**UNIVERSITATEA TEHNICĂ “GH ASACHI” IAŞI**

**FACULTATEA AUTOMATICĂ ŞI CALCULATOARE**

**SPECIALIZAREA CALCULATOARE ŞI TEHNOLOGIA INFORMAŢIEI**

**DISCIPLINA BAZE DE DATE PROIECT**

Gestiunea unui magazin de componente de calculatoare

# Coordonator,

# Prof. Mironeanu Cătălin

# Student,

# Iurașcu Teodor Constantin

# Grupa 1310A

# Iași, 2022

**Titlul proiectului**

**Gestiunea unui magazin de componente de calculatoare**

Acest proiect oferă o modalitate de gestionare a stocului unui magazin de componente de calculatoare, alături de compatibilitatea acestor componente. Mai mult, se ofera și gestionarea clienților alături de comenzile lor, facturile atașate acestor comenzi, și adresele de livrare.

**Descrierea proiectului**

Stocul, compatibilitațile, și clienții, alături de comenzile lor, determină o necesitate a unei organizări bine pusă la punct pentru a evita potențiale pierderi de date. Pentru a asigura asta informațiile de care avem nevoie sunt:

- **Stocul:** totalitatea produselor disponibile, alături de categoriile din care fac parte.

- **Clientii:** datele asociate clientilor și comenzilor plasate de aceștia.

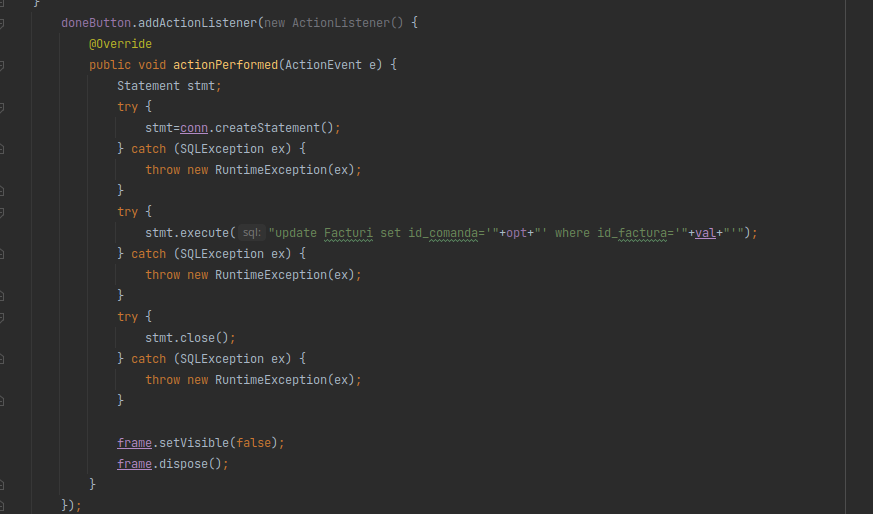
## Descrierea funcțională a aplicației

Principalele funcții ale aplicației sunt:

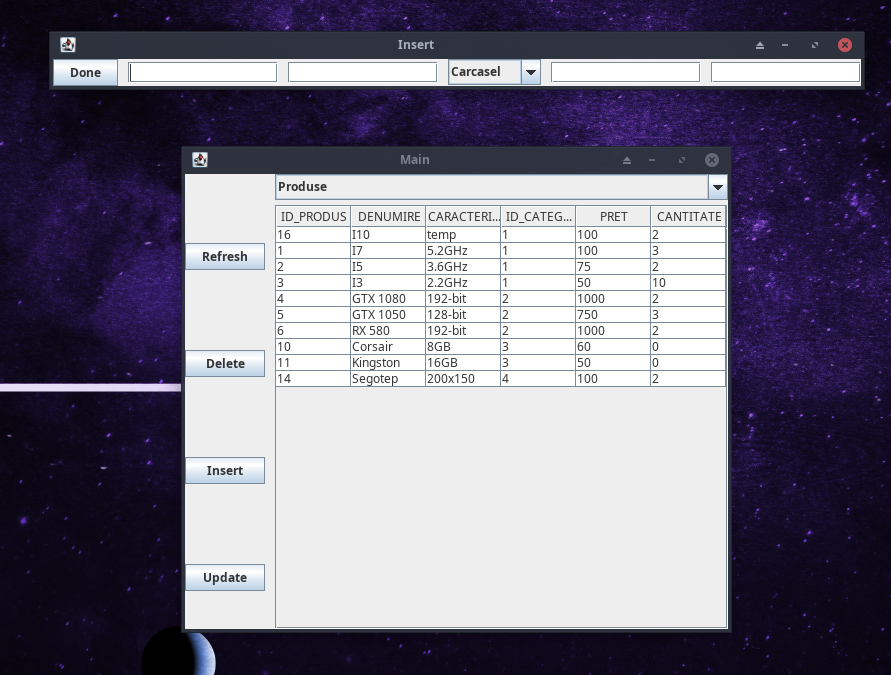
* evidența stocului
* evidența compatibilitaților
* evidența clienților
* evidența adreselor de livrare
* evidența comenzilor
* evidența facturilor

## Tehnologii folosite

Pe partea de backend s-a folosit Java alături de o bază de date Oracle.



Partea de frontend a fost dezvoltată prin Swing.



## Conectarea la baza de date

Conectarea la baza de date Oracle s-a efectuat prin intermediul driverului JDBC pentru Java 8.

## 

## Structura și relațiile dintre entități

Entitățile din această aplicație sunt:

* Categorii
* Produse
* Compatibilități
* Produse\_Pe\_Comandă
* Comenzi
* Clienți
* Detalii\_Livrare
* Facturi

În proiectarea acestei baze de date s-au identificat următoarele tipuri de relații:

1:1 (one-to-one), 1:n (one-to-many).

Între entitatea **Categorii** și entitatea **Produse** se stabilește o relație de 1:n. O categorie poate conține mai multe produse, dar un produs poate face parte doar dintr-o categorie.

Între entitatea **Produse** și entitatea **Compatibilități** se stabilesc doua relații de 1:n. Un produs poate face parte din mai multe compatibilități, iar o compatibilitate conține cate un set de două produse pentru a semnifica faptul că cele două sunt compatibile.

Între entitatea **Produse** și entitatea **Produse\_Pe\_Comandă** se stabileste o relație de 1:n. Un produs poate face parte din mai multe comenzi.

Între entitatea **Clienți** și entitatea **Comenzi** se stabilește o relatie de 1:n. Un client poate avea mai multe comenzi, dar o comandă aparține unui singur client.

Între entitatea **Comenzi** și entitatea **Facturi** se stabilește o relație de 1:n. O comanda poate fi împărțită in mai multe facturi, dar o factură poate face parte dintr-o singură comandă.

Între entitatea **Comenzi** și entitatea **Adrese** se stabilește o relație de 1:1. Fiecare comandă poate fi livrată la o singură adresă.

Între entitatea **Comenzi** și entitatea **Produse\_Pe\_Comanda** se stabilește o relație de 1:n. Fiecare comanda poate avea un singur total de produse, iar produsele pot aparține mai multor comenzi.

Între entitatea **Facturi** și entitatea **Produse\_Pe\_Comanda** se stabileste o relatie de 1:n. O factura apartie unui total de produse, iar produsele pot apartine mai multor facturi.

**Descrierea constrângerilor**

Constrângerea de tip **check** este prezentă în câmpul email al entității **Clienți** și în câmpurile pentru nume și denumiri ale entităților.

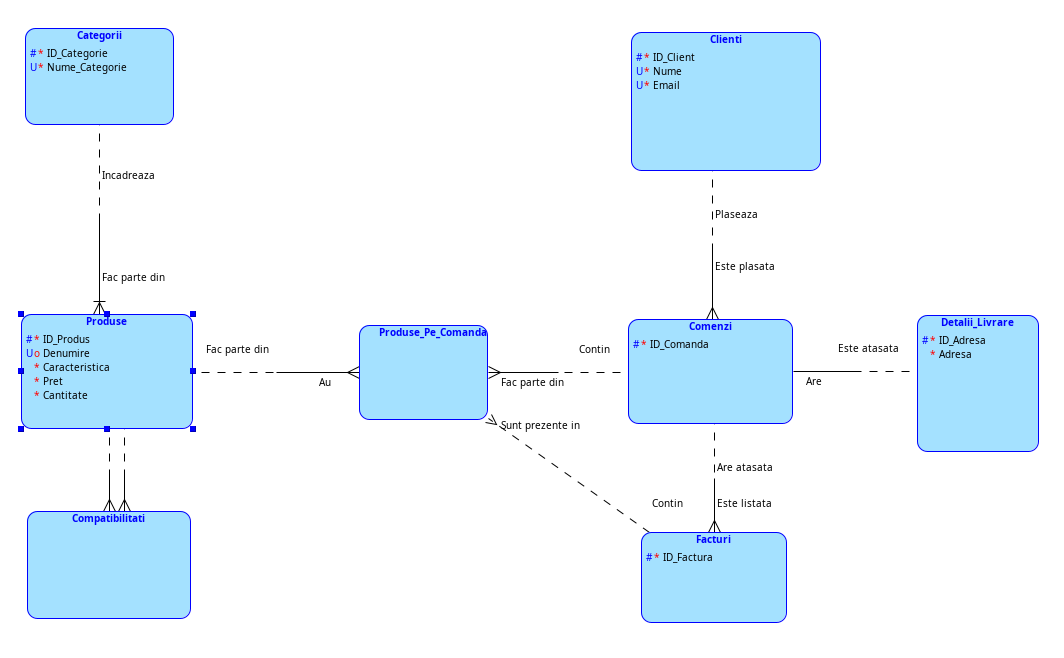
Constrângerile de tip **unique** sunt prezente în majoritatea entităților, în câmpurile pentru id-uri, denumiri, și email.

Constrângerile de tip **not null** sunt prezente în majoritatea tabelelor.

Primary key-urile sunt generate pe baza unui mecanism de autoincrement.

## Modelul logic

Pe baza entităților mai sus descrise, a rezultat următorul model logic:



O descriere amănunțită a atributelor fiecărei entități se va face la prezentarea *modelului relațional*, după ce vor fi fost convertite în coloane ale tabelelor.

**Aspecte legate de normalizare**

Baza de date a fost normalizată, deoarece îndeplinește următoarele condiții:

**A)** Tabelele respectă condițiile primei forme normale:

- un atribut conține valori atomice din domeniul său (și nu grupuri de astfel de valori)

- nu conține grupuri care se repetă

**B)** Tabelele respectă condițiile celei de-a doua forme normale:

- este în prima formă normală

- toate atributele non-cheie depind în totalitate de toate cheile candidat

**C)** Tabelele respectă a treia formă normală:

- este în a doua formă normală

- toate atributele non-cheie sunt direct (non-tranzitiv) dependente de toate cheile candidat.

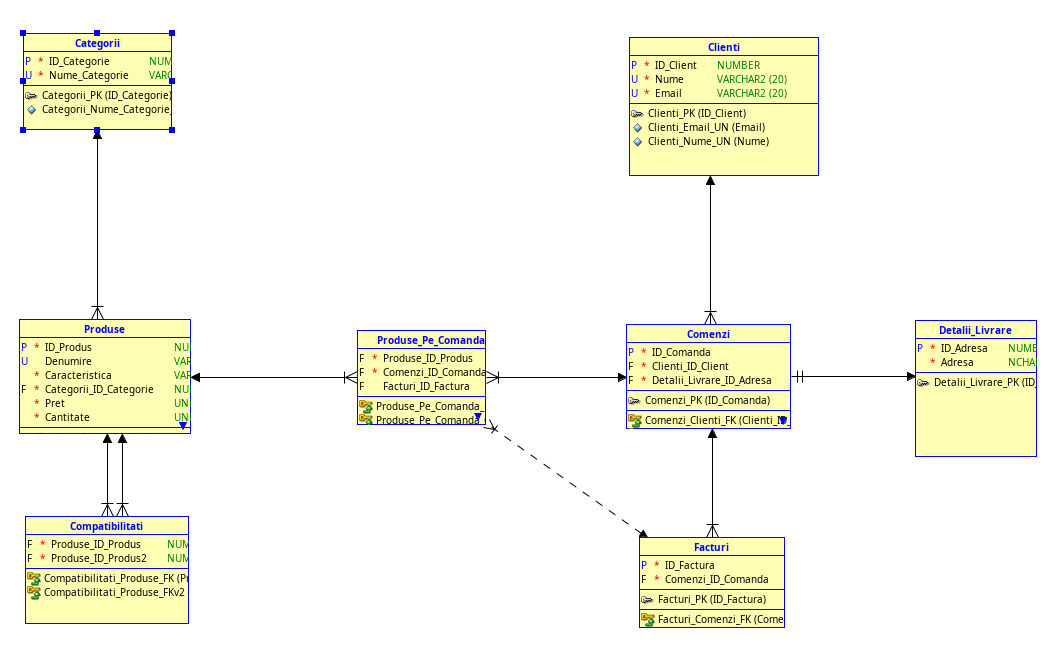
Prima formă normală este îndeplinită în cazul tuturor tabelelor; în fiecare tabelă, un câmp poate conține doar o singură valoare din domeniul acestuia. De exemplu, pentru tabela **Clienți** introducerea a doi clienți cu același email este imposibilă datorită constrangerii **unique** impuse pe acest câmp.

A doua formă normală este îndeplinită de toate tabelele; nicio cheie candidat nu este de tipul cheie multiplă, deci fiecare tabelă are doar o cheie primară.

A treia formă normală este îndeplinită în cazul relației dintre tabela **Produse** și **Categorii.** Atunci când specificăm o categorie a unui produs prin intermediul câmpului ID\_Categorie(FK), categoria acestui produs va fi preluată cu ajutorul cheii primare din tabela menționată.

**Modelul relațional**

Pe baza modelului logic, respectiv și a relațiilor dintre entitățile acestuia, dar și în urma procesului de normalizare, a rezultat următorul model relațional:

****

**Descriere a tabelelor finale și a coloanelor acestora**

Tabelele rezultate sunt:

* Categorii
* Produse
* Compatibilitati
* Produse\_Pe\_Comanda
* Clienti
* Comenzi
* Detalii\_Livrare
* Facturi

Tabela **Categorii** are următoarele câmpuri:

-ID\_Categorie: primary key

-Nume\_Categorie: unique, not null și check

Tabela **Produse** are următoarele câmpuri:

-ID\_Produs: primary key

-Denumire: unique,not null și check

-Caracteristica: not null

-ID\_Categorie: foreign key ce realizează relația 1:n intre categorii și produse

-Pret: not null

-Cantitate: not null

Tabela **Compatibilități** are următoarele câmpuri:

-ID\_Produs: foreign key ce realizează prima relație 1:n intre produse și compatibilitati

-ID\_Produs2: foreign key ce realizează a doua relație 1:n intre produse și compatibilitati

Tabela **Clienți** are următoarele câmpuri:

-ID\_Client: primary key

-Nume: unique, not null și check

-Email: unique,not null și prezintă o constrângere check pentru a verifica dacă adresa de email este validă

Tabela **Comenzi** are următoarele câmpuri:

-ID\_Comanda: primary key

-ID\_Client: foreign key ce realizează o relație 1:n cu entitatea clienți

-ID\_Adresa: foreign key ce realizează o relație 1:1 cu entitatea detalii\_livrare

Tabela **Detalii\_Livrare** conține următoarele câmpuri:

-ID\_Adresa: primary key

-Adresa: unique, not null și check

Tabela **Facturi** are următoarele câmpuri:

-ID\_Factura: primary key

-ID\_Comandă: foreign key ce realizează o relație 1:n cu entitatea comenzi

Tabela **Produse\_Pe\_Comanda** are următoarele câmpuri:

-ID\_Produs: foreign key ce realizează o relație 1:n cu entitatea produse

-ID\_Factura: foreign key ce realizează o relație 1:n cu entitatea facturi

-ID\_Comanda: foreign key ce realizeaza o relațtie 1:n cu entitatea comenzi