

MAGAZIN WEB PENTRU ARTICOLE VESTIMENTARE

Candidat: Mihaela-Crăciunela MUNTIAN

Coordonatori științifici:

Prof. dr. ing. Lăcrămioara STOICU-TIVADAR

As. drd. ing. Stelian-Nicolae NICOLA

Sesiunea: Septembrie 2022

Cuprins

1. INTRODUCERE	5
1.1 DOMENII	5
1.1.1 DEZVOLTARE WEB.....	5
1.1.2 DESIGN WEB.....	6
1.1.3 BAZE DE DATE	6
1.1.4 SECURITATE	6
1.1.5 TESTARE	7
1.2 TEMA LUCRĂRII	7
1.2.1 MOTIVUL REALIZĂRII PROIECTULUI	7
1.2.2 CERINȚELE PROIECTULUI.....	7
1.3 OBIECTIVE	7
1.4 STRUCTURA LUCRĂRII.....	8
2. ANALIZA STADIULUI ACTUAL ÎN DOMENIUL PROBLEMEI	9
2.1. LUCRĂRI ASEMĂNĂTOARE	9
2.2. CONCLUZII	14
3. BAZELE TEORETICE.....	15
3.1. VISUAL STUDIO CODE	15
3.1.1. EXTENSII	15
3.1.2. GIT.....	16
3.2. HTML.....	17
3.3. CSS.....	18
3.4. JAVASCRIPT	19
3.4.1. BIBLIOTECI	19
3.5. FIREBASE.....	21
4. SOLUȚIA PROPUȘĂ ȘI METODOLOGIA DE PROIECTARE	22
4.1. ARHITECTURA GENERALĂ A SISTEMULUI.....	22
4.2. DIAGrame UML.....	22
4.2.1. DIAGrame DE UTILIZARE	22
4.2.2. DIAGrame DE ACTIVITATE	24
4.3. STRUCTURA BAZEI DE DATE.....	27
5. IMPLEMENTARE	30
5.1. CREARE CONT	30
5.2. AUTENTIFICARE	31
5.2.1. METODA signWithRedirect()	31

5.2.2.	METODA signInWithEmailAndPassword()	32
5.2.3.	METODA onAuthStateChanged()	32
5.2.4.	METODA sendPasswordResetEmail()	33
5.3.	ADĂUGAREA ȘI ELIMINAREA PRODUSELOR DE LA FAVORITE	34
5.4.	AFIȘAREA PRODUSELOR RECOMANDATE	35
5.5.	ADĂUGAREA PRODUSELOR ÎN COȘUL DE CUMPĂRĂTURI	36
5.6.	ȘTERGEREA PRODUSELOR DIN COȘUL DE CUMPĂRĂTURI	38
5.7.	PLASAREA UNEI COMENZI	38
5.8.	CĂUTAREA UNUI PRODUS	40
5.9.	SORTAREA PRODUSELOR	40
5.10.	DECONNECTAREA UTILIZATORULUI	40
5.11.	NOTIFICAREA UTILIZATORULUI	41
6.	UTILIZAREA APLICAȚIEI	43
6.1.	CONFIGURAȚII	43
6.2.	MANUAL DE UTILIZARE	43
6.2.1.	PAGINA DE AUTENTIFICARE	43
6.2.2.	PAGINA DE RESETARE A PAROLEI	45
6.2.3.	PAGINA DE ÎNREGISTRARE	46
6.2.4.	PAGINA DE ACASĂ	47
6.2.5.	PAGINA DE PRODUSE	48
6.2.6.	PAGINA DE DETALII	49
6.2.7.	PAGINA DE CONTACT	51
6.2.8.	PAGINA DE PRODUSE FAVORITE	51
6.2.9.	PAGINA COȘULUI DE CUMPĂRĂTURI	52
6.2.10.	PAGINA DE PROFIL	53
6.2.11.	PAGINA DE FINALIZARE COMANDĂ	53
6.3.	PRINCIPII EURISTICE	54
6.3.1.	PRINCIPIUL 1 – ASIGURAREA VIZIBILITĂȚII STĂRII ÎN CARE SE AFLĂ SISTEMUL	55
6.3.2.	PRINCIPIUL 2 – CREAREA UNUI MODEL AL SISTEMULUI CARE SĂ FIE COMPATIBIL CU REALITATEA	55
6.3.3.	PRINCIPIUL 3 – CONTROLUL ȘI LIBERTATEA UTILIZATORULUI	55
6.3.4.	PRINCIPIUL 4 – CONSISTENȚĂ ȘI STANDARDE	56
6.3.5.	PRINCIPIUL 5 – PREVENIREA ERORILOR	56
6.3.6.	PRINCIPIUL 6 – RECUNOAȘTERE MAI DEGRABĂ DECÂT MEMORARE	57
6.3.7.	PRINCIPIUL 7 – FLEXIBILITATE ȘI EFICIENȚĂ ÎN UTILIZARE	57

6.3.8. PRINCIPIUL 8 – PROIECTARE ESTETICĂ ȘI MINIMALISTĂ.....	57
6.3.9. PRINCIPIUL 9 – AJUTAȚI UTILIZATORUL SĂ RECUNOASCĂ, SĂ DIAGNOSTICHEZE ȘI SĂ REVINĂ DINTR-O EROARE	57
6.3.10. PRINCIPIUL 10 – SUPORT (HELP) ȘI DOCUMENTARE	58
7. CONCLUZII.....	59
BIBLIOGRAFIE	60

1. INTRODUCERE

Apariția internetului reprezintă un eveniment important din istoria societății, nu numai pentru că a facilitat comunicarea între calculatoarele și echipamentele folosite la scară largă, ci și datorită faptului că a deschis noi orizonturi în domeniul tehnologiei și al comunicării.

În prezent, numărul de persoane care folosesc internetul în mod regulat este de aproximativ 5.03 miliarde, acesta reprezentând aproape 65% din întreaga populație a planetei [1]. Totodată, studiile statistice preconizează faptul că până la finalul anului 2030, numărul de utilizatori ai internetului va ajunge la 90% din populația planetei [1].

De asemenea, putem afirma cu certitudine faptul că internetul este prezent în majoritatea domeniilor de activitate, precum domeniul auto (mașini inteligente, hărți în timp real etc.), domeniul educațional (învățământ la distanță, catalog virtual etc.), domeniul social (poșta electronică, platforme de comunicare tip social-media), domeniul de marketing (magazin online, platforme de vânzare-cumpărare produse etc.) și lista poate continua.

1.1 DOMENII

În cadrul dezvoltării unui site sunt prezente mai multe domenii fără de care acesta nu ar fi funcțional sau sigur. Cele mai importante fiind prezentate mai jos.

1.1.1 DEZVOLTARE WEB

Dezvoltarea web devine unul dintre factorii importanți în industriile și companiile din întreaga lume. Aceasta se referă la construirea, crearea și întreținerea site-urilor web, care au un rol major în companii, deoarece ele pot crea un alt mediu în vânzare, promovare și publicitate de produse sau servicii care se numește comerț electronic [2].

Tehnologia *full stack* se referă la întreaga profunzime a unei aplicații de sistem computerizat, iar dezvoltatorii *full stack* se încadrează în două domenii separate de dezvoltare web: *front-end* și *back-end* [3].

Front-end și *back-end* sunt unii dintre cei mai cunoscuți termeni folosiți în dezvoltarea web. Acești termeni sunt esențiali pentru dezvoltarea web, dar sunt destul de diferiți unul față de celălalt. Fiecare parte trebuie să comunice și să opereze eficient una cu cealaltă, asemenea unei singure entități, pentru a îmbunătăți funcționalitatea site-ului web [3].

Un dezvoltator *front-end* se ocupă cu partea pe care utilizatorii o văd și cu care interacționează. Acesta creează interfața cu utilizatorul (UI), ce asigură că aspectele vizuale ale unui site web sunt funcționale. Partea de *front-end* mai este denumită și „partea de client” a aplicației, aceasta incluzând tot ceea ce utilizatorii experimentează direct: culori, stiluri de text, imagini, grafice, tabele, butoane și meniul de navigare. *HTML* (Hyper Text Markup Language), *CSS* (Cascading Style Sheets) și *JavaScript* sunt limbajele folosite pentru dezvoltarea *front-end* [4].

Un dezvoltator *back-end* se concentrează pe partea de funcționalitate a site-ului web pe care utilizatorii nu o pot vedea, adică nu intră în contact direct cu utilizatorii. Părțile și caracteristicile dezvoltate de designerii *back-end*, sunt accesate de către utilizatori printr-o aplicație *front-end*. Dezvoltarea *back-end*, denumită și „*partea de server*”, se ocupă de stocarea și utilizarea datelor, iar o bună funcționare a *front-end*-ului este asigurată de această parte. [4].

1.1.2 DESIGN WEB

Designul web se concentrează pe factori estetici, precum interfața cu utilizatorul ce conține elementele vizuale care dau viață site-ului și îl fac mai ușor de utilizat. Orice designer web trebuie să cunoască elementele de bază din *HTML* și *CSS*. *HTML* este folosit pentru structurarea paginilor web, iar *CSS* pentru stilizarea paginilor și a componentelor existente în pagina web [5].

1.1.3 BAZE DE DATE

O bază de date este o colecție de date structurate, de obicei stocate electronic, într-un sistem informatic. Cele mai utilizate tipuri de baze de date sunt cele relaționale care utilizează limbajul *SQL* (Structured Query Language) sau cele non-relaționale, cunoscute și ca *NoSQL* [6].

SQL este un limbaj standard utilizat de bazele de date relaționale, având rolul de a interoga, manipula și defini date, pe când o bază de date *NoSQL* este una non-relațională, ce permite stocarea și manipularea datelor nestructurate. Aceste baze de date *NoSQL* au devenit populare pe măsură ce aplicațiile web au devenit mai comune și mai complexe [6].

Tabelul 1. Tipuri de baze de date [7]

	SQL	NOSQL
Tip	relațională	non-relațională
Date	date structurate și stocate în tabele	date nestructurate stocate în fișiere JSON
Scalabilitate	verticală	orizontală
Limbaj	structurat	nestructurat
Schemă	statică	dinamică

1.1.4 SECURITATE

Această creștere masivă a adoptării comerțului electronic a condus la o nouă generație de amenințări de securitate. Când vine vorba despre un site de comerț electronic, avem în vedere un model standard client-server care conține următoarele trei componente: sistemul server, rețeaua și sistemul client. Prin urmare, fiecare dintre aceste componente trebuie să fie protejate de orice amenințare sau atac cibernetic [8].

1.1.5 TESTARE

Testarea web este o practică de testare a software-ului pentru a verifica dacă există eventuale erori. Înainte de a face publică aplicația sau site-ul web, trebuie efectuată o testare completă, care constă în testarea funcționalității, testare de utilizare, testarea interfeței, testarea de compatibilitate, testarea performanței și testarea de securitate [9].

1.2 TEMA LUCRĂRII

În cadrul acestei lucrări doresc să dezvolt un magazin web pentru articole de îmbrăcăminte destinate persoanelor de sex feminin.

1.2.1 MOTIVUL REALIZĂRII PROIECTULUI

Această alegere de a realiza un magazin online se datorează creșterii considerabile a numărului de comenzi plasate online în ultima perioadă din cauza pandemiei. În contextul pandemiei globale de *COVID-19* care continuă să înregistreze o extindere rapidă în majoritatea țărilor, comerțul electronic a devenit o barcă de salvare și pentru cei mai neexperimentați dintre cumpărătorii din mediul online, determinând o mare parte din consumatori să apeleze la comerțul electronic, chiar dacă a fost pentru prima dată.

Astfel, consumatorii favorizează opțiunea de a comanda produsele dorite din confortul de acasă prin intermediul internetului.

1.2.2 CERINȚELE PROIECTULUI

Proiectul va fi realizat în mediul de dezvoltare Visual Studio Code, utilizând următoarele limbaje: *HTML* (Hyper Text Markup Language), *CSS* (Cascading Style Sheets) pentru partea de *front-end* și *JavaScript*, atât pentru design, cât și pentru funcționalitate. Datele utilizatorilor, produselor, cât și a comenzilor vor fi stocate în *Firebase Realtime Database*.

Acest site va dispune de funcționalitățile clasice pe care le are orice site de comerț electronic, iar autentificarea se va face fie printr-un cont creat pe site, fie direct prin contul de *Google*. Site-ul va dispune și de funcționalități de inserare, selectare, modificare și ștergere dintr-o bază de date *NoSQL*.

1.3 OBIECTIVE

Principalul obiectiv în realizarea lucrării de licență este acela de a dezvolta un proiect utilizând limbajul *Javascript* și baza de date *Firebase*. Am făcut această alegere deoarece, pe parcursul facultății am ajuns să studiez doar bazele acestora, acum venind timpul să aprofundez acele cunoștințe și să le pun în aplicare, fapt pentru care pot afirma că a fost o provocare pentru mine să dezvolt acest proiect.

Fiind primul meu proiect de acest nivel în domeniul dezvoltării web, un scop secundar îl reprezintă descoperirea unei noi pasiuni. În urma acestui proiect urmează să aflu în spre ce domeniu din IT mă voi îndrepta.

1.4 STRUCTURA LUCRĂRII

În privința structurii, proiectul este alcătuit din opt capitole. Pe parcursul acestora au fost descrise noțiuni teoretice care au ajutat la realizare proiectului de licență.

În primul capitol se va face o scurtă punere în revistă a domeniilor ce înglobează tema propusă, cât și a principalelor mele obiective care m-au determinat să finalizez acest proiect.

Se prezintă în al doilea capitol, analiza unor publicații științifice publicate pe teme asemănătoare cu cea a lucrării actuale.

Capitolul trei conține noțiunile teoretice referitoare la tehnologiile folosite pentru implementarea proiectului, cum ar fi, Visual Studio Code, Firebase Realtime Database, dar și limbajele HTML, CSS, JavaScript.

În al patrulea capitol este prezentată arhitectura generală a sistemului, structura fișierelor din baza de date, cât și o serie de diagrame UML, utilizate pentru descrierea funcțiilor.

Capitolul cinci înglobează implementarea lucrării, mai concret, în acest capitol sunt descrise mai amplu funcționalitățile de care dispune site-ul.

Modul în care ar trebui ar trebui utilizat site-ul și minimul de configurații de care trebuie să dispună utilizatorul este descris pas cu pas în capitolul șase.

În capitolul șapte sunt prezentate concluziile lucrării de licență, precum și tendințele de evoluare în vederea extinderii și îmbunătățirii.

Capitolul opt cuprinde o serie de referințe ale lucrării.

2. ANALIZA STADIULUI ACTUAL ÎN DOMENIUL PROBLEMEI

În acest capitol o să descriu două aplicații asemănătoare cu lucrarea mea, fiind vorba despre comerțul electronic. În urma descrierii caracteristicilor și a funcționalităților acestor aplicații, voi realiza un tabel comparativ în privința acestora.

2.1. LUCRĂRI ASEMĂNĂTOARE

În lucrarea [10] este descrisă o aplicație web de comerț electronic care comercializează în prezent articole de îmbrăcăminte pentru bărbați (vezi *Figura 2.1*). Această aplicație permite doar vizualizarea produselor disponibile, care ulterior pot fi achiziționate folosind procesul de plăți *PayPal* (Instant Pay). În cadrul acestei lucrări au fost utilizate următoarele tehnologii: mediul de dezvoltare *Visual Studio*, *Asp.net*, limbajele *HTML*, *CSS*, *JavaScript* pentru partea de *front-end* și *C#* pentru partea de *back-end*, totodată a fost folosită o bază de date *SQL*, utilizând mediul *SQL Server Management Studio Express*.



Figura 2.1. Interfața aplicației [10]

În *Figura 2.2* este prezentată o diagramă generală de utilizare a aplicației. După cum se poate observa în figura de mai jos, utilizatorii acestei aplicații dispun de trei roluri precum, cel de vizitator, utilizator și administrator. Astfel, oricine poate vizualiza produsele, însă doar cei autentificați pot plasa o comandă. Utilizatorii care nu sunt înregistrați fac parte din categoria de vizitatori, aceștia având permisiunea de a se înregistra și de a vizualiza produsele, pe când un utilizator înregistrat are dreptul de a cumpăra produse. Utilizatorii cu rol de administrator, pe lângă permisiunile pe care le are un utilizator obișnuit, are dreptul

de a adăuga noi produse, de a vizualiza comenzile plasate de utilizatori, cât și rolul de a expedia produsele comandate.

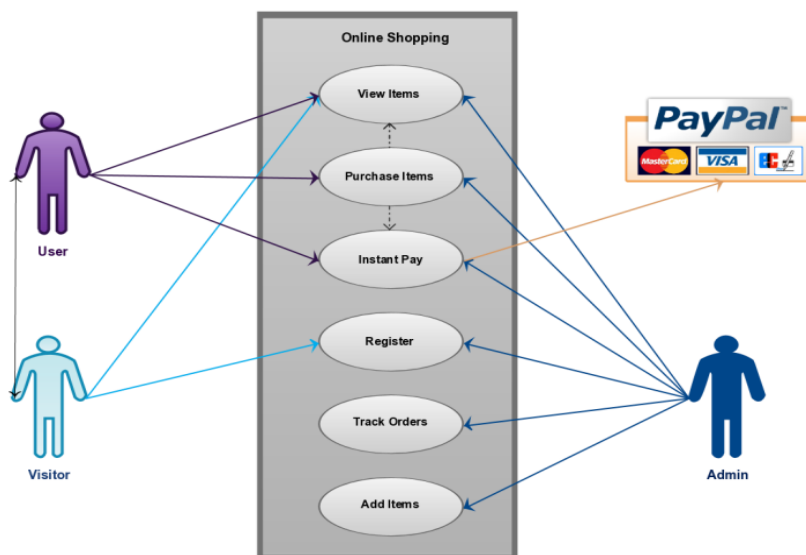


Figura 2.2. Diagrama generală de utilizare [10]

Baza de date SQL este structurată în trei tabele (vezi Figura 2.3), prima tabelă este destinată utilizatorilor, acesta conține datele necesare pentru autentificare, cât și rolul acestuia. A doua tabelă este utilizată pentru datele produselor, iar ultima tabelă pentru datele legate de comenzile plasate.

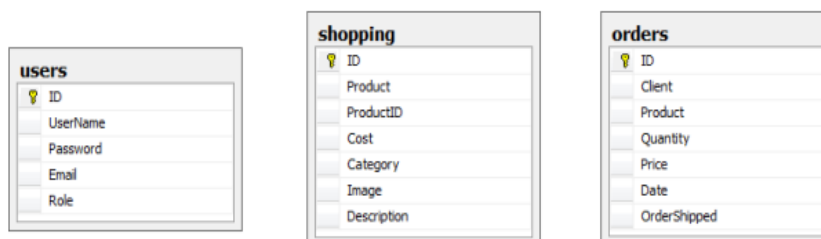


Figura 2.3. Structura bazei de date [10]

În lucrarea [11] este prezentată o aplicație ce comercializează o gamă largă de dispozitive electronice, precum laptop-uri, camere, televizoare și multe altele (vezi Figura 2.4). În realizarea lucrării [11] au fost utilizate următoarele tehnologii: editorul *Bracket*, mediul de dezvoltare *XAMPP* și programul *PhpMyAdmin* pentru gestionarea sistemului *MySQL*. Pentru dezvoltarea aplicației au fost utilizate limbajele *HTML*, *CSS*, *JavaScript* pentru partea de front-end, iar *PHP* pentru partea de back-end [11].

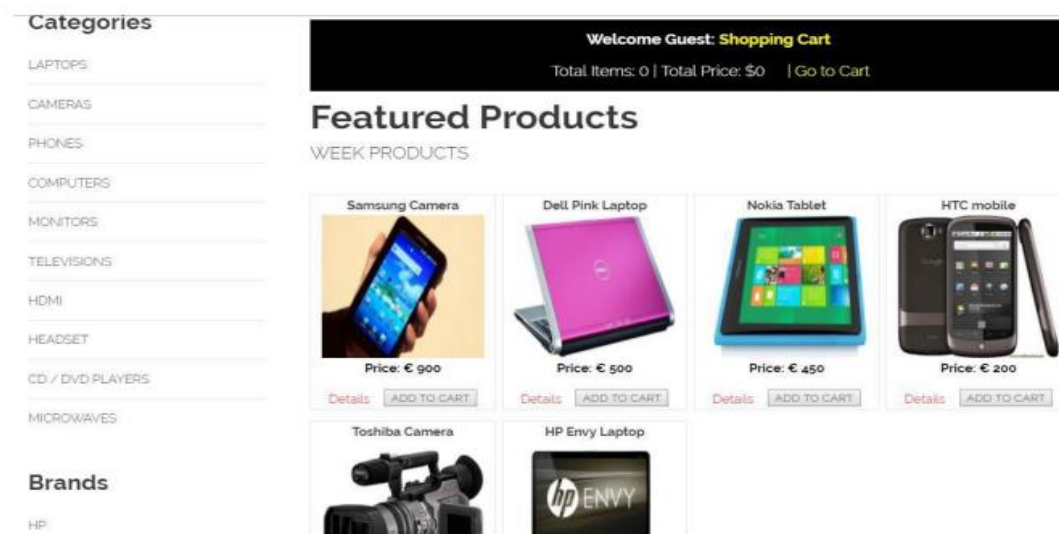


Figura 2.4. Interfața aplicației [11]

Aplicația a fost proiectată pentru a avea o vizualizare diferită pentru administratori față de vizualizarea publică utilizată de clienți sau de vizitatori. După cum se poate observa în Figura 2.5, utilizatorul cu rol de administrator are posibilitatea de administra produsele (actualizarea, eliminarea și adăugarea produselor) și de a gestiona clienții, pe când utilizatorii obișnuiți vor putea doar să-și gestioneze atât datele de contact, cât și produsele comandate. În urma plasării unei comenzi, clientul va fi notificat pe e-mail în legătură cu statusul comenzii.

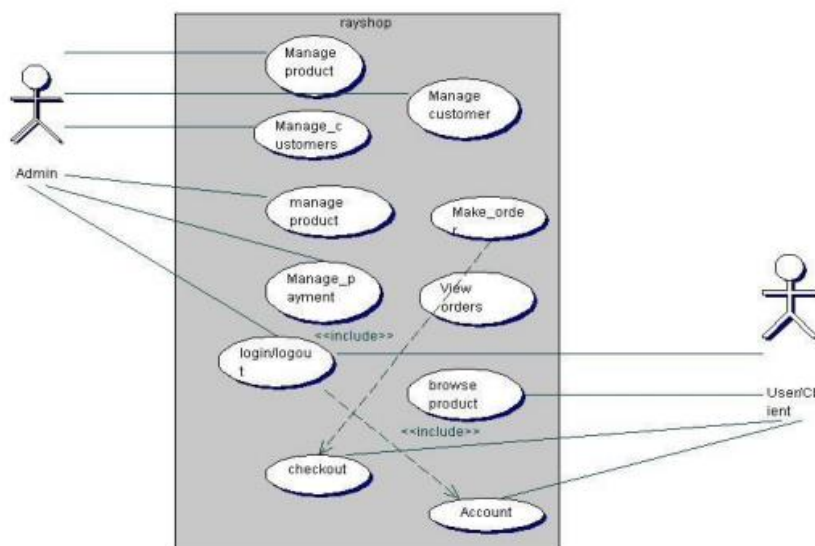


Figura 2.5. Diagrama generală de utilizare [11]

Structura tabelor din baza de date este prezentată în Figura 2.6 care dispune de un număr de opt tabele. Astfel, tabela *admins* are rolul de a stoca datele personale ale administratorilor, pe când în tabela *customers* sunt stocate datele clienților. Pe lângă acestea mai sunt prezentate tabelele corespunzătoare produselor (*products*), brand-urilor

(*brands*), categoriilor (*categories*), comenzilor plasate (*orders*), coșului de cumpărături (*cart*), cât și plăților efectuate (*payments*),

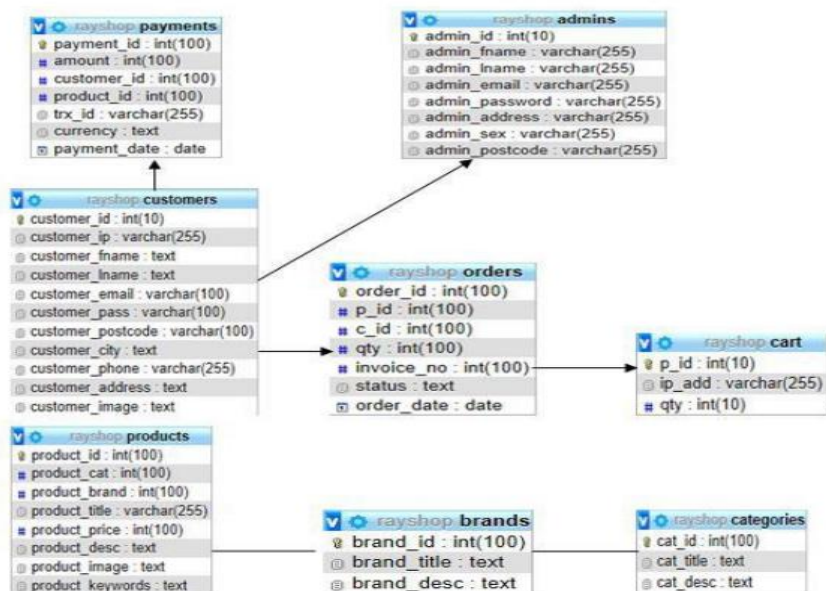


Figura 2.6. Structura tabelor bazei de date [11]

Pe lângă produsele de care dispun proiectele aferente lucrărilor [10] și [11], precum articole vestimentare și dispozitive electronice, în lucrarea [12] mai este adăugată o categorie destinată cărților (vezi Figura 2.7). În dezvoltarea aplicației au fost utilizate limbajele *HTML*, *CSS* și *JavaScript* pentru partea de *front-end*, iar *PHP* pentru partea de *back-end*. Pentru stocarea datelor a fost utilizată o bază de date *MySQL*.



Figura 2.7. Interfața aplicației [12]

În cadrul *lucrării* [12], utilizatorii dispun de trei roluri, precum cel de vizitator (utilizatorii neînregistrați), utilizator (înregistrați) și administrator. În *Figura 2.8* este prezentată o diagramă generală de utilizare a aplicației. Astfel, toți utilizatorii au posibilitatea de a vedea produsele disponibile, de a se înregistra și de a căuta produse. Utilizatorii înregistrați au dreptul de a adăuga produse pe care ulterior să le comande. Pe lângă toate facilitățile de care dispun utilizatorii, atât cei înregistrați, cât și cei neînregistrați, utilizatorul cu rol de administrator are permisiunea de a adăuga și edita produse.

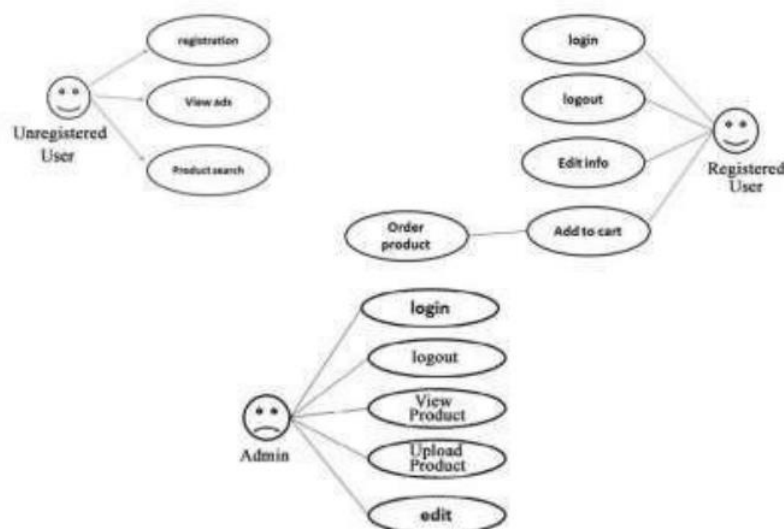


Figura 2.8. Diagrama de utilizare [12]

Baza de date este una relațională și este structurată în trei tabele (vezi *Figura 2.9*). Prima tabelă este destinată utilizatorilor, ce conține datele necesare acestora pentru etapa de autentificare, a doua tabelă este construită pentru reținerea datelor despre un produs adăugat de către administrator și tabela pentru categoriile produselor.

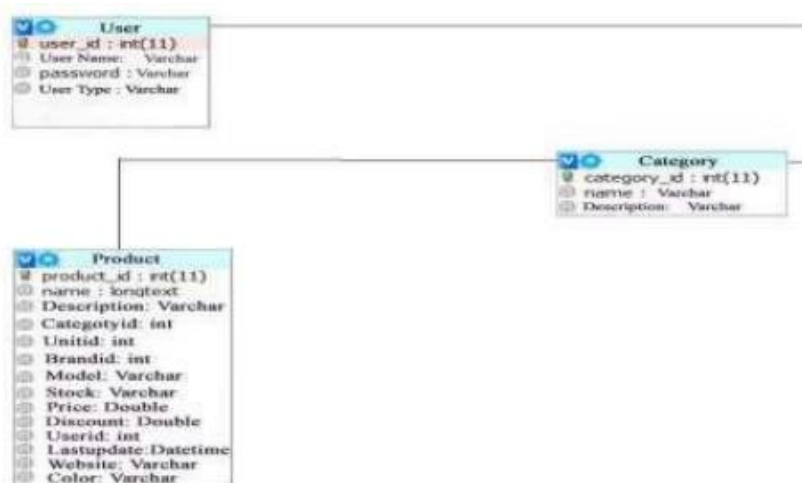


Figura 2.9. Structura bazei de date [12]

2.2. CONCLUZII

În urma studiului lucrării [10], [11] și [12] am realizat o scurtă comparație bazată pe funcționalități, între lucrările menționate și proiectul realizat de mine.

După cum se poate observa, în prima coloană a *Tabelului 2.2* sunt menționate câteva caracteristici importante ce nu ar trebui să lipsească din cadrul proiectului. În următoarele coloane sunt marcate câmpurile cu o bifă în cazul în care lucrarea aferentă coloanei respective deține caracteristica precizată, iar cu un „X” în caz contrar.

Tabelul 2.2. Comparație caracteristici lucrări

Caracteristici	Lucrarea [10]	Lucrarea [11]	Lucrarea [12]	Proiectul de licență
Autentificare utilizând contul Google	x	x	x	✓
Posibilitatea de a vizualiza produsele chiar dacă utilizatorul nu este conectat	✓	✓	✓	x
Pagină dedicată administratorilor	✓	✓	x	x
Criptare parolă	x	x	x	✓
Resetare parolă	x	x	✓	✓
Funcție de căutare	x	x	✓	✓
Hosted (găzduit)	x	x	x	x
Plată cu cardul	✓	x	x	x

3. BAZELE TEORETICE

În acest capitol voi descrie tehnologiile utilizate și modul în care mi-au fost de ajutor în dezvoltarea proiectului.

3.1. VISUAL STUDIO CODE

Visual Studio Code este un editor de cod sursă, realizat de către *Microsoft* cu suport pentru operațiuni de dezvoltare precum depanarea, utilizarea comenzilor rapide, completarea inteligentă a codului și multe altele. Acest editor poate fi utilizat cu o varietate de limbaje de programare, precum *Java*, *JavaScript*, *Go*, *Node.js*, *C++*, *C*, *Python*, *Rust* și *Fortran* [13].

Cele mai importante facilități oferite de către *Visual Studio Code* sunt extensiile și funcționalitățile *GIT* integrate.

3.1.1. EXTENSII

Editorul dispune de o gamă largă de extensii care pot fi instalate direct din mediul de dezvoltare *Visual Studio Code*. Cele mai utilizate extensii în realizarea proiectului sunt: *Live Server*, *Font Awesome Gallery*, *Auto Close Tag* și *IntelliCode*.

După cum se poate observa în *Figura 3.1*, cu ajutorul extensiei *Live Server* putem lansa un server local de dezvoltare cu caracteristică live, atât pentru pagini statice, cât și dinamice [14].

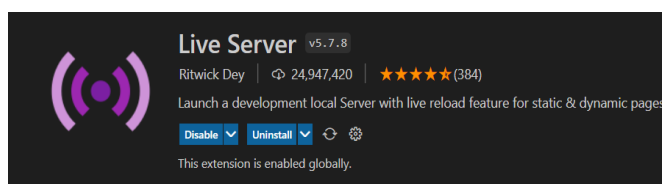


Figura 3.1. Live Server

Cu ajutorul extensiei din *Figura 3.2* putem căuta pictogramele dorite, fie după nume, fie după un cuvânt cheie, ulterior putând fi copiat codul pictogramei doar printr-un click [15]. Pe lângă instalarea extensiei, pentru a utiliza pictogramele dorite, trebuie inclusă biblioteca corespunzătoare (vezi *Figura 3.3*).

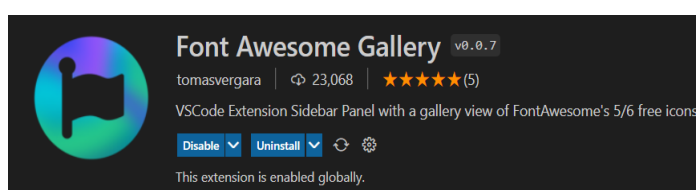


Figura 3.2. Font Awesome Gallery

```
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.15.4/css/all.css"/>
```

Figura 3.3. Bibliotecă pentru pictograme

Prin instalarea extensiei prezentate în *Figura 3.4*, dispunem de închiderea automată a etichetelor. Astfel, pot fi prevenite eventualele erori din cauza neînchiderii unei etichete [16].

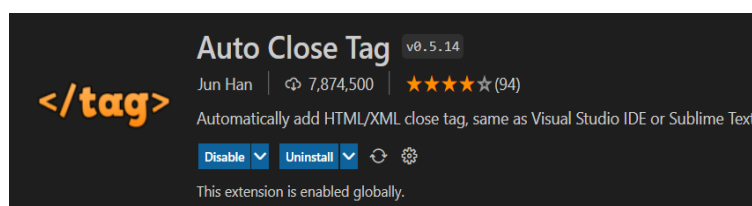


Figura 3.4. Auto Close Tag

Extensia *IntelliCode* (vezi *Figura 3.5*.) oferă caracteristici de dezvoltare asistate de AI (Artificial Intelligence) pentru dezvoltatorii *Python*, *TypeScript/JavaScript* și *Java*, cu informații bazate pe înțelegerea contextului codului, combinată cu învățarea automată [17].

Această extensie oferă *IntelliSense* asistat de AI, oferind completări inteligente de cod bazate pe semantica limbajului și pe o analiză a codului sursă. Datorită extensiei *IntelliCode* lista de sugestii nu mai necesită parcurgerea în ordine alfabetică, deoarece este afișată în funcție de cea mai relevantă sugestie de completare [17].

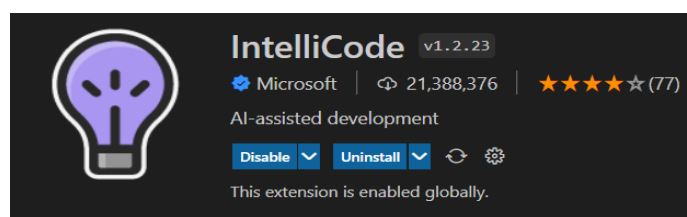


Figura 3.5. IntelliCode

3.1.2. GIT

Git este un sistem de versionare multi-platformă, ce aduce multiple beneficii la nivelul unui proiect, spre exemplu: istoric al modificărilor, posibilitatea de a se întoarce la un anumit moment în cadrul proiectului, posibilitatea de a crea diverse alte căi cu propriul lor istoric denumite *branch-uri* și multe altele [18].

Din punct de vedere al funcționării, *Git* are o zonă locală prezentă pe mașina pe care se află proiectul, și una *remote* (la distanță), care este de regulă disponibilă pentru mai mulți utilizatori și reprezintă zona unde se concentrează toate funcționalitățile unui anumit proiect (vezi *Figura 3.6*) [18].

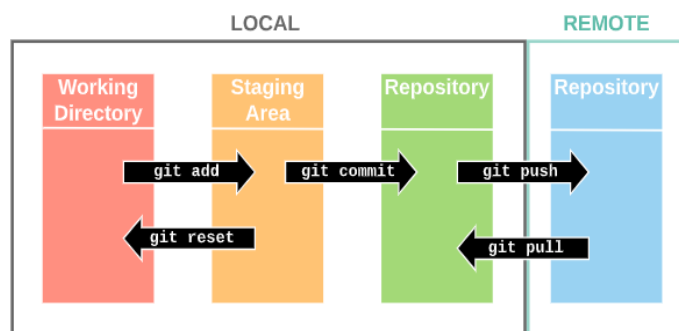


Figura 3.6. Exemplu de funcționare a sistemului de versionare Git [18]

Aceste zone *remote*, pot fi diverși clienți precum *Github*, *GitLab*, *BitBucket* etc.. Dintre toate acestea, proiectul a fost dezvoltat utilizând *GitHub*, fiind un serviciu de găzduire de proiecte folosind ca sistem de versionare *Git*. Acesta este un serviciu gratuit pentru aplicații *open-source* și a fost achiziționat în 2018 de către *Microsoft* [18].

3.2. HTML

HTML reprezintă o abreviere de la *HyperText Markup Language* fiind un limbaj de marcare utilizat pentru crearea paginilor web de Internet [19]. Paginile *HTML* sunt formate din etichete sau tag-uri, având extensia „.html” sau „.htm”. Majoritatea etichetelor sunt pereche, una de deschidere `<eticheta>` și alta de închidere `</eticheta>`.

De-al lungul anilor, *HTML* a dispus de numeroase versiuni, ultima și cea mai apreciată fiind *HTML5*. Sintaxa *HTML5* este mult mai ușoară în comparație cu *HTML4* și oferă multe caracteristici noi (vezi *Tabelul 2.*).

Tabelul 2. Versiuni HTML [20]

	HTML4	HTML5
Declarație DOCTYPE	Sintaxă lungă, cu referințe externe	Sintaxă simplificată într-o linie
Etichete de suport multimedia	Nu există	Există
Utilizare cookie-uri	Da	Nu. Se axează pe stocarea locală în locul cookie-urilor
Desenare forme	Nu este posibil	Este posibil
Rulare JavaScript în fundal	Nu permite	Permite

3.3. CSS

CSS (Cascading Style Sheets) este un limbaj utilizat pe scară largă de un număr mare de dezvoltatori de software pentru stilizarea paginilor web [21]. În timp ce *HTML* folosește etichete, *CSS* folosește seturi de reguli [22].

Definițiile de stil sunt în mod normal salvate în fișiere externe cu extensia *.css* (vezi *Figura 3.7*). Aceasta este cea mai eficientă metodă de a aduce regulile scrise într-un document. *CSS* economisește timp datorită posibilității de a reutiliza același fișier de reguli de stil în mai multe pagini web [22].

```
h1 {  
    color: blue;  
    background-color: yellow;  
    border: 1px solid black;  
}  
  
p {  
    color: red;  
}
```

Figura 3.7. Foaie de stil externă

Pe lângă foile externe, mai există posibilitatea de a stabili reguli de stil și în interiorul unui document *HTML*, plasând regulile într-o etichetă *<style>*, în interiorul etichetei *<head>* (vezi *Figura 3.8*) [23].

```
<head>  
  <meta charset="utf-8">  
  <title>My CSS experiment</title>  
  <style>  
    h1 {  
      color: blue;  
      background-color: yellow;  
      border: 1px solid black;  
    }  
  
    p {  
      color: red;  
    }  
  </style>  
</head>
```

Figura 3.8. Foaie de stil internă

Ultima variantă de utilizarea a regulilor, o reprezintă implementarea unui stil inline în interiorul unui document *HTML* (vezi *Figura 3.9*). Aceasta este cea mai puțin eficientă metodă, deoarece o schimbare de stil poate necesita mai multe modificări într-o pagină web [23].

```
<body>  
  <h1 style="color: blue;background-color: yellow;border: 1px  
solid black;">Hello World!</h1>  
  <p style="color:red;">This is my first CSS example</p>  
</body>
```

Figura 3.9. Stil inline

3.4. JAVASCRIPT

JavaScript, deseori denumit și *JS*, este un limbaj de programare compilat și interpretat, fiind cunoscut ca un limbaj de *scripting* pentru pagini web. Acest limbaj poate fi folosit atât pentru dezvoltări la nivelul clientului, cât și la nivelul serverului [24].

Partea de client furnizează obiecte pentru controlul browser-ului și al modelului de obiecte document (*DOM*), iar partea de server furnizează obiecte relevante pentru rularea *JS* pe un server.

Codul *JavaScript* poate fi adăugat în două moduri în fișierele *HTML*. Fie intern, adăugând o etichetă `<script>` plasată ori în interiorul etichetei `<head>`, ori în interiorul etichetei `<body>`, fie extern într-un fișier cu extensia *.js*, ulterior plasăm fișierul în interiorul etichetei `<head>` a fișierului *HTML*.

Vanilla JavaScript este un nume care se referă la utilizarea *JavaScript* fără a utiliza biblioteci suplimentare, precum *jQuery*. Acest limbaj este foarte elementar, ușor de învățat și utilizat, putând crea proiecte semnificative, precum site-uri web.

Multe site-uri cunoscute utilizează în prezent *Vanilla JavaScript*, precum: *Facebook*, *Google*, *Youtube*, *Microsoft*, *Netflix*, *Twitter*, *Wikipedia* și multe altele.

JavaScript dispune de o gamă largă de biblioteci care sunt folosite pentru a îndeplini anumite funcții.

3.4.1. BIBLIOTECI

Una dintre cele mai utilizate biblioteci din cadrul proiectului a fost *smtp.js* (vezi *Figura 3.10.*). Folosind această bibliotecă, se poate trimite mail direct, utilizând *JavaScript* la nivelul clientului, fără configurații la nivel de server. *SMTP* (Simple Mail Transfer Protocol) este un protocol folosit pentru a trimite date către serverul destinat, urmat de destinatar [25].

```
<!-- Simple Mail Transfer Protocol -->  
<script src="https://smtpjs.com/v3/smtp.js"></script>
```

Figura 3.10. Biblioteca *smtp.js*

Pentru criptarea acreditărilor am creat un server *smtp* pe *elasticmail.com* (vezi *Figura 3.11*), deoarece *smtp* permite doar serverului *smtp elasticmail* să creeze *acreditările*.

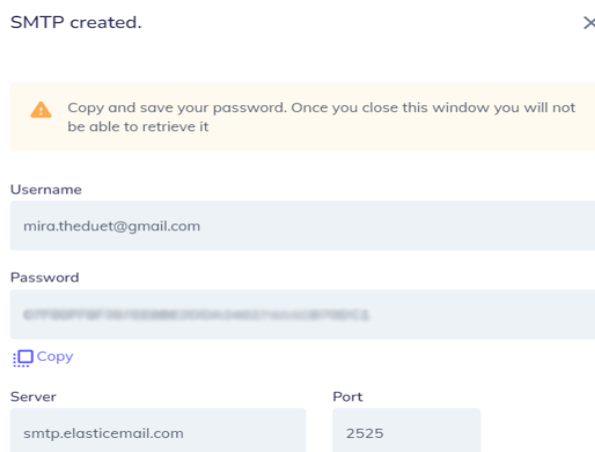


Figura 3.11. Creare server smtp

Pentru a nu fi vizibile întregii lumi, *acreditărilor* le-am criptat transmițând un *token* securizat în locul acestora. *Token-ul* a fost generat direct pe site-ul oficial *smtpjs.com* completând datele necesare (vezi *Figura 3.12*).

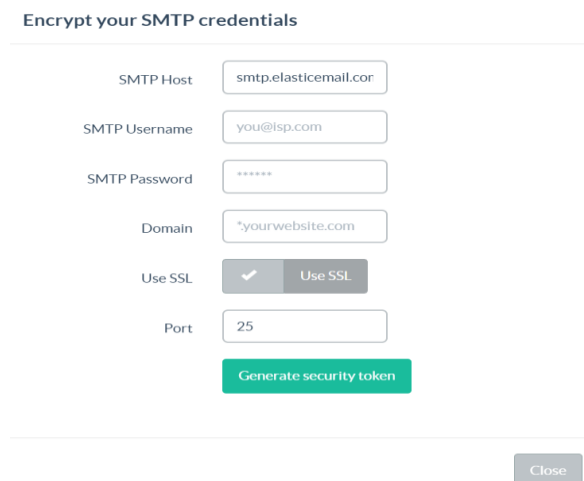


Figura 3.12. Criptarea acreditărilor

Pe lângă biblioteca *smtp.js*, am folosit *jsPDF* (vezi *Figura 3.11*), care este o bibliotecă *open-source* pentru a genera documente *PDF* utilizând limbajul *JavaScript* [26].

```
<!-- biblioteca jsPDF -->  
<script src="https://unpkg.com/jspdf@latest/dist/jspdf.umd.min.js"></script>
```

Figura 3.11. Biblioteca jsPDF

3.5. FIREBASE

Firebase este o platformă de dezvoltare de aplicații, care vă ajută să vă construiți, îmbunătățiți și dezvoltați aplicații web și mobile.

Cea mai importantă caracteristică o reprezintă baza de date în timp real (*Firebase Realtime Database*), pe care am utilizat-o în dezvoltarea aplicației pentru stocarea datelor ce are loc în fișiere *JSON*. Această bază de date este una non-relațională (*NoSQL*), datele fiind sincronizate între toți clienții în timp real și disponibile chiar dacă aplicația este *offline* [26]. Baza de date în timp real oferă un limbaj de reguli bazate pe expresii, sub denumirea de *Firebase Realtime Database Security Rules*, ce este utilizată pentru definirea modului în care ar trebui să fie structurate datele, dar și când vor putea fi citite sau scrise [28]. Aceste reguli sunt aplicate automat în orice moment, iar pentru securitate, regulile nu trebuie să permită accesul nimănui la baza de date [28].

JSON (JavaScript Object Notation) este un format ușor de interschimbare a datelor, precum și un format de reprezentare a acestora, fiind ușor de citit și scris pentru oameni [28]. *JSON* este construit pe două structuri: o colecție de perechi de nume/valoare (un obiect, un dicționar, o tabelă *hash*, o listă de chei etc.) și o listă ordonată de valori (tablou, vector, listă sau șir) [29]. Acest format este o alternativă pentru *XML* (Extensible Markup Language) fiind mai compact și neavând nevoie de biblioteci externe pentru manipulare (vezi *Figura 3.12*).

XML	JSON
<pre><Servers> <Server> <name>Server1</name> <owner>John</owner> <created>123456</created> <status>active</status> </Server> </Servers></pre>	<pre>{ Servers: [{ name: Server1, owner: John, created: 123456, status: active }] }</pre>

Figura 3.12. Structura XML vs JSON [30]

Majoritatea aplicațiilor trebuie să cunoască identitatea utilizatorului, pentru a putea salva în siguranță datele acestuia în *cloud* și să ofere aceeași experiență personalizată pe toate dispozitivele utilizatorului. Astfel, o altă caracteristică utilizată de dezvoltatori este *Firebase Authentication*, ce permite utilizatorilor autentificarea anonimă, cu parolă sau cu diferite rețele de socializare [31].

În urma interacțiunii cu platforma *Firebase* pot afirma că este simplă și ușor de utilizat, fiindcă dispune de numeroase caracteristici fără a avea nevoie de o configurare complicată.

4. SOLUȚIA PROPUȘĂ ȘI METODOLOGIA DE PROIECTARE

4.1. ARHITECTURA GENERALĂ A SISTEMULUI

Site-ul dispune de existența a două tipuri de utilizatori cu atribuții diferite. Un utilizator obișnuit care poate vizualiza sau comanda produse și un utilizator cu rol de administrator. Administratorul pe lângă atribuțiile ale unui utilizator obișnuit mai are permisiunea de a edita, șterge sau adăuga un produs, dar și de a vizualiza comenzile plasate de către clienți, pe care le poate anula sau livra.

Partea de client sau *UI* (User Interface), reprezintă interfața grafică, adică tot ceea ce utilizatorii experimentează direct. Pe când partea de server funcționează ca un intermediar între sistem, bază de date și client, având rolul de a interoga baza de date ca mai apoi să proceseze datele și să le transmită către utilizator (vezi *Figura 4.1*).

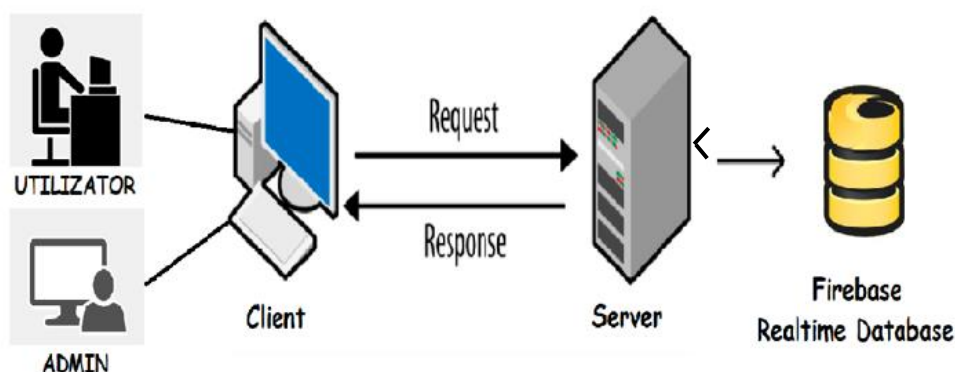


Figura 4.1. Diagrama arhitecturii generale

4.2. DIAGrame UML

UML (Unified Modeling Language) este un limbaj de modelare unificat ce conține diferite diagrame pentru modelarea obiectelor din lumea reală [32]. Pentru descrierea perspectivelor individuale ale unui sistem software sunt utilizate diferite tipuri de diagrame *UML* [32]. Ele sunt împărțite în două categorii, diagrame de structură (diagrame de clase, obiecte etc.) și diagrame comportamentale (diagrame de utilizare, activitate etc.) [33].

Mai jos o să descriu două dintre cele mai cunoscute diagrame *UML*, mai exact diagramele de utilizare și cele de activitate.

4.2.1. DIAGrame DE UTILIZARE

În cazul proiectului meu, am împărțit diagrama de utilizare în două module. Astfel, în primul modul o să ilustrez cazul autentificării utilizatorilor. După cum se poate observa în *Figura 4.2* sunt prezenți doi actori, unul cu rol de utilizator, iar celălalt cu rol de administrator.

În acest caz, actorii au aceleași drepturi. Mai concret, ambii actori au posibilitatea de a se înregistra, autentifica sau schimba parola contului.

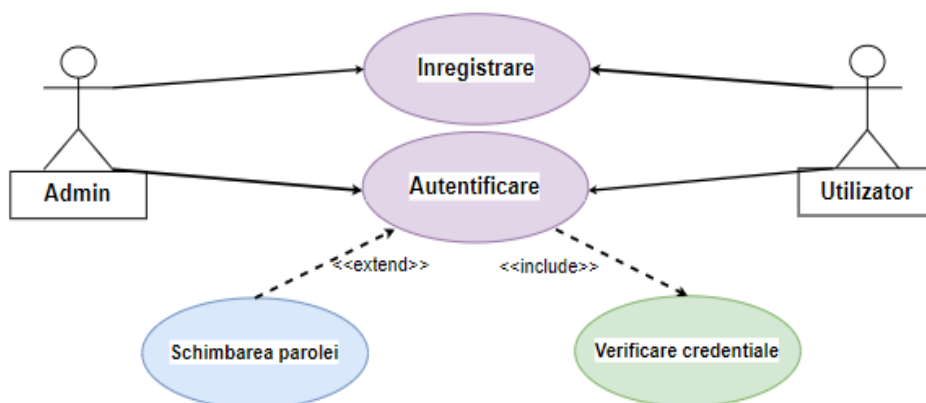


Figura 4.2. Diagrama de utilizare pentru Modulul 1

În cadrul celui de-al doilea modul o să ilustrez diagrama pentru conținutul site-ului de comerț electronic. În Figura 4.3 sunt ilustrate toate atribuțiile de care dispune doar utilizatorul cu rol de administrator. Acesta are dreptul de a adăuga, edita sau șterge produse, dar și de a administra comenzile plasate de utilizatori, având permisiunea de a anula sau livra o comandă.

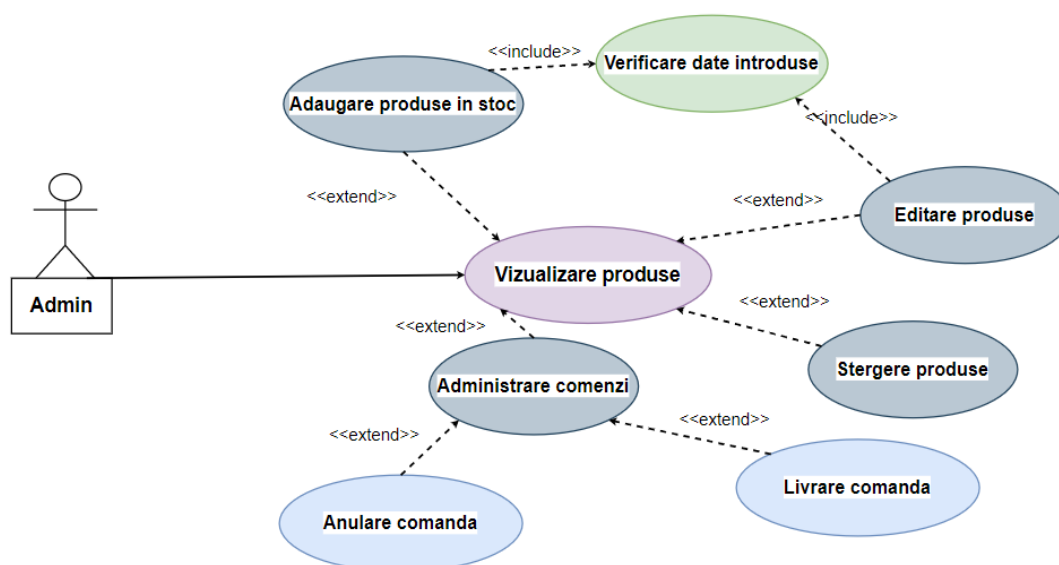


Figura 4.3. Diagrama de utilizare pentru Modulul 2
(administrator)

În *Figura 4.4* sunt ilustrate atribuțiile comune ale utilizatorilor și administratorilor. Astfel, ambii actori au posibilitatea de a vizualiza, filtra, căuta produse și a le adăuga la favorite și în coșul de cumpărături. Pe lângă aceste funcționalități, utilizatorii pot plasa o comandă.

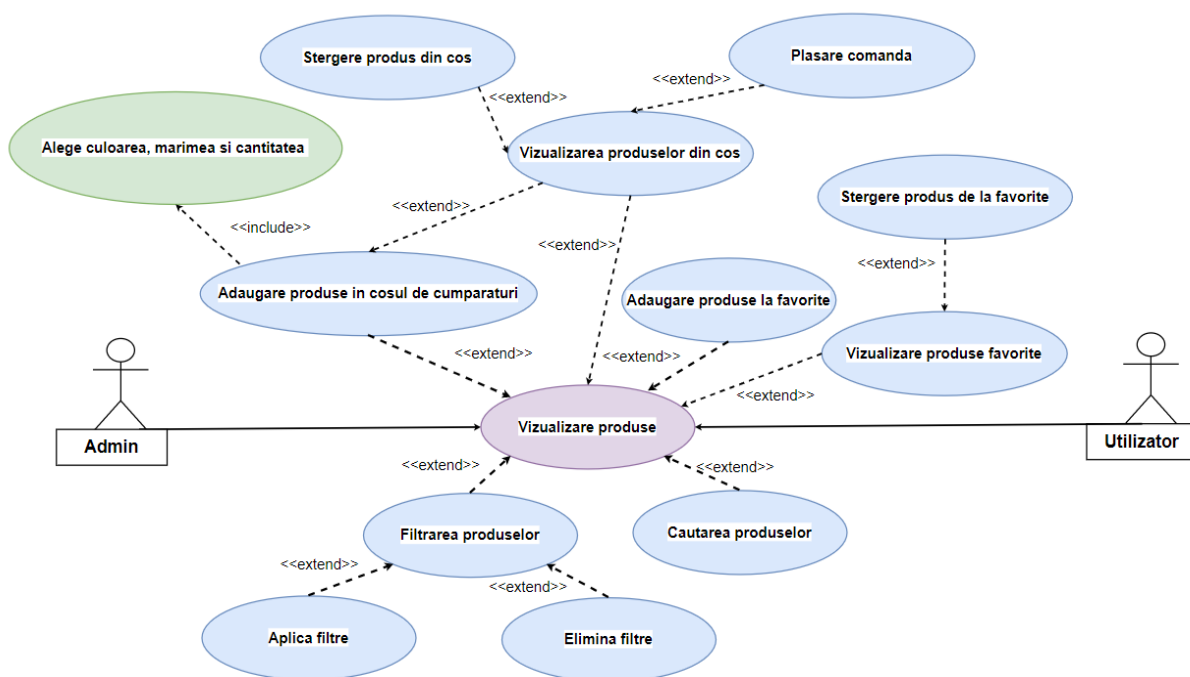


Figura 4.4. Diagrama de utilizare pentru Modulul 2 (administrator și utilizator)

Diagrama cazurilor de utilizare este una dintre metodele care contribuie la proiectarea și dezvoltarea site-ului de comerț electronic. Această diagramă mi-a fost folositoare pentru a cunoaște posibilele scenarii pe care ar trebui să le realizez în cadrul proiectului.

4.2.2. DIAGrame DE ACTIVITATE

În *Figurile 4.5-4.7* vor fi dezvoltate funcționalitățile prezentate în diagramele de utilizare.

Astfel, în *Figura 4.5* este prezentată diagrama aferentă primului modul în care au fost dezvoltate funcționalitățile pentru cazul de autentificare al utilizatorilor. Prima pagină afișată acestora este cea de autentificare, în care utilizatorul dispune de două opțiuni de autentificare, fie cu un cont creat pe site, fie cu contul de *Google*. Prin alegerea opțiunii de autentificare cu contul de *Google*, utilizatorul trebuie doar să aleagă contul dorit. În urma acestui proces se creează un nou cont de utilizator cu acreditările preluate pe care le stochează în baza de date *Firebase*, putând fi folosit pentru a identifica utilizatorul în paginile web ale proiectului.

O altă metodă de autentificare este utilizarea unui cont creat pe site din pagina de *înregistrare*. Pentru a trece la pagina de *produse*, acreditările trebuie să fie valide, la fel și în cazul înregistrării. Astfel, înainte de a se crea contul vor fi verificate câmpurile, iar în urma validării acestora se va crea un nou cont și se va adăuga în baza de date.

De asemenea, există posibilitatea resetării parolei în cazul în care ați uitat sau doriți să o modificați. După solicitarea resetării parolei, va fi trimis pe e-mail un link, iar în urma accesării acestuia va trebui să adăugați noua parolă dorită. După validarea noii parole, aceasta va fi modificată și în baza de date.

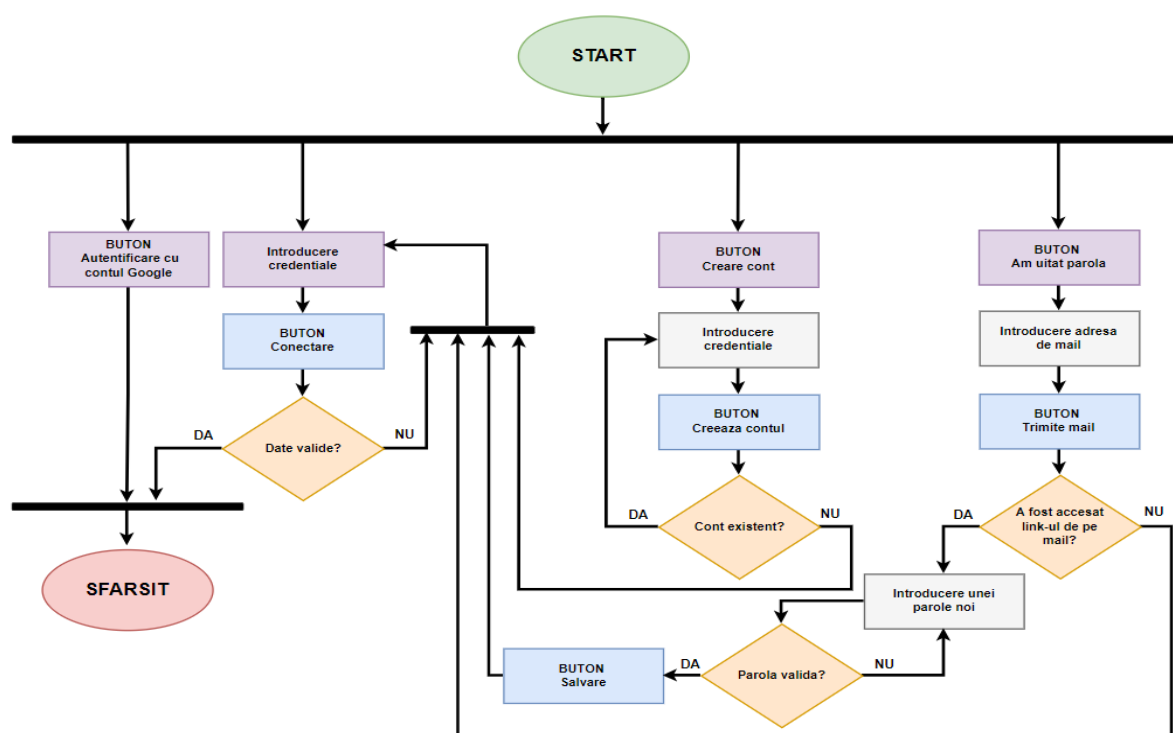


Figura 4.5. Diagrama de utilizare pentru Modulul 1

În Figura 4.6. este prezentată diagrama de activitate aferentă celui de-al doilea modul în care au fost dezvoltate doar atribuțiile de care dispun numai utilizatorii cu rol de administrator. Astfel, aceștia au permisiunea de a adăuga un nou produs în stoc, în urma validării datelor introduse acesta va fi adăugat în baza de date și va fi afișat în pagina de produse. De asemenea, administratorul mai are permisiunea de a șterge definitiv un produs din baza de date și de a edita unul existent. Acesta are posibilitatea de a vizualiza toate comenzile plasate de utilizatori și va putea să le anuleze sau să le trimită către clienți, care vor fi notificați în funcție de alegerea opțiunii.

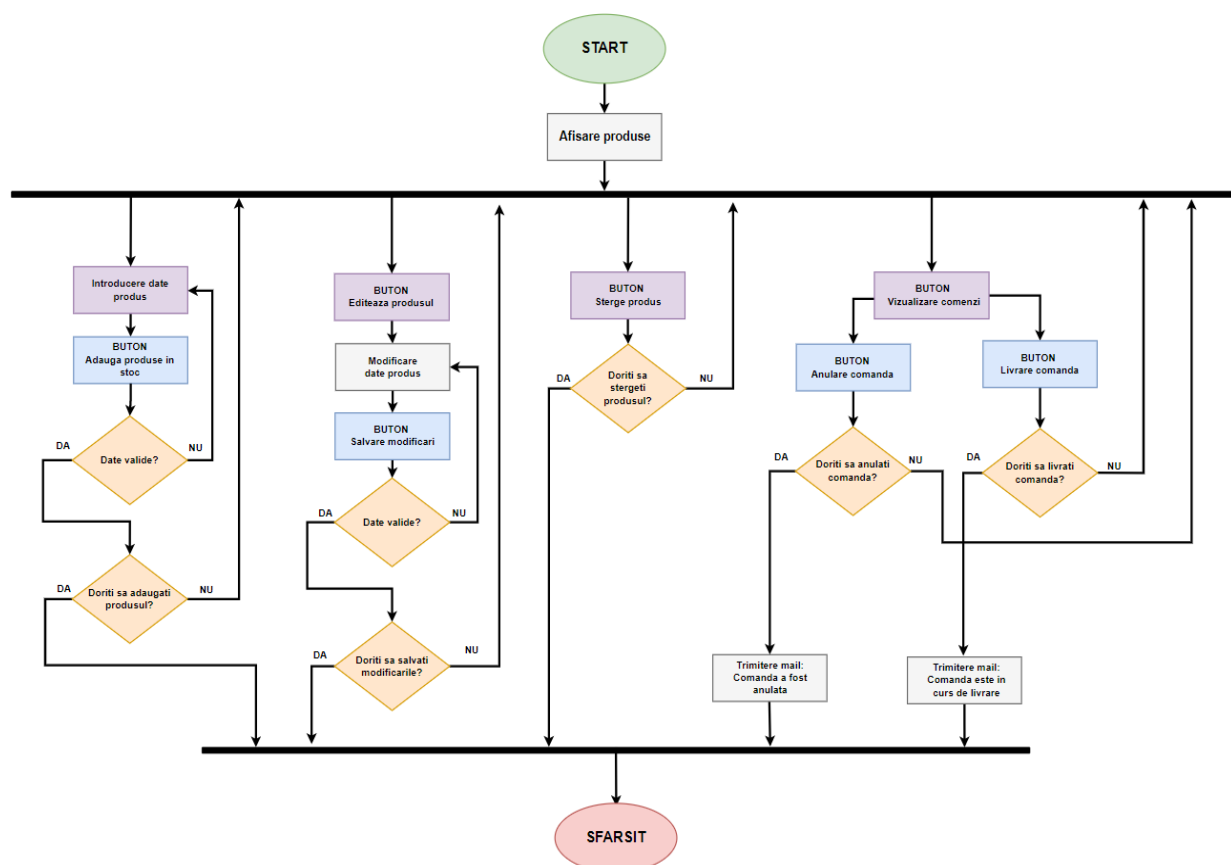


Figura 4.6. Diagrama de activitate aferentă Modulului 2 (administrator)

În Figura 4.7 este ilustrată diagrama aferentă celui de-al doilea modul în care au fost dezvoltate atribuțiile comune ale utilizatorilor obișnuiți, dar și a celor cu rol de administrator. După cum se poate observa în prima parte a diagramei, ambii utilizatori pot adăuga un produs în coșul de cumpărături sau la favorite, ulterior având posibilitatea de a vedea lista de produse adăugate.

Din pagina de vizualizare, fie a produselor favorite, fie a coșului de cumpărături, utilizatorii pot șterge produsul nedorit, acesta fiind eliminat și din baza de date. După adăugarea unor produse în coșul de cumpărături se poate plasa o comandă, care va putea fi finalizată abia după ce utilizatorul va introduce date valide necesare pentru facturare. Clientul, pe lângă primirea unui mail de înștiințare că a fost plasată comanda, va avea atașată și factura, chiar dacă are opțiunea de a o descărca direct de pe site imediat după finalizarea comenzii.

La dispoziția utilizatorilor se regăsesc opțiuni de filtrare a produselor, cât și o căutare după numele produsului. În cazul în care alegerile făcute în filtrare nu se regăsesc printre produsele existente, va fi afișat un mesaj pentru a înștiința utilizatorul. Pentru a ușura munca utilizatorilor am pus la dispoziția acestora un buton pentru a elimina filtrele, astfel se vor afișa toate produsele, nu doar cele afișate după criteriile alese.

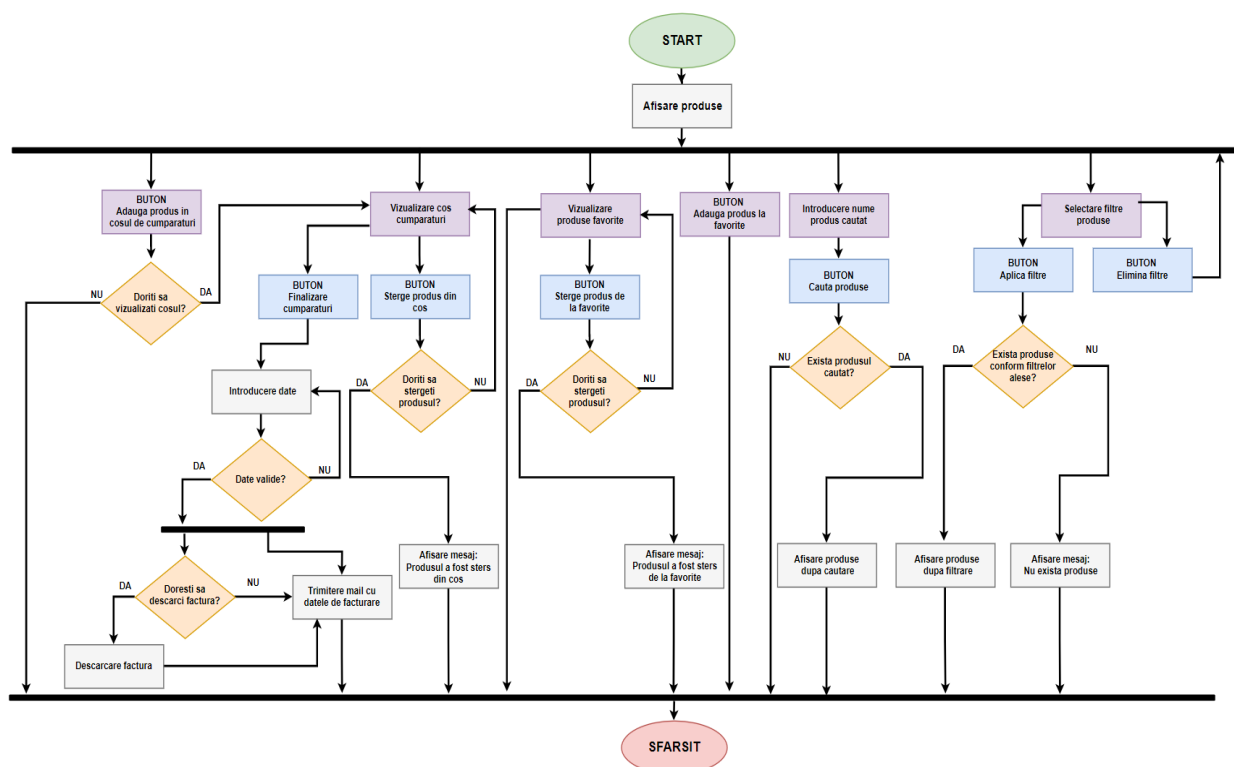


Figura 4.7. Diagrama de activitate aferentă Modulului 2 (administrator și utilizator)

4.3. STRUCTURA BAZEI DE DATE

Construirea unei baze de date necesită multă gândire pentru ca datele să fie bine structurate. Toate datele din *Firebase Realtime Database* sunt stocate ca obiecte *JSON*. Spre deosebire de o bază de date *SQL*, nu există tabele sau înregistrări. Când se realizează adăugarea unor date în arborele *JSON*, acestea se transformă într-un nod în structura *JSON* cu o cheie asociată. Cheile pot fi furnizate fie după preferințe, fie utilizând metoda *push()* care generează automat o cheie nodului.

Baza de date a proiectului cuprinde 4 fișiere mari *JSON*, precum „*Conturi*”, „*Comenzi*”, „*Produse*” și „*Notificări*” (vezi Figura 4.8). Documentul „*Conturi*” are rolul de a stoca detaliile referitoare la utilizatorii înregistrați pe site. Detaliile produselor existente în magazinul online sunt stocate în fișierul „*Produse*”, iar în „*Comenzi*” sunt stocate toate comenzile plasate de utilizatori.

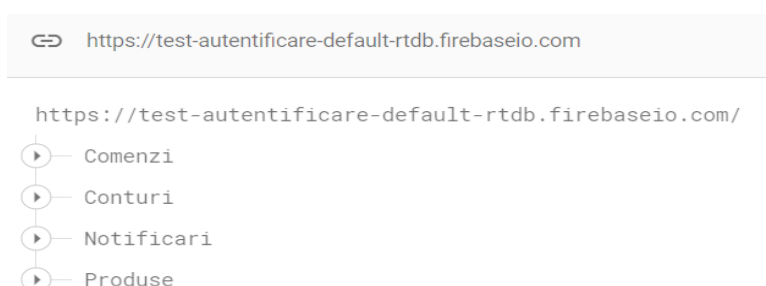


Figura 4.8. Structura fișierelor

Pentru a adăuga clienți în baza de date am utilizat metoda *push()* menționată mai sus, astfel această metodă generează automat o cheie nodului (vezi *Figura 4.9*). În acest fișier pe lângă datele utilizatorului se regăsește și un nod denumit „*CosCumparaturi*” în care se află toate produsele adăugate în coșul de cumpărături de către utilizator. Aceste date fiind preluate din fișierul „*Produse*” în funcție de alegerile făcute de utilizator în momentul adăugării produsului în coșul de cumpărături.

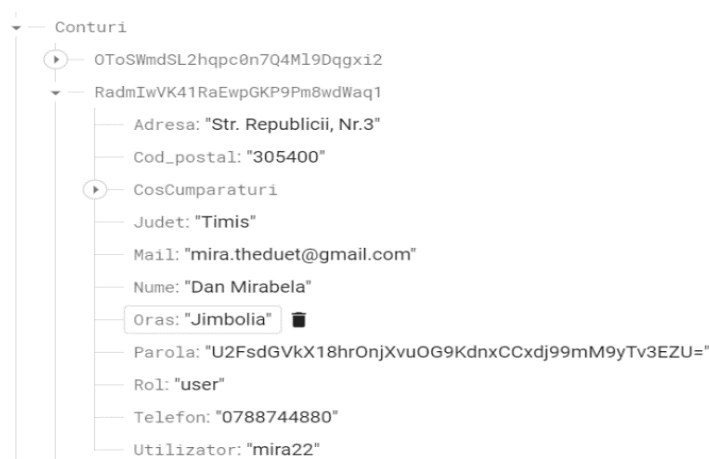


Figura 4.9. Structura fișierului de conturi

La fel ca și în cazul conturilor, adăugarea în baza de date a produselor a fost realizată cu ajutorul metodei *push()* pentru generarea automată a cheilor. Pe lângă datele obișnuite despre produs, precum nume, brand, categorie, poze și prețuri, mai conține un câmp denumit „*NrRecomandare*”, care este utilizat pentru a afișa cele mai apreciate produse în secțiunea de produse recomandate. Nodul „*FavoriteBy*” este o listă care conține id-urile tuturor utilizatorilor care au apreciat respectivul produs, iar câmpul „*NrRecomandare*” este doar o sumă a acestei liste, care se actualizează mereu când are loc o modificare. Nodul „*Detalii*” conține detalii privind materialul produsului, iar nodul „*Culori*” regăsim numărul de produse disponibile în funcție de culoare și mărime.

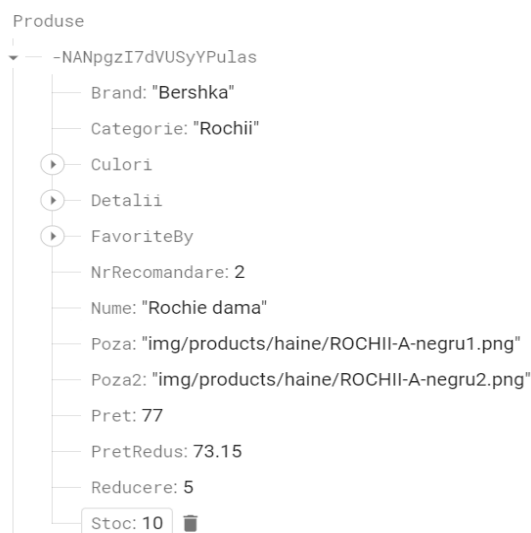


Figura 4.10. Structura fișierului de produse

Fișierul „Comenzi”, la fel cum am menționat mai sus, conține date referitoare la comenzile plasate de către clienți. Astfel în nodul „Client” se regăsesc datele necesare ale clientului pentru facturare (vezi secțiunea 1 din Figura 4.11), aceste date fiind luate din fișierul de conturi ale utilizatorului curent. În nodul „Detalii” vom regăsi date importante referitoare la comanda plasată, date precum *metoda de expediere, statusul plății, metoda de plată (cash sau card), taxa de transport, data și ora* în care a fost plasată comanda (vezi secțiunea 2 din Figura 4.11). Și într-un final ultimul nod denumit „Produse”, care conține o listă cu toate produsele comandate (vezi secțiunea 3 din Figura 4.11).

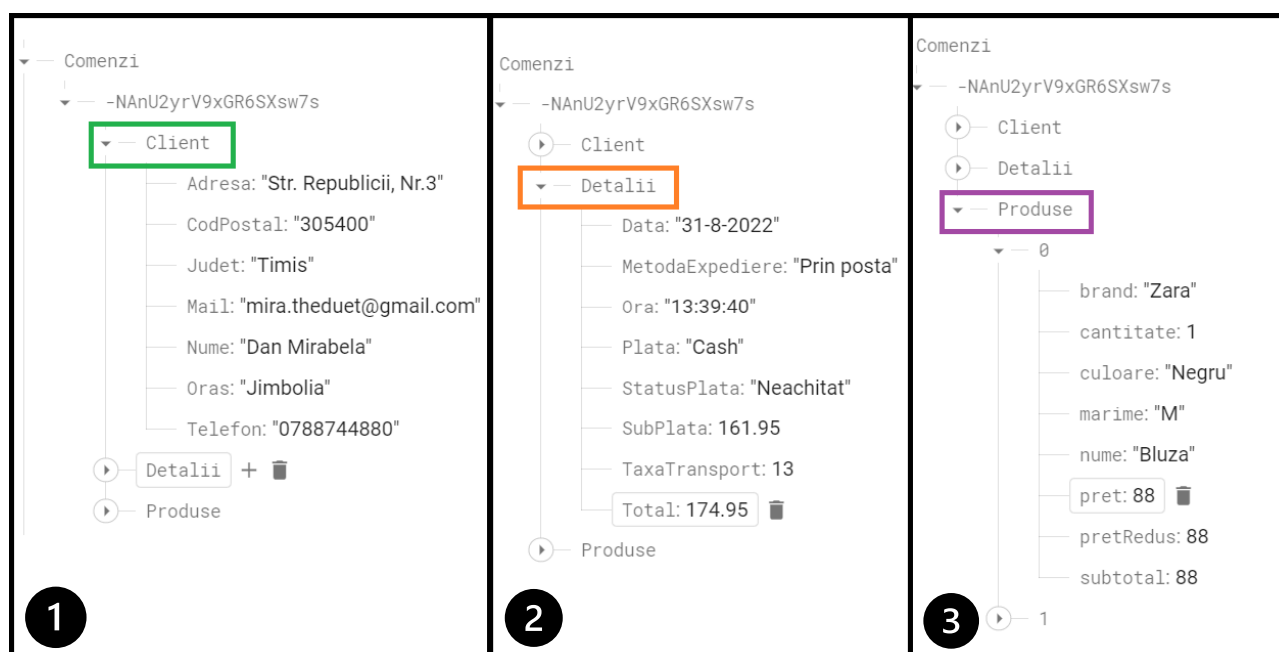


Figura 4.11. Structura fișierului de comenzi

În cadrul fișierului de „Notificări”, datele au fost adăugate prin metoda *push()* pentru generarea automată a cheilor. Acest fișier cuprinde datele necesare din partea unui client care a cerut să fie notificat în cazul în care un produs al cărui stoc este epuizat va deveni disponibil și date despre produsul solicitat (vezi Figura 4.12).

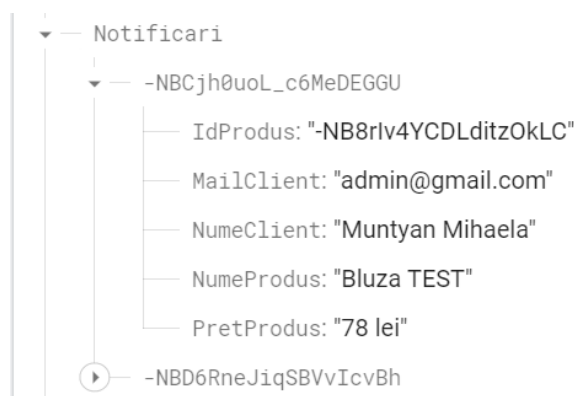


Figura 4.12. Structura fișierului pentru notificări

5. IMPLEMENTARE

În cadrul acestui capitol voi descrie funcționalitățile de care dispune proiectul, atașând imagini reprezentative codului scris care îndeplinește funcționalitatea precizată.

5.1. CREARE CONT

Înregistrarea unui utilizator nou în Authentication Firebase se realizează cu ajutorul metodei `createUserWithEmailAndPassword()`. După validarea datelor preluate dintr-un formular de înregistrare, precum adresa de e-mail și parola furnizate de utilizator, le voi transmite ca parametri la metoda menționată în *Figura 5.1*.

```
createUserWithEmailAndPassword(auth, mail, pass)
```

Figura 5.1. Apelul metodei createUserWithEmailAndPassword()

În cazul în care există deja un utilizator care a folosit aceeași adresă de e-mail se va afișa un mesaj pentru a înștiința utilizatorul (vezi *Figura 5.2*).



Figura 5.2. Mesaj eroare

În urma înregistrării unui nou cont, am salvat datele preluate din formularul de înregistrare în fișierul „Conturi” din baza de date utilizând metoda `set()` (vezi *Figura 5.3*). Această metodă este utilizată pentru a salva date într-o referință specificată, înlocuind orice date existente pe calea respectivă [34].

```
106 get(child(dbRef, "Conturi/" + uid)).then((snapshot) => {
107   if(!snapshot.exists()){
108     set(ref(db, "Conturi/" + uid),{
109       Nume: name,
110       Utilizator: username,
111       Mail: mail,
112       Telefon: phone,
113       Judet: county,
114       Oras: city,
115       Adresa: address,
116       Cod_postal: code,
117       Parola: encPass(),
118       Rol: "user"
119     })
120   }.then(() => {
121     alert("Contul a fost creat cu succes!");
122     signOut(auth)
123     .then(()=>{
124       console.log("Utilizatorul a fost deconectat");
125       window.location.replace("Form_login.html");
126     })
127     .catch((error)=>{
128       console.log(error);
129     })
130   })
131   .catch((error) => {
132     alert("Eroare: " + error);
133   })
134 })
135 });
```

Figura 5.3. Metoda set()

În cazul parolei introduse de către utilizator am aplicat o funcție care returnează parola criptată utilizând *algoritmul AES* al colecției *CryptoJS* (vezi Figura 5.4).

```
147 //funcția de criptare a parolei
148 function encPass(){
149     var psw = document.getElementById("passBox");
150     var password = CryptoJS.AES.encrypt(psw.value, psw.value);
151     // returneaza parola criptata
152     return password.toString();
153 }
```

Figura 5.4. Funcție de criptare

5.2. AUTENTIFICARE

Utilizatorii dispun de două metode de autentificare, fie prin contul de *Google* utilizând metoda *signWithRedirect()*, fie prin un cont creat pe site descris în subcapitolul 5.1., utilizând metoda *signInWithEmailAndPassword()*.

5.2.1. METODA *signWithRedirect()*

Există două posibilități de autentificare cu *Firebase* folosind conturile de *Google*. Fie puteți utiliza *SDK-ul Firebase* pentru a efectua fluxul de conectare *Google*, fie să efectuați fluxul de conectare manual, utilizând biblioteca de conectare cu *Google* și transmitând jetonul de identificare rezultat către *Firebase* [35].

În cadrul proiectului am ales prima variantă, cea de utilizare a *SDK-ului Firebase JavaScript*, fiind cel mai simplu mod de a autentifica utilizatorii cu *Firebase* utilizând contul *Google*. Prima etapă a fost cea de a importa metodele necesare (vezi Figura 5.5).

Prima metodă declarată o reprezintă *getAuth()* care este o poartă de acces către *API-ul* de autentificare *Firebase*. Cu ajutorul acestei metode putem gestiona conturile de utilizator și acreditările acestora. Următorul pas a fost acela de a crea o instanță a obiectului *GoogleAuthProvider*.

După declarațiile menționate mai sus am adăugat un eveniment pentru butonul de conectare cu contul de *Google* în care am apelat metoda de redirectionare *signInWithRedirect()* către pagina de conectare.

```
3 import {getAuth, GoogleAuthProvider, signInWithRedirect}
4 from "https://www.gstatic.com/firebasejs/9.8.1/firebase-auth.js";
5
6 // folosit pentru preluarea datelor suplimentare
7 const auth = getAuth();
8
9 // instanta a obiectului furnizor Google
10 const provider = new GoogleAuthProvider();
11
12 const googleBtn = document.getElementById("googleBtn");
13 // eveniment pentru buton
14 googleBtn.addEventListener('click', function(){
15     // redirectionare catre pagina de conectare
16     signInWithRedirect(auth, provider);
17 })
```

Figura 5.5. Metoda *signInWithRedirect()*

5.2.2. METODA `signInWithEmailAndPassword()`

Pe lângă autentificare prin contul de *Google*, există și posibilitatea de autentificare print-un cont creat pe site (vezi *subcapitolul 5.1*). Această acțiune este realizată prin metoda *signInWithEmailAndPassword* care este apelată cu datele introduse în formularul de autentificare, date precum adresa de e-mail și parola (vezi *Figura 5.6*). În cazul în care autentificare eșuează din cauza introducerii greșite a unor date, va fi afișat un mesaj de eroare. Dacă autentificarea este realizată cu succes va fi apelată metoda *onAuthStateChanged()* care face posibilă preluarea datelor despre utilizatori. Această metodă va fi descrisă mai amplu în următorul subcapitol.

```
57 const logBtn = document.getElementById("logBtn");
58 // eveniment pentru autentificare
59 logBtn.addEventListener('click', function(){
60     const mail = document.getElementById("mailBox").value;
61     const pass = document.getElementById("passBox").value;
62
63     signInWithEmailAndPassword(auth, mail, pass)
64     .then(() => {
65         // autentificare reusita
66         document.getElementById("error-name").innerHTML = ""
67     })
68     .catch((error) => {
69         const errorCode = error.code;
70         const errorMessage = error.message;
71         console.log(errorCode + errorMessage);
72         document.getElementById("error-name").innerHTML = "Mail sau parola gresita!";
73         document.getElementById("error-name").style.color = "red";
74     });
75 })
```

Figura 5.6. Metoda *signInWithEmailAndPassword()*

5.2.3. METODA `onAuthStateChanged()`

Metoda *onAuthStateChange* are rolul de a adaugă un observator pentru a putea vedea modificările stării de conectare a utilizatorului [36]. Acest observator este declanșat atunci când utilizatorii se conectează, deconectează sau când identificadorul utilizatorului se schimbă în situații precum schimbarea parolei [36]. Astfel, în urma autentificării fie prin contul de Google, fie printr-un cont creat pe site, această metodă este apelată automat.

După cum se poate observa în *Figura 5.7*, am utilizat metoda *get()* pentru a obține un *snapshot* al datelor din baza de date [37]. De fiecare dată când se citesc datele din baza de date, acestea sunt primite sub forma unui *DataSnapshot*, care conține datele dintr-o locație a bazei de date, ce pot fi extrase apelând metoda *val()*, alternativ pot fi parcurse în snapshot prin apelul metodei *child()* pentru a returna snapshot-urile copiilor, ce pot fi apelate la rândul lor cu ajutorul metodei *val()* [38]. Aceste date nu se schimbă niciodată, doar prin utilizarea metodei *set()* prezentă și în *Figura 5.7*.


```
22  auth.onAuthStateChanged(function(user){
23    const dbRef = ref(db);
24    if(user){// credentiale utilizator conectat
25      const uid = user.uid;
26      const name = user.displayName;
27      const mail = user.email;
28      get(child(dbRef, "Conturi/" + uid)).then((snapshot) => {
29        sessionStorage.setItem("uid", uid);// stocheaza uid-ul user-ului curent
30        if(snapshot.exists()){
31          window.location="/form_home.html";
32        }else{
33          set(ref(db, "Conturi/" + uid),{
34            Nume: name,
35            Utilizator: '',
36            Mail: mail,
37            Telefon: '',
38            Judet: '',
39            Oras: '',
40            Adresa: '',
41            Cod_postal: '',
42            Parola: ''
43          })
44          .then(() => {
45            alert("Contul a fost creat cu succes!");
46            window.location.replace("form_home.html");
47          })
48          .catch((error) => {
49            alert("Eroare: " + error);
50          })
51        }
52      });
53    }else{
54      console.log("neconectat");
55    }
56  });
57 }
```

Figura 5.7. Metoda onAuthChanged()

5.2.4. METODA sendPasswordResetEmail()

Este indicat ca orice site cu parte de autentificare să dispună de opțiunea resetării parolei pentru a nu fi nevoit să creeze numeroase conturi. Astfel, resetarea parolei în cadrul proiectului se realizează cu ajutorul metodei prezentate în *Figura 5.8*. Această metodă este apelată în evenimentul adăugat butonului de trimitere din cadrul paginii de resetare a parolei (vezi subcapitolul 6.2.2), care va trimite automat un link de resetare la adresa introdusă în cazul în care există un cont cu adresa respectivă. În urma accesării link-ului trebuie să se introducă noua parolă care respectă un anumit format. După toți pașii și criteriile îndeplinite parola va fi modificată automat și în baza de date, datorită metodei *onAuthStateChanged()* descrisă în *subcapitolul 5.2.3*.

```
157  sendPasswordResetEmail(auth, email)
158  .then(function(){
159    document.getElementById("error-mail").innerHTML = "Mail-ul a fost trimis cu succes!";
160    document.getElementById("error-mail").style.color = "green";
161    setTimeout(redirectPage, 1000);
162  })
163  .catch(() => {
164    document.getElementById("error-mail").innerHTML = "Nu exista cont cu aceasta adresa de mail!";
165    document.getElementById("error-mail").style.color = "red";
166  });
167 }
```

Figura 5.8. Apel metoda sendPasswordResetEmail()

5.3. ADĂUGAREA ȘI ELIMINAREA PRODUSELOR DE LA FAVORITE

Odată ce un produs este adăugat la favorite pictograma tip inimă din partea stângă a secțiunii produsului se va modifica în roșu, iar dacă va fi șters produsul de la favorite va deveni albă, datorită claselor de stil pe care le atribui elementului în funcție de adăugare sau ștergere (vezi *Figura 5.9*).

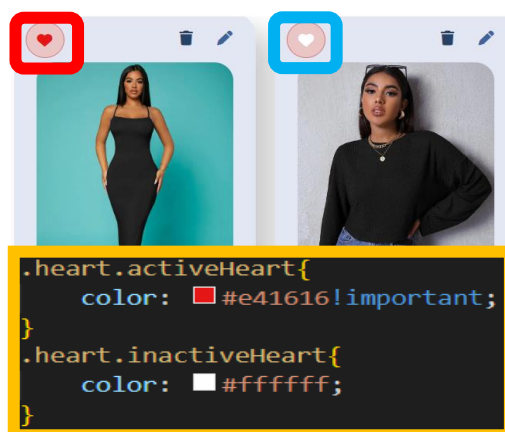


Figura 5.9. Modificarea culorii pictogramei

Variabila *favoriteBy* (vezi *Figura 5.10*) reprezintă de fapt o listă a tuturor utilizatorilor care au apreciat produsul selectat. Pentru început am creat o instanță a obiectului *Set()*. În cazul în care lista *favoriteBy* nu este goală voi adăuga în instanța obiectului, atât uid-ul curent, cât și toate uid-urile existente în listă, pe care am parcurs-o element cu element, prin metoda *forEach()*. În caz contrar, voi adăuga doar utilizatorul curent.

```
let allFavorite = new Set();
if(favoriteBy){
  allFavorite.add(uid);
  favoriteBy.forEach(item => {
    allFavorite.add(item);
  })
}else{
  allFavorite.add(uid);
}
```

Figura 5.10. Lista de utilizatori care au apreciat produsul

După ce am setat utilizatorii în variabila *allFavorite*, vom actualiza lista de utilizatori care au apreciat produsul transformând instanța într-o listă cu ajutorul metodei *Array.from()*, de asemenea se va actualiza și câmpul *NrRecomandare* cu lungimea listei de utilizatori, acest câmp fiind descris mai în detaliu în următorul capitol. Prin utilizarea metodei *update()*, se actualizează valorile secundare de la nivelul inferior a căii menționate (vezi *Figura 5.11*).

```
update(child(dbRef, "Produse/" + key), {
  FavoriteBy: Array.from(allFavorite),
  NrRecomandare: allFavorite.size
})
```

Figura 5.11. Actualizarea listei de utilizatori la adăugare

În cazul eliminării produsului de la favorite se procedează puțin diferit, deoarece vom crea o nouă listă cu ajutorul metodei *filter()*. Astfel în urma filtrării vom returna toate uid-urile utilizatorilor cu excepția utilizatorului curent, după care vom apela metoda *update()* pentru a actualiza atât lista, cât și numărul de recomandări.

```
let updatedList = favoriteBy.filter(item => {  
  return item !== uid;  
});  
update(child(dbRef, "Produse/" + key), {  
  FavoriteBy: updatedList,  
  NrRecomandare: updatedList.length  
})
```

Figura 5.12. Actualizarea listei de utilizatori după ștergere

5.4. AFIȘAREA PRODUSELOR RECOMANDATE

Afișarea produselor recomandate presupune o sortare în ordine descrescătoare a produselor în funcție de *NrRecomandare* despre care am menționat în capitolul anterior. Pe scurt, acest câmp reprezintă numărul total de utilizatori care au apreciat respectivul produs. Metoda de sortare este apelată într-o interogare, metodă denumită *query()*, aceasta având rolul de a ajuta la regăsirea informațiilor conform condițiilor impuse din interiorul acesteia.

Astfel, am apelat în interiorul metodei *query* ce conține referința către baza de date, metoda *orderByChild()* căreia i-am plasat numele câmpului după care doresc să fac sortarea. Totodată am apelat și metoda *limitToLast()* pentru a afișa ultimele produse care sunt cele mai apreciate, adică au cel mai mare număr de recomandări, deoarece sortarea se face în mod implicit în ordine crescătoare. Acel query a fost apelat în cadrul metodei *get()* pentru a citi toate nodurile și a parcurge toate datele acestora pe care ulterior le va adăuga sub forma unui *DataSnapshot* (descriș în cadrul subcapitolului 5.2.3) într-o listă (vezi linia 202 din Figura 5.13).

```
196 function GetAllData(){  
197   const que = query(ref(db, "Produse"),orderByChild("NrRecomandare"), limitToLast(4));  
198   get(que)  
199   .then((snapshot) => {  
200     var prod = [];  
201     snapshot.forEach(childSnapshot => {  
202       prod.push(childSnapshot);  
203     });  
204     AddAllItems(prod);  
205   })  
206   .catch((error) => {  
207     console.log("Mesaj eroare " + error);  
208   })  
209 }  
210 window.onload = GetAllData;
```

Figura 5.13. Ordonarea produselor

În interiorul funcției *AddAllItems()* (vezi *Figura 5.14*) am parcurs fiecare element din lista ordonată în ordine descrescătoare (vezi linia 178 din *Figura 5.14*), după care am generat dinamic produsele în funcția *AddItem()* ce a fost apelată cu datele din listă plasate ca parametrii.

```
177 function AddAllItems(Produse){  
178     Produse.reverse();  
179     divGenerate.innerHTML = "";  
180     Produse.forEach(element => {  
181         AddItem(element.key, element.val().Brand, element.val().Poza,  
182         element.val().Poza2, element.val().Nume, element.val().Pret,  
183         element.val().Reducere, element.val().PretReducus, element.val().FavoriteBy);  
184     })  
185 }  
186
```

Figura 5.15. Parcurgerea listei de produse

5.5. ADĂUGAREA PRODUSELOR ÎN COȘUL DE CUMPĂRĂTURI

Adăugarea produselor în coșul de cumpărături se poate face doar din pagina de detalii (vezi *Figura 5.16.*) descrisă mai în detaliu în capitolul următor (vezi *subcapitolul 6.2.6*).

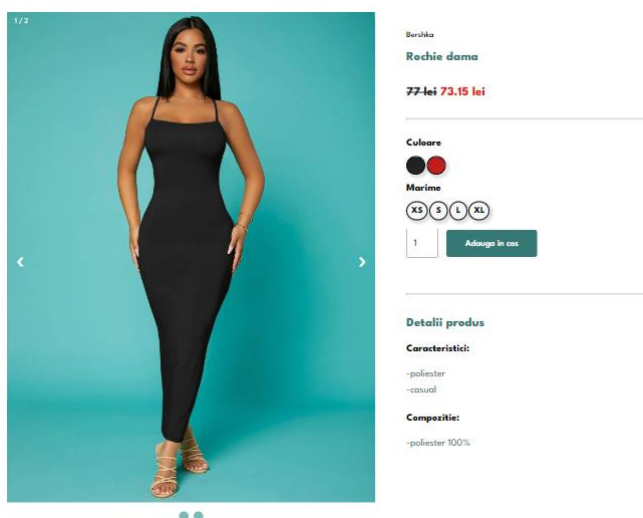


Figura 5.16. Pagina de detalii

În momentul în care are loc trimiterea către pagina de detalii a unui produs este apelată funcția din *Figura 6.17* în care este apelată metoda *setItem()* ce aparține obiectului de stocare *sessionStorage*. Datele stocate cu *sessionStorage* vor fi șterse odată cu încheierea sesiunii. Metoda *setItem()* are ca parametrii un nume de cheie și valoare pe care dorim s-o atribuim acestei chei.

Această metodă de setare am utilizat-o pentru a avea cheia produsului selectat salvată într-o variabilă pentru a afișa datele corespunzătoare produsului selectat (vezi *Figura 5.17*). Astfel am făcut o setare atât pentru cheia produsului, cât și pentru pozele acestuia pentru afișare.

```
2116 function goToDetails(img1, img2, keyProd) {  
2117  
2118     sessionStorage.setItem("image1", img1);  
2119     console.log("SESSION STORAGE IMG 1 ==== " + sessionStorage.getItem("image1"));  
2120  
2121     sessionStorage.setItem("image2", img2);  
2122     console.log("SESSION STORAGE IMG 2 ==== " + sessionStorage.getItem("image2"));  
2123  
2124     sessionStorage.setItem("keyProd", keyProd);  
2125     console.log("SESSION STORAGE KEY ==== " + sessionStorage.getItem("keyProd"));  
2126  
2127     window.location.href = "form_details.html";  
2128 }
```

Figura 5.17. Utilizarea metodei `setItem()`

Astfel, în *pagina de detalii* am preluat valoarea cheii date ca parametru metodei `getItem()`, ce pe a fost setată în Figura 5.17. Pentru a putea citi datele produsului și a le afișa în pagină am utilizat metoda `get()` (vezi Figura 5.18.).

```
var keyProd = sessionStorage.getItem("keyProd");  
get(child(dbRef, "Produse/" + keyProd)).then((snapshot) => {  
    if (snapshot.exists()) {
```

Figura 5.18. Utilizarea metodei `getItem()`

Butonul de adăugare în coș trebuie utilizat abia după selectarea unei culori și a unei mărimi, în caz contrar va fi afișat un mesaj de eroare care va îndruma utilizatorul. Odată cu apăsarea butonului are loc adăugare în coșul de cumpărături utilizând metoda `push()`, metodă care creează automat o cheie unică la adăugarea datelor în baza de date (vezi Figura 5.19). Toate datele din interiorul metodei, precum marcă, nume, preț etc. sunt date preluate din fișierul „Produse”.

```
push(ref(db, "Conturi/" + uid + "/CosCumparaturi/" + keyProd), {  
    Brand: marca,  
    Nume: nume,  
    Pret: pret,  
    PretRedus: pretRedus,  
    Poza: mainImg.src,  
    Model: model,  
    Culoare: color,  
    Marime: size,  
    Cantitate: nr.value,  
    Subtotal: subtotal,  
    MaximStoc : maximStoc  
})  
})
```

Figura 5.19. Utilizarea metodei `push()`

5.6. ȘTERGEREA PRODUSELOR DIN COȘUL DE CUMPĂRĂTURI

Ștergerea produselor din coșul de cumpărături este realizată cu ajutorul metodei `remove()` care are rolul de a ștergere nodul de la calea dată, inclusiv toți copiii acestuia. În urma ștergerii am eliminat produsul din tabelul afișat, iar în cazul în care nu mai există produse în coșul de cumpărături am generat dinamic un mesaj cu ajutorul metodei `createElement()` (vezi Figura 5.20).

```
remove(ref(db, "Conturi/" + uid + "/CosCumparaturi/" + key + "/" + childKey))
.then(() => {
    alert("Produsul a fost sters din cosul de cumparaturi!");
    document.getElementById("tr-" + key).remove();

    let paragraph = document.createElement('p');
    paragraph.id = "message";
    paragraph.innerHTML = '<br>' + 'NU EXISTA PRODUSE IN COSUL DE CUMPARATURI!';
    paragraph.className = "showMessage";
    document.getElementById("divMesaj").appendChild(paragraph);
})
```

Figura 5.20. Ștergerea din coșul de cumpărături

În figura 5.21. este prezentată interfața paginii pentru coșul de cumpărături, înainte și după ștergerea tuturor produselor.

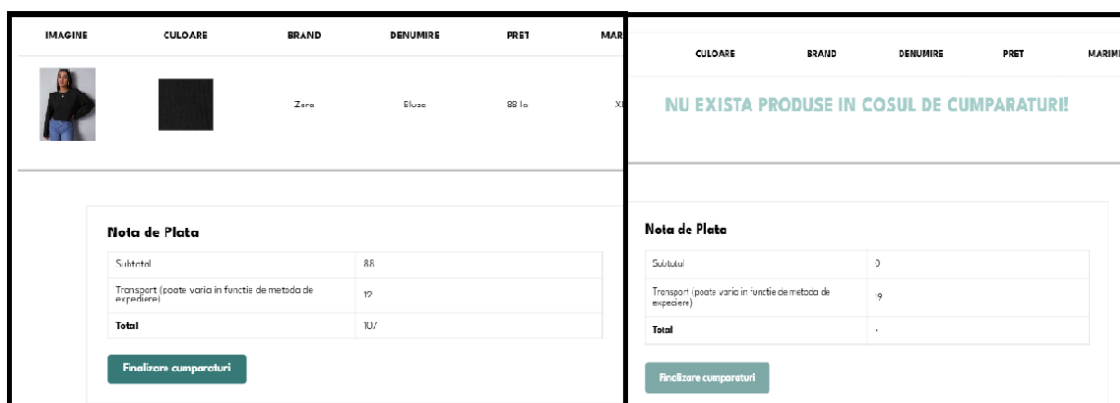


Figura 5.21. Pagina destinată coșului de cumpărături
(înainte și după ștergere)

5.7. PLASAREA UNEI COMENZI

După plasarea comenzii va fi trimis un mail către utilizator în care va avea atașată o factură în format *pdf* (vezi Figura 5.23). Astfel, pentru această funcționalitate am folosit biblioteca *smtp.js*. În cadrul funcției am apelat metoda `send()`, ce conține datele necesare trimiterii unui mail, precum *SecureToken* ce a fost generată pe site-ul oficial *smtpjs.com*, cheie folosită pentru criptarea acreditărilor, acest proces a fost descris mai amplu în *capitolul*

3.4.1. Pe lângă această cheie mai sunt enumerate date precum, destinatar, expeditorul, subiectul, conținutul mail-ului, cât și factura atașată. În cazul atașării unui document trebuie cunoscute numele facturii, cât și conținutul documentului care trebuie convertit la string (vezi Figura 5.22) .

```
410 function sendEmail(factura, url) {  
411  
412     Email.send({  
413         SecureToken : "d4e41b30-45f4-4df0-9742-b6750961dc90",  
414         To : document.getElementById("email").value,  
415         From : "muntyancraciunela@gmail.com",  
416         Subject : "Procesare comanda - " + uid,  
417         Body : 'Multumim ca ati ales produsele noastre!<br><br>' +  
418             'Comanda este in curs de procesare, o sa revenim curand cu un mail'+  
419             ' pentru a va tine la curent cu starea comenzii.'+  
420             '<br><br><br>Multumim,<br> Echipa MIMI',  
421         Attachments: [{  
422             name: factura,  
423             data: url      var pdfBase64 = doc.output('datauristring');  
424         }]  
425     })  
426     .then( () => {  
427         setTimeout(refreshPage, 2000);  
428         document.getElementById("message").display = "none";  
429     });  
430 }
```

Figura 5.22 Funcția de trimitere e-mail

Pentru generarea unui document *pdf*, am utilizat biblioteca *jsPDF*, iar în cadrul acestui pdf am lucrat fie cu tabeluri, fie cu text simplu. După cum vedeți în Figura 5.23 a fost declarată o nouă instanță a obiectului *jsPDF()*, astfel metoda *text()* are trei parametri, primul reprezentând textul care va fi afișat, iar următorii doi reprezintă coordonatele de așezare față de marginea din stânga, respectiv marginea superioară. Metoda *autoTable()* are rolul de a genera un tabel cu datele declarate. De asemenea mai există și metoda *save()* care are rolul de a descărca automat factura pe dispozitiv.

```
var doc = new jsPDF();  
  
doc.text("FACTURA", 40, 30);  
  
doc.autoTable({  
    head: [['Nr. Factura', 'Data emiterii', 'Ora emiterii']],  
    body: [  
        [nr, zi + '/' + luna + '/' + an, ora + ':' + minute + ':' + secunde]  
    ]  
});  
  
doc.save(umeFactura);
```

Figura 2.23. Generarea unui pdf

5.8. CĂUTAREA UNUI PRODUS

Funcția de căutare este apelată într-un *forEach()* care parcurge toate produsele. În cazul în care numele unui produs este format din mai multe cuvinte, am apelat metoda *split* pentru a le despărți, astfel încât să ofer posibilitatea utilizatorului să caute un produs indiferent după care porțiuni din nume a fost căutat.

Cu ajutorul unui *for-of* am parcurs numele tuturor produselor, iar într-o instrucțiune *if* am apelat metoda *toLowerCase()* atât pentru numele din baza de date, cât și pentru numele căutat pentru a ignora cazurile de scriere (vezi *Figura 5.24*). În cazul în care acestea sunt egale se vor afișa produsele după numele căutat.

```
365 function searchKeyword(childSnapshot, input){  
366   let nume = childSnapshot.val().Nume;  
367   let words = nume.split(" ");  
368   for(let word of words){  
369     if(word.toLowerCase() === input.toLowerCase()){  
370       return childSnapshot;  
371     }  
372   }  
373 }  
374 }
```

Figura 5.24. Funcția de căutare produse după nume

5.9. SORTAREA PRODUSELOR

La fel ca în cazul produselor recomandate, pentru sortarea produselor am folosit o interogare în care am apelat metoda *orderByChild()*, transmițându-i ca parametru numele câmpului după care doresc să fie făcută sortare (vezi *Figura 5.25*). Acest *query* fiind la fel apelat într-o metodă *get()* în care vor fi afișate produsele în ordine prețurilor.

```
action getFilterInPrice(){  
  const q = query(ref(db, "Produse"),orderByChild("Pret"));  
  return q;  
}
```

Figura 5.26. Ordonare după preț

5.10. DECONECTAREA UTILIZATORULUI

În cadrul oricărui site trebuie să existe opțiunea de deconectare, astfel aceasta este realizată cu ajutorul metodei *signOut()*, căruia i-a fost transmis ca parametru variabila *auth* pentru a avea acces la credențialele utilizatorului conectat (vezi *Figura 5.27*).

```
document.getElementById("signout").addEventListener('click', () => {  
  signOut(auth)  
    .then(()=>{  
      console.log("Utilizatorul a fost deconectat");  
      window.location.replace("form_login.html");  
    })  
    .catch((error)=>{  
      console.log(error);  
    })  
});
```

Figura 5.27 Deconectarea utilizatorului

5.11. NOTIFICAREA UTILIZATORULUI

În pagina de detalii a produselor va fi afișat un nou buton (vezi *Figura 5.28*) doar dacă respectivul produs nu mai este disponibil, adică stocul acestuia este epuizat. În urma apăsării butonului va fi solicitată o cerere pentru a fi notificat pe e-mail atunci când produsul va reveni în stoc.

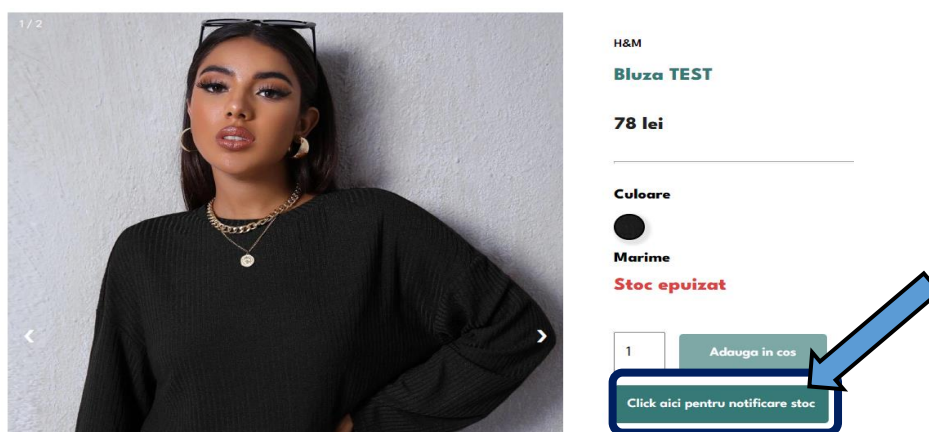


Figura 5.28. Buton notificare client

În cadrul evenimentului adăugat butonului de notificare, am utilizat metoda *push()* (vezi *Figura 5.29*) pentru a adăuga datele necesare, precum numele clientului, adresa acestuia de e-mail pentru a ști la ce adresă va fi trimisă notificarea, cheia produsului, numele produsului și prețul acestuia în fișierul „*Notificări*”.

```
push(ref(db, "Notificari/"), {  
  NumeClient: nume,  
  MailClient: mail,  
  IdProdus: keyProd,  
  NumeProdus: name.textContent,  
  PretProdus: price.textContent  
});
```

Figura 5.29. Adăugarea datelor necesare pentru notificare

În urma procesului de editare al unui produs, voi face o verificare asupra stocului, așa cum este prezentat în *Figura 5.30*. pentru a afla dacă acesta a suferit modificări. Astfel, în cazul în care acesta este diferit de zero (au fost adăugate produse în stoc), voi apela funcția de notificare pe mail, având ca parametru cheia produsului respectiv.

```
if(stoc != 0){  
  |  
  |    sendMail(key);  
  |  
}else{
```

Figura 5.30. Apelarea funcției de notificare

În cadrul funcției de notificare pe mail (vezi *Figura 5.31*), am efectuat o citire a fișierului „*Notificari*” utilizând metoda *get()* pentru a parcurge toate nodurile adăugate ce conțin datele necesare clientului care a solicitat notificarea. În cazul în care cheia dată ca și parametru (cheia produsului care este în curs de editare) este egală cu valoarea stocată în câmpul „*IdProdus*”, atunci se va realiza trimiterea unui mail legat de revenirea produsului solicitat în stoc. Ulterior am apelat metoda *remove()* (vezi linia 2978 din cadrul *Figurii 5.31*) pentru a șterge utilizatorul notificat, așadar clientul nu va fi notificat de fiecare dată când acel produs este modificat.

```
2957 function sendMail(cheie){
2958     console.log("Cheie ", cheie);
2959
2960     get(ref(db, "Notificari/")).then((snapshot) => {
2961         snapshot.forEach( childObj => {
2962             console.log("Adresa de mail ", childObj.val().MailClient);
2963             if(cheie == childObj.val().IdProdus){
2964                 Email.send({
2965                     SecureToken : "d4e41b30-45f4-4df0-9742-b6750961dc90",
2966                     To : childObj.val().MailClient,
2967                     From : "muntyancraciunela@gmail.com",
2968                     Subject : "Stoc Produs" + cheie,
2969                     Body : "Buna " + childObj.val().NumeClient + ",<br> Produsul <b>" + childObj.val().NumeProdus +
2970                           "</b> cu pretul de <b>" + childObj.val().PretProdus + "</b> a revenit in stoc." +
2971                           "<br><br>Cu respect,<br>Echipa MIMI"
2972                 })
2973                 .then( message => {
2974                     console.log("MAIL UL A FOST TRM CU SUCCES");
2975                     console.log(message);
2976                 });
2977             };
2978             remove(child(dbRef, "Notificari/" + childObj.key))
2979             .then(() => {
2980                 console.log("Utilizatorul a fost sters de la notificari: ", childObj.val().MailClient);
2981             })
2982         })
2983     })
2984 }
```

Figura 5.31. Funcție de notificare client pe e-mail

6. UTILIZAREA APLICAȚIEI

6.1. CONFIGURAȚII

Pentru ca un client să se poată bucura de produsele oferite de site, acesta are nevoie de două lucruri: să acceseze site-ul din cadrul unui browser de pe un dispozitiv conectat la internet (PC/smartphone) și să-și folosească contul personal.

În cazul în care clientul este unul nou, acesta își poate crea un cont în doar câteva minute sau chiar să își utilizeze contul de Google pentru a sari peste acest pas.

6.2. MANUAL DE UTILIZARE

Acest manual de utilizare a fost creat cu scopul de a ajuta utilizatorii să folosească site-ul *MIMI* mai ușor și mai rapid. Veți întâlni capturi de ecran detaliate cu instrucțiuni de folosire și va fi prezentată fiecare pagină în parte alături de funcțiile acestora.

6.2.1. PAGINA DE AUTENTIFICARE

Primul pas pe care un utilizator trebuie să-l facă este acela de a se înregistra, de a crea un cont pentru acest magazin online, deoarece nu este posibilă navigarea fără a avea un cont. Acest lucru se face din *pagina de autentificare* apăsând click pe „*Înregistrează-te acum!*” (vezi *Figura 6.1.*), acțiune care vă va redirecționa către *pagina de înregistrare* (descrisă în *subcapitolul 6.2.3.*). Dacă nu doriți să vă creați un cont, puteți utiliza contul de *Google*.

Al doilea pas care trebuie urmat este autentificarea. După cum am menționat mai sus, autentificarea se poate face fie printr-un cont creat pe site, în *pagina de înregistrare*, fie prin contul de *Google*. Dacă alegeți prima opțiune de autentificare, următorul pas ar fi introducerea adresei de e-mail și a parolei pe care le-ați ales în partea de înregistrare, iar mai apoi trebuie să apăsați pe butonul de conectare, cel de culoare albastră (vezi în *Figura 6.1. chenarul marcat cu numărul 3*). În urma verificării existenței contului, dacă acesta a fost scris greșit sau nu există va fi afișat un mesaj pentru a vă înștiința, în acest caz trebuie doar să recompletați câmpurile pentru a reface conectarea.

În cazul în care aveți deja un cont dar v-ați uitat parola sau doriți să o schimbați din alte motive, trebuie doar să apăsați pe „*Ai uitat parola?*” (vezi în *Figura 6.1 chenarul marcat cu numărul 5*) și veți fi redirecționat către o altă pagină, descrisă mai în detaliu în următorul subcapitol.

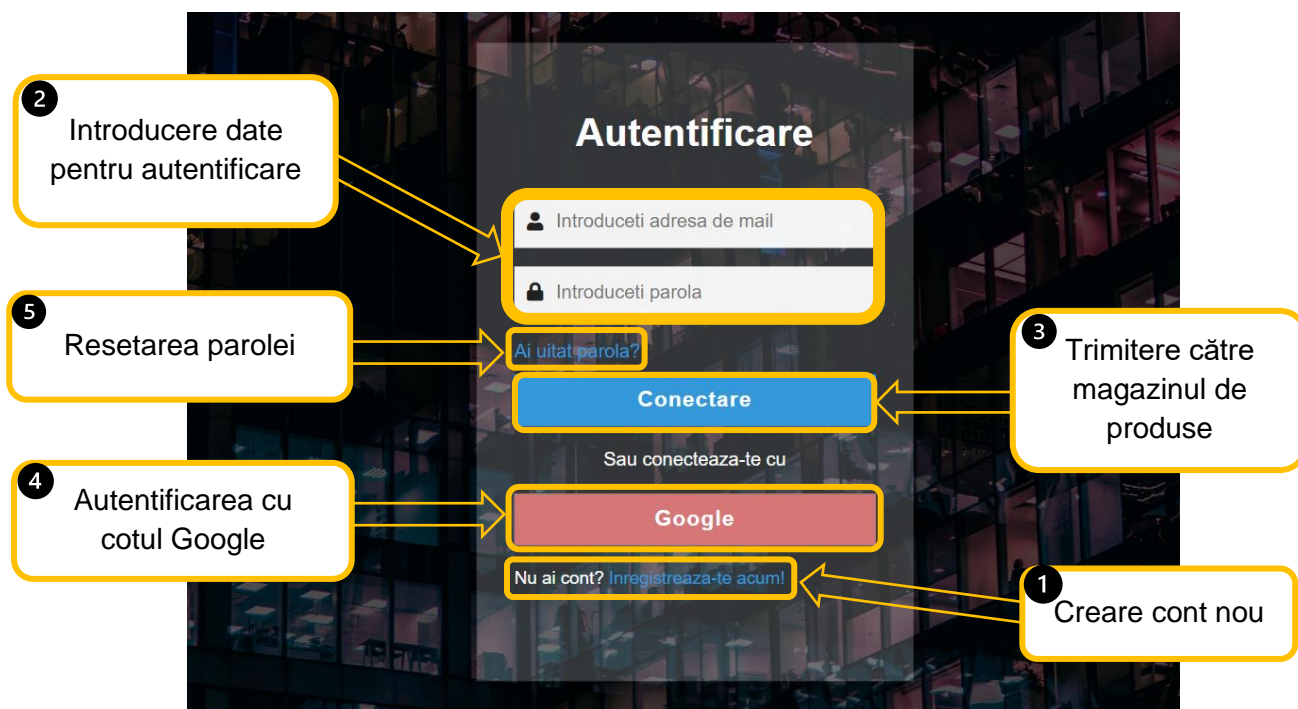


Figura 6.1. Pagina de autentificare

Dacă ați introdus datele valide ale unui cont existent, veți aștepta câteva secunde după care veți fi redirecționat către *pagina de acasă* (descrișă în *subcapitolul 6.2.4*). În cazul în care doriți să vă autentificați cu ajutorul contului de *Google*, trebuie doar să apăsați pe butonul roșu (vezi *Figura 6.1*), după care veți fi redirecționați către o pagină în care vă selectați contul *Google* dorit (vezi *Figura 6.2*). Ulterior veți fi trimis înapoi la *pagina de autentificare*, iar mai apoi, într-un scurt timp la *pagina de acasă*.

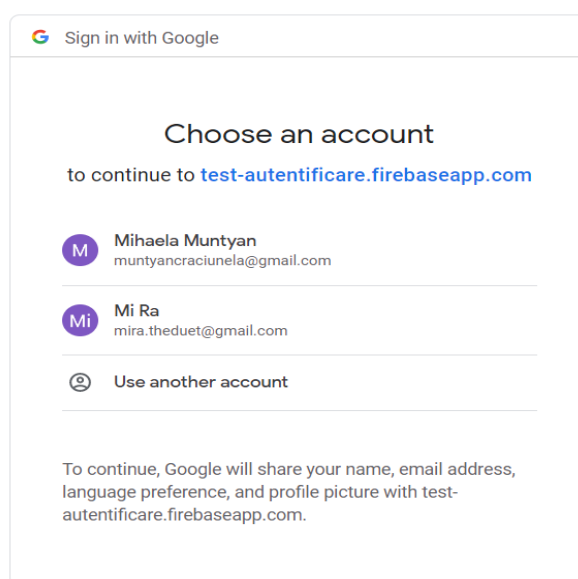


Figura 6.2. Alegere cont google

6.2.2. PAGINA DE RESETARE A PAROLEI

În această pagină se poate ajunge doar din pagina de autentificare. Astfel, pentru a vă reseta parola trebuie să vă introduceți adresa de e-mail a contului în câmpul indicat de săgeata din dreapta regăsită în *Figura 6.3*, iar ulterior trebuie să apăsați pe butonul indicat pentru solitare (vezi *Figura 6.3*).

The screenshot shows a web form titled "Reseteaza parola". It contains a label "Adresa e-mail" above a text input field with the placeholder "Introduceți adresa dvs. de mail". Below the input field is a purple button labeled "Trimite". Annotations include a yellow box on the left with the text "Solicită schimbarea parolei" and an arrow pointing to the form, and a yellow box on the right with the text "Introducerea adresei de mail" and an arrow pointing to the input field.

Figura 6.3. Pagina de resetare a parolei

În urma solicitării dumneavoastră de resetare veți primi un mail, cu un link de resetare pe care trebuie să-l accesați (vezi *Figura 6.4*).



Figura 6.4. E-mail cu link de resetare a parolei

După accesarea link-ului veți fi trimis către o pagină în care veți avea posibilitatea de a vă introduce o parolă nouă (vezi *Figura 6.5*). Parola trebuie să îndeplinească anumite condiții de siguranță, în cazul în care parola introdusă de dumneavoastră nu îndeplinește acele condiții vă va apărea un mesaj cu roșu pentru a vă indica exact ce format trebuie să îndeplinească (vezi *Figura 6.5*).

Condițiile pe care trebuie să le îndeplinească parola

Resetați parola

pentru **mira.theduet@gmail.com**

Parolă nouă
....

Parolele puternice au cel puțin 6 caractere și conțin o combinație de litere și cifre

SALVAȚI

Figura 6.5. Modificarea parolei

După modificarea cu succes a parolei vă veți putea autentifica cu noua parolă urmând pașii din primul subcapitol.

6.2.3. PAGINA DE ÎNREGISTRARE

Cum am precizat și în subcapitolele anterioare, în această pagină aveți posibilitatea de a vă crea un cont pentru a vă putea autentifica și naviga pe site. Pentru înregistrare trebuie să completați toate câmpurile aferente formularului prezentat în *Figura 6.6*. Câmpurile trec printr-o serie de validări odată cu introducerea unei taste, acestea trebuie să îndeplinească anumite condiții precum este și afișat în cazul parolei (vezi *Figura 6.6*). Mesajele de eroare sunt afișate sub câmpurile în care introduceți date și sunt de culoare roșie pentru a ieși în evidență, acestea sunt destul de sugestive cât să vă ajute să le rezolvați. Abia după completarea corectă a acestora veți putea crea contul cu succes prin apăsarea butonului marcat în *Figura 6.6*.

Inregistreaza-te
Completați toate câmpurile!

Nume si prenume
Introduceți numele si prenumele dvs.

E-mail
Introduceți adresa dvs. de mail

Judet
Introduceți judetul

Adresa
Introduceți adresa dvs.

Parola
Parola trebuie sa fie formata din minim 6 caractere, cel puțin o litera, un numar si un caracter special!

Nume utilizator
Introduceți un nume de utilizator

Numar de telefon
Introduceți nr. dvs. de telefon

Oras
Introduceți orasul

Cod postal
Introduceți codul postal

Confirmare parola
Parolele nu coincid!

Creează cont

Parola nu îndeplinește condițiile precizate

Creează contul

Figura 6.6. Pagina de înregistrare

6.2.4. PAGINA DE ACASĂ

După autentificare, pagina de acasă este cea în care veți fi redirectionat. În aproape fiecare pagină veți regăsi meniul din *Figura 6.7*, care este fix și conține numele paginilor către care vă vor redirectiona. Pagina activă va fi evidențiată, cum este și cazul *paginii de acasă* (vezi secțiunea marcată cu roșu din *Figura 6.7*).



Figura 6.7. Pagina de acasă

În această pagină nu sunt întâlnite multe funcționalități, majoritatea este încărcată de reclame, cu excepția zonei de produse recomandate, în care vă vor fi afișate cele mai apreciate produse. Pe lângă meniul afișat în *Figura 6.7*, în marea majoritatea paginilor mai întâlnim butonul din dreapta paginii care vă ușurează munca trimițându-vă la începutul paginii, astfel nu mai sunteți nevoit să derulați pagina până la începutul acesteia.

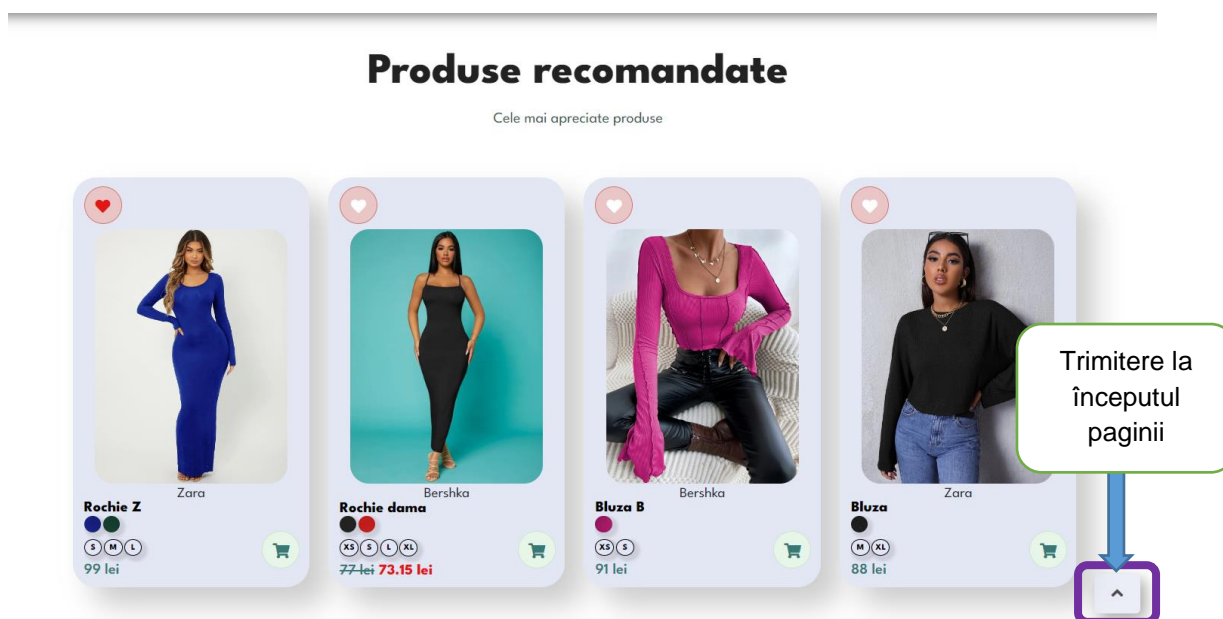
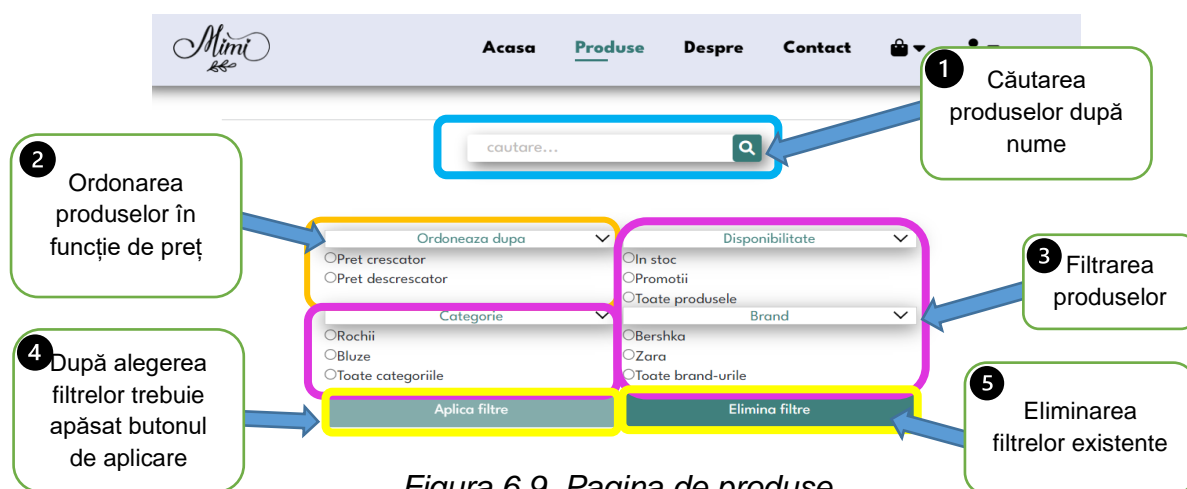


Figura 6.8. Produse recomandate

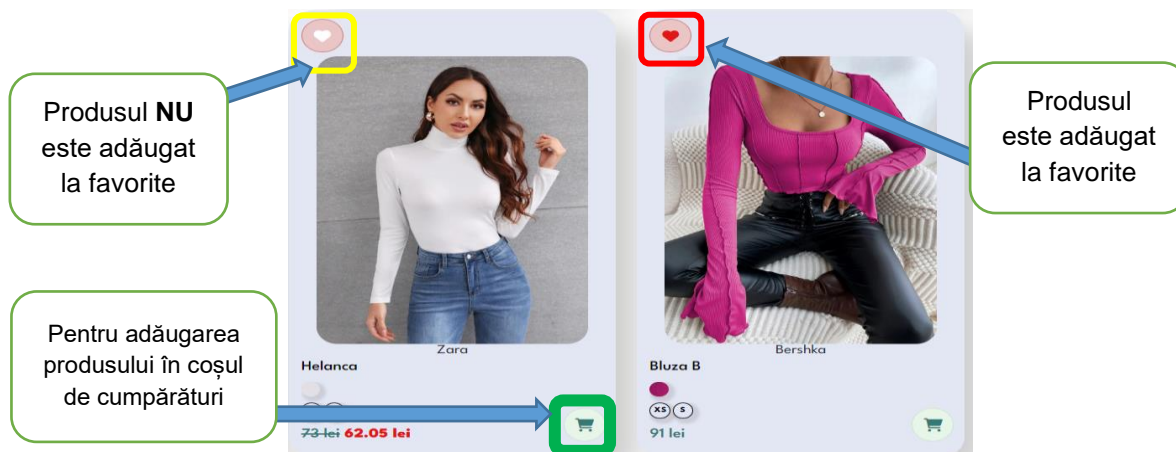
6.2.5. PAGINA DE PRODUSE

În pagina de produse sunt întâlnite cele mai multe funcționalități. Astfel, această pagină vă pune la dispoziție o metodă de căutare a produselor după nume (vezi chenarul 1 din *Figura 6.9*), metode de filtrare, adică vă vor fi afișate doar produsele care îndeplinesc condițiile selectate de dumneavoastră (vezi chenarul 3 din *Figura 6.9*), cât și metode de ordonare a produselor afișate. Pentru a aplica filtrele trebuie să apăsați butonul „Aplică filtre” evidențiat în chenarul 4 din *Figura 6.9*. De asemenea se pot elimina filtrele deja existente (vezi chenarul 5 din *Figura 6.9*).



*Figura 6.9. Pagina de produse
(filtrarea și căutarea produselor)*

Pentru a adăuga un produs la favorite, trebuie doar să faceți click pe inima din colțul din stânga, astfel dacă inima este de culoare roșie atunci produsul a fost adăugat la favorite (vezi *Figura 6.10*). Aceste produse pot fi vizualizate din pagina de produse favorite ce va fi descrisă în subcapitolul 6.2.9. Pentru adăugarea unui produs în coșul de cumpărături trebuie să apăsați pe coșul reprezentativ din secțiune de jos a produsului. Această acțiune vă va trimite către o pagină de detalii pentru a vă alege produsul după preferințe.



*Figura 6.10. Adăugarea produsului la favorite sau
în coșul de cumpărături*

6.2.6. PAGINA DE DETALII

Cum am menționat în subcapitolul anterior, în această pagină puteți realiza adăugarea produselor în coșul de cumpărături. Pentru a face acest lucru trebuie să urmați niște pași. Primul pas este acela de a vă selecta culoarea dorită, după care vă alegeți mărimea și cantitatea, ulterior făcând click pe buton se va adăuga în coșul de cumpărături produsul selectat (vezi *Figura 6.11*).



Figura 6.11. Pagina de detalii

În urma acestei acțiuni veți fi interogat dacă doriți să vedeți întregul coș de cumpărături (vezi *Figura 6.12*). Dacă apăsați „Cancel” nu se va întâmpla nimic, însă în cazul în care apăsați pe „OK” veți fi redirecționat către pagina de vizualizare a coșului de cumpărături.

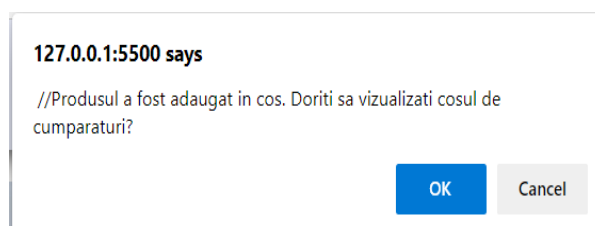


Figura 6.12. Mesaj de confirmare

În cazul în care un produs nu se mai află în stoc va fi afișat mesajul „Stoc epuizat”. Dacă doriți să cumpărați un produs indisponibil, aveți posibilitatea de a solicita o cerere pentru a fi anunțat în momentul în care acesta revine în stoc. Această solicitare se realizează prin apăsarea butonului „Click aici pentru notificare stoc” evidențiat în *Figura 6.13*. Încercarea adăugării în coșul de cumpărături a unui produs indisponibil nu este posibilă.

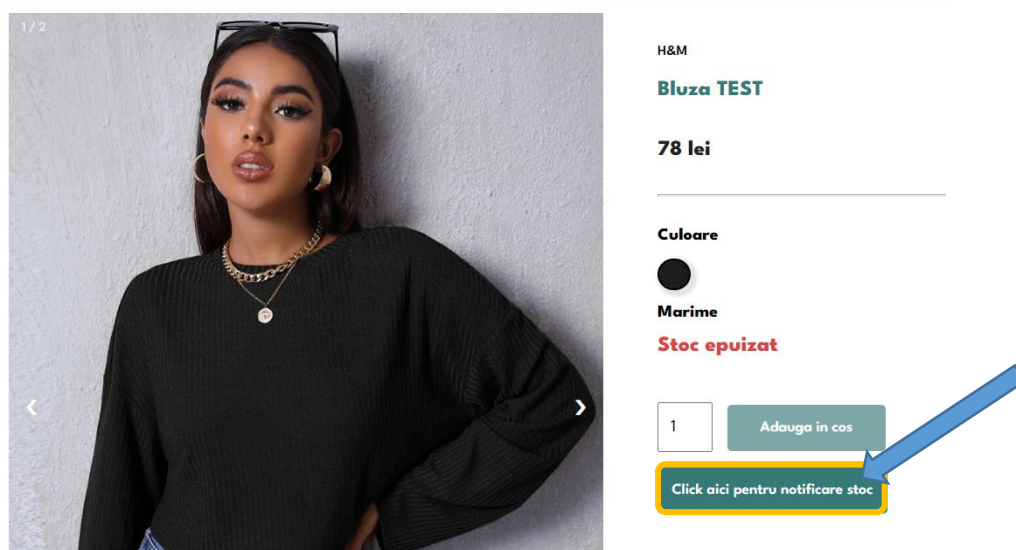


Figura 6.13. Buton notificare stoc

După ce ați apăsăat pe butonul evidențiat în *Figura 6.13*, veți primi un mail de confirmare precum solicitarea dumneavoastră a fost înregistrată (vezi *Figura 6.14*).

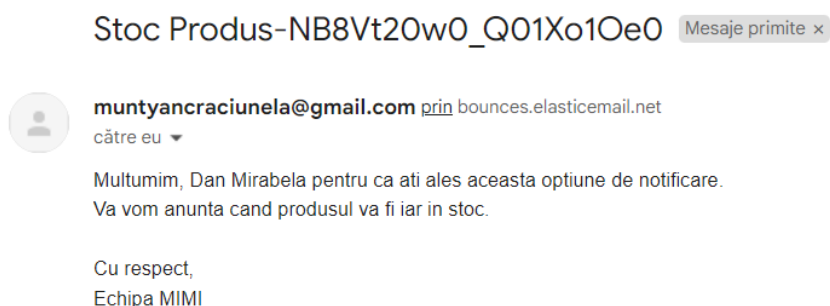


Figura 6.14. Mail de confirmare

În cazul în care produsul solicitat revine în stoc, veți primi un e-mail care vă va anunța acest lucru (vezi *Figura 6.15*).

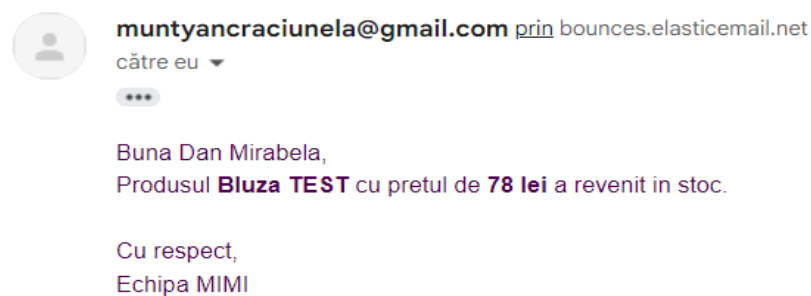


Figura 6.15. Notificare

6.2.7. PAGINA DE CONTACT

Această pagină cuprinde două secțiuni, una de *Help(ajutor)* și una de *Contact* după cum puteți vedea în Figura 6.16. Astfel dacă accesați link-urile (cuvintele subliniate din Figura 6.16) corespunzătoare secțiunilor acesta vă vor trimite la secțiune respectivă.



Figura 6.16. Secțiuni ale paginii de contact

În secțiunea de *Help* sunt prezentate cele mai frecvente întrebări care vă pot fi de ajutor, în cazul în care întrebarea dumneavoastră nu se regăsește printre cele menționate, aveți posibilitatea de a completa un formular în care va trebui să descrieți care este problema întâmpinată, după care să apăsați pe butonul evidențiat în Figura 6.17.

Figura 6.17. Formular de contact

6.2.8. PAGINA DE PRODUSE FAVORITE

Această pagină conține toate produsele pe care le-ați apreciat, având posibilitatea de a le vedea detaliile apăsând click pe pictograma în formă de ochi evidențiată în partea dreaptă, sau de a elimina produsul de la favorite apăsând pe „X”, pictograma evidențiată din partea stângă a tabelului (vezi Figura 6.18).





STERGE	IMAGINE	BRAND	DENUMIRE	PRET	DETALII
		Zara	Helanca	62.05 lei	
		Bershka	Bluza B	91 lei	

Figura 6.18. Produsele favorite

6.2.9. PAGINA COȘULUI DE CUMPĂRĂTURI

În această pagină puteți ajunge fie din pagina de detalii în momentul în care adăugați un produs în coș, fie din meniu cum este prezentat în Figura 6.19.

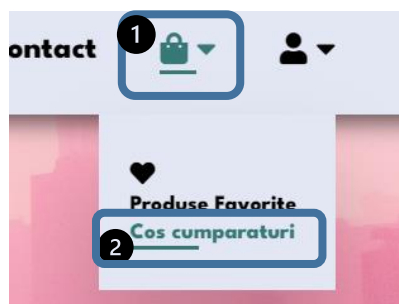






Figura 6.19. Meniu coș

La fel ca și în pagina de produse favorite, se urmează aceiași pași pentru vizualizarea sau ștergerea unui produs. Spre deosebire de pagina anterioară, se poate alege cantitatea dorită și de asemenea dacă doriți să cumpărați produsele din coș, trebuie doar să apăsați pe butonul de *finalizare cumpărături* din partea de jos a paginii (vezi Figura 6.20), acțiune care vă va redirecționa către o altă pagină pentru a completa datele de facturare.

STERGE	IMAGINE	CULOARE	BRAND	DENUMIRE	PRET	MARIME	CANTITATE	SUBTOTAL	DETALII
			Bershka	Bluza B	91 lei	XS	<input data-bbox="1074 1630 1177 1697" type="text" value="1"/>	91 lei	

Nota de Plata	
Subtotal	91
Transport (poate varia in functie de metoda de expediere)	19
Total	110

Finalizare cumparaturi

Figura 6.20. Pagina destinată coșului de cumpărături

6.2.10. PAGINA DE PROFIL

Pentru a ajunge în această pagină să selectați din bara de meniu după cum vă este prezentat în *Figura 6.21*. Din același meniu puteți să vă deconectați de pe site apăsând butonul de deconectare, cel marcat cu roșu.



Figura 6.21. Meniu profil

În această pagină se pot face modificări cu privire la datele de facturare doar printr-un click pe butonul de „Salvare modificări” din partea de jos a *Figurii 6.22*.

Actualizare Profil

Nume si prenume	Nume utilizator
<input type="text" value="Dan Mirabela"/>	<input type="text" value="mira22"/>
E-mail	Numar de telefon
<input type="text" value="mira.theduet@gmail.com"/>	<input type="text" value="0788744880"/>
Judet	Oras
<input type="text" value="Timis"/>	<input type="text" value="Jimbolia"/>
Adresa	Cod postal
<input type="text" value="Str. Republicii, Nr.3"/>	<input type="text" value="305400"/>

Salvare modificari

Figura 6.22. Pagina de editare a profilului

6.2.11. PAGINA DE FINALIZARE COMANDĂ

După cum am precizat și în subcapitolul anterior, veți fi redirecționat către o altă pagină (vezi *Figura 6.23*). Astfel, pentru a finaliza comanda trebuie urmați câțiva pași. Primul pas este acela de a alege metoda de expediere în funcție de preferințele dumneavoastră, următorul pas ar fi alegerea modalității de plată, dacă ați selectat plata „cash” puteți sări peste pasul trei, în caz contrar va trebui să introduceți datele necesare pentru efectuarea plății cu cardul.

În cazul în care nu sunteți sigur că doriți să finalizați comanda puteți să apăsați pe butonul de „Continuă cumpărăturile”, cel evidențiat cu galben în *Figura 6.23*.

Date de facturare

Nume si prenume
Dan Mirabela

Email
mira.theduet@gmail.com

Adresa
Str. Republicii, Nr.3

Oras
Jimbolia

Judet
Timis

Nr. Telefon
0788744880

Cod Postal
305400

Plata

Metode de expediere:
☒ Magazin ☐ Express la domiciliu ☐ Domiciliu ☐ Posta

Modalitati de plata:
☒ Card ☐ Cash

Carduri acceptate
Visa Mastercard

Nume posesor card
Nume Prenume

Numar card
1111 2222 3333 4444

Luna de expirare
09

An de expirare
2024

CVV
352

Cos de cumparaturi

91 lei (XS) x1

Transport
0

Total
91

Continua cumparaturile

Finalizeaza comanda

Figura 6.23. Pagina de finalizare a comenzii

În urma finalizării comenzii veți fi întrebat dacă doriți să descărcați factura, indiferent de alegere aceasta va fi oricum atașată într-un mail care va fi trimis și el la rândul lui după finalizarea comenzii.

6.3. PRINCIPII EURISTICE

În anul 1990, Jakob Nielsen și Rolf Molich au propus zece principii pentru a ajuta la dezvoltarea UI [40]. Principiile euristice ale lui Nielsen sunt generale, adică nu determină reguli specifice de utilizare. În schimb, euristicele sunt reguli generale pe care le poți urma pentru a ajuta la crearea unor produse digitale mai accesibile, mai ușor de utilizat și mai intuitive[40]. Cele 10 principii euristice ale lui Nielsen sunt:

- Principiul 1: Asigurarea vizibilității stării în care se află sistemul
- Principiul 2: Crearea unui model al sistemului care să fie compatibil cu realitatea
- Principiul 3: Controlul și libertatea utilizatorului
- Principiul 4: Consistență și standarde
- Principiul 5: Prevenirea erorilor
- Principiul 6: Recunoaștere mai degrabă decât memorare
- Principiul 7: Flexibilitate și eficiență în utilizare
- Principiul 8: Proiectare estetică și minimalistă
- Principiul 9: Ajutați utilizatorii să recunoască, să diagnosticheze și să revină dintr-o eroare
- Principiul 10: Suport (*Help*) și comunicare

6.3.1. PRINCIPIUL 1 – ASIGURAREA VIZIBILITĂȚII STĂRII ÎN CARE SE AFLĂ SISTEMUL

Vizibilitatea stării sistemului se referă la cât de bine este transmisă starea sistemului utilizatorilor săi. În mod ideal, sistemele ar trebui să țină întotdeauna utilizatorii informați despre ceea ce se întâmplă, printr-un feedback adecvat într-un timp rezonabil [39].

Acest principiu a fost atins prin următoarele funcționalități:

- Evidențierea butonului activ (vezi *Figura 6.24 din secțiunea 1*);
- Validarea câmpurilor în timp real (vezi *Figura 6.24 din secțiunea 3*);
- Afișarea unei iconițe reprezentative în momentul încărcării paginii (vezi *Figura 6.24 din secțiunea 2 marcată cu roșu*).

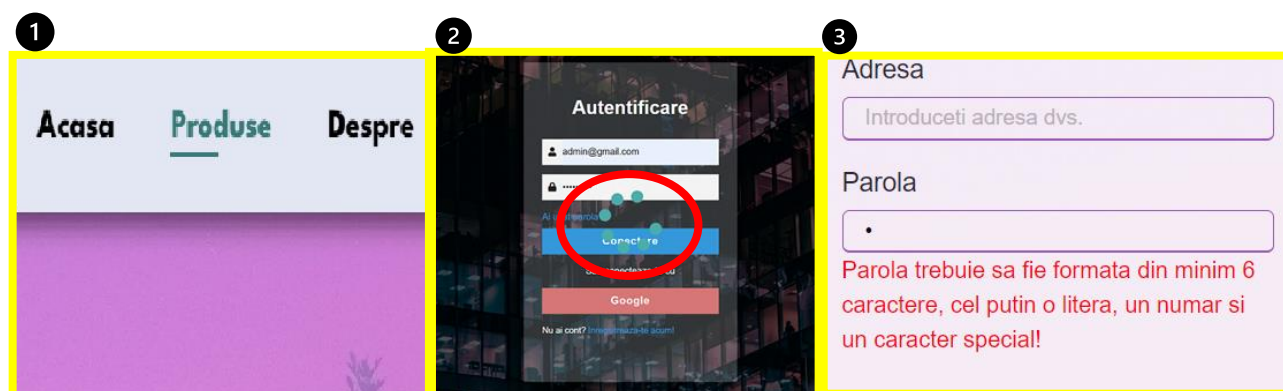


Figura 6.24. Feedback

6.3.2. PRINCIPIUL 2 – CREAREA UNUI MODEL AL SISTEMULUI CARE SĂ FIE COMPATIBIL CU REALITATEA

Sistemul trebuie să comunice cu utilizatorul în limbajul uzual acestuia, folosind fraze, cuvinte și concepte familiare, nu termeni orientați pe sistem. Informația trebuie să apară într-o ordine și urmând o logică "naturală" [39].

Am respectat acest principiu prin utilizarea unui limbaj uzual, folosind fraze, cuvinte și noțiuni familiare, care pot fi ușor de înțeles de către utilizator, astfel acesta se va simți comod și va utiliza informațiile puse la dispoziția lui de pe acest site fără a mai fi nevoit să caute sensul unor anumiți termeni sau în cel mai rău caz, să nu mai folosească site-ul.

6.3.3. PRINCIPIUL 3 – CONTROLUL ȘI LIBERTATEA UTILIZATORULUI

Se întâmplă des ca utilizatorii să selecteze funcții nepotrivite și au nevoie de o "ieșire de urgență" [39]. Aceste ieșiri permit utilizatorului să păstreze controlul asupra sistemului și să evite să rămână blocați și să se simtă frustrați.

Prin utilizarea unui buton de eliminare (vezi *Figura 6.25*), am reușit să respect acest principiu, astfel în pagina de produse, în urma aplicării filtrelor vor fi afișate doar produsele conform opțiunilor alese, ieșirea de urgență o reprezintă butonul de eliminare a filtrelor,

astfel în pagină vor fi afișate toate produsele automat, fără a necesita anularea manuală a fiecărui filtru setat.

The image shows a filter interface with the following elements:

- Ordoneaza dupa** (Sort by): A dropdown menu.
- Disponibilitate** (Availability): A dropdown menu.
- Pret** (Price): Radio buttons for ☐ Pret crescator (increasing) and ☐ Pret descrescator (decreasing).
- Categorie** (Category): A dropdown menu.
- Brand**: Radio buttons for ☐ In stoc (in stock), ☒ Promotii (promotions), and ☐ Toate produsele (all products).
- Brand** (Brand): A dropdown menu.
- Brand**: Radio buttons for ☐ Rochii (dresses), ☒ Bluze (shirts), and ☐ Toate categoriile (all categories).
- Brand**: Radio buttons for ☐ Bershka, ☒ Zara, and ☐ Toate brand-urile (all brands).
- Aplica filtre** (Apply filters) button.
- Elimina filtre** (Remove filters) button.

Figura 6.25. Eliminare filtre

6.3.4. PRINCIPIUL 4 – CONSISTENȚĂ ȘI STANDARDE

Site-urile create pe baza acestui principiu asigură o experiență de utilizare intuitivă pentru clienți, ceea ce face ca aceștia să nu depindă de un manual de utilizare pentru a se putea bucura de serviciile puse la dispoziție. Respectarea acestui principiu este datorată utilizării pictogramelor sugestive: lupa pentru căutare, căruciorul de cumpărături pentru coșul de cumpărături, precum și logo-urile rețelelor de socializare (vezi Figura 6.26).

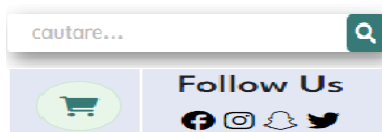


Figura 6.26. Pictograme

6.3.5. PRINCIPIUL 5 – PREVENIREA ERORILOR

Mesajele de eroare sunt importante, dar cea mai bună alegere este de a le evita. Astfel funcționalitățile prin care utilizatorul poate să producă erori sunt dezactivate până îndeplinește toate condițiile necesare de a se angaja în acțiune.

Spre exemplu, în pagina destinată coșului de cumpărături am dezactivat butonul de finalizare cumpărături în momentul în care nu mai există produse în coș pentru a evita să producerea de erori (vezi Figura 6.27).

Nota de Plata		Nota de Plata	
Subtotal	161.95	Subtotal	0
Transport (poate varia in functie de metoda de expediere)	19	Transport (poate varia in functie de metoda de expediere)	19
Total	180.95	Total	-
Finalizare cumparaturi		Finalizare cumparaturi	

Figura 6.27. Dezactivarea butonului

6.3.6. PRINCIPIUL 6 – RECUNOAȘTERE MAI DEGRABĂ DECÂT MEMORARE

Opțiunile, acțiunile și elementele importante trebuie să fie vizibile utilizatorului. Acesta nu ar trebui să fie nevoit să-și amintească informații de la o parte a interfeței la alta.

Astfel, cele mai importante opțiuni au fost afișate în meniu care este unul fix, fiind la dispoziția utilizatorului în orice parte a paginii. De asemenea au fost utilizate iconițe reprezentative pentru profil și coșul de cumpărături (vezi *Figura 6.28*).

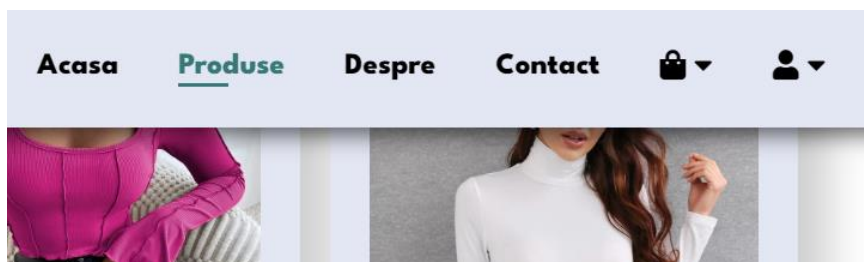


Figura 6.28. Meniu fix

6.3.7. PRINCIPIUL 7 – FLEXIBILITATE ȘI EFICIENȚĂ ÎN UTILIZARE

Site-ul poate fi utilizat atât de utilizatorii experimentați cât și de cei începători. Diferența dintre aceștia este partea de autentificare, deoarece este cel mai probabil ca un utilizator neexperimentat să nu dispună de un cont de *Google*. Astfel, autentificarea o va face utilizând site-ul, chiar dacă nu durează mult acest proces, utilizatorii experimentați sunt scutiți de acest aspect.

6.3.8. PRINCIPIUL 8 – PROIECTARE ESTETICĂ ȘI MINIMALISTĂ

Densitatea informației este redusă punând accent doar pe lucrurile esențiale, excluzând informația irelevantă sau rar necesară.

6.3.9. PRINCIPIUL 9 – AJUTAȚI UTILIZATORUL SĂ RECUNOASCĂ, SĂ DIAGNOSTICHEZE ȘI SĂ REVINĂ DINTR-O EROARE

Utilizatorul poate fi ajutat să revină dintr-o eroare prin mesajele afișate, care ar trebui să fie exprimate într-un limbaj simplu (fără coduri de eroare) și să indice cu precizie problema.

Am îndeplinit acest principiu prin oferirea unor mesaje de eroare precise, care indică problema exactă de generare a erorii (vezi *Figura 6.29*).

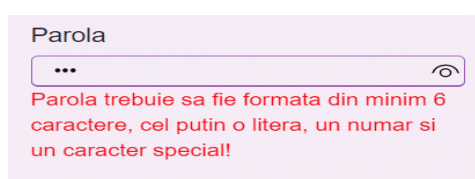


Figura 6.29. Mesaj precis de eroare

6.3.10. PRINCIPIUL 10 – SUPT (HELP) ȘI DOCUMENTARE

Deși este posibil ca sistemul să nu aibă nevoie de o documentație sau o explicație suplimentară, este necesar să o furnizăm pentru a ajuta utilizatorii să înțeleagă cum să-și îndeplinească sarcinile [40].

Astfel, în pagina de contact am introdus o secțiune de Help (vezi *Figura 6.30*) în care sunt afișate câteva întrebări frecvente, ce pot fi de ajutor utilizatorului.

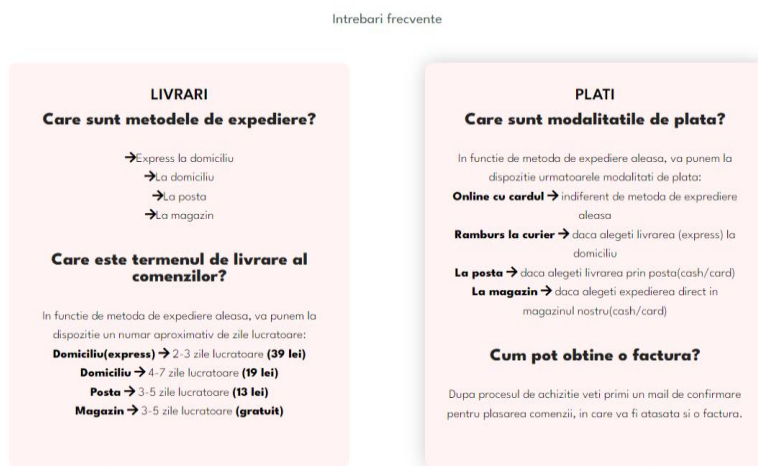


Figura 6.30. Secțiune de Help

În cazul în care răspunsul la întrebările utilizatorului nu se regăsește în această secțiune de *Help*, clientul poate intra în secțiunea de contact pentru a trimite un e-mail către administrator (vezi *Figura 6.31*).

Figura 6.31. Secțiunea de contact

7. CONCLUZII

Proiectul de licență a fost proiectat cu scopul de a oferi utilizatorilor facilitățile de care dispune un magazin de comerț electronic. Astfel, a fost realizat un magazin online funcțional ce comercializează articole vestimentare pentru persoanele de sex feminin, prin intermediul căruia clientul poate plasa comenzi din confortul locuinței și se poate bucura de o gamă largă de produse, având posibilitatea de filtrare a acestora după bunul plac. De cealaltă parte se află administratorul, care are acces la baza de date cu produse pentru a șterge, edita sau adăuga un produs, dar și pentru a gestiona comenzile plasate, având dreptul de a le anula sau trimite spre livrare, moment în care clientul este notificat prin intermediul unui mesaj trimis pe e-mail. De asemenea utilizatorul poate solicita să fie notificat în momentul în care acesta dorește un produs care nu mai există în stoc. Atunci când produsul devine disponibil, clientul care a solicitat notificarea va fi anunțat printr-un e-mail că a revenit pe stoc produsul dorit.

În urma studiului făcut pe baza lucrărilor [10], [11] și [12] descrise în *capitolul 2*, pot afirma că, utilizatorii care folosesc magazinul dezvoltat de mine dispun de mai multe facilități, precum autentificarea în două moduri (fie printr-un cont creat pe site, fie prin intermediul contului de Google), criptarea parolei pentru a stoca în siguranță acreditările utilizatorilor, notificarea acestora în momentul în care un produs dorit revine pe stoc, cât și posibilitatea resetării parolei.

În continuare există posibilități de îmbunătățire și de perfecționare a proiectului implementat. Conținutul site-ului poate fi îmbunătățit prin adăugarea unor articole vestimentare pentru bărbați, cât și produse din alte game, precum cosmetică, încălțăminte, accesorii și lista poate continua. Totodată aș dori să adaug opțiunea de acordare a recenziilor unui produs. Astfel, clientul poate să-și exprime opinia în legătură cu produsul comandat, să acorde o notă și de asemenea, dacă dorește, să atașeze o fotografie cu produsul. Această caracteristică doresc să o fac vizibilă tuturor clienților pentru a putea alege cu ușurință un produs. În ultimă fază aș dori să implementez funcționalitatea de efectuare a plății cu cardul.

Prin dezvoltarea proiectului de licență am reușit să dobândesc noi cunoștințe, deoarece am aprofundat limbajul *JavaScript*, am învățat cum să lucrez cu o bază de date *NoSQL*, dar și cum să integrez sistemul *GIT* în cadrul unui proiect. Aș dori să închei cu faptul că primul meu proiect din cadrul dezvoltării web a fost un succes din punctul meu de vedere, acesta fiind o consolidare a bazei pentru o viitoare carieră în domeniu.

BIBLIOGRAFIE

- [1] Internet: <https://datareportal.com/global-digital-overview>, accesat în iulie 2022
- [2] M. Qadah and S. M. Al-Shomrani, "Teaching web development course in information system department," 2011 3rd International Congress on Engineering Education (ICEED), 2011, pp. 165-168
- [3] Full-stack developer:
<https://bootcamp.learn.utoronto.ca/blog/what-is-a-full-stack-developer/>, accesat în august 2022
- [4] Frontend vs Backend: <https://www.geeksforgeeks.org/frontend-vs-backend/>, accesat în august 2022
- [5] Web Design: <https://www.w3.org/standards/webdesign/htmlcss>, accesat în august 2022
- [6] What is a Database: <https://www.oracle.com/database/what-is-database/>, accesat în august 2022
- [7] SQL vs NoSQL: <https://www.integrate.io/blog/the-sql-vs-nosql-difference/>, accesat în august 2022
- [8] R. Tamimi and M. E. Mohammadpourzarandi, "The application of web usage mining in E-commerce security," 7th International Conference on e-Commerce in Developing Countries:with focus on e-Security, 2013, pp. 1-8
- [9] Web Testing: <https://www.softwaretestinghelp.com/web-application-testing/>, accesat în august 2022
- [10] Prima lucrare asemănătoare:
<https://opus.govst.edu/cgi/viewcontent.cgi?article=1079&context=capstones>, accesat în septembrie 2022
- [11] A doua lucrare asemănătoare:
<https://www.theseus.fi/bitstream/handle/10024/149293/RayThesis1.output.pdf?sequence=1&isAllowed=y>, accesat în septembrie 2022
- [12] A treia lucrare asemănătoare:
https://www.researchgate.net/publication/320701955_ONLINESHOPCOM_AN_E-COMMERCE_WEBSITE, accesat în septembrie 2022
- [13] VS Code: <https://code.visualstudio.com/>, accesat în august 2022
- [14] Live Server:
<https://marketplace.visualstudio.com/items?itemName=ritwickdey.LiveServer>, accesat în august 2022
- [15] Font Awesome Gallery:
<https://marketplace.visualstudio.com/items?itemName=tomasvergara.vscode-fontawesome-gallery>, accesat în august 2022
- [16] Auto Close Tag:
<https://marketplace.visualstudio.com/items?itemName=formulahendry.auto-close-tag>, accesat în august 2022
- [17] IntelliCode:

<https://marketplace.visualstudio.com/items?itemName=VisualStudioExptTeam.vscointellibicode>, accesat în august 2022

[18] GIT: <https://support.nesi.org.nz/hc/en-gb/articles/360001508515-Git-Reference-Sheet>, accesat în august 2022

[19] HTML: P. Skibinski, "Improving HTML Compression," Data Compression Conference (dcc 2008), 2008, pp. 545-545

[20] HTML4 vs HTML5: <https://ipwithease.com/html4-vs-html5/>, accesat în august 2022

[21] D. Mazinianian and N. Tsantalis, "CSSDev: Refactoring Duplication in Cascading Style Sheets," 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C), 2017, pp. 63-66

[22] CSS introduction: <https://www.geeksforgeeks.org/css-introduction/>, accesat în august 2022

[23] Aplicarea CSS în HTML:

https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/How_CSS_is_structured, accesat în august 2022

[24] Introducere în JavaScript: <https://www.geeksforgeeks.org/introduction-to-javascript/>, accesat în august 2022

[25] Biblioteca smtp.js:

<https://netcorecloud.com/tutorials/how-to-send-emails-with-javascript/>, accesat în august 2022

[26] Biblioteca jsPDF: <https://blogs.perficient.com/2015/04/29/jspdf-in-web-portal/>, accesat în august 2022

[27] Firebase Realtime Database: <https://firebase.google.com/docs/database>, accesat în august 2022

[28] Reguli de securitate: <https://firebase.google.com/docs/database/security>, accesat în august 2022

[29] Fișiere JSON: <https://www.json.org/json-en.html>, accesat în august 2022

[30] XML vs JSON: <https://www.devopsschool.com/blog/comparison-between-xml-vs-json-vs-yaml/>, accesat în august 2022

[31] Firebase Authentication: <https://firebase.google.com/docs/auth>, accesat în august 2022

[32] P. Mahanto, S. K. Barisal and D. P. Mohapatra, "Achieving MC/DC using UML Communication Diagram," 2018 International Conference on Information Technology (ICIT), 2018, pp. 73-78

[33] Types of UML diagrams: <https://creatly.com/blog/diagrams/uml-diagram-types-examples/>, accesat în august 2022

[34] Metoda set: <https://firebase.google.com/docs/database/web/read-and-write>, accesat în august 2022

[35] Google Authentication:

<https://firebase.google.com/docs/auth/web/google-signin>, accesat în august 2022

[36] Firebase Authentication:

<https://firebase.google.com/docs/reference/js/v8/firebase.auth.Auth#getRedirectResult>, accesat în august 2022

[37] Metoda get: <https://firebase.google.com/docs/database/web/read-and-write>, accesat în august 2022

[38] Snapshot:

<https://firebase.google.com/docs/reference/node/firebase.database.DataSnapshot>,
accesat în august 2022

[39] Principii euristice: [10 Usability Heuristics for User Interface Design \(nngroup.com\)](https://nngroup.com/articles/10-usability-heuristics-for-user-interface-design/) ,
accesat în iulie 2022

**DECLARAȚIE DE AUTENTICITATE A
LUCRĂRII DE FINALIZARE A STUDIILOR ***

Subsemnatul MUNTYAN MIHAELA-CRĂCIUNELA

legitimată cu C.I. seria TZ nr. 484116

CNP 6001214352149

autorul lucrării MAGAZIN WEB PENTRU ARTICOLE VESTIMENTARE

elaborată în vederea susținerii examenului de finalizare a studiilor de LICENȚĂ organizat de către Facultatea AUTOMATICĂ ȘI CALCULATOARE din cadrul Universității Politehnica Timișoara, sesiunea SEPTEMBRIE 2022 a anului universitar 2021-2022, coordonator AS. DRD. ING. STELIAN NICOLA, luând în considerare conținutul art. 34 din Regulamentul privind organizarea și desfășurarea examenelor de licență/diplomă și disertație, aprobat prin HS nr. 109/14.05.2020 și cunoscând faptul că în cazul constatării ulterioare a unor declarații false, voi suporta sancțiunea administrativă prevăzută de art. 146 din Legea nr. 1/2011 – legea educației naționale și anume anularea diplomei de studii, declar pe proprie răspundere, că:

- această lucrare este rezultatul propriei activități intelectuale,
- lucrarea nu conține texte, date sau elemente de grafică din alte lucrări sau din alte surse fără ca acestea să nu fie citate, inclusiv situația în care sursa o reprezintă o altă lucrare/alte lucrări ale subsemnatului.
- sursele bibliografice au fost folosite cu respectarea legislației române și a convențiilor internaționale privind drepturile de autor.
- această lucrare nu a mai fost prezentată în fața unei alte comisii de examen de licență/diplomă/disertație.

Timișoara,

Data

05.09.2022

Semnătura

Muntyan

* Declarația se completează „de mână” și se inserează în lucrarea de finalizare a studiilor, la sfârșitul acesteia, ca parte integrantă.