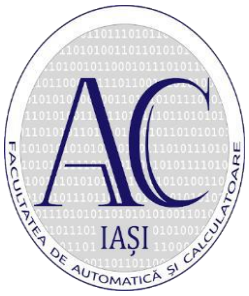


UNIVERSITATEA TEHNICĂ GHEORGHE ASACHI IAȘI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

SPECIALIZARE: CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI
DISCIPLINA: PROIECT BAZE DE DATE

Library-Internet-Cafe

Coordonator: Avram Sorin
Student: Fodor Maria-Gabriela
Grupa: 1308B
Iași, 2022

Baza de date aleasă va modela modul de funcționare a unei cafenele. Ideea principală a afacerii este aceea a unui loc unde studenții/elevii se pot întâlni pentru a studia/lucra împreuna la proiecte.

În cafenea se pot comanda atât băuturi (ceai, ciocolată caldă, etc.), cât și gustări. Cafeneaua dispune de o vastă bibliotecă, cu carți ce acoperă multiple domenii de interes.

Va fi gestionată problema meniului și a ingredientelor primare necesare pentru prepararea fiecarui item si crearea unei note de plata (bon).

****

Descrierea cerințelor și model de organizare al proiectului

Baza de date își propune gestionarea unei cafenele. Acest lucru îi va permite atat proprietarului, cât și angajaților o organizare mai ușoara la nivel administrativ. Principalele probleme aboradate sunt problema bonului, problema bibliotecii și problema meniului.

- Angajatul (chelnerul) va aduce cărțile clientului.
- Clientul poate solicita o carte dupa nume, autor sau gen.
- Angajatorul are posibilitatea să vizualizeze un tabel vizand toate cele 3 criterii.

- Angajatorul poate observa și informatii legate de cărțile aflate la alți clienți (în cazul în care o carte solicitată se prezinta cu numarul de bucati disponibile 0), se va putea afisa numatul toatal de carti din stoc, totusi.
- Pe bon vor aparea atât informații legate de cartea împrumutată, cat și legate de produsele consumate.
- Meniul și ingredientele necesare fiecărui preparat sunt gestionate tot de baza de date in tabelele aferente. (ITEM_MENIU, ITEM_INGREDIENT, INGREDIENT).
- La solicitatrea patronului, în meniu pot fi adăugate noi preparate.

Informațiile de care avem nevoie sunt cele legate de:

- **Cărți** : autor, titlu, genuri, număr de exemplare disponibile (neîmprumutate la momentul curent); în cazul suplimentării numărului de cărți acesta va fi incrementat în conformitate.

- **Item din Meniu** : cantitatea fiecărui ingredient pe care il contine și prețul acestuia.

- **Ingredient** : cantitatea disponibilă fiecărui ingredient și data de expirare.

- **Bon** : ce carte a citit fiecare client, ce a comandat, data de tăiere a bonului și dacă a fost sau nu platit (lucru care este de folos pentru gestionarea disponibilitații cărții).

Diagrama modelului logic

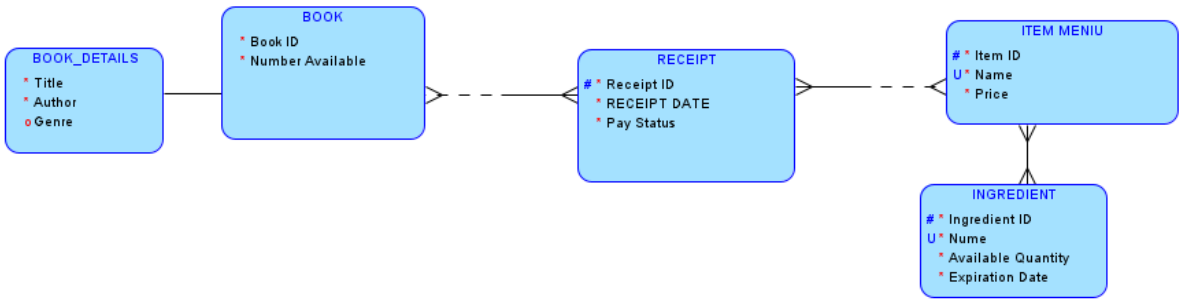
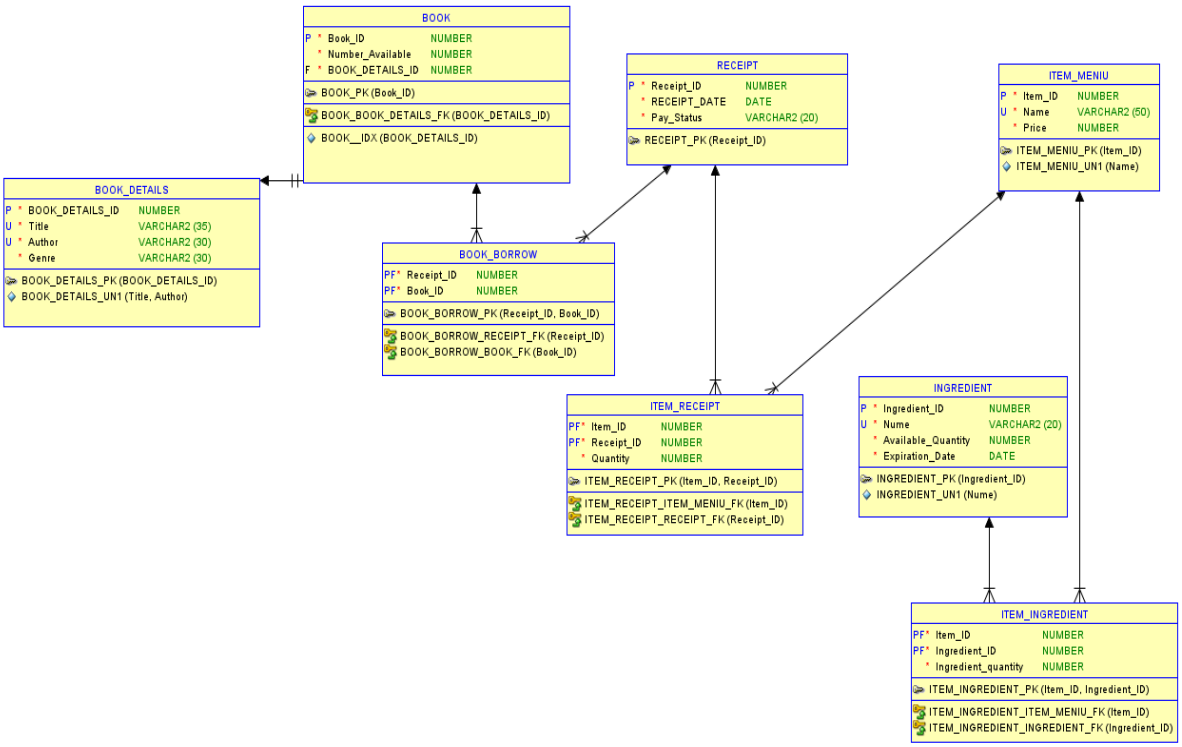


Diagrama modelului fizic



Normalizarea bazelor de date

Reprezintă procesul de organizare (fără a pierde din informații) a atributelor și tabelelor dintr-o bază de date relațională, cu scopul de a minimiza redundanța datelor și implicit de a minimiza potențialele erori care pot apărea în manipularea datelor.

A treia formă normală

➔ O relație este în a treia formă normală dacă satisface următoarele condiții: este în a doua formă normală și toate atributele non-cheie sunt direct (non-tranzitiv) dependente de TOATE cheile candidat.

În a 3-a forma normală se află următoarele tabele:

- Book(book_id -> book_details_id; book_id ->number_avalilable)
- Receipt (receipt_id -> receipt_date && pay_status)
- Book_borrow
- Item_receipt
- Item_meniu (item_id -> name; item_id -> price)
- Ingredient (Ingredient_id -> nume && Available_quantity && expiration_date)
- Item_ingredient

A doua formă normală

➔ O relație este în a doua formă normală dacă satisface următoarele condiții: este în prima formă normală și toate atributele non-cheie depind în totalitate de TOATE cheile candidat.

- Book_Details : Cheia primară a tabelei este <Book_Details_id>, iar toate atributele non-cheie depind in totalitate de ea.

Constrângerile folosite și justificările lor

Pentru fiecare tabelă avem implementate constrângeri de tip PK pentru a putea gestiona ID-uri și FK pentru a gestiona legăturile cu alte tabele.
Inafară de aceste constrângeri se mai impun următoarele restricții pentru a asigura coerența datelor.

1. **BOOK :**
    - Constrangere >= 0 pe Number_Available
2. **BOOK_DETAILS**
    - UK pe TITLE și AUTHOR nu pot avea 2 carți cu același titlu și același autor
    - Constrângere de lungime Check => 2 pe Titlu
    - Check pe campul Author regexp_like(Author, '^[A-Z]') pentru a verifica dacă începe cu liteă mare
    - Constrângere de lungime pe gen > 4
3. **ITEM_MENIU:**
    - UK pe NAME
    - Constrângere de lungime nume (>2) pe campul NUME
    - Constrangere preț > 0 pe câmpul PRICE
4. **RECEIPT:**
    - Constrâgere de apartenență la o lista de valori pe campul Pay_Status(Paid și Unpaid)
5. **INGREDIENT:**
    - Constrângere la insert pentru introducerea unei cantități pozitive
    - Constrângere de lungime pe câmpul NUME > 1
    - Constrângere de data de expirare > data curentă (implementată cu Trigger)
6. **ITEM_INGREDIENT:**
    - Constrangere cantitate > 0 pe câmpul INGREDIENT_QUANTITY
7. **ITEM_RECEIPT:**

- Constrangere cantitate > 0 pe câmpul QUANTITY

Tehnologii folosite

Partea de back-end a fost realizată utilizând limbajul Java. Pentru partea de front-end s-au folosit pachetul javax.swing. Acest pachet constituie un set de instrumente pentru widgeturi GUI pentru Java. Face parte din Oracle Java Foundation Classes – un API pentru furnizarea unei interfețe grafice de utilizator pentru programele Java.

Conexiunea la baza de date

Pentru conectarea la baza de date au fost necesare pachetul java.sql și driver-ele ojdbc11 și mysql-connecter.

Exemplu de cod conextare la baza de date:

```
try{
    Class.forName( className: "oracle.jdbc.driver.OracleDriver");
String url = "jdbc:oracle:thin:@//bd-dc.cs.tuiasi.ro:1539/orcl";
String user = "bd085";
String password = "bd085";
con = DriverManager.getConnection(url, user, password);
          *corp cod apelat pe durata conexiunii*
          } con.close();
}catch (Exception e){System.out.println( x: e);}
```

- Este apelata funcția .getConnection cu parametii URL, user și parola.
- Codul apelat pe durata conexiunii este cuprins într-un bloc try-catch (impus de posibilitatea unui eșec la conectarea cu baza de date) ;
- Totodata, blocul try-catch asigura continuitatea în execuție a programului în cazul unei posibile instrucțiuni sql eșuate(eroare la inserare sau delete etc.)
- Se utilizeaza pentru executarea de instrutiuni SQL clasele java.sql.Connection și java.sql.PreparedStatement.

Exemplu de apel interogare:

```
st = con.prepareStatement("SELECT b.book_id, d.title, d.author, d.genre, b.Number_available \n" +
"FROM book b, book_details d\n" +
"WHERE b.book_details_id = d.book_details_id"
+ " ORDER BY d.Title, d.author");//ordonate alfabetic
ResultSet rs = st.executeQuery();
```

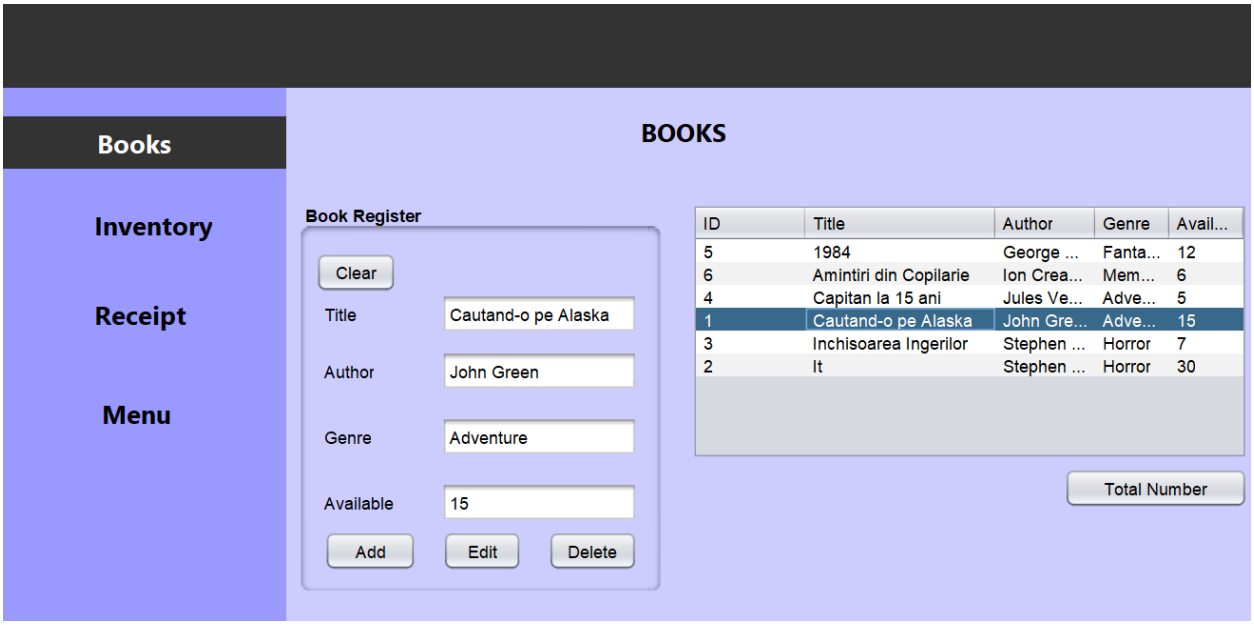
```
rs.getString( string: "book_id"));
rs.getString( string: "title"));
rs.getString( string: "author"));
rs.getString( string: "genre"));
rs.getString( string: "number_available"));
```

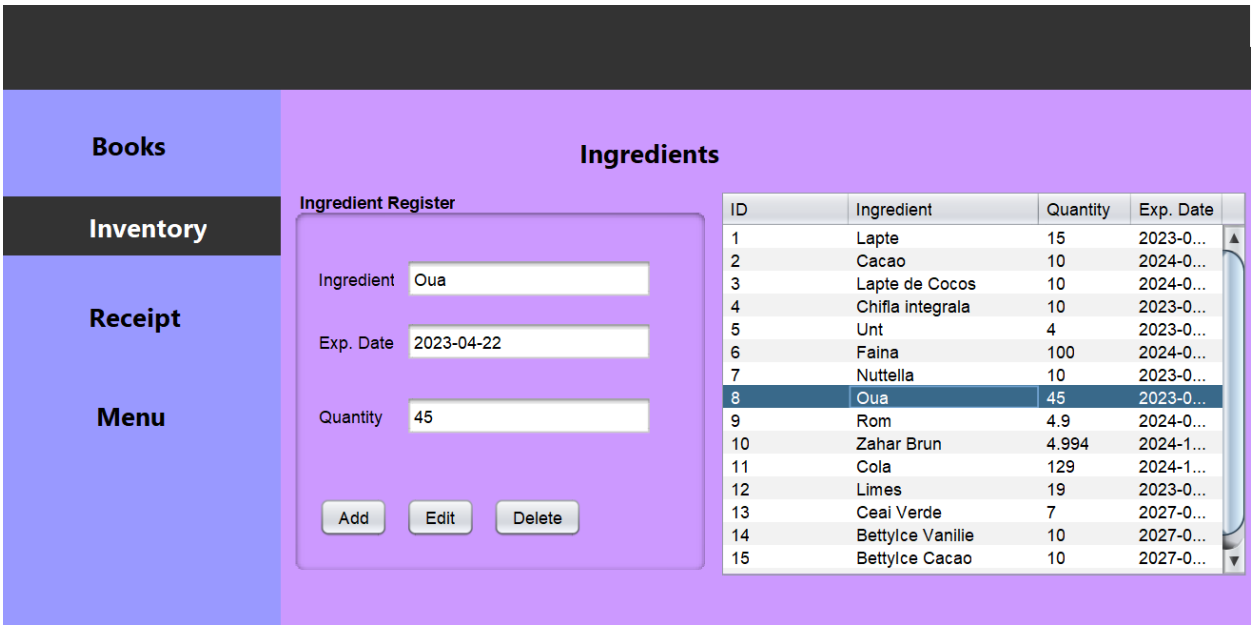
Exemplu de apel .executeUpdate() folosit la adaugarea unui nou preparat în meniu:

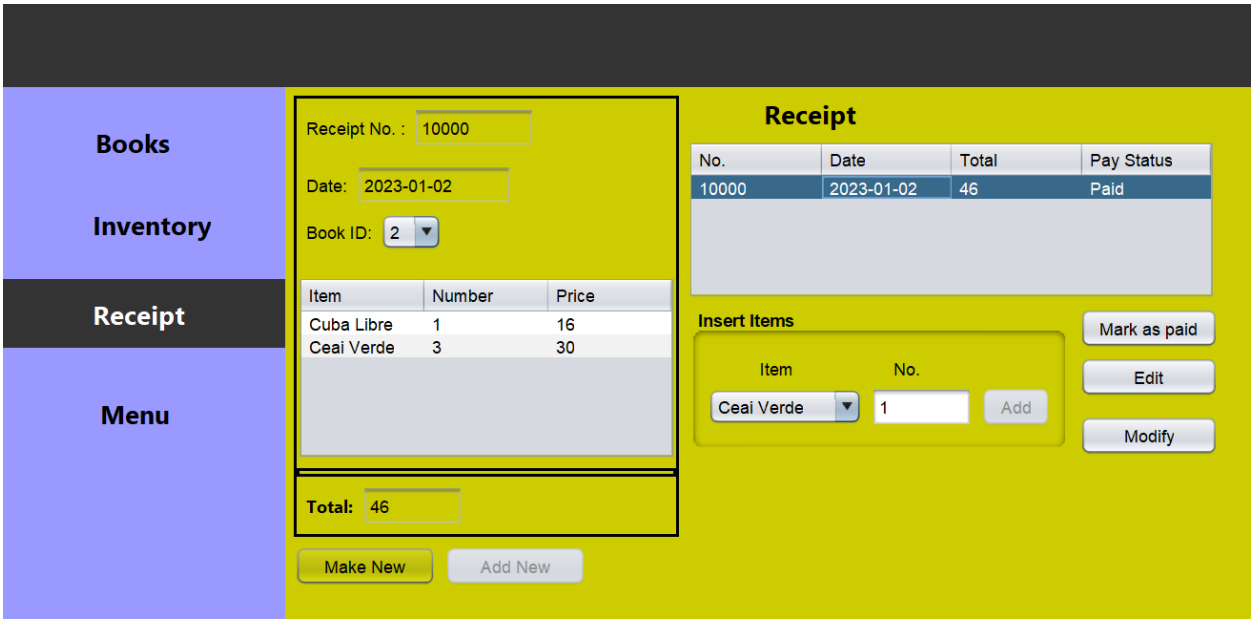
```
st = con.prepareStatement( string: "insert into ITEM_MENIU VALUES (null,?,?)");
st.setString( i: 1, string: name);
st.setString( i: 2, string: price);
st.executeUpdate();
```

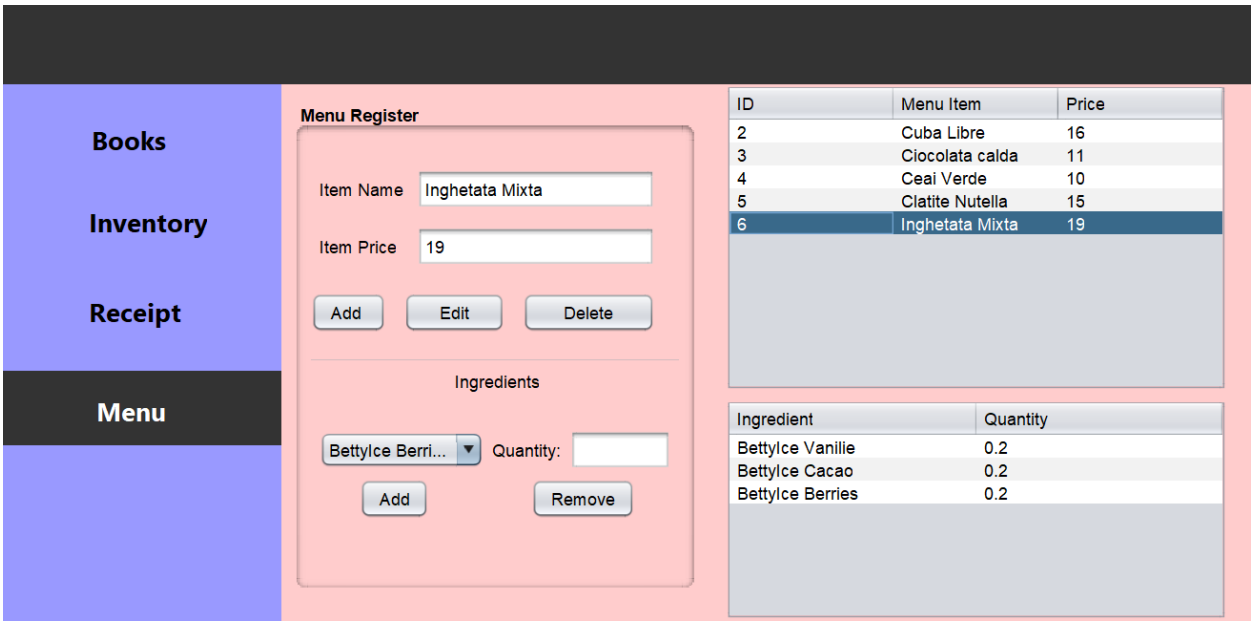
->set_string(al câtelea argument(?) din interogare , valoarea pe care o ia în interogare)

Interfața grafică




Books
Inventory
Receipt
Menu
Ingredients
Ingredient Register
Ingredient
Oua
Exp. Date
2023-04-22
Quantity
45
Add
Edit
Delete
ID
Ingredient
Quantity
Exp. Date
1 Lapte 15 2023-0...
2 Cacao 10 2024-0...
3 Lapte de Cocos 10 2024-0...
4 Chifla integrala 10 2023-0...
5 Unt 4 2023-0...
6 Faina 100 2024-0...
7 Nuttella 10 2023-0...
8 Oua 45 2023-0...
9 Rom 4.9 2024-0...
10 Zahar Brun 4.994 2024-1...
11 Cola 129 2024-1...
12 Limes 19 2023-0...
13 Ceai Verde 7 2027-0...
14 Bettylce Vanilie 10 2027-0...
15 Bettylce Cacao 10 2027-0...


Books
Inventory
Receipt
Menu
Receipt No. :
10000
Date:
2023-01-02
Book ID:
2
Item
Number
Price
Cuba Libre 1 16
Ceai Verde 3 30
Total:
46
Make New
Add New
Receipt
No.
Date
Total
Pay Status
10000 2023-01-02 46 Paid
Insert Items
Item
No.
Ceai Verde
1
Add
Mark as paid
Edit
Modify



1) **Înregistrarea unei cărți noi:** Completare câmpuri + Add

```
   String title = jTitle.getText();
   String author = jAuthor.getText();
   String genre = jGenre.getText();
   String number = jNumber.getText();
   //preluare date din interfata grafica
   //creare conexiune
if(title.length() > 1 && author.substring( beginIndex: 0,  endIndex: 1).matches( regex: "[A-Z]")
        && author.length() >=2 && genre.length() > 4 && number.length() > 0)
{
    try{
       Class.forName( className: "oracle.jdbc.driver.OracleDriver");
       String url = "jdbc:oracle:thin:@//bd-dc.cs.tuiasi.ro:1539/orcl";
       String user = "bd085";
       String password = "bd085";
       con = DriverManager.getConnection(url, user, password);
       st = con.prepareStatement("insert into  BOOK_DETAILS (book_details_id,Title,"
           + "Author,Genre) values(null,?,?,?)");//pe prima valoare avem auto increment
       st.setString( i: 1,  string: title);//conversie la intreg
       st.setString( i: 2,  string: author);
       st.setString( i: 3, string: genre);
       st.executeUpdate();
       //acum inseram in book
       //details_id va fi id_ul maxim(ultimul inserat din book_details)
       st = con.prepareStatement( string: "select max(book_details_id) maxi from book_details")
       ResultSet rs = st.executeQuery();
       rs.next();

       String det_id = rs.getString( string: "maxi");//extragerea book_details_id
       //inserarea in tablea books
       st = con.prepareStatement("INSERT into BOOK "
           + "(book_id,number_available,book_details_id) "
           + "values (null,?,?)");
       st.setString( i: 1,  string: number);
       st.setString( i: 2,  string: det_id);
       st.executeUpdate();
       //mesaj
       JOptionPane.showMessageDialog( parentComponent: this,  message: "Recorded");
```

2) **Modificarea numărului de cărți:** Selectare carte + Modificare Câmp Available + Edit

```
st = con.prepareStatement("Select b.book_id\n" +
"FROM book b, book_details d\n" +
"WHERE b.book_details_id = d.book_details_id\n" +
"AND d.title = ?"
+"AND d.author = ?");
st.setString( i: 1,  string: title);//conversie la intreg
st.setString( i: 2,  string: author);
ResultSet rs = st.executeQuery();
rs.next();

String book_id = rs.getString( string: "book_id");//extragerea book_id
//inserarea in tablea books
st = con.prepareStatement( string: "UPDATE BOOK SET number_available = ? WHERE book_id = ?");
st.setString( i: 1,  string: number);
st.setString( i: 2,  string: book_id);
st.executeUpdate();
//mesaj
JOptionPane.showMessageDialog( parentComponent: this,  message: "Recorded Update");
//si updatam tabelul
table_update();
//
jTitle.setText( t: "");
jAuthor.setText( t: "");
jGenre.setText( t: "");
jNumber.setText( t: "");//eliberare
//focusare cursor
jTitle.requestFocus();
```

3) **Delete Book :** Selectare carte + Delete

```
st = con.prepareStatement("Select b.book_id, b.book_details_id\n" +
"FROM book b, book_details d\n" +
"WHERE b.book_details_id = d.book_details_id\n" +
"AND d.title = ?"
+"AND d.author = ?");//pe prima valoare avem auto increment
st.setString( i: 1,  string: title);
st.setString( i: 2,  string: author);//avem id urile inregistrarilor de sters
ResultSet rs = st.executeQuery();
rs.next();

String det_id = rs.getString( string: "book_details_id");//extragerea book_details_id
String book_id = rs.getString( string: "book_id");//extagere carte
st = con.prepareStatement("DELETE FROM book "
        + "WHERE book_id = ?");
st.setString( i: 1,  string: book_id);
st.executeUpdate();
st = con.prepareStatement("DELETE FROM book_details "
        + "WHERE book_details_id = ?");
st.setString( i: 1,  string: det_id);
st.executeUpdate();
//au fost sterse detaliile
//mesaj
JOptionPane.showMessageDialog( parentComponent: this,  message: "Record Deleted");
//si updatam tabelul
table_update();
```

4) **Adaugarea unui item nou în meniu :** Completare Item Name și Item Price + Add (observăm ca apare în tabelă); după care începem să adăugăm ingrediente (selectăm ingredient, completăm cantitatea + Add);

```
st = con.prepareStatement( string: "insert into ITEM_MENIU VALUES (null,?,?)");//pe prima valoare avem auto increment
st.setString( i: 1,  string: name);
st.setString( i: 2, string: price);
st.executeUpdate();
//acum inseram in ingredient
JOptionPane.showMessageDialog( parentComponent: this,  message: "Item added");
```

La apăsarea Add (Item)

```
st = con.prepareStatement( string: "SELECT item_id FROM ITEM_MENIU WHERE name = ?");
st.setString( i: 1, string: item_name);
ResultSet rs = st.executeQuery();
rs.next();//va stoca oricum doar un item
String item_id = rs.getString( string: "item_id");//avem prima val
System.out.println("Item id "+ item_id);
//inerogare ingredient_id
st = con.prepareStatement( string: "SELECT ingredient_id FROM Ingredient WHERE nume = ?");
st.setString( i: 1, string: ingr_name);
rs = st.executeQuery();
rs.next();//va stoca oricum doar un item
String ingr_id = rs.getString( string: "ingredient_id");//avem prima val


st = con.prepareStatement( string: "insert into ITEM_INGREDIENT VALUES (?,?,?)");//pe prima valoare avem auto increment
st.setString( i: 1,  string: item_id);
st.setString( i: 2, string: ingr_id);
st.setDouble( i: 3, d: Double.parseDouble( s: quant));
```

La apsărea Add (ingredient)

La apăsarea butonului Delete(Item) se va realiza ștergerea din 2 tabele:

```
st = con.prepareStatement( string: "DELETE FROM ITEM_INGREDIENT WHERE item_id = ?");//pe prima valoare avem auto increment
st.setString( i: 1,  string: item_index);
st.executeUpdate();
//del la item_ingredient
st = con.prepareStatement( string: "DELETE FROM ITEM_MENIU WHERE item_id = ?");//pe prima valoare avem auto increment
st.setString( i: 1,  string: item_index);
st.executeUpdate();
//del din meniu
```

5) Receipt interface:

- Atunci când un client sosește își va alege o carte și va plas o comandă inițială (Unpayed). La această comandă se mai pot adauga consumații (în cazul în care se mai decide să se mai comande ceva). La final, clientul returnează cartea și plătește consumația totală.

```
st = con.prepareStatement("INSERT INTO RECEIPT "
        + "(receipt_id, receipt_date, Pay_Status)"
        + " VALUES (null,TO_DATE(?,'yyyy-mm-dd'),'Unpaid')");//pe prima valoare avem auto increment
st.setString( i: 1,  string: date);
st.executeUpdate();
```

```
st = con.prepareStatement( string: "SELECT max(receipt_id) mx FROM receipt");
ResultSet rs = st.executeQuery();
rs.next();

//acum inseram in Book_Borrow
            st = con.prepareStatement("INSERT INTO BOOK_BORROW"
        + "(receipt_id, book_id)"
        + " VALUES (?,?)"
        );//pe prima valoare avem auto increment
st.setDouble( i: 1,  d: Double.parseDouble( s: rs.getString( string: "mx")));
st.setDouble( i: 2, d: Double.parseDouble( s: book_id));
```

```
                //avem in item receipt
                //trebuie sa dam update la inventar
                st = con.prepareStatement("SELECT  ing.NUME , ? * i.INGREDIENT_QUANTITY quan\n" +
"    FROM INGREDIENT ing, ITEM_MENIU im, ITEM_INGREDIENT i\n" +
"    WHERE ing.INGREDIENT_ID = i.INGREDIENT_ID AND im.ITEM_ID = i.ITEM_ID AND im.item_id = ?");
                st.setInt(1, (int) Double.parseDouble((String) Df.getValueAt(r, 1)));//numarul de bucati
                st.setInt(2, (int) Double.parseDouble(rs1.getString("item_id" )));
                ResultSet rs2 = st.executeQuery();//avem nume ... ing_quan pt fiecare item din meiu
                //iteram si scadem din ingredient
                ResultSetMetaData Rss = rs2.getMetaData();
                int c = Rss.getColumnCount();
                while(rs2.next())
                {
                    for(int i = 1;i < c;i++)
                    {
                        //statement de update
                        st = con.prepareStatement("UPDATE INGREDIENT SET available_quantity = "
                                + "available_quantity - ? "
                                + "WHERE nume = ? ");
                        System.out.println("cantitate de scazutt: "+rs2.getString("quan"));
                        st.setString(2, rs2.getString("Nume"));
                        st.setDouble(1, Double.parseDouble(rs2.getString("quan")));
```

- Inserția unei comenzi : Make New + Add(Insert Item) + Add New(odată ce am finalizat comanda inițială)
- Adaugarea de consumație: Selectare + Edit + Add(Insert Item) + Modify (pt. validare)

  Pentru implementarea modificării inventarului la adaugarea de iteme s-a folosit un HashTable în care cheile reprezintă numărul liniei la care s-a gasit item-ul comandat (dacă se dorește să se comande înca o cafea sper exemplu).

  S-au implementat doua funcții apelate la apăsarea butonului Modify și Add new.

```
private int check_and_fix_distinct(String new_item,int nr)//metoda care verifica conditia de integritate in item_receipt
{//daca da adauga doar numarul
 //returneaza nr adaugat
    DefaultTableModel Df = (DefaultTableModel)RComanda.getModel();
    int rows = Df.getRowCount();//numarul de linii
    int linia = -1;
    for(int i = 0;i < rows;i++)
    {
        if ( Df.getValueAt(i, 0).equals(new_item))
        {
            linia = i;
            int val = (int)Double.parseDouble(Df.getValueAt(i, 1).toString()) + nr;
            Df.setValueAt(Integer.toString(val) , i, 1);
        }
    }
    return linia;
}//pt update ne folosim de nr
```

```
//pt modify doar:
row_quant_edit.put(line, Integer.parseInt(RBuc.getText()));//am adaugat linia
```

apel în Add

```
row_quant_edit.clear();//curatam tabela
```

apel în Edit

- Plata : Mark as paid (se reincrementează totodată numărul de cărți)

Tranzacțiile sunt următoarele :

1) Decrementarea câmpului Number_Available (din tabela BOOK) la inserarea în tablea RECEIPT și incrementarea la marcarea ca paid prin update la câmpul Pay_Status.

2) Scăderea stocului de ingrediente la efectuarea comenzilor.

La efectuarea de tranzacții s-au utilizat urmatoarele măsuri de siguranță în ceea ce privește succesul în efectuarea tuturor comenzilor SQL:

```
con.setAutoCommit( bln: false);//avem tranzactii
        *set de instrucțiuni ce alcătuiesc tranzacția*
con.commit();//daca nu iese din try pana aici atunci incheie tranzactia
 con.close();
```