java projet

Rapport Projet fin de module : POO en Java

Tracking des livreurs

Réalisés par

Mohammed AACHABI

Encadré par

El Mokhtar EN-NAIMI

ELAACHAk LOTFI

L'objectif de ce rapport est de présenter la mise en place d'une application desktop Java utilisant les technologies JDBC et JavaFX pour gérer les livreurs, les commandes et les produits, ainsi que de fournir un tableau de bord générique. Cette application est destinée à faciliter la gestion de la livraison de produits en automatisant les processus de commande, de suivi et de gestion des livreurs.

Le rapport se compose de différentes sections, commençant par une présentation de l'objectif et du contexte de l'application. Ensuite, les différentes fonctionnalités de l'application sont détaillées, y compris la gestion des livreurs, des commandes et des produits, ainsi que la création d'un tableau de bord pour une vue d'ensemble des activités.

La section suivante présente l'architecture de l'application, en détaillant les différents composants et leur rôle. Ensuite, le processus de développement de l'application est décrit, en incluant les différentes étapes de conception, d'implémentation et de test.
Enfin, le rapport se termine par une conclusion générale qui résume les principales réalisations de l'application, ainsi que les limitations et les perspectives d'amélioration pour les versions futures.

Objectif

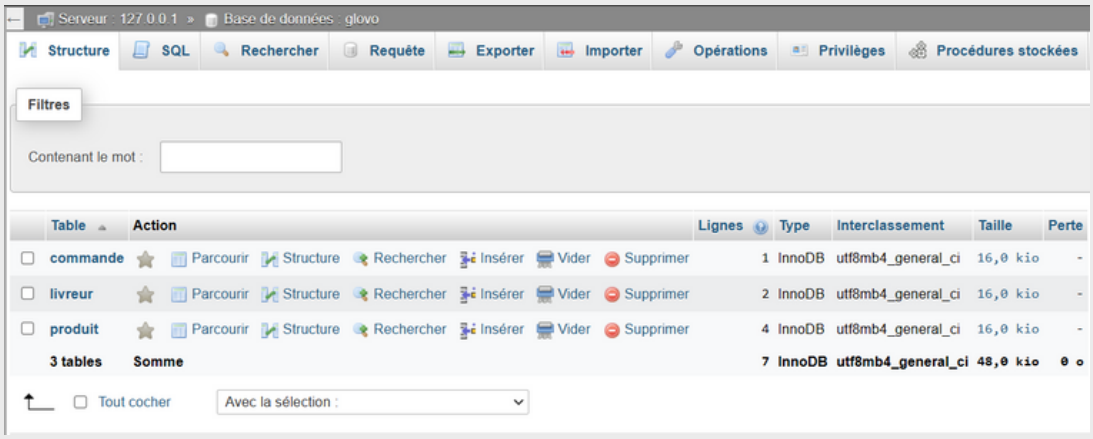
L'objectif de cette application est de simplifier et d'automatiser la gestion de la livraison de produits en fournissant une interface conviviale pour les utilisateurs. Elle permet de gérer les livreurs, les commandes et les produits, ainsi que de fournir un tableau de bord pour une vue d'ensemble des activités.

En utilisant JDBC pour interagir avec une base de données MySQL, l'application peut stocker et gérer efficacement les données relatives aux livreurs, aux commandes et aux produits. La création d'un tableau de bord permet aux utilisateurs de suivre et de visualiser facilement l'état des commandes, le suivi des livreurs et les niveaux de stock des produits.

En somme, l'objectif principal de cette application est d'améliorer l'efficacité et la productivité dans la gestion de la livraison de produits en fournissant une solution automatisée et conviviale pour les utilisateurs.

création de la base de données

Création d'une base de données MySQL <<glovo>> avec les tables pour les livreurs, les commandes et les produits .



création du projet

j'ai créé ce projet Java en utilisant IDE IntelliJ IDEA. et j'ai d'inclus les dépendances nécessaires pour JDBC et JavaFX.

code :

```xml
<!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.32</version>
</dependency>
```

création du class livreur

code Livreur.java :

```java
package ma.fstt.model;
// java bean
27 usages
public class Livreur {
        4 usages
        private Long id_livreur ;

        4 usages
        private String nom ;

        4 usages
        private String telephone ;

    1 usage
    public Livreur() {
    }

    2 usages
    public Livreur(Long id_livreur, String nom, String telephone) {
        this.id_livreur = id_livreur;
        this.nom = nom;
        this.telephone = telephone;
    }

    2 usages
    public Long getId_livreur() { return id_livreur; }

    1 usage
    public void setId_livreur(Long id_livreur) { this.id_livreur = id_livreur; }
```

création du class produit

code Produit.java:

```java
package ma.fstt.model;

32 usages
public class Produit {
    4 usages
    private Long id_Produit ;

    4 usages
    private String libele ;

    4 usages
    private String designation ;

    4 usages
    private String prix;

    2 usages
    public Long getId_Produit() { return id_Produit; }
    no usages
    public Produit (){

    }
```

création du class cammande

code Commande.java:

```java
package ma.fstt.model;
27 usages
public class Commande {
    4 usages
    private Long id_commande;
    4 usages
    private String libele;
    4 usages
    private String code;
    4 usages
    private String date;
    2 usages
    public Commande() {}
    1 usage
    public  Commande(Long id_commande,String libele,String code,String date){
        this.id_commande=id_commande;
        this.libele=libele;
        this.code=code;
        this.date=date;

    }
    2 usages
    public Long getId_commande() { return id_commande; }

    2 usages
    public void setId_commande(Long id_commande) { this.id_commande = id_commande; }
```

création du baseDAO

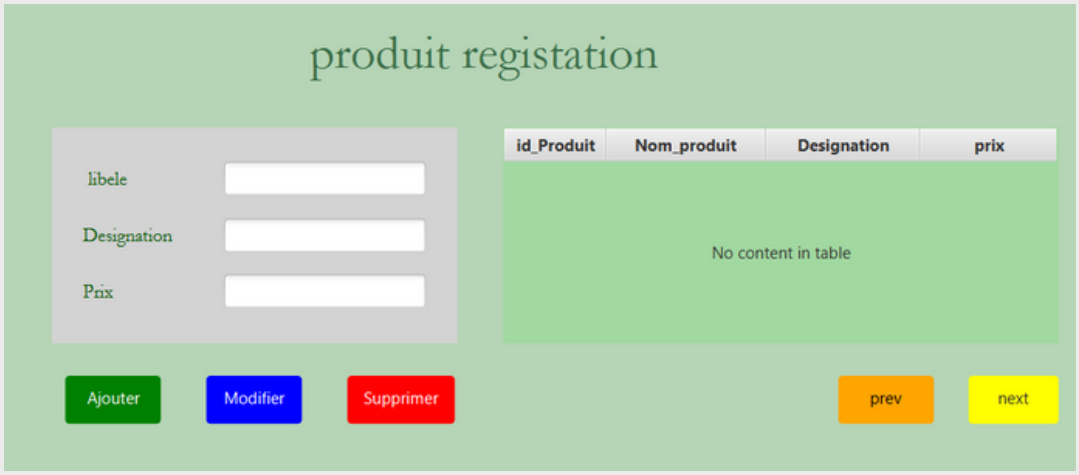
J'ai créé un objet d'accès aux données (DAO) pour chaque entité. Ce DAO utilisés gérer toutes les opérations CRUD (créer, lire, mettre à jour, supprimer) pour cette entité. et j'ai Utilisée JDBC pour interagir avec la base de données.

code baseDAO.java:

```java
package ma.fstt.model;
import java.sql.*;
import java.util.List;
3 usages  3 inheritors
public abstract  class BaseDAO <T>{
        16 usages
        protected Connection connection ;
        6 usages
        protected Statement statement ;
        50 usages
        protected PreparedStatement preparedStatement;
        34 usages
        protected ResultSet resultSet ;
        // connexion avec bdd
            1 usage
            private String url = "jdbc:mysql://127.0.0.1:3306/glovo";
            1 usage
            private String login = "root";
            1 usage
            private String password = "";
    3 usages
    BaseDAO() throws SQLException {
        this.connection = DriverManager.getConnection(url , login ,password );
    }
    4 usages  3 implementations
    public abstract void save( T object ) throws SQLException;
    3 implementations
    public abstract void update( T object ) throws SQLException ;
    3 implementations
    public abstract void update( T object ) throws SQLException ;
    3 implementations
    public abstract void delete( T object ) throws SQLException ;
    4 usages  3 implementations
    public abstract List<T> getAll(  ) throws SQLException ;
    no usages  3 implementations
    public abstract T getOne( Long id  ) throws SQLException  ;
}
```

Hello_view.fxml:

second_interface.fxml:



third_interface.fxml:

liant tous ensemble en créant des classes de contrôleurs qui coordonne les DAO et l'interface utilisateur. Le contrôleur gère les entrées utilisateur et mettre à jour la base de données si nécessaire

code HelloController.java:

```
import java.io.IOException;
import java.net.URL;
import java.sql.SQLException;
import java.util.ResourceBundle;

1 usage
public class HelloController implements Initializable {
    3 usages
    @FXML
    private TextField nom ;
    3 usages
    @FXML
    private TextField tele ;
    5 usages
    @FXML
    private TableView<Livreur> mytable ;
    2 usages
    @FXML
    private TableColumn<Livreur ,Long> col_id ;
    2 usages
    @FXML
    private TableColumn <Livreur ,String> col_nom ;
    2 usages
    @FXML
    private TableColumn <Livreur ,String> col_tele ;
    1 usage
    @FXML
```

liant les DAO et l'interface utilisateur

code HelloController.java:


```java
protected void onSaveButtonClick() {
    // accees a la bdd
    try {
        LivreurDAO livreurDAO = new LivreurDAO();
        Livreur liv = new Livreur( id_livreur: 01 , nom.getText() , tele.getText());
        livreurDAO.save(liv);
        UpdateTable();
    } catch (SQLException e) {throw new RuntimeException(e);}
}

1 usage
@FXML
protected void onUpdateButtonClick() {
    // access to database
    try {
        LivreurDAO livreurDAO = new LivreurDAO();

        // get the selected row from the table
        Livreur selectedLivreur = mytable.getSelectionModel().getSelectedItem();
        // update the object with the new values from the text fields
        selectedLivreur.setNom(nom.getText());
        selectedLivreur.setTelephone(tele.getText());
        // update the object in the database
        livreurDAO.update(selectedLivreur);
        // update the table with the new data
        UpdateTable();

    } catch (SQLException e) {
        throw new RuntimeException(e);
    }
}
```

calss test

la class test contient le méthode main pour faire des test sur les données , par exemple pour la class produit :

code test.java:

```java
package ma.fstt.model;
import java.sql.SQLException;
import java.util.List;
no usages
public class Test {
    no usages
    public static void main(String[] args) {
// trait bloc try catch
        try {

            ProduitDAO produitDAO = new ProduitDAO();

            Produit prod = new Produit( id_Produit: 0l , libele: "prodtest" ,
                    designation: "200000000", prix: "4dh");

            produitDAO.save(prod);

            List<Produit> livlist =  produitDAO.getAll();

            for (Produit liv :livlist) {

                System.out.println(liv.toString());

            }
        } catch (SQLException e) {

            throw new RuntimeException(e);

        }
    }
```

output:

```
Test
C:\Users\lenovo\.jdks\openjdk-20\bin\java.exe ...
Produit{id_produit =1, libele='huile', designation='CB8Y7', prix='20dh'}
Produit{id_produit =2, libele='prod3', designation='99878768', prix='25dh'}
Produit{id_produit =15, libele='danon', designation='786457', prix='2.5dh'}
Produit{id_produit =18, libele='thè', designation='CB8Y7', prix='20dh'}
Produit{id_produit =19, libele='prodtest', designation='200000000', prix='4dh'}

Process finished with exit code 0
```

La mise en place d'une application desktop en Java utilisant JDBC et JavaFX pour gérer les livreurs, les commandes et les produits est un projet ambitieux et techniquement complexe. En utilisant JDBC pour se connecter à une base de données MySQL, l'application est capable de stocker et de récupérer efficacement les données nécessaires pour gérer les livraisons, les commandes et les produits. En utilisant JavaFX, l'interface utilisateur de l'application est conçue de manière élégante et intuitive pour que les utilisateurs puissent facilement accéder aux fonctionnalités de l'application.

L'application offre une plate-forme centralisée pour la gestion des commandes, des produits et des livreurs, ce qui facilite la prise de décisions et le suivi des performances. Grâce à son tableau de bord générique, l'application permet aux utilisateurs de visualiser les données sous forme de graphiques et de tableaux, ce qui leur permet de prendre des décisions plus éclairées sur la gestion des commandes et des livraisons.

Dans l'ensemble, la mise en place de cette application est une réalisation significative qui a nécessité une planification minutieuse, une expertise technique et une compréhension approfondie des outils JavaFX, JDBC et MySQL. L'application résultante est capable de gérer efficacement les livreurs, les commandes et les produits, offrant une plate-forme centralisée pour la gestion des données et des analyses rapides pour la prise de décision.

Remerciements

je tiens à exprimer ma profonde gratitude envers Mr El Mokhtar EN-NAIMI professeur de module POO en Java et Mr ELAACHAk LOTFI pour leur dévouement et leurs efforts tout au long de ce semestre. Leur passion pour le sujet et leur engagement envers notre réussite ont été une source d'inspiration pour nous et pour l'ensemble de la classe.

Grâce à votre enseignement approfondi, j'ai pu acquérir une compréhension solide des principes de la POO en Java, ainsi que des compétences pratiques pour concevoir et développer des programmes orientés objet. Vos explications claires et vos exemples pertinents ont grandement contribué à ma réussite dans ce domaine.

Je suis également reconnaissant pour l'engagement de votre part à nous offrir des outils et des ressources supplémentaires pour approfondir nos connaissances, ainsi que pour votre disponibilité pour répondre à nos questions et nous offrir des conseils utiles.

En somme, je vous suis reconnaissant pour votre passion pour l'enseignement et votre contribution à mon parcours éducatif. Merci encore pour tout ce que vous avez fait pour nous aider à réussir dans ce module.

كلية العلوم و التقنيات بطنجة
ⵜⴰⵖⵍⵓⵍⵜ ⵏ ⵜⵣⴰⵡⵉⵏ ⴷ ⵜⵓⵙⵙⵔⵉⵜⵉ ⵏ ⵟⴰⵏⵊⴰ
Faculté des Sciences et Techniques de Tanger

Coordonnées

Mohammed AACHABI

mohammed.achabi@etu.uae.ac.ma

LST-GENIE INFORMATIQUE

Groupe : Grp2

CNE:N130119025