DOCUMENTATIE

TEMA 3

NUME STUDENT: PiticTudor

GRUPA: 30245

# CUPRINS

[1. Obiectivul temei 3](#_Toc95297885)

[2. Analiza problemei, modelare, scenarii, cazuri de utilizare 3](#_Toc95297886)

[3. Proiectare 3](#_Toc95297887)

[4. Implementare 3](#_Toc95297888)

[5. Rezultate 3](#_Toc95297889)

[6. Concluzii 3](#_Toc95297890)

[7. Bibliografie 3](#_Toc95297891)

# Obiectivul temei

Obiectivul acestei sarcini este de a proiecta și dezvolta o aplicație pentru gestionarea comenzilor clienților într-un depozit folosind tehnici fundamentale de programare. Aplicația trebuie să fie proiectată conform modelului de arhitectură stratificată și să utilizeze principii de programare orientată pe obiecte. Aplicația trebuie să folosească baze de date relaționale pentru stocarea datelor și trebuie create cel puțin trei tabele: Client, Produs și Comandă. Aplicația trebuie să includă o interfață grafică de utilizator cu ferestre pentru operațiunile client și produse, precum și pentru crearea comenzilor de produse.

Atribuirea necesită utilizarea diferitelor tehnici de programare, cum ar fi tehnici de reflecție, pentru a crea clase generice pentru accesarea bazei de date și pentru generarea dinamică de interogări pentru anumite obiecte. În plus, o clasă Bill trebuie să fie definită în pachetul Model folosind înregistrări Java, care sunt imuabile și generează un obiect factura pentru fiecare comandă care este stocată într-un tabel *Log*.

# Analiza problemei, modelare, scenarii, cazuri de utilizare

Cerințele funcționale ale proiectului sunt:

1. Limitarea lungimii claselor la maximum 300 de linii și lungimea metodelor la maximum 30 de linii.
2. Urmați convențiile de denumire Java.
3. Utilizați Java Doc pentru documentarea claselor și generați fișierele Java Doc corespunzătoare.
4. Folosiți baze de date relaționale pentru stocarea datelor pentru aplicație.
5. Minim trei tabele: Client, Produs și Comanda.
6. Crearea unei interfețe care să includă:
   1. O fereastră pentru operațiunile client: adăugați un client nou, editați clientul, ștergeți clientul, vizualizați toți clienții într-un tabel (JTable).
   2. O fereastră pentru operațiunile produsului: adăugați un produs nou, editați produsul, ștergeți produsul, vizualizați toate produsele într-un tabel (JTable).
   3. O fereastră pentru crearea comenzilor de produse:
      1. Utilizatorul va putea să selecteze un produs existent, să selecteze un client existent și să introducă o cantitate dorită pentru produs pentru a crea o comandă validă.
      2. În cazul în care nu sunt suficiente produse, va fi afișat un mesaj de stoc insuficient.
      3. După finalizarea comenzii, stocul de produse este diminuat.
7. Folosiți tehnici de reflectare pentru a crea o metodă care primește o listă de obiecte și generează antetul tabelului prin extragerea prin reflectare a proprietăților obiectului și apoi populează tabelul cu valorile elementelor din listă.
8. Creați o clasă Bill imuabilă în pachetul Model folosind înregistrări Java.
9. Un obiect Bill va fi generat pentru fiecare comandă și va fi stocat într-un tabel Log.
10. Facturile pot fi introduse și citite numai din tabelul Log; nu sunt permise actualizări.
11. Utilizați tehnici de reflecție pentru a crea o clasă generică care conține metodele de accesare a DB (toate tabelele cu excepția Log):
    1. Creați obiect.
    2. Editați obiectul.
    3. Ștergeți obiectul.
    4. Găsiți obiect.
12. Interogările pentru accesarea DB pentru un anumit obiect care corespunde unui tabel vor fi generate dinamic prin reflecție.
13. Urmați modelul Layered Architecture cu cel puțin patru pachete: dataAccessLayer, businessLayer, model și prezentare.

# Proiectare

Ideea din spatele proiectării a acestei aplicații a fost de a folosi pachete si clase care descriu cat mai bine funcționalitatea părților din aplicație, dar in același timp pentru a păstra o eficienta in scrierea de cod. Din acest motiv, aplicația conține cinci pachete: *model*, *presentation*, *service*, *dao* și *connection*.

Pentru a evita scrierea decod boiler plate, s-a folosit dependenta de biblioteca „Lombok” pentru a scurta scrierea de constructori, metode getter si metode setter.

*Se va prezenta proiectarea OOP a aplicatiei, diagramele UML de clase si de pachete, structurile de date folosite, interfetele definite si algoritmii folositi (*daca e cazul)

# Implementare

*Se va descrie fiecare clasa cu campuri si metodele importante. Se va descrie implementarea interfetei utilizator.*

# Rezultate

*Se vor prezenta scenariile pentru testare. In cazul in care in cerinta temei se specifica sa se faca testare unitara cu utilitarul JUnit, se vor integra in acesta sectiune rezultate testari cu JUnit.*

# Concluzii

*Se vor prezenta concluziile, ce s-a invatat din tema, posibile de dezvoltari ulterioare.*

# Bibliografie

*Se vor adauga referintele care au fost consultate de student pe parcursul implementarii temei .*

*Exemplu:*

1. *Bruce Eckel, Thinking in Java (4th Edition), Publisher: Prentice Hall PTRUpper Saddle River, NJUnited States, ISBN:978-0-13-187248-6 Published:01 December 2005.*
2. *What are Java classes? -* [*www.tutorialspoint.com*](http://www.tutorialspoint.com)