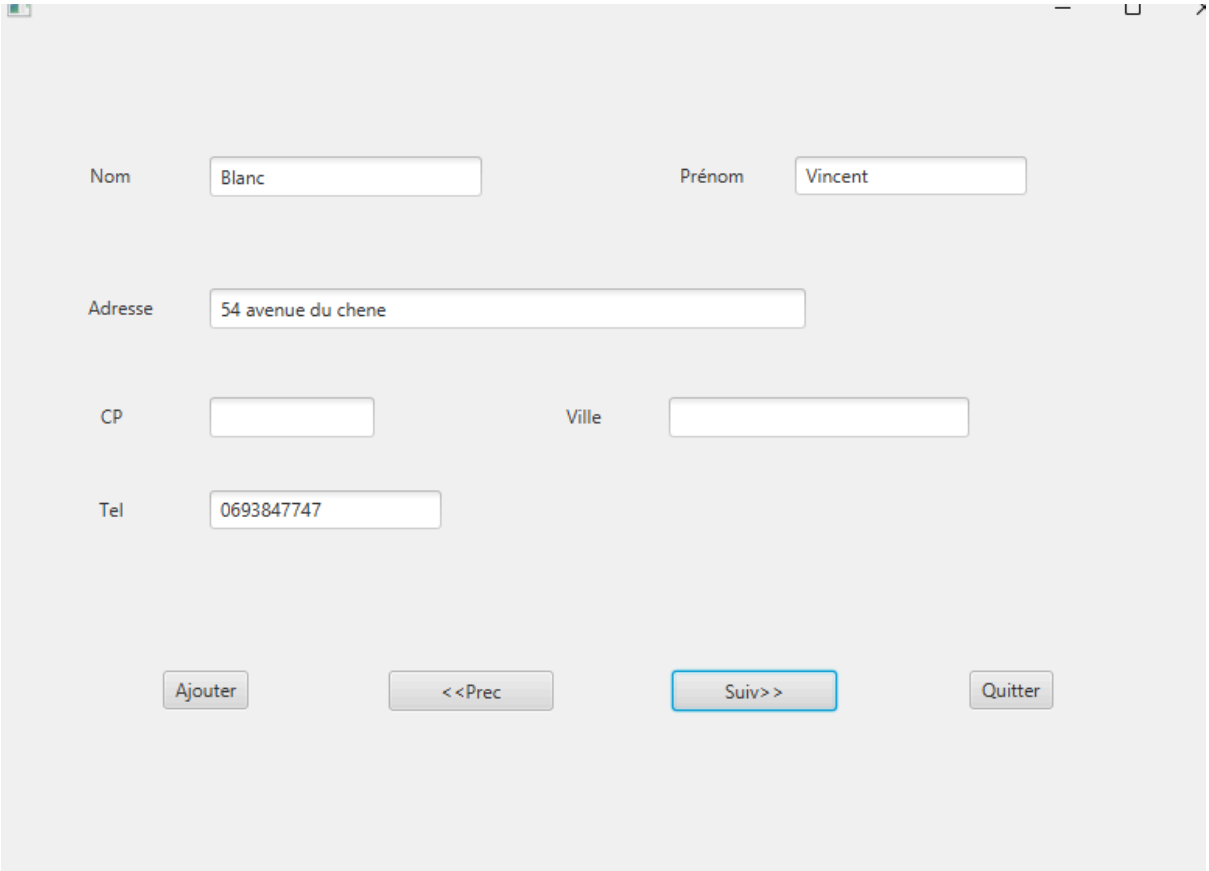
Fonctionnement de l’application

Au démarrage, l’application charge tous les contacts depuis la base via TestJDBC.getAllContacts() et les ajoute au Repertoire. L’utilisateur peut ajouter un nouveau contact en saisissant les informations dans les TextField correspondants (nom, prénom, adresse, CP, ville, téléphone). Le contact est ensuite ajouté à la fois dans le Repertoire et dans la base MySQL via TestJDBC.addContact(). La navigation entre les contacts se fait grâce aux boutons Précédent et Suivant, avec un message d’alerte en cas de dépassement des limites. Enfin, le bouton Quitter ferme l’application proprement.

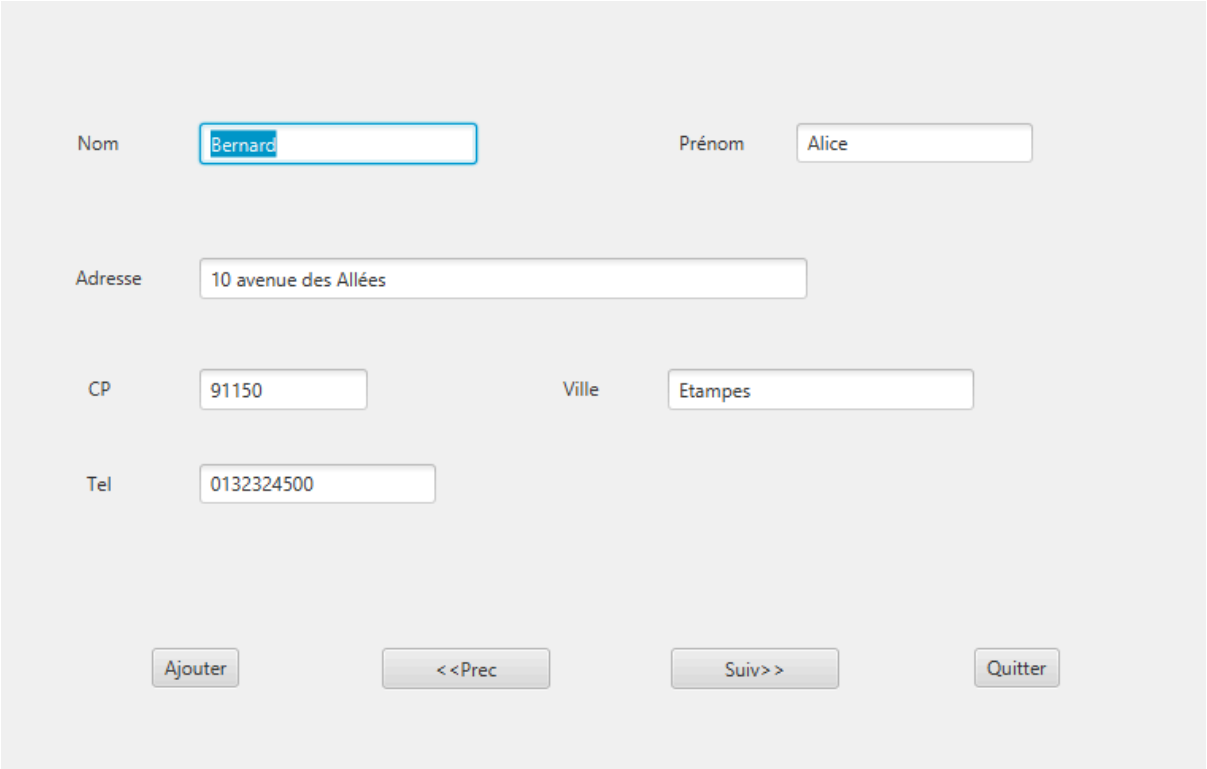
```java
public static List<Personne> getAllContacts() {

    List<Personne> liste = new ArrayList<>();

    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
    } catch(Exception e) {
        System.err.println("Driver non chargé : " + e);
    }

    try (
        Connection conn = DriverManager.getConnection(URL, USER, PASS);
        Statement st = conn.createStatement();
        ResultSet rs = st.executeQuery("SELECT * FROM contact");
    ) {

        while (rs.next()) {
            Adresse adr = new Adresse(
                rs.getString("adresse"),  // rue
                rs.getString("cp"),       // code postal
                rs.getString("ville")     // ville
            );

            Personne p = new Personne(
                rs.getString("nom"),
                rs.getString("prenom"),
```

Problèmes rencontrés et solutions

Pendant le TP, plusieurs problèmes ont été rencontrés : d’abord, une erreur Base 'repertoire' inconnue causée par la différence de nom (répertoire avec accent) dans MySQL. Ensuite, des erreurs Column not found apparaissent car les noms de colonnes dans la table ne correspondaient pas aux noms utilisés dans le code Java (tel ou téléphone). Enfin, la gestion complète de l’adresse a nécessité d’ajouter les colonnes cp et ville et d’adapter les classes Adresse et le mapping SQL.



Mais à la fin je réussi à régler le problème pour afficher la partie CP donc code postale et Ville.



Conclusion

Ce TP a permis de comprendre l'intégration de JavaFX et JDBC pour créer une application CRUD fonctionnelle. L'utilisation de la classe Repertoire pour gérer les contacts en mémoire à faciliter la navigation et la manipulation des données côté application, tout en garantissant la persistance dans la base de données. L'application est désormais capable de charger, afficher, ajouter et parcourir les contacts de manière intuitive, offrant une solution complète pour la gestion d'un répertoire de contacts.

Annexes

Classe TestJDBC

```java
package modele;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class TestJDBC {

    private static final String URL = "jdbc:mysql://localhost/répertoire"; // accent correct
    private static final String USER = "root";
    private static final String PASS = "";

    // Récupérer tous les contacts
    public static List<Personne> getAllContacts() {

        List<Personne> liste = new ArrayList<>();

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch(Exception e) {
            System.err.println("Driver non chargé : " + e);
        }

        try (
            Connection conn = DriverManager.getConnection(URL, USER, PASS);
            Statement st = conn.createStatement();
            ResultSet rs = st.executeQuery("SELECT * FROM contact");
        ) {

            while (rs.next()) {
                Adresse adr = new Adresse(
                    rs.getString("adresse"),  // rue
                    rs.getString("cp"),       // code postal
                    rs.getString("ville")     // ville
                );

                Personne p = new Personne(
                    rs.getString("nom"),
                    rs.getString("prenom"),
                    adr,
                    rs.getString("tel")       // téléphone
                );

                liste.add(p);
            }

        } catch(SQLException e) {
            System.err.println("Erreur SQL : " + e);
        }

        return liste;
    }

    // Ajouter un contact
    public static void addContact(Personne p) {

        String sql = "INSERT INTO contact(nom, prenom, tel, adresse, cp, ville) VALUES(?,?,?,?,?,?)";

        try (
```

Class Controlleur

```java
package application;

import javafx.fxml.FXML;
import javafx.scene.control.Alert;
import javafx.scene.control.Button;
import javafx.scene.control.TextField;
import javafx.scene.control.Alert.AlertType;
import modele.Adresse;
import modele.Personne;
import modele.Repertoire;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class Controlleur {

    private Repertoire rep = new Repertoire();
    private int numCourant = 0;

    @FXML private TextField nom, prenom, adresse, cp, ville, tel;
    @FXML private Button bAjout, bPrec, bSuiv, bQuitter;

    // JDBC
    private static final String URL = "jdbc:mysql://localhost/répertoire";
    private static final String USER = "root";
    private static final String PASS = "";

    // Charger tous les contacts depuis la base
    public static List<Personne> getAllContacts() {

        List<Personne> liste = new ArrayList<>();

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
        } catch(Exception e) {
            System.err.println("Driver non chargé : " + e);
        }

        try (
            Connection conn = DriverManager.getConnection(URL, USER, PASS);
            Statement st = conn.createStatement();
            ResultSet rs = st.executeQuery("SELECT * FROM contact");
        ) {

            while (rs.next()) {
                Adresse adr = new Adresse(
                    rs.getString("adresse"),
                    rs.getString("cp"),
                    rs.getString("ville")
                );

                Personne p = new Personne(
                    rs.getString("nom"),
                    rs.getString("prenom"),
                    adr,
                    rs.getString("tel")
                );
```

Classe Repertoire

```java
package modele;

import java.util.ArrayList;
import java.util.Collection;
import java.util.Collections;

public class Repertoire {

    private ArrayList<Personne> liste;

    public Repertoire() {
        super();
        this.liste = new ArrayList<Personne>();
    }

    public String toString() {
        String laListe = "";
        for(Personne p : liste) {
            laListe += p + "\n";
        }
        return laListe;
    }

    public void ajoutePersonne(Personne p) {
        liste.add(p);
        Collections.sort(liste);
    }

    /* Méthode simple :
     *
    public Personne rechercheNom(String unNom) {
        Personne personeRecherchee = null;
        for(Personne p : liste) {
            if(p.getNom().equals(unNom)) {
                personeRecherchee = p;
            }
        }
        return personeRecherchee;
    }

    */

    public Personne rechercheNom(String unNom) {
        for(Personne p : liste) {
            if(p.getNom().equals(unNom)) {
                return p;
            }
        }
        return null;
    }

    public Personne rechercheNomPrenom(String unNom,String unPrenom) {
        for(Personne p : liste) {
            if(p.getNom().equals(unNom) && p.getPrenom().equals(unPrenom)) {
                return p;
            }
        }
        return null;
    }
                );

                Personne p = new Personne(
                    rs.getString("nom"),
                    rs.getString("prenom"),
                    adr,
                    rs.getString("tel")
                );
```