***Baza de date pentru gestionarea***

***serverului Dummy Chef***

**- proiect –**

**Proiect realizat de:** sd. cap**.** Văduva Marius Octavian

**Grupa de studii:** C 112B

1. Motivația alegerii proiectului

Am ales această temă de proiect întrucât consider că există o nevoie în creștere pentru soluții digitale care să faciliteze accesul la rețete culinare diverse și să simplifice procesul de gătit, atât pentru bucătarii amatori (utilizatorii), cât și pentru profesioniști. Dummy Chef este o platformă centralizată care își propune să ofere utilizatorilor acces la o bază de date extinsă de rețete culinare, să le permită să navigheze și să aleagă rețete în funcție de preferințele și necesitățile lor, să genereze liste de cumpărături personalizate și să faciliteze gestionarea ingredientelor și a furnizorilor.

Funcționalitățile aplicației pot acoperi mai multe necesități din societate:

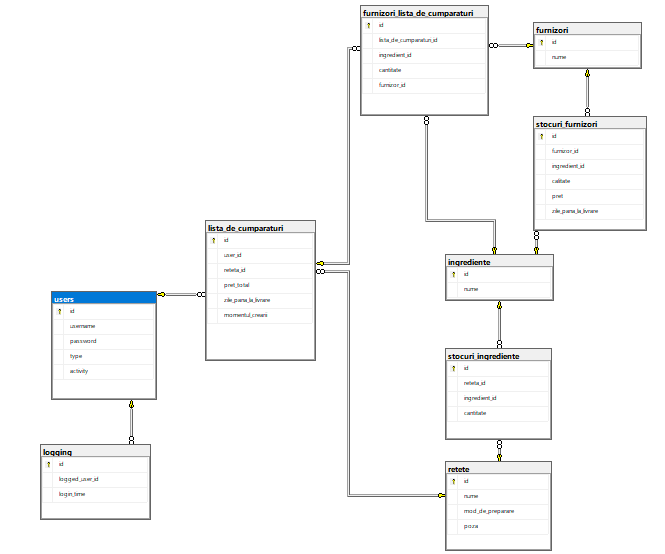
1. **Eficiență în gătit:** O platformă care oferă rețete variate și personalizate poate ajuta oamenii să gătească mai eficient și să încerce noi preparate culinare, înlocuind rutina culinară cu opțiuni mai diverse și creative.
2. **Economie de timp și resurse:** Generarea listelor de cumpărături pe baza preferințelor și bugetului utilizatorilor poate reduce timpul petrecut în magazine și poate ajuta la gestionarea mai eficientă a resurselor.
3. **Accesibilitate:** Prin separarea rolurilor între bucătari și clienți, platforma poate oferi atât experiențe personalizate pentru bucătarii profesioniști, cât și pentru cei amatori, facilitând accesul la informații și resurse specifice fiecărui tip de utilizator.
4. **Transparență în achiziții:** Integrarea informațiilor despre furnizori și stocuri în platformă poate oferi utilizatorilor transparență în privința surselor de alimente și poate ajuta la luarea deciziilor informate în procesul de cumpărare.
5. **Gestionarea eficientă a inventarului:** Funcționalitățile de administrare a ingredientelor și furnizorilor pot fi utile și pentru profesioniștii din industria alimentară, facilitând gestionarea stocurilor și aprovizionarea.

Astfel, baza de date pentru gestionarea serverului Dummy Chef are ca scop gestionarea resurselor acestei aplicații și furnizarea constanta de date catre aplicatie. Industria alimentară este în continuă creștere și evoluție, iar gestionarea eficientă a rețetelor, ingredientelor și furnizorilor este crucială pentru succesul în acest domeniu. O bază de date dedicată poate fi o unealtă valoroasă pentru profesioniștii din industria alimentară.

Potențialul de extindere și personalizare: Tema proiectului oferă oportunitatea de a explora și implementa funcționalități suplimentare sau personalizate în funcție de cerințele specifice ale utilizatorilor și bucătarilor. Această flexibilitate poate fi motivantă pentru a crea o soluție adaptată nevoilor și preferințelor utilizatorilor.

În scop didactic, în cadrul materiei „Programare orientată pe obiecte – proiect” va fi implementată o aplicație care sa utilizeze această bază de date și să permită automatizarea aceteia.

1. Prezentarea și creerea tabelelor



-- Crează tabelul "users"

CREATE TABLE users (

id INT PRIMARY KEY IDENTITY, -- Cheie primară auto-incrementată

username VARCHAR(50) UNIQUE, -- Câmpul pentru username, unic

password VARCHAR(100), -- Câmpul pentru parolă

type VARCHAR(20), -- Câmpul pentru tip (bucătar, utilizator, admin)

activity BIT -- Câmpul pentru activitate (1 pentru activ, 0 pentru inactiv)

);

Acest script va crea tabela **users** în baza de date **DummyChef**, cu următoarele caracteristici:

* Câmpul **username** va fi unic și va fi cheie primară. Este de tip **VARCHAR(50)**, ceea ce înseamnă că poate stoca până la 50 de caractere pentru fiecare username.
* Câmpul **password** va fi de tip **VARCHAR(100)**, ceea ce înseamnă că poate stoca până la 100 de caractere pentru fiecare parolă.
* Câmpul **type** va fi de tip **VARCHAR(20)** și va stoca tipul utilizatorului (bucătar, utilizator, admin).
* Câmpul **activity** va fi de tip **BIT**, care este un tip de date boolean în SQL Server (1 pentru activ, 0 pentru inactiv).

CREATE TABLE logging (

id INT PRIMARY KEY IDENTITY, -- Cheie primară auto-incrementată

logged\_user\_id INT, -- ID-ul utilizatorului care s-a logat

login\_time DATETIME, -- Momentul de timp la care s-a realizat logarea

FOREIGN KEY (logged\_user\_id) REFERENCES users(id) -- Cheie străină către tabela users

);

CREATE TABLE ingrediente (

id INT PRIMARY KEY IDENTITY, -- Cheie primară auto-incrementată

nume VARCHAR(100) UNIQUE -- Câmpul pentru nume, unic

);

Acest script va crea tabela ingrediente în baza de date DummyChef, cu următoarele caracteristici:

Câmpul id va fi cheie primară și va fi auto-incrementat. Acesta va servi ca identificator unic pentru fiecare înregistrare din tabel.

Câmpul nume va stoca numele ingredientului și va fi unic (UNIQUE). Este de tip VARCHAR(100), ceea ce înseamnă că poate stoca până la 100 de caractere pentru fiecare nume de ingredient.

CREATE TABLE furnizori (

id INT PRIMARY KEY IDENTITY, -- Cheie primară auto-incrementată

nume VARCHAR(100) UNIQUE -- Câmpul pentru nume, unic

);

CREATE TABLE stocuri\_furnizori (

id INT PRIMARY KEY IDENTITY, -- Cheie primară auto-incrementată

furnizor\_id INT, -- Cheie străină către tabelul furnizori

ingredient\_id INT, -- Cheie străină către tabelul ingrediente

cantitate DECIMAL(10,2), -- Cantitatea de ingredient în stoc

calitate VARCHAR(50), -- Calitatea ingredientului

pret DECIMAL(10,2), -- Prețul ingredientului

zile\_pana\_la\_livrare INT, -- Numărul de zile până la livrare

FOREIGN KEY (furnizor\_id) REFERENCES furnizori(id), -- Cheie străină către tabelul furnizori

FOREIGN KEY (ingredient\_id) REFERENCES ingrediente(id) -- Cheie străină către tabelul ingrediente

);

* **id**: Cheia primară a tabelului **stocuri\_furnizori**, care este auto-incrementată și servește drept identificator unic pentru fiecare înregistrare din tabel.
* **furnizor\_id**: Cheie străină către tabela **furnizori**, care indică furnizorul asociat cu acest stoc.
* **ingredient\_id**: Cheie străină către tabela **ingrediente**, care indică ingredientul asociat cu acest stoc.
* **cantitate**: Cantitatea de ingredient în stoc, definită ca un număr cu două zecimale.
* **calitate**: Descrierea calității ingredientului, de exemplu, "proaspăt", "congelat", etc.
* **pret**: Prețul ingredientului, definit ca un număr cu două zecimale.
* **zile\_pana\_la\_livrare**: Numărul de zile până la livrare pentru acest stoc de ingredient.

Această structură de tabel respectă relațiile dintre **stocuri\_furnizori**, **furnizori** și **ingrediente**, folosind chei străine către tabelele corecte.

ALTER TABLE stocuri\_furnizori

DROP COLUMN cantitate; -- eliminam cantitate din tabelul stocuri\_furnizori

CREATE TABLE retete (

id INT PRIMARY KEY IDENTITY, -- Cheie primară auto-incrementată

nume VARCHAR(100) UNIQUE, -- Numele retetei, unic

mod\_de\_preparare TEXT, -- Modul de preparare al retetei (text lung)

poza VARBINARY(MAX) -- Imaginea retetei (binar)

);

* **id**: Cheia primară a tabelului "retete", care este auto-incrementată și servește drept identificator unic pentru fiecare înregistrare din tabel.
* **nume**: Numele retetei, definit ca un șir de caractere de maxim 100 de caractere și unic, pentru a asigura că fiecare rețetă are un nume distinct.
* **mod\_de\_preparare**: Descrierea modului de preparare al retetei, definită ca un câmp de tip TEXT, care permite stocarea unor cantități mai mari de text.
* **poza**: Imaginea retetei, definită ca un câmp de tip VARBINARY(MAX), care poate stoca imagini de dimensiuni variabile.

CREATE TABLE stocuri\_ingrediente (

id INT PRIMARY KEY IDENTITY, -- Cheie primară auto-incrementată

reteta\_id INT, -- Cheie străină către retete

ingredient\_id INT, -- Cheie străină către ingrediente

cantitate DECIMAL(10,2), -- Cantitatea de ingredient necesară

FOREIGN KEY (reteta\_id) REFERENCES retete(id), -- Cheie străină către tabela retete

FOREIGN KEY (ingredient\_id) REFERENCES ingrediente(id) -- Cheie străină către tabela ingrediente

);

* **reteta\_id**: Cheie străină către tabela "retete", care indică reteta asociată cu aceste stocuri de ingrediente.
* **ingredient\_id**: Cheie străină către tabela "ingrediente", care indică ingredientul asociat cu aceste stocuri de ingrediente.
* **cantitate**: Cantitatea de ingredient necesară pentru reteta respectivă, definită ca un număr cu două zecimale.

Acest tabel "stocuri\_ingrediente" va permite să gestionezi relația între retete și ingredientele necesare pentru a le pregăti, păstrând înregistrări ale cantităților de ingrediente necesare pentru fiecare retetă.

CREATE TABLE lista\_de\_cumparaturi (

id INT PRIMARY KEY IDENTITY, -- Cheie primară auto-incrementată

user\_id INT, -- Cheie străină către tabelul users

reteta\_id INT, -- Cheie străină către tabelul retete

pret\_total DECIMAL(10,2), -- Prețul total al ingredientelor pentru reteta respectivă

zile\_pana\_la\_livrare INT, -- Numărul de zile până la livrare

momentul\_crearii DATETIME, -- Momentul de timp la care s-a realizat logarea

FOREIGN KEY (user\_id) REFERENCES users(id), -- Cheie străină către tabela users

FOREIGN KEY (reteta\_id) REFERENCES retete(id) -- Cheie străină către tabela retete

);

* **id**: Cheie primară auto-incrementată pentru identificarea unică a fiecărei înregistrări.
* **user\_id**: Cheie străină care face referire la utilizatorul care a creat lista de cumpărături.
* **reteta\_id**: Cheie străină care face referire la reteta pentru care este creată lista de cumpărături.
* **pret\_total**: Prețul total al ingredientelor necesare pentru reteta respectivă.
* **zile\_pana\_la\_livrare**: Numărul de zile până la livrarea ingredientelor.

Acest tabel va fi util pentru a urmări ce utilizatori au creat liste de cumpărături pentru anumite rețete, cât costă ingredientele pentru fiecare listă și câte zile mai sunt până la livrarea acestora.

CREATE TABLE furnizori\_lista\_de\_cumparaturi (

id INT PRIMARY KEY IDENTITY, -- Cheie primară auto-incrementată

lista\_de\_cumparaturi\_id INT, -- Cheie străină către tabela lista\_de\_cumparaturi

ingredient\_id INT, -- Cheie străină către tabela ingrediente

cantitate DECIMAL(10,2), -- Cantitatea de ingredient necesară

furnizor\_id INT, -- Cheie străină către tabela furnizori

FOREIGN KEY (lista\_de\_cumparaturi\_id) REFERENCES lista\_de\_cumparaturi(id), -- Cheie străină către tabela lista\_de\_cumparaturi

FOREIGN KEY (ingredient\_id) REFERENCES ingrediente(id), -- Cheie străină către tabela ingrediente

FOREIGN KEY (furnizor\_id) REFERENCES furnizori(id) -- Cheie străină către tabela furnizori

);

* **id**: Cheie primară auto-incrementată pentru identificarea unică a fiecărei înregistrări.
* **lista\_de\_cumparaturi\_id**: Cheie străină care face referire la lista de cumpărături pentru care se face achiziția.
* **ingredient\_id**: Cheie străină care face referire la ingredientul achiziționat de la furnizor.
* **furnizor\_id**: Cheie străină care face referire la furnizorul de la care se achiziționează ingredientul.

Acest tabel va ajuta la urmărirea legăturii dintre furnizori și listele de cumpărături, precum și legătura dintre ingredientele achiziționate și furnizori.

1. Insert

-- Inserare în tabelul "users"

INSERT INTO users (username, password, type, activity)

VALUES ('utilizator1', 'parola123', 'utilizator', 1),

('bucatar1', 'parola456', 'bucatar', 1),

('admin1', 'parola789', 'admin', 1);

-- Inserare în tabelul "logging"

INSERT INTO logging (logged\_user\_id, login\_time)

VALUES (1, GETDATE()), -- Presupunând că utilizatorul cu ID-ul 1 s-a logat în momentul curent

(2, '2024-03-19 10:30:00'), -- Un alt exemplu cu un timp specific

(3, '2024-03-19 11:45:00');

-- Inserare în tabelul "ingrediente"

INSERT INTO ingrediente (nume)

VALUES ('faina'),

('oua'),

('zahar');

-- Inserare în tabelul "furnizori"

INSERT INTO furnizori (nume)

VALUES ('Furnizor1'),

('Furnizor2'),

('Furnizor3');

-- Inserare în tabelul "stocuri\_furnizori"

INSERT INTO stocuri\_furnizori (furnizor\_id, ingredient\_id, calitate, pret, zile\_pana\_la\_livrare)

VALUES (1, 1, 'buna', 5.99, 2),

(2, 2, 'foarte buna', 7.50, 3),

(3, 3, 'excelenta', 10.25, 1);

-- Inserare în tabelul "retete"

INSERT INTO retete (nume, mod\_de\_preparare, poza)

VALUES ('Placinta cu mere', 'Se amesteca faina cu ouale si zaharul, apoi se adauga merele taiate cubulete. Se coace la 180°C timp de 40 de minute.', NULL),

('Omleta', 'Se bat ouale, se adauga sare si piper, apoi se prajesc intr-o tigaie incinsa.', NULL),

('Tiramisu', 'Se amesteca mascarpone cu zaharul si galbenusurile, apoi se aranjeaza straturi de piscoturi insiropate cu cafea si crema de mascarpone.', NULL);

-- Inserare în tabelul "stocuri\_ingrediente"

INSERT INTO stocuri\_ingrediente (reteta\_id, ingredient\_id, cantitate)

VALUES (1, 1, 300), -- Placinta cu mere: 300 grame de faina

(1, 2, 4), -- Placinta cu mere: 4 oua

(1, 3, 150), -- Placinta cu mere: 150 grame de zahar

(2, 2, 6), -- Omleta: 6 oua

(3, 2, 12), -- Tiramisu: 12 oua

(3, 1, 200), -- Tiramisu: 200 grame de faina

(3, 3, 250); -- Tiramisu: 250 grame de zahar

-- Inserare în tabelul "lista\_de\_cumparaturi"

INSERT INTO lista\_de\_cumparaturi (user\_id, reteta\_id, pret\_total, zile\_pana\_la\_livrare,momentul\_crearii)

VALUES (1, 2, NULL, NULL,GETDATE()), -- Utilizatorul 1 are în lista de cumpărături reteta 1, prețul total și zilele până la livrare vor fi actualizate automat

(2, 2, NULL, NULL,'2024-03-19 10:30:00'), -- Utilizatorul 2 are în lista de cumpărături reteta 2, prețul total și zilele până la livrare vor fi actualizate automat

(3, 3, NULL, NULL,'2024-03-19 10:35:00'); -- Utilizatorul 3 are în lista de cumpărături reteta 3, prețul total și zilele până la livrare vor fi actualizate automat

-- Inserare în tabelul "furnizori\_lista\_de\_cumparaturi"

-- Presupunând că fiecare retetă necesită un ingredient de la un furnizor

INSERT INTO furnizori\_lista\_de\_cumparaturi (lista\_de\_cumparaturi\_id, ingredient\_id, cantitate, furnizor\_id)

VALUES (1, 1, 500, 1), -- Lista de cumpărături 1 necesită 500 grame de făină de la furnizorul 1

(2, 2, 8, 2), -- Lista de cumpărături 2 necesită 8 ouă de la furnizorul 2

(3, 3, 300, 3); -- Lista de cumpărături 3 necesită 300 grame de zahăr de la furnizorul 3

!!!!!!!!

UPDATE lista\_de\_cumparaturi

SET pret\_total = (

SELECT SUM(ROUND(sf.pret \* flc.cantitate, 0))

FROM furnizori\_lista\_de\_cumparaturi flc

INNER JOIN stocuri\_furnizori sf ON flc.furnizor\_id = sf.furnizor\_id AND flc.ingredient\_id = sf.ingredient\_id

WHERE flc.lista\_de\_cumparaturi\_id = lista\_de\_cumparaturi.id

)

WHERE id IN (

SELECT lista\_de\_cumparaturi\_id

FROM furnizori\_lista\_de\_cumparaturi

WHERE reteta\_id = lista\_de\_cumparaturi.reteta\_id

);

Această declarație **UPDATE** va actualiza toate înregistrările din **lista\_de\_cumparaturi**, calculând prețul total pentru fiecare listă de cumpărături. Subinterogarea calculează suma prețurilor din **stocuri\_furnizori** pentru fiecare listă de cumpărături în funcție de rețeta asociată, folosind furnizorii și ingredientele din **furnizori\_lista\_de\_cumparaturi**. Apoi, aceste valori calculate sunt actualizate în câmpul **pret\_total** al fiecărei liste de cumpărături. Suma este calculată înmulțind prețul ingredientului cu cantitatea necesară și rotunjind rezultatul la integer

UPDATE lista\_de\_cumparaturi

SET zile\_pana\_la\_livrare = (

SELECT MAX(sf.zile\_pana\_la\_livrare)

FROM furnizori\_lista\_de\_cumparaturi flc

INNER JOIN stocuri\_furnizori sf ON flc.furnizor\_id = sf.furnizor\_id AND flc.ingredient\_id = sf.ingredient\_id

WHERE flc.lista\_de\_cumparaturi\_id = lista\_de\_cumparaturi.id

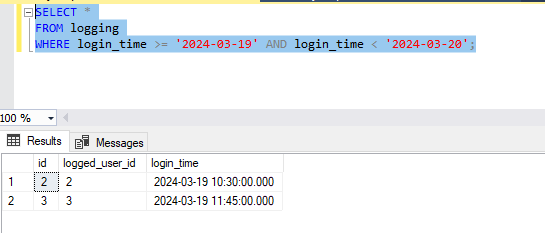
)

Această declarație **UPDATE** va actualiza câmpul **zile\_pana\_la\_livrare** în **lista\_de\_cumparaturi** cu timpul maxim de livrare găsit în **stocuri\_furnizori** pentru fiecare listă de cumpărături. Subinterogarea caută furnizorii și ingredientele corespunzătoare fiecărei liste de cumpărături din **furnizori\_lista\_de\_cumparaturi**, apoi găsește timpul maxim de livrare pentru acești furnizori din **stocuri\_furnizori**. Acest timp maxim este apoi actualizat în câmpul **zile\_pana\_la\_livrare** al fiecărei liste de cumpărături.

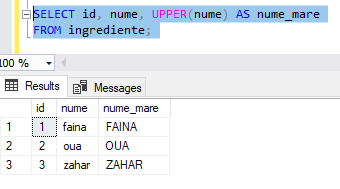
!!!!!!!!

1. Select

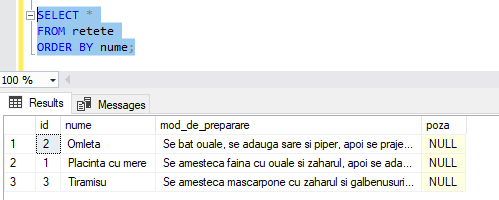
* **Select cu filtrare de date calendaristice:**



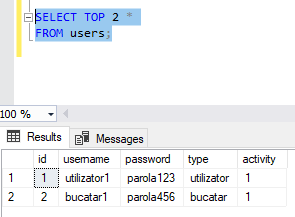
* **Select cu prelucrare de date de tip caracter:**



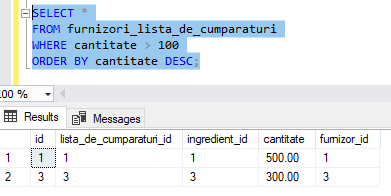
* **Select folosind clauza Order By:**



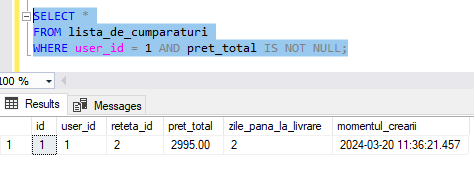
* **Select cu filtrarea datelor folosind TOP:**



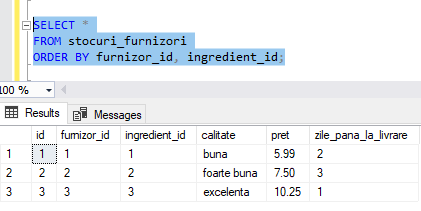
* **Select cu filtrarea și ordonarea datelor:**



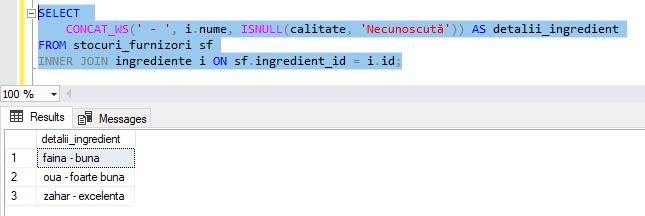
* **Select cu filtrare de date în care se combină cel puțin 2 predicate:**



* **Select cu ordonare pe mai multe coloane:**



* **Select cu concatenare de coloane în care este permis NULL, cu obținerea unui rezultat relevant:**



* **Select cu eliminarea duplicatelor rămase după aplicarea a două filtre. Afișați datele în mod convenabil:**

