**DOCUMENTAȚIE**

**Tema numărul 1**

NUME STUDENT: Bureția Alex-Dian

GRUPA: 30228

**Cuprins**

1. Obiectivul temei..........................3
2. Analiza problemei, modelare, scenarii, cazuri de utilizare.......................................4
3. Proiectare...................................5
4. Implementare.............................7
5. Rezultate....................................11
6. Concluzii.....................................12
7. Bibliografie..................................12
8. **Obiectivul temei**

Obiectivul acestei teme este acela utiliza o baza de date pentru a realiza o aplicatie de comenzi. Astfel, un client poate comanda un anumit produs intr-o anumita cantitate si primeste un bon. Pentru a realiza aceast lucru am utilizat 3 baze de date( pentru clienti, produse si comenzi). In plus, un alt obiectiv, secundar , este de a scrie niste date intr-un fisier .txt si de a scrie comenzile SQL in I D E-ul java ,,IntelliJ”.

**2.Analiza problemei, modelare, scenarii, cazuri de utilizare**

Pentru a realiza aceasta am utilizat Layerded Architecture referitor la pachetele programului, deoarece am utiizat un pachet de BusinessLogic, unde am scris logica aplicatiei, un alt pachet de view-pentru design-ul vizual si pentru user interface-ul aplicatiei, in cadrul careia un client poate introduce date despre si el si poate selecta produse si introduce o anumita cantitate, dataAcces-pentru realizarea conexiunii cu baza de date, dar si pachetul model pentru a a defini un client, o comanda si un produs.

**3.Proiectare**

**4.Implementare**

In cadrul pachetului businessLogic, am utilizat o clasa abstracta care va fi mostenita de altele-AbstractDAO. Aceasta clasa, AbastractDAO, realizeaza un schelet pentru interogarile care vor fi utilizate mai apoi pentru a obtine anumite date, sau pentru a updata tabela de valori. In metoda insert am inserat valori in baza de date client, produs si order. In metoda edit, am editat aceste date, iar in metoda delete a m sters aceste date.

Clasele Client DAO, OrdersDAO si ProductsDAO mostenesc astfel clasa AbstractDAO, implementand pentru fiecare tabela din baza de date aceste op eratii.

In pache tul dataAccess am pus o singura clasa- Connection1. Aceasta clasa am ut ilizat-o pentru a realiza conexiunea la baza noastra de date creata in MYSQL Wo rkbench. Metodel e createConnection si getConnection cr eaza si ret urneaza conexiunea stabilita.

In pachetul model am scris 3 clase ( Client , Products , Orders ) .Aceste clase ca atribute numele de coloane ale tabelelor cu acelasi nume din SQL, acest lucru fiind utilizat pentru a putea realiza conexiunea in mod cor ect. Astfel, clientul e definit de un id(cheie primara), un nume si o adresa). Produsul e definit de un id(cheie primara) un nume , o cantitate si un pret. O comanda e definita atat de atributele unui client ( id, nume , adresa ) , cat si a unui produs (id, nume , cantitate , pret ) , iar in plus are si aceasta un id unic.

public void insert*(*T t*) {* String ss=insertInstruction*()*;  
 String vv=insertValues*()*;  
 Connection connection = null;  
 PreparedStatement statement = null;  
 String interogare= ss+ type.getSimpleName*()*+" " + vv;  
  
 try *{* for *(*Field field : type.getDeclaredFields*()) {* field.setAccessible*(*true*)*;  
 Object obj = field.get*(*t*)*;  
 if*(*obj instanceof String*){* interogare = interogare + " '" +obj.toString*()*+ "',";  
 *}*else*{* interogare = interogare + obj.toString*()* + ",";  
 *}  
 }* interogare = interogare.substring*(*0,interogare.length*()*-1*)*+ ")";  
 connection = Connection1.*getConnection()*;  
 statement = connection.prepareStatement*(*interogare*)*;  
 System.*out*.println*(*interogare*)*;  
 statement.execute*()*;  
 *}* catch *(*IllegalAccessException | SQLException e*){  
 LOGGER*.log*(*Level.*WARNING*, type.getName*()* + "DAO:insert " + e.getMessage*())*;  
 *}* finally *{* Connection1.*close(*statement*)*;  
 Connection1.*close(*connection*)*;  
 *}  
}*

public void update*(*T t*) {* String ss=updateInstruction*()*;  
 String vv=updateValues*()*;  
 Connection connection = null;  
 PreparedStatement statement = null;  
 String interogare = ss + type.getSimpleName*()*+" " + vv;  
 int id=0;  
  
 try *{* for *(*Field field : type.getDeclaredFields*()) {* field.setAccessible*(*true*)*;  
 Object obj = field.get*(*t*)*;  
 if*(*field.getName*()*.equals*(*"id"*)){* id=*(*int*)* obj;  
 *}*else*{* if*(*obj instanceof String*){* interogare = interogare + field.getName*()* + "='" + obj.toString*()* + "',";  
 *}*else*{* interogare = interogare + field.getName*()* + "=" + obj.toString*()* + ",";  
 *}  
 }  
 }* interogare = interogare.substring*(*0,interogare.length*()*-1*)*+ " ";  
 interogare += "WHERE id=?";  
 connection = Connection1.*getConnection()*;  
 statement = connection.prepareStatement*(*interogare*)*;  
 statement.setInt*(*1, id*)*;  
 statement.execute*()*;  
 *}* catch *(*IllegalAccessException | SQLException e*) {  
 LOGGER*.log*(*Level.*WARNING*, type.getName*()* + "DAO:update " + e.getMessage*())*;  
 *}*finally *{* Connection1.*close(*statement*)*;  
 Connection1.*close(*connection*)*;  
 *}  
}*

public void delete*(*T t*){* String ss=deleteInstruction*()*;  
 String vv=deleteValues*()*;  
 Connection connection = null;  
 PreparedStatement statement = null;  
 String interogare = ss + type.getSimpleName*()* +" "+ vv;  
 try *{* interogare = interogare +getId*(*t*)*;  
 connection = Connection1.*getConnection()*;  
 statement = connection.prepareStatement*(*interogare*)*;  
 statement.execute*()*;  
 *}* catch *(*IllegalAccessException | SQLException e*) {  
 LOGGER*.log*(*Level.*WARNING*, type.getName*()* + "DAO:delete " + e.getMessage*())*;  
 *}* finally *{* Connection1.*close(*statement*)*;  
 Connection1.*close(*connection*)*;  
 *}  
}*

package dataAccess;  
import java.sql.\*;  
import java.util.logging.Level;  
import java.util.logging.Logger;  
public class Connection1 *{* private static final Logger *LOGGER* = Logger.*getLogger(*Connection1.class.getName*())*;  
 private static final String *DRIVER* = "com.mysql.cj.jdbc.Driver";  
 private static final String *DBURL* = "jdbc:mysql://localhost:3306/proiect3(4)";  
 private static final String *USER* = "root";  
 private static final String *PASS* = "monedadan";  
 private static Connection1 *singleInstance* = new Connection1*()*;  
 private Connection1*() {* try *{* Class.*forName(DRIVER)*;  
 *}* catch *(*ClassNotFoundException e*) {* e.printStackTrace*()*;  
 *}  
 }* private Connection createConnection*() {* Connection connection = null;  
 try *{* connection = DriverManager.*getConnection(DBURL*, *USER*, *PASS)*;  
 *}* catch *(*SQLException e*) {  
 LOGGER*.log*(*Level.*WARNING*, "An error occured while trying to connect to the database"*)*;  
 e.printStackTrace*()*;  
 *}* return connection;  
 *}* public static Connection getConnection*() {* return *singleInstance*.createConnection*()*;  
 *}* public static void close*(*Connection connection*) {* if *(*connection != null*) {* try *{* connection.close*()*;  
 *}* catch *(*SQLException e*) {  
 LOGGER*.log*(*Level.*WARNING*, "An error occured while trying to close the connection"*)*;  
 *}  
 }  
 }* public static void close*(*Statement statement*) {* if *(*statement != null*) {* try *{* statement.close*()*;  
 *}* catch *(*SQLException e*) {  
 LOGGER*.log*(*Level.*WARNING*, "An error occured while trying to close the statement"*)*;  
 *}  
 }  
 }* public static void close*(*ResultSet resultSet*) {* if *(*resultSet != null*) {* try *{* resultSet.close*()*;  
 *}* catch *(*SQLException e*) {  
 LOGGER*.log*(*Level.*WARNING*, "An error occured while trying to close the ResultSet"*)*;  
 *}  
 }  
 }  
}*

Pachet ul view con tine int erfata gr afica a aplicati ei, prec u m si controlle rul. View realizeaza interfata grafica pentru tabe la de introducere a datelor unui client, ViewOrder realizeaza inter fata grafica pentru a legerea u nui client si a unui produs, iar ViewPro dus realizeaza interfat a grafica pent ru un produs. Clasa controller citeste toate datale din Jtext Are a, p recum si implementeaza Listener-urile pentru butoane le aplicat iei. But onul de afisare clienti, d eschide un nou Frame, in care inserea za un Jtabel, care a fost const ruit pe baza numelor coloa nelor si a datelor din tabel, aces t lucru fii nd re alizat atat pe ntru Cli ent cat si pen tru prod use. Am ut ilizat o ma trice de obiecte pe ntru a l ua da tele din b aza de dat e. Butonu l comanda din aceasta cla sa realiz aza scader e a cantitatii de pro duse , pe baza numarului introdus d e utilizator, dupa conecta rea l a baza de date produs pentru a vedea da ca d ispu nem de atatea produse. In plus, bonul este scris in fisierul fisier.txt, iar totalul este pe baza numarului de produse comandate ori pretul unei bucati.

package model;  
  
public class Orders *{* private int id;  
 private int idc;  
 private String numec;  
 private String address;  
 private int idp;  
 private String numep;  
 private int cantitate;  
 private int pret;  
 public Orders*(){  
  
 }* public void setId*(*int id*) {* this.id = id;  
 *}* public void setAddress*(*String address*) {* this.address = address;  
 *}* public void setPret*(*int pret*) {* this.pret = pret;  
 *}* public String getAddress*() {* return address;  
 *}* public int getPret*() {* return pret;  
 *}* public Client setClient*(*int idc, String numec, String adresa*) {* Client c=new Client*(*idc, numec,adresa*)*;return c;  
 *}* public Products setProdus*(*int idp, String numep, int cantitate,int pret*) {* Products p=new Products*(*idp,numep,cantitate, pret*)*;  
 return p;  
 *}* public void setIdc*(*int idc*) {* this.idc = idc;  
 *}* public void setNumec*(*String numec*) {* this.numec = numec;  
 *}* public void setIdp*(*int idp*) {* this.idp = idp;  
 *}* public void setNumep*(*String numep*) {* this.numep = numep;  
 *}* public void setCantitate*(*int cantitate*) {* this.cantitate = cantitate;  
 *}* public int getIdc*() {* return idc;  
 *}* public String getNumec*() {* return numec;  
 *}* public int getIdp*() {* return idp;  
 *}* public String getNumep*() {* return numep;  
 *}* public int getCantitate*() {* return cantitate;  
 *}* public int getId*() {* return id;  
 *}  
}*

class addVezi1Listener implements ActionListener*{* @Override  
 public void actionPerformed*(*ActionEvent e*){* Connection dbConnection = Connection1.*getConnection()*;  
 String updateStatementString = "SELECT \* FROM products";  
 PreparedStatement updateStatement = null;  
 Object*[][]* data=new Object*[*100*][*100*]*;  
 try *{* Statement statement =dbConnection.createStatement*()*;  
 try *{* updateStatement = dbConnection.prepareStatement*(*updateStatementString, Statement.*RETURN\_GENERATED\_KEYS)*;  
 int i=0;  
 ResultSet rs=statement.executeQuery*(*updateStatementString*)*;  
 while *(*rs.next*()) {*System.*out*.println*(*"aaa"*)*;  
 int id = rs.getInt*(*"id"*)*;  
 String numep = rs.getString*(*"numep"*)*;  
 int cantitate = rs.getInt*(*"cantitate"*)*;  
 int pret = rs.getInt*(*"pret"*)*;  
 data*[*i*][*0*]*=id;  
 data*[*i*][*1*]*=numep;  
 data*[*i*][*2*]*=cantitate;  
 data*[*i*][*3*]*=pret;i++;  
 *}  
 }* catch *(*SQLException ex*) {* System.*out*.println*(*"eroare"*)*;  
 *}* finally *{* Connection1.*close(*updateStatement*)*;  
 Connection1.*close(*dbConnection*)*;  
 *}  
 }* catch *(*SQLException ex*) {* ex.printStackTrace*()*;  
 *}* String*[]* columnNames=*{*"id","numep","cantitate","pret"*}*;  
 viewProdus.setTextJTable*(*data, columnNames*)*;  
  
 *}  
 }*class addComandaListener implements ActionListener*{* @Override  
 public void actionPerformed*(*ActionEvent e*){* //private static final String updateStatementString = "UPDATE client"  
 // + " set idc=?,numec=?,address=?"+"where idc=?";  
 Connection dbConnection = Connection1.*getConnection()*;  
 String updateStatementString = "SELECT pret,cantitate FROM products where numep =?";  
 String schimbareStatementString = "UPDATE products SET cantitate = ? WHERE numep =?";  
 PreparedStatement updateStatement = null;  
 PreparedStatement schimbareStatement=null;  
 Object*[][]* data=new Object*[*100*][*100*]*;  
 //Statement statement =dbConnection.createStatement();  
 // try {  
 // updateStatement = dbConnection.prepareStatement(updateStatementString, Statement.RETURN\_GENERATED\_KEYS);  
 // int i=0;  
 // ResultSet rs=statement.executeQuery(updateStatementString);  
 try *{* Statement statement =dbConnection.createStatement*()*;  
 try *{* updateStatement = dbConnection.prepareStatement*(*updateStatementString, Statement.*RETURN\_GENERATED\_KEYS)*;  
 updateStatement.setString*(*1, viewOrder.getTextNumeProdusOrdin*())*;  
 System.*out*.println*(*updateStatementString*)*;  
 // updateStatement=dbConnection.  
 updateStatement.executeQuery*()*;//aici e baiul  
 //ResultSet rs = updateStatement.getGeneratedKeys();  
 // ResultSet rs=updateStatement.executeUpdate();//are ceva bai aici  
 ResultSet rs=updateStatement.executeQuery*()*;  
  
 if*(*rs.next*()) {* int pr = rs.getInt*(*"pret"*)*;  
 int cantitate = rs.getInt*(*"cantitate"*)*;//aflam cantitatea  
  
 if *(*Integer.*parseInt(*viewOrder.getTextCantitateOrdin*())* < cantitate*) {*System.*out*.println*(*"ccc"*)*;  
 int g =cantitate- Integer.*parseInt(*viewOrder.getTextCantitateOrdin*())* ;  
 schimbareStatement = dbConnection.prepareStatement*(*schimbareStatementString, Statement.*RETURN\_GENERATED\_KEYS)*;  
 schimbareStatement.setInt*(*1, g*)*;  
 schimbareStatement.setString*(*2, viewOrder.getTextNumeProdusOrdin*())*;  
 // ResultSet rs1 = schimbareStatement.executeUpdate(schimbareStatementString);  
 int z=pr\*Integer.*parseInt(*viewOrder.getTextCantitateOrdin*())*;  
 schimbareStatement.executeUpdate*()*;  
 ResultSet rs1 = schimbareStatement.getGeneratedKeys*()*;  
 System.*out*.println*(*"fff"*)*;  
 FileWriter fw = null;  
 try *{* fw = new FileWriter*(*"fisier.txt"*)*;  
 fw.write*(*"Clientul " + viewOrder.getTextNumeClientOrdin*()* + " a cumparat " + viewOrder.getTextNumeProdusOrdin*()* +" in cantitate de " + viewOrder.getTextCantitateOrdin*()* + ". Totalul e:" + z+" lei"*)*;  
 fw.close*()*;  
 JOptionPane.*showMessageDialog(*viewOrder, "in cantitate de " + viewOrder.getTextCantitateOrdin*()* + ". Totalul e:" + z*)*;  
  
 *}* catch *(*IOException ioException*) {* ioException.printStackTrace*()*;  
 *}  
  
  
 }  
 }* else JOptionPane.*showMessageDialog(*viewOrder,"nu avem atata cantitate"*)*;  
  
  
 *}* catch*(*Exception em*) {* em.printStackTrace*(*System.*out)*;  
  
 *}* finally *{* Connection1.*close(*updateStatement*)*;  
 Connection1.*close(*dbConnection*)*;  
 *}  
 }* catch *(*SQLException ex*) {* ex.printStackTrace*()*;  
 *}  
 }  
}*

package view;  
import dataAccess.Connection1;  
  
import java.awt.EventQueue;  
  
import javax.swing.\*;  
import java.awt.Font;  
import java.awt.event.ActionListener;  
import java.sql.\*;  
  
public class ViewOrder extends JFrame*{* private JLabel selectareProdus,selectareClient,introducetiCantitate;  
 private JComboBox comboBox,comboBox\_1;  
 private JTextField textField;  
 private JButton comanda;  
 public ViewOrder*() {* this.setBounds*(*100, 100, 1173, 617*)*;  
 this.setDefaultCloseOperation*(*JFrame.*EXIT\_ON\_CLOSE)*;  
 this.getContentPane*()*.setLayout*(*null*)*;  
  
 selectareProdus = new JLabel*(*"Selectare produs"*)*;  
 selectareProdus.setFont*(*new Font*(*"Tahoma", Font.*PLAIN*, 16*))*;  
 selectareProdus.setBounds*(*28, 31, 134, 59*)*;  
 this.getContentPane*()*.add*(*selectareProdus*)*;  
  
 selectareClient = new JLabel*(*"Selectare client:"*)*;  
 selectareClient.setFont*(*new Font*(*"Tahoma", Font.*PLAIN*, 16*))*;  
 selectareClient.setBounds*(*432, 31, 127, 59*)*;  
 this.getContentPane*()*.add*(*selectareClient*)*;  
  
 introducetiCantitate = new JLabel*(*"Introduceti Cantitate:"*)*;  
 introducetiCantitate.setFont*(*new Font*(*"Tahoma", Font.*PLAIN*, 16*))*;  
 introducetiCantitate.setBounds*(*834, 31, 154, 59*)*;  
 this.getContentPane*()*.add*(*introducetiCantitate*)*;  
  
 comboBox = new JComboBox*()*;  
 comboBox.setBounds*(*28, 110, 154, 21*)*;  
 Connection dbConnection1 = Connection1.*getConnection()*;  
 String updateStatementString1 = "SELECT \* FROM products";  
 PreparedStatement updateStatement1 = null;  
 try *{* Statement statement =dbConnection1.createStatement*()*;  
 try *{* updateStatement1 = dbConnection1.prepareStatement*(*updateStatementString1, Statement.*RETURN\_GENERATED\_KEYS)*;  
 int i=0;  
 ResultSet rs=statement.executeQuery*(*updateStatementString1*)*;  
 while *(*rs.next*()) {*System.*out*.println*(*"aaa"*)*;  
 int id = rs.getInt*(*"id"*)*;  
 String numep = rs.getString*(*"numep"*)*;  
 int cantitate = rs.getInt*(*"cantitate"*)*;  
 int pret = rs.getInt*(*"pret"*)*;  
 comboBox.addItem*(*numep*)*;  
 *}  
 }* catch *(*SQLException ex*) {* System.*out*.println*(*"eroare"*)*;  
 *}* finally *{* Connection1.*close(*updateStatement1*)*;  
 Connection1.*close(*dbConnection1*)*;  
 *}  
 }* catch *(*SQLException ex*) {* ex.printStackTrace*()*;  
 *}* this.getContentPane*()*.add*(*comboBox*)*;  
  
 comboBox\_1 = new JComboBox*()*;  
 comboBox\_1.setBounds*(*405, 110, 154, 21*)*;  
 Connection dbConnection = Connection1.*getConnection()*;  
 String updateStatementString = "SELECT \* FROM client";  
 PreparedStatement updateStatement = null;  
 try *{* Statement statement =dbConnection.createStatement*()*;  
 try *{* updateStatement = dbConnection.prepareStatement*(*updateStatementString, Statement.*RETURN\_GENERATED\_KEYS)*;  
 // ResultSet rs = updateStatement.getGeneratedKeys();  
 int i=0;  
 ResultSet rs=statement.executeQuery*(*updateStatementString*)*;  
 while *(*rs.next*()) {*System.*out*.println*(*"aaa"*)*;  
 int id = rs.getInt*(*"id"*)*;  
 String numec1 = rs.getString*(*"numec"*)*;  
 String address = rs.getString*(*"address"*)*;  
 comboBox\_1.addItem*(*numec1*)*;  
 *}  
 }* catch *(*SQLException ex*) {* System.*out*.println*(*"eroare"*)*;  
 *}* finally *{* Connection1.*close(*updateStatement*)*;  
 Connection1.*close(*dbConnection*)*;  
 *}  
 }* catch *(*SQLException ex*) {* ex.printStackTrace*()*;  
 *}* this.getContentPane*()*.add*(*comboBox\_1*)*;  
  
 textField = new JTextField*()*;  
 textField.setBounds*(*863, 122, 96, 19*)*;  
 this.getContentPane*()*.add*(*textField*)*;  
 textField.setColumns*(*10*)*;  
  
 comanda = new JButton*(*"Comanda"*)*;  
 comanda.setFont*(*new Font*(*"Tahoma", Font.*PLAIN*, 16*))*;  
 comanda.setBounds*(*859, 229, 145, 54*)*;  
 this.getContentPane*()*.add*(*comanda*)*;  
 this.setVisible*(*true*)*;  
 *}* public String getTextNumeProdusOrdin*(){* return comboBox.getSelectedItem*()*.toString*()*;  
 *}* public String getTextNumeClientOrdin*(){* return comboBox\_1.getSelectedItem*()*.toString*()*;  
 *}* public String getTextCantitateOrdin*(){* return textField.getText*()*;  
 *}* public void addComandaListener*(*ActionListener action*) {* comanda.addActionListener*(*action*)*;  
 *}  
}*

**5. Rezultate**

Rezultatele sunt: obtinerea unei aplicatii care utilizeaza lucrul cu baze de date si coenzi SQL.

**6.Concluzii**

In cadrul acestui proiect am invatat sa realizez conectarea la baze de date, am invatat sa abstractizez o clasa astfel incat sa nu scriu aceleasi operatii de n ori, am invatat cum sa realizez o interfata grafica prietenoasa cu utilizatorul. Acest proiect este unul bun pentru o viitoare activitate in acest domeniu si m-a ajutat foarte mult.

**Bibliografie:**