DOCUMENTATIE

TEMA 3

NUME STUDENT: Pântea Marius Nicusor
GRUPA: 30222

CUPRINS

1. Obiectivul temei

Obiectivul principal al temei este de a dezvolta o aplicație de gestionare a comenzilor care să permită utilizatorului să gestioneze clienții, produsele, comenzile și facturile asociate acestora printr-o interfață grafică intuitivă.

Obiective secundare:

- Implementarea funcționalităților de vizualizare, adăugare, editare și ștergere a clienților, produselor
- Implementarea funcționalităților de adăugare si vizualizare a comenzilor
- Implementarea funcționalităților de inserare automata a facturilor si vizualizarea acestora
- Utilizarea reflexiei pentru creare câmpurilor de tabel si popularea acestora
- Utilizarea reflexiei pentru interacțiunea cu baza de date

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

Funcționale:

- Adăugarea, editarea și ștergerea clienților, produselor.
- Adăugarea comenzilor
- Crearea de comenzi asociate cu un client și produse disponibile.
- Generarea facturilor pentru comenzile create.

Non-funcționale:

- Interfața grafică intuitivă și ușor de utilizat.
- Eficiența operațiilor de gestionare a datelor.

Descrieri de use-case:

Adăugare client:

- Utilizatorul introduce datele noului client.
- Aplicația validează datele introduse.
- Datele clientului sunt adăugate în sistem.
- Se va afișa in tabel noul client.

Editare Client:

- Utilizatorul selectează un client din listă.
- Utilizatorul modifică datele clientului(se introduc in câmpuri).

- Aplicația validează modificările.
- Datele clientului sunt actualizate în sistem.
- Se va afișa tabela actualizata.

Ștergere Client:
- Utilizatorul selectează un client din listă.
- Se scrie si id-ul lui in căsuța de id
- Confirmarea ștergerii clientului.
- Ștergerea clientului din sistem.
- Se va afișa tabela actualizata.

Adăugare Produs:
- Utilizatorul introduce datele noului produs.
- Aplicația validează datele introduse.
- Datele produsului sunt adăugate în sistem.
- Se va afișa in tabel noul produs.

Editare Produs:
- Utilizatorul selectează un produs din listă.
- Utilizatorul modifică datele produsului (se introduc in câmpuri).
- Aplicația validează modificările.
- Datele produsului sunt actualizate în sistem.
- Se va afișa tabela actualizata.

Ștergere Produs:
- Utilizatorul selectează un produs din listă.
- Se scrie si id-ul lui in căsuța de id.
- Confirmarea ștergerii produsului.
- Ștergerea produsului din sistem.
- Se va afișa tabela actualizata.

Creare Comandă:
- Utilizatorul selectează un client și produsul dorit .
- Introducerea cantității pentru produs.
- Aplicația validează comanda.
- Generarea comenzii și asocierea acesteia cu clientul și produsele selectate.
- Se va afișa tabela actualizata.

Generare Factură:
- Utilizatorul selectează o comandă din listă.

- Generarea automată a facturii pentru comanda selectată.
- Afișarea facturii generate.
- Factura se generează si automat când se face o comanda

3. Proiectare

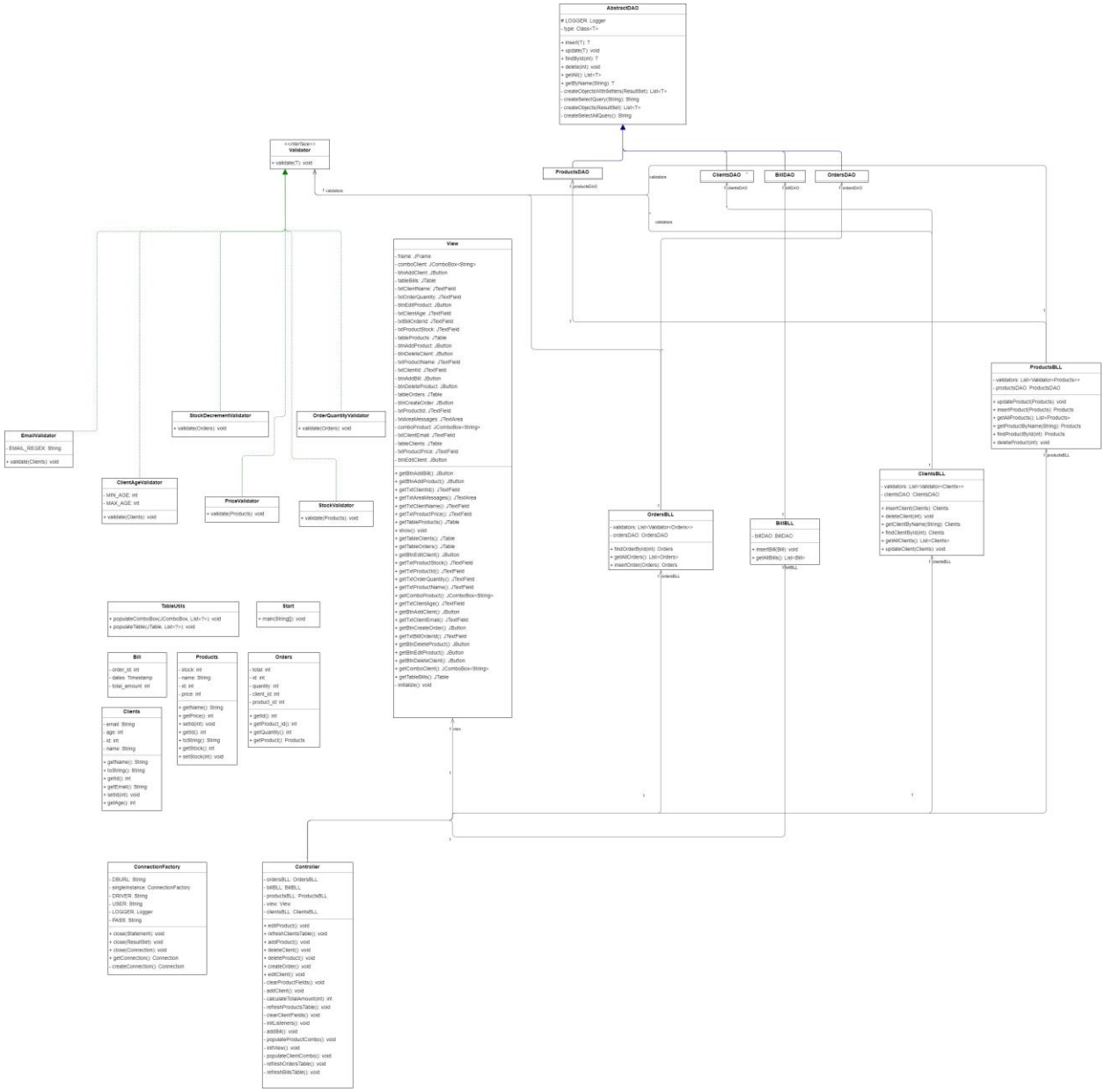


Diagrama UML

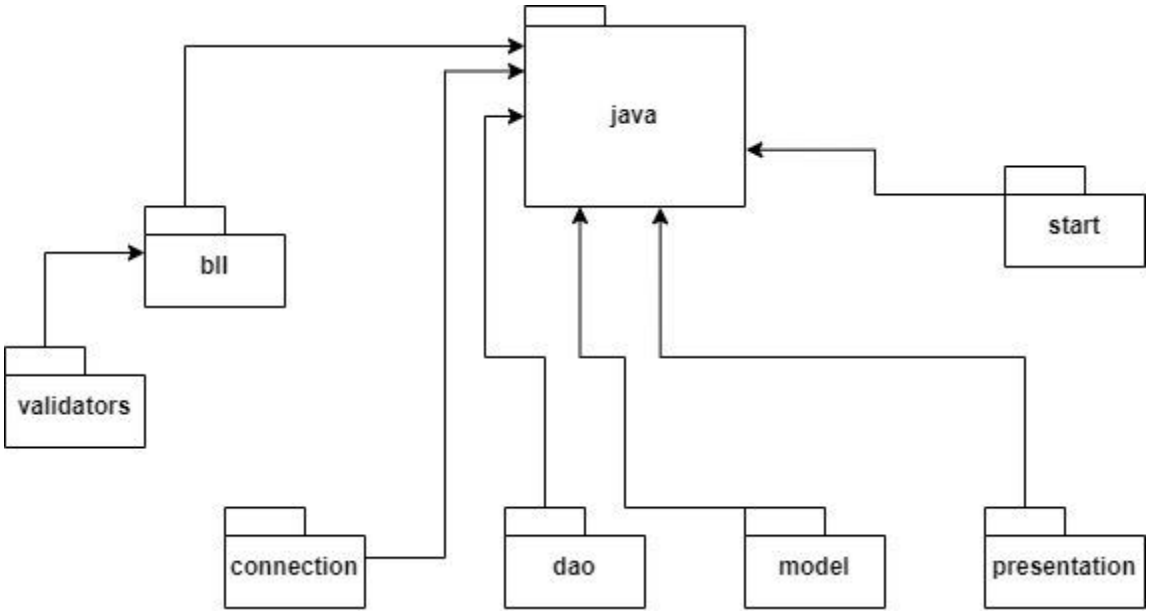


Diagrama de pachete

4. Implementare

Pachetul bll.validators:

- ClientAgeValidator: Validator pentru verificarea varstei unui client
- EmailValidator: Validator pentru verificarea formatului unei adrese de email
- OrderQuantityValidator: Validator pentru verificarea cantității de produse dintr-o comanda
- PriceValidator: Validator pentru verificarea prețului unui produs
- StockDecrementValidator: Validator pentru verificarea decrementului stocului in urma unei comenzi
- StockValidator:Validator pentru verificarea stocului unui produs

Pachetul bll:

- BillBLL: Clasa de logica pentru manipularea facturilor
  - public void insertBill(Bill bill)
  - public List<Bill> getAllBills()
- ClientsBLL: Clasa de logica pentru manipularea clienților
  - public Clients findClientById(int id)
  - public Clients insertClient(Clients client)
  - public void updateClient(Clients client)
  - public void deleteClient(int clientId)
  - public List<Clients> getAllClients()
  - public Clients getClientByName(String name)
- OrdersBLL: Clasa de logica pentru manipularea comenzilor
  - public Orders findOrderById(int id)
  - public Orders insertOrder(Orders order)

- public List<Orders> getAllOrders()

- ProductsBLL: Clasa de logica pentru manipularea produselor
  - public Products findProductById(int id)
  - public Products insertProduct(Products product)
  - public void updateProduct(Products product)
  - public void deleteProduct(int productId)
  - public List<Products> getAllProducts()
  - public Products getProductByName(String name)

Pachetul connection:

- ConnectionFactory

Pachetul dao:

- AbstractDAO: Clasa abstracta pentru operațiile de baza de acces la date private String createSelectQuery(String field)-Creeaza o interogare SELECT pentru un anumit câmp
  - private String createSelectAllQuery()-Creeaza o interogare SELECT pentru a obține toate înregistrările din tabel
  - public T findById(int id)-Extrage entitatea din baza de date folosind interogarea SELECT generata de createSelectQuery()
  - private List<T> createObjects(ResultSet resultSet)-Converteste rezultatul unei interogari SELECT într-o lista de obiecte de tipul generic T
  - private List<T> createObjectsWithSetters(ResultSet resultSet)-Similara cu createObjects(), dar utilizează metodele setter ale obiectului pentru a seta valorile câmpurilor, în loc să utilizeze constructorii
  - public T insert(T t)- Inserează o noua înregistrare în baza de date, utilizând valorile câmpurilor obiectului generic T dat ca parametru
  - public void delete(int id)-Șterge un obiect din baza de date după ID-ul sau

- public void update(T t)-Actualizează un obiect în baza de date
- public T getByName(String name)-Găsește un obiect după numele sau
- public List<T> getAll()-Returnează toate înregistrările din tabel sub forma de lista

- ClientsDAO-Clasa pentru operațiile specifice de acces la date pentru obiecte de tipul Clients
- BillDAO-Clasa pentru operațiile specifice de acces la date pentru obiecte de tipul Bill
- OrdersDAO-Clasa pentru operațiile specifice de acces la date pentru obiecte de tipul Orders
- ProductsDAO-Clasa pentru operațiile specifice de acces la date pentru obiecte de tipul Products

Pachet model:

- Bill-Clasa care reprezintă factura
- Clients-Clasa care reprezintă un client
- Orders-Clasa care reprezintă o comanda
- Products-Clasa care reprezintă un produs

Pachet presentation:

- Controller-Clasa Controller reprezintă controlerul aplicației, care gestionează interacțiunea dintre model si view
  - private void initView()
  - private void initListeners()
  - public void refreshClientsTable()
  - private void refreshProductsTable()
  - private void refreshOrdersTable()
  - private void refreshBillsTable()
  - private void populateClientCombo()

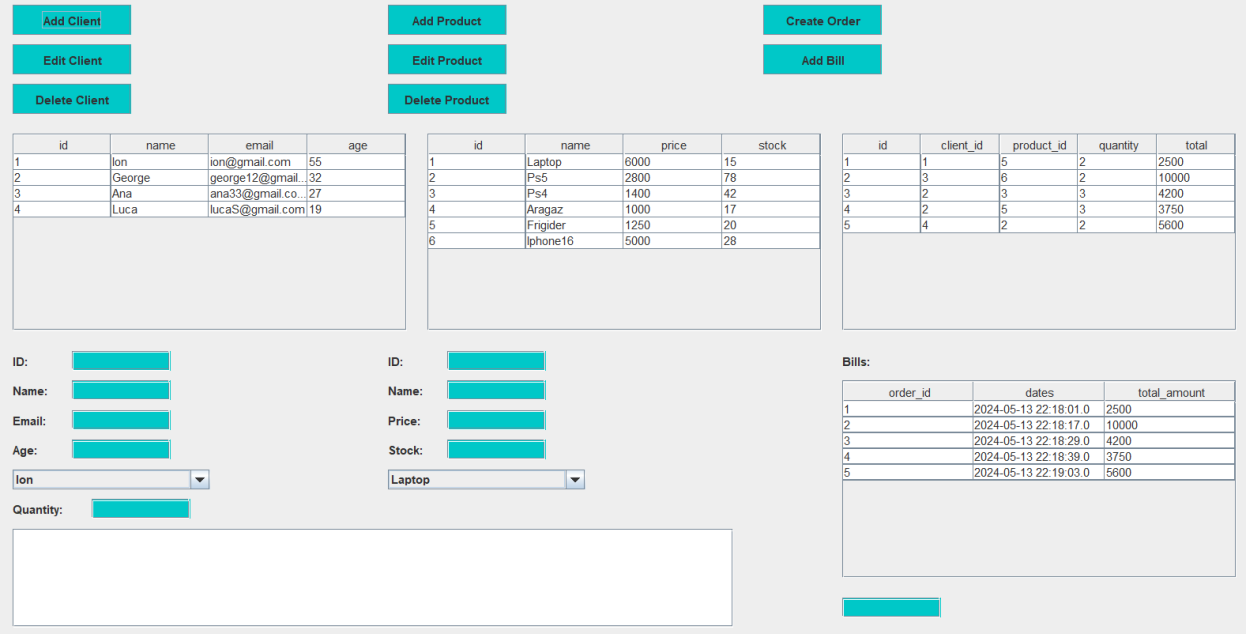
- private void populateProductCombo()
- private void clearProductFields()
- private void clearClientFields()
- private void addClient()
- public void editClient()
- public void deleteClient()
- public void addProduct()
- public void deleteProduct()
- public void createOrder() throws SQLException
- private void addBill()
- private int calculateTotalAmount(int orderId) throws SQLException

- View-Clasa View reprezintă interfața grafica a aplicației
  - private void initialize()

Pachetul view:

- TableUtils-Clasa pentru crearea capurilor de tabele si popularea acestora si pentru popularea combobox-urilor
  - public static void populateTable(JTable table, List<?> objectList)-Populează un tabel cu datele dintr-o lista de obiecte
  - public static void populateComboBox(JComboBox comboBox, List<?> objectList)-Populează un combobox cu elementele dintr-o lista de obiecte
- Start-are metoda main(Metoda principala a aplicației)

5. Rezultate



S-a testat prin inserarea clienților ,produselor , si creare de comenzi. Facturile s-au creat automat ,odată cu efectuarea comenzii.

6. Concluzii

În urma implementării temei, s-a observat importanța unei structuri bine definite și a modularității în dezvoltarea unei aplicații. Utilizarea design patterns și a principiilor de OOP a condus la un cod mai curat și mai ușor de întreținut. Dezvoltarea acestei aplicații a oferit o mai bună înțelegere a conceptelor de gestionare a datelor și a interacțiunii cu o interfață grafică în Java.

7. Bibliografie

https://dsrl.eu/courses/pt/materials/PT2024_A3_S1.pdf

https://dsrl.eu/courses/pt/materials/PT2024_A3_S2.pdf

https://stackoverflow.com/

https://www.baeldung.com/javadoc

https://dev.mysql.com/doc/workbench/en/wb-admin-export-import-management.html

http://tutorials.jenkov.com/java-reflection/index.html

https://www.baeldung.com/java-jdbc