**Universitatea Tehnică a Moldovei**

**Departament Informatică și Ingineria Sistemelor**

**TEZĂ DE AN**

**Disciplina: Baze de date**

**Tema**: **Elaborarea Designului BD** pentru Proiectul "Monitorizarea Reducerilor în Comerț "

**utilizând** SGBD MySQL, instrumentul Open Server

**realizarea unei Aplicații Web** cu ajutorul Python, Flask, HTML, CSS, și JavaScript, pentru suportul utilizatorilor și comercianților în identificarea și analiza oportunităților de economisire.

**Domeniu:** Marketing și Comerț Electronic în cadrul unei Platforme Online

**Efectuat: Voicu Maxim, IA - 221**

**Verificat: Perebinos Mihail**

**Chişinău, 2023**

Cuprins

[1. Introducere în Baze de Date 4](#_Toc153846008)

[1.1. Conceptul Bazelor de Date. Prezentare Generală. 4](#_Toc153846009)

[1.2. Sisteme de Gestiune a Bazelor de Date. Prezentare generală. 5](#_Toc153846010)

[2. Design-ul BD. Modelul Conceptual. Modelul Logic. Schema ER 7](#_Toc153846011)

[2.1. Descrierea Domeniului de Studiu 7](#_Toc153846012)

[2.2. Formularea problemei 8](#_Toc153846013)

[2.3. Scopul și obiectivele proiectului 8](#_Toc153846014)

[2.4. Prezentarea Conținutului/Contentului informațional (Output-ul) 9](#_Toc153846015)

[2.5. Modelul conceptual 12](#_Toc153846016)

[2.6. Schema legăturilor între subdomenii 12](#_Toc153846017)

[2.7. Tabel de acoperire 12](#_Toc153846018)

[2.8. Prezentarea și descrierea legăturilor între obiectele domeniului de studiu 14](#_Toc153846019)

[2.9. Machetul bazei de date 15](#_Toc153846020)

[2.10. Modelul logic al BD. Schema ER 17](#_Toc153846021)

[3. Introducere în Programare WEB 18](#_Toc153846022)

[3.1. HTML Prezentare generală 18](#_Toc153846023)

[2. CSS. Prezentare generală 20](#_Toc153846024)

[3.3. Limbaj de scriptare. Python și framework-ul FLASK 21](#_Toc153846025)

[3.4. Administrarea bazelor de date folosind MySQL 22](#_Toc153846026)

[3.5. Severul OpenServer 23](#_Toc153846027)

[4. Structura Bazei de Date 25](#_Toc153846028)

[4.1. Tabelul discount\_type 25](#_Toc153846029)

[4.2. Tabelul city 26](#_Toc153846030)

[4.3. Tabelul contact\_info 26](#_Toc153846031)

[4.4. Tabelul store\_network 26](#_Toc153846032)

[4.5. Tabelul store 27](#_Toc153846033)

[4.6. Tabelul manufacturer 27](#_Toc153846034)

[4.7. Tabelul product\_category 28](#_Toc153846035)

[4.8. Tabelul product 28](#_Toc153846036)

[4.9. Tabelul discount 29](#_Toc153846037)

[4.10. Tabelul discount\_instance 30](#_Toc153846038)

[4.11. Tabelul percentage\_discount 31](#_Toc153846039)

[4.12. Tabelul fixed\_value\_discount 31](#_Toc153846040)

[4.13. Tabelul quantity\_discount 31](#_Toc153846041)

[4.14. Tabelul complementary\_discount 32](#_Toc153846042)

[4.15. Tabelul complementary\_discount\_product 32](#_Toc153846043)

[5. Structura Aplicației WEB 33](#_Toc153846044)

[5.1. Scopul aplicației WEB 33](#_Toc153846045)

[5.2. Descrierea aplicației WEB 33](#_Toc153846046)

[5.3. Descrierea operațiilor CRUD cu tabelele Bazei de Date 39](#_Toc153846047)

[5.3.1. Operația READ 39](#_Toc153846048)

[5.3.2. Operația UPDATE 40](#_Toc153846049)

[5.3.3. Operația DELETE 41](#_Toc153846050)

[5.3.4. Operația CREATE 41](#_Toc153846051)

[6. Listing-ul Aplicației WEB 43](#_Toc153846052)

[6.1. Exemplu de model definit pentru tabelul product 43](#_Toc153846053)

[6.2. Exemplu de funcție pentru formatare a datelor în JSON 43](#_Toc153846054)

[6.3. Definirea rutelor 43](#_Toc153846055)

[6.4. Exemplu de interogare a bazei de date 44](#_Toc153846056)

[6.5. Exemplu de controller pentru obținerea reducerilor 44](#_Toc153846057)

[6.6. Exemplu de fetching al datelor pe partea de frontend 45](#_Toc153846058)

[Concluzii 46](#_Toc153846059)

[Bibliografie 47](#_Toc153846060)

# 1. Introducere în Baze de Date

## 1.1. Conceptul Bazelor de Date. Prezentare Generală.

Bazele de date reprezintă un pilon central în domeniul tehnologiei informației, oferind un cadru organizat pentru stocarea, prelucrarea și regăsirea eficientă a datelor. Aceste sisteme structurate de informații se caracterizează printr-o organizare riguroasă, concepute pentru a facilita accesul și manipularea rapidă a datelor prin intermediul dispozitivelor informatice. În esență, o bază de date este alcătuită dintr-o serie de înregistrări, fiecare dintre acestea reprezentând o entitate distinctă, cu atribute și caracteristici comune, structurate într-un mod uniform și interconectate.

Aceste înregistrări, sau entități, sunt blocurile de construcție ale oricărei baze de date și pot varia de la informații pur numerice la texte sau reprezentări grafice. Fiecare înregistrare conține câmpuri de date, care descriu atributele specifice ale entității. Există mai multe tipuri de baze de date, fiecare cu particularitățile și utilizările sale:

1. *Baze de date rețea (distribuite)*: Acestea permit stocarea datelor pe mai multe locații fizice, fiind interconectate prin rețele de calculatoare. Sunt folosite în scenarii unde datele sunt distribuite geografic sau unde este necesară o accesibilitate sporită.
2. *Baze de date relaționale*: Se bazează pe modelul relațional, unde datele sunt organizate în tabele. Fiecare tabel reprezintă o relație și conține rânduri și coloane. Limbajul SQL este adesea utilizat pentru manipularea datelor în aceste baze.
3. *Baze de date orientate pe obiecte*: Aceste baze de date stochează informațiile sub forma obiectelor, așa cum sunt ele definite în limbajele de programare orientate pe obiect. Acest tip de bază de date este util în aplicații care necesită modelarea complexă a datelor, precum sistemele CAD sau sistemele de management al documentelor.

O bază de date eficientă trebuie să îndeplinească mai multe condiții esențiale, cum ar fi asigurarea unei interdependențe sporite între date și programe, minimizarea redundanței și permiterea accesului rapid la informații. Arhitectura unei baze de date include, de obicei, trei componente principale: baza de date propriu-zisă, sistemul de gestiune a bazei de date și un dicționar al bazei de date (metabaza de date), care oferă informații esențiale despre structura și semantica datelor.

În contextul dezvoltării continue a tehnologiei, bazele de date devin din ce în ce mai sofisticate și variate, reflectând nevoia crescândă de gestionare eficientă a unor volume mari de date în diverse domenii. Prezenta lucrare își propune să exploreze conceptele fundamentale ale bazelor de date, evidențiind importanța lor în cadrul sistemelor informatice moderne și contribuția lor la evoluția tehnologică.

## 1.2. Sisteme de Gestiune a Bazelor de Date. Prezentare generală.

Sistemele de Gestiune a Bazelor de Date (SGBD) reprezintă o componentă esențială în arhitectura modernă a sistemelor informatice, specializate în stocarea și prelucrarea eficientă a unui volum mare de date. Aceste sisteme joacă un rol vital în organizarea datelor pe suport fizic de memorare și gestionează diverse operații aplicate asupra acestor date.

SGBD-urile sunt configurate ca intermediari între utilizatori și bazele de date, oferind un cadru prin care utilizatorii pot defini, construi, organiza și manipula date pentru diverse aplicații. Ele facilitează interacțiunea cu baza de date prin intermediul unor interfețe și limbaje de programare specifice, permițând utilizatorilor să execute interogări complexe și să actualizeze datele eficient.

Funcționalitățile esențiale ale unui SGBD includ:

1. *Definirea și Descrierea Structurii Bazei de Date*: Se realizează prin intermediul unui limbaj de definire a datelor (LDD), care permite modelarea și structurarea datelor conform unui anumit model de date.
2. *Încărcarea și Manipularea Datelor*: Încărcarea datelor în baza de date și manipularea lor se efectuează folosind limbajul de manipulare a datelor (LMD), care include operații de inserare, ștergere și actualizare a datelor.
3. *Interogarea și Actualizarea Datelor*: SGBD-urile permit interogarea datelor pentru vizualizarea și consultarea informațiilor și actualizarea acestora, respectând restricțiile de integritate ale bazei de date.
4. *Întreținerea și Reorganizarea Bazei de Date*: Aceste operații sunt esențiale pentru menținerea eficienței și actualitatea bazei de date, fiind executate prin utilitare proprii ale SGBD sau de către administratorul bazei de date.
5. *Securitatea Datelor*: Asigurarea confidențialității și securității datelor este o componentă critică, implicând autorizarea și controlul accesului la date, precum și criptarea acestora.

Funcțiile distincte ale unui SGBD sunt:

1. *Funcția de Descriere a Datelor*: Realizată cu ajutorul LDD, aceasta implică descrierea atributelor, legăturilor dintre entități, criteriilor de validare a datelor și metodelor de acces.
2. *Funcția de Manipulare*: Este responsabilă pentru actualizarea și regăsirea datelor, fiind considerată cea mai complexă funcție a unui SGBD.
3. *Funcția de Utilizare*: Asigură interfețele necesare pentru comunicarea tuturor utilizatorilor cu baza de date, facilitând interacțiunea eficientă și accesibilă.

Independența datelor în cadrul SGBD-urilor este un concept cheie, permitând modificarea structurii datelor la un anumit nivel fără a afecta celelalte niveluri. Această independență se manifestă atât la nivel fizic, cât și conceptual, contribuind la flexibilitatea și adaptabilitatea sistemului în fața schimbărilor.

În concluzie, SGBD-urile sunt instrumente indispensabile în gestionarea bazelor de date, oferind o gamă largă de funcționalități care permit manipularea eficientă și sigură a datelor, esențiale în diverse domenii ale tehnologiei informației.

# 2. Design-ul BD. Modelul Conceptual. Modelul Logic. Schema ER

## 2.1. Descrierea Domeniului de Studiu

În domeniul economic, fenomenul reducerilor de preț constituie o tematică de studiu vastă, implicând o serie de aspecte psihologice, comportamentale și strategice. Reducerile de preț, ca instrument de marketing, sunt utilizate pentru a stimula cererea, a gestiona stocurile și a influența percepția consumatorilor asupra valorii produselor. Analiza impactului reducerilor asupra deciziilor de cumpărare și a percepției brandului devine astfel esențială în evaluarea eficienței acestor strategii comerciale. O platformă web pentru monitorizarea reducerilor ar putea oferi beneficii semnificative cumpărătorilor, producătorilor și administratorilor de magazine. Pentru cumpărători, accesul la informații actualizate și personalizate privind reducerile ar putea facilita luarea deciziilor de cumpărare, sporind eficiența și satisfacția acestora. Din perspectiva producătorilor și administratorilor de magazine, utilizarea unei baze de date robuste ar permite o analiză mai precisă a tendințelor de piață și a comportamentului consumatorilor. Astfel, se pot dezvolta strategii de preț și promoții mai eficiente, bazate pe date concrete. De exemplu, analiza datelor de vânzare în timp real poate ajuta la identificarea produselor pentru care reducerile stimulează cel mai mult vânzările sau la ajustarea rapidă a strategiilor de preț în funcție de schimbările de pe piață. O astfel de platformă ar putea integra algoritmi de analiză predictivă, care să anticipeze tendințele viitoare de consum, optimizând astfel ofertele de reduceri. Aceasta ar putea include funcționalități de inteligență artificială pentru a personaliza recomandările de reduceri pentru utilizatori, crescând gradul de relevanță și eficiență al acestora.

În contextul actual, se observă o prezență constantă a publicității stradale și a reclamelor online, care promovează reducerile și ofertele magazinelor. Pe străzi, distribuirea revistelor cu listele de reduceri ale magazinelor reprezintă o practică răspândită, în timp ce, în spațiul virtual, reclamele și promoțiile sunt omniprezente. Acest fenomen reflectă eforturile intense ale comercianților de a atrage atenția potențialilor cumpărători. Cu toate acestea, un segment considerabil de consumatori manifestă o preferință pentru eliminarea acestui tip de publicitate, percepută adesea ca fiind invazivă și obositoare.

Pe de altă parte, realitatea pieței evidențiază o provocare majoră: dificultatea de a monitoriza eficient reducerile oferite de diverse magazine pentru o gamă extinsă de produse. Magazinele se confruntă cu riscul ca ofertele lor atractive să rămână neobservate de către cumpărători, din cauza strategiilor de marketing ineficiente sau a suprasaturării de informații. În acest context, există o cerere crescută pentru o soluție care să permită centralizarea și accesul facil la informații despre reduceri.

## 2.2. Formularea problemei

Problema centrală cu care se confruntă piața reducerilor constă în dificultatea de a echilibra eficacitatea campaniilor de marketing cu percepția și receptivitatea consumatorilor. Reducerile, deși potențial avantajoase atât pentru cumpărători, cât și pentru producători și administratori de magazine, se confruntă adesea cu riscul de a fi ignorate sau considerate superficial din cauza reclamelor invazive și obositoare. În plus, campaniile de promovare nu reușesc întotdeauna să atingă grupul țintă dorit, ceea ce reduce eficiența eforturilor de marketing.

Un alt aspect semnificativ al problemei este necesitatea colectării și analizei individuale a datelor privind reducerile. În prezent, acest proces este fragmentat și consumator de timp, atât pentru comercianți, cât și pentru cumpărători. O platformă centralizată ar putea îmbunătăți considerabil această situație, oferind un punct unic de acces la informații despre reduceri și promoții.

Această platformă ar putea funcționa ca un instrument de feedback valoros pentru producători și administratori de magazine, oferindu-le date analitice despre eficacitatea reducerilor lor. De asemenea, ar putea anunța cumpărătorii atunci când un produs dorit este disponibil la reducere, personalizând experiența de cumpărături și crescând gradul de satisfacție al clientului.

Implementarea unei astfel de platforme presupune utilizarea tehnologiilor avansate de colectare și analiză a datelor, precum și a algoritmilor de inteligență artificială pentru a anticipa și răspunde nevoilor consumatorilor. Acest lucru ar putea duce la o aliniere mai bună între oferta de reduceri și cerințele pieței, optimizând astfel atât experiența de cumpărături a consumatorilor, cât și strategiile de marketing ale comercianților.

## 2.3. Scopul și obiectivele proiectului

Scopul principal al acestei lucrări este dezvoltarea unei platforme web inovative, destinată monitorizării și gestionării eficiente a reducerilor. Această platformă va oferi funcționalități complexe, atât pentru cumpărători, în vederea urmăririi reducerilor disponibile, cât și pentru producători sau entități de vânzare, permițându-le să colecteze date statistice, să analizeze tendințele pieței și să gestioneze reducerile. Utilizatorii vor beneficia de informații detaliate privind durata fiecărei reduceri, avantajele ofertelor, locațiile unde aceste reduceri sunt valabile și condițiile necesare pentru a beneficia de acestea.

Obiective:  
1. Oferirea unei soluții digitale pentru producători și distribuitori pentru a plasa și gestiona reduceri în mediul online, optimizând astfel vizibilitatea și eficiența acestora.

2. Crearea unei experiențe personalizate pentru cumpărători, oferindu-le posibilitatea de a monitoriza și de a accesa reducerile relevante pentru nevoile și preferințele lor.

3. Implementarea unui sistem robust de analiză a datelor, care să permită o înțelegere aprofundată a impactului fiecărei reduceri și a comportamentului consumatorilor.

4. Asigurarea unei interfețe intuitive și ușor de utilizat, care să faciliteze accesul rapid și eficient la informații, indiferent de nivelul de experiență tehnologică al utilizatorilor.

## 2.4. Prezentarea Conținutului/Contentului informațional (Output-ul)

1. Ieșiri informaționale

1. Lista reducerilor (”Subdomeniul Reduceri”)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| discount.id | discount.description | discount.start\_date | discount.end\_date | discount\_type.id | product.id | store.id |

2. Lista tipurilor de reducere (”Subdomeniul Reduceri”)

|  |  |  |  |
| --- | --- | --- | --- |
| discount\_type.id | discount\_type.name | discount\_type.name | discount\_type.table\_name |

3. Lista tuturor reducerilor procentuale (”Subdomeniul Reduceri”)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| discount.id | discount.description | | discount.start\_date | | discount.end\_date | | percentage\_discount.id |
| percentage\_discount.percentage | | percentage\_discount.final\_price | | product.id | |

4. Lista reducerilor fixate (”Subdomeniul Reduceri”)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| discount.id | discount.description | | discount.start\_date | | discount.end\_date | | fixed\_value\_discount.id |
| fixed\_value\_discount.amount | | fixed\_value\_discount.final\_price | | product.id | |

5. Lista reducerilor complementare (”Subdomeniul Reduceri”)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| discount.id | discount.description | | discount.start\_date | | discount.end\_date | | comp\_discount.id | |
| comp\_disc.offer\_product\_count | | comp\_disc.initial\_price | | comp\_disc.final\_price | | comp\_disc\_product.id | | product.id |

6. Reducere cantitativă (”Subdomeniul Reduceri”)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| discount.id | discount.description | | discount.start\_date | | discount.end\_date | | quantity\_discount.id |
| quantity\_discount.required\_quantity | | quantity\_discount.free\_quantity | | quantity\_discount.final\_price | | product.id | |

7. Lista magazine (”Subdomeniul Magazine”)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| store.id | store.name | store.address | store.opening\_hour | store.closing\_hour | store.is\_open\_24\_7 | city.id | store\_network.id |

8. Lista produse reducere (”Subdomeniul Produse”)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| discount.id | percentage\_disc.final\_price | | fixed\_value\_disc.final\_price | | quantity\_disc.final\_price | | | comp\_disc.final\_price |
| product.id | product.name | product.volume | | product.unique\_code | | product.price | manufacturer.id | |

2. Interogări

1. este dată reducerea XXX, să se afișeze lista magazinelor unde e valabilă reducerea

|  |  |  |
| --- | --- | --- |
| store.id | store.name | store.address |

2. este dat prețul XXX, să se afișeze lista reducerilor

|  |  |  |
| --- | --- | --- |
| discount.id | discount.description | discount\_type.id |

3. este dată reducerea XXX, să se afișeze când reducerea se finalizează

|  |  |
| --- | --- |
| discount.description | discount.end\_date |

4. este dată categoria XXX, să se afișeze reducerile disponibile

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| discount.description | | discount.start\_date | | discount.end\_date | discount\_type.id |
| product.name | discount\_instance.id | |

5. este dată rețeaua XXX, să se afișeze lista tuturor magazinelor

|  |  |  |  |
| --- | --- | --- | --- |
| store.id | store.name | store.address | city.name |

6. este dat magazinul XXX, să se afișeze informația disponibilă despre el

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| store.name | store.address | | city.name | | city.region | city.postal\_code |
| contact\_info.phone\_number | | contact\_info.email | |

7. este dat produsul XXX, să se afișeze producătorul și informația de contact

|  |  |  |  |
| --- | --- | --- | --- |
| manufacturer.id | manufacturer.name | contact\_info.phone\_number | contact\_info.email |

8. este dat magazinul XXX, să se afișeze informația despre rețeaua de magazine din care face parte

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| store\_network.id | store\_network.name | store\_network.store\_count | | store\_network.headquarters\_address |
| contact\_info.phone\_number | contact\_info.email | |

3. Servicii

1. să se determine reducerea cu cea mai mare valoare procentuală

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| discount.id | discount.description | discount.start\_date | discount.end\_date | percentage\_discount.percentage |

2. este dat prețul XXX și categoria YYY, să se determine reducerile disponibile

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| discount.id | discount.description | discount.start\_date | discount.end\_date | discount\_type.id |

3. să se determine reducerile ce au prețul final în intervalul Y -> X

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| discount.id | discount.description | discount.start\_date | discount.end\_date | discount\_type.id | product\_id |

4. este dată rețeaua Y, să se determine magazinele disponibile

|  |  |  |
| --- | --- | --- |
| store.name | store.address | contact\_info.id |

5.este dată produsul XXX, să se determine din ce categorie face parte și ce reduceri sunt valabile

|  |  |  |  |
| --- | --- | --- | --- |
| discount.id | discount.description | product\_category.id | product\_category.name |

4. Restricții

- Dacă data de finalizare a reducerii a trecut sau nu mai sunt magazine care oferă această reducere, reducerea și toată informația legată de ea e eliminată din tabelul discount

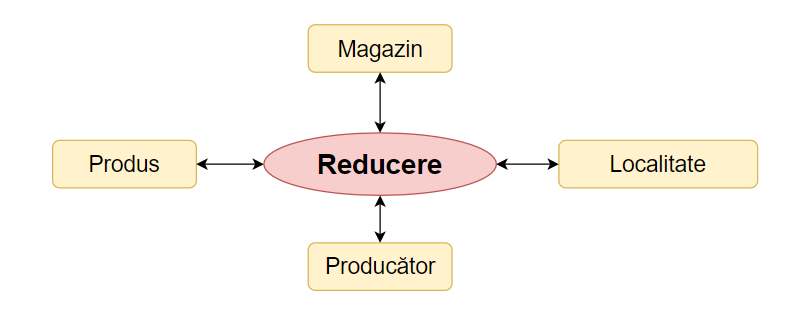
- Valoarea procentuală a reducerii trebuie să fie între 0 și 100

- Fiecare reducere trebuie să aibă minim o instantă valabilă

- Fiecare reducere trebuie să iși regăsească continuarea în un singur tabel ce descrie tipurile de reduceri

## 2.5. Modelul conceptual

## 2.6. Schema legăturilor între subdomenii



## 2.7. Tabel de acoperire

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Ieșiri informaționale | | | | | | | | Interogări | | | | | | | | Servicii | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 1 | 2 | 3 | 4 | 5 |
| discount.id | + |  | + | + | + | + |  | + |  | + |  |  |  |  |  |  | + | + | + |  | + |
| discount.description | + |  | + | + | + | + |  |  |  | + | + | + |  |  |  |  | + | + | + |  | + |
| discount.start\_date | + |  | + | + | + | + |  |  |  |  |  | + |  |  |  |  | + | + | + |  |  |
| discount.end\_date | + |  | + | + | + | + |  |  |  |  | + | + |  |  |  |  | + | + | + |  |  |
| discount\_type.id | + | + |  |  |  |  |  |  |  | + |  | + |  |  |  |  |  | + | + |  |  |
| discount\_type.name |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| discount\_type.description |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| discount\_type.table\_name |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| percentage\_discount.id |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| percentage\_discount.percentage |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  | + |  |  |  |  |
| percentage\_discount.final\_price |  |  | + |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |
| fixed\_value\_discount.id |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| fixed\_value\_discount.amount |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| fixed\_value\_discount.final\_price |  |  |  | + |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |
| comp\_discount.id |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| compl\_discount.offer\_product\_count |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| compl\_discount.initial\_price |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| compl\_discount.final\_price |  |  |  |  | + |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |
| quantity\_discount.id |  |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| quantity\_discount.required\_quantity |  |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| quantity\_discount.free\_quantity |  |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| quantity\_discount.final\_price |  |  |  |  |  | + |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |
| complementary\_discount\_product.id |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| product.id | + |  | + | + | + | + |  | + |  |  |  |  |  |  |  |  |  |  | + |  |  |
| product.name |  |  |  |  |  |  |  | + |  |  |  | + |  |  |  |  |  |  |  |  |  |
| product.volume |  |  |  |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |
| product.unique\_code |  |  |  |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |
| product.price |  |  |  |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |
| product\_category.id |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | + |
| product\_category.name |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | + |
| manufacturer.id |  |  |  |  |  |  |  | + |  |  |  |  |  |  | + |  |  |  |  |  |  |
| manufacturer.name |  |  |  |  |  |  |  |  |  |  |  |  |  |  | + |  |  |  |  |  |  |
| discount\_instance.id |  |  |  |  |  |  |  |  |  |  |  | + |  |  |  |  |  |  |  |  |  |
| store.id | + |  |  |  |  |  | + |  | + |  |  |  | + |  |  |  |  |  |  |  |  |
| store.name |  |  |  |  |  |  | + |  | + |  |  |  | + | + |  |  |  |  |  | + |  |
| store.address |  |  |  |  |  |  | + |  | + |  |  |  | + | + |  |  |  |  |  | + |  |
| store.opening\_hour |  |  |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| store.closing\_hour |  |  |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| store.is\_open\_24\_7 |  |  |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| city.id |  |  |  |  |  |  | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| city.name |  |  |  |  |  |  |  |  |  |  |  |  | + | + |  |  |  |  |  |  |  |
| city.region |  |  |  |  |  |  |  |  |  |  |  |  |  | + |  |  |  |  |  |  |  |
| city.postal\_code |  |  |  |  |  |  |  |  |  |  |  |  |  | + |  |  |  |  |  |  |  |
| store\_network.id |  |  |  |  |  |  | + |  |  |  |  |  |  |  |  | + |  |  |  |  |  |
| store\_network.name |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | + |  |  |  |  |  |
| store\_network.store\_count |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | + |  |  |  |  |  |
| store\_network.headquarters\_address |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | + |  |  |  |  |  |
| contact\_info.id |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | + |  |
| contact\_info.phone\_number |  |  |  |  |  |  |  |  |  |  |  |  |  | + | + | + |  |  |  |  |  |
| contact\_info.email |  |  |  |  |  |  |  |  |  |  |  |  |  | + | + | + |  |  |  |  |  |

## 2.8. Prezentarea și descrierea legăturilor între obiectele domeniului de studiu

## 2.9. Machetul bazei de date

|  |  |  |  |
| --- | --- | --- | --- |
| **Tabel** | **Câmp** | **Tip** | **Comentarii** |
| city | id | INT | PK, A\_I |
| name | VARCHAR(20) | NOT NULL |
| region | VARCHAR(20) | NOT NULL |
| postal\_code | VARCHAR(10) |  |
| complementary\_discount | id | INT | PK, A\_I |
| discount\_id | INT | FK, NOT NULL |
| offer\_product | INT | NOT NULL |
| initial\_price | DECIMAL(10, 2) | NOT NULL |
| final\_price | DECIMAL(10, 2) | NOT NULL |
| complementary\_discount\_product | id | INT | PK, A\_I |
| complementary\_discount\_id | INT | FK, NOT NULL |
| product\_id | INT | FK, NOT NULL |
| contact\_info | id | INT | PK, A\_I |
| phone\_number | VARCHAR(15) |  |
| email | VARCHAR(50) |  |
| discount\_type | id | INT | PK, A\_I |
| name | VARCHAR(30) | NOT NULL |
| description | VARCHAR(255) |  |
| table\_name | VARCHAR(30) | NOT NULL |
| discount | id | INT | PK, A\_I |
| description | VARCHAR(255) | NOT NULL |
| discount\_type\_id | INT | FK, NOT NULL |
| start\_date | DATE | NOT NULL |
| end\_date | DATE | NOT NULL |
| discount\_instance | id | INT | PK, A\_I |
| discount\_id | INT | FK, NOT NULL |
| store\_id | INT | FK, NOT NULL |
| fixed\_value\_discount | id | INT | PK, A\_I |
| discount\_id | INT | FK, NOT NULL |
| product\_id | INT | FK, NOT NULL |
| discount\_amount | DECIMAL(10, 2) | NOT NULL |
| final\_price | DECIMAL(10, 2) | NOT NULL |
| manufacturer | id | INT | PK, A\_I |
| name | VARCHAR(40) | NOT NULL |
| contact\_id | INT | FK, NOT NULL |
| percentage\_discount | id | INT | PK, A\_I |
| discount\_id | INT | FK, NOT NULL |
| product\_id | INT | FK, NOT NULL |
| discount\_percentage | TINYINT UNSIGNED | NOT NULL, BETWEEN 1 AND 99 |
| product | id | INT | PK, A\_I |
| name | VARCHAR(50) | NOT NULL |
| volume | DECIMAL(10,2) |  |
| manufacturer\_id | INT | FK, NOT NULL |
| unique\_code | VARCHAR(255) | NOT NULL |
| price | DECIMAL(10,2) | NOT NULL |
| category\_id | INT | FK, NOT NULL |
| product\_category | id | INT | PK, A\_I |
| name | VARCHAR(100) | NOT NULL |
| quantity\_discount | id | INT | PK, A\_I |
| discount\_id | INT | FK, NOT NULL |
| product\_id | INT | FK, NOT NULL |
| required\_quantity | INT | NOT NULL |
| final\_price | INT | NOT NULL |
| free\_quantity | INT | NOT NULL |
| store | id | INT | PK, A\_I |
| name | VARCHAR(100) | NOT NULL |
| network\_id | INT | FK, NOT NULL |
| address | VARCHAR(100) |  |
| contact\_id | INT | FK, NOT NULL |
| opening\_hour | TIME |  |
| closing\_hour | TIME |  |
| is\_open\_24\_7 | BOOLEAN | NOT NULL |
| city\_id | INT | FK, NOT NULL |
| store\_network | id | INT | PK, A\_I |
| name | VARCHAR(20) | NOT NULL |
| store\_count | INT |  |
| contact\_id | INT | FK, NOT NULL |
| headquarters\_address | VARCHAR(100) |  |

## 2.10. Modelul logic al BD. Schema ER

# 3. Introducere în Programare WEB

Programarea web reprezintă o disciplină esențială în domeniul informaticii, care se ocupă de dezvoltarea și implementarea aplicațiilor și site-urilor web. Aceasta implică utilizarea unui set diversificat de tehnologii și limbaje de programare pentru a crea interfețe utilizator prietenoase și funcționale pe internet. Una dintre cele mai fundamentale componente ale programării web o reprezintă limbajele de marcare, precum HTML (HyperText Markup Language) și CSS (Cascading Style Sheets). HTML este folosit pentru structurarea conținutului paginilor web, în timp ce CSS este responsabil pentru stilizarea acestora. Aceste două limbaje funcționează împreună pentru a da formă și aspect paginilor web. În dezvoltarea web, limbajele de programare, cum ar fi JavaScript, PHP și Python, sunt folosite pentru a crea funcționalități interactive și pentru a gestiona datele pe partea clientului și pe cea serverului. De exemplu, JavaScript poate fi folosit pentru validarea formularilor, manipularea elementelor din pagină și efectuarea de cereri către server prin intermediul API-urilor (Interfețe de Programare a Aplicațiilor).

Pentru dezvoltarea aplicațiilor web mai complexe, framework-urile joacă un rol crucial. Framework-urile web, cum ar fi Angular, React și Vue.js, oferă structuri și șabloane predefinite care facilitează dezvoltarea rapidă și organizată a aplicațiilor. Ele furnizează un cadru de lucru și un set de reguli pentru dezvoltatori, similar modului în care o casă este construită pe baza unui plan arhitectural. În ceea ce privește partea de backend a dezvoltării web, aceasta se referă la partea invizibilă a unei aplicații, care gestionează baza de date, autentificarea utilizatorilor și logica de afaceri. Limbaje de programare precum Node.js, Ruby, și Java sunt utilizate pentru a construi această parte a aplicațiilor web. În concluzie, programarea web implică utilizarea unui set variat de tehnologii și limbaje pentru a crea site-uri web și aplicații web interactive și funcționale. De la HTML și CSS pentru structura și aspectul paginilor web, la JavaScript și framework-uri pentru funcționalități complexe, dezvoltarea web este o disciplină fascinantă și în continuă evoluție, cu un impact semnificativ asupra mediului online.

## 3.1. HTML Prezentare generală

HTML, sau Limbajul de Marcare pentru Hipertext (Hypertext Markup Language), este un pilon fundamental în lumea programării web. Dezvoltat inițial pentru a facilita partajarea documentelor în comunitatea științifică internațională, HTML a devenit limbajul universal de scriere pentru publicarea informațiilor pe Internet. Caracteristicile sale esențiale includ capacitatea de a adăuga textului informații de formatare, precum și de a îngloba diverse tipuri de conținut, cum ar fi imagini și sunete. Spre deosebire de procesoarele de texte convenționale, care folosesc formate specifice, documentele HTML sunt scrise în text pur (ASCII), permițându-le să fie afișate de un număr mare de browsere web pe diverse platforme.

Documentele HTML pot fi create fie cu un editor de texte simplu, fie cu editoare HTML speciale care oferă o experiență vizuală (WYSIWYG - What You See Is What You Get), dar rezultatul final rămâne un document în format text pur. Autorii folosesc HTML pentru a publica documente cu diverse elemente, cum ar fi headere, texte, tabele, liste, fotografii, precum și pentru a crea legături hypertext accesibile printr-un simplu click de mouse. De asemenea, HTML permite proiectarea formularelor pentru tranzacții cu servere la distanță și includerea altor aplicații direct în documente.

HTML se bazează pe patru caracteristici principale:

1. Utilizează un marcaj descriptiv pentru a indica diverse acțiuni, cu etichete descriptive precum <CHAPTER> sau <TITLE>.
2. Definește structuri de documente ierarhice și legături intra- și inter-documente.
3. Este guvernat de o descriere formală printr-un Document Type Definition (DTD) care stabilește specificațiile limbajului.
4. Este accesibil atât pentru oameni, cât și pentru computere, datorită separării clare dintre text și marcaje.

HTML este bazat pe SGML (Standard Generalized Markup Language), un standard internațional care oferă o metodă de reprezentare a structurii documentelor și hyper-documentelor, concentrându-se pe conținutul documentului mai degrabă decât pe aspectul său. Această abordare permite HTML să descrie documente structurate, având la bază ideea că majoritatea documentelor conțin elemente comune ce pot fi marcate cu etichete corespunzătoare.

Documentele HTML se diferențiază de procesoarele de texte prin faptul că nu se concentrează pe modul în care elementele vor fi afișate pe ecran. Acest lucru oferă mai multă flexibilitate programelor care interpretează HTML și afișează documentul, permițând o adaptare la diverse platforme și browsere. Deși posibilitățile de formatare ale HTML sunt limitate, separarea structurii documentului de modul său de afișare asigură transferabilitatea și vizualizarea documentelor pe orice platformă și browser.

Versiunile mai recente ale HTML includ elemente precum text centrat sau aliniat la dreapta, tabele, formule matematice și alinierea imaginilor cu textul. Extensiile, cum ar fi cele introduse de Netscape și Internet Explorer, au permis autorilor să depășească anumite limitări ale standardului. HTML a fost preferat pentru publicații pe Web datorită simplității sale, a capacității de a formata text ASCII cu etichete ASCII, a compresiei eficiente, suportului pentru legături hypertext și ușurinței în scrierea programelor de navigare pentru vizualizarea documentelor.

HTML, CSS și JavaScript formează un triunghi fundamental în dezvoltarea web, fiecare având roluri distincte dar profund interconectate. HTML este structura de bază a unei pagini web, în timp ce CSS se ocupă de stilizare și prezentare, personalizând aspectul elementelor HTML prin definirea de culori, fonturi, spațiere și alte elemente de design. JavaScript adaugă un strat de interactivitate, permițând manipularea dinamică a elementelor HTML și CSS, și răspunde la evenimentele utilizatorului, precum clicurile pe butoane sau introducerea datelor în formulare. Aceste tehnologii colaborează îndeaproape: CSS poate fi inclus în documentele HTML fie inline, direct în etichete, fie prin referințe externe către fișiere CSS, ceea ce oferă o mare flexibilitate în gestionarea stilurilor. JavaScript, lucrând împreună cu HTML, folosește Document Object Model (DOM) pentru a modifica dinamic conținutul și structura paginii web. Această interacțiune permite dezvoltarea de aplicații web single-page, care oferă o experiență fluidă și rapidă, similară aplicațiilor desktop.

CSS, prin capacitatea sa de a crea design-uri responsive, asigură adaptarea paginilor web la diferite dimensiuni de ecran, un aspect crucial în era tehnologiei mobile. De asemenea, CSS permite dezvoltatorilor să definească stiluri diferite pentru aceleași elemente HTML în funcție de context, cum ar fi diferite stiluri pentru imprimare și vizualizare pe ecran. Pe de altă parte, JavaScript poate valida datele din formularele HTML înainte de a fi trimise la server, îmbunătățind astfel interactivitatea și experiența utilizatorului.

În final, combinația dintre HTML, CSS și JavaScript este esențială în dezvoltarea web modernă, fiecare tehnologie contribuind semnificativ la funcționalitate, estetică și interactivitatea paginilor web. Această interdependență asigură că paginile web nu sunt doar funcționale, ci și plăcute vizual și ușor de utilizat.

## 2. CSS. Prezentare generală

CSS, sau Cascading Style Sheets, este esențial în definirea stilului și prezentării paginilor web. Acest limbaj de stilizare controlează cum sunt afișate elementele HTML, transformând paginile web din texte simple în interfețe bogate și atractive vizual. CSS permite separarea conținutului de design, facilitând astfel întreținerea și actualizarea site-urilor. Stilurile CSS pot fi aplicate direct în documentele HTML sau pot fi separate în fișiere externe, ceea ce eficientizează încărcarea paginilor prin posibilitatea stocării în cache a stilurilor.

Fișierele CSS folosesc selecții pentru a identifica elementele HTML care vor fi stilizate, permițând controlul asupra unei game largi de aspecte precum culoare, dimensiunea fontului și spațierea. Versiunea actuală, CSS3, introduce funcționalități avansate precum animații, tranziții și design responsive, esențiale pentru adaptarea paginilor la diferite dispozitive și dimensiuni de ecran. Designul responsive, în particular, asigură o experiență de vizualizare optimă pe orice dispozitiv, fiind un aspect crucial în era digitală.

Metodologia BEM (Block, Element, Modifier) este o convenție de numire în CSS care sprijină crearea unui cod organizat și clar. Această metodologie împarte stilurile CSS în blocuri logice, elemente și modificatori, simplificând procesul de dezvoltare și mentenanță. Blocurile sunt componente independente reutilizabile, elementele sunt părți ale blocurilor care îndeplinesc funcții specifice, iar modificatorii reprezintă variații ale blocurilor sau elementelor care schimbă aspectul sau comportamentul acestora.

Adoptarea BEM în CSS duce la reducerea conflictelor de stil și facilitează debugging-ul, fiind deosebit de utilă în proiectele mari cu componente reutilizabile. Deși poate crește dimensiunea claselor în HTML, beneficiile în termeni de claritate, mentenabilitate și colaborare în cadrul echipei sunt semnificative. BEM ajută la menținerea consistenței stilului și facilitează modularizarea, permițând dezvoltatorilor să gestioneze eficient stilurile chiar și în proiecte extinse.

CSS, îmbunătățit prin metodologia BEM, transformă modul în care sunt stilizate paginile web. Această combinație promovează o abordare clară și sistematică în definirea stilurilor, îmbunătățind scalabilitatea și performanța site-urilor. BEM ajută la identificarea rapidă a relațiilor dintre HTML și CSS, făcând codul mai accesibil pentru dezvoltatori și reducând timpul necesar pentru debug și refactorizare. În concluzie, CSS, îmbunătățit cu BEM, este un instrument puternic pentru crearea de pagini web estetice, funcționale și ușor de întreținut, esențial în dezvoltarea web modernă.

## 3.3. Limbaj de scriptare. Python și framework-ul FLASK

Python și Flask reprezintă un duet puternic în dezvoltarea web. Python este un limbaj de programare de nivel înalt, cunoscut pentru sintaxa sa clară și lizibilă, care facilitează scrierea de cod curat și bine structurat. Este extrem de versatil, fiind utilizat într-o varietate de domenii, de la dezvoltarea web până la știința datelor și inteligența artificială.

Flask, pe de altă parte, este un micro-framework pentru Python, destinat dezvoltării aplicațiilor web. Este caracterizat prin simplitatea și flexibilitatea sa, oferind un nucleu de bază pentru construirea aplicațiilor web, dar permițând extinderea și personalizarea prin diverse extensii și plugin-uri. Flask este potrivit atât pentru proiecte mici și simple, cât și pentru aplicații web complexe. Unul dintre avantajele majore ale utilizării Flask în combinație cu Python este ușurința de a integra tehnologii frontend, cum ar fi HTML, CSS și JavaScript. Flask folosește un sistem de șabloane Jinja2 pentru a genera dinamic conținutul HTML, permițând dezvoltatorilor să creeze interfețe utilizator interactive și responsive.

Dezvoltarea backend-ului cu Flask implică crearea de rute și funcții view, care gestionează cererile și răspunsurile HTTP. Rutele sunt definituri URL care indică Flask ce funcție să execute când un anumit URL este accesat. Aceste funcții view pot prelua date din baze de date, procesa informații și returna răspunsuri sub forma paginilor web sau a datelor JSON. Python și Flask oferă, de asemenea, suport puternic pentru lucrul cu baze de date. Flask poate fi integrat cu diferite sisteme de gestionare a bazelor de date, cum ar fi SQLite, PostgreSQL sau MySQL, folosind extensii precum Flask-SQLAlchemy. Aceasta permite dezvoltatorilor să interacționeze cu baze de date printr-o interfață de nivel înalt, facilitând operațiunile CRUD (Create, Read, Update, Delete).

Pentru aplicațiile mai complexe, Flask permite dezvoltarea de API-uri RESTful, ceea ce înseamnă că frontend-ul poate comunica cu backend-ul prin intermediul unor endpoint-uri definite, făcând schimb de date în mod eficient. Acest lucru este esențial pentru aplicațiile single-page (SPA) și pentru aplicațiile web moderne care necesită o interacțiune dinamică între client și server. În plus, securitatea este un aspect cheie în dezvoltarea web, iar Flask oferă mecanisme integrate pentru protejarea aplicațiilor. De exemplu, Flask-WTF este o extensie populară care facilitează lucrul cu formularele web și include protecții împotriva unor vulnerabilități comune, cum ar fi Cross-Site Request Forgery (CSRF).

În concluzie, Python și Flask reprezintă o alegere excelentă pentru dezvoltatorii care doresc să creeze aplicații web puternice și eficiente, beneficiind de flexibilitatea și puterea Pythonului. Cu un ecosistem bogat și o comunitate activă, Python și Flask continuă să fie atractive atât pentru începători, cât și pentru dezvoltatori experimentați.

## 3.4. Administrarea bazelor de date folosind MySQL

MySQL este un sistem de management al bazelor de date relaționale (RDBMS), bazat pe limbajul SQL (Structured Query Language). Este recunoscut pentru performanța, fiabilitatea și ușurința de utilizare, fiind unul dintre cele mai populare RDBMS-uri în contextul dezvoltării aplicațiilor web, inclusiv în combinație cu Python și Flask. În administrarea bazelor de date, MySQL permite stocarea, recuperarea, modificarea și gestionarea eficientă a datelor structurate. Se caracterizează prin flexibilitatea sa în manipularea datelor, oferind suport pentru diverse tipuri de date, inclusiv numerice, text și date binare.

MySQL implementează un model client-server, unde serverul gestionează bazele de date și clientul interacționează cu serverul pentru a efectua operațiuni asupra datelor. Acest model facilitează accesul la distanță și gestionarea centralizată a datelor. Sistemul suportă tranzacții, care sunt esențiale pentru menținerea integrității datelor în aplicații multi-utilizatori și multi-proces. Tranzacțiile în MySQL permit executarea unor serii de operații ca o unitate singulară, garantând că toate operațiunile se efectuează complet sau deloc.

MySQL include un sistem robust de privilegii, care permite administratorilor să controleze accesul la date pe baza utilizatorilor și a rolurilor acestora. Acest mecanism asigură securitatea datelor și previne accesul neautorizat. În contextul dezvoltării web cu Python și Flask, MySQL se integrează eficient, permitând dezvoltatorilor să creeze aplicații web dinamice care interacționează cu baze de date. Flask suportă integrarea cu MySQL prin intermediul extensiilor precum Flask-MySQLdb și Flask-SQLAlchemy.

SQLAlchemy, un ORM (Object-Relational Mapper) popular în ecosistemul Python, facilitează lucrul cu MySQL prin abstractizarea interogărilor SQL. Acest lucru permite dezvoltatorilor să scrie cod Python în loc de SQL nativ, îmbunătățind lizibilitatea și mentenabilitatea codului. În operațiunile de zi cu zi, MySQL oferă o serie de unelte pentru optimizarea performanței, cum ar fi indexarea, care accelerează recuperarea datelor. De asemenea, suportă proceduri stocate și declanșatoare (triggers), permițând automatizarea unor operațiuni complexe pe server. MySQL este de asemenea compatibil cu replicarea, o tehnică esențială pentru distribuirea datelor între mai multe servere, ceea ce îmbunătățește disponibilitatea și fiabilitatea datelor.

În concluzie, MySQL reprezintă o soluție robustă și eficientă pentru administrarea bazelor de date în aplicații web, oferind un set de caracteristici avansate care se potrivesc nevoilor diversificate ale dezvoltatorilor, inclusiv în contextul utilizării Python și Flask. Capacitatea sa de a se integra cu ușurință în diferite medii de dezvoltare, împreună cu suportul său puternic pentru operațiuni bazate pe date, îl face un instrument valoros în arsenalul oricărui dezvoltator web.

## 3.5. Severul OpenServer

OpenServer, cunoscut și ca Open Server sau OSPanel, este un software care facilitează crearea unui mediu de dezvoltare local pentru aplicații web. Este o soluție amplu utilizată pentru testarea și dezvoltarea aplicațiilor web pe sistemele de operare Windows, oferind un mediu controlat și flexibil pentru dezvoltatori.

Utilizarea OpenServer în contextul dezvoltării unei aplicații web, cum ar fi cele bazate pe Python, Flask și MySQL, oferă mai multe avantaje semnificative. Primul este capacitatea de a configura rapid un mediu de dezvoltare local. OpenServer include componente esențiale precum un server web (de exemplu, Apache sau Nginx), un interpretor PHP, și un sistem de gestionare a bazelor de date, cum ar fi MySQL. Aceasta înseamnă că dezvoltatorii pot configura și rula aplicații web pe propriul computer, fără a necesita un server web extern sau resurse de hosting. Un alt avantaj este ușurința cu care se pot efectua teste. Dezvoltarea locală cu OpenServer permite testarea funcționalităților aplicației într-un mediu izolat, fără a afecta mediul de producție. Acest lucru este esențial pentru identificarea și remedierea erorilor înainte de lansarea aplicației.

OpenServer oferă, de asemenea, flexibilitate în configurare. Dezvoltatorii pot alege versiuni specifice pentru PHP, MySQL și alte componente, asigurând compatibilitatea cu cerințele specifice ale proiectului lor. Această personalizare ajută la crearea unui mediu de dezvoltare care să imite cât mai fidel mediul de producție. În contextul aplicațiilor web bazate pe Python și Flask, OpenServer poate fi utilizat pentru a găzdui aplicația local, permițând dezvoltatorilor să ruleze, să testeze și să depaneze aplicația într-un mediu controlat. Chiar dacă Flask are propriul server de dezvoltare, OpenServer poate oferi o abordare mai apropiată de un mediu de producție real.

Integrarea cu baze de date, cum ar fi MySQL, este simplificată în OpenServer. Oferă un mediu local pentru configurarea și gestionarea bazelor de date, care poate fi deosebit de util pentru dezvoltarea și testarea interacțiunilor cu baza de date înainte de implementarea pe un server live. De asemenea, OpenServer include unelte pentru administrarea și monitorizarea performanței serverului și a aplicațiilor, facilitând identificarea problemelor de performanță și gestionarea resurselor.

În concluzie, OpenServer este o unealtă valoroasă pentru dezvoltatorii de aplicații web, oferind un mediu local complet pentru dezvoltare, testare și depanare. Este compatibil cu tehnologii populare precum Python, Flask și MySQL, făcându-l ideal pentru dezvoltarea aplicațiilor web într-un mediu Windows. Utilizarea sa în cadrul dezvoltării aplicațiilor reduce complexitatea configurării și accelerează ciclul de dezvoltare, permițând dezvoltatorilor să se concentreze mai mult pe dezvoltarea funcționalităților și mai puțin pe gestionarea infrastructurii.

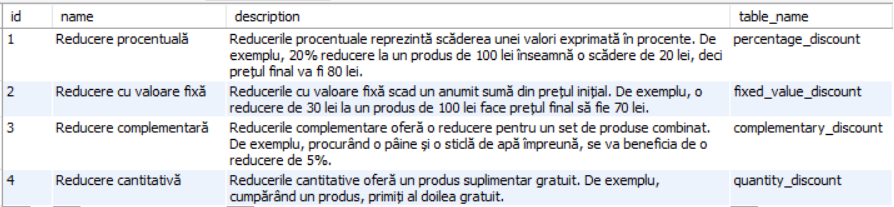
# 4. Structura Bazei de Date

Structurarea corectă a unei baze de date, în special în cazul bazelor de date relaționale precum MySQL, este un element cheie pentru funcționarea eficientă a oricărui sistem bazat pe date. Prin modul în care sunt organizate și stocate datele, această structurare afectează direct eficiența, securitatea și performanța întregului sistem.

O structură bine elaborată a bazei de date presupune nu doar crearea tabelelor, dar și definirea modului în care acestea interacționează între ele, plus stabilirea tipurilor de date pentru stocarea informațiilor. Luând exemplul unei librării, s-ar putea avea tabele pentru cărți, autori și clienți, iar relațiile dintre aceste tabele sunt fundamentale pentru a extrage informații utile. Elemente precum cheile primare și secundare, alături de restricțiile de integritate referențială, joacă un rol crucial în menținerea integrității datelor și în prevenirea incoerențelor sau erorilor. O structură bine concepută nu doar că simplifică interogările și accesul la date, ci este și vitală pentru dezvoltarea aplicațiilor care interacționează cu baza de date. Performanța bazei de date este, de asemenea, strâns legată de structura acesteia. Optimizarea tabelelor, interogărilor și indecșilor poate îmbunătăți semnificativ eficiența și reduce consumul de resurse.

În concluzie, rolul structurii bazei de date în cadrul bazelor de date relaționale MySQL este fundamental. O structură bine proiectată garantează o organizare eficientă a datelor, securitate sporită și performanță optimă, toate fiind esențiale pentru funcționarea corespunzătoare a aplicațiilor și pentru o gestionare fiabilă a datelor.

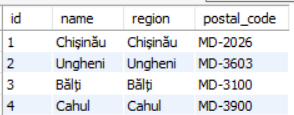
## 4.1. Tabelul discount\_type

Tabelul discount\_type include un identificator unic pentru fiecare tip de reducere (id), care este, de asemenea, cheia primară. Numele reducerii (name) este stocat ca un șir de caractere și este obligatoriu, indicând tipul de reducere, precum o reducere procentuală sau o valoare fixă. Descrierea (description) este un câmp opțional care poate conține până la 255 de caractere, oferind detalii suplimentare despre tipul de reducere. În final, table\_name specifică numele tabelului asociat reducerii și este, de asemenea, un câmp obligatoriu.

**Figura 4.1** Reprezentarea tabelului discount\_type în baza de date

## 4.2. Tabelul city

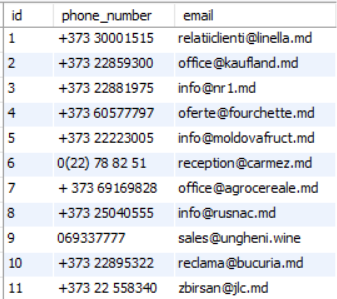
Tabelul city este creat pentru a organiza informațiile referitoare la orașe. Fiecare oraș este identificat unic prin coloana id, care servește drept cheie primară. Numele orașului este înregistrat în coloana name, care este o șir de caractere de lungime maximă 20 și este marcat ca fiind un câmp necesar. Coloana region indică regiunea specifică orașului și este, de asemenea, un câmp obligatoriu cu o lungime similară. În plus, fiecare oraș are asociat un cod poștal în coloana postal\_code, care este un șir de caractere de până la 10 caractere.



**Figura 4.2** Reprezentarea tabelului city în baza de date

## 4.3. Tabelul contact\_info

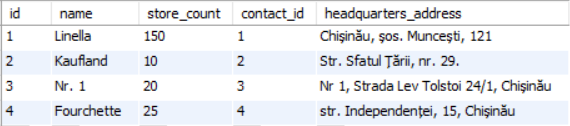
Fiecare înregistrare are un identificator unic reprezentat de coloana id. Numerele de telefon sunt stocate în coloana phone\_number, care acceptă șiruri de caractere de până la 15 caractere. Adresele de e-mail sunt păstrate în coloana email, care are o lungime maximă de 50 de caractere, suficient pentru a acomoda majoritatea adreselor de e-mail standard.



**Figura 4.3** Reprezentarea tabelului contact\_info în baza de date

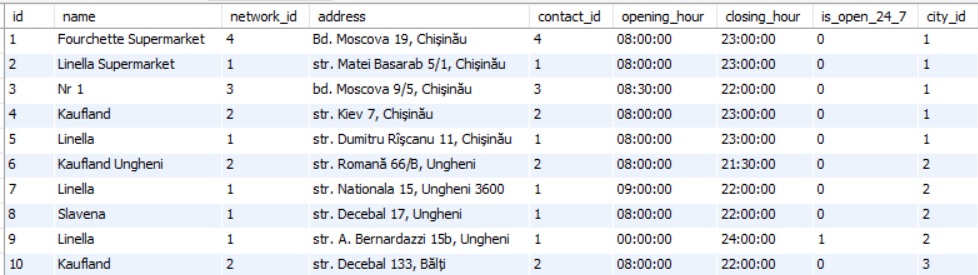
## 4.4. Tabelul store\_network

Tabelul store\_network este proiectat pentru a cataloga rețelele de magazine, având un câmp id care servește drept cheie primară și este auto-incrementat pentru a asigura un identificator unic fiecărei înregistrări. Numele rețelei de magazine este înregistrat în coloana name, care este obligatoriu și restricționat la 20 de caractere. Numărul de magazine din rețea este reflectat în coloana store\_count. Legătura cu informațiile de contact este realizată prin coloana contact\_id, care este un câmp obligatoriu și presupune existența unei relații cu tabelul contact\_info. Adresa sediului central este stocată în headquarters\_address, permițând șiruri de caractere de până la 100, pentru a acomoda adresele complete.



**Figura 4.4** Reprezentarea tabelului store\_network în baza de date

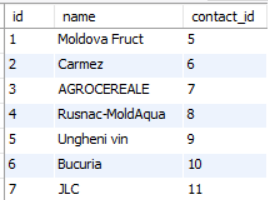
## 4.5. Tabelul store

Tabelul store este configurat pentru a gestiona informațiile specifice magazinelor individuale. Fiecare magazin are un id unic, auto-incrementat, care funcționează ca o cheie primară. Numele magazinului este înregistrat în coloana name. network\_id este o legătură străină ce referă rețeaua de magazine din care face parte magazinul. Adresa fizică a magazinului este capturată în coloana address. Relația cu informațiile de contact este menținută prin contact\_id, care este necesar pentru fiecare înregistrare. Orele de deschidere și închidere sunt indicate în opening\_hour și closing\_hour. is\_open\_24\_7 este un câmp boolean care indică dacă magazinul funcționează non-stop. city\_id leagă magazinul de orașul în care se află.

**Figura 4.5** Reprezentarea tabelului store în baza de date

## 4.6. Tabelul manufacturer

Tabelul manufacturer este proiectat pentru a înregistra producătorii și detaliile lor de contact. Fiecare producător este atribuit un id. Numele producătorului este capturat în coloana name, care este obligatorie și poate conține până la 40 de caractere. Fiecare înregistrare include, de asemenea, un contact\_id, care este un câmp obligatoriu și se presupune că face referire la o înregistrare corespunzătoare într-un tabel separat de informații de contact. Această structură asigură că toate detaliile esențiale ale producătorului sunt accesibile și bine organizate pentru utilizare.



**Figura 4.6** Reprezentarea tabelului manufacturer în baza de date

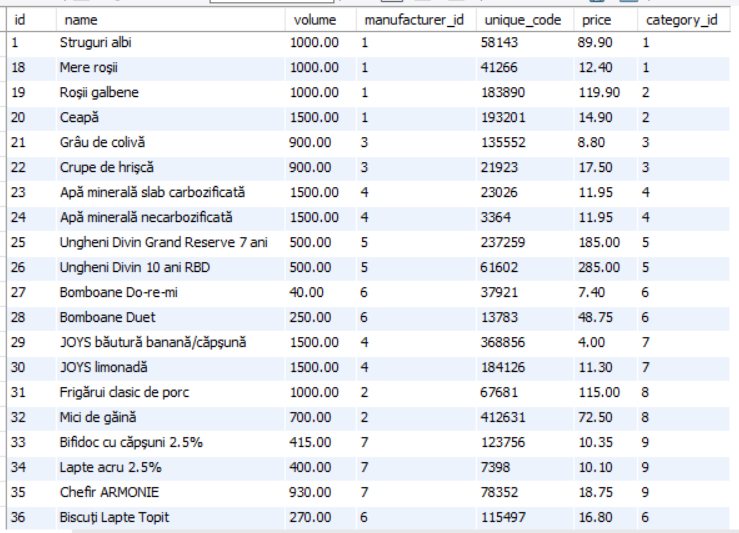
## 4.7. Tabelul product\_category

Tabelul product\_category este destinat clasificării produselor în diferite categorii. Fiecare categorie de produs este identificată printr-un id care este generat automat și este setat ca cheie primară, garantând unicitatea fiecărei categorii. Numele categoriei este stocat în coloana name, care este un câmp obligatoriu și poate conține până la 100 de caractere, astfel încât să permită descrieri detaliate ale categoriilor.



**Figura 4.7** Reprezentarea tabelului product\_category în baza de date

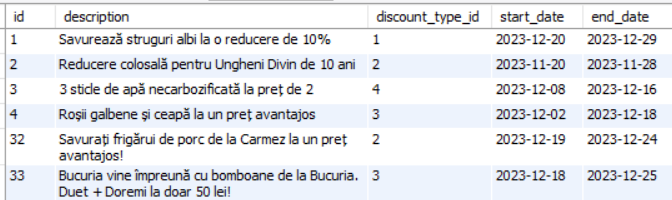
## 4.8. Tabelul product

Tabelul product este proiectat pentru a stoca informații despre diverse produse. Fiecare produs este identificat printr-un id care este generat automat, servind ca o cheie primară. Numele produsului este capturat în coloana name, limitat la 50 de caractere și este obligatoriu. Volumul, exprimat în unități cu două zecimale, este stocat în coloana volume. O legătură importantă cu producătorul este stabilită prin manufacturer\_id, care este necesar pentru fiecare produs. În plus, fiecare produs are un unique\_code, care este esențial pentru identificarea univocă și urmărirea produselor, și un preț (price), care este de asemenea obligatoriu și precis până la două zecimale. Fiecare produs este, de asemenea, asociat cu o categorie prin category\_id, indicând clasificarea sa într-un tabel de categorii.

**Figura 4.8** Reprezentarea tabelului product în baza de date

## 4.9. Tabelul discount

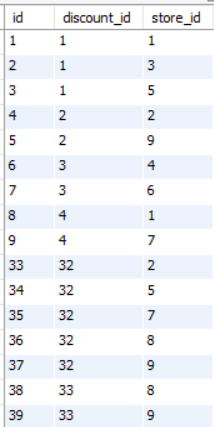
Tabelul discount este creat pentru a gestiona informații despre diverse oferte promoționale. În acest tabel, fiecare promoție este identificată de un id. Descrierea ofertei este detaliată în coloana description, care permite până la 255 de caractere și este obligatorie pentru a furniza informații clare despre promoție. Relația cu tipul de reducere aplicabil este indicată prin discount\_type\_id. Fiecare înregistrare are o start\_date și o end\_date, ambele obligatorii, care definesc perioada de valabilitate a reducerii.



**Figura 4.9** Reprezentarea tabelului discount în baza de date

## 4.10. Tabelul discount\_instance

Tabelul discount\_instance este proiectat pentru a asocia ofertele promoționale cu magazinele specifice unde sunt valabile. Fiecare înregistrare are un id unic generat automat, care funcționează ca o cheie primară. Coloana discount\_id face referire la o reducere specifică din tabelul discount, iar store\_id indică magazinul unde reducerea este aplicabilă, legându-se la un id din tabelul store. Această structură facilitează urmărirea locațiilor individuale în care diversele reduceri sunt oferite, permițând o gestionare eficientă a campaniilor promoționale la nivel de rețea de magazine.



**Figura 4.10** Reprezentarea tabelului discount\_instance în baza de date

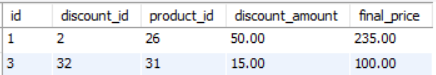
## 4.11. Tabelul percentage\_discount

Tabelul percentage\_discount este conceput pentru a înregistra informații despre reducerile procentuale aplicate produselor. Fiecare înregistrare în tabel are un id unic. Coloana discount\_id face referire la un identificator din tabelul discount, iar product\_id este legat de un identificator specific din tabelul product. Reducerea procentuală este reprezentată de discount\_percentage, o valoare întreagă mică (TINYINT) nesemnată, care este verificată pentru a fi între 1 și 99%, indicând cantitatea de reducere aplicată. Prețul final după aplicarea reducerii este înregistrat în final\_price, care este o valoare decimală cu două zecimale și este un câmp obligatoriu. Structura tabelului asigură că toate reducerile sunt urmărite corespunzător.

**Figura 4.11** Reprezentarea tabelului percentage\_discount în baza de date

## 4.12. Tabelul fixed\_value\_discount

Tabelul fixed\_value\_discount este proiectat pentru a gestiona reducerile cu valoare fixă aplicate produselor. Fiecare intrare are un id unic. Discount\_id se referă la un identificator din tabelul discount, iar product\_id se referă la un identificator din tabelul product. Suma fixă a reducerii este reprezentată în coloana discount\_amount, care este un număr decimal cu două zecimale și este necesară pentru a indica valoarea exactă a reducerii. Final\_price reprezintă prețul produsului după aplicarea reducerii și este, de asemenea, o valoare decimală obligatorie cu două zecimale.



**Figura 4.12** Reprezentarea tabelului fixed\_value\_discount în baza de date

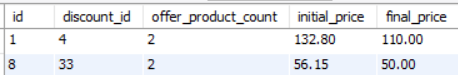
## 4.13. Tabelul quantity\_discount

Tabelul quantity\_discount este destinat înregistrării reducerilor cantitative, unde clienții primesc o cantitate de produse gratuit la achiziționarea unei cantități specificate. Fiecare linie este unic identificată de id, care este cheia primară și se auto-incrementează. Discount\_id se referă la un identificator din tabelul de reduceri, iar product\_id indică produsul la care se aplică reducerea. Required\_quantity este numărul de produse pe care clientul trebuie să le cumpere pentru a beneficia de promoție, iar free\_quantity reprezintă numărul de produse gratuite pe care le primește. Final\_price reprezintă costul total după aplicarea reducerii cantitative și este un câmp obligatoriu cu valoare decimală și două zecimale. Această structură permite urmărirea ofertelor promoționale care încurajează cumpărarea în volum.



**Figura 4.13** Reprezentarea tabelului quantity\_discount în baza de date

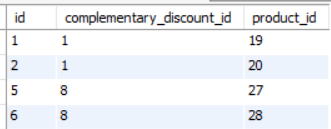
## 4.14. Tabelul complementary\_discount

Tabelul complementary\_discount este proiectat pentru a administra reducerile complementare, care sunt aplicate atunci când clienții achiziționează un set combinat de produse. Fiecare înregistrare în tabel are un id unic, care este auto-incrementat și funcționează ca o cheie primară. Discount\_id face referire la o reducere specificată în tabelul de reduceri, iar offer\_product\_count indică numărul de produse care trebuie achiziționate pentru a beneficia de reducerea respectivă. Initial\_price reprezintă prețul inițial al pachetului de produse, în timp ce final\_price reflectă prețul după aplicarea reducerii. Ambele prețuri sunt înregistrate ca valori zecimale cu două zecimale și sunt câmpuri obligatorii.

**Figura 4.14** Reprezentarea tabelului complementary\_discount în baza de date

## 4.15. Tabelul complementary\_discount\_product

Tabelul complementary\_discount\_product este proiectat să lege reducerile complementare de produsele specifice la care se aplică. Fiecare înregistrare este unic identificată de un id, care se auto-incrementează și servește ca o cheie primară. Complementary\_discount\_id este o referință la o anumită reducere din tabelul complementary\_discount, iar product\_id indică produsul care face parte din oferta de reducere complementară. Prin urmare, tabelul servește ca un mecanism de asociere între reducerile oferite și produsele individuale din baza de date, asigurând o gestiune clară a campaniilor promoționale.



**Figura 4.15** Reprezentarea tabelului complementary\_discount\_product în baza de date

# 5. Structura Aplicației WEB

Capitolul privind structura aplicației web prezintă arhitectura și componentele tehnologice ale soluției dezvoltate. În fundamentul interfeței utilizatorului stau HTML și CSS, care definesc scheletul și stilizarea paginilor web, îmbunătățite prin utilizarea SCSS pentru o gestionare mai eficientă a stilurilor. JavaScript este folosit pentru a adăuga interactivitate și dinamism în partea de client a aplicației. Pe partea de server, Python, împreună cu microframework-ul Flask, facilitează construirea backend-ului, oferind un mecanism flexibil pentru routing și gestionarea cererilor HTTP. Interacțiunea cu baza de date MySQL este realizată prin intermediul ORM-ului SQLAlchemy, care oferă o abstractizare puternică a operațiilor de bază de date, permițând o manipulare obiect-orientată a datelor. Împreună, aceste tehnologii oferă o fundație solidă pentru dezvoltarea unei aplicații web robuste, scalabile și ușor de întreținut. Structura aplicației este astfel concepută pentru a facilita extensibilitatea și pentru a susține bunele practici de dezvoltare software.

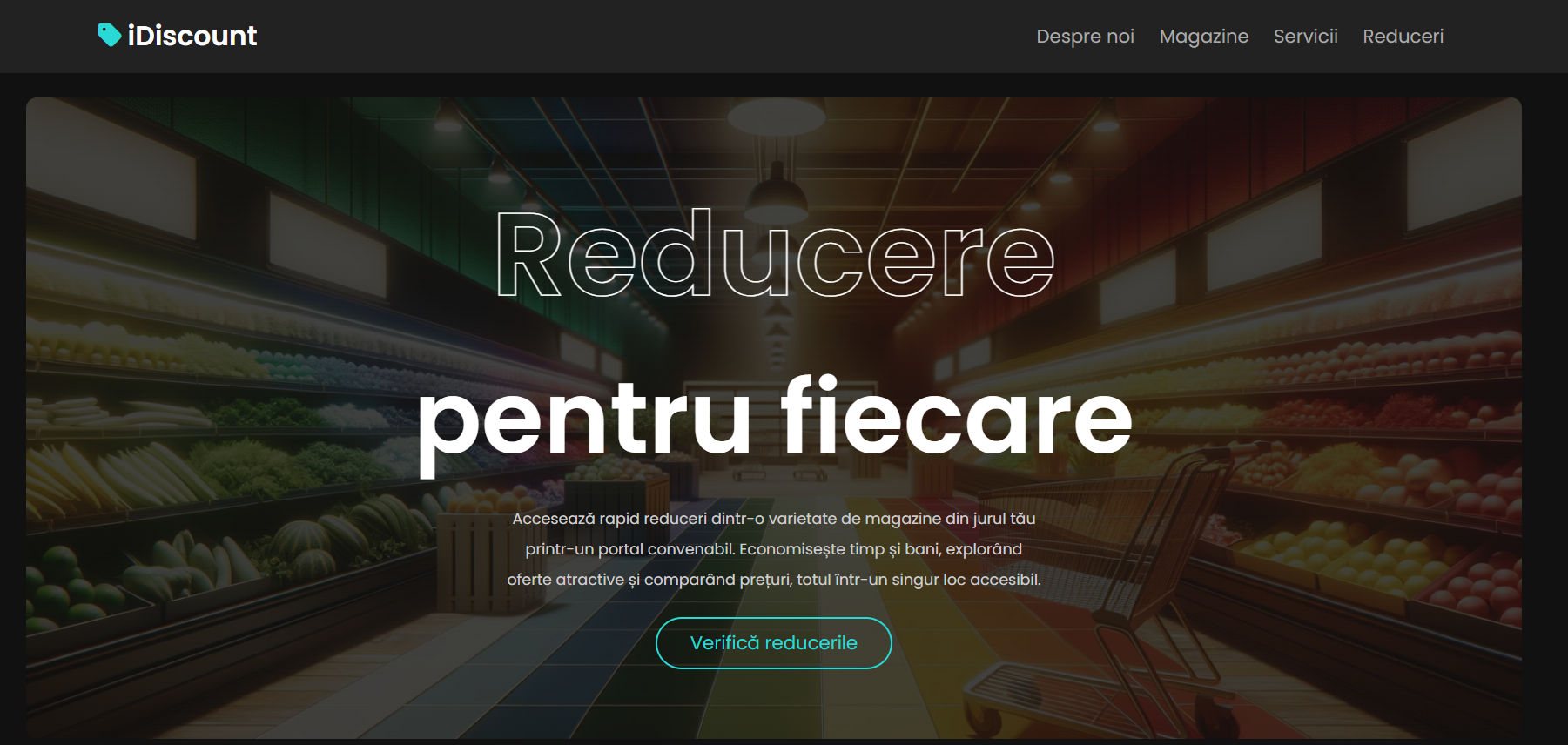
## 5.1. Scopul aplicației WEB

Scopul platformei web dezvoltate este de a oferi o soluție centralizată pentru monitorizarea și gestionarea reducerilor comerciale. Aplicația se adresează în primul rând utilizatorilor care doresc să urmărească reducerile disponibile, permițându-le să filtreze ofertele după preț, categorie și disponibilitatea în diferite magazine, oferind în același timp acces la informații detaliate despre fiecare reducere în parte. Utilizatorii pot astfel să identifice rapid cele mai relevante promoții pentru nevoile și preferințele lor.

Pentru administratorii rețelei, platforma oferă un set de instrumente robuste pentru managementul reducerilor. Aceștia au posibilitatea de a introduce noi oferte promoționale, de a modifica reducerile existente și de a verifica informații detaliate despre reducerile active. Prin această funcționalitate, administratorii pot asigura că reducerile sunt actualizate și relevante, contribuind la menținerea unui nivel înalt de angajament din partea utilizatorilor și la eficientizarea procesului de vânzare. Astfel, platforma servește ca un canal de comunicare eficient între comercianți și clienți, stimulând activitatea comercială și satisfacția consumatorilor.

## 5.2. Descrierea aplicației WEB

Pagina de start a platformei web iDiscount este structurată pentru a facilita accesul rapid la informații, respectând principiile ergonomice și de usability în designul interfeței. Elementul central al paginii este un titlu de dimensiuni considerabile, care comunică obiectivul platformei și funcționează ca un element de focalizare vizuală pentru utilizatori.

Bara de navigație este poziționată strategic în partea superioară a ecranului, urmând convențiile standard de design web, pentru a asigura o experiență de utilizare intuitivă. Imaginea de fundal, care prezintă un ambient de supermarket, este alesă pentru a reflecta domeniul de aplicare al serviciului web, fără a distrage atenția de la mesajul textual. Butonul de acțiune 'Verifică reducerile' este plasat într-un mod care să încurajeze interacțiunea imediată, utilizându-se o schemă de culori care să contrasteze cu elementele din fundal pentru o vizibilitate sporită.

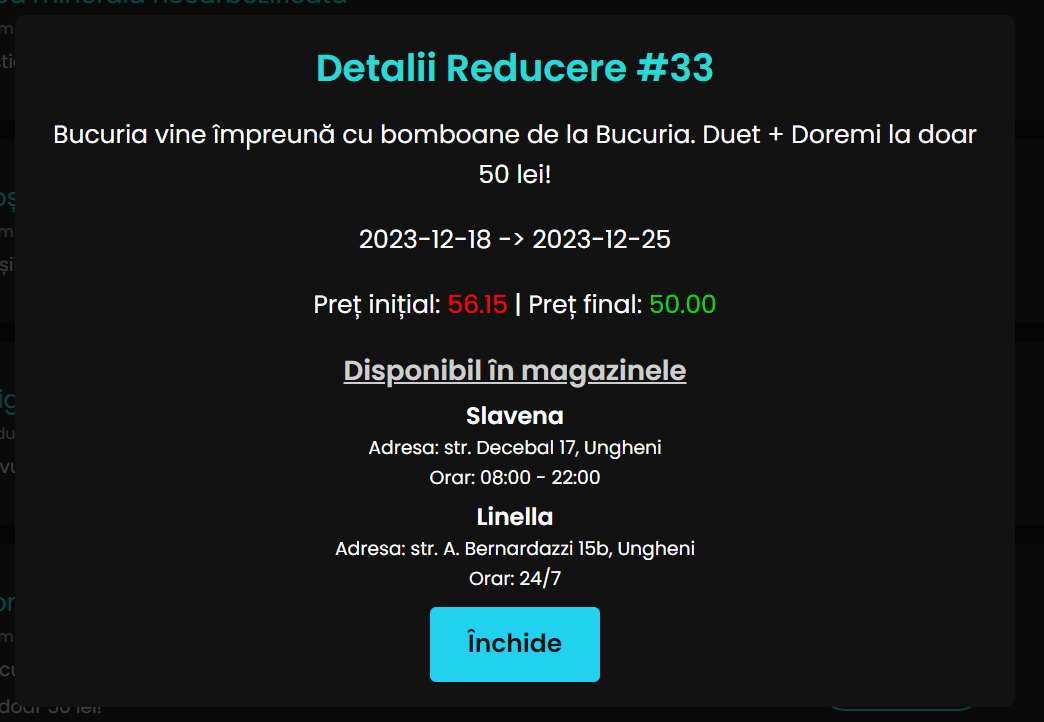
**Figura 5.1** Pagina de start a platformei

Pagina prezentată în fig. 5.2 este o componentă a interfeței utilizator a platformei iDiscount, care servește ca un instrument de filtrare și vizualizare a reducerilor. În partea superioară, utilizatorii sunt îndemnați să „Selectați un produs” printr-un titlu clar. Sub acesta, interfața oferă opțiuni de filtrare, permitând utilizatorilor să aleagă o categorie de produse dintr-un meniu derulant și să specifice un interval de preț prin introducerea valorilor minime și maxime în câmpurile dedicate. Posibilitatea de a căuta reduceri particulare poate fi extinsă prin introducerea unor asemenea filtre ca magazine unde sunt valabile reducerile, căutare specifică după produs, căutare după producător și etc.

Sub secțiunea de filtrare, sunt listate reducerile disponibile, fiecare având un card distinct în care sunt afișate numele produsului, detalii ale reducerii, precum și prețul inițial tăiat, urmat de prețul redus, evidențiind astfel economia realizată. Fiecare card include și un buton 'Mai mult', sugerând posibilitatea de a accesa detalii suplimentare. Designul este coerent, folosind o paletă de culori contrastante pentru a distinge între diferitele elemente ale interfeței și pentru a ghida ochiul utilizatorului prin informațiile prezentate. În colțul dreapta sus al paginii se găsește un link de navigare către „Pagina principală”, oferind o cale rapidă de întoarcere la pagina de start a site-ului.



**Figura 5.2** Pagina de afișare a reducerilor

Figura 5.3 afișează o modalitate pop-up a platformei iDiscount, care furnizează detalii suplimentare despre o anumită reducere, identificată prin numărul #33. În centrul acestui pop-up, descrierea ofertei este prezentată în termeni clari: "Bucuria vine împreună cu bomboane de la Bucuria. Duet + Doremi la doar 50 lei!" Aceasta este urmată de perioada de valabilitate a reducerii, indicată printr-un interval de date de la 18 decembrie 2023 la 25 decembrie 2023. Detaliile financiare sunt expuse printr-o comparație între prețul inițial și prețul final al produsului, subliniind economia realizată prin aplicarea reducerii. În plus, informațiile privind disponibilitatea produsului sunt structurate pe magazine, cu menționarea numelui magazinului, adresei și orele de funcționare, oferind astfel date esențiale pentru localizarea fizică a ofertei. Modalitatea include un buton de închidere, "Închide", plasat în partea inferioară, permițând utilizatorilor să revină la interfața principală a platformei.

**Figura 5.3** Afișarea detaliilor despre o reducere

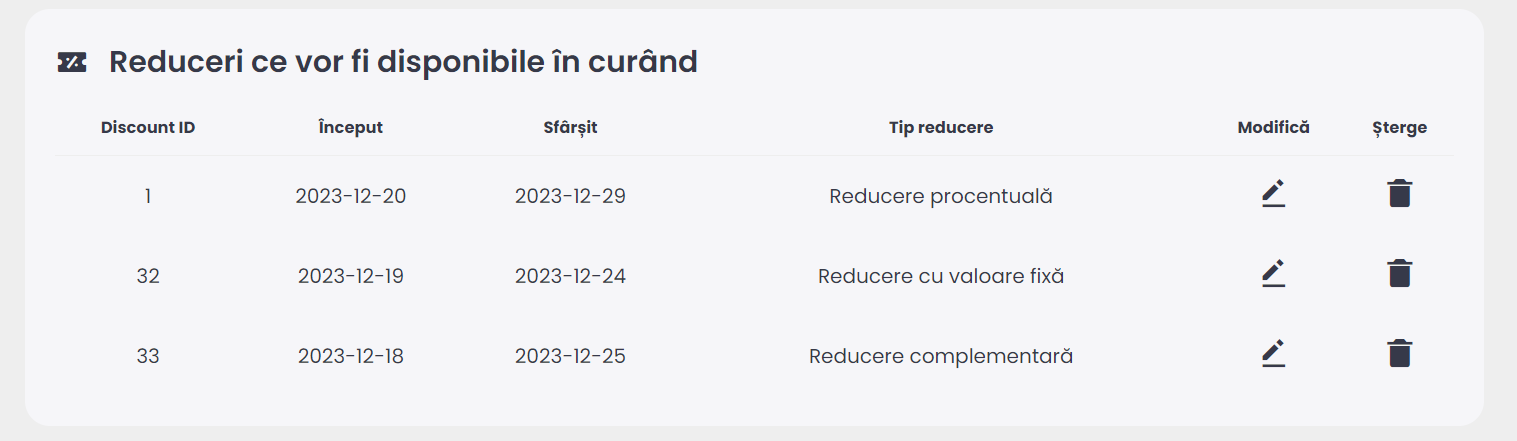
Figura 5.4 reprezintă interfața de administrare pentru rețeaua Linella, destinată gestiunii reducerilor. Panoul de control, sau dashboard-ul, este organizat în trei secțiuni principale, fiecare prezentând o categorie diferită de reduceri: active, anterioare și planificate. Aceste categorii sunt reprezentate de carduri cu simboluri intuitive și un rezumat numeric al reducerilor disponibile în fiecare categorie.

**Figura 5.4** Dashboardul disponibil administratorilor

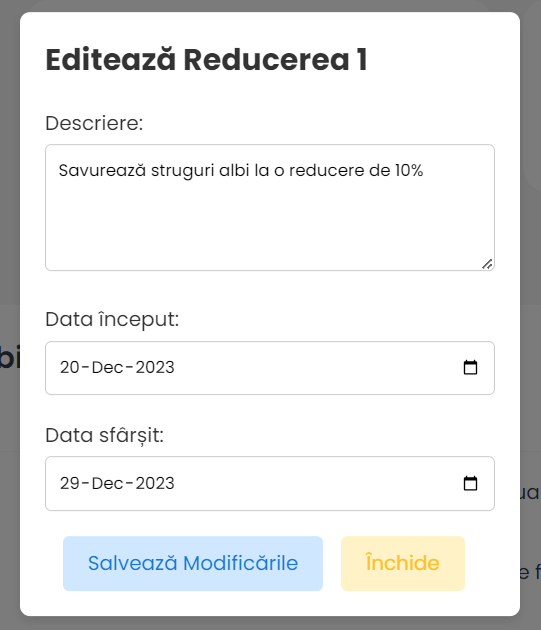
Cardul pentru reducerile active indică existența unei singure oferte în vigoare și include un buton de acțiune pentru vizualizarea detaliilor. Reducerile anterioare sunt sumarizate în al doilea card, cu un total de două, și sunt, de asemenea, însoțite de un buton pentru examinarea acestora. Al treilea card evidențiază reducerile planificate, numărând trei în total, și dispune de un buton similar pentru acces. Centrat în partea de jos a imaginii este un buton proeminent, 'Adaugă o reducere', care invită la introducerea de noi oferte promoționale. Sub acesta, se află o altă secțiune, 'Selectați un tip de reducere', sugerând posibilitatea de a clasifica reducerile noi adăugate în sistem. Designul interfeței este curat și minimalist, favorizând o navigare eficientă și un management simplificat al reducerilor pentru administratorii rețelei.

Figura 5.5 reprezintă un ecran din cadrul platformei iDiscount, specific secțiunii de administrare a reducerilor, ilustrând o listă a promoțiilor ce urmează să fie active. Fiecare rând din tabel corespunde unei reduceri distincte și este atribuit un identificator numeric unic sub coloana 'Discount ID'. Datele de începere și de încheiere a fiecărei promoții sunt clar definite, oferind administratorului o imagine de ansamblu asupra cronologiei ofertelor.

Diversitatea reducerilor este subliniată în coloana 'Tip reducere', care clasifică fiecare promoție în funcție de mecanismul său de aplicare. Posibilitățile de interacțiune cu fiecare reducere sunt reprezentate prin două acțiuni: una pentru editarea detaliilor reducerii ('Modifică') și cealaltă pentru eliminarea acesteia din sistem ('Șterge'), ambele ilustrate prin pictograme intuitive.

În esență, acest ecran funcționează ca un instrument de planificare și coordonare a activităților promoționale în cadrul rețelei de magazine Linella, oferind administratorilor capacitatea de a supraveghea și ajusta campaniile de marketing într-un mod eficient și structurat.

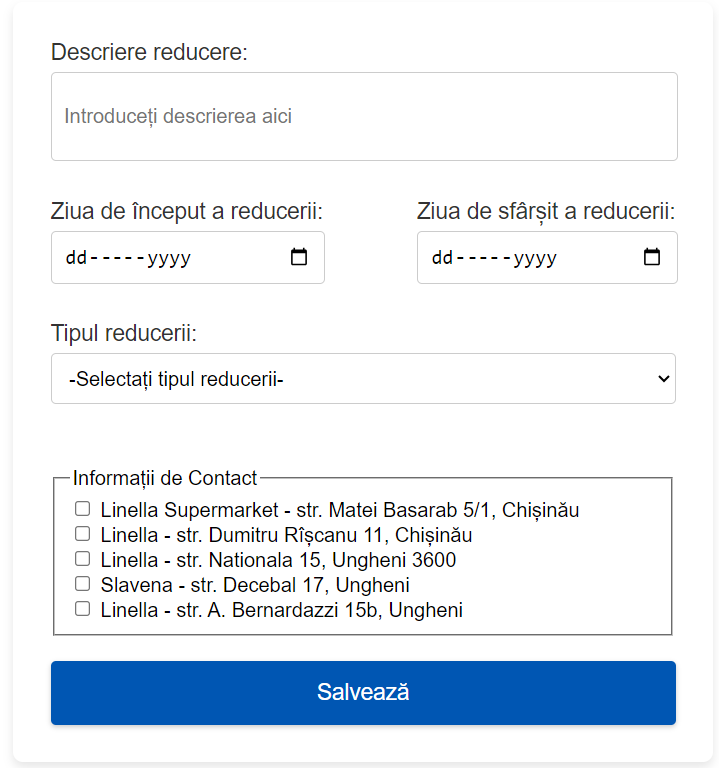
**Figura 5.5** Lista reducerilor planificate

Interfața de editare din figura 5.6 permite utilizatorilor să modifice detalii specifice reducerii curente, identificate prin numărul 1. În partea superioară a ferestrei, utilizatorii sunt îndemnați să editeze reducerea prin intermediul unui titlu clar. Câmpul 'Descriere' oferă un spațiu text unde pot fi introduse sau modificate detaliile ofertei, în acest caz o reducere de 10% pentru struguri albi. Sub aceasta, sunt furnizate două câmpuri de date, 'Data început' și 'Data sfârșit', care permit stabilirea perioadei de valabilitate a promoției. Aceste câmpuri includ un selector de dată pentru o ajustare precisă a intervalului promoțional. La baza ferestrei sunt poziționate două butoane de acțiune, 'Salvează Modificările' și 'Închide', care oferă opțiunile de a finaliza editarea sau de a se retrage din proces fără a salva modificările. Designul este intuitiv și utilizator-friendly, cu o schemă de culori și elemente grafice care păstrează unitatea vizuală cu restul platformei.

**Figura 5.6** Modificarea detaliilor despre o reducere

În imaginea 5.7 este afișată o formă de introducere pentru o nouă reducere în sistemul iDiscount. În primul câmp, utilizatorii sunt invitați să ofere o 'Descriere reducere', cu un spațiu text pentru detalii. Următoarele două câmpuri sunt destinate specificării 'Zilei de început a reducerii' și 'Zilei de sfârșit a reducerii', ambele cu spații de completare și butoane pentru deschiderea unui calendar de selecție a datei.

Sub acestea, se găsește un meniu derulant pentru alegerea 'Tipului reducerii', care permite selectarea categoriei de reducere aplicabile. În partea de jos a formularului, secțiunea 'Informații de Contact' prezintă o listă de casete de selectare asociate cu diferite locații ale rețelei Linella și o locație Slavena, oferind posibilitatea de a marca locațiile specifice unde va fi disponibilă reducerea.

Butonul 'Salvează' este poziționat la baza formularului, gata să confirme și să stocheze informațiile introduse de utilizator în sistem. Designul interfeței este simplu și direct, cu o structură clară și elemente interactive bine definite pentru o experiență de utilizare eficientă și lipsită de ambiguitate.

**Figura 5.7** Crearea unei reduceri noi

Figura 5.8 afișează un tabel din cadrul unei interfețe a platformei iDiscount, care este folosită pentru catalogarea produselor disponibile pentru reducere. Tabelul este structurat în coloane care furnizează informații detaliate despre fiecare produs: 'ID' este identificatorul unic al produsului, 'Nume Produs' indică denumirea articolului, 'Categorie' clasifică produsul într-o categorie precum fructe, legume sau carne, iar 'Producător' numește compania sau marca asociată. 'Volum' reprezintă cantitatea sau mărimea lotului de produse, exprimată în unități standard, iar 'Preț' afișează costul produsului în moneda locală.

**Figura 5.8** Lista de produse disponibile

Ultima coloană, 'Cod Unic', este atribuită fiecărui produs pentru identificare și urmărire în sistemul iDiscount. Tabelul oferă o viziune clară asupra inventarului de produse și este un instrument esențial în procesul de management al reducerilor în rețeaua de magazine. Designul este simplu și profesional, cu accent pe lizibilitate și accesibilitatea datelor.

## 5.3. Descrierea operațiilor CRUD cu tabelele Bazei de Date

CRUD reprezintă operațiile fundamentale de manipulare a datelor într-un sistem bazat pe baze de date, acronimul provenind de la Create (Creare), Read (Citire), Update (Actualizare), și Delete (Ștergere). Create se referă la capacitatea de a crea noi înregistrări în baza de date. Într-o interfață utilizator, acest lucru ar putea fi realizat prin completarea unui formular și apăsarea unui buton de 'Salvează' sau 'Adaugă'. Read implică citirea și afișarea datelor existente. Acest lucru este adesea realizat prin interogarea bazei de date și returnarea rezultatelor pentru a fi prezentate utilizatorilor, fie sub formă de liste, tabele sau alte forme de vizualizare. Update permite modificarea datelor existente. Aceasta este adesea realizată într-o interfață de editare unde utilizatorii pot modifica informații și apoi salva aceste schimbări înapoi în baza de date. Delete oferă posibilitatea de a elimina înregistrările din baza de date. Aceasta operație trebuie utilizată cu precauție, deoarece ștergerea este adesea ireversibilă.

### 5.3.1. Operația READ

În contextul aplicației web prezentate, funcția get\_products() implementează operația de Read. Funcția începe prin a executa o interogare la baza de date (Read), folosind ORM-ul SQLAlchemy pentru a face join între mai multe tabele: Product, Manufacturer, și ProductCategory. Îmbinarea acestor tabele permite colectarea unui set complet de informații despre produse, incluzând id-ul, numele, codul unic, volumul, prețul, numele producătorului și numele categoriei. Odată ce datele sunt extrase, acestea sunt convertite într-o listă de dicționare, cu fiecare dicționar reprezentând un produs și atribuții sale corespunzătoare. Acest pas transformă datele relaționale într-un format structurat JSON, care este apoi returnat și poate fi ușor afișat în interfața utilizatorului.

Controllerul get\_stores\_in\_network\_one() exemplifică o funcție de tip Read în cadrul operațiilor CRUD într-o aplicație web ce utilizează Flask și SQLAlchemy pentru interacțiunea cu o bază de date. Scopul acestei funcții este de a extrage și a returna informații despre toate magazinele dintr-o anumită rețea, identificată prin network\_id egal cu 1. În detaliu, funcția începe prin a defini o interogare SQL care selectează informații din tabela Store și, în același timp, efectuează un join cu tabela City pentru a include numele orașului în care fiecare magazin este situat. Filtrarea este realizată pe baza network\_id, asigurându-se că doar magazinele care aparțin rețelei specificate sunt selectate. După executarea interogării, rezultatele sunt convertite într-o listă de dicționare, cu fiecare dicționar reprezentând un magazin și incluzând detaliile sale, cum ar fi ID-ul, numele, adresa, orașul, orele de deschidere și închidere, și dacă este deschis 24/7. Acest lucru permite o serializare ușoară în format JSON, care este apoi returnată ca răspuns la solicitarea clientului. Metoda strftime este folosită pentru a formata timpul într-un format prietenos pentru orele de deschidere și închidere. În cazul unei excepții, funcția prinde eroarea și returnează un mesaj corespunzător de eroare și un cod de stare HTTP 500, indicând o eroare internă a serverului.

Controllerul get\_categories() ilustrează, de asemenea, o operație de Read în cadrul operațiilor CRUD pentru o aplicație web care utilizează Flask ca framework și SQLAlchemy ca ORM pentru interacțiunea cu baza de date. Funcția are scopul de a extrage toate categoriile de produse din baza de date. Procesul începe prin apelul metodei all() pe clasa ProductCategory, care este un model SQLAlchemy. Această metodă returnează o listă de toate instanțele ProductCategory existente în baza de date. Odată obținute, instanțele sunt iterate pentru a construi o listă de dicționare. Fiecare dicționar din această listă este compus din chei și valori reprezentând id-ul și name-ul fiecărei categorii. Aceasta transformă datele din formatul lor nativ ORM într-o structură de date JSON-friendly, care poate fi apoi trimisă ca răspuns la solicitarea HTTP. În cazul în care se întâlnește o excepție în timpul executării interogării sau a procesării datelor, funcția capturează excepția și returnează un obiect JSON care conține un mesaj de eroare, împreună cu un cod de răspuns 500, care indică o eroare de server internă.

### 5.3.2. Operația UPDATE

Controllerul update\_discount(discount\_id) implementează operația de Update din CRUD într-un context web folosind Flask și SQLAlchemy. Scopul funcției este de a actualiza o înregistrare existentă, identificată prin discount\_id, cu date noi primite printr-o solicitare HTTP. La începutul funcției, se extrag datele trimise de client prin intermediul request.get\_json(), care presupune că datele sunt trimise în format JSON. Controllerul apoi caută obiectul Discount corespunzător în baza de date folosind Discount.query.get(discount\_id). Dacă nu există nicio înregistrare corespunzătoare discount\_id-ului furnizat, se aruncă o excepție ValueError, informând că discountul specificat nu a fost găsit. În scenariul în care înregistrarea este găsită, funcția actualizează atributele obiectului discount cu noile valori furnizate de client, sau le păstrează pe cele existente dacă nu sunt furnizate date noi. Odată ce atributele sunt actualizate, se efectuează un commit în sesiunea bazei de date pentru a persista schimbările. Dacă actualizarea este realizată cu succes, funcția afișează în consolă înregistrarea actualizată și trimite înapoi la client un răspuns JSON pozitiv, care confirmă actualizarea și include un mesaj corespunzător. În cazul unei excepții neașteptate, funcția returnează un răspuns cu un mesaj de eroare și un cod de stare HTTP 500, indicând o eroare internă a serverului.

### 5.3.3. Operația DELETE

Funcția delete\_discount(discount\_id) este un exemplu de operație Delete din setul de operații CRUD într-o aplicație web bazată pe Flask și SQLAlchemy. Această funcție are rolul de a șterge o înregistrare specifică din baza de date, identificată prin discount\_id. Procesul de ștergere este mai complex și implică mai multe etape: În primul rând, funcția caută și șterge toate instanțele asociate cu discount\_id din tabela DiscountInstance, care pare să fie o tabelă de legătură între reduceri și alte entități din baza de date. Apoi, obiectul Discount corespunzător este identificat folosind metoda get\_or\_404, care va returna o eroare 404 dacă nu există nicio înregistrare cu ID-ul specificat. Acest lucru asigură că procesul de ștergere continuă numai dacă înregistrarea este de fapt prezentă în baza de date. În funcție de tipul reducerii, identificat prin discount.discount\_type\_id, funcția execută logica specifică pentru ștergerea detaliilor reducerii din tabelele corespunzătoare. Acest lucru poate include ștergerea reducerilor procentuale, reducerilor cu valoare fixă, reducerilor complementare (și a produselor asociate acestora), sau reducerilor cantitative. După ce toate înregistrările asociate au fost șterse, înregistrarea principală din tabela Discount este eliminată prin db.session.delete(discount). Schimbările sunt apoi confirmate în baza de date prin db.session.commit(), finalizând astfel procesul de ștergere.

Dacă operațiunea este finalizată cu succes, se returnează un răspuns pozitiv către client, indicând că reducerea a fost eliminată. Dacă apare o excepție în timpul acestui proces, funcția returnează un răspuns de eroare cu un mesaj adecvat și un cod de stare HTTP 500, care indică o eroare internă a serverului.

### 5.3.4. Operația CREATE

Funcția add\_discount() exemplifică operația Create din paradigma CRUD într-o aplicație web care utilizează Flask și SQLAlchemy pentru gestionarea datelor. Funcția procesează date JSON primite printr-o cerere HTTP pentru a crea o nouă reducere în sistem. La începutul funcției, se preiau datele necesare din corpul cererii JSON. Un nou obiect Discount este inițializat cu aceste date, inclusiv descrierea reducerii, tipul reducerii, și datele de început și sfârșit. Acest nou obiect Discount este adăugat în sesiunea bazei de date și este urmat de un flush pentru a obține ID-ul atribuit după inserare. Următoarea secțiune de cod se ocupă de asocierea reducerii cu magazinele specifice, iterând prin ID-urile magazinelor furnizate și creând instanțe de DiscountInstance pentru fiecare, care sunt apoi adăugate în sesiunea bazei de date. În funcție de tipul reducerii, diferite blocuri de cod sunt executate pentru a gestiona logica specifică fiecărui tip de reducere. De exemplu, pentru reducerile procentuale, se caută produsul corespunzător, se calculează prețul final cu reducerea aplicată și se creează o instanță de PercentageDiscount. Procese similare sunt efectuate pentru reducerile cu valoare fixă, reducerile complementare (inclusiv crearea legăturilor cu produsele relevante) și reducerile cantitative. După configurarea datelor reducerii, toate modificările sunt persistate în baza de date prin db.session.commit(). În caz de succes, se returnează un răspuns JSON afirmativ. Dacă apare o excepție pe parcursul procesului, se returnează un răspuns cu detalii despre eroare și un cod de stare HTTP 500, semnalând o eroare de server internă.

# 6. Listing-ul Aplicației WEB

## 6.1. Exemplu de model definit pentru tabelul product

from models.product\_category import ProductCategory

from models.manufacturer import Manufacturer

class Product(db.Model):

\_\_tablename\_\_ = 'product'

id = db.Column(db.Integer, primary\_key=True, autoincrement=True)

name = db.Column(db.String(50), nullable=False)

volume = db.Column(db.Numeric(10, 2))

manufacturer\_id = db.Column(db.Integer, db.ForeignKey('manufacturer.id'), nullable=False)

unique\_code = db.Column(db.String(255), nullable=False, unique=True)

price = db.Column(db.Numeric(10, 2), nullable=False)

category\_id = db.Column(db.Integer, db.ForeignKey('product\_category.id'), nullable=False)

manufacturer = db.relationship('Manufacturer')

category = db.relationship('ProductCategory')

def \_\_repr\_\_(self):

return f'<Product {self.name}>'

## 6.2. Exemplu de funcție pentru formatare a datelor în JSON

def format\_product\_data(products):

"""

Transforms product data into a list of dictionaries suitable for JSON response.

Args:

products: A list of tuples containing Product, Manufacturer, and ProductCategory objects.

Returns:

A list of dictionaries with product details.

"""

return [{

'product\_id': product.Product.id,

'product\_name': product.Product.name,

'unique\_code': product.Product.unique\_code,

'product\_volume': str(product.Product.volume),

'product\_price': str(product.Product.price),

'manufacturer\_name': product.Manufacturer.name,

'category\_name': product.ProductCategory.name

} for product in products]

## 6.3. Definirea rutelor

from flask import Blueprint

# Importing controllers

from controllers.category import get\_categories

from controllers.manufacturer import get\_manufacturers

from controllers.store import get\_stores\_in\_network\_one

from controllers.product import get\_products, add\_product

from controllers.discount import get\_discounts, get\_discounts\_by\_params, update\_discount, delete\_discount, get\_discount

from controllers.admin\_discount import get\_expired\_discounts, get\_actual\_discounts, get\_future\_discounts, get\_discount\_counts, add\_discount

# Creating a Blueprint for the main routes

main\_blueprint = Blueprint('main', \_\_name\_\_)

main\_blueprint.route('/category', methods=['GET'])(get\_categories)

main\_blueprint.route('/manufacturer', methods=['GET'])(get\_manufacturers)

main\_blueprint.route('/store', methods=['GET'])(get\_stores\_in\_network\_one)

main\_blueprint.route('/product', methods=['GET'])(get\_products)

main\_blueprint.route('/product', methods=['POST'])(add\_product)

main\_blueprint.route('/discount', methods=['GET'])(get\_discounts)

main\_blueprint.route('/discount', methods=['POST'])(get\_discounts\_by\_params)

main\_blueprint.route('/discount/<int:discount\_id>', methods=['GET'])(get\_discount)

main\_blueprint.route('/discount/<int:discount\_id>', methods=['PUT'])(update\_discount)

main\_blueprint.route('/discount/<int:discount\_id>', methods=['DELETE'])(delete\_discount)

main\_blueprint.route('/admin-discount', methods=['GET'])(get\_discount\_counts)

main\_blueprint.route('/admin-discount', methods=['POST'])(add\_discount)

main\_blueprint.route('/admin-discount/expired', methods=['GET'])(get\_expired\_discounts)

main\_blueprint.route('/admin-discount/actual', methods=['GET'])(get\_actual\_discounts)

main\_blueprint.route('/admin-discount/future', methods=['GET'])(get\_future\_discounts)

## 6.4. Exemplu de interogare a bazei de date

def fetch\_discount\_by\_id(discount\_id):

"""

Fetches a discount by its ID.

Args:

discount\_id: The ID of the discount to fetch.

Returns:

The Discount object if found, otherwise None.

"""

return Discount.query.get(discount\_id)

## 6.5. Exemplu de controller pentru obținerea reducerilor

def get\_discounts():

try:

discounts = fetch\_discounts()

discounts\_list = []

for discount in discounts:

if discount.discount\_type\_id == 1:

discount\_data = format\_discount\_data(discount, fetch\_percentage\_discount\_details)

elif discount.discount\_type\_id == 2:

discount\_data = format\_discount\_data(discount, fetch\_fixed\_value\_discount\_details)

elif discount.discount\_type\_id == 3:

discount\_data = format\_discount\_data(discount, fetch\_complementary\_discount\_details)

elif discount.discount\_type\_id == 4:

discount\_data = format\_discount\_data(discount, fetch\_quantity\_discount\_details)

discount\_instances = fetch\_discount\_instances(discount.id)

stores = [format\_store\_info(instance.store) for instance in discount\_instances]

discount\_data['stores'] = stores

discounts\_list.append(discount\_data)

return jsonify(discounts\_list)

except Exception as e:

return jsonify({"error": str(e)}), 500

## 6.6. Exemplu de fetching al datelor pe partea de frontend

fetch('http://localhost:5555/category')

.then(response => {

if (!response.ok) {

throw new Error('Network response was not ok');

}

return response.json();

})

.then(data => {

const selectElement = document.getElementById('category-select');

data.forEach(category => {

const option = document.createElement('option');

option.value = category.id;

option.textContent = category.name;

selectElement.appendChild(option);

});

})

.catch(error => {

console.error('There has been a problem with your fetch operation:', error);

});

# Concluzii

În cadrul acestei lucrări, s-a dezvoltat o platformă avansată pentru monitorizarea reducerilor, utilizând tehnologii precum Flask, Python, SQLAlchemy, HTML, CSS și JavaScript. Această inițiativă reprezintă un pas important în interacțiunea dintre consumatori și administratorii de magazine. Prin implementarea acestei platforme, s-a demonstrat capacitatea tehnologiilor web de a optimiza procesul de cumpărături, oferind utilizatorilor posibilitatea de a accesa și compara reduceri din diverse magazine într-un mod centralizat și eficient. Platforma a fost concepută cu scopul de a reduce impactul publicității excesive, permițând utilizatorilor să selecteze reduceri în funcție de criterii specifice, facilitând astfel o experiență de cumpărare mai orientată și personalizată. Pentru administratorii de magazine, platforma oferă un instrument valoros în evitarea campaniilor publicitare ineficiente. Utilizarea Flask și SQLAlchemy a oferit o bază de date robustă și flexibilă, esențială pentru gestionarea eficientă a datelor despre reduceri. Acest lucru a permis o scalabilitate și adaptabilitate sporită a platformei. Proiectul a demonstrat, de asemenea, potențialul de integrare a tehnologiilor moderne în domeniul comerțului electronic, subliniind importanța inovației continue în acest sector. Lucrarea de față a evidențiat rolul central al bazelor de date în dezvoltarea aplicațiilor web moderne, cu un accent deosebit pe MySQL, o bază de date relațională robustă și eficientă. Utilizarea MySQL ca fundament pentru stocarea și gestionarea datelor reducerilor a permis o interogare eficientă și o manipulare precisă a datelor, aspecte esențiale pentru performanța platformei. Structura relațională a bazei de date MySQL a facilitat organizarea datelor într-un mod care reflectă natura interconectată a informațiilor despre reduceri, magazine și categorii de produse. Reducerile, ca domeniu de studiu, prezintă un interes substanțial în contextul economic actual, unde consumatorii sunt în căutarea constantă a celor mai bune oferte. Platforma dezvoltată oferă o perspectivă nouă asupra modului în care reducerile sunt monitorizate și utilizate, oferind un cadru pentru studiul comportamentului consumatorilor și al strategiilor comercianților. Acest proiect subliniază importanța bazei de date în analiza și prezentarea reducerilor, demonstrând cum tehnologia informației poate influența pozitiv sectorul de retail.

În concluzie, teza a ilustrat cum integrarea eficientă a unei baze de date relaționale puternice, precum MySQL, cu tehnologii web front-end și back-end poate transforma modul în care reducerile sunt gestionate și exploatate în beneficiul atât al comercianților, cât și al consumatorilor. Rezultatele obținute subliniază rolul crucial al bazelor de date în dezvoltarea soluțiilor tehnologice care abordează nevoile reale ale pieței și ale utilizatorilor finali, în acest caz, în domeniul reducerilor.

# Bibliografie

1. Documentație MySQL [ <https://dev.mysql.com/doc/> ]
2. Documentație Python [ <https://docs.python.org/3/> ]
3. Documentație Flask [ <https://flask.palletsprojects.com/en/3.0.x/> ]
4. Documentație SQLAlchemy [ <https://docs.sqlalchemy.org/en/20/> ]
5. Pagină oficială Linella [ <https://linella.md/ro> ]
6. Pagină oficială Nr.1 [ <https://nr1.md/> ]
7. Reduceri ca domeniu de studiu [ <https://www.omnisend.com/blog/the-advantages-and-disadvantages-of-discounts/> ]
8. Beneficii reduceri [ <https://www.libertatea.ro/publicitate-advertorial/totul-despre-reduceri-de-ce-e-bine-sa-profiti-de-aceasta-oportunitate-4282939> ]
9. Proiectarea unei baze de date [ <https://dbdiagram.io/d/ScheduleBot-6564470a3be1495787c9b4ff> ]
10. Realizare scheme [ <https://app.diagrams.net/> ]
11. Convenții MySQL [ <https://medium.com/@centizennationwide/mysql-naming-conventions-e3a6f6219efe> ]