UNIVERSITATEA ALEXANDRU IOAN CUZA IAȘI

**FACULTATEA DE INFORMATICĂ**



LUCRARE DE LICENȚĂ

**BuyCheap**

propusă de

**Anton Dragoș-Răzvan**

**Sesiunea**: iulie, 2019

Coordonator științific

**Colab. Olariu Florin**

**UNIVERSITATEA ALEXANDRU IOAN CUZA IAȘI**

**FACULTATEA DE INFORMATICĂ**

**BuyCheap**

**Anton Dragoș-Răzvan**

**Sesiunea**: iulie, 2019

Coordonator științific

**Colab. Olariu Florin**

Avizat,

Îndrumător Lucrare de Licență

Titlul, Numele și prenumele \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Data \_\_\_\_\_\_\_\_\_\_\_\_ Semnătura \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**DECLARAȚIE privind originalitatea conținutului lucrării de licență**

Subsemntatul(a) ………………………………………………………………………………………

domiciliul în …………………………………………………………………………………………………..

născut(ă) la data de ………………..…., identificat prin CNP ………….……………..………………..., absolvent(a) al(a) Universității „Alexandru Ioan Cuza” din Iași, Facultatea de ………………………. specializarea …………………………………………………………, promoția …………………………., declar pe propria răspundere, cunoscând consecințele falsului în declarații în sensul art. 326 din Noul Cod Penal și dispozițiile Legii Educației Naționale nr. 1/2011 art.143 al. 4 si 5 referitoare la plagiat, că lucrarea de licență cu titlul: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_elaborată sub îndrumarea dl. / d-na \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, pe care urmează să o susțină în fața comisiei este originală, îmi aparține și îmi asum conținutul său în întregime.

De asemenea, declar că sunt de acord ca lucrarea mea de licență să fie verificată prin orice modalitate legală pentru confirmarea originalității, consimțind inclusiv la introducerea conținutului său într-o bază de date în acest scop.

Am luat la cunoștință despre faptul că este interzisă comercializarea de lucrări științifice in vederea facilitării fasificării de către cumpărător a calității de autor al unei lucrări de licență, de diploma sau de disertație și în acest sens, declar pe proprie răspundere că lucrarea de față nu a fost copiată ci reprezintă rodul cercetării pe care am întreprins-o.

Dată azi, ………………………… Semnătură student …………………………

DECLARAȚIE DE CONSIMȚĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „*BuyCheap*”, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de testetc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza” din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Iași, *data*

Absolvent *Anton Dragoș-Răzvan*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura în original)

**Cuprins**

[Introducere 5](#_Toc12316641)

[Contribuții 7](#_Toc12316642)

[Capitolul 1: Descrierea problemei 8](#_Toc12316643)

[Capitolul 2: Abordări anterioare 9](#_Toc12316644)

[Capitolul 3:Descrierea soluției 10](#_Toc12316645)

[3.1 Tehnologii Folosite 10](#_Toc12316646)

[3.2 Arhitectura aplicației 12](#_Toc12316647)

[3.2.1 Diagrama C4 nivel 1 12](#_Toc12316648)

[3.2.2 Diagrama C4 nivel 2 13](#_Toc12316649)

[3.2.3 Diagrama de clase 14](#_Toc12316650)

[3.3 Implementarea aplicației pe partea de back-end 15](#_Toc12316651)

[3.4 Implementarea aplicației pe partea de front-end 23](#_Toc12316652)

[3.5 Manualul de utilizare al aplicației 28](#_Toc12316653)

[Concluziile lucrării 31](#_Toc12316654)

[Bibliografie 33](#_Toc12316655)

**Introducere**

BuyCheap este o aplicație web care permite căutarea unui produs în scopul comparării prețului acestuia pe site-urile eMag, Flanco și Altex. Am ales această temă deoarece de fiecare dată când cumpărăm un produs trebuie să intrăm pe fiecare din aceste trei site-uri și să verificăm noi înșine care dintre ele ne oferă cel mai avantajos preț, iar acest lucru este o pierdere de timp.

Un astfel de site există deja în România, iar acesta este [www.compari.ro](http://www.compari.ro) , însă pe acest site se compară prețul produsului de pe 200 alte site-uri partenere. Un număr mai mare de site-uri, deși pare o idee bună, îngreunează probabilitatea de a vedea ofertele disponibile pe cele mai mari site-uri de e-commerce din România. De multe ori nu putem avea încredere în anumite site-uri mai ales dacă nu sunt prea cunoscute sau dacă au multe reviewuri negative așa că multă lume preferă să comande lucruri de pe niște site-uri mai mari. Faptul că am ales doar aceste trei site-uri îmi permite să adaug posibilitatea de a vedea dacă obiectul pe care utilizatorul îl caută poate fi găsit în orașul in care se află, fapt care va aduce ceva în plus aplicației mele față de cea de pe [www.compari.ro](http://www.compari.ro) . De asemenea, aplicația BuyCheap permite utilizatorilor să adauge anumite produse la lista de favorite și să aleagă să fie notificați când prețul unui produs favorit scade sau când un produs apare în stoc la unul din cele trei magazine.

Pentru a crea această aplicație am folosit Jsoup pentru a prelua elemente html de pe un anumit link, dar Altex, Flanco și eMag introduc elementele html care corespund produselor prin cod javascript iar acest lucru face ca jsoup să nu găsească toate elementele pe care le putem vedea la un anumit link de pe cele trei site-uri. Pentru a rezolva această problemă am folosit Selenium împreună cu chromeDriver prin care mai întâi deschid linkul căutat într-o fereastră nouă din Google Chrome, acest fapt permite codului javascript de la linkul respectiv să fie rulat, iar apoi folosesc jsoup pe această fereastră pentru a putea prelucra toate elementele html de pe acea pagină. După preluarea produselor de pe cele trei site-uri și introducerea lor în aplicația mea folosesc un algoritm prin care să fac comparația intre cele mai asemănătoare produse găsite de pe cele trei site-uri pentru ca rezultatele afișate pe site-ul aplicației mele să fie cât se poate de exacte.

**Metodologia**

Temelia dezvoltării acestei aplicații a constituit-o prima discuție cu profesorul meu coordonator prin care am stabilit împreună ce tehnologii aș putea să folosesc pentru a dezvolta BuyCheap și, mai ales, ce ar putea aduce nou această aplicație pe lângă ce există deja.

După ce am hotărât ce voi folosi pentru dezvoltarea aplicației și ce ar trebui să poată face aceasta am început dezvoltarea părții de back-end. Primul pas a fost să învăț cum să folosesc Spring Boot. Am construit baza de date inițială a aplicației, pe care am modificat-o de două ori deoarece am mai avut nevoie de anumite coloane în unele tabel sau un anumit tip de date al unei coloane necesita să fie schimbat cu alt tip, și am conectat-o la aplicația mea.

Următorul pas a fost crearea de obiecte DTO pentru a putea separa clasele ce reprezintă instanțe din baza de date de clasele pe care aplicația le expune către front-end. Acest pas a presupus și crearea de servicii de mapare între un obiect din baza de date și un obiect DTO. După ce am terminat cu tot ceea ce ține de baza de date am  creat  servicii care implementează funcționalitățile necesare aplicației.

Am început crearea serviciilor cu cele mai puțin complexe implementări, precum serviciul UserService în care am implementat logarea și crearea unui cont nou. După ce am creat toate serviciile pe care știam deja cum să le implementez a urmat documentarea pentru serviciul de parsare de date de pe un site web. Implementarea acestui serviciu a presupus mai întâi studierea modului în care pot folosi librăria JSOUP pentru a parsa un document HTML. Odată ce am reușit să folosesc această librărie și să extrag date de pe un anumit site a urmat documentelor HTML în care erau afișate rezultatele căutării unui produs pe cele trei site-uri (Emag, Altex și Flanco) pentru a putea selecta ce elemente din document îmi sunt folositoare pentru a afișa produsele găsite în aplicația mea. Deoarece prin analizarea modului în care apar produsele pe aceste trei site-uri am observat că se folosesc scripturi care aduc produse pe site a urmat căutarea unei metode prin care pot parsa un document HTML abia după ce se rulează toate scripturile. Am descoperit cum pot folosi ChromeDriver pentru a deschide un link într-o nouă fereastră din Google Chrome pentru a putea parsa pagina de la linkul respectiv abia după ce toate scripturile și-au terminat execuția. După implementarea tuturor serviciilor am creat controllere prin care să expun aceste funcționalități la anumite endpointuri.

Următorul pas a fost crearea unei aplicații Angular în care să construiesc o interfață  și în care să folosesc serviciile pe care le-am expus în Spring. Am început prin crearea de componente pentru a crea designul paginii web iar apoi am creat servicii în care fac requesturi la serviciile din back-end pentru a aduce date și ale integra în designul paginii.

**Contribuții**

Există deja pe piață o aplicație web care ajută utilizatorii să compare prețul produselor de pe foarte multe site-uri, iar aceasta este Compari.ro . Această pagină web folosește, pentru a compara prețuri, peste 200 de site-uri partenere. Numărul foarte mare face ca utilizatorii să poată vedea o mulțime de oferte la fiecare produs existent pe piață. Acest site este foarte cunoscut și are deja o mulțime de utilizatori. Așadar scopul aplicației BuyCheap este de a aduce ceva nou pe lângă ceea ce ne oferă deja Compari.ro și să rezolve anumite inconveniențe pe care unii utilizatori le pot avea folosind acest site.

Pentru a aduce ceva în plus și pentru a oferi utilizatorilor măcar un motiv bun pentru care ar schimba o aplicație cunoscută și foarte folosită în România, BuyCheap aduce două mari noutăți față de ceea ce ne oferă deja site-ul menționat mai sus. Una dintre aceste două funcționalități o reprezintă faptul că fiecare utilizator are o listă de favorite în care poate adăuga orice produs găsește în urma unei căutări efectuate în aplicație. Acest lucru permite utilizatorului să urmărească mai ușor aceste produse, nefiind nevoit să le caute de fiecare dată când vrea să le vadă. O a doua funcționalitate importantă a aplicației este reprezentată de notificarea utilizatorilor, prin e-mail, atunci când prețul unui produs aflat în lista lor de favorite are un preț mai mic decât atunci când acesta a fost marcat ca și favorit. Pe lângă aceste două funcționalități, BuyCheap funcționează pe principiul că mai mult nu înseamnă întotdeauna mai bine, așa că datele despre produsele care sunt afișate în această aplicației provin de la doar 3 magazine online. Degeaba putem vedea produse de pe 200 de site-uri dacă despre majoritatea dintre acestea nu am auzit niciodată nimic și ne chinuim să găsim rezultate doar de la magazinele online deja foarte cunoscute din România care și-au dobândit încrederea a o mulțime de utilizatori. Așadar,. BuyCheap caută produse la trei dintre cele mai mari magazine online din România, Emag, Flanco și Alex. Afișând rezultate doar de la aceste trei magazine, Aplicația BuyCheap oferă utilizatorilor siguranța faptului că pot alege orice ofertă este afișată fără să fie stresați de faptul că ar putea cumpără de la o companie care nu le oferă siguranța necesară.

În concluzie, contribuția mea este de a duce mai departe ușurința pe care Compari.ro o oferă în compararea de prețuri ale unui produs prin posibilitatea de a marca produse ca și favorite pentru a le urmări mai ușor și prin notificare utilizatorilor când un produs din lista lor de favorite are un preț mai mic decât cel pe care îl știau ei.

**Capitolul 1**

**Descrierea problemei**

Motivul care m-a determinat să aleg să implementez această aplicație web este faptul că niciodată nu putem fi siguri dacă am obținut un produs, pe care tocmai l-am cumpărat, la cel mai bun preț de pe piață.

În prezent există deja o mulțime de magazine online de unde putem cumpăra diverse produse, iar numărul acestora crește pe zi ce trece. Tot mai multe persoane aleg să își cumpere produsele dorite în mediul online deoarece nu mai este necesar să ne deplasăm undeva pentru a obține un produs, acesta fiind livrat la reședința noastră, și deoarece avem posibilitatea dea a vedea care este prețul aceluiași produs pe mai multe site-uri. Aceste două motive sunt principalele beneficii aduse de cumpărarea de produse prin mediul online. Deși aceste două beneficii ne scutesc de timp și de drumuri inutile, ele pot fi îmbunătățite și mai mult.

Căutarea unui produs pe mai multe site-uri pentru a vedea care dintre acestea oferă cel mai bun preț necesită răbdare și timp. De asemenea, și faptul că suntem nevoiți să verificăm în mod frecvent aceste site-uri pentru a vedea dacă produsul pe care ni-l dorim este la reducere pe vreunul dintre aceste magazine online consumă și mai mult timp. Deși am putea să ne facem cont pe anumite site-uri de magazine online și să selectăm să fim notificați atunci când apare o reducere ar putea fi o soluție bună, aceasta necesită la rândul ei multă răbdare și pe lângă notificările pe care le așteptăm vom primi multe alte notificări prin care aceste site-uri încearcă să își promoveze vânzarea altor produse.

Aplicația BuyCheap încearcă să rezolve toate aceste inconveniențe și să economisească timpul utilizatorilor acesteia. Această aplicație ne permite să căutăm un produs iar ca și rezultat al căutării primim exact ceea ce am găsi pe trei dintre cele mai mari magazine de vânzări online (Emag, Altex, Flanco) doar că toate aceste rezultate vor fi pe același site, lucru ce va facilita o mai ușoară comparare a prețurilor. O altă funcționalitate a acestei aplicației o reprezintă rezolvarea inconvenienței de a căuta zile la rând același produs pentru a vedea dacă acesta este la reducere. Aplicația permite utilizatorilor să adauge un produs la lista lor de favorite. Acest lucru permite aplicației să caute din două în două ore și să verifice dacă vreunul din produsele adăugate la lista de favorite, indiferent de furnizorul de la care provin, are acum un preț mai mic decât atunci când utilizatorul a marcat produsul ca și favorit. Utilizatorul va primi o notificare prin e-mail în care i se va transmite ce produs din lista sa de favorite are un preț mai mic acum, care este prețul și ce furnizor oferă acest preț. Implementarea acestor funcționalități fac ca aplicația BuyCheap să rezolve principalele inconveniențe aduse de cumpăratul online.

**Capitolul 2**

**Abordări anterioare**

O abordare anterioară a problemei descrise în capitolul 3 o reprezintă aplicația web Compari.ro. Dezvoltatorii acestui site sunt cei care au avut cel mai mare succes din România în a oferi o rezolvare pentru problema pe care o presupune căutarea de produse online pentru a putea cumpăra un produs la cel mai bun preț disponibil.

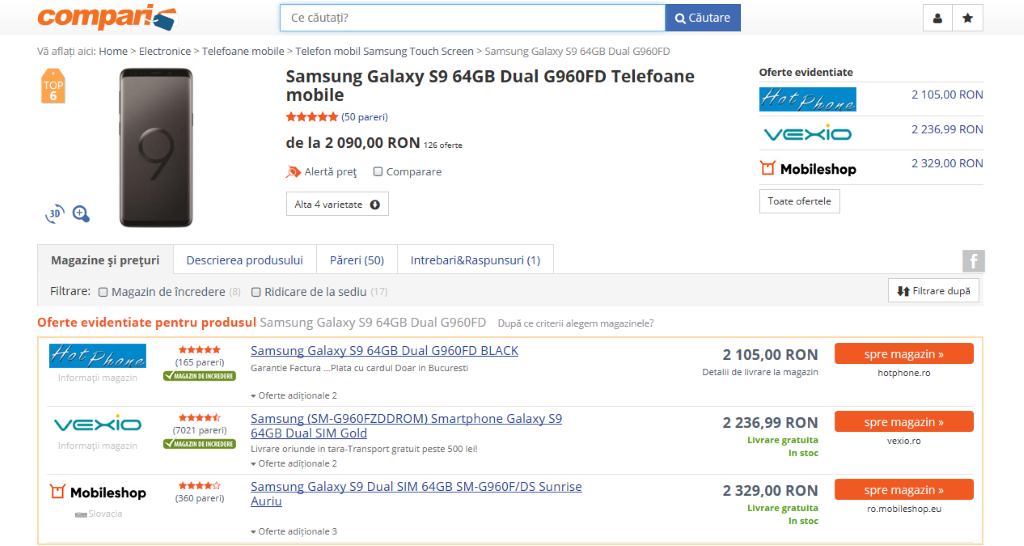
****

figura 1

Site-ul Compari.ro oferă o pagină web cu un design simplu și foarte intuitiv, așa cum se poate observa în figura 1, prin care utilizatorii pot căuta un produs pentru a vedea ofertele oferite pentru acel produs de o mulțime de magazine online din România.  Acest site a reușit să economisească timpul pierdut de mulți oameni care erau nevoiți să acceseze cât mai multe site-uri pentru a putea descoperi de unde ar putea cumpăra un produs la un preț cât mai bun.

Rezolvarea celor ce au implementat a fost făcută prin încheierea de parteneriate cu peste 200 de site-uri de vânzări online. Așadar acest site permite utilizatorilor să caute un produs într-un singur loc și să primească ofertele disponibile de la fiecare site, care conține acel produs, dintre partenerii lor.

Dezvoltatorii acestui site au ales să aducă rezultate de pe cât mai multe site-uri pentru a le oferi utilizatorilor o multitudine de magazine din care să poată alege. Această abordare a problemei a atras o mulțime de utilizatori și au făcut ca site-ul Compari.ro să fie cel mai folosit site de comparare de produse din România.

**Capitolul 3**

**Descrierea soluției**

**3.1 Tehnologii folosite**

Pentru dezvoltarea aplicației BuyCheap au fost folosite următoarele unelte și tehnologii:

* Angular
* Java Spring Boot
* Selenium-WebDriver
* Hibernate
* PostgreSql
* Jsoup

**Angular**

Angular este un framework de dezvoltare web cu sursă deschisă bazat pe limbajul TypeScript. Aceste framework este dezvoltat de Google.

Principalele funcționalități**:**

* Componente**:** ajută la construirea unei aplicații prin mai multe module lucru care duce la o mai ușoară si mai bună mentenanță a aplicației pe o perioadă lungă de timp
* Folosirea limbajului TypeScript**:** un limbaj open-source creat și menținut de Microsoft. Typescript este un super set al Javascript.
* Servicii:sunt bucăți de cod care pot fi împărtășite de mai multe componente ale unei aplicații

**Java Spring Boot**

Este un framework care are la bază frameworkul Spring . Acesta permite o modalitate mai ușoară pentru setarea și rularea aplicațiilor web.

**Selenium-WebDriver**

Este un API creat pentru a ajuta la testarea paginilor web dinamice, ale căror elemente dintr-o pagină se pot schimba fără ca pagina să fie reîncărcată. Selenium-WebDriver face apeluri directe către browser folosind suportul native pentru automatizare al fiecărui browser.

**Hibernate**

Este un framework ORM pentru aplicații Java. Termenul ORM(Object Relational Mapping) se referă la o tehnică de mapare a datelor de la un model de reprezentare a unui obiect la un model relațional de reprezentare a datelor. Hibernate are grijă să mapeze clase Java la tabele din baza de date iar tipurile de date din Java le mapează la tipuri de date din SQL.

**PostgreSql**

Este o bază de date obiect-relațional care folosește și extinde limbajul SQL și stochează și scalează in mod sigur date. PostgreSql și-a câștigat reputația datorită arhitecturii, fiabilității și integrității sale. PostgreSql funcționează pe majoritatea sistemelor de operare majore și respectă principiile ACID.

ACID este un set de proprietăți pentru tranzacțiile din baze de date care intenționează să garanteze validitate chiar și în cazul apariției unei erori sau în cazul unei căderi de curent.

Principiile ACID:

* Atomicitate:tranzacțiile sunt formate adesea din mai multe secvențe dar atomicitatea asigură că fiecare tranzacție este tratată ca o singură unitate iar tranzacția fie are loc complet, fie nu are loc deloc.
* Consistență:asigură faptul că tranzacțiile pot trece baza de date doar de la o stare validă la o altă stare validă păstrând și respectând toate constrângerile și regulile de la nivelul bazei de date.
* Izolare:adesea tranzacțiile au loc în mod concurent iar principiul izolării asigură că tranzacțiile au loc fără să interfereze una cu cealaltă.
* Durabilitate:garantează faptul că odată ce o tranzacție are loc complet, modificările făcute de aceasta sunt stocate și menținute chiar și în cazul unei erori de sistem.

**Jsoup**

Este o librărie Java open-source creată pentru a parsa, a extrage și a manipula date stocate în documente HTML. Hypertext Markup Language (HTML) este limbajul standard pentru documente create cu scopul de a fi afișate în browser.

**3.2 Arhitectura aplicației**

**3.2.1 Diagrama c4 nivel 1**

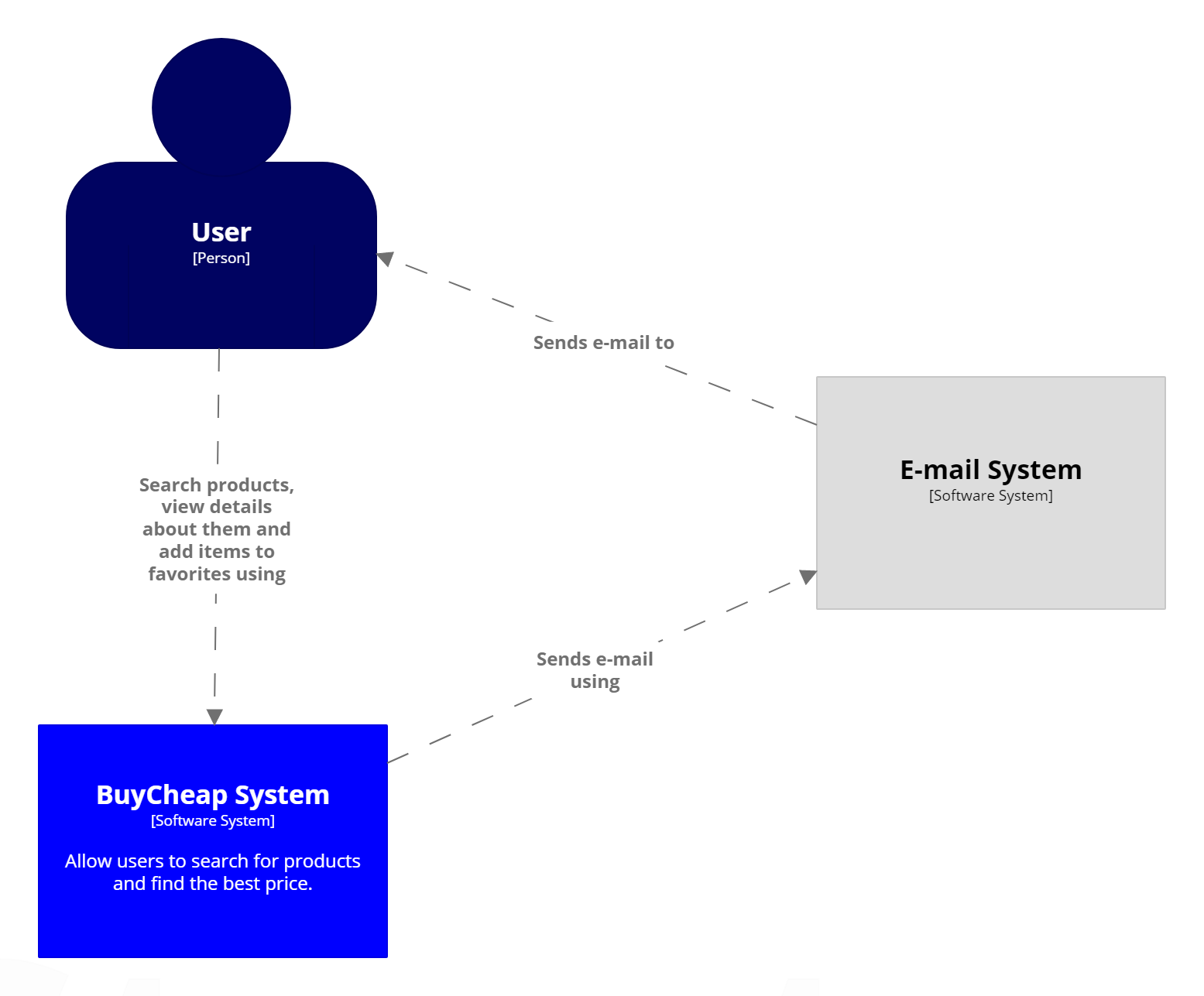


figura 2

Figura 2 este diagrama de context a sistemului aplicației (diagrama de nivel unu din modelul c4). Această diagramă nu se axează pe detalii despre aplicație ci pe imaginea de ansamblu a aplicației. În această diagramă apar doar utilizatorii și sistemele software.

Un utilizator care intră pe aplicația BuyCheap caută produse, vede detalii despre produse și le adaugă la favorite folosind sistemul BuyCheap. Sistemul BuyCheap folosește un sistem de e-mail pentru a trimite emailuri către utilizatori.

**3.2.2 Diagrama c4 nivel 2**

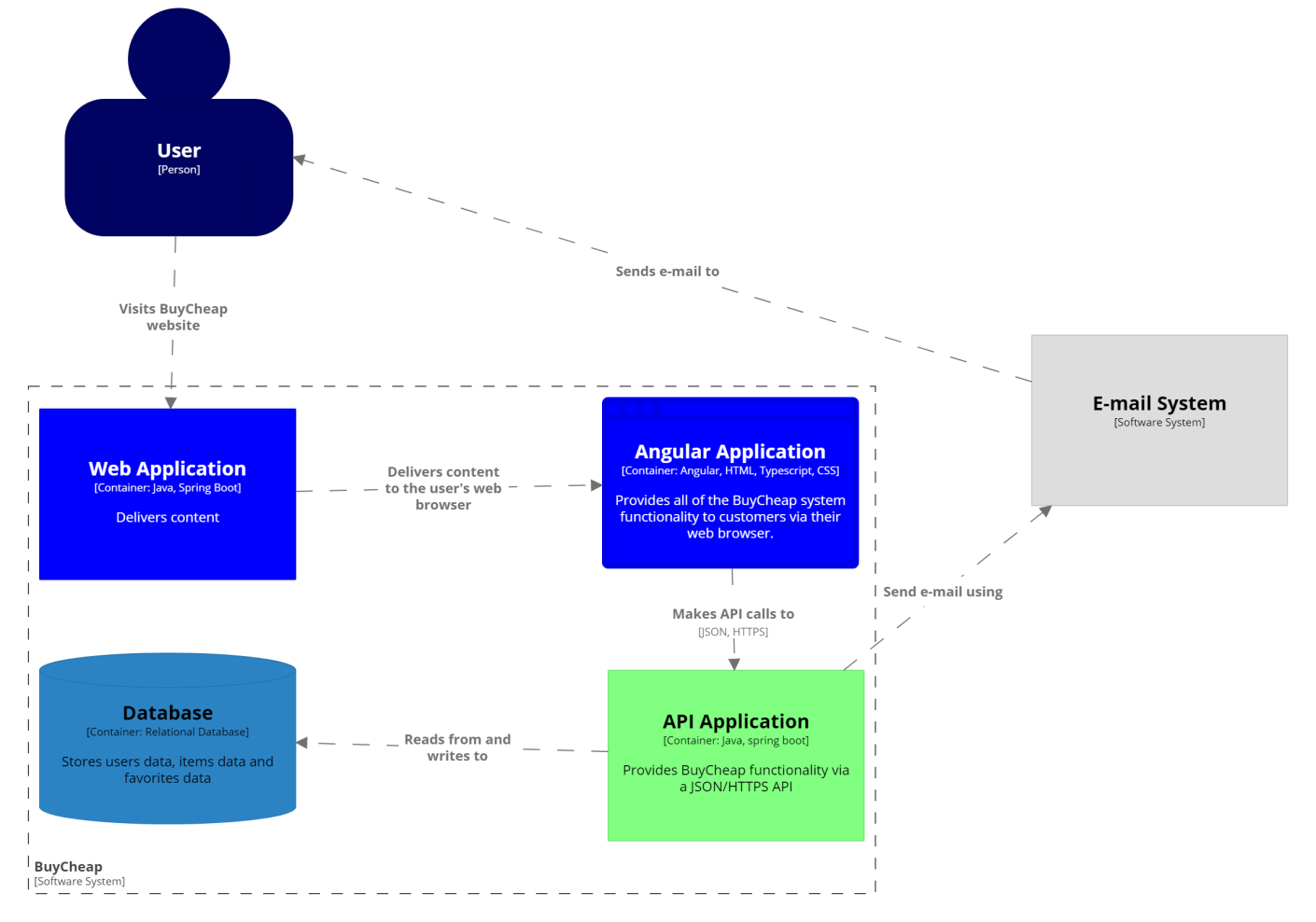


figura 3

Figura 3.2 reprezintă diagrama de containere a aplicației (diagrama de nivel doi din modelul c4). Această diagramă arată cum sunt distribuite sarcinile în arhitectura aplicației. Aici apar majoritatea tehnologiilor folosite.

Atunci când un utilizator accesează site-ul aplicației, rezultatele îi sunt livrate de către aplicația Angular în care este construită partea de front-end a aplicației. Pentru ca Angular să poată afișa rezultatele căutate de utilizator pe pagina aplicației, va face apeluri către API-ul aplicației.

În API sunt expuse metode la anumite endpointuri așa încât componenta Angular să le poată accesa. API-ul este creat în limbajul Java folosind Spring boot și este un RESTful API. Un RESTful API este bazat pe arhitectura REST, aceasta este un stil arhitectural care folosește

Requesturi HTTP pentru a lua , a posta, a pune și a șterge date. Modificările asupra bazei de date pot fi făcute doar prin intermediul API-ului.

Pentru a trimite un e-mail către un utilizator, API-ul se folosește de un sistem software numit E-mail System.

**3.2.3 Diagrama de clase**

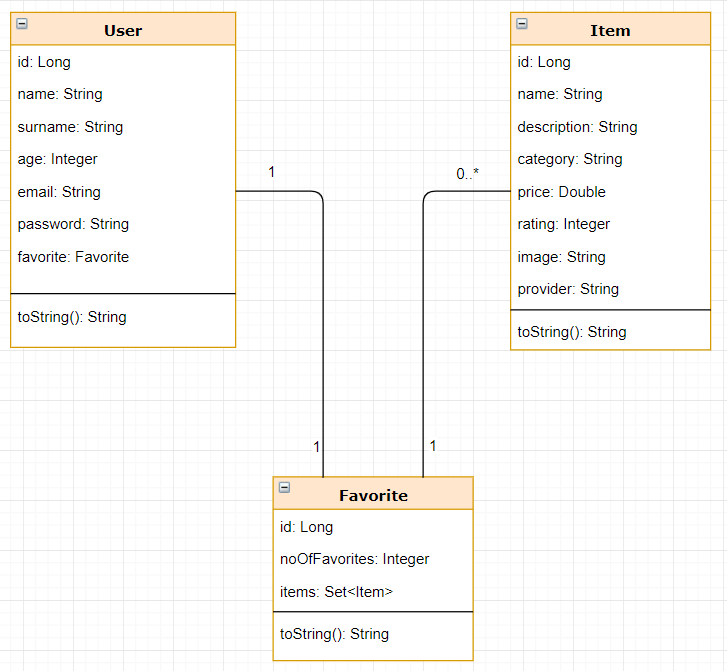
****

figura 4

În figura 4 clasa User cuprinde detaliile necesare despre un utilizator (nume, prenume, etc.) și pe lângă acestea are și un câmp “favorite” care reprezină o instanță a clasei Favorite. Așadar, fiecărui utilizator îi corespunde o instanță a clasei Favorite, rlația dintre acestea fiind una de tiplul one-to-one.

Clasa Favorite cuprinde o listă de elemente de tipul Item, un id și numărul de produse favorite. Această clasă este este folosită pentru a oferi utilizatorului posibilitatea de a marca anumite produse ca și favorite, acest lucru având ca scop facilitarea unei mai ușoare modalități de urmărire a evoluției prețului unui produs.

Clasa Item cuprinde detalii despre un anumit produs (denumire, descriere, preț, etc.).

**3.3 Implementarea aplicației pe partea de back-end**

Aplicația folosește Spring boot pentru partea de back-end. Pentru maparea claselor la tabele din baza de date aplicația folosește Hibernate iar baza de date este PostgreSql. Pentru a expune funcționalitățile de pe back-end aplicația folosește Angular pentru partea de front-end.

Aplicația conține 4 controlere:

* UserController
* ItemController
* FavoriteController
* ScrapingController

În fiecare controller sunt expuse câteva endpointuri unde se pot face requesturi, fiecare endpoint reprezentând o funcție din controller:

* La endpointul „http://localhost:8080/users” este expusă funcția de adăugare a unui user.

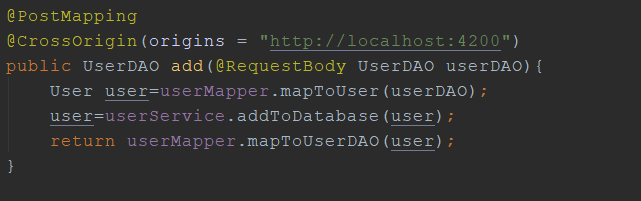


figura 5

Prin adnotarea “@CrossOrigin” este permis accesul adresei <http://localhost:4200> către acest endpoint. Această funție adaugă în baza de date userul primit ca parametru prin apelul funcției “addToDatabase(user)” din serviciul UserService. Fiecare controller folosește cel puțin un serviciu pentru a putea apela metodele care implementează funcționalitățile necesare aplicației.

Aplicația folosește data acces object (DAO) pentru a separa entitățile din baza de date de entitățile pe care le folosesște în API. Așadar, pentru fiecare tabel din baza de date aplicația are o clasa DAO în care fie expune câmpuri în plus față de cele din tabel, fie nu expune anumite câmpuri din baza de date. Pentru a transpune obiectele care reprezintă tabele în obiecte DAO aplicația folosește servicii de mapare. De exemplu pentru clasa User avem serviciul UserMapper care prezintă două funcții:

* Funcția de mapare de la User la UserDAO
* Funcția de mapare de la UserDAO la User

**Serviciul pentru scraping de date**

Data scraping este procesul prin care o aplicație parcurge elementele de pe o pagină web cu scopul de a extrage date de pe acel site.

Pentru această aplicație a fost necesară implementarea a trei funcții care extrag date de pe trei site-uri la fiecare căutare făcută de către un utilizator. Prima funcție este cea care parcurge o pagină de pe site-ul Emag și extrage produsele care se găsesc pe aceasta.

Prima etapă necesară pentru extragerea datelor despre produsul căutat, pe site-ul Emag de exemplu, este transformarea stringului căutat de către utilizator într-un link la care se află rezultatele căutării acelui string pe site-ul Emag.

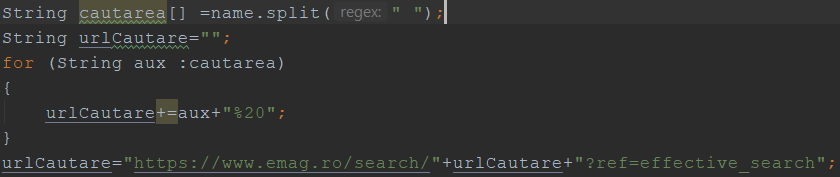


figura 6

După stabilirea linkului la care se află rezultatele căutării executate de către utilizator nu se poate începe extragerea datelor de la acel link deoarece produsele de pe site-ul Emag sunt adăugate pe pagină în mod dinamic, așadar extragerea datelor în acest moment nu va aduce toate elemente de pe pagină. Pentru a rezolva această problemă următorul pas este setare și deschiderea unei instanțe ChromeDriver în care vom accesa linkul .

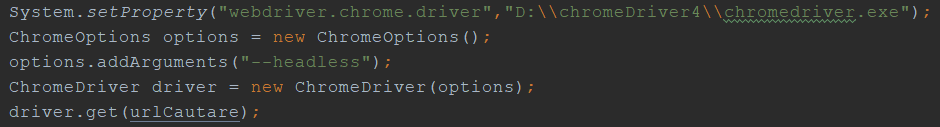


figura 7

Argumentul ”—headless” din figura 7 face ca pagina să fie accesată fără a deschide în Google Chrome o fereastră cu conținutul acestei pagini. Acest argument este necesar deoarece face ca datele să fie extrase mai rapid deoarece elimină timpul de așteptare necesar pentru a deschide interfața grafică a browserului Google Chrome.

Odată executat acest pas putem parsa pagina care se află acum în variabila “driver” deoarece acum aceasta va conține toate elementele care apar la linkul căutat. Pentru a accesa detaliile despre produsele care au fost găsite funcția trebuie să parcurgă elementele html din această pagină și să extragă doar datele necesare despre produse.

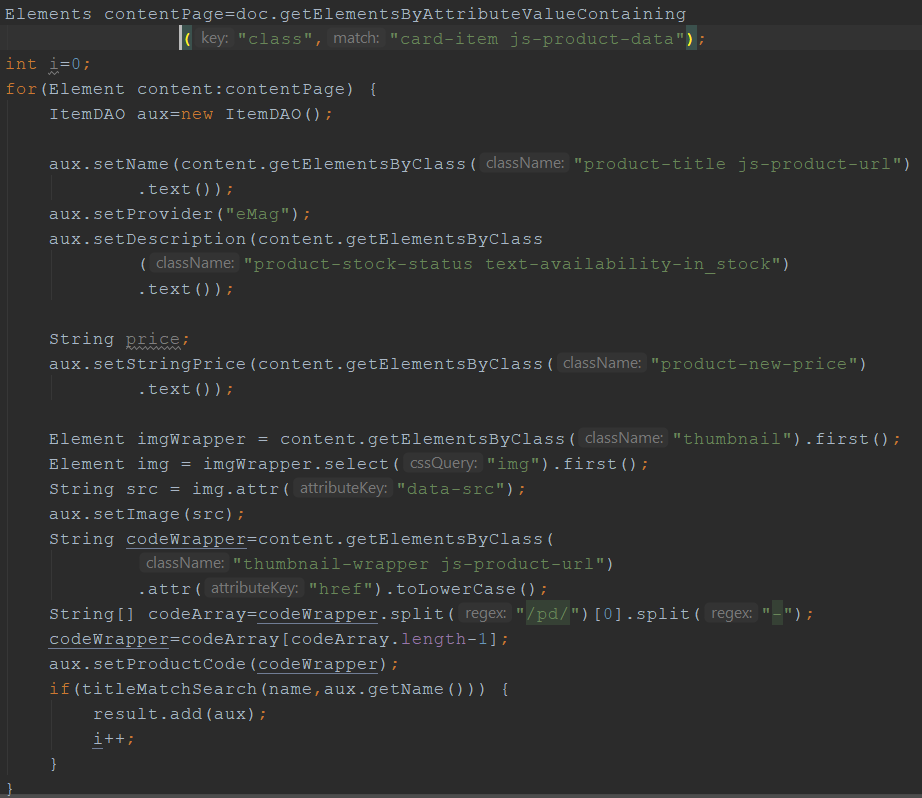


figura 8

Codul din figura 8 parcurge elementele html de la pagina parsată de pe site-ul Emag folosindu-se de numele clasei atribuite elementelor . Fiecare element din lista “contentPage” reprezintă un card în care se află detalii despre un produs care se află pe pagină. Pentru a extrage date despre produs, precum numele, prețul și codul am studiat codul HTML de la pagina parsată și am căutat tagurile care conțin informațiile de care aveam nevoie:

- În tag-ul cu clasa “product-title” se află titlul produsului

- În tag-ul cu clasa “product-new-price” se află prețul curent al produsului, după aplicarea tuturor reducerilor

- În tag-ul cu clasa “product-stock-status” se află detalii despre disponibilitatea produsului

- În elementul cu clasa “thumbnail” se găsește un element img care conține ca și valoare a atributului “data-src” un link către imaginea produsului

- În elementul cu clasa “thumbnail-wrapper js-product-url” se găsește ca și valoare a atributului “href” un url în care putem găsi codul unic pentru Site-ul Emag al unui produs, de aceea după ce este obținut acest url este parcurs și este extrasă doar o anumită porțiune.

Pentru fiecare dintre cele trei magazine online (Emag, Flanco și Altex) există o metodă în diferită în serviciul de data scraping în care am parcurs tagurile html existente la pagina pe care se află rezultatele căutării utilizatorului. Sunt necesare trei metode diferite deoarece pentru fiecare site a fost nevoie să studiez codul HTML și să caut tagurile în care se află detalii despre produsele găsite.



figura 9

Codul din figura 9 parcurge elementele html de la pagina parsată de pe site-ul Flanco într-un mod asemănător cu parcurgerea făcută pe site-ul Emag, dar tagurile HTML în care se află informații de care aplicația BuyCheap are nevoie sunt diferite:

- În tag-ul cu clasa “produs-title” se află titlul produsului

- În tag-ul cu clasa “price” se află prețul

- În tag-ul cu clasa “product -status” găsim disponibilitatea produsului

- În elementul cu clasa “favorite” se află un tag HTML de tipul “<a>” care are ca valoare a atributului “data-item-id” un cod care identifică în mod unic un produs de pe site-ul Flanco

- În elementul cu clasa “product-new-link” se află un tag de tipul “<img>” care are ca valoare a atributului “data-src” sau a atributului “src” un link către imaginea acelui produs



figura 10

Codul din figura 10 parcurge elementele html de la pagina parsată de pe site-ul Altex într-un mod asemănător cu parcurgerea făcută pe site-ul Emag, dar tagurile HTML în care se află informații necesare aplicației mele sunt diferite:

- În tag-ul cu clasa “Product-name” găsim titlul produsului

- În tag-ul cu clasa “Product-name” găsim de asemenea la valoarea atributului “href” un link care cuprinde o porțiune în care se află un cod care identifică în mod unic elementele de pe site-ul Altex

- În tag-ul cu clasa “Status” se găsește disponibilitatea produsului

- În elementul cu clasa “Price-int” se află prețul

- În elementul care are ca și clasă “Product-photo” apare un link către imaginea produsului ca și valoare a atributului “src”

Așadar, din cauza acestor diferențe este necesară implementarea a trei metode diferite pentru a putea extrage date de pe cele trei site-uri. De asemenea, încă o diferență între aceste trei metode o constituie modul în care este construit linkul de pe care se extrage documentul HTML pentru parsare deoarece fiecare site are modul său prin care reprezintă caracterul spațiu dintr-un link .

Aceste trei funcții folosesc o metodă comună prin care titlul produselor găsite este verificat pentru a vedea dacă conține stringul căutat de utilizator.

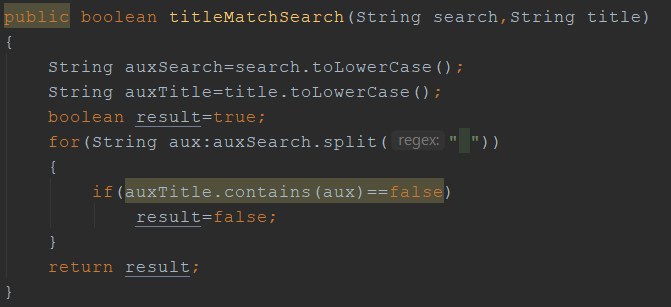


figura 11

Implementarea acestei metode a fost necesară deoarece în urma studierii rezultatelor găsite de către cele trei magazine am observat că fiecare afișează, pe lângă produsul căutat, și alte produse similare sau compatibile cu cel pe care îl căutam. Acest lucru a dus la multe rezultate diferite oferite de cele două site-uri iar această funcție diminuează numărul acestora și duce la o comparare mai bună a rezultatelor.

**Serviciul pentru adăugarea unui produs la lista de favorite**

Înainte ca produsele să fie afișate către utilizator spre a putea fi adăugate la lista de favorite produsele sunt adăugate în baza de date.

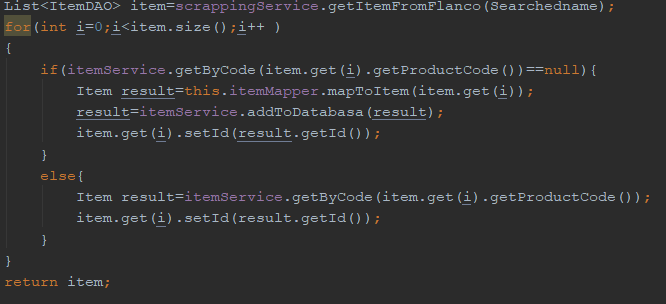


figura 12

Codul din figura 12 se ocupă de adăugarea unei liste de produse în baza de date și de returnarea unei liste de produse care va fi trimisă spre front-end. Înainte ca un produs din listă să fie adăugat în baza de date se verifică dacă există deja un asemenea produs, caz în care adăugarea în baza de date nu se mai face și se aduce produsul deja existent.

Odată ce produsele au fost aduse și afișate către utilizator, acesta are posibilitatea să adauge unul sau mai multe produse la lista sa de favorite. Pentru această funcționalitate am creat serviciul “FavoriteService” în care, printre altele, am implementat o metoda “addItemToList”, cea din figura 13, care se ocupă de asta.

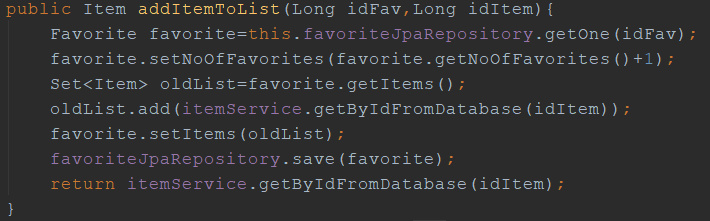


figura 13

**Serviciul de trimitere a e-mailurilor**

Pentru a trimite un e-mail am folosit librăria “javax” și m-am conectat la serverul “Google SMTP”. Acest server este un serviciu, oferit de către Google tuturor deținătorilor unui cont de Gmail, care permite trimiterea unui e-mail folosind un cont de Gmail.

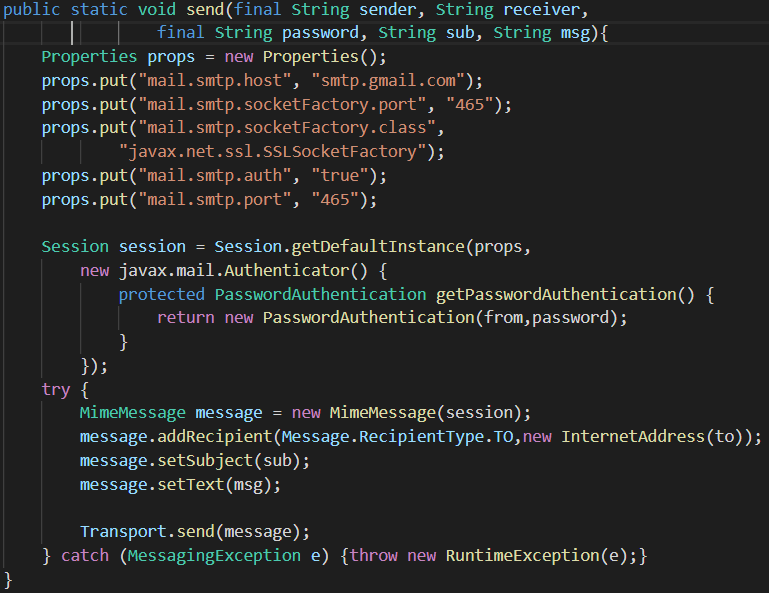


figura 14

În această funcție, din figura 14, am adăugat proprietățile necesare pentru a folosi serviciul oferit de Google folosind contul meu personal de Gmail. După setarea acestor proprietăți am creat o sesiune și am trimis un e-mail care are ca subiect un string primit ca parametru și ca mesaj, de asemenea, un text primit ca parametru.

Această funcție este folosită de serviciu “FavoriteService” pentru a trimite e-mailuri către utilizatorii aplicației atunci când un produs din lista lor de favorite are un preț mai mic decât avea atunci când a fost adăugat la favorite. Pentru a realiza acest lucru aplicația face data scraping pe linkul produselor din două în două ore și verifică dacă prețul afișat pe pagina de la acel link este mai mic decât prețul pe care îl are produsul în baza de date a aplicației, iar dacă da trimite un e-mail către utilizator în care îl anunță că un produs dorit de el are un preț mai mic acum și îi transmite care este mai exact produsul, care este noul preț și la ce furnizor a apărut acest preț.

**3.4 Implementarea aplicației pe partea de front-end**

Pentru a crea front-endul aplicației am folosit Angular. Interfața aplicației este împărțită în șapte componente iar acestea sunt:

1. create-account-page
2. favorites
3. header
4. home-page
5. login-page
6. search-result
7. user-profile

Aproape fiecare din aceste componente folosește anumite servicii create în Angular pentru a face requesturi către endpointurile expuse în Spring. Serviciile pe care le-am creat și le-am folosit în Angular sunt: favoriteService, itemService, scrapingService și userService.

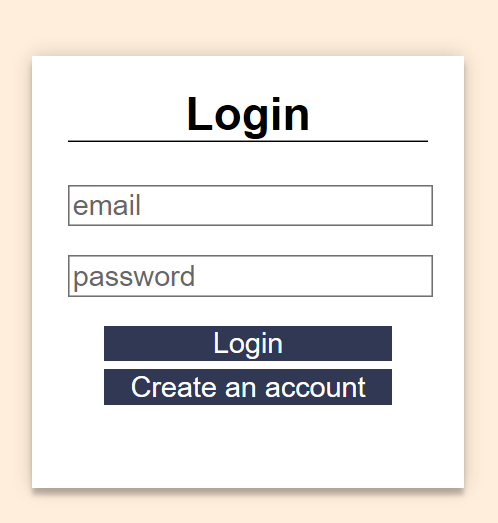
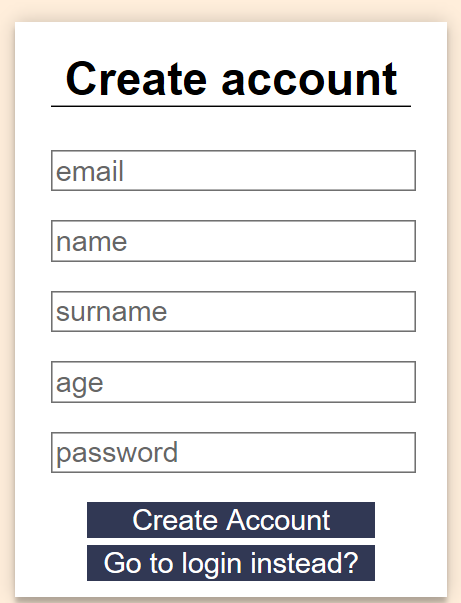
 

figura 15 figura 16

În momentul în care un utilizator accesează pagina web a aplicației, acesta trebuie să se autentifice sau să își creeze un cont nou, pentru fiecare dintre aceste acțiuni utilizatorul va trebui să completeze un formular (figura 15, respectiv figura 16).

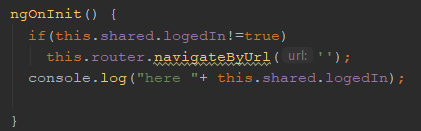


figura 17

Încercarea de a accesa orice altă pagină a aplicației, în afară de pagina de logare și pagina de creare cont, va trimite utilizatorul înapoi către pagina de login atât timp cât acesta nu s-a logat. Am implementat această constrângere, în funcția ngOnInit a fiecărei componente (așa cum se poate vedea în figura 17), deoarece funcționalitățile noi cu care vine această aplicație față de cele existente deja, aici mă refer la posibilitatea de a adăuga un produs la lista de favorite și notificarea prin e-mail când unul din aceste produse este la reducere, au nevoie de un cont de utilizator pentru a putea funcționa. Pentru a verifica dacă există un utilizator logat fiecare componentă apelează serviciul de distribuire a datelor, numit “shared-data”, unde există o variabilă booleană care are valoarea true doar atunci când există un utilizator logat.

După ce utilizatorul se loghează acesta este redirecționat către pagina Home a aplicației unde apare o scurtă descriere despre ceea ce oferă site-ul, iar detaliile despre utilizatorul logat sunt preluate din baza de date făcând un request către un endpoint expus în Spring.

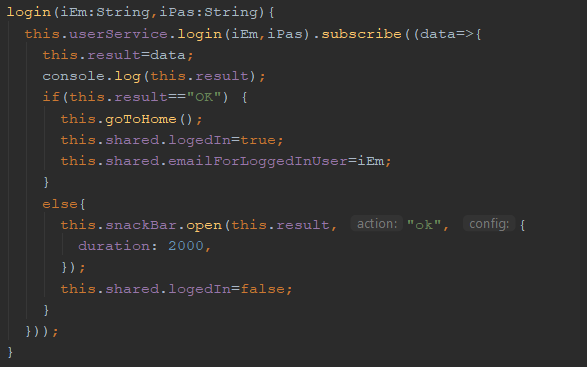


figura 18

Porțiunea de cod din figura 18 face ca, după ce logarea a avut succes, e-mailul utilizatorului să fie păstrat într-un serviciu în care sunt menținute date pentru a putea fi folosite de toate componentele aplicației ( acest serviciu se numește shared-data). Prin memorarea e-mailului utilizatorului logat, putem obține detalii despre utilizator făcând un request către o funcționalitate din Spring care returnează un obiect de tipul User căutând în baza de date utilizatorul al cărui e-mail a fost dat ca parametru ( poate exista un singur cont pentru o adresă de e-mail).

Pentru a căuta un produs, utilizatorul trebuie să acceseze pagina de căutare (către aceasta există un link în header). Odată ajuns pe pagina search-result utilizatorul trebuie doar să introducă numele obiectului pe care dorește să îl caute și apoi să apese pe butonul de căutare. În acel moment este apelată funcția “search” din fișierul “search-result.component.ts” (figura 19).

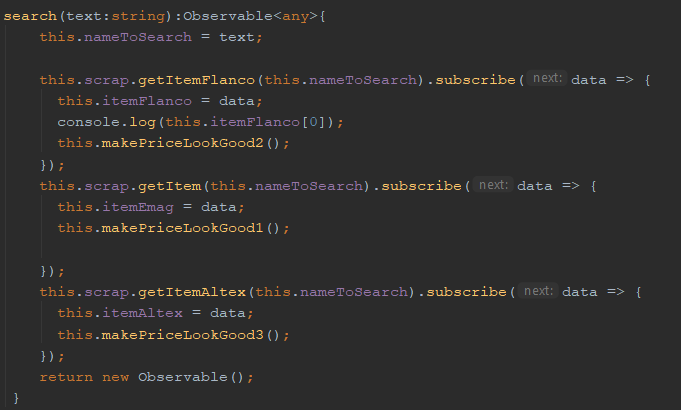


figura 19

Funcția search, din figura 19, apelează metodele din serviciul scraping-service care fac cereri către endpointurile din Spring care expun funcționalitățile pentru căutarea produselor pe site-urile celor trei furnizori. Aceste metode returnează un observabil iar apelurile din funcția search așteaptă răspunsul tuturor celor trei funcții apelate iar apoi salvează listele de produse primite.

Fiecare din aceste trei liste reprezintă produsele aduse de la unul din cei trei furnizori. Datorită faptului că rezultatele obținute în urma căutărilor unui produs pe site-urile celor trei furnizori diferă destul de mult este necesar ca aplicația să se ocupe de aranjarea celor trei liste pentru a putea avea o comparare mai ușoară a produselor. Pentru a aranja aceste trei liste aplicația parcurge lista produselor găsite pe site-ul Emag și caută în celelalte două liste care este produsul cel mai apropiat de produsul la care ne aflăm cu parcurgerea. Această căutare se face pe baza titlului întreg al produsului (acesta conține atât brandul, modelul și tipul produsului cât și anumite detalii despre acesta precum culoarea, memoria internă sau mărimea). Pentru a implementa această căutare a celui mai apropiat titlu de titlul la care ne aflăm cu parcurgerea listei Emag am folosit algoritmul lui Levenshtein. Până să ajung la acest algoritm am încercat să fac algoritm de punctare bazat pe numărul de cuvinte din titlul unui produs din lista Emag care se regăsesc și în titlul unui produs de la Flanco, respectiv Altex. Pentru fiecare produs din prima listă încercam să găsesc cel mai apropiat titlu de la fiecare din celelalte două liste și să aranjez toate cele trei produse pe același rând, dar acest algoritm nu funcționa perfect.

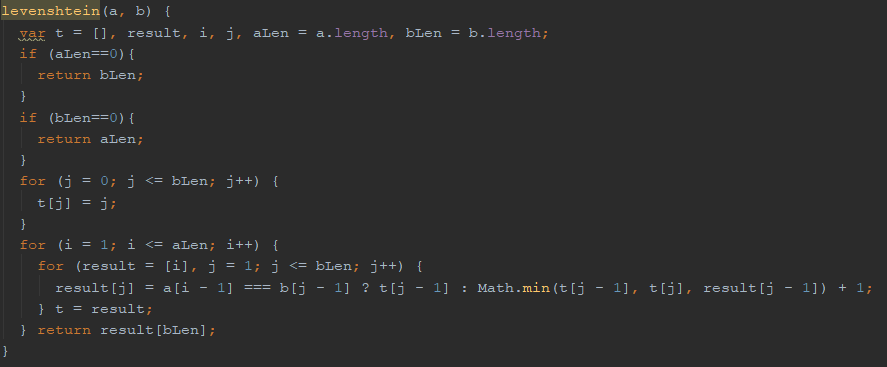


figura 20

sursă: https://rosettacode.org/wiki/Levenshtein\_distance

Algoritmul din figura 20 primește ca și parametri două stringuri și caută numărul minim de operații necesare pentru a face ca cele două stringuri să fie egale. Am studiat acest algoritm până l-am înțeles și l-am implementat în aplicația mea. Folosesc această funcție pentru a vedea distanța dintre două titluri de produse iar apoi memorez această distanță iar după calcularea tuturor distanțelor dintre titlurile de la produsele din lista Emag și titlurile produselor de la Flanco, respectiv Altex, aleg produsele cu distanța cea mai mică față de produsul de la Emag și le așez pe aceeași linie din rezultatele afișate.

După aranjarea celor două liste, aplicația va afișa în trei coloane produsele de la fiecare dintre cele trei magazine online. Atât timp cât un produs există la toți cei trei furnizori, el va fi afișat pe aceeași linie din cele trei coloane, după cum se poate vedea în figura 21.



figura 21

Pentru a adăuga un produs selectat de utilizator la lsita sa de favorite aplicația va apela funcția “addToFav” din serviciul search-result.

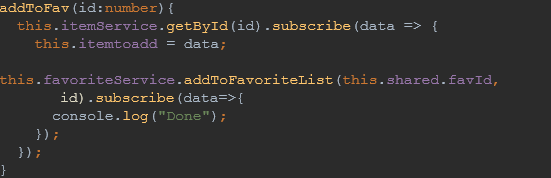


Figura 22

Această funcție, din figura 22, caută în baza de date produsul de pe care a fost apelată funcția, pe baza id-ului acestuia și îl memorează în serviciu iar apoi apelează metoda de adăugare a unui produs la o listă de favorite, care se află în serviciul itemService și care primește ca și parametrii id-ul produsului și id-ul listei de favorite, iar acestă funcție face un request la un endpoint expus în spring în care este implementată metoda de adăugare a unui produs la o listă de favorite.

**3.5 Manualul de utilizare al aplicației**

Când utilizatorul accesează site-ul aplicației acesta este direcționat către pagina de autentificare unde va trebui să își introducă adresa de e-mail și parola. Dacă utilizatorul greșește atunci când completează unul dintre aceste două câmpuri, în partea de jos a site-ului va apărea un mesaj care îi va transmite de ce autentificarea nu are loc, așa cum se poate vedea în figura 23i și 23ii .

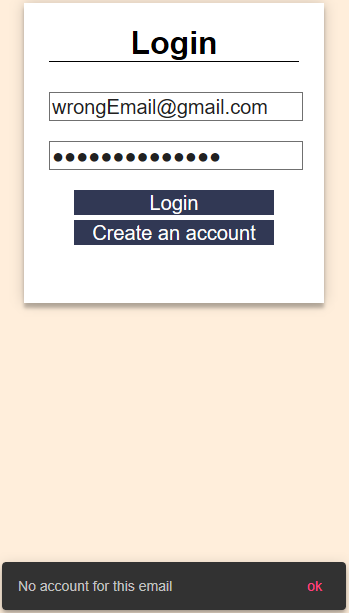
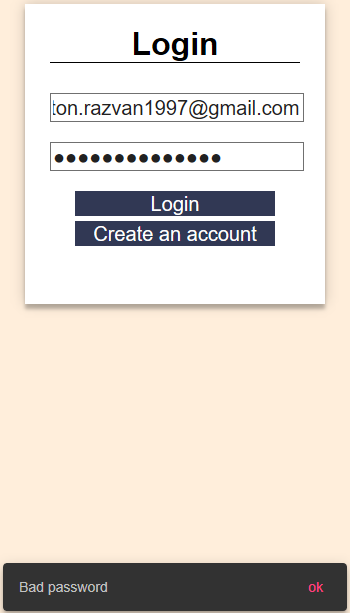
 

figura 23i figura 23ii

Dacă utilizatorul nu are deja un cont, acesta are posibilitatea de a apăsa pe butonul “Create Account” și va fi redirecționat pe pagina prin care își poate crea un cont. Pe această pagină va găsi formularul din figura 16. Odată completat acest formular în mod corect, utilizatorul va trebui să se autentifice. Dacă utilizatorul introduce un e-mail care există deja atunci când încearcă să își creeze un cont, va apărea un mesaj care va exprima acest lucru, așa cum se poate vedea în figura 24.

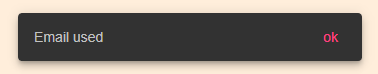


figura 24

După ce utilizatorul a reușit să se autentifice acesta va fi redirecționat pe pagina de Home a aplicației de unde își va putea accesa pagina de profil sau pagina de căutare. Inițial, pe pagina de căutare utilizatorul va găsi doar un input box și un buton (figura 25).

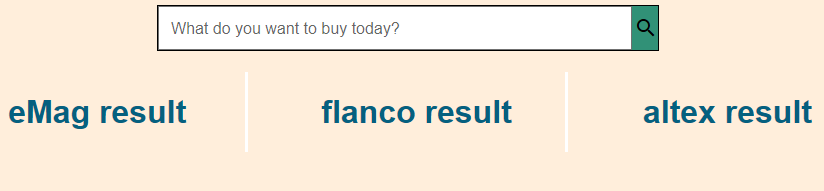


figura 25

La câteva secunde după ce acesta va introduce numele unui produs pe care dorește să îl vadă și apasă butonul de efectuare a căutării, pe pagină vor apărea rezultatele căutării sale precum în figura 21. Odată ce utilizatorul poate vedea lista produselor găsite acesta are posibilitatea de a adăuga unul sau mai multe dintre acestea la lista sa de favorite

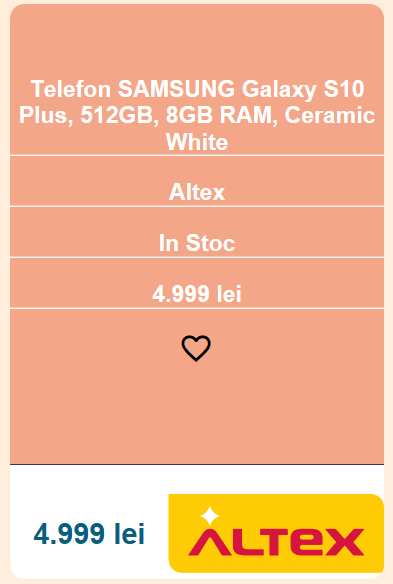
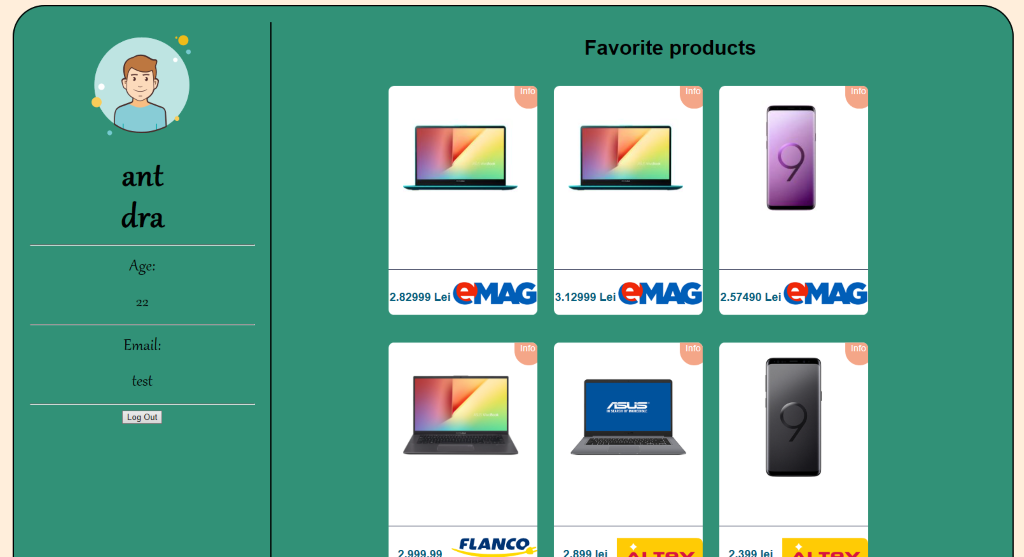


figura 26

Pentru a adăuga un produs la lista de favorite utilizatorul trebuie să ducă cursorul pe secțiunea “info” din interiorul cardului unui produs iar aceasta se va expanda precum în figura 26. În această secțiune va apărea o inimă care este de fapt butonul de adăugare la favorite.

O ultimă pagină pe care o poate accesa utilizatorul este pagina sa de profil, figura 27 reprezentată în Angular de componenta user-profile. În această pagina sunt afișate detalii despre utilizatorul logat (nume, prenume, vârsta și adresa de e-mail) și produsele pe care acesta le-a marcat ca favorite având ca scop primirea de notificări atunci când unul dintre aceste produse are un preț mai mic deât cel pe care l-a avut când a fost adăugat la lista de favorite.

De asemenea, utilizatorul are posibilitatatea de a șterge anumite produse din lista sa de favorite. Acest lucru este posibil în pagina de profil a utilizatorului. Pentru a șterge un obiect din listă, clientul autentificat în aplicație trebuie să ducă cursorul pe sectțiunea info a unui obiect aflat în lista lui de produse marcate ca și favorite iar atunci, la fel ca în figura 26, această secțiune se va expanda și în aceasta va apărea un buton cu textul “Remove from favorites” în locul butonului în formă de inimă care apărea atunci când utilizatorul dorea să adauge un produs la lista sa de favorite.

figura 27

Sursele de proveniență a produselor afișate pe această pagină vor fi verificate din două în două ore de către un serviciu implementat în spring care va verifica dacă s-a modificat prețul vreunuia dintre aceste produse. În cazul în care prețul unui produs de pe această pagină a scăzut, utilizatorul va primi o notificare prin e-mail prin care i se va transmite care produs are acum un preț mai mic, care este acel preț și la ce furnizor a fost modificat prețul. Tot prin intermediul paginii sale de profil utilizatorul are posibilitatea de a se deloga din aplicație.

**Concluziile lucrării**

Crearea acestei aplicații a însemnat o provocare destul de mare pentru mine deoarece am fost nevoit să învăț noi tehnologii de unul singur și să caut metode prin care să rezolv anumite impedimente specifice care au apărut pe parcursul implementării acestei aplicații. Cea mai importantă tehnologie pe care a trebuit să învăț să o folosesc a fost Java Spring boot iar cel mai mare impediment pe care l-am avut în dezvoltarea aplicației este reprezentat de serviciul de scrapping de date pentru care a trebui să studiez timp îndelungat site-urile celor trei furnizori ai aplicației.

Așa cum mi-am propus atunci când am ales acest proiect am reușit să implementez funcționalitățile care fac ca aplicația BuyCheap să inoveze piața aplicațiilor care compară prețuri în România. Aplicația oferă utilizatorilor atât posibilitatea de a adăuga un produs la lista lor de favorite cât și sistemul de notificare prin e-mail atunci când prețul unui favorit scade, acestea două fiind funcționalitățile de care multe persoane care doresc să compare prețuri într-un mod cât mai ușor au nevoie. Consider că aceste funcționalități constituie un mare atu al acestei aplicații și datorită acestora, aplicația BuyCheap a imbunătățit ceea ce oferă deja cel mai mare site de comparare de prețuri din România.

În ceea ce privește direcția viitoare a aplicației, consider că aplicația poate fi îmbunătățită prin dezvoltarea de noi funcționalități care să atragă și mai mulți utilizatori. O funcționalitate ce ar putea fi adusă aplicației este posibilitatea de a căuta produse pri fotografierea lor. De multe ori vedem un produs la o altă persoană pe care nu o cunoaștem și am vrea să aflăm mai multe lucruri despre acesta și chiar să vedem de unde și la ce preț îl putem achiziționa. Implemenarea unei posibilității de a căuta un produs prin simpla fotografiere a acestuia ar face ca aplicația BuyCheap să fie cu mult deasupra tuturor celorlalte aplicații de acest gen care există pe piață. Desigur că o asemenea idee îndrăzneață necesită mult timp și multă muncă, dar nu este imposibil de pus în practică. Aplicația I-ar putea cere utilizatorului să introducă cât mai multe detalii despre acel produs pe care l-a fotografiat și l-a introdus spre a fi căutat pe site pentru a diminua timpul căutării. Un serviciu care să caute prin toate produesele de pe un site pentru a compara imaginea lor cu imaginea introdusă de utilizator ar rula mult prea mul timp, dar dacă utilizatorul introduce și o informație precum faptul că produsul este un telefon ar diminua exponențial timpul de căutare.

O altă funcționalitate ce ar putea fi adusă aplicației ar fi studierea evoluției prețurilor tuturor produselor de la cei trei furnizori pe o perioadă îndelungată de timp, acest lucru având ca scop posibilitatea ca aplicația să estimeze în mod cât mai exact cum va evolua prețul unui produs și în viitor. Acest lucru ar putea ajuta utilizatorul să își dea seama dacă se merită să cumpere acum produsul pe care și-l dorește sau dacă ar trebui să mai aștepte o perioadă.

De asemenea, pe lângă aceste îmbunătățiri, site-ului BuyCheap i-ar prinde bine și o aplicație pe partea de mobile. O astfel de aplicație ar fi foarte folositoare atunci când un utilizator se află într-un magazin și se gândește să cumpere un produs pe care l-a văzut acolo dar nu înainte să se asigure că nu plătește cum mult peste prețul pieței pentru acel produs.

Opinia mea este că aplicația web BuyCheap a îndeplinit deja ceea ce și-a propus și a fost dezvoltată să facă întocmai ceea ce am vrut să ca aceasta să poată face de la bun început. BuyCheap oferă deja funcționalități pe care niciun alt site de compararea de prețuri din România nu le oferă și rezolvă problemele pe care le-am găsit la celelalte aplicații. Deși aplicațiile existente deja oferă produse de la o mulțime de site-uri, acestea nu dau posibilitatea utilizatorului să vadă cât mai ușor produsele care îl interesează, printr-o listă de favorite, și nu notifică utilizatorii cu privire la reducerea prețurilor din această listă.

În concluzie, această aplicație nu numai că deja oferă noi funcționalități în căutarea de produse și compararea de prețuri online, mai are potențialul de a se dezvolta enorm în viitor prin implementarea funcționalităților descrise mai sus.

**Bibliografie**

* https://angular.io/tutorial
* http://chromedriver.chromium.org/getting-started
* https://spring.io/guides/gs/rest-service/
* https://rosettacode.org/wiki/Levenshtein\_distance
* https://mvnrepository.com/
* https://medium.com/@nydiarra/secure-a-spring-boot-rest-api-with-json-web-token-reference-to-angular-integration-e57a25806c50
* https://www.freakyjolly.com/angular-6-7-adding-toastr-messages-in-angular-4-web-application-in-few-steps/#more-2060
* https://material.angular.io/guide/getting-started
* https://www.tutorialspoint.com/angular\_material7/angular\_material7\_snackbar.htm
* https://material.angular.io/components/snack-bar/overview
* https://www.seleniumhq.org/projects/webdriver/
* https://www.seleniumhq.org/docs/03\_webdriver.jsp
* https://www.postgresql.org/about/
* https://jsoup.org/apidocs/overview-summary.html
* https://en.wikipedia.org/wiki/Data\_access\_object
* https://medium.com/front-end-weekly/dont-be-afraid-of-headless-chrome-learn-why-and-how-to-use-it-for-ember-testing-e43710f4d443
* https://www.siteground.com/kb/google\_free\_smtp\_server/
* https://maven.apache.org/guides/introduction/introduction-to-dependency-mechanism.html
* https://basarat.gitbooks.io/typescript/docs/async-await.html
* https://www.w3schools.com/jsref/jsref\_obj\_string.asp
* https://www.w3schools.com/jsref/jsref\_obj\_array.asp
* https://www.w3schools.com/jsref/jsref\_obj\_number.asp
* https://stanford.edu/~mgorkove/cgi-bin/rpython\_tutorials/ Scraping\_a\_Webpage\_Rendered\_by\_Javascript\_Using\_Python.php
* https://www.sitepoint.com/delay-sleep-pause-wait/
* https://blog.bitsrc.io/keep-your-promises-in-typescript-using-async-await-7bdc57041308
* https://www.quora.com/How-do-top-comparison-sites-do-product-data-match-Which-is-the-most-effective-way-to-do-that-considering-product-data-matching-is-an-inexact-science
* https://www.promptcloud.com/blog/Product-Matching-e-Commerce-Success-Stories/
* http://scrapingauthority.com/data-matching
* https://docs.spring.io/spring/docs/current/spring-framework-reference/overview.html
* https://dzone.com/articles/what-is-spring-boot
* https://www.callicoder.com/hibernate-spring-boot-jpa-one-to-one-mapping-example/
* https://hellokoding.com/jpa-one-to-many-relationship-mapping-example-with-spring-boot-maven-and-mysql/
* https://thoughts-on-java.org/best-practices-many-one-one-many-associations-mappings/
* https://www.baeldung.com/java-with-jsoup
* https://offering.solutions/blog/articles/2016/02/01/consuming-a-rest-api-with-angular-http-service-in-typescript/
* https://www.w3schools.com/css/css\_align.asp
* https://stackoverflow.com/questions/31398380/fatal-error-spawn-cmd-enoent-grunt-serve/31399161
* https://angular.io/guide/ngmodule-faq#sharedmodule
* https://www.w3schools.com/cssref/pr\_class\_cursor.asp
* https://forums.asp.net/t/2084202.aspx?Angular+2+http+put+not+calling+Controller
* https://www.baeldung.com/http-put-patch-difference-spring
* https://spring.io/guides/tutorials/rest/
* https://saucelabs.com/blog/headless-browser-testing-101
* https://material.io/tools/icons/?style=baseline
* https://docs.jboss.org/hibernate/orm/5.4/quickstart/html\_single/
* https://angularfirebase.com/lessons/sharing-data-between-angular-components-four-methods/