Sistem de notificare a clienților privind reducerea prețurilor

PROIECT DE DIPLOMĂ

Autor: **Ruben-Sebastian MARKA**

Conducător științific: **As. dr. ing. Claudiu DOMUȚA**

|  |  |
| --- | --- |
| DECAN  **Prof. dr. ing. Mihaela DÎNȘOREANU** | Vizat,  DIRECTOR DEPARTAMENT AUTOMATICĂ  **Prof. dr. ing. Honoriu VĂLEAN** |

Autor: **Ruben-Sebastian MARKA**

Sistem de notificare a clienților privind reducerea prețurilor

1. **Enunțul temei:** *Construirea unei aplicații web pentru notificare clienților privind reducerea prețurilor produselor. Aplicația web permite utilizatorilor să monitorizeze prețurile produselor de pe diverse platforme de comerț online și să primească notificări atunci când prețurile scad sub un prag prestabilit.*
2. **Conținutul proiectului:***Pagina de prezentare, Declarație privind autenticitatea proiectului, Sinteza proiectului, Cuprins, Introducere, Studiu bibliografic,* *Analiză, proiectare, implementare,Concluzii,Bibliografie.*
3. **Locul documentării:** *Universitatea Tehnică din Cluj-Napoca*
4. **Consultanți:** *As. dr. ing. Claudiu Domuța*
5. **Data emiterii temei:** 01.10.2023
6. **Data predării:** 12.07.2024

Semnătura autorului

Semnătura conducătorului științific

**Declaraţie pe proprie răspundere privind**

**autenticitatea proiectului de diplomă**

Subsemnatul(a) **Ruben-Sebastian MARKA**, legitimat(ă) cu CI/BI seria SM nr. 881578 , CNP 5010307303920 ,

autorul lucrării: Sistem de notificare a clienților privind reducerea prețurilor

elaborată în vederea susținerii examenului de finalizare a studiilor de licență la **Facultatea de Automatică și Calculatoare**, specializarea **Automatică și Informatică Aplicată,** din cadrul Universității Tehnice din Cluj-Napoca, sesiunea iulie 2024 a anului universitar 2023-2024, declar pe proprie răspundere, că această lucrare este rezultatul propriei activități intelectuale, pe baza cercetărilor mele și pe baza informațiilor obținute din surse care au fost citate, în textul lucrării, și în bibliografie.

Declar, că această lucrare nu conține porțiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislației române și a convențiilor internaționale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în fața unei alte comisii de examen de licență.

În cazul constatării ulterioare a unor declarații false, voi suporta sancțiunile administrative, respectiv, *anularea examenului de licență*.

Data Ruben-Sebastian MARKA

(semnătura)

**SINTEZA**

proiectului de diplomă cu titlul:

**Sistem de notificare a clienților privind reducerea prețurilor**

Autor: **Ruben-Sebastian MARKA**

Conducător științific: **As. dr. ing. Claudiu DOMUȚA**

1. Cerințele temei:

2. Soluții alese:

3. Rezultate obținute:

4. Testări și verificări:

5. Contribuții personale:

6. Surse de documentare:

Semnătura autorului

Semnătura conducătorului științific

Cuprins

[1 Introducere 4](#_Toc169621341)

[1.1 Context general 4](#_Toc169621342)

[1.2 Obiective 5](#_Toc169621343)

[1.3 Specificații 7](#_Toc169621344)

[1.3.1 Funcțiile Aplicației 7](#_Toc169621345)

[1.3.2 Interfața Utilizatorului 7](#_Toc169621346)

[1.3.3 Performanță și Scalabilitate 8](#_Toc169621347)

[1.3.4 Structuri de Date și Persistență 8](#_Toc169621348)

[1.3.5 Securitate 9](#_Toc169621349)

[1.3.6 Limitări 9](#_Toc169621350)

[2 Studiu bibliografic 10](#_Toc169621351)

[2.1 Stadiul Actual al Temei Abordate 10](#_Toc169621352)

[2.1.1 Aplicații Existente 10](#_Toc169621353)

[2.1.2 Dezavantajele Aplicațiilor Existente 11](#_Toc169621354)

[2.1.3 Noutățile Aduse de Tema Propusă 11](#_Toc169621355)

[2.2 Necesitatea Sistemelor de Monitorizare a Prețurilor 12](#_Toc169621356)

[2.2.1 Cerințele Utilizatorilor Țintă 12](#_Toc169621357)

[2.2.2 Avantajele Sistemelor de Monitorizare a Prețurilor 13](#_Toc169621358)

[2.3 Tehnologii de Web Scraping 14](#_Toc169621359)

[2.3.1 Introducere în Web Scraping 14](#_Toc169621360)

[2.3.2 Tehnici și Instrumente de Web Scraping 14](#_Toc169621361)

[2.3.3 Provocări și Soluții în Web Scraping 14](#_Toc169621362)

[2.4 Sisteme de Notificare a Prețurilor 15](#_Toc169621363)

[2.4.1 Caracteristici și Funcționalități Esențiale 15](#_Toc169621364)

[2.4.2 Studii de Caz și Exemple 15](#_Toc169621365)

[2.5 Interfețe Intuitive de Utilizator (UI) 17](#_Toc169621366)

[2.5.1 Principii de Proiectare a Interfeței Utilizator 17](#_Toc169621367)

[2.5.2 Exemple și Bune Practici 17](#_Toc169621368)

[2.6 Măsuri de Securitate și Confidențialitate a Datelor 18](#_Toc169621369)

[2.6.1 Importanța Securității Datelor 18](#_Toc169621370)

[2.6.2 Tehnologii și Practici de Securitate 19](#_Toc169621371)

[3 Analiză, proiectare, implementare 20](#_Toc169621372)

[3.1 Analiză și Fundamentare Teoretică 20](#_Toc169621373)

[3.1.1 Python 20](#_Toc169621374)

[3.1.2 Selenium 21](#_Toc169621375)

[3.1.3 Framework-ul Spring Boot 21](#_Toc169621376)

[3.1.4 React 23](#_Toc169621377)

[3.1.5 MySQL 24](#_Toc169621378)

[3.1.5.1 Caracteristici Cheie ale MySQL 24](#_Toc169621379)

[3.1.5.2 Utilizări Comune ale MySQL 25](#_Toc169621380)

[3.1.5.3 Beneficii ale Utilizării MySQL 25](#_Toc169621381)

[3.1.6 Apache Maven 25](#_Toc169621382)

[3.1.7 JSON Web Token 26](#_Toc169621383)

[3.1.8 Arhitectura Layered 27](#_Toc169621384)

[3.1.9 Node Package Manager 28](#_Toc169621385)

[3.2 Proiectare și implementare 30](#_Toc169621386)

[3.2.1 Arhitectura conceptuala 30](#_Toc169621387)

[3.2.2 Cazuri de utilizare 32](#_Toc169621388)

[3.2.2.1 Utilizator autentificat 32](#_Toc169621389)

[3.2.2.2 Utilizator neautentificat 33](#_Toc169621390)

[3.2.3 Descrierea cazurilor de utilizare 34](#_Toc169621391)

[3.2.3.1 Cazul 1: Înregistrare 34](#_Toc169621392)

[3.2.3.2 Cazul 2: Adăugare Produs 35](#_Toc169621393)

[3.2.3.3 Cazul 3: Ștergere Produs 36](#_Toc169621394)

[3.2.3.4 Cazul 4: Actualizare Preț de Prag 38](#_Toc169621395)

[3.2.4 Arhitectura componentei backend 40](#_Toc169621396)

[3.2.5 Arhitectura componentei frontend 44](#_Toc169621397)

[3.2.6 Structura bazei de date 46](#_Toc169621398)

[3.2.6.1 Descrierea tabelelor 47](#_Toc169621399)

[3.2.7 Scripturi Python folosite pentru gestionarea produselor 49](#_Toc169621400)

[3.2.7.1 Script pentru Adăugarea Produselor 49](#_Toc169621401)

[3.2.7.2 Script pentru actualizarea prețurilor 50](#_Toc169621402)

[3.3 Testare și validare 51](#_Toc169621403)

[3.3.1 Testarea front-end 51](#_Toc169621404)

[3.3.1.1 Validări existente în aplicație 51](#_Toc169621405)

[3.3.2 Testarea back-end 52](#_Toc169621406)

[4 Concluzii 53](#_Toc169621407)

[4.1 Rezultate obținute 53](#_Toc169621408)

[4.2 Direcții de dezvoltare 54](#_Toc169621409)

[5 Bibliografie 55](#_Toc169621410)

# Introducere

Într-o eră în care achizițiile online devin din ce în ce mai populare și necesare, există o cerere crescândă pentru soluții care să îmbunătățească experiența utilizatorilor și să faciliteze procesul de cumpărare online. Prin urmare, scopul acestui proiect este de a introduce schimbări în domeniul comerțului electronic prin furnizarea unei metode eficiente și ușor de înțeles pentru monitorizarea și gestionarea prețurilor produselor pe internet. Dorim să oferim utilizatorilor un instrument care le va permite să ia decizii mai informate și să economisească timp și bani atunci când fac cumpărături online prin implementarea unui sistem de notificare a scăderii prețurilor.

## Context general

În zilele noastre, lumea se confruntă cu o multitudine copleșitoare de produse și diferențe de prețuri în mediul online, ceea ce face ca urmărirea reducerii prețurilor să fie dificilă și să necesite timp. Se observă o creștere a numărului de achiziții online, permițând economisirea timpului și livrarea rapidă la domiciliu. Acest lucru este util în special pentru cei care au multe de făcut sau nu locuiesc în apropierea unui centru comercial.

Magazinele online permit achiziționarea de produse din întreaga lume fără a fi necesară călătoria, având o ofertă diversificată în funcție de locație. Din această cauză, se observă o creștere rapidă a numărului de magazine online, care acoperă toate domeniile posibile pentru a satisface nevoile utilizatorilor.

Dezvoltarea rapidă a platformelor de comerț online a evidențiat cererea pentru noi instrumente eficiente care să ajute clienții să ia decizii potrivite cu privire la achizițiile online. Printre aceste instrumente se numără filtre de căutare avansate, recenzii și evaluări ale clienților, recomandări personalizate, compararea produselor și monitorizarea prețurilor. Aceste funcții au un rol important în îmbunătățirea experienței utilizatorilor în ceea ce privește utilizarea serviciilor de comerț online.

Filtrele de căutare sunt extrem de importante în timpul cumpărăturilor online, permițând utilizatorilor să-și adapteze căutările în funcție de diferite cerințe. Aceste filtre permit personalizarea căutării conform preferințelor individuale, făcând astfel procesul de găsire a produselor dorite mult mai rapid și eficient. Opiniile și recenziile clienților joacă un rol crucial în procesul de achiziție online, oferind informații valoroase pentru luarea deciziilor informate și pentru crearea încrederii în produsele pe care intenționează să le achiziționeze.

Recomandările personalizate sunt o modalitate utilă pentru utilizatori de a descoperi produse care se potrivesc cu interesele și preferințele lor. Aceste sugestii sunt adaptate fiecărui individ în funcție de istoricul cumpărăturilor, comportamentul online și alte informații relevante.

Funcția de comparare reprezintă o facilitate care permite utilizatorilor să pună în balanță diverse produse, ajutându-i să aleagă cea mai bună variantă pentru nevoile lor. Utilizatorii pot vizualiza și compara caracteristicile, specificațiile și costurile mai multor produse în același timp, facilitând astfel decizii informate și alegerea produsului care se potrivește cel mai bine nevoilor și bugetului lor.

Urmărirea modificărilor de preț pentru produsele dorite și primirea notificărilor atunci când prețurile scad reprezintă o funcție esențială a sistemului de monitorizare a prețurilor. Utilizatorii pot seta alerte pentru produsele pe care intenționează să le achiziționeze și pot profita de cel mai bun moment pentru a cumpăra, economisind bani prin primirea alertelor atunci când prețurile scad sau ating pragul predeterminat.

Sistemul de notificare pentru scăderea prețului este un instrument inovator conceput pentru a ajuta utilizatorii să urmărească reducerile de preț pentru produsele dorite. Utilizatorii pot folosi acest sistem pentru a primi notificări ori de câte ori prețurile produselor urmărite scad sub un prag prestabilit. Această tehnologie își propune să îmbunătățească experiența de cumpărături online și să ofere clienților posibilitatea de a lua decizii mai bine informate.

Datorită beneficiilor sale și caracterului său practic, acest proiect este semnificativ. Sistemul permite urmărirea și monitorizarea prețurilor, crescând independența și transparența comunității de afaceri online și îmbunătățind experiența de cumpărături a clienților. Prin urmare, sistemul poate oferi avantaje semnificative atât clienților, cât și comercianților, încurajând cumpărăturile online mai informate și conștiincioase.

Decizia de a dezvolta acest sistem a fost determinată de necesitatea evidentă în sectorul cumpărăturilor online. Oamenii caută modalități de a cumpăra mai inteligent și de a obține cele mai bune prețuri, prețuindu-și timpul mai mult ca niciodată. Sistemul oferă o modalitate simplă de a gestiona cumpărăturile online și de a profita la maximum de achiziții. Pe măsură ce comerțul electronic crește în popularitate, nevoia de tehnologii care să sprijine clienții și să îmbunătățească experiența de cumpărare online devine tot mai evidentă. Sistemul nostru oferă o soluție cuprinzătoare care nu doar simplifică cumpărăturile, ci și permite clienților să ia decizii de cumpărare mai informate.

Misiunea acestui proiect este de a revoluționa comerțul electronic și de a garanta că clienții obțin întotdeauna cea mai bună ofertă posibilă, oferindu-le mai multă libertate în experiența lor de cumpărare online.

## Obiective

Proiectul se concentrează pe crearea unui sistem de notificare a scăderii prețurilor în comerțul online. Sistemul este creat pentru a permite clienților să monitorizeze costurile anumitor produse pe piețele online și să primească notificări atunci când prețurile respective scad sub pragurile prestabilite.

Obiectivele principale ale proiectului sunt dezvoltarea unei interfețe intuitive de utilizator pentru controlul notificărilor și preferințelor de produs. Pentru a accesa o piață mare și a oferi o varietate de articole, sistemul este destinat să fie integrat cu alte platforme de tranzacționare online. Creșterea performanței sistemului, scalabilității și fiabilității, precum și consolidarea securității și a confidențialității datelor utilizatorilor, sunt alte obiective importante.

Scopul proiectului este crearea unui sistem ușor de utilizat de urmărire a prețurilor online care să ofere utilizatorilor o experiență distractivă și fără stres. Utilizatorii vor putea crea conturi personalizate folosind sistemul, unde pot adăuga produse de interes și pot stabili pragurile de preț la care doresc să fie notificați. Aceste notificări vor fi trimise prin diverse canale, cum ar fi e-mail, pentru a asigura că utilizatorii nu ratează oportunitățile de economisire.

Proiectul va include mai multe elemente importante pentru a îndeplini aceste obiective. Prima etapă este crearea unei aplicații web ușor de utilizat și accesibilă oricărei categorii de utilizatori. Utilizatorii pot adăuga direct URL-urile produselor pe care le doresc și pot configura prețurile prag pentru fiecare produs. Administrarea simplă a acestor produse și a notificărilor legate va fi posibilă cu ajutorul aplicației.

Dezvoltarea aplicației web va folosi tehnologii front-end și back-end contemporane. Pentru front-end, pot fi utilizate React sau Vue.js, iar pentru back-end, Java sau Node.js. Bazele de date relaționale, cum ar fi PostgreSQL sau MySQL, vor fi utilizate pentru a facilita stocarea eficientă a produselor urmărite și a informațiilor despre utilizator. Tehnica Agile va fi utilizată în dezvoltarea aplicației, permițând iterații rapide și feedback continuu al utilizatorilor.

Colectarea automată a datelor privind prețurile produselor prin utilizarea tehnologiei de scraping web este a doua componentă semnificativă. Acest lucru va permite sistemului să monitorizeze în timp real fluctuațiile prețurilor pe mai multe platforme de tranzacționare de pe internet. Tehnologii de web scraping precum BeautifulSoup și Scrapy vor fi utilizate pentru a colecta date relevante de pe site-urile magazinelor online. Informațiile adunate vor fi procesate pentru a trimite notificări de preț după ce au fost salvate într-o bază de date centrală și cerințele predeterminate sunt îndeplinite.

Gestionarea modificărilor frecvente ale structurii site-urilor web și asigurarea conformității cu politicile acestor site-uri este una dintre principalele provocări în utilizarea web scraping-ului. Sistemul va fi conceput pentru a fi flexibil și ușor de actualizat pentru a rezolva aceste probleme. În cazul în care structura unui site web se schimbă, codul de scraping poate fi actualizat rapid. În plus, pentru a preveni blocarea IP-urilor și alte probleme legale, politicile de utilizare ale site-urilor vor fi respectate.

Securitatea și confidențialitatea datelor utilizatorilor sunt încă o componentă crucială a proiectului. Pentru a proteja datele personale ale utilizatorilor și a împiedica accesul neautorizat, sistemul va folosi metode avansate de securitate, cum ar fi criptarea datelor și politici stricte de confidențialitate.

Obiectivul proiectului este crearea unui sistem de urmărire a prețurilor pe internet sigur, ușor de utilizat și eficient. Utilizatorii vor economisi timp și bani folosind sistemul de monitorizare a prețurilor și oferind notificări rapide. Acest lucru va face cumpărăturile pe internet mai ușoare și mai productive. Astfel, utilizatorii vor putea profita de cele mai bune opțiuni fără a fi nevoiți să verifice în mod regulat prețurile.

Implementarea acestui proiect va crește satisfacția clienților și loialitatea platformelor de tranzacționare online, oferind utilizatorilor și platformelor beneficii semnificative. De exemplu, oamenii vor aprecia achizițiile simple și transparente, în timp ce magazinele online vor obține mai mult trafic și o bază de clienți mai loială.

În cele din urmă, dezvoltarea unui sistem de notificare a scăderii prețurilor în comerțul electronic este o misiune îndrăzneață și necesară în prezent în domeniul comerțului digital. Proiectul va oferi utilizatorilor o modalitate eficientă și sigură de a urmări prețurile produselor de interes și de a beneficia de cele mai bune oferte disponibile prin intermediul unei aplicații web bine dezvoltate și a unei infrastructuri robuste de web scraping. Protecția utilizatorilor în toate fazele procesului de monitorizare va fi asigurată prin prioritizarea securității și confidențialitatea datelor. Implementarea cu succes a acestui proiect va oferi platformelor de tranzacționare online și utilizatorilor numeroase avantaje, îmbunătățind experiența de cumpărare online și sporind economiile personale.

Acestea fiind spuse, obiectivele proiectului sunt stabilite pentru a îndeplini nevoile actuale ale pieței de comerț online și pentru a oferi utilizatorilor o experiență mai bună și mai eficientă în procesul de cumpărare online.

## Specificații

Acest capitol descrie cerințele și obiectivele proiectului, detaliind funcțiile aplicației, interfața utilizator, performanța, structurile de date, elementele de securitate, fiabilitatea, calitatea și limitările sistemului. Scopul este de a oferi utilizatorilor o experiență de cumpărare online îmbunătățită și eficientă.

### Funcțiile Aplicației

Aplicația de monitorizare a prețurilor este concepută pentru a oferi utilizatorilor instrumente esențiale pentru a urmări și gestiona prețurile produselor de interes. Aceste funcții sunt dezvoltate pentru a simplifica procesul de cumpărare online, oferind notificări în timp real despre modificările de preț și asigurând o interacțiune ușoară și intuitivă cu sistemul. Funcțiile principale ale aplicației sunt detaliate mai jos:

1. **Monitorizarea Prețurilor:** Aplicația va permite utilizatorilor să adauge URL-uri ale produselor de interes și să seteze prețuri de prag. Atunci când prețul unui produs scade sub pragul stabilit, utilizatorii vor primi notificări.
2. **Notificări:** Utilizatorii vor primi notificări prin e-mail sau alte canale, cum ar fi notificările push, pentru a fi informați în timp real despre scăderile de preț.
3. **Istoricul Prețurilor:** Aplicația va afișa istoricul prețurilor pentru produsele urmărite, permițând utilizatorilor să vadă tendințele prețurilor în timp.
4. **Autentificare și Conturi de Utilizator:** Utilizatorii vor putea crea conturi personalizate, în care să își gestioneze produsele urmărite și setările de notificare.

### Interfața Utilizatorului

Proiectarea interfeței utilizatorului este esențială pentru asigurarea unei experiențe de utilizare plăcută și eficientă. O interfață bine concepută nu doar că face aplicația mai atrăgătoare, dar și facilitează utilizarea acesteia, reducând timpul și efortul necesar pentru realizarea diferitelor sarcini. În continuare, sunt prezentate caracteristicile cheie ale interfeței utilizatorului în cadrul acestui proiect:

1. **Design Intuitiv:** Interfața utilizatorului va fi proiectată pentru a fi ușor de utilizat și intuitivă, asigurându-se că utilizatorii pot naviga rapid și eficient prin diferitele secțiuni ale aplicației. Prin utilizarea unui design minimalist și a unor elemente vizuale clare, aplicația va permite utilizatorilor să acceseze rapid funcțiile dorite fără a se confrunta cu dificultăți .
2. **Responsive Design:** Aplicația va fi optimizată pentru a funcționa pe diverse dispozitive, inclusiv desktopuri, tablete și telefoane mobile. Acest lucru va asigura o experiență de utilizare consistentă și satisfăcătoare, indiferent de dispozitivul folosit. Designul responsive va adapta automat interfața în funcție de dimensiunile ecranului, menținând astfel o utilizare eficientă și estetică.
3. **Personalizare:** Utilizatorii vor avea posibilitatea de a personaliza setările aplicației în funcție de preferințele lor. Aceasta va include opțiuni de personalizare a notificărilor, alegerea temelor de culoare și organizarea produselor urmărite. Prin oferirea acestor opțiuni, aplicația va răspunde mai bine nevoilor și preferințelor fiecărui utilizator, îmbunătățind astfel experiența generală de utilizare.

### Performanță și Scalabilitate

Pentru a asigura o experiență optimă de utilizare și a permite creșterea continuă a numărului de utilizatori și a volumului de date, aplicația de monitorizare a prețurilor trebuie să fie performantă și scalabilă. Performanța se referă la capacitatea sistemului de a răspunde rapid și eficient la solicitările utilizatorilor, iar scalabilitatea implică abilitatea de a se adapta la creșterea cerințelor fără a compromite funcționarea. În această secțiune, sunt detaliate aspectele cheie legate de performanță și scalabilitate care vor fi implementate în aplicația noastră.

1. **Performanță Ridicată:** Sistemul va fi optimizat pentru a oferi performanță ridicată, asigurând timpi de răspuns rapizi și gestionarea eficientă a cererilor multiple simultane.
2. **Scalabilitate:** Aplicația va fi proiectată pentru a fi scalabilă, permițând adăugarea de noi funcționalități și gestionarea unui număr crescut de utilizatori fără a compromite performanța.

### Structuri de Date și Persistență

Pentru a asigura funcționarea eficientă și fiabilă a aplicației, structurile de date și persistența informațiilor joacă un rol esențial. Structurile de date sunt folosite pentru a organiza și gestiona eficient datele, în timp ce persistența datelor asigură stocarea pe termen lung a informațiilor într-un mod sigur și accesibil. În continuare, sunt prezentate aspectele cheie ale structurilor de date și persistenței utilizate în cadrul acestui proiect.

1. **Baza de Date MySQL:** Datele despre utilizatori, produse și istoricul prețurilor vor fi stocate într-o bază de date relațională MySQL, asigurând integritatea și securitatea datelor.
2. **Data Transfer Objects (DTOs):** Structurile de date vor fi gestionate prin intermediul DTO-urilor, facilitând transferul de date între componentele aplicației.

### Securitate

Asigurarea securității datelor și a accesului utilizatorilor este esențială pentru orice aplicație web, mai ales pentru un sistem de notificare a scăderii prețurilor care gestionează informații sensibile despre utilizatori și produsele lor preferate. Această secțiune descrie măsurile de securitate implementate pentru a proteja integritatea și confidențialitatea datelor, prevenind accesul neautorizat și protejând împotriva vulnerabilităților comune. Securitatea sistemului se bazează pe trei piloni principali: autentificare și autorizare, criptarea datelor și protecția împotriva vulnerabilităților. Aceste măsuri asigură că utilizatorii pot utiliza aplicația cu încredere, știind că datele lor sunt în siguranță.

1. **Autentificare și Autorizare:** Sistemul va implementa mecanisme de autentificare și autorizare pentru a asigura accesul securizat la conturile utilizatorilor și la datele acestora.
2. **Criptarea Datelor:** Datele sensibile vor fi criptate atât în tranzit, cât și în repaus, utilizând protocoale de securitate precum SSL/TLS.
3. **Protecție împotriva Vulnerabilităților:** Aplicația va include măsuri de protecție împotriva vulnerabilităților comune, cum ar fi atacurile de tip SQL injection și XSS (Cross-Site Scripting).

### Limitări

Dezvoltarea și implementarea unui sistem de notificare a scăderii prețurilor implică o serie de limitări care pot afecta performanța și utilizabilitatea aplicației. Este important să se recunoască aceste limitări pentru a gestiona așteptările utilizatorilor și pentru a identifica potențiale îmbunătățiri viitoare. Una dintre principalele limitări ale aplicației este compatibilitatea cu site-urile web, deoarece depinde de structura acestora, care poate suferi modificări frecvente. De asemenea, performanța și accesibilitatea aplicației sunt influențate direct de calitatea conexiunii la internet a utilizatorilor, ceea ce poate afecta experiența generală de utilizare.

1. **Compatibilitate cu Site-urile Web:** Fiind dependent de structura site-urilor de comerț electronic, aplicația poate întâmpina dificultăți în cazul modificărilor frecvente ale acestora.
2. **Dependență de Conectivitatea Internet:** Performanța și accesibilitatea aplicației sunt direct legate de calitatea conexiunii la internet a utilizatorilor.

Aceste specificații detaliază cerințele și obiectivele proiectului, asigurând că sistemul de notificare a scăderii prețurilor oferă o experiență îmbunătățită de cumpărături online, combinată cu securitate, performanță și scalabilitate optimă.

# Studiu bibliografic

Acest capitol discută contextul și baza teoretică a proiectului de notificare a scăderii prețurilor, care vor fi discutate mai detaliat în capitolele viitoare. Vom începe prin a explica de ce este nevoie de un astfel de sistem, subliniind ce doresc utilizatorii țintă. Vom continua prin a descrie avantajele oferite de cele mai populare sisteme de monitorizare a prețurilor și vom sublinia caracteristicile importante ale sistemelor existente care folosesc concepte similare. Pentru a oferi utilizatorilor o experiență completă și satisfăcătoare cu sistemele de notificare a scăderii prețurilor, este esențial să găsim și să evaluăm funcționalitățile sistemelor similare care au fost deja cunoscute. Acest efort ne-a permis să menținem funcționalitățile de bază care s-au dovedit eficiente, precum și să adăugăm îmbunătățiri și componente noi. Identificarea și integrarea celor mai bune tehnologii disponibile a fost o altă etapă crucială. Prin urmare, documentarea și compararea sistemelor existente au fost esențiale pentru decizia de a construi acest proiect.

Studiul bibliografic se va concentra pe analiza literaturii și a tehnologiilor relevante pentru crearea unui sistem de notificare a scăderii prețurilor în comerțul online. Vom analiza tehnologiile de web scraping, sistemele de notificare a prețurilor, interfețele intuitive de utilizator și măsurile de securitate și confidențialitate a datelor disponibile. Acest efort va oferi o bază solidă de informații și bune practici, care vor fi esențiale pentru finalizarea cu succes a proiectului nostru.

## Stadiul Actual al Temei Abordate

Acest subcapitol examinează situația actuală a sistemelor de monitorizare a prețurilor, examinând aplicațiile actuale, dezavantajele și îmbunătățirile pe care le oferă tema propusă. Pentru a identifica punctele slabe și a sublinia inovațiile aduse de noua soluție propusă, este esențial să înțelegem situația actuală.

### Aplicații Existente

Numeroase aplicații și platforme oferă servicii de monitorizare a prețurilor și notificare a reducerilor. Printre cele mai populare se numără:

1. **CamelCamelCamel:** Este un instrument recunoscut pentru urmărirea prețurilor produselor de pe Amazon. Această aplicație permite utilizatorilor să vizualizeze istoricul prețurilor și să configureze alerte pentru reduceri de preț [1]. Principalele caracteristici includ:

* Vizualizarea istoricului prețurilor
* Alerte de scădere a prețurilor
* Suport pentru multiple piețe Amazon

2. **Honey:** Este o extensie de browser care oferă cupoane și reduceri pentru achizițiile online. Utilizatorii pot stabili alerte pentru produse specifice și pot vizualiza istoricul prețurilor [2].

Caracteristici notabile:

* Alerte pentru reduceri și cupoane
* Istoric al prețurilor
* Recomandări personalizate bazate pe comportamentul de navigare

3. **Keepa:** O extensie de browser care oferă graficul de istoricul prețurilor pentru produsele de pe Amazon și notificări de preț [3]. Funcționalitățile includ:

* Grafic de istoricul prețurilor
* Notificări de scădere a prețurilor
* Integrare cu Amazon pentru mai multe piețe

### Dezavantajele Aplicațiilor Existente

Analizând aplicațiile existente, se pot identifica mai multe dezavantaje:

**Interfața Utilizator**: Multe dintre aceste aplicații au interfețe complicate, ceea ce poate fi greu pentru utilizatorii noi sau pentru cei care nu sunt familiarizați cu tehnologia. De exemplu, Keepa are o interfață încărcată cu multe informații, ceea ce poate fi copleșitor [3].

**Limitări în Personalizare**: De cele mai multe ori, nu există suficientă flexibilitate pentru a modifica alertele. De exemplu, CamelCamelCamel limitează personalizarea alertelor bazate pe preț [1].

**Restricții Geografice**: Unele aplicații sunt limitate la anumite piețe geografice. Honey, de exemplu, este foarte eficientă în Statele Unite, dar nu oferă aceeași acoperire și funcționalitate în alte regiuni [2].

**Acuratețea Datelor:** Notificările pot fi întârziate sau inexacte dacă datele colectate nu sunt actualizate în timp real. Utilizatorii care doresc să profite de oferte limitate în timp au probleme cu asta [1].

**Lipsa de Integrare:** Majoritatea acestor aplicații se concentrează pe monitorizarea prețurilor doar pe anumite platforme, fără a oferi o integrare completă cu diverse site-uri de comerț electronic. Această limitare poate reduce eficiența și utilitatea aplicațiilor [1] [3] [2].

### Noutățile Aduse de Tema Propusă

Tema propusă pentru sistemul de notificare a scăderii prețurilor aduce mai multe noutăți și îmbunătățiri față de aplicațiile existente:

**Interfață Intuitivă:** Se propune aplicarea principiilor de design al interfeței utilizatorului pentru a crea o interfață care este ușor de folosit și intuitivă, chiar și pentru utilizatorii care nu au experiență tehnică. Pentru a îmbunătăți experiența utilizatorului, simplitatea, consistența și feedback-ul sunt integrate.

**Personalizare Avansată:** Sistemul oferă opțiuni avansate de personalizare a alertelor. Utilizatorii pot seta notificări nu doar bazate pe preț, dar și pe alte criterii, cum ar fi disponibilitatea produsului sau apariția unor oferte speciale. Aceasta adresează limitările aplicațiilor existente, oferind utilizatorilor mai mult control și flexibilitate.

**Acuratețe și Timp Real:** Colectarea și actualizarea datelor în timp real este asigurată de tehnici sofisticate de web scraping și API-uri. Prin urmare, utilizatorii primesc notificări imediat ce au loc modificări ale prețurilor, ceea ce elimină întârzierile și inexactitățile pe care le-au întâlnit alte aplicații.

**Securitate și Confidențialitate:** Pentru a proteja datele utilizatorilor, se iau măsuri stricte de securitate a datelor, cum ar fi criptarea datelor și respectarea regulamentelor de confidențialitate. Aceasta este răspunsul la preocupările crescânde ale utilizatorilor cu privire la siguranța datelor lor personale.

**Integrare Extinsă**: Noul sistem propus va permite integrarea cu multiple platforme de comerț electronic, nu doar Amazon. Aceasta se va realiza prin API-uri și tehnici de web scraping, oferind utilizatorilor o imagine completă și actualizată a pieței.

În plus, sistemul propus permite adăugarea URL-ului unui produs și a unui preț de prag. Când prețul produsului ajunge sub prețul de prag, se va trimite o notificare către utilizator. Produsele adăugate la favorite pot fi urmărite pe profilul utilizatorului, unde se poate actualiza prețul de prag și viziona istoricul prețului acelui produs pe un grafic.

Prin aceste inovații, sistemul propus își propune să ofere o experiență superioară, adaptată nevoilor și așteptărilor utilizatorilor moderni. Se urmărește astfel îmbunătățirea semnificativă a funcționalității și eficienței comparativ cu soluțiile existente.

## Necesitatea Sistemelor de Monitorizare a Prețurilor

### Cerințele Utilizatorilor Țintă

Pentru a înțelege nevoile unui sistem de monitorizare a prețurilor, este esențial să examinăm comportamentul și nevoile utilizatorilor țintă. Potrivit studiilor de piață și rapoartelor despre comportamentul consumatorilor online, consumatorii caută produse eficiente care le permit să economisească timp și bani. Aceștia caută modalități de a urmări prețurile și de a primi alerte despre reduceri de preț pentru produsele de interes în timp real.

Conform studiilor, consumatorii preferă tot mai mult cumpărăturile online datorită convenienței și diversității ofertelor. Aceștia apreciază posibilitatea de a compara prețurile și de a cumpăra bunuri fără a vizita magazinele fizice. Consumatorii sunt tot mai interesați de reduceri de preț și caută modalități de a economisi bani. Pentru a satisface aceste nevoi, un sistem de monitorizare a prețurilor poate oferi notificări rapide și precise atunci când prețurile produselor dorite scad sub un prag stabilit [4].

Studiile oferă o bază teoretică pentru a înțelege modul în care oamenii caută informații online. Utilizatorii participă la un proces iterativ de căutare a informațiilor, în care evaluează în mod regulat relevanța și utilitatea informațiilor pe care le găsesc. Acest lucru se manifestă atunci când cumpărați online, comparând prețurile pentru a găsi cele mai bune opțiuni. Un sistem de monitorizare a prețurilor poate simplifica procesul de colectare și analiză a datelor privind prețurile, oferind consumatorilor informații esențiale pentru luarea deciziilor de cumpărare [5].

Utilizatorii apreciază și sistemele care oferă notificări rapide și precise și ușor de utilizat. Cercetările arată că sistemele de notificare eficiente trebuie să fie personalizabile, permițând utilizatorilor să seteze preferințe specifice pentru produsele și prețurile de interes. Prin urmare, utilizatorii pot primi notificări relevante pentru a reacționa rapid și a beneficia de cele mai bune oferte disponibile.  
Un sistem de monitorizare a prețurilor poate fi construit pe baza acestor cerințe și comportamente, făcând experiența de cumpărături online mai bună și oferindu-le posibilitatea de a economisi timp și bani.

### Avantajele Sistemelor de Monitorizare a Prețurilor

Sistemele de monitorizare a prețurilor oferă numeroase beneficii comercianților și consumatorilor. Aceste avantaje sunt vitale pentru a înțelege semnificația și necesitatea implementării unui astfel de sistem.

Unul dintre principalele avantaje pentru clienți este economisirea timpului și banilor. Utilizatorii își economisesc timpul pierdut examinând manual prețurile pe mai multe site-uri. În plus, aceștia pot obține produsele dorite la prețuri mai scăzute, astfel încât să fie informați imediat ce prețurile ajung sub pragurile stabilite [4].

Sistemele de monitorizare a prețurilor ajută la luarea deciziilor informate de cumpărare. Utilizatorii primesc informații actualizate despre fluctuațiile prețurilor, ceea ce le permite să facă achiziții bazate pe informații relevante. Accesul la informații precise și actualizate în timp real este esențial pentru a obține cele mai bune oferte pe o piață care se schimbă rapid. Personalizarea este o caracteristică cheie a sistemelor eficiente de monitorizare a prețurilor. Utilizatorii pot ajusta setările de notificare în funcție de preferințele lor specifice, asigurându-se că primesc doar alertele relevante pentru produsele și prețurile care îi interesează cel mai mult [5].

Un alt avantaj semnificativ pentru consumatori este comoditatea pe care o oferă aceste sisteme. Utilizatorii pot configura alerte personalizate pentru produsele de interes și pot primi notificări prin mai multe canale, cum ar fi e-mail, SMS sau notificări push. Această flexibilitate le permite să fie informați despre reduceri de preț oriunde se află sau folosesc un dispozitiv. Pentru comercianți, implementarea unor astfel de sisteme aduce beneficii semnificative. Creșterea traficului și a vânzărilor este unul dintre principalele avantaje. Notificările de preț atrag mai mulți vizitatori pe site-urile comercianților, crescând astfel șansele de conversie și de vânzare a produselor [4].

Loialitatea clienților este un alt beneficiu important. Oferirea unui serviciu de monitorizare a prețurilor poate îmbunătăți satisfacția clienților, determinându-i să revină pentru achiziții viitoare. Comercianții pot folosi aceste sisteme pentru a construi relații mai puternice și mai durabile cu clienții lor [5].

De asemenea, sistemele de monitorizare a prețurilor permit comercianților să-și modifice strategiile de marketing folosind datele despre comportamentul de cumpărare și preferințele utilizatorilor. Acest lucru le permite să ofere promoții personalizate și să targeteze audiențele relevante într-un mod mai eficient, ceea ce le permite să maximizeze impactul campaniilor lor de marketing [4].

Pe baza acestor avantaje, este evident că sistemele de monitorizare a prețurilor aduc beneficii semnificative atât pentru consumatori, cât și pentru comercianți. Aceste sisteme contribuie la crearea unui mediu de cumpărături online mai eficient, transparent și convenabil pentru toate părțile implicate.

## Tehnologii de Web Scraping

### Introducere în Web Scraping

Web scraping-ul este o tehnologie crucială pentru colectarea automată a datelor de pe paginile web. Este frecvent folosit pentru a colecta informații despre produse și prețuri de pe platformele de comerț electronic. Scraping-ul web permite obținerea rapidă și eficientă a datelor necesare pentru analiza pieței și monitorizarea prețurilor, automatizând procesul de colectare a datelor [6].

Această tehnologie poate accesa și colecta informații din pagini web prin parsarea codului HTML și, atunci când sunt disponibile, API-uri. Web scraping poate fi realizat cu o varietate de limbaje de programare, dar Python este unul dintre cele mai populare din cauza bibliotecilor sale puternice și a ușurință de utilizare [6].

### Tehnici și Instrumente de Web Scraping

Diverse tehnici și instrumente sunt utilizate pentru web scraping, fiecare având avantajele și limitările sale. Cele mai populare biblioteci și cadre de lucru includ BeautifulSoup, Scrapy și Selenium.

BeautifulSoup este o bibliotecă Python care este capabilă să parseze documente HTML și XML. Este extrem de flexibil și ușor de utilizat, permițând extragerea rapidă a informațiilor din site-uri web [7]. Pentru începători și proiecte mai mici, aceasta este cea mai bună opțiune.

Scrapy, care a fost dezvoltat în Python, este un cadru de lucru puternic pentru scraping web. Este proiectat pentru a permite scraping-ul rapid și scalabil, ceea ce permite colectarea unor volume mari de informații de pe pagini multiple [7]. De exemplu, Scrapy oferă urmărirea link-urilor și gestionarea automată a sesiunilor.

Selenium este un alt program popular care este utilizat în principal pentru testarea automatizată a aplicațiilor web, dar poate fi utilizat și pentru scraping web. Selenium funcționează într-un browser web real, permițând utilizatorilor să interacționeze cu paginile web la fel ca un uman. Aceasta este utilă pentru scraping-ul paginilor JavaScript dinamice [7].

### Provocări și Soluții în Web Scraping

Unele dintre numeroasele probleme cu care se confruntă scraping-ul web sunt conformitatea cu politicile site-urilor web și modificările frecvente ale structurii site-urilor web. Un obstacol major este faptul că structura HTML a unei pagini web poate fi modificată fără a fi anunțată, ceea ce poate duce la eșecul scripturilor de scraping [8].Descoperirea de soluții care sunt adaptabile și ușor de actualizat este esențială pentru a aborda această problemă. Utilizarea tehnicilor de detectare a modificărilor și actualizarea automată a scripturilor de scraping este o soluție comună. Structura paginilor este urmărită în mod regulat și scripturile sunt modificate în funcție de aceasta [8].În plus, respectarea termenilor și condițiilor site-ului și utilizarea practicilor etice de scraping sunt esențiale pentru a preveni aceste probleme [9].

O altă problemă semnificativă este respectarea politicilor site-ului. Multe site-uri au reguli stricte care interzic scraping-ul web, iar nerespectarea lor poate duce la blocarea IP-urilor sau la acțiuni legale. Respectarea termenilor și condițiilor site-ului și utilizarea practicilor etice de scraping sunt esențiale pentru a preveni aceste probleme. De exemplu, este posibil să utilizați tehnici de scraping cu rată limitată pentru a preveni supraîncărcarea serverelor și pentru a face ca scopul și utilizarea datelor colectate să fie transparente [9].

## Sisteme de Notificare a Prețurilor

### Caracteristici și Funcționalități Esențiale

Sistemele de notificare a prețurilor oferă funcționalități esențiale care ajută utilizatorii să fie informați în timp util despre reduceri de preț. Unele dintre aceste caracteristici sunt pragurile de preț, notificările în timp real, personalizarea alertelor și integrarea cu mai multe platforme de comerț electronic. Pragurile de preț le permit utilizatorilor să stabilească cel mai mare preț pe care sunt dispuși să-l plătească pentru un produs. Utilizatorul este informat de îndată ce prețul produsului scade sub acest prag. Aceasta este vitală pentru utilizatorii care doresc să obțină cele mai bune opțiuni disponibile.

Notificările în timp real sunt critice pentru asigurarea că utilizatorii primesc informații la momentul potrivit. Aceste notificări pot fi trimise prin o varietate de canale, cum ar fi e-mail, SMS sau notificările push pe telefoanele mobile. Sistemele de notificare a prețurilor eficiente trebuie să proceseze rapid informațiile despre prețuri și să genereze notificări aproape instantanee, astfel încât utilizatorii să nu rateze oportunitățile de economisire. O altă caracteristică semnificativă este personalizarea notificarilor, care permite utilizatorilor să seteze preferințe specifice pentru produsele și categoriile de interes, precum și modalitatea de primire a notificarilor. Acest grad de personalizare îmbunătățește experiența utilizatorilor și garantează relevanța alertelor.

Aceste sisteme pot obține o imagine completă a pieței prin colectarea de date din mai multe surse prin integrarea cu diferite platforme de comerț electronic. În funcție de disponibilitatea și compatibilitatea platformelor, aceste integrări pot fi realizate prin API-uri sau tehnici de web scraping.

Aceste caracteristici sunt evidențiate în diverse studii și articole. De exemplu, Mitchell (2018) descrie tehnici eficiente de web scraping și integrarea acestora cu platformele de comerț electronic, evidențiind metodele de detectare a modificărilor și generarea notificărilor eficiente [9].

### Studii de Caz și Exemple

Studiile de caz și exemplele din viața reală oferă perspective utile asupra modului în care sistemele de notificare a prețurilor funcționează și cât de eficiente sunt. Analiza acestor studii ajută la înțelegerea modului în care aceste sisteme pot fi puse în aplicare și a efectelor pe care le pot avea utilizatorii și comercianții.

Un exemplu notabil este prezentat de Ferrara și colab. (2014), care analizează dezvoltarea și implementarea unui sistem de alertă de preț pentru o platformă de comerț electronic. Sistemul pe care autorii l-au descris permite utilizatorilor să configureze alerte de preț pentru anumite produse și să primească notificări în timp real atunci când prețurile acestor produse scad sub pragurile stabilite. Acest sistem a fost

creat pentru a satisface nevoile utilizatorilor, economisind timp și bani. În același timp, sistemul oferă o experiență de utilizare simplă și intuitivă. Studiul subliniază că designul ușor de înțeles al interfeței de utilizator și un mecanism robust de colectare și procesare a datelor sunt esențiale. Acest caz demonstrează că succesul unui astfel de sistem depinde în mare măsură de capacitatea de a furniza notificări precise și în timp util, precum și de capacitatea de a personaliza alertele pentru utilizatori în funcție de nevoile lor [10].

Un alt exemplu este platforma CamelCamelCamel1, un instrument popular de urmărire a prețurilor pentru produse de pe Amazon2. CamelCamelCamel permite utilizatorilor să vizualizeze istoricul prețurilor și să configureze alerte pentru reduceri de preț [2]. Conform cercetărilor efectuate pe platformă, utilizatorii apreciază capacitatea de a urmări tendințele istorice ale prețurilor și de a primi notificări personalizate [1]. Această funcționalitate îi încurajează pe utilizatori să facă achiziții informate și le permite să profite de reduceri imediat ce sunt disponibile. Pentru a se asigura că datele prezentate sunt întotdeauna actualizate, platforma folosește tehnici avansate de web scraping pentru a urmări fluctuațiile de preț în timp real [11]. În succesul sistemelor de notificare a prețurilor, accesul la date istorice și capacitatea de personalizare sunt esențiale [3].

Un alt studiu de caz relevant este analiza platformei Honey3, o extensie de browser care oferă utilizatorilor cupoane și reduceri pentru achizițiile online. Honey urmărește prețurile și informează utilizatorii când găsește cupoane sau reduceri [2]. Utilizatorii pot stabili alerte pentru produse specifice și pot vizualiza istoricul prețurilor pe platformă. Studiul Honey demonstrează cum adăugarea unei funcționalități de alertă de preț la un serviciu de economisire mai extins poate crește valoarea oferită utilizatorilor și poate crește adoptarea platformei [2]. Honey îmbunătățește precizia alertelor de preț și recomandările de cupoane prin utilizarea algoritmilor de învățare automată.

Studiile de caz arată că succesul sistemelor de notificare a prețurilor este majorat de funcționalități importante, cum ar fi notificările în timp real și personalizarea notificarilor. Aceste funcționalități îmbunătățesc experiența utilizatorilor și cresc loialitatea și satisfacția clienților, crescând traficul și veniturile comercianților [2]. Prin examinarea acestor exemple, se poate concluziona că atât consumatorii, cât și comercianții pot beneficia substanțial de un sistem de notificare a prețurilor bine conceput și implementat. Acest sistem ar putea crea un mediu de cumpărături online mai clar și mai ușor de utilizat.

## Interfețe Intuitive de Utilizator (UI)

### Principii de Proiectare a Interfeței Utilizator

Proiectarea unei experiențe de utilizare eficiente și plăcute este o parte importantă a designului interfețelor intuitive de utilizator (UI). Proiectarea interfeței utilizatorului se bazează pe claritate, accesibilitate și ușurință de utilizare. Aceste principii includ:

* 1. Simplitatea: Interfețele ar trebui să fie simple și să nu ofere utilizatorului prea multe informații inutile. Elemente de design ar trebui să fie ușor de înțeles și ușor de identificat [12].
  2. Consisteța : Designul interfeței ajută utilizatorii să înțeleagă și să predice funcționalitatea. Utilizarea unui set uniform de culori, fonturi și stiluri de butoane pe întregul site sau aplicație este un exemplu de acest lucru [13].
  3. Feedback-ul utilizatorului: Utilizatorii trebuie să primească feedback clar și imediat asupra acțiunilor lor. De exemplu, atunci când este apăsat un buton, culoarea acestuia ar trebui să schimbe pentru a indica faptul că acțiunea a fost înregistrată [14].
  4. Accesibilitatea: Interfețele trebuie să fie accesibile pentru toți utilizatorii, inclusiv pentru cei cu dizabilități. Utilizarea etichetelor clare, contrastele adecvate de culori și navigarea simplă prin tastatură sunt toate elemente care contribuie la acest lucru [14].
  5. Prevenirea erorilor: Erorile ar trebui evitate prin designul interfeței și corectate ușor atunci când apar. Este necesar ca mesajele de eroare să fie clare și să ofere utilizatorilor soluții [12].
  6. Înțelegerea utilizatorului: Un design bun se concentrează pe oameni. Pentru a crea interfețe care să satisfacă așteptările utilizatorilor, este esențial să știți cum se comportă utilizatorii [13].

Aceste principii sunt esențiale pentru a asigura că utilizatorii au o experiență de utilizare plăcută și eficientă, reducând frustrarea și îmbunătățind satisfacția generală.

### Exemple și Bune Practici

Analiza interfețelor de utilizator bine proiectate și discuția despre bunele practici în designul UI sunt esențiale pentru a înțelege cum să aplici principiile de proiectare în practică.

Platforma Google4 este un exemplu de interfață bine concepută. Interfața sa este renumită pentru simplitatea și ușurința cu care este concepută. Pagina de căutare principală are un design simplu și se concentrează pe bara de căutare. Acest lucru le permite utilizatorilor să se concentreze pe sarcinile lor principale, care este căutarea de informații, fără a pierde timpul inutile [12].

Instagram5 pe telefonul mobil este un alt exemplu. Instagram are un design consistent și ușor de folosit care facilitează navigarea între feed-uri, mesaje și profiluri. Feedback-ul vizual este bine utilizat. De exemplu, atunci când cineva apasă pe un buton Like, acesta se evidențiază și oferă o confirmare instantanee a acțiunii [14].

În plus, aplicațiile de banking online precum Revolut6 sunt exemple excelente de interfețe care combină funcționalitatea complexă cu simplitatea. Revolut are un design curat și ușor de înțeles, ceea ce permite utilizatorilor să-și gestioneze cu ușurință banii. Funcțiile complicate, cum ar fi schimburile valutare și transferurile internaționale, sunt prezentate într-un mod accesibil și ușor de utilizat [13].

Bunele practici în designul UI includ:

1. Testarea efectuată cu utilizatorii: Testarea interfeței cu utilizatorii reali este esențială pentru a identifica problemele și a îmbunătăți designul. Testarea utilizatorilor oferă perspective utile asupra modului în care utilizatorii interacționează cu interfața și unde se întâmplă probleme [13].
2. Iterarea designului: Designul UI este un proces iterativ. Pe baza feedback-ului utilizatorilor și a testelor, interfața trebuie rafinată și îmbunătățită constant [13].
3. Utilizarea prototipurilor: Înainte de implementarea finală, crearea prototipurilor interfeței permite testarea și evaluarea ideilor de design. Acest lucru reduce timpul și cheltuielile și permite modificări rapide [13].
4. Consistența: Menține un design uniform pe toate paginile și elementele interfeței. Consistența face utilizarea interfeței mai rapidă și mai ușoară [13].

Prin aplicarea acestor bune practici, designerii pot crea interfețe care nu doar îndeplinesc nevoile utilizatorilor, dar și oferă o experiență de utilizare superioară.

## Măsuri de Securitate și Confidențialitate a Datelor

### Importanța Securității Datelor

În contextul sistemelor de notificare a prețurilor, securitatea și confidențialitatea datelor utilizatorilor sunt esențiale. Datele personale și preferințele de cumpărături ale utilizatorilor trebuie protejate împotriva accesului neautorizat și a utilizării abuzive. Conform studiilor, utilizatorii sunt din ce în ce mai preocupați de modul în care datele lor sunt colectate, stocate și utilizate. Acest aspect devine și mai critic pe măsură ce tot mai multe informații personale sunt partajate online [15].

Securitatea datelor se referă la protejarea informațiilor împotriva accesului neautorizat, furtului sau distrugerii. În cadrul unui sistem de monitorizare a prețurilor, datele utilizatorilor includ informații sensibile, cum ar fi adrese de e-mail, istoricul căutărilor și preferințele de cumpărături. Aceste informații trebuie protejate pentru a menține încrederea utilizatorilor și pentru a respecta reglementările privind confidențialitatea datelor, cum ar fi GDPR (General Data Protection Regulation) în Europa [15].

Utilizatorii sunt din ce în ce mai conștienți de riscurile asociate cu partajarea datelor lor online. Potrivit unui studiu realizat de Pew Research Center, o majoritate semnificativă a americanilor sunt îngrijorați de confidențialitatea datelor și de modul în care sunt utilizate informațiile lor personale de către companii [16]. Această preocupare sporită subliniază necesitatea implementării unor măsuri de securitate riguroase pentru a proteja datele utilizatorilor.

### Tehnologii și Practici de Securitate

Implementarea unor măsuri de securitate eficiente implică utilizarea tehnologiilor avansate de criptare și a protocoalelor de securitate. De exemplu, criptarea datelor atât în tranzit, cât și în repaus poate preveni accesul neautorizat la informațiile sensibile. Utilizarea protocoalelor SSL/TLS pentru comunicațiile de rețea este esențială pentru a asigura confidențialitatea și integritatea datelor [17].

**Criptarea Datelor:** Criptarea este procesul prin care datele sunt codificate astfel încât doar cei care au cheia de decriptare le pot vedea. În sistemele de notificare a prețurilor, criptarea datelor în tranzit și în repaus garantează că datele utilizatorilor sunt protejate atât în timpul transmiterii lor între server și client, cât și în timpul stocării lor pe servere. Pentru a oferi un nivel înalt de securitate, algoritmii de criptare avansați, cum ar fi AES(Advanced Encryption Standard), sunt utilizați [17].

**Protocoalele SSL/TLS:** Protocoalele criptografice Transport Layer Security (TLS) și Secure Sockets Layer (SSL) garantează comunicații securizate peste o rețea de calculatoare. Aceste protocoale garantează autenticitatea, integritatea și confidențialitatea datelor pe care le transmiteți. Protejarea datelor utilizatorilor de atacuri de tip man-in-the-middle (MIM) necesită implementarea protocolului SSL/TLS [18].

**Autentificare și Controlul Accesului**: Pentru a preveni accesul neautorizat la sistemele și datele utilizatorilor, este esențial să se implementeze mecanisme puternice de autentificare și control al accesului. Autentificarea multi-factor (MFA), care combină doi sau mai mulți factori pentru a verifica identitatea unui utilizator, este una dintre metodele moderne de autentificare. În plus, autorizarea OAuth 2.0 permite utilizatorilor să acceseze resursele într-un mod sigur, fără a compromite datele de autentificare [18].

**Monitorizarea și Auditarea Securității**: Monitorizarea continuă a sistemelor este esențială pentru detectarea și răspunsul rapid la incidente de securitate. Sistemele de prevenire a intruziunilor (IPS) și de detecție a intruziunilor (IDS) identifică și blochează activitățile suspecte din rețea. Auditurile și evaluările de risc frecvente ajută, de asemenea, la menținerea standardelor de securitate și la îmbunătățirea continuă a sistemelor de protecție [19].

Studiul bibliografic a furnizat o bază solidă de informații și bune practici pentru crearea unui sistem de notificare sigur și eficient de scădere a prețurilor. Acest capitol a subliniat importanța creării unui sistem bine conceput care să răspundă cerințelor actuale ale pieței și să ofere o experiență utilizatorilor superioară prin examinarea nevoilor utilizatorilor, analiza aplicațiilor existente și identificarea inovațiilor aduse de tema propusă. Pentru a asigura succesul proiectului nostru, implementarea unor tehnologii sofisticate de web scraping și măsuri riguroase de securitate va asigura funcționalitatea și integritatea sistemului.

# Analiză, proiectare, implementare

## Analiză și Fundamentare Teoretică

Această secțiune detaliază analiza și fundamentarea teoretică necesare pentru dezvoltarea unui sistem de notificare a scăderii prețurilor. În această parte, vor fi discutate metodele aplicate pentru rezolvarea problemei, materialele și procedurile utilizate, tehnicile și calculele relevante, echipamentele folosite și metodologia de proiectare. Scopul este de a oferi o bază solidă și clară pentru implementarea și reproducerea acestei lucrări.

### Python

A logo of a python

Description automatically generated

Python este un limbaj de programare de nivel înalt, interpretat și de uz general, recunoscut pentru sintaxa sa clară și ușor de învățat. Creat de Guido van Rossum, Python a fost lansat pentru prima dată în anul 1991. Limbajul a fost conceput cu scopul de a reduce complexitatea scrierii codului și de a oferi o sintaxă accesibilă și intuitivă [20].

Acest limbaj are o bibliotecă standard largă care include module care îndeplinesc o varietate de funcții, cum ar fi lucrul cu fișiere, rețele, baze de date și interfețe grafice [20]. În plus, există o mulțime de biblioteci terțe care au fost create de comunitatea Python, care ajută la îmbunătățirea capacităților limbajului în diferite domenii, cum ar fi știința datelor (NumPy, Pandas) și inteligența artificială (TensorFlow, Keras) [21]. Programarea procedurală, orientată pe obiecte și funcțională sunt câteva dintre paradigmele de programare pe care limbajul le suportă, ceea ce permite dezvoltatorilor să abordeze o gamă largă de probleme [21].

Utilizările comune ale Python includ dezvoltarea web, știința datelor, machine learning, automatizarea sarcinilor și dezvoltarea de aplicații. Framework-uri precum Django și Flask sunt utilizate pentru dezvoltarea de aplicații web robuste și scalabile. În domeniul științei datelor, biblioteci precum Pandas, NumPy și Matplotlib sunt esențiale pentru analiză și vizualizare de date [21]. În ceea ce privește machine learning și inteligența artificială, framework-uri precum TensorFlow și Scikit-learn permit dezvoltarea și implementarea de modele complexe [21]. Python este, de asemenea, preferat pentru automatizarea sarcinilor repetitive și crearea de scripturi, datorită simplității și eficienței sale. Dezvoltarea de aplicații cu interfață grafică este facilitată de biblioteci precum Tkinter și PyQt [21] .

Comunitatea Python este mare și activă, contribuind continuu la îmbunătățirea limbajului și la dezvoltarea de resurse accesibile pentru învățare și utilizare [20]. Documentația oficială, tutorialele, cursurile online și forumurile de discuții sunt doar câteva dintre resursele disponibile pentru cei care doresc să învețe Python sau să aprofundeze cunoștințele existente [21].

### Selenium

Selenium este un set de instrumente și biblioteci open-source utilizate pentru automatizarea interacțiunilor cu browserele web. Acesta este frecvent utilizat pentru testarea aplicațiilor web, precum și pentru automatizarea sarcinilor repetitive ce implică navigarea pe web. Selenium suportă diverse limbaje de programare, inclusiv Python, Java, C#, Ruby și JavaScript, ceea ce îl face extrem de flexibil și ușor de integrat în diferite medii de dezvoltare [22].

Selenium WebDriver este o parte esențială a Selenium, care utilizează cod în limbaje de programare compatibile pentru a permite controlul direct al browserelor web. Prin intermediul API-ului WebDriver, utilizatorii pot face multe lucruri cu paginile web, cum ar fi navigarea, completarea formularelor, clicarea pe elemente și extragerea de date [23]. Acest lucru face Selenium un instrument excelent pentru dezvoltarea și testarea aplicațiilor web, garantând că aplicațiile funcționează bine pe o varietate de platforme și browsere.

Capacitatea Selenium de a suporta mai multe browsere, cum ar fi Google Chrome, Mozilla Firefox, Microsoft Edge, Safari și Internet Explorer, este un alt avantaj major al acestuia. Aplicațiile web pot fi testate în diferite medii datorită compatibilității cu mai multe platforme, care ajută la asigurarea consistenței și funcționalității aplicațiilor în diferite browsere [22]. O altă componentă vitală este Selenium Grid, care permite testarea simultană pe mai multe mașini și browsere, accelerând procesul de testare [23].

Selenium este utilizat și pentru scraping web, adică extragerea datelor din pagini web. Cu toate că scopul principal al Selenium este automatizarea testării aplicațiilor web, capacitatea sa de a interacționa cu componentele paginilor web în mod programatic îl face o alegere populară pentru sarcinile de scraping web. Dezvoltatorii pot automatiza procesul de colectare a datelor, care ar fi consumator de timp în mod obișnuit dacă ar fi efectuat manual [22].

Comunitatea activă și resursele extinse disponibile pentru Selenium contribuie la ușurința învățării și utilizării acestuia. Documentația oficială, tutorialele, cursurile online și forumurile de discuții sunt resurse valoroase pentru cei care doresc să învețe sau să își îmbunătățească cunoștințele despre Selenium [22].

### Framework-ul Spring Boot

A green logo with black background

Description automatically generated

Spring Boot este un framework open-source bazat pe Java, conceput pentru a simplifica dezvoltarea aplicațiilor stand-alone, de producție, care rulează pe platforma Spring. Lansat de Pivotal Software în 2014, Spring Boot facilitează crearea rapidă a aplicațiilor robuste și scalabile, eliminând necesitatea configurării manuale a infrastructurii și a componentelor esențiale [24].

Configurația sa minimă este o caracteristică importantă a Spring Boot, deoarece permite dezvoltatorilor să creeze și să ruleze aplicații cu o cantitate mică de cod boilerplate. O combinație de convenții predefinite și starter templates fac acest lucru posibil [25]. De exemplu, dezvoltatorii pot folosi starter-ul „boot-boot-starter-web” pentru a crea o aplicație web, care include toate bibliotecile necesare pentru crearea și rularea unei aplicații web [26].

Deoarece Spring Boot se integrează bine cu ecosistemul Spring, permite utilizarea ușoară a modulelor existente precum Spring MVC, Spring Data și Spring Security. Aplicațiile care au fost dezvoltate sunt sigure, robuste și scalabile datorită acestei integrări. În plus, Spring Boot oferă un server web încorporat, cum ar fi Tomcat sau Jetty, care permite aplicațiilor să funcționeze fără a avea nevoie de un server extern [25].

Un alt avantaj major al Spring Boot este suportul pentru configurarea automatizată. Spring Boot poate configura automat o varietate de componente și servicii necesare pentru aplicație, în funcție de dependențele proiectului. Aceasta reduce semnificativ timpul și energia necesare pentru începerea proiectului [24]. În plus, Spring Boot include un sistem de management și monitorizare a aplicațiilor, care facilitează colectarea de metrice, diagnosticarea problemelor și verificarea stării aplicațiilor [26].

Prin capacitatea sa de a crea aplicații autonome și independente, care pot fi scalate orizontal, Spring Boot este o opțiune populară pentru dezvoltatorii de microservicii. Microserviciile care sunt construite cu Spring Boot profită de toate avantajele ecosistemului Spring, cum ar fi gestiunea tranzacțiilor, integrarea cu servicii externe și suportul pentru mai multe baze de date [25].

Comunitatea activă și resursele extinse disponibile pentru Spring Boot contribuie la ușurința învățării și utilizării acestuia. Documentația oficială, tutorialele, cursurile online și forumurile de discuții sunt resurse valoroase pentru cei care doresc să învețe sau să își îmbunătățească cunoștințele despre Spring Boot [24].

### React

A blue atom with black text

Description automatically generated

React este o bibliotecă JavaScript open-source utilizată pentru a construi interfețe de utilizator(UI) sau componente UI. A fost dezvoltată inițial de Facebook în 2011 și lansată publicului in 2013, devenind rapid una dintre cele mai populare biblioteci pentru dezvoltarea de aplicații web datorită unei abordări inovative și datorită performanțelor ridicate [27].

Unul dintre conceptele cheie a acestei biblioteci este componentizarea. În React, interfețele de utilizator sunt construite din componente independete si reutilizabile ce pot fi combinate pentru a forma interfețe complexe. Aceasta abordare modulară facilitează crearea si întreținerea aplicațiilor, permițând dezvoltatorilor să izoleze și să reutilizeze componente [28].

React utilizează un concept numit Virtual DOM pentru a optimiza randarea si actualizarea interfețelor de utilizator. În loc să actualizeze în mod direct modelul de obiect al documentului (DOM) al browserului, React creează o reprezentare virtuală a DOM și realizează actualizări eficiente prin reconcilierea diferențelor dintre DOM-ul virtual și DOM-ul real. Acest proces reduce semnificativ costurile de performanță asociate cu manipularea DOM și asigură o experiență de utilizator mai rapidă și fluidă [29].

React este cunoscut și pentru utilizarea extensivă a JSX (JavaScript XML), care permite dezvoltatorilor să scrie cod HTML direct în JavaScript. JSX face codul mai intuitiv și mai ușor de înțeles, combinând logica de randare a componentelor cu structura UI într-un singur loc [28].

Un alt avantaj al React este ecosistemul său bogat și flexibilitatea în integrarea cu alte biblioteci și framework-uri. React poate fi folosit pentru a construi aplicații cu o singură pagină (SPA) utilizând biblioteci suplimentare, cum ar fi React Router pentru navigare și Redux pentru gestionarea stării. În plus, React poate fi integrat cu alte cadre backend și utilizat pentru a dezvolta aplicații mobile prin React Native [27].

Arhitectura robustă și optimizările de performanță React îl fac o alegere ideală pentru dezvoltarea aplicațiilor web moderne. Accentul bibliotecii pe dezvoltarea bazată pe componente, randarea eficientă prin Virtual DOM și flexibilitatea în integrarea cu alte tehnologii oferă dezvoltatorilor instrumentele necesare pentru a construi interfețe de utilizator dinamice și de înaltă performanță. În plus, documentația oficială extinsă, tutorialele cuprinzătoare și numeroasele resurse online asigură dezvoltatorilor suportul de care au nevoie pentru a implementa și extinde cu succes proiectele lor utilizând React [27].

### MySQL

A logo of a dolphin

Description automatically generated

MySQL este un sistem de gestionare a bazelor de date relaționale (RDBMS) foarte popular, cunoscut pentru eficiența și flexibilitatea sa. Dezvoltat inițial de MySQL AB, a fost achiziționat de Sun Microsystems în 2008 și ulterior de Oracle Corporation în 2010. MySQL este utilizat pe scară largă în aplicații web și software-uri enterprise datorită performanței sale ridicate și a capacității de scalare.

#### Caracteristici Cheie ale MySQL

MySQL oferă o serie de caracteristici care îl fac ideal pentru utilizarea în aplicații web și software-uri enterprise.

Performanță Ridicată: MySQL este optimizat pentru a oferi viteze mari de citire și scriere, fiind capabil să gestioneze volume mari de date și multiple conexiuni simultane. Aceasta îl face deosebit de eficient pentru aplicațiile web cu trafic intens, unde performanța rapidă este esențială pentru o experiență de utilizare fără întreruperi [30] [31].

Compatibilitate și Integrare: MySQL este compatibil cu o gamă largă de limbaje de programare și platforme. Oferă suport pentru diverse API-uri și drivere, inclusiv pentru limbaje populare precum Python, Java, PHP și C#. Aceasta facilitează integrarea cu alte tehnologii și dezvoltarea de aplicații complexe care necesită acces la baze de date fiabile [30].

Securitate: MySQL include măsuri de securitate robuste pentru a proteja datele utilizatorilor. Aceste măsuri includ autentificarea utilizatorilor, criptarea datelor și controlul accesului bazat pe roluri. MySQL suportă, de asemenea, protocoale de securitate precum SSL/TLS pentru a proteja datele în tranzit, asigurând astfel integritatea și confidențialitatea informațiilor sensibile [32].

Scalabilitate: Proiectat pentru a crește odată cu nevoile afacerii, MySQL permite configurarea eficientă pentru date distribuite și replicarea datelor. Aceasta asigură disponibilitatea și performanța optimă a aplicațiilor, indiferent de volumul de date sau de numărul de utilizatori simultani. MySQL poate fi scalat vertical sau orizontal, oferind flexibilitate în funcție de cerințele specifice ale proiectului [31].

Administrare Ușoară: MySQL oferă instrumente de administrare prietenoase, cum ar fi MySQL Workbench, care permit gestionarea vizuală a bazelor de date. Aceste instrumente simplifică sarcini precum proiectarea bazei de date, modelarea datelor, administrarea utilizatorilor și monitorizarea performanței. MySQL Workbench oferă, de asemenea, un set complet de funcționalități pentru dezvoltare SQL, administrare a bazei de date și configurare a serverului [30].

Licențiere: MySQL este disponibil sub două modele de licențiere: licența publică generală GNU (GPL) pentru utilizare gratuită și licențele comerciale pentru medii enterprise care necesită suport tehnic dedicat și funcționalități avansate. Acest model dual de licențiere permite utilizatorilor să aleagă opțiunea care se potrivește cel mai bine nevoilor și bugetului lor, beneficiind în același timp de suport profesional și actualizări constante [32].

#### Utilizări Comune ale MySQL

MySQL este larg utilizat în diverse domenii datorită flexibilității și performanței sale ridicate. În aplicațiile web, MySQL servește ca baza de date pentru multe site-uri populare, inclusiv Facebook, Twitter, YouTube și Flickr. Aceste platforme beneficiază de capacitatea MySQL de a gestiona volume mari de date și de a suporta multiple conexiuni simultane, oferind o experiență de utilizare rapidă și fiabilă. În platformele de comerț electronic, precum Magento și WooCommerce, MySQL este utilizat pentru gestionarea produselor, comenzilor și utilizatorilor, asigurând un management eficient și securizat al datelor comerciale. De asemenea, în sistemele de management al conținutului (CMS) precum WordPress, Joomla și Drupal, MySQL stochează și gestionează conținutul site-urilor, facilitând actualizări rapide și accesibilitatea informațiilor pentru utilizatori și administratori [30] [31].

#### Beneficii ale Utilizării MySQL

MySQL oferă numeroase beneficii, făcându-l o alegere populară pentru dezvoltatori și organizații. Fiind un software open-source, MySQL oferă flexibilitate și costuri reduse, permițând personalizarea și adaptarea la nevoile specifice ale utilizatorilor. Comunitatea activă de dezvoltatori și utilizatori contribuie la îmbunătățirea constantă a software-ului, oferind suport prin forumuri și documentație detaliată. Actualizările regulate lansate de Oracle Corporation și de comunitate adaugă noi funcționalități și îmbunătățesc securitatea și performanța MySQL, asigurându-se că rămâne un instrument modern și eficient pentru gestionarea bazelor de date. Aceasta face ca MySQL să fie nu doar o soluție fiabilă, dar și una în continuă evoluție, capabilă să răspundă cerințelor variate și dinamice ale industriei [32].

### Apache Maven

A colorful feather with black text

Description automatically generated

Apache Maven este un instrument esențial pentru gestionarea și înțelegerea proiectelor de software, utilizat în principal în dezvoltarea proiectelor Java. Maven oferă un model complet pentru construirea, gestionarea și raportarea proiectelor software.

Unul dintre principalele beneficii ale utilizării Maven este gestionarea dependențelor. Prin specificarea și gestionarea dependențelor externe într-un mod centralizat, folosind fișierul pom.xml (Project Object Model), Maven asigură că toate bibliotecile și plugin-urile necesare proiectului sunt disponibile. Acest fișier de configurare descrie toate componentele necesare proiectului, iar Maven le descarcă automat dintr-un repository central sau local. Acest lucru elimină problemele legate de versiuni incompatibile sau lipsa unor biblioteci esențiale, asigurând că aplicația poate fi compilată și rulată fără probleme.

Maven automatizează procesul de build al aplicației, care include compilarea codului sursă, rularea testelor, pachetarea aplicației (de exemplu, într-un fișier JAR sau WAR) și instalarea acesteia în repository-ul local. Fiecare etapă a procesului de build este definită în lifecycle-ul standard al Maven, care cuprinde faze precum validate, compile, test, package, verify, install, și deploy. Această automatizare facilitează reproducerea build-urilor și asigură consistența între diferitele medii de dezvoltare și producție.

Acesta suportă, de asemenea, gestionarea proiectelor multi-modul, permițând organizarea unui proiect mare în module mai mici și mai gestionabile. Fiecare modul are propriul fișier pom.xml, dar toate modulele sunt gestionate centralizat de un fișier pom.xml părinte. Această structură modulară ajută la separarea preocupărilor și simplifică dezvoltarea și întreținerea proiectului.

Un alt avantaj al utilizării Maven este integrarea ușoară cu principalele medii de dezvoltare integrate (IDE-uri), cum ar fi IntelliJ IDEA, Eclipse și NetBeans. Această integrare permite dezvoltatorilor să gestioneze dependențele, să ruleze comenzi Maven și să vizualizeze structura proiectului direct din IDE, îmbunătățind astfel eficiența dezvoltării și simplificând gestionarea proiectului.

Maven are o arhitectură extensibilă bazată pe plugin-uri, care sunt disponibile pentru diverse sarcini, cum ar fi compilarea codului, rularea testelor, generarea documentației și multe altele. Aceste plugin-uri pot fi configurate și personalizate în fișierul pom.xml, permițând adaptarea procesului de build la nevoile specifice ale proiectului.

Prin utilizarea Maven, proiectul beneficiază de gestionarea eficientă a dependențelor, automatizarea procesului de build și integrarea facilă cu diverse medii de dezvoltare. Acest lucru asigură o structură de proiect bine organizată și ușor de întreținut, contribuind astfel la succesul dezvoltării și implementării aplicației.

### JSON Web Token

JSON Web Token (JWT) este un standard deschis pentru transmiterea de informații între părți sub forma unui obiect JSON compact și securizat. JWT-urile sunt utilizate pe scară largă pentru autentificare și autorizare în aplicațiile web și API-uri, oferind un mecanism sigur și eficient pentru transferul datelor de identitate și acces [33].

JWT-urile au o structură specifică, formată din trei părți: header, payload și signature. Fiecare dintre aceste componente joacă un rol crucial în funcționarea token-ului. Header-ul conține informații despre tipul de token și algoritmul de criptare utilizat. Payload-ul include declarații (claims), care sunt informații despre utilizator și alte date necesare pentru a descrie starea autentificării. Aceste declarații pot fi de trei tipuri: registered claims (declarații înregistrate), public claims (declarații publice) și private claims (declarații private) [34].

Signature-ul este utilizat pentru a verifica autenticitatea token-ului și pentru a asigura că datele nu au fost modificate. Acest lucru se realizează prin combinarea header-ului și payload-ului, împreună cu o cheie secretă, și aplicarea unui algoritm de criptare. Verificarea signature-ului permite destinatarilor token-ului să fie siguri că datele sunt intacte și provin dintr-o sursă de încredere [34].

JWT-urile oferă mai multe avantaje importante. Ele sunt compacte, ceea ce le face ușor de transmis prin URL-uri, parametri de POST și header-ele HTTP. De asemenea, JWT-urile sunt auto-conținute, ceea ce înseamnă că toate informațiile necesare sunt incluse în token, eliminând necesitatea de a accesa o bază de date de fiecare dată când se face o solicitare. Acest lucru duce la o performanță îmbunătățită și o scalabilitate mai mare a aplicațiilor [34].

În concluzie, JSON Web Token reprezintă un mecanism puternic și flexibil pentru autentificare și autorizare, utilizat pe scară largă în dezvoltarea aplicațiilor moderne. Capacitatea sa de a furniza un format compact și securizat pentru transferul informațiilor de identitate și acces îl face un instrument valoros pentru dezvoltatori și administratori de sisteme [35].

### Arhitectura Layered

Arhitectura layered, cunoscută și sub denumirea de arhitectură pe straturi, este un model de design utilizat pe scară largă în dezvoltarea software-ului. Acest model împarte aplicația în straturi distincte, fiecare având un set specific de responsabilități. Separarea clară a acestor straturi facilitează dezvoltarea, întreținerea și scalabilitatea aplicațiilor software [36].

Un sistem tipic bazat pe arhitectura layered include mai multe straturi esențiale:

* Stratul de Prezentare: Acesta este stratul responsabil de interacțiunea cu utilizatorul final. Include interfața grafică (GUI) și logica necesară pentru afișarea datelor și preluarea input-ului utilizatorului. În cazul aplicațiilor web, acest strat utilizează tehnologii precum HTML, CSS și JavaScript, în timp ce pentru aplicațiile desktop sunt utilizate framework-uri specifice platformei [37].
* Stratul de Logică a Aplicației: Acest strat, cunoscut și ca stratul de business logic, gestionează regulile de afaceri și logica principală a aplicației. Acesta procesează datele, implementează regulile de afaceri și servește ca intermediar între stratul de prezentare și cel de acces la date. Astfel, asigură manipularea corectă a datelor înainte de a fi prezentate utilizatorului sau stocate în baza de date [37].
* Stratul de Acces la Date: Responsabil pentru interacțiunea cu sursele de date, acest strat se ocupă de operațiunile de citire și scriere în baza de date. Utilizează tehnologii precum SQL pentru manipularea bazelor de date relaționale și asigură accesul eficient și securizat la date [38].
* Stratul de Servicii (opțional): În anumite arhitecturi, există și un strat de servicii care expune funcționalitățile aplicației sub formă de servicii reutilizabile. Acest strat facilitează integrarea cu alte sisteme și permite expunerea API-urilor pentru utilizare externă, ceea ce îmbunătățește modularitatea și reutilizabilitatea componentelor aplicației.

Fiecare strat din arhitectura layered este proiectat pentru a interacționa doar cu stratul imediat inferior sau superior, menținând astfel o separare clară a responsabilităților. Această abordare aduce numeroase avantaje, printre care o gestionare mai bună a complexității, posibilitatea de reutilizare a componentelor și facilitarea testării și întreținerii aplicației.

Arhitectura layered este preferată datorită flexibilității și scalabilității pe care le oferă. Este aplicabilă atât în dezvoltarea aplicațiilor de mici dimensiuni, cât și în proiectele software complexe. Separarea responsabilităților în straturi asigură o structură coerentă și ușor de gestionat, ceea ce facilitează evoluția și extinderea sistemului pe termen lung [36].

### Node Package Manager

A red and white logo

Description automatically generated

Node Package Manager (NPM) este un instrument esențial pentru dezvoltatorii care utilizează JavaScript și Node.js. Acesta simplifică procesul de gestionare a dependențelor proiectului și facilitează instalarea, actualizarea și configurarea pachetelor software. NPM a fost dezvoltat alături de Node.js și a devenit o parte esențială a ecosistemului JavaScript [39].

Printre caracteristicile cheie se numără:

1. **Gestionarea Pachetelor**: NPM permite dezvoltatorilor să gestioneze ușor dependențele proiectului prin instalarea bibliotecilor și a cadrelor necesare pentru dezvoltare. Acest lucru se realizează prin comenzi simple, facilitând adăugarea, actualizarea sau eliminarea pachetelor după cum este necesar [40].
2. **Controlul Versiunilor:** NPM asigură utilizarea versiunilor corecte ale dependențelor în proiecte. Acest lucru este util pentru prevenirea conflictelor de versiune și asigurarea compatibilității, deoarece NPM poate bloca dependențele la versiuni specifice [39].
3. **Scripturi și Automatizare:** NPM suportă definirea scripturilor în fișierul package.json, care pot automatiza sarcinile repetitive, cum ar fi testarea, construirea și implementarea aplicațiilor. Această caracteristică îmbunătățește productivitatea prin simplificarea fluxurilor de lucru [39].
4. **Registru Mare de Pachete:** Registrul NPM găzduiește o vastă colecție de pachete open-source, fiind cel mai mare registru de software din lume. Dezvoltatorii pot găsi aproape orice bibliotecă sau instrument de care au nevoie, contribuind la procesele de dezvoltare mai rapide și mai eficiente [39].
5. **Comunitate și Suport:** NPM beneficiază de o comunitate robustă de dezvoltatori care contribuie cu pachete și oferă suport prin forumuri și documentație. Acest suport extins ajută dezvoltatorii să rezolve rapid problemele și să implementeze cele mai bune practici [40].
6. **Funcții de Securitate:** NPM include instrumente de auditare a securității care ajută la identificarea și remedierea vulnerabilităților din dependențe. Aceasta asigură că aplicațiile rămân sigure prin abordarea promptă a riscurilor de securitate potențiale [40].

Adoptarea pe scară largă a NPM se datorează integrării sale perfecte cu Node.js și capacității sale de a simplifica gestionarea dependențelor. Este utilizat pe scară largă atât în proiecte mici, cât și în aplicații de mare amploare. Companii precum Netflix, LinkedIn și PayPal folosesc NPM pentru a-și gestiona proiectele JavaScript, demonstrând fiabilitatea și eficiența acestuia în medii de producție.

## Proiectare și implementare

### Arhitectura conceptuala

A diagram of a software company

Description automatically generated

Figura 3.1 Arhitectura sistemului

În figura 3.1 este prezentată o diagrama care reprezintă arhitectura conceptuală a sistemului. Această diagramă cuprinde elementele componente ale acestui sistem:

* Client
* Server
* Baza de date
* Script-uri Python

Prima componentă, client-ul, reprezintă componenta de front-end a acestui sistem, realizată folosind biblioteca React. Această componentă pune la dispoziția utilizatorilor interfața grafică, ascunzând toată complexitatea din spate. Componentele React sunt responsabile pentru afișarea informațiilor și colectarea input-ului utilizatorilor. Aceste componente sunt construite utilizând JSX pentru a defini elementele UI și stilurile lor. Trimite solicitări HTTP către backend-ul Spring Boot și primește răspunsuri în format JSON. Aceste date sunt apoi folosite pentru a actualiza starea componentelor și pentru a re-randa UI-ul în mod corespunzător.

Utilizatorii introduc URL-urile produselor și prețurile de prag în interfața React. Aplicația React trimite apoi solicitări HTTP către serverul Spring Boot, care se află pe backend. Răspunsurile sunt primite în format JSON de la serverul Spring Boot, iar aceste date sunt folosite pentru a actualiza interfața utilizatorului în timp real.

A doua componentă, serverul, reprezintă componenta back-end a sistemului. Serverul (back-end) este realizat folosind framework-ul Spring Boot, care oferă un ecosistem robust și scalabil pentru dezvoltarea aplicațiilor enterprise. Această componentă gestionează request-urile primite de la client și oferă răspunsuri adecvate, interacționând cu baza de date pentru a obține sau a stoca informațiile necesare. Design-ul arhitectural aplicat este Layered, structurat pe mai multe nivele:

1. Controller: La acest nivel sunt definite metodele RESTful asociate endpoint-urilor create. Aceste metode primesc cererile HTTP de la client și le procesează.
2. Service: Acest nivel conține logica aplicației și comunică prin intermediul Data Transfer Objects (DTOs). Aici sunt implementate regulile și procesele de afaceri ale aplicației.
3. Repository: Acest nivel face translatarea informațiilor extrase din baza de date. Comunicarea cu nivelul de servicii se face prin entități, punându-i acestuia la dispoziție toate datele necesare pentru construirea informației de transmis către nivelele superioare.
4. Domain: Acest strat conține entitățile de domeniu și obiectele de transfer de date (DTOs). Entitățile de domeniu reprezintă obiectele persistente și structurile lor de date.

Serverul Spring Boot primește solicitările HTTP de la clientul React. Acesta interacționează cu baza de date MySQL pentru a stoca sau recupera informațiile necesare, cum ar fi detalii despre produse și prețurile de prag setate de utilizatori. După procesarea solicitărilor, serverul trimite răspunsuri în format JSON înapoi către clientul React, asigurându-se că interfața utilizatorului este actualizată cu cele mai recente informații.

A treia componentă este baza de date, unde are loc persistența datelor din aplicație. Baza de date este gestionată de MySQL, un sistem de gestionare a bazelor de date relaționale cunoscut pentru performanța și fiabilitatea sa. Aceasta componentă stochează informațiile despre utilizatori, produse și pragurile de preț. Componenta de back-end comunică cu baza de date prin nivelul de Repositories, asigurându-se că toate datele sunt gestionate corect și eficient.

Baza de date MySQL joacă un rol crucial în stocarea informațiilor despre produse, inclusiv URL-uri și prețuri de prag. Aceasta este actualizată periodic de scripturile Python pentru a reflecta cele mai recente prețuri ale produselor. De asemenea, backend-ul Spring Boot interoghează baza de date pentru a obține datele necesare pentru răspunsurile către clientul React, asigurându-se că utilizatorii au acces la informații corecte și actualizate.

Scripturile Python joacă un rol crucial în acest sistem. Scriptul "addProduct" are funcția de a adăuga produse noi în baza de date MySQL, trimițând datele colectate către baza de date pentru a fi stocate. Acest script colectează datele produselor și le trimite către baza de date MySQL pentru a fi stocate. Asigură că noile produse sunt adăugate corect în baza de date, facilitând astfel gestionarea eficientă a catalogului de produse.

De asemenea, scriptul "updateProducts" este responsabil pentru actualizarea prețurilor produselor deja existente în baza de date, asigurând astfel că informațiile despre prețuri sunt mereu actualizate și corecte. Acest script este responsabil pentru actualizarea datelor prețurilor produselor existente în baza de date MySQL. Se asigură că informațiile despre prețuri sunt actualizate și corecte, contribuind la menținerea integrității datelor și la funcționarea eficientă a sistemului de notificare a scăderii prețurilor. Aceste scripturi facilitează menținerea integrității și acurateței datelor stocate, contribuind la funcționarea eficientă a sistemului de notificare a scăderii prețurilor.

În ansamblu, aceste componente lucrează împreună pentru a asigura un flux de date eficient și pentru a oferi utilizatorilor notificări precise și în timp real despre scăderile de prețuri ale produselor de interes.

### Cazuri de utilizare

În cadrul acestei secțiuni, se vor prezenta principalele cazuri de utilizare ale sistemului, evidențiind interacțiunile dintre utilizatori și sistem. Se vor descrie două tipuri de utilizatori:

* utilizatori autentificați
* utilizatori neautentificați

Fiecare categorie de utilizatori are acces la diferite funcționalități ale aplicației. Utilizatorul autentificat are acces la cele mai multe funcționalități, cel neautentificat fiind limitat doar la a naviga până într-un anumit punct, unde i se va cere să se logheze. Pentru ca utilizatorul neautentificat să devină interesat de aplicație și de funcționalitățile acesteia, acesta poate să navigheze până într-un anumit punct, după care trebuie să se autentifice.

Astfel, pentru a ilustra funcționalitățile sistemului pentru fiecare utilizator, figurile următoare prezintă posibilele acțiuni ale utilizatorilor.

#### Utilizator autentificat

A diagram of a product

Description automatically generated

Figura 3.2 Diagrama UML Use Case Utilizator Autentificat

Utilizatorul autentificat are acces la următoarele acțiuni:

* Înregistrare
* Autentificare
* Adăugarea unui produs
* Vizualizarea produselor favorite
* Ștergerea unui produs
* Actualizarea prețului de prag pentru un anumit produs
* Deconectarea din cont

#### Utilizator neautentificat

A diagram of a person's choice

Description automatically generated with medium confidence

Figura 3.3 Diagrama UML Use Case Utilizator neautentificat

Utilizatorul neautentificat este mai limitat în ceea ce privește acțiunile pe care le poate realiza. Acesta poate doar să realizeze următoarele acțiuni:

* Înregistrare
* Vizualizare pagină de start

### Descrierea cazurilor de utilizare

#### Cazul 1: Înregistrare

**Nume:** Înregistrare

**Actor:** Utilizator ne/înregistrat

**Descriere:** Utilizatorii care nu sunt înregistrați în aplicație pot să realizeze doar anumite operațiuni, de aceea este necesar ca aceștia să se înregistreze pentru a putea să realizeze mai multe operații.

**Precondiții:** Unicitatea numelui de utilizator și a email-ului. Să nu se găsească în baza de date a aplicației același email sau nume de utilizator cu care se încearcă înregistrarea.

**Postcondiții:**

* Contul de utilizator este creat iar informațiile acestuia sunt salvate in baza de date
* Utilizatorul este redirecționat către pagina de start

**Fluxul de evenimente:**

1. Utilizatorul accesează pagina de înregistrare.
2. Se afișează un formular de înregistrare care conține următoarele câmpuri: nume de utilizator, email și parolă.
3. Utilizatorul completează toate câmpurile din formular.
4. Utilizatorul apasă butonul de trimitere a formularului.
5. Utilizatorul este redirecționat către pagina de logare.

**Flux alternativ de evenimente:**

1. Utilizatorul accesează pagina de înregistrare.
2. Se afișează un formular de înregistrare care conține următoarele câmpuri: nume de utilizator, email și parolă.
3. Utilizatorul completează câmpurile din formular, mai puțin unul sau mai multe câmpuri sau valorile introduse nu sunt compatibile cu cerințele.
4. Utilizatorul este avertizat că trebuie să completeze corect toate câmpurile.
   1. Utilizatorul corectează problemele și este înregistrat cu succes.
   2. Utilizatorul abandonează înregistrarea.

În figura 3.4 este prezentat fluxul de evenimente pentru înregistrarea în aplicație a unui utilizator.

A diagram of a flowchart

Description automatically generated

Figura 3.4 Diagrama de evenimente pentru înregistrare

#### Cazul 2: Adăugare Produs

**Nume:** Adăugare Produs

**Actor:** Utilizator înregistrat

**Descriere:** Obiectivul principal al unui utilizator înregistrat este acela de a urmării scăderea de preț pentru un anumit produs. Pentru a putea fi notificat în legătură cu scăderea de preț, utilizatorul trebuie să adauge produsul la favorite și să-i seteze un preț de prag. În acest scop, utilizatorul trebuie să urmeze câțiva pași.

**Precondiții:**

* Utilizatorul să fie autentificat în aplicație.
* Utilizatorul să se afle pe pagina „Products”.

**Postcondiții:**

* Utilizatorul poate viziona produsul adăugat, pe pagina „Profile”.

**Fluxul de evenimente:**

1. Pe ecran este prezent un formular care cere completarea câmpurilor pentru URL-ul produsului și prețul de prag.
2. Utilizatorul completează toate câmpurile din acest formular.
3. Utilizatorul apasă butonul „Submit”.
4. Sistemul prezintă un mesaj de alertă, confirmând adăugarea produsului la favorite.

**Flux alternativ de evenimente:**

1. Pe ecran este prezent un formular care cere completarea câmpurilor pentru URL-ul produsului și prețul de prag.
2. Utilizatorul completează câmpurile din formular, mai puțin unul sau mai multe câmpuri sau valorile introduse nu sunt compatibile cu cerințele.
3. Utilizatorul este avertizat că trebuie să completeze corect toate câmpurile.
   1. Utilizatorul corectează problemele și produsul este adăugat cu succes.

4.2 Utilizatorul abandonează acțiunea.

În figura 3.5 este prezentat fluxul de evenimente pentru adăugarea la favorite a unui produs de către utilizator.

A diagram of a company

Description automatically generated

Figura 3.5 Diagrama de evenimente pentru adăugare produs

#### Cazul 3: Ștergere Produs

**Nume:** Ștergere Produs

**Actor:** Utilizator înregistrat

**Descriere:** Utilizatorii înregistrați au posibilitatea de a sterge un anumit produs de la favorite, dacă aceștia consideră că nu mai au nevoie să urmărească prețul acestuia. Pentru a șterge un produs, utilizatorul trebuie să urmeze anumiți pași.

**Precondiții:**

* Utilizatorul să fie autentificat în aplicație.
* Utilizatorul să se afle pe pagina „Profile”.
* Utilizatorul trebuie să aibă cel puțin un produs adăugat la favorite.

**Postcondiții:**

* Utilizatorul poate observa că produsul a dispărut din lista de produse,de pe pagina „Profile”.

**Fluxul de evenimente:**

1. Utilizatorul poate să vadă lista de produse, fiecare produs având pictogramă sub formă de coș de gunoi sub imaginea produsului.
2. Utilizatorul apasă pe pictograma sub formă de coș de gunoi.
3. Produsul este șters din baza de date.
4. Sistemul prezintă un mesaj de alertă, confirmând ștergerea produsului din baza de date.

**Flux alternativ de evenimente:**

1. Utilizatorul poate să vadă lista de produse, fiecare produs având pictogramă sub formă de coș de gunoi sub imaginea produsului.
2. Utilizatorul nu apasă pe pictograma sub formă de coș de gunoi.
3. Utilizatorul abandonează acțiunea.
4. Utilizatorul părăsește pagina.

În figura 3.6 este prezentat fluxul de evenimente pentru adăugarea la favorite a unui produs de către utilizator.

A diagram of a process

Description automatically generated

Figura 3.6 Diagrama de evenimente pentru adăugare produs

#### Cazul 4: Actualizare Preț de Prag

**Nume:** Actualizare preț de prag

**Actor:** Utilizator înregistrat

**Descriere:** Utilizatorii înregistrați au posibilitatea să actualizeze prețul de prag, în cazul în care aceștia consideră că prețul inițial a fost prea mic, iar produsul nu va ajunge niciodată la acel preț. Pentru a actualiza prețul de prag al unui produs, utilizatorul trebuie să urmeze anumiți pași.

**Precondiții:**

* Utilizatorul să fie autentificat în aplicație.
* Utilizatorul să se afle pe pagina „Profile”.
* Utilizatorul trebuie să aibă cel puțin un produs adăugat la favorite.

**Postcondiții:**

* Utilizatorul poate să observe că prețul de prag s-a actualizat.

**Fluxul de evenimente:**

1. Utilizatorul poate să vadă lista de produse pe pagina „Profile”.
2. Utilizatorul apasă pe imaginea produsului.
3. Utilizatorul este redirecționat pe pagina produsului.
4. Utilizatorul modifică valoarea din câmpul „Threshold Price”.
5. Utilizatorul apasă butonul „Update Threshold Price”.
6. Prețul de prag al produsului este modificat și în baza de date.
7. Sistemul prezintă un mesaj de alertă, confirmând actualizarea prețului de prag.

**Flux alternativ de evenimente:**

1. Utilizatorul poate să vadă lista de produse pe pagina „Profile”.
2. Utilizatorul nu apasă pe imaginea produsului.
3. Utilizatorul abandonează acțiunea.
4. Utilizatorul părăsește pagina.

În figura 3.7 este prezentat fluxul de evenimente pentru actualizarea prețului de prag al unui produs din lista de favorite.

A diagram of a flowchart

Description automatically generated

Figura 3.7 Diagrama de evenimente pentru adăugare produs

### Arhitectura componentei backend

Componenta backend a sistemului este dezvoltată utilizând framework-ul Spring Boot. Acest framework este ales datorită capacității sale de a facilita dezvoltarea aplicațiilor enterprise în Java, oferind un mediu robust și scalabil. Necesitatea unui backend puternic și ușor de gestionat a determinat alegerea Spring Boot, cunoscut pentru eficiența sa în gestionarea proiectelor complexe și pentru suportul comunității sale active.

În figura 3.8 este prezentată diagrama de pachete a componentei backend a aplicației, organizată conform principiilor arhitecturii layered. Diagrama ilustrează structura modulară a proiectului backend și arată cum sunt grupate diferitele clase și interfețe în pachete distincte pentru a asigura claritatea, mentenabilitatea și scalabilitatea codului.

A computer screen shot of a computer screen

Description automatically generated

Figura 3.8 Diagrama de pachete pentru componenta backend

Componenta de backend (server-ul), este formată din șapte pachete, fiecare având un rol specific în funcționarea aplicației. Aceste pachete includ: controllers, services, repository, models, configuration, exceptions și utils.

Pachetul **controllers** este responsabil pentru gestionarea cererilor HTTP venite de la frontend. Aceste clase definesc endpoint-uri RESTful care mapează cererile utilizatorilor la operațiunile corespunzătoare din serviciile de business. De exemplu, atunci când un utilizator adaugă un nou produs pentru monitorizare, cererea HTTP POST este gestionată de un controler specific care validează datele și apelează serviciul relevant pentru a procesa și stoca informațiile în baza de date.

Pachetul **services** conține logica de afaceri a aplicației. Clasele din acest pachet sunt responsabile pentru procesarea datelor, aplicarea regulilor de afaceri și coordonarea interacțiunilor dintre diferitele componente ale sistemului. De exemplu, serviciile gestionează logica de notificare atunci când prețul unui produs scade sub pragul stabilit de utilizator, asigurându-se că notificările sunt trimise corect și la timp.

Pachetul **repository** include interfețele și clasele care interacționează cu baza de date MySQL. Aceste componente sunt configurate pentru a efectua operațiuni CRUD (Create, Read, Update, Delete) și pentru a accesa datele stocate în baza de date. De exemplu, un repository poate fi utilizat pentru a găsi toate produsele urmărite de un anumit utilizator și pentru a actualiza prețurile acestor produse în baza de date.

Pachetul **models** conține definițiile entităților de domeniu, care reprezintă obiectele persistente din baza de date. Aceste entități sunt adnotate pentru a defini structura tabelelor și relațiile dintre ele. De exemplu, entitatea Product include atribute precum id, title, price, productUrl și relațiile cu alte entități, cum ar fi UserProduct și PriceHistory.

Pachetul **configuration** cuprinde clasele care configurează diverse aspecte ale aplicației, cum ar fi setările de securitate, conexiunile la baza de date și alte configurări specifice framework-ului Spring Boot. De exemplu, acest pachet poate include configurări pentru autentificarea și autorizarea utilizatorilor, asigurându-se că doar utilizatorii autorizați au acces la anumite funcționalități ale aplicației.

Pachetul **exceptions** este dedicat gestionării excepțiilor și erorilor. Clasele din acest pachet capturează și tratează diverse tipuri de erori care pot apărea în timpul execuției aplicației. De exemplu, excepțiile legate de accesul neautorizat sau de validarea datelor sunt gestionate aici, oferind feedback adecvat utilizatorilor și logând erorile pentru analiză ulterioară.

Pachetul **utils** conține clase și metode utilitare care oferă funcționalități auxiliare folosite în diferite părți ale aplicației. Acestea pot include funcții de conversie, validare și alte operațiuni comune. De exemplu, utilitarele pot include metode pentru formatarea datelor, generarea de token-uri de securitate sau alte operațiuni repetitive necesare în diferite componente ale sistemului.

Fiecare pachet din componenta backend are un rol specific și bine definit, contribuind la funcționalitatea generală a aplicației. Controller-ele primesc cererile utilizatorilor și apelează serviciile pentru a executa logica de business. Serviciile, la rândul lor, utilizează repository-urile pentru a interacționa cu baza de date, iar modelele definesc structura datelor persistente. Configurările asigură funcționarea corectă și securizată a aplicației, în timp ce gestionarea excepțiilor și utilitarele contribuie la stabilitatea și eficiența sistemului.

Această structură modulară facilitează dezvoltarea clară și organizată a aplicației, permitând gestionarea eficientă a codului. Fiecare pachet are un rol bine definit, promovând separarea responsabilităților, reutilizarea codului și extensibilitatea sistemului.

A screenshot of a computer

Description automatically generated

Figura 3.9 Diagrama de clase a pachetului models

În figura 3.9 este prezentată diagrama de clase a pachetului models, care ilustrează structura principalelor entități și relațiile dintre acestea în cadrul sistemului. Aceasta evidențiază clasele și obiectele esențiale, cum ar fi ApplicationUser, Product, UserProduct, Role, și PriceHistory, precum și obiectele de transfer de date (DTOs) asociate acestora.

Diagrama de clase a pachetului models include entitățile esențiale care reprezintă utilizatorii, produsele, istoricul prețurilor și rolurile utilizatorilor. Fiecare dintre aceste entități este definită cu atribute specifice și metode de acces (getter și setter) care permit gestionarea eficientă a datelor.

Clasa **ApplicationUser** reprezintă utilizatorii aplicației și gestionează autentificarea și autorizarea acestora. Atributele sale includ userid, username, password, email și roles. Metodele getter și setter asociate fiecărui atribut permit accesul și modificarea datelor utilizatorilor. Această clasă include, de asemenea, metode pentru verificarea stării contului, cum ar fi isAccountNonExpired, isAccountNonLocked, isCredentialsNonExpired și isEnabled, asigurând astfel gestionarea securității conturilor utilizatorilor.

Clasa **Product** stochează informațiile esențiale despre produsele monitorizate de utilizatori. Atributele principale includ id, title, price, productUrl și imageUrl. Metodele getter și setter permit accesul și actualizarea acestor informații. Această clasă menține, de asemenea, relația many-to-many cu clasa UserProduct, care facilitează asocierea produselor cu utilizatorii care le urmăresc.

Clasa **UserProduct** reprezintă relația dintre utilizatori și produsele pe care aceștia le monitorizează. Atributele includ id, userId, productId și thresholdPrice. Această clasă permite stocarea pragului de preț setat de utilizatori pentru fiecare produs și gestionează notificările de preț. Metodele getter și setter asigură accesul și modificarea acestor date, facilitând astfel funcționarea corectă a sistemului de notificare.

Clasa **Role** definește rolurile utilizatorilor în cadrul aplicației. Atributele includ role\_id și authority, iar metodele getter și setter permit accesul și modificarea acestor informații. Relația many-to-many dintre ApplicationUser și Role este gestionată prin intermediul tabelei de legătură user\_role\_junction, asigurând astfel flexibilitatea și extinderea sistemului de roluri în cadrul aplicației.

Clasa **PriceHistory** menține un istoric al fluctuațiilor de prețuri pentru fiecare produs. Atributele sale includ id, productId, date și price. Această clasă permite stocarea prețurilor istorice și a datelor corespunzătoare, facilitând astfel analiza și monitorizarea evoluției prețurilor în timp. Relația many-to-one dintre PriceHistory și Product asigură că fiecare înregistrare de preț este asociată corect cu produsul corespunzător.

**DTO-urile** joacă un rol crucial în transferul de date între straturile aplicației, facilitând separarea logicii de afaceri de interfața utilizatorului și de nivelul de persistență a datelor. Aceste obiecte sunt concepute pentru a transporta date între client și server într-un mod eficient și securizat, reducând astfel complexitatea operațiilor și asigurând integritatea informațiilor transferate. Designul claselor și relațiilor din pachetul models este orientat către menținerea integrității și consistenței datelor. Atributelor esențiale, cum ar fi cheile primare și străine, li se acordă o atenție deosebită pentru a garanta că toate referințele sunt corect menținute și că nu apar discrepanțe sau redundanțe în datele stocate.

A screen shot of a computer program

Description automatically generated

Figura 3.10 Conținutul fișierului “application.properties”

Configurarea componentei backend a aplicației este esențială pentru asigurarea conectivității cu baza de date, gestionarea securității și trimiterea de notificări prin email. În imaginea prezentată mai jos se găsește fișierul de configurare application.properties al framework-ului Spring Boot, care definește setările necesare pentru funcționarea corectă a aplicației.

Prima parte a fișierului de configurare specifică portul pe care rulează serverul Spring Boot. În acest caz, serverul este configurat să ruleze pe portul 8000. Această setare este importantă pentru a evita conflictele de porturi și pentru a permite accesul corect la serverul aplicației.

Conexiunea la baza de date MySQL este configurată prin intermediul unor setări detaliate în application.properties. URL-ul bazei de date, driver-ul JDBC, numele de utilizator și parola sunt specificate pentru a permite aplicației să se conecteze la baza de date. De asemenea, se specifică dialectul Hibernate pentru MySQL și opțiunile pentru generarea automată a schemelor bazei de date. Aceste setări includ:

* URL-ul bazei de date MySQL și specificarea creării bazei de date dacă aceasta nu există.
* Clasa driver-ului JDBC necesară pentru conectarea la baza de date.
* Credențialele (nume de utilizator și parolă) pentru accesul la baza de date.
* Platforma specifică Hibernate pentru MySQL și setările pentru afișarea interogărilor SQL și actualizarea automată a schemei bazei de date.

O altă parte crucială a configurării backend-ului este trimiterea de email-uri. Configurația pentru trimiterea email-urilor folosește serverul SMTP al Gmail. Aceste setări includ adresa serverului SMTP, portul, numele de utilizator, parola și alte proprietăți SMTP necesare pentru securitate. Este important de menționat că setările pentru starttls și auth sunt activate pentru a asigura că conexiunea SMTP este securizată.

Aceste configurații sunt esențiale pentru asigurarea funcționării corecte și securizate a aplicației. Configurarea automată a bazei de date permite crearea și gestionarea tabelelor fără intervenție manuală, ceea ce simplifică procesul de dezvoltare și mentenanță. De asemenea, configurarea pentru trimiterea email-urilor asigură că utilizatorii primesc notificări în timp util, îmbunătățind astfel experiența generală de utilizare a aplicației.

În concluzie, figura 3.10 reflectă o configurare bine structurat a aplicației Spring Boot, menită să asigure conectivitatea eficientă cu baza de date MySQL și capacitatea de a trimite email-uri prin serverul SMTP. Aceste setări contribuie la funcționarea robustă și securizată a aplicației, oferind o bază solidă pentru extinderea și îmbunătățirea ulterioară a funcționalităților sale.

### Arhitectura componentei frontend

Componenta frontend a aplicației este dezvoltată folosind biblioteca React, cunoscută pentru modularitatea și eficiența sa în construirea interfețelor de utilizator dinamice și interactive. Arhitectura frontend este organizată în mai multe pachete, fiecare având un rol bine definit, pentru a asigura claritatea și mentenabilitatea codului. Aceste pachete includ: Assets, Components, Helpers, Pages, și Styles.

Această componentă joacă un rol esențial în interacțiunea utilizatorilor cu aplicația. Utilizatorii pot introduce URL-urile produselor și prețurile de prag în interfața React. React trimite solicitări HTTP către backend-ul Spring Boot, care se află pe server. Răspunsurile sunt primite în format JSON și folosite pentru a actualiza starea componentelor și pentru a re-randa UI-ul în mod corespunzător. Acest flux de date asigură că utilizatorii beneficiază de informații actualizate în timp real, contribuind la o experiență de utilizare fluidă și eficientă.

Structura directorului de frontend, ilustrată în figura 3.10, este bine organizată pentru a permite o dezvoltare clară și eficientă. Această structură include mai multe directoare și fișiere esențiale pentru funcționarea și aspectul vizual al aplicației.

Directorul **Assets** conține resursele statice ale aplicației, cum ar fi imagini și iconuri utilizate în interfața grafică. Aceste resurse sunt cruciale pentru un design vizual atractiv și pentru menținerea consistenței elementelor UI. De exemplu, fișierele "app-logo.png", "email.png", "password.png" și "person.png" sunt folosite pentru a îmbunătăți aspectul vizual al paginilor aplicației.

Directorul **Components** include componentele reutilizabile ale aplicației, cum ar fi "Footer.js", "Navbar.js" și "ProductItem.js". Aceste componente sunt proiectate pentru a fi folosite în mai multe părți ale aplicației, promovând reutilizarea codului și menținerea unei structuri modulare. De exemplu, "Navbar.js" afișează bara de navigare în toate paginile aplicației, asigurând o navigare consistentă și intuitivă pentru utilizatori. "ProductItem.js" gestionează afișarea informațiilor despre produsele individuale în diverse secțiuni ale aplicației. Reutilizarea componentelor permite dezvoltarea rapidă și reducerea duplicării codului, contribuind la o mai bună mentenanță și extindere a aplicației.

Directorul **helpers** conține fișiere auxiliare, cum ar fi "alerts.ts", care oferă funcționalități suplimentare sau funcții de utilitate necesare în diverse părți ale aplicației. Aceste funcții sunt utilizate pentru a centraliza logica comună și pentru a facilita mentenanța codului.

Directorul **pages** include componentele principale ale paginilor aplicației, cum ar fi "FavoriteProduct.js", "Home.js", "LoginSignup.js", "Product.js" și "Profile.js". Fiecare componentă de pagină este responsabilă pentru gestionarea logicii și afișarea conținutului specific al paginii respective. Home.js" afișează pagina principală a aplicației, "LoginSignup.js" gestionează autentificarea și înregistrarea utilizatorilor, iar "Profile.js" permite utilizatorilor să își vizualizeze și să își editeze informațiile personale. Aceste componente sunt punctele de intrare pentru utilizatori, oferind interfețe dedicate pentru diferitele funcționalități ale aplicației. Fiecare pagină este construită astfel încât să ofere o experiență de utilizare optimă, concentrându-se pe funcționalitatea specifică necesară utilizatorului.

Directorul **styles** conține fișierele CSS asociate fiecărei componente, asigurând stilizarea și aspectul vizual al aplicației. Fiecare componentă are un fișier CSS dedicat, cum ar fi "FavoriteProduct.css" sau "Navbar.css", care definește stilurile specifice pentru acea componentă, asigurând un aspect consistent și atrăgător. De exemplu, "FavoriteProduct.css" definește stilurile pentru afișarea produselor favorite ale utilizatorilor, contribuind la o experiență de utilizare plăcută și intuitivă. Stilizarea coerentă și bine definită este esențială pentru a menține o interfață de utilizator modernă și atractivă.

Fluxul de date în componenta frontend este gestionat prin intermediul React, care utilizează state și props pentru a manipula și actualiza datele în timp real. Solicitările HTTP sunt trimise către backend folosind librării precum Fetch API, iar răspunsurile sunt procesate pentru a actualiza starea componentelor. Această arhitectură asigură o interacțiune fluidă între frontend și backend, permițând utilizatorilor să primească notificări și actualizări în timp real despre produsele și prețurile de interes.

A screenshot of a computer

Description automatically generated

Figura 3.11 Diagrama de pachete a componentei frontend

În concluzie, arhitectura componentei frontend a aplicației este concepută pentru a oferi o experiență de utilizare optimă, asigurând în același timp claritatea, modularitatea și scalabilitatea codului. Utilizarea React permite dezvoltarea rapidă și eficientă a interfeței de utilizator, facilitând extinderea și mentenanța aplicației pe termen lung.

### Structura bazei de date

Baza de date este o parte esențială a aplicației, folosită pentru gestionarea eficientă a datelor utilizatorilor și produselor monitorizate. Astfel, pentru a stoca aceste date, se folosește baza de date MySQL, fiind una dintre cele mai populare baze de date din lume și având numeroase avantaje prezentate în capitolul anterior, în secțiunea 3.1.5 .

În figura 3.12 este prezentată diagrama bazei de date, unde se pot observa cele 8 tabele, cât și relațiile dintre acestea. Baza de date a fost normalizată în conformitate cu a treia formă normală (3NF). Aceasta înseamnă că toate tabelele au fost organizate astfel încât să minimizeze redundanța datelor și să mențină integritatea referențială. Prin normalizare la acest nivel, se elimină dependențele tranzitive, ceea ce garantează că datele stocate sunt consistente și corecte, asigurând astfel o structură eficientă și coerentă pentru gestionarea informațiilor.

În acest proiect, baza de date este creată automat din aplicația de backend, utilizând adnotări specifice pentru entități și relațiile dintre acestea. Aceste adnotări permit configurarea automată a tabelelor și relațiilor din baza de date MySQL, bazându-se pe specificațiile definite în codul sursă. Parametrii necesari pentru conexiunea la baza de date, precum și alte setări de configurare, sunt specificați în fișierul de resurse al aplicației. Acest proces facilitează identificarea și crearea automată a tabelelor la inițializarea aplicației, asigurând coerența și corectitudinea structurii bazei de date.

A computer screen shot of a diagram

Description automatically generated

Figura 3.12 Diagrama bazei de date

#### Descrierea tabelelor

Baza de date utilizată în acest proiect se numește "dblic" și este concepută pentru a sprijini funcționalitatea aplicației de monitorizare și notificare a scăderilor de prețuri. Aceasta conține opt tabele, fiecare având un rol specific în gestionarea datelor de autentificare a utilizatorilor, produse și relațiile dintre acestea. Tabelele users, roles și user\_role\_junction sunt esențiale pentru autentificarea și autorizarea utilizatorilor, asigurând gestionarea corectă a rolurilor și permisiunilor.

**Tabela users** stochează informațiile utilizatorilor înregistrați în sistem și conține următoarele câmpuri:

* **user\_id:** este cheie primară de tip int și reprezintă un identificator unic pentru fiecare utilizator.
* **email:** este de tip varchar și stochează adresa de e-mail a utilizatorului.
* **password:** este de tip varchar și stochează parola criptată a utilizatorului.
* **username:** este de tip varchar și stochează numele de utilizator.

Această tabelă are o relație de One-to-Many cu tabela user\_product și Many-to-Many cu tabela roles prin tabela user\_role\_junction.

**Tabela roles** definește rolurile diferite care poti fi atribuite utilizatorilor,având o relație de Many-to-Many cu tabela users și fiind compusă din următoarele câmpuri:

* **role\_id:** este cheie primară de tip int și reprezintă un identificator unic pentru fiecare rol.
* **authority**: este de tip varchar și stochează numele rolului(ex: USER).

**Tabela user\_role\_junction** este un tabel de legătură care gestionează relația Many-to-Many dintre utilizatori și rolurile acestora. Această tabelă conține câmpurile de tip int pentru user\_id și role\_id, ambele reprezentând chei străine.

**Tabela products** stochează informațiile despre produsele pe care fiecare utilizator le urmărește și conține următoarele câmpuri:

* **id:** este cheie primară de tip int și reprezintă identificatorul unic pentru fiecare produs.
* **price:** este de tip varchar și stochează prețul actual al produsului.
* **product\_ul:** este de tip varchar și stochează URL-ul către pagina produsului.
* **title:** este de tip varchar și stochează numele produsului.
* **image\_url:** este de tip varchar și stochează URL-ul către imaginea produslui.

**Tabela user\_product** este un tabel care gestionează relația dintre utilizatori și produsele pe care aceștia le urmăresc. Acest tabel are o relație de Many-to-One cu tabela users și tabela products, fiind alcătuită din următoarele câmpuri:

* **id**: este cheie primară de tip int și reprezintă identificatorul unic pentru fiecare relație.
* **threshold\_price:** este de tip varchar și reprezintă prețul de prag setat de utilizator pentru produsul respectiv.
* **product\_id:** cheie străină de tip int, reprezentând identificatorul produsului.
* **user\_id:** cheie străină de tip int, reprezentând identificatorul utilizatorului.

**Tabela price\_history** este folosită pentru a stoca istoricul prețurilor pentru fiecare produs din baza de date, urmărind fluctuațiile de preț în timp. Aceasta are o relație de many-to-oen cu tabela products, permițând înregistrarea multiplelor prețuri pentru fiecare produs în diferite momente de timp. Câmpurile prezente în acest tabel sunt:

* id: este cheie primară de tip int și reprezintă identificatorul unic pentru fiecare produs.
* product\_id: cheie străină de tip int, reprezentând identificatorul produsului.
* date: este de tip date și reprezintă data și ora la care a fost stocat prețul.
* price: este de tip varchar și reprezintă prețul produsului la acel timp.

Pentru a gestiona identificatorii unici ai fiecărui tabel, se utilizează secvențe specifice, numite secvențe de generare.

* **roles\_seq**
* **users\_seq**
* **products\_seq**

Aceste secvențe asigură generarea de identificatori unici pentru tabelele roles, users și products, facilitând astfel gestionarea corectă și eficientă a datelor.

Baza de date " dblic " este concepută pentru a facilita gestionarea eficientă a datelor referitoare la utilizatori și produse, precum și a relațiilor dintre acestea. Structura sa bine definită asigură atât scalabilitatea, cât și performanța necesară pentru a susține funcționalitatea aplicației de monitorizare a prețurilor. Această organizare detaliată permite o administrare simplă și eficientă a informațiilor, garantând în același timp că aplicația poate crește și funcționa fără probleme pe măsură ce numărul de utilizatori și produse urmărite crește.

### Scripturi Python folosite pentru gestionarea produselor

În cadrul componentei backend a aplicației, două scripturi Python sunt esențiale pentru monitorizarea și gestionarea prețurilor produselor: unul pentru adăugarea de produse în baza de date și altul pentru actualizarea prețurilor produselor existente. Aceste scripturi folosesc Selenium pentru web scraping și MySQL pentru stocarea și actualizarea datelor.

#### Script pentru Adăugarea Produselor

Acest script colectează informații despre produse de pe site-urile de comerț electronic și le stochează în baza de date MySQL. Funcționează astfel:

1. Extracția Prețului: Utilizând expresii regulate, scriptul identifică și extrage prețul produsului din textul paginii web. Sunt utilizate două tipare pentru a acoperi formatele de prețuri cu sau fără separator de mii.
2. Formatarea Prețului: Prețul extras este formatat pentru a asigura consistența datelor stocate.
3. Funcții de Web Scraping: Funcțiile get\_product\_info\_amazonshop și get\_product\_info folosesc Selenium pentru a naviga pe paginile produselor și pentru a extrage detaliile relevante, cum ar fi prețul, titlul și URL-ul imaginii.
4. Inserarea în Baza de Date: Informațiile despre produse sunt inserate în baza de date MySQL, gestionând atât adăugarea de produse noi, cât și actualizarea celor existente. Scriptul gestionează secvențele pentru identificatorii produselor și istoricul prețurilor.

#### Script pentru actualizarea prețurilor

Acest script se execută periodic pentru a verifica dacă prețurile produselor s-au schimbat și pentru a actualiza aceste prețuri în baza de date. Dacă un produs nu mai este disponibil, acesta este șters din baza de date. Scriptul include:

1. Actualizarea Prețurilor: Verifică periodic prețurile produselor și actualizează baza de date dacă sunt detectate schimbări.
2. Gestionarea Erorilor: În cazul în care prețul unui produs nu poate fi extras sau produsul nu mai este disponibil, scriptul șterge produsul respectiv din baza de date.
3. Notificări Backend: Scriptul trimite notificări backend-ului despre modificările de prețuri folosind un endpoint dedicat.

Script-ul utilizează biblioteca schedule, așa cum se observă în figura 3.13, pentru a fi rulat periodic, asigurând astfel că prețurile produselor sunt mereu actualizate. Acest mecanism automatizează procesul de monitorizare a prețurilor și notificarea utilizatorilor, garantând funcționarea eficientă a sistemului.

A screen shot of a computer

Description automatically generated

Figura 3.13 Programarea script-ului să ruleze odata la 5 minute

În figura 3.14 se observă cum se realizează conexiunea cu baza de date, pentru adăugarea sau actualizarea produselor.

A computer screen shot of a program code

Description automatically generated

Figura 3.14 Conexiunea cu baza de date

## Testare și validare

### Testarea front-end

Testarea manuală a componentei frontend s-a realizat pentru a testa funcționalitățile aplicației și pentru a asigura o experiență de utilizator fără probleme. Acest lucru a implicat testarea directă din browser, atât pe PC, cât și pe telefonul mobil, pentru a acoperi diferite scenarii de utilizare și pentru a asigura compatibilitatea și responsivitatea acestuia pe diferite dispozitive.

Testarea pe PC s-a realizat folosind browsere multiple, precum Chrome, Firefox și Edge pentru a testa consistența UI-ului și a funcționalitatea pe toate platformele mari. S-au realizat teste pentru toate funcționalitățile principale, cum ar fi adăugarea de produse la lista de favorite prin introducerea URL-urilor, setarea și actualizarea prețurilor de prag, și primirea notificărilor privind scăderea prețurilor.

În ceea ce privește testarea de pe un telefon mobil, s-au folosit telefoane diferite și cadre diferite pentru a testa responsabilitatea și funcționarea completă a UI-ului pe ecrane mai mici. S-au verificat toate aspectele aplicației, inclusiv navigarea prin meniuri, afișarea corectă a imaginii produselor și utilizarea interactivă a componentelor UI, precum butoanele și formularele de introducere a datelor.

Aceste teste manuale au asigurat că aplicația funcționează fără probleme pe toate dispozitivele și platformele mari, permițând unificarea în experiența perfectă a utilizatorului.

În ceea ce privește partea de comunicare cu server-ul, s-a testat comparând datele primite și afișate în aplicație cu datele existente în baza de date, dar la fel se verifică dacă datele trimise din aplicație spre server au fost salvate.

#### Validări existente în aplicație

1. Validare câmpuri la înregistrarea unui utilizator

A screenshot of a login form

Description automatically generated

Figura 3.15 Validare înregistrare

Pentru a testa modalitatea în care aplicația solicită anumite date folosind câmpurile pentru numele utilizatorului și adresa de email, a fost verificat faptul că aplicația nu permite înregistrarea cu date invalide. Astfel, pentru adresa de email, comparându-se email-ul introdus manual cu standardul de email, s-a constatat că eroarea este afișată corect de către sistem, acesta din urmă nepermițând înregistrarea utilizatorului.

Pentru câmpul de parolă, s-a testat că aplicația solicită respectarea cerințelor de securitate, precum o lungime minimă de 6 caractere, incluzând litere mari și mici, și cel puțin o cifră. În cazul unei greșeli, mesajele de corectare sunt afișate corect, sub câmpul respectiv, în echilibru cu denumirea sa pentru o experiență facilă de utilizare.

1. A screenshot of a web page

   Description automatically generatedValidare URL corect la adăugarea unui produs

Figura 3.16 Validare URL

În figura 3.16 , se poate observa pagina de adăugare a unui produs favorit în cadrul aplicației. Testarea acestei funcționalități a fost efectuată pentru a verifica corectitudinea validării URL-urilor introduse de utilizatori și a prețului de prag. Pentru URL-ul produsului, sistemul validează dacă acesta provine de la unul dintre site-urile acceptate (emag.ro sau amazonshop.ro). Dacă URL-ul introdus nu corespunde acestor criterii, utilizatorul primește un mesaj de eroare specific, indicând că doar URL-uri de la emag.ro sau amazonshop.ro sunt permise. Aceasta asigură că aplicația funcționează corect doar cu sursele de date autorizate.

### Testarea back-end

# Concluzii

## Rezultate obținute

Obiectivele proiectului de monitorizare a prețurilor au fost îndeplinite, oferind un sistem eficient și simplu de utilizat pentru urmărirea prețurilor pe platformele de comerț electronic. Sistemul a fost capabil să monitorizeze prețurile produselor de pe site-uri precum eMag și AmazonShop, utilizând tehnici avansate de web scraping. Scripturile Python implementate au permis colectarea și actualizarea datelor de preț în timp real, stocându-le într-o bază de date MySQL.

Sistemul permite utilizatorilor să adauge produse la lista de favorite prin simpla introducere a URL-ului acestora. Acest lucru simplifică procesul de urmărire a produselor dorite, permițând utilizatorilor să fie notificați în timp real atunci când prețurile scad sub pragurile stabilite de ei. Funcția de notificare a prețurilor a ajutat utilizatorii să economisească, informându-i rapid despre scăderile de prețuri ale produselor pe care le urmăresc.

Frontend-ul aplicației, care a fost construit folosind React, oferă o interfață de utilizator ușor de înțeles, cu stiluri bine definite și componente reutilizabile. Acest lucru asigură o navigare ușoară și o experiență de utilizare plăcută. Backend-ul aplicației, implementat în Spring Boot, a demonstrat performanță și scalabilitate, gestionând eficient cererile și răspunsurile și menținând un timp de răspuns scăzut chiar și la volume mari de date.

Măsurile de securitate implementate, inclusiv autentificarea și autorizarea, criptarea datelor și protecția împotriva vulnerabilităților comune, au asigurat integritatea și securitatea datelor utilizatorilor și ale produselor.

Proiectul a adus o serie de contribuții semnificative în domeniul monitorizării prețurilor. Automatizarea procesului de colectare și actualizare a datelor despre produse este o contribuție semnificativă, deoarece reduce timpul și efortul necesar utilizatorilor în urmărirea prețurilor. Utilizarea scripturilor Python pentru scraping web a permis o colectare rapidă și eficientă a datelor despre produse de pe o varietate de site-uri de comerț electronic, garantând informații precise și actuale.

O altă contribuție semnificativă a fost implementarea unei arhitecturi modulare atât în frontend, cât și în backend. Această abordare a facilitat dezvoltarea și întreținerea aplicației, permițând adăugarea ușoară de noi funcționalități și scalarea sistemului în funcție de necesități. Utilizarea React pentru dezvoltarea frontend-ului a permis crearea unei interfețe de utilizator moderne și responsive, oferind o experiență plăcută utilizatorilor. Pe de altă parte, Spring Boot a oferit un mediu robust și scalabil pentru dezvoltarea backend-ului, asigurând gestionarea eficientă a cererilor și răspunsurilor.

În ceea ce privește securitatea, proiectul a integrat mecanisme avansate de autentificare și autorizare, precum și criptarea datelor sensibile, asigurând protecția informațiilor utilizatorilor. Implementarea notificărilor în timp real pentru scăderile de preț a oferit utilizatorilor posibilitatea de a reacționa rapid la oportunitățile de economisire, îmbunătățind astfel experiența de cumpărare online.

Proiectul a demonstrat, de asemenea, importanța utilizării unor tehnologii moderne și eficiente în dezvoltarea de aplicații web complexe, contribuind astfel la creșterea calității și performanței soluțiilor software în domeniul comerțului electronic.

## Direcții de dezvoltare

Există multiple direcții de dezvoltare pentru această aplicație care ar putea îmbunătăți și extinde funcționalitatea sa. Extinderea capabilităților de web scraping pentru a include mai multe site-uri de comerț electronic ar permite utilizatorilor să monitorizeze o gamă mai largă de produse. Dezvoltarea unei aplicații mobile ar oferi utilizatorilor acces mai facil la serviciile de monitorizare a prețurilor, oriunde s-ar afla.

Implementarea unor opțiuni avansate de notificare, cum ar fi notificările push și alertele personalizate, ar îmbunătăți considerabil experiența utilizatorilor. Adăugarea unor funcționalități de analiză și raportare ar oferi utilizatorilor insight-uri detaliate despre tendințele prețurilor și istoricul achizițiilor lor, facilitând decizii de cumpărare mai informate.

Integrarea algoritmilor de machine learning pentru a prezice modificările viitoare ale prețurilor ar oferi utilizatorilor un avantaj suplimentar în planificarea achizițiilor. În plus, optimizarea aplicației pentru motoarele de căutare (SEO) ar atrage mai mulți utilizatori și ar crește vizibilitatea pe piața online.

În concluzie, sistemul de monitorizare a prețurilor a demonstrat eficiența și viabilitatea unui sistem automatizat pentru gestionarea prețurilor produselor, oferind utilizatorilor un instrument valoros pentru economisirea timpului și banilor. Direcțiile de dezvoltare propuse vor contribui la îmbunătățirea și extinderea funcționalităților aplicației, consolidându-i poziția pe piața de comerț electronic.

# Bibliografie

|  |  |
| --- | --- |
| [1] | Clear The Shelf, „CamelCamelCamel Review - is this Amazon price tracking tool worth it?,” n.d.. [Interactiv]. Available: https://www.cleartheshelf.com. [Accesat 29 May 2024]. |
| [2] | Techlicious, „The Best Amazon Price History Tracker,” n.d.. [Interactiv]. Available: https://www.techlicious.com/tip/the-best-amazon-price-history-tracker/. [Accesat 29 May 2024]. |
| [3] | FBA Master, „Camel Camel Camel Review: The Only Amazon Price Tracker You Need,” n.d.. [Interactiv]. Available: https://www.fbamaster.com. [Accesat 29 May 2024]. |
| [4] | M. Smith A. & Anderson, „Online Shopping and E-Commerce,” 2016. |
| [5] | B. Jansen și S. Y. Rieh, „The Seventeen Theoretical Constructs of Information Searching and Information Retrieval,” *Journal of the American Society for Information Science and Technology,* 2010. |
| [6] | R. Mitchell, Web Scraping with Python, vol. 51, O'Reilly Media, 2015, pp. 30-44. |
| [7] | A. Sweigart, Automate the boring stuff with Python: Practical programming for total beginners, No Starch Press, 2015. |
| [8] | D. Glez-Peña, A. Lourenço, H. López-Fernández și M. &. F.-R. F. Reboiro-Jato, „Web scraping technologies in an API world,” *Briefings in Bioinformatics,* 2014. |
| [9] | R. Mitchell, Web Scraping with Python: Collecting Data from the Modern Web (2nd ed.), O'Reilly Media, 2018. |
| [10] | E. Ferrara, P. De Meo, G. Fiumara și R. Gaumgartner, „Web data extraction, applications and techniques: A survey,” *Knowledge-Based Systems,* vol. 70, pp. 301-323, 2014. |
| [11] | Commerce Crafted, „CamelCamelCamel Review: Is It Worth Your Time?,” n.d.. [Interactiv]. Available: https://www.commercecrafted.com. [Accesat 29 May 2024]. |
| [12] | S. Krug, Don't Make Me Think: A Common Sense Approach to Web Usability, New Riders, 2014. |
| [13] | J. Johnson, Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Guidelines., Morgan Kaufmann, 2014. |
| [14] | J. Tidwell, C. Brewer și A. Valencia, Designing Interfaces: Patterns for Effective Interaction Design, O'Reilly Media, 2020. |
| [15] | B. Schneier, Data and Goliath: The Hidden Battles to Collect Your Data and Control Your World, W.W. Norton & Company, 2015. |
| [16] | A. Monica, R. Lee, A. Brookie, P. Andrew , K. Madhu și T. Erica, „Americans and Privacy: Concerned, Confused and Feeling Lack of Control Over Their Personal Information,” Pew Research Center, 2019. |
| [17] | W. Stallings, Cryptography and Network Security: Principles and Practice, Pearson, 2017. |
| [18] | K. J.F. și R. K.W., Computer Networking: A Top-Down Approach, Pearson, 2017. |
| [19] | K. Scarfone și P. Mell, „Guide to Intrusion Detection and Prevention Systems,” *NIST Special Publication,* 2007. |
| [20] | Python Software Foundation, „About Python,” n.d.. [Interactiv]. Available: https://www.python.org/doc/essays/blurb/. [Accesat 31 May 2024]. |
| [21] | Real Python, „What is Python?,” n.d.. [Interactiv]. Available: https://realpython.com/what-is-python/. [Accesat 31 May 2024]. |
| [22] | SeleniumHQ, „Selenium Documentation,” n.d.. [Interactiv]. Available: https://www.selenium.dev/documentation/en/. [Accesat 31 May 2024]. |
| [23] | Real Python, „Getting Started with Selenium for Automated Testing in Python,” n.d.. [Interactiv]. Available: https://realpython.com/selenium-python/. [Accesat 31 May 2024]. |
| [24] | Spring, „Spring Boot Reference Documentation,” n.d.. [Interactiv]. Available: https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/. [Accesat 31 May 2024]. |
| [25] | Baeldung, „Introduction to Spring Boot,” n.d.. [Interactiv]. Available: https://www.baeldung.com/spring-boot. [Accesat 31 May 2024]. |
| [26] | DZone, „What is Spring Boot,” n.d.. [Interactiv]. Available: https://dzone.com/articles/what-is-spring-boot. [Accesat 31 May 2024]. |
| [27] | React, „Getting Started – React,” n.d.. [Interactiv]. Available: https://reactjs.org/docs/getting-started.html. [Accesat 31 May 2024]. |
| [28] | FreeCodeCamp, „An Introduction to React,” n.d.. [Interactiv]. Available: https://www.freecodecamp.org/news/an-introduction-to-react-js/. [Accesat 31 May 2024]. |
| [29] | Smashing Magazine, „Virtual DOM and Internals,” n.d.. [Interactiv]. Available: https://www.smashingmagazine.com/2020/02/virtual-dom-react/. [Accesat 31 May 2024]. |
| [30] | P. DuBois, MySQL, Addison-Wesley Professional, 2005. |
| [31] | S. Baron, . Z. Peter și T. Vadim, High Performance MySQL: Optimization, Backups, Replication, and More, O'Reilly Media, 2012. |
| [32] | Oracle Corporation, „MySQL Documentation,” n.d.. [Interactiv]. Available: https://dev.mysql.com/doc/. [Accesat 02 June 2024]. |
| [33] | J. M., B. J. și S. N., „JSON Web Token (JWT),” *IETF,* 2015. |
| [34] | H. D., „The OAuth 2.0 Authorization Framework,” *IETF,* 2012. |
| [35] | J. Wayne și G. Timothy, „Guidelines on Security and Privacy in Public Cloud Computing,” *NIST Special Publication.* |
| [36] | F. Martin, Patterns of Enterprise Application Architecture, Addison-Wesley Professional, 2002. |
| [37] | H. Kevlin, B. Frank și S. Douglas C., Pattern-Oriented Software Architecture Volume 4: A Pattern Language for Distributed Computing, Wiley, 2007. |
| [38] | G. Erich, H. Richard, J. Ralph și V. John, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1994. |
| [39] | D. Herron, Node.js Web Development - Fourth Edition: Server-side development with Node 10 made easy 4th ed. Edition, Packt Publishing, 2018. |
| [40] | A. Osmani, Learning JavaScript Design Patterns, O'Reilly Media, 2010. |