

# DOCUMENTATIE

## TEMA 3

NUME STUDENT: RUS ANA-MARIA CARINA  
GRUPA: 30226

## CUPRINS

Obiectivul temei.....	3
Analiza problemei, modelare, scenarii, cazuri de utilizare .....	3
Proiectare .....	5
Implementare .....	7
Rezultate .....	10
Concluzii.....	10
Bibliografie... ..	11

# 1. OBIECTIVUL TEMEI

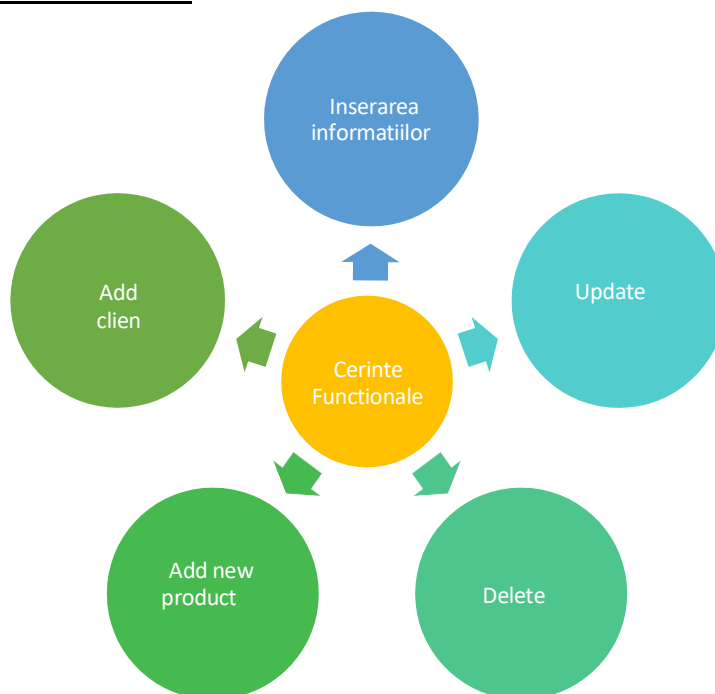
Obiectivul temei este de a crea un proiect Orders Management pentru procesarea comenzilor clientilor.

## OBIECTE SECUNDARE:

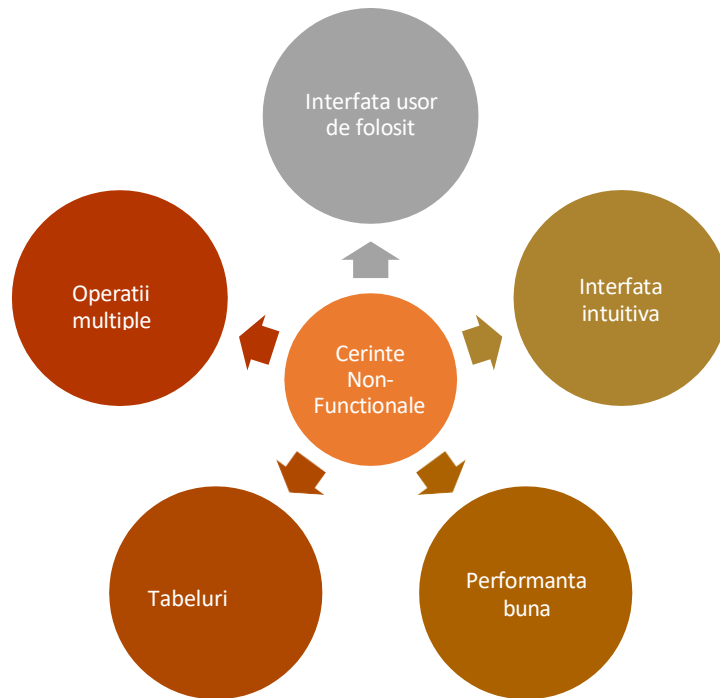
1. Intelegerea problemei
2. Crearea unei interfete potrivite pentru introducerea informatiilor
3. Implementarea aplicatiei
4. Testarea aplicatiei

# 2. ANALIZA PROBLEMEI, MODELARE, SCENARII, CAZURILE DE UTILIZARE

## Cerinte functionale



## Cerinte non-functionale



## User-Case

**User Case:** adauga produs

**Actor primar:** User-ul

**Scenariu pentru succes:**

1. User-ul selecteaza tabelul Products
2. User-ul selecteaza butonul pentru Add
3. Aplicatia afiseaza care detalii trebuie inserate
4. User-ul insereaza detaliile si apasa pe Add
5. Aplicatia salveaza produsul in baza de date si afiseaza tabelul de produse impreuna cu noua valoare

**Secventa alternatica:** Valori invalide pentru datele produsului

- User-ul introduce valori invalide
- Aplicatia afiseaza un mesaj care informeaza useu-ul ca valorile sunt invalide
- Scenariul se reintoarce la punctul 1.

### 3. PROIECTARE

Diagrama Main

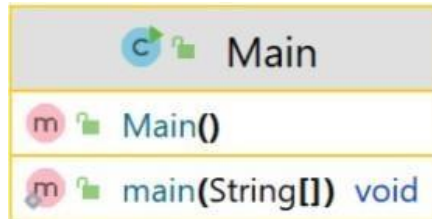


Diagrama ControllerView

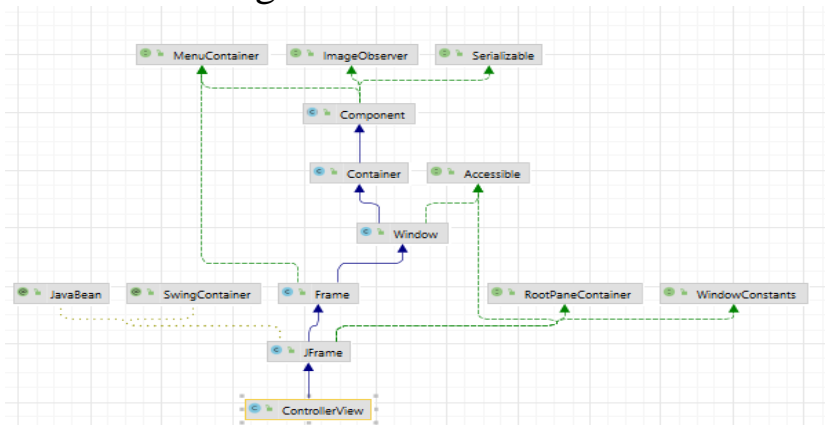
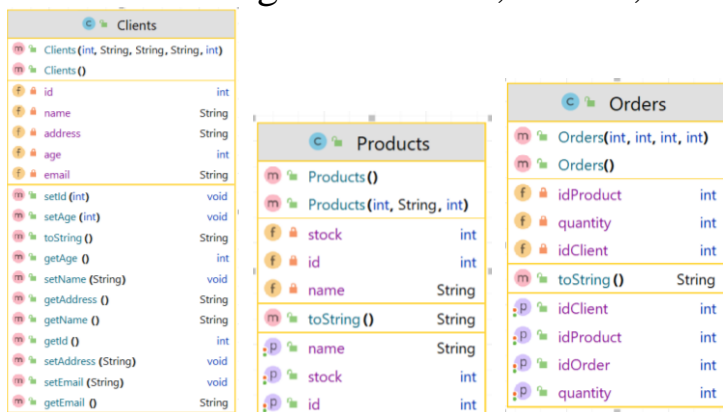


Diagrama Clients, Orders, Products



## Diagrama AbstractDAO

AbstractDAO<T>		
AbstractDAO()		
createObjects(ResultSet)	List<T>	
delete(int)	void	
createSelectQuery()	String	
createInsertQuery(ArrayList<String>)	String	
findAll()	List<T>	
insertOrder(int, int, int, Products)	void	
createSelectWhereQuery(String)	String	
findById(int)	T	
getValuesFromT(T, ArrayList<Object>, ArrayList<String>)	void	
createUpdateQuery(ArrayList<String>)	String	
insert(T)	void	
update(T, int)	void	
createDeleteQuery(String)	String	

## Diagrama ClientsBLL, OrdersBLL, ProductsBLL

ClientsBLL		
ClientsBLL()		
findById(int)	Clients	
insertClient(Clients)	void	
deleteClient(int)	void	
updateClient(Clients, int)	void	
findAll()	List<Clients>	

OrdersBLL		
OrdersBLL()		
findAll()	List<Orders>	
findById(int)	Orders	
insertOrder(Orders)	void	
placeOrder(int, int, int)	void	

ProductsBLL		
ProductsBLL()		
findById(int)	Products	
findAll()	List<Products>	
insertProduct(Products)	void	
deleteProduct(int)	void	
updateProduct(Products, int)	void	

Am folosit 6 pachete:

1. Org-example
2. Presentation
3. Model
4. Dao
5. Connection
6. BLL

## 7. IMPLEMENTARE

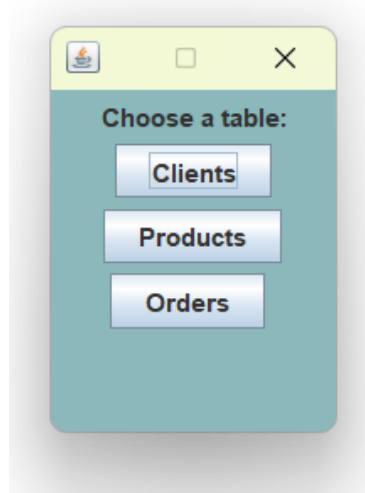
### 1. Clasa Main

In clasa Main am initializat interfata ControllerView.

### 2. Clasa ControllerView

In clasa CalcView am creat interfata folosind butoane, text fields, panel-ului si frame-uri pentru fiecare tabel si operatie.

Am folosit ActionListener pentru a implementa operatiile facute de fiecare buton.



### 3. Clasele Clients, Orders, Products

Aceste clase definesc obiectele de tipul clienti, produse si comenzi fiecare cu id-ul respectiv si detaliile cerute. Aici sunt definite gettere si settere.

#### 4. Clasele AbstractDAO, ClientsDAO, OrdersDAO, ProductsDAO

Clasele ClientsDAO, OrdersDAO, ProductsDAO extind clasa AbstractDAO care contine query-urile pentru select, insert, delete si operatiile pentru find all si find by id. La clasele DAO se foloseste reflexia.

#### 5. Clasa ConnectionFactory

Clasa ConnectionFactory face legatura cu baza de date.

#### 6. Clasele ClientsBLL, OrdersBLL, ProductsBLL

Clasele acestea fac legatura dintre interfata si clasele DAO.



## 7. REZULTATE

User-ul poate alege tabelul pe care vrea să facă operații și operațiile dorite după care este afișat rezultatul în urma operațiilor.

The screenshot displays three windows from a web application:

- Choose a table:** A window with three buttons: **Clients**, **Products**, and **Orders**.
- Choose an operation:** A window with five buttons: **Add**, **Edit**, **Delete**, **ViewAll**, and **ViewOne**.
- Orders Management:** A window containing a table of client data and a form for adding new clients.

The **Orders Management** table contains the following data:

Id	Name	Address	Email	Age
2	anaa	izlazlui	anananana@gmail.com	20
3	mihai	mihail	maihaaaa@gmail.com	25
4	alin	str scolii	alinalin20@gmail.com	23
6	cata	eroilor	cataaa@gmail.com	80
7	aurel	izlazului	aurelian@gmail.com	54
8	aurel	izlazului	aurelian@gmail.com	54

The **Orders Management** form includes the following fields:

- Id:** 6
- Name:** aurel
- Address:** izlazului
- Email:** aurelian@gmail.com
- Age:** 54
- Add** button

## 8. CONCLUZII

Am învățat să creez interfața mai sofisticată, să folosesc bazele de date și să organizez codul și clasele în mai multe pachete și să folosesc reflection.

## 9. BIBLIOGRAFIE

1. [https://gitlab.com/utcn\\_dsrl/pt-reflection-example/-/blob/master/src/main/java/bll/validators/Validator.java](https://gitlab.com/utcn_dsrl/pt-reflection-example/-/blob/master/src/main/java/bll/validators/Validator.java)
2. [https://www.w3schools.com/sql/sql\\_insert.asp](https://www.w3schools.com/sql/sql_insert.asp)