

UNIVERSITATEA DIN BUCUREȘTI
FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

LUCRARE DE LICENȚĂ

COORDONATOR ȘTIINȚIFIC

Lect. Dr. Florentin Ipate

STUDENT

Andra-Luana Eftene

BUCUREȘTI

IUNIE 2019

UNIVERSITATEA DIN BUCUREȘTI
FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

LUCRARE DE LICENȚĂ
Aplicație Web pentru o firmă de retail

COORDONATOR ȘTIINȚIFIC

Lect. Dr. Florentin Ipate

STUDENT

Andra-Luana Eftene

BUCUREȘTI

IUNIE 2019

1. Introducere	3
1.1. Motivație	3
1.2. Descrierea și scopul aplicației	3
1.3. Aria de acoperire a aplicației și grupuri de interese	5
2. Preluarea datelor	8
2.1. Necesitate	8
2.2. Implementare	8
3. Funcționalitățile aplicației	12
3.1. Modulul client	12
3.2. Modulul administrator	15
3.3. Plasarea unei comenzi	17
4. Implementarea funcționalităților	19
4.1. Modelarea bazei de date	19
4.2. Algoritmi	21
5. Design	27
5.1. UX	27
5.2. UI	31
6. Search Engine Optimization	33
7. Securitate	35
8. Tehnologii	37
9. Concluzii	42
10. Anexe	43
11. Bibliografie și referințe	44

1. Introducere

1.1. Motivație

Am ales această temă pentru că este un exemplu practic și concret de aplicare a cunoștințelor învățate în cei 3 ani de facultate. Dezvoltarea aplicației prezentate în continuare m-a ajutat să parcurg toate etapele procesului de implementare. Am început cu discuția cu clientul aplicației, apoi am continuat cu stabilirea arhitecturii aplicației, găsirea tehnologiilor potrivite, schițarea și concretizarea design-ului aplicației, modelarea bazei de date. Am trecut apoi la implementarea funcționalităților, design-ului, și tehnicilor SEO.

Deși nu prezintă o inovație în domeniu, aplicația prezentată va ajuta cel puțin o firmă să-și crească veniturile, dar m-a ajutat și pe mine să folosesc multe metode învățate la cursurile facultății și să transform concepte și noțiuni teoretice într-un produs final.

Voi descrie, în continuare, aplicația și scopul acesteia, după care voi explica partea de migrare a datelor și voi prezenta funcționalitățile. În cele din urmă, voi vorbi despre procesele tehnologice și etapele dezvoltării aplicației, voi ilustra diagramele folosite la arhitectura aplicației și voi prezenta tehnologiile folosite.

1.2. Descrierea și scopul aplicației

Scopul aplicației

Obiectul acestei lucrări este o aplicație dezvoltată prin intermediul ASP .NET MVC și este, în sine, un magazin online cu o interfață specială pentru administratorul magazinului, astfel încât acesta să aibă acces la informații relevante pentru propria afacere: care sunt cele mai vândute produse, care sunt cei mai activi clienți etc. Deși competiția este foarte mare întrucât există o mulțime de astfel de platforme deja existente, cred că această aplicație își va face loc printre aplicațiile concurente având drept potențiali clienți firmele de mici dimensiuni - firme pentru care aderarea la aceste platforme nu ar mai fi la fel de profitabilă.

În fapt, aplicația are deja un potențial client, și anume o firmă de retail din București care oferă o gamă largă de echipamente pentru abatoare și industria alimentară. Firma menționată are un site de prezentare construit pe o versiune învechită a platformei Drupal, fără a oferi persoanelor care vizitează site-ul posibilitatea de a plasa o comandă online sau măcar de a afla prețul produselor.

Aplicația prezentată în această lucrare vine ca soluție online pentru firma menționată mai sus: o apropiere față de clienți, o eficientizare a procesului de vânzare, o promisiune de creștere a profitului și o interfață pentru administrator care nu necesită cunoștințe ale tehnicilor sau limbajelor de programare. Totodată, ea va putea fi ușor adaptată pentru alte firme cu aceleași nevoi prin modificări de stilizare ale design-ului site-ului, astfel încât stilul să rezoneze cu partea deja existentă de *branding* a firmei respective.

Descriere

Garanția acestei soluții online este făcută având în vedere atât funcționalități de bază și un design simplu, cât și metode actuale de promovare a site-ului, și anume metode de optimizare pentru motoarele de căutare (SEO).

Funcționalitățile de bază se referă la posibilitatea unui client de a vizualiza produsele împărțite pe categorii, dar și de a vizualiza detalii ale produselor, de a căuta produse în funcție de denumirea acestora, de a plasa comenzi, de a lăsa comentarii sau recenzii produselor cumpărate, de a primi factura pe mail sau chiar de a vizualiza istoricul comenzilor. În același timp, administratorul site-ului poate adăuga, modifica sau șterge produse, poate update stocul, prețul inițial, prețul promoțional. Acesta mai poate vedea comenzile făcute pe site și le poate onora, amâna sau anula. Nu în ultimul rând, administratorul poate promova utilizatori la rangul de administratori ai site-ului, în cazul în care dorește ca mai mulți angajați să se ocupe de traficul pe site.

Funcționalitățile de bază sunt însoțite de un design orientat pe Experiența Utilizatorului (denumit în continuare UX - *User Experience Design*) astfel încât navigarea pe site să fie facilă și să denote încredere în firma pe care aplicația o reprezintă. Am ținut cont în special de accesibilitatea informației și de navigarea instinctuală a utilizatorului: produsele să fie ușor și rapid de vizualizat din prima pagină, iar butoanele și așezarea în pagină să anticipeze funcționalitățile.

Pe plan personal, pot spune că implementarea aplicației mi-a fost un bun prilej de a aplica noțiuni predate la facultate din materii precum: Metode de dezvoltare software, Programare Orientată pe Obiecte, Tehnici Web, Baze de date, Ingineria Programării, Dezvoltarea Aplicațiilor Web, dar și de a învăța lucruri noi pe parcursul dezvoltării aplicației. Mi-a fost de mare ajutor și experiența acumulată pe parcursul orelor de lucru la un stagiul de practică în UX Design și, ulterior, a job-ului de Designer și Front-End Developer.

1.3. Aria de acoperire a aplicației și grupuri de interese

Firma la care m-am referit până acum a fost înființată în anul 2009. În cei 10 ani trecuți, aceasta și-a creat un număr semnificativ de clienți care cumpără produse fie periodic, fie o singură dată. Clienții au ajuns să cunoască firma încă de la începuturile existenței acesteia cu ajutorul site-ului de prezentare.

Procesul de vânzare - cumpărare se face, în prezent, mai ales prin intermediul apelurilor telefonice. Pentru o singură comandă pot avea loc și 4 convorbiri telefonice (plasarea comenzii, verificarea stocului, anunțarea costurilor finale, anunțarea firmei de curierat responsabilă), pierzându-se astfel foarte mult timp. Câteodată este necesară și conversația pe mail, în cazul în care comenzile sunt prea mari și nu pot fi reținute sau notate pe loc. Datele despre clienți nu sunt mereu stocate într-o bază de date (impropriu spus, ea fiind de fapt o fișă de calcul Excel), așa că în cazul în care un astfel de client face o a doua comandă, angajații firmei consumă și mai mult timp căutând facturile anterioare pentru aflarea datelor necesare.

Odată cu avansarea tehnologiei și creșterea popularității comerțului online, firma la care mă refer a început să fie prezentă și pe platforme dedicate pentru a putea ține pasul cu firmele concurente. Cu toate acestea, numărul produselor este prea mare, iar angajații au alte priorități și prea puțin timp la dispoziție, motiv pentru care înregistrarea tuturor produselor pe platformele de comerț online ar fi fost inefficientă.

Aplicația prezentată în această lucrare rezolvă problemele prezentate, iar grupul țintă este format din clienții deja existenți în mediul offline, online, dar și din clienți noi. Un client care va folosi aplicația este o persoană cu vârsta între 30 și 60 de ani, aflat în poziții de conducere sau responsabil cu logistica unor afaceri locale din industria alimentară, precum: abatoare, măcelării, restaurante, producători locali. Aria de acoperire este teritoriul României, întrucât livrările se fac doar la nivel național. Din acest motiv,

limba folosită pe site este limba română, și nu cea engleză. De asemenea, figura de mai jos dovedește că limba română este cea mai bună variantă pentru a atrage potențialii cumpărători.

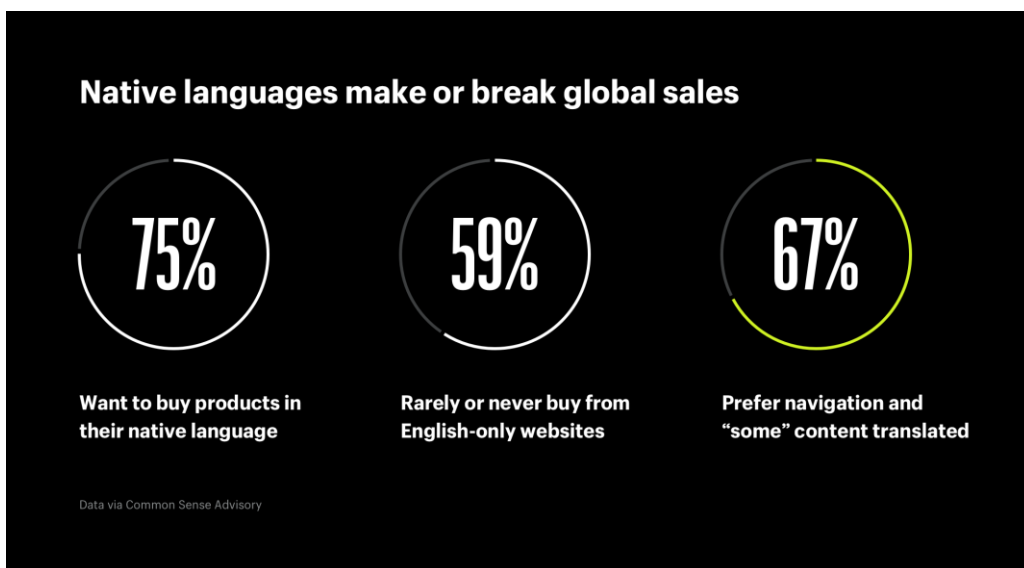


Fig.1 - Statistici despre utilizarea limbii native pe site-uri pentru comerț online

Clienții aplicației pot fi persoane juridice, pentru cazul descris mai sus, dar și persoane fizice, în cazul în care aceștia au nevoie de diferite produse pentru propriul uz casnic.

Figura de mai jos ilustrează creșterea vânzărilor în mediul online comparativ cu vânzările offline în perioada 2003-2013. Deși studiul a fost realizat pentru Statele Unite ale Americii, putem generaliza și aprecia că și în România, lucrurile sunt asemănătoare în ultimii ani.

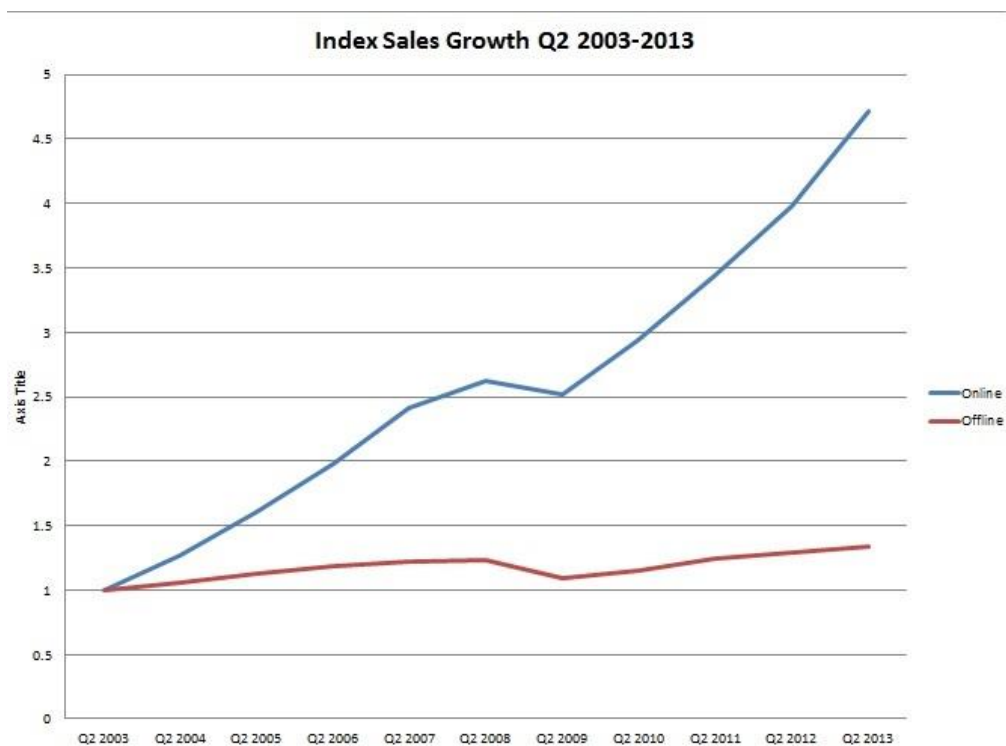


Fig. 2 - Creșterea comerțului online vs comerțului offline

2. Preluarea datelor

2.1. Necesitate

Site-ul actual al firmei are aproximativ 500 de produse stocate, fiecare având un titlu, o descriere și o poză asociată. Salvarea manuală a datelor ar fi consumat foarte mult timp, la fel ca înscrierea produselor pe alte platforme de comerț online. Soluția găsită este prezentată în continuare.

2.2. Implementare

Platforma care găzduiește site-ul actual (*Drupal*) permite accesarea și organizarea conținutului prin intermediul unui cont special de administrator. Accesarea conținutului se poate face creând vizualizări personalizate din baza de date sub formă de tabele cu proprietățile selectate. Figurile de mai jos ilustrează crearea vizualizării și tabelul rezultat.

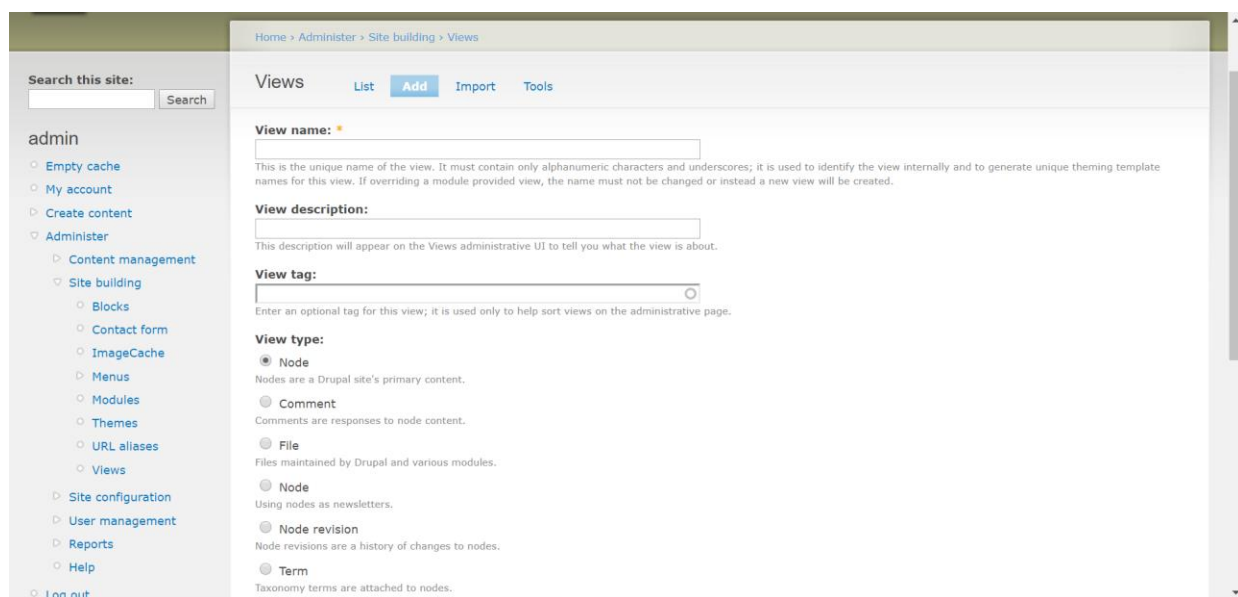


Fig. 3 - Crearea unui view pe platforma Drupal

Am selectat toate “nodurile” care reprezintă produse sau caracteristici ale produsului, precum titlu, descrierea, link-ul către poza reprezentativă și categoria, respectiv gama din care face parte.

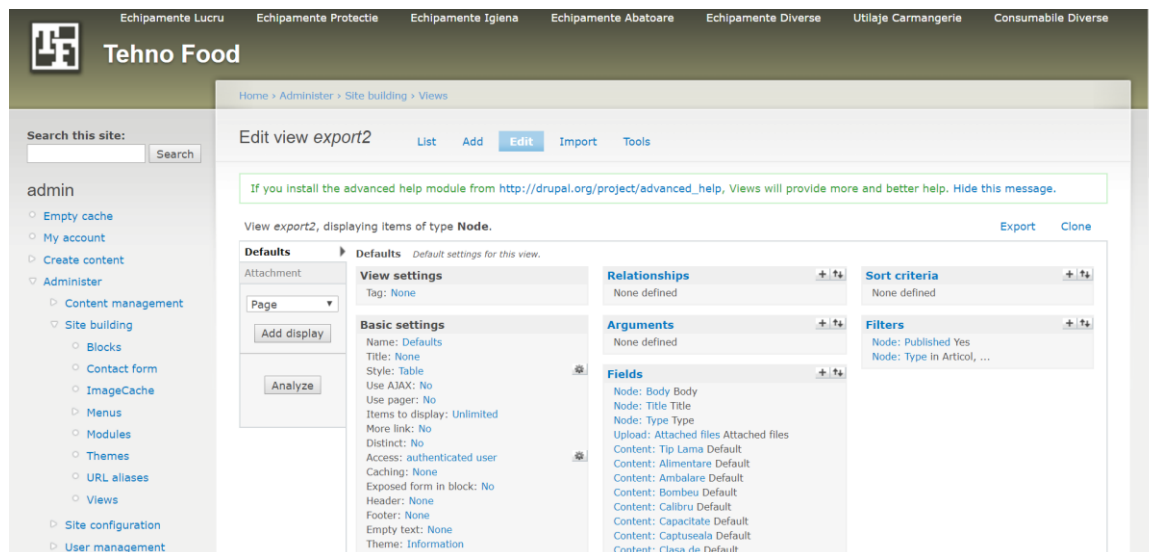


Fig. 4 - View pe platforma Drupal

Am ales să afișez toate elementele care au fost publicate și au toate câmpurile selectate, iar rezultatul a fost afișarea unui tabel cu toate datele într-o pagină HTML nouă.

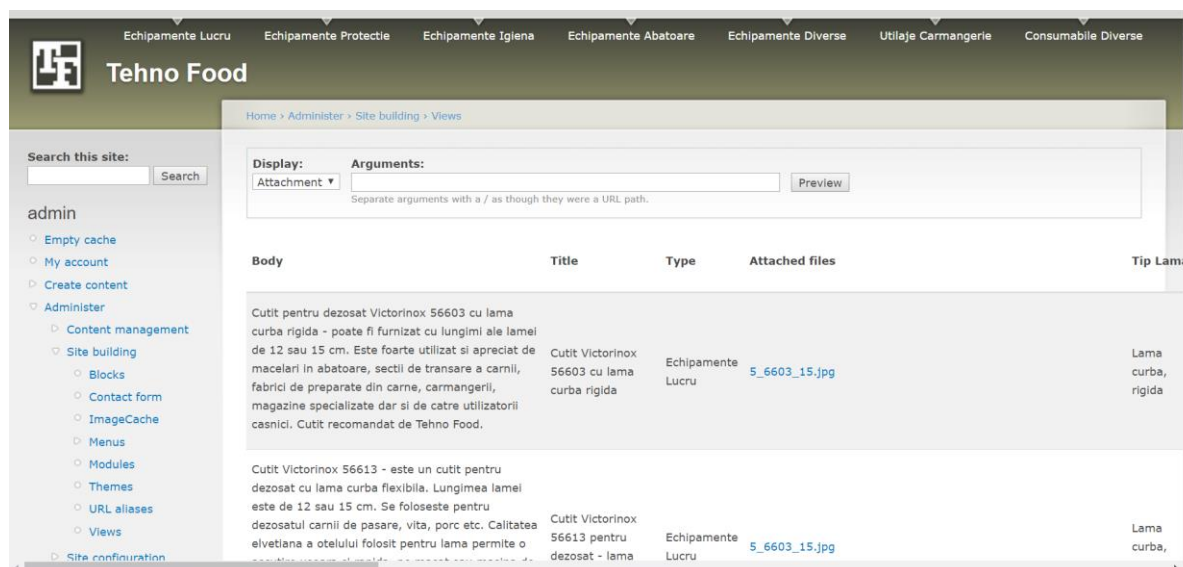


Fig. 5 - Rezultatul view-ului

Pentru a putea adăuga informațiile obținute într-o bază de date nouă, am salvat pagina HTML și am folosit *tool-ul* online de la adresa: <http://www.convertcsv.com/html-table-to-csv.htm> care transformă un tabel HTML (definit prin *tag-uri* specifice: *table*, *th*, *td*, *tr*) într-un fișier cu extensia .xlsx.

Fișierul astfel obținut conține pe prima coloană numele categoriei principale, pe a doua coloană sub-categoria din care face parte produsul, iar pe restul coloanelor informații referitoare la produs. Fiecare rând reprezintă un produs. Figura de mai jos ilustrează parțial informațiile din fișier.

	A	B	C	D	E	F
1	Categorie	Subcategorie	Title	Body	Attached files	Pret
2	Consumabile	Ambalaje	Plasa elastica alimentara pentru preparate din carne	Comercializam o gama variata de plas	https://www.tehnofood.r	50
3	Consumabile	Ambalaje	Folie polietilena PERLAFOLE pentru hamburgeri	Folie polietilena PERLAFOLE pentru har	https://www.tehnofood.r	50
4	Consumabile	Ambalaje	Hartie absorbanta 220x420 mm pentru carne si semi-preparate din carne	Hartie absorbanta 220x420 mm pentr	https://www.tehnofood.r	50
5	Consumabile	Bete frigari	Betisoare din bambus pentru frigari	Betisoare (tepuse) din bambus pentr	https://www.tehnofood.r	50
6	Consumabile	Bete frigari	Bete din lemn pentru masina automata de format frigari	Bete din lemn pentru masina automa	https://www.tehnofood.r	50
7	Consumabile	Bete frigari	Bete din lemn de fag pentru masina automata de format frigari	Putem livra la comanda, pentru cantit	https://www.tehnofood.r	50
8	Consumabile	Bete frigari	Bete din bambus cu un capat ascutit si un capat aplatizat	Betele din bambus cu capat aplatizat	https://www.tehnofood.r	50
9	Consumabile	Diverse	Cutite Dick pentru macelari	Firma Tehno Food distribuie cutite pr	https://www.tehnofood.r	50
10	Consumabile	Diverse	Tub din plastic pentru aplicat plasa elastica alimentara	Tuburi din plastic pentru aplicat plasa	https://www.tehnofood.r	50
11	Consumabile	Diverse	Cutite pentru masini desoricate MAJA - 434x20x1 mm	Cutite pentru masini desoricate MAJA	https://www.tehnofood.r	50
12	Consumabile	Diverse	Spray ulei alimentar pentru lubrifiere	Spray pentru lubrifiere 400 ml - pt. ec	https://www.tehnofood.r	50
13	Consumabile	Diverse	Unsoare alimentara pentru lubrifiere	Unsoare alimentara pentru lubrifiere	https://www.tehnofood.r	50
14	Consumabile	Diverse	Disc din pasla pentru masini de ascutit cutite	Disc din pasla pentru masini de ascuti	https://www.tehnofood.r	50
15	Consumabile	Diverse	Pasta polisare pentru disc pasla - baton 1 kg (alba sau verde)	Pasta polisare pentru disc de pasla - b	https://www.tehnofood.r	50
16	Consumabile	Diverse	Rezistenta 1500w pentru sterilizator cutite cu apa calda	Rezistenta 1500W pentru sterilizator	https://www.tehnofood.r	50
17	Consumabile	Diverse	Rezistenta 1200W pentru sterilizator cutite	Rezistenta 1200W pentru sterilizator	https://www.tehnofood.r	50
18	Consumabile	Diverse	Placa polietilena alimentara - grosime intre 8 - 100 mm	Placa polietilena alimentara - grosime	https://www.tehnofood.r	50
19	Consumabile	Diverse	Banda abraziva pt. masina de ascutit cutite	Banda abraziva pt. masina de ascutit	https://www.tehnofood.r	50
20	Consumabile	Diverse	Sfoara alimentara din bumbac pentru salamuri si specialitati din carne	Sfoara alimentara din bumbac - se uti	https://www.tehnofood.r	50
21	Consumabile	Diverse	Tricot elastic pentru carcase animale abatorizate	Tricotul elastic pentru protectia sanita	https://www.tehnofood.r	50
22	Consumabile	Marcare carne	Creioane marcat carne	Creioanele de marcat Faber Castell 69	https://www.tehnofood.r	50
23	Consumabile	Marcare carne	Coloranti alimentari pentru marcat carne	Firma TEHNO FOOD COM SERV srl cor	https://www.tehnofood.r	50

Fig. 6 - Conținutul fișierului .xlsx

Limbaajul C# și *framework-ul* ASP .NET permit deschiderea fișierelor de tip .xlsx și citirea/scrierea din/în aceste fișiere. Astfel, am creat 3 funcții de importare a datelor, câte una pentru fiecare model relevant din baza de date: categoria principală, sub-categoria și produsul. Fiecare funcție parcurge fișierul pe rânduri și, la fiecare valoare nouă dintr-o celulă stabilită, crează un obiect specific, salvează datele necesare și îl adaugă în baza de date în tabelul potrivit cu cheile externe necesare. Cele 3 funcții se apelează în ordinea dependențelor date de cheile externe. Voi vorbi despre acestea în capitolul *Modelarea bazei de date*.

În figura de mai jos se poate observa funcția de importare a sub-categoriilor (tabel legat prin cheie externă cu categoriile principale). În continuare, categoria principală se numește *SuperCategory*, iar sub-categoria se numește, simplu, *Category*.

```

[Authorize(Roles = "Administrator")]
public ActionResult Import()
{
    //Create COM Objects. Create a COM object for everything that is referenced
    Excel.Application xlApp = new Excel.Application();
    Excel.Workbook xlWorkbook = xlApp.Workbooks.Open(@"D:\ASP .Net\Licenta\OnlineStore\OnlineStore\App_Data\TehnoFood.xlsx");
    Excel.Worksheet xlWorksheet = xlWorkbook.Sheets[1];
    Excel.Range xlRange = xlWorksheet.UsedRange;
    SortedSet<Tuple<string,string>> categories = new SortedSet<Tuple<string, string>>();

    for (int i = 2; i <= xlRange.Rows.Count; i++)
    {
        if (xlRange.Cells[i, 2] != null && xlRange.Cells[i, 2].Value != null && xlRange.Cells[i, 1] != null && xlRange.Cells[i, 1].Value != null)
        {
            var categ = Tuple.Create(xlRange.Cells[i, 1].Value.ToString(), (xlRange.Cells[i, 2].Value.ToString()));
            categories.Add(categ);
        }
    }
    ViewBag.FromExcel = categories;

    foreach (Tuple<string,string> categ in categories)
    {
        Category newCategory = new Category();
        newCategory.Name = categ.Item2;

        var superCategId = from scat in db.SuperCategories
                           where scat.Name == categ.Item1
                           select scat.SuperCategoryId;

        newCategory.SuperCategoryId = superCategId.First();
        db.Categories.Add(newCategory);
        db.SaveChanges();
    }

    //cleanup
    GC.Collect();
    GC.WaitForPendingFinalizers();
    Marshal.ReleaseComObject(xlRange);
    Marshal.ReleaseComObject(xlWorksheet);
    //close and release
    xlWorkbook.Close();
    Marshal.ReleaseComObject(xlWorkbook);
    //quit and release
    xlApp.Quit();
    Marshal.ReleaseComObject(xlApp);
}

```

Fig. 7 - Metoda de importare a informațiilor în baza de date

Am reușit astfel să populez baza de date cu aproximativ 500 de produse într-un timp relativ scurt, raportat la cât ar fi durat ca cineva să introducă manual toate informațiile colectate. În plus, datele sunt astfel mereu stocate în fișierul Excel, putând fi importate din nou după fiecare reinițializare a bazei de date.

3. Funcționalitățile aplicației

3.1. Modulul client

Scopul acestui modul este, în principal, acela de a permite utilizatorilor să plaseze comenzi direct în aplicație. În plus, unele funcționalități implementate încurajează traficul pe site pentru creșterea popularității site-ului și un rezultat mai bun al motoarelor de căutare. Clienții pot fi utilizatori înregistrați sau neînregistrați. Anumite funcționalități descrise mai jos obligă utilizatorul să se înregistreze pe site. Parcursul ecranelor propus pentru clienți este definit în diagrama următoare.

Utilizator client

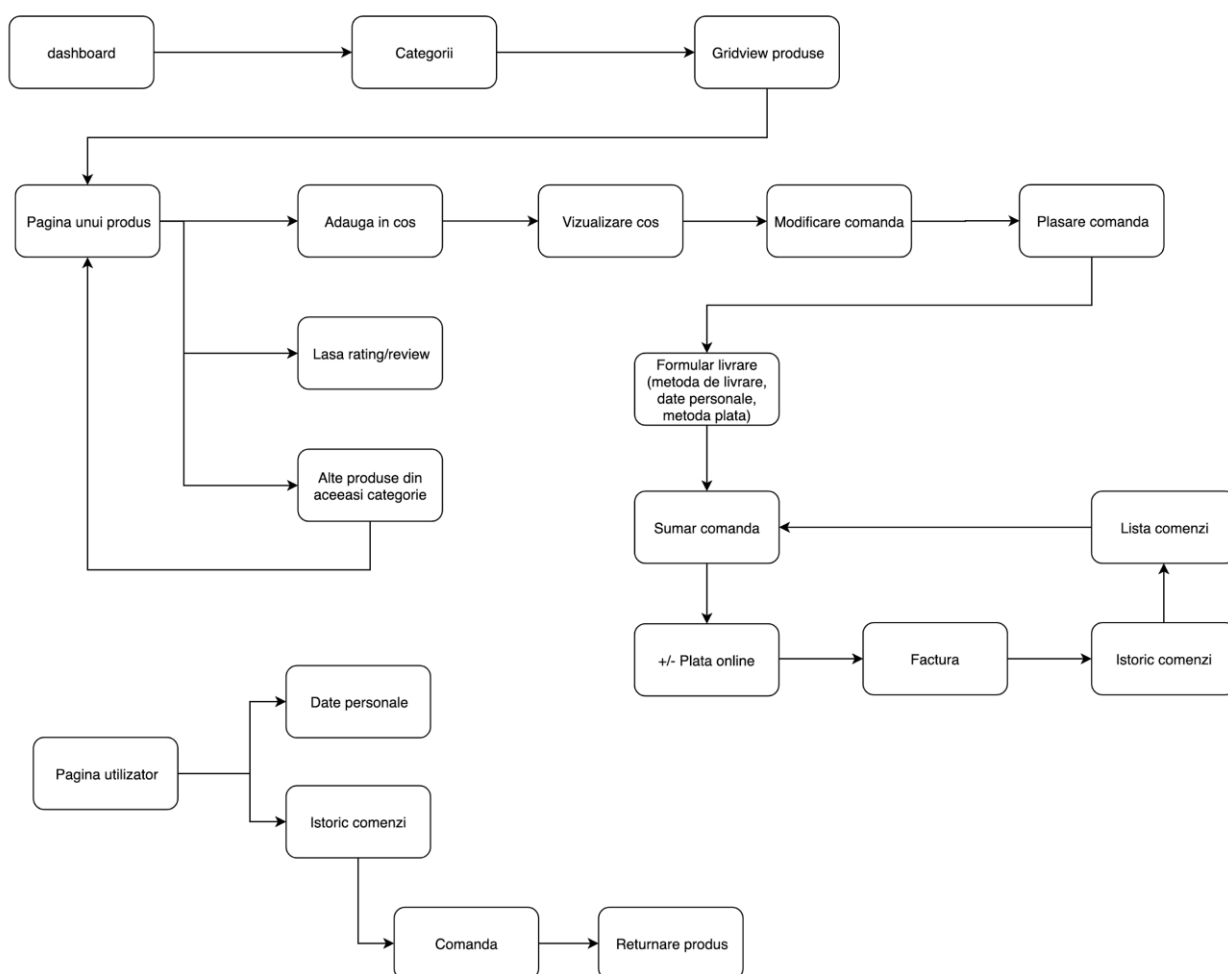


Fig. 8 - User flows - modulul client

Utilizatorul neînregistrat

Începând din pagina principală, utilizatorul neînregistrat poate:

- vizualiza categoriile principale, dar și fiecare sub-categorie a acestora și produsele aparținătoare.
- naviga în site atât prin intermediul unui meniu orizontal, cât și prin intermediul unui meniu vertical ce conține categoriile și subcategoriile definite.
- vizualiza produsele împărțite pe categorii sau chiar pe sub-categorii.
- căuta produse în funcție de titlu sau de categoria din care face parte. Căutarea se poate face prin intermediul unei bare de căutare din meniul orizontal.
- rezultatele căutării pot fi sortate crescător sau descrescător în funcție de preț sau alfabetic, după titlu.
- vizualiza detaliile unui produs: titlu, preț, imagine, descriere și un videoclip care să arate modul de utilizare al produsului. Imaginea poate fi mărită, iar videoclipul poate fi vizualizat pe tot ecranul.
- vizualiza produse similare și produse din aceeași categorie cu un produs selectat.
- crea un cont pe platformă.

Utilizatorul înregistrat

Următoarele funcționalități obligă utilizatorul să se înregistreze pe site, fie prin crearea unui cont nou, fie prin intrarea într-un cont existent folosind o adresă de e-mail și o parolă sigură.

Odată înregistrat, utilizatorul poate, în plus:

- să adauge produse în coșul de cumpărături, dacă acestea sunt disponibile.
- să ceară o ofertă pentru un produs indisponibil (al cărui stoc a fost epuizat), completând un formular care conține deja adresa de e-mail a utilizatorului și titlul produsului.
- să șteargă produse din coșul de cumpărături.
- să plaseze o comandă pentru toate produsele din coșul său de cumpărături, fiind necesare date pentru livrarea comenzii, date de facturare și pentru modalitatea de plată. Facturarea se poate face pentru

persoane fizice, dar și pentru persoane juridice, iar modalitatea de plată poate fi ramburs, prin firma de curierat, sau prin ordin de plată, pentru persoanele juridice.

- să verifice datele comenzii și are opțiunea de a finaliza comanda sau de a o anula, la sfârșitul procesului.
- să acceseze pagina sa de profil.
- să schimbe parola contului său.
- să vizualizeze istoricul comenzilor făcute în site.
- să vizualizeze datele corespunzătoare unei anumite comenzi.
- să returneze produse prin completarea unui formular specific.
- să lase recenzii pentru produse prin note de la 1 la 10 și comentarii.

3.2. Modul administrator

Administratorul poate folosi aplicația ca un utilizator înregistrat, dar are și alte beneficii în plus. Accesul acestuia în aplicație se face prin intermediul unui cont special căruia i-a fost atribuit rolul de Administrator. Următoarele funcționalități sunt disponibile exclusiv acestuia. Diagrama de mai jos arată alternanța ecranelor și acțiunilor pe care administratorul le poate face în plus.

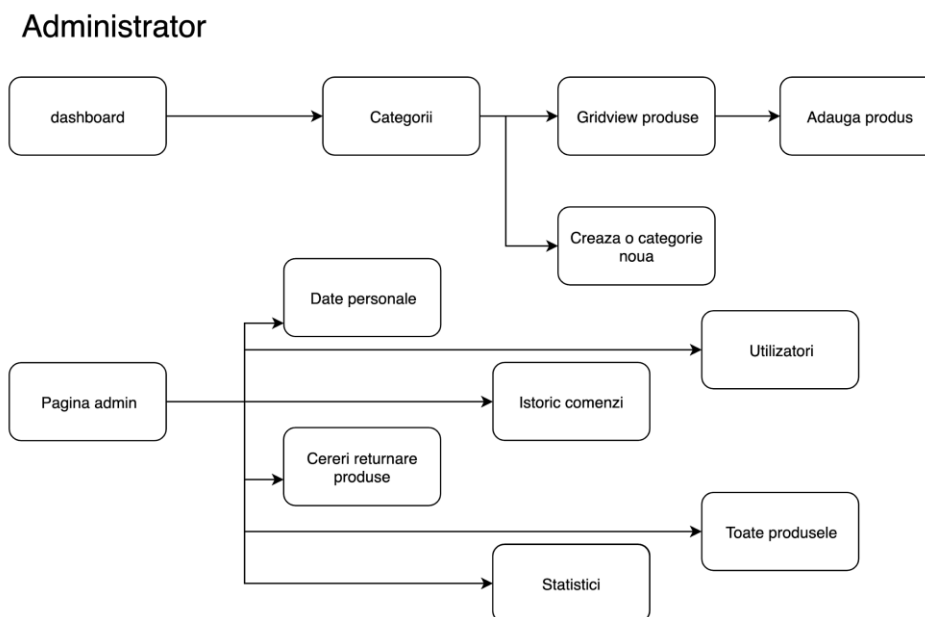


Fig. 9 - User Flows - modul administrator

Administratorul mai poate:

- adăuga produse, categorii, sub-categorii.
- modifica datele produselor, categoriilor, sub-categoriilor.
- șterge produse, categorii, sub-categorii. Ștergerea se face în mod cascadă: odată ștearsă o sub-categorie, sunt șterse și toate produsele aparținente. Analog ștergerea categoriilor principale.
- vizualiza o listă cu toate produsele, de unde poate modifica direct anumite date ale unui produs.
- filtra produsele din lista de mai sus în funcție de titlu.

- ordona produsele din lista de mai sus în ordine alfabetică sau în funcție de preț (crescător sau descrescător).
- vizualiza o listă cu toate comenzile făcute de către clienții înregistrați, de unde poate merge la sumarul unei anumite comenzi sau schimba statusul acesteia.
- căuta o comandă în funcție de numărul de ordine al acesteia.
- ordona comenzile crescător sau descrescător după data sau numărul acestora.
- vizualiza o listă cu toți utilizatorii aplicației.
- promova alți utilizatori la rol de Administrator.
- vizualiza date generale despre comenzi: valoarea comenzilor în așteptare, valoarea comenzilor onorate, cele mai cumpărate produse.
- adăuga și șterge prețuri promoționale pentru un produs. Existența prețului promoțional încadrează produsul în categoria speciala denumită “Promoții”.

3.3. Plasarea unei comenzi

Plasarea unei comenzi necesită preluarea mai multor informații despre utilizator: cantitatea produselor comandate, detalii pentru livrare, date de facturare, modalitate de plată. Pentru a ușura acest proces, am împărțit cererea datelor și formularele corespunzătoare în mai multe pagini. Alternanța acestora poate fi urmărită în diagrama următoare.

Plasarea unei comenzi

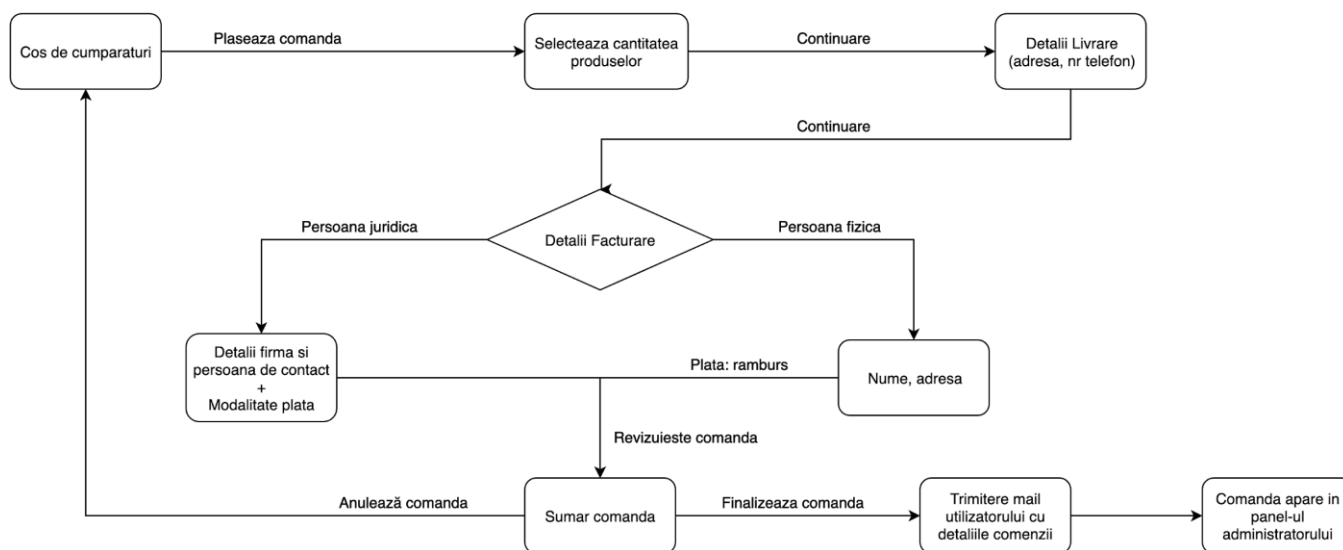


Fig. 10 - User flows - Plasarea unei comenzi

După ce utilizatorul inițiază plasarea comenzii, el poate selecta cantitatea dorită a fiecărui produs. Baza de date este interogată în ambii pași, verificându-se dacă produsele sunt în continuare disponibile, respectiv dacă stocul este suficient de mare. Odată trecuți acești pași, produsele sunt considerate a fi cumpărate de utilizator, iar stocul acestora se micșorează conform.

În continuare, utilizatorul trebuie să completeze un formular în legătură cu detaliile de livrare, și anume adresa de livrare și numărul de telefon. Acesta poate alege dacă detaliile de facturare sunt pentru persoană fizică sau juridică și apoi completa datele necesare: numele și adresa pentru persoana fizică, respectiv numele companiei, C.U.I., numărul de înregistrare, adresa sediului central și numele unei persoane de contact pentru persoana juridică.

În cele din urmă, utilizatorul poate revizui toate detaliile și poate finaliza comanda sau o poate anula. În cazul în care comanda este finalizată, utilizatorul primește pe mail confirmarea înregistrării comenzii,

coșul de cumpărături este golit și administratorul primește comanda în contul său. În cazul în care comanda este anulată, utilizatorul este redirecționat la coșul de cumpărături.

Notă

Implementarea funcționalităților este descrisă în următorul capitol.

4. Implementarea funcționalităților

4.1. Modelarea bazei de date

Am încercat să proiectez o diagramă entitate-relație respectând regulile de integritate teoretice și ținând cont de formele normale. Figura de mai jos arată diagrama obținută.

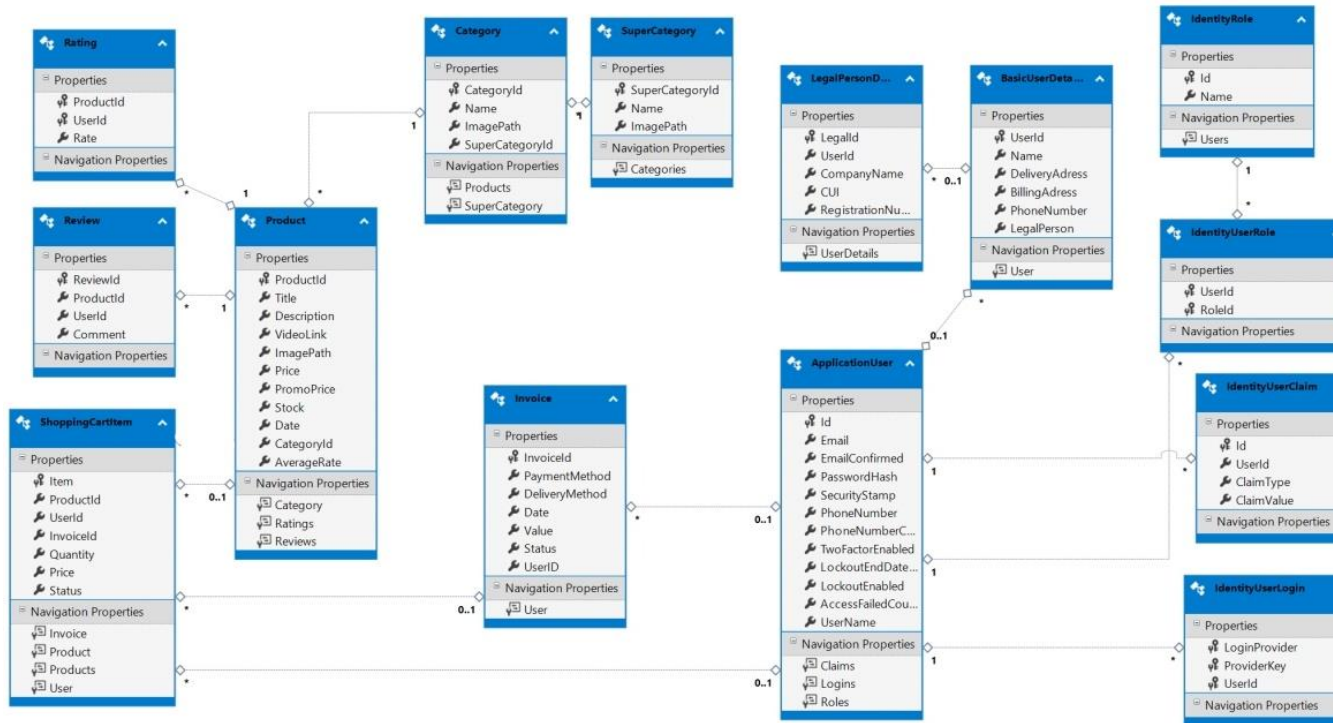


Fig. 11 - Diagrama Entitate Relație (ERD)

Entitatea *Product* reprezintă produsele firmei și pe lângă cheia primară *ProductId*, are următoarele proprietăți, tipuri de date și restricții:

[ProductId]	INT	IDENTITY (1, 1) NOT NULL
[Title]	NVARCHAR (120)	NOT NULL
[Description]	NVARCHAR (MAX)	NOT NULL
[VideoLink]	NVARCHAR (MAX)	NULL
[ImagePath]	NVARCHAR (MAX)	NULL
[Price]	INT	NOT NULL
[PromoPrice]	INT	NOT NULL
[Stock]	INT	NOT NULL
[Date]	NVARCHAR (MAX)	NULL
[CategoryId]	INT	NOT NULL
[AverageRate]	FLOAT (53)	NOT NULL

Entitățile *Category* și *SuperCategory* reprezintă sub-categoriile și categoriile principale și sunt legate prin relația *1 to many* (o categorie principală poate avea mai multe sub-categorii). De asemenea, o sub-categorii poate avea mai multe produse.

Entitatea *Rating* reprezintă rating-ul dat de un utilizator unui produs, motiv pentru cheia sa primară este compusă din *UserId* și *ProductId*.

Entitatea *ShoppingCartItem* reprezintă un produs plasat în coșul de cumpărături și, eventual, cumpărat ulterior. Diferența dintre cele două cazuri este reținută în proprietatea *Status*, care poate lua valorile *în coș* sau *cumpărat*. De asemenea, dacă produsul este cumpărat, acesta este legat prin cheie externă cu factura emisă în momentul cumpărării. Proprietatea *price* stochează prețul produsului în funcție de cantitatea aleasă ($\text{cantitate} * \text{pret/buc}$).

Entitatea *Invoice* rezolvă problema existenței relației *Many to many* dintre produse și utilizatori (un utilizator poate cumpăra mai multe produse și un produs poate fi cumpărat de mai mulți utilizatori). Aceasta mai conține date despre metoda de livrare, metoda de plată, valoarea comenzii și statusul instanței. Tabelul de mai jos explică valorile proprietății *status* și consecințele acestora.

Status	Când ia valoarea	Ce se întâmplă în plus
inițiată	Utilizatorul inițiază procesul de plasare a unei comenzi (din Coșul de cumpărături) și alege cantitatea pentru fiecare produs din coș.	O nouă factură este creată, iar produsele din coș primesc <i>id</i> -ul acesteia. Stocul produselor este modificat conform.
verificare	Utilizatorul a completat toate datele necesare și verifică sumarul comenzii.	-
trimisă	Utilizatorul a verificat toate datele și alege să finalizeze comanda.	Utilizatorul primește un e-mail cu datele comenzii, iar administratorul poate începe procesarea comenzii.
onorată	Administratorul a predat comanda firmei de curierat.	-
anulată	Utilizatorul a verificat toate datele și alege să anuleze comanda <i>sau</i> Administratorul contactează clientul și decide să anuleze comanda.	Stocul fiecărui produs din comandă crește conform datelor comenzii proaspăt anulate.

Entitatea *ApplicationUser*, alături de entitățile *IdentityRole*, *IdentityUserRole*, *IdentityUserClaim* și *IdentityUserLogin* sunt entități create prin framework-ul ASP .NET pentru adăugarea funcționalității de logare în site.

Entitățile *BasicUserDetails* și *LegalPersonDetails* reprezintă informații pentru livrare și facturare pentru persoane fizice, respectiv juridice. Tabelele rezultate sunt populate cu informațiile colectate în timpul procesului de plasare a comenzii. În cazul persoanelor juridice, ambele tabele conțin instanțe pentru persoana care face comanda.

Cheile primare sunt inițializate automat în momentul creării obiectului prin sintaxa `IDENTITY(1,1)`.

4.2. Algoritmi pentru implementarea funcționalităților

Pe lângă algoritmul descris în capitolul 2 pentru importarea datelor din fișiere Excel, am implementat algoritmi de creare, adăugare, modificare și ștergere a instanțelor în baza de date și alți algoritmi pentru interogarea bazei de date, preluarea informațiilor din mai multe tabele etc. Algoritmii au fost implementați în fișierele de tip *Controller*, trimițând date în fișierele asociate de tip *View*. Tipurile de date necesare sunt definite de clase din fișierele de tip *Model*, clase care definesc și tabelele din baza de date. Prin urmare, fiecare entitate din figura anterioară are asociat un tip de date analog. Diagrama claselor este asemănătoare diagramei Entitate - Relație.

Operațiile *CRUD* (*Create, Read, Update, Delete*)

Operațiile *CRUD* implementează funcționalitățile de adăugare, vizualizare, modificare și ștergere a produselor, sub-categoriilor, categoriilor principale, facturilor, produselor din coșul de cumpărături.

Vizualizarea unui obiect

Algoritmul pentru vizualizarea unui obiect în general este descris în pseudo-cod mai jos:

```
Vizualizare(id)  
    obiect = db.Obiecte.Find(id)
```

Trimite obiect către *View*-ul corespunzător

Vizualizarea unui produs, spre exemplu, necesită atât găsirea produsului cu cheia primară *id*, cât și preluarea tuturor recenziilor legate prin cheie externă, a comentariilor, produselor similare și produselor din aceeași categorie principală.

Vizualizarea unei comenzi, după finalizarea acesteia, necesită și preluarea obiectelor de tip *ShoppingCartItem* legate prin cheie externă de comanda respectivă, dar și preluarea obiectelor de tip *BasicUserDetails*, respectiv *LegalPersonDetails*, dacă este cazul.

Vizualizarea unei categorii principale necesită preluarea tuturor sub-categoriilor și produselor aparținente.

Adăugarea unui obiect

Adăugarea unui obiect necesită două metode: una de tip *Put*, care eventual preia date necesare din baza de date, și una de tip *Post*, care preia datele din *View* și le salvează în baza de date. Algoritmul pentru adăugarea unui obiect este descris în pseudo-cod mai jos:

```
Adăugare(Obiect obj)
    db.Obiecte.Add(obj)
    db.SaveChanges()
    Redirecționează utilizatorul către Vizualizarea obiectului nou creat
```

Adăugarea unui produs preia întâi toate sub-categoriile existente în metoda *Put* și le trimite către *View*, oferind administratorului posibilitatea de a încadra viitorul produs într-o categorie existentă.

Obiectele de tip *BasicUserDetails* și *LegalPersonDetails* sunt create și adăugate în baza de date în timpul procesului de plasare a comenzii, deci nu au metode proprii.

Modificarea unui obiect

Modificarea unui obiect este similară adăugării unuia. Diferența este că în metoda de tip *Put* obiectul care trebuie modificat este preluat din baza de date și trimis către *View*. Câmpurile formularului

primesc datele obiectului, iar modificările sunt preluate de metoda *Post*. Obiectul inițial este updatat în baza de date și modificările sunt, din nou, salvate.

```
Editare(id, Obiect obj_nou)
    obiect = db.Obiecte.Find(id)
    if (TryUpdateModel(obiect))
        obiect = obj_nou
        db.SaveChanges()
    Redirecționează utilizatorul către Vizualizarea obiectului modificat
```

Modificarea unui produs necesită preluarea tuturor sub-categoriilor în avans, prin metoda *Put*.

În unele funcții, algoritmul este apelat pentru mai multe obiecte de tipuri diferite.

Obiectele de tip *Invoice* sunt modificate în mai multe etape ale procesului de plasare a unei comenzi.

Ștergerea unui obiect

Ștergerea unui obiect preia obiectul din baza de date în funcție de cheia primară și îl șterge.

Regulile de ștergere ale obiectelor legate prin cheie externă de alte obiecte declanșează ștergerea în cascadă. Metodele de ștergere au nevoie de tag-ul `[HttpDelete]` plasat anterior definiției funcției.

```
[HttpDelete]
Ștergere(id)
    obiect = db.Obiecte.Find(id)
    db.Obiecte.Remove(obiect)
    db.SaveChanges()
    Redirecționează utilizatorul către o pagină existentă
```

Metodele de ștergere sunt apelate din *Views* printr-un formular specific, conform codului următor:

```
<form method="post" action="/Obiect/Delete/@ViewBag.Obiect.Id">
    @Html.HttpMethodOverride(HttpVerbs.Delete)
    <button type="submit" onclick="return confirm('Are you sure?');">
        Sterge produs
    </button>
</form>
```

Metodele Index

Metodele Index preiau toate obiectele de același tip din baza de date și îl trimit către *View*-ul asociat. Astfel, pagina principală a site-ului este creată prin metoda Index asociată categoriilor principale, iar paginile la care are acces doar Administratorul sunt metodele Index ale obiectelor de tip *Product*, *Invoice* și *User*. Algoritmul general al acestora este exemplificat în continuare.

```
Index()
```



```

var obiecte = from obj in db.Obiecte
              select obj
ViewBag.Obiecte = obiecte

```

Metoda Index a obiectelor de tip *ShoppingCartItem* va selecta doar produsele al căror status este *în coș* și al căror *UserId* coincide cu cel al utilizatorului logat.

Plasarea unei comenzi

Procesul de plasare a unei comenzi este format din mai multe metode de adăugare și modificare a diferitor obiecte. Tabelul de mai jos ilustrează metodele și pseudo-codul acestora. Ele sunt apelate succesiv - fiecare *view* al unei metode redirecționează utilizatorul către următoarea metodă din tabel.

Metoda	Algoritm
New [HttpPut]	<pre> preia toate produsele din Coșul de cumpărături al utilizatorului curent pentru fiecare produs dacă (produs.stoc == 0) șterge produs din coș mesaj eroare sfârșit dacă sfârșit pentru </pre>
New [HttpPost]	<pre> crează un obiect inv de tip Invoice inv.status = inițiată suma = 0 preia toate produsele din Coșul de cumpărături al utilizatorului curent pentru fiecare produs dacă produs.stoc > cantitate_ceruta if(TryUpdateModel(produs)) produs.cantitate = cantitate_ceruta produs.invoice = inv produs.stoc -= cantitate_ceruta produs.pret = cantitate_ceruta * pret_bucata db.SaveChanges() suma += produs.pret sfârșit if altfel mesaj eroare sfârșit dacă sfârșit pentru inv.valoare = suma adaugă inv în baza de date </pre>
DeliveryDetails [HttpPost]	<pre> preia inv de tip Invoice curentă var details = db.Find.BasicUserDetails(id_user_curent) </pre>

(date preluate din <i>View</i> : - adresa și numărul de telefon pentru livrare, - dacă e persoană fizică sau juridică)	dacă (details există) update details în funcție de valorile preluate din <i>View</i> altfel crează un obiect de tip <i>BasicUserDetails</i> cu valorile preluate
BillingDetails [HttpPost]	preia details = db.Find.BasicUserDetails(id_user_curent) if (TryUpdateModel(details)) modifică adresa și numele sfârșit if dacă (details.LegalPerson == "da") legalDetails = new LegalPersonDetails() legalDetails.Data = datele preluate din formular Aduagă legalDetails în baza de date sfârșit dacă preia inv de tip Invoice curentă if (TryUpdateModel(inv)) inv.PaymentMethod = preluată din formular inv.Status = "verificare" end if
InvoiceCheck	preia toate datele acumulate în pașii anteriori pentru verificare
SendMail	inv.Status = "trimisă" trimite mail utilizatorului pentru fiecare produs din coșul de cumpărături Produs.status = "cumpărat"

Alte metode

Metoda "ChangeStatus"

Metoda schimbă statusul unui obiect de tip *Invoice*, modificând câmpul în baza de date. Dacă statusul este *anulată*, stocul produselor care sunt legate prin cheie externă de factura respectivă este modificat: `produs.stoc += produs.cantitate`.

Metoda "EmptyShoppingCart"

Metoda selectează toate produsele adăugate în coșul utilizatorului curent și le șterge.

Metoda "Demand"

Metoda trimite administratorului un mail cu date despre utilizatorul care a inițiat procedura și titlul unui produs. Metoda este inițiată atunci când un produs nu mai este disponibil pe stoc și utilizatorul este interesat, apăsând butonul *Cere ofertă* de pe pagina produsului respectiv.

Metodele “ChangePrice”, “ChangeStock”, “ChangePromo”

Metodele permit administratorului să modifice datele unui produs direct din pagina *Toate produsele*, fără redirectionări.

Metoda “Promotions”

Metoda afișează toate produsele care au un preț promoțional setat (diferit de 0) și poate fi accesată de pe prima pagină sau din meniul orizontal. Administratorul stabilește ce produse au prețuri promoționale.

Metoda “Search”

Metoda primește un text și returnează toate produsele care conțin textul respectiv în titlu, numele categoriei sau numele sub-categoriei din care face parte.

5. Design

5.1. User Experience Design

Experiența utilizatorului (*UX*) este procesul de creare a produselor care oferă utilizatorilor experiențe semnificative și relevante. Aceasta implică proiectarea întregului proces de creare și integrare a produsului, inclusiv aspecte legate de *branding*, design, utilitate și funcționalitate.^[1] În acest caz, produsul este obiectul acestei lucrări, și anume aplicația web pentru firme de retail.

Dacă în capitolele anterioare am discutat despre utilitatea și funcționalitatea aplicației, în continuare mă voi referi la deciziile luate în legătură cu design-ul și *branding*-ul aplicației: structura site-ului, structura paginilor și motivul deciziilor luate. Consider că acest aspect este important deoarece studiile arată că 88% din persoanele care cumpără din mediul online nu au de gând să se întoarcă pe site-ul care le-a oferit experiență nefavorabilă ^[2]. Menținerea unei relații bune cu clienții site-ului este vitală pentru afacerea implicată.

Accesibilitate

Scopul acestei secțiuni este ca utilizatorul să poată găsi cât mai repede informația căutată în momentul accesării site-ului. Pentru îndeplinirea acestui scop, am implementat funcționalitatea de căutare a produselor și am poziționat bara de căutare în meniul orizontal de navigare, meniu care este mereu disponibil și la vedere, orice pagină ar vizualiza utilizatorul.

În plus, am adăugat un meniu orizontal secundar cu toate categoriile principale (fiecare categorie are un sub-meniu ascuns cu toate sub-categoriile aparținente) astfel încât utilizatorul să știe mereu ce fel de produse poate găsi pe site.

Pentru o și mai bună orientare și recunoaștere/găsire a produselor, paginile care afișează mai multe produse cu caracteristici comune (fac parte din aceeași categorie/sub-categorie) au și un meniu vertical pentru observarea ierarhiei și conținutului categoriilor principale.

Coșul de cumpărături este și el accesibil din meniul de navigare, pentru plasarea unei comenzi cu ușurință. La fel și profilul utilizatorului, de unde acesta are acces la comenzile făcute anterior.

Ușurința parcurgerii informației

Un studiu recent ^[3] arată că oamenii care accesează site-uri de comerț online nu citesc în totalitate informațiile oferite, motiv pentru care este important ca datele afișate să fie bine structurate și concise. De asemenea, *calls to action* trebuie să iasă în evidență astfel încât utilizatorul să ia acțiunile corespunzătoare.

Pentru a respecta acest principiu, am limitat afișarea parțială a unui produs la afișarea titlului, imaginii și prețului acestuia - cele mai importante caracteristici pe care un utilizator le caută. Informațiile sunt însoțite de un buton colorat, contrastant care îndeamnă utilizatorul să vizualizeze detaliile produsului (*call to action*).

Odată accesată pagina unui produs, acesta vede informația bine segmentată, din nou datele prioritare fiind imaginea, titlul și prețul. Utilizatorul mai poate găsi și descrierea produsului, alături de care se află videoclipul descriptiv al produsului. Principalul *call to action* de pe această pagină este plasarea produsului în coșul de cumpărături (dacă el este disponibil), sau cererea unei oferte (dacă nu este disponibil pe stoc). Figura de mai jos arată schița paginii unui produs.

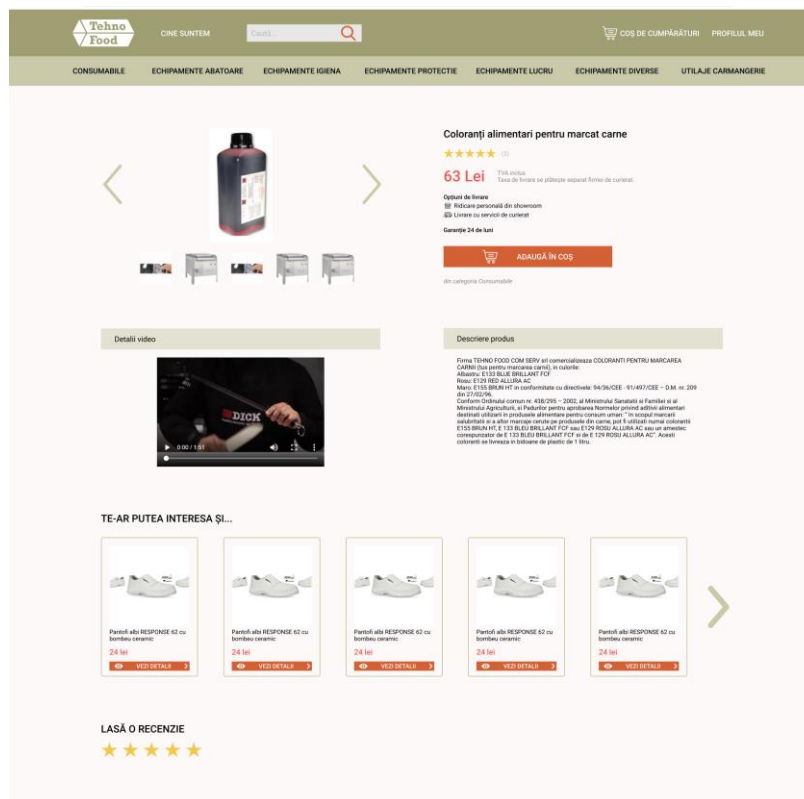


Fig. 12 - schița paginii unui produs

Credibilitatea aplicației pentru utilizatori

Credibilitatea site-ului este alcătuită din două dimensiuni: încredere și experiență. ^[4] Utilizatorii dobândesc încredere dacă informația prezentată este clară, poate fi verificată și detaliile oferite sunt suficiente. Din acest motiv, am ales ca pagina de vizualizare a unui produs să conțină detalii despre livrare, garanție și rating-ul produselor. Rating-ul și comentariile nu doar cresc traficul pe site, dar și oferă noilor utilizatori o dovadă că site-ul este real și actual.

În plus, datele firmei sunt mereu accesibile în *footer*-ul paginilor: adresa de e-mail a firmei, adresa sediului (cu poziționare pe *Google Maps*) și număr de telefon de contact. Site-ul conține și o pagină *Despre*, în care este descrisă activitatea firmei. Logo-ul și culorile principale ale firmei au fost păstrate, astfel încât clienții existenți ai firmei să recunoască ușor apartenența noului site la *brand*-ul creat anterior.

Ușurința completării datelor

Formularele sunt secționate în funcție de tema informațiilor cerute pentru o organizare clară și vizibilă pentru utilizator. *Input*-urile sunt aliniate vertical, într-o singură direcție, pentru o orientare mai ușoară. Plasarea unei comenzi are mai mulți pași, iar progresul parcurgerii acestora este afișat la fiecare pas. Figura de mai jos ilustrează înregistrarea progresului.

Logo	Cine suntem	Categorii	Cauta in site...	Cos de cumparaturi	Profilul meu
Categorie 1	Categorie	Categorie	Categorie	Categorie	Categorie





Cos de cumparaturi	Metoda livrare	Detalii facturare	Modalitate plata	Sumar comanda	Emitere factura
	Titlu	43 lei	1		
	Titlu	43 lei	1		
	Titlu	43 lei	1		
	Titlu	43 lei	1		
		Total	403 lei		

Fig. 13 - Afișarea pașilor plasării unei comenzi

5.2. User Interface Design

User Interface Design este procesul de realizare a interfețelor software sau dispozitivelor computerizate, punându-se accent pe aspect sau stil ^[5]. Scopul principal al *UI Design*-ului este ca produsele sale să fie plăcute din punct de vedere estetic și ușor de utilizat. Studiile arată că 94% din prima impresie a unui utilizator este dată de estetica site-ului ^[6]. Credibilitatea aplicației depinde și ea foarte mult de estetica acestuia.

Elementele la care mă voi referi aici sunt font-urile și culorile folosite la design-ul site-ului, dar și motivele pentru care ele au fost alese.

Font-uri

Am folosit două font-uri: unul de tip *Display*, pentru evidențierea titlurilor și subtitlurilor, și unul de tip *Sans Serif*, pentru restul conținutului. Ele denotă seriozitate și profesionalism, caracteristici necesare dobândirii încrederii în site.

Font-urile folosite sunt: , respectiv . Dimensiunile elementelor de tip text se găsesc în tabelul următor.

Ecran	h1	h2	Titlul produsului	p, ul, body	butoane	navbar
desktop	24px	18px	18px	14px	14px	13px
mobile						

Culori

Am păstrat culorile principale ale *branding*-ului firmei, adăugând nuanțe noi și diferite nuanțe de gri.

Calls to action au mereu aceeași culoare pentru construirea unui tipar consistent și provocarea utilizatorului să intuiască acțiunile. Am ținut cont de contrastul dintre culoarea fundalului și cea a textului pentru ca acesta să fie ușor de citit. Culoarea butoanelor este puternică, pentru a sugera dinamism. Culorile utilizate sunt des întâlnite în reprezentarea domeniului industriei alimentare și pot fi observate în figura de mai jos.



Fig. 14 - Culorile folosite și codurile acestora (HEX)

6. Search Engine Optimization (SEO)

SEO este practica optimizării paginilor web pentru a le face să ajungă pe o poziție cât mai înaltă în rezultatele căutării motoarelor de căutare. *SEO* se concentrează pe îmbunătățirea clasamentului în rezultatele căutării organice (rezultate directe și nu reclame plătite). Această practică este folosită pentru creșterea traficului pe site în mod *natural* ^[7].

Deși rezultatele depind foarte mult de conținutul site-ului, performanța acestuia este un factor important când vine vorba de SEO. De asemenea, *tag*-urile specifice știute drept *metadata* sunt și ele definitorii, la fel și relevanța legăturii dintre titlul paginii și conținutul *heading*-urilor, spre exemplu.

Metadata

Următoarele *metadata* sunt implementate implicit odată cu crearea unei aplicații ASP .NET și se referă la codificarea textului și la dimensiunea ecranului pe care pagina va fi încărcată.

```
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Am adăugat *metadata* și titluri specifice pentru fiecare pagină în parte, în mod asemănător cu exemplul de mai jos:

```
<title>{{ .Title }}</title>
<meta name="Description" content="{{ .Description }}">
<meta name="Keywords" content="{{ .Keywords }}">
```

Titlul paginii coincide de cele mai multe ori cu textul aflat în *tag*-ul *h1*, meta-descrierea unui produs coincide cu descrierea unui produs, respectiv detalii despre anumite categorii. Cuvintele-cheie sunt alese de administratorul firmei.

Pagination

Paginile care conțin mai multe produse au fost împărțite în mai multe pagini prin practica numită *pagination*. Astfel, dacă o categorie are 100 de produse, acestea vor fi afișate pe două pagini diferite:

prima va afișa primele 50 de produse, iar a doua va afișa produsele rămase. Deși titlul și *metadatele* se aseamănă, conținutul este diferit, astfel că distribuirea produselor în acest mod nu crează conținut duplicitar. Paginile se vor încărca astfel mai ușor, experiența utilizatorului va fi îmbunătățită și performanța site-ului (viteza de încărcare) va crește.

Robots.txt

Am adăugat un fișier numit *robots.txt* pentru a spune *web-crawlers* ce fișiere nu trebuie parcurse. Am inclus paginile pentru logare, schimbarea parolei, crearea unui cont deoarece nu conțin informații relevante indexării site-ului.

Internal linking

Link-urile interne sunt importante pentru indexarea site-ului și ierarhizarea paginilor. Pentru a implementa acest beneficiu, am adăugat link-uri specifice categoriei principale și sub-categoriei unui produs pe pagina acestuia, pe lângă link-urile din paginile de prezentare ale categoriilor.

External linking

Footer-ul site-ului conține link-uri cu cărți de specialitate și cataloage de produse.

Proprietatea *alt* a imaginilor

Fiecare imagine are completată proprietatea *alt* cu titlul produsului. Numele imaginii este de asemenea reprezentativ. Astfel, titlul produsului apare deja de minim 4 ori pe pagină (titlul paginii, titlul produsului - h1, titlul imaginii și proprietatea *alt*), obținându-se un rezultat bun pentru SEO fără exagerări contextuale.

7. Securitate

Framework-ul ASP .NET are implementate implicit măsuri de securitate de care dezvoltatorii pot profita. Măsurile țin cont de obiectivele criptografiei moderne, și anume: confidențialitatea informației, integritatea datelor, disponibilitatea datelor, autentificarea entităților și non-repudierea ^[8]. Voi dezvolta, în continuare, îndeplinirea fiecărui obiectiv.

Confidențialitate

Confidențialitatea se referă la păstrarea secretului informației, accesul la informația sensibilă fiind disponibilă doar persoanelor autorizate.

Distingem, în cazul aplicației prezentate, două tipuri de informații sensibile: istoricul comenzilor și datele furnizate administratorului despre propria afacere.

Istoricul comenzilor se poate vizualiza doar prin intermediul autentificării utilizatorului.

Datele furnizate administratorului pot fi accesate prin butoane vizibile doar administratorului. Totuși, presupunând că cineva ar bănuî link-ul unei anumite metode, am adăugat acestor metode o cheie care obligă utilizatorul să fie logat cu un cont al cărui rol este cel de Administrator.

Integritatea datelor

Integritatea datelor se referă la eliminarea posibilității de modificare (schimbare, inserare, ștergere) neautorizată a informației. Cel mai întâlnit atac la integritatea datelor este cel numit *SQL Injection*, adică încercarea de a trimite diferite *queries* către baza de date prin intermediul unor parametri de tip text. ASP .NET rezolvă însă această problemă prin utilizarea limbajului *LINQ* și funcțiilor prestabilite de preluare a datelor din baza de date.

Disponibilitatea datelor

Disponibilitatea datelor se referă la permiterea entităților autorizate să acceseze în timp util și fiabil informația. Disponibilitatea aplicației depinde de conexiunea la internet a utilizatorului și a serverului.

Autentificarea și non-repudierea

Autentificarea este sigură întrucât parola utilizatorului este transformată în *hash code* înainte de inserarea ei în baza de date. Astfel, parola nu este niciodată reținută în *plain text*. În plus, parola trebuie să conțină caractere de tip: litere mari, litere mici, cifre și simboluri, crescând foarte mult spațiul parolelor.

8. Tehnologii

Platforma .NET

.NET este un *framework* de dezvoltare software unitară care permite realizarea, distribuirea și rularea aplicațiilor-desktop Windows și aplicațiilor Web. Tehnologia .NET unește mai multe tehnologii (ASP, XML, OOP, SOAP, WDSL, UDDI) și limbaje de programare (VB, C++, C#, J#), asigurând atât portabilitatea codului compilat între diferite calculatoare cu sistem Windows, cât și reutilizarea codului în programe, indiferent de limbajul de programare utilizat.

Componenta .NET Framework stă la baza tehnologiei .NET, fiind ultima interfață între aplicațiile .NET și sistemul de operare și conține:

- Limbajele C#, VB.NET, C++ și J#. Pentru a fi integrate în platforma .NET toate aceste limbaje respectă specificațiile de Programare Orientată pe Obiecte. Ele au ca elemente de bază: clase, interfețe, tipuri valoare și referință, mecanisme de moștenire, polimorfisme și tratarea excepțiilor.
- Platforma comună de executare a programelor numită Common Language Runtime (CLR).
- Ansamblul de biblioteci necesare în realizarea aplicațiilor desktop sau Web numit Framework Class Library (FCL).^[9]

Figura următoare ilustrează arhitectura *framework*-ului prezentat.

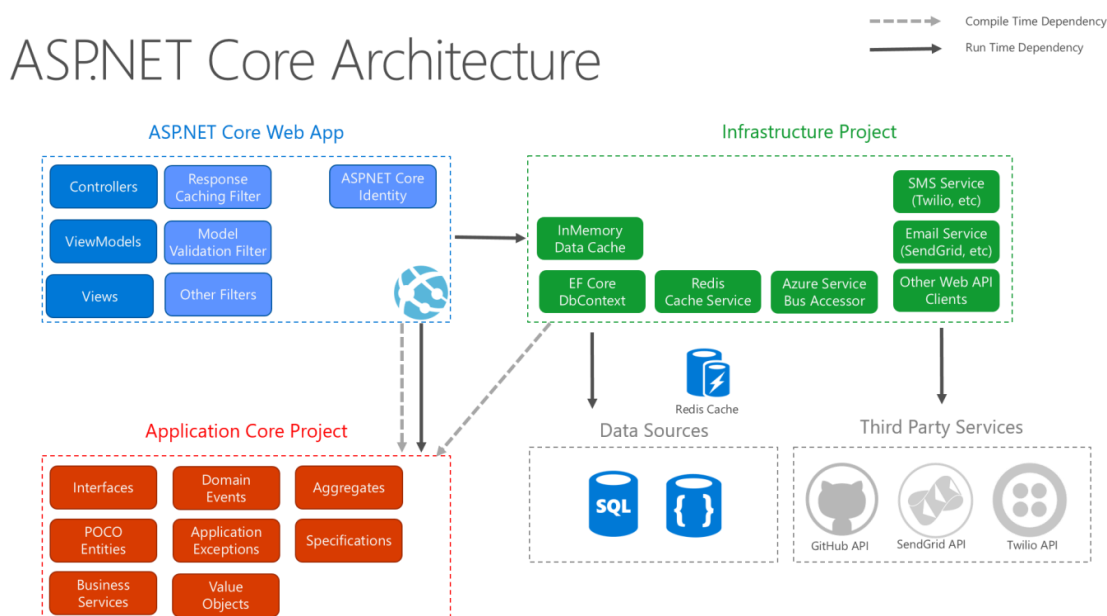


Fig. 15 - Arhitectura ASP .NET

Am folosit .NET Framework versiunea 4.6.1 cu arhitectura MVC - *Model View Controller*. Figura următoare ilustrează *layerele* acestei arhitecturi.

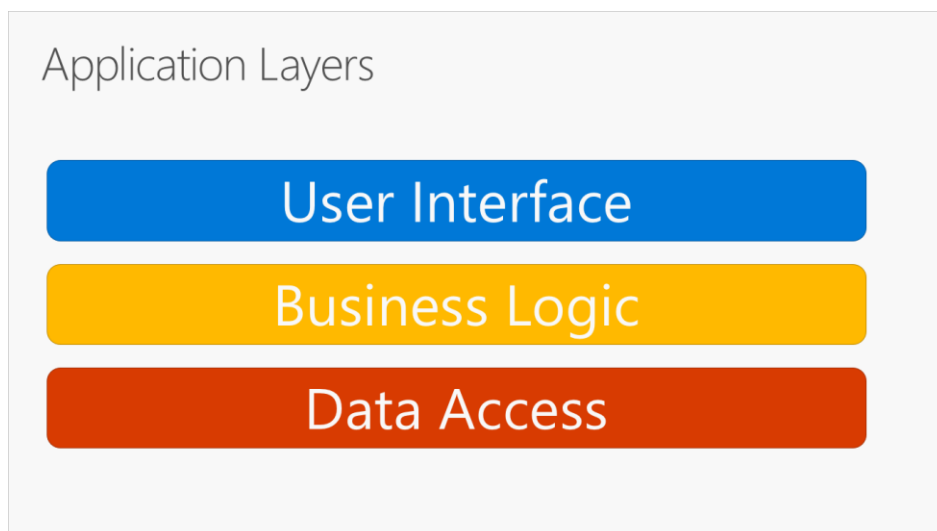


Fig. 16 - MVC

Arhitectura MVC în ASP .NET

MVC este un *design pattern* folosit pentru a segmenta interfața utilizatorului (*view*), informația stocată (*model*), și logica aplicației (*controller*). Acest model ajută la separarea modurilor de implementare.^[10] Succesul modelului se datorează izolării logicii aplicației (*business logic*) față de interfața cu utilizatorul, rezultând o aplicație unde aspectul vizual sau nivelele inferioare ale regulilor de business sunt mai ușor de modificat, fără a afecta alte nivele.^[11]

Controllers

Controller-ul este reprezentat de clase scrise cu limbajul C# și este partea aplicației care se ocupă de interacțiunea cu utilizatorul. În *controller* se citesc datele introduse de utilizator, se trimit către *Model*, se execută operațiile, după care se trimite răspunsul către *View*.

Model

Modelul este responsabil cu gestionarea datelor din aplicație și manipularea acestora. Acesta răspunde cererilor care vin din *View*, cât și instrucțiunilor din *Controller*. Este cel mai de jos nivel care se ocupă cu procesarea și manipularea datelor, reprezentând nucleul aplicației (aici se face legătura cu baza de date).

View

Viewul reprezintă interfața cu utilizatorul și este partea în care se afișează datele, adică înregistrările din baza de date și informațiile generate de aplicație. *Viewul* conține de cele mai multe ori limbaj HTML, CSS, JS și este partea văzută de către utilizator prin intermediul browserului.

Limbajul C#

Limbajul C# este un limbaj compilat, orientat pe obiecte utilizat mai ales împreună cu *framework-ul* .NET. Am folosit acest limbaj pentru implementarea Controllerelor și a claselor din Model.

Librăriile folosite sunt afișate în figura următoare.

```
using Microsoft.AspNet.Identity;
using OnlineStore.Models;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Runtime.InteropServices;
using System.Web;
using System.Web.Mvc;
using Excel = Microsoft.Office.Interop.Excel;
using PagedList;
using System.Net.Mail;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
```

Fig. 17 - Librării folosite

Entity Framework

Entity Framework este un pachet care poate fi instalat folosind *NuGet* și care suportă tehnica *Code First*. Tehnica *Code First* oferă dezvoltatorilor posibilitatea de a scrie clase (modele) prin intermediul cărora baza de date va fi generată automat. Acest lucru duce la o dezvoltare curată și rapidă a aplicațiilor cu baze de date. Figura de mai jos arată meniul de instalare al *framework-ului*.

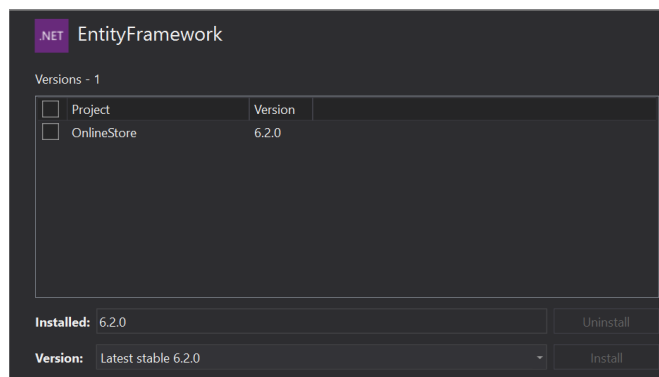


Fig. 18 - EntityFramework via NuGetPackageManager

Fiecare clasă definită în folderul *Models* reprezintă un tabel din baza de date. Asocierea dintre acestea se face prin linia de cod următoare:

```
public DbSet<Obiect> Obiecte { get; set; },
```

unde *Obiect* = clasa definită,

Obiecte = numele tabelului asociat.

Restricțiile pentru baza de date sunt stabilite prin intermediul *Annotations*, iar cheile externe prin declararea obiectelor de tip *virtual*. Figura următoare arată clasa denumită *Product*, asociată tabelului de produse din baza de date.

```
public class Product
{
    [Key]
    public int ProductId { get; set; }
    [Required(ErrorMessage = "Please insert a unique title")]
    [StringLength(120, ErrorMessage = "Maximum of length is 120 characters")]
    public string Title { get; set; }
    [Required(ErrorMessage = "Please insert a description")]
    [MinLength(10, ErrorMessage = "Minimum of length is 10 characters")]
    public string Description { get; set; }
    public string VideoLink { get; set; }

    public string ImagePath { get; set; }
    [NotMapped]
    public HttpPostedFileBase ImageFile { get; set; }

    [Required(ErrorMessage = "Please insert a price")]
    public int Price { get; set; } = 54;
    public int PromoPrice { get; set; } = 0;
    public int Stock { get; set; } = 10;

    public string Date { get; set; }
    [Required(ErrorMessage = "Category field is mandatory")]
    public int CategoryId { get; set; }
    public virtual Category Category { get; set; }

    public double AverageRate { get; set; }
    public virtual ICollection<Review> Reviews { get; set; }
    public virtual ICollection<Rating> Ratings { get; set; }
}
```

Fig. 19 - Modelul unui produs - restricții și chei externe

HTML, CSS, JavaScript, ASP .NET Razor

HTML, CSS și JavaScript sunt limbaje de programare utilizate pentru construirea paginilor web. Am utilizat CSS și JavaScript pentru stilizarea paginilor prin regulile descrise în capitolul *Design*. Am folosit, în plus, librăriile oferite de *framework*-ul Bootstrap versiunea 4.2.1, *framework* importat automat în momentul creării unei aplicații web ASP .NET MVC.

Sintaxa ASP .NET Razor permite scrierea de cod preluat din Controller în fișiere de tip *View*.

9. Concluzii

Experiența ultimilor 3 ani m-a ajutat să transform cunoștințele acumulate într-un produs finit: o aplicație web pentru firme de retail de mici dimensiuni, dezvoltată cu *framework-ul* ASP .NET MVC, cu ajutorul Entity Framework și limbajelor de stilizare a paginilor web.

Am proiectat aplicația având în vedere funcționalitățile oferite de alte platforme, platforme pe care le-am încadrat într-o analiză competitivă a firmei-client.

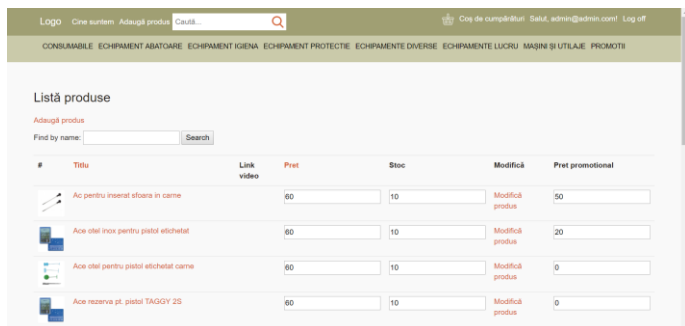
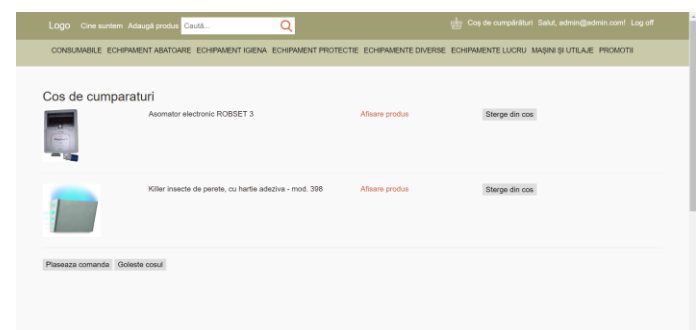
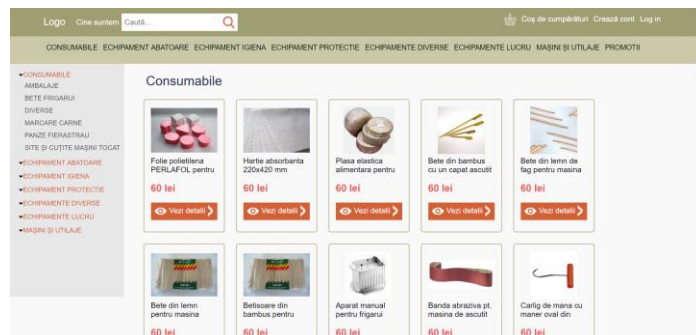
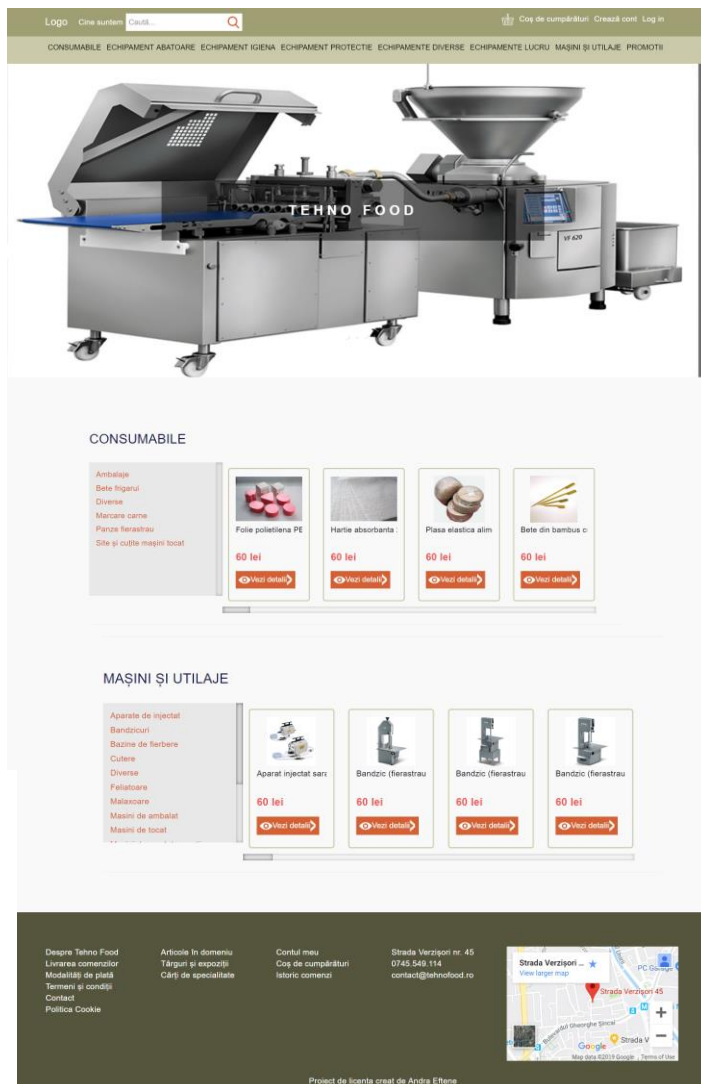
Pe parcursul dezvoltării aplicației, m-am confruntat cu probleme de proiectare a bazei de date sau a design-ului, cu probleme legate de transmiterea datelor către *Views*, dar legate și de preluarea datelor și inserarea lor în baza de date. Experiența anterioară cu aceste tehnologii a fost scurtă, iar acomodarea cu mediul de programare a fost cea mai grea parte a procesului de dezvoltare. Am rezolvat aceste probleme căutând soluții în mediul online sau în cursurile predate la facultate. Sursele acestor rezolvări pot fi găsite în capitolul *Bibliografie și referințe*.

Cel mai mare beneficiu al acestei lucrări este că aplicația va fi în curând lansată, nu înainte de a o testa îndeaproape din punct de vedere al accesibilității și al performanței. Versiunea prezentată în această lucrare va fi îmbunătățită, în funcție de cerințele adiționale care vor apărea pe parcursul colaborării cu clientul. Mai menționez că unele funcționalități au fost cerute în mod special de către client. Cel mai probabil, produsele vor avea mai multe caracteristici în funcție de tipul acestora (îmbrăcămintea va avea mărimi, accesoriile vor avea dimensiuni diferite etc). Plata cu cardul va fi implementată după ce angajații firmei se vor obișnui cu administrarea aplicației pentru a evita experiențe neplăcute pentru potențialii clienți.

Totodată, caracterul general al aplicației va permite atragerea altor clienți din industria de retail și adaptarea proiectului la noi cerințe.

10. Anexe

Figurile următoare ilustrează capturi de ecran ale aplicației.



11. Bibliografie și referințe

Bibliografie

Următoarele articole m-au ajutat la implementarea aplicației.

1. Tutorial: Add sorting, filtering, and paging with the Entity Framework in an ASP.NET MVC application: <https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/getting-started-with-ef-using-mvc/sorting-filtering-and-paging-with-the-entity-framework-in-an-asp-net-mvc-application>
2. Sourabh Mishra, Sending An E-Mail Using ASP.NET MVC: https://www.c-sharpcorner.com/UploadFile/sourabh_mishra1/sending-an-e-mail-using-Asp-Net-mvc/
3. <https://www.drupal.org/node/2283649>
4. <https://stackoverflow.com>
5. Cezara Benegui, Curs Dezvoltarea Aplicatiilor Web, Facultatea de Matematica si Informatica, Universitatea din Bucuresti

Referințe

Am citat următoarele surse:

- [1] The Interaction Design Foundation: <https://www.interaction-design.org/literature/topics/ux-design>
- [2] David Moth, Site speed: case studies, tips and tools for improving your conversion rate: <https://econsultancy.com/site-speed-case-studies-tips-and-tools-for-improving-your-conversion-rate/>
- [3] 9 UX Tips on How to Create an Ecommerce Store That Really Sells: https://rubygarage.org/blog/ecommerce-ux-tips#article_title_8
- [4] B.J. Fogg: https://rubygarage.org/blog/ecommerce-ux-tips#article_title_8
- [5] The Interaction Design Foundation: <https://www.interaction-design.org/literature/topics/ui-design>
- [6] Jozef Toth, 13 impressive statistics on user experience: <https://www.invisionapp.com/inside-design/statistics-on-user-experience/>
- [7] Marieke van de Rakt, What is SEO: <https://yoast.com/what-is-seo/>
- [8] Ruxandra Olimid, Curs Criptografie și Securitate, Facultatea de Matematică și Informatică, Universitatea din București
- [9] M.E.C.T, Microsoft, Introducere în Programarea .Net Framework: http://tminfo.ro/upload/files/410682583Manual_final.pdf
- [10] <https://dotnet.microsoft.com/apps/aspnet/mvc>

[11] Cezara Benegui, Curs Dezvoltarea Aplicațiilor Web, Facultatea de Matematică și Informatică, Universitatea din București

Figuri

1. Common Sense Advisory

2. Yeoman Research Team, US Retail Ecommerce Sales Jump 18% in Q2 2013 - Continuing 10 Year Growth Trend

<http://www.yeomantechnologies.com/profiles/blog/us-retail-e-commerce-sales-jump-18-in-q2-2013-10-year-trend>

3,4,5. Platforma Drupal a firmei-client

15, 16. Common Web Application Architecture <https://docs.microsoft.com/en-us/dotnet/standard/modern-web-apps-azure-architecture/common-web-application-architectures>